Segmentation Framework Based on Label Field Fusion

Pierre-Marc Jodoin, Max Mignotte, and Christophe Rosenberger

***Abstract*—In this paper, we put forward a novel fusion framework that mixes together *label fields* instead of observation data as is usually the case. Our framework takes as input two label fields: a quickly estimated and to-be-refined segmentation map and a spatial region map that exhibits the shape of the main objects of the scene. These two label fields are fused together with a global energy function that is minimized with a deterministic iterative conditional mode algorithm. As explained in the paper, the energy function may implement a pure fusion strategy or a fusion-reaction function. In the latter case, a data-related term is used to make the optimization problem well posed. We believe that the conceptual simplicity, the small number of parameters, the use of a simple and fast deterministic optimizer that admits a natural implementation on a parallel architecture are among the main advantages of our approach. Our fusion framework is adapted to various computer vision applications among which are motion segmentation, motion estimation and occlusion detection.**

***Index Terms*—Color segmentation, label fusion, motion estimation, motion segmentation, occlusion.**

I. Introduction

In the field of imagery, computer vision is frequently considered as a research area in which applications aim at estimating high-level models learned from input images. This is the case for applications such as stereovision [1], motion estimation [2], and motion detection whose goal is to estimate depth labels, optical flow vectors and the presence (or absence) of movement in a video sequence. Most methods used to solve these kinds of imagery problems are built upon a to-be-optimized energy function made up of low-level image features such as color, spatial gradient or texture features. Years of research have demonstrated that significant improvements may be achieved by using more complex features (e.g., wavelets coefficients instead of Fourier coefficients), better designed energy models (e.g., robust instead of quadratic or L1-norm energy functions) and better optimizers (stochastic instead of deterministic optimizers and/or a multiresolution instead of monoresolution optimization schemes).

With the ever-growing computational power of modern computers, researchers tend to use an increased number of features to enforce the result accuracy. The main advantage of using many features resides in the fact that many features blended together often mutually compensate for their respective limitations. For example, when segmenting cluttered color images, the use of color *and* texture features has shown great improvement as compared to color-only or texture-only segmentation approaches [3].

However, using numerous features raises the question of how these metrics can be fused together. The most intuitive and simple solution is probably to fuse those features inside one single N-dimensional vector. In this perspective, each input pixel may be associated not only to 3-D RGB color values, but also to texture values, gradient values, edge data, or any other input features. Among the applications that benefit from this approach of "all features in one vector" are motion segmentation [4], texture segmentation [5]–[8], image retrieval [9], [10], and face recognition [11]–[13], to name a few.

However, despite the obvious advantages of using high-dimensional data vectors, an increased number of features raises new challenges. One such challenge is the well known problem of the "curse of dimensionality" [14], [15] related to the rapid increase of extra dimensions. Donoho [16] points out that if we consider a unit dimension divided in bins of size 1/10, a minimum of ten points is needed to fill each bin with at least 1 point. However, for a 20-dimension unit hypercube, no less than 10^{20} points are needed to fill the bins. This means that an increase in dimension often means a need for more input data which, for some applications, is not realistic.

The typical solution used to avoid the curse of dimensionality is to reduce the number of dimensions. To do so, one may try to identify the "right" features in the stream of input data and minimize the dimensionality by getting rid of the "less useful" features [5], [14], [17], [18]. One simple but efficient way for selecting features is to retain a subset of X features that best help the algorithm produce precise results [14]. Although this approach is viable in many applications, it has the disadvantage of requiring a training data set and, thus, being deficient for unsupervised applications. Other approaches for reducing dimensionality focus more on the "right" data space dimension than on the "right" feature space. This is the case for methods such as principal component analysis (PCA) [5], [6], [8], [11], [14], [15], singular value decomposition (SVD) [9], and the Fisher linear discriminant (FLD) [15], [17] which projects linearly the input data onto a lower dimensional subspace. In cases where features have complicated interactions, a nonlinear component

Manuscript received October 3, 2006; revised June 1, 2007. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ercan E. Kuruoglu.

P.-M. Jodoin is with the Département d'informatique, Université de Sherbrooke, Sherbrooke QC J1K 2R1, Canada (e-mail: pierre-marc.jodoin@usherbrooke.ca).

M. Mignotte is with the Département d'informatique et de recherche opérationnelle, Université de Montréal, Montréal QC H2L 2W5, Canada (e-mail: mignotte@iro.umontreal.ca).

C. Rosenberger is with the Laborotoire Vision and Robotics, Université d'Orléans, Bourges 18020, France (e-mail:christophe.rosenberger@ensi-bourges.fr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2007.903841

analysis may also be considered [15]. Another way to deal with high-dimensional data is to give more *weight* to some features and less to others. This strategy has been used by Pichler *et al.* [6], [7] with their feature contrast method and indirectly by many authors who implement fuzzy logic-based fusion procedures [19]–[22]. In a similar perspective, some researchers have underlined the fact that data gathered form many sensors can be fused together with the help of Dempster–Shafer theory of evidence [23], [24]. Also, feature weighting is what back-propagation training algorithms (neural networks) are meant to do [14], [15]. However, since a training data set is needed to weigh the input features, this approach is not suited to all applications.

Another problem that often occurs when fusing different features concerns the need for *axis rescaling*. This situation occurs when features with different units are blended together, making the N-dimensional space anisotropic. This typically happens when blending, say, RGB values ranging between (0,0,0) and (256,256,256) with motion vectors ranging between (-5.0, -5.0) and (5.0,5.0). In this case, the usual similarity measures (used to evaluate the distance between data points) will give an overwhelming importance to features having a larger unit range such as the RGB values. Although data normalization [6], [15] may be used to rescale the axes, for some applications rescaling may have the effect of reducing the class separability [15].

To alleviate dimensionality problems, other methods use one large energy function [4], [25]–[29] made of a series of smaller energy functions built around one (or sometimes two) specific features. A typical example is the method used by Heitz and Bouthemy [28] in which a large Markovian energy function composed of five gradient-based, edge-based and motion-based energy functions is minimized. This kind of approach has the advantage of using multiple features without having to deal explicitly with the inherent problems related to multiple dimensions. It also allows intuitive and yet *ad hoc* energy function formulation. However, despite the undeniable advantages of using large energy functions, these applications often contain many parameters to tweak. Furthermore, large and complex energy functions are more likely to have an erratic profile with several local *minima* that are not trivial to minimize, especially with a deterministic optimizer.

Another family of fusion approaches used in imagery are the so-called *decision fusion* approaches [30]–[32]. With these methods, a series of energy functions are first minimized before their outputs (their decisions) are merged. In this case, the energy functions are defined on different basic features and/or different cost functions. With this perspective, Reed *et al.* [33] successfully implemented a similar fusion method to segment sonar images. Their method fuses segmentation maps involving identical classes (here, segmentation maps of the seafloor in sonar imagery) with a voting scheme followed by a Markov random field (MRF) in-painting procedure.

The last data fusion trend that we mention involves the so-called region-based approaches that are typically used in stereovision [1], [26], [34]–[36], motion segmentation [37]–[39], motion estimation [2], [40], and image deconvolution [41]. Region-based methods generally use a region map R initially obtained after segmenting an input image into regions of uniform color. Under the assumption that these regions contain precise information on the main objects of the scene, the regions are used to help regularize the optimization process. Although some of these methods implement a *soft* region-based constraint [26], an imprecise region map R often generates errors that are difficult if not impossible to compensate for.

In this paper, we propose a new fusion framework that mixes together label fields instead of features; label fields containing different and yet complementary information. We will show that a MRF framework can be efficiently used to fuse, in a versatile way, the knowledge of these two preliminary label maps through a data-related term and a regularizing prior. More specifically, our framework takes as input two label fields: a region map (called r) obtained after segmenting one (or two) input images into regions of uniform color, and a rough estimate (called $x^{[0]}$) of the application label field. Note that $x^{[0]}$ is application specific and may contain occlusion labels, motion labels, or any other high-level information. Once r and $x^{[0]}$ have been estimated, they are merged together with a fusion procedure expressed as an energy function minimization. Here, the optimization process searches for a new label field $\hat{x}_{opt}$ whose content is close to that of $x^{[0]}$ but adapted to fit the regions of r. As is the case for most conventional region-based methods, r is assumed to contain precise information on the overall shape of the scene. However, by the very nature of our fusion procedure, our framework is tolerant to imprecisions in r and reacts smoothly to any modification of its parameters. Let us mention that the to-be-minimized energy function U may be a pure *fusion* function or a *a fusion-reaction* function including a data-related term. In both cases, our framework can be implemented on a parallel architecture such as a graphics processor unit (GPU).

To our knowledge, fusion of label fields involving labels of different natures (i.e., classes estimated with different image features), has never been proposed in the literature before, and/or developed (up to now) into a coherent theoretical framework.

The rest of the paper is organized as follows. In Section II, our framework is first introduced and summarized with an algorithm. Then, Section III illustrates three applications (namely, motion segmentation, motion estimation, and occlusion detection) that can benefit from our framework. This section also includes experimental results. Further experimental results are presented in Section IV while Section V draws conclusions.

II. Label Field Fusion

A. Framework

As mentioned previously, our method fuses together two label fields, both estimated with different low-level image features [42], [43]. The reason for blending label fields is to alleviate dimensionality problems and, more specifically, the rescaling problem that arises when blending features with different units such as color and motion vectors for example. Furthermore, since the two label fields are estimated separately, they allow the minimization of simple energy functions (i.e., functions with few local *minima*) defined on one or two features.

The first label field considered is a region map $r = \{r_s | s \in S\}$ defined on a rectangular lattice S made up of $\mathcal{N} \times \mathcal{M}$ sites. The map r is obtained by segmenting one (or two) input images

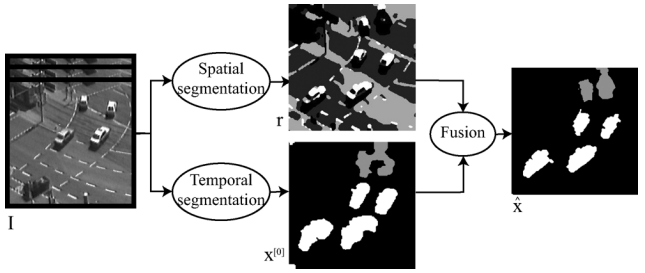

Fig. 1. Schematic view of our fusion framework adapted to motion segmentation. In this example, the fusion of the region map r and the motion segmentation field $x^{[0]}$ makes the resulting label field $\hat{x}$ more precise and less blobby.

into uniform regions. Here, "uniformity" may be defined in the sense of color, texture or any other image features that best suit the image content. For the purpose of this paper, r is estimated based on the color feature. Every element r_s in r takes a value in $\Lambda = \{\omega_0, \omega_1, \ldots, \omega_{c-1}\}$ where "c" is the number of color classes in which the input image is segmented. In this way, every input pixel associated with a given class $r_s = \omega_i$ has a color with distribution $P(\text{color}|\omega_i)$.

The second input label field for our framework is the so-called *application label field*, i.e., a field made of application-specific labels. For instance, this label field (which we call $x^{[0]}$) may contain motion labels, optical flow labels, occlusion labels or any label specific to the current application. As is the case for r, $x^{[0]}$ is defined on a rectangular lattice of size $\mathcal{N} \times \mathcal{M}$ whose sites take a value in $\Gamma = \{\zeta_0, \zeta_1, \ldots, \zeta_{d-1}\}$ where d is the number of categories. For instance, for a typical occlusion detection application, $d = 2$ and $\zeta_0 = \{\text{No\_Occlusion}\}$ and $\zeta_1 = \{\text{Occlusion}\}$. Here, the occlusion concept related to stereovision or video applications and refers to pixels that are visible in one image but occluded in a second image (c.f. Fig. 12).

