






Deep Neural Network-Based Cooperative Visual Tracking Through Multiple Micro Aerial Vehicles

Eric Price , Guilherme Lawless , Roman Ludwig, Igor Martinovic , Heinrich H. Bühlhoff, Michael J. Black , and Amir Ahmad 

Abstract—Multicamera tracking of humans and animals in outdoor environments is a relevant and challenging problem. Our approach to it involves a team of cooperating microaerial vehicles (MAVs) with on-board cameras only. Deep neural networks (DNNs) often fail at detecting small-scale objects or those that are far away from the camera, which are typical characteristics of a scenario with aerial robots. Thus, the core problem addressed in this letter is how to achieve on-board, online, continuous, and accurate vision-based detections using DNNs for visual person tracking through MAVs. Our solution leverages cooperation among multiple MAVs and active selection of most informative regions of image. We demonstrate the efficiency of our approach through simulations with up to 16 robots and real-robot experiments involving two aerial robots tracking a person, while maintaining an active perception-driven formation. ROS-based source code is provided for the benefit of the community.

Index Terms—Visual tracking, aerial systems: Perception and autonomy, multirobot systems.

I. INTRODUCTION

HUMAN/ANIMAL pose tracking and full body pose estimation/reconstruction in outdoor, unstructured environments is a highly relevant and challenging problem. In indoor settings, applications usually make use of body-mounted sensors, artificial markers and static cameras. While such markers might still be usable in outdoor scenarios, dynamic ambient lighting conditions and the impossibility of having environment-fixed cameras make the overall problem difficult. On the other hand, body-mounted sensors are not suitable for some kinds of subjects (e.g., animals in the wild or large crowds of people). Therefore, our approach to the aforementioned problem involves a team of micro aerial vehicles (MAVs), tracking subjects by us-

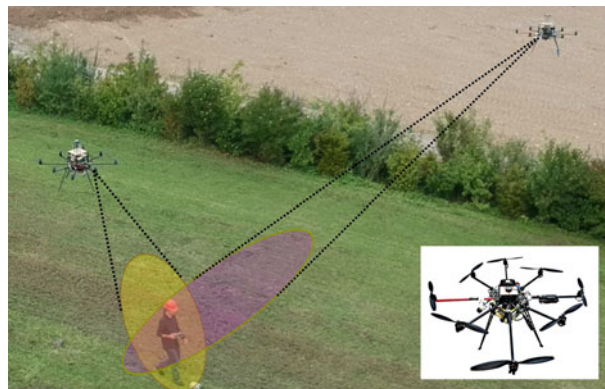


Fig. 1. Two of our self-designed octocopters cooperatively tracking a person while maintaining a perception-driven formation. (In box) Closer view of one octocopter.

ing only on-board monocular cameras and computational units, without any subject-fixed sensor or marker. Among the several challenges involved in developing such a system, multirobot cooperative detection and tracking (MCDT) of a subject's 3D position is one of the most important. It is also the main focus of this letter.

The key component of our MCDT solution is the person detection method suitable for outdoor environments and marker/sensor-free subjects. Deep convolutional neural network-based (DNN) person detection methods are, unarguably, the state-of-the-art. DNNs have consistently shown to outperform traditional techniques for object detection [1]. However, there are only few works that exploit the power of DNNs for visual object detection on board MAVs. The limitations of using DNNs on-board an MAV include i) their computational requirements, ii) high communication bandwidth, if images are shared and iii) information loss when down-sampling high resolution images.

In our approach, the mutual world knowledge about the tracked person is jointly acquired by the multi-MAV system during cooperative person tracking. Leveraging this and using single shot multibox detector (SSD) [3] on a light-weight GPU (Jetson TX1 [2]), we introduce a method that actively selects the relevant region of interest (ROI) in images from each MAV that supplies the highest information content. Our method not only reduces the information loss incurred by down-sampling the high-res images, but also increases the chance of the tracked person being completely in the field of view (FOV) of all MAVs.

The core contributions of this letter are as follows.

- A real-time, continuous and accurate DNN-based multi-robot cooperative detection and tracking method, which is

Manuscript received February 24, 2018; accepted June 6, 2018. Date of publication June 25, 2018; date of current version July 13, 2018. This work was funded by i) the Max Planck Grassroots project MAVOCAP, and ii) the internal budget of the Perceiving Systems department, MPI for Intelligent Systems. This letter was recommended for publication by Associate Editor E. Marchand and Editor F. Chaumette upon evaluation of the reviewers' comments. (*Corresponding author: Amir Ahmad.*)

E. Price, G. Lawless, R. Ludwig, I. Martinovic, M. J. Black, and A. Ahmad are with the Max Planck Institute for Intelligent Systems, Tübingen 72076, Germany (e-mail: eric.price@tuebingen.mpg.de; guilherme.lawless@tuebingen.mpg.de; roman.ludwig@tuebingen.mpg.de; igor.martinovic@tuebingen.mpg.de; black@tuebingen.mpg.de; aamir.ahmad@tuebingen.mpg.de).

H. H. Bühlhoff is with the Max Planck Institute for Biological Cybernetics, Tübingen 72076, Germany (e-mail: hhb@tuebingen.mpg.de).

This letter has supplemental downloadable multimedia material available at <http://ieeexplore.ieee.org>, provided by the authors. The supplementary materials contain a video graphically illustrating the algorithm developed. It shows the video footages of the real and simulation experiments where multiple UAVs track and follow a moving person. This material is 40.9 MB in size.

Digital Object Identifier 10.1109/LRA.2018.2850224

designed and evaluated rigorously for a team of MAVs operating in outdoor environments with only on-board camera perception & computation.

- A method for statistically characterizing the DNN-based detection measurement noise.
- Fully open source ROS-based implementation of our method.

In the next section we situate our work within the state-of-the-art. This is followed by a description of our system design, theoretical details of our proposed MCDT approach and characterization of detection measurement noise in Section III. Section IV and V present our experimental results in real and simulated scenarios, respectively. Section VI concludes the article.

II. STATE-OF-THE-ART

Our approach to motion capture in outdoor scenarios involves multiple MAVs. Unlike [4], where authors use MAV-mounted depth sensors (Xtion) and stream depth images to a ground station, we use only on-board RGB cameras and processing. This allows us to do on-board motion capture in outdoor ambient light conditions. In this paper, we address the key issue of MCDT involved in developing such an outdoor motion capture system.

Multirobot cooperative tracking has been researched extensively over the past years [5]–[7]. While the focus of most of these methods is to improve the tracked target’s pose estimate by fusing information obtained from teammate robots, recent methods, e.g., [6], simultaneously improve the localization estimates of the poorly localized robots in addition to the tracked target’s pose, within a unified framework. However, it is hard to find any cooperative target tracking method that directly facilitates detections through cooperation among the robots. In this letter we do so by sharing independently obtained measurements among the robots, regarding a mutually observed object in the world. Using these shared measurements, our MCDT method at each MAV allows its detector to focus only on the relevant and most informative ROIs for future detections.

