Computer Vision for Autonomous UAV Flight Safety: An Overview and a Vision-based Safe Landing Pipeline Example

EFSTRATIOS KAKALETSIS, CHARALAMPOS SYMEONIDIS, MARIA TZELEPI, IOANNIS MADEMLIS, ANASTASIOS TEFAS, NIKOS NIKOLAIDIS, and IOANNIS PITAS *, Aristotle University of Thessaloniki, Department of Informatics, Greece

Recent years have seen an unprecedented spread of Unmanned Aerial Vehicles (UAVs, or "drones"), which are highly useful for both civilian and military applications. Flight safety is a crucial issue in UAV navigation, having to ensure accurate compliance with recently legislated rules and regulations. The emerging use of autonomous drones and UAV swarms raises additional issues, making it necessary to transfuse safety- and regulations-awareness to relevant algorithms and architectures. Computer vision plays a pivotal role in such autonomous functionalities. Although the main aspects of autonomous UAV technologies (e.g., path planning, navigation control, landing control, mapping and localization, target detection/tracking) are already mature and well-covered, ensuring safe flying in the vicinity of crowds, avoidance of passing over persons or guaranteed emergency landing capabilities in case of malfunctions, are generally treated as an afterthought when designing autonomous UAV platforms for unstructured environments. This fact is reflected in the fragmentary coverage of the above issues in current literature. This overview attempts to remedy this situation, from the point of view of computer vision. It examines the field from multiple aspects, including regulations across the world and relevant current technologies. Finally, since very few attempts have been made so far towards a complete UAV safety flight and landing pipeline, an example computer vision-based UAV flight safety pipeline is introduced, taking into account all issues present in current autonomous drones. The content is relevant to any kind of autonomous drone flight (e.g., for movie/TV production, news-gathering, search and rescue, surveillance, inspection, mapping, wildlife monitoring, crowd monitoring/management, etc.), making this a topic of broad interest.

CCS Concepts: • **Computing methodologies** → **Vision for robotics**; *Image processing*; Neural networks.

Additional Key Words and Phrases: UAV flight Safety, crowd/person detection, landing site detection, path planning, obstacle avoidance, semantic mapping

Efstratios Kakaletsis, Charalampos Symeonidis, Maria Tzelepi, Ioannis Mademlis, Anastasios Tefas, Nikos Nikolaidis, and Ioannis Pitas . . Computer Vision for Autonomous UAV Flight Safety: An Overview and a Vision-based Safe Landing Pipeline Example.

1 INTRODUCTION

Technological progress has led to an unprecedented spread of Unmanned Aerial Vehicles (UAVs, or "drones") during the last decade. They have proven useful for many civilian and military applications, such as search and rescue operations, surveillance, inspection, mapping [96], wildlife monitoring, crowd monitoring/management, as well as in aerial media production [126], [79], [82], [66], [106]. Flight safety is a crucial issue in UAV navigation, having to ensure accurate compliance with recently legislated rules and regulations.

*The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement number 731667 (MULTIDRONE). This publication reflects only the authors views. The European Union is not liable for any use that may be made of the information contained therein.

Authors' address: Efstratios Kakaletsis, ekakalets@csd.auth.gr; Charalampos Symeonidis, charsyme@csd.auth.gr; Maria Tzelepi, mtzelepi@csd.auth.gr; Ioannis Mademlis, imademlis@csd.auth.gr; Anastasios Tefas, tefas@csd.auth.gr; Nikos Nikolaidis, nnik@csd.auth.gr; Ioannis Pitas, pitas@csd.auth.gr, Aristotle University of Thessaloniki, Department of Informatics, AIIA Laboratory, Thessaloniki, Greece, GR-54124.

Increasingly autonomous functionalities and massive commercialization render UAV flight safety assurance an even more pressing issue, particularly given the high rate of progress in automating various UAV operations for market products. Autonomous UAV flight safety consists of several interacting building blocks, each one accompanied by a set of related algorithms and technologies. Computer vision methods are at the forefront of this ecosystem, due to the wide availability of visual sensors and the high progress in image/video analysis during the past decade.

For instance, emergency landing in the face of critical malfunction should be an irreplaceable component of proper UAV deployment. The drone must be equipped with appropriate software, maps and operational sensors that allow it to discover any potential emergency landing sites in real-time and "on-the-fly", or off-line in the pre-flight checks, in order not to crash and get destroyed or injure people on the ground. Identifying potential landing areas, which in general should be sufficiently flat, large and unoccupied by human crowds, vegetation or buildings/cars/water, is important for normal and emergency landing alike. Both should be carried out safely, without causing damages to third parties, avoiding human crowds and destruction of human property, by having a relevant mechanism in place. The literature concerning landing site detection is multifaceted, but a common thread is the utilization of a-priori topological terrain information. Relevant methods based on on-the-fly analysis of visual input (e.g., from on-board RGB/depth cameras or LiDARs) have also surfaced over the years.

The advent of Deep Neural Networks (DNNs) for visual information analysis has significantly boosted UAV safety-related research that relies on RGB camera feed. However, this is still an area with a lot of room for further improvements. For instance, although several works utilize deep Convolutional Neural Networks (CNNs) for crowd analysis and understanding, research in the topic of crowd detection, especially on aerial images/videos, is rather limited. In recent years, deep CNNs have been established as one of the most efficient learning architectures in computer vision, accomplishing outstanding results in a plethora of relevant tasks such as object detection or tracking, semantic image segmentation, etc. The main reasons behind their success are the availability of large annotated datasets and computational power, as well as the affordability of GP-GPUs enabling massive parallelism. Crowd detection with deep learning in drone-captured images bears additional challenges (e.g., small person size, occlusions etc.) and it is a mostly uncharted territory. A first attempt utilizing state-of-the-art deep CNNs is presented in [138], where a pretrained model is finetuned for the task of crowd detection, as well as in [104], where a multitask CNN-based approach for visible crowd segmentation is adopted.

Coming from a different research direction, several approaches aiming to augment topological maps [114] with semantic information [158] [40] and high-level attributes have been proposed, allowing aerial robots to handle more expressive concepts or be deployed for more sophisticated tasks. The goal is to segment the environment into regions that have a coherent semantic meaning. Recently, the robotics community has focused on the presence of semantics in maps [108] [13] [102], to develop autonomous robots capable of understanding the semantic representation [110], [25], [93] and relationships between the objects in the environment, besides exploiting occupancy grid maps for navigation. Indeed, a 3D map representing the UAV flight environment could be enriched with semantic region annotations relating to safety, such as crowd gathering locations [63] or no-fly zones in general. This is crucial for drone path planning and navigation. Temporal coherence, which permits online use of the semantic map in various robotic tasks, graph-based map representation and metric distances within 2D or 3D maps, on either indoors or outdoors environments, constitute additional important features of modern semantic maps.

Another aspect of drone flight safety, related to crowd detection and emergency landing, is person detection and avoidance. Indeed, safe UAV landing requires that the UAV detects any individuals present around the landing area so as to avoid harming them in case of a malfunction; airspace above humans should constitute no-fly zone. However, person detection from aerial imagery is a challenging problem, mainly due to the small size of a human body observed from an aerial platform. The problem can get even more challenging when person detection is applied to top-down views, due to variations in the orientation of human silhouettes. Literature on general person detection, particularly pedestrian detection from ground platforms, is extensive, while state-of-the-art object detectors may enable specialized, high-accuracy person detection running on-board UAVs.

As far as person/obstacle avoidance is concerned, reactive methods are common; they adjust the UAV flight trajectory so that it is unobstructed by obstacles or individuals, using well-known computer vision algorithms. Similar methods based on input data from cameras (or other sensors, such as LiDAR, radar, ultrasonic, IR, etc.) are utilized commonly in GPS-denied environments, in order to offer efficient UAV navigation by exploiting either video optical flow, or Simultaneous Localization and Mapping (SLAM) output. Additionally, protection of human lives in the case of a possible drone malfunction also requires a systematic and intelligent approach for guiding a path planner in the most efficient and safe possible way while landing. This can be done either by exploiting the 3D map and UAV/camera 3D pose, or in a reactive, vision-based manner for GPS-denied environments.

This paper overviews all of the above subfields under the unifying premise of computer vision for autonomous UAV flight safety. Thus, the relevant gap in existing literature is covered and the fragmentary handling of these issues is remedied. Regulations across the world and the state-of-the-art in semantic mapping, vision-based obstacle avoidance, landing site detection, and crowd/person detection are all systematically overviewed. Additionally, path planning is briefly presented due to its great importance for ensuring UAV flight safety, particularly safe landing. Finally, due to the sparsity of attempts to treat the UAV safety issue through a complete chain of inter-operating algorithms, this paper presents a unified, computer vision-based algorithmic pipeline for UAV safe landing, relying on suitably finetuned, commercial off-the-shelf or open-source building blocks, as an operational example of the relevant technologies.

The remainder of this paper is organized as follows. In Section 2, we briefly describe the details of the imposed legislation across the world regarding the UAV deployment for a safe flight. In Section 3, we present vision-based technologies contributing to UAV flight safety, such as landing site detection, semantic annotation of 3D maps, person detection and obstacle avoidance. Path planners are also covered, since they contribute to safe autonomous landing. Subsequently, Section 4 presents the example computer vision-based UAV safe landing pipeline. Finally, conclusions are drawn from the preceding discussion and future research directions are briefly presented in Section 5.

2 CURRENT UAV FLIGHT REGULATIONS

Different flight regulations apply depending on employed UAV types and their application. The regulations in many countries impose restrictions to the UAV weight and permitted flight radius, while also defining special prerequisite conditions (e.g., licensed pilot requirements and insurance policies). An important issue is that flight restrictions vary over different countries, while professional pilot licenses and insurance policies may not be internationally valid.

UAVs are typically classified into different categories, depending on their weight. Adjusting/replacing components may impact the category classification. For instance, UAVs exceeding 2kg of weight may be required to carry emergency parachutes in some countries [6].

Flying UAVs exceeding 15kg of weight might require special license or even be prohibited [5]. Maximum drone flight altitude is typically restricted to 400ft or 500ft (120m or 150m) within several European countries [6], [5], [2], [4], [1]. Visual Line-of-Sight (LOS) should be maintained by the licensed pilot of the UAV, either physically, or using visual aids (e.g., VR goggles), while the horizontal distance between the drone and the pilot may be limited to specific meters (e.g., 500m).

In addition, due to safety considerations, outdoor UAV flight in most countries is forbidden above congested areas, crowds of people and airports, leading to permissible flight zones delineated by law (geofencing). Inherently complying with such a complex and varying web of regulations is a challenge for all autonomous UAV applications. Thankfully, current aerial path planning algorithms are able to deal with complex dynamic and kinematic constraints in real-time, resulting in nearly-optimal collision-free paths being computed on-line [153]. Thus, the challenge mainly lies in properly estimating the geofences according to current legislation.

For example, the drone flight regulation rules in UK define that drones should not be flown within 50m of people and within 150m of a crowd of over 1000 people, while in Italy it is not allowed for the drone to operate at a distance less than 50m of human crowds. In Germany, it is prohibitive for a UAV to operate at a distance of less than 100m from crowds (where an assemblage of more than 12 individuals is defined as a crowd). The term "over" refers to flight directly above any part of a person. For example, a small UAV that hovers directly above a person's head, shoulders, or extended arms or legs, would be an operation over people. Therefore, it is crucial for a UAV to be capable of detecting crowds or individuals in order to define no-fly zones and proceed to re-planning its path during flight.

UAV flight regulations for countries across the world are summarized in the supplementary material. It contains the regulatory body that drafts and then oversees the correct application of regulations, legal requirements about specific operations in conjunction with the registration policies that shall be followed for different categories of UAVs, as well as the necessary pilot licenses and flight authorisations required for drone deployment. Moreover, the possibility of operation outside the visual line of sight is discussed, in the context of flying in the vicinity of crowds, disaster/military areas, or near main roads, important facilities and state buildings, national parks, natural reserves, or above residential buildings/areas, as specified by the regulations. Finally, the altitude restrictions are also considered.

3 UAV FLIGHT SAFETY TECHNOLOGIES

Commercial UAVs regularly employ rudimentary automated functionalities for a number of basic safety-related tasks, mainly focusing on safer manual piloting and mostly relying on GPS and simple sensors. However, a multitude of more advanced computer vision-based algorithms and systems that have appeared over the years for implementing autonomous functionalities can be pivotal in assuring UAV flight and landing safety. Over the past few years, similar methods that rely on visual processing have started appearing in state-of-the-art commercial UAVs. This Section first presents a list of basic relevant functionalities commonly found in commercial vehicles (Subsection 3.1) and, subsequently, a short overview of advanced vision-based methods that can enable and/or advance such capabilities (Subsections 3.2-3.7). Finally, real-world transferability issues concerning vision-based algorithms are discussed in Subsection 3.8.

The presented algorithms have been grouped into Subsections according to the application-specific subtask each one has been designed for: a) potential landing site detection, b) semantic mapping, c) obstacle avoidance, d) person detection, or e) path planning/tracking for autonomous landing. Additionally, a more general introductory Subsection on recent semantic image segmentation methodologies is presented, since semantic segmentation can be applied for solving several of these subtasks.

In addition to the sub-task specific algorithms presented in Subsections 3.2-3.7, a very small number of safety pipelines, typically based on visual and others sensors, has appeared in the literature. Good examples are [86], detailed below in Subsection 3.3.2, and [94], discussed in 4.9.4, with the latter one presenting an autonomous vision-based system for UAVs equipped with bioradars, tasked to assist Urban Search and Rescue (USAR) operations in post-disaster environments.

3.1 State-of-the-art in Commercial UAVs

UAV manufacturers have taken great lengths to ensure that drones are a safe addition to low-altitude airspace. An overview of major safety features found in commercial vehicles is provided below [28]:

- *Geofencing*: It is a GPS-based technology that regulates a UAV's operation and automatically prevents it from flying within pre-designated sensitive locations, such as airports, prisons, nuclear power plants, etc.

Thus, UAV pilots are automatically protected from inadvertently flying in high-risk, sensitive locations without authorization. These forbidden regions are marked on-map as no-fly zones, in shapes ranging from simple circles to three-dimensional "bowties", inspired by aviation authorities safety principles.

- *Altitude Limit*: UAVs are equipped with a functionality that automatically limits the highest possible flight altitude, according to airspace authorities. These legal altitude limitations are different across regions, but have been typically defined so as to minimize risks to traditional aircraft operations.
- *Remote Identification*: This capability consists in automatically determining the location, direction, altitude and serial number of UAVs flying nearby, as well as in showing on-map the location of their pilots. These information is useful in regions close to airports or large crowd-gathering events (marathons, parades, public celebrations, etc.).
- *Obstacle Sensing*: A typical, safety-related robotics operation. Its purpose is to automatically determine any rigid obstacles in the immediate flight environment, based on sensor readings, thus permitting automated corrective maneuvers in case of pilot error. Advanced, vision-based Obstacle Sensing can be found in state-of-the-art commercial UAVs (e.g., [132]).
- *Return to Home* (RTH): Automatic navigation back to the take-off position when the battery is about to be exhausted, or when the UAV is losing contact with the ground control (also known as *home location*). Combined with Obstacle Sensing, RTH significantly improves flight safety, even in manual operation scenarios.
- *Airsense*: This is a capability present specifically in DJI UAVs. It allows the reception of remote Automatic Dependent Surveillance-Broadcast (ADS-B) telemetry signals, in order to prevent potential hazards from nearby vehicles flying at a low altitude, such as helicopters and aircrafts. By automatically alerting the UAV pilots for their approach, they provide substantially greater awareness of nearby air traffic.

