A Low-level Active Vision Framework for Collaborative Unmanned Aircraft Systems

Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg, Karl Granström, Fredrik Heintz, Piotr Rudol, Mariusz Wzorek, Jonas Kvarnström, Patrick Doherty

Linköping University

Abstract. Micro unmanned aerial vehicles are becoming increasingly interesting for aiding and collaborating with human agents in myriads of applications, but in particular they are useful for monitoring inaccessible or dangerous areas. In order to interact with and monitor humans, these systems need robust and real-time computer vision subsystems that allow to detect and follow persons.
In this work, we propose a low-level active vision framework to accomplish these challenging tasks. Based on the LinkQuad platform, we present a system study that implements the detection and tracking of people under fully autonomous flight conditions, keeping the vehicle within a certain distance of a person. The framework integrates state-of-the-art methods from visual detection and tracking, Bayesian filtering, and AI-based control. The results from our experiments clearly suggest that the proposed framework performs real-time detection and tracking of persons in complex scenarios.

Keywords: Visual Tracking, Visual Surveillance, Micro UAV, Active Vision

1 Introduction

Micro unmanned aerial vehicles (micro UAVs) are becoming popular for aiding in numerous applications such as search and rescue, inspection, early warning, forest-fire reconnaissance and remote localization of hazardous radio-active materials. Generally these platforms have been remotely piloted with no active autonomous vision capabilities. However, in recent years significant amount of research has been done to develop active vision based functionalities for such platforms. The purpose of a vision component is to interpret the rich visual information captured by onboard cameras. In this paper, we propose a robust active vision framework for collaborative unmanned aircraft systems.

Several active vision frameworks for micro UAVs have been reported in recent years [20, 24, 18]. A vision based method for path planning of micro UAVs using a three-dimensional model of the surrounding environment is proposed in [20]. Yu et al. [24] propose a 3D vision system for estimating the UAV height over ground, used in the control loop of the helicopter. The work of [18] proposes a

hardware and software system for micro UAVs that is capable of autonomous flight using onboard processing for computer vision. In this work, we tackle the challenging problem of developing an active vision framework for robust detection and tracking of persons in complex environments, which is necessary for stable virtual leashing, i.e. following a person at a predetermined distance.

Generally most approaches to object detection are based on the learning-from-examples paradigm [19, 4, 7]. In recent years, discriminative, part-based methods [25, 7] have been shown to provide excellent performance for person detection. These methods rely on intensity information for image representation [15, 4] and latent support vector machines for classification. A sliding window technique is then employed to scan an image at multiple scales. Contrary to the intensity based methods, Khan et al. [12] propose to use color information within the part-based framework of Felzenszwalb et al. [7]. The method employs color attributes for color representation while providing excellent detection performance on benchmark datasets. In our framework, we have the option to select both intensity and color based detection models for person detection.

Tracking of visual objects in an image sequence is a challenging computer vision problem. Typically methods employ either generative or discriminative approaches to tackle the visual tracking problem. The generative methods [1, 13, 14] work by searching for regions that are most similar to the target model. A template or subspace based model is usually employed. The discriminative approaches [8, 26, 10] work by differentiating the target from the background using machine learning techniques. Recently, Danelljan et al. [5] proposed an adaptive color attributes based tracking approach that outperforms state-of-the-art tracking methods while operating at real-time. In our framework, we incorporate this tracking approach due to its robustness and computational efficiency.

Multiple object tracking (MOT) is the processing of multiple detections from multiple objects such that reliable estimates of the number of objects, as well as each object's state, can be obtained. The PHD filter is a computationally feasible first order approximation of the Bayesian multiple object tracking filter [16, 17], and its output is a joint Bayesian estimate of the number of objects and their respective states. In comparison to classic MOT filters such as Joint Probabilistic Data Association Filter (JPDAF) or Multi-Hypothesis Tracker (MHT), see e.g. [2], the PHD filter does not require a solution to the data assocation problem. In our framework, we use a PHD filter to improve the tracking results obtained by the adaptive color based attributes based tracking.

In this work we propose a low-level active vision framework for unmanned aircraft systems based on the LinkQuad platform. Our framework employs state-of-the-art object detection, object tracking, Bayesian filtering and AI-based control approaches. Our experimental results clearly demonstrate that the proposed framework efficiently detects and tracks persons in both indoor and outdoor complex scenarios. Our framework can thus be used for stable virtual leashing.

The rest of the paper is organized as follows. Section 2 describes the used Micro UAV platform. Section 3 presents our active vision framework. Experimental results are provided in section 4. Finally, conclusions are provided in section 5.

Fig. 1: LinkQuad platform with the color camera sensor module.

