IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2008, Anchorage, Alaska. ©IEEE.

High Resolution Motion Layer Decomposition using Dual-space Graph Cuts [*]

Thomas Schoenemann and Daniel Cremers
Department of Computer Science
University of Bonn, Germany

Abstract

We introduce a novel energy minimization method to decompose a video into a set of super-resolved moving layers. The proposed energy corresponds to the cost of coding the sequence. It consists of a data term and two terms imposing regularity of the geometry and the intensity of each layer.

In contrast to existing motion layer methods, we perform graph cut optimization in the (dual) layer space to determine which layer is visible at which video position. In particular, we show how arising higher-order terms can be accounted for by a generalization of alpha expansions. Moreover, our model accurately captures long-term temporal consistency. To the best of our knowledge, this is the first work which aims at modeling details of the image formation process (such as camera blur and downsampling) in the context of motion layer decomposition. The experimental results demonstrate that energy minimization leads to a reconstruction of a video in terms of a superposition of multiple high-resolution motion layers.

1. Introduction

Related Work. The decomposition of videos into a superposition of independently moving layers is of central importance to scene interpretation, video coding and movie compression. The idea dates back to the seminal paper of Wang and Adelson [15], who used a two-step optimization of two different functionals to alternatingly estimate motion fields and layers. Subsequent methods [2, 5] were able to produce similar motion segmentation results by minimization of a single energy using either graph cuts or level sets in alternation with parametric motion estimates.

Since motion segmentation is based on intensity comparisons of consecutive frames it typically suffers from two limitations: Firstly, it does not exploit global temporal consistency - the fact that the *same* intensity layer is deformed over the entire sequence is not taken into account. Secondly, occlusion is either penalized heuristically or not modeled at all. Although not resolving these drawbacks in a rigorous manner, Dupont et al. [7] introduced a sophisticated layer model into this framework.

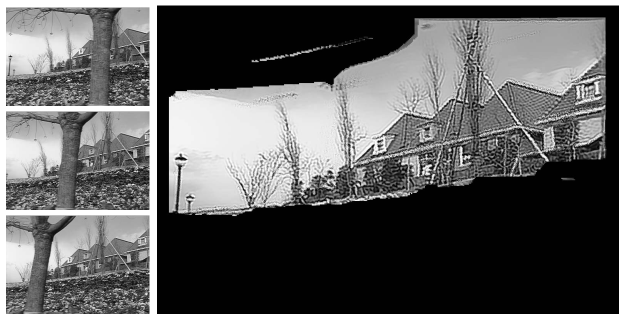
Figure 1. The proposed method is able to remove the tree as well as increase the resolution of the input video.

The above drawbacks are resolved by layer decomposition approaches [1, 10, 16, 13]. Ayer and Sawhney [1], for example, present a coding cost formulation where the layer partitioning is obtained by thresholding soft decisions. No spatial regularity is imposed.

Using Generalized Expectation Maximization, Jojic and Frey [10] were able to produce very convincing reconstructions of the sequence as a *real-valued* superposition of layers. Yet, their approach does not resolve the problem addressed in this work, namely solving the *hard* decision of which video pixel belongs to which layer and simultaneously estimating spatially regularized coherently moving intensity layers.

More recently, Xiao and Shah [16] suggested to compare each frame of the video to a reference frame. They introduced the occlusion order constraint, penalized occlusions by heuristic cost and optimized by a multi-step approach involving a combination of graph cuts and level sets. As the occlusion order constraint only holds for short sequences, they need to decompose the sequence into smaller parts.

Kumar et al. [13] propose a sophisticated model including motion blur and changes in lighting. They use a seven-step optimization involving graph cuts and belief propagation. While this leads to good results on sequences con-

[*]Research supported by the German Research Foundation, grants #CR-250/1-1 and CR-250/2-1.

taining articulated motion, in this paper we show that on sequences containing rigid body motion one can do better.

Contribution. In this paper, we introduce an algorithm to decompose a video sequence into a set of moving, spatially super-resolved layers. By minimizing an energy reflecting the coding cost, we simultaneously estimate layer geometries, super-resolved layer intensities, layer ordering and motion fields.

Our work aims at reconstructing sharp layer images as demonstrated in Figure 1. To this end we introduce a realistic model of the image formation process (also known as super-resolution) into our framework.

Both super-resolution and coding cost formulation give rise to regularization in the layer space. While optimization with respect to the layer intensities leads to a variant of *total variation* filtering [14, 17, 8], optimizing the layer geometries is a major contribution of this paper: We show how to convert the arising constrained optimization problem into an unconstrained one and deal with arising high-order terms by a generalization of expansion moves. Lastly we emphasize that our method allows to model occlusions very naturally.

