Deep Learning with Semi-Synthetic Training Images for Detection of Non-Cooperative UAVs

Christoph Briese, Lukas Guenther
Institute of Flight Systems
German Aerospace Center (DLR),
Braunschweig, Germany
christoph.briese@dlr.de

Abstract— **This paper presents a method to generate a dataset for training a deep convolutional network to detect a non cooperative unmanned aerial vehicle in video data. Deep convolutional network have shown a great potential for tasks like object detection and have been continuously improved in the last years. Still, the amount of training data is large and their generation can be complex and time consuming, especially if the appearance of the detected object is not clearly specified. The concept presented here is to train a deep convolutional neural network just with a few two dimensional images of unmanned aerial vehicle to simplify the process of generating training data. Performance of the trained network is evaluated with data from real experimental flights and compared with hand-labeled ground truth data to validate the correctness. To cover situations when the classifier fails at the detection, the output is integrated in a image processing pipeline for object tracking in order to establish a continuous tracking.**

I. Introduction

Due to the increasing number of small unmanned aircraft (UAV) over the last years, especially low-cost ready-to-fly drones, the need of a system to detect and track small moving objects has increased. This significant increase of ready-to-fly UAVs threatens not only the safety of airspace but also the safety of critical infrastructure as well as privacy and security issues in these areas. Geo-fencing-based solutions and official regulations can help to cope with some of these problems, however there are situations which are not covered by these methods. A experienced pilot can fly independently from GPS signals by using a first person view (FPV) setup. As a consequence, active drone defense and monitoring of airspace becomes an important and interesting field of research and development.

There are already several methods established on the commercial market like the one presented in [1] to detect and localize UAV which are penetrating (or entering) restricted airspace. They rely on scanning for navigation and control radio signals. Currently, most of the presented solutions focus on detection of UAV, but not on active counter measures.

Ground-based active solutions like hand held devices are presented in [2] or [3] but have limits in range or reaction time. In that case, protection becomes difficult.

The authors would like to expresses their gratitude to the Unmanned Aircraft Team at DLR, namely Endre-Aladar Papp, Pascal Lipke, Lars Kellermeier and Stefan Krause for organizing and performing flight operations and assist on data analysis

Spoofing or jamming radio control signals can also be critical because of legal issues or the potential possibility of tying down the own infrastructure which needs to be protected. To overcome the limitations mentioned before, the goal of this work is to develop a fast and highly agile UAV (called: *defender*) which is equipped with environmental sensors to detect other UAVs (called: *intruders*) near to it. Due to the requirements in terms of cost, weight and detection range, cameras are the sensors of choice. Once an intruder is successfully detected and tracked, the defender can intercept the intruder and apply appropriate counter measures.

The further parts of this paper are organized as follows. Section II gives a short overview on common established computer vision and machine learning methods for object detection in aerial images. In section III the generation of labeled data and the training of a convolutional network is described which includes the implementation of the network and the integration into an image processing pipeline. Section IV gives an overview about the flight experiments performed with two UAVs in order to generate realistic test data. In section V the performance of the trained network is tested with the recorded data from the flight experiments and finally section VI provide a conclusion and outlook.

II. Background and related Work

The scenario considered in this work is based on an autonomous UAV equipped with an on-board camera and companion computer. The mission of the UAV is to survey the airspace for other UAVs entering that airspace in order to intercept them. Therefore the focus of the on-board image processing system should be to detect all UAVs. This problem statement is highly related to sense-and-avoid applications and shows some commonalities.

Many image processing solutions for sense-and-avoid work with rather simple detection of dark spots in front of a bright background (for example blue sky or clouds), one criteria often used to detect moving objects at a long distance is their visible movement against a nearly fixed background. A majority of solutions found in literature focus on changes in illumination of subsequent images in a recorded sequence. One example, including successful flight tests with a ScanEagle UAV and a Cessna 172R are is presented in [4]. This algorithm is a later version of a method presented in [5] and consists of image stabilization, background subtraction, and spatial-temporal filtering. This method works also when the UAV is below the horizon in the camera image. Another method based on multi frame phase correlation is presented in [6] and can be computed very fast but with a small robustness against structured backgrounds and dynamic movements of and within the background. Due to their application context, these systems are designed to detect every possible object in the aircraft flight path, because they all represent a possible collision threat. Discrimination between different kinds of objects like bird or plane is often not necessary.

