https://doi.org/10.2352/ISSN.2470-1173.2020.16.AVM-200

A tool for semi-automatic ground truth annotation of traffic videos

Florian Groh, Dominik Schörkhuber, Margrit Gelautz; Institute of Visual Computing and Human Centered Technology, TU Wien; Vienna, Austria

Abstract

We have developed a semi-automatic annotation tool – "CVL Annotator" – for bounding box ground truth generation in videos. Our research is particularly motivated by the need for reference annotations of challenging nighttime traffic scenes with highly dynamic lighting conditions due to reflections, headlights and halos from oncoming traffic. Our tool incorporates a suite of different state-of-the-art tracking algorithms in order to minimize the amount of human input necessary to generate high-quality ground truth data. We focus our user interface on the premise of minimizing user interaction and visualizing all information relevant to the user at a glance. We perform a preliminary user study to measure the amount of time and clicks necessary to produce ground truth annotations of video traffic scenes and evaluate the accuracy of the final annotation results.

Introduction

In the context of computer vision and machine learning algorithms for assisted/autonomous driving, the need for training and evaluation data in the automotive industry is increasing significantly. The goal is to ultimately deploy autonomous vehicles into traffic that is subject to unpredictable environmental influences, such as changing weather and lighting conditions. Various scientific groups and companies have created and published road scene ground truth datasets (e.g. Argoverse [2], CityScapes [3], BDD100K [4], KITTI [7], CamVid [5], D^2-City [6], VIPER [8]) to further research on autonomous vehicles and machine learning.

The work presented in this paper is embedded in the CarVisionLight (CVL) project, which aims to develop an object detection algorithm for night scenes with temporal consistency (see also [1]). To achieve this goal with a supervised machine learning algorithm, we defined the following requirements, which a training dataset should ideally meet:

- non-urban roads (e.g. highway or country roads)
- nighttime
- at least 20FPS temporal density
- realistic lighting in a real-world environment.

Table 1 gives an overview of various datasets (both road scenes as well as general scenes) we have reviewed. Several datasets (e.g. GOT10k [18], VOT2017 [19], VIPER [8], D^2-City [6], BDD100K [4]) have the temporal density needed for our application. While temporally dense night scenes are included in some of them ([2], [6]), we noticed a shortage of footage from non-urban roads. In the case of snythetically generated videos, such as in ([8]), we observed a lack of natural lighting variability (high dynamic contrast, glaring, halos or reflections).

Dataset	**Non-Urban**	**Night**	**$\geq$ 20 FPS**	**Real**
Argoverse [2]	✗	✓	✓	✓
BDD100k [4]	✓	✓	✗	✓
CamVid [5]	✗	✗	✗	✓
CityScapes [3]	✗	✗	✗	✓
D^2-City [6]	✗	✓	✓	✓
KITTI [7]	✓	✗	✓	✓
VIPER [8]	✓	✓	✓	✗
GOT10k [18]	✗	✗	✓	✓
VOT2017 [19]	✗	✗	✓	✓

Table 1: Overview of datasets regarding selected requirements.

From our investigation of these existing datasets we concluded that currently there is no publicly available ground truth dataset which fully meets our requirements. This motivated the development of an annotation tool that supports the efficient ground truth generation for self-recorded nighttime traffic scenes. The further parts of this paper are organized as follows. The *Related Work* section outlines publicly available video annotation tools. In the subsequent *Method* section, we review the design process of our newly developed CVL Annotator (CVLA) tool including considerations on tracker selection and user interface (UI) design. In the *Results* section, we present the findings of our preliminary user study, comparing CVLA to the Scalabel annotation tool [4] regarding annotation accuracy and time. Finally, in the last section of this paper, we discuss these results and suggest possible future work.

