Online Motion Segmentation using Spatially-constrained J-linkage in Dynamic Scene

Jungwon Kang, Sang Un Park and Myung Jin Chung

Abstract— **We present a method for online motion segmentation in dynamic scenes. Here the dynamic scene is a scene without restrictions on the motion of objects and the motion of a camera. Such a scene causes complex interaction of objects and a camera, leading to the generation of trajectories corrupted by noise and outliers. Moreover, no prior knowledge of the number of objects is given, and the number of objects can vary as the motion of objects. In this paper, we deal with clustering of trajectories in dynamic scenes, while estimating the number of objects at the same time. The basic idea is to find motion models that support the motion of trajectories, and cluster the trajectories according to the support motion models, instead of handling trajectories directly for clustering. To do so, we adopted online J-linkage algorithm proposed in [7], an online multiple model estimation method. Based on the observation that points on one object are located nearby, we applied spatially-constrained agglomerative clustering to the J-linkage algorithm. This spatial constraint can drastically reduce searching time in clustering. The proposed method operates in an online and incremental fashion, so that it is applicable to real-time applications. We tested our method on the Hopkins datasets, demonstrating the effectiveness of the method in dynamic scenes.**

I. INTRODUCTION

Motion-based object segmentation, commonly called *motion segmentation* in computer vision community, is a problem of segmenting or clustering objects according to their motions. The motion segmentation is an essential step for many applications such as driver assistance [1], robot navigation [2], video surveillance, and human-computer interface. Extensive literatures of the motion segmentation are summarized in [3]. Among a variety of approaches of the motion segmentation, one interesting approach is to cluster trajectories of feature points according to motion of the trajectories, where the trajectories are generated by tracking a set of feature points through a seqeunce of frames. Each output cluster is a set of trajectories with similar motion. The clustering output shows objects of interest, in the form of sparse points. The output can be further processed to provide dense pixel-wise regions [4][5] or bounding boxes [6].

Many applications need to operate in dynamic scenes. Here the *dynamic scene* is a scene where no restriction on the motion of objects and the motion of a camera is given. In such a scene, by the movement of an object itself or the view change by the camera movement, objects can appear or disappear in the scene. The interaction between objects causes occlusion in the scene. These situations cause trajectories corrupted by noise and outliers. Moreover, each trajectory has its own starting and ending frame in time. Furthermore, no prior knowledge of the number of objects is given, and the number of objects can vary as the motion of objects. The motion segmentation in such a scene is a challenging problem. In this paper, we deal with motion-based clustering of trajectories of feature points in dynamic scenes, while estimating the number of objects at the same time. In addition, the proposed method needed to run in an online fashion in order to be applied to real-time applications. This is the problem of clustering, at current frame f_n, trajectories $\mathbf{X}$ visible at the frame f_n into k clusters while estimating k. Here, $\mathbf{X} = \{\mathbf{x_i}\}$ where the ith trajectory $\mathbf{x_i} = \{(u_i^f, v_i^f)\}_{f=f_s^i,\cdots,f_n}$ that means $\mathbf{x_i}$ consists of points from f_s^i, the starting frame of $\mathbf{x_i}$ to the current frame f_n. Each point of $\mathbf{x_i}$ might be corrupted by noise or outliers.

The basic idea for motion segmentation in dynamic scenes is to find motion models that support the motion of trajectories, and cluster the trajectories according to the support motion models. Based on this idea, we can treat motion segmentation as a multiple motion model estimation problem. As a framework for multiple model estimation, we adopted online J-linkage algorithm [7] that finds multiple models from the given samples using agglomerative clustering [8] of the samples. Based on the observation that samples from one model tend to be close, the online J-linkage used the k-NN to limit the search in agglomerative clustering. To generalize the online J-linkage that uses the k-NN, we applied spatially-constrained agglomerative clustering to the J-linkage. Then, we performed clustering of trajectories on the spatially-constrained J-linkage framework. Compared to methods [9] that performed clustering on subspace of trajectories, our method can cluster trajectories from dynamic scenes as we focus on finding motion models instead of handling the trajectories directly.

The contribution of this paper is two-fold. First, we generalized the online J-linkage that uses the k-NN to the spatially-constrained J-linkage. Second, we presented a method of clustering trajectories from dynamic scenes on the spatially-constrained J-linkage framework.

This work was supported by the Korea Evaluation Institute of Industrial Technology (KEIT). (The Development of Low-cost Autonomous Navigation Systems for a Robot Vehicle in Urban Environment, 10035354)

Jungwon Kang and Sang Un Park are Ph.D candidates of the Department of Electrical Engineering, KAIST, 291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Republic of Korea kctown99@gmail.com, psu@rr.kaist.ac.kr

Myung Jin Chung is a professor of the Department of Electrical Engineering, KAIST, 291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Republic of Korea mjchung@ee.kaist.ac.kr

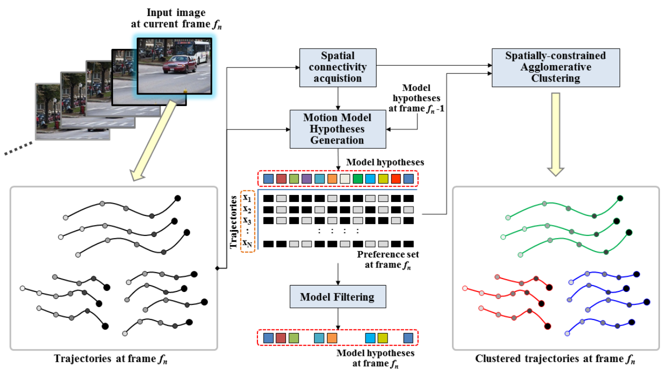

Input image
at current frame f_n
Spatial connectivity acquistion
Spatially-constrained Agglomerative Clustering
Model hypotheses at frame f_n -1
Motion Model Hypotheses Generation
Model hypotheses
Trajectories
x_1
x_2
x_3
x_N
Preference set at frame f_n
Model Filtering
Model hypotheses at frame f_n
Trajectories at frame f_n
Clustered trajectories at frame f_n

Fig. 1. An overview of the proposed method that clusters trajectories in an online and incremental fashion on the spatially-constrained J-linkage framework.

