Tree-Structured Models for Efficient Multi-Cue Scene Labeling

Marius Cordts*, Timo Rehfeld*, Markus Enzweiler, Uwe Franke, and Stefan Roth, *Member, IEEE*

Abstract—We propose a novel approach to semantic scene labeling in urban scenarios, which aims to combine excellent recognition performance with highest levels of computational efficiency. To that end, we exploit efficient tree-structured models on two levels: pixels and superpixels. At the pixel level, we propose to unify pixel labeling and the extraction of semantic texton features within a single architecture, so-called *encode-and-classify trees*. At the superpixel level, we put forward a multi-cue segmentation tree that groups superpixels at multiple granularities. Through learning, the segmentation tree effectively exploits and aggregates a wide range of complementary information present in the data. A tree-structured CRF is then used to jointly infer the labels of all regions across the tree. Finally, we introduce a novel object-centric evaluation method that specifically addresses the urban setting with its strongly varying object scales. Our experiments demonstrate competitive labeling performance compared to the state of the art, while achieving near real-time frame rates of up to 20 fps.

Index Terms—Scene labeling, automotive, decision forests, segmentation tree, depth cues, superpixels, stixels

1 Introduction

Visual semantic scene understanding has gained increasing interest in recent years in both the scientific community and in applications. While considerable progress has been made, a sizable gap to human performance levels remains.

General settings of scene understanding require a large number of object classes to be recognized and allow imposing only few constraints on the overall scene layout [1], [2], [3]. In contrast, task-specific settings typically involve a more restricted domain, *e.g.* scene labeling for indoor [4] or urban outdoor scenes [5], [6]. In those scenarios, the number of different object classes to be recognized is typically much smaller; moreover, assumptions on the scene layout aid the recognition task. While simplifying the problem somewhat, a number of significant challenges remain, such as highly varying object scale and motion, partial occlusions, and strong demands on computational efficiency, *e.g.* for real-time mobile or robotics applications. Hence, a number of recent approaches have focused on these task-specific domains [6], [7], [8], [9].

To approach the challenges of urban scene labeling, we make three observations: First, multiple cues are readily available, but have not been exploited to their full extent. Second, high quality region proposals are crucial for classification. Third, tree-structured models allow for efficient prediction. Therefore, our approach leverages tree structures to maximize the re-use of intermediate computations, and at the same time to integrate multiple cues wherever they provide complementary information.

- *M. Cordts and T. Rehfeld are with Department of Environment Perception, Daimler AG, Böblingen, Germany and with Department of Computer Science, TU Darmstadt, Darmstadt, Germany.*
  E-mail: {marius.cordts, timo.rehfeld}@daimler.com
- *M. Enzweiler and U. Franke are with Department of Environment Perception, Daimler AG, Böblingen, Germany.*
  E-mail: {markus.enzweiler, uwe.franke}@daimler.com
- *S. Roth is with Department of Computer Science, TU Darmstadt, Darmstadt, Germany.*
  E-mail: sroth@cs.tu-darmstadt.de

*The first two authors contributed equally.

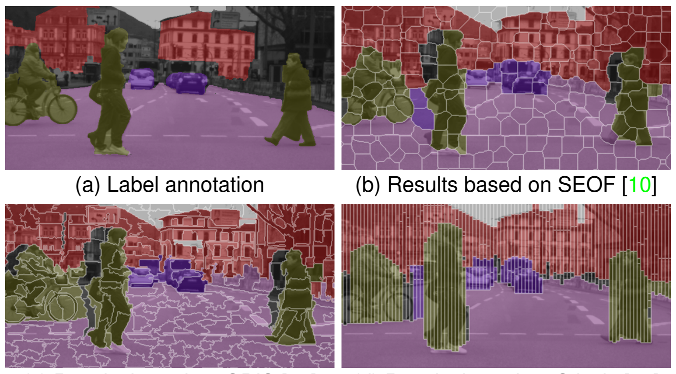

(a) Label annotation (b) Results based on SEOF [10] (c) Results based on GBIS [11] (d) Results based on Stixels [12]

Fig. 1. Example results of our proposed method on three kinds of superpixels, and comparison to ground truth annotation.

Besides appearance, we also use cues such as depth, geometry, object detectors, and motion. Figure 1 shows example results.

Our focus is to build a state-of-the-art scene labeling system with lowest possible execution time. To this end, we propose a system that integrates and extends well-known components in an efficient and clever way. Two tree structures build the foundation of our work: an extension of randomized decision forests (RDFs) at the pixel level and segmentation trees at the superpixel level. While both types of approaches have been studied extensively, we demonstrate the benefit of combining them. Our method extracts semantic labels at the pixel level jointly with texton histograms through a novel variant of RDFs, so-called *encode-and-classify trees*. While accurate pixel-level labels help to construct our segmentation tree, cumulative texton histograms support efficient classification of all its segments.

Figure 2 shows an overview of our method. Superpixels are extracted and serve as the lowest level of our multi-cue segmentation tree. Each region proposal from this tree is classified and forms a node in a tree-structured conditional random field (CRF). Both feature extraction and inference are very efficient due to

To appear in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI).* doi:10.1109/TPAMI.2016.2592911

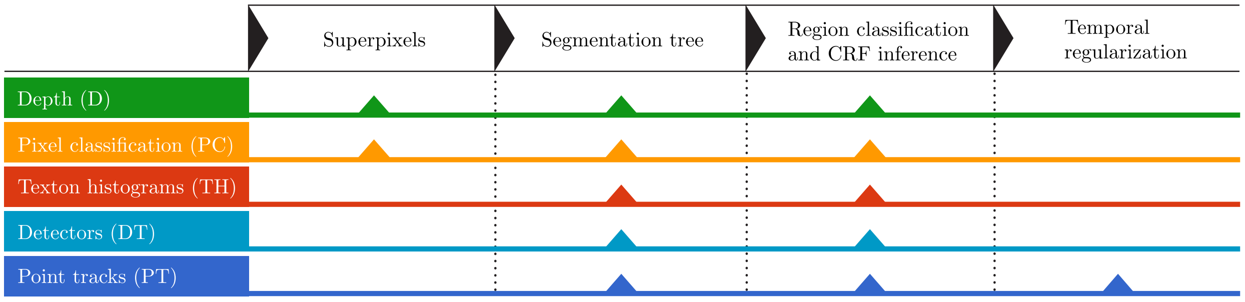


Fig. 2. Method overview. Arrows indicate the contribution of each available cue (rows) to the individual processing steps (columns). Central to our approach are the proposed encode-and-classify trees, which generate pixel-level classification and texton histograms at the same time, and the segmentation tree. Pixel classification and texton histograms are detailed in Sec. 3.1, the other cues in Sec. 5.1.

this tree structure. Features are computed once for the leaves and accumulated in the parents on the path from leaf to root. For inference, we apply acyclic belief propagation (BP) yielding exact marginals.

2 Related Work

There are four major related categories of research in scene labeling. The first involves the use of *segmentation trees*. A multitude of approaches [4], [7], [8], [13], [14], [15], [16], [17], [18] relies on the hierarchical segmentation of Arbeláez *et al*. [19], whereas others [20], [21], [22] construct the tree from multiple runs of simpler superpixel algorithms, *e.g*. [11]. Some of the methods exploit the tree structure for inference [16], [17], [18], [20], [21], as we do here. However, to the best of our knowledge, only two systems have exploited additional cues to generate the segmentation tree, particularly depth [4], [7]. A true multi-cue segmentation tree, as proposed here, has not been pursued.

The second category covers *multi-cue scene labeling*. Recognition performance can be significantly improved through the use of cues that are complementary to the image channel, *e.g*. depth from stereo or RGB-D [6], [7], [23], [24], object detectors [7], [8], [13] or motion [6], [23], [24], [25]. Multiple modalities have been used for superpixel generation [7], [24], [26], but, except for [4], not for a subsequent grouping of region proposals. In our approach, various cues are jointly utilized for all algorithmic stages, *i.e*. superpixel extraction, grouping, and classification, see Fig. 2.

