Segmentation and Recognition using Structure from Motion Point Clouds

Gabriel J. Brostow[1], Jamie Shotton[2], Julien Fauqueur[3], and Roberto Cipolla[4]

[1] University College London and ETH Zurich
[2] Microsoft Research Cambridge
[3] University of Cambridge (now with MirriAd Ltd.)
[4] University of Cambridge

Abstract. We propose an algorithm for semantic segmentation based on 3D point clouds derived from ego-motion. We motivate five simple cues designed to model specific patterns of motion and 3D world structure that vary with object category. We introduce features that project the 3D cues back to the 2D image plane while modeling spatial layout and context. A randomized decision forest combines many such features to achieve a coherent 2D segmentation and recognize the object categories present. Our main contribution is to show how semantic segmentation is possible based *solely on motion-derived 3D world structure*. Our method works well on sparse, noisy point clouds, and unlike existing approaches, does not need appearance-based descriptors.
Experiments were performed on a challenging new video database containing sequences filmed from a moving car in daylight and at dusk. The results confirm that indeed, accurate segmentation and recognition are possible using only motion and 3D world structure. Further, we show that the motion-derived information complements an existing state-of-the-art appearance-based method, improving both qualitative and quantitative performance.

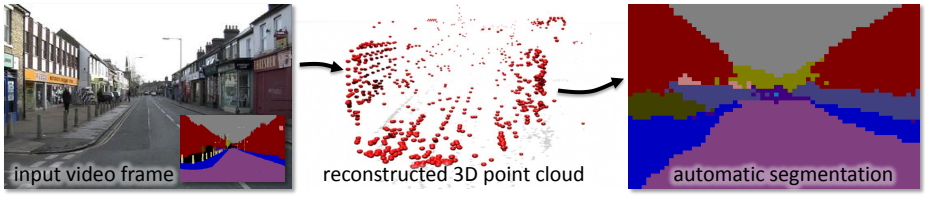


Fig. 1. The proposed algorithm uses 3D point clouds estimated from videos such as the pictured driving sequence (with ground truth inset). Having trained on point clouds from other driving sequences, our new motion and structure features, based purely on the point cloud, perform 11-class semantic segmentation of each test frame. The colors in the ground truth and inferred segmentation indicate category labels.

1 Introduction

We address the question of whether motion and 3D world structure can be used to accurately segment video frames and recognize the object categories present. In particular, as illustrated in Fig. 1, we investigate how to perform semantic segmentation from the sparse, noisy 3D point cloud given by structure from ego-motion. Our algorithm is able to accurately recognize objects and segment video frames without appearance-based descriptors or dense depth estimates obtained using *e.g.,* dense stereo or laser range finders. The structure from motion, or SfM, community [1] has demonstrated the value of ego-motion derived data, and their modeling efforts have even extend to stationary geometry of cities [2]. However, the *object recognition* opportunities presented by the inferred motion and structure have largely been ignored[1].

The proposed algorithm uses camera-pose estimation from video as an existing component, and assumes ego-motion is the dominant cause of pixel flow [4]. Tracked 2D image features are triangulated to find their position in world space and their relationship to the moving camera path. We suggest five simple motion and structure cues that are indicative of object categories present in the scene. Projecting these cues from the 3D point cloud to the 2D image, we build a randomized decision forest classifier to perform a coherent semantic segmentation.

Our main contributions are: (i) a demonstration that semantic segmentation is possible based *solely on motion-derived 3D world structure*; (ii) five intuitive motion and structure cues and a mechanism for projecting these 3D cues to the 2D image plane for semantic segmentation; and (iii) a challenging new database of video sequences filmed from a moving car and hand-labeled with ground-truth semantic segmentations. Our evaluation shows performance comparable to existing state-of-the-art appearance based techniques, and further, that our motion-derived features complement appearance-based features, improving both qualitative and quantitative performance.

Background. An accurate automatic scene understanding of images and videos has been an enduring goal of computer vision, with applications varying from image search to driving safety. Many successful techniques for 2D object recognition have used individual still images [5–7]. Without using SfM, Hoiem et al. [8, 9] achieve exciting results by considering several spatial cues found in single images, such as surface orientations and vanishing points, to infer the camera viewpoint or general scene structure. This, in turn, helps object recognition algorithms refine their hypotheses, culling spatially infeasible detections. 3D object recognition is still a new research area. Huber et al.[10] matched laser rangefinder data to learned object models. Other techniques build 3D object models and match them to still images using local descriptors [11–14]. None of these methods, however, can exploit the motion-based cues available in video sequences. Dalal et al. [15] is a notable exception that used differential optical flow in pairs

[1] The work of [3] was similarly motivated, and used laser-scans of static scenes to compute a 3D planar patch feature, which helped to train a chain of binary classifiers.

of images. In this paper, we reason about the moving 3D scene given a moving 2D camera. Our method works well on sparse, noisy point clouds, and does not need appearance-based descriptors attached to 3D world points.

