Asian Conference on Computer Vision (ACCV)
January 2004, Jeju Island, Korea

PARAMETRIC 2D LAYER INFERENCE FROM UNCALIBRATED IMAGES

Eun-Young Kang, Isaac Cohen and Gérard Medioni
{elkang,icohen,medioni}@usc.edu

Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, CA 90089-0273, USA

ABSTRACT

Grouping based on motion and spatial support is essential, and is challenging. This paper focuses on extracting 2D layers from uncalibrated images. Targeted motion models are affine transformations. Our approach reliably extracts multiple 2D affine motion layers from noisy initial matches. This approach is based on: (1) a parametric method to detect and extract 2D affine; (2) the representation of the matching points in decoupled joint image spaces; (3) the characterization of the property associated with affine transformation in the defined spaces; (4) a process to extract multiple 2D motions simultaneously based on tensor-voting; (5) layer refinement based on a hybrid property: motion and color homogeneity. The robustness of the approach is demonstrated with several results.

1. INTRODUCTION

A layer specifies a 2D image region with time-varying information generated by camera motion and object motion. Each layer is associated with a motion descriptor and a compact form of image regions from the original images. Layers can efficiently represent non-rigid objects, transparent object's motion and shallow 3D surfaces. Also, each layer facilitates encoding additional information such as texture and blurring masks. Many researchers have popularized the concept and usage of the layered representation over a decade [1][2][5][9][10][12][17][20]. They have demonstrated that a layered representation plays a promising role between a 2D image and 3D structures. However, many methods are still based on assumptions, such as a pre-specified number of layers, and are sensitive to initial setting or noise.

There are two types of layers: 3D layers and 2D layers. In a 3D-layer based representation, each layer consists of a 3D plane equation, the texture of the plane, a depth ordering per pixel, and many more information. In A 2D-layer based representation, each layer is associated with a 2D transformation equation and its corresponding image texture. Extracting 3D layers is a non-trivial, computationally sensitive and excessive task for many applications, such as video coding and analysis. For these applications, 2D-layered representation is sufficient. This paper focuses on extracting 2D layers from uncalibrated images and presents a robust layer inference method.

Parametric approaches to extract motion layers use a specific motion model to define grouping constraint. Each pixel or regions are grouped into layers based on an object motion model. There are iterative parametric approaches [1][9][21]. The iterative approaches either start with a pre-specified number of layers to fit pixel motions, or perform repetitive extraction-and-removal of dominant motions. In other words, they are based on an assumption of a known number of layers or an existence of conspicuously dominant motion. The regularization-based method focuses on the boundary discontinuity between regions. This approach attempts to apply different smoothing factors and detect accurate layers based on unreliable discontinuity around motion boundaries extracted from locally measured correlation [10]. Approaches based on the use of Markov Random Fields (MRF) focus on handling discontinuities in the optical flow [4][7][8][13]. These methods give some good results, but they rely on a spatial segmentation result from the early stage of algorithm, which may not be practical in many cases.

The main issue associated with layer-based segmentation is the selection of the number of motion regions and the likelihood measure for clustering pixels in the presence of multiple motions. To cope with problems associated with multiple motions, several EM-based methods were proposed [2][3][5] [18][20][22]. EM-based methods have showed good results. However, EM-based methods also require specifying the number of regions and a significant amount of iteration to reach satisfactory performance. In case that EM-based methods start without a specified number of layers, they usually require exhaustive search, which might not guarantee the optimal grouping of regions.

As apposed to parametric approaches, non-parametric and non-iterative region segmentation methods based on tensor voting were proposed [6][15][17] proposed a motion segmentation method by using normalized cuts,

which is non-parametric. The grouping is based on the probability distribution of the image velocity and the distance between motion profiles at two pixels. Generally, grouping based on normalized graph cut method suffers from its performance as the image size grows. Non-parametric approaches solve general clustering problem. However, the applications requiring parameter recovery, such as compression, benefit from a motion segmentation approach that groups pixels into region of similar parametric motions.

