Segment Cluster Tracking

Mark Ross
Image Recognition Laboratory,
University of Koblenz,
56075 Koblenz, Germany
ross@uni-koblenz.de

http://www.uni-koblenz.de/~lb/

Abstract

Segment Cluster Tracking is a new approach of motion analysis in the class of mapping methods/algorithms. It solves the correspondence problem by matching image regions, resp. features of image regions.

The image regions are created by combining color segments - the result of a color segmentation- to segment clusters. The tracking task is to find best correspondences between clusters of segments in consecutive frames of a sequence.

Segment Cluster Tracking has better computation time in comparison to the prior approach of so called n:m-Matching [Reh98] and in addition the possibility of using backward information. This leads to an increasing tracking quality and robustness [Ros03].

1. Introduction

One of the most important problems in image sequence analysis is motion estimation. Example applications involving motion estimation are vehicle guidance, driving assistance, or traffic monitoring.

1.1. Previous works

Most researchers use methods of motion tracking based on any kind of difference between successive images.

Ellis and Xu [EX01] partition frames into foreground and background depending on changes in successive images. As images captured by a moving camera are completely in change this approach fails in that case.

Stauffer and Grimson [SG98] describe a tracking algorithm which separates foreground - the moving objects - from background. For this purpose the pixel values are modeled as a mixture of Gaussians. Pixel values that do not fit the background distributions are considered foreground until there is a Gaussian that includes them with sufficient, consistent evidence supporting it.

All these approaches are limited on sequences with a non-moving and almost static background. They will fail on sequences captured by a moving camera.

Another group of motion estimation techniques is based on tracking image features over a sequence of images, e.g. points with prominent features or edges. However, the density of tracked points or edges is often not sufficient for motion-based segmentation without scene knowledge.

Heisele, Kreßel and Ritter [HKR97] present an approach based on pixel classification. The first image is initial clustered by a k-means algorithm with both pixels color and location as features. The cluster centers are each projected in the next frame for detecting membership of each pixel to a certain cluster center. Now each cluster center is updated with the new pixels features. The temporal movement of the location of cluster centers gives the motion trajectory. This approach can not handle sequences with rapidly moving objects.

1.2. Our approach

Detection of moving objects is a key function in a driving assistance system. If object detection is based on motion segmentation, the first essential step is tracking object parts or object features. Basically there are two different problems to be addressed in feature tracking: Which image features can be tracked successfully (high quality and high performance) and how well do these features describe the objects being tracked. We have shown in [Reh98], that colored regions, determined by color segmentation [RP98], yield very promising results in detecting and tracking moving vehicles on highway sequences captured by a non-stationary camera in a moving car. There are two main advantages to this method: good object descriptions achieved by color segments[1], and early reduction of data in the segmentation step, which speeds up the tracking process. So the task of motion segmentation is simplified to grouping a few hundred color segments instead of thousands of pixels.

[1] A segment is a connected set of pixels, which are pairwise similar in color.

Furthermore there is no need on any scene assumption as static camera, slow motion, rigid objects, or smooth motion vector fields.

A segmentation of natural, moving objects leads to differing results from frame to frame – so the simple idea of tracking single segments fails. We introduced so called n:m–Matching [Reh98] for solving the correspondence problem. This deals with sets of segments (segment clusters) instead of single segments; so compositions and decompositions of color segments are considered (see fig. 2):

An improvement of the n:m–Matching is *Segment Cluster Tracking*, which uses information about previous matches in the current match. The main idea is based on the assumption and practical justification, that there is a higher probability that those segment clusters of frame $\mathcal{F}_{t-1}$ (at time $t-1$), which were already tracked in the past ($t-2$), will get a match partner in the current frame $\mathcal{F}_t$, than any other, arbitrarily generated segment clusters. Hence those segment clusters were processed at first; in general most of segment clusters get a match partner here. The remaining segments - in general a very small set - were processed in the following, which is combining to clusters and finding each a match partner.

