Smoothness constraints in recursive search motion estimation for picture rate conversion.

Citation for published version (APA):

Bartels, C. L. L., & Haan, de, G. (2010). Smoothness constraints in recursive search motion estimation for picture rate conversion. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(10), 1310-1319. <https://doi.org/10.1109/TCSVT.2010.2058474>

DOI:

[10.1109/TCSVT.2010.2058474](https://doi.org/10.1109/TCSVT.2010.2058474)

Document status and date:

Published: 01/01/2010

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

Smoothness Constraints in Recursive Search Motion Estimation for Picture Rate Conversion

Chris Bartels and Gerard de Haan

Abstract—Many motion compensation algorithms are based on block matching. The quality of the block correlation depends on the validity of the brightness constancy assumption and the assumption of fixed translational motion within a block. These assumptions are invalid in areas with texture changes, noise, lighting changes, and rapid deformations. Smoothness priors should enforce stable estimates in these regions by propagating neighboring estimates, while preserving hard object boundaries (piecewise smoothness). Most motion estimation algorithms that successfully implement these constraints are computationally complex. In this paper, we show an intuitive and computationally efficient way to implement them within the framework of (real-time) recursive search, targeting consumer-market embedded devices with limited resources.

Index Terms—Block matching, motion compensation (MC), motion estimation (ME), recursive search (RS).

I. INTRODUCTION

BLOCK matching motion estimation (ME) algorithms have become a defining factor in the video quality of modern displays. To improve the poor dynamic resolutions of flat panel displays, manufacturers are resorting to higher refresh rates (up to 240 Hz), motion blur reduction techniques [1], and motion compensation (MC) [2] in the video processing back-end [3]–[5]. These techniques require consistent motion vector fields (VFs) calculated in real-time on high definition (HD) content. Spatiotemporal prediction methods such as recursive search (RS) [6]–[12] have addressed this challenge and are currently implemented in many products. In this paper, we improve the RS block matching algorithm by focusing on VF smoothness, without compromising the real-time applicability.

ME methods are typically modeled by a set of energy functions that contain (in)direct luminance comparisons and smoothness constraints. The luminance comparisons are determined per region R (pixel, or block of pixels) by direct measurement, e.g., block correlation [13], phase-plane correlation [14], [15], or the outcome of the optic flow gradient

constraint equation [16]. The smoothness constraints limit the solution based on our assumptions; often a (piecewise) locally constant or a (piecewise) locally linear VF is assumed. When smoothness constraints are applied, these impose a connectedness that change the required optimization from local to global; the solution cannot be determined for an individual image patch as the smoothness energy depends on its neighbors. Instead, the full VF that minimizes the sum of the local energies over the frame needs to be optimized.

Finding the global optimum efficiently is a non-trivial, usually iterative, task. In optical flow methods, continuous optimization algorithms are popular, i.e., methods that use variational calculus where the Euler–Lagrange differential equation is solved [16]. This is an elegant approach as the motion field can be derived directly from the equation system. However, large displacements¹ require the use of warping schemes, e.g., [17], and sophisticated energy terms are needed to accurately model piecewise smoothness. The numerical methods that solve these large sets of (non)linear equations are more expensive than we judge affordable in our applications.

Discrete optimization algorithms can handle the non-convex minimization more directly. Algorithms for maximum *a posteriori* (MAP) inference on Markov random field (MRF) models, such as iterated conditional modes (ICM) [18], belief propagation [19], [20], and graph cuts (GC) [21], are well-known. These discrete methods approximate solutions to a label-assignment problem, where each node in the MRF (a pixel or block of pixels) must be assigned a label representing a motion vector. Due to the large (2-D) label space in ME² and the implementation complexity (both in memory size/bandwidth and operations count), these methods are generally not suitable for real-time applications. Methods such as the RS algorithm [6]–[12] can also be considered a form of discrete optimization. These block matching methods have a low implementation complexity due to the use of small, changing, spatiotemporal candidate sets (i.e., typically less than ten “labels” per node) in a simple recursive optimization. Random “update” candidates allow for a continuous adjustment of the label set in the optimization loop. Although these algorithms have become popular in video enhancement and conversion products, they rely mostly on *implicit* smoothness constraints.

¹The linearized brightness constancy assumption only holds for displacements smaller than 1 pixel.

²A limited 100 pixel search window with quarter pixel accuracy yields $1.6e5$ possible label assignments for each node.

Manuscript received March 11, 2009; revised December 9, 2009; accepted March 28, 2010. Date of publication July 15, 2010; date of current version October 8, 2010. This paper was recommended by Associate Editor Z. He.

C. Bartels is with the Electronic Systems Group, Eindhoven University of Technology, Eindhoven 5600MB, The Netherlands (e-mail: chris@ics.ele.tue.nl).

G. de Haan is with the Electronic Systems Group, Eindhoven University of Technology, Eindhoven 5600MB, The Netherlands, and also with Philips Research, Eindhoven 5656AA, The Netherlands (e-mail: g.de.haan@philips.com).

Digital Object Identifier 10.1109/TCSVT.2010.2058474

Implicit smoothness constraints are inherent to the algorithm and are not separately defined as a prior in the energy minimization, e.g., when a limited candidate set based on previous spatial or temporal estimates is evaluated. *Explicit* smoothness constraints are defined as a prior in the energy minimization, i.e., these construct the hidden variable node relations in the MRF network. Example constraints include the pairwise vector difference [22], robust constraints [23], anisotropic smoothness regularizers [24], or bilateral constraints [25]. Although the constraints and optimization algorithms vary greatly, they all have in common the objective to achieve piecewise smooth VFs. The algorithm should retain sharp object boundaries, while regulating areas where the brightness constancy assumption is invalid. The optimization algorithms that achieve this are typically complex and require many iterations.

In this paper, we show that the RS algorithm can be modified to attain piecewise smooth VFs in a computationally efficient way. We evaluate different implementations of smoothness priors and illustrate that a previous generation of the algorithm was based on constant flow assumptions [23]. The new approach based on “locally linear” flow priors greatly improves on this, with resulting VFs showing better consistency, while retaining sharp object boundaries. The tradeoffs in correlation quality and smoothness prior strength are illustrated. Furthermore, we show the relation to [11], which describes a related concept to improve the consistency of the RS algorithm, through the introduction of linear flow candidates in the candidate set mechanism.

After a short introduction of ME and the relation to MAP optimization in Section II, we describe the RS algorithm in Section III. Here, we present the original candidate set and energy minimization procedure. The linear flow-based RS is introduced in Section IV, which details the changes in candidate set and smoothness priors. Section V introduces the metrics used to compare the performance of different ME algorithms. The results and a discussion on the selection of the best tradeoff can be found in Section VI.

II. ME AND MAP OPTIMIZATION

A fundamental assumption underlying most ME algorithms is brightness constancy (1), which states that pixels retain the same luminance over their spatiotemporal displacement path

$$I(x, y, t) = I(x + \partial x, y + \partial y, t + \partial t). \quad (1)$$

In the above equation, $I(x, y, t)$ denotes a continuous approximation of the 3-D luminance pixel lattice.