2. Motion Layer Decomposition by Minimizing Coding Cost

In the following, we show that layer decomposition can be seen as a problem of video coding, where the layers and their motion in time are estimated by minimizing the coding cost. The input video is given on a discrete set of spatio-temporal pixels $\mathbb{X}$:

$$I : \mathbb{X} \to I\!R$$

The task is to decompose the sequence into a set of N layers (with pre-defined N) and their motion over time, both modeled as continuous variablesLayers are modeled as functions

$$I_i : \Omega_i \to I\!R, \quad i = 1, \ldots, N$$

defined on domains $\Omega_i \subset I\!R^2$ which themselves are unknowns. These domains define the support of the layer. They correspond to the shaded areas in Figure 2. In the following we encode layer domains in terms of their characteristic function:

$$l_i : I\!R^2 \to \{0,1\}, \quad l_i(\hat{\mathbf{x}}) = \begin{cases} 1, & \text{if } \hat{\mathbf{x}} \in \Omega_i \\ 0, & \text{else.} \end{cases} \tag{1}$$

The motion of layer i over time is modeled by a motion field which defines where a particular point of a layer appears in the video at a given time in the sequence:

$$\mathbf{h}_i : \Omega_i \times [0, T) \to I\!R^2,$$

We fix the coordinate system by enforcing that $\mathbf{h}_i(\mathbf{x}, 0) = \mathbf{x}$ for all i, i.e. at time 0 the layer i maps directly to the

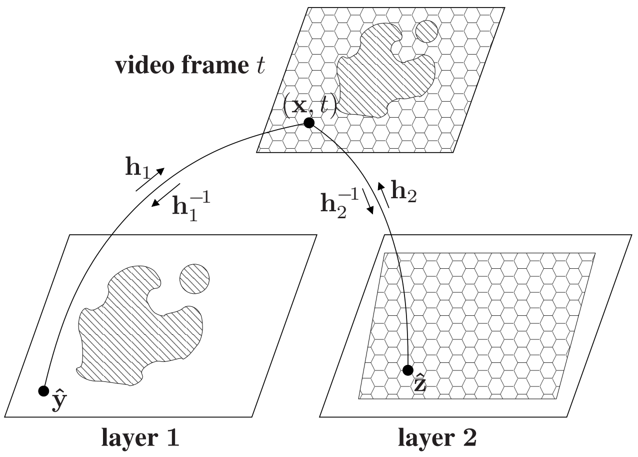

Figure 2. Illustration of layer order (here for $N = 2$ layers), motion functions and labelings. White regions are not in the support of the layer. The following equations hold: $(\mathbf{x}, t) = \mathbf{h}_1(\hat{\mathbf{y}}, t) = \mathbf{h}_2(\hat{\mathbf{z}}, t)$ and $\hat{\mathbf{y}} = \mathbf{h}_1^{-1}(\mathbf{h}_2(\hat{\mathbf{z}}, t), t)$.

video, without distortion. For the present we assume that the motion functions are invertible for given t, denoting the inverse for t as $\mathbf{h}_i^{-1}(\mathbf{x}, t)$.

We impose a layer order: Layer i may temporarily occlude parts of layer j only if $j > i$. Then, from given layer domains Ω_i one can infer which layer $l(\mathbf{x}, t)$ is visible at which spatio-temporal position in the video:

$$l : \mathbb{X} \to \{1, \ldots, N\}.$$

$$l(\mathbf{x}, t) = \min\{i \mid \mathbf{h}_i^{-1}(\mathbf{x}, t) \in \Omega_i\} \tag{2}$$

This function provides an accurate model of occlusions without heursitic penalty terms. Enforcing this model (in the context of hard decisions w.r.t. layer domains) is a major contribution of this work.

Since the labeling l is induced by the domains Ω_i, it should not be optimized explicitly, as many state-of-the-art layer approaches do. Instead we optimize the layer domains directly. However, it has to be ensured that the entire input video is explained (or coded), which is expressed as the constraint:

$$\forall (\mathbf{x}, t) \in \mathbb{X} \; : \; \{i \mid \mathbf{h}_i^{-1}(\mathbf{x}, t) \in \Omega_i\} \neq \emptyset \tag{3}$$

This constraint enforces that each video pixel be modeled by at least one layer (or equivalently, that l as defined in (2) is well-defined).

2.1. A Basic Coding Cost Formulation

To encode a video sequence by a set of layers, one needs to encode the layer domains, layer appearance within the domains, the motion of layers over time and finally noise on the input video to allow perfect reconstructions. We defer the choice of motion models until section 3.1. The cost for encoding them are denoted as $R(\mathbf{h}_i)$. To encode the layer domains it suffices to encode their boundaries $\partial\Omega_i$. The cost for encoding the layer appearance depends on the compressibility of the intensity, which is well reflected as the integral of the absolute intensity gradient – also known as Total Variation. Assuming Gaussian noise on the input sequence, we arrive at minimizing the coding cost:

Layer Decomposition

$$\begin{aligned} E(\{\Omega_i, I_i, \mathbf{h}_i\}) = &\sum_{(\mathbf{x},t)} \Big(I(\mathbf{x},t) - I_{l(\mathbf{x},t)}\big(\mathbf{h}^{-1}_{l(\mathbf{x},t)}(\mathbf{x},t)\big)\Big)^2 \\ &+ \sum_i R(\mathbf{h}_i) \; + \; \nu \sum_i \big|\partial\Omega_i\big| \\ &+ \lambda \sum_i \int\limits_{\Omega_i} |\nabla I_i(\hat{\mathbf{x}})| \, d\hat{\mathbf{x}} \qquad (4) \end{aligned}$$

subject to (2), (3)

2.2. Coding Cost for Combined Motion Layer Decomposition and Super Resolution

Minimizing (4) typically results in blurry layers. This is partly due to imperfectly estimated motion models but also due to a rather simple the image formation model: The intensity of a pixel is assumed to reflect a single scene point.

