Autonomous inspections of power towers with an UAV

Maitre, Guillaume; Tuci, Elio; Martinot, Dimitri; Fontaine, Dimitri

Published in:
Proc. of the XV International Workshop on Artificial Life and Evolutionary Computation (WIVACE)

Publication date:
2021

Autonomous inspections of power towers with an UAV*

Guillaume Maître[1[0000−0003−4462−8492]], Dimitri Martinot[2[]], Dimitri Fontaine[2[]], and Elio Tuci[1[0000−0001−7345−671X]]

[1] University of Namur, Namur, Belgium
{guillaume.maitre, elio.tuci}@unamur.be
https://www.unamur.be/info/
[2] Qualitics SPRL, Ans, Belgium
{dimitri.martinot, dimitri.fontaine}@qualitics.eu

Abstract. Electrical grid maintenance is a complex and time consuming task. In this study, we test a way to perform electrical grid tower inspection by using camera images taken from an autonomous UAV. The images are segmented with a type of convolutional neural network called U-Net [9]. The results of the segmentation process is used to generate the movements of the UAV around the tower. The training of an U-Net model requires a large amount of labelled images. In order to reduce the time and financial costs of the generation of a large data-set of labelled images of physical towers, we develop a physics-based simulation environment that models the UAV dynamics and graphically reproduces electric towers in multiples environmental conditions. We extract labelled images for U-Net models training from the simulator. We perform multiple training, test conditions with different amount of natural world and simulated images and we evaluate which training condition generates the most effective U-Net model for the natural world image segmentation task. The contribution of the study is to details the characteristics of the training condition that allows to maximise the U-Net performances with the minimum amount of physical world images in the training set. With the best performing U-Net, we create post-processing analysis on the result of the segmentation to extract the required pieces of information to properly move the UAV.

Keywords: Computer Vision · Convolutional Neural Network · Autonomous System

1 Introduction

UAVs are particularly suitable to automate the exploration and monitoring of remote areas such as glaciers, and volcanos, but also the inspection of infrastructures which would otherwise requires complex and/or costly operations if carried

* G.M. thanks the Walloon Region for the financial support "Doctorat-en-Entreprise" fellowship.

out without UAVs [2, 6]. In this latter case, the inspection process is generally performed with cameras or other sensors mounted directly on the UAV, which flies around the target in order to optimise the data collection process relative to its status [8]. The objective of our work is to develop new methodologies to improve the efficiency as well as the autonomy of UAVs during high-precision inspection tasks. In particular, we describe a set of experiments aimed to automate the inspection of electricity grid pylons by using autonomous and visually-guided UAVs. The vehicle gets close to and flies all around a target pylon while collecting images which are, in the first instance, used by the UAVs itself to navigate and to make an exhaustive exploration of the target while avoiding collisions with any object. The images are also collected and processed offline to create a SVG images of the precise shape pylon, and to detect damaged areas or any element requiring further attention.

Currently, electric pylon inspection is a critical and costly task due to the time and effort required. If we exclude the use of UAVs, the inspection process can be performed in one of the following ways:

- by helicopter: this is an extremely expensive method. It requires expensive specific material such as a camera with high distance zoom and different experts to pilot, to take photos and to analyse the collected data. This method does not provide a bottom-up point of view of the electric pylon since the helicopter can not fly underneath the cables.
- unaided-eye visual inspection from ground: this method only provides a mild inspection and only from a bottom-up perspective. Nevertheless, it is the cheapest and also the fastest method.
- by escalating the tower: this is the more informative method, which gives a clear insight of the state of the electric pylon. Unfortunately, it requires cutting completely the electric line. The fact that it requires to cut a line makes this method the rarest one. This method is also the most dangerous since some electric pylon might be too rusted and metal bar could collapse under the weight of the climber.

The automation of the inspection process of electric pylon by UAVs presents several clear advantages over the above mentioned alternative methods. With autonomous UAVs, the human factor is completely removed from the inspection process by significantly improving the safety and comfort of the personnel, without having to cut the electric line. Plus, images or any other type of data relative to the pylons can be taken from perspective generally inaccessible to non-UAV-based inspection systems. Also, the automation enables a consistency and precision that a human remote pilot can hardly achieve, thus improving the quality of the audit and the accuracy of the diagnosis.