3.2 Semantic Image Segmentation

Semantic segmentation consists in assigning a discrete class label (out of K pre-specified object classes) to each pixel of an input image. The field advanced significantly during the past decade, mainly by employing supervised deep Convolutional Neural Networks (CNNs) for semantic per-pixel classification from raw images. CNNs do not rely on handcrafted algorithms manually endowed with human intuitions, but only on large training datasets and a training procedure based on error back-propagation coupled with first-order optimization, such as stochastic gradient descent variants. The training procedure aims at automatically learning both the weights of the filters, so that they can extract visual concepts from the raw image content, and a suitable classifier. The initial layers of the network, after they have been trained, typically identify low-level concepts such as edges and details, whereas the final layers are able to combine low-level features so as to identify complex visual concepts. The neuronal activations of each convolutional layer constitute an order-3 tensor with two spatial dimensions and a width dimension, the latter referring to the number of layer output channels. After training, the neuronal weights of each channel are actually learned convolutional filters. Semantic segmentation CNNs are typically composed of an encoding and a decoding subnetwork, arranged in a consecutive fashion. The encoder extracts from the input semantic features with progressively lower spatial resolution, while the decoder receives the final encoder output and upsamples it. The final output of the decoder is a semantic map, having the same spatial resolution as the input and as many channels as the supported number of discrete classes.

Fully convolutional networks [75] and dilated convolutions [155] are typically used, for upsampling the computed abstract feature maps and for enlarging neuronal receptive fields, respectively. Large receptive fields significantly aid semantic segmentation by enriching local-scale image representation with task-relevant wider region semantic context, for more accurate per-pixel classification. Grasping global image context while retaining spatial detail is an important consideration in relevant research, with current DNNs attempting to explicitly capture it and properly enhance local image representation, using multi-scale [164], attention-based relational [157], or network branching approaches [154]. PSPNet (Pyramid Scene Parsing Network) [164] offers a rather good balance between speed and accuracy, scoring 82.6% and 80.2% on the public datasets PASCAL VOC 2012 and Cityscapes, respectively, when using the typical mean Intersection-over-Union (mIoU) evaluation metric and a ResNet-101 base CNN. Thus, PSPNet was selected to form the backbone of more recent real-time segmentation networks, such as ICNet [163]. Its main novelty is a PPM (Pyramid Pooling Module) decoder, able to enrich local image representation with more global context information from larger image regions of various scales. Semantic information from each image region is aggregated using global average pooling within the region, separately for each tensor channel. DeepLabV3+ [21] has a structure similar to PSPNet, but relies on the so-called ASPP module instead of the PPM; the former employs multiple dilated convolutions with different dilation factors, for achieving the same purpose as the latter. The advantage compared to PPM is that fine spatial information is not lost, as is the case with global average pooling. The method achieves a maximum mIoU accuracy of 87.8% and 82.1% on the PASCAL VOC 2012 and Cityscapes test sets, respectively, when using an Xception base CNN.

Up to now, semantic image segmentation in the context of UAV flight safety has been employed for computer vision-based landing site detection, semantic mapping and person detection. Relevant application-specific methods are described below in the following Subsections 3.3, 3.4 and 3.5, respectively.

3.3 Vision-based Outdoor Mapping for Landing Site Detection

Maps play a substantial role and are significant prerequisites for safe UAV navigation in general, but also for landing site detection specifically, since they contain all knowledge about the terrain in the flight area. A map may have been constructed a-priori and be relied upon during flight under the assumption of negligible changes in the environment since the map was originally obtained. Of course, vision-based methods that dynamically update the existing map during flight offer a significant increase in robustness. In principle, if such vision-based on-the-fly landing site detection solutions are sophisticated enough, they may entirely ignore a-priori terrain information, thus making the use of a map redundant. However, if safety is the highest priority, this is not yet a reliable approach at the current level of technology.

3.3.1 Maps for UAVs. Maps may include 3D terrain information that can be exploited to better navigate and control the UAV both in normal and emergency situations. Typically, they are represented as 3D occupancy grids. Two different 3D mapping solutions are worth mentioning: a) Octomap [57], and b) Voxblox [99]. The first one provides a probabilistic 3D occupancy map compressed in a hierarchical way for memory efficiency. It consists in a volumetric representation of occupied, free and unknown areas in space, based on octrees. These maps are usually generated by post-processing data from LiDAR (Light Detection And Ranging) surveying. Voxblox, on the other hand, provides a field distance map, i.e., for each voxel, the Euclidean distance to the nearest obstacle is provided.

An alternative source offering terrain information is Digital Elevation Models (DEM) [89], which come in two forms. The first is Digital Terrain Models (DTM) [92],[146], that include information regarding the height variations of an area's bare ground, without any man-made structures or vegetation. The second type is Digital Surface Models (DSM) [109],[159] providing a representation of the elevation values for exposed ground, road surfaces, tree crowns, vegetation and buildings. Thus, DSMs include information for both the ground and the man-made structures, or vegetation, that lie on it. They can be generated by data coming from various sources, such as LiDAR (Light Detection And Ranging) surveying. DTMs are usually derived by post-processing DSMs. DSMs and DTMs often come in raster format, i.e., as geo-referenced images where a pixel's value denotes elevation of the corresponding location.

UAVs should have the ability to discover landing sites by exploiting terrain information or information from visual or other types of UAVs sensors, so that they could land safely either in emergency cases (of engine malfunction or poor operation) or in normal conditions. Literature concerning landing site detection is discussed in the following paragraphs.

3.3.2 Landing Site Detection Without Topological Terrain Map Knowledge. In general, landing site detection methods that do not exploit topological terrain map knowledge can be clearly divided in two eras: a) approaches mainly relying on traditional image processing, computer vision and machine learning algorithms, and b) approaches that employ modern Deep Neural Networks (DNNs), typically under the assumption of their on-board execution (although this is not always the case). DNN-based methods typically perform better, providing more accurate results, at the expense of increased computational requirements.

In [50], a UAV landing site detection system using mid-level sub-image discriminative patches is described. Image patch detectors are trained and tuned with several images obtained from Google Maps, weakly labeled as "Very dangerous", "Not Recommended", or "Safe". This way, a heatmap of the examined video frame is created. The final landing sites are defined by thresholding the heatmap. Image patch detection training initially proceeds by partitioning a training image set in two subsets of positive and negative images, respectively, and extracting HOG features [24] from 8x8 cells in the patches. K-Means clustering of HOG features into several image clusters follows. The authors then iteratively train linear Support Vector Machines (SVMs) using the initial image clusters. Thus, the resulting SVMs constitute detectors for a particular class of the respective image patch label. A tuning stage follows where various combinations of the system parameters are applied and the system tries to detect patches in a tuning image set, comparing the results with the actual labels. Final detection of the landing areas in input video frames is conducted on the output heatmap, describing the level of landing risk in the examined image regions.

In [27], a vision-based rooftop landing site method includes two methodological stages. The first one is the 3D reconstruction of the flight environment below the UAV, using dense Structure-from-Motion (SfM). The authors use a calibrated moving camera with known intrinsic and extrinsic camera parameters, generating the two views of a scene from different viewpoints instead of using a conventional stereo rig. It achieves scaling from metric coordinates to world coordinates via additional information, such as pose, and accurate calculation of translation and rotation in the several captured images, as well as depthmap estimation using temporaly distant viewpoints. The quality of motion estimation depends on feature extraction from the images (e.g., STAR [8] features and MSURF [118] descriptor). Subsequently, a SAD5 stereo correspondence algorithm is employed for calculating disparity maps [56] which are used for deriving a 3D point cloud modeling the environment below the flying vehicle. The second stage is the analysis of the scene in order to evaluate landing sites. Areas which are planar, flat and sufficiently large to allow unobstructed UAV landing/take-off are then identified using the disparity map variance, evaluated by utilizing a kernel which depends on the disparity value and the altitude of the vehicle. Finally, the landing confidence of the candidate points is modeled by a Gaussian process whose thresholding indicates the appropriate landing sites. The system was implemented on a ODROID-U2 processing board and the landing site detection part was executed at a frequency of 1Hz. The authors conducted two overflights over a one story building and could successfully identify a valid landing target in over 90 % of the video frames, where at least a part of the safe landing zone on top of the building was visible in the disparity images. Performance-wise, the system can avoid 99.7% of dangerous areas (correctness), whilst still detecting 9.3% of the safe landing sites (completeness).

An aerial image classification system for UAV forced landing site detection task is proposed in [49]. Several sampled segments of images captured by a UAV camera are utilized in order to extract color and texture feature vectors which are then classified via a Gaussian Mixture Model (GMM) or an SVM into two categories, "safe" or "unsafe". The procedure leads to a binary image which indicates the landing areas. Feature extraction involves color features (HSV) [78] and texture features (HOG and LBP [120]) in specific regions of the images prescribed by a spoke wheel operator [60], [48].

The atmospheric and functional state, including fixed wing UAV-body poses, velocities, IMU biases, and the wind field as measured by sensors like GNSS, IMU, magnetometer, and pressure measurement sensor, are employed for landing site detection (LSD) in [55]. The specific landing site spot is selected by considering possible risks (terrain roughness and slope, surrounding obstacles, local wind field) and is based on its texture and geometric shape, without utilizing any pre-existing knowledge about the local environment. The system consists of a front-end that segments images and classifies the segments to grass and not-grass classes, using a random forest method. More promising candidates are then fed into the back-end part that performs 3D reconstruction of the area and uses terrain slope and roughness along with the classification results to reach the final decision and, finally, evaluate an approach path.

In [91], a traditional image processing-based segmentation algorithm was proposed for dynamic vision-based landing site detection in UAV emergency scenarios. Segmentation relied on an edge detector and employed spatial distribution of the edge pixels in a geometry test that attempted to discover visible areas that are large and clear enough of obstacles. The algorithm did not discriminate between different classes of ground texture, such as grass or water, since it prioritized discovery of areas without obstacles, using image gradient maps. The underlying intuition was that the unobstructed areas would present less edge density and, thus, would be suitable for landing.

In [86], SafeUAV-Net is proposed as an embedded deep CNN-based system for dynamic UAV landing site detection, relying on computer vision. The neural model is trained on a synthetic aerial 3D dataset obtained from 3D scene reconstruction and designed for depth and safe landing area estimation, using only the RGB video input. This task is strongly related to semantic image segmentation as the authors predict a categorical class value for each pixel of the input video frames, which are thus segmented into "safe-landing" and "obstacle" regions by predicting depth and classifying their depicted plane orientation into three classes: "horizontal", "vertical" and "other". The goal is to enhance flight safety in urban or suburban areas at high speeds, while relying only on a limited range of on-board sensors. Evidently, scene semantics are ignored and only ground morphology is considered. Besides the complete SafeUAV-Net neural architecture (called "SafeUAV-Net-Large"), a faster, more lightweight variant is also studied (the so-called "SafeUAV-Net-Small"). The former/latter one achieves a runtime performance, i.e., processing speed, of 35/130 video frames per second (FPS), respectively, when being executed on the embedded AI compute board NVIDIA Jetson TX2, thus meeting real-time requirements on actual UAV hardware. A relevant synthetic aerial image dataset is also presented in [86], containing ground-truth per-pixel annotations for the "horizontal", "vertical" and "other" classes. On this dataset, SafeUAV-Net-Large, SafeUAV-Net-Small and DeepLabV3+ achieve a mIoU of 60.7%, 55.1% and 59.7%, respectively.

3.3.3 Landing Site Detection Employing Topological Terrain Map Knowledge. Landing site detection methods exploiting a-priori terrain information are actually the most prominent. The authors in [42] detect landing sites for fixed-wing UAVs in emergency situations by using the average height and height variance inside quadtree-based DEM partitions. Partitions whose height variance is below a limit are selected as landing sites and merged with neighboring ones, if they have similar average heights. Moreover, in [14], the authors determine suitable landing areas on topographical maps for emergency landing of UAVs during flight utilizing surface fitting on coarse elevation models by using Least Squares Error and slope calculation. In addition, in [61] the authors propose a system for landing zone selection based on a relatively simple geometric analysis of terrain roughness and slope. [121] proposes a scheme for selection or validation of landing zones for unmanned helicopters with terrain assessment, incorporating factors such as terrain/vehicle interaction, wind direction and mission constraints.

In [88], the authors create a system for efficiently and reliably assessing the safety of landing zones covered by low vegetation, via combining a volumetric occupancy map with a 3D Convolutional Neural Network (CNN). The CNN architecture is trained from raw occupancy data derived from LiDAR point clouds. Synthetic and semi-synthetic data (real data where synthetic obstacles have been inserted) were used during the training. The

Table 1. A comparison of landing site detection algorithms.

	Pros	Cons
[50]	three-category safety labels, relies only on RGB camera input	validated on a custom dataset instead of a well-known public one, relies only on obsolete algorithms/methods
[27]	3D reconstruction of the flight environment, relies only on a moving RGB monocular camera instead of requiring a 3D rig, 3D point cloud modeling	Structure-from-Motion(SfM) and disparity map creation prone to approximation environment modeling errors
[49]	color and texture features are taken into account for determining landing sites, relies only on RGB camera input	only two-category safety labels, relies only on obsolete algorithms/methods
[55]	multimodal functional state checking from a variety of sensors, 3D reconstruction of the flight environment	3D reconstruction prone to approximation environment modeling errors, relies only on obsolete algorithms/methods
[91]	relies only on RGB camera input	only detects unobstructed areas (one-category safety label), relies only on obsolete algorithms/methods
[86]	lightweight CNN-based, 3D reconstruction of flight environment, relies only on RGB camera input, three-category plane orientation/geometric labels	does not extract scene semantics, two-category final safety labels,
[42],[14],[61], [121]	exploits a-priory terrain information in the form of Digital Elevation Models (DEMs), simple geometric analysis of terrain roughness and slope	relies only on obsolete algorithms/methods
[88]	3D CNN-based, takes into account multimodal properties (constraints and atmospheric conditions)	increased computational and system complexity, requires LiDAR sensors

input of the trained system is a stream of globally registered point clouds in conjunction with a predefined candidate region of interest. The output is a probabilistic safety prediction indicating the safe landing zones. The output prediction takes into account mission constraints and objectives, atmospheric conditions and other factors. When tested on a semi-synthetic dataset, the proposed system managed to achieve an area under curve (AUC) of 0.95.

A summary of the comparison of landing site detection algorithms is illustrated in Table 1.

3.4 Semantic Mapping

Semantic mapping is an essential component of safe UAV deployment. Semantic information, in the form of high-level attributes and characteristics, is typically extracted from visual input in the form of class labels having natural meaning to human (e.g., describing object type, place, size, etc.). These labels are assigned specific spatial coordinates in a consistent manner and, optionally, semantically annotate a geometric 3D map of the surrounding area. In certain cases, the process of geometric mapping and semantics extraction are combined into a unified algorithm, while in other cases the geometric 3D map is pre-existing. In general, the goal is cognitive comprehension of the outdoors environment where the UAV, moves/flies and operates, in order to understand the scene below and facilitate better path planning.

The Keyhole Markup Language (KML) is a frequently used tool of semantic annotation. It is a file format used to display geographic data and to overlay annotations on a map such as Google Earth. KML uses a tag-based structure with nested elements and attributes and is based on the XML standard.

Map representation (graph-based, 2D/3D, indoors/outdoors), is an important choice in modern semantic maps. Due to the nature of UAV flight, this overview focuses on outdoor environments.

3.4.1 Outdoors Semantic Mapping for UAVs. To begin with, in [87], a semantic mapping system for scouting with UAVs is described. The proposed system aims at quickly assessing a large space for significant points of interest such as survivors in a disaster area, by semantically annotating a 2.5D navigation map with semantic classes of interest using forward-looking visual cameras and state calculation. This is accomplished by a deep learning 2D segmentation algorithm combined with a novel mapping procedure for projecting the 2D semantic class labels onto the 2.5D map grid as 2D annotations. The semantic segmentation part was evaluated on the MAVCAR dataset (created by the authors using video frames from aerial Youtube sequences) on the task of identifying / segmenting cars. The method achieved an IoU of 44.9%, 70.1% Precision and 55.5% Recall. Good field results were also obtained, concerning scouting for cars and collecting high-resolution data if a car was found, with the proposed semantic mapping pipeline running on-board a custom aerial platform equipped with an NVIDIA Jetson TX2 embedded compute board. Moreover, in [19], the authors semantically enhance UAV scene understanding and awareness by target tracking methods with ontological contextual information, in the form of a semantic map. This way, object tracking, identification and labeling are sequentially employed for inferring semantic content. In a similar manner, in [71] semantic maps utilizing UAVs are built, through an online gradient boost classifier. More specifically, a semantic map of the environment below the flying UAV, is superimposed upon the navigational maps and combined with specific targets detectors, such as a tree detector, for landing obstacle avoidance, or a car detector for target localization.

