Semi-Automatic Image Annotation of Street Scenes

Andra Petrovai, Arthur D. Costea and Sergiu Nedevschi

Abstract— **Scene labeling enables very sophisticated and powerful applications for autonomous driving. Training classifiers for this task would not be possible without the existence of large datasets of pixelwise labeled images. Manually annotating a large number of images is an expensive and time consuming process. In this paper, we propose a new semi-automatic annotation tool for scene labeling tailored for autonomous driving. This tool significantly reduces the effort of the annotator and also the time spent to annotate the data, while at the same time it offers the necessary features to produce precise pixel-level semantic labeling. The main contribution of our work represents the development of a complex annotation framework able to generate automatic annotations for 20 classes, which the user can control and modify accordingly. Automatic annotations are obtained in two separate ways. First, we employ a pixelwise fully-connected Conditional Random Field (CRF). Second, we perform grouping of similar neighboring superpixels based on 2D appearance and 3D information using a boosted classifier. Polygons represent the manual correction mechanism for the automatic annotations.**

I. INTRODUCTION

Researchers, universities, large automotive companies study and work together to design the autonomous cars of the future, which will bring safer traffic, reduced number of accidents and quality commuting time. Autonomous driving is an incredible hard task for a machine due to the complexity of the environment. Cars are equipped with various sensors: cameras, laser scanners, radars which constantly monitor the surrounding. Computer vision enables the computer to see the environment and machine learning gives the ability to understand it. Visual understanding of traffic scenes is a very challenging task for a machine due to the diversity and complexity of the environment. One of the most important functions for autonomous driving is to identify the components of the scene, their shape and their location. The semantic scene parsing technique provides a detailed understanding of the surrounding by associating semantic classes to each pixel in the image. These classes could be for example sky, road, vehicle, pedestrian, building, street scene elements which we encounter in various traffic scenes.

Algorithms designed for autonomous driving need to be robust and accurate. Therefore, major research effort has been directed to creating machine learning algorithms for training classifiers to generate semantic segmented images. Nowadays, many state-of-the-art approaches are using deep neural networks [1]-[4]. Two major factors contributed to the return of deep learning classifiers: larger datasets and more powerful hardware. Therefore, a very important part in achieving top-performance with semantic labeling algorithms is using large datasets for training and evaluation. General purpose semantic labeled datasets are publicly available: Pascal VOC [6], Pascal Context [7], Microsoft COCO [8], ADE20K [9]. They contain various indoor and outdoor scenes and images of objects, persons and animals. Moreover, medium-sized datasets that focus solely on the complex traffic scene have progressed the research on autonomous driving and advanced driving assistance systems: CamVid [10], Daimler Urban Segmentation Dataset [11], Cityscapes [12]. From these, the Cityscapes dataset is the most complex, containing 5000 pixelwise annotated images and another 20000 images with coarse annotations. The dataset assures environment variability by recording inner-city scenes in 50 different cities.

Our work, the development of an annotation tool, is motivated by the need of having large datasets that could be used for training classifiers for semantic segmentation. Although, there are some publicly available datasets, the problem of annotation still remains since there are a lot of variables that differ from one camera setup to another. Vehicles could be equipped with various types of cameras with different fields of view, different orientations (lateral, back, front) and positions, making the scene structures in an image to differ greatly. Therefore, we still consider an open issue the design and development of an automatic or semi-automatic annotation tool to create datasets adapted to particular camera setups and conditions. This annotation framework supports our laboratory research effort for database creation using our camera setup. Therefore, in this paper we aim to provide details about a general framework that could be used for any kind of camera configuration.