The third category involves *RDFs* and their application to fast feature encoding and pixel labeling. First proposed by Moosmann *et al*. [27], their use has been popularized by Shotton *et al*. [28] in terms of texton histograms. These infer a hierarchical partitioning of texture and avoid the need to compute expensive descriptors. However, they can only provide a rough semantic category prior for each pixel and the resulting histograms are very high-dimensional, which is impractical from an application point-of-view (*c.f*. [28], Sec. 7). We solve this by proposing a novel tree structure with explicit *encoder nodes* to obtain both compact and highly discriminative texton histograms, as well as accurate pixel-level labels simultaneously.

The fourth category involves Conditional Random Fields (CRFs) for pixel-level classification tasks, *e.g*. [17], [22], [23], [29], [30], [31], [32]. CRFs typically model statistical dependencies between pixels and effectively couple unary classifier scores with pairwise or higher-order potentials. The spatial potentials are often constructed based on a bottom-up segmentation and support label consistency of pixels in the same segment [17], [22], [23], [30]. We follow this idea and connect parent with child regions in our segmentation tree via CRF potentials. In doing so, the CRF allows for an information flow between different segmentation granularities.

Three recent approaches are highly related to ours. Silberman *et al*. [4] segment a scene into support relations using RGB-D cues. Our system employs a segmentation tree constructed in a similar spirit. However, in our case information from semantics are the driving force in tree construction, while being a side effect in [4]. The work of Gupta *et al*. [7] exploits tree structures as well as multiple cues, and shows impressive results on the NYUD2 dataset. To obtain region proposals, a segmentation tree is generated using a boundary detector based on appearance and depth. The resulting regions are classified using a depth-augmented CNN-based detector. Eventually, region proposals and detections are used for semantic labeling. In contrast to both works, we build the segmentation tree using robust and efficient features, leverage the tree structure for efficient inference, and integrate additional cues such as motion and detectors. In doing so, we propose a near real-time method with a focus on outdoor street scenes. Also related is the urban scene labeling approach of Scharwächter *et al*. [6], which uses multi-cue scene labeling and RDFs. Competitive results are shown by combining fast features with spatio-temporal regularization. However, they only rely on a single layer of greedily generated region proposals and thus cannot recover from errors on this level. In addition, they do not use object detectors and strongly depend on the particular choice of superpixels.

In light of the preceding discussion, the main contributions of this paper are: *(1)* randomized decision forests with explicit *encoder nodes* to simultaneously provide discriminative texton histograms and accurate pixel-level labels; *(2)* a *multi-cue segmentation tree* that effectively exploits complementary information during tree construction and classification; *(3)* an *object-centric evaluation measure* that is suitable for typical outdoor street scenes and complements traditional metrics.

3 Tree-Structured Models

We build our approach around two tree-structured models: encode-and-classify trees on the pixel level for efficient dense feature encoding and pixel classification (Sec. 3.1), as well as a segmentation tree that integrates multiple cues on the segment level (Sec. 3.2). In Sec. 3.3, we briefly discuss the three superpixel variants used for evaluation. Region classification and CRF inference in the segmentation tree are described in Sec. 3.4. Within all stages of our pipeline, we leverage depth, 3D motion in the form of point

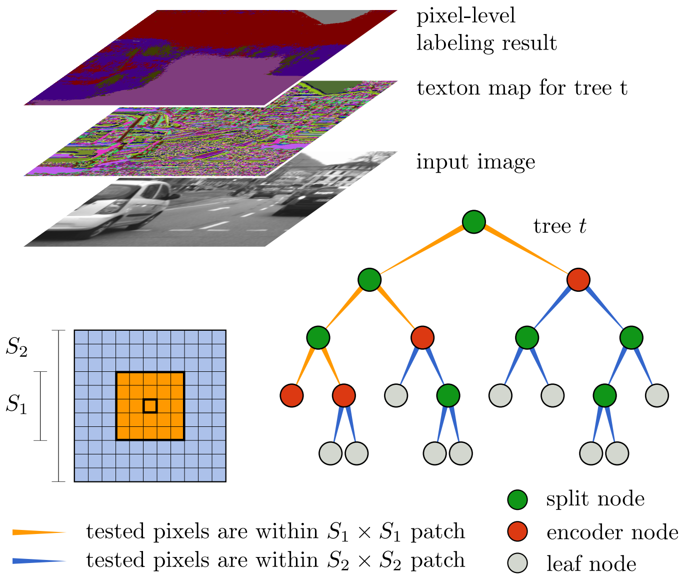

pixel-level labeling result
texton map for tree t
input image
tree t
S_2
S_1
split node
encoder node
leaf node
tested pixels are within $S_1 \times S_1$ patch
tested pixels are within $S_2 \times S_2$ patch

Fig. 3. Our proposed encode-and-classify tree structure for pixel-level classification and texton extraction (bottom). Encoder nodes define a sub-tree that operates on small local patches of size $S_1 \times S_1$ to obtain highly discriminative texton histograms. The remaining nodes have full access to all pixels in the larger region $S_2 \times S_2$ for accurate classification. Example of a texton map generated using one tree and the pixel-level labeling result (top).

tracks, and object detectors; see Fig. 2 for an overview of all cues and where they contribute. Note that our approach is independent of the actual realization of these cues. Therefore, details are given later in Sec. 5.1.

3.1 Encode-and-classify trees

Rather than building our approach solely on top of superpixels, we start on the level of individual pixels using a novel randomized decision forest. This classifier simultaneously provides pixel-level semantic scores as well as texton maps to generate segment-level bag-of-features histograms. As shown in [28], the tree structure of the RDF provides a discriminative clustering of the feature space and pooling the IDs of visited nodes over an image region yields a texton histogram. We follow the idea of [28], but instead of using two decision forests, one for texton extraction and one for pixel-level labeling, we use a single forest for both tasks. To maximally re-use computational resources, we use the output of this classifier as higher-order features for superpixel generation, segmentation tree construction, and classification of region proposals. Both, semantic pixel-level labels and a texton map are visualized in Fig. 3.

The encode-and-classify concept put forward in this work is based on the observation that pixel-level classification and discriminative texton generation have different demands on the algorithm parameters. In particular, we find that good pixel-level accuracy requires medium to large patches while texton histograms are more discriminative if textons are extracted on rather small patches. Our intuition is that visited node IDs of neighboring pixels are less correlated when smaller patches are used, so that in turn the benefit of histogram pooling is more pronounced. We support this finding experimentally in Sec. 5.2. Furthermore, we observe that deeply trained trees improve pixel-level accuracy. However, with increased tree depth the number of nodes, *i.e.* the histogram length and with it the region classification runtime, increases quickly. This fact has also been pointed out in the conclusion of [28] and is confirmed empirically in Sec. 5.2.

To overcome these conflicting demands on the tree parameters, one could naturally use separate models for pixel labeling and region encoding, similar to [28], which however comes at the cost of additional runtime. We therefore propose to combine the ideal operating points for both individual tasks within a single dual-use model, by introducing two concepts: sub-trees for feature encoding and range restriction. We use explicit *encoder nodes*, which are special nodes in the decision trees, as depicted in Fig. 3. Starting from the root node, encoder nodes can be seen as leaf nodes of a sub-tree that has the main purpose of texton extraction. At the same time, every encoder node can be the root of a following sub-tree dedicated for pixel labeling, so that there is always only one encoder node on the path from the root node to any leaf node. In contrast to [27], [28], only those explicit encoder nodes contribute to the texton histogram. At the same time, split tests of the texton extraction sub-tree are restricted to be within a small local range, while the following sub-tree has access to all pixels within a larger range, as indicated by the corresponding colors of tree edges and patches in Fig. 3. By combining both concepts, we gain full control over the histogram length to find the best trade-off between region-level accuracy and runtime and at the same time keep pixel-level labeling accuracy high. Training of the decision tree is detailed in Sec. 4.

