2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Bio-inspired visual relative localization for large swarms of UAVs

Martin Křížek[1,3], Matouš Vrba[1], Antonella Barišić Kulaš[2], Stjepan Bogdan[2] and Martin Saska[1]

***Abstract*— We propose a new approach to visual perception for relative localization of agents within large-scale swarms of Unmanned Aerial Vehicles (UAVs). Inspired by biological perception utilized by schools of sardines, swarms of bees, and other large groups of animals capable of moving in a decentralized yet coherent manner, our method does not rely on detecting individual neighbors by each agent and estimating their relative position, but rather we propose to regress a neighbor density over distance. This allows for a more accurate distance estimation as well as better scalability with respect to the number of neighbors. Additionally, a novel swarm control algorithm is proposed to make it compatible with the new relative localization method. We provide a thorough evaluation of the presented methods and demonstrate that the regressing approach to distance estimation is more robust to varying relative pose of the targets and that it is suitable to be used as the main source of relative localization for swarm stabilization.**

***Index Terms*— Aerial systems: perception and autonomy, field robots, multi-robot systems, recognition.**

I. Introduction

A novel marker-less relative localization system for large swarms of UAVs is presented in this paper. This approach is inspired by the behavior of large homogeneous animal groups capable of coherent, collision-free and decentralized movement such as fish or insects. To achieve the impressive large and agile swarms, these animals rely on various sensing modalities (vision, sound, water flow sensors, etc.) but in general they coordinate based on information about the distance and relative bearing of their neighbors [1].

To emulate this behavior in robotic systems, various relative localization approaches are currently employed, most of which rely on a detector providing estimates of relative positions of the agent's neighbors. In this work we focus on monocular visual relative localization because a camera is a light-weight and relatively inexpensive sensor suitable for deployment onboard Size Weight and Power (SWaP)-constrained autonomous vehicles. Specifically, because our motivation is developing a large-scale swarm of lightweight and small aerial robots, we focus on marker-less methods that also reduce the payload requirements of the robots.

Marker-less visual relative localization methods typically estimate relative poses of the targets from bounding boxes of their projection onto the image plane provided by a state-of-the-art detection algorithm such as [2], [3] or [4]. However, this approach suffers inherent bias in the position estimation because dimensions of the bounding box depend not only on the target's size and distance but also on its relative pose. Furthermore, detecting individual neighboring team-members and then estimating their accurate relative pose is unnecessarily complex in case of agile swarming and does not scale well to very large numbers of team-members. Instead, we propose to predict the density of UAVs in the image over distance from the camera. As we demonstrate in sec. IV, this enables reducing the number of parameters of the Convolutional Neural Network (CNN) and thus faster execution and training while improving accuracy of the predictions.

A. Related work

Relative localization sensors carried onboard of UAVs often require markers placed onboard the neighbors to be localized. Typical examples of such methods are the UVDAR system [5] that relies on ultraviolet LED markers and UV-sensitive cameras for detection, or the AprilTag system [6] and similar methods that detect a black-and-white marker and can estimate its pose relative to the camera given its physical dimensions are known.

Marker-less detection and relative localization is in general a harder problem and marker-less methods are more diverse regarding sensor modality with various sensors compromising between accuracy, robustness, price and SWaP restrictions. There are works relying on a LiDAR sensor [7], Stochastic Gradient Descent (UWB) ranging with multi-lateration [8], depth-images from a stereo camera [9] and other modalities, but in this work, we focus on light-weight monocular RGB cameras.

Typical vision-based approaches to marker-less relative localization rely on a CNN-based object detector estimating bounding boxes of projections of the detected targets to the image plane such as the work presented in [10]. State-of-the-art CNN architectures for object detection include approaches focused on maximal detection precision such as YOLO [4] or CenterNet [3], or approaches focused on fast execution with limited computational resources like the EfficientDet [2]. In [11], output of the detector is used to initialize a second CNN which predicts positions of selected keypoints in the image that are then used for 6-DoF estima-

[1]Authors are with the Faculty of Electrical Engineering, Czech Technical University in Prague, Technická 2, Prague 6, krizema3@fel.cvut.cz.

[2]Authors are with the Laboratory for Robotics and Intelligent Control Systems (LARICS), Faculty of Electrical Engineering and Computing, University of Zagreb, 10000 Zagreb, Croatia

[3]This paper is based on the master thesis of the first author https://dspace.cvut.cz/handle/10467/109452.