Related Work

In order to increase the speed of video data ground truth annotation, the scientific community has already put a lot of effort developing tools and algorithms to help in this matter. Notable video annotation tools include VATIC [9], ViTBAT [12], CVAT [10], Scalabel [4] or BeaverDam [20]. In this section, we take a look at the platform, user interface and data propagation choices of these tools.

Platform

With the exception of ViTBAT, the aforementioned tools work as web applications with the browser acting as the user facing front end and a web server acting as the back end, keeping track of all of the data. The focus on web technologies is primarily rooted in the fact that annotation tasks can then be accessed through a simple URL and can therefore be included into crowdsourcing services such as Mechanical Turk. ViTBAT, on the other hand, chose to offer a tool that runs locally on the annotator's machine, removing the latency associated with network connections and thus theoretically enabling higher interactive speeds.

User Interface

Regarding the user interfaces of video annotation tools, we found that Shen has done an excellent analysis in his thesis for the BeaverDam tool [20]. Here are his main findings, which we mostly incorporated into CVLA (see *Method* section):

- **Keyframe visibility**:
  displaying a keyframe icon increases awareness, requires less guesswork, and therefore increases annotation speed
- **Fast playback**
  caching the whole video in advance eliminates server timeouts on frame changes
- **Click reduction**
  drawing new objects without the need to click "new object", and the object type is pre-selected as the most common class ("car") or the previous selection
- **Frame exit/enter**
  being able to drag bounding boxes outside of the frame increases speed, as annotators do not have to align their mouse perfectly with the image border

Data Propagation

Regarding data propagation, we found that the existing video annotation tools all offer linear interpolation between keyframes. This can be very helpful when dealing with footage from a stationary camera. But when the camera itself is moving, the amount of keyframes needed to follow objects in screen space greatly increases due to abrupt movements in the camera path (e.g. road bumps, sharp turns).

Method

In this section, we explain our way of evaluating which trackers to include into CVLA as well as the user interface choices we made. Additionally, we present the design of the preliminary user study we performed to assess possible improvements of CVLA compared to Scalabel [4] on metrics such as (i) annotation time per bounding box, and (ii) annotation data accuracy.

As opposed to existing annotation tools reviewed in the *Related Work* section, which contain only linear interpolation as their propagation method, the incorporation and evaluation of state-of-the-art object tracking algorithms is a major component of our work. We have performed an extensive test of different state-of-the-art algorithms on synthetic night scenes of the VIPER [8] dataset to determine which tracking algorithms are the most promising candidates for nighttime footage.

User Interface

We opted to develop CVLA in the programming language Python, as we have found that the research community releases most of the state-of-the-art trackers in Python. Secondly, we chose to build an application running directly on the annotator's machine because we wanted to achieve fast responses times without network latency. Figure 1 shows a screenshot of CVLA's user interface. Our tool is aiming to reduce user interaction while giving a clear overview of the annotation data in a timeline view (bottom half of Figure 1) as proposed by Shen [20]. As in Shen's work, we reduce mouse clicks by not forcing the user select the type of object they are annotating every time after drawing a bounding box, but instead automatically assign the same type as the previously annotated object. The same applies for selecting the current tracker for the object. The red and green lines on top of the object tracks in the timeline view indicate, whether the bounding box has already been propagated by the selected algorithm or not. This propagation can either be done in the background by enabling "Automatic Tracking" (top right of Figure 1), or when going through the video on a frame-by-frame basis. On the top right of Figure 1, one can also see our gamma correction slider, which can help improve visibility when trying to identify object boundaries in dark scenes.

