Optical Flow Estimation Using Spatially Varying Smoothing

Theodosios Gkamas

MASTER THESIS

—♦—

Ioannina, October 2010

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF IOANNINA

ΕΚΤΙΜΗΣΗ ΤΗΣ ΟΠΤΙΚΗΣ ΡΟΗΣ ΜΕ ΜΕΘΟΔΟΥΣ ΧΩΡΙΚΑ ΜΕΤΑΒΑΛΛΟΜΕΝΗΣ ΕΞΟΜΑΛΥΝΣΗΣ

Η
ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

Υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης
του Τμήματος Πληροφορικής
Εξεταστική Επιτροπή

από τον

Θεοδόσιο Γκάμα

ως μέρος των Υποχρεώσεων

για τη λήψη

του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ ΣΤΙΣ ΤΕΧΝΟΛΟΓΙΕΣ-ΕΦΑΡΜΟΓΕΣ

Οκτώβριος 2010

DEDICATION

The present thesis is dedicated to my parents, Niko and Anna as well to my brother Dimitri and my sister Vasiliki.

ACKNOWLEDGMENTS

I would like to thank my supervisor Assistant Professor Christophoro Nikou, for the collaboration but also for his innovative ideas which have been instrumental in implementing this thesis.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Gkamas, Theodosios, N.
MSc, Computer Science Department, University of Ioannina, Greece. October, 2010.
Optical flow estimation using spatially varying smoothing.
Thesis Supervisor: Christophoros Nikou.

The problem of estimating the optical flow in a sequence of images is an important research problem in the area of computer vision with applications in visual object tracking, stereopsis and motion segmentation, among others. Optical flow is the 2D velocity field, describing the apparent motion in the image that results from independently moving objects in the scene or from observer motion. Its estimation is a particularly difficult problem due to several factors. At first, the massive image data which produce small and/or large scale linear systems that must be solved to obtain the solution in as little as possible and competitive period of time. Furthermore, the problems that occur because of the nature of the images, such as motion discontinuities and object occlusion must be addressed. To overcome these difficulties, the majority of the state of the art optical flow computation techniques rely on the imposition of a smoothness constraint on the motion field. In this work, we propose two methods for the accurate estimation of the optical flow where the smoothness constraint varies with respect to the image content. The first method is based on image segmentation and the smoothness constraint is applied to image areas belonging to the same segment and simultaneously presenting low spatial gradient information, to avoid smoothing probable motion boundaries. The second method relies on a probabilistic modeling of the optical flow problem where the motion vectors are considered as unobserved random variables generated by a Student's *t*-distribution with spatially varying parameters. In that case, as the complete data likelihood is intractable we recur to the variational-Bayes methodology for inference of the model parameters and variables.

ΕΚΤΕΝΗΣ ΠΕΡΙΛΗΨΗ ΣΤΑ ΕΛΛΗΝΙΚΑ

Θεοδόσιος Γκάμας του Νικολάου και της Αννούλας.
MSc. Τμήμα Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Οκτώβριος, 2010.
Εκτίμηση της οπτικής ροής με μεθόδους χωρικά μεταβαλλόμενης εξομάλυνσης.
Επιβλέπων: Χριστόφορος Νίκου.

Το πρόβλημα εκτίμησης της οπτικής ροής σε μια ακολουθία εικόνων, αποτελεί σημαντικό ερευνητικό πρόβλημα στον τομέα της υπολογιστικής όρασης, με εφαρμογές στην οπτική παρακολούθηση αντικειμένων, την στερεοσκοπία και την κατάτμηση κίνησης, μεταξύ άλλων. Οπτική ροή ονομάζουμε το 2Δ πεδίο μετατοπίσεων, που περιγράφει την εμφανή κίνηση μέσα σε μια εικόνα η οποία προκύπτει από ανεξάρτητα κινούμενα αντικείμενα στην σκηνή ή από την κίνηση του παρατηρητή. Η εκτίμησή της είναι ένα ιδιαίτερα δύσκολο πρόβλημα που οφείλεται σε διάφορους παράγοντες. Καταρχάς, τα ογκώδη δεδομένα της εικόνας που παράγουν μικρής και/ή μεγάλης διάστασης γραμμικά συστήματα τα οποία πρέπει να επιλυθούν για να λάβουμε την λύση μέσα σε όσο το δυνατόν μικρό και ανταγωνιστικό χρονικό διάστημα. Επιπλέον, τα προβλήματα που προκύπτουν λόγω της φύσης των εικόνων, όπως οι μη συνεχείς κινήσεις και οι επικαλύψεις αντικειμένων πρέπει να αντιμετωπιστούν. Για να ξεπεραστούν αυτές οι δυσκολίες, η πλειοψηφία των κορυφαίων τεχνικών υπολογισμού της οπτικής ροής βασίζονται στην εισαγωγή περιορισμών εξομάλυνσης στο πεδίο κίνησης. Στην παρούσα διατριβή, προτείνουμε δύο μεθόδους για την ακριβή εκτίμηση της οπτικής ροής, στις οποίες ο περιορισμός εξομάλυνσης μεταβάλλεται ανάλογα με το περιεχόμενο της εικόνας. Η πρώτη μέθοδος στηρίζεται στην κατάτμηση εικόνας και ο περιορισμός εξομάλυνσης εφαρμόζεται σε περιοχές της εικόνας που ανήκουν στο ίδιο τμήμα και ταυτόχρονα παρουσιάζουν χαμηλή πληροφορία στην χωρική παράγωγο, αποφεύγοντας έτσι την εξομάλυνση σε πιθανά όρια κίνησης. Η δεύτερη μέθοδος βασίζεται σε ένα πιθανοτικό μοντέλο του προβλήματος της οπτικής ροής όπου τα διανύσματα κίνησης θεωρούνται ως κρυφές τυχαίες μεταβλητές παραγόμενες από μια Student’s *t*-κατανομή με χωρικά μεταβαλλόμενες παραμέτρους. Σε αυτή την περίπτωση, επειδή δεν μπορούμε να υπολογίσουμε ακριβώς την συνολική πιθανοφάνεια των δεδομένων ανατρέχουμε στην Μπεϋζιανή (variational-Bayes) μεθοδολογία για την προσέγγιση των παραμέτρων και των τυχαίων μεταβλητών του μοντέλου.

CHAPTER 1. INTRODUCTION

1.1. Objectives of the Thesis

In this thesis, we deal with the problem of optical flow containing small movements. Without doubt, the measurement of optical flow is one of the fundamental problems in computer vision. It is the problem of approximating the movement of brightness patterns in an image sequence and, thus, provides useful information for the determination of the 3D structure of the environment and the object in the image [2] but also can be used for image registration. In the last two decades the quality of optical flow estimation methods has increased dramatically. Starting from the original approaches of Horn and Schunck [25] as well as Lucas and Kanade [28], research developed many new concepts for dealing with shortcomings of previous models. In order to handle discontinuities in the flow field, the quadratic regulariser in the Horn and Schunck model was replaced by smoothness constraints that permit piecewise smooth results [7]. Some of these ideas are close in spirit to methods motivated from robust statistics where outliers are penalized less severely [9]. Coarse-to-fine strategies [2, 9] as well as non-linearised models [7] have been used to tackle large displacements.

However, not only new ideas have improved the quality of optical flow estimation techniques. Also efforts to obtain a better understanding of what the

methods do in detail, and which effects are caused by changing their parameters, gave an insight into how several models could work together. Furthermore, variational formulations of models gave access to the long experience of numerical mathematics in solving partly difficult optimization problems. Finding the optimal solution to a certain model is often not trivial, and often the full potential of a model is not used because concessions to implementation aspects have to be made. Moreover, one method using the variational inference and belonging to the state of the art is the algorithm proposed by T. Brox, A. Bruhn, N. Papenberg and J. Weickert [12] in the year of 2004. Finally, our contribution to this area, is to introduce three method, two from the combination of [7, 32] and a novel approach created via variational inference.

1.2. Structure of the Thesis

The structure of the thesis is as follows: chapter 2 shows three classic methods, the Lucas-Kanade (LK) method [28], the affine optical flow method [39] and the Horn-Schunck method [25]. The last section of this chapter describes the error metrics which were used in order to evaluate the methods. Chapter 3 shows two proposed methods, firstly, the Joint Lucas-Kanade method [7] and secondly, the method of Nagel *et al.* [32]. Additionally, in this chapter, we proposed two variations derived by the combination of [7, 32]. Moreover, chapter 4 introduces a novel algorithm for the estimation of the optical flow, by using the variational Bayes inference. Finally, chapter 5 is the conclusion of the thesis and the future work which worth to be studied further in order to improve the proposed methods.

CHAPTER 2. OPTICAL FLOW

2.1. Definition of the Problem

First of all let's give the definition of the problem. As there are many definitions for optical flow let's start with a short one: optical flow is the observed motion of intensity patterns on the image plane. Another one according to B. Horn and B. Schunck [25], who are among the pioneers in that field, optical flow is the distribution of apparent velocities of movement of brightness patterns in an image. Additional to this, optical flow can arise from relative motion of objects and the viewer [20, 21]. Consequently, optical flow can give important information about the spatial arrangement of the objects viewed and the rate of change of this arrangement [22].

Additionally to the definition, we have to make one fundamental assumption regarding the nature of the scene the moving objects maintain constant intensity profile throughout their motion. This assumption is the famous **brightness constancy assumption** and forms the basis of all the approaches for estimating optical flow. Let

I be an image and $I(x(t), y(t), t)$ denote the intensity of a point projected onto the image at the location $(x(t), y(t))$ at time t . At a time $t + \Delta t$, the projected point moves to a new location $(x(t + \Delta t), y(t + \Delta t))$. According to the *brightness constancy assumption*, the point has the same intensity at both locations, which means

$$I(x(t + \Delta t), y(t + \Delta t), t + \Delta t) = I(x(t), y(t), t). \quad (2.1)$$

Expanding the above equation using Taylor series about the point $(x(t), y(t))$ and taking the limits, a familiar form of the optical flow equation is obtained which is given by

$$f(u, v; t) = I_x u + I_y v + I_t = 0, \quad (2.2)$$

where I_x and I_y represent the partial derivatives of the image in x and y directions respectively, I_t represents the temporal derivative of the image, and u and v are the horizontal and vertical components of the unknown pixel velocity respectively. Given a pair of images and their spatial and temporal derivatives, the goal is to determine $[u, v]^T$. Since there is only one equation involving two unknowns, the system is under-constrained, and an unambiguous solution cannot be obtained. This is the well known **aperture problem**, and herein lays the biggest challenge in estimating the optical flow.

The way to address the *aperture problem* is to add more constraints so as to obtain a required set of equations at least equal in number to the unknowns. Solving for $[u, v]^T$ requires an additional equation which can be obtained, for example, by considering motion of two pixels together instead of one. This results in two equations, and the system can be solved. In practice, multiple pixels are considered together to obtain a set of equations such that their solution minimizes some error function. Most optical flow approaches differ from each other in the way they bunch pixels together for the estimation of their combined velocity, or the kind of error function they minimize.

2.2. Optical Flow Methods

The prominent optical flow approaches can be classified into one of the following categories:

- **Block matching methods:** estimating the optical flow vectors for a window of pixels by computing its warp in the consecutive frame using techniques like *normalized cross correlation* (NCC), *sum of absolute differences* (SAD), or *sum of squared differences* (SSD) [2].

- **Differential methods:** using the spatial and temporal derivatives of the image to estimate the pixel displacement. This can be achieved by computing local displacement of image patches (*Lucas-Kanade* [28]), or imposing a global smoothness function on the flow field (*Horn-Schunck* [25]), or a combination of both (*Bruhn et al.* [13], *Birchfield-Pundlik* [7]). *Lucas-Kanade* appeals more to the idea of sparse optical flow while *Horn-Schunck* approach is more suited for computing dense flow.

- **Variational methods:** involving use of additional terms based on the calculus of variations in the energy functional to be minimized to obtain optical flow. Such techniques have become popular recently because of their ability to model the discontinuities in the motion and produce highly accurate optical flow estimates (*Cremers-Soatto* [17], *Brox et al.* [11]).

The next section describes three classic algorithms for estimating the optical flow.

2.3. Classic Algorithms for Computing Optical Flow

In this section we are going to describe three classic methods for estimating optical flow, which are *Lucas–Kanade* [28], *Affine Optical Flow* [39] and *Horn–Schunck* [25].

2.3.1. Lucas-Kanade (LK) Method

The basic assumption in the Lucas-Kanade (LK) method is that the pixels in a local neighborhood undergo a constant but unknown displacement $\mathbf{u} = [u \quad v]^T$. This additional constraint is used to overcome the aperture problem as it yields one optical flow Equation (see 2.2) per pixel in the neighborhood. The constant displacement of neighboring pixels implies two basic assumptions, namely, the spatial coherence (neighboring pixels belong to the same 3D surface projected onto the image plane) and the temporal persistence (motion of the pixel neighborhood changes gradually over time). Let I and J be the two frames between which the flow has to estimated and let $\mathbf{x} = [x \quad y]^T$ denote a pixel location. Optical flow equation (2.2) for a single pixel $\mathbf{x}$ can be rewritten as

$$[\, I_x(\mathbf{x}) \quad I_y(\mathbf{x}) \,] \begin{bmatrix} u \\ v \end{bmatrix} = -I_t(\mathbf{x}) = I(\mathbf{x}) - J(\mathbf{x}) \tag{2.3}$$

Considering that the n points $\mathbf{x_1}, \ldots, \mathbf{x_n}$ in a local neighborhood have the same amount of displacement, all of the n pixels will then follow equation (2.3), leading to

$$\begin{bmatrix} I_x(\mathbf{x_1}) & I_y(\mathbf{x_1}) \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ I_x(\mathbf{x_n}) & I_y(\mathbf{x_n}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_t(\mathbf{x_1}) \\ \cdot \\ \cdot \\ \cdot \\ I_t(\mathbf{x_n}) \end{bmatrix} \tag{2.4}$$

$$\begin{bmatrix} I_x(\mathbf{x_1}) & \ldots & I_y(\mathbf{x_1}) \\ I_x(\mathbf{x_n}) & \ldots & I_y(\mathbf{x_n}) \end{bmatrix} \begin{bmatrix} I_x(\mathbf{x_1}) & I_y(\mathbf{x_1}) \\ \cdot & \cdot \\ \cdot & \cdot \\ \cdot & \cdot \\ I_x(\mathbf{x_n}) & I_y(\mathbf{x_n}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} I_x(\mathbf{x_1}) & \ldots & I_y(\mathbf{x_1}) \\ I_x(\mathbf{x_n}) & \ldots & I_y(\mathbf{x_n}) \end{bmatrix} \begin{bmatrix} I_t(\mathbf{x_1}) \\ \cdot \\ \cdot \\ \cdot \\ I_t(\mathbf{x_n}) \end{bmatrix} \tag{2.5}$$

$$\sum_{j=1}^{n}\begin{bmatrix} I_x^2(\mathbf{x}_j) & I_x(\mathbf{x}_j)\,I_y(\mathbf{x}_j) \\ I_x(\mathbf{x}_j)\,I_y(\mathbf{x}_j) & I_y^2(\mathbf{x}_j) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} = \sum_{j=1}^{n}\begin{bmatrix} I_x(\mathbf{x}_j)I_t(\mathbf{x}_j) \\ I_y(\mathbf{x}_j)I_t(\mathbf{x}_j) \end{bmatrix} \qquad (2.6)$$

Equation (2.6) consolidates the optical flow by summing the spatial and temporal derivatives over the neighborhood. Instead of performing a summation over a spatial window, a weighted window such as a Gaussian with its mean at the center pixel can also be used. Hence, a general case of Lucas-Kanade equation is given by

$$\begin{bmatrix} \mathbf{K}_p * \left(I_x^2\right) & \mathbf{K}_p * \left(I_x I_y\right) \\ \mathbf{K}_p * \left(I_x I_y\right) & \mathbf{K}_p * \left(I_y^2\right) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \mathbf{K}_p * \left(I_x I_t\right) \\ . \\ \mathbf{K}_p * \left(I_y I_t\right) \end{bmatrix}, \qquad (2.7)$$

where K_p is a suitable convolution kernel whose size determines the number of neighboring pixels to be aggregated and assigns appropriate weights to the pixels inside the window. The size of K_p has to be selected carefully because a small sized window may not be enough to overcome the aperture problem due to the presence of image noise. On the other hand, a very large window size may lead to the breakdown of spatial coherency assumption. Equation (2.7) can be written in a simplified form as

$$\mathrm{Z}\ \mathbf{u} = \mathrm{e} \qquad (2.8)$$

It can be seen that Z looks like a covariance matrix with squares of gradients in the *x* and *y* directions along the diagonal, and it is symmetric, which is why it is called the gradient covariance matrix or the Hessian.

Displacements **u** of a local neighborhood of pixels can be directly determined by solving (2.8) via least squares, i.e. by minimizing

$$E_{KL}(\mathbf{u}) = K_p * \left(f\,(\mathbf{u}, t)^2 \right), \qquad (2.9)$$

or equivalently, solving for the estimate $\hat{u} = \mathrm{Z}^{-1}\mathrm{e}$. However, this may not yield an accurate estimate because (2.6) is a linear approximation of a nonlinear function (the original optical flow equation is nonlinear if all the terms in the Taylor series are considered). To obtain an accurate estimate, iterative schemes such as Newton-Raphson [15] are used. Newton-Raphson is a popular technique of approximating the values of the roots of a real valued function given the initial estimate of the roots. Consider a 1D case, where if $u^{(k)}$ (pixel displacement in 1D) is the estimate of the root of function $f(u,t) = I_x u + I_t = 0$ (1D counterpart to the optic flow function) at the k^{th} iteration, then its update value at $(k+1)^{th}$ iteration is given by $u^{(k)} - \frac{f\left(u^{(k)}\right)}{f'\left(u^{(k)}\right)}$. From inspection it can be seen that $f\left(u^{(k)}\right) = I_x u^{(k)} + I_t$ and $f'\left(u^{(k)}\right) = I_x$, which means $u^{(k+1)} = -\frac{I_t}{I_x}$. Every iteration yields a value of *u* that is added to the overall displacement and convergence is obtained when *u* does not change significantly between two iterations. Extending this idea to two dimensions, every iteration of the Newton-Raphson technique gives a displacement $\mathbf{u}^{(k)}$ of the window. The window in the next frame is shifted by **u** and warped with the first image to obtain a new value of I_t at each iteration and a new displacement estimate is found using $\hat{\mathbf{u}} = \mathrm{Z}^{-1}\mathrm{e}$ (see Algorithm Lucas-Kanade for a complete description).

To efficiently compute the optical flow using LK, some implementation issues should be addressed. The computational cost of the algorithm depends on the nature of mathematical operations performed and the time it takes to converge. Since the same set of steps are applied to each point (or each pixel) for which the flow field is computed, reducing the computation time of one flow vector directly affects the overall computation cost. Looking at the description of the Lucas-Kanade algorithm (figure 2.1) it can be seen that the mathematical operations include computing Z^{-1}, spatial derivatives of the image *I* and warping of the window in image *J* to compute I_t. Of the above mentioned quantities, image derivatives can be computed beforehand along with their squares and products (hence, *Z* for each point can be computed beforehand). Solving for a system of equations shown in (2.8) yields **u**, but it is more efficient to use Gaussian elimination rather than actually computing Z^{-1}.

The only computation that needs to be iteratively performed is the warping of the window in the second image and computation of **e**. Usually, the location of the shifted window is given by non-integers. Hence, methods like bilinear interpolation are utilized to compute the value of image intensity at sub-pixel precision. This improves the accuracy of estimation of **u**. Regarding the convergence, Newton-Raphson reaches an optimum solution within a few iterations if the initial estimate of the root is close enough. In this case it also depends on ε_{LK}, the threshold for minimum displacement obtained during one iteration.

Algorithm: Lucas-Kanade

Input: two images I and J of a sequence

Output: optical flow field

1. pre-compute the spatial derivatives I_x and I_y
2. initialize K_p
3. for each point i
   - (a) compute gradient covariance matrix, Z_i
   - (b) initialize $\mathbf{u}_i = (0, 0)$
   - (c) repeat until convergence
     - i. compute I_t from first image and shifted second image,

       $I_t = I(\mathbf{x}_i) - J(\mathbf{x}_i + \mathbf{u}_i)$

     - ii. compute $\mathbf{e}_i$
     - iii. find the estimate of displacement, $\hat{u}_i = Z_i^{-1} e_i$
     - iv. $\mathbf{u}_i = \mathbf{u}_i + \hat{u}_i$
     - v. if $\left\| \hat{\mathbf{u}}_i \right\| < \varepsilon_{KL}$ (minimum displacement threshold), exit

Figure 2.1: The standard Lucas-Kanade algorithm.

Many implementations of LK adopt a coarse-to-fine refinement strategy to accurately estimate optic flow [6, 10]. The idea here is to sub-sample the images progressively and build image pyramids such that the coarsest scale is at the top. Then **u** is computed starting from the coarsest level to the finest level. At every level,

the **u** is scaled up according to the scale factor of that level and the warp is computed between corresponding levels of the two image pyramids. There are two main advantages of such an approach. First, it reduces the effect of temporal aliasing and the high frequency component introduced as a result in the image signal. Second, it can estimate large motions (where inter-frame displacement of the feature window is large). Since velocity is reduced at the coarsest level, estimates at the coarsest level can be scaled up and determined accurately at the finer levels. Computational cost in this kind of implementation is increased as compared to the standard case and is directly proportional to the number of levels of the pyramid used. A pyramidal implementation of LK is $O(nNm)$ as compared to $O(Nm)$ of the single scale implementation, where N is the number of points, m is average number of Newton-Raphson iterations and n is the number of pyramid levels.

2.3.2. Affine Optical Flow

Affine optical flow is an extension of the previously described *Lucas-Kanade* method.

❖ *Two Models of Image Motion*

As the camera moves, the patterns of image intensities change in a complex way. However, away from occluding boundaries and near surface markings, these changes can often be described as image motion,

$$I(x, y, t+\tau) = I(\, x - \xi\,(x, y, t, \tau),\;\; y - \eta(x, y, t, \tau)\,)\,, \qquad (2.10)$$

Thus, a later image taken at time $t+\tau$ can be obtained by moving every point in the current image, taken at time t, by a suitable amount. The amount of motion $\delta = (\xi, \eta)$ is called the *displacement* of the point at $\mathbf{x} = (x, y)$.