This work was funded by CTU grant no SGS23/177/OHK3/3T/13, by the Czech Science Foundation (GAČR) under research project no. 23-07517S, and by the European Union under the project Robotics and advanced industrial production (reg. no. CZ.02.01.01/00/22_008/0004590). This work was partially funded by the European Union's Horizon Europe research program Widening participation and spreading excellence, through project Strengthening Research and Innovation Excellence in Autonomous Aerial Systems (AeroSTREAM) - Grant agreement ID: 101071270.

tion of the target's pose. The authors report accuracy up to several centimeters in a small-scale laboratory environment with three UAVs. A similar approach is presented in [12] which proposes a CNN regressor to estimate distance of people in an image based on 2-D positions of their joints. There are also methods for 3-D detection such as [13] where a 3-D bounding box is estimated from RGB data. However, detecting and estimating the relative pose of individual objects does not scale well to large numbers of targets and are overly complex for swarm stabilization.

As mentioned above, our proposed approach is inspired by biological systems. It is assumed that e.g. a sardine does not estimate the relative position of every other sardine within its field of view ([14] reports a mean density of Pacific sardines within a school of $81\,\mathrm{fish\,m^{-3}}$ and smallest school sizes of 5000 fish). Our method aiming to design a system with similar properties is then relevant to crowd-counting CNNs such as [15], [16]. A typical output of such methods is a 2-D density image of the target objects (typically human heads) within an input RGB image [17]. Tackling accurate density estimation for large as well as small groups, [15] proposes a relative-count training loss in addition to the commonly employed density map loss. In the proposed method, a similar loss function is utilized to ensure that even close objects with lower density are accurately detected which is crucial for mutual collision avoidance within a swarm. In contrast to most crowd-counting methods, we do not estimate a 2-D map of object density in an image, but instead a density over distance from the camera, which is crucial for robust and agile swarming in large UAV groups. To our best knowledge, it is an entirely new approach towards reliable stabilization of compact swarms.

B. Contributions

Instead of detecting and localizing individual objects in an image, we propose to directly regress their distribution over distance, which is robust to the number of objects being localized. We demonstrate a CNN architecture implementing this approach that outperforms a relative localization approach utilizing bounding boxes from a state-of-the-art detection CNN while having less parameters and being faster. Comparison with an ideal object detector shows that the proposed method outperforms even the best classical approaches detecting the neighbors in the images individually. Finally, we show that the proposed approach is suitable as a sensing system for swarm stabilization and collision avoidance in several simulated experiments.

C. Problem definition

We assume an UAV equipped with one or multiple monocular cameras, self-localization sensors, and an onboard computer running the necessary software for autonomous flight. Let us call it the *focal UAV*. This focal UAV observes other UAVs in the cameras' field of view and uses the output of a visual relative localization algorithm in feedback to control its position relative to the observed neighbors. The relative localization must be sufficiently accurate and scalable to enable robust, collision-free swarming and it must be efficient enough to process data with a high rate onboard an UAV with limited computational capabilities.

II. Density over distance estimation

The proposed network architecture is a CNN that takes an RGB image represented as a $w_{\text{in}} \times h_{\text{in}} \times 3$ tensor of pixel values as the input. This image is divided into a regular $w_{\text{out}} \times h_{\text{out}}$ grid. The output of the network estimates a discrete distribution $l_{\text{raw}}[d]$ of UAVs over distance for each cell in the grid. We define $l_{\text{raw}}[d]$ as

$$l_{\text{raw}}[d] = |\{\boldsymbol{p} \mid \boldsymbol{p} \in \mathcal{M},\ \|\boldsymbol{p}\| \in [d, d + {}_{\Delta}d]\ \}|, \tag{1}$$

where $d \in \{0,\ {}_{\Delta}d,\ 2{}_{\Delta}d,\ 3{}_{\Delta}d\ \dots\}$ is a distance from the camera, ${}_{\Delta}d$ is a distance discretization step, $\mathcal{M}$ is a set of positions of UAVs in the corresponding image grid cell, and $\boldsymbol{p}$ is a position of an UAV (expressed in the camera's optical frame).

The function $l_{\text{raw}}[d]$ can be interpreted as a histogram where each bin represents a distance range with a width ${}_{\Delta}d$ (i.e. for ${}_{\Delta}d = 1\,\mathrm{m}$, the first bin represents the distance range from $0\,\mathrm{m}$ to $1\,\mathrm{m}$ etc.) and the value of $l[d]$ represents the estimated amount of UAVs in the image within the corresponding range (see Fig. 1). For practical reasons, the number of discretization bins is limited to n_{bin} and UAVs beyond $d_{\max} = {}_{\Delta}d(n_{\text{bin}} - 1)$ are considered to belong to the last bin:

$$l_{\text{raw}}[d_{\max}] = |\{\boldsymbol{p} \mid \boldsymbol{p} \in \mathcal{M},\ \|\boldsymbol{p}\| > d_{\max}\}|. \tag{2}$$

