UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
MAESTRÍA EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

MANIPULACIÓN NATURAL DE OBJETOS VIRTUALES EN REALIDAD VIRTUAL

TESIS
QUE PARA OPTAR POR EL GRADO DE
MAESTRA EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

PRESENTA:
YADIRA FLEITAS TORANZO

TUTOR O TUTORES PRINCIPALES
DR. ALFONSO GASTÉLUM-STROZZI ICAT

CIUDAD DE MÉXICO, ENERO 2022

"Si no puedo dibujarlo,
es que no lo entiendo."
Albert Einstein

Agradecimientos

Quisiera empezar agradeciendo a mis padres, abuelita, hermana y al actual miembro de la familia, Matteo, mi sobrino querido. Gracias por su gran apoyo incondicional, compañía y por haberme impulsado a seguir mis sueños. Sin ustedes no podría estar en donde me encuentro ahora.

Estudiar una maestría en la UNAM ha sido una de las oportunidades más grandes e importantes que he tenido en mi vida. Me ha permitido crecer tanto profesional como personalmente, cuestionándome el por qué de las cosas; experimentando y cometiendo errores para así aprender de ellos. A lo largo de la maestría, he descubierto un mundo diferente al que conocía, en donde diversas maneras de pensar nos brindan perspectivas diferentes; donde el interés mutuo por lograr especializarse en diversos temas, nos une en una gran comunidad.

Es por ello que agradezco de todo corazón a mi asesor el Dr. Alfonso Gastelum Strozzi, por su paciencia, tiempo, conocimiento y entusiasmo. Gracias por la oportunidad que me dió al escuchar mis propuestas de proyecto y abrir las puertas a una tecnología que actualmente en México se encuentra evolucionando.

Además, agradezco a mi ex tutor de Licenciatura, el Dr. Luis Felipe Marín Urías, por la continua retroalimentación que me proporcionó y la oportunidad de compartir diferentes perspectivas en el desarrollo de la tesis; así como mensajes de apoyo que me motivaron a continuar con este trabajo. Gracias de todo corazón.

Agradezco a mis maestros Dr. Edgar Garduño Angeles, Dr. Alfonso Gastelum Strozzi, Dr. Jesús Savage Carmona, Dr. Jorge Luis Pérez González, Dr. Julián Bravo Castillero, Dr. Miguel Angel Padilla Castañeda, que de una u otra forma, parte del conocimiento reflejado en el trabajo se debe a ellos. Así como a mis sinodales por su tiempo y su voto de confianza.

Asimismo, agradezco a mis compañeros de generación Mtro. Diego Isla López, Mtro. Eduardo Acuña Yeomans, Mtro. Saul Ivan Rivas Vega y compañero de laboratorio Mtro. Cesar Victoria, cuya compañía emocional y apoyo en temas de estudio, permitieron una estancia mucho más amena y feliz.

Agradezo además, a mis amigos Ing. Angel Eduardo Moguel Cattaneo, Ing. Arielm Navarrete Aguilar, Lic. Andrea Fernández Santana, Ing. Salvador Fernández Santana, Mtra. Edith Barrera Arizmendi, Mtro. Guillermo Hernandez Zamudio, por su apoyo incondicional en los momentos más difíciles de estos dos años, así como su participación en la aplicación del sistema y apoyo en la redacción del manuscrito.

Gracias al Consejo Nacional de Ciencias y Tecnología (CONACYT) por la oportunidad de estudiar con su apoyo financiero de dos años.

Natural Manipulation of virtual objects in virtual reality

by

Yadira Fleitas Toranzo

Abstract

One of the challenges in the Human-Computer Interaction area is to know which are the best ways of interaction for a system to be accepted by the public in general. Currently, in Virtual Reality, it is possible to interact with virtual objects through the learning of specific simplified gestures, however, this set of preconfigured, gestures prevents the user from easily carrying out all kinds of activities. In this work, natural ways of interaction will be analyzed and evaluated, with the objective to obtain the degree of similarity in the interaction with virtual objects and the closes corresponding interaction in the real world. The main goal of this research is to improve the efficiency of the detection and interpretation of virtual gestures and activities and with this create a greater degree of immersion. A comparison is presented through 3 different scenarios that involved 3 types of interaction with the hands, which correspond to interaction with a learned gesture (pinch gesture), natural interaction without touch, and finally natural interaction with passive haptic feedback. For this last interaction, an RGB sensor was used to detect and recognize real objects through a convolutional neural network to identify the type of object and its correspondence with the previously defined virtual model. The experiments have been performed with 12 users with a range of ages from 27 to 62 years old. Preliminary results of this work show that the age of the user is an important factor in deciding the type of interaction in the virtual environment. Another important conclusion of this work is the importance of continually developing techniques and methods for natural interaction since in general, they allow for a more flexible and immersed experience in the virtual worlds.

Manipulación natural de objetos virtuales en realidad virtual

por

Yadira Fleitas Toranzo

Resumen

Uno de los retos del área de Interacción Humano-Computadora es conocer cuáles son las mejores vías de interacción para que un sistema sea aceptado por el público general. Actualmente en Realidad Virtual es posible interactuar con objetos virtuales a través del aprendizaje de gestos con las manos, sin embargo, esta limitación de posturas impide al usuario realizar con facilidad todo tipo de actividades, por lo que se analizará y evaluarán vías de interacción naturales, cuyo grado de similitud en la interacción con los objetos en el entorno virtual sea bastante cercano a la interacción correspondiente en el mundo real. El objetivo principal es mejorar la eficiencia en las actividades y tener un mayor grado de inmersión en el uso de esta tecnología. Se propone, por tanto, a través de 3 escenarios diferentes, una comparativa de 3 tipos de interacción con las manos, las cuales corresponden a interacción con un gesto aprendido (gesto de pinza), interacción natural sin tacto y finalmente interacción natural con retroalimentación háptica pasiva. Para este último escenario de interacción, se utilizó un sensor de color para detectar y reconocer los objetos reales en donde a través de una red neuronal convolucional se identificará el tipo de objeto y a partir de ello, se hará su correspondencia con los modelos virtuales previamente definidos en el ambiente virtual. Los experimentos se realizaron con 12 usuarios de edades entre 27 y 62 años, en donde los resultados preliminares muestran que la edad es un factor importante en la decisión del tipo de interacción. Además, el desarrollo continuo de técnicas y métodos de interacción natural, en general, permiten una experiencia más flexible e inmersiva en los mundos virtuales.

Índice general

Índice de tablas

Índice de figuras

Introducción

La presencia de nuestras manos como herramienta y medio de comunicación ha permitido que las actividades diarias del ser humano sean más eficientes. Con la era de la información digital, éstas han permanecido constantes en la interacción humano-máquina, en donde precisamente se concentra un área de investigación llamada *Human Computer Interaction* (*HCI*). Así, la manera en la que interactuamos con los dispositivos electrónicos se ha ido transformando dado los avances tecnológicos y por tanto nuevas investigaciones surgen con respecto a los equipos (computadoras, celulares, consolas de videojuegos) y los tipos de manipulación que se pueden tener a través de las manos, con el objetivo de tener mayor presencia e interés en el uso de dichos dispositivos. Por ejemplo, los teléfonos con tecnología táctil han permitido un mercado amplio de usuarios que se adaptan más a la interacción directa a través de la acción táctil de los dedos en la pantalla, en vez de los celulares que utilizan teclas para realizar las mismas actividades.

Con el surgimiento de la Realidad Virtual (*Virtual Reality* o *VR*), cuya tecnología permite generar un mundo digital en el que somos inmersos [Mihelj *et al.*, 2014], los medios físicos de interacción generalmente se dividen en dos tipos: a través de uno o más controles que debemos mantener en nuestras manos en todo momento; o por medio del aprendizaje de gestos utilizando solamente nuestras manos.

Actualmente en VR, el uso de gestos para interactuar con objetos virtuales se ha limitado a pocas posturas en las manos que el usuario debe aprender, no obstante, podemos ver que existe un contacto cada vez más directo con la información virtual, surgiendo interrogantes respecto a qué otros gestos deberíamos adoptar en las manos para interactuar de la mejor manera, lo

que nos lleva a la siguiente pregunta ¿Es posible mejorar la interacción si manipulamos objetos virtuales de forma natural?

La naturalidad, en el ámbito de la tecnología, se puede evaluar respecto al grado de similitud de las actividades en el entorno virtual con esas mismas actividades realizadas en el mundo real [Bowman *et al.*, 2012a]. Por ejemplo, si quisiéramos agarrar una pelota virtual, el proceso más natural sería acercar nuestra mano a la pelota y tomarla a través de algún gesto conocido para agarrar objetos redondos.

El uso de este tipo de tendencia hacia lo natural resulta necesaria en situaciones en donde se requiere efectuar actividades de entrenamiento (cirugías, procesos industriales, tratamiento de fobias), ya que simular estas actividades con un grado de similitud elevado, contribuiría en un mejor desempeño y experiencia, evitando el riesgo que podría resultar la práctica de estas en escenarios reales [Aïm *et al.*, 2016; Howard, 2017].

Como resultado, surge la necesidad de cuestionarnos hasta qué punto sería posible aplicar la interacción natural en todo tipo de actividades dentro de la Realidad Virtual, ya que instintivamente podríamos pensar que, si los objetos virtuales se asemejan a formas, texturas y comportamientos reales, por qué deberíamos interactuar diferente a la manera en la que normalmente interactuamos con los objetos reales.

Sin embargo, la interacción natural no siempre es sinónimo de algo fácil, rápido y eficiente. En situaciones en donde se desea manipular objetos muy grandes, lejanos o trasladarlo de un lugar a otro de forma rápida, la simulación de estos procesos como la vida real podría resultar tedioso e ineficiente. Además, trabajar directamente con los gestos en el aire, conlleva una desventaja sustancial en comparación con los equipos físicos o guantes hápticos: la falta de retroalimentación del tacto, pudiendo ocasionar inestabilidad en las posturas. Para ello, diversas soluciones a estos dos problemas han sido propuestas, ya sea a través de pequeñas versiones del escenario virtual al alcance de las manos, de modo que se puedan manipular directamente y ver su comportamiento en tiempo real [Kang *et al.*, 2020b], así como, a través de háptica pasiva que ofrece una solución a bajo costo (comparado con los guantes hápticos) que permite mapear elementos reales al mundo virtual [Shaik *et al.*, 2015; Muender *et al.*, 2019; Feick *et al.*, 2020].

El resultado de estas propuestas logra solucionar diferentes limitantes en la interacción a través de las manos. En específico, [Kang *et al.*, 2020b] realizan una comparativa a través de diferentes formas de interactuar en donde concluye que la interacción natural es la más aceptada por usuarios, sin embargo, el experimento fue realizado con Realidad Mixta, que combina elementos reales y virtuales. La Realidad Virtual es una tecnología que aún no termina de configurarse, por lo que es necesario el análisis de diferentes maneras de interacción que nos permita conocer las vías más efectivas y flexibles para lograr mayor interés en el uso de esta. Por consiguiente, se propone identificar estas vías a través de manipulación natural con objetos virtuales, por medio de una herramienta que permita evaluar diferentes formas de manipulación, con el objetivo de establecer un estándar de interacción más completo.

Capítulo 1

Problema de investigación

1.1. Planteamiento del Problema

Dado el creciente incremento en el mercado de dispositivos que permiten visualizar objetos tridimensionales [Zervos, 2016], la forma en la que se interactúa con las aplicaciones a través de controles o gestos en las manos sigue siendo un tema de investigación, debido a factores como el aprendizaje en el uso de las mismas, la restricción de utilizar pocos gestos para interactuar exclusivamente de una sola forma, y no poder utilizarlo en escenarios en donde se requiera simular la realidad.

Uno de los gestos más utilizados es el conocido como gesto de pinza, que se identifica cuando el pulgar y el índice se encuentran juntos. La razón por la cual ha sido tan popular es debido a que es un gesto que solemos adoptar cuando tomamos pequeños objetos entre el pulgar e índice; además, se puede detectar más fácil desde la perspectiva de la persona (en donde generalmente se encuentran las cámaras de los lentes) y se logra obtener retroalimentación háptica con el toque de los dedos.

Siguiendo la tendencia de querer algo más cercano a la forma en la que interactuamos con los objetos reales, ¿qué sucedería si manipuláramos los objetos virtuales considerando su volumen? En este caso se debería obtener en todo momento la posición de las manos para poder interpretar correctamente cuando existe una colisión entre la mano y el objeto. Esto permitiría

al usuario, tener mayor libertad en la interacción, pudiendo adoptar gestos propios sin previo aprendizaje de estos. Sin embargo, utilizar gestos en el aire, es decir sin tener retroalimentación háptica podría verse afectado en cuanto a la comodidad en la manipulación de los objetos, por lo que varias investigaciones proponen soluciones a bajo costo (háptica pasiva) utilizando formas físicas mapeadas al mundo virtual para permitir el contacto real con el objeto virtual [Hettiarachchi y Wigdor, 2016; Jiang *et al.*, 2018; Muender *et al.*, 2019]. No obstante, depender de objetos físicos limita las propiedades físicas de los objetos virtuales y el mapeado de los objetos reales al mundo virtual de los trabajos relacionados depende de varios sensores.

Este tipo de investigaciones que proponen soluciones a los tipos de interacciones en los mundos virtuales, han sido abordadas evaluando exclusivamente la herramienta sin comparar con lo que realmente existe. Esto nos lleva a la siguiente pregunta: ¿Utilizar gestos naturales facilitaría la manipulación de objetos virtuales comparado con los gestos artificiales? Es por ello que se presentará, un sistema que permitirá identificar qué gestos naturales y/o artificiales permiten al usuario realizar con mayor facilidad la interacción con los objetos virtuales, a través de tres posibles escenarios en donde el primero permitirá interactuar exclusivamente con el gesto de pinza, el segundo le permitirá al usuario interactuar pensando que el objeto virtual tuviese volumen físico real, y el tercer escenario permitirá además tener retroalimentación háptica pasiva a través de un mapeado físico-virtual de los objetos con el objetivo de evaluar el factor háptico como variable importante en la interacción natural.

1.2. Justificación

El estudio de la forma en la que interactuamos con la información tridimensional en ambientes virtuales para el área de HCI, ha sido motivo de investigación desde los inicios de la Realidad Virtual. En comparación con lo que tenemos actualmente reflejado en las pantallas digitales, donde la información es dada en dos dimensiones, las ventajas de un espacio virtual tridimensional traen consigo un abanico de posibilidades en donde la búsqueda de una mejor interacción se complica por la adición de una nueva dimensión. Es por ello que surge la necesi-

dad de proponer nuevas técnicas para obtener una mejor interpretación y manipulación de los datos, optimizando los tiempos en interacción y facilitando el uso de este tipo de tecnologías con la menor cantidad de dispositivos posibles.

Por tanto, la aproximación de interacción natural humana en ambientes virtuales, comprende una gran oportunidad para entornos de simulación de actividades que se necesite realismo para en entrenamientos (prácticas médicas, tratamiento de fobias, entre otros), alcanzar un mayor interés en el desempeño de actividades que requieran cierta inmersión [Kang *et al.*, 2020a] y ampliar la barrera de edad en el uso de este tipo de tecnologías, ya que actualmente, los dispositivos comerciales de Realidad Virtual no abarcan un amplio espectro de edades [Xue *et al.*, 2020], impidiendo a los más jóvenes, a los mayores, y las personas con menor capacidad de aprendizaje el acceso intuitivo a los recursos.

Asimismo, es importante definir las pautas que se deben tener para una correcta interacción espacial, ya que una vez se tenga una estructura o base de donde partir, esta pueda dar paso a los investigadores en el desarrollo de aplicaciones más específicas dirigidas a diferentes áreas, siendo posiblemente una razón por la que actualmente no tiene gran presencia en el mercado [Chen y Duh, 2019].

1.3. Objetivos

1.3.1. Generales

Identificar los gestos que faciliten la manipulación con objetos virtuales, a través de una herramienta que permita comparar entre gestos naturales y artificiales para su uso en Realidad Virtual, obteniendo los parámetros necesarios para evaluar el grado de usabilidad y eficiencia en este tipo de tecnologías.

1.3.2. Específicos

- Investigar, analizar y comparar:

    - Estado del arte de las diferentes soluciones presentadas en interacción humano-computadora en Realidad Virtual
    - Las diferentes formas de manipulación natural con los objetos físicos reales, para poder adaptarlo al ambiente virtual
    - Escenarios posibles en donde se realizarán las pruebas que incluya diferentes tipos de interacción

- Desarrollar:

    - Detección del escenario real y corrección de la perspectiva
    - Reconocimiento del objeto real
    - Delimitación del escenario virtual y mapeo del objeto real al objeto virtual
    - Implementación del método de interacción de pinza y de interacción natural
    - Creación del demo para pruebas en usuarios

- Analizar los datos:

    - Determinando los parámetros a medir para realizar un análisis cuantitativo y cualitativo de las tres formas de interacción

1.4. Hipótesis

H1. A través de la interacción natural en Realidad Virtual es posible facilitar la manipulación de objetos virtuales y permitir mayor inmersión.

H2. La interacción natural tendrá mejores resultados si existe retroalimentación háptica.

Capítulo 2

Marco Teórico

2.1. Visión por computadora

La visión, en términos biológicos juega un rol muy importante en el sistema humano. Entre un 90-92 % de la información sensorial recuperada del entorno es a través de nuestro sistema de visión [Baritz y Cotoros, 2011]; sin embargo, la información obtenida a través de nuestros ojos depende de otros elementos para la interpretación de la misma, por ejemplo, la percepción de cada individuo, por lo que definir visión en términos generales resulta complejo.

En lo que respecta al área computacional, una de las definiciones más acertadas es la del neurocientífico David Marr [Marr y Nishihara, 1978], que dice:

> *"Visión es un proceso que a partir de imágenes del mundo exterior produce una descripción que es útil para el observador y que no tiene información irrelevante"*

Es decir, visión computacional o también llamada visión artificial, es una abstracción del sistema de visión humana a nivel computacional, cuyo objetivo es poder interpretar una señal obtenida por uno o más sensores y extraer información relevante que va desde la detección de bordes hasta un sistema que permita detectar, reconocer, reconstruir elementos tridimensionales, etc. En este sentido, dependiendo de la aplicación que se utilizará, la visión artificial se puede dividir en tres niveles fundamentales [Sucar y Gómez, 2011]:

- **Nivel bajo de procesamiento:** extracción de características tales como bordes, color, profundidad, entre otros; donde se involucran técnicas del área de procesamiento digital de imágenes.

- **Nivel intermedio de procesamiento:** agrupación a través de características similares (obtenidas en el nivel bajo) como son contornos, regiones, etc.

- **Nivel alto de procesamiento:** descripción de los elementos de las imágenes, a través de los resultados de los niveles inferiores, con base en modelos y/o conocimiento previo

A lo largo de esta sección se sintetizarán las bases de los métodos utilizados en el marco metodológico, en donde se involucran técnicas de nivel bajo, intermedio y alto con el objetivo de reconocer objetos en una imagen.

2.1.1. Formación de la imagen

Parecido a lo que sucede con el sistema de visión humano, la formación de una imagen ocurre siempre que existan tres elementos principales: un medio físico, una o más fuentes de iluminación, y una o más cámaras que reciban la señal de luz rebotada por el medio [Gonzalez y Woods, 2018] (Figura 2-1).

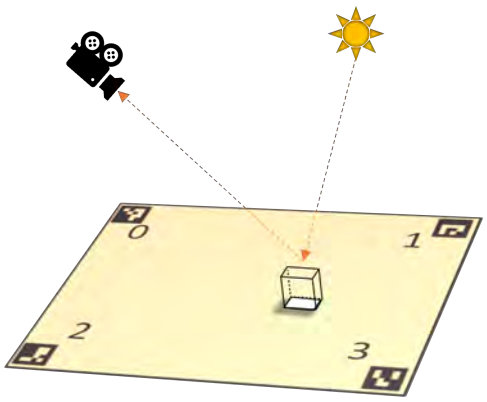

Figura 2-1: Elementos principales para la formación de la imagen. Medio físico, una o más fuentes de luz y una o más sensores de color

La fuente de luz es un tipo de energía que se representa como una onda electromagnética que posee cierta longitud y que dependiendo de esa longitud podemos percibir diferentes colores en el medio. Llamemos a una luz monocromática o acromática, si se obtiene del entorno intensidades de la luz, es decir, una imagen a escala de grises. En cambio, la luz cromática la podemos percibir por las diferentes longitudes de onda en un rango de valor conocido como el espectro visible, cuyos valores se encuentran aproximadamente entre 400nm (violeta) y 700nm (rojo) (Figura 2-2).

Figura 2-2: Espectro electromagnético visible por el ojo humano. Recuperado en octubre 2021 de: https://www.flickr.com/photos/clinicadyto/11072737554

El medio físico, está conformado por superficies cuyas propiedades son las que determinan qué porcentaje de longitud de onda será absorbida por el material y qué porcentaje será reflejado. La luz reflejada será la que obtendrá el sensor y se representará como una combinación de las fuentes de iluminación y del porcentaje de reflexión por los materiales en el medio.

El ojo humano percibe el color a través de unas células foto-receptoras que poseemos en nuestra retina llamadas conos, que se identifican como L, M y S (*Long*, *Medium* y *Short*, refiriéndose del inglés a largas, medianas y cortas ondas) y que se diferencian por el grado de sensibilidad al obtener las longitudes de onda, donde el 65 % de estos conos son sensibles al color rojo (*Red*), el 33 % al verde (*Green*) y el 2 % al color azul (*Blue*). Es por ello que estos tres colores son conocidos como los colores primarios (*RGB*) y a partir de su combinación es que podemos formar otros colores. Debido a que las cámaras intentan imitar la forma en la

que vemos el mundo, se utilizan sensores cromáticos cuyo objetivo precisamente es distinguir diferentes niveles de longitudes de onda.

Una vez sensado estos valores de onda, la salida de la mayoría de estos sensores es representada por ondas de voltaje continuo que son convertidas a datos discretos a través de muestro y cuantificación, en donde la calidad de la imagen estará determinada por el número de muestras y la cantidad discreta de niveles de intensidad permitida.

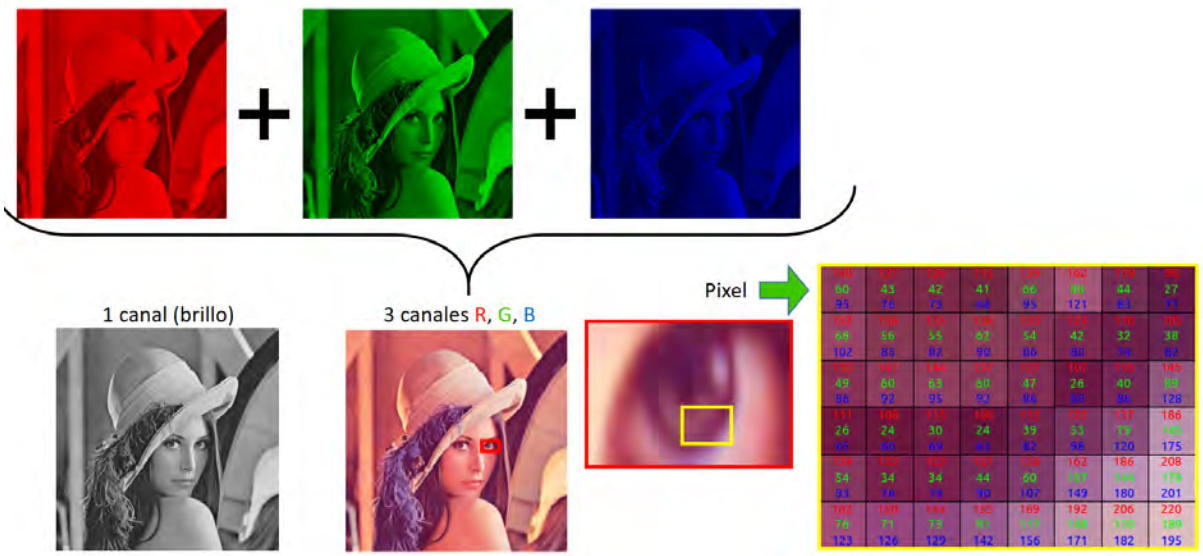

Figura 2-3: Formación de la imagen en escala de grises y en espacio de color RGB. Recuperado en octubre 2021 de: https://bryanmed.github.io/ImagenDigital/

Finalmente, la imagen es definida como una matriz de pixeles, en donde un pixel expresa la menor unidad discreta posible dentro de una imagen. Esto es, dada una imagen a color visualizada desde un monitor, tendremos que para cada pixel existirán la combinación de los colores primarios o tres canales RGB, y en cada canal tendremos un valor de intensidad discreto, por ejemplo, de 8 bits (entre 0 y 255), donde el valor mínimo significará la ausencia del color presente en ese canal y el valor máximo la mayor exposición de color en el mismo (Figura 2-3).

2.1.1.1. Espacios de color

La manera en la que podemos diferenciar un color de otro está dada por su brillo y cromaticidad (combinación de matiz y saturación). El brillo (*brightness*) que es una medida subjetiva que se relaciona con la descripción de la luz monocromática, es decir, toma en cuenta los cambios en la intensidad; la matiz (*hue*) se refiere particularmente al color (longitud de onda) dominante de la mezcla de estos valores, o sea, el color resultado que nos permite identificarlo de los demás; y la saturación (*saturation*) se refiere a la pureza del color, es decir, que tanta luz blanca en la mezcla de los colores se tiene, por ejemplo el color rojo tiene más saturación que el color rosado (mezcla de rojo y blanco).

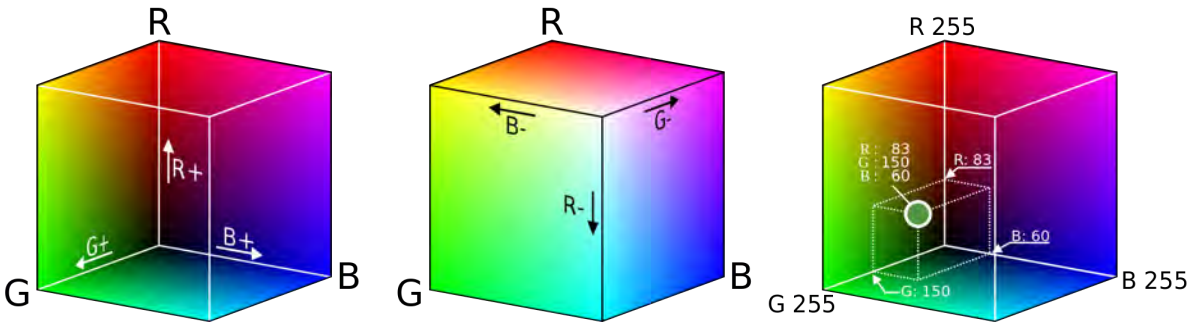

Figura 2-4: Espacio de color RGB en coordenadas cartesianas. Recuperado en octubre 2021 de: https://www.pngegg.com/es/png-nolzf

Para ello, existen diferentes modelos o espacios de color que sirven como estándar para poder interpretar los colores en una imagen dependiendo del propósito. Es decir, si se requiere a nivel hardware como impresoras se ocupa CMY (cian, magenta, amarillo) o CMYK (cian, magenta, amarillo, negro) o bien, para monitores se suele utilizar RGB.

Este modelo se puede visualizar como un cubo en el espacio 3D de coordenadas cartesianas, en donde a cada eje se la asigna uno de los valores RGB y las esquinas restantes del cubo le corresponden los colores secundarios (cian, magenta y amarillo), el color blanco (total exposición de color) y el color negro en el origen de estas coordenadas (total ausencia de color) (Figura 2-4). Para obtener la imagen de monocromática (escala de grises) nos deberemos fijar en la

diagonal principal que parte del negro y llega al color blanco, es decir, el promedio de los 3 canales.

En caso de que se requiera interpretar la imagen a un nivel de percepción del color, se suele utilizar diferentes modelos tales como HSI (matiz, saturación, intensidad) en donde permite separar la información de cromática de la acromática (niveles de grises) [Gonzalez y Woods, 2018]; HSL(matiz, saturación, luminosidad) que toma en cuenta la sensibilidad con ciertos colores, por ejemplo la distinción del amarillo y verde bajo diferentes cambios de iluminación [Nishad, 2013] y HSV (matiz, saturación, valor) cuyo sistema de color se asemeja a la forma en la que percibimos el color [Su *et al.*, 2011].

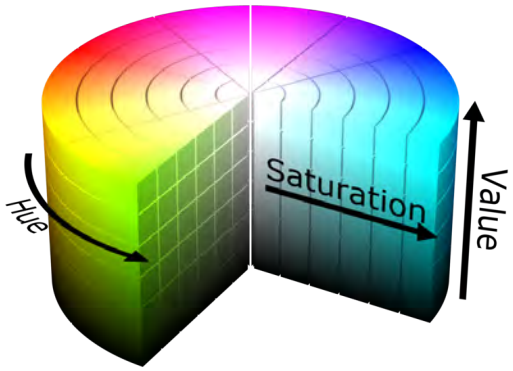

Figura 2-5: Modelo de color HSV en coordenadas cilíndricas. Recuperado en octubre 2021 de: https://es.m.wikipedia.org/wiki/Archivo:HSV_color_solid_cylinder.png

En todos estos modelos, tenemos un conjunto de planos perpendiculares a un eje vertical principal, que pueden tomar diferentes interpretaciones en la forma (triangular, circular, hexagonal), y en donde cada plano presenta los colores primarios y secundarios en los bordes separados por ángulos de 120^o y 60^o. Comúnmente se suele utilizar una pirámide hexagonal para entender el concepto de estos modelos, por lo que la matiz va a ser el ángulo alrededor del eje cilíndrico, la saturación será la magnitud del vector partiendo del origen que marca el eje vertical principal a un punto en el plano y la intensidad está dada por el eje vertical en una escala de negro a blanco (Figura 2-5).

Cabe mencionar que podemos tener ciertas equivalencias y cambiar entre un sistema y otro según nos convenga, para finalmente poder mostrar el resultado través del modelo de color RGB.

Actualmente el modelo HSV ha sido muy utilizado por la sencillez en las equivalencias, ya que no requiere funciones trigonométricas minimizando el costo computacional, por lo que es utilizado en aplicaciones de graficación por computadora [Chen *et al.*, 2007], segmentación de rostros [Shaik *et al.*, 2015], entre otros. Para conocer más a fondo sobre este modelo, es necesario entender sus características y cómo se relaciona con respecto al espacio RGB a través de sus equivalencias.

Modelo HSV Creado por Alvy Ray Smith en 1978, donde la dimensión valor (*value*), se describe como el canal que va desde el color negro hasta el tono puro del color, es decir, completamente saturado [Smith, 1978], por lo que cuando mezclamos los colores puros con el color blanco, se reduce la saturación, pero si son mezclados con negro, entonces el valor máximo de saturación no cambia. Esto ofrece una gran ventaja en regular los niveles de oscuridad en las imágenes sirviendo como auxiliar en la eliminación de sombras [Huang *et al.*, 2015; Surkutlawar y Kulkarni, 2013].

Para pasar del espacio de color RGB a HSV, debemos tener normalizado los valores RGB, entre (0,1) y calcular el valor máximo $M = max(R, G, B)$, mínimo $m = min(R, G, B)$ de los tres canales, así como la distancia entre ese máximo y mínimo $\Delta = M - m$. Por tanto, tenemos que:

El matiz está dado como:

$$H = \begin{cases} \text{No definido} & \Delta = 0 \\ 60^o * \frac{G-B}{\Delta} & M = R \;\; y \;\; G \geq B \\ 60^o * \frac{G-B}{\Delta} + 360^o & M = R \;\; y \;\; G < B \\ 60^o * \frac{B-R}{\Delta} + 120^o & M = G \\ 60^o * \frac{R-G}{\Delta} + 240^o & M = B \end{cases} \tag{2-1}$$

La saturación por:

$$S = \begin{cases} 0 & M = 0 \\ \frac{\Delta}{M} & M \neq 0 \end{cases} \tag{2-2}$$

Finalmente, **el valor** es precisamente el canal predominante de RGB, es decir

$$V = M \tag{2-3}$$

Los valores de S y V son normalizados $[0, 1]$, mientras que el valor H debe estar en el intervalo $(0, 360)$

En caso de que queramos pasar de HSV a RGB, es decir, el proceso inverso, lo podemos resumir en la siguiente tabla 2-1, en donde dependerá del ángulo que tenga Hue, y por tanto el valor RGB se calculará con respecto a la sección donde se encuentre el punto [Xiao y Ohya, 2007]. Recordemos que la forma cilíndrica se divide en 6 secciones, conformado por los colores primarios y secundarios con 60^o de separación. Además, recordemos que H se refiere al Hue, S a la Saturación, V al valor, y R,G,B corresponde a los 3 canales del modelo de color RGB, por lo que a partir de las siguientes ecuaciones complementarias:

$$Hi = (\frac{H}{60^o}) mod\, 6 \tag{2-4}$$

$$Ht = (\frac{H}{60^o}) - Hi \tag{2-5}$$

$$P = V(1 - S) \tag{2-6}$$

$$Q = V(1 - Ht * S) \tag{2-7}$$

$$T = V(1 - (1 - Ht)S) \tag{2-8}$$

En donde tendríamos en la tabla 2-1 (recordemos que V corresponde al valor de HSV) que:

Hi	***R***	***G***	***B***
0	V	T	P
1	Q	V	P
2	P	V	T
3	P	Q	V
4	T	P	V
5	V	P	Q

Tabla 2-1: Conversión HSV a RGB con base en las ecuaciones 2-6, 2-7, 2-8 y Value (V)

2.1.2. Procesamiento de la imagen

El procesamiento digital de imágenes tiene como objetivo mejorar la calidad de estas permitiendo resaltar las características más importantes que ayuden al área de visión por computadora a una correcta interpretación del entorno [Gonzalez y Woods, 2018].

2.1.2.1. Filtrado

Los filtros son un conjunto de técnicas que nos permiten destacar o atenuar características en una imagen, por ejemplo, filtros para la reducción en el ruido, detección de bordes, suavizado, entre otras. Estos filtros se podrán utilizar en dos dominios diferentes dependiendo de lo que se necesite. Por lo que para aplicaciones en donde se requiera utilizar filtros de gran tamaño, precisión y flexibilidad, será más práctico utilizar el dominio de la frecuencia [Gonzalez y Woods, 2018; Sucar y Gómez, 2011]. En los casos donde se requiera un nivel simple en la eliminación de ruido se pueden utilizar los filtros en el dominio espacial y aplicarlo directamente a la imagen a través de la convolución espacial (Apéndice A), esto es, operar directamente sobre los pixeles y sus vecinos en la imagen de entrada, sin necesidad de transformarla en otro espacio. Nos centraremos en este último dominio para explicar los temas siguientes.

2.1.2.2. Suavizado en Imágenes

El proceso de suavizado tiene como objetivo pasar las frecuencias bajas y por tanto remover cambios abruptos en la imagen, como por ejemplo bordes. Se pueden utilizar diferentes filtros paso bajo (*lowpass filter*) para lograr este efecto (ver en Apéndice A.1.1), como el filtro promedio (o media aritmética), gaussiano o una aproximación de este conocido como filtro binomial [Gonzalez y Woods, 2018] (Figuras 2-6).

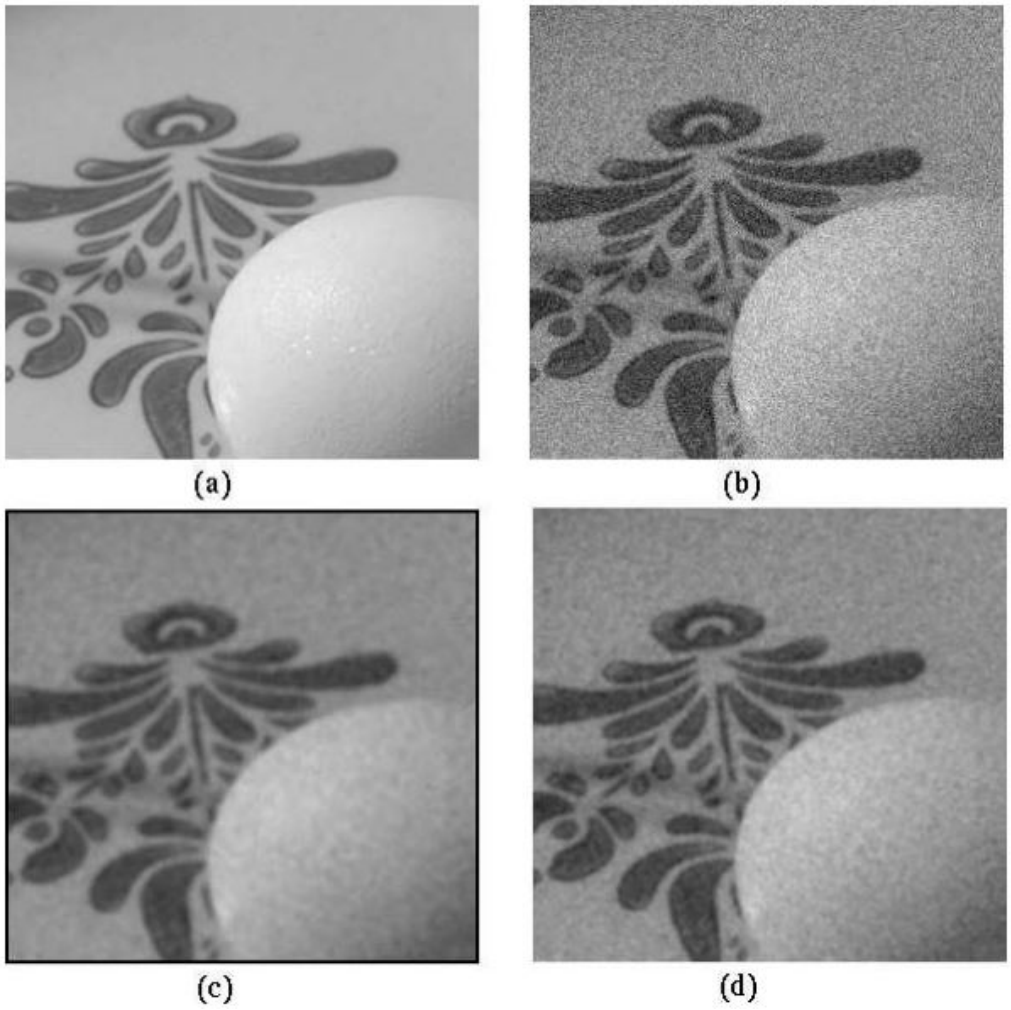

Figura 2-6: Ejemplos de filtros paso bajos. (a) Imagen original, (b) Imagen con ruino gaussiano, (c) Filtrado de la imagen b, con filtro promedio de 5x5, (d) Filtrado de la imagen b, con filtro gaussiano $\sigma = 1$. Imagen recuperada de [Sucar y Gómez, 2011]

Se dice que el filtro Gaussiano es el que más se aproxima a la manera en la que los humanos difuminan el fondo del entorno (sobre todo cuando estamos enfocando la atención hacia un elemento) [Sucar y Gómez, 2011]. Además, ha sido muy utilizado en algoritmos de detección de bordes por la capacidad que tiene de no distorsionar en exceso la imagen [Basu, 2002].

2.1.2.3. Detección de Bordes

Dado que los bordes se pueden interpretar como cambios abruptos en una imagen, se utilizan filtros paso alto (*highpass filter*) que a diferencia de los filtros *lowpass*, permiten el flujo de frecuencias altas eliminando las pequeñas variaciones. Para obtener estos filtros generalmente se calculan a través de los filtros *lowpass* aplicando operaciones de sustracción o derivación, como por ejemplo el filtro highpass de sustracción de la media, y los filtros Prewitt y Roberts (ver Apéndice A.1.2).

Estos filtros que actúan exclusivamente sobre la diferenciación son muy sensibles a ruidos en la imagen, ya que los cambios de intensidades provocarían un cambio brusco como se muestra en la imagen 2-7. Para ello se propone un filtro que, a través de la combinación del filtro gaussiano y la primera derivada del mismo, arroja resultados más estables, y es conocido como filtro Sobel [Sobel y Feldman, 1973] (consultar más información en Apéndice A.1.2).

