ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

TELEOPERACIÓN DE UN HEXACÓPTERO CON REALIMENTACIÓN DE FUERZA MEDIANTE UN SIMULADOR EN AMBIENTES SEMIESTRUCTURADOS.

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y CONTROL

GUADALUPE OYAQUE NATASHA MERCEDES
natasha.guadalupe.ng@gmail.com

DIRECTOR: Dr. Danilo Geovanny Chávez García
danilo.chavez@epn.edu.ec

Quito, Julio 2016

DECLARACIÓN

Yo, Natasha Mercedes Guadalupe Oyaque, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

_____

Natasha Mercedes Guadalupe Oyaque

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Natasha Mercedes Guadalupe Oyaque, bajo mi supervisión.

_____

Dr. Danilo Chávez

DIRECTOR DEL PROYECTO

AGRADECIMIENTO

A Dios por darme la fuerza para alcanzar esta meta y la bendición de mantener a mi familia unida. A mis padres Fernando y Nancy por la paciencia, el amor, el apoyo y la entrega. A mis hermanas Mónica y María Fernanda por la alegría de tenerles siempre cerca.

Agradezco a mi abuelita Victoriana porque le dio a mi corazón la paz que necesitaba.

A Fabián por caminar a lado mío siempre e iluminar mi vida.

Al Doctor Danilo Chávez por permitirme trabajar con él, brindándome siempre el apoyo necesario.

También a mis amigos y familiares por su atención y aliento.

Natasha Mercedes Guadalupe Oyaque

DEDICATORIA

A mi amada familia Papi, Mami, Mony y Mary que son inspiración y ejemplo para intentar siempre ser una mejor persona.

Natasha Mercedes Guadalupe Oyaque

CONTENIDO

RESUMEN

El presente proyecto simula el vuelo de hexacópteros en ambientes internos. A través del desarrollo, diseño, modelado e implementación de entornos, se replican ambientes semiestructurados con obstáculos estáticos, estos ambientes son una casa, una iglesia y un supermercado.
Cada uno de ellos presenta diferentes obstáculos, desplegando distintos niveles de dificultad en el manejo del simulador de vuelo, así como cinco tareas por cumplir, que pueden ser seleccionadas por el usuario, o si se desea también se puede volar sin ningún objetivo por alcanzar.

La aplicación brinda la posibilidad de elegir entre dos modelos de hexacópteros, logrando así, ser más útil e interactivo.
El simulador incluye la opción de realimentación de fuerza, la cual se logra con el Joystick Microsoft® SideWinder® Force Feedback 2, permitiendo advertir al usuario sobre posibles colisiones, y entregarle la sensación de presencia al operador. La fuerza realimentada aplicada está limitada por el dispositivo háptico.

En este trabajo, se realiza la teleoperación de dos modelos de hexacópteros con realimentación de fuerza mediante un simulador, enfocada a aplicaciones de escenarios tridimensionales en ambientes internos semiestructurados, reproduciendo el ambiente y las variables (rasgos, apariencia, características, contexto) de un sistema real, en ese sentido la implementación del simulador de vuelo, constituye una técnica de bajo costo que permite ofrecer varios escenarios posibles de una situación y tener errores que no tengan efectos sobre el mundo real.

PRESENTACIÓN

El presente documento consta de cinco capítulos que detallan el desarrollo del simulador de vuelo, y se distribuyen de la siguiente manera:

El primer capítulo contiene el marco teórico del proyecto, donde se encuentran los conceptos y definiciones necesarios que permiten comprender el proceso para la implementación del simulador de vuelo.

En el segundo capítulo se presenta el modelo dinámico de los hexacópteros, sus especificaciones y configuraciones, en las que se basa el diseño y la programación del simulador de vuelo.

En el capítulo tres se describe el conjunto de aplicaciones que permitieron la creación y manipulación de los modelos 3D, tanto de los escenarios como de las aeronaves. Se explica el funcionamiento del motor gráfico Unity 3D, la programación en el software Visual Studio y la arquitectura del software, para la comprensión de la aplicación. Cada explicación tiene adjunto un diagrama de flujo que especifica los detalles del proceso.

El cuarto capítulo presenta las pruebas realizadas en el simulador de vuelo, los resultados y las respuestas de usuarios de acuerdo a encuestas realizadas.

El capítulo cinco contiene las conclusiones y recomendaciones obtenidas del desarrollo del trabajo, que se refieren a la experiencia adquirida en la realización del proyecto y a los resultados de pruebas y encuestas.

CAPÍTULO 1

MARCO TEÓRICO

En este capítulo se definen conceptos fundamentales de los elementos que forman parte del proyecto, los cuales permiten comprender el proceso para la implementación del simulador de vuelo.

1.1 VEHÍCULO AÉREO NO TRIPULADO

1.1.1 INTRODUCCIÓN

La aviación no tripulada se originó con el diseño y construcción de modelos realizados por varios inventores entre los cuales se pueden mencionar a Cayley, Stringfellow, Du Temple, estos personajes junto con otros pioneros de la aviación, construyeron y volaron sus propias aeronaves, y continuaron trabajando en el desarrollo de vehículos aéreos tripulados durante inicios del siglo XIX.

Trabajo que contribuyó al avance de este tipo de vehículos sirviendo como bancos de pruebas tecnológicos para la creación de modelos más robustos con piloto a bordo, hecho por el cual son considerados constituyentes de la aviación tripulada [1].

Al hablar de aviación no tripulada, se abarca un amplio espectro de aeronaves. Comenzando desde sus raíces que se encuentran en el desarrollo de torpedos aéreos, y continuando más adelante con los misiles crucero, los cuales abrirían las puertas al desarrollo de bombas guiadas no propulsadas, blancos aéreos, señuelos, modelos recreacionales y/o deportivos de radio control, aeronaves de investigación, aeronaves de reconocimiento, aeronaves de combate, y también algunos modelos de vuelo extra atmosférico [1].

Algunos de los modelos más relevantes se puede observar en la Figura 1.1.

Figura 1. 1. Sistemas no tripulados más relevantes en la década de los 80. a) Lockheed Aquila, b) MBLE Epervier, c) Westland Wisp, d) Compass Cope [1].

Un vehículo aéreo no tripulado conocido como VANT por sus iniciales en español, o UAV por sus iniciales en inglés (Unmanned Aircraft Vehicle) es definido como una aeronave sin tripulación, que puede ser reutilizable y tener la capacidad de mantener un nivel de vuelo controlado y sostenido. Este vehículo es propulsado por uno o más motores [2].

1.1.2 CLASIFICACIÓN

Las primeras aeronaves no tripuladas eran manejadas remotamente, sin embargo a medida que pasa el tiempo y se desarrollan nuevos modelos, el uso del control autónomo en las VANT es cada vez mayor. Entonces dentro de este tipo de vehículos se pueden especificar dos variantes: las que vuelan con un control desde una ubicación remota, y las que lo hacen de manera autónoma sobre la base de planes de vuelo preprogramados usando sistemas más complejos de automatización dinámica [2].

Tomando en cuenta estos antecedentes se puede mencionar una extensa variedad en el diseño de los VANT, sin embargo, las clasificaciones nunca son absolutas. De este modo se establecen algunas posibles [3]:

Por tipo de misión

- Reconocimiento, observación: cabe desde control de fronteras a tráfico marítimo o vigilancia de carreteras, reconocimiento estratégico, fotografía aérea.
- Blancos Aéreos.
- Combate: caza –aire-aire, apoyo aéreo cercano (CAS) o estratégico. VC
- Investigación.
- Salvamento.
- Anti incendios.
- Transporte.

Por origen de la misión

- Civil
- Militar

Por tamaño

- Grandes.
- Medianos.
- Pequeños.
- Micro UAV: Mosquito, Monocopter.

Por la forma de obtener la sustentación

- Más pesados que el aire
  - Ala fija
    - De fuselaje convencional.
    - Ala volante.
  - Fuselaje sustentador.
  - Ala rotatoria.
  - Convertiplanos e híbridos.
- Más ligeros que el aire
  - Dirigibles: UAV Airships.
- Híbridos
  - Cuerpo sustentador + dirigible.

Por su motor

- Alternativo
- Turbinas: turbofanes, turbohélices, turboejes, etc.
- Eléctricos: solares, pila de combustible, pila de hidrógeno.

Por el origen del diseño

- Dedicado.
- Procedente de un avión no tripulado, modificado.

Por la forma de despegue

- Desde una pista.
- Lanzado con catapulta u otros medios mecánicos.
- Lanzados a mano.

Por la duración de la misión

- Larga duración (LE – Long Endurance)
- Media duración (ME - Medium Endurance)
- Corta duración (SE - Short Endurance)

Por cota de vuelo

- Alta cota/Muy alta cota (HA – Hight Altitude /VHA)
- Media cota (MA-Medium altitude)
- Baja cota (LA – Low Altitude)

Por el tipo de control

- Autónomo y Adaptativo: El UAV está totalmente gobernado por sus sistemas de abordo, sin intervención del operador en tierra. El UAV tiene la capacidad de re-planificar su vuelo en función de los cambios producidos en su entorno. El UAV puede interactuar con otros UAVs (de su tipo o no) y tomar decisiones solo.

- Monitorizado: El UAV opera de forma autónoma. Un operador controla la retroalimentación del UAV. El operador no puede controlar el UAV (no controla sus mandos), pero puede tomar decisiones por él.
- Supervisado: El UAV realiza unas pocas operaciones de forma autónoma. El control recae en su gran mayoría sobre el operador.
- Autónomo-no adaptativo (o preprogramado): El UAV obedece a una rutina pre-programada, y no tiene la capacidad de cambiar esa rutina para adaptarla a los cambios externos.
- Mando directo por un operador(R/C): El UAV responde directamente a los mandos de un operador.

1.1.3 APLICACIONES

El avance en vehículos aéreos no tripulados ha sido significativo, por lo que sus aplicaciones han ido aumentando con el paso del tiempo. Una de las principales razones es que pueden ser usados para tareas que involucran algún tipo de dificultad o riesgo para vehículos convencionales tripulados por personas [4],

Las motivaciones para su uso son cuantiosas, se pueden mencionar situaciones en las que es conveniente disponer de una vista aérea o tener la capacidad de situar un sensor en un determinado punto del espacio. También son muy necesarios en entornos desestructurados, ya que en ambientes de este tipo el acceso por tierra hasta el objeto a ser inspeccionado se puede convertir en una tarea irrealizable [5]. Asimismo los VANT adquieren gran importancia en la realización de tareas de inspección, control y sensado en ambientes radiológicos y de alta toxicidad química, lugares accidentados o de concurrencia masiva, reduciendo de esta manera la exposición humana [4].

Otra de las áreas en las que estos vehículos son de gran importancia es en la seguridad, permitiendo y facilitando las labores de vigilancia de amplias extensiones de terreno en las que no se tenga la capacidad de un oportuno traslado de personal o de equipos, debido a la inadecuada infraestructura, es decir, en lugares donde las inspecciones terrestres se trocarían en ineficientes.

Ampliando el campo de aplicaciones de los VANT, el uso de estos vehículos tiene lugar en las inspecciones de estructuras civiles de gran tamaño, en donde se requiere del montaje de grandes andamios o de colgamientos de personas con cuerdas [4].

Cada una de las tareas en las que se refiere el uso de aeronaves no tripuladas, implicarían un elevado costo, además de riesgo para las personas, y pérdidas globales de productividad, si son descartadas como una opción para realizarlas [5].

En las Figuras 1.2, 1.3 y 1.4 se observa imágenes de algunas de las aplicaciones de los UAV´s que van desde usos domésticos, educativos o de inspección en lugares inaccesibles o inestables.

Figura 1. 2. Vuelo de un UAV's en interiores [6].

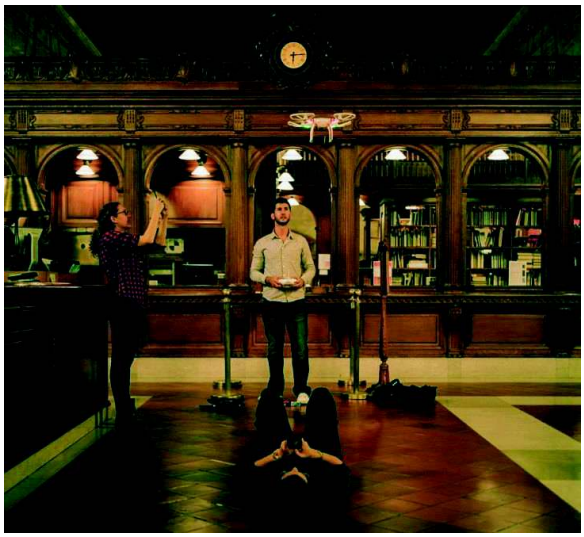

Figura 1. 3. Bibliotecas donde se prestan drones [7].

Figura 1. 4. Drones autónomos que vuelan por edificios inestables o espacios peligrosos a 20m/s, evitando la necesidad de presencia física [8].

1.2 HEXACÓPTERO

Un hexacóptero es un tipo de vehículo aéreo no tripulado, que consta de seis rotores dispuestos en tres diferentes configuraciones posibles. Este multirotor es controlado por giroscopios digitales, los que se encargan de monitorear la estabilidad y el nivel de la nave, información que es procesada por un computador que posteriormente la transmitirá a los operadores, y que puede ser dirección, altura o estado de batería, para así tener control total de la nave. Esta información dependerá de la instrumentación con que cuente el hexacóptero, algunos pueden ser muy sofisticados y otros muy básicos, lo que dependerá siempre del uso que se le vaya a dar al mismo [9].

Los tipos de los hexacópteros más avanzados están equipados en su mayoría con sistemas de GPS que mantiene al multirotor estático en una posición determinada, en conjunto con un sensor o altímetro barométrico, respetando también la altura determinada por los operadores en tierra. Algunos también pueden ser programados con una ruta de vuelo específica con diferentes posiciones, mediante sistemas de navegación, y con la opción de almacenar la ruta en una computadora para después repetir el mismo vuelo, altura y posición en otras fechas diferentes [9].

Entre las características principales de este tipo de vehículo que por su construcción son simples mecánicamente, se puede señalar su buena estabilidad,

debido al sentido opuesto de giro de sus hélices, que evita que giren sobre su propio eje de forma continua debido a la inercia de las estas. Además tienen buen tiempo en vuelo con relación a mutirotores de menor número de rotores, debido a que pueden llevar baterías más grandes y a que los motores trabajan a menos revoluciones. Finamente es importante recalcar su gran potencia y capacidad de carga [10].

1.2.1 CONFIGURACIONES

Los brazos de un hexacóptero pueden estar distribuidos en cuatro posibles configuraciones, nombradas de acuerdo a la disposición de los mismos, y son: cruz (+), equis (x), Y6 y H.

Las dos más importantes y utilizadas son: cruz (+) y equis (x). Llevan ese nombre por la orientación de los motores delanteros con respecto al tren de aterrizaje y la placa de control [10].

La diferencia entre las dos configuraciones antes indicadas se puede observar en la Figura 1.5. El uso de cualquiera las dos configuraciones no afecta el comportamiento de la aeronave, se las elige de acuerdo a la vista que se desee obtener, lo que dependerá de la posición de la cámara integrada en el hexacóptero [10].

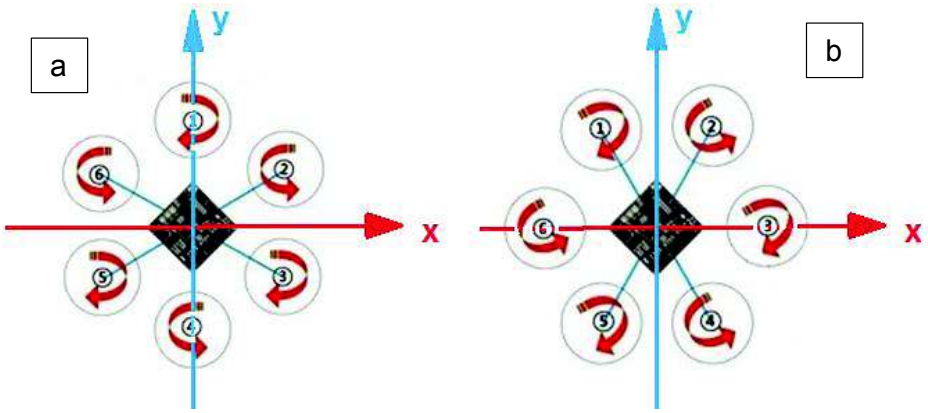

Figura 1. 5. Configuraciones de un hexacóptero: a) Cruz (+), b) Equis (x) [11].

La tercera configuración se diferencia de las dos anteriores, por la forma geométrica de colocar los 6 motores, y ésta se muestra en la Figura 1.6

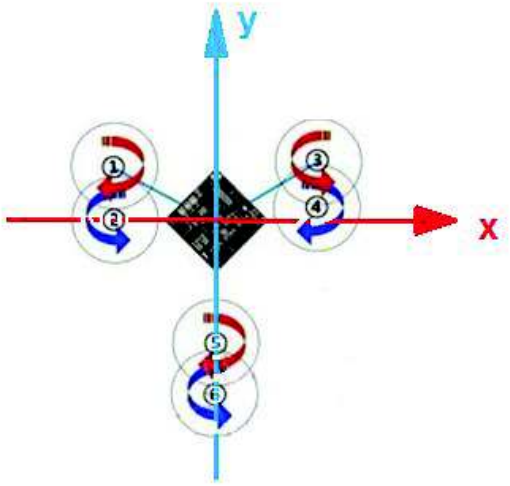

Figura 1. 6. Configuración Y6 para hexacóptero [4]

Esta configuración es coaxial y se denomina Y6, debido a la forma que conciben los brazos, en la cual por cada viga, se colocan 2 motores: uno en la parte superior y otro en la inferior. Dado que estos giran en sentido contrario, los momentos angulares se cancelan entre sí [4].

Aun cuando esta configuración es más compacta y con menos partes, el vehículo tiene un empuje total menor que el de un hexacóptero con geometría radial como el caso de las configuraciones cruz y equis, a causa de que las hélices que se encuentran en la parte inferior, obtienen un flujo de aire turbulento a causa de la hélice superior, disminuyendo así el rendimiento [4].

La cuarta configuración no es muy utilizada, aquí los motores se colocan en dos pares de líneas rectas, a los costados del agarre central en forma de V. Esto permite tener una visibilidad más amplia, que puede servir para la inspección de terreno a medida que se realiza una trayectoria [4]. En la Figura 1.7 se observa una imagen de esta configuración.

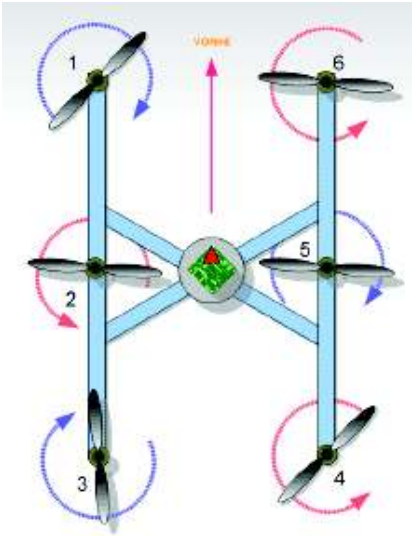

Figura 1. 7. Configuración H para hexacóptero [12].

1.2.2 VUELO

Un hexacóptero, tiene un funcionamiento de vuelo similar al del helicóptero, con la diferencia de que el hexacóptero tiene 4 ejes y 4 motores más. Este vehículo logra su movimiento mediante la rotación de las hélices que son las que generan el empuje adecuado para lograrlo [13].

Ya que cada uno de los motores puede ser controlado de manera independiente, es posible generar diferentes traslaciones y rotaciones, con la variación de las velocidades sobre cada eje de acuerdo a las necesidades del control [13].

Para el vuelo del hexacóptero, se toman en cuenta las fuerzas y torques que actúan sobre él, como son la gravedad, la fricción del aire y los torque producidos por las hélices, que se representan en la Figura 1. 8 [13].

Los movimientos posibles para la aeronave son tres traslaciones en las direcciones de los ejes cartesianos *x*, *y* y *z* y asimismo la rotación sobre cada uno de estos ejes [4].

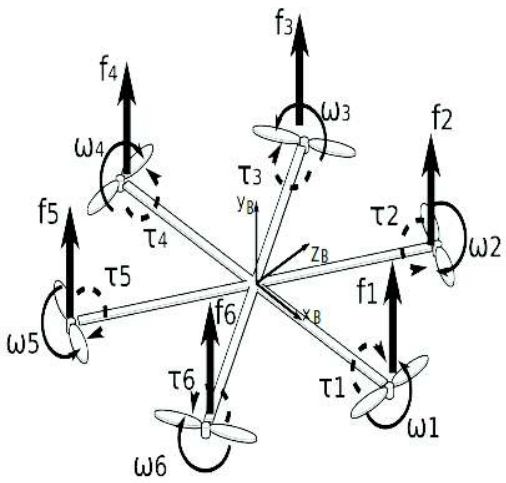

Figura 1.8. Empujes, torques y velocidades angulares de un hexacóptero [13].

Estos movimientos son posibles mediante la variación de rotación en las hélices del hexacóptero de una manera armónica respecto un motor del otro [4]. Cada uno de estos movimientos se explica a continuación, de acuerdo al sistema de referencia de la Figura 1. 9. Y son válidos para la configuración cruz (+) y equis (x) que son las más importantes, y las que se implementaron en el simulador de vuelo.

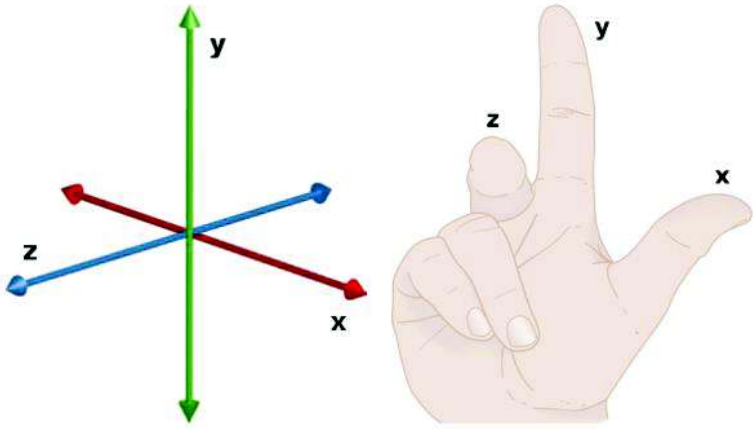

Figura 1.9. Sistema de coordenadas en el que se basa los movimientos del hexacóptero [14]

1.2.2.1 Elevación (Throttle)

El movimiento más sencillo, es el ascenso y descenso vertical de la aeronave, lo que se hace posible con el aumento y disminución de la velocidad de todos los rotores de forma igual y simultáneamente (Figura 1.10). Esta variación no produce ningún desequilibrio en los brazos, por lo que la estructura permanece horizontal.
A medida que aumenta la aceleración de las hélices, también lo hace el empuje generado y consecuentemente, la velocidad del ascenso. De la misma forma, al disminuir gradualmente la velocidad de las hélices, el empuje producido no es capaz de sustentar el peso del hexacóptero, entonces disminuirá la altura [4].

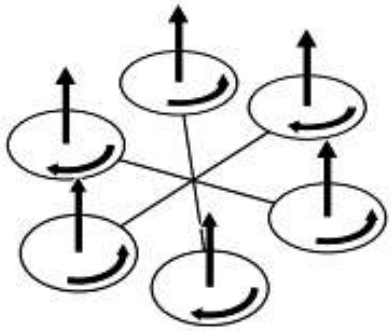

Figura 1. 10. Movimiento de elevación [13].

1.2.2.2 Balanceo (Roll) [13]

Este ángulo representa la rotación sobre el eje “x” y genera el movimiento a la izquierda o derecha, como se observa en la Figura 1.11.

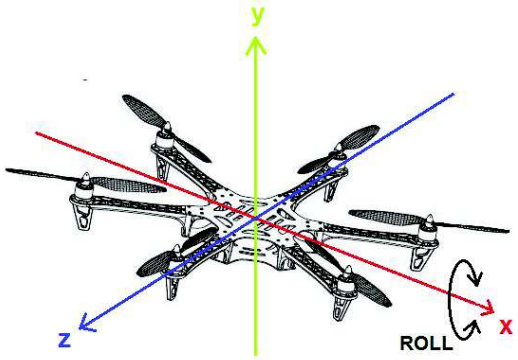

Figura 1. 11. Movimiento de Alabeo (Roll)

Para generar el movimiento a la izquierda se disminuye el empuje producido por las hélices del lado izquierdo del hexacóptero, mientras que aumenta el empuje que se produce por las hélices del lado derecho.

En cambio el movimiento hacia la derecha se logra disminuyendo el empuje producido por las hélices del lado derecho, al mismo tiempo que se aumenta el empuje que producen las hélices del lado izquierdo.
Al realizar cualquiera de los dos movimientos el empuje total permanecerá sin cambios, produciéndose solamente la rotación deseada.

1.2.2.3 Cabeceo (Pitch) [13]

Este ángulo representa la rotación sobre el eje “z” y genera los movimientos adelante y atrás de la aeronave, como se observa en Figura 1.12.
Para el movimiento hacia adelante se incrementa el empuje producido por las hélices de la parte posterior del hexacóptero y se disminuye el que se produce por las hélices delanteras.
Si se desea realizar el movimiento hacia atrás se tendrá que disminuir el empuje que producen las hélices de la parte posterior, al tiempo que se aumenta el empuje producido por las hélices delanteras.
De esta manera al igual que para el ángulo roll, el empuje total permanecerá sin cambios, produciéndose solamente una rotación.

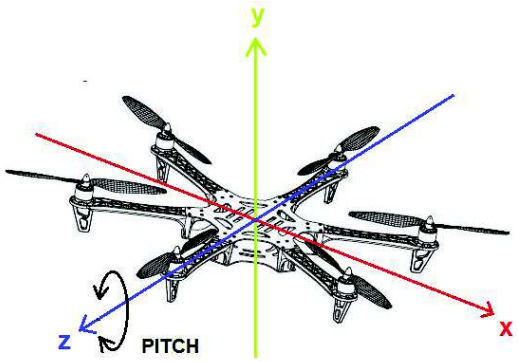

Figura 1. 12. Movimiento de Cabeceo (Pitch)

1.2.2.4 Guiñada (Yaw) [13]

Este ángulo representa la rotación alrededor el eje “y”, y se genera al incrementando el empuje producido por las hélices que rotan en sentido horario mientras disminuye el empuje producido por las hélices que rotan en sentido anti-horario, como se ilustra en la Figura 1.13.

Y el giro en el sentido contrario se consigue disminuyendo el empuje producido por las hélices que rotan en sentido horario, mientras incrementa el empuje producido por las hélices que rotan en sentido anti-horario. El empuje total, al igual que en los dos movimientos anteriores, permanecerá constante, y solo habrá rotación.

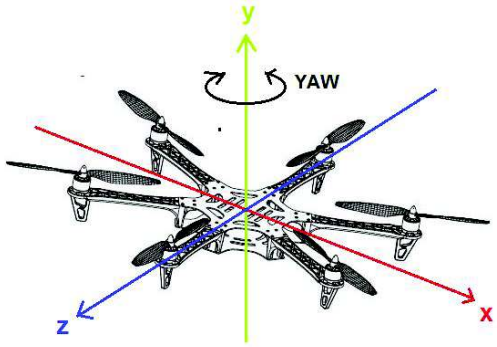

Figura 1. 13. Movimiento de Guiñada (Yaw).

1.3 SIMULADORES DE VUELO

Hoy en día los simuladores de vuelo han tomado gran importancia, abarcando un amplio campo de aplicaciones. Estos simuladores se definen según varios autores como sistemas que intentan replicar, o simular, la experiencia de pilotar una aeronave de la forma más precisa y realista posible.

En la búsqueda del conocimiento de estos sistemas, se encuentran diferentes tipos de simuladores de este tipo, que van desde videojuegos hasta réplicas de cabinas en tamaño real ensambladas en accionadores hidráulicos o electromecánicos y que pueden ser controlados por

sistemas modernos computarizados [15], con la posibilidad de generar todos los movimientos estudiados en la sección anterior [16].

Entre un gran número de aplicaciones para los simuladores de vuelo, el más acreditado es el entrenamiento tanto de pilotos de la aviación civil como los de la aviación militar acaparando especial importancia en la preparación de los operadores en ambientes de riesgos como fallas en pleno vuelo y situaciones de desastre [16].

Debido a que realizar un vuelo real llega a ser peligroso para los operadores sin un entrenamiento previo, desde los principios de la aviación tripulada y no tripulada, diversos esquemas se usan para que estas personas puedan sentir la sensación de volar sin ser realmente aerotransportados [16].

1.3.1 SIMULADORES DE VUELO PROFESIONALES [15]

En la actualidad se han desarrollado varios tipos de simuladores de vuelo con el fin de preparar a los operadores y como cualquier otro sistema se los puede encontrar de todo tipo, desde sistemas básicos de entrenamiento hasta simuladores de vuelo complejos con hasta seis ángulos de movimiento.

Todo tipo de simulador de vuelo, desde los más básicos hasta los de última generación, tiene la función fundamental de desarrollar habilidades en los operadores de las aeronaves, preparándolos para situaciones normales y de emergencia, además de brindar la experiencia necesaria en despegue y aterrizaje del mismo.

Algunos escenarios de emergencia pueden ser: fallas en los sistemas electrónicos, pérdidas de potencia, vientos de cola. Estas acciones en una aeronave real resultarían peligrosas e incluso representarían costos adicionales.

En algunos países los simuladores de vuelo necesitan ser certificados por instituciones gubernamentales, por ejemplo la Administración Federal de Aviación

de estados unidos (FAA) y Direcciones de Aeronáutica Civil de diferentes países, estas entidades regulan y certifican estos sistemas según su categoría en niveles A, B, C y D, de acuerdo a las Guías de Pruebas de Aprobación (ATG) que corresponde a la documentación donde se especifica cada una de las características técnicas del simulador y cómo se prueba y comprueba su correcto funcionamiento.

La principal exigencia para la certificación de los simuladores consiste en demostrar que sus características de vuelo coinciden exactamente con las de la aeronave para la cual fue fabricado. En estas categorías la certificación de nivel D es la más avanzada y garantiza que el simulador reproduce con la máxima fidelidad el comportamiento de la aeronave real y por tanto es apto tanto para el entrenamiento.

1.3.2 APRENDIZAJE MEDIANTE SIMULADORES DE VUELO

Los simuladores de vuelo buscan brindar principalmente práctica y experiencia para que posteriormente los usuarios de estos sistemas tengan la capacidad de operar la aeronave real. Como en cualquier otra actividad el manejar vehículos aéreos no tripulados, conlleva gran dedicación a esta actividad, involucrando varias horas de práctica para mejorar las técnicas de vuelo, en la Figura 1.14 se observa un ejemplo de simulador de vuelo de un vehículo aéreo. Es aquí donde aparecen los simuladores buscando ejecutar su objetivo, y evitando la necesidad de asistir diaria o regularmente a un campo de vuelo, e impidiendo las reparaciones de las aeronaves en el caso de ser estrelladas [17].

Desde los inicios de la aviación tripulada y no tripulada, se percibió la necesidad de la práctica del manejo de estos sistemas, sin embargo durante su evolución y para llegar a los alcances que actualmente existen en este campo, existieron percances que advirtieron caminos equivocados, así como los senderos del progreso [18]

Figura 1. 14. Simulador de vuelo de un vehículo aéreo [19]

Teniendo en cuenta estos antecedentes, el entrenamiento de operadores de aeronaves ha tomado fuerza con el paso de los años, llegando a ser parte fundamental en su formación el uso de simuladores, familiarizándolos de esta manera con el comportamiento de la aeronave a la que tendrán que controlar físicamente [18].

Al igual que un instructor o guía en cualquier área, el simulador de vuelo le presentará al operador situaciones adversas, obligándolo a enfrentarlas, para que así sea capaz de conocer, aprender y explotar sus habilidades muchas veces desconocidas por él mismo, o de la misma manera enfrentarse a sus limitaciones. Esto le permite progresar en el aprendizaje, sin causar pérdidas económicas y sobretodo sin comprometer su seguridad y la de otras personas.

La única manera de especializarse en el vuelo es adquirir experiencia con la propia práctica. Dándoles la oportunidad al cerebro, la memoria y los sentidos, de asimilar eventos que a medida que pasa el tiempo se volverán familiares, y que serán útiles en el momento conveniente [18].

Con estas referencias es indiscutible la necesidad del uso de simuladores de vuelo como herramienta de aprendizaje, lo que resulta factible gracias a los avances tecnológicos que ya existen y otros que se siguen desarrollando. En la

actualidad se han llegado a obtener experiencias que se acercan hasta un 80% al comportamiento real del vehículo. Con la ayuda de desarrolladores de aplicaciones y programadores el avance en esta área es prometedor [18].

Resumiendo, las ventajas que demuestra el uso de un simulador de vuelo como facilitador en la adquisición de experiencia son sumamente altas, ya que este logra llevar al aprendiz al extremo de sus capacidades, exigiéndole un entrenamiento de alto rendimiento, el cual involucra el aspecto mental, físico y psicológico. Obteniendo así técnicas adecuadas de vuelo, correcta ejecución de las mismas así como pérdida del miedo a cometer errores. Todo esto siempre en base a conocimientos sólidos de la teoría que implica la aeronave que se desea volar [18].

Se debe recalcar que la introducción a los simuladores de vuelo necesitará en su mayoría de un guía que tiene ya la experiencia suficiente para proporcionar las pautas necesarias al principiante. Además se debe comprender que aún cuando la tecnología presente todos los escenarios de manera casi real, estos no lo son. Para tener la capacidad de hacer volar de manera física una aeronave es necesario combinar la teoría, la práctica simulada y la experiencia única de la práctica real [18].

