

PER2025–056 - Type : Développement Contrôle de vol d'un nano
drone et détection de dérive comportementale

État de l'art

Etudiant(s) : Sacha CASTILLEJOS & Noé Florence (M2 IoT-CPS), Logan
LUCAS & Momen TAKROUN (SI5 IoT-CPS)

Encadrant(s) : M. Gérald Rocher

Note : Certains paragraphes de ce document ont été reformulés avec l'assistance d'un outil d'intelligence artificielle.

| | |
|------------------------------------------------------------------------------------------------------|-----------|
| 1) Contexte..... | 3 |
| 2) Questions de recherche..... | 3 |
| 3) Méthodologie..... | 4 |
| 3.1) Queries..... | 4 |
| 3.2) Critères d'inclusion/exclusion..... | 4 |
| 3.3) Processus de sélection des papiers..... | 4 |
| 4) Critères d'analyse et lecture des publications retenues..... | 5 |
| 5) Résultats / réponses..... | 6 |
| 5.1) Cartographie 3D..... | 6 |
| 5.1.1) Capteurs..... | 6 |
| 5.1.2) Représentation de l'environnement..... | 6 |
| 5.1.3) Algorithmes et performances..... | 6 |
| 5.1.4) Synthèse des résultats..... | 7 |
| 5.2) Localisation dans un environnement 3D..... | 7 |
| 5.2.1) Localisation par filtrage particulaire et OctoMap..... | 7 |
| 5.2.2) Odométrie LiDAR directe et structures de données recyclées..... | 8 |
| 5.2.3) Localisation par ajustement de plans..... | 8 |
| 5.2.4) Fusion multi-capteurs par filtre de Kalman..... | 8 |
| 5.2.5) Localisation par primitives géométriques pour capteur Solid-State..... | 9 |
| 5.3) Planification de trajectoire 3D..... | 9 |
| 5.3.1) Familles d'algorithmes identifiés..... | 9 |
| 5.3.1.1) Algorithmes basés sur l'Échantillonnage (Sampling-Based)..... | 10 |
| 5.3.1.2) Algorithmes de Recherche Discrète / Graph-Based..... | 10 |
| 5.3.1.3) Algorithmes Méta-heuristiques (Bio-inspirés)..... | 10 |
| 5.3.1.4) Algorithmes d'Optimisation Mathématique..... | 11 |
| 5.3.1.5) Algorithmes Réactifs (Champs de Potentiel)..... | 11 |
| 5.3.1.6) Algorithmes Hybrides..... | 11 |
| 5.3.1.7) Algorithmes d'Apprentissage (Learning-Based)..... | 12 |
| 5.3.1.8) Tableau donnant avantages et inconvénients de chaque famille d'algorithmes..... | 12 |
| Réactifs..... | 14 |
| 5.3.2) Tableau récapitulatif des familles d'algorithme utilisés dans les différents articles : | 15 |
| Références..... | 17 |

1) Contexte

Dans ce projet, nous travaillons sur le nano-drone Crazyflie 2.1 de Bitcraze. Équipé du Flow v2 Deck pour l'estimation et le Multi-ranger Deck pour détecter les obstacles. Ce drone nous permet non seulement de tester des stratégies de pilotage autonome, mais aussi d'analyser les dérives, c'est-à-dire les écarts entre le comportement attendu et ce que le drone fait réellement. Le vrai défi technique est de réussir à mettre en place ce pilotage autonome de A à Z en 3D. Même si la navigation en 2D est déjà gérée par Bitcraze, notre objectif est de passer à un environnement 3D.

2) Questions de recherche

Ainsi, nous avons été amenés à poser les 3 questions suivantes pour nous aider à réaliser notre objectif :

- Comment cartographier un environnement en trois dimensions dans le cadre du SLAM ?
(How to map an environment in 3D for SLAM?)
Cette sous-question permet d'étudier les différentes méthodes de reconstruction 3D, ce qu'elles utilisent comme capteurs, leur précision et les ressources nécessaires (mémoire, ...) et leurs différents avantages et inconvénients pour la navigation.
- Comment localiser le drone dans une carte 3D pour le SLAM ?
(How to do localization in a 3D map for SLAM?)
Cette sous-question permet de recenser et comparer les algorithmes d'estimation de position et d'orientation, et de comprendre leurs limites en termes de précision et rapidité d'exécution (nécessaire pour une localisation en temps réel).
- Quels algorithmes peuvent être utilisés pour générer un chemin entre deux points dans une carte 3D avec évitement d'obstacle pour drones ?
(Which algorithms can be used to generate a trajectory between 2 points in a 3D map with obstacle avoidance for UAV ?)
Cette partie permet d'explorer l'état de l'art des algorithmes de planification de trajectoire dans une carte 3D, cela nous permettra de comprendre lesquels produisent des trajets optimaux, dans quels types d'environnement (statique/dynamique, extérieur/intérieur), leur complexité et les ressources nécessaires pour exécuter l'algorithme.

