Journal of Intelligent Learning Systems and Applications, **2011, 3, 45-54**
doi:10.4236/jilsa.2011.31006 Published Online February 2011 (http://www.SciRP.org/journal/jilsa)

Apply GPCA to Motion Segmentation

Hongchuan Yu, Jian Jun Zhang

National Centre for Computer Animation, The Media School, Bournemouth University, Poole, UK.
Email: {hyu, jzhang}@bournemouth.ac.uk

Received October 14th, 2010; revised November 25th, 2010; accepted December 7th, 2010

ABSTRACT

In this paper, we present a motion segmentation approach based on the subspace segmentation technique, the generalized PCA. By incorporating the cues from the neighborhood of intensity edges of images, motion segmentation is solved under an algebra framework. Our main contribution is to propose a post-processing procedure, which can detect the boundaries of motion layers and further determine the layer ordering. Test results on real imagery have confirmed the validity of our method.

Keywords*: *Generalized PCA, Motion Segmentation, Layer Ordering

1. Introduction

An important problem in computer vision is to segment moving objects of a scene from a video source, and partly recover the structure or motion information, such as foreground and background. With widespread demands on video processing, motion segmentation has found many direct applications. Video surveillance systems seek to automatically identify people, objects, or activities of interest in a variety of environments with a set of stationary cameras. Motion segmentation can provide low level motion detection and region tracking cues. Another relatively new application is markerless motion capture for computer animation. It aims to estimate the human body configuration and pose in the real world from a video by locating the joint positions over time and extracting the articulated structure.

Motion segmentation is expected to partly recover the structure and motion information of moving objects from a mutually occluded scene. This includes the following main tasks, (1) labeling the regions of a motion layer segmentation, *i.e.* pixels are assigned to several motion layers; (2) finding their motion, e.g. each layer has its own smooth flow field while discontinuities occur between layers; (3) determining the layer ordering, as the different layers might occlude each other. But motion segmentation is not equivalent to object tracking. Roughly speaking, object tracking is to track the segmented objects over an image sequence, although the extension of the rigidity constraint to multiple frames is nontrivial. Motion segmentation aims at the motion layers of a scene rather than the moving objects. For example, if a moving object contains multiple motions at a moment, it may be divided into several motion layers. When these motion layers share the same motion, they could be merged into a single layer. Hence, motion segmentation usually uses the information from a few successive frames. In contrast, object tracking focuses on a moving object in a scene. It utilizes the information from an image sequence. Motion segmentation plays a role of fundamental module in motion analysis and tracking. [1] presented a subspace segmentation method to estimate the motion models of the motion layers based on two successive frames. Built on this subspace segmentation method, this paper will further aim at two other basic problems of motion segmentation, *i.e.* the detection of motion layer boundaries and depth ordering based on two successive frames. The basic idea is to refine a global segmentation to solve these two problems. We first address this subspace segmentation approach for motion model estimation. We then incorporate it with the intensity edge information into a post-processing procedure, which refines the layer boundaries and infers the layer order between two successive frames. These two procedures form a complete algorithm for motion segmentation. Our specific contributions in this paper include 1) the Polysegment algorithm (a special case of the generalized PCA [2]) is employed to detect the layer boundaries in our post-processing procedure, and 2) the cues from the intensity edges of images are utilized in the detection of the layer boundaries and depth ordering.

Previous Works

Although motion segmentation has long been an active

area of research, many issues remain open in computer vision, such as the layered motion descriptions [3,4], occlusion detection and depth ordering [5-7], and estimation of multiple motion models [8,9].

Most popular approaches to motion segmentation revolve around parsing the optical flow field in an image sequence. Because of the well-known aperture problem, the motion vector from optical flow computation can only be determined in the direction of the local intensity gradient. For the sake of completeness of optical flow field, it is assumed that the motion is locally smooth. Obviously, depth discontinuities and multiple independently moving objects usually result in discontinuities of the optical flow. The usual approaches are to parameterize the optical flow field and fit a different model (e.g. 2D affine model) to each moving object, such as the layered representation of the motion field [3]. The challenges of the optical flow-based techniques involve identifying motion layers (or pixel grouping), detecting layer boundaries, and depth ordering. Previous research can mostly be grouped into two categories. The first category is to determine all of the motion models simultaneously. This can be achieved by parameterising the motions and segmentation, and using sophisticated statistical techniques to predict the most probable solution. For example, Smith *et al*. in [6] presented a layered motion segmentation approach under a Bayesian framework by tracking edges between frames. In the implementation of their proposed scheme, the region edge labels were not directly applied to the Bayesian model. They were implicitly determined by the foreground-background orders of the motion layers and the motion layer labels for each region. Kumar *et al*. in [10] presented the learning approach of a generative layered representation of a scene for motion segmentation. In order to get the initial estimates of model, they utilized the loopy belief propagation, and further refined the initial estimate by using αβ-swap and α-expansion algorithms. The large number of undetermined parameters in their Bayesian models leads to the difficult tracking problem in a high dimensional parameter space. The second category is the dominant motion approach [11-13]. A single motion is first fitted to all pixels, and then to test for pixels that agree with that motion. This process can be repeated recursively on the outlier pixels to provide a full set of layers [12]. The central problem faced by this kind of approaches is that it is extremely difficult to determine the occluded edges of the moving regions (or motion layers). Furthermore, this problem can result in the failure of depth ordering of motion layers. However, analytically reasoning such complex cases is impractical. The main reasons are three fold. First, the smoothing required by the optical flow algorithms makes it difficult to localize the layer boundaries. Second, the optical flow field is usually parameterized by some 2D motion models (e.g. 2D affine), which is the first order approximation of the perspective model. It is unreliable to apply a 2D model to the boundaries of moving regions. Third, pixels in a neighborhood of the boundaries are in the areas of high intensity gradient. Slight errors or image noise can result in pixels of a very different intensity, even under the correct motion estimate [6]. In this paper, we will simplify the problem of motion segmentation based on an algebraic framework. We will first obtain a rough global segmentation and then refine it afterwards.