1.4 INTERFAZ HOMBRE MÁQUINA

El concepto de Interfaz Hombre Máquina (HMI) o conocida también como interfaz de usuario asistida por ordenador, según la Normativa Técnica Sobre el Trabajo Con Ordenadores (ISO 9241-110), se define como "todas las partes de un sistema interactivo (software o hardware) que proporcionan la información y el control necesarios para que el usuario lleve a cabo una tarea con el sistema interactivo". La interfaz hombre-máquina (HMI) se puede comprender como el punto de acción en que un hombre entra en contacto con una máquina. Para que una interfaz hombre-máquina (HMI) sea útil y significativa para el usuario, debe estar adaptada a sus requisitos y capacidades [20].

La interfaz hombre máquina se la representa de diferentes maneras según el campo de aplicación en el que se esté trabajando. En el caso específico de este proyecto que es la simulación, esta interfaz proporciona la capacidad de representar el sistema, este sistema consta de ambientes 3D y de aeronaves sometidas a fuerzas físicas, que se convierten en los aspectos que darán al usuario la sensación de estar en una situación paralela, es decir, controlando una aeronave real. En la Figura 1.15 se observa al usuario realizando una simulación dinámica e interactiva en tiempo real, con dispositivos externos, como casco de realidad virtual y joystick.

Figura 1. 15. Simulación dinámica e interactiva en tiempo real, con dispositivos externos [21]

El usuario puede interactuar con el sistema, ya que no solamente observa una situación, sino que la controla. Además se introduce a esta realidad, experimentándola y explorándola por medio de diferentes opciones de HMI que van desde teclados, joystick, gamepad, hasta trajes sensoriales, cada uno con distinta tecnología, que será la que determine el nivel de interacción con la aplicación. Esta interacción del usuario con el simulador busca que éste sea capaz de producir cambios en la realidad virtual, brindándole percepción que generalmente están dirigidos a los sentidos visual, auditivo y táctil por medio de elementos externos, también existen otros que tratarán de llegar al cerebro para

de esta manera evitar la interfaces externas, y están los más simples que se dirigen a la imaginación del hombre para percibir una realidad virtual parcial [21].

Algunos investigadores plantean que las tres bases de la realidad virtual son la interacción, la inmersión y la imaginación [21] características mencionadas en los párrafos anteriores, por lo tanto el desarrollo de este proyecto que constituye la teleoperación de una aeronave no tripulada mediante un simulador que incluye realimentación de fuerza, corresponde a un sistema virtual.

1.4.1 INTERFACES HÁPTICAS [22]

"El término de dispositivo háptico y en sí la palabra háptica son realmente modernos. Proviene del griego hápto (tocar, relativo al tacto) y se refiere a la ciencia que estudia todo lo relativo al tacto y sus sensaciones como medio de control e interacción con máquinas y computadores".

Durante los inicios del desarrollo de entornos virtuales se tomaban en cuenta los sentidos de la vista y el oído, ya que son los que más se utilizan para relacionarse con este tipo de entornos, sin embrago, a medida que transcurre el tiempo y la tecnología evoluciona se involucra también al tacto permitiendo introducirse en un mundo virtual más cercano a un ambiente real. Se implementa entonces las simulaciones de sensaciones hápticas, las cuales pueden realizarse de varias formas.

A continuación se detallan las interfaces y los dispositivos más conocidos y utilizados, los que se clasifican en dos grupos: los táctiles y los de realimentación de fuerzas.

Con la interfaz táctil lo que se pretende es estimular a los mecanorreceptores que se encuentran en la piel. Los mecanorreceptores no se encuentran distribuidos de forma homogénea a lo largo de la toda la piel, sino que se encuentran concentrados en zonas como son las yemas de los dedos, la mano, etc. Si bien se podría hacer dispositivos para cualquier zona, se suele realizar para las yemas

de los dedos, ya que es una zona muy estudiada y donde se encuentran los distintos tipos de mecanorreceptores. Se distinguen entre cinco tipos de interfaces posibles que son vibradores, neumáticos, mecánicos, electrocutáneos y térmicos.

Existen también las interfaces de realimentación de fuerza, ya que las táctiles sólo simulan la presión ejercida al usuario de forma localizada y existe un conjunto de características de los objetos como elasticidad, viscosidad, adherencia, etcétera, que no se pueden conseguir con éstas interfaces, sin embargo con las interfaces de fuerza sí es posible. Es por ello que actualmente éstas son las más aplicadas en realidad virtual. Cuando en un mundo virtual se desea tocar un objeto (apretarlo, manipularlo, etc.) se puede hacer de forma realista, notando que el objeto ocupa realmente un volumen determinado en el espacio. Con estas interfaces se podría establecer un plano virtual, y cuando el usuario lo toque y quiera traspasarlo, de alguna forma el sistema virtual se lo impida.

Las interfaces de fuerza pueden ser de dos tipos:

Los primeros son los exoesqueletos que son armazones colocados sobre algunas articulaciones y miembros del usuario que de forma controlada permiten aplicar una resistencia al movimiento, limitan la libertad de movimiento del usuario.

Los segundos son interactuadores puntuales, los que basan su funcionamiento en no permitir al usuario tocar de forma directa el objeto, sino a través de un medio físico intermedio, como pueda ser una varilla o dedal. Las principales utilidades que tienen estos dispositivos son para el entrenamiento en cirugías y vuelos, ya que son habilidades manuales muy precisas y costosas de realizar.

Phantom (Figura 1.16) es uno de los dispositivos de este tipo más populares. Es un brazo con articulaciones motorizadas que terminando en una varilla, dedal o herramienta (según la versión utilizada), permite al usuario moverse en el espacio virtual e interactuar con él, mediante ese instrumento puntual.

Figura 1. 16. Phantom. Dispositivo del tipo interactuador puntual [22]

Adicional a estos existen nuevas interfaces que están tomando fuerza entre las cuales se pueden mencionar la interfaz táctil, multitáctil y skinput o pantalla táctil en la piel.

1.4.1.1 Joystick Force Feedback

Un joystick según las definiciones expuestas en esta sección es una interfaz hombre máquina, muy demanda y se la utiliza para diferente tipo de aplicaciones dentro de las cuales la más importante es para simuladores de vuelo de pilotos profesionales. Existe una gran variedad en el mercado, este dispositivo puede ser muy sencillo, con unos pocos botones y potenciómetros, hasta uno muy sofisticado, que puede incluir conversores A/D, memorias RAM, mayor número de botones o potenciómetros [23].

Para conocer de qué se trata un Joystick Force Feedback, es importante entender el concepto de “Force Feedback”, que traduciendo desde el inglés, se puede decir que se trata de una realimentación de fuerza. Teniendo en cuenta esta definición para el tipo de joystick que tiene esta característica, se puede decir que además de brindar un efecto de realidad, se refiere a una fuerza que aparece en la palanca de mando como contraposición al movimiento ejercido por el usuario o al relacionado con algún evento o circunstancia propios de la aplicación [23].

Se puede resumir que este tipo de joystick usa una tecnología basada en Tecnología Háptica que genera empuje o resistencia en un dispositivo electrónico, simulando las fuerzas de los elementos de un simulador para que el usuario las sienta directamente a través de vibraciones y retrocesos en el periférico [24].

El usuario será capaz de recibir la sensación de realismo, complementando los estímulos táctiles con los visuales recibidos a través de la pantalla y con los auditivos proporcionados por el simulador de vuelo. Las sensaciones que recibe el tacto son posibles gracias al uso de pequeños servomotores ubicados en el interior de la base del joystick. En muchos casos, este también incorpora un sensor óptico para determinar si, efectivamente, la mano del usuario está sobre la palanca de mandos, como es el caso del joystick Microsoft Force Feedback Pro, que se observa en la Figura 1.17.

Figura 1. 17. Elementos del Joystick Microsoft Force Feedback Pro [25]

Desde ya que los movimientos serán reales siempre y cuando estén incluidos en las rutinas del programa, porque los servomotores deben ser controlados por los impulsos que el joystick envía a la PC, de acuerdo con las alternativas planteadas durante el desarrollo de la aplicación desarrollada [23].

En el mercado se encuentran ya joystick force feedback que tienen aún más prestaciones, se puede encontrar modelos inalámbricos que combinan todas las sensaciones propias de este tipo de componente. Brindando al usuario libertad de movimiento, sin embargo con un requerimiento energético muy alto debido al consumo de los servomotores que también incluyen transmisión inalámbrica, demandando un alto consumo de baterías.

En la Figura 1.18 se puede observar un Joystick con realimentación de fuerza de una marca relativamente nueva en el mercado. Es importante señalar que se habla de elementos cuya comunicación con el computador se realiza en ambos sentidos, dado que a un movimiento del joystick que provoca una transferencia de datos hacia el computador le sigue una reacción del programa vía servomotores, que tendrá una dirección desde el computador hacia el joystick y viceversa [23].

Figura 1. 18. Extreme 3D Pro, Force Feedback Joystick [26]

1.5 FUERZAS FICTICIAS [27]

Este tipo de fuerzas son conocidas también como fuerzas de inercia o pseudofuerzas, y explican la aceleración aparente de un cuerpo visto desde un sistema de referencia no inercial.

Para comprender este concepto se acude al ejemplo de la Figura 1.19, en donde se ilustra el movimiento de un pasajero (en verde) que se encuentra en el interior

de un autobús que acelera, visto por un observador inercial O y por un observador no inercial O' que se encuentra en el interior del autobús [51].

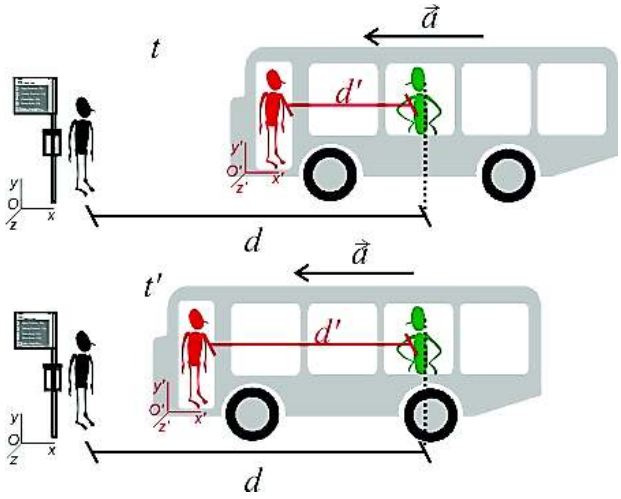

Figura 1. 19. Ejemplo de fuerzas ficticias.

Si el sistema de referencia es O, el pasajero está en reposo y es el autobús el que se acerca a la parada con aceleración a. Sin embargo, desde el punto de vista de *O',* el pasajero se aleja de él con aceleración a, y esta aceleración debe estar provocada por una fuerza.

La fuerza Fi (en rojo en la Figura 1.20) que percibe *O'* es una fuerza de inercia, que no está producida por ninguna interacción física sino que es consecuencia de la aceleración del observador.

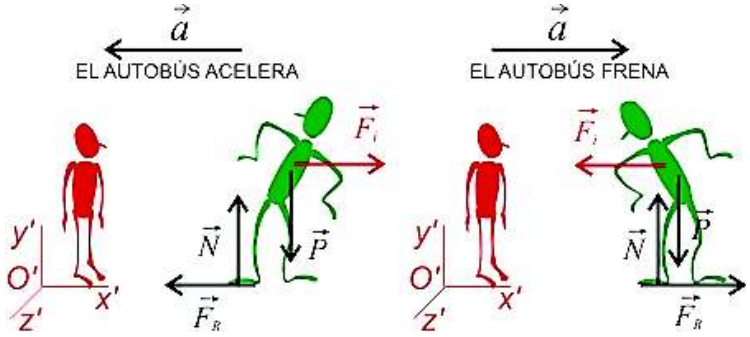

Figura 1. 20. Fuerzas que actúan sobre el pasajero (verde) desde el punto de vista de O'

En la Figura 1.20 se puede observar la fuerza ficticia en color rojo, que puede ser comparada con la sensación de ir hacia atrás cuando el autobús acelera, y hacia adelante cuando frena.

Una vez expuesto el concepto físico de las fuerzas ficticias, es importante mencionar el papel que desenvuelven dentro del estudio e implementación del simulador de vuelo.

Uno de los objetivos que persigue el proyecto es la evasión de obstáculos estáticos en ambientes semiestructurados, para lo que se propone la creación de fuerzas ficticias cuando la aeronave identifique un obstáculo, por medio de la distancia entre estos dos elementos: el hexacóptero y el obstáculo.

El uso de fuerzas ficticias es un método simple y eficiente, estas fuerzas se encuentran dispuestas de tal manera que cubren el entorno de la aeronave y cumplen la función de repeler cualquier obstáculo encontrado en su campo, incrementando esta repulsión cada vez que la distancia entre los dos objetos disminuya.

El diámetro del entorno de la aeronave se definió teniendo en cuenta las dimensiones de los hexacópteros y de los ambientes semiestructurados diseñados estableciendo así, de acuerdo al criterio de la autora del simulador de vuelo, una longitud de tres veces el diámetro de los hexacópteros, considerando este valor como prudencial para la evasión de obstáculos.

En la Figura 1.21 se puede observar el esquema de evasión de obstáculos, donde d es la distancia en la cual empieza la repulsión del hexacóptero hacia el obstáculo encontrado, y F es la fuerza ficticia que será enviada como realimentación hacia el dispositivo háptico, entregándole al usuario la información de cambio de dirección.

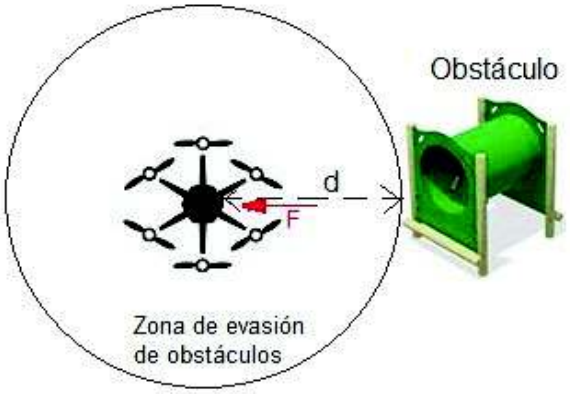

Figura 1. 21. Esquema de evasión de obstáculos

El algoritmo para el cálculo de estas fuerzas está en función de la magnitud de la distancia, y se explica detalladamente en la sección 3.4.8.2.

CAPÍTULO 2

ANÁLISIS DE LOS MODELOS DE HEXACÓPTEROS

En este capítulo se presenta el modelo dinámico del hexacóptero, sus especificaciones y configuraciones, en las que se basa el diseño y la programación del simulador de vuelo.

2.1 MODELADO

Para la obtención del modelo dinámico fundamentado en ecuaciones que describen la posición y orientación de los hexacópteros, se asume a cada aeronave como un cuerpo rígido en el espacio, sujeto a una fuerza principal, que corresponde al empuje, y a tres momentos, que permitirán generar los movimientos del vehículo [28]. Este comportamiento es controlado mediante el ajuste de las velocidades angulares de los rotores que se hacen girar mediante los motores eléctricos [29], y el centro de masa y origen del sistema de coordenadas fijo se considera coincidente, suponiendo que la estructura del hexacóptero es simétrica [28].

Los scripts que controlan los hexacópteros recogen la implementación del modelo dinámico de la plataforma bajo estudio, que es el hexacóptero. En el cual se usa la orientación angular de Euler para parametrizar la orientación de la aeronave; por medio de sus tres ángulos; este método es considerado muy eficaz para explicar la dinámica de drones multirotor como es el caso del presente proyecto [29]

2.2 SISTEMA DE REFERENCIA

Para describir el movimiento del hexacóptero se precisan dos sistemas de referencia [29]:

1. Sistema fijo a tierra

2. Sistema de cuerpo.

El primer sistema que es el fijo a tierra, este sistema es visto como inercial, en el cual la posición lineal absoluta del hexacóptero usa la convención (*x*, *y*, *z*). En la Figura 2.1 puede observarse la orientación del marco móvil (XB, YB, ZB), que corresponde al marco de cuerpo fijo y que se centra en el centro de gravedad de la aeronave.

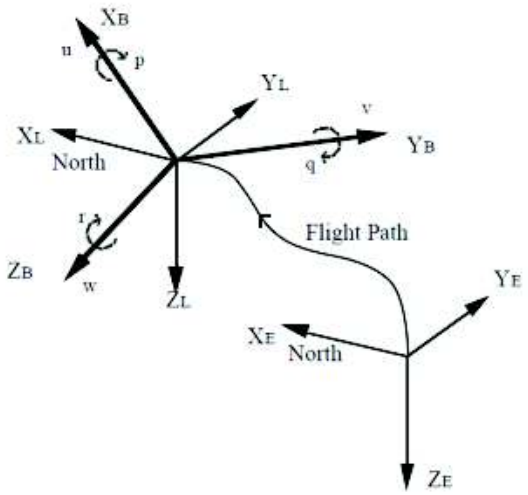

Figura 2.1. Eje Inercial (E), Eje local (L), Eje del cuerpo (B) [29]

El segundo sistema que es el de cuerpo, usa las coordenadas norte, este y sur. El origen de este sistema de referencia se fija en un punto situado en la superficie de la tierra y los ejes son dirigidos de la siguiente forma:

x - hacia el norte
y - hacia el este
z - hacia abajo

La posición angular del sistema de cuerpo con respecto al sistema inercial, es definida por medio de los ángulos de Euler: roll, pitch y yaw.

El vector de posición inercial y el vector angular Euler, se expresa con ξ y η respectivamente, como se observa a continuación:

$$\xi = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{2.1}$$

$$\eta = \begin{bmatrix} \emptyset \\ \theta \\ \psi \end{bmatrix} \tag{2.2}$$

Es necesario encontrar las transformaciones del sistema del cuerpo al sistema inercial, para lo que se recurre al uso de la matriz de rotación, iniciando con las rotaciones alrededor de los ejes [28]:

Rotación Yaw

$$R(\psi) = \begin{bmatrix} \cos(\psi) & sen(\psi) & 0 \\ -sen(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.3}$$

Rotación Pitch

$$R(\theta) = \begin{bmatrix} \cos(\theta) & 0 & -sen(\theta) \\ 0 & 1 & 0 \\ sen(\theta) & 0 & \cos(\theta) \end{bmatrix} \tag{2.4}$$

Rotación Roll

$$R(\emptyset) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\emptyset) & sen(\emptyset) \\ 0 & -sen(\emptyset) & \cos(\emptyset) \end{bmatrix} \tag{2.5}$$

Mediante la rotación de cada uno de los ángulos, de las ecuaciones (2.3), (2.4) y (2.5), se obtiene la matriz de rotación completa R, la cual es ortogonal y se muestra en la ecuación (2.6).

$$R = R(\emptyset)R(\theta)R(\psi)$$

$$= \begin{bmatrix} \cos(\theta)\cos(\psi) & \cos(\psi)\, sen(\theta)sen(\emptyset) - \cos(\emptyset)\, sen(\psi) & \cos(\emptyset)\cos(\psi)\, sen(\theta) + sen(\emptyset)sen(\psi) \\ \cos(\theta)sen(\psi) & \cos(\emptyset)\cos(\psi) + sen(\theta)sen(\emptyset)sen(\psi) & \cos(\emptyset)\, sen(\theta)sen(\psi) - \cos(\psi)\, sen(\emptyset) \\ -sen(\theta) & \cos(\theta)sen(\emptyset) & \cos(\theta)\cos(\emptyset) \end{bmatrix} \tag{2.6}$$

Se tiene entonces la matriz de transformación del sistema inercial o fijo al sistema de cuerpo.

2.3 MODELO CINEMÁTICO [30]

Se plantea el modelo cinemático de la aeronave con el objetivo de facilitar la comprensión de las ecuaciones matemáticas que rigen el comportamiento dinámico del hexacóptero, para lo que se recurre a una generalización del modelo de un cuadricóptero.

Para el desarrollo de este modelo se debe tener en cuenta que el movimiento de un cuerpo rígido se descompone en traslación y rotación, y que las ecuaciones que rigen estos movimientos son las de Newton – Euler.

Este modelo se expresa en función de la velocidad lineal de cada eje de traslación V_x, V_y y V_z, que serán las entradas del sistema, y las velocidades de la aeronave $\dot{x_B}$, $\dot{y_B}$ y $\dot{z_B}$, como se observa a continuación:

$$\begin{bmatrix} \dot{x_B} \\ \dot{y_B} \\ \dot{z_B} \end{bmatrix} = \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} \tag{2.7}$$

Con el modelo definido, se utiliza la matriz de rotación (2.6) como enlace entre el sistema de referencia del centro de masa y el sistema fijo a tierra.

$$\begin{bmatrix} \dot{x_E} \\ \dot{y_E} \\ \dot{z_E} \end{bmatrix} = R(\emptyset, \theta, \psi) \begin{bmatrix} \dot{x_B} \\ \dot{y_B} \\ \dot{z_B} \end{bmatrix} \tag{2.8}$$

Reemplazando (2.6) en (2.8), se obtiene el modelo cinemático completo en función de los ángulos $\emptyset$, θ y ψ:

$$\begin{bmatrix} \dot{x_E} \\ \dot{y_E} \\ \dot{z_E} \end{bmatrix} =$$

$$\begin{bmatrix} \cos(\theta)\cos(\psi) & \cos(\psi)\, sen(\theta)sen(\emptyset) - \cos(\emptyset)\, sen(\psi) & \cos(\emptyset)\cos(\psi)\, sen(\theta) + sen(\emptyset)sen(\psi) \\ \cos(\theta)sen(\psi) & \cos(\emptyset)\cos(\psi) + sen(\theta)sen(\emptyset)sen(\psi) & \cos(\emptyset)\, sen(\theta)sen(\psi) - \cos(\psi)\, sen(\emptyset) \\ -sen(\theta) & \cos(\theta)sen(\emptyset) & \cos(\theta)\cos(\emptyset) \end{bmatrix} \begin{bmatrix} \dot{x_B} \\ \dot{y_B} \\ \dot{z_B} \end{bmatrix}$$

(2.9)

Sustituyendo (2.7) en (2.9)

$$\begin{bmatrix} \dot{x_E} \\ \dot{y_E} \\ \dot{z_E} \end{bmatrix} =$$

$$\begin{bmatrix} \cos(\theta)\cos(\psi) & \cos(\psi)\, sen(\theta)sen(\emptyset) - \cos(\emptyset)\, sen(\psi) & \cos(\emptyset)\cos(\psi)\, sen(\theta) + sen(\emptyset)sen(\psi) \\ \cos(\theta)\, sen(\psi) & \cos(\emptyset)\cos(\psi) + sen(\theta)sen(\emptyset)sen(\psi) & \cos(\emptyset)\, sen(\theta)sen(\psi) - \cos(\psi)\, sen(\emptyset) \\ -sen(\theta) & \cos(\theta)\, sen(\emptyset) & \cos(\theta)\cos(\emptyset) \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix}$$

(2.10)

Para obtener el modelo cinemático reducido se utiliza la técnica de aproximación de ángulo pequeño, que resulta conveniente en la simplificación de las leyes trigonométricas y presenta una precisión aceptable cuando el ángulo tiende a cero, asumiendo $\theta \to 0$ y $\emptyset \to 0$, se tiene $cos\emptyset \cong 1$, $cos\theta \cong 1$ y $sin\emptyset \cong 0$, $sin\theta \cong 0$, obteniendo:

$$\begin{bmatrix} \dot{x_E} \\ \dot{y_E} \\ \dot{z_E} \end{bmatrix} = \begin{bmatrix} \cos(\psi) & -\text{sen}(\psi) & 0 \\ sen(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} \tag{2.11}$$

2.4 MODELO DINÁMICO

El modelo dinámico implementado en la programación del simulador de vuelo es válido para el esquema del hexacóptero mostrado en la Figura 2.2, en la representación se aprecia el sistema de coordenadas en el cual este modelo fundamenta.

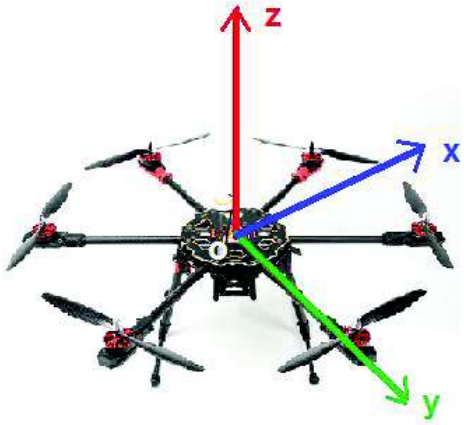

Figura 2. 2. Esquema de los hexacópteros

Como se mencionó en la sección 2.3, el desarrollo del modelo dinámico se lo realiza mediante las ecuaciones de Euler Lagrange y las leyes de Newton [31].

El hexacóptero tiene doce estados, que son los siguientes [31]:

$$X = \left[x, \dot{x}, y, \dot{y}, z, \dot{z}, \emptyset, \dot{\emptyset}, \theta, \dot{\theta}, \psi, \dot{\psi}\right] \tag{2. 12}$$

Donde $x,\ y$ y z son las posiciones en los ejes X, Y, Z; $\dot{x},\ \dot{y}$ y $\dot{z}$ son las velocidades en los ejes. $\emptyset, \theta, \psi$ son los ángulos de Euler roll, pitch y yaw respectivamente; $\dot{\emptyset}, \dot{\theta}$ y $\dot{\psi}$ son las velocidades en los ángulos.

2.4.1 EULER LAGRANGE [30]

El lagrangiano es la suma de la energía traslacional y rotacional menos la energía potencial, lo que está definido por:

$$L(q, \dot{q}) = Ec_{tras} + Ec_{rot} - Ep \tag{2.13}$$

Donde Ec_{tras} es la energía cinética traslacional, Ec_{rot} es la energía cinética rotacional y Ep es la energía potencial del hexacóptero.

Debido a que el lagrangiano no contiene energía cinética en términos combinacionales se tiene el vector de velocidades lineales $\dot{\xi} = [\dot{x}, \dot{y}, \dot{z}]$ y el vector de velocidades angulares $\dot{\eta} = [\dot{\phi}, \dot{\theta}, \dot{\psi}]$, por lo cual el modelo dinámico completo usando las ecuaciones de Euler-Lagrange puede ser dividida en dinámica rotacional y traslación.

$$\begin{bmatrix} F_\xi \\ \tau_\eta \end{bmatrix} = \frac{d}{dt}\left(\frac{\partial L(q,\dot{q})}{\partial \dot{q}_i}\right) - \frac{\partial L(q,\dot{q})}{\partial q_i} \tag{2.14}$$

Donde F_ξ es la fuerza traslacional aplicada al hexacóptero y τ_η son los momentos de roll, pitch y yaw.

Para la dinámica traslacional del vehículo la ecuación de Euler-Lagrange es:

$$F_\xi = \frac{d}{dt}\left(\frac{\partial L(\xi,\dot{\xi})}{\partial \dot{\xi}}\right) - \frac{\partial L(\xi,\dot{\xi})}{\partial \xi} \tag{2.15}$$

Además se conoce que:

$$L(\xi, \dot{\xi}) = Ec_{tras} - Ep \tag{2.16}$$

Donde Ec_{tras} y Ep es:

$$Ec_{tras} = \frac{m}{2}\dot{\xi}^T\dot{\xi} \tag{2.17}$$

$$Ep = mgz \tag{2.18}$$

Reemplazando (2.17) y (2.18) en (2.16), se obtiene:

$$L(\xi, \dot{\xi}) = \frac{m}{2}[\dot{x} \quad \dot{y} \quad \dot{z}]\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} - mgz \tag{2.19}$$

La ecuación (2.19) puede expresarse de la siguiente forma:

$$L(\xi, \dot{\xi}) = \frac{m}{2}(\dot{x}^2 + \dot{y}^2 + \dot{z}^2) - mgz \tag{2.20}$$

Derivando (2.20) se obtiene:

$$\frac{\partial L(\xi,\dot{\xi})}{\partial \xi} = -mg \tag{2.21}$$

$$\frac{d}{dt}\left(\frac{\partial L(\xi,\dot{\xi})}{\partial \dot{\xi}}\right) = m\ddot{x} + m\ddot{y} + m\ddot{z} \tag{2.22}$$

Reemplazando (2.21) y (2.22) en (2.14):

$$F_\xi = m\ddot{x} + m\ddot{y} + m\ddot{z} - mg \tag{2.23}$$

$$F_\xi = m\ddot{\xi} - mgE_z \tag{2.24}$$

Donde $E_z = [0 \quad 0 \quad 1]^T$, de la figura 2.1, con el sistema de referencia en (O) se tiene:

$$\hat{F} = \begin{bmatrix} 0 \\ 0 \\ U_1 \end{bmatrix} \tag{2.25}$$

La señal U_1 es el empuje total de los rotores. Si $F_\xi = R(\emptyset, \theta, \psi)\,\hat{F}$ y reemplazando (2.25) se obtiene:

$$F_\xi = \begin{bmatrix} \cos(\psi)\, sen(\theta)\cos(\phi) + sen(\psi)sen(\phi) \\ \text{sen}(\psi)\, sen(\theta)\cos(\phi) - sen(\phi)\cos(\psi) \\ \cos(\theta)\cos(\phi) \end{bmatrix} \tag{2.26}$$

Finalmente, el modelo dinámico traslacional es:

$$m\ddot{x} = (cos\psi sen\theta cos\phi + sen\psi sen\phi)U_1 \tag{2.27}$$

$$m\ddot{y} = (sen\psi sen\theta cos\phi - sen\phi cos\psi)\, U_1 \tag{2.28}$$

$$m\ddot{z} = (cos\theta cos\phi)U_1 - mg \tag{2.29}$$

Que reescribiendo se obtiene:

$$\ddot{x} = (\cos\emptyset \sin\theta \cos\psi + \sin\emptyset \sin\psi)\frac{1}{m}U_1 \tag{2.30}$$

$$\ddot{y} = (\cos\emptyset \sin\theta \sin\psi - \sin\emptyset \cos\psi)\frac{1}{m}U_1 \tag{2.31}$$

$$\ddot{z} = -g + (\cos\emptyset \cos\theta)\frac{1}{m}U_1 \tag{2. 32}$$

Para la dinámica rotacional la ecuación es:

$$\tau_\eta = \frac{d}{dt}\left(\frac{\partial LRot}{\partial \dot{\eta}}\right) - \frac{\partial LRot}{\partial \eta} \tag{2.33}$$

Se sabe que:

$$Ec_{ROT} = \frac{1}{2}\omega^{\mathrm{T}}\mathrm{I}\omega \tag{2.34}$$

Donde ω es el vector de la velocidad angular e I es la matriz de inercia definida por:

$$I = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \tag{2.35}$$

La variación de tiempo de los ángulos $\dot{\eta} = [\dot{\phi} \quad \dot{\theta} \quad \dot{\psi}]^T$ es una función discontinua, por lo tanto, el vector de velocidad angular teórica del cuerpo $\omega = [p \quad q \quad r]^T$ es diferente. Para relacionar ω con $\dot{\eta}$ se utiliza la siguiente relación:

$$\omega = W_n\dot{\eta} \tag{2.36}$$

Donde W_n se define como:

$$W_n = \begin{bmatrix} 1 & 0 & -sen\theta \\ 0 & cos\phi & sen\phi cos\theta \\ 0 & -sen\phi & cos\phi cos\theta \end{bmatrix} \tag{2.37}$$

Reemplazando (2.37) en (2.36):

$$\omega = \begin{bmatrix} 1 & 0 & -sen\theta \\ 0 & cos\phi & sen\phi cos\theta \\ 0 & -sen\phi & cos\phi cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{2.38}$$

Desarrollando y derivando (2.38) respecto a η se tiene:

$$\frac{\partial \omega}{\partial \eta} = \begin{bmatrix} 0 & -\dot{\psi}cos\theta & 0 \\ -\dot{\theta}sen\phi + \dot{\psi}cos\phi cos\theta & -\dot{\psi}sen\phi sen\theta & 0 \\ -\dot{\theta}cos\phi - \dot{\psi}sen\phi cos\theta & -\dot{\psi}cos\phi sen\theta & 0 \end{bmatrix} \tag{2.39}$$

Reemplazado (2.34), se obtiene:

$$\tau_\eta = \frac{d}{dt}\left(\omega^{\mathrm{T}} \mathrm{I} \frac{\partial \omega}{\partial \dot{\eta}}\right) - \omega^{\mathrm{T}} \mathrm{I} \frac{\partial \omega}{\partial \eta} \tag{2.40}$$

Derivando cada uno de los términos de la ecuación (2.40) y planteando como el punto de operación alrededor del punto de equilibrio para $\phi = 0$, $\theta = 0$ y $\psi = 0$, es decir $cos\phi \cong 1$, $cos\theta \cong 1$ y $cos\psi \cong 1$ y $sen\phi \cong 0$, $sen\theta \cong 0$ y $sen\psi \cong 0$, donde se tiene la estabilización del sistema.

Por lo tanto se pueden expresar los momentos generalizados roll, pitch y yaw con τ_ϕ, τ_θ y τ_ψ respectivamente:

$$\tau_\phi = I_x(\ddot{\phi}) - \dot{\psi}\dot{\theta}(I_y - I_z) \tag{2.41}$$

$$\tau_\theta = I_y(\ddot{\theta}) - \dot{\psi}\dot{\phi}(I_z - I_x) \tag{2.42}$$

$$\tau_\psi = I_z(\ddot{\psi}) - \dot{\theta}\dot{\phi}(I_x - I_y) \tag{2.43}$$

Donde I_x, I_y e I_z es la inercia del hexacóptero en X, Y y Z respectivamente.

Ya que cada rotor es considerado como una estructura rígida, la dinámica de cada disco del rotor en su eje de rotación puede ser considerada como un sistema

desacoplado en la variable generalizada ω_i, que es la velocidad angular de un rotor alrededor de su eje.