2 Active Vision Platform

The micro UAV platform used for the system evaluation is a LinkQuad, see figure 1. It is a highly versatile autonomous UAV. The platform's airframe is characterized by a modular design which allows for easy reconfiguration to adopt to a variety of applications. Thanks to a compact design (below 70 centimeters tip-to-tip) the platform is suitable for both indoor and outdoor use. It is equipped with custom designed optimized propellers which contribute to an endurance of up to 30 minutes. Depending on the required flight time, one or two 2.7 Ah batteries can be placed inside an easily swappable battery module. The maximum take-off weight of the LinkQuad is 1.4 kilograms with up to 300 grams of payload.

The LinkQuad is equipped with in-house designed flight control board - the LinkBoard. The LinkBoard has a modular design that allows for adjusting the availabl computational power depending on mission requirements. Due to the available onboard computational power, it has been used for computationally demanding applications such as the implementation of an autonomous indoor vision-based navigation system with all computation performed on-board. In the full configuration, the LinkBoard weighs 30 grams, has very low power consumption and has a footprint smaller than a credit card. The system is based on two ARM-Cortex microcontrollers running at 72 MHz which implement the core flight functionalities and optionally, two Gumstix Overo boards for user software modules. The LinkBoard includes a three-axis accelerometer, three rate gyroscopes, and absolute and differential pressure sensors for estimation of the altitude and the air speed, respectively. The LinkBoard features a number of interfaces which allow for easy extension and integration of additional equipment. It supports various external modules such as a laser range finder, analogue and digital cameras on a gimbal, a GPS receiver, and a magnetometer.

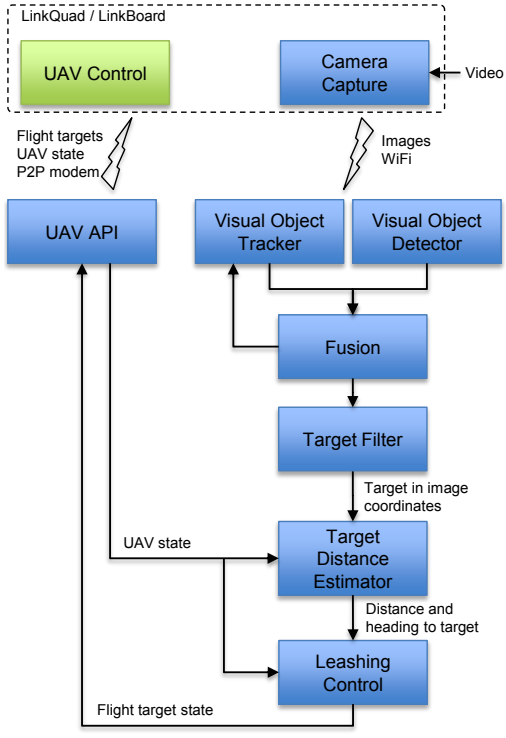


Fig. 2: Overview of the system components in our framework. Images captured by the camera are input to the object detection and tracking modules. The outputs of the two modules are combined in the fusion module. The results are further refined in the filtering component, which provides the image coordinate locations of the targets. The distance estimation component calculates the distance to the targets, which is used for leashing control.

Experiments presented in this paper are performed using the LinkQuad UAV platform with a mounted FireFly MV color camera sensor manufactured by Point Grey Research Inc.[1]. The camera module also includes two servo mechanisms that allow for chaining the pan and tilt of the camera. The sensor is interfacing with the onboard Gumstix modules over the USB 2.0 interface.

3 Active Vision Components

The active vision framework consists of five main parts, namely object detection, visual tracking, target filtering, distance estimation and leashing control. Figure 2 depicts a logical schematics of the presented framework. Images captured onboard the UAV are distributed through a ROS topic by a *Camera Capture* ROS node. The node is running on a Gumstix module. The *Visual Object Tracker*

[1] http://ww2.ptgrey.com/USB2/fireflymv

and *Visual Object Detector* use this image stream as input. The object detector generates person detections in the images, while the visual tracker estimates the locations of visual object hypotheses in image coordinates. The results from these two components are fused at a later stage in the framework. In the target filtering component, the visual object estimates are further refined by modeling kinematics and detector noise. This component stabilizes the object identities and counters false detections. Image location estimates generated by the filtering component are then used to compute the distance and heading to the targets. This information is input to the leashing control module, which provides flight destinations to the UAV Control system.