There are several major computational issues in order to extract layers. These are: (1) the determination of the number of layers; (2) the representation of motion for each layer; and (3) the assignment of pixels to layers. Our approach reliably extracts multiple 2D affine motion layers from noisy initial matches without pre-specified number of layers. Discontinuity between layers is defined by motion difference computed from the motion parameters. Our approach is based on the representation of the matching points in decoupled joint image spaces and the characterization of the property associated with affine transformation in the defined spaces. In decoupled joint image spaces, affine transformations are identified as planes. Each pixel for a layer is clustered based on plane property. The process of our approach extracts multiple affine motions simultaneously based on tensor-voting. Tensor voting identifies and estimates locally salient multiple affine motions while removing outliers. This process is non-iterative. In addition, it does not require any prior assumptions, such as a pre-specified number of layers. The assumptions made are the neighborhood size for voting and the sub-pixel discontinuity for clustering. After dominant motion layer extraction, layers are refined by using color-region homogeneity.

2. AFFINE MOTION

Motion estimation using parametric models is widely used for video processing such as image mosaics, video compression and video surveillance. Among all 2D transformation, the affine motion model is a commonly used method for these applications due to its simplicity and the small inter-frame camera motion.

2.1. Affine model and joint image space

In this section, the relationship between affine motion and the decoupled joint image space is explained. 2D affine motion is defined by six parameters. And affine joint image space is defined by the 4D space (x, y, x',y') where (x,y) and (x',y') are an image correspondence. In this representation, each point is a combination of 2D image vectors, is represented by a vector q defined below. The affine transformation can be rewritten as:

$$(q^T 1)P^T P\begin{pmatrix} q \\ 1 \end{pmatrix} = 0$$

where

$$q = (x, y, x', y') \text{ and } P = \begin{pmatrix} a & b & -1 & 0 & t_x \\ c & d & 0 & -1 & t_y \end{pmatrix}^T$$

In the affine model, the two equations are independent, and therefore the joint space *(x, y, x',y')* can be decoupled into two joint spaces *(x,y,x')* and *(x,y,y')*. The decoupled joint image space allows to reduce the dimension of the space and to enforce the affine constraint directly to the tensor voting formalism. By defining $p_x = (a, b, -1, t_x)^T$ and $p_y = (c, d, -1, t_y)^T$, we have two separate joint spaces $q_x = (x, y, x')^T$ and $q_y = (x, y, y')^T$. We obtain the following equations in the decoupled joint image spaces: $p_x^T \begin{pmatrix} q_x \\ 1 \end{pmatrix} = 0$ and $p_y^T \begin{pmatrix} q_y \\ 1 \end{pmatrix} = 0$. These equations define two plane equations in decoupled joint image spaces. They are written as:

$$p_x^T \begin{pmatrix} q_x \\ 1 \end{pmatrix} = ax + by - x' + t_x = 0, p_y^T \begin{pmatrix} q_y \\ 1 \end{pmatrix} = cx + dy - y' + t_y = 0$$

Therefore, in decoupled affine joint image space, each point defined by q_x and q_y lies on a plane parameterized by p_x and p_y. Consequently, the correspondences constrained by an affine transformation are lying on a *plane*. Each plane has a normal direction *(a, b, -1)* or *(c, d, -1)*. The parameter t_x and t_y constrained the perpendicular distance between the origin of the space and the plane. This property gives the criteria to decide which pixel belongs to which plane. In other words, the embedded structure that represents the affine motion is a 2D plane in a 3D decoupled affine joint image space. This approach formulates the problem in a geometric space and minimizes geometric distance in a non-iterative manner. It differs from classical techniques, in that they attempt to minimize the algebraic errors iteratively.

If input correspondences consist of perfect correspondences, each plane can be parameterized by the normal and distance measure. When multiple affine motions are present in the image, the same number of corresponding 2D planes is defined in the embedded 3D space. Consequently, grouping points belonging to the same plane (same motion) naturally leads us to region segmentation and parameter estimation associated with it. In reality, the dense correspondence contains many mismatches. For this reason, we use tensor voting for achieving such selection.