Over the last years, object detection techniques using machine learning techniques with large neural networks networks have raised and became more and more precise and reliable. In [7] a variant of SegNet is used to detect aircraft at long distances in images with low signal to noise ratio (SNR) with respect to the background. The method presented in [8] compensate for camera ego motion with a feature-based algorithm and uses a convolutional network to search for UAVs in the created difference images. For efficient training of a large neural network often a large amount of data in necessary to achieve a satisfying quality. Yet the acquisition of such labeled data for training a model is still challenging. Recording data in real flight experiments with potential collision scenarios is a hard and dangerous task, while generating hand labeled ground truth data by hand is quite expensive and time consuming.

There have been methods investigated to overcome these limitations by generating synthetic training data. In [9] a small number of real images was used to extract parameters like the standard deviation of the Gaussian kernel for motion blur in order to create similar looking images with common computer graphic methods. In total, as set of over $10\,000$ training images was generated to train an object detector. To optimize the eligibility and variation of the data, artifacts like motion blur or random noise were added to the generated images. The authors of [10] used 3D models of different aircraft from an open source flight simulator on a user defined trajectory in a simulated 3D scene. The generated images have been combined with real video sequences to obtain a labeled data set. With variation of the illumination and rendering parameter, the system was tested with about $14\,000$ images.

III. Detection and Tracking Approach

A. Deep Convolutional Network for Object Detection

The model used in this work is based on a deep convolutional network called tiny YOLOv3, described in [11]. TinyYOLOv3 is an improved and lightweight version of YOLO, see [12] and [13] for further details. These models of deep convolutional networks are designed as single shot detector, reducing high computational costs by combining the tasks of object detection and classification at the same time. Output of the network are the position of each detected object indicated by a bounding box around the object. Together with the position of each object comes a label of the object class together with a certainty value.

Therefore, they provide the ability of object detection as well as object classification at the same time. In the early stage the focus is more on the task of detecting moving objects in the environment next to the defender. Classification of the detected object will, in the future be important part of the project. In a later stage, the system should be able not only to detect moving objects, but also to identify them in order distinguish between UAVs and other moving objects like birds. To solve this classification task properly, a more detailed set of training data must be presented to the network. This data set should include beside examples for small UAVs like quadro- or octocopter also examples for objects like birds or manned aircraft. Still, this is not the scope of this work.

All versions of YOLO are implemented in the Darknet framework, an open source framework for neural networks written in C++ and CUDA. Providing complete trained weight data of the network together with the topological configuration, the network can also be ported to other frameworks like Caffe or Tensorflow. This makes YOLO and its variations ideal candidates for later use cases on different platforms like android systems or ARM computer in combination with a neural compute stick.

B. Preparation of Data and Training

The first step when training a deep convolutional network is the acquisition of annotated training data. Regarding the presented scenario, collecting useful data is a challenging task due to the many possible visual structures a possible intruder can have. In addition, also environmental parameter like illumination of the scene, which are hard to predict can have influence the representation of the intruder in the camera image.

To automate this process, a generator for semi-synthetic images was developed to create a annotated data set by blending background segmented images from UAVs into landscape images. As landscape images, images from different time points during the flight experiments (see section IV for details) were selected. Selecting images from distributed time points results in a variation of the background data in the data set. An overview of selected landscape images is shown in figure 1.

Due to the fact that until now most of dangerous situations or alerts next to controlled airspace or critical infrastructure have been triggered by small multirotor aircraft, this work focuses on the detection and tracking of such multirotor aircraft. Small fixed wing aircraft or helicopters are at the moment not considered as the most urgent threat. So neither fixed wing aircraft nor other objects like birds are included in the training data set for now. As training examples for the model 5 images of multirotor aircraft were selected. Figure 2 shows the images used as examples for generating the training data. These images are freely available to download from different internet sources like [17] or [18].

To generate an image for the training data set, a image of an UAV from the example data was blended into one of the landscape images at randomly chosen position. The position was stored in a database and later used as annotation when training the model. Size of the blended image was in a range between 80 to 400 pixels which corresponds to 5 to 30 percent of the image size captured during the flight experiments (see section IV-A for details). By varying the size of the blended example image it is also possible to simulate UAVs in different distances from the camera and train the model to detect the same objects in different sizes. Each of the 5 selected UAV images was blended 16 times at random positions in randomly scaled size in each of the 10 landscape image, so in total 800 annotated training images have been generated. Fig 3 shows the color-coded distribution of the object patches in the image plane. Most of the patches have been placed in the middle part of the image. Positions close the border had a lower probability based on the assumption that a observed UAV will be visible in the middle of the camera image during most of the time.
Training of the model was done on a NVIDIA GTX 750 TI GPU with 640 CUDA cores. The training was done for $500\,000$ iterations, while 100 iterations took approximate $11\,s$. The learning rate of the model was set to 0.01 for the first $40\,000$ iterations and then changed to 0.001.