When training the CNN, we do not use the raw histogram $l_{\text{raw}}[d]$ directly for the ground-truth labels, but rather a partially smoothed real-valued function $l_{\text{gt}}[d]$ obtained as

$$l_{\text{gt}}[d] = (l_{\text{raw}} * g)[d], \tag{3}$$

$$g[d] = \begin{cases} \delta[d], & \text{for the closest } k \text{ bins,} \\ G_{\sigma}[d], & \text{otherwise,} \end{cases} \tag{4}$$

where $G_{\sigma}[d]$ denotes a Gaussian kernel with a standard deviation σ, $\delta[d]$ is the identity kernel, and $(f * g)[x]$ denotes discrete convolution of functions $f[x]$ and $g[x]$ w.r.t. x. As discussed in sec. III and sec. IV, this significantly improves training convergence and overall performance of the CNN. Formally, the domain and range of $l_{\text{gt}}[d]$ are

$$\begin{aligned} &l_{\text{gt}} : \mathcal{D} \to \mathbb{R}, \\ &\mathcal{D} = \{0,\ {}_{\Delta}d,\ 2{}_{\Delta}d,\ \dots,\ d_{\max}\},\ {}_{\Delta}d \in \mathbb{R},\ d_{\max} \in \mathbb{R}. \end{aligned} \tag{5}$$

The network's output $l_{\text{o}}[d]$ is then trained to estimate $l_{\text{gt}}[d]$.

For mathematical conciseness, let us interpret the network's output $l_{\text{o}}[d]$ as an n_{bin}-dimensional vector $\boldsymbol{l}_{\text{o}} \in \mathbb{R}^{n_{\text{bin}}}$ so that the i-th element of $\boldsymbol{l}_{\text{o}}$ is equal to $l_{\text{o}}[{}_{\Delta}d(i-1)]$. We shall denote this as

$$\boldsymbol{l}_{\text{o}} = l_{\text{o}}[d]. \tag{6}$$

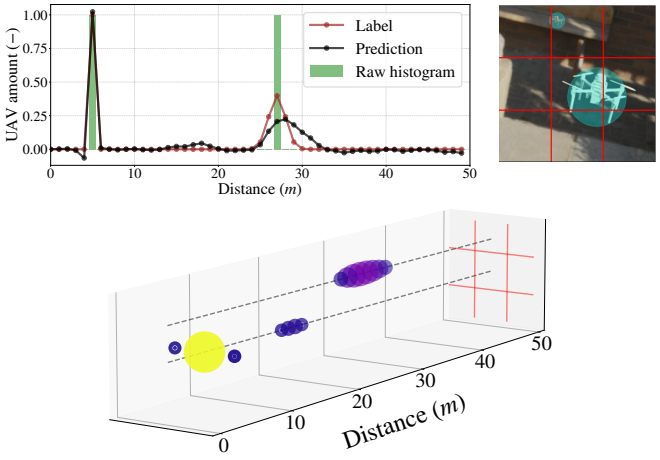


Fig. 1: **Top:** An illustration of the Gaussian smoothing and the CNN prediction of the distribution of UAVs in the image over distance. The input image is shown on the right. **Bottom:** The output of the CNN with $w_{\text{out}} = h_{\text{out}} = 3$ for the input image. The predicted density for a given grid cell and distance is marked with a circle of the corresponding size and color from blue to yellow.

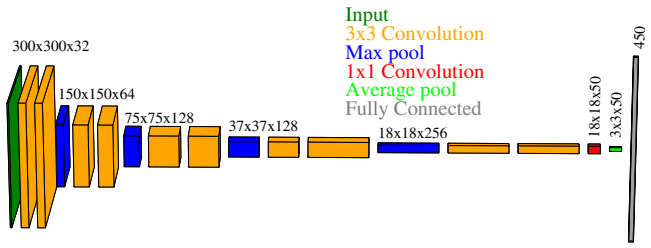


Fig. 2: Visualization of the network architecture for $w_{\text{in}} = h_{\text{in}} = 300$, $w_{\text{out}} = h_{\text{out}} = 3$, and $n_{\text{bin}} = 50$.

A. Network

The neural network (see Fig. 2) consists of a feature extraction head followed by a custom tail with the output as specified in the previous section. For the head, a VGG-like feature extractor structure [18] was employed.

The head is followed by a single convolutional layer with a kernel of size 1×1 and a filter dimension of n_{bin}. This 1×1 convolutional layer similarly as in the related crowd counting approaches [19], [15] weights the feature maps according to their importance for every distance bin in the output vector. Moreover, utilizing the 1×1 convolution instead of dense layers, which are often used at the top of the network, proved to have better performance with a significantly reduced number of parameters which is crucial for online deployment.

The output of this layer is connected to an average pooling layer that produces $w_{\text{out}} \times h_{\text{out}}$ vectors of dimension n_{bin} which are then connected via a single fully connected layer (consisting of $w_{\text{out}} \times h_{\text{out}} \times n_{\text{bin}}$ nodes) to the output of the network. These last layers facilitate regression of the output UAV density vectors $\boldsymbol{l}_{\text{o}}$ for each grid cell from the features acquired by the feature extractor.

