DISEÑO E IMPLEMENTACIÓN DE UNA ESTRATEGIA PARA PLANEACIÓN DE TRAYECTORIAS CON EVASIÓN DE OBSTÁCULOS DE UN DRON TIPO QUADROTOR UTILIZADO EN LA CARACTERIZACIÓN DE AFLORAMIENTOS GEOLÓGICOS.

FABIANA PAOLA MONTES MONTERREY
RICHARD LEONARDO MOGOLLÓN MENDOZA

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
FACULTAD DE INGENIERÍAS
Ingeniería mecatrónica
BUCARAMANGA
2021

DISEÑO E IMPLEMENTACIÓN DE UNA ESTRATEGIA PARA PLANEACIÓN DE TRAYECTORIAS CON EVASIÓN DE OBSTÁCULOS DE UN DRON TIPO QUADROTOR UTILIZADO EN LA CARACTERIZACIÓN DE AFLORAMIENTOS GEOLÓGICOS.

TESIS DE PROYECTO DE GRADO PARA OPTAR A TÍTULO DE INGENIERO MECATRÓNICO

DIRECTOR:
CAMILO ENRIQUE MONCADA GUAYAZAN, MSc

CODIRECTOR:
SEBASTIAN ROA PRADA, PhD

UNIVERSIDAD AUTÓNOMA DE BUCARAMANGA
FACULTAD DE INGENIERÍAS
PROGRAMA DE INGENIERÍA MECATRÓNICA
BUCARAMANGA
2021

NOTA DE ACEPTACIÓN

_____

_____

_____

MSc. Ing. CAMILO MONCADA GUAYAZAN
Director del proyecto de grado
Universidad autónoma de Bucaramanga

_____

PhD. Ing. SEBASTIAN ROA PRADA
Codirector del proyecto de grado
Universidad autónoma de Bucaramanga

MSc. Ing. JESSICA MARADEY LAZARO
Calificadora del proyecto de grado
Universidad autónoma de Bucaramanga

_____

MSc. Ing. HERNANDO GONZALEZ ACEVEDO
Calificador del proyecto de grado
Universidad autónoma de Bucaramanga

_____

Bucaramanga, 2021

AGRADECIMIENTOS

A nuestro director Camilo Moncada y Codirector Sebastián Roa, muchas gracias por habernos acompañado a lo largo de este proceso.

A nuestros compañeros del semillero de investigación "modelado y simulación" que conformar el grupo de Kondor-Map, muchas gracias por apoyarnos y acompañarnos en este proceso.

A nuestros familiares, amigos y compañeros que estuvieron apoyándonos, alentándonos al progreso y éxito de nuestro trabajo.

INDICE DE FIGURAS

1. RESUMEN

TÍTULO

DISEÑO E IMPLEMENTACIÓN DE UNA ESTRATEGIA PARA PLANEACIÓN DE TRAYECTORIAS CON EVASIÓN DE OBSTÁCULOS DE UN DRON TIPO QUADROTOR UTILIZADO EN LA CARACTERIZACIÓN DE AFLORAMIENTOS GEOLÓGICOS.

AUTORES

Fabiana Paola Montes Monterrey
Richard Leonardo Mogollón Mendoza

PALABRAS CLAVES

Trayectoria, Obstáculo, Diseño, estrategia, 2D, 3D

DESCRIPCIÓN

Este proyecto de grado del programa ingeniería mecatrónica tiene como objetivo desarrollar un algoritmo de planeación de trayectorias que incluya la evasión de obstáculos de un dron tipo quadrotor que será utilizado para la exploración y caracterización de afloramientos geológicos. Estos afloramientos son rocas que se presentan desnudas de vegetación y en algunos casos tienen formas complejas con distintos tamaños, por lo cual un dron sería una herramienta indicada para el estudio de estos afloramientos.

El proyecto de distribuye en dos fases, la primera la fase diseño donde se creó algoritmos de planeación de trayectorias en ambientes conocidos en dos etapas: se crea un modelo de campo de fuerza artificial (APF) para cada elemento en el ambiente usando funciones sigmoides, posteriormente se calculan las trayectorias utilizando estrategias basadas en descenso de gradiente. Se desarrollaron 2 estrategias, la primera basada en puntos móviles que interconectan el dron con la meta, donde posteriormente cada punto se mueve hacia zonas libres de la influencia de los obstáculos siguiendo el campo potencial, lo que hace que se encuentren caminos libres de obstáculos. El segundo se basa en el uso del concepto de zonas seguras, el cual se utiliza como criterio para actualizar la posición de los puntos.

2. ABSTRACT

TITLE

DESIGN OF A STRATEGY FOR PLANNING OF TRAJECTORIES WITH AVOIDANCE OF OBSTACLES OF A QUADROTOR DRONE USED IN THE CHARACTERIZATION OF GEOLOGICAL OUTCROPS

AUTHORS

Fabiana Paola Montes Monterrey
Richard Leonardo Mogollón Mendoza

KEYWORDS

Path, Obstacle, Design, Strategy, 2D, 3D

DESCRIPCIÓN

This degree project of the mechatronics engineering program aims to develop a trajectory planning algorithm that includes obstacle avoidance of a quadrotor drone that will be used for the exploration and characterization of geological outcrops. These outcrops are rocks that are bare of vegetation and in some cases have complex shapes with different sizes, so one day it would be a suitable tool for the study of these outcrops.

The project is divided into two phases, the first the design phase where trajectory planning algorithms were created in known environments in two stages: an artificial force field (APF) model is created for each element in the environment using sigmoid functions, Subsequently, the trajectories will be calculated using strategies based on gradient descent. Two strategies were developed, the first based on mobile points that interconnect the drone with the goal, where later each point moves towards areas free of the influence of the following following the potential field, which makes pathways free of obstacles. The second is based on the use of the concept of safe zones, which is used as a criterion to update the position of the points.

3. INTRODUCCIÓN

En el estudio de la naturaleza se encuentran varias ramas como lo es la geología la cual nos ayuda a conocer el estado de los afloramientos geológicos y demás características del suelo, esto se convierte en una tarea muy tediosa y tanto complicada cuando no se conoce el terreno o es de difícil acceso como lo son varias partes de nuestro territorio nacional, siendo atravesado por la cordillera de los Andes y siendo principalmente de zonas montañosas y boscosas, lo cual complica la tarea de la caracterización de afloramientos geológicos a la hora de estudiarlos [1] [2].

En lo mencionado anteriormente podemos notar que el problema principalmente es el difícil acceso a las zonas para el estudio de los afloramientos geológicos, por lo cual la convocatoria Innova 2017 (reto 2), propone la solución de desarrollar una estrategia para vehículos no tripulados que nos permita acceder a estos sitios teniendo también como prioridad la evasión de obstáculos ya que al ser un terreno desconocido no sabemos a ciencia cierta que puede haber allí, en cuanto a estos vehículos es más fácil usar un dron debido a que es más compacto y puede realizar mejores capturas de la zona para su estudio, siendo la solución más óptima a la hora de caracterizar a fondo cualquier afloramiento geológico gracias a su maniobrabilidad y autonomía de vuelo. Las limitaciones que se presentan actualmente en los UAVs son varias, pero el enfoque principal del proyecto es seguir trayectorias y la evasión de obstáculos ya que no son entornos controlados los territorios a caracterizar siendo un reto amplio para el dron ya que para esta tarea debe ser capaz de responder de manera eficiente ante las situaciones que se presenten como la aparición de objetos en diferentes direcciones y dimensiones.

Este documento planteará una estrategia se desarrollan dos técnicas, basadas en una extrapolación del descenso del gradiente 2D tradicional al caso tridimensional, la primera se basa en puntos móviles que interconectan el Quadrotor con la meta, donde posteriormente cada punto se mueve hacia áreas libres de la influencia de obstáculos siguiendo el campo potencial, lo que hace que los caminos estén libres de obstáculos. El segundo se basa en la utilización del concepto de zonas seguras, que se utiliza como criterio para actualizar la posición de los puntos.

4. ANTECEDENTES

En las últimas décadas, un gran número de innovaciones tecnológicas han sido aplicadas para mejorar la movilidad del vehículo aéreo no tripulado inteligente. El progreso en los sensores, las comunicaciones y las estrategias prometen mejoras sustanciales al permitir a los UAVs operar de forma automática, seleccionar entre las posibles trayectorias, evasión de obstáculos y evitar colisiones. En términos generales, desde finales de los 60's se han publicado artículos sobre el tema de la planeación del movimiento [3], estos artículos tienen un elemento en común, el espacio continuo de movimiento es discretizado y a través de esto se crean algoritmos de búsqueda de soluciones gráficas, además, no se tenía en cuenta la dinámica del vehículo. En la actualidad se siguen utilizando ampliamente estos algoritmos, en particular en la planeación de trayectorias en escenarios conocidos. Existen varias revisiones exhaustivas sobre estados del arte en el tema de planeación de trayectorias y evasión de obstáculos para UAV, [4] presenta dos técnicas de planificación de trayectorias y una estrategia para la evasión de obstáculos que puedan ser implementadas en cuadricópteros para ambientes estáticos o controlados; el primer algoritmo o la primera técnica se conoce como el descenso del gradiente o método clásico (DG), la cual es aplicada de forma bidimensional y que usa una estrategia iterativa basada en campos potenciales. Esta estrategia consiste en formular el problema como un caso de optimización para descomponerlo en dos partes, la primera, generar una función objetivo y segundo el encontrar el mínimo de la función. El segundo algoritmo está basado en puntos independientes móviles el cual permite generar trayectorias libres de obstáculos entre dos puntos en el espacio. Está basado en el principio de campos potenciales artificiales y utiliza para su funcionamiento el concepto de zonas seguras. [5] esta tesis de la Universidad Nacional de Colombia nos presenta algoritmos para reconocer el entorno y posteriormente habiliten el desplazamiento de los vehículos dentro de él, realizando un proceso de evasión de obstáculos estáticos o dinámicos restringidos a velocidades constantes, este articulo tiene similitud con el anterior ya que trabajan las mismas estrategias y complementan con una cuarta para la planeación de trayectorias en ambientes dinámicos tridimensionales, llamado "algoritmo de banda móvil". Aunque el principio de funcionamiento es igualmente aplicable con cualquier técnica basada campos potenciales artificiales presentadas previamente, para el caso de esta tesis, se trabaja sobre la delimitación de obstáculos dinámicos que solo tienen movimientos rectilíneos con velocidad constante y adicionalmente su trayectoria es siempre conocida, restricción generada debido a la elevada complejidad que tendría extender el método a obstáculos que tengan movimientos acelerados o con cambios de dirección arbitrarios. Por otro lado, existen situaciones en las que un agente robótico debe moverse en un entorno desconocido, pero cuenta con información precisa sobre su ubicación. [6] En este documento se hace uso del sistema de posicionamiento más utilizado y difundido a nivel mundial, como lo es el GPS, se desarrolló una arquitectura GNC (Guidance, Navigation and control) que se encargó de realizar y

controlar la planeación de la trayectoria, el control y la navegación del Quadrotor. Esto se hizo combinando dos conceptos: GPS Waypoints y Perfil de velocidad trapezoidal. Para el desarrollo de este algoritmo se deben tener en cuenta primero una serie de datos que se obtienen a partir de coordenadas GPS, puesto que el algoritmo consiste en: primero ingresar una cantidad de puntos (waypoints) por los que se desea que el vehículo se desplace, pues de ello depende el número de iteraciones a realizar. De igual forma se debe conocer la cantidad de puntos de ruta en coordenadas UTM (Universal Transversal Mercator), con el fin de encontrar o definir puntos válidos, para entrar a un bucle donde se realicen los cálculos iterativos y encontrar una trayectoria entre punto y punto. Esto para aplicarlo a la generación y planeación de trayectorias. Sin embargo, existen casos, aún más complejos, en los que no se conoce el entorno y tampoco se tiene información sobre la ubicación exacta del UAV. En este caso el UAV deberá generar un mapa y mantener su ubicación en el mismo de forma concurrente. Esta tarea resulta compleja por el hecho que para poder localizarse de forma precisa se necesita un mapa, y, por otro lado, para poder crear un mapa es menester estar localizado en forma precisa. Esta es la tarea que estudia el SLAM, localización y armado de mapas simultáneo [7]. La investigación acerca de SLAM ha sido un tema de investigación desde 1975 y un tema activo desde 1985. Sin embargo, el 83.9% de las investigaciones sobre el tema se encuentran en el periodo 2000-2014, según un análisis semántico. Con la introducción del filtro de Kalman extendido en 2000 se logra resolver el problema SLAM en 2001, [8] para ambientes estáticos, cerrados y con el uso de varias cámaras. Dada la trayectoria del problema SLAM en la comunidad investigativa, se han formulado múltiples soluciones, cada una orientada hacia el uso de un tipo específico de sensor o la generación de mapas con estructuras definidas. [9] Este trabajo se orienta hacia las técnicas SLAM que hacen uso de sensores RGB-D y que generan mapas voxelized, en esta área de desarrollo se usan actualmente 2 técnicas independientes, basados en el alineamiento de características visuales (Keypoint Alignement) [10], [11], [12], [13] y los que se basan en la fotoconsistencia visual (Dense visual odometry) [14], [15], [16].

5. OBJETIVOS

5.1. Objetivo general

Desarrollar una estrategia de generación de trayectorias con evasión de obstáculos en un dron tipo Quadrotor usado en la exploración y caracterización de afloramientos geológicos.

5.2. Objetivos específicos

- Investigar el estado del arte de las estrategias para la generación de trayectorias y evasión de obstáculos para UAVs.

- Diseñar dos estrategias de planeación de trayectorias que incluya evasión de obstáculos haciendo uso de métodos de optimización o técnicas de aprendizaje automático.

- Seleccionar la estrategia con mejor desempeño de acuerdo con los resultados obtenidos en las etapas de diseño y simulación.

- Validar por medio de simulación en un entorno virtual la estrategia seleccionada.

- Implementar la estrategia seleccionada en el entorno del Quadrotor. Validar experimentalmente la estrategia mediante los resultados obtenidos en la etapa de simulación en un ambiente controlado.

- Validar experimentalmente las estrategias implementadas en el Quadrotor en un ambiente controlado y uno no controlado (campo abierto).

6. METODOLOGÍA DEL PROYECTO

Para el desarrollo del proyecto se plantea la siguiente metodología, donde se tienen dos etapas principales: **desarrollo** y **validación**. Cada una tiene sus correspondientes subetapas respectivamente. En caso de que durante el desarrollo del proyecto no se cumplan los requerimientos y resultados esperados se procede a volver a las etapas anteriores para correcciones y replanteamientos en el diseño, simulación o etapa que corresponda. Por último, se realiza la correspondiente recolección y redacción documental para recopilación del informe final. Dicha metodología se puede observar la Figura 1.

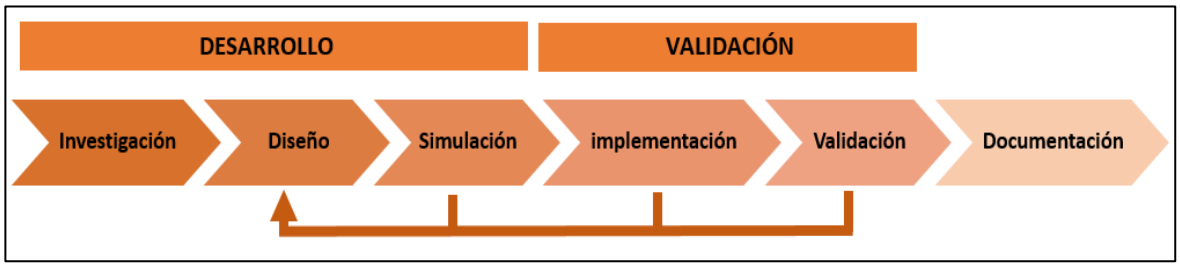


Figura 1 .Metodología del proyecto [25].

En la **primera etapa** de investigación procedemos a realizar la respectiva búsqueda bibliográfica necesaria para el proyecto, luego en la **segunda etapa** de diseño de las estrategias inicialmente realizamos un pseudocódigo de las estrategias a implementar y luego diseñar las estrategias de planeación de trayectorias y evasión de obstáculos de manera independiente y ya realizado el diseño, se procede a programar en software las estrategias anteriormente diseñadas para la **tercera etapa** de simulación y familiarización con ellas de forma independiente. A continuación, se unifican las estrategias de evasión y generación de trayectorias y finalmente realizar pruebas simuladas en un entorno predefinido para validar el funcionamiento de la estrategia.

En la **etapa cuatro** se investiga acerca de los sensores más adecuados según la estrategia que se utilice, ya seleccionados procedemos a adquirirlos y realizar la curva de aprendizaje de su funcionamiento. Se realizan pruebas para la validación del funcionamiento del dron con su configuración por defecto. Luego se procede a implementar el algoritmo de la estrategia unificada previamente desarrollada.

Para la **etapa cinco** se realizan pruebas para lo cual es necesario verificar una serie de ítems para tener éxito en las misiones, luego se realizan las pruebas en campo abierto en un entorno controlado definiendo los obstáculos. Con eso se recolectan datos y se realiza un documento final.

7. DISEÑO DE ESTRATEGIAS DE PLANEACIÓN Y EVASIÓN

7.1. Planificación de Trayectorias

La planificación de trayectorias es la búsqueda de una sucesión de posiciones para un robot, que permitirán llevarlo desde un estado inicial a uno final, entendiéndose por estado a la descripción de la ubicación del robot referida a un marco de referencia absoluto.

La configuración que adquiere una determinada trayectoria queda definida por la distribución de los obstáculos a lo largo de todo el espacio de trabajo, y por supuesto, por la geometría del robot y sus capacidades de movimiento. De esta manera, la topología del ambiente de trabajo restringirá el espacio libre de obstáculos en el cual se pueden expresar las posibles trayectorias para alcanzar el estado final deseado.

Generalmente, se recurre a una representación realizada a partir de la discretización del espacio del ambiente de trabajo, con lo que se extrae una representación segura, es decir, se tendrá la garantía de que el espacio libre podrá ser ocupado por el robot (sin riesgo de colisión), por lo tanto, es necesario que tal discretización se haga en base a las características geométricas, tanto del robot como de los obstáculos [18].

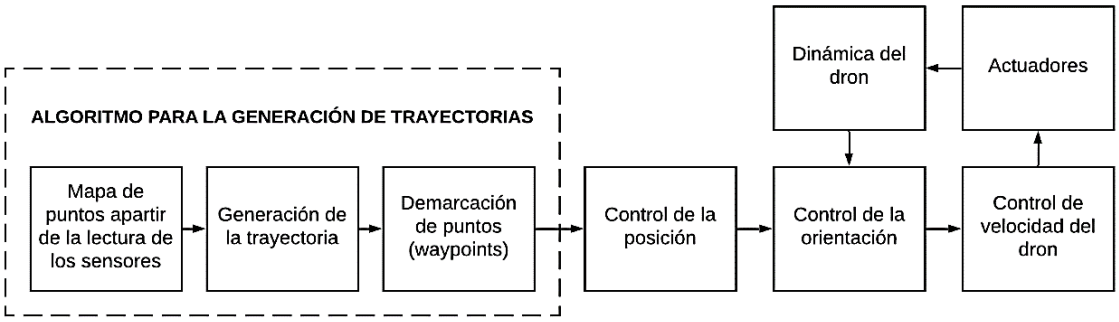


Figura 2. Diagrama funcional de la planeación de trayectorias [25].

7.2. Tecnicas para la planeación y evasión

Basado en los antecedentes, se realizaron las siguientes tablas donde se puede observar las ventajas y desventajas de las tecnicas mas relevantes para nuestro proyecto.

<table>
<tr><th colspan="2">Técnica del descenso del gradiente o método clásico</th></tr>
<tr><td>Ventajas</td><td>Desventajas</td></tr>
<tr><td rowspan="3">Si se seleccionan valores correctos de distancia de avance por iteración, pesos relativos de las funciones de obstáculos, meta y factor de ruido se puede obtener un contraste entre la suavidad de la trayectoria y la complejidad computacional del algoritmo.</td><td>Para algunos escenarios se presenta la posibilidad de que el algoritmo se quede en un mínimo local y por este motivo nunca encuentre el punto de destino.</td></tr>
<tr><td>No es posible mantener un control directo sobre la distancia a la que se acerca a los obstáculos, pudiendo esto aumentar la probabilidad de hacer colisión en algún momento</td></tr>
<tr><td>cuando el punto de meta está muy cerca de un obstáculo, lo que produce que el algoritmo no converja.</td></tr>
</table>

Tabla 1. Ventajas y desventajas de la tecnica del descenso del gradiente [25].

<table>
<tr><th colspan="2">Técnica basada en la permanencia en la zona segura</th></tr>
<tr><td>Ventajas</td><td>Desventajas</td></tr>
<tr><td>Permite controlar la distancia mínima que tendrán los puntos de la trayectoria a los obstáculos.</td><td rowspan="3">Presenta dificultades para la sintonización de sus parámetros, debido a que para lograr convergencia el valor del umbral de la zona segura debe ser bajo.</td></tr>
<tr><td>No genera oscilaciones inestables ya que los puntos de la trayectoria tienden a quedar sobre la frontera de la zona segura.</td></tr>
<tr><td>La distancia de la trayectoria es menor en comparación con la segunda técnica presentada.</td></tr>
</table>

Tabla 2. Ventajas y desventajas de la técnica basada en zonas seguras [25].

Técnica basada en puntos independientes móviles	
Ventajas	Desventajas
No cae en los mínimos locales debido a que no actualiza la dirección del gradiente en cada iteración.	En contraste la complejidad computacional aumenta proporcionalmente al número de puntos que se utilicen.
Cada punto de la trayectoria se desplaza siguiendo su propia dirección del gradiente la cual mantiene hasta que se llegue a la zona segura.	
La técnica se puede paralelizar ya que la actualización de un punto no depende de sus puntos vecinos.	
No se presentan oscilaciones inestables ya que el modelo de los objetos genera un campo continuo a través de todo el espacio, lo que hace que los puntos de la trayectoria tiendan a quedar en el límite de la zona segura, que a su vez es continua en el espacio.	

Tabla 3. Ventajas y desventajas de la técnica basada en puntos independientes móviles [25].

SLAM	
Ventajas	Desventajas
Hace una buena aproximación de la zona gracias a visión artificial o sensorica.	Necesita hardware sofisticado para poder hacer un buen mapeo.
Es más exacto para el desarrollo de las trayectorias y la evasión de obstáculos.	Requiere bastante capacidad de procesamiento para realizar el proceso.
Es ampliamente usado en el tema de robots e inteligencia artificial.	No es muy claro la manera de desarrollar un SLAM optimo ya que al ser mundialmente usado es más una filosofía o una base de diseño más no nada en específico.