3.4.2 Outdoors Semantic Mapping for mobile robots. Multiple approaches exist for general mobile robot semantic mapping in outdoor scenarios, which can easily be applied in UAV applications. In [149], the authors introduce an information fusion scheme for scene understanding, considering image understanding as a two level problem which consists of object detection and semantic segmentation, specialized in the context of advanced driver assistance systems for intelligent vehicles. Their solution distinguishes the main components of the image, such as ground, sky, pedestrians, vegetation, etc. Similarly, [123] presents a method for creating a semantic map from multi-view street-level imagery, including semantic labels such as car, road and pavement. The method back-projects onto a ground plane map the street image views aggregated from multiple image samples, relying solely on a segmentation algorithm for semantic label extraction from the images. A similar segmentation-based semantic mapping is introduced in [122] assigning a label (e.g., car, pavement, road, etc.) to every voxel of 3D reconstructed map. This is accomplished by utilizing conditional random fields (CRF) and images from stereo cameras mounted on a vehicle to construct a 3D semantic model of large-scale urban environments. In [107], the segmented meshes resulting from the respective point clouds derived by a rotating laser scanner device, are being classified using a multi-class Gaussian Process (GP). The classifier processes feature vectors extracted from all mesh segments, while the possibility of detecting new classes during scanning is taken into account resulting to an active and life-long learning system. In addition, in [131], besides semantic model extraction using an omnidirectional camera, the authors also extract the spatial structure of street scenes and classify them as street intersection or non-intersection.

Typically, a single cue (e.g., place or object) is utilized for semantically labeling each map region. For instance, in [147] semantic mapping is performed by simultaneously solving two problems, i.e., semantic terrain mapping and semantic activity mapping, with the goal to analyze and semantically label each area with regard to its navigability. The analysis is approached as a classification problem, using Support Vector Machines (SVMs) or Hidden Markov Models (HMMs). In [51], a method for 3D outdoors street environment understanding is presented that unifies and tries to solve simultaneously non-parametric semantic image segmentation and 3D scene reconstruction from stereoscopic input. The algorithm produces a dense, semantically annotated and geometrically consistent point cloud, which is then converted into a semantic 3D occupancy grid containing moving scene objects.

In [36], a multimodal approach is presented that combines a laser sensor with visual camera input. A CRF model is utilized incorporating visual and laser features in order to classify the laser output into several class categories (car, wall, tree trunk, foliage, person, grass, and other).

In [16] a semantically annotated topological (instead of geometric) map is constructed on-line, using input from a camera mounted to a ground vehicle. The semantic image segmentation method in [128] is employed for extracting semantic labels, while a graph-based approach is proposed for building the topological map. Thus, each semantic segment/object (e.g., dirt, grass, road) is represented by a node in the topological graph, whose edges represent the neighborhood of the segments.

A summary of the semantic mapping literature is illustrated in Table 2.

Table 2. A comparison of semantic mapping algorithms.

	Pros	Cons
[87]	relies only on RGB camera input, DNN-based, multiple semantic class labels	uses a 2.5D navigation map instead of a fully 3D occupancy map,
[19],[71]	can be combined with specific target detectors and trackers (object tracking, tree-car detector), semantic map is superimposed on navigational maps	relies only on obsolete methods/algorithms
[149], [123], [122], [131]	utilizes visual odometry, multiple semantic class labels, relies only on RGB camera input	prone to projection errors onto the aggregated image plane views or 3D reconstructed maps, has increased computational complexity/requirements
[107]	multiple semantic class labels	relies on obsolete methods/algorithms for semantics extraction, requires laser sensor instead of an RGB monocular camera
[36]	multiple semantic class labels	relies on obsolete methods/algorithms for semantics extraction, requires laser sensor along with an RGB monocular camera
[147]	enhanced role of semantic mapping (terrain mapping, activity mapping)	relies on obsolete methods/algorithms for semantics extraction, requires laser sensor along with an RGB monocular camera
[51]	Exploits 3D occupancy maps	prone to 3D reconstruction errors, relies on obsolete methods/algorithms for semantics extraction, requires stereoscopic 3D camera
[16],[128]	Relies only on RGB camera input, on-line constructed topological map, multiple semantic class labels	only topological map (does not contain ground morphology information)

3.5 Person Detection

Another aspect of drone flight safety is person detection. Indeed, safe UAV landing requires that the drone detects any individuals present around the landing area, so as to avoid harming them in case of a malfunction; airspace above humans should constitute no-fly zone. Thus, the challenging problem of aerial person detection comes into the forefront. It is highly similar to the more traditional task of pedestrian detection from ground-level images, but the aerial perspective may induce additional difficulties. The output of a person detection system, as in the more general object detection case, is a rectangular Region-of-Interest (ROI) tightly enclosing the detected object image in pixel coordinates.

Early detectors relied on hand-made feature descriptors based on wavelets [103], edges [12], Gabor filter responses [127] etc. Histogram based-features, such as SIFT and HOG, subsequently became very common due to their efficiency and performance. In [24], the combination of HOG features and linear SVMs is proposed. In [140], a selective search approach is presented that relies on spatial pyramid features [70], on dense SIFT vectors [76] and an additive kernel SVM. Another group of detectors employs local patterns of qualitative gray-level differences, including Local Binary Patterns (LBP) [10] [142], and Local Ternary Patterns (LTP) [136].

More involved methods proposed for person/pedestrian detection include the Deformable Part Model (DPM) [39], which handles large pose variations. DPM learned a mixture of local templates for each body part and attained state-of-the-art results at the time of its introduction. Another effective detector was the Integral Channel Feature detector (ICF) [29] [32] which relied on oriented gradient and color features (HOG+LUV), selected by boosted decision forests, and surpassed previous detectors by a large margin. Later improvements on ICF managed to reduce its computational cost by approximating features across different scales [30] and neighboring windows [31].

More recently, CNNs have achieved great success in classification and object detection on large-scale datasets, such as ImageNet [68] and MS-COCO [52]. Early works [58] [161] that used CNNs for person/pedestrian detection are based on the R-CNN[44] structure, which relies on high-quality externally provided proposals to achieve good performance. Faster R-CNN [115] also constitutes a similar method, but its object proposals are built using a deep neural structure, while Fast R-CNN [43] is then employed in order to classify them. In [160], the problems Faster R-CNN faces for person detection were studied and a novel detector is presented, using a Region Proposal Neural Network followed by boosted forests on shared, high-resolution convolutional feature maps.

More recent single-stage detectors use an end-to-end system, which means that raw images (without object proposals) are given to a network, which then both localizes objects and classifies them. YOLO [113] and SSD [74] are the most promising examples. YOLO comes with its own custom backbone CNN for feature extraction, the

Table 3. A comparison of person detection algorithms.

	Pros	Cons
[103],[12], [127],[24], [140],[70],[76],[10] [142],[136]	simple computer vision methods	relies on obsolete methods/algorithms
[39],[29], [32],[30],[31]	robust and efficient	relies on obsolete methods/algorithms
[68],[52], [58], [161], [44],[115], [43],[160]	good performance, DNN-based	large-scale dataset in training,
[113],[74], [69], [15], [20]	good performance, DNN-based, end-to-end systems, increased speed	accuracy suffering in comparison to Faster R-CNN
[105],[98], [73]	DNN-based semantics extraction, multiple semantic class labels, concurrently performs person detection/pose estimation/instance segmentation with increased accuracy	requires large-scale dataset in training, increased computational complexity

so-called DarkNet, while SSD can be plugged on top of conventional CNNs, such as ResNet, Xception, etc. The end-to-end trainable nature of YOLO and SSD results in significantly increased speed, although accuracy slightly suffers in comparison to Faster R-CNN. Recent single-stage person detectors are presented in [69], [15] and [20], with the latter two handling aerial images. [20] studies the impact of the small size of persons and the deformation of camera's perspective projection in the overall accuracy of YOLOv2, trying to overcome these problems by using high-resolution inputs and a novel image preprocessing pipeline, composed of cropping-resizing and projective geometry-based vanishing point transformations. Evaluation on images with a very high input resolution of 3840×2160 pixels, captured using a DJI Phantom 4 UAV, shows that the proposed approach achieves a detection rate of 49.13% at a very low processing rate of 0.66 FPS (on a desktop NVIDIA Geforce GTX Titan X GP-GPU). When retaining the same input resolution while simultaneously removing the proposed image preprocessing stage, vanilla YOLOv2 achieves a corresponding detection rate of 17.33% at 4.34 FPS. In [15], the Okutama-Action aerial video dataset for human action detection is presented. For evaluating it on person detection tasks, an SSD detector on top of a VGGNet CNN backbone is implemented, resulting in a test set accuracy of 72.3%, using mean Average Precision at 0.5 IoU as the evaluation metric.

In general, however, person detection methods for ground vehicles (i.e., pedestrian detectors) are much more prominent in the literature than algorithms for aerial ones. Semantic image segmentation methods have also been applied to this end, instead of visual object detectors. For instance, in [105], a bottom-up CNN model is presented which jointly addresses the problems of person detection, pose estimation, and instance segmentation, using a unified part-based modeling approach. The proposed *PersonLab* CNN is trained to detect individual keypoints and predict their relative displacements, allowing a grouping of keypoints into person pose instances. Once the keypoints have been localized, a greedy decoding process is used to aggregate them into person instances. Moreover, the model predicts dense instance segmentation masks for each person associating semantic person pixels with their corresponding person instance, delivering instance-level person segmentations. In [98], along with real-time pedestrian detection based on shape and size features of the candidate region, semantic image segmentation through a CNN with simpler architecture can provide an accurate solution to the pedestrian detection for an Advanced Driving Assistance Systems (ADAS). Finally, in [73], a pedestrian detector utilizes semantic image segmentation through a deep learning method to pixel-wise label images into 11 semantic classes with the HOG+LUV features in order to improve the filtered channel feature based detector.

A summary of person detection algorithms with application on UAV operation along with the advantages and disadvantages is contained in Table 3.

3.6 Vision-based Obstacle Avoidance for Safe UAV Navigation

Another crucial component of autonomous UAV flight safety is obviously the avoidance of obstacles, such as rigid objects, persons and crowd gathering locations, or no-fly zones depicted in the 3D navigation map. This is a reactive process that adjusts the UAV flight trajectory to render it unobstructed by obstacles or individuals, by exploiting well-known computer vision approaches for analyzing video input. Such methods, relying on input data from cameras (or other sensors, such as LiDAR, radar, ultrasonic, IR, etc.), are utilized commonly in GPS-denied environments, in order to offer efficient UAV navigation by exploiting the video (or other sensors) analysis results. Since the early 2000s, a number of relevant methods started appearing: Bayesian filter-based Simultaneous Localization and Mapping (SLAM) approaches [144] [145], fast reactive obstacle avoidance heuristics [11], stereoscopic vision [18] [54], or optical flow estimation [133] [46] [156], [59].

Two vision-based method categories are of particular note: Visual Simultaneous Localization and Mapping (Visual SLAM) and Image-Based Visual Servoing (IBVS). Visual SLAM [95] can be used to detect and avoid obstacles during flight time, by mapping the immediate environment and localizing the drone with respect to that 3D map. Localization includes an estimation for both the position and the orientation of the UAV-mounted camera at each time instance. Visual SLAM performs an incremental 3D scene reconstruction based on the camera feed, using a real-time, on-line variant of Structure-from-Motion algorithms, augmented by visual place recognition, graph-based map modelling and loop closure modules. The computed map is typically a 3D point cloud, either sparse, semi-dense or dense, with the first estimated location of the UAV employed as the arbitrary origin of the map coordinate system. However, since a point cloud cannot distinguish between unobserved and observed-to-be-empty space, different approaches are typically employed for safe map representation in autonomous vehicles. The Octomap, previously described in Section 3.3, is a popular choice.

IBVS can be used for sending suitable motion commands to the UAV motors, so as to achieve a specific control purpose (e.g., avoid an obstacle, land, etc.) in an autonomous manner. In essence, it is a visual feedback control loop where control laws are formulated directly on 2D image space, without attempting to estimate 3D state. The goal is to control 3D camera velocity based on the current deviation of the positions (in 2D pixel coordinates) of on-target landmark point images from pre-specified desired ones. Typically, IBVS requires knowledge of camera intrinsic parameters and a depth map per video frame, at least in principle. In scenarios such as UAV obstacle avoidance or landing control, the camera orientation/position on the vehicle may be considered fixed, so that 3D UAV velocity is the quantity in need of control [124].

Since most of the methods that were presented before 2010 are now obsolete, only more recent approaches are presented below.

An obstacle avoidance method for low-cost UAVs is proposed in [9], where SURF features are extracted from the image and utilized for fast processing and detection of obstacles. They are matched against images stored in a database and, when obstacles are recognised, the avoidance mechanism is triggered with a specific control law. Another feature-based obstacle avoidance method for small UAVs using monocular vision is [72]. More specifically, multi-scale-oriented patches (MOPS) are utilized in order to detect the outlines of observable obstacles via a combination of the spatial coordinates of SIFT interest points, thus perceiving the three-dimensional shape of the obstacles. This method is most useful in GPS-denied environments where the visual sensor/monocular camera is the only source of information.

In [85], the authors present a fast, Bayesian filter-based terrain mapping method using a sequential Extended Kalman Filter (EKF) and visual features extracted from monocular camera images. The feature measurements are used in the inverse depth parameterization in order to enable fast depth convergence. The converged features that are stored as a six-element vector consisting of an anchor point, azimuth, elevation and inverse distance to the feature, are added to the terrain map for the obstacle observation and navigation.

Table 4. A comparison of vision-based obstacle avoidance methods.

	Pros	Cons
[9], [72], [85],[156], [46],[133], [100], [11], [90]	relies only on RGB camera input	relies on obsolete methods/algorithms
[54], [18]	relies only on RGB camera input	requires stereoscopic 3D camera
[26], [144], [145]	applicable in GPS-denied environments	requires laser sensor along with an RGB monocular camera
[144], [145], [95]	relies only on RGB camera input	prone to approximation reconstruction errors
[124], [100], [90]	relies only on RGB camera input and known camera intrinsics,	increased complexity of control laws
[133], [156], [59]	relies only on RGB camera input, 3D map of the flight environment	prone to errors/not robust
[46]	3D map of the flight environment	prone to errors/not robust, requires stereoscopic 3D camera

In [46] an optical flow-based obstacle avoidance method is proposed, exploiting the differential algorithm of Lukas-Kanade [77]. Thus, the perceived 3D structure of the 3D flight environment is estimated jointly with the video camera image's depth map. In addition to localizing obstacles, their shape is also determined by taking into account the direction of flight or maneuvers performed by UAVs. [100] presents a micro-UAV visual servoing approach for obstacle detection and avoidance through a fuzzy logic controller for automating the collision avoidance. It is capable of adjusting the UAV heading, keeping the flight trajectory such that obstacles remain to the side of the camera until they disappear from the field-of-view.

In [90], a novel vision-based collision avoidance controller is proposed that uses a combination of spherical imaging, properties of conical spirals and IBVS with predictive control. The method exploits the prediction and a practical implementation of a vision-based control law which is extended to spherical imaging single feature points, using specifications of conical spiral motion and ensuring collision avoidance. [26] proposes a monocular vision odometry system based on laser sensor assistance, through estimation of the objects distance in order to avoid them in indoor environments. This is performed by a combination of template matching for determining the laser's point of contact with the observed obstacles and the gray centroid method [148] to obtain the coordinates of said point in the 2D image.

A summary of vision-based obstacle avoidance methods for safe navigation is contained in Table 4.

3.7 Path Planners for Safe UAV Landing

Path planning and path following, i.e., trajectory generation and tracking in 3D air-space, are essential operations for UAVs in general. When it comes to safety, besides geo-fencing that considers people and human crowds, the most important relevant issue is the trajectory followed by the UAV while it attempts to land. Protection of human lives in case of possible drone malfunctions, or more generally in any scenario where emergency landing must be executed, requires a systematic and intelligent approach for guiding the UAV towards the landing site. This can be done either by exploiting the 3D map and UAV/camera 3D pose, or in a reactive, vision-based manner for GPS-denied environments. In the first case, the 3D coordinates of both the UAV and the desired landing site need to be known at each time instance, while the actual vehicle control for following the planned path in 3D space is delegated to a trajectory tracking controller, operating in 3D space and relying on GPS/IMU fusion. In the second case, the landing site and/or UAV 3D locations may not be known accurately, or at all, while reactively planning and actually tracking a trajectory are interleaved operations. Relevant methods from both categories, specially designed for UAV landing, will be presented below.