For any region R of pixels, the motion (v^x, v^y) can be derived using gradient-based [16] or correlation-based [13] measurements, (2) and (3), respectively, as follows:

$$E_L^{\text{grad}} = \min_{v^x, v^y} \sum_{(x, y) \in R} \rho \left(\frac{\partial I(x, y, t)}{\partial x} v^x + \frac{\partial I(x, y, t)}{\partial y} v^y + \frac{\partial I(x, y, t)}{\partial t} \right) \quad (2)$$

$$E_L^{\text{corr}} = \min_{v^x, v^y} \sum_{(x, y) \in R} \rho(I(x + v^x \partial t, y + v^y \partial t, t + \partial t) - I(x, y, t)). \quad (3)$$

The gradient-based approaches minimize the optical flow constraint equation in which $\frac{\partial I(x, y, t)}{\partial x}$, $\frac{\partial I(x, y, t)}{\partial y}$, and $\frac{\partial I(x, y, t)}{\partial t}$ denote spatial and temporal derivatives in x , y , and t directions at a position (x, y, t) in the luminance grid. The correlation-based approaches are based on the minimization of the luminance differences between a region and its displaced counterpart. The cost function ρ determines the type of correlation, i.e., for $\rho(x) = x^2$, (3) denotes the sum of squared differences. For $\rho(x) = |x|$ the correlation term denotes the sum of absolute differences (SAD), e.g., [6]. The correlation-based approaches are generally more robust than the gradient-based approaches, as the former do not depend on linearizations and an accurate determination of the spatiotemporal gradients.

From here on we use a shorthand notation using bold font to denote vectors, e.g., $\mathbf{v} = \{v^x, v^y\}$, and a discrete temporal index $I_n, I_{n+1}, \dots$. The VF that is estimated, $\mathbf{D}_{n \rightarrow n+1}(\mathbf{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}^2$, contains the displacement vectors $(\mathbf{d} = \mathbf{v} \partial t)$ for a single frame pair. For example, the calculation of the SAD for a region R using the vectors in the forward VF can be abbreviated as follows:

$$\sum_{\mathbf{x} \in R} |I_n(\mathbf{x}) - I_{n+1}(\mathbf{x} + \mathbf{D}_n(\mathbf{x}))|. \quad (4)$$

Equations (2) and (3) denote a measurable quantity, the likelihood of the tested motion vector. ME methods that output the result of these measurements directly are denoted maximum likelihood methods. If certain constraints on the VF are added in the form of smoothness priors, the estimation can be modeled as a MAP method [22], [26].

The (posterior) probability of the VF, given any two concurrent frames, is given by Bayes rule and is to be maximized as

$$p(\mathbf{D} | I_n, I_{n+1}) = \frac{p(I_n | I_{n+1}, \mathbf{D}) p(\mathbf{D})}{p(I_n | I_{n+1})}. \quad (5)$$

The probability of a frame following another $p(I_n | I_{n+1})$ is assumed constant. The likelihood term, denoted by $p(I_n | I_{n+1}, \mathbf{D})$, can be modeled by a normal distribution $\exp(-E_L/\sigma_L)$ based on the correlation or gradient energy measurements E_L . The smoothness priors are included in $p(\mathbf{D})$; these are also modeled by a normal distribution $\exp(-E_S/\sigma_S)$. MAP methods seek to maximize the posterior probability as

$$\mathbf{D} = \arg \max_{\tilde{\mathbf{D}} \in \Theta} (p(I_n | I_{n+1}, \tilde{\mathbf{D}}) p(\tilde{\mathbf{D}})) \quad (6)$$

where a vector field $\mathbf{D}$ is selected out of a space Θ , which contains all possible VFs. This can be simplified to an energy minimization problem using the negative log of the probabilities as the following:

$$\mathbf{D} = \arg \min_{\tilde{\mathbf{D}} \in \Theta} (E_L(\tilde{\mathbf{D}}) + E_S(\tilde{\mathbf{D}})). \quad (7)$$

E_S is an explicit smoothness energy term based on comparisons with the motion vectors of direct (spatial) neighbors of a pixel, block, or region. In [23], a distinction is made between two types of smoothness priors: first order, locally constant flow constraints and second order, locally linear flow constraints (termed “affine flow” in [23]).

Let us use subscript notation l_x, l_y to denote the partial derivatives in x and y directions of the VF $\mathbf{D}$ at a position $\mathbf{x}$. Double subscripts denote second order partial derivatives, e.g., l_{xx} . The combined $(E_S^x + E_S^y)$ quadratic smoothness energies for linear flow constraints are denoted in (8) and (9), respectively, with superscripts referring to x and y components of the motion as follows:

$$E_S(\mathbf{x}) = l_x^2 + l_y^2 + l_x^2 + l_y^2 \quad (8)$$

$$E_S(\mathbf{x}) = l_{xx}^2 + l_{xy}^2 + l_{yx}^2 + l_{yy}^2 + l_{xx}^2 + l_{xy}^2 + l_{yx}^2 + l_{yy}^2. \quad (9)$$

The *constant flow* energy function is minimal if the motion of a pixel is identical to the motion of the spatial neighboring pixels, a zero slope. The *linear flow* constraint minimizes the spatial “acceleration” of the motion vectors; the preferred motion of a pixel is the average of its neighbors, a linear slope. The original RS algorithm, introduced in the following section, relies on a discrete version of a constant flow constraint.

III. RS-ME

The RS algorithm [6] sequentially maximizes the local conditional probabilities, by minimizing a variant of $E_L + E_S$. The components of the algorithm listed below are separately described in the following sections.

Algorithm 1 RS

For each block at spatial position $\mathbf{x}$ in the VF $\mathbf{D}$ (in a meandering scanning order):

- 1) **select** a vector candidate set, $C(\mathbf{x})$, from vectors in the spatiotemporal environment of the current block.
- 2) **for each** candidate $\mathbf{c}_i$ in $C(\mathbf{x})$
 - a) **determine** the likelihood term $\text{SAD}(\mathbf{c}_i)$;
 - b) **determine** the smoothness prior (penalty) $\text{Pen}(\mathbf{c}_i)$.
- 3) **assign** the candidate with minimal energy to the current block: $\mathbf{D}(\mathbf{x}) = \arg \min_{\mathbf{c}_i \in C} \text{SAD}(\mathbf{c}_i) + \lambda \text{Pen}(\mathbf{c}_i)$.

The algorithm above may be repeated over the same frame pair, e.g., upward and downward scans over the block grid, until convergence is reached. Multiple iterations over the same frame pair improve convergence, but the number of iterations can be reduced in the case of video. The use of motion vectors from the previous frame $\mathbf{D}_{n-1}$ in the candidate set of the current estimation $\mathbf{D}_n$ accelerates convergence. Due to this “temporal” convergence, one or two iterations per frame turn out to be sufficient and are used in real-time implementations.

The algorithm as outlined above has similarities with the ICM method [18], [22]. The key element in both methods is the iterative maximization of the local conditional probabilities. The main differences are the candidate set mechanism and the way new vectors are iteratively assigned. In ICM methods, the newly determined vector for a block is assigned at the end of each frame iteration; hence, it does not modify the local conditional probabilities at the next block during the current spatial iteration. The RS does modify these probabilities as it assigns a new vector immediately after processing a block. We see significantly faster convergence due to this (aggressive) spatial iteration.

Fig. 1. (a) Recursion through all blocks in the VF by meandering scanning order. For each block, candidate vectors from neighboring blocks (within a spatial distance S from the current block, indicated by the bold line) are evaluated ([11] and [12]). Dark gray blocks have been processed in the current pass and can be used as “spatial” candidates, and light gray blocks have motion vector values from either a previous estimate or previous passes and are “temporal” (or “iteration”) candidates. In practice, it is sufficient to select a subset of neighboring vectors. (b) Reduced candidate structure for the reference RS implementation. (c) Reduced candidate structure for the simple block-based prediction [11].