To obtain sharp layer images we therefore integrate a sensor model [17, 8] which accounts for camera blur and the area averaging of sensor elements. For scenes generated by a single layer, the intensity recorded in a sensor element (or video pixel) is modeled as a convolution with a Gaussian blurring kernel b followed by an integral over the area $V_{[\mathbf{x},t]}$ of the sensor element:

$$I_{\text{syn}}(\mathbf{x},t) = \frac{1}{\Big|V_{[\mathbf{x},t]}\Big|} \int\limits_{V_{[\mathbf{x},t]}} b * I_1(\mathbf{h}_1^{-1}(\mathbf{x}',t)) \, d\mathbf{x}'$$

Without loss of generality we set the area of sensor elements to $|V_{[\mathbf{x},t]}| = 1$. To model the input sequence as a superposition of N layers we introduce a function expressing whether a layer is visible at a given video pixel or not:

$$\chi_i(\mathbf{x},t) = l_i(\mathbf{h}_i^{-1}(\mathbf{x},t)) \cdot \prod_{j<i} \Big(1 - l_j\big(\mathbf{h}_j^{-1}(\mathbf{x},t)\big)\Big) \quad (5)$$

This expression is 1 if and only if layer i models the video pixel and no layer with smaller order also models the pixel. Now the image formation process is expressed as

$$\begin{aligned} I_{\text{syn}}(\mathbf{x},t) \;&=\; \int\limits_{V_{[\mathbf{x},t]}} b * \left[\sum_i \chi_i(\mathbf{x}',t) I_i(\mathbf{h}_i^{-1}(\mathbf{x}',t)) \right] d\mathbf{x}' \\ &\approx \int\limits_{V_{[\mathbf{x},t]}} \sum_i \chi_i(\mathbf{x}',t) \big[b * I_i(\mathbf{h}_i^{-1}(\mathbf{x}',t)) \big] d\mathbf{x}' \quad (6) \end{aligned}$$

where for computational simplicity we have neglected motion blur across layer boundaries. The corresponding coding cost are

Superresolution Layer Decomposition

$$\begin{aligned} E(\{\Omega_i, I_i, \mathbf{h}_i\}) \;=\; &\sum_{(\mathbf{x},t)} \big(I(\mathbf{x},t) - I_{\text{syn}}(\mathbf{x},t) \big)^2 \\ &+ \sum_i R(\mathbf{h}_i) \; + \; \nu \sum_i \big|\partial\Omega_i\big| \\ &+ \lambda \sum_i \int\limits_{\Omega_i} |\nabla I_i(\hat{\mathbf{x}})| \, d\hat{\mathbf{x}} \qquad (7) \end{aligned}$$

subject to (3), (5)

3. Optimization

Minimizing (7) – subject to (3) – is a difficult problem. Our algorithm is based on two phases, each one using the alternating minimization principle: In the first phase, we minimize functional (4) using a multi-scale scheme. In the second phase we include the more accurate sensor model and minimize (7).

In either phase we alternatingly solve for the appearance and support of the layers as well as their motion, keeping the other quantities constant. We also optimize the layer order: For two layers, we try both possible orders when updating layer support. For $N > 2$ we assume that faster moving layers are in front of slower moving ones. This assumption includes - among others - all cases of a static scene with a moving camera.

3.1. Choice and Update of the Motion Functions

So far we have left the motion models $\mathbf{h}_i$ unspecified. For the first phase we use a parametric model. For the second phase this can optionally be augmented by a nonparametric field.

3.1.1 Parametric Layer Motion

We use a parametric model of the form:

$$\mathbf{h}_i(\hat{\mathbf{x}},t) = \hat{\mathbf{x}} + \underbrace{\begin{pmatrix} \hat{x}t & \hat{y}t & t & t^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \hat{x}t & \hat{y}t & t & t^2 \end{pmatrix}}_{\mathbf{S}(\hat{\mathbf{x}},t)} \boldsymbol{\vartheta}_i \quad (8)$$

In particular this model includes affine motion. The costs for coding the parameter vectors $\boldsymbol{\vartheta}_i$ are negligible, so we set $R(\mathbf{h}_i)$ to 0.