The annotation process is very tedious work as acknowledged in [12], where the labeling task takes 1.5 hours for a human annotator that uses only layered polygons. We want to reduce the annotation effort by automating parts of the process, therefore we propose

*The authors are with the Department of Computer Science, Technical University of Cluj-Napoca, Romania.
E-mails: Andra.Petrovai@cs.utcluj.ro, Arthur.Costea@cs.utcluj.ro, Sergiu.Nedevschi@cs.utcluj.ro

a faster semantic image annotation method based on three important concepts: superpixel-based segmentation, fully-connected Conditional Random Field (CRF) and polygon-based segmentation. For now, we focus on annotating individual images, therefore we do not make any assumptions on the temporal consistency between consecutive frames. First, the image is segmented into superpixels. The number of superpixels can be chosen by the user. After that, at any moment in time the user can make one of the following actions: select one superpixel to label, use the superpixel merge tool (select one superpixel and the label is propagated to neighbor superpixels), use the polygon tool to correct objects (by adding or erasing parts of an object) or use the Conditional Random Field tool (labels pixels in region of the stroke drawn by the user). Therefore, one action is to select individual superpixels and label them by choosing one of the 20 classes. Besides that, the user can use the merge tool, which enables the grouping of multiple superpixels based on their similarity and class. For example, the user chooses a seed by clicking one time on a vehicle and the superpixels belonging to the vehicle will all be labeled as vehicle. This is possible by training a binary boosted classifier which learns the similarity between neighboring superpixels specific to the class. The user can control the estimated score returned by the Adaboost classifier and decide how much the seed label should extend to neighboring superpixels. In some cases, it is possible that the label spreads outside the boundary of the object. The user can use the polygon tool to cut or add missing parts to an object. On the other hand, we experimented with semantic labeling using inference in a Conditional Random Field (CRF) defined on the pixels of the image. The annotator who wants to label a class will draw one or more strokes on the image and the label will propagate to adjacent pixels. The user can control several parameters of the CRF to ensure that the label extends to the boundary of the selected object. From our experiments, the superpixel-based labeling yields faster results than CRF-based labeling since it requires less user interaction.

The paper is organized as follows: section II presents an overview of different annotation tools from the literature, section III describes the proposed solution with the focus on the two main contributions of our work: the superpixel-based annotation (subsection III-B) and the Conditional Random Field-based annotation (subsection III-C). We continue with the evaluation of the proposed methods in section IV.

II. Related work

Previous works in the literature tackle the subject of image annotation from two perspectives, first starting from scratch, with no prior information about the scene [13] and second using weakly labeled data in the form of bounding boxes or image tags from which pixelwise annotations are inferred [16][22][17]. Most of the approaches are based on the interaction of the annotator, which controls the labeling process.

One of the most straightforward methods for annotations is presented in [13]. The authors developed a web application in which the annotators draw polygons for each object. They support the initiative of crowdsourcing the labeling process and provide a 1000 fully annotated image database. Since this dataset is based on collaborative working, the accuracy of annotations varies from one image to another. Instead of starting from scratch, we leverage the idea of using polygons to correct automatic generated segmentations or to annotate difficult objects at far distance in cluttered urban scenes.

The authors in [14] propose a method to distinguish objects from background by placing seeds and then employing a graph-cut algorithm. A globally optimal segmentation is achieved when the user adds or removes seeds, thus achieving the desired segmentation. The method in [15] GrabCut enables the extraction of foreground-background segmentation by placing a 2D bounding box around the object. In the same manner, in [16] the authors propose several methods to generate object boundary annotations using bounding box object detector improving over fully supervised methods. Another interesting approach to image labeling represents the use of tags to describe an image [17]. Having this priors, the authors formulate the problem as a structured prediction with latent variables and show the effectiveness of the algorithms on the SIFT-flow dataset. Another idea points to recursively exploiting segmented images to guide new image labeling. This non-parametric approaches retrieve the most similar images from large databases and then a label transferring process takes place [18], [19], [20]. This methods focus more on instance labeling, while we aim at scene parsing.

Besides color information, some labeling methods take advantage of the existence of 3D point clouds obtained from a LIDAR sensor or from a stereo camera system. In [21], pixelwise semantic instance annotations are generated by annotating 3D point clouds using cuboids and primitives and then the model transfers the information into 2D by non-local multi-field CRF. The method yields good results, but is aimed only at scenes with static elements. Dynamic elements such as vehicles or pedestrians are handled in [22] as 3D CAD models. The authors employ a Markov Random Field (MRF) which uses appearance, smoothness and shape models from CAD to perform figure/ground classification on 3D bounding boxes. Multiple view semantic segmentation is achieved in [23], where a MRF is constructed across multiple images of the same scene. The point

cloud obtained from structure-from-motion is labeled using rectangular or polygonal structures and labels are transferred in the image using the 3D projections in the image. In our work, we do not label objects in 3D but we project the point cloud into the image in order to localize superpixels in the 3D world.