The displacement vector δ is a function of the image position **x**, and variations in δ are often noticeable even within the small windows used for tracking. It then

makes little sense to speak of “the” displacement of a feature window, since there are different displacements within the same window. An *affine motion field* is a better representation:

$$\delta = D\mathbf{x} + \mathbf{d} \ , \tag{2.11}$$

where

$$D = \begin{bmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{bmatrix},$$

is a deformation matrix, and **d** is the translation of the feature window’s center. The image coordinates **x** are measured with respect to the window’s center. Then, a point **x** in the first image *I* moves to point *A***x** + **d** in the second image *J*, where A = **1** + *D* and **1** is the 2 *x* 2 identity matrix:

$$J\left(A\mathbf{x} + \mathbf{d}\right) = I(\mathbf{x}) \ , \tag{2.12}$$

Given two images *I* and *J* and a window in image *I*, tracking means determining the six parameters that appear in the deformation matrix *D* and displacement vector **d**. The quality of this estimate depends on the size of the feature window, the texturedness of the image within it, and the amount of camera motion between frames. When the window is small, the matrix *D* is harder to estimate, because the variations of motion within it are smaller and therefore less reliable. However, smaller windows are in general preferable for tracking because they are less likely to straddle a depth discontinuity. For this reason, a *pure translation* model is preferable during tracking, where the deformation matrix *D* is assumed to be zero:

$$\delta = \mathbf{d}.$$

According to J. Shi and C. Tomasi [39], experiments had shown that the best combination of these two motion models is pure translation for tracking, because of its higher reliability and accuracy over the small inter-frame motion of the camera, and affine motion for comparing features between the first and the current frame in order to monitor their quality.

❖ *Computing Image Motion*

Because of image noise and because the affine motion model is not perfect, (2.12) is in general not satisfied exactly. The problem of determining the motion parameters is then that of finding the A and $\mathbf{d}$ that minimize the *dissimilarity*

$$\varepsilon = \int_W \int \left[J(A\mathbf{x} + \mathbf{d}) - I(\mathbf{x}) \right]^2 w(\mathbf{x}) d\mathbf{x} \ , \tag{2.13}$$

where W is the given feature window and $w(\mathbf{x})$ is a weighting function. In the simplest case, $w(\mathbf{x}) = 1$. Alternatively, w could be a Gaussian-like function to emphasize the central area of the window. Under pure translation, the matrix A is constrained to be equal to the identity matrix. To minimize the residual (2.13), we differentiate it with respect to the unknown entries of the deformation matrix D and the displacement vector $\mathbf{d}$ and set the result to zero. We then linearize the resulting system by the truncated Taylor expansion

$$J(A\mathbf{x} + \mathbf{d}) = J(\mathbf{x}) + g^T(\mathbf{u}) \, . \tag{2.14}$$

This yields (see [40]) the following linear 6 *x* 6 system:

$$T \, \mathbf{z} = \mathbf{a} \, , \tag{2.15}$$

where $\mathbf{z}^T = [d_{xx} \ \ d_{yx} \ \ d_{xy} \ \ d_{yy} \ \ d_x \ \ d_y]$ collects the entries of the deformation D and displacement $\mathbf{d}$, the error vector

$$\mathrm{a} = \iint_{\mathrm{W}} \left[I(\mathbf{x}) - J(\mathbf{x}) \right] \begin{bmatrix} xg_x \\ xg_y \\ yg_x \\ yg_y \\ g_x \\ g_y \end{bmatrix} w \, d\mathbf{x}$$

depends on the difference between the two images, and the 6 *x* 6 matrix *T*, which can be computed from one image, can be written as

$$T = \iint_W \begin{bmatrix} U & V \\ V^T & Z \end{bmatrix} w \, d\mathbf{x} \ , \qquad (2.16)$$

where

$$U = \begin{bmatrix} x^2 g_x^2 & x^2 g_x g_y & xyg_x^2 & xyg_x g_y \\ x^2 g_x g_y & x^2 g_y^2 & xyg_x g_y & xyg_y^2 \\ xyg_x^2 & xyg_x g_y & y^2 g_x^2 & y^2 g_x g_y \\ xyg_x g_y & xyg_y^2 & y^2 g_x g_y & y^2 g_y^2 \end{bmatrix}, \quad V^T = \begin{bmatrix} xg_x^2 & xg_x g_y & yg_x^2 & yg_x g_y \\ xg_x g_y & xg_y^2 & yg_x g_y & yg_y^2 \end{bmatrix},$$

$$Z = \begin{bmatrix} g_x^2 & g_x g_y \\ g_x g_y & g_y^2 \end{bmatrix}.$$

Even when affine motion is a good model, equation is only approximately satisfied, because of the linearization of (2.14). However, the correct affine change can be found by using (2.15) iteratively in a Newton-Raphson style minimization [40].

During tracking, the affine deformation *D* of the feature window is likely to be small, since motion between adjacent frames must be small in the first place for tracking to work at all. It is then safer to set *D* to the zero matrix. In fact, attempting to determine deformation parameters in this situation is not only useless but can lead to poor displacement solutions: in fact, the deformation *D* and the displacement **d** interact through the 4 *x* 2 matrix *V* of equation (2.16), and any error in *D* would cause errors in **d**. Consequently, when the goal is to determine **d,** the smaller system

$$Z\,\mathbf{d} = \mathbf{e} \ , \qquad (2.17)$$

should be solved, where **e** collects the last two entries of the vector **a** of equation (2.15).

2.3.3. Horn-Schunck (HS) Method

The main difference between Lucas-Kanade and Horn-Schunck is that in the first method we used a window in which we consider all pixels having the same displacement, while in the second method we handle every pixel independently.

First of all, let's see the optical flow equation without the summing window. We will derive an equation that relates the change in image brightness at a point to the motion of the brightness pattern. Let the image brightness at the point (x, y) in the image plane at time t be denoted by *I(x,* y, t). Now consider what happens when the pattern moves. The brightness of a particular point in the pattern is constant, so that

$$\frac{dI}{dt} = 0. \tag{2.18}$$

Using the chain rule for differentiation we see that

$$\frac{\partial I}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t} = 0. \tag{2.19}$$

(See Appendix A for a more detailed derivation).

If we let $u = \frac{dx}{dt}$ and $v = \frac{dy}{dt}$, then it is easy to see that we have a single linear equation in the two unknowns u and v,

$$I_x u + I_y v + I_t = \mathrm{O}\ , \tag{2.20}$$

where we have also introduced the additional abbreviations I_x, I_y, and I_t for the partial derivatives of image brightness with respect to *x*, *y* and *t*, respectively. The constraint on the local flow velocity expressed by this equation is illustrated in figure 2.2, where we can see that the basic rate of change of image brightness equation constrains the optical flow velocity. The velocity (*u*, *v*) has to lie along a line perpendicular to the

brightness gradient vector (I_x, I_y). The distance of this line from the origin equals I_t divided by the magnitude of (I_x, I_y). Writing the equation in still another way,

$$(I_x, I_y) \cdot (u, v) = - I_t. \qquad (2.21)$$

Thus the component of the movement in the direction of the brightness gradient (I_x, I_y) equals: $-\frac{I_t}{\sqrt{I_x^2 + I_y^2}}$.

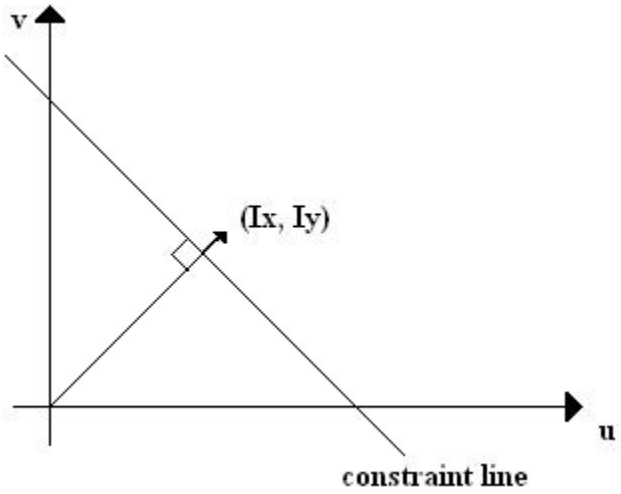


Figure 2.2: The constraint on the local flow velocity.

We cannot, however, determine the component of the movement in the direction of the iso-brightness contours, at right angles to the brightness gradient. As a consequence, the flow velocity (u, v) cannot be computed locally without introducing additional constraints.

Now we will see some more complex issues inside the method.

First of all, we will analyze, what we call, the **smoothness constraint**. If every point of the brightness pattern can move independently, there is little hope of recovering the velocities. More commonly we view opaque objects of finite size undergoing rigid motion or deformation. In this case neighboring points on the objects have similar velocities and the velocity field of the brightness patterns in the image

varies smoothly almost everywhere. Discontinuities in flow can be expected where one object occludes another. An algorithm based on a smoothness constraint is likely to have difficulties with occluding edges as a result.

One way to express the additional constraint is to minimize the square of the magnitude of the gradient of the optical flow velocity:

$$\left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 \text{ and } \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2, \tag{2.22}$$

Another measure of the smoothness of the optical flow field is the sum of the squares of the Laplacians of the *x*- and *y*-components of the flow. The Laplacians of *u* and *v* are defined as

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \text{ and } \nabla^2 v = \frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}, \tag{2.23}$$

In simple situations, both Laplacians are zero. If the viewer translates parallel to a flat object, rotates about a line perpendicular to the surface or travels orthogonally to the surface, then the second partial derivatives of both *u* and *v* vanish (assuming perspective projection in the image formation). Horn-Schunck here uses the square of the magnitude of the gradient as smoothness measure.

Secondly, let's see how Horn-Schunck estimates the **partial derivatives**. We must estimate the derivatives of brightness from the discrete set of image brightness measurements available. It is important that the estimates of I_x, I_y, and I_t, be consistent. That is, they should all refer to the same point in the image at the same time. While there are many formulas for approximate differentiation [16, 23] we will use a set which gives us an estimate of I_x, I_y, I_t, at a point in the center of a cube formed by eight measurements. The relationship in space and time between these measurements is shown in figure 2.3. Each of the estimates is the average of four first differences taken over adjacent measurements in the cube. More analytically, the

column index *j* corresponds to the *x* direction in the image, the row index *i* to the *y* direction, while *k* lies in the time direction.

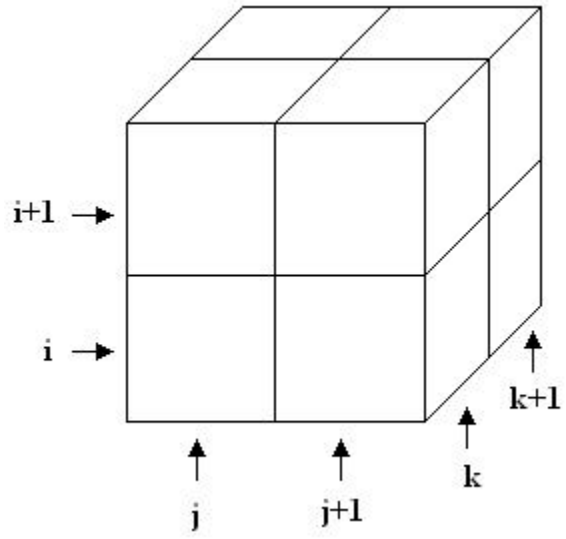


Figure 2.3: The relationship in space and time between I_x, I_y, I_t.

$$I_x \approx \frac{1}{4}\left\{I_{i,j+1,k} - I_{i,j,k} + I_{i+1,j+1,k} - I_{i+1,j,k} + I_{i,j+1,k+1} - I_{i,j,k+1} + I_{i+1,j+1,k+1} - I_{i+1,j,k+1}\right\},$$

$$I_y \approx \frac{1}{4}\left\{I_{i+1,j,k} - I_{i,j,k} + I_{i+1,j+1,k} - I_{i,j+1,k} + I_{i+1,j,k+1} - I_{i,j,k+1} + I_{i+1,j+1,k+1} - I_{i,j+1,k+1}\right\},$$

$$I_t \approx \frac{1}{4}\left\{I_{i,j,k+1} - I_{i,j,k} + I_{i+1,j,k+1} - I_{i+1,j,k} + I_{i,j+1,k+1} - I_{i,j+1,k} + I_{i+1,j+1,k+1} - I_{i+1,j+1,k}\right\},$$

(2.24)

Here the unit of length is the grid spacing interval in each image frame and the unit of time is the image frame sampling period**.**

We also need to approximate the Laplacians of the flow velocities *u* and *v*. One convenient approximation takes the following form

$$\nabla^2 u \approx \kappa\left(\bar{u}_{i,j,k} - u_{i,j,k}\right) \text{ and } \nabla^2 v \approx \kappa\left(\bar{v}_{i,j,k} - v_{i,j,k}\right), \qquad (2.25)$$

where the local averages $\bar{u}$ and $\bar{v}$ are defined as follows

$$\bar{u}_{i,j,k} \approx \frac{1}{6}\left\{u_{i-1,j,k} + u_{i,j+1,k} + u_{i+1,j,k} + u_{i,j-1,k}\right\} + \frac{1}{12}\left\{u_{i-1,j-1,k} + u_{i-1,j+1,k} + u_{i+1,j+1,k} + u_{i+1,j-1,k}\right\}$$

$$\bar{v}_{i,j,k} \approx \frac{1}{6}\left\{v_{i-1,j,k} + v_{i,j+1,k} + v_{i+1,j,k} + v_{i,j-1,k}\right\} + \frac{1}{12}\left\{v_{i-1,j-1,k} + v_{i-1,j+1,k} + v_{i+1,j+1,k} + v_{i+1,j-1,k}\right\} \quad (2.26)$$

The proportionality factor κ equals 3 if the average is computed as shown and we again assume that the unit of length equals the grid spacing interval. In figure 2.4 we can see that the Laplacian is estimated by subtracting the value at a point from a weighted average of the values at neighboring points.

1/12	1/6	1/12
1/6	-1	1/6
1/12	1/6	1/12

Figure 2.4: The Laplacian operator.

Now we have to analyze the **minimization problem**. Horn-Schunck minimizes the sum of the errors in the equation for the rate of change of image brightness,

$$E_b = I_x u + I_y v + I_t , \quad (2.27)$$

and the measure of the departure from smoothness in the velocity flow,

$$E_c^2 = \left(\frac{\partial u}{\partial x}\right)^2 + \left(\frac{\partial u}{\partial y}\right)^2 + \left(\frac{\partial v}{\partial x}\right)^2 + \left(\frac{\partial v}{\partial y}\right)^2 , \quad (2.28)$$

What should be the relative weight of these two factors? In practice the image brightness measurements will be corrupted by quantization error and noise so that we cannot expect E_b to be identically zero. This quantity will tend to have an error

magnitude that is proportional to the noise in the measurement. This fact guides us in choosing a suitable weighting factor, denoted by α^2, as will be seen later.
Let the total error to be minimized be

$$E^2 = \iint \left(\alpha^2 E_c^2 + E_b^2\right) \mathrm{d}x\, \mathrm{d}y\,, \tag{2.29}$$

The minimization is to be accomplished by finding suitable values for the optical flow velocity (u, v). Using calculus of variation (see Appendix C) we obtain

$$\begin{cases} I_x^2 u + I_x I_y v = \alpha^2 \nabla^2 u - I_x I_t \\ I_x I_y u + I_y^2 v = \alpha^2 \nabla^2 v - I_y I_t \end{cases}, \tag{2.30}$$

Using the approximation to the Laplacian introduced previously we will get,

$$\begin{cases} (\alpha^2 + I_x^2) u + I_x I_y v = (\alpha^2 \bar{u} - I_x I_t) \\ I_x I_y u + (\alpha^2 + I_y^2) v = (\alpha^2 \bar{v} - I_y I_t) \end{cases}, \tag{2.31}$$

The determinant of the coefficient matrix equals $\alpha^2(\alpha^2 + I_x^2 + I_y^2)$. Solving for u and v we find that

$$\begin{cases} (\alpha^2 + I_x^2 + I_y^2) u = +(\alpha^2 + I_y^2)\bar{u} - I_x I_y \bar{v} - I_x I_t \\ (\alpha^2 + I_x^2 + I_y^2) v = -I_x I_y \bar{u} + (\alpha^2 + I_x^2)\bar{v} - I_y I_t \end{cases}, \tag{2.32}$$

Let us now see the difference of the flow at a point by using local average in comparison with the LK method. Firstly, (2.32) can be written in the alternative form

$$\begin{cases} (\alpha^2 + I_x^2 + I_y^2)(u - \bar{u}) = -I_x (I_x \bar{u} + I_y \bar{v} + I_t) \\ (\alpha^2 + I_x^2 + I_y^2)(v - \bar{v}) = -I_y (I_x \bar{u} + I_y \bar{v} + I_t) \end{cases}, \tag{2.33}$$

This shows that the value of the flow velocity (u, v) which minimizes the error E^2 lies in the direction towards the constraint line along a line that intersects the

constraint line at right angles. This relationship is illustrated geometrically in figure 2.5, and the value of the flow velocity which minimizes the error lies on a line drawn from the local average of the flow velocity perpendicular to the constraint line. The distance from the local average is proportional to the error in the basic formula for rate of change of brightness when $\bar{u}$, $\bar{v}$ are substituted for *u* and *v*. Finally we can see that α^2 plays a significant role only for areas where the brightness gradient is small, preventing haphazard adjustments to the estimated flow velocity occasioned by noise in the estimated derivatives. This parameter should be roughly equal to the expected noise in the estimate of $I_x^2 + I_y^2$.

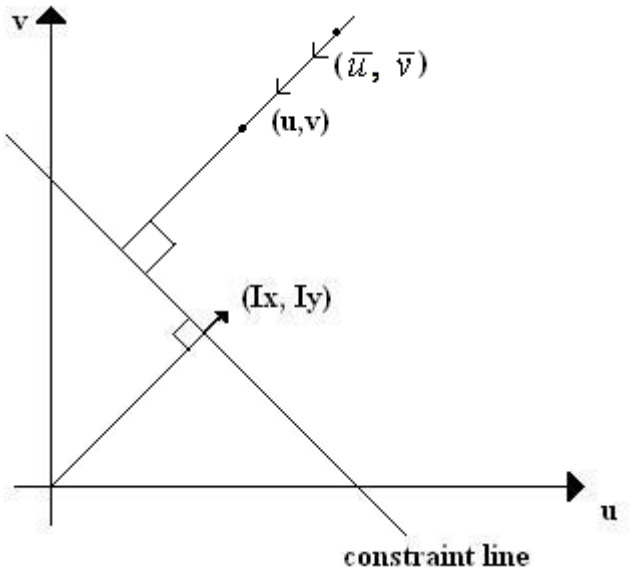


Figure 2.5: The relationship between (u, v), $(\bar{u},\bar{v})$ *,* I_x *and* I_y*.*

Additionally to the previous part, we are going to analyze the impact of parameter α^2. When we allow α^2 to tend to zero we obtain the solution to a constrained minimization problem. Applying the method of Lagrange multipliers [36, 43] to the problem of minimizing E_c^2 while maintain $E_b = 0$ leads to

$$I_x \nabla^2 u = I_x \nabla^2 v \ , \quad I_x u + I_y v + I_t = 0$$

Approximating the Laplacian by the difference of the velocity at a point and the average of its neighbors then give us

$$\begin{cases} \left(I_x^2 + I_y^2\right)\left(u - \bar{u}\right) = -I_x \left[I_x\bar{u} + I_y\bar{v} + I_t \right] \\ \left(I_x^2 + I_y^2\right)\left(v - \bar{v}\right) = -I_y \left[I_x\bar{u} + I_y\bar{v} + I_t \right] \end{cases}, \tag{2.34}$$

Referring again to figure 2.5, we note that the point computed here lies at the intersection of the constraint line and the line at right angles through the point $(\bar{u}, \bar{v})$. We will not use these equations since we do expect errors in the estimation of the partial derivatives.

We now have a pair of equations for each point in the image, let's see which will be the **iterative solution**. It would be very costly to solve these equations simultaneously by one of the standard methods, such as Gauss-Jordan elimination [23, 24]. The corresponding matrix is sparse and very large since the number of rows and columns equals twice the number of picture cells in the image. Iterative methods, such as the Gauss-Seidel method [23, 24], suggest themselves. We can compute a new set of velocity estimates (u^{n+1}, v^{n+1}) from the estimated derivatives and the average of the previous velocity estimates $\left(\bar{u}^n, \bar{v}^n\right)$ by

$$\begin{cases} u^{n+1} = \bar{u}^n - I_x \left[I_x\bar{u}^n + I_y\bar{v}^n + I_t \right] / \left(\alpha^2 + I_x^2 + I_y^2\right) \\ v^{n+1} = \bar{v}^n - I_y \left[I_x\bar{u}^n + I_y\bar{v}^n + I_t \right] / \left(\alpha^2 + I_x^2 + I_y^2\right) \end{cases}, \tag{2.35}$$

It is interesting to note that the new estimates at a particular point do not depend directly on the previous estimates at the same point.

The natural boundary conditions for the variational problem turn out to be a zero normal derivative. At the edge of the image, some of the points needed to compute the local average of velocity lie outside the image. Here we simply copy velocities from adjacent points further in.

The next point we are going to analyze is the case how we have to **fill in uniform regions**. In parts of the image where the brightness gradient is zero, the velocity estimates will simply be averages of the neighboring velocity estimates. There is no local information to constrain the apparent velocity of motion of the brightness pattern in these areas. Eventually the values around such a region will propagate inwards. If the velocities on the border of the region are all equal to the same value, then points in the region will be assigned that value too, after a sufficient number of iterations. Velocity information is thus filled in from the boundary of a region of constant brightness.

If the values on the border are not all the same, it is a little more difficult to predict what will happen. In all cases, the values filled in will correspond to the solution of the Laplace equation for the given boundary condition [1, 31, 35].

The progress of this filling-in phenomena is similar to the propagation effects in the solution of the heat equation for a uniform fiat plate, where the time rate of change of temperature is proportional to the Laplacian. This gives us a means of understanding the iterative method in physical terms and of estimating the number of steps required. The number of iterations should be larger than the number of picture cells across the largest region that must be filled in. If the size of such regions is not known in advance one may use the cross-section of the whole image as a conservative estimate.

Another part we have to discuss is the **tightness of constraint**. When brightness in a region is a linear function of the image coordinates we can only obtain the component of optical flow in the direction of the gradient. The component at right angles is filled in from the boundary of the region as described before. In general the solution is most accurately determined in regions where the brightness gradient is not too small and varies in direction from point to point. Information which constrains both components of the optical flow velocity is then available in a relatively small neighborhood. Too violent fluctuations in brightness on the other hand are not desirable since the estimates of the derivatives will be corrupted as the result of under-sampling and aliasing.

Also we have to choose the **iterative scheme**. As a practical matter one has a choice of how to interlace the iterations with the time steps. On the one hand, one could iterate until the solution has stabilized before advancing to the next image frame. On the other hand, given a good initial guess one may need only one iteration per time-step. A good initial guess for the optical flow velocities is usually available from the previous time-step.