There is a long history of fascinating research about motion-based recognition of human activities [16]. Laptev and Lindeberg [17] introduced the notion of space-time interest points to help detect and represent sudden actions as high gradient points in the xyt cube for motion-based activity recognition. Our focus is rather object recognition, and our features do not require a stationary camera.

While it is tempting to apply other detectors (*e.g.,* pedestrians [18]) directly to the problem of recognizing objects from a moving camera, motion compensation and motion segmentation are still relatively open problems. Yin et al. [19] use low-level motion cues for bi-layer video segmentation, though do not achieve a semantic labeling. Computer vision for driving has proven challenging and has previously been investigated with a related focus on motion segmentation [20]. For example, Kang et al. [21] have recently shown an improvement in the state of the art while using a structure consistency constraint similar to one of our motion cues. Leibe et al. [22] address recognition of cars and pedestrians from a moving vehicle. Our technique handles both these and nine further categories, and additionally semantically segments the image, without requiring their expensive stereo setup.

Optical flow has aided recognition of objects for static cameras [23], but forward ego-motion dominates the visual changes in our footage. Depth-specific motion compensation may help, but requires accurate dense-stereo reconstruction or laser range-scanning. We instead employ features based on a sparse SfM point cloud and avoid these problems.

2 Structure from Motion Point Clouds

We use standard structure from ego-motion techniques to automatically generate a 3D point cloud from video sequences filmed from moving cars. The dominant motion in the sequences gives the camera world-pose and thereby the relative 3D point cloud of all tracked 2D features, including outliers.

We start by tracking 2D image features. Specifically, we use Harris-Stephens corners [24] with localized normalized cross correlation to track 20×20 pixel patches through time in a search window 15% of the image dimensions. In practice, this produced reliable 2D trajectories that usually spanned more than 5 frames. To reduce the number of mis-tracks, each initial template is tracked only until its correlation falls below 0.97.

Footage is obtained from a car-mounted camera. We assume, for purposes of 3D reconstruction, that changes between images are the result of only ego-motion. This allows us to compute a single world-point $W = (x, y, z, 1)^T$ for each point tracked in 2D image space, (u_t, v_t). A best-fit $\tilde{W}$ is computed given at least two corresponding 3x4 camera projection matrices P_t from the sequence. Matrices P are inferred in a robust pre-processing stage, for which we simply use a commercial product [4], which normalizes the resulting up-to-scale solutions

to 1.0. Then P is split into row vectors $p_{1:3}$, so W projects into the camera C_t as $[u_1, v_1]^T \equiv [\tilde{u}_1, \tilde{v}_1, \lambda]^T = [p_1, p_2, p_3]^T [x, y, z]^T$, and dividing through by λ gives $u_1 = \frac{p_1 W}{p_3 W}, v_1 = \frac{p_2 W}{p_3 W}$, and similarly for (u_2, v_2), P_{t+1}, and C_{t+1}. As long as the feature was moving, a least squares solution exists for the three unknowns of $\tilde{W}$, given these four or more (in the case of longer feature tracks) equations. We reconstruct using only the most temporally separated matrices P, instead of finding a $\tilde{W}$ based on the whole 2D track. This strategy generally gives maximum disparity and saves needless computations. After computing the camera poses, no outlier rejection is performed, so that an order of magnitude more tracked points are triangulated for the point cloud.

3 Motion and 3D Structure Features

We now describe the new motion and 3D structure features that are based on the inferred 3D point cloud. We suggest five simple cues that can be estimated reliably and are projected from the 3D world into features on the 2D image plane, where they enable semantic segmentation. We conclude the section by explaining how a randomized decision forest combines these simple weak features into a powerful classifier that performs the segmentation.

3.1 Cues from Point Clouds

Just as there are many ways to parameterize the colors and texture of appearance, there are numerous ways to parameterize 3D structure and motion. We propose five motion and structure cues. These are based on the inferred 3D point cloud, which, given the small baseline changes, is rather noisy. The cues were chosen as robust, intuitive, efficient to compute, and general-purpose but object-category covariant, though these five are by no means exhaustive. The cues also fit nicely with the powerful 3D to 2D projection mechanism (Sect. 3.2). With the driving application in mind, they were designed to be invariant to camera pitch, yaw, and perspective distortion, and could generalize to other domains.