III. Semantic Annotation

We propose a solution for semi-automatic labeling of street scene images that can reduce the effort of the human annotator by using automatic generated annotations. In the process, we exploit appearance and 3D information obtained from LIDAR or stereo image pairs. There are three main tools that the user can choose: the superpixel merge tool, the polygon tool and the CRF tool. Each performs well in certain situations and they complement each other. In this section, we will describe in detail each tool, what are they advantages and disadvantages and how they can be used together in a single annotation framework.

A. Classes

Our framework is aimed at traffic scenes, therefore we have chosen the most relevant 19 visual classes in accordance with other datasets [12] plus an unknown class. The classes consist of static and dynamic objects and also other street elements. In Figure 1, the 20 classes are represented from which 19 represent traffic elements and one class is the unknown class. The number of classes was chosen in order to fit our needs in the process of database creation, but could be changed.

Fig. 1: 19 classes with traffic elements plus an unknown class

B. Superpixel based annotation tool

The human annotator will choose a label and click a pixel on an object he wants to annotate. The label will extend to the margin of the object, the user having the power to control how much to extend. In Figure 2 the process of semi-automatic labeling using the superpixel based tool is presented.

Next, we will describe the mechanism behind this process. First, the image is segmented into superpixels. The number of superpixels can be varied, being dependant on the image size. A smaller number of superpixels could be used when labeling large elements in the image such as sky or road and a larger number of superpixels could be used when labeling smaller elements. As the number of superpixels increases, so their dimension decreases.

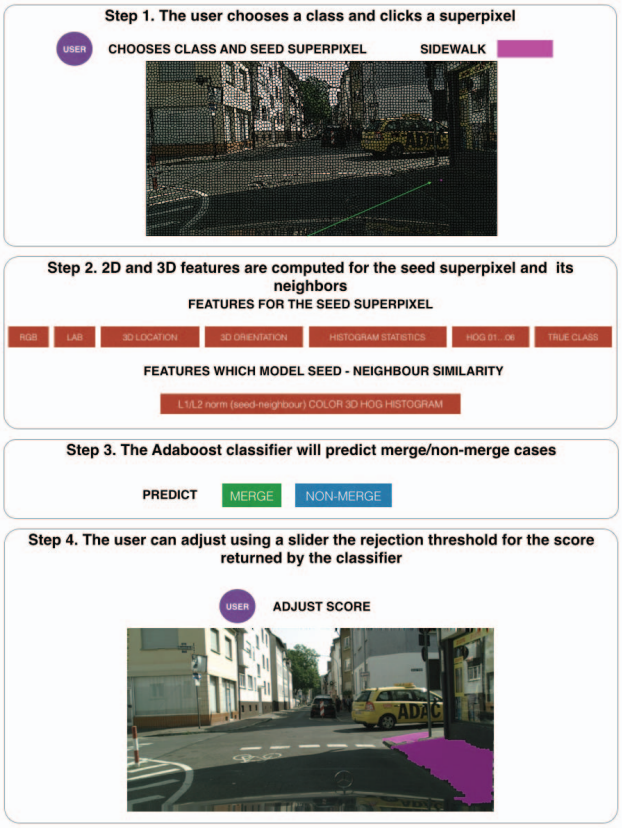


Fig. 2: Semi-automatic annotation using the superpixel merge tool: a classifier will predict merge/non-merge cases for neighboring superpixels starting from a seed superpixel

The small dimension of the superpixels assures that they adhere well to boundaries and they belong to one single object (with small exceptions). For the segmentation process we use the SLIC superpixel segmentation [24] which clusters pixels based on LAB color and position. Equation 3 represents the distance used by SLIC superpixel segmentation algorithm, where N_c and N_s are normalization factors and l, a, b color components and x, y pixel location.