Aunque el filtro Sobel presenta mejores resultados comparado a los demás presentados, tenemos desventajas tales como la presencia de bordes gruesos, redundantes y un bajo rendimiento en la detección de bordes diagonales (Figura 2-7). En consecuencia, en 1986 se desarrolla una de las técnicas más utilizadas hasta la fecha para la detección de bordes [Canny, 1986], llamada Algoritmo de Canny.

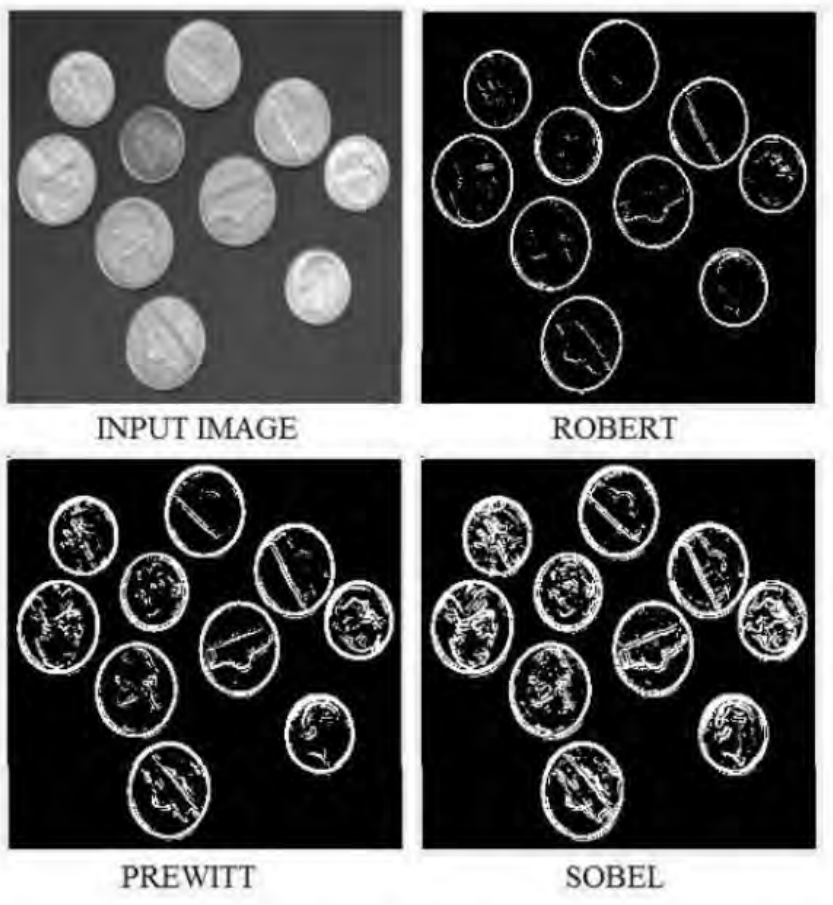

Figura 2-7: Ejemplo de detección de bordes a través de Prewitt, Roberts y magnitud del gradiente de Sobel. Imagen recuperada de [Mahalle y Shah, 2017]

Algoritmo de Canny Presentada por Canny y conocido justo por su nombre, esta técnica tiene como objetivo encontrar los bordes de manera óptima con la menor cantidad de falsos positivos y eliminar bordes redundantes. Se divide en 4 principales pasos para lograr un buen resultado:

1. Aplicar suavizado en la imagen a través de un filtro gaussiano de 5x5 para remover ruido
2. A partir del filtro Sobel en horizontal y vertical, se obtiene la primera derivada en ambos sentidos de modo que podemos encontrar los gradientes y su dirección en toda la imagen
3. Utilizando la técnica *Non-maximum suppression* (*NMS*), se eliminan pixeles que no con-

tribuyan a los bordes encontrando los máximos locales con respecto a la dirección del gradiente. Es decir, se adelgazan los bordes gruesos

4. La etapa final, conocida como Umbral de Histéresis, identifica los bordes bien definidos en la imagen. Para ello se debe delimitar el grado mínimo y máximo permitido a través de dos umbrales: minVal y maxVal. El umbral mínimo descarta bordes de menor intensidad identificados. A menor umbral mínimo mayor presencia de bordes superficiales. El umbral máximo define a partir de que valor de intensidad del gradiente es considerado un borde, por lo que, a mayor umbral máximo, menor número de bordes, pero mayor probabilidad de bordes bien definidos. Los bordes que se encuentran dentro de estos dos umbrales podrán formar parte de la clasificación de bordes siempre que exista conectividad con los bordes ya identificados (véase imagen 2-8)

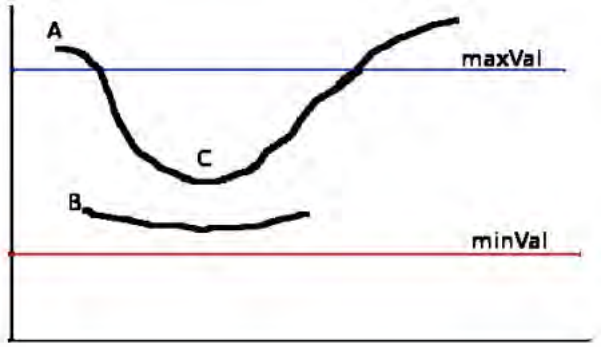

Figura 2-8: Umbral de Histéresis para decidir qué bordes se considerarán en el resultado final. Imagen recuperada de `https://docs.opencv.org/3.4/da/d22/tutorial/py/canny.html`

Por tanto, dado el umbral definido en la figura 2-8, el borde que incluye los puntos A y C, será considerado como borde final; el borde que incluye el punto B no será considerado ya que se encuentra dentro del rango del umbral minVal-maxVal, y no tiene conectividad con algún borde por encima del umbral máximo.

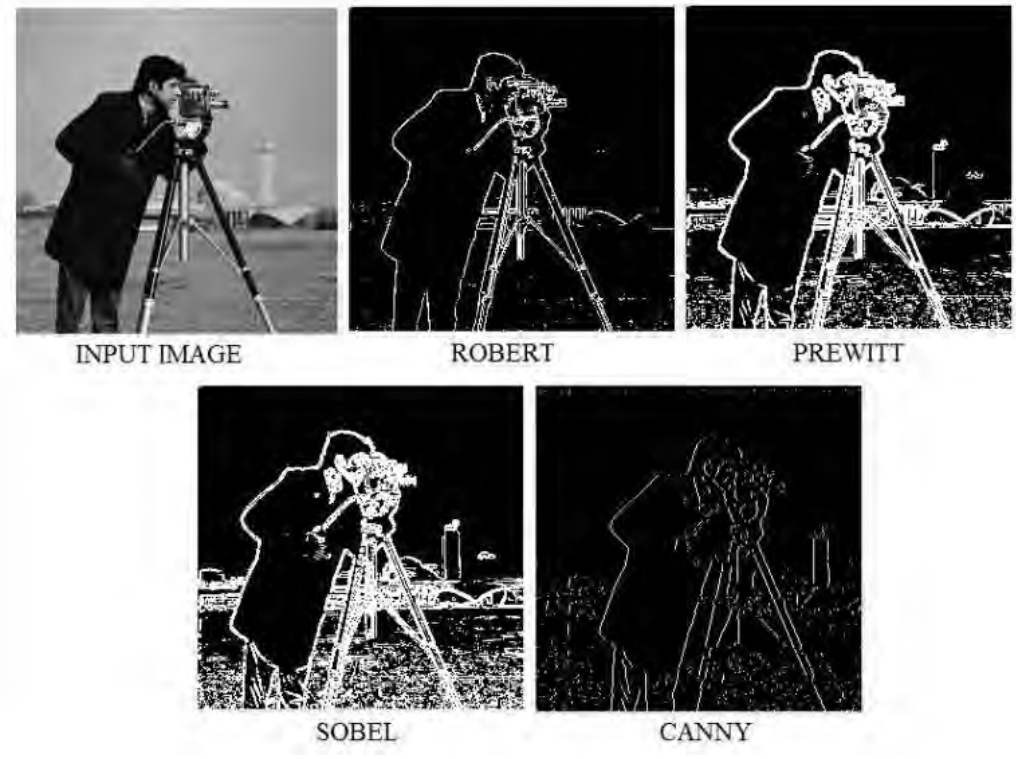

Figura 2-9: Ejemplo de la operación del algoritmo canny con respecto a los demás filtros paso altos. Imagen recuperada de [Mahalle y Shah, 2017]

Si comparamos el filtro Sobel y el algoritmo Canny podemos notar que el método es más preciso (Figura 2-9). A través del filtro de Canny es posible detectar figuras convexas (contornos) que permitan localizar los elementos en una imagen y extraer características locales.

2.1.2.4. Filtros morfológicos

A partir de imágenes binarizadas, sobre todo en resultados de detección de bordes y contornos, el uso de estos filtros nos permite eliminar ruido dado por líneas excedentes, así como ser auxiliar en completar figuras abiertas. Para ello, existen técnicas como dilatación, erosión, combinación de ambas, etc. Se detallarán a continuación las utilizadas en este trabajo de tesis.

Dilatación La operación de dilatación, incrementa el área de las regiones binarizadas a través de un kernel definido. La lógica detrás de esto es ir pasando un kernel por toda la imagen y

cada vez que se encuentre al menos un valor de 1 dentro de la región del kernel, toda la región abarcada por este tendrá el mismo valor 1. Esto permite crecer el área de los bordes de la imagen, que además de intensificar el borde, permitirá cerrar extensiones para identificar contornos dentro de la imagen (Figura 2-10).

Erosión La operación de erosión sigue una lógica similar a la operación de dilatación. En este caso sucede el resultado contrario, es decir, esta operación reduce el área de las regiones binarizadas a través de un kernel de tamaño definido. Este va pasando por toda la imagen y cada vez que se encuentre al menos un valor de 0 dentro de la región del kernel, toda la región abarcada por este tendrá el mismo valor de 0 (Figura 2-10).

Figura 2-10: Ejemplo de operaciones de filtros morfológicos. Recuperada en octubre 2021 de: https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_morphological_ops/py_morphological_ops.html

2.1.2.5. Transformaciones geométricas en imágenes

Además de los filtros de convolución y las modificaciones en los espacios de color, es posible aplicar a las imágenes transformaciones geométricas que nos permitan modificarlas cambiando la ubicación de los pixeles [García *et al.*, 2015], esto es, partiendo de la imagen de entrada, y dada una función de mapeado, obtendremos las nuevas posiciones de los pixeles que corresponderán a la imagen de salida como se muestra en la ecuación 2-9.

$$O(x, y) = I(f_x(x, y), f_y(x, y)) \tag{2-9}$$

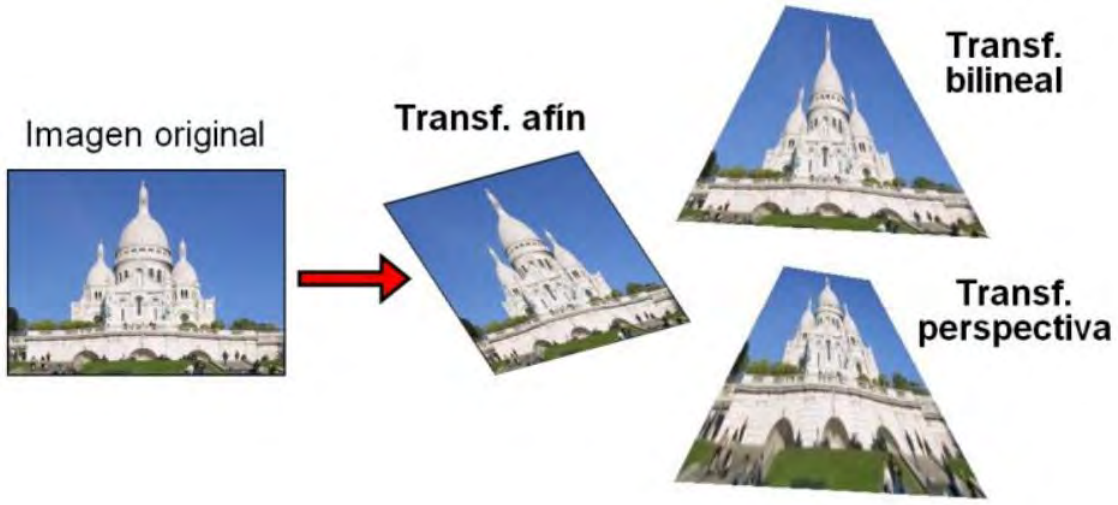

Figura 2-11: Ejemplo de transformaciones afines, bilineal y de perspectiva. Recuperada de [Villamarín, 2015]

A partir de la ecuación 2-9, pueden surgir dos problemas: extrapolación e interpolación, en donde la primera ocurre cuando la función de mapeado resulta en un valor fuera de los límites de la imagen original; y la segunda se debe a que las nuevas posiciones obtenidas de los pixeles están dadas por valores reales, es decir, números con punto flotante, por lo que existen diferentes estrategias para lograr pasar esos valores al espacio de número enteros, tales como redondear al entero más cercano (*Nearest-neighbor interpolation*), interpolación bilineal, entre otros [García *et al.*, 2015].

Entre los tipos de transformaciones geométricas más utilizadas a partir de una única imagen, se encuentran las transformaciones afines (rotación, reflexión traslación, escala, etc.) que conservan las líneas paralelas (no existe deformación en las formas); y las transformaciones bilineales y de perspectiva, que se basan en plasmar en el plano de la imagen una proyección perspectiva del mundo real (Figura 2-11). Esto último, nos permite corregir deformaciones u orientaciones obtenidas por la posición y características de los sensores, siendo la corrección de

la perspectiva una de las estrategias utilizadas para tratar este tipo de problemas.

Corrección de perspectiva Una interpretación de cómo vemos el mundo se puede representar a través de la geometría proyectiva, en donde líneas paralelas se cortan en un punto en el infinito. Para que un sistema de visión artificial reconozca estas líneas como paralelas y por tanto interpretar primitivas (bordes, figuras, etc.), es necesario utilizar transformaciones de perspectiva (Figura 2-12).

Figura 2-12: Proceso de corrección de perspectiva a partir de 4 pares de puntos conocidos. Ejemplo recuperado en octubre 2021 de: https://www.youtube.com/watch?v=fVJeJMWZcq8&list=PL_PCEPK-EoGsxS3Rctp5xSTtVVLKk7fwo&index=12&t=1473s

Para corregir la perspectiva de una imagen, debemos encontrar la matriz que permita transformar los puntos de la imagen de entrada (X) en los nuevos puntos de la imagen de salida (x), la cual es conocida como la matriz de homografía (H) (ecuación 2-10).

$$\mathbf{x} = H\mathbf{X}$$
$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{2-10}$$

Por lo que a partir de la ecuación 2-10, se necesita encontrar las 8 incógnitas que representan los valores de la matriz de homografía. Dado que cada punto en el plano está dado por dos ecuaciones que corresponden a los ejes coordenados (ecuación 2-11), la ecuación anterior 2-

10, se puede representar como un sistema de ecuaciones lineales como se muestra en 2-12, en donde la matriz A representa los 8 puntos conocidos y el vector columna h corresponderá a las incógnitas de la matriz H, por lo que sólo es necesario conocer cuatro puntos de entrada y cuatro de salida para obtener los valores de la matriz de homografía.

$$\begin{aligned} x &= \frac{h_{11}X + h_{12}Y + h_{13}}{h_{31}X + h_{32}Y + 1} \\ y &= \frac{h_{21}X + h_{22}Y + h_{23}}{h_{31}X + h_{32}Y + 1} \end{aligned} \tag{2-11}$$

$$A\mathbf{h} = \mathbf{x}$$

$$\begin{bmatrix} X_1 & Y_1 & 1 & 0 & 0 & 0 & -x_1X_1 & -x_1Y_1 \\ 0 & 0 & 0 & X_1 & Y_1 & 1 & -y_1X_1 & -y_1Y_1 \\ X_2 & Y_2 & 1 & 0 & 0 & 0 & -x_2X_2 & -x_2Y_2 \\ 0 & 0 & 0 & X_2 & Y_2 & 1 & -y_2X_2 & -y_2Y_2 \\ X_3 & Y_3 & 1 & 0 & 0 & 0 & -x_3X_3 & -x_3Y_3 \\ 0 & 0 & 0 & X_3 & Y_3 & 1 & -y_3X_3 & -y_3Y_3 \\ X_4 & Y_4 & 1 & 0 & 0 & 0 & -x_4X_4 & -x_4Y_4 \\ 0 & 0 & 0 & X_4 & Y_4 & 1 & -y_4X_4 & -y_4Y_4 \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \end{bmatrix} = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{bmatrix} \tag{2-12}$$

Teniendo los valores de la matriz H, es posible aplicar la transformación a cada punto de la imagen de entrada, por lo que si se desea corregir las líneas paralelas en una imagen, deberemos identificar puntos conocidos que pertenezcan a una recta en el plano proyectivo, los puntos destino cuya relación corresponden a líneas paralelas en el espacio euclídeo y a través de estos valores conocidos, obtener la matriz H para posteriormente aplicar la transformación a todos los puntos de la imagen (Figura 2-12).

Para resolver este sistema de ecuaciones podemos utilizar el método SVD para $AH = 0$, teniendo la restricción de $|H| = 1$, para evitar la solución obvia de que todos los valores de h serían cero. Por tanto, H se obtendrá a partir de $A = U\Sigma V^T$, en donde sus valores

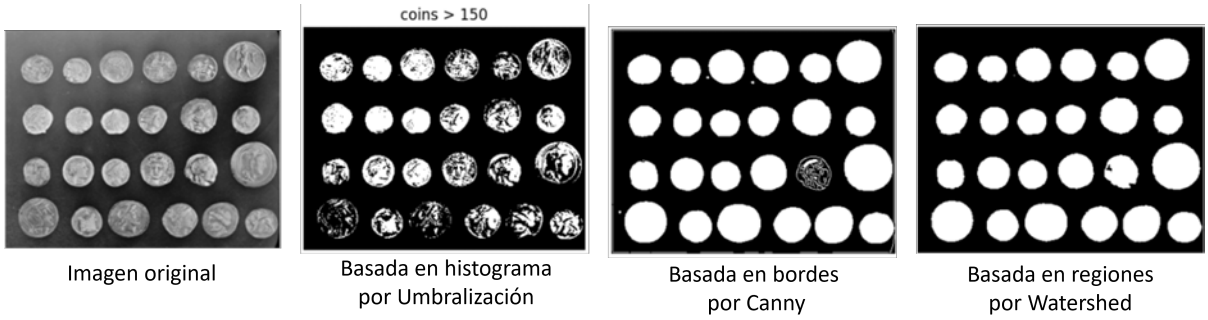

Figura 2-13: Ejemplo de segmentación a través de técnicas basadas en pixeles, bordes y regiones. Ejemplo extraído en octubre 2021 de: `https://scikit-image.org/docs/dev/auto_examples/applications/plot_coins_segmentation.html`

corresponderán a la última columna de V ya que sería el vector singular que se necesita [Demmel, 1997].

2.1.3. Segmentación

A través del sistema de percepción visual, somos capaces de separar con gran precisión los elementos (o regiones) que se encuentran en el entorno. Este proceso tiene como objetivo identificar los objetos de interés para su posterior análisis o interpretación. En términos computacionales, esto es conocido como segmentación, en donde cada pixel de la imagen pertenecerá una determinada región (o segmento) y ha sido utilizado como paso previo para localización, seguimiento, medición y reconocimiento (o identificación) de elementos de interés en diferentes campos como la medicina [Ramesh *et al.*, 2021], industria [Jayanthi y Shashikumar, 2017], compresión de archivos [Maheswari y Radha, 2011], entre otros.

Su presencia en la visión por computadora ha sido uno de los problemas de investigación más estudiados a lo largo del tiempo, debido a la complejidad que se puede llegar a presentar dependiendo de los dispositivos utilizados, es decir, si usamos uno o más sensores (o imágenes desde diferentes ángulos a través de un sólo sensor), el error en la separación será menor dado que estaríamos obteniendo información espacial (profundidad) [Marco-Antonio, 2019]. El mayor desafío se encuentra cuando sólo se tiene una imagen a través de un sensor de color en posición fija, donde los métodos de segmentación tradicionales se pueden dividir en tres categorías [Jong

y Meer, 2004]:

- Basado en pixeles, en donde se dividen en regiones a través de la umbralización de los espacios de color y no se basa en un contexto o conocimiento previo, es decir, está únicamente relacionado al histograma de una imagen (Figura 2-13)

- Basado en bordes, en los cuales se enfocan en los cambios de intensidad que puedan formar contornos o figuras cerradas (Figura 2-13)

- Basado en regiones, cuyas técnicas dependen de la vecindad de los pixeles en los diferentes espacios de color, utilizando técnicas como crecimiento de regiones, watershed [Szeliski, 2004], entre otros (Figura 2-13)

El resultado de una correcta segmentación dependerá de las características que se presentan en el entorno o del tipo de aplicación. Las técnicas basadas en bordes se asemejan a la forma en la que podemos distinguir los objetos, sobre todo en imágenes con alto contraste. Una ventaja de la detección de bordes es que nos aporta información respecto a las líneas, esquinas, curvas presentes en la imagen en donde podemos reducir la cantidad de información a procesar en las etapas siguientes.

Una de las técnicas más utilizadas es el algoritmo de Canny que comparado con otros métodos de detección de bordes arroja el mejor resultado [Dubey y Gupta, 2018]. Sin embargo, utilizar canny para segmentación por sí solo, no suele obtener muy buena precisión en la identificación de estos bordes de imágenes con poco contraste, por lo que se suele utilizar modificaciones como detectores de bordes adaptables [Kaganami y Beiji, 2009].

Además de las técnicas tradicionales de segmentación, existen otros tipos de métodos actuales que se basan en diferentes modelos, incluyendo aprendizaje máquina como redes neuronales en donde es necesario un entrenamiento o conocimiento previo [Adams *et al.*, 2020; Arganda-Carreras *et al.*, 2017] y cuya desventaja es la necesidad de un gran número de muestras para lograr un buen resultado.

Detección de casas

Reconocimiento de casa de un individuo

Figura 2-14: Diferencia entre detección de objetos y reconocimiento de instancias. Imagen extraída en octubre 2021 de: https://www.pxfuel.com/es/free-photo-xwhrj

2.1.4. Reconocimiento de Objetos

Uno de los objetivos finales en visión artificial es lograr reconocer uno o más objetos en una imagen. Llamaremos objeto, a todo elemento de interés con características específicas que formará parte de un conjunto de características similares, es decir, una clase, en donde, dependerá del contexto para determinar el nivel específico de las clases que se desean reconocer [Szeliski, 2004], es decir, no es lo mismo identificar una casa dentro de un conjunto de casas (reconocimiento de instancias), que reconocer si existen una o más casas dentro de una imagen (detección de objetos) como se muestra en la figura 2-14. Además, es posible combinar ambos conceptos, variando el orden en la aplicación para filtrar los elementos de interés en la imagen, por lo que el proceso de reconocimiento estará compuesto por estos dos componentes fundamentales conocidos como clasificación y detección de objetos [Russakovsky *et al.*, 2015].

- *Image Clasification* (Clasificación de la imagen), cuya tarea es predecir el tipo de clase que pertenecen los objetos en la imagen
- *Objet Detection* (Detección de Objetos), que se basa en localizar los objetos en una imagen con base en una o más clases de interés

El problema de reconocimiento ha sido abordado por medio de diferentes técnicas a lo largo del tiempo, en donde, se han presentado enfoques a un nivel bajo de procesamiento

como *template matchine* [Cole *et al.*, 2004], así como técnicas de *Machine Learning o ML* (aprendizaje máquina) tales como *SVM* (Máquinas de vector de soporte), *Neural Networks* (Redes Neuronales), entre otros [Athira y Khan, 2020].

Tradicionalmente se necesita una etapa inicial conocida como extracción de características que se puede obtener a través de la aplicación de diferentes filtros o técnicas más avanzadas tales como *SIFT*, *HOG* [Athira y Khan, 2020].

Sin embargo, con el surgimiento de las redes neuronales (Apéndice B) y uno de sus modelos dentro del área de *Deep Learning o DL* (Aprendizaje Profundo) conocido como *Convolutional Neural Networks o CNN* (Red Neuronal Convolucional) la etapa de extracción de características se incluye dentro del modelo de clasificación, permitiendo mayor precisión y rapidez a la hora de reconocer los objetos, sobre todo, en tiempo real [Athira y Khan, 2020].

2.1.4.1. Redes Neuronales Convolucionales

Uno de los modelos que permite resolver los problemas de reconocimiento, detección y segmentación en imágenes es la reconocida *Convolutional Neural Network (ConvNet o CNN)*, cuyos inicios fueron utilizados en el reconocimiento de caracteres con un resultado bastante satisfactorio [Le Cun *et al.*, 1997] dado que son espacialmente invariantes. El principio de esta red se asemeja a la forma en la que nosotros interpretamos el entorno a través de nuestros ojos, esto es, por medio de la interpretación de primitivas tales como bordes, líneas, contornos, colores, de modo que la combinación de estos elementos nos permita reconocer formas y asociarlas a objetos que conocemos.

Precisamente, la peculiaridad de este modelo de red comparado, por ejemplo, con un modelo *MLP* (Apéndice B) se encuentra en la adición de un conjunto de capas que nos permitirán obtener características en una imagen a través de filtros de convolución (Apéndice A.1) y cuya salida será la entrada a un modelo de red completamente conectado que servirá para clasificar con base en un banco de conocimiento, aprendido por entrenamiento (Figura 2-15).

La cantidad de neuronas en cada capa varía dependiendo de la aplicación. Existen diversas arquitecturas de *CNN* tales como *Inception-ResNets* Szegedy *et al.* [2017], *ResNeXt-50* [Xie

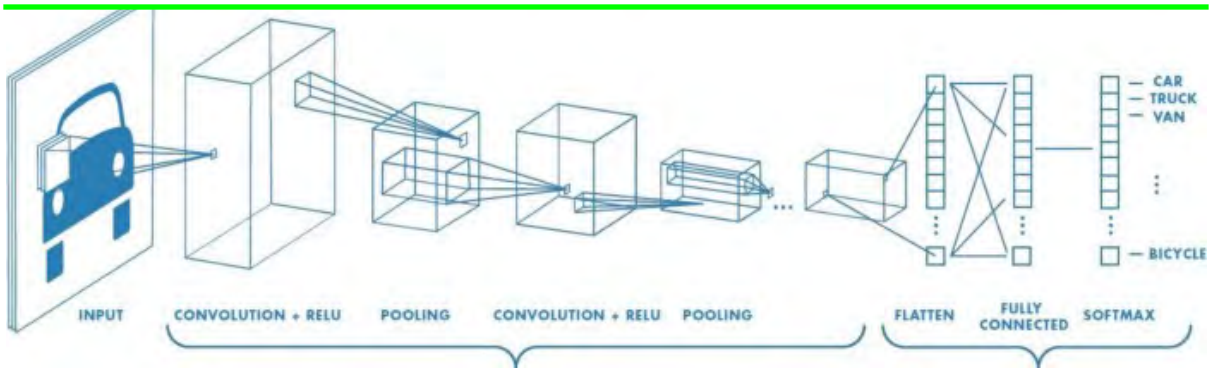

Figura 2-15: Ejemplo de estructura básica de una Red Neuronal Convolucional. Imagen extraída en octubre 2021 de: https://medium.com/swlh/computer-vision-with-convolutional-neural-networks-22f06360cac9

et al., 2017], así como diferentes modelos o variaciones de CNN tales como *R-CNN* (*Region-Based Convolutional Neuronal Network*) [Girshick *et al.*, 2014] y *YOLO* (*You Only Look Once*) [Redmon *et al.*, 2016] que se basan primero en separar la imagen en regiones para reducir el costo computacional y poder utilizarlo en tiempo real.

No obstante, a pesar de los diferentes modelos de *CNN* que existen, tenemos que, dentro de la arquitectura básica de una *ConvNet* se encuentran 4 operaciones principales:

- Convolución
- Función de activación $ReLU$
- Submuestreo (*Subsampling o Pooling*): A través de esta operación se reduce la dimensión espacial del volumen obtenido por la convolución. Existen diferentes
- Clasificación (Modelo de red MLP)

Capa de convolución Por medio de estas capas, se irá aumentando la profundidad de datos a partir de la imagen de entrada a través de diversos filtros que permitan extraer diferentes características (Imagen 2-16).

Es muy común utilizar más de un filtro, ya que aumentará la probabilidad de un reconocimiento satisfactorio.

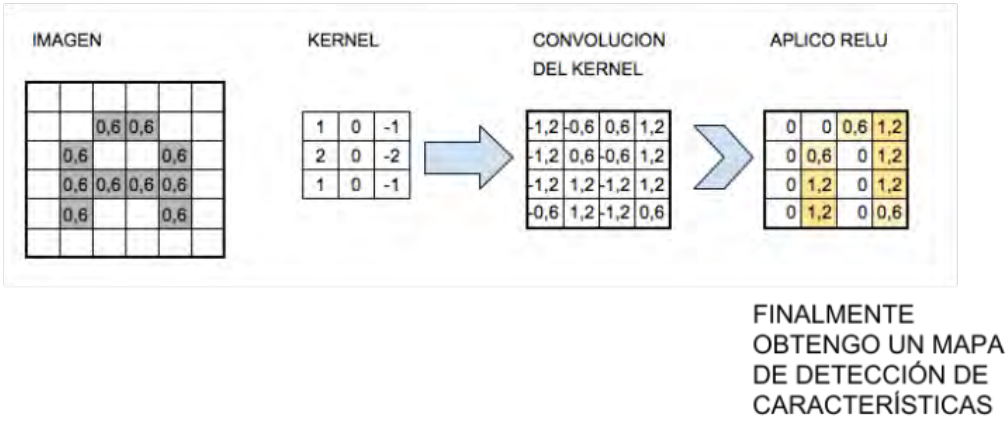

Figura 2-16: Ejemplo de convolución aplicada a una imagen. Por medio de la función de activación ReLU se tiene el mapa de características. Los filtros irán adoptando diversas variaciones para poder obtener diferentes mapas de características. Ejemplo recuperado en octubre 2021 de: https://www.juanbarrios.com/redes-neurales-convolucionales/#La_funcion_de_Activacion

Función de activación Correspondiendo con el modelo básico de una red neuronal, la salida de todas las neuronas deberá ser procesada por una función de activación.

Tanto para las capas de convolución como para el *pooling*, la función más utilizada es la conocida como $ReLU$, dada por la función $f(x) = max(0, x)$, es decir, solo permite en la salida valores positivos.

Submuestreo o Pooling Al aplicar diferentes filtros en la capa de convolución, tendríamos un número bastante grande de neuronas correspondientes a cada una de las imágenes resultantes. Por lo que aplicar la etapa de submuestreo es muy útil ya que nos permite conservar las características más importantes dadas por las imagenes filtradas. Existen diversos tipos de submuestreo, sin embargo, el más utilizado es max-pooling, que se basa en obtener el valor máximo en donde se encuentre el kernel. Así por ejemplo si quisiéramos aplicar max-pooling después de una capa de convolución, tendríamos que el número de neuronas se reduciría a un cierto porcentaje 2-17. El costo computacional de entrenamiento dependerá de la cantidad de neuronas que tendrá la red, por lo que utilizar submuestreo permitirá agilizar el proceso tanto

de entrenamiento como de clasificación.

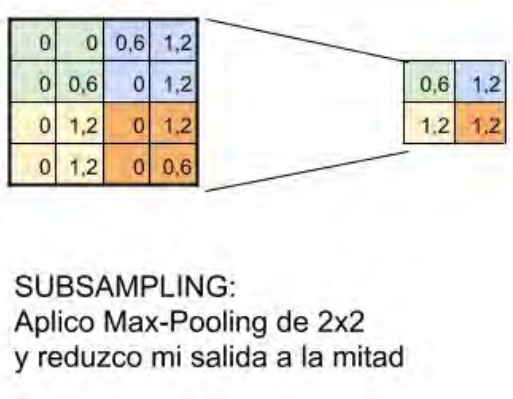

Figura 2-17: Reducción de las dimensiones del mapa de características a partir de max-pooling. Ejemplo recuperado en octubre 2021 de: https://www.juanbarrios.com/redes-neurales-convolucionales/#La_funcion_de_Activacion.

2.1.4.2. Aumento de datos en Imágenes

Una desventaja del uso de *CNN* es la cantidad de muestras necesarias en el entrenamiento para poder lograr una mayor precisión en los resultados de la clasificación [Shorten y Khoshgoftaar, 2019].

Sin embargo, en la mayoría de las situaciones resulta complejo obtener muchas muestras reales para el entrenamiento de una red, por lo que una de las soluciones que suele utilizarse es conocida como aumento de datos, que es un conjunto de técnicas que generan imágenes nuevas a partir de las muestras originales variando por ejemplo su forma, sus propiedades en los colores e incluso agregando ruidos artificiales que permitan que la red sea más robusta y logre interpretar las imagenes en entornos no controlados.

Estas variaciones se pueden obtener ya sea por medio de métodos tradicionales como por técnicas de aprendizaje profundo, por lo que se pueden clasificar de la siguiente manera [Shorten y Khoshgoftaar, 2019]:

- Manipulaciones básicas:
    - Filtros convolucionales

- Transformaciones geométricas
- Transformaciones en el espacio de color
- Borrado aleatorio
- Mezclado de imágenes

- Manipulaciones a través de *Deep Learning*:
  - Entrenamiento adversativo (*Adversarial training*)
  - Transferencia de estilo (*Neural Style Transfer*)
  - Aumentación a través de Red Generativa Antagónica *(GAN)*

Dada la simplicidad en las operaciones y la similitud de sus variantes con escenarios reales, las transformaciones geométricas juegan un rol muy importante y es una de las técnicas más utilizadas en la generación de nuevas imágenes para entrenamiento [Shorten y Khoshgoftaar, 2019].

2.2. Realidad Virtual

A lo largo del tiempo, el término de Realidad Virtual (*Virtual Reality o VR*) ha tenido diversas definiciones que han intentado adaptarse a una técnica que aún no ha terminado de lograr su máximo potencial. La incorporación de la palabra "Realidad" involucra incluso cuestiones filosóficas, y se puede definir como algo que es verdad o existe [RAE, 2014], lo que conlleva a cierta contradicción al incorporar la palabra "Virtual", ya que significa algo que no es real o artificial [RAE, 2014].

Por lo que su combinación, nos llevaría entonces a un escenario ideal, donde si dispusiéramos de infinitos recursos computacionales podríamos definir Realidad Virtual como una especie de máquina del tiempo o de sueños, que fuese capaz de recrear artificialmente espacios tridimensionales con un grado de similitud imperceptible con la realidad [Martínez *et al.*, 2011], es decir, un sistema que sea capaz de engañar tanto a nuestros sentidos interoceptivos como los exteroceptivos de tal modo que se pueda considerar como realidad.

Sin embargo, la limitante en los recursos informáticos nos obliga a darle un sentido más realista al concepto de esta tecnología, como se indica en [Mihelj *et al.*, 2014] que definen VR como:

> *"La Realidad Virtual se basa en una simulación por computadora de manera interactiva, que detecta el estado y la operación del usuario y reemplaza o aumenta la información de retroalimentación sensorial a uno o más sentidos de manera que el usuario tenga la sensación de estar inmerso en la simulación (Entorno Virtual)."*

Así, todo sistema VR deberá constar de los 4 elementos básicos siguientes:

- **El entorno virtual (*Virtual Environment*)**: Determinado por los objetos y elementos dinámicos, es un entorno generado artificialmente por una computadora donde cada elemento debe obedecer ciertas reglas y relaciones entre esos objetos [Stanney, 2001]
- **Presencia virtual (*Virtual Presence*)**: Se puede dividir en dos estados cuya interconexión genera un estado de presencia elevada (inmersión) [1]:
  - **Presencia física:** Se presenta a través de la correspondencia entre las partes reales del cuerpo y las virtuales del usuario que lo representan en el entorno virtual, así como su posición espacial en ese mundo.
  - **Presencia mental:** Se involucran elementos psicológicos y de percepción de cada individuo, representando un estado de "trance", esto es, generación de expectativas y sensación de formar parte del mundo virtual.
- **Retroalimentación sensorial (*Sensory Feedback*)**: El sistema deberá dar una respuesta al usuario a través de la estimulación de sus sentidos, esto es, por medio de elementos visuales, auditivos, e incluso háptico.

[1] Sin embargo, existen situaciones en donde la presencia del propio usuario no es tomada en cuenta debido a que se intentan recrear situaciones externas al mismo. Por ejemplo, algunas novelas requieren cierto desapego del espectador [Mihelj *et al.*, 2014]

- **Interactividad (*Interactivity*)** Para que un sistema VR sea completo, deberá responder a las acciones que realice el usuario en el entorno de alguna forma. Esto podría ser, a través de la modificación espacial de los objetos en el entorno así como permitirle al usuario desplazarse alrededor del mismo.

2.2.1. Breve Historia

La invención de un dispositivo o medio que fuese capaz de generar una realidad artificialmente era sólo contemplado como un tema de ciencia ficción. No fue hasta 1957, que el inventor norteamericano Morton Heilig, considerado por algunos como el padre de VR, desarrollara y patentara el precursor de la VR conocido como *Sensorama* [Heilig, 1992].

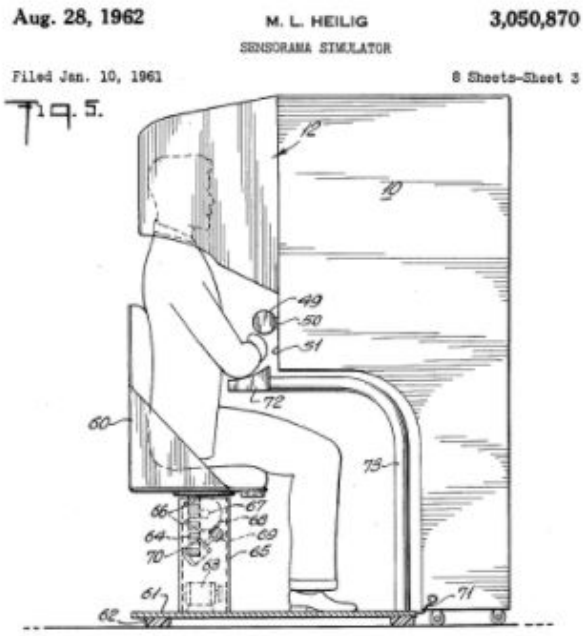

Figura 2-18: Sensorama. Imagen extraída de la patente US3050870A, en: https://patents.google.com/patent/US3050870A/en

Esta máquina, parecida a una cabina de fotos (Figura 2-18), simulaba un recorrido en bicicleta por la ciudad, en donde se mostraban las imágenes en 3D a través de un visor estereoscópico, estimulando los sentidos auditivos, hápticos y olfativos, ya que el asiento vibraba

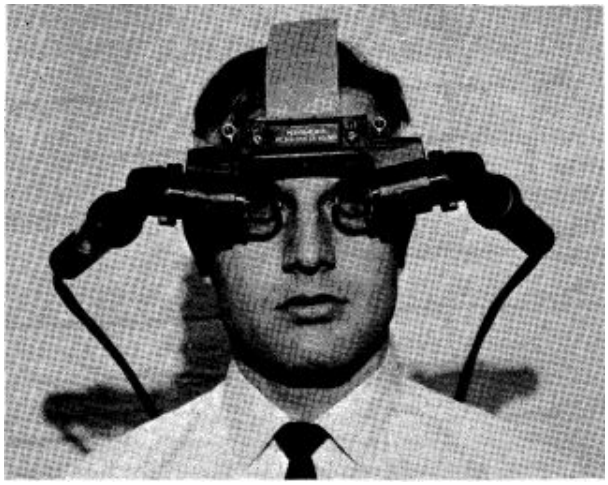

Figura 2-19: Sword Of Damocles. Imagen extraída de [Sutherland, 1968]

y poseía pequeños ventiladores que permitían simular el viento. Sin embargo, este dispositivo nunca logró comercializarse.

Años más tarde, en 1968 Ivan Sutherland, desarrolló un dispositivo, llamado *Sword of Damocles* [Sutherland, 1968], que se ubicaba directamente en la cabeza (*Head-mounted Display*) y que a través de unos lentes y dos pantallas (tubos de rayos catódicos monocromáticos) ubicadas una en cada ojo, se podía visualizar un entorno virtual dependiendo de la orientación de la cabeza del usuario (Figura 2-19). Sin embargo, dado que la pantalla era transparente se podía visualizar el entorno físico lo cual no permitía la completa inmersión en el entorno. Esto dio paso a otra tecnología conocida actualmente como Realidad Aumentada, que precisamente se basa en la superposición de elementos virtuales en nuestro mundo físico.