3.2 Multi-cue segmentation tree

The key motivation of the segmentation tree is to provide region proposals at multiple granularities, capable of capturing the significant range of scales present in street scenes. Typically, each object in the scene is then accurately covered by one region in the tree. This maximizes its spatial support, which is especially beneficial for the robustness of region classifiers as used in Sec. 3.4. Note that all features on a region level, such as histograms, geometry, occlusion, and average pixel classifier scores, can be computed in a cumulative fashion. Thus, they need to be computed only once for all superpixels on the lowest level, and can then be accumulated from leaf to root. Due to the multiple levels of the segmentation tree, each superpixel is covered by several overlapping regions. The final label decision is thus made in a subsequent inference stage; a tree-structured CRF allows to do this in an efficient and globally consistent way.

Instead of hand-crafting similarity measures between adjacent superpixels, we use a binary classifier incorporating multiple proximity cues. Our encode-and-classify trees provide the most

Algorithm 1 Segmentation tree construction

```
Input: Superpixels, edges $\mathcal{E}$
Parameters: Merging rate $p$, number of levels $n_l$
  Level $l \leftarrow 1$
  Regions in first level $\mathcal{R}_l \leftarrow$ Superpixels
  while $l \leq n_l$ do
    $l \leftarrow l + 1$
    Threshold $w_{\text{th}} \leftarrow p$th percentile of weights$(\mathcal{E})$
    Merged edges $\mathcal{E}_{\text{m}} \leftarrow \{e \mid e \in \mathcal{E}, \text{weight}(e) \leq w_{\text{th}}\}$
    $\mathcal{R}_l \leftarrow \text{merge}(\mathcal{R}_{l-1}, \mathcal{E}_{\text{m}})$
    $\mathcal{E} \leftarrow \text{update}(\mathcal{E}, \mathcal{R}_l)$
  end while
Output: Tree $\{\mathcal{R}_l \mid l \in \{0 \ldots n_l\}\}$
```

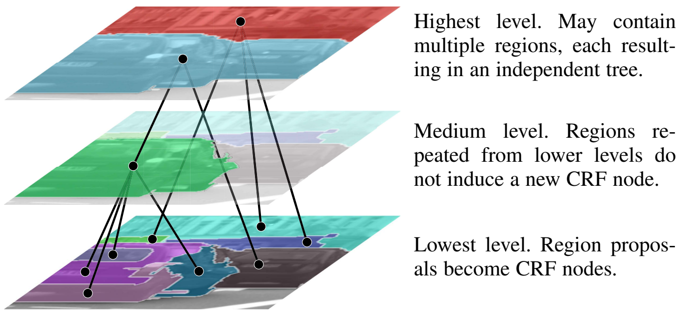

Highest level. May contain multiple regions, each resulting in an independent tree.
Medium level. Regions repeated from lower levels do not induce a new CRF node.
Lowest level. Region proposals become CRF nodes.

Fig. 4. Segmentation tree and resulting CRF model. Regions are encoded by false colors, CRF nodes by black dots, and parent-child-relations by lines.

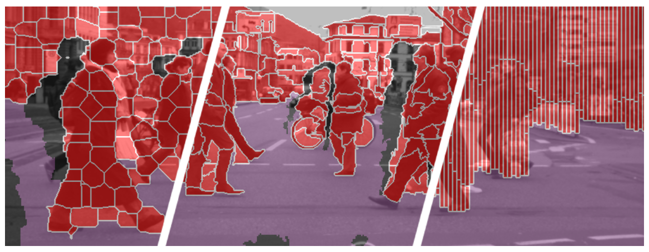


Fig. 5. Three superpixel variants we employ at the lowest level of our region proposal tree. From left to right: SEOF [10], GBIS [11], and Stixels [12]. In all variants, superpixels on the ground surface and sky region are already grouped (see text) and not considered in the segmentation tree. Superpixels with invalid depth information are ignored and visualized transparently.

important similarity measures: texton histogram intersection and pixel classifier label agreement. In addition, we employ features based on *(1)* 2D region geometry, *i.e.* spatial proximity, relative location, common boundary length, relative size difference; *(2)* depth, *i.e.* relative depth difference, absolute depth, spatial proximity in world coordinates, height above ground; *(3)* object detectors, *i.e.* overlap with vehicle and pedestrian detector bounding boxes; and *(4)* point tracks, *i.e.* velocity difference and average. Please refer to the supplementary material for a full list of features. Using those features, our binary classifier learns to group or separate pairs of superpixels. This task is related to the boundary strength classifier in [4], the same-label classifier in [33], or classifier-based pairwise potentials in CRFs, *e.g.* [32]. Subsequently, we assign the classifier scores as weights to edges connecting adjacent superpixels. The segmentation tree is constructed using Algorithm 1, which only requires two parameters, the merging rate p and the number of levels n_l. Both parameters are independent of the choice or number of superpixels. As an example, three layers of our segmentation tree are visualized in Fig. 4.

3.3 Superpixels

Superpixels are often considered as the smallest unit for scene labeling. This allows to extract rich features and efficiently model long-range dependencies between pixels. To demonstrate the generality of our region proposal tree in Sec. 3.2, we use three different kinds of superpixels: superpixels in an energy optimization framework (SEOF) [10], graph based image segmentation (GBIS) [11], and Stixels [12]. To bring all three superpixel variants to a comparable level, we introduce modifications as described in the following.

Incorporating depth. As Stixels use depth information, we also add depth to SEOF and GBIS. Both superpixel variants are appealing for a multi-cue setup, since they explicitly formulate a weight between two adjacent pixels. We extend these weights to use the disparity image $\boldsymbol{\mathcal{D}}$ in addition to the gray value image $\boldsymbol{\mathcal{I}}$ for obtaining a segmentation that accurately captures both kinds of edges. For GBIS, we define the weight between two adjacent pixels $(p,q) \in \mathcal{N}$ as

$$w_{pq} = \frac{1}{\mu_i} \left|\boldsymbol{\mathcal{I}}(p) - \boldsymbol{\mathcal{I}}(q)\right| + \frac{1}{\mu_d} \left|\boldsymbol{\mathcal{D}}(p) - \boldsymbol{\mathcal{D}}(q)\right| \ . \quad (1)$$

The normalization terms μ_i and μ_d are computed as the average absolute differences of all adjacent pixels and balance both channels against each other. For SEOF the weights are defined similarly as

$$w_{pq} = e^{-\frac{(\boldsymbol{\mathcal{I}}(p)-\boldsymbol{\mathcal{I}}(q))^2}{2\sigma_i^2}} + e^{-\frac{(\boldsymbol{\mathcal{D}}(p)-\boldsymbol{\mathcal{D}}(q))^2}{2\sigma_d^2}} \ . \quad (2)$$

Analogously, the normalization terms σ_i and σ_d are the average squared differences of all adjacent pixels.

Finally, the median disparity is assigned to each superpixel. Superpixels without valid disparity measurements due to stereo occlusions are labeled as *void* and removed from further processing, *c.f.* the transparent areas in Fig. 5.

Incorporating pixel classification. Stixel superpixels are extracted from depth information only and do not take into account gray value or color information. To also make Stixels comparable to GBIS and SEOF, we alter the computation to use the pixel classification scores as an additional data term, as presented in [34]. These modifications help especially in regions with weak disparity information, *e.g.* the sky.

Since the Stixel representation explicitly separates ground and sky regions, we also do so for GBIS and SEOF. Specifically, we first compute the average score of our pixel classifier in each superpixel. Superpixels for which *sky* or *ground* have maximal average classification score are removed from further consideration. Consequently, all three variants deliver a comparable representation, visualized in Fig. 5, where the remaining *obstacle* superpixels are highlighted in red. The segmentation tree as introduced in Sec. 3.2 is constructed from these *obstacle* superpixels only.