The cues are: height above the camera, distance to the camera path, projected surface orientation, feature track density, and residual reconstruction error. These are intentionally weak; stronger features would not work with the sparse noisy point clouds, though dense feature tracking could someday enable one to apply [25]. We use machine learning to isolate reliable patterns and build a strong classifier that combines many of these cues (Sect. 3.3). By projecting from the 3D point cloud to the 2D image as described in Sect. 3.2, we are able to exploit contextual relationships. One of the benefits of video is that analysis of one frame can often be improved through information in neighboring frames. Our cues take advantage of this since feature tracks exist over several frames.

Height above the camera f_{H}. During video of a typical drive, one will notice that the only fairly fixed relationship between the 3D coordinate frames of the camera C and the world is the camera's height above the pavement (Fig. 2). Measuring height in image-space would be very susceptible to bumps in the

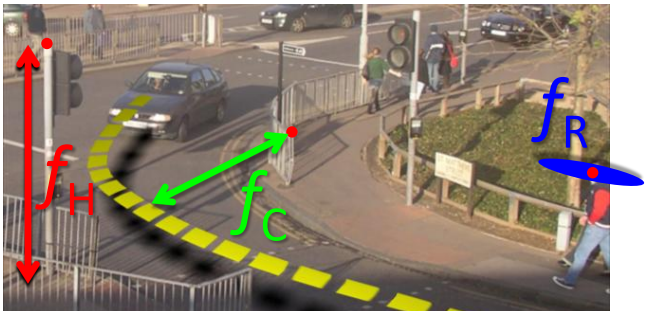


Fig. 2. The height, camera distance, and residual error features are illustrated for a car following the dotted yellow path. The red vertical arrow shows how f_{H} captures the height above the ground of a 3D point (red dot) reconstructed at the top of the stop light. The green arrow reflects the smallest distance between the point on the railing and the car's path. The blue ellipse for f_{R} illustrates the large residual error, itself a feature, in estimating the world coordinate $\tilde{W}$ of a point on the moving person's head.

road. Instead, after aligning the car's initial "up" vector as the camera's $-y$ axis, the height of each world point $\tilde{W}$ is compared to the camera center's y coordinate as $f_{\mathrm{H}}(\tilde{W}) = \tilde{W}_y - C_y$. By including a fixed offset C_y, the algorithm can be trained on point clouds from one vehicle, but run on other cameras and vehicles. Our experiments use footage from two different cars.

Closest distance to camera path f_{C}. The paths of moving vehicles on road surfaces are less repeatable than a class's absolute height in world coordinates, but classes such as buildings and trees are normally set back from driving roads by a fixed distance (Fig. 2). This feature, using the full sequence of camera centers $C(t)$, gives the value of the smallest recorded 3D separation between C and each $\tilde{W}$ as $f_{\mathrm{C}}(\tilde{W}) = \min_t \|\tilde{W} - C(t)\|$. Note that the smallest separation may occur after a feature in the current frame goes out of view. Such is the case most obviously with features reconstructed on the surface of the road.

Surface Orientation $f_{\mathrm{O_x}}, f_{\mathrm{O_y}}$. The points $\tilde{W}$ in the point cloud are too sparse and inaccurate in depth to allow an accurate 3D reconstruction of a faceted world, but do still contain useful spatial information. A 2D Delaunay triangulation [26] is performed on all the projected $\tilde{W}$ points in a given frame. Each 2D triangle is made of 3D coordinates which have inaccurate depths but, heuristically, acceptable relative depth estimates, and thus can give an approximate local surface orientation. The 3D normal vector for each triangle is projected to an angled vector on the image plane in 2D. The x and y components of this 2D angle are encoded in the red and green channels of a false-rendering of the triangulation, shown in the supplementary data online.

Track Density f_{D}. Faster moving objects, like oncoming traffic and people, often yield sparser feature tracks than stationary objects. Further, some object classes have more texture than others.We thus use the track density as one of the motion-derived cues. $f_{\mathrm{D}}(t)$ is the 2D image-space map of the feature density,

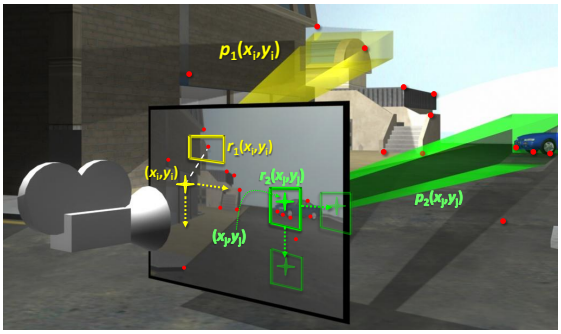


Fig. 3. Points in the 3D point cloud are marked as red dots, as are their projections from world space to the camera's image plane. Any feature information associated with a 3D point also lands on the image plane and is summed in Equations 1, 2 or 3. The yellow and green crosses illustrate how the algorithm slides over each pixel in turn to classify it using a randomized decision forest. Feature responses are calculated at a fixed relative 2D offset (white dashed line) and rectangle r. Here we show two example rectangles r_1 (yellow) and r_2 (green) with their associated truncated pyramids p_1 and p_2. Rectangle r_1 is offset up and to the left of pixel (x_i, y_i), and thus can use the context of *e.g.,* f_{C} to help determine the category at (x_i, y_i). Rectangle r_2 is centered on pixel (x_j, y_j) (*i.e.,* no offset), and thus pools the local information of *e.g.,* $f_{\mathrm{O_x}}$.