B. Loss function

A weighted Euclidean distance between the predicted densities and ground-truth labels was employed as the loss function for training and evaluation of the CNN. Assuming $w_{\text{out}} = h_{\text{out}} = 1$, the error function for a single image is then defined as

$$\mathcal{L} = \|(\boldsymbol{l}_{\text{o}} - \boldsymbol{l}_{\text{gt}}) \circ \boldsymbol{w}\|, \tag{7}$$

where $\boldsymbol{l}_{\text{gt}}$ is the corresponding ground-truth density vector, $\boldsymbol{w}$ is a vector of (non-learnable) bin weights, and the symbol $\circ$ represents element-wise multiplication between vectors. For image grids with $w_{\text{out}},\ h_{\text{out}} > 1$, the vectors $\boldsymbol{l}_{\text{o}}$, $\boldsymbol{l}_{\text{gt}}$, and $\boldsymbol{w}$ for all cells are stacked into meta-vectors $\boldsymbol{L}_{\text{o}}$, $\boldsymbol{L}_{\text{gt}}$, $\boldsymbol{W}$ and then the loss function is calculated analogically to eq. (7) using these meta-vectors.

The weight vector $\boldsymbol{w}$ scales the errors of each distance bin which allows to target specific distance levels that are of higher interest – typically those that are close to the camera (and thus the focal UAV). Having reliable detections near the focal UAV is crucial for mutual collision avoidance and collision-free swarming. However, the majority of the UAVs in our training dataset are far away from the focal drone as discussed in sec. III. The weight vector $\boldsymbol{w}$ counters this bias in the data and significantly improves the detection success rate of the nearby UAVs (see sec. IV).

C. Training setup

The dataset described in the following section was split into 10000 training, 1000 validation, and 5000 testing samples. The training data was used for the Stochastic Gradient Descent (SGD) back-propagation with a batch size of 32 samples. During the training, we gradually decreased the starting learning rate of 0.001 by 20% if the validation loss has not decreased by at least 0.0001 in the last 5 epochs. After 60 epochs, the weights with the lowest loss on the validation data were selected as the best result. The testing data was used for comparing different variants of the CNN model (see sec. IV). We used the ADAM [20] optimizer with the loss function defined in the previous section for the training. The Tensorflow library [21] was employed to implement, train and test our neural network model.

III. Dataset

To our best knowledge, a sufficiently sizable, properly labeled dataset with the large numbers of UAVs required to fully utilize the presented method is not currently publicly available. Therefore, we opted to generate a realistic synthetic dataset of 16000 images using the approach introduced in [22]. The images were generated with 272 HDRI maps for background including natural and urban environments with a wide variety of lighting conditions. To improve the detector's ability to handle challenging environmental conditions and different camera sensors, we applied different color textures to the UAVs as described in [22].

The MRS F450 UAV platform [23] was selected for the dataset as well as the real-world swarming experiments. A 3D model of the platform was designed in Blender and used to generate the data. Our work expands upon [22], which only involved up to 4 UAVs, by developing a procedural pipeline capable of generating UAV groups within the scene

(a) Synthetic image. (b) Real-world photo.

Fig. 3: Comparison of synthetic (a) and real (b) images of the MRS F450 platform [23] used in this work.

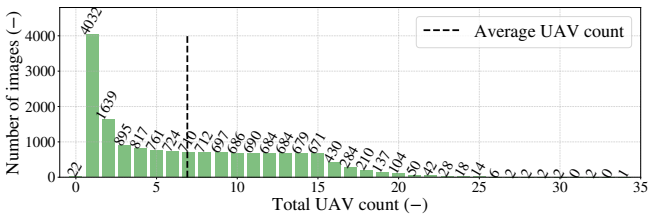


Fig. 4: Distribution of images in the dataset based on the total number of UAVs in the given image. The exact counts are displayed above the bars.

and containing a random number of targets. To avoid introducing too many occlusions, we create only up to one group in close proximity to the camera sensor in the virtual scene, while further away, we create multiple groups positioned randomly. We opt for random generation to maintain a high level of data variance. An example image from the dataset and a comparison with a real-world photo of the platform during the experimental deployment can be seen in Fig. 3.

The images used for the dataset crops were originally generated with optical parameters corresponding to the color camera of the Intel RealSense D435, which is carried onboard the MRS F450 platforms.

Because of a finite resolution of the simulated camera and non-zero dimensions of the UAVs, images with higher numbers of UAVs are less available within a 300×300 pixel crop (see Fig. 4). Enforcing the dataset balancing for all numbers of UAVs would thus degrade its variability, so we only enforce it up to an empirically determined amount of 15 targets as a trade-off between uniformity of the resulting dataset and sufficient data variability.

Furthermore, the image cropping was also biased to select more crops with UAVs near the camera (see Fig. 5) because correctly detecting and estimating the distance of close targets is crucial for safe swarming and collision-avoidance. This causes images with a single or two UAVs to be overrepresented in the resulting dataset (see Fig. 4) due to the nature of the camera sensor.