C. Temporal filtering

A known issue when using deep neural networks for image classification is the fact, that the detection of the same object at almost the same position in a stream of images can fail in some frames of the video stream. To avoid this problem and establish a constant tracking over the time, we track the position predicted by the neural network with a temporal tracking filter, which manages a list $P_t^{tracked}$ of tracked objects and updates this list with the new prediction from the neural network.
The estimated center position of each connected component $c_{(x,y)}$ and therefore also the position of candidate points can change from frame to frame. However a smooth trajectory for each tracked point is expected and therefore a Kalman filter is used to track the position $c_{(x,y)}$ and to compensate this effect. In addition to the position c_x and c_y of the detected object in the image, the two-dimensional optical speed of the object in x and y direction in the image plane given by Δc_x and Δc_y is also estimated by the filter, resulting in a 4-dimensional state space p of the system. For simplicity, no disturbance and no system noise will be considered by the Kalman filter.
The propagation step of the Kalman filter provides an estimated object position even if the detection algorithm fails to find the object in one or more frames, see section V for results. Further details on the implementation of the temporal tracking filter and the integration in an image analysis pipeline can be found in [15].

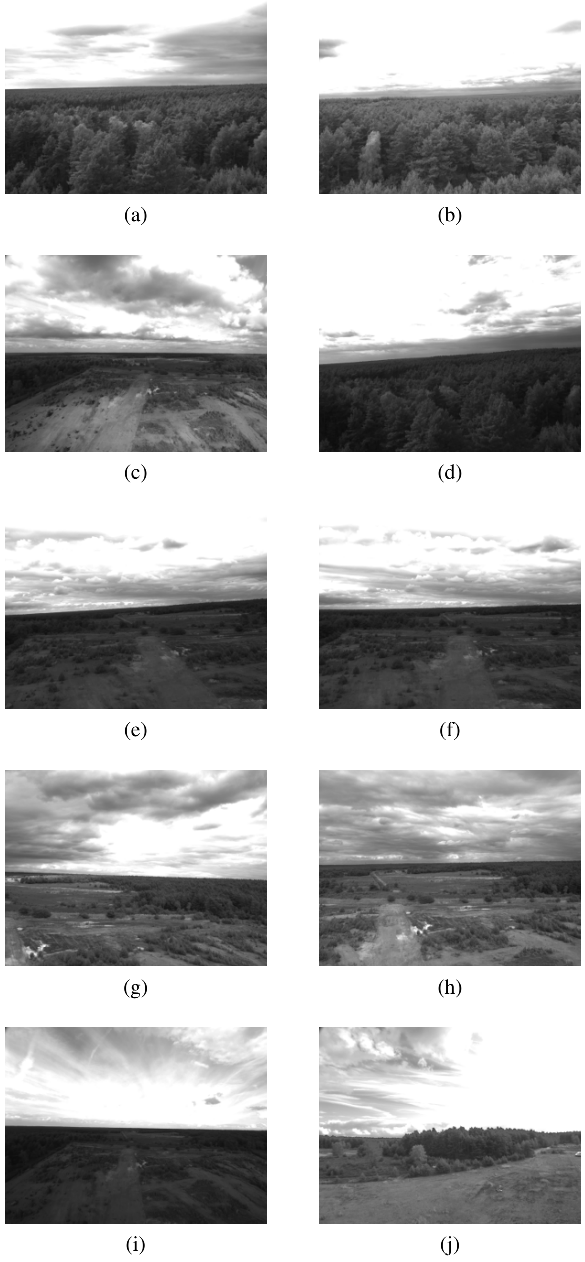

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 1: Landscape images used for data generation.

IV. Flight Experiments

A. Fligth Test Setup

In the flight experiments, the data sequences were recorded on two days with nearly same weather conditions. The defender was represented by DLR's unmanned helicopter ARTIS (Fig. 4), a SwissDrones SDO-50 V2 with $85\,\mathrm{kg}$ MTOW and two $2.8\,\mathrm{m}$ inter meshing rotors. Most of the time during the experiments, the helicopter was operated remotely by a ground control station. Only the takeoff and the landing maneuver have been flown by an experienced safety pilot. The test flights took place in a flat area free from obstacles and with the helicopter hovering at fixed position, with constant heading and about $30\,\mathrm{m}$ altitude over