Cooperative tracking of targets using multiple MAVs has also attracted attention recently [8]–[10]. While some address the problem of on-board visual detection and tracking using MAVs [11], they still rely on hand-crafted visual features and traditional detection approaches. Moreover, cooperation among multiple MAVs using on-board vision-based detections is not well addressed. In this letter we address both of these issues by using a DNN-based person detector that runs on board the MAVs and sharing the detection measurements among the MAVs to perform MCDT.

Deep CNN-based detectors currently require the parallel processing power of GPUs. For MAVs, light-weight GPUs or embedded solutions are critical requirements. In [12], Rallapalli *et al.* present an overview of the feasibility of deep neural network-based detectors for embedded and mobile devices. Also, several recent works now consider airborne applications of GPU-accelerated neural networks for computer vision tasks. However, they mostly evaluate their performance in offline scenarios, without a flight capable implementation [13]. On the other hand, there are some networks suitable for real time detection and localization of arbitrary objects in arbitrary poses and backgrounds. These include networks, such as, YOLO [14] or Faster R-CNN [15], which are both outperformed in speed and detection accuracy by the SSD Multibox [3]. Hence, in our

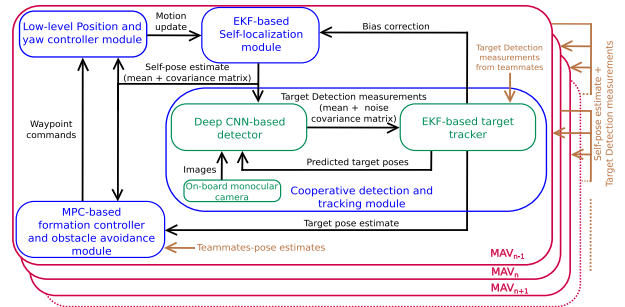


Fig. 2. Overall architecture of our multi-MAV system.

work we use the latter. Furthermore, we ensure its real-time operation for a camera of any given resolution. We achieve this through our cooperative approach involving active selection of the most informative ROI, a method that is novel to the best of our knowledge.

III. SYSTEM OVERVIEW AND THE PROPOSED APPROACH

A. System Overview

Our multi-MAV system does not consist of a central computational unit. Each of our MAVs is equipped with an on-board CPU and GPU to perform all computations. Although our architecture does not depend on a centralized communication network, the field implementation is done through a central wifi access point. Each MAV runs its own instance of the software modules (blue and green blocks) as outlined in Fig. 2. The data shared among MAVs consists of their self-pose estimates and the detection measurements of the person.

B. Preliminaries

Let there be K MAVs R_1, \dots, R_K tracking a person P . Let the 6D pose of the k th MAV in the world frame at time t be given by $[\mathbf{x}_t^{R_k} \ \Phi_t^{R_k}] \in \mathbb{R}^6$, obtained using a self-localization system. $\mathbf{x}_t^{R_k}$ denotes the 3D position and $\Phi_t^{R_k}$ represents the 3 orientation angles. The uncertainty covariance matrix associated to the MAV pose is given as $\Sigma_t^{R_k} \in \mathbb{R}^{6 \times 6}$. Each MAV R_k has an on-board, monocular, perspective camera, rigidly attached to the MAV’s body frame. These cameras do not independently pan, tilt or physically zoom, i.e., they are not attached using a gimbal. Let the position of the tracked person in the world frame at time t be given by $\mathbf{x}_t^P \in \mathbb{R}^3$. We assume that the person is represented by the position of its centroid in the 3D Euclidean space. The uncertainty covariance matrix associated to it is given by $\Sigma_t^P \in \mathbb{R}^{3 \times 3}$. Note that all variables are in the world frame coordinates, unless a left-superscript is used (e.g., I for image frame).

C. DNN-Based Cooperative Detection and Tracking

Algorithm 1, which is a recursive loop, outlines our CDT approach for MAV R_k . Each MAV runs an instance of this algorithm in real time. The algorithm is based on an EKF where the inputs are the tracked person’s 3D position estimate \mathbf{x}_{t-1}^P at the previous time step $t-1$, the covariance matrix Σ_{t-1}^P associated to that estimate and $\mathbf{ROI}_{t-1}^{R_k}$, also computed at $t-1$. The other input is the image $I_t^{R_k}$ at t . Lines 1–11 correspond to the iteration of the algorithm at t .

Algorithm 1: Cooperative Detector and Tracker (CDT) on MAV R_k with inputs $\{\mathbf{x}_{t-1}^P, \Sigma_{t-1}^P, \mathbf{ROI}_{t-1}^{R_k}, \mathbf{I}_t^{R_k}\}$.

- 1: $\{\mathbf{z}_t^{P,R_k}, \mathbf{Q}_t^{P,R_k}\} \leftarrow$ DNN person detector ($\mathbf{I}_t^{R_k}, \mathbf{ROI}_{t-1}^{R_k}$).
 - 2: **Transmit** $\{[\mathbf{x}_t^{R_k}, \Phi_t^{R_k}], \Sigma_t^{R_k}, \mathbf{z}_t^{P,R_k}$ and $\mathbf{Q}_t^{P,R_k}\}$ to all MAVs R_m ; $m = 1, \dots, K$; $m \neq k$
 - 3: **Receive** $\{[\mathbf{x}_t^{R_m}, \Phi_t^{R_m}], \Sigma_t^{R_m}, \mathbf{z}_t^{P,R_m}$ and $\mathbf{Q}_t^{P,R_m}\}$ from all other MAVs R_m ; $m = 1, \dots, K$; $m \neq k$
 - 4: $\{\bar{\mathbf{x}}_t^P, \bar{\Sigma}_t^P\} \leftarrow$ EKF Prediction $\{\mathbf{x}_{t-1}^P, \Sigma_{t-1}^P\}$ using exponentially decelerating state transition model.
 - 5: **for** $m = 1$ to K **do**
 - 6: $\{\mathbf{x}_t^P, \Sigma_t^P\} \leftarrow$ EKF Update $\{\bar{\mathbf{x}}_t^P, \bar{\Sigma}_t^P, \mathbf{z}_t^{P,R_m}, \mathbf{Q}_t^{P,R_m}\}$
 - 7: **end for**
 - 8: $\{\hat{\mathbf{x}}_{t+1}^P, \hat{\Sigma}_{t+1}^P\} \leftarrow$ Predict for next ROI $\{\mathbf{x}_t^P, \Sigma_t^P\}$
 - 9: $\mathbf{ROI}_t^{R_k} \leftarrow$ Calculate next ROI $\{\hat{\mathbf{x}}_{t+1}^P, \hat{\Sigma}_{t+1}^P\}$
 - 10: Update self-pose bias $\{\mathbf{z}_t^{P,R_k}, \mathbf{Q}_t^{P,R_k}, \mathbf{x}_t^P, \Sigma_t^P\}$
 - 11: **return** $\{\mathbf{x}_t^P, \Sigma_t^P, \mathbf{ROI}_t^{R_k}\}$
-