3.1 Visual Object Detection

We implement two object detection methods, namely the HOG and Color-HOG based detectors. The former is the standard approach proposed by Dalal and Trigs [4]. It works by computing a feature representation using histogram of oriented gradients (HOG) on positive and negative samples of persons from the training set. A Support Vector Machine (SVM) classifier is then trained on these samples. Given a test image, the learned model is applied in a sliding window fashion to find potential detection responses. Our framework employs the HOG based classifier implemented in OpenCV.

As a second option, we use the Color-HOG detector proposed by Khan et al. [12]. The detector augments the standard HOG based method with color information. A late fusion scheme is employed to combine the color and shape information. We use color attributes [22] as an explicit color representation and fuse it with HOG features.

The visual object detector is implemented as a ROS node and runs in a separate thread. When the detections from a camera image are computed, the result is published. The detector then starts to process the latest available image from the camera.

3.2 Visual Object Tracking

We use the Adaptive Color Tracker (ACT) proposed recently by Danelljan et al. [5]. It has shown to achieve state-of-the-art performance on a large number of benchmark videos. The method is simple and computationally efficient, making it especially suitable for robotic applications. Here we use the ACT to track humans, but the method is generic and can be applied to track any visual object.

The ACT works by learning a discriminative classifier on the target appearance. Its low computational cost is primarily due to two properties possessed by this tracking method. First, it assumes a periodic extension of the local image patch, which allows the usage of the fast Fourier transform (FFT) for the heavy computations. Second, the tracker applies a dynamically adaptive dimensionality reduction technique to reduce the number of features while preserving the important characteristics of the target appearance.

To update the tracker model at some frame n, a template t_n of size $M \times N$ centred around the target is first extracted. Danelljan et al. [5] suggests using a pixel dense representation of color name features [22] augmented with the usual grayscale values. These features are preprocessed by a normalization procedure followed by a windowing operation. The resulting template t_n is used to compute the kernelized auto-correlation $a_n(x, y)$ for all cyclic shifts x and y (in pixels) along the first and second coordinate respectively.

$$a_n(x, y) = \kappa(\tau_{x,y} P_n t_n, P_n t_n) \tag{1}$$

Here, κ is a Gaussian radial basis function kernel and $\tau_{x,y}$ is the cyclic shift operator. The projection operator P_n, that is computed by the dimensionality reduction technique, maps the pixel features onto a low-dimensional linear subspace. The desired output score y_n (i.e. the classifier labels) is set to a $M \times N$ sampled Gaussian function with a centred peak. The numerator $\hat{\alpha}_n$ and denominator $\hat{\beta}_n$ of the Fourier transformed classifier coefficients are updated with the new sample template using:

$$\hat{\alpha}_n = (1 - \gamma)\hat{\alpha}_{n-1} + \gamma \hat{y}_n \hat{a}_n \tag{2a}$$

$$\hat{\beta}_n = (1 - \gamma)\hat{\beta}_{n-1} + \gamma \hat{a}_n (\hat{a}_n + \lambda) \tag{2b}$$

Here, γ denotes a scalar learning rate parameter and λ is a scalar regularization parameter. The multiplication between signals is point-wise and $\hat{f}$ denotes the discrete Fourier transform (DFT) of a signal f. The tracker model also includes a template appearance u_n, which is updated as:

$$u_n = (1 - \gamma)u_{n-1} + \gamma t_n. \tag{3}$$

The tracking model is applied to a new image at time step n to locate the object by first extracting a $M \times N$ sample template v_n. This is done at the predicted target location and the extraction procedure is the same as for t_n. The kernelized cross-correlation between the sample template and the learned template appearance is given by:

$$b_n(x, y) = \kappa(\tau_{x,y} P_{n-1} v_n, P_{n-1} u_n) \tag{4}$$

The confidence scores s_n over the patch v_n are then computed as a convolution in the Fourier domain.

$$s_n = \mathscr{F}^{-1} \left\{ \frac{\hat{\alpha}_{n-1} \hat{b}_n}{\hat{\beta}_{n-1}} \right\} \tag{5}$$

Here $\mathscr{F}^{-1}$ denotes the inverse DFT operator. Henriques et al. [10] showed that the kernelized correlations a_n and b_n can be computed efficiently using the FFT.

The feature projection operator P_n is represented by a matrix that projects the feature vector of each pixel in a template onto a linear subspace. This projection matrix is obtained through an adaptive Principal Component Analysis proposed by [5]. A symmetric matrix $L_n = (1 - \eta)K_{n-1} + \eta C_n$ is computed as

a linear combination between the feature covariance matrix C_n of the current template appearance u_n and a symmetric matrix K_n. Here, η is a scalar learning rate parameter. The matrix Q_n depends on the previously chosen projection matrices and is updated as $K_n = (1-\eta)K_{n-1} + \eta P_n^{\mathrm{T}} D_n P_n$ in each frame, where D_n is a diagonal weight matrix. This term ensures smoothness, which prevents the classifier coefficients to become outdated. The new projection matrix P_n is obtained by performing an eigenvalue decomposition on the matrix L_n and selecting the eigenvectors corresponding to the largest eigenvalues. This scheme for calculating the projection matrix minimizes a loss function formulated in [5], which regards both the current appearance and the set of previously selected feature spaces.