i.e., features with the requisite lifespan (3 frames) that were being tracked at a given time. For example, buildings and vegetation have high density, roads and sky have low density, and cars have both types of regions locally.

Backprojection Residual f_{R}. Having computed a 3D position $\tilde{W}$ for each trajectory (u_t, v_t), we compute $q(\tilde{W})$, the 2D variance of its reprojection error with respect to that track in pixels (Fig. 2). This serves to measure the accuracy of the rigid-world assumption, and highlights objects that move. We use a logarithmic scaling $f_{\mathrm{R}}(\tilde{W}) = log(1+q(\tilde{W}))$ to prevent apparent corners and tracking errors on distant objects from dominating the residuals caused by real moving objects. This motion-covariant feature is naturally dependent on the extent to which objects move, so should help separate buildings from cars, for example.[2] This cue is illustrated in the supplementary video.

3.2 Projecting from 3D to 2D

We extend the features suggested in [27] to project our cues from the 3D point cloud to the 2D image plane, illustrated in Fig. 3. A classifier is trained to compute a segmentation output for each pixel, scanning across the image. When classifying pixel (x, y) in the image, the randomized decision forest, described in Sect. 3.3, computes feature responses using rectangles $r(x, y)$ defined relative to (x, y). Given the camera center, each 2D rectangle implicitly defines a 3D

[2] Of course, however, it may also separate parked cars from moving ones.

truncated pyramid $p(x, y)$ forward of the image plane. For visible 3D world points within a truncated pyramid, the cue values are summed to give the feature responses, as follows. For heights f_{H}, camera path distances f_{C}, and residuals f_{R} the response is calculated as:

$$F_T(x, y) = \sum_{\tilde{W} \in p(x,y)} f_T(\tilde{W}) \quad \text{for} \quad T \in \{\mathrm{H}, \mathrm{C}, \mathrm{R}\}. \tag{1}$$

For surface orientation, the triangulated mesh is projected directly into the image, and the sum is over image pixels rather than world points:

$$F_{\mathrm{O_x}}(x, y) = \sum_{(x',y') \in r(x,y)} f_{\mathrm{O_x}}(x', y'), \tag{2}$$

and similarly for $F_{\mathrm{O_y}}$. For track density, the response is

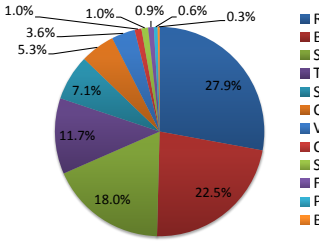
$$F_{\mathrm{D}}(x, y) = |\{\tilde{W} \in p(x, y)\}|, \tag{3}$$

i.e., the number of tracked points within pyramid p. Given this projection, we can make use of integral images [28] in the image plane, one for each cue, for fast feature response computation.

By defining the rectangles (and thereby truncated pyramids) relative to pixel (x, y), we can capture contextual relationships. For example, when classifying for a car pixel, it may be useful to know that a rectangle under the car has a road-like structure (see Fig. 3).

3.3 Randomized forest

Recent work [7] has employed randomized decision forests for fast and accurate segmentation using appearance features. We implemented a similar randomized forest classifier for segmentation based on our motion and structure features. It serves as a simple to implement and fast algorithm, that crucially, allows us to compare our motion and structure cues to the newest appearance results, on a level playing field. A number of randomized decision trees are averaged together to achieve robust segmentation and avoid over-fitting [29]. Each decision tree recursively branches down from root to leaf nodes. The non-leaf nodes compare a feature response F from (1), (2) or (3) to a learned threshold. At the leaf nodes, there is a class distribution learned from the training data, implicitly sharing features between classes. The MAP classification is given as the segmentation at each pixel. We use the extremely randomized trees algorithm [30] to train the forests. This recursively splits the training data, taking at each split the feature and threshold that maximizes the expected gain in information about the node categories. We follow the idea suggested in [7] of balancing the categories to optimize the category average performance versus the global performance.