3.4 Region classification and CRF inference

A conditional random field (CRF) is used to jointly infer the labels $\boldsymbol{y} \in \mathcal{L}^n$ of all n regions in the segmentation tree. Each region corresponds to a node Y_i in the CRF with the label space $\mathcal{L}$ consisting of the sought after classes plus an additional *void* label. If a parent region has a single child only, both, child and parent regions contain the same information. Thus, the parent node is excluded from the CRF, see Fig. 4. Given all input data $\boldsymbol{x}$, the posterior probability is defined as

$$P(\boldsymbol{Y} = \boldsymbol{y} \mid \boldsymbol{X} = \boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x})} \prod_{i=1}^{n} \Phi_i(y_i) \prod_{(c,p)\in\mathcal{A}} \Psi_c(y_c, y_p) \ , \quad (3)$$

where $\mathcal{A}$ denotes the set of parent-child relations as defined by the segmentation tree and $Z(\boldsymbol{x})$ the partition function. In the following, the unary potentials Φ_i, the parent-child potentials Ψ_c, and the inference scheme are described.

Unaries. For each region i in the segmentation tree, we obtain three different classifier scores. Each is normalized to 1 and augmented such that the *void* label has a score of $|\mathcal{L}|^{-1}$. First, we average the RDF pixel classification results within the region, yielding s_{PC}. Second, we classify the aggregated texton histograms to obtain s_{TH}. Third, we build a multi-cue classifier providing s_{MC}. This classifier exploits features that can be efficiently extracted using the region's tree structure and are based on: *(1)* 2D region geometry, *i.e.* bounding box aspect ratio, type of boundary pixels; *(2)* depth, *i.e.* height above ground, bounding box extent in world coordinates, occlusion strength; *(3)* detectors, *i.e.* overlap with vehicle and pedestrian detector bounding boxes; and *(4)* point tracks, *i.e.* average velocity. Details are given in the supplementary material.

Finally, we interpret the scores on the pixel level and compute the region likelihood as the product of all pixel scores. Let n_i denote the number of pixels in region i. Then, the unary is defined as

$$\Phi_i(y_i) = \left(s_{\text{PC}}(y_i)^{\lambda_{\text{PC}}}\, s_{\text{TH}}(y_i)^{\lambda_{\text{TH}}}\, s_{\text{MC}}(y_i)^{\lambda_{\text{MC}}}\right)^{n_i} , \quad (4)$$

where the (learned) weights λ_{PC}, λ_{TH}, and λ_{MC} capture the different reliabilities of the individual classifiers.

Parent-child. Smoothness priors are expressed using factors between parent p and child nodes c, defined as

$$\Psi_c(y_c, y_p) = \begin{cases} 1 & \text{if } y_c = y_p \text{ or } y_p = \textit{void} \\ e^{-n_c} & \text{otherwise} \end{cases} . \quad (5)$$

Again, n_c denotes the number of pixels in the child's region. The influence of this factor is similar to Robust P^N potentials [35] and expresses our expectation that parent and child nodes often belong to the same class. However, if a node is likely to contain multiple classes, it can be assigned to *void* and does not influence its child nodes anymore. Further, a small node may have a different label than the parent, even if the parent's label is meaningful, *e.g.* a pedestrian in front of a large building.

Inference. Since Eq. (3) is tree-structured, running sum-product belief propagation is very efficient and provides exact marginals $P(Y_i = y_i \mid \boldsymbol{X} = \boldsymbol{x})$ for all nodes i. For all n_i pixels in a superpixel-level node i, we assign $P(Y_i = y_i \mid \boldsymbol{X} = \boldsymbol{x})^{\frac{1}{n_i}}$ as marginal posteriors.

3.5 Temporal regularization

To obtain a consistent labeling over time, we again employ the point tracks and feed the pixel-level CRF marginals into a time-recursive regularization scheme [6]. The filtered posteriors are assigned back to superpixels by averaging over all covered point tracks. The final labeling is obtained as the label maximizing the marginal posterior, and assigned to all pixels in each superpixel.

4 Training and Testing

For all RDFs involved, we follow [36] to build extremely randomized binary trees. Training the pixel-level encode-and-classify trees is performed in two steps. First, we generate unary pixel tests within the small $S_1 \times S_1$ patch around a pixel of interest. We train all trees to full depth and then prune them back bottom up to the desired number of encoder nodes, which gives us full control over the histogram length. In each pruning iteration, we randomly select a node from the list of all candidates for pruning, where a candidate is a split node with both children being leafs, and prune its children, so that the split node becomes a new leaf. Second, we start at the encoder nodes and continue training with access to the larger $S_2 \times S_2$ patches to improve the pixel-level labeling performance. At test time, we apply the trees to every pixel on a regular grid with three pixels distance to reduce runtime. During traversal of each tree, the unique ID of the passed encoder node is stored in a tree-specific texton map. For labeling, we calculate the average leaf node posterior distribution over all trees.

Next, we use a binary RDF classifier for segmentation tree construction, where the objective is to decide whether or not two adjacent superpixels should be merged to a larger region. For training we treat all adjacent superpixels with identical majority ground truth label as positive samples and other pairs as negative. If both superpixels have the *void* label assigned, they do not contribute as training sample.

Of the three region-level scores we employ in Sec. 3.4, s_{PC} is directly provided by the averaged pixel-level label decision. For the other two scores, s_{TH} and s_{MC}, we train additional classifiers at the region level. We use a one-*vs*-all histogram intersection kernel SVM for s_{TH} (due to its superior performance on histogram features) and perform a sigmoid mapping to convert the SVM output to a positive score. For the s_{MC} score, we train an RDF classifier using our multi-cue features. In both cases, we compute the segmentation tree on images in the training set and use all regions in the tree with a ground truth overlap of $50\,\%$ or more as region masks to extract the features. In all our experiments, we construct the segmentation tree with a merging rate of $p = 0.2$ until we obtain $n_l = 10$ levels.

The CRF weights λ_{PC}, λ_{TH}, and λ_{MC} are optimized using grid search on the training set with the Pascal VOC intersection-over-union (IoU) score [1] as objective function.

5 Evaluation

We use two public datasets for our experimental evaluation, the Daimler Urban Segmentation Dataset (DUS) [6] and KITTI [5]. Both provide stereo image pairs and intermediate frames for motion cues. The DUS dataset contains 500 images with pixel-level semantic class annotations. For KITTI, we collect annotations provided alongside previous publications [9], [37], [38], [39], [40]. We report numbers on all 216 annotated images provided for the visual odometry challenge and use the remaining 211 images for training.

5.1 Cues

The main cues of our approach as depicted in Fig. 2, pixel classification (PC) and texton histograms (TH), have already been described. The remaining cues are briefly outlined in the following. Note that our approach is independent of the particular algorithm used. Our choice is mainly motivated by balancing the trade-off between quality and runtime.

Depth (D). To integrate depth information, we use dense disparity maps computed using semi-global matching.

