Robust Segmentation of Hidden Layers in Video Sequences

Romain Dupont
Olivier Juan
Renaud Keriven

Research Report 06-21
January 2006

Centre d'Enseignement et de Recherche
en Technologies de l'Information et Systèmes

CERTIS - ENPC
http://www.enpc.fr/certis
Ecole Nationale des Ponts et Chaussées
77455 Marne - Paris - France

Robust Segmentation of Hidden Layers in Video Sequences

Segmentation Robuste des Couches Cachées dans les Séquences Vidéos

Romain Dupont[1]
Olivier Juan[1] Renaud Keriven[1]

[1] CERTIS - ENPC - http://www.enpc.fr/certis - 77455 Marne - Paris - France

Abstract

In this paper, we propose a novel and robust method for extracting motion layers in video sequences. Taking advantage of temporal continuity, our framework considers both the visible and the hidden parts of each layer in order to increase robustness. Moreover, *the hidden parts of the layers are recovered*, which could be of great help in many high level vision tasks. Modeling the problem as a labeling task, we state it in a MRF-optimization framework and solve it with a graph-cut algorithm. Both synthetic and real video sequences show a visible layers extraction comparable to the one usually performed by state of the art methods, as well as a novel and successful segmentation of hidden layers.

Résumé

Dans ce papier, on propose une nouvelle et robuste méthode pour extraire les couches de même mouvement dans les séquences vidéos. En prenant profit de la continuité temporelle, notre cadre considère aussi bien les parties visibles et cachées de chaque couche dans le but d'améliorer la robustesse. D'autre part, *les parties cachées sont extraites*, permettant de nombreuses applications dans le domaine de la vision (opérations de haut niveaux notamment). Modélisant le problème en tant que problème de classification, on le formule dans un cadre d'optimisation MRF et nous le résolvons avec l'algorithme des Graph Cuts. Des séquences vidéos à la fois réelles et synthétiques montrent une extraction avec succès des couches visibles similaire aux méthodes de l'état de l'art, de même pour les couches cachées.

Contents

1 Introduction

We consider the extraction of the layers composing a video sequence, each of them being approximated by a planar set of objects having the same parametric motion. This well studied representation (see [1, 2, 3, 4, 6, 8, 10, 11, 12, 13, 14]) offers a good trade-off between low- and high-level of information for numerous applications, such as robust motion segmentation, efficient video compression, 3D reconstruction of urban scenes, etc. The main issues addressed in this context are the estimation of the motion of the layers, the outliers and occlusion detection, the determination of the number of layers, the choice of regularization criteria and the accuracy and robustness of the segmentation.

In [15], Xiao and Shah present a method based on temporal constraints between a frame and its successors ($1 \mapsto 2$, $1 \mapsto 3$, $1 \mapsto 4$, ...) that takes into account what they call occlusions (actually, point modeling two distinct phenomenons: (i) objects becoming hidden and (ii) noisy point with impossible tracking). Their method does not intrinsically give smooth segmentations from one frame to the other as frames are processed independently.

On the contrary, our method takes advantage of temporal information for the whole sequence. Indeed, it simultaneously processes all the sequence considering temporal constraints between successive frames $1 \mapsto 2 \mapsto 3 \mapsto 4 \mapsto$ ..., guaranteeing a smooth labeling. Furthermore, it explicitly recovers the hidden parts of the layers, that can disappear behind an another one and re-appear a few frames later: *a disappearing point is not only detected like in [15] but also tracked while being hidden until it re-appears*! Finally, tracking both visible and hidden parts of layers reduces segmentation ambiguities, namely the number of *undefined* points (see further).

Hidden layers. For each pixel, we consider its corresponding visible layer and all hidden layers if any. Given n, the number of layers, we associate each pixel $\mathbf{x}$ with its label $l_{\mathbf{x}} = (v_{\mathbf{x}}, \mathbf{h}_{\mathbf{x}}) \in \mathcal{L}$, with $\mathcal{L} = (\mathcal{V} \times \mathcal{H}) \setminus \mathcal{F}$, where $\mathcal{V} = [1, n] \cup \{\varnothing_{\mathcal{V}}\}$ is the visible space, $\mathcal{H} = \{\mathbf{false}, \mathbf{true}\}^n$ is the hidden one and $\mathcal{F}$ refers to forbidden combinations (see further). The special label $\varnothing_{\mathcal{V}}$ corresponds to an indetermination on the visible layer choice (*undefined pixels* or *"outliers"*). The i^{th} coordinate $\mathbf{h}^i_{\mathbf{x}}$ of vector $\mathbf{h}_{\mathbf{x}}$ indicates the hidden state of the i^{th} layer ($\mathbf{true}$ if hidden, **false** if visible or non present). For a given pixel, a layer cannot be both visible and hidden, i.e. $\mathbf{h}^{v_{\mathbf{x}}}_{\mathbf{x}} \neq \mathbf{true}$: $\mathcal{F}$ is the set of such forbidden cases. Figure 1 illustrates such a labeling.

The reminder of this paper is organized in the following way. Section 2 presents the energy used for classification. Section 3 provides some important information about the implementation and shows results on both synthetic and real data. The last section gives some conclusion and future directions.