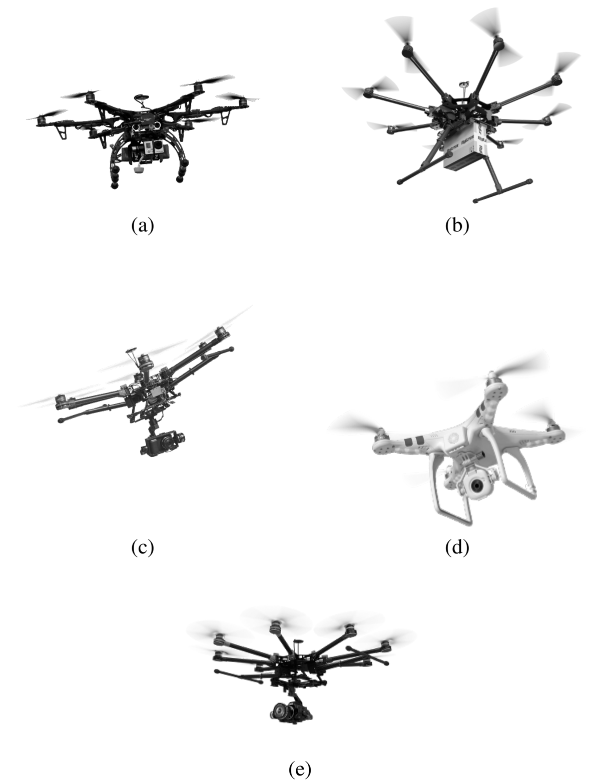


Fig. 2: UAV images used for data generation.

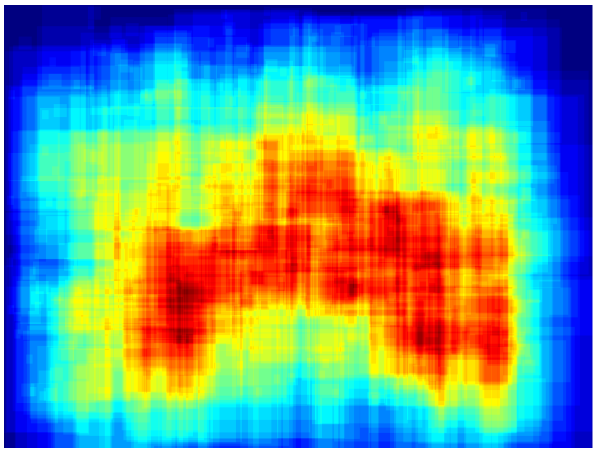


Fig. 3: Color-coded distribution of objects

ground. During the recording of the image sequences, the helicopter's pitch angle was in the range between $\pm 5°$, so that the horizontal line was visible at almost the same vertical location in the cameras field of view. The flight times from take-off to landing range from 7 to 10 minutes for each flight. The used camera was an AVT GT-1380 producing gray scale images of 1360×1024 pixels with a frame rate of 25 fps. The optic is a 4.8 mm RICOH lens for the first seven experiments and a 10.0 mm Cinegon lens for the other seven experiments.
The intruder used in the flight tests was a DJI Inspire Mark 1 (Fig. 5), flown in manual mode the whole time by a second pilot. This UAV has a size of $44 \times 45 \times 38$ cm, with a

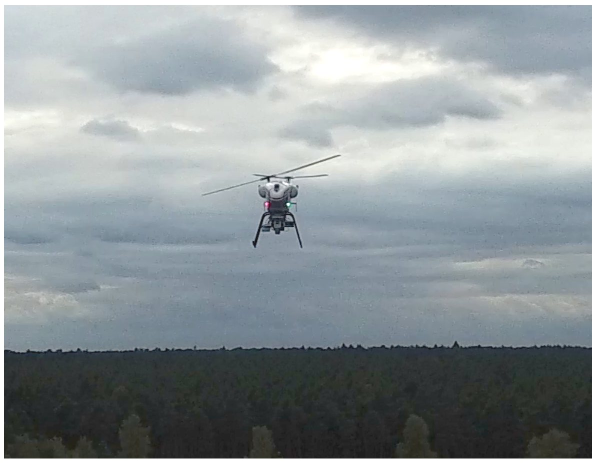


Fig. 4: DLR's unmanned helicopter, representing the defender.

weight of about 2.9 kg an a maximum top speed of 22 m/s. During the experiment, the intruder was flying different maneuvers in front of the defender. The maneuvers and their representation in the image of the camera are described in section IV-B.