These parameters are updated in phase 1, then kept fixed for phase 2. In phase 1 we compute parameter increments

$$\begin{aligned} \boldsymbol{\Delta\vartheta_i} \;=\; &\mathbf{M}_i^{-1} \cdot \Big(\sum_{\hat{\mathbf{x}},t} \chi_i(\mathbf{h}_i(\hat{\mathbf{x}},t),t) \cdot \\ &\Big(I_i(\hat{\mathbf{x}},t) - I(\mathbf{h}_i(\hat{\mathbf{x}},t),t)\Big) \mathbf{S}(\hat{\mathbf{x}},t)^\top \boldsymbol{\nabla} I(\mathbf{h}_i(\hat{\mathbf{x}},t),t) \Big) \end{aligned}$$

with

$$\mathbf{M}_i \;=\; \lambda\,\mathbf{I} + \sum_{\hat{\mathbf{x}},t}\Big[\chi_i(\mathbf{h}_i(\hat{\mathbf{x}},t),t)\cdot \mathbf{S}(\hat{\mathbf{x}},t)^\top \boldsymbol{\nabla} I(\mathbf{h}_i(\hat{\mathbf{x}},t),t)\boldsymbol{\nabla} I(\mathbf{h}_i(\hat{\mathbf{x}},t),t)^\top \mathbf{S}(\hat{\mathbf{x}},t)\Big]$$

Here $\mathbf{I}$ is the identity matrix. If the update results in a smaller energy it is accepted and λ divided by 10, otherwise λ is multiplied by 10 and the update discarded. This is the well-known Levenberg-Marquardt algorithm.

3.1.2 Nonparametric Layer Motion

In the second phase optionally a nonparametric field is added. Here we drop the requirement of the invertible motion model. Instead we only model the mapping of video pixels to layer positions:

$$\tilde{\mathbf{h}}_i^{-1}(\mathbf{x},t) = \mathbf{h}_i^{-1}(\mathbf{x},t) + \mathbf{h}_{i,\mathrm{np}}^{-1}(\mathbf{x},t) \tag{9}$$

Coding the nonparametric fields $\mathbf{h}_{i,\mathrm{np}}^{-1}$ now invokes significant cost, where the cost depends on the compressibility of the motion fields. We choose

$$R(\mathbf{h}_{i,\mathrm{np}}^{-1}) = \alpha \sum_{(\mathbf{x},t)\in\mathbb{X}} \left|\nabla \mathbf{h}_{i,\mathrm{np}}^{-1}(\mathbf{x},t)\right|^2$$

where ∇ denotes the spatial (2D) derivative. This is the well-known regularization term of Horn and Schunck [9].

Updates of the nonparametric fields are computed by first warping the layer to each video frame using the existing motion model, then computing an increment between the warped layer and the input frame with the method of Horn and Schunck.

3.2. Update of the Layer Appearance

With respect to the layer intensities, functional (7) is convex, so gradient descent leads to the globally optimal layer appearance. The functional derivative in the direction η is given as

$$\left.\frac{\partial E}{\partial I_i}\right|_{\eta} = \lim_{\epsilon\to 0}\frac{1}{\epsilon}\Big[E(I_i+\epsilon\eta) - E(I_i)\Big]$$

A detailed calculation leads to the gradient descent

$$\begin{aligned}\frac{\partial I_i}{\partial t} &= -\frac{\partial E}{\partial I_i}\\ &= 2\Big[\sum_{\mathbf{x},t}\chi_i(\mathbf{x},t)\Big(I(\mathbf{x},t) - \int_{V_{[\mathbf{x},t]}} b * I_i(\mathbf{h}_i^{-1}(\mathbf{x}'))d\mathbf{x}'\Big)\\ &\qquad\cdot \int_{V_{[\mathbf{x},t]}} b(\mathbf{x}'-\mathbf{h}_i(\hat{\mathbf{y}}))\,d\mathbf{x}'\Big]\left|\frac{d\mathbf{h}_i}{d\hat{\mathbf{y}}}\right|\\ &\quad+\lambda\,\mathrm{div}\left(\frac{\nabla I_i(\hat{\mathbf{y}})}{|\nabla I_i(\hat{\mathbf{y}})|}\right)\end{aligned}$$

This evolution equation can be interpreted as follows: The first term drives the layer intensities (after blurring) towards the observed intensities, whereas the second leads to a non-linear, discontinuity preserving diffusion.

To deal with the rather large number of intensity variables we resort to a GPU-based implementation. Each of the layers in Figure 6 is then estimated in 30 seconds (30 input frames of size 350×240, triple super-resolution).

In the first phase the integral and the convolution are removed and the TV-term neglected. The layer intensities are then given by averaging over input intensities.

3.3. Update of the Layer Support

When updating the layer support one has to keep in mind that the functionals (4) and (7) are minimized subject to the constraint (3) that they explain the whole video. The following construction applies to both functionals, so we just write $E()$ for the energy. The key idea is to cast the constrained optimization problem as an unconstrained one, expressing the domains Ω_i by their characteristic functions l_i (1):

$$\min_{\{l_i\}} E(\{l_i\},\{\mathbf{h}_i, I_i\}) + \gamma\sum_{\mathbf{x},t}\prod_{i=1}^{N}(1 - l_i(\mathbf{h}_i^{-1}(\mathbf{x},t))) \tag{10}$$

The additional term is 0 exactly when the layer supports explain (or code) the whole video. Hence the constraint is fulfilled by choosing γ large enough. The resulting optimization problem is a *binary* labeling problem, independent of the number N of layers.