Tabla 4. Ventajas y desventajas de la técnica SLAM [25].

Teniendo en cuenta las tablas anteriores, se decidió integrar las diferentes técnicas para obtener dos algoritmos, los cuales se diseñarán y simularán paralelamente para finalmente escoger el mejor y el que finalmente se implementará más adelante. Las técnicas que se desarrollaran estarán basadas en el gradiente, con esta técnica obtendremos la dirección en las zonas donde se esté libre de influencia de obstáculos y para la toma de decisiones a la hora de enfrentarse con un obstáculo se usaran los conceptos de zonas seguras y puntos independientes móviles.

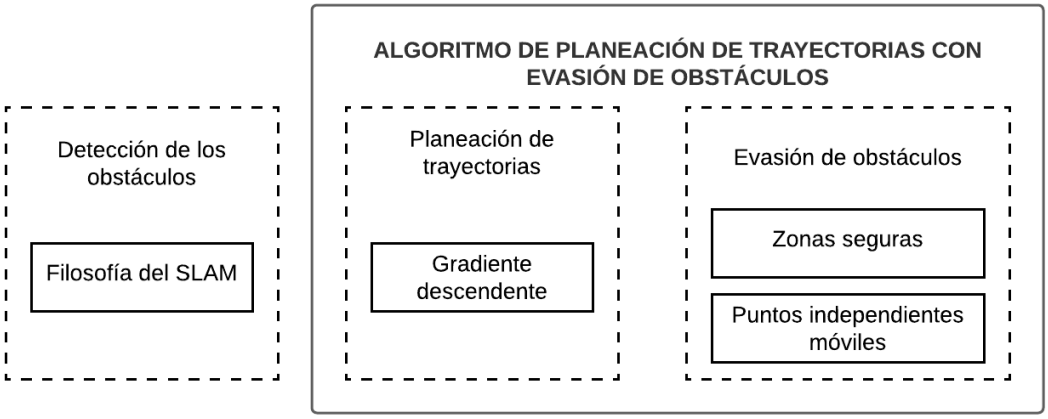


Figura 3. Diagrama explicativo de los algoritmos a trabajar [25].

7.3. Representación del entorno

7.3.1. El método APF (Artificial Potential Field):

Fue originalmente propuesto por Oussama Khatib in 1986 [20]. Se basa en la construcción de un campo escalar que comprende colinas artificiales que representan obstáculos y valles que representan atractores. El gradiente del campo potencial genera las fuerzas de repulsión y de atracción apropiadas para garantizar la evasión de obstáculos y para permitir que se pueda alcanzar el punto de destino. El método APF presenta algunos inconvenientes como mínimos locales en ciertas configuraciones del ambiente, oscilaciones en la ruta encontrada, problema de convergencia si la meta está muy cerca a un obstáculo. Adicionalmente puede no funcionar para formas arbitrarias de los obstáculos debido a que los modela como un elemento puntual.

7.3.2. Representación de los obstáculos

El dron se puede desplazar sobre un ambiente en el cual podrá encontrarse con diversos obstáculos, por eso es necesario encontrar una representación matemática que permita determinar cuáles son las zonas a través de las cuales se puede circular libremente, cuales zonas presentarían un alto riesgo de colisión y en qué puntos la

colisión en inminente. La estrategia para generar un modelo del ambiente seleccionada en este caso está basada en campos potenciales.

Utilizando funciones sigmoides se puede generar un campo potencial en todo el espacio dependiendo del lugar y la forma de los elementos que componen el ambiente. Se considera el uso de funciones sigmoides debido a que permite crear un modelo suave de campo de fuerza en todo el ambiente de diseño y solo es necesario ajustar un parámetro para cambiar el rango de acción efectiva del campo generado de manera individual por cada obstáculo [5].

7.3.3. Representación en dos dimensiones

Para la representación del entorno y los obstáculos en el caso bidimensional se debe tener en cuenta la Ecuación 1, en la que Xobs representa el valor en X o Y del obstáculo y *gamma* es un coeficiente que permite modificar la suavidad del cambio de la función, el signo de *gamma* determina si la función monótona es creciente o decreciente.

$\beta: factor\ de\ escala$
$\gamma: coeficiente\ de\ suavidad$
$X_{ini}: posición\ en\ X$
$Y_{ini}: posición\ en\ Y$
$X_{obs}: posición\ en\ X\ de\ los\ obstáculos$
$Y_{obs}: posición\ en\ Y\ de\ los\ obstáculos$

$$f_{sig}(x) = \frac{\beta}{1 + e^{-\gamma(X_{ini} - X_{obs})}} * \frac{1}{1 + e^{-\gamma(Y_{ini} - Y_{obs})}} * \frac{1}{1 + e^{\gamma(X_{ini} - X_{obs})}} * \frac{1}{1 + e^{\gamma(Y_{ini} - Y_{obs})}} \quad (1)$$

Para hallar el gradiente descendiente de la funcion sigmoide se deriva *Fsig* en respecto a cada eje X y Y.

Derivada de *Fsig* con respecto a X:

$$Grad_X(x) = \beta * (\gamma * (e^{-\gamma(X_{ini} - X_{obs})} - e^{\gamma(X_{ini} - X_{obs})}) \quad (2)$$

$$Grad_X(x) = \frac{Grad_X}{1 + e^{\gamma(Y_{ini} - X_{obs})} * 1 + e^{-\gamma(Y_{ini} - X_{obs})}} \quad (3)$$

$$Grad_X(x) = \frac{Grad_X}{(1 + e^{-\gamma(X_{ini} - X_{obs})})^2 * (1 + e^{\gamma(X_{ini} - X_{obs})})^2} \quad (4)$$

Derivada de *Fsig* con respecto a Y:

$$Grad_y(y) = \beta * (\gamma * (e^{-\gamma(Y_{ini}-X_{obs})} - e^{\gamma(Y_{ini}-X_{obs})}) \quad (5)$$

$$Grad_y(y) = \frac{Grad_y}{1 + e^{\gamma(X_{ini}-X_{obs})} * 1 + e^{-\gamma(X_{ini}-X_{obs})}} \quad (6)$$

$$Grad_y(y) = \frac{Grad_y}{(1 + e^{-\gamma(Y_{ini}-X_{obs})})^2 * (1 + e^{\gamma(Y_{ini}-X_{obs})})^2} \quad (7)$$

Teniendo la derivada con respecto a cada eje, se acomoda el vector $Grad_{sig}$:

$$Grad_{sig} = -\begin{bmatrix} Grad_X \\ Grad_y \end{bmatrix} \quad (8)$$

Este vector se normaliza:

$$Grad_{sig} = \frac{Grad_{sig}}{|Grad_{sig}|} \quad (9)$$

Con el vector normalizado se forma el nuevo vector V_{sig}:

$$V_{sig} = \begin{bmatrix} Grad_{sig}(x) \\ Grad_{sig}(y) \end{bmatrix} \quad (10)$$

Al cual se le halla un vector perpendicular V_{90}:

$$V_{90} = \begin{bmatrix} -Grad_{sig}(y) \\ Grad_{sig}(x) \end{bmatrix} \quad (11)$$

Finalmente con el uso de funciones sigmoides y por medio de la herramienta Matlab se logra representar un obstáculo como se muestra en la Figura 4, en este tipo de representaciones se puede observar el valor de nivel que proporciona dependiendo que tan cerca se esté del obstáculo.

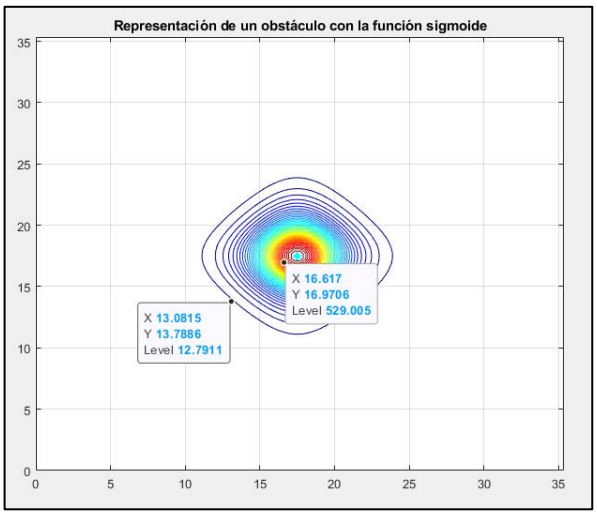


Figura 4. Representación en dos dimensiones de un obstáculo con los valores de nivel [25].

7.3.4. Representación en tres dimensiones

En un caso tridimensional la funcion sigmoide representa el campo potencial artificial generado por un obstaculo en un punto en el espacio y se representa de la misma manera que en dos dimensiones, añadiendo el eje Z.

$\beta: factor\ de\ escala$
$\gamma: coeficiente\ de\ suavidad$
$X_{ini}: posición\ en\ X$
$Y_{ini}: posición\ en\ Y$
$X_{obs}: posición\ en\ X\ de\ los\ obstáculos$
$Y_{obs}: posición\ en\ Y\ de\ los\ obstáculos$

$$f_{sig}(x) = \frac{\beta}{1+e^{-\gamma(X_{ini}-X_{obs})}} * \frac{1}{1+e^{-\gamma(Y_{ini}-X_{obs})}} * \frac{1}{1+e^{-\gamma(Z_{ini}-X_{obs})}} * \frac{1}{1+e^{\gamma(X_{ini}-X_{obs})}} * \frac{1}{1+e^{\gamma(Y_{ini}-X_{obs})}} * \frac{1}{1+e^{\gamma(Z_{ini}-X_{obs})}} \quad (12)$$

$$Grad_x(x) = \beta * (\gamma * (e^{-\gamma(X_{ini}-X_{obs})} - e^{\gamma(X_{ini}-X_{obs})}) \quad (13)$$

$$Grad_x(x) = \frac{Grad_x}{1+e^{\gamma(Y_{ini}-X_{obs})} * 1+e^{-\gamma(Y_{ini}-X_{obs})}} \quad (14)$$

$$Grad_x(x) = \frac{Grad_x}{1 + e^{\gamma(Z_{ini} - X_{obs})} * 1 + e^{-\gamma(Z_{ini} - X_{obs})}} \quad (15)$$

$$Grad_x(x) = \frac{Grad_x}{(1 + e^{-\gamma(X_{ini} - X_{obs})})^2 * (1 + e^{\gamma(X_{ini} - X_{obs})})^2} \quad (16)$$

$$Grad_y(y) = \beta * (\gamma * (e^{-\gamma(Y_{ini} - X_{obs})} - e^{\gamma(Y_{ini} - X_{obs})}) \quad (17)$$

$$Grad_y(y) = \frac{Grad_y}{1 + e^{\gamma(X_{ini} - X_{obs})} * 1 + e^{-\gamma(X_{ini} - X_{obs})}} \quad (18)$$

$$Grad_y(y) = \frac{Grad_y}{1 + e^{\gamma(Z_{ini} - X_{obs})} * 1 + e^{-\gamma(Z_{ini} - X_{obs})}} \quad (19)$$

$$Grad_y(y) = \frac{Grad_y}{(1 + e^{-\gamma(Y_{ini} - X_{obs})})^2 * (1 + e^{\gamma(Y_{ini} - X_{obs})})^2} \quad (20)$$

$$Grad_z(z) = \beta * (\gamma * (e^{-\gamma(Zini - Xobs)} - e^{\gamma(Zini - Xobs)}) \quad (21)$$

$$Grad_z(Z) = \frac{Grad_z}{1 + e^{\gamma(X_{ini} - X_{obs})} * 1 + e^{-\gamma(X_{ini} - X_{obs})}} \quad (22)$$

$$Grad_z(Z) = \frac{Grad_z}{1 + e^{\gamma(Y_{ini} - X_{obs})} * 1 + e^{-\gamma(Y_{ini} - X_{obs})}} \quad (23)$$

$$Grad_z(Z) = \frac{Grad_z}{(1 + e^{-\gamma(Z_{ini} - X_{obs})})^2 * (1 + e^{\gamma(Z_{ini} - X_{obs})})^2} \quad (24)$$

Teniendo la derivada con respecto a cada eje, se acomoda el vector $Grad_{sig}$:

$$Grad_{sig} = \begin{bmatrix} Grad_X \\ Grad_Y \\ Grad_z \end{bmatrix} \quad (25)$$

Este vector se normaliza:

$$Grad_{sig} = \frac{Grad_{sig}}{|Grad_{sig}|} \quad (26)$$

Con el vector normalizado se forma el nuevo vector V_{sig}:

$$V_{sig} = \begin{bmatrix} Grad_{sig}(x) \\ Grad_{sig}(y) \\ Grad_{sig}(z) \end{bmatrix} \quad (27)$$

Al cual se le halla un vector perpendicular V_{90}:

$$V_{90} = \begin{bmatrix} -Grad_{sig}(y) \\ Grad_{sig}(x) \\ 0 \end{bmatrix} \quad (28)$$

Ya teniendo esta representación matemática organizada en una función del herramienta de simulación que se esta usando, en este caso MATLAB, se procede a concatenar los datos para obtener todos los valores de la función objetivo para ser analizados punto a punto. Se observa en la Figura 5 la representación de 4 obstáculos en diferentes puntos y su respectivos campos potenciales.

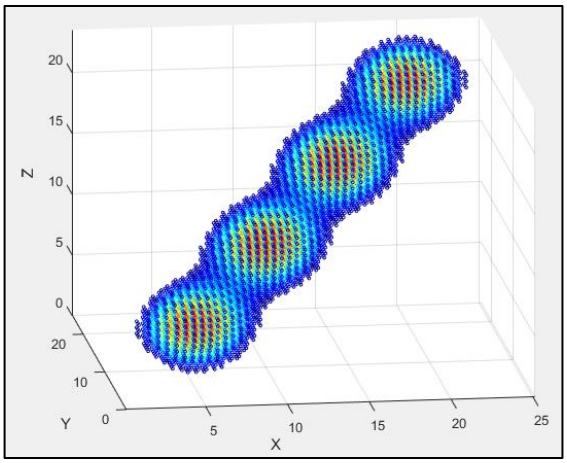


Figura 5. Representación de campo potencial de 4 obstáculos [25].

7.4. Planeación de trayectorias

En este sección se presenta el desarrollo de la planeación de trayectorias utilizando campos potenciales artificiales, lo cual trae como resultado el desarrollo de dos algoritmos, presentados en ambientes estáticos. El primer algoritmo puntos independientes móviles, es una técnica basado en el movimiento independiente de una serie de puntos que buscan moverse hasta encontrar una región libre de obstáculos. El segundo, zonas seguras, donde se busca descender a través del gradiente sin salir de la zona segura.

7.4.1. Gradiente descendente:

El método del gradiente descendente trata de encontrar el mínimo de una función, sea local o global. El método se basa en el uso del gradiente negativo, pues en esa dirección obtendremos el descenso máximo en los valores de la función.

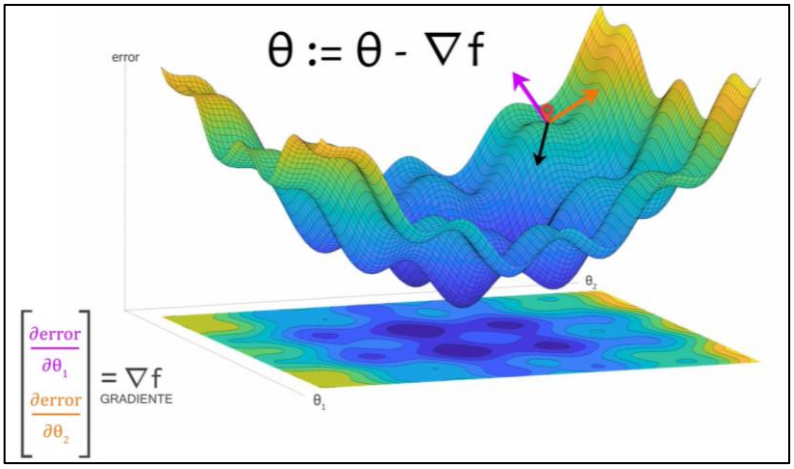


Figura 6. Representación de una función de coste [19].

Como se puede observar en la Figura 6. el eje X y Y son parámetros, y el eje Z es el error del modelo, para hallar el gradiente descendente se calcula la derivada de la función en dicho punto, como la función es tridimensional equivale a calcular la derivada parcial para cada uno de los parámetros y cada uno de estos valores nos indica la pendiente en el eje de dicho parámetro, conjuntamente todas estas derivadas parciales conforman un vector que nos indica la dirección a donde asciende el gradiente, al invertir este vector obtendríamos el descenso del gradiente y se realizara esto cíclicamente hasta que la pendiente sea próxima a nula. para completar este algoritmo se necesita añadir un parámetro más denominado tasa (ratio) de aprendizaje (α) que define cuanto se avanza en cada paso, definiendo así el comportamiento del algoritmo, si se utiliza un valor muy pequeño se podría afectar el comportamiento del algoritmo ya que calcularía una gran cantidad de iteraciones y esto podría tomar más tiempo o podría quedar atrapado con mayor facilidad en un

mínimo local. Y si por el contrario el valor es muy grande el algoritmo podría llegar a no converger [19].

$\theta: variable$
$\alpha: tamaño\ del\ paso$
$\nabla f: gradiente$

$$\theta = \theta - \alpha\, \nabla f \qquad (29)$$

7.4.1.1. Algoritmo base o descenso del gradiente

Ya definida la dirección a la cual avanzar para descender, se tiene que definir cuánto descender. Una estrategia sería si la función crece “mucho” descender mucho, y si crece “poco” descender poco. Es decir dar el paso en forma proporcional al gradiente. Para poder controlar la proporción del paso utilizaremos un parámetro alpha de tal forma que el paso a dar se cuantifica.

Dada una posición Xini se avanza a una nueva posición Xini+1 que dependerá del paso que demos, tamaño y dirección. Para cada nueva posición podemos medir la diferencia entre Xini+1 y Xini si esta es menor que cierto umbral o si alcanzamos un máximo de iteraciones, podemos decidir que hemos alcanzado un punto mínimo.

$Xgoal: posición\ final$
$Xini: posición\ inicial$
$B: tamaño\ del\ paso$

$$F_{goal} = \sum \left(X_{goal} - X_{ini}\right)^2 \qquad (30)$$

$$F_{grad} = 2 * \left(X_{goal} - X_{ini}\right) \qquad (31)$$

$$X = X_{ini}(k) + B * F_{grad} \qquad (32)$$

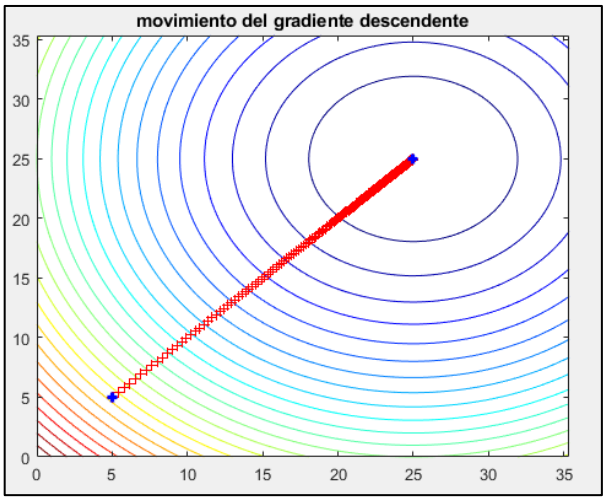


Figura 7. Recorrido iterativo del descenso del gradiente [25].

En la Figura 7 se puede observar el recorrido iterativo de las nuevas posiciones. En un principio cada paso es más agresivo porque el gradiente es mayor, y conforme nos acercamos al punto mínimo los pasos son más pequeños. Inicialmente se definio 500 pasos y cada uno de 0,1.

7.4.2. Zona segura:

La zona segura es un concepto abstracto que se utiliza como criterio para tomar decisiones en las técnicas de planeación de trayectorias.

Una vez se tiene la función analítica que permite calcular el APF[1] de cada obstáculo, es posible identificar regiones seguras, las cuales se pueden interpretan como puntos en el espacio sobre los cuales no existirá colisión con ningún obstáculo si el quadrotor se ubicara allí. Las zonas seguras se caracterizan por tener un valor de APF muy bajo que tiende a 0 y se seleccionará de acuerdo con que tanto se desea que el quadrotor se acerque a los obstáculos cuando esté navegando cerca de ellos, la selección de esta distancia dependerá del desempeño que tenga en el control del sistema para seguir la referencia dada por la trayectoria.

Se puede determinar la zona segura fijando con valor de umbral U_{safe}, si el valor de APF calculado para un punto q es menor al valor de U_{safe}, se puede afirmar que el punto se encuentra en la zona segura. Si se selecciona valor U_{safe} muy alto es posible que el quadrotor pase muy cerca de los obstáculos y corra peligro de colisión, si el valor es muy bajo el vehículo intentará evitar los obstáculos alejándose una gran distancia de ellos lo que implicará que la trayectoria encontrada puede tener una distancia de recorrido mucho más grande.

El tamaño del quadrotor puede ser tenido en cuenta extendiendo las dimensiones de la frontera de cada obstáculo de tal forma que pueda ser considerado simplemente como un punto, basta con dilatar la frontera de los obstáculos de acuerdo con las dimensiones del quadrotor [5].

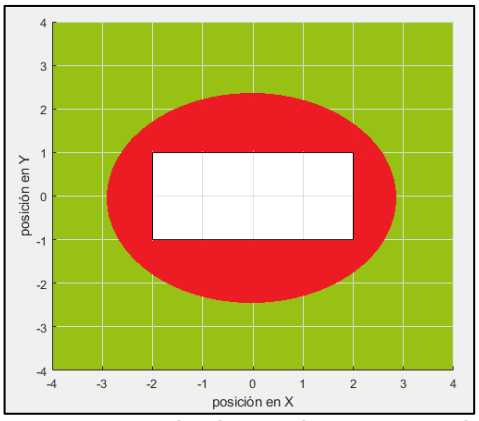


Figura 8. Visualización de la zona segura, el color verde representa la región que corresponde a la zona segura, el color rojo es una región fuera de la zona segura, el color blanco es el interior del obstáculo [5].

[1] APF: definido en el español como "campo de potencial artificial" es la combinación entre el campo potencial atractivo con los campos potenciales repulsivos de los obstáculos.