B. Image Sequences

From the recorded flight data 14 different sequences of 250 frames each where extracted, to cover a wide combination of imaging situations. First eight sequences have been selected by the following criteria: camera optics, movement of the intruder and the structure of the background.
As mentioned in section IV-A two different optics have been used during the flight tests. With the defender hovering at almost the same location over ground and the intruder performing maneuvers in a similar distance, the effect of increasing the cameras focal length from 4.8 mm to 10.0 mm results in a stronger displacement of the intruder on the camera image. The field of view on the scene can be seen in Fig. 6 a) - b).
Regarding the movement of the intruder relative to the

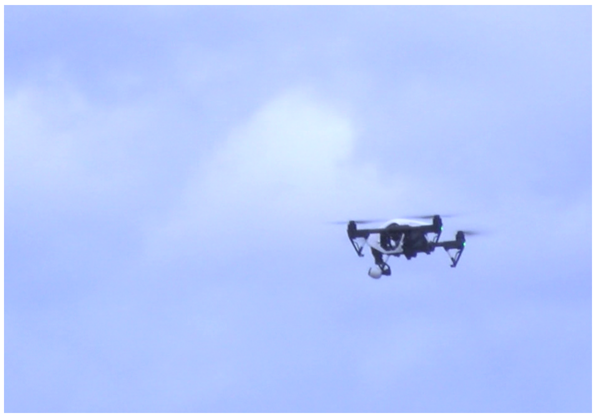


Fig. 5: DLR's DJI Inspire 1, representing the intruder.

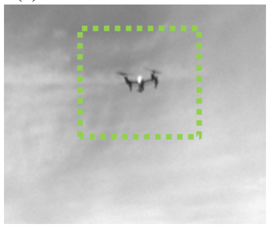
camera image, the movement is differentiated between scaling and translational. A scaling movement occurs when the distance between intruder and defender is increased or decreased, resulting in a variation of size of the intruder on the image plane. A translational movement describes a movement which is parallel to the camera image plane along the x or y axis, see Fig. 6 c) - d).
The other separation of the sequences is given by the image structure of the background in a small area around the position of the intruder. A simple background structure is present when the intruder can bee seen against the bright sky or clouds forming a uniform structure. Complex background is present when the intruder flying in front of more structured background like trees or surface, Fig. 6 e) - f).

(a) Focal length 4.8mm

(b) Focal length 10.0mm

(c) Translational movement

(d) Scaling movement


(e) Simple background

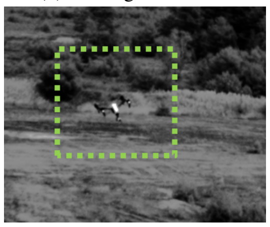

(f) Complex background

Fig. 6: Different sequence characteristics.

In addition to the data presented in [15] the data set was extended by some relevant sequences. Four sequences have been determined, where the intruder entered the scene after about half the time elapsed (ca. 125 frames) for each background and lense type. Due to the design of the experiment, the movement of the intruder is always translational in these sequences. Finally to complete the test data, finally two sequences have been extracted from the recorded material where the intruder was not visible during the whole time (S1NV and S2NV). Considering also sequences with no intruder visible allows to calculate significant parameter like true negatives for the performance of the classificator, see section IV-D for results.
Ther naming convention of the sequences is as follows: the first two characters name the camera setup used for recording, $S1$ denote the lens with $4.8\,mm$ focal length while $S2$ denotes the $10.0\,mm$ optic. The third character describes the movement of the intruder (S=scaling, T=translational, A=appears) while the last character describes the background type (S=simple, C=complex). Sequences with no intruder during the whole time are denoted as NV (not visible). For example S1TC describes a sequence recorded with the $4.8\,mm$ optic where the intruder performs a translational movement against a complex background. Table I gives a detailed overview on the sequences, including the background type (Bkgd), camera focal length F, distance between intruder and defender and the calculated expected size of the intruder on the image plane.