3) Méthodologie

Bases de données bibliographiques utilisées :

- Google Scholar
- Sci-Hub
- IEEE Xplore
- Elicit (fait plus que ça)

3.1) Queries

Nous avons utilisé les questions vues en 2) comme requêtes.

3.2) Critères d'inclusion/exclusion

Critères d'inclusion :

- La publication aborde un ou plusieurs de nos mots-clés ("drones", "SLAM", "aerial" ...)
- La publication a été évaluée par ses pairs
- La publication est écrite dans un de ces deux langues: anglais ou français
- La publication date d'il y a moins de vingt ans
- La publication a été publiée dans un journal qui est au moins Q1 ou Q2

Critères d'exclusion :

- Duplicata
- Pas de traduction
- La publication ne fournit pas de résultats
- La publication est issue de "predatory publishers"
- La publication se concentre totalement sur des capteurs que le crazyflie n'a pas

3.3) Processus de sélection des papiers

Avec ces différentes requêtes dans divers moteurs de recherche et bases de données nous avons pu observer plusieurs centaines de résultats sur lesquels nous avons appliqué les critères mentionnés ci-dessus, majoritairement à la lecture du titre et aux métadonnées de la publication, pour ne répertorier qu'une centaine de publications.

Une première passe de tri a ensuite été effectuée à partir de la lecture des abstracts, déjà plus chronophages que la simple lecture du titre (une à trois minutes contre quelques secondes), permettant d'évaluer la pertinence scientifique des travaux vis-à-vis de notre problématique globale et de nos sous-questions de recherche. Cette étape a conduit à l'élimination des publications trop éloignées de notre contexte ou que nous avons jugées trop compliquées à mettre en œuvre.

Une seconde et dernière étape, avec les résumés en tête, a consisté à ne conserver que les publications permettant d'assurer la plus grande diversité possible d'approches, de méthodes et de solutions proposées, afin d'obtenir une vision large et représentative de

l'état de l'art. Dans le but de garantir une sélection finale de publication la plus qualitative possible, le nombre de citations, le classement de la conférence et la notoriété de l'éditeur ont été pris en compte pour cette étape.

Ainsi, nous avons abouti à une vingtaine de papiers couvrant les trois questions de recherche.

4) Critères d'analyse et lecture des publications retenues

Pour analyser la pertinence des publications retenues vis-à-vis de notre projet, nous avons établi une grille de lecture pour trier les articles.

- **Matériel utilisé (Raspberry, microcontrôleur, ...)** : Très important pour savoir si le papier a utilisé des ressources plus ou moins conséquentes ou s'il n'y a seulement eu une simulation pour tester l'algorithme (cartographie, localisation, planification de trajectoire).
- **Véhicule terrestre / aérien / domaine d'utilisation** : Identifier pour quels objets s'appliquent les différents algorithmes afin de savoir si nous pouvons l'appliquer.
- **Capteurs utilisés** : Pour pouvoir identifier les algorithmes que l'on peut utiliser avec les capteurs du crazyflie
- **Type de map utilisée / générée** : structure de map qui est utilisée pour générer un trajet et qui est générée par le drone au cours du trajet
- **Environnement (indoors/outdoors)** : Savoir si les tests effectués dans les papiers le sont dans un environnement plutôt libre ou dans un environnement contraint avec donc moins de liberté
- **Environnement (statique / dynamique)** : savoir si la solution est capable de gérer un environnement inconnu ou connu et s'il peut gérer des obstacles statiques ou dynamiques
- **Consommation** : Le crazyflie étant un robot avec peu d'autonomie de vol ce critère est essentiel notamment pour la planification de trajectoire
- **Précision** : précision et incertitude de la solution utilisé/étudié
- **Temps réel** : savoir si le trajet est généré un amont ou en temps réel
- **Reproductibilité** : savoir si la solution traitée dans les articles est facilement applicable ou non à notre projet
- **Prise en compte d'un essaim** : savoir si l'article utilise ou non un essaim de drone
- **Famille d'algorithmes de planification** : Les différentes catégories d'algorithmes pour mieux catégoriser car dans ces différentes familles il existe plusieurs différentes variantes
- **Représentation de l'espace libre pour la planification** : comment le drone se représente les espaces libres pour la planification du trajet
- **Gestion de l'incertitude** : La capacité à réagir face à l'inconnu que ce soit lors de la cartographie, lors de la localisation ou même la planification de trajectoire
- **Stratégie de "Re-planning"** : capacité du système à recalculer la trajectoire si un obstacle est découvert
- **Horizon de planification** :
- **Localisation** : méthode utilisée pour localiser le drone dans l'espace