In our framework, $x^{[0]}$ is a rough estimate of the *true* label field that we wish to estimate. Thus, $x^{[0]}$ is typically obtained with a simple method and may be imprecise near edges, and contain false positives and false negatives due to noise or lack of texture. In fact, $x^{[0]}$ serves as an initialization for the ensuing iterative fusion process (hence the "$_{[0]}$" exponent).

Once r and $x^{[0]}$ have been estimated, they are blended together with a fusion procedure (see Fig. 1 for a schematic view). The goal of this fusion procedure is to modify the content of $x^{[0]}$ based on the regions of r which describe the overall shape of the predominant objects of the scene. Here, by the very nature of r and $x^{[0]}$, it is reasonable to assume that the regions of the to-be-estimated label field $\hat{x}$ follow the ones in r. In other words, we assume that no transition in $\hat{x}$ occurs inside a uniform region of r and, thus, that the edges in $\hat{x}$ correspond to edges in r.

B. Fusion Procedure

In the light of the assumption described at the end of the previous section, the validity of a solution $\hat{x}$ may be evaluated by measuring how well the regions in $\hat{x}$ follow the ones in r. Locally, this means that a uniform section in r will also be uniform in $\hat{x}$ and that an edge in $\hat{x}$ should also correspond to an edge in r. In other words, no transition in $\hat{x}$ is expected to occur inside a uniform region of r.

In our model, we assume that r and $\hat{x}$ are realizations of a pair of joint Markov random fields (MRFs) and that, by the properties of the Hammersley–Clifford theorem [44], the joint probability $P(r_s, \hat{x}_s|\Psi_s)$ is a Gibbs distribution, such as

$$P(r_s, \hat{x}_s|\Psi_s) = \frac{1}{Z} \exp\left(-E_s(r, \hat{x}|\Psi_s)\right) \tag{1}$$

where Z is a normalization factor, $E_s(r, \hat{x}|\Psi_s)$ is a local energy function measuring how well $\hat{x}$ and r fit together around site s, and Ψ_s is a $L \times L$ local joint neighborhood surrounding site s. In the absence of any prior geometric knowledge on $\hat{x}$ and r, we define the potential function E_s as

$$E_s(r, \hat{x}|\Psi_s) = -\sum_{t \in \Psi_s} \delta(r_t, r_s)\delta(\hat{x}_t, \hat{x}_s) \tag{2}$$

where $\delta(a, b)$ is the Kronecker delta function ($\delta(a, b) = 1$ if $a = b$ and 0 otherwise). Note that this potential function to some extent resemble the Potts model, i.e., an N-class generalization of the well-known Ising model [45]. Equation (2) is what we call the *fusion model* that, by its very nature, measures the spatial homogeneity of the joint couple of Markovian random fields x and r. More specifically, E_s is small when the regions in $\hat{x}$ fit locally the regions in r, i.e., when no edges in $\hat{x}$ cross a uniform region in r. Also, E_s can be seen as a function that counts the number of neighbors "t" around site "s" whose label "r_t" and "x_t" is the same as "r_s" and "x_s." In this way, the best label field $\hat{x}$ may be expressed as

$$\hat{x} = \arg\min_x \sum_{s \in S} E_s(r, x|\Psi_s). \tag{3}$$

Although this formulation can produce very decent results [$\hat{x}$ in Figs. 1–3 has been computed with (3)] it nonetheless contains a weakness. In fact, E_s allows more than one global *minimum* that may correspond to trivial yet uninteresting solutions such as the constant label field $\hat{x}_s = \zeta_0, \forall s \in S$. Of course, most of the time when (3) is minimized with a deterministic downhill search algorithm, the estimated solution $\hat{x}$ lies in a *minima* close to the initial estimate $x^{[0]}$. However, when minimizing (3) with a stochastic optimizer such as simulated annealing, the resulting solution $\hat{x}$ may be quite far from $x^{[0]}$ and not useful in practice. To overcome this problem, a *reaction* term is added to E_s to make sure the problem is well-posed and that there exists only one global solution. Mathematically, this is formulated as

$$E'_s\left(r, \hat{x}, x^{[0]}|\Psi_s\right) = -\alpha\delta\left(x_s^{[0]}, \hat{x}_s\right) - \sum_{t \in \Psi_s} \delta(r_t, r_s)\delta(\hat{x}_t, \hat{x}_s) \tag{4}$$

where $\delta(x_s^{[0]}, \hat{x}_s)$ can be viewed as a *reaction* term and α is a constant. In this case, the best label field $\hat{x}$ is given by

$$\hat{x} = \arg\min_x \sum_{s \in S} E'_s\left(r, x, x^{[0]}|\Psi_s\right). \tag{5}$$

Since neither (5) nor (3) have an analytical solution, we use Besag's iterative conditional mode (ICM) optimization algorithm [44] to estimate $\hat{x}$ as shown in Algorithm 1. ICM is a deterministic update optimization algorithm introduced by Besag [44] to optimize the energy function of a Gibbs distribution.

More precisely, it consists in finding the conditional modes, i.e., for each site, the value that maximizes the local conditional probability density function (pdf). This deterministic algorithm is not guaranteed to find the global *minima*; nevertheless, it drastically reduces computational time as compared to stochastic relaxation techniques such as simulated annealing.

Algorithm 1 Fusion-Reaction Algorithm

Require: I Input image

Ensure: $x^{[k]}$ Label field after the k^{th} iteration

Initialization

1: Estimate $x^{[0]}$ based on features picked in I

2: $r \leftarrow$ spatial segmentation of I into "c" classes

3: $k \leftarrow 0$

ICM Optimization (Fusion).

4: **repeat**

5: $k \leftarrow k+1$

6: **for** each site $s \in S$ **do**

7: **for** each class $\zeta_i \in \Gamma$ **do**

8: $tab[\zeta_i] \leftarrow -\alpha\delta(\zeta_i, x^{[0]})$

9: $-\sum_{t\in\Psi_s}\delta(\zeta_i, x_t^{[k]})\delta(r_s, r_t)$

10: **end for**

11: $x_s^{[k]} \leftarrow \arg\min_{\zeta_i\in\Gamma} tab[\zeta_i]$

12: **end for**

13: **until** $x^{[k-1]} \neq x^{[k]}$

14: **return** $x^{[k]}$

The way our fusion procedure works is illustrated in Fig. 2 ($\alpha = 0$ in this example). In image r, site γ is part of the black class (which is a section of the moving vehicle) but has the *static* label in $x^{[0]}$. When considering the sites that are *part of the black section of the vehicle in* r inside Ψ_γ, we see that a majority of those sites have a *mobile* label in $x^{[0]}$. In other words, within the nearest neighbors around site γ with a black label in r, there is a majority of *mobile sites*. For this reason, after minimizing the energy function E', the site γ is assigned a *mobile* label in $\hat{x}$. Note that, since $\alpha = 0$ in this example, each ICM iteration of our fusion method works in a similar way to the well-known K nearest neighbor algorithm does.

C. Markovian Segmentation

As mentioned previously, r and $x^{[0]}$ are label fields estimated with different image features. Since $x^{[0]}$ contains application-specific labels, we will see in Section III how it can be estimated in the context of three specific applications. As for r, although any valid segmentation algorithm may be used to estimate it, we use a Markovian approach that we shall describe in the following paragraphs. The reason for this choice is twofold. First, the segmentation method we have implemented is *unsupervised* and, thus, requires no human intervention at runtime. Second, this segmentation method can be parallelized and implemented on a parallel architecture such as a graphics processor unit (GPU) [46]. With such an implementation, r can be computed in interactive time.

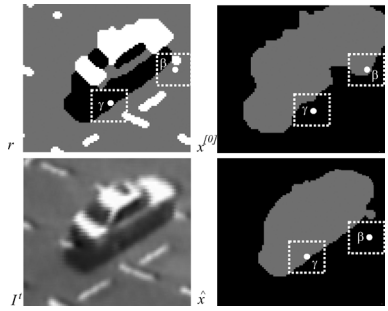

Fig. 2. Zoom on KARLSRUHE sequence. Top left is the label field r and top right is motion label field $x^{[0]}$. In this example, the motion label field $x^{[0]}$ contains two classes which can be understood as the "static" and the "moving upward" classes. Bottom left is the image frame at time t while bottom right shows the motion label field at convergence (here, $\alpha = 0$). Note how the region in $\hat{x}$ is well localized as compared to the one in $x^{[0]}$.

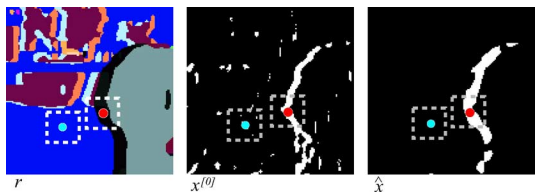

Fig. 3. Zoom on the TSUKUBA scene. After fusing r (the region map) and $x^{[0]}$ (an occlusion map), the number of isolated false positives and false negatives in $x^{[0]}$ has significantly reduced because the region map r is locally homogeneous.

Let us consider $Z = \{R, I\}$, a pair of random fields where $R = \{R_s, s \in S\}$ and $I = \{\vec{I}_s, s \in S\}$ represent, respectively, the color label field and the observation field, both defined on a $\mathcal{N} \times \mathcal{M}$ lattice $S = \{s = (i,j) | i \in [0,\mathcal{N}[, j \in [0,\mathcal{M}[\}$. Here, I is a known input image and r (a realization of R) is to be estimated. As mentioned previously, r_s takes a value in $\Lambda = \{\omega_0, \ldots, \omega_{c-1}\}$, where c is the number of color classes. Note that $\vec{I}_s$ is a 3-D vector for color images and a scalar for grayscale images.

Segmentation can be viewed as a statistical labeling problem, i.e., a problem for which each observation vector $\vec{I}_s$ most be associated with the *best* color class $r_s = \omega_i \in \Lambda$. Thus, inferring a label field can be seen as an optimization problem that searches for *the best* r in the sense of a given statistical criterion. Among the available statistical criteria, the maximum *a posteriori* (MAP) criterion states that a label field r is optimal according to I when it maximizes the *a posteriori* pdf: $P(r|I)$. In this way, r is optimal whenever $r = \arg\max_{r'} P(r'|I)$ [44].

Because $P(r|I)$ is often complex or undefined, it is common to assume that r and I are realizations of MRFs and that, by the Hammersley–Clifford theorem [44], the posterior distribution is defined by a Gibbs distribution of the form $P(r|I) \propto \exp -U(r,I)$ where $U(r,I)$ is an *energy* function [44]. By the

properties of the Bayes theorem [14], the *a posteriori* distribution can be represented as

$$P(r, I) \propto \exp\{-(U_1(r, I) + U_2(r))\} \tag{6}$$

where U_1 and U_2 are the likelihood and prior energy functions. By assuming independence between the random variables $\vec{I}_s$ (i.e., $P(I|r) = \prod_{s\in S} P(\vec{I}_s|r_s)$), the corresponding posterior energy to be minimized is

$$U(r, I) = \sum_{s\in S} \left(\underbrace{L_s(r_s, \vec{I}_s)}_{U_1(r_s, I_s)} + \beta \underbrace{\sum_{\langle s,t\rangle} [1 - \delta(r_s, r_t)]}_{U_2(r_s)} \right)$$

where U_2 corresponds to the isotropic Potts model. Here, $\delta(a, b)$ is the Kronecker function, β is a constant, $\langle s, t\rangle$ is a set of binary *cliques*, and $L_s(r_s, \vec{I}_s) = -\ln P(\vec{I}_s|r_s)$. Note that the cliques defined here are on a second-order neighborhood.

The conditional distribution $P(\vec{I}_s|r_s)$ models the distribution of the observed data $\vec{I}_s$ given a class $r_s \in \Lambda$. In this paper, this distribution is modeled with a Normal law that depends on the two parameters $(\vec{\mu}_{r_s}, \Sigma_{r_s})$. Since there are c different classes, there are c different Normal laws and a total of $2c$ Gaussian parameters $\Phi = [(\vec{\mu}_0, \Sigma_0), \ldots, (\vec{\mu}_{c-1}, \Sigma_{c-1})]$. Because these parameters are initially unknown, they need to be estimated. To this end, we use a Markovian and stochastic method called Iterated conditional estimation (ICE) [47]. Note that K-means or the EM algorithms could have also been used.