Point tracks (PT). Motion cues are integrated in terms of KLT feature tracks [41], which give a set of sparse but reliable long-term point trajectories. With the odometry information provided in the dataset, motion induced by camera movement is compensated using a Kalman filter, which provides an estimate of the 3D velocity of observed obstacles. Motion cues are used for grouping and classification of region proposals. Additionally, the tracks are

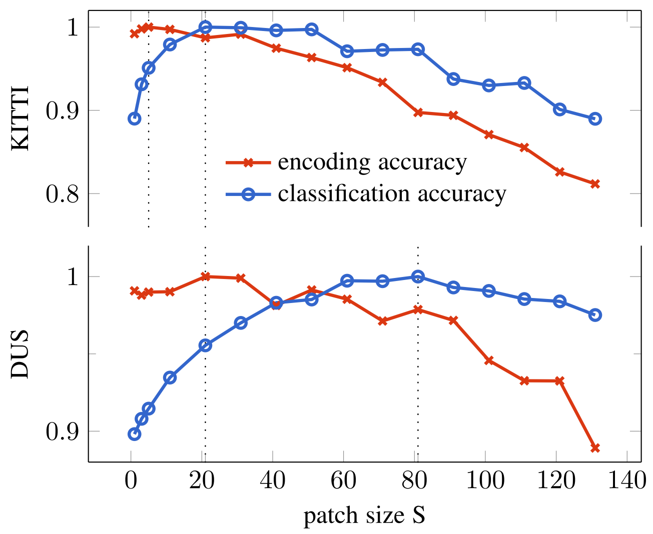


Fig. 6. Encoding accuracy (red) and pixel classification accuracy (blue) as a function of patch size. Note that each curve is normalized such that its maximum value is 1. We observe that smaller patches are better for encoding, while medium to large patches are good for pixel classification. This trend is consistent on both datasets: Daimler Urban Segmentation (DUS) and KITTI. The maximizing patch sizes are highlighted for each curve.

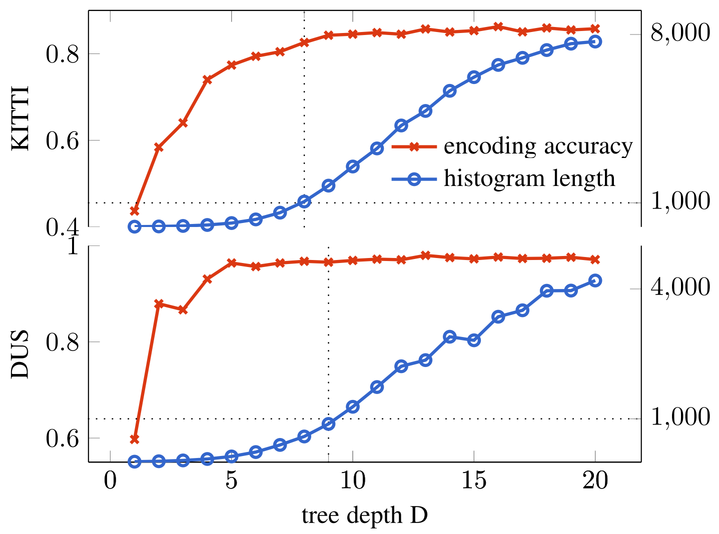


Fig. 7. Encoding accuracy (red, left axis) and histogram length (blue, right axis) as a function of tree depth. It can be seen that with increased depth encoding accuracy saturates rather quickly, but the histogram length and thus histogram classification runtime grows continuously. This shows that limiting the number of encoder nodes, *i.e.* the histogram length, is important to set the optimal trade-off between accuracy and runtime. We set the histogram length to 1000, as highlighted.

also used to stabilize the final labeling over time, as described in Sec. 3.5.

Detectors (DT). We employ object detectors for pedestrians and vehicles given that they are the most prominent dynamic objects in street scenes. We use a two-stage detection system for both classes, consisting of a fast Viola-Jones cascade [42] coupled with a three-layer convolutional neural network [43] as an additional verification module. Multiple detections across location and scale are addressed using mean shift-based non-maximum suppression.

5.2 Encode-and-classify trees

In Fig. 6, we demonstrate the different demands on patch size for pixel classification and region encoding, by training separate standard RDF models for the two tasks. To assess region encoding accuracy, we generate and classify histograms on the segment level. As segments we use ground truth regions in the dataset, as well as segments obtained by greedily grouping superpixels using 3D proximity. Performance is reported using the average F-score over all classes. We normalized both curves to a maximum value of 1, as only the trends are important here. It can be seen that the highlighted maxima of both curves do not coincide, hence the maxima cannot both be obtained at the same time using a single standard RDF. Executing two separate models with different patch sizes would overcome this problem, however at the cost of additional runtime. In contrast, a single forest of our encode-and-classify trees with the restricted patch size $S_1 < S_2$ yields a performance close to both maxima, while having the same runtime as a single standard RDF as shown in *c.f.* Table 1. Despite the small remaining gap in accuracy compared to using two separate forests, we did not observe a drop of performance in the overall system, while affording an increased efficiency.

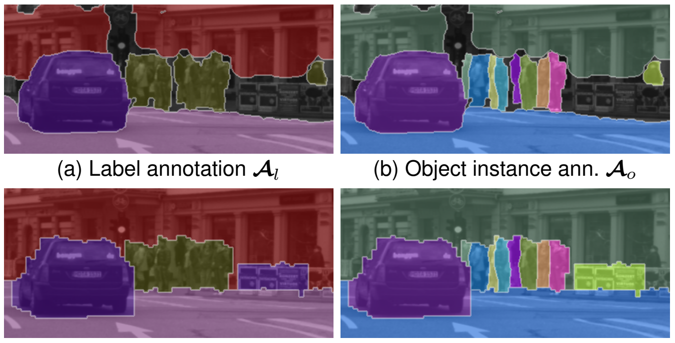
In Fig. 7, we further investigate the trade-off between encoding accuracy and histogram length by plotting the region classification average F-score over all classes together with histogram length as a function of tree depth D. We use the length as a proxy metric for runtime, as the runtime of our SVM classifier scales linearly with histogram dimensionality. Furthermore, convergence of the histogram length also indicates saturation of pixel-level classification accuracy, as the trees have eventually solved the training set completely. It is interesting to see that on both datasets the region-level classification accuracy saturates rather quickly, while the histogram length continues to grow, until it also starts to saturate, but much later. This finding supports our claim that high pixel-level classification accuracy requires deeply trained trees, while shallow trees with less leaf nodes are sufficient for texton extraction and also keep histogram classification runtime down. Our encode-and-classify concept combines both objectives in a single model, where only a sub-tree of an entire decision tree is used for texton extraction, and the number of encoder nodes can be chosen freely and independently of the remaining tree parameters.

In all our experiments, we use 5 trees in total with 200

TABLE 1. Standard RDFs and encode-and-classify (E&C) trees compared on the DUS dataset with different patch size combinations. Accuracy is given as average F-score, relative to the maximum possible encoding and classification performance, according to Fig. 6. Using our E&C trees with a smaller patch size S_1 for encoding and a larger one for classification S_2 yields a performance close to the individual maxima of standard RDFs. At the same time, they are more efficient compared to using two separate standard models for encoding and classification (center column).

	standard RDF			E&C trees	
Patch size S_1/S_2	21^a	81^a	$21/81^b$	21/81	31/81
Encoding acc.	1.0	.979	1.0	1.0	.999
Classif. acc.	.955	1.0	1.0	.97	.987
Runtime [ms]	8	8	14	8	8

[a] Single model with one patch size for both encoding and classification
[b] Two separate models with different patch sizes for encoding and classification


(a) Label annotation $\mathcal{A}_l$ (b) Object instance ann. $\mathcal{A}_o$

(c) Label prediction $\mathcal{P}_l$ (d) Object instance pred. $\mathcal{P}_o$

Fig. 8. Overview of different annotation and prediction formats. Object instance prediction (d) is generated from Algorithm 2. Note the false positive *vehicle* prediction to the right of the pedestrians.

encoder nodes each, resulting in a 1000-dimensional region-level histogram, *c.f.* Fig. 7. As feature channels we employ *(1)* the raw gray scale image, *(2)* a 15-dimensional Haar wavelet transform [44], *(3)* the 3D height above ground, *(4)* the vertical disparity gradient image, and *(5)* normalized color channels (KITTI only). According to the maxima in Fig. 6 and the results in Table 1, we choose the patch sizes as $S_1/S_2 = 31/81$ for DUS and $S_1/S_2 = 5/21$ for KITTI.