The advantages of the latter approach include an ability to deal with more images per unit time and better estimates of optical flow velocities in certain regions. Areas in which the brightness gradient is small lead to uncertain, noisy estimates obtained partly by filling in from the surround. These estimates are improved by considering further images. The noise in measurements of the images will be independent and tend to cancel out. Perhaps more importantly, different parts of the pattern will drift by a given point in the image. The direction of the brightness gradient will vary with time, providing information about both components of the optical flow velocity. A practical implementation would most likely employ one iteration per time step for these reasons.

2.4. Error Metrics

The first measure of performance that we use in the comparison is the **average angular error** (*AAE*) [4]. This is the most common measure of performance for optical flow [3]. Let $\mathbf{v}_0 = (u_0 , \upsilon_0)$ be the correct velocity and $\mathbf{v}_1 = (u_1 , \upsilon_1)$ be the estimated velocity. The **angular error** (*AE*) between these two vectors is

$$\psi_{AE} = \arccos\left(\vec{\mathbf{v}}_0 \cdot \vec{\mathbf{v}}_1\right) \quad , \tag{2.36}$$

where $\vec{\mathbf{v}}_0$, $\vec{\mathbf{v}}_1$ are the 3D normalized representations of $\mathbf{v}_0$, $\mathbf{v}_1$, respectively and they are defined as

$$\vec{\mathbf{v}}_0 = \frac{1}{\sqrt{u_0^2 + \upsilon_0^2 + 1}}\left(u_0 , \upsilon_0 , 1\right) \quad , \tag{2.37}$$

$$\vec{\mathbf{v}}_1 = \frac{1}{\sqrt{u_1^2 + \upsilon_1^2 + 1}}(u_1, \upsilon_1, 1) \quad . \tag{2.38}$$

The *AAE* is then obtained by calculating the average of all angular errors between correct and estimated velocities in the optical flow. However, it can be seen from (2.36) that errors in regions of large flows are penalized less in *AE* than errors in regions of small flows [3]. One needs to be cautious when using the *AAE* metric as estimates with the same error magnitude may result in significantly different angular error values.

Another error metric is the **normalized magnitude** of the vector difference between the correct and estimated flow vectors [29]. The magnitude of the correct velocity is used as the normalization factor. The magnitude of difference error is defined as

$$E_M = \begin{cases} \dfrac{\|\mathbf{v}_0 - \mathbf{v}_1\|}{\|\mathbf{v}_0\|}, & \text{if } \|\mathbf{v}_0\| \geq T \\ \left| \dfrac{\|\mathbf{v}_1\| - T}{T} \right|, & \text{if } \|\mathbf{v}_0\| < T \quad \text{and } \|\mathbf{v}_1\| \geq T \\ 0, & \text{if } \|\mathbf{v}_0\| < T \quad \text{and } \|\mathbf{v}_1\| < T \end{cases} , \tag{2.39}$$

where T is a threshold, whose purpose is to ignore smaller vectors' norms than T . The algorithm is not expected to reliably produce accurate flow vectors in areas where the actual flow magnitude is less than T [29]. We used $T = 0.5$ in all of our experiments. The **average magnitude of difference error** (*AME*) is then calculated as the average of the normalized magnitude of difference errors.

A third error metric, which is slightly similar with *AAE*, is the *absolute* error, which is the error in flow **endpoint** (*EP*) [3] defined by

$$EP = \sqrt{(u_0 - u_1)^2 + (\upsilon_0 - \upsilon_1)^2} = \|\mathbf{v}_0 - \mathbf{v}_1\| \ , \tag{2.40}$$

CHAPTER 3. COMPUTING OPTICAL FLOW THROUGH SYNERGY OF ADAPTIVE SMOOTHING AND SEGMENTATION

In this chapter we study two methods. Firstly, S. Birchfield's and S. Pundlik's method [7] (section 3.1) and secondly, H. Nagel's and W. Enkelmann's method [32] (section 3.2). Additionally, we propose two variations resulting from the combination of the previously mentioned methods (sections 3.3, 3.4). In section 3.5, we present experimental results which are discussed in section 3.6.

3.1. Joint Lucas-Kanade (JLK) method

S. Birchfield and S. Pundlik [7] proposed a combination of Lucas-Kanade and Horn-Schunck energy functionals respectively which resulted in an energy functional to be minimized for Joint Lucas-Kanade (JLK):

$$E_{JLK} = \sum_{i=1}^{N} \left(E_D(i) + \lambda_i E_S(i) \right), \tag{3.1}$$

where N is the number of pixels, and the data and smoothness terms are given by

$$E_D(i) = K_p * \left(\left(f\left(u_i, v_i; I\right) \right)^2 \right) \tag{3.2}$$

$$E_S(i) = \left(\left(u_i - \hat{u}_i \right)^2 + \left(v_i - \hat{v}_i \right)^2 \right) \tag{3.3}$$

where K_p is a suitable convolution kernel whose size determines the number of neighboring pixels to be aggregated and assigns appropriate weights to the pixels inside the window.

In these equations, the energy of pixel i is determined by how well its displacement $(u_i, v_i)^T$ matches the local image data, as well as how far the displacement deviates from the expected displacement $(\hat{u}_i, \hat{v}_i)^T$. Note that the expected displacement can be computed in any desired manner and is not necessarily required to be the average of the neighboring displacements. According to [7], they predict the motion displacement of a pixel by fitting an affine motion model to the displacements of the surrounding pixels, which are inversely weighted according to their distance to the pixel. They use a Gaussian weighting function on the distance, with σ = 10 pixels.

Differentiating E_{JLK} with respect to the displacements $(u_i, v_i)^T$, $i = 1, \ldots, N$, and setting the derivatives to zero, yields a large $2N \times 2N$ sparse matrix equation, whose $(2i - 1)$th and $(2i)$th rows are

$$Z_i \mathbf{u}_i = \mathbf{e}_i, \tag{3.4}$$

where

$$Z_i = \begin{bmatrix} \lambda_i + K_p * (I_x I_x) & K_p * (I_x I_y) \\ K_p * (I_x I_y) & \lambda_i + K_p * (I_y I_y) \end{bmatrix}, \quad \mathbf{e}_i = \begin{bmatrix} \lambda_i \hat{u}_i - K_p * (I_x I_t) \\ \lambda_i \hat{v}_i - K_p * (I_y I_t) \end{bmatrix}.$$

This sparse system of equations can be solved using Jacobi iterations of the form

$$u_i^{(k+1)} = \hat{u}_i^{(k)} - \frac{J_{xx}\hat{u}_i^{(k)} + J_{xy}\hat{v}_i^{(k)} + J_{xt}}{\lambda_i + J_{xx} + J_{yy}} \tag{3.5}$$

$$v_i^{(k+1)} = \hat{v}_i^{(k)} - \frac{J_{xy}\hat{u}_i^{(k)} + J_{yy}\hat{v}_i^{(k)} + J_{yt}}{\lambda_i + J_{xx} + J_{yy}} \tag{3.6}$$

where $J_{xx} = K_p * (I_x^2)$, $J_{xy} = K_p * (I_x I_y)$, $J_{xt} = K_p * (I_x I_t)$, $J_{yy} = K_p * (I_y^2)$, and $J_{yt} = K_p * (I_y I_t)$.

To sum up, the (JLK) algorithm is presented in figure 3.1:

Algorithm: `Joint Lucas-Kanade`

1. For each pixel i,

    (a) Initialize $\mathbf{u}_i \leftarrow (0, 0)^T$
    (b) Initialize λ_i

2. For pyramid level $n - 1$ to 0 step −1,

    (a) For each pixel i, compute Z_i
    (b) Repeat until convergence:

        i. For each pixel i,
            (a) Determine $\hat{\mathbf{u}}_i$
            (b) Compute the difference I_t between the first image and the shifted second image: $I_t(x, y) = I_1(x, y) - I_2(x + u_i, y + v_i)$
            (c) Compute $\mathbf{e}_i$
            (d) Solve $Z_i \mathbf{u'}_i = \mathbf{e}_i$ for incremental motion $\mathbf{u'}_i$
            (e) Add incremental motion to overall estimate: $\mathbf{u}_i \leftarrow \mathbf{u}_i + \mathbf{u'}_i$
    (c) Expand to the next level: $\mathbf{u}_i \leftarrow k\mathbf{u}_i$, where k is the pyramid scale factor

Figure 3.1: The Joint Lucas-Kanade algorithm [7].

3.2. Optical flow with adaptive smoothing

H. Nagel and W. Enkelmann proposed in [32] to adaptively introduce smoothness constraints into the problem of optical flow.

We recall the basic optical flow equation of Horn-Schunck [25]:

$$\min_{u,v}\left\{\iint\left(\left(\nabla I^T\mathbf{u}+I_t\right)^2+\lambda\left(u_x^2+u_y^2+v_x^2+v_y^2\right)\right)dx\,dy\right\}. \tag{3.7}$$

where $\mathbf{u} = (u, v)^{\mathrm{T}}$ and $\nabla\mathbf{u}=\begin{pmatrix}u_x & v_x\\ u_y & v_y\end{pmatrix}$, represents the matrix of partial derivatives of the displacement vector components with respect to the image coordinates. The second term in (3.7) represents the smoothness requirement introduced by Horn and Schunck [25]. Parameter λ denotes the strength of the smoothness requirement relative to the first term.

Horn and Schunck used one parameter λ, same for all the pixels. This means that one pixel *i*, inside a texture and one pixel *j*, on the borders of an object use the same smoothness constraint. As a result, for pixel *i* the estimated optical flow is computed well, but for the pixel *j*, which is located on an edge of an object, the estimated optical flow tend to lose its accuracy because it expands its flow around that edge.

Therefore, the main idea was to introduce a weight matrix $\mathbf{C}^{-1}$ into the smoothness term, whose purpose is to give zero weight for pixels located on edges and greater values than zero for other pixels located inside textured areas. So, in that case, the optical flow problem becomes:

$$\min_{u,v}\left\{\iint\left(\left(\nabla I^T\mathbf{u}+I_t\right)^2+\lambda\ \mathrm{trace}\left(\left(\nabla\mathbf{u}\right)^T\mathbf{C}^{-1}\left(\nabla\mathbf{u}\right)\right)\right)dx\,dy\right\}. \tag{3.8}$$

where

$$\mathbf{C}^{-1}=\frac{F}{\det F}\quad\text{and}\quad F=\left\{\begin{pmatrix}I_y\\ -I_x\end{pmatrix}\begin{pmatrix}I_y\\ -I_x\end{pmatrix}^T+b^2\begin{pmatrix}I_{yy} & -I_{xy}\\ -I_{xy} & I_{xx}\end{pmatrix}\begin{pmatrix}I_{yy} & -I_{xy}\\ -I_{xy} & I_{xx}\end{pmatrix}^T\right\}. \tag{3.9}$$

The factor b^2 denotes the relative weight of the two contributions. If we carefully examine matrix F we will see that it is a 2 x 2 matrix and after some manipulation we obtain:

$$F = \begin{pmatrix} I_y^2 + b^2\left(I_{yy}^2 + I_{xy}^2\right) & -\left(I_x I_y + b^2\left(I_{xy} I_{yy} + I_{xy} I_{xx}\right)\right) \\ -\left(I_x I_y + b^2\left(I_{xy} I_{yy} + I_{xy} I_{xx}\right)\right) & I_x^2 + b^2\left(I_{xx}^2 + I_{xy}^2\right) \end{pmatrix} \hat{=} \begin{pmatrix} F_{11} & F_{12} \\ F_{12} & F_{22} \end{pmatrix}$$

At this point, we have to find the solution to the minimization problem of (3.8). Firstly, let us rewrite the problem in a more convenient way:

$$\begin{aligned} &\min_{u,v}\left\{E\left(u, v, u_x, u_y, v_x, v_y\right)dxdy\right\} \Leftrightarrow \\ &\min_{u,v}\iint \left(I_x u + I_y v + I_t\right)^2 + \lambda\left(F_{11}\, u_x^2 + 2F_{12}\, u_x\, u_y + F_{22}\, u_y^2\right. \\ &\qquad\qquad \left. + F_{11}\, v_x^2 + 2F_{12}\, v_x\, v_y + F_{22}\, v_y^2\right)dx\, dy \ , \end{aligned} \tag{3.10}$$

The solution of (3.10) is obtained by using the Calculus of Variations theory, (see Appendix C for details) where the related Euler-Lagrange equations are

$$\begin{cases} \min_u\left\{\dfrac{\partial E}{\partial u} - \dfrac{d}{dx}\left(\dfrac{\partial E}{\partial u_x}\right) - \dfrac{d}{dy}\left(\dfrac{\partial E}{\partial u_y}\right)\right\} = 0, & \text{(3.11a)} \\ \min_v\left\{\dfrac{\partial E}{\partial v} - \dfrac{d}{dx}\left(\dfrac{\partial E}{\partial v_x}\right) - \dfrac{d}{dy}\left(\dfrac{\partial E}{\partial v_y}\right)\right\} = 0 & \text{(3.11b)} \end{cases}$$

From equation (3.11a), in order to find a solution for u we have to compute the following expressions

- $\dfrac{\partial E}{\partial u} = 2\left(I_x u + I_y v + I_t\right) I_x$
- $\dfrac{\partial E}{\partial u_x} = 2F_{11}\lambda u_x + 2F_{12}\lambda u_y$
  - $\dfrac{d}{dx}\left(\dfrac{\partial E}{\partial u_x}\right) = 2F_{11}\lambda u_{xx} + 2F_{12}\lambda u_{xy}$
- $\dfrac{\partial E}{\partial u_y} = 2F_{22}\lambda u_y + 2F_{12}\lambda u_x$

- $\dfrac{d}{dy}\left(\dfrac{\partial E}{\partial u_y}\right) = 2F_{22}\lambda u_{yy} + 2F_{12}\lambda u_{xy}$

Substituting the above expressions into (3.11a) we come up with the following equation:

$$I_x^2 u + I_x I_y v = -I_x I_t + \lambda\left(F_{11} u_{xx} + F_{22} u_{yy} + F_{12} u_{xy}\right).$$

By repeating the same procedure for equation (3.11b) we find a similar equation for *v*. Finally, we end up with the following linear system:

$$\begin{cases} I_x^2 u + I_x I_y v = -I_x I_t + \lambda\left(F_{11} u_{xx} + F_{22} u_{yy} + F_{12} u_{xy}\right) \\ I_x I_y u + I_y^2 v = -I_y I_t + \lambda\left(F_{11} v_{xx} + F_{22} v_{yy} + F_{12} v_{xy}\right) \end{cases}$$

A usual approach to solve the above linear system is to proceed iteratively. For the computation of $u^{(t+1)}$ and $v^{(t+1)}$ at step $(t+1)$ we employ their derivatives computed at time step *t*.

$$\begin{cases} I_x^2 u^{(t+1)} + I_x I_y v^{(t+1)} = -I_x I_t + \lambda\left(F_{11} u_{xx}^{(t)} + F_{22} u_{yy}^{(t)} + F_{12} u_{xy}^{(t)}\right) \\ I_x I_y u^{(t+1)} + I_y^2 v^{(t+1)} = -I_y I_t + \lambda\left(F_{11} v_{xx}^{(t)} + F_{22} v_{yy}^{(t)} + F_{12} v_{xy}^{(t)}\right) \end{cases} \Leftrightarrow$$

$$\begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \begin{bmatrix} u^{(t+1)} \\ v^{(t+1)} \end{bmatrix} = \begin{bmatrix} -I_x I_t + \lambda\left(F_{11} u_{xx}^{(t)} + F_{22} u_{yy}^{(t)} + F_{12} u_{xy}^{(t)}\right) \\ -I_y I_t + \lambda\left(F_{11} v_{xx}^{(t)} + F_{22} v_{yy}^{(t)} + F_{12} v_{xy}^{(t)}\right) \end{bmatrix} \hat{=}\ \hat{A}x^{(t+1)} = \hat{b}^{(t)}, \quad (3.12)$$

In (3.12), if $F_{11} = F_{22} = 1$ and $F_{12} = 0$ we obtain the linear system we have at the Horn and Schunck scenario [25]. Finally, the solution to (3.12) is obtained by solving the previous iterative scheme.

3.3. Combination of JLK and adaptive smoothing

The first method we propose in this chapter is the combination of the two previously mentioned methods of S. Birchfield – S. Pundlik [7] and H.Nagel–

W. Enkelmann [32]. The main idea is to keep the proposed scheme of Nagel *et al.* [32], in order to have adaptive smoothness constraints and modifying it by adding the neighboring area proposed by Lucas-Kanade [28] and then also used by S. Birchfield–S. Pundlik [7]. In other words, the linear system (3.12) proposed in section 3.2 now becomes:

$$\begin{bmatrix} J_{xx} & J_{xy} \\ J_{xy} & J_{yy} \end{bmatrix} \begin{bmatrix} u^{(t+1)} \\ v^{(t+1)} \end{bmatrix} = \begin{bmatrix} -J_{xt} + \lambda \left(F_{11}\, u_{xx}^{(t)} + F_{22}\, u_{yy}^{(t)} + F_{12}\, u_{xy}^{(t)} \right) \\ -J_{yt} + \lambda \left(F_{11}\, v_{xx}^{(t)} + F_{22}\, v_{yy}^{(t)} + F_{12}\, v_{xy}^{(t)} \right) \end{bmatrix} \tag{3.13}$$

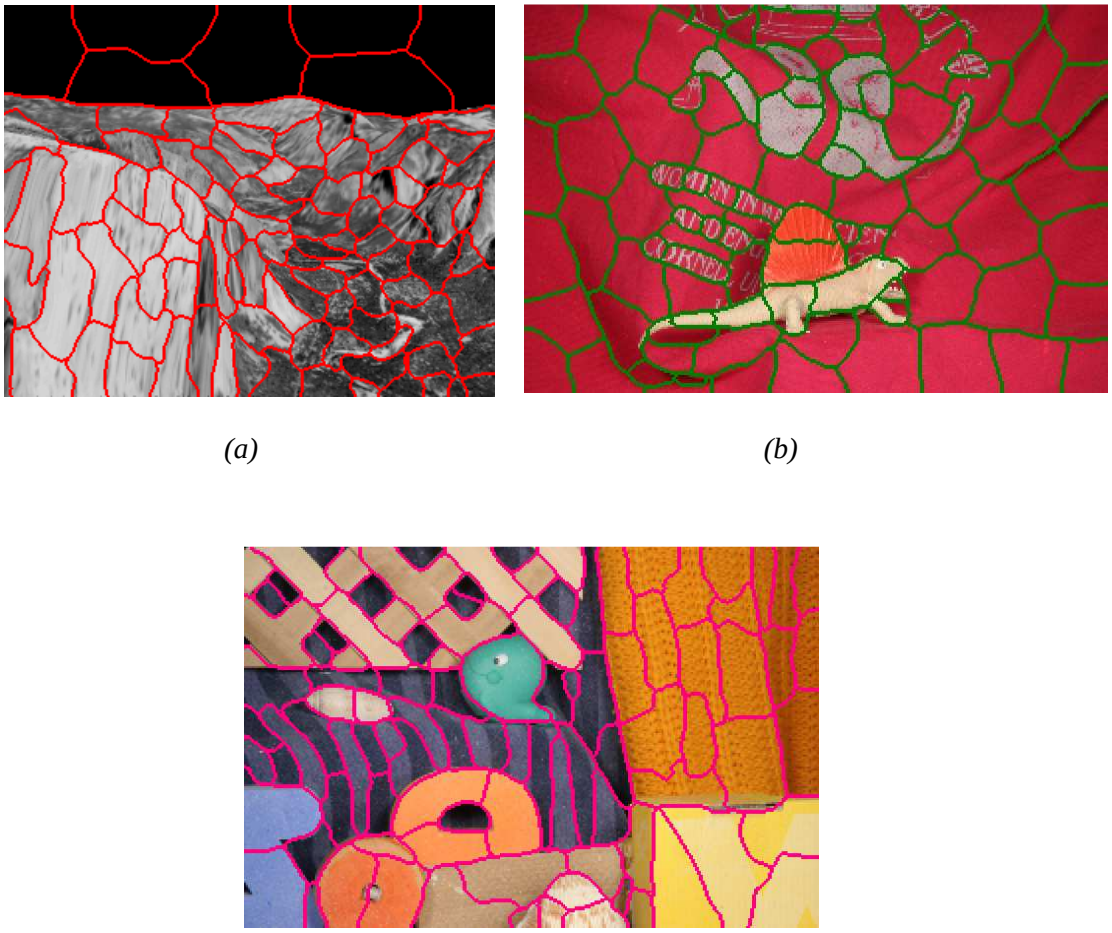
where $J_{xx} = K_p * (I_x^2)$, $J_{xy} = K_p * (I_x I_y)$, $J_{xt} = K_p * (I_x I_t)$, $J_{yy} = K_p * (I_y^2)$, and $J_{yt} = K_p * (I_y I_t)$ and K_p is a suitable convolution kernel whose size determines the number of neighboring pixels to be aggregated and assigns appropriate weights to the pixels inside the window. For our experiments, which we show in section 3.5, we use a 7x7 average kernel.

3.4. Guiding optical flow using segmentation

The second method we propose in this chapter is a variation of the method described in section 3.3. The innovation here is that we "carefully" choose which of the neighboring pixels are going to participate into the convolution matrix K_ρ. The choice is taken by examining if the neighboring pixel i' belongs to the same super-pixel with the current pixel i. If it doesn't belong to the same super-pixel, then the value of K_ρ for that neighboring pixel is equal to zero.

Here appears the need to analyze what we mean by the term "super-pixel" and how it is produced. It is common to use the term super-pixel in order to name a unit – a piece from the result of the procedure called image segmentation. Another name you may be seen in bibliography instead of super-pixel is segment. Additionally, image segmentation is the procedure in which we group together pixels of an image that appear to have the same features (or simpler the same behaviour). The most common feature that is used in this scientific area is the intensity value of the pixel.

An example of image segmentation is shown in figure 3.2.



(a) *(b)*

(c)

Figure 3.2: Various image segmentations. (a) Yosemite's without clouds, (b) Dimetrodon's and (c) Rubberwhale's image segmentation.

In our experiments, the super-pixels were produced by using the method purposed in [38], where G. Mori proposed a method based on normalized cuts (spectral clustering). For each experiment (except from the trivial artificial images of sections 3.5.1-3.5.3 where the number of the super-pixels does not affect the result), we are showing various combinations between the window size and the number of the super-pixels at the appendix E.

3.5. Experimental Results and Discussion

The proposed methods were tested on image sequences including both synthetic and real scenes. Some of the synthetic images were synthesized from us and other were taken from publicly available data sets as for example the Middlebury public flow dataset [3], [4, 29] in order to guarantee the objectiveness of our evaluations.