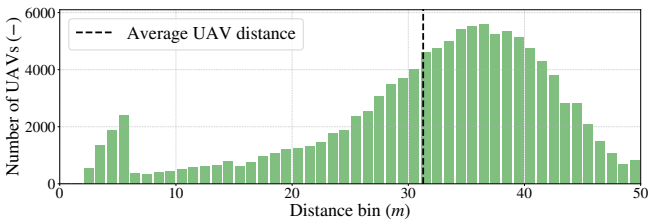


Fig. 5: Distribution of UAVs in the dataset images based on their corresponding distance bin.

A. Labeling

Ground-truth labels for the images in the dataset were generated using known poses of the UAVs in the image relative to the camera according to eqs. (1)-(3). As expressed by eq. (3), we do not train the CNN directly on the ground-truth histogram of UAV count over distance, but rather smooth the UAV distribution by a Gaussian kernel, which is inspired by related works on crowd counting [19], [15]. The kernel is normalized so that the sum of the distribution does not change and thus the total number of UAVs in the image is the same as for the non-smoothed label. Such Gaussian smoothing of the labels reduces the penalization of cases when the CNN makes a minor error in distance estimation for some neighbors and shifts them to a nearby bin. This significantly improves training convergence and the overall performance of the CNN (see sec. IV-B).

The dataset was also labeled for standard object detection algorithms by calculating an axis-aligned bounding box of the projection of each UAV to the camera image. The whole dataset including the ground-truth labels is available online[1].

B. Real-world data

To evaluate capability of the proposed architecture and artificial dataset to generalize to real-world data, we have created a manually annotated dataset of 40 photos from our real-world experiments with UAVs. Because the ground-truth position of the targets in these images is not available, we have employed the nearest-neighbor method for distance estimation described in sec. IV-C. The dataset contains between 1 to 8 UAVs per image in various environments (desert, forest, field) and some of the targets have slightly different hardware configuration (and therefore also visual appearance) than used in training. The evaluation results are described in sec. IV.

IV. Evaluation

To quantify the performance of the proposed method, we use two metrics: an average absolute per-bin error $\bar{e}[d]$, and a total integral error $\bar{T}$. Using notation from the previous sections, these errors are defined as

$$e^i[d] = l^i_{\mathrm{o}}[d] - l^i_{\mathrm{raw}}[d], \qquad \bar{e}[d] = \frac{\sum_{i \in \mathcal{I}} \left| e^i[d] \right|}{\sum_{i \in \mathcal{I}} l^i_{\mathrm{raw}}[d]}, \tag{8}$$

$$T^i = \frac{\left| \sum_{d \in \mathcal{D}} e^i[d] \right|}{\sum_{d \in \mathcal{D}} l^i_{\mathrm{raw}}[d]}, \qquad \bar{T} = \sum_{i \in \mathcal{I}} \frac{T^i}{|\mathcal{I}|}, \tag{9}$$

where i is a single image from the input dataset $\mathcal{I}$ and the upper index denotes a value for the specific image.

The average absolute per-bin error $\bar{e}[d]$ can have two fundamental sources: misdetections (false positives and false negatives), and incorrect estimation of the distance of detected objects. These two causes represent different problems but cannot be distinguished solely based on the per-bin error which is why we also measure the total integral error $\bar{T}$. Together, these measures allow us to estimate which of the

[1] mrs.felk.cvut.cz/perception-for-swarming2023

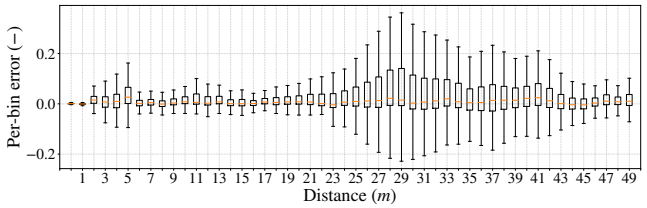


Fig. 6: A boxplot graph of the per-bin errors $e^i[d]$ for $w_{\text{out}} = h_{\text{out}} = 1$.

above-mentioned causes is the main source of error for a given image. A lower $\bar{T}$ indicates that most of the per-bin error is caused by inaccuracy in the distance estimation whereas a higher $\bar{T}$ indicates misdetections of objects in the image. The errors are normalized by the UAV density so that they are suitable for comparing the CNN's performance between images with different numbers of UAVs.

For easier comparison of the different relative localization methods, we define two additional summary metrics

$$\bar{E} = \sum_{d \in \mathcal{D}} \bar{e}[d], \qquad \bar{E}' = \sum_{d=2}^{11} \bar{e}[d], \tag{10}$$

where $\bar{E}$ is the total average per-bin error and $\bar{E}'$ is the total per-bin error for close distances which we have empirically determined to be crucial for collision-free swarming.