Tracker Selection

To assess which trackers to include in our annotation tool, we performed an extensive test of five different trackers on night scenes of the VIPER dataset. The trackers we chose to evaluate are: ATOM [13], SiamRPN [14], MedianFlow [15], KCF [16] and CSRT [17]. The first two were chosen because of their good results in the VOT challenge [19], whereas the last three were chosen for their fast update times. Out of all the night scenes in the VIPER dataset, we included all object tracks, where the bounding box has a minimum area of 30 pixels over at least 10 frames, and where the bounding box area difference between consecutive frames was at most 20%. In total, we evaluated the trackers on 3159 different object tracks. To measure the quality of the trackers we chose two weakly correlated measures: *Mean IoU* ($\hat{\phi}$, equation 1) and *reset rate* ($\hat{r}$, equation 3) as described by Kristan et al. [19]. The IoU measure (ϕ_t) is described in equation 2, where R_t^G denotes the ground truth region at time t, and R_t^T is the tracker's proposed region. Figure 2 shows a visual explanation of this.

$$\hat{\phi} = \sum_t \frac{\phi_t}{N} \tag{1}$$

$$\phi_t = \frac{R_t^G \cap R_t^T}{R_t^G \cup R_t^T} \tag{2}$$

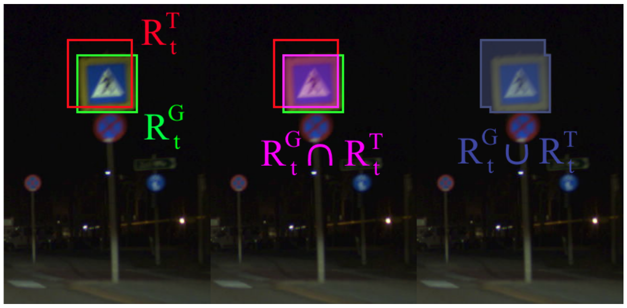


Figure 2. Visual explanation of IoU measure. Leftmost image shows ground truth (R_t^G, green) and tracker's proposed region (R_t^T, red), middle image shows region intersection ($R_t^G \cap R_t^T$, magenta), and rightmost image shows union ($R_t^G \cup R_t^T$, blue).

Reset rate ($\hat{r}$) describes the amount of frames where the IoU went below a threshold (τ) and had to be reset, divided by the total