5.3 Object-centric evaluation

The Pascal VOC intersection-over-union (IoU) score [1] is a common metric for evaluating scene labeling and is also the preferred evaluation method on the DUS dataset. However, we do not consider this score alone to be sufficient for the high range of scales of objects in urban scenes. Since the IoU score is based on counting pixels, it is dominated by objects at a large scale, whereas smaller ones have only a minor impact. However, for most practical applications, a metric that answers the question of how well individual objects are represented in the scene is desirable. Therefore, we complement the IoU score with an object-centric evaluation, *c.f.* [45].

Algorithm 2 Generate object instances from label prediction. These are used for an evaluation at the instance level.

```
Input: Label prediction $\mathcal{P}_l$, Object instance annotation $\mathcal{A}_o$
  Allocate searched object instance prediction $\mathcal{P}_o$
  $\mathcal{R}_p \leftarrow \text{connectedRegions}(\mathcal{P}_l)$
  for all regions $R \in \mathcal{R}_p$ do
      $l \leftarrow \text{label}(R)$
      // Find all candidate object instances for the region:
      $\mathcal{C}_o \leftarrow \{o \,|\, o \in R(\mathcal{A}_o) \wedge \text{label}(o) = l\}$
      if $\mathcal{C}_o = \emptyset$ then
          $\mathcal{P}_o(R) \leftarrow$ new object instance (false positive)
      else
          for all pixels $p \in R$ do
              // Find closest candidate object instance for $p$:
              $\mathcal{P}_o(p) \leftarrow \operatorname{argmin}_{o \in \mathcal{C}_o} \min_{\substack{\text{pixel } q \in \mathcal{A}_o\text{, with} \\ \text{label}(q)=\text{label}(o)}} \text{dist}(p, q)$
          end for
      end if
  end for
Output: $\mathcal{P}_o$
```

To obtain such a metric, we divided the *vehicle* and *pedestrian* annotations for the DUS dataset into individual instance labels that will be made publicly available. Since our work aims to perform scene labeling, it does not produce object instance predictions. Yet, we argue that it is beneficial to assess the scene labeling performance by taking instances into account.

To that end, we define an evaluation protocol, where we create artificial instances given the pixel-level labeling output, see Fig. 8 for an example and Algorithm 2 for details. This procedure allows for an interpretation of the pixel-level output in terms of true positive, false positive or false negative instances. Therefore, the contribution of per-pixel errors can be normalized with the scale of the instance and the evaluation protocol allows instance-aware evaluation without instance-level prediction.

A recently introduced metric to evaluate instance-level predictions is the AP^r score from [49]. We follow this idea and compute the IoU individually for each of our artificial instance predictions. These overlaps are then thresholded at 0.5. In [49] a precision-recall curve is plotted by varying a threshold on the prediction likelihood. Since we do not have such likelihoods here, we are restricted to a single operating point on this curve, for which we report the F-score.

5.4 Results

On the DUS dataset, we compare our method to five baselines. The authors either published their performance [6], [48] or have code available [28], [46], [47]. For [47], we use the pairwise results from the *multiSeg* example. The numbers for [28] are generated with C# code provided by Matthew Johnson, one of the co-authors of [28] and represent the final result of their two step approach. Compared to this baseline, which is most closely related to our encode-and-classify concept, we significantly improve labeling accuracy. For all experiments we conduct with public software, we use the default parameters set in the code. Links to the used software packages are provided with the respective reference. The authors of [6], [48] kindly provided their inferred results to allow for an evaluation with our novel object-centric metric. Figure 9 shows qualitative results of our approach and the best baseline [46]. For the KITTI dataset, there are no reported results available that make use of all annotated images. Therefore, we only compare to [28], [46], [47].

Tables 2 and 3 clearly indicate that our approach outperforms the baselines on both datasets regarding scene labeling quality. In particular, traffic participants (*vehicle*, *pedestrian*) are recognized with superior performance in terms of pixel and object accuracy, mostly as a result of incorporating the additional object detectors at multiple levels in our system; on KITTI our approach outperforms the baselines by a margin of $30\,\%$. We further observe that the results based on Stixels outperform GBIS, which in turn outperform those based on SEOF. We attribute this to the main properties of the individual superpixel variants. Stixels are specifically designed for street scenes, whereas GBIS and SEOF are more generic.

In addition to those three superpixel variants, we also show results when using the *gPb*-owt-ucm segmentation tree from [19]. This method does not take depth into account, but is still highly competitive and has been used as the basis for the segmentation trees in [4] and [7], which we consider as related to our approach. For the *UCM SPs* column in Table 2, we chose a single threshold in the ucm hierarchy to extract superpixels with a size similar to

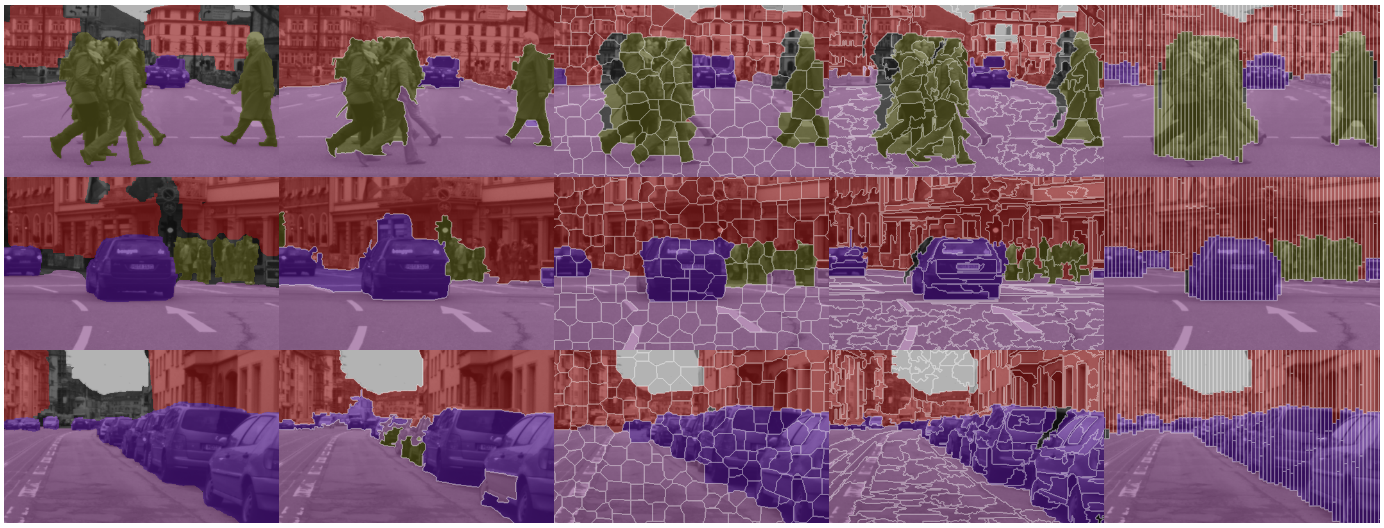


Fig. 9. Qualitative results on the DUS dataset compared to the ground truth and the best performing baseline. From left to right: ground truth, baseline [46], and our results for SEOF, GBIS and Stixels superpixels.

TABLE 2. Quantitative results on the DUS dataset with official train/test split compared to five baselines (left). We show pixel accuracy (IoU), object accuracy (F-score), and runtime. Additionally, we report results using Stixels, where detections (-DT), point tracks (-PT), and their combination (-DT, -PT) are removed from the full system to demonstrate their influence on the overall performance (right).