	# Frames	Dataset
Seq 1	101	Day Train
Seq 2	204	Day Train
Seq 3	62	Dusk Train
Seq 4	171	Day Test
Seq 5	62	Dusk Test

Fig. 4. Left: Breakdown by category (listed clockwise from 12 o'clock) of the proportion of pixels in the 600 manually segmented frames in our driving video database. Right: 30Hz high-definition videos for which every 30th frame was painted manually with per-pixel semantic labels. Sequences were used as either training or testing data.

4 Experiments

The extensive experiments evaluated whether the simple ego-motion-derived cues could perform object recognition and segmentation. Since no existing database met those needs, we created a new labeled dataset of driving sequences. We then evaluated our motion and structure cues and compare them to existing appearance-based features. We finally show how our motion and structure cues can be combined with these appearance cues to improve overall performance. Further results including videos are available online.

Data Acquisition. Existing databases of labeled images do not include frames taken from video sequences, and usually label relevant classes with only bounding boxes. It takes the same amount of human effort to semantically label the pixels of N images drawn from video sequences as is needed for N independent photographs. The difference is that in the case of video, each labeled frame could have potentially many other temporally related images associated with it. Without an existing corpus of such data, we proceeded to film 55 minutes of daytime footage, 31 minutes of footage at dusk. Pedestrians and cyclists are visible at almost all times, but usually occupy only a small proportion of the field of view (see Fig. 4 left). The footage includes a variety of urban, residential, and mixed use roads. We developed a special purpose labeling tool for use in hand-segmenting the images. This is essentially a paint program with various edge detection and flood filling capabilities, but it also logs the amount of time and order of paint strokes a user employed to label each class. This data will be publicly available and we anticipate this will be of use to the community.

We selected daytime and dusk sequences, as listed in Fig. 4's table. Labeled images for each set are available at 1 fps, and ego-motion features and camera poses were computed at 30 fps. The labeled data has 11 categories: Building, Tree, Sky, Car, Sign-Symbol, Road, Pedestrian, Fence, Column-Pole, Sidewalk, and Bicyclist. There is also a small number of 'void' pixels not belonging to one of these classes that are ignored.

	Building	Tree	Sky	Car	Sign-Symbol	Road	Pedestrian	Fence	Column-Pole	Sidewalk	Bicyclist	**Average**	**Global**
Mot & Struct	43.9	46.2	79.5	44.6	19.5	82.5	24.4	**58.8**	0.1	**61.8**	18.0	43.6	61.8
Appearance	38.7	60.7	**90.1**	**71.1**	**51.4**	88.6	**54.6**	40.1	**1.1**	55.5	**23.6**	52.3	66.5
Combined	**46.2**	**61.9**	89.7	68.6	42.9	**89.5**	53.6	46.6	0.7	60.5	22.5	**53.0**	**69.1**

Table 1. Results in pixel-wise percentage accuracy on all three training and both test sequences, including both day and dusk frames. Note that (i) accurate semantic segmentation is possible using only motion and structure features, without any appearance information, and (ii) by combining our new motion and structure features with existing appearance features, we obtain a small but significant improvement. See text for more analysis.

Accuracy is computed by comparing the ground truth pixels to the inferred segmentation. We report per-class accuracies (the normalized diagonal of the pixel-wise confusion matrix), the class average accuracy, and the global segmentation accuracy. The average accuracy measure applies equal importance to all 11 classes, despite the widely varying class prevalences (Fig. 4 left), and is thus a much harder performance metric than the global accuracy measure. As a baseline for comparison with our results below, chance would achieve a global accuracy of about 9%. This rises to about 20% if the baseline chooses randomly according to the category priors.

4.1 Testing Motion and Structure Features

We trained a randomized decision forest based on our five motion and structure cues, using combined day and dusk sequences for both training and testing. The results are shown in the top row of Table 1 and the middle row of Fig. 7. These show the main contribution of the paper: that using only motion and structure information derived from sparse and noisy point clouds (Fig. 1), one can accurately segment images from video sequences and recognize the categories present. Observe in Figs. 1 and 7 that our algorithm segments the global scene well and even recognizes some of the smaller classes (*e.g.,* bicycle, sign). In terms of global accuracy, 61.8% of pixels are classified, and the strong average accuracy of 43.6% shows good consistency across the different categories. The perhaps low raw numbers highlight the difficulty of our new data set, but as we discuss shortly are comparable to a state-of-the-art appearance algorithm.