Tanto *Sensorama* como *Sword of Damocles* carecían de uno de los elementos básicos para ser considerado lo que hoy se entiende como VR: interacción. No fue hasta alrededor de 1970 que Myron Krueger desarrolló una serie de entornos virtuales (como *Videoplace*) que permitía la interacción a través de varios sensores, cuyo objetivo era poder mover objetos en el mundo virtual. Mas adelante, se desarrollaron sistemas que incluía retroalimentación háptica a través de una pantalla y guantes especiales, como por ejemplo *Grope III* que se enfocaba en visualizar moléculas a gran escala; o *Sayre Glove* que permitía reconocer los movimientos de los dedos de las manos, entre otros.

Figura 2-20: *Virtual Visual Environment Diplay* o *VIVED* desarrollado por la Nasa. Imagen extraída en Noviembre 2021 de: https://www.engadget.com/2013-12-15-time-machines.html

En 1985, la NASA desarrolló el primer casco comerciable estereoscópico *HMD* con gran amplitud en su campo de visión llamado *Virtual Visual Environment Diplay* o *VIVED* (Figura 2-20), que permitió dar a conocer la tecnología a diferentes empresas comerciales. En ese mismo año, Jaron Lanier acuñó el término de "Realidad Virtual" además de desarrollar guantes hápticos, HMDs y software para aplicaciones VR, incluyendo el Dataglove para la NASA, que tenía sensores ópticos para medir la flexión en los dedos.

La década de los 90 fue un estallido respecto la investigación de diferentes aplicaciones y experimentación de escenarios de localización espacial en VR, sobre todo en el entretenimiento, en donde varias compañías estuvieron involucradas (Virtuality, Division, Sega, Disney, entre otras), así como universidades y cuerpos militares. Uno de los desarrollos más significativos y famosos en los años 90s, era el conocido como *CAVE* (*Cave Automatic Virtual Environment*), que como se muestra en la figura 2-21 se basa en un cuarto vacío, en donde se muestra la información virtual en cada una de sus superficies (paredes y techo). Esto permitía al usuario cierta inmersión, sobre todo con el uso de lentes especiales para dar la ilusión de profundidad. Contaba además, con sensores electromagnéticos que permitían obtener el movimiento del usuario. Sin embargo, dado los recursos informáticos y altos costos de los equipos VR no logró el resultado esperado con el público general.

En los primeros años del siglo XXI hubo un período de incertidumbre sobre el futuro de

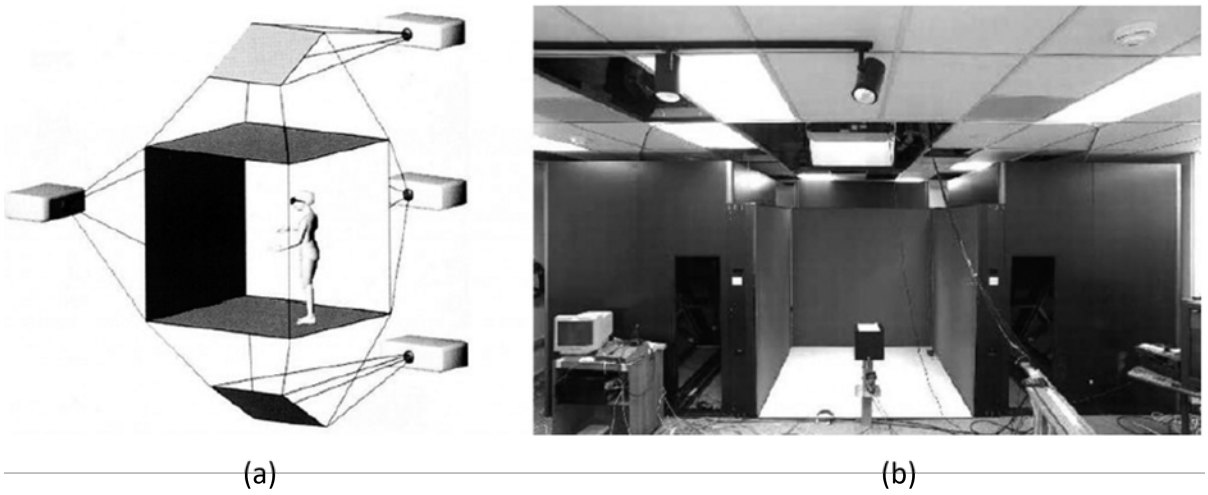

Figura 2-21: Ejemplo conceptual y real de un escenario CAVE. (a) Imagen recuperada de [Cruz-Neira *et al.*, 1992]. (b) Imagen recuperada de [Daily *et al.*, 1999]

VR, sobre todo por la limitante que tenían los HMDs y su corto campo de visión. Esto impedía al usuario experimentar la sensación de estar inmerso en un mundo virtual, por lo que, en 2006, Mark Bolas e Ian McDowall crearon un casco con un campo de visión de 150^o, llamado Wide5, en donde se estudió el efecto del campo de visión con respecto a la experiencia del usuario. Esto condujo a la creación de un equipo de bajo costo llamado *Field of View To Go (FOV2GO)*, siendo el precursor de la mayoría de los HMDs de hoy en día.

Finalmente, uno de los dispositivos más utilizados por la comunidad científica y público en general es conocido como Oculus y todas sus versiones (Figura 2-22), fundado por Palmer Luckey en 2012, cuyo objetivo principal era crear un casco bajo costo y que fuera efectivo comparado a lo que se tenía en el momento. Dos años más tarde Facebook compró la compañía y actualmente es uno de los cascos más famosos del mercado permitiendo el desarrollo de aplicaciones VR en diferentes áreas de la ciencia.

Figura 2-22: Modelo de HMDs de la empresa Oculus por Facebook. Extraído en Noviembre 2021 de https://about.fb.com/news/2021/03/five-years-of-vr-a-look-at-the-greatest-moments-from-oculus/

2.2.2. Aplicaciones de la Realidad Virtual

Aunque actualmente VR está muy asociada al entretenimiento, sobre todo por la gran demanda que se tiene de los diferentes dispositivos VR. Además, el uso de esta tecnología como método de entrenamiento o simulación de actividades de riesgo ha sido de gran utilidad en diferentes áreas.

Una de las primeras aplicaciones VR fueron implementadas en el ámbito militar, sobre todo en la sustitución de los costosos simuladores de vuelo de origen analógico, cuyo objetivo al usar VR, además de reducir costos, era permitir adaptar la interfaz a diferentes modelos de aviones sin necesidad de cambiar elementos físicos [Valentino *et al.*, 2017]. Otro tipo de aplicaciones militares fueron la base para los sistemas VR complejos que se tienen actualmente, incluyendo simuladores de manejo de autos [Taheri *et al.*, 2017], prácticas de tiro en VR [Bhagat *et al.*, 2016], donde se involucran variables externas que permiten simular situaciones más reales.

En el ámbito de la medicina, el entrenamiento de las prácticas médicas resulta un paso fundamental antes de ejercer la carrera. Es importante proporcionar escenarios cuasi reales que

permitan experimentar sin el riesgo de sacrificar vidas humanas, por lo que actualmente se tienen sistemas VR complejos, que permiten, por ejemplo, simular diferentes tipos de cirugía [Aïm *et al.*, 2016], así como visualizar un modelo 3D real del paciente de modo que se tenga conocimiento previo de su anatomía [Satava, 1995] y poder obtener resultados satisfactorios al momento de la aplicación. Además, en el campo de la rehabilitación, el uso de la VR ha incrementado el interés en los ejercicios que debe hacer el paciente a través de juegos en ambientes virtuales, generando mayor eficiencia que los métodos tradicionales [Howard, 2017]

En el diseño y la visualización encontramos aplicaciones que recrean ambientes reales históricos o futuros proyectos de construcción. Un ejemplo de esto está dado con recorridos de edificios famosos [Guerra *et al.*, 2015] en donde el usuario tiene la posibilidad de caminar por las instalaciones e interactuar con los elementos dentro del mundo virtual.

Actualmente con la presencia de diferentes dispositivos a bajo costo, una nueva brecha de oportunidades se ha presentado en el mercado de entretenimiento para los usuarios finales, a través del desarrollo de diversos videojuegos para el hogar, así como atracciones en parques temáticos, museos virtuales, obras de arte interactivas, entre otros temas que permite dar una experiencia divertida y diferente [Spence, 2021].

2.2.3. Dispositivos de respuesta sensorial

Para que un sistema sea considerado como VR, deberá cumplir con los elementos básicos mencionados anteriormente, en donde se incluyen las respuestas sensoriales que estarán dadas a través de diferentes dispositivos correspondiendo a los sentidos de la vista, háptico, auditivo, olfativo y gustativo.

Por medio del sistema visual, podemos distinguir fácilmente y de manera casi instantánea elementos reales y virtuales, lo que lo convierte en un componente básico y presente en todo sistema VR. Además, el tacto, ha sido otro de los sentidos que ha tenido una gran popularidad desde los inicios de VR, dado que aumenta el porcentaje de información que podemos percibir del entorno virtual, elevando el nivel de inmersión.

En este contexto de investigación, se mencionarán dispositivos con respuesta visual y háp-

tica, dado que fueron los sentidos abordados en el trabajo.

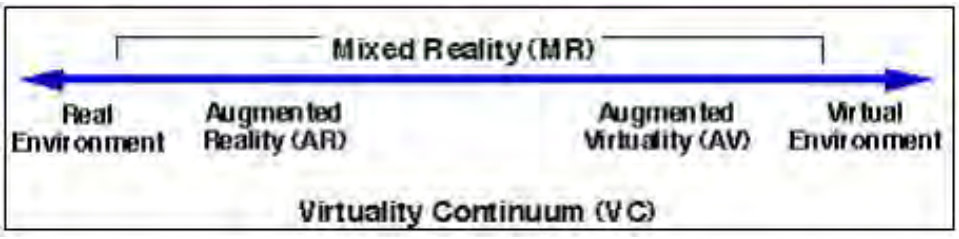

Figura 2-23: Continuo de la virtualidad expuesto por Milgram en [Milgram y Kishimo, 1994]

2.2.3.1. Respuesta visual

La importancia del sentido visual ha sido tal, que la mayoría de las tecnologías que involucran elementos virtuales se rigen en torno a este sentido a través del uso de diferentes dispositivos. Es por ello que existen situaciones en donde resulta difícil identificar el tipo de tecnología que se está utilizando.

De acuerdo con el "continuo de la virtualidad" definido por Milgram y Kishimo [1994], toda tecnología que contempla la visualización de elementos reales y virtuales es considerada como Realidad Mixta (Figura 2-23). En uno de estos extremos, encontramos el mundo completamente virtual en donde se cuenta con una aproximación de lo que se tiene actualmente de Realidad Virtual en términos visuales.

Mucho de los trabajos que clasifican los diferentes dispositivos VR contemplan el uso de pantallas planas para visualizar la información llamándolos dispositivos no inmersivos [Bamodu y Ye, 2013; Martínez *et al.*, 2011; Freina y Ott, 2015], sin embargo, combinar las palabras "no inmersivo" con una tecnología cuyo objetivo es lograr el mayor grado de inmersión posible resulta contradictorio, sobre todo con los recursos computacionales actuales. Además, según [Milgram *et al.*, 1999] a mayor número de elementos virtuales, mayor será la distinción entre Realidad Virtual y Realidad Mixta[2].

[2]Actualmente se ha debatido formalmente el concepto de Realidad virtual y su delimitación con Realidad Mixta, ya que si simuláramos todas las respuestas de los sentidos externos faltarían los sentidos interoceptivos (saciar el hambre, sed, etc) por lo cual seguirían existiendo cuestiones reales dentro del mundo virtual [Skarbez *et al.*, 2021]

Por lo que los dispositivos más utilizados que representan a la VR serían:

- *Room-based:* Basado en lugares cerrados con proyección virtual en la mayoría de las superficies visibles, en donde tenemos simulaciones de conducción de vehículos [Muhanna, 2015] y CAVE [de Back *et al.*, 2020]

- *Head-Mounted Display (HMD)*: Conocidos coloquialmente como lentes o cascos de VR.

HMD Los HMDs (Figura 2-24) se encuentran ubicados en la cabeza con el objetivo de abarcar todo nuestro campo de visión y reemplazarlo con contenido virtual. Además, a través de estos podemos obtener la orientación y ubicación espacial del usuario siendo posible navegar por el entorno virtual permitiendo la presencia física y mental, por lo que es considerado como la mejor herramienta visual de inmersión que se puede tener hasta el momento [Slater *et al.*, 2010]

Este tipo de dispositivo generalmente se puede dividir en 2 grandes categorías [Gilson *et al.*, 2011]:

- *Non-see-through HMDs (NST HMDs)* Cuyo objetivo es evitar que el usuario vea el entorno real, de manera que pueda proporcionar la inmersión más completa que se pueda. Este tipo de dispositivos es el ideal para aplicaciones de Realidad Virtual.

- *See-through HMDs (ST HMDs)* Combina elementos reales con virtuales, ya sea directamente o a través de cámaras. Esta tecnología está enfocada en la rama de la Realidad Mixta (en donde se incluye la Realidad Aumentada). Dentro de la misma existen dos tipos de tecnologías:

    - *Optical-see-through HMDs (OST HMDs)* Se utilizan superficies semitransparentes (por ejemplo, combinadores ópticos) de modo que a través de la proyección de elementos gráficos en esa superficie se combinen lo real y lo virtual.

    - *Video-see-through HMDs (VST HMDs)* Cuyo objetivo es capturar el entorno real a través de cámaras y proyectarlo en pequeñas pantallas situadas una en cada ojo

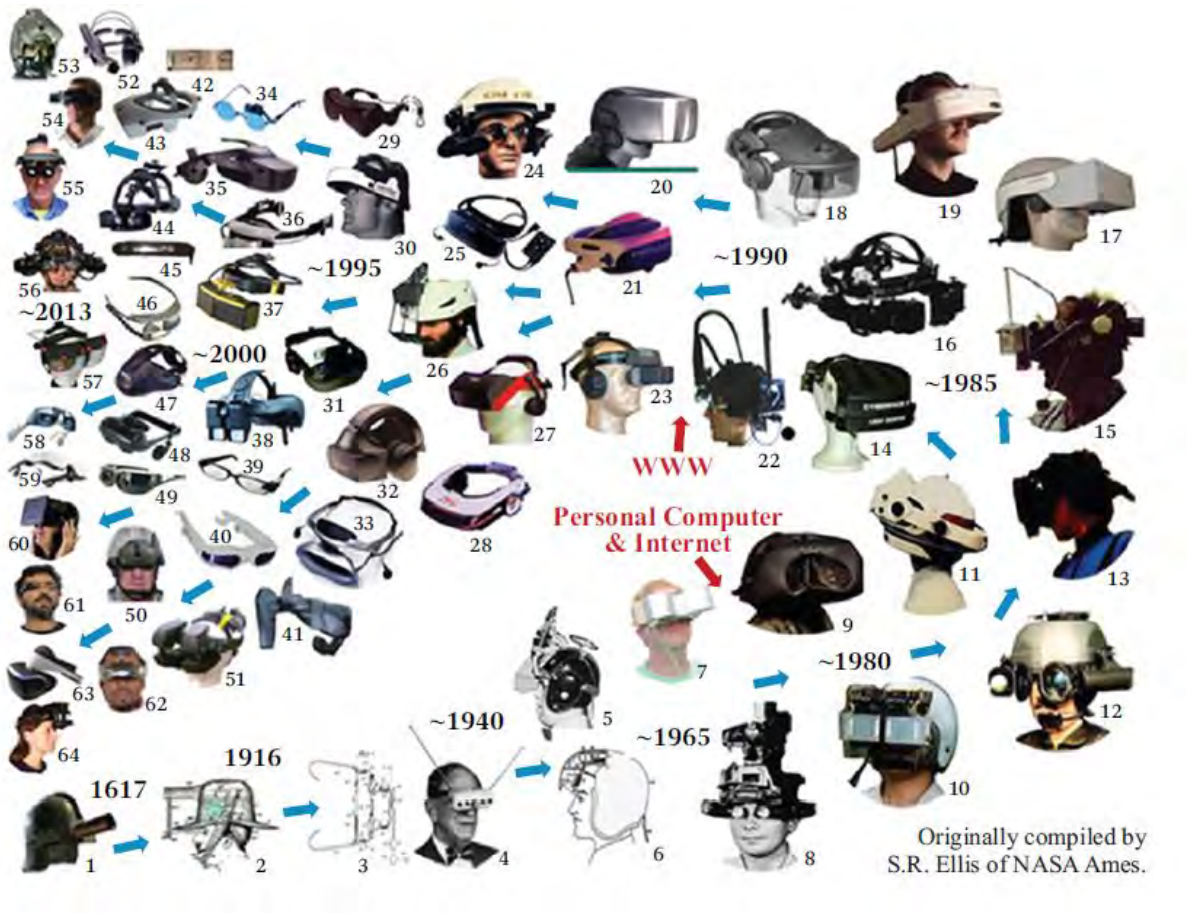

Figura 2-24: Evolución de los HMDs. Recuperado de [Ellis, 2014]

Dada la popularidad que ha estado teniendo VR sobre todo en el ámbito de los videojuegos, el costo de NST HMDs ha bajado de manera que muchos usuarios e investigadores tienen acceso al uso y desarrollo de sistemas VR [Wang y Lindeman, 2015]. Dentro de los dispositivos con mayor popularidad dado su grado de inmersión y resolución de pantallas elevados, tenemos a Oculus (con sus versiones Rift y Quest), HTC Vive y Sony Playstation VR [Kim *et al.*, 2020].

En particular, Oculus Rift (desarrollado por Facebook) luego de haberse introducido en el 2016 ha logrado según [Kim *et al.*, 2020] un 84 % mayor interés en el uso y desarrollo de HMDs similares dadas las características presentadas por Rift cuya tecnología permite 6 grados de libertad y un avanzado sistema de seguimiento con baja latencia y alta precisión.

La desventaja principal de este HMD es que el procesamiento depende de una computadora,

por lo que años más tarde se introdujo Oculus Quest, cuya capacidad computacional permite procesar gráficos sin depender de una. Además, su sistema es compatible con las funcionalidades de Rift, pudiendo desarrollar aplicaciones híbridas que dependan o no de las capacidades de un equipo externo conectado al mismo.

Actualmente, los Oculus Quest, son los únicos dispositivos que no dependen de elementos externos para poder utilizarlos. Además, su costo promedio es similar a lo que costaría una consola de videojuegos, por lo que la popularidad de VR ha incrementado surgiendo una nueva era de posibilidades al alcance de nuestras manos.

2.2.3.2. Respuesta háptica

Uno de los sentidos que nos aporta información adicional acerca de los elementos que percibimos en el entorno virtual es el conocido como sentido háptico. Recibir una respuesta física al tocar objetos virtuales puede mejorar el mapeo cognitivo del mundo virtual, haciendo que todo lo demás parezca más real [Jerald, 2015].

Los sistemas hápticos se pueden clasificar como *Passive* (objetos estáticos físicos) o *Active* (retroalimentación física a través de la computadora); *Tactile* (a través de la piel) o *Proprioceptive force* (a través de articulaciones o músculos); y *Self-grounded* (portables, se mueven con el usuario) o *world-grounded* (estático en el mundo real).

El método pasivo (*passive*) es capaz de proporcionar retroalimentación háptica a muy bajo costo, ya que se basa en coincidir objetos físicos reales con los elementos virtuales [Lindeman, 1999]. Estos objetos pueden adaptarse a diferentes formas, o bien ser una réplica similar en forma del elemento virtual, lo cual eleva el nivel de inmersión y mejora el rendimiento en situaciones donde se necesita realizar entrenamiento [Insko, 2001].

Otro de los métodos comúnmente utilizados por la mayoría de los sistemas VR comerciales, son dispositivos o controles que aportan retroalimentación háptica a través de vibración, palancas, etc.

Las demás técnicas suelen ser muy costosas y sólo son utilizadas para situaciones específicas en áreas como la medicina, sobre todo en entrenamiento para cirugías o rehabilitación, así como

cuestiones militares.

2.3. Interacción en Realidad Virtual

Así como los demás conceptos abordados en este trabajo de tesis, definir interacción resulta complejo ya que depende del contexto [Kiousis, 2002]. El área de *Human-Computer Interaction o HCI* (Interacción Humano-Computadora), específicamente en términos de un sistema VR [Jerald, 2015] se define como:

> *"La interacción es la comunicación que ocurre entre un usuario y una aplicación VR que está mediada a través del uso de dispositivos de entrada y salida"*

Por tanto, la comunicación Usuario-VR, estará dada a través de diferentes partes del cuerpo, en donde el uso de las manos nos permite abordar diferentes formas de interacción a través de diversos dispositivos, por lo que un sistema interactivo se puede dividir en equipos que son operados a través de las manos (*Hand input devices*) y dispositivos que no las usan (*Non-hand input devices*) [Jerald, 2015], como se muestra a continuación:

- *Hand Input Devices* (Dispositivos de entrada usando las manos)
    - *World-Grounded Input* Son dispositivos que se encuentran físicamente en el mundo real y son mapeados al mundo virtual, por ejemplo, palancas, joysticks, botones, volantes, que funcionan muy bien en situaciones en donde se requiere manipular controles virtuales.
    - *Non-tracked Hand-Held Controllers* Este tipo de dispositivo no es mapeado en el mundo virtual y permite la navegación e interacción de una manera indirecta (como controles de videojuegos).
    - *Tracked Hand-Held Controllers* Comúnmente poseen 6 grados de libertad (se conocen también como wands) y pueden también ser *Non-tracked Hand-Held Controllers*. Este tipo de dispositivos se puede utilizar para interactuar directamente a través de un

mapeo entre el control real y el modelo virtual de las manos. Usualmente se utiliza tecnología electromagnética, óptica o inercial para realizar el seguimiento de estos controles. Además, proporcionan retroalimentación háptica a través de vibraciones

- *Hand-Worn* Son dispositivos que se adaptan a la anatomía de la mano, por ejemplo, guantes hápticos, sensores *muscle-tension* (*EMG sensors*) y anillos. Sin embargo, suelen tener un costo muy elevado y su tecnología aún no se adapta a las diversas variaciones que puede adoptar las manos
- *Bare Hands* Este tipo de técnica no incluye elementos en la mano lo cual permite bastante libertad de movimiento e inmersión. Son sensadas vía cámaras u otro tipo de sensores. Sin embargo, esto trae desventajas como la falta de retroalimentación háptica y la fatiga al tener que visualizar siempre las manos para poder sensarlas correctamente. No obstante, actualmente es uno de los retos más presentes en HCI, ya que ofrece un sentido de presencia bastante elevado, mejorando así la experiencia del usuario

- *Non-hand Input Devices* (Dispositivos de entrada sin el uso de las manos)
  - *Head tracking* (Seguimiento de cabeza) El más utilizado y el elemento más básico de interacción ya que permite orientarse y poder visualizar todo el entorno
  - *Eye tracking* (Seguimiento de ojos): Aún es un campo inexplorable, que permite conocer hacia donde el usuario dirige la atención
  - Micrófonos: A través de reconocimiento de voz se pueden detectar comandos que permitan realizar acciones
  - *Full-Body tracking* (Seguimiento de cuerpo completo) Generalmente utilizado en el cine a través de diferentes sensores que pueden realizar el seguimiento del cuerpo

De este modo la presencia de las manos juega un papel muy importante, ya que de manera intuitiva y natural podemos adaptarnos a diferentes tipos de tecnologías en dispositivos de entrada, o bien sin el uso de estos, a través de sensores que permitan capturar e interpretar el movimiento de las manos, es decir, comunicarse a través de gestos.

Según [Jojic *et al.*, 2000] existen diversas maneras en las que se puede comunicar información a través de los gestos. Una de las vías es conocida como información espacial que nos permite adoptar diferentes posturas para manipular objetos, por ejemplo, empujar, apuntar, dibujar, cambiar de tamaño, etc. Este tipo de información, por tanto, permite una interacción directa, la cual se refiere a un contacto sin elementos intermedios entre la mano y el objeto virtual [Jerald, 2015].

Otro tipo de información comunicada a través de gestos es la simbólica, en donde a través de posturas conocidas (el símbolo de la paz) el sistema puede reaccionar dependiendo del significado de cada una. En este caso los símbolos se pueden interpretar como un tipo de interacción indirecta, esto se debe a que se requiere cierta configuración e interpretación para poder realizar la acción al objeto de interés.

Actualmente, la mayoría de los sistemas que involucran el reconocimiento de gestos se enfocan en información simbólica, es decir, interacción indirecta, dado que sólo se necesitan identificar ciertas posturas estáticas. Este tipo de interacción disminuye la experiencia del usuario e inmersión, sobre todo en situaciones en donde se requiere manipulación sin necesidad de elementos externos, es decir, interacción natural.

2.3.1. Interacción Natural

Así como la influencia de los recursos informáticos actuales en la definición de "Realidad Virtual", el objetivo de querer algo "natural", ha ido cambiando a medida que la tecnología ha evolucionado con el tiempo. Para HCI, la interacción natural, se encuentra dentro de un continuo que evalúa la fidelidad de la interacción (FI), esto es, el grado de similitud de las acciones realizadas en una tarea virtual con relación a las acciones necesarias para la tarea equivalente en el mundo real [Bowman *et al.*, 2012b].

En efecto, como máximo exponente del espectro de FI, se encuentra la interacción realista o natural [Jerald, 2015], en donde el usuario interactúa con el entorno virtual de la manera más cercana posible a lo real. Así, uno de los medios más utilizados para lograr este tipo de interacción es a través de las manos, ya que es una de las vías que naturalmente solemos usar

para realizar diferentes acciones diarias, y por tanto puede ser interpretadas en el mundo virtual como un medio en donde podríamos agarrar diferentes objetos y manipularlos como si fueran elementos del mundo real.

Interactuar de manera natural permite al usuario adaptarse fácilmente al entorno virtual ya que no requiere conocimiento previo en la interacción. Además, es de gran importancia en aplicaciones en donde se requiera entrenamiento de modo que se puedan transferir las acciones físicas del mundo real al entorno virtual y adquirir así la experiencia sin dañar elementos físicos.

Como se mencionó anteriormente, la naturalidad depende de los recursos informáticos actuales, por lo que para identificar el grado de fidelidad en los tipos de interacción existentes [McMahan *et al.*, 2015] propone un análisis en donde divide FI en 3 conceptos principales:

- Simetría biomecánica: representa el grado de similitud en el que los movimientos del cuerpo (sobre todo las manos) al interactuar virtualmente correspondan a los movimientos físicos en el mundo real.
- Veracidad en la entrada: en donde se mide la exactitud, precisión y latencia en las acciones del usuario en el mundo virtual.
- Control de la simetría: es el nivel de control que puede tener un usuario en la interacción con los elementos virtuales equivalente a la realidad.

2.3.2. Trabajos relacionados: Interacción Natural

En los sistemas VR actuales que involucran el reconocimiento de gestos solo toman en cuenta el aprendizaje de diferentes posturas para poder comunicarse con el entorno [Clark y Moodley, 2016]. La interacción natural a través de gestos que permitan agarrar objetos sin volumen real no ha sido abarcada de manera oficial en el ámbito de Realidad Virtual.

Trabajos relacionados al uso de *bare hands* con gestos naturales corresponden a ambientes de Realidad Mixta, como por ejemplo [Kang *et al.*, 2020b], que realizan una comparativa de tres tipos de interacción, en donde los usuarios prefieren el uso de gestos naturales comparado con los gestos aprendidos. Sin embargo, como parte de la discusión, mencionan que la falta

de retroalimentación háptica podría mejorar la experiencia en la manipulación teniendo un comportamiento más natural con los objetos virtuales.

En efecto, el término de gestos naturales en VR actualmente se encuentra asociado con la disposición de guantes hápticos, sensores inerciales, entre otros [Perret y Vander Poorten, 2018; LI *et al.*, 2019]. Esto es debido a que la presencia del tacto permite al sistema de reconocimiento ahorrarse el trabajo de interpretar los tipos de gestos que deberán estar involucrados en los momentos de interacción, así como proporcionar retroalimentación háptica. Sin embargo, además de ser equipos muy costosos, resultan de gran peso y tamaño de modo que imposibilitan una interacción más flexible.

Una de las soluciones a este problema es a través de sistemas hápticos pasivos, que son técnicas a bajo costo que permiten reemplazar el funcionamiento de guantes hápticos a través del mapeado de elementos físicos reales a los elementos virtuales.

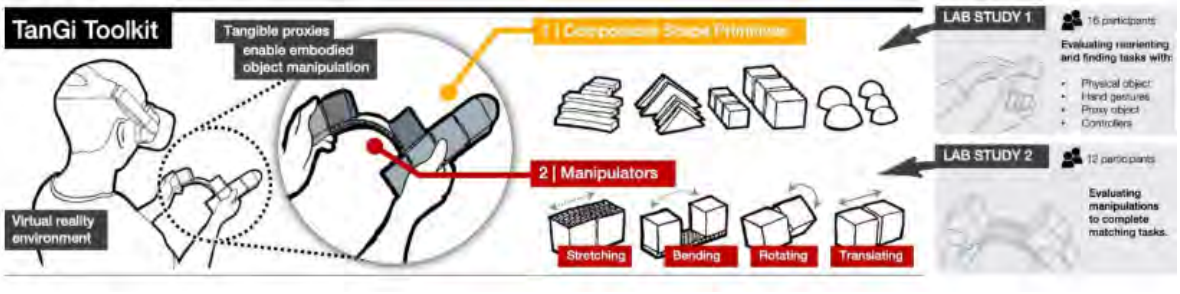

Figura 2-25: Figuras primitivas que permiten adaptarse a diferentes formas virtuales y realizar operaciones físicas de interacción directa. Imagen recuperada de [Feick *et al.*, 2020].

Estos trabajos se dividen principalmente en el uso de dispositivos físicos dinámicos que permiten adaptarse a diversas formas y así poder usarlo con diferentes objetos virtuales (Figura 2-25) [Feick *et al.*, 2020]; así como, trabajos en los que existe una representación muy similar al elemento virtual, por ejemplo, en [Muender *et al.*, 2019], el grado de similitud entre los elementos reales-virtuales no requiere de un equivalente completamente exacto, ya que al combinar el elemento virtual y el tacto superficial, crean una sensación bastante inmersiva; sin embargo, en cuanto a costos, a pesar de ocupar una de las técnicas hápticas más baratas, utilizan más de

19 sensores (OptiTrack y cámaras) para hacer el mapeado lo cual resulta contraproducente.

Precisamente, una de las desventajas de estos trabajos de háptica pasiva, es que requieren de muchos sensores externos para realizar el mapeado real-virtual de los objetos, por lo que se propone entonces una solución a bajo costo que permita evaluar gestos naturales (*bare hands*) a través de una comparación con gestos aprendidos. Además, se desarrollará un sistema que permitirá evaluar la influencia de la respuesta tactil en la interacción con gestos naturales por medio de háptica pasiva.

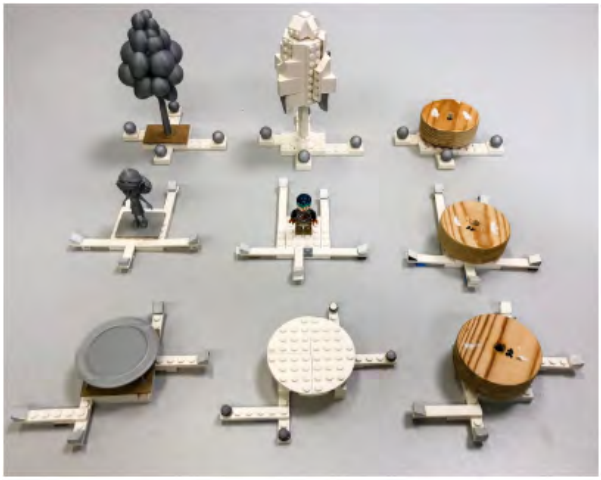

Figura 2-26: Representaciones físicas para retroalimentación háptica, en donde se concluye que a través de legos se logra una fidelidad de la interacción bastante elevada. Imagen extraída de [Muender *et al.*, 2019].

Capítulo 3

Metodología

3.1. Contexto de la investigación

Ya que la Realidad Virtual ha tenido mayor aceptación en el mercado debido a los videojuegos y su reducción en costos comparados con equipos de Realidad Mixta (Hololens ©) el público alrededor de esta tecnología actualmente ha estado presente en jóvenes cuyas habilidades están arraigadas en el uso de consolas de videojuegos [Xue *et al.*, 2020]. Dado que se busca mayor aceptación en el uso de estas tecnologías, se pretende aplicar el experimento en usuarios de diversas edades (entre 25 - 65) que no tengan experiencia en el uso de aplicaciones VR para que los resultados no presenten un sesgo respecto a un tipo de interacción en específico.

En cuanto al tipo de investigación, se pretende realizar una investigación de campo, a través de un análisis comparativo entre el gesto de pinza y los gestos naturales, ya sea sin tacto o con tacto, en donde se podrá analizar cuáles gestos son más apropiados cuando se requiere interactuar directamente con objetos virtuales.

3.2. Variables de estudio

Para medir la experiencia en el uso de las herramientas, se tomarán en cuenta las siguientes métricas que han sido definidas en estudios para evaluar la satisfacción y usabilidad en ambientes virtuales [Rebelo *et al.*, 2012; Tcha-Tokey *et al.*, 2016]

- **Usabilidad**: Definida como la facilidad del aprendizaje de la herramienta y el uso.
  - Se mide la eficiencia, la efectividad y la satisfacción del usuario con la herramienta
  - Se puede medir a través de *System Usability Scale* (*SUS*) [Brooke *et al.*, 1996]
- **Adaptación**: En donde se determina la capacidad de adaptación del usuario con la tecnología
  - Se miden parámetros como contexto social, actitud frente a la tecnología, ansiedad, comportamiento, entre otros.
  - Se puede medir a través de *Unified Technology Acceptance and User of Technology* (*UTAUT*) [Venkatesh *et al.*, 2003]
- **Nivel Emocional**: Definido como los sentimientos que pueden percibir en el uso de la interacción
  - Se miden parámetros como disfrute, orgullo, estrés, ansiedad, frustración, aburrimiento.
  - Se puede medir a través de *Achievement Emotions Questionnaire* (*AEQ*) [Pekrun *et al.*, 2011]
- **Tiempos y errores de interacción**: En donde se determina la eficiencia y la rapidez de la interacción [Rebelo *et al.*, 2012]. Se medirán parámetros cuantitativos como:
  - Tiempo en completar una tarea
  - Número de errores al completar una tarea
  - Tiempo para reparar cada error
  - Tiempo para aprender una instrucción

A través de estas variables, se diseñará un instrumento de evaluación tipo encuesta (Apéndice C) y se grabaran las actividades realizadas por los usuarios para posteriormente medir los tiempos como medida cuantitativa.

3.3. Desarrollo del sistema

Uno de los objetivos al momento de diseñar el sistema a nivel software ha sido el factor de escalamiento hacia algo cada vez más robusto y amplio en variedad de elementos involucrados (integrar más objetos reales, generar un escenario más amplio, crear objetos virtuales encima de los reales, etc.), por lo que se analizaron y diseñaron módulos de manera independiente cuya integración componen el sistema general de interacción natural.

Se evaluarán tres escenarios posibles de interacción, en donde el escenario más complejo abordará retroalimentación háptica pasiva (escenario 3). Esto se debe a que es una de las soluciones a bajo costo más utilizadas para solucionar el problema de respuesta táctil. Para ello, se sensarán objetos del mundo real a través de un único sensor, ya que se desea tener un sistema económico y flexible sin la necesidad de utilizar demasiados equipos externos. La manera en la que se sensará, será a través de un área delimitada por marcadores como se muestra en la figura 3-1, ya que a través de la detección de los mismos se podrá posicionar la cámara desde cualquier posición y a través de corrección de perspectiva, se podrá obtener la posición espacial de los objetos reales para poder mapearlos en el entorno virtual.

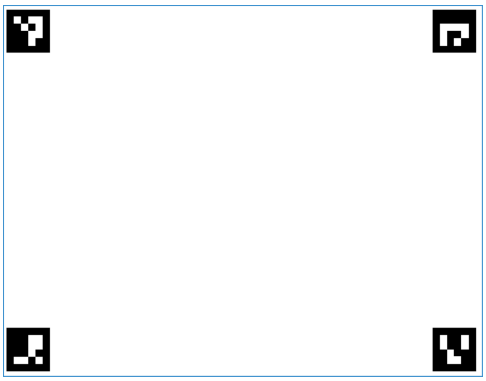

Figura 3-1: Área delimitada por marcadores donde se reconocerán los objetos reales

Los otros dos escenarios, se enfocan en gestos aprendidos (escenario 1) y gestos naturales (escenario 2), esto es, a través del seguimiento de mano se interpretarán las posturas y dependiendo del escenario, será la respuesta del sistema. Se pretende entonces, enfocar y explicar el

diseño del escenario más complejo (háptica pasiva) ya que compone todos los módulos necesarios e incluidos en los dos escenarios restantes.

3.3.1. Estructura general del sistema

El sistema se dividirá en 3 etapas principales. Las primeras dos etapas son exclusivas del escenario 3, mientras que la última etapa involucra todos los escenarios. Así entonces, la primera se encarga de obtener los variables necesarias para que el sistema permita mapear correctamente los objetos reales al mundo virtual. Después de tener dichos valores se procede a la segunda etapa que permite identificar los objetos que se encuentran en el escenario y generarlos en el mundo virtual. Finalmente, la última etapa se enfoca en la lógica de la interacción y el comportamiento del objeto al existir colisión mano-objeto (Figura 3-2)

Figura 3-2: Funcionamiento general del sistema de interacción

3.3.2. Etapa de Configuración

Por medio de esta fase, obtendremos matrices y valores que se utilizarán posteriormente para los cálculos de las posiciones y orientaciones de los objetos (Figura 3-3). Dado que en esta etapa intervienen tanto el sistema de visión como el virtual, se dividirá en dos partes la sección.

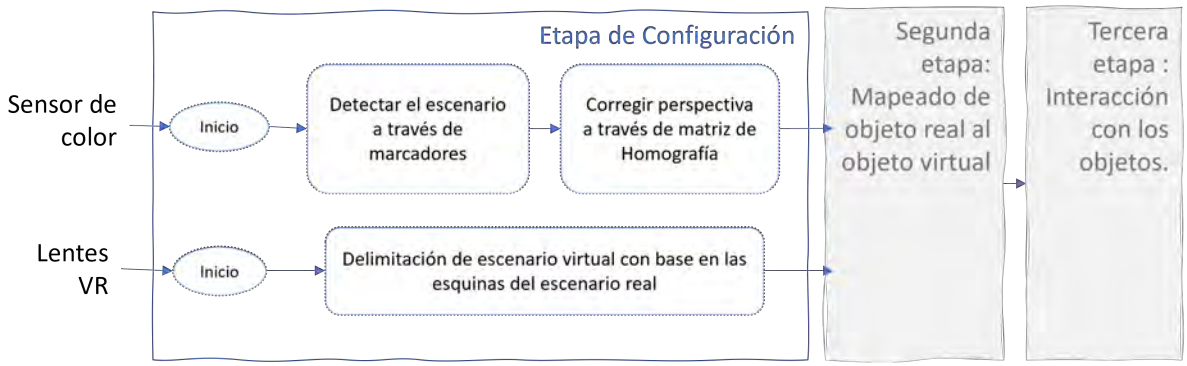

Figura 3-3: Módulos de la etapa de configuración. Sistema asíncrono en donde se configurarán ambas partes (Sistema de visión y virtual) para obtener valores que permitirán el mapeado en la siguiente etapa

3.3.2.1. Sistema de Visión

El objetivo de este módulo es obtener la matriz de homografía que permita corregir la perspectiva del escenario y poder obtener correctamente la posición de los objetos para el siguiente módulo.

Dado que se detectará el escenario a través de marcadores, es importante mantener un orden específico de modo que se pueda determinar el punto origen del escenario. Así, sin importar la ubicación de la cámara (y la orientación), siempre se podrá obtener la imagen desde la perspectiva de la vista de la persona. Es decir, si la cámara se encuentra en frente de nosotros detectando el escenario, la imagen original de la escena se encuentra rotada unos 180 grados (Figura 3-4). Por lo que a través de la corrección de la perspectiva se transformará la imagen en el sentido correcto de la vista de la persona debido al orden correcto de los marcadores.