Although the complexity class (theoretical worst case) of Segment Cluster Tracking is the same as in n:m–Matching, namely $O(2^\nu \cdot 2^\nu)$ with ν the number of segments in one frame, the work in practice is distinctly reduced.

Section 2 describes the basic concepts of *Segment Cluster Tracking*, section 3 explains its algorithm, and section 4 presents its evaluation.

2. The Method

2.1. Fundamentals

The initial step of region based motion estimation algorithms is to partition images into regions. For this purpose a a non-linear filter (SNN - Symmetric Nearest Neighbor, [PH86]) and a color segmentation (CSC - Color Structure Code, [RP98]) is used. But no segmentation algorithm can prevent changes of the segmentation result of real objects (e.g. cars) over time. So objects may consist of a different number of segments with instable shape and size in consecutive frames. So a simple approach of detecting correspondences of single segments in one frame to single segments in the consecutive frame will fail (see 5).

So the way is to find best correspondences between sets of segments in consecutive frames of a sequence. But it is an exponential complexity ($O(2^\nu)$, with ν the number of segments), to create all possible sets of segments (power set of the set of segments), which can not be calculated in real-time.

2.2. The n:m–Matching

One way to handle the complexity is the so called n:m-Matching[2] [Reh98], which uses two effective concepts for reducing the work:

1. **Find clear matches first:** Before creating any segment clusters try to find a clear matching partner with a very high match quality for each single segment (1:1–Matching). After that step segments with no clearly matching partner remain for clustering.

2. **Create only homogenous, connected sets of segments:** Only those segment clusters are generated, which are connected and whose involved segments are similar in color[3] to each other.

The task of n:m–Matching is finding the best corresponding segment cluster for each segment cluster in the consecutive frame. In case of a conflict, i. e. different, non-disjunct[4] clusters have got a match partner, the correspondence with the better match quality is chosen.

2.3. Segment Cluster Tracking

Segment Cluster Tracking is an improvement of the prior approach called n:m-Matching. As in the n:n-Matching only those sets of segments are regarded, which are connected and whose segments are similar in color. These sets are called segment clusters and are described as a feature vector containing information about size, mean color, position, shape, and neighborhood relations to other segment clusters.

Because of combining segments to clusters, a single segment can belong to many different cluster, but of course only one of these clusters will get a match partner; most clusters are only temporary created and do not get a match partner. So it is not necessary to create all possible clusters for solving the correspondence problem.

Our new idea is based on the assumption and practical justification, that there is a higher probability that certain segment clusters of frame $\mathcal{F}_{t-1}$ (at time $t-1$), which were already tracked in the past ($t-2$), will get a correspondence partner in the current frame $\mathcal{F}_t$, than any other, arbitrarily generated segment cluster. So at first only those clusters of $\mathcal{F}_{t-1}$ were regarded which already have a correspondence partner in the past ($t-2$) for finding a correspondence partner in the following frame $\mathcal{F}_t$. This leads to a consecutive tracking. Following a search for partners of not yet matched segments and new built clusters runs.

[2] Try to match sets of n segments in one frame with m segments in an other frame, $1 \leq n, m \leq M$, $M \approx 5$.

[3] With a weaker similarity criterion as in the color segmentation

[4] Non–empty intersection of clusters (sets of segments).

This approach speeds up the algorithm and does not worsen the good tracking quality [Ros03].

For final detection of moving objects, motion vectors are calculated out of the correspondence information and then segments and clusters with similar motion are combined to form a motion object.

3. The algorithm

As a result of processing the previous frames $\mathcal{F}_{t-2}$ and $\mathcal{F}_{t-1}$ at time $t-1$ there is a set of successfully matched segment clusters $\mathcal{C}_{t-1}$ and possibly a set of segments $\mathcal{S}_{t-1}$ which have not got any match partner in the past. The preprocessing of the current frame $\mathcal{F}_t$ provides a set of segments $\mathcal{S}_t$.

