Artículo de Investigación

Planificación de trayectoria determinista basado en recompensas para entornos discretos 3D

Reward-based deterministic path planning for discrete 3D environments

Gloria Vanegas Zabala [1*], Andrea Liger Yépez [2]

[1] *Tecnología Superior en Redes y Telecomunicaciones, Instituto Superior Tecnológico Bolívar, Ambato, Ecuador, 180103*

[2] *Tecnología Superior en Contabilidad, Instituto Superior Tecnológico Bolívar, Ambato, Ecuador, 180103; andre.ligery@gmail.com*

*Correspondencia: glory.vanegas@gmail.com

Citación: Vanegas, G., & Liger, A., (2023). Planificación de trayectoria determinista basado en recompensas para entornos discretos 3D. *Novasinergia. 6*(2). 154-167.

https://doi.org/10.37135/ns.01.12.10

Recibido: 15 mayo 2023
Aceptado: 21 junio 2023
Publicación: 14 julio 2023

Novasinergia
ISSN: 2631-2654

Resumen: Diversas ramas de estudio e investigación surgen de la tecnología de los vehículos aéreos no tripulados (UAV). Una tarea relevante en vuelo UAV se centra en la planificación de trayectorias tri-dimensional (3D), tarea que implica un alto costo computacional, en consecuencia, debe ser resuelta mejorando el tiempo de respuesta. El objetivo de este trabajo es optimizar el tiempo de cálculo y determinar una trayectoria completa de vuelo 3D. En este sentido, se considera un entorno de vuelo 3D como una malla discreta adaptativa 3D, la cual se somete a un refinamiento mínimo en busca de espacios libres de colisiones. Con la construcción de la malla discreta 3D, se aplica una metodología de coste-respuesta a modo de un Autómata Finito Determinista Discreto (DDFA), metodología que da como resultado un conjunto de respuestas óptimas parciales (calculadas recursivamente) que indican los espacios libres de colisión en la trayectoria 3D final para el vuelo del UAV. Como resultado, el algoritmo de planificación de trayectorias 3D ha mostrado un ahorro en tiempo computacional y recursos de memoria en comparación con las técnicas clásicas.

Palabras clave: Planificación de trayectoria 3D, trayectoria óptima, UAV.

Abstract: *Several branches of study and research arise from unmanned aerial vehicle (UAV) technology. A relevant in-flight task focuses on path planning in 3D, which implies a high computational cost and, consequently, must be achieved by improving the response time. The aim of this work is to optimize the computation time and determine a complete 3D path. In this sense, a 3D flight environment as a 3D adaptive discrete mesh is considered that is subjected to minimal refinement in search of collision-free spaces. With the construction of the discrete mesh, a cost response methodology is applied in the manner of discrete deterministic finite automaton (DDFA), which results in a set of optimal partial responses (recursively computed) that indicate the collision-free spaces in the final 3D path for the UAV flight. As a result, the path-planning 3D algorithm saves computational time and memory resources compared to classical techniques.*

Keywords*: Optimal path, path planning 3D, UAV.*

1. Introducción

Los avances científicos de los UAVs en los campos de la detección (Nevalainen et al. 2017), las comunicaciones (Amorim et al. 2017), los sistemas de control (Yongqiang et al. 2009), etc., han dado lugar a una explotación constante de la tecnología actual. En particular, el mercado mundial de los UAVs se encuentra en plena expansión, hasta el punto que en la actualidad existen varias previsiones y proyecciones dentro del mercado de los UAV. En específico, en el Sistema Nacional de Espacio Aéreo en Estados Unidos se prevé un impacto económico de la integración de los UAVs, y un crecimiento sustancial que alcanzará más de 82,1 mil millones de dólares entre 2015 y 2025 (Valavanis y Vachtsevanos 2015).

Durante décadas, el desarrollo de los UAVs ha tenido como objetivo sustituir a los pilotos humanos en diversas misiones. Las diferentes tecnologías de los UAVs, están en constante evolución, por lo que su desarrollo tecnológico no se ha detenido, independientemente de las soluciones que se presenten sobre determinados problemas. Así, incluso cuando se han completado soluciones a problemas concretos, dichas soluciones han sido objeto de nuevos cambios y modificaciones, lo que demuestra que la resolución sobre las diferentes problemáticas presentes en la temática UAV continúa en estado abierto y de ahí la necesidad de un desarrollo más profundo.

La tarea de planificación de trayectorias y evasión de obstáculos en un espacio euclídeo 3D, ha sido abordada desde diversos paradigmas metodológicos, ya sean empíricos o heurísticos. Estas diversas metodologías parten de una base de muestreo continuo o discreto (Aguilar y Morales 2016; Yao, Wang, y Su 2015), algoritmos basados en nodos (Dijkstra 1959; Hart y Nils 1968; Nosrati, Hasanvand, y Original 2012; Verscheure et al. 2010), algoritmos bioinspirados (Gautam y Verma 2014; Goel et al. 2018; Iswanto, Wahyunggoro, y Cahyadi 2016), entre otras técnicas (Wang, Chu, y Mirjalili 2016; YongBo et al. 2017). Metodologías desarrolladas sobre entornos: con o sin obstáculos (estáticos o dinámicos), por lo tanto, presentan ventajas y desventajas (Szirmay-Kalos y Márton 1998). Por otro lado, es importante destacar que, la teoría de la complejidad y análisis de algoritmos (AofA) involucrada en esta diversidad de metodologías conlleva una alta carga computacional $O(n \log n)$ (Chivers et al. 2015) como se puede ver en la Tabla 1.

Tabla 1: Costo computacional y complejidad en la estructura del grafo, donde n es el número de vértices y m es el número de aristas.

Método	Tiempo de Complejidad	Memoria	Tiempo Real
Algoritmos de muestreo	$O\ (n \log n)$	$O(n^2)$	On-line
Algoritmos basados en nodos	$O(m \log n)$	$O(n^2)$	On-line
Algoritmos bioinspirados	$O(n \log n)$	$O(n^2)$	On-line

En concreto, este trabajo se enfoca en la determinación de trayectorias en entornos 3D, mediante muestreo discreto. En este sentido, se ha tomado como punto de partida la metodología de descomposición adaptativa de celdas (ACD), puesto que presenta sólidas características que resuelven sistemas físicos dominados por ecuaciones diferenciales parciales (Berger y Oliger 1984). Es importante destacar que, esta técnica ofrece una sustancial mejora en tiempo computacional, además que la discretización no se rige por un sistema de ecuaciones dominante. De la misma forma, la ACD se utiliza en reconstrucciones cartesiana 3D (Hasbestan y Senocak 2018). No obstante, el enfoque presentado en este trabajo no intenta realizar una reconstrucción refinada del entorno, se centra en la determinación de los espacios ocupados y espacios libres dentro del espacio cartesiano 3D. Por tanto, se busca alcanzar un ahorro de esfuerzo computacional y de memoria, tomando una

estructura computacional por medio de un etiquetado rápido de la figura geométrica del entorno como un sólido 3D de forma rectangular.

