Universidad del Bío-Bío
Facultad de ciencias empresariales
Departamento de sistemas de información

UNIVERSIDAD DEL BÍO-BÍO
FACULTAD DE CIENCIAS EMPRESARIALES

# Modelación del rostro humano con mallas de superficie o triangulaciones 3D para la construcción de retratos hablados.

---

Memoria para optar al título de Ingeniero de Ejecución en Computación e Informática

Alumnos: Antonio Montecinos Osorio
Felipe Rubio Neira

Profesor guía: Dr. Pedro A. Rodríguez M.

Fecha: 17 de Agosto de 2018

# Agradecimientos

Este proyecto de título fue respaldado por el grupo de investigación Computación Ubicua, código GI 150115/EF.

# Resumen

Este proyecto se presenta para dar conformidad a los requisitos exigidos por la universidad del Bío-Bío en el proceso de titulación para la carrera de Ingeniería de Ejecución en Computación e Informática.

El proyecto titulado “Modelación del rostro con mallas de superficie para la construcción de retratos hablados” se desarrolla dentro del marco de la universidad del Bío-Bío. Dicho proyecto tiene por objetivo disminuir la cantidad de recursos que son utilizados para la realización un retrato hablado, por diferentes instituciones a cargo la búsqueda criminal y forense. Perimiendo agilizar, mediante el uso de un software de visualización y modelado 3d, muchos aspectos del desarrollo y construcción de un retrato hablado.

El principal beneficio que otorga la utilización de esta aplicación es la reducción considerable en el tiempo que toma la construcción de un retrato hablado, añadiendo el factor tridimensional e interactivo. Permitiendo además que se pueda disponer de manera rápida un rostro que puede ser editado y almacenado; dando como resultado una aplicación efectiva y competente.

Para la realización de esta aplicación se utiliza la metodología de desarrollo ágil extreme programming, la cual facilita el desarrollo de requerimientos esenciales de manera rápida y en colaboración constante con cliente. El software fue elaborado mediante el lenguaje de programación Java, el cual permite su utilización en diversas plataformas y medios, no obstante, la implementación de esta aplicación es concebida para ser operada de forma local como aplicación de escritorio.

# Índice

# 1.- introducción

El presente documento pretende dar al lector una vista amplia del desarrollo de una herramienta computacional que permite mediante la utilización de mallas de superficie construir un rostro en tres dimensiones, en donde mediante la modificación del mismo podemos obtener un conjunto de rasgos y características que permiten distinguir a una persona y de esta manera elaborar un retrato hablado. Este tipo de herramientas son fuertemente requeridas por instituciones que están ligadas al área criminalística o forense, es decir, que buscan obtener, a través de descripciones orales la representación del rostro de una persona que ha sido extraviada o que ha cometido algún delito. De esta manera conseguimos aportar a los recursos con los que cuentan dichas instituciones, a fin de ser una herramienta capaz de acelerar la construcción de un retrato hablado y también almacenar posteriormente la información que entrega dicho retrato para su utilización en otros contextos.

En la primera parte del documento se describe de forma espacio temporal la utilización de los retratos hablados como medio de identificación, y cómo estos comienzan de forma muy sencilla hasta que, con el paso de los años, junto con el desarrollo industrial y tecnológico, llegan a complejizarse de tal manera que hoy existen profesionales que requieren años de experiencia para poder llegar a participar del proceso de construcción de un retrato hablado. Junto con ello se declara la historia y los autores más influyentes que han contribuido en el desarrollo teórico y empírico de las mallas de superficie, las cuales son esenciales en el área de la ciencia, la investigación y también el entretenimiento.

Posteriormente se detalla la manera en la que estas dos disciplinas pueden unirse para contribuir al desarrollo de una aplicación que permita la elaboración de un retrato hablado mediante la utilización de mallas de superficie. Se profundiza en cuanto a los elementos principales que poseen las mallas, así como también los

factores más importantes que permiten la diferenciación entre las personas, desde el punto de vista antropológico, así como también los elementos que deben ser tomados en cuenta para realizar un proceso de abstracción que permitirá plasmar en un software cada uno de estos detalles.

A continuación, se detalla el proceso de análisis, diseño y alcances de la aplicación, el cual permite dar una estructura a la misma e identificar cada uno de los componentes que harán que el sistema desempeñe correctamente su trabajo, así como las limitaciones que estarán incluidas dentro del mismo. De igual manera se especifica cómo es que estos componentes van a interactuar entre sí para intercambiar datos e información.

Finalmente, se expone la construcción de la herramienta computacional de retratos hablados
a través de la implementación del lenguaje Java como base para la elaboración de dicha aplicación Se detallan los elementos más importantes para su construcción tales como clases, métodos y unidades de software que permiten enriquecer la misma, así como también la integración de un algoritmo de refinamiento localizado que permite aumentar la cantidad de puntos dentro de la malla y lograr una mayor precisión en la manipulación de la superficie en la que se está trabajando.

# 2.- Definición del proyecto

## 2.1.- objetivos del proyecto

### 2.1.1 Objetivo general

Elaborar un software para la visualización de una cabeza humana para la creación de un retrato hablado, a partir de un rostro base generado usando una malla de superficie, en un entorno 3D.

### 2.1.2 Objetivos Específicos

1-Investigar sobre la utilización de mallas de superficie para la visualización y el modelado de diferentes objetos en tres dimensiones.

2-Desarrollar un software que permita modificar un rostro en tres dimensiones cuya información está almacenada en un archivo de texto plano.

3-Implementar una interfaz gráfica que admita la interacción con el usuario mediante el uso del ratón y el teclado.

4-Incorporar un algoritmo de refinamiento capaz de realizar un refinamiento local sobre la malla 3D que permita aumentar el nivel de detalle de la misma.

## 2.2.- Justificación del proyecto

El proyecto tiene como propósito ayudar en las investigaciones realizadas por diversas instituciones a fin de ser una herramienta en la búsqueda y visualización de rostros, así como también en las investigaciones policiales. Permitiendo la construcción de rostros en tres dimensiones a partir de la edición de archivos base cargados con puntos y triángulos, disminuyendo de manera considerable el tiempo que demora modelar un rostro humano de forma manual. Además, tiene aplicaciones en el área criminalística, ciencias forenses y antropología, recreando

rostros de individuos para ayudar a identificar una víctima desconocida, reconstrucción de rostros de personajes históricos, antepasados humanos, etc.

Cabe destacar que en la actualidad no existe una herramienta que esté completamente orientada a la construcción de retratos hablados en tres dimensiones, ya que solo se enfocan en los aspectos más básicos y elementales de los mismos, dejando fuera muchas de las necesidades de los clientes, además de que no se da prioridad al refinamiento localizado el cual permite trabajar de mejor manera ciertas zonas del rostro que desean ser esculpidas con mayor detalle. Por otro lado, herramientas que se enfocan en la construcción de retratos en dos dimensiones solo cuentan con características preestablecidas que pueden llegar a limitar la construcción de los mismos.

Por otra parte, este proyecto puede ayudar en el área médica, principalmente en el plano de la cirugía ya sea curativa o estética, que busca la corrección o mejoramiento de anormalidades de origen congénito y/o adquirido, permitiendo una visualización previa de lo que se quiere lograr sin ser necesario intervenir al paciente. Cabe destacar que en la actualidad este tipo de intervenciones son mucho más comunes que antes, por ende, resulta una herramienta muy importante que permitirá tanto a cirujanos como doctores recrear rostros de personas que han sido afectadas por algún accidente o aquellas que quieran mejorar estéticamente su aspecto, permitiendo reconstruir el rostro de la persona, modificar específicamente alguna parte de la cara, visualizar un antes y un después, etc.

## 2.3 Ambiente de ingeniería de software

- ### Metodología de desarrollo

  En el proceso de desarrollo se utilizó la metodología **extreme programming** (xp), de esta forma se realizan avances rápidos en cuanto a las principales funcionalidades del software, demás se efectúan reuniones semanales con el profesor guía para exponer avances.

Se pueden añadir continuamente y de forma paralela nuevos requerimientos que permitirán enriquecer tanto el diseño de la interfaz de usuario como la experiencia del mismo a la hora de utilizar el software, generando a través de estas reuniones retroalimentación de parte del profesor guía hacia los alumnos y viceversa.

- Herramientas de desarrollo de software

  NetBeans IDE 8.2

  Microsoft Word 2016

- Hardware para el desarrollo de Software

  Se trabajó con un ordenador que cuenta con un procesador Intel Core i5 de séptima generación, 8 Gb de Ram y 500 Gb libres para almacenamiento en disco.

- Lenguaje de programación

  Java Version 8 Update 181

## 2.4.- Definiciones siglas y abreviaciones

**Java:** Lenguaje de programación perteneciente a la compañía Oracle, el cual permite codificar aplicaciones multiplataforma que pueden funcionar de manera independientes en un ordenador o también en el ambiente web.

**IDE:** Siglas de entorno de desarrollo integrado. Corresponde a una aplicación informática que cuenta con una interfaz gráfica que proporciona todas las herramientas para el desarrollo ágil de proyectos. Posee un editor de código, un compilador y un depurador como elementos esenciales.

**Modulo:** Unidad de software capaz de ejecutar una funcionalidad especifica o un conjunto de ellas.

**Malla:** Estructura de datos que es utilizada para discretica objetos de la vida real en pequeñas porciones sobre las cuales se pueden aplicar diversos procedimientos o cálculos matemáticos

**Extreme Programming:** Estilo de desarrollo que permite una rápida codificación de los aspectos básicos y más fundamentales de un software. Permite que sean aceptados cambios en cualquier fase del desarrollo de la aplicación y se mantiene comunicación constante con el cliente.

**UML:** Siglas de Lenguaje de modelado unificado. Consiste en un sistema de representación visual de la estructura de una aplicación a través de diagramas, lo cual permite aumentar la comprensión del funcionamiento de la aplicación para usuarios que no están familiarizados con lenguajes de programación.

**Obj:** También conocido como Wavefront 3D Object File, es un tipo de extensión para archivos de texto plano, desarrollado por Wavefront 3D Object File, que es usado para la representación de objetos tridimensionales que contiene coordenadas 3D. Se incluyen líneas poligonales, puntos y mapas de texturas.

**M2D**: Formato para archivos de texto plano que contiene información de vértices distribuidos en el espacio, así como también de triángulos que se encuentran representados como arreglos de vértices, también incluye información de mapas de texturas. Finalmente se agrega información de las vecindades que son compartidas por cada triangulo.

**3D:** Técnica de representación gráfica que intenta simular tres dimensiones dentro del contexto de la resolución de un ordenador, limitado por el ancho y el alto.

**Binding:** Método que permite vincular en tiempo de ejecución una variable a otra, permitiendo que la modificación de una resulte en la alteración de otra.

# 3.- Especificaciones de requerimientos de software

## 3.1.- Alcances

La aplicación de retratos hablados permite a los usuarios importar archivos en formato obj o m2d, visualizar un rostro en tres dimensiones, ya sea realizando rotaciones de cámara o zoom mediante el uso del ratón, así como también realizar modificaciones en cualquier zona de la cara que se estime conveniente. El sistema además permite la exportación de archivos obj al formato m2d, el cual puede ser almacenado en el disco duro para futuras ediciones.

El sistema cuenta con una unidad de software de refinamiento de malla, el cual incrementar el número de vértices y triángulos, permitiendo mejorar la forma de la misma, otorgándole mayor realismo y un acabado más natural.

Cabe mencionar que el sistema excluye las ediciones de zonas que no tengan piel tales como los ojos y cabello, así como también la incorporación de tatuajes, bigote, barba, arrugas u otros elementos adicionales.

## 3.2.- Objetivos del software

### 3.2.1.- Objetivo general

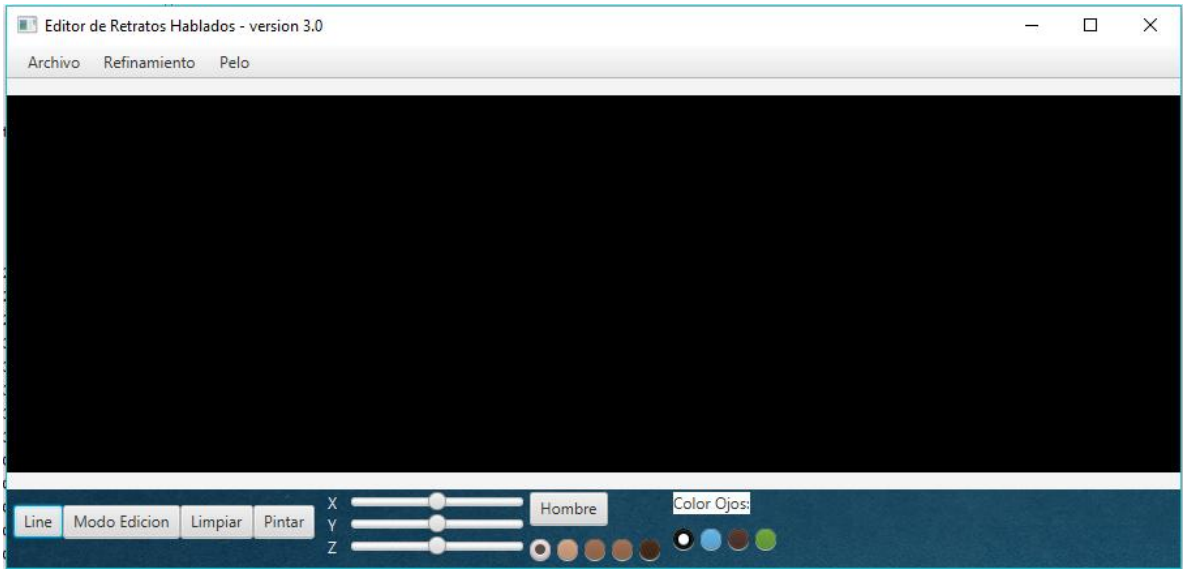
Permitir la realización de un retrato hablado mediante la edición y modelado de un rostro que es visualizado en un ambiente 3D, a partir de un modelo almacenado en un archivo de texto plano y posteriormente realizar un refinamiento de cualquier zona que se desee del rostro.

### 3.2.2.- Objetivos específicos

1) Obtener una nube de puntos que represente el rostro humano para su posterior procesamiento y edición.
2) Obtener una triangulación de la nube de puntos para representar cada polígono como un arreglo de vértices.
3) Generar una malla de superficie en tres dimensiones del rostro humano para posteriormente visualizarla de forma dinámica.
4) Obtener coordenadas de texturas para simular una piel realista que cubra toda la superficie de la malla.
5) Manipular los vértices de la malla para modificar cualquier parte del rostro que se desee.
6) Realizar un refinamiento en cualquier zona de la malla para obtener un acabado más realista.

## 3.3.- Descripción global del producto

### 3.3.1.- Interfaz de usuario




El software se despliega a través de una ventana en la que se tiene acceso a diferentes funcionalidades. Podemos distinguir 3 secciones: la primera es la barra de menú, la cual nos proporciona diferentes opciones como cargar y exportar archivos, realizar refinamiento de la malla y agregar cabello en el caso que sea necesario.

En segundo lugar, encontramos al centro del software el espacio principal de trabajo el cual nos permite desplazarnos y realizar diferentes gestos por medio del mouse y el teclado. Es la zona central es donde se llevarán a cabo las principales funcionalidades ya sea de visualización o edición que se hagan al retrato.

Finalmente encontramos la zona inferior la cual cuenta con variados elementos tales como botones, etiquetas, radio botones y controles desplazables. Cada uno de estos nos permitirá modificar de alguna manera el retrato que estemos construyendo, así como también limpiar el área en la que estemos trabajando para comenzar a editar un nuevo retrato.

### 3.3.2.- Interfaz de hardware

El software no interactúa con hardware específico.

### 3.3.3.- Interfaz de software

El sistema es autónomo y no tiene conexión con software externo.

# 4.- Factibilidad

## 4.1.- Factibilidad técnica

Para llevar a cabo este proyecto se requiere de un ordenador que cuente con la versión 8 de java, este elemento es básico para compilar el código fuente que será desarrollado posteriormente en el entorno de desarrollo integrado NetBeans. Por otra parte, se requiere a lo menos un programador que desarrolle la aplicación. Para este proyecto se tiene contemplado los dos alumnos tesistas. Finalmente se requiere una conexión a internet que permita realizar búsqueda de información y fuentes.

A continuación, se procede a detallar las características con las que cuenta el ambiente de desarrollo.

### 4.1.1Hardware

El sistema será desarrollado en un ordenador portátil.

Marca: HP

Procesador: Intel Core i5 2.71Ghz

Ram: 8Gb

Sistema Operativo: Microsoft Windows 10 Home Single Languaje

Para el desarrollo de la aplicación no son requeridos elementos de hardware adicionales

### 4.1.2.- Software

Nombre: NetBeans

Versión: 8.2

Tipo licencia: Libre

## 4.2.- Factibilidad Operativa

El sistema está diseñado para usuarios que estén familiarizados con el uso del ratón y el teclado, no se requieren grandes conocimientos de computación, sin embargo, se deben conocer acciones básicas de todo software como el menú de barra de navegación, así como también el uso de botones dentro del contexto del software. Junto con ello, se debe tener nociones básicas de los gestos con el ratón tales como clicar, arrastrar y soltar.

La mayoría de los programas informáticos que existen hoy en día cuentan con estas acciones y por este motivo puede resultar muy cómodo para aquellos usuarios que se inician dentro de la utilización del software, así como también otros usuarios más avanzados que conocen los atajos de teclado o “shortcuts” universales para abrir o guardar archivos.

## 4.3.- Factibilidad económica

Para este proyecto la utilización de recursos económicos es cero ya que al ser un proyecto de pregrado en donde se utiliza software de licencia libre y se cuentan con recursos computacionales hardware propios no se requiere incurrir en gastos adicionales. Además, el desarrollo de la aplicación es realizado por alumnos memoristas lo cual implica un ahorro en la mano de obra. Sin embargo, si se consideran todas estas variables el proyecto podría alcanzar un valor de aproximadamente $4.000.000.

## 4.4.- Conclusión de la factibilidad

Luego de realizar el estudio de factibilidad en donde se incluye las principales variables tales como el financiamiento, desarrollo y puesta en marcha, podemos concluir que el proyecto es complemente factible de realizar en un periodo de 6 meses.

- Desde la perspectiva de la factibilidad operativa no existe ningún impedimento para que tanto personas que recién se están iniciando en la

computación como usuarios avanzados puedan utilizar el software y probar las funcionalidades que tiene, sin miedo a frustrarse, ya que la curva de aprendizaje es muy rápida y con tan solo unas horas de práctica, cualquier usuario puede realizar la construcción de un retrato hablado.

- Desde el punto de vista económico se contemplan elementos como la mano de obra, el uso de recursos computacionales hardware y software. Esto nos permite excluir el coste que posee el proyecto si es realizado fuera del contexto universitario. De esta manera se puede considerar un ahorro significativo tanto para los alumnos resistas como para el proyecto en sí mismo.
- Finalmente, desde el enfoque de la factibilidad técnica es totalmente realizable, puesto que se cuenta con los recursos humanos, los conocimientos y las fuentes necesarias para la investigación de los puntos que así lo requieran. Además, se dispone del material necesario para la ejecución satisfactoria del proyecto.

Con todo lo mencionado anteriormente, se puede concluir que este proyecto puede llevarse a cabo pues posee las cualidades necesarias para su financiamiento, desarrollo y posterior puesta en marcha.

# 5.- Estado del Arte

Los retratos hablados datan desde el viejo Oeste en EE. UU, desde el siglo XIX, en donde los sheriffs de las distintas localidades hacían uso de retratos hablados para identificar a los criminales que amenazaban a las personas en esa época. Creaban carteles con el rostro del criminal junto a su nombre y además una recompensa por su cabeza.

Existen dos métodos para la realización de un retrato hablado, a mano alzada o a través de la utilización de un software, ambos métodos consiguen altos porcentajes de éxito. Sin embargo, una de las primeras técnicas o sistemas que precedieron la utilización de softwares es el Photofit, que fue presentado por el policía inglés Jackes Penry en la década de los 60. Este sistema estaba compuesto por un conjunto de fotos tomadas de diversas personas con distintas características faciales, y consistía en recortar cinco de los principales componentes faciales: los ojos, la nariz, la boca, la barbilla y el peinado. Estas imágenes se juntaban para conformar la cara que el testigo relataba a los oficiales. Lamentablemente, esta técnica era bastante limitada, pues solo era factible en ese país y época. Otro sistema bastante parecido al de Jackes fue el Identikit, del norteamericano Hugh McDonald que pertenecía al departamento de policía de Los Ángeles. El sistema consistía en una serie de transparencias que graficaban los rasgos corporales más comunes de las personas de esa ciudad. Después de obtener un rostro base, el dibujante procedía a sumarle características propias que aportaba el testigo.

Como ya se mencionó, la segunda línea de trabajo en los retratos hablados reside en la era digital, pues los softwares computacionales tomarían la delantera en retratos hablados. De esta forma, uno de los primeros software en aparecer es el Faces, creado por IQ Biometrix. Este programa lleva más de 12 años siendo utilizado por la CIA, FBI e incluso el ejército de los EE.UU probando su fiabilidad, pues con él han logrado rastrear y detener a numerosos delincuentes. Así, Faces hace uso de su base de datos que posee, aproximadamente, 4.400 rasgos faciales

distintos, entre los que se cuentan el color del cabello, el tamaño de las orejas, distintos tamaños y colores de ojos, boca, etc. Además, posee la característica de distinguir entre grupos étnicos como el americano, africano y asiático, que, a su vez, pueden ser creadas diferenciando el género. Este programa puede ser utilizado en cualquier equipo computacional. Asimismo, existen programas que son fabricados para su país de origen, pues poseen características faciales de esa cultura. CaraMex es un software mexicano que lanza su primera versión en 1996 y la segunda versión en 2002. Al igual que Faces, posee un alta gamma de características faciales, pero con la particularidad de que estos rasgos son asociados a la cultura étnica mexicana. Entre las características que posee se destacan: arrugas, barbas, cejas, ojos, labios, bigotes y accesorios como lentes, aros, gorros, etc. Hasta hoy, Caramex y Faces siguen siendo utilizados por la policía mexicana y estadounidense, aun así, estos softwares son utilizados como apoyo al clásico retrato hablado.

Las desventajas que poseen los programas mencionados, es el uso de presets, que en este caso son imágenes almacenadas en una base de datos con diseños predefinidos de las distintas características que posee el rostro. Los creadores de Caramex se dedicaron durante 3 años a tomar fotografías a más de 5,000 mexicanos, de frente y sus dos perfiles, así lograron obtener las características faciales más comunes del pueblo de México. Los presets se van montando en un rostro predefinido, para lograr un gran parecido a la persona que se va a retratar. El problema principal de esta metodología, son las limitaciones que presenta al momento de querer hacer una modificación del rostro terminado. Esta dificultad se puede solucionar haciendo uso de mallas de superficie, que permiten un alto grado de flexibilidad al momento de querer modificar zonas específicas del rostro.