5) Résultats / réponses

5.1) Cartographie 3D

5.1.1) Capteurs

Il existe trois approches majeures pour la cartographie 3D d'environnements. Celles-ci se distinguent dès le choix des capteurs, lequel conditionne les méthodes de calcul mises en œuvre : les systèmes à base de LiDAR, de caméras RGB-D et de dispositifs stéréoscopiques. Nous avons noté une répartition assez équitable de ces différents capteurs dans les publications, montrant que chacun possède des domaines d'application privilégiés.

Les systèmes dotés de LiDAR démontrent une supériorité marquée dans les environnements extérieurs de grande taille, où ils sont capables de produire des cartes géométriquement très précises, parfois avec des erreurs de fermeture inférieures au millimètre selon les techniques employées. À l'inverse, les systèmes basés sur des caméras RGB-D présentent une portée et une précision plus limitées, mais offrent une densité de points bien plus élevée ainsi qu'une richesse visuelle accrue, ce qui les rend particulièrement adaptés aux environnements intérieurs structurés.

Afin de pallier les limites propres à chaque capteur, plusieurs publications récentes privilégient des approches hybrides reposant sur la fusion multi-capteurs (LiDAR, caméra, IMU), améliorant ainsi la robustesse globale du système dans des environnements complexes ou dégradés.

5.1.2) Représentation de l'environnement

Les méthodes de représentation de l'environnement se déclinent selon trois paradigmes principaux : les approches basées sur les points, les lignes et les plans. Les méthodes basées sur des points, utilisant notamment des descripteurs SIFT, dominent dans 8 des 10 études analysées et offrent une applicabilité générale, bien que limitée en termes d'information sémantique. Les approches exploitant des primitives linéaires, paramétrées via les coordonnées de Plücker, se révèlent particulièrement efficaces dans les environnements structurés riches en arêtes, typiques des constructions humaines. Enfin, la représentation par segments planaires, intégrée dans des frameworks probabilistes de type filtre de Kalman étendu, permet une description compacte et informative des espaces intérieurs structurés.

Les travaux récents convergent vers des représentations hybrides combinant plusieurs types de primitives géométriques, telles que les structures height-voxel (HMAP), qui surmontent les limitations des approches purement 2D ou 2.5D tout en maintenant une efficacité computationnelle acceptable.

5.1.3) Algorithmes et performances

Les composantes algorithmiques centrales identifiées incluent l'algorithme ICP pour l'affinement de l'alignement, l'ajustement de faisceaux (bundle adjustment) pour l'optimisation globale, et les mécanismes de détection de boucles combinant des approches visuelles de type bag-of-words avec l'alignement géométrique. On observe une transition

méthodologique des approches par filtrage (EKF) vers l'optimisation par graphes de poses, reflétant l'évolution des capacités de calcul et permettant une réduction significative de la dérive accumulée.

Les performances quantitatives rapportées démontrent des améliorations substantielles : le système 3D-CSTM atteint une erreur de fermeture de 0,78%, tandis que LV-SLAM réduit l'erreur de trajectoire absolue (ATE) de 6,90 m à 2,48 m sur les benchmarks KITTI, maintenant une précision géométrique inférieure à 5 cm. Les optimisations computationnelles, notamment par sélection d'observations guidée par l'information, permettent des réductions de temps de traitement de 57% tout en préservant la couverture des primitives géométriques.