Line 1 of Algorithm 1 performs the person detection using a DNN-based detector on a ROI $\mathbf{ROI}_{t-1}^{R_k}$ provided to it from the previous time step $t-1$ and the image $\mathbf{I}_t^{R_k}$ at the current time step t . Using raw detection measurements on the image (explained below), the detection measurement in the world frame is computed as a mean \mathbf{z}_t^{P,R_k} and a noise covariance matrix \mathbf{Q}_t^{P,R_k} . The detection is performed on the latest available image and uses only the GPU. If this resource is busy processing a previous image, the detection (and therefore the EKF update, Line 6) is skipped. Note that the detection computation time of our DNN is independent of the size of the ROI. It operates on a fixed input size of 300×300 pixel and takes approximately 250 ms for one detection on our dedicated on-board hardware. To this end, we scale down the ROI to 300×300 pixel regardless of the ROI's original size before the DNN takes it as input.

Raw detection measurements on the image consist of a set of rectangular bounding boxes with associated confidence scores and a noise covariance matrix. To obtain a model of this noise, we performed a characterization of the DNN-based detector, explained in Section III-C1. The raw detection measurements are transformed first to the camera coordinates and finally to the world coordinates. This transformation incorporates the noise covariances in the raw detection measurements and the MAV's self-pose uncertainty covariance. Note that the raw measurements are in the 2D image plane, where as the final detection measurements are computed in the 3D world frame. For this, we make a further assumption on the height of the person being tracked. We assume that the person's height follows a distribution $H \sim \mathcal{N}(\mu_H, \sigma_H^2)$. While we assume that the person is standing or walking upright in the world frame, the model can be adapted to consider a varying pose (e.g., sitting down), for instance, by increasing σ_H^2 . Finally, the overall transformation of detections from image frame to world frame measurements also takes σ_H^2 into account. For mathematical details regarding the transformations that include propagating noise/uncertainty covariances, we refer the reader to [16]. Note that we also do not assume that the tracked person is on a flat surface. The terrain could be uneven or sloped. As the 3D world coordinate refers to

the GPS and barometer-derived world coordinate system used by the MAV to self-localize, the assumption on the human height distribution suffices to compute the 3D position of the person in the GPS world coordinate system.

In Lines 2–3 of Algorithm 1 we transmit and receive data among the robots. This includes self-pose estimates (used for inter-robot collision avoidance) and the detection measurements, both in the world frame. Line 4 performs the prediction step of the EKF. Here we use an exponentially decelerating state transition model. When the algorithm continuously receives measurements allowing continuous update steps, predictions behave similar to a constant velocity model. However, when the detections measurements stop arriving, the exponential decrease in velocity allows the tracked person's position estimate to become stationary. This is an important property of our CDT approach, as the ROI is calculated from the tracked person's position estimate (Line 8). In the case of having no detection measurements, the uncertainty in the person's position estimate would continuously grow, resulting in a larger ROI. This is further clarified in the explanation of Lines 8–9.

In Lines 5–7 we fuse measurements from all MAVs (including self-measurements). Since these measurements are in the world frame, fusion is done by simply performing an EKF update for each measurement.

In Lines 8–9 of Algorithm 1 lies the key novelty of our approach. We actively select a ROI ensuring that future detections are performed on the most informative part of the image, i.e., where the person is, while keeping the computational complexity independent of camera image resolution. As the computational complexity of DNN-based detectors grows very fast with the image resolution, using our approach we are still able to use a DNN-based method in real-time and with high detection accuracy. The ROI is calculated as follows. First (Line 8), using a prediction model similar to the EKF prediction and the estimates at the current time step (\mathbf{x}_t^P and Σ_t^P), we predict the state of the person in the next time step $t+1$ as $\{\hat{\mathbf{x}}_{t+1}^P, \hat{\Sigma}_{t+1}^P\}$. Then, in Line 9, using the predicted 3D position of the person, we calculate the position and associated uncertainty of the person's head $\{\hat{\mathbf{x}}_{t+1}^{P_h}, \hat{\Sigma}_{t+1}^{P_h}\}$ and feet $\{\hat{\mathbf{x}}_{t+1}^{P_f}, \hat{\Sigma}_{t+1}^{P_f}\}$. For this we assume the height distribution model for a person, as introduced previously, and that the person is in an upright position. $\hat{\mathbf{x}}_{t+1}^{P_h}$, $\hat{\mathbf{x}}_{t+1}^{P_f}$ and $\hat{\mathbf{x}}_{t+1}^{P_f}$ are back-projected onto the image frame along with the uncertainties and are denoted as $\{I \hat{\mathbf{x}}_{t+1}^P, I \hat{\Sigma}_{t+1}^P\}$ (person's center), $\{I \hat{\mathbf{x}}_{t+1}^{P_h}, I \hat{\Sigma}_{t+1}^{P_h}\}$ (head) and $\{I \hat{\mathbf{x}}_{t+1}^{P_f}, I \hat{\Sigma}_{t+1}^{P_f}\}$ (feet). The center-top pixel of the ROI is now given by

$$\mathbf{ROI}_t^{R_k} \{\mathbf{center}, \mathbf{top}\} = \left(I \hat{\mathbf{x}}_{t+1}^P, I \hat{\mathbf{y}}_{t+1}^{P_h} + 3 I \hat{\sigma}_{\mathbf{y}_{t+1}}^{P_h} \right) \quad (1)$$

and the center-bottom pixel given by

$$\mathbf{ROI}_t^{R_k} \{\mathbf{center}, \mathbf{bottom}\} = \left(I \hat{\mathbf{x}}_{t+1}^P, I \hat{\mathbf{y}}_{t+1}^{P_f} - 3 I \hat{\sigma}_{\mathbf{y}_{t+1}}^{P_f} \right), \quad (2)$$

where $I \hat{\mathbf{x}}_{t+1}^P$ is the x-axis component of $I \hat{\mathbf{x}}_{t+1}^P$, $I \hat{\mathbf{y}}_{t+1}^{P_h}$ and $I \hat{\mathbf{y}}_{t+1}^{P_f}$ are the y-axis components of $I \hat{\mathbf{x}}_{t+1}^{P_h}$ and $I \hat{\mathbf{x}}_{t+1}^{P_f}$, respectively. $I \hat{\sigma}_{\mathbf{y}_{t+1}}^{P_h}$ and $I \hat{\sigma}_{\mathbf{y}_{t+1}}^{P_f}$ are the standard deviations in the y-axis computed directly from $I \hat{\Sigma}_{t+1}^{P_h}$ and $I \hat{\Sigma}_{t+1}^{P_f}$, respectively. Lastly, the left and right borders of the ROI are calculated to match a desired aspect ratio. We chose the aspect ratio (4 : 3) that