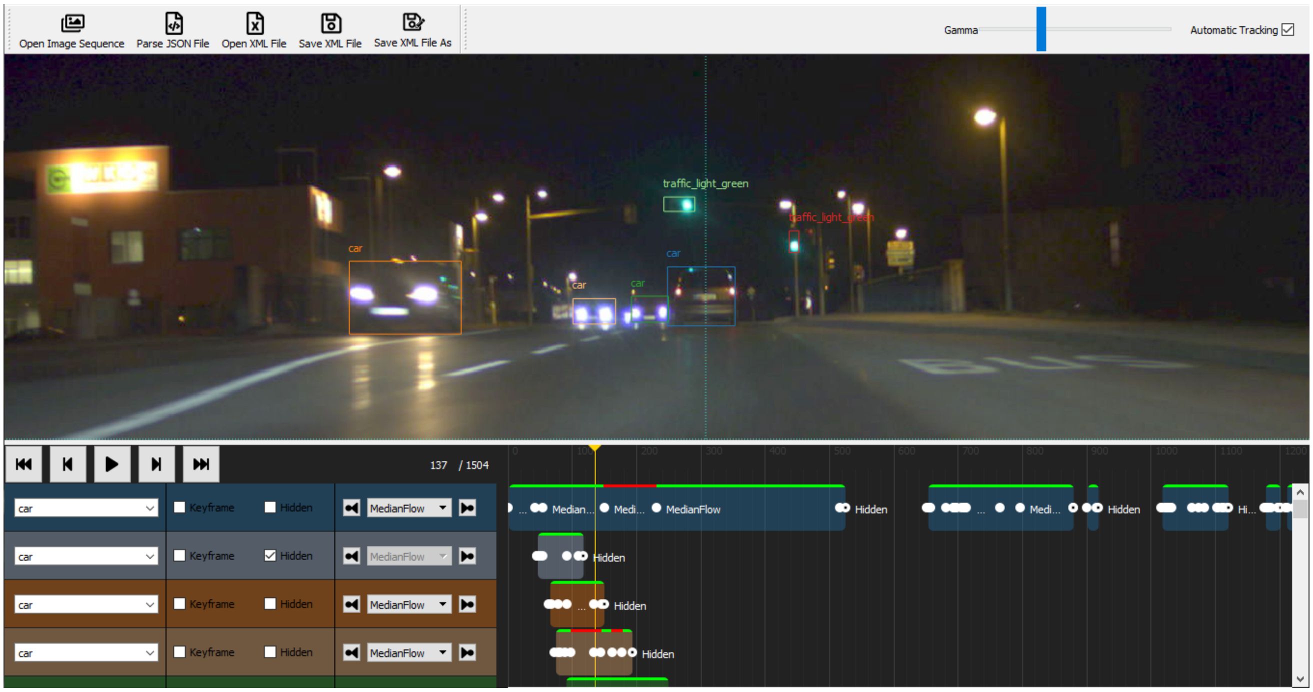


Figure 1. *Screenshot of CVLA tool, showing gamma correction slider, video area, and data overview/timeline area.*

number of frames N. See figure 3 for an example of an IoU / time graph, with two resets. The reset threshold we chose was 51%, and in order to have a fair calculation of the mean IoU for each tracker, we only took tracked frames and no initialization frames – which would have an IoU of 100% – into account.

$$\hat{r} = \frac{||\{t|\phi_t < \tau\}_{t=1}^{N}||}{N} \tag{3}$$

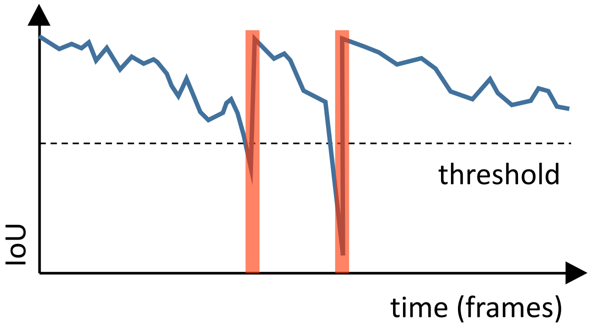


Figure 3. *Plot of IoU over time, with two tracker resets – where IoU falls below threshold – shown in red.*

Preliminary User Study Design

To compare CVLA against Scalabel, we performed a preliminary user study with two annotators, annotating 12 videos with 3349 frames and 150 object tracks. We again used the VIPER dataset as our ground truth data and chose nighttime scenes on non-urban roads. As stated above, we were focused on comparing the annotation process with regards to time, keystrokes, mouse movement, clicks and annotation accuracy; the latter is represented by mean IoU. In order to have a fair comparison of these values we had to make sure that the annotators were focusing on the same 150 object tracks regardless of the tool used. This was accomplished by displaying a visual anchor (ground truth downsized to 40% of actual size) over the objects of interest (see Figure 4). We expect this overlay to introduce a bias towards more accurate annotations and higher IoU values. However as the overlay was shown in both annotation tools, we do not expect this bias to affect the relative differences in comparing the annotation accuracy of both tools.

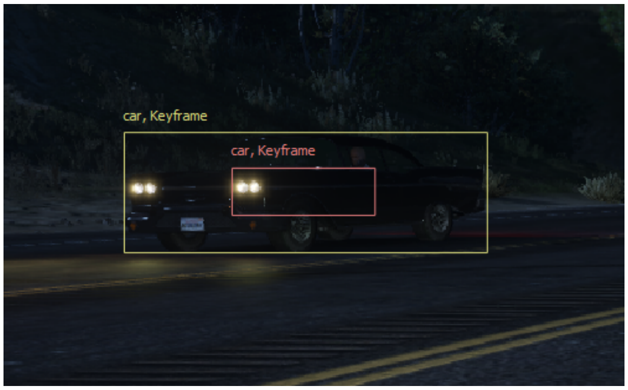


Figure 4. *Visual anchor in red, user annotation in yellow.*

Since we focused on relative improvements between tools, we made sure that the individual annotators worked on the same test sequences in both tools (i.e. annotator A did sequence 1-4 and annotator B did sequence 5-12 in both tools). To track annotation times and clicks, we used a mouse tracking application called “Mousotron” [11], which enabled us to keep score of the number of clicks, keystrokes, travelled mouse distance, and scroll wheel invocations.