Gran parte de la literatura en el campo de la planificación de trayectorias se centra en la optimización de la distancia. Partiendo de este hecho, este trabajo busca incluir no sólo la distancia como parámetro principal en la generación de una trayectoria 3D, sino que también se pueden incluir otras características, que influyan tanto en el desplazamiento (geométricas del entorno), como posibles restricciones de vuelo (velocidad, capacidad de giro, batería, etc.), entonces, el objetivo es alcanzar una trayectoria 3D que tenga en cuenta todas las restricciones posibles en un tiempo de cálculo mínimo, como se ha definido en la sección 2, donde se detallan diferentes restricciones de vuelo. Por tanto, el lector puede agregar características y condiciones adicionales para la generación de una trayectoria 3D, partiendo de sus requerimientos geométricos y restricciones de vuelo.

Es importante señalar que la tarea de planificación de trayectorias implica completar dos sub tareas fundamentales dentro del Espacio de trabajo $\mathbb{W}$. Estas tareas implican posición y orientación, entendiéndose como posición $p(x, y, z)$ y orientación $o(\varphi, \psi, \theta)$. Entonces, el objetivo es alcanzar otra posición y orientación definidas dentro de $\mathbb{W}$, ya sea individualmente o ambas a la vez. En este sentido, se destaca que el procedimiento descrito a continuación, completa la tarea de planificación de trayectorias en posición, para lo cual se definen un punto inicial $q_i = [x_i, y_i, z_i]$ y un punto meta $q_g = [x_g, y_g, z_g]$ (la orientación será tratada en un próximo trabajo, a través de curvas suaves, siendo ejemplos relevantes de estudio los trabajos planteados por (Armesto, Vanegas, y Girbés-Juan 2022; Girbés, Vanegas, y Armesto 2019; Vanegas et al. 2018, 2022).

Entonces, si se asume las características completas del $\mathbb{W}$. Así, como las dimensiones 3D (X, Y, Z) del $\mathbb{W}$, los obstáculos (dimensiones y ubicaciones) dentro del $\mathbb{W}$, el punto de partida q_i del UAV, y el punto de meta q_g. Entonces, el objetivo es realizar una descomposición discreta (parcial y recursiva) del $\mathbb{W}$, con el fin de determinar el conjunto finito de octantes contiguos y libres de colisiones que unen q_i con q_g. La trayectoria final genera un vector de puntos 3D $\rho_k = [x_k, y_k, z_k]$ con $k = 1, \dots, n$, siendo n el número total de puntos del vector.

La figura 1 muestra un ejemplo de un entorno urbano con edificios de varias dimensiones en distintas ubicaciones. El entorno se somete a una descomposición discreta, que da lugar a espacios libres de colisiones en forma de octantes 3D (color azul). Esto significa que, en función de las características del obstáculo, puede determinarse la trayectoria de escape alrededor de los obstáculos (hacia arriba, hacia abajo, hacia la izquierda o hacia la derecha). Por lo tanto, el UAV puede evitar los obstáculos realizando maniobras 3D verticales y/o horizontales.

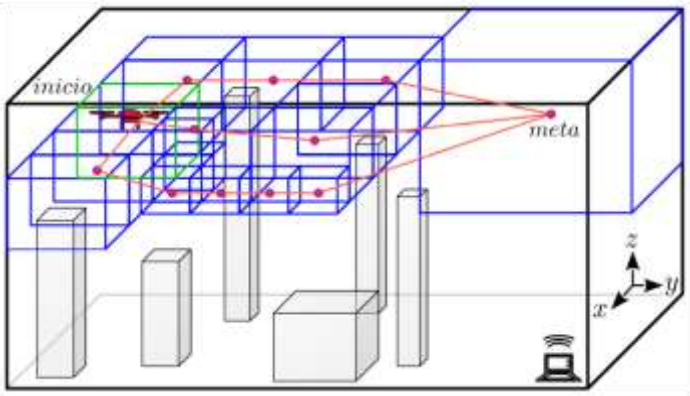


Figura 1: Ejemplo de entorno urbano en 3D (lleno de edificios). Se muestran varias trayectorias posibles (líneas rosas) desde la ubicación actual del UAV y el punto objetivo. Los puntos magenta indican el número de puntos necesarios para construir la trayectoria. Los octantes azules muestran espacios libres de colisiones. El octante verde indica la posición actual del UAV.

Este trabajo se organiza de la siguiente forma: En la Sección 2, se explica el nuevo algoritmo de planificación de trayectorias 3D. En la Sección 3 se describen ejemplos de experimentos y sus resultados. La sección 4 expone una breve disción de los resultados. Finalmente, las conclusiones y trabajos futuros se consideran en la sección 5.

2. Metodología

Para completar el problema de planificación de trayectorias 3D, a continuación, se describe la metodología desarrollada en este trabajo. Entonces, partiendo de la definición de las condiciones iniciales (características espaciales de $\mathbb{W}$, dimensiones y ubicaciones de los obstáculos O , estados inicial q_i y final q_g), se determina cada posible estado futuro del sistema, donde, cada estado actual se define como un octante S_{free} y el conjunto de posibles estados futuros se define como el conjunto de octantes vecinos S_{free}. Por lo tanto, se asume Γ como un espacio de trabajo discreto 3D, que contiene un conjunto finito de octantes libres de colisiones S_{free} y un conjunto finito de octantes ocupados S_{occup}.

Si se asume que el UAV está incluido dentro de la localización inicial, que a su vez se considera como el estado inicial s_k, el objetivo es alcanzar el punto objetivo q_g. Entonces, se asume un conjunto formado por octantes de diferentes tamaños S_{k+1} como vecindad de S_k . En este contexto, se puede desarrollar un modelo de estado y una matriz de transición, donde se determina la transición óptima de S_k a cualquier estado perteneciente a S_{k+1} basándose en dos medidas de transición (D_1 y D_2). Por lo tanto, partiendo del estado actual s_k , el método intenta obtener el estado vecino óptimo perteneciente a $S_{k+1}(s_k \rightarrow S_{k+1} = D_1)$. A continuación, se localiza la sub trayectoria desde cada vecino de S_{k+1} hasta el punto final q_g, siendo la mejor $\left(S_{k+1} \rightarrow q_g = D_2\right)$.

