22nd ITS World Congress, Bordeaux, France, 5–9 October 2015

Paper number ITS-2643

Semi-automatic tracking-based labeling tool for automotive applications

Andoni Cortés, Orti Senderos, Nerea Aranjuelo, Marcos Nieto and Oihana Otaegui
Vicomtech-IK4, Intelligent Transport Systems and Engineering Dpt.,
Paseo Mikeletegi 57, 20009, Donostia-San Sebastián, Spain.
{acortes,osenderos,naranjuelo,mnieto,ootaegui}@vicomtech.org

Abstract
The trend toward smart cars continues to build momentum in the automotive industry. As vision based sensors proliferate in the vehicles the quantity and variety of data will become overwhelming and difficult to be processed effectively. Computer vision-based approaches can be trained and evaluated with such data sources once adequately labeled. However, accurately annotating entities in video is labor intensive and an expensive task. As the quantity of available video grows, traditional solutions to this task are unable to scale to meet the needs of sectors like automotive or security. We present a semi-automatic multi-purpose annotation tool which reduces the manual annotation effort by enabling the user to verify automatically generated annotations, rather than annotating from scratch. This tool has been successfully used in a variety of example applications in the domain of Advanced Driver Assistance Systems.

Keywords: Massive labelling, video annotation, video summarization

Introduction

In-Vehicle Infotainment (IVI) and Advanced Driver Assistance System (ADAS) technologies are becoming increasingly relevant as they promote improved passenger experiences and increased safety within vehicles. Although developing ADAS algorithms is quite a challenging task, more time is in comparison spent in testing and validating the system. In order to validate a computer vision-based ADAS, normally thousands of hours of video are necessary. Suppliers spend a lot of time collecting videos that cover all the possible scenarios, including different illumination, weather and road conditions and eventually without the appropriate tools to analyze them.

Computer vision algorithms can help managing this big data in many ways: from improving the richness of the collected metadata, to enhancing the measurements of data reliability.

Video annotation is known as the ability to add this type of metadata to the raw video footage. Current practices provide a temporary solution by paying dedicated workers to label a fraction of the total frames and or even hire external people to collaboratively accomplish this task. However, such solutions usually presents some disadvantages such as losing control of the quality of the annotations or the commitment of involved external workers. As a consequence, annotations quality needs to be revised again by the consumers to ensure good annotations.
In particular, most of ADAS applications require an annotation process of videos collected from in-vehicle cameras in order to train and test (Figure 1) systems such as:

- Forward vehicle detection in Collision Warning Systems 1
- Pedestrian detection systems 2 or other objects detection in vehicle’s proximities to turn driving a safer process.
- Traffic signal recognition systems 345 to notify and warn the driver which restrictions may be effective on the current road segment

Driver behaviors or gestures recognition in Driver Drowsiness Detection systems67 to analyze the state of the driver.