Para los movimientos roll, pitch y yaw del hexacóptero, cada rotor cambia su sentido de giro y su velocidad, como se explicó en la sección 1.2.2, dichos movimientos deben ser llevados a cabo con la fuerza principal constante, es así que de forma matemática puede escribirse:

$$\tau_\eta = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} \frac{\sqrt{3}}{2} l(\omega_2^2 + \omega_3^2 - \omega_5^2 - \omega_6^2) \\ \frac{bl}{2}(2\omega_1^2 + \omega_2^2 - \omega_3^2 - 2\omega_4^2 - \omega_5^2 + \omega_6^2) \\ bl(-\omega_1^2 + \omega_2^2 - \omega_3^2 + \omega_4^2 - \omega_5^2 + \omega_6^2) \end{bmatrix} = \begin{bmatrix} lU_2 \\ lU_3 \\ lU_4 \end{bmatrix} \tag{2.44}$$

Donde $\omega_1,\ \omega_2, \omega_3, \omega_4, \omega_5, \omega_6$ son las velocidades angulares de cada rotor, b es el factor de empuje y l es la distancia entre el rotor y el centro del hexacóptero.

Reemplazando (2.41), (2.42) y (2.43) en (2.44), se obtiene:

$$\tau_\eta = \begin{bmatrix} \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = \begin{bmatrix} I_x(\ddot{\phi}) - \dot{\psi}\dot{\theta}(I_y - I_z) \\ I_y(\ddot{\theta}) - \dot{\psi}\dot{\phi}(I_z - I_x) \\ I_z(\ddot{\psi}) - \dot{\theta}\dot{\phi}(I_x - I_y) \end{bmatrix} = \begin{bmatrix} lU_2 \\ lU_3 \\ lU_4 \end{bmatrix} \tag{2.45}$$

Reescribiendo tenemos:

$$\ddot{\emptyset} = \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) + \frac{l}{I_x}U_2 \tag{2. 46}$$

$$\ddot{\theta} = \dot{\emptyset}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) + \frac{l}{I_y}U_3 \tag{2. 47}$$

$$\ddot{\psi} = \dot{\emptyset}\dot{\theta}\left(\frac{I_x - I_y}{I_z}\right) + \frac{l}{I_z}U_4 \tag{2. 48}$$

Las rotaciones de las hélices producen un efecto giroscopio [30]:

$$\tau_{giroscopio} = \begin{bmatrix} -J_r\dot{\theta}\omega \\ -J_r\dot{\phi}\omega \\ 0 \end{bmatrix} \tag{2.49}$$

Donde J_r es la inercia rotacional de las hélices y ω es la velocidad total de las hélices, donde [31]:

$$\omega = -\omega_1 + \omega_2 - \omega_3 + \omega_4 - \omega_5 + \omega_6 \quad (2.\ 50)$$

Añadiendo este efecto se obtiene el siguiente modelo dinámico rotacional [31]:

$$\ddot{\emptyset} = \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{J_r}{I_X}\dot{\theta}\omega + \frac{l}{I_x}U_2 \quad (2.\ 51)$$

$$\ddot{\theta} = \dot{\emptyset}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{J_r}{I_y}\dot{\emptyset}\omega + \frac{l}{I_y}U_3 \quad (2.\ 52)$$

$$\ddot{\psi} = \dot{\emptyset}\dot{\theta}\left(\frac{I_x - I_y}{I_z}\right) + \frac{l}{I_z}U_4 \quad (2.\ 53)$$

Finalmente, el modelo dinámico completo del hexacóptero, que se implementó es el siguiente [31]:

$$\ddot{x} = (\cos\emptyset \sin\theta \cos\psi + \sin\emptyset \sin\psi)\frac{1}{m}U_1 \quad (2.54)$$

$$\ddot{y} = (\cos\emptyset \sin\theta \sin\psi - \sin\emptyset \cos\psi)\frac{1}{m}U_1 \quad (2.55)$$

$$\ddot{z} = -g + (\cos\emptyset \cos\theta)\frac{1}{m}U_1 \quad (2.\ 56)$$

$$\ddot{\emptyset} = \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{J_r}{I_X}\dot{\theta}\omega + \frac{l}{I_x}U_2 \quad (2.\ 57)$$

$$\ddot{\theta} = \dot{\emptyset}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{J_r}{I_y}\dot{\emptyset}\omega + \frac{l}{I_y}U_3 \quad (2.\ 58)$$

$$\ddot{\psi} = \dot{\emptyset}\dot{\theta}\left(\frac{I_x - I_y}{I_z}\right) + \frac{l}{I_z}U_4 \quad (2.\ 59)$$

2.5 ESPECIFICACIONES DEL MODELO

Para la implementación del modelo de los hexacópteros, expuesto en la sección 2.1, dentro de la programación, es necesario definir sus parámetros, los cuales se

encuentran publicados junto al modelo dinámico [31] y se presentan en la Tabla 2.1.

Tabla 2.1. Parámetros del modelado [31]

Nombre	Descripción	Valor	Unidad
m	Masa	1.83	kg
b	Factor de empuje	2.98e-6	Ns²
l	Longitud del centro de masa al rotor	0.30	m
g	Gravedad	9.8	m/s²
J_r	Inercia rotacional	3.357e-5	kgm²
I_x, I_y	Inercia en x, y	0.0216	kgm²
I_z	Inercia en z	0.0432	kgm²

El motor que se utiliza para el desarrollo del modelo es el Tarot 4006. Según sus especificaciones y características, estos motores están diseñados específicamente para su uso en aeronaves de rotores múltiples. Para conocer la velocidad en revoluciones por minuto de cada motor es importante mencionar los siguientes parámetros:

- KV: 620 RPM
- Voltaje de alimentación: 14.8V
- Potencia: 426W

A continuación se explica la definición de cada uno de los parámetros anteriores de acuerdo a [29]:

- KV: Constante en el motor que indica la cantidad de vueltas en revoluciones por minuto, que da el motor por cada voltio aplicado al controlador.
- Voltaje de alimentación: Indica el voltaje requerido para la operación del equipo, se debe escoger baterías que cumplan con el requisito de voltaje de los motores.

- Potencia: Es la potencia nominal del equipo, se relaciona directamente con la carga máxima que puede soportar.

Estos parámetros son importantes para el cálculo de la velocidad y torque de los motores, información necesaria para la implementación del modelo dinámico en el software Visual Studio.

Con los valores mencionados, la velocidad máxima de cada uno de los motores se calcula de acuerdo a [29] de la siguiente manera:

$$velocidad\ (RPM) = 620KV * 14.8V \tag{2.37}$$

$$velocidad\ (RPM) = 9176 \tag{2.38}$$

2.6 CONFIGURACIONES

Los modelos de hexacópteros disponibles en el simulador de vuelo, tienen sus brazos distribuidos en dos configuraciones: cruz (+) y equis (x). La diferencia entre las dos configuraciones fue indicada en la sección 1.2.1.

El uso de cualquiera las dos no afecta el comportamiento de la aeronave, en las Figuras 2.2 y 2.3 se observan los hexacópteros implementados.
Y en las secciones 3.2 y 3.3 se explican detalladamente el modelado de cada uno de los hexacópteros del simulador.

Figura 2.4. Hexacóptero con configuración X

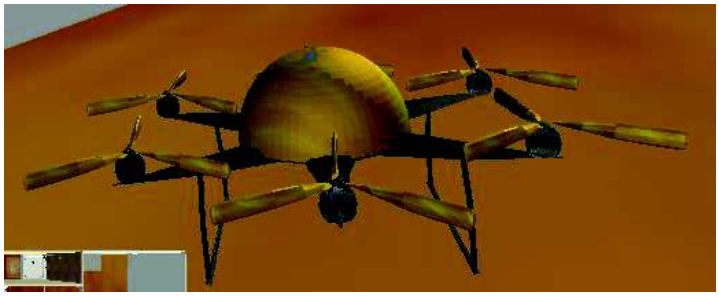

Figura 2. 5. Hexacóptero con configuración +

CAPÍTULO 3

DESARROLLO DEL SOFTWARE

Para el desarrollo del software del simulador de vuelo, se utilizaron diferentes tipos de programas, los cuales son de distribución sin costo, disponibles para su uso, y con recursos de aprendizaje en línea.

El conjunto de aplicaciones que permitieron la creación y manipulación de modelos 3D, tanto de escenarios como de aeronaves son Sweet Home 3D, SketchUp y Blender 3D.

Y finalmente para la implementación del simulador de vuelo de los hexacópteros se usó el motor de videojuego Unity 3D, en el cual se integran los modelos de los ambientes y los hexacópteros realizados. La programación en Unity 3D, se desarrolló mediante el software Visual Studio, en el cual se implementó el modelo dinámico de las aeronaves, y también la realimentación de fuerza, la cual se hace posible mediante el uso del paquete Force Feedback Controller.

La teleoperación del hexacóptero, se la puede realizar mediante un Joystick Force Feedback o directamente desde el teclado del computador.

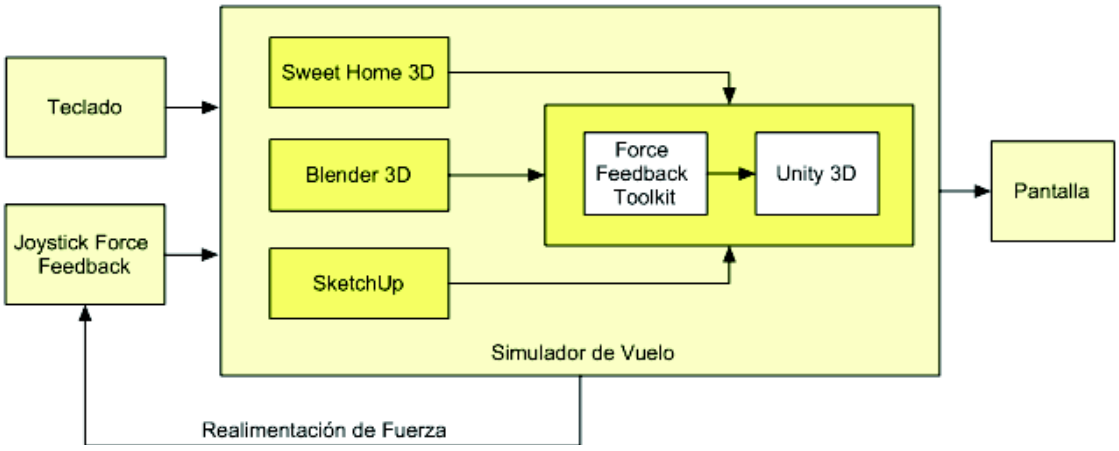

Figura 3. 1. Diagrama de bloques de la aplicación

En la Figura 3.1 se puede observar el diagrama de bloques que especifica el proceso de la aplicación desarrollada y a continuación se describe como fueron utilizadas cada una de las herramientas mencionadas previamente. Además se puede encontrar información detallada de cada una de estas herramientas en los anexos B, C y D de este documento.

3.1 SWEET HOME 3D [32]

Sweet Home 3D es un software de distribución gratuita, que se lo utiliza específicamente para el diseño de interiores, permitiendo partir desde un plano básico, continuando con la construcción de paredes y la ubicación del mobiliario en un plano 2D, siempre teniendo disponible una vista previa 3D, que facilita la visualización del diseño, así como el catálogo y la lista del mobiliario, interfaz que se observa en la Figura 3.2.

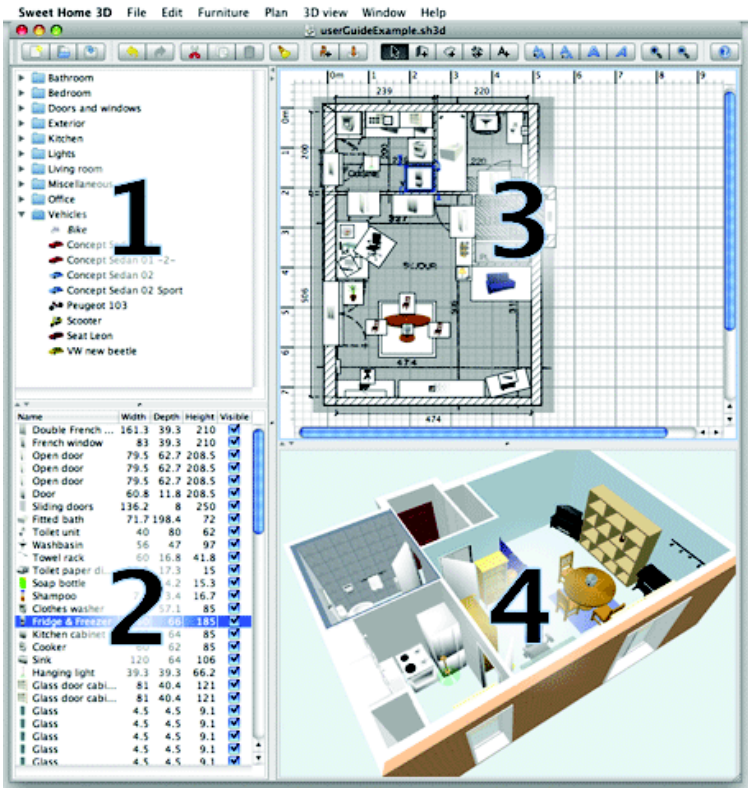

Figura 3. 2. Paneles 3D: 1. Catálogo del mobiliario, 2. Lista del mobiliario del ambiente, 3. Plano 2D de la casa, 4. Vista 3D de la casa [32]

A continuación se presenta en la Figura 3.3 el diagrama de flujo correspondiente al desarrollo del diseño de un modelo en Sweet Home 3D, para posteriormente ser usado en Unity 3D.

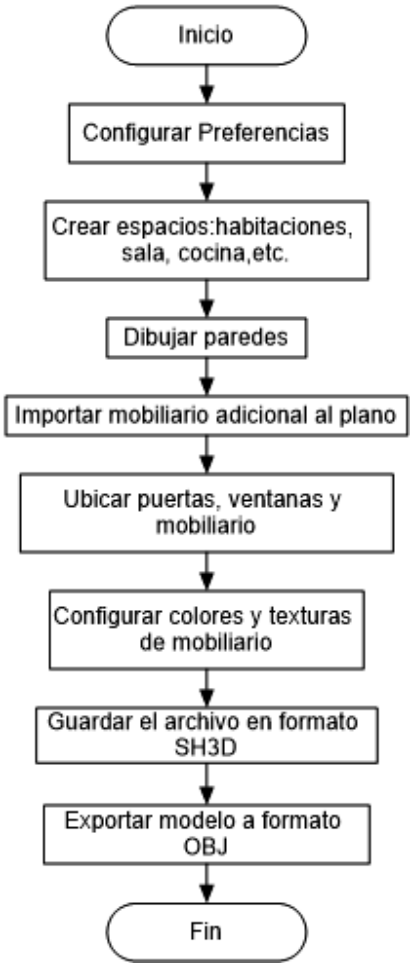

Figura 3. 3. Diagrama de flujo para la creación de un diseño en Sweet Home 3D

De acuerdo al diagrama de flujo de la figura anterior, se realizó el diseño del primer ambiente del simulador de vuelo, el plano 2D de la casa diseñada se puede ver en la Figura 3. 4.

En cualquier momento durante el diseño de la casa, se puede cambiar el punto de vista utilizado en la vista 3D.

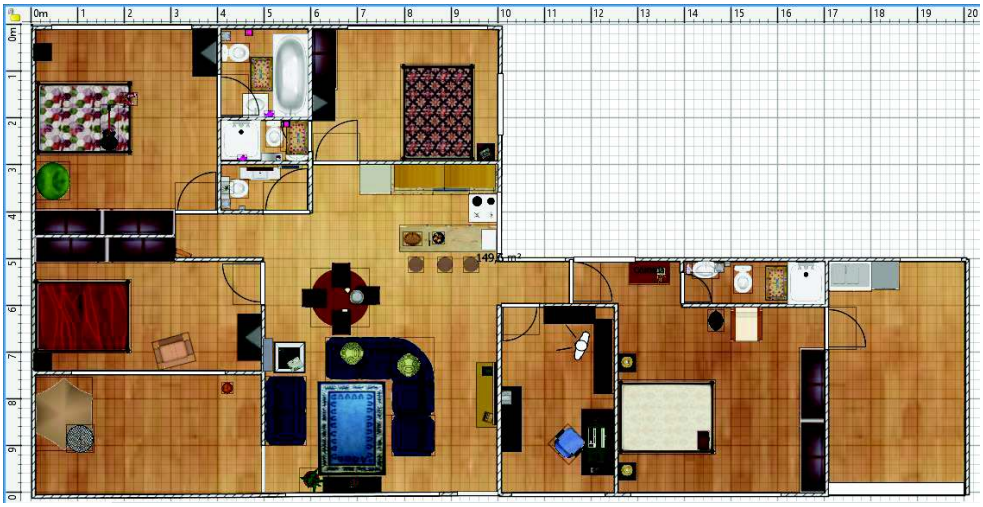

Figura 3. 4. Plano 2D del primer ambiente del simulador de vuelo.

Existen dos puntos de vista que son la virtual y la aérea que es la establecida por defecto. En ambos modos, se puede utilizar el ratón o las flechas del teclado para cambiar el punto de vista actual, como se explica en las Figuras 3. 5 y 3. 6.

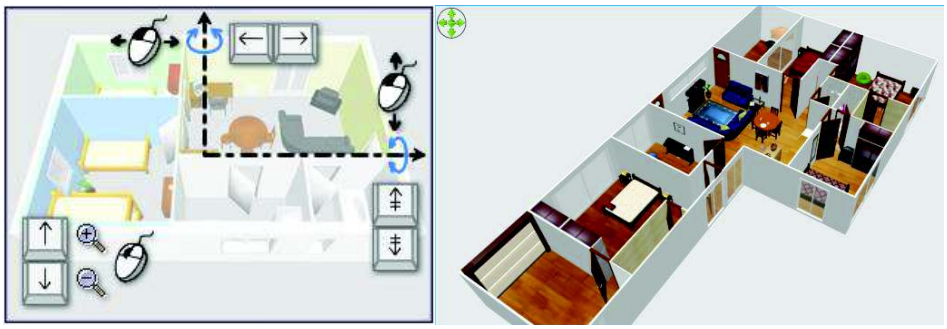

Figura 3. 5. Acciones del ratón y del teclado en el modo vista aérea [32]

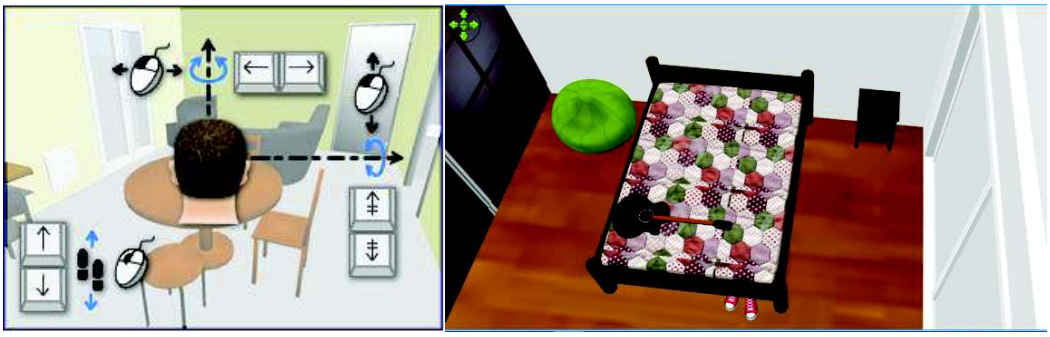

Figura 3. 6. Acciones del ratón y del teclado en el modo visita virtual [32]

Para que el resultado del diseño pueda ser utilizado en Unity 3D, además de guardar el archivo de manera habitual, es indispensable exportarlo con formato OBJ, al hacerlo se escribirá en el archivo seleccionado OBJ la descripción de todos los objetos expuestos en la vista 3D, se creará un archivo MTL describiendo su color y, finalmente, se guardarán las imágenes de las texturas, que se haya utilizado, por lo que es recomendable guardar el archivo exportado en una carpeta vacía de tal manera que el momento de utilizarlo en Unity 3D no haya perdida de información, presentando un modelo incompleto y diferente al diseñado.

3.2 SKETCHUP

SketchUp es un software diseñado para el modelado 3D, está dirigido a diferentes especialidades, según Google SketchUp [33] fue desarrollado para la arquitectura, ingeniería civil, desarrollo de películas y videojuegos. También es conocido por la característica del diseño de estructuras que pueden ser situadas dentro de Google Earth [34].

Al igual que Sweet Home 3D, este presenta una interfaz intuitiva y flexible, que facilitará el aprendizaje del mismo, además viene incluido en su menú de ayuda un video tutorial que demuestra cómo se crea un proyecto desde un archivo nuevo [33].

En la Figura 3.7 se observa el diagrama de flujo del software, que representa gráficamente el proceso para la creación de un diseño o importación de modelos 3D, y su posterior uso en Unity.

Para el desarrollo de este proyecto se acudió a la galería 3D Warehouse, de donde se obtuvieron los modelos de dos ambientes semiestructurados, de una extensión grande y una construcción compleja. Estos modelos corresponden, el primero a la Basílica de Nuestra Señora Aparecida, ubicada en el estado de Sao Paulo en Brasil, y el segundo a un supermercado. Asimismo se utilizó el modelo del hexacóptero comercial DJI_Phantom.

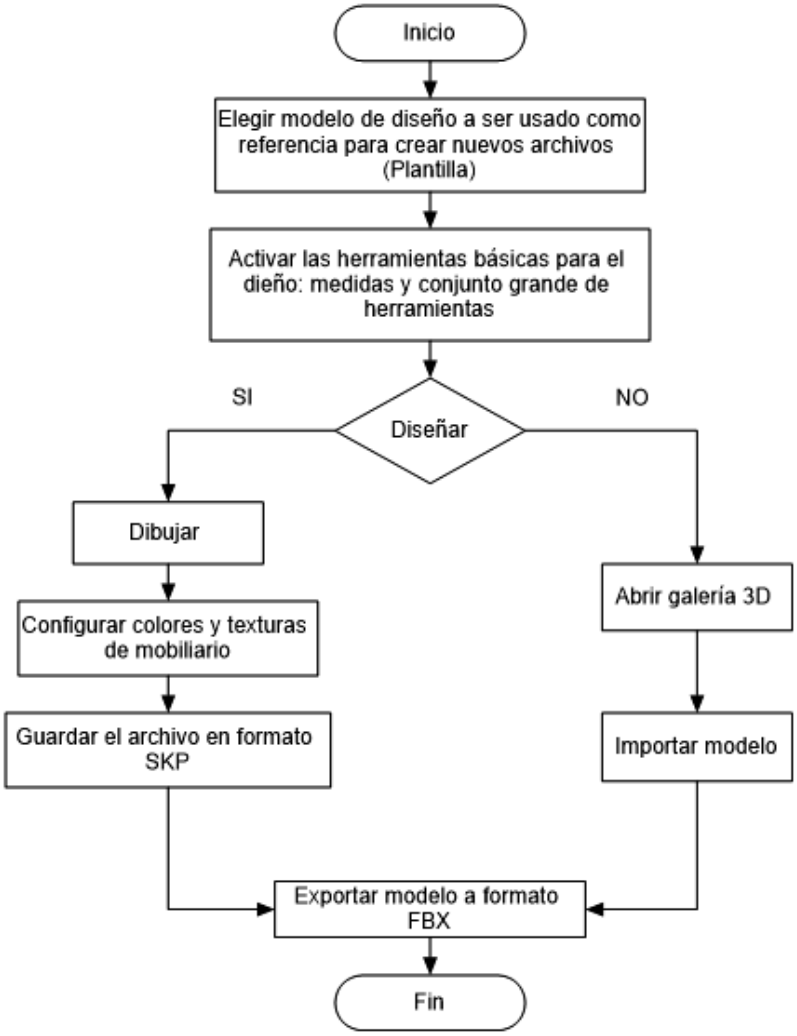

Figura 3. 7. Diagrama de Flujo para la creación de un diseño en SketchUp

Los modelos descargados se observan en la Figura 3. 8 y 3. 9.

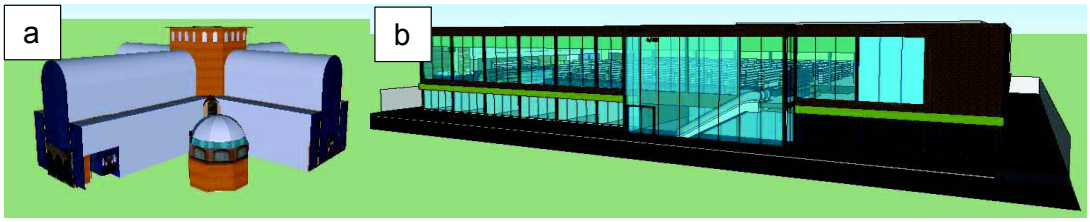

Figura 3. 8. Modelos de ambientes semiestructurados. a) Basílica de Nuestra Señora Aparecida. b) Supermercado

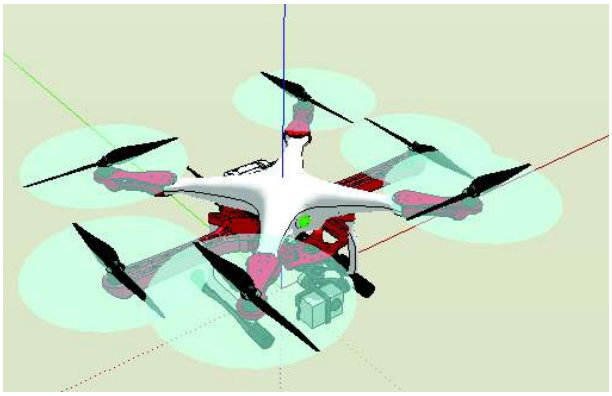

Figura 3. 9. Modelo del hexacóptero comercial DJI_Phantom.

Una vez descargados los modelos es importante conocer que para importar un modelo de SketchUp hacia Unity 3D y poder trabajar de manera adecuada todas las paredes y texturas deben ser visibles, para asegurar esto, el usuario debe exportar el diseño en formato FBX, como se muestra en la Figura 3. 10.

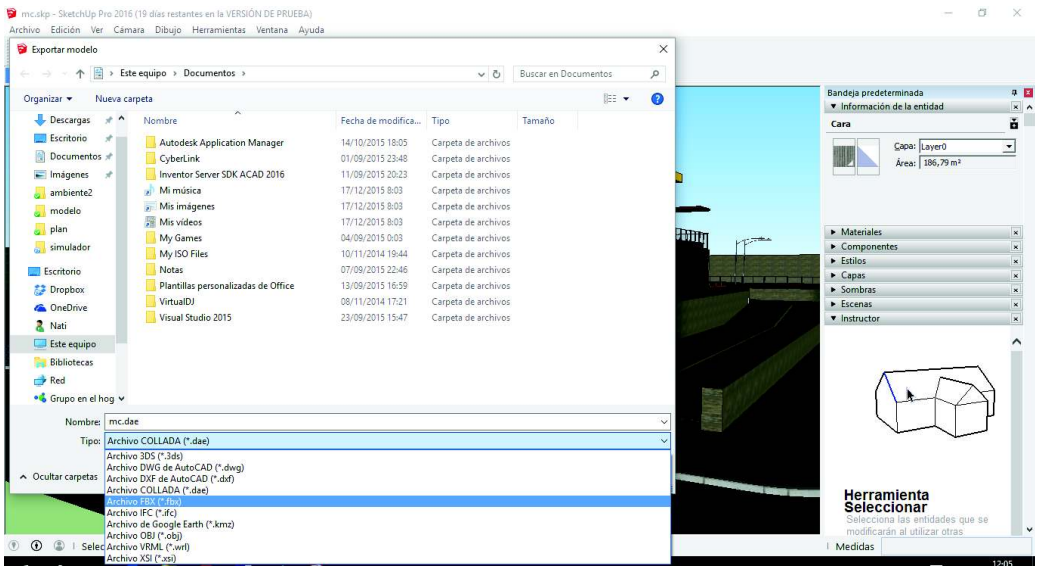

Figura 3. 10. Exportando el modelo a formato FBX

Después de completar el proceso se arrastra o importa el archivo FBX junto con la carpeta de texturas que se generó, hacia la ventana del proyecto de destino en Unity 3D y el modelo está listo para poder trabajar sobre él.

3.3 BLENDER 3D

Blender 3D corresponde al mismo tipo de software de los dos antes mencionados, se orienta principalmente al modelado 3D, animación y creación de gráficos tridimensionales. Este software es libre y compatible con todas las versiones de Windows, Mac OS X, GNU/Linux, Solaris, FreeBSD e IRIX [35].

A diferencia de Sweet Home 3D y SketchUp, este software no tiene una interfaz fácil de comprender, por lo que es casi forzoso acudir al manual, existente en el menú de ayuda del programa, así como videos tutoriales disponibles en línea.

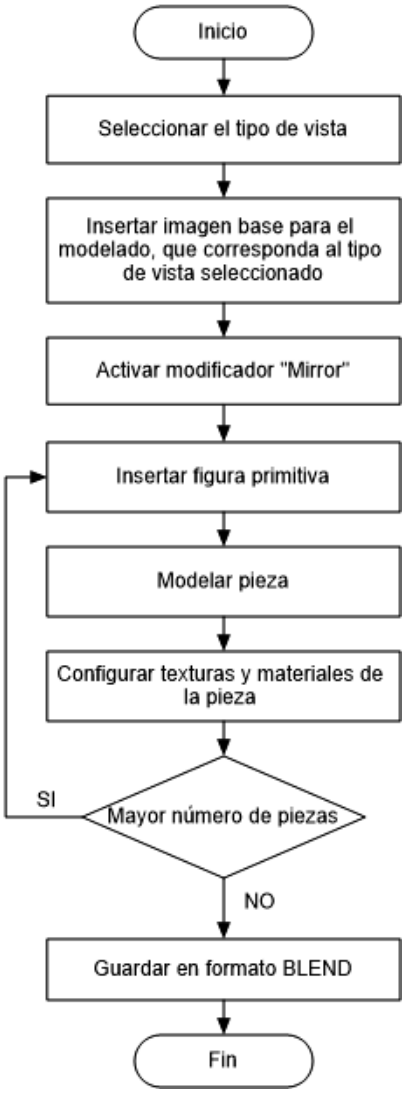

Figura 3. 11. Diagrama de Flujo para la creación de un diseño en Blender 3D

Se puede observar en la Figura 3.11 el diagrama de flujo, que ofrece la descripción visual de las actividades que implica el proceso de modelado de gráficos tridimensionales en Blender 3D. Y se presenta en la Figura 3.12 el modelo realizado de acuerdo a la referencia del diagrama de flujo anterior.

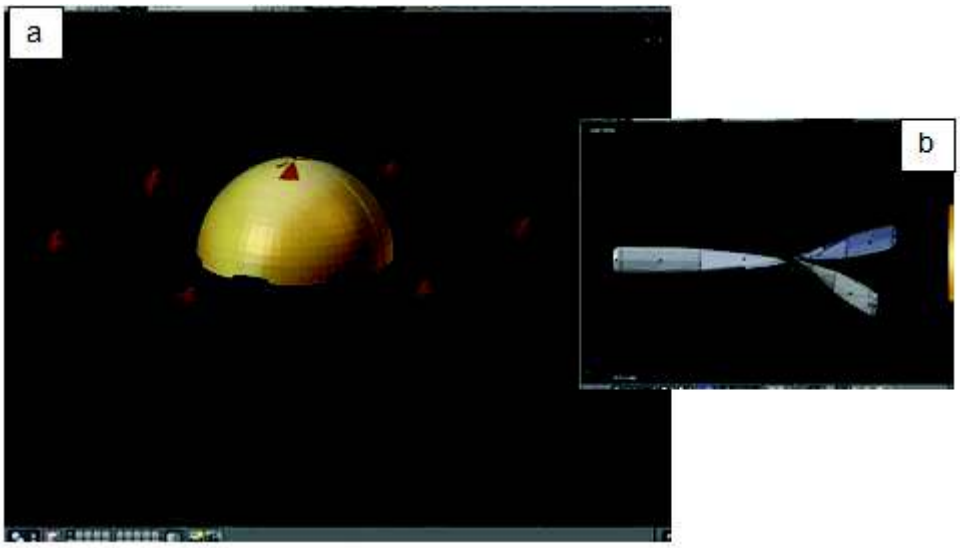

Figura 3. 12. Modelo 3D de hexacóptero desarrollado en Blender 3D: a) cuerpo del hexacóptero, b) hélice

Para implementar el modelo realizado, en Unity 3D, el procedimiento es sencillo. Se debe abrir la carpeta donde se guardó el archivo BLEND, se selecciona todos los elementos existentes y se arrastra hacia la carpeta de destino en el motor 3D.

3.4 UNITY 3D

Unity 3D, es un motor gráfico 3D compatible con Windows y Mac OS, está orientado al desarrollo de juegos, aplicaciones interactivas, visualizaciones y animaciones en 2D y 3D, es decir, aplicaciones tanto educativas como de entretenimiento [36].

Las aplicaciones desarrolladas en este software pueden ser exportadas a diferentes tipos de plataformas como PC, Mac, Linux Standalone, iOS, tvOS,

Android, BlackBerry, Tizen, Xbox 360, Xbox One, PS3, PS Vita, PS4, Windows Store, WebGL, Samsung TV y la web usando el plugin “Unity web player” [36].

Unity ha tomado fuerza y ha tenido éxito al convertirse en guía y ofrecer recursos a desarrolladores y programadores independientes, facilitándoles el acceso a un poderoso motor gráfico 3D de manera gratuita con limitaciones mínimas de recursos, los cuales se activan con la obtención de la licencia al adquirir la versión profesional [36].

Al igual que cualquier producto en el mercado, Unity 3D tiene grandes competidores en el ámbito de desarrollo de videojuegos, sin embargo, Unity 3D va más allá de este tipo de aplicaciones, logrando posicionarse en el ámbito educativo y profesional. Este software presenta además la ventaja de poder ser utilizado en cualquier tipo de sistema operativo, y adicional a eso soporta la importación de archivos de programas como 3ds Max, Maya, Softimage, Blender, Modo, ZBrush, Cinema 4D, Cheetah3D, Adobe Photoshop, Adobe Fireworks, Allegorithmic, Sweet Home 3D, SketchUp [36].

De acuerdo al manual de Unity 3D [36], la compañía que distribuye el software tiene la visión de "democratizar el desarrollo de juegos", abarcando el mayor número de usuarios posibles y poniendo al alcance de sus manos contenidos interactivos en 2D y 3D para que puedan trabajar sobre ellos [36].