Hoy en día existen programas que hacen uso de mallas de superficie, que son utilizadas en distintas áreas de las ciencias e ingeniería. Uno de estos softwares es AutoCad, que existe desde el año 1982. Es un programa utilizado en computadores, en el que se pueden realizar dibujos en 2 y 3 dimensiones. Esta herramienta es utilizada para modelar planos de edificios o para la recreación de imágenes en 3D,

además es capaz de hacer uso de mallas de superficie a las que permite hacerles distintas modificaciones de acuerdo al elemento que se quiere modelar. Otro software que hace uso de estas mallas es FaceGen, que fue creado en 1998 y que sigue en vigencia. Este programa hace uso de una malla de superficie compuesta por cuadriláteros y es utilizado para la creación automática de rostros humanos en 3D, ha sido utilizado por compañías como Electronics Arts, Sony, Microsoft y Sega. Su principal función es generar rostros que pueden ser exportados en distintos formatos para ser utilizados como Avatares de juegos online, para softwares de simulación, para evaluar algoritmos de reconocimiento facial, etc. Por otro lado, FaceGen no permite realizar refinamientos de manera local, pues el algoritmo que utiliza, solo permite realizar un refinamiento de manera global a su malla de cuadriláteros generando demasiado puntos en zonas de poca relevancia.

En chile, el Os9 de Carabineros hace uso del retrato hablado para lograr identificar a criminales, asesinos o gente extraviada. Siguen más familiarizados con el dibujo a mano alzada, pues afirman que de esta manera se logra mayor empatía con el testigo que realiza las declaraciones. Además, utilizan imágenes que apoyan al dibujante para lograr mayor exactitud en los detalles que quiere obtener de la persona a retratar.

En definitiva, en la actualidad, no existe un software que cumpla con los requisitos del área criminalística para obtener un retrato hablado con la precisión y exactitud que se puede lograr por medio de la técnica de dibujo a mano alzada. Si bien existen softwares que dan soporte a los dibujantes, no existen mayores avances en estos programas, pues en cuanto a detalles de los rasgos faciales más destacados en las personas, solo se hace uso de imágenes o cara predefinidas (presets) que logran un parecido al retratado. Además, no logran concretar una mayor empatía con el testigo, puesto que hay una cierta inseguridad al momento trabajar con programas computacionales.

# 6.- Marco Teórico

Los retratos hablados hoy en día son los más utilizados en el área de la criminalística, pero existe una alta crítica a esta técnica, ya que el resultado va depender de la veracidad de la descripción del testigo. Los retratos hablados se utilizan como guía para identificar a una persona y las probabilidades de éxito por lo general son muy bajos, pues realizar una versión detallada del rostro de una persona requiere de mucha precisión. De esto surge una posibilidad para lograr aumentar esa tasa de éxito por medio de tecnologías que colaboren con el testigo a dar una mayor precisión en los rasgos faciales de la persona que se quiere identificar. El software que proponemos hace uso de mallas de superficie compuesta de triángulos, pues, nos brinda la posibilidad de generar una herramienta que pueda aportar a obtener un retrato más detallado, además el testigo podrá guiarse por medio de un programa que es amigable con el usuario y que posee un rostro virtual que puede modificarse en tiempo real.

Las mallas de superficie están descritas en Boris Vintimilla (2000) y nos dice que son un conjunto de puntos o vértices y segmentos que unidos entre sí configuran algún tipo de elemento geométrico como triángulos, cuadriláteros, polígonos, etc. Las mallas poseen la particularidad de ser representadas en espacios 2D y 3D. Existe dos modelos de mallas: Estructuradas y no estructuradas, una de las principales diferencias es que en una malla estructurada la cantidad de puntos adyacentes a un vértice es la misma para todos a excepción de los bordes, en cambio en una malla no estructurada se permite cualquier número de puntos adyacentes. Las mallas de superficie permiten representar la superficie de un objeto como un conjunto de polígono adyacentes. Como se explicaba, existen distintos tipos de malla que pueden estar compuestas de triángulos, cuadrilátero o polígonos. Para lograr conformar la malla que usaremos, se debe hacer uso de un proceso llamado triangulación. Tal proceso se puede realizar en 2 y 3 dimensiones, en el que, por medio de una nube de puntos, que posee vértices o puntos distribuidos en el plano, se crea un conjunto de triángulos unidos entre si con algún criterio de

optimización, estos criterios poseen la cualidad de crear triángulos con propiedades únicas.

Para nuestro propósito, haremos uso de una malla no estructurada compuesta por triángulos, pues, posee la cualidad de ser más flexible para adaptarse a dominios complicados como lo es un rostro humano. Además, haremos uso del algoritmo de refinamiento Lepp-Bisection (Rivara, 2013), que hace uso de triángulos para lograr un refinamiento de la malla inicial y así lograr más detalles en ciertas zonas del rostro que deseamos modelar.

Los algoritmos de refinamiento de mallas, permiten insertar nuevos vértices y elementos que son anidados a los ya existentes, y consigue que la calidad de la malla inicial se mantenga. La necesidad del refinamiento de generar más vértices dentro de los elementos que componen una malla, nos permite pasar de lo que llamaremos una malla gruesa, la que posee muy pocos elementos iniciales a una malla fina, con mayor cantidad de elementos y que a su vez son más pequeños. El algoritmo de refinamiento Lepp-Bisection, permite realizar inserción de nuevos vértices a través del concepto de la arista más larga, en el que se produce una bisección del triángulo a través del punto medio de su arista más larga y esta a su vez se puede propagar usando el concepto de Lepp (Longest Edge propagation path) al triángulo vecino con el que comparte tal arista.

# 7.- Estructura, componentes y proceso de creación del rostro en 3D

## 7.1.- Geometría Computacional

La geometría computacional corresponde a un área de investigación que se desprende de la geometría clásica y que añade conceptos de la teoría del diseño y análisis de algoritmos y estructuras de datos, correspondientes a la rama de la ciencia de la computación. Esto, ya que se interesa en dar solución a problemas encontrando algoritmos y estructuras de datos eficientes, expresados en términos geométricos; teniendo suma importancia la complejidad temporal y espacial de dichos algoritmos.

La geométrica computacional se define como el estudio de problemas algorítmicos que involucran geometría (Ralston, 2000).

Otro concepto que se asocia a la geometría computacional, y que está estrechamente relacionado, es la computación gráfica. Esta permite a través de algoritmos generar gráficos que se fundamentan en cálculos matemáticos y que permiten generar figuras como triángulos, cuadrados, cubos, esferas, y también formas irregulares. Esta nace de la mano del crecimiento y desarrollo del hardware que comienza ya a mediados de los años 60 y que ha seguido avanzando y perfeccionándose junto a diversas técnicas algorítmicas para conseguir resultados eficientes y con un buen rendimiento.

Ya sea en el entorno 2D o 3D, hoy en día contamos con diversas herramientas que permiten la manipulación de datos y la visualización de los mismos. Entre ellas podemos señalar “Google heart “o “AutoCad” que basan su funcionamiento en diferentes elementos gráficos con los que podemos interactuar. A su vez existe una capa abstracta en la que se procesan datos obtenidos de dichos

gráficos y se emplean cálculos matemáticos para realizar diferentes procedimientos que entregan posteriormente una salida.

Encontramos diferentes áreas en las que la geometría computacional, de la mano de la computación gráfica, están presentes. Entre las más comunes podemos destacar la geografía: para representar zonas territoriales en el plano o en tres dimensiones, ya sea replicando ciudades o lugares del mundo a cualquier escala. En diseño industrial: se moldean piezas de maquinaria para su posterior construcción y ensamble. Aeronáutica: Unas de las industrias que mayor impacto tiene en la movilización de masas y que requiere plena exactitud en la construcción de los elementos que conforman las aeronaves.

Finalmente encontramos las simulaciones, las cuales nos permiten estudiar de manera cercana y a la vez controlada diversos entornos o situaciones que en la vida real serian demasiado riesgosas y/o costosas de experimentar. Estas buscan simular un modelo abstracto de un determinado sistema y replicar los factores más incidentes sobre el área de estudio para ilustrar a los investigadores sobre el comportamiento del sistema cuando varían dichos factores.

## 7.2.- Computación grafica

La computación grafica está complemente extendida en diversas áreas del conocimiento y otras muchas industrias que necesitan mostrar imágenes a sus clientes, por otra parte, existe el rubro del entretenimiento en cual encontramos por ejemplo los video juegos, películas que llevan incorporadas imágenes generadas por computadora, asistentes virtuales que muestran alguna.  Esta se encarga de generar graficas por ordenador. Ya sea en negocios, educación, ocio, etc. Esto se debe principalmente al rápido crecimiento que ha tenido la capacidad computacional en los últimos años.

En la mayoría de las aplicaciones podemos distinguir una capa que nos ayuda a interactuar con ella, se conoce como interfaz gráfica y nos permite manipular elementos y/o entregar cierta información para posteriormente obtener un resultado. A este sistema de comunicación entre la máquina y el humano se le conoce como interfaz gráfica, un medio de comunicación con el que el humano puede entender los datos que se le están requiriendo por alguna razón y posteriormente puede ver el resultado de los datos que ha ingresado. Entre las interfaces más conocidas encontramos ventanas como las de los diversos sistemas operativos tales como Windows, Mac OS, Linux, etc. Todas estas emplean sistemas de representación de información a través de gráficos generados por computadora. Otros como los procesadores de texto despliegan diferentes herramientas para la manipulación del mismo. Por otra parte, distinguimos aquellos programas enfocados a la edición gráfica, tanto en imagen como video. Entonces podemos decir que la computación grafica juega un papel muy importante en la estandarización de la entrada de información a un sistema, es decir, solo se pueden ingresar los datos que se están requiriendo y en el formato que se precisan.

En otras áreas como la de los negocios o la educación se busca representar la información a través de distintas graficas moldear la información, tanto en 2D como en 3D, que se tiene de funciones matemáticas, estadísticas, física, economía, etc. Así podríamos seguir enumerando muchas áreas más como cartografía que busca reproducir lo más fielmente un terreno o área natural incorporando diferentes elementos como el relieve o vegetación.

Después de ver la importancia que tienen los gráficos en el día a día de las personas, empresas y diversas áreas del conocimiento podemos llegar a la conclusión de que la representación de la información ha llegado a ser muy relevante para entender y comprender de lo que se nos está hablando.

## 7.3.- Formas de representar los gráficos

Ya que sabemos que podemos mantener un tipo de comunicación con el usuario a través de las interfaces gráficas es ahora donde debemos tomar el camino para la representación de dicha información, el cómo más que el qué. Es por esto que existen diversas formas de construir imágenes generadas por computador, esto dependerá de la plataforma en la que estemos trabajando, el lenguaje de programación que estemos utilizando y/o finalmente el público al que queramos llegar. Es por esto que nos centraremos una parte que hoy en día tiene mayor peso al momento de representar imágenes. Estas son las mallas geométricas, estas son representaciones de cierta forma abstractas de cuerpos en 3 dimensiones pero que cuentan con un componente especial. A diferencia de lo que normalmente vemos en un cuerpo sólido, las representaciones que se forman a partir de mallas 3D no tienen un relleno, por lo menos las mallas de superficie. Existen otras como las mallas volumétricas que si cuentan con información que nos permite elaborar un modelo capaz de representar dicha información y también concebir una simulación del comportamiento de dichos objetos al interactuar. No es el caso de las mallas de superficie pues el único propósito de estas es representar información que se extrae del mundo real y que pretende virtualizar para su posterior visualización y edición por ordenador. Parece simple el propósito de las mallas de superficie, pero está mucho más allá de la simple percepción que puedan tener los usuarios finales. Para explicar un poco más cómo funcionan las mallas de superficie podemos pensar en la capacidad de abstracción que necesitábamos cuando éramos niños y nos sentábamos a dibujar, si queríamos dibujar una manzana simplemente traíamos la imagen a nuestra cabeza y nos abstraíamos, a un nivel básico, logrando finalmente algún garabato que se parecía en algún grado a una manzana. ¿Pero qué es lo que hace el parecido a algo? La respuesta está mucho más dentro de nosotros de lo que creemos. Este proceso de abstracción es el que nos permite tomar los rasgos más característicos de alguna cosa y plasmarlos en otra, generando una presentación que cuenta con los aspectos fundamentales de lo queremos representar.  Por esto podemos decir que si queremos dibujar una manzana no

podemos dibujar desde a dentro hacia a fuera, quiere decir, las semillas, la pulpa y finalmente la piel de la manzana; simplemente dibujamos su exterior porque eso es lo que identifica a una manzana. Quizá una forma de corazón, un palo en la zona alta y un color rojo o verde. Todos elementos característicos corresponden a un proceso de abstracción.

Lo mismo sucede cuando queremos representar un cuerpo a través de una malla, no podemos emplear tantos recursos en los elementos internos que componen dicho cuerpo, sino solo los elementos característicos del cascarón. Hacerlo de esta manera nos permite ahorrar recursos en tiempo de procesamiento y hardware.

A la hora de representar una malla de superficie podemos hacerlo empleando diferentes figuras geométricas, más específicamente polígonos. Estos pueden ser cuadrados, triángulos, hexágonos, etc. Todo dependerá del modelo que queramos representar. Si bien cada una de las figuras anteriormente nombradas tiene sus propiedades y características, estimamos que el triángulo posee características que nos permiten operar sobre el mismo para extraer información o aplicar cálculos matemáticos que posteriormente servirán tanto para refinar la malla o influir sobre otros triángulos.

## 7.4.- El Triángulo

Desde la estructura de una bicicleta hasta la medición de las estrellas más lejanas en el cosmos, los triángulos nos proporcionan la utilidad de obtener información valiosa, han sido utilizados por la humanidad para crear asombrosas estructuras, como las pirámides, puentes y hasta lo más cotidiano en la vida del ser humano. Las aplicaciones que se pueden realizar con los triángulos son fundamentales para algunas ramas de la ingeniería. Por ejemplo, en la construcción de puentes, los pilares que les dan soporte están construidos por triángulos, pues, es la única figura geométrica que distribuye el peso de manera uniforme y no se deforma cuando actúa una fuerza sobre ella, no así un cuadrado al que es necesario triangular para proporcionarle un soporte. En la rama de astronomía, el Pársec es

un tipo de medición, con el que por medio de tres elementos: Tierra, Sol y Estrella, al unirlos forman un triángulo, así, haciendo uso de la trigonometría, los Astrónomos pueden obtener la distancia aproximada de las estrellas que se observan en el firmamento.

Un triángulo es una región cerrada del plano delimitada por tres segmentos (o aristas) que se unen en sus extremos a los que denominaremos vértices. Existen distintos tipos de triángulos, se pueden clasificar por sus lados o por sus ángulos. El triángulo equilátero que tiene la propiedad de que todos sus lado y ángulos son iguales, el isósceles que posee dos de sus lados y dos de sus tres ángulos iguales. Pero existen una propiedad indispensable para hablar de una figura geométrica como el triángulo, y es que la suma de sus tres ángulos interiores equivale a 180°. Otra propiedad que posee el triángulo es la de no deformarse cuando se aplica una fuerza de comprensión en uno de sus vértices, este siempre mantiene su forma, no así un cuadrado, rectángulo o un pentágono que poseen una estructura que no es rígida por lo cual es fácil de deformar, esto se soluciona triangulando la figura de manera que quede firme y no se deforme (fig 1.1).

En la figura 1.2 tenemos un triángulo que se encuentra en tres dimensiones, esta figura tridimensional es uno de los elementos esenciales para lograr nuestro objetivo, pues, la malla que usaremos está compuesta por estos elementos. Además, al hacer uso de una malla no estructurada, los triángulos que generaremos no están limitados a un solo tipo, sino que tendrán distintas características:

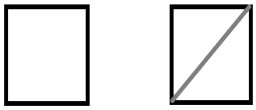


*Figura 1.1: Triangulación de un Cuadrado.*

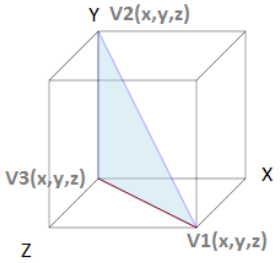



*Figura 1.2: Triángulo en 3D*

Tipos del Triángulo:

a) Triángulo Escaleno: Son polígonos que poseen sus tres lados de distinta medida y a su vez sus tres ángulos interiores poseen distintas magnitudes.
b) Triángulo Equilátero: son aquellos polígonos que poseen tres de sus lados y sus tres ángulos de igual medida.
c) Triángulo Isósceles: Son los polígonos que poseen dos lados de igual longitud y uno de diferente medida. También poseen dos de sus ángulos de igual medida.

Características del Triángulo:

- Vértice: para un plano 2D, es un punto que posee un ancho y un alto, así, la ubicación de este se expresa con las coordenadas X e Y respectivamente. Para nuestro propósito, vamos a requerir de un punto en un contexto 3D, por lo que se requiere de una tercera coordenada que posea información de la profundidad en la que se encuentra el punto, de esta forma obtendremos el ancho, alto y la profundidad. Esta última coordenada la denotaremos con la letra Z.
- Arista: es la línea o segmento que se crea al enlazar dos vértices. Esta unidad posee información de la distancia entre un punto y otro. También puede ser definida como borde, puesto que marca el límite de una figura generando una figura cerrada.
- Angulo: Cuando se intersecan dos aristas o segmentos en un vértice se crea un ángulo, estos se miden en grados.
- Caras: son los lados planos del triángulo, posee información del área que cubre la figura, en un plano tridimensional posee dos caras una de frente y una en la parte posterior.

En conclusión, un triángulo es un polígono que está conformado por 3 vértices, 3 aristas y 3 ángulos internos que suman 180°. Tal elemento es esencial en nuestro camino a concretar una malla de superficie de calidad,

flexible y que pueda hacer uso de algoritmos que hacen uso de triángulos para mejorar detalles en ciertas áreas del rostro.

## 7.5.- Triangulación

Es la unión de puntos generados en un espacio 2d o 3d o la subdivisión de una figura geométrica generando una malla de triángulos o tetraedros. Para realizar este proceso de triangulación se requiere de un conjunto de puntos como datos de entrada, estos puntos (también conocidos como nube de punto) se unen para generar una malla de triángulos. La diferencia entre un triángulo y triangulación es que la primera es una figura geométrica y la segunda es un procedimiento que consiste en la creación o subdivisión de una figura geométrica en triángulos. Existen varios tipos de triangulaciones, va depender de varios factores como del objeto que se quiere subdividir, si se quiere realizar algún criterio o el modo en que se realizará y en qué espacio se realizará (2D o 3D).

### 7.5.1 Tipos de Triangulaciones

Existen distintas técnicas para realizar triangulaciones, algunas son dependientes del contexto en el que se utiliza. Pueden ser realizadas en ambientes 2D o 3D, pues, cada contexto contiene características independientes. Si bien son parecidos, ya que ambas poseen coordenadas x e y, el proceso de generar triangulaciones que están fabricadas para ambientes 2D, requieren de una gran modificación al momento de querer utilizarlas en 3D, pues, bajo este contexto se añade la coordenada z que dificulta el proceso de triangulación. Aun así, existen técnicas que pueden ser utilizadas en ambos ambientes. De las qué podemos destacar tres tipos (Boris Vintimilla, 2000):

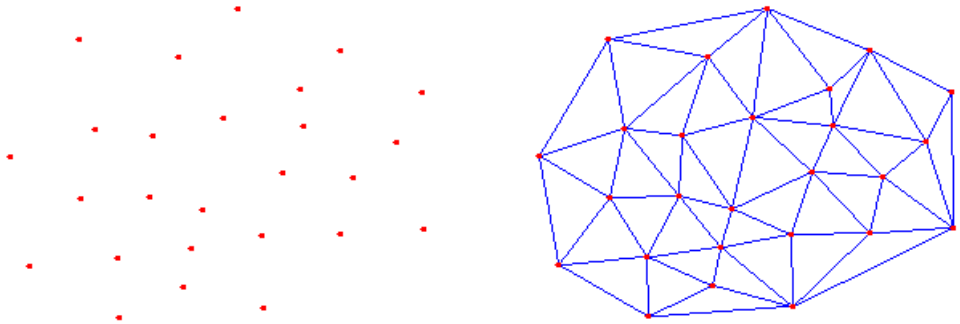
Triangulación 2D: Consiste en un conjunto de puntos en un plano que unidos conforman triángulos que no se superponen. Existen dos tipos de algoritmos de triangulación para este modelo, con o sin optimización.

a) Triangulación 2D sin optimización: Son algoritmos que no requieren de un criterio para realizar la triangulación. Lo que si considera es la propiedad de que los triángulos no deben superponerse.

b) Triangulación 2D con optimización: Estos algoritmos consideran al menos un criterio de calidad para realizar la triangulación. Estos criterios están directamente relacionados con el triángulo, puesto que la calidad se mide de acuerdo a sus ángulos interiores, o a la longitud de sus aristas, a la altura y al área del triángulo generado. Uno de los algoritmos más utilizado para la creación de triangulaciones es el de Delaunay, que considera la optimización de varias medidas mencionadas anteriormente.

Triangulación de Delaunay

En geometría computacional, uno de los primeros algoritmos que genera una triangulación es el de Delaunay, que contiene múltiples usos en el área de la ciencia y la ingeniería. En base a una nube de puntos es capaz de generar una triangulación con la condición de que la circunferencia que circunscribe al triangulo no posea ningún vértice de otro triangulo que pertenezca a la malla (Fig. 1.2). Generando así una triangulación con la condición de Delaunay, en la que los triángulos generados son lo más equilátero posible.



(A) (B)

*Figura 1.3: A) Nube de Puntos B) Triangulación de Delaunay*

Triangulación 2 ½ D: Para generar este tipo de triangulación es necesario comenzar con la lista de puntos creadas en un espacio 3D, este modelo requiere de tres pasos fundamentales. Primero se requiere llevar los puntos desde un espacio 3D a un plano en 2D. Segundo, se aplica algún tipo de triangulación descrito anteriormente para un plano 2D y finalmente se trasladan nuevamente los puntos al espacio 3D de acuerdo a la triangulación utilizada, que puede ser con o sin optimización.

Triangulación 3D: Este tipo de triangulación es generada completamente en el espacio 3D, también es llamada tetrahedrización y es utilizada cuando no se puede realizar una triangulación de tipo 2 ½ D, ya que al llevar los puntos a un plano estos se superponen provocando una alteración en los puntos originales. Así como en 2D es posible generar triangulaciones con o sin criterio de optimización.

Las triangulaciones pueden ser usadas en distintas áreas de las ciencias e ingeniería, de las que se pueden obtener aplicaciones como: modelación de terrenos, análisis de fenómenos físicos que son modelados mediante ecuaciones diferenciales, en computación grafica e incluso en videojuegos, entre otros.

En conclusión, nuestro trabajo hará uso de una triangulación en 3D para generar la malla de superficie que estará compuesta por triángulos y no tendrá ningún criterio de optimización, pues, dicha triangulación quedará en manos de la librería javaFx, la que será explicada más adelante.

## 7.6.- Método de Elementos Finitos

Es en esencia un método numérico para la resolución de ecuaciones diferenciales, estas ecuaciones rigen el comportamiento de varios fenómenos estudiados en la historia de la humanidad y algunas de ellas deben sus nombres a personalidades científicas en el área de la ciencia tecnológica aplicada. Así, surgen modelos matemáticos asociados a varios fenómenos de la física como el movimiento vibratorio o la difusión de calor, en química los procesos de reacción-combustión y en ingeniería para obtener diseños óptimos de vigas o flexión de vigas, entre otros. Las ecuaciones diferenciales utilizan la discretización para dividir el dominio de alguna estructura. Por ejemplo, un puente hecho de concreto, se puede discretizar en un numero finito de partes para lograr un parecido, entre más elementos posea esta estructura, más cercano estará al dominio. Los cálculos que pueden llegar a generar estas ecuaciones para que sean resueltas por una persona son excesivas, estas pueden ser desarrolladas con la aparición de los ordenadores, pues esto entregan el resultad en segundos y a lo más en minutos. Este método proporciono la aparición de las mallas geométricas.