Otro aspecto importante es que se desea un escenario que se adapte a diferentes tamaños, y así ampliar o reducir el área de interacción, por lo que se desconoce al ancho y alto del mismo. Para poder corregir la perspectiva, como se menciona en la sección 2.1.2.5, es necesario conocer mínimo 4 pares de puntos. Los primeros 4 puntos corresponden a las esquinas del escenario desde la vista real de la cámara. Esto se puede obtener a través de la información que

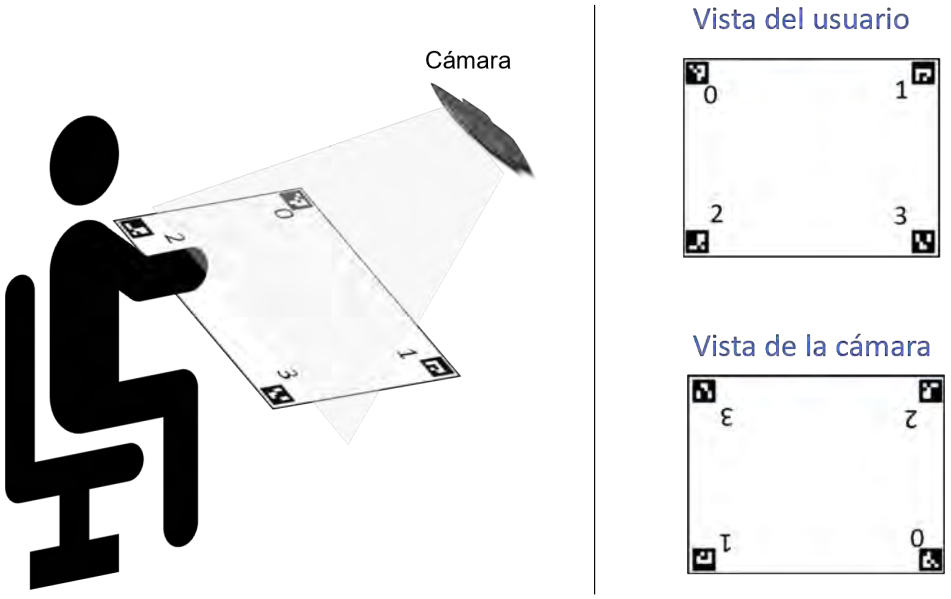

Figura 3-4: Importancia de mantener un orden específico en los marcadores. La vista del escenario a través de la cámara se encuentra rotada 180^o por lo que a través del orden de los vértices se podrá rotar y corregir la perspectiva en torno a la vista del usuario

proporcionan los marcadores (cada marcador tiene 4 vértices, por lo que el vértice más lejano del centro del escenario será una de las esquinas del escenario). Los 4 puntos restantes, siendo estos los puntos destino, deberán formar un rectángulo lo cual nos permitirá corregir líneas a partir de la imagen en proyectiva de la cámara. Sin embargo, una vez que se obtiene la matriz homográfica, y obtenemos una imagen con líneas paralelas, dado que no se conoce la relación de aspecto (ratio) del escenario, se deberá realizar un ajuste en alguna de sus dimensiones (alto o ancho), a través de algún valor conocido, por lo que por medio de las medidas reales de los marcadores, y su relación de aspecto luego de ser transformada la imagen, se puede ajustar la matriz homográfica de modo que el ratio del escenario sea bastante similar al real (Figura 3-5) y así, poder obtener correctamente las ubicaciones de los objetos.

En resumen, a través del siguiente diagrama se podrá obtener la matriz homográfica la cual transformará la imagen de entrada (cada frame del video) en una imagen de salida cuyas líneas serán paralelas y cuyos límites serán los mismos definidos por los marcadores del escenario real (Figura 3-6).

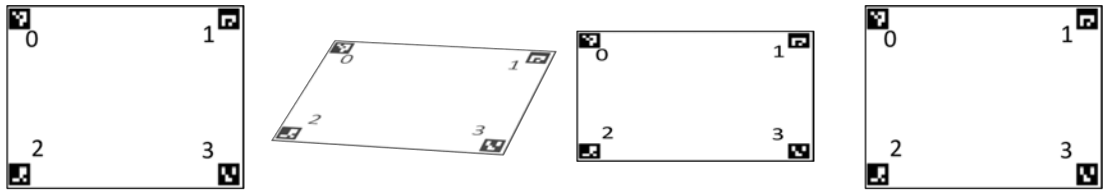

Figura 3-5: Correción de perspectiva con base en la matriz homográfica y la relación de aspecto de los marcadores

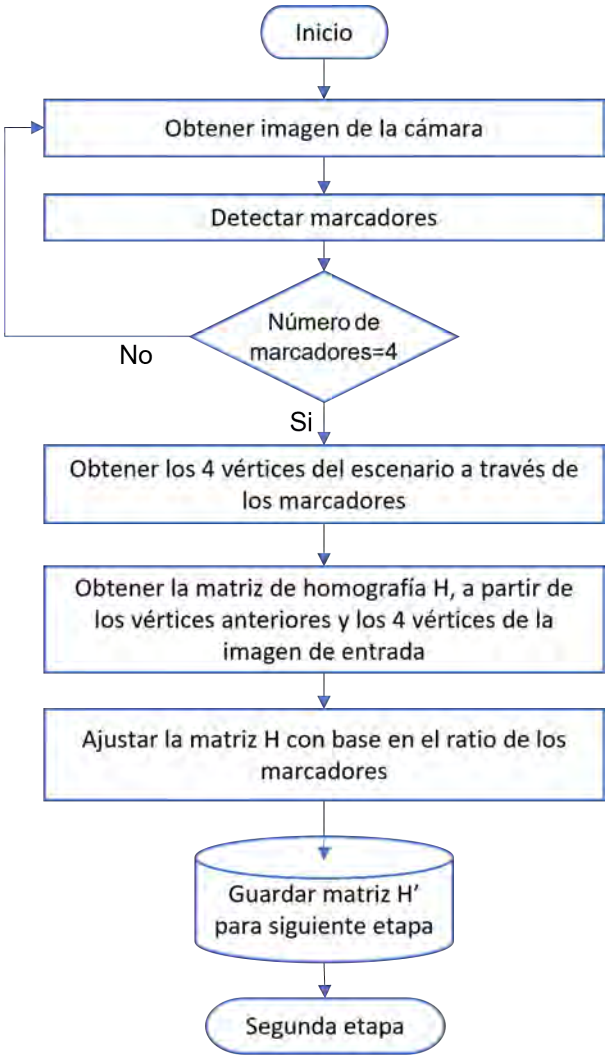

Figura 3-6: Diagrama de flujo para obtención de la matriz de homografía

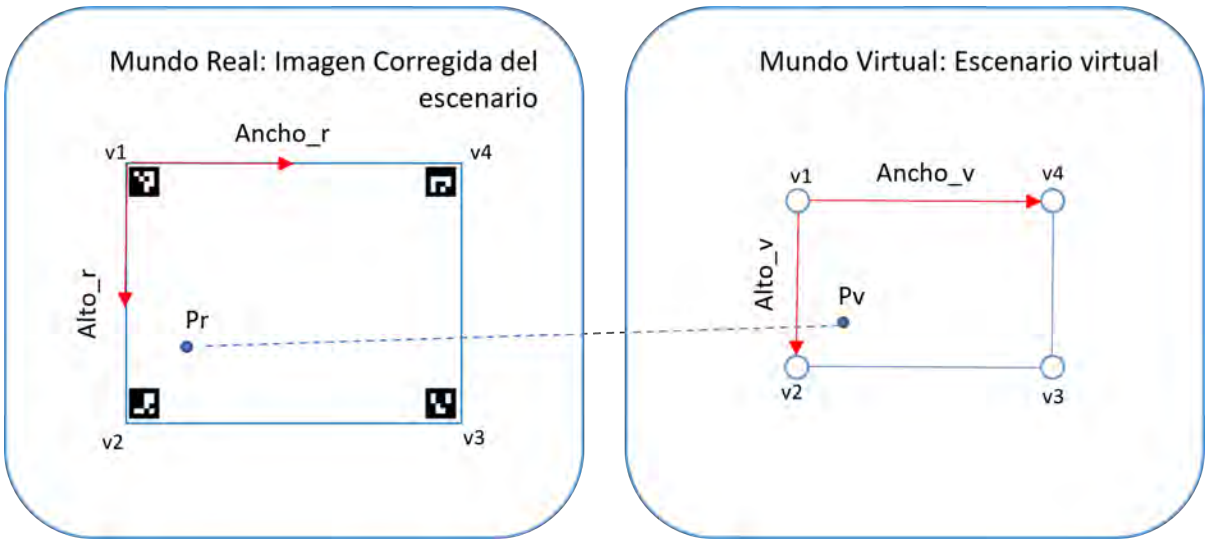

Figura 3-7: Ejes coordenados alineados en ambas realidades. El punto Pr del escenario real corresponderá al punto Pv del escenario virtual

3.3.2.2. Sistema virtual

Debido a que no se puede obtener información de las cámaras en los lentes de Realidad Virtual por cuestiones de privacidad con el usuario, la forma de obtener información del mundo real puede ser a través de los controles o de las manos.

Ya que se pretende utilizar las manos sin el uso de controles externos, los lentes de VR deberán tener soporte en la detección y seguimiento de las mismas. Se propone al usuario señalar las esquinas del escenario real que se registrarán como coordenadas en el mundo virtual y así poder obtener el modelo del escenario en el entorno virtual. Esto con el objetivo de servir como cuerpo físico para ubicar los objetos virtuales en el entorno virtual y que correspondan a la ubicación de los mismos en el entorno real. Los ejes del sistema de coordenadas local del escenario virtual corresponderán a los ejes de la imagen para facilitar los cálculos (Figura 3-7).

Por medio de los vértices del escenario virtual (obtenidos a través de las posiciones registradas por la mano) se obtendrá un nuevo marco de referencia local que permitirá mapear las ubicaciones de los objetos del plano de la imagen al plano virtual. Es decir, se calculará la posición local del objeto en el escenario virtual con respecto a su ubicación en la imagen (a

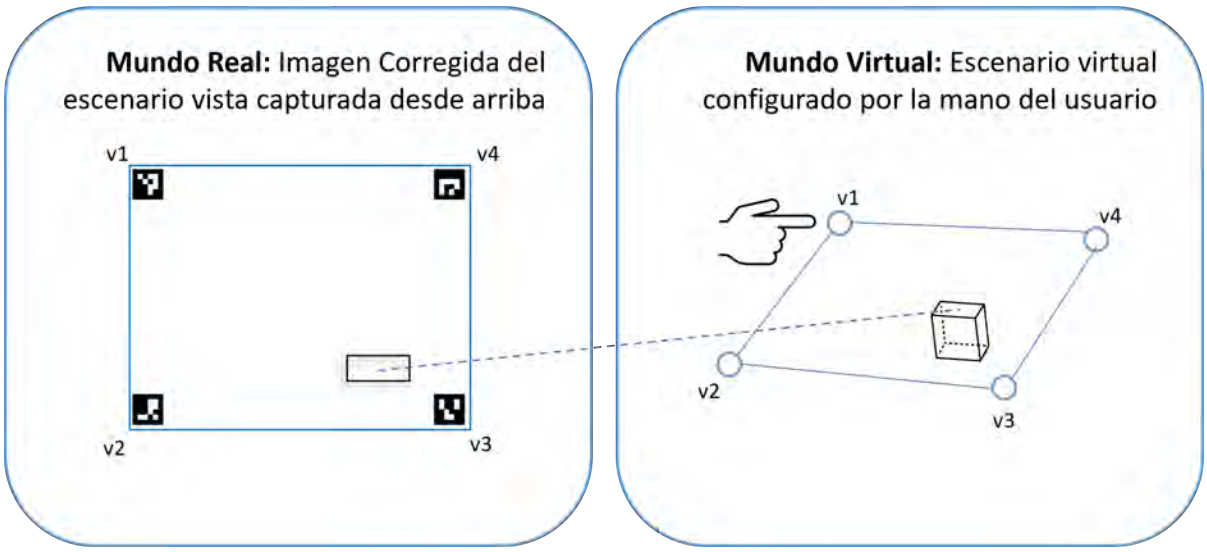

Figura 3-8: Correpondencia de los vértices entre el escenario real y el escenario virtual

través del ancho y alto de ambos escenarios) y a través de la matriz que representa el nuevo marco de referencia del escenario virtual y ese punto local, se aplicará la transformación para obtener las coordenadas globales del objeto en el mundo virtual (Figura 3-8).

Para la orientación del objeto, se obtienen los ángulos de rotación del marco de referencia local del escenario virtual y se suma el ángulo de rotación del eje vertical con respecto al ángulo obtenido en el plano de la imagen, ya que se pretende interactuar siempre con objetos en una mesa.

Por tanto, los valores que se deben calcular y que se utilizarán en los siguientes módulos serán: matriz del sistema de referencia local, altura y anchura del escenario virtual y ángulos de rotación de los 3 ejes de coordenadas como se muestra en el diagrama de flujo de la figura 3-9.

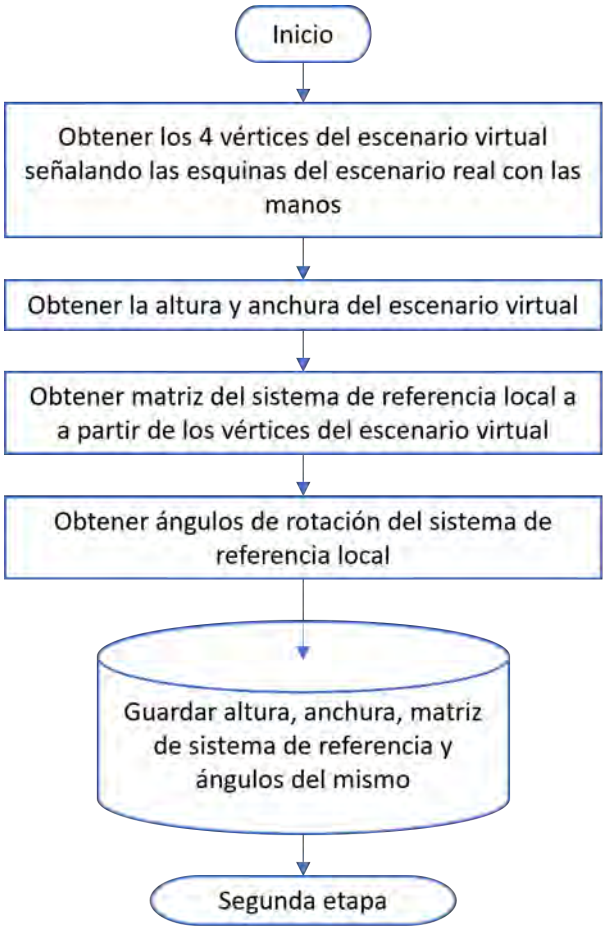

Figura 3-9: Diagrama de flujo de configuración del escenario virtual en el mundo virtual

Obtención de la altura y la anchura Ya que el escenario virtual debe corresponder a lo que sensa la cámara, se seguirá la lógica de ancho y alto con respecto a cómo se interpreta en una imagen. Los vértices virtuales tendrán la correspondencia como se muestra en la figura 3-8.

Aclarado este punto, para obtener la altura y anchura del escenario virtual aplicaremos la distancia euclidiana en R^3. Es decir,

$$d(p_1, p_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{3-1}$$

Por tanto, con respecto a la notación de la figura 3-8, el ancho corresponderá a la distancia

de los vértices v1 y v4; la altura corresponderá a la distancia entre los puntos v1 y v2

Cálculo del sistema de referencia local Para poder mapear las coordenadas de la imagen al plano virtual, es necesario obtener primero la correspondencia de puntos en ambos planos y utilizando la matriz que representa el sistema de coordenadas local, podremos convertir el punto local del escenario virtual a las coordenadas globales del mundo virtual.

Dado que ya tenemos los vértices del escenario virtual (registrados por el usuario en un inicio), podemos obtener los vectores que representan el sistema de ejes. Siguiendo la orientación de las coordenadas del plano de la imagen, los ejes coordenados del sistema local se representarían de la siguiente manera:

- Eje de las abscisas: $\vec{u} = \frac{v4-v1}{|v4-v1|}$
- Eje de los ordenadas: $\vec{v} = \frac{v2-v1}{|v2-v1|}$
- Eje Z: $\vec{w} = \vec{u} \times \vec{v}$

A través del sistema de coordenadas global y el local en un punto que representa el origen del escenario virtual, tendríamos relación que corresponde a la ecuación 3-2.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \\ 1 \end{pmatrix} = \begin{bmatrix} u_1 & v_1 & w_1 & v1_x \\ u_2 & v_2 & w_2 & v1_y \\ u_3 & v_3 & w_3 & v1_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{pmatrix} \tag{3-2}$$

$$\begin{pmatrix} m_1 \\ m_2 \\ m_3 \\ 1 \end{pmatrix} = \begin{bmatrix} u_1 & v_1 & w_1 & v1_x \\ u_2 & v_2 & w_2 & v1_y \\ u_3 & v_3 & w_3 & v1_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ 1 \end{pmatrix} \tag{3-3}$$

$$\mathbf{m} = \begin{bmatrix} M_r & \mathbf{v1} \\ \vec{0}^T & 1 \end{bmatrix} * \mathbf{p} \tag{3-4}$$

$$\mathbf{m} = H * \mathbf{p} \tag{3-5}$$

Donde a partir de un punto **p** en el sistema local del escenario y la matriz de transformación H cuyo origen se encuentra en el vértice **v1** (véase Figura 3-8), podremos obtener el punto **m** que se encuentra en el sistema global del mundo virtual (Ecuación 3-5)

Obtención de los ángulos de rotación del sistema de referencia del escenario Las matrices de transformaciones afines están conformadas por un conjunto de matrices identidades que permiten modificar los puntos de entradas a través de ángulos de rotación de cada eje, escalamiento, reflexión, entre otros.

Las matrices que permiten transformar la orientación de los puntos en el espacio R^3, se definen para cada eje como se muestra en el sistema de matrices 3-6.

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix}, R_y = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix}, R_z = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3-6}$$

A través de la composición de estas matrices, se puede obtener una matriz de transformación cuyos ejes tengan cierta orientación dependiendo del ángulo de cada matriz de rotación, esto es: $R = R_z R_y R_x$

Puesto que se obtuvo la matriz que representa el nuevo sistema de coordenadas del escenario, se puede obtener el proceso inverso dada la matriz de rotación, es decir, descomponer la matriz de modo que obtengamos los ángulos de rotación de cada eje de la siguiente manera. Dada la matriz M_r obtenida en la ecuación 3-4:

$$M_r = \begin{bmatrix} u_1 & v_1 & w_1 \\ u_2 & v_2 & w_2 \\ u_3 & v_3 & w_3 \end{bmatrix} \tag{3-7}$$

Tenemos que los ángulos de cada eje se pueden obtener como se muestran de la ecuación 3-8 a la 3-10.

$$\theta_x = \arctan(\frac{v_3}{w_3}), \text{ cuyo rango es } (-\pi, \pi) \tag{3-8}$$

$$\theta_y = \arctan(\frac{-u_3}{\sqrt{v_3^2 + w_3^2}}), \text{ cuyo rango es } (-\frac{\pi}{2}, \frac{\pi}{2}) \tag{3-9}$$

$$\theta_z = \arctan(\frac{u_2}{u_1}), \text{ cuyo rango es } (-\pi, \pi) \tag{3-10}$$

3.3.3. Segunda etapa: Mapeado del objeto real al entorno virtual

Luego de obtener las variables que permiten corregir la perspectiva del sistema de visión y la matriz que permite orientar cada objeto en el mundo virtual, continuaría la fase de detectar, identificar y mapear el objeto real al mundo virtual.

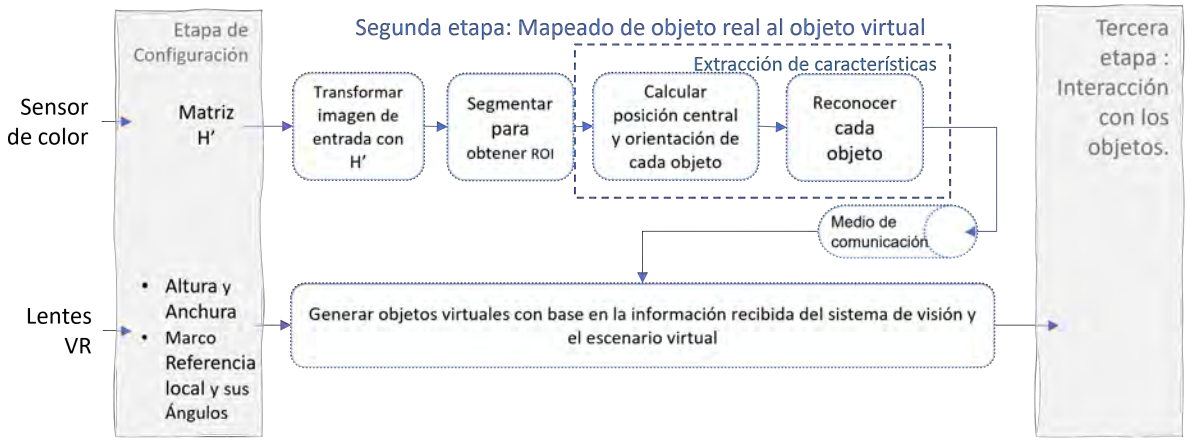

Figura 3-10: Módulos de la segunda etapa. Sistema síncrono en donde a partir de la identificación de los objetos en el escenario real se generarán estos objetos en el ambiente virtual.

Para lograr identificar qué objeto se encuentra en el escenario y su información de posición y orientación, se deberán seguir los pasos que se muestran en la figura 3-10. Es decir, aplicando la

matriz de corrección de perspectiva a la imagen de entrada, se pretende segmentar con base en bordes, obteniendo las regiones de interés (ROI) de los objetos que se encuentren en el escenario real, para así calcular la posición central, orientación, el tipo de objeto, y poder enviar dicha información al sistema virtual para generar y ubicar los modelos virtuales en relación con los elementos reales.

Figura 3-11: Diagrama de flujo del proceso de segmentación.

3.3.3.1. Segmentación por bordes

Dado que se utilizará un escenario controlado, cuya área de detección será delimitada por los marcadores, el trabajo de segmentación ayudará a identificar las regiones de interés que pertenecerán a los objetos dentro del mismo. Una de las técnicas más utilizadas en tiempo real y con entornos controlados, es la segmentación a través de la identificación de bordes, por lo que

se propone, segmentar la imagen transformada por la matriz H' (cuyos límites están definidos por los marcadores) por medio de detección de bordes para poder identificar figuras cerradas (Figura 3-11).

Una de las desventajas de segmentar utilizando esta técnica es el ruido que provocan las sombras de los objetos en la superficie, generando contornos más amplios y deformados, lo cual interfiere en la precisión del cálculo de la posición y orientación de los objetos (Figura 3-12). Es por ello que será importante una etapa previa de procesamiento de la imagen, eliminando en cierto grado el ruido ocasionado por sombras.

El objetivo final, es obtener una máscara de segmentación que indique las regiones en donde se encuentran los objetos, para poder etiquetarlas y posteriormente reconocer cada uno de los elementos.

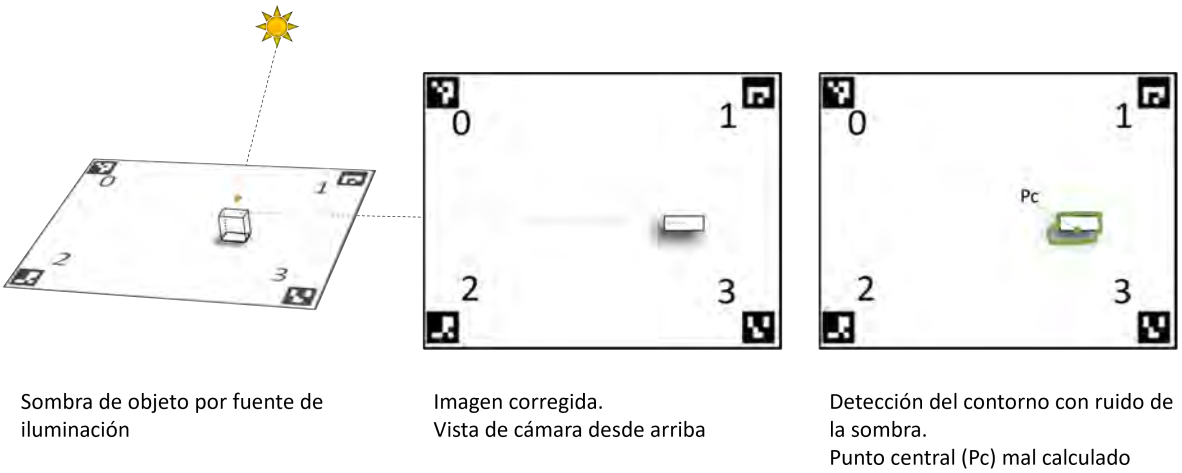

Figura 3-12: El ruido de la sombra interfiere en el cálculo de la posición del objeto y su orientación.

Procesado de la imagen Ya que el objetivo principal de este procesado es reducir lo más posible las sombras en tiempo real, se propone un filtrado a través de un ajuste en el espacio de color HSV. Como se menciona en la sección 2.1.1.1, las imágenes con el modelo HSV, permiten regular los niveles de oscuridad. Si fijáramos el canal *Value* del modelo HSV a un valor promedio con respecto a todos sus valores, podríamos reducir las variaciones de oscuridad en la imagen,

como se muestra en la figura 3-13, por lo que ésta será la técnica utilizada para reducir el ruido ocasionado por las sombras.

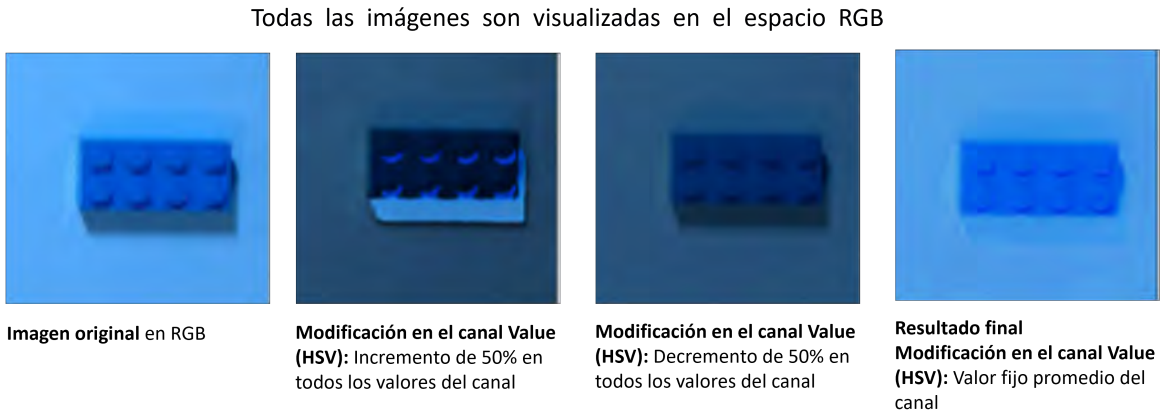

Figura 3-13: Ejemplo de ajuste del canal Value en el espacio de color HSV para reducir sombras

Algoritmo de Canny para detectar bordes Una vez reducido el ruido, a pesar de seguir notando una leve variación o residuo de lo que fue la sombra (Figura 3-13), este tipo de técnica de detección de bordes permite filtrar ciertas variaciones de intensidad y por tanto evitar bordes suaves [Canny, 1986].

Al aplicar este algoritmo, el resultado será una imagen binaria cuyos valores verdaderos representarán a todas las curvas detectadas y los falsos corresponderán al fondo. A partir de estos valores, podemos obtener figuras cerradas que corresponderán a la ROI de los objetos. Se entiende por figura cerrada a una continuidad de puntos, cuyo punto inicial y final es el mismo.

Filtros morfológicos para reducción de ruido y cerramiento Luego de obtener los bordes, puede ocurrir que ciertas figuras queden abiertas por el procesado en el espacio de color y la eliminación de líneas gruesas por canny, lo cual no permitiría identificar uno de los objetos. Es por ello que se aplicará un filtro morfológico de dilatación (sección 2.1.2.4) que permitirá cerrar aquellas figuras que por unos pocos pixeles quedaron abiertas, como se muestra en el ejemplo de la figura 3-14.

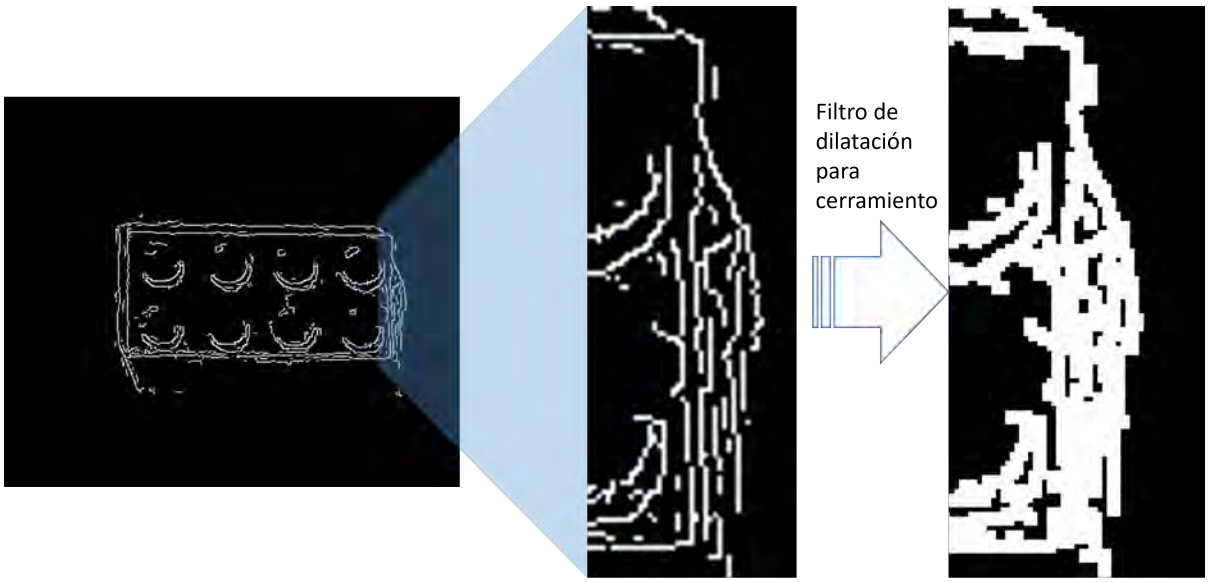

Figura 3-14: Cerramiento de figura a través de filtro de dilatación. Ejemplo obtenido de la figura 3-13

Una vez realizado este filtrado, se pretende rellenar las figuras cerradas (a través de un filtro morfológico basado en dilatación interna) para poder distinguir figuras, de líneas excedentes en torno a las mismas. Terminado este paso, se aplicará una combinación de filtros de erosión seguido de dilatación que nos permitirá reducir el ruido y eliminar específicamente las líneas detectadas por el residuo de la sombra (Figura 3-15). Para alisar los bordes de estas regiones, obteniendo figuras convexas (técnica conocida como *Convex Hull*) se usará el algoritmo *gift wrapping* [Jarvis, 1973]. El objetivo de este es, a partir del punto que se encuentre más hacia la izquierda encontrar los puntos subsecuentes cuyo ángulo interior entre el actual y el siguiente, sea el más grande. Este procedimiento se realiza hasta volver a coincidir con el punto inicial, teniendo como resultado final el ejemplo que se muestra en la misma figura 3-15.

Finalmente, dado que pueden existir figuras cerradas que no pertenezcan a los objetos de interés, se procederá a filtrar a través de una umbralización con respecto al área de cada una de las figuras cerradas.

Con estos pasos, obtendremos la máscara de segmentación, por lo que el siguiente módulo sería el etiquetado y extracción de características con base en la posición central de cada figura, orientación y su reconocimiento

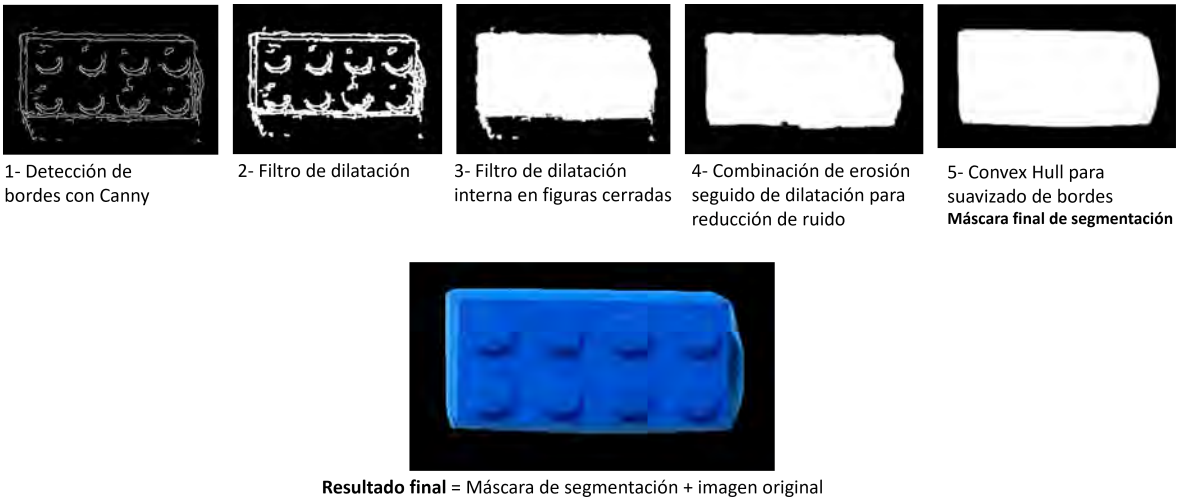

Figura 3-15: Pasos para obtener la máscara de segmentación. Ejemplo obtenido de la figura 3-13

3.3.3.2. Extracción de características

Una vez obtenida la máscara de segmentación cuya ventaja es ubicar las regiones de interés, seguirían los pasos de extracción de características tanto para el sistema de reconocimiento de objetos como el sistema VR (posición, orientación e identificación de objeto), como se muestra en el diagrama de flujo de la figura 3-16.

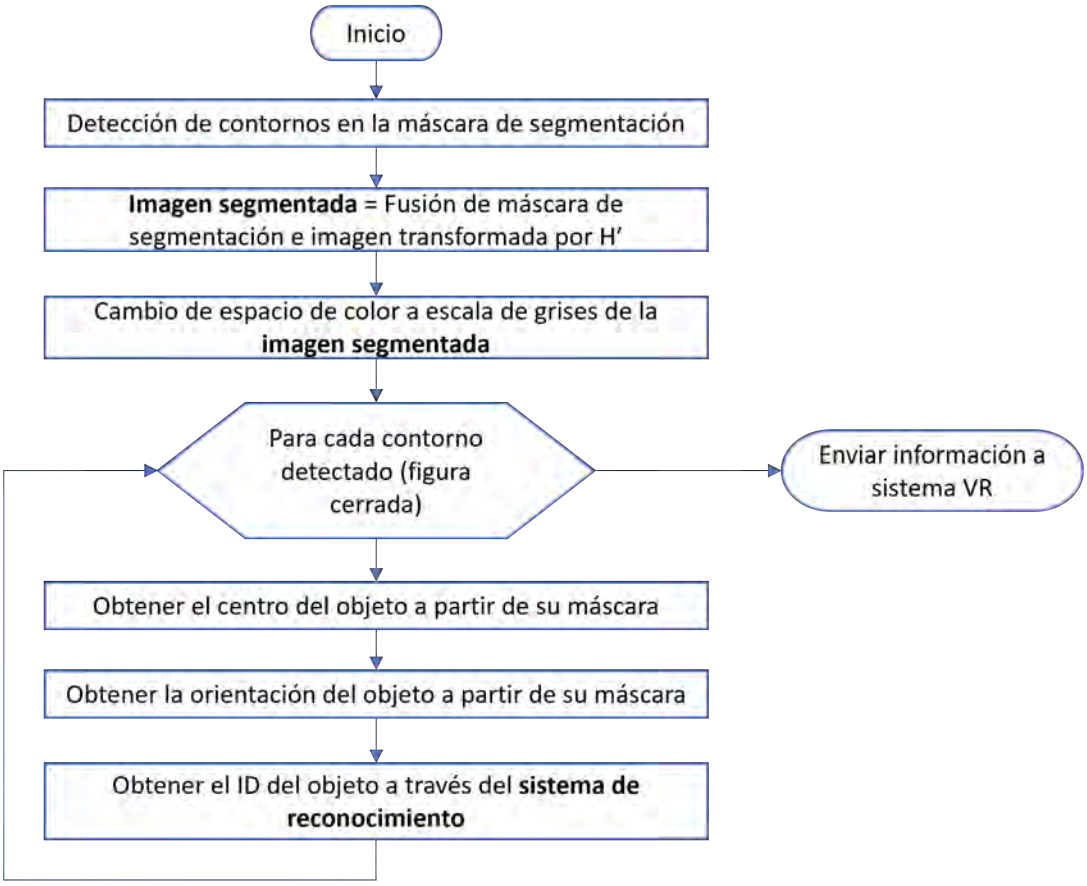

Figura 3-16: Diagrama de flujo para extraer características y enviar información a sistema virtual

Detección de contornos en la máscara de segmentación Una vez obtenida la máscara de segmentación, se procede a etiquetar estas regiones por medio de los contornos, es decir, obtener aquellos bordes cuya extensión forme figuras cerradas.

Se utilizará un algoritmo de aproximación de contornos desarrollado por Suzuki (Suzuki *et al.* [1985]), dado que puede jerarquizar los bordes dependiendo de la ubicación de los mismos. En este caso el procedimiento será muy rápido debido a que solo se tienen figuras cerradas en la máscara de segmentación y este proceso sólo será utilizado para etiquetado.

Obtención de la posición central y la orientación A través del cálculo de la posición, su orientación y el tipo de objeto, se podrá generar el objeto real en el mundo virtual. Los valores de posición y orientación se pueden calcular a través de la distribución de puntos del contorno y el área de la figura cerrada

Centroide La ubicación del objeto estará dada a través de su centroide, que se calculará con el promedio de los pixeles dentro de cada región.

Orientación sencilla en objetos simples Para obtener una orientación sencilla necesitamos una línea principal que se ajuste a la distribución de los puntos del contorno de cada región y que a través de ella se pueda obtener el ángulo de rotación con respecto al sistema de coordenadas de la imagen. Para ello podemos calcular esta línea a través de la técnica de mínimos cuadrados promedio (LMS) como una solución para encontrar la recta que dado a un conjunto de datos (en este caso los puntos del contorno) la suma de todas las distancias de los puntos con respecto a la recta sea la menor posible (figura 3-17).

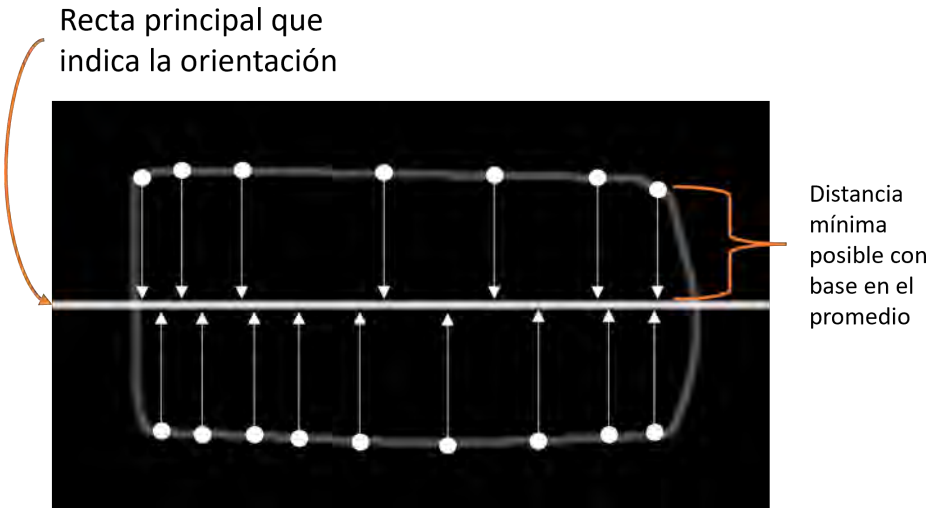

Figura 3-17: Cálculo de recta principal utilizando LMS para obtener la orientación del objeto. Ejemplo obtenido de la máscara de segmentación de 3-15

Para obtener el ángulo de rotación a partir del vector director $[vx, vy]$ de la recta, tendríamos

que: $\theta = \arctan(vy/vx)$, donde precisamente este será el ángulo que se sumará al ángulo vertical del objeto en el mundo virtual.