The visual tracking is implemented as a separate node in ROS in our framework. It processes all targets sequentially. All parameters of the ACT are set as suggested by the authors.

3.3 Combining Tracking and Detection

We use a separate component to fuse the tracking and detector results. It is implemented as a separate ROS node, and thus runs in a separate thread. When new tracking results are available for a visual object, the location and appearance model for this object is simply replaced with the ones returned by the tracker.

Person detections received by the detector component is used for the following puropses: to initialize new object candidates, verify existing object candidates, identify tracking failures and to correct the location and size of the visual object. A new object candidate is initialized when a detection is received that is not overlapping with any current objects or candidates. The image region that corresponds to this object candidate is then tracked until it is either verified or discarded. A candidate is verified if additional overlapping detections are received during the next-coming frames. If this occurs, the candidate is upgraded to a *known object*, otherwise it is discarded and removed.

To identify tracking failures, each known object must be verified with an overlapping detection within a certain number of frames. The object is identified as a tracking failure if no overlapping detection is received within the specified number of frames since the last verification. This leads to the removal of that object. To counter tracker drift, we also correct the target location and size with a partially overlapping detection if the overlap is less than a threshold.

3.4 Target Filtering

The target tracking module has three main parts: Bayesian estimation of

1. kinematics: velocity vectors are estimated for each target;
2. state uncertainty: full covariance matrices are estimated for each target state;
3. target ID: the visual tracking IDs are stabilized using the information contained in the estimated state vectors and covariance matrices.

The visual tracking output at time step k is a set $\mathbf{Z}_k = \{\mathbf{z}_k^{(j)}\}_{j=1}^{N_{z,k}}$, where each element $\mathbf{z}_k^{(j)} = (I_k^{(j)}, d_k^{(j)})$ consist of an ID $I_k^{(j)} \in \mathbb{N}$ a detection window $d_k^{(j)} \in \mathbb{R}^4$ that defines the position in the image and the windows width and height.

The purpose of the multiple object tracking (MOT) filter is to use the sets $\mathbf{Z}_k$ to estimate the object set $\mathbf{X}_k = \{\xi_k^{(i)}\}_{i=1}^{N_{x,k}}$, where both the number of objects $N_{x,k}$ and the object states $\xi_k^{(i)}$ are unknown. The object state at time step k is defined as $\xi_k^{(i)} = (\mathbf{x}_k^{(i)}, J_k^{(i)})$, where $J_k^{(i)}$ is the object's ID and $\mathbf{x}_k^{(i)}$ is the object state vector,

$$\mathbf{x}_k = \left[p_k^x \ , p_k^y \ , v_k^x \ , v_k^y \ , w_k \ , h_k\right]^{\mathrm{T}} \tag{6}$$

where $[p_k^x, p_k^y]$ is the position, $[v_k^x, v_k^y]$ is the velocity, and w_k and h_k is the width and height of the detection window.

The process and detection models are

$$\mathbf{x}_{k+1} = F_{k+1}\mathbf{x}_k + \mathbf{w}_{k+1} = \begin{bmatrix} \mathbf{I}_2 & T_s\mathbf{I}_2 & \mathbf{0}_2 \\ \mathbf{0}_2 & \mathbf{I}_2 & \mathbf{0}_2 \\ \mathbf{0}_2 & \mathbf{0}_2 & \mathbf{I}_2 \end{bmatrix} \mathbf{x}_k + \mathbf{w}_{k+1} \tag{7}$$

$$\mathbf{z}_k = H_k\mathbf{x}_k + \mathbf{e}_k = \begin{bmatrix} \mathbf{I}_2 & \mathbf{0}_2 & \mathbf{0}_2 \end{bmatrix} \mathbf{x}_k + \mathbf{e}_k, \tag{8}$$

where $\mathbf{w}_{k+1}$ and $\mathbf{e}_k$ are zero-mean Gaussian noice processes with covariance matrices Q_{k+1} and R_k, respectively.

The visual tracking output is used as input in a Probability Hypothesis Density (PHD) filter [16, 17]. Specifically we use a Gaussian mixture implementation [21] with a uniform distribution for the position component of the birth PHD intensity [3].