Our work is partially inspired by the subspace segmentation approach to motion model estimation proposed in [1]. This approach can provide a non-iterative and global estimation of motion layer segmentation. But it is incomplete, since the depth ordering and the detection of layer boundaries are ignored. In this paper we provide a complete solution by developing a novel post-processing procedure using the intensity structures of edges for the detection of (1) motion layer boundaries and (2) the layer order.

In the remainder of this paper, we first briefly review the subspace segmentation approach to motion model estimation [1] in section 2. In section 3, a post-processing procedure is presented for the detection of the layer boundaries and depth ordering. The experimental results and analysis are given in section 4. Our conclusion and future work are given in section 5.

2. Motion Segmentation by GPCA-PDA

The core of our proposed motion segmentation approach is the scheme of segmenting hyperplanes in R^K, which is called the generalized PCA (GPCA) in [2]. Applying the GPCA method to motion model estimation has been proposed in [1]. But the resulting motion model estimation can only yield coarse motion segmentation, *i.e.* the boundary of the motion layers is very blurry. Our basic idea is to further refine the boundary of the resulting motion layers by a post-processing procedure. Before introducing our post-processing procedure, we firstly review the motion model estimation approach in [1] briefly. The two used algorithms, GPCA-PDA Alg. and Polysegment Alg., can be found in [2]. (We also briefly introduce these two algorithms in Appendix.)

The first problem to motion segmentation is to obtain the layered motion models corresponding to independently moving regions in a scene, (*i.e.* layer segmentation). We address an algebra approach in terms of a known optical flow field which has been presented in [1]. Its distinct advantage over the other approaches is that it can determine all motion layers simultaneously.

Given N measurements of the optical flow

$\{(u,v)_i\}_{i=1}^{N}$ at the N pixels $\{(x_1,x_2)_i\}_{i=1}^{N}$, we can describe them through a affine motion as follows,

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13} - u = 0 \\ a_{21}x_1 + a_{22}x_2 + a_{23} - v = 0 \end{cases}.$$

In terms of the hyperplane representation in the Appendix, the solution to the multiple independent affine models can be rephrased as follows. Let $x = (x_1, x_2, 1, u, v)^T \in R^5$ and hyperplane S_i be spanned by the basis of $b_1 = (a_{11}, a_{12}, a_{13}, a_{14}, 0)^T$ and $b_2 = (a_{21}, a_{22}, a_{23}, 0, a_{24})^T$. We need to segment a mixture of the hyperplanes of dimension $d = 3$ in R^5, which is expressed as,

$$S_i = \left\{ x \in R^5 : (b_1, b_2)_i^T \, x = 0 \right\}.$$

The original equations of optical flow have finished the projection from $x \in R^5$ to two individual subspaces of R^4 in a natural way, *i.e.* each new hyperplane in R^4 can be expressed as,

$$(a_{11}, a_{12}, a_{13}, a_{14}) \cdot (x_1, x_2, x_3, x_4) = 0 .$$

Applying the scheme of Equations (A1)-(A4) in Appendix can yield the desired basis $B^{(i)} = (b_1, b_2)_i$ for each hyperplane S_i in R^3.

Up to now, one can obtain the initial estimation of all of the motion layers simultaneously. This is insufficient for motion segmentation, since we also need to determine the layer boundaries and the occlusion relationship. Beside that, it can be observed that each segmented layer contains some small and isolated spurious regions, and the resulting layer boundaries wander around the real ones. This makes the detection of the layer boundaries difficult. The occluded regions take place in the neighborhood of the layer boundaries. If the occluding edges can be determined correctly, the occluded regions can be segmented correctly. Furthermore, the resulting motion layers can also be linked to the occluded regions in terms of the occluding edges for the depth ordering. Hence, it is a crucial step to determine the occluding edges. Our development is based on the following observations (1) the intensity edges include the boundaries of motion layers; (2) the layer boundaries are not always the occluding edges; (3) determining the occluding edges and inferring the occlusion relationship can be fulfilled by testing the neighborhood of edges.

We will introduce the intensity edges of images into the potential occlusion areas for the detection of occluding edges in the next section.

3. Post-Processing Procedure

Let us consider a single viewpoint. The central problem is to detect the occluding edges, because the erroneous edge labeling can cause incorrect depth ordering. Most of the techniques considered so far employed only the motion field information for motion segmentation. For each frame, all edges, including edges of motion layers and textured edges of objects, are presented in the image intensity structure, which can provide the wealth of additional information to motion estimation. Due to their extreme length, a number of measurements might be taken along (or around) them. This leads to a more accurate estimation of motion.

