IMAV2018-31
http://www.imavs.org/pdf/imav.2018.31

Real-Time Landing Zone Detection for UAVs using Single Aerial Images

L. Oyuki Rojas-Perez*[1], Roberto Munguia-Silva[1], and Jose Martinez-Carranza[1,2]

[1]Instituto Nacional de Astrofisica, Optica y Electronica , Luis Enrique Erro 1, Sta. Mara Tonantzintla, CP 72840, Puebla
[2]University of Bristol, Bristol, UK, BS8 1UB

Abstract

In this paper, we propose a methodology for automatic detection of a landing and no landing zone for unmanned aerial vehicles. This framework consists of two processes carried out in a frame-to-frame basis: (1) an aerial depth estimated with a multiple layer Convolutional Neural Network architecture; (2) zone classification carried out with another Neural Network, based on the Inception Convolutional Neural Network, that learns from the aerial depth estimation. The novel aspects in this work are related to the training of these networks. Depth examples associated to aerial images are difficult to obtain since no public datasets of this sort are available. Likewise, diverse examples of landing and no landing zones based on aerial depth estimation are difficult to generate due to flight restrictions in urban areas. Motivated by this, we exploit public datasets meant for autonomous cars and synthetic data generated with simulation. We carried out evaluations using different synthetic datasets to those of training, real images obtained from the Internet, and flights in real scenarios with promising results.

1 Introduction

Nowadays, deployment of drones in outdoor missions is a common practice. There is an increasing number of applications where drones are piloted or sent to fly autonomously over rural, urban or natural environments, flying several kilometres away from the take-off position or even beyond the line-of-sight. In this context, it might happen that the mission has to be aborted or simply that the drone has to be retrieved due to low battery, which in many modern drones triggers the autonomous function *return home*. However, low battery is exactly a major cause of drone accidents since the battery may run out way before the drone returns. If the drone is kilometres far away, rather than bringing it home, the pilot may decide to take over and try to land it in a safe spot, using the transmitted video to identify an adequate landing zone. Nevertheless, interferance or delay in video transmission could affect such task.

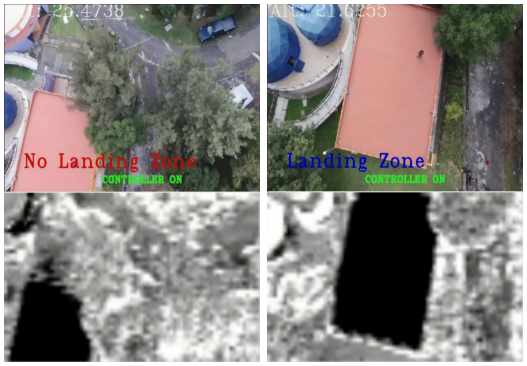


Figure 1: Our proposed system uses two processes, a depth image estimator and a zone classification of single aerial images, aiming at automatically detecting zones that are safe for landing, see video example in: `https://youtu.be/E8TZfErP8F8`

The above calls for a method that enables the drone to autonomously decide whether an observed area is a landing zone or not. Thus, we propose a two-step methodology for landing zone detection where only an onboard monocular camera is used to carry out the detection. The process involves two steps carried out in a frame-to-frame basis. First, a single aerial image captured by the onboard camera s processed by a Convolutional Neural Network (CNN), whose output will be that of a depth image estimation. A second step will grab the estimated depth image and pass it through another CNN network, referred by us as LandNet, that will output one of two classes: *Landing* and *No-Landing*, see Figure 1.

To achieve the above, the first CNN architecture for depth estimation has been trained using the well-known KITTI dataset, where RGB images and corresponding depth data are available. However, the chosen KITTI dataset was recorded with a camera pointing forward and mounted on a car ridding around a city. The scene is far from similar to what is observed in aerial images captured with a camera on board a drone with camera angle pointing downwards. Yet, we argue that the KITTI images and depth data could be exploited

*Department of Computer Science at INAOE. Email addresses: {oyukirojas, carranza}@inaoep.mx

to learn depth associated to texture patterns, in particular for depths in between the 20 and 60 metres, and can be *transferred* to the textures observed from the aerial cameras, thus useful to estimate depth. We followed this approach since there are no datasets available for aerial images with corresponding depth images, something difficult to generate with current stereo or depth cameras that work only for short depth ranges and under artificial light.