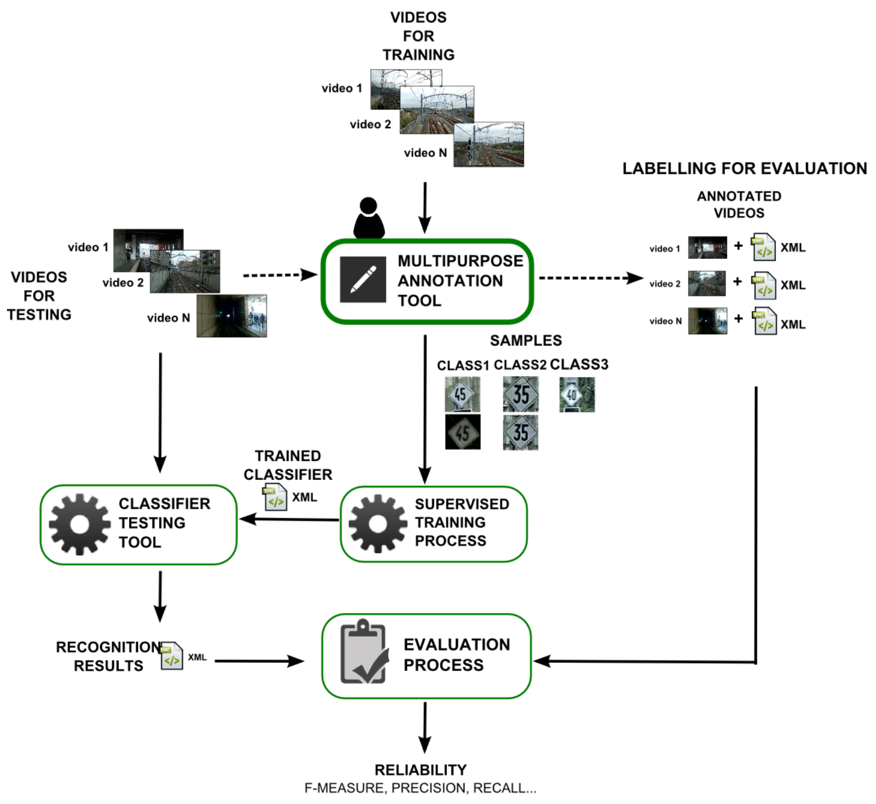


Figure 1 - Block diagram of the process of training and evaluation of a computer

vision-based system.

One of the existing approaches for designing these annotation systems is to make use of supervised classification techniques in order to train classifiers, which can recognize particular objects or behaviors in real-time. The training process itself require the annotation of a large set of samples of the object or event to be detected. This big amount of data available and necessary makes the annotation task quite tedious. Additionally, the human attention and perception decreases when executing repetitive tasks 89 which usually makes the annotation process prone to mislabeling errors directly affecting the quality of the annotated metadata. Considering these drawbacks, any help minimizing the human factor will be a key element in the development and test of new and existing ADAS systems.
Video annotation can also play an important role in algorithm and system evaluation and validation. The evaluation is methodologically carried out by comparing the system output (e.g. warnings to drivers, measured distances, etc.) with gold-reference or ground-truth annotations to obtain correlation metrics that determines the quality of the system. The definition of comparison metrics allows obtaining valuable numerical information to objectively evaluate the system performance (typically in terms of confusion matrices).

APPROACH OVERVIEW

Existing video annotation protocols 9 typically work by having users label a sparse set of key frames followed by either linear interpolation or nonlinear tracking.

In this paper, we propose an annotation strategy (annotation tool) based on trained classifiers that automatically propose label hypotheses. These hypotheses can afterwards be supervised and corrected if needed by a human operator using the annotator tool. In this way, the tool enables the user to verify the automatically generated annotations, rather than annotating the entire content from scratch. The proposed solution leads to improved performance and quality of the annotation process.

This tool has been tested in several ADAS applications developed with our Vision and Image Utility Library (Viulib(R) – version October 2013 11) such as Driver Drowsiness Detection, Collision Avoidance Systems, lane Departure warning and Traffic Sign Recognition. We have used the tool for getting samples to train supervised classifiers, as well as to evaluate our algorithms results, facing classifiers output results for different videos against their respective descriptive annotations.

As a result, the usage of this tool has become a substantial help within the design, implementation and evaluation cycle of computer vision-based systems in such a demanding sector.

IMPLEMENTATION DETAILS

The annotator is a C++ written application based on Qt, which allows developing a friendly and intuitive Graphical User Interface to ease the annotation activity as much as possible. The major properties of our annotation tool are listed below.

1) Video Navigation

The application includes a navigator for visualizing and managing video files. This navigator includes playback controllers allowing the interaction and accessibility of each frames of the video. An easy GUI has been designed to ease the video manipulation and annotation tasks.

2) Persistent storage.