A. Candidate Set

The key component in the algorithm is the selection of the candidate set C . By keeping this set small, few luminance comparisons are required. By keeping the vectors in this set, close to the vectors in the neighboring blocks, an implicit smoothness constraint results. As objects in the video sequence are larger than blocks, the “correct” candidate motion vectors will often be passed on via neighboring blocks and reside in the set C .

Typically, the candidate set consists of spatial, temporal, and update candidates:

$$C(\mathbf{x}) = \left\{ \begin{array}{l} C_{\text{spat}}(\mathbf{x}) \\ C_{\text{temp}}(\mathbf{x}) \\ C_{\text{upd}}(\mathbf{x}) \end{array} \right\}. \quad (10)$$

Spatial and temporal candidates are vectors from the spatiotemporal neighborhood. These can be vectors from neighboring blocks within a spatial distance less than S from the current block

$$C_{\text{spat}}(\mathbf{x}) = \left\{ \mathbf{D}_n \left(\mathbf{x} + \begin{pmatrix} x \\ y \end{pmatrix} \right) \mid -S \leq \frac{x}{y} \leq S \right\} \quad (11)$$

$$C_{\text{temp}}(\mathbf{x}) = \left\{ \mathbf{D}_{n-1} \left(\mathbf{x} + \begin{pmatrix} x \\ y \end{pmatrix} \right) \mid -S \leq \frac{x}{y} \leq S \right\} \quad (12)$$

or a subset of the above, examples of which are illustrated in Fig. 1(b) and (c). As the RS algorithm scans through the block grid, some of the neighboring blocks contain motion vectors determined in the current pass (spatial candidates) and some contain motion vectors that have been determined, either in the previous passes on the current frame or the motion estimates from the previous frame (temporal candidates). The limited set $\{S_1, S_2, T_1\}$ of spatiotemporal predictions outlined in Fig. 1(b) is used throughout this paper. Note that a very

Fig. 2. Update candidates are selected based on multiple normal distributions, the Gaussian means determined by the spatial and temporal candidates. The above example shows this for one dimension, e.g., the v_x vector component.

similar structure, Fig. 1(c), is used in the simple block-based prediction method [11].

The update candidates are created by the addition of normal distributed noise to the spatiotemporal candidates as follows:

$$C_{\text{upd}}(\mathbf{x}) = \left\{ \mathbf{x} + \begin{pmatrix} n_x \\ n_y \end{pmatrix} \mid \mathbf{x} \in \{C_{\text{spat}}(\mathbf{x}), C_{\text{temp}}(\mathbf{x})\}, n_{x,y} \sim N(0, \sigma_{x,y}^2) \right\}. \quad (13)$$

The probability density functions for the update candidates are illustrated in Fig. 2. By means of the standard deviation σ , and the number of update candidates, the implicit smoothness can be controlled. For a very large number of update candidates, and a large σ , the method degenerates to a full search. In our implementations, two update candidates are used, with pseudorandom update “noise” values cyclically fetched from a lookup table. The noise table can be tuned toward certain content properties (e.g., horizontal accelerations that are typically larger than vertical, $\sigma_x > \sigma_y$).

B. Energy Minimization

The next step encompasses the maximization of the local conditional probabilities, by minimization of the energies $E_L + E_S = \text{SAD}(\mathbf{c}_i) + \lambda \text{Pen}(\mathbf{c}_i)$. The SAD (3) using $\rho(x) = |x|$, models the likelihood. The prior, E_S , is typically modeled by a fixed penalty system; spatial candidates get a lower fixed energy than temporal candidates, which get a lower fixed energy than update candidates ($P_{\text{spat}} < P_{\text{temp}} < P_{\text{upd}}$) [6]. The penalty system is an explicit smoothness constraint that reduces the selection of the update candidates, as these are the cause of most change in the VF:

$$P_i(\mathbf{c}_i) = \begin{cases} P_{\text{spat}} & \text{if } \mathbf{c}_i \text{ from } C_{\text{spat}}(\mathbf{x}) \\ P_{\text{temp}} & \text{if } \mathbf{c}_i \text{ from } C_{\text{temp}}(\mathbf{x}) \\ P_{\text{upd}} & \text{if } \mathbf{c}_i \text{ from } C_{\text{upd}}(\mathbf{x}) \end{cases}. \quad (14)$$

If the estimator is in a non-converged state, then, at each block in the spatial iteration, some of the update candidates will have a lower total energy than the candidate vector from the previous block (spatial candidate): $\text{SAD}(\mathbf{c}_{\text{upd}}) + P_{\text{upd}} < \text{SAD}(\mathbf{c}_{\text{spat}}) + P_{\text{spat}}$. The output of the estimator is noisy, as many update candidates are chosen, until at some point an update candidate is similar to the “correct” vector value for a block. After this point, the update candidates will not be able to (significantly) improve on the spatial candidate and the spatial candidates will be chosen as $\text{SAD}(\mathbf{c}_{\text{upd}}) + P_{\text{upd}} > \text{SAD}(\mathbf{c}_{\text{spat}}) + P_{\text{spat}}$. This allows for a very fast spatial spreading of the correct spatial (or temporal) motion vector. Only when an improvement by using an update candidate is a sufficiently large step over the threshold defined by the

Fig. 3. (a) “Smooth” transition in a VF. (b) Estimated result with update steps visualized. Vector that is chosen per block is either a spatial or an update candidate. The candidate system with a fixed penalty smoothness prior implicitly yields a segmented result.

penalties, e.g., $\text{SAD}(\mathbf{c}_{\text{spat}}) - \text{SAD}(\mathbf{c}_{\text{upd}}) > P_{\text{upd}} - P_{\text{spat}}$, the update is chosen, causing the VF to change.

This mechanism essentially resembles a crude constant flow constraint. Although the spatial gradients are not explicitly calculated as in (8), the motion is constrained to be constant over a spatial region. This becomes visible in the output of the estimator; the VF is separated into islands of constant motion (Fig. 3). There is a tradeoff in $P_{\text{upd}} - P_{\text{spat}}$, smaller values will decrease the smoothness, and the erroneous clustering effect, but at the cost of a noisier output VF. In addition, a constant flow smoothness constraint results inherently from the limited set of motion vector candidates; the spatial candidates bias the estimator toward a constant motion field. Example VFs from the RS estimator with this penalty mechanism are shown in Fig. 8(b). The clustering is particularly visible in the motion VF of the wheel sequence.

IV. LINEAR FLOW RS

To combat the clustering artifact described above, we investigated the use of locally linear candidate sets and smoothness priors. The modifications in the candidate set are directed at making available linear flow candidates in the estimation process, the modifications in the prior at selecting them. In (9), the second order spatial gradients are minimized; the energy function becomes zero if the location in the VF exhibits a linear spatial slope. To accomplish a locally linear flow smoothness, without determining (9) explicitly, we add priors that penalize the distance to the neighborhood vectors.

A. Candidate Sets

The “correct” vector for the currently processed block can only be selected if it is available in the candidate set. We can guarantee the availability of this vector by including a large random update set. However, this is computationally expensive and results in noisy output if there is no explicit smoothness prior.³ Therefore, we calculate some of the linear-flow candidates explicitly, and limit the increase of the candidate set. Such an approach is similar to [11], where the use of temporal gradient compensated candidates is described. The spatial VF gradient in the direction of the spatial candidate position is determined in the previous VF ($\mathbf{D}_{n-1}$), and used to refine the current spatial candidates in $\mathbf{D}_n$. This approach can be categorized as a temporal extrapolation.