## 7.7.- Mallas geométricas

Una malla corresponde a un conjunto de figuras poligonales que forman una geometría, ya sea en el plano o en el espacio, a diferencia de la triangulación una malla es el producto que se obtiene del proceso algorítmico de realizar la triangulación. La que se genera a través de diferentes procedimientos, uno de estos es unir los puntos que ese encuentran en un contexto 2D o 3D, para formar una

triangulación con o sin algún criterio de optimización. En resumen, una malla corresponde a la triangulación de un cierto dominio que modela un objeto del mundo real.

Existen varios tipos de mallas, podemos categorizarlas en aquellas que se sitúan en el plano, estas cuentan con puntos que solo contienen 2 coordenadas x e y, sin embargo, otras como las 3D cuentan con 3 dimensiones en cada uno de sus puntos, otorgándoles además una altura z en el espacio. Este tipo de mallas son las que nos permiten obtener una mayor fidelidad a la hora de representar objetos de la vida real ya sea para visualización o simulación. Otra categoría en la que podemos clasificas las mallas de 3 dimensiones es en si estas están rellenas o huecas, a las segundas les llamaremos mallas de superficie pues solo nos importa su exterior. Podemos encontrarlas en diversas aplicaciones, pero las más comunes están en el área del entretenimiento ya sean los video juegos o animaciones. Por otra parte, tenemos las mallas volumétricas donde las partes que no son visibles cobran protagonismo, es decir, podemos de igual manera obtener datos que están siendo aparentemente no utilizado. Para ilustrar este tipo aplicaciones de mallas podemos pensar en la simulación y análisis de estructuras, deformaciones, fisuras, etc. Todas ellas comparten estructuras de datos que sirven para realizar una representación más fiel de la realidad.

### 7.7.1.-Malla de superficie

Este tipo de malla surge del proceso de realizar una triangulación con algún criterio de optimización. Sirven para modelar estructuras de la vida real por medio del método de elementos finitos mencionado anteriormente y es una de las herramientas más poderosas para la comunidad de la ingeniería hoy en día. Existen muchos métodos por los cuales modelar una malla, pueden ser manuales, automáticos o semi-automáticos. Cada uno con sus respectivas ventajas y desventajas. Los métodos más utilizados para generar este tipo de mallas es la

triangulación de Delaunay, pues existe muchos algoritmos que permiten trabajar bajo este proceso. Si bien en 2 dimensiones, los algoritmos funcionan robustamente, encuentran dificultades al momento de trabajar en mallas tridimensionales. Por otra parte, existen mallas sin criterios de optimización, que requieren de un trabajo posterior para mejorar su calidad, este proceso se denomina refinamiento. Por lo general este tipo de malla sin optimización, son del tipo no estructurada, ósea que no posee alguna propiedad que las caracterice, sino que simplemente requieren de algún modelo de discretización que mejore la flexibilidad y calidad de estas. Existen varios algoritmos que mejoran la calidad de la malla, como el Lepp-Bisection, que permite refinar mallas en 2D y 3D.

## 7.8.- Refinamiento de mallas

Las mallas de superficie son un elemento primordial en el área de la ingeniería, ya que sirven para modelar estructuras complejas por medio de elementos poligonales como triángulos, cuadriláteros o tetraedros. Para nuestro caso, requerimos del modelamiento de una cabeza humana. Nuestra cara es el medio por el cual nos comunicamos, pues nuestros cinco sentidos se encuentran en esta área, por lo tanto, es complejo hacer un análisis de los múltiples gestos que se pueden realizar. Así, cuando se requiere modelar algo tan complejo como el rostro es necesario poseer una malla de calidad. Esto se logra a través de algoritmos de refinamiento que permiten mejorar considerablemente la calidad de una malla, y así lograr mayor exactitud al modelar estructuras tan detalladas como el rostro.

## 7.9.- Proceso de refinamiento

El proceso de refinamiento consiste en insertar nuevos vértices y nuevos elementos poligonales a la malla inicial. Esto se realiza por medio de métodos de selección que pueden ser de forma local o global. En la forma local se hace la selección de uno o más triángulos de un área específica que se quiera refinar, como por ejemplo el cuenco del ojo, los labios, la frente, etc. Por otro lado, la forma global hace un refinamiento a todos los elementos de la malla y se puede realizar más de una vez.

Algunos algoritmos de refinamiento permiten realizar mejoras en alguna de las propiedades del elemento geométrico de la malla, como su volumen, el tamaño de los ángulos internos o la superficie de estos. Por tanto, lo ideal del proceso de refinamiento es realizarlo sobre una malla gruesa, es decir que la malla inicial posea pocos elementos de entrada, y, en nuestro caso, nuestra malla posee pocos triángulos. Así, el proceso de refinamiento logra crear más elementos o triángulos, dando como resultado una malla fina que posee más triángulos que al principio, pero no pierde su calidad, sino que la mejora significativamente. Además, resulta más fácil modificar el rostro, pues la cantidad de puntos iniciales es menor.

## 7.10.- Algoritmo Lepp-Bisection

Es un algoritmo que fue diseñado para realizar refinamiento en mallas compuestas por triángulos. Hace uso del método de bisección de la arista más larga para mejorar la calidad de la malla y generar secuencias de discretización flexibles. Es capaz de realizar un refinamiento local de forma iterativa logrando mantener la calidad de la malla inicial. Además, el algoritmo posee un costo de tiempo lineal, por lo que el proceso de refinar mantiene el tiempo de ejecución sin aumentar o reducir su costo. Existen varios algoritmos de refinamiento bajo el concepto de Lepp o arista más larga, como el Lepp Delaunay el que es bastante eficaz en un contexto 2D. En nuestro caso, el uso de Lepp-Bisection nos permite obtener un refinamiento de calidad en el espacio 3D. Estos algoritmos heredan las propiedades del método de

bisección de triángulos, en el que se garantiza la bisección de triángulos de mantera iterativa a través de la arista más larga.

Los matemáticos Rosenberg, Stenger, Stynes y Andler, fueron los pioneros en el estudio de los métodos de bisección, de ellos se pueden desprender las propiedades matemáticas que posee este algoritmo (Rivara, 2008):

a) Para cualquier triangulo t de ángulo más pequeño α, la bisección iterativa para para las aristas más largas junto con sus descendientes producen triángulos t' cuyos ángulos interiores son siempre mayores o iguales a α/2.
b) Cada triángulo que se genera en la iteración produce triángulos que están asociados pero que no necesariamente son similares entre sí.
c) La bisección iterativa global de un triángulo, cubre de manera interna su área con triángulos aproximadamente equiláteros.

En Lepp-bisection se realiza una búsqueda iterativa para encontrar la arista más larga de un triángulo seleccionado. Esta arista es compartida por otro triángulo que es llamado vecino, luego, a partir de esa arista, se obtiene su punto medio y se crea un nuevo vértice. El nuevo vértice produce una bisección en el triángulo, dando como resultado dos nuevos triángulos. Por lo tanto, la búsqueda iterativa se puede propagar a más de un vecino, puesto que se dan situaciones en que los triángulos que se refinan pueden quedar como cuadriláteros, y es necesario buscar un punto terminal hasta que el refinamiento sea óptimo. La figura 1.4 muestra la propagación que realiza el algoritmo de refinamiento para lograr obtener la arista terminal más larga y, así, crear los nuevos triángulos. En definitiva, este algoritmo hereda las propiedades mencionadas anteriormente, para mantener y conformar la calidad de la malla inicial. Un segmento o arista S se denomina terminal en el refinamiento R si los triángulos vecinos comparten también ese segmento más largo S. Además, los triángulos que comparten esa arista se denominan triángulos terminales. Si la arista S es compartida por dos triángulos terminales, esa arista es interior, pero si S es compartida por un solo triángulo terminal, entonces S es una arista terminal de borde.

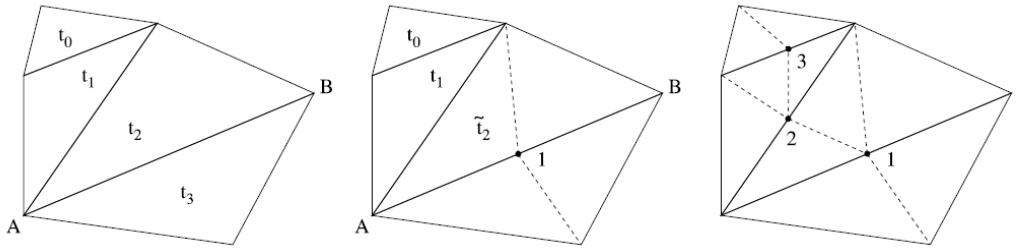



*Fig. 1.4: Propagación del algoritmo Lepp –Bisection*

Lepp-Bisection funciona automáticamente, o más bien como una caja negra, pues se le pasan los triángulos que se quieren refinar, luego realiza todo el proceso mencionado anteriormente y devuelve los nuevos triangulo y además modifica los triangulo que le fueron entregados. Por esta razón, podemos reflejar ciertas ventajas que este algoritmo otorga a la malla que se utilizará en este trabajo, de las que podemos mencionar:

1.- El proceso de refinamiento se hace de manera local, lo que nos garantiza que la malla mejore o mantenga su calidad.

2.- El algoritmo es fácil de implementar.

3.- A pesar que el algoritmo se ejecuta principalmente en 2D, es posible hacerle una leve modificación para ejecutarlo en 3D, manteniendo la función de refinar de forma local.

En definitiva, el algoritmo de refinamiento Lepp-bisection, funciona perfectamente en la malla de superficie generada, pues al ser gruesa, ósea con pocos elementos grandes de entrada, nos permite realizar un refinamiento adecuado a las necesidades de un retrato hablado, pues nos permite refinar zonas específicas del rostro en muy poco tiempo.

# 8.- Descripción del problema

## 8.1.- Rostro Humano: formas y elementos

El rostro humano está compuesto de múltiples elementos que permiten distinguir a una persona de otra, entre los elementos principales que caracterizan la cara de una persona podemos encontrar la forma de la misma, existen diferentes características en cuanto a la contextura de la cara, esta puede ser delgada, más ancha, ovalada o incluso cuadrada, todo dependerá del país y el grupo étnico que provenga. Aunque el concepto de raza empezó a decaer desde 1950 con la llegada de nuevas corrientes antropológicas, pues se busca evitar prácticas racistas hablaremos de naturaleza humana, según lo describe la antropología clásica y la nueva antropología biológica, la cual toma una amplia variedad de rasgos cualitativamente superior a la antropología física propuesta por O'Rourkey y Petersen, 1983.

Existen tantas clasificaciones como autores, pero para efectos de simplificar los conceptos y no caer en términos tan estrictos de la misma, hablaremos desde las investigaciones que han realizado desde el siglo XVIII las cuales corresponden a las siguientes:

Podemos distinguir 3 tipos de naturalezas humanas predominante:

- Naturaleza blanca, caucásica o europea
- Naturaleza negra o africana
- Naturaleza Amarilla o asiática

El elemento característico más importante a la hora de reconocer una persona es la forma de la cara, esta depende de la estructura del cráneo y este a su vez de la naturaleza que provenga.

Podemos distinguir 3 tipos de cráneos:

El primero tiene forma alargada y estrecha llamado también cráneo dolicefálico, el segundo tipo es un cráneo más homogéneo en cuanto a sus medidas, pues tiene

una dimensión mediana y más estándar y redondeado, es también llamado cráneo mesocefálico. Finalmente encontramos el cráneo con forma cuadrada es más achatado que los demás y también es llamado cráneo branquifálico.

Según estudios antropológicos se le atribuye el primer tipo de cráneo a las personas de naturaleza negra, en la segunda clasificación entrarían las personas de naturaleza blanca y finalmente estarían las personas de raza amarilla que pertenecerían a las ultima clasificación.

Aunque como se mencionó antes, esta es solo una aproximación, pues con el pasar de los años muchos de estos pueblos se han mezclado dando origen a múltiples combinaciones.

Otro factor importante es la edad de la persona pues el tamaño y forma del cráneo varía a medida que pasan los años.  Otro factor importante que incide en la forma de la cabeza es el sexo de la persona en cuestión.

Todo esto con el fin de contextualizar los tipos de cráneos que existen, así como la pigmentación de la piel y los rasgos faciales más característicos de cada grupo étnico. Si bien estos pueden variar ya que el fenotipo se puede expresar de diversas maneras en la unión de las características heredadas por los hijos, es necesario dejar claro cuáles son aquellos rasgos por los cuales podemos distinguir a una persona y porqué esta es así.

## 8.2.- La nariz, los ojos y la boca

Estos 3 elementos permiten diferenciar a una persona de otra. Existen muchos tipos y combinaciones diferentes de narices, ojos y boca. Aunque todos ellos varían según su tamaño, forma y pigmentación.

### 8.2.1.- Nariz

Partiendo por la nariz podemos encontrar diferentes tipos, entre ellas, las narices aguileñas, que se caracterizan por ser lo contrario a las narices respingadas, estas se dirigen hacia abajo y forman una curvatura convexa. Otro tipo de nariz es la carnosa, la cual tiene una forma bulbosa, se caracterizan por ser grandes y prominente. Además, encontramos la nariz respingada, la cual se caracteriza por ser relativamente pequeña con una depresión en el medio del puente y la punta sobresaliente. Es el estereotipo de nariz más deseado en las cirugías plásticas.

Otro tipo de nariz muy común es la nariz romana, en esta sobresale el hueso del puente de la misma y lleva una leve curvatura. También encontramos la nariz Nubia la cual presenta un puente largo y finaliza con una base ancha, se encuentra con mayor frecuencia en personas afrodescendientes. Finalmente tenemos la nariz griega, esta se caracteriza por tener un puente notablemente recto, sin jorobas ni curvas.

Cabe señalar que podemos encontrar en los diferentes rostros muchas combinaciones de estas narices, estas son solo las bases que permiten realizar múltiples combinaciones.

### 8.2.2.- Ojos

Si bien el ojo consiste en todo el órgano que permite al cerebro percibir lo que está sucediendo en el entorno, para este trabajo nos vamos a referir al ojo como toda la zona que encierra tanto el globo ocular y todos los elementos que lo acompañan ya sea el iris, la córnea, etc. Pero también incluiremos todos los aspectos que permiten distinguir a una persona de otra, hablamos de los parpados, su tamaño, la forma, también incluiremos las cejas.

Estos 3 elementos combinados, el ojo, los parpados y la ceja, permiten distinguir una persona de otra y al mismo tiempo encasillar los diferentes tipos de “ojos” que existen.

Ojos normales o almendrados: deben su nombre a la apariencia que tienen con las almendras, este tipo de ojos son los más comunes y los tienen casi todas las personas.

Ojos estrechos: este tipo de ojos se caracteriza por tener los ojos bastante juntos, es decir, a poca distancia entre ellos.

Ojos saltones: corresponden a ojos que sobresalen más de lo normal, los párpados no cubren todo el globo ocular y dejan visible una amplia zona de estos.

Ojos asiáticos: también llamados ojos rasgados. En un principio son más pequeños de lo común y demás son alargados en los laterales, permitiendo ver una figura del ojo más que redonda, ovalada y estirada a los lados.

Otro tipo de ojos son los ojos caídos, los cuales se caracterizan por tener el parpado superior un poco más bajo de lo normal, lo cual da una impresión de sueño.

Finalmente encontramos los ojos apartados, los cuales como bien dice su nombre están distanciados uno del otro y por lo general se ven más pequeños con respecto al resto de los rasgos faciales.

Para anexar algo a los ojos podemos dejar las cejas como una parte que está presente en la zona de los ojos que también presenta diferentes combinaciones y estructura en cuanto al tamaño, color, cantidad de bello y ubicación en la cara. Entre las más comunes podemos destacar las cejas delgadas, éstas cuentan como poco bello y son bastante definidas. Otro tipo de cejas son las cejas gruesas, éstas cuentan como mucho más bello en diferentes direcciones y la ceja no esta tan definida. Otro tipo de ceja es la ceja unida en la cual no hay un límite entre el inicio y termino de una ceja y la otra. Entre los diferentes tipos de cejas que existen, cada una varía según su tamaño y cantidad de bello.

### 8.2.3.- Boca

Finalmente encontramos un rasgo facial que pasa muchas veces desapercibido por hombres pero que en mujeres es mucho más visible. Este rasgo corresponde a los labios, la boca en general. Ésta está compuesta por los labios y dientes. Sin embargo, para efectos de este trabajo solo tomaremos los labios y la forma de la boca exterior como parte del proyecto. Podemos distinguir principalmente 2 componentes de la boca el labio superior y el inferior. Estos son independientes uno del otros y no necesariamente tienen que ser del mismo tamaño ni grosor. De esta manera podemos generar diferentes combinaciones que nacen la unión de labios gruesos y delgados tanto en la parte superior como la inferior.

## 8.3.- Elaboración del retrato hablado

El concepto de retrato hablado parte de la práctica que se creó en 1887 mediante la cual se describe el rostro de una persona, sin embargo, la forma correcta de llamarlo es retrato compuesto, ya que a partir de una serie de variables tales como la memoria de la víctima, descripciones orales, dibujo, programas de computadora, etc. Se puede reconstruir el rostro de una persona.

El trabajo que se realiza es mediante los datos que llegan, las descripciones que la persona realiza y la posterior interpretación de los mismos. Cabe señalar que muchas de las personas que se ven en la situación de describir a otro no son capases de extraer características específicas del sujeto por el cual fueron perjudicadas y entonces en esta instancia se procede a buscar los rasgos faciales más importantes de la persona en cuestión. Aquí juega un rol fundamental el perito en arte forense, quien debe aterrizar el lenguaje al que utilizan las personas y a la vez establezca una empatía con la persona afectada. Para esto se utiliza un lenguaje coloquial.

Por este motivo es muy importante que en la entrevista la persona que está describiendo al sujeto lo haga de la forma más precisa posibles pues es esta quien provee la materia prima para la posterior construcción del retrato hablado.

Es necesario señalar que cuando se realiza un retrato hablado con las herramientas que existen en la actualidad no se pueden abarcar todos los tipos de razas y todos los tipos de países, sino que cada país cuenta como una herramienta propia que toma las características principales de sus ciudadanos para obtener un conjunto de tipologías que irán variando dentro de un contexto. Podemos dar el ejemplo de Caramex, un software utilizado en la procuraduría general de justicia en ciudad de México para la creación de retratos hablados en 2 dimensiones, el cual toma como referencia muchas de las características de las personas que viven en México .Por esta razón el software tendrá una alta eficacia dentro del rasgo de personas que viven en ese país, sin embargo si se quisiera identificar personas que rompen con el estereotipo mexicano tales como orientales o europeos del norte tendría un menor grado de calidad en el boceto realizado. Otro ejemplo es el del software faces de estados unidos el cual es utilizado por la CIA, el FBI y el ejército de los Estados Unidos para el mismo propósito. Este último se puede obtener por cualquier persona o agencia que lo necesite.

## 8.4.- Técnicas y herramientas

Como se mencionó anteriormente existen herramientas como Faces o Caramex que cuentan con un entorno 2D en donde se pueden sobreponer capas para al igual que Photoshop crear una composición, en este caso de un rostro, agregando rasgos primarios como el tipo de cara, los ojos, la nariz y finalmente la boca. Posteriormente se pueden añadir rasgos secundarios como cicatrices, espinillas, lunares, arrugas, o también otros elementos como las cejas, bigote o barba. Todo dependerá de la descripción que de la persona.

Si bien estas herramientas nos permiten realizar un retrato hablado en menor tiempo que el que tomaría hacerlo en papel y además nos permiten tener una mayor

flexibilidad en cuanto el transporte de esta información y el formato en que se está trabajando puede ser compartido con otros ordenadores existe un problema y es que estos rasgos con los que usualmente trabajan estos programas son muestras que ya vienen listas y que no se les puedes aplicar modificaciones más allá del tamaño y la inclinación tanto hacia la derecha como a la izquierda del rostro, esto nos genera una limitante en cuanto a la cantidad de combinaciones que podemos realizar (el software Faces posee alrededor de 4.400 muestras en las que encontramos caras, peinados, ojos, bocas, narices, etc). Por ende, debemos buscar entre todos los elementos con los que cuenta la biblioteca del programa y ver si encaja con la descripción que se no ha dado. Si bien elegir muchas combinaciones diferentes es un hecho que estas son finitas y limitadas. La idea subyacente es poder manipular cada uno de los elementos en base a mallas de triángulos, esto nos permite deformarlas de la manera que quedamos y obtener infinitas combinaciones entre ojos, narices, bocas y la cabeza misma.

Existen programas informáticos que están dedicados el modelado de rostros y cuerpos en 3D o en general a cualquier elemento que podamos imaginar, hablamos de programas como Blender, una herramienta que nos permite a partir de una malla de cuadriláteros darle forma y jugar con las herramientas que tiene incorporada el programa. Aunque cabe destacar que esta aplicación requiere de horas de entrenamiento para su correcto uso y no está enfocada a la realización de retratos hablados pues su fin es crear modelos en 3 dimensiones para su posterior importación en otros softwares para realizar animaciones o efectos especiales donde se requieran imágenes generadas por computadora. Por este motivo no es el más idóneo para el modelamiento de retratos hablados, entonces se busca una herramienta capaz de modelar rostros en 3 dimensiones que sea fácil de usar, intuitiva, portable y que ofrezca un sistema de edición dinámico.

En general solo existen herramientas que están orientadas al trabajo en 2 dimensiones y bajo el concepto de unir diferentes muestras o imágenes para lograr una estructura a partir de la descripción que haya dado la persona en cuestión. Este

paradigma tiene limitantes pues no se pueden explorar todas las posibilidades de tipos caras rasgos faciales

## 8.5.- Historia del retrato hablado

Desde que se crean las primera ciudades, estados o aglomeraciones de personas, se busca identificar a estas para, de cierta forma, ejercer un control e individualizar a cada ciudadano y así tener un registro de cada uno para mantener la seguridad al interior de las ciudades.

Uno de los antecedentes más importantes que da comienzo a la utilización de los retratos hablados comienza en Francia a finales del siglo XIX, el antropólogo y médico francés Alphonse Bertillon (1853 – 1914) desarrolló una técnica llamada “bertillonaje”, en honor a su apellido, esta técnica consistía en identificar y clasificar a los criminales de forma que pudieran primeramente medir ciertas zonas de la cabeza y las manos. Posteriormente se añaden nuevos elementos que permiten identificar a los criminales de manera más fácil como son las fotografías o las fichas de identificación que contenían más datos de cada uno de ellos.

Posteriormente los servicios forenses empiezan a utilizar los retratos hablados de manera común en casi todo el mundo, sin embargo, se dan cuenta de que el artista que realizaba los dibujos se guiaba bajo un mismo patrón, es decir que la mayoría de los rostros tenían cosas en común, estas se podían relacionar con el entorno en el que estaba envuelto el artista.

A medida que pasa el tiempo en Estados unidos se comienza a desarrollar un sistema para el FBI que permite estandarizar los valores que se le otorgaban a cada parte del rostro, de esta manera se hacía un poco más imparcial la creación de un retrato hablado ya que se contaba con un amplio catálogo de imágenes en donde se podía seleccionar cara parte de la cara con la que la víctima se sintiera más identificada.

Se sigue perfeccionando la técnica de confección de retrato hablado y se comienza a utilizar impresiones de acetato para montarlas una sobre otra y así unir diferentes elementos que finalmente terminaban por definir un rostro más o menos exacto.