The second part of our approach exlploits the idea that depth image estimates belong to a metric space in the 3D world, not in the texture space of RGB images. Hence, we decided to test our depth image estimator on a simulated urban environment where we could obtain depth images. We observed that our depth estimator exhibits an error of up to 10 meters for aerial images captured at 40 or more meters of height. However, the depth estimates appear coherent regarding planar surfaces, which are exactly what we desire to detect as landing zones. Thus, by using simulation, we generate loads of annotated depth images with landing and no landing labels, and train a second CNN aim at generating a model that learns to separate the 3D shape of a landing zone from a no landing zone.

To present our approach, this work has been organized as follows: section 2 presents the related work; section 3 describes our methodology; section 4 presents our results and section 5 discusses our conclusions and future work.

2 Related Work

Autonomous landing problem for UAVs has been widely studied using different approaches and over diverse scenarios. We can find two main scenarios for autonomous landing, those where the selected site is static [1, 2, 3, 4, 5, 6, 7, 8], and others when is in moving (e.g., ships or other kinds of vehicles) [9, 10, 11, 12, 13, 14, 15]. Nevertheless, one element in common in both scenarios is the use of the markers that are placed on the selected landing site.

Among the approaches applied to the problem of autonomous landing, the use of sensors such as Light Detection and Ranging or Laser Imaging Detection and Ranging (LiDAR) [1, 2, 3, 4] provides of point clouds that can be processed aiming at detecing adecuate landing surfaces. For instance, in [3], the authors proposed a deep learning approach, where LiDAR point clouds are classified into a volumetric occupancy by using a 3D CNN architecture. In contrast, other works are opted to use infrared vision information [7, 8, 9, 10]. The works in [9, 10] describe a cooperative system where the objects at the ground form a T-shaped, being their detection used to estimate UAV's pose and performing an autonomous landing. In [7], a system based on an infrared stereo vision is proposed. The system is fixed on the ground and is used to track the UAV's position during the landing process. In [8], an infrared camera array guidance system is used to drive the landing. Although these systems are shown to work, they are tailored for specific zones, needless to say that those using infrarred cameras are prone to misbehave in zones with high temperatures. .

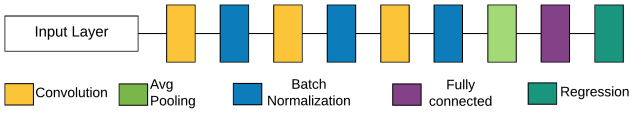


Figure 2: CNN architecture used for aerial depth estimation.

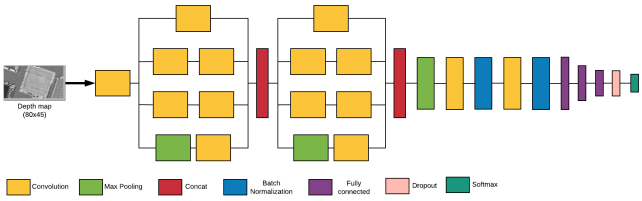


Figure 3: Architecture of the zone classifier based on Inception modules for the feature extraction section of the network.

Other systems for autonomous landing rely only on using RGB imagery transmitted by on board cameras [5, 6, 11, 12, 13, 14, 15, 16]. In [5], GPS in conjunction with vision are used to locate a landing target and land on it, being the vision used to leverage the target detection and recognition.

Finally, another set of works propose to use landmark detection to recognize landing areas. For instance, in [16], the authors propose a CNN architecture based on two efficient nets: YOLO [17] and Squeezenet [18]. The main idea is that of detecting different landmarks, in addition to maintain the efficiency of the detection. In contrast, in this work, we propose to carry out the detection without depending on any visual texture or marks.

