Assisted Video Object Labeling By Joint Tracking of Regions and Keypoints

Julien Fauqueur Gabriel Brostow Roberto Cipolla
University of Cambridge
Computer Vision Group, Department of Engineering, Trumpington street, Cambridge CB2 1PZ, UK
http://mi.eng.cam.ac.uk/∼cipolla/

Abstract

Manual labeling of objects in videos is a tedious task. We present an approach which automatically propagates the labels from a single frame to the next ones.

We tackle the challenging problem of tracking segmented regions by combining keypoint tracking with an advanced multiple region matching strategy, based on inclusion similarity and connected regions.

We ran experiments on a 101 frame driving video sequence for which we produced the corresponding hand-labeled groundtruth. We make this valuable dataset available for the research community. We show our technique can accommodate variations in segmentation (and correct them), even in presence of multiple independent motions and partial occlusion. Results show that most of the labeled pixels can be correctly propagated even after a hundred frames. The performance of this automatic propagation mechanism over many frames can greatly reduce the user effort in the task of video object labeling.

1. Introduction

The representation of objects in video has different applications in Medical Imaging, Content Analysis, Film Industry. However, in order for objects to be identified, a human operator has to explicitly label them. Even automatic object recognition algorithms require the production of hand-labeled visual data to be trained. The task of labeling objects is tedious, in particular for videos.

Different approaches have been proposed in the literature which can facilitate this task. In [8], an interactive approach is proposed to precisely extract a distinct foreground object from its background. The user is closely involved in the refinement of the segmentation. A related problem to label propagation is the Colorization problem [2]. With a few colored strokes on a grayscale video, the user specifies how to colorize it in a realistic manner. However converting the produced colors into a label map is not straightforward. Other segmentation based techniques [4][6] are more directly addressing the problem of label propagation. Both these systems rely on motion and color information to propagate regions through the sequence. Comparison of these techniques are difficult in the absence of labeled groundtruth data.

Label propagation is a very challenging problem because it requires to track object regions which lack of "visual identity" and are inherently unstable. Other local representations exist for robust tracking, such as MSER [5] or SIFT [3], however they are sparse features and are not sufficient to track entire regions.

We introduce a new technique which combines keypoints and regions to propagate region labels. Regions are tracked using both tracked keypoints and flexible region similarity measures to benefit from both local representations : robustness and density. The matching process is specifically designed to match *segmented regions* whose shapes can vary greatly. Based on this joint region and keypoint tracking, the propagation of labels comes down to a simple vote. The accuracy of this method will be demonstrated on a driving video sequence which contains multiple objects with independent motions.

We also provide for download a new video dataset (this driving video sequence) for which we manually produced the associated labeled groundtruth.

2. Region and keypoint detection, and their association

In this section we describe the techniques used to detect regions and keypoints in each frame. A method to associate keypoints and regions is also be presented.

Oversegmentation. Each frame of the sequence is segmented individually. We chose the Felzenswalb and Huttenlocher graph-based segmentation [1], which produces good results to capture areas of similar color and texture. Given the rather large size of our images (960×720 pixels) its

computational efficiency is also an advantage. Based on the assumption that a very homogeneous region belongs to the same object, we produce an oversegmentation of the video frame (see figure 1). It reduces the risks of having two objects merged together, while it allows for a better approximation of the hand-labeled regions. We chose parameters with the following values : variance of gaussian smoothing $\sigma = .8$, granularity parameter $k = 50$ and minimum region size of 50. On average this segmentation produced 1811 regions per image. As a comparison, our groundtruth hand-labeled images have 150 hand-labeled regions on average on the same sequence. A median filter is applied to the segmented image to smooth the region boundaries, which tend to be more regular when drawn by a human.

Figure 1. Result of the image oversegmentation.