5.1.4) Synthèse des résultats

L'analyse des différentes études montre une tendance claire vers des architectures hybrides qui combinent plusieurs types de capteurs et de représentations géométriques dans des systèmes d'optimisation unifiés. La distinction entre approches LiDAR et vision n'indique pas tant une supériorité de l'une sur l'autre qu'une adaptation aux différents types d'environnements : les systèmes LiDAR se montrent particulièrement performants dans les grands espaces extérieurs avec des erreurs submillimétriques, alors que les approches RGB-D trouvent leur place dans les environnements intérieurs comportant de larges surfaces planes, là où les méthodes basées uniquement sur la géométrie rencontrent des difficultés.

Les systèmes combinant plusieurs capteurs apparaissent comme une solution particulièrement robuste, maintenant de bonnes performances aussi bien dans les espaces intérieurs privés de GPS que dans les environnements extérieurs à faible texture.

Dans le cas d'usage d'un LiDAR simple, les composantes clés à implémenter sont donc l'extraction de primitives géométriques à partir du nuage de points, l'Iterative Closest Point afin d'aligner les scans et pour finir l'ajout d'un mécanisme de détection de boucle pour retrouver des endroits déjà visités.

Si possible, le simple ajout d'un IMU pour de la fusion de capteur a l'air de drastiquement augmenter la robustesse du système.

5.2) Localisation dans un environnement 3D

5.2.1) Localisation par filtrage particulaire et OctoMap

Pour la navigation d'un drone en environnement intérieur, la localisation est assurée par un filtre particulaire. Le système génère un ensemble de particules représentant des poses

hypothétiques, qui sont mises à jour selon un modèle de mouvement intégrant les mesures d'une IMU et d'un altimètre laser. Chaque particule est pondérée en comparant les données d'un réseau circulaire de capteurs de distance avec les obstacles répertoriés dans une OctoMap. La pose finale du véhicule est déterminée par la moyenne pondérée des particules ayant le score de confiance le plus élevé, garantissant une estimation robuste malgré l'absence de signal GPS.

5.2.2) Odométrie LiDAR directe et structures de données recyclées

Une autre méthodologie privilégie la rapidité de calcul en utilisant des nuages de points denses avec un prétraitement minimal. La localisation s'appuie sur le solveur NanoGICP, une variante optimisée du GICP qui réalise un alignement. Pour maximiser l'efficacité sur des plateformes à ressources limitées, les structures de données et les calculs de voisinage sont systématiquement recyclés entre les modules de localisation locale et de cartographie globale. Le système s'ancre sur des keyframes dont la création est modulée par la configuration de l'espace, assurant une précision constante tout en limitant la charge mémoire.

5.2.3) Localisation par ajustement de plans

Dans les environnements intérieurs riches en structures géométriques, la localisation repose sur l'utilisation des plans comme amers permanents. La pose du robot est estimée en minimisant la distance point-à-plan entre les mesures laser et les surfaces planes modélisées. Une optimisation globale, appelée Global Plane Adjustment (GPA), est utilisée pour affiner conjointement les paramètres de ces plans et les poses du robot. Cette méthode permet de corriger la dérive dès qu'une surface plane connue est identifiée, établissant des contraintes spatiales fortes même si le robot ne revient pas exactement à une position géographique déjà visitée.

5.2.4) Fusion multi-capteurs par filtre de Kalman

Pour la conduite autonome en milieu urbain, la localisation est obtenue par la synchronisation de plusieurs sources de données via un filtre ESKF (Error State Kalman Filter). Ce filtre fusionne les prédictions de mouvement issues de l'IMU et des capteurs de vitesse du véhicule avec les corrections de pose fournies par le LiDAR et le GNSS-RTK. L'alignement LiDAR est effectué par l'algorithme NDT (Normal Distributions Transform), qui modélise le nuage de points par des distributions normales pour accroître la tolérance au bruit. Un descripteur global, le Scan Context, permet d'identifier les lieux déjà visités pour valider la position et éliminer la dérive

5.2.5) Localisation par primitives géométriques pour capteur Solid-State

L'utilisation de capteurs LiDAR à état solide, caractérisés par un champ de vision étroit, impose une localisation basée sur l'extraction de caractéristiques stables. Le système analyse la courbure locale des points laser pour isoler les bords (coins de murs, cadres) et les surfaces planes. La pose est estimée en calculant les distances entre ces primitives extraites en temps réel et leurs correspondants dans une carte de probabilité intégrée à une OctoMap. Cette méthode permet une localisation centimétrique à haute fréquence en se focalisant uniquement sur les repères géométriques les plus fiables de l'environnement.