For minimization we distinguish two cases: For two layers we find the global optimum, whereas for more than two layers an iterative scheme is used.

3.3.1 The Case of Two Layers

For the case of two layers (10) can be written as a sum of unary and binary submodular terms. This can be minimized globally using graph cuts [11].

Each layer position corresponds to a variable in the labeling problem. The spatial smoothness term (the third term in (7)) is approximated by connecting every layer pixel with its 8 neighbors in the same layer (compare [3]). The TV-term is approximated by unary terms, one for every layer pixel[1].

The data terms and the constraint terms in (10) are grouped together: For each *video* pixel $(\mathbf{x},t)$ there is a binary term connecting the *layer* pixel $\mathbf{h}_1^{-1}(\mathbf{x},t)$ in layer 1 and the *layer* pixel $\mathbf{h}_2^{-1}(\mathbf{x},t)$ in layer 2. By inverting the labeling of layer 2, submodular terms are obtained. For functional (4), the arising terms are given in table 1. For functional (7) it suffices to change the data term.

[1]At the domain boundary of the layers this is only an approximation.

$E^{\hat{\mathbf{y}},\hat{\mathbf{z}}}(0,0)$	$\left[I(\mathbf{x},t) - I_2(\mathbf{h}_2^{-1}(\mathbf{x},t))\right]^2$
$E^{\hat{\mathbf{y}},\hat{\mathbf{z}}}(0,1)$	γ
$E^{\hat{\mathbf{y}},\hat{\mathbf{z}}}(1,0)$	$\left[I(\mathbf{x},t) - I_1(\mathbf{h}_1^{-1}(\mathbf{x},t))\right]^2$
$E^{\hat{\mathbf{y}},\hat{\mathbf{z}}}(1,1)$	$\left[I(\mathbf{x},t) - I_1(\mathbf{h}_1^{-1}(\mathbf{x},t))\right]^2$

Table 1. Binary term arising for a pixel $\hat{\mathbf{y}}$ in layer 1 and a pixel $\hat{\mathbf{z}}$ in layer 2, mapping to the same video pixel $(\mathbf{x}, t)$ for time t (i.e. $\mathbf{h}_1^{-1}(\mathbf{x},t) = \hat{\mathbf{y}}$ and $\mathbf{h}_2^{-1}(\mathbf{x},t) = \hat{\mathbf{z}}$). For $\hat{\mathbf{y}}$, a label 1 means that $\hat{\mathbf{y}} \in \Omega_1$, for $\hat{\mathbf{z}}$ label 0 means $\hat{\mathbf{z}} \in \Omega_2$. Submodularity is given as $E(0,0) + E(1,1) \leq E(0,1) + E(1,0)$ for γ large enough. For the data terms, we use bilinear interpolation on the layer intensities.

3.3.2 The Multi-layer Case

For the N-layer case, functional (10) contains terms of order N, two for every video pixel. One is the constraint term introduced in (10). The other is the data term: It depends on which layer is visible at the video pixel. Both terms depend on the same set of layer positions (one in each layer).

As these terms are not submodular, the functional cannot be optimized globally using graph cuts. Instead we adapt the notion of expansion moves [4] to the *binary* labeling problem (10). So far expansion moves are only defined for *multi*-label problems.

In the expansion move for layer i a layer position in the space of layer i can *join* the support of layer i. Simultaneously, positions in the space of layers $j \neq i$ are allowed to *leave* the support of the respective layer. Starting from a set of supports which explain the whole video, this constraint is kept satisfied during the entire expansion move process.

With the help of an auxiliary variable for every *video* pixel $(\mathbf{x}, t)$, the N-th order terms are decomposed into a sum of N binary terms. For the new variables a binary labeling function $\bar{l}(\mathbf{x}, t)$ is introduced. This function encodes the two possibilities concerning $(\mathbf{x}, t)$ in the expansion move of layer i: Either the pixel becomes modeled by layer i or it remains modeled by the previously indicated layer – say j:

$$\bar{l}(\mathbf{x},t) = \begin{cases} 1 & \text{if } i \text{ is visible at } (\mathbf{x},t) \\ 0 & \text{if } j \text{ is visible at } (\mathbf{x},t) \end{cases}$$

The data term is now a sum of unary terms, depending only on $\bar{l}(\cdot)$. To ensure that $\bar{l}$ is consistent with the layer supports l_i, for each video position two terms are added. One penalizes the constellation where $\bar{l}(\mathbf{x}, t)$ is 1 but $\mathbf{h}_i^{-1}(\mathbf{x}, t)$ is not in the support of layer i, the other the case where $\bar{l}(\mathbf{x}, t)$ is 0 but $\mathbf{h}_j^{-1}(\mathbf{x}, t)$ is not in the support of layer i. Both are weighted with γ. When combined with these terms, the labeling $\bar{l}(\cdot)$ ensures that the entire video is explained.