3.5 Distance Estimation

Controlling the UAV by leashing requires a distance estimate to the target. This is obtained by assuming a horizontal ground plane and a fixed person height h. Figure 3 contains a simple illustration of the scenario. The angle φ between the optical axis and the projection ray of the top of the target is calculated as

$$\varphi = \arctan\left(\frac{y}{f}\right) \tag{9}$$

where y is the normalized image top-coordinate of the bounding box and f is the effective focal length. Using the known altitude z and camera pitch angle ρ, the distance d can be obtained from simple trigonometry.

$$d = \frac{z - h}{\tan(\rho - \varphi)} \tag{10}$$

Since we have a camera with a narrow field of view, a small yaw angle of the camera relative the target can be assumed. We therefore approximate the effective focal length to $f = 1$ m. We also assume that the UAV is flying approximately upright when extracting the top coordinate of the target. However, these assumptions have minimal impact due to other dominant model errors and measurement inaccuracies.

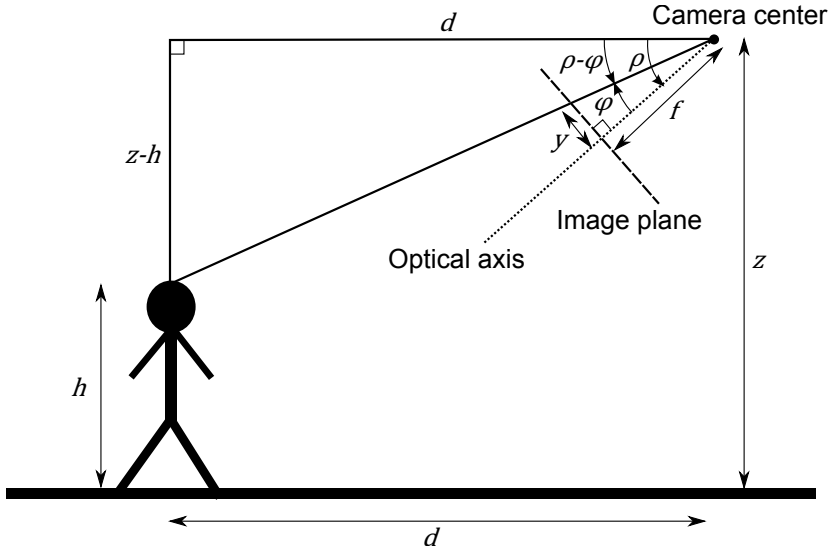


Fig. 3: To estimate the ground distance d between the UAV and the person, we assume a horizontal ground plane and a person height h. The angle φ is obtained from normalized image coordinate y of the upper edge of the bounding box and the effective focal length f. The UAV altitude z and camera pitch ρ are known. The distance is obtained using simple geometry by considering the larger triangle in the figure.

3.6 Leashing Control Module

The task of the *Leashing Control* module is to keep specified distance between the UAV and the target, and point the UAV towards the target. To achieve it the *Leashing Control* outputs the flight target state vector i.e. $[v_t^x, v_t^y, v_t^z, \psi_t]$, where $[v_t^x, v_t^y, v_t^z]$ are the target velocities, and ψ_t is the target heading.

The target state vector is used by the UAV velocity controller which in turn calculates target angle values used by the attitude stabilisation inner control loops. Flight target velocities i.e. $[v_t^x, v_t^y, v_t^z]$ are calculated in the following way. First, a new flight target position is calculated on a line which includes the current UAV position and the target position at a specified distance from the target. Then, the target velocities are proportional to the difference between the current UAV position and the new flight target's position.

Flight target heading ψ_t is calculated based on the UAV and target positions in the world coordinate frame.

Additionally the *Leashing Control* module implements a strategy for finding a target. This is done by commanding a sweeping motion using the heading target in order to increase chances of reacquiring a lost target or finding one in case it has not yet been found.

	CT	TLD	EDFT	Struck	LSHT	ACT
basketball	171	*65.2*	108	159	156	**9.29**
bolt	371	*88*	355	391	122	**4.2**
boy	32.1	4.09	**2.34**	*3.35*	32.6	4.39
couple	77.8	*64.3*	89.4	**12.7**	114	123
david	14.3	34.3	*9.2*	43.2	14.8	**7.73**
david3	68.5	136	**6.46**	107	53.7	*9.11*
human	428	110	*5.77*	**5.36**	6.59	7.25
singer1	15	*10.6*	16.6	12.4	21	**9.21**
skating1	184	104	199	*82.3*	82.3	**7.95**
trellis	51.1	55.9	59.6	**15.3**	61.2	*20.8*
walking	214	110	*5.77*	**5.36**	6.59	7.25
walking2	64.6	63.1	*28.7*	**12.9**	50.6	47.7
Average CLE	141	70.5	73.8	70.8	*60.1*	**21.5**

(a) Center location error.