Recent applications have motivated a renewal of motion segmentation by tracking edges [6,14]. Ogale *et al.* [7] classified the occlusions into three classes. In order to deduce the ordinal depth, they had to fill the occluded regions. This is to implicitly approximate the occluding edges by filling the neighborhood of the layer boundaries. [6] provides three fundamental assumptions of the relationship between regions and edges to identify the edges of moving regions. We add an extra assumption (*i.e.* the 4th below) and emphasize these four assumptions as follows.

1) As an object moves all of the edges associated with that object move, with a motion, which may be approximately described by some motion model.

2) The motions are layered, *i.e.* one motion takes place completely in front of another, and the layers are strictly ordered. Typically the layer farthest from the camera is referred to as the background, with nearer foreground layers in front of this.

3)An arbitrary segmented image region only belongs to one motion model, and hence any occluding boundary is visible as a region edge in the image.

4) For each frame, the intensity edges involve the edges of motion layers.

An important conclusion from these four assumptions is that the layer ordering can be uniquely determined if the layer of each moving region is known and the occluding edges are known. [7] presented the relationship of motion layers and occluded regions, and further emphasized that the motion layer involving the occluded region must be behind another one. Even when the layers of motion regions are known, ambiguities may still be presented in the layer boundary labeling, as shown in **Figure 1**. In **Figure 1(a)**, due to the occluded region *C*, we can infer the occlusion relationship between the motion regions *A* and *B*, while, in **Figure 1(b)**, we cannot find out the layer order according to the distinct edges of the motion layers. The layer boundaries are not the same as the occluding edges. The layer boundaries involve the occluding edges, but the layer boundaries are not always the occluding edges. It is infeasible to infer the layer order only by the layer boundaries. We can therefore con-

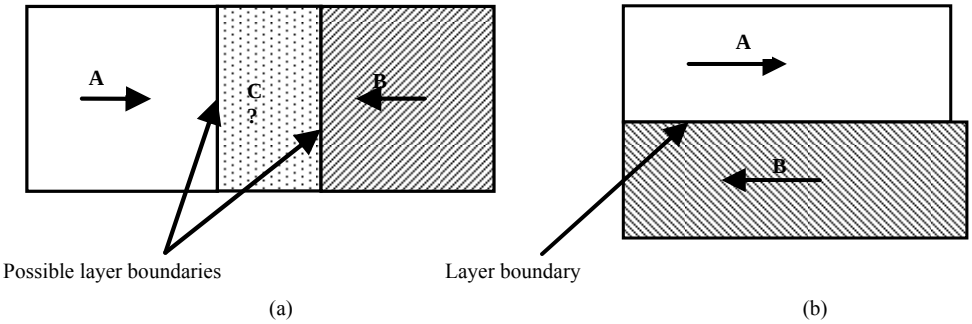

Figure 1. Illustration of moving regions (A,B) and occluded region (C). (a) The probable layer boundaries are determined by extending the moving region to the occluded region; (b) there is no occlusion region between layers A and B.

clude that the occlusion relationship hides behind the occluded regions, and identifying the occluding edges can reveal the occlusion relationship. The optical flow computation can usually identify the coarse occlusion regions as a by-product [15], which will be adopted in this paper.

The subspace segmentation approach described in section 2 is carried out on a given optical flow field instead of the image intensities. Due to the errors from the optical field (e.g. aperture problem etc.), each resulting motion layer contains two kinds of artifacts: (1) small isolated regions with texture and (2) dark holes over the image plane. It can be observed that a single hole in the middle of a foreground layer runs through to the background layer. Similar problems also exist in the occluded regions. Moreover, the resulting boundaries of motion layers and their neighborhood are, in general, highly unreliable areas. Therefore, the segmentation by the subspace segmentation method and the occluded regions detected by the optical flow computation cannot offer a valid solution to the above two problems.

Consider the neighborhood of the layer boundaries. It can be observed that the occluded regions are involved in the neighborhood of the layer boundaries as shown in **Figure 1(a)**. The edges' neighborhood contains the wealthy intensity structures of image. This can provide us sufficient cues to find the layer boundaries and occluding edges. We rephrase the problem of layer edge detection and depth ordering, and present our postprocessing procedure as follows.

The motion models of the layers are determined by the subspace segmentation approach described in section 2, while the layer boundaries and the layers of the occluded regions are undetermined. The problem we face here is how to determine the layer boundaries and infer the occlusion relationships. In order to do that, we will consider the intensity structures of each frame, the relevant occlusion region map (obtained by [15]) and the relevant boundary map of the initial motion layers (obtained by the subspace segmentation approach). Let us denote intensity edge map as M_I, occlusion region map as M_O and layer edge map as M_L thereafter. The motion of intensity edges dominates that of their neighborhood. It is straightforward to utilize the intensity structures of the neighborhood of the edges for detecting the layer boundaries and inferring the occlusion relationship. The proposed post-processing procedure given below is performed over two successive frames, but evidence could be accumulated over an image sequence for a more robust segmentation.

3.1. Construct Pending Areas

For each frame, we first determine some pending areas, which should involve all potential layer boundaries. Then, the detection of layer boundaries is carried out on the resulting pending areas accordingly. To this end, we place a set of windows w of size $n \times n.$ along the edges of M_L These small windows might be overlapped to each other. Usually each window w_i is determined by the M_O and M_L without a fixed size, *i.e.* it is expected to be so large that the resulting set of windows can cover the occlusion regions M_O and layer edge map M_L on the current frame. In our experiments, the minimal size n of w_i is set to 10 pixels.