En general Unity trabaja con la distribución de escenas, en las cuales se ubican los objetos. Estos objetos necesitarán propiedades que pueden ser controladas mediante programación con el lenguaje de script [36].

El presente proyecto se basa en el diseño de un simulador de vuelo, desarrollado con la versión gratuita 5.3.2f1 de 32 bits de Unity 3D, con la importación de diseños 3D realizados en diferentes programas de modelado, diseños que incluyen ambientes semiestructurados y hexacópteros, que pueden observarse en las secciones 3.1, 3.2 y 3.3 de este documento.

A continuación se exponen los conceptos necesarios del software, junto con el desarrollo del simulador de vuelo.

3.4.1 INTERFAZ DE USUARIO [37]

La ventana principal del editor que se presenta en la Figura 3.13, está compuesta por varios paneles conocidos como Views. Hay varios tipos de Views y todas tienen funciones específicas, las cuales se describen a continuación.

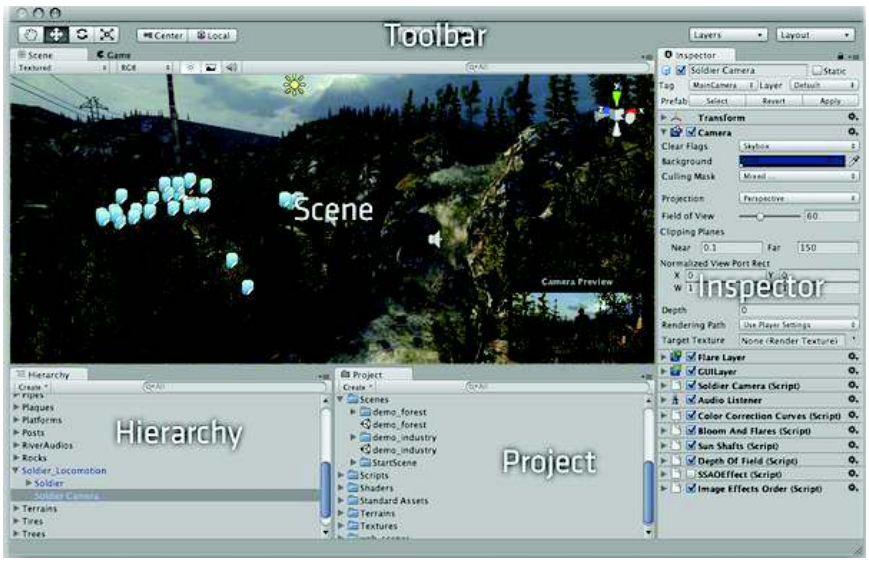

Figura 3. 13. Ventana principal del editor de Unity 3D [37].

El primer elemento de denomina administrador de objetos o vista del proyecto (project), que corresponde a la librería del proyecto. Aquí se encuentra todo el material de la aplicación, es decir, objetos 3D importados, texturas, audio, imágenes, scripts, etc., por lo que recomendable organizarlos por carpetas para facilitar el uso de cada elemento en el momento que se requiera.

Jerarquía (Hierachy) contiene los objetos (GameObject) de la escena actual. Cada elemento agregado directamente a la escena, será agregado también a la jerarquía.

Dentro de la vista jerarquía, Unity usa un concepto llamado Parenting. Para hacer que cualquier GameObject sea el hijo de otro, se arrastra el hijo deseado al padre deseado en la jerarquía. Un hijo va a heredar el movimiento y rotación de su padre. Se puede utilizar la flecha despegable del padre de un objeto para mostrar o esconder el hijo según sea necesario. En las Figuras 3. 14 y 3. 15 se observa un ejemplo de este concepto.

Figura 3. 14. Objetos no emparentados

Figura 3. 15. Objeto emparentado a otro

La barra de herramientas (Toolbar) proporciona acceso a las principales funciones, a la izquierda están las herramientas básicas de manipulación de objetos dentro de la escena, en la parte central los controles de reproducción de la escena y a la derecha se encuentra el acceso a la unidad de servicio y a la cuenta de usuario, seguido de un menú de visibilidad de las capas, y finalmente el menú Editor de diseño (que proporciona algunos diseños alternativos para las ventanas de edición, y le permite guardar su propio diseño).

La vista de escena (Scene View) es un sandbox interactivo, es decir, un ambiente libre donde se puede crear, se usa para seleccionar, posicionar, escalar o rotar ambientes, personajes, cámaras y todos los objetos que sean necesarios.

Esta vista cuenta con una barra de control, que permite encender o apagar el audio o la iluminación, activar el modo 2D y también controlar objetos, efectos de imagen y gizmos que se mostrarán.

La Vista del Juego (Game View) es reproducido por la cámara(s) existente en el proyecto, esta vista representa la aplicación finalizada. Por lo general es necesaria más de una cámara para tener una aplicación completa.

La vista Inspector muestra información detallada sobre el objeto seleccionado, incluyendo todo los componentes adjuntos y sus propiedades, permitiendo modificar su funcionalidad. Incluso variables de script pueden ser cambiadas sin la necesidad de modificar en sí mismo el script.

3.4.2 CONCEPTOS BÁSICOS

El elemento fundamental de Unity es el ***GameObject***, que es el nombre que se le da a cada objeto dentro de la aplicación. No obstante, estos elementos no hacen nada por sí mismos, sino que necesitan propiedades especiales antes de que puedan volverse un personaje, un ambiente, o un efecto especial.

Estos objetos se ubican en una ***escena*** en concreto. Las escenas ayudan a dividir la aplicación en niveles, diferentes pantallas de interacción, etc.

Los GameObject se crean a partir de ***Assets***, que son recursos de todo tipo (geometrías 3D, texturas, sonidos, etc.) junto a otros elementos propios creados en Unity (escenas, prefabs, scripts, shaders, etc.) [38].

Los GameObject se pueden considerar como contenedores o cajas vacías que pueden guardar diferentes piezas, las cuales llevan el nombre de ***Components***, estas piezas se agregan de acuerdo al comportamiento que se desee que tenga el GameObject [38].

En resumen los tres componentes básicos para el desarrollo de una aplicación son las escenas, los GameObjects y los Componentes, como se señala en la Figura 3. 16 [38].

Figura 3. 16. Componentes básicos de Unity

Para agregar un Component se puede usar el Component Browser, que puede ser activado con el botón Add Component en el inspector del objeto, como se ve en la Figura 3.17.

Esta herramienta permite navegar los components eficazmente por categoría y también tiene un cuadro de búsqueda que se puede utilizar para ubicar Components por su nombre. Se puede adjuntar cualquier número o combinación de Components a un solo GameObject [37].

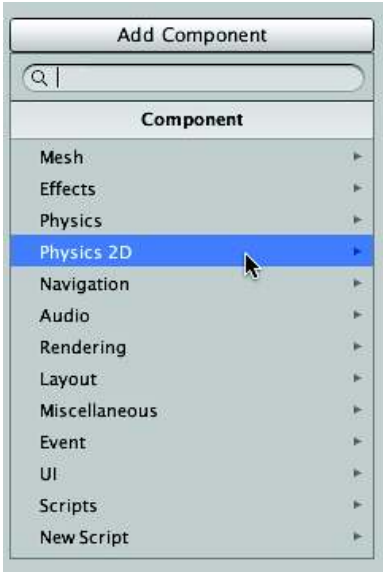

Figura 3. 17. Component Browser de Unity

En el proyecto uno de los Components más importantes que se asocia al GameObject, que este caso el hexacóptero, es un ***Script***, que modelará su comportamiento durante la evolución del simulador.

Son muchos los GameObject que se pueden desplegar en una escena, en la Figura 3.18 se observa la ventana de esto elementos, y a continuación se mencionan los más utilizados:

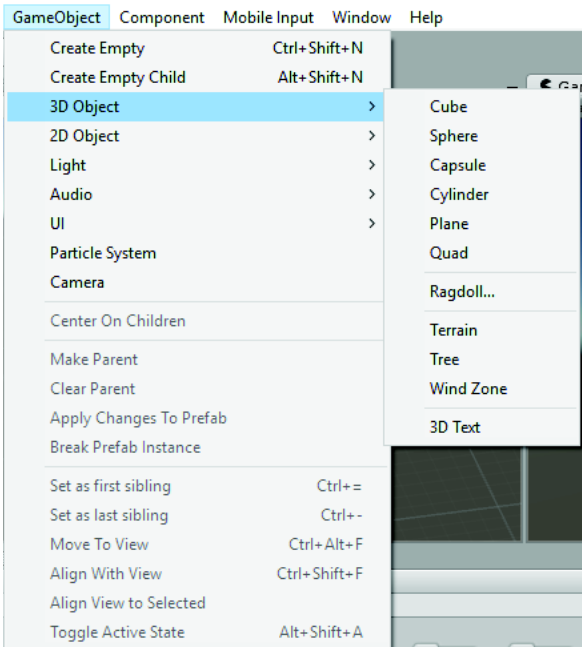

Figura 3. 18. Ventana de GameObject de Unity

- *Empty*: Se trata de un GameObject que únicamente contiene su componente Transform y que suele crearse para que se convierta, en padre de una colección de GameObjects [38].
- *Cubo, esfera, cápsula, cilindro, plano y quad*: Objetos sencillos creados por Unity 3D para formas sencillas, perfectos para desarrollar ejemplos sin necesidad de importar geometrías [38].
- *Luces*: Son una parte esencial de cada escena. Mientras meshes y texturas definen la forma y la apariencia de una escena, las luces (lights) definen el color y ánimo de su entorno en 3D. Generalmente se trabaja con más de

una luz en cada escena. Hacerlas trabajar juntas requiere un poco de práctica, pero los resultados pueden ser bastante sorprendentes [37].

- *Cámaras*: Además de la principal, que se añade por defecto en cada escena, se puede añadir varias cámaras, que se pueden activar/desactivar selectivamente o mantener de forma simultánea (por capas o renderizando en diferentes áreas de la pantalla) [38].
- *Otros*: elementos de interfaz (GUI), cloth, ragdolis, árboles, etcétera [38].

De igual manera existen muchos componentes que se pueden añadir a un GameObject, siendo los principales:

- *Transform*: Siempre está presente en un GameObject y almacena la posición, rotación y escala del objeto [38].
- *Mesh Filter*: Almacena la malla poligonal del GameObject [38].
- *Mesh Renderer*: Se encarga de definir como la malla del objeto va a ser renderizada en la posición que define Transform [38].
- *Collider*: Geometría que utilizará el GameObject cuando se tenga que detectar su colisión con otros objetos. Por razones de eficiencia el collider suele ser más sencillo que la geometría real del objeto [38].
- *RigidBody*: cuando es necesario que el objeto se encuentre sometido al motor físico, es decir, leyes físicas como la gravedad, reacción ante colisión, etc [38].
- *Sonido*: Una aplicación estaría incompleta sin algún tipo de audio, ya sea la música de fondo o efectos de sonido. El sistema de audio de Unity es flexible y poderoso. Puede importar la mayoría de formatos estándares de audio y tiene características sofisticadas para reproducir sonidos en un espacio 3D, opcionalmente con efectos como echo y filtración aplicadas. Unity también puede grabar audio de cualquier micrófono disponible de la máquina del usuario para uso durante la reproducción de la aplicación o para almacenamiento y transmisión [37].
- *Otros*: sistemas de partículas, animaciones, etcétera [37].

3.4.3 ESPACIO 3D

Unity utiliza el sistema de coordenadas de la "mano izquierda". Se conoce así por la regla mnemotécnica usada para recordar "hacia dónde apunta el eje Z" [39], una representación gráfica de esta regla se muestra en la Figura 3.19.

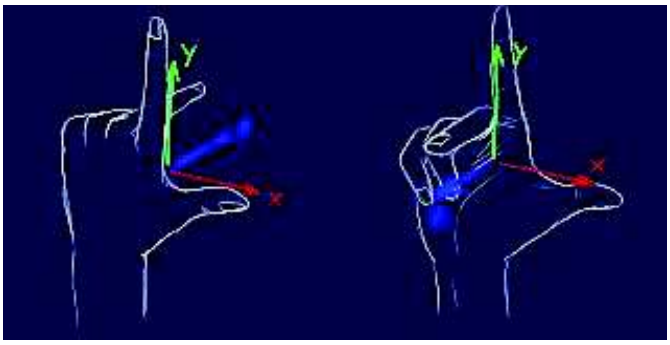

Figura 3. 19. Sistema de Coordenadas de Unity, usando la regla de la “mano derecha” [39]

El espacio 3D en Unity viene siendo el lugar donde se ubican los objetos, es decir la vista de escena (Scene View). El sistema de coordenadas de este mundo 3D es precisamente “el sistema de coordenadas del mundo” o world space como se le conoce en Unity, y los ejes de este sistema se referencian en la esquina superior derecha donde se puede ver un pequeño cubo con 3 conos de colores apuntando en las direcciones X (rojo), Y (verde) y Z (azul), como se distingue en la Figura 3. 20 [14].

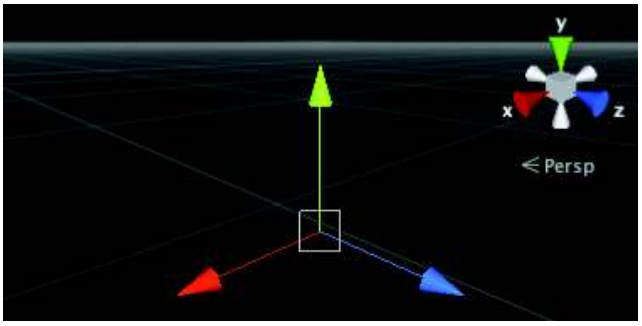

Figura 3. 20. Sistema de Coordenadas del mundo en Unity (World Space)

Como se mencionó antes, los ejes del sistema de coordenadas global son dados por el cubo en la esquina superior derecha de la vista de escena. Sin embargo cada objeto como tal tiene su propio sistema de coordenadas, lo cual se conoce como el sistema de coordenadas local. Para ver cuáles son los ejes del sistema de coordenadas local de un objeto, basta con seleccionar al objeto, y seleccionar la opción “local” en la parte superior de los modos de edición en Unity, en la Figura 3. 21 se aprecia este sistema de coordenadas [14].

Para cambiar el sistema de coordenadas de un objeto, basta con modificar cualquiera de sus propiedades del componente Transform (position, rotation o scale) de sus valores por defecto, los cuales son: Position = (0,0,0); rotation = (0,0,0), y scale = (1,1,1) [14].

Es esencial tener presente estos conceptos de coordenadas globales y coordenadas locales, para el desarrollo de la programación del comportamiento de los hexacópteros, ya que de ello dependerá el cálculo de los Ángulos Euler entre otros parámetros del modelo [14].

Figura 3. 21. Coordenadas locales de un objeto [14]

3.4.4 FÍSICA 3D

Para poder hacer uso del motor físico del que dispone Unity 3D, se recurre al componente RigidBody, el cual pone a un objeto bajo el control de las fuerzas físicas. Con un Rigidbody adjunto al GameObject, este responderá inmediatamente a la gravedad. Si adicional a esto también se le adjunta un componente Collider entonces el objeto se moverá por colisiones entrantes [38].

En un objeto que tiene activadas las propiedades físicas, no se debe manejar los parámetros del componente Tranform como lo son la posición y rotación, desde scripts, en este caso se debe aplicar fuerzas para mover el objeto y permitirle al motor físico calcular los resultados [38].

También se tiene la posibilidad de que un GameObject tenga un RigidBody pero no esté sometido por el motor físico. Este tipo de movimiento que se produce desde el script y no como resultado de fuerzas físicas es conocido como movimiento kinematic. Precisamente el componente RigidBody cuenta con la propiedad llamada Is Kinematic que lo quitará del control del motor de física y le permite moverlo cinemáticamente mediante un script [38].

Se puede activar o desactivar estas propiedades desde el código de un script, teniendo en cuenta que esto implica una carga de rendimiento y se debe procurar usar escasamente [38].

También se tiene la posibilidad de adjuntar solamente el componente Collider, excluyendo el RigidBody, si lo que se desea es tener elementos estáticos, como paredes, con los que los otros objetos pueden colisionar y rebotar [38]. Este tipo de objetos son los que se utilizan como obstáculos para el simulador de vuelo implementado.

Entonces se puede diferenciar dos tipos de colliders que son los estáticos y los dinámicos. Los primeros corresponden a objetos que no tengan adjunto un

RigidBody y como consecuencia no se moverán en respuesta a las colisiones, y los dinámicos son los tiene adjunto un RigidBody [37].

En la Figura 3. 22 se observa el componente RigidBody y las propiedades que se pueden controlar, las más importantes se indican a continuación [38]:

- La masa (Mass): El peso del objeto en kilogramos.
- Rozamiento (Drag): Cuánta resistencia se tiene del aire al moverse por fuerzas.
- Rozamiento angular (Angular Drag): Cuánta resistencia se tiene del aire al moverse por fuerzas de torsión (torque).
- Use Gravity: Si el objeto está o no bajo el efecto de la gravedad.
- Is Kinematic: Si el RigidBody es o no cinemático.

De forma adicional, se puede definir materiales físicos (Physic Material) para ajustar la fricción y el efecto de rebote entre objetos que colisionan [38], siempre de acuerdo con los requerimientos de la aplicación.

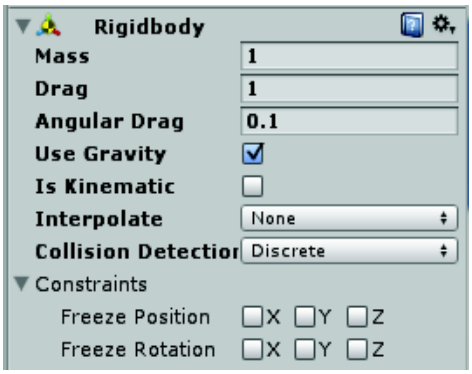

Figura 3. 22. Componente RigidBody de Unity

3.4.5 SCRIPTING

Los scripts permiten la definición de la lógica de la aplicación. El funcionamiento es muy sencillo, primero se define el comportamiento deseado en un script y este se agrega como un componente al objeto para el cual se desea asignar ese comportamiento, incluso se puede agregar un script a un objeto vacío [36].

Existen 3 funciones básicas para los scripts [36]:

- Función Update ( ): función que define el bucle principal del script por lo cual se ejecuta una vez por cada frame que se renderice.
- Función Awake ( ): Las variables definidas dentro de esta función se inicializarán justo antes de que empiece a correr la aplicación. Esta función se ejecutará una sola vez durante el tiempo que dure la instancia del script.
- Función Start ( ): Esta función es muy parecida a “Awake( )” pero tiene una diferencia vital; solamente se ejecutará si la instancia del script está habilitada. Esto permitirá retardar la inicialización de algunas variables de estado. También se debe tener presente que la función “Start( )” siempre se ejecutará después de que haya terminado la función “Awake()”.

Una función adicional a las básicas y también de mucha importancia en el simulador debido a sus características es FixedUpdate( ), esta función se llamará en intervalos regulares y todas las actualizaciones físicas (paso del motor físico) se llevarán a cabo tras el FixedUpdate. Por tanto, es el sitio perfecto para todo lo relacionado con cambios en el RigidBody del GameObject, es decir, en la aplicación de fuerzas [36].

El lenguaje de programación que se utilizó en este proyecto es C#. Unity también permite el uso de Javascript, pero es C# el lenguaje preferido en estos momentos por los desarrolladores por su potencia y cualidades. C# es un lenguaje estándar fácil de aprender si se tienen conocimientos previos de Java o C++. El dominio del lenguaje permite hacer más y mejores cosas mediante los scripts [36].

La escritura básica de un script C# de Unity 3D se observa en la Figura 3. 23 [40].

Figura 3. 23. Estructura básica de un Script [40].

A continuación se describen elementos, características y funciones necesarios para la programación de un script en C#.

- Ciclo del Juego [40]

  - El elemento principal de la arquitectura del motor es el ciclo del juego, el cual se encarga de dibujar y actualizar los objetos de la escena actual, siempre y cuando la escena este actualmente activa.
  - Delta time: indica el tiempo transcurrido desde la iteración anterior, se usa para hacer que la velocidad de la aplicación sea independiente en frames/segundo, para que sea en relación al tiempo y no a frames, actualizando la escena adecuadamente. Para poder acceder a este elemento se usa la función Time.deltaTime.

- Acceso a las propiedades del objeto [40]

Desde el código del script se accede a las propiedades del GameObject al que pertenece, con la función GetComponent(). Un ejemplo se puede observar en la Figura 3. 24.
Si el objeto no dispone del componente correspondiente, este método devuelve null.

Los componentes que se usan frecuentemente con esta función son:

- Transform: Transformación del nodo en la escena.
- Render: Permite cambiar el material del objeto.

- AudioSource: Acceso al componente Audio Source.
- Camera: Permite configurar la cámara.
- Light: Permite configurar una fuente de luz.

```
void Start () {
    GetComponent<Transform>().position = new Vector3(0,0,5);
}
```

Figura 3. 24. Inicialización del nodo en una posición [40]

El script como tal, se le considera como un componente, al igual que los que incorpora el motor. Por lo tanto también se puede acceder a un script desde otro script. Un ejemplo de esto se muestra en la Figura 3. 25.

Figura 3. 25. Accediendo a un script desde otro script [40]

- Tipos de Variables [41]

Por convención de nomenclatura, las variables en un script de Unity empiezan con una letra minúscula. Las variables se usan para almacenar información sobre cualquier aspecto de un estado de la aplicación, y pueden ser de varios tipos.

Las variables tienen que ser declaradas de la siguiente manera en lenguaje C# [41]:

TipoVariable NombreVariable = valor;

- Variables Públicas: permiten comunicarse entre los Scripts del mismo GameObject u otros GameObjects (Si se publica en Inspector)

```
public String DeoValor = "Hola a todos";
```

- Variables Públicas Ocultas: para crear variables públicas y estas no sean mostradas en el inspector, se debe escribir esta línea delante de la variable que se desea no mostrar y si después hay más variables definidas si se mostraran en el inspector: [System.NonSerialized].

- Variables Privadas: permiten comunicarse entre los Scripts del mismo GameObject u otros GameObjects (No se publica en Inspector).

```
private String DeoValor = "Hola a todos";
String DeoValor = "Hola a todos";
```

- Variables Globales: permiten crear variables globales para que se puedan comunicar entre todos los Scripts de la escena (No se publica en Inspector).

```
static Strign DeoValor = "Hola a todos";
```

- Creación y destrucción de objetos [40]

Para la creación de un nuevo objeto se usa la función instanciar, esto se lo realiza mediante la clonación del objeto proporcionado. Un ejemplo claro se puede observar en la Figura 3. 26.

```
Proyectil clone = Instantiate(proyectil) as Proyectil;
Proyectil clone = Instantiate(proyectil, posicion,
                  Quaternion.identity) as Proyectil;
```

Figura 3. 26. Instanciación de la variable proyectil [40]

Cuando se destruye un objeto, éste también se elimina de la escena, y se lo realiza con la función Destroy( ) [40], como se observa en la Figura 3. 27.

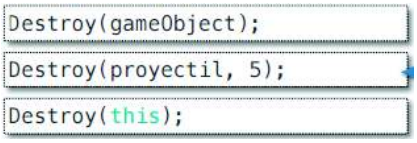

Figura 3. 27. Uso de función Destroy() [40].

Ambos pueden aplicarse a cualquier tipo de objeto.

- Detección de tecla [40]

Esta es una función muy útil en la programación de Unity. A continuación se muestra un ejemplo de código en C#, donde se utiliza la función GetKeyUp (), para realizar una acción cuando la tecla O deja de ser presionada.

```
if (Input.GetKeyUp (KeyCode.O)) {
    //Realizar acción       }
```

Si lo que se desea es realizar la acción cuando la tecla es presionada se puede utilizar la función GetKey(), sin embargo, no es lo más recomendable, ya que el uso de esta función crea el efecto de rebote, es decir, se ejecutará ilimitadamente mientras la tecla esté presionada.

3.4.6 **INPUT** [37]

Para la simulación del vuelo de los hexacópteros, hay que tener en cuenta que Unity soporta dispositivos de entrada convencionales como teclado, joypad, joystick, etcétera, pero también pantallas táctiles y además tiene la capacidad de detección de movimiento de dispositivos móviles.

Adicionalmente Unity puede hacer uso del micrófono del computador y la cámara web para entrada de datos de audio y video.

En el caso del proyecto desarrollado se usa el teclado y un Joystick con fuerza de realimentación, cuyo funcionamiento no responde al sistema convencional de

entradas, debido a su característica de Force Feedback necesita el uso de un toolkit específico, el cual se expondrá más adelante. Estos dos tipos de entradas pueden ser seleccionadas por el usuario mediante un menú.

Para el uso del teclado se usa la herramienta Input Manager, en donde es posible configurar y crear los ejes y botones necesarios para la aplicación, la ventana de esta herramienta se observa en la Figura 3. 28.

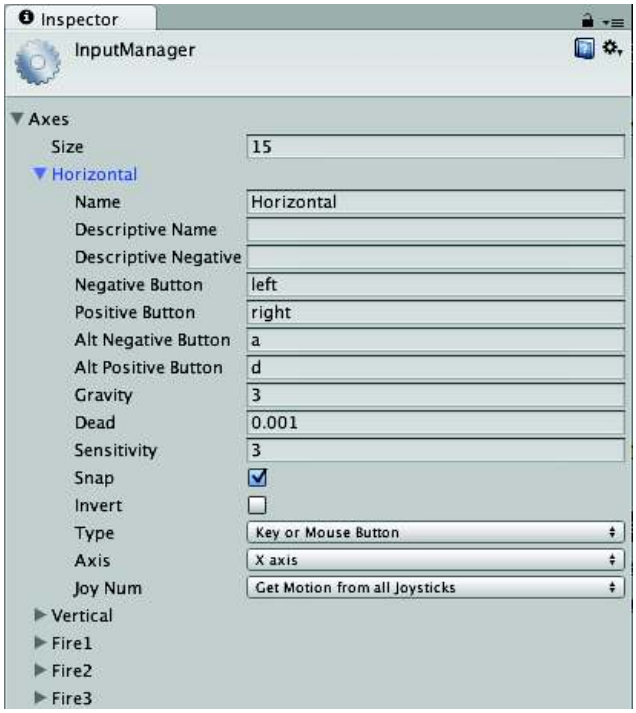

Figura 3. 28. Herramienta Input Manager de Unity 3D

Una vez creados los ejes necesarios para el simulador, que son: Horizontal, Vertical, Mouse X y Mouse Y, como se observa en la Figura 3.29, se puede acceder a ellos desde el script que controlará comportamiento de los hexacópteros, por su nombre como se indica en la Figura 3. 30. Teniendo en cuenta que los límites de los ejes son -1 y 1 y el valor neutral es 0, información importante durante la programación.

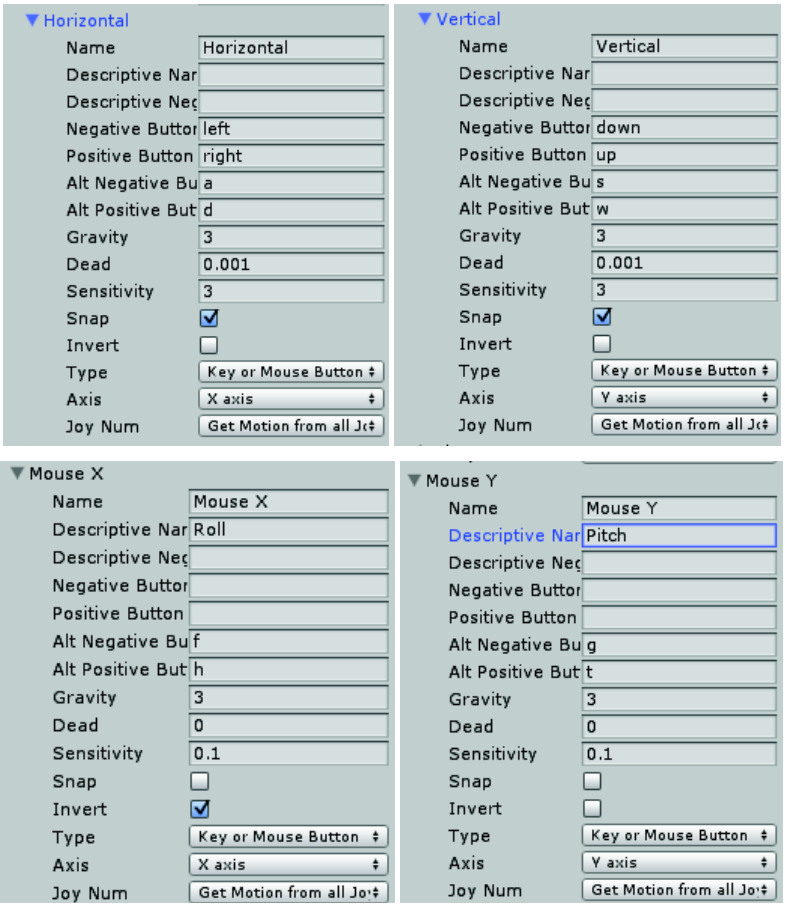

Figura 3. 29. Ejes usados en el simulador de vuelo

```
acelerador = Input.GetAxis("Vertical");
Angulo_Yaw_in = Input.GetAxis("Horizontal");
Angulo_Pitch_in = Input.GetAxis("Mouse X");
Angulo_Roll_in = Input.GetAxis("Mouse Y");
```

Figura 3. 30. Configuración de ejes de entrada.

Es posible crear ejes múltiples con el mismo nombre. Esto hace que sea posible asignar más de un dispositivo input al nombre de un eje.

3.4.7 CREACIÓN DE MENÚS [37]

Unity cuenta con sistemas específicos para la creación de menús, que son el UI y el GUI Legacy, el primero ha tomado mayor fuerza que el segundo, debido a las mayores opciones de personalización que tiene, sin embargo el uso de este tipo

de menú implica un mayor uso de espacio en el proyecto, razón por la que se optó por el sistema GUI Legacy, ya que es funcional para el tipo de aplicación que se desarrolló, sin ocupar espacio innecesario.

El proyecto cuenta con distintos tipos de menú que fueron implementados para la interacción de los diferentes elementos del simulador de vuelo.

Iniciando el simulador se podrá visualizar el primer menú, que cuenta con un botón de inicio a la teleoperación de los hexacópteros, un botón de ayuda que desplegará un manual de las instrucciones necesarias para poder manipular el programa, y finalmente un botón que brindará información acerca de las características de la aplicación. Este menú se observa en la Figura 3. 31.

Figura 3. 31. Menú de Inicio a simulador de vuelo

Para la implementación de este menú y todos los que contiene el programa, es importante mencionar que los controles del Unity GUI usan la función OnGUI(). Esta función es llamada cada cuadro siempre y cuando el script contenido este habilitado, tal como funciona la función Update().

Este tipo de controles es sencillo, a continuación se los nombra y explica su estructura.

3.4.7.1 Anatomía de un control [37]

Un control GUI debe ser declarado con tres elementos y de la manera que a continuación se muestra:

Tipo (Posición, Contenido)

El tipo de control se selecciona de acuerdo a la función que se le desee asignar, este es declarado al llamar una función en la clase GUI o GUILayout de Unity, en siguiente sección se detallará cada tipo de control que puede usarse.

La posición es el primer argumento en cualquier control GUI. El argumento en sí mismo es proporcionado con una función Rect () que define cuatro propiedades: la posición en el eje x, la posición en el eje y, el tamaño en el eje x (ancho) y el tamaño en el eje y (altura). Todos estos valores corresponden a valores de píxel. Todos los controles GUI trabajan en Screen Space, que es la resolución del reproductor publicado en píxeles. Para utilizar esta propiedad hay que tener en cuenta que el sistema de coordenadas toma su origen en la parte superior izquierda de la pantalla.

El contenido, es el segundo argumento para un Control GUI, y este representa el contenido actual para ser mostrado por el Control. El mismo que puede ser texto, imágenes o ambos. Todo dependerá de los conocimientos del programador.

Un ejemplo de la anatomía de un control GUI es la siguiente:

GUI.Label(Rect(10, 20, 300, 100),”Texto”);

El tipo de control es Label. La posición está dada por la función Rect, que define un Rectángulo que comienza en las coordenadas: 10,20 y terminan en las coordenadas 310,120. Y finalmente el contenido que se mostrará en la pantalla es de tipo string.

3.4.8 REALIMENTACIÓN DE FUERZA

Para la realimentación de fuerza que contiene el simulador de vuelo se utilizó el paquete Force Feedback Toolkit v1.2 Released, el cual se puede descargar desde el foro publicado por el propietario, disponible en la sección de foros de la página

web de Unity [42], de manera gratuita. La herramienta facilita la integración de esta característica a la aplicación, además adjunta un manual de instrucciones que ayuda a la comprensión de su funcionamiento, mediante ejemplos y descripciones de las funciones que contiene.

Para el uso de este paquete se requiere trabajar en el sistema operativo Windows de 32 bits, y conjuntamente se necesita el complemento DirectX 9, que le dará apoyo en la parte gráfica que requiere el toolkit.

El complemento DirectX9 es una interfaz de programación de aplicaciones, su tarea es la comunicación entre componentes de software, permitiéndole a Unity 3D acceder a una serie de funciones multimedia de software [43], necesarios para la ejecución de la aplicación.

Una vez instalado el complemento y descargado el paquete de realimentación de fuerza, se lo importa a Unity 3D. El proceso es sencillo y se lo observa en la Figura 3.32.