2.2. Tensor voting for structure inference

Tensor voting formalism provides a robust approach for extracting salient structures, such as curve and surface [14]. The input data is encoded as second order symmetric tensors to capture first order differential geometry information and its singularities, and then each input (called *token*) propagates its information to the neighboring data by voting. The extraction of salient

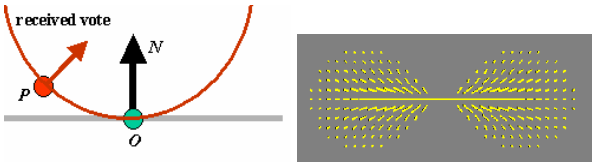
structures is inferred from the canonical description of an arbitrary tensor by its eigensystem representing the local geometric properties of the data.

In the 2D case, the salient structures to be interested in are points and curves. Each encoded tensor is an ellipse (2D second order symmetric tensor). An ellipse is decomposed into its 2x2 eigensystem, where eigenvectors $\hat{\boldsymbol{e}}_1$ and $\hat{\boldsymbol{e}}_2$ represent the ellipse orientation and corresponding eigenvalues λ_1 and λ_2 represent the ellipse size. To express it in a compact form, the tensor $\boldsymbol{S}$ is represented as $\boldsymbol{S} = \lambda_1\hat{\boldsymbol{e}}_1\hat{\boldsymbol{e}}_1^T+\lambda_2\hat{\boldsymbol{e}}_2\hat{\boldsymbol{e}}_2^T$, where $\lambda_1\geq\lambda_2\geq0$. If a token represent a curve element, $\hat{\boldsymbol{e}}_1$ represents the curve normal, while $\lambda_1=1$ and $\lambda_2=0$. If a token represents a point element, it has no distinctive orientation with $\lambda_1=0$ and $\lambda_2=0$. In a 3-D case, a salient curve element is represented by a thin ellipse, whose major axis represents the estimated tangent direction, and whose length reflects the saliency of the estimation. In a more compact form, $\boldsymbol{S} = \lambda_1\hat{\boldsymbol{e}}_1\hat{\boldsymbol{e}}_1^T+\lambda_2\hat{\boldsymbol{e}}_2\hat{\boldsymbol{e}}_2^T+\lambda_3\hat{\boldsymbol{e}}_3\hat{\boldsymbol{e}}_3^T$, where $\lambda_1\geq\lambda_2\geq\lambda_3\geq0$ are the eigenvalues, and $\hat{\boldsymbol{e}}_1$, $\hat{\boldsymbol{e}}_2$, $\hat{\boldsymbol{e}}_3$ are the eigenvectors corresponding to λ_1, λ_2, λ_3 respectively. The eigenvectors represent the principal directions of the ellipsoid and the eigenvalues encode the size and shape of the ellipsoid.

An efficient voting process allows to propagate each data to neighborhood. Each token casts a vote at each site in its neighborhood. The size and shape of the neighborhood and the vote strength and orientation are encapsulated in predefined voting fields for each type. There are two types of fields, for curve element and point element, in 2D. The fields are generated based on the scale factor σ (size of neighborhood). Vote orientation corresponds to the smooth local curve continuation from voter to votee. The vote strength $DF(\vec{d},\rho,\sigma)$ decays with curvature ρ and distance $\vec{d}$ between voter and votee as following.

$$DF(\vec{d},\rho,\sigma) = e^{-\left(\frac{|\vec{d}|^2+\rho^2}{\sigma^2}\right)}$$

Figure 1(a) shows how to generate voting field. A tensor O having curve normal N cast a vote at its neighbor P. The tensor P gets a stick vote from O, that smoothly interpolates normal N along the most appealing circle between O and P with decay factor $DF(\vec{d},\rho,\sigma)$. All neighborhood of O gets a stick vote with respect to the stick field as seen Figure 1(b) and sum them up using tensor addition. As a result of the voting procedure, we produce *arbitrary* second-order, symmetric tensors, therefore we need to handle any generic tensor. The spectrum theorem states that any tensor can be expressed as a *linear* combination of these three cases, i.e., $\boldsymbol{S}=(\lambda_1-\lambda_2)\hat{\boldsymbol{e}}_1\hat{\boldsymbol{e}}_1^T+(\lambda_2-\lambda_3)(\hat{\boldsymbol{e}}_1\hat{\boldsymbol{e}}_1^T+\hat{\boldsymbol{e}}_2\hat{\boldsymbol{e}}_2^T)$, where $\hat{\boldsymbol{e}}_1\hat{\boldsymbol{e}}_1^T$ describes a stick and $(\hat{\boldsymbol{e}}_1\hat{\boldsymbol{e}}_1^T+\hat{\boldsymbol{e}}_2\hat{\boldsymbol{e}}_2^T)$ describes a ball in the 3D case. At each location, the estimate of each of the two types of information, and their associated saliency, is therefore captured as follows. The point saliency has no orientation, and the saliency is given by λ_2. The curve saliency has a tangent orientation given by $\hat{\boldsymbol{e}}_1$, and saliency by $\lambda_2-\lambda_3$.