3.2. Match Scores

Consider the resulting pending areas $W = \bigcup_i w_i$, which contains many intensity edges $l \subset M_I$. The potential layer boundaries are involved in M_I. in terms of the assumption (4). Thus, for each window w_i, we can compute the profile of every point p, which is defined as a vector $pf(p)$ by sampling the intensity derivative in the positive and negative directions of the intensity gradient at p. This is illustrated in **Figure 2**. The point profile is then normalized as,

$$pf(p) = \frac{pf(p)}{\sum_j |pf_j(p)|}. \quad (1)$$

According to the optical flow field, one can get a pair of corresponding points p and p' respectively on two

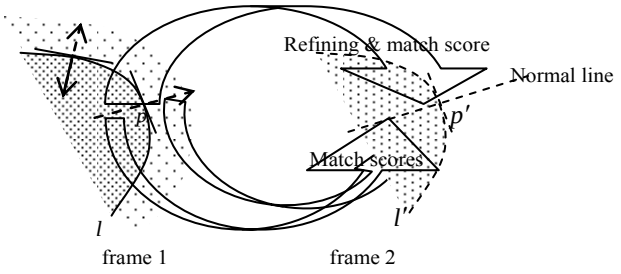

Figure 2. Illustration of point profiles and refining the matching point. Refining procedure is carried out as 1D search along the normal line on the frame 2.

successive frames. The match score is taken as the residual error of their profiles as follows,

$$e_0(p) = Exp\left\{-\frac{\left\|pf^{(i)}(p) - pf^{(i+1)}(p')\right\|^2}{\sigma^2}\right\},$$

where σ is a reference distance and is determined empirically. When the point p is far away from layer boundaries, $e_0(p)$ should approach one, *i.e.* the neighborhood of *p* obeys a single motion. Otherwise its neighborhood contains multiple motions. Furthermore, we can obtain other two match scores respectively along either side profile of the current point p, denoted as $e_1(p), e_2(p)$. If point *p* belongs to a layer boundary, one of these two scores should approach one while the other should approach zero, otherwise both of them should approach one.

3.3. Matching

Because of the aperture problem, the motion of edges can only be determined in the direction normal to the edge. This means that the corresponding point p' of the next frame ($i+1$) lies on the normal line, which is the normal at point *p* on the current frame (*i*). This is useful as it restricts matching isointensity contour on the frame ($i+1$) along the edge normal. In order to enhance the intensity edge matching, we add a new match score that is the residual error $e_0^{\perp}(p)$ of the profiles of p and p' along the edge tangent line, which is shown in **Figure 2**. Refining the matching point p' on the next frame ($i+1$) is thus implemented as a 1D search based on the match score of $\left(e_0(p) + e_0^{\perp}(p)\right)$ along the direction of point *p*'s gradient (*i.e.* the normal line) instead of point p''s gradient, which is also illustrated in **Figure 2**. After that, one can re-compute the match scores e_0, e_1, e_2 of the points p and p' in terms of their individual intensity gradients rather than the normal line.

3.4. Segmentation by Polysegment Alg.

Based on the match scores e_0, e_1, e_2 in the pending areas W, we apply the Polysegment algorithm as described in the Appendix respectively to the match scores of e_0, e_1, e_2 for the layer edge detection. There are two groups here, one is the group of layer edge points and the other is that of non-layer edge points. For each match score, we can thus get two cluster centers $\mu^{(i)} = \left(\mu_1^{(i)}, \mu_2^{(i)}\right), i = 0,1,2$. Moreover, to the points $p \in W$, there are eight cluster centers. The layer edge points should cluster around the two centers of

$$cent_1 = \left(\min \mu^{(0)}, \min \mu^{(1)}, \max \mu^{(2)}\right)$$

and

$$cent_2 = \left(\min \mu^{(0)}, \max \mu^{(1)}, \min \mu^{(2)}\right).$$

The segmentation of W is obtained as follows,

$$i = \arg \min_{j=1,\cdots,8} \left\|e(p) - cent_j\right\|^2, \quad (2)$$

where $e(p) = (e_0, e_1, e_2)(p)$. On this basis, the points of W can be classified into two groups, layer edge points and non-layer edge points.

3.5. Region Merging

For the group of non-layer edge points, one can merge most of small spurious regions in a big motion layer, *i.e.* merging small regions with a motion layer by comparing their areas with their individual neighbors'. This can lead to the connected layer. But it can be observed that the detected layer boundaries usually have discontinuities with the group of layer edge points, *i.e.* a set of layer edge segments. This is due to the fact that some layer edge points are incorrectly classified into the group of non-layer edge points. Based on the areas of the segmented layer regions, it is impossible to make a correct decision of region merging when these small regions may contain layer edge segments. This is because the layer edge segments indicate that the both sides should respectively occupy different motion layers and could not be merged into a single layer at anytime. These regions are thus left as the undetermined regions temporarily.