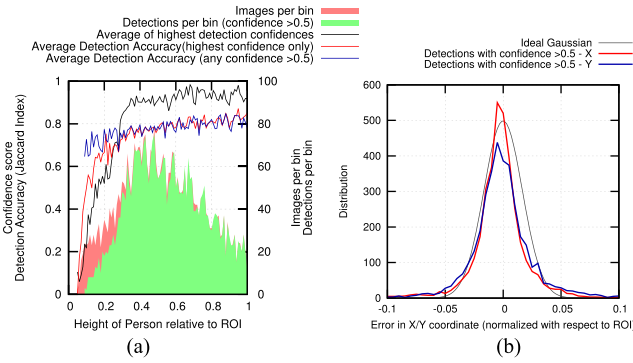


Fig. 3. (a) Detection accuracy w.r.t. the relative person height in the ROI. Images are divided into bins according to the height. (b) Error distribution of SSD Multibox detections in X and Y components.

corresponds to the majority of the training images for optimal detection performance.

Line 10 of Algorithm 1 performs the bias update of the MAV self-pose. Self-pose estimates obtained using GPS, barometer and IMU sensors (as is the case of our self-localization system) often have a time-varying bias [17]. The self-pose biases of each MAV cause mismatches when fusing detection measurements in the world frame, which are detrimental to our MCDT approach because i) the calculated ROI for person detection will be biased and will often not include the person, and ii) the person's 3D position estimate will be a result of fusing biased measurements. To truly benefit from our MCDT approach, we track and compensate the bias in each MAV's localization system (see III-A). We track these biases using an approach based on [17]. The difference between a MAV's own detection measurements and the tracked estimate (after fusion) is used to update the bias estimate.

1) *Noise Quantification of a DNN-based Object Detector:* Our MCDT approach depends on a realistic noise model of the person detector. To this end, we performed a noise quantification of the pre-trained SSD Multibox detector [3]. The output of the detector is, for each detection, a bounding box, a class label and a confidence score. The input image size is defined at training time. We used a pre-trained 300×300 pixel network (SSD300 trained on the PASCAL VOC 2007 + 12 dataset) in this work. We quantify the detection noise with respect to (w.r.t.) the size of the detections, i.e., smaller and distant, or larger and closer to the camera. To this end, we created an extended test set from the PASCAL VOC 2007 dataset with varying levels of downscaling and upscaling of the detected person similar to SSD's training data augmentation in the learning phase. We selected images with a single, non-truncated person to avoid the detection association problem.

Statistical analysis using the pre-trained person detector in our extended test set was performed. As the test set has images of different sizes, we calculate all measures relative to their image size. Fig. 3(a) shows the detection accuracy, using the Jaccard index, relative to the person height. It is evident that even though the detection accuracy for a given minimum confidence threshold is nearly constant w.r.t. the analyzed ROI, the absolute error decreases with a smaller ROI. The chance of successful detection falls significantly for relative sizes below 30% of the ROI and goes down to zero at 10%, thus forming an upper boundary for the desired ROI size. This analysis clearly justifies

the necessity of our MCDT approach with active selection of ROIs.

Further analysis is presented Fig. 3(b), where we show the relative error in the detected person's position over all test images. The error is shown to be well described by a Gaussian distribution. We performed similar analysis on the errors of each detection for the top, bottom, left and right-most points of the detection bounding box at different relative sizes. We found all noises to be similarly well described by Gaussian distributions, without significant correlations between them. Thus, the person detection noise can be well approximated by a constant variance model. The values we obtained for SSD Multibox detection's noise variances for each side of the detection bounding boxes, relative to ROIs, are the following. $\sigma_{\text{top}}^2 = 0.0014$, $\sigma_{\text{bottom}}^2 = 0.0045$, $\sigma_{\text{left}}^2 = 0.0039$ and $\sigma_{\text{right}}^2 = 0.0035$.

D. MPC-Based FC and Obstacle Avoidance Module

The objective of each MAV in our FC is to maintain i) a pre-specified horizontal (normal to gravity direction) distance d_{per} to the tracked person, ii) a pre-specified altitude h_{fix} (in world frame) and iii) its yaw orientation towards the tracked person. Additionally, each MAV must adhere to the constraints of i) maintaining a pre-specified collision distance threshold d_{col} from every teammate MAV and ii) staying within a pre-specified bounding box (e.g., legally permitted flyable zone). Algorithm 2 outlines our FC strategy. Each MAV R_k runs an instance of Algorithm 2 at every time step t . In line 1, MAV R_k computes its desired pose $\tilde{\mathbf{x}}_t^{R_k}$ using simple trigonometry. In line 2, an MPC based planner similar to [18] solves a 3D optimal control problem (OCP) with receding time horizon of size N . We consider nominal accelerations $[\mathbf{u}_t^{R_k}(0) \cdots \mathbf{u}_t^{R_k}(N)]^\top$ as the input vector of the OCP describing the translational motion of the MAV. The discrete-time state of the OCP consists of the MAV R_k 's position $\mathbf{x}_t^{R_k}(n)$ and velocity $\dot{\mathbf{x}}_t^{R_k}(n)$. The discrete-time state-space equations at the n th MPC step (different from time step t) with sampling time Δt are given by

$$\mathbf{u}_t^{R_k}(n) = \ddot{\mathbf{x}}_t^{R_k}(n) \quad (3)$$

$$\begin{bmatrix} \mathbf{x}_t^{R_k}(n+1) \\ \dot{\mathbf{x}}_t^{R_k}(n+1) \end{bmatrix}^\top = \mathbf{A} \begin{bmatrix} \mathbf{x}_t^{R_k}(n) \\ \dot{\mathbf{x}}_t^{R_k}(n) \end{bmatrix}^\top + \mathbf{B} \mathbf{u}_t^{R_k}(n) \quad (4)$$

$$\text{where } \mathbf{A} = \begin{bmatrix} \mathbf{I}_3 & \Delta t \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} \frac{\Delta t^2}{2} \mathbf{I}_3 \\ \Delta t \mathbf{I}_3 \end{bmatrix}.$$