Continuous saving functionality has been program to avoid any unexpected situations that might lose the ongoing work. To avoid data lost, the annotated ground truth is constantly saved in “xml” format file. Besides, this format is commonly used for evaluation purposes. Finally is worth to mention that any saved annotation work finished or unfinished can be loaded and modified easily.

3) Data types

The video annotation tools need to be flexible for dealing with diverse type of metadata in different application domains. To identify or include annotation types, the application provides a widget to add customized classes. This widget lists all existing and included classes to avoid constant class typing repetition. Hence, the classes are added once and they can be used during the whole labeling process. In case of ambiguity or other issues, the widget permits to modify or delete it. Basically, two type of elements can be annotated: “objects” which do have spatial properties that can be annotated using geometric forms (e.g. the position of a vehicle or lane in an image, as shown in Figure 2), and “events” which are defined by a time slot and a given class name (e.g. an overtaking maneuvers, lane invasions, eye closure periods, etc.)

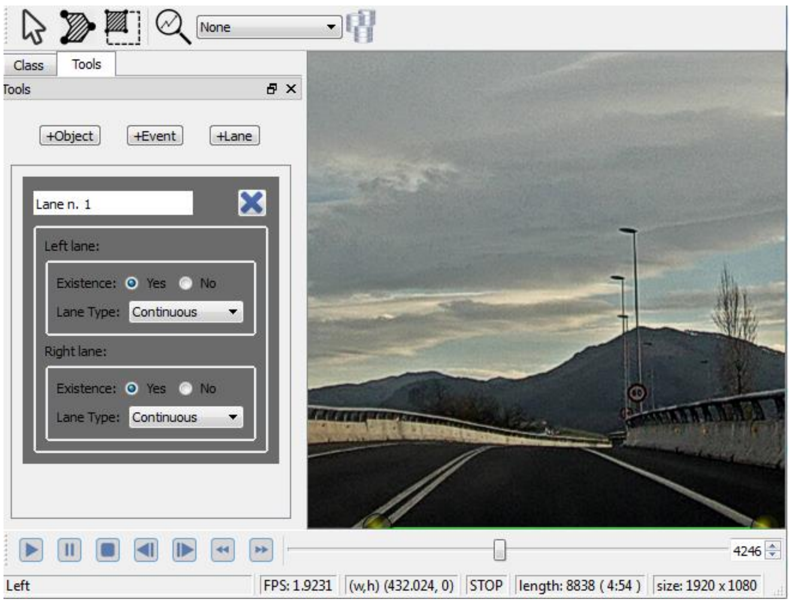


Figure 2 - Example screenshot of the GUI of the annotation tool. In this example, the annotation is generating information about the lane markings of the road.

4) Automated labeling choices

A couple of automated methods have been implemented to speed up the annotation process.

- *Automatic annotation for user validation:*

Previously labeled classes are used to train a classifier which can be used to automatically detect such classes in the video (e.g. presence of vehicles, open/closed eyes, etc.). The more videos are labeled, the better the classifier becomes, therefore the labeling process time will be considerably shortened. These hypotheses can be supervised by the user.