3 Methodology

The task of detecting landing zones were divided into two secondary tasks, these tasks are the estimation of aerial depth maps and the classification of possible landing zones. Our approach on solving these tasks is based on machine learning techniques. For the task of aerial depth estimation we designed and trained a CNN following a methodology based on extraction of patches from RGB images. For the problem of zone classification from the depth maps, we propose a classifier based on the Inception modules [19].

3.1 Aerial depth estimation approach

The estimation of depth from aerial images is addressed using a methodology that exploits visual information by processing tiny patches from publicly available datasets imagery. For our purpose we processed images from the KITTI dataset to extract these patches from both RGB images and depth maps. The idea behind this approach is to take advantage of the farthest depth data available in the dataset imagery,

(a) City to generate aerial synthetic images for training.

(b) City to generate synthetic aerial images for evaluation of our approach.

Figure 4: Simulated urban scenarios created in Gazebo 7.

the greatest depth is located at the central part of the image, which is corresponds to the horizon view in the KITTI dataset. The patches used for processing are extracted from the central part of the image and are extracted after dividing the RGB image into equal-size patches. Once the patches are extracted, each patch is mapped to the average depth of the pixel neighbourhood in the depth map, the patches without depth information were excluded. This methodology lead to explore the design of a CNN architecture, which is shown in Figure 2.

This CNN architecture accepts the 25×25 patches as input and is followed by 3 convolutions with a batch normalization, in the final part of the architecture a max pooling operation is performed and a regression layer is used for the estimation of the depth of the corresponding patch. We designed this architecture given that most of the current CNN architectures work with images of a higher resolution to that of our extracted patches.

3.2 *LandNet*

The aim of the LandNet network is to detect possible landing zones from a given depth map, in our case this depth map is obtained from the mentioned depth estimator. We choose this approach due to the fact that while our depth estimator is good at recovering the shape of scene view it lacks of precision on the estimated depth and it would affect approaches based on geometrical detection of landing zones.

The proposed network accepts as input a depth map with size of 80×45 pixels and is followed by two Inception modules, corresponding to the feature extraction part of the network, see Figure 3. We use these Inception modules to extract different size of features from the depth map and to give the network the capability of learning more details from the depth map. After these two modules, we use a max pooling layer of size $(2, 2)$ followed by two convolutions and batch normalization. In the last part of the network we added three fully connected layers and a final softmax activation function. All the convolutions in our network, including the ones composing the Inception modules, use rectified linear activation. The output of the network is a probability of which of the two classes the actual view corresponds to.

3.3 *Datasets*

Our datasets consists on images collected from a simulated environment using the Gazebo simulator. We designed two different cities for this purpose, each city is composed by buildings, roads and trees as shown in Figure 4. We used an AR Drone MAV in the simulation for collecting the datasets. The images were collected from the bottom camera of the MAV. These images were feed to our aerial depth estimator to obtain the corresponding depth map for each of the RGB images on our dataset. In order to consider a depth map containing a possible landing zone the estimated map should contain a plain surface, to achieve this we monitored the RGB image and the estimated depth map and when the MAV reached a plain surface the estimated depth was stored into a file, Figure 5(a). The depth maps corresponding to views of trees, mountains or non-plain roofs were considered as no-landing zones, Figure 5(b). A total of 2000 images were obtained from the simulations environment for training and 1000 images for evaluation, with an equal number of samples for each of the two classes.