3.7.1 Path Planners Exploiting the 3D Map and the UAV/Camera 3D Pose. Firstly, [42] uses two path planners (Rapidly Exploring Random Tree, or RRT [33], and Particle Swarm Optimization, or PSO [116]) for path planning in limited time from the current UAV position to the closest detected landing site. RRT is a method that systematically connects a goal point with a starting reference point while avoiding rigid obstacles, by expanding the nodes of a tree trajectory generating random points which are uniformly distributed all over the free space. The new generated points, lying close to the previous nearest points, are added to the tree and create an unobstructed, collision-free trajectory. Sampling-based path planning variants of this algorithm are Waypoint RRT (WRRT) and Dynamic RRT (DDRRT) [23], as well as a multiple-robot variant called Rapidly-exploring Random Graphs (RRG) [64]. Finally, multi-RRT [137] exploits two expanding random trees, one from the starting point and another from the goal point (possible landing site). In environments with dynamic conditions and multiple moving obstacles, multi-RRT is more efficient than RRT. When the two expanding trees connect to a random node, a feasible path is created between the two points.

PSO-based path planning relies on a metaheuristic algorithm, inspired from the behaviour of flocks of birds [67]. It exploits and evaluates particles which represent possible solution paths. Assuming that the swarm μ has n candidate particles, of known velocities and positions, these attributes are updated according to an iterative procedure ensuring that each particle connects the start and the end position and is close to the optimal solution of a cost function minimization problem. Eventually, the PSO finds the best path solution, according to a condition verification which checks if the candidate particle lies in the terrain or not and computes its respective height.

Additionally, in [27] a trajectory generation method towards a rooftop landing site is proposed. An optimal polynomial trajectory is computed from the current position to the selected landing site minimizing the angular velocity and the jerk, estimating concurrently the coefficients of the trajectory B-splines. A second objective function is calculated in order to maximize the gained information for continued observation of the landing site point so as to reduce the confidence uncertainty. Finally, an alternative to this information objective is the minimization of the distance to the landing site, or generally to an interest point/surveillance target. The trajectory is recomputed online as the perception system provides additional observations and the landing site confidence evolves with an optimization formula for determining the B-splines coefficients combining the minimum jerk and maximum information objectives.

3.7.2 Path Planning in a Reactive, Vision-based Manner or in GPS-denied Environments. Reactive vision-based methods for UAV path planning/tracking have also been proposed, mainly aiming to achieve safe landing. [7] presents an approach for vision-based UAV landing on a moving platform in GPS-denied environments using motion prediction. A vision-based control system is proposed, which enables a UAV to track an Unmanned Ground Vehicle (UGV) and land on it using an integrated vision sensor without external localization systems, such as GPS sensors. Tracking of the UGV relies on the latter's presence within the camera Field-of-View (FOV) and is performed utilizing a 6-DOF controller in cascade form, when the vehicle lies outside the FOV, or a normal IBVS tracking, when the candidate UGV is back within the visual camera's FOV.

A similar method for quadrotor landing on a moving target using dynamic IBVS control is described in [124]. A control law is implemented for the quadrotor landing in a flat and textured target plane which is identified by visual features and may be moving with bounded linear acceleration in any direction. The proposed controller ensures the convergence to the moving landing point with dense conditions such as bounded, possibly time-varying and wind-induced force disturbances or measurement errors from the motion of the target plane. The position and the velocity of the moving platform is given by depicting the centroid of features points (landmarks) lying on the target plane and the translational optical flow, respectively.

[34] presents an integrated approach to autonomous artificial landpad detection, approach and landing, for a low-cost quadrotor equipped with a monocular camera. The method exploits a landpad marker's geometrical characteristics for detecting it on video. Notably, the landpad need not be located directly underneath the drone.

Table 5. A comparison of path planners for safe UAV landing.

	Pros	Cons
[33],[23], [137]	high accuracy	requires external self-localization system (e.g., GPS+IMU) and known landing spot coordinates, dynamic memory allocation may lead to memory overflows
[64]	supports multiple-robot fleets, high accuracy	requires external self-localization system (e.g., GPS+IMU) and known landing spot coordinates, dynamic memory allocation may easily lead to memory overflows, supports only 2D map
[116],[67], [42]	high accuracy	requires external localization system (e.g., GPS+IMU) and known landing spot coordinates, stochasticity in the obtained solution
[27]	exploits multiple information sources (distance, angular velocity, jerk) for increased accuracy, 3D world coordinates of the landing spot do not need to be known, relies only on RGB camera input to visually detect landing spot, does not depend on external self-localization systems (suitable for GPS-denied environments)	increased complexity due to minimization of multiple cost functions, prone to errors
[7]	does not depend on external self-localization systems (suitable for GPS-denied environments) relies only on RGB camera input to visually detect landing spot, supports moving landing spots (UGV-mounted)	increased complexity of control laws, prone to errors
[124], [34]	relies only on RGB camera input to visually detect landing spot, supports moving landing spots (UGV-mounted)	requires external self-localization system (e.g., GPS+IMU), increased complexity of control laws

A SLAM algorithm is utilized in order to on-the-fly map the area near the landpad, so that a proper trajectory for safe landing can be designed and executed.

Recent literature concerning path planning for safe UAV landing is summarized in Table 5.

3.8 Transferability and Adaptability Issues

Evidently, most advanced computer vision-based methods rely on Deep Neural Networks (DNNs), thus giving rise to issues of transferability and adaptability of the pretrained DNN models to real-world scenarios. This issue is especially pronounced when training occurs using simulators (e.g., training data obtained through photorealistic 3D graphics engines) and, subsequently, the model is tested in a real-world environment; the reduced test-stage accuracy resulting from the differences between the simulated and the real-world data is called the *reality gap*. However, this is in fact simply a facet of a larger problem, endemic in all machine learning approaches: model performance suffers at the inference stage if there is a shift in data distribution between the training and the test stage. For example, the well-known Cityscape dataset for semantic segmentation may not accurately reflect the distribution and appearance of semantic classes in other cities, leading to reduced inference accuracy in case similar distribution shifts take place. Transferability/adaptability issues can be aggravated if a DNN model is compressed after training, so as to be deployed on embedded computing devices (e.g., on UAV hardware).

The active research field of *domain adaptation* is an effort to tackle the above-detailed issues. In this paradigm, a machine learning model is trained in an initial source domain and tested on a different target domain; the two data domains contain identical semantic information, but the data are distributed differently within them. A good example would be training a semantic segmentation DNN model in a photorealistic 3D graphics-based simulator for a specific set of semantic scene/object classes, and then deploying the trained model on a real-world environment containing the exact same scene/object classes. Domain adaptation methods intervene at the inference/test stage and actively try to decrease the impact of the cross-domain distribution shift in model accuracy. Algorithmic strategies such as sample re-weighting [141], or intermediate subspace transformation categories [22, 162] do not rely themselves on domain adaptation DNNs, but most advanced approaches are in fact DNN-based, such as discrepancy-based methods [165], generative Adversarial models [65], discriminative adversarial models [152] and reconstruction-based methods [151]. For instance, in the semantic image segmentation case, [143] proposes an unsupervised domain adaptation method relying on Generative Adversarial Networks (GANs), where the goal is to automatically align convolutional (CNN-derived) image features from the source domain with respective ones from the target domain. The underlying assumption is the availability of a labeled/unlabeled training set for the source/target domain, respectively. The method attempts to avoid the over-alignment of features corresponding to background classes, since they typically vary little across domains, while for foreground/object features different representations are derived per individual visible object instance; the target-domain instance is forced to align at the feature level with the most similar one in the source domain. In [101], a two-step domain adaptation approach is proposed that minimizes the inter-domain and intra-domain gap together, by separating target domain into an easy and a hard split using an entropy-based ranking function and, generally, utilizing a self-supervised adaptation method. For the specific case of person/crowd/pedestrian detection tasks, [47] exploits multimodal information (e.g., RGB, thermal) and utilizes a domain adaptation approach to generate pseudo-annotations in the source domain. Subsequently, these are updated according to the multispectral detector's parameters, aligning visible and infrared image pairs of the target domain. More closely related to the reality gap issue, [129] and [17] explore pixel-level domain adaptation, while [41] focuses on feature-level domain adaptation. These advances required a rethinking of the approaches used to solve the simulation-to-reality domain shift problem for robotic manipulation as well.

4 COMPUTER VISION-BASED UNIFIED UAV SAFE LANDING PIPELINE

In this Section, synergistic exploitation of the above technologies is presented through our example algorithmic pipeline for UAV safe landing. It consists of a number of modules, where each one implements a computer vision/machine learning algorithm. We assume that the vehicle is equipped with proper embedded AI compute boards (e.g., NVIDIA Jetson Xavier, Intel NUC, etc.) and GPS/IMU localization sensors.

Although it is not strictly necessary, easy inter-process communication and synchronization among all modules may be transparently assured with minimum fuss by relying on the wide-spread Robotic Operating System (ROS) middleware [111]. Additionally, a 3D flight area map is assumed to be available in Octomap format. Overall, this is a reasonable platform architecture with fairly common design choices [80] [81] [84] [83], easily assembled by using only commercial off-the-shelf or open-source HW/SW building blocks. Note that a ROS-based architecture inherently ensures modularity and software isolation between different modules, thus preventing the existence of a single point of failure.

The software modules of our example algorithmic pipeline are the following ones:

- Potential Landing Site Detection (PLSD)
- Visual Landing Site Detection Using Semantic Segmentation (VLSD)
- Crowd Detector (CD)
- Semantic Map Region Projector (SRP)
- Semantic Map Manager (SMM)
- Person Detector (PD)
- Simple Path Planner (SPP)

Each of these components is described below. Note that PLSD must run off-line, before UAV deployment and mission execution, while the other modules are executed on-line and repeatedly, as needed. Actual trajectory tracking, both for approaching a landing site and for landing itself, can be performed by any off-the-shelf controller operating in 3D space, since 3D coordinates of both the UAV and the landing sites are assumed known at all times.

4.1 Potential Landing Site Detection (PLSD)

A simple, but efficient algorithm for UAV potential landing site detection [62] has been adapted as a building block for our example pipeline. The main aim is to identify (sufficiently) flat and large areas in a topographical map for UAV safe landing in normal or emergency situations. The method is briefly described below.

DSM and DTM raster files are exploited in order to identify regions containing vegetation, buildings, and generally the objects upon the bare ground, by simply evaluating the height difference between the DTM and the DSM model. In addition, flat areas are discovered by evaluating the local terrain slope through the use of an image gradient operator on the DEM file and by thresholding the resulting image, in order to keep areas with small terrain slope. Subsequently, connected components analysis is used on the resulting binary image in order to find and retain regions whose area is above the minimum potential landing area size.

The final result is a list of sufficiently large map areas with no buildings/vegetation and small terrain slope, constituting areas which can be characterized as landing zones. Since the method is based on pure image processing and not machine learning, it foregoes the need for complicated training. The only prerequisite is the existence of the DSM and DTM height maps, which are available a-priori with sufficient resolution for large parts of the globe.

Map-based landing site detection can provide only information regarding potential landing sites, based on terrain geometry. Such landing sites may be precomputed and be available as annotations on the map. Whether these potential landing sites can actually be used for UAV landing at a certain time instance depends on whether the site is free from water, people/crowds, or cars, at the actual landing time. Such a check can be done either manually by the pilot through the drone video feed, or automatically by on-the-fly applying person/crowd/car detection and semantic segmentation algorithms on the UAV-captured video.

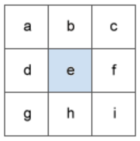


Fig. 1. 8-neighborhood of a DSM.

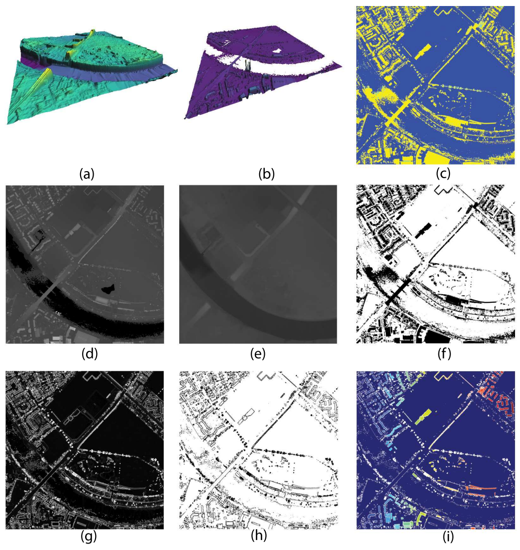
PLSD input consists of a DSM and a DTM model in raster format, i.e., as a regular grid of elevation values of a terrain. As discussed in Section 3.3, the DTM (Figure 2aa) depicts just the terrain and no man-made structures or vegetation, whereas the DSM (Figure 2ab) depicts the terrain along with buildings and vegetation. Due to sensor inefficiencies during DSM acquisition, relevant files often contain pixels with no value (no elevation information). As DTM is constructed by post-processing the DSM, these pixels are usually assigned values through some sort of interpolation. In the context of our example pipeline, DSM pixels with no values are assigned elevation values from the corresponding pixels of the DTM file. The algorithm consists of the following steps:

1) Detection of man-made structures and vegetation. By subtracting DTM from DSM and applying a threshold to the outcome, a binary image is derived (Figures 2af, 2be) which marks pixels depicting man-made structures and vegetation whose height is above a selected (small) threshold.

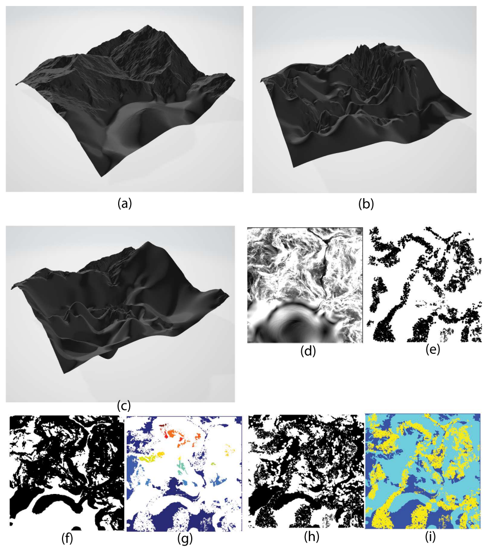
2) Terrain slope determination (Figures 2ag, 2bd). Subsequently, the local slope of the depicted areas in the DSM is calculated. According to GIS literature [3], slope is the maximum rate of change in value (elevation) from a pixel (cell) to its neighbors. The lower the slope value, the flatter the terrain. As far as the slope calculation is concerned, the rates of change of the surface elevation in the horizontal ($\frac{dz}{dx}$) and vertical ($\frac{dz}{dy}$) directions from the central cell determine the slope. Slope, in degrees, is calculated as [3]:

$$slope_{degrees} = \frac{180}{\pi} \arctan \sqrt{\left(\left[\frac{dz}{dx} \right]^2 + \left[\frac{dz}{dy} \right]^2 \right)} \tag{1}$$

The values of the center cell and its eight neighbors determine the horizontal and vertical rates of elevation change.



(a) Real-world dataset



(b) Synthetic dataset

Fig. 2. **Real-world dataset** (map1): (a) 3D view of DTM, (b) 3D view of DSM, (c) final map (meaning of colors is explained in the text) (d) DSM, (e) DTM, (f) binary image of buildings/vegetation, in black, (g) Terrain slope determination via Sobel operator, (h) binary image of low slope areas, in white, (i) connected components analysis result, **Synthetic dataset:** (a) 3D view of map4 (sub-images (d)-(i) refer to this map), (b) 3D view of map5, (c) 3D view of map6, (d) Terrain slope determination via Sobel operator, (e) binary image of vegetation, in black, (f) binary image of low slope areas, in white, (g) connected components analysis result, (h) ground truth information, potential landing sites in white (i) final map (meaning of colors is explained in the text).