The convex quadratic cost function of the OCP is

$$\begin{aligned} J_{\text{OCP}} = \arg \min_{\mathbf{u}} & \left(\left(\sum_{n=0}^N \left(\|\mathbf{u}_t^{R_k}(n)\|_{\Omega_C}^2 \right) \right) \right. \\ & \left. + \left\| \left[\mathbf{x}_t^{R_k}(N+1) \dot{\mathbf{x}}_t^{R_k}(N+1) \right]^\top - \left[\tilde{\mathbf{x}}_t^{R_k} \mathbf{0} \right]^\top \right\|_{\Omega_{\text{term}}}^2 \right), \quad (5) \end{aligned}$$

subject to the dynamics mentioned above and the following state and input constraints.

$$\mathbf{u}_{\min} \leq \mathbf{u}_t^{R_k}(n) \leq \mathbf{u}_{\max} \quad (6)$$

$$\mathbf{x}_{\min} \leq \mathbf{x}_t^{R_k}(n) \leq \mathbf{x}_{\max}, \dot{\mathbf{x}}_{\min} \leq \dot{\mathbf{x}}_t^{R_k}(n) \leq \dot{\mathbf{x}}_{\max}. \quad (7)$$

Algorithm 2: MPC-Based Formation Controller and Obstacle Avoidance on MAV R_k with Inputs $\{\mathbf{x}_t^P, \mathbf{x}_t^{R_m}; m = 1 : K\}$.

- 1: $\{\hat{\mathbf{x}}_t^{R_k}\} \leftarrow$ Compute Destination Pose $\{\mathbf{x}_t^P, \mathbf{x}_t^{R_k}, d_{\text{per}}, h_{\text{fix}}\}$
 - 2: $\{\mathbf{x}_t^{R_k}(1)\} \leftarrow$ Solve MPC for $\{\hat{\mathbf{x}}_t^{R_k}, \mathbf{x}_t^{R_k}\}$
 - 3: $\{\gamma_t^{R_k}(1)\} \leftarrow$ Compute Desired Yaw $\{\mathbf{x}_t^{R_k}(1), \mathbf{x}_t^P\}$
 - 4: **if** $\mathbf{x}_t^{R_k}(1)$ within collision distance threshold (d_{col}) for any $\{\mathbf{x}_t^{R_m}; m = 1 : K, m \neq k\}$ **then**
 - 5: $\{\hat{\mathbf{x}}_t^{R_k}(1)\} \leftarrow$ Potential field avoidance $\{\mathbf{x}_t^{R_k}(1)\}$
 - 6: $\{\hat{\gamma}_t^{R_k}(1)\} \leftarrow$ Re-compute Desired Yaw $\{\hat{\mathbf{x}}_t^{R_k}(1), \mathbf{x}_t^P\}$
 - 7: **Transmit** $\hat{\mathbf{x}}_t^{R_k}(1), \hat{\gamma}_t^{R_k}(1)$ to Low-level Controller.
 - 8: **else**
 - 9: **Transmit** $\mathbf{x}_t^{R_k}(1), \gamma_t^{R_k}(1)$ to Low-level Controller.
 - 10: **end if**
-

Ω_{term} and Ω_C are the weight matrices for terminal state and input cost. Terminal state is set to be the computed desired position $\hat{\mathbf{x}}_t^{R_k}$ and zero velocity. The MPC solver results in the optimal control inputs $[\mathbf{u}_t^{R_k}(0) \cdots \mathbf{u}_t^{R_k}(N)]^\top$ and the corresponding trajectory $[\mathbf{x}_t^{R_k}(1) \dot{\mathbf{x}}_t^{R_k}(1) \cdots \mathbf{x}_t^{R_k}(N+1) \dot{\mathbf{x}}_t^{R_k}(N+1)]^\top$ towards the desired position (except the desired yaw). We use the first step position component $\mathbf{x}_t^{R_k}(1)$ of the output trajectory as the input to the low-level flight controller (line 9). In line 3, the desired yaw angle is calculated using simple trigonometry with the next 3D position of the MAV and the current pose of the person.

The MPC solver considers obstacle-free space. This is done to avoid non-convexity introduced by considering obstacles directly within the MPC formulation. Subsequently, to account for dynamic obstacles in the environment (teammate MAVs), we use the classical potential-field based obstacle avoidance algorithm on top of the MPC solution. Lines 3–5 check the presence of teammate MAVs within a distance threshold d_{col} of the MAV R_k to activate the avoidance and re-compute $\mathbf{x}_t^{R_k}(1)$ as $\hat{\mathbf{x}}_t^{R_k}(1)$. Consequently, desired yaw angle is also recomputed in line 6. Recall that MAVs communicate their self-pose to each other over wireless network. Generalization of the avoidance method to static obstacles in the environment would be straightforward. An alternative approach would be to enforce a formation geometry that makes the MAVs keep a distance between each other. However, we did not employ that approach for two main reasons. First, its formulation will lead to non-convex constraints and would require a non-convex MPC-based (NMPC) approach. Second, by following a fixed geometry we could be limiting the viewpoints of the UAVs from where the tracked person is visible, e.g., due to static obstacles between the UAV and the tracked person. Finally, either in line 7 or 9, way-point commands consisting of the next time-step pose and desired yaw angle are sent to the low-level flight controller. The latter is essentially assumed to be a position and yaw controller.

IV. REAL ROBOT EXPERIMENTS AND RESULTS

A. Hardware, Software and the Experimental Setup

To evaluate our approach, we conducted real robot experiments on a team of two self-designed 8-rotor MAVs (see Fig. 1)

TABLE I
TRACKED PERSON'S WORLD-FRAME ESTIMATION ERRORS W.R.T GT

| | Twilight | | Noon | |
|----------|----------|-----------------------|----------|-----------------------|
| | Mean (m) | Var (m ²) | Mean (m) | Var (m ²) |
| 3D error | 1.36 | 0.19 | 2.06 | 0.30 |
| 2D error | 0.91 | 0.23 | 1.67 | 0.47 |

tracking a person. Each MAV is equipped with a 2MP HD camera, a computer with an Intel i7 processor, an NVIDIA Jetson TX1 embedded GPU and an OpenPilot Revolution¹ flight controller board. We use the flight controller's position and yaw controller as well as its GPS and IMU-based self-pose estimation (localization) functionalities. Self-pose estimates and IMU sensor data are updated and logged at 100 Hz.

We also mounted an Emlid Reach differential GPS receiver on each MAV and 2 such receivers on either shoulder of the tracked person in order to obtain the ground truth (GT) position estimates of the MAVs as well as the person. Note that the data from this differential GPS system was not used online for flight control or target state estimation during the experiments.