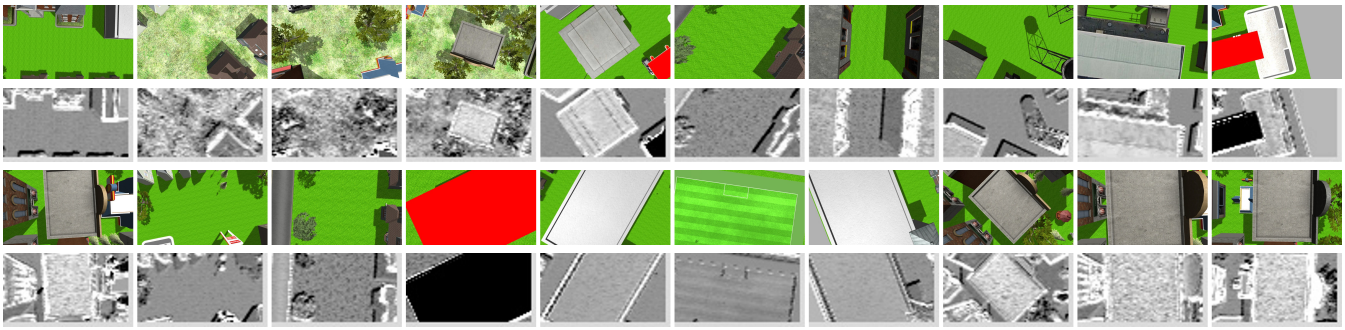
4 Experiments and Results

In this section, we describe the evaluation results, the control architecture employed, the vehicle hardware implemented and the experiments realised in simulation and outdoor environments.

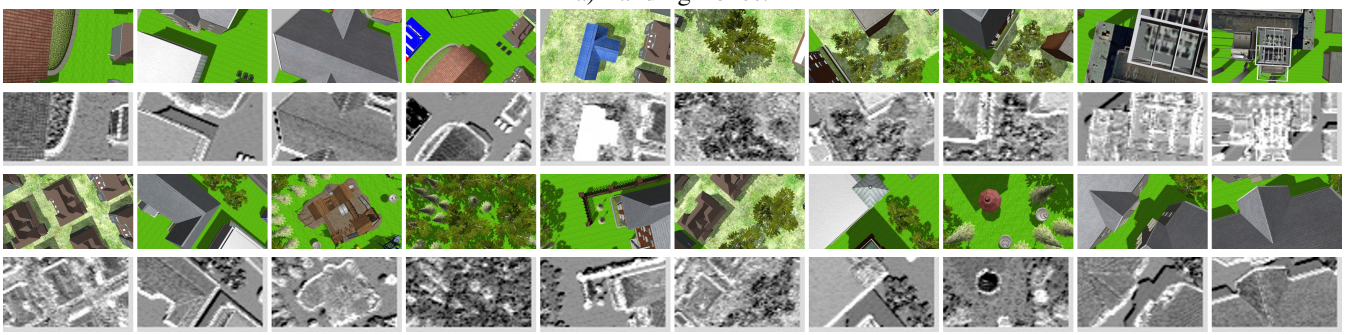
4.1 *Evaluation*

To evaluate the performance of our landing zone detection, we used two new datasets. Each dataset includes 1000 aerial images, 500 images of landing zones and 500 no landing zones. The first dataset was collected from a second simulated city on Gazebo, see Figure 6(a), and the second dataset was collected from the internet images, Figure 6(b).

We compared the evaluation results of our proposed classifier to the classifier trained with RGB images. For the first dataset, our classifier achieved 82% of accuracy to detect a secure landing zone and 75% to detect no landing zone, Figure 7(a), and the RGB classifier achieved 67% and 72% of accuracy to detect a secure landing zone and no landing zone,