Para resolver este problema se ha definido un autómata finito determinista discreto (DDFA) (Skoldstam, Akesson, y Fabian 2007), siendo $F = (S, G, D, q)^T$ un conjunto de funciones parciales R_m, donde, los parámetros q son dos puntos en el espacio del entorno 3D, mientras que q_i representa el punto inicial, y q_g representa el punto final. Por otra parte, S define el conjunto finito de M estados actuales, para lo que S_{free} define el conjunto finito de octantes libres de colisión, divididos en el octante actual $s_k = [s_k(x), s_k(y), s_k(z)]$ y el conjunto de sus vecinos $s_{k+1} = [s_{k+1}(x), s_{k+1}(y), s_{k+1}(z)]$. En este sentido, S_{occup} representa el conjunto finito de octantes ocupados. Finalmente, $R_m, m = 1 \dots N$ define el conjunto de las N funciones parciales que intervienen en las características de navegación 3D del UAV y determinan el progreso factible. En este trabajo, se definen $N = 2$ parámetros de vuelo, definidos tal que:

$$R_1(i,j) = \frac{M_d}{M_{trD}} \in \mathbb{R}: [0,1]$$
$$M_d\left(s_i \rightarrow s_j\right) = \sqrt{\left(s_i - s_j\right)^2} \quad (1)$$

donde, $M_{dist}\left(s_i \rightarrow s_j\right)$ es la distancia euclídea entre dos estados cualesquiera y M_{trD} es la distancia en línea recta entre q_i y q_g, lo cual busca alcanzar una minimización de distancia en trayectoria de vuelo directo entra octantes S_{free}.

Por otra parte, es importante destacar que toda batería consume energía, la cual se agota después de un determinado tiempo, entonces, el cálculo de la trayectoria debe considerar este consumo. Por una parte, para aeronaves comerciales, el cálculo del consumo se define por medio de la relación:

$$\frac{d_m}{d_t} = -c \quad (2)$$

donde m representa la masa del vehículo y c el gasto másico de combustible. La consecuencia del consumo de combustible, significa la masa es una función del tiempo. Por lo tanto, cuanto menor sea la variación de la masa, menor será el consumo de combustible.

No obstante, debido a las características del UAV tomado para este trabajo (se considera una masa constante puesto que se requiere de una alimentación de energía exclusiva de baterías eléctricas), se buscar realizar una estimación de energía consumida durante la trayectoria de vuelo. Entonces, a partir de método de Euler mejorado, se ha realizado una aproximación del cálculo de consumo, lo cual se define como:

$$R_2(i,j) = \frac{csm}{M_{trD}}$$
$$csm(s_i \rightarrow s_j) = \iiint_W f(x,y) = \int_{a_1}^{a_2}\int_{b_1}^{b_2} f(d_x, d_y) \quad (3)$$

donde, $csm(s_i \rightarrow s_j)$ representa el consumo de batería para la trayectoria entre los estados vecinos. Trayectoria definida como el área de vuelo 3D, la cual es menor al área total de vuelo desde el punto inicial hasta el punto final de trayectoria en vuelo directo. De esta forma, el área de vuelo se define a partir de los límites definidos por los ejes coordenados (x, y, z), tal que: $b_1 \leq y \leq b_2$ y $a_1 \leq x \leq a_2$.

Además, se utiliza una función gaussiana $g(R_m)$ para determinar la recompensa en la ejecución de una posible acción y se define como:

$$g(R_m) = \frac{\sin(\pi * Rm + \pi/2) + 1}{2} \quad (4)$$

donde, los valores de coste de transición $(s_k \rightarrow S_{k+1}, S_{k+1} \rightarrow q_g)$ se han normalizado dentro de los límites [0,1]. Obsérvese cómo cuanto mayor es el esfuerzo R_m, menor es la recompensa $g(R_m)$ y viceversa. Por lo tanto, la ejecución de una acción desde el estado s_i a diferentes estados s_j produce transiciones de estado a diferentes costes.

Todas estas recompensas pueden expresarse como un vector $G(i,\ j)$ de parámetros de vuelo como:

$$G(i,j) = \left[g(R_1(i,j)), g(R_2(i,j)), \dots, g(R_N(i,j))\right]^T \quad (5)$$

$D \in \mathbb{R}^{M-2}$ es la recompensa recibida asociada a una prioridad $\mathfrak{p} \in \mathbb{R}^N$ por ejecutar una acción sobre una función $g(R_m)$ y se expresa como la suma de dos vectores de prioridad de transición (D_1 y D_2) definidos como:

$$D_1 = (\mathfrak{p} \times G(i,j)) + \xi \quad [i = 1, j = 2 \dots (M-1)] \quad (6)$$
$$D_2 = (\mathfrak{p} \times G(i,j)) + \xi \quad [i = 2 \dots (M-1), j = M] \quad (7)$$
$$D = D_1 + D_2 \quad (8)$$

donde ξ es un valor de recompensa negativo predefinido en cada estado perteneciente a S_{k+1}. Se debe notar que los valores de la distribución de probabilidad de las funciones $g(R_1), g(R_2), \dots, g(R_N)$ son independientes y el conjunto de vectores de respuesta D son mapeos de $S \times S_{free}$. En consecuencia, este mapa se genera con una distribución de probabilidad independiente del tiempo. Lo que significa que, la probabilidad de pasar de un instante al siguiente no cambia.

Por lo tanto, el mejor valor de recompensa del vector D genera la mejor trayectoria final x, denotada por $\rho_x(F)$, definida por un grafo finito etiquetado con vértices $S_x \in S_{free}$.

2.1 *Ejemplo de aplicación*

Para explicar la metodología, a continuación, se plantea un problema de planificación de trayectoria, donde el punto inicial se define como $q_i \in s_k$ (definido como estado actual), mientras que el punto meta se define como q_g. En este sentido, se considera un octante maestro que contiene un obstáculo en su interior. La figura 2 muestra el escenario inicial, así como los diferentes estados S_{k+1} (observables y vecinos) que pueden convertirse en un nuevo S_k. En este caso, en $t = 0$, el costo de moverse a la derecha, al frente o hacia abajo, implica el mismo costo. Esta situación aparece debido a la simetría inherente a la metodología de descomposición, en frente a tal situación, el azar decide el siguiente estado.