On the other hand, a connected layer has a continuous boundary M_L. These layer edge segments only prune the region of the layer, but do not form new closed regions within the layer. In our experiments, we simply replace some parts of M_L with the new layer edge segments according to the nearest neighbor criterion. Then, the area comparison strategy is employed to those undetermined regions nearby the layer boundary for region merging. A layer edge segment separates one region into two motion layers. When merging two or more undetermined regions which share a layer edge segment,

the merging procedure should be terminated.

3.6. Depth Ordering

After region merging, one can obtain the desired boundaries of motion layers M_L. If the occluded regions belong to a motion layer, this layer must be behind another one. Our problem can now be rephrased as HOW to assign the occluded regions to the known motion layers.

With the occlusion region map M_O, one can first determine which points of the layer boundaries belong to the occluding edges, since the layer boundaries involve the occluding edges. The worst case is that the points of the occluding edges are not within M_O. But both side profiles of these points should overlap with M_O at that moment. On this basis, one can determine the points of the occluding edges by checking if they are within M_O or their profiles overlap with M_O. Since some points of layer boundaries may not belong to the occluding edges, such as in **Figure 1(b)**, the depth ordering can only carry out on the detected occluding edges. Then, for the points of the occluding edges, one can extend their profiles in the direction of the intensity gradient to the known motion layers for their profile labeling, *i.e.* inferring which layers both sides of the occluding edges respectively belong to.

Furthermore, inferring the occlusion relationship can be fulfilled by comparing the match scores $e_1(p), e_2(p)$ of each point p of the occluding edges. This is because an occluded region only shares the same motion layer with one of the profiles of an occluding edge. The smaller match score corresponds to the real occlusion region. This implies that one side of an occluding edge with the smaller match score is behind the other side, since it involves the occlusion region. In terms of the profile labeling of the occluding edge points, we can therefore find ordinal depth.

The **Post-processing procedure** is summarized as follows:

1) Extracting the pending areas W on each frame;

2) Refining the corresponding points p' on the next frame ($i+1$), and then re-compute the match scores $e_0(p), e_1(p), e_2(p)$;

3) Applying the polysegment algorithm to the match scores of *W* for detecting the points of layer boundaries;

4) Merging the spurious regions for the continuous boundaries of the motion layers;

5) Determining the occluding edges in terms of M_O;

6) Extending the profiles of the occluding edge points to the known motion layers for the profile labeling;

7) Comparing the match scores $e_1(p), e_2(p)$ of the occluding edge points p for depth ordering, *i.e.* $\min\{e_1(p), e_2(p)\}$ corresponds to the occluded region.

This post-processing procedure and the subspace segmentation approach described in section 2 constitute a complete algorithm of motion segmentation. Note that in our algorithm, the estimation of all the motion models in a scene is undertaken at the first procedure (*i.e.* subspace segmentation method), and the detection of the layer boundaries and depth ordering are carried out at the second procedure (*i.e.* post-processing procedure). This is different from the previous approaches. Usually the motion model estimation was mixed with the later processing. This makes the algorithms complicated and the implementation difficult.

4. Experiments and Analysis

Our algorithm was tested on several image sequences. In this section, the two results of the "flower garden" and "Susie calling" are presented. All programs have been implemented on the MatLab platform using a publicly available package—GPCA-PDA [16] and the optical flow code in [15]. All the image sequences used in our experiments are available at [17].

4.1. Flower Garden

In this experiment, we applied our motion segmentation approach to the flower garden sequence of resolution 175×120 pixels. The tree trunk in front of a garden is taken by a camera undergoing translation from the left to the right. Our goal is to determine the boundaries of the motion layers, and find the layer order over two successive frames. Our approach found out two motion layers, the tree trunk and garden background.

Figure 3 gives the segmentation of the affine model using the subspace segmentation approach described in section 2. It can be noted that the occlusion regions from the optical flow fields are crude, and contain many spurious small regions. The red arrows illustrate the possible occlusion regions in the successive frame 1 and 2. We show the results of motion segmentation of frame 1 in **Figure 3(1)-(3)**. Note that the occlusion regions in **Figure 3(2) and (3)**, which are not the layer boundaries, are the interim areas between the foreground and background. It is impossible to determine the depth ordering using the obtained motion layers before determining the layers of the occluded regions. In addition, the resulting layer boundaries are also unreliable. Similar to the occluded regions, there are many small and isolated spurious regions on the obtained layers. We need to refine the layer boundaries and find out the layer order.

Figure 4 gives the segmentation results of the subspace segmentation approach followed by the post-processing procedure described in section 3. The boundaries of motion layers can go across the occluded regions and converge to the desired locations. But we can also note that a patch of ground is classified as the fore-