Table I presents comparison of the different relative localization methods considered in sections IV-A-IV-C using the previously defined metrics. We also state the relative number of learnable parameters and the measured average execution time t_{exec} from 100 inference runs on a GeForce GTX 1050M GPU. All CNNs variants in these sections were trained and tested using the dataset discussed in sec. III.

A. Grid division comparison

Let us first examine the output of the CNN for a case with a single cell in the output grid (meaning that $w_{\text{out}} = h_{\text{out}} = 1$) to evaluate its distance estimation capabilities. The CNN was trained as described in sec. II-C. Output of the CNN on the testing dataset was then used to calculate the metrics defined in eq. (9) for each data sample. Fig. 6 shows the resulting distribution of the per-bin errors $e^i[d]$. Although the distribution of the error depends on the distribution of the UAVs in the dataset, it may still be observed that the error is symmetrical and the CNN does not significantly bias towards over- or under-estimating the number of objects in a bin. The total average absolute per-bin error was $\bar{E} = 1.31$ and the total integral error was $\bar{T} = 0.142$, so we conclude that the main source of the per-bin error is not misdetection but rather inaccuracies in distance estimates.

Next, we evaluate the proposed CNN for a 3D localization scenario with $w_{\text{out}} = h_{\text{out}} = 3$. As evident from Fig. 7, the performance is almost identical for the 1×1 and 3×3 division. The total errors were also similar to the 1×1 version as evident from Table I. The more granular grid division increases the number of learnable parameters of the architecture by approx. 10%. Therefore, we conclude that estimating the density vectors for multiple cells at once does not negatively impact the performance.

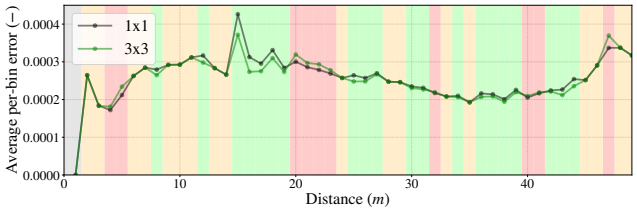
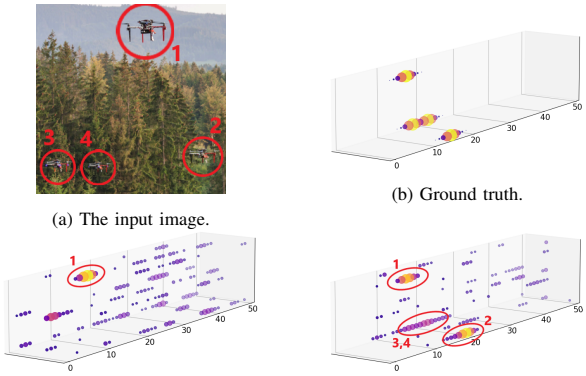


Fig. 7: Comparison of the average absolute per-bin error for the 1×1 and 3×3 output grid divisions. (green if 1×1 is better, red if 3×3 is better, orange if equal)



(a) The input image.

(b) Ground truth.

(c) Only synthetic data.

(d) Fine-tuned on real-world data.

Fig. 8: Comparison of the output of the CNN on a real-wold photo for different training data. The longest axis represents the distance in meters. The amount of UAVs at a given distance and division is represented by color and size of the points. Detections corresponding to the ground-truth UAV positions are highlighted with red ellipses and enumerated.

B. Ablation study

An ablation study of the various design choices made for the proposed CNN architecture was performed. The influence of using a 1×1 convolution kernel as the second-last layer instead of the more conventional choice of a fully connected layer was tested as well as an alternative labeling method to the one described in sec. III-A. Namely, we trained the CNN with the raw labels $l_{\text{raw}}[d]$ (see eq. (1)), partially smoothed labels $l_{\text{gt}}[d]$ (eq. (2)) and fully smoothed labels

$$l_{\text{s}}[d] = (l * G_\sigma)[d]. \tag{11}$$

Although using only the smoothed labels improves the total integral error $\bar{T}$, indicating a lower misdetection rate, the distance estimation accuracy is reduced – esp. in the interval close to the observer as evident by the increased error $\bar{E}'$ (see Table I). This may not be concerning for

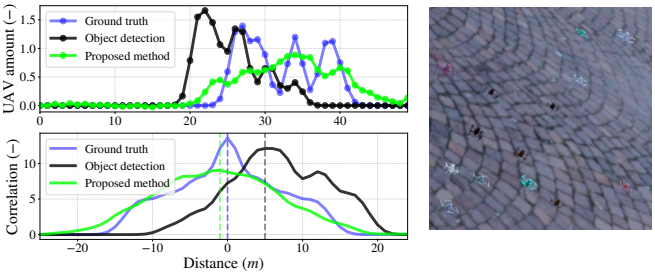


Fig. 9: An example of a challenging image for bounding box-based distance estimation. The top graph shows the predicted and ground-truth UAV densities over distance. The bottom graph presents a correlation (as in signal processing) between the predictions and the ground truth. The highest correlation value is marked with a vertical line.