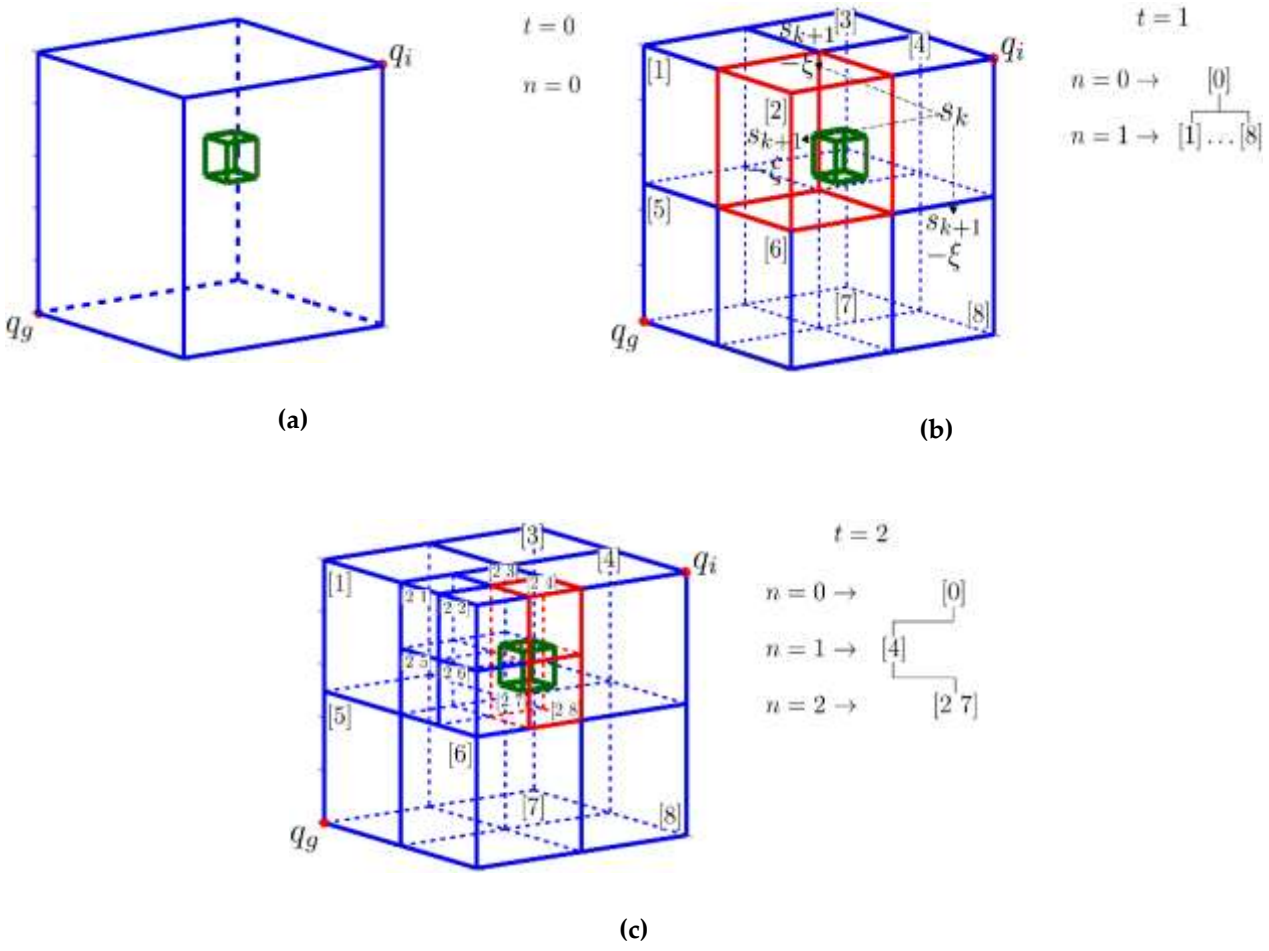


Figura 2: Ejemplo de proceso de inicio MACD de recompensa recursiva (3D-PTDR). (a) Entorno completo. (b) Primer nivel MACD $n = 1$. (c) Segundo nivel MACD $n = 2$.

Como puede verse, el árbol octante se divide en diferentes niveles de octantes con diferentes características dimensionales en diferentes ubicaciones espaciales. Con el objetivo de obtener el conjunto finito de espacios libres de colisiones S_{free}, MACD se realiza recursivamente. Por otra parte, dado que un estado no puede apuntar a sí mismo y sólo puede visitarse una vez, cuando se visitan y evalúan los S_{k+1} estados, se selecciona uno de ellos produciendo un movimiento de reenvío (véase la Figura 5b). Por tanto, el estado seleccionado es el nuevo punto inicial s_k, y el proceso se repite de nuevo (véase la Figura 5c). La estructura de red mostrada indica un movimiento de avance s_k hacia el estado inmediatamente siguiente s_{k+1}. Este movimiento es independiente de cualquier estado anterior s_k.

Los principales límites del entorno han sido definidos desde las coordenadas iniciales $q_i = (x_i, y_i, z_i)$ hasta la meta $q_g \equiv (x_f, y_f, z_f)$, resultando una forma rectangular, siendo $env =$

$([x_i, x_f], [y_i, y_f], [z_i, z_f])$. Cada obstáculo se define como $h_i(x, y, z) \in \mathbb{R}^3 \rightarrow (x, y, z) = \lambda$, y este es definido como:

$$h_i(\lambda)|_t = h_i(\lambda)|_{t+1} \Rightarrow obstáculo\ estático \tag{9}$$

donde $h_i(\lambda) \rightarrow i > 0$ representa un conjunto de obstáculos estáticos.

La figura 5a muestra la definición del entorno (líneas azules) como nodo principal $n = 0$ con un obstáculo $h_1(\lambda)$ situado en su interior (recuadro líneas verdes). Una vez realizada la primera descomposición (ver figura 5b), un primer nivel octeto $n = 1 \rightarrow \{[1] \dots [8]\}$ se genera con nodos que tienen diferentes propiedades de ocupaciones. Cada uno pertenece a S_{free} si no hay $h_i(\lambda)$ dentro de los límites de su octante ($s_k \cap h_i(\lambda) = 0$) (líneas azules), o a S_{occup} si el octante está parcial o totalmente ocupado por el obstáculo $\big((s_k \cap h_i(\lambda) = 1) \vee (s_k \in h_i(\lambda))\big)$ (líneas rojas).

El primer paso consiste en determinar el octante contenedor q_i (para este ejemplo, es $n = 1 \rightarrow \{[4]\}$). En caso de detección de obstáculos en {[4]}, se realizaría una descomposición recursiva en la ubicación. En este nivel, el nodo {[4]} es el nuevo estado inicial s_k y los vecinos observables S_{k+1} son los nodos {[2] [3]} y {[8]}. En esta fase, el modelo de estado se representa en la Figura 3.

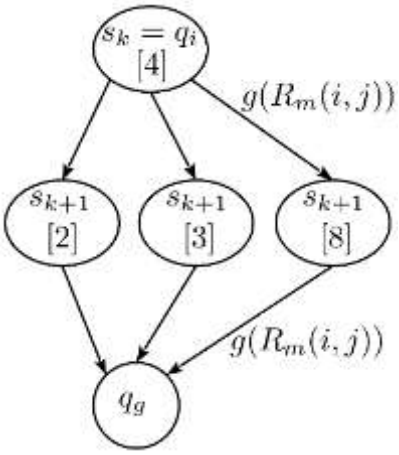


Figura 3: Distribución de probabilidad para los estados discretos, los cuales se derivan de la matriz multidimensional de dimensiones $M \times M \times N$, donde M es el número de estados y N es el número de funciones parciales $R(_m)$ con $m = 1 \dots N$, que intervienen en la navegación 3D del UAV.

De esta forma, se construye una matriz multidimensional con la información del modelo de estado y los valores de recompensa. El objetivo es encontrar las respuestas parciales procedentes de la primera fila y la última columna, y dividirlas en dos vectores de prioridad de transición $(D_1, D_2) \in \mathbb{R}^{(M-2)}$. Utilizando el vector de prioridad $\mathfrak{p}$ y el desplazamiento ξ, los vectores D_1 y D_2 entregan distribuciones parciales de recompensa a un posible siguiente estado.