7.4.2.1. Algoritmo basado en la permanencia en la zona segura

Cuando un punto k se encuentra dentro de la región en el espacio denominada zona segura, se procede a encontrar la posición para el punto de la trayectoria k + 1, para esto se calcula el vector gradiente que dará la dirección del movimiento, el cual está dado por la dirección de la recta que interconecta el punto k con el punto objetivo.

Por el contrario, cuando el punto k se encuentra fuera de la región en el espacio denominada zona segura, se procede a utilizar los valores del APF para determinar un vector perpendicular al campo. Este vector se calcula encontrando la derivada parcial en cada dirección sobre un punto seleccionado y posteriormente normalizándolo. Ahora se inicia un proceso de búsqueda en el cual, desde la posición actual del punto k comienza un desplazamiento siguiendo la dirección del vector gradiente, pero con un movimiento perpendicular a esta recta hasta el momento en el que el punto se encuentre dentro de la zona segura, una vez ahí solo se tendrá en cuenta el movimiento en gradiente.

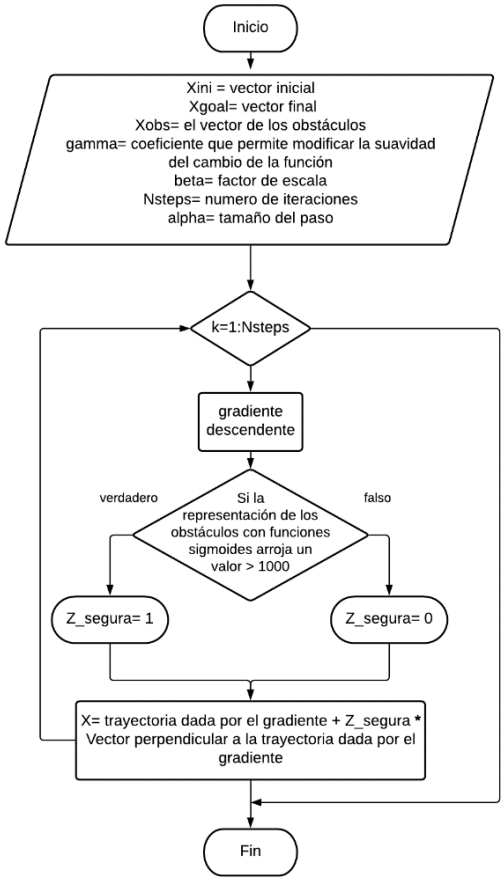


Figura 9. Diagrama de flujo de algoritmo de zonas seguras [25].

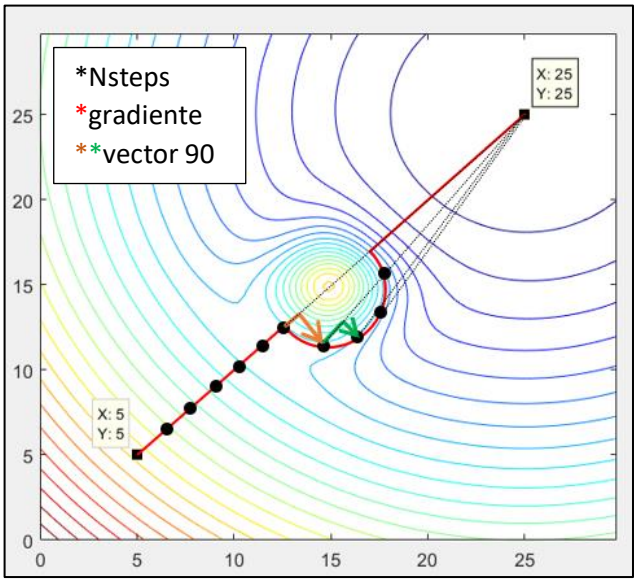


Figura 10. Recorrido paso a paso de la trayectoria de zonas seguras [25].

Nota: tener en cuenta que para el diseño y simulación tanto el dron como los obstáculos se toman como puntos en el espacio.

Entrada: *punto inicial* $\boldsymbol{X_{ini}}$ *, punto objetivo* $\boldsymbol{X_{goal}}$ *y punto obstaculo* $\boldsymbol{X_{goal}}$

Salida: *vector* $\boldsymbol{X}$

1: $\boldsymbol{N_{steps}}$ *numero maximo de iteraciones*

2: $\boldsymbol{\gamma}$ *coeficiente de suavidad*

3: $\boldsymbol{\beta}$ *factor de escala*

4: $\boldsymbol{\alpha}$ *tamaño del paso*

5: $\boldsymbol{U_{safe}}$ *Umbral del zona segura*

5: ***Por*** $K=1: N_{steps}$

6: $F_{goal} = \sum(X_{goal} - X_{ini})^2$ *planeación con el gradiente*

7: $F_{grad} = 2 * (X_{goal} - X_{ini})$ *planeación con el gradiente*

8: ***SI F_ZS*** *(Función que evalua los obstáculos) >* U_{safe} *(valor definido por el usuario)*

9: Z_segura= 1

10: ***SINO***

11: Z_segura= 0

12: ***FIN SI***

13: $Xa = X_{ini}(k) + \beta * F_{grad}$ *planeación con el gradiente*

*14: X = Xa + Z_segura*F_ZS (vector 90) (un vector perpendicular a la trayectoria original)*

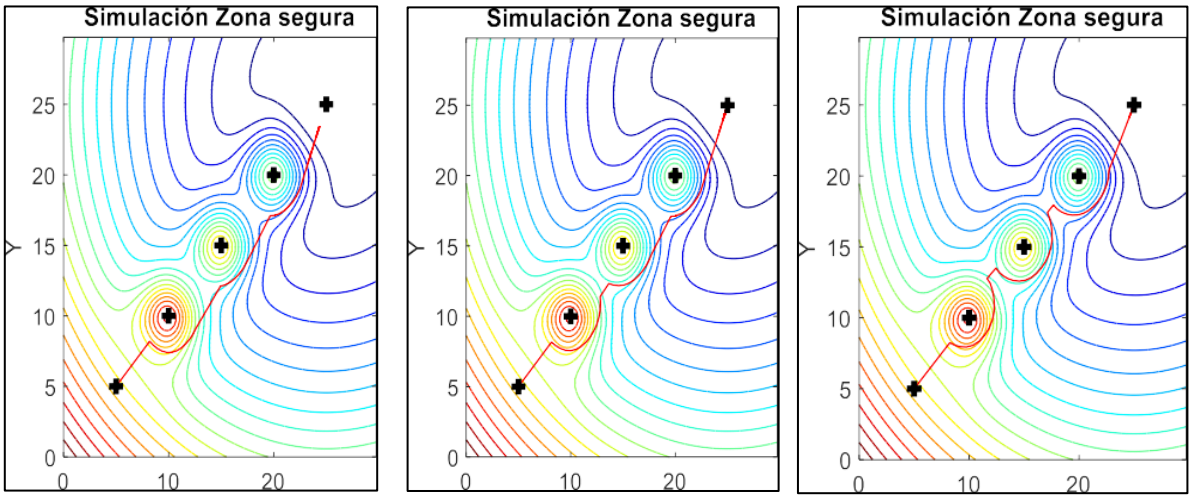
15: ***Fin Por***

7.4.2.2. Resultados de la simulación de puntos Zonas seguras en dos dimensiones

En esta sección se presentan las simulaciones en varios escenarios posibles para el algoritmo de zonas seguras sobre campos potenciales. Inicialmente se presenta un análisis de los efectos de la variación de los parámetros del algoritmo, posteriormente se presentan algunos escenarios en los cuales se presenta la convergencia del algoritmo.

Efectos de la variación de parámetros: En este algoritmo los parámetros que influye en el resultado son los siguientes.

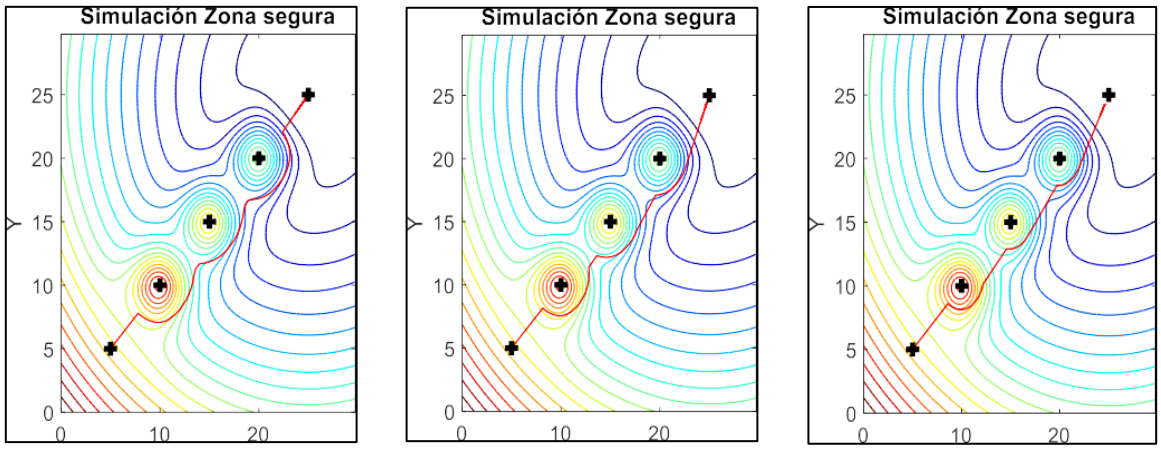
En la Figura 11 se observa la variación del radio de detección (***r***) en el algoritmo de zonas seguras, se puede notar que este parámetro es bastante susceptible a los cambios ya que con el valor justo en este caso 0.002 podemos ver una trayectoria suave que converge y evade los obstáculos satisfactoriamente, pero si el valor es menor notamos que la trayectoria no llega a su punto objetivo y si es mayor realiza un movimiento que puede provocar una posible colisión.



a) r=0,001 *b) r=0,002* *c) r=0,004*

Figura 11. Simulación variando el radio de detección [25].

Como ya se comentó el umbral de zona segura es el valor el cual vamos a comparar para tomar la decisión de cual es la dirección libre de obstáculos, como se observa en la Figura 12, este parámetro afecta la distancia la cual nos acercaremos al obstáculo. Este valor de umbral es criterio del usuario dependiendo del tamaño del dron y de que tan lejos queramos pasar del obstáculo.



a) U_{safe}:150 b) U_{safe}:230 c) U_{safe}:340

Figura 12. Simulación de la varición del umbral de zona segura [25].

7.4.2.3. Resultados de convergencia del algoritmo

Esta simulación presentada en la Figura 13 presenta una trayectoria muy optima, ya que como se observa el algoritmo converge al punto meta con variaciones muy pequeñas, no presenta oscilaciones, evade los obstáculos con un espacio prudencial y su recorrido es corto.

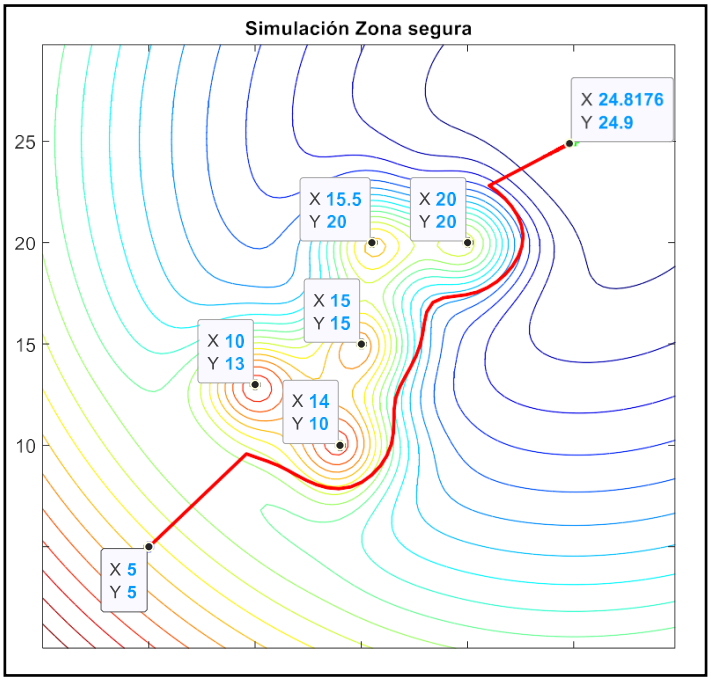


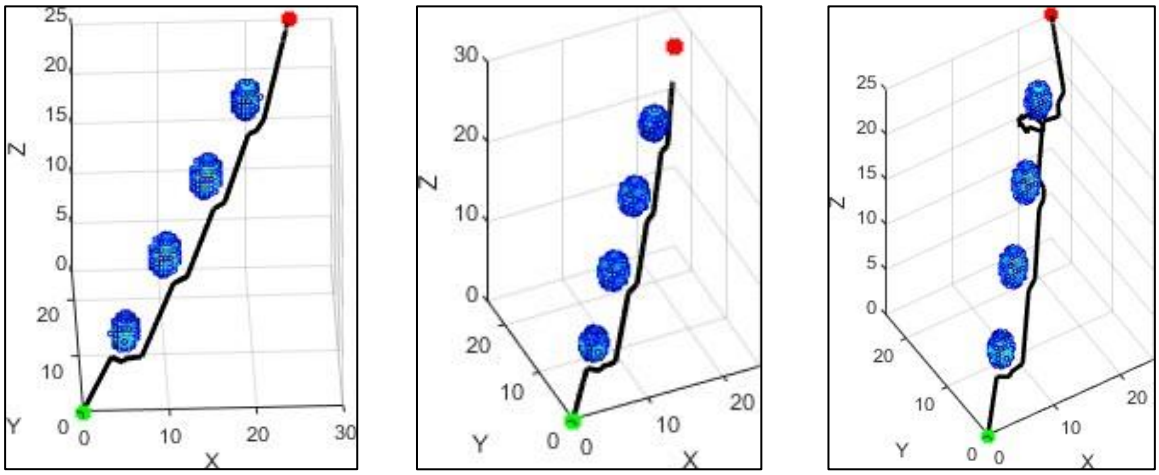
Figura 13. Simulación de la trayectoria de zonas seguras [25].

7.4.2.4. Resultados de la simulación de zonas seguras en tres dimensiones

Igual que en la sección anterior se presentan las simulaciones en varios escenarios 3D posibles para el algoritmo de zonas seguras sobre campos potenciales. Inicialmente se presenta un análisis de los efectos de la variación de los parámetros del algoritmo, posteriormente se presentan algunos escenarios en los cuales se presenta la convergencia del algoritmo.

Efectos de la variación de parámetros: En este algoritmo los parámetros que influye en el resultado son los siguientes.

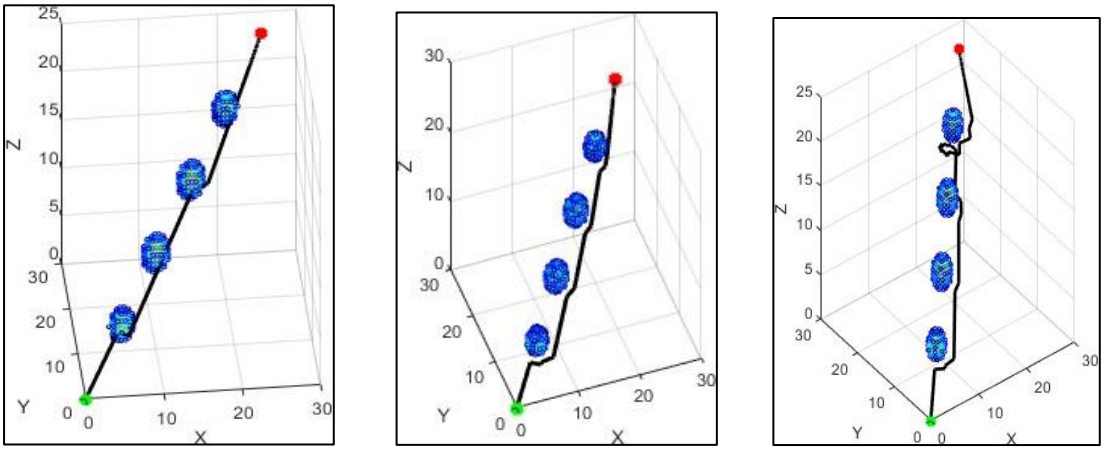
Como se observa en la Figura 14 el radio de detección es un parámetro muy susceptible a los cambios, ya que como se puede notar con mínimas variaciones puede que la trayectoria no converja o comienza a tomar direcciones equivocadas.



a) r:0.004 *b) r:0.002* *c) r:0.005*

Figura 14. Simulacion de la variación del radio de detección [25].

El la Figura 15 se observa que al igual forma que en 2D el valor de umbral es un valor de gran importancia para definir la suavidad de la trayectoria y la distancia entre el dron y los obstáculos. Este valor puede llegar a un limite en el cual la trayectoria pierde su rumbo y presenta diversas oscilaciones.



a) U_{safe}:1500 b) U_{safe}:1000 c) U_{safe}:997.5

Figura 15. Simulación de la varición del umbral de zona segura [25].

7.4.2.5. Resultados de convergencia del algoritmo

La simulación del algoritmo de zonas seguras como se presenta en la Figura 16 da como resultado una trayectoria muy suave, con movimientos muy leves, una convergencia satisfactoria y evadiendo los obstáculos permaneciendo en la zona segura.

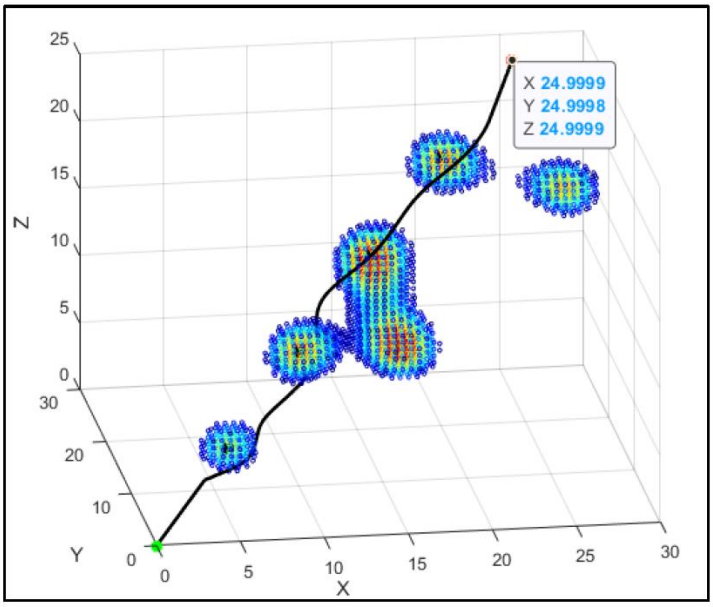


Figura 16. Simulacion de la trayectoria de zonas seguras en 3D [25].

7.4.3. Puntos independientes moviles

Una vez definidas las dimensiones del dron se tiene la posibilidad de evaluar sobre ciertos puntos que se encuentran sobre una circunferencia en un plano perpendicular al enfoque de la cámara del dron, ya que, será solo este el campo de acción de la cámara, de igual forma se tendrá en cuenta el puntos anterior en el cual se haya ubicado el dron y el punto que se encuentre justo frente a el.

Para que este técnica funcione de la mejor manera se tendrá que tener en cuenta tres distintos parámetros: la distancia entre el dron y la circunferencia de puntos, la cantidad de puntos a evaluar, el diámetro de la circunferencia donde irán situados los puntos. Ademas tambien hay que tener en cuenta al momento de implementar que tipo de sensor se va a utilizar ya que puede variar la posición y la cantidad de puntos.

Luego de ya definidos los puntos se evaluaran para determinar si son puntos libres de influencia de algun obstáculo y se tomara la decisión de cual es la mejor opción para la planificación de la trayectoria.

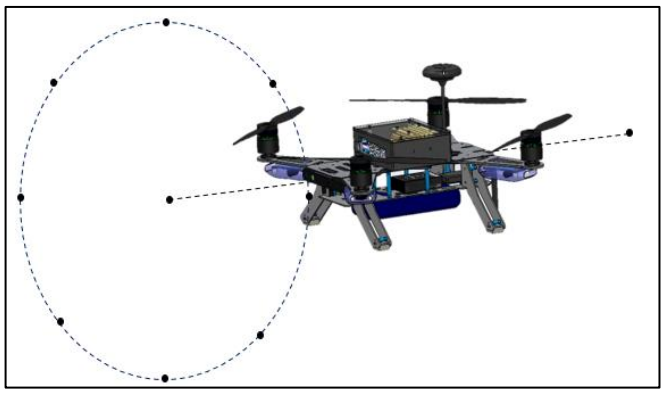


Figura 17. Representación grafica del patrón de puntos en de técnica de puntos independientes móviles [25].

7.4.3.1. Algoritmo basado en puntos independientes móviles

Como se define en [5] este algoritmo permite generar trayectorias libres de obstáculos entre dos puntos en el espacio, este algoritmo en particular es muy sencillo de programar y consta de pocos pasos, aunque su sencillez contrasta con las ventajas que entrega al ser implementado. Está basado en el principio de campos potenciales artificiales y utiliza para su funcionamiento el concepto de zonas seguras.

Inicialmente se definen los parámetros y las ubicaciones de punto inicial, del punto objetivo y de los obstáculos. También definimos una circunferencia de puntos que

posteriormente serán evaluados para determinar si están libres de la influencia de algún obstáculo.

Los puntos de la trayectoria inicialmente se ubican sobre el segmento de recta que une la posición de partida y el punto de destino. A continuación, es necesario encontrar el gradiente del APF generado por los obstáculos asociado a cada punto, esto determinará un vector con la dirección hacia la cual cada punto se puede desplazar en el espacio. Seguidamente se evalúa el patrón de puntos mencionado anteriormente con el fin de saber si en algún punto de la trayectoria se cruza con algún obstáculo. Finalmente, exceptuando el primer y del último punto, cada punto se mueve en la dirección obtenida con el gradiente hasta encontrar una posición en la cual el valor del APF esté libre de la influencia de algún obstáculo.