For a neighbourhood such as the one depicted in Figure 1, the rates of change in the x and y direction for cell "e" can be calculated as:

$$\frac{dz}{dx} = \frac{(c + 2f + i) - (a + 2d + g)}{8 * x_{cellsize}} \tag{2}$$

$$\frac{dz}{dy} = \frac{(g + 2h + i) - (a + 2b + c)}{8 * y_{cellsize}} \tag{3}$$

Essentially, the rates of change in the x and y direction as used in the GIS literature are the horizontal and vertical derivative approximations generated by the well known Sobel operator [38], scaled by a factor of $-8 * cellsize$.

3) Sobel operator gradient image thresholding (Figure 2ah, 2bf): After extracting the elevation gradient magnitude image, it is thresholded in order to classify the DSM pixels in flat or non-flat areas, based on the local slope. Obviously, near-flat areas are retained as potential landing areas. The thresholding utilizes a predefined terrain slope threshold, based on slopes which are appropriate for UAV landing.

4) Binary image connected components evaluation (Figure 2ai 2bg): Connected components analysis is applied on the binary image resulting from the previous step. Connected components with sufficiently large number of pixels, i.e., of sufficient area, are retained.

5) Creation of the final map (Figures 2a-c, 2b-i): In order to create the final map, image regions that overlap with buildings and vegetation (found in Step 1) are removed from the large low slope areas found in Step 4. The final map consists of three categories of pixels:

- Blue pixels: landing zones, i.e., the regions in the DSM map which are characterized from small terrain slope and large enough area for UAV landing.

- Light blue pixels: no landing zones, i.e., the regions in the DSM map with large terrain slope or very few pixels (small area).
- Yellow pixels: no landing zones due to buildings and vegetation.

4.2 Visual Landing Site Detection Using Semantic Segmentation (VLSD)

Using semantic segmentation, 2D image regions corresponding to ground areas that are most likely suitable for landing (such as grass, or pavement) can be identified and discriminated from areas unsuitable for landing (such as trees, people, water, or buildings). Such a semantic segmentation CNN may be executed separately per video frame, with different accuracy/speed trade-offs. In our example UAV safe landing pipeline, PSPNet was adopted and properly adapted. Instead of complex encoding/base feature extracting networks (pretrained on the ImageNet dataset for whole-image classification) that are typically used, such as ResNet [53] or VGG [130], we coupled the PSPNet network layers with a MobileNetV2 base feature extractor [119], enhanced with dilated convolutions. This resulted in a relatively fast implementation at a low accuracy penalty. Additionally, the base feature extraction network was pruned by removing 5 intermediate convolutional layers, so as to speed-up execution during inference, resulting in a lightweight variant of the full MobileNetV2+PSPNet network.

The ADE20K semantic scene parsing dataset was selected for training [166]. It is a large dataset containing $K = 150$ distinct object classes from both indoors and outdoors environments, including all important classes relevant to the landing site detection task. Since the 150 discrete object classes of ADE20K are not necessary for potential landing site detection, they were manually clustered into 11 classes, relevant to outdoor UAV flight: Landing Area, Building, Background, Man-Made Structures, Vegetation, Sky, People/Animals, Vehicles, Light, Man-Made Objects, Water. Reducing the number of discrete classes simplifies the problem and allows the segmentor to generalize better. To achieve a good accuracy/speed trade-off during inference, each input image was resized so that the short edge length (in pixels) is 180px, while the aspect ratio was preserved under the constraint that the longer edge length cannot be greater than 1000 pixels. The network was trained for 220 epochs. The learning rate and the batch size were set to 0.01 and 8, respectively.

Despite the care taken for implementing a VLSD module that can be as fast as possible on embedded AI compute hardware, there is no actual need for it to run either continuously or with near-real-time performance. It may be used for verifying actual potential landing site eligibility from a single video frame, once the site becomes visible on the UAV-mounted camera.

4.3 Crowd Detector (CD)

A recent human crowd detection module from [138] [139], able to rapidly extract fairly accurate 2D crowd heatmaps from video frames captured by a UAV-mounted camera, has been adapted for use in our example pipeline. The underlying algorithm is a very lightweight Fully Convolutional Neural Network (FCNN), satisfying computational and memory limitations of on-board UAV execution, that can effectively distinguish between crowded and non-crowded image areas. The network is trained using a novel generic regularization approach for supervised learning, based on the Graph Embedding Framework, applicable to different deep neural architectures for classification tasks.

The module operates in the following manner. First the current video frame/image is fed to the CNN, which (thanks to its fully convolutional nature) essentially segments the input into overlapping, spatially arranged patches and assigns a probability of human crowd existence to each one. Thus, the image is "mapped" into a 2D semantic heatmap denoting the estimated probability of crowd depiction in each location within the input video frame. The 2D input is fully spatially aligned with the 2D output, since the convolutional neural layers preserve spatial information due to the arrangement of the activations, as opposed to fully connected layers that discard it.

The fully convolutional nature of the proposed model is essential both for handling input images of arbitrary dimension and for estimating the heatmap. Additionally, it keeps computational and memory requirements low, thus permitting execution on-board a UAV, given that a CNN without fully connected layers has a drastically decreased number of model parameters.

The proposed CNN model contains six convolutional layers. The network accepts RGB images of size 128×128×3. The output of the last convolutional layer is fed to a Softmax layer which produces a distribution over the 2 classes of *Crowd* and *Non-Crowd*. Each convolutional layer except for the last one is followed by a Parametric Rectified Linear Unit (PReLU) activation layer, which learns the parameters of the rectifiers, since it has been proven to enhance the classification performance, while max-pooling layers follow the first and the fifth convolutional layers.

Regularization is essential in deep learning for enhancing generalization ability, since deep neural networks are prone to overfitting due to their high capacity. Thus, in the employed algorithm, a multiple-loss training approach is used that combines the Graph Embedding (GE) framework [150] with CNNs, in the form of a regularizer term for classification purposes. This approach is described below.

Considering a deep neural architecture for classification purposes with a *softmax loss*, one or multiple additional loss layers are attached which impose certain constraints, motivated by the GE algorithms. The *Euclidean loss* (sum of squares) layer can be easily employed to implement them, although more sophisticated losses may also be defined.

Given a set of N input images $\mathcal{X} = \{\mathbf{X}_i, i = 1, \dots, N\}$, let $\boldsymbol{y}_i^L$ be the corresponding representations of the feature space generated by a specific deep neural layer, L, as the corresponding projection of training data in the GE concept. Then, the softmax loss is defined as:

$$L_s = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{K} l_{i,k} log(p_{i,k}), \tag{4}$$

where K is the number of classes, $l_{i,k} \in \{0, 1\}$ is a binary indicator with a value of 1 if the class label k is the correct classification for sample i, and $p_{i,k}$ is the predicted softmax probability that the sample i belongs to class k.

The Euclidean loss for a target representation $\boldsymbol{t}_i$ determined by a specific GE algorithm is defined as:

$$L_e = \frac{1}{2N}\sum_{i=1}^{N} \|\boldsymbol{y}_i^L - \boldsymbol{t}_i\|_2^2. \tag{5}$$

The Euclidean loss layer acts as a regularizer, transfusing the ideas of GE in deep learning. It can be attached to all layers of the deep neural architecture, to individual layers, as well as to combinations of layers. For a deep model of N_L layers, the total loss in the regularized training scheme is computed by summing the above losses0 :

$$L = L_s + \sum_{i=1}^{N_L} \eta_i L_e, \tag{6}$$

where parameter $\eta_i \in [0, 1]$ controls the relative importance of the Euclidean loss of each deep layer; $\eta_i = 0$ implies that no regularizer is attached to the i-th layer.

Different regularizers, motivated by different GE approaches, can be simultaneously applied to a deep neural layer. Discriminant Analysis (DA) and Minimum Enclosed Ball (MEB) regularizers are specific instances of this framework, introduced and described in detail in [139]. Training is performed in the typical manner for feed-forward neural networks, using error back-propagation and Stochastic Gradient Descent on the Crowd-Drone dataset [63]. This dataset includes Youtube videos depicting crowded events (e.g.marathon, festival, parade, political rally, protests, etc). and non-crowded generic drone videos. Non-crowd images (e.g. cars, buildings, bikes, etc.) from the senseFly-Example-drone and the UAV123 datasets are also included. Crowd regions are manually annotated in the extracted frames. The dataset includes a total of 5,920 crowded regions and an equal number of non-crowded images.

4.4 Person Detector (PD)

Person detection is a vital part of any autonomous system regarding human safety [135]. Although crowd detectors can achieve exceptional results in detecting dense crowds, they mostly fail in detecting sparse individuals; therefore, employing the CD module for this task is not advisable. Additionally, person detection should be accurate in its analysis, running constantly in the background and performing as close to real-time as possible. Thus, employing the VLSD for solving this task as well, via semantic segmentation, is highly suboptimal, since VLSD runs slow on embedded AI hardware and is only really necessary when a landing site is approached (i.e., very sporadically). As a result, an additional Person Detector module was added to our example UAV safe landing pipeline, exclusively for detecting individuals from aerially captured video frames/images.

This task poses challenges due to the small size of objects/persons (especially in high flight altitudes), as well as due to unforeseen and wide-ranging variations in illumination, camera orientation, etc. An off-the-shelf one-stage CNN-based object detector was selected for powering the relevant module of our example pipeline, since these detectors are faster and more computationally efficient compared to two-stage approaches [115][43].

YOLOv3[112] was selected and adapted for use in our example pipeline, since it has proven to be suitable for embedded AI applications [97]. The network was adapted to receive as input, images with a resolution of 608×608 pixels. The deployed model was pre-trained on the MS-COCO [52] dataset.

4.5 Semantic Map Region Projector (SRP)

The Semantic Map Region Projector (SRP) is a crucial module of our example pipeline. It is responsible for back-projecting the 2D crowd heatmaps onto the 3D volumetric map handled by the Octomap. Such a heatmap is produced every time the CD module analyzes a video frame captured by UAV's camera.

The process is composed of the following stages:

First, all the necessary information is gathered and synchronized with the currently received heatmap, based on accompanying timestamps. This information, including the UAV position, gimbal orientation and camera intrinsic parameters (e.g., focal length), must match the moment that the original video frame was captured.

By applying thresholding on the heatmap in order to retain only image locations with high probabilities of crowd existence, the heatmap is converted into a binary image where groups of adjacent pixels with a value of 1 (white) represent 2D regions occupied by crowd. Next, a contour following algorithm is applied in order to extract the contours of this image, resulting in a new binary image indicating the boundaries (white pixels) of the crowd regions as 2D polygons. If necessary, the resulting polylines are simplified while maintaining their shape, according to the Ramer-Douglas-Peucker algorithm [35]. The latter receives a curve composed of line segments and outputs a similar one with fewer points.

By traversing the points (pixels) of the regions' boundaries in a counter-clockwise manner, ray casting is performed [45], [117]. This contour image lies on the focal plane of the UAV camera which has the following known parameters (specific to the moment the corresponding video frame was captured):

- a) the location of the center of projection (COP) in the 3D world (derived from the UAV location).
- b) the camera orientation (derived from the gimbal state).
- c) the distance of the focal plane from the COP (the camera focal length).

Thus, one can cast a ray from each of the boundary contour points towards the voxels of the Octomap (as shown in Figure 3). This results in finding the occupied voxel hit by each ray, leading to the evaluation of the X, Y, Z terrain coordinates where each of the contours' points is projected, as the Octomap is coordinates-referenced.

By this procedure, and since the 2D boundary contour points are traversed sequentially, the points of the 3D boundary contour (polyline) are extracted.

4.6 Semantic Map Manager (SMM)

The Semantic Map Manager (SMM) is a module of our example pipeline with the following responsibilities. Firstly, it receives from the SRP the crowd gathering locations on the 3D map (see Section 4.5). As the drone moves, and its camera "sees" new terrain areas, the newly generated polygonal lines are merged with previous ones using the union operator.

Subsequently, the constantly updated geometric annotations are stored in an internal data layer (as ROS messages, in our implementation) that will be exported as KML files. These can be used for drone navigation and control purposes, as well as for post-flight manual visual inspection by human personnel. In the presented pipeline, they are employed for storing the coordinates of the Earth surface locations that form the points of the polyline, delineating a crowd area.

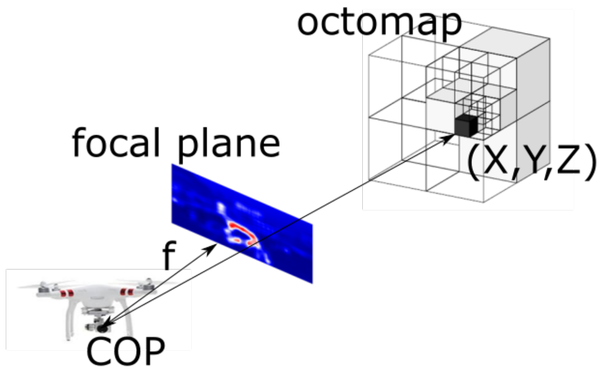


Fig. 3. Ray casting procedure for crowd heatmap contours projection onto the 3D occupancy grid of Octomap as semantic annotations.

4.7 Simple Path Planner (SPP)

Each path planned for a UAV must be safe, i.e., it has to ensure the lack of collisions with obstacles. Moreover, each path should also avoid no-fly zones (e.g., private areas, crowds, people, etc.) and not exceed the limits of a pre-defined geo-fencing area. In our example pipeline, the Simple Path Planner (SPP) module takes care of these tasks in an on-line manner.

In order to compute an obstacle-free path to the preferred closest landing site, SPP receives as input the available geometric map (an Octomap file in our implementation) and the semantic map, containing geo-referenced annotations of flight area limits, no-fly zones and 3D crowd polylines; the latter are continuously being extracted through the collaboration of CD, SRP and SMM.

SPP computes routes modeling the environment as a grid: the entire pre-defined area is bounded by a rectangle and then segmented into smaller quadrangular zones, limitations according to a given grid resolution. Each zone represents a cell of the grid. If a cell contains any vertex of a no-fly zone, or is just outside the geo-fencing area, that cell is set as an obstacle cell; otherwise the cell is set as free. The underlying path planning algorithm searches a route to the destination without using obstacle cells. Horizontal, vertical and diagonal movements between adjacent free cells are allowed. The grid search is carried out by an A* algorithm [37], i.e., a heuristic search algorithm formulated for weighted graphs. It searches within a graph with weighted edges the optimal path between two different nodes that minimizes the cost (planar distance, in this case) between them. Exhaustive search is avoided by relying on a heuristic.

4.8 The overall synthesis

The pipeline's operation is achieved through the synergy of the different modules described above, that must remain functional during the entire flight duration. The UAV camera constantly sends video frames to the Crowd/Person Detector modules (C/PD). SRP receives the 2D crowd heatmaps and person ROIs from C/PD and outputs all known crowd regions and person locations in 3D space, including the ones that have just been discovered. The SMM receives all crowd/person 3D annotations and updates the 3D area map used for navigation accordingly. It also has access to annotations regarding potential landing zones, pre-discovered off-line by the PLSD module before mission start. If the need for landing arises, the SPP plans a path to a known potential landing site, using the stored 3D area map and the vehicle's localization sensors. As soon as the target site becomes visible on the UAV camera, its actual eligibility for landing is verified via the VLSD module and, in case it is found to be unsuitable, a new path is requested from the SPP. The overall process is detailed below, divided into temporal stages which are actually interleaved to a limited degree:

Stage 1: The 3D map of the flight area (in Octomap form), along with the geo-referenced potential landing sites derived from PLSD, becomes available to the SMM. Computing both of them (off-line, before flight) requires the DSM and DTM height maps, which are publicly available with sufficient resolution for large parts of the globe. In a real-world scenario these maps may be outdated (e.g., an area's vegetation growth is highly volatile, even in a short time span).

Stage 2: During flight, the UAV constantly collects information about the presence of visible human crowds (using the CD module) or visible individual people (using the PD module), so as to avoid flying over them. However, once a safe landing spot is requested, the UAV must maintain a even higher awareness of its environment as far as people/crowds are concerned. Thus, it periodically hovers and rotates the camera's gimbal at multiple pitch angles scanning for visible people/crowds located either far-away or near-by. This information initially takes the form of 2D crowd heatmaps and person ROIs (in pixel coordinates), which are on-the-fly transformed into 3D annotated regions/locations in geo-referenced 3D coordinates by the SRP. Then, they are stored on-line by the SMM along with the 3D area map, thus updating it with current semantic annotations.