More specifically, we tested our method on three synthetic sequences. The first sequence is showing a *Textured Square* moving from the center to the top left corner by one pixel. The second one is showing two *Textured Triangles*. The upper left triangle moves about one pixel to the bottom left corner, while the bottom right triangle moves about one pixel to the bottom right corner. The third one is showing two *Textured Triangles* with the only difference from the second synthetic sequence that here the bottom right triangle moves about 2 pixels to the bottom right corner. The background color for all the previously mentioned sequences is black, without loss of generality.

Additionally, we tested our methods on the well-known *Yosemite* sequence without clouds, the *Dimetrodon* sequence and the *Rubberwhale* sequence [3] (which contain hidden texture ~ occlussions). We compared our approaches with the algorithms of Pundlik's method [7] and Nagel's method [32]. For the evaluation of our method we used the error metrics described in section 2.4.

3.5.1. Textured-Square Sequence

This is a simple 256 x 256 example, which consists of a textured square located at the center of the first frame, while at the second frame it moves by one pixel towards the top left corner. Figure 3.3 shows a frame of the image and figure 3.4 shows optical flow estimations from the compared methods along with the ground truth. Figure 3.5 shows the angular error and figure 3.6 presents the flow by using color coding [3]. We do not show the end-point error for each pixel as it has too small values (but we show the average end-point error, which is equivalent and more meaningful).

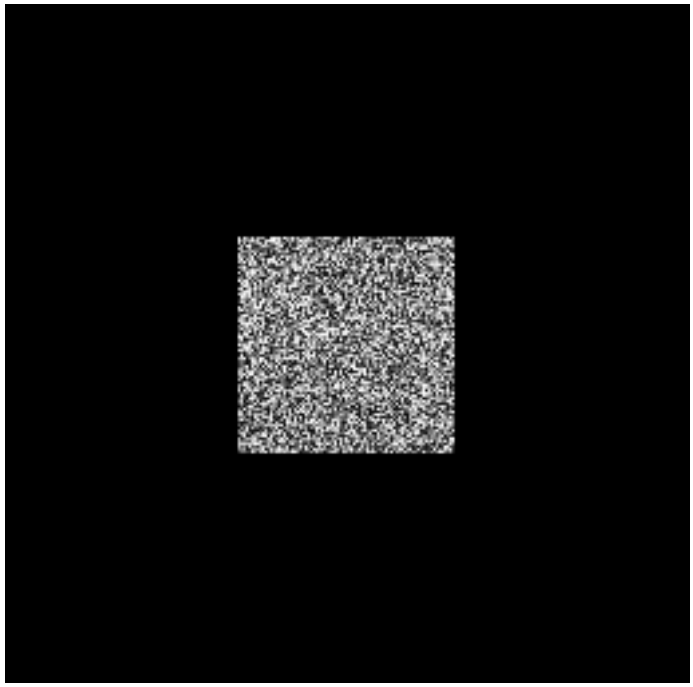
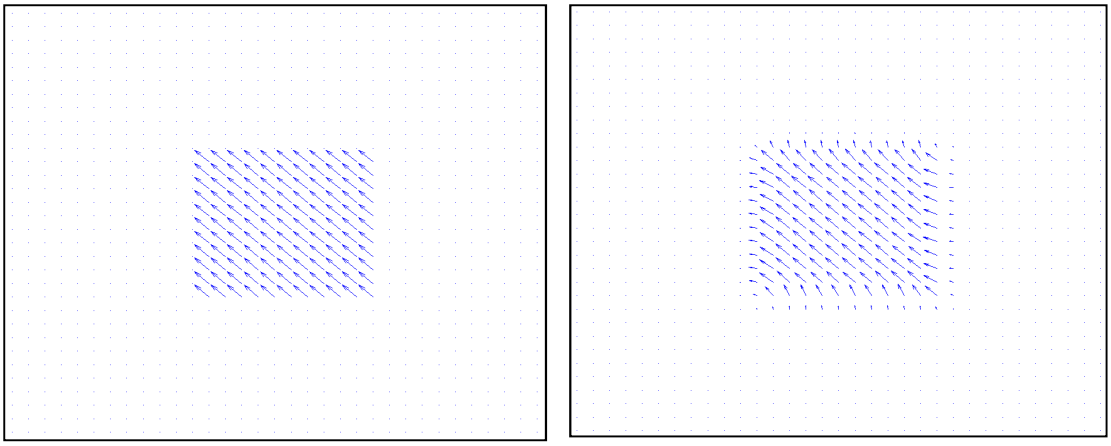
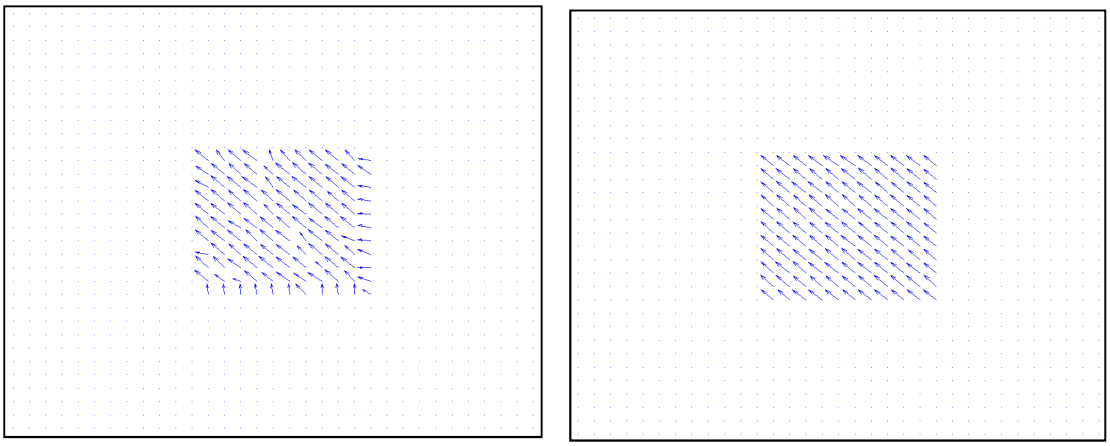


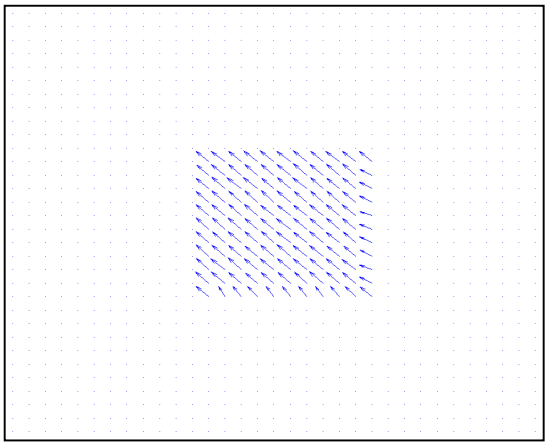
Figure 3.3: Textured-square sequence: first frame of the sequence.



(a) *(b)*

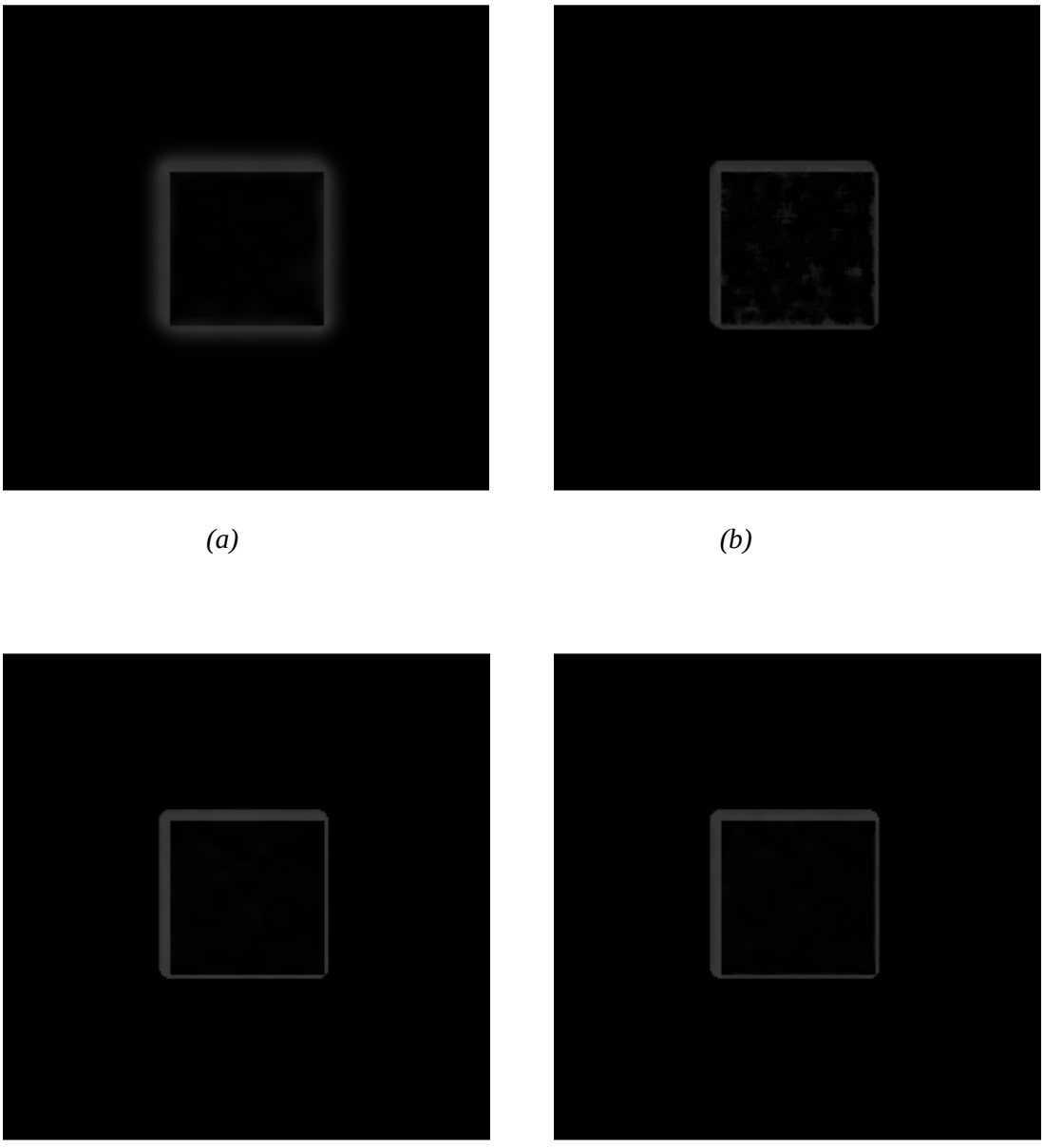


(c) *(d)*



(e)

Figure 3.4: Textured-square sequence: (a) ground truth optical flow, (b) optical flow using the JLK method [7], *(c) flow using the method of Nagel et al.* [32], *(d) resulting optical flow of the proposed method of section 3.3 and (e) resulting optical flow of the proposed method of section 3.4.*



(a) *(b)*

(c) *(d)*

Figure 3.5: Textured-square's Angular Error (AE) of the compared methods. (a) JLK method [7], *(b) method of Nagel et al.* [32] *(c) JLK with adaptive smoothing (section 3.3) and (d) guided optical flow using image segmentation (section 3.4).*

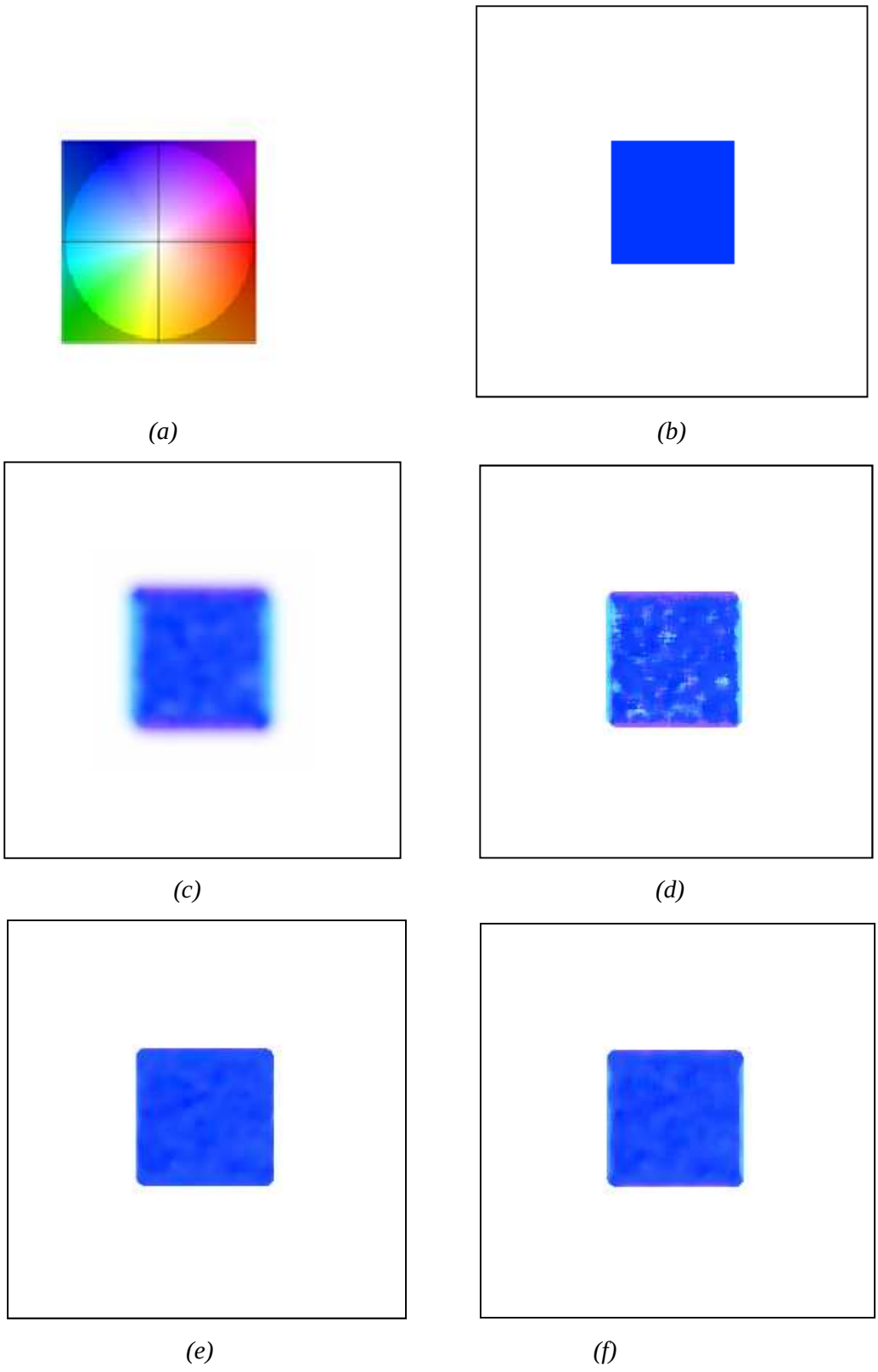


Figure 3.6: Textured-square sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) flow field using the JLK method [7], *(d) flow field using the method of Nagel et al.* [32], *(e) flow field using the method proposed in section 3.3 and (f) flow field using the method proposed in section 3.4.*

Table 3.1: Average error metrics for the Textured-square sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	3.09	0.08	0.08
Horn-Schunck [25]	1.84	0.04	0.04
Joint Lucas-Kanade [7]	2.67	0.04	0.05
Nagel et al. [32]	1.60	0.04	0.04
Method of section 3.3	1.50	0.05	0.04
Method of section 3.4	**1.46**	**0.04**	**0.04**

As we can see from table 3.1 our approaches achieve smaller errors than Nagel's *et al.* [32] and Joint Lucas-Kanade method [7], for all the error metrics.

3.5.2. Textured-Triangles with equal in Norm Moves

This is a slightly more complicated 256 x 256 example, which consists of two textured triangles located at the top left corner and at the bottom right corner of the first frame, while at the second frame the upper left triangle moves by one pixel to the bottom left corner, while the bottom right triangle moves by one pixel to the bottom right corner. Figure 3.7 shows the image and figure 3.8 the estimated optical flows. Figure 3.9 shows the angular error and figure 3.10 presents the flow by using color coding [3]. We do not show the end-point error for each pixel as it has too small values (but we show the average end-point error, which is equivalent and more meaningful).

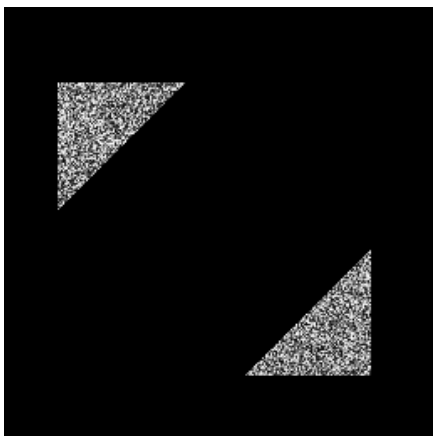


Figure 3.7: Textured-triangles (with equal in norm moves) sequence: first frame of the sequence.

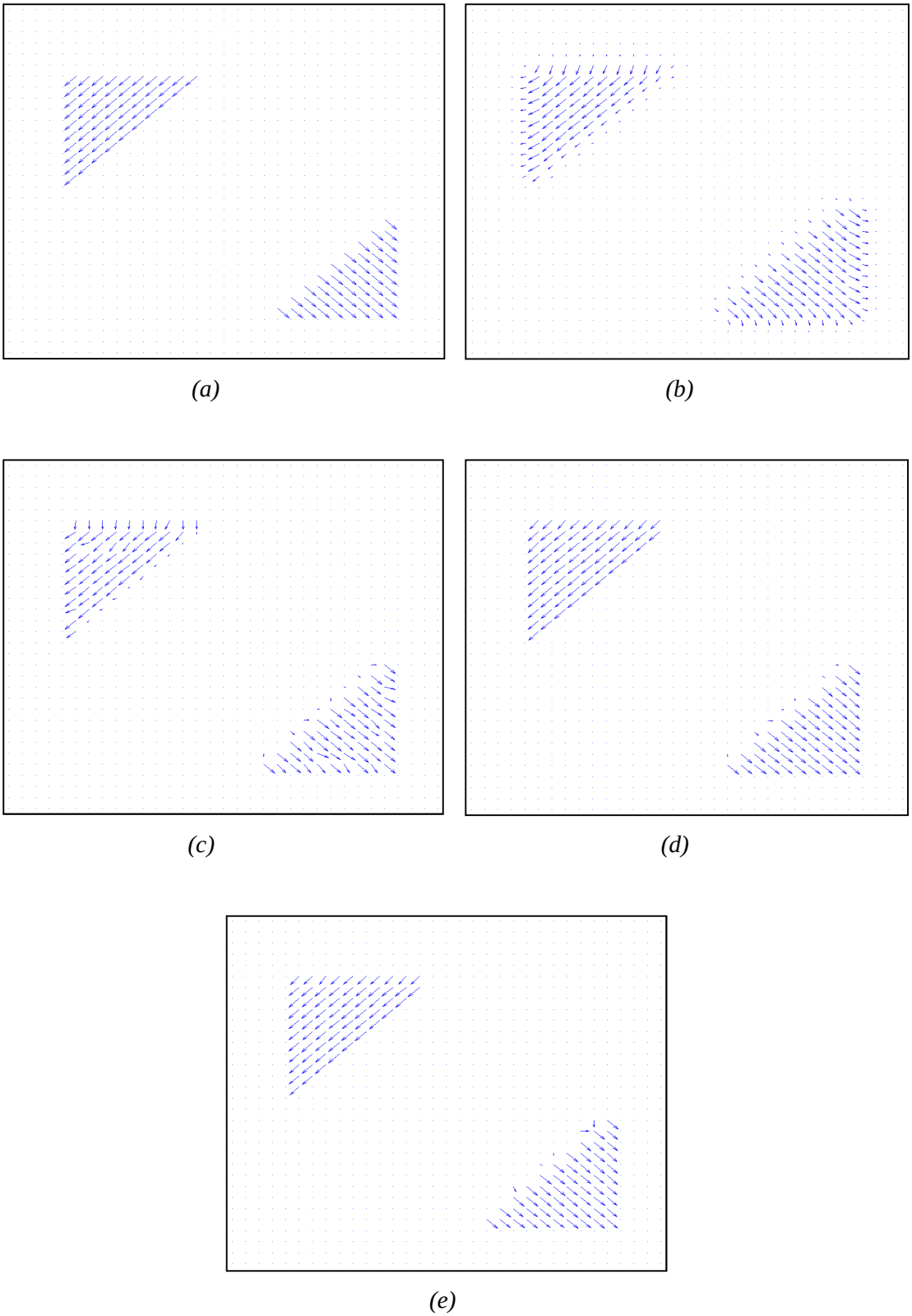
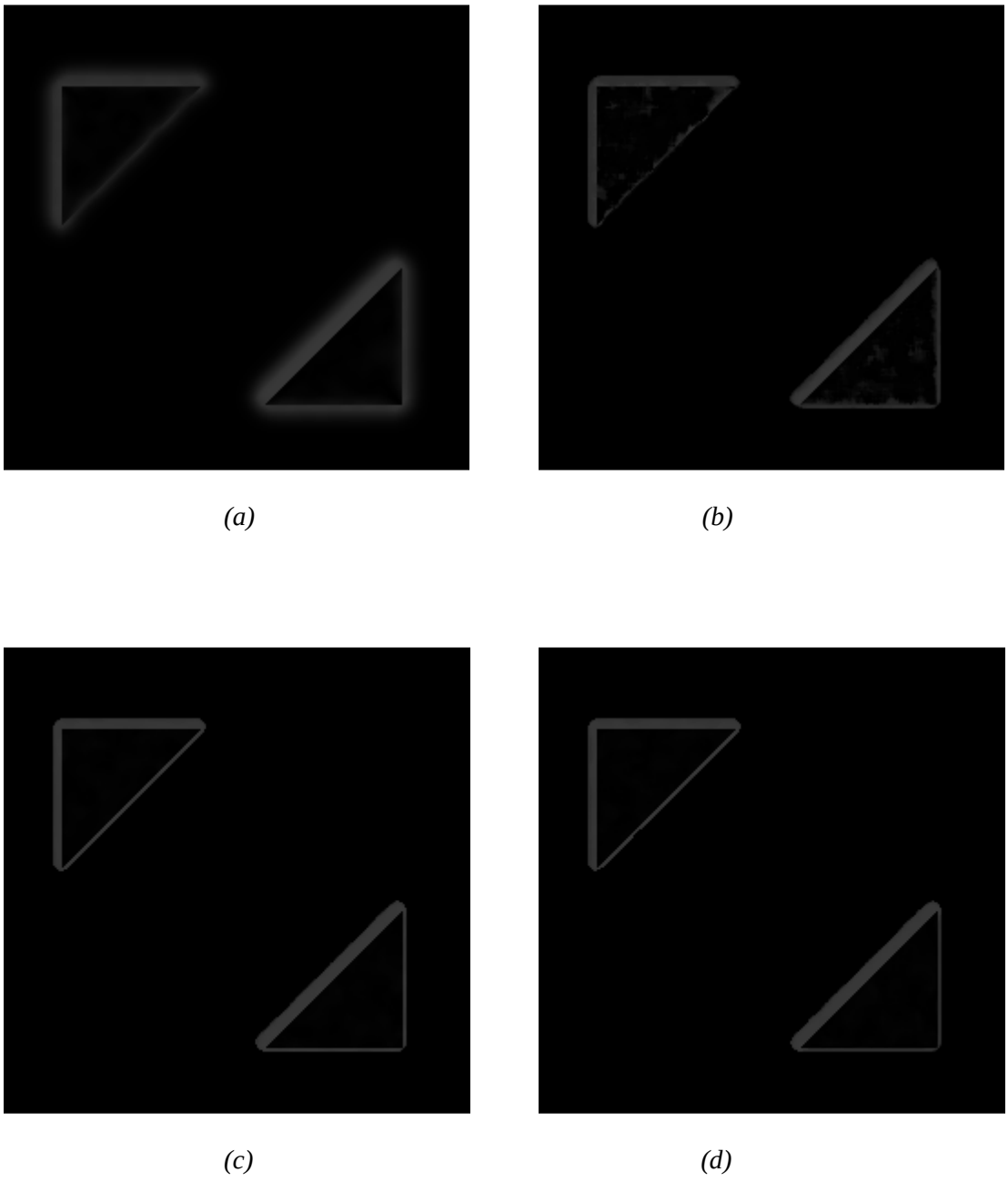