Results

In this section, we present our findings regarding tracker evaluation as well as the results of our preliminary user study.

Tracker Evaluation

Our evaluations on night scenes from the VIPER dataset suggest that current state-of-the-art trackers (e.g., ATOM [13], SiamRPN [14]) are not necessarily more suitable for our application than the classic Medianflow [15] approach. Table 2 shows the results of this evaluation. ATOM [13] was the best performing tracker with a mean IoU of 72.6% and a reset rate of 7.3%. However, Medianflow [15] performed nearly as well, with 71.7% mean IoU and 7.2% reset rate while being much faster at 15.4ms compared to 88.2ms.

Tracker	Mean IoU	Reset rate	Mean Time
KCF [16]	54.8%	23.5%	17.1ms
SiamRPN [14]	59.4%	20.5%	364ms
CSRT [17]	70.8%	10.4%	59.5ms
Medianflow [15]	71.7%	**7.2%**	**15.4ms**
ATOM [13]	**72.6%**	7.3%	88.2ms

Table 2: Tracker Evaluation results, best performing values per column shown in bold.

Preliminary User Study Evaluation

Our evaluation of the annotation time and accuracy suggests that using CVLA for video annotation results in faster annotation speeds as well as more accurate data. Table 3 contains a summary of our evaluation. The total time needed to annotate the 3349 chosen frames in two different tools divided between two annotators was 18 hours and 41 minutes. 6 hours and 55 minutes were spent in CVLA, and 11 hours and 46 minutes in Scalabel, thus resulting in a speed increase of about 1.69. Additionally, using our tool, the mean IoU increased by about 1.06. Mouse (click, scroll) and keyboard invocations could be significantly reduced (2.28), whereas the distance the mouse moved over the screen was only decreased by a factor of 1.04.

	Time	IoU	Invocations	Dist.
Scalabel [4]	11h 46m	78.74%	63965	1.27km
CVL [Ours]	6h 55m	83.46%	28001	1.22km
Improvement factor	**1.69**	**1.06**	**2.28**	**1.04**

Table 3: Preliminary User Study results. CVLA performs better in all measured categories.

Figures 5 and 6 show the measured values per video sequence and tool, while also indicating which values refer to which annotator. It can be seen that there are no inherent differences between the two annotators, and that the improvements rather vary depending on the underlying video data. In Figure 5 we show the average annotation time per bounding box, where we count each bounding box per frame separately (e.g. 3 objects of interest each visible on 5 frames result in 15 bounding boxes). This graph shows that CVLA had a shorter time per box on all but one video of our test set with speedup factors ranging from 0.9 to 2.6 (video 7 and 12 respectively).

Mouse and Keyboard invocations can be seen in Figure 5 (bottom). We observed that invocations varied a bit less when using CVLA compared to Scalabel (relative standard deviation of 31% vs. 43%), while CVLA always needed far fewer invocations (improvement factor between 1.3 and 3.9). This consistent improvement regarding mouse and keyboard invocations can most likely be explained by the fact that we keep zoom and pan information across frames, whereas Scalabel loses this information on frame changes. It resets the zoom and shows the whole frame resized to the dimensions of the view port.