Bi-directional keypoint detection. Since we will be using the scale of keypoints to associate them with regions (see below), we need to obtain robust keypoints (with good repeatability across frames) along with an estimate of their scale. The SIFT algorithm [3] is a relevant choice to meet these requirements. The keypoints are matched across each pair of consecutive frames A and B. We first consider all matches from A to B then from B to A and which are mutual (i.e. a keypoint k_A in A which matches k_B in B is kept if k_B matches k_A and vice versa). This bi-directional matching technique prove to be very effective to obtain numerous robust keypoint matches. By skipping the rejection step based on the distance to the second closest match (as suggested in [3]), we allowed for more correct keypoint matches. We also reject matches for which the inter-keypoint distance in the image space is larger than 200 pixels. About 2000 keypoints are kept based on this bi-directional strategy. Note that the keypoint matching produces very few false positives since frames are very similar.

The **association between keypoints and regions** will be used both for tracking regions and propagating labels. It is not an obvious task since keypoints can lie near a region boundary and using only their pixel location and see in which region they lie could be unstable. Instead we view a keypoint as a disc region whose radius is given by its scale which defines its support. A keypoint is associated with a region if at least half of its support disc intersects with the region. For efficiency we consider the square embedded in the disc rather than the entire disc. As we will see in section 3, only regions with at least one keypoint are candidates for stable matches.

3. Region tracking

In this section, we describe the process of tracking the segmented regions between consecutive frames. It is a challenging problem because the changes in segmentation imply that most regions can potentially match many regions. We will first discuss the necessity for adequate descriptors and similarity measures to compare region appearance. We will describe the two step tracking process which first relies on the matching of stable regions which are then used to match the other ambiguous ones.

3.1. Appearance matching for segmented regions

Our first attempt for tracking regions involved using a combination of state-of-the-art regions descriptors for color (average color, color histogram), shape (compactness and eccentricity) and texture (Gabor-based descriptor). We observed that they did very well to match only very similar regions, *when they exist*. However, between two consecutive frames, even if they are very similar, most of the segmented regions change noticeably in their geometry (boundaries, area and shape). As a consequence, the global appearance of regions can change significantly and these descriptors will fail. In this section we present similarity measures which can match segmented regions in a flexible and accurate way.

The key idea for matching segmented regions is to measure *how much they have in common* rather than *how similar they are overall*. A simple and efficient solution is the *Histogram Intersection* similarity measure [7]. Given two regions r_A and r_B and their respective color histograms h_A and h_B, the Histogram Intersection $HI(r_A, r_B)$ counts how many pixels are similar in r_A and r_B :

$$HI(r_A, r_B) = \sum_{i=1}^{n} min(h_A(i), h_B(i)) \qquad (1)$$

where n denotes the number of quantized colors. We use $n = 6^3 = 216$ quantized colors in the Luv color space. Unlike classic L^P distances, the strength of the Histogram Intersection is to completely ignore pixels which fall in different bins. We now detail two measures based on it which

model two different types of appearance similarity between regions.

The first one is the normalized Histogram Intersection [7] which we propose to use as an **inclusion similarity** measure between two regions r_A and r_B :

$$m_I(r_A, r_B) = HI(r_A, r_B)/card(r_A) \quad (2)$$

Note it is not a symmetric measure. It takes values in $[0; 1]$ and is maximum when all pixels in r_A have a similar pixel in r_B, i.e. $r_A \subset r_B$. We will explain in section 3.3 how this inclusion measure is relevant to match similar regions of different shapes obtained from different segmentations.

The second one measures an **overall similarity** :

$$m_O(r_A, r_B) = HI(r_A, r_B)/max(card(r_A), card(r_B)) \quad (3)$$

Unlike m_I, m_O is maximum (i.e. $= 1$) when both regions have the same number of similar pixels.