In addition, we investigate the use of different average candidate structures, such as the directional averages, illustrated

³Remember, we lose the implicit smoothness constraint provided by a limited candidate set mechanism.

Fig. 4. Different filter structures calculating averages: box filter of size N and directional average regions a1–a4 around the current block C.

in Fig. 4. These combine both blocks from the current scan and the previous temporal/iterative⁴ scans (implicitly the gradient is calculated using both current and previous iteration). This approach resembles more closely a spatiotemporal or spatioiterative interpolation. Furthermore, we expect it to be more robust than [11] as more blocks are included in the computation of the average candidate.

Table I lists the different candidate modes tested, with varying linear-flow candidates. It denotes the number of candidates per type, as well as the size of the “filter” region for the linear-flow candidates (i.e., the width and height of the region N in Fig. 4 expressed in the number of blocks). Mode A denotes the reference candidate structure as illustrated in Fig. 1(b). Modes B and C denote candidate sets with added box averages, calculated over the region N , as illustrated in Fig. 4. (Note that Mode C adds three box averages with different sizes as indicated in Table I.) Modes D and E denote sets with four “directional” averages, calculated over the regions a1–a4, as illustrated in Fig. 4. Mode F lists the temporal motion-compensated strategy [11] applied on the reference candidate structure. Mode G denotes a modification where instead of vectors from the previous VF ($\mathbf{D}_{n-1}$), vectors from the previous iteration ($\mathbf{D}_n$) are used to compute the gradient-compensated motion vectors.

B. Smoothness Priors

The selection of the final candidate for a block is, next to the likelihood term and the candidate set, influenced by the smoothness prior. We outline simple smoothness priors, with a focus on “locally linear flow” preference and low computational cost.⁵

To illustrate the effect of different priors, we calculated the penalty value for two example regions with motion distributions as in Fig. 5(a) and (b). The regions show two clusters, for a single component of the motion vector, at some point in the estimation process. If we assume that the estimated VF is almost converged and closely reflects the final distribution, we can discern three likely values for the motion vector component of the currently processed (middle) block. It should either correspond with one of the two clusters or take the value of the “cross-cluster” average. The

⁴We use the term iterative, or iterative candidates, if multiple scans are used in the ME algorithm; in this case, the temporal candidates (from the VF of the previous frame) are replaced by vectors estimated on the current frame.

⁵We try to achieve sharp motion boundaries without the use of more complex anisotropic smoothness priors or image-driven regularization.

TABLE I
DIFFERENT CANDIDATE SETS

Cand. Mode	# Spat. Cand.	# Temp. Cand.	# Flow Cand.	Flow Type	Flow Cand. Region Size
A	2	1	—	—	—
B	2	1	1	Box avg.	9
C	2	1	3	Box avg.	3, 9, 15
D	2	1	4	Dir. avg.	3, 3, 3, 3
E	2	1	4	Dir. avg.	9, 9, 9, 9
F	2	1	3	Temp. comp.	3, 3, 5
G	2	1	3	Iter. comp.	3, 3, 5

smoothness prior must bias the candidate selection toward these values, however, the bias should be weak enough to allow convergence and prevent over-smoothing.

The graphs in Fig. 5(c) and (d) show the penalty value over the candidate range for the different priors detailed in the remainder of this section. Note that the offset of the curves has no influence on the motion estimators. A system without penalty can be envisioned as a level line; all candidates have equal prior probability of being chosen. The final selection is influenced only by the likelihood term. Greater variation in the prior value negatively affects the convergence performance⁶ of the estimator, but positively affects spatial coherency.

P_I , in (14), denotes the fixed penalties introduced in [6]. All candidate values get equal penalties, with the exception of spatial and temporal candidates. The fixed penalty value for an update candidate promotes convergence speed, while the lower penalty values for the spatial and temporal candidates affect spatial coherency. Although this penalty system allows for a fast convergence, it behaves much like a constant flow constraint.

P_{II} , in (15), describes a prior that biases the estimation toward the average vector value of the region surrounding the current block as follows:

$$P_{II}(\mathbf{c}_i) = \rho(\mathbf{c}_i, \frac{1}{|N|} \sum_{\mathbf{x} \in N} \mathbf{D}(\mathbf{x})) \quad (15)$$

where $|N|$ denotes the cardinality of the set of blocks in the region N , and ρ denotes the vector distance function, e.g., Manhattan, Euclidean, or squared vector difference. If a region contains two motion vector clusters, the prior prefers candidates that smooth out the two clusters. The penalty does not prevent over-smoothing, the likelihood component of the energy function should be sufficiently strong to preserve object edges.

P_{III} , in (16), denotes a prior that sums the distances of the candidate vector to the region vectors individually as follows:

$$P_{III}(\mathbf{c}_i) = \frac{1}{|N|} \sum_{\mathbf{x} \in N} \rho(\mathbf{c}_i, \mathbf{D}(\mathbf{x})). \quad (16)$$

P_{III} behaves different from P_{II} ; instead of a single minimum at the average value of the region, the prior is relatively flat for candidate values between the two cluster centers. The prior has the lowest value when the candidate is similar to the dominant (largest) cluster in the spatial region. This explains an edge preserving, clustering, quality. However, if a region is perfectly symmetrical, the penalty becomes flat and candidate values

⁶Here, convergence performance is defined as the ability and the time it takes to find new object motions.

Fig. 5. Two example blocks (marked with thick edges) and their smoothness prior value over the candidate vector range, for one component of the motion vector. (a) Asymmetric region with two clusters of MVs uneven size. (b) Symmetric region. (c) and (d) Smoothness prior values for the non-symmetric and symmetric region.

that lie between the two cluster centers have an equal prior probability. This enables the selection of either “cross-cluster” or cluster candidates, depending on their relative correlation component strength.

P_{IV} and P_V , in (17) and (18), show priors that have a stronger bias toward the cluster that contains the currently processed block. Both take a number of directional averages as illustrated in Fig. 4. The assumption is made that, in most of the regions, one directional average will lie fully within a cluster (alongside the cluster-edge) and one directional average will lie orthogonal to the cluster-edge. P_{IV} determines the penalty by the distance between the candidate and the closest cluster average. The example has two minima that lie at the dominant cluster center and at the cross-cluster average, respectively, as follows:

$$P_{IV}(\mathbf{c}_i) = \min(\rho(\mathbf{c}_i, \frac{1}{|a1|} \sum_{\mathbf{x} \in a1} \mathbf{D}(\mathbf{x}), \rho(\mathbf{c}_i, \frac{1}{|a2|} \sum_{\mathbf{x} \in a2} \mathbf{D}(\mathbf{x}), \rho(\mathbf{c}_i, \frac{1}{|a3|} \sum_{\mathbf{x} \in a3} \mathbf{D}(\mathbf{x}), \rho(\mathbf{c}_i, \frac{1}{|a4|} \sum_{\mathbf{x} \in a4} \mathbf{D}(\mathbf{x}))) \quad (17)$$

$$P_V(\mathbf{c}_i) = \min(\frac{1}{|a1|} \sum_{\mathbf{x} \in a1} \rho(\mathbf{c}_i, \mathbf{D}(\mathbf{x})), \frac{1}{|a2|} \sum_{\mathbf{x} \in a2} \rho(\mathbf{c}_i, \mathbf{D}(\mathbf{x})), \frac{1}{|a3|} \sum_{\mathbf{x} \in a3} \rho(\mathbf{c}_i, \mathbf{D}(\mathbf{x})), \frac{1}{|a4|} \sum_{\mathbf{x} \in a4} \rho(\mathbf{c}_i, \mathbf{D}(\mathbf{x}))). \quad (18)$$

P_V applies the methodology of P_{III} on the directional averages. The example contains one minimum, for candidates whose values are similar to the dominant cluster value.