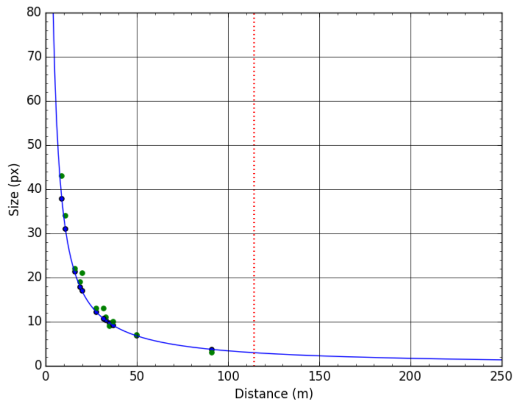
TABLE I: Test sequences

Name	Movement	Bkgd	F (mm)	Size (px)	Dist. (m)
S1SS	scaling	simple	4.8	3 - 7	56 - 94
S1SC	scaling	complex	4.8	9 - 13	28 - 37
S1TS	translational	simple	4.8	21 - 22	16 - 21
S1TC	translational	complex	4.8	34 - 43	8 - 10
S1AS	translational	simple	4.8	10 - 11	33 - 37
S1AC	translational	complex	4.8	13 - 19	19 - 32
S1NV	none	none	4.8	n/a	n/a
S2SS	scaling	simple	10.0	10 - 12	56 - 60
S2SC	scaling	complex	10.0	5 - 6	131 - 145
S2TS	translational	simple	10.0	18 - 26	43 - 53
S2TC	translational	complex	10.0	26 - 33	23 - 25
S2AS	translational	simple	10.0	21 - 25	32 - 33
S2AC	translational	complex	10.0	18 - 22	39 - 41
S2NV	none	none	10.0	n/a	n/a

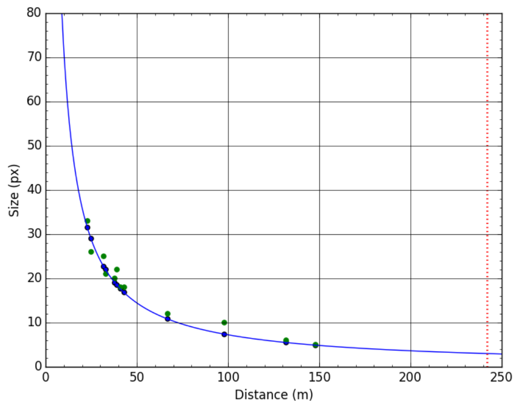
C. Aircraft Positions

On both aircraft, GNSS position data have been logged during the whole experiments. Together with the camera frames from the defenders on-board camera tagged with a time stamp from the flight computers internal system clock, this allows calculation of both aircraft positions for each frame and the distance between them in Cartesian coordinates. Using the intrinsic camera parameter, resulting of the camera calibration, the theoretical size of the intruder on the camera image plane can be computed for each frame. To this aim the size of the intruder in the camera image was approximated by calculating the size of a projection of a simple cube, which has the same size as the intruder and the same distance from the camera.
Fig. 7 shows the theoretical size of a cube with the same dimensions than the intruder (440 mm) on the camera image plane as function of the distance. The green dots indicate the size measured on the real image from the different experiment sequences at the minimum and maximum distance between the intruder and the defender in each sequence. The vertical dotted red line in both figures indicates the distance of the object from the camera for which the size of the object is equal to three pixel. Beyond this distance it is assumed that it is nearly impossible to detect the intruder in the image.

One has to notice that for some data points the size of the intruder is slightly bigger than the expected size calculated from the camera calibration data which is caused by several effects, e.g. the rotor disk of the UAV which was not taken into account for the size of the intruder here.



(a) 4.8mm



(b) 10.0mm

Fig. 7: Size of the intruder in the image depending on the camera distance for both focal lengths.

D. Precision Evaluation

In order to create a usable set of ground truth data, the position of the intruder was marked by hand in all considered frames. To get the position, every frame was presented to a user who tagged the x- and y-position of the intruder. The received position $P_{User}(x, y)$ could then be compared to the calculated output position $P_{Algorithm}(x, y)$ of the detection algorithm to estimate the precision of the detection algorithm. Fig. 8 shows the hand-marked position of the intruder (green circle) and the position detected by the algorithm (blue cross). For each frame F_t from a sequence, the Euclidean distance D_t (in pixel) between the hand marked-position and position detected by the algorithm was computed.

Because it is almost impossible to mark the exact position of the intruder in consecutive frames with the same precision and also the calculated position detected intruder can vary from the ground truth, it is a challenging task to calculate a reliable rate for the quality of the detection. This holds especially for situations, where the distance between intruder and defender is small, and so the area on the cameras image plane is quite large. When the distance between both aircraft is quite large it becomes hard for a user to mark the right spot in the image. Therefore the quality of the presented method was calculated by checking if the distance D_t between the P_{User} and $P_{Algorithm}$ was smaller than the diameter of the calculated area of the intruder on the image plane.