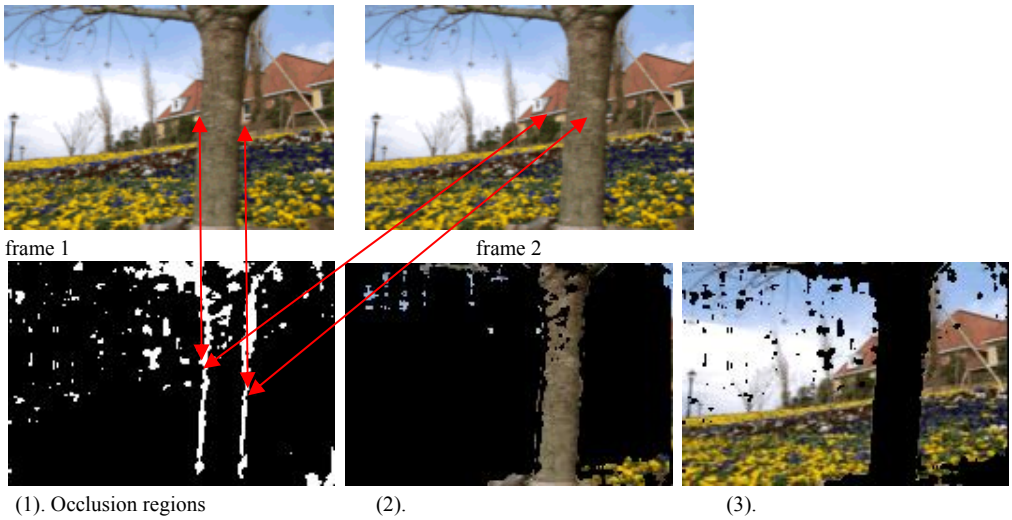

Figure 3. Motion segmentation results by the subspace segmentation approach only. (1) The occlusion regions from the optical flow field between frame 1 and 2; (2) and (3) are the segmentation results only using the subspace segmentation approach described in section 2.

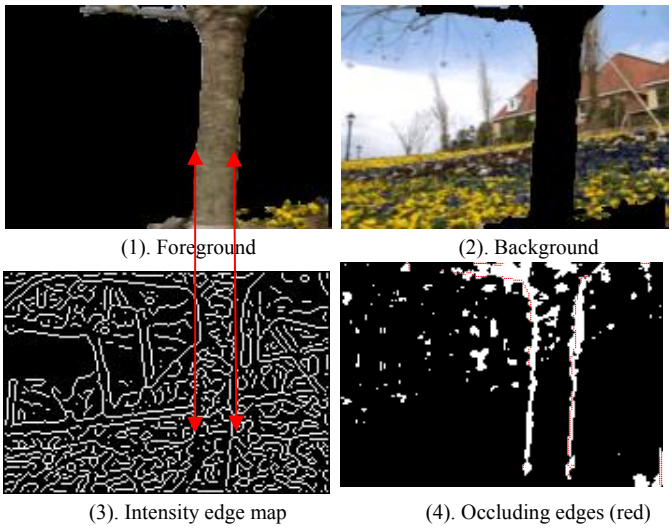

Figure 4. Refined and layered motion segmentation. (1) and (2) are the segmentation results; (3) the intensity edges of the image; (4) occluded region and occluding edges.

ground as shown in **Figure 4(1)**. This is due to the fact that the motion variance of this patch between the successive frames 1 and 2 is close to that of the "tree", away from the background. If going through over multiple frames, the motion of this patch should be distinguished from the "tree", since the motion of the ground is prone to be modeled by a single affine model. The intensity edge map of frame 1 is obtained by the Canny edge detector, and also shown in **Figure 4(3)**. It can be observed that the boundaries of motion layers are involved in the intensity edge map, e.g. the red arrows illustrate the corresponding edges between the layer boundaries and the intensity edges of the image. Moreover, we also show the occlusion edges (red) in **Figure 4(4)**. Partial occluding edges are not involved in the initial occlusion regions. But, the profiles of these edge points overlap with the occlusion regions. These points can thus be joined with the occluding edges. Additionally, it can be noted that the layer boundaries involve the occluding edges, but the layer boundaries are not always the occluding edges. Locating the occluding edges can help us find the depth ordering.

4.2. Susie Calling

This sequence presents a hand holding a phone while the head is rising slightly. The image resolution is 170×120

Figure 5. The segmentation of the Susie calling. (1)-(3) are the results of the subspace segmentation method; (4)-(6) are the results of the post-processing procedure; (7) the layer boundary goes through the occluded regions.

pixels. It can be observed that the region of the phone is enwrapped by the head region. Our segmentation approach aims at separating the phone region from the head region. The segmentation results are shown in **Figure 5**. The region of phone is in front of the regions of head and background. The background region is behind the head region.

Due to the rich texture of the hair, the segmented head region contains many small holes, particularly in the hair area. It is difficult to determine the boundaries of the hair. For example, in **Figure 5(2)**, a patch of hair image is incorrectly classified into the group of the phone region. The post-processing could not merge this patch into the head region either, as shown in **Figure 5(4)-(6)**. This is because the detected layer edge segments goes through the occluded regions as shown in **Figure 5(7)**. Moreover, it has been judged that the regions of the phone and the hair patch are in front of the head region. This seems to be a bit strange. In general, the hair patch belongs to the head region. All the hair should be regarded as a whole body on the head and there is no occlusion to each other (unless the hairlines are considered). But it can be observed that the occluded regions overlying on the layer boundary appear at the bottom right of the image, *i.e.* around the boundaries between the shoulder and the hair. The motion of the hair is independent of that of the shoulder. The shoulder region is classified into the head region. Thus, it is acceptable to preserve this patch as an independent layer as shown in **Figure 5(5)**.

5. Conclusions

In this paper, we proposed a novel approach for motion segmentation based on the subspace segmentation techniques. The novelty is that by incorporating the intensity structures of images, our proposed approach can effectively detect the motion layer boundaries and the depth ordering. Different from the previous motion segmentation approaches, our approach provides a non-iterative and global solution to motion segmentation under a unified algebra framework, *i.e.* the generalized PCA [2,18].