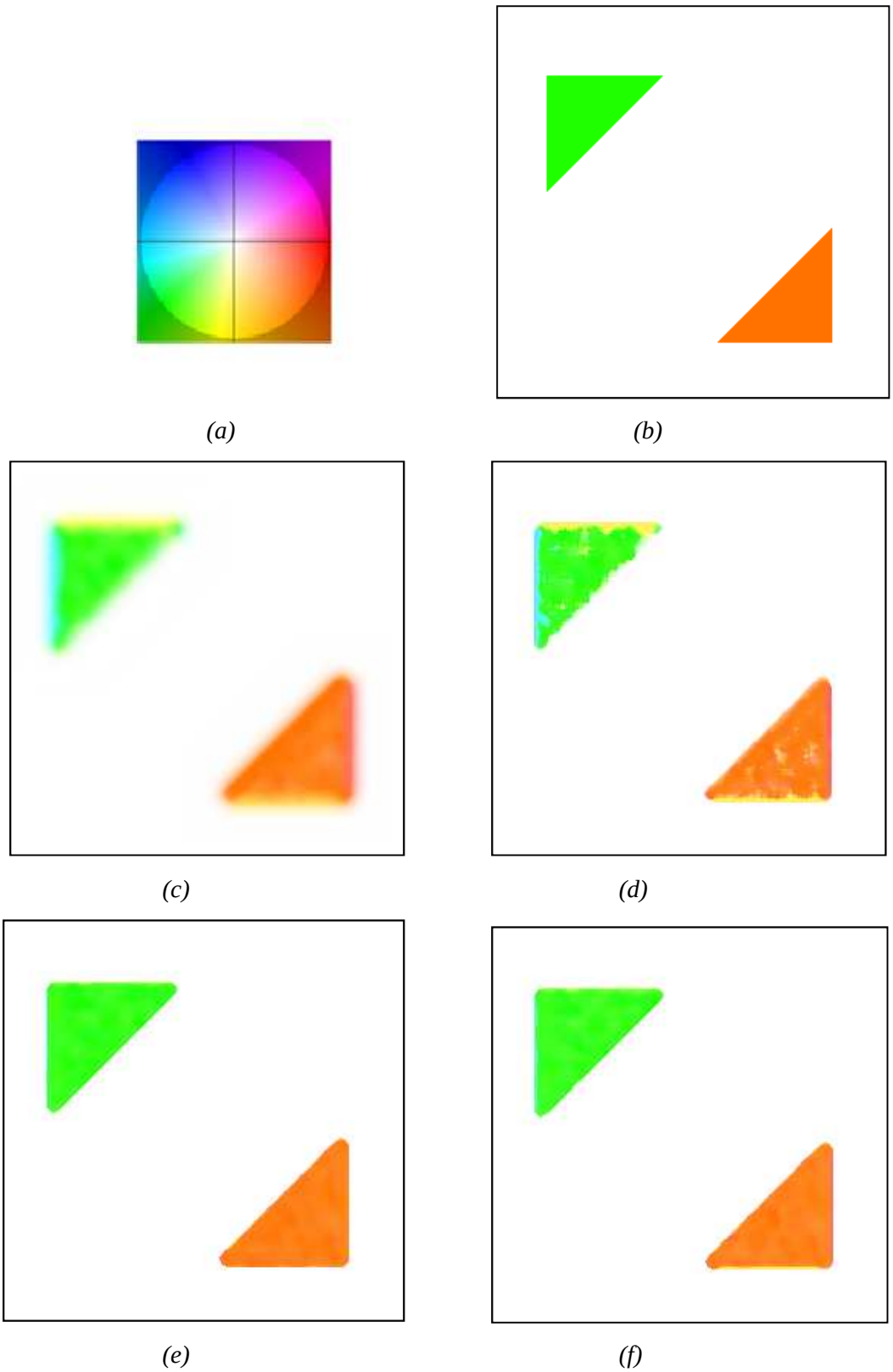
Figure 3.8: Textured-triangles (with equal in norm moves) sequence: (a) ground truth optical flow, (b) optical flow using the JLK method [7], *(c) flow using the method of Nagel et al.* [32], *(d) resulting optical flow of the proposed method of section 3.3 and (e) resulting optical flow of the proposed method of section 3.4.*



(a) *(b)*

(c) *(d)*

Figure 3.9: Textured-triangles' (with equal in norm moves) Angular Error (AE) of the compared methods. (a) JLK method [7]*, (b) method of Nagel et al.* [32] *(c) JLK with adaptive smoothing (section 3.3) and (d) guided optical flow using image segmentation (section 3.4).*



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figure 3.10: Textured-triangles (with equal in norm moves) sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) flow field using the JLK method [7], *(d) flow field using the method of Nagel et al.* [32], *(e) flow field using the method proposed in section 3.3 and (f) flow field method proposed in section 3.4.*

Table 3.2: Average error metrics for the Textured-triangles (with equal in norm moves) sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	5.91	0.15	0.14
Horn-Schunck [25]	2.47	**0.05**	0.05
Joint Lucas-Kanade [7]	4.10	0.07	0.08
Nagel et al. [32]	**2.25**	0.06	0.05
Method of section 3.3	2.33	0.08	0.05
Method of section 3.4	2.26	0.07	**0.05**

As we can see from table 3.2 our approaches are slightly worse than Nagel's *et al.* approach [32] (~ 0.01 difference in *AAE*) although our results in fig. 3.10 are more coherent our lost comes because our methods expands the optical flow slightly outside the edges of the triangles, but better than Joint Lucas-Kanade method [7], for all the error metrics.

3.5.3. Textured-Triangles with unequal in Norm Moves

A next experiment consists in increasing the difficulty of the previous configurations. We have a 256 x 256 example, which consists of two textured triangles located at the top left corner and at the bottom right corner of the first frame, while at the second frame the upper left triangle moves by one pixel to the bottom left corner, while the bottom right triangle moves by two pixel to the bottom right corner. Figure 3.11 shows the image and figure 3.12 the estimated optical flows. Figure 3.13 shows the angular error and figure 3.14 presents the flow by using color coding [3]. We do not show the end-point error for each pixel as it has too small values (but we show the average end-point error, which is equivalent and more meaningful).

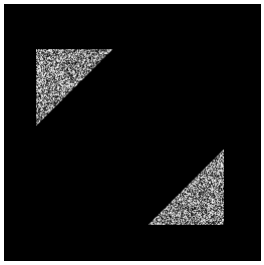
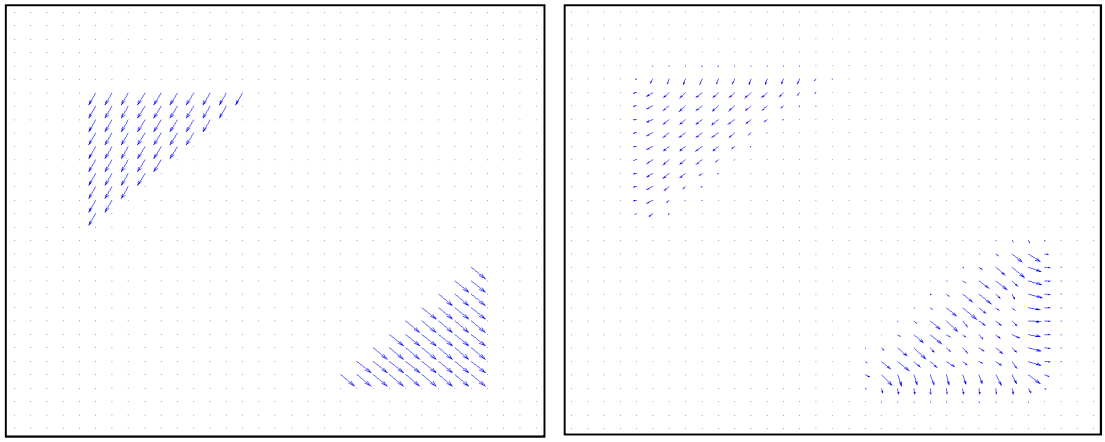
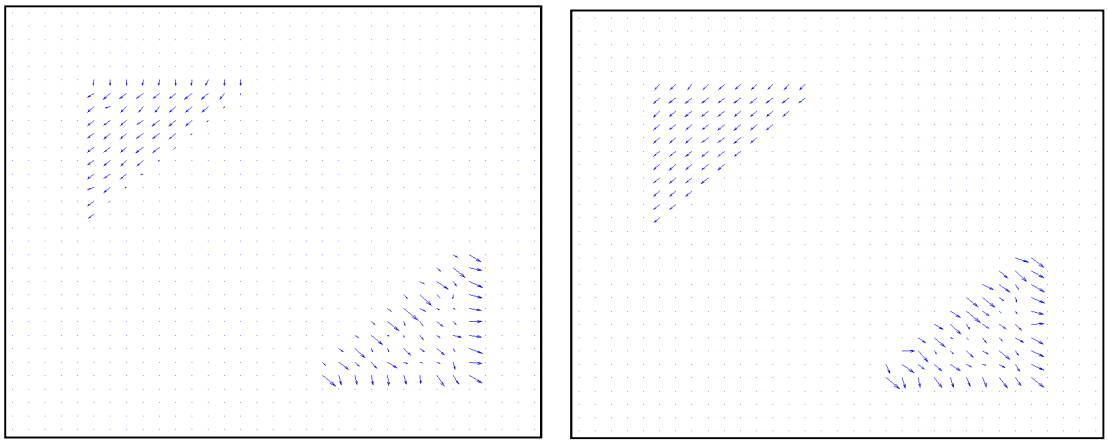


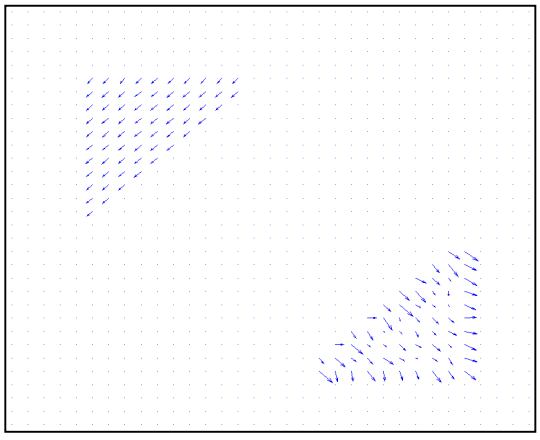
Figure 3.11: Textured-triangles (with unequal in norm moves) sequence: first frame of the sequence.



(a) *(b)*

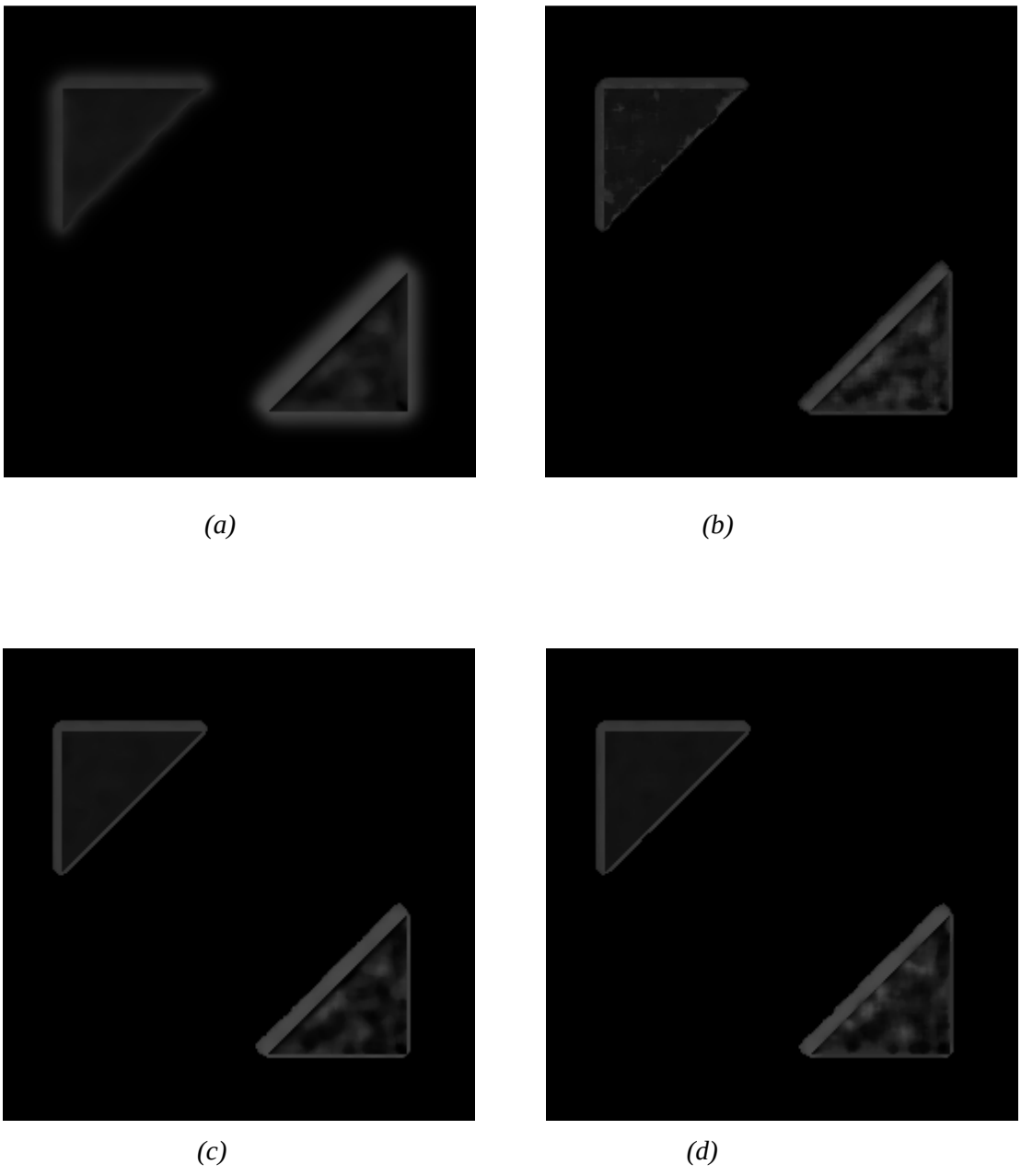


(c) *(d)*



(e)

Figure 3.12: Textured-triangles (with unequal in norm moves) sequence: (a) ground truth optical flow, (b) optical flow using the JLK method [7]*, (c) flow using the method of Nagel et al.* [32]*, (d) resulting optical flow of the proposed method of section 3.3 and (e) resulting optical flow of the proposed method of section 3.4.*



(a) *(b)*

(c) *(d)*

Figure 3.13: Textured-triangles' (with unequal in norm moves) Angular Error (AE) of the compared methods. (a) JLK method [7]*, (b) method of Nagel et al.* [32] *(c) JLK with adaptive smoothing (section 3.3) and (d) guided optical flow using image segmentation (section 3.4).*

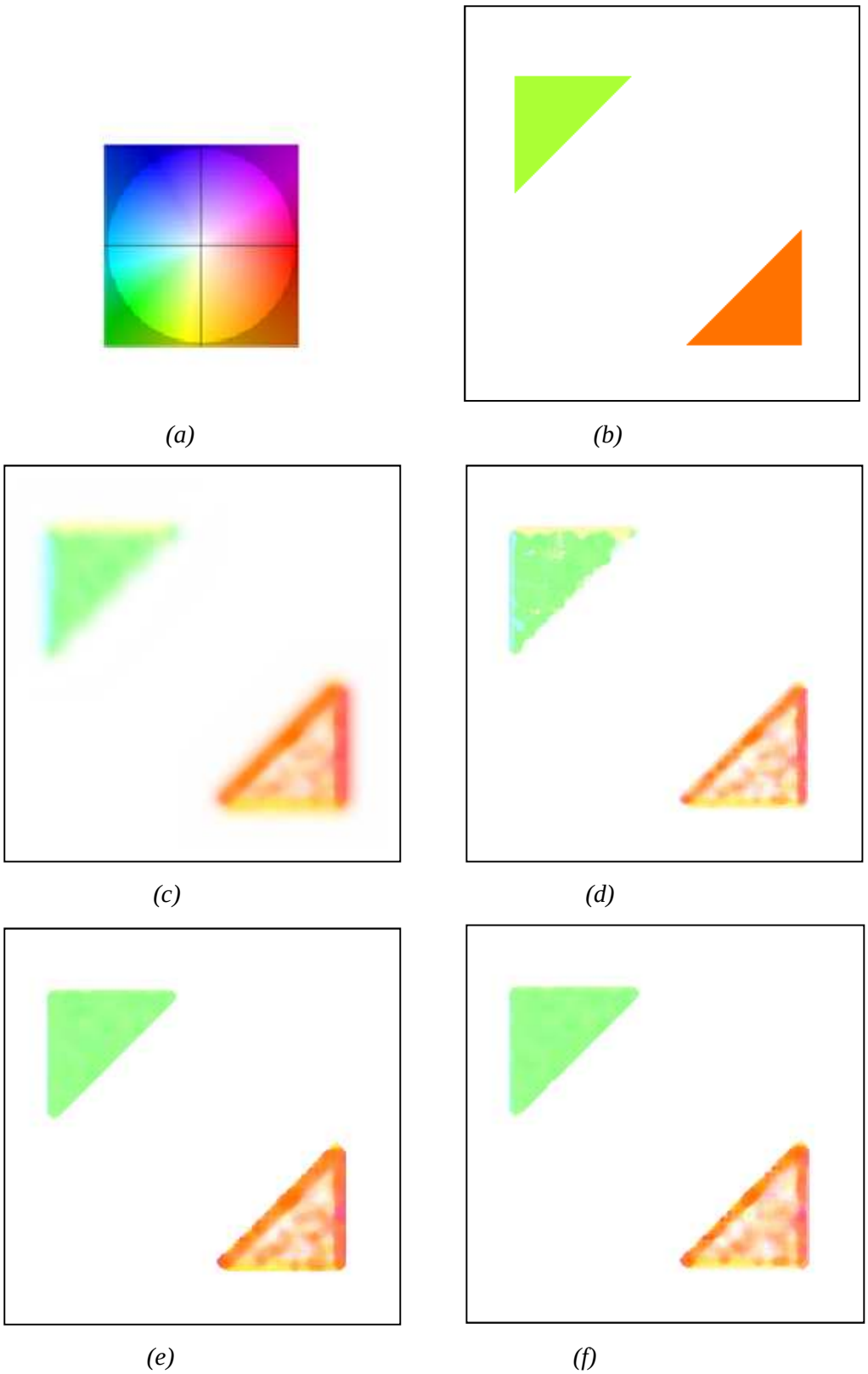


Figure 3.14: Textured-triangles (with unequal in norm moves) sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) flow field using the JLK method [7], *(d) flow field using the method of Nagel et al.* [32], *(e) flow field using the method proposed in section 3.3 and (f) flow field method proposed in section 3.4.*

Table 3.3: Average error metrics for the Textured-triangles (with unequal in norm moves) sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	8.58	0.17	0.26
Horn-Schunck [25]	5.57	0.14	0.19
Joint Lucas-Kanade [7]	6.95	0.18	0.22
Nagel et al. [32]	4.79	**0.13**	0.18
Method of section 3.3	**4.67**	0.17	**0.17**
Method of section 3.4	4.78	0.16	0.18

As we can see from table 3.3 our approach achieves smaller errors than Nagel's *et al.* approach [32] in *AAE* and *EP* and slightly worse in *AME*, while in comparison with the Joint Lucas-Kanade method [7] our methods performs better for all the error metrics, although all the methods did not have perfectly estimations.

3.5.4. Yosemite Sequence without Clouds

The *Yosemite* sequence without clouds, is available at `http://www.cs.brown.edu/people/black/images.html`.

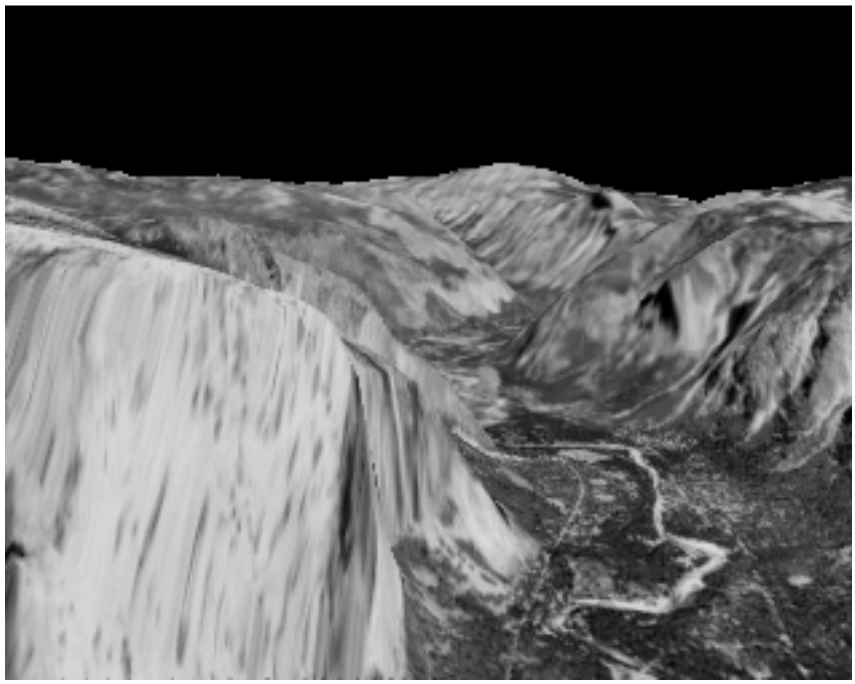


Figure 3.15: Yosemite sequence without clouds: first frame of the sequence.