Tableau récapitulatif des méthodes et algorithmes utilisés dans les différents articles :

| Algorithmes et méthodes utilisées identifiées | Autonomous Full 3D Coverage Using an Aerial Vehicle, Performing Localization, Path Planning, and Navigation towards Indoors Inventorying for the Logistics Domain | Direct LiDAR Odometry: Fast Localization With Dense Point Clouds | LiDAR SLAM With Plane Adjustment for Indoor Environment | LiDAR-Based Sensor Fusion SLAM and Localization for Autonomous Driving Vehicles in Complex Scenarios | Lightweight 3-D Localization and Mapping for Solid-State LiDAR |
|------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|---------------------------------------------------------|------------------------------------------------------------------------------------------------------|----------------------------------------------------------------|
| Localisation / Odométrie | MCL | Direct LiDAR Odometry | Enregistrement global en temps réel | ESKF | Lightweight LiDAR |
| Représentation de l'espace / Cartographie | | NanoGICP | Point-to-Plane | NDT | Feature extraction |
| SLAM / Cartographie | OctoMap | Cartographie basée sur les Keyframes | Plane Adjustment | Pose Graph & Point Clouds | Probability Map |
| Optimisation | TSP | Convex Hull | Global Plane Adjustment | Pose Graph | |

5.3) Planification de trajectoire 3D

5.3.1) Familles d'algorithmes identifiés

5.3.1.1) Algorithmes basés sur l'Échantillonnage (Sampling-Based)

Cette approche explore l'environnement en connectant des points aléatoires, ce qui évite de devoir cartographier l'intégralité de l'espace libre.

- Algorithmes cités :
 - **RRT (Rapidly-exploring Random Tree)** : L'algorithme forme un arbre depuis le point de départ en tirant des points aléatoires dans l'espace. Si un point est dans une zone libre, il est connecté à la branche la plus proche. Très efficace pour explorer rapidement de grands espaces vides. (**Environnement Statique/Dynamique**)
 - **RRT*** : Une amélioration du RRT. L'algorithme cherche le chemin le plus court. (**Statique**)
 - **PRM (Probabilistic RoadMap)** : Construit un graphe en lançant des milliers de points aléatoires dans l'espace libre et en les reliant s'ils sont proches. La recherche de chemin se fait ensuite sur ce graphe fixe. (**Statique**)

5.3.1.2) Algorithmes de Recherche Discrète / Graph-Based

Ces méthodes divisent l'environnement en cellules pour former une grille.

- Algorithmes cités :
 - **A* (A-Star)** : L'algorithme le plus utilisé selon les papiers. Il explore la grille en privilégiant les nœuds les plus proches de la cible (grâce à une heuristique). Il garantit le chemin le plus court tout en explorant moins de nœuds que Dijkstra. (**Statique**)
 - **Dijkstra** : Explore uniformément dans toutes les directions. Garantit le plus court chemin mais est plus lent que A*. (**Statique**)
 - **D* (Dynamic A*)** : D* est capable de trouver un autre chemin si un nouvel obstacle apparaît. (**Dynamique**)
 - **JPS (Jump Point Search)** : Une optimisation de A* pour les grilles uniformes qui "saute" les nœuds inutiles (ceux représentant généralement les lignes droites) pour accélérer le calcul. (**Statique**)

5.3.1.3) Algorithmes Méta-heuristiques (Bio-inspirés)

Inspirés par la nature, ces algorithmes gèrent des problèmes complexes avec de multiples objectifs (batterie, sécurité, longueur) sans garantir que le chemin trouvé soit optimal.

- Algorithmes cités :
 - **PSO (Particle Swarm Optimization)** : Simule un essaim d'oiseaux où chaque particule ajuste sa trajectoire selon sa propre expérience et celle du groupe. (**Statique**)
 - **GA (Genetic Algorithm)** : Utilise la sélection naturelle (mutation, croisement) sur une population de trajectoires que l'on génère aléatoirement pour faire évoluer la meilleure solution. (**Statique**)

- **ACO (Ant Colony Optimization)** : Imité les fourmis déposant des phéromones pour marquer le chemin le plus court (celui avec le plus de phéromones). **(Statique)**
- **GWO (Grey Wolf Optimizer)** : Imité la hiérarchie sociale et la technique de chasse des loups gris pour encercler la solution (la cible). **(Statique)**

5.3.1.4) Algorithmes d'Optimisation Mathématique

Utilisés principalement pour le lissage de trajectoire. Ils agissent sur un chemin déjà trouvé mais en l'améliorant.