In this paper, we investigate the issue of how to train a convolution neural network (CNN) to identify the structure of pylons from images taken by a camera mounted on the UAV. The image segmentation process has to be informative enough to allow the UAV to autonomously move around the pylons without crashing into their structure or into the electric cables supported by the electric pylon. Our CNN-based controller is an U-Net model [9], a CNN model that

can perform real time segmentation without requesting much data for training. The CNN guides the UAV in the task of detecting the electric pylon and in exhaustively exploring all its parts.

U-Net is made of two parts: a contraction path to extract a maximum of features and contexts out of an image; and a symmetric expanding path to extract a pixel wise precision mask of the object in the image. A mask is an image in which the intensity of pixels corresponding to a target object are set to non-zero values and all the other pixels are set to zero. The main advantage of U-Net is the capacity to work with few available images in a training data-set. A series of comparative tests detailed in [9] shows that U-net has been able to outperform other types of convolution neural networks in the 2015 IEEE-ISBI cell tracking challenge. The model proved to be able to perform precise cell detection in less than a second for 512*512 size images on 2015 GPU (NVIDIA GTX Titan 6Gb) [9].

Compared to other object detection models, such as Fast R-CNN [5], that creates a bounding box around the object to be detected, U-Net proposes a pixel wise detection which precisely identify the object within an image. The disadvantage of U-Net is that it does not offer the possibility for multiple instances segmentation. This means that it is not possible to extract different bodies of the same types in an image or same objects that are next to each other. For this kind of task it is suggested to use other convolution neural network models like Fast R-CNN. However, given that electric pylons are usually represented as a single instance object in images, U-Net is a suitable model for their identification through detection in camera images. Thanks to the segmentation process and few mask analysis, it is possible to retrieve important information not only concerning the conditions of the electric pylon but also its is possible to easily determine the position of the camera (and consequently of the drone on which the camera is mounted) with respect to the electric pylon.

In this paper, we look at a way to use U-Net to perform a precise enough mask of a electric pylon in an image. The complexity of this task is determined by the shape of the electric pylon which, due to the void space between the electric pylon's structures, makes it hard to be properly detected. Moreover, the creation of a large data-set of natural world images of electric pylon, generally required for the training of convolution neural networks, is a costly and difficult task due to the relatively large variability in electric pylon's shape. With respect to this challenges, U-net offers several advantages compared to other models since it can successfully be trained to perform object segmentation tasks without having to rely on large data-sets. Moreover, U-Net can perform the image segmentation tasks quite fast without requiring particularly powerful computational resources.

The contribution of this study is to show that U-Net can be effectively trained for this inspection task with hybrid set of images in which the large majority of them are generated with a simulation environment that models the main features of the target scenarios. To generate the simulated images, we make use of AirSim [11], the simulation developed by Microsoft powered by Unreal Engine. In particular, we show that the training performed with sets of simulated images is

as effective in the segmentation task as the training performed with sets of images from the natural world. In the following sections, we first describe the simulation environment used to generate the simulated images, and the experimental design used to test the hypothesis that hybrid sets of simulation-generated and physical world images are as effective for the training as sets of only physical world images (see section 2). In section 3 we illustrate and discuss the results of our analysis. In section 4, we draw our conclusions.

2 Methods

In this section, we illustrate the methodological aspects of our study. First of all, we introduce the simulator used to create the simulated scenarios. Then, we describe the characteristics of the UAV employed. After that, we explain the architecture of the U-Net model and the analysis performed on the camera images to generate the UAV movement.

2.1 The simulator and the UAVs model

(a) (b)

Fig. 1. Images from the simulator rendering: (a) an urban scenario; (b) a rural scenario.

The simulation environment is created using the software AirSim [11], made of Unreal Engine physics engine and high quality graphics for rendering, with which we model the dynamics of the flying vehicle, the pylon of the electricity grid and details of the surrounding area. For this study, we have modelled a single type of pylon, located in multiple types of background, one urban, one forest, two rural, and one mountain scenario (see Figure1). In each scenario, there are 3 electric pylon placed at 400 m from each other. In one of the rural scenario there is also a windmill with moving blades which represent dynamic elements into the scene. In every type of scenario, the horizon is represented by distant mountains which are drawn in a very ragged way to avoid the network to exploit the horizontal skyline for its movements. Each scenario is also replicated