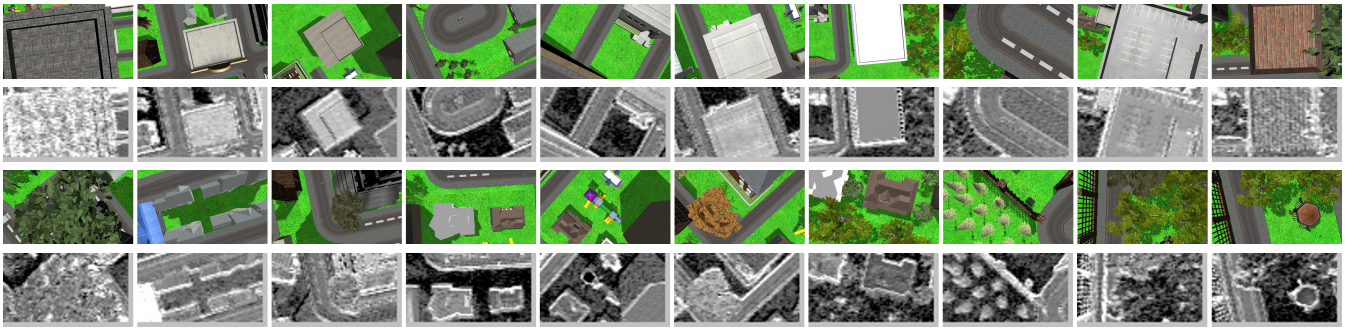


a) Landing Zones.

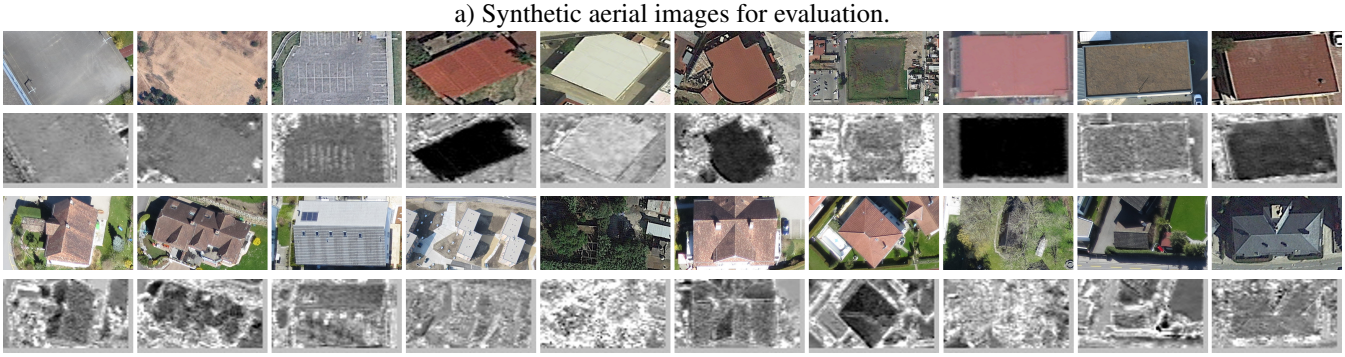


b) No landing zones.

Figure 5: Examples of synthetic aerial images, including their estimated depth image with our approach. Note that planar surfaces, ideal for landing, exhibit similar gray tone. For depth images, the farther the depth, w.r.t. the camera, the darker the gray value.

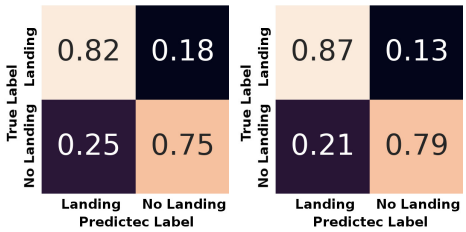


a) Synthetic aerial images for evaluation.



b) Aerial images downloaded from the Internet for evaluation.

Figure 6: Examples of aerial images, and their corresponding estimated depth, used for evaluation: the first set corresponds to synthetic images generated with the second simulated city, shown in Figure 4.b; the second set corresponds to aerial images downloaded from the Internet.



(a) Training: depth; evaluated: synthetic images.

(b) Training: depth; evaluated: Internet images

(c) Training: RGB; evaluated: synthetic images..

(d) Training: RGB; evaluated: Internet images.

Figure 7: Confusion matrix of our CNNs trained with depth images and compared against when trained with RGB images. Better results are obtained with depth.

respectively, Figure 7(c). According to the confusion matrices, our classifier trained with depth maps is more effective to detect secure landing zones compared to the classifier trained with RGB images, this is attributed to the capability of the network for learning depths of planar regions. For the second dataset, our classifier increments by 5% on the accuracy to detect a secure landing zone and by 4% to detect no landing zone, Figure 7(b) and the classifier trained with RGB images decreases its accuracy 8% to detect a no landing zone, Figure 7(d). This confusion matrix proves that our classifier based on depth maps is more stable to identify secure zones to land compared to RGB classifier.