V. TEST METHODOLOGY

To measure the performance of different ME algorithms, we use a combination of two different evaluation criteria: the modified mean square error (M2SE) metric and the spatial inconsistency (SI) metric.

The M2SE metric [6] is a correlation metric that uses the estimated motion between I_n and I_{n+1} , the VF $\mathbf{D}_{n,n+1}$, to create a motion-compensated image using I_{n-1} and I_{n+1} at temporal position n . The mean square error (MSE) is calculated between

Fig. 6. The M2SE metric temporally extends the estimated VF, $\mathbf{D}_{n,n-1}$ is created by reversing $\mathbf{D}_{n,n+1}$. $\mathbf{D}_{n,n-1}$ and $\mathbf{D}_{n,n+1}$ are used to compute a motion-compensated image at position n , which is compared with the original image I_n .

this motion-compensated image I_{mc} and the original image I_n . The M2SE metric yields lower values if motion vectors align with the true object motion, as it uses a temporal extension (Fig. 6). Essentially, $\mathbf{D}_{n,n-1}$ is created by mirroring $\mathbf{D}_{n,n+1}$ and their combined validity is tested as follows:

$$\text{M2SE}(\mathbf{D}_{n,n+1}) = \sum_{\mathbf{x} \in I} I_n(\mathbf{x}) - I_{mc}(\mathbf{x}) \quad (19)$$

$$I_{mc}(\mathbf{x}) = \frac{1}{2} I_{n-1}(\mathbf{x} - \mathbf{D}_{n,n+1}(\mathbf{x})) + \frac{1}{2} I_{n+1}(\mathbf{x} + \mathbf{D}_{n,n+1}(\mathbf{x})). \quad (20)$$

SI (21) is the inverse of the global VF smoothness. It is determined by summation of the differences of the vector at each block with vectors of eight direct neighboring blocks. The value is averaged over the total number of blocks. $|\mathbf{D}|$ denotes the cardinality of the VF block grid and ρ the vector distance function:

$$\text{SI}(\mathbf{D}) = \frac{1}{8|\mathbf{D}|} \sum_{\mathbf{x} \in I^{bg}} \sum_{\substack{-1 \leq u \leq 1 \\ -1 \leq v \leq 1 \\ (u,v) \neq (0,0)}} \rho(\mathbf{D}(\mathbf{x}), \mathbf{D}(\mathbf{x} + \begin{pmatrix} u \\ v \end{pmatrix})). \quad (21)$$

In the evaluation of different motion estimators, we jointly plot the M2SE and SI values. Each point in the graphs of Fig. 7 denotes the M2SE and SI performance for a different motion estimator, averaged over a set of frames from different sequences [Fig. 8(a)]. The plots illustrate the tradeoff between motion estimator smoothness and correlation quality. The “optimal”

Fig. 7. Result plots with M2SE vs. SI for different estimator parameters, penalty and candidate modes. Motion estimators with values near the lower left corner perform better in both respects. The results are averages over the video sequences as in Fig. 8(a). All plots use the same scale, except (a) and (b).

estimators can be found in the bottom left corners, where a good performance is achieved with respect to both metrics.

VI. RESULTS

In the following sections, we outline the results for different candidate modes and smoothness priors, using the metrics described in Section V. We also evaluate the proposed methodology in the Middlebury optical flow benchmark [27].

A. M2SE vs. SI

The results are outlined in Fig. 7. We distinguish two modes of operation: either the motion estimator starts from a zero VF (unconverged content) or a fixed penalty-type estimator (P_1) is

used to provide an initial converged VF (converged content). This is indicated in the captions of the graphs.

For reference, Fig. 7(a) shows the M2SE and SI values of some existing block matching methods, i.e., Full Search, Diamond Search [28], Hexagonal Search [29], and predictive motion vector field adaptive search technique (PMVFAST) [30] (with early termination disabled). Note that the M2SE metric favors “true” VFs, e.g., PMVFAST outperforms a Full Search in M2SE value, whereas this is never the case for the MSE metric.

The performance of different candidate sets in the RS algorithm, without explicit smoothness constraints, is outlined in Fig. 7(b). The addition of different types of average candidates alters the implicit estimator smoothness. The figure shows a

Fig. 8. Images of the vector fields. On the left the block matching results from the HD sequences that were used to compute the M2SE and SI metrics, on the right the result from the Middlebury benchmark. Left: (a) four HD test sequences that contain fast arm movements, rotating wheels and scrolling text overlays, (b) vector field from RS with fixed penalties P_I , (c) vector field from RS with penalty type P_{III} . As the vector fields have a high dynamic range, the visuals luminance values were clipped and contrast-adjusted to emphasize irregularities. The motion boundaries are sharp, but soft transitions appear in some occlusion areas as the motion vector fields have not yet been corrected for occlusion (e.g., left arm). Right: results on the Middlebury benchmark sequences. (d) “Army” sequence, (e) “Mequon” sequence, (f) “Wooden” sequence, with on the right side the ground truth vector fields. Occlusion areas are indicated with black in the ground truth images.

small SI and M2SE performance benefit for all modes with added averages (candidate Modes B–E). The temporal refinement strategy (Mode F) shows an increase in inconsistency, potentially caused by an unstable gradient determination due to the use of the previous VF. If we modify this strategy to iteratively use the gradient in the current VF instead (Mode G), an improvement is visible. The performance of this mode is comparable with the directional average candidate modes (Modes D and E).

Explicit smoothness priors have a much larger influence on performance. Fig. 7(c) shows various smoothness priors with the gain factor λ varied over a logarithmic scale to illustrate the smoothness/correlation tradeoff curves. We use the best performing candidate mode (Mode D) in all tests. Note that for gain factors approaching zero, all estimators converge to the same point in the result space. That is a system without penalty, similar to candidate Mode D in Fig. 7(b). For high gain factors, the correlation quality quickly declines (sharp M2SE increase), with little additional smoothness benefit. “Optimal” estimators, which represent the best tradeoff in smoothness and correlation quality, lie in the bottom left corner. The gain factor λ for fixed (P_I) and no penalty systems is not varied; these are plotted as single points in the figure. Fig. 7(d) and (e) shows variations on Fig. 7(c), with more iterations⁷ and a larger set of update candidates, respectively.

The figures show that for multiple iterations P_{II} , P_{III} , and P_{IV} perform well. For a single iteration, the curves are spaced further apart and P_{III} performs best in SI. This is possibly related to the relatively flat penalty shape, as described in Section IV, which enables faster convergence in comparison to P_{II} . More iterations and update candidates further improve performance and drive most curves closer together.

For the best performing prior P_{III} , the influence of the number of iterations and the number of update candidates is illustrated in Fig. 7(f) and (g). These plots show that the relative performance gain of multiple iterations and larger update sets is limited. Note that for an increase in the number of update candidates, the algorithms that rely on implicit smoothness (low penalty gains λ) decrease in performance; the curves stretch to the right [Fig. 7(g)].

The influence of the smoothness prior aperture (N) is limited. Fig. 7(h) plots for a single smoothness prior, P_{III} , the influence of the aperture. Only for very large or small apertures, the performance decreases. The curve for filter aperture 3 is notably short; even for very large smoothness gains, filtering with a small aperture has limited effect on the correlation performance.

