Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

A review of UAV autonomous navigation in GPS-denied environments

Yingxiu Chang[a], Yongqiang Cheng[b,*], Umar Manzoor[b], John Murray[b]

[a] School of Computer Science, University of Hull, Cottingham Road, Hull, HU6 7RX, UK
[b] Faculty of Technology, University of Sunderland, St. Peter's Campus, Sunderland, SR6 0DD, UK

ARTICLE INFO

Keywords:
UAV
GPS-denied navigation
Self-exploration
Performance evaluation metrics

ABSTRACT

Unmanned aerial vehicles (UAVs) have drawn increased research interest in recent years, leading to a vast number of applications, such as, terrain exploration, disaster assistance and industrial inspection. Unlike UAV navigation in outdoor environments that rely on GPS (Global Positioning System) for localization, indoor navigation cannot rely on GPS due to the poor quality or lack of signal. Although some reviewing papers particularly summarized indoor navigation strategies (e.g., Visual-based Navigation) or their specific sub-components (e.g., localization and path planning) in detail, there still lacks a comprehensive survey for the complete navigation strategies that cover different technologies. This paper proposes a taxonomy which firstly classifies the navigation strategies into Mapless and Map-based ones based on map usage and then, respectively categorizes the Mapless navigation into Integrated, Direct and Indirect approaches via common characteristics. The Map-based navigation is then split into Known Map/Spaces and Map-building via prior knowledge. In order to analyze these navigation strategies, this paper uses three evaluation metrics (Path Length, Deviation Rate and Exploration Efficiency) according to the common purposes of navigation to show how well they can perform. Furthermore, three representative strategies were selected and 120 flying experiments conducted in two reality-like simulated indoor environments to show their performances against the evaluation metrics proposed in this paper, i.e., the ratio of Successful Flight, the Mean time of Successful Flight, the Mean Length of Successful Flight, the Mean time of Flight, and the Mean Length of Flight. In comparison to the CNN-based Supervised Learning (directly maps visual observations to UAV controls) and the Frontier-based navigation (necessitates continuous global map generation), the experiments show that the CNN-based Distance Estimation for navigation trades off the ratio of Successful Flight and the required time and path length. Moreover, this paper identifies the current challenges and opportunities which will drive UAV navigation research in GPS-denied environments.

1. Introduction

The portability, maneuverability and flexibility of UAVs provide potential flying capabilities in complex 3D environments which may contain unexpected obstacles. With the development of high-precision sensors such as GPS, cameras and Light Detection and Ranging (LiDAR) sensor, UAVs have abilities for localization, object detection and obstacle avoidance and can be deployed in many diverse applications such as agriculture [1], traffic monitoring [2] and facilities inspection [3] as shown in Fig. 1.

Many UAV applications require complete autonomous navigation, which consists of multiple sub-components without human intervention, which need to consider: (a) Where am I going? (b) Where am I? and (c) How do I get there? [4] For example, Gyagenda et al. [5] proposed a comprehensive bottom-up review to access multiple sub-components of perception, localization and motion planning. They indicated that only 16% of the research presented complete navigation strategy while 62% of sub-component research focuses on UAV localization. According to their survey, the potential future research is expected to employ more advanced sensors such as signal-of-opportunity sensors, event cameras and miniaturized radar combining with advanced algorithms like adaptive AI techniques to improve the perception capability. In addition, localization technologies with higher attentions (62% of sub-component research) require high-fidelity simulators to reduce the risks of collision due to the localization errors.

Several review papers focus on vision-based navigation such as Lu et al. [6] divided the strategies into mapless, map-building and map-based and further introduced the sub-components such as localization and mapping, obstacle avoidance and path planning. Arafat et al. [7] investigates more UAV vision-based navigation system which covers

* Corresponding author.
E-mail addresses: Y.Chang-2020@hull.ac.uk (Y. Chang), Yongqiang.Cheng@sunderland.ac.uk (Y. Cheng), umar.manzoor@sunderland.ac.uk (U. Manzoor), john.murray@sunderland.ac.uk (J. Murray).

https://doi.org/10.1016/j.robot.2023.104533
Received 24 September 2022; Received in revised form 18 August 2023; Accepted 21 September 2023
Available online 29 September 2023

Table 1
Summary of proposed surveys.

Ref.	Focuses	Contributions
[5]	Complete GPS-denied navigation	Comprehensively reviewed the complete navigation strategies with multiple sub-components of perception, localization and motion planning. Concluded the future research of combining advanced sensors with AI-based technologies, and high-fidelity simulators.
[6,7]	Vision-based navigation	[6] Concluded the challenges of: (1) The hardware constraints of power and storage consumption, and computational complexity for 3D navigation; (2) Poor synchronization of multi-sensor fusion raised by different sensors' noise features. [7] Summarized four major issues to be considered when designing the navigation system, which are the scalability, computational power, reliability and robustness.
[8]	Vision-IMU fusion navigation	Summarized the Visual Odometry based on Visual-IMU fusion achieved less memory and power requirements comparing to SLAM, and also proposed a modular multi-sensor fusion approach for better pose estimation.
[9]	OF-based navigation	Summarized the navigation tasks of using OF approaches (e.g., distance and velocity estimations and vertical landing) and indicated the challenges of lack of quantitative evaluation and real-time processing of OF approaches.
[10]	AI-based navigation	Comparing the AI-based navigation technologies based on a uniform criteria including features, time-complexity and parameter number. Indicated the future research in developing big data processing, computing power, energy efficiency and fault handling.
[11,12]	3D path planning	[11] comprehensively reviewed and compared path planning algorithms based on time complexity, environments and real-time. [12] compared path planning algorithms based on a uniform evaluation criteria including complexity, adaptiveness, environments, fault tolerance and etc.
[13]	Conventional path planning	Analyzed path planning algorithms by quantitatively comparing the results of Computational Time and Error in the same environment.
[14]	Computational intelligence path planning	Summarized that the current research trend of path planning is online processing in 3D environments.
[15]	Bio-inspired path planning	Focused on the bio-inspired algorithm and compared each other based on the energy efficiency, communication delay, environmental complexity and computational burden.
[16]	Outdoor localization	Compared the positioning methods based on trajectory length, environmental size and localization error, also indicated the potential improvements of proposing public datasets, source codes and hardware accelerators.
[17]	Indoor localization	Compared the performances between different sensors (e.g., camera, IMU, WiFi, etc.) by using positioning accuracy and indicated their pros and cons.

(a) Agriculture [1] (b) Traffic [2] (c) Facilities [3]

Fig. 1. Example of tasks.

from indoor to outdoor environments and analyzes the advantages and limitations of the sub-components of collision avoidance and path planning. Balamurugan et al. [8] proposed another vision-based navigation survey and revealed that most works which fuse monocular/stereo cameras with Inertial Measurement Unit (IMU) for pose estimation are faster, demanding less memory and power requirements compare to Simultaneous Localization and Mapping (SLAM). Optical flow (OF) as one of the vision technologies was used for autonomous navigation. Chao et al. [9] reviewed the OF-based navigation methods and indicated its challenges of quantitation evaluation, real-time high-resolution image processing and multi-vision system.

Current research trends in autonomous UAV navigation points towards the growing use of Artificial Intelligence (AI) technologies, which are reviewed by Rezwan and Choi [10]. They classify the AI-based technologies into optimized-based approaches (e.g., PCO, ACO, GA) and learning-based approaches (e.g., RL, DRL and DL) and analyze by comparing the features, time complexity and parameter number. Future research can be summarized as the improvement and novel ideas in big data processing, computing power, energy efficiency and fault handling.

Instead of focusing on the complete navigation strategy, other surveys have paid more attention to sub-components such as Path Planning, Obstacle Avoidance and Localization. Path planning and obstacle avoidance have been surveyed by [11–15] with a different focus and contributions. Yang et al. [11] classified the 3D path planning methods into five categories according to their features and tried to find the optimal method by analyzing the time complexity, available environments (static/dynamic) and real-time applicability for each category. Amarat and Zong [12] proposed a uniform performance evaluation metric to answer the questions of the best to use and perspective areas for operation. Radmanesh et al. [13] summarized the features of conventional algorithms and compared the results of timing costs and deviations with the optimal solutions in simulated environments. Zhao et al. [14] concluded the real-time applicability in available environments (2D/3D) for 231 Computational Intelligence-based path planning methods and indicated the potential trends with a statistical methodology. Poudel et al. [15] classified the Bio-inspired path planning algorithms into the groups: swarm intelligence, evolutionary algorithm, behavior-based and bio-inspired neural network, and further compared with each other in terms of their specific features, advantages and limitations.

As another sub-component of autonomous navigation, localization was also studied based on different sensors. For example, Couturier and Akhloufi [16] reviewed absolute vision-based localization for applications in outdoor environments and compared the strategies according to popular methods, practical environments and performances to draw a conclusion for future improvements. As a result of GPS signal attenuation in indoor environments, Perez Rubio et al. [17] sorted strategies based on available sensors including camera, IMU, infrared, radio transmitters/receivers, ultrasonic and others in addition to briefly discussing their advantages/drawbacks.

To intuitively illustrate the differences of the aforementioned surveys, Table 1 shows their primary focuses and contributions.

Although the above surveys have investigated UAV navigation strategies, path planning and localization, they still have limitations as follows: (1) Lack of a comprehensive survey for the complete navigation strategies which include the specific and possibly independent sub-components of navigation systems covering aspects from data capturing

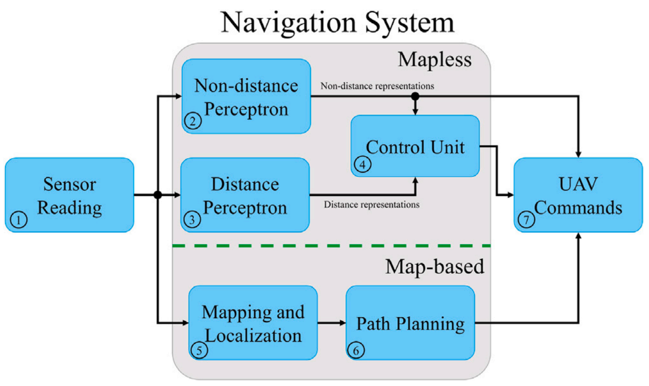


Fig. 2. Overall process of the complete navigation strategy considered in this paper. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

to the control policy or path planning. (2) The limited evaluation does not allow direct study on how well the strategies can perform. (3) Do not have verification in reality-like environments which lack an effective data comparison to draw the potential improvements for future works. Therefore, contributions in this paper are as follows:

- According to the summarized overall process of the complete navigation strategy shown in Fig. 2, this work firstly proposes a taxonomy, see Fig. 3, to review and classify the complete UAV navigation strategies in GPS-denied environments according to the map usage, common characteristics and prior knowledge. Then survey their corresponding sub-components based on the diagrams of Figs. 4 and 13.
- Analyzing relevant technologies of complete navigation strategies in detail according to the processing diagrams shown in Figs. 4 and 13 as well as proposing a new set of evaluation metrics for theoretical performance evaluation.
- Quantitatively analyzing three representative navigation strategies which respectively conducted 120 flying experiments in reality-like environments based on Microsoft AirSim [18] for simulated evaluation.

The rest of the paper is organized as follows. Section 2 introduces the summarized process of navigation, taxonomy and the definitions of terminologies used in this paper. Sections 3 and 4 focus on the detailed study of the existing complete Mapless and Map-based navigation strategies along with their technologies. Section 5 provides performance evaluations which include a theoretical evaluation based on the results presented by the literature and a simulated evaluation to verify the representative strategies in two reality-like environments. Section 6 identifies the current challenges and opportunities while Section 7 concludes the paper.

2. Process of navigation, taxonomy and definition

In recent decades, various strategies for UAV autonomous navigation have been proposed. In this review, the focus is drawn to the complete UAV navigation strategies which include the whole process (i.e., covering all sub-components) from data capturing to the control policy or path planning [19]. The overall navigation strategy with the sub-components concerned in this paper is shown in Fig. 2.

In Fig. 2, the blue boxes represent the sub-components of the overall process and are numbered as ①-⑦. The gray box has grouped the major sub-components. The taxonomy firstly considers the map usage which classifies the navigation strategies into Mapless and Map-based at the first level since their system architectures and control policies are quite different. The following will illustrate the basis and terminology definitions of the Mapless and Map-based categories.

Mapless category Mapless Navigation, as shown above the green dash line in Fig. 2, usually does not need any prior knowledge of the environment but rather perceive the environment as they navigate it without creating any maps during navigation [20,21]. The ② Non-distance Perceptron processes sensor data to obtain different Non-distance representations while the ③ Distance Perceptron obtains the Distance representations from sensor data. The ④ Control Unit is able to process Non-distance representations (i.e., from ② Non-distance Perceptron) or Distance representations (i.e., from ③ Distance Perceptron) and outputs the ⑦ UAV Commands (i.e., $[V_x, V_y, V_z, roll, pitch, yaw]$).

The workflows of Mapless Navigation are ①-②-⑦, ①-②-④-⑦ and ①-③-④-⑦ while the first one is defined as Integrated Approaches and the rest of them are Non-integrated Approaches. The reason is that ①-②-⑦ integrates the Control Unit within the entire algorithm while the others require an independent ④ Control Unit to get the UAV Commands. Moreover, the Non-integrated approaches can be further classified as ①-②-④-⑦ Direct Approaches and ①-③-④-⑦ Indirect Approaches depending on the ④ Control Unit calculates the direct Non-distance or indirect Distance representations. We place the Integrated, Direct and Indirect Approaches at the same level for better understanding and will respectively explain their features in detail as below.

- ①-②-⑦ Integrated Approaches: There is no independent ④ Control Unit in these approaches, that is, the Control Unit is integrated within the entire algorithm. The Non-distance representations of these approaches are less correlated with environments but directly corresponded to the UAV control. For example, a Convolutional Neural Network acts as an encoder which extracts the sensor data and directly outputs UAV control commands (e.g., steering and speed).
- ①-②-④-⑦ Direct Approaches: In comparison with Integrated Approaches, the Non-distance representations of these approaches are more relevant with environments but less correspond to the UAV control such as relative positions in corridors and OF differences. Therefore, these approaches require an independent ④ Control Unit as the Decision Tree to directly map the environmental representations to the UAV Commands.
- ①-③-④-⑦ Indirect Approaches: In order to achieve precise control, the Distance representations of these approaches are usually extracted from ③ Distance Perceptron and formulated as a distance matrix. The ④ Control Unit summarizes these large amounts of distance information and outputs the UAV Commands. Therefore, the strategies using Distance representations (e.g., CNN-based, OF-based and Reflection-based Distance Estimation) are classified as Indirect Approaches.

Map-based category Map-based Navigation, below the green dash line in Fig. 2, requires the metrical or topological representations of environments [21] and consists of Obstacle Mapping, UAV Self-localization and Path Planning sub-components which are independent to each other. Concerning whether prior knowledge is provided or not, we classify strategies as Known Map/Spaces and Map-building [22] where the former provides prior environmental information such as specific landmarks and accessing points while the latter requires constructing 2D/3D maps simultaneously during navigation (i.e., SLAM-based navigation) in the unknown spaces.

- Known Map/Spaces: The strategies have prior spatial details about the environments such as the positions of access points, start and destinations, obstacles or even the entire maps before navigation which is followed by UAV Self-localization and Path Planning.
- Map-building: Counting on Leonard and Whyte [23], it can be seen as SLAM-based navigation which was defined as simultaneous map building and localization during navigation without any prior information.

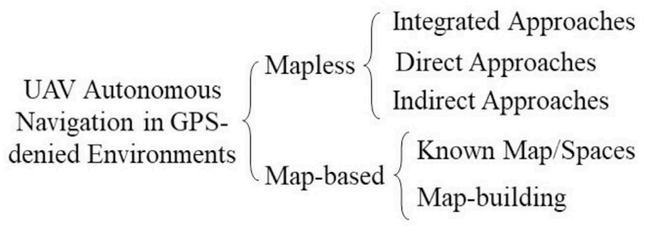


Fig. 3. Taxonomy of the UAV complete navigation strategy in GPS-denied environments.

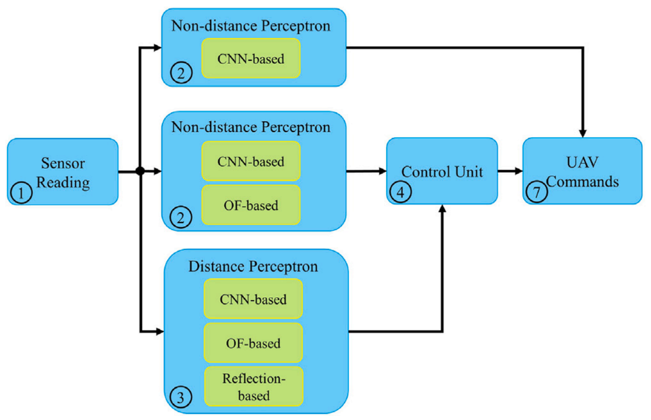

Sensor Reading ①
Non-distance Perceptron CNN-based ②
Non-distance Perceptron CNN-based OF-based ②
Distance Perceptron CNN-based OF-based Reflection-based ③
Control Unit ④
UAV Commands ⑦

Fig. 4. Mapless navigation.

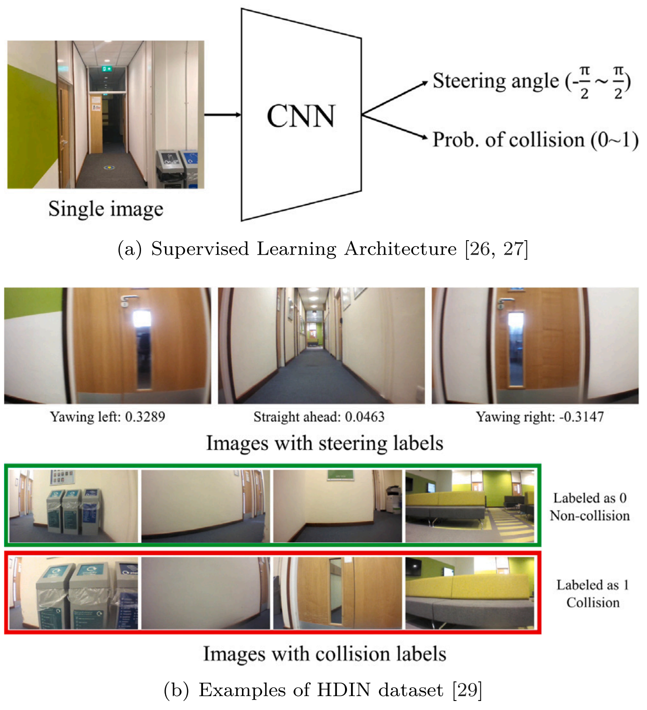

Single image
CNN
Steering angle ($-\frac{\pi}{2} \sim \frac{\pi}{2}$)
Prob. of collision (0~1)
(a) Supervised Learning Architecture [26, 27]
Yawing left: 0.3289
Straight ahead: 0.0463
Yawing right: -0.3147
Images with steering labels
Labeled as 0 Non-collision
Labeled as 1 Collision
Images with collision labels
(b) Examples of HDIN dataset [29]

Fig. 5. Examples of supervised learning. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

According to the aforementioned illustrations in Section 2, we summarize a taxonomy of the UAV complete navigation strategy in GPS-denied environments as Fig. 3 shows.

3. Mapless navigation

According to the Overall Process in Fig. 2 and the taxonomy in Fig. 3, we further show the specific technologies of Integrated, Direct and Indirect Approaches as the processing schematic diagram in Fig. 4. The workflows of ①-②-⑦ Integrated Approaches, ①-②-④-⑦ Direct Approaches and ①-③-④-⑦ Indirect Approaches in Fig. 4 are corresponded to Fig. 2. It also shows the interior technologies of ② Non-distance and ③ Distance Perceptron as yellow boxes in detail.