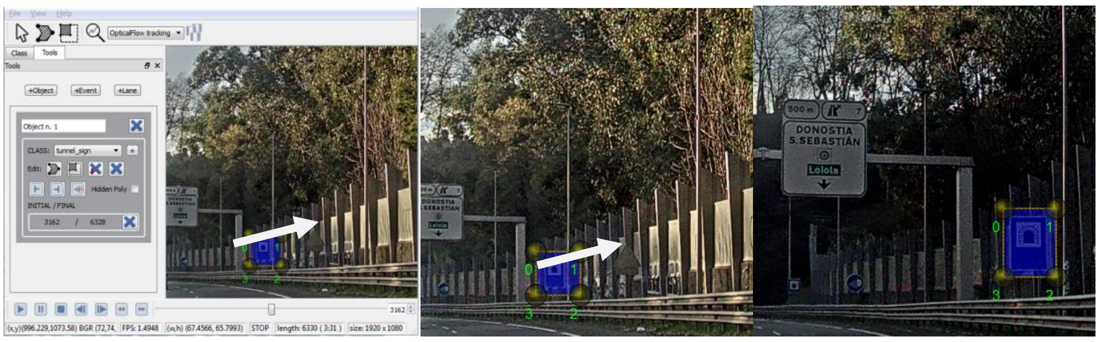
- *Semi-automated labeling:*

As a simpler approach, different types of semiautomatic algorithms have been implemented which require the initial input of the user and then track the objects automatically. Those algorithms are explained in the *Semi-automated tracked annotation* section.

LABELING PROCESS

Once the application is initiated the user can select and open any video file to start, continue or modify the labeling process. The labeling process is very simple, the user waits for objects and/or event to appear, afterwards the user selects the type of annotation that is required. The annotation is labeled and can be customizable.

For annotating objects the application offers two labeling methods. The first one consists on an automatic method based on a trained classifier. This method will automatically label objects and the user is asked only for supervising the automatic labeling process and to correct or modify the proposed labels. The second method is an iterative annotation process that works as shown in Figure 3:



(a) *(b)* *(c)*

Figure 3 - Object labeling process.

The user manually marks the object to provide the tracker with the target's initial position (Figure 3.a). The tracker will then predict the next positions of the object, Figure 3.b and Figure 3.c within the forthcoming frames. Finally the user must ensure that the prediction corresponds with the actual location of the objects. In case that the prediction is inaccurate or ended unexpectedly, the user can easily correct the shape position manually in the corresponding frame and the cycle will be repeated. It is worth noting that this annotation process is simple and repetitive, which helps to accelerate the labeling process.

Once the video is all labeled, it is saved in an xml format. This format helps in the evaluation process or can also be used to extract a database for data-driven modeling.

SEMI-AUTOMATED TRACKING METHODS

Linear approach

Linear interpolation is the simplest method of placing any shape at any position in the video file, and effectively track that object without the need of labeling every frame. The algorithm requests the user to first mark the initial location of the object and then to manually correct the annotations if needed. Every corrected frame is defined as a key frame for later interpolation. The interpolation is calculated using the shape positions between key frames. As objects can experience both location and geometry modifications, each item's node is independently interpolated.

This implementation dramatically reduces the annotation time by assuming that the annotated objects move at constant speed, which in turn works well for many scenes. For nonlinear behavior, more complex algorithms are required.

Pattern matching approach

One of the major problems of linear tracking is the lack of robustness against non-linear motion. This second tracking approach is based on the textures contained in the image patches corresponding to each annotation. The tracking will be performed between two key frames, searching for the original histogram in the in-between frames. Searching for the correct histogram in the neighborhood of the previous frame annotation is challenging because the texture or appearance is not constant throughout time in most cases.

For each annotation, defined by its positions in the image $a_t = \{x_0, x_1, y_0, y_1\}$, the appearance histogram is calculated. Afterwards the algorithm needs to search for the most similar candidate region within the next frame.

To calculate histogram, HSV color space is used instead of RGB, removing luminance channel to get a more robust matching result. Bi-dimensional histograms can be represented as follows:

$$H(a_t) = (h_0(a_t), \dots, h_B(a_t))^T, \text{ where } \sum_{b=0}^{B} h_b(a_t) = 1 \qquad (1)$$

Where a_t corresponds to the annotation a in frame t, and B is the number of bins in the histogram. $H(a_t)$ corresponds to the computed histogram, composed of a list of B normalized value bins.

Each bin value of the histogram is calculated using the following expression:

$$h_b(a_t) = \sum_{i=1}^{n_t} \delta\left(f\left(I(p_t^i)\right) - m\right) \qquad (2)$$

$$\sigma(i) = \begin{cases} 1 & i = 0 \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

Where n_t is the number of pixels which are located inside the target area at frame t and $p_t^i\ (t = 1, \dots, n_t)$ the image coordinates of the i-th pixel. The function $\delta(\cdot)$ represents the Kronecker delta function and the function $I(p_t^i)$ the i-th pixel colour at the location p_t^i. The function $f(\cdot)$ maps $I(p_t^i)$ from the value of luminance of that position in the image to the index of its bin in the histogram feature space.