(a) Vote generation (b) Stick field

Figure 1 Voting in 2D

2.3. Tensor voting for affine motion

The tensor voting framework is general for extracting substantial structure. In our case, we showed that the structure of affine motion is a plane, and therefore we want to focus on extracting plane in a decoupled joint image space.

2.3.1. Tensor encoding for affine motion

When we encode initial correspondences into tensors, we do not have any prior knowledge about normal direction of planes and displacement from the origin of the decoupled joint image space. Therefore, we encode each initial input as an ellipsoid with equal eigenvalues for each eigenvector that propagates point information to all direction in any dimension. However, it is computationally expensive compared to starting with a stick tensor. In the case of dense correspondences, it is more desirable to avoid encoding points into ball tensor. To avoid this, we encode initial points as stick tensors, which a normal direction is defined based on the following assumptions:

$$a=1, b=0, T_x=(x'-x)c=0, d=1, T_y=(y'-y)$$

where *a, b, c, d, Tx* and *Ty* are the six affine parameters. This assumption means that initial affine motion considered as the identity. And, this assumption is valid because an affine motion between two consecutive video frames dost not include any dramatic affine-warp represented by parameters, *a,b,c* and *d*. If there appears a subtle warp-motion, the plane direction can be still inferred within voting range. Translation Tx and Ty are implicitly encoded in the defined space. Based on this assumption, each point is considered on the plane having the normal direction.

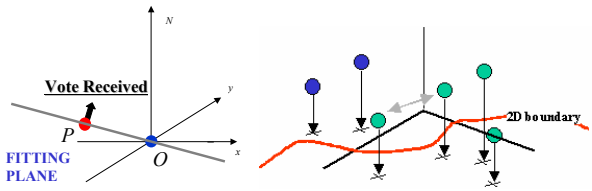
$$(a,b,-1)=(1,0,-1),(c,d,-1)=(0,1,-1)$$

2.3.2. Two phase tensor voting

Once we have all initial tensors, we perform a voting to find exact plane parameters. In a decoupled joint image space, which is a 3D space, each point is represented as 3D ellipsoid and the 3D stick voting field is generated based on 2D stick field shown in Figure 1(b). There are two phases in the voting process. During the first voting, each point collects votes cast by its neighbors and infers a

principal direction defining a plane. The normal orientation of 2D plane is defined by the eigenvector *e1* associated to the largest eigenvalue of the decomposed tensor. The saliency of the extracted plane is given by (λ_1 -λ_2) and characterizes the support of neighboring pixels to the characterized plane orientation. Therefore, isolated random noise has a small saliency due to the little support from its neighboring correspondences. During the second voting phase, voting is performed with a planar voting field, a thinner voting field, defined by the normal direction derived during the first step. At this phase, only highly salient points (defined by a threshold of the saliency values from the first voting) participate in the voting and cast their tensor properties to neighboring pixels.

Here, we enforce an additional property into the voting field in terms of shape and saliency propagation. The selected field propagates the planar geometry we are looking for in the processed data. The field considered is depicted in ??. We also illustrate its variation compared to the circle-based propagation mechanism used classically in tensor voting implementations.