Hasta antes de los años 80 que es donde se utilizaban este tipo de técnicas, la computación aun no llegaba a incorporarse del todo y no fue sino hasta los años 90 en donde el uso de herramientas software potencio la manipulación de imágenes y a la vez amplió las posibilidades para mejorar en alto grado la técnica actual.

Luego que el software comienza a incorporarse en la mayoría de las oficinas de peritaje surgen en primera instancia programas informáticos que utilizan capas sobre puestas con partes del rostro por separado que finalmente serán colocadas una encima de otra para realizar un rostro completo incorporando todas estas muestras, esta técnica es conocida como “identikit”. Cabe señalar que países como estados unidos y Europa incorporan software que se ajusta a los tipos de rostros que son de la misma zona. Por otra parte, países como Venezuela prefiere no utilizar este tipo de programas informáticos ya que las muestras que vienen incorporadas no se ajustan al fenotipo de los habitantes de ese país. Queda a criterio de cada país la utilización de programas que utilizan este tipo de muestras o imágenes para crear retratos hablados. Aunque debemos señalar que no es el único método de identificación, este es solo una parte del abanico de posibilidades con las que cuenta los sistemas de seguridad al interior de las ciudades. Debemos señalar que este método de la identificación de rostros tiene una gran relevancia ya que permite individualizar las características principalmente fenotípicas de los individuos para efectos de identificación y captura.

Dentro de los principales motivos para utilización de un retrato hablado están obviamente los delitos tales como secuestros, robo, hurto, violaciones, etc. Y un factor fundamentar son los testigos que tuvieron contacto con el delincuente. Son estos dos elementos los principales componentes dentro de la identificación de personas y la elaboración de retratos hablados.

Ya desde el antiguo Egipto se venían dando sistemas de identificación de personas, dentro de los años 330ac al 30ac. Si bien era un precario sistema servía para

aquellos criminales que buscaban diferenciarse. En un principio tan solo el cambiarse de ropa y cortarse el cabello podía permitir que un delincuente se camuflara dentro de la multitud y pasara desapercibido; por esta razón es que se comienza a dar este tipo de prácticas en donde las personas que están relacionadas a un delito aportan datos para que se registren los rasgos más característicos del delincuente y así los ciudadanos que eran afectados podían contribuir a mejorar la seguridad en la ciudad y de esta manera las autoridades recibían retroalimentación de parte de los ciudadanos .

En primera instancia eran las grandes ciudades las que se preocupaban por el identificar a los criminales otorgándoles alguna marca distintiva o elemento diferenciador. En algunos casos se les amputaban partes del cuerpo a los delincuentes que eran capturados para que así a futuro estos pudieran ser identificados de manera rápida. Otro elemento diferenciador eran las marcas por fuego o hierro caliente tanto en la cara como en la espalda o incluso se les llegaba a cortar la lengua a aquellos calumniadores. Evidentemente estas prácticas cesaron en algún momento, si bien se dieron por muchos siglos, fue a mediados del siglo XX que los derechos humanos buscaron contribuir a la modificación de ciertas prácticas que hoy pueden considerarse barbáricas.

Cuando se instauró ya el concepto de retrato hablado se buscó proceder de una forma en la que se pudiera mantener un registro más detallado y exacto de las características que tenían las personas: hablamos por ejemplo del señalamiento antropométrico que busca (según Ross y Kerr 1988) estimar la composición corporal donde se utilizan modelos anatómicos para estudiar las medidas de las personas tales como el peso, la talla, el diámetro de cabeza o la medición del cuerpo en general. También se encarga de medir la longitud de los brazos, piernas, el tórax, cadera, antebrazo, pantorrilla, muslo, así como también el diámetro óseo, etc. Todo esto con el fin de diferenciar a una persona de otra en cuanto a las diferentes medidas que posee en su cuerpo. Este fue uno de los primeros pasos que se dieron para la construcción de retratos hablados con mayor nivel de exactitud y diferenciación con respecto a otras personas.

Cabe señalar que cuando se realiza un buen retrato hablado entonces se está más cerca de capturar al sospechoso y de que no se abandone la búsqueda por falta de antecedentes o por inconsistencia en los datos que son entregados.

En primera instancia se busca reconocer las medidas de las personas pues son elementos cuantitativos que permiten desechar opciones de manera rápida y precisa, lo segundo corresponde (según Fernando Rodes Lloret, laboratorio forense) al señalamiento descriptivo que es precisamente eso, describir los rasgos del delincuente a fin de conocer cuáles eran las características principales en cuanto a color de piel, color del iris, el largo del cabello, también conocer otros rasgos morfológicos tales como el tamaño de la frente, la anchura de la misma, su inclinación, prominencia, etc. Así también con otros elementos de la cara como la nariz, la forma de las orejas, etc. También existen otros rasgos que son complementarios y sirven para aumentar el parecido con la persona en ella encontramos la altura naso labial, que corresponde a la distancia que existe entre la nariz y el labio superior La prominencia del labio también es importante, así como su grosor, cabe destacar el color del cabello, si existe barba, la forma de las cejas, si hay pliegues distintivos en la piel. Y así con un conjunto de rasgos que se han ido puliendo para aumentar el parecido a la hora de realizar un retrato hablado. Todo esto en el marco del desarrollo y la evolución que ha tenido la disciplina de la criminalística y la actividad de la creación de rostros a través de retratos hablados.

Finalmente nos encontramos con las problemáticas que pueden tener tanto los actuales métodos de elaboración de retratos hablados como los clásicos procedimientos que se realizan en papel.  Cada uno de ellos cuenta con ciertas ventajas y desventajas.

Si realizamos una regresión desde los primeros días en los que se realizaban retratos hablados a mano podemos fácilmente encontrar que el tiempo que se debe dedicar a la realización de uno de estos dibujos fácilmente puede superar los 60 minutos, siempre y cuando se tengan la información necesaria del rostro del individuo y todas las características de las que ya se han hablado con anterioridad. Por esta razón la principal desventaja que presenta el retrato hablado elaborado en

papel es el tiempo que toma realizarlo además de la necesidad de un artista encargado de plasmar la imagen que está tratando de dar la persona. Por esta razón es de suma importancia que el artista primero sea empático con la víctima, pues es necesario indagar en la mente de la misma sin despertar los sentimientos que esta puede alargar al recordar la escena vivida. Siempre se puede dar el caso de que la víctima se vea sobrepasada por los recuerdos evocados en ese momento, aunque no es un elemento que poner en riesgo la elaboración del retrato hablado, si es un factor que puede demorar aún más su construcción.

Pasando a la siguiente forma de elaborar retratos hablados encontramos el modo de sobre posición de imágenes transparentes en la cual como se mencionó anteriormente se sobreponen diferentes imágenes de la cara en donde encontramos elementos tales como la nariz, la boca, los ojos y la forma de la cara como base de dicho retrato. Este método tiende a ser más confiable en cuanto a la calidad del retrato final ya que las imágenes que son seleccionadas cuentan con una alta calidad y es la misma persona la que en conjunto con el “detective” elaboran el retrato hablado. Este método que se utilizaba a finales de los años 80, aunque mejor que el anterior, tiene algunos inconvenientes pues el formato que hasta ese momento se utilizaba era papel con cierto grado de transparencia o en los mejores casos cuando se realizaron posteriores avances podían ser elaborados con láminas de acetato transparente que brindan una mayor versatilidad a la hora de guardar las muestras, pero también cuando era necesario construir el retrato pues otorgaba una mayor calidad que el papel. El principal inconveniente de este método era la forma en la que este papel transparente era transportado pues era muy fácil de malograr y cuando en el caso de que se necesitara compartir este documento era necesario transportar la carpeta completa que contenía todos los elementos que correspondían al retrato.

Si bien después de los años 90 y con la llegada de la computación y la construcción de diferentes herramientas dedicadas al desarrollo de retratos hablados se logró un avance cuantitativamente mayor al que ya se había alcanzado hasta entonces; la principal traba con la que se encuentra este tipo de herramientas es primeramente

la cantidad finita de combinaciones que se pueden lograr con los actuales softwares y esto nos lleva al siguiente problema que es la poca maleabilidad que poseen las muestras, si bien las actuales herramientas nos permiten construir retratos a partir de imágenes pre establecidas alterando el tamaño, inclinación, pudiente además escalar cada uno de los elementos para dar una forma armoniosa al rostros y junto con ello obtener un parecido de alta calidad sigue el inconveniente de que estas muestras no se pueden alterar mucho más allá de como vienen, se necesita además capacitación en cuanto a la utilización de estas herramientas y finalmente no están al alcance de todo el público, ya sea por el precio o por el tipo de distribución que poseen las licencias de los mismo. Cerrada solo para organizaciones policiales o de investigación.

Quizá el elemento diferenciador dentro de las aplicaciones actuales que existen para la creación de retratos hablados es el contexto gráfico en el cual están basadas, es decir, la cantidad de dimensiones en las que podemos trabajar el retrato hablado. El primer retrato hablado que se construyó se realizó en papel, la persona dibujaba trazos sobre una hoja y a medida que iba avanzando podía darse cuenta de cómo el retrato se iba acercando cada vez más a la forma deseada, cuando posteriormente se implementaron láminas de acetato se sobreponía una tras otra como una pila de diferentes elementos, cada uno perteneciente a una parte de la cara. Actualmente los programas informáticos trabajan de forma muy similar a como lo han estado haciendo con anterioridad, sobreponiendo muestras en forma de capas para dar finalmente una imagen parecida a lo que se busca y también a lo que la víctima es capaz de describir. Sin embargo, si echamos volar un poco la imaginación y nos ponemos en la mente de la persona que está intentando describir cómo era el individuo en cuestión, sería mucho más fácil que la misma persona fuera plasmando la imagen que tiene en su mente en una cara exterior con que está viendo constantemente y que además puede modificar, para dar con el rostro que se desea. Hablamos de una aplicación que permita el modelamiento de un rostro en tres dimensiones a cargo de una persona que no necesariamente tiene conocimientos de formas de rostros, de antropología o de cualquier otra área relacionada con el rostro y con el ser humano. Sería mucho más práctico que una

persona estuviera sentada frente a una cabeza y fuera ella misma quien la moldeara según su propio criterio. Este es lo que actualmente aún no se ha desarrollado y podría ser un elemento diferenciador dentro de las actuales aplicaciones informáticas y un nuevo cambio de paradigma dentro de las investigaciones policiales.

Como lo plantea Angel Sierra (2014) la forma en la que se inicia la identificación de una persona parte ya en el siglo XIV cuando en china se almacenaba la comida que se recolectaba de forma común, entonces las personas una vez almacenaban diferentes alimentos en un recinto cerrado volvían después de un tiempo para retirar algunos de sus víveres y así subsistir en los meses más difíciles, de esta forma se debía identificar a la persona para realmente fuera ella quien cogiera estos alimentos y no fuera otra quien retirara esta comida de forma mal intencionada. Por esta razón se confiaba en la memoria visual que pudiera tener el cuidador de dicho recinto.

Por esta razón el concepto de identificación se hace presente para evitar malos entendidos entre los diferentes campesinos que para ese entonces colaboraban en la búsqueda de alimento para subsistir. Este concepto es importante ya que posteriormente se va perfeccionando y se buscan otro tipo de características en las personas, tales como su vestuario, alguna marca característica; posteriormente el nombre pasó a ser un elemento diferenciador dentro de las personas que habitaban el lugar. Sin embargo, a medida que la población fue aumentando fue necesario otros métodos de identificación tales como el apellido paterno, en primera instancia, y hoy en día, además, se utiliza la combinación de 2 nombres y 2 apellidos los cuales corresponden al paterno y materno respectivamente. (crecimiento poblacional)

Es así como a medida que la humanidad se va acrecentando en número es necesario un conjunto de nuevas técnicas de identificación. Ya no basta solo con un nombre o un apellido, ahora se agrega diversos elementos diferenciadores como las huellas dactilares o incluso muestras de ADN para el reconocimiento de las distintas personas.

Otro concepto importante dentro de la identificación de personas es la identidad. Parecen sinónimos, pero existen ciertos matices que hacen que estas 2 palabra pueden diferenciarse una de otra. Según Salvador Martínez y Luis Saldívar el concepto de identidad es el conjunto de caracteres que sirven para individualizar a una persona, diferenciándola por tanto de los demás, mientras que la identificación es el procedimiento para reconocer un individuo vivo o muerto.

## 8.6.- Entrevista descriptiva

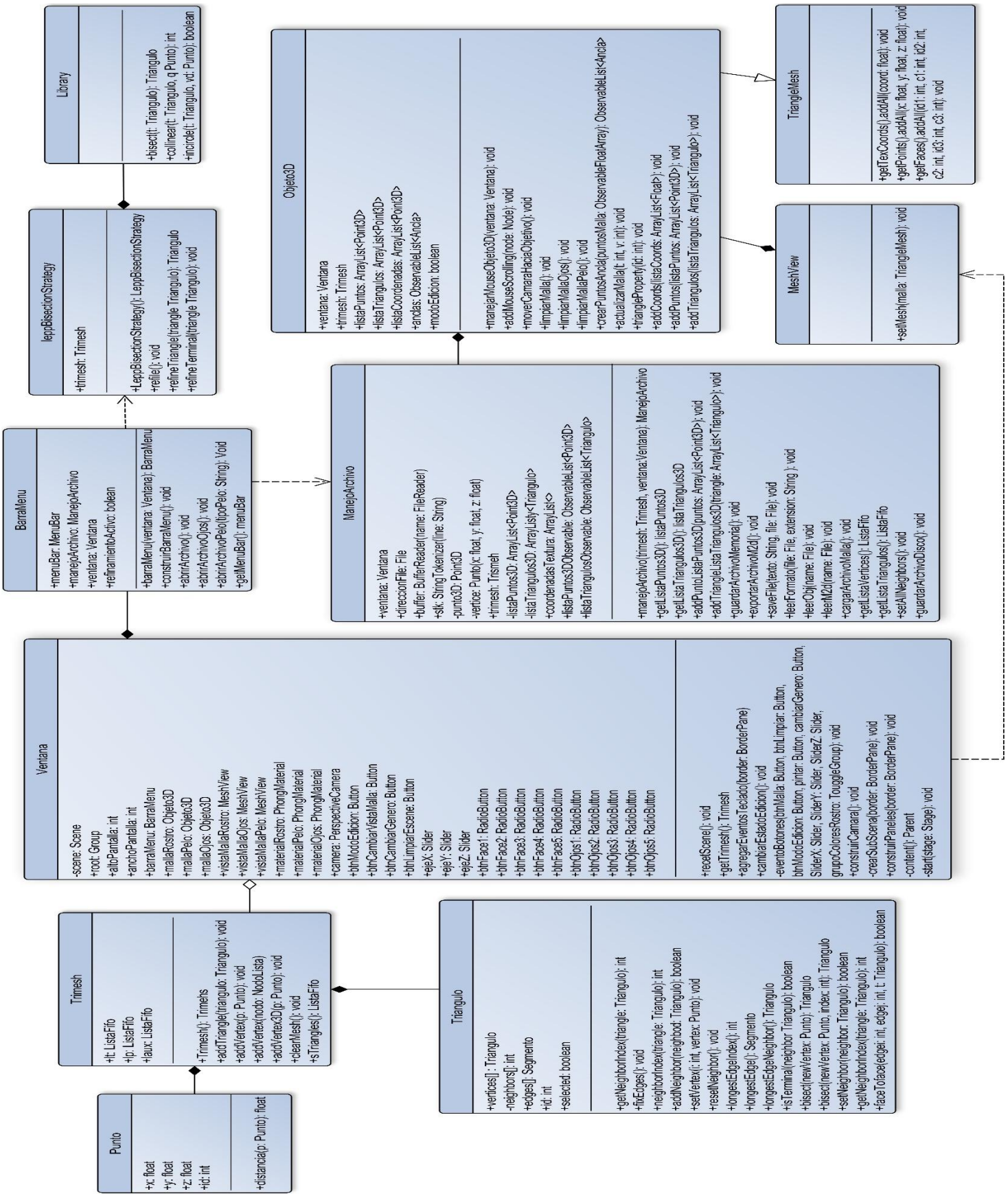
Para centrarnos de lleno en la parte más importante que nos compete como proyecto, presentamos una opción que se puede abordar para el modelamiento del rostro. En todo el proceso de investigación que se va desde las primeras pistas que deja un sospechoso hasta la captura del mismo y la correcta verificación de que es él a quien se está investigando hay muchos pasos intermedios. Sin embargo, es en el proceso de la entrevista y la elaboración del retrato hablado, dónde nos centraremos como proyecto e investigación ya que estos 2 pasos son fundamentales para posteriormente tener éxito en el caso. Si bien existen fuentes variables de investigación para llegar a un sospechoso, el realizar un retrato hablado que cumpla con las características que se buscan permitirá realizar una acelerada investigación en los archivos que posee y de esta manera se contribuye a mejorar los tiempos de respuesta y el dinero que se emplea para localizar a la persona en cuestión.

Según Nelson Farías, la entrevista se define de la siguiente manera : “en el ámbito de la investigación de los delitos, la entrevista podría definirse como una técnica investigativa que consiste en una serie de preguntas efectuadas a las distintas personas que tienen conocimiento o pueden brindar antecedentes acerca de un hecho que se investiga, fundamentalmente con el objeto de obtener información que conduzca al esclarecimiento del delito que se investiga, reunir evidencia y poder llegar al o a los responsables del crimen”.

Por esta razón es que la entrevista es tan importante para la recopilación de datos y el posterior volcamiento de los mismos en el retrato hablado.

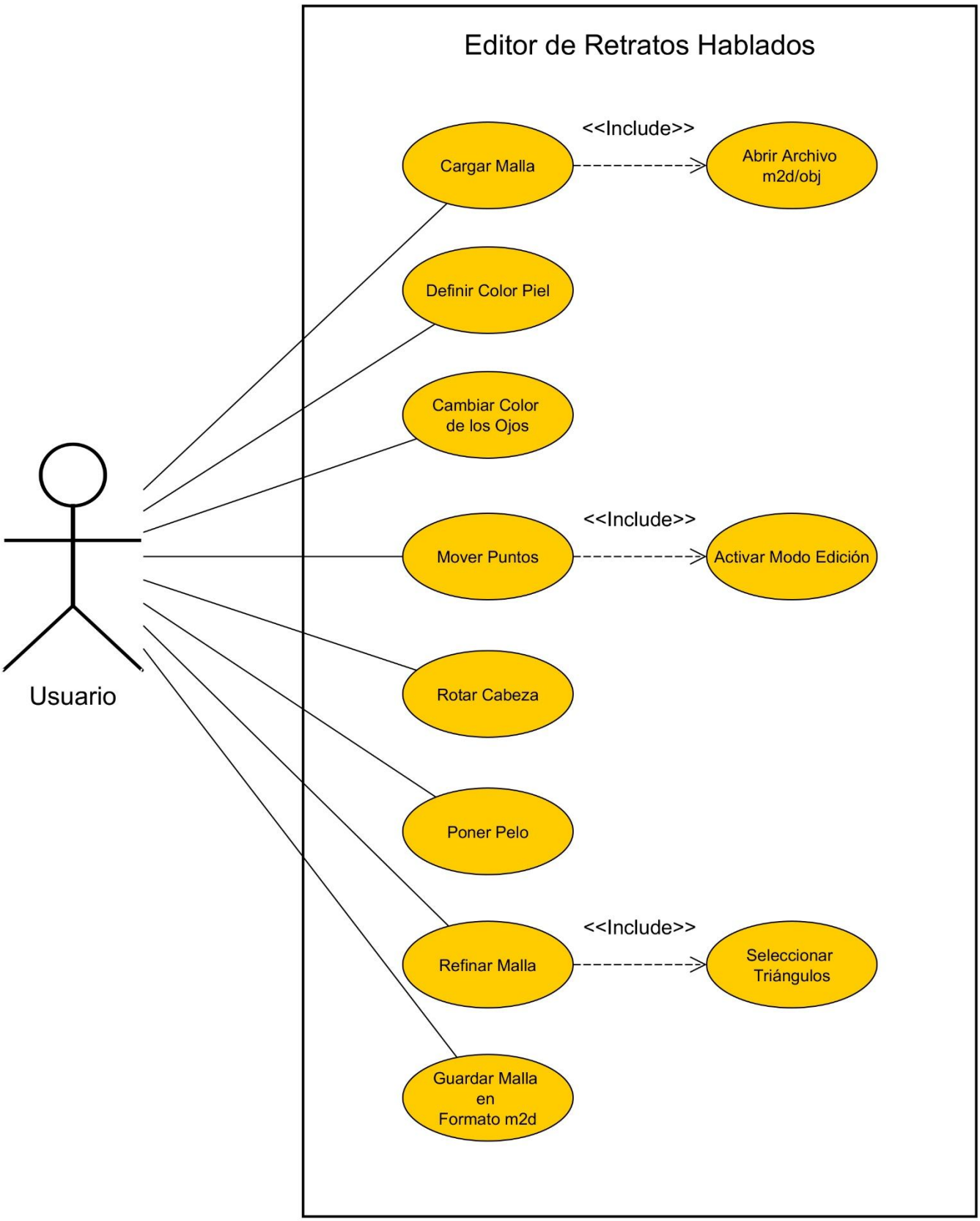
# 9.- Análisis y diseño de la aplicación

## 9.1 Diagrama de clases (UML)

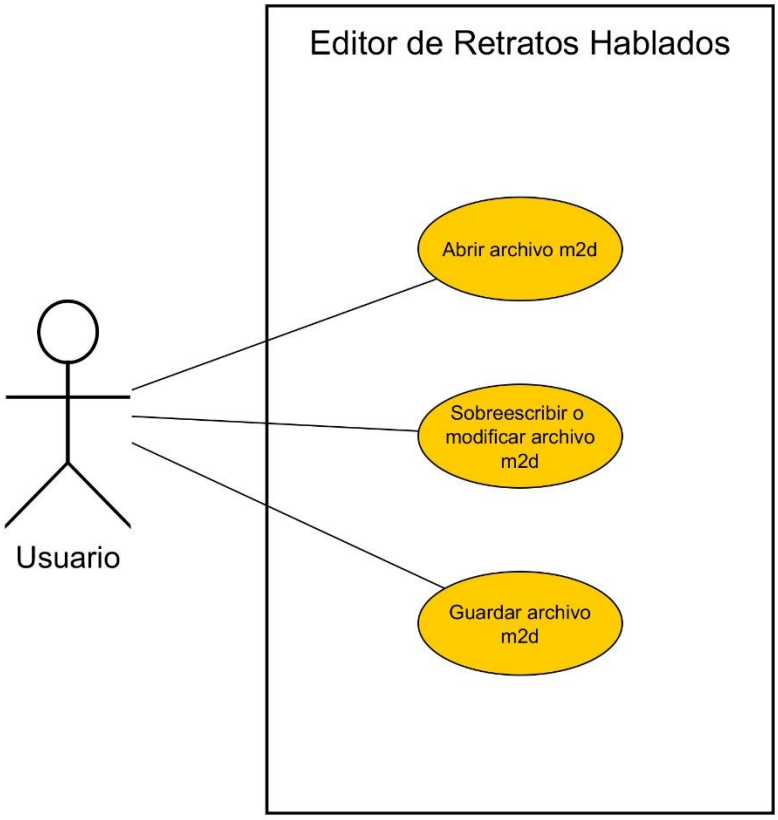


## 9.2 Diagramas de caso de uso

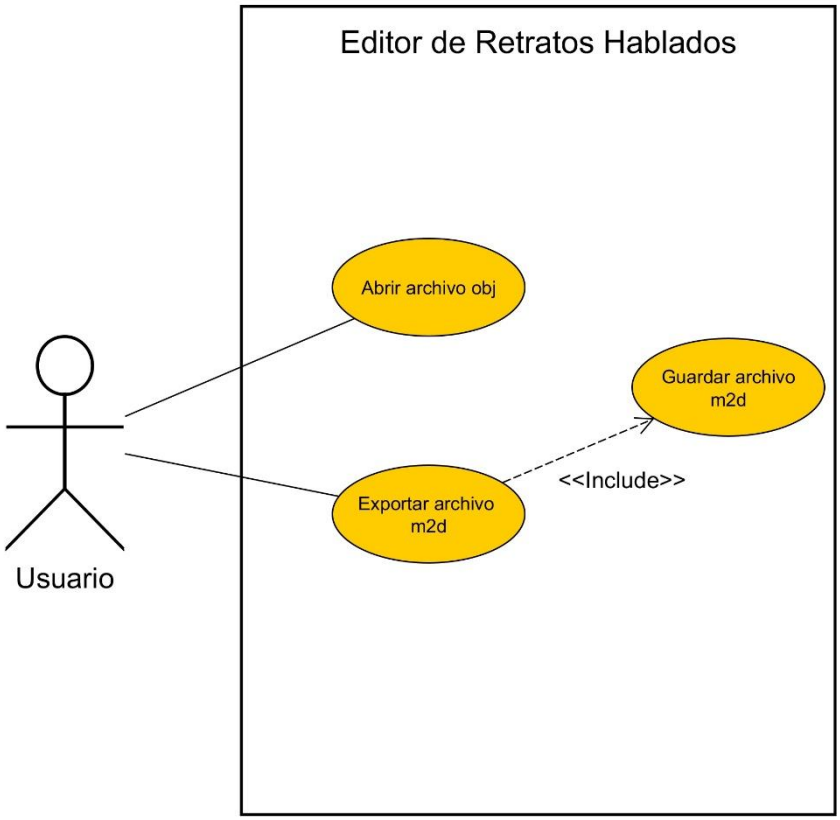
A continuación, se presentan los diagramas de casos de uso que grafican los requerimientos funcionales del software.