Mean IoU per video is shown in Figure 6. Here we see more consistent improvements, ranging from 1.02 to 1.10. The main contribution of this consistent IoU increase can be attributed to bounding boxes with relatively small areas (up to 400 pixels) as shown in Figure 7, where the mean IoU is 72.16% in CVLA vs. 57.68% in Scalabel. There are two likely explanations for this rather high increase for small bounding boxes: (i) unlimited zoom in CVLA and (ii) fairly consistent visual appearance for smaller objects which means that they are easier to track.

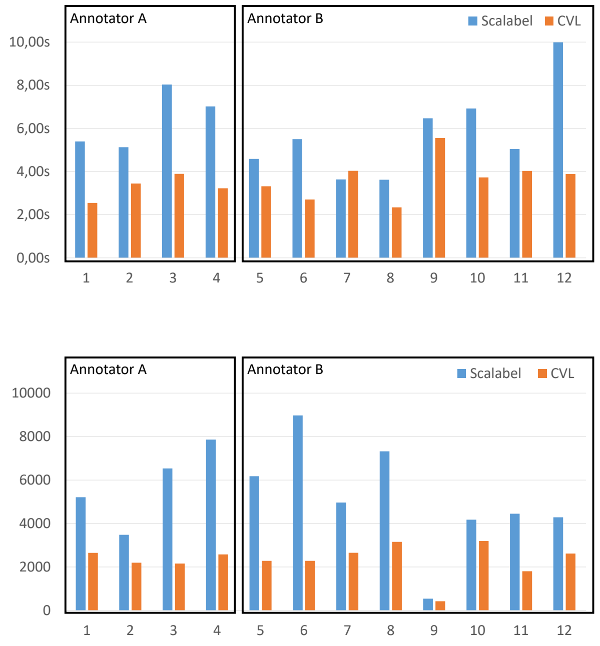


Figure 5. *Annotation time per bounding box per video and annotation tool (top). Mouse and keyboard invocation count (summed up) per video and annotation tool (bottom).*

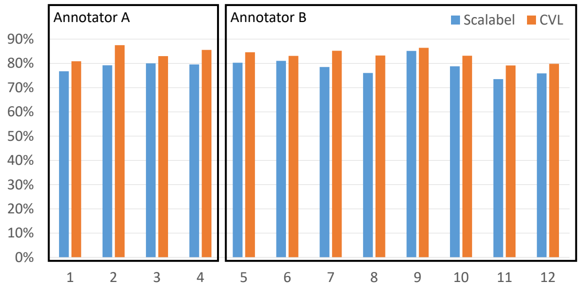


Figure 6. *Mean IoU value comparison per video and annotation tool.*

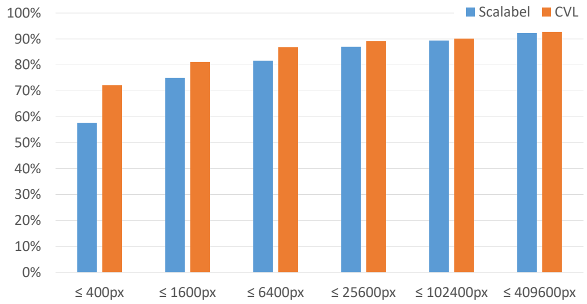


Figure 7. *Mean IoU value comparison grouped by maximum pixel area.*

Discussion and Future Work

We have proposed a semi-automatic video annotation tool (CVLA) with a focus on nighttime traffic scenes. Our tool includes state-of-the-art tracking algorithms, which we selected based on an analysis on the VIPER dataset. Furthermore, it features a user interface that focuses on minimizing the number of clicks and keystrokes needed to annotate video data. We have conducted a preliminary user study based on two users, which has shown promising results regarding the speed and accuracy increase of CVLA – using tracking algorithms – compared to an existing tool (Scalabel [4]) – using linear interpolation as its data propagation mechanism. On average, the annotations created with our tool have been 1.06 times more accurate in terms of mean IoU value, while taking 1.69 times less time to create. The average number of mouse and keyboard invocations was reduced by a factor of 2.28. To confirm these results with a more representative group of annotators, we plan to perform a larger user study with at least 10 participants with varying levels of experience in annotating videos.