We use a ROS Multi-Master setup over Wifi. Each MAV continuously captures and stores images from the camera at 40 Hz. The on-board GPU runs SSD-Multibox [3]. As described in Section III, ROIs of images down-sampled to 300×300 pixel are sent to the GPU. The detection frame rate achieved is 3.89 Hz. Using these detections, each MAV runs Algorithms 1 and the FC, described in the previous section, to achieve a perception-driven formation in order to accurately track and follow a person walking on the ground with variable speeds and in varying trajectories. In all flight experiments we set $d_{\text{col}} = 5$ m, $d_{\text{per}} = 6$ m and $h_{\text{fix}} = 6$ m above the origin of the experiment field. Note that our actual implementation of the EKF presented in Algorithms 1 also has a backtracking capability.

To prevent inter-MAV collisions, communicated teammate positions are used in the potential field-based method as described previously. Moreover, to compensate for their self-pose inaccuracies, the MAVs need to avoid each other with high enough distance margin. It is therefore necessary that at least nearby robots can communicate with enough bandwidth to avoid colliding with each other. Detection measurements of the person and the MAVs self-pose estimates are very-low-bandwidth data points. We can therefore assume that congestion will not occur even in a relatively larger team of MAVs and at least a percentage of all sent messages reach their destination at all times.

We performed 2 separate flight experiments. While the first was performed at twilight and lasted 220 s, the second was performed at noon for 180 s, both under clear skies. Note that we obtained very similar person tracking estimates from both MAVs as there were little issues with inter-MAV communication. Thus, tracking results from only MAV 1 are discussed.

B. Results and Comparisons w.r.t. Ground Truth

Results of the tracked position estimate of the person from both flight experiments are presented in Table I, Fig. 4 and in the accompanying video and multimedia material. The error is calculated as the difference between the estimated position by our proposed method online during the flight on-board the

¹OpenPilot: <http://www.librepilot.org/site/index.html>

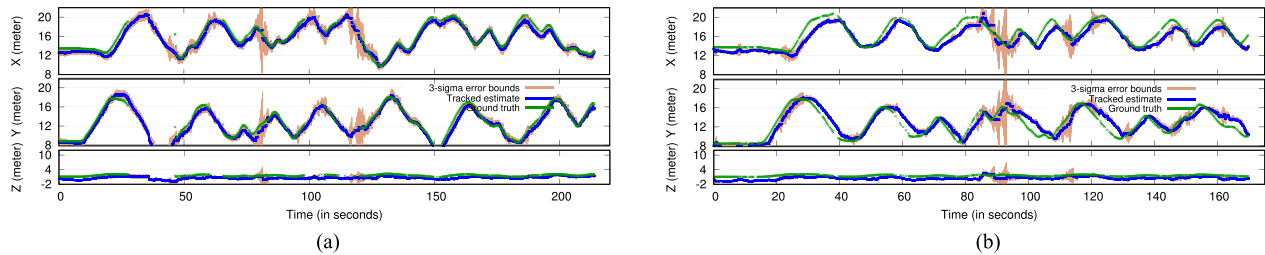


Fig. 4. Tracked person's trajectory comparison for both experiments. (a) Twilight Experiment. (b) Noon Experiment.

TABLE II
MAV 1 LOCALIZATION ERRORS W.R.T GT

| | Twilight | | Noon | |
|----------|----------|-----------------------|----------|-----------------------|
| | Mean (m) | Var (m ²) | Mean (m) | Var (m ²) |
| 3D error | 0.46 | 0.03 | 1.02 | 0.06 |
| 2D error | 0.37 | 0.04 | 0.58 | 0.16 |

TABLE III
MAV 2 LOCALIZATION ERRORS W.R.T GT

| | Twilight | | Noon | |
|----------|----------|-----------------------|----------|-----------------------|
| | Mean (m) | Var (m ²) | Mean (m) | Var (m ²) |
| 3D error | 1.06 | 0.23 | 1.27 | 0.21 |
| 2D error | 0.89 | 0.24 | 0.77 | 0.30 |

MAVs and the corresponding GT obtained by the differential GPS at each time-step. Let the errors in each dimension of the person's position estimate be e_x , e_y and e_z . In Table I, we present the mean and the variance of i) 3D Euclidean distance error ($\sqrt{e_x^2 + e_y^2 + e_z^2}$) and ii) 2D Euclidean distance error ($\sqrt{e_x^2 + e_y^2}$). Accompanying videos display the tracked person's GT (in green) and the estimated trajectories (in blue) with 3-sigma error bounds (in pink).

From Tables II and III, it is evident that the error in the self-pose estimate of the MAVs is high. This is mainly due to the fact that during flight experiments, the MAVs compute a (biased) self pose estimate based on their regular on-board GPS receiver and IMU, both comparable to those used in consumer grade drones and smartphones. This regular GPS receiver contributes to the majority of the self localization error, which can be several meters and different for different GPS modules. Its accuracy also depends on how many satellites it receives data from. Hence, it can vary over the course of a day. This is evident from the different self-pose accuracies of the same MAVs for noon and twilight experiments.

From Table I, we see that the mean error in the tracked estimate is in the order of decimeters. This occurs mainly due to a noticeable high bias in the z-component of the estimate. Contributing factors to this are i) deviation between the actual and the assumed height of the person ii) on-board GPS being less accurate in vertical than in horizontal direction and iii) air pressure fluctuations caused by wind, degrading the MAV's self-localization by affecting the barometric altimeters. While the first reason is likely responsible for the constant part of the bias, the other two reasons contribute to drifts in the bias over time.

When interpreting the tracking results from Table I, it should be noted that all estimates and errors are in the world reference frame. As MAVs have significant self-localization errors of ~ 1 m (see Tabs. II and III), the tracked person's world-frame estimates are heavily affected by them. This also implies that person's tracked position estimate has a significantly lower error in the MAVs' local frames. Furthermore, the tracked estimate (see Tab. I) has a low variance of 0.19 m² for the twilight experiment and 0.30 m² for the noon experiments in the 3D Euclidean errors. This shows that our method is quite precise in jointly

tracking the target and keeping it in the field of view of both MAVs for the whole duration of the experiment, except for a few frames. This can be visualized in the image streams of both MAV's cameras.² The precision of the estimate is also visible in the trajectory plots in these videos.

There is a visible time-lag between the person's GT and the estimated trajectory. This lag is more pronounced in the noon experiment. It is caused mostly by the false negative detections, which happened much more in the noon experiment (see experiment footage). Processing an image frame on-board a MAV takes approximately 250 ms, even if the detection is not successful. For example, algorithm 1 will perform an EKF update only $F + 1$ times 250 ms after F consecutive unsuccessful detections followed by a successful one, assuming that meanwhile no teammate MAV performs a successful detection. The prediction model of EKF, which models the motion dynamics of the tracked person, copes up with this lag to some extent, as the predictions are made at every image frame. However, the mismatch between this model and actual person motion, e.g., sudden change in person's walking direction results in very visible estimation lag (especially noticeable in the video of the noon experiment and the corresponding trajectory plots in Fig. 4(b)). Finally, it must be noted that we do not intend to have a swarm of MAV as a practical method of providing centimeter-accurate absolute position tracking of the target. Our main intent is to have a system that is suitable for human motion capture, for which we require the MAVs to have good relative localization w.r.t. the tracked target and most importantly have convergence and stability in the tracked target estimate. This is demonstrated from the reduced variance in the error using our approach as compared to the other approaches.