The rest of these Section will respectively survey the Integrated, Direct and Indirect Approaches of Mapless navigation strategies as well as their technologies.

3.1. Integrated approaches

Based on the definition in Section 2 and processing diagram in Fig. 4, the Integrated Approaches directly map the input ① Sensor Reading to the ⑦ UAV Commands without an independent ④ Control Unit. Currently, Deep learning algorithms, especially the CNN-based Supervised Learning and Reinforcement Learning such as ② Non-distance Perceptron for visual-based navigation show remarkable capabilities in this direct mapping [24].

Supervised learning When a large dataset along with sufficient labels is provided, supervised learning maps the input images into the UAV commands, shown in Fig. 5(a). For example, one of the residual convolutional models [25] was used by Loquercio et al. [26] named DroNet which directly predicts the steering angle in the range of $[-\frac{\pi}{2}, \frac{\pi}{2}]$ and the probability of collision $[0, 1]$ from the input continuous gray images. A low-pass filter was used to smooth the steering angle and the probability of collision was transformed into velocity. Further research carried out by Palossi et al. [27] applied DroNet to a nano-UAV equipped with a PULP-GAP8 embedded processor for unknown indoor navigation. For the purpose of improving power efficiency on a limited resource device, Palossi et al. [27] not only minimized the pooling size and replaced the float-point computation with fixed-point computation, but also added batch normalization in a convolutional layer to keep the same original accuracy. The extensive evaluations of [26,27] illustrate that their DroNet networks have competitive generalization capability for unknown environments but are sensitive to 'line-like' patterns (i.e., focus on car lanes) since their networks were trained based on the Udacity's self-driving car dataset [28].

We also proposed a HDIN dataset [29] shown as Fig. 5(b) based on Integrated Approaches. The HDIN dataset contains steering images which their labels are normalized into the range of [−1, 1] (shown as 'Images with steering labels' in Fig. 5(b)). The HDIN dataset also separates collision images into 'Non-collision' and 'Collision' classes depending on whether the distance to collision is smaller than 50 cm. The Non-collision and Collision images are respectively shown in the green and red box, and respectively labeled as 0 and 1 to represent the probabilities of collision. The residual convolutional model [25] is able to simultaneously predict the steering angle and the probability of collision for UAV indoor navigation.

Reinforcement learning Supervised learning requires a large number of images with relevant and accurate labels for training which poses a great challenge for annotations. To solve this problem, self-supervised learning methods such as Reinforcement Learning have arisen which does not require any labels but interacts with environments to obtain rewards as Fig. 6 shows. AlMahamid and Grolinger [30] reviewed the Reinforcement Learning methods used for UAV navigation and indicated that the UAV's commands is one of the basis for classification.

Specifically, the discrete commands such as the preset directions: forward, left, right, up, and down were used by Abhik Singla et al. [31], who proposed a recent reinforcement learning model by combining a conditional generative adversarial network (cGAN) [32], a deep Q-learning network and a recurrent network [33] to directly output UAV commands (i.e., 'Go straight', 'Turn left' and 'Turn right'). It firstly

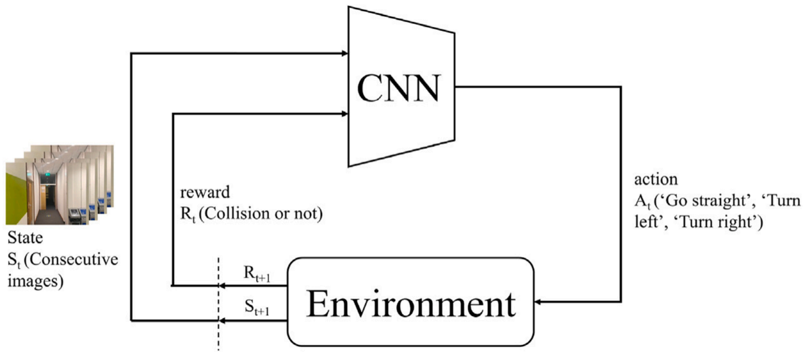

CNN
reward
R_t(Collision or not)
action
A_t('Go straight', 'Turn left', 'Turn right')
State
S_t(Consecutive images)
R_{t+1}
S_{t+1}
Environment

Fig. 6. Reinforcement learning architecture [31].

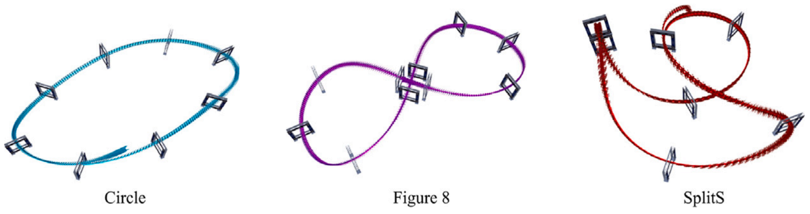
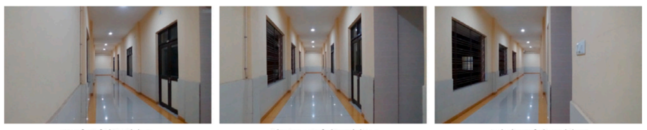


Fig. 7. Teacher–student vision-based policy trajectories [39].

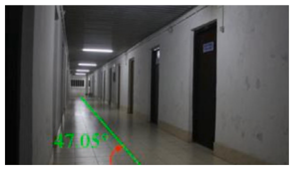
applies the cGAN to generate the depth maps from RGB images and then, feeds l generated depth maps simultaneously into l streams deep Q-learning network, finally followed by a recurrent network layer for temporal attention and outputted optimal Q-value for desire state–action. Unlike most CNN-based Reinforcement Learning navigation which requires start and destination for its reward function, Abhik Singla et al. [31] mainly take collision situations into account in unknown spaces.

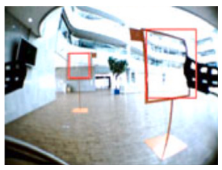
Continuous commands oppose to discrete commands which specify the quantity of movement in various directions. For example, Xue and Gonsalves [34] firstly pretrained the Convolutional Variational Autoencoder [35] (a encoder–decoder structure) to turn the RGB images into depth images, then used the latent features extracted from the encoder as current State S_t with collision and non-collision rewards to train the Reinforcement Learning model. Similarly, Vemprala et al. [36] also employed the latent features extracted from the encoder of Convolutional Variational Autoencoder to train the Reinforcement Learning model, but the inputs are asynchronous measurements of changes in per-pixel brightness at a microsecond level (i.e., event streams) which are provided by event-based camera.

In addition to being used in random navigation and obstacle avoidance, Reinforcement Learning is also used in drone racing (i.e., passing gates) in order to push the vehicle to higher speed and robustness. Several competitions have been organized such as IROS 2016–19's Autonomous Drone Racing series [37] and NeurIPS 2019's Game of Drones [38]. Fu et al. [39] applied a 2-step Teacher–Student Policy training, where the Teacher Policy training takes the ground truth of gate's and UAV's states to train the Reinforcement Learning model and the Student Policy training is based on the contrastive learning [40] to match up the extracted latent features from Teacher and Student Policy. Fu et al. [39] conducted simulated experiments along the trajectories shown in Fig. 7 (i.e., Circle, Fig. 8 and SplitS trajectories), and obtained the same level of racing performances (Circle: 4.95 s, Fig. 8: 6.76 s, SplitS: 8.58 s) comparing with the state-based policy [41] (Circle: 4.97 s, Fig. 8: 6.84 s, SplitS: 8.74 s). Fu et al. [39] also changed the environmental brightness, color and obstacles as visual disturbances and distractors, and obtained similar racing performances in three trajectories (Circle: 4.95±0.08 s, Fig. 8: 6.76±0.12 s, SplitS: 8.58±0.16 s).



(a) Positions in Corridor [42]





(b) Angle in Corridor [43] (c) Gate Position in Corridor [44]

Fig. 8. Examples of relative positions obtained by CNN-based extraction.

3.2. Direct approaches

In Direct Approaches, the Non-distance representations are normally extracted from the Field of View (FOV) based on CNN or OF algorithms and more related to the environments (e.g., corresponding to corridor's positions and object's positions) than the Integrated Approaches (e.g., corresponding to UAV Commands such as roll, pitch, yaw). The independent Control Unit outputs the corresponding reactive commands for autonomous navigation according to this extracted information. Specifically, the current complete navigation strategies focus on using CNN-based and OF-based algorithms to extract relative positions in the corridors and the difference of optical flow magnitudes for the Control Unit.

CNN-based extraction The existing complete navigation strategies applied CNN-based classification algorithms to understand the current position and followed by a specific Control Unit for behavior arbitration. For example, Padhy et al. [42] applied DenseNet-161 [43] as the backbone CNN model and modified the outputs of the last latent layer with four nodes which respectively inferred as $Center$, $Left$, $Right$ and End of corridor positions. Fig. 8(a) respectively shows the proposed images of left, center and right position of corridor excluding the end of corridor from Ref. [42]. The Control Unit receives the classification results to control the UAV shifting away from the side walls and flying along the center of the corridor. Another example relies on estimating the angle θ (0∼180°) between the central line of the corridor and the bottom edge in the input image (as Fig. 8(b) shows). By combining the genetic algorithm with Deep CNN which encodes the hyperparameters as genes, Chhikara et al. [44] proposes a DCNN-GA which firstly adjusted the hyperparameters of VGG-16 [45] during training and then, transferred the last five layers of VGG-16 by stacking them after the Xception [46] model to predict the angle θ. The Control Unit of DCNN-GA corrects the UAV's direction by adjusting the angle θ to 90°, that is, center-forward of corridors. Moreover, the CNN-based object detection identifies specific objects from visual perceptions to correct the UAV's directions. Jung et al. [47] modified the backbone network AlexNet [48] to alter the UAV's heading angle by detecting approaching gates in the images for drone racing (as Fig. 8(c) shows).

Semantic information such as Semantic Segmentation is one of the image processing methods, which is responsible for classifying the different objects in images. This method is widely used for UAV path tracking and planning. For example, Cao et al. [49] improved the light-weight ENet [50] with low computational complexity to obtain the row-to-row segmentation of farmland crops from RGB images, then applied a random sampling consensus algorithm to extract the navigation line for precise spraying in intelligent agricultural management. Similarly, Luca et al. [51] used the extracted semantic masks as inputs to train the Deep Reinforcement Learning model for path planning, where the predictive paths should be able to avoid texture-less areas that are problematic for visual odometry such as water and woodlands. However, both of them were only able to navigate at a relative higher altitude without obstacle avoidance.

OF-based extraction Optical Flow can be treated as the change of structural light in the image especially the movements of pixels due to the relative motions between the camera and scenes [52]. The OF features extracted by the OF-based Magnitude Estimation which enables the Control Unit to directly adjust orientation based on the difference of OF magnitudes on the left and right side (OF^L, OF^R) of FOV.

As shown in Fig. 9, the pixel of the red dot where the yellow lines cross is the Focus of Expansion in 3D space. The vertical blue line crossing the red dot separates the entire OF magnitudes into OF^L, OF^R and their difference is $OF^R - OF^L$. It is used by Yoo et al. [53] where the yaw angle is calculated towards the lower magnitude of OF^L and OF^R (i.e., named as balance strategy [54]), but it requires the relative distance-to-collision and the UAV's velocity. Rather than pre-defining velocity and obtaining distances from extra sensors, Agrawal et al. [55] transformed the horizontal and vertical velocities of specific pixels to the relative speed in the real world for OF-based Magnitude Estimation. An inverse strategy was followed to overcome the collision probability when the small OF difference of balance strategy causes insufficient steering rate. However, the limitation of the inverse strategy is still caused by the low OF magnitude differences between two sides which lead to left/right jitters.

3.3. Indirect approaches

Distance Perceptron in Figs. 2 and 4 provides the distance-to-collision information to the independent Control Unit for shifting away from obstacles and making decisions for navigation. Unlike the Control Unit in Direct Approaches processing the limited extracted information such as relative positions, the large amount of distance-to-collision information is usually formulated as a matrix which requires the Control Unit to comprehensively compress data by clusterization and classification, that is, to minimize the information size for navigation guidance. Both the CNN and OF algorithms provide distance estimation from captured input images while the distances detected from Reflection-based sensors such as Infrared, Ultrasonic, and Laser are fused and estimated.

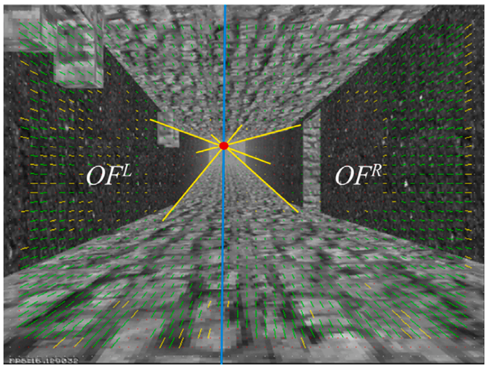
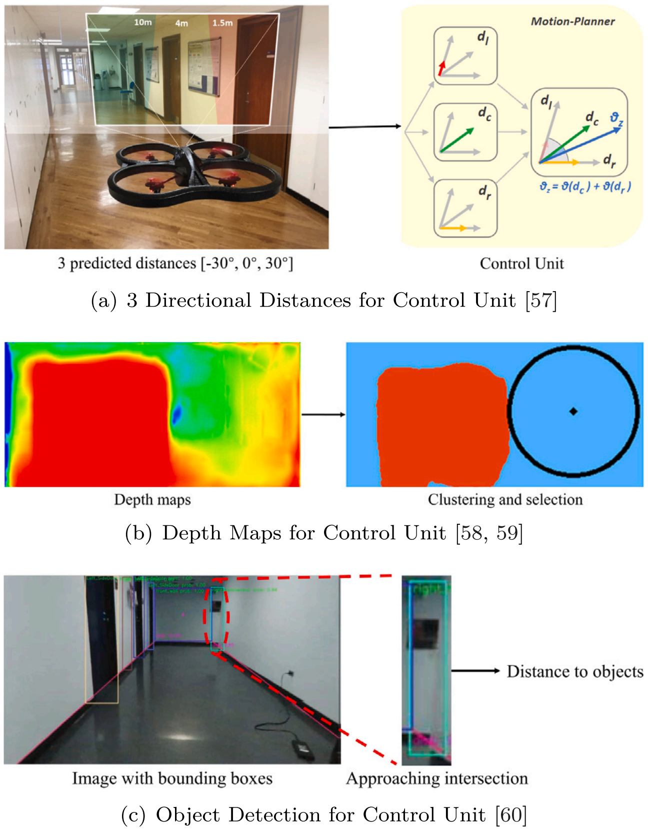


Fig. 9. OF^L and OF^R magnitudes [56]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) 3 Directional Distances for Control Unit [57]

(b) Depth Maps for Control Unit [58, 59]

(c) Object Detection for Control Unit [60]

Fig. 10. Examples of CNN-based distance estimation.

CNN-based distance estimation Kouris et al. [57] proposed a customized dataset, where each image has 3 distance labels pointing towards the $[-30°, 0°, 30°]$ of FOV shown as Fig. 10(a). A 2-stream CNN model was fed with 2 sequential frames t and $t-1$ simultaneously to optimize the distance prediction by extracting spatial–temporal features. The Control Unit receives these 3 predicted distances and calculates their sum of vectors as the next available direction, also the next linear forward velocity is processed based on the minimum distance threshold simultaneously. However, it requires large efforts to create their customized dataset by specifically installing 3 pairs of infrared and ultrasonic sensors with sensor-fusion and time-synchronization, and manually controlling the UAV flying in corridors [29].

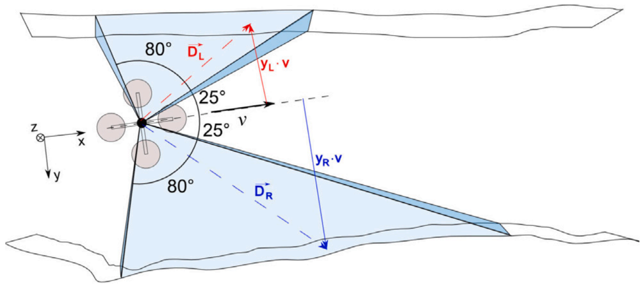
Different to collecting a customized dataset with limited distance labels, The CNN-based Distance Estimation was trained based on the public datasets to generate depth maps which can be used by the independent Control Unit to achieved self-navigation [58]. For example, Chakravarty et al. [59] used a Global Coarse Scale Network [60] to generate depth maps where the distances in the central vertical and horizontal stripes were used to steer the drone away from obstacles if it is closer than the distance threshold. To minimize the accuracy degradation of distance estimation in different scenes and improve optimal path selection, Yang et al. [61] proposed an obstacle avoidance system based on ResNet-50 [25] which includes an average measured depth error when approaching obstacles. This error occurs when the limited FOV only observes part of the structures since it is too close to the obstacles (i.e., structure incompleteness). The further research of Yang [62] used predicted confidence from the outputs of distance estimation instead of just using a measured depth error to effectively minimize the deviations. As shown in Fig. 10(b), the Control Unit of these avoidance systems [61,62] clusters the similar depth values and selects the nearest and maximum free space within FOV as candidate waypoints (i.e., the center of the black circle).

The CNN-based object detection is also a good way to estimate distances such as Garcia et al. [63] who applied YOLO v3 [64] to only identify the useful indoor structures such as corridor intersections (e.g., red dash circle), walls and doors rather than recognizing all objects in public datasets. As shown in Fig. 10(c), since the different distances to the objects lead to different bounding boxes' dimensions, the Control Unit regresses the bounding boxes' dimension to real-world distances.

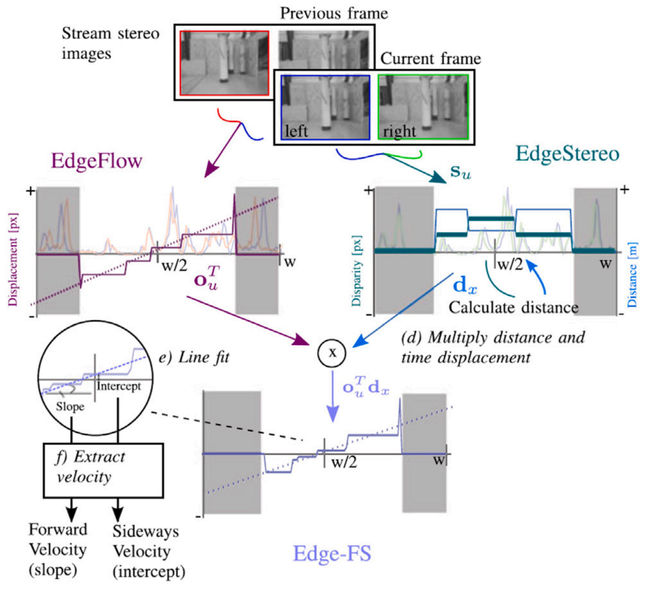
OF-based distance estimation As for OF-based Distance Estimation, Zingg et al. [65] illustrated that the OF magnitudes compensated by IMU data can be used to estimate the distance from the UAV to both sides shown as Fig. 11(a). 250 distances are respectively estimated within the 80° areas on left and right sides of the UAV while the frontal 50° section is not taken into account. $\vec{D_L}$ and $\vec{D_R}$ respectively indicate all distance vectors pointing to left and right and their distance values are indicated by y_L and y_R. Through zeroing the normalized error $e_n = \frac{|\widetilde{y_R}|-|\widetilde{y_L}|}{|\widetilde{y_R}|+|\widetilde{y_L}|}$ where $\widetilde{y_L}$ and $\widetilde{y_R}$ are respectively absolute median values of all y_L and y_R, the UAV can navigate along the central line of corridors but require precise IMU data to eliminate deviations.