The first matching step of *Segment Cluster Tracking* is to detect correspondences between clusters $c \in \mathcal{C}_{t-1}$ and segments $s \in \mathcal{S}_t$ with a very high quality. To avoid creating unused segment clusters in $\mathcal{F}_t$ the strategy of first finding 'clear' matches is used as in the n:m–Matching mentioned above. So it is tried to find match partners for existing clusters of $/Framet - 1$ in the set of single segments $\mathcal{S}_t$ instead of directly creating new segment clusters.

After this and after all following matching steps the respective sets (after this step: $\mathcal{C}_{t-1}$, $\mathcal{S}_t$) are reduced as:

$$\mathcal{S}'_t = \mathcal{S}_t - \{s \in \mathcal{S}_t \mid s \text{ has no correspondence in } \mathcal{F}_{t-1}\}.$$

Normally the set of segments $\mathcal{S}_t$ is reduced by 10-20% after this matching phase.

The remaining segments of $\mathcal{S}_t$ (which is the set $\mathcal{S}'_t$) are then used to create the set of clusters $\mathcal{C}_t$ and the second matching step between elements of $\mathcal{C}_{t-1}$ and $\mathcal{C}_t$ follows. At this stage, most of the clusters in $\mathcal{C}_{t-1}$ should be tracked successfully. Note that only in one frame clusters are computed in contrast to n:m-Matching which creates clusters in two frames after its 1:1-Matching is finished.

Now the participating segments of the remaining clusters in $\mathcal{C}_{t-1}$ are added to the set $\mathcal{S}_{t-1}$ of not backwards matched segments

$$\mathcal{S}'_{t-1} = \mathcal{S}_{t-1} \cup \{s \in c \mid c \in \mathcal{C}_{t-1}\}.$$

The processing step of extracting participating segments of clusters is called 'split'.

A third matching step detects clear matches (high quality is required) between $\mathcal{S}'_{t-1}$ and $\mathcal{C}_t$. The remaining segments of $\mathcal{S}'_{t-1}$ are used to create a new set of clusters $\mathcal{C}'_{t-1}$ [5] and a last matching step finds correspondences between $\mathcal{C}'_{t-1}$ and $\mathcal{C}_t$.

[5] Note the difference: clusters of $\mathcal{C}_{t-1}$ have backward information, clusters of $\mathcal{C}'_{t-1}$ not.

Note that the sets $\mathcal{C}_{t-1}$ and $\mathcal{C}'_{t-1}$ are much more smaller in comparison to the set of clusters $\mathcal{C}^{n:m}_{t-1}$ in n:m–Matching, $\mathcal{C}_{t-1} + \mathcal{C}'_{t-1} \subset \mathcal{C}^{n:m}_{t-1}$, which explains the better computation time.

Figure 1 gives an overview of processing steps and information flow of the *Segment Cluster Tracking* algorithm.

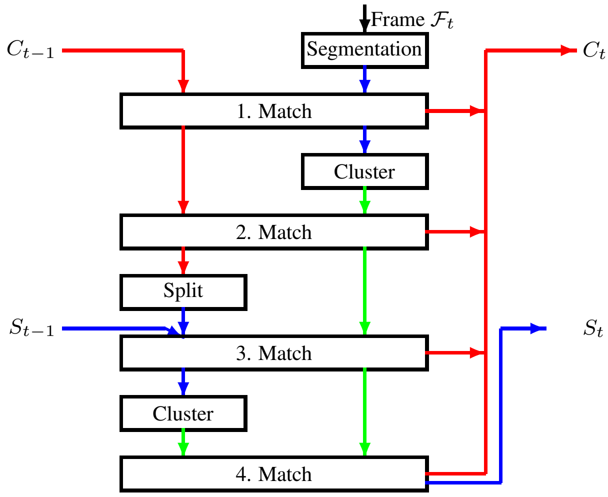

Figure 1: Algorithm of Segment Cluster Tracking
Processing steps and information flow at time t. The sets C_{t-1} and S_{t-1} are a result of prior time step.
red: set of matched segment clusters
blue: set of not matched segments
green: set of not matched segment clusters

4. Evaluation

4.1. Computation Time

The computation time was analyzed with 4 sequences of about 300 frames of size 340×275 (RGB). The average time of preprocessing each frame was about 80 ms (filter 15 ms, segmentation 55 ms, feature extraction 10 ms)[6].