- Algorithmes cités :
 - **Minimum Snap** : Génère une courbe polynomiale qui minimise les changements brusques de moteur, rendant le vol très fluide. **(Statique)**
 - **SFC (Safe Flight Corridors)** : Construit une suite de volumes géométriques (tunnels) libres d'obstacles dans lesquels le drone peut évoluer en sécurité. **(Statique)**
 - **Programmation Quadratique (QP)** : Consiste à minimiser une fonction de coût quadratique. Permet de minimiser l'effort et facilite les trajectoires par waypoints **(Statique)**
 - **Optimisation Convexe (SOCP)** : Elle permet de lisser des courbes de haut degré (des courbes de Bézier) **(Statique)**

5.3.1.5) Algorithmes Réactifs (Champs de Potentiel)

- Algorithmes cités :
 - **APF (Artificial Potential Fields)** : L'environnement est vu comme un champ magnétique. La cible attire le drone (force positive), et les obstacles le repoussent (force négative). Si le drone se rapproche d'un obstacle situé sur sa droite, il va aller vers la gauche. **(Dynamique car basé sur la lecture des capteurs)**

5.3.1.6) Algorithmes Hybrides

L'objectif est de combiner deux algorithmes pour compenser les faiblesses de l'un par les forces de l'autre.

- Algorithmes cités :
 - **PSO-GA** : Combine la vitesse de l'essaim pour foncer vers la meilleure solution connue, et les mutations aléatoires de la génétique pour s'échapper si le drone se retrouve piégé dans un cul-de-sac.
 - **PSO-HSA (Harmony Search)** : Combine PSO avec la recherche harmonique pour équilibrer parfaitement l'exploration (chercher ailleurs) et l'exploitation (affiner le chemin actuel).
 - **RRT/A*/JPS + Opt** : Combine un algorithme de planification (RRT/A*/JPS) qui génère un chemin approximatif ou plus généralement des SFC qui seront ensuite optimisés par un des algorithmes d'optimisation mathématique pour obtenir une trajectoire lisse.

5.3.1.7) Algorithmes d'Apprentissage (Learning-Based)

Un agent (réseau de neurones) apprend à naviguer par essais et erreurs (Récompense/Punition) via des millions de simulations.

- Algorithmes cités :
 - **RL** (Reinforcement Learning),
 - **DRL** (Deep Reinforcement Learning).

5.3.1.8) Tableau donnant les avantages et inconvénients de chaque famille d'algorithmes

| Famille d'Algorithmes | Algorithmes Cités | Type de Map | Avantages (par famille) | Inconvénients (par famille) |
|------------------------------------|----------------------------|-------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| Basés sur l'Échantillonnage | RRT RRT* PRM | Nuage de points Continuous Configuration Space | Rapide dans les espaces complexes et 3D Ne nécessite pas de cartographier tout l'espace vide Probabilistiquement complet (finit par trouver un chemin) | Trajectoires saccadées et aléatoires (nécessite du lissage) Non-déterministe Difficulté à passer par des passages étroits |

| | | | | |
|----------------------------------|------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| Basés sur les Graphes | <p>A* (A-Star)</p> <p>Dijkstra</p> <p>D*</p> <p>JPS</p> | <p>Voxel Map / Grid</p> <p>Octomap</p> <p>Occupancy Grid</p> | <p>Optimalité garantie (A* trouve le plus court chemin mathématique)</p> <p>Complet et stable (résultat toujours identique)</p> | <p>Coût mémoire conséquent en 3D</p> <p>Lent sur les très grands espaces</p> <p>Trajectoires anguleuses contraintes par la grille</p> |
| Optimisation Mathématique | <p>Minimum Snap</p> <p>QP (Quadratic Programming)</p> <p>SFC</p> <p>SOCP</p> | <p>Safe Flight Corridors (Tunnels)</p> <p>Convex Hulls (Polytopes)</p> <p>ESDF (Euclidean Signed Distance Fields)</p> | <p>Produit des trajectoires lisses et continues (courbes parfaites)</p> <p>Prend en compte la vitesse du drone)</p> <p>Idéal en combinaison avec un autre algorithme pour une approche hybride</p> | <p>Risque de blocage dans des culs-de-sac</p> <p>Nécessite un chemin initial valide pour démarrer</p> <p>Calcul matriciel lourd pour le processeur</p> |
| Méta-heuristiques | <p>PSO</p> <p>GA</p> <p>ACO</p> <p>GWO</p> | <p>Fonctions de coût</p> <p>DEM (Digital Elevation Model)</p> <p>Obstacles géométriques</p> | <p>Gère l'optimisation multi-objectifs (batterie + sécurité + temps)</p> <p>Robuste face aux environnements complexes.</p> <p>Facilement parallélisable</p> | <p>Demande beaucoup d'itérations</p> <p>Pas de garantie d'optimalité parfaite.</p> <p>Réglage des hyper-paramètres délicat</p> |