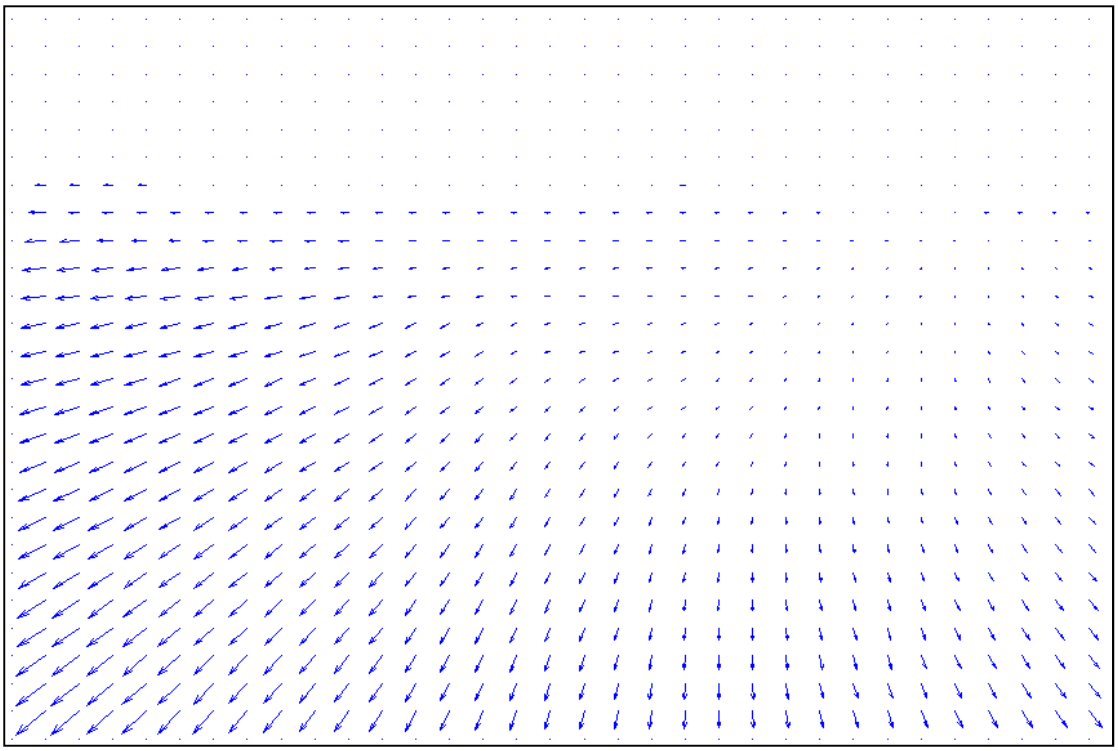


Figure 3.16: Yosemite sequence without clouds: ground truth optical flow.

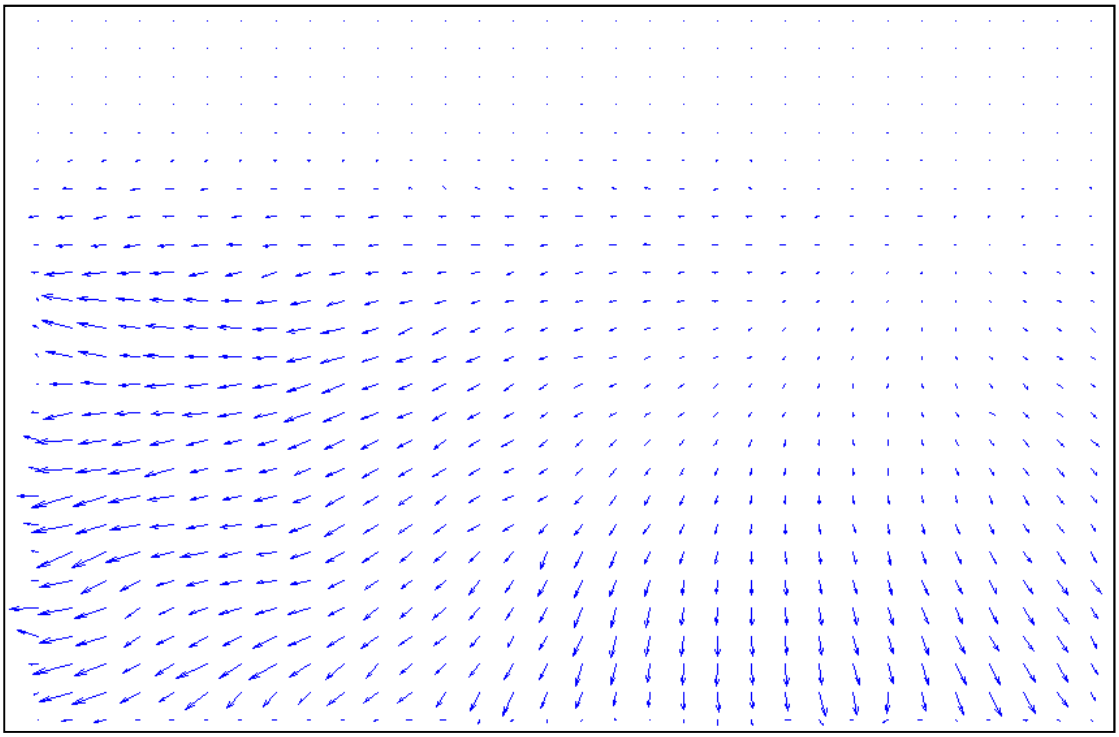


Figure 3.17: Yosemite sequence without clouds: optical flow using the JLK method [7].

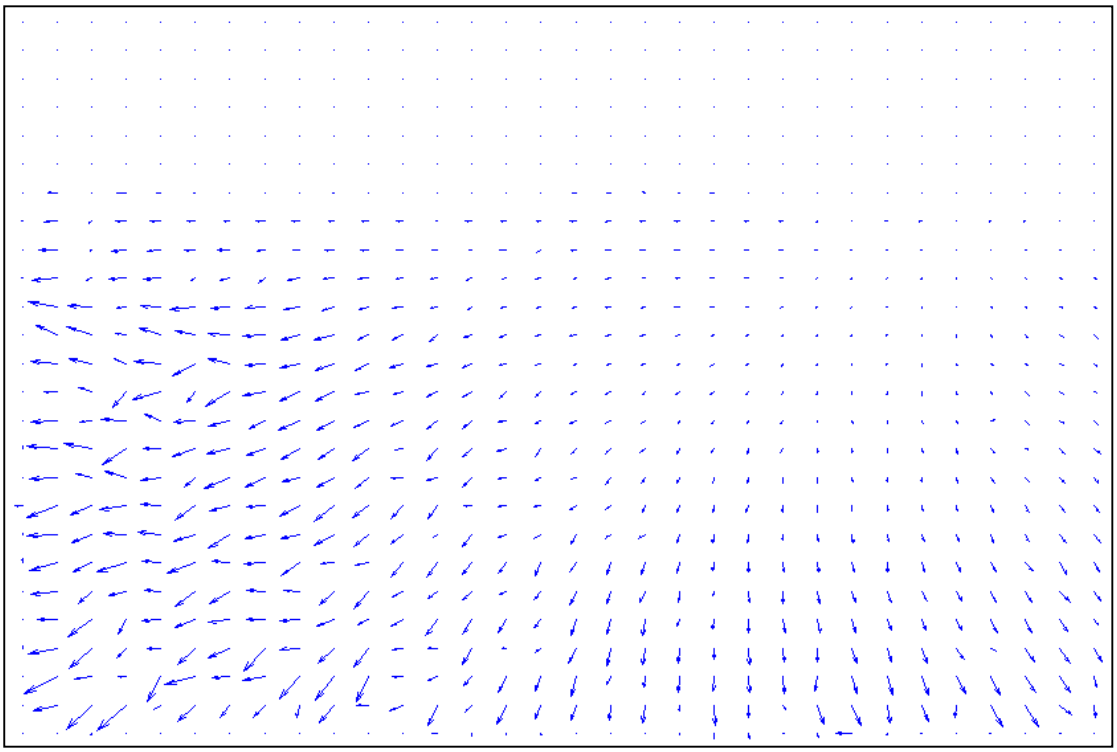


Figure 3.18: Yosemite sequence without clouds: optical flow using the method of Nagel et al. [32].

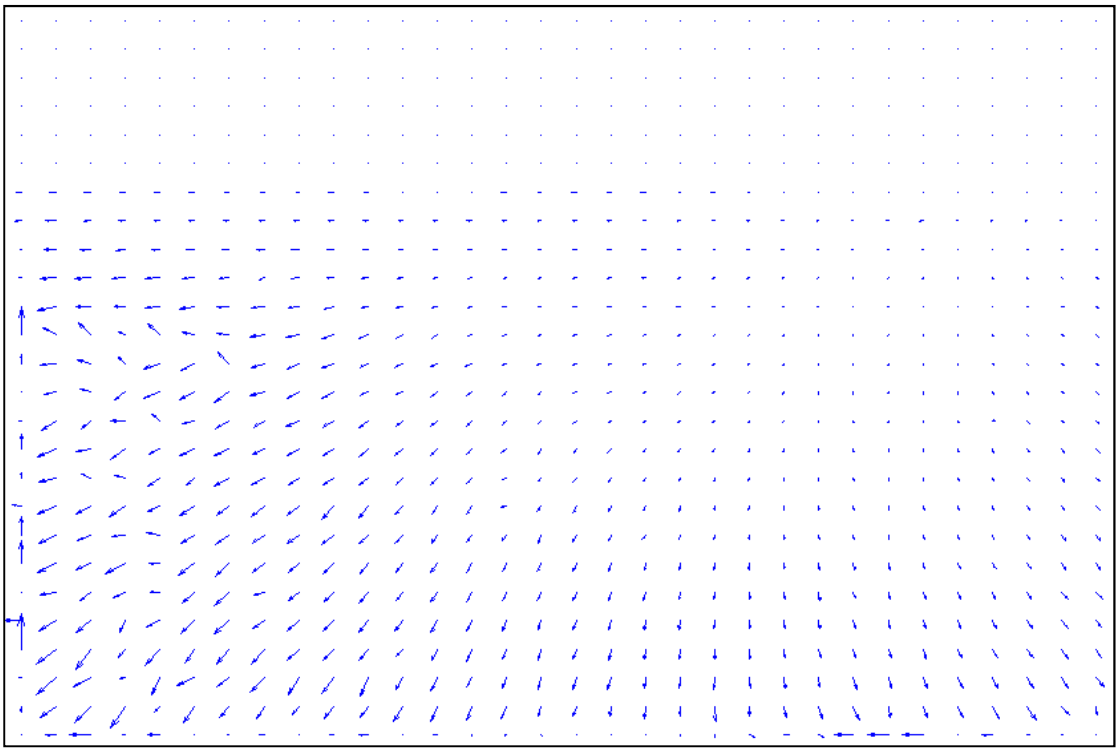


Figure 3.19: Yosemite sequence without clouds: resulting optical flow of the proposed method of section 3.3.

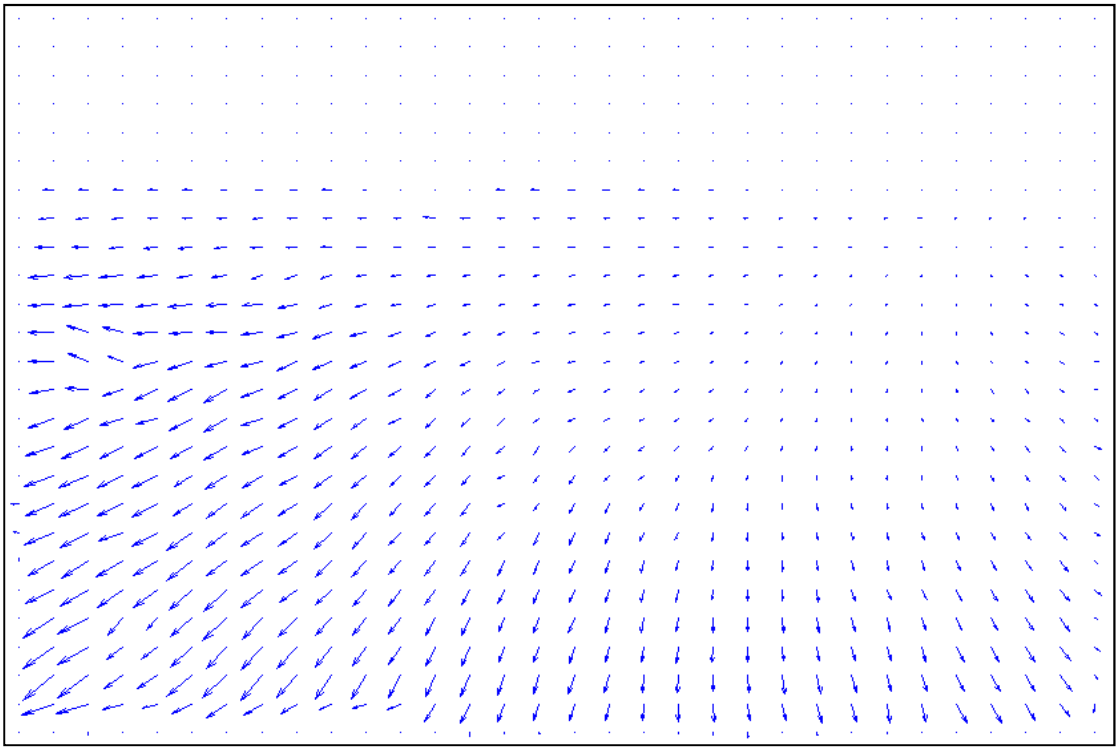


Figure 3.20: Yosemite sequence without clouds: resulting optical flow of the proposed method of section 3.4.

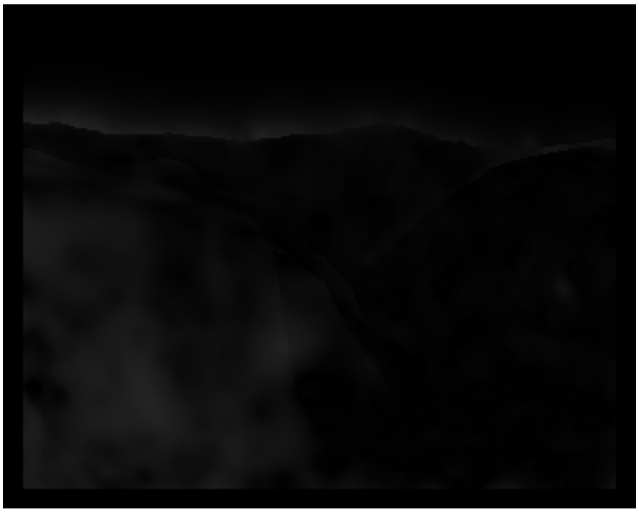
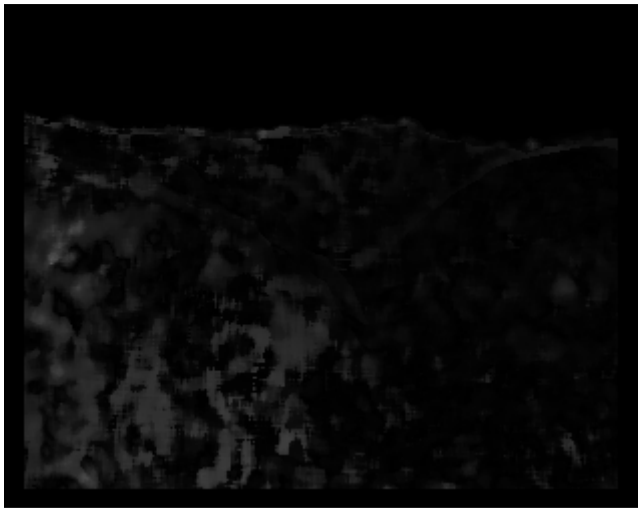
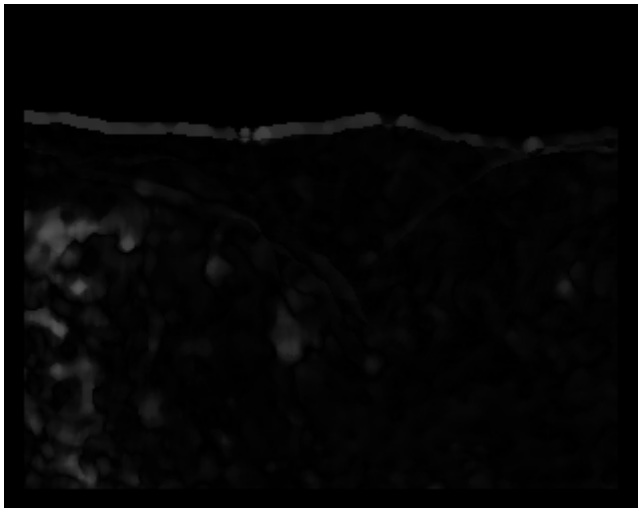


Figure 3.21: Yosemite without clouds' Angular Error (AE) of the JLK method [7].



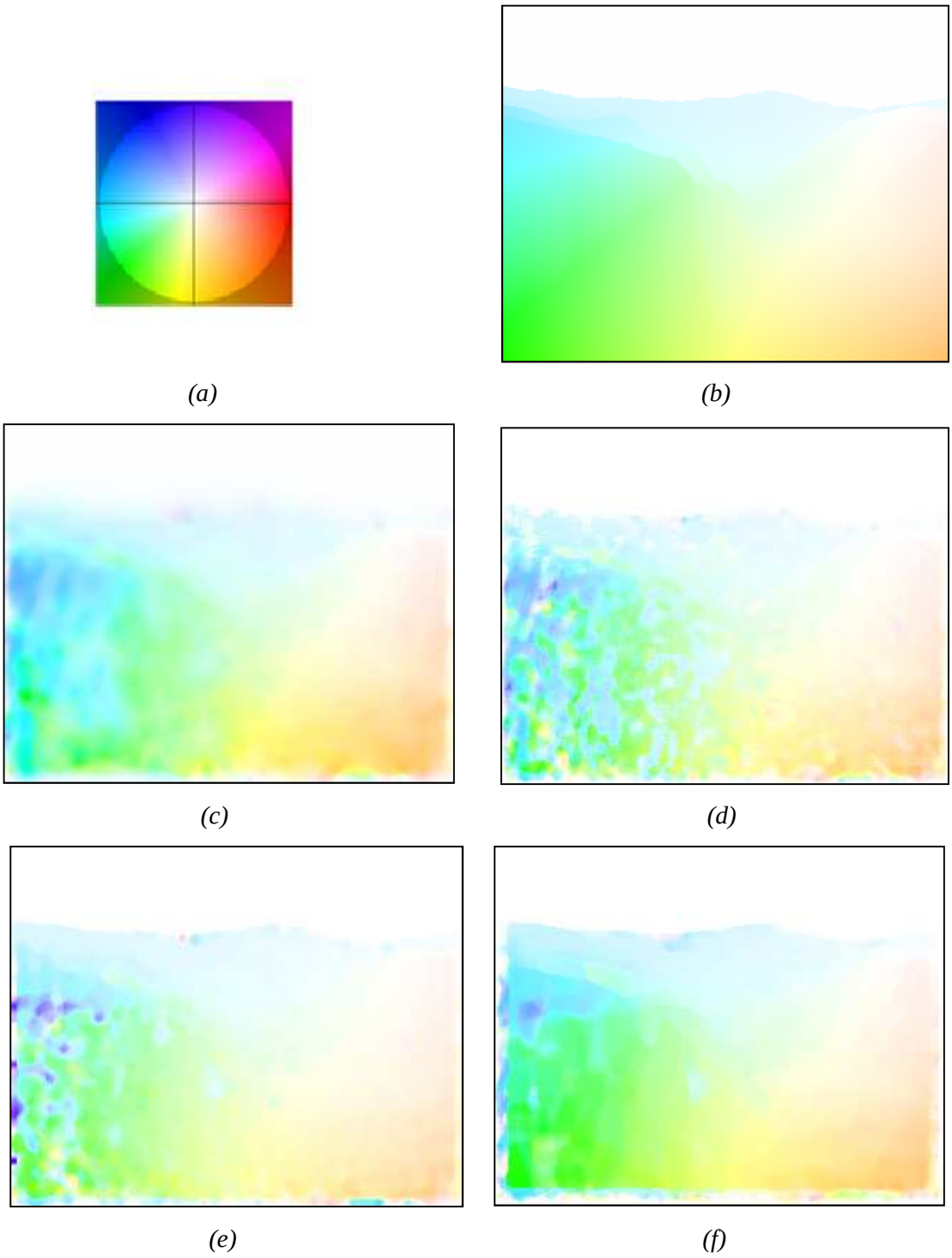
(a)



(b)

(c)

Figure 3.22: Yosemite without clouds' Angular Error (AE) of the compared methods. (a) Method of Nagel et al. [32], *(b) JLK with adaptive smoothing (section 3.3) and (c) guided optical flow using image segmentation (section 3.4).*



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figure 3.23: Yosemite sequence without clouds: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) flow field using the JLK method [7]*, (d) flow field using the method of Nagel et al.* [32]*, (e) flow field using the method proposed in section 3.3 and (f) flow field using the method proposed in section 3.4.*

Table 3.4: Average error metrics for the Yosemite without Clouds sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	11.65	0.23	0.48
Horn-Schunck [25]	5.43	0.10	0.20
Joint Lucas-Kanade [7]	7.97	0.17	0.35
Nagel et al. [32]	9.15	0.19	0.36
Method of section 3.3	5.12	0.12	0.22
Method of section 3.4	**3.79**	**0.09**	**0.15**

As we can see from the table 3.4 our approaches are better than Nagel's *et al.* method [32], JLK [7], LK [28] and HS [25] for all the error metrics. In order to obtain those results, we used 40 super-pixels and a 19x19 window, representing the neighborhood. See appendix E for more combinations between the number of the super-pixels and the window size.

3.5.5. Yosemite Sequence with Clouds

The original version of the *Yosemite* sequence with cloudy sky was created by Lynn Quam and is available at `ftp://ftp.csd.uwo.ca/pub/vision`.