Entonces, el vector de prioridad de transición D_1 se construye con el conjunto de columnas $j = 2 \dots (M - 1)$ y la fila $i = 1$ tal que:

$$D_1 = \mathfrak{p} \times G(i,j) + \xi, i = 1, j = 2 \dots (M-1)$$
$$G(1,(M-1)) = \begin{bmatrix} g\left(\mathrm{R}_1(1,(\mathrm{M}-1))\right) \\ g\left(\mathrm{R}_2(1,(\mathrm{M}-1))\right) \\ . \\ . \\ . \\ g(R_N(1,(M-1))) \end{bmatrix} \tag{10}$$

El segundo vector de prioridad de transición, D_2, se construye con el conjunto de filas $i = 2 \dots (M - 1)$ y la columna $j = M$.

$$D_2 = \mathfrak{p} \times G(i,j) \ + \xi, i = 2 \ldots (M-1), j = M$$
$$G((M \ - \ 1), M) \ = \begin{bmatrix} g\left(\mathrm{R}_1\left((\mathrm{M}-1), M\right)\right) \\ g\left(\mathrm{R}_2\left((\mathrm{M}-1), M\right)\right) \\ . \\ . \\ . \\ g(R_N((M-1), M)) \end{bmatrix} \quad (11)$$

Finalmente, el vector de recompensa final D expresado como la suma de D_1 y D_2 contiene el valor óptimo que señala el mejor estado para continuar la búsqueda del camino ρ_x:

Para continuar con el ejemplo de la Figura 4, se asume que $x \ = \ best(D)$ apunta hacia el estado {[6]}. Sin embargo, el estado {[2]} está ocupado (pertenece a S_{ocup}), por lo que se invoca a MACD, creando un nuevo nivel en la estructura de datos, compuesto por {[2 1] [2 2] [2 3] [2 4] [2 5] [2 6] [2 7] [2 8]} (véase la Figura 4c). Aunque el estado s_k permanece en [4], la nueva descomposición en [2] devuelve un nuevo conjunto de vecinos, que se unen a los anteriores y definen el nuevo conjunto S_{k+1} definido por {[3] [2 3] [2 4] [2 7] [2 8] [8]}.

Por tanto, es necesario buscar el óptimo dentro de S_{k+1}. Entonces, asumiendo que el mejor nodo desde s_{k+1} es [2 3] (nótese que MACD se invoca una vez hasta ahora) y por tanto el nuevo estado s_k se reasigna a [2 3] y en consecuencia, la vecindad del nuevo estado s_k, queda conformada por $S_{k+1} = \{[2\ 1]\ [2\ 4]\ [2\ 7]\ [1]\ [4]\}$.

Las acciones anteriores producen el desplazamiento desde un estado s_k actual al siguiente mejor estado, y hacia el punto final. Aunque el octante contenedor del mejor estado resultante x no contiene el punto de meta, existe la posibilidad de que x tenga vecinos en distintos niveles de descomposición. Por lo tanto, el proceso continúa hasta que se alcanza el punto de meta. Una vez completada la fase anterior, el camino óptimo $\rho_x(F)$ queda totalmente determinado y la búsqueda finaliza.

En realidad, el procedimiento se divide en dos etapas. En primer lugar, se define una ubicación de octante inicial, entonces, si existe un $h_i(\lambda)$ en el entorno, se realiza una descomposición simple inicial, en consecuencia, se define el octante libre de colisiones que contiene el punto inicial. Una vez asignado el estado inicial s_k que también es un ρ_x inicial, se procede en función de su ocupación. Si s_k está libre de colisiones, se asignan los vecinos S_{k+1}, se calculan las recompensas y se localiza el óptimo x. Por tanto, el mejor estado x se convierte en el nuevo s_k y se añade a ρ_x. En el caso que este nuevo s_k este ocupado, el proceso requiere otra descomposición en el s_k actual, y s_k volverá a su estado anterior. El procedimiento finaliza cuando el s_k actual contiene el punto objetivo y s_k está libre de colisiones.

La metodología descrita presenta varias propiedades, siendo: (a) Define un proceso estocástico en tiempo discreto, entonces, la distribución de probabilidad para un estado futuro depende únicamente de sus valores presentes y es independiente del pasado del estado actual. (b) La suma de las prioridades definidas en el vector $\mathfrak{p}$ no es igual a 1. $\sum_{i=1}^{N} \mathfrak{p}_i \ i \neq 1$. (c) La suma de los valores de cada vector de prioridades, no es igual a 1. $\sum D_1 \neq 1$ y $\sum D_2 \neq 1$.

Finalmente, es importante destacar que la descomposición discreta del entorno da lugar a octantes cada vez más pequeños y de distintos tamaños. El nivel de descomposición de los octantes es variable en función de dos objetivos que deben cumplirse, siendo: (1) El diseñador define el nivel máximo de descomposición (tamaño mínimo del octante); sin embargo, el algoritmo intenta reducir el coste computacional, en consecuencia, se suele evitar el tamaño mínimo del octante. (2) En cuanto un espacio libre cumple las restricciones definidas, el algoritmo lo selecciona independientemente del tamaño del octante.

3. Resultados

Con el objetivo de analizar el rendimiento del algoritmo propuesto, se han aprovechado las prestaciones del sistema de cálculo numérico "MATLAB versión 9.9.0.1467703 (R2020b)". El algoritmo se ha ejecutado en "Intel(R) Core(TM) i7-7700HQ CPU @ 2.80 GHz, 1 procesador físico; 4 núcleos; 8 hilos" con 8Gb RAM y S.O. Ubuntu Linux 20.04.2 LTS.

Para evaluar el algoritmo 3D-PTDR propuesto, se han diseñado 10 escenarios de simulación ($\mathbb{W}$ que incluyen uno o más obstáculos estáticos en su interior), no obstante, para efectos de simplificación en la Tabla 3 se detallan 2 escenarios. En concreto, cada escenario tiene dimensiones espaciales de $X = (100[m], Y = 100[m], Z = 100[m])$. De esta forma, cada escenario ha sido sometido a 10 ejecuciones, resultados que han sido comparados con los resultados generados por MACD.

Tabla 2: Definición de los distintos $\mathbb{W}$ 3D para los ejemplos de simulación. Tanto las ubicaciones de los objetivos como las de los obstáculos y sus dimensiones se han definido en metros [m].