| | | | | |
|----------------------|----------------------------------------------------------------------------|------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| Hybrides | PSO-GA PSO-HSA RRT + Opt. A* + Opt. JPS + Opt. | Dépend de la combinaison utilisé mais en | Meilleure performance globale grâce à au combinaison Trajectoire optimale adaptée aux capacités du drone et permet à ce dernier d'aller potentiellement plus vite | Complexité de code élevée Demande beaucoup de ressources (CPU) pour faire tourner les algorithmes conjoints |
| Réactifs | APF | Potential Field Map Sans Map | Temps réel (très léger en calcul) Réactivité immédiate pour l'évitement d'urgence | Problème critique de Minima Locaux (le drone se piège dans les cul-de-sacs) Oscillations possibles près des obstacles. |
| Apprentissage | RL / DRL Neural Networks | Données capteurs Sans carte | Adaptabilité aux environnements inconnus Gère des situations non modélisables Exécution rapide (Inférence) | Décisions prises par l'IA inexpliquées Phase d'entraînement très longue Difficile de passer de la simulation au réel |

5.3.2) Tableau récapitulatif des familles d’algorithme utilisés dans les différents articles :

| Famille d’algorithme identifiée | Sampling-based | Méta-heuristique | Graph-based | Algorithmes Réactifs | Optimization-based | Algorithmes Hybrides |
|-------------------------------------------------------------------------------------------------------------------------|----------------|--------------------------|---------------|----------------------|----------------------------|----------------------|
| Real-Time Safe Trajectory Generation for Quadrotor Flight in Cluttered Environments | | | A* | | Quadratic Programming (QP) | oui |
| Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments | RRT* | | | | SOCP | oui |
| Plan3D: Viewpoint and Trajectory Optimization for Aerial Multi-View | RRT * | | | | Recursive Greedy Algorithm | oui |
| Search-based Motion Planning for Quadrotors using Linear Quadratic Minimum Time Control | | | A* | | QP | oui |
| Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments | RRT* | | | | QP | oui |
| Planning Dynamically Feasible Trajectories for Quadrotors using Safe Flight Corridors in 3-D Complex Environments | | | JPS | | QP | oui |
| A survey of 3D Space Path-Planning Methods and Algorithms | | PSO ACO, GWO, GA, GSA... | | | | oui |
| Hybrid PSO-HSA and PSO-GA algorithm for 3D path planning in autonomous UAVs | | PSO et GA | | | | oui avec HSA |
| Metaheuristic Algorithms in | RRT, PRM. | ACO, PSO | Dijkstra, A*, | | | oui |

| | | | | | | |
|-----------------------------------------------------------------------|-------------|----------------------------------------------------|-----------------------------|-----|----------------|-----|
| UAV Path-Planning Optimization: A Systematic Review (2018–2022) | | | D* | | | |
| Path Planning for Autonomous Drones: Challenges and Future Directions | PRM,RRT,RT* | GA, PSO, ACO, Memetic Algorithms , Neural Networks | Dijkstra , A* , D* , Theta* | APF | LP, MILP , NLP | oui |