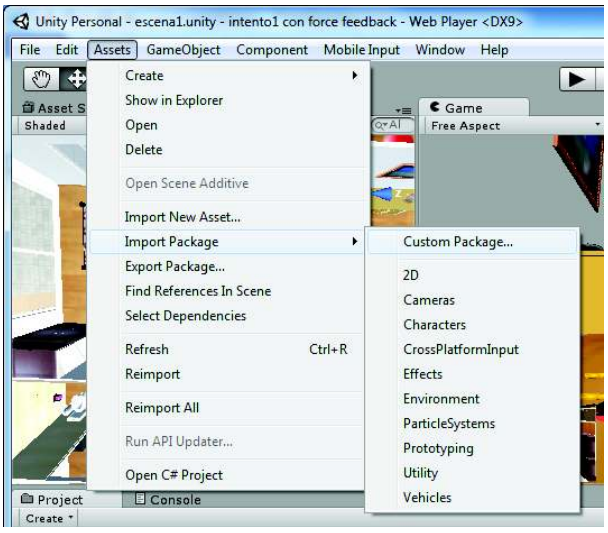

Figura 3. 32. Importación del toolkit a Unity 3D

Cuando se completa la carga, ya se tiene acceso a las funciones que brinda el paquete, que además de la realimentación de fuerza, incluye el reconocimiento de dispositivos de entrada, y manejo de los mismos.

La manera de tener acceso a las funciones del toolkit se detallan a continuación [44]:

En la función Start () se debe declarar la variable wheelMenu:

```
WheelMenu wheelMenu = Camera.main.GetComponent<WheelMenu>();
```

Y al usar cualquiera función se la llamará así:

```
wheelMenu.dinput.Función ();
```

La función utilizada para la realimentación de fuerza es:

```
SetForce2DCoordinate(float x, float y)
```

Esta función permite establecer la dirección de la fuerza del controlador en dos ejes el X y el Y.
La fuerza en el eje X, especifica la fuerza horizontal.
La fuerza en el eje Y, especifica la fuerza vertical.

En cualquiera de los dos casos un valor negativo invierte la dirección de la fuerza. Los intervalos válidos para ambos ejes son de -100 a 100. Cualquier valor fuera del rango válido se ajusta a los límites establecidos.

Otra función utilizada es la que permitirá la lectura de los ejes y botones del Joystick, y es la siguiente:

```
getInputStatus(int inputId)
```

La cual retorna el valor actual de un eje o botón del controlador, indicado en la variable inputId de tipo int (entera).

Para conocer la manera de definir los ejes o botones es necesario acudir al manual de usuario del toolkit [44], a continuación se muestra la definición de cada uno en la Figura 3.33.

```
switch (inputId)
{
    case 100: retVal = "XAxis"; break;
    case 101: retVal = "YAxis"; break;
    case 102: retVal = "ZAxis"; break;
    case 103: retVal = "XRot"; break;
    case 104: retVal = "YRot"; break;
    case 105: retVal = "ZRot"; break;
    case 106: retVal = "Slider0"; break;
    case 107: retVal = "Slider1"; break;
    case 108: retVal = "POV0"; break;
    case 109: retVal = "POV1"; break;
    case 110: retVal = "POV2"; break;
    case 111: retVal = "POV3"; break;
    case 112: retVal = "ForceSlider0"; break;
    case 113: retVal = "ForceSlider1"; break;
    case 114: retVal = "ForceX"; break;
    case 115: retVal = "ForceY"; break;
    case 116: retVal = "ForceZ"; break;
    case 117: retVal = "TorqueX"; break;
    case 118: retVal = "TorqueY"; break;
    case 119: retVal = "TorqueZ"; break;
    case 120: retVal = "VelS0"; break;
    case 121: retVal = "VelS1"; break;
    case 122: retVal = "VelX"; break;
    case 123: retVal = "VelY"; break;
    case 124: retVal = "VelZ"; break;
}
```

Figura 3. 33. Definición de ejes y botones del paquete de realimentación de fuerza [44]

3.4.8.1 Joystick Force Feedback

Para la teleoperación del hexacóptero se utilizó el Joystick Microsoft® SideWinder® Force Feedback 2, este modelo se puede apreciar en la Figura 3.34.

Unity 3D soporta la tecnología Force Feedback, que proporciona este joystick, sin necesidad de la instalación del software del mismo, basta con conectarlo para que el software lo reconozca.

La información que se encuentra en línea, acerca de este modelo, es escasa, ya que el producto es antiguo, sin embargo, se debe mencionar que es un controlador muy sofisticado y costoso. Este dispositivo fue introducido

idealmente para simuladores de vuelo [25], sin embargo Microsoft dejó de fabricarlos, por lo que la adquisición del mismo es complicada.

Figura 3. 34. Joystick Microsoft® SideWinder® Force Feedback 2 [25]

Una de las ventajas que presenta el joystick es que cuenta con puerto USB, así que la conexión a equipos modernos no representa una dificultad. Las principales características de este controlador son [45]:

- Adaptador A/C: Integrado en el joystick.
- Se pueden programar hasta 16 funciones con SideWinder® Game Control Software.
- 8 botones disponibles.
- Funciona bajo Windows 98 o superior.

Como otros dispositivos de este tipo, este controlador está construido de materiales resistentes, brindando una sensación de solidez. En la base del joystick se localizan los mecanismos que provocan todos los efectos en la palanca y es totalmente ergonómico [45].

3.4.8.2 Cálculo de la realimentación de fuerza

Para realizar el cálculo de la fuerza que aparece en la palanca de mando como contraposición al movimiento ejercido por el usuario, cuando la aeronave está cerca de un obstáculo, se toma en cuenta la distancia entre los dos objetos.

El método escogido para el cálculo de esta distancia, como se indicó antes, es la creación de fuerzas ficticias alrededor del hexacóptero, creando una zona de repulsión hacia los obstáculos que se encuentren a la distancia establecida en la programación, de acuerdo a las dimensiones de la aeronave.

Las direcciones que toman las fuerzas ficticias que percibe el usuario a través del controlador, se establecen mediante la integración de ocho rayos alrededor del hexacóptero, de los cuales, cuatros están ubicados en los ejes positivos y negativos de X y Y, y los otros cuatro corresponden a la suma vectorial de los primeros, como se puede observar en la Figura 3. 35.

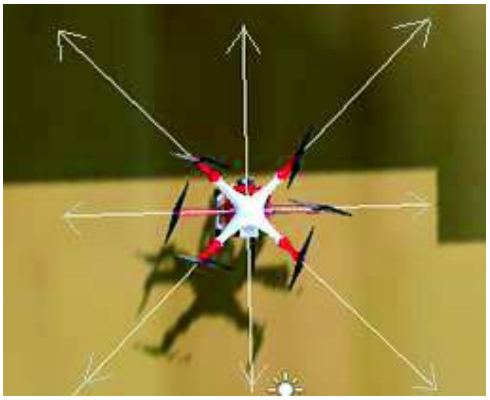

Figura 3. 35. Direcciones de las fuerzas ficticias para la evasión de obstáculos

Es importante comprender la manera en que se relaciona la fuerza de realimentación con la distancia entre el hexacóptero y el obstáculo, pues a medida que el hexacóptero se acerca más al objeto estático, es decir, cada vez que la distancia es menor, esta fuerza aumentará.

Con este preámbulo, se puede concluir que la fuerza de realimentación debe ser inversamente proporcional a la distancia. Por lo que se ha establecido una

correspondencia entre ambos parámetros, mediante la propuesta del algoritmo (3.1) que más adelante se explica detalladamente.

$$F = 87 * \frac{1}{2 + \log\left(\frac{d}{3}\right)} \tag{3.1}$$

Donde F es la fuerza de realimentación que aparece en el joystick y d es la distancia entre el hexacóptero y el obstáculo.

El algoritmo propuesto en (3.1), implementa la técnica de fuerzas ficticias planteada previamente, y es ejecutado por el simulador de vuelo, mediante la programación del mismo. El código donde se incluye el algoritmo formulado se encarga de medir la distancia entre el hexacóptero y el obstáculo encontrado, y realiza el cálculo de la fuerza en cada iteración, cumpliendo de esta manera el propósito de incrementar el valor de la fuerza a medida que disminuye la distancia.

La magnitud de la fuerza calculada mediante el algoritmo de (3.1), se encuentra limitada al intervalo [57.1, 35.1], valor dependiente de la distancia que se ubica en el intervalo [1, 9]. Considerando que se diseñaron hexacópteros con una medida de 3 unidades.

El algoritmo propuesto se desarrolló tomando en cuenta que los valores de la fuerza permitida en el dispositivo háptico van de 0 a 100, y que el usuario debe percibir una fuerza que le advierta la cercanía de objetos, mas no que le obligue a realizar el cambio de dirección. En la Figura 3. 36 se observa gráficamente el rango de valores de fuerza considerados como adecuados, en función de los valores de distancia existente entre el hexacóptero y el obstáculo estático.

Para poder tener una idea en el mundo real, de las dimensiones de los hexacópteros y distancias tomadas en cuenta en el entorno generado por el simulador de vuelo, es importante comprender las unidades en que se encuentran los valores de estas dimensiones. Para esto se multiplica su valor por 10, y de

esta manera se tiene un valor equivalente en centímetros, deduciendo entonces, que el simulador cuenta con hexacópteros de 30 centímetros de diámetro, y una zona de evasión de obstáculo de 90 centímetros de diámetro.

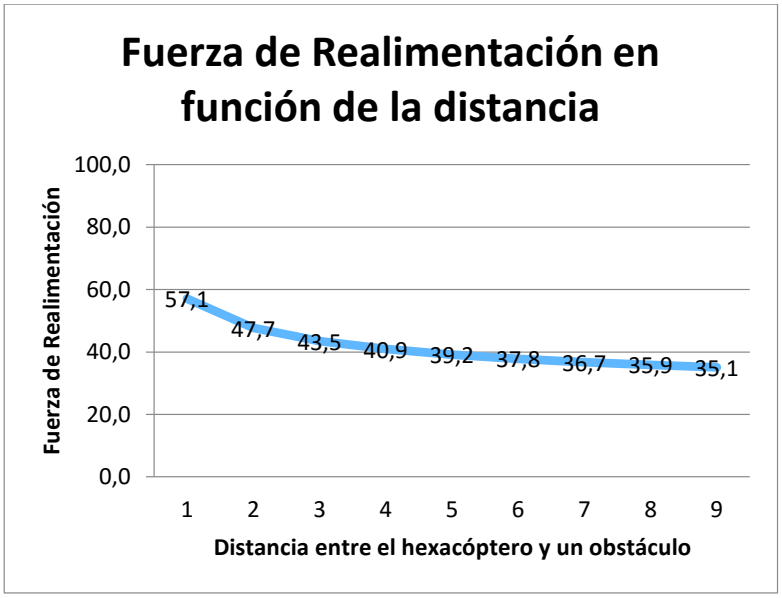

Figura 3. 36. Fuerza de Realimentación en función de la distancia

En la Tabla 3.1 se puede observar la correspondencia entre valores de la distancia y la fuerza calculada, graficada en la Figura 3.36 y calculados mediante el algoritmo (3.1).

Tabla 3.1. Valores de fuerza y distancia determinados a partir de la ecuación (3.1)

Distancia	Fuerza
1	57.1
2	47.7
3	43.5
4	40.9
5	39.1
6	37.8
7	36.7
8	35.9
9	35.1

En la Figura 3. 37 se distingue la dirección que se le asigna a la fuerza que es aplicada al controlador, cuando un obstáculo es detectado cerca del hexacóptero.

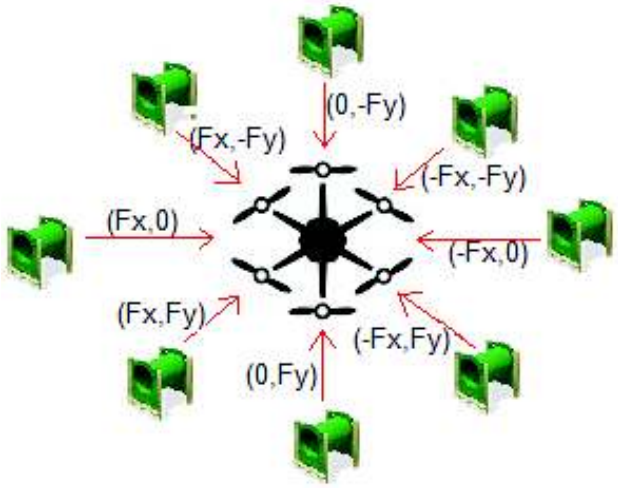

Figura 3. 37. Representación de las fuerzas de realimentación en las 8 direcciones posibles.

Se debe resaltar que el código de programación que contiene el algoritmo de cálculo de esta fuerza de realimentación, aplica una fuerza a la vez, con el objetivo de optimizar el paso de la aeronave por lugares angostos como paredes o pasillos.

3.5 ARQUITECTURA DEL SOFTWARE

La arquitectura del software es una parte fundamental para la comprensión de la aplicación, permitiendo tener una visión más clara de los procesos y subprocesos de la programación. A continuación se explica el funcionamiento de cada script, junto con su diagrama de flujo, que especificará los detalles algorítmicos para el desarrollo del proyecto.

3.5.1 BOTONINICIO.CS

Este script gobernará el menú principal y cuenta con dos funciones: Update y OnGUI. En la Figura 3.38 se observa el diagrama de flujo del script botoninicio.cs.

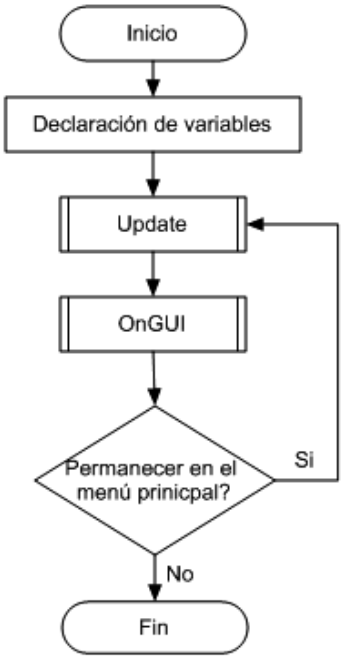

Figura 3. 38. Diagrama de flujo correspondiente al script botoninicio.cs

La función Update está encargada de mantener el reloj en modo de pausa, como de observa en el diagrama de flujo de la Figura 3.39.

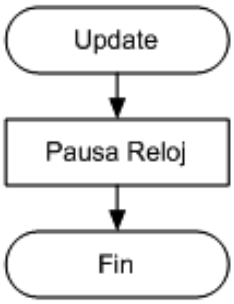

Figura 3. 39. Diagrama de flujo de la función Update del script botoninicio.cs

La función OnGUI del script botoninicio.cs, muestra en la pantalla las tres opciones del menú: Comenzar, Ayuda y Acerca de. Esta función se detalla en la Figura 3. 40.

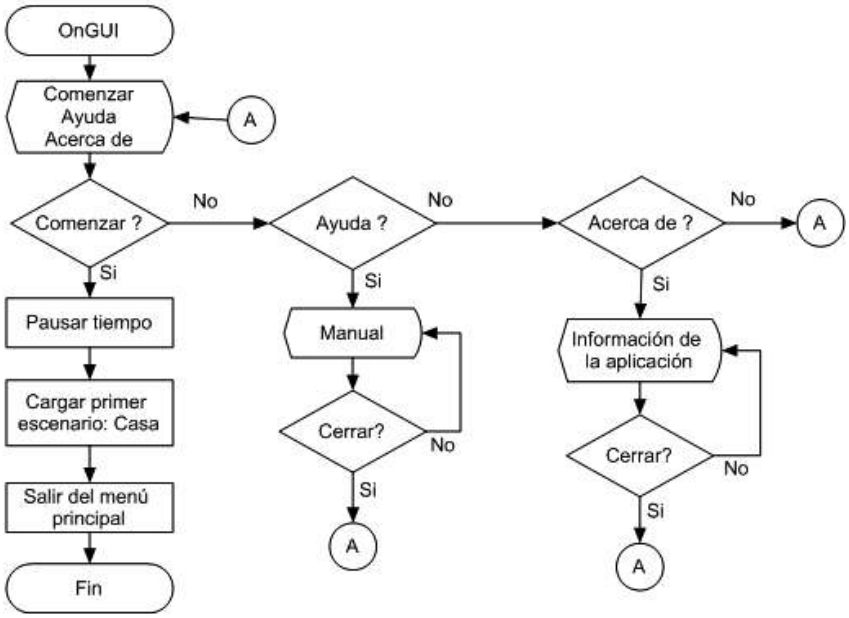

Figura 3. 40. Diagrama de flujo de la función OnGUI del script botoninicio.cs

La primera opción que es “Comenzar” permite iniciar la aplicación, cambiando de escena al primer escenario, la segunda opción que es “Ayuda” muestra el manual de instrucciones y la tercera opción que es “Acerca de” expone una ventana con información acerca del simulador. Las dos opciones finales cuentan con un interruptor que permite cerrar las ventanas, en cualquier momento.

3.5.2 MENU.CS

Este script maneja el menú que permite acceder a los hexacópteros, escenarios y tareas. Cuenta con las funciones: Start y OnGUI, y será siempre visible en la parte superior izquierda de la pantalla. En la Figura 3. 41 se observa el diagrama de flujo de este script.

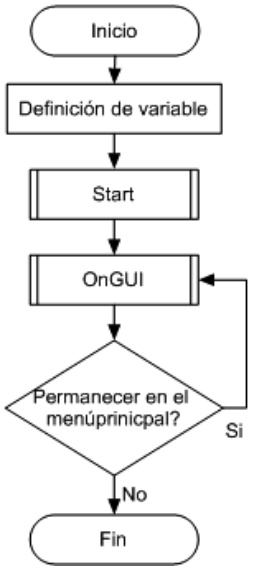

Figura 3. 41. Diagrama de flujo correspondiente al script menu.cs

La función Start de este script, se ejecuta al abrirse la escena de cualquiera de los escenarios, y en ella se definen, inicializan y desactivan componentes como el sonido y objetos como los hexacópteros. El diagrama de flujo de esta función se puede observar en la Figura 3.42.

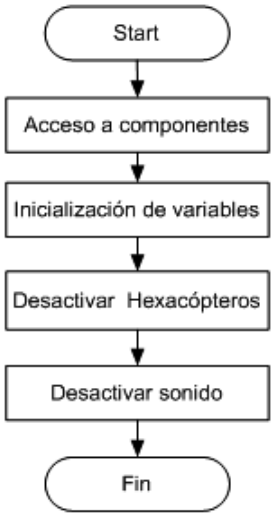

Figura 3. 42. Diagrama de flujo de la función Start del script menu.cs

La función OnGUI de este script, se detalla mediante el diagrama de flujo de la Figura 3.43, en el cual se observa que la pantalla muestra cinco opciones: Elegir Hexacóptero, Elegir Escenario, Volar, Menú Principal y Tarea.

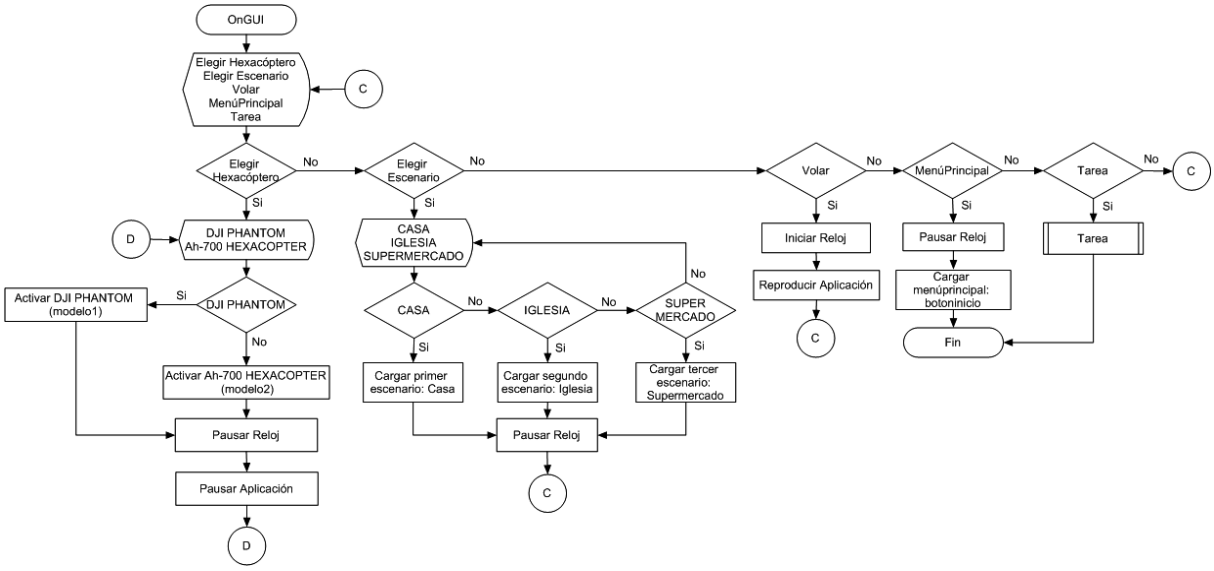

Figura 3. 43. Diagrama de la función OnGUI del script menu.cs

Para la primera opción se tiene disponible dos modelos de hexacópteros, para la segunda se tiene tres escenarios disponibles, con la tercera opción se activa el vuelo del hexacóptero seleccionado y se inicia el reloj, el cuarto botón permite regresar a la pantalla inicial y con el último botón se puede elegir la tarea a cumplir durante el vuelo.

El diagrama de flujo de la función Tarea se observa en la Figura 3.44.

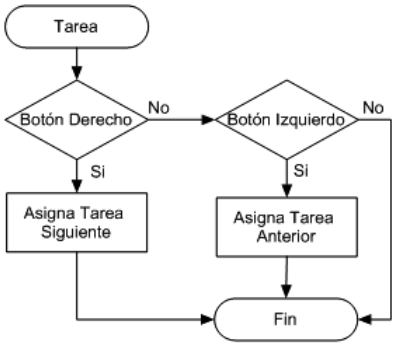

Figura 3. 44. Diagrama de la función tarea del script menu.cs

Esta función permite la selección de la tarea a realizar durante el vuelo del hexacóptero mediante dos botones: derecho e izquierdo, que irán mostrando los objetivos disponibles.

3.5.3 ENTRADAS.CS

El script entradas.cs cuenta con tres funciones: Start, Update y OnGUI, y permite seleccionar el controlador que comandará el hexacóptero durante su vuelo.

El diagrama de flujo de este script se observa en la Figura 3. 45.

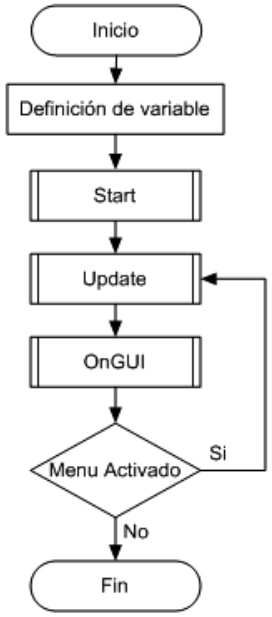

Figura 3. 45. Diagrama de flujo correspondiente al script entradas.cs

La función Start accede a componentes e inicializa variables correspondientes al paquete de realimentación de fuerza, como se puede observar en el diagrama de flujo de la Figura 3.46.

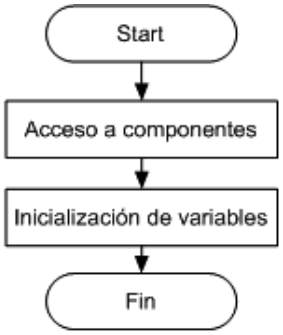

Figura 3. 46. Diagrama de flujo de la función Start del script entradas.cs

La función Update se encarga de la activación y desactivación de este menú, aquí se comprueba si la tecla P ha sido presionada o no.

En el diagrama de la Figura 3. 47 se explica en detalle esta función, que en el caso de que el menú este activado, la tecla P lo desactivará y viceversa. Además siempre que sea visible el menú, la aplicación estará en modo de pausa, deteniendo también el reloj.

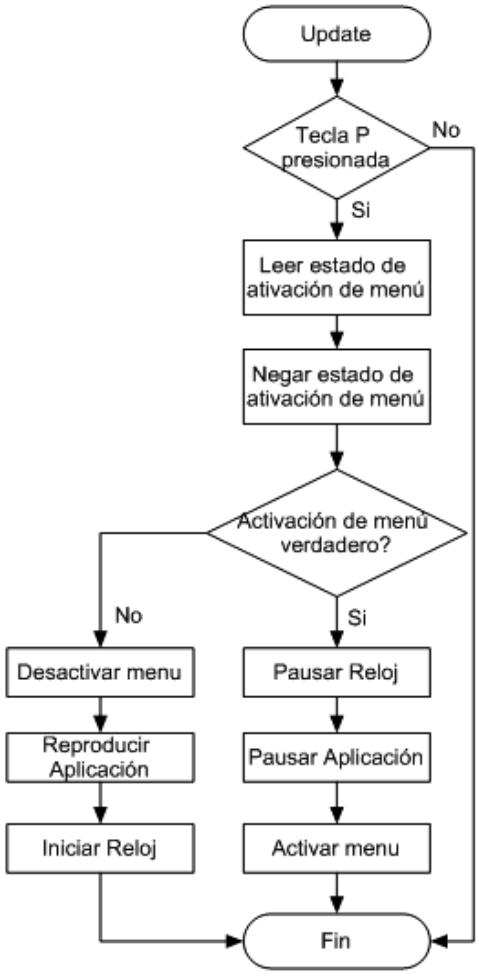

Figura 3. 47. Diagrama de flujo de la función Update del script entradas.cs

La función OnGUI comprueba si el menú está activado, si este es el caso, se presenta en la pantalla dos botones que son: teclado y joystick, el usuario podrá seleccionar el que prefiera presionando el botón correspondiente.

Si se elige el teclado, se desactiva el paquete de fuerza de realimentación, caso contrario, se activa el paquete para se pueda manejar el simulador mediante el Joystick.

En la Figura 3.48 puede observarse el diagrama de flujo de la función OnGUI.

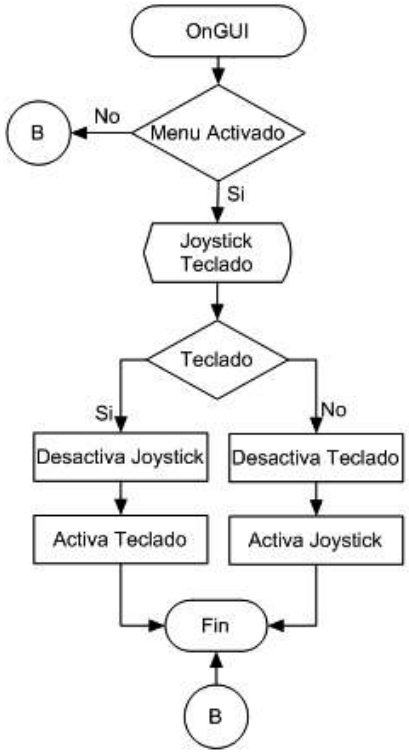

Figura 3. 48. Diagrama de flujo de la función OnGUI del script entradas.cs

3.5.4 SCRIPTS PARA LOS HEXACÓPTEROS

Finalmente se tiene los scripts que controlarán el comportamiento de los hexacópteros, para esto es importante tener en cuenta los parámetros del modelo publicados en [31] y presentados en la Tabla 2.1.

Estos parámetros son definidos al iniciar los scripts modelos1.cs y modelo2.cs, además cada script cuenta con una función Start y una FixedUpdate, como se observa en las Figura 3.49, 3.50 y 3.51, y se explican a continuación.

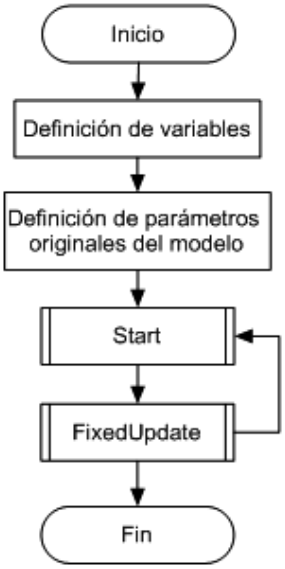

Figura 3. 49. Diagrama de flujo correspondiente al script de los modelos de los hexacópteros

La función Start accede a componentes e inicializa variables correspondientes al paquete de realimentación de fuerza y al sonido de vuelo de los hexacópteros, también define GameObjects como las cuatro cámaras: tres adjuntas a la aeronave y una que cumple la tarea de minimapa.

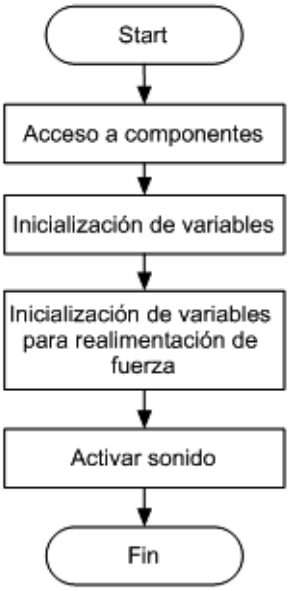

Figura 3. 50. Diagrama de flujo de la función Start del script que controla los modelos de los hexacópteros

La función FixedUpdate inicia verificando cuál de los dos controladores ha sido seleccionado por el usuario, para asignar las variables de entrada al que corresponda y activar o desactivar el paquete de fuerza de realimentación según sea el caso.

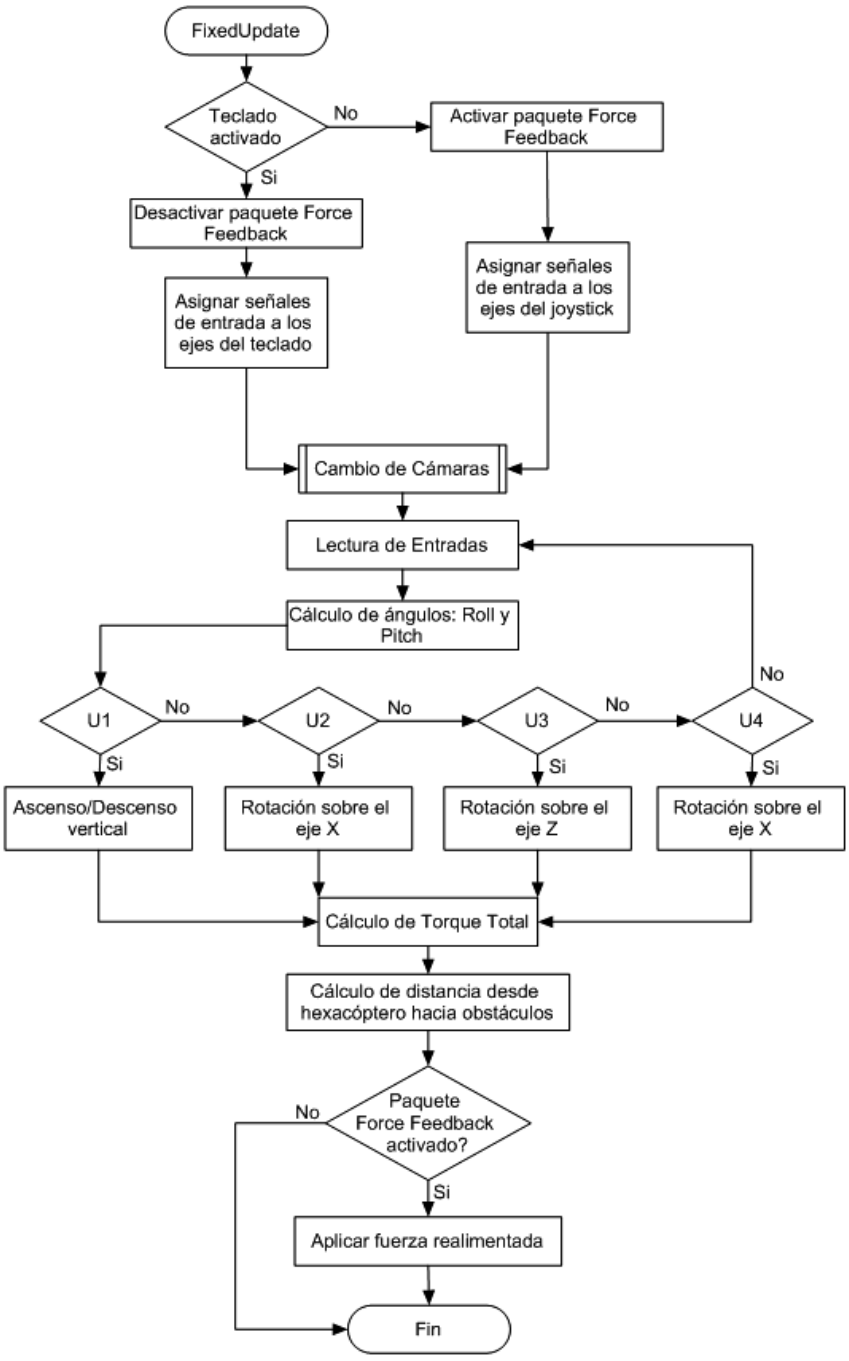

Figura 3. 51. Diagrama de flujo de la función FixedUpdate del script que controla los modelos de los hexacópteros

En esta función se encuentra el modelo dinámico de los hexacópteros, una vez que se lea las entradas, se inicia el movimiento deseado por el usuario y si la realimentación de fuerza está activada, se la envía al joystick cuando encuentre un obstáculo.

La función cambio de cámaras verificará si las teclas o botones correspondientes a cada cámara son presionados durante cualquier momento del vuelo del hexacóptero, y realizará el cambio que le corresponda. El diagrama de la figura 3.52 explica detalladamente el desarrollo de esta función.