Motion model. We note $\mathcal{T}^t_v$ the parametric motion of layer v between frames t

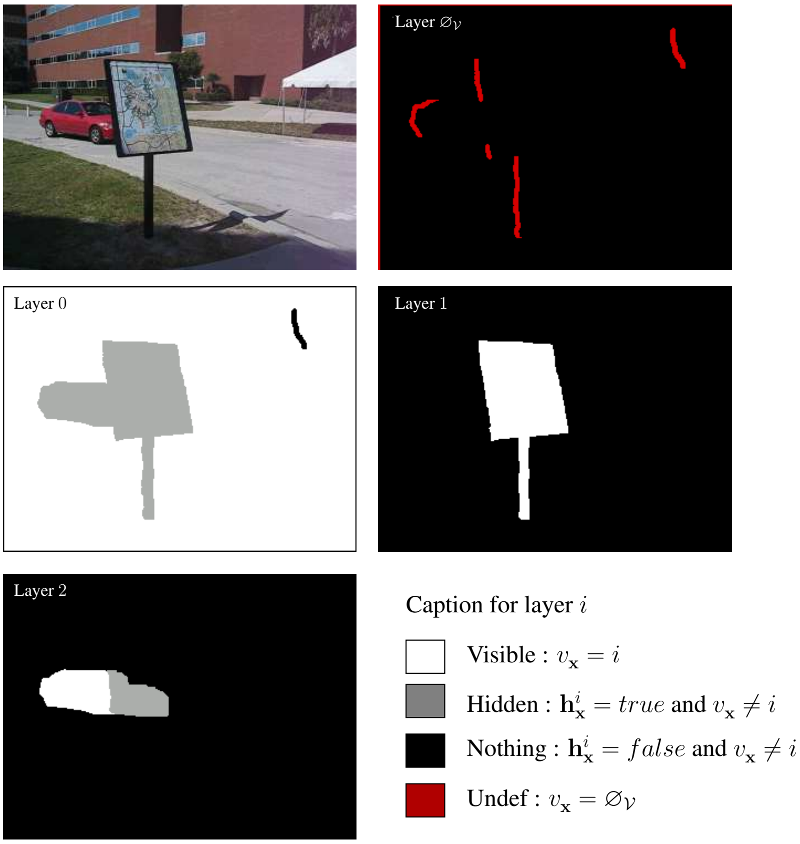

Layer $\varnothing_{\mathcal{V}}$
Layer 0
Layer 1
Layer 2
Caption for layer i
□ Visible : $v_{\mathbf{x}} = i$
▨ Hidden : $\mathrm{h}^i_{\mathbf{x}} = true$ and $v_{\mathbf{x}} \neq i$
■ Nothing : $\mathrm{h}^i_{\mathbf{x}} = false$ and $v_{\mathbf{x}} \neq i$
🟥 Undef : $v_{\mathbf{x}} = \varnothing_{\mathcal{V}}$

Figure 1: Example of labeling. Note that these images **are not the results** obtained by our algorithm but an example of what could be a reasonable segmentation.

and $t+1$. No motion is associated to layer $\varnothing_{\mathcal{V}}$. Our experiments use classical projective motions, thus approximates the scene by three-dimensional plane objects, although any other model could be used (e.g. affine). Motion estimation follows our previous work [6] and will not be detailed here, though any other equivalent method could be used.

Initialization. Our method is initialized with n pre-computed layers (accurate or not), obtained through pre-existent methods like the ones in [6, 15]. When the correspondences between the layers of successive frames is not explicitly given by this initial segmentation, we recover it easily, associating a layer v at time t to the one at time $t+1$ that most overlaps its image through $\mathcal{T}tt + 1v$.

Overall process. Our method consists in alternating, until some stabilization: (i) layer segmentation and (ii) refinement of the motion parameters from the visible part of the layers (which, again, will not be detailed here).

2 Classification

Given T frames, n layers, and $\mathcal{T}_v^t(v \in [1,n], t \in [1,T])$ their motion models [2], we consider the labeling problem consisting in determining a function $L : (\mathbf{x}, t) \mapsto l_\mathbf{x}^t = (v_\mathbf{x}^t, \mathbf{h}_\mathbf{x}^t) \in \mathcal{L}$. We plug the problem into a variational framework and will design in the sequel an energy that L should minimize. Note that we consider a constant number of layers throughout the sequence. Such a limitation could be relaxed through appropriate methods.

2.1 Motion energy

The motion energy is based on visible parts of the layers and is indeed related to the images ("*data term*"). The *forward* motion residual $r_v(\mathbf{x})$ for the pixel $\mathbf{x}$ under motion $\mathcal{T}_v$ is defined by:

$$r_v^t(\mathbf{x}) = \left\| I^t(\mathbf{x}) - I^{t+1}(\mathcal{T}_v^t(\mathbf{x})) \right\| \tag{1}$$

where I^t is the image at time t. To reduce the influence of high motion residuals, we apply a smoothed Heaviside operator ψ (Fig. 2) given by:

$$\psi(r_v) = \tan^{-1}\left(r_v^2 - \tau\right) + \pi/2 \tag{2}$$

We define a labeling cost function d_I by:

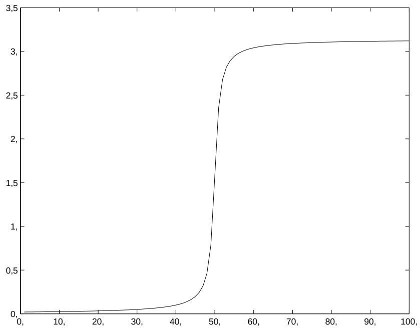


Figure 2: Smoothed Heaviside operator ψ shape (with $\tau = 50$).

$$d_I(l_\mathbf{x}, \mathbf{x}) = \begin{cases} \psi\left(r_{v_\mathbf{x}}(\mathbf{x})\right) & \text{if } v_\mathbf{x} \in [1,n] \\ \psi_{undef} & \text{if } v_\mathbf{x} = \varnothing_V \end{cases} \tag{3}$$

where the parameter ψ_{undef} adjusts the classification of pixels as undefined. The *forward* motion energy E_{FM}^t is then, for a given frame t:

$$E_{FM}^t(L) = \int_\Omega d_I(l_\mathbf{x}^t, \mathbf{x}) d\mathbf{x} \tag{4}$$

[2] when explicitly needed, the frame number t will be indicated by a superscript

where Ω is the image domain. To increase robustness, we also consider the *backward* motion residual (as in [11]) and its associated energy noted $E^t_{BM}(L)$. It is defined similarly, considering frame $t-1$ instead of frame $t+1$ and the reverse motion $(\mathcal{T}^t_v)^{-1}$ instead of $\mathcal{T}^t_v$. Moreover, we embed this criterion into a temporal-multiscale framework, considering also the motion residuals between frame t and frames $t+1, t+2, t+3, \cdots, t-1, t-2, t-3, \cdots$ to handle small motion cases.