II. OVERVIEW

Fig. 1 shows an overview of our algorithm. Given I_{f_n}, an image at current frame f_n, trajectories visible at frame f_n are stored in the trajectory queue $Q_{\mathbf{X}}$. Then, we get spatial connectivity between trajectory points at frame f_n. To get the spatial connectivity, we used the Delaunay triangulation. However, we can use any methods that provide the spatial connectivity. The model hypotheses at frame f_n are generated by sampling of trajectories based on the spatial connectivity, and augmentation from model hypotheses at frame $f_n - 1$. The model hypotheses are stored in the model hypotheses queue $Q_{\mathbf{M}}$. Then, a preference set is generated by matching each trajectory with each model hypothesis. A preference set of a trajectory is a vector whose size equals the number of model hypotheses, and the vector element indicates whether a model hypothesis describes the trajectory well or not. Then, the trajectories represented by a preference set are clustered by the spatially-constrained agglomerative clustering, providing the clustered trajectories and the number of clusters at frame f_n.

III. ONLINE MOTION SEGMENTATION ON SPATIALLY-CONSTRAINED J-LINKAGE

A. *Spatially-constrained J-linkage*

The *J-linkage*, originally proposed in [10], is a multiple model estimation method that finds unknown models from given samples. The J-linkage has been used for plane detection [11] and trajectory clustering [12]. The basic idea of the J-linkage is to cluster samples that are supported by the same model, and find the model from the clustered samples. The J-linkage begins with generation of model hypotheses from samples. Then, each sample is represented by a preference set, where the preference set is generated by matching between the sample and each model hypothesis. The preference set of a sample indicates whether the sample is supported by each model hypothesis or not, and is a vector consisting of 1(supporting) and 0(not supporting). Samples represented by a preference set are clustered by the agglomerative clustering [8]. Starting with samples where each sample is set to a single cluster, the clustering algorithm merges two closest clusters into one merged cluster, forming a binary tree, until there remains only one cluster. During the clustering, the J-linkage uses the Jaccard distance to measure a distance between clusters, and the intersection to define a preference set of the merged cluster. By the Jaccard distance, the distance between clusters is calculated according to the support model hypotheses of each cluster. Two clusters with similar support model hypotheses have a distance close to 0. On the contrary, if each cluster is supported by different model hypotheses, the distance between the two clusters becomes close to 1. Meanwhile, when two clusters are merged into one merged cluster, a preference set of the merged cluster is defined as the intersection of preference sets of two clusters to be merged. Hence, only common model hypotheses that support both two clusters to be merged support the merged cluster. Finally, the output clusters are obtained by cutting the binary tree. The cut-off distance is set to 1 so that samples with at least one common support model hypothesis become a cluster. Then, each final model is estimated from the samples in each output cluster.

The original J-linkage [10] runs in a batch fashion. In [7], it was modified to online J-linkage that runs in an incremental fashion with online input. The online J-linkage was successfully applied to real-time plane detection and segmentation [7]. While the original J-linkage generates model hypotheses at once, the online J-linkage handles model hypotheses dynamically, which means the algorithm generates and removes model hypotheses at each frame. Meanwhile, in order to speed up, the online J-linkage used the k-NN to limit the search in agglomerative clustering. Hence, a cluster is merged only with one of the k-NN of the cluster. This spatial constraint is acceptable as samples belonging to the same model tend to be spatially close.

We have generalized the online J-linkage that uses the k-NN to spatially-constrained J-linkage, by applying spatially-constrained agglomerative clustering to the online J-linkage.

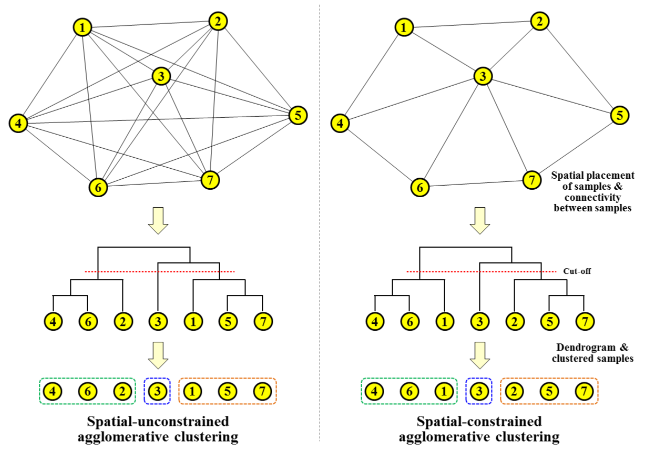

Spatial placement of samples & connectivity between samples
Cut-off
Dendrogram & clustered samples
Spatial-unconstrained agglomerative clustering
Spatial-constrained agglomerative clustering

Fig. 2. Spatially-unconstrained/constrained agglomerative clustering. Spatially-unconstrained clustering has no spatial restriction on choosing samples to be merged as samples are assumed to be fully connected, whereas spatially-constrained clustering selects samples according to the connectivity of the samples.