3.2. Matching stable regions

We focus on the pairs of regions between the two frames for which the matching is the most reliable by using their associated keypoints. They reduce the error and the complexity of the overall tracking process. In section 2 we explained how keypoints were associated with regions in the same frame and matched with keypoints in the next frame. By transitivity, we use the matched keypoints between two frames to match their corresponding regions. Since regions can have more than one keypoint, a region is matched to the one which shares the most matching keypoints. This strategy is a sufficient condition to ensure that the two regions cover a common patch of the scene (up to the keypoint matching accuracy). We also add a constraint on overall similarity to match those regions to avoid, for instance, a tiny region matching a much larger one. We use the overall appearence similarity measure m_O (formula 3). Therefore, two regions are called *stable regions* if they have a maximum number of matching keypoints in common (which is non-zero) and if $m_O(r_A, r_B) \geq \tau_O$. A value of $\tau_O = 0.3$ was found to be sufficient to reject unlikely matches. Both regions are said to constitute a *stable match* for eachother. We denote as $M_s(r)$ the stable matches for a region r.

3.3. Matching ambiguous regions

After matching stable regions, we are left with regions with no keypoints in them or with keypoints which could not be associated with regions and we refer to them as "ambiguous regions". Typically those regions exhibit very little variation (such as sky, grass, road, uniform areas) and are harder to register. They tend to be fragmented by the oversegmentation into similar regions whose shapes vary greatly across frames. As a consequence, they are the most challenging regions to match.

By counting the number of similar pixels in common between two regions, the inclusion similarity measure m_I is a natural solution to match similar regions obtained from different segmentations. Let us denote as r_A a region in frame A and r_B a region in the subsequent frame B. We determine all non-stable regions in A potentially included in a region in B and pick the closest one which we define as $M_a(r_A) = argmin\{|| r_B - r_A || \mid m_I(r_A, r_B) \geq \tau_I\}$, where $|| \ . \ ||$ denotes the inter-region distance defined as the euclidean distance between their centre of mass. Conversely for frame B, we determine non-stable regions in B potentially included in a region in A and pick the closest one $M_a(r_B) = argmin\{|| r_A - r_B || \mid m_I(r_B, r_A) \geq \tau_I\}$. We require that r_A has at least half its pixels in common with r_A, by setting $\tau_I = 0.5$.

Note that ambiguous matches are asymmetric whereas stable matches are symmetric : if r_A is a stable match for r_B then r_B is a stable match for r_A, but if an ambiguous region r_A matches r_B (which can either be a stable or an ambiguous region), then r_B is not necessarily an ambiguous match for r_A.

3.4. Consolidate multiple matches for adjacent regions

So far we have looked at individual matches for both stable and ambiguous regions. Without additional contextual information, some false matches may persist. In particular, since we previously allowed multiple regions to match a same region (to handle changes in segmentation), we now want to keep only the best group of *connected* regions which match a same region.

For each region in a frame A, we consider the ambiguous region that the region matched as well as the regions that match it in the next frame B. This group of matching regions is denoted as M. Among those regions, we only consider the group of connected matching regions whose histogram maximizes the similarity for the region. The workflow is the following :

1. List matching regions in B : for each region r_A, $M_a(r_A)$ denotes its ambiguous match. The ambiguous regions which match r_A are denoted as $M_a^{-1}(r_A) = \{r_B \mid M_a(r_B) = r_A\}$ and the stable ones as $M_s^{-1}(r_A) = \{r_B \mid M_s(r_B) = r_A\}$. So the set of potential matches for r_A is $M = M_a(r_A) \cup M_a^{-1}(r_A) \cup M_s^{-1}(r_A)$

2. List groups of connected matching regions in B : among regions in M we only want to consider groups of connected regions which contain stable regions (if any). Such a group is written as $G = \{r \in M | \bigcup r$ is connected and $\bigcup r \supset M_s^{-1}(r_A)\}$

3. Find best adjacent group : we define the best group

of regions as the set G^* which maximizes the overall appearance similarity between r_A and all groups G : $G^* = argmax\{m_O(r_A, G)\}$[1]. Regions in $M \setminus G^*$ are released, i.e. match the "void" class.

The same workflow is then applied between the frame B and A (simply by swapping A's and B's in the notations).