2.2 Spatial regularization

As in every noisy and under-constrained problem, spatial regularization has to be introduced. Both visible and hidden parts of the layers are regularized through the following energy:

$$E^t_S(L) = \iint_{\Omega^2} \phi(||\mathbf{x}-\mathbf{y}||) d^t_S\left(l^t_{\mathbf{x}}, l^t_{\mathbf{y}}\right) d\mathbf{y} d\mathbf{x} \tag{5}$$

where ϕ is some kernel (e.g Gaussian) and $d^t_S(.,.)$ is a dissimilarity measure between two labels. Discontinuous labels for both visible and hidden layers must be penalized. We encourage also the frontier of the layer to belong to pixels with high image gradient. This gives the following function:

$$\begin{aligned} d^t_S(l_{\mathbf{x}}, l_{\mathbf{y}}) &= \mu_V \mathcal{I}\left(v_{\mathbf{x}} \neq v_{\mathbf{y}}\right) \exp\left(-\frac{\|I^t(\mathbf{x}) - I^t(\mathbf{y})\|^2}{2\sigma^2}\right) \\ &+ \mu_H \sum_{i=1}^{n} \mathcal{I}\left(\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\mathbf{y}}\right) \end{aligned} \tag{6}$$

where $\mathcal{I}(i)$ equals 1 if i is true, 0 otherwise, σ is the standard deviation of the norm of the gradient of the images, and (μ_V, μ_H) some constants adjusting spatial regularization with respect to the other energy terms.

2.3 Temporal constraints

Temporal constraints are designed for both temporal smoothness and temporal consistency between visible and hidden layers. To this end, using motion information, we penalize discontinuous labeling between frames. To simplify notations, we note $\mathbf{x}_i = \mathcal{T}^t_i(\mathbf{x})$ the image of $\mathbf{x}$ in frame $t+1$ through the motion of layer i at time t. Our *forward* temporal energy is written as follows:

$$\begin{aligned} E^t_{FT}(L) = \int_{\Omega} \Big[& \mathcal{I}(v_{\mathbf{x}} \neq \varnothing_{\mathcal{V}}) d_V\left(l^t_{\mathbf{x}}, l^{t+1}_{\mathbf{x}_{v_{\mathbf{x}}}}\right) \\ & + \sum_{i=1}^{n} \mathcal{I}(\mathbf{h}^i_{\mathbf{x}} = \mathbf{true}) d^i_H\left(l^t_{\mathbf{x}}, l^{t+1}_{\mathbf{x}_i}\right) \Big] d\mathbf{x} \end{aligned} \tag{7}$$

where $d_V(.,.)$ and $d^i_H(.,.)$ are dissimilarity measures given by:

$$d_V(l_{\mathbf{x}}, l_{\mathbf{y}}) = \begin{cases} 0 & \text{if } v_{\mathbf{x}} = v_{\mathbf{y}} \\ \lambda_H & \text{if } \mathbf{h}^{v_{\mathbf{x}}}_{\mathbf{y}} = \mathbf{true} \\ \lambda_D & \text{otherwise} \end{cases} \tag{8}$$

and:

$$d^i_H(l_{\mathbf{x}}, l_{\mathbf{y}}) = \begin{cases} 0 & \text{if } \mathbf{h}^i_{\mathbf{y}} = \mathbf{h}^i_{\mathbf{x}} \\ \lambda_V & \text{if } v_{\mathbf{y}} = i \\ \lambda_D & \text{otherwise} \end{cases} \tag{9}$$

where λ_H, λ_V and λ_D respectively penalize the following events: hiding, reappearing, and completely disappearing. It can be shown that λ_D has to be chosen greater than λ_V and λ_H (see section 3) and that the following inequality $\lambda_H + \lambda_V \leq \lambda_D$ must be respected .

As in the data term, we also consider *backward* constraints, leading to a symmetric temporal energy E^t_{BT}. Moreover, similarly as for the motion residual, we also embed these temporal constraints into a temporal multiscale framework to increase robustness (especially in cases of slow motions) considering also constraints between frame t and frames $t+1, t+2, t+3, t-1, t-2, t-3, \cdots$ and so on.

2.4 Overall energy

Our overall energy to extract the optimal partition of the T images is finally:

$$E(L) = \sum_{t=1}^{T} \underbrace{E^t_{FM}(L) + E^t_{BM}(L)}_{\text{data term (motion)}} + \underbrace{E^t_S(L)}_{\text{spatial regularization}} + \underbrace{E^t_{FT}(L) + E^t_{BT}(L)}_{\text{temporal constraints}} \tag{10}$$

Next section will describe the optimization process used to minimize this global energy.

3 Energy minimization

We plug our spatially continuous energy minimization problem into a discrete Markov Random Field framework [7]. The global energy (EQ. 10) is discretized considering a 4- or 8- neighborhood for the spatial constraints. Due to its efficiency, we use the *alpha*-expansion algorithm [5, 9] provided that distance function d_S is sub-modular (easy to verify) and that temporal constraints fit also submodularity requirement.

3.1 About the submodularities of the temporal constraints

First, we remind what a submodular function is.