Once Φ has been estimated with ICE, r can be obtained by minimizing the global energy function U

$$r = \arg\min_{r'} U(r', I). \tag{7}$$

To do so, we again use Besag's ICM algorithm [44]. For more details on our implementation of ICM and ICE, please refer to [46].

III. Computer Vision Applications

In this section, three computer vision applications are described and adapted to our fusion framework. These applications are motion segmentation, motion estimation/segmentation, and occlusion detection. As the name suggests, motion segmentation is a procedure that groups together pixels having a uniform displacement in a video sequence. As for motion estimation, it refers to the task of estimating the optical flow visible in a video sequence. Since our motion estimation procedure estimates a parametric flow together with a motion label field, we call this operation *motion estimation/segmentation*. Finally, occlusion detection is a procedure that takes as input two images and that locates the pixels that are visible in one image but occluded in the second image. Consequently, occlusion detection is closely related to optical flow and stereovision.

In order to gauge performance of our algorithm, we used sequences representing different challenges. Some sequences are real while others have been computer generated and come with a perfect ground-truth label field g. The results presented in this Section illustrate how stable and robust our algorithm is with respect to the window size $\Psi = L \times L$, the α coefficient (4) and the precision of the region map r. Note that for these results, the region maps r have been computed with a number of classes ranging between 4 and 7 and that the window size Ψ ranges between 5×5 and 11×11. Also, the number of iterations $[k]$ needed by the fusion procedure to converge (algorithm 1) depends on the nature of the scene and the application. For the motion segmentation and the motion estimation applications, an average of 30 iterations is needed whereas an average of five iterations is needed for the occlusion detection.

A. Motion Segmentation

1) Introduction: Motion segmentation refers to the general task of labeling pixels with uniform displacement [48], [49]. Consequently, motion segmentation has often been linked to motion estimation. Actually, a common way to segment an image sequence is to estimate an optical flow field and then segment it into a set of regions with uniform motion vectors. Such an approach is sometimes called *motion-based* [48] since segmentation is performed on the basis of displacement vectors only. This kind of segmentation is rather easy to implement and generates more accurate results than, say, an 8×8 block segmentation procedure.

To enforce precision, some authors propose segmentation models based on additional features, such as brightness and edges. These models are sometimes referred to as *spatio-temporal* segmentation techniques. In this context, Black [25] presented an MRF approach that minimizes a three-term energy function using a stochastic relaxation technique. In Black's work, the motion label field is estimated on the basis of motion *and* intensity. In [39], Altunbasak *et al.* proposes a region-based motion segmentation approach. Assuming that color regions are more accurate than the motion regions, a region-based motion segmentation is performed, whereby all sites contained in a color region are assigned the same motion label. In a similar vein, Bergen and Meyer [50] show that an image segmentation may be used to eliminate error due to occlusion in an animated scene. For completeness, let us also mention the work by Khan and Shah [4] in which a MAP framework is proposed to softly blend color, position and motion cues to extract motion layers. In this contribution, each cue has its own pdf. These pdfs are combined together with feature weights that give more or less importance to a cue depending on certain specified observations.

2) Motion Segmentation and Our Framework: As mentioned in Section II, our framework is supplied with two label fields: r, a region map and $x^{[0]}$ a motion map. Although $x^{[0]}$ could be obtained with any valid motion segmentation approach, we decided to use the same unsupervised statistical Markovian procedure that we use to compute r. In this way, $x^{[0]}$ is obtained by segmenting V, a vector field computed with an iterative and multiresolution version [51] of the well-known Lukas–Kanade algorithm [52], [53]. Note that, for this segmentation, the vector field V stands for the observation field that we called I in Section II-C and that every element $\vec{V}_s$ is a 2-D real vector. For every

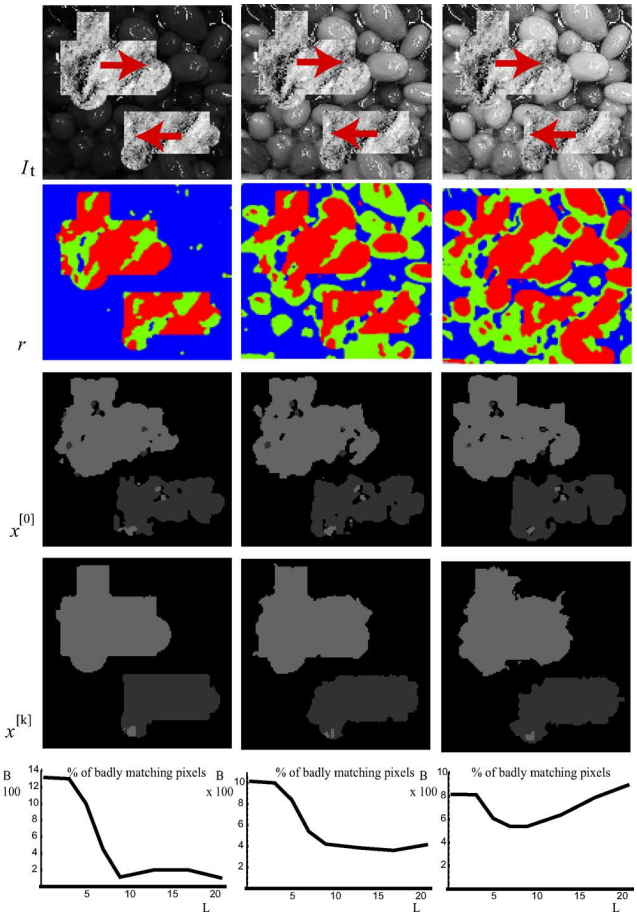

Fig. 4. Three different versions of SEQUENCE A. From left to right, the sequence exhibits a precise, a medium and an imprecise region map r. In every case, the resulting label field $x^{[k]}$ is more precise than the initial one $x^{[0]}$. The last row contains graphics of the percentage of badly matching pixels versus the window size $\Psi = L \times L$. These curves are discussed in Section III-A3.

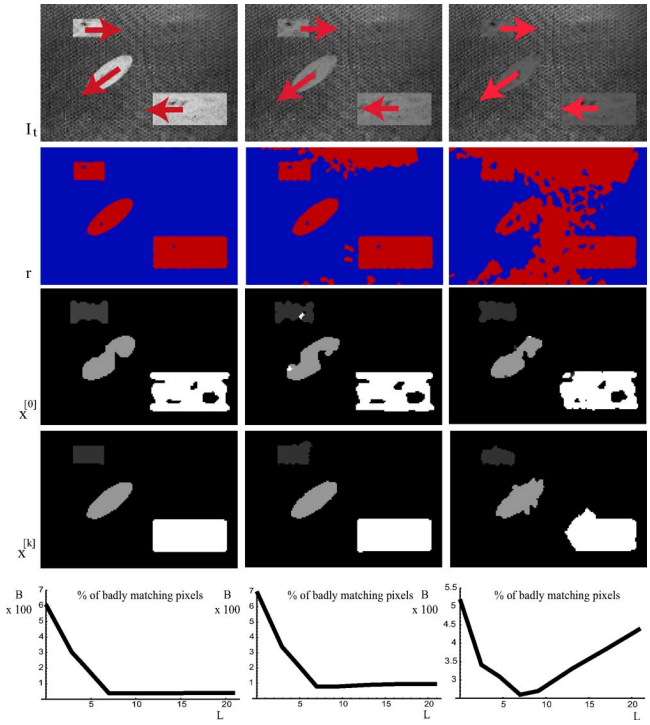

Fig. 5. Three different versions of SEQUENCE B. From left to right, the sequence exhibits a precise, a medium and an imprecise region map r. In every case, the resulting label field $x^{[k]}$ is more precise than the initial one $x^{[0]}$. The last row contains graphics of the percentage of badly matching pixels versus the window size $\Psi = L \times L$. These curves are discussed in Section III-A3.

sequence we have tested, V was computed with a two-level pyramid and an integration window of size 7×7 pixels [51].

3) Motion Segmentation Results: To test the robustness of our motion segmentation framework, different real and synthetic sequences have been segmented. At first, we segmented two synthetic sequences (called sequence "A" and sequence "B") both having a ground truth label field g (see Figs. 4 and 5). Note that both synthetic sequences are made of real images pasted on computer generated shapes. To measure how precise the label fields $x^{[k]}$ returned by our algorithm are, we use the percentage of badly matching pixels between $x^{[k]}$ and the ground-truth label field g, i.e.,

$$B\left(g, x^{[k]}\right) = \frac{100}{N_S} \sum_{s \in S} \left(1 - \delta\left(x_s^{[k]}, g_s\right)\right) \tag{8}$$

where N_S is the number of sites in S and $\delta(x_s^{[k]}, g_s)$ is the Kronecker delta function.

We computed the label field $x^{[k]}$ for both sequences with a different region map r exhibiting precise, medium and imprecise regions. These region maps are used to illustrate how robust our method is with respect to r. Here, the percentage of badly matching pixels is presented in Table I (and on the last row of Figs. 4 and 5). In Table I, our fusion procedure is compared with Altunbasak *et al.*'s method [39] which also relies on a pre-estimated region map r. Since our fusion method does not assign one motion class to every pixel of a region (as is the case for Altunbasak *et al.*'s), our approach is less sensitive to imprecisions in r. This observation illustrates the fact that our algorithm reacts smoothly to a change in its parameters L and r.

Also, as shown in Fig. 6, four real video sequences have been segmented. To illustrates the precision of the resulting label fields, $x^{[0]}$ and $x^{[k]}$ have been superimposed to the image I^t. From left to right, the sequences were segmented with, respectively, three, three, four, and six motion classes. As can be seen in most cases, the label field $x^{[k]}$ returned by our fusion framework is more accurate than the ones with no fusion procedure $(x^{[0]})$.

B. Motion Estimation

1) Introduction: Motion estimation is one of the most studied area in computer vision. Among the solutions proposed for this problem, let us mention variational methods [27], [54]–[59], local methods [54], [60]–[62], frequency-based methods [63], correlation-based methods [64], phase-based methods [65], and Markovian methods [66], [67].

Another class of motion estimation algorithms include those assuming that the overall motion in a video sequence is piecewise parametric [48], [68], i.e., that the motion field may be divided into regions whose motion can be expressed with a parametric motion model. Thus, the goal of these approaches is to jointly estimate the motion regions together with their associate parametric motion model. To this end, the motion regions

TABLE I
PERCENTAGE OF BADLY MATCHING PIXELS COMPUTED WITH THREE DIFFERENT VERSIONS OF TWO SYNTHETIC IMAGE SEQUENCES. FROM LEFT TO RIGHT: RESULTS OBTAINED WITH ALTUNBASAK *et al.* [39], OUR UNSUPERVISED STATISTICAL MARKOVIAN SEGMENTATION ALGORITHM AND RESULTS OBTAINED WITH OUR FUSION ALGORITHM. THE FIVE RIGHTMOST COLUMNS MEASURE THE EFFECT OF THE WINDOW SIZE ($L \times L$). THE QUALITY OF THE SPATIAL PARTITION r IS RANKED FROM PRECISE TO IMPRECISE, DEPENDING ON HOW WELL OBJECTS HAVE BEEN SEGMENTED (SEE FIGS. 4 AND 5)

Sequence A

Partition r	Alt.	$x^{[0]}$	3×3	7×7	11×11	21×21	31×31
precise	0.8	13.2	13.1	5.0	1.9	1.0	0.9
medium	12.5	10.8	10.7	5.4	4.0	4.2	5.3
imprecise	25.5	8.1	8.1	5.4	5.3	8.3	9.3

Sequence B

Partition r	Alt.	$x^{[0]}$	3×3	7×7	11×11	21×21	31×31
precise	0.4	6.2	2.9	0.4	0.4	0.4	0.5
medium	8.9	6.7	3.3	0.7	0.8	0.9	1.3
imprecise	42.6	5.2	3.3	2.0	2.6	2.7	5.4

Fig. 6. Sequences KARLSRUHE, TAXI, TENNIS, TREVOR WHITE, SEQUENCE A, and SEQUENCE B. The first row presents frames at time t, the second row spatial partitions r and the last two rows the motion label fields $x^{[0]}$ and $x^{[k]}$ superposed to I^t. As can be seen, the moving objects are more precisely located after the fusion process ($x^{[k]}$) than before ($x^{[0]}$).

and the motion model parameters are generally estimated with a two-step procedure [69]–[71] that iterates until convergence. The first step consists in estimating the motion model parameters according to the current motion label field [68], [72]. In contrast, the second step estimates new motion regions while the motion models are kept unchanged. Following the work of Murray and Buxton [73], Odobez and Bouthemy [72], and Stiller and Konrad [68], Tekalp [74], [75] summarizes these two steps with a *maximum likelihood* (ML) and MAP procedure. The difference between the former and the latter is the use of an *a priori* energy function that helps smooth the resulting motion label field.