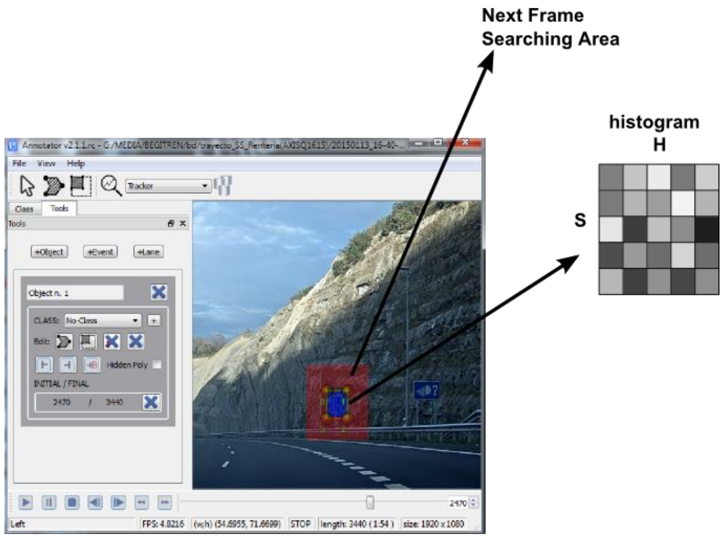


Figure 4 - Example definition of Search Area for a given annotation and its corresponding histogram.

When searching for a candidate area in the surroundings of an annotation (Figure 4), a rectangle shape will be used as a mask to mitigate the background effect. Usually the object is annotated in the shape centered, therefore more importance shall be given to pixels near the center than those closer to the boundaries. To achieve this, the Epanechnikov profile is used as a monotonic and convex kernel function to assign smaller weights to the pixels that are farther from the center 12 . Inside the rectangle a centered ellipse is defined. This ellipse is used as a mask to weight the calculated histogram.

The tracking is done by matching the parameterized template model and the candidate region with the most similar histogram distribution. The matching cost between the regularized histograms is measured with the Battacharyya distance. This distance is a measure for statistical distributions. The associated matching will be obtained by minimizing the distance between initial histogram and candidate regions histograms. This optimization process will be calculated using a conservative convex separable approximation method (CCSA)13. This kind of methods are used for inequality-constrained nonlinear programming problems when the objective function is differentiable everywhere inside the region of interest, in this particular case, the ellipse.

This method performs well in the absence of sudden changes in light that affect directly to texture illumination and tend to fail with non-rigid objects whose appearance change with time.

Optical flow-based tracking approach

Template-matching can fail when deformable objects are present in the image. The reason is that movable objects tends to change its shape or appearance.

To overcome such drawbacks, an optical-flow based approach has been included to the annotation tool application. This algorithm works by detecting keypoints or salient points for each annotation (for example, FAST detector 14). These points are then tracked using different feature point trackers, such as the KLT, or descriptor-based matchers. After the keypoint matching process is completed, the (sparse) optical flow can be used to calculate the motion of the object to update the track of the annotation.

