BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2024
http://excel.fit.vutbr.cz

# Real-time drone detection and tracking using second drone camera

Samuel Ján Mucha

**Abstract**
My project focuses on enhancing drone target following through a visual detection and tracking system. Many target following methods rely heavily on GPS, which can be unreliable. Therefore, I developed a fast onboard detector that operates independently from ground stations. By integrating the YOLOv8 detector with a lightweight tracker, I achieved real-time and accurate performance. Simulation tests demonstrated the system's applicability, showcasing its potential for improved aerial target following in various environments. Notably, this project underscores the underutilized potential of existing drone cameras for complex detection tasks, all while maintaining real-time inference speeds.

*xmucha13@vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

## 1. Introduction

Drones are used in various fields, like target following and delivery, but tracking them precisely can be tricky. Many current solutions [1] use GPS, but it has limitations, especially in complex environments. This project focuses on making drone tracking better by combining camera-based detection and tracking.

My aim was to improve drone target following by developing a visual detection and tracking system independent of ground stations, addressing the limitations of GPS unreliability, by using camera images for target following. The core solution involves creating a fast onboard detector composed of a YOLOv8 detector and a lightweight tracker for real-time, accurate detection. Evaluation focuses on the system's real-time processing and accuracy.

My solution marks progress in drone target following, achieved through diligent efforts and modest accomplishments. Employing YOLOv8 on a dataset of over 10,000 carefully chosen images, the model achieved a precision of 94% and a recall of 92%; which I find respectable. To address real-time needs, I introduced a simple tracker between detections, aiming for efficiency without sacrificing accuracy. During simulation tests, I aimed to validate my approach's practicality; cautiously optimistic about its potential in real-world scenarios.

## 2. Dataset

My visual detection and tracking system for drone target following relied on a comprehensive dataset. I gathered over 10,000 images featuring drones from various sources, capturing them at different distances and obtained from different vendors. This freely available dataset, hosted on Kaggle, serves as a valuable resource for researchers and practitioners. Covering a wide range of drone types and scenarios, my dataset facilitated thorough training and evaluation of detection algorithm, thus contributing to the advancement of drone target following technology and making possible for YOLOv8 model to achieve its results.

## 3. Detector

As the detector for my system, I opted for YOLOv8. The model underwent training for 20 epochs, utilizing a dataset split into 70% for training, 15% for testing, and 15% for validation. It achieved a precision of 94% and a recall of 92%, what is a good result according to complexity of enviroment. However, despite my efforts to optimize the model by employing techniques such as pruning, quantization, and modifying the backbone, I was unable to attain real-time inference speeds on edge hardware. Additionally, I explored alternative inference engines, notably PyTorch and ONNX, recognizing the importance of selecting the right engine for optimal performance on different hardware platforms. Despite these investigations, YOLOv8 remains the primary choice for its balance of accuracy and speed.

## 4. Trackers

To address the limitations of on-board detection speed, I implemented detection directly on the drone, however at slower speeds. To get rid of blind spots during operation, I integrated a lightweight tracker. Various trackers, including MOSSE, KCF, TLD, CSRT, and VIT, were evaluated to bridge the detection gap, as they offer faster inference than the detector alone. Evaluation metrics encompass parameters such as TLFF (Tracking Length to First Failure), RL (Recover Lenght), RF (Recover Failed percentage), and UNR (UNneccesarry Recover percentage). As a testing base I used a subset of TrackingNet[2] dataset composed of about 2,500 videos averaging 478 frames per video.

From the evaluation results in Table 1, it is evident that faster trackers like MOSSE and KCF tend to exhibit quicker detection but have shorter durations until first failure (TLFF) and longer recovery lengths (RL). For instance, MOSSE and KCF have relatively low TLFF values of 127.12 and 158.01, respectively, indicating faster initial detections but higher failure rates. Conversely, slower trackers such as CSRT and VIT demonstrate longer TLFF values of 188.72 and 228.69, respectively, suggesting more robust initial detections but with slower processing speeds.

Furthermore, while faster trackers may have shorter TLFF values, they often exhibit higher failed recovery percentages (RF) due to their tendency to fail sooner. In contrast, slower trackers like CSRT and VIT demonstrate lower RF values, indicating more reliable recovery from failures. Additionally, slower trackers tend to have lower unnecessary full sequence tracking percentages (UNR), signifying more accurate tracking and fewer instances of unnecessary tracking.

Additionally, I created an ideogram Figure 1 illustrating the workflow, showcasing the sequential process where tracking occurs between every detection. The tracker reinitialization leverages historical images, using bounding box information provided by the detector from past images. This approach ensures continuous target following coverage while maximizing the utilization of available data for effective tracking and detection.

| | TLFF | RL | RF | UNR |
|---|---|---|---|---|
| MOSSE | 127.12 (26.67%) | 32.71 (6.85%) | 46% | 14% |
| KCF | 158.01 (33.34%) | 31.66 (6.57%) | 39% | 16% |
| TLD | 41.67 (8.66%) | 27.08 (5.81%) | 14% | 2% |
| CSRT | 188.72 (39.69%) | 21.05 (5.68%) | 18% | 19% |
| VIT | 228.69 (48%) | 17.75 (4%) | 6% | 28% |

**Table 1.** Evaluation of trackers

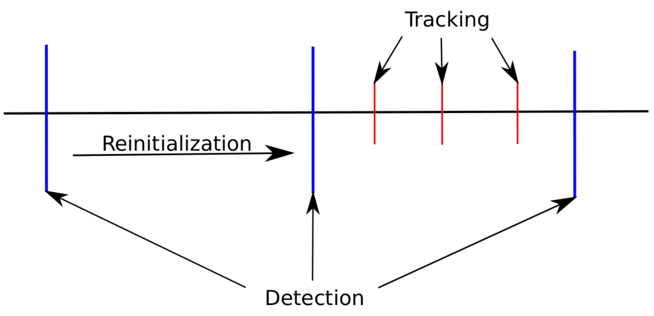

**Figure 1.** Enter Caption

## 5. Simulation

To validate the efficacy of the proposed concept, I conducted testing of detection and tracking within a simulator environment using Gazebo, in conjunction with ROS2[1] and the PX4[2] drone stack. I chose the PX4 drone stack because it is an open-source, widely used controller for drones, offering robust functionality and flexibility. Additionally, I opted for ROS2, which is the de facto standard in the robotics community and is known for its ease of use and extensive support for various robotic applications. The simulated environment consisted of a relatively large grey plane, providing space for drone maneuvers and target following activities. To simulate target following scenarios, I implemented a straightforward box minimization algorithm. This simulation setup enabled simple evaluation of my system's performance in a simulated environment, providing valuable insights into its functionality and potential real-world applicability.

## Acknowledgements

I would like to thank my supervisor doc. Ing. Vítězslav Beran Ph.D. for their help.

[1]https://www.ros.org/
[2]https://px4.io/

## References

[1] RÓBERT HUBINÁK. Increasing the drone pilot orientation by using a second drone, 2023.

[2] Matthias Müller, Adel Bibi, Silvio Giancola, Salman Al-Subaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild, 2018.