2) Motion Estimation and Our Framework: The goal is to estimate a label field $x^{[0]}$ whose pixels are associated with labels ranging between ζ_0 and ζ_{d-1}. Since the optical flow field is assumed to be piecewise parametric, each class ζ_i is assigned a parameter vector $\vec{A}_{\zeta_l}$. A commonly used parametric model is the six-parameter *affine* model $\vec{A}_{\zeta_l} = (a_{l1}, a_{l2}, a_{l3}, a_{l4}, a_{l5}, a_{l6})$ defined as [75]

$$\vec{v}_s = \begin{pmatrix} v_x \\ v_y \end{pmatrix} = \begin{pmatrix} 1 & i & j & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & i & j \end{pmatrix} \begin{pmatrix} a_{l1} \\ a_{l2} \\ a_{l3} \\ a_{l4} \\ a_{l5} \\ a_{l6} \end{pmatrix} \tag{9}$$

$$= \mathcal{X}_s \vec{A}_{\zeta_l} \tag{10}$$

where (i, j) are the Euclidean coordinates of site s and (v_x, v_y) are the horizontal and vertical components of motion vector $\vec{v}_s$.

In this paper, the motion label field $x^{[0]}$ (as well as the parameters $\vec{A}_{\zeta_l}$, $l \in [0, d[$) are estimated with a MAP procedure

[73], [75] whose objective is to maximize the *a posteriori* pdf $P(x^{[0]}|I)$ where I is the input image sequence (note that for the rest of this subsection, $x^{[0]}$ will be replaced by x to simplify the notation). By the well known Bayes theorem, $P(x|I)$ may be rewritten as

$$P(x|I) = \frac{P(I|x)P(x)}{P(I)} \tag{11}$$

where $P(I|x)$ is the likelihood pdf that measures how well the motion label field x (together with its d motion models) fits the input observation I and $P(x)$ is the prior pdf. Observe that since $P(I)$ is constant with respect to x, it will be ignored during the optimization process.

If we assume that the *true* vector field is perturbed with a zero mean white noise with standard deviation σ and that the mismatch between I and x is modeled with the well-known motion constraint equation

$$E(x_s) = (\partial_x I v_x + \partial_y I v_y + \partial_t I)^2 \tag{12}$$
$$= (\nabla I \vec{v} + \partial_t I)^2 \tag{13}$$

where $\partial_x I, \partial_y I, \partial_t I$ denote the spatial and temporal partial derivatives at site s and time t, the likelihood pdf $P(I, x)$ can be expressed at each pixel as

$$P(I_s|x_s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-E(x_s)/2\sigma^2\right\}. \tag{14}$$

As for the prior pdf $P(x_s)$, the Potts model may be used to incorporate local constraints on the segmentation. This is given by

$$P(x_s) = \frac{1}{Z} \exp\left\{-\beta \underbrace{\sum_{\langle t,s\rangle} (1 - \delta(x_s, x_t)}_{U(x_s)}\right\} \tag{15}$$

where Z is a normalizing constant, $\delta(x_s, x_t)$ is the Kronecker delta, and β is a constant. If we assume that each random variable I_s is independent, the *optimal* solution can be formulated as

$$\hat{x} = \arg\max_x \prod_{s\in S} P(I_s|x_s)P(x_s) \tag{16}$$

or, if we assume that the noise level σ is the same for each class

$$\hat{x} = \arg\min_x \sum_{s\in S} (\nabla I \vec{v}_s + \partial_t I)^2 + \beta U(x_s). \tag{17}$$

Combining (17) and (10) leads to

$$\hat{x} = \arg\min_x \sum_{s\in S} \left(\nabla I \left[\mathcal{X}_s \vec{A}_{x_s}\right] + \partial_t I\right)^2 + \beta U(x_s)$$

which emphasizes the need for a joint estimation of $\vec{A}_i$ and x. As proposed by Murray and Buxton [73] and Tekalp [75], x and $\vec{A}_i$ may be estimated with stochastic optimizers such as simulated annealing or the Metropolis algorithm. In this paper, we use the simulated annealing approach presented in Algorithm 2.

Algorithm 2 Stochastic algorithm used for the motion estimation/segmentation procedure. This algorithm returns a motion label field x as well as the affine motion model $\vec{A}_l$ for each motion class.

Require I Input video sequence

Ensure $x, \vec{A}_l$

Initialization

1: Initialize x with random values.

2: $T = T_0$, the initial temperature

Stochastic optimization.

3: **Repeat**.

4: Update $\vec{A}_0, \vec{A}_1, \ldots, \vec{A}_{d-1}$ according to

$$\vec{A}_l = \arg\min_{\vec{A}} \sum_{\substack{s\in S \\ x_s=\zeta_l}} \left(\nabla I[\mathcal{X}_s \vec{A}] + \partial_t I\right)^2. \tag{18}$$

This minimization is done with a least-square estimation

$$\vec{A}_l = \left[\left(\sum_{\substack{s\in S \\ x_s=\zeta_l}} (\nabla I \mathcal{X}_s)^{\mathrm{T}} (\nabla I \mathcal{X}_s)\right)^{-1} \sum_{\substack{s\in S \\ x_s=\zeta_l}} (\nabla I \mathcal{X}_s)^{\mathrm{T}} \partial_t I\right].$$

5: For each site, compute the probability $P(x_s = \zeta_l|I)$ related to each label $\zeta_i \in \Gamma$ as

$$P(x_s = \zeta_l|I) = \frac{1}{Z} \exp -E_s(\zeta_l|I)/T$$
$$E_s(\zeta_l|I) = \sum_{s\in S} \left(\nabla I \left[\mathcal{X}_s \vec{A}_{\zeta_l}\right] + \partial_t I\right)^2 + \beta U(x_s)$$

and randomly assign a label $\zeta_j \in \Gamma$ to x_s according to its probability $P(\zeta_j|I)$.

6: $T \leftarrow T \times \mathrm{coolingRate}$.

7: **Until** T reaches a minimum temperature.

8: **Return** $x^{[k]}$.

Once the motion label field x has been estimated, it can be fused with r using our fusion procedure (Algorithm 1).

3) Motion Estimation Results: For this application, we segmented four synthetic sequences having a ground-truth label field g and ground-truth vector field V_g. These sequences are presented in Figs. 7–9. In Fig. 7(a), the sequence (which we call sequence "C") exhibit two shapes moving in different directions on a flat grayscale background. In this case, since the region map r contains highly precise edges, the results $x^{[k]}$ also exhibits precise regions. This is shown by the percentage of badly matching pixels (in Table II) which is much larger for $x^{[0]}$ than for $x^{[k]}$. As

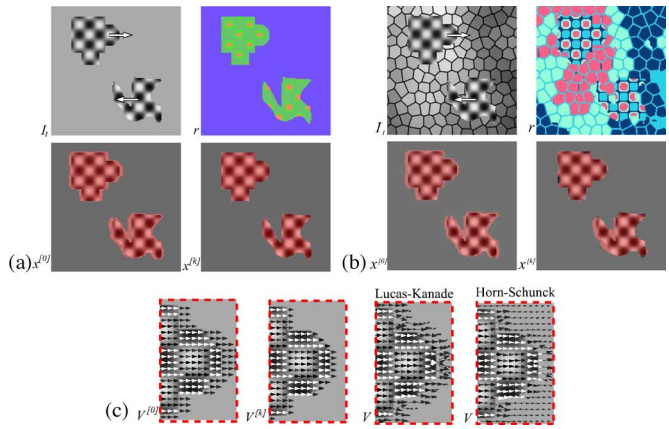

Fig. 7. Sequence "C" (a) with flat background and (b) with textured background. Note that the textured background has been removed from $x^{[0]}$ and $x^{[k]}$ to help visualize the results. In (c), vector fields obtained with Algorithm 2 ($V^{[0]}$), our fusion method ($V^{[k]}$), Lucas–Kanade (LK) [52], and Horn–Schunck (HS) [54] optical flow methods.

for the sequence of Fig. 7(b), it exhibits the same moving shapes as in Fig. 7(a) but in front of a textured background. With this textured background, the region map r contains regions less precise than those of Fig. 7(a). This is because the background is composed of shades similar to the ones on the moving shapes. However, nevertheless, as shown in Table II, even with an imprecise region map r, $x^{[k]}$ is still significantly more precise than $x^{[0]}$.

The third synthetic sequence we segmented is the one shown in Fig. 8. It contains an image of the Parthenon moving to the right, on top of an image performing a counterclockwise rotation. From left to right are the ground-truth, the scene estimated with algorithm 2, and the result of our fusion procedure. Because of the highly textured background, the region map r contains local imprecisions, especially around the first, second, third, and fifth columns. As can be seen in image $x^{[k]}$ and $||V^{[k]}||$, the local imprecisions in r have induced local errors in the resulting fields, especially around the first and the fifth column. However, nevertheless, the results returned by our fusion procedure are less noisy and globally exhibit more precise edges than the ones returned by the MAP procedure. This observation is underlined by the percentage of badly matching pixels presented in Table II which is almost three times smaller for $x^{[k]}$ than for $x^{[0]}$.

The fourth synthetic sequence is the famous YOSEMITE [53] sequence presented in Fig. 9. As can be seen, even if r has local imprecisions (especially in the top left section of the scene) the resulting label field $x^{[k]}$ (as well as $V^{[k]}$) is much smoother than $x^{[0]}$.