Algorithm 1 Spatially-constrained agglomerative clustering

```
    ($S$: samples, $L$: connection matrix, $O_1$, $O_2$, $O_3$: options for 'distance', 'cluster merge', 'cutting tree' type, respectively)
 1: procedure SC_AC($S$, $L$, $O_1$, $O_2$, $O_3$)
 2:     $N_S \leftarrow |S|$                                    ▷ $N_S$: # of samples
 3:     $Z \leftarrow [\,]$                                      ▷ $Z$: list for dendrogram
 4:     Calculate distance $D$ of samples $i$, $j$ s.t. $L(i,j) = 1$
 5:     with $O_1$.
 6:     for $l \leftarrow 1, (N_S - 1)$ do
 7:         $(p,q) \leftarrow \arg\min_{(i,j)\backslash(i \neq j)} D(i,j)$ s.t. $L(i,j) = 1$
 8:         $Z \leftarrow Z \cup \{p, q, D(p,q)\}$
 9:         $S \leftarrow S \backslash \{p,q\}$
10:         Define a new cluster $j \notin S$ from $p, q$ with $O_2$.
11:         Update $L$ of $j$.
12:         Update $D$ between $j$ and all $i \in S$ s.t. $L(i,j) = 1$
            with $O_1$.
13:         $S \leftarrow S \cup \{j\}$
14:     end for
15:     Create clusters $C$ by cutting $Z$ with $O_3$.
16:     return $C$
17: end procedure
```

The agglomerative clustering without spatial constraints assumes implicitly that samples are fully connected and selects samples that need to be merged without any spatial limitation, whereas the spatially-constrained agglomerative clustering chooses only spatially connected samples during the merging process. The spatially-constrained agglomerative clustering is summarized in Algorithm 1. An example of showing the difference of two agglomerative clustering is given in Fig. 2. The spatially-constrained J-linkage, which is the online J-linkage with spatially-constrained agglomerative clustering, runs in an incremental fashion with online input. The spatially-constrained J-linkage is presented in Algorithm 2, and its application to the motion segmentation is given in Algorithm 3. In our implementation, we used the Delaunay triangulation to build the connection matrix.

B. Trajectory generation

At current frame f_n, we extract a set of feature points P_{f_n} from I_{f_n}. Each point $\mathbf{x}_i^{f_n - 1}$ of trajectory i is associated with a point p ($p \in P_{f_n}$ or $p \notin P_{f_n}$) on I_{f_n}. Then, the point p is registered in $Q_{\mathbf{X}}$ as a point $\mathbf{x}_i^{f_n}$ of existing trajectory i. If the association fails, the trajectory is removed from $Q_{\mathbf{X}}$. A point $p \in P_{f_n}$ but not associated with the existing trajectories in $Q_{\mathbf{X}}$ is registered in $Q_{\mathbf{X}}$ as a new point. Hence, after all the association, only points and trajectories visible at the current frame f_n are maintained in $Q_{\mathbf{X}}$. We maintain only a limited number of latest points for a trajectory in $Q_{\mathbf{X}}$ so that only the latest points can affect clustering at current frame.

To implement $Q_{\mathbf{X}}$, we adopted a circular queue. For each trajectory, $Q_{\mathbf{X}}$ stores points up to $N_{\mathbf{x}}^{max}$, the maximum number of points for a trajectory in $Q_{\mathbf{X}}$. If a point of a trajectory at current frame f_n is stored in $Q_{\mathbf{X}}$, a point at frame $f_n - N_{\mathbf{x}}^{max}$ is removed from $Q_{\mathbf{X}}$ if the number of stored points is greater than $N_{\mathbf{x}}^{max}$, maintaining the number of points to at most $N_{\mathbf{x}}^{max}$.

Algorithm 2 Online incremental spatially-constrained J-linkage

```
    ($X^f$: input samples up to frame $f$, $L_f$: connection matrix at frame $f$)
 1: procedure OI_SC_J_LINKAGE($X^f$, $L_f$)
 2:     Generate models $M_f$ from $X^f$ using $L_f$.
 3:     Add $M_f$ to model set $M$.
 4:     Build preference set $S_f$ from $X^f$ and $M$.
 5:     $O_1 \leftarrow$ 'Jaccard distance'
 6:     $O_2 \leftarrow$ 'Intersection'
 7:     $O_3 \leftarrow$ '1 as a cut-off threshold'
 8:     $C_f \leftarrow$ SC_AC($S_f$, $L_f$, $O_1$, $O_2$, $O_3$)
 9:     Build final model set $M'$ from $C_f$.
10:     Remove unnecessary models in $M$.
11:     return $C_f, M'$
12: end procedure
```

Algorithm 3 Motion segmentation using online incremental spatially-constrained J-linkage

```
    ($I_f$: an input image at frame $f$)
 1: procedure MOTION_SEGMENTATION($I_f$)
 2:     Generate trajectories $\mathbf{X}^f$ from $I_f$.
 3:     Put $\mathbf{X}^f$ into the trajectory queue $Q_{\mathbf{X}}$.
 4:     Build spatial connectivity matrix $L_f$ from $\mathbf{X}^f$.
 5:     $\{C_f, M_f\} \leftarrow$ OI_SC_J_LINKAGE($\mathbf{X}_f$, $L_f$)
 6:     return $C_f, M_f$
 7: end procedure
```

C. Motion model hypotheses generation

A motion model hypothesis $\mathbf{m}_j$ of motion j describes motions of a point across frames, and consists of models representing a motion between consecutive frames. A motion model for consecutive frames that we adopted is an affine transformation. The affine transformation, described by six parameters $\{m_1, \cdots, m_6\}$, moves a location (u, v) of a point to (u', v') by (1).

$$\begin{aligned} u' &= m_1 u + m_2 v + m_3 \\ v' &= m_4 u + m_5 v + m_6 \end{aligned} \tag{1}$$

The consecutive frame motion model $\mathbf{m}_j^{(f-1,f)}$ is composed of forward parameters $\mathbf{m}_j^{(f-1,f)^+} = \{m_1^+, \cdots, m_6^+\}$ for a motion from frame $f-1$ to f, and backward parameters $\mathbf{m}_j^{(f-1,f)^-} = \{m_1^-, \cdots, m_6^-\}$ for a motion from frame f to $f-1$, Hence, $\mathbf{m}_j = \{\mathbf{m}_j^{(f-1,f)^+}, \mathbf{m}_j^{(f-1,f)^-}\}_{f=f_s+1,\cdots,f_e}$ describes motions from frame f_s to f_e.