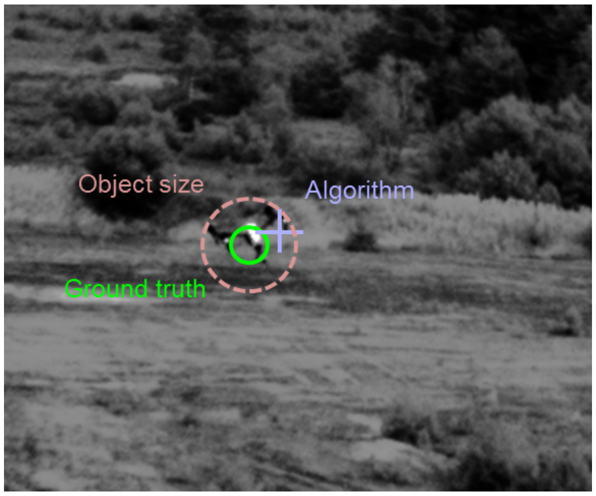


Fig. 8: Hand-labeled and calculated position of the intruder.

V. EVALUATION

Based on the ground truth data, we evaluated the quality of the detection. In parallel, the results were compared with the values calculated by the method presented in [15]. Table II shows the results for the method from [15], the results with prediction from the convolutional network only, and in combination with the temporal tracking filter. The best results for each sequences are marked in bold. Sequences S1SS and S2SC show very poor results, based on the large distance between intruder and defender in that sequences. Here, the image of intruder has a diameter between 3 to 7 pixel. That is by far the smallest diameter in all sequences, see table I for details. For the other sequences, the results of the trained model could be improved by the temporal filter. This effect can also been seen in the confusion matrices in table III and IV. The first one shows the results for the detection by the convolutional network only, the second table shows the results in combination with the tracking filter. True positive and true negative values increase, while false positive and false negative values decrease, indicating a higher performance of the detection algorithm in combination with the tracking filter.

As mentioned in section III-A, an object detection model like tinyYOLOv3 computes not only the position of each detected object. It also computes the certainty level which indicates how well the detected object fits into the estimated class. In the evaluation, every result with a certainty $\geq 25\,\%$ was considered as a valid result. Figure 9 shows the

TABLE II: Test sequence evaluation

Sequence	OptFlow (%)	YOLO (%)	YOLO+Tracking (%)
S1SS	**93.6**	29.6	29.2
S1SC	93.2	66.4	**97.2**
S1TS	93.2	99.6	**99.8**
S1TC	93.6	97.6	**99.8**
S1AS	93.2	**98.0**	96.0
S1AC	83.3	95.6	**96.0**
S1NV	99.6	97.6	**100.0**
S2SS	80.1	89.2	**93.2**
S2SC	**93.6**	0.0	0.0
S2TS	91.2	97.6	**98.8**
S2TC	93.6	98.0	**98.8**
S2AS	82.0	**100.0**	98.8
S2AC	91.6	**77.6**	77.2
S2NV	88.1	98.8	**100.0**

histogram of certainty values. Certainty values have been collected for each frame where an object was detected. Blue bars show certainty values for true positive detected objects, based on the ground truth data. Red bars indicate certainty values of false positive detected objects. About 66 percent of the positive results have a very high certainty rate of 90 percent and higher. For a certainty threshold at 70 percent a positive result would be obtained for about 82 % of these frames.

TABLE III: Confusion matrix

	UAV visible	No UAV visible
UAV detected	1936	67
No UAV detected	569	928

TABLE IV: Confusion matrix with tracking

	UAV visible	No UAV visible
UAV detected	2015	53
No UAV detected	490	942

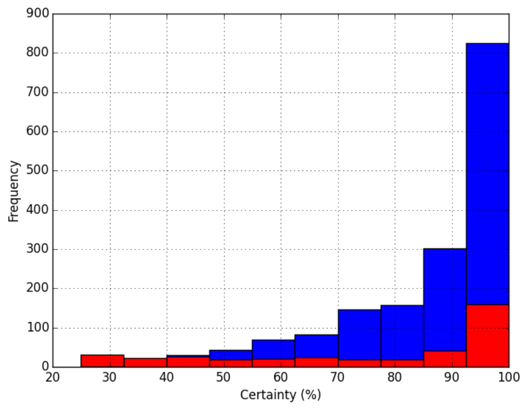


Fig. 9: Histogram of certainty for all evaluated frames.

VI. CONCLUSION AND OUTLOOK

In this paper a method for generating a set of semi-synthetic images for training a deep convolutional network was presented and evaluated. The aim was to train a model with the possibility to detect an unknown UAVs in images. The classifier trained with the generated images showed its ability to detect objects with a high certainty rate, with the special remark that the objects of interest have not been inside the training data set, neither in respect to their shape, pose, size or color.