One issue remains to be encoded: The visibility must respect the layer order. If a video pixel $(\mathbf{x}, t)$ becomes modeled by i then no layer position in $\mathbf{h}_k^{-1}(\mathbf{x}, t)$ in the space of a layer $k < i$ can be in the support of the respective layer. Likewise, if $(\mathbf{x}, t)$ remains modeled by j and j is smaller than i, then $\mathbf{h}_i^{-1}(\mathbf{x}, t)$ cannot join the support of layer I. The respective constellations are penalized by γ.

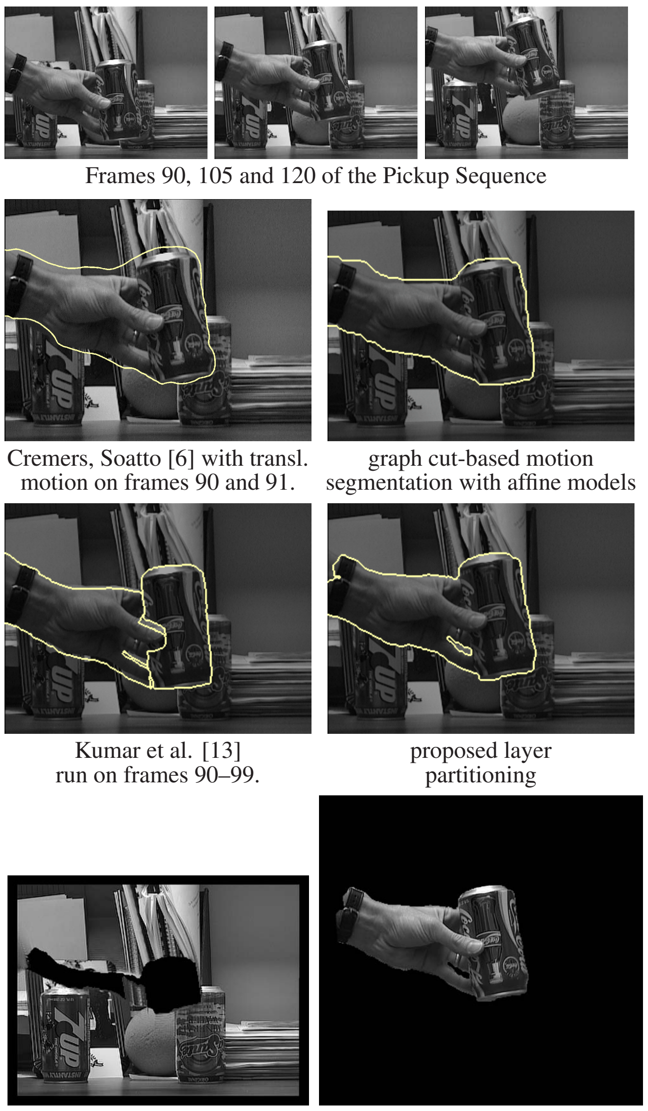
Frames 90, 105 and 120 of the Pickup Sequence
Cremers, Soatto [6] with transl. motion on frames 90 and 91.
graph cut-based motion segmentation with affine models
Kumar et al. [13] run on frames 90–99.
proposed layer partitioning
high-resolution layer images

Figure 3. Comparison on the Pickup Sequence: With the proposed method, the tightest boundaries are obtained.

When representing l_i straight and l_k inverted for $k \neq i$, the resulting terms are submodular. Spatial smoothness and the TV terms are included as in the two-layer case.

4. Experiments

We demonstrate that the proposed method is able to decompose real image sequences into sets of sharp layers. Without the physical details of the image formation process, blurry layers are obtained. Moreover, the generated partitionings compare favorably to those generated by state-of-the-art methods in layer estimation and motion segmentation. We use a TV-weight of $\lambda = 60$ and a Gaussian blurring

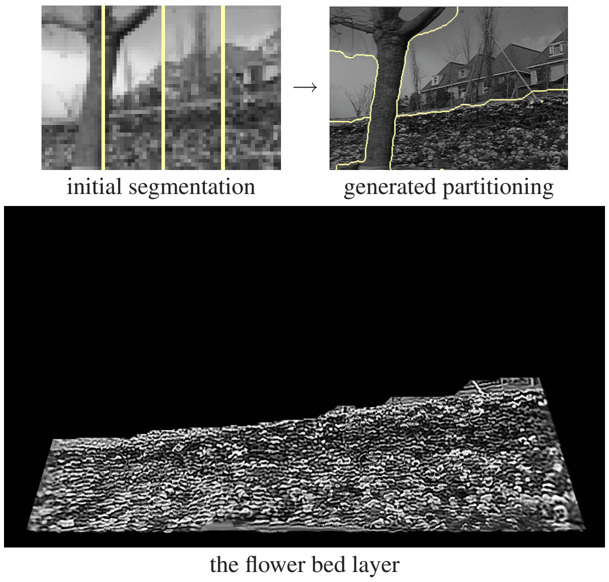

Figure 5. Starting from a generic initialization the algorithm generates a partitioning as well as a set of super-resolved layers.

kernel with $\sigma = 2.25$ for the superresolution model.