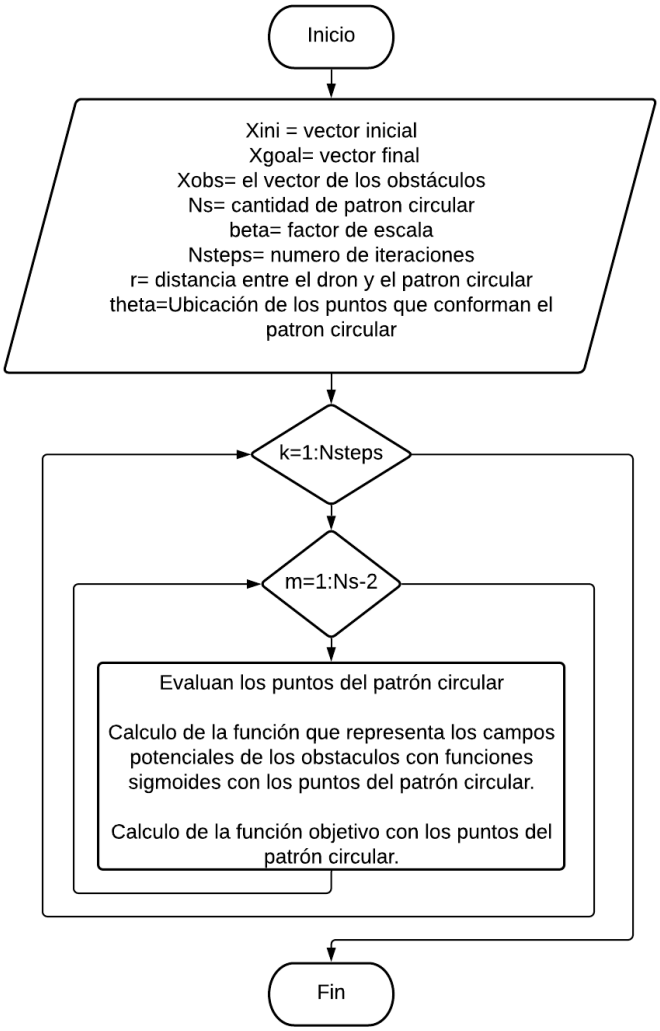


Figura 18. Diagrama de flujo del algoritmo de puntos independientes móviles [25].

Entrada: *punto inicial* $\boldsymbol{X_{ini}}$, *punto objetivo* $\boldsymbol{X_{goal}}$ *y punto obstaculo* $\boldsymbol{X_{obs}}$

Salida: *vector* $\boldsymbol{X}$

1: $\boldsymbol{N_s}$ *cantidad de puntos en el patron circular*

2: $\boldsymbol{N_{steps}}$ *numero maximo de iteraciones*

3: $\boldsymbol{\beta}$ *factor de escala*

4: $\boldsymbol{r}$ *distancia entre el dron y patron de puntos*

5: $\boldsymbol{\theta}$ *ubicación de los puntos (en radianes)*

6: ***Por*** $K=1: N_{steps}$

Evaluamos los puntos del patron circular

7: ***Por*** *m=1:Ns-2*

8: $F_{goal} = \sum(X_{goal} - X_{ini})^2$ *planeación con el gradiente*

9: $F_{grad} = 2 * (X_{goal} - X_{ini})$ *planeación con el gradiente*

10: $X = X_{ini}(k) + \beta * F_{grad}$ *planeación con el gradiente*

*11: xs(:,m)= [X(1,k)+r; X(2,k)+r*cos(θ(m,1)); X(3,k)+r*cos(θ(m,1))]*

12: Jo(m,1)=F_PIM(xs) (Función que evalua el patron de puntos)

13: Jg(m,1)= objetivo(xs) (función que evalua el punto actual con respecto al objetivo)

14: J(m,1)=Jo(m,1)+Jg(m,1)

15: ***Fin Por***

Evaluamos el puntos anterior y posterior

16: xs(:,9)= [x(1,k)+r; x(2,k); x(3,k)]

17: Jo(9,1) =F_PIM(xs(:,9))

18: Jg(9,1) = objetivo(xs(:,9), X_{goal}*)*

19: J(9,1) = Jo(9,1)+Jg(9,1)

20: xs(:,10)= [x(1,k)+r; x(2,k); x(3,k)]

21: Jo(10,1) =F_PIM(xs(:,10))

22: Jg(10,1) = objetivo(xs(:,10), X_{goal}*)*

23: J(10,1) = Jo(10,1)+Jg(10,1)

Se escoge el valor minimo

24: [val,best]=min(J);

Se genera la trayectoria libre de obstáculo

25: x(:,k+1)=xs(:,best);

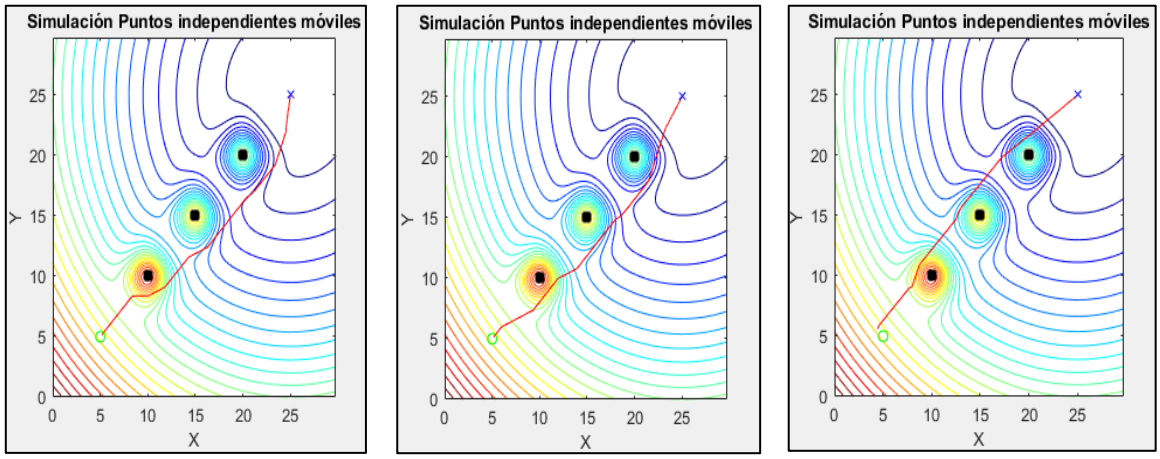
26: ***Fin Por***

7.4.3.2. Resultados de la simulación de puntos independientes moviles en dos dimensiones

En esta sección se presentan las simulaciones en varios escenarios posibles para el algoritmo de puntos independientes móviles sobre campos potenciales. Inicialmente se presenta un análisis de los efectos de la variación de los parámetros del algoritmo, posteriormente se presentan algunos escenarios en los cuales se presenta la convergencia del algoritmo.

Efectos de la variación de parámetros: En este algoritmo los parámetros que influyen en el resultado son los siguientes.

Como podemos observar en la Figura 19 la trayectoria se ve afectada de manera evidente con el cambio del parámetro llamado radio de detección. Cuando el valor es 1 se observa una trayectoria que converge evadiendo los obstáculos a una distancia prudencial de manera de no ser afectada por el obstáculo, sin embargo, cuando aumentamos este valor podemos notar que la trayectoria se acerca más a los obstáculos incluso cambia su dirección desde un inicio y se desvía un poco de punto inicial, pero sin perder la convergencia en su punto objetivo.

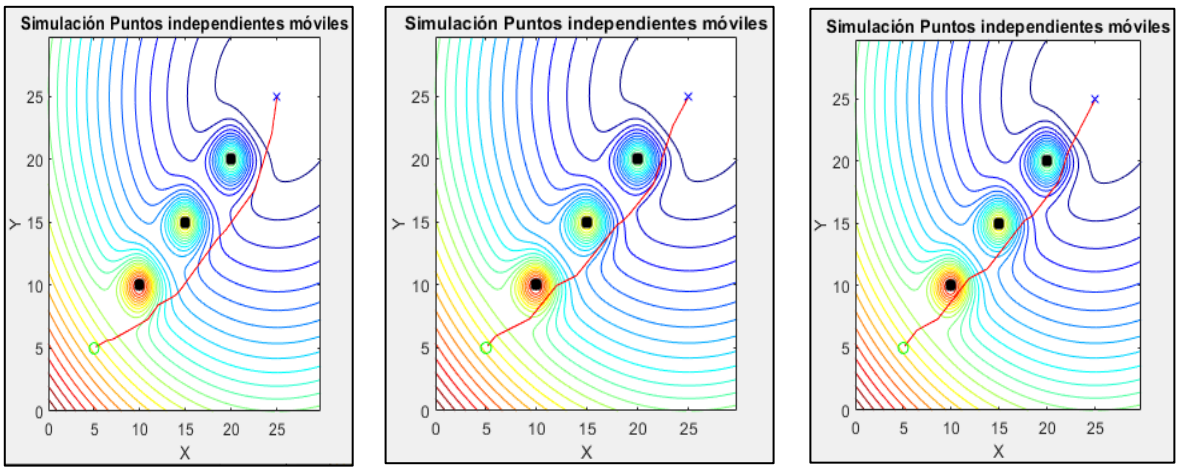


a) r = 1 *b)* r = 4 *c) r = 8*

Figura 19.Simulacion de la variación del radio de detección [25].

Para el siguiente caso de la Figura 20 se vario el valor relativo del peso de la función objetivo J_g, entre mayor sea el valor podemos observar la trayectoria converge de manera satisfactoria, sin embargo, realiza movimientos bruscos y se aproxima mucho a los obstáculos y esto podría provocar una colisión con alguno de ellos.

$$J_g = w * \left(x - X_{goal}\right)' * \left(x - X_{goal}\right) \qquad (33)$$



a) w = 1e-05 *b) w = 5e-05* *c) w = 14e-05*

Figura 20. Simulacion de la variación del valor relativo del peso de la función objetivo [25].

7.4.3.3. Resultados de convergencia del algoritmo

Ya teniendo clara la teoría y el funcionamiento del algoritmo podemos evidenciar los resultados en la Figura 21, se puede observar que la trayectoria trazada llega a su punto objetivo evadiendo los 5 obstáculos propuestos, sin embargo, la trayectoria no es suave, se puede evidenciar los cambios bruscos en algunos puntos.

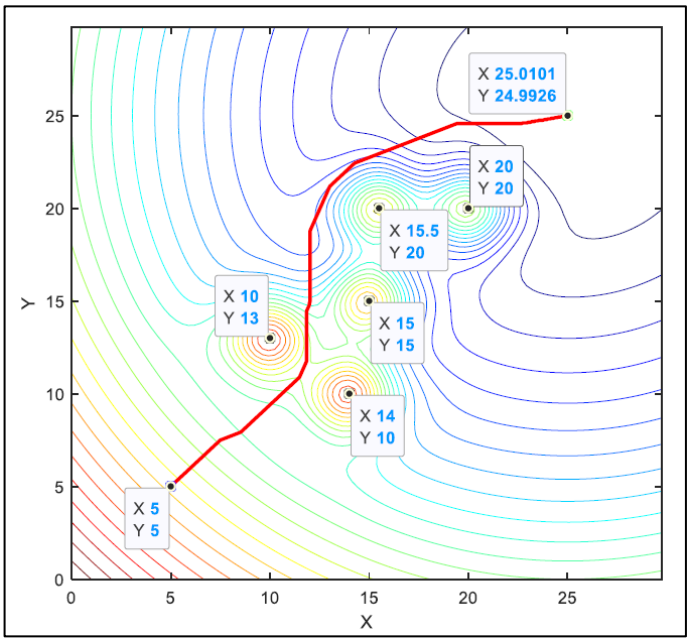


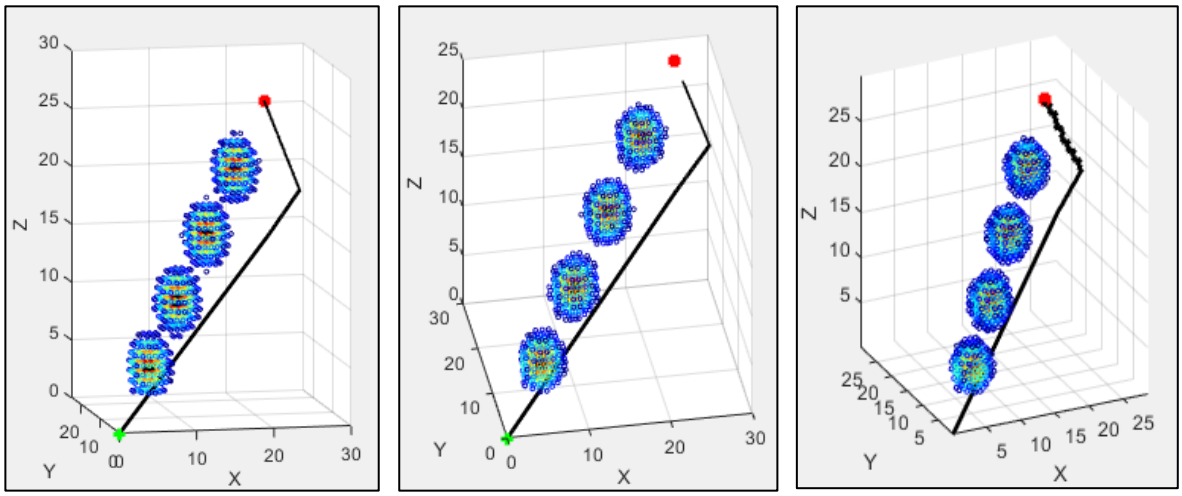
Figura 21. Trayectoria de algoritmo de puntos independientes moviles [25].

7.4.3.4. Resultados de la simulación de puntos independientes móviles en tres dimensiones

Igual que en la sección anterior se presentan las simulaciones en varios escenarios 3D posibles para el algoritmo de puntos independientes móviles sobre campos potenciales. Inicialmente se presenta un análisis de los efectos de la variación de los parámetros del algoritmo, posteriormente se presentan algunos escenarios en los cuales se presenta la convergencia del algoritmo.

Efectos de la variación de parámetros: En este algoritmo los parámetros que influye en el resultado son los siguientes.

Como se puede observar en la Figura 22, el tamaño del paso en un parámetro que evidentemente afecta la convergencia y la forma de la trayectoria, en este caso sino se escoge el valor correcto, la trayectoria puede que no converja en su punto meta o que presente oscilaciones bruscas cerca al punto objetivo.



a) α = 0.08 b) α = 0.05 c) α = 0.03

Figura 22.Simulacion de la variación del tamaño del paso [25].

En la Figura 23, se puede observar la variación de otro parámetro, el radio de detección, se observa que con las tres variaciones la trayectoria converge, sin embargo, podemos notar que si los valores son pequeños la trayectoria se acerca considerablemente al primero obstáculo. Pero si se observa la figura en la que valor del radio de detección es 8 se puede apreciar que evita el obstáculo satisfactoriamente en contraste con que la trayectoria abarque mas espacio antes de llegar a la meta.

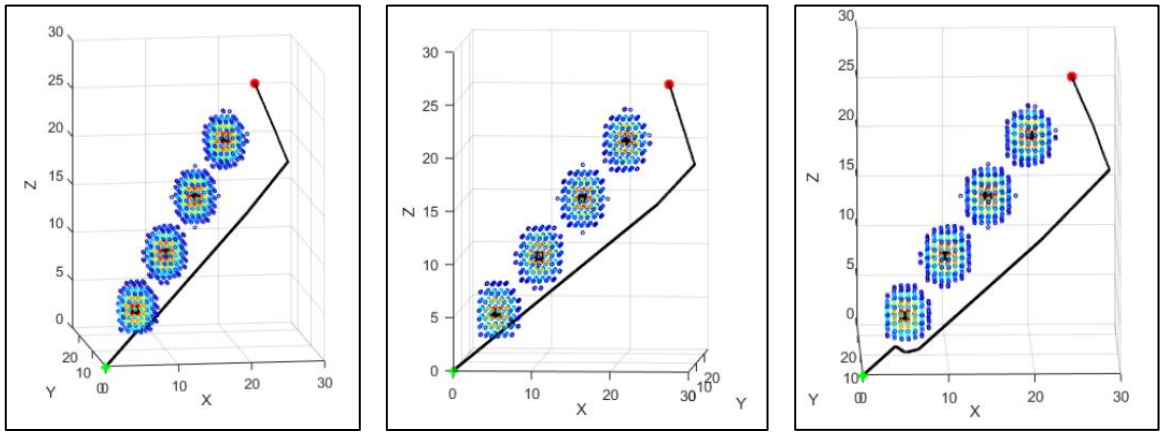


Figura 23.Simulación de la variación del radio de detección [25].

7.4.3.5. Resultados de convergencia del algoritmo

Ya con los parámetros correctos, se observa en la Figura 24 que el algoritmo de puntos independientes móviles presenta una trayectoria la cual converge exitosamente evadiendo los 5 obstáculos presentados que se organizaron de manera que obstruyera el mayor paso posible entre el punto inicial y el punto meta, sin embargo, algo que no se puede notar en la figura, pero se puede resaltar en las oscilaciones que presenta la hora de rodear los obstáculos.

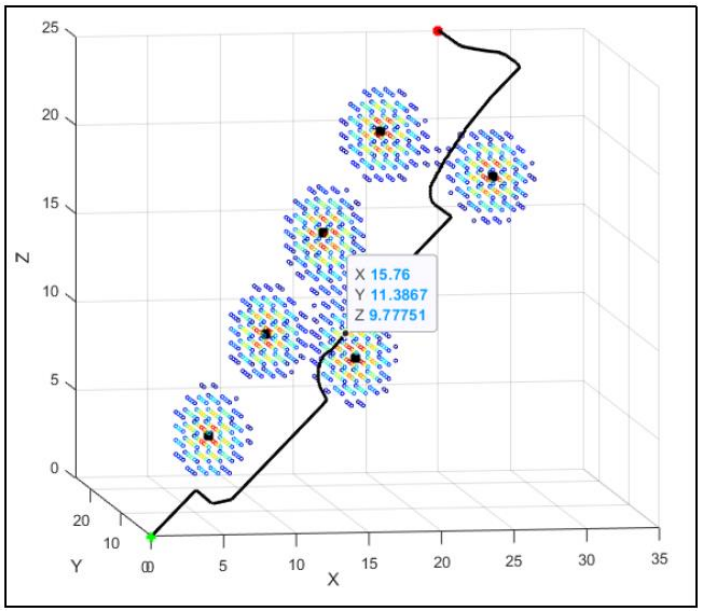


Figura 24. Trayectoria del algoritmo de puntos independientes moviles en 3D [25].

En la Tabla 5 se presentan algunos criterios evaluados que permiten realizar un contraste entre los algoritmos. Los elementos contenidos son los siguientes:

Criterios	***Puntos independientes móviles***	***Permanencia en zonas seguras***
Convergencia de la trayectoria	Sí	Sí
Regiones con oscilaciones inestables	4	0
Máximo ángulo de cambio de dirección	90°	45°
Cambios de dirección con ángulos mayores a 30°	7	4

Tabla 5. Evaluación de los algoritmos de planeación de trayectoria.

- Resultado de convergencia de las trayectorias en los dos algoritmos, es decir, si el algoritmo puede encontrar una ruta de forma satisfactoria o si cae en un mínimo local.

- Número de oscilaciones inestables que se generan.

- El máximo ángulo de cambio de dirección se utiliza para evaluar la suavidad de la trayectoria generada.

- El número de cambios de dirección con ángulos mayores a 30°, el criterio tiene en cuenta cómo cambian las direcciones de los puntos con respecto a las posiciones de los puntos adyacentes.

Con base en resultados anteriormente mencionados el algoritmo de **zonas seguras** es el que presenta grandes aportes al objetivo principal de este proyecto al tener como base el descenso del gradiente y el método de búsqueda: zonas seguras. Entre ellos:

- Se tiene un control sobre la distancia que hay entre la trayectoria y los obstáculos.
- No se presentan oscilaciones inestables, ya que los puntos de la trayectoria tiendan a quedar sobre la frontera de la zona segura, la cual es continua en el espacio de trabajo.
- La distancia de la trayectoria es menor en comparación con el algoritmo de puntos independientes móviles.

8. COMPONENTES Y ENSAMBLADO DEL SISTEMA

En esta sección se hará una descripción más técnica y profunda del sistema implementado. El sistema se define como la composición de la Ground Control Station (GCS[2]) y el Dron.

El Quadrotor utilizado en este trabajo es “aero ready to fly” drone by Intel, el cual está compuesto por una estructura de soporte en fibra de carbono sobre la que se montan los motores, la placa madre y la placa de navegación. Para controlar sus acciones es necesario conectarlo a un dispositivo que contenga una aplicación de control a través de una conexión Wi-Fi por lo que su alcance de movimiento está limitado al alcance de la conexión (entre 30 y 100 metros dependiendo de las condiciones climáticas). Su estructura es modular permitiendo un recambio muy simple de todas y cada una de sus piezas.

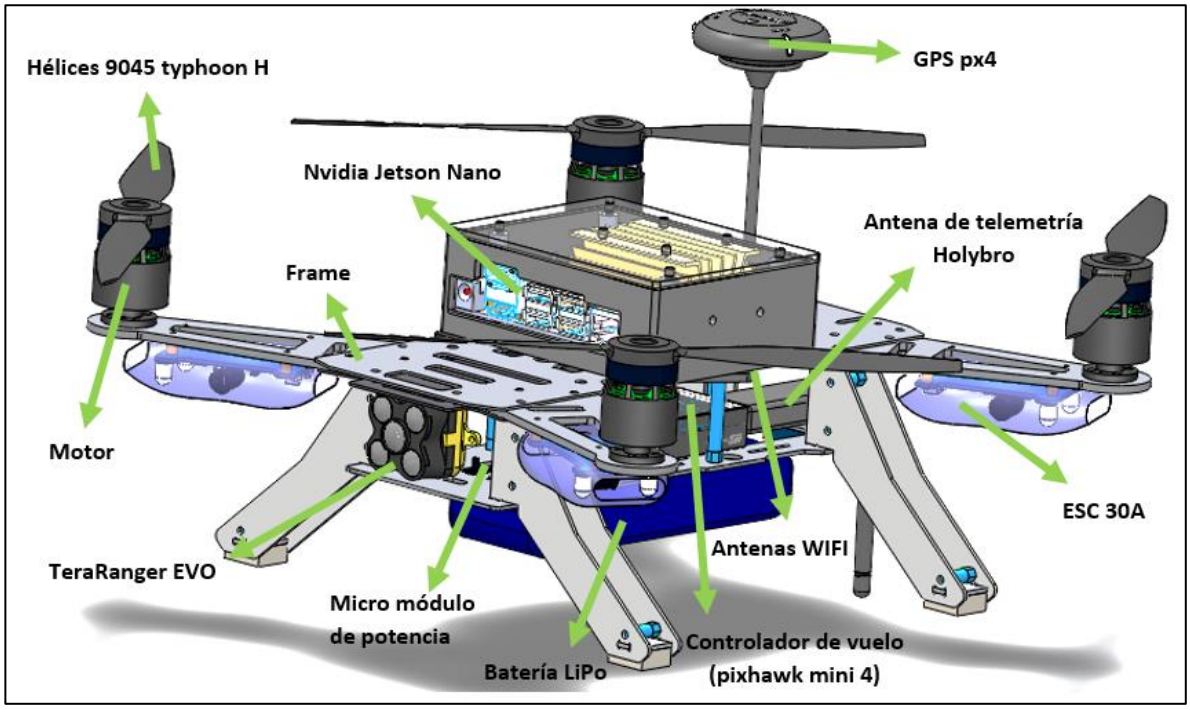


Figura 25. CAD del quadrotor con sus diversos componentes [25].