	Baselines					This paper					Cues removed (Stixels)		
	[28]	[47]	[6]	[48]	[46]	UCM Tree	UCM SPs	SEOF	GBIS	Stixels	-DT	-PT	-DT, -PT
IoU[a] [%]	70.1	73.5	80.6	84.5	**86.0**	81.8	80.7	83.5	84.3	85.7	84.2	84.9	82.6
IoU[b] [%]	58.3	44.9	72.4	73.8	74.5	79.6	**81.0**	78.4	79.1	79.9	75.9	78.8	73.2
F-score[b] [%]	53.3	65.1	81.4	85.1	83.8	82.8	79.7	79.6	81.2	**86.4**	82.7	84.5	80.4
Runtime [ms]	**125**	5200	150[c]	2800	10^5	10^4	10^4	10^4	410	163	118	138	93

[a] Average over all classes
[b] Average over dynamic objects: *vehicle*, *pedestrian*
[c] This also includes computation time for stereo and Stixels, which are neglected in the runtimes reported in [6].

the other superpixel variants and compute our hierarchy on top of it. For the *UCM Tree* column, we choose 10 representative thresholds to remove our hierarchy generation completely from the system and use the ucm tree instead. We find that performance drops drop when using the ucm tree, which shows the benefit of our proposed multi-cue segmentation tree.

For a second experiment, we limit ourselves to using Stixels as superpixels and evaluate the contributions of the external cues introduced in Sec. 5.1, *i.e.* detectors and point tracks. If a single cue is omitted, the overall performance decreases, but our method is still competitive to all baselines. As soon as both cues are left out, performance drops more significantly.

Runtime is evaluated using an Intel Core i7 CPU and a NVIDIA GTX 770 GPU. We report the total amount of time spent to label a single image including the computation of all cues in Table 2. For depth, we assume 50 ms runtime as reported for the dataset [6]. As the reference implementation of [28] is not tuned for fast execution, we quote the optimized timing results originally reported in [28]. In summary, our system has a faster parsing time compared to most baselines. Only [6] and [28] are slightly faster, but at the cost of a large drop in scene labeling quality. In Fig. 10, we break down the overall system runtime of our method with Stixel superpixels into individual component runtimes, to demonstrate the efficiency of the tree-structured models we focus on in this work. The combined execution time of our encode-and-classify trees and our multi-cue segmentation tree (construction, classification, and inference) takes up less than 10% of the overall

TABLE 3. Quantitative results on the KITTI dataset compared to three baselines.

	Baselines			This paper
	[28]	[47]	[46]	Stixels
IoU[a] [%]	53.7	70.7	73.9	**75.8**
IoU[b] [%]	27.1	41.9	50.4	**65.2**

[a] Average over all classes
[b] Average over dynamic objects: *vehicle*, *pedestrian*

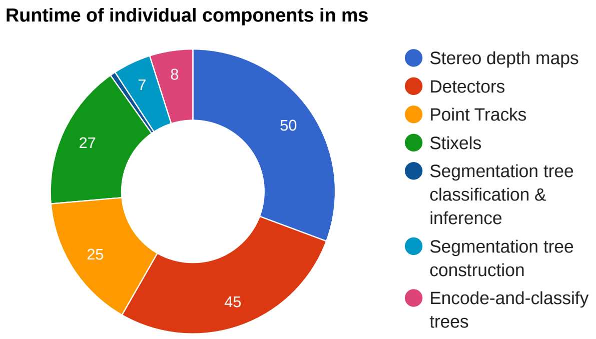


Fig. 10. Breakdown of the 163 ms runtime reported with Stixel superpixels in Table 2 into the individual components of our overall system. The two tree-structured models we focus on in this work, *i.e.* encode-and-classify trees and the multi-cue segmentation tree, only use about 10% of the overall time, clearly indicating its efficiency.

runtime, which clearly indicates its efficiency. We can further reduce the overall runtime in an online setup by pipelining the components. In doing so, we achieve a throughput of 20 fps at the cost of one frame delay, as no individual component takes longer than 50 ms, and stereo depth maps, detectors, and point tracks can be computed in parallel.

Conceptually, our approach combines the central ideas from the two most runtime efficient methods [6], [28] and extends them with tree-structured CRF inference, multiple cues and the encode-and-classify concept. In this way, we are able to significantly improve labeling accuracy compared to these methods, while maintaining fast inference time.

6 Conclusion

In this paper, we proposed a system for urban scene labeling based on two tree-structured models: encode-and-classify trees at the pixel level and a multi-cue segmentation tree at the superpixel level. Both tree structures are tightly coupled to achieve high labeling performance while being computationally efficient at the same time. Most prominently, we increase the labeling performance for the two most relevant and challenging classes (vehicles and pedestrians) by a significant margin despite near real-time speeds. We take this as evidence for the suitability of our coupled trees and the holistic integration of multiple cues.

References

[1] 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," *IJCV*, vol. 111, no. 1, pp. 96–136, 2014.

[2] 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," *IJCV*, vol. 115, no. 3, pp. 211–252, 2015.

[3] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common objects in context," in *ECCV*, 2014.

[4] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *ECCV*, 2012.

[5] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *IJRR*, vol. 32, no. 11, pp. 1231–1237, 2013.

[6] T. Scharwächter, M. Enzweiler, U. Franke, and S. Roth, "Stixmantics: A medium-level model for real-time semantic scene understanding," in *ECCV*, 2014.

[7] S. Gupta, R. B. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *ECCV*, 2014.

[8] J. Yao, S. Fidler, and R. Urtasun, "Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation," in *CVPR*, 2012.

[9] S. Sengupta, E. Greveson, A. Shahrokni, and P. H. S. Torr, "Urban 3D semantic modelling using stereo vision," in *ICRA*, 2013.

[10] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *ECCV*, 2010.

[11] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *IJCV*, vol. 59, no. 2, pp. 167–181, 2004.

[12] D. Pfeiffer and U. Franke, "Towards a global optimal multi-layer Stixel representation of dense 3D data," in *BMVC*, 2011.

[13] P. Arbeláez, B. Hariharan, and C. Gu, "Semantic segmentation using regions and parts," in *CVPR*, 2012.

[14] J. Lim, P. Arbeláez, C. Gu, and J. Malik, "Context by region ancestry," in *ICCV*, 2009.

[15] X. Ren, L. Bo, and D. Fox, "RGB-(D) scene labeling: Features and algorithms," in *CVPR*, 2012.

[16] C. Farabet, C. Couprie, L. Najman, and Y. Lecun, "Learning hierarchical features for scene labeling," *Trans. PAMI*, vol. 35, no. 8, pp. 1915–1929, 2012.

[17] V. Lempitsky, A. Vedaldi, and A. Zisserman, "A pylon model for semantic segmentation," *NIPS*, vol. 24, 2011.

[18] S. Nowozin, P. Gehler, and C. H. Lampert, "On parameter learning in CRF-based approaches to object class image segmentation," in *ECCV*, 2010.

[19] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *Trans. PAMI*, vol. 33, no. 5, pp. 898–916, 2011.

[20] J. Reynolds and K. Murphy, "Figure-ground segmentation using a hierarchical conditional random field," in *CRV*, 2007.

[21] N. Plath, M. Toussaint, and S. Nakajima, "Multi-class image segmentation using conditional random fields and global classification," in *ICML*, 2009.

[22] L. Ladický, C. Russell, P. Kohli, and P. H. S. Torr, "Associative hierarchical random fields," *Trans. PAMI*, vol. 36, no. 6, pp. 1056–1077, 2013.

[23] G. Floros and B. Leibe, "Joint 2D-3D temporally consistent semantic segmentation of street scenes," in *CVPR*, 2012.

[24] O. Kähler and I. Reid, "Efficient 3D scene labeling using fields of trees," in *ICCV*, 2013.

[25] B. Mičušík and J. Košecká, "Semantic segmentation of street scenes by superpixel co-occurrence and 3D geometry," in *ICCV Workshops*, 2009.

[26] J. Zhang, C. Kan, A. G. Schwing, and R. Urtasun, "Estimating the 3D layout of indoor scenes and its clutter from depth sensors," in *ICCV*, 2013.