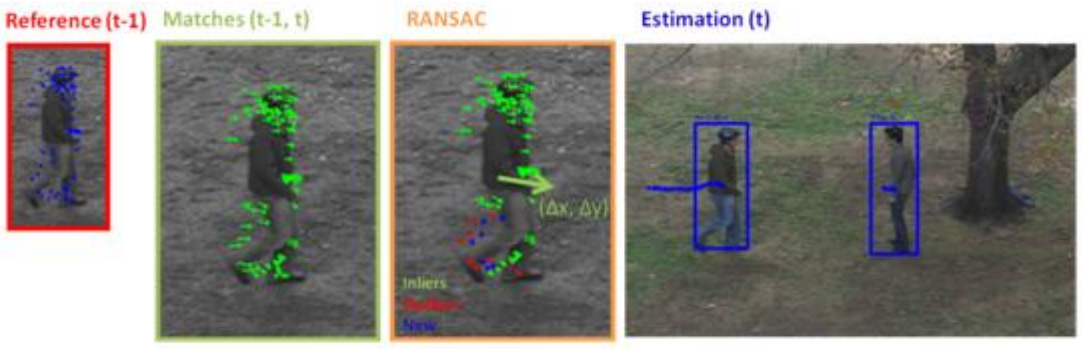


Figure 5 - Robust optical flow computation for the last track position.

In Figure 5 we can see an example calculation of the optical flow for a given track. It is worth to mention that not all the vectors of the optical flow follow the real motion of the object. These vectors can be considered as outliers of the motion model, while the vectors correctly representing the motion of the object are considered as inliers of the translational model. Therefore, a RANSAC procedure is used to separate inliers from outliers, using the 2D translational model as the function to compute the minimal sample set, the error cost and the estimation function.

In this tracking approach keypoints of the annotation area are matched from one video frame to the next one using optical flow. This approximation works successfully when the objects are visible but fails when dealing with occlusions which actually represents an absence of the object in the data and a correct representation of the scene.

USE CASES

We have used this annotation application widely for a number of system in the fields of ADAS, ITS or surveillance15. In this section we focus on ADAS use cases to demonstrate the potential of this tool in the automotive sector.

Driver Drowsiness Detection

Driver Drowsiness Detection System is a system which determines the attention, drowsiness and fatigue level of the driver by means of analyzing biometrics like eyelid closure, blinking speed and frequency. The detection of open and closed eyes with computer vision techniques can lead to such analysis. The correct detection of these two eye status require the annotation of eye samples from a wide variety of persons in order to create robust detectors that work well in real driving conditions (i.e. day and night time, presence of glasses, different color and shape of eyes, etc.). An example of open and closed eyes annotation is shown in Figure 6.

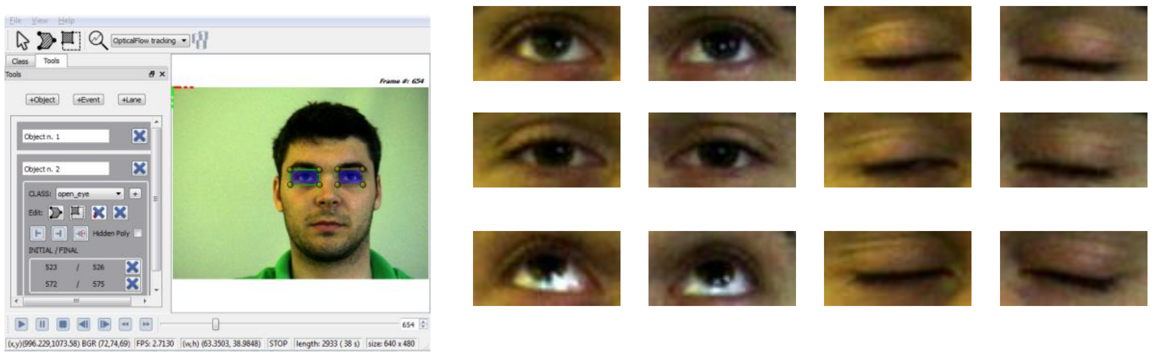


Figure 6 - Eye samples annotation for Driver Drowsiness Detection System.

Collision Avoidance System

The detection of other vehicles (Figure 7) in the road can lead to systems like Collision Avoidance or Stop and Go systems. The detection of the vehicles can be done with computer vision by training a model (or a set of models) of vehicle appearances using a large dataset of samples collected from videos. Using the annotation tool can lead to the creation of such dataset.