We also implemented a metric to measure how good the vector fields $V^{[k]}$ returned by our framework are. Following Barron *et al.* [53], we used to the average angular error metric to evaluate the distance between the ground truth vector field V_g and the estimated vector field (be it $V^{[0]}$ or $V^{[k]}$), namely

$$\bar{\Psi}_E(V, V_g) = \frac{1}{\mathcal{N} \times \mathcal{M}} \sum_{s \in S} \arccos\left(\vec{v}_s . \vec{v}_{g_s}\right) \qquad (19)$$

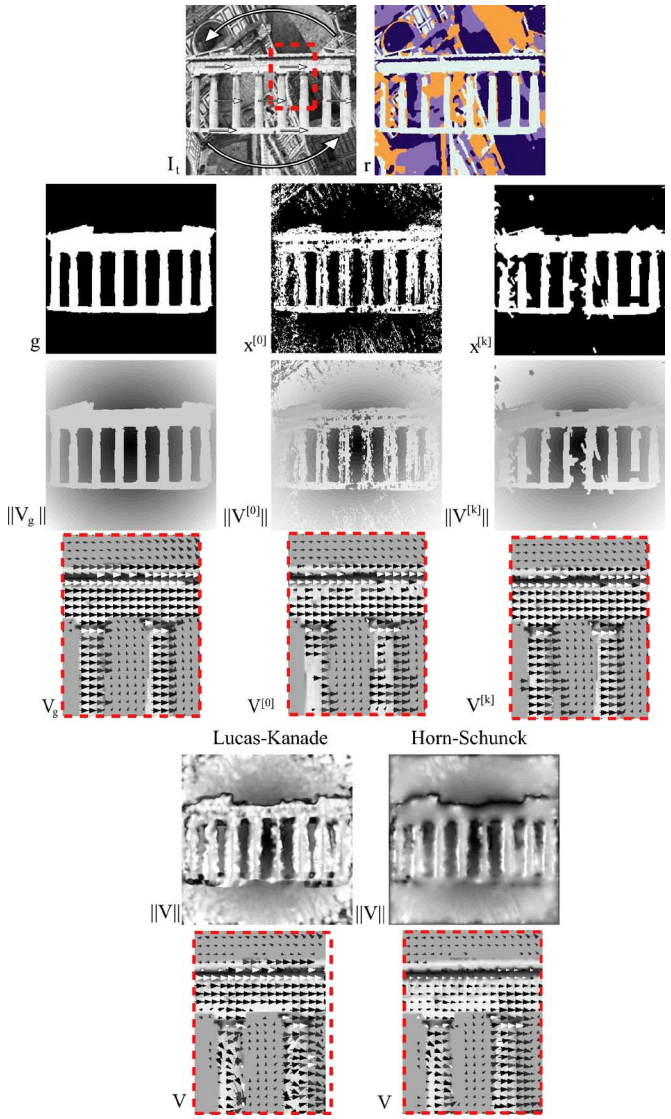

Fig. 8. PARTHENON sequence with the region map r, ground-truth images, initial estimates, and the result of our fusion procedure. Note that the third row exhibits the vector fields' magnitude.

where $\vec{v}_s$ and $\vec{v}_{g_s}$ are normalized 3-D vectors: $\vec{v}_s = ((u, v, 1)/\sqrt{u^2 + v^2 + 1})$. Using this metric, the vector field $V^{[k]}$ of the four synthetic sequences are compared to the ones returned by the MAP procedure. The angular errors are presented in Table III and again, the results clearly favors our method.

We also segmented a real image sequence, namely the RHEINHAFEN sequence presented in Fig. 10. Again, our results shows sharper edges and less isolated false positives and false negatives.

C. Occlusion Detection

1) Introduction: The goal of most optical flow and stereovision algorithms is to estimate a matching function (be it a disparity map [76] or an optical flow field [53]) between the pixels of two (or more) input images. Due to motion or to a parallax effect between a *left* and a *right* image, most scenes contain areas that are visible in only one frame. Generally speaking, these

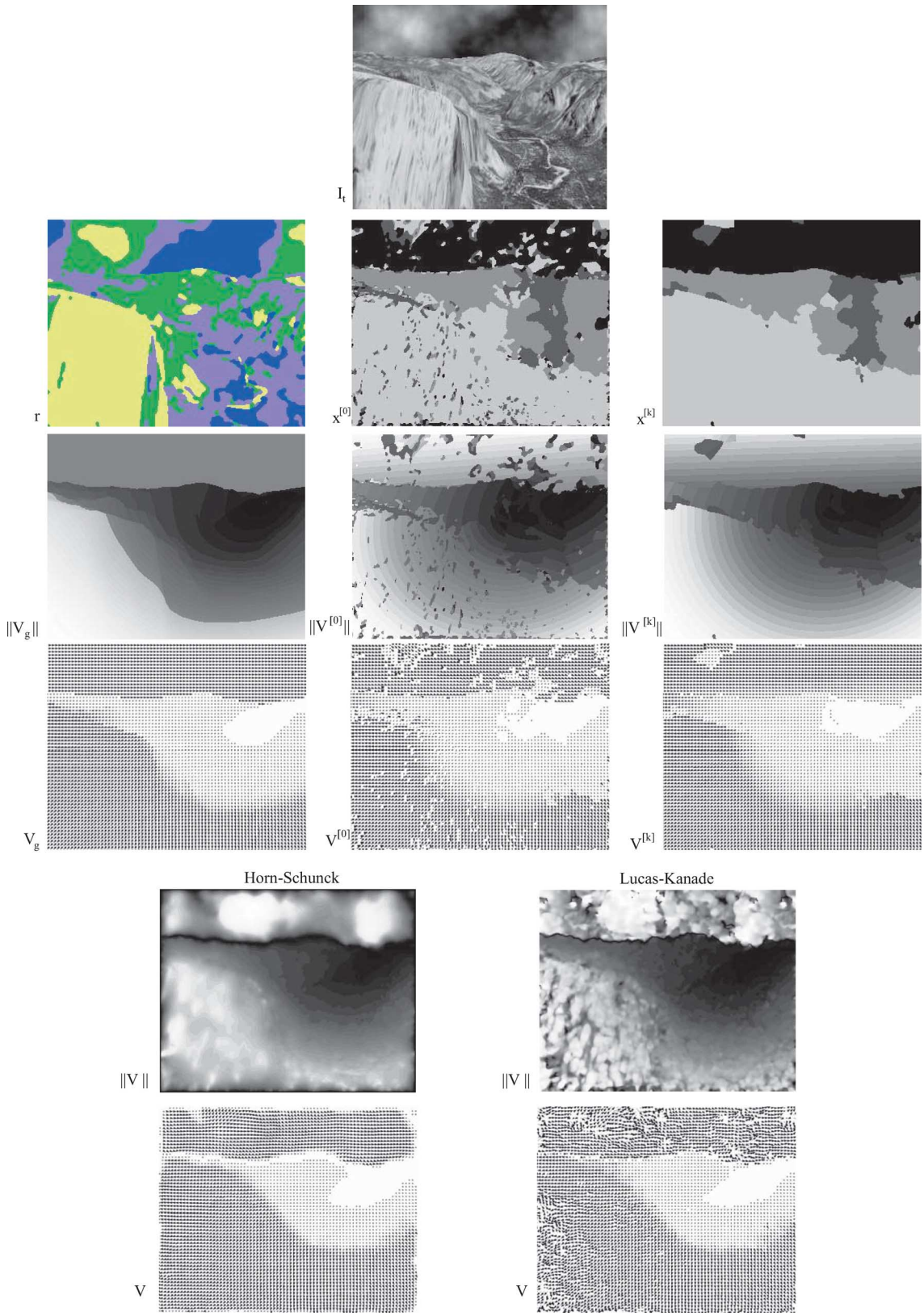

Fig. 9. Yosemite sequence with ground-truth, results from the MAP estimation/segmentation procedure, and results from our fusion framework. Although r contains local imprecisions, the fusion procedure significantly reduces the number of false positives and false negatives in x.

half-occluded areas are either *newly exposed* or *newly occluded* [77], [78]. Since these areas have no direct correspondence in the second image, they are a classical source of error for most motion or depth estimation algorithms.

While many authors have considered occlusion as a source of noise that is to be fought with spatial smoothing [76], others have explicitly included an occlusion criterion in their energy function [26], [79]–[83]. During the past few years, a variety

TABLE II
PERCENTAGE OF BADLY MATCHING PIXELS COMPUTED FOR THREE SYNTHETIC SEQUENCES

Sequence	$x^{[0]}$	$x^{[k]}$
Seq.C, flat back.	4.1%	0.006%
Seq.C, textured back.	4.4%	2.4%
Parthenon	16.4%	5.5%

TABLE III
AVERAGE ANGULAR ERROR WITH STANDARD DEVIATION COMPUTED FOR FOUR DIFFERENT SEQUENCES. THESE RESULTS SHOW THE ANGULAR ERROR WITH ($V^{[k]}$) AND WITHOUT ($V^{[0]}$) FUSION, AS WELL AS FOR THE LK AND THE HS METHODS

Sequence	$V^{[0]}$	$V^{[k]}$
Seq.C, flat back.	(4.1, 15.4)	(0.3, 0.5)
Seq.C, textured back.	(4.0, 15.3)	(3.2, 13.8)
Parthenon	(13.7, 24.0)	(8.1, 15.2)
Yosemite	(16.3, 18.5)	(9.5, 10.9)
	HS	LK
Seq.C, flat back.	(8.0,18.1)	(16.5,20.1)
Seq.C, textured back.	(8.0,18.1)	(16.5,20.1)
Parthenon	(13.3, 21.6)	(13.6, 17.8)
Yosemite	(11.0, 16.6)	(10.6, 11.3)

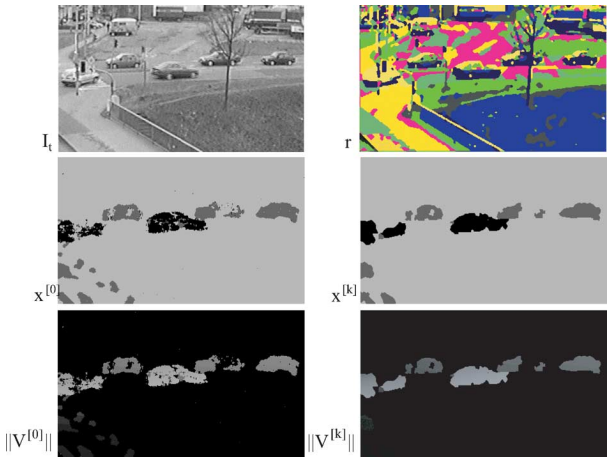

Fig. 10. Zoom on the RHEINHAFEN sequence. In this case, the fusion process has significantly reduced the number of false positives and false negatives. As a result, the moving areas are more homogeneous.

of occlusion metrics have been proposed among which the one Egnal and Wildes [84] call the *left-right-check* (LRC) has drawn a lot of attention. This approach stipulates that the matching function between the left and the right image should differ only by sign with the right-left matching function. In this context, every pixel for which the difference between the left-right match and the right-left match is above a given threshold is considered as being occluded. Although the LRC can be useful within a global energy function [83], [85], [86], many researchers have noted that the LRC is error-prone in noisy areas [78] and in areas having little or no texture [82], [84]. Others have also argued that estimating the forward *and* the backward matching functions can be prohibitive time wise.

Another idea that enjoys a great deal of popularity is Marr–Poggio's [87] uniqueness assumption. This assumption stipulates that there is always a one-to-one correspondence between the pixels of the two frames. Kolmogorov and Zabih [88] incorporated that assumption into their graph-cut algorithm and stipulated that each pixel in one image should correspond to *at most* one pixel in the other image. A pixel with no match would then be considered as being occluded. A variation of this approach has been proposed by Sun *et al.* [26] for which a nonoccluded pixel must have *at least* one match. Although the difference between the two approaches is conceptually slight, Sun *et al.* [26] demonstrate that their method is superior in scenes containing slanted surfaces.

Let us also mention that some authors use the so-called *ordering constraint* [26], [81], [84] which stipulates that a point P laying to the right of a point Q in one image should also lie to the right of Q in the other image. Although this assumption is often true, it can easily be violated by narrow front-ground objects (what Sun *et al.* [26] call the "double nail illusion").

2) Occlusion Detection and Our Framework: After thorough evaluations of many occlusion detection criteria, we came to realize that the ones based on Marr–Poggio's uniqueness assumption are the most accurate, at least in the context of our framework (in their review paper, Egnal and Wildes [84] came to a similar conclusion). More specifically, the Ince–Konrad [78] metric was retained to compute $x^{[0]}$, a "rough" occlusion map estimate. The Ince–Konrad [78] metric can be seen as a generalization of the uniqueness constraint: instead of counting the number of matches for each pixel independently, they count the number of matches within a given local neighborhood. Consider $\Lambda = \{s | s \in S\}$ the set of pixels in the reference image I^{ref} and $\Delta = \{u | u = s + d_s, s \in S\}$, the set of *matching* pixels in I^{mat}.[1] Based on Δ, an accumulation function M is computed

$$M_s = \sum_{i \in \Delta} \zeta_{i,s} \tag{20}$$

where $\zeta_{i,s} = 1$ if the Euclidean distance between pixel $s \in S$ and $i \in \Delta$ is lower than or equal to D, and zero otherwise. The occlusion map $x^{[0]}$ is obtained by thresholding M_s

$$x_s^{[0]} = \begin{cases} 1, & \text{if } M_s < \tau \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

As suggested by the authors [78], we set D to 2.