4.2 System Architecture

The proposed landing detection system was implemented in ROS Kinetic Kame and consists of three nodes. One enables the communication with the drone for image transmission from the the drone to the computer and enables the control of the drones from the computer. The second node acquires the RGB image from the drone to obtain the depth image using the described method in section 3.1. Then we used the classifier described in section 3.2, to define if the current zone is a secure zone for landing and publish a flag. In the last node, we implement a controller. This node receives the flag and uses for two options: 1) to continue the navigation until finding a secure zone to land; 2) to keep the drone on hovering and then send a signal to land. To guarantee a secure landing, we use the average of 10 classified frames to decide whether a landing zone is detected or not.

4.3 Experiments in Simulation

We first present experiments in a simulated environment, and we use tum_simulator package and Gazebo 7. The simulated drone navigates upon the city shown in the Figure 4(b). We flew the drone in manual mode at 15 metres in height. The path flight consisted of cover areas to detect possible landing zones when the classifier detects a landing zone the drone stay on hovering and then lands. The Figure 8, shows the simulation test, each figure show the simulated environment, the image from the drone and the estimated depth.

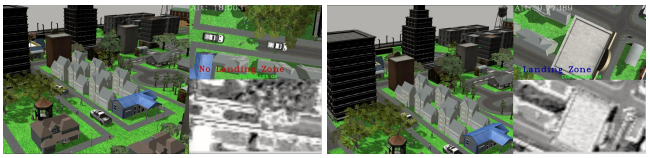


Figure 8: Two illustrative examples of our approach tested in a simulated urban environment.

4.4 Experiments in Outdoor Environment

For these experiments, the tests consisted of flying the drone in a real outdoor environments at 25 meters in height. The drone flew over trees, buildings with flat and spherical roofs. In Figure 9, the images show examples of the detection, whether it is a landing or a no landing zone. When flying over the roof of a building, our system correctly identifies it as a landing zone. Our system exhibit a processing time of $5Hz$, which includes depth estimation and landing/no landing classification.

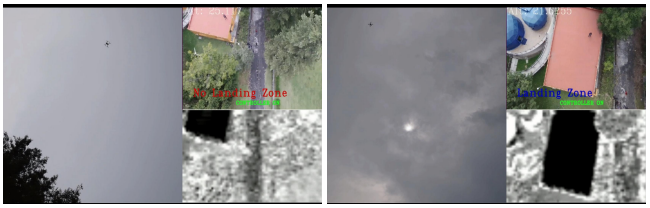


Figure 9: Two illustrative examples of our approach tested in real outdoor, see `https://youtu.be/E8TZfErP8F8`

5 Conclusion

We have presented a method for automatic detection of landing zones for UAVs using aerial images captured with an onboard RGB camera. To address this problem we have used two main ideas: (1) to estimate a depth image from the aerial image; (2) to use the estimated depth images, annotated accordingly, to train a computational model that will learn landing zones and no landing zones. For both tasks, we have implemented two CNNs, where we have used a public dataset from another domain (autonomous vehicles) to obtain training examples for aerial depth estimation, and also us-

ing synthetic data generated with a simulator of urban scenes, useful to generate loads of aerial depth images.

We carried out tests with different synthetic data, images from the Internet and during flight missions in real challenging scenarios, obtaining around 80% of accuracy for synthetic and real images. We have also studied the case of when our approach is trained with RGB images instead of depth images, for the landing/no landing detection, obtaining and average of 65% in accuracy, proving the benefit of using depth data rather than texture. In average, our whole system works at $5Hz$, however, we are confident that we can speed up this process by refining our CNN architectures.

As future work, we will improve upon the aerial depth estimation as much as to increase the processing speed.

REFERENCES

[1] M. Whalley, M. Takahashi, P. Tsenkov, and G. Schulein. Field-testing of a helicopter uav obstacle field navigation and landing system. In *Proceedings of the 65th Annual Forum of the AHS, Grapevine, TX.*, 2009.