Acknowledgments

This work was supported by the project CarVisionLight (project no. 861251), which is funded by the Austrian Research Promotion Agency (FFG) and the Austrian Federal Ministry BMVIT under the program “ICT of the Future”.

References

[1] Unger A., Gelautz M., Seitner F., Hödlmoser M. “A Study on Training Data Selection for Object Detection in Nighttime Traffic Scenes”, Proc. Electronic Imaging, (2020).

[2] Chang M. F., Lambert J., Sangkloy P., Singh J., Bak S., Hartnett A., Wang D., Carr P., Lucey S., Ramanan D., Hays J., “Argoverse: 3D Tracking and Forecasting with Rich Maps”, Proc. CVPR, pp. 8748-8757. (2019).

[3] Cordts M., Omran M., Ramos S., Rehfeld T., Enzweiler M., Benenson R., Franke U., Roth S., Schiele B. “The Cityscapes Dataset for Semantic Urban Scene Understanding”, Proc. CVPR, pp. 3213-3223. (2016).

[4] Yu F., Xian W., Chen Y., Liu F., Liao M., Madhavan V., Darrell T., “BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling“ arXiv preprint, 1805.04687 [cs.CV] (2018).

[5] Brostow G. J., Shotton J., Fauqueur J., Cipolla R., “Segmentation and Recognition Using Structure from Motion Point Clouds”, Proc. ECCV, pp. 44-57. (2008).

[6] Che Z., Li G., Li T., Jiang B., Shi X., Zhang X., Lu Y., Wu G., Liu Y., Ye J., “D^2-City: A Large-Scale Dashcam Video Dataset of Diverse Traffic Scenarios.”, arXiv preprint, 1904.01975 [cs.LG] (2019).

[7] Geiger A., Lenz P., Stiller C., Urtasun R., “Vision Meets Robotics: The KITTI Dataset”, The Intl. J. of Robotics Research, 32(11), pp. 1231-1237 (2013).

[8] Richter S. R., Hayder Z., Koltun V., “Playing for Benchmarks”, Proc. ICCV, pp. 2213-2222. (2017).

[9] Vondrick C., Patterson D., Ramanan, D., “Efficiently Scaling up Crowdsourced Video Annotation”, Intl. J. of Computer Vision, 101(1), pp. 184-204 (2013).

[10] Sekavech B., Manovic N., “Computer Vision Annotation Tool”, https://github.com/opencv/cvat/

[11] Blacksun Software, “Mousotron”, http://www.blacksunsoftware.com/mousotron.html

[12] Biresaw T. A., Nawaz T., Ferryman J., Dell A. I., “ViTBAT: Video Tracking and Behavior Annotation Tool”, Proc. AVSS, pp. 295-301. (2016).

[13] Danelljan M., Bhat G., Khan F. S., Felsberg M., “ATOM: Accurate Tracking by Overlap Maximization”, Proc. CVPR, pp. 4660-4669. (2019).

[14] Li B., Yan J., Wu W., Zhu Z., Hu X., “High Performance Visual Tracking with Siamese Region Proposal Network”, Proc. CVPR, pp. 8971-8980. (2018).

[15] Kalal Z., Mikolajczyk K., Matas J., “Forward-Backward Error: Automatic Detection of Tracking Failures”, Proc. ICPR, pp. 2756-2759. (2010).

[16] Henriques J. F., Caseiro R., Martins P., Batista, J., “High-Speed Tracking with Kernelized Correlation Filters”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 37(3), pp. 583-596 (2014).

[17] Lukezic A., Vojir T., Čehovin Zajc L., Matas J., Kristan M., “Discriminative Correlation Filter with Channel and Spatial Reliability”, Proc. CVPR, pp. 6309-6318. (2017).