One important advantage of the tinyYOLOv3 model is that objects can be successfully detected in a single image, so there is no need to regard temporal aspects in computations. Nevertheless, in existence of a continuous stream of images, this results can be combined with a temporal tracking filter to cover the rare situations when the detection by the convolutional neural network fails. Another idea for future implementations is the combination of a convolutional neural network and a temporal tracking filter in a threaded processing pipeline, in order to implement a real time detection system for embedded system.

Independent from the presented method, the image sequences from the experiments cover a wide range of different scenarios regarding different types of backgrounds, movements or sizes of the intruder. In combination with the recorded ground truth data, this yields a useful database for further development and verification of algorithms for UAV detection or sense-and-avoid methods.

For the future, the data and so the number of classes learned by the neural network should be extended in order to distinguish between multirotor, helicopter, fixed wings or birds and especially manned aircraft. Possible research ideas at this time are to analyze the trajectory or the shape and their variation over time of a tracked object in order to distinguish between UAVs and birds. The advantage of using a deep convolutional network like tinyYOLOv3 for this task lies in its capability to identify objects of different classes in a single image, and so to generate a significant answer much faster. Training with more classes and samples will not affect the computational complexity of the model but only change the values of the models weights. The training time might increase and the success rate of the classificator might decrease, still the prediction time will remain the same. The authors believe that these factors will remain in acceptable levels.

References

[1] *Dedrone GmbH*, https://www.dedrone.com, 2018

[2] *TAR Ideal Concepts Ltd*, http://www.tarideal.com/Solutions/ANTI_DRONE_SOLUTIONS/Anti_Drone_Solutions/TA05066/DRONE_JAMMING_VEHICLE, [Online], January 2018

[3] *OpenWorks Engineering*, https://openworksengineering.com/skywall, 2018

[4] D. Bratanov, L. Mejias, J. J. Ford, ”A vision-based sense-and-avoid system tested on a scaneagle UAV”, in *In Unmanned Aircraft Systems (ICUAS)*, 2017 International Conference on, pages 1134-1142. IEEE, 2017.

[5] J. Lai, L. Mejias, J. J. Ford, ”Airborne vision-based collision-detection system”, in *Journal of Field Robotics*, 28(2):137-157, 2011.

[6] F. Schubert, K. Mikolajczyk, “Robust registration and filtering for moving object detection in aerial videos,” in *Pattern Recognition ICPR*, 22nd International Conference 2014, pages 2808-2813

[7] J. James, J. J. Ford, T. L. Molloy, ”Learning to detect aircraft for long range, vision-based sense and avoid systems”, in *IEEE Robotics and Automation Letters*, 2018

[8] D. H. Ye, J. Li, Q. Chen, J. Wachs, C. Bouman, ”Deep Learning for Moving Object Detection and Tracking from a Single Camera in Unmanned Aerial Vehicles (UAVs)”, in *Electronic Imaging*, Imaging and Multimedia Analytics in a Web and Mobile World, pp 466-1-466-6, 2018

[9] A. Rozantsev,V. Lepetita,P. Fuaa, ”On Rendering Synthetic Images for Training an Object Detector”, in *Computer vision and image understanding* ,2014

[10] A. Carrio, C. Fu, J. F. Collumeau, P. Campo, ”SIGS: Synthetic Imagery Generating Software for the Development and Evaluation of Vision-based Sense-And-Avoid Systems”, in *Journal of Intelligent and Robotic Systems*, Volume 84, Issue 1-4, pp 559-574, 2016

[11] J. Redmon, A. Farhadi, ”YOLOv3: An Incremental Improvement” in *IEEE International Conference on Advanced Video and Signal based Surveillance*, 14nd International Conference 2017

[12] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, ”You Only Look Once: Unified, Real-Time Object Detection” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 29nd International Conference 2016

[13] J. Redmon, A. Farhadi, ”YOLO9000: Better, Faster, Stronger” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 30nd International Conference 2017

[14] C. Aker, S. Kalkan, ”Using Deep Networks for Drone Detection”, in *IEEE International Conference on Advanced Video and Signal based Surveillance*, 14nd International Conference 2017

[15] C. Briese, A. Seel, F. Andert, ”Vision-based detection of non-cooperative UAVs using frame differencing and temporal filter.”, in *The 2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2018

[16] S. Petridis, C. Geyeer, S. Singh, ”Learning to Detect Aircraft at Low Resolutions”,

[17] *Free Icons Png*, https://www.freeiconspng.com, 2018

[18] *Stick Png*, http://www.stickpng.com, 2018