	CT	TLD	EDFT	Struck	LSHT	ACT
basketball	4.14	*51.3*	30.5	11.6	5.1	**99.9**
bolt	2.57	32	2.57	2.86	*37.4*	**100**
boy	66.6	**100**	**100**	**100**	56.3	*99.8*
couple	30.7	*31.4*	21.4	**83.6**	10.7	10.7
david	*79.2*	65.8	**100**	32.7	76	**100**
david3	43.3	35.3	**100**	33.7	75	*90.5*
human	0.485	*42.2*	**100**	**100**	**100**	**100**
singer1	86.9	**100**	49.3	*98.3*	40.2	95.7
skating1	8.5	27	16.3	51	*56*	**100**
trellis	20.9	44.5	47.6	**73.5**	44.8	*68.9*
walking	12.1	*42.2*	**100**	**100**	**100**	**100**
walking2	40	37.6	40.2	**85.6**	39.8	*42.4*
Average DP	33	50.8	59	*64.4*	53.4	**84**
Average FPS	*62*	31.6	19	11.2	11.3	**106**

(b) Distance precision.

Table 1: The results of the visual tracking evaluation. Six methods are compared on 12 benchmark sequences using center location error (a) and distance precision (b). The average frame-rate is also included in table (b). The best and second best results are shown in red and blue respectively.

4 Experimental Evaluation

In this section we provide our experimental results. First, we evaluate the employed visual tracking method on benchmark videos and compare it to other state-of-the-art real-time visual tracking methods. Second, we show the impact of the target filtering component in our framework. Finally, we provide some fight test results.

4.1 Visual Tracking

We compare the ACT [5] with five other recent tracking methods from the literature with real-time performance, namely CT [26], TLD [11], EDFT [6], Struck [8] and LSHT [9]. We use the code and the suggested parameter settings provided by the respective authors. The recent evaluation protocol provided by Wu et al. [23] is used.[2] From their dataset of 50 videos, we select the 12 videos of human or face tracking, where the setting is most similar to our application. The performance is measured in center location error (CLE) and distance precision (DP). CLE is defined as the average distance between the centroids of the tracked and ground truth bounding boxes, over the sequence. Distance precision is the relative number of frames in a sequence for which the distance between the centroids is less than 20 pixels. We also compare the frame rates of the different approaches. The experiments were performed on an Intel Xenon 2 core 2.66 GHz CPU with 16 GB RAM.

The results are shown in table 1. ACT and Struck perform best on the same largest number of videos (five for CLE and six for DP). However, ACT obtains

[2] The evaluation code and benchmark image sequences are available at https://sites.google.com/site/trackerbenchmark/benchmarks/v10

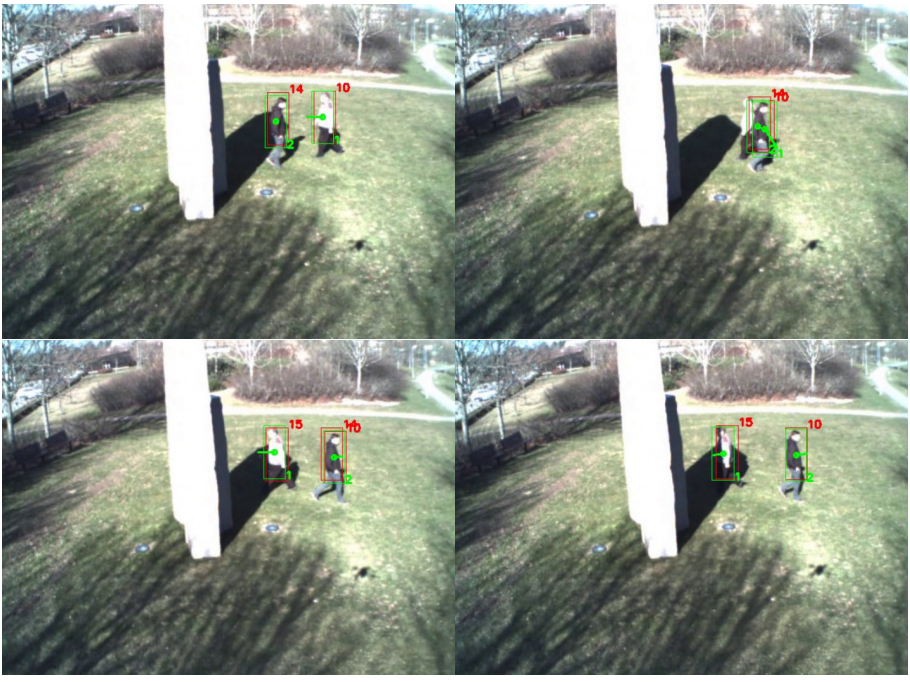


Fig. 4: Example multiple object tracking results with two targets: one heading left and one heading right in the image. Shown are bounding boxes with IDs: visual tracking is indicated by red, multiple object tracking is indicated by green. During the crossing the targets overlap significantly, however the multiple target tracking maintains correct target IDs despite the visual tracking is associating erroneous IDs.