El último paso dentro de este proceso de obtención de información es el reconocimiento del objeto, de modo que indique que tipo de objeto generará en el entorno virtual.

3.3.3.3. Reconocimiento del tipo de objeto

Dado que se pretende escalar a un sistema en donde se puedan reconocer diferentes objetos y mapear su estructura al mundo virtual, se implementó un modelo de red neuronal convolucional. De este modo, la extracción de características dependerá de los tipos de objetos que se desean mapear, dado que este tipo de red la incluye como etapa inicial. Tanto para el conjunto de entrenamiento como para la predicción, las imágenes de entrada deberán cumplir con las mismas dimensiones, dado que el número de neuronas de entrada depende de un número de pixeles fijos.

Ajuste de la imagen de entrada a la ConvNet Una característica importante a tomar en cuenta es el espacio de color con el que entrarán las imagenes a la red. Se propone utilizar escala de grises, de modo que objetos similares sean considerados iguales sin importar el color de los mismos. Esto se debe a que una de las características más importantes deberá ser la forma y no su color. El objetivo de mapear el objeto real a un mundo virtual es recibir retroalimentación háptica por lo que el color del objeto no se considera un aspecto principal. Esto a su vez permitirá reducir la cantidad de neuronas involucradas en la red, es decir, dada una imagen a color de dimensiones 100x100x3 sería un total de 30000 neuronas en la capa de entrada, comparado a una imagen en escala de grises de 100x100x1=10000 neuronas.

Otra característica importante para tomar en cuenta es la dimensión en anchura y altura, por lo que se propone una imagen cuadrada (ancho = alto). Para evitar deformar la forma de los objetos al realizar el ajuste de dimensionalidad y dado que la imagen de entrada es el objeto segmentado, se propone extender altura y/o anchura de valores nulos (color negro) como se muestra en el ejemplo de la figura 3-18.

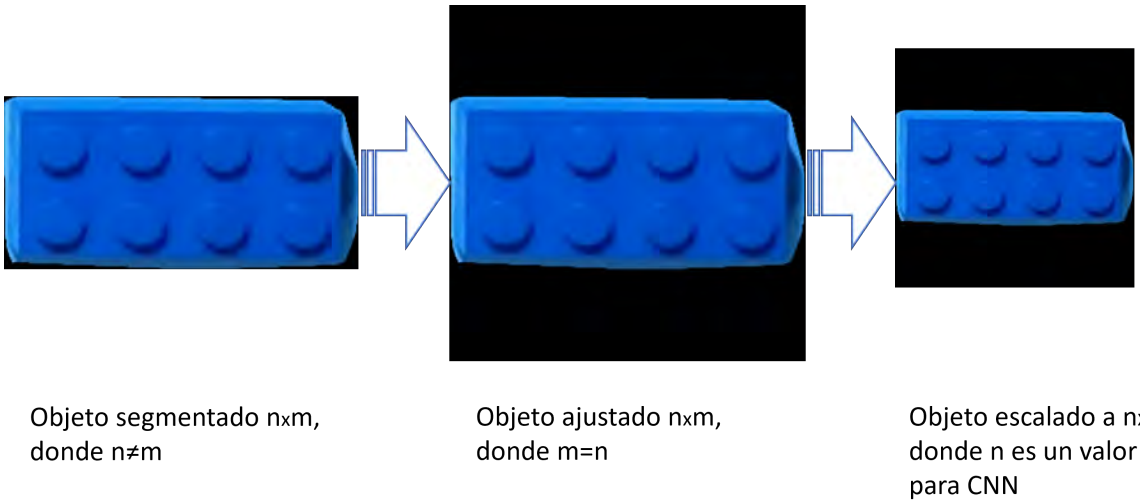

Figura 3-18: Ajuste del objeto segmentado para la entrada a la CNN. Ejemplo obtenido de la figura 3-13

Entrenamiento y aumento de datos Una de las desventajas de utilizar CNN es el número elevado de muestras necesarias para un correcto entrenamiento. Es por ello que se suelen utilizar técnicas de aumento de datos para solventar este problema.

Así, se propone extraer por medio de un video diferentes perspectivas de los objetos a entrenar, así como utilizar transformaciones geométricas y en el espacio de color, de modo que la red pueda reconocer tanto en variaciones de intensidad, como en deformaciones de las formas de los objetos, debido principalmente al resultado dado por la corrección de perspectiva.

3.3.3.4. Envío de la información

Dado que, el sistema de visión y virtual, son ambientes diferentes de desarrollo, se necesita de una vía de comunicación para enviar la información de los objetos.

Para ello se propone un sistema cliente-servidor. Siendo el sistema de visión el servidor que permitirá mandar información al cliente (el sistema de visión), respecto al tipo de objeto, posición en la imagen y su vector director.

3.3.3.5. Mapeado del objeto real al sistema virtual

Para la generación del objeto virtual se deben considerar los siguientes puntos:

- Los modelos virtuales de los objetos a mapear serán modelados con anterioridad, y deberán corresponder a medidas reales de nuestro mundo
- El centro de masa de los objetos virtuales debe encontrarse en el centro de la base del mismo, para facilitar los cálculos en la correspondencia entre el centroide del objeto en la imagen y el objeto virtual.

Una vez recibida la información del tipo de objeto, su ubicación en la imagen y el vector director, se generará el objeto y se realizarán los siguientes cálculos para ubicarlo en el entorno virtual.

Ubicación del objeto virtual Partiendo del sistema de coordenadas local cuyo origen corresponde al v1 como se muestra en la figura 3-7, y con base en la altura y la anchura del escenario virtual (sección 3.3.2.2), podemos calcular el punto que representará la ubicación del objeto en el plano virtual a partir de la ubicación de ese punto en el plano de la imagen.

Dado un punto $Pr = (x_r, y_r)$ en la imagen, las coordenadas en el plano del escenario virtual se obtendrían a través de la siguiente relación:

$$\frac{Pr - Pr_{min}}{Pr_{max} - Pr_{min}} = \frac{Pv - Pv_{min}}{Pv_{max} - Pv_{min}} \tag{3-11}$$

$$Pv = \frac{Pr - Pr_{min}}{Pr_{max} - Pr_{min}} * (Pv_{max} - Pv_{min}) + Pv_{min} \tag{3-12}$$

Sabiendo que Pr_{min} corresponde al punto $(0, 0)$ de la imagen, Pr_{max} corresponderían al tamaño de la imagen (Ancho_r, Alto_r), y de igual manera, en el escenario virtual Pv_{min} sería el punto $(0, 0)$ en el sistema de coordenadas local del plano y el punto Pr_{max} sería (Ancho_v, Alto_v), la ecuación para encontrar el punto en el mundo virtual sustituyendo los valores conocidos en la ecuación 3-12 sería como muestra en la ecuación 3-13.

$$Pv = \frac{Pr}{Pr_{max}} * Pv_{max} \tag{3-13}$$

Una vez obtenido el punto en el plano virtual, se procedería a transportarlo al sistema de coordenadas global, a través de la matriz de transformación H (ecuación 3-5) obtenida en la etapa inicial (sección 3.3.2.2). Es decir, el punto $Pv = (Pv_x, Pv_y, 0, 1)$ del sistema de coordenadas local del escenario virtual se convertiría en el punto Pm correspondiendo al sistema de coordenadas global del mundo virtual (ecuación 3-15)

$$\begin{pmatrix} Pm_x \\ Pm_y \\ Pm_z \\ 1 \end{pmatrix} = \begin{bmatrix} u_1 & v_1 & w_1 & v1_x \\ u_2 & v_2 & w_2 & v1_y \\ u_3 & v_3 & w_3 & v1_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} Pv_x \\ Pv_y \\ 0 \\ 1 \end{pmatrix} \tag{3-14}$$

$$Pm = H * Pv \tag{3-15}$$

Orientación del objeto virtual Dado que, en la etapa inicial, a través de la matriz del marco de referencia local se obtuvieron los ángulos de rotación del escenario virtual (sección 3.3.2.2). A cada uno de los objetos como fase inicial se deberá ajustar los ángulos de su sistema de referencias global. Una vez orientado con respecto al escenario virtual, se obtendrá el ángulo de rotación dado por el vector director recibido por parte del sistema de visión calculado en la sección 3.3.3.2, y se sumará al ángulo θ_z del sistema de referencia del objeto virtual.

3.3.4. Tercera etapa: Interacción con los objetos virtuales

Una vez finalizado el mapeado entre el mundo real y virtual, seguiría la última etapa cuyo objetivo es establecer las físicas para manipular de una manera directa y natural los objetos. Por tanto, se propone generar eventos de colisiones en los dedos que permitirán avisarnos cuando existe una colisión con los objetos involucrados. Dependiendo de los dedos que estén en contacto con los objetos podremos agarrarlos y moverlos alrededor del mundo virtual, lo cual simularía

la interacción directa que tenemos cuando agarramos un objeto en el mundo real.

Los tipos de interacción con el objeto virtual (OV) que se proponen dependiendo del escenario son los siguientes:

- **Tocar directamente.** En todos los escenarios de interacción se permite esta acción, la cual se interpreta como una colisión entre el modelo de la mano virtual y el OV, y en donde este último, realizará una reacción física simulada dada la acción de la mano
- **Agarrar.** En este caso se tienen dos opciones: a través de gestos aprendidos (se propone gesto de pinza) o por medio de interacción natural (libertad de agarre por parte del usuario). Existirá por tanto una colisión que bloque una reacción de salida del OV y que su ubicación dependa de la región abarcada por los dedos.
- **Mover.** El movimiento permite al usuario cambiar de posición a los OVs en donde sólo será posible al tocar o agarrar el objeto
- **Soltar.** Éste última acción será posible al liberar o dejar de colisionar los dedos de la mano con el OV

Capítulo 4

Implementación

Una vez realizado el estudio respecto al diseño de cada uno de los módulos y justificado cada una de las técnicas utilizadas, se procederá a implementar cada una de las etapas mostradas en el esquema general del sistema (Figura 3-2), a través de un ejemplo de aplicación.

Como en el capítulo anterior, dado que se pretende utilizar tres escenarios de interacción, se tomará como ejemplo el último de ellos (interacción natural con retroalimentación háptica) ya que contiene todos los módulos del sistema.

4.1. Equipo y software utilizado

Con el objetivo de tener un sistema a bajo costo, se eligieron los siguientes elementos:

- **Hardware**
  - **Lentes de Realidad Virtual:** Oculus Quest v1
  - **Sensor de color para reconocimiento de objetos:** Logitech C920
  - **Computadora:** Dell Inspiron 15 7000 Gaming para operar el sistema de visión y virtual
  - **Escenario de reconocimiento:** Fondo blanco delimitado con marcadores
  - **Objetos reales:** Diferentes legos para las pruebas del mapeado real-virtual

- **Software**
  - **Lenguaje para el sistema visual:** Python con diferentes bibliotecas
  - **Motor de juegos para el sistema virtual:** Unreal Engine 4.26 a través de Blueprints

A continuación, se ahondará en las especificaciones de cada uno de estos elementos

4.1.1. Lentes de Realidad Virtual: Oculus Quest

Desarrollado por ©*Facebook Technologies, LLC.* Actualmente es uno de los cascos de Realidad Virtual con mayor popularidad dado que no requiere una computadora para su funcionamiento y tiene soporte para el seguimiento de las manos (Figura 4-1).

Figura 4-1: Oculus Quest v1 desarrollado por ©*Facebook Technologies, LLC.*. Recuperado en octubre 2021 de: https://www.flickr.com/photos/13815526@N02/44896768862

Cuenta con las siguientes especificaciones [Facebook Technologies, 2021]:

- **Pantalla**:
  - **Modelo:** Dual OLED 1600x1440
  - **Tasa de refresco:** 72Hz (por defecto), pudiéndose configurar a 60Hz

- **Espacio de color:** Rec.2020 gamut, 2.2 gamma, D65 punto blanco
    - **Estándar CIE 1931 xy valores de los colores primarios:**
        - **Rojo:** (0.708, 0.292)
        - **Verde:** (0.17, 0.797)
        - **Azul:** (0.131, 0.046)
        - **Blanco:** (0.3127, 0.3290)

- 1 Conector USB 3.0
- **Sistema de seguimiento:** *Inside Out* de 6 grados de libertad a través de 4 cámaras monocromáticas externas
- **Audio integrado** *in-strap*
- **CPU:** Qualcomm®Snapdragon 835 (puede correr a una frecuencia máxima de 2.3 ghz)
- **GPU:** Qualcomm®Adreno ™540 GPU (frecuencia máxima de 710 MHz)
- **Memoria RAM:** 4GB
- **Almacenamiento máximo:** 128 GB

El sistema de detección y seguimiento de manos lo realiza a través de las 4 cámaras externas y técnica de Machine Learning.

4.1.2. Sensor de color: Logitech C920

Desarrollado por la empresa ©*Logitech*. Proporciona una elevada resolución a un costo relativamente bajo (aproximadamente MXN 1,171.78.00) (Figura 4-2). Sus especificaciones [Logitech, 2021] son las siguientes:

- Modelo: C920
- Maxima resolución: 1080 p/30 fps - 720p/ 30 fps

- Campo de visión diagonal(*Diagonal Field of View*): 78^o
- Micrófono estéreo

Figura 4-2: Logitech C920. Recuperado en octubre 2021 de: https://www.flickr.com/photos/11435686@N03/36833914363

4.1.3. Computadora

Para el procesamiento de la imagen capturada por el sensor Logitech, entrenamiento de la CNN y los Oculus Quest, se utilizó una computadora marca Dell con sistema operativo Windows 10 Home. Las especificaciones son las siguientes:

- **Modelo del equipo:** Inspiron 15 7000 Gaming
- **Memoria Ram:** DDR4 de 16GB a 2400MHz
- **Almacenamiento:** 1 unidad SSD de 128GB + Disco Duro de 1TB y 5400 rpm
- **Puertos:** 3 puertos USB 3.0, 1 con PowerShare
- **CPU:** Intel®Core™i7-7700HQ de 7^a generación a 2.80GHz
- **GPU:** NVIDIA®GeForce®GTX 1050 Ti, memoria de video GDDR5 de 4GB

4.1.4. Configuración del Escenario Real y Objetos reales

Dado que se pretende interactuar en un lugar fijo se imprimió un escenario con los marcadores para poder delimitar la zona de interacción y mapeados. Se ubicó la cámara en frente del usuario para poder obtener la información del escenario y los objetos (Figura 4-3).

Figura 4-3: Configuración del escenario real para realizar las pruebas

Se utilizaron 5 tipos de legos diferentes de colores variables (Figura 4-4), con el objetivo de analizar el nivel de precisión con objetos pequeños y hasta qué volumen es posible tener un

sistema estable con base en el mapeado real-virtual

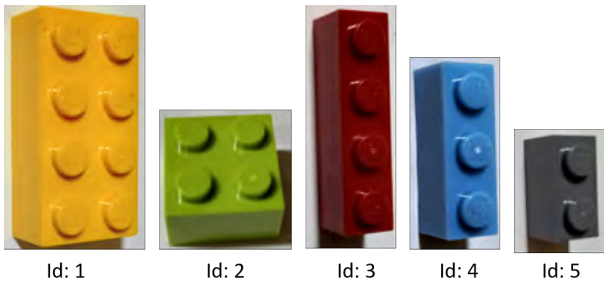

Figura 4-4: Objetos que se utilizarán para interactuar

4.1.5. Lenguaje para el sistema visual: Python

Este lenguaje de programación interpretado nos permite de una manera clara y directa comunicar las instrucciones fácilmente. A pesar de ser considerado un lenguaje lento en términos de tiempo comparado con un lenguaje compilado (ejecutados directamente por el procesador), actualmente a través del apoyo de diversas bibliotecas para visión computacional y otras aplicaciones, es considerado un lenguaje bastante robusto. La versión que se utilizó es la 3.8.7 que tiene compatibilidad con las bibliotecas utilizadas. Por tanto, a través de este lenguaje se podrá a la cámara y reconocer los objetos en el entorno.

4.1.5.1. Numpy

Dado que las imágenes se pueden interpretar como matrices, esta biblioteca de código abierto [Harris *et al.*, 2020] nos permitirá utilizar diversas funciones matemáticas que nos serán de gran apoyo para realizar las operaciones de transformación y procesamiento de imágenes en este sistema . Esta potente biblioteca es muy utilizada y es requisito para el uso de otras bibliotecas. La versión que se utilizó fue la 1.19.5.

4.1.5.2. OpenCV

Es una biblioteca de código abierto [Bradski, 2000], muy utilizada en el área de visión por computadora. El uso de ésta a través del lenguaje python agilizará el proceso de obtención de la imagen de la cámara, segmentación y operaciones de procesamiento de imágenes, lo cual es una gran ventaja ya que el sistema se realiza en tiempo real. La versión utilizada es la 4.5.1.

ArUco Dentro de la biblioteca OpenCV, se encuentra una biblioteca en torno a aplicaciones de Realidad Aumentada, llamada ArUco [Romero-Ramirez *et al.*, 2018; Garrido-Jurado *et al.*, 2015]. A través de la misma podemos reconocer los marcadores obteniendo propiedades como su número de identificación, la posición de sus vértices, el vértice origen y su orientación, lo cual nos será de gran ayuda al momento de detectar el escenario real.

4.1.5.3. Scipy

Es una biblioteca de código abierto Virtanen *et al.* [2020] exclusivamente para python, en donde podemos obtener funciones especiales de procesamiento de imágenes enfocadas principalmente en el área de computación científica. Versión utilizada 1.6.0.

4.1.5.4. Keras

Esta biblioteca [Chollet *et al.*, 2015] nos ayuda a que la estructura y los cálculos internos de la red sean construidos automáticamente sin tener que programarlos desde cero. Keras (versión 2.4.1) es un API de alto nivel que utiliza Tensorflow (biblioteca que nos aporta una gran variedad de recursos para el Machine Learning) como *backend* y se utiliza para construir y entrenar modelos de aprendizaje profundo incluyendo redes neuronales convolucionales.

4.1.6. Motor de juegos para el sistema virtual: Unreal Engine 4

Para el desarrollo de aplicaciones en Oculus Quest 4, se pueden utilizar dos tipos de Motor de Juegos: Unreal Engine o Unity. La ventaja de utilizar Unreal Engine [Epic Games], es la compatibilidad que tiene con el sistema operativo Windows 10 dado que se utiliza el lenguaje de

C++ para desarrollar las aplicaciones. Además, nos permite programar a través de un lenguaje basado en gráficos (Blueprints), lo cual nos ayudará a prototipar el sistema, así como combinar desarrollos con código crudo de C++.

Es importante destacar que la versión utilizada es 4.26, ya que a partir de ésta se incluye una extensión de programación para utilizar el seguimiento de las manos por parte de Oculus.

Blender Blender [Community, 2018] es un software utilizado principalmente en el modelado y animación de gráficos en 3D. Se utilizó esta herramienta para el modelado de los objetos virtuales con medidas similares a las del mundo real. Versión utilizada 2.82.

4.1.7. Configuración del Mundo Virtual

Se propone el ambiente virtual mostrado en la figura 4-5. Esto, con el objetivo, de crear un entorno relajado, libre de distracciones por elementos externos que puedan alterar la experiencia del usuario con respecto a la evaluación de los tipos de interacción.

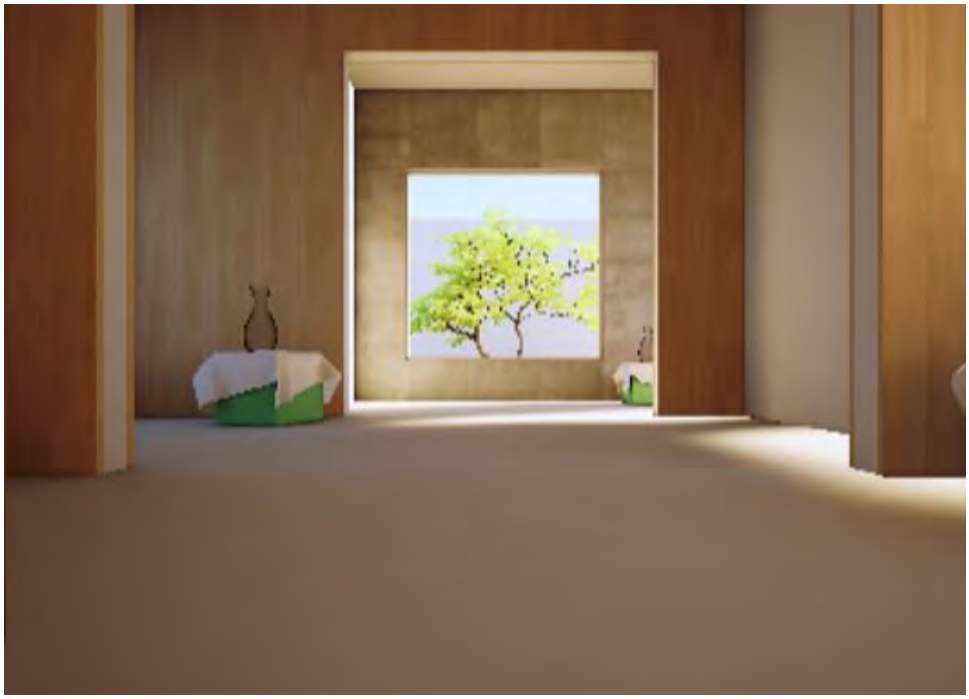

Figura 4-5: Configuración del escenario real para realizar las pruebas

4.2. Etapa de Configuración

Como se analizó en el capítulo anterior, se deben configurar y almacenar algunas variables necesarias para obtener el mapeado de los objetos en las etapas siguientes del sistema. El sistema de visión será desarrollado en lenguaje Python ocupando librerías tales como OpenCV, Tensorflow y Aruco. El sistema virtual será desarrollado en Unreal Engine 4, por medio del lenguaje de bloques Blueprints.

4.2.1. Sistema de visión

El objetivo en esta etapa es obtener la matriz de transformación H' que corrija la perspectiva y delimite el área en donde se van a reconocer los objetos. Siguiendo los pasos del diagrama de flujo de 3-6, tendríamos lo siguiente.

4.2.1.1. Detección de los marcadores en el escenario

Para este caso se seleccionaron 4 marcadores proporcionados por la base de datos de la biblioteca ArUco. Cada uno de estos marcadores (o markers) poseen un Id que permite reconocerlos y diferenciarlos (Figura 4-6).

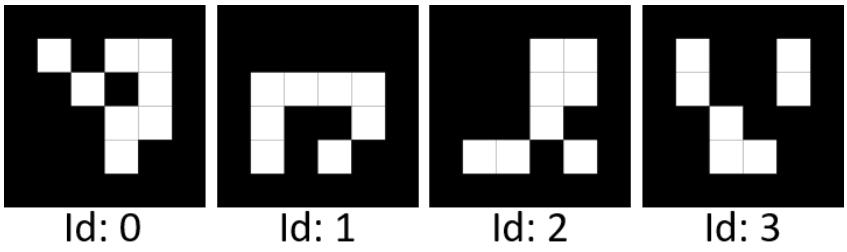

Figura 4-6: Marcadores utilizados para delimitar la zona de reconocimiento de los objetos

Para que se obtenga siempre la imagen desde la perspectiva del usuario y mapeado de puntos para corregir la perspectiva con base a los puntos de la ventana de visualización se deben colocar los markers en un orden definido por el desarrollador, por lo que se estableció el orden respecto

al id de cada marcador como se muestra en la figura 4-7. El objetivo de ocupar marcadores es permitir al usuario ampliar o reducir el área de interacción.

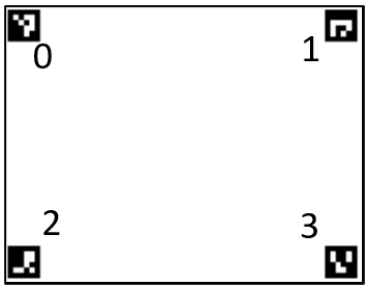

Figura 4-7: Orden de los marcadores desde la vista del usuario

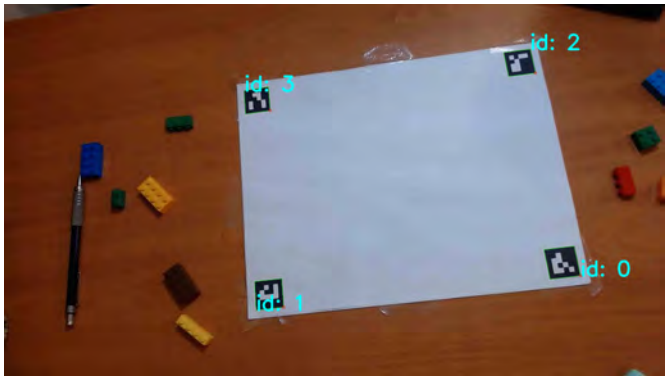

Figura 4-8: Detección e identificación de los markers

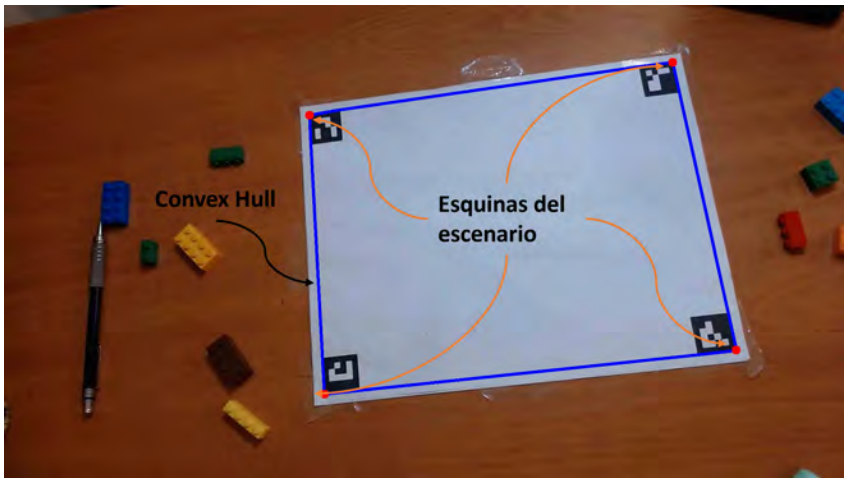

Figura 4-9: Obtención de las esquinas del escenario a través de Convex Hull

Por tanto, dado la imagen obtenida por la cámara, se detectan los marcadores obteniendo los vértices de cada uno de estos. Para obtener las esquinas del escenario que serán utilizadas para transformar geométricamente dicha figura en perspectiva a una forma rectangular que permita

corregir las líneas paralelas. Para ello necesitamos obtener las esquinas externas del escenario, obteniendo la figura convexa a través de la función convexHull() de la librería OpenCV. Esto nos dará el rectángulo más grande que abarca los 4 marcadores, cuyos vértices serán los límites de este escenario (Figura 4-9).

4.2.1.2. Correción de perspectiva

A través de la obtención de la matriz de transformación de perspectiva entre la figura del escenario en perspectiva y la figura rectangular que se desea mapear para corregir el problema de la proyección tendríamos dos procesos para obtener una correcta matriz de transformación como se mencionó en el capítulo anterior.

Primero se procedería a obtener la matriz de transformación H con base a las coordenadas actuales de las esquinas del escenario y las coordenadas en donde cada vértice del escenario debería corresponder (Figura 4-10)

Figura 4-10: Esquinas del escenario virtual.

Figura 4-11: Corrección de perspectiva

Una vez corregida las líneas paralelas del escenario, notamos que la relación de aspecto de la imagen con respecto al escenario real sufre una deformación en la altura. Esto es debido a que la corrección de la perspectiva se basa en corregir las líneas paralelas, rotación y traslación, mas no preserva las formas reales de los objetos. Para corregir esto, conocemos que los marcadores

son figuras cuadradas cuya relación de ancho y alto es la misma, por lo que sabremos si la imagen muestra una figura similar al escenario cuando los marcadores tenga una relación de alto y ancho igual. Es por ello que se obtiene el valor del ratio máximo a través de la división entre ancho y alto de cada uno de los 4 marcadores en la imagen (esto da mejor resultado que obtener el ratio promedio de los 4) y se multiplica por la segunda fila de la matriz de rotación logrando la correcta visualización del escenario como se muestra en la comparativa de la imagen 4-13

Figura 4-12: Correción de las líneas paralelas

Figura 4-13: Correción de la deformación

En este ejemplo, la matriz de transformación H' quedó de la siguiente manera:

$$H' = \begin{bmatrix} -2.5201 & 0.5592 & 3778.0967 \\ -0.2510 & -2.3643 & 2283.2446 \\ 0.0001 & 0.0002 & 1 \end{bmatrix} \tag{4-1}$$

4.2.2. Sistema virtual

Para el mundo virtual, debemos generar el escenario virtual para poder realizar la correspondencia de puntos entre el plano de la imagen y el plano del escenario virtual.

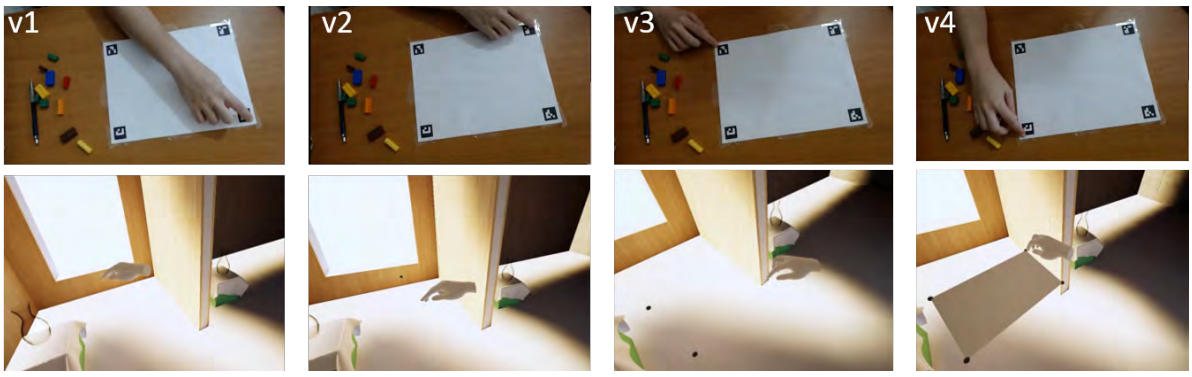

Figura 4-14: Pasos para generar el escenario virtual. Ejemplo basado en la figura 4-8 cuya cámara se encuentra en la misma posición

Para ello, a partir del índice de la mano derecha, se procede registrar las esquinas del escenario para obtener dichos vértices. El orden que se sigue es el mismo que se muestra en la figura 3-8. Una vez obtenido los vértices del escenario, se genera un elemento de Unreal Engine 4 conocido como *Procedural Mesh*, que permita crear objetos a partir de triángulos. Por tanto, se formará el triángulo v1-v2-v3 y el triángulo v3-v4-v1, que juntos conformarán el rectángulo que representa el escenario. Luego, se calcula la matriz del sistema de referencia local y los ángulos de rotación.

En el ejemplo de la imagen 4-14, se obtuvo la matriz de transformación al sistema global como se muestra en 4-2.

$$H = \begin{bmatrix} 0.132 & -0.9862 & -0.0458 & 36.9385 \\ 0.9910 & 0.1581 & 0.0276 & 75.1511 \\ -0.0214 & -0.0495 & 0.9981 & 195.482 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4-2}$$

Para los valores de rotación se obtuvo lo que se muestra en 4-3.

$$\theta_x = -2.844°, \theta_y = 1.226°, \theta_z = 82.413° \tag{4-3}$$

Las medidas de anchura y altura fueron de 27.2950cm y 18.9019cm. Que, comparado con las medidas reales del escenario, cuya anchura es 26.8cm y altura de 20.6cm presentan una diferencia de 0.495 cm en la anchura y 1.6981cm en la altura. Esto se debe tanto al tamaño del modelo de la mano, a la estimación de su profundidad en el mundo real y variaciones en la selección de las esquinas por parte del usuario.

4.3. Segunda Etapa: Mapeado del objeto real al entorno virtual

Luego de obtener la matriz de transformación H', seguiríamos la segmentación para localizar los objetos de interés y reconocerlos.

4.3.1. Segmentación por bordes

Los pasos para obtener la máscara de segmentación son los siguientes:

1. De la imagen obtenida por la cámara fija, se transformarán los puntos a través de la matriz H' para corregir la perspectiva como se muestra en la figura 4-16.

Figura 4-15: Imagen obtenida a través de la cámara

Figura 4-16: Correción de perspectiva con la matriz H'

2. Se realiza el procesado de la imagen para eliminar el ruido de la sombra poniendo el canal V del espacio de color HSV fijo (valor promedio del canal)

Figura 4-17: Imagen del escenario antes de quitar sombras

Figura 4-18: Imagen del escenario después de quitar sombras

3. Se convierte la imagen en escala de grises y se hallan los bordes utilizando el algoritmo de Canny. Para ello se utilizó valor de umbral mínimo de 20 y máximo de 30, con el objetivo de destacar detalles en las figuras.

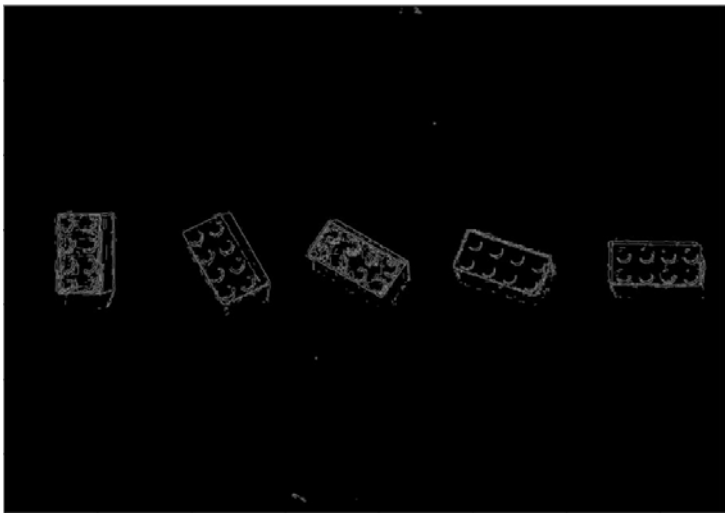

Figura 4-19: Obtención de bordes a partir de la imagen procesada

4. Se procede a dilatar los bordes para terminar de cerrar algunas figuras con un kernel 3x3

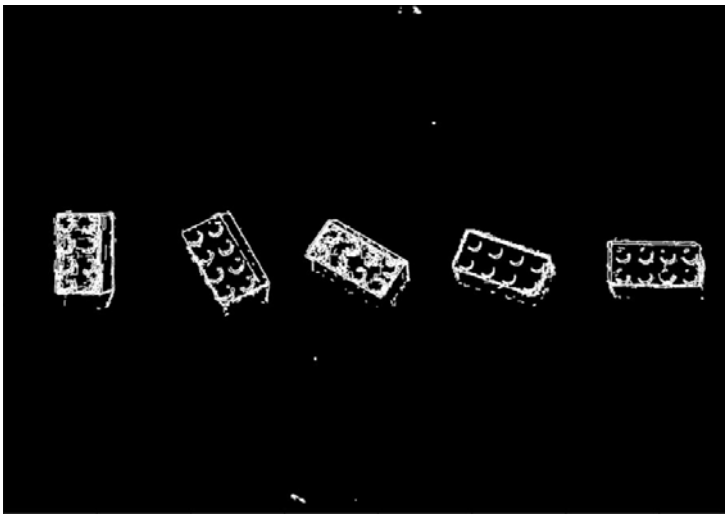

Figura 4-20: Filtro morfológico de dilatación con kernel de 3x3 para terminar de cerrar figuras obtenidas a través de bordes

5. Se rellenan las figuras cerradas a través de la función de cerrado de huecos proporcionado por Scipy [2021].

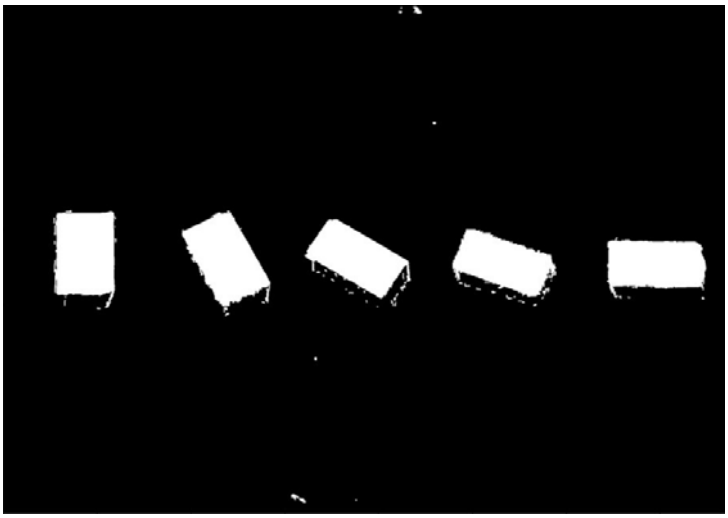

Figura 4-21: Llenado de figuras cerradas

6. Podemos observar en la figura 4-21 que existe el residuo de las sombras y otros ruidos obtenidos en la detección de bordes. Para ello se procede a limpiar aplicando filtro mor-

fológico conocido en OpenCV como opening = erosión + dilatación, con un kernel de 13x13

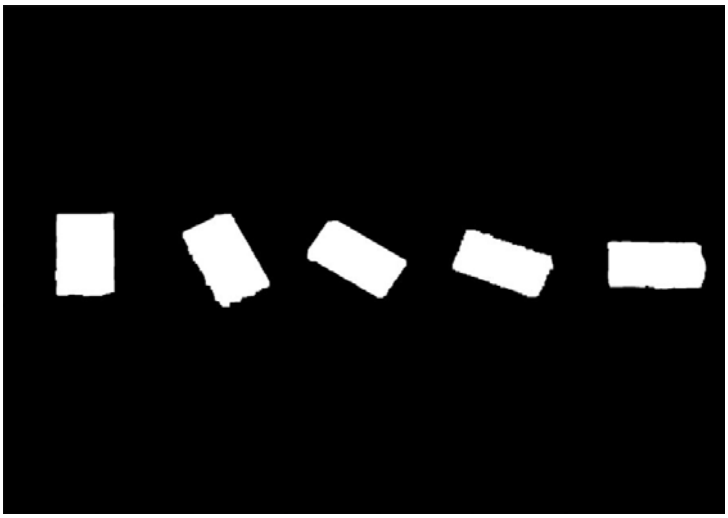

Figura 4-22: Eliminación de ruido excedente en figuras cerradas y alrededor del escenario a través de Opening con kernel de 13x13

7. Alisamos los bordes aplicando convexHull de [OpenCV, 2021]

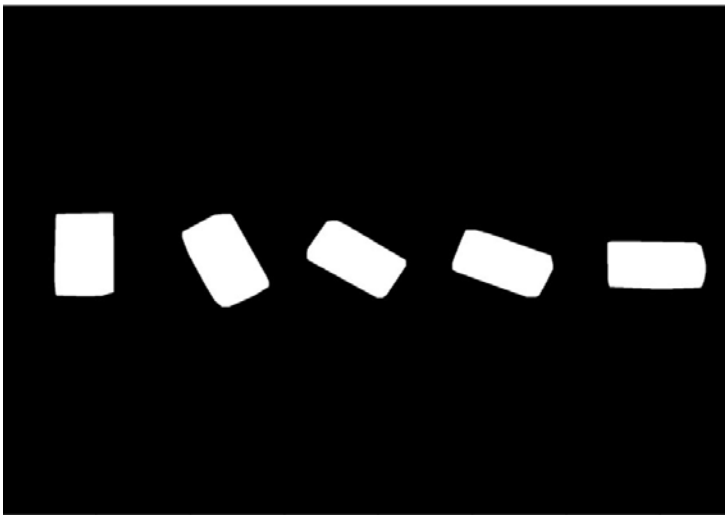

Figura 4-23: Convex Hull para alisar los bordes de las figuras. Resultado de máscara de segmentación final

8. Finalmente se procede a filtrar la imagen ya que pueden existir pequeños residuos detectados como figuras cerradas. Para ello se deberán encontrar las componentes conectadas, que es proporcionada por OpenCV con el algoritmo de [Grana *et al.*, 2010]. Dado que ésta función retorna el área de cada uno de los elementos se filtrará a través del área mínima permitida (en este caso $5000px^2$) teniendo como resultado en este caso la misma imagen 4-23

Dada la máscara de la figura 4-23 y la imagen de los legos corregidos 4-16, restaría multiplicar ambas imágenes para obtener el resultado final de la segmentación (Figura 4-24)

Figura 4-24: Resultado de la imagen segmentada a través de la máscara de segmentación de 4-23 y la imagen corregida de 4-16

4.3.2. Extracción de características

Como se explicó en el capítulo anterior, este módulo se encarga de obtener las propiedades de ubicación, orientación e identificación para poder notificarle al sistema virtual la existencia de objetos reales que deberá generar virtualmente.