[18] Huang L., Zhao X., Huang K., “GOT-10k: A Large High-Diversity Benchmark for Generic Object Tracking in the Wild”, IEEE Transactions on Pattern Analysis and Machine Intelligence. (2019).

[19] Kristan M., Matas J., Leonardis A., Vojir T., Pflugfelder R., Fernandez G., Nebehay G., Porikli F. Čehovin L., “A Novel Performance Evaluation Methodology for Single-Target Trackers”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(11), pp. 2137-2155 (2016).

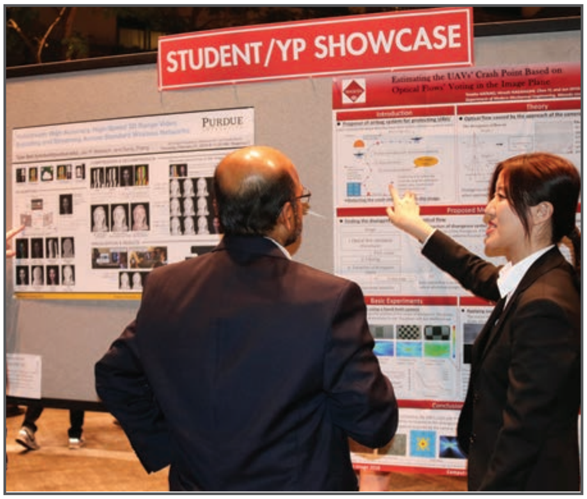
[20] Shen A. “BeaverDam: Video annotation tool for computer vision training labels”, Master Thesis, EECS Department, University of California, Berkeley, (2016).

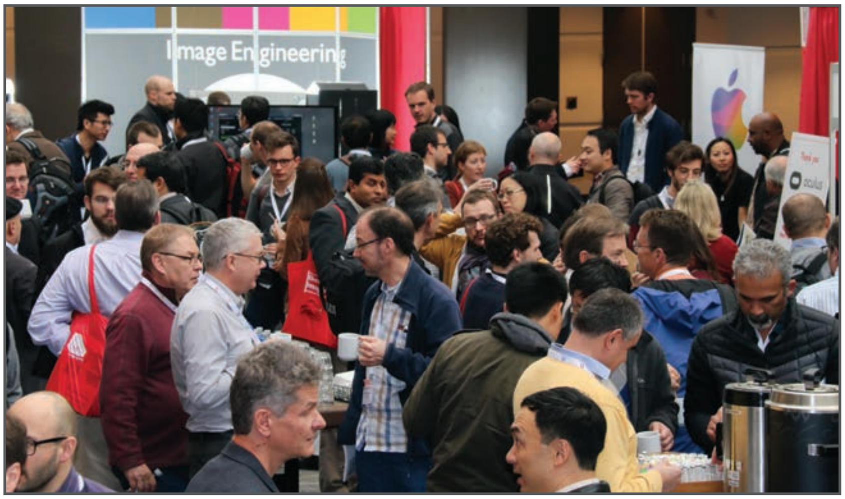
Author Biography

Florian Groh received his BSc in Computer Science from the Vienna University of Technology in 2015. He is currently enrolled at Vienna University of Technology for his MSc in Visual Computing. His focus is on semi-automatic ground truth data generation.

Dominik Schörkhuber received his MSc in Computer Science from the Vienna University of Technology in 2019. He is currently working on his PhD at Vienna University of Technology focusing on tracking algorithms.

Margrit Gelautz is an associate professor at Vienna University of Technology (TU Wien), Austria. She received her PhD in Telematics from Graz University of Technology in 1997. Her research focuses on 3D vision, stereo processing, motion analysis and image matting, with special interest in autonomous driving and human-robot interaction.





AR is a fundamental and transformative interface and platform.

20+ years from my initial working demo to today's AR displays

Optimistic that it won't take another 20 to reach ubiquitous consumer AR!