However, it can be noted that our algorithm relies on a given optical flow field. In our experiments, many available optical flow algorithms do not seem suitable for the scenarios with a salient rotation element. This will restrict the applications of our algorithm. It is crucial to further develop a robust optical flow algorithm. Our future work will aim to tackle this challenge.

REFERENCES

[1] R. Vidal and Y. Ma, "A Unified Algebraic Approach to 2-D and 3-D Motion Segmentation," *Journal of Mathematical Imaging and Vision*, Vol. 25, No. 3, 2006, pp. 403-421. doi:10.1007/s10851-006-8286-z

[2] R. Vidal, Y. Ma and S. Sastry, "Generalized Principal Component Analysis (GPCA)," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 12, 2005, pp. 1-15. doi:10.1109/TPAMI.2005.244

[3] J. Y. A. Wang and E. H. Adelson, "Layered Representation for Motion Analysis," *IEEE Conference on Computer Vision and Pattern Recognition*, New York, 15-17 June 1993, pp. 361-366. doi:10.1109/CVPR.1993.341105

[4] R. Szeliski, S. Avidan and P. Anandan, "Layer Extraction from Multiple Images Containing Reflections and Trans-

parency," *IEEE Conference on Computer Vision and Pattern Recognition*, Hilton Head, 13-15 June 2000, pp. 246-253.

[5] M. J. Black and D. J. Fleet, "Probabilistic Detection and Tracking of Motion Boundaries," *International Journal of Computer Vision*, Vol. 38, No. 3, 2000, pp. 231-245. doi:10.1023/A:1008195307933

[6] P. Smith, T. Drummond and R. Cipolla, "Layered Motion Segmentation and Depth Ordering by Tracking Edges," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 4, 2004, pp. 479-494. doi:10.1109/TPAMI.2004.1265863

[7] A. S. Ogale, C. Fermuller and Y. Aloimonos, "Motion Segmentation Using Occlusions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 6, 2005, pp. 988-992. doi:10.1109/TPAMI.2005.123

[8] D. J. Fleet, M. J. Black, Y. Yacoob and A. D. Jepson, "Design and Use of Linear Models for Image Motion Analysis," *International Journal Computer Vision*, Vol. 36, No. 3, 2000, pp. 171-193. doi:10.1023/A:1008156202475

[9] W. Yu, G. Sommer and K. Daniilidis, "Multiple Motion Analysis: In Spatial or in Spectral Domain," *Computer Vision and Image Understanding*, Vol. 90, No. 2, 2003, pp. 129-152. doi:10.1016/S1077-3142(03)00011-0

[10] M. P. Kumar, P. H. S. Torr and A. Zisserma, "Learning Layered Motion Segmentation of Video," *Proceedings of the* 10*th IEEE International Conference on Computer Vision*, Beijing, 17-20 October 2005, pp. 33-40. doi:10.1109/ICCV.2005.138

[11] M. Irani, P. Anandan, J. Bergen, R. Kumar and S. Hsu, "Efficient Representations of Video Sequences and Their Representations," *Signal Processing*: *Image Communication*, Vol. 8, No. 4, 1996, pp. 327-351. doi:10.1016/0923-5965(95)00055-0

[12] M. Irani, B. Rousso and S. Peleg, "Computing Occluding and Transparent Motions," *International Journal of Computer Vision*, Vol. 2, No. 1, 1994, pp. 5-16. doi:10.1007/BF01420982

[13] G. Csurka and P. Bouthemy, "Direct Identification of Moving Objects and Background from 2D Motion Models," *Proceedings of IEEE International Conference of Computer Vision*, Kerkyra, 20-27 September 1999, pp. 566-571. doi:10.1109/ICCV.1999.791274

[14] T. Papadimitriou and K. I. Diamantaras, *et al*., "Video Scene Segmentation Using Spatial Contours and 3D Robust Motion Estimation," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 14, No. 4, 2004, pp. 485-497. doi:10.1109/TCSVT.2004.825562

[15] A. S. Ogale and Y. Aloimonos, "A Roadmap to the Integration of Early Visual Modules," *International Journal of Computer Vision*, Vol. 72, No. 1, 2007, pp. 9-25. doi:10.1007/s11263-006-8890-9

[16] Generalized Principal Components Analysis matlab codes available at http://perception.csl.uiuc.edu/gpca/

[17] Video sequences available at http://www.cipr.rpi.edu/resource/sequences/

[18] R. Vidal, "Generalized Principal Component Analysis (GPCA): An Algebraic Geometric Approach to Subspace Clustering and Motion Segmentation," Ph.D. Thesis, Electrical Engineering and Computer Sciences, University of California at Berkeley, 2003.