Table 1 shows the comparison of the average computation time for n:m–Matching and Segment Cluster Tracking, its maximum time, and its standard deviation. The Segment Cluster Tracking shows the better performance. The average computation time is reduced of about the half time in comparison to n:m–Matching. Also the variance and the maximum values are significantly reduced. The system is working with about 10 fps[7].

4.2. Tracking Quality

The first two steps of *Segment Cluster Tracking* refer to tracking of 'known' clusters with information of previ-

[6] Common personal computer: Intel Pentium 4 with 2.4 GHz and 512 MB, SuSE Linux 8.2

[7] frames per second

	μ	σ	max.
n:m–Matching	18,6 ms	3,2 ms	23,4 ms
Segment Cluster Tracking	9,4 ms	1,6 ms	11,1 ms
n:m–Matching	20,3 ms	3,5 ms	44,7 ms
Segment Cluster Tracking	10,4 ms	2,6 ms	30,9 ms
n:m–Matching	19,0 ms	2,2 ms	24,7 ms
Segment Cluster Tracking	9,8 ms	1,1 ms	12,7 ms
n:m–Matching	18,7 ms	2,7 ms	20,6 ms
Segment Cluster Tracking	9,7 ms	1,4 ms	10,8 ms

Table 1: Comparison of computation times of n:m–Matching and Segment Cluster Tracking (4 sequences, over 300 frames). In each case average μ, standard deviation σ and maximum of the computation time are given.

ous frames, e. g. a motion trajectory or a temporal trend of size. This information can be used powerfully with certain constraints, e. g. a smoothness constraint for the temporal gradient of the size: then the best match partner for a cluster $c \in \mathcal{C}_{t-1}$ is not necessarily that cluster $c' \in \mathcal{C}_t$ with the same size as c, but that cluster $c'' \in \mathcal{C}_t$, which leads to a monotonic smooth size variation. Thus the task changes from finding a cluster $c_i \in \mathcal{C}_t$ similar to cluster $c \in \mathcal{C}_{t-1}$ to finding a cluster $c_j \in \mathcal{C}_t$ similar to a prediction $p(c)$ of cluster c. Obviously this approach leads to a better tracking quality.

For the purpose of evaluating this new tracking approach a certain quality measure [Ros03] was developed with a tri–state hand segmentation as a ground truth. There images were segmented manually in 3 regions: clear foreground (object(s) to be tracked), clear background, and undefined (e. g. unclear transitions). The quality $q \in [0, 1]$ is defined as

$$q = \frac{t - e_-}{t + e_+}$$

with t is the true number of pixels,e_+ and e_- are the respectively positive and negative errors of automatic object tracking. The efficiency of this quality measure is shown in [Ros03].

First experiments have shown an advancement of the tracking quality of Segment Cluster Tracking in comparison to n:m–Matching.

5. Conclusion

Segment Cluster Tracking has both better computation time and the possibility of using backward information for improving the tracking quality in comparison to n:m–Matching.

As there is no need on any scene assumption like static camera, slow motion, rigid objects, or smooth motion vector fields, Segment Cluster Tracking is an universal, realtime tracking method.

References

[EX01] Tim Ellis and Ming Xu. Object detection and tracking in an open and dynamic world. In *Proceedings 2nd IEEE Int. Workshop on PETS, Kauai, Hawaii*, December 2001.

[HKR97] Bernd Heisele, U. Kreßel, and Werner Ritter. Tracking non-rigid, moving onjects based on color cluster flow. *IEEE Conference on Computer Vision and Pattern Recognition, San Juan*, pages 253–257, 1997.

[PH86] M. Pietikäinen and D. Harwood. Segmentation of color images using edge-preserving filters. In *Advances in Image Processing and Pattern Recognition*, pages 94–99, 1986.