Definition. A sub-modular function $D(.,.)$ verifies $D(l_x, l_y) + D(l_\alpha, l_\alpha) \leq D(l_x, l_\alpha) + D(l_\alpha, l_y)$ for two given pixels $\mathbf{x}$ and $\mathbf{y}$ (see [9] for more details).

To demonstrate that the temporal constraints fit the submodularity requirement, we introduce these two following functions V and H (which depend on d_V and d_H) :

$$\begin{aligned} V_{\mathbf{x},\mathbf{y}}(l_\mathbf{x}, l_\mathbf{y}) &= \mathcal{I}\left(\mathbf{y} = T_{v_\mathbf{x}}(\mathbf{x}) \wedge v_\mathbf{x} \neq \varnothing_\mathcal{V}\right) \cdot d_V(l_\mathbf{x}, l_\mathbf{y}) & (11) \\ H^i_{\mathbf{x},\mathbf{y}}(l_\mathbf{x}, l_\mathbf{y}) &= \mathcal{I}\left(\mathbf{y} = \mathcal{T}_i(\mathbf{x}) \wedge \mathbf{h}^i_\mathbf{x} = \mathbf{true}\right) \cdot d^i_H(l_\mathbf{x}, l_\mathbf{y}) & (12) \end{aligned}$$

Theorem. The function $(V + \sum_i H^i)$ is submodular if λ_D is greater than λ_V and λ_H.

Proof. Summary of the proof: we will show that functions D and H^i are submodular providing $\lambda_V = \lambda_H = \lambda_D$. However, considering some particular cases, we will also show that the function $(D + \sum_i H^i)$ is submodular providing $\lambda_V \leq \lambda_D$ and $\lambda_H \leq \lambda_D$. For the other cases, we use the fact that the sum of two submodular functions is submodular.

First, we consider the function $D()$: the table 1 shows all the cases which give information about the constraints between λ_H and λ_D. Cases V5 and V8 are impossible as a change of visible labeling to v_α implies a change of projected pixel $\mathcal{T}_{v_\alpha}$ to consider: as a consequence, the requirement $\mathbf{y} = \mathcal{T}_{v_\alpha}$ will not then be satisfied anymore except if $\alpha = v_\mathbf{x}$ [3]. Valid cases V3 and V6 show that the following equality $\lambda_D = \lambda_H$ must be respected. And similarly for H^i as shown in table 2 : valid cases H3 and H6 constrain the following equality $\lambda_D = \lambda_V$.

However, for the cases V3 and H3 (which force λ_H and λ_V to be greater than λ_D), one can see that the function $(D + \sum_i H^i)$ is actually submodular without any constraints on λ_H, λ_V and λ_D (as shown in figure 3).

For the other valid cases, $D()$ and $H^i()$ (and so $D + \sum_i H^i$) are submodular providing that $\lambda_V \leq \lambda_D$ and $\lambda_H \leq \lambda_D$.

□

Furthermore, one can see that the following inequality $\lambda_H + \lambda_V \leq \lambda_D$ must be respected if we want hidden parts of the layers to be recovered. Indeed, if not, the cost of a disparition to a hidden layer (cost : λ_H) followed by an apparition to a visible layer (cost : λ_V) would be coster than a disparition to '*nothing*', i.e. to any hidden layer, which would only cost λ_D (indeed, in such case, there is no apparition constraint, so no apparition cost).

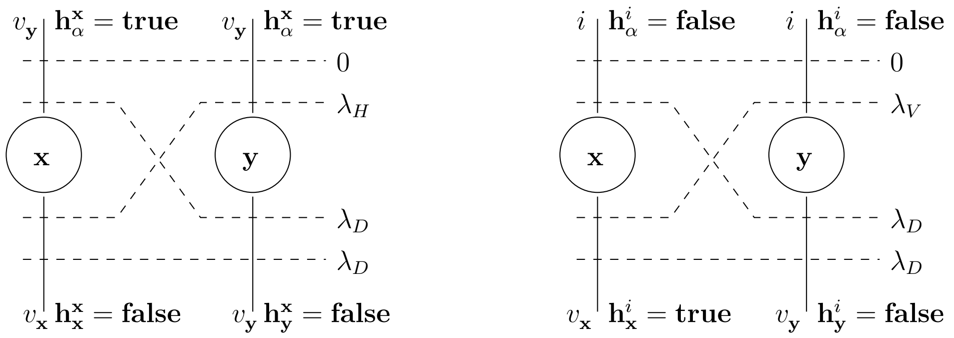
[3] We consider here that all motion models have different parameters.