Given $\{(u_i^{f-1}, v_i^{f-1}), (u_i^f, v_i^f)\}_{i=1,2,3}$, three pairs of a point set, we can estimate the forward parameters $\mathbf{m}^{(f-1,f)^+}$ by obtaining the least square solution $\mathbf{m}^{(f-1,f)^+} = (\mathbf{A}^+)^{-1}\mathbf{b}^+$, where $\mathbf{m}^{(f-1,f)^+} = [m_1^+, m_2^+, m_3^+, m_4^+, m_5^+, m_6^+]^T$, $\mathbf{A}^+ = [\mathrm{A}_1^+; \mathrm{A}_2^+; \mathrm{A}_3^+]$

with $\mathrm{A}_{\mathrm{i}}^{+} = \begin{bmatrix} u_i^{f-1} & v_i^{f-1} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & u_i^{f-1} & v_i^{f-1} & 1 \end{bmatrix}$, and $\mathbf{b}^{+} = [u_1^f, v_1^f, u_2^f, v_2^f, u_3^f, v_3^f]^T$. The backward parameters $\mathbf{m}^{(f-1,f)^-}$ are estimated, in the similar manner, by solving $\mathbf{m}^{(f-1,f)^-} = (\mathbf{A}^-)^{-1}\mathbf{b}^-$. In our implementation, we utilized vertices of triangles from the Delaunay triangulation for sampling three pairs of a point set.

A motion model hypothesis $\mathbf{m}_j$ of motion j at frame f is generated from three trajectories i, i', i'' by linking a model hypothesis of motion j up to frame $f-1$ with a consecutive frame motion model between frame $f-1$ and f. The generated model hypothesis is stored in $Q_{\mathbf{M}}$. If $|\det(\mathbf{A}^+)|$ or $|\det(\mathbf{A}^-)| < \varepsilon$ where ε is a small constant, e.g. $\varepsilon = 0.01$, we remove the model hypothesis as we can not obtain numerically stable parameters in this case.

D. Preference set generation

A preference set of a trajectory is a vector that indicates whether the trajectory matches well with each motion model hypothesis or not. Given a trajectory i, $\mathbf{x}_i = \{\mathbf{x}_i^f\}_{f=f_s^i,\cdots,f_e^i}$ where $\mathbf{x}_i^f = (u_i^f, v_i^f)$ and a model hypothesis j, $\mathbf{m}_j = \{\mathbf{m}_j^{(f-1,f)}\}_{f=f_s^j,\cdots,f_e^j}$, the preference φ_i^j of a trajectory i to a model hypothesis j is given by 1 if $r(\mathbf{x}_i, \mathbf{m}_j) \le \tau$ or 0 otherwise, where $r(\mathbf{x}_i, \mathbf{m}_j)$ is a residue that indicates how well $\mathbf{x}_i$ matches $\mathbf{m}_j$, and τ is an inlier threshold.

Since the magnitude of motion varies with motion models and points to be transferred, the inlier threshold τ needs to be a variable depending on the motion models and the points. Let $\hat{\mathbf{x}}' = g(\mathbf{x}, \mathbf{m})$ represent that a point $\mathbf{x}$ is transferred to a point $\hat{\mathbf{x}}'$ by a motion model $\mathbf{m}$. Then, the residue r can be defined as $||\mathbf{x}' - \hat{\mathbf{x}}'||$ where $\mathbf{x}'$ is a measured point that moved from $\mathbf{x}$. Letting the magnitude of motion $\zeta = ||\mathbf{x} - \hat{\mathbf{x}}'||$, we define $\tau = \alpha\zeta + \beta$ so that τ becomes a function of $\mathbf{x}$, $\hat{\mathbf{x}}'$ and $\mathbf{m}$, where α, β are constants. If we define new residue $\hat{r} = r/\tau$, then $\hat{r} \le 1$ indicates a good match.

To calculate the preference φ_i^j, we compute the number of overlapping frames between a trajectory i and a model hypothesis j. The number of overlapping frames, n_f is given by $f_e - f_s$, where the starting frame of the overlapping frames $f_s = \max(f_s^i, f_s^j - 1)$ and the ending frame $f_e = \min(f_e^i, f_e^j)$. If $n_f \le 0$, it means there are no overlapping frames. Then, $\varphi_i^j = 1$ if $\hat{r}(\mathbf{x}_i, \mathbf{m}_j) \le 1$ and $n_f > 0$, $\varphi_i^j = 0$ if $\hat{r}(\mathbf{x}_i, \mathbf{m}_j) > 1$ and $n_f > 0$. If $n_f \le 0$, $\varphi_i^j = -1$ which means this match can not be evaluated.

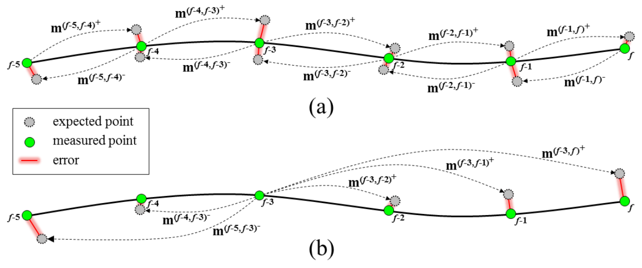
Here, we present two types of $\hat{r}(\mathbf{x}_i, \mathbf{m}_j)$. The first type, given by (2) with (3) and (4), is the mean residue of a set of two consecutive frames.