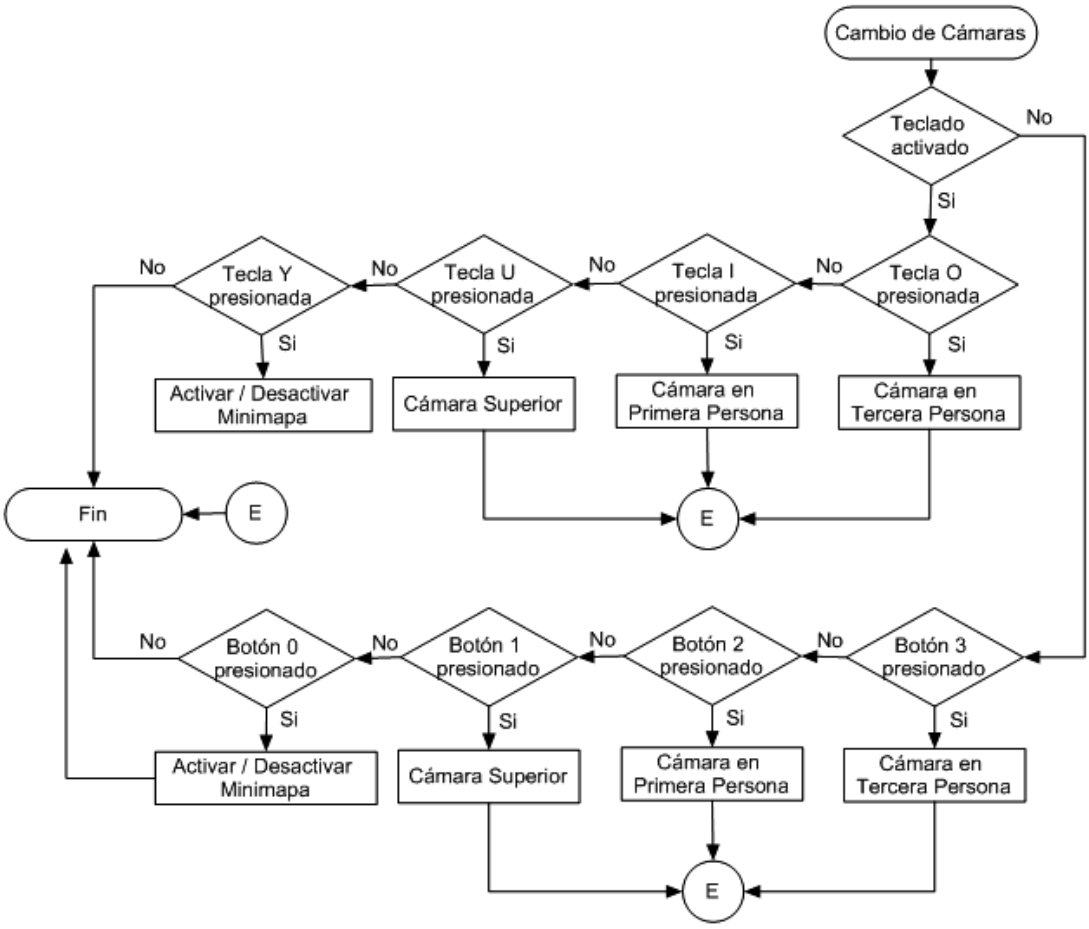

Figura 3. 52. Diagrama de flujo de la función Cambio de Cámaras del script que controla los modelos de los hexacópteros

Como se observa en la Figura 3.52, cada hexacóptero tiene disponible cuatro cámaras que a continuación se explican:

La cámara en primera persona brinda una vista del entorno desde la perspectiva del lente de la cámara integrada en cada hexacóptero, brindando al usuario una visión más realista durante la teleoperación de la aeronave.

La cámara en tercera persona, se encuentra detrás del hexacóptero y permite al usuario tener una mejor visión del entorno que lo rodea.

La cámara superior, como su nombre lo indica, se encuentra en la parte superior de la aeronave y su objetivo es brindar una mejor visión de la distancia del hexacóptero con los obstáculos del ambiente en el que se encuentra.

Las tres cámaras antes mencionadas están adjuntas a las aeronaves, es decir, utilizan el concepto de parentesco (parenting) señalado en la sección 3.4.1. Por lo que heredan el movimiento y rotación del hexacóptero al que corresponden.

Finalmente la cuarta cámara está ubicada en la parte superior del ambiente y abarca una visión completa del lugar, esta cámara es estática y su visión se coloca en la parte inferior izquierda de la pantalla, mostrando la ubicación y cumpliendo la función de un minimapa.

CAPÍTULO 4

PRUEBAS Y RESULTADOS

Una vez integradas cada una de las partes del sistema, se llevaron a cabo una serie de pruebas para evaluar su funcionamiento. Como parte inicial, se realiza la validación de menús, escenarios y mapas. A continuación se realizan pruebas del comportamiento de los hexacópteros durante su vuelo. Finalizando con la obtención de una realimentación de los usuarios acerca del manejo del simulador de vuelo, mediante una encuesta que complementa las evaluaciones realizadas previamente.

En el presente capítulo se presentará todas las pruebas mencionadas, los resultados y las respuestas de usuarios de acuerdo a las encuestas realizadas, con el objetivo de comprobar el adecuado desempeño del sistema.

4.1 VALIDACIÓN DE MENÚS

Como se mencionó en la sección 3.4.7 para la creación de menús, el simulador de vuelo inicia con un menú que dispone de tres botones como el que se observa en la Figura 4.1.

Mediante las pruebas de validación realizadas a cada uno de los elementos, se explica y muestra a continuación el correcto funcionamiento de cada uno de los elementos de este menú, el mismo que es comandado por el script “botoninicio.cs”

Figura 4.1. Menú de inicio del simulador de vuelo

Comenzando desde la parte inferior del menú se encuentra el botón “Acerca de” que al ser presionado lanza una pantalla con la información del tipo de aplicación y las características con la que cuenta, como se observa en la Figura 4.2.

Figura 4 2. Pantalla del botón "Acerca de “

El siguiente botón es el de “Ayuda” que en el momento de seleccionarlo abrirá una pantalla con un manual de instrucciones gráfico, donde está disponible el manejo de la aplicación desde el teclado o desde el Force Feedback Joystick, este manual puede distinguirse en la Figura 4.3.

Figura 4. 3. Manual de Instrucciones del botón “Ayuda”

Las dos pantallas anteriores cuentan con un toggle o interruptor, en la parte superior izquierda con el nombre “Cerrar” que permite salir de las pantallas en el momento deseado.

Finalmente el botón “Comenzar”, permite ingresar a la selección de hexacópteros, ambientes y tareas de vuelo, es decir, da paso al manejo de otros menús. Al presionarlo se abre por defecto el primer ambiente, el cual se ve en la Figura 4. 9. En donde se observa un nuevo menú, en la parte superior izquierda de la pantalla, el cual estará disponible durante toda la aplicación.

Una vez comprobado el funcionamiento adecuado del menú principal, se realizará las pruebas al siguiente menú, que cuenta con cinco opciones que se las puede apreciar en la Figura 4.4.

Figura 4. 4. Menú para elección de hexacópteros, escenarios y tareas de vuelo

Se inicia probando el funcionamiento del botón “Elegir Hexacóptero”, que al presionarlo despliega los dos modelos disponibles, como se observa en la Figura 4.5. Cada uno al ser seleccionado mostrará el modelo correspondiente dentro del ambiente que haya sido elegido. En las Figuras 4.14 y 4.15 se pueden observar los modelos.

Figura 4. 5. Funcionamiento del botón “Elegir Hexacóptero”

El siguiente botón “Elegir Escenario” permite navegar entre los tres escenarios con los que cuenta la aplicación, los cuales se pueden ver en la Figura 4.6. Al ser seleccionado, cada uno abrirá el escenario que le corresponde, y realizando esta acción se comprueba que las pantallas correctas son desplegadas, las cuales se pueden ver en las Figuras 4.9, 4.10 y 4.11.

Figura 4. 6. Funcionamiento del botón “Elegir Escenario”

Siempre que estén abiertas las opciones de estos dos primeros botones, el simulador estará en modo de pausa, para poder empezar es necesario presionar el tercer botón “Volar”, el cual activará los controles que permiten el vuelo del hexacóptero seleccionado.

Asimismo, el usuario tiene disponible el botón “Menú Principal” que le llevará a la pantalla de inicio de la Figura 4.1, en el caso de que sea necesario.

La última opción de este menú tiene el nombre “Tarea”, que tiene la finalidad de asignar un objetivo durante el vuelo del hexacóptero, todos los ambientes tienen 5 tareas disponibles y también la opción de volar sin ningún objetivo.

Para poder seleccionar la tarea que se va a cumplir durante el vuelo, se usan los botones de siguiente y anterior que tiene adjunto. Cada tarea irá apareciendo en el minimapa que tiene el escenario en la parte inferior izquierda, donde se podrán ver los objetivos como un punto rojo, tal como se muestra en la Figura 4. 7.

Una vez seleccionado el objetivo se podrá observar en la parte superior central de la pantalla un mensaje que indica que la tarea ha sido seleccionada, y cuando ésta sea cumplida el simulador borrará la tarea, para iniciar nuevamente con la selección.

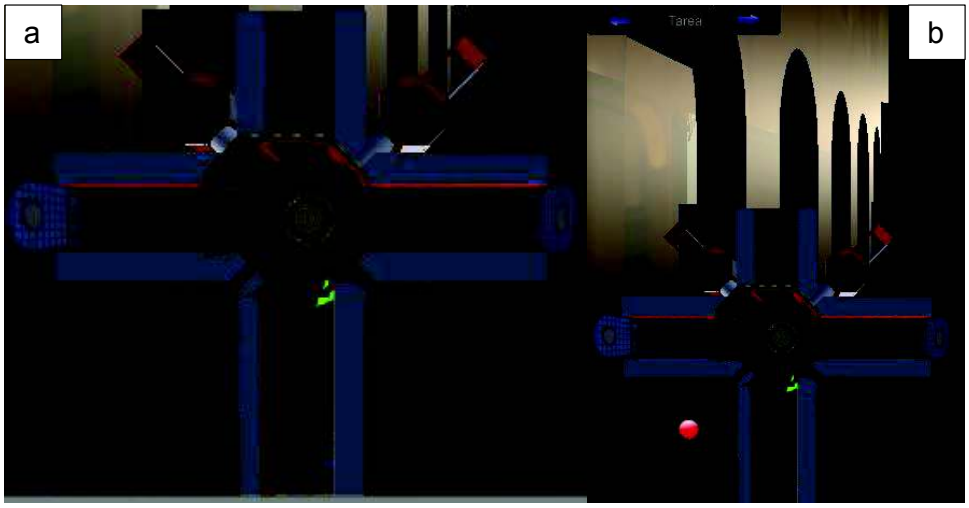

Figura 4. 7. a) Mapa sin objetivo, b) Objetivo seleccionado

El último menú por probar es el que permite seleccionar el controlador que se usará para el vuelo del hexacóptero. Este menú es visible al presionar la tecla “P” una vez, y desaparecerá al presionar la misma tecla por segunda vez. Está

ubicado en la parte central de la pantalla y al aparecer pondrá a la aplicación en modo de pausa.

Como se observa en la Figura 4.8 este menú ofrece las dos opciones de controlador y basta con un clic en el que se vaya a utilizar para activarlo.

Figura 4. 8. Selección del Controlador

4.2 VALIDACIÓN DE ESCENARIOS Y MAPAS

Mediante las pruebas realizadas en el segundo menú, se muestra como se observa cada uno de los escenarios.

En las siguientes Figuras (4.9, 4.10 y 4.11) se aprecia la parte exterior de cada uno, y en el minimapa está disponible el interior del mismo.

Figura 4. 9. Primer escenario: Casa

Figura 4. 10. Segundo escenario: Iglesia

Figura 4. 11. Tercer escenario: Supermercado

La integración del minimapa en la parte inferior izquierda, ayuda para la ubicación del usuario dentro del escenario en el que se encuentra. Y cuenta con un indicador que le permite conocer la posición y la dirección del hexacóptero, como se presenta en la Figura 4.12.

Figura 4. 12. Vuelo de hexacóptero con minimapa adjunto

El usuario también tiene la posibilidad de ocultar este minimapa en el caso de que limite su visibilidad, o le represente incomodidad. Para esto basta presionar una vez la tecla “Y” y la pantalla quedará despejada como se ve en la Figura 4.13. Si se desea volver a ver este mapa se presiona nuevamente esta tecla.

Figura 4. 13. Vuelo de hexacóptero sin minimapa

4.3 PRUEBAS DE VUELO DE HEXACÓPTEROS

Para las pruebas de vuelo de los hexacópteros, se inicia con la revisión de las cámaras.

Como se puede observar en el diagrama de flujo de la Figura 3. 53, se tiene a disposición cuatro cámaras para obtener una visualización adecuada de la ubicación actual de la aeronave y así poder decidir las acciones que se tomarán.

En las Figuras 4.14 y 4.15 se observa cada modelo de hexacóptero con la vista de las tres cámaras adjuntas al modelo, cada una será activada en diferentes situaciones durante el vuelo, dependiendo de la facilidad que le brinde al usuario.
La cuarta cámara es estática y cumple la función de minimapa, para guiar el recorrido del usuario a través del ambiente en el que se encuentra, su funcionalidad se explica detalladamente en la sección 4.2.

Figura 4. 14. Primer modelo. a) Cámara en tercera persona, b) Cámara superior, c) Cámara en primera persona

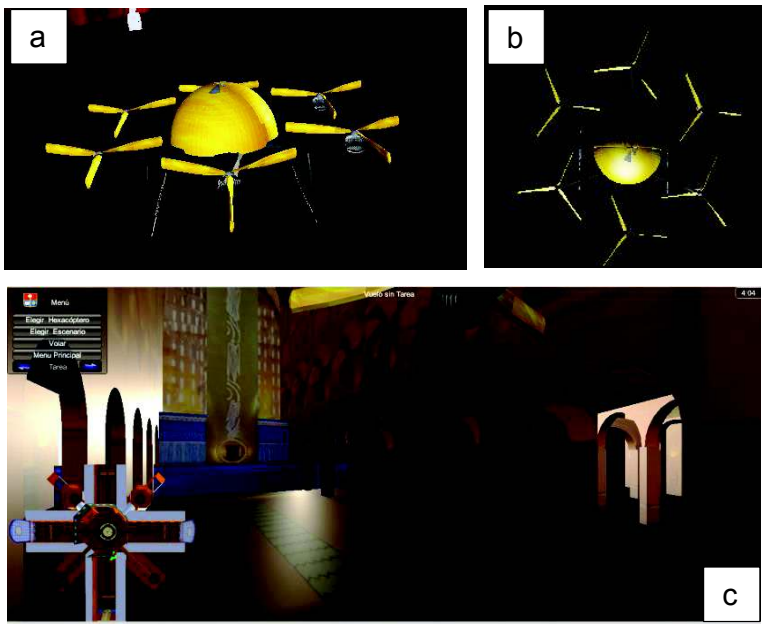

Figura 4. 15. Segundo modelo. a) Cámara en tercera persona, b) Cámara superior, c) Cámara en primera persona

Para la simulación del vuelo de los hexacópteros, la aplicación cuenta con sonido durante su funcionamiento.

Continuando con las pruebas, se analiza a continuación la característica de fuerza de realimentación, para lo cual se recurre a la medición de la distancia entre el hexacóptero y cualquier obstáculo que este alrededor, en la Figura 4.16 se observa la dirección que tendrán los rayos que medirán dicha distancia

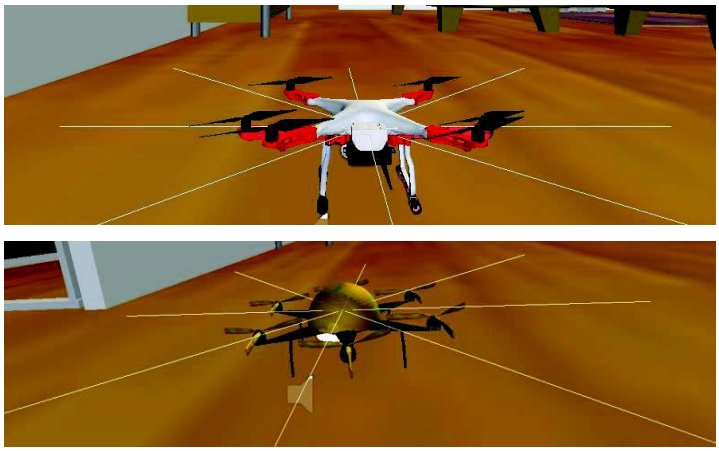

Figura 4. 16. Dirección de rayos para la medición de distancia entre los hexacópteros y los obstáculos

En la Figura 4.17 se observa el hexacóptero durante el vuelo, y se distingue el rayo que calculará la fuerza de oposición proporcional a la distancia del obstáculo, que será entregada al controlador.

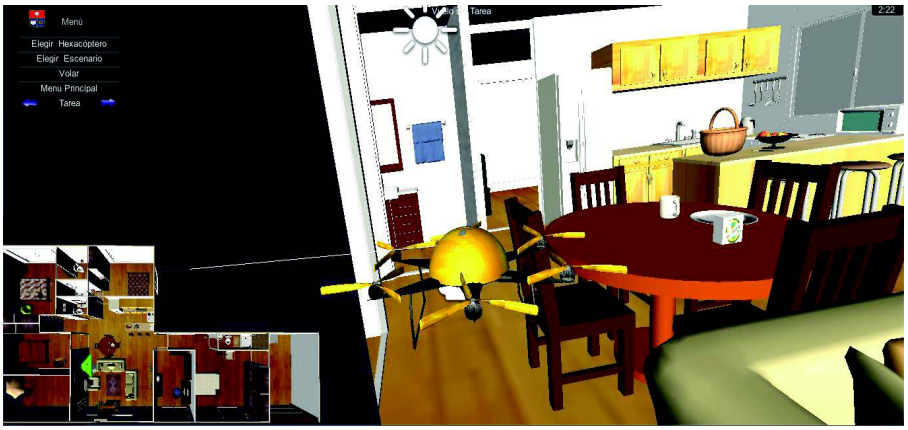

Figura 4. 17. Vuelo del hexacóptero cerca de un obstáculo

4.4 ENCUESTAS

Para obtener una realimentación del usuario acerca del funcionamiento del simulador de vuelo, se realizó una encuesta que complementa las evaluaciones realizadas en las secciones anteriores.

La elaboración de esta encuesta plantea como objetivos los siguientes:

- Valorar el grado de dificultad que presenta el aprendizaje de la teleoperación de los hexacópteros.
- Determinar que controlador da más comodidad al usuario durante la realización de maniobras de vuelo.
- Evaluar la simplicidad y consistencia de la interfaz; y comprobar si su interactividad tiene impacto en el usuario.
- Analizar si el sistema de realimentación de fuerza implementado, ayuda a advertir sobre posibles colisiones.
- Estimar la visualización que brinda cada tipo de cámara, y establecer cuál beneficia más al usuario.
- Comprobar si las tareas asignadas son alcanzadas por los usuarios.

El tamaño de la muestra fue de 20 usuarios, en cada uno se solicitó información general como: edad, género y experiencia con aplicaciones afines.
El diseño de la encuesta realizada, y los resultados de la misma se observan a continuación.

ENCUESTA

Es importante conocer su experiencia en el manejo del simulador de vuelo, le agradecemos su colaboración al responder la siguiente encuesta.

Edad _____

Género

Femenino		Masculino	

Tiene experiencia en el manejo de aplicaciones similares.

Si		No	

El grado de dificultad que presenta el aprendizaje de la teleoperación de los hexacópteros es:

Alto		Medio		Bajo	

¿Qué controlador resultó más cómodo y le permitió desenvolverse mejor dentro del simulador de vuelo?

Teclado		Joystick	

Interfaz de Usuario

Simplicidad:

En la aplicación toda la información y acciones están presentes o son accesibles con facilidad, sin exceso de contenidos. La interfaz está bien organizada y presentada oportunamente.

Consistencia:

Se refiere a la secuencia de acciones, métodos o colores comunes. Capacidad de la interfaz de lograr ser adecuado al uso para el que está destinado.

Interactividad:

Se refiere a la relación de participación entre el usuario y la aplicación.

	Alta	Media	Baja
Simplicidad			
Consistencia			
Interactividad			

La fuerza de realimentación que presenta el joystick, le ayudó a evitar colisiones.

Si		No	

Qué cámara le facilito el manejo de los hexacópteros.

Primera Persona		Tercera Persona		Cámara superior	

Logró completar la tarea escogida.

Si		No	

Comentarios / sugerencias

_____

_____

4.4.1 ANÁLISIS DE LOS RESULTADOS

4.4.1.1 Edad y género

La muestra de encuestas se la tomó de manera aleatoria sin distinguir un grupo específico de personas, para así poder tener una visión amplia del funcionamiento del simulador de vuelo.

De las estadísticas del diagrama de la Figura 4.18 se puede destacar que el grupo de personas encuestadas está dentro de un amplio rango de edades.

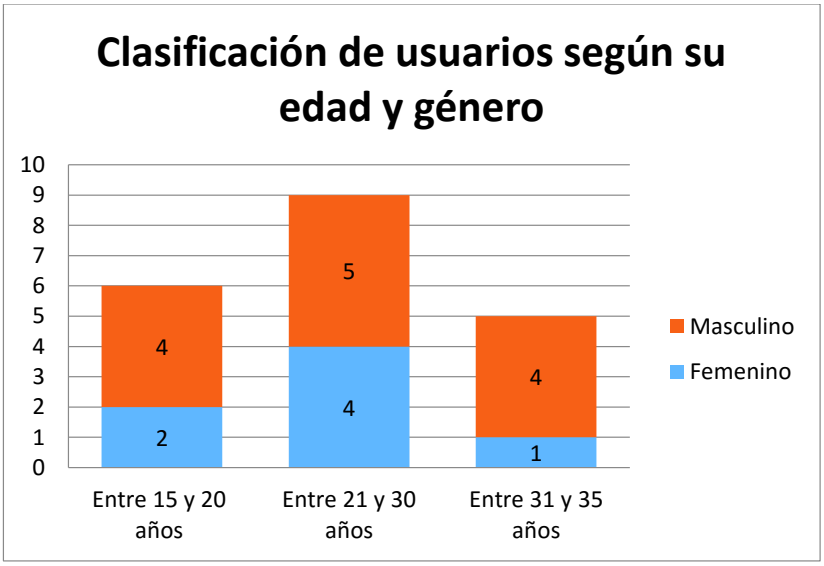

Figura 4. 18. Diagrama de clasificación de usuarios según su edad y género

4.4.1.2 Experiencia con aplicaciones afines

Esta pregunta permite conocer si los usuarios obtendrán ventaja para el aprendizaje del manejo de la aplicación, ya que una experiencia previa en este tipo de aplicaciones implica el dominio de habilidades psicomotrices desarrolladas, las que involucran precisión y un elevado nivel de coordinación de la vista y la mano. El diagrama de la Figura 4.19 se pueden observar los resultados obtenidos.

Figura 4. 19. Diagrama de experiencia con aplicaciones afines de los usuarios

4.4.1.3 Grado de dificultad en el aprendizaje

La información que entrega el diagrama de la Figura 4.20 muestra que un 65% de la muestra tuvo facilidad en la comprensión y dominio de la aplicación. Un 30% de los encuestados, es decir, 6 usuarios indicaron que el grado de dificultad fue medio y un 5% que corresponde a 1 usuario indicó un grado de dificultad alto.

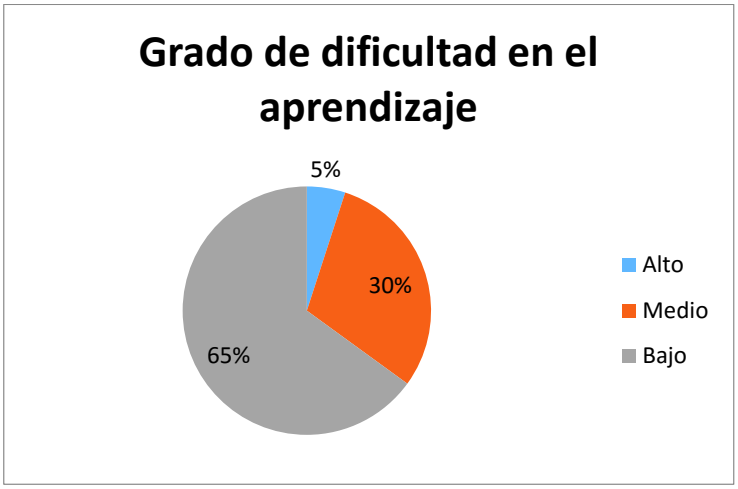

Figura 4. 20. Diagrama del grado de dificultad en el aprendizaje de la teleoperación de los hexacópteros

4.4.1.4 Elección del controlador

Se cuestiona al usuario que controlador le brinda mayor comodidad durante la realización de maniobras de vuelo, para conocer si es justificable el uso del teclado como opción de reemplazo del joystick.

De acuerdo a los resultados presentados en la Figura 4.21, se distingue que el controlador de preferencia es el joystick ya que es un 80% lo prefirió, sin embargo, existe un 20% que aprenderá a dominar el simulador de vuelo con el teclado.

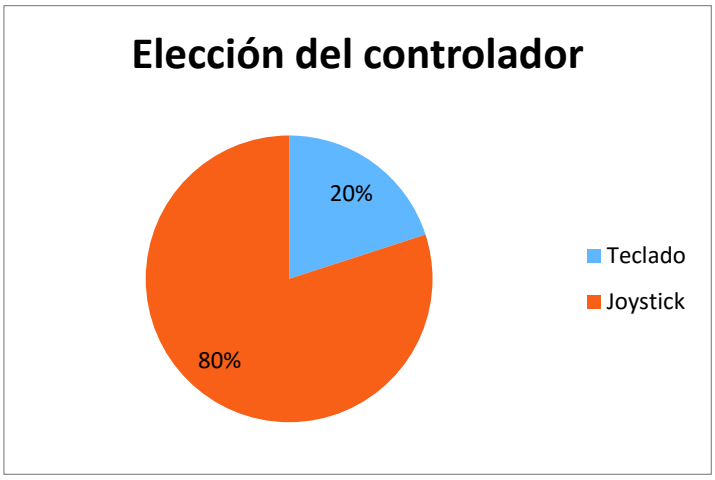

Figura 4. 21. Diagrama de la elección del controlador

4.4.1.5 Interfaz de usuario

Para este punto de la evaluación el usuario califica que tan simple le resulta la interfaz, considerando aspectos como posición y orden de controles y botones, también si la interfaz le facilita el aprendizaje y reconocimiento de su lenguaje gráfico y si la ayuda e información que ofrece es suficiente.

En la encuesta se evalúa la simplicidad, la consistencia y la interactividad de la aplicación, cada uno de estos conceptos se explican a continuación [46]:

- Simplicidad: En la aplicación toda la información y acciones están presentes o son accesibles con facilidad, sin exceso de contenidos. La interfaz está bien organizada y presentada oportunamente.

- Consistencia: Se refiere a la secuencia de acciones, métodos o colores comunes. Capacidad de la interfaz de lograr ser adecuado al uso para el que está destinado.

- Interactividad: Se refiere a la relación de participación entre el usuario y la aplicación.

De acuerdo a los resultados que se observan en la Figura 4.22, la interfaz desarrollada tiene una simplicidad del 95% (19 usuarios), es decir, las acciones que realiza y la información que recibe el usuario se presentan y organizan oportunamente. En cuanto a la consistencia se obtuvo un 75% (15 usuarios), notando que en general los menús no son abiertos y cerrados todos de la misma manera, lo que provoca que un 20% de usuarios consideraran como media esta característica. Finalmente la interactividad de la interfaz logró un 85% (17 usuarios), es decir, el control que tiene el usuario sobre la aplicación es amplia.

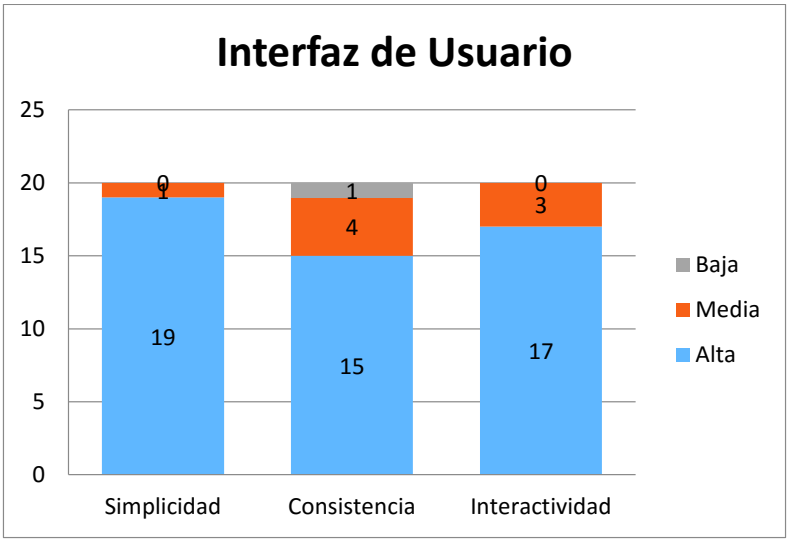

Figura 4. 22. Diagrama de opinión de la interfaz de usuario

4.4.1.6 Realimentación de fuerza

La realimentación de fuerza del Joytstick para el 100% de los usuarios ayuda a evitar posibles colisiones, además de brindar un efecto de realidad, como se observa en el diagrama de la Figura 4.23.

Lo que demuestra la importancia de la presencia de esta característica en el simulador de vuelo.

Figura 4. 23. Diagrama de opinión de la interfaz de usuario

4.4.1.7 Selección de cámaras

Es importante conocer cuál de las tres cámaras disponibles brinda mayor comodidad y dota de la visualización adecuada al usuario.

Un 65% de los usuarios indicaron que prefieren la cámara en tercera persona como principal, un 25% prefirió la cámara en primera persona y un 10 % la cámara superior, estos resultados son presentados en la Figura 4.24.

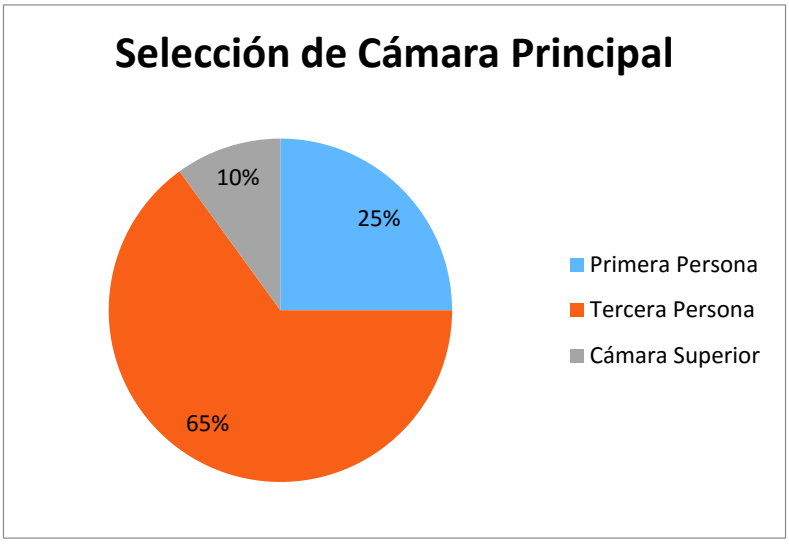

Figura 4. 24. Diagrama de selección de la cámara principal para la teleoperación de los hexacópteros.

Los resultados del diagrama anterior demuestran que todas las cámaras implementadas en el simulador son útiles, además por comentarios adicionales de los usuarios, se pudo conocer que todas las cámaras fueron utilizadas en alguna situación durante el vuelo del hexacóptero, y que en una implementación real es más difícil.

4.4.1.8 Cumplimiento de tareas

Para el vuelo de los hexacópteros el usuario tiene disponibles cinco tareas que pueden ser seleccionadas en cualquier momento. Es importante conocer si los usuarios pudieron completarlas o no.

En el diagrama de la Figura 4.25 se observa que el 85% de los usuarios lograron llegar a completar la tarea. El 15% que no alcanzó a cumplirla corresponde a los usuarios con menos experiencia que necesitan un mayor entrenamiento en el uso del simulador.

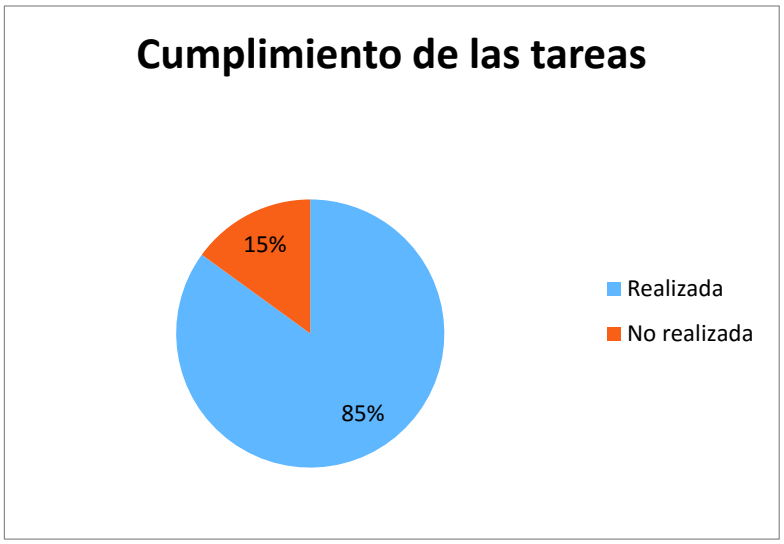

Figura 4. 25. Diagrama de cumplimiento de tareas.

De la encuesta se puede concluir que el simulador permite a los usuarios fortalecer el desarrollo de habilidades y adquirir conocimientos, con respecto a la teleoperación de hexacópteros. Les brinda la sensación de realidad mediante la realimentación de fuerza, y les ofrece además una experiencia satisfactoria.

CAPÍTULO 5

CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

Mediante el desarrollo de un simulador de vuelo, que ofrece tres ambientes semiestructurados con distintos niveles de dificultad, se logró realizar la simulación de la teleoperación (mando a distancia) de dos modelos de hexacópteros, con la característica de realimentación de fuerza.

Se diseñó los mapas de tres ambientes semiestructurados, mediante el uso de programas especializados en modelado 3D, logrando la integración adecuada entre el motor de desarrollo Unity 3D, y SketchUp, Blender y Sweet Home 3D.

El análisis del modelo dinámico de los hexacópteros, permitió el estudio y comprensión de su comportamiento, logrando la implementación en el simulador mediante la programación en lenguaje C#.

Mediante el uso del software Visual Studio, se realizó la implementación de los modelos 3D de dos hexacópteros, comandados con los scripts desarrollados en lenguaje C#.