4.1. Comparison to the State-of-the-art

Figure 3 presents results on 30 frames of the pickup sequence, using the parametric motion model (8). Compared to motion segmentation, our method results in tighter layer boundaries especially where layers become occluded. We implemented Motion Competition [6] as well as a graph cuts-based variant of Cremers' and Soatto's Space-Time Motion Competition [5]. The latter additionally includes warping in a multi-scale scheme.

In contrast to the method of Kumar et al. [13] our method captures the entire foreground region in a single layer. As their method includes intensity-based segmentation, it separates the hand from the can and identifies a part of the finger nail as background. For this experiment, only the first ten frames were used: When run their method on the entire sequence, the rotation of the can caused substantial drift [12].

As demonstrated in Figure 4, our super-resolved layers reveal much finer details than present in any of the input frames. Such fine details are not obtained by the traditional way of layer estimation.

4.2. Multiple Layers and Nonparametric Motion

The Figures 5 and 6 demonstrate that our method is able to produce multi-layer decompositions: The proposed method was run with 4 layers, one of which vanished during optimization. Three of the input frames are shown in Figure 1 on page 1. Figure 5 demonstrates that starting from a very simple initialization our method produces a layer partitioning close to human perception. The Figure also shows the flower bed layer.

The tree layer and the layer containing the houses are shown in Figure 6. Here objects of different depths are grouped together in the same layers: The twigs of the tree have higher depth than the stem. Also the lantern is closer to the camera than the houses. Here the use of nonparametric motion fields is sensible. As shown in Figure 6 the mentioned objects become much sharper with this model. Experimentally we found that the smoothness weight α for the motion fields does not much influence results. We used a value of 500.

5. Conclusion

We propose an energy minimization method to decompose a video sequence into a superposition of high-resolution moving layers. In contrast to existing approaches to layer decomposition, we provide a consistent model of occlusion and incorporate details of the image formation process. This gives rise to layers which are sharper than individual input images. The proposed energy is minimized by combining discrete and continuous optimization: Layer geometry is estimated by graph cuts in the (dual) layer space which allows to impose regularity and consistency of the layers, while layer motion and intensity are estimated by optic flow methods and total variation deblurring respectively.

References

[1] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum likelihood estimation of mixture models and MDL encoding. In *IEEE Int. Conf. on Comp. Vision*, pages 777–784, Boston, USA, 1995.

[2] S. Birchfield and C. Tomasi. Multiway cut for stereo and motion with slanted surfaces. In *IEEE Int. Conf. on Comp. Vision*, pages 489–495, 1999.

[3] Y. Boykov and V. Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *IEEE Int. Conf. on Comp. Vision*, volume 1, pages 26–33, Nice, France, 2003.

[4] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 23(11):1222–1239, 2001.

[5] D. Cremers and S. Soatto. Variational space-time motion segmentation. In B. Triggs and A. Zisserman, editors, *IEEE Int. Conf. on Comp. Vision*, volume 2, pages 886–892, Nice, Oct. 2003.

[6] D. Cremers and S. Soatto. Motion Competition: A Variational Framework for Piecewise Parametric Motion Segmentation. *Int. J. of Comp. Vision*, 62(3):249–265, May 2005.

[7] R. Dupont, O. Juan, and R. Keriven. Robust segmentation of hidden layers in video sequences. In *Int. Conf. on Pattern Recognition*, Hong Kong, 2006.

[8] S. Farsiu, M. D. Robinson, M. Elad, and P. Milanfar. Fast and robust multiframe super resolution. *IEEE Trans. on Image Processing*, 13(10):1327–1344, 2004.