The Ince–Konrad method can be intuitively understood following the synthetic example of Fig. 12. In this example, with $D = 2$, the accumulation function M_s equals 11 for pixel A, 5 for pixel B and zero for pixel C. With a threshold τ of 3 for example, pixel C would be considered as being occluded.

Because occlusion is a mismatch between *two* images, the way the region map r is computed is slightly different than for the other applications. In fact, the input frames I^{ref} and I^{mat} are, respectively, segmented into two label fields, namely r^{ref} and r^{mat}. These two region maps are then linearly combined together: $r = r^{\text{ref}} + c \times r^{\text{mat}}$ where c is the number of classes in which I^{ref} and I^{mat} have been segmented. This last operation results in a label field r whose regions are uniform in the sense of both input images. An example of such a region map is presented in Fig. 11. Once r and $x^{[0]}$ have been computed, they can be fused with Algorithm 1.

3) Occlusion Detection Results: To validate our method, we detected occlusion on various data sets commonly used for such

[1] I^{ref} and I^{mat} stands for the "left" and "right" image in stereovision and for the images at time t and $t + 1$ in optical flow. As for d_s, it represents the disparity value linking pixel I_s^{ref} to its projection in I_s^{mat} as shown in Fig. 12.

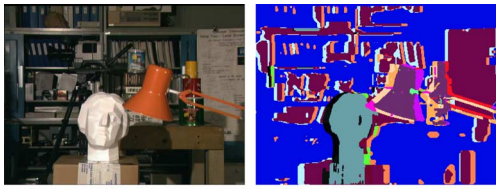

Fig. 11. TSUKUBA left image with the region map r obtained after segmenting the two input images $I^{[ref]}$ and $I^{[mat]}$.

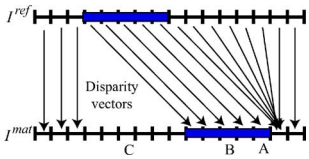

Fig. 12. Synthetic example of a disparity map between two images $I^{[ref]}$ and $I^{[mat]}$. In this example, "C" is an occluded pixel since it is visible in $I^{[mat]}$ but occluded by the blue object in $I^{[ref]}$.

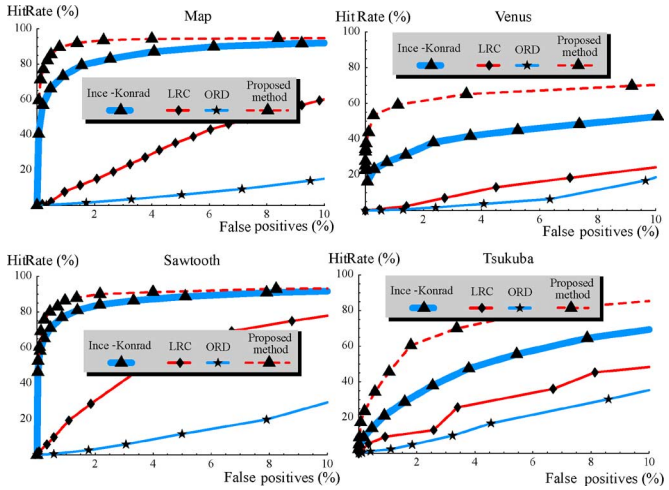

Fig. 13. Hit rate versus false positive rates obtained with four different data sets. The proposed method significantly reduces the number of false positives and false negatives.

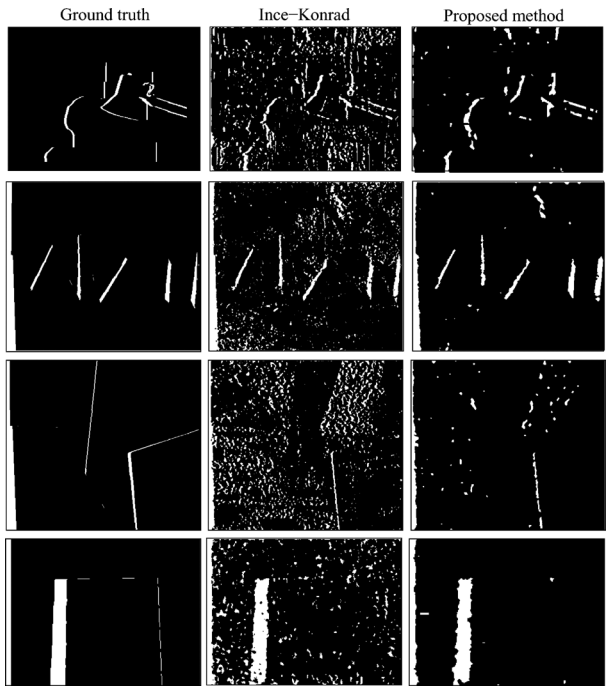

Fig. 14. From top to bottom, ground truth and results obtained for TSUKUBA, SAWTOOTH, VENUS, and MAP data set. Hit rate for every result is, respectively, 60%, 90%, 45%, and 90%.

an application. Here, our framework is compared with other frequently used approaches, namely the *left-right check* (LRC) [84], the *ordering constraint* (ORD) [84], and Ince–Konrad's [78] uniqueness-based approach.

Four sequences with ground truth taken from Middlebury web page [89] have been used to test the methods. As mentioned in Section III-C, occlusion detection is closely related to applications involving a disparity map (such as optical flow and stereovision). A disparity map is a function that links the pixels of one image (I^{ref}) to their projection in a second image (I^{mat}). In this context, an occluded area is a section of the scene that is visible in one image but hidden in the second image. Although a disparity map can be estimated in a variety of ways as mentioned in most optical flow and stereovision review papers [53], [76], we estimate it with a simple pixel-based window matching strategy taken from the work of Scharstein and Szeliski [76]. This method, called *winner-take-all*, is a greedy algorithm looking at each pixel for the disparity value that best minimizes a matching cost. Here, the matching cost is a simple squared difference of intensity values. The resulting disparity map is filtered out with a 3×3 *shiftable* aggregation filter, i.e., a box filter that locally adapts to the scene. For more details on this algorithm, please refer to the work of Scharstein and Szeliski [76].

Following Egnal and Wildes' methodology [84], we have plotted the hit rate/false positive rate curve of every method by varying their threshold (see Fig. 13). According to these graphs, an optimal method is one that maximizes the surface below its curve. Consequently, as can be seen on every graphic of Fig. 13, our method appears to be more precise than the others we have implemented. This is especially true for those sequences containing large textureless areas such as VENUS and TSUKUBA. This can be explained by the fact that, as mentioned by Egnal and Wildes [84], most common occlusion detection methods are error-prone in textureless areas. In this context, using a region-based approach to eliminate isolated false positives provides a clear advantage.

A qualitative comparison has also been made in Fig. 14. To make the results objectively comparable, each method has been tuned to return an occlusion map with a specific hit rate. In this way, the results in the second and third column of Fig. 14 have, respectively, hit rates of 60%, 90%, 45%, and 90%. Although the hit rate is the same for both approaches, the false positive rate is clearly better for our method.

As for the FLOWERGARDEN sequence of Fig. 15, our method produce again a significantly lower number of false positives. Note that for this sequence, the matching function was computed with a pixel-based window-matching strategy [53].

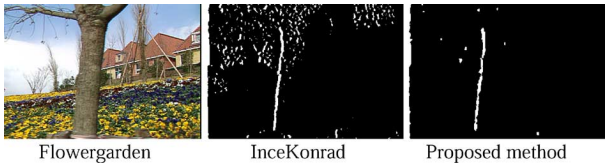

Fig. 15. Comparison of the proposed method with the Ince Konrad's method on the FLOWERGARDEN sequence.

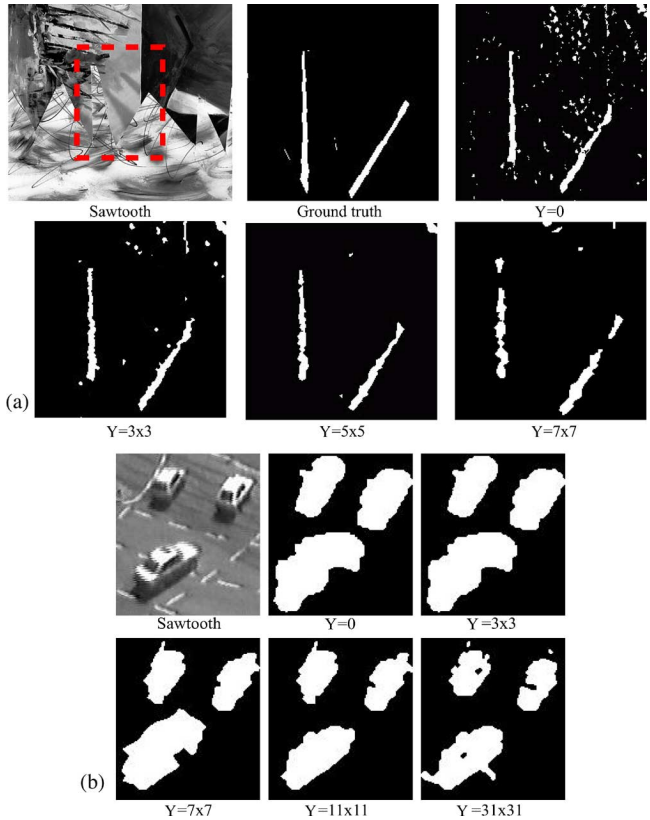

Fig. 16. This figure illustrates the influence of the neighborhood size $\Psi = L \times L$. In (a), occlusion maps of the SAWTOOTH sequence and in (b), motion maps of the KARLSRUHE sequence.

IV. FURTHER EXPERIMENTAL RESULTS

A. Testing the Influence of Ψ and α

Since our fusion framework depends very much on the size of the 2-D neighborhood Ψ, we illustrate its influence with various qualitative and quantitative results. In Fig. 16, the effect of varying the window size is shown on a synthetic and a real sequence. As can be seen, a modification of this variable brings a smooth and predictive variation in the resulting image. However, it should be noted that the use of a too large window size can cause unexpected local errors as shown in the lower right image of both thumbnails of Fig. 16.

We also evaluated quantitatively the influence of Ψ on the percentage of badly matching pixels for two synthetic sequences. This is presented in the last row of Figs. 4 and 5 (and in Table I). These examples show that the use of our fusion procedure does indeed help enhance the quality of the results even for sequences having an imprecise region map r. However, using a too large window Ψ for sequences having an imprecise region map can induce local errors and, thus, raise the percentage of badly matching pixels. This being said, we observed that for every sequence segmented in this paper, a window size ranging between 5×5 and 11×11 produces the most satisfying results.

As for the coefficient α of (4), we tested it on three sequences shown in Fig. 17. As we mentioned previously, the reason for this coefficient is to give a relative influence to $x^{[0]}$ and to r during the fusion procedure (and, thus, make the optimization process well posed). In fact, when r exhibits sharp regions (as is the case for the first sequence on top to Fig. 17) a small value might be assigned to α (i.e., a value that gives more weight to r than to $x^{[0]}$). However, when the region map r contains imprecise regions as is the case for the two other examples in Fig. 17, a nonzero value for α (here between $0.1 * (11 \times 11)$ and $0.05 * (11 \times 11)$) is preferable. In this way, the to-be-minimized energy function has a nonzero reaction that prevents the resulting vector field $x^{[k]}$ from being too different from the initial guess $x^{[0]}$. As mentioned previously, a nonzero α value allows stochastic optimizers to converge towards a meaningful solution $x^{[k]}$. To illustrate that assertion, we minimized (4) with the deterministic optimizer ICM and with the stochastic simulated annealing optimizer. The results are illustrated in Fig. 18. As can be seen, although the global energy of the ICM label field is slightly higher, the results are very much similar. This illustrates the fact (4) is nearly convex, i.e., not too erratic and can be efficiently minimized with a simple (yet suboptimal) ICM minimization procedure.