[2] GCS: estación en tierra.

8.1. Descripción funcional

El sistema que se encuentra representado a nivel funcional en la Figura 26, inicialmente permite establecer la comunicación entre el usuario y el dron por medio de la GCS, la cual recibe y transmite los parámetros de la nueva misión al dron, reporta al usuario el estado del dron y además recibe datos emitidos por los sensores del dron para ser analizados y registrados por el usuario.

El dron está conformado por dos partes principales, el quadrotor y la unidad de procesamiento el cual está compuesta por el sensor de profundidad. A su vez, el quadrotor está compuesto por un frame, motores, hélices ESC[3], distribuidores de energía, batería, antena de telemetría, antena GPS y una placa controladora de vuelvo.

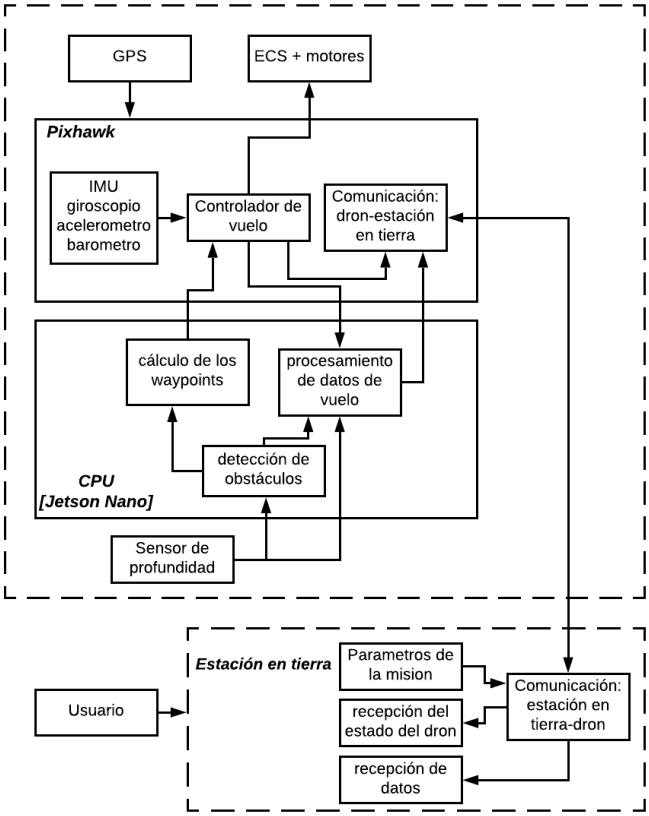


Figura 26. Diagrama funcional del sistema [25].

[3] Electronic Speed Controller: es un circuito electrónico cuya función es variar la velocidad de un motor eléctrico, generando energía en tres fases de baja tensión.

A continuación, se ofrece un listado y una breve descripción de los componentes restantes necesarios para construir el sistema y se indican los componentes que se seleccionaron para este proyecto en particular y la justificación de estas elecciones.

8.2. Selección de los componentes

Para poder implementar el sistema y dotarlo de las funciones mencionadas anteriormente, es necesario establecer un listado de los componentes físicos que van a conformar dicho sistema y un proceso de selección, comparando distintas soluciones y equipos de varios fabricantes de manera de poder tomar la decisión más acertada a la hora de efectuar la compra de los componentes. Errores en esta etapa se traducen en tiempos de espera de importaciones, perdida de dinero y hasta reformulación de la solución planteada.

8.2.1. Motores y ESCs

El motor es uno de los componentes más importantes a la hora de construir un dron. Las cuatro características básicas de un motor son: el voltaje de funcionamiento, los rpm por Volt (V), el thrust o empuje y su corriente nominal. Para este tipo de aplicaciones lo usual es utilizar motores sin escobillas con rotor exterior (brushless outrunner). Este es un motor de corriente continua controlado por un ESC[4], elemento que se introduce entre el controlador de vuelo y el motor, y que cumple la función de recibir una señal PWM del controlador para luego entregar una tensión trifásica escalonada al motor. El conjunto motor brushless más ESC sustituye el conmutador y las escobillas de los motores DC por switches de estado sólido que funcionan con una lógica para la conmutación de los bobinados, lo que implica una ventaja ya que no requieren mantenimiento por el desgaste del contacto móvil [23]. Se decidió que los ESCs fueran de 30 A debido a la disponibilidad de productos en el mercado.

Especificaciones:

- UBEC: 5.5V / 3A
- Batería de LiPo: 2 - 4s (celdas)
- Tamaño: 54 mm x 26 mm x 11 mm
- Peso: 32 g

Figura 27. ESC de 30 A.

[4] Electronic Speed Controller: es un circuito electrónico cuya función es variar la velocidad de un motor eléctrico, generando energía en tres fases de baja tensión.

8.2.2. Bateria

Normalmente los drones utilizan baterías de LiPo (Polímero de Litio) por ser estas las que mejor se adaptan a las necesidades del dron. El Litio (Li) es el metal más ligero conocido debido a que cuenta con solo 3 protones, por tanto, su peso atómico es muy bajo, permitiéndole tener un gran potencial químico, lo cual da gran capacidad de almacenaje con poco peso.

Se opto por una batería de cuatro celdas de LiPo (14.8 V) debido a la amplia disponibilidad comercial y que el voltaje máximo solicitado por los componentes del dron es de 12 V.

- Capacidad mínima: 4000 mAh
- Configuración: 4S1P / 14.8v / 4 Celdas
- Descarga constante: 30C
- Descarga máxima (10 seg): 40 ° C
- Peso del paquete: 452g
- Tamaño del paquete: 148x50x29 mm

Figura 28. Bateria LiPo.

8.2.3. Controlador de vuelo

El controlador de vuelo es un componente crucial para el dron. Cuenta con los sensores necesarios para controlar la estabilidad y trayectoria del sistema y con el programa que permite su funcionamiento. A su vez es quien manda la señal a los ESCs para actuar sobre la velocidad de giro de los motores en función de todas las variables que está sensando. Generalmente todos los controladores de vuelo cuentan con los mismos componentes: CPU, memoria, giroscopio, acelerómetro, magnetómetro, barómetro y entradas y salidas digitales para recibir datos de sensores externos y enviar comandos e informaci´on al exterior [23].

Finalmente se decidió por el controlador “Pixhawk PX4” debido a que es, dentro de las opciones comerciales, el que tiene un mejor procesador y más memoria, lo cual es más que suficiente para las misiones que se desean realizar con el dron.

Especificaciones:

- Procesador principal de FMU: STM32F765, 32 bit Arm® Cortex®-M7
- Memoria 512KB RAM
- Velocidad 216MHz
- Sensores a bordo Accel / Gyro: ICM-20689
- Magnetómetro IST8310
- Barómetro: MS5611

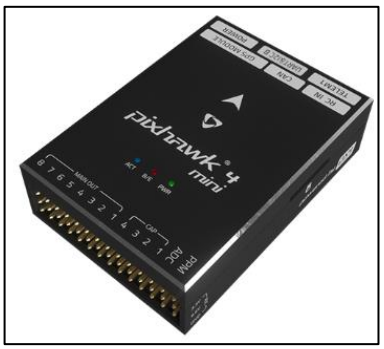


Figura 29. Controlador de vuelo Pixhawk 4 mini.

8.2.4. Antena GPS

Holybro habilitó este nuevo GPS para trabajar con el Pixhawk 4. Tiene un módulo UBLOX M8N en él, así como una brújula IST8310 e indicador LED de tres colores. Además, el interruptor de seguridad en él hará que la conexión sea más cómoda y sencilla.

Especificaciones:

- Regulador de bajo ruido de 3,3 V.
- LED indicadores.
- Funda protectora.
- Pixhawk4 - Cable de 10 pines (10.2 in)
- NEO-M8 con módulo de brújula

Figura 30. Antena GPS de HolyBro

8.2.5. Modulo de potencia

El modulo de potencia es un componente electrónico que detecta la caída de tensión causada cuando la batería tiene poca carga. Este sistema disminuye la alimentación del motor con el fin de proporcionar la potencia suficiente al dron para que sea capaz de volver de manera segura al operador. La potencia a la hélice se recorta pero la operación de los controles se mantiene.

Especificaciones:

- Corriente PCB: 120 A continua
- Alimentación: DC 7V~42V(2S~10S)
- Salida: DC 5.1V~5.3V
- Dimensiones: 35x35x5mm

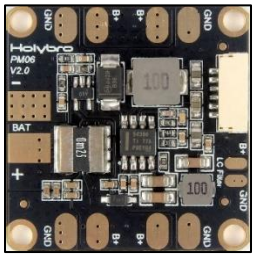


Figura 31. Módulo de potencia (PM06 v2).

8.2.6. Antenas de telemetria

El kit de telemetria utilizado es de la marca HolyBro, este kit conforma una plataforma de radio de código abierto pequeña, liviana y económica que generalmente permite rangos de más de 300 m.

- Frecuencia: 433 MHz
- Alimentación: 3.7-6 VDC
- Tamaño: 1.024 x 2.087 x 0.413 in
- Peso: 0.95 oz.
- Voltaje de alimentación: 5 VDC.
- Corriente de transmisión: 100 mA a 20 dBm.
- Corriente de recepción: 25 mA
- Interfaz de serie: 3.3 V UART

Figura 32. Antenas de telemetria kit V3.

8.2.7. Unidad de procesamiento

Jetson Nano, ésta fue seleccionada gracias a su compacto tamaño a diferencia de otras tarjetas diseñadas por Nvidia, teniendo también como sistema operativo Ubuntu 18.04 de 64 bits con drivers propios de Nvidia que la hacen compatible a diferentes cámaras y sensores lo que hace que haya una base para trabajar en ésta, también contando con soporte por parte de los desarrolladores de NVidia, y parte de desarrollo previo respecto a vehículos no tripulados tales como jetbots utilizados en el campo de la visión por computadora.

Especificaciones:

- GPU: Maxwell de 128 núcleos
- UPC: ARM A57 de 4 núcleos a 1,43 GHz
- Memoria: LPDDR4 de 4 GB y 64 bits a 25,6 GB/s
- Cámara: 2 carriles MIPI CSI-2 DPHY
- Conectividad: Gigabit Ethernet, clave M.2 E

Figura 33. NVidia Jetson Nano.

8.2.8. Sensor de distancia Teraranger EVO

Proporciona lecturas de distancia calibradas en milímetros y tiene un alcance de hasta 60 m. TeraRanger Evo 60m utiliza tecnología LED. Una de las ventajas de esto es que permite que el sensor de tiempo de vuelo tenga un "campo de visión" de modo que, en lugar de medir la distancia en función de un punto de luz muy pequeño, el sensor mide sobre un área. A 1 m de distancia, el área es de aproximadamente 3 cm por 3 cm. A 10 m, mide aproximadamente 30 cm por 30 cm, aumentando linealmente con el rango. Para muchas aplicaciones, esta es una ventaja significativa y proporciona un flujo de datos más apropiado y estable.

- Precisión: ±4cm en14m, 1.5%
- Interfaz: UART, USB 2.0 Micro-B, I2C
- Rango: 0.5 – 60 m

Figura 34. TeraRanger EVO.

8.3. Ensamble y distribución

Como se mencionó anteriormente gracias a su estructura modular su pudo hacer el cambio de algunos componentes y se redistribuyeron los componentes en el dron de manera que su peso fuera equilibrado, con respecto al centro de gravedad, ya que este es un factor de importancia a la hora de realizar el vuelo.

Inicialmente la distribución de los motores depende del marco, ya que el de este proyecto es en “X”, la orientación de los motores es la siguiente:

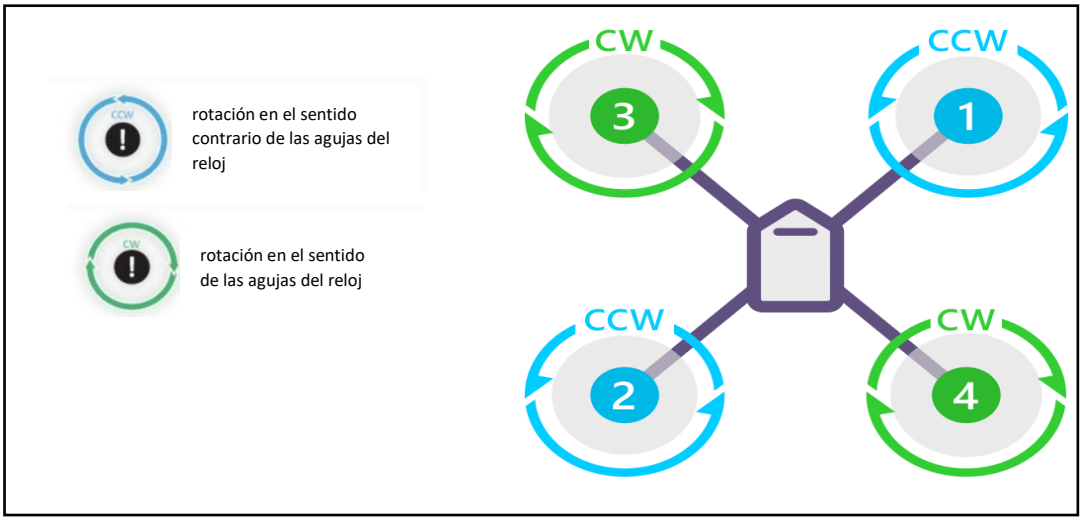


Figura 35. Distribgución de los motores y orientación.

Luego desde la batería se realiza una conexión en paralelo para alimentar el módulo de potencia donde se conecta la alimentación de los ESC para luego llegar a los motores y la alimentación de la PX4 mini como se observa en la Figura 36.

La PX4 mini nos indica cual es el frente de la misma, ya que este debe está orientada correctamente para el funcionamiento del giroscopio, además, esta debe encontrarse ubicada en el centro del dron. Ya ubicada la px4 se procede a conectar los diferentes componentes, esta conexión se aprecia en la Figura 37.

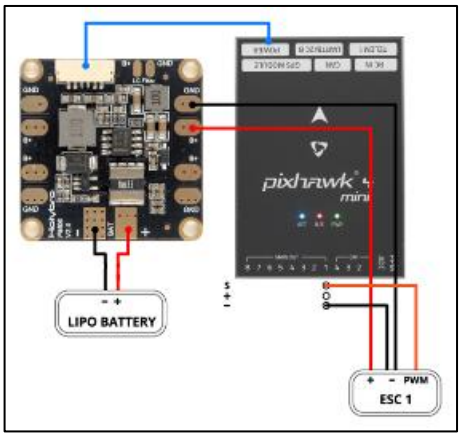
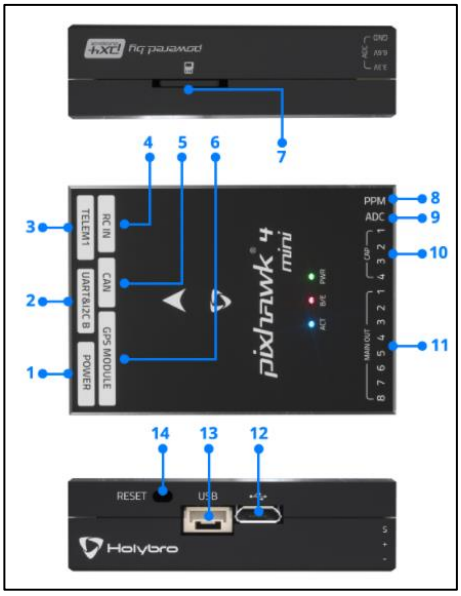


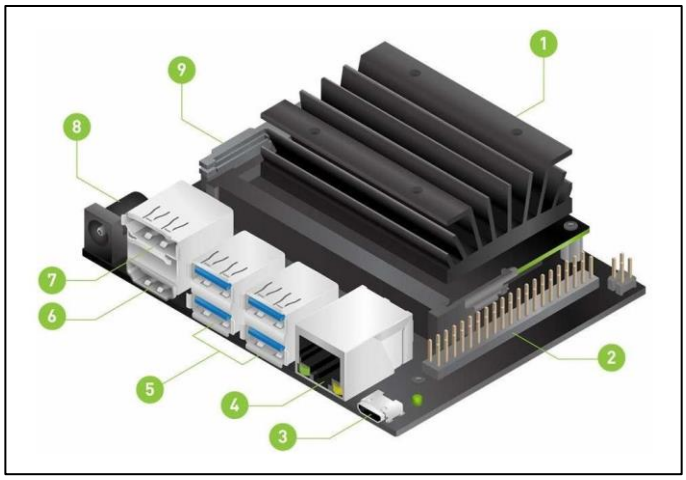
Figura 36. Conexión de la bateria con el modulo de potencia y la PX4 [25].



1. Alimentación
2. UART Y I2C
3. Antena de telemetría
4. Entrada del receptor de radio control (DSM/SBUS)
5. CAN
6. Modulo GPS
7. SD card
8. Entrada del receptor de radio control (PPM)
9. Entrada ADC
10. Entrada PWM
11. Salidas de PWM
12. Puerto Micro-USB
13. USB

Figura 37. Entradas y salidas de la PX4.

Finalmente por medio de otro modulo de potencia alimentamos el procesador que para este caso es la Jetson nano, y a esta conectamos la PX4 y al sensor de profundidad Teraranger EVO.



1. Disipador de calor
2. Cabezal de expansión de 40 pines
3. Micro B USB
4. Puerto Gigabit Ethernet
5. Puerto USB 3.0
6. Puerto de salida HDMI
7. Conector DisplayPort
8. Conector Barril DC 5V
9. Conectores de cámara MIPI CSI

Figura 38.Nvidia Jetson Nano.

8.4. Sistema armado

En la Figura 39 se puede apreciar el sistema completamente ensamblado y su respectivo radio control.

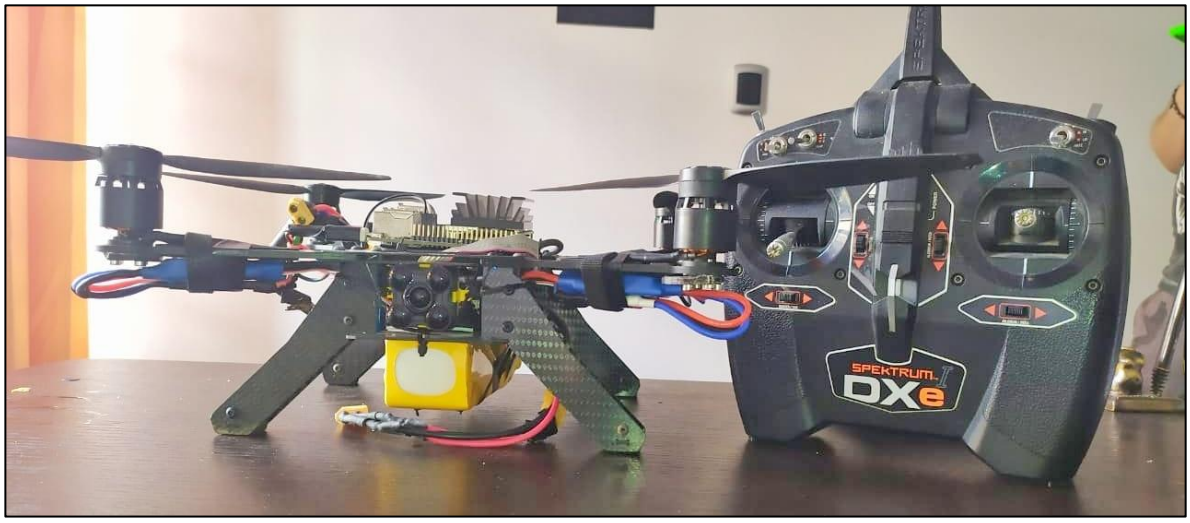


Figura 39. Sistema armado (GCS y Dron) [25].

8.5. Protocolos de comunicación

8.5.1. Protocolo de comunicación SSH (Secure SHell)

Es un protocolo que facilita las comunicaciones seguras entre dos sistemas usando una arquitectura cliente/servidor y que permite a los usuarios conectarse a un host remotamente. En este proyecto se utilizó para comunicar la tarjeta madre y poder acceder a ella desde un computador remoto y poder monitorear mejor los procesos (envió y recepción de datos) internos de la misma.

8.5.2. Protocolo de comunicación TCP/UDP

El protocolo UDP y TCP es el nivel de transporte basado en el intercambio de datagramas. Permite el envío de datagramas a través de la red sin que se haya establecido previamente una conexión, ya que el propio datagrama incorpora suficiente información de direccionamiento en su cabecera. Tampoco tiene confirmación ni control de flujo, por lo que los paquetes pueden adelantarse unos a otros; y tampoco se sabe si ha llegado correctamente, ya que no hay confirmación de entrega o recepción. Este protocolo se utilizó para enviar diferentes órdenes a la tarjeta madre en cuanto a comunicación y estado del algoritmo a correr, ya que estos protocolos no aseguran la llegada de los paquetes de información, se utilizó un tracker para suplir esto.