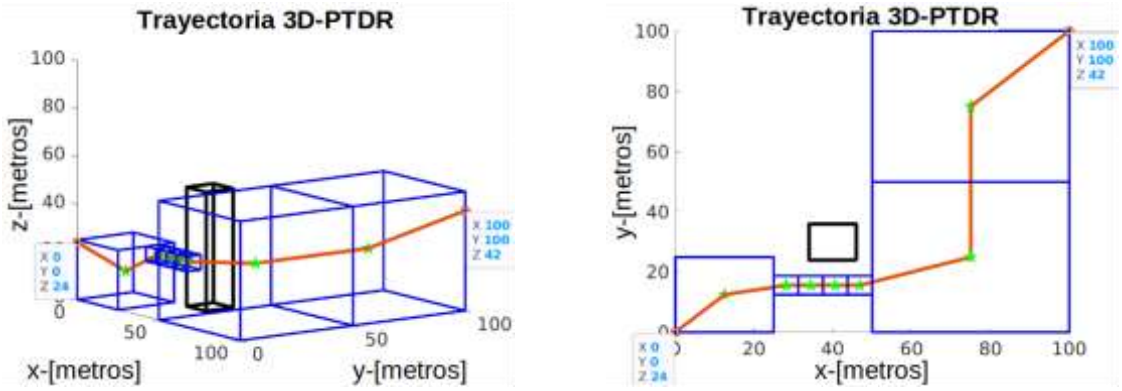
Ubicaciones de la trayectoria objetivo [m]							**Obstáculos [m]**						
#	**Inicio**			**Meta**				**Dimensiones**			**Ubicación**		
							#	**x**	**y**	**z**	**x**	**y**	**z**
1	100	100	42	0	0	24	**1**	12	12	50	40	30	25
2	96	60	30	12	15	45	**1**	10	10	10	40	40	50
							2	5	5	5	60	60	80
							3	6	6	6	70	80	50
							4	15	15	15	70	70	70

En específico, la Tabla 2 se divide en 2 columnas principales que son: "Ubicaciones de la trayectoria objetivo" y "Obstáculos". En la primera columna se detallan las ubicaciones específicas de los objetivos inicio y meta (se han definido diferentes objetivos 3D para cada escenario), mientras que en la segunda columna se muestran las dimensiones, ubicaciones y número de obstáculos para cada escenario. Es importante resaltar que, en el primer escenario, se ha localizado un obstáculo, mientras que, en el segundo escenario se han definido cuatro obstáculos.

La Figura 6 muestra el ejemplo de trayectoria construida por 3D-PTDR para el primer escenario, donde las cajas negras son los obstáculos $hi(\lambda)$, las estrellas verdes muestran el conjunto de vértice en el estado s_k y la línea naranja muestra la trayectoria final $\rho_x(F)$. De esta forma, la Figura 6(a) muestra una vista 3D de la trayectoria construida mediante 3D-PTDR en el primer escenario, mientras que la Figura 6(b) muestra una vista perpendicular (desde el plano ortogonal al eje-z) del mismo escenario #1. Finalmente, la Figura 7 representa los resultados 3D-PTDR y MACD para el escenario #2, donde las Figuras 7(a) y 7(c) muestran una vista 3D, finalmente, las Figuras 7(b) y 7(d) muestran los resultados de ρ_x con vista perpendicular al eje-z.

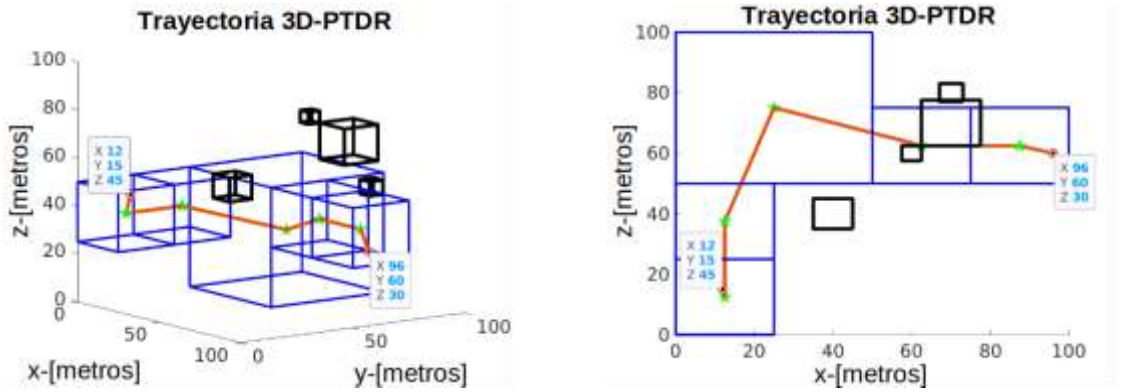
4. Discusión

Con el fin de proporcionar una comparación de rendimiento entre los algoritmos descritos, la Tabla 3 detalla los resultados en tiempo de descomposición discreta a lo largo de cada escenario, el número de octantes libres de colisión S_{free}, el número de nodos generados en la trayectoria final ρ y la longitud de la trayectoria. Es importante destacar que cuando un obstáculo $h_i(\lambda)$ se cruza con una descomposición de octantes, MACD continúa recursivamente hasta el nivel n predefinido (por esta razón el tiempo de búsqueda de MACD es considerablemente mayor que 3D-PTDR), además, MACD requiere cálculos adicionales definidos por el planificador de Dijkstra's para obtener el camino óptimo final ρ.

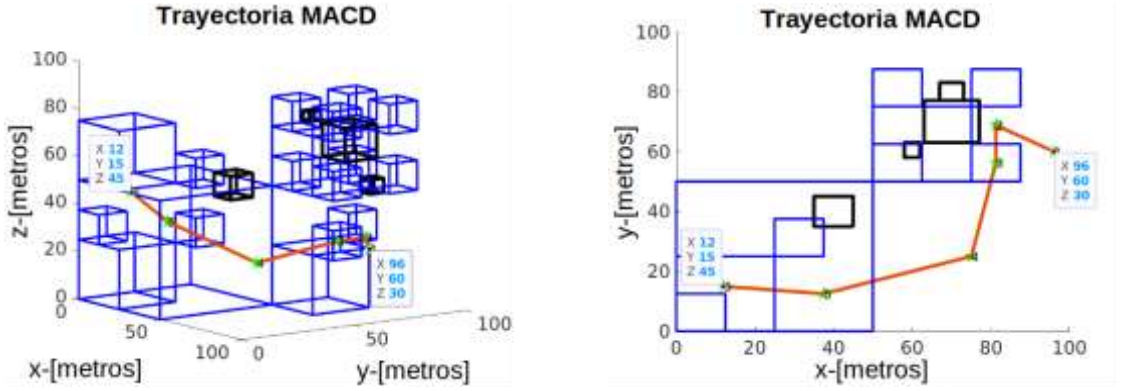


(a) **(b)**

Figura 7: Resultados de trayectoria del escenario #1 visto desde diferentes perspectivas. (a) Trayectoria generada por 3D-PTDR con perspectiva de vista 3D (54, 12). (b) Resultados del escenario #1 con la vista desde el *plano* (x, y).



(a) **(b)**



(c) **(d)**

Figura 8: Escenario #2, visto desde diferentes perspectivas. (a) Trayectoria generada por 3D-PTDR con perspectiva de vista (54°, 12°). (b) Trayectoria generada por 3D-PTDR con perspectiva de vista del plano (x, y). (c) Trayectoria generada por MACD con perspectiva de vista (54,12). (d) Trayectoria generada por MACD con perspectiva del plano (x, y).