One by-product of balancing the categories during training is that the areas of smaller classes in the images tend to be overestimated, spilling out into the background (*e.g.,* the bicycle in Fig. 7). This suggests a shortcoming of the segmentation forest algorithm suggested in [7], that all pixels of a certain class are treated equally. The method in [31] may help with this. There is also considerable

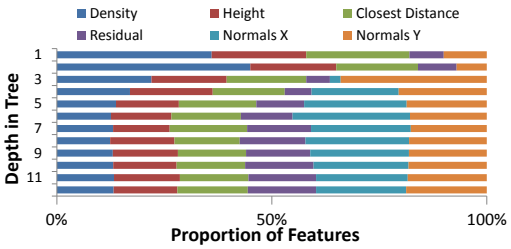


Fig. 5. Proportions of features used in the randomized segmentation forest, as a function of node depth. At the top of the tree there is some bias toward our density, height and closest distance cues. But deeper in the tree all cues are informative and used in roughly equal proportions.

Cues Used	Balanced Ave. Score	Global Score
All	**43.3%**	**63.0%**
Just Height	39.1%	55.3%
Just Distance	41.9%	57.1%
Just Orient.	37.3%	59.0%
Just Density	40.2%	60.0%
Just Residual	36.2%	58.1%

Fig. 6. We combine all the cues, but here each cue was also tested in isolation. Scores were computed by either optimizing the balanced per-category average, or the global % correct - of pixels assigned to the same class as in the ground truth.

confusion between fence and building which we believe to be shortcomings in the ground truth.

To determine the relative importance of the five motion and structure cues, we analyzed the proportion of each chosen by the learning algorithm, as a function of depth in the randomized forest. In Fig. 5 we observe near the tree roots that there is some bias toward the density, height, and closest distance cues. Further down the tree however, all five cues play an important and balanced role (normals were split into x and y components in the figure). This suggests that the density, height, and closest distance cues work well to segment the rough global structure of the scene, and that the finer details are tackled more consistently by all five cues.

These result used a randomized forest containing 50 trees trained to a maximum depth of 13, testing 500 random features (cue choice and offset rectangles) at each step of building the tree. The learning takes only about 15 minutes, and testing takes less than one second per frame.[3] Our system should scale well, at worst linearly with the number of object classes and training images.

4.2 Comparison with Appearance Features

We compared with a state-of-the-art technique [7]. It uses dense pixel patches to semantically segment images using only appearance information (no motion or structure). Table 1 includes the comparison between our motion and structure features vs. the appearance features of [7]. As one might expect, given much denser and less noisy image features, appearance works somewhat better than motion and structure, though clearly this does not diminish our contribution that the new motion and structure cues work at all. We discuss below how these two complementary types of feature can be combined to improve overall results.

[3] These timings assume pre-computed SfM point clouds. Recent work [22] has moved towards making this real-time.

	Train Day – Test Dusk		Train Dusk – Test Day	
	Average	Global	Average	Global
Mot & Struct	**29.2%**	**45.5%**	**31.0%**	**59.4%**
Appearance	14.2%	21.7%	25.4%	50.5%

Table 2. By training in one lighting condition (day or dusk) and testing in the other, we compare the lighting invariance of our motion and structure features with appearance based features. Observe much better generalization of our motion and structure features to novel lighting conditions.

Motion and structure features do however have an obvious advantage over appearance features: generalization to novel lighting and weather conditions. We compare in Table 2 the global and average segmentation accuracies obtained when training in one lighting condition (day or dusk) and testing in the other. Figure 8 and the online materials show segmentation results. We see for both combinations that the new motion and structure features generalize much better than the appearance features. Extra labeled data could be used to improve the appearance features, but obtaining labeled data is very expensive. Without *any* extra data, our motion and structure features can reasonably be expected to generalize to other lighting and weather conditions such as night, snow or rain, since they are almost independent of image appearance (up to obtaining feature tracks).

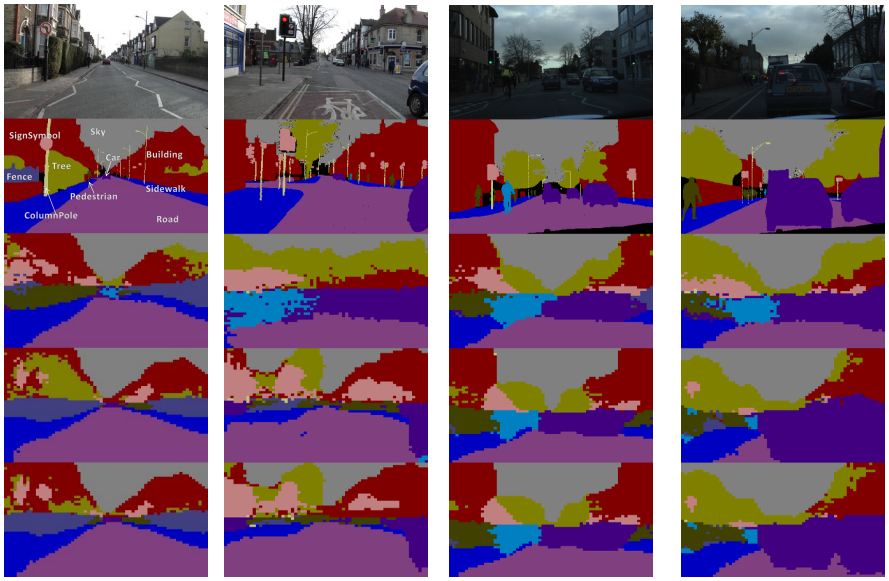
4.3 Combined Ego-Motion & Texton Features