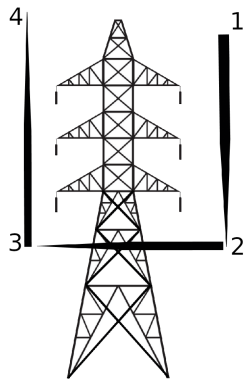
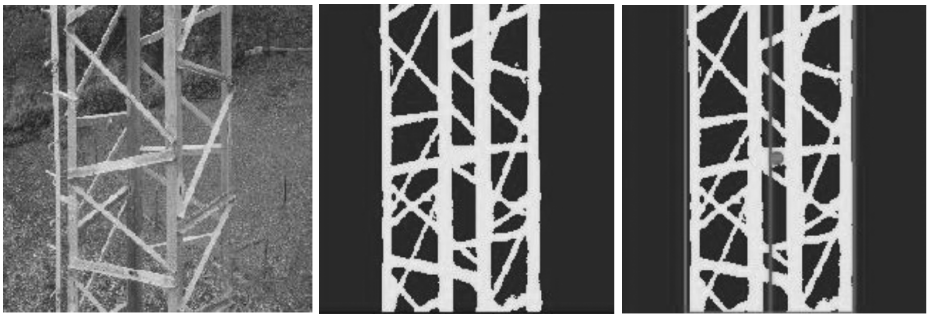


Fig. 2. Schematic view of the electricity grid pylon with marks to represent the UAvs movements during exploration.

in different weather and luminosity conditions (e.g., with sun, rain, snow, fog, etc.). The variability in the simulated scenarios in which the electric pylons are placed is needed to make sure that the convolution neural network can identify the structure of the tower regardless of the characteristics of the background and meteorological conditions.

For what concerns the structure of the UAV, AirSim has already an built-ub UAV model (quadcopter) and it is offering the possibilities to add our own models. The built-in model has the possibility to be customised in terms of sensorial capabilities. In this study we use the built-in quadcopter with a front facing camera, that generate images with a resolution of 5280*3956 pixels; the FOV is 72○; the focal distance is 200.0 and the focal region is 200; no specific filter is applied to the image. The convolution neural network segments in real time the images generated by the camera in order to identify the pixel-wise position of the electric pylon in the image. A hand-designed control system generates high level instructions from the results of the segmentation process. Both the segmentation process and the rules to generate the UAV movements are described below.

The PixHawk PX4 fight controller takes care of regulating the speed of the rotors in order to execute the high level motor commands by taking into account the physical features of the UAVs. Thanks to the PX4 flight controller, the behaviour of the simulated UAV can be easily replicated in physical UAVs. The UAV is also equipped with a ground facing radar to measure the distance from ground or any object placed underneath its body. The information generated by the radar is used to manoeuvre the UAVs safely (without collisions) while moving round the pylon. The UAV is also equipped with a seven points front-facing LiDAR to precisely measure the distance between the UAVs and any object in the space in front of it. The LiDAR makes possible to keep the UAVs always at more than 4 m away from any element of the electric pylon.



(a) (b) (c)

Fig. 3. (a) An image taken by the UAV's camera; (b) The image in (a) after segmentation by the U-net; (c) The image from (b) after post-processing with Canny-edge and HOUGH.

For what concerns the movements, the UAV moves by using PIDs for pitch and roll. A constant value is used for height adjustment.

The UAV moves around the electric pylon as illustrated in Figure 2. The UAV stabilise and adjust to be centred at the same height as the upper part of the pylon (see position 1 in Figure 2). While in position 1, the camera starts taking snapshots which are processed by the convolution neural network to detect the position of the pylon. The results of the image-segmentation process is used by the control system to move the UAVs downward while remaining at a constant safe distance from the pylon. When the down facing Radar get close enough to the ground or an obstacle and the front facing LiDAR detect no object in front of the UAV (see position 2 in Figure 2) the downward movement is stopped. At this point the UAVs generate a parabolic movement to reach the other side of the pylon moving underneath the electric cables (see position 3 in Figure 2). Once of the other side, the UAVs starts an ascending movement always using the results of the image-segmentation process to remain in the proximity of the pylon. The ascending movement terminate when the UAV reaches the top of the electric pylon (see position 4 in Figure 2). From position 4 the UAVs returns to position 1 following the same trajectory. This exploration pattern makes possible to have images of the two side of the electric pylon taken while the UAV is both in an ascending and descending movement.

As the UAV is following this path, the UAV takes an image (resized to 320X320 pixels) once every 130 milliseconds (see Figure 3a). Each images is segmented by the U-net to generate a pixel-wise segmented images in which pixels belonging to the pylon are in white and the background is black (see Figure 3b). On these images we apply the Canny Edge detection algorithm [3] to extract the border of the pylon. Then, with a Probabilistic Hough Line Transform [7], we extract the lines of the border of the electric pylon (see Figure 3c).