- SSH (puerto 22)
- Jetson nano y px4 mini (conexión UART por canal de telemetría 1)

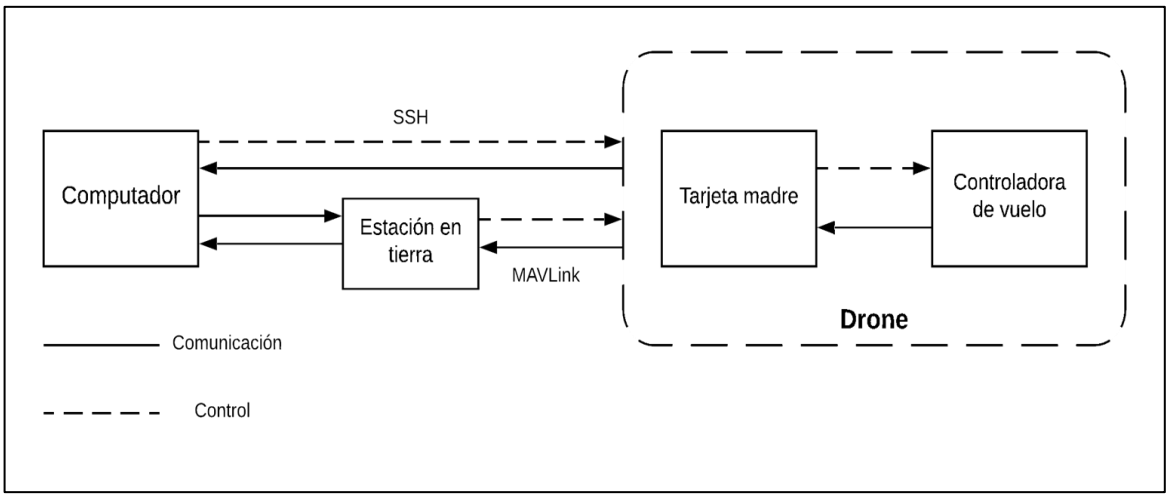


Figura 40.Diagrama de comunicación entre los diferentes procesadores [25].

8.6. Características de los sensores

Para obtener una actualización de la información acerca de la posición, altitud, orientación e información acerca del ambiente que rodea al Quadrotor, se cuenta con 4 diferentes sensores, los cuales dos de estos se encuentran ubicados internamente de la px4 mini, una antena GPS externa y una cámara para lograr obtener datos de los objetos que se encuentren entre el punto inicial y el final.

La PX4 mini tiene:

- un **barómetro MS5611** que cuenta con un módulo de alta resolución: 10 cm, realiza una conversión rápida de hasta 1 ms.

- Un **sensor de presión digital** integrado (ADC ΔΣ de 24 bits) y su rango de operación: 10 a 1200 mbar, -40 °C a +85 °C que ofrece una excelente estabilidad a largo plazo.

- Un **acelerómetro / giroscopio ICM-20689** es un dispositivo MotionTracking de 6 ejes que combina un giroscopio de 3 ejes, un acelerómetro de 3 ejes.

- Una **antena GPS** que tiene el módulo UBLOX M8N en él.

- Una **brújula IST8310** y el **indicador** LED tricolor.

- Un **interruptor de seguridad** hará que la conexión sea más práctica y sencilla. Este módulo se envía con una tasa de baudios de 38400 5Hz.

9. IMPLEMENTACIÓN Y VALIDACIÓN

En esta etapa del proyecto de procede a realizar la implementación del codigo en el prototipo fisico siguiendo una serie de pasos para tener una correcta comunicación entre la estación en tierra y el UAV.

9.1. Calibración de los sensores

Para poder tener una correcta recolección de datos que nos suministran los sensores es necesario realizar una calibración de los mismos, para esto es necesaria la conexión a la plataforma QGroundControl, de esta manera, se puede conocer su estado de energía y sus parámetros de vuelo, la configuración de los modos de vuelo, adicionalmente la calibración del radio control.

El primer paso es encender el Dron y el radio control, una vez se reconozca la señal del WiFi del Dron, se enlazará a este como una red normal llamada Aero-XXXX, en este instante el QGroundControl recibe la señal del Drone y sus parámetros, iniciando en el mapa donde se encuentra el Drone gracias al GPS.

En la interfaz del QGroundControl se tiene presente el nivel de la batería del UAV, la intensidad de la señal, su conexión a la computadora, la información de giroscopio, acelerómetro, modos de vuelo, waypoint y punto de home.

Figura 41.Imagen de inicio del QGC.

Luego al entrar en la ventana de información podemos encontrar un resumen del estado de los sensores, modos de vuelo, canales, airframe, la información de la batería y la información de seguridad.

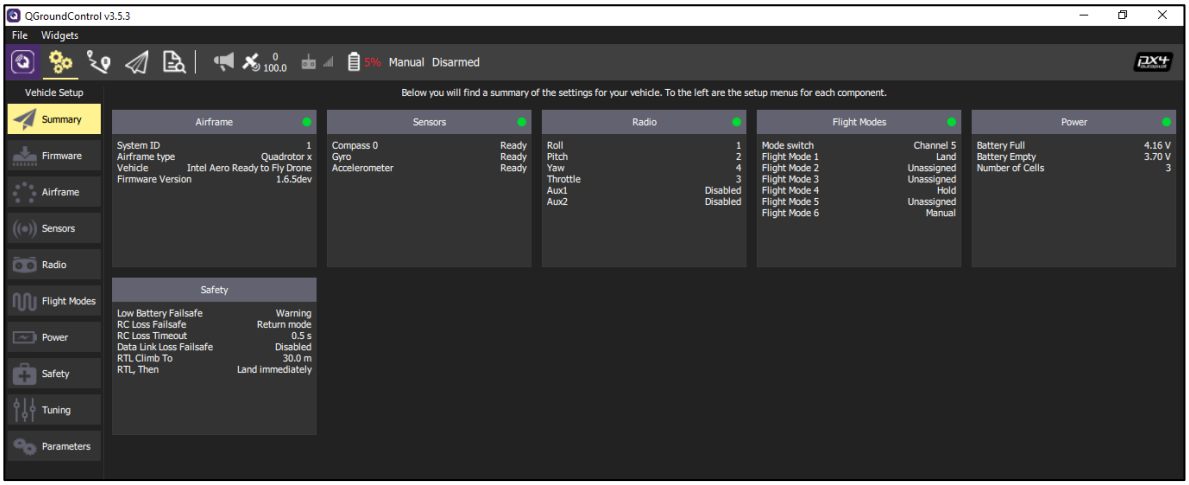


Figura 42.Ventana de información general de QGroundControl.

Verificando la información del vehículo en las demás ventanas, encontramos el airframe o el cuerpo del Dron enlazándolo con su versión y tipo de cuerpo correspondiente que en este caso es el “Intel Aero Ready ro Fly Drone”.

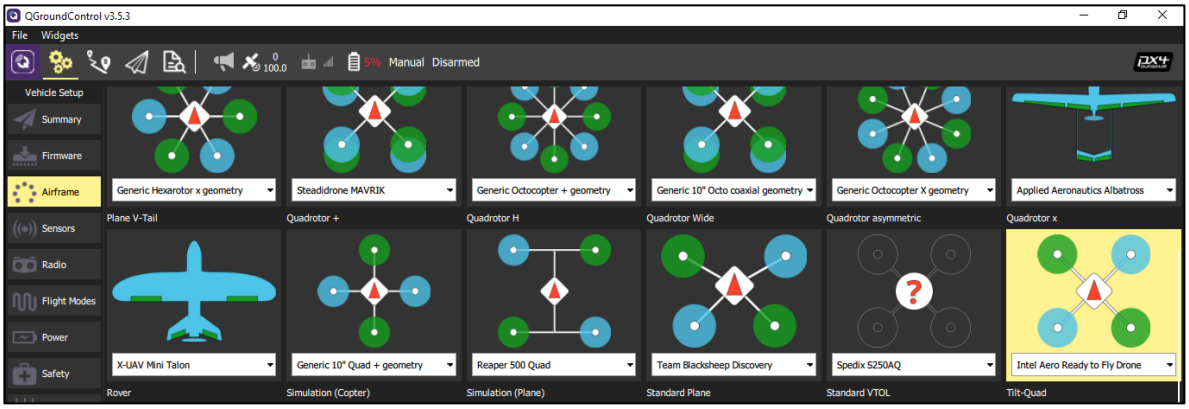


Figura 43. Selección de la versión y tipo de cuerpo.

Los sensores deben ser calibrados antes de volar o después de algún tiempo de no uso, para realizar la calibración se ingresa a la ventana de sensores ubicada en la parte izquierda y encontraremos una lista de los sensores, al seleccionar cada uno encontraremos una lista de pasos a seguir, estos pasos consisten en una serie de

movimientos que el usuario tiene que realizar repetitivamente hasta que se complete al 100 % la calibrada.

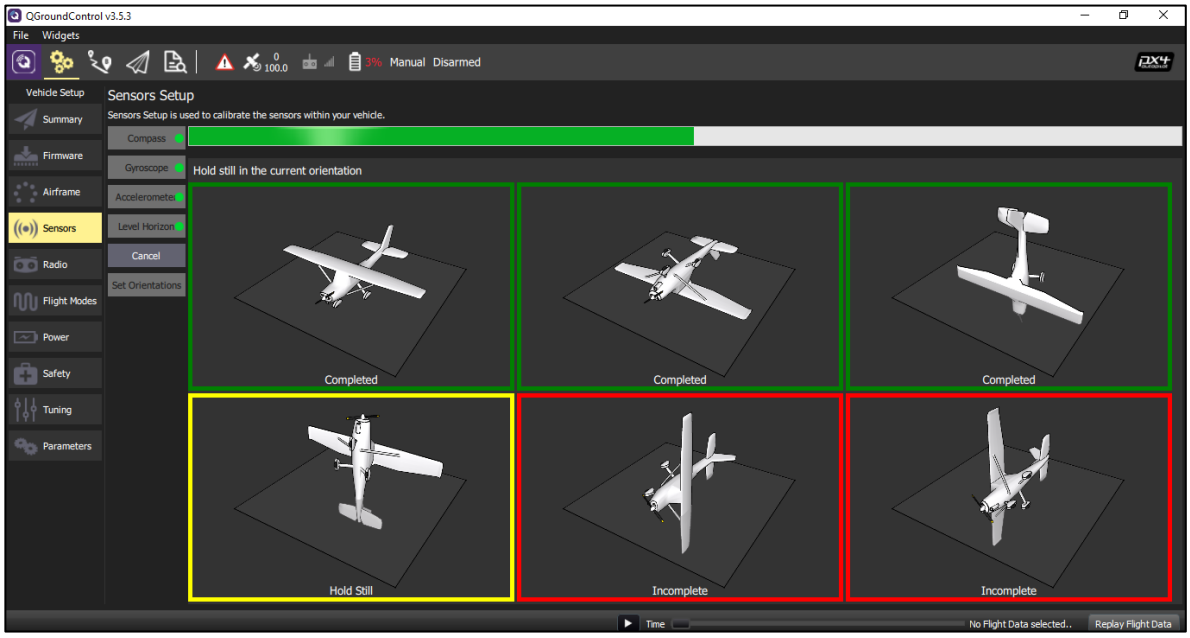


Figura 44. Calibración de los sensores.

Con esto completaríamos la calibración de los sensores así que lo siguiente es calibrar el radio control, para el cual solo se necesita mover las palancas en las posiciones indicadas.

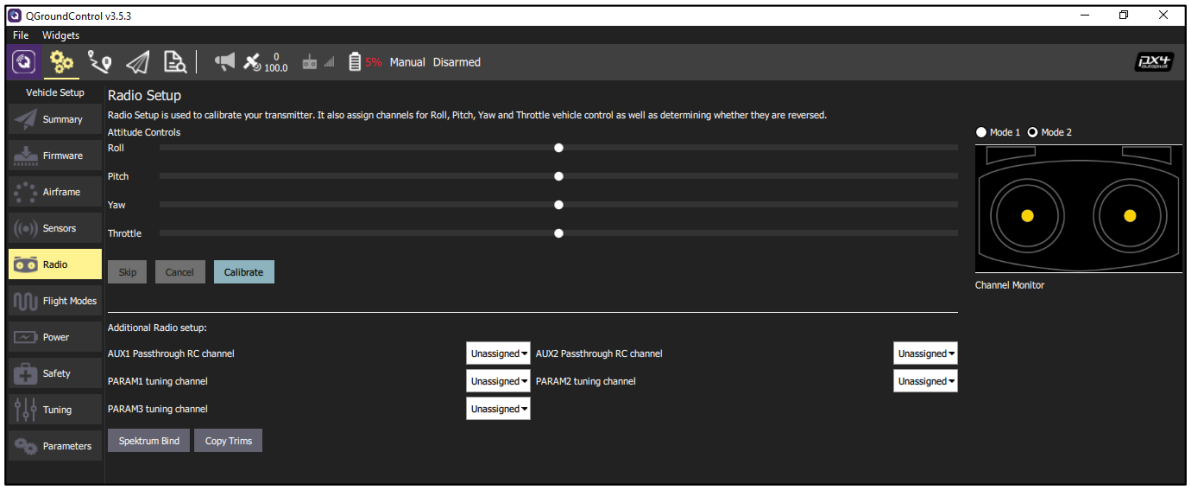


Figura 45. Calibración del radio control.

Luego de haber calibrado el radio, se deben ajustar los modos de vuelo para poder ajustarlos al radio control y asignarle uno a cada una de las palancas asignadas para esta función.

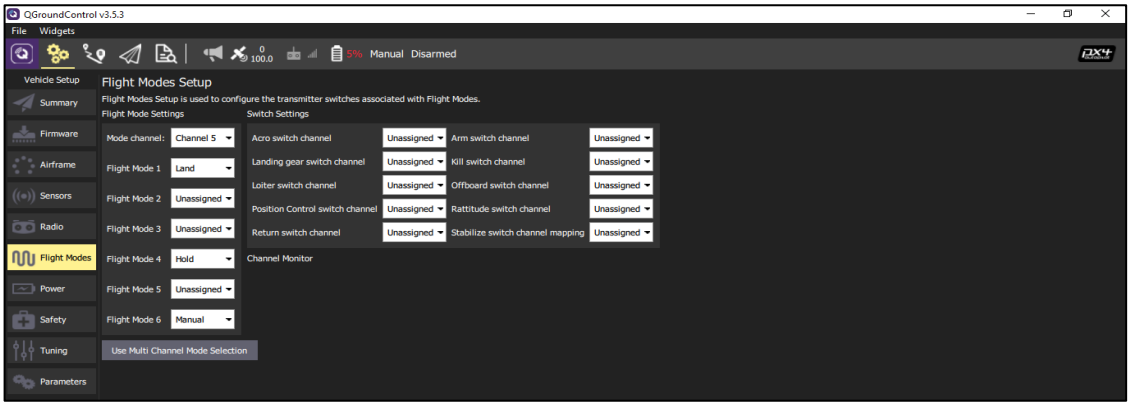


Figura 46.Ajuste de los modos de vuelo.

Lo siguiente a ajustar sería el nivel de la batería el cual depende del número de celdas que posea, su voltaje máximo y voltaje mínimo, en nuestro caso utilizamos una batería de 3 celdas, 3.7 [v] por celda, con un máximo de 12.5 [v] y un mínimo de 11.1 [v], QGC se encarga de hacer el divisor de voltaje para ajustar el nivel de batería y no dar niveles erróneos de batería lo cual sería perjudicial a la hora de volar.

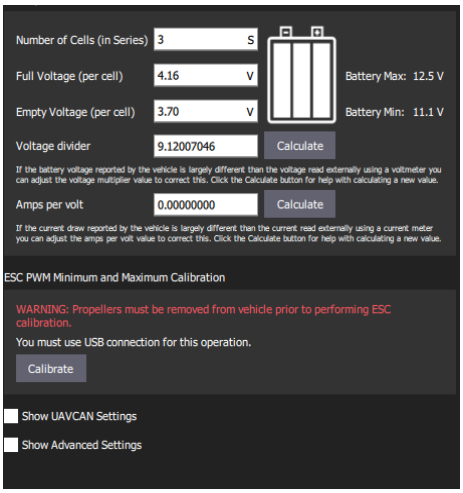


Figura 47.Ajuste de la batería.

Por último es necesario revisar los parámetros del filtro extendido de Kalman (EKF) propio del UAV ya que este se encarga también de la estimación de las posiciones y quitar el ruido al momento de ser dirigido por WayPoints como solemos hacerlo en las pruebas tomadas, el EKF lo dejamos en sus parámetros por defecto.

9.2. Dificultades presentadas en el desarrollo del proyecto

Como propuesta inicial se tenía planteado utilizar una Cámara Intel® RealSense™ R200, en la Figura 41 se observan dos fotogramas captados por la cámara en prueba de su funcionamiento individual, el primer fotograma representa imagen de profundidad y en el segundo fotograma de la imagen a color, estas son las posibles imágenes que nos aporta la intel realsense, adicionalmente una nube de puntos que es la utilizada para calcular la distancia entre el dron y el obstáculo, sin embargo, no se continuo el trabajo con esta cámara porque presentó un inconveniente, ya que este contenía un imán el cual nos interfería con nuestro GPS y no permitía que este tomara bien el dato de los satélites, por esto se tenían datos incorrectos de ubicación.

a) Imagen de prufundidad

b) Imagen normal

Figura 48.Imagenes tomadas por la Intel Realsense [25].

Como **alternativa de solución para el presente proyecto** se utilizó un sensor “TeraRanger Evo 60m”, un sensor de medición de distancia de largo alcance que reconoce solo un punto de distancia.

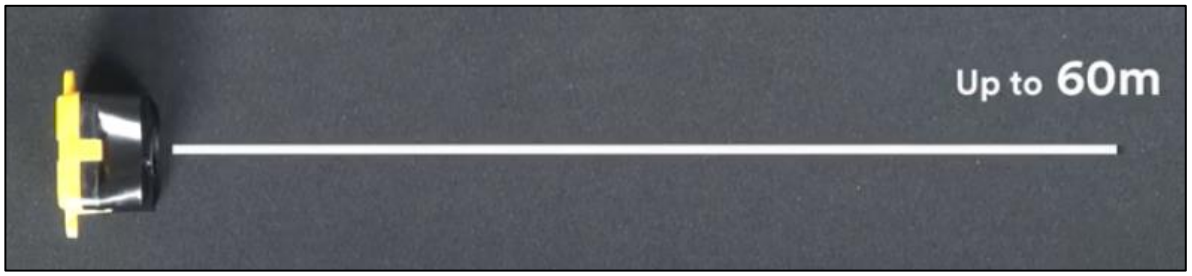


Figura 49. TeraRanger Evo 60m: el sensor de distancia.

Con el uso de este sensor obtuvimos un dato de profundidad en bits el cual se mapea entre una distancia de 0.5 a 60 metros. Con este sensor se logra detectar la presencia de un obstáculo al frente del dron, sin embargo, no logra dimensionar el tamaño del obstáculo, por esto para lograr el objetivo que el evadir el obstáculo se realizó una modificación en el código, en el cual cuando el sensor detecte la presencia de un obstáculo el dron realice un moviendo en dirección lateral una k distancia para que el objeto quede fue del alcance tanto del sensor como del dron y pueda seguir su recorrido al punto objetivo.

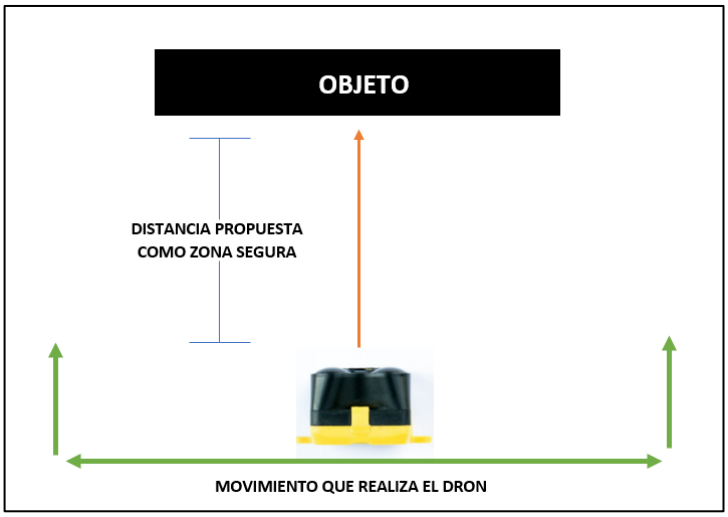


Figura 50. Función del sensor TeraRanger en el proyecto [25].

Una **alternativa a futuro** para poder implementar el algoritmo de manera adecuada sería una cámara de tipo RGBd (RGBdepth) que permite capturar la profundidad de los obstáculos y la forma de este. Un ejemplo de cámara RGBd sería la cámara 3d-tof (tiempo de vuelo) producida por terabee la cual cuenta con un campo de visión de 80x60 píxeles y un ángulo de 74° x 57°.

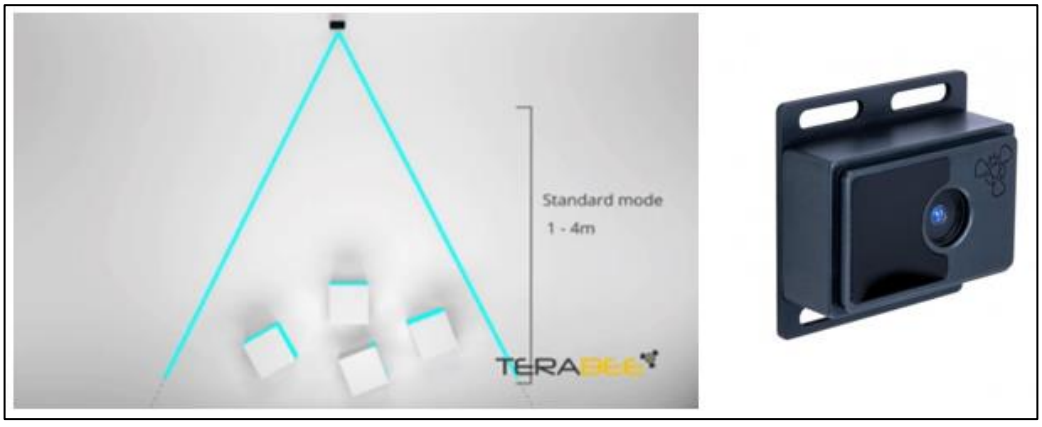


Figura 51. Cámara 3d-tof Terabee [24].

Otra alternativa puede ser el lidar “Teraranger Evo” de 64 píxeles, ya que este genera una matriz de números de 8 x 8 píxeles permitiendo identificar el obstáculo que se encuentre frente al dron con una percepción máxima de 5 metros de profundidad, de esta manera podríamos identificar de manera más adecuada las dimensiones del obstáculo y las zonas seguras por donde puede transitar el dron sin peligro de colisión.