Stage 3: When the need for landing arises, the SPP constructs on-line a route towards the closest approachable known safe landing zone, using the 3D semantic annotations projected onto a 2D map. The flight controller is ordered to follow the planned route and, as soon as the landing zone becomes visible via the camera, its actual eligibility for landing is evaluated using the VLSD module. At all times during this process a new route may be requested by the SPP, in order to: (a) avoid newly identified no-fly zones, consisting of air-space near/above human crowds or individuals, or (b) select a different landing site site (among the ones pre-computed by PLSD), if the originally selected one is not judged by the VLSD to be actually eligible.

Stage 4: As the UAV flies towards a safe landing zone, it may dynamically perceive a nearby individual person/crowd. If the vehicle's position at that moment lies within the no-fly zone defined by the person/crowd, the UAV cancels the current trajectory, dismisses the remaining part of the planned path and immediately flies to the closest point outside the no-fly zone. Once this point has been reached, a new path to that target landing site is requested by the SPP.

4.9 Pipeline Performance Evaluation

To evaluate our example UAV safe landing pipeline, a set of realistic simulations were created. A combination of AirSim [125] and ROS was used to that end. A large-scale ($568 \times 379m^2$) realistic countryside environment (shown in Fig. 6) was populated with people and employed for the evaluation process. In order to adapt the CNN-based modules of the example pipeline to the visual differences between real-world data and the simulated world, the VLSD model was first fine-tuned for 20 epochs, using 600 images collected from the AirSim simulator and manually annotated. The learning rate and the batch size were again set to 0.01 and 8, respectively. Similarly, the PD CNN module was fine-tuned on a mixed dataset, consisting of manually collected images from aerial views and synthetic images generated using AirSim. No fine-tuning was performed for the CD model.

The NVIDIA Jetson Xavier embedded AI compute board was selected for executing the overall pipeline. A series of 15 independent missions (flight and landing) were simulated, using different randomly generated UAV starting points. Thus, each mission was conducted in a different part of the simulated environment with different characteristics in terms of terrain morphology and the presence of people in the area, ensuring testing under a variety of conditions. In all cases, the UAV had access to the DTM and the DSM of the simulated environment. With the modules described above operating in synergy, the UAV would fly from its starting point towards the closest safe landing spot, avoiding no-fly zones and, finally, land on it. In certain cases, the vehicle would have to on-line re-plan its path and land on a different landing site, since the original would be deemed as ineligible. The overall system was able to run steadily on NVIDIA Jetson Xavier with the most demanding modules, namely the PD and VLSD modules, achieving 9.5 FPS and 0.82 FPS respectively. In addition, the average number of times that a path was requested from SPP in a single mission was 1.53. This evaluation was performed using only individual people and the PD module, but similar results were obtained using human crowds and the CD module.

Performance of our example UAV safe landing pipeline was evaluated using the minimum lateral (on-ground projection) distance of the UAV trajectory waypoints from any person location in the terrain of the flight environment as a metric, as shown in Figure 4a. High minimum distance ensures the avoidance of flying over humans and, thus, the overall safety of the autonomous landing process. Based on [134], the minimum legitimate lateral flight distance from individual people for several countries, including UK, Italy and Australia, is 30-50m. Thus, we set the minimum acceptable lateral distance to 50m. In our evaluation setup, relying on the vision-based detection of individuals from the PD in a scenario taking place at a mountainous environment populated with people, the mean minimum UAV-to-person distance along the actual UAV trajectory (averaged over all 15 independent missions) is 69.4637m, as shown in Figure 4a for several safe landing cases. Therefore, the minimum is exceeded on average by almost 20 meters.

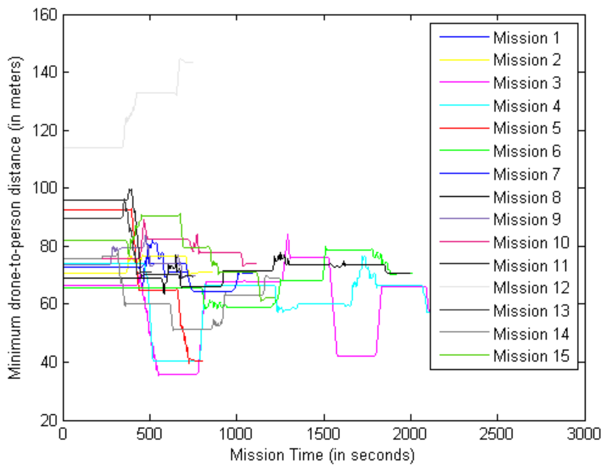
The graph in Figure 4a depicts the minimum lateral distance of the UAV trajectory waypoints from people, varying over time. The straight horizontal lines of the graph represent the time required for executing Stage 2 of the pipeline. Note that, rarely, the minimum drone-to-person distance falls below the minimum acceptable threshold for brief intervals (e.g., in mission 3). This may occur as an individual person or a crowd could not be detected until that point in time, e.g., due to visual occlusions. However, as soon as the person/crowd is detected, the UAV automatically moves away from it (Stage 4), if necessary, and a new path is requested from SPP.

4.9.1 Ablation Study. An ablation study was conducted to determine the efficiency and the relative importance of individual pipeline components. Figure 4b depicts the minimum lateral distance of UAV trajectory waypoints from undetected people, over time, if person/crowd detection (Stage 2) is disabled. As it can be inferred from these plots, the minimum lateral distance achieved is heavily decreased in comparison to Figure 4a. Evidently, C/PD is the most important module of the described safety pipeline, as expected, since the vehicle may fly near people without it. On the other hand, omitting only the VLSD does not incur severe negative impact, since the PLSD is still employed for determining possible landing sites in terms of ground morphology; the only loss is that any rigid obstacles/water/cars that may lie on the pre-identified landing spots are not visually detected, thus there is an increased risk of inappropriate landing. SPP or SMM-SRP cannot be omitted, since they are fundamental to the pipeline's operation.

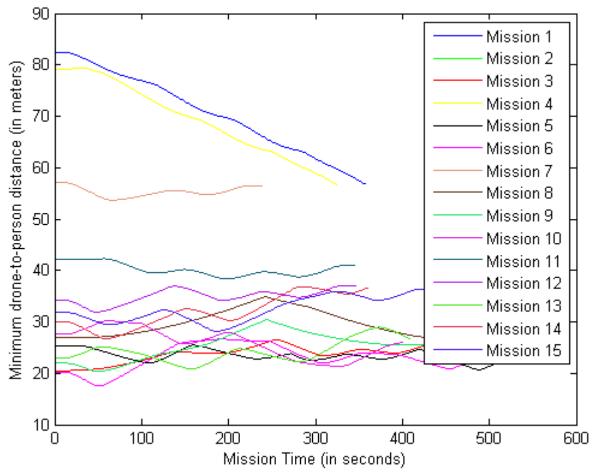
4.9.2 Example missions. Four example missions are visualized in Figure 7. In all of them, the UAV constantly detects people while it is flying and projects their 2D ROIs on the 3D navigation map as annotations. When it needs to land, it plans a trajectory to the closest a-priori known landing site utilizing the SPP and avoiding people. The trajectory consists in a smooth path of waypoints. Once the UAV is close to its target landing site, its actual eligibility for landing is evaluated using the VLSD module.

Mission A is a simple example mission. The SPP provides a path towards the closest known landing site E_1 (Figure 7b). Following that path, the UAV flies from point S to point E (Figure 7a) and lands safely, since the VLSD module has evaluated the validity of the site.

The PD\CD modules and their successful utilization is shown in Mission B. The SPP provides a path from point S_1 to point E_1 at the start of the mission (Figure 7d). While the UAV reaches E_1, a human presence is detected at point H_2. The UAV redefines its no-fly zones and perceives that it is currently flying within one. The UAV dismisses the rest of the path and flies towards the closest fly-zone (Stage 4). Then, the SPP provides a new path from point S_2 to point E_2 (Figure 7e). Finally the UAV follows that path and lands safely, since the VLSD has evaluated the validity of the site. The UAV trajectory in this corresponding mission is visualized in (Figure 7c).



(a)



(b)

Fig. 4. [a]Minimum UAV-to-person distance (in meters) vs Mission time (in seconds), [b]Minimum UAV-to-person distance (in meters) vs Mission time (in seconds) with Stage 2 disabled.

Mission C and Mission D are two example missions where the use of the VLSD module is evident. In these missions, we simulate the case where the DSMs are outdated and several potential landing sites, computed by the PLSD module, are in fact sites occupied with water, vegetation or buildings; thus the UAV should avoid landing on them. In Mission C, the SPP provides a path from point S_1 to point E_1 at the start of the mission (Figure 7g). Once the UAV reaches E_1, the VLSD module rejects its validity, as a tree is located on that point. The annotations regarding the landing-sites are redefined and the SPP module computes a path to a different landing site E_2

(Figure 7h). The UAV follows that path and lands safely, since the VLSD has evaluated the validity of the site. The UAV trajectory during Mission C is visualized in (Figure 7f). Similarly, in Mission D, the SPP provides a path towards a landing site, but, once reached, the latter is judged by the VLSD module as occupied by water and, therefore, as unsuitable (Figures 7j, 7k). Subsequently, SPP is called again and finally, the UAV lands successfully on point E. The UAV trajectory during Mission D is visualized in (Figure 7i).

Table 6. A comparison of required runtimes. Performance of our on-line modules was measured on an embedded AI computer (NVIDIA Jetson Xavier) while reported figures for [94]/[86] refer to execution on regular desktop computer/NVIDIA Jetson TX2, respectively. PLSD is the only component of the pipeline presented here that is being executed off-line/pre-flight, on a desktop computer.

	On-line	Off-line	FPS/Hz
PLSD		✓	0.68
VLSD	✓		0.82
C/PD	✓		9.5
SRP+SMM	✓		19
SPP	✓		1.53
SafeUAV-Net-Large [86]	✓		35
SafeUAV-Net-Small [86]	✓		130
[94]	✓		5.76

4.9.3 Response Time Study. Embedded AI compute boards, such as the popular Jetson Xavier, are incapable of executing all pipeline functionalities in real-time, due to their limited computational resources. Thus, on-the-fly running all of the presented components on current commercial UAV hardware in real-time is impossible. However, the demanding PLSD module can be executed off-line, at the pre-flight stage and on conventional hardware; no real-time requirements are imposed to it. Using PLSD to process the map of the simulated region we employed for the experimental evaluation (with an area of $568 \times 379m^2$) required 1.456 sec (corresponding to 0.68 Hz), on a typical desktop computer. Moving on to the on-board pipeline components, the most computationally demanding module proved to be VLSD, which achieves a performance of 0.82 FPS on a Jetson Xavier. The example pipeline is designed to only rarely require activation of the VLSD module and does not depend on its real-time on-board execution. In fact, a processing rate of approximately 1 FPS, or even less, is adequate given the rather low speed of typical multicopters and the role of the VLSD in the overall pipeline; analyzing a few images of each pre-identified landing site, as the UAV approaches it, is enough to judge whether it is indeed suitable for landing or not.

On the other hand, the critical C/PD modules were jointly measured to perform at a rate of 9.5 FPS on a Jetson Xavier, which is obviously lower than real-time, but adequate given typical multicopter flying speeds. Moreover, output frequency of our multithreaded SMM+SRP implementation was measured at approximately 19Hz on average. Although the input frequency to SMM is equal to the output frequency of C/PD (9.5Hz), the SMM implementation publishes a new overall crowd map as soon as a new crowd polygon has been evaluated/projected, i.e., it publishes multiple crowd maps per processed video frame (if more than one crowds are visible in the scene). Finally, the Simple Path Planner (SPP) is executed on-board a Jetson Xavier at a rate of 1.53Hz. This frequency permits normal operation, since SPP is only sporadically executed according to the pipeline design, thus no near-real-time requirements are imposed to it. These findings are summarized in Table 6.

4.9.4 Pipeline Comparison Study. There are no publicly presented, comprehensive vision-based UAV safety pipelines such as the example of this paper, since most relevant literature is only concerned with specific subtasks. Still, for comparison purposes, [86] (briefly described in Subsection 3.3.2) was selected to be juxtaposed against our example pipeline. The main advantage of the latter one is the presence of the C/PD, VLSD and SPP modules, which act jointly to heavily increase safety and robustness in case of human presence and obstacles in the area. However, since [86] is limited to depth and binary semantic map estimation (with the two semantic classes being “obstacle" and “safe") using a lightweight, embedded CNN, it runs significantly faster than our complete pipeline, whose computational complexity increases along with its capabilities. Moreover, our example pipeline needs a-prior known area morphology DSM/DTM maps for the off-line PLSD module to be executed, while [86] performs a task similar to PLSD, but in an on-line manner and in unknown areas.

More similar to the presented example pipeline is [94], which still however is significantly different. It builds the area map dynamically and on-the-fly, but it requires a stereoscopic camera in order to be able to do that. In contrast, our example pipeline relies on pre-existing and a-priori known DSM/DTM maps, but only demands a conventional monocular RGB camera mounted on the UAV during flight. [94] also employs an on-board path planner for computing trajectories towards safe landing sites, but actual landing site identification relies only on geometrical properties (e.g., flatness) and not on semantics. In contrast, the example pipeline presented here pre-identifies candidate landing sites using geometric properties at an off-line stage (via the PLSD module) and refines this list using on-the-fly semantic properties identified during flight (via the vision-based VLSD module). Finally, [94] does not employ at all a vision-based module similar to our C/PD, because it presupposes that the UAV is equipped with a bioradar. In contrast, our pipeline relies only on a monocular RGB camera. With regard to runtime requirements, [94] demands 173.4ms per video frame (corresponding to 5.76 FPS) when running on a desktop computer (Intel Core i7 CPU). In contrast, our complete example pipeline is executed at a rate of 0.82 FPS on a Jetson Xavier, when taking into account all of its on-board components. However, the pipeline presented here is heavily modular and, thus, the most critical components (C/PD) can independently run and update the 3D map at a rate of 9.5 FPS on a Jetson Xavier.

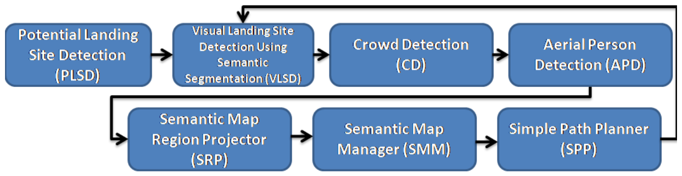

Potential Landing Site Detection (PLSD)
Visual Landing Site Detection Using Semantic Segmentation (VLSD)
Crowd Detection (CD)
Aerial Person Detection (APD)
Semantic Map Region Projector (SRP)
Semantic Map Manager (SMM)
Simple Path Planner (SPP)

Fig. 5. Overview of the presented UAV safe landing pipeline example.

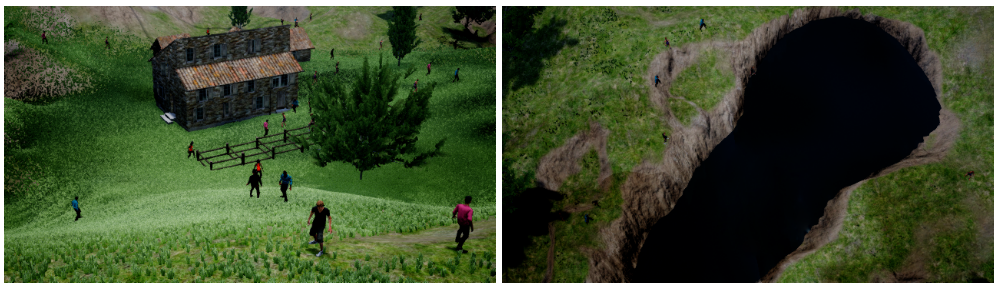


Fig. 6. Countryside environments used in simulations.