C. Results and Comparisons w.r.t. Baseline Methods

Using the recorded MAV camera images and the self-localization pose logs of both MAVs, we ran our MCDT method offline under two different situations. In the first situation, we considered only single MAVs and used the recorded images from each MAV, separately. Boxes in green and pink in

²Complete footage of the experiment: Noon Experiment https://youtu.be/_ZOsw5MWZiQ Twilight Experiment https://youtu.be/LV_82bT25Bc Simulation <https://youtu.be/NHhly19KnrRQ>

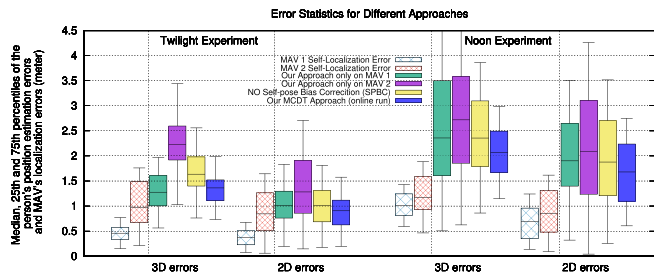


Fig. 5. Box plots of the errors in different situations. Refer Section IV-C for details. Note that blue boxes of the online run also correspond to the plots in the accompanying videos.

Fig. 5 correspond to this situation. In the second situation, we switched off only the self-pose bias correction (SPBC) feature (see sec. III-C) of our MCDT method (yellow boxes in Fig. 5). The blue boxes represent the estimates from the actual online runs. Pattern-filled boxes summarize the localization errors of each MAV obtained during the online run and used during the aforementioned offline runs. The box-plots in Fig. 5 show that both the cooperative aspect and the self-pose bias correction features of our method significantly reduce the tracking errors and make the estimator more precise. Most importantly, MAV 2 is able to keep a good person tracking estimate, despite being poorly localized. Note that the active ROI selection feature of our method is not directly comparable by running offline experiments on the recorded images and turning off the active ROI selection feature. This is because the images were recorded in the actual online run with that feature enabled, which is mainly responsible for keeping the person in the FOV of all MAVs. An offline run without active ROI selection on these images will be biased since the person is present in nearly all recorded images.

D. False Detections and Other Limitations

Any potential false-positive in the environment outside the actively-selected ROI would not be detected. However, multiple persons within the ROI will create ambiguity and could lead to actual false positives. Therefore, within the ROI, we associate detections by discarding all but one, based on a comparison with the tracked estimate. Nevertheless, some failure cases are possible, e.g., i) if multiple persons are present in the FOV before the EKF has converged on a stable estimate, or ii) if a MAV does not detect the true person for a prolonged time period while continuously having false positive detections. In our approach, a MAV in such situation can still recover to the correct person’s estimate if enough other teammates have true positive detections.

V. SIMULATION EXPERIMENTS AND RESULTS

A. Experiments Setup

Simulations were conducted as a software-in-the-loop setup on a single computer with 12-core Intel(R) Core(TM) i7-3970X CPU @ 3.50 GHz with an NVIDIA GK110 (GeForce GTX 780). All software components are ROS nodes, which communicate over TCP/IP networking. The detector runs for each simulated MAV including delay logic to simulate the detection delay of the real-MAV hardware. Physics simulation was handled by Gazebo. The simulated MAVs (firefly model) were equipped with virtual cameras. Their parameters, e.g., framerate, resolution, etc., were set to the same as the real MAVs’ cameras. We

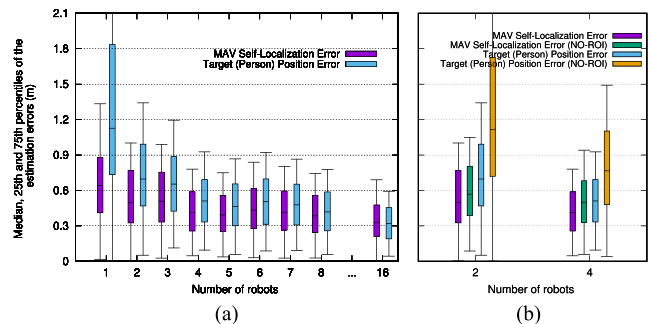


Fig. 6. Simulation experiment 1 and 2. (a) scalability w.r.t. number of MAVs. (b) contribution of active ROI selection.

achieved a simulation real-time factor of 0.15 with 16 MAV’s. GPS inaccuracy and drift were simulated by superimposing a random-walk offset on the GT position of each simulated MAV to match the slowly drifting behavior of the real MAV’s GPS and IMU-based position estimate. During the experiment we recorded each MAV’s target position and self-pose estimate, as well as the corresponding GT directly available from Gazebo.

We define a simulation run as the following sequence: i) spawning of K simulated MAVs and a human actor (at the origin) in Gazebo, ii) waiting for all MAVs to detect the person, assume formation and converge on a stable estimate (~ 20 s), iii) the simulated human actor starts walking on a predefined random trajectory within a 10×10 m perimeter, iv) simulation and recording are stopped after 120 seconds.

B. Experiment 1 - Scalability w.r.t. no. of MAVs

The goal of this experiment is to demonstrate the scalability of the approach with the growing number of MAVs. We conducted runs with $K = [1..8, 16]$ MAVs and perfect networking, i.e., with a detection message loss rate of 0%. Since the behavior is affected by the random GPS drift behavior, each run was repeated 9 times and the results averaged over all repetitions and all MAVs in a run.

Results of this experiment (see Fig. 6(a)) show that the largest improvement in both self-pose and target 3D position estimation is achieved when using 2 MAVs as opposed to a single MAV. Accuracy continues to improve up to 16 MAVs in a team. Beyond that, we could not conduct the experiments because the MAVs would violate the safety threshold distance while maintaining the required distance to the person.

C. Experiment 2 - Contribution of Active ROI Selection

The goal of this experiment is to demonstrate that active ROI selection in the image has a significant impact on the target tracking accuracy. Experiments are conducted with 2 and 4 MAVs, 9 times each in which the region of interest is always the entire camera image. The results are compared to the corresponding runs in simulation experiment 1 (since those already have the active ROI selection enabled).