El primer paso sería obtener el contorno de estas figuras cerradas (imagen 4-25), ya que a partir de estos puntos se podrán obtener las características de posición y orientación.

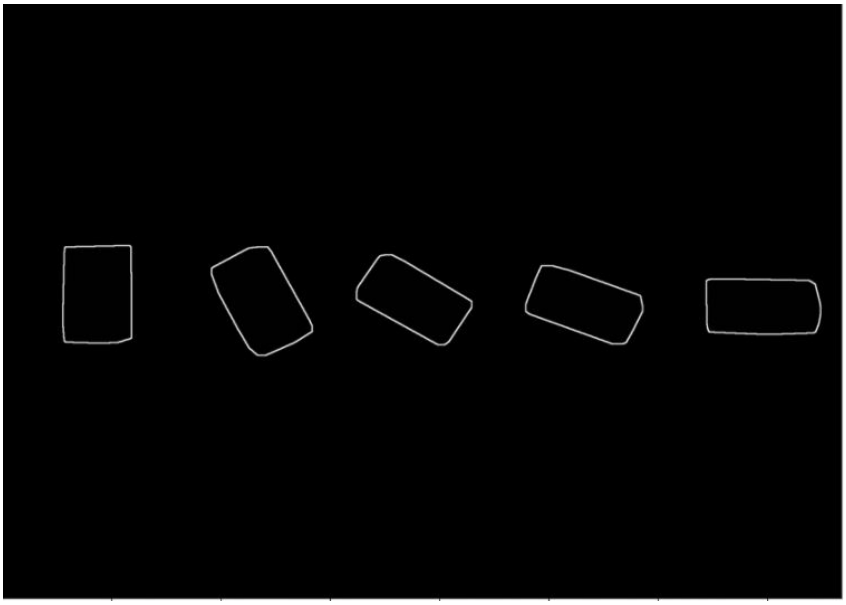

Figura 4-25: Obtención de contornos a partir de la máscara de segmentación

4.3.2.1. Obtención de la posición central y orientación

A través del cálculo de la recta que proporciona OpenCV y que define la orientación, podemos obtener el punto central y el ángulo. Para ello, aplicamos la función fitLine(), que se basa en la técnica de mínimos cuadrados promedio (LMS), tomando uno de los legos de la imagen 4-25, teniendo como resultado lo que muestra la figura 4-26.

Como resultado de la orientación del lego de la figura 4-26, se tiene el vector director de dicha recta $[0.4856, 0.8742]$ cuyo ángulo correspondería a $60.9487°$.

Es importante destacar que todos los legos tendrán un rango de orientación entre $[-90°, 90°]$ y no importará un origen dado que las esquinas son iguales.

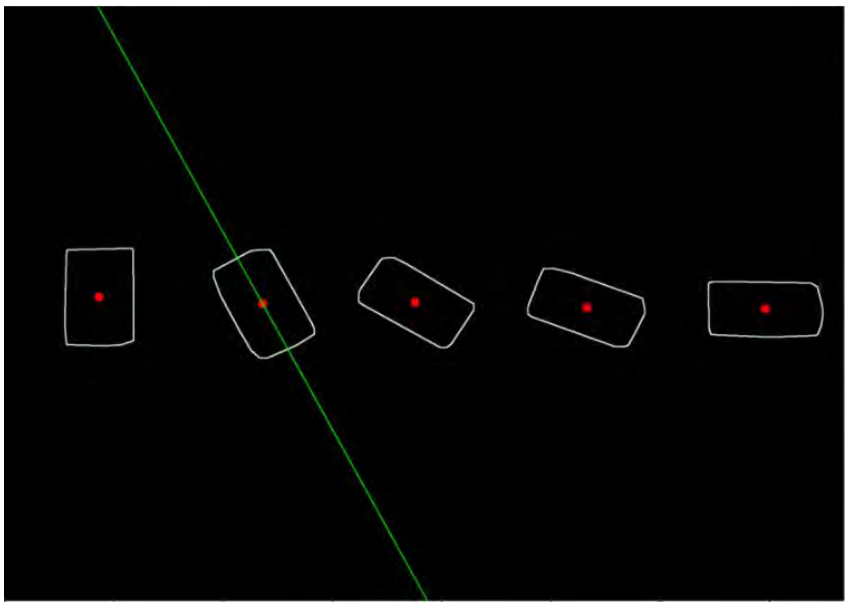

Figura 4-26: Posición central de cada lego y ejemplo del cálculo de la recta principal en el segundo lego para obtener el ángulo de rotación

4.3.3. Reconocimiento del tipo de objeto

Finalmente, el último paso antes de mandar la información al sistema virtual es reconocer el tipo de objeto. En este caso, se utilizaron 5 tipos de legos como se muestra en la figura 4-4.

4.3.3.1. Arquitectura de la red utilizada

Se propone la siguiente arquitectura de CNN en orden respectivo

- Entrada: 10000 neuronas (imagen 100x100 en escala de grises)
- Capas de Convolución:
    - Primera Capa de convolución: 32 filtros de tamaño 3x3
    - Submuestreo: Max pooling de tamaño 2x2
    - Segunda Capa de convolución: 64 filtros de tamaño 2x2
    - Submuestreo: Max pooling de tamaño 2x2

- Capa de clasificación:
  - Capa oculta: 256 con función de activación sigmoide
  - Dropout del 50 %. Función que permite apagar aleatoriamente ciertas neuronas para hacer la red más robusta
  - Capa de salida: 5 neuronas (Correspondiendo a los 5 tipos de legos)

Además, se utilizaron las siguientes variables:

- Tasa de aprendizaje: 0.001
- Función de coste: Entropía Cruzada Categórica (Categorical Cross Entropy). Método probabilístico
- Optimizador: Adam (algoritmo de descenso del gradiente estocástico)
- Función de activación en capas de convolución y pooling: relu
- Función de activación en red MLP: sigmoide
- Épocas: 20

4.3.3.2. Entrenamiento

Como se mencionó en el capítulo anterior, cada uno de los legos segmentados, se reajustará su tamaño a 100x100 en escala de grises, se normalizará el valor de los pixeles (entre 0 y 1) y estos entrarán al modelo de red, cuyo resultado será el id de dicho lego de entrada en código *One-Hot*[1] (ver ejemplo en Figura 4-27).

Se entrenó la red con un total de 5000 imágenes por cada lego y un conjunto de validación de 800 por cada uno, obtenidos de videos tomados desde diferentes perspectivas con legos en diferentes posiciones. A estos conjuntos se aplicaron transformaciones en las propiedades de color con un incremento del brillo variable entre 50 % y 100 %, y variación en los ángulos de rotación en el rango de $[0^o, 90^o]$.

[1] Grupo de bits en donde un valor en decimal corresponderá a uno de estos en 1 y todos los demás en 0

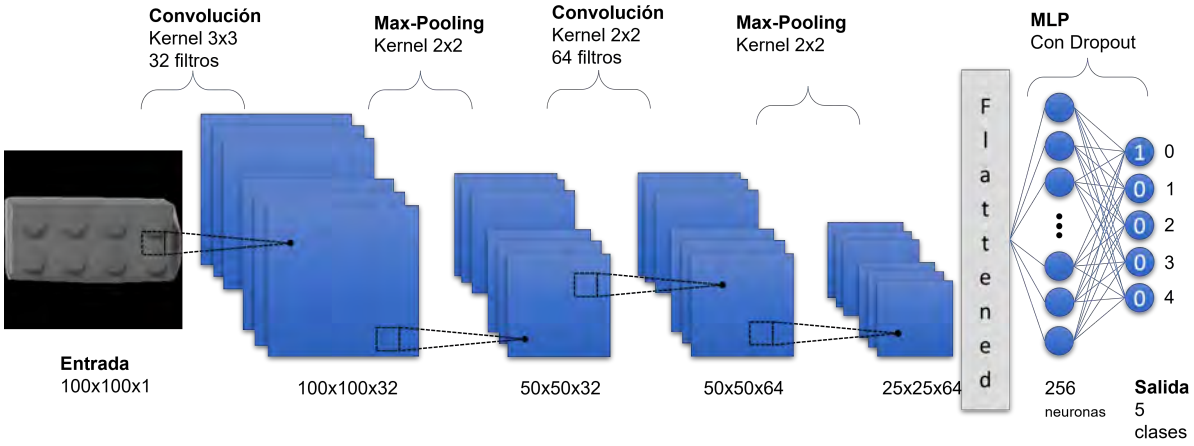

Figura 4-27: Ejemplo de predicción a través de la arquitectura de CNN propuesta

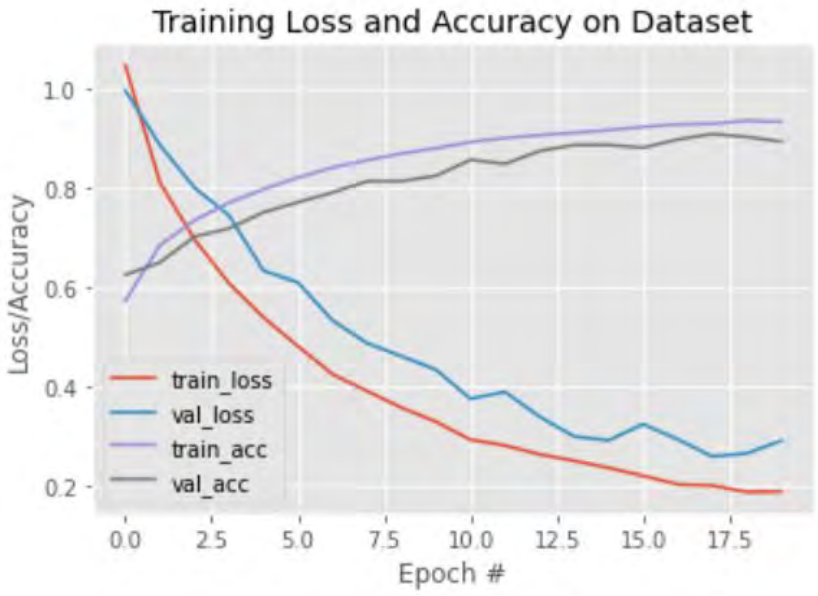

Figura 4-28: Resultado del entrenamiento con 25000 imágenes para entrenar y 4000 para validar

El resultado del entrenamiento se muestra en el gráfico 4-28, en donde se puede notar que tanto el conjunto de entrenamiento como el de validación logran alcanzar una precisión mayor a 80 % en 20 épocas, específicamente para el entrenamiento se logró 93.18 % y de validación un 89.23 %. El tiempo que tardó en converger fue aproximadamente 1 hora.

4.3.4. Envío de la información entre los dos ambientes

El envío de la información será a través de conexión TCP, implementando un modelo cliente-servidor como se muestra en la figura 4-29.

Figura 4-29: Modelo cliente-servidor para comunicar ambos sistemas

4.3.5. Mapeado del objeto real al sistema virtual

Para la generación del objeto virtual se deben aclarar los siguientes puntos:

- Las unidades de medida del sistema virtual corresponden a las medidas reales de nuestro mundo
- Partiendo del punto anterior, los modelos virtuales de los objetos a mapear serán modelados correspondiendo a las medidas reales de nuestro mundo

- El punto de origen de los objetos virtuales debe encontrarse en el centro de la base del mismo, para facilitar los cálculos en la correspondencia entre el centroide del objeto en la imagen y el objeto virtual

Modelos de los legos Para generar los legos, se crearon los modelos a través de un Software auxiliar modelador conocido como Blender, con las medidas reales como se muestra en la figura 4-30.

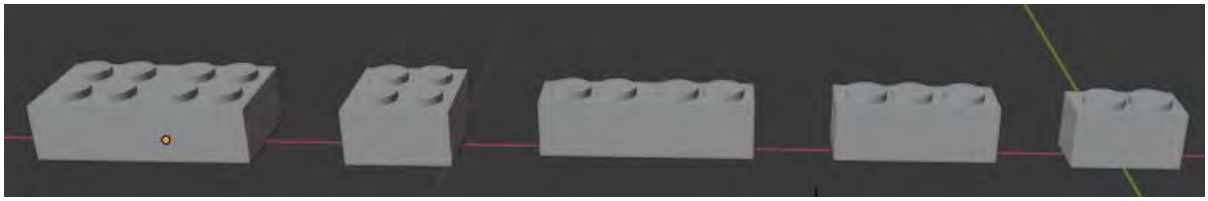

Figura 4-30: Modelos de los legos a través de Blender

Ejemplo de mapeado de objetos Por tanto dada la información recibida por el servidor, y los valores calculados en la etapa final, se procede a ubicar el modelo virtual en el escenario virtual, como se muestra en la figura 4-31.

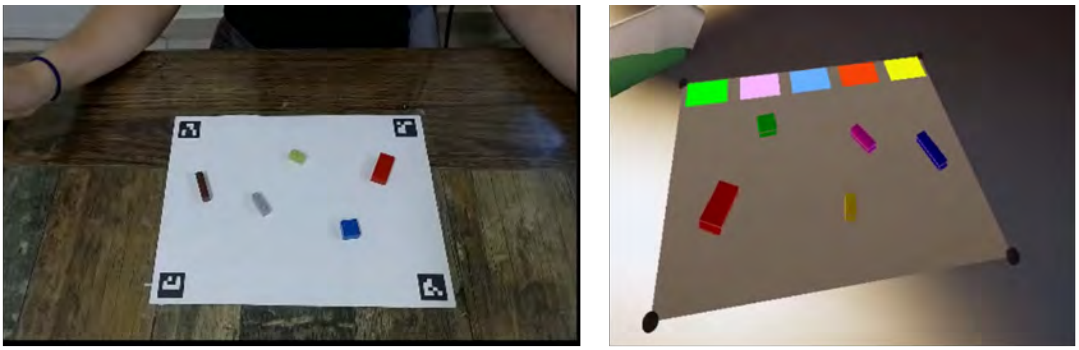

Figura 4-31: Mapeado de legos reales a los legos virtuales

4.4. Tercera etapa: Interacción con los objetos virtuales

Para el proceso final, el usuario deberá agarrar los objetos y dependiendo del escenario de pruebas será la forma de interacción utilizada.

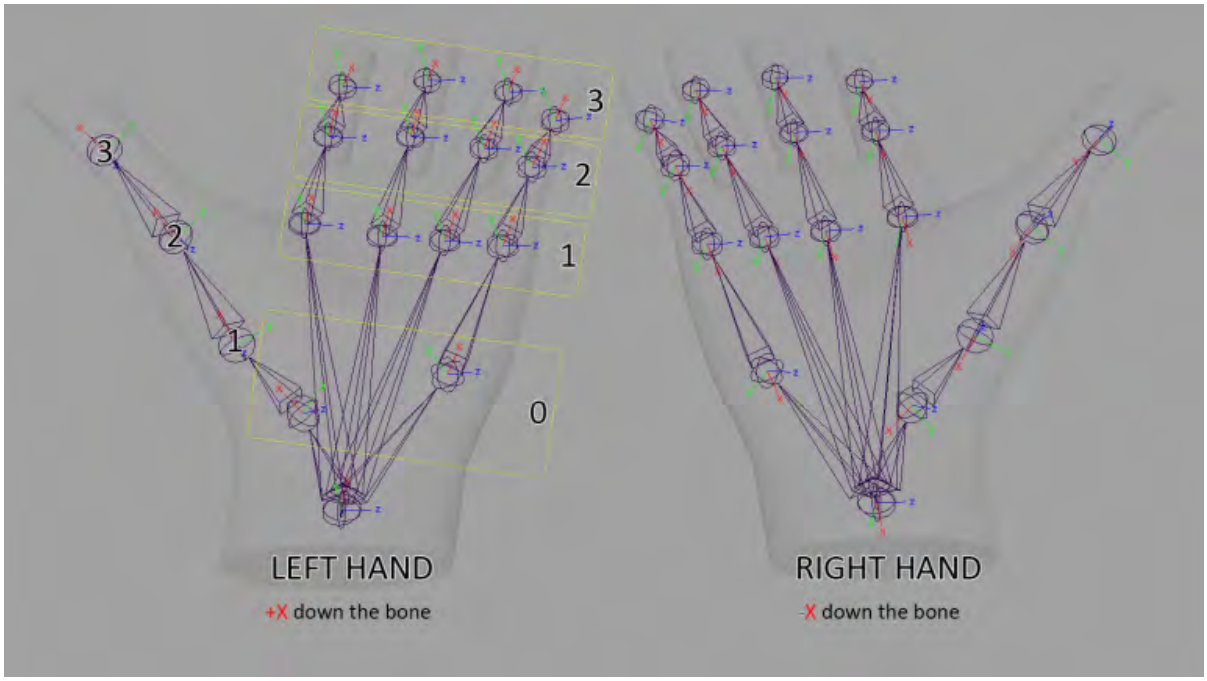

Figura 4-32: Esqueleto del modelo de las manos en Oculus. Imagen recuperada de https://uploadvr.com/oculus-quest-hand-tracking-sdk/

4.4.1. Modelo de las Manos en Oculus

El modelo de las manos del oculus, está compuesto por un conjunto de vértices y aristas que representan las uniones de los dedos y la muñeca, conocido como *skeleton*. Cada uno de estos elementos tiene su posición local con respecto al nodo de la muñeca y su posición global en el mundo virtual. Como se muestra en la figura 4-32, el modelo de la mano posee 18 grados de libertad que permitirán simular con mayor similitud las posturas de las manos.

El sistema de reconocimiento y seguimiento de las manos (o *hand tracking*), se obtienen a través de las cuatro cámaras monocromáticas que se encuentran a los costados del Oculus

en donde a través de técnicas de Machine Learning es capaz de detectar, seguir y estimar las posturas de las manos.

4.4.2. Colisión mano-objeto

A través de un componente conocido como *capsule collision* dentro de Unreal Engine 4, podemos obtener propiedades que permitan generar volumen a los elementos virtuales y evitar traspasarlo con las manos. Además, se pueden desencadenar eventos que avisen cuando existe una colisión con otro elemento y cuando se ha dejado de tocar dicho objeto. Esto nos ayuda a determinar cuándo es posible agarrar los objetos y cuando se puede liberar o soltar.

Por tanto, se asigna un *capsule collision* a cada dedo (nodos del nivel 3 en la imagen 4-32) y además, se debe asignar un *box collision* (parecido a capsule pero en forma de cubo) a los objetos virtuales para poder obtener las mismas propiedades y eventos sin afectar el modelo general del objeto.

4.4.2.1. Tocar el objeto

Dado que la punta de los dedos posee un colisionador, al existir contacto con el objeto virtual, este responderá como si existiera una fuerza externa, simulando el comportamiento cuando un objeto real es tocado o movido sin el agarre.

4.4.2.2. Agarrar el objeto

Cuando queremos agarrar un objeto en el mundo real, sin importar que posturas adoptemos de las manos se deben cumplir los siguientes casos:

- Debe existir el contacto de al menos dos dedos con el objeto
- La presencia del pulgar es constante, sobre todo en objetos de gran tamaño, excepto cuando existen objetos pequeños y ligeros donde se puede agarrar con el índice y el medio (no se considerará esta excepción en el sistema)

Por tanto, para implementar el agarre se considera el siguiente procedimiento, donde cada vez que exista un contacto con el objeto por cualquiera de los dedos, se ejecutará para determinar si es posible agarrar el objeto o no. En todo momento se estará actualizando una lista que indica cuál de los dedos actualmente tiene contacto con el objeto.

1. Si el pulgar y alguno de los otros dedos está tocando el objeto, ir al paso 2; si no, no agarrar el objeto.

2. Obtener el punto promedio entre la punta de los dedos que han tocado el objeto

3. Generar una esfera de colisión en la ubicación del punto promedio

4. Anclar el objeto a la esfera de colisión siempre que el pulgar y alguno de los dedos estén tocando el objeto

4.4.2.3. Soltar el objeto

Siguiendo las dos reglas establecidas en la sección 4.4.2.2, en caso de que no se cumpla alguna de estas, y el objeto se encuentra sostenido por la mano, este se liberará de la esfera de colisión creada por el promedio de los dedos en contacto.

4.4.3. Interacción con gesto de pinza

Para este tipo de interacción, solo se podrá agarrar el objeto con el gesto de pinza, que se identifica cuando el índice y el pulgar tienen contacto. Como se muestra en la figura 4-33, se pueden adoptar variaciones en los demás dedos, sin embargo, el contacto entre el pulgar y el índice siempre debe estar presente. La lógica de colisión se mantiene, es decir, siempre que exista un solo dedo en contacto con el objeto, este no será agarrado.

Figura 4-33: Variaciones de las posturas de las manos al realizar el gesto de pinza

Un ejemplo de este tipo de agarre interactuando con los objetos virtuales se muestra en la imagen 4-34.

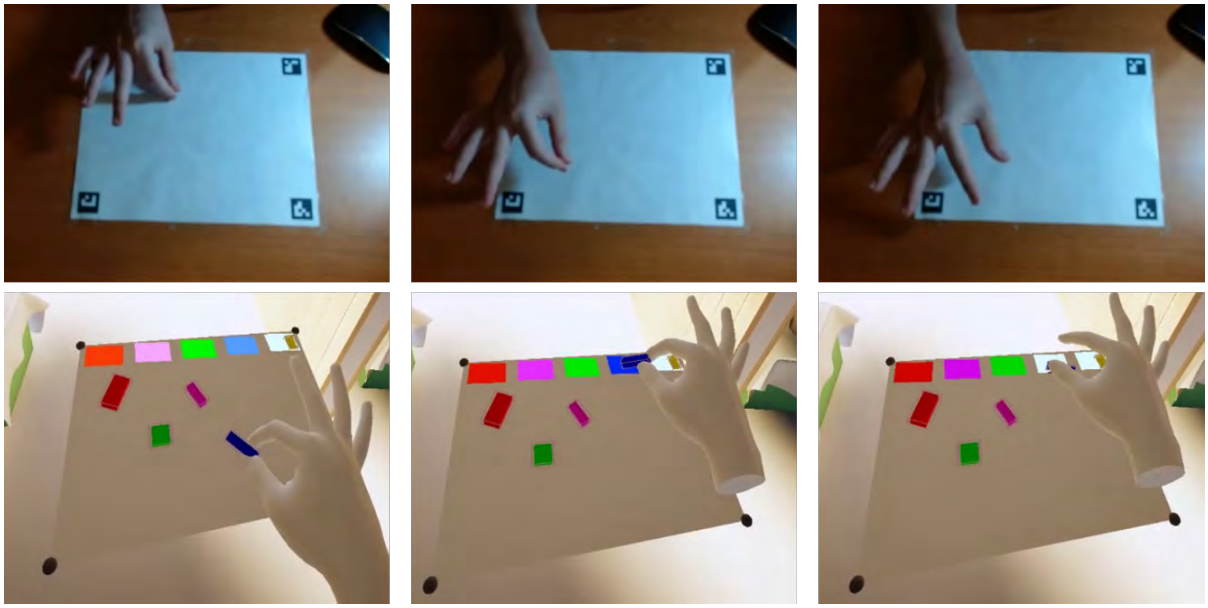

Figura 4-34: Ejemplo de interacción con gesto de pinza

4.4.4. Interacción con gesto natural sin tacto

Para esta interacción se le comunica al usuario que puede agarrar el objeto como si tuviera volumen (como si fuera real). A través de este método el usuario puede adoptar cualquier posición de las manos para agarrar el objeto.

Un ejemplo de este tipo de agarre se muestra en la imagen 4-35.

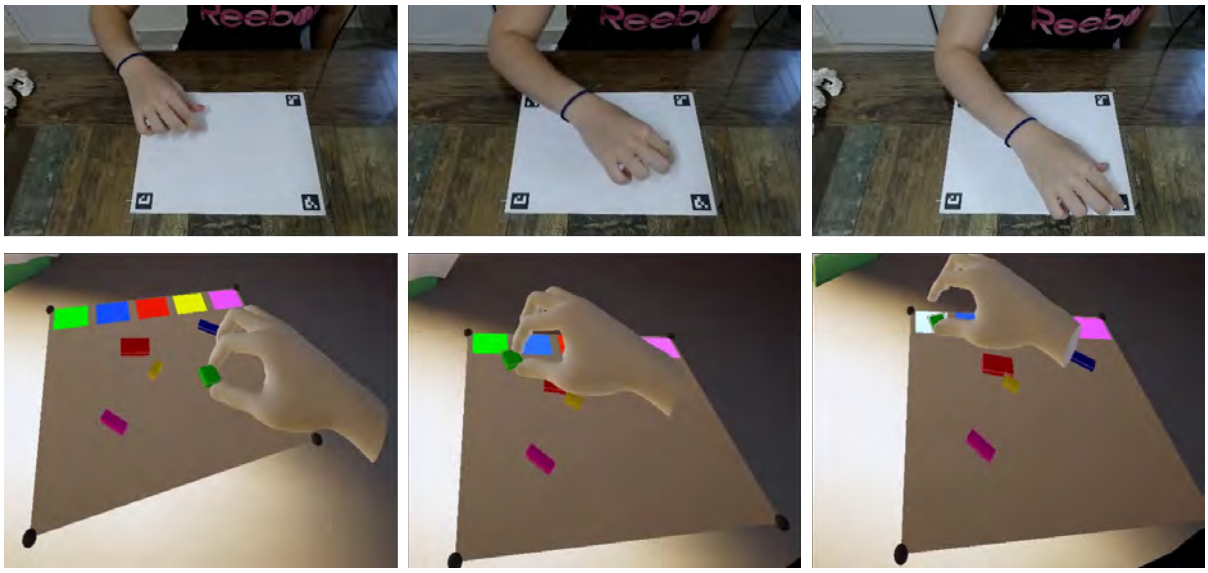

Figura 4-35: Ejemplo de interacción con gestos naturales sin háptica pasiva

4.4.5. Interacción con gesto natural con háptica pasiva

La única diferencia en la interacción con el método anterior es que el usuario recibirá retroalimentación háptica a través del mismo objeto en el mundo real.

Un ejemplo de este tipo de agarre se muestra en la imagen 4-36

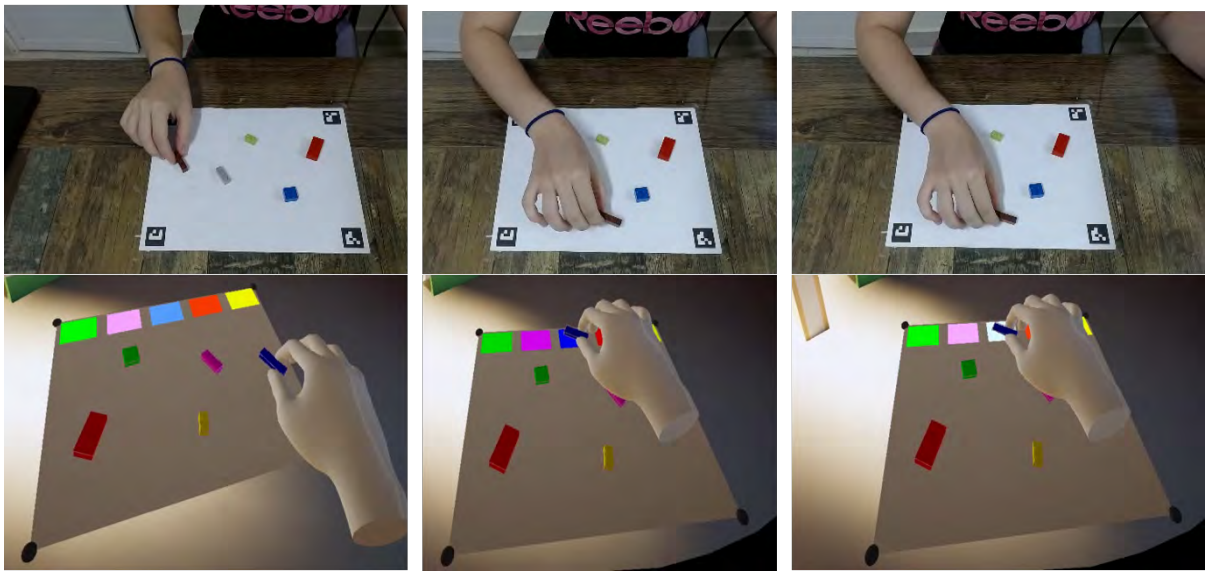

Figura 4-36: Ejemplo de interacción con gestos naturales con háptica pasiva

4.4.6. Demo para evaluar las 3 formas de interacción

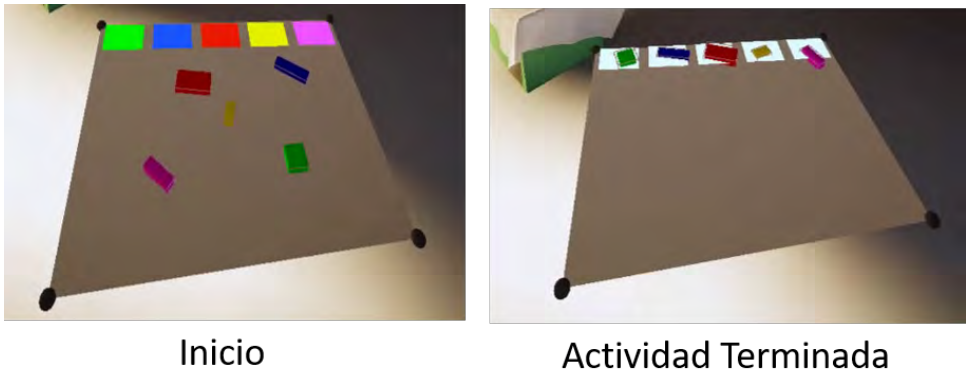

Figura 4-37: Actividad a realizar para probar los modos de interacción

El objetivo del demo es evaluar las variables consideradas en la sección 3.2, a través de una sencilla actividad.

Cada lego tendrá un color asignado. Dado el número de legos mapeados, se generará el

mismo número de regiones del color correspondiente a cada lego. El objetivo es ir colocando cada lego en el color correspondiente de la región generada (Figura 4-37).

Capítulo 5

Resultados y discusión

Aunque el objetivo final de este trabajo de tesis es identificar si los gestos naturales permiten mayor inmersión y eficiencia en usuarios finales, es importante considerar además los resultados con base en el funcionamiento del sistema ya que el resultado final depende del mismo. Para ello se dividirá la sección en resultados funcionales, relacionado con medidas comparativas entre el mundo real y el generado artificialmente, y, por causa de la contingencia sanitaria del Covid-19, las pruebas en usuarios finales fueron pocas, por lo que se dividió una sección de pruebas por un sólo individuo (el desarrollador) enfocadas en diferentes escenarios de interacción y otra sección en donde se aplicó el sistema a 12 usuarios tomando las medidas de higiene necesarias para evitar riesgos en la salud.

Los escenarios del demo 1, 2, 3, corresponderán a la interacción con el gesto de pinza, interacción natural sin háptica e interacción natural con háptica pasiva respectivamente.

5.1. Resultados Funcionales

Siguiendo la misma estructura que se tiene en la metodología de desarrollo esta sección se dividirá en cada una de las etapas, comparando medidas reales con respecto a las obtenidas tanto por el sistema de visión como por el sistema virtual.

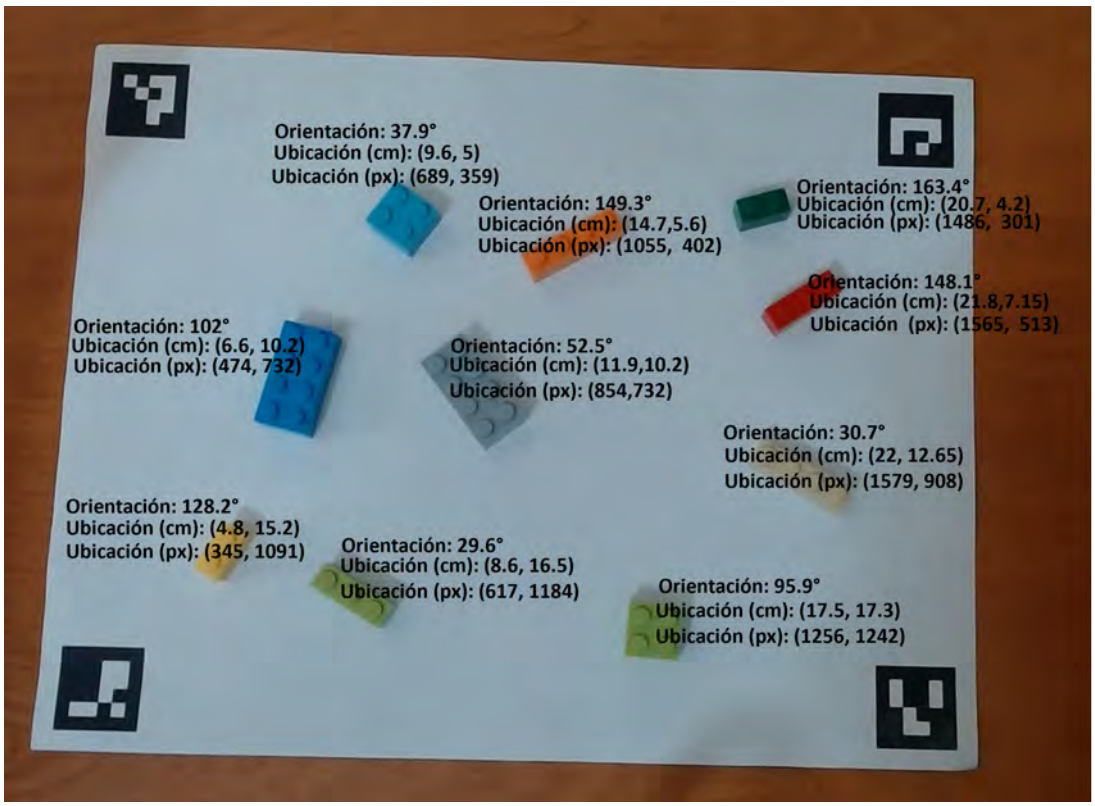

Figura 5-1: Posición de los legos en centímetros y su relativo en pixeles. El ancho y alto del escenario es de 26.75cm y 20.5 cm respectivamente; y sus dimensiones relativas en pixeles serían 1920 px y 1471 px

Las medidas del escenario real son de 26.75cm de anchura y 20.5cm de altura. Además, se configuró un escenario con legos en posiciones fijas y se tomaron las medidas reales que se muestran en la figura 5-1. Para conocer las posiciones correspondientes en una imagen las cuales se muestran en la misma figura 5-1, se tiene que el ancho siempre será fijo, en este caso, 1920 pixeles, por lo que la dimensión del alto será la que irá variando, dependiendo de la posición y condiciones de luz. Si consideramos que los pixeles serán cuadrados, entonces $1920px/26.75cm \approx 71.7757px/cm$, por lo que el alto ideal del escenario en pixeles debería ser $20.5cm * 71.7757px/cm \approx 1471px$. De este modo, los posiciones en centímetros de la figura 5-1

al ser multiplicadas por $71.7757 px/cm$ nos daría las ubicaciones ideales en pixeles.

Con respecto al sistema virtual, dado que las unidades de medidas son una aproximación a los valores reales, la comparación será centímetros.

5.1.1. Etapa de configuración: Sistema visual

El objetivo principal de esta sección es calcular la matriz de corrección de perspectiva para poder obtener correctamente las posiciones y orientaciones de los legos, por lo que se tomaron 9 imágenes desde diferentes perspectivas (distancia de cámara en centímetros, ángulo de visión en x y ángulo en y) y se midió la altura de la imagen en pixeles con respecto a la altura ideal del escenario de 1471 px, teniendo un error promedio absoluto de 99 pixeles o 1.379 cm (Tabla 5-1).

Distancia (cm)	**Ángulo en X**	**Ángulo en Y**	**Alto**	**Error (px)**	**Error (cm)**
16	55	90	1353	118	1.644026472
25	49	90	1359	112	1.560431905
26	135	90	1343	128	1.783350749
29	119	115	1405	66	0.91954023
33	92	93	1369	102	1.421107628
33	106	90	1370	101	1.4071752
34	115	100	1383	88	1.22605364
35	145	90	1372	99	1.379310345
52	66	110	1394	77	1.072796935
			Promedio	99	1.379310345

Tabla 5-1: Error absoluto en la altura del escenario a través del ajuste de la corrección de perspectiva

A través de los datos proporcionados por la tabla 5-1, podemos notar que a mayor distancia se tiende a disminuir el error como se muestra en la gráfica 5-2, por lo que es recomendable posicionar la cámara a una mayor altura. Además, existe una variación aproximada de 7.2 pixeles (0.1 cm) con posición fija de la cámara y cambios bruscos de intensidad.

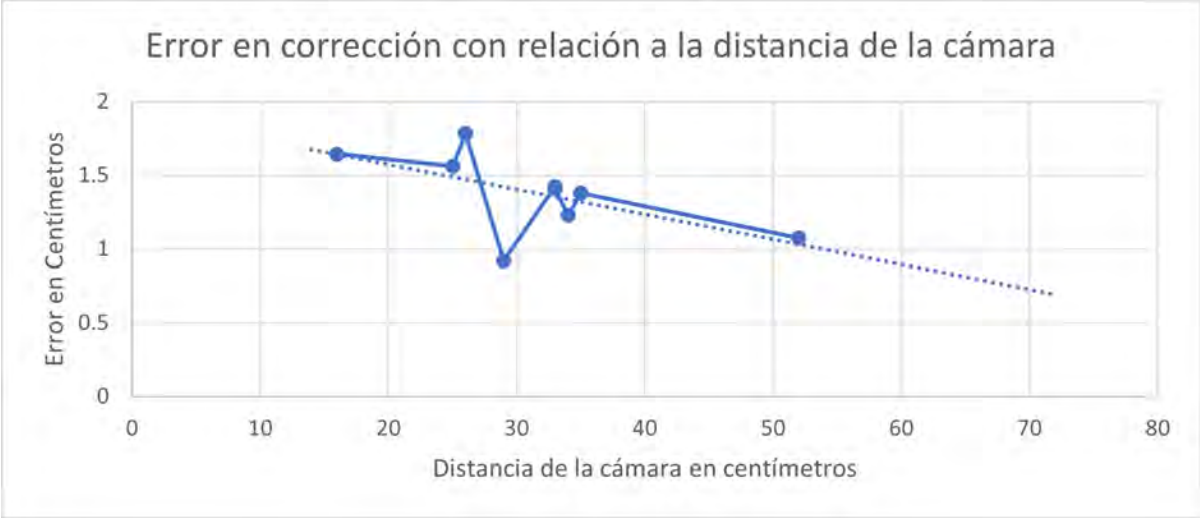

Figura 5-2: Tendencia a menor error absoluto en corrección de perspectiva a mayor distancia de la cámara

5.1.2. Etapa de configuración: Sistema virtual

Como se explicó en la implementación, las medidas del escenario virtual son obtenidas a través de la información espacial de las manos por el registro de las esquinas del mismo, por lo que se configuró el escenario 9 veces bajo diferentes condiciones de manos y cambios de luz (tabla 5-2) teniendo un error absoluto promedio de anchura de 0.66cm y 0.67cm de altura.