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2} \quad (1)$$

$$d_s = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \quad (2)$$

$$D = \sqrt{(\frac{d_c}{N_c})^2 + (\frac{d_s}{N_s})^2} \quad (3)$$

In the next part of the paper, we will discuss the chosen features (both 2D and 3D features) for the Adaboost classifier that predicts merge/non-merge cases for superpixels and the training protocol.

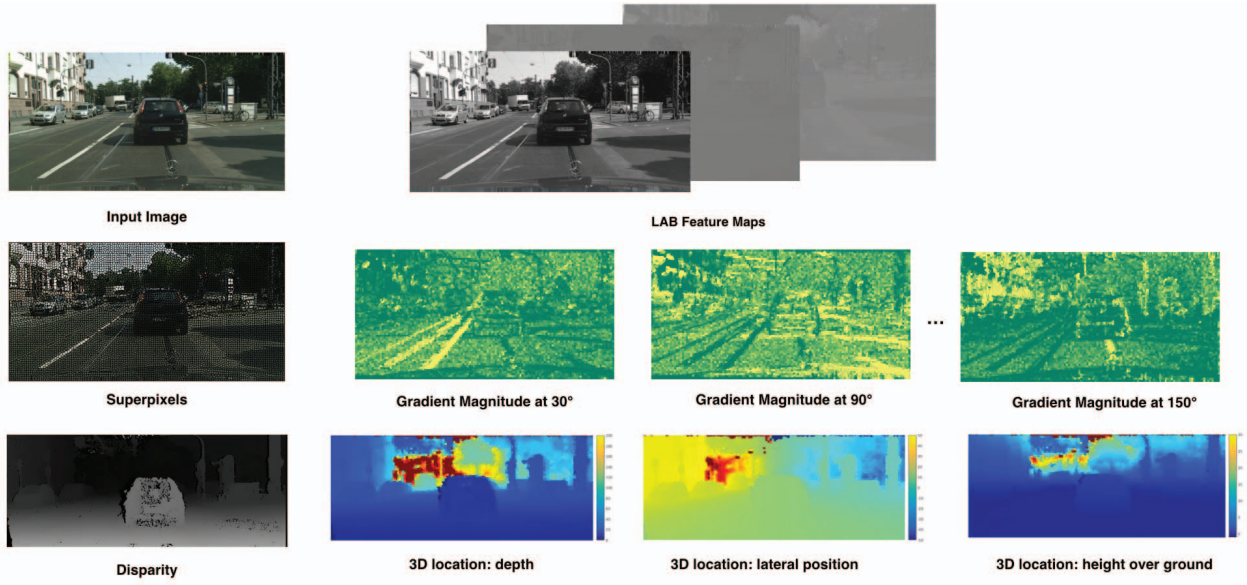


Fig. 3: The original image is segmented into superpixels. For each superpixel 2D and 3D features are computed: 3D location and orientation, HoG, color (LAB), histogram statistics etc.

1) Features: For each superpixel we compute some features: such as mean RGB color, mean LAB color, histogram statistics on the pixel of the superpixel, Histogram of Oriented Gradients. Other features are based on the disparity maps: 3D location and orientation. A representation of the features can be seen in Figure 3.

a) 2D features: Starting from a seed superpixel for which we know the label, we want to determine what neighbouring superpixels belong to the same label. For each superpixel-neighbor pair, we compute a 92-dimensional feature vector based on 2D appearance and also 3D. First, the feature vector contains the label of the seed superpixel since we want to classify the pair's similarity based on the class they belong to. Moreover, we discriminate between superpixels belonging to the foreground and those belonging to the background. Sky, vegetation, terrain, road, sidewalk, wall, building and fence represent background labels, while the others are foreground elements. Besides these features, we add the location of the seed superpixel in the image (superpixel center), thus the classifier can learn that the sky is always in the upper part of the image and the road in the lower part. Color information can give insights into the characteristics of the class, therefore we use as features the mean RGB and LAB computed over the entire seed pixels. Statistical measures over the superpixel represent its texture indirectly by modeling the properties of the distributions of gray-level intensities. We add first-order histogram based features computed over the pixels inside the seed superpixel. We analyze the shape of the histogram and its distribution by calculating the mean, standard deviation, skewness, energy (equation 4) and entropy (equation 5). All histogram statistics are computed on the grayscale levels $0..G-1$ using the probability density function $p(i)$.