Références

1. [Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments](#), Fei Gao, William Wu, Wenliang Gao, S. Shen, J. Field Robotics, 2018
2. [Plan3D: Viewpoint and Trajectory Optimization for Aerial Multi-View](#), Benjamin Hepp, M. Nießner, Otmar Hilliges, ACM Transactions on Graphics, 2017
3. [Search-based motion planning for quadrotors using linear quadratic minimum time control](#), Sikang Liu, Nikolay A. Atanasov, K. Mohta, Vijay R. Kumar, IEEE/RJS International Conference on Intelligent Robots and Systems, 2017
4. [Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments](#), Charles Richter, A. Bry, N. Roy, International Symposium of Robotics Research, 2016
5. [Real-Time Safe Trajectory Generation for Quadrotor Flight in Cluttered Environments](#), Jing Chen, Kunyue Su, and Shaojie Shen, IEEE Conference on Robotics and Biomimetics, Zhuhai, China, 2015
6. [Planning Dynamically Feasible Trajectories for Quadrotors using Safe Flight Corridors in 3-D Complex Environments](#), Sikang Liu, Michael Watterson, Kartik Mohta, Ke Sun, Subhrajit Bhattacharya, Camillo J. Taylor and Vijay Kumar, IEEE ROBOTICS AND AUTOMATION LETTERS, 2017

7. [A survey of 3D Space Path-Planning Methods and Algorithms](#), Hakimeh Mazaheri, Salman Goli, Ali Nouroollah, ACM Computing Surveys, 2024
8. [Hybrid PSO-HSA and PSO-GA algorithm for 3D path planning in autonomous UAVs](#), B. Abhishek, S. Ranjit, T. Shankar, Geoffrey Eappen, P. Sivasankar & A. Rajesh, SN Applied Sciences, 2020
9. [Metaheuristic Algorithms in UAV Path-Planning Optimization: A Systematic Review \(2018–2022\)](#), Maral Hooshyar, Yueh-Min Huang, Drones 2023
10. [Path Planning for Autonomous Drones: Challenges and Future Directions](#), Gopi Gogan and Anwar Haque, Drones, 2023
11. [Lightweight 3-D Localization and Mapping for Solid-State LiDAR](#), Han Wang, Chen Wang, Lihua Xie, IEEE Robotics and Automation Letters, 2021
12. [Autonomous Full 3D Coverage Using an Aerial Vehicle](#), Performing Localization, Path Planning, and Navigation towards Indoors Inventorying for the Logistics Domain, Kosmas Tsiakas, Emmanouil Tsardoulis and Andreas L. Symeonidis, Robotics, 2024
13. [LiDAR-Based Sensor Fusion SLAM and Localization for Autonomous Driving Vehicles in Complex Scenarios](#), Kai Dai, Bohua Sun, Guanpu Wu, Shuai Zhao, Fangwu Ma, Yufei Zhang and Jian Wu, J. Imaging, 2023
14. [LiDAR SLAM With Plane Adjustment for Indoor Environment](#), Lipu Zhou; Daniel Koppel; Michael Kaess, IEEE Robotics and Automation, 2021
15. [Direct LiDAR Odometry: Fast Localization with Dense Point Clouds](#), Kenny Chen, Brett T. Lopez, Ali-akbar Agha-mohammadi, Ankur Mehta, IEEE Robotics and Automation Letters, 2021
16. [A 3D continuous spatio-temporal mapping method](#). Cong, Y., Chen, C., Yang, B., Li, J., Wu, W., Li, Y., & Yang, Y. 3D-CSTM: ISPRS Journal of Photogrammetry and Remote Sensing, 186, 232–245, 2022
17. [Information-Efficient 3-D Visual SLAM for Unstructured Domains](#). Weizhen Zhou, Miro, J. V., & Dissanayake, G. . IEEE Transactions on Robotics, 24(5), 1078–1087., 2008
18. [3D SLAM using planar segments](#). Weingarten, J., & Siegwart, R. 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006
19. [A robust RGB-D SLAM system for 3D environment with planar surfaces](#). Su, P.-C., Shen, J., & Cheung, S. IEEE International Conference on Image Processing, 2013
20. [3D reconstruction of indoor scenes by casting visual rays in an occupancy grid](#). Shaker, S., Asmar, D. C., & Elhajj, I.. IEEE International Conference on Robotics and Biomimetics, 2010
21. [HMAPs - Hybrid Height- Voxel Maps for Environment Representation](#), Garrote, L., Premebida, C., Silva, D., & Nunes, U. J.. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 1197–1203, 2018
22. [LiDAR-Based Sensor Fusion SLAM and Localization for Autonomous Driving Vehicles in Complex Scenarios](#), Kai Dai, Bohua Sun, Guanpu Wu, Shuai Zhao, Fangwu Ma, Yufei Zhang, Jian Wu, Journal of Imaging, 2023
23. [Lightweight 3-D Localization and Mapping for Solid-State LiDAR](#), Han Wang, hen Wang, Lihua Xie, IEEE Robotics and Automation Letters, 2021