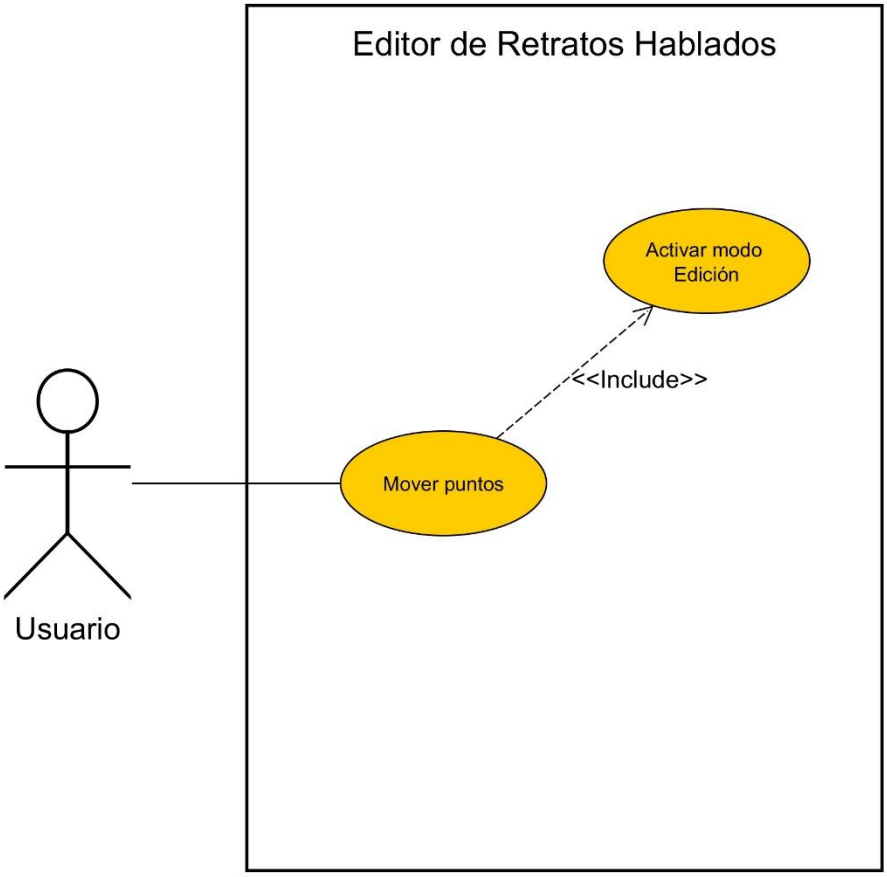
| | |
|---|---|
| **Caso de Uso** | Crear Retrato Hablado |
| **Actores** | Usuario |
| **Objetivo** | Crear por medio del software el retrato de una persona |
| **Precondiciones** | Modelo inicial cargado |
| **Postcondiciones** | Terminar el retrato y proceder a guardarlo. |
| **Descripción** | Por medio del Software editor de retratos, conseguir retratar el rostro de una persona |



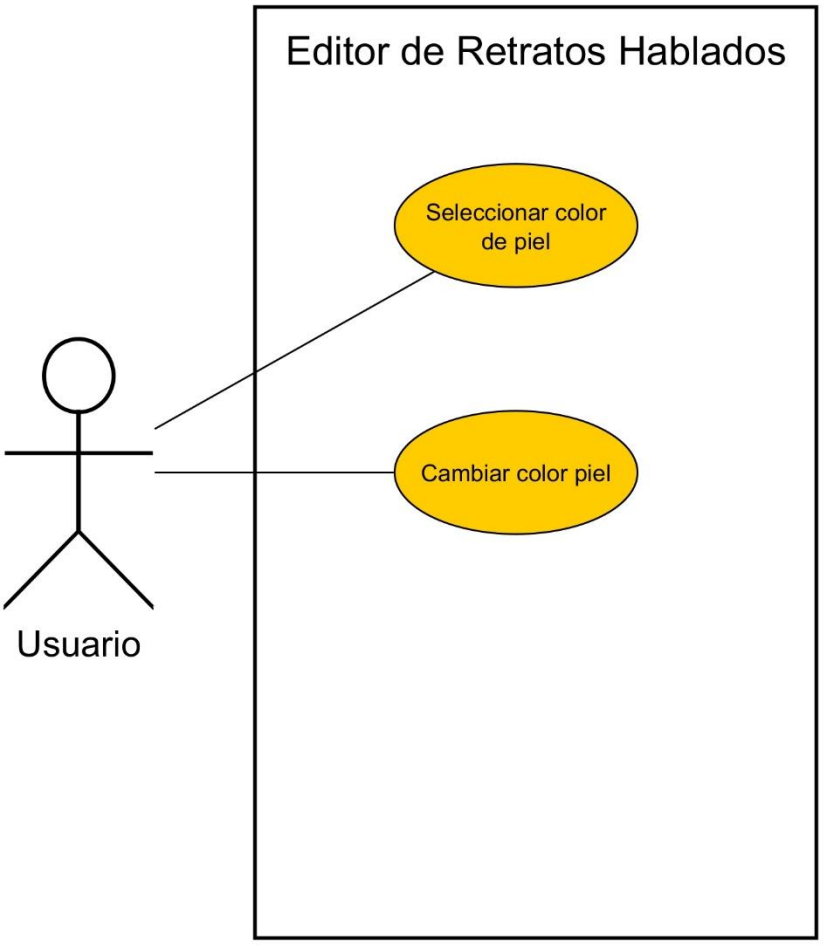

| | |
|---|---|
| **Caso de Uso** | Guardar archivo m2d |
| **Actores** | Usuario |
| **Objetivo** | Guardar el archivo cargado |
| **Precondiciones** | Archivo con formato m2d abierto |
| **Postcondiciones** | Archivo m2d guardado |
| **Descripción** | Se procede a guardar un archivo, sobrescribiendo el que se ha cargado. |



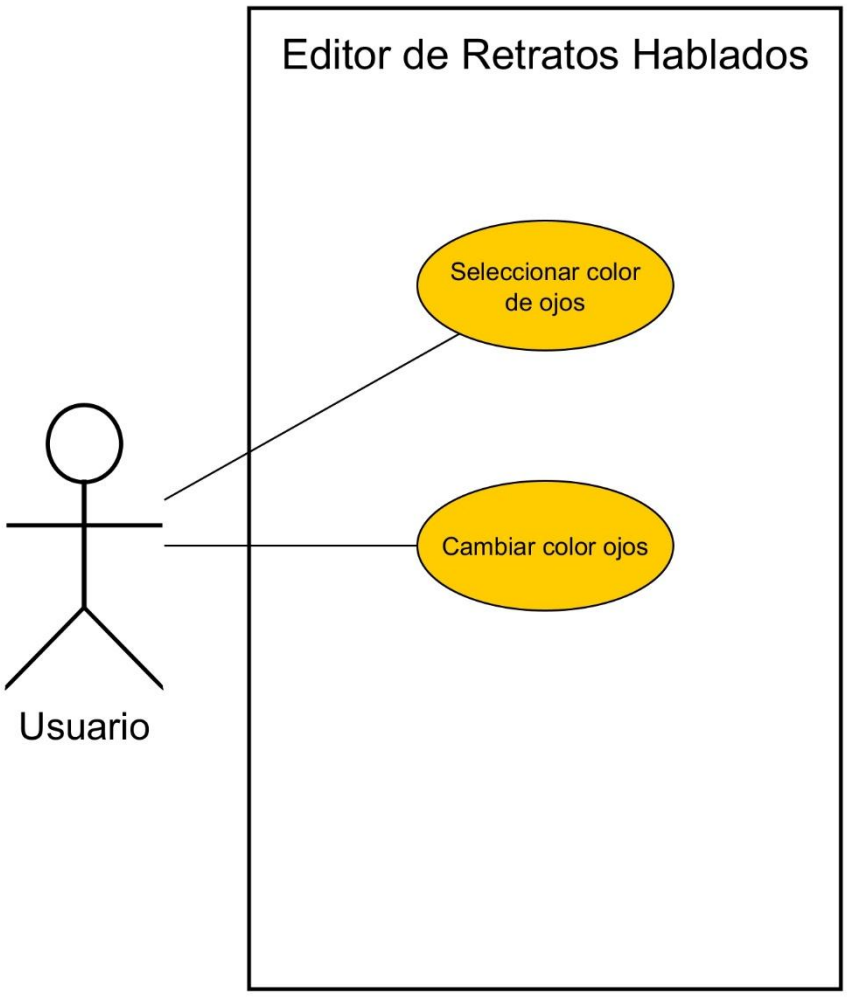

| | |
|---|---|
| **Caso de Uso** | Exportar archivo m2d |
| **Actores** | Usuario |
| **Objetivo** | Cambiar el formato de un archivo obj a un formato m2d |
| **Precondiciones** | Archivo con formato obj abierto |
| **Postcondiciones** | Archivo m2d guardado |
| **Descripción** | Luego de haber cargado un archivo con formato obj, se procede a exportarlo cambiando su extensión a m2d |



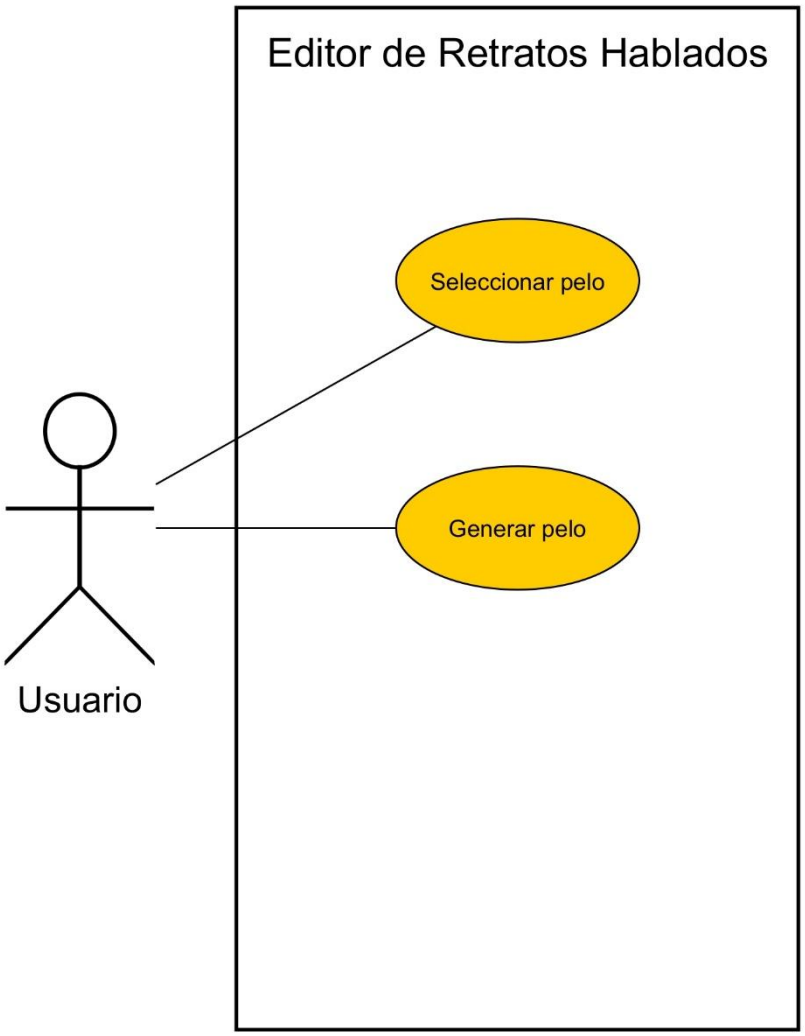

| | |
|---|---|
| **Caso de Uso** | Mover puntos de la Malla |
| **Actores** | Usuario |
| **Objetivo** | Realizar la modificación del rostro, moviendo los vértices o puntos de la malla |
| **Precondiciones** | Archivo m2d abierto y modo edición activado |
| **Postcondiciones** | No se requieren. |
| **Descripción** | Una vez abierto el archivo m2d y el modo edición activado, se podrán visualizar los vértices pertenecientes a la malla, los que podrán ser editados mediante el ratón. |



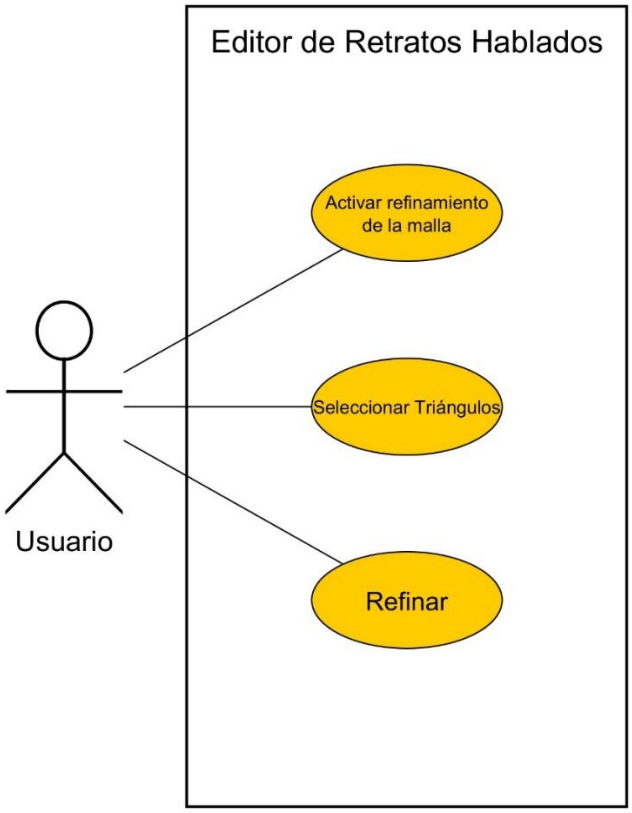

| | |
|---|---|
| **Caso de Uso** | Cambiar color de la piel |
| **Actores** | Usuario |
| **Objetivo** | Cambiar entre las distintas texturas de piel que posee el software |
| **Precondiciones** | Archivo con formato m2d abierto |
| **Postcondiciones** | No se requieren |
| **Descripción** | Luego de haber cargado un archivo, se puede hacer uso de las distintas texturas de piel. |




| | |
|---|---|
| **Caso de Uso** | Cambiar color de los ojos |
| **Actores** | Usuario |
| **Objetivo** | Cambiar entre las distintas texturas de ojos que posee el software |
| **Precondiciones** | Archivo con formato m2d abierto |
| **Postcondiciones** | No se requieren |
| **Descripción** | Luego de haber cargado un archivo, se puede hacer uso de los distintos colores de ojos. |



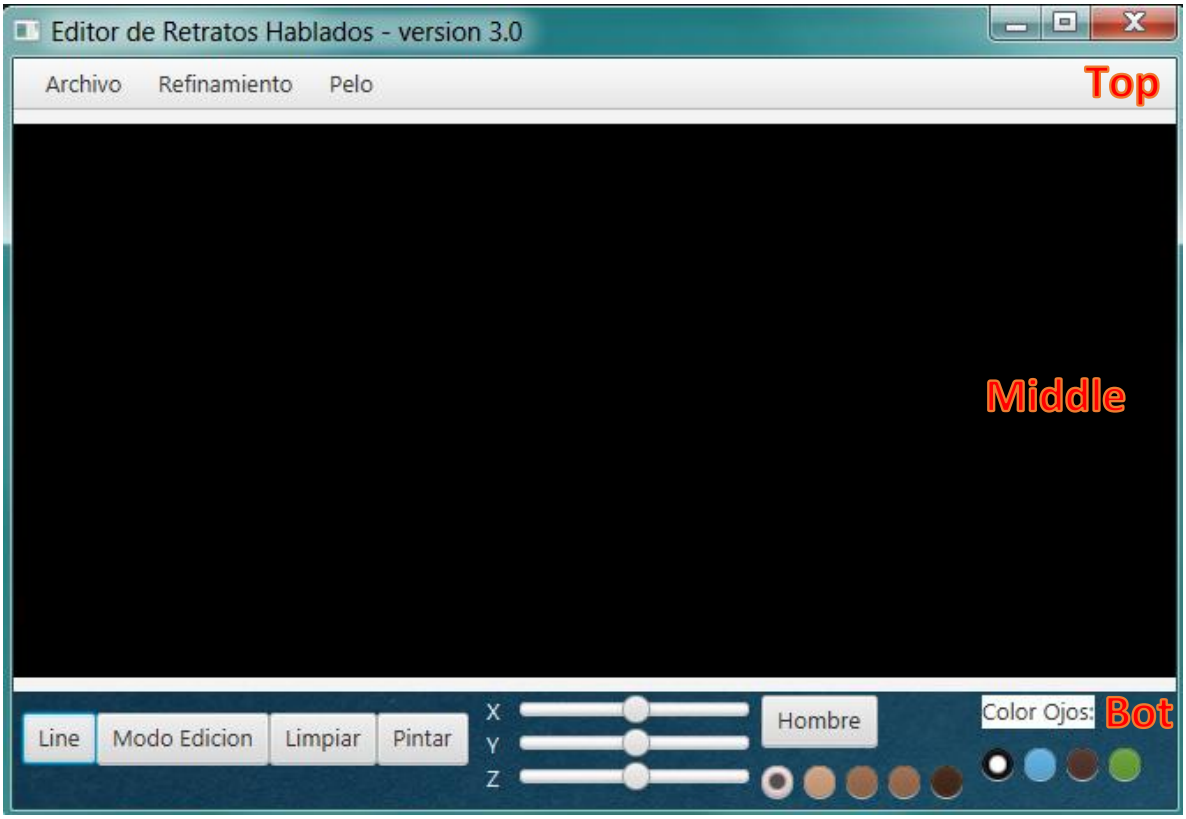

| | |
|---|---|
| **Caso de Uso** | Generar Pelo |
| **Actores** | Usuario |
| **Objetivo** | Cargar una malla que posee dos opciones de textura de cabello. |
| **Precondiciones** | Archivo con formato m2d abierto |
| **Postcondiciones** | No se requieren |
| **Descripción** | Luego de haber cargado un archivo, se puede acceder a la barra de menú para cargar dos tipos de pelo, corto o largo. |




| | |
|---|---|
| **Caso de Uso** | Activar refinamiento de la malla |
| **Actores** | Usuario |
| **Objetivo** | Generar refinamiento en zonas específicas de la malla |
| **Precondiciones** | Archivo con formato m2d abierto y seleccionar triángulos específicos de la malla. |
| **Postcondiciones** | Malla refinada |
| **Descripción** | Luego de haber seleccionado los triángulos que se quieren refinar, se procede a hacer uso del algoritmo Lepp-Bisection. |

## 9.3 Diseño de interfaz y navegación

Interfaz de Usuario: Es el medio por el cual el usuario puede comunicarse con el programa, permite realizar acciones sobre él, haciendo uso del ratón o el teclado.




La interfaz estará dividida en tres zonas principales, estas son:

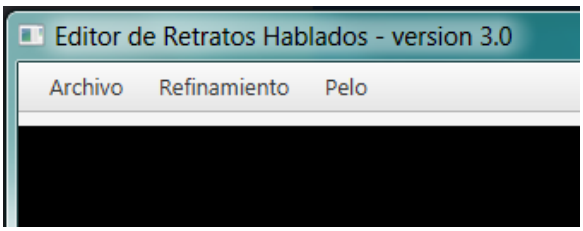
1.- Barra de menús (Top): Contiene opciones y herramientas de la aplicación.

2.- Espacio de trabajo (Middle): Contiene la parte grafica del software, en donde se muestra la malla o rostro.

3.- Barra de acciones (Bottom): Contiene botones y acciones que puede ser utilizadas para realizar modificaciones en la malla o rostro que ha sido cargado.

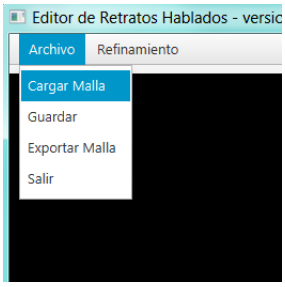
## Barra de menús:

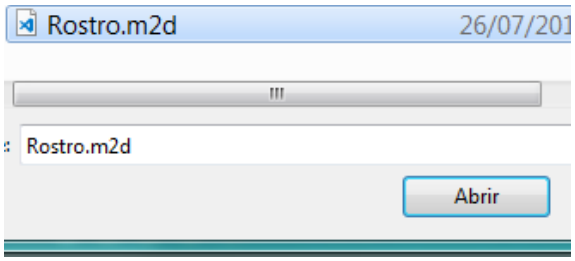
Está compuesta por 3 elementos.



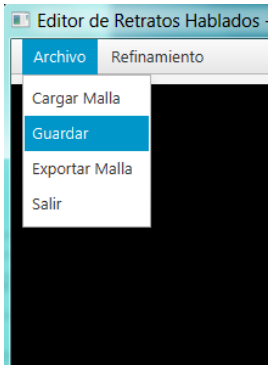

A) Archivo: Está compuesto por 4 ítems (Cargar Malla, Guardar, Exportar Malla, Salir).

a. Cargar Malla: En ella se procede a seleccionar una malla con formato obj o m2d desde la carpeta Rostros que se encuentra en el directorio del programa.