$$Energy = \sum_{i=0}^{G-1} p(i)^2 \tag{4}$$

$$Entropy = -\sum_{i=0}^{G-1} p(i) log_2[p(i)] \tag{5}$$

To capture gradient orientations, the Histogram of Orientated Gradients is computed using 6 orientation bins of 30 degrees each. Since our purpose is to classify similar superpixels, we also append to the description vector measures of the similarity of the neighboring superpixel pair. Therefore, we compute the L1 norm between features of the two superpixels: RGB, LAB, histogram statistics, HOG and L2 norm for the mean color values.

b) 3D features: Spatial proximity between superpixels can determine if they belong to the same component in the image. Consequently, we localize each superpixel in the 3D world. The point cloud from LIDAR or stereo is projected onto the image. Since each superpixel belongs supposedly to one element, we compute its 3D position using a median filter to remove outliers. The following features are added to the descriptor: the 3D mean, median and standard deviation values computed over the 3D points that project inside the seed superpixel. Moreover, we are interested in the spatial distance between the seed and its neighbor since

this can give insights whether they have the same label. The L1 and L2 norms between the selected 3D features of the two superpixels are computed. Another important descriptor is the 3D orientation of the superpixel. The orientation gives information about the type of structure, whether we have a horizontal surface like the road, terrain, sidewalk or a vertical one such as buildings, poles, walls, fences. Superpixels that have the same orientation are more likely to belong to the same labeled element. To capture this intuition, we fit a plane to each superpixel using least squares fitting. Finally, the orientation descriptors will be the direction of the normal to the plane of the seed superpixel and the angle between the normal to the seed plane and the normal to the neighbor superpixel plane.

2) Training protocol: Let's assume that there is a superpixel for which we know the label, having been chosen by the user. In order to extend the label to the entire element in the image, we want to find the superpixel's neighbours which belong to the same component and to propagate the label towards them. Therefore, starting from a seed superpixel the label is transferred to neighbors using a breadth first search manner.

To decide which superpixels are part of the same component as the seed, we train an AdaBoost classifier. We use 512 weak learners with 3-level decision trees. For generating the training samples, we collect for each label 190000 positive examples of cases where 2 neighboring superpixels have the same label and they are allowed to merge. The same number of negative samples was extracted representing the non-merge case, where superpixels have different labels. In order to balance the frequency of the classes, we choose the exact same number of positive/negative samples for each label. The samples were collected from 2975 pixelwise annotated images of the Cityscapes dataset [12]. Images from the train set were segmented into superpixels and each received a label according to the ground truth. We consider only superpixels where a single label covers more than 85% of the pixels inside, the others being ambiguous since they extend on more than one object in the scene. The user can control in the interface of the application through a slider the rejection threshold for the score returned by the AdaBoost classifier for two superpixels. The annotator will choose the best threshold that labels correctly the majority of the superpixels from the chosen element.

C. Conditional Random Field Annotation Tool

Independent of the superpixel based annotation tool, our framework offers another feature for semi-automatic labeling based on the Conditional Random Field. The flow of the semi-automatic CRF based annotation can be seen in Figure 4.

Fig. 4: CRF based semi-automatic annotation

Given an image to be labeled, we define a fully-connected Conditional Random Field as in [25] over the entire image. We want to model the relationships between every pair of pixels in the image by defining a smoothness term that maximizes label agreement between similar pixels. Let $G = \langle \mathcal{P}, E \rangle$ be a graph where each vertex $p \in \mathcal{P}$ represents a pixel and E are the set of edges connecting every pair of pixels in the image.

Every node in the graph should receive one label $s = s_i$. This problem can be solved by minimizing a Gibbs energy function:

$$E(s) = \sum_{p_i \in \mathcal{P}} \psi_i(s_i) + \sum_{e_{ij} \in E} \psi_{ij}(s_i, s_j) \qquad (6)$$

$\psi_{ij}(s_i, s_j)$ is the pairwise potential which models the relations between two pixels i and j having labels s_i and s_j.