After this processing, we can extract the information necessary to move the UAV in order to have the pylon at the centre of the image. To compute the UAV

distance from the electric pylon with the image we compute the pixel width of the tower x and we compare x with the expected pixel-width of the electric pylon $dist$ to have the tower at 4 m from the UAV. $dist$ is computed in the following:

$$dist = \left(\left(\frac{y}{FOV \times z} * 100 \right) * 3,2 \right) \tag{1}$$

where y is the width of the cage of the pylon in centimetres; and $z = 4$ represent the security distance to respect between the UAV and the pylon. If x is bigger than $dist$, the UAv is too close to the tower. If instead x is smaller than $dist$, the UAV is more than 4 m away from the tower.

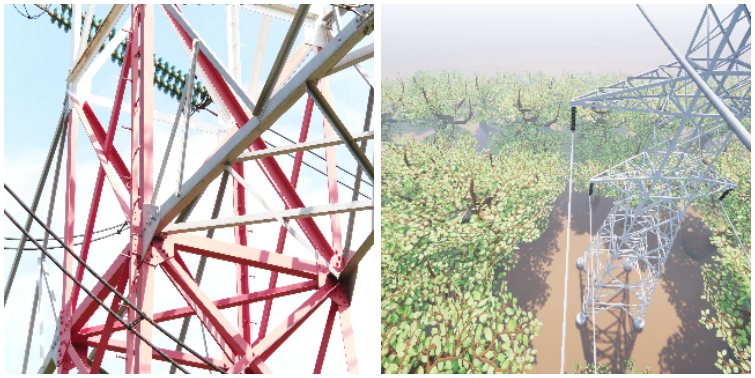
2.2 The U-net and our experimental design

In order to train the U-net to identify the tower in the images we have generated 1200 images, out of which 400 images from the natural world and 800 generated from the simulator. This data set benefit from data augmentation (e.g., flipping, rotation and removing one of the colour channel from the RGB spectrum, etc) on the original images. The natural world images have been manually labelled in order to identify pixels belonging to the tower and those belonging to the background. This labelling process is a long and relatively expensive process that is however necessary for the training. The images from the simulator are automatically labelled.

Our experimental design features four training conditions and three test conditions described in Table 1. The training conditions differ in the number of natural world and simulated images used for training. Note that, in the training condition A we do not use any natural world images. Condition D is the one with the highest number of natural world images. We assume that the highest the number of natural world images in the training set, the better the network performances in the segmentation task. The objective of this study is to verify whether and the extend to which natural world images can be replaced with simulated images without significant loss of performances. For this reason, the performances of the U-net trained in condition A, B, and C will be compared with the performances of the U-net trained in condition D. Note that, we do

data-set name	% simulated images	% physical images
Training A	100	0
Training B	80	20
Training C	90	10
Training D	50	50
Test I	50	50
Test II	100	0
Test III	0	100

Table 1. Table summarising the experimental design with the four training conditions and the three test conditions



(a) (b)

Fig. 4. (a) A natural world image used for trainig/test. (b) A simulated image.

not have a training condition made by all natural world images simply because the number of these images at our disposal at the time of this study was not sufficient to train the network.

The training sets were also divided with a k-fold into a 80-20 percent training and validation sets using a k-fold data loader during the training of the models. The structure of the U-net detailed in Figure 5. The U-net takes as input RGB images of 320X320 pixels and generate a vector of 320X320 real-valued numbers in $[0, 1]$. These values represent the probabilities that each pixels has to belong to the tower. The U-net is modelled using the programming framework tensorflow [1]. We have tested different parametrisation of the training process, by varying the number of epochs, the function optimiser, the nature of the loss function, the size of the data-set, the training stop-criteria (see Table 2 for details).