Se logró el desarrollo e implementación dentro de la programación, de un algoritmo de control para el cálculo de la fuerza que aparece en la palanca de mando como contraposición al movimiento ejercido por el usuario, cuando la aeronave se encuentra cerca de un obstáculo. Estableciendo que esta fuerza debe ser inversamente proporcional a la distancia del hexacóptero con el obstáculo estático.

El rango de fuerza de realimentación que es percibida en el Joystick Force Feedback, se seleccionó de tal manera que advierta la cercanía de un obstáculo

estático, sin obligar el cambio de dirección, lo que permitió que esta acción sea decisión exclusiva del usuario.

Se entregó al usuario un sentido de presencia, a través del dispositivo háptico utilizado, logrando ofrecerle una aplicación que le advertirá sobre posibles colisiones de los hexacópteros con obstáculos estáticos.

Se diseñó una plataforma simuladora de vuelo de hexacópteros en diferentes ambientes semiestructurados, permitiendo mostrar a los usuarios el comportamiento y mando de estas aeronaves.

Con la integración de dos tipos de controladores en el simulador de vuelo, que son el teclado y el Joystick Force Feedback, se logró que la forma de teleoperar el hexacóptero mediante el simulador, sea escogida por el usuario de acuerdo a la comodidad y seguridad que cada uno le brinde.

El desarrollo de una interfaz sencilla, intuitiva y amigable para los usuarios, permitió ofrecer un simulador de vuelo que permite navegar por la aplicación sin dificultad, y que ofrece la información y ayuda oportuna.

Mediante el análisis de los resultados de la encuesta realizada, se logró conocer la experiencia directa de los usuarios con el simulador de vuelo, encontrando su aceptación con la aplicación.

Se realizaron pruebas con el simulador de vuelo, las cuales permitieron comprobar su correcto funcionamiento para la teleoperación de los hexacópteros con realimentación de fuerza.

5.2 RECOMENDACIONES

Para el desarrollo de la aplicación se recomienda instalar todos los complementos necesarios y verificar que el sistema cumpla con los requerimientos para su instalación y funcionamiento.

Se recomienda el uso de programas desarrollados exclusivamente para el modelado 3D, ya que permiten realizar diseños más detallados obteniendo una aplicación final sofisticada y completa.

Se recomienda utilizar el mismo lenguaje de programación en todos los scripts que forman parte de la aplicación, para así evitar problemas durante la comunicación de variables de diferentes scripts.

Un trabajo a futuro que se recomienda a partir del simulador de vuelo desarrollado, es la implementación de la teleoperación de un hexacóptero real, permitiéndole a la aeronave ser gobernada mediante la aplicación y con la ayuda del Joystick Force feedback utilizado.

Se recomienda también como trabajo a futuro, ampliar la simulación de la teleoperación de un hexacóptero, implementando una red de hexacópteros, lo que se hace posible gracias al toolkit Force Feedback implementado en el simulador de vuelo, ya que admite hasta cuatro controladores en una misma aplicación.

REFERENCIAS BIBLIOGRÁFICAS

[1] Aplicaciones y Operación con Drones/RPAS, 9 Julio 2015. [En línea]. Available: http://drones.uv.es/origen-y-desarrollo-de-los-drones/.

[2] Grupo de aviación de caza, Julio 2015. [En línea]. Available: http://www.simbolicodecaza.org/index.php?option=com_content&view=article&id=124:drones&catid=52:articulos-profesionales-lunes&Itemid=194.

[3] J. M. G, Abril 2010. [En línea]. Available: http://blog.sandglasspatrol.com/index.php/articulos/41-militar/758-uavs-clasificacion-tendencias-y-normativa-de-espacio-aereo.

[4] A. Bachfischer, Julio 2014. [En línea]. Available: http://ricabib.cab.cnea.gov.ar/477/1/1Bachfischer.pdf.

[5] Asociación de Robótica, Febrero 2014. [En línea]. Available: http://asrob.uc3m.es/index.php/Introducci%C3%B3n_a_los_UAVs.

[6] C. Bernard, Septiembre 2010. [En línea]. Available: https://www.youtube.com/watch?v=t3XjkUex50Y.

[7] F. Gutierrez, Enero 2015. [En línea]. Available: http://www.infotecarios.com/los-drones-en-las-bibliotecas-todavia-lejos-en-america-latina/.

[8] P. Shadbolt, Enero 2015. [En línea]. Available: http://cnnespanol.cnn.com/2015/01/14/alas-roboticas-drones-militares-que-imitan-halcones-e-insectos/.

[9] HelicamPeru, 2011. [En línea]. Available: http://helicamperu.com/hexacopter.

[10] A. López, Septiembre 2011. [En línea]. Available: http://diebotreise.blogspot.de/2011/09/eleccion-del-multicoptero.html.

[11] AliExpress, Diciembre 2015. [En línea]. Available: http://es.aliexpress.com/store/product/F05114-D-RC-HexaCopter-ARF-Electronic-KK-Multicopter-V2-3-Hex-Rotor-Flight-Control-Board-30A/404049_1032647703.html.

[12] Plejad, Octubre 2013. [En línea]. Available: http://www.plejad.net/en/build_h6.htm.

[13] T. Magnusson, 2014. [En línea]. Available: http://liu.diva-portal.org/smash/get/diva2:736152/FULLTEXT01.pdf.

[14] R. Castañeda, Marzo 2013. [En línea]. Available: https://davidtheory.wordpress.com/category/unity3d/.

[15] Asociación cultural de simulación aérea, 2016. [En línea]. Available: http://www.simulacionaerea.org/.

[16] Ecured, Mayo 2016. [En línea]. Available: http://webcache.googleusercontent.com/search?q=cache:http://www.ecured.cu/Simulador_de_vuelo.

[17] D. d. aeromodelista, Enero 2015. [En línea]. Available: http://diariodelaeromodelista.blogspot.com/2015/01/simuladores-de-vuelo.html.

[18] E. Mercado, Mayo 2013. [En línea]. Available: http://aerowiki-info.blogspot.com/2013/05/los-simuladores-de-vuelo-su-importancia.html.

[19] H-sim, 2009. [En línea]. Available: http://www.h-sim.com/new_uav_sims.php.

[20] Copadata, 2015. [En línea]. Available: http://www.copadata.com/es/productos/product-features/interfaz-hombre-maquina-hmi.html.

[21] M. Simarro, Octubre 2010. [En línea]. Available: http://simarromatias-imd2010.blogspot.com/2010/10/evaluacion-teorica-libro-abierto-tema.html.

[22] L. Barrios y I. Galeano, 2014. [En línea]. Available: http://jeuazarru.com/wp-content/uploads/2014/10/HMI.pdf.

[23] D. Cottino, D. Bustamante, D. Spaciuk, D. Robaina, E. Coquet y F. Liotine, 2008. [En línea]. Available: https://books.google.com.ec/books?id=i_ofrmcZ_DkC&pg=PA190&lpg=PA190&dq=force+feedback+joystick+significado&source=bl&ots=LyzTzq3I8y&sig=cySVtI_EjtpyoB71ayt-Wr4RZ-U&hl=es&sa=X&sqi=2&ved=0ahUKEwjP3Pqiyf_JAhXBKyYKHfQYCycQ6AEILDAD#v=onepage&q=force%20feedbac.

[24] RaceSimOnline, Febrero 2012. [En línea]. Available: http://www.racesimonline.com/articulos/forceFeedback.php.

[25] Duiops, 2009. [En línea]. Available: http://www.duiops.net/hardware/joystick/msffp.htm.

[26] Logitech, 2016. [En línea]. Available: http://gaming.logitech.com/es-es/product/extreme-3d-pro-joystick.

[27] T. Martín y A. Serrano, 2013. [En línea]. Available: http://acer.forestales.upm.es/basicas/udfisica/asignaturas/fisica/dinam1p/fine

rcia.html.

[28] P. Rodriguez, 2014. [En línea]. Available: http://bibing.us.es/proyectos/abreproy/90168/fichero/Memoria.pdf.

[29] P. Dávila y J. Orna, Diseño, construcción y control de un hexacóptero de monitoreo, Quito: Escuela Politécnica Nacional, 2015.

[30] V. Espinosa y V. Moya, DISEÑO Y SIMULACIÓN DE UN ALGORITMO DE CONTROL ROBUSTO PARA FORMACIÓN DE CUADRICÓPTEROS, Quito: Ecuador, 2016.

[31] H. Nabil, Dynamic Modeling and Control of a Quadrotor Using Linear and Nonlinear Approaches, Cairo: The American University in Cairo, 2014.

[32] Sweet Home 3D, 2016. [En línea]. Available: http://www.sweethome3d.com/es/userGuide.jsp#drawingRooms.

[33] Warehouse 3D, 2016. [En línea]. Available: https://3dwarehouse.sketchup.com/search.html?q=interior&backendclass=entity&sortby=reviewCount%20DESC.

[34] Google SketchUp, 2016. [En línea]. Available: http://sabia.tic.udc.es/gc/Contenidos%20adicionales/trabajos/Tutoriales/SketchUp/introduccion.html.

[35] G-Blender, 2016. [En línea]. Available: http://www.g-blender.org/uso-de-texturas-en-blender-ii-uv.

[36] A. Cerón y P. Bedoya., «Manual Básico De Unity 3d Como Apoyo Al Desarrollo Turístico Nacional,» Pereira, Universidad Tecnológica de Pereira, 2014.

[37] Unity 3D, 2015. [En línea]. Available: http://docs.unity3d.com/es/current/Manual/UnityManualRestructured.html.

[38] UPV formación online, 2014. [En línea]. Available: https://upvx.es/courses/LenguajesYSistemasInformaticos/unity/2015-01/courseware/e0524c33573f422591512f2085c197ea/66f22802d8c44b478a8b57c56f4d5a29/.

[39] J. Abad, Mayo 2013. [En línea]. Available: http://cursosdekencho.blogspot.com/2013/05/IndieLeccion01IntroUnity.html.

[40] Videojuegos 2, 2015. [En línea]. Available: https://moodle2014-15.ua.es/moodle/pluginfile.php/117881/mod_resource/content/1/vii-10-unityscript.pdf.

[41] D. Pechir. [En línea]. Available: http://tutosunity.arredemo.org/files/Tutoriales/d7eaba_unity3d_manualscripting_profesional.pdf?ckattempt=1.

[42] Unity 3D, Agosto 2012. [En línea]. Available: http://forum.unity3d.com/threads/force-feedback-controller-toolkit.159297/.

[43] ConfigurarEquipos, Junio 2007. [En línea]. Available: http://www.configurarequipos.com/doc532.html.

[44] F. F. C. Toolkit, *Manual de uso,* 2012.

[45] Todoexpertos, Noviembre 2011. [En línea]. Available: http://www.todoexpertos.com/categorias/tecnologia-e-internet/hardware/respuestas/64679/joystick.

[46] U. d. Oviedo, 2014. [En línea]. Available: http://di002.edv.uniovi.es/~alguero/eaac/Ensenianza/GUIAP.pdf.

[47] S. H. 3D, 2016. [En línea]. Available: http://www.sweethome3d.com/freeModels.jsp.

[48] Arts Digital Institute, 2015. [En línea]. Available: http://cadmiami.com/blog_arts_institute/conoce-las-mejores-ventajas-de-sketchup-2015/ .

[49] Formación en red, 2012. [En línea]. Available: http://www.ite.educacion.es/formacion/materiales/181/cd/indice.htm.

[50] Dreamstime, 2014. [En línea]. Available: http://es.dreamstime.com/stock-de-ilustraci%C3%B3n-vista-superior-del-hexacopter-con-la-cubierta-de-la-fibra-de-carbono-image54695743.

[51] G-Blender, Agosto 2014. [En línea]. Available: http://www.g-blender.org/uso-de-texturas-en-blender-ii-uv .

[52] J. Baldeón y J. Escorza, Automatización de un hexacóptero para despegue, aterrizaje y vuelo en un camino cerrado, Quito: Escuela Politécnica Nacional, 2015.

ANEXO A

MANUAL DE USUARIO

A.1 Descripción de la aplicación

El presente proyecto simula el vuelo de hexacópteros en ambientes internos. A través del desarrollo, diseño, modelado e implementación de entornos, se replican ambientes semiestructurados con obstáculos estáticos, estos ambientes son una casa, una iglesia y un supermercado.
Cada uno de ellos presenta diferentes obstáculos, desplegando distintos niveles de dificultad en el manejo del simulador de vuelo, así como cinco tareas por cumplir, que pueden ser seleccionadas por el usuario, o si se desea también se puede volar sin ningún objetivo por alcanzar.

La aplicación brinda la posibilidad de elegir entre dos modelos de hexacópteros, logrando así, ser más útil e interactivo.
El simulador incluye la opción de realimentación de fuerza, la cual se logra con el Joystick Siderwinder Force Feedback 2, permitiendo advertir al usuario sobre posibles colisiones, y entregarle la sensación de presencia al operador. La fuerza realimentada aplicada está limitada por el dispositivo háptico.

A.2 Ingreso a la aplicación

Para ingresar a la aplicación se debe abrir la carpeta que la contiene, en la cual se encuentran cinco archivos.

Se abre el archivo ejecutable: Simulador de vuelo, con el icono del escudo de la Escuela Politécnica Nacional, que se ve seleccionado en la figura A.1.

Figura A. 1. Carpeta que contiene la aplicación

A.3 Instalación de complementos

Para el funcionamiento del simulador de vuelo es necesaria la instalación del complemento DirectX9.

DirectX9

Este componente es necesario, ya que mejora el rendimiento de las interfaces gráficas, reproducción de música y de sonido. La descarga es sencilla y gratuita desde cualquier buscador de internet.

Manual de Usuario

Una vez realizada la descarga e instalación del complemento, se inicia la aplicación con la pantalla de Figura A.2, y se debe asegurar que los parámetros estén configurados de la manera que se indica en la imagen.

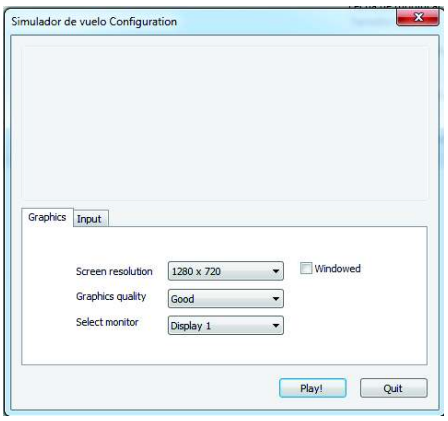

Figura A. 2. Inicio de la aplicación

Cargada la aplicación, se abre la interfaz de inicio del simulador que es la de la Figura A.3. Esta pantalla tiene un menú con tres botones.

Figura A. 3. Interfaz de inicio

A continuación se explica el funcionamiento del menú, desde el tercer botón. El botón “Acerca de” despliega una pantalla con la descripción de la aplicación, y puede ser cerrada mediante el interruptor que tiene en la parte superior izquierda, como se ve en la Figura A.4.

Figura A. 4. Funcionamiento del botón “Acerca de”

El segundo botón es “Ayuda”, al presionarlo se puede observar una pantalla con un manual de instrucciones, que mostrará la manera de maniobrar los

hexacópteros, tanto con el teclado como con el Joystick Force Feedback, así como una breve descripción de sus movimientos. Esta pantalla puede observarse en la Figura A.5 y al igual que la pantalla anterior puede cerrarse con el interruptor de la parte superior izquierda.

Figura A.5. Funcionamiento del botón “Ayuda”

Finalmente el botón “Comenzar” que es el primero del menú, abrirá el primer escenario del simulador, visto desde la parte exterior, como se ve en la Figura A.6. En donde se observa un menú en la parte superior izquierda, que permitirá navegar por la aplicación, y un mapa que representa la parte interior del escenario.

Figura A. 6. Funcionamiento del botón “Comenzar”

En este punto o durante el vuelo de los hexacópteros se puede elegir el controlador que se desea usar, presionando la tecla “P” aparece el menú que se observa en la Figura A.7 en donde se presiona una vez la opción que se desea, y para esconder el menú se presiona nuevamente la misma tecla.

Figura A. 7. Elección del controlador: Joystick o Teclado

A continuación se explica el funcionamiento del menú que aparece en los escenarios del simulador de manera permanente en la parte superior izquierda.

El botón “Elegir Hexacóptero” despliega los dos modelos de hexacópteros disponibles como se ve en la Figura A.8.

Figura A. 8. Funcionamiento del botón “Elegir Hexacóptero”

Al presionar la opción “DJI PHANTOM” aparece la pantalla con el modelo correspondiente, como se observa la Figura A.9.

Figura A. 9. Funcionamiento del botón “DJI PHANTOM”

Del mismo modo el segundo botón “Ah-700 HEXACOPTER” muestra el modelo que le corresponde como se ve en la Figura A.10.

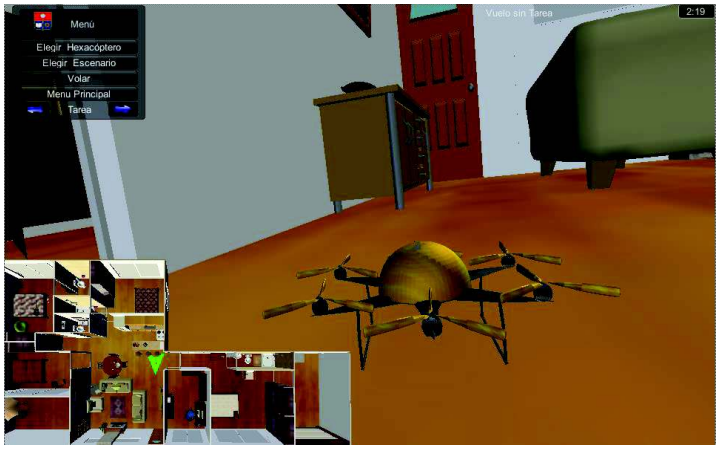

Figura A. 10. Funcionamiento del botón “Ah-700 HEXACOPTER”

El siguiente botón “Elegir Escenario” muestra los tres ambientes como se ve en la Figura A.11. El primer botón “CASA” muestra la pantalla de la Figura A.6. El botón “IGLESIA” abrirá la pantalla que se ve en la Figura A.12. Y finalmente el botón “SUPERMERCADO” entrega la pantalla que se observa en la Figura A.13.

Figura A. 11. Funcionamiento del botón “Elegir Escenario”

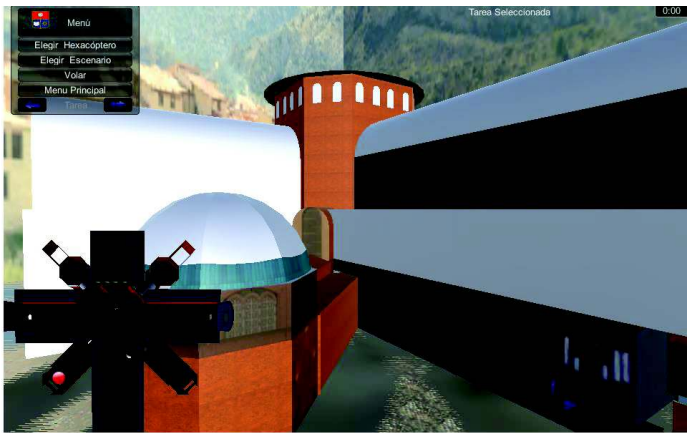

Figura A. 12. Funcionamiento del botón “IGLESIA”

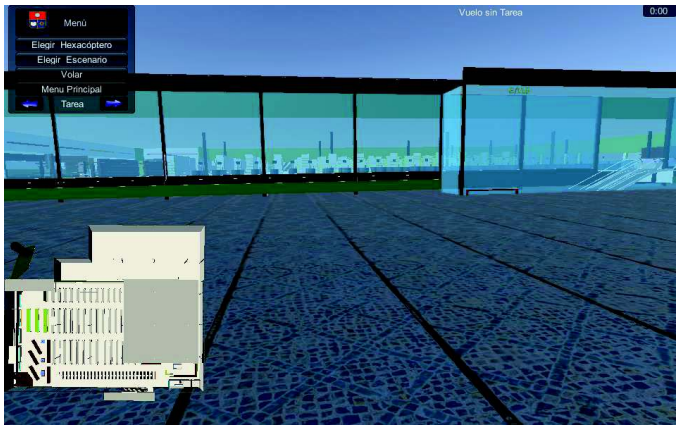

Figura A. 13. Funcionamiento del botón “SUPERMERCADO”

Cualquier elección de las dos anteriores pausará la reproducción de la aplicación, y para activarla es necesario seleccionar el botón “Volar” que es el tercero del menú. Con el cual se inicia el manejo del hexacóptero seleccionado.

El cuarto botón del menú “Menú Principal” permite ir a la interfaz de inicio de la aplicación, si se desea ver las instrucciones o su descripción nuevamente.

Finalmente se tiene el botón “Tareas” que está formado de dos botones, uno derecho y uno izquierdo. Con estos botones se elige la tarea a cumplir durante el vuelo.
Dentro de estas tareas existe la posibilidad de seleccionar un vuelo sin ninguna tarea asignada “Vuelo sin tarea” mensaje que es mostrado en la pantalla (Figura A.14).

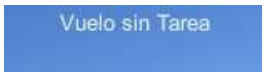

Figura A. 14. Mensaje del vuelo sin tarea asignada

Además cada escenario cuenta con cinco objetivos que son elegidos por el usuario mediante sus dos botones. Cada objetivo va a apareciendo en el minimapa que se encuentra en la parte inferior izquierda de la pantalla. Como se ve en la Figura A.15.

Figura A. 15. Tarea seleccionada para el vuelo del hexacóptero

Estas tareas pueden ser seleccionadas al abrir el escenario o durante cualquier momento durante el vuelo.

Como funciones de ayuda, se tiene el cambio de cámaras que son: primera persona, tercera persona y una cámara superior.
Asimismo se puede activar o desactivar el minimapa.
Estas funciones pueden ser manejadas desde el teclado o desde el joystick, y se observan en la Figura A.16.

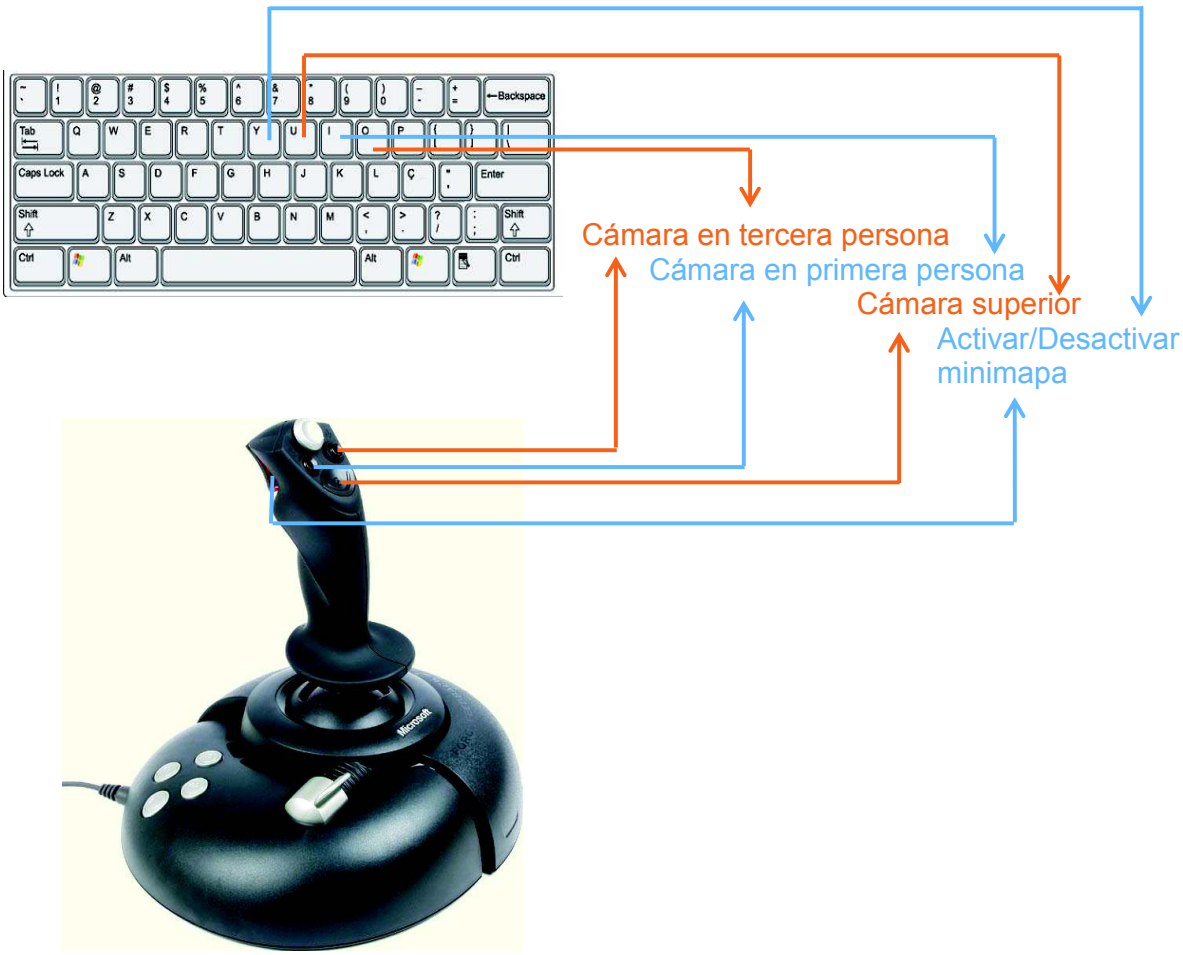

Figura A. 16. Cambio de cámaras y minimapa

Para el vuelo del hexacóptero, se muestran las instrucciones en el manual, que se muestra en la Figura A.17.

Figura A.17. Manual de instrucciones para el vuelo de los hexacópteros

ANEXO B

SWEET HOME 3D

Este software está disponible en su página oficial www.sweethome3d.com, y se encamina hacia el diseño de interiores. Debido a su libre distribución es muy utilizado y se encuentran modelos en línea que pueden ser descargados y editados; si lo que se busca es personalizar el ambiente, el software no presenta dificultad para su aprendizaje en la creación de un nuevo proyecto, la elección dependerá de las necesidades y capacidades del usuario [32].

La interfaz de Sweet Home 3D es amigable e intuitiva para el usuario, lo que facilita el dibujo del plano deseado. Lo principal es crear la base del plano para desde ahí poder dividir ambientes con la implementación de paredes; a continuación para disponer el estilo de los ambientes creados, basta colocar el mobiliario, el cual está disponible en el catálogo organizado por categorías. Además se tiene la posibilidad de colocar y cambiar texturas en cada uno de los elementos utilizados. Todo el proceso realizado se actualiza en el plano 3D al mismo tiempo que se trabaja sobre el plano 2D [32].

Para el diseño del primer ambiente del simulador de vuelo se inicia el programa, y como paso inicial se abre el cuadro de diálogo de preferencias para definir las características que se desea en el mismo, las cuales se muestran en la Figura B.1.

Cada una de las características es activada de acuerdo a la funcionalidad del modelo que se creará. Una vez definidas las preferencias de la casa que se desea diseñar se inicia con el plano, creando las 6 habitaciones con las que se cuenta, adicional a esto la sala, comedor, cocina y patio. A continuación se añaden las puertas y ventanas para obtener una visión objetiva de la casa vacía, tarea que se facilita con la propiedad magnetismo que se activó en las preferencias, la cual permite colocar estos elementos sobre las paredes, con la

orientación y tamaño adecuado de acuerdo a la orientación y el grosor de la pared correspondiente.

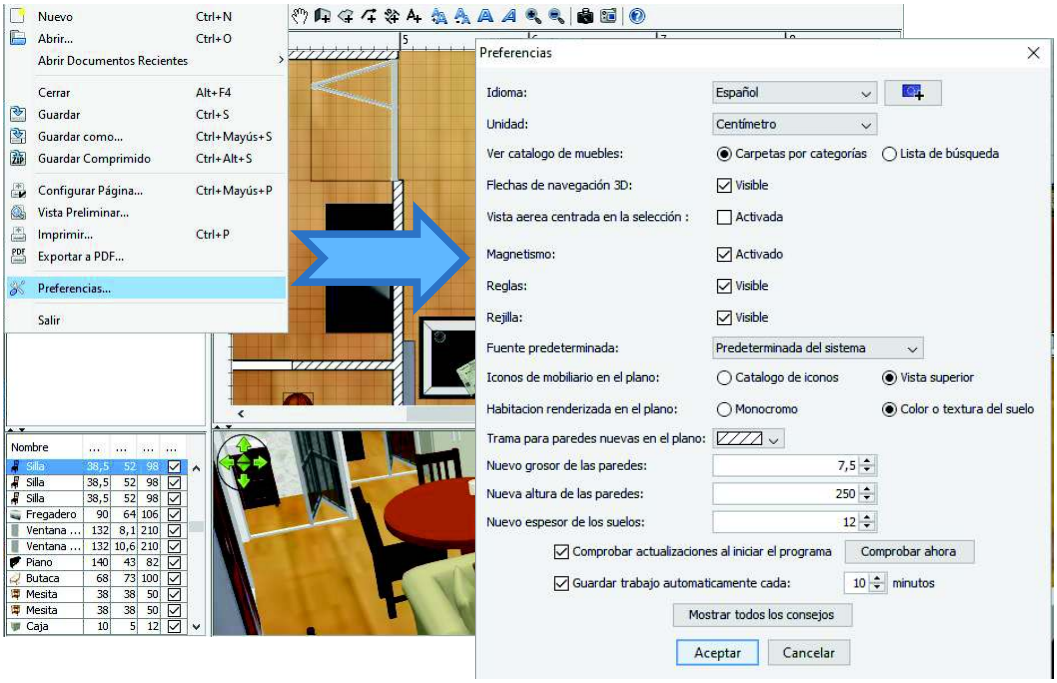

Figura B. 1. Editando preferencias

Finalizado este proceso se realiza la implementación del mobiliario en cada una de las áreas, ubicando en ellas los muebles que se consideren adecuados. Para añadir los muebles a la casa, se los arrastra y coloca desde el catálogo disponible en la parte izquierda de la interfaz (Figura 3.1), hacia el plano de la casa o a la lista de muebles, como se muestra en la Figura B.2, o se escoge la pieza deseada en el catálogo y se selecciona en botón *Añadir muebles* en la barra de herramientas.

De la misma forma que para puertas y ventanas, la propiedad magnetismo, ayuda a ajustar la ubicación, el ángulo y tamaño del mobiliario. Permitiendo que la posición de la pieza seleccionada se ajuste de manera automática al espacio donde será ubicada. Así, al colocarla tendrá una rotación tal que su parte posterior este junto a la pared, o tendrá la altura necesaria de acuerdo al espacio,

es decir, si no hay más mobiliario por defecto la elevación es a nivel del piso, y si se lo coloca sobre otro objeto será elevado sobre este último [32].

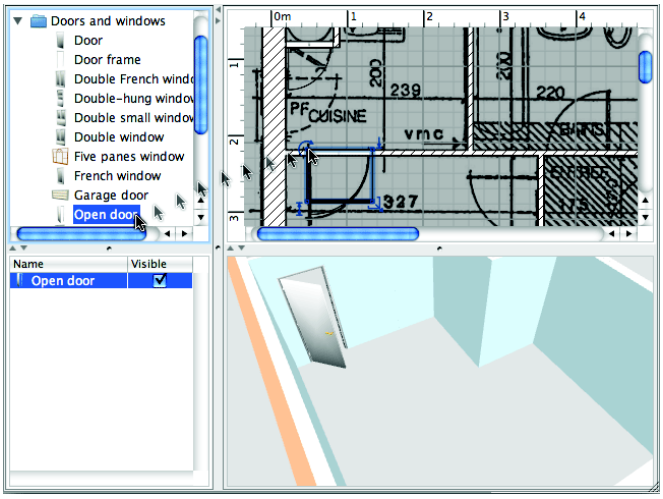

Figura B. 2. Plano listo para ubicación de mobiliario desde el catálogo [32].

Para poder ajustar los muebles a un tamaño específico, se debe seleccionar el elemento, y aparecerán cuatro indicadores en cada esquina de la pieza seleccionada, como se puede apreciar en la Figura B. 3. En esta Figura se distingue que además del tamaño es posible modificar el ángulo de rotación y elevación del elemento.

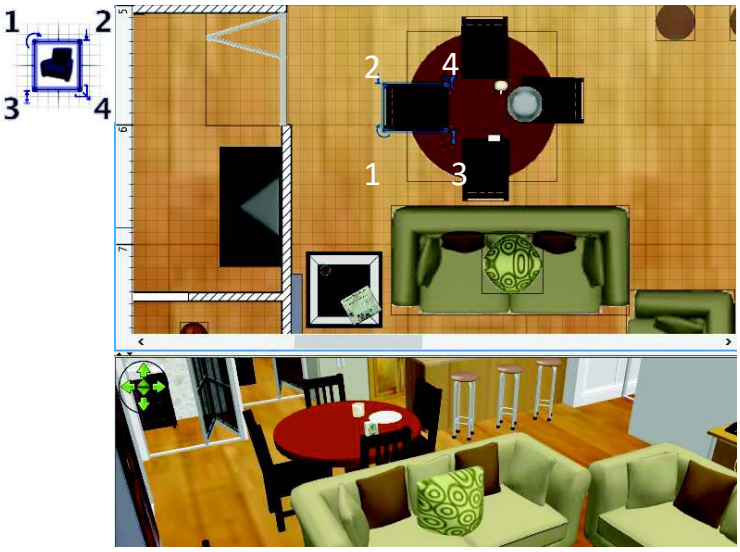

Figura B. 3. Indicadores de una pieza: 1. Indicador de rotación, 2. Indicador de elevación, 3. Indicador de altura, 4. Indicador de anchura

Para la misma tarea se tiene más opciones, se puede hacer doble clic sobre la pieza de mobiliario deseada, o elegir *Mobiliario* > *Modificar* desde el menú para realizarlo desde el cuadro de diálogo. Como se muestra en la Figura B. 4, este cuadro de diálogo permite editar además de las características mencionadas, el nombre del elemento seleccionado, su ubicación, su color o textura, su visibilidad y si su modelo 3D debe ser reflejado (como en un espejo).