Tabla 3: Resultados de las diversas ejecuciones sobre los escenarios descritos.

#	MACD					3D-PTDR			
	Tiempo de descomp. (s)	**Tiempo Dijkstra (s)**	**# S_{free}**	**# ρ**	**Longitud (m)**	**Tiempo de descomp. (s)**	**# S_{free}**	**# ρ**	**Longitud (m)**
1	0.14265	0.00189	65	7	173.43682	0.00166	43	9	177.36750
2	0.15187	0.00119	55	5	132.73110	0.00016	19	6	146.76870
3	0.17352	0.01513	107	6	130.32670	0.00514	27	7	152.44530
4	0.12765	0.00130	44	7	175.81597	0.00017	19	6	171.25050
5	0.18137	0.00314	77	7	147.79795	0.00022	27	5	121.72630
6	0.41876	0.00158	49	5	152.67823	0.00476	11	6	199.9279
7	0.29923	0.00147	61	5	152.67823	0.00018	11	6	169.2270
8	0.26390	0.00118	55	9	184.97604	0.00023	19	5	139.02570
9	0.23216	0.00118	66	7	158.83471	0.00339	67	13	127.6446
10	0.15697	0.00088	42	6	173.12528	0.01133	27	7	208.0392

De esta forma, la tabla 4 muestra las diferencias en términos porcentuales entre los algoritmos. Se distingue una diferencia sustancial en los distintos parámetros. Por lo tanto, la diferencia en el tiempo de cálculo es evidente debido al tiempo adicional de Dijsktra en MACD. Por ejemplo, en el primer escenario, el algoritmo 3D-PTDR sólo necesita el 1,15248% del tiempo computacional que MACD necesita para la descomposición discreta, el 66,15385% del total de número de S_{rfee} que MACD, +28,57143% más nodos que MACD necesita para construir ρ, por último, la trayectoria 3D-PTDR es +2,26635% más largo que MACD.

Tabla 4: 3D-PTDR vs. MACD (%) muestra los recursos medios (tiempo de descomposición (s), número de S_{free} y número de nodos en la ruta final "ρ") utilizados para 3D-PTDR en comparación con MACD.

#	3D-PTDR vs MACD			
	Tiempo de descomp. (s)	**# S_{free}**	**# ρ**	**Longitud (m)**
1	1.15248	66.15385	+28.57143	+2.26635
2	0.11041	34.54545	+20.00000	+10.57597
3	0.02966	0.25233	+16.66667	+16.97165
4	0.13725	43.18182	85.71429	97.40327
5	0.12302	35.06494	71.42857	82.35993
6	1.13405	22.44898	+20.00000	+30.94722
7	0.06218	31.14754	+40.00000	+10.83898
8	0.08714	34.54545	55.55556	75.15876
9	1.45357	+1.51515	+85.71429	80.36316
10	7.18182	64.28571	+16.66667	+20.16685

En términos generales, el tiempo computacional conseguido en 3D-PTDR mejora significativamente, debido a la forma en la que se divide $\mathbb{W}$, por lo que el número de S_{free} también es menor.

5. Conclusiones

El primer paso relevante para completar la tarea de planificación de trayectoria 3D, ha estado enfocado en el trabajo sobre un entorno en forma de malla adaptativa, lo cual ha dado lugar a la metodología para la descomposición discreta del entorno 3D. De esta forma, os resultados de la descomposición del entorno han sido utilizados para alcanzar las trayectorias 3D deterministas. No

obstante, es importante destacar que el refinamiento 3D en los diversos experimentos ha sido mínimo (teniendo en cuenta diversas restricciones de vuelo, además de las características geométricas de la $\mathbb{W}$), lo cual provocado un tiempo computacional de cálculo aceptable.

Luego de revisada la bibliografía referente a la planificación de trayectoria 3D en entonos discretos, se ha llegado a un punto relevante en que se menciona de forma reiterada que diversas metodologías requieren de planificadores adicionales como los de Dijkstra, A*, D*, etc., mientras que 3D-PTDR evita la necesidad de utilizar planificadores adicionales. Por otra parte, es importante resaltar que, al proponer una metodología que considera un conjunto de condiciones para la determinación de trayectorias (restricciones de vuelo, que además puede ser ampliada por el lector), las cuales son sometidas a evaluación simultánea. Dando como resultado un conjunto de recompensas, a partir de las cuales se determina la trayectoria final óptima, con un tiempo computacional reducido y menor complejidad de cálculo, en comparación con técnicas similares aplicadas a la tarea de planificación de trayectorias 2D y 3D.

Finalmente, se puede destacar que los experimentos y resultados han demostrado que la metodología propuesta alcanza su objetivo final de generación de trayectoria 3D, no obstante, queda abierta la posibilidad de incrementar las funciones de optimización de trayectoria, tanto en geometría como en restricciones de vuelo.

Contribuciones de los autores

En concordancia con la taxonomía establecida internacionalmente para la asignación de créditos a autores de artículos científicos (https://casrai.org/credit/). Los autores declaran sus aportes en la siguiente matriz de contribuciones:

	Vanegas, G	*Liger, A.*
nceptualización		
Análisis formal		
Investigación		
Metodología		
Recursos		
Validación		
Redacción–revisión y edición		

Conflicto de Interés

Los autores del presente artículo declaran que no existe ningún tipo de conflictos de intereses de ninguna naturaleza.

Referencias

Aguilar, Wilbert, y Stephanie Morales. 2016. 3D Environment Mapping Using the Kinect V2 and Path Planning Based on RRT Algorithms. *Electronics* 5(4):70. doi: 10.3390/electronics5040070

Amorim, Rafhael, Huan Nguyen, Preben Mogensen, István Z. Kovács, Jeroen Wigard, y Troels B. Sørensen. 2017. Radio Channel Modeling for UAV Communication over Cellular Networks. *IEEE Wireless Communications Letters* 6(4):514-17. doi: 10.1109/LWC.2017.2710045

Archdeacon, Dan. 1996. Topological Graph Theory. *A survey. Congressus Numerantium* 115(18):5-54, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.1728

Armesto, Leopoldo, Gloria Vanegas, y Vicent Girbés-Juan. 2022. Elementary Clothoid-Based Three-Dimensional Curve for Unmanned Aerial Vehicles. *Journal of Guidance, Control, and Dynamics* 45(12):2421-31, https://doi.org/10.2514/1.G006935

Berger, Marsha J., y Joseph Oliger. 1984. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics* 53(3):484-512. doi: 10.1016/0021-9991(84)90073-1

Chivers, Ian, Jane Sleightholme, Ian Chivers, y Jane Sleightholme. 2015. An introduction to Algorithms and the Big O Notation. *Introduction to Programming with Fortran: With Coverage of Fortran 90, 95, 2003, 2008 and 77* 359-64, https://doi.org/10.1007/978-3-319-17701-4_23

Dijkstra, E. W. 1959. A note on two problems in connexion with graphs. *Numerische Mathematik* 1(1):269-71, https://doi.org/10.1145/3544585.3544600

Gautam, S. Aditya, y Nilmani Verma. 2014. Path planning for unmanned aerial vehicle based on genetic algorithm & artificial neural network in 3D. Pp. 1-5 en *2014 International Conference on Data Mining and Intelligent Computing (ICDMIC)*. IEEE, https://doi.org/10.1109/ICDMIC.2014.6954257

Girbés, Vicent, Gloria Vanegas, y Leopoldo Armesto. 2019. Clothoid-Based Three-Dimensional Curve for Attitude Planning. *Journal of Guidance, Control, and Dynamics* 42(8):1886-98. doi: 10.2514/1.G003551.