[2] S. Scherer, L. Chamberlain, and S. Singh. Autonomous landing at unprepared sites by a full-scale helicopter. *Robotics and Autonomous Systems*, 60(12):1545 – 1562, 2012.

[3] D. Maturana and S. Scherer. 3d convolutional neural networks for landing zone detection from lidar. In *2015 ICRA*, pages 3471–3478, May 2015.

[4] Oscar G. Lorenzo, Jorge Martínez, David L. Vilariño, Tomás F. Pena, José C. Cabaleiro, and Francisco F. Rivera. Landing sites detection using lidar data on manycore systems. *The Journal of Supercomputing*, 73(1):557–575, Jan 2017.

[5] S. Saripalli, J. F. Montgomery, and G. S. Sukhatme. Vision-based autonomous landing of an unmanned aerial vehicle. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 3, pages 2799–2804, 2002.

[6] Xiaoming Li. A software scheme for uav's safe landing area discovery. *AASRI Procedia*, 4:230 – 235, 2013. 2013 AASRI Conference on Intelligent Systems and Control.

[7] W. Kong, D. Zhang, X. Wang, Z. Xian, and J. Zhang. Autonomous landing of an uav with a ground-based actuated infrared stereo vision system. In *2013 IEEE/RSJ IROS*, pages 2963–2970, Nov 2013.

[8] Tao Yang, Guangpo Li, Jing Li, Yanning Zhang, Xiaoqiang Zhang, Zhuoyue Zhang, and Zhi Li. A ground-based near infrared camera array system for uav auto-landing in gps-denied environment. *Sensors*, 16(9), 2016.

[9] G. Xu, Y. Zhang, S. Ji, Y. Cheng, and Y. Tian. Research on computer vision-based for uav autonomous landing on a ship. *Pattern Recognition Letters*, 30(6):600 – 605, 2009.

[10] G. Xu, X. Qi, Q. Zeng, Y. Tian, R. Guo, and B. Wang. Use of land's cooperative object to estimate uav's pose for autonomous landing. *Chinese Journal of Aeronautics*, 26(6):1498 – 1505, 2013.

[11] Stephen M. Chaves, Ryan W. Wolcott, and Ryan M. Eustice. NEEC research: Toward GPS-denied landing of unmanned aerial vehicles on ships at sea. *Naval Engineers Journal*, 127(1):23–35, 2015.

[12] H. Lee, S. Jung, and D. H. Shim. Vision-based uav landing on the moving vehicle. In *2016 ICUAS*, pages 1–7, June 2016.

[13] Oualid Araar, Nabil Aouf, and Ivan Vitanov. Vision based autonomous landing of multirotor uav on moving platform. *Journal of Intelligent & Robotic Systems*, 85(2):369–384, Feb 2017.

[14] Alexandre Borowczyk, Duc-Tien Nguyen, AndrPhu-Van Nguyen, Dang Quang Nguyen, David Saussi, and Jerome Le Ny. Autonomous landing of a quadcopter on a high-speed ground vehicle. *Journal of Guidance, Control, and Dynamics*, 40(9):2378–2385, 2017.

[15] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, Z. Milosevic, A. Garcia-Vaquero, and P. Campoy. Towards fully autonomous landing on moving platforms for rotary unmanned aerial vehicles. In *2017 ICUAS*, pages 170–178, June 2017.

[16] Leijian Yu, Cai Luo, Xingrui Yu, Xiangyuan Jiang, Erfu Yang, Chunbo Luo, and Peng Ren. Deep learning for vision-based micro aerial vehicle autonomous landing. *International Journal of Micro Air Vehicles*, 10(2):171–185, 2018.

[17] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *The IEEE CVPR*, June 2016.

[18] Forrest N. Iandola, Matthew W. Moskewicz, Khalid Ashraf, Song Han, William J. Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size. *CoRR*, 2016.

[19] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE CVPR*, pages 1–9, June 2015.