Fig. 8 provides a visual impression of the VFs for P_I and an “optimal” P_{III} . The smoothness improvement according to the VF inconsistency metric, Fig. 7(d), is approximately 50%. The improvement in the VFs is visible in areas with little or no texture (arm, clothing), strong VF gradients (rotating board),

⁷An iteration denotes an upward and a downward scan over the block grid.

Average end-point error	avg. rank	Army (Hidden texture)			Mequon (Hidden texture)			Schefflera (Hidden texture)			Wooden (Hidden texture)			Grove (Synthetic)			Urban (Synthetic)			Yosemite (Synthetic)			Teddy (Stereo)				
		GT im0 im1			GT im0 im1			GT im0 im1			GT im0 im1			GT im0 im1			GT im0 im1			GT im0 im1							
		all	disc	untest	all	disc	untest	all	disc	untest	all	disc	untest	all	disc	untest	all	disc	untest	all	disc	untest					
Adaptive [25]	5.2	0.69	0.26	0.06	0.23	0.75	0.18	0.54	1.19	0.21	0.18	0.81	0.10	0.84	1.25	0.73	0.83	1.26	0.31	0.14	0.16	0.13	0.22	0.10	0.53	1.37	0.79
Complementary OF [26]	6.8	0.11	0.28	0.10	0.18	0.63	0.12	0.31	0.75	0.18	0.19	0.87	0.12	0.87	1.31	1.00	1.78	1.73	1.00	0.77	0.15	0.22	0.10	0.84	1.48	0.95	
Aniso. Huber-L1 [27]	7.2	0.10	0.28	0.08	0.31	0.68	0.10	0.56	1.13	0.29	0.20	0.92	0.13	0.84	1.20	0.70	0.39	1.23	0.28	0.17	0.15	0.27	0.18	0.84	1.36	0.79	
DPOF [28]	7.4	0.13	0.35	0.09	0.25	0.79	0.17	0.24	0.49	0.21	0.18	0.62	0.15	0.74	1.09	0.49	0.68	1.80	0.63	0.19	0.16	0.36	0.25	0.50	1.08	0.55	
Spatially variant [21]	7.9	0.10	0.27	0.08	0.22	0.75	0.19	0.43	1.00	0.18	0.19	0.87	0.10	0.85	1.41	1.16	0.98	1.61	0.43	0.13	0.11	0.28	0.18	0.96	1.72	1.21	
TV-L1-improved [19]	8.8	0.09	0.26	0.07	0.20	0.71	0.16	0.53	1.18	0.22	0.21	1.24	0.11	0.90	1.31	0.72	0.51	1.93	0.84	0.18	0.17	0.31	0.26	0.73	1.82	1.07	
Multiscale MRF [23]	9.3	0.11	0.26	0.11	0.19	0.63	0.17	0.24	0.49	0.19	0.24	1.13	0.15	0.79	1.10	0.72	0.47	1.60	0.85	0.28	0.19	0.31	0.29	0.78	1.13	1.09	
CBF [14]	9.3	0.10	0.28	0.09	0.34	0.80	0.37	0.43	0.95	0.26	0.21	1.14	0.13	0.90	1.27	0.82	0.41	1.23	0.30	0.23	0.19	0.39	0.24	0.76	1.56	1.02	
Occulsion bounds [28]	9.9	0.11	0.32	0.10	0.29	0.94	0.24	0.39	0.93	0.26	0.22	1.10	0.12	1.10	1.37	1.50	0.93	1.77	0.57	0.10	0.14	0.11	0.87	1.71	1.13	1.06	
Rannacher [29]	10.3	0.11	0.31	0.09	0.25	0.84	0.21	0.57	1.27	0.26	0.24	1.32	0.17	0.91	1.33	0.72	0.49	1.95	0.78	0.15	0.14	0.26	0.89	1.58	0.86		
Brox et al. [7]	10.4	0.11	0.32	0.11	0.27	1.03	0.22	0.39	0.94	0.24	0.24	1.25	0.13	1.10	1.39	1.43	0.89	1.77	0.55	0.10	0.13	0.11	0.91	1.83	1.13	1.15	
Second-order prior [10]	10.7	0.11	0.31	0.09	0.26	0.93	0.20	0.57	1.25	0.26	0.20	1.04	0.12	0.94	1.34	0.83	0.61	1.93	0.47	0.20	0.16	0.34	0.22	0.77	1.64	1.10	
F-TV-L1 [17]	10.8	0.14	0.35	0.14	0.34	1.08	0.26	0.59	1.19	0.26	0.27	1.36	0.16	0.90	1.30	0.76	0.54	1.62	0.36	0.13	0.15	0.20	0.68	1.56	0.66		
Fusion [8]	11.6	0.11	0.34	0.10	0.19	0.69	0.16	0.29	0.66	0.23	0.20	1.19	0.14	1.07	1.42	1.22	1.35	1.49	0.86	0.20	0.20	0.26	0.13	1.07	1.20	0.79	
Dynamic MRF [9]	14.0	0.12	0.34	0.11	0.22	0.89	0.16	0.44	1.13	0.20	0.24	1.29	0.14	1.11	1.52	1.13	1.54	2.37	0.93	0.13	0.12	0.31	0.20	1.27	2.33	1.86	
SeqOF [12]	14.3	0.15	0.36	0.10	0.57	1.16	0.59	0.68	1.24	0.64	0.32	1.86	0.26	1.81	2.50	1.47	1.63	2.09	0.96	0.08	0.13	0.12	0.70	1.50	0.69		
Bartels [30]	15.5	0.14	0.35	0.15	0.35	1.01	0.37	0.51	1.09	0.35	0.41	1.51	0.34	0.96	1.36	0.93	0.99	1.53	0.68	0.29	0.24	0.26	0.17	0.66	1.70	1.04	
Learning Flow [13]	16.2	0.11	0.32	0.08	0.29	0.99	0.23	0.55	1.24	0.29	0.36	1.56	0.25	1.25	2.64	1.41	1.55	2.32	0.85	0.14	0.11	0.24	0.13	1.09	1.90	1.27	
Filter Flow [22]	17.4	0.17	0.39	0.13	0.43	1.09	0.38	0.75	1.34	0.78	0.70	2.54	0.68	1.13	2.01	1.51	0.57	1.32	0.44	0.22	0.23	0.26	0.13	0.96	1.66	1.12	
GraphCuts [16]	18.0	0.16	0.38	0.14	0.59	1.21	0.46	0.56	1.07	0.64	0.26	1.14	0.17	0.96	1.35	1.04	1.25	2.29	1.79	0.22	0.17	0.43	0.26	1.22	2.05	1.78	
Black & Anandan [3]	18.6	0.18	0.42	0.19	0.58	1.31	0.50	0.95	1.58	0.70	0.49	2.11	0.45	1.08	1.42	1.22	1.43	2.28	0.83	0.15	0.17	0.17	0.16	1.11	2.08	1.30	
SPSA-learn [15]	19.1	0.18	0.45	0.17	0.57	1.32	0.51	0.84	1.50	0.72	0.52	2.64	0.49	1.12	1.42	1.39	1.75	2.14	1.06	0.13	0.13	0.19	0.17	1.32	2.08	1.73	
GroupFlow [11]	19.4	0.21	0.51	0.21	0.79	1.69	0.72	0.86	2.84	0.74	0.30	1.07	0.26	1.29	2.81	0.82	1.94	2.30	1.26	0.11	0.14	0.19	0.17	1.08	1.96	1.35	
2D-CLG [3]	21.5	0.28	0.62	0.21	0.67	2.21	0.70	1.12	3.80	0.99	0.77	2.06	0.26	1.23	3.22	1.52	1.62	3.61	1.54	0.10	0.11	0.16	0.15	1.38	2.28	1.83	