The pairwise potential ensures label coherence in the image and is defined as a sum of Gaussian kernels:

$$G_1 = e^{-\frac{||p_i - p_j||^2}{2\sigma_\gamma^2}} \qquad (7)$$

$$G_2 = e^{-\frac{||p_i - p_j||^2}{2\sigma_\alpha^2} - \frac{||I_i - I_j||^2}{2\sigma_\beta^2}} \qquad (8)$$

$$\psi_{ij}(s_i, s_j) = \mu(s_i, s_j)(w^{(1)}G_1 + w^{(2)}G_2) \quad (9)$$

$$\mu(s_i, s_j) = [s_i \neq s_j] \quad (10)$$

In equation 7, G_1 is the smoothness term. It is used to remove isolated labels. p_i represents the position of one pixel and p_j is the position of the other pixel.

In equation 8, G_2 represents the appearance kernel and is computed as the difference between a Gaussian kernel over the L2 norm on the positions of the two pixels p_i, p_j and a Gaussian kernel over the L2 norm on the intensities of the two pixels I_i, I_j. This term describes the intuition that pixels in nearby location that have similar colors should have the same label.

$\psi_i(s_i)$ is the unary potential of a pixel and is usually the output of a classifier which computes a probability distribution over the labels.

The Conditional Random Field is often used as a post-processing step in a segmentation process. We propose to use it as a labeling tool. The user will choose a class s_i and draw a stroke on one selected element in the image. For that pixels the corresponding unary potential will be set as following: $\psi(s = s_i) = 0$ and $\psi(s \neq s_i) = 1$.

The weights $w^{(1)}$, $w^{(2)}$ and the values for the standard deviation σ_α, σ_β, σ_γ are unknown. From experiments we found that $w^{(1)} = 2$ and $\sigma_\gamma = 2$ give good results for our labeling process. Moreover, σ_β is set to 20 as in [25]. We let the user set the weight for the appearance term $w^{(2)}$ and the standard deviation for location σ_α by moving a slider in the interface. This is an efficient way in which the annotator receives live feedback from the system and can control how extensive the neighborhood from seed pixels should extend. In this neighborhood we consider that pixels with similar color are likely to be in the same class.

IV. Evaluation

In the following section we evaluate the superpixel based annotation tool and the CRF based annotation tool. For evaluation, we used the validation dataset from the Cityscapes dataset [12] which contains 500 pixelwise annotated images. We take into consideration 19 classes specific to road scenes as mentioned before and the unknown class.

A. Analysis of the dataset

We analyze distributions of ground truth labels in superpixels and the distributions of classes in the dataset. In this experiment we segment the images into 8000 superpixels. Each superpixel receives a class based on the majority number of labels from the ground truth image on its pixels.

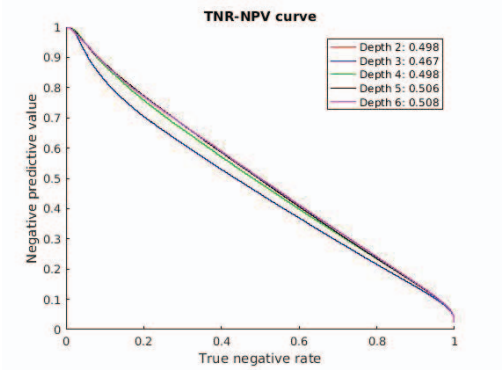


Fig. 5: True negative rate vs Negative predictive value for different number of levels for the 512 weak learners of Adaboost

We computed the distribution of classes per superpixel on the entire validation set as seen in Table I. The road class is the most predominant in the image, followed by building, vegetation and unknown.

B. Superpixel based annotation tool evaluation

In order to evaluate the performance of our classifier we measure the area under the curve for true negative rate vs negative predictive value. True negative rate relates to the ability of the classifier to detect non-merge cases. Non-merge cases occur at the boundaries of objects and are fewer than the merge cases. In our first experiment, we vary the number of levels for the decision tree. The area under the curve measure indicates that 3 levels gives the best results for the validation set, Figure 5.

In the next experiment, we tested the performance of our classifier by changing the number of weak learners. Table II summarizes the performance. One can observe that there is no increased gain by using a larger number of weak learners, therefore we decided to have a faster classifier with 512 weak learners.