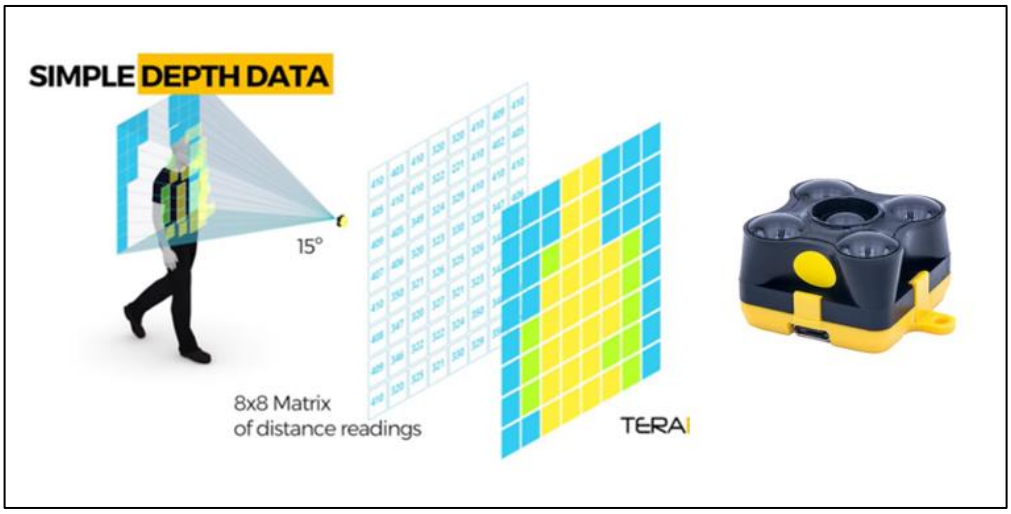


Figura 52. Teraranger evo 64px [24].

Estas dos alternativas mencionadas son de fácil de configuración y transmisión de datos, a través de la interfaz USB en el sistema operativo Windows o Linux. Viene con un SDK basado en OpenNI, muestras de C / C ++, muestras de Python y paquete ROS. Son productos específicos para este tipo de aplicaciones con UAVs por lo tanto son compatibles con sistemas embebidos como la Jetson Nano.

9.3. Modos de vuelo utilizados

Stabilized: el modo 'Estabilizador' te permite volar el vehículo manualmente, pero auto nivela los niveles de 'roll' y 'pitch'.

Auto RTL: en el modo 'Vuelta al lugar de lanzamiento' el UAV navega desde su actual posición, para volar alrededor de su zona de lanzamiento. El comportamiento del modo RTL se puede ajustar por varios parámetros reajustables.

Modo Offboard: el vehículo obedece a un punto de ajuste de posición, velocidad o actitud proporcionado, es decir, en este modo la “PX4” sólo recibe órdenes y no se ajusta a ningún modo de vuelo, sólo sigue las instrucciones dadas.

9.4. Código implementado

Teniendo claro lo anteriormente mencionado se procede a presentar el diagrama de flujo del código implementado en el dron, el cual se basa en el algoritmo seleccionado **zonas seguras.**

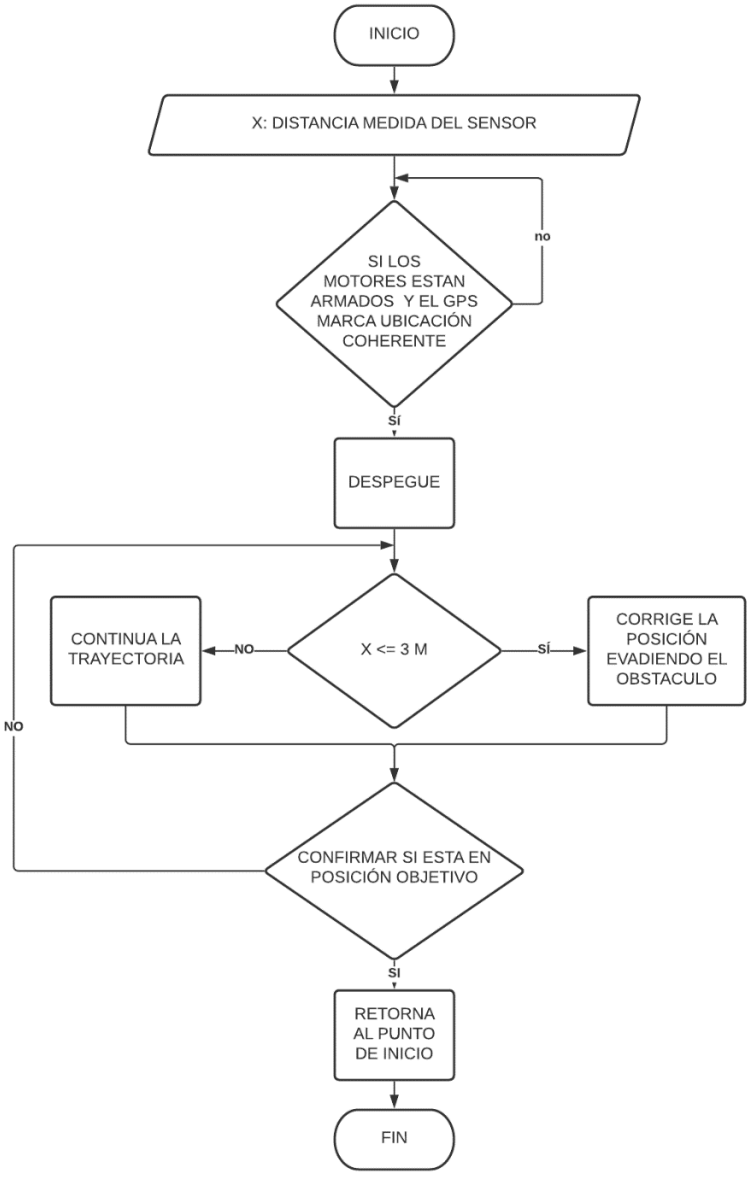


Figura 53. Diagrama de flujo del codigo implementado en el dron [25].

El pseudocodigo que se presenta a continuación hace referencia al código implementado en el dron en modo de vuelo offboard.

Código implementado "zona segura"

Inicio

1: Seleccionar el punto objetivo

2: Definir U_{safe} la distancia entre el UAV y el obstáculo (X_{obs})

*3: Definir **k** la distancia con la que el UAV se mueve lateralmente para evadir el obstáculo*

4: $X(z) = X_{obs}(z)$ Subir a la misma altura del punto objetivo (movimiento en el eje z)

5: Iniciar la lectura del sensor

6: Acción de yaw hacía el objetivo

7: Avanzar en dirección al punto objetivo (X_{goal})

***Si** Posición actual (X_{ini}) ≠ punto objetivo (X_{goal})*

8: Avanzar hacia el punto objetivo

***Si** lectura de sensor <= zona de tolerancia (U_{safe})*

*9: Mantener posición actual X_a y desplazar **k** metros a una dirección lateral*

La dirección lateral (izquierda o derecha) la define el usuario en el código y será siempre la misma

***Mientras** lectura de sensor <= zona de tolerancia (U_{safe})*

*10: desplazar **k** metros a una dirección lateral*

Fin mientras

El movimiento lateral se hará las veces necesarias hasta que el sensor deje de marcar un obstáculo, luego planeará una trayectoria al objetivo

11: Dirigirse al punto objetivo (X_{goal})

Sino

12: Continuar con la trayectoria inicial (X_a)

***Si** Posición actual (X) = punto objetivo (X_{goal})*

Fin si

Fin si

13: Aterrizar

9.5. Pasos para el despegue del dron:

1. Despues de realizar la calibración de los sensores del dron se procede a realizar la conexión entre la computadora a bordo (Jetson Nano) y la estación en tierra.

2. Luego de haber establecido la conexíon se procede a activar la comunicación entre computadora a bordo (Jetson Nano) y el controlador de vuelo (PX4) por medio de comando ya prestablecidos por ROS.

3. Siguiente a esto se abre otro terminal[5] en el cual se ejecuta los codigos previamente realizados y almacenados en la computadora a bordo.

9.6. Items a tener en cuenta:

- Los sensores deben estar calibrados de una manera coherente y correcta.

- El control debe estar encendido y estar en una posición diferente a STABILIZED ó AUTO RTL.

- La velocidad de comunicación entre la computadora a bordo (Jetson Nano) y la controladora de vuelo (PX4) debe ser de 921600 baudios para poder ejecutar los codigos.

- Activación de switch de seguridad para permitir el vuelo.

- Confirmar que la computadora a bordo y la controladora de vuelo estan comunicadas correctamente.

- Comprobar que el espacio frente al sensor de profundidad esta libre.

9.7. Funcion de seguridad

En el momento en el que se realiza la calibración se ajustan los modos de vuelo en el canal de radio control, de manera que sí el usuario siente que el dron ya no responde a los codigos establecidos activa por medio del radio control esta función la cual lleva al dron a tomar una altura de mas o menos 8 metros y luego vuelve a la posición de aterrizaje.

[5] Terminal: una consola que permite a los usuarios controlar los detalles del sistema operativo, permitiendo ejecutar todo tipo de binarios o archivos.

10. RECOLECCIÓN DE DATOS

10.1. Planeación de trayectorias

Finalmente al finalizar la misión el dron almace información sobre la prueba que se realizo, arrojando datos de posición en X,Y y Z, la respuesta de los controladores el yaw, pitch y roll, entre otros datos.

A continuación presentaremos la información adquirida por el dron en una prueba de planeación donde el objetivo era realizar una trayectoria a 4 puntos distantes.

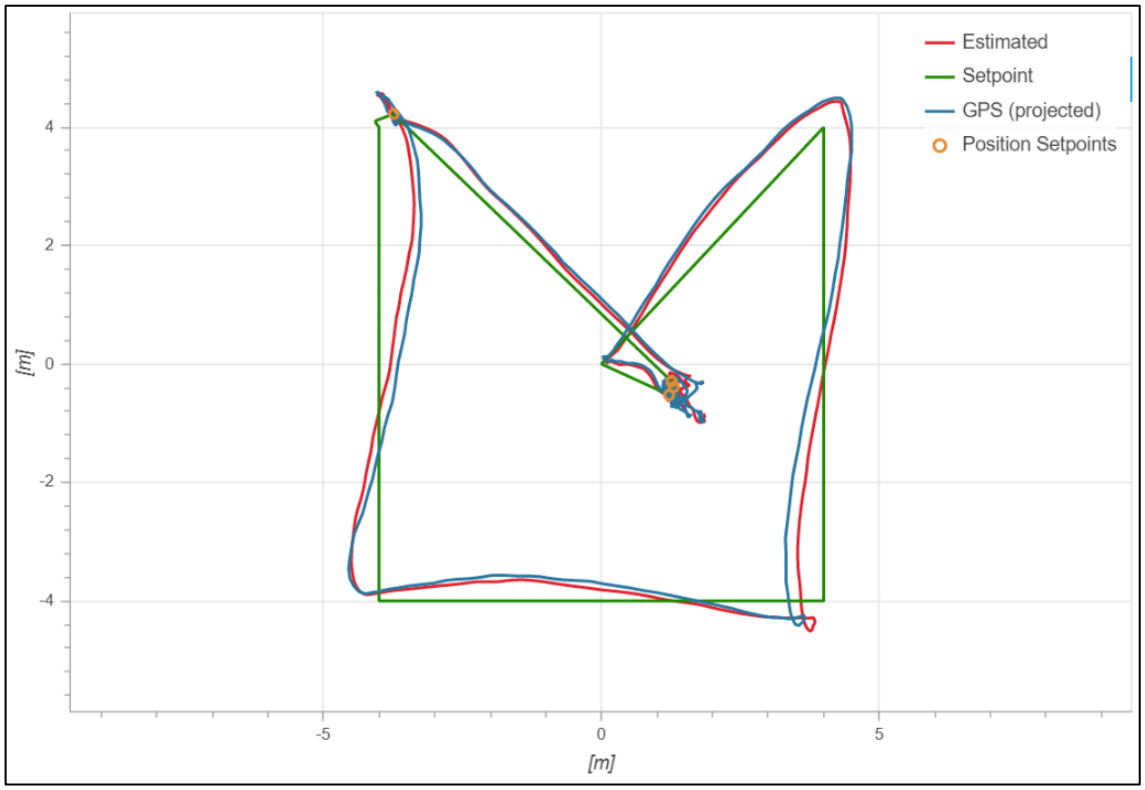


Figura 54. Datos de la prueba de planeación "4 puntos". En verde la trayectoria deseada, en rojo la trayectoria obtenida, en azul la posición obtenida por el GPS y el naranja los set points [25].

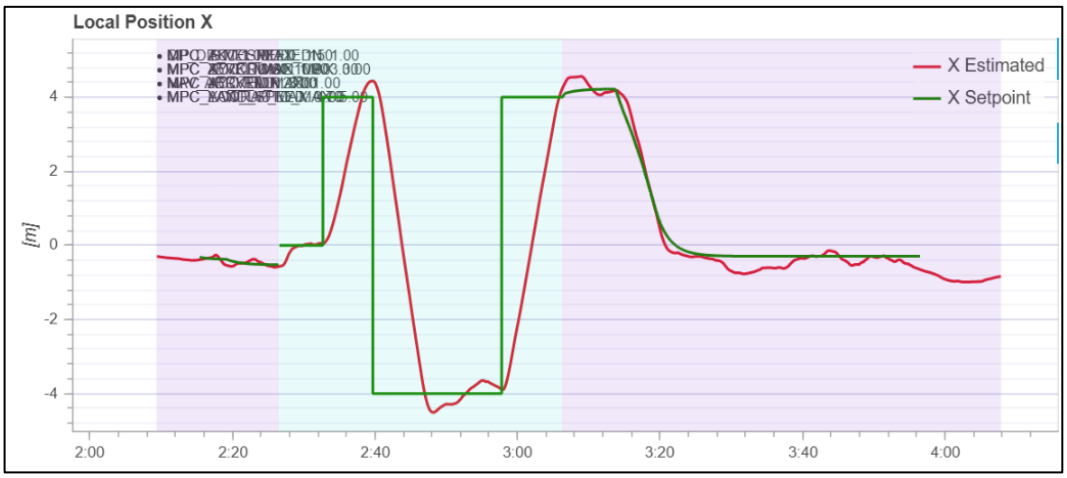


Figura 55. Datos de la posición en X (TCE[6], metros) de la prueba "4 puntos" [25].

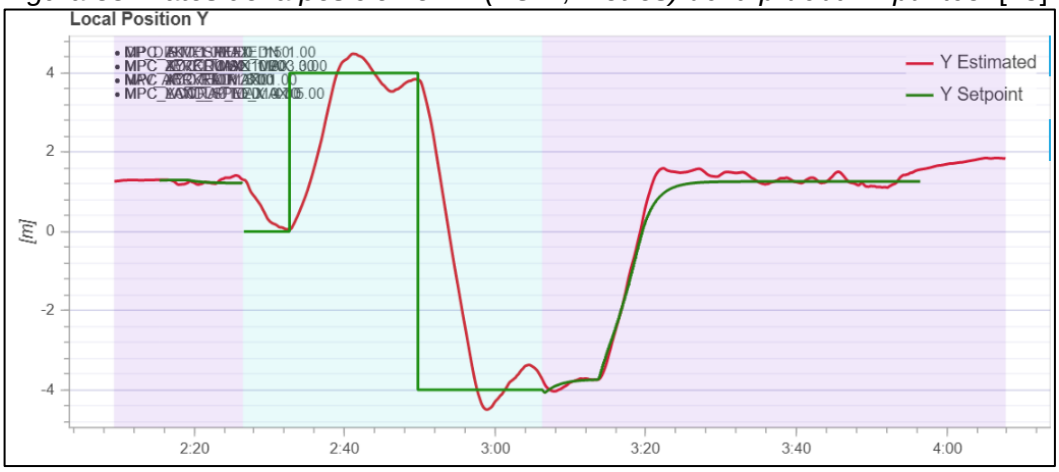


Figura 56. Datos de la posición en Y (TCE, metros) de la prueba "4 puntos" [25].

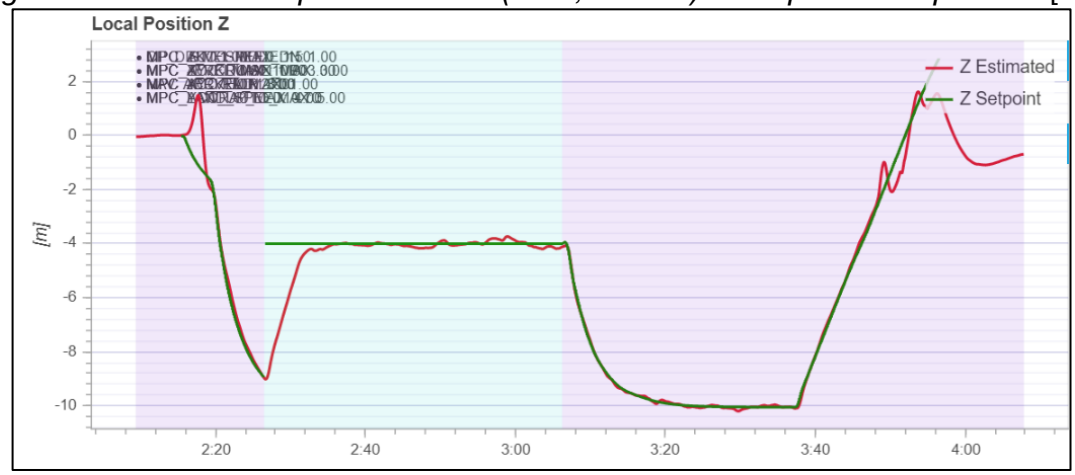


Figura 57. Datos de la posición en Z (TCE, metros) de la prueba "4 puntos" [25].

[6] TCE: tiempo que lleva la controladora de vuelo encendida

Figura 58. Datos de ángulo (TCE, grados) de la prueba "4 puntos" con respecto al eje de aviación Roll [25].

Figura 59. Datos de ángulo (TCE, grados) de la prueba "4 puntos" con respecto al eje de aviación Pitch [25].

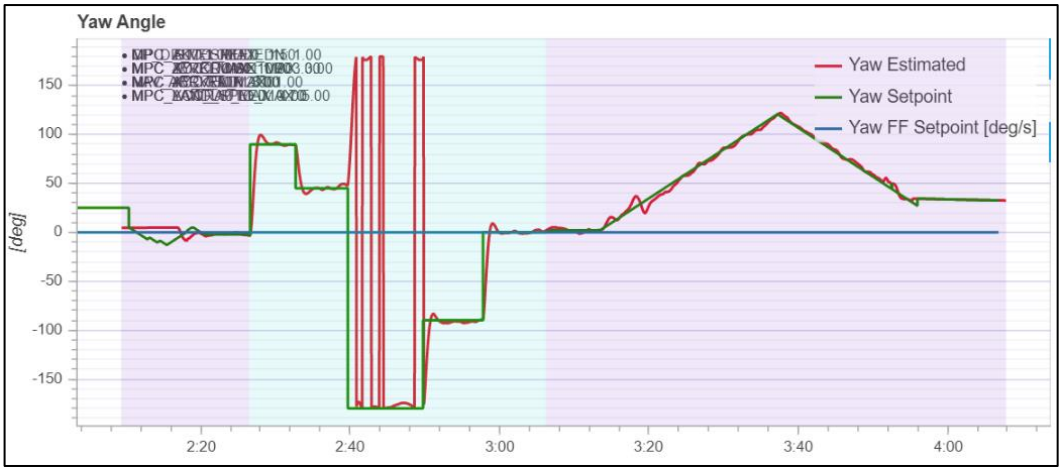


Figura 60.Datos de ángulo (TCE, grados) de la prueba "4 puntos" con respecto al eje de aviación Yaw [25].

10.2. Evasión de obstáculos

El objetivo de esta prueba era realizar la evasión del arco de futbol que se encontraba en la sede campo alegre de Cajasan al cual se le adecuó para ser detectada por el drone (en este caso se le colocó una sábana) y el sensor realizara la lectura adecuada.

Las condiciones de operación en este caso eran que el drone recorriera 10 metros y en tal caso de encontrar un objeto a menos de 1 metro durante la trayectoria se moviera 3.5 metros hacia la derecha de su posición actual para evitar el obstáculo y retomar el punto objetivo inicial.

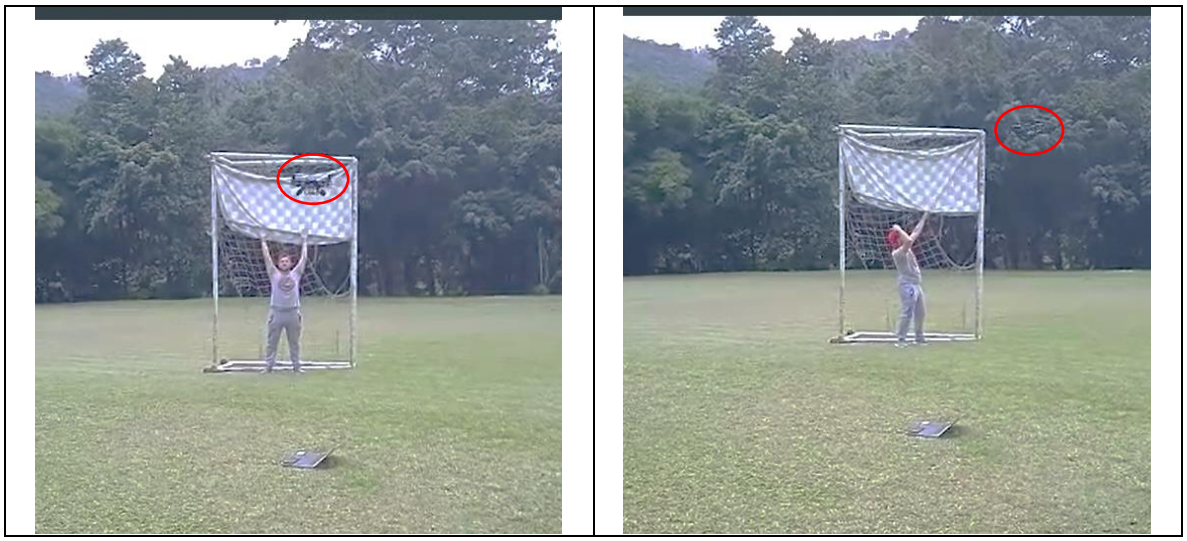


Figura 61. Validación fisica del algoritmo de planeación de trayectorias con evasión de obstáculos en un ambiente controlado [25].