(a) Plane-fitted voting field (b) Contour consideration

Figure 2 Tensor voting for affine motion

Figure 1(a) illustrates a general voting procedure given a token O and a normal direction N at that point. In affine case, a point P collects the vote cast by a point O with a normal direction N, based on the relative orientation of the plane fitted to points P and O, and the distance separating these points (Figure 2(a)). This plane-fitting enforces the affine metric during voting stage. In addition, our approach utilizes boundary information during first voting stage. The contour information is considered to inhibit voting between different color regions or across boundaries. The concept is depicted in Figure 2(b). In an image, the discontinuity appears along the image boundary. Therefore we use boundary information to maximize voting interaction within the same smooth regions. This is performed only for the first voting stage because the motion discontinuity might not coincide with color discontinuity.

The performance or the outcome of the result is dependent on sigma value to generate voting field. The sigma value is adaptively used. If the dense correspondences of the voting field are relatively small and if the correspondences are irregular and sparse then the voting field is relative large. In general, sigma value is set to cover at least 8x8 or 16x16 image blocks

3. LAYER INFERENCE

3.1. Initial motion layer extraction

The layer segmentation is performed by clustering points into regions from the most salient points characterized by the second voting step described previously. This is a seed-based region growing method. However, our approach does not require specifying the number of regions in advance, and the seeds are not selected randomly. The seeds are determined by the inferred saliency from voting process, and their saliencies retain the likelihood measures. The coherence is based on plane-smoothness in the decoupled joint image space and is measured by the angular difference of the normal directions (reflected by the coefficients *a, b, c, d*) and the distance from the space's origin (reflected by the parameters t_x and t_y). At each point, the normal orientation of the plane is encoded by the eigenvector e_1 associated to the largest eigenvalue. The saliency of the extracted plane is given by (λ_1 -λ_2) and characterizes the support of the neighbors to the characterized plane.

Given a seed point p_s and its plane description, we compare and cluster neighboring points, p_i, according to their likelihood. This likelihood measure is provided by the distance functions

$$d_s = e^s_{11}x_s + e^s_{12}y_s + e^s_{13}x'_s \quad d_i = e^i_{11}x_i + e^i_{12}y_i + e^i_{13}x'_i$$

where (x_s, y_s, x_s') and (x_i, y_i, x_i') are locations of the seed point, p_s and neighbor point, p_i, and each vector (e_{11}, e_{12}, e_{13}) represents the normal direction of the plane at that location. The likelihood measure is provided by the similarity: $\|d_s - d_i\|^2$. This function approximates the Euclidean distance between two parameterized motions. If the motion difference is smaller than 1 Euclidean distance in an image space, two points get clustered together. After one region is clustered, the clustering procedure iterates using the next highest salient points. This clustering step is performed in each decoupled joint image step. In order to fully estimate affine parameters and extract image region, the clustered sets from each joint image space should be merged.

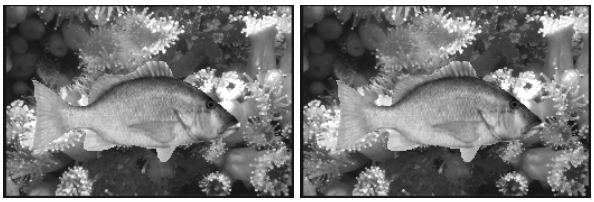
Layer clustering described in the previous section produces over-segmented regions. This is due to the fact that we enforce very strict threshold despite of the imperfect normal direction inferred from the voting method. We therefore merge similar motion layers based on the estimated affine parameters. The similarity between motion clusters is measured by the *symmetric transfer error*. If the overall error caused by two affine motions is less than 1 pixel (sub-pixel accuracy), the two regions are merged. Notice that this merging threshold value is same as the threshold used for plane clustering process in the voting space. However, we can combine some layers because the affine parameter in each region is estimated in least square manner and they compensate the

strict difference observed in the previous steps. Merging step is iterative and repeats until no more merging takes place. For each clustered layer, the affine parameters are estimated by analyzing the correlation matrix of the clustered points in each space. The parameters of the affine transform are therefore characterized by the eigenvector associated to the smallest eigenvalue of the correlation matrix [11].