case	$V(l_{\mathbf{x}},l_{\mathbf{y}}) \le V(l_{\mathbf{x}},l_\alpha)$ $+V(l_\alpha,l_{\mathbf{y}})$	obtained if	state
V1	$\lambda_H \le 0+0$	$\{v_{\mathbf{x}} \neq v_{\mathbf{y}}\} \wedge \{v_{\mathbf{x}} = v_\alpha = v_{\mathbf{y}}\}$	⇒ impossible
V2	$\lambda_D \le 0+0$	equiv. to previous case	⇒ impossible
V3	$\lambda_D \le \lambda_H + 0$	$\{v_{\mathbf{x}} \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_{\mathbf{x}}} = \mathbf{false}\}$ $\wedge\{v_{\mathbf{x}} \neq v_\alpha \wedge \mathbf{h}_\alpha^{v_{\mathbf{x}}} = \mathbf{true}\}$ $\wedge\{v_\alpha = v_{\mathbf{y}}\}$	⇒ possible !
V4	$\lambda_D \le 0 + \lambda_H$	$\{v_{\mathbf{x}} \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_{\mathbf{x}}} = \mathbf{false}\}$ $\wedge\{v_{\mathbf{x}} = v_\alpha\}$ $\wedge\{v_\alpha \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_\alpha} = \mathbf{true}\}$	⇒ impossible
V5	$\lambda_D \le \lambda_H + \lambda_H$	$\{v_{\mathbf{x}} \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_{\mathbf{x}}} = \mathbf{false}\}$ $\wedge\{v_{\mathbf{x}} \neq v_\alpha \wedge \mathbf{h}_\alpha^{v_{\mathbf{x}}} = \mathbf{true}\}$ $\wedge\{v_\alpha \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_\alpha} = \mathbf{true}\}$ $\wedge\{v_\alpha = v_{\mathbf{x}}\}$	⇒ impossible
V6	$\lambda_H \le \lambda_D + 0$	$\{v_{\mathbf{x}} \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_{\mathbf{x}}} = \mathbf{true}\}$ $\wedge\{v_{\mathbf{x}} \neq v_\alpha \wedge \mathbf{h}_\alpha^{v_{\mathbf{x}}} = \mathbf{false}\}$ $\wedge\{v_\alpha = v_{\mathbf{y}}\}$	⇒ possible !
V7	$\lambda_H \le 0 + \lambda_D$	$\{v_{\mathbf{x}} \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_{\mathbf{x}}} = \mathbf{true}\}$ $\wedge\{v_{\mathbf{x}} = v_\alpha\}$ $\wedge\{v_\alpha \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_\alpha} = \mathbf{false}\}$	⇒ impossible
V8	$\lambda_H \le \lambda_D + \lambda_D$	$\{v_{\mathbf{x}} \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_{\mathbf{x}}} = \mathbf{true}\}$ $\wedge\{v_{\mathbf{x}} \neq v_\alpha \wedge \mathbf{h}_\alpha^{v_{\mathbf{x}}} = \mathbf{false}\}$ $\wedge\{v_\alpha \neq v_{\mathbf{y}} \wedge \mathbf{h}_{\mathbf{y}}^{v_\alpha} = \mathbf{false}\}$ $\wedge\{v_\alpha = v_{\mathbf{x}}\}$	⇒ impossible

Table 1: Cases considered for the submodularity of $D()$.

case	$H^i(x,y) \leq H^i(x,\alpha)$ $+H^i(\alpha,y)$	obtained if,	state
H1	$\lambda_V \leq 0+0$	$\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\mathbf{y}}\} \wedge \{\mathbf{h}^i_{\mathbf{x}} = \mathbf{h}^i_{\alpha} = \mathbf{h}^i_{\mathbf{y}}\}$	⇒ impossible
H2	$\lambda_D \leq 0+0$	equiv. to previous case	⇒ impossible
H3	$\lambda_D \leq \lambda_V + 0$	$\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} \neq i\}$ $\wedge\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\alpha} \wedge v_{\alpha} = i\}$ $\wedge\{\mathbf{h}^i_{\alpha} = \mathbf{h}^i_{\mathbf{y}}\}$	⇒ possible !
H4	$\lambda_D \leq 0 + \lambda_V$	$\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} \neq i\}$ $\wedge\{\mathbf{h}^i_{\mathbf{x}} = \mathbf{h}^i_{\alpha}\}$ $\wedge\{\mathbf{h}^i_{\alpha} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} = i\}$	⇒ impossible
H5	$\lambda_D \leq \lambda_V + \lambda_V$	$\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} \neq i\}$ $\wedge\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\alpha} \wedge v_{\alpha} = i\}$ $\wedge\{\mathbf{h}^i_{\alpha} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} = i\}$ $\wedge\{\mathbf{h}^i_{\alpha} = \mathbf{h}^i_{\mathbf{x}}\}$	⇒ impossible
H6	$\lambda_V \leq \lambda_D + 0$	$\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} = i\}$ $\wedge\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\alpha} \wedge v_{\alpha} \neq i\}$ $\wedge\{\mathbf{h}^i_{\alpha} = \mathbf{h}^i_{\mathbf{y}}\}$	⇒ possible !
H7	$\lambda_V \leq 0 + \lambda_D$	$\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} = i\}$ $\wedge\{\mathbf{h}^i_{\mathbf{x}} = \mathbf{h}^i_{\alpha}\}$ $\wedge\{\mathbf{h}^i_{\alpha} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} \neq i\}$	⇒ impossible
H8	$\lambda_V \leq \lambda_D + \lambda_D$	$\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} = i\}$ $\wedge\{\mathbf{h}^i_{\mathbf{x}} \neq \mathbf{h}^i_{\alpha} \wedge v_{\alpha} \neq i\}$ $\wedge\{\mathbf{h}^i_{\alpha} \neq \mathbf{h}^i_{\mathbf{y}} \wedge v_{\mathbf{y}} \neq i\}$ $\wedge\{\mathbf{h}^i_{\alpha} = \mathbf{h}^i_{\mathbf{x}}\}$	⇒ impossible

Table 2: Cases considered for the submodularity of $H()$.