CNN variant	$\bar{T}$	$\bar{E}$	$\bar{E}'$	params	t_{exec}
Ours (3×3)	0.174	**1.30**	1.27	100%	18 ms
Ours (1×1)	**0.142**	**1.31**	**1.25**	91%	7 ms
F.C. layer (3×3)	0.261	1.60	1.71	285%	8 ms
raw labels (3×3)	0.410	1.53	1.42	100%	18 ms
smooth labels (3×3)	**0.119**	1.34	1.55	100%	18 ms
SotA detector	0.02	**1.31**	**0.93**	N/A	N/A
ideal detector	0 (N/A)	1.35	0.99	N/A	N/A

TABLE I: Comparison of all considered variations of the relative localization approach using various metrics. The two best results are highlighted in bold.

some applications, but for the swarm control considered in this work, we choose the combined smoothing as the best labeling method.

C. Object detection comparison

To show that the proposed approach is not only viable but also performs better than an state-of-the-art detection CNN-based algorithms such as [2], [3], [4], we compare our method to a YOLOv4 Tiny detector with an added YOLO layer. Furthermore, to accommodate for possible future advances in object detection CNN architectures, we also compare the results with a hypothetical ideal object detector which we assume to output predicted bounding boxes of individual UAVs in the image with absolute precision.

For each distance bin, the average width and height of the ground-truth bounding boxes in the training dataset are calculated. Distance of UAVs is then estimated from the detector's output based on dimensions of the predicted bounding boxes by comparing it with the average bounding box width and height for each distance bin. The resulting estimated distance is then the nearest-neighbor bin. The width and height of the bounding box can differ for a single UAV at a given distance if the relative pose of the UAV to the camera changes (e.g. a different pitch or roll rotation). This causes an inherent error when estimating the distance of a non-spherical target solely from its bounding box even for an absolutely accurate bounding box prediction.

The hypothetical ideal detector performs the best for close ranges but our method has a better total accuracy distance because it learns to regress the distance directly from the visual features and thus does not suffer from this inherent error. An illustrative example is shown in Fig. 9, where the observing (focal) UAV is tilted. It may be seen in the correlation graph in the Figure that our method has a lower bias in the distance estimation as the correlation maximum is closer to zero. The hypothetical detector has a zero total integral error $\bar{T}$, but this result has low significance because we did not consider misdetections by the ideal detector.

Our method outperforms the state-of-the-art detector using all metrics because not all of the UAVs are successfully detected by the EfficientDet CNN and the bounding boxes are less accurate which negatively influences the distance estimate. Furthermore, our method has a lower number of learnable parameters and a faster execution time.

D. High density range data

The following section evaluates the performance of the proposed model on data with high density range (difference between the numbers of UAVs in the images with the least and the most UAVs). The density range of the dataset described in sec. III is from 1 UAV up to 35 UAVs per image. We generated an additional dataset containing all counts from 1 UAV up to 150 UAVs per image. We both trained and tested the proposed model on the high density range dataset to demonstrate the scaling capability of the proposed method. We trained and tested the object detector on the additional dataset as well to compare with the results of the proposed method. Table II presents the results and comparison of both methods.

CNN variant	$\bar{T}$	$\bar{E}$	$\bar{E}'$	params	t_{exec}
SotA - dense	0.38	1.09	1.02	N/A	N/A
Ours (3×3) - dense	**0.157**	**0.61**	**0.54**	100%	18 ms

TABLE II: Comparison of the performance of the proposed method and the object detector-based method on the data with high density range. The best results are highlighted in bold.

E. Real-world data generalization

The dataset described in sec. III-B was used for a qualitative analysis of the proposed method's capability to generalize to real-world data. The data were split to 30 images which were used for fine-tuning the trained model and 10 images that were used for testing. Fig. 8 shows a comparison of the CNN's output on a sample from the real-world data with and without fine-tuning. After carefully analyzing the results, we conclude that fine-tuning on real-world data significantly improves performance of the CNN even with such a relatively small dataset size and that the CNN is capable of reproducing the results on the synthetic dataset using this method, which is in line with the results previously reported in [22] for single object detection.

V. Conclusion

A novel technique for large-scale visual relative localization of agents within a swarm of UAVs is introduced in this paper. We propose a regression-based approach that employs a CNN for the estimation of the neighbor density over distance, which enables a more precise distance estimation compared to the state-of-the-art single object detection methods and improves scalability with respect to the number of neighbors. Our evaluation on both synthetic and real-world data shows the robustness of the proposed approach to various poses of the target UAVs and its suitability as the primary source of online onboard relative localization for swarm stabilization. We believe this work will help with the transition from relatively small-scale swarm deployments in controlled environments to large, decentralized, and infrastructure-independent swarms in real-world conditions. In the future, we aim to design a swarming rule intended to utilize the proposed relative localization method and to extend the real-world dataset to improve the fine-tuning for real-world deployments. Finally, we intend to carry out real-world swarming experiments using the introduced method in feedback with the swarming rule.