B. Real-Time Processing

Since our fusion framework process every pixel independently, it can be implemented on a parallel architecture. To this end, we implemented our method on a graphics processor unit (GPU) [90]. A GPU is a processor embedded on most graphics card nowadays available on the market which, among other things, can load, compile and execute programs implemented with a C-like language. The key feature of the GPUs is their fundamental ability to process *in parallel* each pixel of the scene, making all kinds of applications much more efficient than when implemented on traditional sequential CPUs. For example, the fusion procedure (with $\Psi_s = 5 \times 5$) can process at a rate of 25 fps a scene of size 384×288 such as TSUKUBA.[2] Also, the region map r of this scene can be computed in 1 second or, if the Gaussian parameters are re-used from a previous calculation, in 0.05 s. These processing rates outperformed by a factor of almost 100 what we obtained with a traditional CPU implementation. For an application such as motion estimation/segmentation (Algorithm 2), computing $x^{[0]}$ requires approximately 50 s whereas estimating $x^{[n]}$ $(x^{[0]} + r + \text{fusion})$ requires no more than 52 seconds for a 256×256 scene. For more details on how r can be computed with a GPU, please refer to [46].

V. CONCLUSION

In this paper, we considered the issue of fusing label fields instead of features as is usually the case. The core of our method is a fusion framework that blends together a region map r and an application label field $x^{[0]}$. With the assumption that the color regions in r are more detailed than the regions in $x^{[0]}$, the goal of the fusion procedure is to iteratively modify the application

[2]Since there are no efficient ways to access the framebuffer content to verify if the ICM algorithm has converged (see part 2 of Algorithm 1), a predefined number of ten ICM iterations has been used.

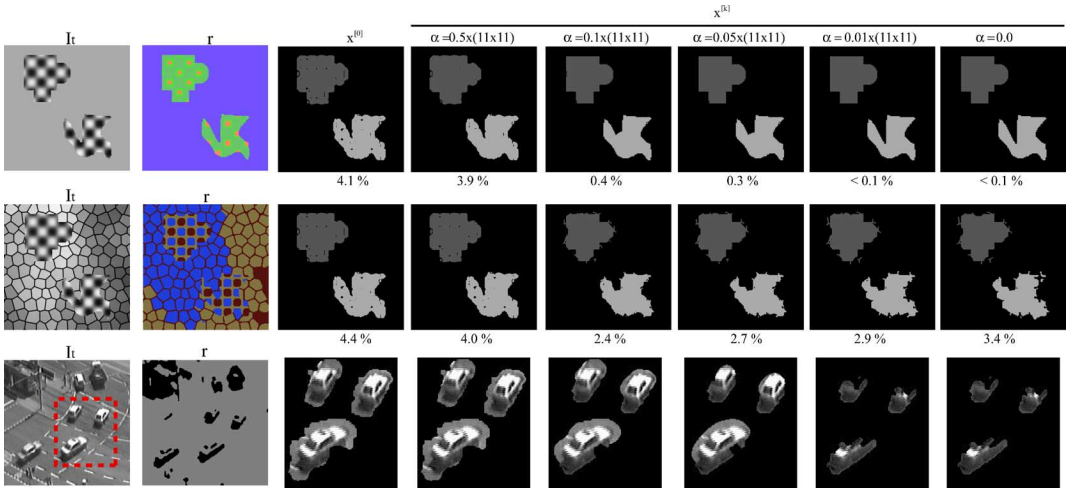

Fig. 17. Three sequences segmented with different α values. The fusion procedure has a window size $\Psi = 11 \times 11$ and the percentage of badly matching pixels is shown below the synthetic label Fields. Among the three region maps, the first one exhibits precise regions, the second one less precise regions and the last one very imprecise two-class regions.

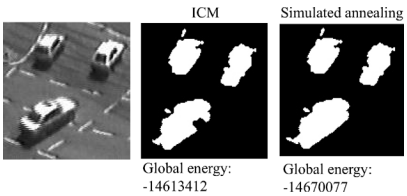

Fig. 18. Results obtained after minimizing eq. (4) with ICM and Simulated annealing. In this example, $\Psi = 9 \times 9$ and $\alpha = 0.01 \times (9 \times 9)$. The ICM label field has been obtained after 24 iterations whereas 250 iterations have been needed for simulated annealing.

field x to make its regions fit the color regions. In this way, false positives and false negatives are filtered out and blobby shapes are sharpened resulting in a more precise label field $\hat{x}$. The fusion is performed by an ICM optimization procedure that minimizes an energy function that may be a pure fusion function (and, thus, works at each ICM iteration in a similar way to the well-known K-nearest neighbor algorithm) or a fusion-reaction energy function involving a data-related term.

Although such a segmentation framework based on pre-estimated label fields might appear as a step backward when compared to methods that minimize one large multidimensional energy function [4], [25] or to methods using dimensionality reduction algorithms [14], [15], our method has decided advantages. To start off with, traditional multidimensional methods often rely on weighting factors that give more or less influence to some features. Since a bad choice of these weighting parameters can lead to a bad segmentation, they must be carefully adjusted at runtime which, of course, can be cumbersome for most unsupervised applications. Also, because these parameters generally depend on the sequence content, they need to be re-estimated when a new sequence is processed. Furthermore, tweaking these weighting factors is not a small task, especially when their number is large. For instance, the method proposed by Black requires that a number of eight weighting factors need to be tweaked [25]. Moreover, large energy functions are generally less stable than smaller ones and, thus, often need to be implemented with a stochastic (and slow) optimizer such as simulated annealing.

The point of our method is to alleviate these problems by individually minimizing two energy functions in order to blend their label fields. Thus, our method does not rely on weighting factors, scaling functions or projection functions often used when blending features. Our method uses simple energy functions that can be minimized with a deterministic optimizer (such as ICM) which is much faster than say, simulated annealing. Finally, we believe our fusion method is simple to implement and can be easily transposed to a parallel architecture such as a GPU.

Results obtained on real and synthetic image sequences show that our algorithm is stable and precise. It reacts well to changes of its parameters Ψ and α and performs well even when supplied with poorly estimated region maps r.

As mentioned previously, our method mostly depends on two parameters, namely the window size Ψ and the coefficient α. In our implementation, these two parameters are specified by the user in a supervised way. However, if an application had access to a corpus of images with ground-truth data, it would be possible to automatically estimate an "optimal" value for Ψ and α. In this context, the search for the "best" value for Ψ and α could be done by successively segmenting every image of the corpus and comparing the resulting segmentation maps with the ground-truth maps. This search could be done in a brute force way by testing a large number of values for Ψ and α and keeping the ones associated to the best results. On the other hand, a simplex optimizer could also be implemented to reduce the computational cost.

ACKNOWLEDGMENT

The authors would like to thank the Middlebury stereovision research team (http://www.cat.middlebury.edu/stereo/) for providing the stereo-pair images, J. Barron for his precious help with the Yosemite sequence, and the group of Prof. H.-H. Nagel for providing most video sequences (http://i21www.ira.uka.de/image_sequences/).

REFERENCES

[1] H. Tao, H. Sawhney, and R. Kumar, "A global matching framework for stereo computation," in *Proc. ICCV*, 2001, pp. 532–539.

[2] M. Black and A. Jepson, "Estimating optical flow in segmented images using variable-order parametric models with local deformations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 18, no. 10, pp. 972–986, Oct. 1996.

[3] Z. Kato, T. Pong, and S. Qiang, "Multicue MRF image segmentation: Combining texture and color features," in *Proc. ICPR*, 2002, vol. 1, pp. 660–663.

[4] S. Khan and M. Shah, "Object based segmentation of video color, motion and spatial information," in *Proc. CVPR*, 2001, pp. 746–751.

[5] A. Solberg and A. Jain, "Texture fusion and feature selection applied to SAR imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 35, no. 2, pp. 475–479, Mar. 1997.

[6] D. Clausi and H. Deng, "Design-based texture feature fusion using Gabor filters and co-occurrence probabilities," *IEEE Trans. Image Process.*, vol. 14, no. 7, pp. 925–936, Jul. 2005.

[7] O. Pichler, A. Teuner, and B. Hosticka, "An unsupervised texture segmentation algorithm with feature space reduction and knowledge feedback," *IEEE Trans. Image Process.*, vol. 7, no. 1, pp. 53–61, Jan. 1998.

[8] J. Bigun, "Unsupervised feature reduction in image segmentation by local transforms," *Pattern Recognit. Lett.*, vol. 14, pp. 573–583, 1993.

[9] R. Kanth, D. Agrawal, and A. Singh, "Dimensionality reduction for similarity searching in dynamic databases," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 1998, pp. 166–176.

[10] W. Jiang, G. Er, Q. Dai, and J. Gu, "Similarity-based online feature selection in content-based image retrieval," *IEEE Trans. Image Process.*, vol. 15, no. 3, pp. 702–712, Mar. 2006.

[11] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proc. CVPR*, 1991, pp. 586–591.

[12] I. Dagher and R. Nachar, "Face recognition using IPCA-ICA algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 6, pp. 996–1000, Jun. 2006.

[13] C. Liu, "Capitalize on dimensionality increasing techniques for improving face recognition grand challenge performance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 5, pp. 725–737, May 2006.

[14] C. Bishop, *Neural Networks for Pattern Recognition*. Oxford, U.K.: Oxford Univ. Press, 1996.

[15] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. New York: Wiley, 2000.

[16] D. Donoho, "High-dimensional data analysis: The curses and blessings of dimensionality," Lecture, Amer. Math. Soc., Math Challenges of the 21st Century, 2000.

[17] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, 2003.

[18] A. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," *Pattern Recognit.*, vol. 24, no. 12, pp. 1167–1186, 1991.

[19] V.-P. Onana, E. Trouve, G. Mauris, J.-P. Rudant, and E. Tonye, "Linear features extraction in rain forest context from interferometric SAR images by fusion of coherence and amplitude information," *IEEE Trans. Geosci. Remote Sens.*, vol. 41, no. 11, pp. 2540–2556, Nov. 1999.

[20] H. Li, R. Deklerck, B. De Cuyper, A. Hermanus, E. Nyssen, and J. Cornelis, "Object recognition in brain ct-scans: Knowledge-based fusion of data from multiple feature extractors," *IEEE Trans. Med. Imag.*, vol. 14, no. 2, pp. 212–229, Feb. 1995.

[21] A. Filippidis, L. Jain, and N. Martin, "Using genetic algorithms and neural networks for surface land mine detection," *IEEE Trans. Signal Process.*, vol. 47, no. 1, pp. 176–186, Jan. 1999.

[22] F. Bujor, E. Trouve, L. Valet, J.-M. Nicolas, and J.-P. Rudant, "Application of log-cumulants to the detection of spatiotemporal discontinuities in multitemporal SAR images," *IEEE Trans. Geosci. Remote Sens.*, vol. 42, no. 10, pp. 2073–2084, Oct. 2004.

[23] A. Bendjebbour, Y. Delignon, L. Fouque, V. Samson, and W. Pieczynski, "Multisensor image segmentation using Dempster-Shafer fusion in markov fields context," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 8, pp. 1789–1798, Aug. 2001.

[24] W. Pieczynski and D. Benboudjema, "Multisensor triplet markov fields and theory of evidence," *Image Vis. Comput.*, vol. 24, no. 1, pp. 61–69, 2006.

[25] M. Black, "Combining intensity and motion for incremental segmentation and tracking over long image sequences," in *Proc. ECCV*, 1992, pp. 485–493.

[26] J. Sun, Y. Li, S. Kang, and H.-Y. Shum, "Symmetric stereo matching for occlusion handling," in *Proc. CVPR*, 2005, pp. 399–406.

[27] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. ECCV*, 2004, pp. 25–36.

[28] F. Heitz and P. Bouthemy, "Multimodal estimation of discontinuous optical flow using markov random fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 12, pp. 1217–1232, Dec. 1993.