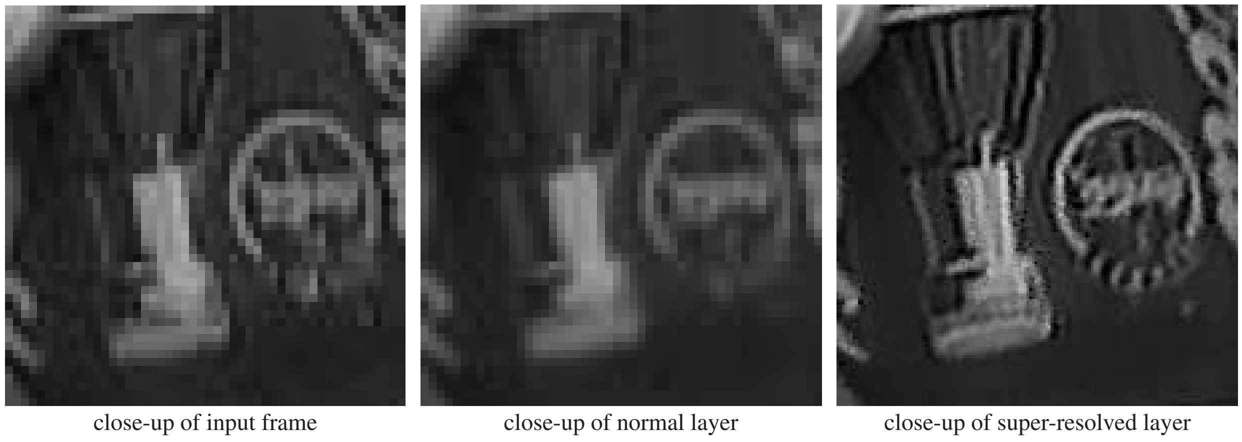

Figure 4. Close-up on Figure 3: While the normal layer is smooth and blurry due to averaging along the trajectories, the super-resolved layer recovers details that are not visible in any input frame.

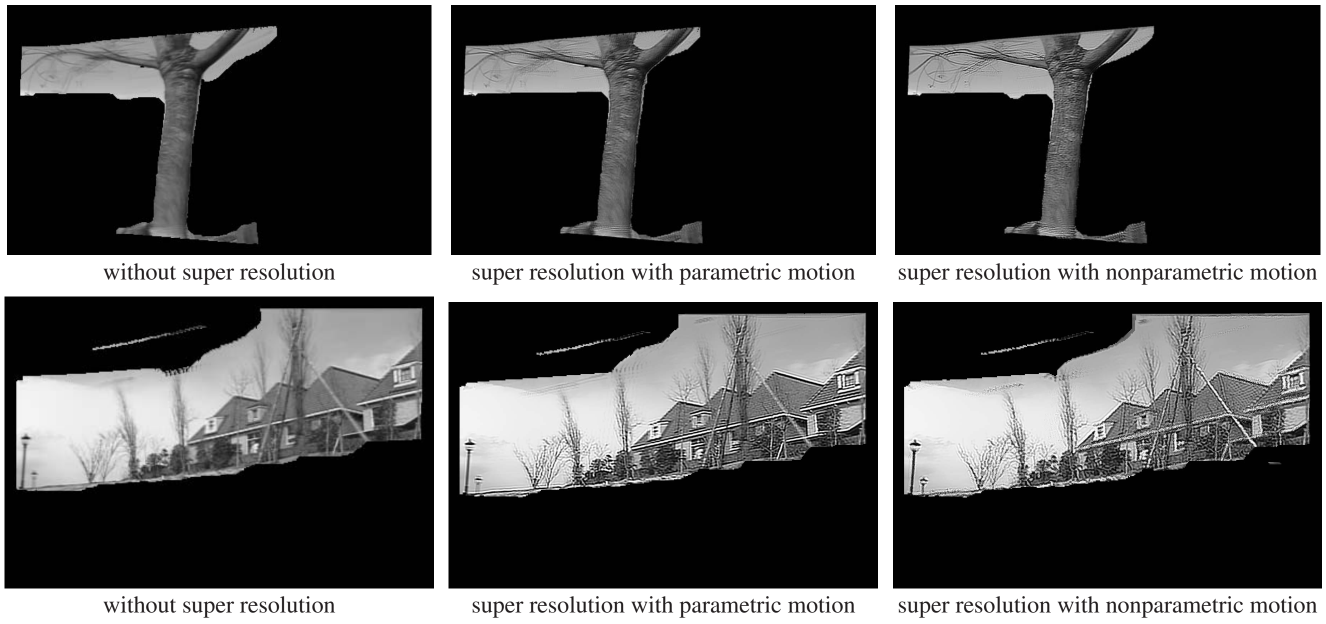
without super resolution
super resolution with parametric motion
super resolution with nonparametric motion
without super resolution
super resolution with parametric motion
super resolution with nonparametric motion

Figure 6. Effect of the choice of motion models: By combining super-resolution and nonparametric motion estimation, scene details such as the stem and the twigs of the tree as well as the lantern become sharper.

[9] B. Horn and B. Schunck. Determining optical flow. *A.I.*, 17:185–203, 1981.

[10] N. Jojic and B. Frey. Learning flexible sprites in video layers. In *IEEE Int. Conf. on Comp. Vision and Patt. Recog.*, Maui, Hawaii, 2001.

[11] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 24(5):657–673, 2004.

[12] M. P. Kumar and P. H. S. Torr. Personal correspondence, 2007.

[13] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Learning layered motion segmentations of video. *Int. J. of Comp. Vision*, 2007. To appear.

[14] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[15] J. Wang and E. Adelson. Representing moving images with layers. *IEEE Trans. on Image Processing*, 3(5):625–638, 1994.

[16] J. Xiao and M. Shah. Motion layer extraction in the presence of occlusion using graph cuts. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 27(10):1644–1659, 2005.

[17] Y. You and M. Kaveh. A regularization approach to joint blur identification and image restoration. *IEEE Trans. on Image Processing*, 5:416–428, 1996.