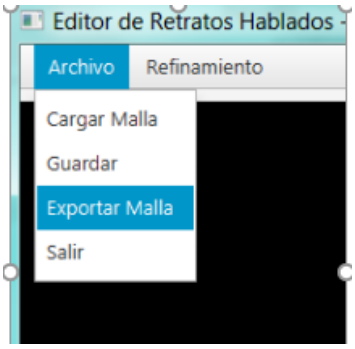


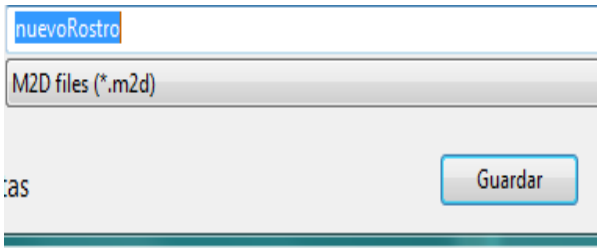




b. Guardar: Se procede a guardar un archivo que ha sido modificado o editado.



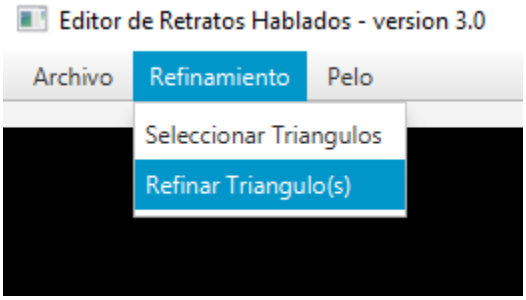
c. Exportar Malla: Esta opción permite transformar un archivo con formato obj al formato m2d.



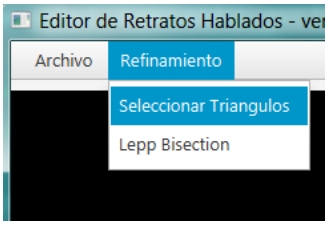



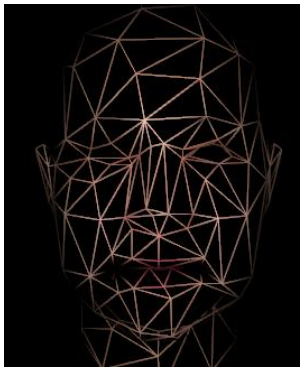

d. Salir: Permite cerrar la aplicación.

B) Refinamiento: Este menú posee los ítems que permiten hacer uso del algoritmo de refinamiento Lepp-Bisection.

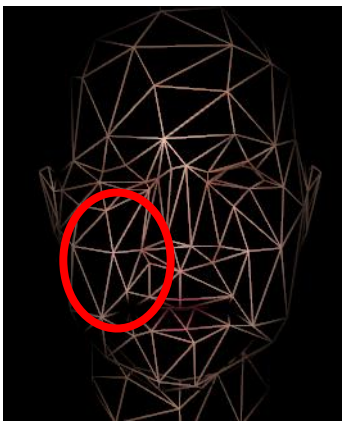


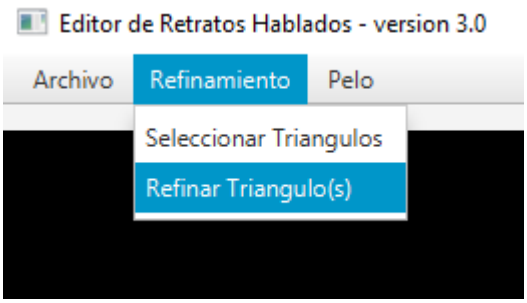

a. Seleccionar Triángulos: Permite seleccionar triángulos de un área específica de la malla, estos triángulos quedan marcados, los que luego serán utilizados por el algoritmo Lepp-Bisection.

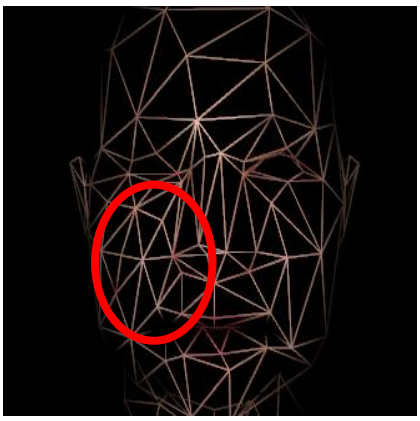


b. Lepp-Bisection: Permite realizar el refinamiento de la malla luego de haber seleccionado los triángulos.





C) Pelo: Permite carga una malla con texturas de pelo corto o largo


a. Pelo corto: Genera una malla con pelo corto.

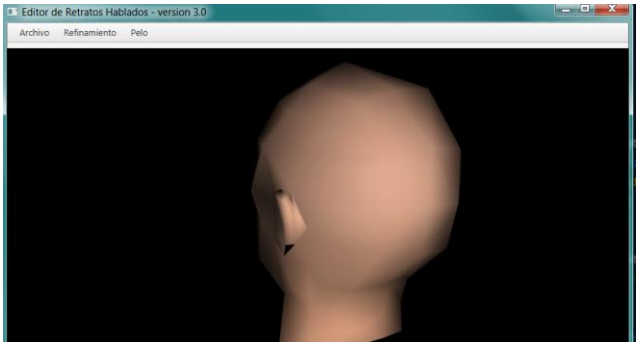
b. Pelo Largo: Genera una malla con pelo largo.

## Espacio de trabajo:

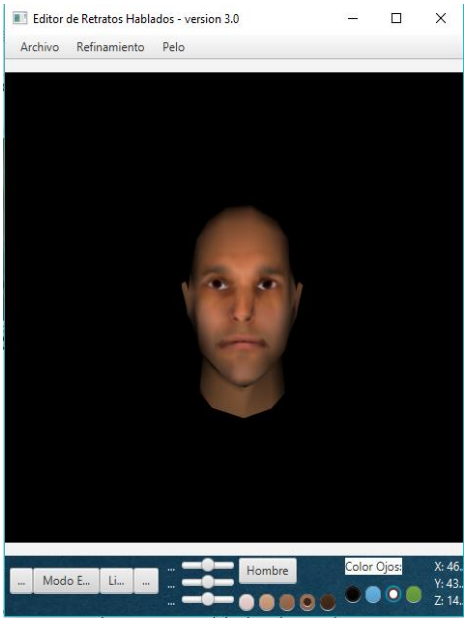
En esta área es donde se carga un archivo con formato obj o m2d, el cual crea una malla de superficie que modela distintos rostros. En esta zona es en donde se puede realizar modificaciones al rostro. Permite funciones como:

- Rotar la cabeza en todos los sentidos.

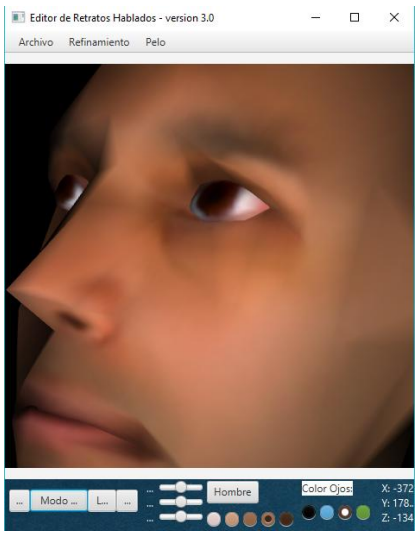



- Aleja o acercar la cabeza (Zoom +, Zoom -).

- Hacer zoom a un área específica del rostro (Ejemplo: ojo - Boca).



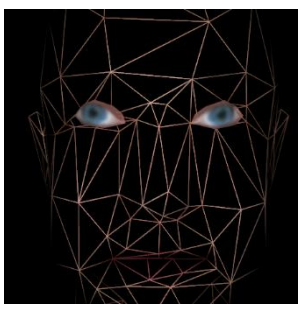


## Barra de Acciones:

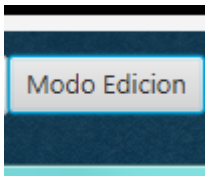
Esta zona posee botones que permiten hacer distintas acciones o alteraciones al rostro. También permite hacer modificaciones a la malla y a las texturas.

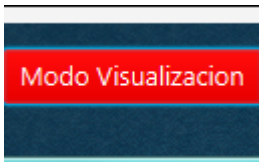

Pasaremos a explicar las funciones que poseen los principales botones del software:

- Boton Line-fill: Permite mostrar la estructura de la malla y las texturas de la malla respectivamente.



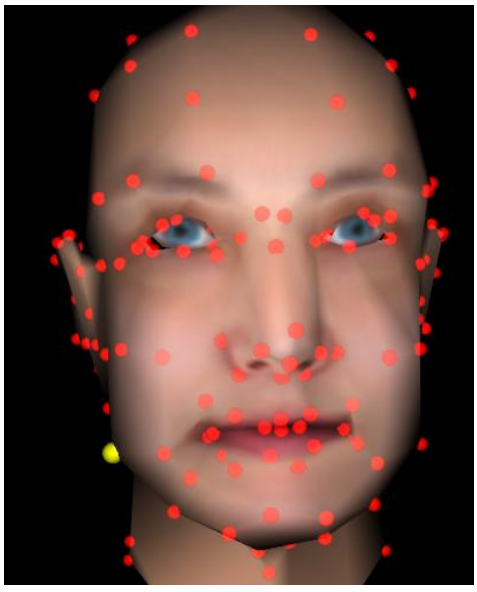
- Botón Edición/Visualización: Este botón permite activa o desactivar la funcionalidad de edición del rostro. Al activar el modo edición, aparecerán reflejado con color rojo los vértices de cada triangulo, lo que permite agarrarlo y moverlo a voluntad.

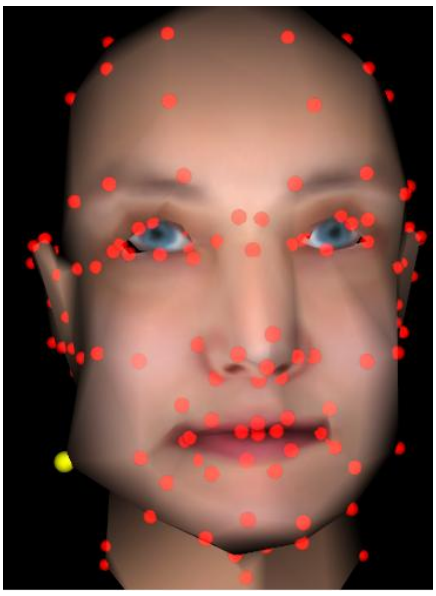





- En estas imágenes se puede ver como al pasar con el ratón por sobre uno de los vértices, este se iluminará (de color amarillo) lo que permitirá moverlo en vertical, horizontal o diagonal.





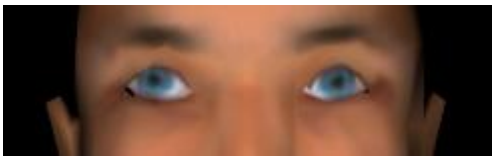
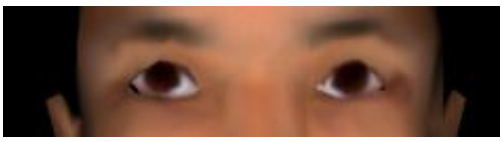
La zona Bottom también posee otro tipo de botón, a los que llamaremos Radio Button, que poseen una forma circular:

- RadioButton Color Piel: Permite cambiar la textura de la malla, este posee 5 tipos de tonalidades de piel.




- RadioButton Ojos: Permite cambiar la textura de los ojos, este posee 4 tipos de tonalidades de ojos (Negro, Azul, Café, Verde).



## 9.4 Especificación de módulos

| **N° Módulo: 01** | | **Nombre Módulo:** Obtención de nube de puntos | |
|---|---|---|---|
| **Parámetros de entrada** | | **Parámetros de salida** | |
| **Nombre:** | **Tipo dato:** | **Nombre:** | **Tipo dato:** |
| Línea de texto | String | Lista de puntos | ArrayList<Point3D> |

| **N° Módulo: 02** | | **Nombre Módulo:** Obtención de triangulación de nube de puntos | |
|---|---|---|---|
| **Parámetros de entrada** | | **Parámetros de salida** | |
| **Nombre:** | **Tipo dato:** | **Nombre:** | **Tipo dato:** |
| Línea de texto | String | Lista de triángulos | ArrayList<Triangulo> |

| **N° Módulo: 03** | | **Nombre Módulo:** Obtención de coordenadas de textura | |
|---|---|---|---|
| **Parámetros de entrada** | | **Parámetros de salida** | |
| **Nombre:** | **Tipo dato:** | **Nombre:** | **Tipo dato:** |
| Línea de texto | String | Lista de coordenadas | ArrayList<Float> |

| **N° Módulo: 04** | | **Nombre Módulo:** Generación malla superficie 3D | |
|---|---|---|---|
| **Parámetros de entrada** | | **Parámetros de salida** | |
| **Nombre:** | **Tipo dato:** | **Nombre:** | **Tipo dato:** |
| Lista puntos | ArrayList<Point3D> | Malla de triángulos 3D | TriangleMesh |
| Lista triángulos | ArrayList<Triangulo> | | |
| Lista coordenadas | ArrayList<Float> | | |

| **N° Módulo: 05** | | **Nombre Módulo:** Manipulación de vértices para transformación de rostro | |
|---|---|---|---|
| **Parámetros de entrada** | | **Parámetros de salida** | |
| **Nombre:** | **Tipo dato:** | **Nombre:** | **Tipo dato:** |
| Vértice de la malla | Ancla | | |

| **N° Módulo: 06** | | **Nombre Módulo:** Refinamiento local de la malla | |
|---|---|---|---|
| **Parámetros de entrada** | | **Parámetros de salida** | |
| **Nombre:** | **Tipo dato:** | **Nombre:** | **Tipo dato:** |
| Triangulo a refinar | Triangulo | Triangulo refinado | Triangulo |

| **N° Módulo: 07** | | **Nombre Módulo:** Exportar archivo m2d | |
|---|---|---|---|
| **Parámetros de entrada** | | **Parámetros de salida** | |
| **Nombre:** | **Tipo dato:** | **Nombre:** | **Tipo dato:** |
| Archivo obj | File | Archivo m2d | File |

# 10.- Desarrollo del software

## 10.1.- Abstracción del retrato hablado

Para poder llegar a montar un sistema que permita realizar retratos hablados hay que tener en cuenta todos los factores que influyen en el desarrollo del mismo. Primeramente, hay que realzar una abstracción de un retrato hablado, es decir, escoger los elementos que más destacan en este proceso y dejar a fuera otros factores que, aunque también aportan realismo no se pueden llevar a la práctica por diferentes cuestiones tales como tiempo, dinero, complejidad computacional, etc. Sin embargo, a la hora de realizar esta abstracción se toman un conjunto de características que permitirán más adelante visualizar un retrato hablado en 3 dimensiones con los elementos característicos del mismo. Ya se mencionó anteriormente cuales eran los elementos principales en la construcción de un retrato hablado, entonces, en este capítulo abarcaremos como se llega a construir un software capaz de elaborar un retrato hablado en tres dimensiones mediante la utilización de lenguaje java en el entorno de desarrollo Netbeans.

## 10.2.- Biblioteca javaFX

Primeramente, utilizamos la biblioteca de javaFX ya que está enfocada al desarrollo de interfaces ricas, es decir, interfaces de usuario que están cargadas de elementos con los que se puede interactuar, según Java.com, permite a los desarrolladores integrar gráficos vectoriales, animaciones y diversos elementos multimedia; a la vez influir dentro de otros objetos en tiempo real, teniendo una retroalimentación prácticamente instantánea. Esta biblioteca es usada además para aplicaciones móviles y páginas que corren en la web, aunque en nuestro caso realizaremos el desarrollo de una aplicación de escritorio.

Cabe destacar que esta biblioteca es compatible con otras bibliotecas anteriores a esta, por ende, podemos utilizar toda la potencia que ya existe sumado a la alta velocidad que ofrece javaFX y su aspecto vanguardista que brinda a las aplicaciones sobriedad y elegancia.

Antes de seguir con la descripción de la clase principal en el proyecto debemos señalar que bajo el paradigma de la biblioteca javaFx existen variados conceptos que nos van a permitir asociarnos mucho más en el desarrollo con la realidad. Por este motivo hay que entender cómo es que está diseñado javaFX.

El primero elemento característico dentro de toda aplicación javaFx es el “Escenario”, como tal un escenario es un espacio en donde se puede ubicar diferentes elementos, puede que en un lugar existan diferentes escenarios y por ende podrá haber diferentes cosas dentro de cada escenario. Este concepto viene a ser lo que comúnmente conocemos como ventanas en cualquier aplicación informática, en una ventana podemos encontrar imágenes, videos botones, barras, etc. Finalmente, el concepto de escenario es simplemente el lugar donde los actores toman protagonismo, es decir, el contexto donde se ejecutará la aplicación. Podemos ejecutar al mismo tiempo varias ventanas o escenarios cada una independiente de la otra.

El segundo elemento más importante dentro de una aplicación javaFX es la “Escena”, como se mencionó anteriormente un escenario puede tener dentro de sí mismo diversos elementos, sin embargo, es la escena la que rellena el ambiente dentro del escenario, una escena como la conocemos corresponde a un conjunto de actores que interactuaran entre sí, pueden ocurrir cosas, enviarse mensajes, obtener respuestas y generar finalmente una interacción que se puede o no visualizar. Entonces podemos decir que si queremos mostrar algo en un escenario debemos montar una escena, a su vez solo puede haber una escena en un escenario a la vez; si se quiere cambiar de escena debe terminar la primera y dar

paso a la segunda. Una escena puede contener elementos totalmente diferentes de otra y esto genera que el usuario pueda pasar de un momento a otro, a otro tipo de interacción. Un ejemplo clásico que se puede dar son los video juegos donde en una primera escena podemos tener el menú inicial en donde elegimos por ejemplo un personaje, una vez que esta seleccionado puede venir otra escena en donde ya podemos jugar con el personaje, ganar puntos, etc. Si pasamos la etapa entonces puede venir otro nivel, en ese caso vendría otra escena.

Después haber explicada los 2 elementos más importantes dentro de la creación de una aplicación javaFx debemos señalar los elementos secundarios que puede o deben ir dentro de cada escena. Es necesario que existan diferentes componentes que irán dentro de la aplicación si es que queremos que nuestro programa sea más que una ventana en blanco.

Finalmente, el tercer componente más importante para el desarrollo de una aplicación javaFX son los “Nodos” estos nos van a permitir conectar diferentes componentes que a su vez pueden estar contenidos dentro de otros nodos como una estructura de árbol. Existen nodos principales como puede ser el escenario, este nodo sería la raíz, el siguiente nodo que es la escena sería hijo de este nodo padre y para efectos de este proyecto, hablando a grandes rasgos, el retrato hablado sería el nodo hijo que está contenido dentro del escenario.

Podemos ver este agrupamiento de nodos que se da dentro la filosofía de javaFx el cual nos permite tener un mayor rendimiento en nuestra aplicación, tanto en la inserción, búsqueda o eliminación de nodos, así también una buena gestión de nuestros recursos dentro de la misma aplicación.

## 10.3.- Malla geométrica

El concepto de malla que ya conocemos permite que se puedan albergar elementos dentro para su posterior manipulación y edición; esto nos demuestra que la malla va a ser fundamental para la construcción del rostro. Podemos pensar en la malla geométrica como un esqueleto que posteriormente se va a recubrir con diferentes capas que van a dar un mayor realismo y a la vez nos va a permitir añadir una gran cantidad de características al sujeto al que se busca representar.

Por otra parte, para la creación de la malla se tomará como elemento básico el triángulo ya que al ser el polígono más simple se puede hacer uso de sus propiedades tales como el tamaño de sus lados, medianas y puntos notables; además existen relaciones con circunferencias ya sea inscrita, circunscrita, etc. Estas nos pueden ayudar a realizar diversos cálculos que nos van a permitir realizar posteriormente diferentes operaciones sobre los mismos. Esta sencilla figura hace que sea sumamente útil para crear mallas de superficie que discretizen figuras reales.

## 10.4.- Clase TriangleMesh

La clase TriangleMesh es una clase que extiende de Mesh y que permite definir una malla triangular que cuenta con diferentes componentes tales como vértices por separado, puntos, normales, coordenadas de texturas y un arreglo de caras en donde se definen los triángulos individuales de la malla.

Existen ciertas limitaciones con las que se debe lidiar a la hora de desarrollar un proyecto que contenga una malla de javaFx. Por ejemplo, la cantidad de vértices debe estar en concordancia con el número de triángulos con los que cuenta la malla, es decir, por cada triángulo deben haber 3 vértices, a su vez cada vértice debe contar con 3 coordenadas (x,y,z) para ser ubicado en un espacio de 3 dimensiones.

Otro elemento importante son las coordenadas de texturas que deben ser agregadas por obligación, aunque las mismas no se ocupen posteriormente. Dejando fuera estos elementos que vienen por defecto en la construcción misma del lenguaje, podemos decir que con estos 3 elementos básicos se puede crear una forma tan básica como compleja que queramos.

Para nuestro caso, se creó una clase que extiende de TriangleMesh, de esta forma podemos personalizar dicha clase y obtener una nueva que nos va a permitir manipular más libremente los elementos que contiene dentro. Como se mencionó anteriormente cada malla de triángulos debe contener a lo menos una serie de vértices, triángulos que estén asociados a dichos vértices y coordenadas de textura que permita darle un color base a dicha malla.

## 10.6.- Obtención de vértices

El primer paso para la creación de la malla en javaFX es ubicar los puntos en el espacio, para ellos nos podemos valer de diferentes fuentes tales como archivos que tienen nubes de puntos, o documentos que podemos encontrar en la web que ya vienen con una estructura definida. En nuestro caso nos valemos de un archivo que es importado mediante otro software llamado FaceGen el cual permite la generación de rostros aleatorios, sin embargo, existen diversos softwares y fuentes que permiten exportar mallas en formatos legibles. Posteriormente, se exporta el documento a un formato comprensible, es decir, un formato que se pueda leer por un humano. Este documento que es importado contiene diferentes características e información que nos van a ser útiles posteriormente para el manejo y edición de la malla.

Una vez que tiene la información de los puntos debemos almacenarlos en memoria para su posterior integración dentro del objeto triangleMesh.

El método que tiene la clase TriangleMesh para añadir los vertices es getPoints(), este método nos retorna un arreglo observable, es decir, una lista de elementos que

apenas sean afectados por algún eventos serán modificados. Seguido de eso debemos invocar el método addAll() para incluir definitivamente los vértices que tenemos almacenados de esta forma.

TriangleMesh malla = new TriangleMesh();

Malla.getPoints().addAll(c1,c2,c3 … cn);

Donde c1 corresponde a la coordenada X del primer vértice, c2 a la coordenada Y y c3 a la coordenada Z. De esta forma debemos mediante un bucle añadir todos los puntos respectivamente.

## 10.7.- Obtención de las caras

Luego de que se han guardado en memoria los vértices y se han añadido a la malla se deben cargar las caras o triángulos, sin embargo, en parte es donde más se debe tener cuidado ya que cada triángulo dentro de la malla contendrá las referencias a 3 vértices, por ende, debemos dentro de cada triángulo guardar el identificador de los vértices por los que están compuesto cada triángulo.

Donde cada vértice contiene dentro de sí su ubicación en el espacio y además una referencia en memoria. Posteriormente se debe crear un triángulo que contendrá dichas referencias de la ubicación espacial de los vértices.