$$\hat{r}(\mathbf{x}_i, \mathbf{m}_j) = 0.5 \cdot (s^+ + s^-)/n_f \tag{2}$$

$$s^+ = s^+|_{f_s+1}^{f_e} = \textstyle\sum_{f=f_s+1}^{f_e}(e^+/\tau^+) \tag{3}$$

where $e^+ = ||\mathbf{x}_i^f - g(\mathbf{x}_i^{f-1}, \mathbf{m}_j^{(f-1,f)^+})||$, $\tau^+ = \alpha\zeta^+ + \beta$ with $\zeta^+ = ||\mathbf{x}_i^{f-1} - g(\mathbf{x}_i^{f-1}, \mathbf{m}_j^{(f-1,f)^+})||$.

$$s^- = s^-|_{f_s}^{f_e-1} = \textstyle\sum_{f=f_s}^{f_e-1}(e^-/\tau^-) \tag{4}$$



$\mathbf{m}^{(f\text{-}5,f\text{-}4)^+}$
$\mathbf{m}^{(f\text{-}4,f\text{-}3)^+}$
$\mathbf{m}^{(f\text{-}3,f\text{-}2)^+}$
$\mathbf{m}^{(f\text{-}2,f\text{-}1)^+}$
$\mathbf{m}^{(f\text{-}1,f)^+}$
f-5
f-4
f-3
f-2
f-1
f
$\mathbf{m}^{(f\text{-}5,f\text{-}4)^-}$
$\mathbf{m}^{(f\text{-}4,f\text{-}3)^-}$
$\mathbf{m}^{(f\text{-}3,f\text{-}2)^-}$
$\mathbf{m}^{(f\text{-}2,f\text{-}1)^-}$
$\mathbf{m}^{(f\text{-}1,f)^-}$
(a)
expected point
measured point
error
$\mathbf{m}^{(f\text{-}3,f)^+}$
$\mathbf{m}^{(f\text{-}3,f\text{-}1)^+}$
$\mathbf{m}^{(f\text{-}3,f\text{-}2)^+}$
$\mathbf{m}^{(f\text{-}4,f\text{-}3)^-}$
$\mathbf{m}^{(f\text{-}5,f\text{-}3)^-}$
(b)

Fig. 3. The two types of residue $\hat{r}(\mathbf{x}_i, \mathbf{m}_j)$. (a) residue type 1, (b) residue type 2 showing an example when the seed point is a point at the frame $f-3$.

where $e^- = ||\mathbf{x}_i^f - g(\mathbf{x}_i^{f+1}, \mathbf{m}_j^{(f,f+1)^-})||$, $\tau^- = \alpha\zeta^- + \beta$ with $\zeta^- = ||\mathbf{x}_i^{f+1} - g(\mathbf{x}_i^{f+1}, \mathbf{m}_j^{(f,f+1)^-})||$.

The residue in (2) can decompose into the residue at previous frame, currently inputted residue, and the oldest residue, as given in (5) and (6). Thus, the residue in (2) can be computed incrementally, when a new point at current frame comes in $Q_{\mathbf{X}}$ and the oldest point disappears in $Q_{\mathbf{X}}$.

$$s^+|_{f_s+1}^{f_e} = s^+|_{f_s}^{f_e-1} + \Delta_{f_e}^+ - \Delta_{f_s}^+ \tag{5}$$

where $\Delta_f^+ = e^+/\tau^+$ with
$e^+ = ||\mathbf{x}_i^f - g(\mathbf{x}_i^{f-1}, \mathbf{m}_j^{(f-1,f)^+})||$,
$\zeta^+ = ||\mathbf{x}_i^{f-1} - g(\mathbf{x}_i^{f-1}, \mathbf{m}_j^{(f-1,f)^+})||$.

$$s^-|_{f_s}^{f_e-1} = s^-|_{f_s-1}^{f_e-2} + \Delta_{f_e-1}^- - \Delta_{f_s-1}^- \tag{6}$$

where $\Delta_f^- = e^-/\tau^-$ with
$e^- = ||\mathbf{x}_i^f - g(\mathbf{x}_i^{f+1}, \mathbf{m}_j^{(f,f+1)^-})||$,
$\zeta^- = ||\mathbf{x}_i^{f+1} - g(\mathbf{x}_i^{f+1}, \mathbf{m}_j^{(f,f+1)^-})||$.

The second type of $\hat{r}(\mathbf{x}_i, \mathbf{m}_j)$, suggested in [12], is given by (7). To compute (7), a residue r_{f_c} in (8) is obtained by comparing measured trajectory points with points transferred by combinations of the two consecutive frames motion models from a point in the measured trajectory as a seed point. Here, a seed point at frame f_c is transferred to a point at frame f by $\mathbf{m}^{(f,f_c)^-}$ if $f < f_c$ and $\mathbf{m}^{(f_c,f)^+}$ if $f > f_c$, where $\mathbf{m}^{(f,f_c)^-}$ is a combination of the backward two consecutive frames motion models from frame f_c to f, $\mathbf{m}^{(f_c,f)^+}$ is a combination of forward models. Then, we get total $n_f + 1$ residues of r_{f_c} as the seed point frame f_c varies from f_s to f_e. Among the $n_f + 1$ residues, the minimum residue is set as $\hat{r}(\mathbf{x}_i, \mathbf{m}_j)$. While the first type residue in (2) depends on motion between two consecutive frames only, the residue in (7) exploits motion between multiple frames.

$$\hat{r}(\mathbf{x}_i, \mathbf{m}_j) = \min\{r_{f_c}\}_{f_c=f_s,\cdots,f_e} \tag{7}$$

$$r_{f_c} = (s^+ + s^-)/n_f \tag{8}$$

$$s^- = s^-|_{f_s}^{f_c-1} = \textstyle\sum_{f=f_s}^{f_c-1}(e^-/\tau^-) \tag{9}$$

where $e^- = ||\mathbf{x}_i^f - g(\mathbf{x}_i^{f_c}, \mathbf{m}^{(f,f_c)^-})||$,
$\zeta^- = ||\mathbf{x}_i^{f_c} - g(\mathbf{x}_i^{f_c}, \mathbf{m}^{(f,f_c)^-})||$.

$$s^+ = s^+|_{f_c+1}^{f_e} = \textstyle\sum_{f=f_c+1}^{f_e}(e^+/\tau^+) \tag{10}$$

where $e^+ = ||\mathbf{x}_i^f - g(\mathbf{x}_i^{f_c}, \mathbf{m}^{(f_c,f)^+})||$,
$\zeta^+ = ||\mathbf{x}_i^{f_c} - g(\mathbf{x}_i^{f_c}, \mathbf{m}^{(f_c,f)^+})||$.