$v_{\mathbf{y}}$ $\mathbf{h}^{\mathbf{x}}_{\alpha} = \mathbf{true}$
$v_{\mathbf{y}}$ $\mathbf{h}^{\mathbf{x}}_{\alpha} = \mathbf{true}$
0
λ_H
x
y
λ_D
λ_D
$v_{\mathbf{x}}$ $\mathbf{h}^{\mathbf{x}}_{\mathbf{x}} = \mathbf{false}$
$v_{\mathbf{y}}$ $\mathbf{h}^{\mathbf{x}}_{\mathbf{y}} = \mathbf{false}$
i $\mathbf{h}^{i}_{\alpha} = \mathbf{false}$
i $\mathbf{h}^{i}_{\alpha} = \mathbf{false}$
0
λ_V
x
y
λ_D
λ_D
$v_{\mathbf{x}}$ $\mathbf{h}^{i}_{\mathbf{x}} = \mathbf{true}$
$v_{\mathbf{y}}$ $\mathbf{h}^{i}_{\mathbf{y}} = \mathbf{false}$

Case V3 Case H3

Figure 3: Cases V3 and H3 (with resp. $\mathbf{y} = \mathcal{T}_{v_{\mathbf{x}}}(\mathbf{x})$ and $\mathbf{y} = \mathcal{T}_i(\mathbf{x})$) . Both cases are *graph-representable* as the inequalities $\lambda_D \leq \lambda_D + \lambda_H$ for case V3 and $\lambda_D \leq \lambda_D + \lambda_V$ for case H3 are respected $\forall \lambda_D, \lambda_H$ and $\lambda_V \geq 0$ (see tables 1 and 2 for details) .

3.2 Minimization process

Even then, labeling cannot be achieved in reasonable time using a straightforward *alpha*-expansion since the number of possible labels $(v, \mathbf{h})$ increases dramatically with the number of layers: $(n+2)2^{n-1}$ possible expansions! However the problem could be circumvented limiting *alpha*-expansions to a sub-space of $\mathcal{L}$.

3.2.1 First minimization method

One can consider only a change of the visible layer and one hidden layer, i.e. $(v, \mathbf{h}^i)$-expansions for successive choices of i. Using this approach, we reduce the number of optimization steps to $2n^2$: for each visible layer j (so n iterations), we process $2n$ $(v_j, \mathbf{h}^i)$-expansions, testing in same time if the j-th layer is visible and if the i-th layer is hidden or not (with $i \neq j$).

However, some labelings are impossible to obtain. Consider the following example (figure 4): the optimal solution should be $v_{\mathbf{x}} = 0, v_{\mathbf{x}'} = 1, v_{\mathbf{x}'} = 2$ and $\mathbf{h}^0_{\mathbf{x}'} = \mathbf{h}^0_{\mathbf{x}''} = \mathbf{true}$ (all other hidden layers set to $false$). If we consider initial labelings such as $v_{\mathbf{x}} = 0, v_{\mathbf{x}'} = 1, v_{\mathbf{x}'} = 2$ but $\mathbf{h}^0_{\mathbf{x}'} = \mathbf{h}^0_{\mathbf{x}''} = \mathbf{false}$, there is any $(v_j, \mathbf{h}^0)$-expansion which could give the optimal solution. Indeed, neither the $(v_1, \mathbf{h}^0 = \mathbf{true})$-expansion, nor the $(v_2, \mathbf{h}^0 = \mathbf{true})$-expansion could change the labels of $\mathbf{x}'$ and $\mathbf{x}''$. Note that such a limitation is also encountered even if we change not only one hidden layer but also all the other ones at same time.

Only a change of hidden labeling without modifying any visible labeling could handle such case. Hence, we propose a second minimization process to solve the problem.

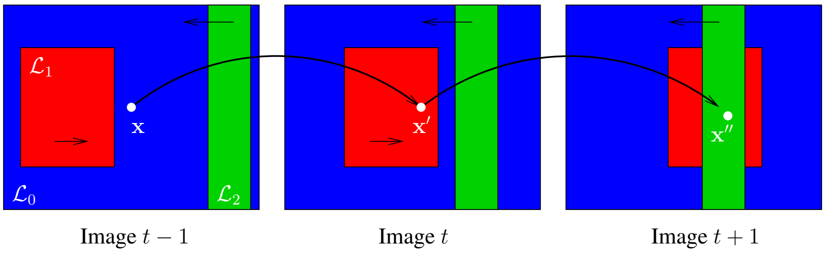


Figure 4: Example of sequence where optimal solution could not be obtained through $(v_j, \mathbf{h}^i)$-expansions. Here, there are three layers $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2$ (resp. in blue, red and green), the white pixel $\mathbf{x}$ and the projected ones $\mathbf{x}' = \mathcal{T}_0(\mathbf{x})$ and $\mathbf{x}'' = \mathcal{T}_0(\mathbf{x}')$.

3.2.2 Second minimization method

One can consider alternatively

- only a change of the visible layer without modifying the hidden layer states (except for the corresponding hidden layer $\mathbf{h}^v$ which is set to **false**)

  $\Rightarrow (v, \mathbf{h}^v = \mathbf{false})$-expansions *for successive choices of* v.

- and only a change of one hidden layer, without modifying visible layer

  $\Rightarrow (\mathbf{h}^i = \mathbf{false}/\mathbf{true})$-expansions *for successive choices of* i.

Using this approach, we reduce the number of optimization steps to $3n$: we process each visible layer (so n iterations) and $2n$ $(\mathbf{h}^i)$-expansions, testing if the i-th layer is hidden or not. This yields in practice to acceptable minimization times, without modifying noticeably the segmentation.

But such an approach has also some drawbacks: some labelings can be unreachable. For example, if a pixel $\mathbf{x}$ is currently labeled as $(v_\mathbf{x} = 1, \mathbf{h}^0_\mathbf{x} = \mathbf{h}^1_\mathbf{x} = \mathbf{false})$ and if the optimal label is $(v_\mathbf{x} = 0, \mathbf{h}^1_\mathbf{x} = \mathbf{true})$, it is not yet guaranteed that a $(v = 0)$-expansion will decrease the overall energy, changing the label of $\mathbf{x}$ to $(v_\mathbf{x} = 0, \mathbf{h}^1_\mathbf{x} = \mathbf{false})$.