[27] F. Moosmann, E. Nowak, and F. Jurie, "Randomized clustering forests for image classification," *Trans. PAMI*, vol. 30, no. 9, pp. 1632–1646, 2008.

[28] J. Shotton, M. Johnson, and R. Cipolla, "Semantic texton forests for image categorization and segmentation," in *CVPR*, 2008. [Online]. Available: http://www.6d-vision.com/scene-labeling/stf_reference_code.zip

[29] B. Triggs and J. J. Verbeek, "Scene segmentation with CRFs learned from partially labeled images," in *NIPS*, 2007.

[30] A. Ion, J. Carreira, and C. Sminchisescu, "Probabilistic joint image segmentation and labeling by figure-ground composition," *IJCV*, vol. 107, no. 1, pp. 40–57, 2014.

[31] J. M. Gonfaus, X. Boix, J. van de Weijer, A. D. Bagdanov, J. Serrat, and J. Gonzalez, "Harmony potentials for joint classification and segmentation," in *CVPR*, 2010.

[32] Q. Huang, M. Han, B. Wu, and S. Ioffe, "A hierarchical conditional random field model for labeling and segmenting images of street scenes," in *CVPR*, 2011.

[33] D. Hoiem, A. A. Efros, and M. Hebert, "Recovering surface layout from an image," *IJCV*, vol. 75, no. 1, pp. 151–172, 2007.

[34] T. Scharwächter and U. Franke, "Low-level fusion of color, texture and depth for robust road scene understanding," in *IV Symposium*, 2015.

[35] P. Kohli, L. Ladický, and P. H. S. Torr, "Robust higher order potentials for enforcing label consistency," *IJCV*, vol. 82, no. 3, pp. 302–324, 2009.

[36] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, 2006.

[37] H. He and B. Upcroft, "Nonparametric semantic segmentation for 3D street scenes," in *IROS*, 2013.

[38] L. Ladický, J. Shi, and M. Pollefeys, "Pulling things out of perspective," in *CVPR*, 2014.

[39] G. Ros, S. Ramos, M. Granados, D. Vazquez, and A. M. Lopez, "Vision-based offline-online perception paradigm for autonomous driving," in *WACV*, 2015.

[40] P. Xu, F. Davoine, J.-B. Bordes, H. Zhao, and T. Denoeux, "Information fusion on oversegmented images: An application for urban scene understanding," in *MVA*, 2013.

[41] C. Tomasi and T. Kanade, "Detection and tracking of point features," Carnegie Mellon University, Tech. Rep. CMU-CS-91-132, 1991.

[42] P. Viola and M. J. Jones, "Robust real-time object detection," *IJCV*, vol. 57, no. 2, pp. 137–154, 2004.

[43] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *Trans. PAMI*, vol. 31, no. 12, pp. 2179–2195, 2009.

[44] E. J. Stollnitz, T. D. DeRose, and D. H. Salesin, "Wavelets for computer graphics: A primer," *IEEE Computer Graphics and Applications*, vol. 15, no. 3, pp. 76–84, 1995.

[45] 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.

[46] L. Ladický, P. Sturgess, C. Russell, S. Sengupta, Y. Bastanlar, W. Clocksin, and P. H. S. Torr, "Joint optimisation for object class segmentation and dense stereo reconstruction," in *BMVC*, 2010. [Online]. Available: http://www.inf.ethz.ch/personal/ladickyl/

[47] S. Gould, "DARWIN: A framework for machine learning and computer vision research and development," *JMLR*, vol. 13, pp. 3533–3537, 2012. [Online]. Available: http://drwn.anu.edu.au/

[48] A. Sharma, O. Tuzel, and J. W. David, "Deep hierarchical parsing for semantic segmentation," in *CVPR*, 2015.

[49] B. Hariharan, P. Arbeláez, R. B. Girshick, and J. Malik, "Simultaneous detection and segmentation," in *ECCV*, 2014.

Marius Cordts received the M.Sc. degree in electrical engineering from RWTH Aachen University, Germany, in 2013. He is currently working toward the PhD degree under supervision of Prof. Stefan Roth in Technische Universität Darmstadt, Germany. During his studies he worked for a semester in the field of medical image computing for cardiovascular analysis at Siemens Corporate Research in Princeton, NJ. He received a scholarship from Studienstiftung des deutschen Volkes (German National Academic Foundation) and was supported by the Siemens Masters program. He won various prizes, such as the Friedrich-Wilhelm-Prize and the SEW Eurodrive Prize. Since graduation he is working at Daimler Research & Development, Böblingen, Germany. Since 2015, he is a full-time employee at Daimler R&D. His research focuses on efficient urban scene understanding for autonomous driving.

Timo Rehfeld received the Diploma degree in computer engineering (Technische Informatik) from RWTH Aachen University, Germany, in 2013. During his studies he spent an exchange semester at KEIO University, Tokyo, Japan, where he studied Japanese and conducted research in teleoperation robotics. After graduation he started to work toward the PhD degree with Daimler Research & Development, Böblingen, Germany, under supervision of Prof. Stefan Roth at Technische Universität Darmstadt, Germany, where his research focus was efficient scene labeling for automotive applications. He recently joined Mercedes-Benz Research & Development North America, CA, USA, to work on sensor fusion for autonomous vehicles. Over the last years, he received several awards, including the GCPR Main Paper Prize in 2013.

Markus Enzweiler received the M.Sc. degree in computer science from the University of Ulm, Germany in 2005, and the PhD degree in computer science from the University of Heidelberg, Germany, in 2011. Since 2010, he has been a research scientist with Daimler Research & Development, Böblingen, Germany. His current work focuses on scene understanding for self-driving cars. He received graduate and PhD scholarships from the Studienstiftung des deutschen Volkes (German National Academic Foundation). In 2012, he was awarded both the IEEE Intelligent Transportation Systems Society Best PhD Dissertation Award and the Uni-DAS Research Award for his work on vision-based pedestrian recognition. He was part of the Daimler team that received the IEEE Intelligent Transportation Systems Society Outstanding Application Award in 2014. Since 2014, he is a Junior-Fellow of the German Informatics Society.

Uwe Franke received the PhD degree in electrical engineering from Technical University of Aachen, Germany, in 1988, for his work on content based image coding. Since 1989, he has been with Daimler Research and Development and has been constantly working on the development of vision based driver assistance systems. He developed Daimlers lane departure warning system introduced in 2000. Since 2000, he has been head of Daimlers Image Understanding Group. The stereo technology developed by his group is the basis for the Mercedes Benz stereo camera system introduced in 2013. Recent work is on image understanding for autonomous driving. He was nominated for the *Deutscher Zukunftspreis*, Germany's most important award for Technology and Innovation given by the German President and awarded the Karl-Heinz Beckurts Prize 2012.

Stefan Roth received the Diploma degree in computer science and engineering from the University of Mannheim, Germany, in 2001. He received the ScM degree in computer science and the PhD degree in computer science from Brown University, in 2003 and 2007, respectively. Since 2007, he is on the faculty of computer science at Technische Universität Darmstadt, Germany (Juniorprofessor 2007-2013, Professor since 2013). His research interests include probabilistic and statistical approaches to image modeling, motion estimation, human tracking, and object recognition. He received several awards, including honorable mentions for the Marr Prize at ICCV 2005 (with M. Black) and ICCV 2013 (with C. Vogel and K. Schindler), the Olympus-Prize 2010 of the German Association for Pattern Recognition (DAGM), and the Heinz Maier-Leibnitz Prize 2012 of the German Research Foundation (DFG). He served as an area chair for ICCV 2011, ECCV 2012, 2014 & 2016, and CVPR 2013, and is member of the editorial board of the *International Journal of Computer Vision (IJCV)*, the *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, and *PeerJ Computer Science*.