McGuire et al. [66] combined EdgeFlow [67] and EdgeStereo as Fig. 11(b) shows without using the inertial sensors to provide the relative velocity. The EdgeFlow processes two consecutive left-eyed frames (i.e., red and blue images) to estimate forward and sideways velocity while the EdgeStereo captures two current frames from both eyes of the binocular camera (i.e., blue and green images) to calculate the distance. However, when the UAV's sides are close to the obstacle such as walls, the narrow angle between the obstacle and UAV's orientation leads to incorrect distance estimation and further results in collision. Also, the limited resolution from onboard cameras results to insufficient long-distance estimation.

Reflection-based distance estimation The Reflection-based algorithms use particular sensors integrated with transmitter and receiver to measure distances based on the time of flight of specific electromagnetic waves such as radio. For example, by placing the ultrasonic sensors around the UAV as shown in Fig. 12(a), the surrounding distances to obstacles are directly collected within the limited range (i.e., 3 m) [68,69]. To compensate the limitations of shorter range and surfaces which may absorb the signals like clothes and textiles [70], a series of Kalman Filter in the Control Unit are used to fuse Infrared (red circles in Fig. 12(b)) and Ultrasonic sensors (yellow circles in Fig. 12(b)) [71,72]. The fusion-estimated distances can be used for a simple Avoidance Strategy which sets the *near*, *medium* and *far* areas to adjust the UAV's velocity based on PID control [70,71]. However, since these Ultrasonic/Infrared sensors only return the frontal distances without relative directions to the UAV, there is no steering commands to navigate through the obstacles. As a result, Du and Liu [73] applied a Fuzzy algorithm in the Control Unit to process the obstacle's direction and distance for adjusting the UAV's orientation. Nevertheless, the specific sensors' IDs and simulation experiments are required for the fuzzification of obstacle's direction. To simplify the requirements, LiDAR (yellow circle in Fig. 12(c)) which provides distances along with corresponding directions with a wide range of FOV was used by Moffatt et al. [74]. In their work, the Artificial Potential Field (APF) in the Control Unit calculates the sum of repulsive vectors to steer away from obstacles without mapping and localization.



(a) Both Sides Distance Estimation by OF [65]



(b) Edge-FS [66]

Fig. 11. Examples of OF-based distance estimation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

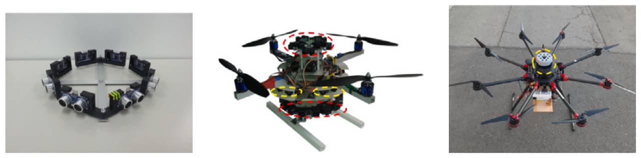

(a) Ultrasonic Sensors [69] (b) Ultrasonic-Infrared Sensor Fusion [71] (c) LiDAR [74]

Fig. 12. Examples of reflection-based distance estimation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4. Map-based navigation

According to the Overall Process in Fig. 2 and the taxonomy in Fig. 3, the processing schematic diagram of the Map-based Navigation Strategy along with its various sub-components is shown as Fig. 13 and

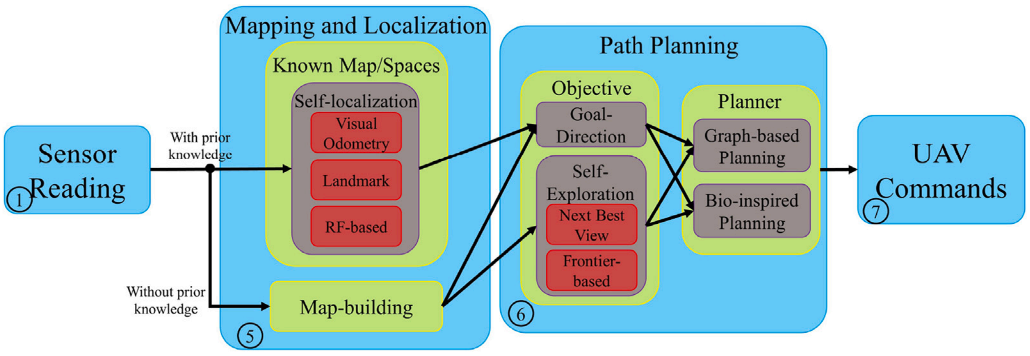


Fig. 13. Map-based navigation. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the sub-components at the same level shown as same colors. The major sub-components are ⑤ Mapping and Localization and ⑥ Path Planning where the former can be divided into Known Map/Spaces and Map-building (shown as yellow boxes in ⑤) depending on whether using prior knowledge or not. Specifically, strategies in Known Map/Spaces not just read sensor data, but also obtain prior knowledge of environments such as the positions of access points, start and destinations, obstacles or even the entire maps for UAV self-localization. In comparison, strategies in Map-building do not obtain any prior environmental information but only include map reconstruction along navigation.

⑥ Path Planning which contains different Objectives and Planners is possibly independent to the Mapping and Localization. The Objectives can be summarized as the Goal-Direction and Self-Exploration depending on navigation with a defined destination or global exploration without a destination. The Planners include the Graph-based Planning and Bio-inspired Planning. Specifically, according to the introductions of Wilson [75], a graph $G = (V, E)$ consists of a finite set V of nodes (or vertexes) corresponding to pixels/voxels of original image and a finite set E of edges connecting neighboring nodes. The Graph-based Planning includes Graph Searching and Graph Sampling algorithms, which the Graph Searching is responsible to search through a set of nodes (V) on a graph or map (G) for optimal path searching while the Graph Sampling algorithms usually sample the environment as a set of nodes (V), or other forms, then map and search to find an optimal path [11]. The Bio-inspired Planning can be literally understood as mimicking biological behaviors to deal with path planning and optimization. The rest of this Section will respectively illustrate the relevant technologies within ⑤ Mapping and Localization and ⑥ Path Planning for the complete UAV navigation strategies.

4.1. Mapping and localization

4.1.1. Known map/spaces

Prior Knowledge includes the positions of accessing points, obstacles, start and destination and even the entire 2D/3D maps. Therefore, it is not always necessary to reconstruct the maps during navigation, but the UAV Self-localization technologies such as Visual Odometry (VO), Landmark and Radio Frequency-based (RF-based) localization of the existing complete UAV navigation strategies are required.

VO Couturier and Akhloufi [16] defines VO as a process which compares the current and the previous visual observed from a single or multiple cameras to analyze differences in egomotion (i.e., egomotion estimation or pose estimation). Methods of VO pose estimation can be divided into two classes, Feature-based and Direct Methods, where the former relies on invariant feature descriptors and the latter estimates motions directly from intensity values. For example, Al-Kaff et al. [76] applied Feature-based VO based on the normal procedures which include the feature detector, feature matching in sequential images and frame-to-frame pose estimation. The Direct Methods of VO was further developed by Forster et al. [77] who proposed a semi-direct VO which matches few feature patches instead of the entire images for pose estimation. To increase the pose estimation accuracy, Forster et al. [77] also minimized the deviations of feature patch matching based on iterative Gauss Newton procedure. These two approaches [76,77] are based on the downward monocular camera but the VO pose estimation can also be used based on forward stereo camera. Through generating depth images from stereo camera, Fu et al. [78] tracked the features of depth images based on the common Feature-based VO for pose estimation.

Landmark Landmarks with specific representations that are different from the background in the FOV have shown great opportunities for localization in cooperation with visual recognition [79,80].

Dawadee et al. [81] proposed a landmark-based navigation system which used a three-stage detector [82] to recognize roof landmark patterns as waypoints, by comparing with stored features in database for UAV's drifting correction during point-to-point navigation as Fig. 14(a) shows.

Apart from matching features in database, the CNN-based object detection, SSD Inception V2 was used to detect the QR code as landmarks (shown in Fig. 14(b)) which contain coordinate information for the UAV's calibration [83].

Similarly, Mac et al. [84] used ground patterns that contained global positions as landmarks to calculate the UAV's orientation and position, further converted to the world coordinate. As Fig. 14(c) shows, each row includes three bits, the bit values are zero and one corresponding to the white and black bit. The global coordinate is calculated based on Eq. (1).

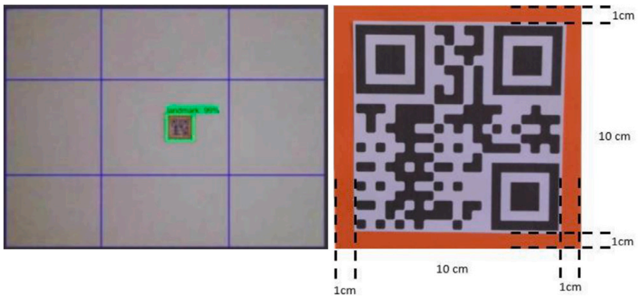
$$\begin{aligned} x &= 2^0 x_0 + 2^1 x_1 + 2^2 x_2 \\ y &= 2^0 y_0 + 2^1 y_1 + 2^2 y_2 \end{aligned} \tag{1}$$

The ground landmarks can also be treated as path for trajectory tracking shown in Fig. 14(d), such as Grijalva and Aguilar [85] who used ORB algorithm [86] to extract feature points of ground landmarks for each divided region of interest as well as setting the median of feature points as the route interpolation for path generation.

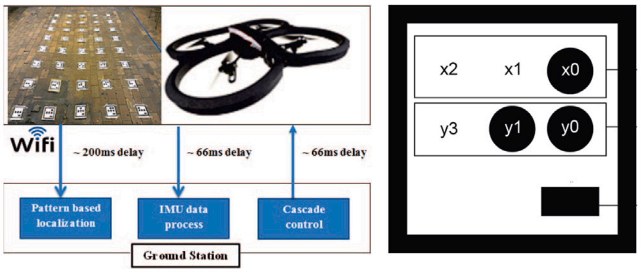
These large number of manually placed landmarks are mostly used for localization calibration, but do not have placement management. Therefore, Wang et al. [87] firstly segmented the values between [0, 1] to represent the possibility p that the landmark k can be sensed by the UAV v in different distances $dis(\cdot,\cdot)$ according to the Eq. (2). Based on the predefined trajectories, a minimal number of landmarks will be optimally placed at specific locations to ensure that the UAV can continuously acquire information from all landmarks with a possibility over than 1 (i.e., $\sum_k p_k$).

$$p = \begin{cases} 1, & 0 < dis(k,v) \le 15 \\ 0.6, & 15 < dis(k,v) \le 25 \\ 0.3, & 25 < dis(k,v) \le 35 \\ 0, & dis(k,v) > 35 \end{cases} \tag{2}$$

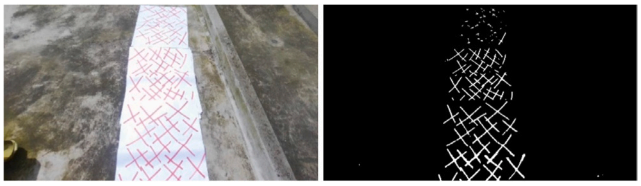
(a) Roof Landmark Pattern [81, 82]



(b) QR Landmark Pattern [83]



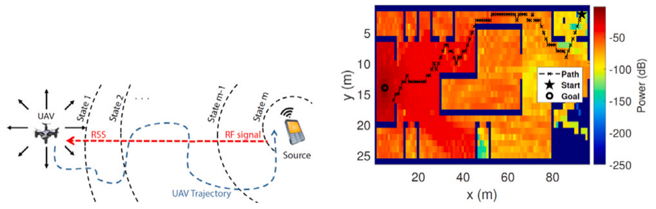
(c) Ground Landmark Pattern [84]



(d) Ground Landmark Path [85]

Fig. 14. Examples of landmarks.

RF-based The communication infrastructures that can transmit and receive RF signals such as WiFi access points and cellular towers are installed in many urban environments. The calculations based on Radio Signal Strength Indicator (RSSI) and Time difference of Arrival enable distance measurements in GPS-denied environments [88]. For example, Nunns et al. [89] presented a modified formula from [90] to explain the relationship between the RSSI and distance by adjusting the parameters through experiments. If the distances and at least 3 exact positions of RSSI sources are known, the trilateration is used to localize the receiver (i.e., UAV localization) such as Stojkoska et al. [91] and Marasigan et al. [92]. Another RF source such as the cellular tower allows to calculate the pseudorange to the receiver based on the fundamental of Time Difference of Arrival [93]. The positions of cellular towers can be treated as landmarks to minimize the accumulative errors during dead-reckoning navigation [94], or using an Extended Kalman Filter to estimate the receiver's own states while simultaneously estimating the positions of cellular towers [95].



(a) RF Signal Search (b) Heatmap of RSSI

Fig. 15. RF-based localization for UAV aided search and rescue operation [96–98].

Table 2
UAV states with respect to RSSI.

State	RSSI (dBm)
s = 1	$P_R > -40$
s = 2	$-50 \leq P_R \leq -40$
s = 3	$-60 \leq P_R \leq -50$
s = 4	$-70 \leq P_R \leq -60$
s = 5	$-80 \leq P_R \leq -70$
s = 6	$-90 \leq P_R \leq -80$
s = 7	$-100 \leq P_R \leq -90$
s = 8	$-110 \leq P_R \leq -100$
s = 9	$-120 \leq P_R \leq -110$
s = 10	$P_R < -120$

The group of North Carolina State University [96–98] dedicated to sense RSSI values which help to lead the UAV towards the victim as Fig. 15(a) shows. Instead of using RSSI or Time difference of Arrival for distance estimation, they defined the received RSSI values at each position as states and converted into a heatmap (Fig. 15(b)), then applied Q-learning [99] to plan a path towards the target with higher signal strength. Ciftler et al. [96] firstly transformed the observed RSSI value P_R into the corresponding UAV state as shown in Table 2. To solve the challenge of choosing inappropriate state from the rough given range (Table 2), Chowdhury et al. [97] assigned state labels based on the new RSSI values which referred to the fact that each grid (separated by 1 meter) has a unique RSSI value. Furthermore, in order to locate the target as fast as possible, Kulkarni et al. [98] conducted experiments and revealed that the directional antenna has a shorter convergence time and 2.4 GHz has better propagation through constructions in indoor areas. Similarly, Jayasekara et al. [100] proposed an RSSI measurement model characterized by the likelihood function modeled with Gaussian mixture to locate the potential RF sources.

4.1.2. Map-building

Map-building navigation strategies are also known as SLAM-based navigation [101] shown as Fig. 16. It concentrates on constructing local/global maps during navigation in totally unknown areas to overcome the challenges of chicken-and-egg dilemma, that is because localization requires the 3d model of the world while building 3D model in turn requires vehicle poses. According to two commonly used sensors (cameras and LiDARs), We will briefly introduce the technologies of V-SLAM and LiDAR-SLAM, which act as Mapping and Localization of existing complete UAV navigation strategies.

V-SLAM Currently, monocular SLAM can be divided into Feature-based and Direct methods such as the popular ORB-SLAM [102] and LSD-SLAM [103] shown as the different processing schematic diagrams in Fig. 17.

Specifically, the Featured-based ORB-SLAM requires descriptors (e.g., SIFT, SURF and BRIEF) to extract and match key-points in two consecutive images. According to these matched key-points, minimizing their reprojection errors (the pixel-position error between the projected pixel of the real-world 3D-point and the reprojected pixel 3D-point based on the current estimated pose) to estimate the camera pose and generate sparse 3D maps. However, the sparse 3D maps might

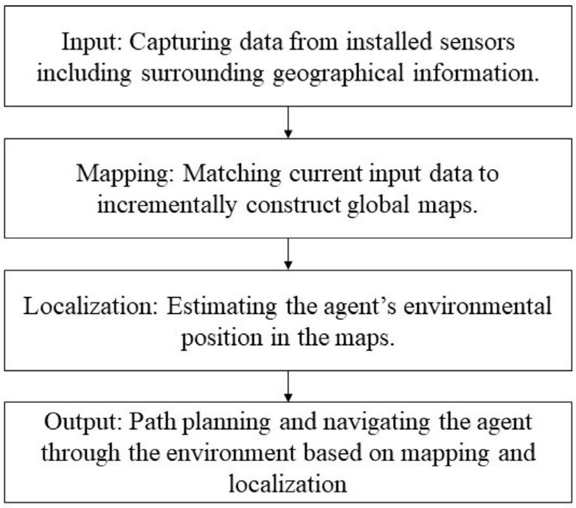

Input: Capturing data from installed sensors including surrounding geographical information.
Mapping: Matching current input data to incrementally construct global maps.
Localization: Estimating the agent's environmental position in the maps.
Output: Path planning and navigating the agent through the environment based on mapping and localization

Fig. 16. General steps of SLAM-based navigation [101].

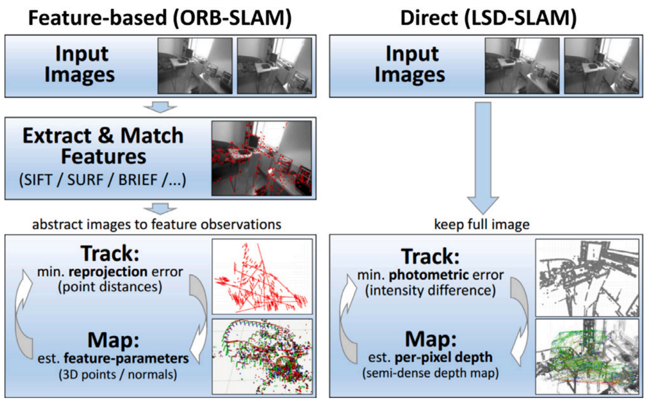


Fig. 17. Feature-based ORB-SLAM and Direct LSD-SLAM [104].

miss important information. Therefore, Esrafilian and Taghirad [105] enriched the UAV's surrounding points which is generated by ORB-SLAM based on the plane fitting, clustering and classification for map-building.

In contrast to Feature-based ORB-SLAM, Direct LSD-SLAM does not use descriptors to search key-points from images but instead uses the image intensities to estimate the location and surrounding area. For example, Stumberg et al. [106] applied the Direct monocular LSD-SLAM to optimize the camera pose by minimizing the photometric errors of the high gradient pixels and generate semi-dense maps. In order to overcome the difficulties of observing texture-less areas, Stumberg et al. [106] developed a *star discovery* motion around the UAV's current position for 360° local map construction to increase the surrounding occupied voxels.

Apart from monocular V-SLAM, RGB-D cameras use structured Light to directly provide depth information and RGB images for dense 3D reconstruction without any motion [107]. For example, Xu et al. [108] used the OctoMap packages [109] with RGB-D SLAM [110] which extracts and matches the RGB visual feature points for pose estimation consecutively and then, together with dense point clouds to generate 3D coordinate system.

Researchers also integrated the instantaneous IMU readings with consecutive visual frames for better pose estimation and localization, which inspired Chen et al. [111] to propose a FLVIS [112] stable control system with a stereo camera. Its IMU model is a loop closing thread with decreased gradient feedback for accurate and robust orientation

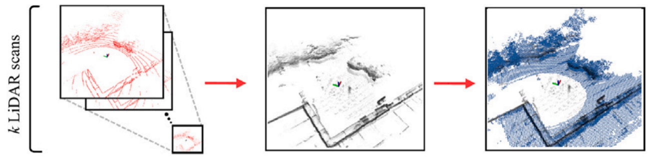


Fig. 18. Cartographer submap generation [120,121]. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

estimation. Then the IMU states are fed into the RGB-D feature-based pose estimation for correction and the errors are fed back to the IMU model. Another Visual-Inertial SLAM algorithm, OKVIS [113], was used by Alzugaray et al. [114] for monocular-inertial SLAM. Different from FLVIS [112] which uses gradient feedback for orientation estimation at first, OKVIS [113] uses the instantaneous IMU readings to predict the current pose state for keypoint extraction and matching. The key-point reprojection and IMU errors (i.e., camera and IMU pose error, consecutive images relative pose error) are conjunctly optimized.