The corresponding graph is a three-dimensional one, the third dimension being time. The data and spatial regularization terms of the energy are standard in the graph-cut framework. During a $v-$ or $\mathbf{h}^i-$expansion, the *backward* and *forward* spatial constraints yield links between each pixel $\mathbf{x}$ at time t and $2(2+n)$ other pixels: $\mathbf{x}_v$ or $\mathbf{x}_\mathbf{h}$, $\mathbf{x}_{v_\mathbf{x}}$ and $\mathbf{x}_{\mathbf{h}^i_\mathbf{x}}$ $(i \in [1, n])$ at time $t+1$ and similarly at time $t-1$ (see figure 5).

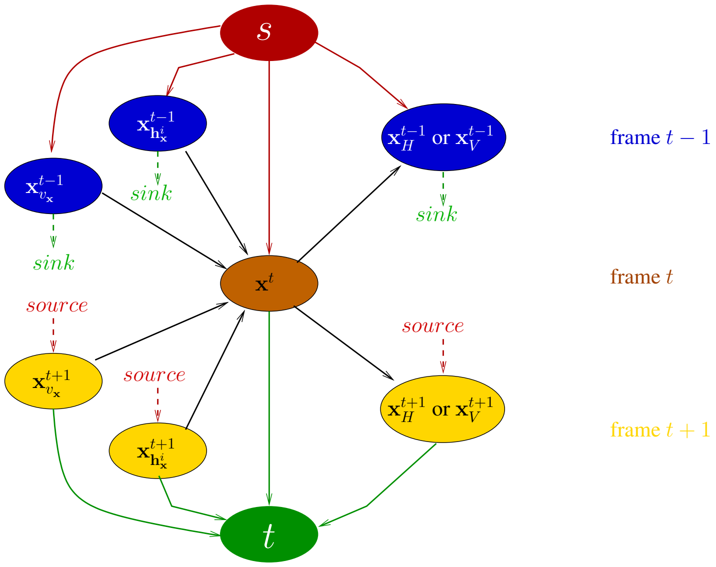

s
$\mathbf{x}^{t-1}_{\mathbf{h}^i_\mathbf{x}}$
$\mathbf{x}^{t-1}_H$ or $\mathbf{x}^{t-1}_V$
frame $t-1$
$\mathbf{x}^{t-1}_{v_\mathbf{x}}$
$sink$
$sink$
$sink$
$\mathbf{x}^t$
frame t
$source$
$source$
$source$
$\mathbf{x}^{t+1}_{v_\mathbf{x}}$
$\mathbf{x}^{t+1}_H$ or $\mathbf{x}^{t+1}_V$
frame $t+1$
$\mathbf{x}^{t+1}_{\mathbf{h}^i_\mathbf{x}}$
t

Figure 5: Graph construction: Source t-links are shown in red, sink t-links in green. Considering a $(V, \mathbf{h}^V = \mathbf{false})$-expansion (or $\mathbf{h}^H$-expansion), temporal n-links are shown in black and link the pixel $\mathbf{x}$ (frame t) to pixels $\mathbf{x}_{v_\mathbf{x}}$,$\mathbf{x}_{\mathbf{h}^i_\mathbf{x}}$,$\mathbf{x}_V$ (or $\mathbf{x}_H$) of frames $t-1$ and $t+1$. *Note*: for clarity, only the links relative to the i-th hidden layer are shown.

4 Results

4.1 Synthetic sequence

Figure 6 shows the results obtained on a synthetic sequence ($n = 3$). Throughout the sequence, the proportion of misclassified visible pixels is 0.06% and the proportion of pixels where the complete label l (visible *and* hidden parts) is incorrect is also 0.06%: for each pixel, classification fails or succeeds globally. Note that in this particular sequence, no pixel is classified as undefined. Indeed, only noise or aliasing could generate such pixels. Because hidden parts are modelized, the undefined label do not account anymore for points that become hidden like in [15].

4.2 Real sequences

As a first step[4] toward comparing our results to state of the art methods like [6, 15], we show the results obtained for a real sequence (fig. 7). One can see that the segmentation of the visible layers is comparable to the usually obtained segmentation. Note that the wheels of the car are sometime classified as undefined because the

[4] No ground truth is provided here!

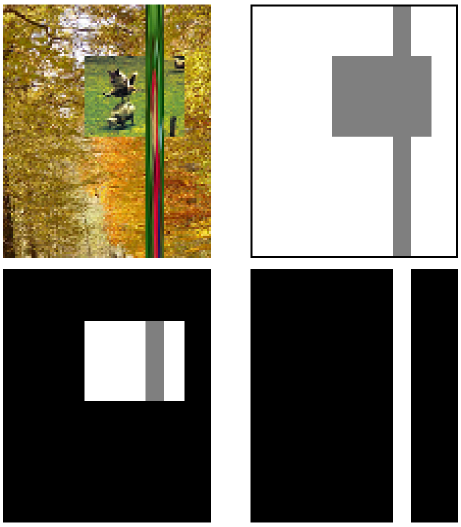


Figure 6: A synthetic sequence. From top to bottom, left to right: original sequence, layers 1, 2 and 3 (white=visible, grey=hidden) (Note: on this particular image of the sequence, no pixel is classified as undefined)

number n of layers is fixed too small (the wheels have their own motion). A splitting/merging approach could be used to choose n dynamically. We are in the process of implementing this.

Moreover, our goal was to extract the hidden parts of the layers and this is correctly done. Continuous labeling between frames is obtained, providing non-disrupted segmentation throughout the sequences. Again, note that the number of undefined pixels is rather small: unlike in [15] where these pixels code also for points that are going to be hidden, in our method $v_{\mathbf{x}} = \varnothing_V$ only stands for a lack of image information (e.g. too much noise).

5 Conclusion and discussion

We have presented a novel global optimization process for motion layer segmentation in a video sequence. Considering the hidden parts of the layers, we achieve a continuous labeling, even is case of occlusion: when hidden, a point is tracked until reappearance. Ongoing work includes dealing with (i) processing longer sequences through shifting windows, (ii) more robustness thanks to multi-scale analysis in time and (iii) coping with a robust determination of a variable number of layers.