Goel, Utkarsh, Shubham Varshney, Anshul Jain, Saumil Maheshwari, y Anupam Shukla. 2018. Three Dimensional Path Planning for UAVs in Dynamic Environment using Glow-worm Swarm Optimization. *Procedia Computer Science* 133:230-39. doi: 10.1016/j.procs.2018.07.028.

Hart, Peter E., y J. Nils. 1968. A formal basis for the Heuristic Determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics* 4(2):100-107, https://doi.org/10.1109/TSSC.1968.300136

Hasbestan, Jaber J., y Inanc Senocak. 2018. Binarized-octree generation for Cartesian adaptive mesh refinement around immersed geometries. *Journal of Computational Physics* 368:179-95. doi: 10.1016/j.jcp.2018.04.039

Iswanto, Iswanto, Oyas Wahyunggoro, y Adha Imam Cahyadi. 2016. Quadrotor Path Planning Based on Modified Fuzzy Cell Decomposition Algorithm. *TELKOMNIKA (Telecommunication Computing Electronics and Control)* 14(2):655. doi: 10.12928/TELKOMNIKA.v14i1.2989.

Latombe, Jean-Claude, y Jean-Claude Latombe. 1991. Approximate cell decomposition. *Robot Motion Planning* 248-94, https://doi.org/10.1007/978-1-4615-4022-9_6

LaValle, Steven M. 2006. *Planning Algorithms*. Cambridge University Press.

Nevalainen, Olli, Eija Honkavaara, Sakari Tuominen, Niko Viljanen, Teemu Hakala, Xiaowei Yu, Juha Hyyppä, Heikki Saari, Ilkka Pölönen, Nilton N. Imai, y Antonio M. G. Tommaselli. 2017. Individual tree detection and classification with UAV-Based photogrammetric point clouds and hyperspectral imaging. *Remote Sensing* 9(3). doi: 10.3390/rs9030185.

Noborio, Hiroshi, Tomohide Naniwa, y Suguru Arimoto. 1990. A quadtree-based path-planning algorithm for a mobile robot. *Journal of Robotic Systems* 7(4):555-74, https://doi.org/10.1002/rob.4620070404

Nosrati, Masoud, Ronak Karimi Hojat Allah Hasanvand, y including D. Original. 2012. Investigation of the * (Star) Search Algorithms: Characteristics, Methods and Approaches. *World Applied Programming* 2(4):251-56.

Samaniego, Franklin, Javier Sanchis, Sergio Garcia-Nieto, y Raul Simarro. 2017. UAV motion planning and obstacle avoidance based on adaptive 3D cell decomposition: Continuous space vs discrete space. Pp. 1-6 en *2017 IEEE Second Ecuador Technical Chapters Meeting (ETCM)*. IEEE, https://doi.org/10.1109/ETCM.2017.8247533

Samet, H., y A. Kochut. 2002. Octree approximation an compression methods. Pp. 460-69 en *Proceedings. First International Symposium on 3D Data Processing Visualization and Transmission*. IEEE Comput. Soc, https://dx.doi.org/10.1109/TDPVT.2002.1024101

Skoldstam, Markus, Knut Akesson, y Martin Fabian. 2007. Modeling of discrete event systems using finite automata with variables. Pp. 3387-92 en *Decision and Control, 2007 46th IEEE Conference on*. IEEE, https://dx.doi.org/10.1109/CDC.2007.4434894

Szirmay-Kalos, L., y G. Márton. 1998. Worst-case versus average case complexity of ray-shooting. *Computing* 61(2):103-31. doi: 10.1007/BF02684409.

Valavanis, Kimon P., y George J. Vachtsevanos. 2015. *Handbook of Unmanned Aerial Vehicles*. editado por K. P. Valavanis y G. J. Vachtsevanos. Springer Netherlands.

Vanegas, Gloria, Leopoldo Armesto, Vicent Girbés-Juan, y Joaquín Pérez. 2022. Smooth Three-Dimensional Route Planning for Fixed-Wing Unmanned Aerial Vehicles With Double Continuous Curvature. *IEEE Access* 10:94262-72, https://doi.org/10.1109/ACCESS.2022.3203069

Vanegas, Gloria, Franklin Samaniego, Vicent Girbes, Leopoldo Armesto, y Sergio Garcia-Nieto. 2018. Smooth 3D path planning for non-holonomic UAVs. Pp. 1-6 en *2018 7th International Conference on Systems and Control (ICSC)*. IEEE, https://doi.org/10.1109/ICoSC.2018.8587835

Verscheure, L., L. Peyrodie, N. Makni, N. Betrouni, S. Maouche, y M. Vermandel. 2010. Dijkstra's algorithm applied to 3D skeletonization of the brain vascular tree: Evaluation and application to symbolic. Pp. 3081-84 en *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*. IEEE, https://doi.org/10.1109/IEMBS.2010.5626112

Wang, Gai-Ge, Haicheng Eric Chu, y Seyedali Mirjalili. 2016. Three-dimensional path planning for UCAV using an improved bat algorithm. *Aerospace Science and Technology* 49:231-38. doi: 10.1016/j.ast.2015.11.040.

Yao, Peng, Honglun Wang, y Zikang Su. 2015. Hybrid UAV path planning based on interfered fluid dynamical system and improved RRT. Pp. 000829-34 en *IECON 2015 - 41st Annual Conference of the IEEE Industrial Electronics Society*. IEEE, https://doi.org/10.1109/IECON.2015.7392202

YongBo, Chen, Mei YueSong, Yu JianQiao, Su XiaoLong, y Xu Nuo. 2017. Three-dimensional unmanned aerial vehicle path planning using modified wolf pack search algorithm. *Neurocomputing* 266:445-57. doi: 10.1016/j.neucom.2017.05.059.

Yongqiang, Wang, Steven X. Ding, Ye Hao, Wei Li, Zhang Ping, y Wang Guizeng. 2009. Fault detection of networked control systems with packet based periodic communication. *International Journal of Adaptive Control and Signal Processing* 23(8):682-98. doi: 10.1002/acs.