Lidar-SLAM Unlike visual sensors which may require relative motions and/or is sensitive to textures and illumination for environment reconstruction [115], LiDAR calculates the time of flight of a laser to directly output the relative distance and position of obstacles, creating what are known as point clouds. If obstacle information and current vehicle location is known, local/global environments can be easily reconstructed. Therefore, several environment simulation engines such as Gazebo [116] with UAV model plugins which can provide vehicle ground truth pose estimation were used by [117–119] and assumed there is no error for vehicle localization. The obstacle information from LiDAR scanning can be mapped into the multi-resolution OctoMap [109] for environment reconstruction.

In contrast to assuming no errors for localization, Batinovic et al. [120] applied the Google's Cartographer SLAM algorithm [121] to generate 3D maps shown as Fig. 18. They introduced the concept of submaps and replaced the scan-to-scan to scan-to-submap matching. Each LiDAR scan (Fig. 18 (red)) tries to insert into the local current submap (Fig. 18 (black)) by placing more hit points at the occupied grids based on loop closure optimization until the current submap completes creation (Fig. 18 (blue)). Then adjusting the pose of each submap to minimize the overlap errors between them to construct global maps. Another LiDAR-SLAM was proposed by Youn et al. [122], which uses the error-state Kalman filter to fuse the visual odometry and IMU status for pose estimation. Then the obstacle information from 2D LiDAR and pose estimation are conjunctly mapped into the OctoMap for 2D reconstruction.

4.2. Path planning

4.2.1. Objective

The Map-based navigation strategies contains different objectives for planning as shown in Fig. 13. The Objective can be divided into Goal-Direction [122] and Self-Exploration [123] where the former can be literally understood as giving potential destination in the environments for autonomous navigation. The Self-Exploration is more complicated since it requires the discovery of the entire unknown spaces and the problem is considered to be fully solved when $V_{free} \cup V_{occ} = V \backslash V_{res}$ [118], where the $V_{free} \subset V$, $V_{occ} \subset V$ and $V_{res} \subset V$ respectively represent the free, occupied and residual areas of the entire 3D space $V \subset R^3$. Therefore, the rest of 4.2.1 will study the guidance system of Self-Exploration, that is the Next Best View (NBV) and Frontier-based methods, which make decision for the next waypoint.

Frontier-based The Frontier-based algorithm was introduced by Yamauchi [124] and used on ground vehicle platform at the beginning. The common procedures of Frontier-based algorithm are: (a) Defining the $Frontiers$ regions at the edges between the explored and unexplored areas as candidate waypoints; (b) Calculating the comprehensive gain of candidate waypoints containing benefit (e.g., information gain) and cost (e.g., path length) metrics; (c) Selecting the next waypoint from candidates with the best comprehensive gain.

According to these procedures, Yamauchi [124] groups the candidate waypoints at frontiers as a set and guided the vehicle to the nearest accessible and unvisited boundaries. Faria et al. [117] further developed the frontier selection by considering if the next waypoint is visible, accessible and has enough free space enabling the UAV to fly in a circle for 3D reconstruction. Also, the frontier selection searches locally (i.e., around the UAV) when no frontiers are found, then escalates to global search. Instead of traversing local and global frontiers to select the next unknown point, Batinovic et al. [120] performed mean-shift clustering [125] to decrease the number of candidate waypoints from dense frontiers for faster selection. Their works defined the information gain as the number of unknown voxels around the candidate waypoints and then, combined with navigation distance to comprehensively make decision.

NBV In comparison with Frontier-based, NBV methods consider the candidate waypoints within the range of entire known spaces including the current sensor measurement rather than just frontiers and then, calculate the comprehensive gain for selection. The Frontier-based can also be regarded as an NBV algorithm, but the candidate waypoints are generated at frontier regions.

Bircher et al. [118] followed the principles of NBV. After the current point clouds, depending on the sensor scope (FOV and maximum ranges), were matched into the global maps, an incremental geometric tree was built from the current position to the surrounding unmapped voxels in known space. The information gain extracted the best node which has more explorable unmapped voxels and longer paths. The experiments show that the NBV is able to find more hidden areas than the Frontier-based exploration [118]. Wang et al. [119] also built an incremental tree as a topological roadmap for self-exploration but the key difference is to only generate within the sensor scope. With just detecting the frontiers around each node on the roadmap, the evaluation efficiency of information gain was improved and used along with cost-to-go to decide candidate waypoints. Similarly, Yang et al. [126] used Mutual Information [127] to select a path towards the information-rich area. Another contribution is about overcoming the restrictions of resolution and orientation of the grid-based occupancy map, which uses Gaussian Process occupancy maps to predict the probabilities of obstacles.

4.2.2. Planner

The Planner is the last component of the Map-based navigation strategies (shown as Fig. 13) and aims to find a point-to-point optimal path with collision avoidance. According to definitions in the second paragraph of Section 4, planners can be summarized as Graph-based Planning and Bio-inspired Planning shown in Fig. 13. Specifically, Graph-based Planning includes **Graph Searching** and **Graph Sampling** algorithms which are all based on the finite nodes of environments while the Bio-inspired Planning driven by biological behaviors which aims to optimize the planning results.

Graph-based planning As shown in Fig. 19, if the nodes in the environments had been assigned such as using multiple regions with same size to represent nodes [128], the **Graph Searching** is responsible to plan an optimal path from start to goal. **Graph Searching** includes various algorithms such as Dijkstra [129], A* [130], Theta* [131], Lazy Theta* [132] and Jump Point Search (JPS) [133]. **Graph Sampling** is responsible to sample environments as a set of nodes, or other forms, then search to find an optimal path from start-to-goal according to [11]. There are Active and Passive sampling methods where the former indicates algorithms can form a path to goal all by its own processes (e.g., APF [134] and Rapid Random Tree (RRT) [135]) while the latter (e.g., Probabilistic Roadmap (PRM) [136] and Voronoi Diagram [137]) generates a set of paths as a map net which requires a combination of **Graph Searching** algorithm. The followings will detailly introduce the specific algorithms shown in Fig. 19.

As for **Graph Searching**, the classical Dijkstra [129] is a Breadth-First Search which expands ranges in circles around the start until this expansion reaches the goal. It was used by Maini and Sujit [138] who defined the obstacles' vertexes as nodes and considered the steering angle constraints [139], also applied a goal-to-start reverse search if start-to-goal searching fails to find the destination.

A* [130] expands Dijkstra by taking goal-cost into consideration, which is more goal-directional than the Breadth-First Search of Dijkstra. For example, Feng et al. [140] reduced the number of candidate waypoints by only calculating the vertexes approaching to the goal on the obstacles' boundaries that satisfy the dimension of UAV. Although A* [130] optimized the planing efficiency with goal-cost function, the agent's next move is one of the eight surrounding directions, which makes stiff and jittered path planning. As a result, Theta* [131] uses the framework of A* but proposes a line-of-sight (LOS) checking to delete the intermediate nodes between two direct-connected waypoints which has no collision. This helps to smooth the planned path but raised computational demands of LOS, so Lazy Theta* [132] delayed LOS checking until a point when the nodes are completely opened. It was used by Faria et al. [117] to reach an accessible unknown point for self-exploration.

Another optimization of A* path searching is Jump Point Search (JPS) proposed by Harabor et al. [133]. It introduces the Force Neighbor to define the Jump Point and only put the Jump Points rather than all unnecessary intermediate nodes into *openlist* to reduce searching processes. Chen et al. [111] applied the improved JPS [141] as global planner to find the shortest path and a Heuristic Angular Search based local planner [142] to find a kinetically feasible path.

APF [134] is a well-known Active **Graph Sampling** method which refers to the significant points such as the positions of obstacles and goals as the repulsive and attractive potential nodes for goal-direction path planning [143]. However, there are two instinct limitations of conventional APF. Firstly, the complicated surrounding obstacles of the UAV produce the zero resultant force may lead to local minima. Secondly, the obstacles placed close to the destination which results to the non-zero resultant force of destination, further causes unreachable destinations. To overcome these limitations, Liu et al. [144] improves on conventional APF with an extra repulsive force pointing toward the goal when it is activated while Mac et al. [84] added the 'to-goal' distance and the 'agent's velocity' respectively into repulsive and attractive. APF can also combine with A* graph searching [145], where the A* searches a path to the next waypoint, further respectively integrating an APF local planner for obstacle avoidance and another APF direction controller pointing towards the information-rich areas [119].

Rapid Random Tree (RRT) [135] is another Active **Graph Sampling** method, which incrementally generates a geometric tree as paths towards the given points and contains random branches. For example, self-exploration strategies [118,120,126] applied RRT within the scope of explored areas from the current position to the next waypoints (e.g., searched by APF [105]) for path planning. Considering randomly generating tree nodes within the range of explored areas results in additional computational burdens, Youn et al. [122] proposed a goal-directional incremental RRT named RRT*-GD-Smart. It combines the advantages of the low-cost connection of RRT*-Smart [146] and the fast computation of RRT-GD [147] which only generates the new nodes towards the destination.

Probabilistic Roadmap (PRM) [136] and Voronoi Diagram [137] are the Passive **Graph Sampling** methods which require additional graph searching based on the generated map nets. PRM [136] for self-exploration is a node-based multi-query planner which contains the

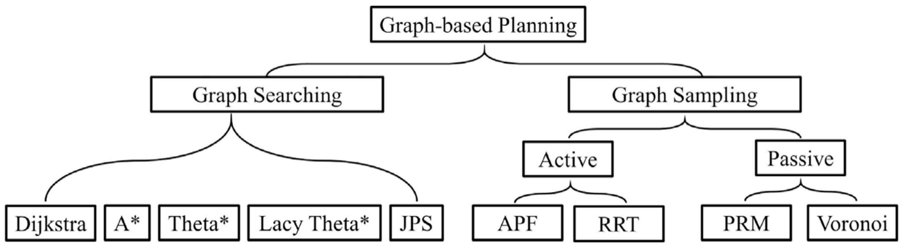


Fig. 19. Graph-based planning algorithms in the complete navigation strategies.

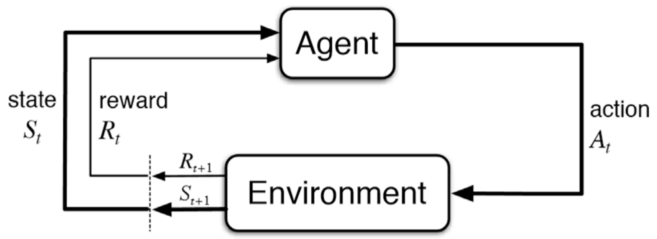


Fig. 20. The agent–environment interaction in Reinforcement Learning [154].

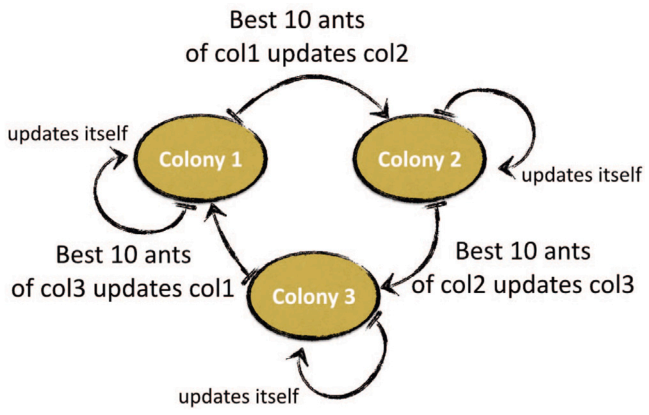

Best 10 ants of col1 updates col2
updates itself
Colony 1
Colony 2
updates itself
Best 10 ants of col3 updates col1
Best 10 ants of col2 updates col3
Colony 3
updates itself

Fig. 21. Knowledge sharing of multi-colony ACO [157].

roadmap expansion, the feasible path planning and the random-bounce walking. The limitation of the original PRM is the unused nodes which may lead to the redundant computation. Therefore, Xu et al. [108] proposed a dynamic exploration planner by respectively using incremental roadmaps which only adds new nodes with high scores for unknown areas, and uses a Euclidean Signed Distance Function to optimize execution time and path length.

The generated Voronoi Diagram [137] contains a cell's boundaries as secure trajectories by regarding obstacles' boundaries as seed nodes to decompose a visible graph. However, the limitation of the Voronoi Diagram is the large computational demand which results in a limited execution frequency. Therefore, Gruter et al. [148] gave a tailored and analytical method for Voronoi Diagram generation containing safe paths with parabola borders, and used Dijkstra [129] to construct a tree like shortest-path with shortcut path checking.

Bio-inspired planning Path Planning of the complete navigation strategies include Bio-inspired Planning algorithms such as CNN-based Reinforcement learning [149], Ant Colony Optimization (ACO) [150, 151], Genetic Algorithm (GA) [152] and Particle Swarm Optimization (PSO) [153].

Reinforcement Learning contains four prime components, the Agent, Environment, Reward and Action, which interact with each other as shown in Fig. 20. The current Agent's Action A_t in the Environment obtains the next Reward R_{t+1} and State S_{t+1}, then feeds back to the Agent as a loop. For example, an insect visual system such as the Lobula Giant Moment Detector is the inspiration for the perception network proposed by He et al. [155], where the State is the current UAV's vision and the manual-designed Reward function determines the best yaw and velocity for the next movement. Q-learning [99] is an improved Reinforcement Learning technique which combines the States and Actions into Q values $Q(s, a)$ to represent the current estimation. The works of North Carolina State University [96–98] defined RSSI values as States, and eight uniformly-spaced direction actions as the output of the Reward function which are calculated from the difference between the last two received RSSI signals ($RSSI_t - RSSI_{t-1}$). Moreover, regarding static 2D maps as input images, Theile et al. [156] introduced a Double Deep Q Network with two uniform CNN models (i.e., policy and target) which were fed with a local map around the agent and the global map simultaneously for immediate collision avoidance and general direction decision respectively. The system shows that the proposed strategy can plan a path to trade off the multi-objective of coverage path planning and data harvesting.

ACO [150,151] mimicked the behavior characteristics of ant colonies, that is, iteratively tracking pheromone trails between waypoints to find the optimal path solution. However, a single ant colony has probabilities to output sub-optimal solutions because of the early stop. To avoid this problem, Cekmez et al. [157] proposed a Multi-Colony optimization for knowledge sharing between colonies shown in Fig. 21. When all colonies complete one iteration and their ants have been sorted according to their fitness values, the first half of the ants in each colony directly update their pheromone tables while a local optimization is applied to the second half to update themselves. At each tenth iteration, the best 10 ants of each colony share their pheromones to colony next to them. The results indicated that the multi-colony optimization outperforms a single colony solution in selection of different paths without being trapped in a sub-optimal path.

GA [152] is a global optimization that imitates genetic procedures containing core operators of Selection, Crossover and Mutation to search for the optimal solution shown as the processing diagram in Fig. 22. Specifically, GA used in path planning usually encodes the feasible regions or waypoints in environments as genes to generate chromosomes, then calculates fitness values and selects two parents chromosomes (i.e., Parent 1 and Parent 2) from start (green 1) to destination (red 20). Two point Crossover exchanges the yellow gene segments between two parent chromosomes to generate Offspring 1 and 2 while Swap Mutation swaps the gray genes in both two Offspring chromosomes to generate Offspring 1′ and Offspring 2′. Optimization will retain the better chromosome by calculating the fitness values until iteration complete.

Original GA principles inspired Sonmez et al. [158] to solve the Traveling Salesman Problem in a 3D virtual environment. This method encodes the intermediate waypoints as genes and the paths as chromosomes, then uses tournament selection to choose two better parent chromosomes. The following 2 point Crossover and the Swap Mutation of Sonmez et al. [158] are same as the processing diagram of Fig. 22 to generate better offsprings. Another GA path planning was proposed by Golabi et al. [128], which decomposed a 3D environment into finite

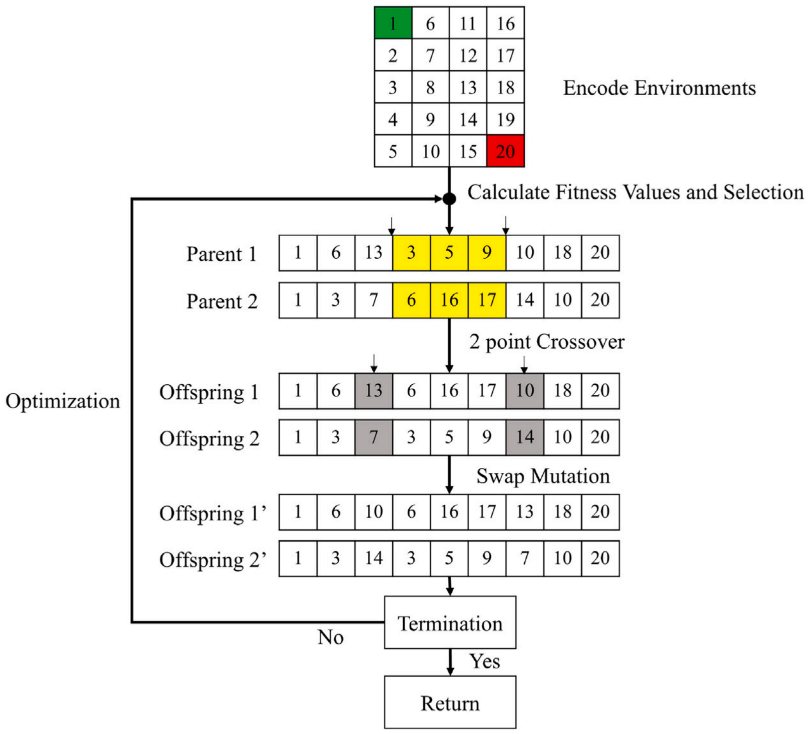


Fig. 22. Processing diagram of GA. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

grids as waypoints according to different altitude levels and ground surface cells. The path length, energy consumption and path risk were taken into consideration for multi-objective GA which structured two-row chromosomes where the first-row genes are visiting cells and the second-row represents random altitudes.

To eliminate the requirements of encoding chromosome for GA and high sensitivity of swarm's size for ACO, PSO adjusts the speed and position vectors of particles based on their own experiences (individual best) and swarm communications (global best) to seek the optimum values. For example, Singh et al. [94] defined the cellular towers as waypoints and a path consists of a set of waypoints, and the best particle will be found by updating the swarm by considering the time taken from start to goal. To improve the performance of convergence and global/local best searching, Mirshamsi et al. [159] applied parallel computing for multi-directional particles (i.e. x, y and, z), and stop until the maximum allowed iterations or the cost is less than the given threshold. The experiments illustrated that the convergence time is lower than 1 s and it has potential to be utilized in real-time dynamic path planning in 3D GPS-denied environments.

5. Performance evaluation

The taxonomy of Fig. 3 is based on common characteristics of sub-components and their processing flowcharts, which are possibly independent and different. Therefore, Section 5.1 will firstly separate the complete UAV navigation strategies into Random Navigation, Goal-Directional Navigation and Self-Exploration categories based on corresponding termination conditions. Specifically, the Random Navigation means the UAV Mapless navigation in unknown spaces without destination and its termination condition is collision on obstacles; The Goal-Directional Navigation represents the UAV Map-based navigation which generates optimal path towards destination and its termination condition is reaching target; The Self-Exploration is the UAV Map-based Self-Exploration in unknown spaces and its termination condition is complete exploring spaces. 5.1.1 will explain the three proposed Evaluation Metrics used to respectively analyze these three categories and followed by discussions in 5.1.2.

Although the 5.1 Theoretical Evaluation comprehensively analyzes and discusses the complete navigation strategies, they were conducted in different environments based on various devices and platforms. Therefore, the comparison between these strategies based on the same environments, devices and conditions is necessary, and it is also scarce in the current review papers. As a result, the simulated performances of the representative complete navigation strategies in two reality-like environments will be evaluated in our own implemented experiments in 5.2 as guidance for future researches.