3 Results

From all the possibles values represented in Table 2, we tried every combination of parameters. Each training condition illustrated in Table 1 has been replicated for each possible combination of the parameters listed in Table 2. In this section, we only discuss the results generated from the best set of parameters, which is

parameter	tested values
epoch	25, 50, 75
Optimiser	AdaMax, Adam, Nadam
Loss Function	Mean Square Error, Absolute Square Error
Images in the training data set	400, 800, 1600
Early stopper	None, 2,3,4

Table 2. The training parameters

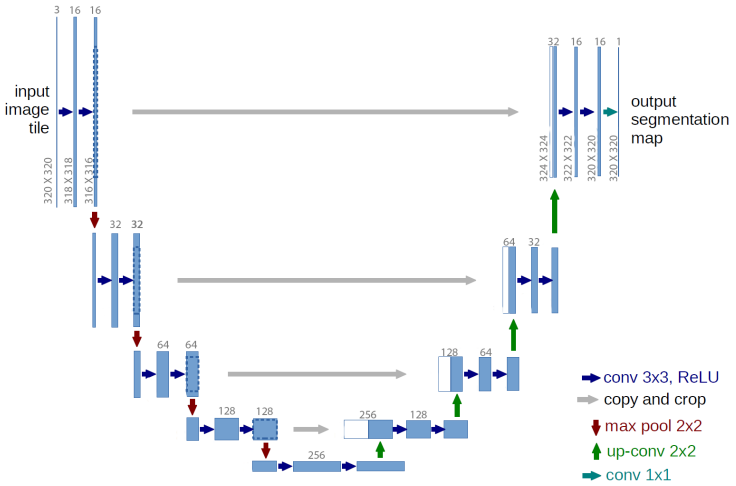


Fig. 5. The structure of the U-Net model used in the study

characterised by the following: 50 epochs with an early stopper of 2; a total of 800 images in the training set, Adam and the Mean Square Error as optimiser and loss function in F1-score. Recall that the objective of our study is to train the U-Net model to be able to extract the information required to move the UAV around an electric pylon by using primarily computer vision. To do so, the U-Net model has to be able to properly segment the images to precisely determine where the pylon is and to perform the right movements for the audit. To find out the best performing U-Net model, we experimentally vary the proportion of physical world and simulated images in the training set.

The performances of the U-Net model are evaluated with the F1-score. This score is computed in the following:

$$F1 = \frac{2}{recall^{-1} + precision^{-1}} \tag{2}$$

$$Precision = \frac{TrueDetectedPixels}{AllDetectedPixels} \tag{3}$$

$$Recall = \frac{TrueDetectedPixels}{AllRelevantPixels} \tag{4}$$

data-set name	Test I	Test II	Test III
Training A	0.485666	0.74248765	0.25558049
Training B	0.807414	0.86110346	0.81913397
Training C	0.767606	0.80255722	0.79024012
Training D	0.848198	0.82709646	0.89200856

Table 3. Table showing the F1-score for each combination of training and test condition.

Where $TrueDetectedPixels$ represent the detected pylon's pixels by the U-net model that are correct. $AllDetectedPixels$ represent the pylon's pixels generated by the U-Net model segmentation. $AllRelevantPixels$ all the real pylon's pixels in the image, detected or not by the U-Net model. F1-score is wildly used in computer vision for accuracy calculation. This type of accuracy measurement takes into account the false negative compared to standard accuracy measurement. This offer a more precise score compared to the classic accuracy calculation.

The results of our simulations are illustrated in Table 3, which shows the F1-score for each combination of training and test condition. The U-Net model generated with the training condition A turns out to be the worst performing model in all the possible test conditions. This relatively negative performance can be accounted for by the reduce variability in the scenarios represented in the training set. Such reduced variability generates what is referred to as an "over-fitting", since the model can not generalised its performances beyond the images presented in the training condition. The training condition B and C generated better performing U-Net models than condition A, with condition B being the best one in all possible test conditions. The lower F1-score of the U-Net from training condition C can also be accounted for by some "over-fitting", caused by the reduced number of physical world images compared to condition B. When comparing training conditions B and D, training condition D out perform B on test condition I of 0.04 and test condition III of 0.07. This shows that more natural world images enable the U-Net model to perform better on natural world images and both types together. It is important that the model can perform well while remaining the less expensive as possible in terms of cost and time to create. So, even if the training condition D is performing better, the training condition B is the most valuable for us. Training condition B offers the best cost to performance ratio since it only require 160 natural world images compared to 400 for the training condition D. As far as it concerns the UAV's movements, with the U-Net model trained with the training condition B we are able to perform all the movement required to perform an audit by following the trajectory illustrated in Figure 2. The sequence of movements requires between 4 minutes and 24 seconds and 6 minutes 21 seconds depending on the scenarios and the starting position. The UAV is able to keep its safe distance with the pylon without much difficulties. The difference in time is due to different starting positions which require the UAV to centre itself before starting and the search for obstacles that can activate the position 2 of Figure 2 earlier.