REFERENCES

[1] J. E. Herbert-Read, "Understanding how animal groups achieve coordinated movement," *Journal of Experimental Biology*, vol. 219, no. 19, pp. 2971–2983, 2016.

[2] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," in *Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 781–10 790.

[3] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "CenterNet: Keypoint triplets for object detection," in *International Conference on Computer Vision*, 2019, pp. 6569–6578.

[4] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *preprint, arXiv:1804.02767*, 2018.

[5] V. Walter, M. Vrba, D. B. Licea, and M. Saska, "Distributed formation-enforcing control for uavs robust to observation noise in relative pose measurements," *preprint, arXiv:2304.03057*, 2023, submitted to T-RO.

[6] M. Krogius, A. Haggenmiller, and E. Olson, "Flexible layouts for fiducial tags," in *International Conference on Intelligent Robots and Systems*, 2019, pp. 1898–1903.

[7] M. Vrba, V. Walter, and M. Saska, "On onboard LiDAR-based flying object detection," *preprint, arXiv 2303.05404*, 2023, submitted to T-RO.

[8] K. Guo, X. Li, and L. Xie, "Ultra-wideband and odometry-based cooperative relative localization with application to multi-uav formation control," *Transactions on Cybernetics*, vol. 50, no. 6, pp. 2590–2603, 2019.

[9] M. Vrba, D. Heřt, and M. Saska, "Onboard marker-less detection and localization of non-cooperating drones for their safe interception by an autonomous aerial system," *Robotics and Automation Letters*, vol. 4, no. 4, pp. 3402–3409, 2019.

[10] M. Vrba and M. Saska, "Marker-less micro aerial vehicle detection and localization using convolutional neural networks," *Robotics and Automation Letters*, vol. 5, no. 2, pp. 2459–2466, 2020.

[11] M. Pavliv, F. Schiano, C. Reardon, D. Floreano, and G. Loianno, "Tracking and relative localization of drone swarms with a vision-based headset," *Robotics and Automation Letters*, vol. 6, no. 2, pp. 1455–1462, 2021.

[12] L. Bertoni, S. Kreiss, and A. Alahi, "Monoloco: Monocular 3D pedestrian localization and uncertainty estimation," in *International Conference on Computer Vision*, 2019, pp. 6861–6871.

[13] G. Brazil and X. Liu, "M3D-RPN: Monocular 3D region proposal network for object detection," in *International Conference on Computer Vision*, 2019, pp. 9287–9296.

[14] R. H. Love, J. M. Fialkowski, and T. H. Jagielo, "Target strength distributions of Pacific sardine schools: Model results at 500 Hz to 10 kHz," *The Journal of the Acoustical Society of America*, vol. 140, no. 6, pp. 4456–4471, 2016.

[15] L. Zhang, M. Shi, and Q. Chen, "Crowd counting via scale-adaptive convolutional neural network," in *Winter Conference on Applications of Computer Vision*, 2018, pp. 1113–1121.

[16] G. Gao, J. Gao, Q. Liu, Q. Wang, and Y. Wang, "CNN-based density estimation and crowd counting: A survey," *preprint, arXiv:2003.12783*, 2020.

[17] V. A. Sindagi and V. M. Patel, "A survey of recent advances in CNN-based single image crowd counting and density estimation," *Pattern Recognition Letters*, vol. 107, pp. 3–16, 2018.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *preprint, arXiv:1409.1556*, 2014.

[19] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Conference on Computer Vision and Pattern Recognition*, 2016, pp. 589–597.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *preprint, arXiv:1412.6980*, 2014.

[21] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A system for large-scale machine learning," in *Symposium on Operating Systems Design and Implementation*, vol. 16, 2016, pp. 265–283.

[22] A. Barisic, F. Petric, and S. Bogdan, "Sim2Air-synthetic aerial dataset for UAV monitoring," *Robotics and Automation Letters*, vol. 7, no. 2, pp. 3757–3764, 2022.

[23] D. Hert, T. Baca, P. Petracek, V. Kratky, V. Spurny, M. Petrlik, M. Vrba, D. Zaitlik, P. Stoudek, V. Walter, P. Stepan, J. Horyna, V. Pritzl, G. Silano, D. Bonilla Licea, P. Stibinger, R. Penicka, T. Nascimento, and M. Saska, "MRS modular UAV hardware platforms for supporting research in real-world outdoor and indoor environments," in *International Conference on Unmanned Aircraft Systems*, 2022, pp. 1264–1273.