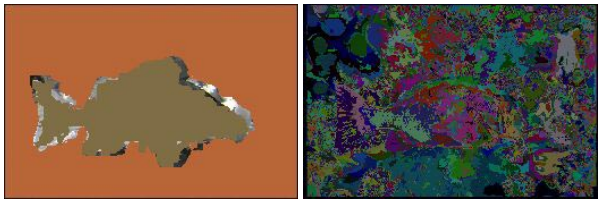
3.2. Layer refinement

Our method robustly extracts multiple affine motion layers, while rejecting outliers. After layer clustering, un-clustered points have to be processed. These points correspond to outliers or regions where the motion could not be estimated (uniform regions). These outliers create many holes in the image as shown in left image of Figure 3(b). Filling these holes using the estimated local affine motion does not provide satisfactory results. Indeed, robust motion estimates cannot be inferred in holes nearby motion boundaries. This prevents from clustering these regions based on the motion estimates. To address this problem, we make the following assumption: image pixels in a uniform color region belong to the same motion layer. This assumption is used for refining the extracted motion layers and it provides a complete labeling of image pixels. Our approach utilizes color homogeneity to refine motion layers. The used assumption is that each homogeneous color region belongs to one uniform motion. This color homogeneity property has been extensively used in many motion analysis [16] and stereo [19].

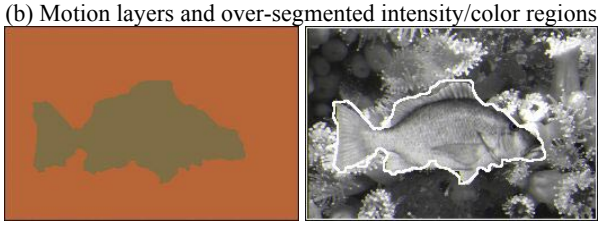
Our approach performs as follow: we compute a color based segmentation of the reference image as shown in right image of Figure 3(b) and propagate that segmentation to the next image using the computed motion layers and their corresponding affine model. We subsequently measure the Sum of Square Difference (SSD) in order to characterize the residual errors generated by mapping color properties of the pixels to its estimated location on the second image. For pixels that were not labeled by the motion layer segmentation, we consider all adjacent layers and estimate the corresponding SSD residual for the selected pixel. The pixel is then associated to the layer minimizing the SSD residual.



(a) Synthetic images, a foreground fish object moved 6 pixels to the right on the background



(b) Motion layers and over-segmented intensity/color regions



(c) Two recovered layers and their boundaries

Figure 3: Layer inference on a set of synthetic images

(a) Two video frames

(b) Refined motion layers and their boundaries

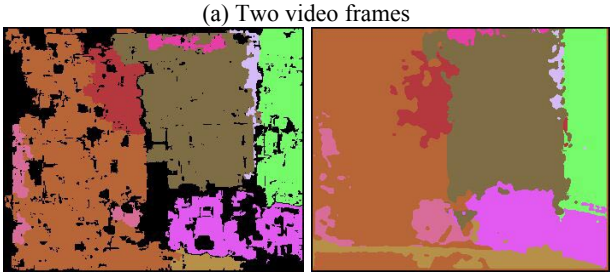
Figure 4 Layer inference

The demonstration of the performance of our approach is shown by several results. First of all, to show the accuracy of our approach, two synthetic images are tested. Synthetic images are generated with a known affine motion as shown in Figure 3. A foreground fish object is manually moved to 6 pixels to the right. Our approach extracted accurate affine motion groups and corresponding parameters based on tensor voting, then refined layer boundaries. The results show that affine parameters are correct and the boundary is very accurate. In Figure 4 and Figure 5, results for the multiple affine motions are demonstrated.

We analyzed the time complexity of our approach in sub-task. Major time consuming subtasks are initial matching, which is a multi-resolution Lukas and Kanade optical flow estimation or cross-correlation evaluation method, and tensor voting for a decoupled joint image spaces. The complexity of initial matching is O(Nsc), where N is the number of image pixel, s is the search window size and c is the convolution size. The complexity of tensor voting is O(Nm), where N is the

number of image pixel and *m* is the neighboring pixels within sigma values.