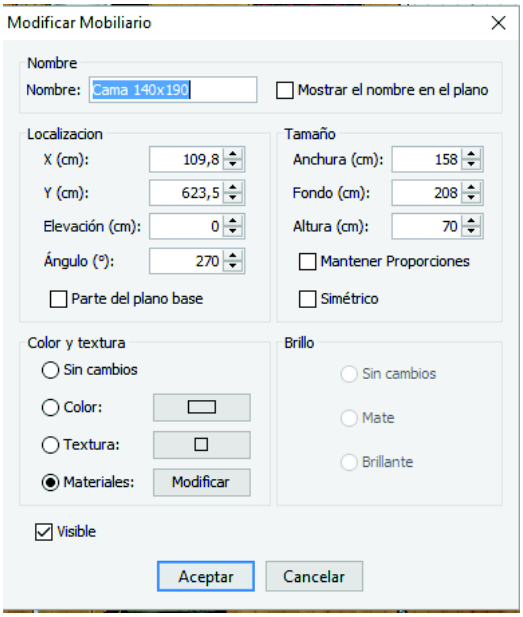

Figura B. 4. Editando propiedades de los muebles

Si el mobiliario existente en el catálogo no cumple con las especificaciones deseadas por el usuario, este tiene la opción de acudir a la herramienta de importación de mobiliario 3D que ofrece el software. Para ello se accede a la página oficial de Sweet Home 3D o mediante el menú en la opción *Mobiliario* > *Importar Mobiliario* que se aprecia en la Figura B. 5. Esta herramienta permite definir el tamaño y clasificación dentro del catálogo del programa, que tendrá el modelo 3D importado.

Dentro de la página web se tienen disponibles numerosos diseños de piezas de mobiliario, que pueden ser descargados y utilizados en el diseño. Estos modelos no tienen costo y se encuentran clasificados por tipo y por contribuyente del diseño, un ejemplo se puede observar en la Figura B. 6.

Además de estas opciones el usuario puede crear sus propios modelos de mobiliario en programas como Blender o Art of Illusion. Ya que Sweet Home 3D soporta diseños de modelos 3D con formato OBJ, DAE, 3DS, en un archivo ZIP que contiene un archivo de este tipo, o en un archivo KMZ. Logrando así ser una gran herramienta en el diseño de interiores.

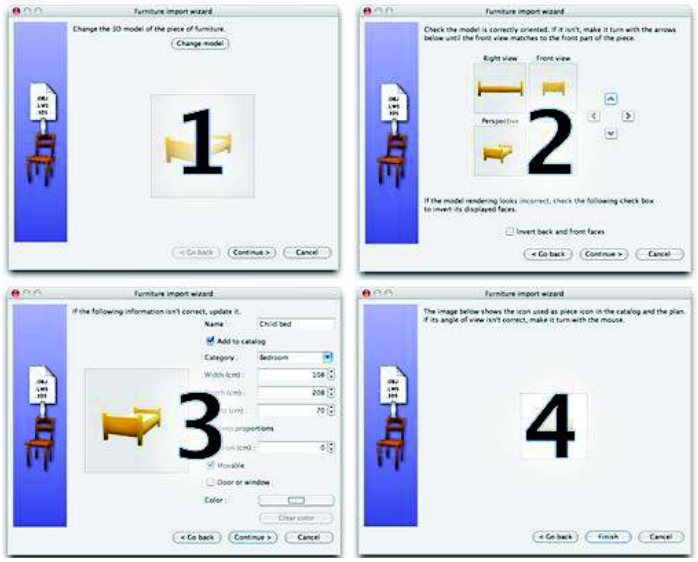

Figura B. 5. Asistente de importación de muebles [32]

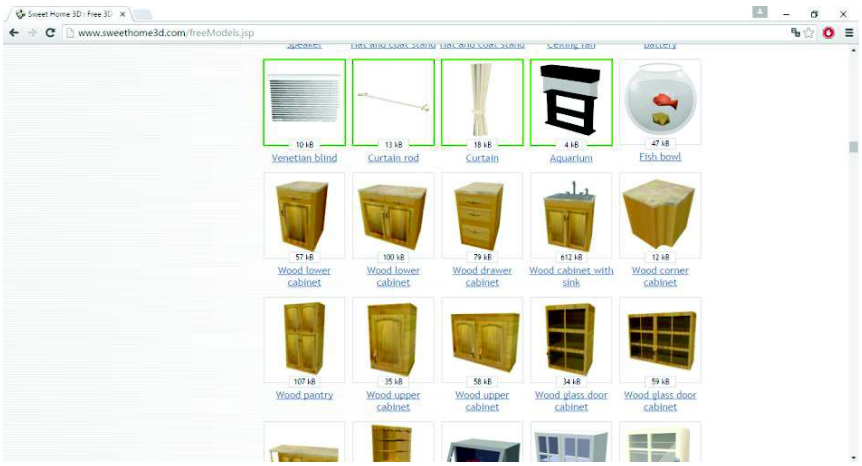

Figura B. 6. Página web con modelos 3D para Sweet Home 3D [47]

Cuando ya la casa tiene los espacios distribuidos y el mobiliario ubicado, se puede modificar el color o la textura tanto de paredes, piso, techo y muebles, de esta manera se podrá ver el modelo de manera más real.

Esto al igual que para la configuración del mobiliario es posible realizarlo de varias maneras, que puede ser haciendo doble clic en el objeto a modificar o mediante el cuadro de diálogo *Modificar Mobiliario*, como el de la Figura B. 4, eligiendo la opción *Color y textura*.

Con esta opción se puede realizar el cambio con colores y texturas disponibles en el programa, con los que el usuario haya descargado o tenga disponible en sus archivos. El procedimiento se lo observa en las Figuras B. 7 y B. 8.

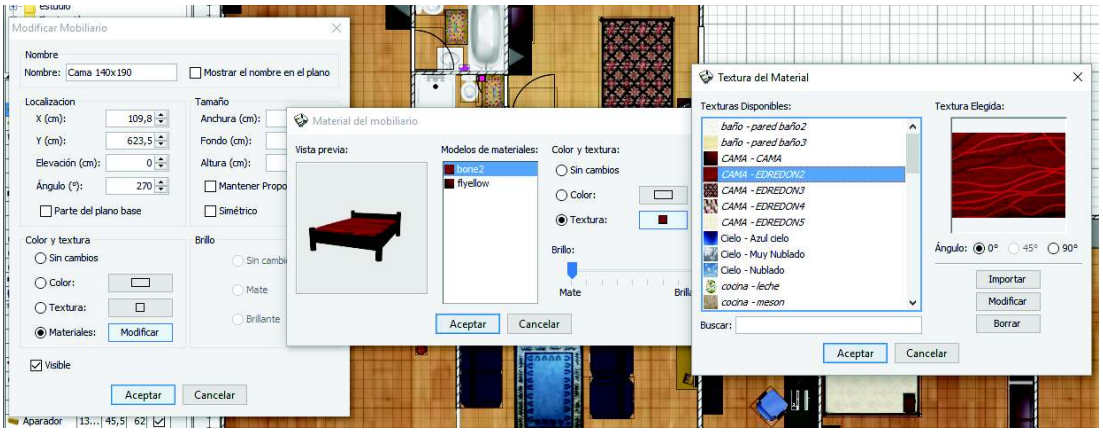

Figura B. 7. Elección de texturas que ofrece Sweet Home 3D

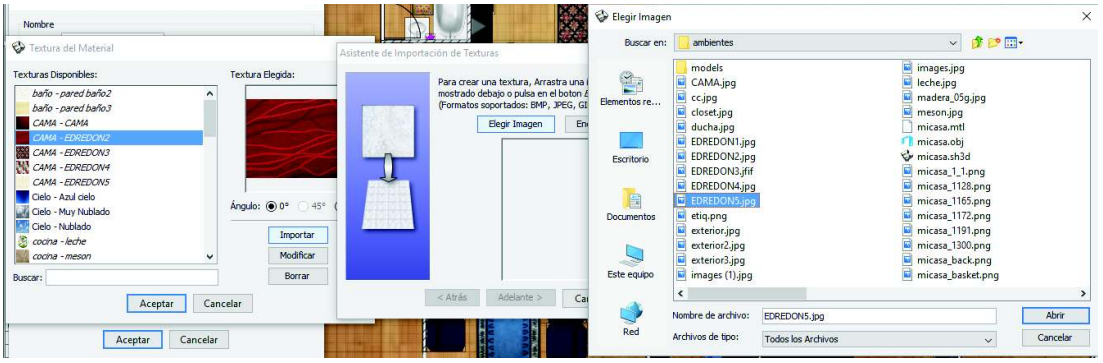

Figura B. 8. Importar texturas descargadas por el usuario

Después del proceso realizado, está lista la casa que corresponde al primer ambiente del simulador de vuelo. Este es un espacio pequeño con varios obstáculos estáticos que pondrán a prueba las destrezas de vuelo de los hexacópteros, ya que el usuario se enfrenta al paso por puertas y pasillos, lo que hace interesante la experiencia.

ANEXO C

SKETCHUP

Es importante conocer que SketchUp no se limita al modelado de edificaciones, sino que permite también crear modelos 3D de personas, animales, vehículos, etc. Además el software cuenta con amplia galería de modelos 3D, texturas e imágenes, disponibles en la herramienta 3D Warehouse, que se encuentra fácilmente en un buscador en internet (Figura C. 1), o en la pestaña Archivo del menú (Figura C. 2).

Esta galería dispone de todos sus modelos en diferentes formatos que son soportados por SketchUp, tiene archivos desde muy básicos hasta muy complejos, algunos de ellos son de descarga gratuita y otros deben ser comprados para poder ser utilizados. Así mismo el usuario que disponga de una cuenta de Google tiene la posibilidad de ofrecer sus diseños a los demás usuarios [34].

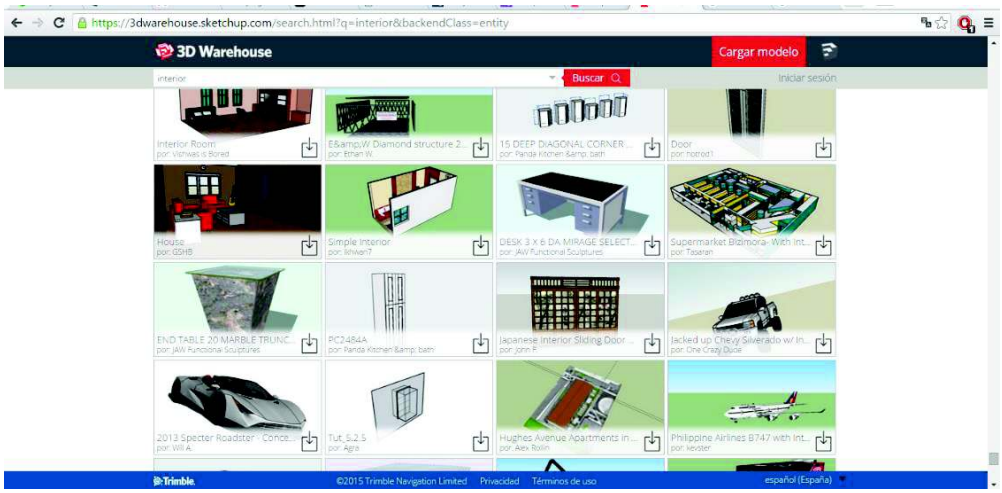

Figura C. 1. Galería de objetos, texturas e imágenes para descargar por internet [33]

Según el manual de SketchUp publicado por Google [34], sus características destacables son:

• Sistema de cursores de dibujo “inteligente” que permite dibujar objetos 3D usando una pantalla 2D y un ratón.

• Simulador de ángulo del sol.

• Habilidad para animar la cámara y movimientos del sol.

• Los modelos pueden ser coloreadas individualmente con una surtida colección de colores, texturas y materiales.

• Interoperabilidad con Google Earth.

• Simplicidad en su uso sobre otros programas de modelado.

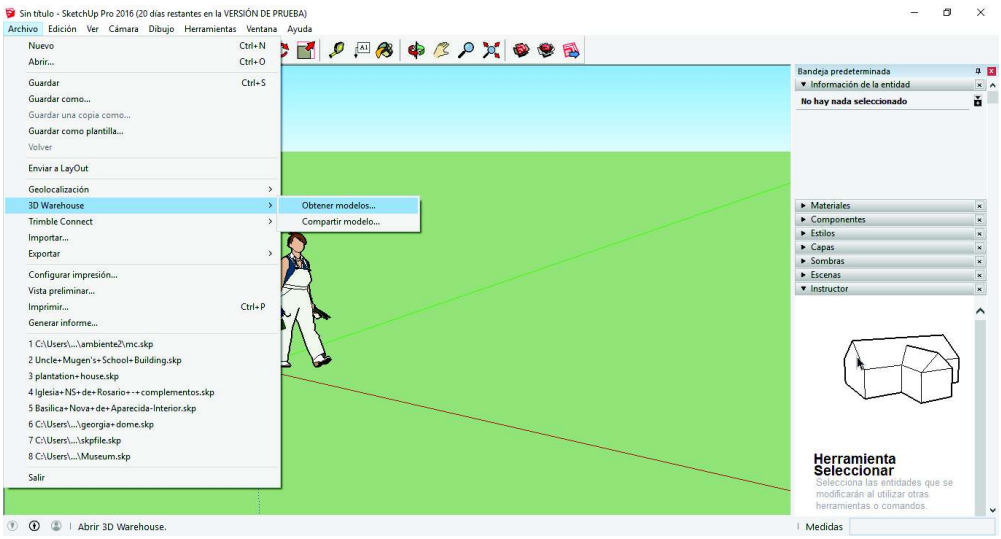

Figura C. 2. Importando modelos a SketchUp d

Este software se caracteriza por los sofisticados modelos que se pueden diseñar en él, debido a su precisión que es de hasta una milésima de milímetro, el modelo puede contar con los más mínimos detalles. Otra de sus ventajas como lo menciona Alejandro Ríos experto en el manejo del software [48] es que ofrece una perfecta integración con Google Earth, importando fácilmente imágenes aéreas y datos del terreno en 3D, imágenes Street View y modelos contextuales.

ANEXO D

BLENDER 3D

Esta interfaz aun siendo poco intuitiva, presenta la ventaja de poder ser personalizada por el diseñador, seleccionando cuales ventanas son necesarias en el momento adecuado y ubicándolas donde más le convenga.

Al iniciar el programa se observan pocas herramientas, sin embargo, éstas están ocultas para ser usadas en función a las necesidades del usuario, cambiando de un entorno a otro según se avanza en el modelo que se está creando.

El entorno de trabajo se divide en diferentes áreas las cuales se denominan editores (Figura D.1) y el usuario tiene la opción de desplegar el menú correspondiente al icono de cada área.

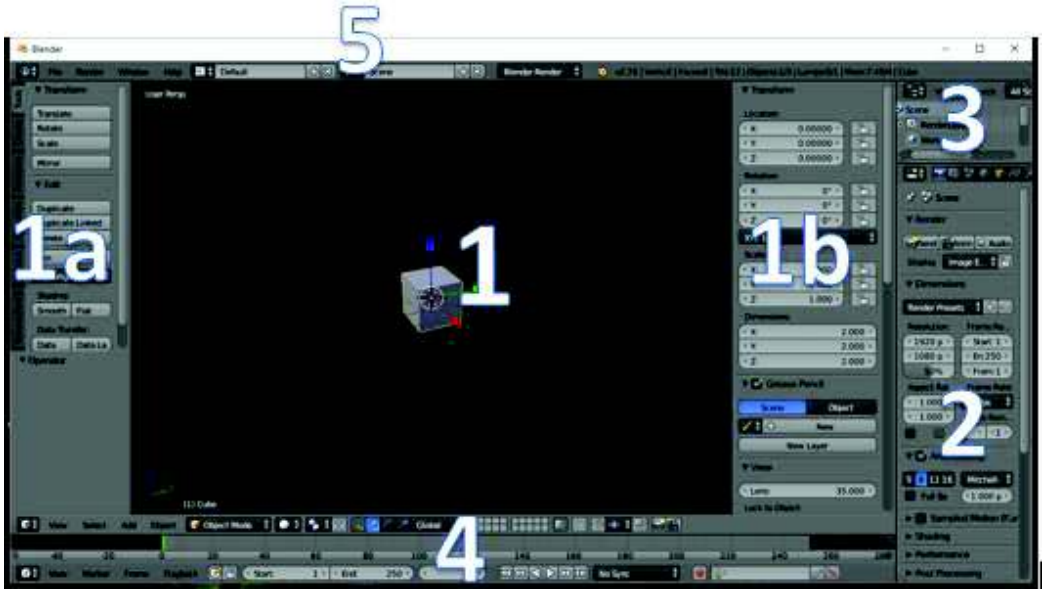

Figura D. 1. Entorno por defecto de Blender 3D

El área central (1) se denomina Vista 3D, en este espacio se puede visualizar el desarrollo del proyecto. Está áera tiene disponibles dos ventanas las cuales pueden ser visibles o no visibles (1a y 1b).

El cuadro 1a corresponde a las herramientas y se hace visible/no visible con Vista/Herramientas o con el atajo de teclado "T", mientras el cursor del ratón esté en la zona Vista 3D.

El cuadro 1b, se denomina Propiedades y se accede a él con Vista/Propiedades o con el atajo de teclado "N".

En la parte derecha de la interfaz está el editor Paneles (2), en esta área se encuentran las herramientas que permitirán personalizar los modelos, brindando matices que les darán calidad, belleza, elegancia, rendimiento, etcétera.

La zona inferior de la Vista 3D es el editor Línea de tiempo (4). Debido a la característica de animación del software, desde aquí se puntualiza el fotograma actual con la línea vertical de color verde.

Finalmente el editor Info (5) corresponde al clásico menú Archivo [49], en donde se puede guardar, abrir, importar o exportar archivos.

Es importante tener en cuenta que existen muchas y muy variadas técnicas para el diseño de modelos 3D, a las cuales podrá acudir el diseñador, sin embargo se debe considerar que “la esencia del modelado 3D está en el llamado box modeling o modelado de caja en el que se parte de una Figura primitiva (cubo, por norma general) de la que se van obteniendo nuevas caras, lados y vértices” como lo señala Joaquín Herrera en su publicación “Técnicas de Modelado” [49]. Este concepto es primordial para el desarrollo del proyecto en general y para el caso específico del modelado en 3D en Blender.

Con las nociones expuestas del funcionamiento del programa, se puede iniciar el diseño del hexacóptero eligiendo como base la fotografía de la vista superior de un hexacóptero (Figura D.2), que será el modelo a desarrollar. Creando una superficie que imitará la forma de un hexacóptero del mundo real.

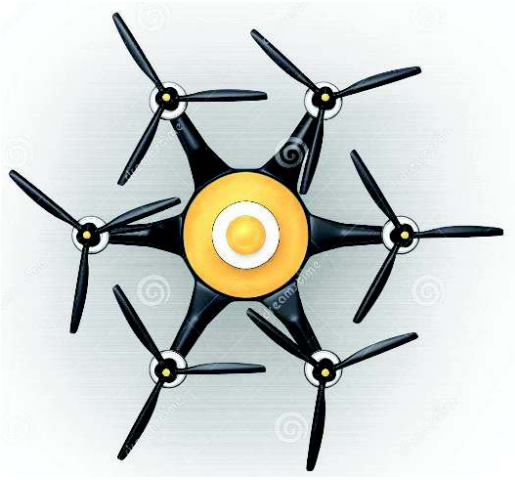

Figura D. 2. Vista superior del hexacóptero a modelar en Blender 3D [50]

Para facilitar el diseño de la aeronave se puede hacer uso de los siguiente atajos del teclado: rotar (R), mover (G), escalar (S), extrusión o prolongación de un área (E).

Al iniciar Blender, en primera instancia se observa un cubo al centro de la pantalla en un plano tridimensional, con los ejes X (rojo), Y (verde) y Z(azul). Como se observa en la Figura D.1.

Antes de insertar la imagen de referencia, se debe elegir la vista ortogonal en el menú view, y a continuación se elige la vista superior (top) ya que la imagen D.2 corresponde a este tipo de vista.

Para colocar como base de vista en Blender la imagen elegida, y usarla de guía para el modelado, se usa el editor de Propiedades, que corresponde al entorno de trabajo 2 de la Figura D.1. Seleccionando *Background image* y haciendo clic en *add image*, aparecerá la opción de agregar la referencia y al hacerlo ésta aparecerá en la pantalla de vista 3D, como se muestra en la Figura D.3.

En el diseño se usó el “box modeling”, y se tomó como Figura primitiva una esfera. Para añadirla se activa el modo de edición y se arrastra la Figura desde el entorno de trabajo *Herramientas* hacia la pantalla de trabajo, como se observa en la Figura D. 3, la esfera está de color anaranjado, lo que significa que está

seleccionada y lista para ser editada. Se procede entonces a darle forma a la aeronave, modelando la parte central y también lo brazos.

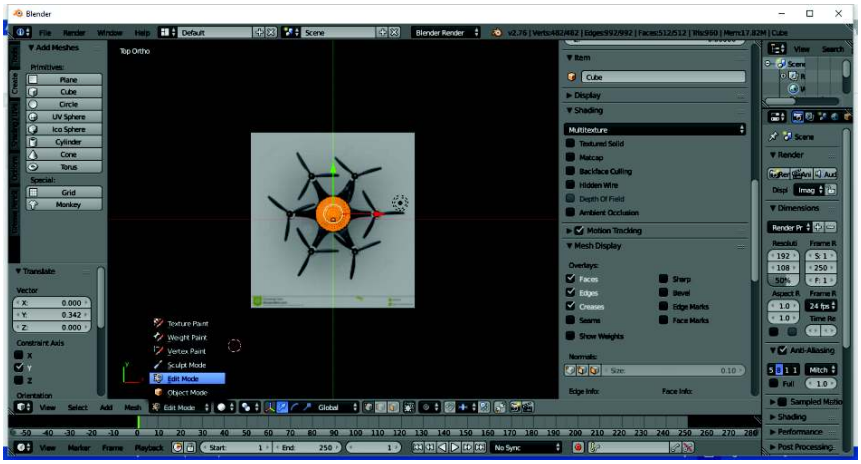

Figura D. 3. Imagen base insertada en Blender 3D

Para facilitar el diseño del hexacóptero y reducir el tiempo de elaboración del mismo, se activa la herramienta *Mirror*, el cual invierte automáticamente una malla con respecto a sus ejes locales X, Y y Z, que atraviesan el centro del objeto (por tanto el plano de reflexión está definido por los otros dos ejes). Para activarla, dentro del editor Paneles se elige en el menú *Type of active data to display and edit*, la opción *Add Modifier*, como se muestra en la Figura D.4, se mostrarán todas opciones disponibles dentro del cual se encuentra la herramienta de interés.

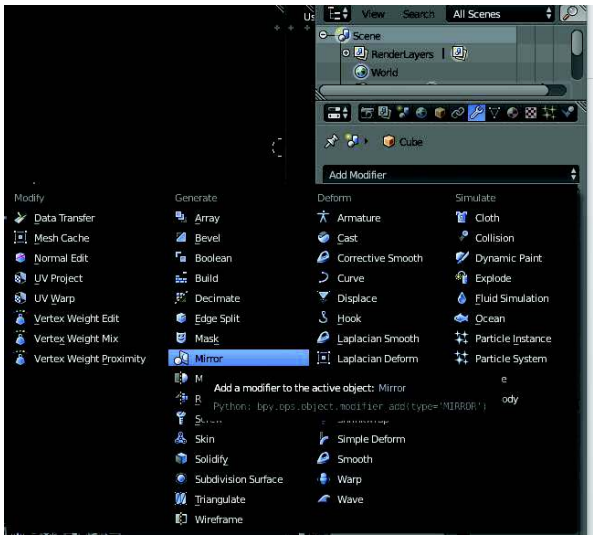

Figura D. 4. Activando la herramienta Mirror en Blender 3D

Realizado ya el cuerpo del hexacóptero con la herramienta *Mirror* activada, se puede ya tener una idea del modelo 3D que se está realizando (Figura D.5).

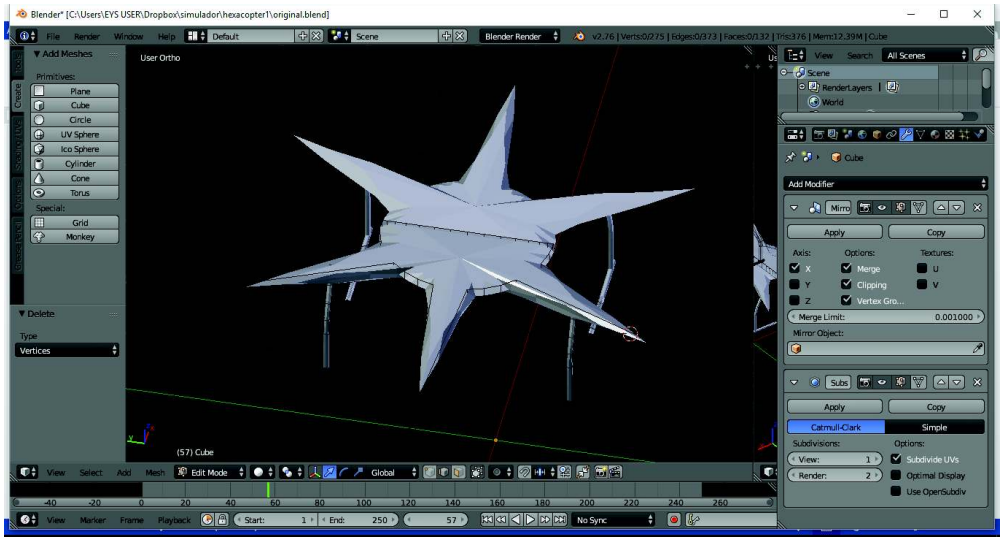

Figura D. 5. Desarrollo del modelo del hexacóptero a partir de una esfera

Siguiendo el mismo procedimiento se agregan los soportes del hexacóptero, y así también se agregan los motores a partir de un cilindro.
Finalmente se agrega una carcasa a la aeronave y se lo hace partiendo de una esfera de la misma manera que las piezas anteriores.

Para que el modelo se pueda exportar hace falta colocar texturas, y así lograr tener detalles en las superficies. En Blender, las Texturas pueden aplicarse a un Material, a un Pincel y al Entorno. Las texturas que se manejan para el diseño son aplicadas a un material, para esto se elige el tipo de editor *UV/Image Editor*, como en la Figura D. 6, este mapeo es el procedimiento de “desenvolver” un modelo 3D para generar un mapa UV el cual representa la malla del modelo en un mapa 2D que se puede visualizar simultáneamente con el modelo 3D. Así se puede determinar cómo colocar la textura en la malla [51].

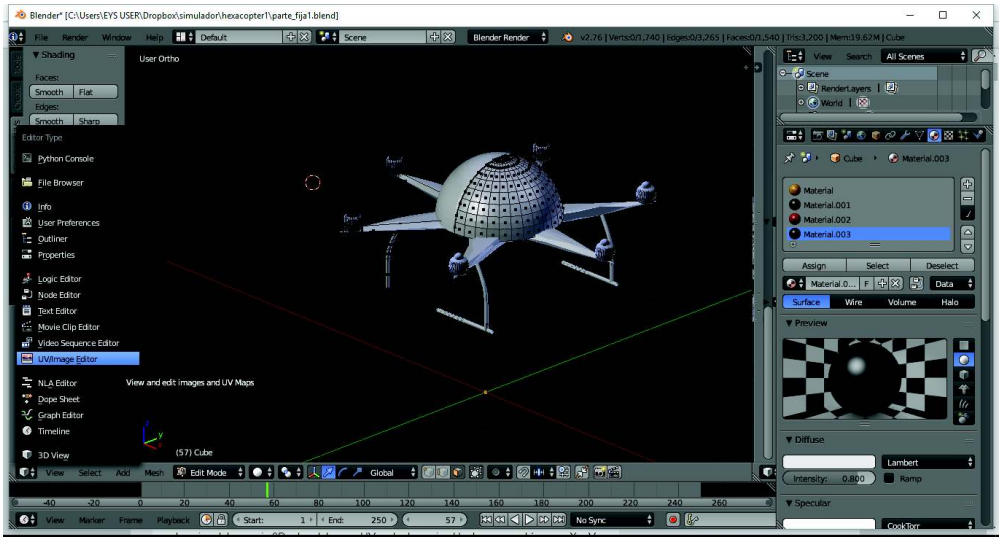

Figura D. 6. Activando el editor UV/Image Editor

Las texturas deseadas pueden ser descargadas fácilmente de internet, y una vez que están guardadas en el computador, pueden ser agregadas como un material nuevo. En el icono de material que se encuentra en el editor Paneles, como se indica en la Figura D. 7.

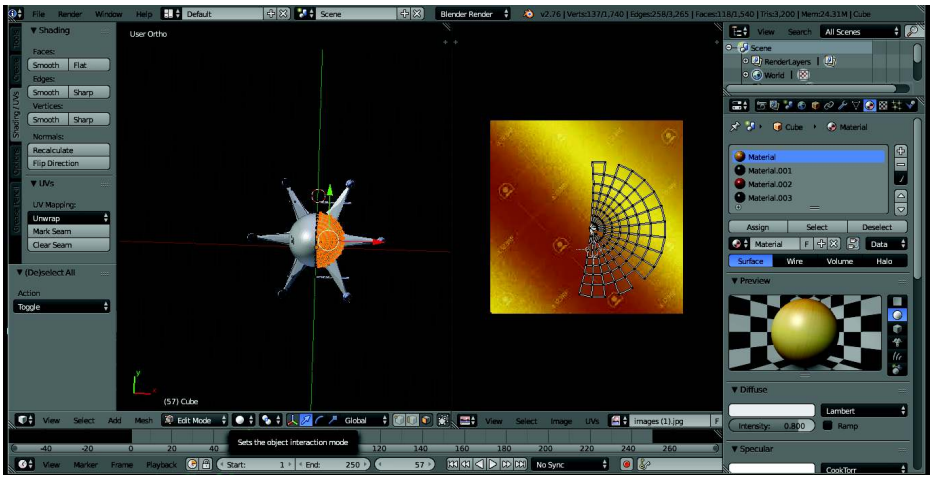

Figura D. 7. Añadiendo texturas a las piezas de hexacóptero

De esta manera se puede agregar una textura diferente a cada pieza del diseño, obteniendo finalmente el modelo con el que se trabajará en Unity 3D, el mismo que se muestra en la Figura D. 8.

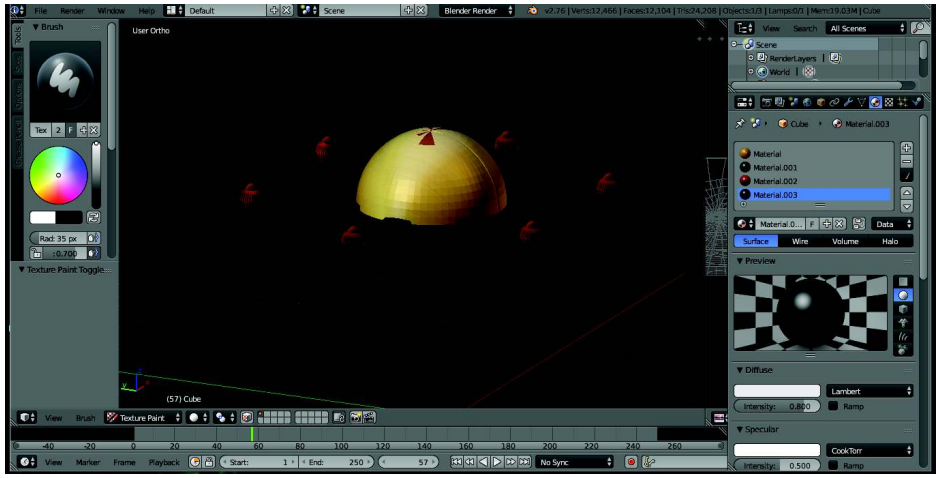

Figura D. 8. Modelo del hexacóptero sin hélices.

Para el diseño de las hélices se sigue el mismo procedimiento que para el cuerpo del hexacóptero, con la diferencia de que en este caso la Figura primitiva fue un cubo, en la Figura D. 9 se observa el modelo de la hélice terminado. Se procede entonces a agregarle las texturas y después de esto está lista para ser exportada a Unity 3D.

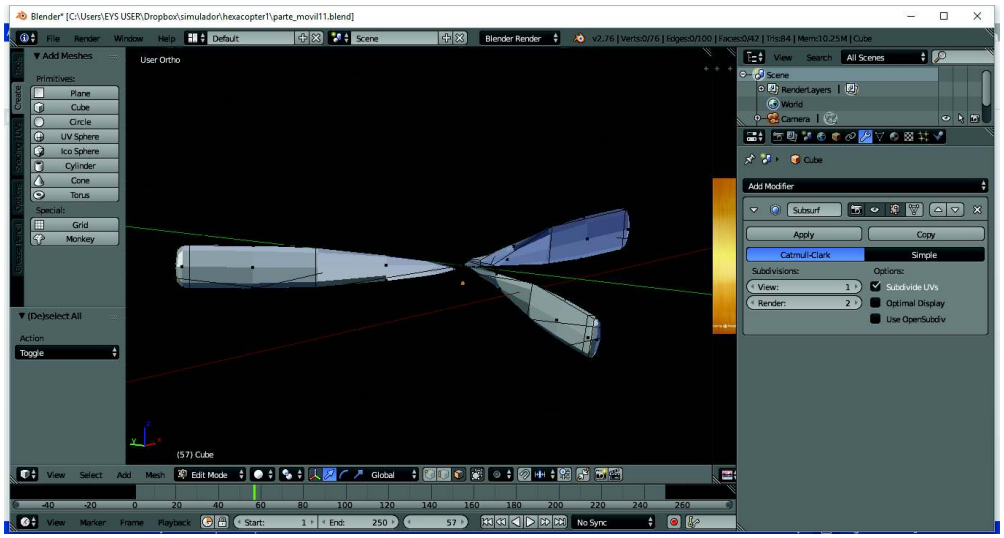

Figura D. 9. Modelo de hélice para el hexacóptero.