[29] S.-H. Lai and B. Vemuri, "Reliable and efficient computation of optical flow," *Int. J. Comput. Vis.*, vol. 29, no. 2, pp. 87–105, 1998.

[30] S. Jabri, Z. Duric, H. Wechsler, and A. Rosenfeld, "Detection and location of people in video images using adaptive fusion of color and edge information," in *Proc. ICPR*, 2000, pp. 4627–4631.

[31] J. Kittler, M. Hatef, R. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 226–239, Mar. 1998.

[32] A. Kushki, P. Androutsos, K. Plataniotis, and A. Venetsanopoulos, "Retrieval of images from artistic repositories using a decision fusion framework," *IEEE Trans. Image Process.*, vol. 13, no. 3, pp. 277–292, Mar. 2004.

[33] S. Reed, I. R. Ruiz, C. Capus, and Y. Petillot, "The fusion of large scale classified side-scan sonar image mosaics," *IEEE Trans. Image Process.*, vol. 15, no. 7, pp. 2049–2060, Jul. 2006.

[34] M. Bleyer and M. Gelautz, "A layered stereo matching algorithm using image segmentation and global visibility constraints," *ISPRS J. Photogr. Remote Sens.*, vol. 59, no. 3, pp. 128–150, 2005.

[35] Y. Wei and L. Quan, "Region-based progressive stereo matching," in *Proc. CVPR*, 2004, pp. 106–113.

[36] L. Hong and G. Chen, "Segment-based stereo matching using graph cuts," in *Proc. CVPR*, 2004, pp. 74–81.

[37] P. Smith, T. Drummond, and R. Cipolla, "Layered motion segmentation and depth ordering by tracking edges," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 4, pp. 479–494, Apr. 2004.

[38] W. Thompson, "Combining motion and contrast for segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 2, no. 6, pp. 543–549, Jun. 1980.

[39] Y. Altunbasak, P. Eren, and M. Tekalp, "Region-based parametric motion segmentation using color information," *Graph. Models Image Process.*, vol. 60, no. 1, pp. 13–23, 1998.

[40] C. Fuh and P. Maragos, "Region-based optical flow estimation," in *Proc. CVPR*, 1989, pp. 130–135.

[41] M. Mignotte, "A segmentation-based regularization term for image deconvolution," *IEEE Trans. Image Process.*, vol. 15, no. 7, pp. 1973–1974, Jul. 2006.

[42] P.-M. Jodoin and M. Mignotte, "Motion segmentation using a k-nearest-neighbor-based fusion procedure of spatial and temporal label cues," in *Proc. ICIAR*, 2005, pp. 778–785.

[43] P.-M. Jodoin, C. Rosenberger, and M. Mignotte, "Detecting half-occlusion with a fast region-based fusion procedure," presented at the BMVC, 2006.

[44] J. Besag, "On the statistical analysis of dirty pictures,," *J. Roy. Statist. Soc.*, vol. 48, no. 3, pp. 259–302, 1986.

[45] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 6, no. 6, pp. 721–741, Jun. 1984.

[46] P.-M. Jodoin and M. Mignotte, "Markovian segmentation and parameter estimation on graphics hardware," *J. Electron. Imag.*, vol. 15, no. 3, p. 033005, 2006.

[47] W. Pieczynski, "Statistical image segmentation," *Mach. Graph. Vis.*, vol. 1, no. 1, pp. 261–268, 1992.

[48] D. Zhang and G. Lu, "Segmentation of moving objects in image sequence: A review," *Circ. Syst. Signal Process.*, vol. 20, no. 2, pp. 143–183, 2001.

[49] R. Megret and D. DeMenthon, "A survey of spatio-temporal grouping techniques," Tech. Rep., Univ. Maryland, College Park, 2002.

[50] L. Bergen and F. Meyer, "A novel approach to depth ordering in monocular image sequences," in *Proc. CVPR*, 2000, pp. 536–541.

[51] J.-Y. Bouguet, "Pyramidal implementation of the Lucas Kanade feature tracker: Description of the algorithm," Tech. Rep., Intel Corporation, 1999.

[52] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision (darpa)," in *Proc. DARPA Image Unders. Workshop*, 1981, pp. 121–130.

[53] J. Barron, D. Fleet, and S. Beauchemin, "Performance of optical flow techniques," *Int. J. Comput. Vis.*, vol. 12, no. 1, pp. 43–77, 1994.

[54] B. Horn and B. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, pp. 185–203, 1981.
[55] H. Nagel, "On the estimation of optical flow: Relations between different approaches and some new results," *Artif. Intell.*, vol. 33, no. 3, pp. 298–324, 1987.
[56] J. Weickert, "On discontinuity-preserving optic flow," in *Proc. CVMR*, 1998, pp. 115–122.
[57] G. Aubert, R. Deriche, and P. Kornprobst, "Computing optical flow via variational techniques," *SIAM J. Appl. Math.*, vol. 60, no. 1, 1999.
[58] J. Weickert and C. Schnörr, "Variational optic flow computation with a spatio-temporal smoothness constraint," *Int. J. Comput. Vis.*, vol. 14, no. 3, pp. 245–255, 2001.
[59] L. Alvarez, J. Weickert, and J. Sanchez, "Reliable estimation of dense optical flow fields with large displacements," *Int. J. Comput. Vis.*, vol. 39, no. 1, pp. 41–56, 2000.
[60] E. Simoncelli, E. Adelson, and D. Heeger, "Probability distributions of optical flow," in *Proc. CVPR*, 1991, pp. 310–315.
[61] J. Weber and J. Malik, "Robust computation of optical flow in a multi-scale differential framework," *Int. J. Comput. Vis.*, vol. 14, no. 1, pp. 67–81, 1995.
[62] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods," *Int. J. Comput. Vis.*, vol. 61, no. 3, pp. 211–231, 2005.
[63] M. Langer and R. Mann, "Optical snow," *Int. J. Comput. Vis.*, vol. 55, no. 1, pp. 55–71, 2003.
[64] P. Anandan, "A computation framework and an algorithm for the measurement of visual motion," *Int. J. Comput. Vis.*, vol. 2, pp. 219–232, 1989.
[65] D. Fleet and A. Jepson, "Computation of component image velocity from local phase information," *Int. J. Comput. Vis.*, vol. 5, no. 1, pp. 77–104, 1990.
[66] J. Konrad and E. Dubois, "Bayesian estimation of motion vector fields," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 9, pp. 910–927, Sep. 1992.
[67] E. Memin and P. Perez, "Hierarchical estimation and segmentation of dense motion fields," *Int. J. Comput. Vis.*, vol. 46, no. 2, pp. 129–155, 2002.
[68] C. Stiller and J. Konrad, "Estimating motion in image sequences: A tutorial on modeling and computation of 2D motion," *IEEE Signal Process. Mag.*, vol. 16, no. 4, pp. 70–91, Jul. 1999.
[69] P. Bouthemy and P. Lalande, "Recovery of moving object masks in an image sequence using local spatiotemporal contextual information," *Opt. Eng.*, vol. 32, no. 6, pp. 1205–1212, 1993.
[70] M. Chang, M. Tekalp, and I. Sezan, "Simultaneous motion estimation and segmentation," *IEEE Trans. Image Process.*, vol. 6, no. 9, pp. 1326–1333, Sep. 1997.
[71] C. Stiller, "Object-oriented video coding employing dense motion fields," in *Proc. ICASSP*, 1994, pp. 273–276.
[72] J.-M. Odobez and P. Bouthemy, "Robust multiresolution estimation of parametric motion models," *J. Vis. Commun. Image Represent.*, vol. 6, no. 4, pp. 348–365, 1995.
[73] D. Murray and B. Buxton, "Scene segmentation from visual motion using global optimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 2, pp. 220–228, Feb. 1987.
[74] M. Tekalp, *Digital Video Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1995.
[75] A. Bovik, Ed., *Handbook of Image and Video Processing*. New York: Academic, 2000.
[76] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," presented at the IEEE Workshop on Stereo and Multi-Baseline Vision, 2001.
[77] R. Depommier and E. Dubois, "Motion estimation with detection of occlusion areas," in *Proc. ICASSP*, 1992, pp. 269–272.
[78] S. Ince and J. Konrad, "Geometry-based estimation of occlusions from video frame pairs," in *Proc. ICASSP*, 2005, vol. 2, pp. 933–936.
[79] A. Luo and H. Burkhardt, "An intensity-based cooperative bidirectional stereo matching with simultaneous detection of discontinuities and occlusions," *Int. J. Comput. Vis.*, vol. 15, no. 3, pp. 171–188, 1995.
[80] C. Strecha, R. Fransens, and L. Van Gool, "A probabilistic approach to large displacement optical flow and occlusion detection," in *Proc. ECCV Workshop SMVP*, 2004, pp. 71–82.
[81] D. Geiger, B. Ladendorf, and A. Yuille, "Occlusions and binocular stereo," *Int. J. Comput. Vis.*, vol. 14, no. 3, pp. 211–226, 1995.
[82] P. Fua, "A parallel stereo algorithm that produces dense depth maps and preserves image features," *Mach. Vis. Appl.*, vol. 6, pp. 35–49, 1993.
[83] K. Lim, A. Das, and M. Chong, "Estimation of occlusion and dense motion fields in a bidirectional bayesian framework," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 712–718, May 2002.
[84] G. Egnal and R. Wildes, "Detecting binocular half-occlusions: Empirical comparisons of five approaches," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 8, pp. 1127–1133, Aug. 2002.
[85] C. Chang, S. Chatterjee, and P. Kube, "On an analysis of static occlusion in stereo vision," in *Proc. CVPR*, 1991, pp. 722–723.
[86] L. Alvarez, R. Deriche, R. Papadopoulo, and J. Sanchez, "Symmetrical dense optical flow estimation with occlusion detection," in *Proc. ECCV*, 2002, pp. 721–735.
[87] D. Marr and T. Poggio, "Cooperative computation of stereo disparity," *Science*, vol. 194, no. 4262, pp. 283–287, 1976.
[88] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions via graph cuts," in *Proc. ICCV*, 1999, pp. 508–515.
[89] [Online]. Available: http://www.middlebury.edu/stereo2007
[90] [Online]. Available: http://www.gpgpu.org/2007

Pierre-Marc Jodoin received the M.Sc. and Ph.D. degrees in computer vision and image processing from the University of Montréal, Montréal, QC, Canada, in 2007.

He is an Engineer and Assistant Professor at the University of Sherbrooke, Sherbrooke, QC, Canada. He currently works at the Modélisation en Imagerie, Vision et Réseaux de Neurones (MOIVRE) Center in motion detection/segmentation, tracking, medical imagery, 3-D/4-D reconstruction, and segmentation.

Max Mignotte received the D.E.A. (postgraduate degree) in digital signal, image, and speech processing from the INPG University, Grenoble, France, in 1993 and the Ph.D. degree in electronics and computer engineering from the University of Bretagne Occidentale (UBO) and the Digital Signal Laboratory (GTS) of the French Naval Academy, in 1998.

He was an INRIA Postdoctoral Fellow at the University of Montréal (DIRO), QC, Canada, from 1998 to 1999. He is currently with DIRO at the Computer Vision and Geometric Modeling Lab as an Assistant Professor (Professeur Adjoint). He is also a member of the Laboratoire de recherche en Imagerie et Orthopédie (LIO), Centre de Recherche du CHUM, Hôpital Notre-Dame) and a researcher at CHUM. His current research interests include statistical methods and Bayesian inference for image segmentation (with hierarchical Markovian, statistical templates, or active contour models), hierarchical models for high-dimensional inverse problems from early vision, parameters estimation, tracking, classification, shape recognition, deconvolution, 3-D reconstruction and restoration problems.

Christophe Rosenberger received the Ph.D. degree from the University of Rennes I, France, in 1999.

He is an Assistant Professor at ENSI, Bourges, France, and works at the Laboratory of Vision and Robotics, Bourges, in the Signal, Image, and Vision Research Unit. His research interests include evaluation of image processing and quality control by artificial vision.