5.1. *Theoretical evaluation*

5.1.1. *Evaluation metrics*

The current review papers mostly focused on the sub-components of autonomous navigation (e.g., Localization and Path Planning) which can be concluded with specific evaluation metrics (e.g., accuracy of localization and computational complexity of path planning) since they have almost similar application. However, the review papers for the complete navigation strategies which start from data capturing to the control policy or path planning are scarce, and lack of a uniform criteria to evaluate them since the technologies and purposes are diverse.

Therefore, three uniform Evaluation Metrics (i.e., Path Length, Deviation Rate and Exploration Efficiency) are proposed to evaluate the various strategies according to their common characteristics of objectives including corresponding termination conditions. The original results from the literature in 5.1.2 is summarized to illustrate how well the strategies can perform.

• Path Length - Len (m): This is the evaluation metric for the Mapless Random Navigation. Since the termination condition of these strategies is occurrence of collision, the usual evaluation metrics are Flight Time [57] and Path Length [26]. However, the Flight Time can vary based on the UAV's speed and battery power, therefore, it is proposed that the Path Length Len (m) be used for fair comparison, i.e. the greater Len (m) the UAV is able to travel without collision, the better performance it is.

Table 3
Environmental complexity.

Envs detail	Envs type
Open spaces	Envs 1
Enclosed rooms (e.g., office, seminar, café rooms)	Envs 2
Corridor	Envs 3

• Deviation Rate - $Rate$ (%): This is the evaluation metric for the Map-based Goal-Directional Navigation which has no prior environmental information (e.g., the entire maps) except the given start and destination. The UAV generates the shortest path from start to destination, and terminates navigation until reaching the destination without collision. Therefore, we use Deviation Rate $Rate$ (%) for fair comparison by taking the errors between the ideal shortest path length $SLen$ (m) and the actual generated path length Len (m). The smaller the calculated $Rate$ (%) based on the following Eq. (3), the better the performance.

$$Rate = \frac{Len - SLen}{SLen} \times 100\% \tag{3}$$

• Exploration Efficiency - $Effc$: This is the evaluation metric of the Map-based Self-Exploration. The termination condition relies on the completion of discovering the entire environment. The most common evaluation metric is the ratio of the explored areas (m^3) divided by Flight Time or Path Length, where the explored areas represented by various indexes such as explored volumes [117–119]. We use the actual Path Length Len (m) and the $Size$ (m^3) of the environments to evaluate their Exploration Efficiency $Effc$ expressed in Eq. (4). The shorter Len to explore the larger $Size$, the better performance of exploration, i.e. the larger $Effc$ the better.

$$Effc = \frac{Size}{Len} \tag{4}$$

Since the navigation strategies were conducted in various environments which seems impossible to compare, we group the environments into three major types according to specific details such as Open Spaces, Enclosed Rooms and Corridor shown as Table 3. It helps to minimize the impact from environments. Specifically, The Open Spaces represent the environments with no boundaries while the Enclosed Rooms mean the environments with boundaries such as office, seminar and café rooms.

5.1.2. Theoretical analysis

This part will analyze the theoretical performances of Random Navigation, Goal-Directional Navigation and Self-Exploration by listing their environmental complexity, essential approaches and technologies, evaluation metrics and limitations in Tables 4, 5 and 6. In these three tables, **Envs** indicates the environmental complexity including the environment types (i.e., Envs1, Envs2, Envs3) from Table 3 and Dynamic/Static (D/S) obstacles (e.g., Poles, Walls of Static obstacles, and Human or dynamic obstacles). Len (m), $Rate$ (%) and $Effc$ are three evaluation metrics explained in 5.1.1 while the **Limitations** are concluded from original papers. The contents in evaluation metrics column includes '*' which represents the values are calculated based on the data from reference papers, and '-' which indicates the values are not provided.

According to the taxonomy of Fig. 3 and processing diagram of Mapless Navigation in Fig. 4, **Approaches** in Table 4 are Integrated, Direct and Indirect while the **Methods** are the technologies at the beginning of each paragraph in Section 3.

In Table 5, **Waypoints** represent whether it contains intermediate positions to separate a long trajectory into several segments between start and destination. According to Fig. 13, since strategies with given destination are Goal Direction, it requires the sub-components of Mapping & Localization (**M&L**) and **Planner**.

As for Self-Exploration strategies, Table 6 adds the **Exploration** methods (i.e., NBV, Frontier-based and etc.) but removes the **Waypoints** in comparison with Table 5.

By inspecting the strategies in Table 4, the Integrated Approach with Supervised Learning [26,27] implemented the same CNN model (i.e., DroNet). They obtained the maximum Len of 113 m in Corridors with Static obstacles which outperformed other Approaches with corresponding Methods. However, the most significant limitation is their dependence of 'line-like' patterns as strong features since their training datasets are collected based on outdoor car-driving which has lanes to follow. Regarding the strategies conducted in Enclosed Spaces with Static obstacles, the Indirect Approach with CNN-based Distance Estimation [59] achieved the best Len of 43.92 m. This strategy is the same as [61,62] which trained the CNN model by feeding the public visual depth datasets, but its limitations show that it is unavailable for glass surfaces such as windows on the walls and requires texture-rich backgrounds for the CNN extractor. Notably, the Integrated Approach with Reinforcement Learning [31,34,36,39] was able to execute in Enclosed Spaces with both Dynamic and Static obstacles. However, the generated depth maps of [31] have visual deviations which affect feature extraction while the simple and limited environments provided by simulators [34,36,39] lead to overfitting and affect generalization capabilities.

Because Waypoints represent the intermediate positions which help to calibrate the UAV's position to follow the trajectory, the strategies in [94] achieved the smallest $DeviationRate$ of 4.2% which outperformed other works without Waypoints in Table 5. In comparison, the LiDAR-SLAM with Graph-based RRT planner [122] obtained similar $DeviationRate$ of 14.1% to the RF-based localization with Bio-inspired Reinforcement Learning planner [98] (average 12.8%), but the latter one was conducted in relative more complex environments (i.e., Corridors). Though the Graph-based RRT planner can generate optimal path from current position to destination, the sudden appearance of obstacles due to the FOV of LiDAR and blind spots affects UAV's velocity. The RF-based localization does not face the challenges of blind spots and has a larger covered size in comparison with LiDAR-SLAM, however, the manually designed state based on RSSI values for Reinforcement Learning has limitations. For example, the specific state-definition is difficult to be used in other environments, and the unstable RSSI values lead to incorrect UAV's commands since radio signal reflect.

The strategies' Exploration Efficiency $Effc$ can be seen in Table 6. Bircher et al. [118] and Batinovic et al. [120] both applied the LiDAR-SLAM and Graph-based RRT planner for global exploration in similar environments but the only difference is the NBV and Frontier-based methods. Although Bircher et al. [118] states that the NBV exploration is able to find more hidden areas than the Frontier-based exploration, Batinovic et al. [120] uses flexible multi-resolution frontiers and mean-shift clustering which allows to decline the number of candidate frontier points, further decrease the computational burden and repeated routes. This performance can also be observed in Enclosed Spaces which Batinovic et al. [120] obtained $Effc$ of 37.5 outperforming other works. Both Xu et al. [108] and Wang et al. [119] were experimented in similar environments, it is still difficult to recognize which affects $Effc$ since they have different SLAM technologies and planners. Based on careful comparison if ignoring the localization and planner sub-technologies, both NBV exploration strategies are slightly different related to addition of nodes. Wang et al. [119] only added nodes within FOV to generate roadmaps while Xu et al. [108] added nodes in all unknown areas. As a result, the latter somewhat enlarges the checking processes for next waypoint decision.

5.2. Simulated evaluation

The purposes of this simulated experiment is to eliminate the influences caused by different environments, hardware devices and platforms, and to quantitatively compare the performances of selected

Table 4
Mapless random navigation.

Envs	Approaches	Methods	Ref	*Len* (*m*)	Limitations
Envs1 S	Indirect	Reflection-based DE[a]	[73,74]	–	[73] require multi-directional sensors with their IDs for Fuzzy Algorithm in the Control Unit. [74] required IMU data to estimate UAV's status for APF Algorithm in the Control Unit.
Envs2 D&S	Integrated	RL[b]	[31]	–	Generated depth maps with deviations affect feature extraction.
Envs2 S	Integrated	RL[b]	[34,36,39]	max 100 of [36]	The environments provided by the simulators are limited, simple and lack of real-world scenarios, which led to limited generalization capability.
	Indirect	CNN-based DE[a]	[57,59,61,62]	max 43.92 of [59]	[57,59] are unavailable to detect glass surfaces. [59,61,62] require texture-rich background.
		OF[c]-based DE[a]	[66]	36.6[e] (122 s × 0.3 m/s)	Stuck at corners since the small heading angles to walls.
Envs3 S	Integrated	SL[d]	[26,27]	max 113 of [27]	Sensitive to 'line-like' patterns.
	Direct	CNN-based Extraction	[42,44,47]	–	[42,44] affected by insufficient illumination and limited samples. [47] specifically requires gate placement.
		OF[c]-based Extraction	[53,55]	max 40 of [53]	Insufficient yawing and left/right jitters caused by close two-side OF magnitudes ($OF^R - OF^L$).
	Indirect	CNN-based DE[a]	[59,63]	max 74 of [63]	Limited FOV leads to collision if close to walls.
		OF[c]-based DE[a]	[65]	60	Require precise inertial sensor to estimate UAV's speed with deviation elimination.

[a] Distance Estimation.
[b] Reinforcement Learning.
[c] Optical Flow.
[d] Supervised Learning.
[e] Calculated based on data from reference papers in ().

Table 5
Map-based goal-directional navigation.

Waypoints	Envs	M&L[a]	Planner	Ref	*Rate* (%)	Limitations
Present	Envs1 S	RF[b]-based	Graph-based PSO[c]	[94]	4.2%[d] ($Len = 540,\ SLen = 518$)	Dead-reckoning navigation between waypoints leads to accumulative drifts.
	Envs2 S	Landmark	Graph-based APF[d]	[84]	–	Envs require to be full-covered by landmarks.
		V-SLAM	Graph-based JPS[e]	[111]	–	Require texture-rich background and more computational resources.
Absent	Envs2 S	LiDAR-SLAM	Graph-based RRT[f]	[122]	14.1%[h] (Len = 16.3, SLen = 14.28)	Recalculate optimal path when encountering unknown obstacles which affects UAV's speed.
	Envs3 S	V-SLAM	Graph-based RRT[f]	[105]	–	The insufficient sparse map enrichment leads to longer path.
		RF[b]-based	Bio-inspired RL[g]	[96] [97,98]	[98]: 7.1%[h] (Len = 45.11, SLen = 42.13); 13.4%[h] (Len = 111.34, SLen = 98.18); 17.8%[h] (Len = 111.34, SLen = 94.48)	Similar RSSI values lead to error state decision and signal reflection affects shortest path finding.

[a] Mapping & Localization.
[b] Radio Frequency.
[c] Particle Swarm Optimization.
[d] Artificial Potential Field.
[e] Jump Point Search.
[f] Rapid Random Tree.
[g] Reinforcement Learning.
[h] Calculated based on data from reference papers in ().

representative strategies in a completely unknown GPS-denied spaces for investigating their advantages and drawbacks. This Subsection will firstly give our justifications on the selection mechanisms along with corresponding strategy, then describe the simulation environments followed by our discussions and conclusions.

5.2.1. Strategies selection

Since it is impossible to verify all strategies, we used the following criteria to make our selection: (a) We considers the strategies which are involved in both Mapless and Map-based category; (b) The complete navigation strategies can be conducted in totally unknown environments; (c) The strategies based on two commonly used sensors, cameras and LiDARs, will be considered; (d) Both conventional and novel strategies with competitive performances are also taken into account. According to criteria (a) and (b), the Map-less Random Navigation strategies in Table 4 and Self-exploration strategies in Table 6 are suitable. In addition, the criteria (c) helps to select the visual-based navigation strategies in Table 4 and LiDAR-SLAM navigation strategies in Table 6. Furthermore, novel and conventional strategies with competitive performances based on criteria (d), that is novel visual-based Integrated Approach with Supervised Learning [26] and Indirect Approach with CNN-based Distance Estimation [61] in Table 4

Table 6
Map-based self-exploration.

Envs	M&L[a]	Exploration	Planner	Ref	$Effc$	Limitations
Envs1 S	LiDAR-SLAM	NBV[b]	Graph-based RRT[c]	[118]	13.9[f] (Size = 18.2k, Len = 1314)	Limited FOV affects finding the NBV.
				[126]	–	Require to pretrain GP maps for distance detection.
		Frontier-based	Graph-based RRT[c]	[120]	71.4[f] (Size = 20k, Len = 280)	Information gain as guidance may lead to repeated routes.
Envs2 S	V-SLAM	Star Discovery	–	[106]	–	Additional movements towards the measurable structures (e.g., edges and textures).
		NBV[b]	Graph-based PRM[d]	[108]	13.9[f] (Size = 0.6k, Len = 43.12)	Adding nodes within unknown spaces leads to repeated routes.
	LiDAR-SLAM	NBV[b]	Graph-based APF[e]	[119]	19.8[f] (Size = 0.9k, Len = 45.4)	Adding node within limited FOV of 2D laser scanner affects finding the NBV.
		Frontier-based	Graph-based Lazy Theta[f]	[117]	17.8[f] (Size = 75.8k, Len = 4260)	Require look around maneuver for 3D maps reconstruction.
			Graph-based RRT[c]	[120]	37.5[f] (Size = 6k, Len = 160)	Information gain as guidance may lead to repeated routes.
Envs3 S	V-SLAM	NBV[b]	Graph-based PRM[d]	[108]	8.2[f] (Size = 1.2k, Len = 318.58)	Adding nodes within unknown spaces leads to repeated routes.
	LiDAR-SLAM	NBV[b]	Graph-based APF[e]	[119]	14.3[f] (Size = 1.5k, Len = 105.2)	Adding node within limited FOV of 2D laser scanner affects finding the NBV.

[a] Mapping & Localization.
[b] Next Best View.
[c] Rapid Random Tree.
[d] Probabilistic Roadmap.
[e] Artificial Potential Field.
[f] Calculated based on data from reference papers in ().

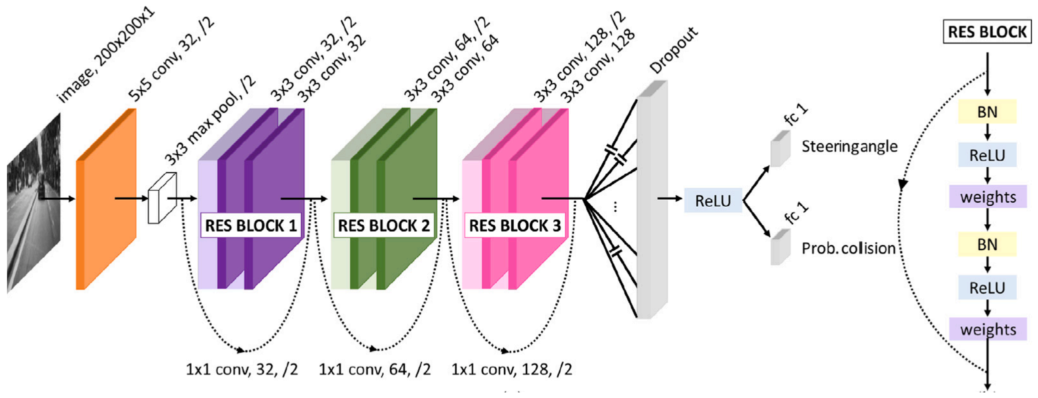


Fig. 23. DroNet architecture [26].

proposed in the most recent years, and the conventional Frontier-based navigation [124] with LiDAR-SLAM in Table 6 are selected for verification.

Mapless integrated approach with supervised learning Loquercio et al. [26] constructed a residual CNN named DroNet which directly predicts the steering angle and collision probability from the input gray images shown in Fig. 23. In Alg. 1, the Input of DroNet are gray image I_k and global 2D position $\xi_k = (x_k, y_k)$ which respectively captured by the front camera and simulator while the Output is $q_k = (V_{x_k}, \psi_k)$ which represents the UAV's rigid body command at time k. In order to smooth the continuous forward velocities V_{x_k} and yaw angles ψ_k, a low-pass filter was used for predictions shown as Eqs. (5) and (6) where p_t indicates the collision probability range from [0, 1] and s_k represents a steering angle within the range of $[-\frac{\pi}{2}, \frac{\pi}{2}]$. The DroNet model, origin coordinate and the Euclidean Distance between two points are respectively shown as N, $\xi_o = (0,0)$ and $EucDis$. We set the $LandingZone$ within 0.5 m radius centered at $\xi_o = (0,0)$, UAV will navigate out of the $LandingZone$ and the navigation task will be terminated once it reaches the landing zone again. When the path length d_p is over 1.5 m, we calculate the distance to $\xi_o = (0,0)$ as d_o. If d_p is over 1.5 m but the UAV is still in the $LandingZone$, we record it as a failure.

$$V_{x_k} = (1-\alpha)V_{x_{k-1}} + \alpha(1-p_t)V_{max} \tag{5}$$

$$\psi_k = (1-\beta)\psi_{k-1} + \beta\frac{\pi}{2}s_k \tag{6}$$

Mapless indirect approach with CNN-based distance estimation CNN-based Distance Estimation for Navigation [61] used ResNet-50 [25] as backbone CNN architecture to predict the depth maps $\hat{Y}_k$ directly from input RGB image I_k at first, then partition $\hat{Y}_k$ into i clusters via $MeanShift$ algorithm [160] as clustered depth map $Q(d_{obs_k})$. Because of the limited FOV, structure incompleteness SI_n was calculated based on Eq. (7), where (w_n, h_n) and (W, H) respectively represents the width and height of cluster n and image I_k. To reduce the prediction errors for approximating objects, Ego Dynamic Space [161] was used to compute effective distances d_{eff_k} via Eq. (8) which contains a measured mean depth error $e = 0.45$ and a predefined threshold T_{SI}.

Algorithm 1 DroNet Navigation

```
Input: Gray Image $I_k$, Global 2D Position $\xi_k$
Output: UAV's control command $q_k$
 1: Initialize $q_k$ and $\xi_o$
 2: $TakingOff$
 3: Record ground truth 2D position $\xi_k$ from the simulator
 4: Calculate $d_p = \sum_1^k EucDis(\xi_k, \xi_{k-1})$
 5: If $d_p > 1.5m$, go to step 6; Otherwise jump to step 8
 6: Calculate $d_o = EucDis(\xi_k, \xi_o)$
 7: If $d_o > 1m$, go to step 8; Otherwise jump to step 11
 8: Capture gray image $I_k$
 9: Get $q_k \leftarrow [V_{x_k}, \psi_k] = N(I_k)$
10: Go to step 3
11: $Landing$ and end the program
```

The cost functions of clusters determine the next UAV's command via Eq. (9), where r_n, ρ_n, d_{ks} and S respectively represent the radius of cluster n, the distance to the UAV's Y-Z plane, the minimum distance to the FOV boundary, and the UAV's size. If d_{eff} of all clusters are over 0 and when the size of cluster along UAV's X-axis S_c is greater than $1.5 \times S$, UAV will navigate towards it; Otherwise it will select the next direction according to the maximum C_n. Autonomous navigation of [61] is similar to DroNet which allows UAV to navigate out of and return to the $LandingZone$, but the differences are the UAV's rigid body command $q_k = (V_{x_k}, V_{y_k}, \psi_k)$ and the ResNet-50 N as Alg. 2 shows.