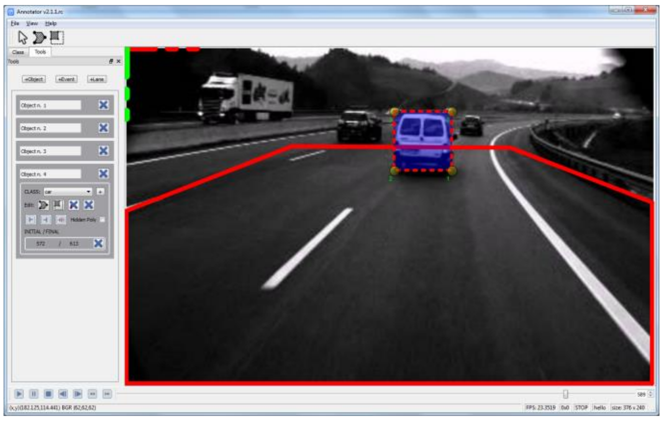


Figure 7 - Vehicle samples annotation for Collision Avoidance Systems.

Traffic Sign Recognition

Using a single camera, a system can detect the presence of signs and determine the meaning of the sign by comparing its appearance with a given database. The training process requires the annotation of sufficient samples of each sign, and the evaluation of the system can be done annotating time events defining the presence of signs or their meaning, as depicted in Figure 8.

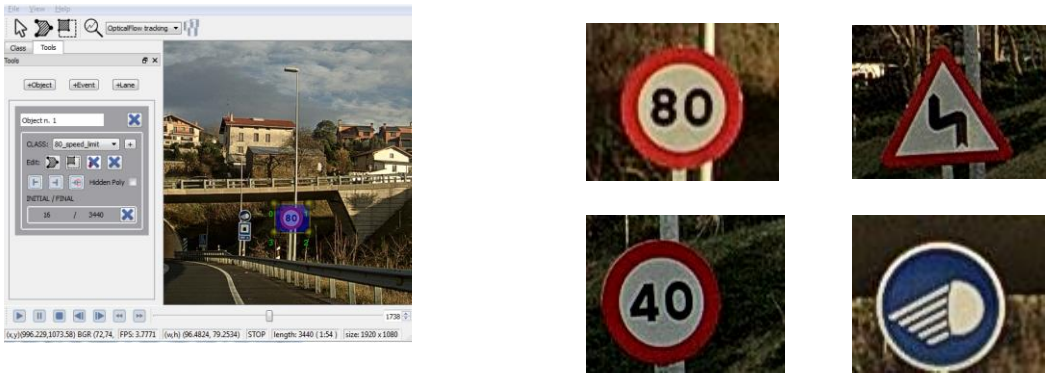


Figure 8 - Traffic signs samples annotation for Traffic Sign Recognition Systems.

Lane Departure Warning

The accuracy of the well-known lane departure warning systems can be measured using the proposed annotation tool. We have shown that the tool can be configured to annotate user-specific objects, such as a pair of numbers which can define the presence and position of lane markings in the image with respect to a given reference point. The result of the lane marking detection algorithm can then be compared with the generated ground truth information and objectively determine the accuracy and robustness of the system. This type of evaluation can provide objective evaluations of the system without incurring in the cost of an additional golden reference sensor or a dedicated controlled test circuit.

CONCLUSIONS AND FUTURE WORK

In this paper, we have shown a semi-automated multipurpose annotation tool for generating annotations (metadata) from video footage, able to adapt to a wide variety of applications and type of annotations. We have illustrated its application to ADAS such as Driver Drowsiness Detection, Collision Avoidance Systems, Lane Departure Warning or Traffic Sign Recognition.

The tool has been devised to allow interactively associating graphical annotations to independently moving video objects, creating new classes and concepts, generating spatial and time annotations, navigating through video using a intuitive GUI, etc.. Our contributions include the application of an automated tracking methods to decrease the user interaction and workload, a fluid interface for creating graphical annotations, a customizable label generation tool and an xml annotation generation.