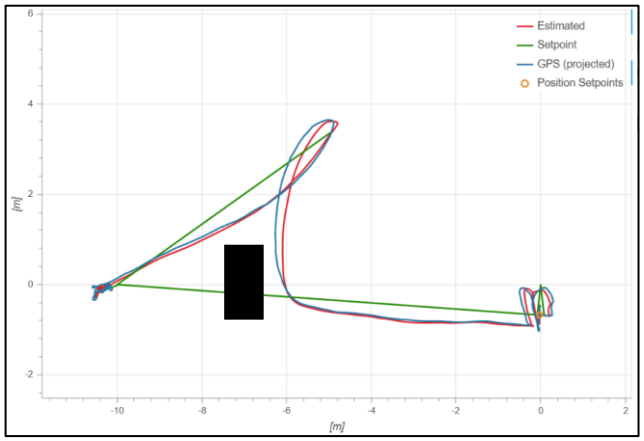


Figura 62. Datos de la posición estimada vs el setpoint [25].

Nota: En la figura anterior el recuadro negro representa el obstáculo que en este caso fue un arco, no viene integrado en los plot de arroja el controlador.

Como se puede observar en la Figura 59, el setpoint inicial era que se moviera 10 metros y al detectar el obstáculo lo que hizo fue detener el recorrido de la trayectoria inicial moviéndose 3.5 metros hacia la derecha de su posición actual (en este caso fue aproximadamente a los 5 metros de iniciar el recorrido) hasta evadir el obstáculo y al realizar dicho movimiento generar la trayectoria nuevamente al punto meta inicial. Se puede observar que la trayectoria luego de la evasión converge en el punto meta que se destinó al comienzo de la prueba.

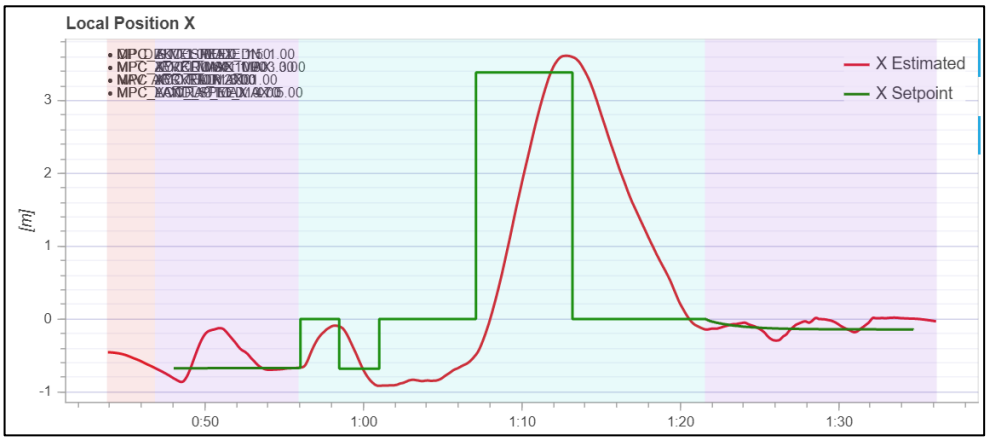


Figura 63. Datos de la posición en X [25].

En la Figuera 60, se puede observar como varía la trayectoria planteada inicialmente para luego converger en el punto establecido al comienzo.

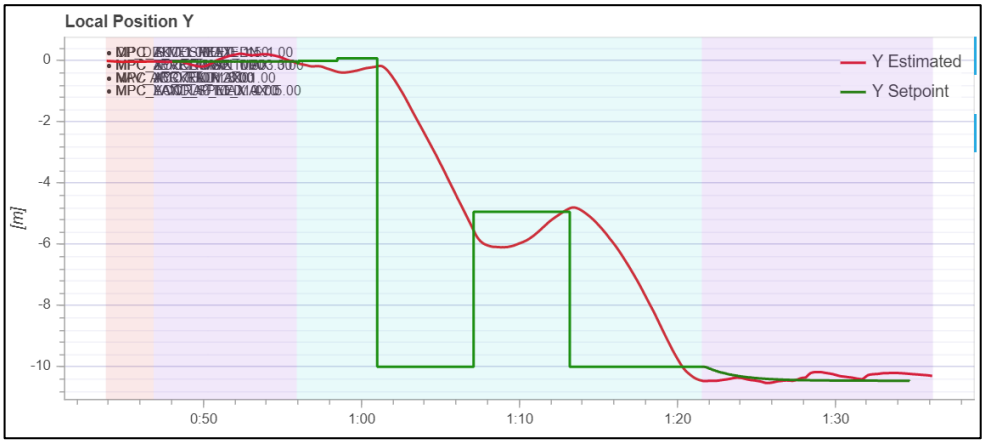


Figura 64. Datos de la posición en Y [25].

En la Figura 61 podemos observar el momento en el que se realiza la evasión mediante la creación de un setpoint a aproximadamente 3.5 metros para evadir el obstáculo encontrado durante el vuelo.

11. CONCLUSIONES

- Se presentaron dos algoritmos que permiten que el dron tipo Quadrotor se movilice de un punto inicial a un punto objetivo realizando un proceso de evasión de obstáculos estáticos. Los algoritmos necesitan como datos de entrada información del entorno, que, para el caso de simulación, conocer de antemano la ubicación de los obstáculos y para el caso de la implementación en el dron, ubicación, altitud, orientación, nivel y los datos que nos proporciona el sensor de profundidad acerca de los obstáculos. Se presento la simulación que permite observar el funcionamiento de los algoritmos en algunos escenarios y con diferentes parámetros de sintonización.

- Fue una decisión acertada integrar las técnicas del descenso del gradiente con las técnicas de puntos independientes móviles y zonas seguras, ya que el gradiente es una técnica muy utilizada en el estado del arte para realizar planeación de trayectorias, esto permite obtener buena información acerca de sus ventajas, desventajas y sus posibles problemas. Además, es una técnica sencilla de programar y que permite ajustar de manera fácil la trayectoria a través de la variación de parámetros.

En base a la simulación de Matlab

- El algoritmo de puntos independientes móviles como se pudo observar en los resultados presenta una trayectoria que converge, que está totalmente libre de obstáculos y no es tan susceptible a la hora de realizar variaciones en los parámetros, sin embargo, se observa que en 3D su trayectoria no es la más optima, ya que al mínimo cambio de parámetros puede agregar oscilaciones incluso puede que no complete su recorrido. Finalmente, en un resultado de convergencia se observa que es una trayectoria un poco agresiva ya que presenta cambios de dirección con ángulos mayores a 35° y al rodear los obstáculos presenta oscilaciones.

- El algoritmo de permanencia en la zona segura presenta una trayectoria en 2D y 3D susceptible a la variación de parámetros, sin embargo, en los resultados de convergencia se aprecia una trayectoria muy controlada en los dos casos: 2D y 3D. Este algoritmo a diferencia del mencionado anteriormente no presenta cambios de ángulo bruscos, no presenta oscilaciones, por el contrario, presenta una trayectoria suave y controlada.

- Una de las mayores ventajas que tiene el algoritmo de permanencia en la zona segura contra el algoritmo de puntos independientes móviles es el control que tiene el usuario sobre la distancia que hay entre el obstáculo y el dron, aunque, con el algoritmo de puntos independientes móviles algunos parámetros, como el radio de detección, presenta un notable cambio en el distanciamiento no es tan

controlado como con el algoritmo de zonas seguras por lo que en este se tiene un parámetro especifico que nos define esta distancia.

- En la simulación los algoritmos puntos independientes móviles tiene una ventaja sobre permanencia en zona seguras porque su tiempo de ejecución es menor al tiempo de ejecución de permanencia en la zona segura. Esta ventaja contrasta con su cantidad de líneas de código ya que el algoritmo de zonas seguras es más reducido por el uso de una función en comparación a puntos independientes móviles que requiere de dos.

En base a la implementación y validación

- Para la implementación de deber tener en cuenta varios aspectos que permiten que el UAV realice un correcto despegue y aterrizaje, la distribución de sus componentes, la correcta calibración de sus sensores y la buena comunicación entre sus componentes principales, para así a medida que el dron recorra la trayectoria puede obtener datos exactos del ambiente y pueda realizar una evasión de obstáculos satisfactoria.

- Para la selección de los sensores se debe tener en cuenta que no generen interferencia entre ellos, en algunos casos las cámaras cuentan con magnetómetro que genera ruidos o valores diferentes en los datos que arrojan el resto de los sensores utilizados para reconocer el entorno.

- Un parámetro clave para el buen desarrollo de las pruebas de campo es la ganancia del controlador que viene predeterminado y se puede configurar, en este caso en la plataforma QGround control. Al tener un cambio en los componentes del quadrotor, la distribución y el peso cambian y por consecuencia el valor de la ganancia del controlador sería erróneo, provocando así accidentes y pruebas de vuelo fallidas.

- De la recolección de datos se concluye que: el dron logró realizar una planeación satisfactoria siguiendo una trayectoria apropiada, pasando por los puntos indicados, cumpliendo con los sets point de altura y de posición en el eje X y Y finalmente realizando un retorno al punto donde inicio, además, evita el obstáculo propuesto en su trayectoria recta, sin embargo, por algunas inconsistencias en los datos tomados por el sensor no se logró evadir con una prudente distancia el obstáculo.

- En la simulación de software in the loop (SITL) se debe tener en cuenta que el drone no recibe los mismos cambios de parámetros proporcionados en código

como sí lo es en la implementación y la simulación de hardware in the loop (HITL) se debe tener en cuenta la lectura del sensor de la misma manera que se va a implementar ya que el HITL es lo más cercano a la implementación.

12. BIBLIOGRAFÍA

[1] H. T. Herrera y J. Escobar, «aproximación al patromonio geológico y geodiversidad en santafé de Antioquia, Olaya y Sopetrán, departamento de Antioquia, Colombia,» bdigital, 2020.

[2] G. d. Colombia, «RELIEVE COLOMBIANO,» [En línea]. Available: https://colombia-sa.com/geografia/geografia.html.

[3] G. Giralt, R. Sobek y R. Chatila, «A multi-level planning and navigation,» San Francisco, CA, USA, 1979.

[4] A. Rivera, D. Pinzón y F. P. Ortiz, «Planeación de trayectorias para cuadricópteros en ambientes dinámicos tridimensionales,» Universidad Pedagógica Nacional, Bogotá-Colombia, 2015.

[5] D. R. Pinzón, «Desarrollo de un algoritmo de evasión de obstáculos para Quadrotors en ambientes dinámicos utilizando una cámara de profundidad,» Universidad Nacional de Colombia, Bogotá, 2012.

[6] F. Rizo y P. Restrepo, «Planeación de trayectorias para un robot aéreo AR. Drone 2.0 usando GPS,» Pontificia universidad Javeriana. Bogotá, Colombia, 2014.

[7] F. Llofriu, «SLAM estado del arte,» Universidad de la república, Montevideo, Uruguay, 2012.

[8] P. Newman, S. Clark y H. Durrant-Whyte, «A solution to the simultaneous localization and map building (slam) problem,» Robotics and Automation, IEEE Transactions on, 2001.

[9] D. H. Gómez, «Desarrollo de una técnica SLAM para ambientes dinámicos tridimensionales,» Universidad Nacional de Colombia, Bogotá, 2015.

[10] V. Hogman, «Building a 3d map from rgb-d sensors,» Royal Institute of Technology, 2011.

[11] A. E. Ichim, «Rgb-d handheld mapping and modeling».

[12] N. Engelhard, F. Endres, J. Hess, J. Sturm y W. Burgard, «Real-time 3dvisual slam with a hand-held rgb-d camera,» de *Proceedings of the RGB-D workshop on 3D perception in robotics at the European robotics forum*, 2011.

[13] P. Henry, M. Krainin, E. Herbst, X. Ren y D. Fox, «Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments, » de *Proceedings of the IEEE*, 2011.

[14] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers y W. Burgard, «An evaluation of the rgb-d slam system, » de *International Conference on Robotics and Automation*, 2012.

[15] M. A. Borrego, «Desarrollo e implementacion de un metodo de generacion de mapas 3d usando el sensor kinect,» Universidad de Malaga, 2012.

[16] J. Engel, T. Schops y D. Cremers, «Lsd-slam: Large-scale direct monocular slam,» de *Proceedings of 13th European Conference in Computer Vision*, 2014.

[17] de *Vehículos aéreos no tripulados para uso civil. Tecnología y aplicación*, Universidad Politécnica de Madrid, 2007.

[18] UdeSantiagoVirtual, *Planificación de trayectorias,* Recuperado de http://www.udesantiagovirtual.cl/moodle2/mod/book/view.php?id=24819, 2010, octubre 13.

[19] F. Hernández, *¿Qué es el Descenso del Gradiente? Algoritmo de Inteligencia Artificial,* 2018, [Archivo de video]. Recuperado de https://www.youtube.com/watch?v=A6FiCDoz8_4&feature=emb_logo.

[20] O. Khatib, *Real-time obstacle avoidance for manipulators and mobile robots,* In Robotics and Automation Proceedings. 1985 IEEE International Conference on, volume 2, pages 500 – 505, mar 1985.

[21] *MATLAB vs. R.,* Retrieved 21 November 2020, from https://www.mathworks.com/discovery/matlab-vs-r.html, 2020.

[22] I. Meza, *Descenso por gradiente (Gradient descent),* Recuperado de https://turing.iimas.unam.mx/~ivanvladimir/posts/gradient_descent/, 2016.

[23] D. Vallejo, A. Barriola y I. Reyes, *TERMODRON Dron de vuelo autónomo y reconocimiento por termografía,* Uruguay, 2017.

[24] terabee. (2019, 6 marzo). Terabee 3Dcam 80x60, the compact and affordable 3D Time-of-Flight camera! [Vídeo]. YouTube. https://www.youtube.com/watch?v=mL79jjGGbvo

[25] Montes, F., Mogollon, R. (2020). Estudiantes Universidad Autónoma de Bucaramanga. Programa de Ingeniería Mecatrónica. Bucaramanga: UNAB.

13. ANEXOS

Para poder tener una perspectiva más clara de la realidad a la hora de simular los algoritmos se realizó un entorno controlado en “Build Virtual Reality Worlds” para poder simular diferentes objetos con diferentes tamaños y realizar la simulación con un dron tipo Quadrotor la configuración realizada fue la siguiente:

Virtual Reality Toolbox es un paquete de herramientas de MATLAB que permite visualizar simulaciones de sistemas dinámicos en una escena 3D de realidad virtual creada en el lenguaje estándar V RML971. Mediante una sencilla interfaz de funciones orientadas a objetos y bloques, este toolbox enlaza, respectivamente, MATLAB y Simulink con las gráficas de realidad virtual. Utilizando señales de Simulink es posible controlar parámetros como la posición, orientación y dimensión de los objetos definidos en el entorno 3D, generando así una animación de la simulación.

Bloque VR Sink

Este es el bloque que sirve como interfaz entre las señales de un modelo y la escena virtual. Al pulsar dos veces con el ratón sobre este bloque incluido en un modelo de Simulink se abre la ventana de configuración que se muestra en la Figura 62.

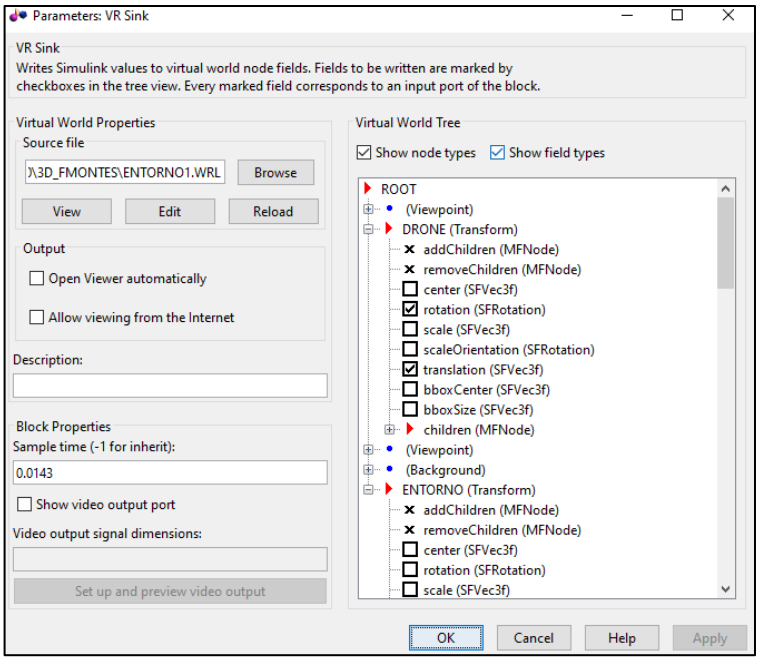


Figura 65. Bloque de parametros.

Los parámetros más importantes son:

Botón Browse: sirve para buscar y seleccionar un archivo. wrl que contenga la escena 3D.

Botón View: abre una ventana del visor con la escena seleccionada.

Botón New / Edit: abre el editor V RML para crear una escena 3D nueva o editar la escena seleccionada (requiere que V- Real Builder esté instalado).

Botón Reload: sirve para recargar la escena seleccionada en caso de haber la editado.

Casilla Open VRML Viewer automatícally: cuando está seleccionada, se abre la ventana del visor automáticamente cuando se abre el modelo.

Sample time: el tiempo de muestreo. Es el intervalo de tiempo entre cada actualización de valores.

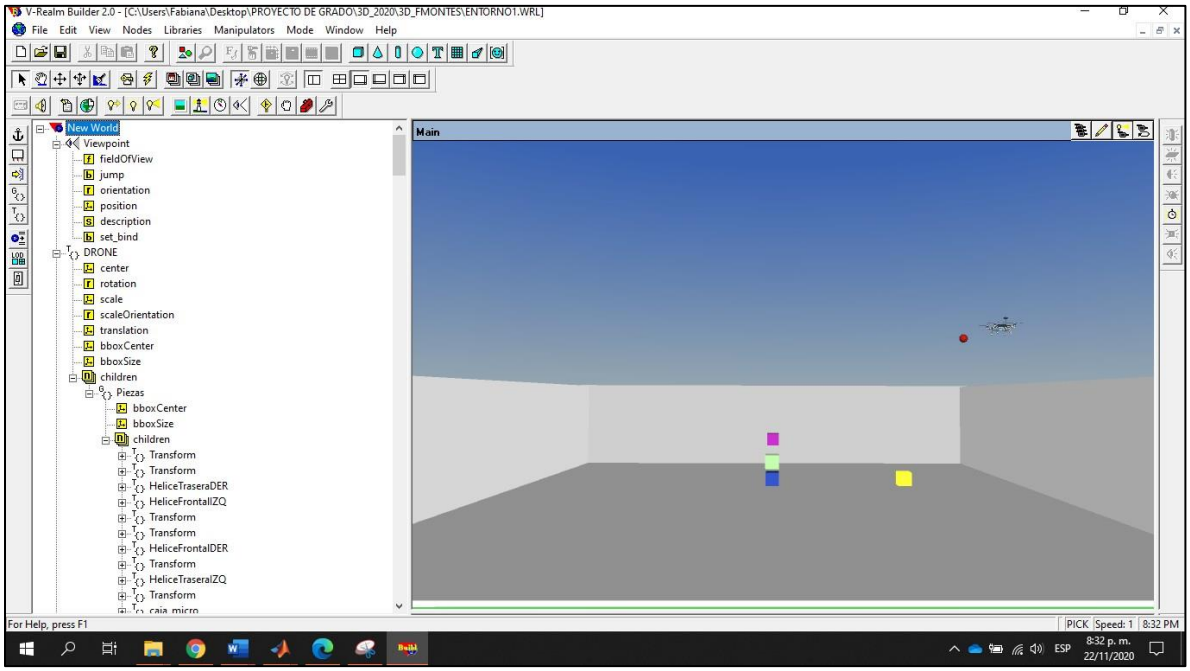


Figura 66.Captura de pantalla del editor del Virtual Reality Toolbox.

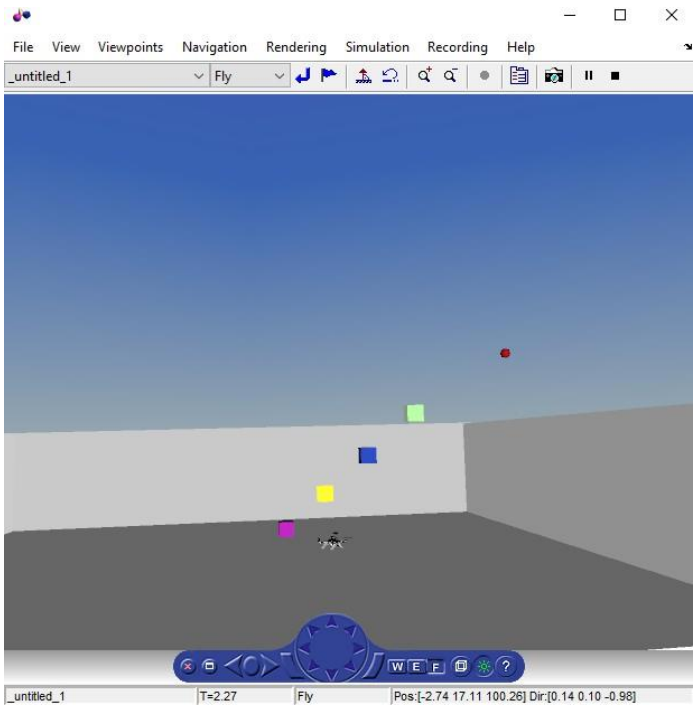


Figura 67. Simulación de la estrategia de puntos independientes moviles iniciando su trayectoria en una escena 3D de realidad virtual.

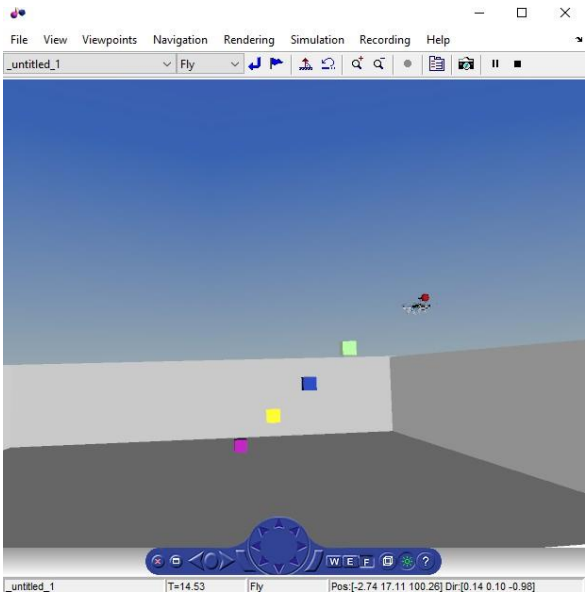


Figura 68. Simulación de la estrategia de puntos independientes moviles finalizando su trayectoria en una escena 3D de realidad virtual.