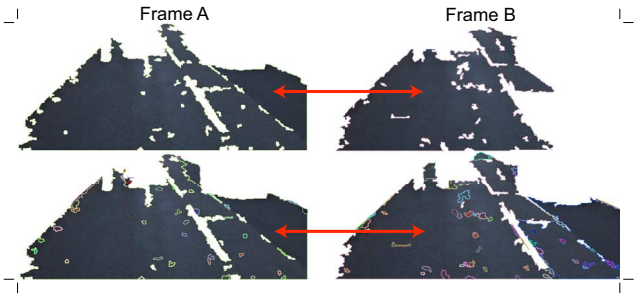

Figure 2. Example of multiple region match across steps 1 to 3 for both frames. We show two segmented road regions from two consecutive frames (left versus right figures). The top regions were mutually matched in step 1 (applied to each frame), in spite of their shape difference. In steps 2 and 3, many small connected regions matched the big region in the other frame, thanks to the inclusion similarity measure and the adjacent region grouping strategy. The bottom row shows the final optimal match between the groups of connected regions (tiny and big ones) in both frames. In the propagation step, both groups will be considered as two single regions which are more coherent than the initial big ones alone. This shows that this strategy can cope with variable regions and even improve the final segmentation.

Figure 2 illustrates how this iterative workflow can match two groups of adjacent road regions of different shapes but similar appearance. By selecting the best arrangement of connected regions, the two final matched groups of connected regions produce two more coherent regions and hence a better match. More generally, an important property of our matching scheme is its ability to improve the segmentation in each frame by grouping adjacent regions and matching them to a larger one. Note that a classic region merging algorithm applied to each frame *separately* could not offer the same performance, because it is harder to assess whether two adjacent regions belong to the *same* region.

These steps conclude the region matching process. The outcome is an association between regions across two frames in which each region matches another region and can be matched by many regions from the other frame.

At the end of this matching process, a few regions can be left unmatched when they did not meet the above criteria because they are too ambiguous or because they belong to an object which has disappeared.

[1] $m_O(r_A, G)$ denotes the similarity between r_A and the set of regions G which is defined as $\sum_{r \in G} HI(r_A, r) / max(card(r_A), \sum_{r \in G} card(r))$

4. Label propagation

So far, we have not used the label information. All the previous steps for extracting regions and keypoints and matching are done automatically without user interaction. As stated earlier, we require the user to provide an input labeled image only for the first frame of the sequence.

In this section, we explain how the labels from the input map are associated with regions and keypoints in the first frame, and then how labels are propagated in subsequent frames using the tracked regions (as explained in section 3).

Let A denote a frame and B its subsequent one. Given a labeled frame L_A corresponding to A (whether it be the first hand labeled frame or the previous estimated frame), we want to produce the estimated labeled frame L_B for B. A labeled map consists of uniform color blobs which identify each object with a color (e.g. red for building in figure 3). Black color identifies the "void" label and is used to refer to semantically ambiguous areas.

Keypoint-to-label association in A and B. For the first frame, keypoints are associated with labels by intersecting their support regions in A (see section 2) with L_A, if the most frequent label appears at least on half its support region. Since the keypoint matches are robust and their support regions are small, this association is robust. So we produce directly the keypoint-label association for frame B based on the corresponding matching keypoints. For the subsequent frames, the keypoint-label association will then be provided by the previous frame. **Region-to-label association in A.** Likewise, by intersecting the segmented map of A with L_A, each region is associated with the most frequent label within its area. **Region-to-label association in B.** Many regions in A (who have now a label) are likely to match a same region r_B from the previous matching process (section 3). For each r_B we consider the most frequent label l_R among the r_A's which match r_B. If r_B contains keypoints which have been assigned a label then we consider, l_K, the most frequent of these labels within r_B. Most of the time l_R and l_K will coincide. If they do not and one of them is "void" then we assign the other label to r_B. If they are different and not void, then we assign l_K if more than one keypoint has it or l_R otherwise.