4 Conclusions

We achieved our main goal during this study which was to train a U-Net model to visually guide an autonomous UAV during an inspection task of an electric grid pylon. The main challenge was to find an economic and fast dataset of labelled images to train our U-Net model. To do so, the model was trained with different type of data-sets made of a different amount of natural world and simulated images mixed together.

The U-Net model generated with the Training condition B is performing relatively well in both natural world and simulated scenarios. The mixing of both natural world and simulated images helps the U-Net model to reduce the effect of the Reality Gap [10] for the world perception trough the camera. The training condition B and D are doing good enough to perform similarly in natural world and simulated world. We decide that the training condition B is better even with a lesser score than training condition D. The cost and the time required to create the training condition B being a factor to prefer such condition for training. It maintains a good score everywhere while only requiring 0.2% of natural world images.

In terms of F1-Score, a score superior to 0.8 is considered a good enough score for a convolution neural network required to generalise the segmentation capabilities beyond the training set. The pylons are a really complex shape with a lot of details and smaller parts. That complex shape makes it difficult to properly segment the pylon especially with the 320X320 pixels image. With a score superior to 0.8 the post-processing have the required information to generate all the requested data for the movement.

In our future works, we plan to test this system on a natural world electric pylon and see how well the UAV is able to perform when facing other elements not taken into account in our images (e.g., wind, sun glare, sensors noise, etc). Those natural world experiments are expected to show the more complex interactions that can change the behaviour of the UAV compared to the simulation. Moreover, since the time of running our experiments, we have received a lot more natural world images. With those additional images, we could test different neural networks that could not properly work with the few images we had during this study. Concerning the audits of the pylon, we will be looking at a way to perform instance object detection for flaws (e.g., rust, damage, missing component, bird nest, etc). Automated advanced audit would help to create a clear database of the state of the electric grid pylon for the owner. The final objective is to create a data-driven self-regulating systems where a UAV detects the flaws and multiple other UAVs would be able to react to such detection to perform the required repairs. Another future work would be to look at more complex models to extract depth in the image while still using one camera, (e.g, M4Depth [4]). This way, we could remove the LiDAR by combining both the U-Net model and the M4Depth model to extract proper distance for security and the position of the pylon. A better understanding of the surrounding can also helps the UAV to perform better obstacles avoidance in more complex scenarios.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng,

X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), https://www.tensorflow.org/, software available from tensorflow.org
2. Albani, D., Manoni, T., Arik, A., Nardi, D., Trianni, V.: Field coverage for weed mapping: Toward experiments with a uav swarm. In: Compagnoni, A., Casey, W., Cai, Y., Mishra, B. (eds.) Bio-inspired Information and Communication Technologies. pp. 132–146. Springer International Publishing, Cham (2019)
3. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679698 (Jun 1986). https://doi.org/10.1109/TPAMI.1986.4767851, https://doi.org/10.1109/TPAMI.1986.4767851
4. Fonder, M., Ernst, D., Droogenbroeck, M.V.: M4depth: A motion-based approach for monocular depth estimation on video sequences (May 2021)
5. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (December 2015)
6. Hallermann, N., Morgenthal, G.: Unmanned aerial vehicles (uav) for the assessment of existing structures. vol. 101 (09 2013). https://doi.org/10.2749/222137813808627172
7. Kiryati, N., Eldar, Y., Bruckstein, A.: A probabilistic hough transform. Pattern Recognition **24**(4), 303 – 316 (1991). https://doi.org/https://doi.org/10.1016/0031-3203(91)90073-E, http://www.sciencedirect.com/science/article/pii/003132039190073E
8. Mth, K., Buoniu, L.: Vision and control for uavs: A survey of general methods and of inexpensive platforms for infrastructure inspection. Sensors **15**(7), 14887–14916 (2015). https://doi.org/10.3390/s150714887, https://www.mdpi.com/1424-8220/15/7/14887
9. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation (2015)
10. Scheper, K.Y.W., Croon, G.D.: Abstraction, sensory-motor coordination, and the reality gap in evolutionary robotics. Artificial Life **23**, 124–141 (2017)
11. Shah, S., Dey, D., Lovett, C., Kapoor, A.: Airsim: High-fidelity visual and physical simulation for autonomous vehicles. CoRR **abs/1705.05065** (2017), http://arxiv.org/abs/1705.05065