[Reh98] Volker Rehrmann. Object oriented motion estimation in color image sequences. In *ECCV 1998, Vol. I, LNCS 1406*, pages 704–719, 1998.

[Ros03] Mark Ross. Evaluation and improvement of region–based motion segmentation. In *8th Workshop on Vision, Modeling, and Visualization*, pages 55–61. Akademische Verlagsgesellschaft Aka GmbH, Berlin, 2003.

[RP98] Volker Rehrmann and Lutz Priese. Fast and robust segmentation of natural color scenes. In *Proc. of 3rd Asian Conf. on Computer Vision, Special Session on Advances in Color Vision*, volume I, pages 704–719. Springer Verlag, 1998.

[SG98] Chris Stauffer and W.E.L Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, 1998.

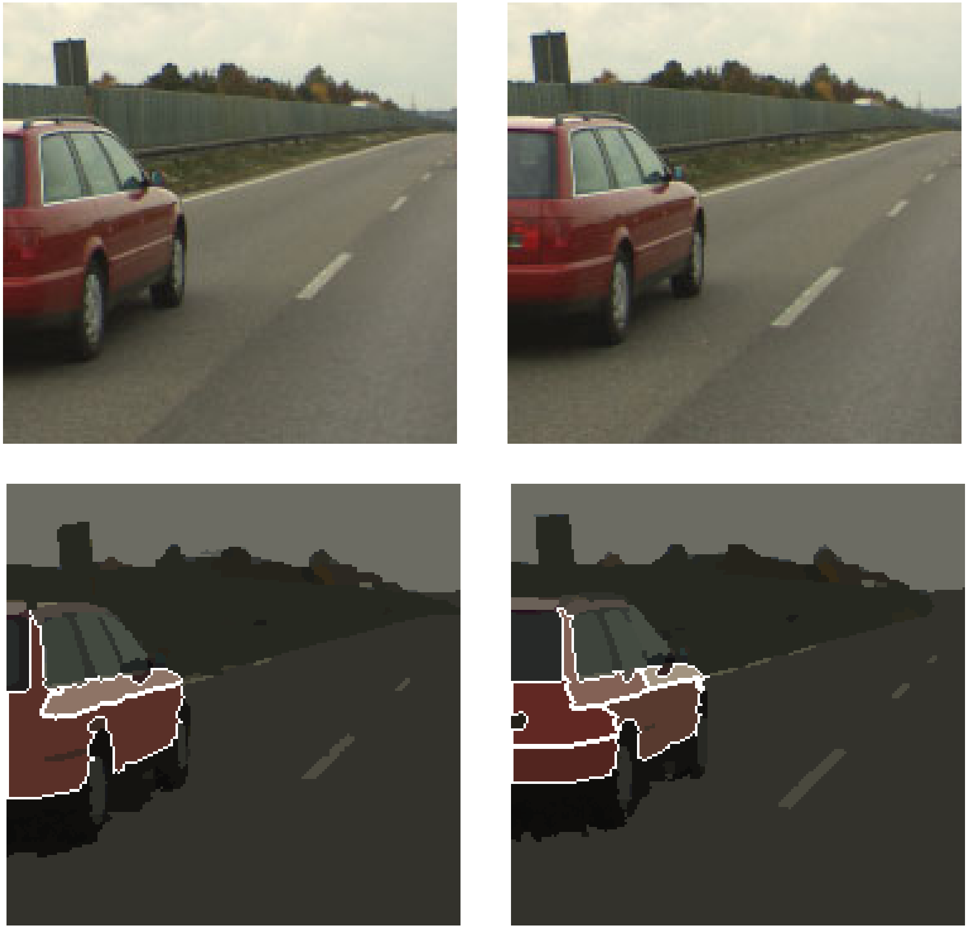

Figure 2: Example of a segment cluster match
A part of car body falls in two consecutive frames at time $t-1$ (left) in 2 segments and at time t (right) in 5 segments. The frames are shown as original (top) and as region image (bottom), where segments of corresponding clusters are visualized by a white border on a shaded background.