Since our motion and structure features contain rather different information to the appearance features of [7], one would expect the two to be complementary. We investigated a simple method of combining the features, by taking a geometric interpolation of the two classifiers. We denote our randomized decision forest classifier based on motion and structure cues as $P(c|M)$, and the appearance-based classifier from [7] as $P(c|A)$. These were trained independently and then combined as

$$P(c_{(x,y)}|M,A) = \frac{1}{Z}\ P(c_{(x,y)}|M) \times P(c_{(x,y)}|A)^{\alpha}, \tag{4}$$

where α is a weighting parameter chosen by holdout validation, and Z is used to renormalize the distribution. The two distributions $P(c|M)$ and $P(c|A)$ should reinforce their decisions when they agree and flatten the distribution when they disagree, a kind of soft 'AND' operation. This was found better in practice than an arithmetic average ('OR').

The results for this combination can be seen in the last row of Table 1 and Fig. 7, and in the online video, using $\alpha = 2.5$. The soft AND operation does not guarantee an improvement for all categories, but still we observe a small but significant improvement in both average and global accuracy. The qualitative appearance of the segmentations is also consistently improved. These results are very encouraging, suggesting that our motion and structure features are indeed complementary to appearance features.



(A) DayTest #0450 (B) DayTest #2460 (C) DuskTest #8550 (D) DuskTest #9180

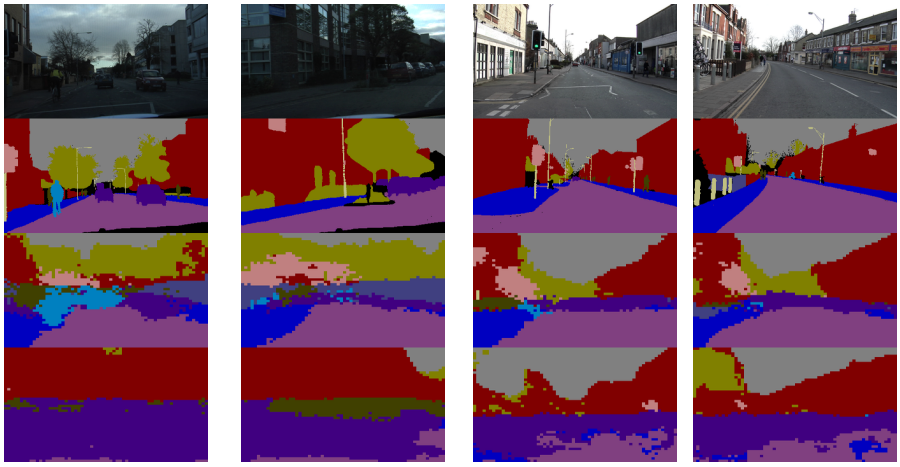
Fig. 7. Sample segmentation results. From top to bottom: test image, ground truth, motion and structure inferred segmentation, appearance inferred segmentation, and combined segmentation. Note that accurate segmentation and recognition is possible using only motion and structure features, and that combining our new cues with existing appearance cues gives improved segmentation. The whole video sequence is online.

5 Conclusions

Using motion and 3D world structure for segmentation and object recognition is a fundamentally new challenge. Our main contribution has been to show that accurate results are possible using only ego-motion derived 3D points clouds. Experiments on a challenging new database of naturally complex driving scenes demonstrate that our five new motion and structure cues can be combined in a randomized decision forest to perform accurate semantic segmentation. These five cues were also shown to generalize better to novel lighting conditions than existing appearance-based features. By then combining motion and structure with appearance, an overall quantitative and qualitative improvement was observed, above what either could achieve individually.

The worst performance of our system is for those categories least well represented in the training data, despite balancing categories during training. We hope that semi-supervised techniques that use extra partially labeled or unlabeled training data may lead to improved performance in the future.