(a) Two video frames



(b) Initial motion layers and refined layers

Figure 5: Inferred layers in the presence of multiple moving objects

4. CONCLUSION

In this paper, we presented a robust method for extracting accurate motion layers based on velocity and color homogeneity. Substantial motion layers from noisy input correspondences are simultaneously extracted based on the representation of the correspondences in decoupled joint image spaces and tensor voting. The extracted layers are refined by the color homogeneity constraints. Each segmented homogeneous color region fits into one of the extracted motions that leads to the minimum image disparities. In the future, we plan to refine the motion boundaries around occlusion areas.

5. REFERENCES

[1] E. Adelson, Layered Representation for Image Coding, Technical Report 181, MIT Media Lab. 1991.
[2] S. Ayer and H. Sawhney, Layered Representation of Motion Video, ICCV, 195, pp. 777-784.
[3] S. Baker, R. Szeliski and P. Anandan, A Layered Approach to Stereo Reconstruction, CVPR, 1998.
[4] Y. Boykov, O. Veksler, R. Zabih, Markov Random Fields with Efficient Aproximations, CVPR, 1998, pp. 648-655.
[5] T. Darrell and A. Pentland, Cooperative Robust Estimation Using Layers of Support, M.I.T Media Vision and Modeling Group Tech Report, No 163, Feb., 1991.
[6] L, Gaucher and G. Medioni, Accurate Motion Flow Estimation with Discontinuities, ICCV, 1999.
[7] M. Gelgon and P. Bouthemy, A Region-Level Graph Labeling Approach to Motion-Based Segmentation, CVPR, 1997, pp. 514-519.
[8] F. Heitz and P. Bouthemy, Multimodal Estimation of Discontinuous Optical Flow Using Markov Random Fields, IEEE Trans. on PAMI, vol. 15, no. 12, pp. 1217-1232, 1993.
[9] M. Irani and S. Peleg, Motion Analysis for Image Enhancement: Resolution, Occlusion, and Transparency, Journal of Visual Communication and Image Representation, Vol. 4, No. 4, pp. 324-335, December 1993.
[10] S. X. Ju, M. J. Black and A. Jepson, Skin and Bones: Multi-layer, Locally Affine, Optical Flow and Regularizaton with Transparency, CVPR, 1996.
[11] E.Y. Kang, I. Cohen and G. Medioni, Robust Affine Motion Estimation in Joint Image Space using Tensor Voting, ICPR, 2002.
[12] Q. Ke and T. Kanade, A Robust Subspace Approach to Layer Extraction, IMVC, Orlando, Florida, Dec. 2002.
[13] C. Kervrann, F. Heitz, A Markov Random Field Model-Based Approach to Unsupervised Texture Segmentation Using Local and Global Spatial Statistics, IEEE Trans. on Image Processing, 4:6, pp. 856-862, 1995.
[14] G. Medioni, M.S. Lee, and C.K. Tang, A Computational Framework for Feature Extraction and Segmentation, Elsevier, 2000.
[15] M. Nicolescu and G. Medioni, Perceptual Grouping from Motion Cues Using Tensor Voting in 4-D, ECCV, 2000, vol. III, Copenhagen, Denmark, May 2002, pp. 423-437.
[16] M. Pardàs, P. Salembier, and B. González. Motion and region overlapping estimation for segmentation-based video coding. ICIP, 1994, pp. 428-432.
[17] J. Shi and J. Malik, Motion Segmentation and Tracking Using Normalized Cuts, ICCV, 1998.
[18] R. Szeliski, P. Anandan and S. Baker, From 2D Images to 2.5 Sprites: A Layered Approach to Modeling 3D Scenes, ICMCS, Florence, Italy, Vol. 1, June, 1999, pp. 44-50.
[19] H. Tao, H. S. Sawhney, and R. Kumar, A global matching framework for stereo computation, ICCV, 2001.
[20] P. Torr, R. Szeliski and P. Anandan, An Integrated Bayesian Approach to Layer Extraction from Image Sequences, IEEE Trans. on PAMI, 23(3), pp. 297-303, March 2001.
[21] J. A Wang and E. Adelson, Representing Moving Images with Layers, IEEE Transaction on Image Processing Special Issue: Image Sequence Compression, 3(5), pp.625-638, 1994.
[22] Y. Weiss, Smoothness in Layers: Motion segmentation using nonparametric mixture estimation, CVPR, 1997, pp.520-526.