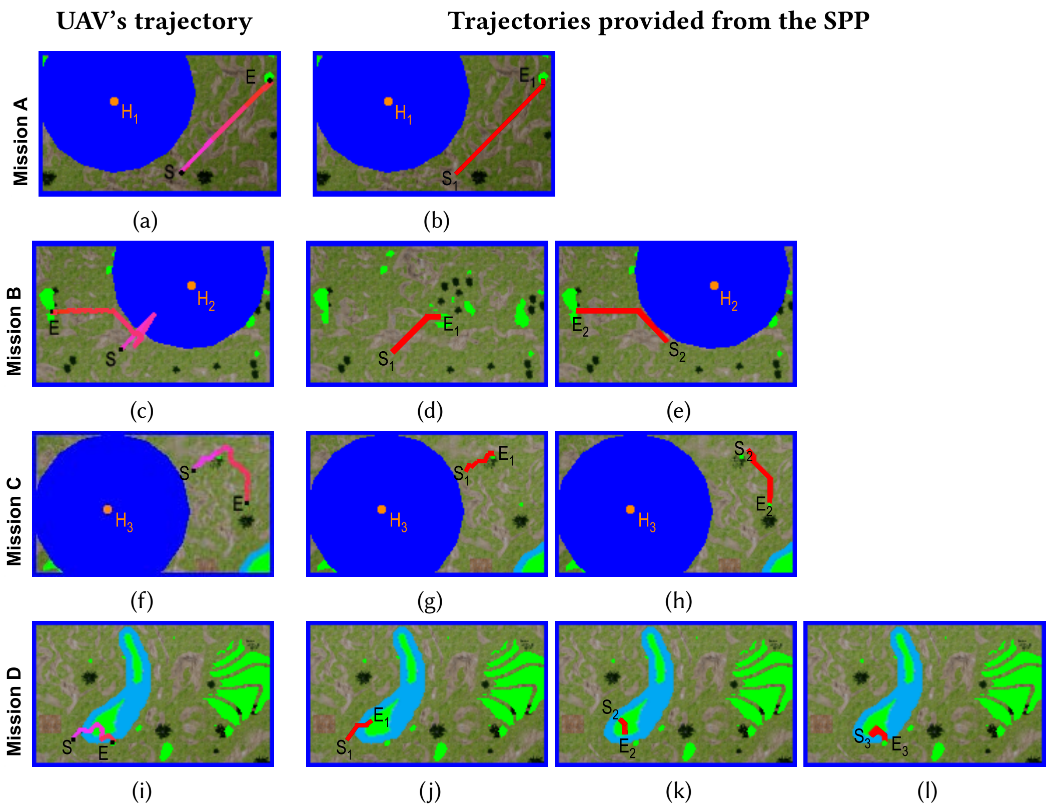


Fig. 7. Each row represents a separate UAV mission. The first column depicts the UAV's trajectories, while the rest depict the trajectories provided from the SPP ordered based on the time they were requested. In each Subfigure, the UAV's starting locations are marked with the letter "S", the UAV's target locations are marked with the letter "E", the detected people are marked with the letter "H", the pre-computed landing sites from Stage 1 are colored in light-green, areas surrounded by water are highlighted as light-blue and the detected no-fly zones (humans or borders) are colored in dark-blue.

5 CONCLUSIONS

During the last decade, adoption of UAVs with autonomous functionalities has been steadily increasing. This is due to their numerous applications in transport, surveillance, mapping, manufacturing, healthcare, agriculture, infrastructure inspection, media production, etc. Successful deployment of such autonomous systems involves tackling a number of challenging tasks, while concurrently assuring human safety. A subset of these tasks, including UAV obstacle avoidance, localization, mapping, landing site detection, target detection/tracking, etc., can be addressed successfully by using computer vision/machine learning methodologies, provided that they respect constraints and limitations inherent to embedded systems, such as real-time operation, limited available computing resources, need for robustness in real-life scenarios, etc. The aim of this overview was to provide a coherent coverage of these methods under the unifying premise of UAV flight/landing safety assurance. Additionally, an example computer vision-based autonomous UAV safe landing algorithmic pipeline was presented and evaluated in simulation, using actual embedded AI compute hardware. Autonomous flight and landing safety is maximized by on-line extraction of rich scene semantics. Results indicate successful, real-time operation with automatic conformance to safety regulations.

REFERENCES

[1] 2016. The Air Navigation Order. (2016). http://www.legislation.gov.uk/uksi/2016/765/made
[2] 2016. ANACOM - Regulamento n. 1093/2016, de 14 de dezembro. (2016). https://www.anacom.pt/render.jsp?contentId=1401209
[3] 2017. ArcMap: How Slope works. http://desktop.arcgis.com/en/arcmap/10.3/tools/spatial-analyst-toolbox/how-slope-works.htm. Accessed: 14/12/2017.
[4] 2020. Drones-Trabajos areos-Compaas o empresas-AESA-Agencia Estatal deSeguridad Area - Ministerio de Fomento. (2020). http://www.seguridadaerea.gob.es/langcastellano/ciasempresas/trabajos/rpas/default.aspx
[5] Apr. 2017. Verordnung zur Regelung des Betriebs von unbemannten Fluggerten. 17 (Apr. 2017), 6831. http://www.bgbl.de/xaver/bgbl/start.xav?startbk=BundesanzeigerBGBl&jumpTo=bgbl117s0683.pdf
[6] Mar. 2017. FFD, Fdration Franaise de Drone (FFD), Nouveau guide DGAC , Activits Particulires, v1.2 (10 Janvier 2017). (Mar. 2017). /guide-dgac-activites-particulieres-v1-2/
[7] R. Acuna, D. Zhang, and V. Willert. 2018. Vision-based UAV landing on a moving platform in GPS denied environments using motion prediction. In *Proceedings of the Latin American Robotic Symposium, Brazilian Symposium on Robotics (SBR) and Workshop on Robotics in Education (WRE)*.
[8] M. Agrawal, K. Konolige, and M. R. Blas. 2008. Censure: Center surround extremas for realtime feature detection and matching. In *Proceedings of European Conference on Computer Vision, (ECCV)*. Springer.
[9] W. G. Aguilar, V. P. Casaliglla, and J. L. Pólit. 2017. Obstacle avoidance for low-cost UAVs. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC)*.
[10] T. Ahonen, A. Hadid, and M. Pietikainen. 2006. Face description with local binary patterns: Application to face recognition. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 12 (2006), 2037–2041.
[11] S. Ahrens, D. Levine, G. Andrews, and J. P. How. 2009. Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.
[12] Y. Amit and D. Geman. 1999. A computational model for visual selection. *Neural computation* 11, 7 (1999), 1691–1715.
[13] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena. 2013. Contextually guided semantic labeling and search for three-dimensional point clouds. *The International Journal of Robotics Research* 32, 1 (2013), 19–34.
[14] M. Aydin and E. Kugu. 2016. Finding smoothness area on the topographic maps for the unmanned aerial vehicle's landing site estimation. In *Proceedings of the IEEE International Conference on Digital Information and Communication Technology and its Applications (DICTAP)*.
[15] M. Barekatain, M. Martí, H.F. Shih, S. Murray, K. Nakayama, Y. Matsuo, and H. Prendinger. 2017. Okutama-action: an aerial view video dataset for concurrent human action detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.
[16] F. Bernuy and J. Ruiz del Solar. 2015. Semantic mapping of large-scale outdoor scenes for autonomous off-road driving. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*. IEEE.
[17] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. 2017. Unsupervised pixel-level domain adaptation with Generative Adversarial Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[18] J. Byrne, M. Cosgrove, and R. Mehra. 2006. Stereo based obstacle detection for an unmanned air vehicle. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.
[19] D. Cavaliere, V. Loia, A. Saggese, S. Senatore, and M. Vento. 2017. Semantically enhanced UAVs to increase the aerial scene understanding. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 49, 3 (2017), 555–567.
[20] Y.-C. Chang, H.-T. Chen, J.-H. Chuang, and I.-C. Liao. 2018. Pedestrian Detection in Aerial Images Using Vanishing Point Transformation and Deep Learning. In *Proceedings of the IEEE International Conference on Image Processing (ICIP)*.
[21] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.
[22] S. Chen, L. Han, X. Liu, Z. He, and X. Yang. 2020. Subspace distribution adaptation frameworks for domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems* 31, 12 (2020), 5204–5218.
[23] A. B. Curtis. 2008. Path planning for unmanned air and ground vehicles in urban environments. (2008).
[24] N. Dalal and B. Triggs. 2005. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*.
[25] R. de Nijs, S. Ramos, G. Roig, X. Boix, L. Van Gool, and K. Kühnlenz. 2012. On-line semantic perception using uncertainty. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.
[26] X. Deng and Q. Zeng. 2013. Research on laser-assisted odometry of indoor UAV with monocular vision. In *Proceedings of the IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems*.
[27] V. R Desaraju, N. Michael, M. Humenberger, R. Brockers, S. Weiss, and L. H. Matthies. 2014. Vision-based Landing Site Evaluation and Trajectory Generation Toward Rooftop Landing. In *Robotics: Science and Systems*.
[28] DJI. 2021. Protecting The Skies In The Drone Era, Elevating Safety. (2021).

[29] P. Dollár, R. Appel, S. Belongie, and P. Perona. 2014. Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 8 (2014), 1532–1545.

[30] P. Dollár, R. Appel, and W. Kienzle. 2012. Crosstalk cascades for frame-rate pedestrian detection. In *Proceedings of European Conference on Computer Vision (ECCV)*. Springer.

[31] P. Dollár, S. Belongie, and P. Perona. 2010. The fastest pedestrian detector in the west. (2010).

[32] P. Dollár, Z. Tu, P. Perona, and S. Belongie. 2009. Integral channel features. (2009).

[33] Y. Dong, C. Fu, and E. Kayacan. 2016. RRT-based 3D path planning for formation landing of quadrotor UAVs. In *Proceedings of the IEEE International Conference on Control, Automation, Robotics and Vision (ICARCV)*.

[34] S. Dotenco, F. Gallwitz, and E. Angelopoulou. 2014. Autonomous approach and landing for a low-cost quadrotor using monocular cameras. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer.

[35] D. H Douglas and T. K. Peucker. 1973. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization* 10, 2 (1973), 112–122.

[36] B. Douillard, D. Fox, F. Ramos, et al. 2008. Laser and Vision Based Outdoor Object Mapping.. In *Robotics: Science and Systems*, Vol. 8.

[37] F. Duchoň, A. Babinec, M. Kajan, P. Beňo, M. Florek, T. Fico, and L. Jurišica. 2014. Path planning with modified a star algorithm for a mobile robot. *Procedia Engineering* 96 (2014), 59–69.

[38] R.O. Duda and P.E. Hart. 1973. *Pattern Classification and Scene Analysis.* Wiley.

[39] P. F. Felzenszwalb, D. A. McAllester, D. Ramanan, et al. 2008. A discriminatively trained, multiscale, deformable part model. In *Proceeding of the IEEE Conference on Computer Vision and Pattern Recognition CVPR*.

[40] S. Friedman, H. Pasula, and D. Fox. 2007. Voronoi Random Fields: Extracting Topological Structure of Indoor Environments via Place Labeling. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*.

[41] A.-J. Gallego, J. Calvo-Zaragoza, and R. B. Fisher. 2020. Incremental unsupervised domain-adversarial training of neural networks. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[42] M. Garg, A. Kumar, and PB. Sujit. 2015. Terrain-based landing site selection and path planning for fixed-wing UAVs. In *Proceedings of the IEEE International Conference on Unmanned Aircraft Systems (ICUAS)*.

[43] R. Girshick. 2015. Fast R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

[44] R. Girshick, J. Donahue, T. Darrell, and J. Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[45] A. S. Glassner. 1989. *An introduction to ray tracing.* Elsevier.

[46] Z. Gosiewski, J. Ciesluk, and L. Ambroziak. 2011. Vision-based obstacle avoidance for unmanned aerial vehicles. In *Proceedings of the IEEE International Congress on Image and Signal Processing, (ICISP)*.

[47] D. Guan, X. Luo, Y. Cao, J. Yang, Y. Cao, G. Vosselman, and M. Ying Yang. 2019. Unsupervised domain adaptation for multispectral pedestrian detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

[48] X. Guo, D. Dean, S. Denman, C. Fookes, and S. Sridharan. 2011. Evaluating automatic road detection across a large aerial imagery collection. In *Proceedings of the IEEE International Conference on Digital Image Computing: Techniques and Applications, (DICTA)*.

[49] X. Guo, S. Denman, C. Fookes, L. Mejias, and S. Sridharan. 2014. Automatic UAV forced landing site detection using machine learning. In *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications (DICTA)*.

[50] X. Guo, S. Denman, C. Fookes, and S. Sridharan. 2016. A robust UAV landing site detection system using mid-level discriminative patches. In *Proceedings of the IEEE International Conference on Pattern Recognition (ICPR)*.

[51] H. He and B. Upcroft. 2013. Nonparametric semantic segmentation for 3D street scenes. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

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

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

[54] L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys. 2011. Autonomous obstacle avoidance and maneuvering on a vision-guided MAV using on-board processing. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

[55] T. Hinzmann, T. Stastny, C. Cadena, R. Siegwart, and I. Gilitschenski. 2018. Free LSD: Prior-free visual landing site detection for autonomous planes. *IEEE Robotics and Automation Letters* 3, 3 (2018), 2545–2552.

[56] H. Hirschmüller, P. R. Innocent, and J. Garibaldi. 2002. Real-time correlation-based stereo vision with reduced border errors. *International Journal of Computer Vision* 47, 1-3 (2002), 229–246.

[57] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. 2013. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots* 34, 3 (2013), 189–206.

[58] J. Hosang, M. Omran, R. Benenson, and B. Schiele. 2015. Taking a deeper look at pedestrians. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[59] S. Hrabar, G. S. Sukhatme, P. Corke, K. Usher, and J. Roberts. 2005. Combined optic-flow and stereo-based navigation of urban canyons for a UAV. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[60] J. Hu, A. Razdan, J. C. Femiani, M. Cui, and P. Wonka. 2007. Road network extraction and intersection detection from aerial images by tracking road footprints. *IEEE Transactions on Geoscience and Remote Sensing* 45, 12 (2007), 4144–4157.

[61] A. E. Johnson, A. R. Klumpp, J. B. Collier, and A. A. Wolf. 2002. LiDAR-based hazard avoidance for safe landing on Mars. *Journal of Guidance, Control, and Dynamics* 25, 6 (2002), 1091–1099.

[62] E. Kakaletsis and N. Nikolaidis. 2019. Potential UAV landing sites detection through Digital Elevation Models analysis. In *Proceedings of the European Signal Processing Conference (EUSIPCO), Satellite Workshop: Signal Processing, Computer Vision and Deep Learning for Autonomous Systems*. IEEE.

[63] E. Kakaletsis, M. Tzelepi, P. I. Kaplanoglou, C. Symeonidis, N. Nikolaidis, A. Tefas, and I. Pitas. 2019. Semantic map annotation through UAV video analysis using deep learning models in ROS. In *Proceedings of the International Conference on Multimedia Modeling (MMM)*. Springer.

[64] R. Kala. 2013. Rapidly exploring random graphs: motion planning of multiple mobile robots. *Advanced Robotics* 27, 14 (2013), 1113–1122.

[65] Q. Kang, S. Yao, M. C. Zhou, K. Zhang, and A. Abusorrah. 2020. Effective Visual Domain Adaptation via Generative Adversarial Distribution Matching. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[66] I. Karakostas, I. Mademlis, N. Nikolaidis, and I. Pitas. 2020. Shot type constraints in UAV cinematography for autonomous target tracking. *Information Sciences* 506 (2020), 273–294.

[67] J. Kennedy and R. Eberhart. 1995. Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks (ICNN)*.

[68] A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. Imagenet classification with deep Convolutional Neural Networks. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*.

[69] W. Lan, J. Dang, Y. Wang, and S. Wang. 2018. Pedestrian Detection Based on YOLO network model. In *Proceedings of the IEEE International Conference on Mechatronics and Automation (ICMA)*.

[70] S. Lazebnik, C. Schmid, and J. Ponce. 2006. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[71] B. Le Saux and M. Sanfourche. 2013. Rapid semantic mapping: Learn environment classifiers on the fly. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*.

[72] J.-O. Lee, K.-H. Lee, S.-H. Park, S.-G. Im, and J. Park. 2011. Obstacle avoidance for small UAVs using monocular vision. *Aircraft Engineering and Aerospace Technology* 83, 6 (2011), 397–406.