Algorithm 2 CNN-based Distance Estimation for Navigation

```
Input: RGB image $I_k$, Global 2D Position $\xi_k$
Output: UAV's control command $q_k$
 1: Initialize $q_k$ and $\xi_o$
 2: $TakingOff$
 3: Record ground truth 2D position $\xi_k$ from the simulator
 4: Calculate $d_p = \sum_1^k EucDis(\xi_k, \xi_{k-1})$
 5: If $d_p > 1.5m$, go to step 6; Otherwise jump to step 8
 6: Calculate $d_o = EucDis(\xi_k, \xi_o)$
 7: If $d_o > 1m$, go to step 8; Otherwise jump to step 19
 8: Capture RGB image $I_k$
 9: Predict depth map $\hat{Y}_k = N(I_k)$
10: Clustered depth map $Q(d_{obs_k}) = MeanShift(\hat{Y}_k)$
11: for $n = 1 \rightarrow i$, do
12:     Calculate $d_{eff_k}$ based on Eq.(7) and (8)
13: If $\forall\ d_{eff} > 0$, go to step 14; Otherwise $q_k = (0, 0, -\frac{\pi}{2})$ and iterate
    from step 8
14: If $S_c > 1.5 \times S$, $q_k = (V_{x_k}, 0, 0)$ and iterate from step 3; Otherwise go
    to step 15
15: for $n = 1 \rightarrow i$, do
16:     Calculate and sort $C_n$ based on Eq.(9)
17: Get $q_k \leftarrow (V_{x_k}, V_{y_k}, \psi_k)$ = the center of $max(C_n)$
18: Go to step 3
19: $Landing$ and end the program
```

$$SI_n = max\left[\frac{w_n}{W}, \frac{h_n}{H}\right] \tag{7}$$

$$d_{eff_k} = \begin{cases} Q(d_{obs_k}) - e - (v * T - \frac{a*T}{2}), & if\, SI_n > T_{SI} \\ null, & otherwise \end{cases} \tag{8}$$

$$C_n = \begin{cases} \frac{r_n}{\rho_n}, & if\, d_{ks} > S \\ 0, & otherwise \end{cases} \tag{9}$$

Algorithm 3 Frontier-based Navigation

```
Input: LiDAR data $pcl_k$, Global 2D Position $\xi_k$
Output: UAV's command $Trajectory$
 1: Initialize $Trajectory$, $\xi_o$, Global Map and Unvisited Frontiers
 2: $TakingOff$
 3: Record ground truth 2D position $\xi_k$ from the simulator
 4: Capture LiDAR data $pcl_k$
 5: Generate Local Map and update Global Map based on $\xi_k$ and $pcl_k$
 6: Update Unvisited Frontiers based on Global Map
 7: If Unvisited Frontiers=$\emptyset$, jump to step 11; Otherwise go to step 8
 8: Get the nearest $goal \in$ Unvisited Frontiers
 9: Navigation: $Trajectories = DWA(\xi_k, goal)$
10: Go to step 3
11: Come back to $\xi_o$: $Trajectories = DWA(\xi_k, \xi_o)$
12: $Landing$ and end the program
```

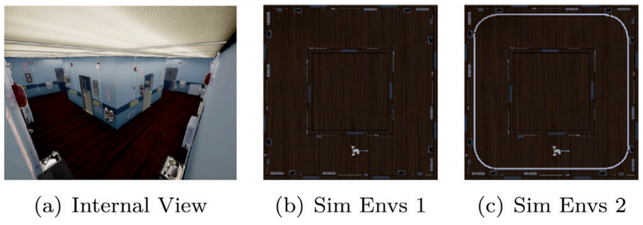


Fig. 24. Simulated environments.

Map-based frontier-based navigation The conventional Frontier-based navigation [124] classified each cell into $Open$, $Unknown$ and $Occupied$ classes. At current position ξ_k, we capture LiDAR data pcl_k to generate a Local Map and match with Global Map for update. The places that have been scanned without obstacles are denoted as $Open$, the places that have not been scanned are $Unknown$, and the places with obstacles are $Occupied$. Frontiers are the boundaries between $Open$ and $Unknown$, through updating the set of unvisited frontiers, finding out the nearest position as $goal$ and using Dynamic Window Approach (DWA) [162] from Topiwala et al. [163] as path planner to generate a $Trajectory$ which includes a set of 2D positions, UAV will explore the whole spaces and return to the start position until the set of unvisited frontiers is empty. Frontier-based Navigation is shown in Alg. 3.

5.2.2. Environments

We design two indoor environments with the same layout based on Unreal Engine 4 [164] shown in Fig. 24. Fig. 24(a) is the internal view of these two environments which shows the layout of a square area with a close corridor. Fig. 24(b) and (c) respectively represent the real-world environments, Sim Envs 1 and 2, where the difference is that the Sim Envs 2 has a square-shape white line on the ground which is used to represent the lanes. Since the limitation of DroNet [26] is sensitive to the 'line-like' pattern and is pretrained based on outdoor self-driving datasets, we use these two environments to determine how significantly the white lines impact the navigation.

A new simulator, AirSim [18], built on Unreal Engine [164] offers physically and visually realistic simulations including controlling the UAV and capturing sensor data. Therefore, Algorithms 1, 2 and 3 were able to capture RGB/Gray image I_k and LiDAR data pcl_k respectively from simulated physical camera[1] and LiDAR.[2] Furthermore, in order to minimize the impact of localization for navigation since this paper focuses on the complete navigation strategy instead of localization

[1] https://microsoft.github.io/AirSim/camera_views/.
[2] https://microsoft.github.io/AirSim/lidar/.

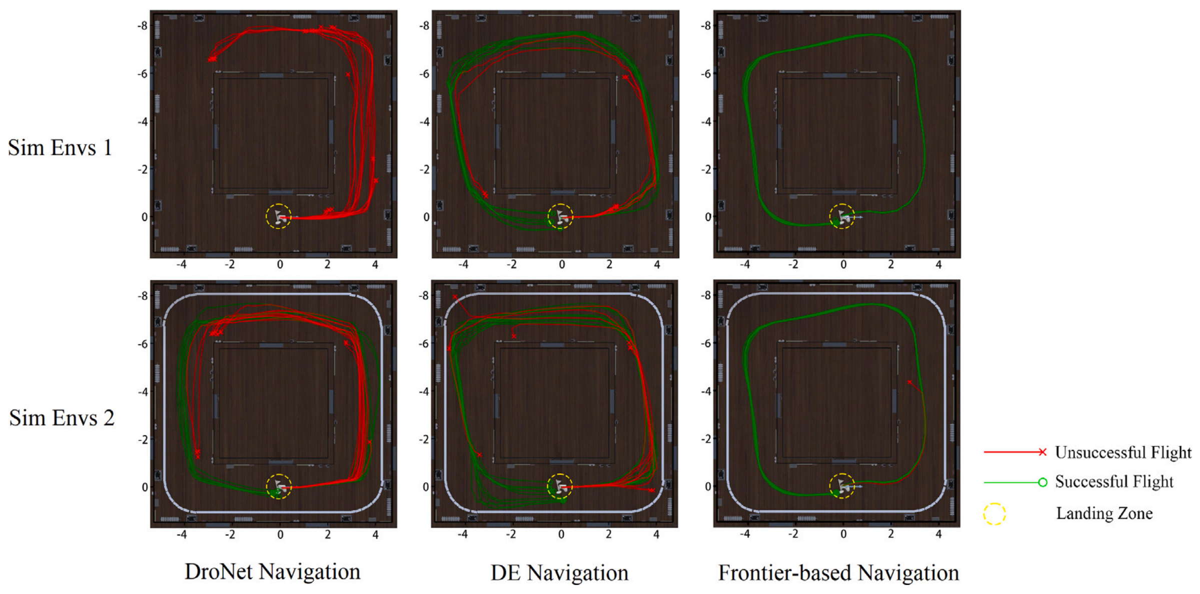


Fig. 25. Simulation trajectories. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 7
Simulation experiments.

Methods	Sim Envs 1					Sim Envs 2				
	SF[a]	MTSF[b]	MLSF[c]	MTF[d]	MLF[e]	SF[a]	MTSF[b]	MLSF[c]	MTF[d]	MLF[e]
DroNet navigation	0%	0 s	0 m	50.5 s	10.2 m	35%	97.2 s	26.2 m	74.3 s	19.3 m
DE navigation	45%	89.4 s	30.2 m	60 s	18.6 m	55%	115 s	29.32 m	90.5 s	22.3 m
Frontier-based navigation	100%	381.7 s	25.8 m	381.7 s	25.8 m	95%	379.3 s	25.7 m	366.9 s	24.8 m

[a] Ratio of Successful Flight.
[b] Mean time of Successful Flight.
[c] Mean Length of Successful Flight.
[d] Mean time of Flight.
[e] Mean Length of Flight.

accuracy, 'simple-flight' mode[3] which enables the collections of ground truth 2D position ξ_k based on a well-calibrated state estimation was applied.

The flagpole with a gamepad in Fig. 24(b) and (c) is the position where the UAV takes off. In order to generate a more realistic environment, several indoor items have been placed including doors, windows, water faucets, heaters, fire extinguishers, etc. The floor is made of dark walnut wood which is used to distinguish the overall blue wall while the white ceiling is embedded with lights. The overall dimension of this environment is $10\ \text{m} \times 10\ \text{m} \times 3\ \text{m}$. Considering safe flights and trying to simulate real environments, we calculate the width of corridors as 2.5 m and the inner walls as $5\ \text{m} \times 5\ \text{m} \times 3\ \text{m}$ according to the same proportions of $\frac{CorridorWidth}{UAV}$ in simulated and real environments.

5.2.3. Experiments and discussion

From Fig. 25, the start position is set at (0,0). The UAV will navigate from the start point along the corridor and eventually return to the $LandingZone$ shown as the yellow dotted circle with 0.5 m radius at the start position. The maximum velocity and the height above the ground were respectively set as 1.5 m/s and 0.5 m.

DroNet, CNN-based Distance Estimation and Frontier-based navigation strategies illustrated as Algorithms 1, 2 and 3 were performed in Sim Envs 1 and 2 for 20 times, that is totally 120 experiments to collect the data. The performance indicators in Table 7 include the Ratio of Successful Flight, the Mean time of Successful Flight, the Mean Length of Successful Flight, the Mean time of Flight, and the Mean Length of Flight. Among them, the Ratio of Successful Flight shows the generalization capability and stability of navigation algorithms conducting in unknown spaces. We denote the Successful Flight when the UAV navigates through the corridor and returns to the $LandingZone$ without collision and show its trajectory as a green line with a circle in Fig. 25, otherwise the Unsuccessful Flight is represented as the red line with a cross. The Mean time and Mean Length of Successful Flight calculate the average time and path length of successful navigation that is required to indicate the best performance of navigation. In comparison, the Mean time and Mean length of Flight were used to illustrate the comprehensive performance of all experiments (in our case this is 20) for each strategy.

It is clear from results that the white line significantly affects the performance of DroNet navigation especially the Ratio of Successful Flight which improves from 0% to 35%. The reason is related to the DroNet training which was only based on Udacity's outdoor self-driving datasets [28] causing lane-dependence. In contrast to DroNet, CNN-based Distance Estimation and Frontier-based navigation have higher and more constant Ratio of Successful Flight between two environments, that is average of 50% and 100% respectively which shows better indoor generalization capability and stability. Notably, the ground truth 2D position ξ_k of DroNet and CNN-based Distance Estimation for navigation is only responsible for termination by checking UAV's position whether within or out of the $LandingZone$, that is unrelevant to control the UAV's flight. In contrast, the Frontier-based navigation which requires the ground truth 2D position to reconstruct 2D maps for path planning. Therefore, the failures of DroNet and CNN-based Distance Estimation for navigation are completely related to their CNN models. The Frontier-based navigation obtaining constant

[3] https://microsoft.github.io/AirSim/simple_flight/.

successful ratio and trajectories shows the reliable ground truth 2D position for stable path planning and the only failure relies on the minor error of localization.

Regarding the best performance of navigation, CNN-based Distance Estimation and Frontier-based navigation respectively obtain 89.4 s and 97.2 s of the Mean time of Successful Flight in Sim Envs 1 and 2. In comparison, the longest Mean time of Successful Flight of Frontier-based navigation (around 380 s) shows the drawback of the required longer processing time for map construction. Another aspect of the best performance of navigation, the Mean Length of Successful Flight, indicates that the Frontier-based navigation gains the average 25.7 m of the shortest path length since it has unvisited frontiers exploration and DWA local planner to avoid circuitous routes compared to other strategies.

In order to quantitatively evaluate the difference between the best and the most comprehensive performances, we calculate the ratios ($MTF/MTSF$ and $MLF/MLSF$) based on the results in Sim Envs 2. The ratios show that the Frontier-based navigation has the smallest gap with $MTF/MTSF$ of 96.7% and $MLF/MLSF$ of 96.5% while the DroNet navigation has the largest difference with only $MTF/MTSF$ of 76.4% and $MLF/MLSF$ of 73.7%. As considering to trade off the Ratio of Successful Flight with the required time and path length, CNN-based Distance Estimation for navigation shows the balanced and competitive performance.

Fig. 25 respectively shows the successful and failure trajectories for each strategy in two environments. The prime failures of DroNet and CNN-based Distance Estimation for Navigation are the collisions at the entrance of the 90° angles since the small angle between the UAV and the corner entrance results to difficult feature extraction for the similar near and distant backgrounds on the walls. It is can be seen that the Frontier-based Navigation has more consistent trajectories because of the higher accuracy of LiDAR data and the ground truth 2D position.

6. Challenges and opportunities

Previous sections investigate in detail the diversity of complete navigation strategies and their corresponding sub-components along with the uniform performance evaluations. Nevertheless, there are still diversity challenges and opportunities for the UAV complete navigation strategies including:

Public Datasets: The rising interests in supervised learning for UAV autonomous navigation require reliable samples and labels. However, our previous publication [29] indicated the lack of real-world and open-source navigation datasets especially the indoor scenarios datasets. As a result, there is a need for the community to propose as much public indoor navigation datasets with corresponding labels as possible.

Sample Labels: The onerous labeling works draw attention to self-supervised learning that requires samples with limited or no labels. For example, the novel contrastive learning methods [40,165] are expected to gain the common representations based on pre-training without labels for better generalization in different environments, also speed up the training processes for down-stream tasks.

Hardware Acceleration: Most GPS-denied autonomous navigation containing Map-building task such as Frontier-based Navigation which requires intensive on-board computational resources leads to spending more time than the Mapless navigation. For the purposes of reducing computational and memory requirements and improving real-time capability, there is a need to use software-defined hardware accelerator such as FPGA to speed up the processes of map construction.

Sim-to-Real Generalization: Environments simulators like Gazebo [116] and Unreal Engine [164] combining with Flight simulators such as ROS [166] and AirSim [18] are able to collect data since they provide various real-like environments, and also for evaluation to reduce the cost of UAV replacement parts because of crashes. However, the conditions of the virtual environments provided by simulators such as wind disturbance, illumination changes and moving objects have considerable gaps between real-world environments. Therefore, a potential improvement is to reduce the Sim-to-Real generalization limitations by developing high-fidelity simulators with various environmental conditions and constraints.

7. Conclusion

In this paper, we review the complete navigation strategies and classify them by proposing the taxonomy based on their common features. We first divide the strategies into Mapless and Map-based according to map usage. The Mapless navigation methods are further classified into **Integrated Approaches**, **Direct Approaches** and **Indirect Approaches** based on whether independent Control Unit and Distance Estimation module are used. The Map-based navigation is further classified into **known map/spaces** and **Map-building** depending on whether mapping and localization module exist. The detailed study and analysis were provided for each strategy including technologies followed by theoretical and simulated performance evaluation.

Three Evaluation Metrics (i.e., Path Length, Deviation Rate and Exploration Efficiency) have been proposed to evaluate the theoretical performances for their corresponding objectives (i.e., Random Mapless Navigation, Map-based Navigation Generating optimal Path to Destination and Self-Exploration). The theoretical evaluation draws the conclusion for the Random Mapless Navigation where the Integrated Supervised Learning achieved competitive flight sustainability with longer path length. The waypoints along trajectories obviously help to calibrate UAV's status for generating shortest path and the RF-based localization has larger covered size compared to LiDAR-SLAM. As for Self-Exploration, the strategies of Frontier-based exploration with Multi-Resolution maps benefit the Exploration Efficiency.

We also select three representative strategies (i.e., DroNet, CNN-based Distance Estimation and Frontier-based navigation) for simulated performance evaluation in two reality-like indoor environments. The results show that the DroNet navigation performs the shortest Mean time for Successful Flight but has limitation of generalization capability since it lacks of training samples with indoor scenes. CNN-based Distance Estimation for navigation is a good way to trade off the performances especially the time-costs and path length with generalization and stability in different environments. The Frontier-based navigation obtains the most consistent performance and trajectory along with the shortest path length to the destination but requires to speed up the processes of map reconstruction.

Finally, analysis of the current complete UAV autonomous navigation strategies identified the following challenges and opportunities: the necessity for more open-source navigation datasets with reliable labels, the potential research of using the state-of-the-art self-learning algorithms based on limited or without labels, the need to develop hardware accelerators and high-fidelity simulator.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgment

The authors gratefully acknowledge financial support from China Scholarship Council (CSC).

References

[1] F.G. Costa, J. Ueyama, T. Braun, G. Pessin, F.S. Osório, P.A. Vargas, The use of unmanned aerial vehicles and wireless sensor network in agricultural applications, in: 2012 IEEE International Geoscience and Remote Sensing Symposium, 2012, pp. 5045–5048, http://dx.doi.org/10.1109/IGARSS.2012.6352477.

[2] Y. Qu, L. Jiang, X. Guo, Moving vehicle detection with convolutional networks in UAV videos, in: 2016 2nd International Conference on Control, Automation and Robotics (ICCAR), 2016, pp. 225–229, http://dx.doi.org/10.1109/ICCAR.2016.7486730.

[3] M. Nieuwenhuisen, J. Quenzel, M. Beul, D. Droeschel, S. Houben, S. Behnke, ChimneySpector: Autonomous MAV-based indoor chimney inspection employing 3D laser localization and textured surface reconstruction, in: 2017 International Conference on Unmanned Aircraft Systems (ICUAS), 2017, pp. 278–285, http://dx.doi.org/10.1109/ICUAS.2017.7991427.

[4] P. Jiang, Z. Feng, Y. Cheng, Y. Ji, J. Zhu, X. Wang, F. Tian, J. Baruch, F. Hu, A mosaic of eyes, IEEE Robot. Autom. Mag. 18 (3) (2011) 104–113, http://dx.doi.org/10.1109/MRA.2011.942119.

[5] N. Gyagenda, J.V. Hatilima, H. Roth, V. Zhmud, A review of GNSS-independent UAV navigation techniques, Robot. Auton. Syst. 152 (2022) 104069, http://dx.doi.org/10.1016/j.robot.2022.104069.

[6] Y. Lu, Z. Xue, G.-S. Xia, L. Zhang, A survey on vision-based UAV navigation, Geo-Spatial Inf. Sci. 21 (1) (2018) 21–32, http://dx.doi.org/10.1080/10095020.2017.1420509.

[7] M.Y. Arafat, M.M. Alam, S. Moh, Vision-based navigation techniques for unmanned aerial vehicles: Review and challenges, Drones 7 (2) (2023) http://dx.doi.org/10.3390/drones7020089, URL https://www.mdpi.com/2504-446X/7/2/89.

[8] G. Balamurugan, J. Valarmathi, V.P.S. Naidu, Survey on UAV navigation in GPS denied environments, in: 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), 2016, pp. 198–204, http://dx.doi.org/10.1109/SCOPES.2016.7955787.