5. Experiments and the hand-labeled dataset

We are interested in the problem of automated driving cars and recognizing the main objects relevant to a driver, such as road, pedestrians, vehicles.

Dataset capture. In the absence of available video footage for our problem, we produced our own dataset, with a camera mounted on the passenger seat in a car. A high-definition digital camera was used, capturing 960×720 pixel frames at 30fps. We filmed about 90 minutes in the streets of our city. The results we report here were obtained

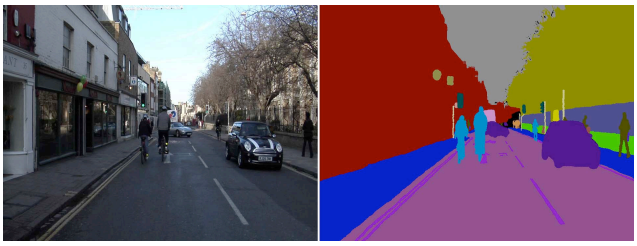

Figure 3. Original image (first frame of the sequence) and corresponding hand labeled image.

Figure 4. Original image (last frame of the sequence) and corresponding hand labeled image. The bicyclist is the same but most of the scene has changed.

on a sub-sequence of 101 consecutives frames subsampled every other frames (from 202 frames). It corresponds to 6 seconds of continuous driving. No part of the scene remains static and various events occur involving multiple moving objects : crossing two cars (one makes a turn), following then overtaking bicyclists, three groups of pedestrians walking in both direction and a remote bus takes a turn. The first and last frames are shown in figures 3 and 4 with their corresponding labeled frames.

Hand labeling. In order to validate our approach, we hired people to produce manually the labeled maps for each of the 101 frames. They painted the areas corresponding to a predefined list of 32 objects of interest given a specific palette of colors. They have used a program that we developed for this task which offers various automatic segmentations along with floodfilling and manual painting capabilities. We corrected the labeled sequence to make sure that no object was left out and that the "labeling style" (which does vary depending on the person) was consistent across the sequence. By logging and timing each stroke, we were able to estimate the hand labeling time for one frame to be around 20-25 minutes (this duration can vary greatly depending on the complexity of the scene). The first labeled frame was used to initialize our algorithm while the following 100 ones were used to compare our estimated labeled maps and report the accuracy of our method. Both the original image sequence and its corresponding hand-labeled sequence have been made available for download[2]. We believe this unique dataset can constitute a valuable testbed in the research community to validate various Computer Vision algorithms.

Implementation. Our program was written in C^{++} using the OpenCV[3] library and run on a 2GHz dual core PC. We used the authors' implementation of the SIFT keypoint extraction[4] and the segmentation algorithms[5]. The total runtime for processing the 101 frames was 82 minutes (about 49s/frame on average) of which 23 minutes were taken by the segmentation, keypoint extraction and region detection and description. Region tracking and propagation took the 59 remaining minutes, most of which are taken by a few cases when many region groupings are inspected in the step 3 of the region matching. The actual propagation step, as described in section 4, takes only 0.6s per image.

6. Results

Automatic region segmentation. We observed that the region tracking strategy (section 3.3) was able to merge similar adjacent regions consistently by matching them to larger regions in consequent frames. In spite of the oversegmentation, there were still situations where some part of an object was merged with the background if they had very similar appearance. This type of error is inherent to any unsupervised segmentation algorithm and only manual interaction could correct this type of error.

Region tracking. Compared to an early version of our algorithm, we noted that the addition of keypoints was crucial to register the regions to the same location in the scene and thus avoid severe propagation of labeling errors. In the absence of any motion model, keypoints improved the region tracking in ambiguous cases, such as moving and repetitive pattern (e.g. as lane markings or a facade behind a fence). The use of keypoint and histogram intersection proved to be flexible and effective to match regions correctly in very challenging conditions : multiple moving objects, partial occlusion and changes in segmentation.