Our experience before and after using this tool, make us to consider a very suitable way to achieve fluent and correct data to improve classifiers and a very convenient manner of accomplishing supervised classifiers training and classifiers evaluation processes.

In conclusion, we believe interactive video object manipulation can become an important tool for augmenting video as an informational and interactive medium, and that the proposed tool represents a further step towards managing sensorial vehicle data for automotive applications.

As future works, we are focusing our efforts in the direction of automatic annotation. We consider that it is possible to online feed a classifier with the user-corrected annotations the own classifier has misclassified, so that the annotation tool can be used as a platform where automatism and user validation can be efficiently optimized to make feasible the effective annotation of large volumes of sensorial data.

References

1. Chih-Li, H., Yu-Hsiang, Y., Jhih-Cheng, S., Tsung-Ying, S. (2011). Vehicle warning system for land departure and collision avoidance: Using fuzzy decision making, *Fuzzy Systems (FUZZ)*, IEEE International Conference on, pp.1554-1559.

2. Kira, Z., Hadsell, R., Salgian, G., Samarasekera, S. (2012). Long-Range Pedestrian Detection using stereo and a cascade of convolutional network classifiers, *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 2396-2403.
3. Escalera, S., Baró, X., Pujol, O., Vitrià, J., Radeva, P. (2011). Traffic-Sign Recognition Systems, *Springer Briefs in Computer Science Springer*.
4. Behloul, A., Saadna, Y. (2014). A Fast and Robust Traffic Sign Recognition, *International Journal of Innovation and Applied Studies*, vol. 5, no. 2, pp. 139–149.
5. Zaklouta, F., Stanciulescu,B. (2014). Real-time traffic sign recognition in three stages, *Robot. Auton. Syst.*, vol. 62, issue 1 ,pp. 16-24
6. Nieto, M., Otaegui, O., Vélez, G., Ortega, J.D., Cortés, A. (2014). On creating vision-based advanced driver assistance systems , *IET Intelligent Transport Systems*, p.p. 1-8, 2014
7. Jo, J., Lee, S.J., Park, K.R., Kim, I.J., Kim, J. (2014). Detecting driver drowsiness using feature-level fusion and user-specific classification, *Expert Systems with Applications*, vol. 41, issue 4, part 1, pp. 1139-1152
8. Velastin, S.A., Boghossian, B.A., Vicencio-Silva, M.A. (2006). A motion-based image processing system for detecting potentially dangerous situations in underground railway stations, *Trasportation Research Part C: Emergin Technology*, vol 14, n. 2, pp. 96-113.
9. Ainsworth, T. (2002). Buyer beware, *Security Oz*, vol. 19, pp. 18-26.
10. Vondrick, C., Ramanan, D., Patterson, D. (2010). Efficiently scaling up video annotation with crowdsourced marketplaces, *In Proc. of the European Conference on Computer Vision.*
11. 'Viulib 13.10'. Available at http://www.vicomtech.es/viulib
12. Comaniciu, D., Meer, P. (2002). Mean shift: A robust approach toward feature space analysis, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, n. 5, pp. 603–619.
13. Svanberg, K. (2002). A class of globally convergent optimization methods based on conservative convex separable approximations, *SIAM J. Optim.*, vol. 12, n. 2, pp. 555-573.
14. Rosten, E., Drummond, T. (2006). Machine Learning for High-speed Corner Detection, *in Lecture Notes in Computer Science Computer Vision – ECCV*, vol. 3951, pp. 430-443.
15. Nieto, M., Leskovsky, P., Ortega, J. D. (2014). Person detection, tracking and masking for automated annotation of large CCTV datasets, *in Proc. 8th Int. Conf. on Ubiquitous Computing & Ambient Intelligence*, LNCS 8867, pp. 519-522.