(a)

Average normalized interpolation error	avg. rank	Mequon (Hidden texture)			Schefflera (Hidden texture)			Urban (Synthetic)			Teddy (Stereo)			Backyard (High-speed camera)			Basketball (High-speed camera)			Dumpruck (High-speed camera)			Evergreen (High-speed camera)		
		im0 GT im1			im0 GT im1			im0 GT im1			im0 GT im1			im0 GT im1			im0 GT im1			im0 GT im1					
		all	disc	untest	all	disc	untest	all	disc	untest	all	disc	untest	all	disc	untest	all	disc	untest	all	disc	untest			
Spatially variant [21]	4.4	0.61	0.84	0.64	0.79	1.11	0.64	0.93	1.15	0.73	0.99	1.02	1.22	1.04	1.05	1.15	1.05	1.37	1.03	0.67	1.21	0.68	0.95	1.42	0.63
Aniso. Huber-L1 [27]	6.8	0.62	0.80	0.68	0.84	1.13	0.69	1.03	1.44	0.93	0.97	1.03	1.26	1.06	1.09	1.15	1.08	1.46	1.03	0.64	1.12	0.66	0.99	1.48	0.63
CBF [14]	7.1	0.61	0.79	0.68	0.77	1.07	0.69	1.00	1.50	0.90	0.98	1.02	1.31	0.99	0.96	1.18	1.05	1.33	1.06	0.80	1.59	0.74	0.89	1.29	0.67
Second-order prior [10]	7.2	0.61	0.78	0.68	0.80	1.11	0.64	1.05	1.85	0.97	0.96	1.04	1.21	1.05	1.07	1.15	1.05	1.38	1.05	0.69	1.28	0.65	1.00	1.20	0.66
Brox et al. [7]	9.3	0.67	1.04	0.65	0.72	1.02	0.63	0.95	1.34	0.83	0.98	0.99	1.24	1.02	1.02	1.15	1.20	1.78	1.11	0.67	1.38	0.66	0.86	1.23	0.62
Bartels [30]	9.9	0.66	1.05	0.69	0.63	1.12	0.67	1.00	1.46	0.96	1.03	1.16	1.30	1.03	1.02	1.17	1.18	1.69	1.09	0.62	1.65	0.77	0.85	1.23	0.65
Occulsion bounds [28]	10.0	0.67	1.04	0.64	0.72	1.02	0.64	1.06	1.33	0.97	0.97	0.99	1.24	1.04	1.04	1.15	1.20	1.77	1.09	0.63	1.60	0.64	0.86	1.26	0.62
F-TV-L1 [17]	10.8	0.67	1.09	0.68	0.68	1.15	0.70	0.97	1.51	0.86	1.01	1.08	1.28	1.03	1.04	1.14	1.04	1.31	1.06	0.65	1.73	0.79	1.07	1.61	0.63
Multiscale MRF [23]	10.7	0.61	0.84	0.64	0.65	0.91	0.61	2.14	1.53	0.73	1.01	1.09	1.32	1.11	1.17	1.33	1.18	1.64	1.07	0.63	1.88	1.04	0.87	1.26	0.64
Fusion [8]	11.0	0.64	0.94	0.65	0.70	0.96	0.61	1.35	1.48	1.70	1.06	1.26	1.22	1.12	1.20	1.22	1.29	2.07	1.19	0.78	1.54	0.72	0.85	1.24	0.64
Adaptive [25]	12.8	0.64	0.91	0.66	0.88	1.22	0.71	1.06	1.76	1.05	1.03	1.17	1.33	1.09	1.12	1.12	1.20	1.78	1.14	0.86	1.76	0.71	0.93	1.38	0.63
Black & Anandan [3]	12.8	0.68	0.96	0.69	0.94	1.21	0.76	2.33	1.75	1.25	1.08	1.16	1.25	1.04	1.05	1.16	1.17	1.54	1.07	0.73	1.37	0.70	0.87	1.26	0.66

(b)

Fig. 9. Screenshots, taken at the time of writing, of the Middlebury optical flow benchmark [27] with our method highlighted in dark gray. (a) The end-point error rankings. (b) The normalized interpolation rankings. Columns represent the Middlebury test sequences and rows represent the tested algorithms. The latest result tables are available at <http://vision.middlebury.edu/flow/eval/>.

and difficult motion disruptions (wheel spokes overlapping other rotating board).

Fig. 7(i) shows the performance of different motion estimators on unconverged content. The estimators do not converge to the same performance level as the estimators in Fig. 7(d). The (averaging) smoothness priors prevent the motion estimator from converging on smaller objects, whereas the initialized estimators are able to smooth these areas without losing convergence. This trapping in local minima is a fundamental limitation of ICM-type algorithms. Good initialization (by means of a gradual increase in smoothness prior strength) can improve performance. We used this principle by doing a two-stage approach: a weak prior is used to speed up initial convergence and a stronger prior to refine the result. The additional cost of this extra iteration can, in practice, be reduced by exploiting the temporal consistency, i.e., initializing from previous motion estimates.

Last, we note that the operations count and required memory bandwidth of the RS algorithm, for an implementation as described here, is mainly determined by the number of block correlations. An upper bound can be derived directly from the size of the candidate set, the resolution of the block grid, and the number of iterations. The reference RS algorithm described in this paper computes ten block correlations for each block (five candidates per block and two passes). A variant with the above smoothness constraints, i.e., with one additional “flow” candidate, and two consecutive iterations of P_I and P_{III} (four passes), requires 24 block matches per block. Note that this is an upper bound; in practice, optimizations such as the removal of candidate vectors with duplicate values from

the candidate set and the reuse of earlier vector evaluations can greatly reduce the number of block correlations.

B. Middlebury Results

Middlebury University provides a benchmark [27] based on a comparison with ground truth VFs and images. The benchmark is popular in recent optical flow publications, which describe methods with sophisticated regularization. In contrast, the methods and optimization we propose target consumer market embedded devices that process real-time HD video with (severely) constrained resources. Consequently, sophistications that have little effect on the perception of motion-compensated content are eliminated. In particular, very high levels of sub-pixel precision and accurate estimates in texture-less regions are expensive to compute but have little to no effect on the perceived quality. Because these properties are important when benchmarking against a “ground truth” VF, a comparison of our method with the “Middlebury methods” has to take these requirements difference into account.

However, to show that our methodology can compete with recent optical flow methods, we benchmarked a version of our algorithm that contains, at the finest level, “blocks” of 1×1 pixel. A hierarchical coarse-to-fine approach (image pyramid with scaling factor 0.5) is used to speed up convergence, with multiple iterations per level. In addition, the block correlation was adapted to reduce the dc-sensitivity. This reduces the chance of ambiguities in the VFs, in areas where the motion of shadows is being detected instead of the “ground truth” flow (an effect visible in, e.g., the Mequon sequence). The results are shown in Figs. 8 and 9. Despite the simplicity of the

optimization and prior, the algorithm has a top-ten ranking (at the time of writing) in the interpolation tests and outperforms more complex methods, such as GC in the ground truth comparison. We expect that better parameter tuning, the inclusion of occlusion detection/correction, and the addition of image-driven smoothness priors can very likely yield higher rankings.