Mismatch problems usually include two moving objects which are connected in the image plane and for which a keypoint or a region switches from one object to another. This type of error could not be corrected automatically. Small objects such as sign poles were hardly tracked because of their small size and their lack of saliency. However, sometimes a single keypoint could help tracking a non-trivial region, such as a small shop sign.

Label propagation. The overall performance of our label propagation method was measured based on the confusion matrix for all object classes between the groundtruth labeled frames and each estimated labeled frame. The results of the progagation can be viewed in the supplementary

[2] http://www.eng.cam.ac.uk/∼jf330/CamSeq01/

[3] http://sourceforge.net/projects/opencv/

[4] http://www.cs.ubc.ca/∼lowe/keypoints/

[5] http://people.cs.uchicago.edu/∼pff/segment/

video. We measure the propagation accuracy rates for the object classes normalized by the object size. In other words, this rate indicates the percentage of pixels for a given class which were correctly labeled in each frame. In figures 5 and 6, we show the evolution of propagation accuracy through frames of the most frequent classes along with their population per frame. The correct propagation rates vary significantly from <1% to 98% across the sequence depending on the class. In particular, the highest rates are generally obtained by the largest objects such as road and building (figure 5). Cars (figure 5) and bicyclists (figure 6), although representing only 1% and 2% in size on average, achieve reasonably well. Those latter classes are particularly important to detect in a driving environment. In figure 6, we can observe the "LaneMkgsDriv" (drivable lane markings) accuracy drops rapidly although they are still present in the sequence (bottom left figure). Unlike large continuous regions such as building, lane markings are composed of interrupted similar patches on the road and their track is quickly lost.

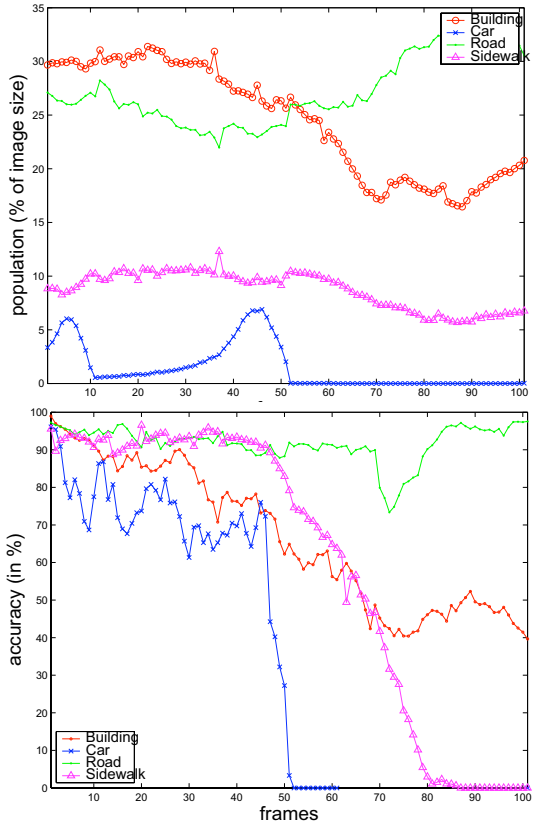

Figure 5. Label propagation normalized accuracy per frame across the sequence for the biggest classes and size of the classes in each frame. Both size and accuracy measures are calculated using the groundtruth frames. The presence of local minima in most of the accuracy curves show that partial occlusion is handled (i.e. part of an object is lost and then recovered).

The reported rates are normalized by their population.