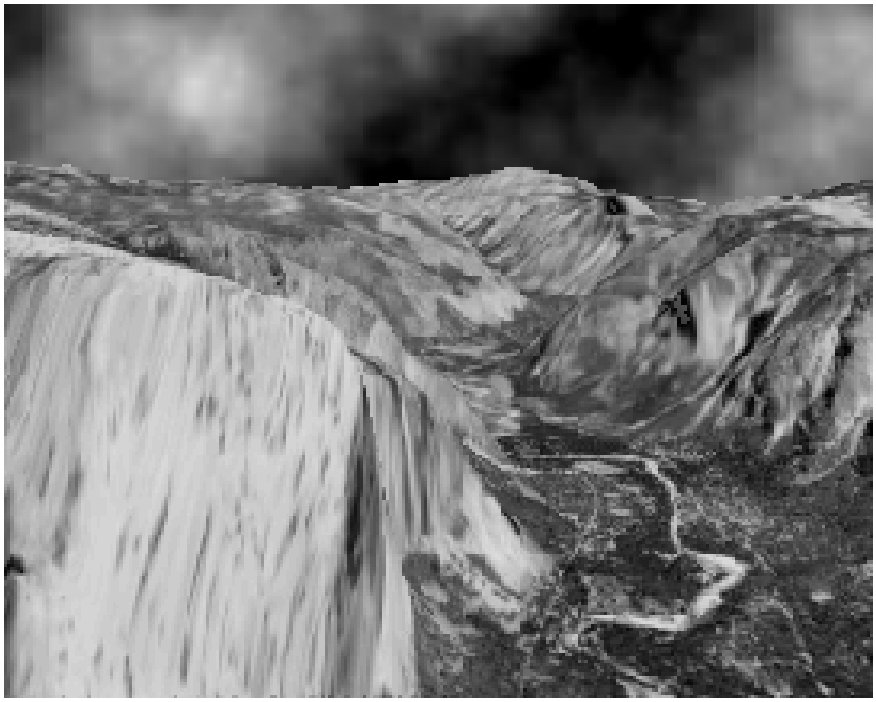


Figure 3.24: Yosemite sequence with clouds: first frame of the sequence.

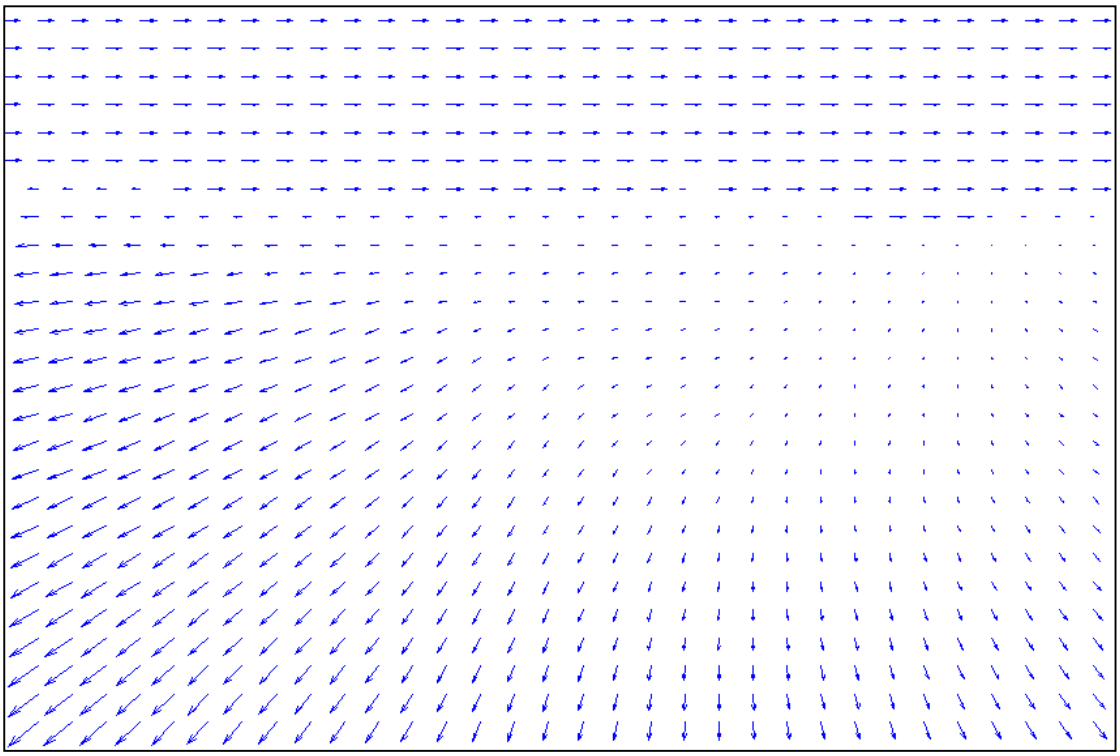


Figure 3.25: Yosemite sequence with clouds: ground truth optical flow.

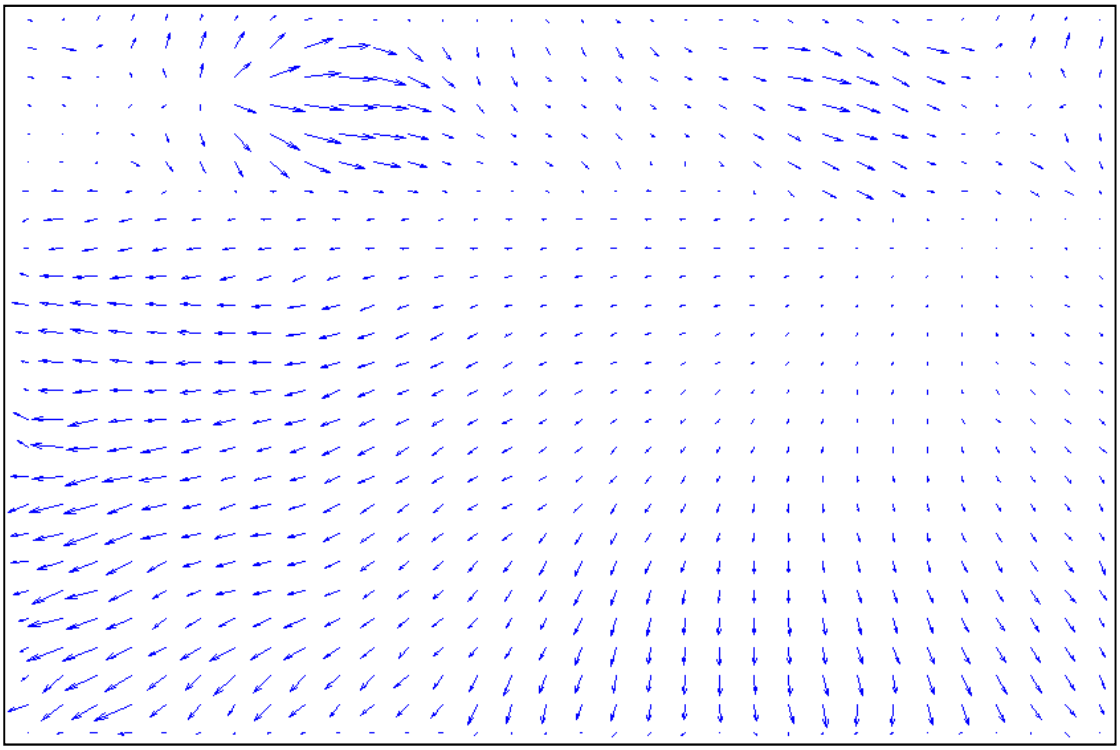


Figure 3.26: Yosemite sequence with clouds: optical flow using the JLK method [7].

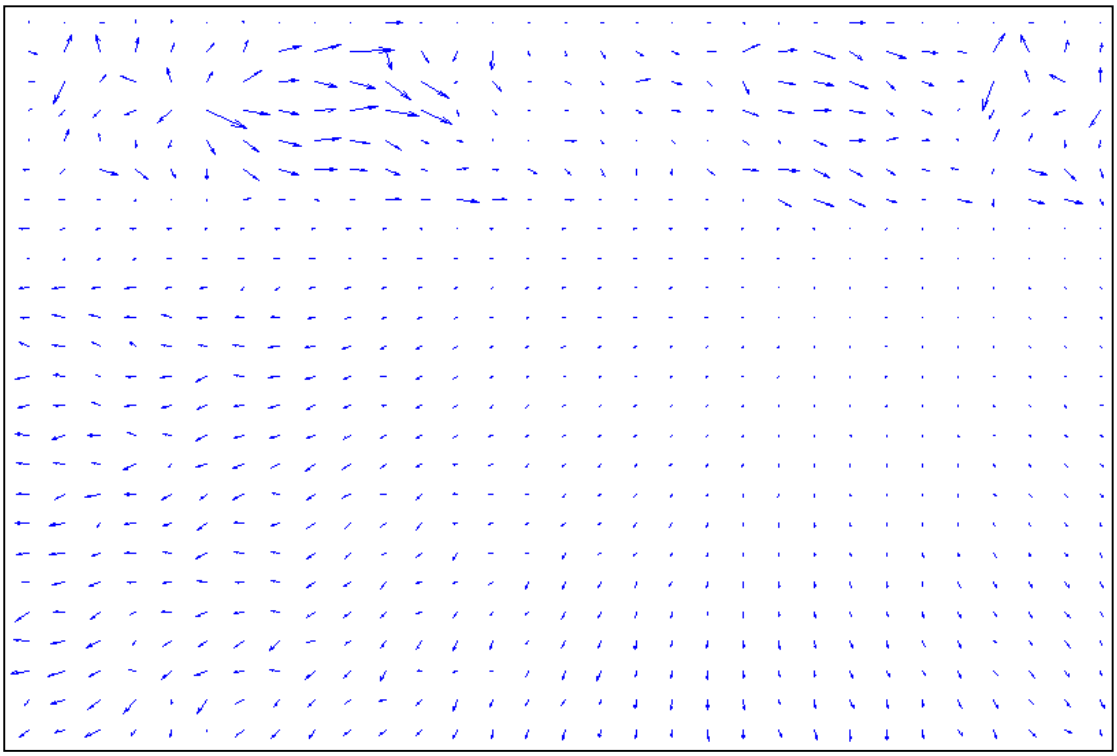


Figure 3.27: Yosemite sequence with clouds: optical flow using the method of Nagel et al. [32].

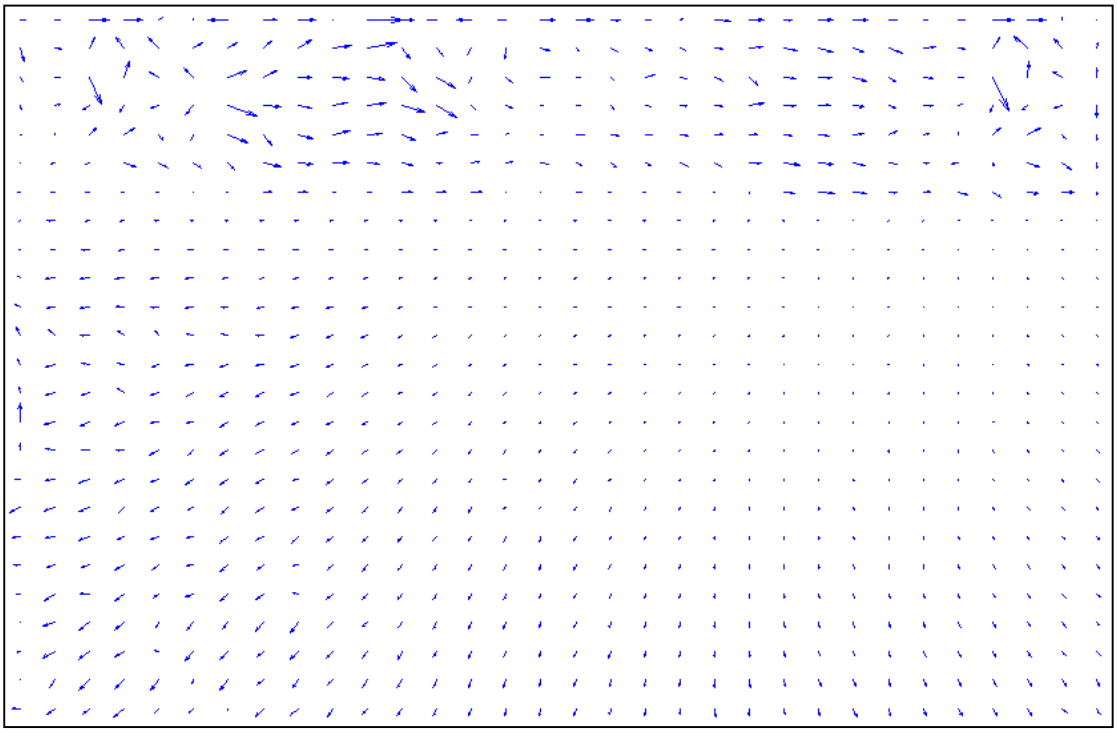


Figure 3.28: Yosemite sequence with clouds: resulting optical flow of the proposed method of section 3.3.

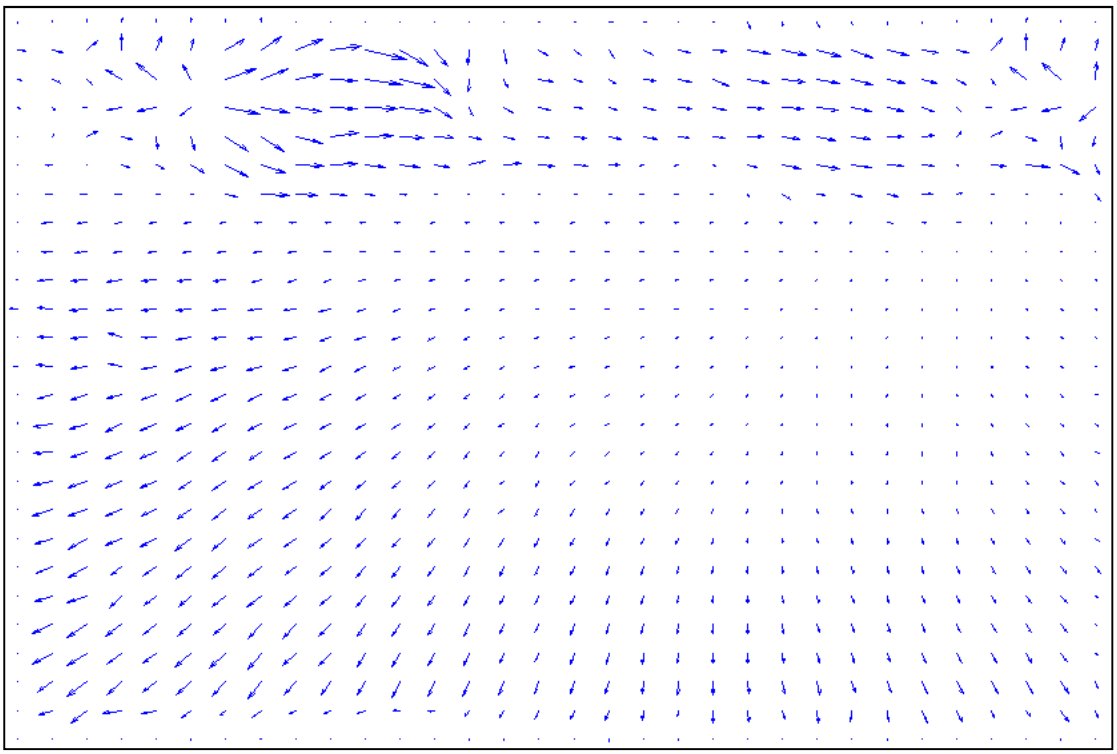


Figure 3.29: Yosemite sequence with clouds: resulting optical flow of the proposed method of section 3.4.

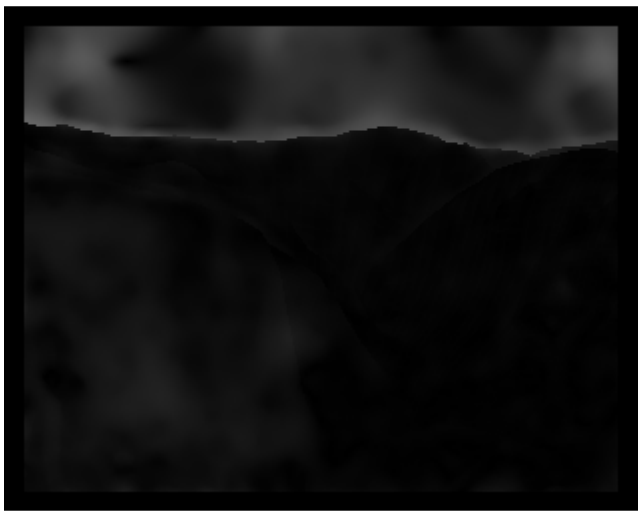
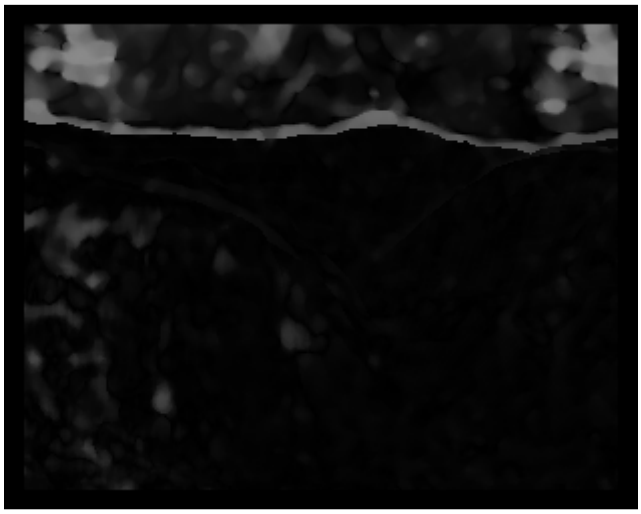


Figure 3.30: Yosemite with clouds' Angular Error (AE) of the JLK method [7].

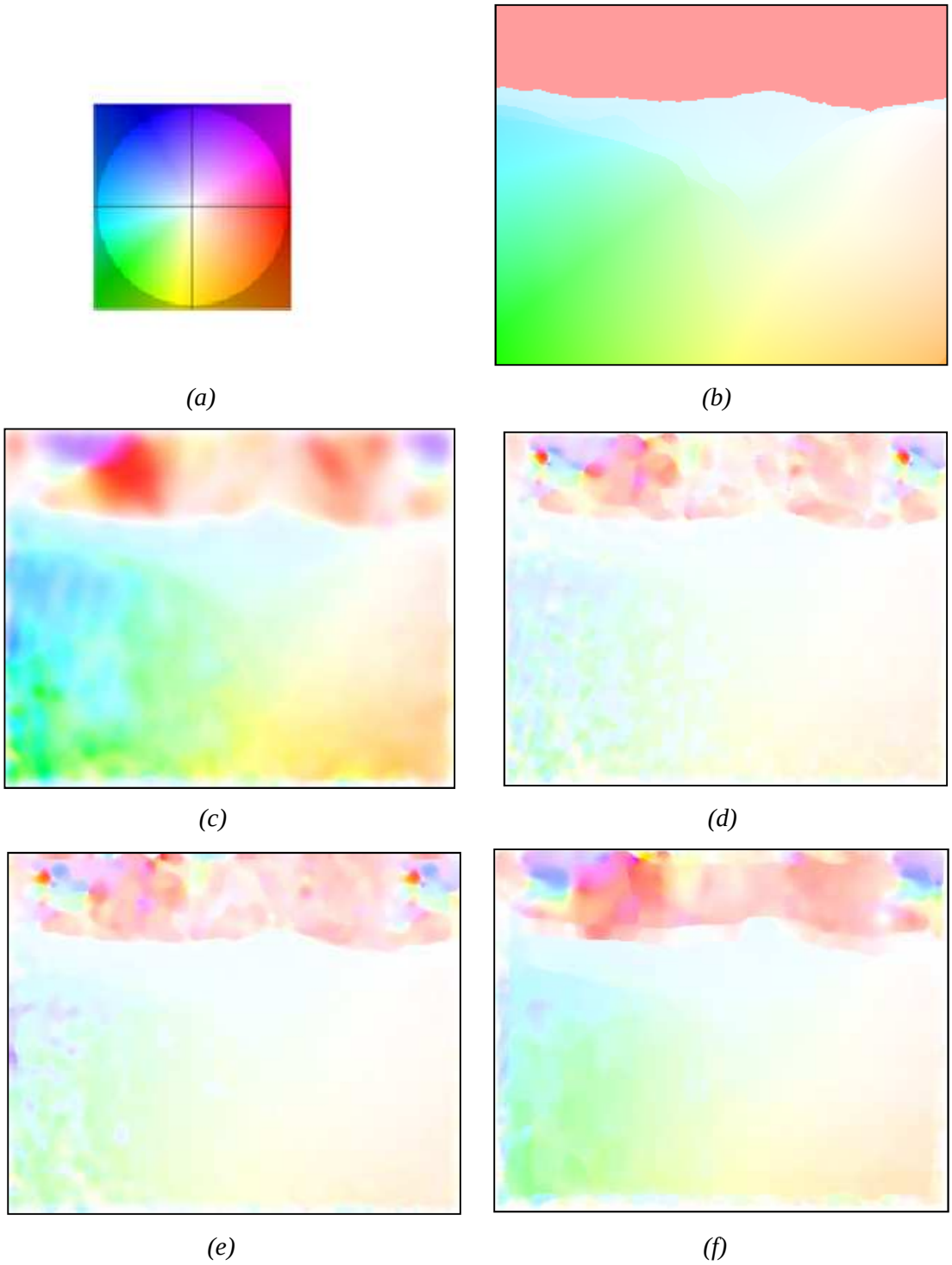
(a)



(b)

(c)

Figure 3.31: Yosemite with clouds' Angular Error (AE) of the compared methods. (a) Method of Nagel et al. [32], *(b) JLK with adaptive smoothing (section 3.3) and (c) guided optical flow using image segmentation (section 3.4).*



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figure 3.32: Yosemite sequence with clouds: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) flow field using the JLK method [7], *(d) flow field using the method of Nagel et al.* [32], *(e) flow field using the method proposed in section 3.3 and (f) flow field using the method proposed in section 3.4.*

Table 3.5: Average error metrics for the Yosemite with Clouds sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	20.75	0.46	0.87
Horn-Schunck [25]	12.57	0.32	0.55
Joint Lucas-Kanade [7]	16.69	0.35	0.63
Nagel et al. [32]	19.78	0.47	0.84
Method of section 3.3	13.46	0.38	0.66
Method of section 3.4	**11.86**	**0.30**	**0.52**

As we can see from the table 3.5 our approaches are better than Nagel's *et al.* method [32], Joint Lucas-Kanade [7], LK [28] and HS [25] for all the error metrics. In order to obtain those results, we used 40 super-pixels and a 21x21 window, representing the neighborhood. See appendix E for more combinations between the number of the super-pixels and the window size.