VII. CONCLUSION

In this paper, we evaluated improved real-time RS-ME algorithms that target piecewise smooth VF gradients. We made modifications to the candidate set and penalty mechanisms that encourage “locally linear” vector candidates. To assess the quality of the produced VFs, we used metrics that jointly show the tradeoff in M2SE and smoothness. The inclusion of different types of gradient and average candidates showed no significant improvement in SI if no mechanism is in place that biases toward the selection of these candidates. We demonstrated that modification of the penalty mechanism has far greater influence on performance (an approximate 50% improvement in spatial consistency with 10% M2SE improvement). We tested a selection of smoothness priors that can be determined computationally and bandwidth efficient. The results show that, with the right smoothness priors, real-time block matching algorithms are capable of producing high-quality, piecewise smooth VFs.

REFERENCES

- [1] E. B. Bellers, J. G. W. M. Janssen, and M. Penners, “Motion compensated frame rate conversion for motion blur reduction,” *SID Dig. Tech. Papers*, vol. 38, no. 1, pp. 1454–1457, 2007.
- [2] C. N. Cordes and G. de Haan, “Key requirements for high quality picture-rate conversion,” *SID Dig. Tech. Papers*, vol. 15, no. 2, pp. 850–853, 2009.
- [3] *NXP Video Back-End Processor PNX5100*, document 9397 750 16130, NXP, Aug. 2007.
- [4] *Micronas FRC 94xyH*, document 6251-663-3PI, Micronas, Aug. 2006.
- [5] *Trident’s DVB-T System Software Powered by HiDTV Pro-QX Integrated with MEMC Technology Enters 100 Hz European iDTV Market*, Trident Press Release, Dec. 2008 [Online]. Available: <http://www.tridentmicro.com>
- [6] G. de Haan, P. W. A. C. Biezen, H. Huijgen, and O. A. Ojo, “True-motion estimation with 3-D recursive search block matching,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 3, no. 5, pp. 368–379, Oct. 1993.
- [7] G. de Haan and P. W. A. C. Biezen, “Sub-pixel motion estimation with 3-D recursive search block-matching,” *Signal Process.*, vol. 6, no. 3, pp. 229–239, Jun. 1994.
- [8] G. de Haan and P. W. A. C. Biezen, “An efficient true-motion estimator using candidate vectors from aparametric motion model,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 1, pp. 85–91, Feb. 1998.
- [9] G. de Haan, “Progress in motion estimation for consumer video format conversion,” *IEEE Trans. Consum. Electron.*, vol. 46, no. 3, pp. 449–459, Aug. 2000.
- [10] N. Atzpadin, P. Kauff, and O. Schreer, “Stereo analysis by hybrid recursive matching for real-time immersive video conferencing,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 3, pp. 321–334, Mar. 2004.
- [11] J. Wang, D. Wang, and W. Zhang, “Temporal compensated motion estimation with simple block-based prediction,” *IEEE Trans. Broadcast.*, vol. 49, no. 3, pp. 241–248, Sep. 2003.
- [12] G. Lee, M. Wang, H.-Y. Lin, D. W.-C. Su, and B.-Y. Lin, “Algorithm/architecture co-design of 3-D spatio-temporal motion estimation for video coding,” *IEEE Trans. Multimedia*, vol. 9, no. 3, pp. 455–465, Apr. 2007.
- [13] B. Furht and B. Furht, *Motion Estimation Algorithms for Video Compression*. Norwell, MA: Kluwer, 1996.
- [14] C. Kuglin and D. Hines, “The phase correlation image alignment method,” in *Proc. Int. Conf. Cybern. Soc.*, vol. 4, 1975, pp. 63–165.
- [15] L. G. Brown, “A survey of image registration techniques,” *ACM Comput. Surveys*, vol. 24, no. 4, pp. 325–376, 1992.
- [16] B. Horn and B. Schunck, “Determining optical flow,” *Artificial Intell.*, vol. 17, nos. 1–3, pp. 185–203, 1981.
- [17] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, “High accuracy optical flow estimation based on a theory for warping,” in *Proc. ECCV*, vol. 4, May 2004, pp. 25–36.
- [18] J. Besag, “On the statistical analysis of dirty pictures,” *J. Appl. Statist.*, vol. 20, no. 5, pp. 63–87, 1993.
- [19] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann, 1988.
- [20] J. Yedidia, W. Freeman, and Y. Weiss, “Understanding belief propagation and its generalizations,” in *Exploring Artificial Intelligence in the New Millennium*. San Francisco, CA: Morgan Kaufmann, 2003, pp. 239–269.
- [21] V. Kolmogorov and R. Zabih, “Computing visual correspondence with occlusions using graph cuts,” in *Proc. ICCV*, vol. 2, 2001, p. 508.
- [22] F. Kelly, “Fast probabilistic inference and GPU video processing,” Ph.D. dissertation, Dept. Electron. Elect. Eng., Trinity College Dublin, Dublin, Ireland, 2006.
- [23] M. J. Black, “Robust incremental optical flow,” Ph.D. dissertation, Dept. Comput. Sci., Yale Univ., New Haven, CT, 1992.
- [24] J. Weickert and C. Schnörr, “A theoretical framework for convex regularizers in PDE-based computation of image motion,” *Int. J. Comput. Vision*, vol. 45, no. 3, pp. 245–264, 2001.
- [25] J. Xiao, H. Cheng, H. Sawhney, C. Rao, and M. Isnardi, “Bilateral filtering-based optical flow estimation with occlusion detection,” in *Proc. ECCV*, vol. 1, 2006, pp. 211–224.
- [26] 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.
- [27] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. Black, and R. Szeliski, “A database and evaluation methodology for optical flow,” in *Proc. 13th Int. Conf. Comput. Vision*, Oct. 2007, pp. 1–8.
- [28] S. Zhu and K. Ma, “A new diamond search algorithm for fast block-matching motion estimation,” *IEEE Trans. Image Process.*, vol. 9, no. 2, pp. 287–290, Feb. 2000.
- [29] C. Zhu, X. Lin, and L.-P. Chau, “Hexagon-based search pattern for fast block motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 5, pp. 349–355, May 2002.
- [30] A. Tourapis, O. Au, and M. L. Liou, “Highly efficient predictive zonal algorithms for fast block-matching motion estimation,” *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 10, pp. 934–947, Oct. 2002.

Chris Bartels received the M.S. degree in electrical engineering from the Eindhoven University of Technology, Eindhoven, The Netherlands, in 2006. He is currently pursuing the Ph.D. degree from the same university on a grant from Philips Consumer Lifestyle.

His current research interests include algorithms for real-time video processing, in particular, motion estimation, and frame rate conversion techniques.

Gerard de Haan received the M.S. and Ph.D. degrees from the Delft University of Technology, Delft, The Netherlands.

He joined Philips Research, Eindhoven, The Netherlands, in 1979, where he is currently a Research Fellow active in the area of video processing and analysis. His research resulted in various commercially available integrated circuits. He has authored 3 books, 2 handbook chapters, 145 papers, and more than 100 international patents and patent applications.

Dr. de Haan received five Best Paper Awards, the IEEE Chester Hall Award, the Gilles Holst Award, and bronze, silver, and gold patent medals, while his work on motion received the EISA European Video Innovation Award, and the Wall Street Journal Business Innovation Award. Since 2000, he has been the Chair of *Video Processing for Multimedia Systems* at the Eindhoven University of Technology, Eindhoven, The Netherlands.