The results of this experiment (see Fig. 6(b)) show an increase of 113% mean error and 60% median error in the target position estimation when the entire image is used instead of actively-selected ROI in the 2 robot case. For the 4-robot case, the increase is 60% and 50% in mean and median errors, respectively. The main reason for such an increase in estimation error is the reduced number of successful person detections if

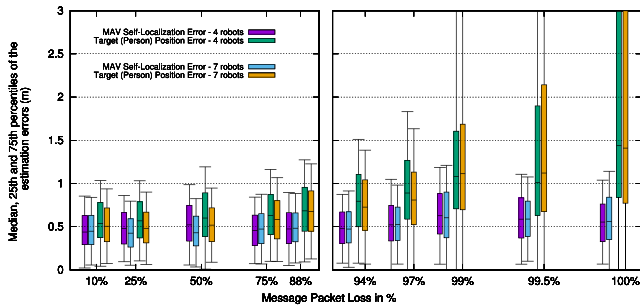


Fig. 7. Simulation experiment 3 - robustness w.r.t. communication loss.

the person's projected size on the image is significantly smaller w.r.t. the processed image size.

D. Experiment 3 - Robustness w.r.t. Communication Loss

The goal of this experiment is to demonstrate the robustness of the approach w.r.t. network packet loss. We conducted runs with $K = [4, 7]$ MAVs and loss rates of $[10, 25, 50, 75, 88, 94, 97, 99, 99.5, 100]\%$. Each run was repeated 6 times and the results averaged over all repetitions and all MAVs in a run. With a simulated loss rate of 100%, each MAV will only receive its own detections, but not from any other MAVs. MAVs were still receiving position information from the other MAVs to prevent collisions.

The results of this experiment (see Fig. 7) show that only around 20% of detection measurements need to be exchanged between MAVs to ensure convergence of the entire swarm to a high accuracy estimate (relative to the MAV self-poses). If the loss rate is higher than 80%, the error in both target and self-pose estimates increases exponentially. It is important to note that a network outage in our real-MAV scenario would also prevent the MAVs from receiving their teammates positions. In such a situation, collision avoidance cannot be guaranteed.

VI. CONCLUSION AND FUTURE WORK

In this letter we presented a novel method for real-time, continuous and accurate DNN-based multi-robot cooperative detection and tracking. Leveraging cooperation between robots in a team, our method is able to harness the power of deep convolutional neural network-based detectors for real-time applications. Through real robot experiments involving only on-board computation and comparisons with ground truth and baseline approaches, we demonstrated the effectiveness of our proposed method. We also showed the feasibility of real-time person detection and tracking with high precision from a team of MAVs which maintain a perception-driven formation. Additionally, we performed noise quantification of the DNN-based detector that allowed us to use it within a Bayesian filter for person tracking. Through extensive simulation experiments we verified that our approach is scalable w.r.t. the number of robots as well as robust to communication failures and gaps.

The work in this letter paves the way for our future goal³ of performing full-body pose capture in outdoor unstructured

scenarios. To this end, we are developing a perception-driven formation controller that not only attempts to minimize the joint 3D position estimate uncertainty in real-time, but also considers the full-body pose reconstruction errors learned offline.

ACKNOWLEDGMENT

We would like to thank the reviewers and the editor for their valuable feedback. We would also like to thank Rahul Tallamraju for his help in implementing the simulations.

REFERENCES

- [1] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, 2015.
- [2] N. Otterness *et al.*, "An evaluation of the nvidia tx1 for supporting real-time computer-vision workloads," in *Proc. IEEE Real-Time Embedded Technol. Appl. Symp.*, 2017, pp. 353–364.
- [3] W. Liu *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.* New York, NY, USA: Springer, 2016, pp. 21–37.
- [4] L. Xu *et al.*, "Flycap: Markerless motion capture using multiple autonomous flying cameras," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 8, pp. 2284–2297, Aug. 2018.
- [5] S. S. Dias and M. G. S. Bruno, "Cooperative target tracking using decentralized particle filtering and RSS sensors," *IEEE Trans. Signal Process.*, vol. 61, no. 14, pp. 3632–3646, Jul. 2013.
- [6] A. Ahmad, G. Lawless, and P. Lima, "An online scalable approach to unified multirobot cooperative localization and object tracking," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1184–1199, Oct. 2017.
- [7] Z. Wang and D. Gu, "Cooperative target tracking control of multiple robots," *IEEE Trans. Ind. Electron.*, vol. 59, no. 8, pp. 3232–3240, Aug. 2012.
- [8] K. Hausman, J. Müller, A. Hariharan, N. Ayanian, and G. S. Sukhatme, "Cooperative control for target tracking with onboard sensing," in *Experimental Robotics*. New York, NY, USA: Springer, 2016, pp. 879–892.
- [9] W. Zheng-Jie and L. Wei, "A solution to cooperative area coverage surveillance for a swarm of MAVs," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 12, p. 398, 2013.
- [10] M. Zhang and H. H. T. Liu, "Cooperative tracking a moving target using multiple fixed-wing UAVs," *J. Intell. Robot. Syst.*, vol. 81, no. 3/4, pp. 505–529, Mar. 2016.
- [11] C. Fu, R. Duan, D. Kircali, and E. Kayacan, "Onboard robust visual tracking for UAVs using a reliable global-local object model," *Sensors*, vol. 16, no. 9, p. 1406, 2016.
- [12] S. Rallapalli *et al.*, "Are very deep neural networks feasible on mobile devices," *IEEE Trans. Circuits Syst. Video Technol.*, 2016. [Online]. Available: https://scholar.google.com/scholar?cluster=8752542356527329884&hl=en&as_sdt=2005
- [13] D. C. De Oliveira and M. A. Wehrmeister, "Towards real-time people recognition on aerial imagery using convolutional neural networks," in *Proc. IEEE 19th Int. Symp. Real-Time Distrib. Comput.*, 2016, pp. 27–34.
- [14] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.
- [15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.
- [16] A. Ahmad, E. Ruff, and H. Bühlhoff, "Dynamic baseline stereo vision-based cooperative target tracking," in *Proc. IEEE 19th Int. Conf. Inf. Fusion*, 2016, pp. 1728–1734.
- [17] W. Whitacre and M. Campbell, "Cooperative geolocation and sensor bias estimation for UAVs with articulating cameras," in *Proc. AIAA Guid. Navig. Control Conf.*, 2009, pp. 1934–1939.
- [18] Y. Liu *et al.*, "Robust nonlinear control approach to nontrivial maneuvers and obstacle avoidance for quadrotor UAV under disturbances," *Robot. Auton. Syst.*, vol. 98, pp. 317–332, 2017.

³Our project AirCap homepage (with links to ROS source code): <http://aircap.is.tuebingen.mpg.de>