De esta forma se debe hacer para todos los triángulos que queramos añadir a la malla.

```
//pseudocódigo
Vértice v1 = new Vértice (25, 60, 90);
Vértice v2 = new Vértice (-45, 61, 88);
Vértice v3 = new Vértice (12, 30, 105);
Triangulo T= new Triangulo (v1, v2, v3);
```

En el ejemplo podemos ver la creación de 3 nuevos vértices con sus respectivas coordenadas (x,y,z), luego de la creación de cada uno de los vértices se procese a instanciar un objeto de la clase triangulo que contiene información de sus 3 vértices. Es importante señalar que dado que el lenguaje Java está orientado a la representación de objetos del mundo real es necesario o importante distinguir dentro del código cada elemento por separado, es decir, podemos crear solamente un triángulo que contenga información de sus 3 vértices sin crear objetos de la clase vértice, pero en este caso estaríamos perdiendo control a la hora de manipular los objetos por separado ya que no existiría la clase vértice. Es recomendable encapsular el código en diferentes clases y objetos para que posteriormente se pueda acceder a cada uno de ellos por separado y no sea obligatorio tener que acceder solamente a la clase triangulo para ubicar elementos que pretenden representar la ubicación se sus vértices en el espacio.

Si bien en un principio puede parecer que no es necesario separar esta información veremos que posteriormente nos servirá de mucha ayuda para recorrer la malla y gestionar cada uno de los elementos que la componen.

## 10.8.- Pintado de la malla

Antes de seguir adelante en manipulación de la malla se necesita un elemento que aportará características especiales y que a priori distinguen este tipo de retratos hablados de los clásicos realizados a la mano o en softwares dedicados al modelado de retratos hablados, ya sea en 2D o 3D. Hablamos de otorgarle color a la malla para que esta pueda parecer más real y a la vez pueda tomar diferentes tipos de coloraciones y que en conjunto cada uno de los triángulos aporte un color a la malla; de esta forma podemos aportar aún más realismos a la misma y que la persona sienta el real parecido a quien está describiendo.

Por esta razón el pintado de la malla se realiza mediante la colocación de diferentes vértices que unidos dan el color final a cada una de las caras de la malla triangular.

## 10.9.- Sombreado de Phong

En honor a “Bui Tuong Phong” quien desarrolló su tesis doctoral en la universidad de Utah y propuso este modelo de sombreado. Este método se ha utilizado bastante en la actualidad para el uso de aplicaciones de renderización

El sombreado de phong se utiliza para suavizar las uniones de nuestras caras, este corresponde a un efecto visual que evita que sepamos donde termina cada cara. A diferencia de otros efectos aplicables tales como el “Hyper Nurbs” este no divide nuestra malla, ni crea más elementos dentro de la misma.

De esta forma podemos obtener un sombreado más realista con ciertos toques de iluminación y evitar que se muestren los bordes de las caras mediante un procedimiento que se llama interpolación.

La interpolación corresponde a la capacidad de generar un número a partir de 2 valores establecidos, de cierta forma es una aproximación que se puede dar para un nuevo valor a partir de otros 2 elementos ya existentes. Así se puede generar un color o textura que no existe a partir de la información de cada vértice dentro de la malla. Sabemos que la malla está compuesta por triángulos y a su vez cada triángulo contiene 3 vértices, estos poseen información del color que representan. Al estar relacionados con un material que implementa el sombreado de phong estos intercambian sus valores para generar un color que es de cierta forma la aproximación de los 3 ya mencionados, de esto se encarga a alto nivel el material de phong de javaFX. De esta manera logramos conseguir una textura que contiene colores y/o tonalidades de forma difuminada y a la vez posee detalles de iluminación que varían según el lugar en que haga contacto la fuente de luz con la superficie de la malla.

La forma en que se relacionan estos términos tiene que ver primeramente con la utilización de un recurso que para este caso será una imagen, de esta forma podemos cubrir una superficie 3D con un material. Este marial posee ciertas características como sombra, iluminación, ciertos efectos que estarán dados por la imagen que se añade a dicho material y de esta forma podemos utilizar dicho

material para recubrir finalmente la superficie de esta malla en 3 dimensiones. A grandes rasgos ese es el camino para poder lograr que un cuerpo en 3 dimensiones tenga sobre sí un acabado realista y acorde a los gráficos computacionales que se utilizan en la actualidad.

## 10.10.- Materiales

En este punto las cosas pueden tomar 2 rumbos. En primera instancia podemos agregar las coordenadas de textura a la malla para que esta quede de un color base, que por defecto es blanco. Se puede realizar de la siguiente forma:

*Malla.getTexCoords().addAll(0,0);*

De esta manera podemos evitar tener que cargar cada uno de las coordenadas para la malla.

Sin embargo, esto supone un inconveniente ya que lo que se busca es el mayor realismo posible dentro de los límites que tiene el proyecto y el evitar añadir coordenadas de textura resta realismo, por esta razón es que debemos tomar el camino largo y añadir las texturas.

Aunque en principio las coordenadas de textura no son algo tan importante para la conformación de la malla, sí nos van a otorgar un realismo extra pues nos permiten simular una textura en una superficie que es plana, de esta forma podemos ganar como se mencionó anteriormente, realismo pero también el componente del rendimiento se ve afectado pues si quisiéramos realizar este tipo de textura manualmente en 3 dimensiones necesitaríamos demasiados recursos computacionales y como se mencionó al principio, en la realización de un retrato hablado buscamos abstraernos de los elementos más importantes balanceando el rendimiento de la aplicación pero también la parte visual. Por este motivo creemos que el implementar coordenadas de textura supone una vía algo tediosa pero que finalmente contribuirá de manera positiva.

Si bien ya tenemos material que puede recubrir una forma 3D necesitamos ubicar dicho material que está en 2 dimensiones sobre la superficie de este objeto. De forma contraria solo tendríamos una imagen que posee ciertas características pero que no es capaz de adaptarse a la forma que se le está asignando. Por este motivo es que se pone a disposición el mapeado de texturas el cual nos permite distribuir dicha imagen o material de forma equitativa sobre la superficie de la forma 3D y hacer que esta ahora tenga una textura.

## 10.11.- Mapeado de texturas

El mapeo o distribución de texturas corresponde a la distribución de la imagen sobre una superficie. Se requieren ciertos valores para poder distribuir esta imagen sobre la forma 3D. De estos hablaremos a continuación.

## 10.12.- Agregado de texturas

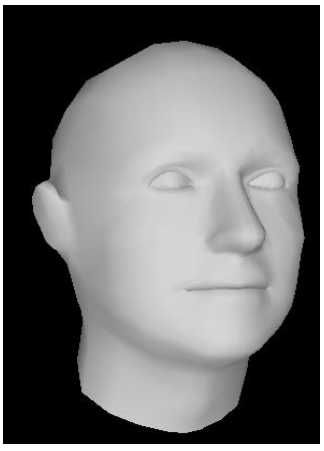
Primeramente, es necesario definir lo que son las coordenadas de textura o el mapeado de texturas.

Estas se pueden entender de la misma forma en la que alguien envuelve un regalo. Existe una caja o un objeto que tiene un alto, un ancho y un largo. De esta forma ponemos sobre si una capa que corresponde al envoltorio o papel de regalo. Si bien el papel también tiene 3 dimensiones podemos ignorar por un momento su grosor y centrarnos solamente en el largo y ancho, este papel tiene un diseño que puede repartirse a lo largo del objeto que queremos cubrir, este diseño se conoce como motivo o textura. Cada envoltorio tiene un tipo de diseño, de nosotros depende elegir con qué vamos a cubrir la forma 3D. Lo importante acá es que este envoltorio será puesto sobre la superficie del objeto, no dentro, ni abajo, ni a los lados sino cubriendo cada parte de la superficie del mismo. En este caso el papel de regalo corresponde a una imagen que pretende representar una textura como puede ser madera, cabello, piel, agua, metal, etc. Existen muchas texturas dentro de la naturaleza y dependerá del objeto que queramos representar.

Las coordenadas de texturas o el mapeado de texturas son como bien dice su nombre, son mapas, que permiten ubicar una imagen que está representada en dos dimensiones sobre la superficie de un objeto que está en tres dimensiones. El mapa en si cuenta con coordenadas que nos van a permitir disponer cada pixel de la imagen sobre la superficie de un cuerpo de tres dimensiones.

Podemos agregar que para este tipo de texturizado se utiliza la forma de mapeado UV, el cual es una forma de asignar la parte de una imagen a un polígono en el modelo 3D que se tiene. Cada vértice del polígono es asignado a un par de coordenadas 2D que definen qué parte de la imagen es mapeada. Estas coordenadas 2D se nombran como UVs para distinguirse de las coordenadas xyz del objeto 3D.

Entenderemos textura como el concepto de dar a una superficie plana un aspecto diferente. Si bien la superficie no cambiará, sí dará la impresión de que ahora tiene algún relieve, también podemos encontrar texturas que aparenten vellosidad, alguna superficie maderosa, etc. Todo dependerá de la imagen que estemos utilizando para dar textura al objeto en cuestión.



a)

*Fig 2.1: Muestras malla de superficie sin texturas*

b)

*Fig 2.2: Muestras malla de superficie con texturas*

Un tipo de mapeo en el mapeo difuso el cual tiene la peculiaridad de asignar una coordenada para cada vértice del objeto que más adelante se va a interponer.

## 10.13.- Mapa Difuso

En la utilización de la librería javaFX manejamos para el texturizado o tipo de mapeo de textura que es el difuso. Este tipo de mapa nos permite asignar a cada vértice una coordenada que se traduce posteriormente a la ubicación de la imagen en 2 dimensiones, por este motivo cada triángulo contará, de cierta forma, con 3 colores o secciones de la imagen que se desea poner como textura, esto quiere decir que cuando se realice el renderizado de la textura, el triángulo que contiene los 3 vértices y a su vez cada vértice las 3 textura necesitará realizar una mezcla de estos 3 colores y por esta razón es que se otorga un color que no es ninguno de los 3 colores que posee cada vértice sino **un color difuso** que permite combinar cada una de las tonalidades que posee cada vértice para mostrar en última instancia una textura realista y que no permite ver cortes o diferencias de coloración entre una cara y otra.

Hay que señalar que este tipo de mapeo es solo una de las cuantas formas en las que se puede añadir una imagen a una superficie o cuerpo 3D.

## 10.14.- Clase MeshView

Una vez que ya tenemos nuestra malla en 3 dimensiones con cada una de sus caras y estas con cada trio de vértices es necesario la utilización de otra clase que nos permite mostrar esta malla ya que la clase triangleMesh por sí sola no puede mostrarse y requiere ser añadida a un objeto nodo de escena gráfica visible.

Esta clase no solo nos permite añadir y mostrar mallas de triángulos sino también personalizar elementos como la luz, el relleno, el material que utilizará y otras características más.

Tal como lo define Wallace Jacson en “pro java 9 games development” la clase meshView es usada para definir una superficie 3D utilizando un modelo de datos contenido en un objeto malla.

Esta clase que extiende de “Shape3D” nos permite agregar dentro de sí un material que posteriormente será mostrado sobre la figura 3D que estemos utilizando. De esta manera la clase triangleMesh proporciona la base para que posteriormente MeshView utilice toda la información contenida dentro de la malla y se pueda agregar un tipo de material.

El material corresponde a un recurso computacional que puede ser un color del tipo RGB o también puede corresponder a una imagen que hayamos seleccionado previamente.

Para fines de este proyecto utilizamos una imagen que contiene la textura de la piel, así como también la textura del cabello y los ojos de la persona. Es necesario utilizar imágenes en formatos jpg o png para la asignación del recurso a la clase meshView.

De esta forma separamos lo que se ve, de los datos necesarios para la visualización de dicho objeto. Es muy común que utilicen siempre capas al mostrar un objeto ya que no queremos que estos estén acoplados, es decir, que podamos mantener los datos separados de lo que ve el usuario tal como se podría usar en la arquitectura modelo-vista.

## 10.15.- Visualización de la malla

Luego que hemos puesto a disposición de la clase meshView un conjunto de estructuras de datos necesarios para su posterior visualización, estamos en tierra derecha para la manipulación de esta malla y la posterior incorporación de eventos

para poder mediante el uso del teclado y el ratón mover dicha malla y también realizar cambios en esta.

Ya que conocemos los conceptos básicos de javaFx tales como el escenario, la escena y los gráficos estamos en posición de avanzar a la siguiente etapa que es la visualización de dicho gráfico que en este caso será el objeto instanciado a partir de la clase meshView. Este será el gráfico que se mostrará dentro de la escena.

Una forma simplificada en la que podemos representar la incorporación de dicho gráfico en la escena es la siguiente.

```
ClaseEscenacio escenario = new ClaseEscenario();
ClaseEscena escena = new ClaseEscena();
Escenario.setScene(escena);
TriangleMesh mallaTriangulos = new TriangleMesh(Vertices, Triangulos, coordenadasText);
Meshview vistaMalla = new MeshView(MallaTriangulos);
escena.getChildren().addAll(vistaMalla);
```

De esta forma podemos incorporar dentro del contexto en donde estamos trabajando un objeto capaz de mostrar una malla en 3 dimensiones. A la vez podemos mediante este método añadir diferentes mallas u objetos tan sencillos como pirámides o cubos hasta figuras más elaboradas como en este caso un rostro que cuenta con casi 400 triángulos y alrededor de 200 vértices.

## 10.16.- Visualización dinámica

Entendemos por visualización dinámica una forma en la que se puede ver un cuerpo en 3 dimensiones de diferentes ángulos y distancias. Entre las características que se requieren para poder visualizar de forma dinámica un objeto dentro del entorno que hemos definido encontramos 2 elementos importantes. El primero de ellos es la rotación y el segundo es la distancia con la que se puede observar dicha forma desde un punto determinado.

Acá entra el concepto de cámara, el cual nos va a permitir visualizar este objeto en el espacio de una forma tal como se haría dentro de un escenario. Al igual que en el mundo real, cuando queremos visualizar dentro de una escena un objeto o un conjunto de ellos y a la vez queremos mostrar esto a los espectadores que no están dentro del escenario sino en otro lugar necesitamos una cámara que nos permite visualizar dicha información y posteriormente entregarla a estas espectadores que requieren la imagen, de esta forma existe una cámara que está ubicada en algún lugar del espacio, dentro de la escena, y está dirigida al objeto que queremos visualizar. Posteriormente es el usuario quien puede visualizar el contenido que está observando dicha cámara.

Es necesario trabajar de esta manera cuando se tiene un conjunto de elementos dispuestos dentro de una escena y se necesita observar a todos ellos al mismo tiempo, por esta razón es que se emplea una cámara que permite en tiempo real realizar rotaciones y traslaciones para efecto de mejorar la experiencia de usuario y a la vez permitirle al mismo manipular dicha cámara para moverse dentro de la escena. Es un concepto que se asemeja mucho a la realidad y que permite otorgar el usuario control y a la vez eficiencia en la disposición de los recursos dentro de la escena, ya que no se necesita visualizar cada uno de los elementos por separado, sino que todos ellos son englobados por un solo objeto cámara.

## 10.17.- Clase Camera

Desde un punto de vista más técnico la clase Camera, que extiende de la clase Node, permite renderizar una escena dentro de las coordenadas de un espacio que a su vez está contenido dentro de un escenario. Gracias a esta cámara podemos desplazarnos dentro de la escena de diferentes maneras, ya sea rotando la cámara o también trasladándola a cualquier posición. Estos últimos 2 puntos son realmente importantes ya que de ellos depende la visualización dinámica de la que hemos estado hablando.

Antes de hablar del concepto de rotación o traslación de cámara debemos señalar que al igual que en el mundo real una cámara debe estar dirigida a algún lugar. Existe dentro de este universo de 3 dimensiones muchos lugares en donde podemos ubicar la cámara, pero también es necesario conocer la ubicación de objeto que queremos observar. Por esta razón es que se llega al consenso de ubicar el centro de la forma 3D en la posición 0,0,0, donde cara uno de los puntos corresponde a las coordenadas x,y,z respectivamente. Esto no implica que los puntos o vértices comenzarán en esa posición, sino que el objeto en sí rodeará de manera equitativa dicho punto. En otras palabras, la malla 3D estará alineada perfectamente con la coordenada central 0,0,0.

En primer lugar, la rotación implica un cambio en el ángulo en el que está dispuesta la cámara, es decir debemos modificar las coordenadas x e y de la ubicación de la cámara para poder cambiar el ángulo en el que ésta enfoca la imagen. Si bien se puede pensar que es posible modificar la malla para que esta cambie de posición, es mucho más eficiente el enfoque de la cámara ya que si pretendemos rotar la forma 3D y mantener la cámara mirando fijamente debemos trasladar cada uno de los puntos, y esto implicaría una mayor cantidad de tiempo en el procesamiento de cada uno de los nodos, sobre todo en mallas que contienen un gran número de vértices. Por esta razón es que tomamos el primer enfoque el cual nos permite manipular solamente un nodo, que sería la cámara y dejar libre la malla.

En segundo punto importante para poder visualizar de forma dinámica una malla 3D en el concepto de traslación de cámara o “zoom” el cual nos permite acércanos o alejarnos. Esto otorga facilidades al usuario mediante algún gesto o tecla que gatilla un evento que va a trasladar finalmente dicha cámara. En cuanto al concepto de traslación de cámara debemos tener en cuenta las coordenadas que rigen a la misma. Al estar dentro de un entorno 3D distinguimos 3 coordenadas, x, y, z. Ya que utilizamos las coordenadas x e y para realizar una rotación en la cámara la última coordenada que nos queda es la Z y esta nos permitirá realizar un acercamiento o alejarnos dependiendo de nuestras necesidades. Es importante poder desplazar la cámara o hacer un zoom in o zoom out ya que posteriormente precisaremos modificar algunos puntos y a la vez tener una visión general de cómo está quedando la forma 3D que estamos alterando. Aunque el punto de la modificación de la malla se hablará posteriormente.

Una vez que podemos visualizar de forma dinámica la figura 3D podemos sentirnos libres para realizar gestor dentro de la escena como rotar en cualquiera de los ejes o también desplazarnos en la coordenada Z para acercarnos o alejarnos.

## 10.18.- Traslación de nodos

Ya que estamos implementando una aplicación bajo la filosofía de la biblioteca javaFX debemos ceñirnos al concepto básico de nodo. El nodo corresponde a una forma abstracta en la que se puede representar un elemento dentro de una escena. Dejando fuera el escenario por un momento nos centraremos solo en la escena; esta está compuesta de diferentes elementos tales como controles, diseños, etc. Sin embargo, cada uno de estos corresponde a un nodo, estos pueden a su vez contener más elementos dentro que pueden ser llamados hijos. Tanto los hijos como los padres pueden contener más hijos y es de esta forma aprovechamos la capacidad que nos otorga java para poder almacenar dentro de la escena muchos nodos que llamaremos “anclas”. El ancla corresponderá a una representación

abstracta de un punto en el espacio, que contiene 3 dimensiones, pero no pertenece a la malla. En principio las anclas estarán ubicadas en la misma posición de cada uno de los vértices, sin embargo, cada ancla utiliza el vértice que se le asigna para posteriormente mediante el uso del mouse mover dicha ancla y trasladar el vértice a otra posición, de esta forma utilizamos como un elemento externo el concepto de ancla para la manipulación de los vértices.

## 10.19.- Clase Ancla

Esta es una clase que extiende de esfera, de la biblioteca javaFX y por ser un tipo de gráfico se puede agregar libremente a la escena.

Un ancla es una esfera que está por sobre la malla y contiene información de la posición de un vértice, es más, cada ancla está creada a partir de la referencia en memoria de cada vértice. Por esta razón cuando se instancia un ancla se toma como parámetro uno de los vértices de la malla. De esta forma se busca que cuando se realice una traslación, en cualquiera de los ejes, a una de estas anclas también se vea afectado el vértice por el cual está creada dicha ancla. En un principio puede parecer simple el concepto de ligar un ancla a un vértice de la malla; sin embargo, solo es posible dicho procedimiento mediante la nueva versión de java que incorpora un concepto llamado “Binding” o “enlazamiento”: esto nos permite que cuando realicemos un cambio en la posición de un ancla, instantáneamente se modifique la posición en el espacio del vértice con el cual dicha ancla está entrelazada.

## 10.20.- Enlazamiento (Binding)

Se puede realizar 2 tipos de enlazamiento en java. El primero corresponde a un entrelazamiento simple o de una sola vía, el cual nos permite modificar un elemento a partir de la información de otro. El segundo tipo de enlazamiento corresponde al de doble vía el cual nos permite la modificación de un elemento A a partir de la información de un elemento B pero también de forma contraria, es decir, realizar cambios a un objeto B a partir de un elemento A. Para nuestro caso solo necesitamos el enlazamiento simple o de una vía ya que será el elemento ancla el que afecte posteriormente el elemento vértice.

## 10.21.- Clase Observable

No podemos seguir adelante sin hacer mención del concepto “Observable”. Este concepto es fundamentar en el desarrollo de aplicaciones que están bajo el paradigma del modelo-vista ya sea de forma explícita o implícita. No necesariamente toda la aplicación debe estar bajo esta arquitectura, sin embargo, existen ciertas clases que necesitan tener un protocolo de comunicación en el que se distinga claramente cuáles son los datos con los que se está trabajando y cuál es la representación de esos datos en la escena. Separar correctamente estos 2 elementos nos va a proporcionar mayor rendimiento en nuestra aplicación y también una respuesta visualmente rápida y atractiva para el usuario final.

Entonces el concepto de observable viene dado por un objeto que puede tener uno o más observadores, es decir, otros objetos que están atentos a cualquier cambio que se realice a un elemento con cual que estén vinculados. Por este motivo el concepto de enlazamiento y observable van de la mano puesto que en primer lugar tenemos un elemento A, que, en este caso, es un vértice que corresponde a la malla, y en segundo lugar tenemos un elemento B que es un ancla, estos 2 están enlazados y es la clase Ancla la que está siendo observada por el elemento vértice. Cada vez que se realice un cambio en un ancla, el vértice al cual está enlazado

dicha ancla, será notificado del cambio y además tendrá información de la posición anterior al cambio y la nueva posición. De esta forma cada vez que un ancla sea trasladada en el espacio el vértice al cual está enlazado será notificado y obtendrá, seguido de ello, una nueva posición que será la nueva posición que tiene el ancla después del cambio.

## 10.22.- Traslación de anclas y trasferencia de información

A continuación, explicaremos la manera en la que mediante código java podemos realizar la traslación de nodos en diferentes ejes a partir de los movimientos que se realizan al arrastrar y soltar el mouse en diferentes posiciones de la cara.

Para poder modificar los vértices que están contenidos en cada triángulo dentro de la malla, añadimos en la posición de cada vértice un objeto ancla que se representa con una esfera; a esta la dotamos de diferentes característica y eventos tales como el cliquear, arrastrar y soltar, además de pasar por encima con el mouse. De esta forma podemos entrarle información al ancla mediante la posición del mouse y los botones con los que presionamos sobre ella.

La clase ancla como se mencionó anteriormente extiende de la clase “Sphere” la cual permite ser añadida a una escena. De esta forma podemos instanciar diferentes anclas dentro de la escena y a la vez visualizarlas, otorgarles un tamaño, una posición en el espacio con sus correspondientes coordenadas x, y, z. un color y un vértice al cual estarán enlazadas.

Para realizar en enlazamiento entre las anclas y los vértices es que hemos dispuesto de un método que recibe como parámetro a la lista de vértices por los que está compuesto el objeto triangleMesh.