[9] H. Chao, Y. Gu, M. Napolitano, A survey of optical flow techniques for UAV navigation applications, in: 2013 International Conference on Unmanned Aircraft Systems (ICUAS), 2013, pp. 710–716, http://dx.doi.org/10.1109/ICUAS.2013.6564752.

[10] S. Rezwan, W. Choi, Artificial intelligence approaches for UAV navigation: Recent advances and future challenges, IEEE Access 10 (2022) 26320–26339, http://dx.doi.org/10.1109/ACCESS.2022.3157626.

[11] L. Yang, J. Qi, J. Xiao, X. Yong, A literature review of UAV 3D path planning, in: Proceeding of the 11th World Congress on Intelligent Control and Automation, 2014, pp. 2376–2381, http://dx.doi.org/10.1109/WCICA.2014.7053093.

[12] S.B. Amarat, P. Zong, 3D path planning, routing algorithms and routing protocols for unmanned air vehicles: a review, Aircr. Eng. Aerosp. Technol. (2019) http://dx.doi.org/10.1108/AEAT-01-2019-0023.

[13] M. Radmanesh, M. Kumar, P.H. Guentert, M. Sarim, Overview of path-planning and obstacle avoidance algorithms for UAVs: a comparative study, Unmanned Syst. 6 (02) (2018) 95–118, http://dx.doi.org/10.1142/S2301385018400022.

[14] Y. Zhao, Z. Zheng, Y. Liu, Survey on computational-intelligence-based UAV path planning, Knowl.-Based Syst. 158 (2018) 54–64, http://dx.doi.org/10.1016/j.knosys.2018.05.033.

[15] S. Poudel, M.Y. Arafat, S. Moh, Bio-inspired optimization-based path planning algorithms in unmanned aerial vehicles: A survey, Sensors 23 (6) (2023) http://dx.doi.org/10.3390/s23063051, URL https://www.mdpi.com/1424-8220/23/6/3051.

[16] A. Couturier, M.A. Akhloufi, A review on absolute visual localization for UAV, Robot. Auton. Syst. 135 (2021) 103666, http://dx.doi.org/10.1016/j.robot.2020.103666.

[17] M.D.C. Pérez Rubio, D. Gualda Gómez, J.D. Vicente Ranera, J.M. Villadangos Carrizo, J. Ureña Ureña, et al., Review of UAV positioning in indoor environments and new proposal based on US measurements, 2019.

[18] S. Shah, D. Dey, C. Lovett, A. Kapoor, Airsim: High-fidelity visual and physical simulation for autonomous vehicles, in: Field and Service Robotics, Springer, 2018, pp. 621–635, http://dx.doi.org/10.1007/978-3-319-67361-5_40.

[19] Y. Cheng, P. Jiang, Y.-F. Hu, A snake-based scheme for path planning and control with constraints by distributed visual sensors, Robotica 32 (4) (2014) 477–499, http://dx.doi.org/10.1017/S0263574713000805.

[20] F. Bonin-Font, A. Ortiz, G. Oliver, Visual navigation for mobile robots: A survey, J. Intell. Robot. Syst. 53 (3) (2008) 263–296, http://dx.doi.org/10.1007/s10846-008-9235-4.

[21] M.S. Güzel, Autonomous vehicle navigation using vision and mapless strategies: a survey, Adv. Mech. Eng. 5 (2013) 234747, http://dx.doi.org/10.1155/2013/234747.

[22] G. Desouza, A. Kak, Vision for mobile robot navigation: a survey, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2) (2002) 237–267, http://dx.doi.org/10.1109/34.982903.

[23] J. Leonard, H. Durrant-Whyte, Simultaneous map building and localization for an autonomous mobile robot, in: Proceedings IROS '91:IEEE/RSJ International Workshop on Intelligent Robots and Systems '91, Vol. 3, 1991, pp. 1442–1447, http://dx.doi.org/10.1109/IROS.1991.174711.

[24] A. Carrio, C. Sampedro, A. Rodriguez-Ramos, P. Campoy, A review of deep learning methods and applications for unmanned aerial vehicles, J. Sensors 2017 (2017) http://dx.doi.org/10.1155/2017/3296874.

[25] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[26] A. Loquercio, A.I. Maqueda, C.R. del Blanco, D. Scaramuzza, DroNet: Learning to fly by driving, IEEE Robot. Autom. Lett. 3 (2) (2018) 1088–1095, http://dx.doi.org/10.1109/LRA.2018.2795643.

[27] D. Palossi, F. Conti, L. Benini, An open source and open hardware deep learning-powered visual navigation engine for autonomous nano-UAVs, in: 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS), 2019, pp. 604–611, http://dx.doi.org/10.1109/DCOSS.2019.00111.

[28] Udacity, An open source self-driving car, 2016.

[29] Y. Chang, Y. Cheng, J. Murray, S. Huang, G. Shi, The HDIN dataset: A real-world indoor UAV dataset with multi-task labels for visual-based navigation, Drones 6 (8) (2022) http://dx.doi.org/10.3390/drones6080202, URL https://www.mdpi.com/2504-446X/6/8/202.

[30] F. AlMahamid, K. Grolinger, Autonomous Unmanned Aerial Vehicle navigation using Reinforcement learning: A systematic review, Eng. Appl. Artif. Intell. 115 (2022) 105321, http://dx.doi.org/10.1016/j.engappai.2022.105321.

[31] A. Singla, S. Padakandla, S. Bhatnagar, Memory-based deep reinforcement learning for obstacle avoidance in UAV with limited environment knowledge, IEEE Trans. Intell. Transp. Syst. 22 (1) (2021) 107–118, http://dx.doi.org/10.1109/TITS.2019.2954952.

[32] P. Isola, J.-Y. Zhu, T. Zhou, A.A. Efros, Image-to-image translation with conditional adversarial networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[33] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780, http://dx.doi.org/10.1162/neco.1997.9.8.1735.

[34] Z. Xue, T. Gonsalves, Monocular vision guided deep reinforcement learning UAV systems with representation learning perception, Connect. Sci. (2023) 1–22, http://dx.doi.org/10.1080/09540091.2023.2183828.

[35] R. Bonatti, R. Madaan, V. Vineet, S. Scherer, A. Kapoor, Learning visuomotor policies for aerial navigation using cross-modal representations, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020, pp. 1637–1644, http://dx.doi.org/10.1109/IROS45743.2020.9341049.

[36] S. Vemprala, S. Mian, A. Kapoor, Representation learning for event-based visuomotor policies, in: M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J.W. Vaughan (Eds.), Advances in Neural Information Processing Systems, Vol. 34, Curran Associates, Inc., 2021, pp. 4712–4724, URL https://proceedings.neurips.cc/paper/2021/file/251c5ffd6b62cc21c446c963c76cf214-Paper.pdf.

[37] H. Moon, J. Martinez-Carranza, T. Cieslewski, M. Faessler, D. Falanga, A. Simovic, D. Scaramuzza, S. Li, M. Ozo, C. De Wagter, et al., Challenges and implemented technologies used in autonomous drone racing, Intell. Serv. Robot. 12 (2019) 137–148, http://dx.doi.org/10.1007/s11370-018-00271-6.

[38] R. Madaan, N. Gyde, S. Vemprala, M. Brown, K. Nagami, T. Taubner, E. Cristofalo, D. Scaramuzza, M. Schwager, A. Kapoor, AirSim drone racing lab, in: H.J. Escalante, R. Hadsell (Eds.), Proceedings of the NeurIPS 2019 Competition and Demonstration Track, in: Proceedings of Machine Learning Research, 123, PMLR, 2020, pp. 177–191, URL https://proceedings.mlr.press/v123/madaan20a.html.

[39] J. Fu, Y. Song, Y. Wu, F. Yu, D. Scaramuzza, Learning deep sensorimotor policies for vision-based autonomous drone racing, 2022, arXiv preprint arXiv:2210.14985.

[40] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar, B. Piot, k. kavukcuoglu, R. Munos, M. Valko, Bootstrap your own latent - A new approach to self-supervised learning, in: H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, H. Lin (Eds.), Advances in Neural Information Processing Systems, Vol. 33, Curran Associates, Inc., 2020, pp. 21271–21284, URL https://proceedings.neurips.cc/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf.

[41] Y. Song, M. Steinweg, E. Kaufmann, D. Scaramuzza, Autonomous drone racing with deep reinforcement learning, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021, pp. 1205–1212, http://dx.doi.org/10.1109/IROS51168.2021.9636053.

[42] R.P. Padhy, S. Verma, S. Ahmad, S.K. Choudhury, P.K. Sa, Deep neural network for autonomous UAV navigation in indoor corridor environments, Procedia Comput. Sci. 133 (2018) 643–650, http://dx.doi.org/10.1016/j.procs.2018.07.099, International Conference on Robotics and Smart Manufacturing (RoSMa2018).

[43] G. Huang, Z. Liu, L. van der Maaten, K.Q. Weinberger, Densely connected convolutional networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[44] P. Chhikara, R. Tekchandani, N. Kumar, V. Chamola, M. Guizani, DCNN-GA: A deep neural net architecture for navigation of UAV in indoor environment, IEEE Internet Things J. 8 (6) (2021) 4448–4460, http://dx.doi.org/10.1109/JIOT.2020.3027095.

[45] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 2015, arXiv:1409.1556.

[46] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[47] S. Jung, S. Hwang, H. Shin, D.H. Shim, Perception, guidance, and navigation for indoor autonomous drone racing using deep learning, IEEE Robot. Autom. Lett. 3 (3) (2018) 2539–2544, http://dx.doi.org/10.1109/LRA.2018.2808368.

[48] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in: F. Pereira, C.J.C. Burges, L. Bottou, K.Q. Weinberger (Eds.), Advances in Neural Information Processing Systems, Vol. 25, Curran Associates, Inc., 2012.

[49] M. Cao, F. Tang, P. Ji, F. Ma, Improved real-time semantic segmentation network model for crop vision navigation line detection, Front. Plant Sci. 13 (2022) http://dx.doi.org/10.3389/fpls.2022.898131.

[50] A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, Enet: A deep neural network architecture for real-time semantic segmentation, 2016, http://dx.doi.org/10.48550/ARXIV.1606.02147.

[51] L. Bartolomei, L. Teixeira, M. Chli, Semantic-aware active perception for UAVs using deep reinforcement learning, in: 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021, pp. 3101–3108, http://dx.doi.org/10.1109/IROS51168.2021.9635893.

[52] F. Raudies, Optic flow, Scholarpedia 8 (7) (2013) 30724, http://dx.doi.org/10.4249/scholarpedia.30724, revision #149632.

[53] D.-W. Yoo, D.-Y. Won, M.-J. Tahk, Optical flow based collision avoidance of multi-rotor uavs in urban environments, Int. J. Aeronaut. Space Sci. 12 (3) (2011) 252–259, http://dx.doi.org/10.5139/IJASS.2011.12.3.252.

[54] K. Souhila, A. Karim, Optical flow based robot obstacle avoidance, Int. J. Adv. Robot. Syst. 4 (1) (2007) 2, http://dx.doi.org/10.5772/5715.

[55] P. Agrawal, A. Ratnoo, D. Ghose, Inverse optical flow based guidance for UAV navigation through urban canyons, Aerosp. Sci. Technol. 68 (2017) 163–178, http://dx.doi.org/10.1016/j.ast.2017.05.012.

[56] S. Temizer, Optical flow based robot navigation, 2009.

[57] A. Kouris, C.-S. Bouganis, Learning to fly by myself: A self-supervised CNN-based approach for autonomous navigation, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018, pp. 1–9, http://dx.doi.org/10.1109/IROS.2018.8594204.

[58] J. Hu, Y. Zhang, T. Okatani, Visualization of convolutional neural networks for monocular depth estimation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

[59] P. Chakravarty, K. Kelchtermans, T. Roussel, S. Wellens, T. Tuytelaars, L. Van Eycken, CNN-based single image obstacle avoidance on a quadrotor, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 6369–6374, http://dx.doi.org/10.1109/ICRA.2017.7989752.

[60] D. Eigen, C. Puhrsch, R. Fergus, Depth map prediction from a single image using a multi-scale deep network, 2014, arXiv:1406.2283.

[61] X. Yang, H. Luo, Y. Wu, Y. Gao, C. Liao, K.-T. Cheng, Reactive obstacle avoidance of monocular quadrotors with online adapted depth prediction network, Neurocomputing 325 (2019) 142–158, http://dx.doi.org/10.1016/j.neucom.2018.10.019.

[62] X. Yang, J. Chen, Y. Dang, H. Luo, Y. Tang, C. Liao, P. Chen, K.-T. Cheng, Fast depth prediction and obstacle avoidance on a monocular drone using probabilistic convolutional neural network, IEEE Trans. Intell. Transp. Syst. 22 (1) (2021) 156–167, http://dx.doi.org/10.1109/TITS.2019.2955598.

[63] A. Garcia, S.S. Mittal, E. Kiewra, K. Ghose, A convolutional neural network feature detection approach to autonomous quadrotor indoor navigation, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019, pp. 74–81, http://dx.doi.org/10.1109/IROS40897.2019.8968222.

[64] J. Redmon, A. Farhadi, YOLOv3: An incremental improvement, 2018, arXiv:1804.02767.

[65] S. Zingg, D. Scaramuzza, S. Weiss, R. Siegwart, MAV navigation through indoor corridors using optical flow, in: 2010 IEEE International Conference on Robotics and Automation, 2010, pp. 3361–3368, http://dx.doi.org/10.1109/ROBOT.2010.5509777.

[66] K. McGuire, G. de Croon, C. De Wagter, K. Tuyls, H. Kappen, Efficient optical flow and stereo vision for velocity estimation and obstacle avoidance on an autonomous pocket drone, IEEE Robot. Autom. Lett. 2 (2) (2017) 1070–1076, http://dx.doi.org/10.1109/LRA.2017.2658940.

[67] K. McGuire, G. de Croon, C. de Wagter, B. Remes, K. Tuyls, H. Kappen, Local histogram matching for efficient optical flow computation applied to velocity estimation on pocket drones, in: 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 3255–3260, http://dx.doi.org/10.1109/ICRA.2016.7487496.

[68] M. Guanglei, P. Haibing, The application of ultrasonic sensor in the obstacle avoidance of quad-rotor UAV, in: 2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC), 2016, pp. 976–981, http://dx.doi.org/10.1109/CGNCC.2016.7828918.

[69] K. Niwa, K. Watanabe, I. Nagai, A detection method using ultrasonic sensors for avoiding a wall collision of Quadrotors, in: 2017 IEEE International Conference on Mechatronics and Automation (ICMA), 2017, pp. 1438–1443, http://dx.doi.org/10.1109/ICMA.2017.8016028.

[70] N. Gageik, T. Müller, S. Montenegro, Obstacle Detection and Collision Avoidance Using Ultrasonic Distance Sensors for an Autonomous Quadrocopter, University of Wurzburg, Aerospace information Technologhy (germany) Wurzburg, 2012, pp. 3–23.

[71] N. Gageik, P. Benz, S. Montenegro, Obstacle detection and collision avoidance for a UAV with complementary low-cost sensors, IEEE Access 3 (2015) 599–609, http://dx.doi.org/10.1109/ACCESS.2015.2432455.

[72] R. Rambabu, M.R. Bahiki, S. Azrad, Multi-sensor fusion based uav collision avoidance system, J. Teknol. 76 (8) (2015) http://dx.doi.org/10.11113/jt.v76.5630.

[73] B. Du, S. Liu, A common obstacle avoidance module based on fuzzy algorithm for Unmanned Aerial Vehicle, in: 2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER), 2016, pp. 245–248, http://dx.doi.org/10.1109/CYBER.2016.7574830.

[74] A. Moffatt, E. Platt, B. Mondragon, A. Kwok, D. Uryeu, S. Bhandari, Obstacle detection and avoidance system for small UAVs using a LiDAR, in: 2020 International Conference on Unmanned Aircraft Systems (ICUAS), 2020, pp. 633–640, http://dx.doi.org/10.1109/ICUAS48674.2020.9213897.

[75] R.J. Wilson, Introduction To Graph Theory, Pearson Education India, 1979.

[76] A. Al-Kaff, J.M. Armingol, A. de La Escalera, A vision-based navigation system for Unmanned Aerial Vehicles (UAVs), Integr. Comput.-Aided Eng. 26 (3) (2019) 297–310, http://dx.doi.org/10.3233/ICA-190601.

[77] C. Forster, M. Pizzoli, D. Scaramuzza, SVO: Fast semi-direct monocular visual odometry, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), 2014, pp. 15–22, http://dx.doi.org/10.1109/ICRA.2014.6906584.

[78] C. Fu, A. Carrio, P. Campoy, Efficient visual odometry and mapping for Unmanned Aerial Vehicle using ARM-based stereo vision pre-processing system, in: 2015 International Conference on Unmanned Aircraft Systems (ICUAS), 2015, pp. 957–962, http://dx.doi.org/10.1109/ICUAS.2015.7152384.

[79] L. Jayatilleke, N. Zhang, Landmark-based localization for Unmanned Aerial Vehicles, in: 2013 IEEE International Systems Conference (SysCon), 2013, pp. 448–451, http://dx.doi.org/10.1109/SysCon.2013.6549921.

[80] T. Wang, Y. Zhang, C. Wang, J. Liang, H. Gao, M. Liu, Q. Guan, A. Sun, Indoor visual navigation system based on paired-landmark for small UAVs, in: 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014), 2014, pp. 1703–1708, http://dx.doi.org/10.1109/ROBIO.2014.7090580.

[81] A. Dawadee, J. Chahl, N. Nandagopal, A method for autonomous navigation of uavs using landmarks, in: AIAC16: 16th Australian International Aerospace Congress, Engineers Australia, Barton, ACT, 2015, pp. 146–152, 2015.

[82] A. Dawadee, J. Chahl, D. Nandagopal, Z. Nedic, Illumination, scale and rotation invariant algorithm for vision-based UAV navigation, Int. J. Pattern Recognit. Artif. Intell. 27 (05) (2013) 1359003, http://dx.doi.org/10.1142/S0218001413590039.

[83] L.C. Chie, Y.W. Juin, Artificial landmark-based indoor navigation system for an autonomous unmanned aerial vehicle, in: 2020 IEEE 7th International Conference on Industrial Engineering and Applications (ICIEA), 2020, pp. 756–760, http://dx.doi.org/10.1109/ICIEA49774.2020.9102082.

[84] T.T. Mac, C. Copot, A. Hernandez, R. De Keyser, Improved potential field method for unknown obstacle avoidance using UAV in indoor environment, in: 2016 IEEE 14th International Symposium on Applied Machine Intelligence and Informatics (SAMI), 2016, pp. 345–350, http://dx.doi.org/10.1109/SAMI.2016.7423032.

[85] S. Grijalva, W.G. Aguilar, Landmark-based virtual path estimation for assisted UAV FPV tele-operation with augmented reality, in: International Conference on Intelligent Robotics and Applications, Springer, 2019, pp. 688–700, http://dx.doi.org/10.1007/978-3-030-27529-7_58.

[86] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, ORB: An efficient alternative to SIFT or SURF, in: 2011 International Conference on Computer Vision, 2011, pp. 2564–2571, http://dx.doi.org/10.1109/ICCV.2011.6126544.

[87] B. Wang, S. Rathinam, R. Sharma, Landmark placement for cooperative localization and routing of unmanned vehicles, in: 2019 International Conference on Unmanned Aircraft Systems (ICUAS), 2019, pp. 33–42, http://dx.doi.org/10.1109/ICUAS.2019.8798276.

[88] R. Kapoor, S. Ramasamy, A. Gardi, R. Sabatini, UAV navigation using signals of opportunity in urban environments: A review, Energy Procedia 110 (2017) 377–383, http://dx.doi.org/10.1016/j.egypro.2017.03.156, 1st International Conference on Energy and Power, ICEP2016, 14-16 December 2016, RMIT University, Melbourne, Australia.