Intento	Condiciones	Ancho (cm)	Alto (cm)	Error Ancho (cm)	Error Alto (cm)
1	Condición de luz baja Manos con anillos	26.546579	21.775055	0.203421	1.275055
2	Condición de luz baja Semiguante oscuro	27.931269	18.794512	1.181269	1.705488
3	Condición de luz baja Guante completo gris	26.596674	19.622629	0.153326	0.877371
4	Condición de luz baja Guantes blancos	25.321651	21.617891	1.428349	1.117891
5	Condición de luz baja Manos libres de objetos	25.428276	20.879358	1.321724	0.379358
6	Condición de luz alta Manos libres de objetos	26.268555	20.359125	0.481445	0.140875
7	Condición de luz alta Manos libres de objetos	26.774496	20.314848	0.024496	0.185152
8	Condición de luz alta Manos libres de objetos	26.617279	20.799086	0.132721	0.299086
9	Condición de luz alta Manos libres de objetos	25.694098	20.408293	1.055902	0.091707
			Promedio	**0.664739**	**0.674665**

Tabla 5-2: Error absoluto promedio en la anchura y altura del escenario a través del registro en el mundo virtual

5.1.3. Segunda etapa: Sistema Visual

A través del sistema de segmentación podemos detectar los objetos que se encuentran dentro del escenario. Las posiciones calculadas en las imágenes de cada uno de los legos tienen un error

promedio de 11px (0.15cm) en la anchura y 62px (0.863cm) en la altura. El error promedio en la orientación es de 2.71^o. La efectividad en la detección de objetos, en 10 imágenes con 10 objetos en cada una es de 98 %.

Para el reconocimiento de objetos, a partir de 100 muestras en condiciones controladas (20 muestras por cada lego), se tiene una exactitud de 91 % y una precisión promedio del clasificador de 91.25 %, correspondiendo a la matriz de confusión 5-3.

		Clasificador CNN				
		1	2	3	4	5
Ground Truth	1	20				
	2		20			
	3			14	6	
	4			3	17	
	5					20

Tabla 5-3: Matriz de confusión de 100 objetos en un entorno controlado

En condiciones de luz variable, a partir de 10 imágenes y con un total de 99 elementos detectados se tiene la siguiente matriz de confusión 5-4 con 70 % de exactitud y precisión promedio del clasificador de 71 %. Se observó que en condiciones con mucha luz el clasificador reconoce mejor en comparación con entornos con poca luz.

		Clasificador CNN				
		1	2	3	4	5
Ground Truth	1	19	1			
	2	2	16	1		
	3		1	13	6	
	4	3	2	2	10	2
	5	1	3	1	4	12

Tabla 5-4: Matriz de confusión de 99 elementos en un entorno con luz variable

5.2. Resultados de los escenarios de interacción

Se efectuaron 6 repeticiones por cada demo, en donde se utilizaron los mismos 5 legos sin variar su posición para poder evaluar los 3 escenarios de interacción de una manera más precisa. Se midieron tiempos promedios para determinar la eficiencia de la interacción como se muestra en la tabla 5-5. Estas actividades se realizaron con diferentes variaciones de iluminación y artefactos en las manos tales como guantes blancos, negros y anillos para validar el sistema de reconocimiento de manos del oculus.

Se pudo observar que, al utilizar guantes oscuros, el sistema era inestable, sin embargo, con guantes blancos la precisión era similar a las manos sin ningún accesorio. Además, el seguimiento de las manos responde mejor en condiciones elevadas de iluminación.

	Tiempo promedio (segundos)		**Cantidad**		**Actividades Completadas (%)**
	Actividad Completada	**Reparar error**	**Errores**	**Actividades completadas**	
Escenario 1	3.31	2.4	21	30	100
Escenario 2	1.516	2.5	3	30	100
Escenario 3	2.266	2.66	3	30	100

Tabla 5-5: Medidas cuantitativas para medir la eficiencia de interacción en los diferentes escenarios de prueba.

Los errores que se cometen se refieren a cuando el objeto se cae de las manos o que no lo logren agarrar fácilmente.

5.3. Resultados de Aplicación en usuarios

El demo de los 3 escenarios se realizó con un total de 12 usuarios de edades entre 27 y 62 años. La relación entre la preferencia del tipo de interacción con respecto a las edades se muestra en la figura 5-3. Con esto se puede observar que existe preferencia hacia algo más

natural en usuarios menores de 35 años, lo que concuerda con el estudio de [Xue *et al.*, 2020] que menciona que las edades con mayor aceptación de VR se encuentran entre 18 y 34 años. En usuarios mayores se observa una tendencia hacia una interacción más limitada, pero más sencilla.

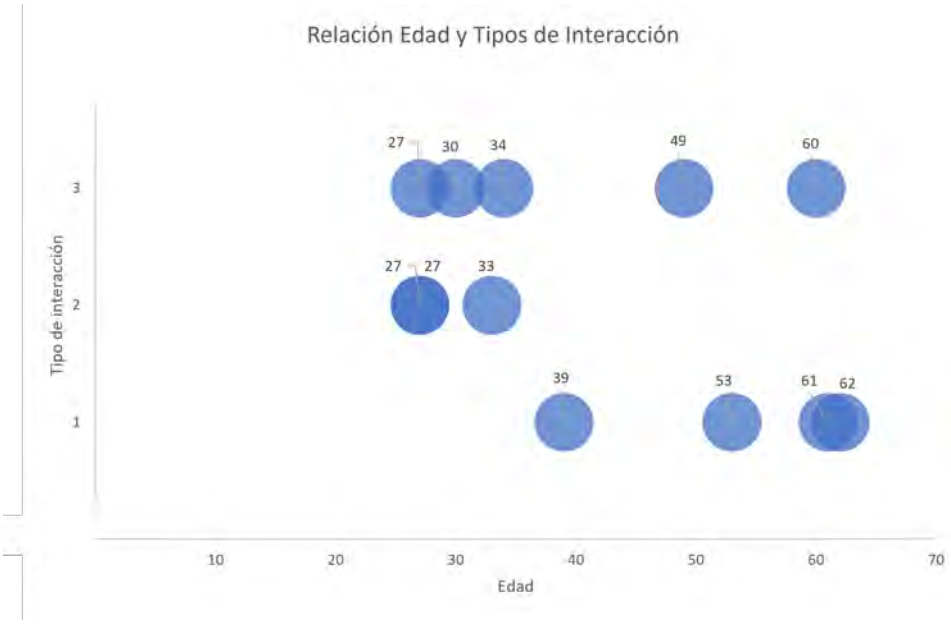

Figura 5-3: Relación entre la edad y el tipo de interacción

En cuanto a las variables cualitativas respecto a cada una de las formas de interacción se puede observar en la figura 5-4, que las interacciones naturales permiten una conexión más directa con el objeto virtual, lo cual implica una mayor inmersión.

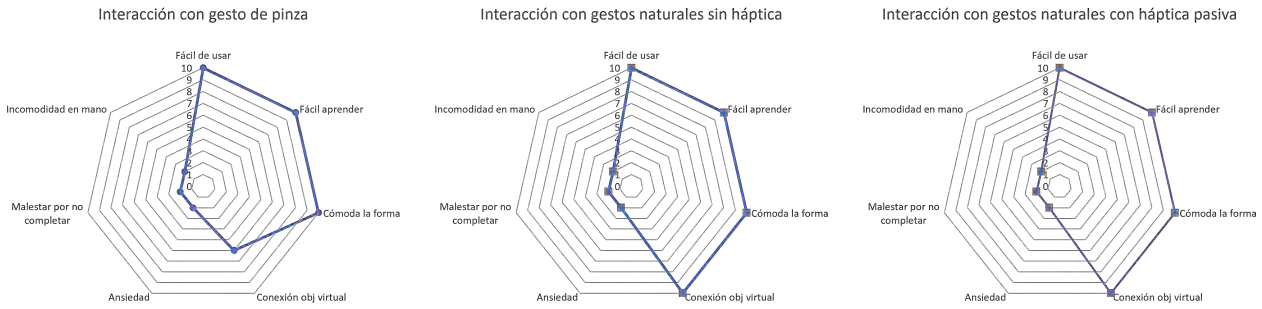

Figura 5-4: Variables cualitativas que miden la experiencia del usuario por cada interacción

Finalmente se muestran los resultados respecto a los valores cuantitativos (tiempos, etc.) para determinar la eficiencia de los tipos de interacciones en la tabla 5.3.

	Tiempo promedio (segundos)		**Cantidad**		**Actividades Completadas (%)**
	Actividad Completada	**Reparar error**	**Errores**	**Actividades completadas**	
Escenario 1	6.29412	5.0625	35	51	85
Escenario 2	6.2963	1.05	34	54	90
Escenario 3	5.80645	0.1	2	34	56.66

Tabla 5-6: Medidas cuantitativas para medir la eficiencia de interacción a partir de 12 usuarios de edades entre 27 y 62.

5.4. Discusión

Con respecto al funcionamiento del sistema se observa lo siguiente:

- En la etapa de configuración por parte del sistema de la cámara, a mayor distancia de la misma, mejor corrección de la perspectiva, esto podría deberse a que la cámara presente cierta distorsión, por tanto, si se encuentra a una distancia mayor, el área que abarcaría el escenario contemplaría menos niveles de distorsión. Además, mientras el ángulo de

la cámara se encuentre entre $[55^o, 145^o]$ la detección de los marcadores por parte de la biblioteca arUco no presenta problemas en la detección. Sin embargo, es importante que la superficie donde se encuentran los marcadores no presente deformaciones ya que afectaría en el ajuste de la corrección de la perspectiva.

- En la etapa de configuración por parte del sistema virtual, tenemos que el sistema de seguimiento de manos responde mejor en condiciones de luz elevada y color de las manos claras. El registro de los puntos del escenario tiende a variar en la esquina superior derecha, dado que el sistema de reconocimiento le cuesta predecir la posición del dedo índice desde esa perspectiva, por lo que es recomendable acercar las cámaras del lente (mirar con mayor atención al dedo índice) para que pueda capturar de manera espacial la posición del vértice del escenario.

- En la segunda etapa por parte del sistema visual, tenemos que la corrección de sombras responde muy bien en entornos donde no se encuentren muy oscuras y los legos contrasten en el color con respecto a la sombra. El modelo de legos utilizados refleja en algunas situaciones la luz impidiendo la correcta detección de los mismos. La etapa de segmentación responde mejor a legos que contrasten con el fondo del escenario (blanco), esto se debe a que se utiliza una variación del color junto con detección de bordes cuyo objetivo principal es encontrar cambios de intensidad, por lo que a menor contraste mayor probabilidad de no ser detectados. El sistema de reconocimiento responde mejor en condiciones de luz elevada y tiene mayor confusión entre los legos 3 y 4, dado que presentan características muy similares.

- En la etapa de interacción el modelo de las manos obtenida por el Oculus suele tener cierta inestabilidad en las posiciones de los dedos que afectan las condiciones de agarre y desplazamiento de los objetos virtuales. Esto se debe a que se tienen modelos de colisión en cada dedo, condicionados a estar en contacto o no con el objeto. Al existir cierta variación puede ocurrir que el pulgar u otro dedo principal pierda el contacto con el objeto y este se caiga. Las físicas de los objetos virtuales lograban una simulación bastante similar a

la de los objetos reales al momento del agarre y desplazamiento, sin embargo, cuando se soltaba el lego el comportamiento era bastante diferente, dado que no se toma en cuenta valores reales de fricción, por lo que en situaciones en donde se caía de la mano por error, la orientación y ubicación podía cambiar dificultando la búsqueda del objeto real-virtual

Con respecto a las evaluaciones en usuarios, la interacción con retroalimentación háptica pasiva nos ofrece una solución a bajo costo de modo que se pueda acomodar la postura de la mano y por tanto ofrece flexibilidad en la interacción, sin embargo, predecir este tipo de posturas a partir de las cámaras de los lentes puede generar errores en el modelo de la mano, ya que desde ciertos ángulos no es posible obtener las posiciones de los dedos. Esto ocasiona errores funcionales cuya consecuencia puede generar disonancia cognitiva, perdiendo completamente el concepto de inmersión en ambientes virtuales. La simulación de las físicas de los objetos virtuales no es exacta, por lo que existieron situaciones en donde dado un mal agarre del usuario, el comportamiento del objeto virtual con el real desincronizó la ubicación espacial entre ambos. En efecto, en los escenarios de prueba, ocurrieron este tipo de situaciones, lo cual influyó enormemente en la decisión del tipo de interacción natural sin tacto.

En la relación con la gráfica de la figura 5-3, se muestran dos usuarios mayores de 34 años cuya preferencia fue la interacción con tacto. Estos dos usuarios son considerados como *outliers*, ya que ambos tenían experiencia en videojuegos y un gusto muy amplio de diferentes tecnologías, por lo que la flexibilidad en la interacción y el interés influyeron en su resultado final.

En las pruebas con los usuarios se pudo notar que los legos más grandes y alargados tienen mayor precisión en el agarre, optando siempre por sujetar la parte más alargada del objeto; mientras que el lego de forma cuadrada generó en el usuario dudas de como agarrarlo por lo que su respuesta fue más lenta y con mayores errores.

Dado el análisis de los tiempos promedios de la tabla 5-5 realizado por un usuario, se observa que existe mayor eficiencia en completar las actividades y menor error en la interacción natural sin el tacto. En las pruebas con los 12 usuarios la interacción háptica logró menor tiempo, no obstante, sólo se lograron completar 34 actividades de un total de 60 debido a la inestabilidad en la detección de las manos y las físicas de los objetos virtuales.

Además, luego de realizar las pruebas en los 12 usuarios podemos observar lo siguiente:

- Uno de los usuarios adulto mayor, sintió incomodidad y mareo al estar en el mundo virtual, lo cual pudo haber influenciado en el resultado final
- Existe una tendencia a que la edad defina el grado de libertad en la interacción. A mayor edad, mayor preferencia de un gesto sencillo que simplifique la interacción. Los usuarios que eligieron la interacción de pinza justifican su preferencia debido a que "sienten mayor estabilidad y comodidad en la interacción sin tanto temblor en las manos". El sentido de inmersión no es una motivación ya que comentan que "tienen muy separada la realidad del mundo virtual" y les causa "inestabilidad emocional la sensación del mapeado objeto real-virtual"
- En usuarios más jóvenes, tener libertad de manipulación es sinónimo de mejor respuesta y mayor preferencia del modo de interactuar naturalmente. Además, disfrutan la inmersión y la sensación de que están tocando algo como si fuera real
- A pesar de que no existe retroalimentación háptica en el escenario 2, varios usuarios comentaron que tenían la sensación de "tacto" al ocluir los dedos con el objeto virtual. La elección del escenario 3 siempre fue puesta en duda con el escenario 2, por lo que es posible que por sesgo de sentir el tacto en pocas ocasionan se decidieran más por el escenario 3
- En personas de piel oscura, la detección y predicción de posturas de las manos por el Oculus fue inestable, por lo que en 2 usuarios (edades 34 y 39) existieron errores en la forma de interactuar debido a este problema y pudo haber alterar el resultado final respecto al modo de interacción, dado que generó ansiedad en la ejecución de las actividades. Es por ello que en estos casos cuando la mano no responde correctamente, se descarta la actividad con el lego correspondiente.
- La elección del mundo virtual fue satisfactoria dado que a todos los usuarios le pareció relajante, tranquilo y no existieron distracciones en torno a las actividades.

- Se obtuvo mejores tiempos y menores errores con el lego más grande (lego 1), por lo que la tendencia hacia objetos mayores podría permitir menores errores y mejores tiempos al completar las actividades

Es necesario realizar pruebas con más usuarios para tener un resultado más sólido, sin embargo, la tendencia de preferir un modo de interacción está estrechamente relacionada a la edad del usuario.

Capítulo 6

Conclusiones y trabajo futuro

En el presente trabajo se mostró un sistema de interacción en Realidad Virtual en el cual se pudo comprobar que el uso de una manipulación natural permite mayor libertad en la interacción lo cual implica mayor inmersión y conexión real-virtual. Además, la interacción natural sin tacto ha sido el tipo de interacción más eficiente, esto es, mayor rapidez en realizar las actividades y menor números de errores al agarrar los objetos virtuales, sin embargo, la preferencia de los tipos de interacción ha sido diferenciada con respecto a la edad de los usuarios. Por el momento, podemos observar que la interacción natural es elegida por personas menores de 35 años y el uso del gesto de la pinza por mayores de 38. La elección del gesto de pinza en los adultos mayores está relacionado respecto a la estabilidad del sistema locomotor de la mano y a la sencillez del gesto.

La interacción con retroalimentación háptica pasiva, contrario a lo que se pensó que mejoraría en eficiencia e inmersión, fue el escenario más inestable en cuestiones de tiempo y experiencia de usuario. Esto podría deberse a la inestabilidad del sistema en el seguimiento de las manos por parte del Oculus, así como la sincronización real-virtual, ya que no se actualiza la posición y orientación en tiempo real luego de mandar los datos al sistema virtual, de modo que si existe una variación en las físicas del objeto virtual el desfase podría generar incomodidad y sesgo en la elección del tipo de interacción. Además, el uso de háptica pasiva tiene la desventaja de estar sujetos a lugares fijos de interacción.

Se propone finalmente el uso de los gestos naturales en actividades generales incluyendo el gesto de pinza, ya que en algunos casos puede formar parte de lo que es un gesto natural (gesto similar al que realizamos al interactuar con elementos muy pequeños). Así, de manera híbrida se podrá interactuar con los objetos virtuales para permitir al usuario la opción más conveniente dependiendo del tipo de actividad general a realizar. El tipo de retroalimentación al contacto con los objetos virtuales se puede proporcionar vía visual, cambiando el color de los objetos, etc. En situaciones en donde sea necesario la retroalimentación háptica, por ejemplo, rehabilitación en las manos a través de objetos físicos, una opción a bajo costo y llamativa sería por medio de los gestos naturales con tacto, generando mayor interés en las actividades médicas en el usuario.

6.1. Trabajo Futuro

Las limitantes en el aspecto visual y virtual se encuentran presentes sobre todo en el escenario 3 (retroalimentación háptica), dado que es el caso más complejo computacionalmente. Si se mejora el funcionamiento de éste, es posible que exista mayor aceptación, por lo que en este aspecto de háptica pasiva se proponen las siguientes mejoras del trabajo actual:

- Seguimiento de los objetos reales por medio de la cámara, para realizar un mapeado en tiempo real y así evitar la inestabilidad en la simulación de las físicas de los objetos virtuales
- Incorporación de ambas manos para ampliar el modo de interacción con objetos de mayor tamaño
- Generación de objetos virtuales que complementen o agregue más información en torno a los objetos virtuales conocidos por el sistema
- Reconstrucción virtual de objetos reales por medio de la forma de estos (se propone utilizar un sensor de profundidad)

Además, con el objetivo de evaluar la comodidad en la interacción natural, se propone la incorporación de objetos de diferentes tamaños y formas, de modo que se pueda obtener un estándar de los tamaños máximos permitidos para que la interacción natural sea eficiente y cómoda.

Finalmente, se propone la aplicación de la técnica de manipulación natural en áreas como:

- **Interacción natural con háptica pasiva en:** Rehabilitación, tratamiento de fobias, cirugías a través del uso de instrumentos reales
- **Interacción natural sin háptica:** juegos de construcción, instrumentos musicales, consolas de diferentes tipos (producción musical), simulación del teclado de la computadora para escritura tipográfica, juego de manipulación de plastilina virtual para el desarrollo de formas

Apéndice A

Convolución espacial

La convolución es una operación matemática que tiene como objetivo a partir de dos funciones, llámese f y g (en cualquier tiempo o espacio), obtener otra función $h = f * g$ que representa un conjunto de valores acumulados respecto a la superposición entre f y una versión traslada e invertida de g (Figura A-1)

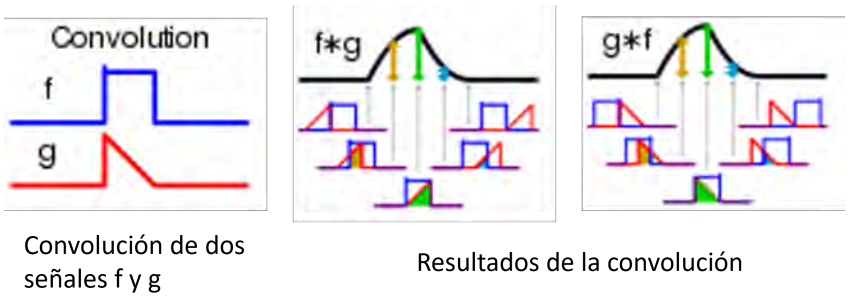

Figura A-1: Convolución de dos señales. Imagen recuperada en septiembre 2021 de: https://es.wikipedia.org/wiki/Convoluci%C3%B3n

En el dominio continuo tendríamos que la ecuación que representa a la convolución en los intervalos $(-\infty, \infty)$ está dada por la siguiente integral:

$$f * g = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \tag{A-1}$$

En el caso de que esta convolución ocurra en un intervalo de tiempo finito $[0, t]$:

$$h(t) = (f * g)(t) = \int_0^t f(\tau)g(t - \tau)d\tau \tag{A-2}$$

Donde en el dominio discreto, la ecuación A-2 resultaría en:

$$h[n_c] = f[n_a] * g[n_b] = \sum_{\tau=0}^{n_a-1} f[\tau]g[n_c - \tau], \text{ donde } n_c \text{ estaría en el rango de } 0 \leq n_c < n_a + n_b - 1 \tag{A-3}$$

Dado que una imagen posee dos dimensiones, si quisiéramos aplicar la convolución discreta en este dominio espacial, tendríamos entonces que:

$$h[x_c, y_c] = f[x_a, y_a] * g[x_b, y_b] = \sum_{\tau_1=0}^{x_a-1} \sum_{\tau_2=0}^{y_a-1} f[\tau_1, \tau_2]g[x_c - \tau_1, y_c - \tau_2] \tag{A-4}$$

Los valores $[x_a, y_a]$ corresponden a los pixeles de la imagen de entrada y llamaremos a la función $g[x_b, y_b]$ el filtro (o kernel) de convolución que al operar de forma iterativa con la imagen de entrada $f[x_a, y_a]$, tendremos la imagen de salida filtrada $h[x_c, y_c]$ (Figura A-2).

El tamaño de la imagen de salida $n_{out} \times n_{out}$, el tamaño de la imagen de entrada $n_{in} \times n_{in}$ y el tamaño del filtro $f \times f$ estaría dada entonces por:

$$n_{out} = (n_{in} - f + 1) \tag{A-5}$$

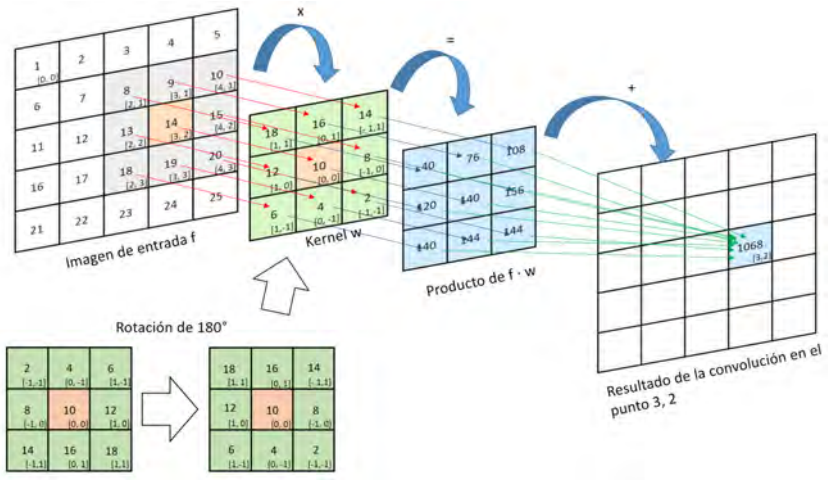

Figura A-2: Ejemplo de convolución espacial aplicada a una imagen de 5x5 con un kernel de 3x3. El resultado sería una imagen de 3x3 con bordes negros. Imagen recuperada en septiembre 2021 de: `https://bryanmed.github.io/conv2d/`

A.1. Filtros de convolución

Estos filtros, también conocidos como kernels o máscaras de convolución permiten obtener imágenes filtradas, que se utilizan para mover, rotar, definir bordes, líneas horizontales y verticales, así como suavizar la imagen para eliminar ruidos, entre muchas otras aplicaciones (poner citas). El tamaño de los kernels está dado como una matriz cuadrada de $n \times n$ (en algunos casos separable reduciendo la complejidad computacional (Podlozhnyuk [2007])), con diferentes tamaños dependiendo del grado de filtrado que se requiera, por lo que, a mayor tamaño de filtro, mayor grado de filtrado. Un paso importante, en la mayoría de estos filtros, es la normalización, es decir, la suma de todos los elementos debe ser igual a 1, ya que así no afectaría el resultado final.

Al aplicar los filtros de convolución debemos tomar en cuenta dos parámetros importantes: padding (rellenado) y stride (paso).

Padding Ocurre en situaciones en donde queramos que la imagen de salida tenga el mismo tamaño (o incluso mayor) que la de entrada obteniendo más información respecto a los bordes.

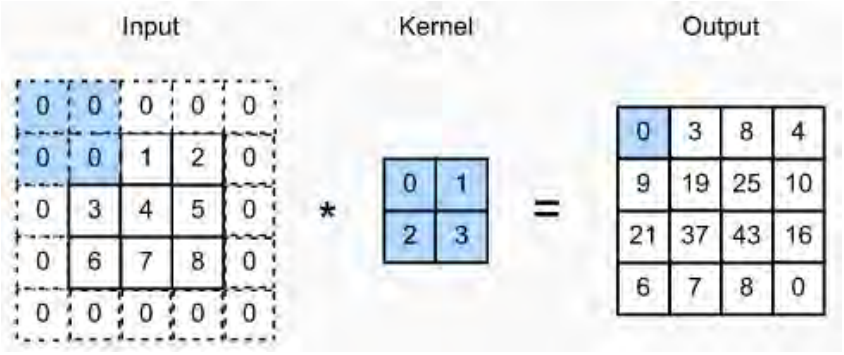

Figura A-3: Ejemplo de padding, con el objetivo de preservar el tamaño de entrada. Recuperado en septiembre 2021 de: https://medium.com/analytics-vidhya/convolution-padding-stride-and-pooling-in-cnn-13dc1f3ada26

Este problema se puede resolver al extender una o más columnas y filas en ambos lados de la imagen, utilizando o bien, el valor de cada pixel que corresponde en el sentido horizontal o vertical de la imagen original, o utilizando un valor fijo, como cero o uno (Figura A-3). Originalmente, con los valores por defecto, tenemos que para saber el tamaño de la imagen de salida utilizaríamos la ecuación A-5, al agregar padding p a la imagen de entrada, la dimensión de la imagen filtrada $n_{out} \times n_{out}$ estaría dada por la ecuación A-6.

$$n_{out} = (n_{in} + 2p - f + 1) \tag{A-6}$$

En caso de que queramos obtener la imagen de salida con un tamaño mayor a la imagen original debemos agregar un padding de $p = f - 1$, ya que sustituyendo éste p en A-6, tendríamos una salida equivalente a la suma de la dimensión de entrada y el valor de $f - 1$ (ecuación A-7).

$$\begin{aligned} n_{out} &= n_{in} + 2(f-1) - f + 1 \\ &= n_{in} + 2(f-1) - (f-1) \\ &= n_{in} + (f-1) \end{aligned} \tag{A-7}$$

Stride O pasos, son las unidades de desplazamiento que puede tener el filtro a lo largo de la imagen. Por defecto, se entiende que el desplazamiento del filtro corresponde a una unidad en

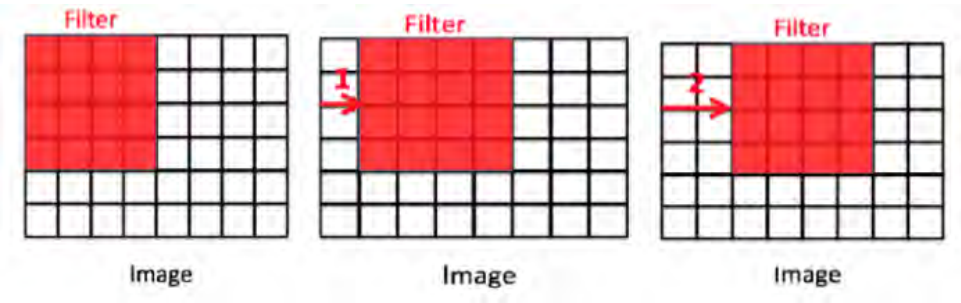

Figura A-4: Ejemplo de uso de stride a partir de un filtro de 4x4. Recuperado en septiembre 2021 de: https://medium.com/analytics-vidhya/convolution-padding-stride-and-pooling-in-cnn-13dc1f3ada26

horizontal y una vertical, sin embargo, se puede aumentar el número para reducir el tamaño de la imagen de salida y ver que tanta información relevante se mantiene (Figura A-4).

La ecuación A-8 permite conocer la dimensión de salida a través de las variables stride (s) y padding (p), para $s > 0$ y $p \geq 0$, y es la ecuación general de la cual se derivan las demás ecuaciones mencionadas.

$$n_{out} = \lfloor \frac{n_{in} + 2p - f}{s} \rfloor + 1 \tag{A-8}$$

A.1.1. Filtro Paso Bajo

Este tipo de filtro paso bajo (low pass) pasa las frecuencias bajas resultando en un filtrado de cambios abruptos como por ejemplo los bordes (Figura A-5)

Uno de los **filtros de convolución uniforme** , es el conocido como filtro promedio que permite suavizar la imagen dependiendo del tamaño de este. En expresión matricial, se puede expresar como la ecuación A-9 donde representa el promedio de los pixeles vecinos de una ventana que se va desplazando en toda la imagen.

$$\frac{1}{n^2}\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \dots & a_{nn} \end{bmatrix}, \; donde \; a_{ij} = 1 \; para \;\; i, j \in \{1, 2, ..., n\} \; y \;\; n > 1. \tag{A-9}$$

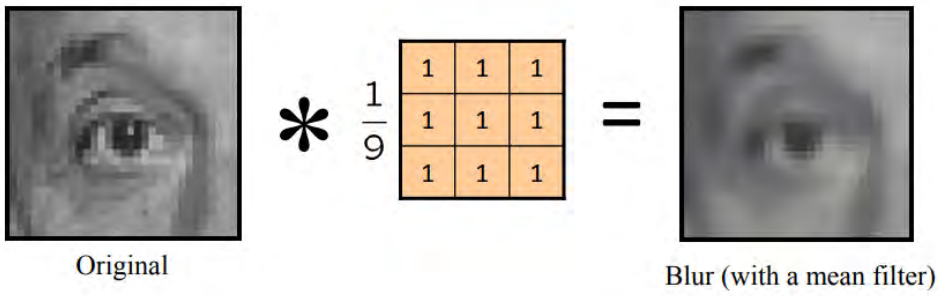

Figura A-5: Ejemplo de filtro paso bajo para suavizado de imagenes, a través del filtro promedio de 3x3. Recuperado en septiembre 2021 de: https://www.cs.cornell.edu/courses/cs6670/2011sp/lectures/lec02_filter.pdf

Supongamos que el tamaño del filtro es de 3×3, por lo que la representación del kernel sería como se muestra en la ecuación A-10 (Figura A-5)

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} \tag{A-10}$$

A.1.1.1. Filtro Gaussiano

Entre los **filtros de convolución no uniformes** tenemos el **filtro Gaussiano**. La función que representa la distribución en una dimensión está dada por la ecuación A-11. Dado que estamos trabajando con imágenes (espacio bidimensional), partiendo de la ecuación anterior, tendríamos la fórmula A-12, que al llevarla a un espacio discreto acotando la función en un rango $[-k, k]$, la dimensión de la matriz dependerá del valor de k a través de $n = 2k + 1$ y se expresa como la ecuación A-13

$$G(k) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{k^2}{2\sigma^2}} \tag{A-11}$$

$$G(k_1, k_2) = \frac{1}{2\pi\sigma^2} e^{-\frac{k_1^2 + k_2^2}{2\sigma^2}} \tag{A-12}$$

$$\begin{bmatrix} G(-k_1,-k_2) & \dots & G(-k_1,0) & \dots & G(-k_1,k_2) \\ \vdots & \dots & \vdots & \dots & \vdots \\ G(0,-k_2) & \dots & G(0,0) & \dots & G(0,k_2) \\ \vdots & \dots & \vdots & \dots & \vdots \\ G(k_1,-k_2) & \dots & G(k_1,0) & \dots & G(k_1,k_2) \end{bmatrix}, \; donde \;\; k_1,k_2 \in \{-k,-k+1,...,0,...,k-1,k\} \tag{A-13}$$

Nótese que la dimensión de la matriz Gaussiana siempre deberá ser impar, ya que se obtiene a través de la fórmula $2k+1$ que representa los números impares.

Por ejemplo, para un filtro gaussiano de tamaño 3×3, tendríamos que el valor de k sería 1 por $k = \frac{n-1}{2}$ siempre que n sea impar, de modo que $k_1, k_2 \in \{-1, 0, 1\}$ y la matriz quedaría de la siguiente manera (ecuación A-14)

$$\begin{bmatrix} G(-1,-1) & G(-1,0) & G(-1,1) \\ G(0,-1) & G(0,0) & G(0,1) \\ G(1,-1) & G(1,0) & G(1,1) \end{bmatrix} = \begin{bmatrix} 0.0585 & 0.0965 & 0.0585 \\ 0.0965 & 0.1592 & 0.0965 \\ 0.0585 & 0.0965 & 0.0585 \end{bmatrix} \tag{A-14}$$

Nótese que el valor del medio es el mayor valor de la matriz representando el pico de la función gaussiana. Además, si disminuimos el valor de la desviación estándar, el valor del medio incrementará haciéndolo un filtro más selectivo (Figura A-6)

A.1.1.2. Filtro Binomial

Otro **filtro no uniforme** que representa una aproximación del filtro gaussiano es el **filtro binomial**. La fórmula para obtener los coeficientes binomiales está dada por la ecuación A-15.

$$\binom{N}{k} = \frac{N!}{k!(N-k)!}, \;\; donde \;\; k \in \{0,1,...,N\} \tag{A-15}$$

Por lo que una aproximación al modelo Gaussiano sería a través de la ecuación A-11, donde $k = k - \frac{N}{2}$, la desviación estándar sería $\sigma = \frac{\sqrt{N}}{2}$ y el valor N representaría el número de

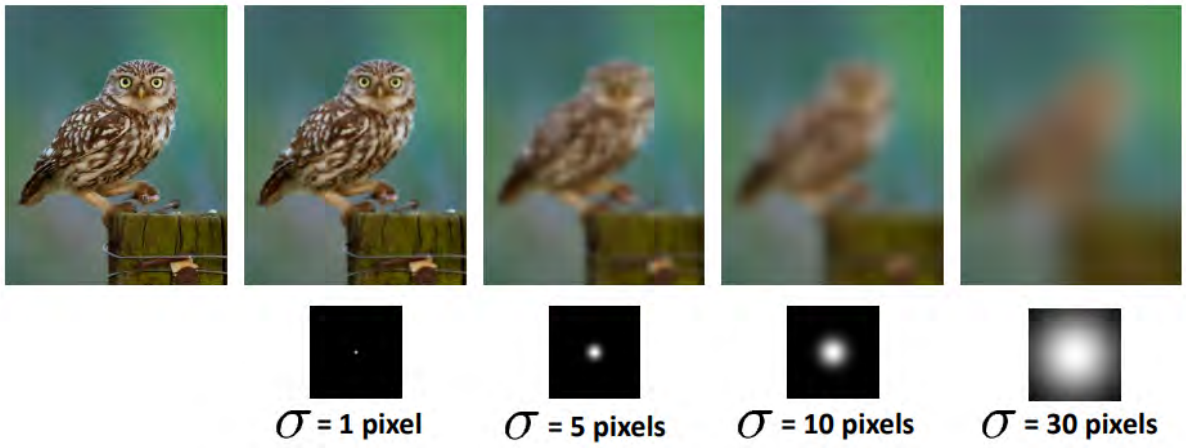

Figura A-6: Variaciones de σ en donde a mayor valor, mayor suavizado de la imagen. Recuperado en septiembre 2021 de: https://www.cs.cornell.edu/courses/cs6670/2011sp/lectures/lec02_filter.pdf

convoluciones en cascada que se necesitan para obtener el filtro binomial. Esto es, a partir del filtro low pass más simple que existe $[1, 1]$, se aplica convolución sobre el mismo N veces.

$$F_N = \underbrace{[1,1] * [1,1] * \ldots * [1,1]}_{\text{N veces}} \tag{A-16}$$

Este resultado corresponde a los que conocemos como el triángulo de pascal, en donde la suma del número de coeficientes será 2^N (Tabla A.1.1.2) y la unión de dos filtros unidimensionales de tamaño impar nos dará un filtro bidimensional binomial que representará una aproximación al modelo gaussiano. Por ejemplo, un filtro binomial de 3×3 estará dado por la multiplicación de $[1, 2, 1]$ y $[1, 2, 1]^T$, y será normalizado con la suma de sus componentes como se muestra en la ecuación A-17.

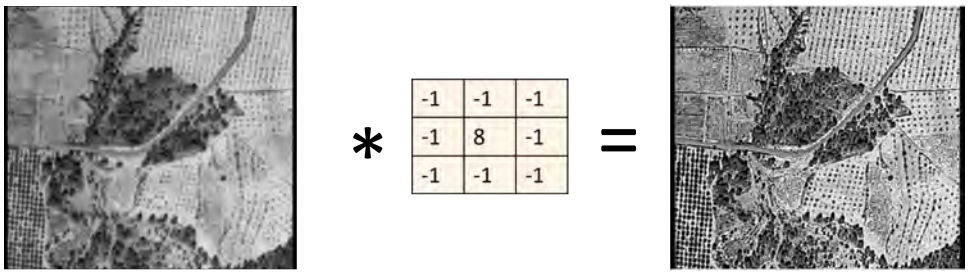

Figura A-7: Ejemplo de imagen filtrada con técnica paso alto, a través de filtro promedio menos identidad. Imagen recuperada en septiembre 2021 de: um.es/geograf/sigmur/teledet/tema06.pdf

Desviación estándar ($\sigma = \frac{\sqrt{N}}{2}$)	**Suma de coeficientes** (2^N)	**Coeficientes (Convolución N veces de [1,1])**	**N**	**Triángulo de pascal** **1**
$\frac{1}{2}$	$2^1 = 2$	**[1 1]**	1	1 1
$\frac{\sqrt{2}}{2}$	$2^2 = 4$	[0 1 1 0] * [1 1] = [1 2 1]	2	1 2 1
$\frac{\sqrt{3}}{2}$	$2^3 = 8$	[0 1 2 1 0] * [1 1] = [1 3 3 1]	3	1 3 3 1
1	$2^4 = 16$	[0 1 3 3 1] * [1 1] = [1 4 6 4 1]	4	1 4 6 4 1
⋮	⋮	⋮	⋮	⋮

Tabla A-1: Coeficientes del filtro binomial a través de convolución

$$F_2 = \frac{1}{4}\begin{bmatrix}1\\2\\1\end{bmatrix} \times \frac{1}{4}\begin{bmatrix}1 & 2 & 1\end{bmatrix} = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\2 & 4 & 2\\1 & 2 & 2\end{bmatrix} \tag{A-17}$$

A.1.2. Filtro Paso Alto

Los filtros paso alto o high pass, son el caso opuesto a low pass, permitiendo el flujo de frecuencias altas, es decir cambios abruptos en la imagen y eliminando las pequeñas variaciones. Esto se traduce a una imagen en donde que muestra los bordes (Figura A-7).

Una manera de obtener el filtro (h) es a través de la substracción del filtro identidad (i)

que al convolucionarlo con la imagen original da la misma imagen y un filtro en low pass (l) (promedio, gaussiano, binomial), como se muestra en el ejemplo de la ecuación utilizando un filtro promedio A-18.

$$
\begin{aligned}
h &= i - l \\
&= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \\
&= \frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}, \text{ podemos ignorar } \frac{1}{9} \text{ ya que la suma de los elementos es } 0 \\
h &= \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}
\end{aligned}
\tag{A-18}
$$

A.1.2.1. Gradiente de una imagen

Otra manera de obtener diferentes filtros paso alto, es a través de la operación matemática de la derivación con respecto a las filas y columnas de una imagen, es decir, el gradiente [Sucar y Gómez, 2011], cuya aproximación en términos discretos está dado por el cálculo de las diferencia (Figura A-8).