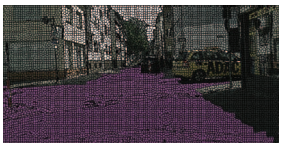
C. Methodology for labeling

The superpixel based annotation can be used together with the Conditional Random Field annotation. The CRF method works best on very well defined elements with homogeneous color, for example sky, vegetation, road. On the other hand, superpixels combine appearance information with 3D data. 3D reconstruction from stereo gives good results closer to the camera with very large errors at more than 100 meters. One of the advantages of using the Adaboost classifier is that it also acts as a feature selector: it can learn that at very large distances the 3D data is not relevant. Superpixel-based annotation can be used successfully on almost every class. Derived from experiments, the proposed methodology for annotation is the following:

TABLE I: Distribution of classes measured for superpixels on the validation set

Total number of superpixels	Class %																			
	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle	unknown
4101571	35.07	4.54	19.21	0.6	0.7	0.74	0.13	0.53	14.96	0.67	2.66	1.02	0.15	5.88	0.27	0.35	0.1	0.06	0.53	11.83

(a) Original image



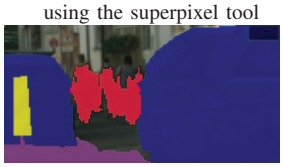
(b) Road annotation using the superpixel annotation tool

(c) One click car annotation using the superpixel tool

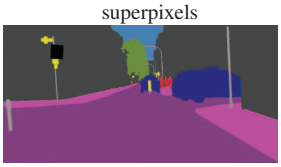
(d) Vegetation annotation using superpixels

(e) One stroke CRF based sky annotation

(f) Intermediate step: correct edges using additive or erasing polygons



(g) One click superpixel based annotation for far away pedestrians: needs correction



(h) Final result

Fig. 6: Annotation of one street scene image using various techniques

Nr. weak learners	AUC for TNR-NPV
512	0.498
1024	0.499
2048	0.5
4096	0.502

TABLE II: Area under the curve for true negative rate vs negative predictive value for different number of 4-level weak learners

- choose one element in the foreground and annotate using superpixels or CRF
- correct the element if necessary by using the superpixel eraser or polygon based eraser
- polygons can also be used to fill in holes
- individual superpixels can be drawn using strokes with the superpixel drawing tool
- when drawing individual superpixels the number of superpixels can be controled by the user (thus the size of a superpixel)
- once the element is done start with another element
- the most predominant element in the scene can be labeled last by filling the remaining pixels in the image with a certain class (especially helpful with buildings)

Large scene elements such as sky, vegetation, road are easy to label since they have a smooth appearance. On the other hand, buildings can be more difficult to annotate because they exhibit variation in texture and color and often are situated at large distances where the 3D information is not reliable. All in all, the superpixel based annotation tool may be more effective and fast than CRF due to its reduced number of operations (1 vs 4 parameters to be controlled). Depending on the scene structure, illumination, shadows, presence or absence of 3D information, in order to label one street element the user does one or more operations by choosing the class, the tool, controlling the parameters, correcting the automatic annotations. From our experiments the time required to annotate one image has been reduced approximately 3 times in comparison with classical methods (polygons), depending on the complexity of the scene. In Figure 6, we present the workflow of annotating one image using both superpixels and CRF based semi-automation methods. In Subfigure 6b the road was annotated using the superpixel annotation tool. The user clicked on a superpixel on the road and the superpixels merged starting from the seed superpixel receiving the road label. Next, in SubFigure 6c the vehicle was annotated with one click in the same manner. We also used the superpixel merge tool when annotating the vegetation, Subfigure 6d. The sky was labeled with the CRF tool by drawing a stroke on the sky, Subfigure 6e. Some objects may not be perfectly labeled, therefore the user can correct the annotations using the polygon tool, which makes it easy to add new parts or erase parts of an object with great precision. Subfigure 6f presents such a case where the vehicle objects are corrected with polygons.

As far as time is concerned, it takes 30 minutes on average to annotate one image.