[89] G.J. Nunns, Y.-J. Chen, D.-K. Chang, K.-M. Liao, F.P. Tso, L. Cui, Autonomous flying WiFi access point, in: 2019 IEEE Symposium on Computers and Communications (ISCC), 2019, pp. 278–283, http://dx.doi.org/10.1109/ISCC47284.2019.8969672.

[90] J. Xu, W. Liu, F. Lang, Y. Zhang, C. Wang, Distance measurement model based on RSSI in WSN, Wirel. Sensor Netw. 2 (8) (2010) 606, http://dx.doi.org/10.4236/wsn.2010.28072.

[91] B.R. Stojkoska, J. Palikrushev, K. Trivodaliev, S. Kalajdziski, Indoor localization of unmanned aerial vehicles based on RSSI, in: IEEE EUROCON 2017 -17th International Conference on Smart Technologies, 2017, pp. 120–125, http://dx.doi.org/10.1109/EUROCON.2017.8011089.

[92] R.I. Marasigan, Y.D. Austria, J.B. Enriquez, L. Lolong Lacatan, R.M. Dellosa, Unmanned aerial vehicle indoor navigation using Wi-Fi trilateration, in: 2020 11th IEEE Control and System Graduate Research Colloquium (ICSGRC), 2020, pp. 346–351, http://dx.doi.org/10.1109/ICSGRC49013.2020.9232623.

[93] J. Khalife, K. Shamaei, Z.M. Kassas, Navigation with cellular CDMA signals—Part I: Signal modeling and software-defined receiver design, IEEE Trans. Signal Process. 66 (8) (2018) 2191–2203, http://dx.doi.org/10.1109/TSP.2018.2799167.

[94] S. Singh, P. Sujit, Landmarks based path planning for UAVs in GPS-denied areas* This work is partially funded by FCT grant SFRH/BPD/103962/2014, IFAC-PapersOnLine 49 (1) (2016) 396–400, http://dx.doi.org/10.1016/j.ifacol.2016.03.086, 4th IFAC Conference on Advances in Control and Optimization of Dynamical Systems ACODS 2016.

[95] Y. Yang, J. Khalife, J.J. Morales, Z.M. Kassas, UAV waypoint opportunistic navigation in GNSS-denied environments, IEEE Trans. Aerosp. Electron. Syst. 58 (1) (2022) 663–678, http://dx.doi.org/10.1109/TAES.2021.3103140.

[96] B.S. Ciftler, A. Tuncer, I. Guvenc, Indoor UAV navigation to a Rayleigh fading source using Q-learning, 2017, arXiv:1705.10375.

[97] M.M.U. Chowdhury, F. Erden, I. Guvenc, RSS-based Q-learning for indoor UAV navigation, in: MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM), 2019, pp. 121–126, http://dx.doi.org/10.1109/MILCOM47813.2019.9020894.

[98] S. Kulkarni, V. Chaphekar, M.M. Uddin Chowdhury, F. Erden, I. Guvenc, UAV aided search and rescue operation using reinforcement learning, in: 2020 SoutheastCon, Vol. 2, 2020, pp. 1–8, http://dx.doi.org/10.1109/SoutheastCon44009.2020.9368285.

[99] C.J.C.H. Watkins, Learning from delayed rewards (Ph.D. thesis), King's College, Cambridge United Kingdom, 1989.

[100] S. Jayasekara, A. Al-Hourani, B. Ristic, A. Skvortsov, Autonomous UAV search for an RF source in urban environments, in: 2020 14th International Conference on Signal Processing and Communication Systems (ICSPCS), 2020, pp. 1–6, http://dx.doi.org/10.1109/ICSPCS50536.2020.9310037.

[101] A.R. Khairuddin, M.S. Talib, H. Haron, Review on simultaneous localization and mapping (SLAM), in: 2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE), 2015, pp. 85–90, http://dx.doi.org/10.1109/ICCSCE.2015.7482163.

[102] R. Mur-Artal, J.M.M. Montiel, J.D. Tardós, ORB-SLAM: A versatile and accurate monocular SLAM system, IEEE Trans. Robot. 31 (5) (2015) 1147–1163, http://dx.doi.org/10.1109/TRO.2015.2463671.

[103] J. Engel, T. Schöps, D. Cremers, LSD-SLAM: Large-scale direct monocular SLAM, in: European Conference on Computer Vision(ECCV), Springer, 2014, pp. 834–849, http://dx.doi.org/10.1007/978-3-319-10605-2_54.

[104] WaveLab, Direct methods in visual odometry, 2016.

[105] O. Esrafilian, H.D. Taghirad, Autonomous flight and obstacle avoidance of a quadrotor by monocular SLAM, in: 2016 4th International Conference on Robotics and Mechatronics (ICROM), 2016, pp. 240–245, http://dx.doi.org/10.1109/ICRoM.2016.7886853.

[106] L. von Stumberg, V. Usenko, J. Engel, J. Stückler, D. Cremers, From monocular SLAM to autonomous drone exploration, in: 2017 European Conference on Mobile Robots (ECMR), 2017, pp. 1–8, http://dx.doi.org/10.1109/ECMR.2017.8098709.

[107] R. Mur-Artal, J.D. Tardós, ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras, IEEE Trans. Robot. 33 (5) (2017) 1255–1262, http://dx.doi.org/10.1109/TRO.2017.2705103.

[108] Z. Xu, D. Deng, K. Shimada, Autonomous UAV exploration of dynamic environments via incremental sampling and probabilistic roadmap, IEEE Robot. Autom. Lett. 6 (2) (2021) 2729–2736, http://dx.doi.org/10.1109/LRA.2021.3062008.

[109] A. Hornung, K.M. Wurm, M. Bennewitz, C. Stachniss, W. Burgard, OctoMap: An efficient probabilistic 3D mapping framework based on octrees, Auton. Robots 34 (3) (2013) 189–206, http://dx.doi.org/10.1007/s10514-012-9321-0.

[110] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, W. Burgard, An evaluation of the RGB-D SLAM system, in: 2012 IEEE International Conference on Robotics and Automation, 2012, pp. 1691–1696, http://dx.doi.org/10.1109/ICRA.2012.6225199.

[111] S. Chen, H. Chen, W. Zhou, C.Y. Wen, B. Li, End-to-end UAV simulation for visual SLAM and navigation, 2020, arXiv:2012.00298.

[112] S. Chen, C.-Y. Wen, Y. Zou, W. Chen, Stereo visual inertial pose estimation based on feedforward-feedback loops, 2020, arXiv:2007.02250.

[113] S. Leutenegger, P. Furgale, V. Rabaud, M. Chli, K. Konolige, R. Siegwart, Keyframe-based visual-inertial slam using nonlinear optimization, in: Proceedings of Robotis Science and Systems (RSS) 2013, 2013, http://dx.doi.org/10.3929/ethz-b-000236658.

[114] I. Alzugaray, L. Teixeira, M. Chli, Short-term UAV path-planning with monocular-inertial SLAM in the loop, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 2739–2746, http://dx.doi.org/10.1109/ICRA.2017.7989319.

[115] C. Debeunne, D. Vivet, A review of visual-LiDAR fusion based simultaneous localization and mapping, Sensors 20 (7) (2020) http://dx.doi.org/10.3390/s20072068.

[116] Gazebo, Gazebo environment engine, 2016.

[117] M. Faria, A.S. Ferreira, H. Pérez-Leon, I. Maza, A. Viguria, Autonomous 3D exploration of large structures using an UAV equipped with a 2D LIDAR, Sensors 19 (22) (2019) http://dx.doi.org/10.3390/s19224849.

[118] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, R. Siegwart, Receding horizon "next-best-view" planner for 3D exploration, in: 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 1462–1468, http://dx.doi.org/10.1109/ICRA.2016.7487281.

[119] C. Wang, H. Ma, W. Chen, L. Liu, M.Q.-H. Meng, Efficient autonomous exploration with incrementally built topological map in 3-D environments, IEEE Trans. Instrum. Meas. 69 (12) (2020) 9853–9865, http://dx.doi.org/10.1109/TIM.2020.3001816.

[120] A. Batinovic, T. Petrovic, A. Ivanovic, F. Petric, S. Bogdan, A multi-resolution frontier-based planner for autonomous 3D exploration, IEEE Robot. Autom. Lett. 6 (3) (2021) 4528–4535, http://dx.doi.org/10.1109/LRA.2021.3068923.

[121] W. Hess, D. Kohler, H. Rapp, D. Andor, Real-time loop closure in 2D LIDAR SLAM, in: 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 1271–1278, http://dx.doi.org/10.1109/ICRA.2016.7487258.

[122] W. Youn, H. Ko, H. Choi, I. Choi, J.-H. Baek, H. Myung, Collision-free autonomous navigation of a small UAV using low-cost sensors in GPS-denied environments, Int. J. Control Autom. Syst. 19 (2) (2021) 953–968, http://dx.doi.org/10.1007/s12555-019-0797-7.

[123] S.S. Mansouri, C. Kanellakis, D. Kominiak, G. Nikolakopoulos, Deploying MAVs for autonomous navigation in dark underground mine environments, Robot. Auton. Syst. 126 (2020) 103472, http://dx.doi.org/10.1016/j.robot.2020.103472.

[124] B. Yamauchi, A frontier-based approach for autonomous exploration, in: Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. 'Towards New Computational Principles for Robotics and Automation', 1997, pp. 146–151, http://dx.doi.org/10.1109/CIRA.1997.613851.

[125] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (5) (2002) 603–619, http://dx.doi.org/10.1109/34.1000236.

[126] K. Yang, S. Keat Gan, S. Sukkarieh, A Gaussian process-based RRT planner for the exploration of an unknown and cluttered environment with a UAV, Adv. Robot. 27 (6) (2013) 431–443, http://dx.doi.org/10.1080/01691864.2013.756386.

[127] B. Grocholsky, Information-Theoretic Control of Multiple Sensor Platforms (Ph.D. thesis), University of Sydney. Department of Aerospace, Mechatronic and Mechanical Engineering, 2002.

[128] M. Golabi, S. Ghambari, J. Lepagnot, L. Jourdan, M. Brévilliers, L. Idoumghar, Bypassing or flying above the obstacles? A novel multi-objective UAV path planning problem, in: 2020 IEEE Congress on Evolutionary Computation (CEC), 2020, pp. 1–8, http://dx.doi.org/10.1109/CEC48606.2020.9185695.

[129] E.W. Dijkstra, et al., A note on two problems in connexion with graphs, Numer. Math. 1 (1) (1959) 269–271, http://dx.doi.org/10.1007/BF01386390.

[130] P.E. Hart, N.J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, IEEE Trans. Syst. Sci. Cybern. 4 (2) (1968) 100–107, http://dx.doi.org/10.1109/TSSC.1968.300136.

[131] K. Daniel, A. Nash, S. Koenig, A. Felner, Theta*: Any-angle path planning on grids, J. Artificial Intelligence Res. 39 (2010) 533–579, http://dx.doi.org/10.1613/jair.2994.

[132] A. Nash, S. Koenig, C. Tovey, Lazy Theta*: Any-angle path planning and path length analysis in 3D, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 24, 2010, pp. 147–154.

[133] D. Harabor, A. Grastien, Online graph pruning for pathfinding on grid maps, Proc. AAAI Conf. Artif. Intell. 25 (1) (2011) 1114–1119, URL https://ojs.aaai.org/index.php/AAAI/article/view/7994.

[134] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, in: Proceedings. 1985 IEEE International Conference on Robotics and Automation, Vol. 2, 1985, pp. 500–505, http://dx.doi.org/10.1109/ROBOT.1985.1087247.

[135] S.M. Lavalle, J.J. Kuffner Jr., Rapidly-exploring random trees: Progress and prospects, in: Algorithmic and Computational Robotics: New Directions, 2000, pp. 293–308.

[136] L. Kavraki, P. Svestka, J.-C. Latombe, M. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Trans. Robot. Autom. 12 (4) (1996) 566–580, http://dx.doi.org/10.1109/70.508439.

[137] P.S. Blaer, Robot path planning using generalized voronoi diagrams, 2009.

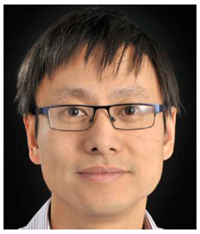
[138] P. Maini, P.B. Sujit, Path planning for a UAV with kinematic constraints in the presence of polygonal obstacles, in: 2016 International Conference on Unmanned Aircraft Systems (ICUAS), 2016, pp. 62–67, http://dx.doi.org/10.1109/ICUAS.2016.7502625.

[139] L.E. Dubins, On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents, Am. J. Math. 79 (3) (1957) 497–516, http://dx.doi.org/10.2307/2372560.

[140] Q. Feng, J. Gao, X. Deng, Path planner for UAVs navigation based on A* algorithm incorporating intersection, in: 2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC), 2016, pp. 2275–2279, http://dx.doi.org/10.1109/CGNCC.2016.7829147.

[141] D. Harabor, A. Grastien, Improving jump point search, in: Proceedings of the International Conference on Automated Planning and Scheduling, Vol. 24, 2014.
[142] H. Chen, P. Lu, Computationally efficient obstacle avoidance trajectory planner for UAVs based on heuristic angular search method, 2020, arXiv:2003.06136.
[143] H. Miao, Y. Wang, Optical flow based obstacle avoidance and path planning for quadrotor flight, in: Chinese Intelligent Automation Conference, Springer, 2017, pp. 631–638, http://dx.doi.org/10.1007/978-981-10-6445-6_69.
[144] L. Lifen, S. Ruoxin, L. Shuandao, W. Jiang, Path planning for UAVS based on improved artificial potential field method through changing the repulsive potential function, in: 2016 IEEE Chinese Guidance, Navigation and Control Conference (CGNCC), 2016, pp. 2011–2015, http://dx.doi.org/10.1109/CGNCC.2016.7829099.
[145] W. Zeng, R.L. Church, Finding shortest paths on real road networks: the case for A, Int. J. Geograph. Inf. Sci. 23 (4) (2009) 531–543, http://dx.doi.org/10.1080/13658810801949850.
[146] F. Islam, J. Nasir, U. Malik, Y. Ayaz, O. Hasan, RRT*-Smart: Rapid convergence implementation of RRT* towards optimal solution, in: 2012 IEEE International Conference on Mechatronics and Automation, 2012, pp. 1651–1656, http://dx.doi.org/10.1109/ICMA.2012.6284384.
[147] J. Ge, F. Sun, C. Liu, RRT-GD: An efficient rapidly-exploring random tree approach with goal directionality for redundant manipulator path planning, in: 2016 IEEE International Conference on Robotics and Biomimetics (ROBIO), 2016, pp. 1983–1988, http://dx.doi.org/10.1109/ROBIO.2016.7866620.
[148] B. Grüter, D. Seiferth, M. Bittner, F. Holzapfel, Emergency flight planning using voronoi diagrams, in: AIAA Scitech 2019 Forum, 2019, pp. 1056–1068, http://dx.doi.org/10.2514/6.2019-1056.
[149] H. van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double Q-learning, Proc. AAAI Conf. Artif. Intell. 30 (1) (2016).
[150] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. B 26 (1) (1996) 29–41, http://dx.doi.org/10.1109/3477.484436.
[151] C. Huang, Y. Lan, Y. Liu, W. Zhou, H. Pei, L. Yang, Y. Cheng, Y. Hao, Y. Peng, A new dynamic path planning approach for unmanned aerial vehicles, Complexity 2018 (2018) http://dx.doi.org/10.1155/2018/8420294.
[152] O. Kramer, Genetic algorithms, in: Genetic Algorithm Essentials, Springer, 2017, pp. 11–19, http://dx.doi.org/10.1007/978-3-319-52156-5_2.
[153] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of ICNN'95 - International Conference on Neural Networks, Vol. 4, 1995, pp. 1942–1948, http://dx.doi.org/10.1109/ICNN.1995.488968, vol.4.
[154] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 2018.
[155] L. He, N. Aouf, J.F. Whidborne, B. Song, Integrated moment-based LGMD and deep reinforcement learning for UAV obstacle avoidance, in: 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 7491–7497, http://dx.doi.org/10.1109/ICRA40945.2020.9197152.
[156] M. Theile, H. Bayerlein, R. Nai, D. Gesbert, M. Caccamo, UAV path planning using global and local map information with deep reinforcement learning, 2021, arXiv:2010.06917.
[157] U. Cekmez, M. Ozsiginan, O.K. Sahingoz, Multi colony ant optimization for UAV path planning with obstacle avoidance, in: 2016 International Conference on Unmanned Aircraft Systems (ICUAS), 2016, pp. 47–52, http://dx.doi.org/10.1109/ICUAS.2016.7502621.
[158] A. Sonmez, E. Kocyigit, E. Kugu, Optimal path planning for UAVs using Genetic Algorithm, in: 2015 International Conference on Unmanned Aircraft Systems (ICUAS), 2015, pp. 50–55, http://dx.doi.org/10.1109/ICUAS.2015.7152274.
[159] A. Mirshamsi, S. Godio, A. Nobakhti, S. Primatesta, F. Dovis, G. Guglieri, A 3D path planning algorithm based on PSO for autonomous UAVs navigation, in: International Conference on Bioinspired Methods and their Applications, Springer, 2020, pp. 268–280, http://dx.doi.org/10.1007/978-3-030-63710-1_21.
[160] Y. Cheng, Mean shift, mode seeking, and clustering, IEEE Trans. Pattern Anal. Mach. Intell. 17 (8) (1995) 790–799, http://dx.doi.org/10.1109/34.400568.
[161] J. Minguez, L. Montano, O. Khatib, Reactive collision avoidance for navigation with dynamic constraints, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 1, 2002, pp. 588–594, http://dx.doi.org/10.1109/IRDS.2002.1041455, vol.1.
[162] D. Fox, W. Burgard, S. Thrun, The dynamic window approach to collision avoidance, IEEE Robot. Autom. Mag. 4 (1) (1997) 23–33, http://dx.doi.org/10.1109/100.580977.
[163] A. Topiwala, P. Inani, A. Kathpal, Frontier based exploration for autonomous robot, 2018, arXiv:1806.03581.
[164] E. Games, Epic games unreal engine home page.
[165] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, A simple framework for contrastive learning of visual representations, in: H.D. III, A. Singh (Eds.), Proceedings of the 37th International Conference on Machine Learning, in: Proceedings of Machine Learning Research, 119, PMLR, 2020, pp. 1597–1607, URL https://proceedings.mlr.press/v119/chen20j.html.
[166] R.O. System, ROS home page.

Yingxiu Chang received his B.Eng and a M.Eng in Electronics Science and Technology from Anhui University and in Integrated Circuits from Peking University (China), respectively. He is currently a Ph.D. candidate in Computer Science and Technology at the University of Hull, United Kingdom. He has interests in unmanned aerial vehicles, computer vision, autonomous navigation and exploration and embedded accelerator for neural networks.



Yongqiang Cheng is currently a Professor with the Faculty of Technology at the University of Sunderland. His research interests include digital healthcare technologies, AI, UAV, control theory and applications, embedded system, secure communication, and data mining.

Umar Manzoor is currently an Associate Professor with the Faculty of Technology at the University of Sunderland. His research interests include machine learning, AI, Natural language processing, digital healthcare applications, simulation and modelling, multi-agent systems and data mining.

John Murray is Professor of Robotics and Autonomous Systems and the Academic Dean for the Faculty of Technology at the University of Sunderland. His academic experience spans 20 years in various posts and institutions. Professor Murray holds a Ph.D. in Computational Robotics and Neuroscience from the University of Sunderland, UK. His interests are in Artificial Intelligence, Human Robot Interaction, Machine Learning and Computer Vision applied to Autonomous and Mobile Systems.