Our combination of segmentation classifiers (Equation 4) is somewhat simplistic, and we are investigating other methods. Learning a histogram for each pair of (motion and structure, appearance) tree leaf nodes could better model



(A) DuskTest #8580 (B) DuskTest #10020 (C) DayTest #0960 (D) DayTest #4680

Fig. 8. Inherent invariance of motion and structure to novel lighting conditions. From top to bottom: test image, ground truth, motion and structure inferred segmentation, and appearance inferred segmentation. When trained on daytime footage and tested on dusk footage and vice-versa, our motion and structure cues are still able to accurately recognize and segment the scene. In contrast, the appearance inferred segmentation degrades drastically.

the joint dependencies of the two classifiers, but care must be taken so that in avoiding overfitting, quadratically more training data is not required.

Acknowledgements Thanks to John Winn for advice and for driving one of the capture cars.

References

1. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Second edn. Cambridge University Press (2004)
2. Merrell, P., Akbarzadeh, A., Wang, L., Mordohai, P., Frahm, J.M., Yang, R., Nister, D., Pollefeys, M.: Real-time visibility-based fusion of depth maps. In: Proceedings of the International Conference on Computer Vision (ICCV). (2007)
3. Posner, I., Schroeter, D., Newman, P.M.: Describing composite urban workspaces. In: ICRA. (2007)
4. Boujou: 2d3 Ltd., http://www.2d3.com. (2007)
5. Chum, O., Zisserman, A.: An exemplar model for learning object classes. In: CVPR. (2007)
6. Li, L.J., Fei-Fei, L.: What, where and who? classifying events by scene and object recognition. In: ICCV. (2007)
7. Shotton, J., Johnson, M., Cipolla, R.: Semantic texton forests for image categorization and segmentation. In: CVPR. (2008)

8. Hoiem, D., Efros, A.A., Hebert, M.: Putting objects in perspective. In: CVPR. Volume 2. (2006) 2137 – 2144
9. Hoiem, D., Efros, A.A., Hebert, M.: Geometric context from a single image. In: ICCV. Volume 1. (2005) 654 – 661
10. Huber, D., Kapuria, A., Donamukkala, R., Hebert, M.: Parts-based 3d object classification. In: CVPR. (2004) II: 82–89
11. Hoiem, D., Rother, C., Winn, J.: 3d layout crf for multi-view object class recognition and segmentation. In: CVPR. (2007)
12. Kushal, A., Schmid, C., Ponce, J.: Flexible object models for category-level 3d object recognition. In: CVPR. (2007)
13. Pingkun, Y., Khan, S., Shah, M.: 3d model based object class detection in an arbitrary view. In: ICCV. (2007)
14. Savarese, S., Fei-Fei, L.: 3d generic object categorization, localization and pose estimation. In: ICCV. (2007)
15. Dalal, N., Triggs, B., Schmid, C.: Human detection using oriented histograms of flow and appearance. In: ECCV. (2006)
16. Cedras, C., Shah, M.: Motion-based recognition: A survey. IVC **13**(2) (March 1995) 129–155
17. Laptev, I., Lindeberg, T.: Space-time interest points. In: ICCV. (2003) 432–439
18. Viola, P.A., Jones, M.J., Snow, D.: Detecting pedestrians using patterns of motion and appearance. In: ICCV. (2003) 734–741
19. Yin, P., Criminisi, A., Winn, J.M., Essa, I.: Tree-based classifiers for bilayer video segmentation. In: CVPR. (2007)
20. Wiles, C., Brady, M.: Closing the loop on multiple motions. In: ICCV. (1995) 308–313
21. Kang, J., Cohen, I., Medioni, G.G., Yuan, C.: Detection and tracking of moving objects from a moving platform in presence of strong parallax. In: ICCV. (2005) 10–17
22. Leibe, B., Cornelis, N., Cornelis, K., Gool, L.J.V.: Dynamic 3d scene analysis from a moving vehicle. In: CVPR. (2007)
23. Efros, A.A., Berg, A.C., Mori, G., Malik, J.: Recognizing action at a distance. In: ICCV. (2003) 726–733
24. Harris, C., Stephens, M.: A Combined Corner and Edge Detector. In: 4th ALVEY Vision Conference. (1988) 147–151
25. Mitra, N.J., Nguyen, A., Guibas, L.: Estimating surface normals in noisy point cloud data. In: special issue of International Journal of Computational Geometry and Applications. Volume 14. (2004) 261–276
26. Shewchuk, J.R.: Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In: Applied Computational Geometry: Towards Geometric Engineering. Volume 1148 of LNCS. (1996) 203–222
27. Shotton, J., Winn, J., Rother, C., Criminisi, A.: Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In: ECCV. (2006)
28. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR. (2001) 511–518 vol.1
29. Amit, Y., Geman, D.: Shape quantization and recognition with randomized trees. Neural Computation **9**(7) (1997) 1545–1588
30. Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine Learning **36**(1) (2006) 3–42
31. Winn, J., Shotton, J.: The layout consistent random field for recognizing and segmenting partially occluded objects. In: CVPR. (2006)