The residue in (7) can also be computed incrementally by rearranging (9) to (11) and (10) to (12).

$$s^-|_{f_s}^{f_c-1} = s^-|_{f_s-1}^{f_c-1} - \Delta^-_{f_s-1} \tag{11}$$

where $\Delta^-_{f_s-1} = e^-/\tau^-$ with
$e^- = ||\mathbf{x}_i^{f_s-1} - g(\mathbf{x}_i^{f_c}, \mathbf{m}^{(f_s-1,f_c)^-})||$,
$\zeta^- = ||\mathbf{x}_i^{f_c} - g(\mathbf{x}_i^{f_c}, \mathbf{m}^{(f_s-1,f_c)^-})||$.

$$s^+|_{f_c+1}^{f_e} = s^+|_{f_c+1}^{f_e-1} + \Delta^+_{f_e} \tag{12}$$

where $\Delta^+_{f_e} = e^+/\tau^+$ with
$e^+ = ||\mathbf{x}_i^{f_e} - g(\mathbf{x}_i^{f_c}, \mathbf{m}^{(f_c,f_e)^+})||$,
$\zeta^+ = ||\mathbf{x}_i^{f_c} - g(\mathbf{x}_i^{f_c}, \mathbf{m}^{(f_c,f_e)^+})||$.

E. Model filtering

After building a preference set, unnecessary model hypothesis are removed from $Q_\mathbf{M}$. The first unnecessary one is the model hypothesis that has no overlapping frames with all trajectories. The model hypothesis is removed as it can be no longer used to calculate the preference. The second one is the model hypothesis that supports very small set of trajectories. Such a model hypothesis is likely to represent a wrong motion.

IV. EXPERIMENTAL RESULT

A. Datasets & experimental setup

We have tested our algorithm on the Hopkins 155 dataset and the Hopkins 155+16 dataset. The Hopkins datasets[1], designed for motion segmentation, provide trajectories and their ground-truth labels. The detail of the datasets is presented in [13]. The Hopkins 155 dataset consists of a total of 155 sequences of 'checkerboard', 'traffic' and 'articulated/non-rigid' sequences. Each sequence has two or three motions. The dataset provides trajectories with almost no noise and identical starting and ending frames. Hence, the dataset was used for evaluating the clustering performance itself. On the other hand, the 155+16 dataset provides 16 sequences that contain missing trajectories, outliers, and more than three motions. Thus, the 155+16 dataset was used for testing the algorithm in dynamic scenes.

To evaluate the clustering performance, we calculated the pair-counting precision P, recall R, and F-measure F of the clustering results, where $F = (P \cdot R)/(P + R)$. The pair-counting measure is applied to the set of sample pairs, and calculated according to the expected clustering of the sample pairs and the ground-truth clustering. These measures are appropriate here as the measures enable us to compare clusterings with different numbers of clusters.

In the experiment on the Hopkins 155 dataset, we compared our method with three other existing methods[2] including GPCA, RANSAC, LSA, introduced in [13]. The results were compared at the last frame of a sequence so that both

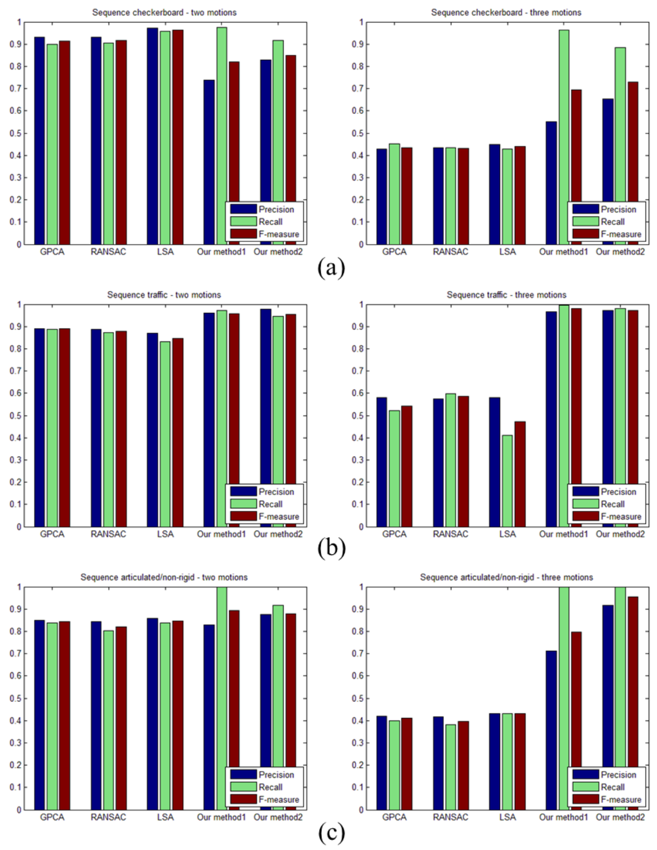


Fig. 4. Average pair-counting precision, recall and F-measure on (a) checkerboard, (b) traffic, (c) articulated/non-rigid seqeunces of the Hopkins 155 dataset. The graphs on the left side show results on two motions, the right side indicates results on three motions. The 'Our method1' is our method with the residue type 1, and the 'Our method2' is with the residue type 2.