Para esto se tienen diferentes alternativas en representar la expresión discreta, por ejemplo, la ecuación A-19

$$
\begin{aligned}
\frac{\partial f[x,y]}{\partial x} &\approx f[x+1,y] - f[x,y] \\
\frac{\partial f[x,y]}{\partial y} &\approx f[x,y+1] - f[x,y]
\end{aligned}
\tag{A-19}
$$

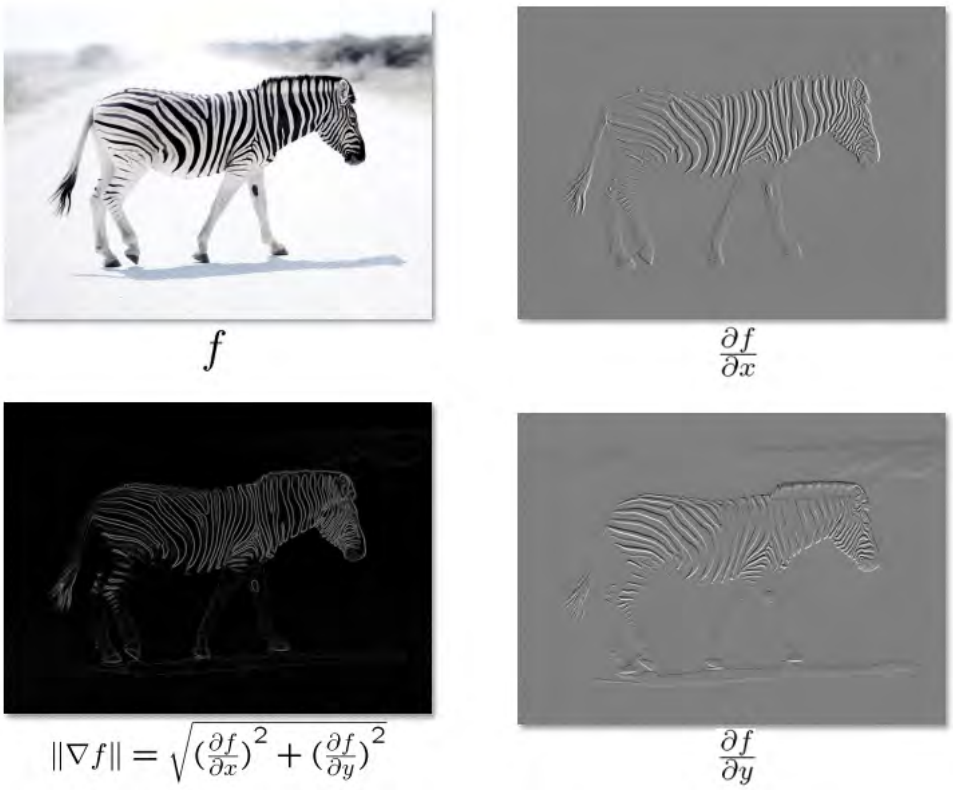

Figura A-8: Gradiente de la imagen. Recuperado en septiembre 2021 de: https://www.cs.cornell.edu/courses/cs6670/2011sp/lectures/lec02_filter.pdf

Por lo que podemos definir una imagen con base en su gradiente que define la dirección de los cambios y además la magnitud de los mismos que nos permitirá conocer el grado de sensibilidad en la detección de los bordes (ecuaciones A-20, A-21)

$$\nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right) \tag{A-20}$$

$$|\nabla f| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}\ [1] \tag{A-21}$$

Esto se puede traducir a máscaras tales como la figura (adjuntar imagen que incluya la derivada en x (-1,1) derivada en y (-1,1) y las derivadas diagonales) que corresponden los

[1] Para evitar costo computacional se suele dejar la ecuación en términos absolutos

conocidos operadores Roberts y Prewitt (Figura A-9).

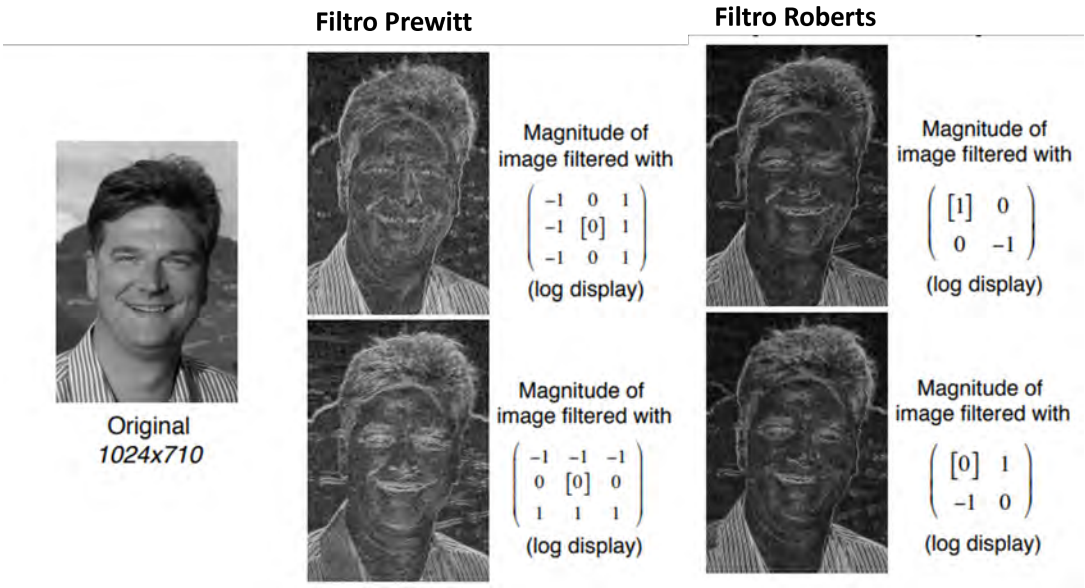

Figura A-9: Ejemplo de operación de filtros Prewitt y Roberts. Recuperada en septiembre 2021 de: https://web.stanford.edu/class/ee368/Handouts/Lectures/2019_Winter/11-EdgeDetection.pdf

Sin embargo, estos filtros son sensibles a ruidos como se muestra en la imagen (poner imagen) por lo que para eliminar el ruido es necesario utilizar además un filtro de suavizado, como lo utiliza el filtro Sobel.

A.1.2.2. Filtro Sobel

La ventaja de utilizar este filtro es que podemos tener cierta consistencia en la detección de estos bordes (Figura A-10). Se puede ver la presencia de la mezcla de un filtro lowpass y highpass ya que el filtro de sobel se puede obtener con la combinación de los coeficientes del filtro binomial (aproximación gaussiana), como por ejemplo [1 2 1] que nos aporta el suavizado y la primera derivada en los coeficientes del filtro binomial (deben corresponder al mismo tamaño) que en este caso sería [-1 0 1] [Nixon y Aguado, 2012], por lo que el gradiente en xy en y se presentan en las ecuaciones A-22 y A-23 y el filtro para obtener la magnitud aproximada estaría

dado por la suma de los gradientes (ecuación A-24)

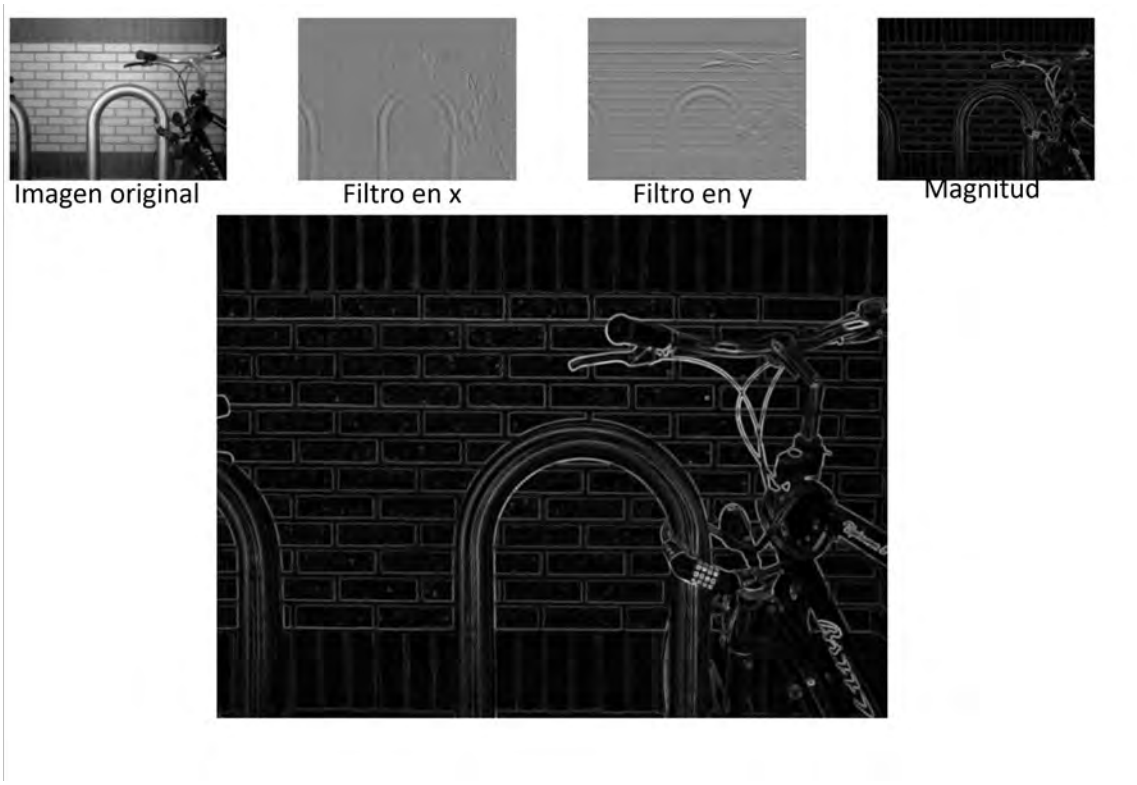

Figura A-10: Ejemplo de operación de filtro Sobel. Recuperada en septiembre 2021 de: https://www.cs.cornell.edu/courses/cs6670/2011sp/lectures/lec02_filter.pdf

$$G_x = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{A-22}$$

$$G_y = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \tag{A-23}$$

$$|\nabla f| = G_x + G_y = \begin{bmatrix} -2 & -2 & 0 \\ -2 & 0 & 2 \\ 0 & 2 & 2 \end{bmatrix} \tag{A-24}$$

Apéndice B

Redes Neuronales

Durante muchos años, se ha intentado abstraer el funcionamiento del cerebro humano de la manera más simple posible para poder obtener un modelo que nos permita simular la capacidad que tenemos de razonamiento. Así, la transferencia de información que poseemos se debe en parte a pequeñas células interconectadas llamadas neuronas, que en términos computacionales fue recreada artificialmente por primera vez en 1958 [Rosenblatt, 1958], tomando el nombre de perceptrón (Figura B-1).

Este modelo simple de una sola neurona, básicamente imitaba el comportamiento de un modelo de regresión lineal de predicción binaria, sólo que las constantes de los parámetros que multiplican a las entradas son llamados pesos (*weights*), y se iban ajustando hasta obtener el resultado esperado. Este primer concepto fue capaz de simular el funcionamiento de una compuerta lógica AND o de una OR, pero no de una XOR, dejando un largo período de incertidumbre sobre el futuro de este modelo.

Muchos años después, se propone agregar más neuronas (*Multilayer Perceptrons o MLP*), que permite resolver el problema XOR permitiendo abordar problemas más complejos. En este modelo la regresión lineal se filtra a través de una función llamada función de activación que obtiene un resultado no lineal acercándose a una solución más real.

Las funciones de activación más utilizadas se muestran en la figura B-2.

Una de las formas de obtener una salida esperada sin tener que estar ajustando estos pesos

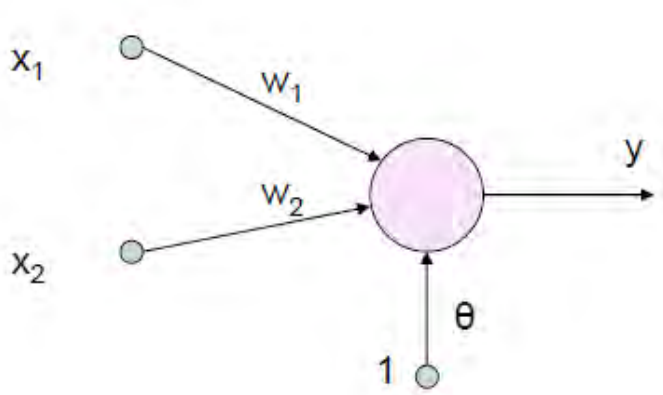

Figura B-1: Representación conceptual del perceptrón. Recuperada en octubre 2021 de: http://tecnologia2014ohanna.blogspot.com/2014/03/el-perceptron-simple-fue-introducido.html

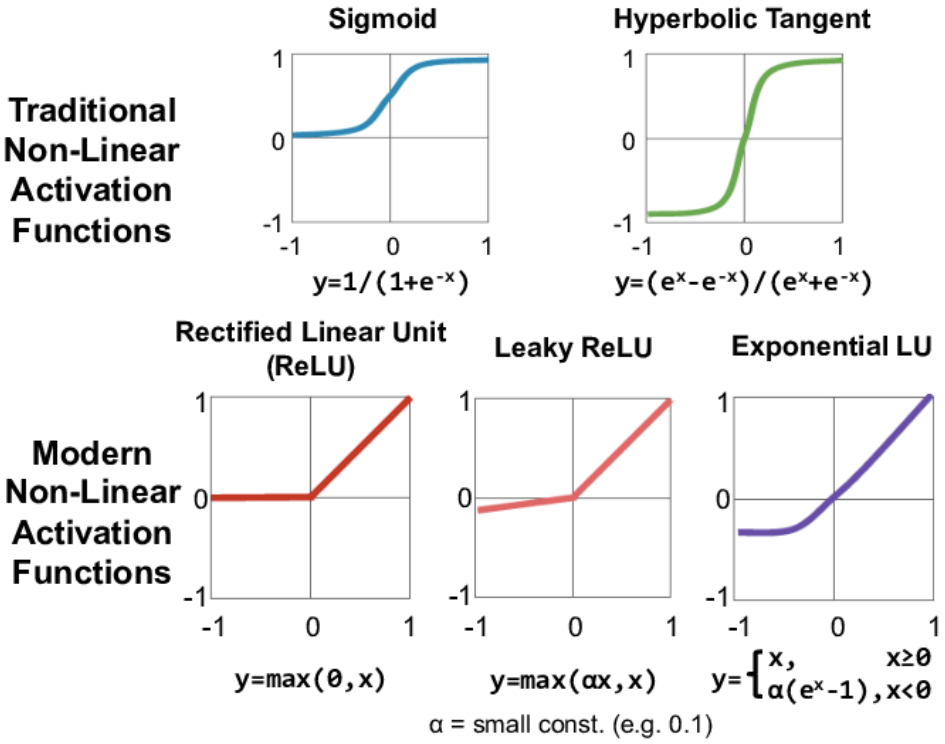

Figura B-2: Funciones de activación comúnmente utilizadas en modelos de redes neuronales. Ejemplo extraido en octubre 2021 de: https://ignaciogavilan.com/catalogo-de-componentes-de-redes-neuronales-ii-funciones-de-activacion/

manualmente, es utilizando un algoritmo llamado *Backpropagation* [Rumelhart *et al.*, 1986] que con base en el error entre el resultado obtenido y el esperado, utilizando alguna función de coste y una función que permita minimizar éste costo (comúnmente conocida como función del descenso de gradiente), se van ajustando estos pesos de forma iterativa con el objetivo de optimizar el error obtenido, y que el resultado se acerque cada vez más al esperado (etapa conocida como entrenamiento)

B.1. Arquitectura de una Red Neuronal

Actualmente existen diversos modelos de redes neuronales cuya estructura o distribución de las neuronas varía en cierto grado, sin embargo, en términos generales se pueden dividir en 3 secciones o bloques importantes.

- Capa de entrada, donde el número de neuronas depende de la cantidad de características con las que la red podrá clasificar (pixeles de una imagen, bordes, entre otros)
- Capa oculta, donde se encuentra un conjunto de neuronas interconectadas cuyos pesos serán modificados en la etapa de entrenamiento de modo que permitan obtener un resultado aproximado al real
- Capa de salida, compuesta por neuronas que muestran el resultado entre 0 y 1, de modo que suelen ocuparse notación binaria o

B.2. Modelos de Redes Neuronales

Actualmente existen diferentes modelos de red con el objetivo de abordar problemas de clasificación dependiendo de la aplicación y sus datos. Una de las diferencias entre los modelos de red se encuentra en la etapa de la extracción de características, esto es, para situaciones en donde se requiera clasificar con base a un conjunto de características obtenidas, se utilizarán modelos de red compactos cuyo objetivo principal es únicamente la clasificación, sin embargo, existen modelos en donde se incluye la extracción de características dentro del mismo (*deep*

learning), cuya desventaja radica en que se requiere un gran número de muestras para poder obtener una clasificación satisfactoria.

Los modelos de redes neuronales más reconocidos actualmente [Goled, 2020] son los siguientes:

- ***Multilayer Perceptrons* (*MLP*):** Es el precursor de la primera neurona (perceptrón) que se creó para simular una compuerta lógica AND o una OR, cuya solución para resolver el problema de la compuerta XOR fue utilizar varias neuronas (perceptrones) interconectadas. Este tipo de modelo utiliza la arquitectura básica (capa de entrada, oculta y salida), la técnica de *backpropagation* para entrenamiento. MLP se suele utilizar en proyectos que requieran resolver problemas de predicción en clasificación y regresión.

- ***Convolution Neural Network* (*CNN*):** Este tipo de modelo está relacionado con imágenes debido a que posee una capa encargada de aplicar filtros de convolución para extraer características en estas imágenes, por lo que es muy utilizado en clasificación de imágenes, detección de objetos y segmentación.

- ***Recurrent Neural Networks* (*RNN*):** En este tipo de modelos, la salida de un estado actual será la entrada al paso siguiente en donde las capas ocultas se encargarán de entender el proceso, creando un sistema que tenga "memoria" y que permita utilizar esa información anterior con el objetivo de aprender de manera automática y dinámica. Este tipo de red es muy utilizado en el aprendizaje del lenguaje escrito y hablado.

- ***Deep Belief Network* (*DBN*):** Este tipo de red utiliza probabilidad y aprendizaje no supervisado[1]. Los algoritmos codiciosos (*greedy algorithms*) se utilizan para entrenar cada capa a la vez, esto es, se procesará versiones diferentes en los datos de entrada para cada capa y la salida de estas será la entrada a la capas siguientes. Podemos encontrar *DBN* principalmente en aplicaciones de reconocimiento de imagenes y videos.

[1]En el aprendizaje no supervisado las muestras no son etiquetadas, por lo que el sistema tendrá que detectar patrones y agrupará estas muestras de forma automática

- ***Restricted Boltzmann Machine* (*RBM*):** Es un modelo que aprende a partir de la distribución de probabilidad del conjunto de entrada a través de algoritmos generativos y no deterministas, por lo que son muy utilizados en problemas de reducción en la dimensionalidad de los datos y sistemas de recomendación.

Apéndice C

Cuestionario

Se encuentra en la siguiente página

CUESTIONARIO DEL SISTEMA DE INTERACCIÓN

Edad: _____ Sexo: _____ Ocupación: _____

Marque con una X las opciones que correspondan a la pregunta

Uso de Tecnologías				
Utilizo mi celular para	Fotos	Redes sociales	Compras	Videojuegos
He utilizado alguno de estos dispositivos electrónicos	Laptop	Computadora de escritorio	Consola de videojuegos	Lentes de Realidad Virtual

Marque con una X la puntuación que considere más acorde con el servicio recibido (1 completamente en desacuerdo, 5 Completamente de acuerdo)

Evaluación del sistema general	1	2	3	4	5
La actividad por realizar se entendió fácilmente					
El entorno me resultó agradable y cómodo					
El entorno me distraía de la actividad a realizar					
Sentí mareos al estar en el mundo virtual					

Interacción con gesto de pinza	1	2	3	4	5
Encuentro esta interacción fácil de usar					
Considero que fue fácil de aprender la forma de interactuar					
Me resultó cómoda la forma de interactuar					
Siento mayor conexión y contacto con el objeto virtual					
Sentí ansiedad al utilizar este modo de interacción					
Me sentí mal por no completar rápido la actividad					
Luego de un tiempo de interactuar sentí incomodidad en la mano					
Comentarios:					

Interacción con gesto natural sin tacto	1	2	3	4	5
Encuentro esta interacción fácil de usar					
Considero que fue fácil de aprender la forma de interactuar					
Me resultó cómoda la forma de interactuar					
Siento mayor conexión y contacto con el objeto virtual					
Sentí ansiedad al utilizar este modo de interacción					
Me sentí mal por no completar rápido la actividad					
Luego de un tiempo de interactuar sentí incomodidad en la mano					
Comentarios:					

Interacción con gesto natural con tacto (lego físico)	1	2	3	4	5
Encuentro esta interacción fácil de usar					
Considero que fue fácil de aprender la forma de interactuar					
Me resultó cómoda la forma de interactuar					
Siento mayor conexión y contacto con el objeto virtual					
Sentí ansiedad al utilizar este modo de interacción					
Me sentí mal por no completar rápido la actividad					
Luego de un tiempo de interactuar sentí incomodidad en la mano					
Comentarios:					

¿Cuál fue la interacción que más le gustó?		
Interacción con gesto de pinza	Interacción natural sin tacto	Interacción natural con tacto

No completar este apartado:

Tiempos y errores de interacción (Stage 1)					
Id de Legos	1	2	3	4	5
Tiempo en completar la tarea					
Número de errores al completar la tarea					
Tiempo en reparar el error					
Comentarios:					

Tiempos y errores de interacción (Stage 2)					
Id de Legos	1	2	3	4	5
Tiempo en completar la tarea					
Número de errores al completar la tarea					
Tiempo en reparar el error					
Comentarios:					

Tiempos y errores de interacción (Stage 3)					
Id de Legos	1	2	3	4	5
Tiempo en completar la tarea					
Número de errores al completar la tarea					
Tiempo en reparar el error					
Comentarios:					

Referencias

Real Academia Española. *Diccionario de la lengua española*, 23a ed., [versión 23.4 en línea]. <https://dle.rae.es> (2014). [Online; accessed 19-October-2021]

Adams, J., Qiu, Y., Xu, Y., y Schnable, J.C. Plant segmentation by supervised machine learning methods. *The Plant Phenome Journal* **3**(1):e20001 (2020)

Aïm, F., Lonjon, G., Hannouche, D., y Nizard, R. Effectiveness of virtual reality training in orthopaedic surgery. *Arthroscopy: the journal of arthroscopic & related surgery* **32**(1):224–232 (2016)

Arganda-Carreras, I., Kaynig, V., Rueden, C., Eliceiri, K.W., Schindelin, J., Cardona, A., y Sebastian Seung, H. Trainable Weka Segmentation: a machine learning tool for microscopy pixel classification. *Bioinformatics* **33**(15):2424–2426 (2017). https://academic.oup.com/bioinformatics/article-pdf/33/15/2424/25157856/btx180.pdf

Athira, M. y Khan, D.M. Recent trends on object detection and image classification: a review. En *2020 International Conference on Computational Performance Evaluation (ComPE)*, págs. 427–435. IEEE (2020)

Bamodu, O. y Ye, X.M. Virtual reality and virtual reality system components. En *Advanced materials research*, tomo 765, págs. 1169–1172. Trans Tech Publ (2013)

Baritz, M. y Cotoros, D. MODELING VISUAL TRAINING FOR YOUNG PEOPLE VISION EYECARE. *eLearning & Software for Education* (2011)

Basu, M. Gaussian-based edge-detection methods-a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* **32**(3):252–260 (2002)

Bhagat, K.K., Liou, W.K., y Chang, C.Y. A cost-effective interactive 3D virtual reality system applied to military live firing training. *Virtual Reality* **20**(2):127–140 (2016)

Bowman, D.A., McMahan, R.P., y Ragan, E.D. Questioning naturalism in 3D user interfaces. *Communications of the ACM* **55**(9):78–88 (2012a)

Bowman, D.A., McMahan, R.P., y Ragan, E.D. Questioning naturalism in 3D user interfaces. *Communications of the ACM* **55**(9):78–88 (2012b)

Bradski, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* (2000)

Brooke, J. *et al.* SUS-A quick and dirty usability scale. *Usability evaluation in industry* **189**(194):4–7 (1996)

Canny, J. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence* (6):679–698 (1986)

Chen, S.S.C. y Duh, H. Interface of mixed reality: from the past to the future. *CCF Transactions on Pervasive Computing and Interaction* **1**(1):69–87 (2019)

Chen, W., Shi, Y.Q., y Xuan, G. Identifying Computer Graphics using HSV Color Model and Statistical Moments of Characteristic Functions. En *2007 IEEE International Conference on Multimedia and Expo*, págs. 1123–1126 (2007)

Chollet, F. *et al.* Keras (2015)

Clark, A. y Moodley, D. A system for a hand gesture-manipulated virtual reality environment. En *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists*, págs. 1–10 (2016)

Cole, L., Austin, D., Cole, L. *et al.* Visual object recognition using template matching. En *Australian conference on robotics and automation* (2004)

Community, B.O. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam (2018)

Cruz-Neira, C., Sandin, D.J., DeFanti, T.A., Kenyon, R.V., y Hart, J.C. The CAVE: audio visual experience automatic virtual environment. *Communications of the ACM* **35**(6):64–73 (1992)

Daily, M., Sarfaty, R., Jerald, J., McInnes, D., y Tinker, P. The CABANA: A Reconfigurable Spatially Immersive Display. En *Projection Technology Workshop*, págs. 123–132 (1999)

de Back, T.T., Tinga, A.M., Nguyen, P., y Louwerse, M.M. Benefits of immersive collaborative learning in CAVE-based virtual reality. *International Journal of Educational Technology in Higher Education* **17**(1):1–18 (2020)

Demmel, J.W. *Applied numerical linear algebra*. SIAM (1997)

Dubey, S. y Gupta, Y.K. Computational Comparison of Various Existing Edge Detection Techniques for Medical Images **16**:185–193 (2018)

Ellis, S. Where Are All the Head Mounted Displays? Retrieved April 14, 2015 (2014)

Epic Games. Unreal Engine (????)

Facebook Technologies, L. Oculus Device Specifications. Sitio web: `https://developer.oculus.com/resources/oculus-device-specs/` (2021). [Online; accessed 2021]

Feick, M., Bateman, S., Tang, A., Miede, A., y Marquardt, N. TanGi: Tangible Proxies for Embodied Object Exploration and Manipulation in Virtual Reality. En *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, págs. 195–206. IEEE (2020)

Freina, L. y Ott, M. A literature review on immersive virtual reality in education: state of the art and perspectives. En *The international scientific conference elearning and software for education*, tomo 1, págs. 10–1007 (2015)

García, G.B., Suarez, O.D., Aranda, J.L.E., Tercero, J.S., Gracia, I.S., y Enano, N.V. *Learning image processing with OpenCV*. Packt Publishing Birmingham (2015)

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F., y Medina-Carnicer, R. Generation of fiducial marker dictionaries using Mixed Integer Linear Programming. *Pattern Recognition* **51** (2015)

Gilson, S.J., Fitzgibbon, A.W., y Glennerster, A. An automated calibration method for non-see-through head mounted displays. *Journal of Neuroscience Methods* **199**(2):328–335 (2011)

Girshick, R., Donahue, J., Darrell, T., y Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 580–587 (2014)

Goled, S. Top 5 Neural Network Models For Deep Learning and Their Applications (2020)

Gonzalez, R.C. y Woods, R.E. *4TH EDITION Digital image processing* (2018)

Grana, C., Borghesani, D., y Cucchiara, R. Optimized block-based connected components labeling with decision trees. *IEEE Transactions on Image Processing* **19**(6):1596–1609 (2010)

Guerra, J.P., Pinto, M.M., y Beato, C. Virtual reality-shows a new vision for tourism and heritage. *European Scientific Journal* (2015)

Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., del Río, J.F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., y Oliphant, T.E. Array programming with NumPy. *Nature* **585**(7825):357–362 (2020)

Heilig, M.L. El cine del futuro: The cinema of the future. *Presence: Teleoperators & Virtual Environments* **1**(3):279–294 (1992)

Hettiarachchi, A. y Wigdor, D. Annexing reality: Enabling opportunistic use of everyday objects as tangible proxies in augmented reality. *Conference on Human Factors in Computing Systems - Proceedings* págs. 1957–1967 (2016)

Howard, M.C. A meta-analysis and systematic literature review of virtual reality rehabilitation programs. *Computers in Human Behavior* **70**:317–327 (2017)

Huang, W., Kim, K.Y., Yang, Y., y Kim, Y.S. Automatic shadow removal by illuminance in HSV color space. *Comput. Sci. Inf. Technol* **3**(3):70–75 (2015)

Insko, B.E. *Passive haptics significantly enhances virtual environments.* The University of North Carolina at Chapel Hill (2001)

Jarvis, R.A. On the identification of the convex hull of a finite set of points in the plane. *Information processing letters* **2**(1):18–21 (1973)

Jayanthi, M. y Shashikumar, D. Survey on Agriculture Image Segmentation Techniques. *Asian Journal of Applied Science and Technology (AJAST)* **1**(8):143–146 (2017)

Jerald, J. *The VR book: Human-centered design for virtual reality.* Morgan & Claypool (2015)

Jiang, H., Weng, D., Zhang, Z., Bao, Y., Jia, Y., y Nie, M. HiKeyb: High-Efficiency Mixed Reality System for Text Entry. *Adjunct Proceedings - 2018 IEEE International Symposium on Mixed and Augmented Reality, ISMAR-Adjunct 2018* (October):132–137 (2018)

Jojic, N., Brumitt, B., Meyers, B., Harris, S., y Huang, T. Detection and estimation of pointing gestures in dense disparity maps. En *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*, págs. 468–475. IEEE (2000)

Jong, S.M.d. y Meer, F.v.d. *Remote sensing image analysis : including the spatial domain.* Remote sensing and digital image processing: v. 5. Kluwer Academic (2004)

Kaganami, H.G. y Beiji, Z. Region-based segmentation versus edge detection. En *2009 Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, págs. 1217–1221. IEEE (2009)

Kang, H.J., Shin, J.H., y Ponto, K. A Comparative Analysis of 3D User Interaction: How to Move Virtual Objects in Mixed Reality. *Proceedings - 2020 IEEE Conference on Virtual Reality and 3D User Interfaces, VR 2020* págs. 275–284 (2020a)

Kang, H.J., Shin, J.h., y Ponto, K. A comparative analysis of 3d user interaction: How to move virtual objects in mixed reality. En *2020 IEEE conference on virtual reality and 3D user interfaces (VR)*, págs. 275–284. IEEE (2020b)

Kim, Y.M., Rhiu, I., y Yun, M.H. A systematic review of a virtual reality system from the perspective of user experience. *International Journal of Human–Computer Interaction* **36**(10):893–910 (2020)

Kiousis, S. Interactivity: a concept explication. *New media & society* **4**(3):355–383 (2002)

Le Cun, Y., Bottou, L., y Bengio, Y. Reading checks with multilayer graph transformer networks. En *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, tomo 1, págs. 151–154. IEEE (1997)

LI, Y., HUANG, J., TIAN, F., WANG, H.A., y DAI, G.Z. Gesture interaction in virtual reality. *Virtual Reality & Intelligent Hardware* **1**(1):84–112 (2019)

Lindeman, R.W. *Bimanual interaction, passive-haptic feedback, 3D widget representation, and simulated surface constraints for interaction in immersive virtual environments*. The George Washington University (1999)

Logitech, . C920 HD PRO WEBCAM. Sitio web: https://www.logitech.com/en-hk/products/webcams/c920-pro-hd-webcam.960-001062.html#specs (2021). [Online; accessed 2021]

Mahalle, A. y Shah, A. FPGA Implementation of Gradient Based Edge Detection Algorithms (2017)

Maheswari, D. y Radha, V. Enhanced hybrid compound image compression algorithm combining block and layer-based segmentation. *The International Journal of Multimedia & Its Applications* **3**(4):119 (2011)

Marco-Antonio, G.H.E. Segmentación de obstáculos y detección del área transitable de un robot móvil usando cámara de profundidad **8**(1):1 – 21 (2019)

Marr, D. y Nishihara, H.K. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London. Series B. Biological Sciences* **200**(1140):269–294 (1978)

Martínez, F.P. *et al.* Presente y Futuro de la Tecnología de la Realidad Virtual. *Creatividad y sociedad* **16**:1–39 (2011)

McMahan, R., Ragan, E., Bowman, D., Tang, F., y Lai, C. Fifa: The framework for interaction fidelity analysis. *University of Texas at Dallas, Dept. of Computer Science, Technical UTDCS-06-15* (2015)

Mihelj, M., Novak, D., y Beguš, S. Virtual reality technology and applications (2014)

Milgram, P. y Kishimo, F. A taxonomy of mixed reality. *IEICE Transactions on Information and Systems* **77**(12):1321–1329 (1994)

Milgram, P., Colquhoun, H. *et al.* A taxonomy of real and virtual world display integration. *Mixed reality: Merging real and virtual worlds* **1**(1999):1–26 (1999)

Muender, T., Reinschluessel, A.V., Drewes, S., Wenig, D., Döring, T., y Malaka, R. Does It Feel Real? Using Tangibles with Different Fidelities to Build and Explore Scenes in Virtual Reality. En *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, págs. 1–12 (2019)

Muhanna, M.A. Virtual reality and the CAVE: Taxonomy, interaction challenges and research directions. *Journal of King Saud University - Computer and Information Sciences* **27**(3):344–361 (2015)

Nishad, P. Various colour spaces and colour space conversion. *Journal of Global Research in Computer Science* **4**(1):44–48 (2013)

Nixon, M.S. y Aguado, A.S. Chapter 4 - Low-level feature extraction (including edge detection). En M.S. Nixon y A.S. Aguado (editores), *Feature Extraction & Image Processing for Computer Vision (Third Edition)*, third edition edición, págs. 137–216. Academic Press, Oxford (2012)

OpenCV, . Documentación de la función convexHull. Sitio web: https://docs.opencv.org/4.x/d3/dc0/group__imgproc__shape.html#ga014b28e56cb8854c0de4a211cb2be656 (2021). [Online; accessed 2021]

Pekrun, R., Goetz, T., Frenzel, A.C., Barchfeld, P., y Perry, R.P. Measuring emotions in students' learning and performance: The Achievement Emotions Questionnaire (AEQ). *Contemporary educational psychology* **36**(1):36–48 (2011)

Perret, J. y Vander Poorten, E. Touching virtual reality: a review of haptic gloves. En *ACTUATOR 2018; 16th International Conference on New Actuators*, págs. 1–5. VDE (2018)

Podlozhnyuk, V. Image convolution with CUDA. *NVIDIA Corporation white paper, June* **2097**(3) (2007)

Ramesh, K., Kumar, G.K., Swapna, K., Datta, D., y Rajest, S.S. A Review of Medical Image Segmentation Algorithms. *EAI Endorsed Transactions on Pervasive Health and Technology* (2021)

Rebelo, F., Noriega, P., Duarte, E., y Soares, M. Using virtual reality to assess user experience. *Human Factors* **54**(6):964–982 (2012)

Redmon, J., Divvala, S., Girshick, R., y Farhadi, A. You only look once: Unified, real-time object detection. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 779–788 (2016)

Romero-Ramirez, F., Muñoz-Salinas, R., y Medina-Carnicer, R. Speeded Up Detection of Squared Fiducial Markers. *Image and Vision Computing* **76** (2018)

Rosenblatt, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review* **65**(6):386 (1958)

Rumelhart, D.E., Hinton, G.E., y Williams, R.J. Learning representations by back-propagating errors. *nature* **323**(6088):533–536 (1986)

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., y Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge (2015). 1409.0575

Satava, R.M. Virtual reality, telesurgery, and the new world order of medicine. *Journal of Image Guided Surgery* **1**(1):12–16 (1995)

Scipy, . Documentación de la función binaryFillHoles. Sitio web: https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.ndimage.morphology.binary_fill_holes.html (2021). [Online; accessed 2021]

Shaik, K.B., Ganesan, P., Kalist, V., Sathish, B., y Jenitha, J.M.M. Comparative study of skin color detection and segmentation in HSV and YCbCr color space. *Procedia Computer Science* **57**:41–48 (2015)

Shorten, C. y Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *Journal of Big Data* **6**(1):1–48 (2019)

Skarbez, R., Smith, M., y Whitton, M.C. Revisiting Milgram and Kishino's Reality-Virtuality Continuum. *Frontiers in Virtual Reality* **2**:27 (2021)

Slater, M., Spanlang, B., y Corominas, D. Simulating virtual environments within virtual environments as the basis for a psychophysics of presence. *ACM Transactions on Graphics (TOG)* **29**(4):1–9 (2010)

Smith, A.R. Color gamut transform pairs. *ACM Siggraph Computer Graphics* **12**(3):12–19 (1978)

Sobel, I. y Feldman, G. A 3×3 isotropic gradient operator for image processing. *Pattern Classification and Scene Analysis* págs. 271–272 (1973)

Spence, C. Scenting Entertainment: Virtual Reality Storytelling, Theme Park Rides, Gambling, and Video-Gaming. *i-Perception* **12**(4):20416695211034538 (2021)

Stanney, K. Handbook of virtual environments. Lawrence Earlbaum (2001)

Su, C.H., Chiu, H.S., y Hsieh, T.M. An efficient image retrieval based on HSV color space. En *2011 International Conference on Electrical and Control Engineering*, págs. 5746–5749. IEEE (2011)

Sucar, L.E. y Gómez, G. *Vision Computacional* (2011)

Surkutlawar, S. y Kulkarni, R.K. Shadow suppression using RGB and HSV color space in moving object detection. *International Journal of Advanced Computer Science and Applications* **4**(1) (2013)

Sutherland, I.E. A head-mounted three dimensional display. En *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, págs. 757–764 (1968)

Suzuki, S. *et al.* Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing* **30**(1):32–46 (1985)

Szegedy, C., Ioffe, S., Vanhoucke, V., y Alemi, A.A. Inception-v4, inception-resnet and the impact of residual connections on learning. En *Thirty-first AAAI conference on artificial intelligence* (2017)

Szeliski, R. *Computer vision* (2004)

Taheri, S.M., Matsushita, K., Sasaki, M. *et al.* Virtual reality driving simulation for measuring driver behavior and characteristics. *Journal of transportation technologies* **7**(02):123 (2017)

Tcha-Tokey, K., Loup-Escande, E., Christmann, O., y Richir, S. A questionnaire to measure the user eXperience in immersive virtual environments. *ACM International Conference Proceeding Series* (December) (2016)

Valentino, K., Christian, K., y Joelianto, E. Virtual reality flight simulator. *Internetworking Indonesia Journal* **9**(1):21–25 (2017)

Venkatesh, V., Morris, M.G., Davis, G.B., y Davis, F.D. User acceptance of information technology: Toward a unified view. *MIS quarterly* págs. 425–478 (2003)

Villamarín, D. Estado del Arte, Herramientas y Aplicaciones para Transformaciones geométricas 3D. En *Congreso de Ciencia y Tecnología ESPE*, tomo 10, págs. 226–231 (2015)

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., y SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* **17**:261–272 (2020)

Wang, J. y Lindeman, R. Coordinated hybrid virtual environments: Seamless interaction contexts for effective virtual reality. *Computers & Graphics* **48**:71–83 (2015)

Xiao, D. y Ohya, J. Contrast enhancement of color images based on wavelet transform and

human visual system. En *Proceedings of the IASTED International Conference on Graphics and Visualization in Engineering*, págs. 58–63. Citeseer (2007)

Xie, S., Girshick, R., Dollár, P., Tu, Z., y He, K. Aggregated residual transformations for deep neural networks. En *Proceedings of the IEEE conference on computer vision and pattern recognition*, págs. 1492–1500 (2017)

Xue, L., Parker, C.J., y Hart, C.A. How to engage fashion retail with virtual reality: A consumer perspective. En *Augmented Reality and Virtual Reality*, págs. 23–35. Springer, Cham (2020)

Zervos, H. 14-1: Augmented Reality and Virtual Reality Smart Eyewear: Forecasts for the Next Decade. En *SID Symposium Digest of Technical Papers*, tomo 47, págs. 150–152. Wiley Online Library (2016)