V. Conclusions

In this work we introduced a novel annotation framework for labeling street scene images. The aim of this application is to ease the work of the annotator by

automatically generating annotations which can then be corrected. Our annotation framework is versatile and can be used on various images acquired by different cameras in different configurations. For automatic generation of labels we propose two novel methods: one based on a fully-Conditional Random Field and one based on the idea that an element in the scene is composed of similar superpixels which can be merged into a single object. We train an Adaboost classifier to decide merge and non-merge situations between neighboring superpixels. As corrective methods we employ individual superpixels coloring and polygons. We evaluated our method on the Cityscapes dataset and proved that the application provides useful tools for faster pixelwise annotation compared to classical methods. As future work, we plan to investigate how to improve the application and develop more features to further reduce the work of the annotator. One idea would be to generate an initial segmentation using state-of-the-art segmentation methods which can be corrected in a second step.

ACKNOWLEDGEMENT

This work was supported by the EU H2020 project UP-Drive under grant nr. 688652.

REFERENCES

[1] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation", in CVPR, 2015

[2] G. Lin, A. Milan, C. Shen, and I. Reid, "RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation", arXiv preprint:1611.06612, 2016

[3] G. Ghiasi, and C. C. Fowlkes, "Laplacian Pyramid Reconstruction and Refinement for Semantic Segmentation", in ECCV, 2016

[4] M.Treml, J, Arjona-Medina, T. Unterthiner, R. Durgesh, F. Friedmann, P. Schuberth, A. Mayr, M. Heusel, M. Hofmarcher, M. Widrich, B. Nessler, S. Hochreiter, "Speeding up Semantic Segmentation for Autonomous Driving", in NIPS Workshop, 2016

[5] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge", in IJCV, 2015.

[6] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I.Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes challenge: A retrospective", in IJCV, 2015

[7] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille, "The role of context for object detection and semantic segmentation in the wild", in CVPR, 2014

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollr, and C. L. Zitnick, "Microsoft COCO: Common objects in context", in ECCV, 2014

[9] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso and A. Torralba, "Semantic Understanding of Scenes through ADE20K Dataset", arXiv preprint:1608.05442, 2016

[10] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database", Pattern Recognition Letters, 30(2), pp. 88-97, 2009.

[11] T. Scharwchter, M. Enzweiler, U. Franke, and S. Roth, "Efficient multi-cue scene segmentation", in GCPR, 2013

[12] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in CVPR, 2016

[13] B. Russell, A. Torralba, K. Murphy, W. T. Freeman, "LabelMe: a database and web-based tool for image annotation", in International Journal of Computer Vision, 2007

[14] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images", in ICCV, 2001

[15] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut-interactive foreground extraction using iterated graph cuts", in SIGGRAPH, 2004

[16] A. Khoreva, R. Benenson, M. Omran, M. Hein, and B. Schiele, "Weakly supervised object boundaries", in CVPR, 2015

[17] J. Xu, A. Schwing, and R. Urtasun, "Tell me what you see and i will show you where it is", in CVPR, 2014

[18] M. Guillaumin, D. Kuttel, and V. Ferrari, "Imagenet auto- annotation with segmentation propagation", in International Journal of Computer Vision (IJCV), 110(3), pp.328-348, 2014

[19] C. Liu, J. Yuen, and A. Torralba, "Nonparametric scene parsing via label transfer", in IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI), 33(12), pp. 2368-2382, 2011

[20] G. Singh and J. Kosecka, "Nonparametric scene parsing with adaptive feature relevance and semantic context", in CVPR, 2013.

[21] J. Xie, M. Kiefel, M.-T. Sun, and A. Geiger, "Semantic instance annotation of street scenes by 3D to 2D label transfer", in CVPR, 2016.

[22] L.-C. Chen, S. Fidler, A. L. Yuille, and R. Urtasun, "Beat the mturkers: Automatic image labeling from weak 3d supervision", in CVPR, 2014

[23] J. Xiao and L. Quan, "Multiple view semantic segmentation for street view images", in ICCV, 2009

[24] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P.Fua and S. Ssstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods", IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(11), pp.2274-2282

[25] P. Krahenbuhl and V. Koltun, "Efficient inference in fully connected crfs with gaussian edge potentials", in Advances in Neural Information Processing Systems, vol.24, pp. 109-117, 2011