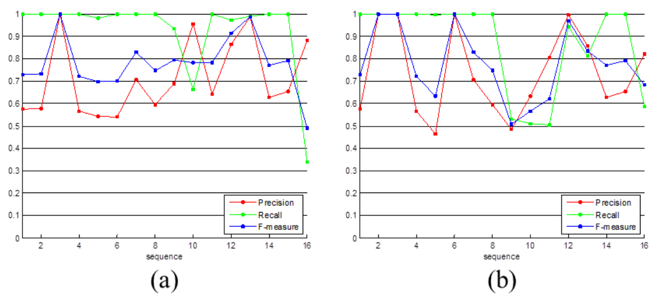


Fig. 5. Pair-counting precision, recall and F-measure of our method with (a) residue type 1, (b) residue type 2 on each seqeunce of the Hopkins 155+16 dataset.

our algorithm and the other methods performed clustering with trajectories from all the frames. While our method estimated the number of motions, the number of motions was given to the other methods in advance. In the case of the experiment on the 155+16 dataset, we tested only our algorithm as the other methods does not work on the missing and incomplete trajectories. In the experiment, we set $N^{max}_\mathbf{x}$, the maximum number of points for a trajectory in $Q_\mathbf{X}$ to the number of frames in a sequence, and α to 0.1, β to 2.5 pixel for the inlier threshold τ.

[1]The datasets are available at http://www.vision.jhu.edu/data/hopkins155/
[2]The code is available at http://www.vision.jhu.edu/code/

B. Results on Hopkins dataset & Discussion

The results on the Hopkins 155 dataset are given in Fig. 4. One interesting thing is the change of performance according to the number of motions. As the number of motions increased, the performance of the three existing methods decreased rapidly, while our method almost kept the performance although there were slight variations.

The results of our method were strongly affected by the residue type. If the motions of objects between two consecutive frames are not discriminative, as seen in the sequence 'checkerboard', the use of the residue type 1 that entirely depends on the motion between two consecutive frames does not work well. This is the reason of poor performance on the sequence 'checkerboard'. On the other hand, the three existing methods were less affected by motion between two consecutive frames as they clustered trajectories gathered across all the frames. The use of the residue type 2 that exploits motion between multiple frames can alleviate the problem of the residue type 1. The performance when using the residue type 2 was generally better than when using the residue type 1, in Fig. 4. However, the residue type 2 is sensitive to noise in trajectories, because the noisy point influences calculation of the residue through the entire frames. This is why the use of residue type 2 did not outperform the use of residue type 1 in the experiment on the 155+16 dataset, as shown in Fig. 5.

The clustering results of our algorithm on some sequences of the Hopkins 155 dataset and 155+16 dataset are shown in Fig. 6 and Fig. 7, respectively. Our algorithm runs in one second per frame on an unoptimized MATLAB code.

V. SUMMARY AND FUTURE WORK

To deal with motion segmentation in dynamic scenes, we treated motion segmentation as a multiple model estimation problem. As a framework for multiple model estimation, we adopted the online J-linkage algorithm. We generalized the online J-linkage algorithm to the spatially-constrained J-linkage. Then, we clustered trajectories from dynamic scenes on the spatially-constrained J-linkage framework. We verified our method on the Hopkins dataset. Our algorithm runs in an incremental fashion, and handles online input data. Hence, it is applicable to real-time applications.

The future works include clustering that has temporal consistency of clustering results. Although our algorithm incrementally calculates the preference set of trajectories, the clustering itself is newly performed at each frame, leading to the clustering result that does not have temporal consistency. The change of 2D motion model to 3D is also an interesting challenge. If 3D motion model is used, it could be expanded to multibody structure-from-motion [14].

REFERENCES

[1] A. Barth and U. Franke, "Estimating the driving state of oncoming vehicles from a moving platform using stereo vision," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 10, No. 4, pp. 560 - 571, December 2009.

[2] A. Kundu, K. M. Krishna, J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4306 - 4312, 2009.

[3] L. Zappella, X. Llado and J. Salvi, "New trends in motion segmentation," *Pattern Recognition*, Peng-Yeng Yin (Ed.), ISBN: 978-953-307-014-8, InTech, 2009.

[4] J. Kang, S. Kim, T. J. Oh, M. J. Chung, "Moving region segmentation using sparse motion cue from a moving camera," *The 12th International Conference on Intelligent Autonomous Systems*, Jeju Island, Korea, June 26-29, 2012.

[5] A. Bugeau and P. Perez, "Detection and segmentation of moving objects in complex scenes," *Computer Vision and Image Understanding*, Vol. 113, Issue. 4, pp. 459 - 476, April 2009.

[6] A. Basharat, Y. Zhai, M. Shah, "Content based video matching using spatiotemporal volumes," *Computer Vision and Image Understanding*, Vol. 110, Issue 3, pp. 360 - 377, June 2008.

[7] R. Toldo, A. Fusiello, "Real-time incremental J-linkage for robust multiple structures estimation," *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, 2010.

[8] D. Müllner, "Modern hierarchical, agglomerative clustering algorithms," *Lecture Notes in Computer Science*, 3918(1973):29, 2011.

[9] R. Vidal, "Subspace clustering," *IEEE Signal Processing Magazine*, Volume 28, Issue 2, March 2011.

[10] R. Toldo, A. Fusiello, "Robust multiple structures estimation with J-linkage," *Proceedings of the 10th European Conference on Computer Vision (ECCV)*, pp. 537 - 547, 2008.

[11] D. Fouhey, D. Scharstein, A. Briggs, "Multiple plane detection in image pairs using J-linkage," *International Conference on Pattern Recognition (ICPR)*, 2010.

[12] M. Fradet, P. Robert, P. Perez, "Clustering point trajectories with various life-spans," *Conference for Visual Media Production*, pp. 7 - 14, 2009.