significantly better average performance, where it improves 23.6% in average distance precision over Struck. Moreover, the ACT is the fastest tracker in our evaluation, with almost ten times higher frame rate than Struck.

4.2 Target Filtering

The benefits of the target tracking module are most apparent in terms of stabilizing the tracking IDs. In ambiguous situations, such as when two or more targets pass each other in the image, the visual tracking IDs may become mixed up. Here the target tracking module uses the additional information contained in the kinematics and uncertainty estimates to maintain the target IDs correctly.

An example with two targets is given in Fig. 4. In the top left the two targets are approaching each other, with visual tracking/target IDs 14/2 (to the left, heading right) and 10/1 (to the right, heading left). During the crossing, shown in top right, the two targets overlap significantly. After the crossing, shown bottom left, both Visual tracker 14 and Visual tracker 10 have stuck to the

target heading right in the image, and for the target heading left in the image a new Visual tracker with ID 15 has been initialized. However, the target tracking module has correctly fused the Visual tracking information with the estimated kinematics and uncertainty information, and the target IDs are correct. Shortly thereafter, shown bottom right, the Visual tracking has correctly deleted Visual tracker with ID 14.

4.3 System Evaluation

The presented system has been evaluated in a number of autonomous flights during which a micro UAV was following a person. A short description of a platform used for the flight tests is presented. Below follows a description of the experimental setup and the results of the flight tests.

Experimental setup The presented system has been evaluated in a number of autonomous flights performed in a lab equipped with a motion capture system manufactured by Vicon Motion Systems company[3]. The system captures positions in a volume of $11 \times 11 \times 5$ meters and it was used both to provide a reference for the vision system performance evaluation and the state of the UAV used for the control (i.e. position and velocity in x, y, and z axes and heading). The state used for control was sent wirelessly and with an update rate of 10 Hz.

Figure 2 presents the interconnections and placement of the system components during the experimental flights.

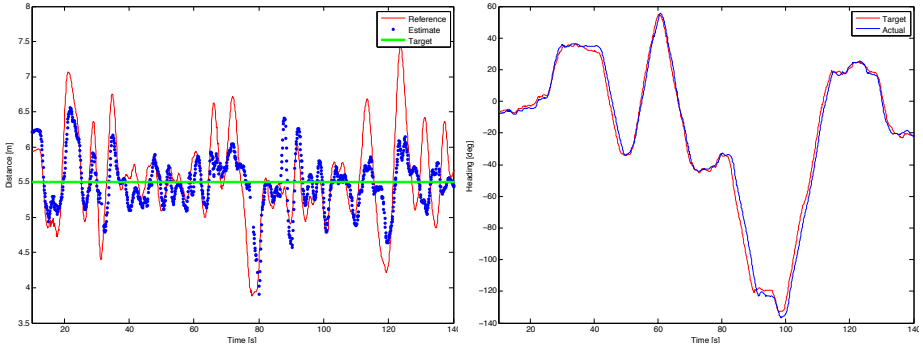
Flight test results During several flight tests the performance of the whole system was evaluated in terms of the accuracy of the vision-based distance estimation and the tracking performance of the control system.

Figure 5a presents the distance between the target and the UAV platform during a flight. The distance to keep was set to 5.5 meters and is depicted with the green line. The blue dotted curve shows the distance estimated based on vision as described in section 3. The red curve is the distance calculated using the Vicon reference system. As can be seen, during the flight the distance was kept with maximum error of 1.6 meters from the estimate and 1.9 meters from the reference.

Figure 5b presents the target heading ψ_t as described in section 3.6 along with the actual heading during the leashing experiment. In case of slow changes in the target heading, our framework accurately tracks the target. For faster changes, the heading error increases due to the limited maximum allowed heading rate of the UAV. It is worthy to mention that it has little impact on the overall leashing performance.

In summary, our framework is able to perform the leashing control task using the active vision component described earlier. The accuracy of the distance estimation can be improved further by taking into account the size of the detection

[3] http://www.vicon.com/



(a) Distance estimation. (b) Heading control.