Appendix

Segmenting Hyperplanes of Dimension K–1 in R^K

Given a set of points $X = \left\{ x^{(j)} \in R^K \right\}_{j=1}^{N}$ in a homogeneous coordinate system, and linear hyperplanes $\left\{ S_i \subset R^K \right\}_{i=1}^{n}$ of dimension $k_i = \dim(S_i) = K-1$, we need to identify S_i. Usually the subspace is given as, $b^{(i)} \cdot x = 0, b^{(i)} \in R^K$. Then, this hyperplane can be represented as,

$$S_i = \left\{ x \in R^K : x^T b^{(i)} = 0 \right\}.$$

Furthermore, an arbitrary point x lies on one of the hyperplanes if and only if,

$$\left(x \in S_1\right) \bigcup \cdots \bigcup \left(x \in S_n\right) \equiv \bigcup_{i=1}^{n} x^T b^{(i)} = 0$$

$$\equiv p_n(x) = \prod_{i=1}^{n} x^T b^{(i)} = 0$$

$$= y^T c_n = 0,$$

where, $y = \left(x_1^n x_2^0 \cdots x_K^0, x_1^{n-1} x_2^1 \cdots x_K^0, \cdots, x_1^0 \cdots x_{K-1}^0 x_K^n \right)^T \in R^m$ and $c_n \in R^m$ is a coefficient vector consisting of a set of monomials of $\{b^{(i)}\}$ and $m(n,K) = \frac{(n+K-1)!}{(K-1)!n!}$. For a given point set X, we have a linear system on c_n as follows,

$$L_n c_n = \begin{pmatrix} y(x^{(1)})^T \\ \cdots \\ y(x^{(N)})^T \end{pmatrix} c_n = 0 \in R^N \qquad \text{(A1)}$$

where, $L_n \in R^{N \times m}$. When the number of hyperplanes n is known, c_n can be obtained from the null space of L_n. In practice, n is always determined in terms of L_n. For a unique solution of the coefficient vector c_n, it is expected that $rank(L_n) = m(n,K) - 1$, which is a func-

tion of variable *n*. In the presence of noise, let $rank(L_i) = r$ when $n = i$ and $\lambda_{r+1} \Big/ \sum_{j=1}^{r} \lambda_j < \varepsilon$, where L_i is the data matrix with $rank = r$, λ_j is the *j*th singular value of L_i and ε is a given threshold.

For any x, we have $p_n(x) = c_n^T y(x)$. Each normal vector $b^{(i)}$ can be obtained from the derivatives of p_n. Consider the derivative of $p_n(x)$ as follows

$$\nabla p_n(x) = \frac{\partial p_n(x)}{\partial x} = \frac{\partial}{\partial x} \prod_{i}^{n} x^T b^{(i)} = \sum_{i}^{n} b^{(i)} \prod_{j \neq i} x^T b^{(j)} .$$

For a point $x^{(l)} \in S_l$, $\prod_{j \neq i} b^{(j)T} x^{(l)} = 0$ for $l \neq i$. It can be noted that there is only one non-zero term in $\nabla p_n(x)$, *i.e.* $\nabla p_n(x^{(l)}) = b^{(i)} \prod_{j \neq i} b^{(j)T} x^{(l)} \neq 0$ for $l = i$. Then, the normal vector of S_i is yielded as,

$$b^{(i)} = \frac{\nabla p_n(x^{(l)})}{\left\| \nabla p_n(x^{(l)}) \right\|} \tag{A2}$$

In order to get a set of points lying on each hyperplane respectively, so as to determine the corresponding normal vectors $b^{(i)}$, we can choose a point in the given *X* close to one of the hyperplanes as follows,

$$i = \arg \min_{\nabla p_n(x) \neq 0} \frac{|p_n(x)|}{\left\| \nabla p_n(x) \right\|}, \text{ where } x \in X \tag{A3}$$

After given the normal vectors $\{b^{(i)}\}$, we can classify the whole point set *X* into *n* hyperplanes in R^K as follows,

$$label = \arg \min_{j} \left(x^T b^{(j)} \right), j = 1 \cdots n \tag{A4}$$

This algorithm is called GPCA-PDA Alg. in [2,18].

Polynomial Segmentation Algorithm

Consider a special case of piecewise constant data. Given *N* data points $x \in R$, we hope to segment them into an unknown number of groups *n*. This implies that there exist *n* unknown cluster centers $\mu_1 \neq \cdots \neq \mu_n$, so that,

$$(x = \mu_1) \cup \cdots \cup (x = \mu_n),$$

which can be described in a polynomial form as follows,

$$p_n(x) = \prod_{i=1}^{n} (x - \mu_i) = \sum_{k=0}^{n} c_k x^k = 0 \tag{A5}$$

To *N* data points, we have,

$$L_n c \equiv \begin{pmatrix} 1 & x_1 & \cdots & x_1^n \\ \vdots & & & \vdots \\ 1 & x_N & \cdots & x_N^n \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_{n-1} \\ 1 \end{pmatrix} = 0 \tag{A6}$$

where $L_n \in R^{N \times (n+1)}$ and $c \in R^{n+1}$. Usually, the group number is estimated as,

$$n = \min \{ i : \lambda_{i+1} / \lambda_i < \varepsilon \} \tag{A7}$$

where λ_i is the *i*th singular value of L_i, which is the collection of the first $i+1$ columns of L_n, and ε is a given threshold that depends on the noise level.

After solving the coefficient vector *c* of Equation (A6), we can compute the *n* roots of $p_n(x)$, which correspond to the *n* cluster centers $\{\mu_i\}_{i=1}^{n}$. Finally, the segmentation of the date is obtained by,

$$i = \arg \min_{j=1,\cdots,n} (x - \mu_j)^2 \tag{A8}$$

The scheme of Equations (A5)-(A8) is called as the Polysegment algorithm in [18].