[73] T. Liu and T. Stathaki. 2017. Enhanced pedestrian detection using deep learning based semantic image segmentation. In *Proceedings of the IEEE International Conference on Digital Signal Processing (DSP)*.

[74] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. 2016. Ssd: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer.

[75] J. Long, E. Shelhamer, and T. Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[76] David G. Lowe. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60, 2 (2004), 91–110.

[77] B. D. Lucas and T. Kanade. 1981. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence, (IJCAI)*.

[78] J.-Q. Ma. 2009. Content-based image retrieval with HSV color space and texture features. In *Proceedings of the IEEE International Conference on Web Information Systems and Mining*.

[79] I. Mademlis, V. Mygdalis, N. Nikolaidis, M. Montagnuolo, F. Negro, A. Messina, and I. Pitas. 2019. High-level multiple-UAV cinematography tools for covering outdoor events. *IEEE Transactions on Broadcasting* 65, 3 (2019), 627–635.

[80] I. Mademlis, V. Mygdalis, N. Nikolaidis, and I. Pitas. 2018. Challenges in autonomous UAV cinematography: an overview. In *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*.

[81] I. Mademlis, N. Nikolaidis, A. Tefas, I. Pitas, T. Wagner, and A. Messina. 2018. Autonomous unmanned aerial vehicles filming in dynamic unstructured outdoor environments. *IEEE Signal Processing Magazine* 36 (2018), 147–153. Issue 1.

[82] I. Mademlis, N. Nikolaidis, A. Tefas, I. Pitas, T. Wagner, and A. Messina. 2019. Autonomous UAV cinematography: a tutorial and a formalized shot type taxonomy. *Comput. Surveys* 52, 5 (2019), 105.

[83] I. Mademlis, P. Nousi, C. Le Barz, T. Gonçalves, and I. Pitas. 2019. Communications for autonomous Unmanned Aerial Vehicle fleets in outdoor cinematography applications. In *Proceedings of the EURASIP European Signal Processing Conference (EUSIPCO) Satellite Workshop: Signal Processing, Computer Vision and Deep Learning for Autonomous Systems*.

[84] I. Mademlis, A. Torres-González, J. Capitán, R. Cunha, B. Guerreiro, A. Messina, F. Negro, C. Le Barz, T. Gonçalves, A. Tefas, and I. Pitas. 2019. A multiple-UAV software architecture for autonomous media production. In *Proceedings of the EURASIP European Signal Processing Conference (EUSIPCO) Satellite Workshop: Signal Processing, Computer Vision and Deep Learning for Autonomous Systems*.

[85] D. Magree, J. G. Mooney, and E. N. Johnson. 2013. Monocular visual mapping for obstacle avoidance on UAVs. In *Proceedings of the IEEE International Conference on Unmanned Aircraft Systems (ICUAS).*

[86] A. Marcu, D. Costea, V. Licaret, M. Pîrvu, E. Slusanschi, and M. Leordeanu. 2018. SafeUAV: Learning to estimate depth and safe landing areas for UAVs from synthetic data. In *Proceedings of the European Conference on Computer Vision (ECCV).*

[87] D. Maturana, S. Arora, and S. Scherer. 2017. Looking forward: A semantic mapping system for scouting with micro-aerial vehicles. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).*

[88] D. Maturana and S. Scherer. 2015. 3D convolutional neural networks for landing zone detection from LiDAR. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA).*

[89] D. F. Maune. 2007. *Digital Elevation Model technologies and applications: the DEM users manual.* ASPRS Publications.

[90] A. Mcfadyen, L. Mejias, P. Corke, and C. Pradalier. 2013. Aircraft collision avoidance using spherical visual predictive control and single point features. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[91] L. Mejias and D. Fitzgerald. 2013. A multi-layered approach for site detection in UAS emergency landing scenarios using geometry-based image segmentation. In *Proceedings of the IEEE International Conference on Unmanned Aircraft Systems (ICUAS).*

[92] C. L. Miller and R. A. Laflamme. 1958. *The Digital Terrain Model-: Theory & Application.* MIT Photogrammetry Laboratory.

[93] N. Mitsou, R. de Nijs, D. Lenz, J. Frimberger, D. Wollherr, K. Kühnlenz, and C. Tzafestas. 2012. Online semantic mapping of urban environments. In *Proceedings of the International Conference on Spatial Cognition (ICSC).* Springer.

[94] M. Mittal, R. Mohan, W. Burgard, and A. Valada. 2019. Vision-based autonomous UAV navigation and landing for urban search and rescue. *arXiv preprint arXiv:1906.01304* (2019).

[95] R. Mur-Artal, J. M. M. Montiel, and J.D. Tardos. 2015. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics* 31, 5 (2015), 1147–1163.

[96] F. Nex and F. Remondino. 2014. UAV for 3D mapping applications: a review. *Applied Geomatics* 6, 1 (2014), 1–15.

[97] P. Nousi, I. Mademlis, I. Karakostas, A. Tefas, and I. Pitas. 2019. Embedded UAV real-time visual object detection and tracking. In *Proceedings of the IEEE International Conference on Real-time Computing and Robotics (RCAR).*

[98] A. Nurhadiyatna and S. Lončarić. 2017. Semantic image segmentation for pedestrian detection. In *Proceedings of the IEEE International Symposium on Image and Signal Processing and Analysis.*

[99] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto. 2017. Voxblox: Incremental 3D euclidean signed distance fields for on-board MAV planning. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).*

[100] M. A. Olivares-Mendez, L. Mejias, P. Campoy, and I. Mellado-Bataller. 2012. Quadcopter see and avoid using a fuzzy controller. In *Uncertainty Modeling in Knowledge Engineering and Decision Making.* World Scientific, 1239–1244.

[101] F. Pan, I. Shin, F. Rameau, S. Lee, and I. S. Kweon. 2020. Unsupervised intra-domain adaptation for semantic segmentation through self-supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

[102] D. Pangercic, B. Pitzer, M. Tenorth, and M. Beetz. 2012. Semantic object maps for robotic housework-representation, acquisition and use. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).*

[103] C. Papageorgiou and T. Poggio. 2000. A trainable system for object detection. *International Journal of Computer Vision* 38, 1 (2000), 15–33.

[104] C. Papaioannidis, I. Mademlis, and I. Pitas. 2021. Autonomous UAV safety by visual human crowd detection using multi-task Deep Neural Networks. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).*

[105] G. Papandreou, T. Zhu, L.-C. Chen, S. Gidaris, J. Tompson, and K. Murphy. 2018. Personlab: Person pose estimation and instance segmentation with a bottom-up, part-based, geometric embedding model. In *Proceedings of the European Conference on Computer Vision (ECCV).*

[106] F. Patrona, I. Mademlis, A. Tefas, and I. Pitas. 2019. Computational UAV cinematography for intelligent shooting based on semantic visual analysis. In *Proceedings of the IEEE International Conference on Image Processing (ICIP).*

[107] R. Paul, R. Triebel, D. Rus, and P. Newman. 2012. Semantic categorization of outdoor scenes with uncertainty estimates using multi-class Gaussian process classification. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[108] R. Polastro, F. Corrêa, F. Cozman, and J. Okamoto. 2010. Semantic mapping with a probabilistic description logic. In *Proceeding of the Brazilian Symposium on Artificial Intelligence.* Springer.

[109] G. Priestnall, J. Jaafar, and A. Duncan. 2000. Extracting urban features from LiDAR digital surface models. *Computers, Environment and Urban Systems* 24, 2 (2000), 65–78.

[110] A. Pronobis and P. Jensfelt. 2012. Large-scale semantic mapping and reasoning with heterogeneous modalities. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).*

[111] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. 2009. ROS: an open-source Robot Operating System. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) Workshop on Open Source Robotics.*

[112] Joseph R. and Ali F. 2018. YOLOv3: an incremental improvement. *ArXiv* abs/1804.02767 (2018).

[113] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. 2016. You Only Look Once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[114] E. Remolina and B. Kuipers. 2004. Towards a general theory of topological maps. *Artificial Intelligence* 152, 1 (2004), 47–104.
[115] S. Ren, K. He, R. Girshick, and J. Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*.
[116] V. Roberge, M. Tarbouchi, and G. Labonté. 2012. Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning. *IEEE Transactions on Industrial Informatics* 9, 1 (2012), 132–141.
[117] Scott D. Roth. 1982. Ray casting for modeling solids. *Computer graphics and image processing* 18, 2 (1982), 109–144.
[118] Z. Saishang, C. Yifu, Z. Min, X. Zhong, and L. Can. 2015. MSURF: A new image matching algorithm which combines homography and SURF algorithm. In *Proceedings of the IEEE International Conference on Geoinformatics*.
[119] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. 2018. MobileNetV2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[120] A. Satpathy, X. Jiang, and H.-L. Eng. 2014. LBP-based edge-texture features for object recognition. *IEEE Transactions on Image Processing* 23, 5 (2014), 1953–1964.
[121] S. Scherer, L. Chamberlain, and S. Singh. 2012. Autonomous landing at unprepared sites by a full-scale helicopter. *Robotics and Autonomous Systems* 60, 12 (2012), 1545–1562.
[122] S. Sengupta, E. Greveson, A. Shahrokni, and P. HS. Torr. 2013. Urban 3D semantic modelling using stereo vision. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.
[123] S. Sengupta, P. Sturgess, L. Ladickỳ, and P. HS. Torr. 2012. Automatic dense visual semantic mapping from street-level imagery. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*.
[124] P. Serra, R. Cunha, T. Hamel, D. Cabecinhas, and C. Silvestre. 2016. Landing of a quadrotor on a moving target using dynamic image-based visual servo control. *IEEE Transactions on Robotics* 32, 6 (2016), 1524–1535.
[125] S. Shah, D. Dey, C. Lovett, A. Kapoor, and W. Burgard. 2017. AirSim: High-fidelity visual and physical simulation for autonomous vehicles. In *Proceedings of the Field and Service Robotics Conference*.
[126] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani. 2019. Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges. *IEEE Access* 7 (2019), 48572–48634.
[127] L. Shams and J. Spoelstra. 1996. Learning Gabor-based features for face detection. In *Proceedings of the World Congress in Neural Networks*. International Neural Network Society.
[128] J. Shotton, J. Winn, C. Rother, and A. Criminisi. 2006. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer.
[129] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. 2017. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
[130] K. Simonyan and A. Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).
[131] G. Singh and J. Košecká. 2012. Acquiring semantics induced topology in urban environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.
[132] Skydio. [n.d.]. Skydio R1. https://robots.ieee.org/robots/skydior1/.
[133] M. V. Srinivasan, S. Thurrowgood, and D. Soccol. 2006. An optical system for guidance of terrain following in UAVs. In *Proceedings of the IEEE International Conference on Video and Signal Based Surveillance*.
[134] C. Stöcker, R. Bennett, F. Nex, M. Gerke, and J. Zevenbergen. 2017. Review of the Current State of UAV Regulations. *Remote Sensing* 9 (2017), 459.
[135] C. Symeonidis, I. Mademlis, N. Nikolaidis, and I. Pitas. 2019. Improving neural Non-Maximum Suppression for object detection by exploiting interest-point detectors. In *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*.
[136] X. Tan and W. Triggs. 2010. Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Transactions on Image Processing, (TIP)* 19, 6 (2010), 1635–1650.
[137] Y.-J. Tsai, C.-S. Lee, C.-L. Lin, and C.-H. Huang. 2015. Development of flight path planning for multirotor aerial vehicles. *Aerospace* 2, 2 (2015), 171–188.
[138] M. Tzelepi and A. Tefas. 2017. Human crowd detection for drone flight safety using Convolutional Neural Networks. In *Proceedings of the EURASIP European Signal Processing Conference (EUSIPCO)*. IEEE.
[139] M. Tzelepi and A. Tefas. 2019. Graph-embedded Convolutional Neural Networks in human crowd detection for drone flight safety. *IEEE Transactions on Emerging Topics in Computational Intelligence* (2019).
[140] J. RR. Uijlings, K. EA. Van De Sande, T. Gevers, and A. WM. Smeulders. 2013. Selective search for object recognition. *International Journal of Computer Vision* 104, 2 (2013), 154–171.
[141] R. Vogel, M. Achab, S. Clémençon, and C. Tillier. 2020. Weighted empirical risk minimization: Sample selection bias correction based on importance sampling. *arXiv preprint arXiv:2002.05145* (2020).

[142] X. Wang, T. X. Han, and S. Yan. 2009. A HOG-LBP human detector with partial occlusion handling. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

[143] Z. Wang, M. Yu, Y. Wei, R. Feris, J. Xiong, W.-M. Hwu, T. S. Huang, and H. Shi. 2020. Differential treatment for stuff and things: A simple unsupervised domain adaptation method for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[144] Y. Watanabe, P. Fabiani, and GL. Besnerais. 2010. Towards a UAV visual air-to-ground target tracking in an urban environment. In *Proceedings of the International Congress of the Aeronautical Sciences (ICAS)*.

[145] Y. Watanabe, C. Lesire, A. Piquereau, P. Fabiani, M. Sanfourche, and G. Le Besnerais. 2010. The onera ressac unmanned autonomous helicopter: Visual air-to-ground target tracking in an urban environment. In *American Helicopter Society 66th Annual Forum (AHS 2010)*.

[146] R. Weibel and M. Heller. 1993. *Digital terrain modelling*. Oxford University Press.

[147] D. F. Wolf and G. S. Sukhatme. 2008. Semantic mapping using mobile robots. *IEEE Transactions on Robotics* 24, 2 (2008), 245–258.

[148] W. Xian, M. Qinwei, M. Shaopeng, and W. Hongtao. 2011. A marker locating method based on gray centroid algorithm and its application to displacement and strain measurement. In *Proceedings of the IEEE International Conference on Intelligent Computation Technology and Automation, (ICICTA)*.

[149] P. Xu. [n.d.]. *Information fusion for scene understanding*. Ph.D. Dissertation. Université de Technologie de Compiègne.

[150] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin. 2007. Graph embedding and extensions: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29, 1 (2007), 40–51.

[151] J. Yang, W. An, S. Wang, X. Zhu, C. Yan, and J. Huang. 2020. Label-driven reconstruction for domain adaptation in semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer.

[152] J. Yang, H. Zou, Y. Zhou, and L. Xie. 2021. Robust adversarial discriminative domain adaptation for real-world cross-domain visual recognition. *Neurocomputing* (2021).

[153] L. Yang, J. Qi, J. Xiao, and X. Yong. 2014. A literature review of UAV 3D path planning. In *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*.

[154] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang. 2018. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[155] F. Yu and V. Koltun. 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122* (2015).

[156] C. Yuan, F. Recktenwald, and H. A. Mallot. 2009. Visual steering of UAV in unknown environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[157] Y. Yuan, X. Chen, and J. Wang. 2019. Object-Contextual Representations for Semantic Segmentation. *arXiv preprint arXiv:1909.11065* (2019).

[158] H. Zender, O M. Mozos, P. Jensfelt, G-JM Kruijff, and W. Burgard. 2008. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems* 56, 6 (2008), 493–502.

[159] L. Zhang. 2005. *Automatic Digital Surface Model (DSM) Generation from Linear Array Images*. Institute of Geodesy and Photogrammetry.

[160] L. Zhang, L. Lin, X. Liang, and K. He. 2016. Is faster R-CNN doing well for pedestrian detection?. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer.

[161] S. Zhang, R. Benenson, M. Omran, J. Hosang, and B. Schiele. 2016. How far are we from solving pedestrian detection?. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[162] Y. Zhang and B. D. Davison. 2020. Domain adaptation for object recognition using subspace sampling demons. *Multimedia Tools and Applications* (2020), 1–20.

[163] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. 2018. ICNet for real-time semantic segmentation on high-resolution images. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[164] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. 2017. Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[165] S. Zhao, X. Yue, S. Zhang, B. Li, H. Zhao, B. Wu, R. Krishna, J. E. Gonzalez, A. L. Sangiovanni-Vincentelli, S. A. Seshia, et al. 2020. A Review of Single-Source Deep Unsupervised Visual Domain Adaptation. *IEEE Transactions on Neural Networks and Learning Systems* (2020).

[166] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. 2017. Scene Parsing through ADE20K Dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.