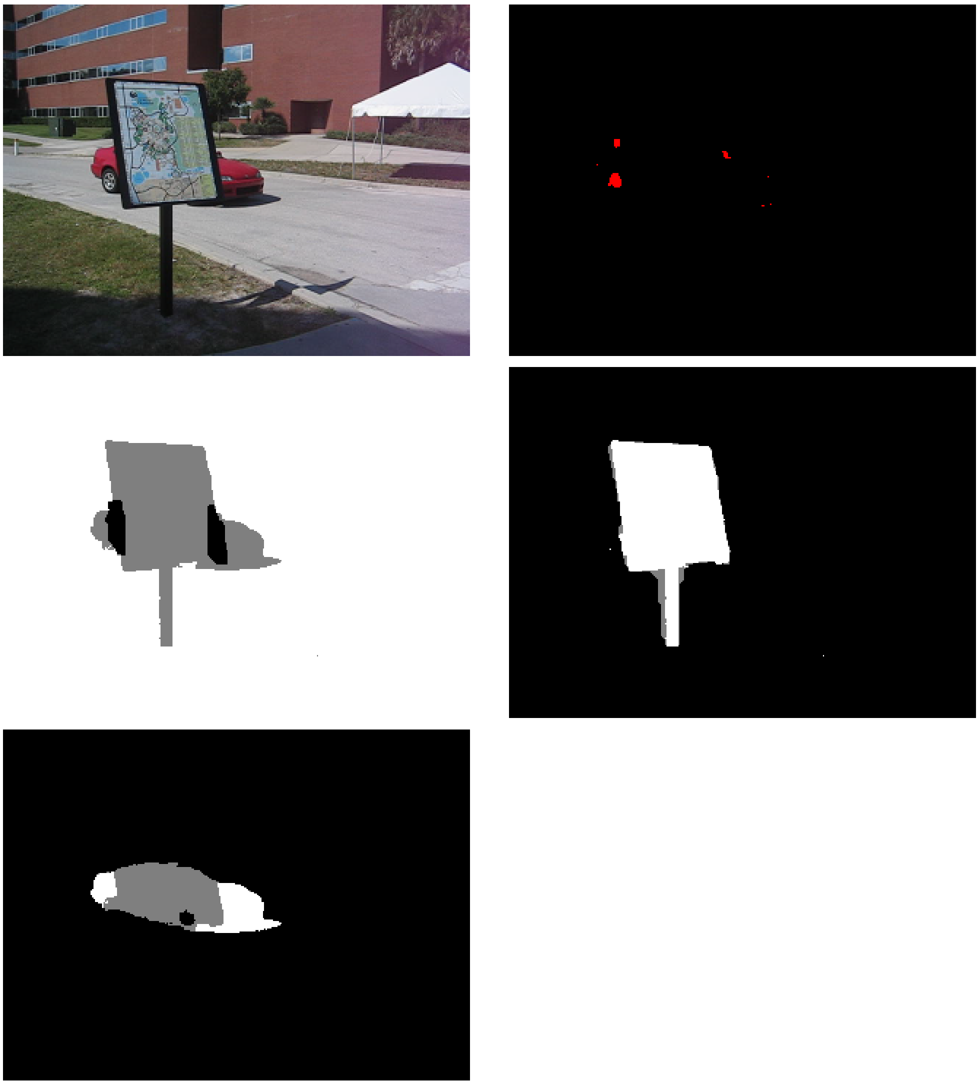


Figure 7: Carmap sequence. From top to bottom, left to right: original sequence, undefined pixels (in red), layers 1, 2 and 3 (white=visible, grey=hidden).

References

[1] S. Ayer and H. S. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. In *ICCV*, page 777, 1995.

[2] Michael J. Black and P. Anandan. The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. *Comput. Vis. Image Underst.*, 63(1):75–104, 1996.

[3] Michael J. Black and Allan D. Jepson. Estimating optical flow in segmented images using variable-order parametric models with local deformations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(10):972–986, 1996.

[4] Patrick Bouthemy and Edouard Francois. Motion segmentation and qualitative dynamic scene analysis from an image sequence. *Int. J. Comput. Vision*, 10(2):157–182, 1993.

[5] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(11):1222–1239, 2001.

[6] Romain Dupont, Nikos Paragios, Renaud Keriven, and Phillipe Fuchs. Extraction of layers of similar motion through combinatorial techniques. In *EMMCVPR*, 2005.

[7] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE. Trans. on PAMI*, 1984.

[8] Qifa Ke and Takeo Kanade. A robust subspace approach to layer extraction. In *MOTION '02: Proceedings of the Workshop on Motion and Video Computing*, page 37, 2002.

[9] Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? In *ECCV*, pages 65–81, 2002.

[10] M. P. Kumar, P. H. S. Torr, and A. Zisserman. Learning layered motion segmentations of video. In *ICCV*, 2005.

[11] Jean-Marc Odobez and Patrick Bouthemy. Direct incremental model-based image motion segmentation for video analysis. *Signal Processing*, 66(2):143–155, 1998.

[12] Allan D. Jepson Shanon X. Ju, Michael J. Black. Skin and bones: Multi-layer, locally affine, optical flow and regularization with transparency. In *CVPR*, page 307, 1996.

[13] J.Y.A. Wang and E.H. Adelson. Layered representation for motion analysis. In *CVPR93*, pages 361–366, 1993.

[14] Yair Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *CVPR*, 1997.

[15] Jiangjian Xiao and Mubarak Shah. Accurate motion layer segmentation and matting. In *CVPR*, pages 698–703, 2005.