Si bien los vértices son un conjunto de 3 coordenadas que están representadas por el tipo de dato “float” la única forma de poder crear elementos que sean modificados a partir de la información de otro elemento es a través de la utilización de un “arreglo observable”. De esta manera el objeto triangleMesh, al cual desde ahora

llamaremos “Malla Rostro”, nos permite mediante el método getPoints(); obtener un arreglo de puntos que están en formato de arreglo observable, es decir, una lista de elementos flotantes que están atentos a cualquier cambio que se realice en la malla.

## 10.23.- Propiedades (Properties)

Las propiedades corresponden a una clase que envuelve o encapsula un tipo de dato, permitiéndole añadir funcionalidades extras como la capacidad de notificar los cambios de sus valores o también permitir enlaces entre propiedades similares de diferentes objetos. De esta forma podemos por ejemplo hacer que si la variable X es igual 2 y la variable Y esta inicializada en 0 pero está enlazada a la propiedad (X)*2 entonces cuando aumente o disminuya la variable X, esta aumentará o disminuirá el doble. Podemos jugar con las propiedades para que estas nos notifiquen de cambios o podemos afectar a otros objetos que comparten propiedades del mismo tipo.

Existen propiedades de lectura/escritura o solo lectura, también existen diferentes implementaciones de propiedades para cada tipo de dato, ya sea para datos primitivos como float, int o double o también para objetos complejos. Podemos utilizar SimpleIntProperty y ReadOnlyProperty para tipos de datos enteros (int); la primera nos permite leer y escribir sobre esa propiedad, la segunda solo nos permite consultar el valor que posee dicha propiedad. A su vez podemos utilizar ObjectProperty<T> para tipos de datos complejos como objetos que son creados a partir de tipos de datos primitivos.

## 10.24.- Creación de Puntos-Ancla

Llegado este momento es donde necesitamos, una vez extraído los puntos mediante el método “getPoints();”, crear cada una de las anclas y añadir dichas esferas a la escena. Para esto nos valemos de un bucle for que recorre todo el arreglo y permite la extracción de los valores de cada vértice. El arreglo nos retorna la coordenada x,y,z de cada vértice. De esta manera cada 3 vueltas del ciclo for obtendremos un nuevo vértice. También podemos hacer que el bucle for se incremente en 3 unidades con cada ciclo que pasa y dentro de cada uno de estos obtener el elemento en la posición i, que corresponde a la coordenada X, luego i+1, que pertenece a la coordenada Y. Finalmente el elemento en la posición i+2, que pertenece a la coordenada Z, y así con todos los demás elementos que quedan en la lista, de esta forma podemos obtener todos los vértices con sus 3 coordenadas respectivamente.

No solo nos basta con obtener cada coordenada de cada vértice, sino que debemos almacenarla en algún lugar y permitir que esa información sirva de base para el punto ancla que vamos a crear. Sin embargo, no podemos simplemente obtener la coordenada que es de tipo “float” y asignársela a un punto ancla ya que si hacemos eso no podremos posteriormente alterar el vértice cuando se mueva el ancla puesto que estos valores son cierta manera estáticos y deberíamos recargar toda la malla nuevamente con los puntos alterados para ver el cambio que se ha realizado en los vértices y a continuación en la malla.

La forma en la que podemos modificar instantáneamente la información que posee un vértice mediante la traslación de un punto ancla es a través de la utilización de propiedades.

Esta es la manera en la que podemos crear un ancla a partir de la información que nos entrega la lista de elementos observable del objeto Malla rostro

Primeramente, obtenemos el valor que nos entrega la lista a ejecutar el método “get”, pasándole como parámetro el índice del valor que estamos buscando. Los

primeros 3 valores corresponderán al primer vértice, los siguientes 3 al segundo vértice y así sucesivamente.

*arregloPuntos3D.get(i));*

*/* i corresponde a la variable que está dentro del ciclo for que se está ejecutando para recorrer todos los puntos de objeto malla rostro*/*

Entonces como el tipo de datos que nos retorna el método “get()” es flotante debemos crear nuevo “envoltorio” que nos permita almacenar dicho dato y posteriormente recibir notificaciones de las modificaciones que se hagan entiempo de ejecución a dichos putos anclas.

Instanciamos 3 objetos de la clase “FloatProperty” a los cuales llamaremos xProperty, yProperty, zProperty respectivamente. Los cuáles serán posteriormente parámetros del constructor de la clase ancla, de esta forma:

```
FloatProperty xProperty = new SimpleFloatProperty(arregloPuntos3D.get(i));
FloatProperty yProperty = new SimpleFloatProperty(arregloPuntos3D.get(i + 1));
FloatProperty zProperty = new SimpleFloatProperty(arregloPuntos3D.get(i + 2));
Ancla ancla= new Ancla(Color, xProperty, yProperty, zProperty, this);
```

Dicho punto-ancla recibe un color, las 3 coordenadas de uno de los vértices a los cuales estará enlazado y finalmente “this” que es una referencia a la clase malla rostro pues es necesario posteriormente acceder a algunos atributos y métodos de dicha clase.

Para recibir dichas notificaciones se hace necesario implementar el evento que proporciona la interfaz ChangeListener<T> la cual notifica si el valor de una propiedad es cambiado, es decir, nos alerta si cualquiera de nuestras propiedades es modificada en tiempo de ejecución. Podemos gestionar dentro de este evento la manera en la que se procederá al recibir la notificación de un cambio en el valor de dicha variable, de esta forma:

```
xProperty.addListener((objeto, antiguo, nuevo) ->{
        arregloPuntos3D.set(idx, (float) valueX);
});

 yProperty.addListener(((objeto, antiguo, nuevo)-> {
        arregloPuntos3D.set(idx + 1, (float) valueY);
});
zProperty.addListener((objeto, antiguo, nuevo) -> {
        arregloPuntos3D.set(idx + 2, (float) valueZ);
});

//Este es un ejemplo de la forma en la que se añade un evento escuchador.
```

Es necesario señalar que cuando añadimos dicho evento estamos haciendo uso de la nueva funcionalidad de java8, la función lamda, la cual nos permite utilizar la flecha -> seguido de lleves { } para abrir una función anónima que se ejecutará cada vez que sea gatillado el evento por el cual dicha función fue creada, en este caso la función se ejecutará cuando detecte que cualquiera de las propiedades, ya sea, X, Y o Z han sido alteradas.

El método addListener(); requiere como parámetro un objeto que implemente la interfaz Observable, sin embargo podemos abreviar dicho método y evitar añadir el objeto acotando la información a solo 3 parámetros.

```
xProperty.addListener((obj , oldValue, newValue) -> { … });
```

El primero de los parámetros corresponde a una referencia en memoria de la propiedad que está detectando el evento. El segundo parámetro corresponde al valor anterior al cambio, o valor antiguo. Finalmente, el tercer parámetro pertenece al nuevo valor que ha tomado la propiedad de dicho objeto.

Luego que hemos añadido un escuchador a cada una de las propiedades que son instanciadas dentro del bucle for, debemos desarrollar dentro de las llaves del objeto observable las acciones que serán llevadas a cabo luego de recibir la notificación del cambio. Estas acciones serán llevadas a cabo para cada ancla individualmente afectando a sus tres propiedades (x,y,z). De esta forma logramos que cada ancla esté alerta a los cambios que pueden producirse, y al instante modificar el contenido del vértice que le fue asignado.

Cada vértice cuenta con un identificador único por el cual puede ser reconocido. Este identificador corresponde a la posición dentro de la lista observable, así, podemos valernos de este identificador para localizar posteriormente el elemento dentro de la lista, y eso es lo que hacemos. Cada vez que se recibe una notificación se busca la referencia dentro de la lista de puntos, que corresponde a la lista que es devuelta por el método “getPoints()” del objeto Malla Rostro  y se procede a modificar el valor que contiene dicha referencia, es decir, cuando se modifica cualquiera de las coordenadas de un punto-ancla entonces se recibe una notificación de la coordenada que fue modificada y se procede a tomar dicho valor y modificar la coordenada respectiva del vértice al cual está ligado dicho punto-ancla.

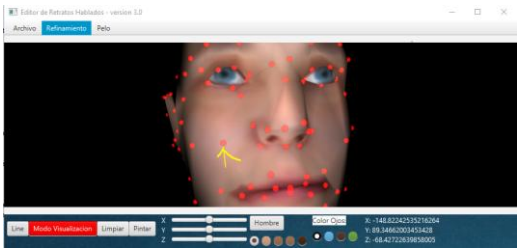


*Fig 2.3: Se selecciona un vértice a trasladar*

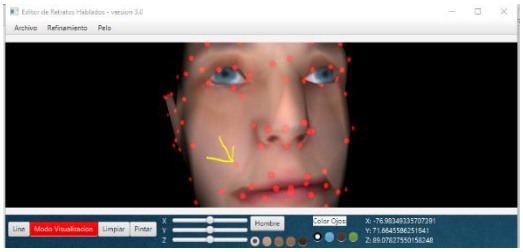


*Fig 2.4: Se traslada el vértice*

De esta manera es que se obtiene el desplazamiento de los vértices y al mismo tiempo se logra cambiar la estructura de la malla, es decir, la ubicación de los puntos que están distribuidos sobre ella, permitiendo modificar la ubicación de cualquier punto en cualquiera de sus 3 dimensiones. Podemos modificar además partes específicas del rostro como la posición de los ojos, el grosor de la nariz, la altura de

la boca, el grosor de los labios, la altura de las cejas, el ancho de los pómulos, etc. De este modo podemos llegar a moldear un rostro según las características que el entrevistado nos está dando.

## 10.25.- Importación de archivos

Una de las características principales en el software es la visualización de mallas de superficie en 3 dimensiones, y dado que sería demasiado lento estar ubicando cada punto dentro del espacio, y posteriormente asociar cada vértice a un triángulo, es que requerimos archivos que contengan dicha información para su posterior visualización y modificación. Se necesita como mínimo para el correcto funcionamiento del software un archivo que contenga la información de la ubicación de los vértices dentro del espacio, con sus 3 dimensiones, y la información de los triángulos. Cada triángulo contiene una referencia a 3 vértices por los cuales está compuesto. De esta forma obtenemos finalmente un paquete de coordenadas y referencias que se traduce en una forma 3D que representa un rostro. Dejando fuera elementos como el pintado de la malla y el refinamiento de la misma podemos realizar el trabajo básico de visualización dinámica y edición de un retrato mediante la utilización de la información que nos es otorgada a través del archivo.

Existe dentro de la web diversas fuentes de las cuales se pueden extraer archivos que contengan mallas de superficie, entre ellos podemos hallar infinidad de formatos tales como: .max, .3ds, .c4d, .obj, .ma, .wrl, .xsi, etc. Dentro de estos formatos existen solo algunos que son legibles, es decir, escritos en un archivo de texto plano, de una manera en que no tan solo el ordenador puede interpretarlos sino también el humano, esto nos deja un número reducido de formatos a los que podemos intervenir.

Decidimos utilizar el formato obj, desarrollado por Wavefront Technologies, ya que nos entrega de forma sencilla todos los elementos que componen la malla, tales como la posición de cada vértice y las caras que hacen que cada polígono se defina

como una lista de vértices, además para este formato los vértices se almacenan en orden contrario a las agujas del reloj.

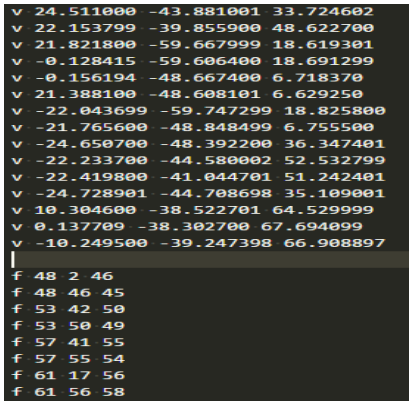



*Fig 2.5: Estructura de un archivo Obj*

Esta es una muestra de un archivo en formato obj en donde se puede apreciar elementos que empiezan con la letra “v”, que representan vértices junto a sus respectivas coordenadas. En segundo lugar, se encuentran los elementos que comienzan por la letra “f” los cuales representan una cara de la malla final.

Un elemento importante a la hora de importar un archivo es el identificador de cada vértice y/o triángulo. Como podemos ver en la imagen estos no cuentan con ningún número que los anteceda, este número-identificador es necesario para cuando se realiza la modificación de alguno de los vértices, por ende, nos permite saber a cuál de todos los vértices nos estamos refiriendo y de esta manera llevar un control de los puntos sobre los que queremos influir, ya que no solamente podemos trasladar un vértice, sino que muchos de ellos a la vez y no queremos que al mover un vértice sea otro el que realice la traslación de sus coordenadas.

## 10.26.- Gestión de archivos

Dentro de la estructura de clases que manera el software se encuentra una clase que está destinada específicamente a la lectura y escritura de archivos, esta clase permite la lectura en diferentes formatos tales como el obj y el m2d. Este último fue proporcionado por el profesor Pedro Rodríguez.

Este último, además permite guardar diferentes configuraciones de la malla y atributos especiales. Más adelante se hablará del formato m2d.

Cuando se requiere leer un archivo en formato obj el software se da a la tarea de crear virtualmente estos identificadores para cada uno de los elementos en orden creciente, partiendo desde el número 1 en adelante. Podemos tener tantos vértices como se requieran, no existe un límite; sin embargo, entre más puntos existan, más recursos computacionales se requerirán al ordenador en el que se está trabajando para su visualización y almacenamiento en memoria.

En un principio la lectura y enumeración de los vértices podría no tienen mucho sentido ya que si solo posicionáramos dichos vértices sobre el espacio podríamos de igual manera visualizar una nube de puntos con la correcta ubicación de cada uno de ellos. Sin embargo, es en la lectura de los triángulos cuando cobra sentido la idea de identificar cada uno de los vértices. Las caras o triángulos vienen escritas dentro del archivo en orden correlativo al igual que los vértices, es decir, cada vértice que fue enumerado corresponde a uno o más triángulos que comparten dicho vértice. De esta forma sino realizamos la identificación de cada vértice posteriormente nos encontraremos con problemas tales como la visualización de una malla totalmente amorfa, triángulos que contienen vértices que no le corresponden, etc.

Por este motivo podemos ver que cada triángulo contiene tres números los cuales indican los 3 vértices por los cuales está compuesto. Este identificador es un número entero que hace referencia al vértice anteriormente enumerado.

*f 48 2 46*
*f 48 46 45*
*f 53 42 50*

La primera cara contiene una referencia al vértice 48, 2 y 46. Estos vértices solo existen si anteriormente son enumerados o, dicho de otra manera, solo existirán si antes de la lectura de los triángulos se les asigna un número o identificador; así podemos buscar posteriormente, dentro de la lista de vértices que tenemos almacenados en memoria, el vértice 48 y este nos entregará sus 3 coordenadas. De la misma forma con todos los demás triángulos o cara en el archivo que se está leyendo.

## 10.27.- Formato M2D

Ya se habló de la importancia de los archivos en formato obj, los cuales nos proporcionan información relevante de cada vértice y de las caras que hacen que cada polígono se defina como un arreglo de vértices. Para este proyecto el formato obj es utilizado nada más como una plantilla que nos permite extraer los datos que están contenidos dentro del archivo, así como también guardar en memoria estos valores y posteriormente manipular cada uno de los elementos. Los archivos con extensión obj son recursos que están ampliamente distribuidos en internet sin embargo el formato nativo que utiliza el software del que estamos hablando es el m2d. Podemos decir que este es un formato mucho más rico y cargado de elementos que nos permiten incorporar funcionalidades como la del refinamiento de triángulos para generar una malla más fina, así como también guardar las coordenadas texturas que se asocian a cada vértice de la malla otorgando color y vida al retrato hablado.

*v 1 22.990599 -35.861034 166.68318*
*v 2 148.54443 -26.13856 -35.59018*

*v 3 124.004 -140.392 -60.8541*

*t 1 53/1 26/2 7/3*
*t 2 53/1 104/4 105/5*
*t 3 3/6 29/7 79/8*

*n 1 11 87 173*
*n 2 141 142 103*
*n 3 169 170 26*

*c 1 0.107059 0.378859*
*c 2 0.081434 0.302244*
*c 3 0.065751 0.372869*

Este es un extracto de un archivo en formato m2d el cual está escrito de forma similar al formato md2, sin embargo, cambian algunos elementos y también se agregan otros.

Los primeros 3 elementos corresponden objetos de tipo vértice que contienen 3 coordenadas, x, y, z. Podemos ver que además cada elemento cuenta con un identificador que sucede al tipo de dato que estamos leyendo. Cada tipo de dato, ya sea un vértice, un triángulo, un vecino o coordenada de textura posee un identificador único que lo diferencia de otro elemento similar. Aunque exista otro elemento que posea el mismo identificador siempre será un elemento de diferente naturaleza lo cual no influye en la identificación de elementos, así como en sus posterior lectura o escritura.

En segundo lugar, encontramos los triángulos, estos están compuestos por 6 elementos los cuales corresponden a 3 pares ordenados de vértices e índices de coordenadas de texturas respectivamente. El elemento t1, por ejemplo, cuenta con la información de los índices o referencias a sus vértices **53**/1 **26**/2 **7**/3, (los cuales

están marcados en negrita). A continuación, encontramos una barra diagonal, la cual nos permite separar la información; el número que acompaña vértice pertenece al identificador de una coordenada de textura, la cual viene dada en la cuarta parte del archivo m2d.

En tercer lugar, encontramos las vecindades, éstas contienen la referencia a un triángulo, sin embargo, ya no poseen la información de los vértices por los cuales está compuesto dicho triángulo sino de otros triángulos con los que comparte una arista o, dicho de otra manera, cada vecino contiene información de otro triángulo con el que comparte 2 vértices en común.

Finalmente nos hallamos con las coordenadas de textura, representadas por la letra c, el identificador de cada coordenada de textura permitirá posteriormente a cada vértice saber qué parte de la imagen, que es puesta sobre la malla, le corresponde. De esta manera cada par ordenado del objeto c determinará una ubicación UV dentro de la imagen. Utilizamos las letras UV ya que las letras x e y se encuentran utilizadas por los vértices de cada triángulo. Sin embargo, podemos decir que la letra U representa el eje X sobre la imagen que envuelve la malla y la letra V corresponde al eje Y.

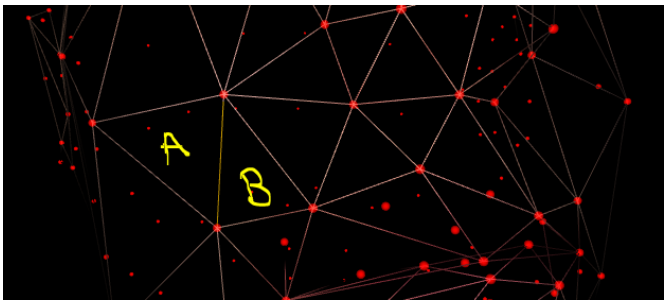



*Fig 2.6: Muestra la arista en común que comparten los Triángulos A y B*

*El triángulo A comparte 2 vértices con el triángulo B, de estos 2 vértices se forma una artista que es necesaria para la posterior ejecución del algoritmo de refinamiento lepp bisection.*

## 10.28.- Exportación de archivos

Una vez que hemos importado un archivo en formato obj, podemos visualizar un rostro, realizar modificaciones y también refinar la malla para que se creen nuevos triángulos y de esta forma podamos obtener una malla más fina. Lo que viene a continuación es guardar este archivo en disco para que cuando necesitemos volver a editar este rostro o realizar alguna modificación, tengamos la malla de la misma forma en la que queremos que se vea. Es necesario almacenar el archivo en disco puesto que la memoria RAM, en donde se encuentran los elementos que estamos modificando, es volátil y una vez que se desconecta la energía o se apaga el ordenador todos los vértices y nuevos triángulos que se hayan creado se borrarán y el archivo volverán a tener los mismos valores que al inicio; por esta razón es imperativo realizar el almacenamiento en disco.

Como se mencionó anteriormente los archivos en formato m2d son los recursos nativos con los que cuenta el software de retratos hablados. Por esta razón es que una vez que se han hecho las modificaciones pertinentes al rostro y se desea guardar el archivo se haga en formato m2d.

En este momento el archivo obj que estaba siendo manipulado por el programa es liberado y se procede a escribir en disco un nuevo documento de texto plano con extensión m2d, el cual es guardado junto con todos los nuevos elementos que contiene la malla, entre ellos encontramos los vértices, los triángulos, los vecinos y las coordenadas de textura, sin olvidar que cada elemento contiene ahora un identificador que permitirá leer de forma rápida los elementos y cargarlos en RAM en una futura edición.

Es necesario señalar que el algoritmo interno que permite establecer las vecindades entre los vecinos solo se ejecuta para archivos obj ya que estos no contienen esa información, mientras que para archivos m2d que sí contienen esta información, este algoritmo de búsqueda y configuración de vecindades es omitido.

# 11.- Conclusiones

El uso de retratos hablados es una de las herramientas preferidas en el área de la criminalística, pues es utilizada para la búsqueda de criminales y personas extraviadas. Los software que existen actualmente no logran alcanzar el nivel de detalle que se puede obtener mediante esta técnica. Esto se debe a la poca versatilidad en la modificación de los rasgos del rostro, pues los programas en cuestión utilizan la superposición de imágenes predefinidas. Así, el software creado presenta una nueva metodología de trabajo, ya que hace uso de mallas de superficie, compuesta de triángulos, lo que permite hacer modificaciones en zonas específicas del rostro. Por tanto, es innecesaria la utilización de imágenes predefinidas y se trabaja en tiempo real. En esta misma línea, cabe mencionar la función del algoritmo de refinamiento Lepp Bisection, pues mejora la calidad de la malla en zonas localizadas de la cara humana que se trabaja.

En cuanto a la interacción usuario-software, es preciso destacar la interfaz amigable del Editor de retratos hablados, ya que presenta opciones concretas y sencillas para la modificación y/o edición de la malla. Así, el usuario es capaz de manejar a cabalidad las funciones del programa, sin la necesidad de recurrir a ayuda externa como un curso o manual de usuario. Además, la universalidad en cuanto a las características del rostro que posee el software, reside en la posibilidad de asignar una malla inicial sobre la cual trabajar y, en base a esto, modificar los rasgos faciales que pueden ser típicamente africanos, asiáticos, occidentales, americanos, indígenas, latinoamericano, etc.

En definitiva, el software presenta una nueva forma de interacción con el usuario para generar retratos hablados, haciendo uso de mallas de superficie que nos permiten mejorar la calidad y flexibilidad en la modificación de los rostros. Esto, con tal de obtener una mejoría en la apariencia y parecido con la persona real que se quiere retratar.

# 12.- Referencias


- Sierra A.G (2014). Análisis del retrato hablado y sus sistemas actuales como medio de identificación en la investigación criminal.
- Singular Inversions. (2018). FaceGen. https://facegen.com
- Vintimilla B.B (2000). Approximation and Geometric Processing of digital images with adaptive triangular meshes.
- Kappraff J. (2014). A Participatory Approach to Modern Geometry.
- Harold E. (2007). Introduction to Non-Euclidean Geometry. Geometría plana.
- Hernandez G.P. (2002). La geometría computacional a nuestro alrededor. Geometría Computacional.
- Bahn P. (1998). La naturaleza y los propósitos de la arqueología. Theories, Methods and Practice.
- Rivara M.C. (2008). Lepp-Bisection algorithms, applications and mathematical properties. Applied Numerical mathematical.
- Rivara M.C. (2011). Multithread parallelization of Lepp-bisection algortihms. Applied Numerical mathematical.
- Ignacia A. (2015). Arte Forense: Retratos hablados con tecnología de punta. México.
- Java Documentation (2018). Oracle Company. https://www.oracle.com.