Fig. 5: The estimated relative distance of the target relative to UAV (a), target and actual heading (b) during the leashing experiment.

box. However, our results suggest that for the leashing task the simple distance estimation approach provides consistent results on several indoor scenarios. For most leashing applications, a small bias in the distance estimate is tolerable since the purpose of our framework is to follow a person at a roughly constant distance.

5 Conclusion

In this paper, we present a low-level active vision framework for unmanned aircraft systems. Our framework is implemented on the LinkQuad platform and employs state-of-the-art object detection, object tracking, Bayesian filtering and AI-based methods. It efficiently detects and tracks persons in real-time which is used for virtual leashing. Future work involves recognizing human actions such as hand waiving, clapping etc. for advanced virtual leashing scenarios. Another potential research direction is to integrate efficient person re-identification techniques to encounter heavy occlusions and out-of-view scenarios.

References

1. Bao, C., Wu, Y., Ling, H., Ji, H.: Real time robust l1 tracker using accelerated proximal gradient approach. In: CVPR (2012)
2. Bar-Shalom, Y., Willett, P.K., Tian, X.: Tracking and data fusion, a handbook of algorithms. YBS (2011)
3. Beard, M., Vo, B., Vo, B.N., Arulampalam, S.: A partially uniform target birth model for Gaussian mixture PHD/CPHD filtering. IEEE Transactions on Aerospace and Electronic Systems 49(4), 2835–2844 (Oct 2013)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR (2005)

5. Danelljan, M., Shahbaz Khan, F., Felsberg, M., van de Weijer, J.: Adaptive color attributes for real-time visual tracking. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2014)
6. Felsberg, M.: Enhanced distribution field tracking using channel representations. In: ICCV Workshop (2013)
7. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.A., Ramanan, D.: Object detection with discriminatively trained part-based models. PAMI 32(9), 1627–1645 (2010)
8. Hare, S., Saffari, A., Torr, P.: Struck: Structured output tracking with kernels. In: ICCV (2011)
9. He, S., Yang, Q., Lau, R., Wang, J., Yang, M.H.: Visual tracking via locality sensitive histograms. In: CVPR (2013)
10. Henriques, J., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: ECCV (2012)
11. Kalal, Z., Matas, J., Mikolajczyk, K.: P-n learning: Bootstrapping binary classifiers by structural constraints. In: CVPR (2010)
12. Khan, F.S., Anwer, R.M., van de Weijer, J., Bagdanov, A., Vanrell, M., Lopez, A.: Color attributes for object detection. In: CVPR (2012)
13. Kwon, J., Lee, K.M.: Tracking by sampling trackers. In: ICCV (2011)
14. Liu, B., Huang, J., Yang, L., Kulikowski, C.: Robust tracking using local sparse appearance model and k-selection. In: CVPR (2011)
15. Lowe, D.G.: Distinctive image features from scale-invariant points. IJCV 60(2), 91–110 (2004)
16. Mahler, R.: Multitarget Bayes filtering via first-order multi target moments. IEEE Transactions on Aerospace and Electronic Systems 39(4), 1152–1178 (Oct 2003)
17. Mahler, R.: Statistical Multisource-Multitarget Information Fusion. Artech House, Norwood, MA, USA (2007)
18. Meier, L., Tanskanen, P., Fraundorfer, F., Pollefeys, M.: Pixhawk: A system for autonomous flight using onboard computer vision. In: ICRA (2011)
19. van de Sande, K., Uijlings, J.R.R., Gevers, T., Smeulders, A.W.M.: Segmentation as selective search for object recognition. In: ICCV (2011)
20. Sinopoli, B., Micheli, M., Donato, G., Koo, T.J.: Vision based navigation for an unmanned aerial vehicle. In: ICRA (2001)
21. Vo, B.N., Ma, W.K.: The Gaussian mixture probability hypothesis density filter. IEEE Transactions on Signal Processing 54(11), 4091–4104 (Nov 2006)
22. van de Weijer, J., Schmid, C., Verbeek, J.J., Larlus, D.: Learning color names for real-world applications. TIP 18(7), 1512–1524 (2009)
23. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: CVPR (2013)
24. Yu, Z., Nonami, K., Shin, J., Celestino, D.: 3d vision based landing control of a small scale autonomous helicopter. International Journal of Advanced Robotic Systems 4(1), 51–56 (2007)
25. Zhang, J., Huang, K., Yu, Y., Tan, T.: Boosted local structured hog-lbp for object localization. In: CVPR (2010)
26. Zhang, K., Zhang, L., Yang, M.: Real-time compressive tracking. In: ECCV (2012)