[13] R. Tron and R. Vidal, "A benchmark for the comparison of 3-D motion segmentation algorithms," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1 - 8, 2007.

[14] K. E. Ozden, K. Schindler, and L. van Gool, "Multibody structure-from-motion in practice," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1134 - 1141, 2010.

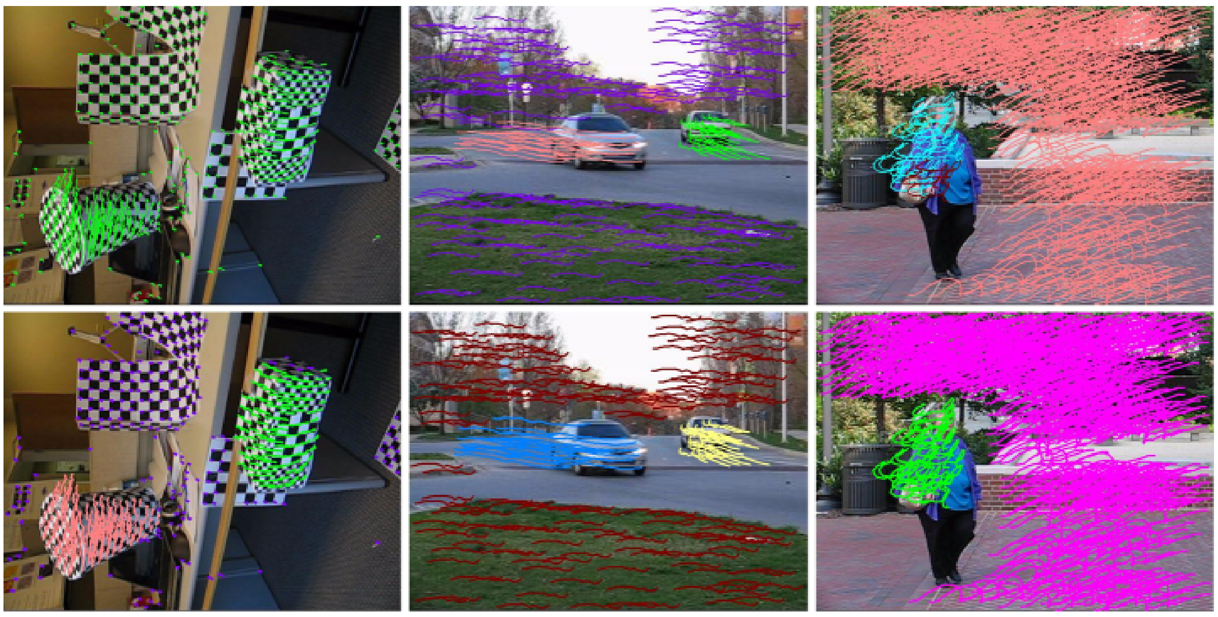


Fig. 6. Sequence 1R2RC, cars2_07, people2 (from left to right) of the Hopkins 155 dataset, overlayed with clustered trajectories. The images at the upper side are results with the residue type 1, and the images at the bottom are from the residue type 2. The trajectories with the same color belong to the same cluster.

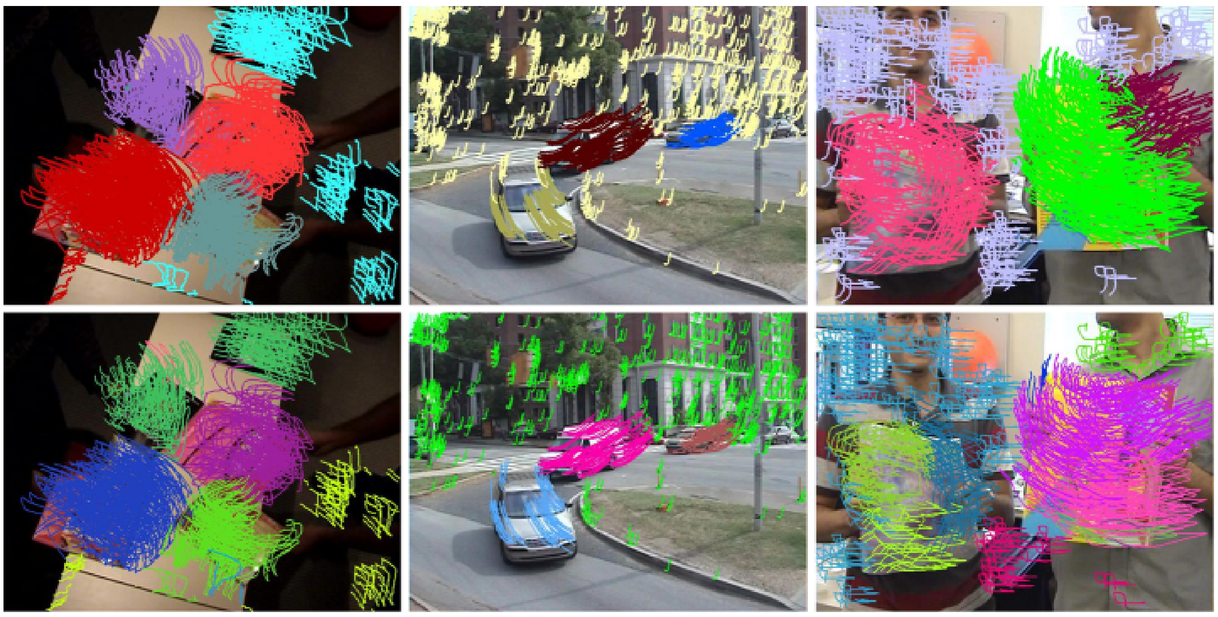


Fig. 7. Sequence books, carsTurning, nrbooks3 (from left to right) of the Hopkins 155+16 dataset, overlayed with clustered trajectories. The images at the upper side are results with the residue type 1, and the images at the bottom are from the residue type 2. The trajectories with the same color belong to the same cluster.