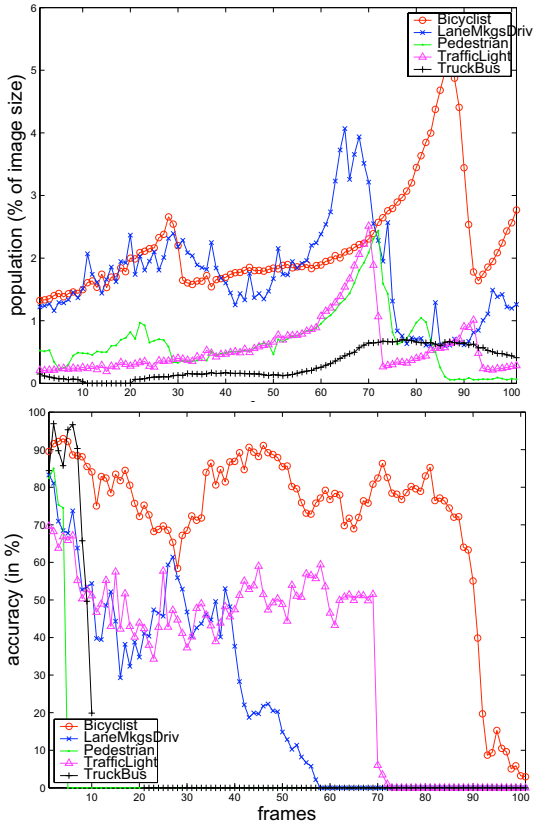

Figure 6. Same as figure 5, but for the smaller classes.

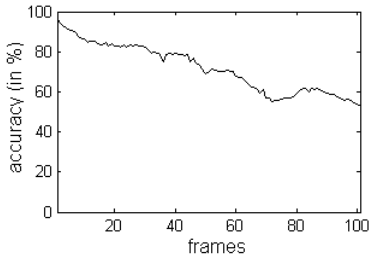

Figure 7. Overall accuracy for label propagation averaged over all classes.

But from the human labeler point of view, the overall number of correctly classified pixels really matters, as shown in figure 7. It drops from 98% to 53% at the last frame. Given the difficulty of this sequence and the absence of user input throughout the sequence, those rates are very encouraging. The area below this curve can be viewed as the number of pixels the user will not have to label. Given that each image takes 20-25 minutes to label, it means a considerable time gain.

7. Conclusion

We have presented a new flexible yet robust technique to automate the propagation of object labels in video based on

joint keypoint and region tracking. Salient parts of objects are correctly tracked thanks to combination of keypoints and regions. Other difficult object parts (such as textured regions) are efficiently registered using similarity measures dedicated to oversegmented regions.

Tests were run on a driving video sequence and the accuracy of propagation for different classes was measured against a labeled groundtruth. The overall propagation accuracy is very promising on a challenging video sequence, which reduces greatly the task of the human labeler.

The performance can be further improved by allowing the user to interactively correct errors in difficult cases and imposing regularity constraints on the track to handle full occlusion.

Acknowledgement: : We are grateful to Toyota Motor Europe for supporting this work.

References

[1] P. Felzenszwalb and D. Huttenlocher. Image segmentation using local variation. *International Journal of Computer Vision (IJCV)*, 59(2), september 2004. 1

[2] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. *ACM Transactions on Graphics (SIGGRAPH)*, August 2004. 1

[3] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004. 1, 2

[4] B. Marcotegui, F. Zanoguera, P. Correia, R. Rosa, F. Marques, R. Mech, and M. Wollborn. A video object generation tool allowing friendly user interaction. *IEEE International Conference on Image Processing (ICIP)*, October 1999. 1

[5] J. Matas, O. Chum, U. Martin, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. *British Machine Vision Conference (BMVC)*, 1:384–393, 2002. 1

[6] I. Patras, E. Hendriks, and R. Lagendijk. Semi-automatic object-based video segmentation with labeling of color segments. *Signal Processing: Image Communication*, 18(1):51–65, January 2003. 1

[7] M. Swain and D. Ballard. Color indexing. *International Journal of Computer Vision (IJCV)*, 7(1):11–32, 1991. 2, 3

[8] J. Wang, P. Bhat, A. Colburn, M. Agrawala, and M. Cohen. Interactive video cutout. *ACM Transactions on Graphics (SIGGRAPH)*, 2005. 1