3.5.6. Dimetrodon Sequence

The dimetrodon sequence is obtained from the Middlebury database [3]. It contains non-rigid motion and large areas with little (hidden or not) texture.

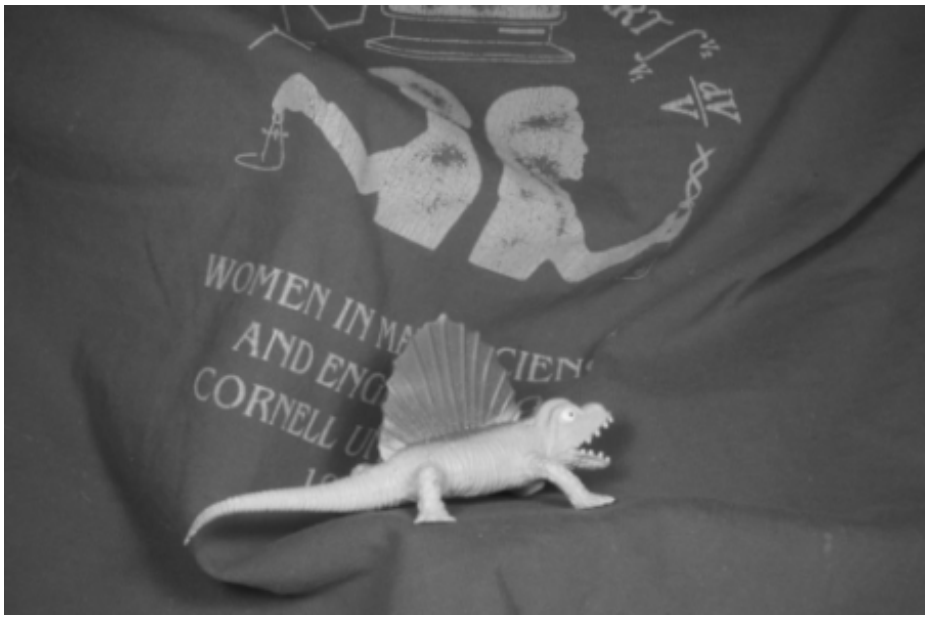


Figure 3.33: Dimetrodon sequence: first frame of the sequence.

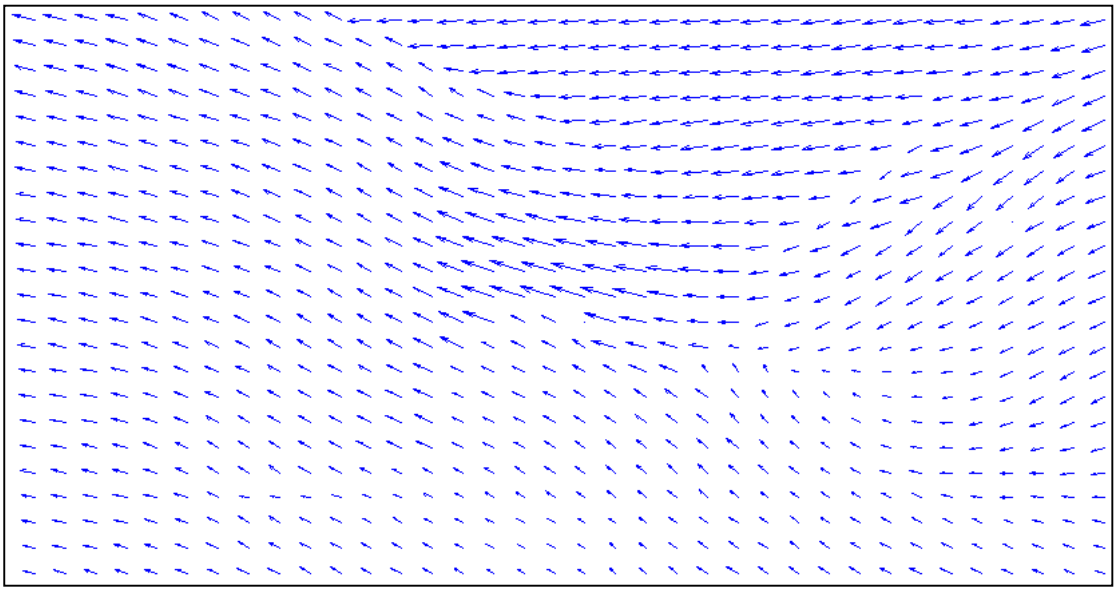


Figure 3.34: Dimetrodon sequence: ground truth optical flow.

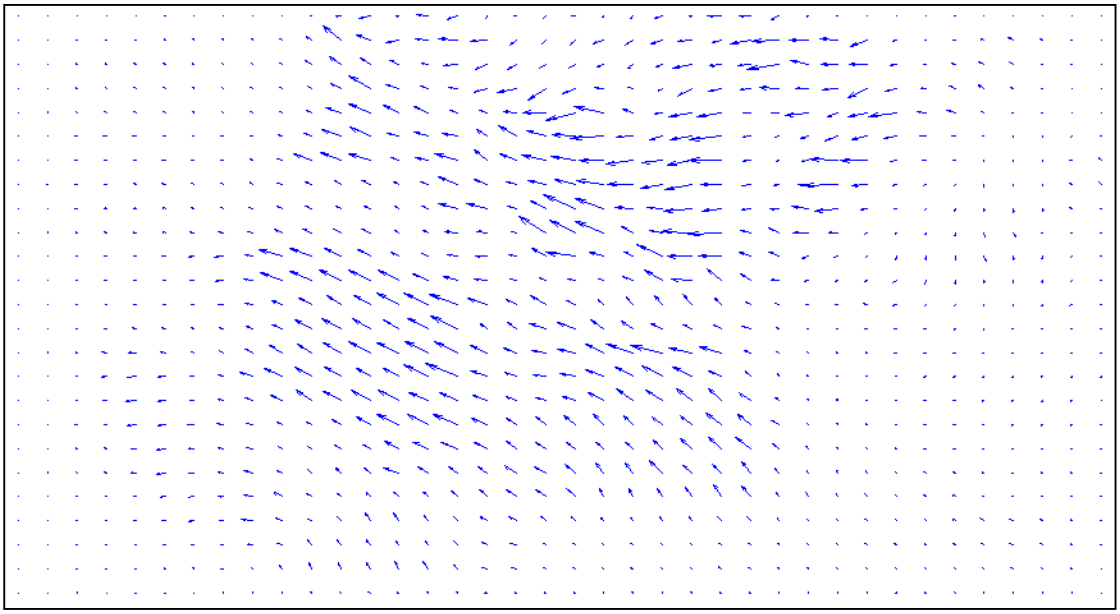


Figure 3.35: Dimetrodon sequence: optical flow using the JLK method [7].

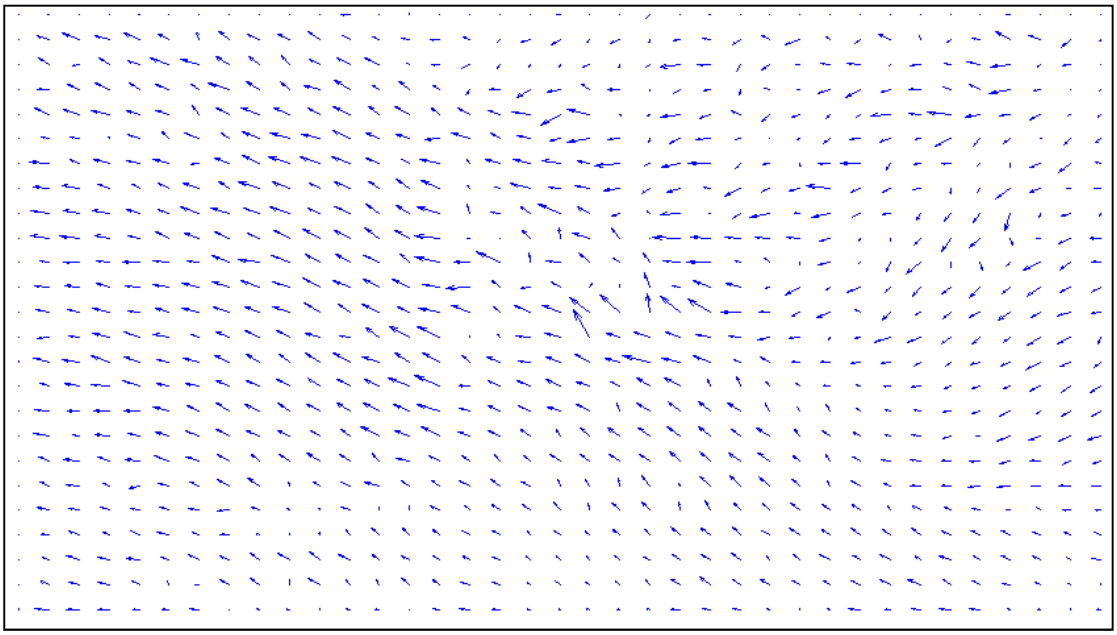


Figure 3.36: Dimetrodon sequence: optical flow using the method of Nagel et al. [32].

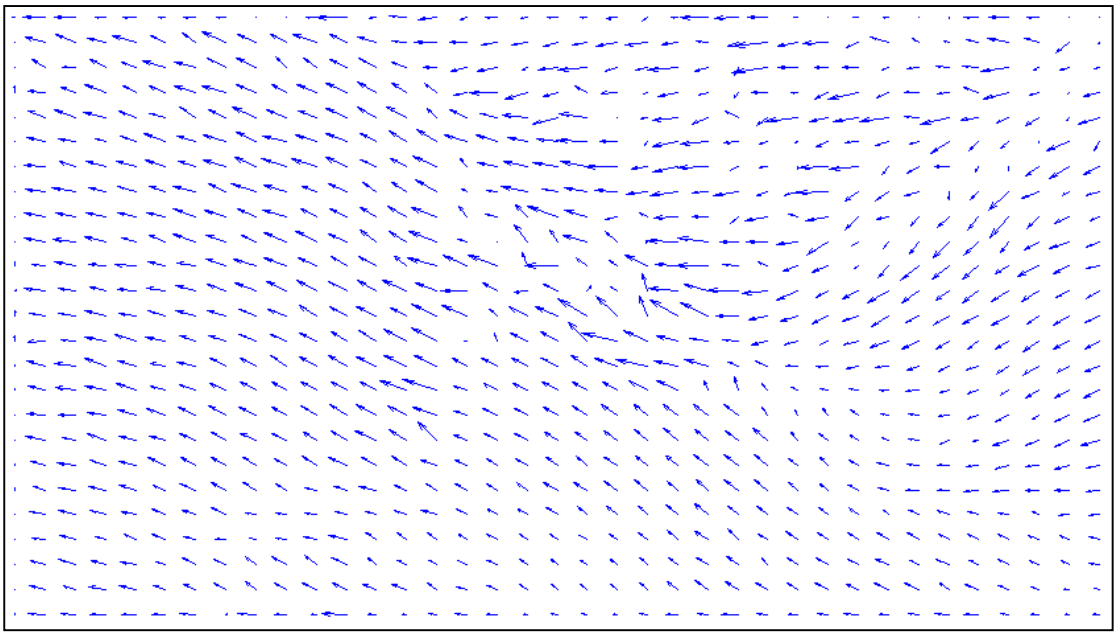


Figure 3.37: Dimetrodon sequence: resulting optical flow of the proposed method of section 3.3.

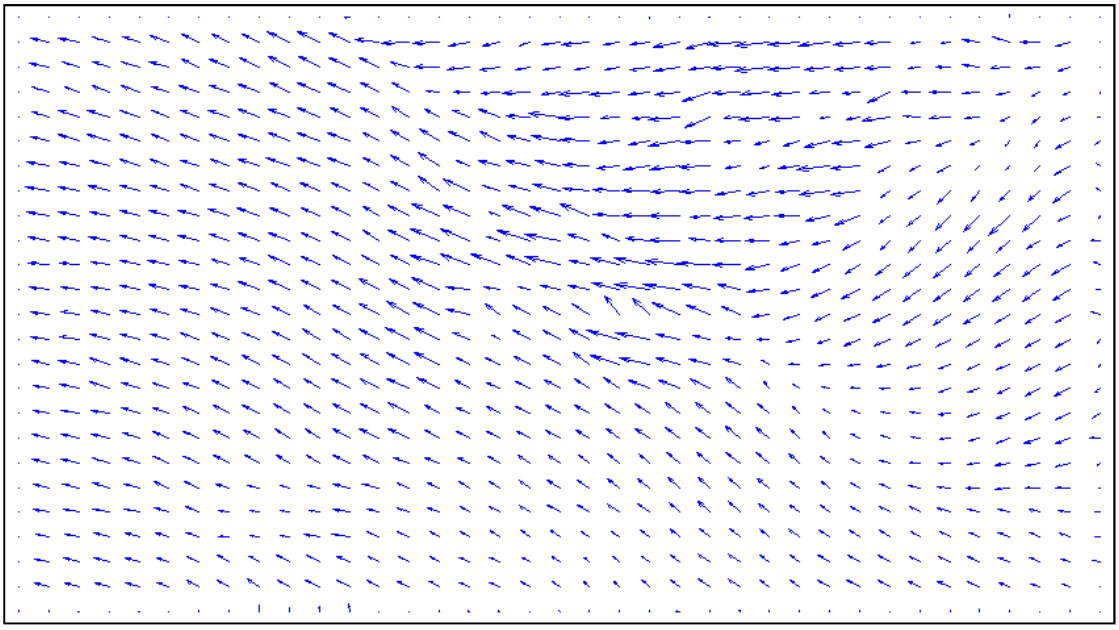


Figure 3.38: Dimetrodon sequence: resulting optical flow of the proposed method of section 3.4.

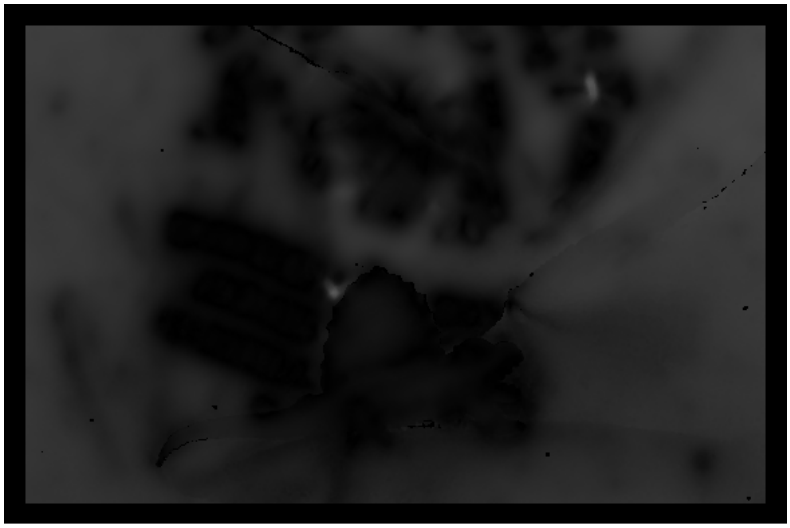
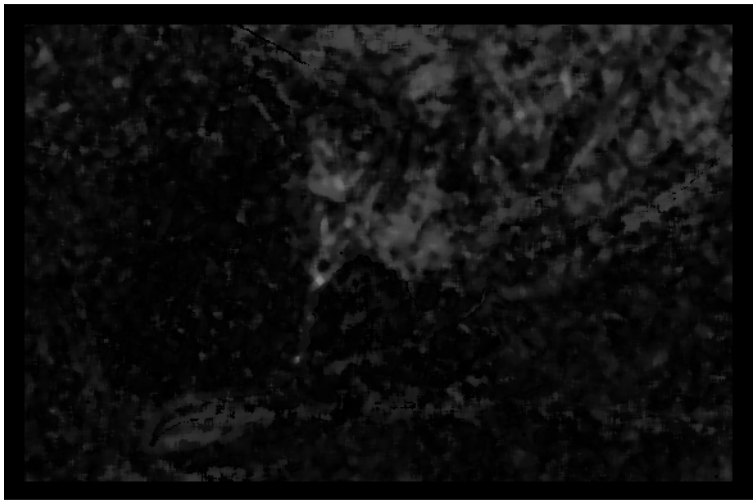
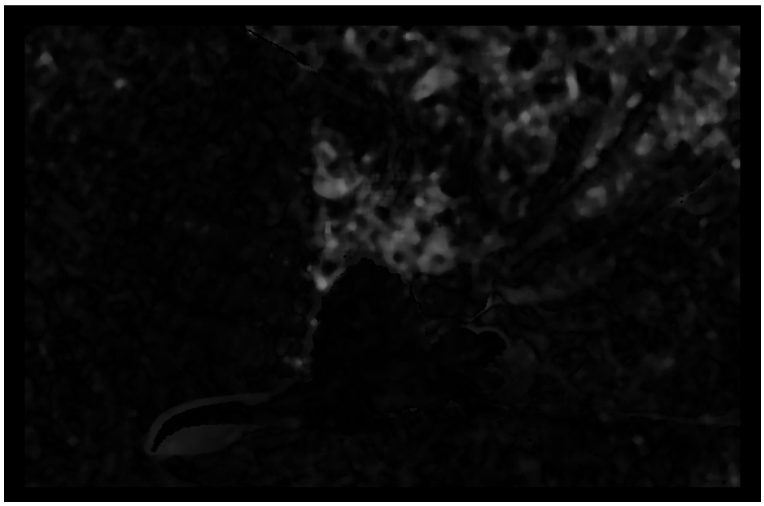


Figure 3.39: Dimetrodon's Angular Error (AE) of the JLK method [7].



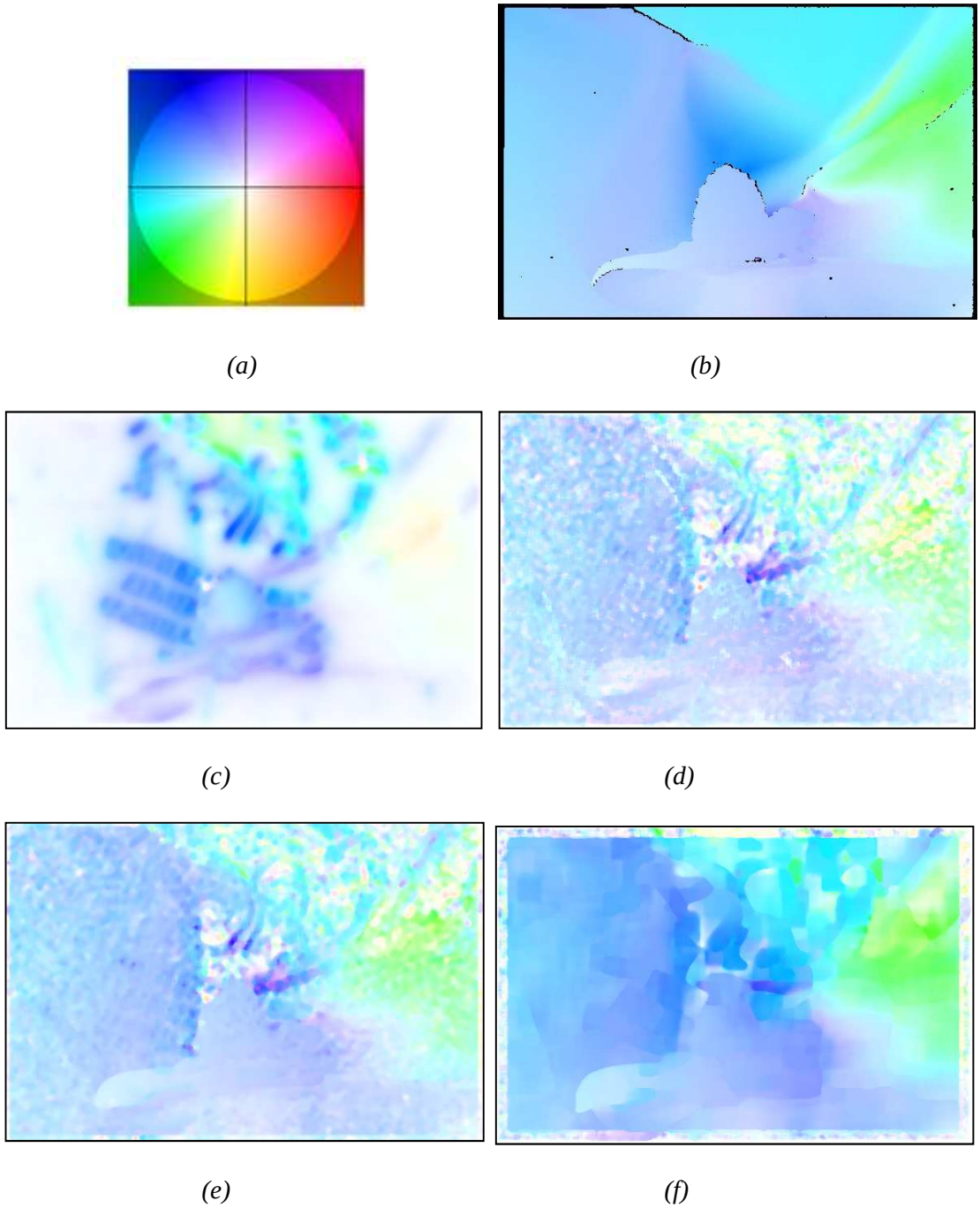
(a)



(b)

(c)

Figure 3.40: Dimetrodon's Angular Error (AE) of the compared methods. (a) Method of Nagel et al. [32], *(b) JLK with adaptive smoothing (section 3.3) and (c) guided optical flow using image segmentation (section 3.4).*



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figure 3.41: Dimetrodon sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) flow field using the JLK method [7], *(d) flow field using the method of Nagel et al.* [32], *(e) flow field using the method proposed in section 3.3 and (f) flow field using the method proposed in section 3.4.*

Table 3.6: Average error metrics for the Dimetrodon sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	27.52	0.56	1.07
Horn-Schunck [25]	8.51	0.24	0.49
Joint Lucas-Kanade [7]	33.14	0.65	**0.35**
Nagel et al. [32]	17.58	0.38	1.17
Method of section 3.3	10.17	0.24	0.52
Method of section 3.4	**6.24**	**0.18**	0.36

As we can see from table 3.6 our approaches are better than Nagel's *et al.* method [32], Joint Lucas-Kanade [7], HS [25] and LK [28], for all the error metrics. The *EP* = 0.35 for the JLK method is misleading since JLK failed in *AAE* metric. In order to obtain those results, we used 40 super-pixels and a 29 x 29 window, representing the neighborhood. See appendix E for more combinations between the number of the super-pixels and the window size.

3.5.7. Rubberwhale Sequence

The rubberwhale sequence is obtained from the Middlebury database [3].

Figure 3.42: Rubberwhale sequence: first frame of the sequence.

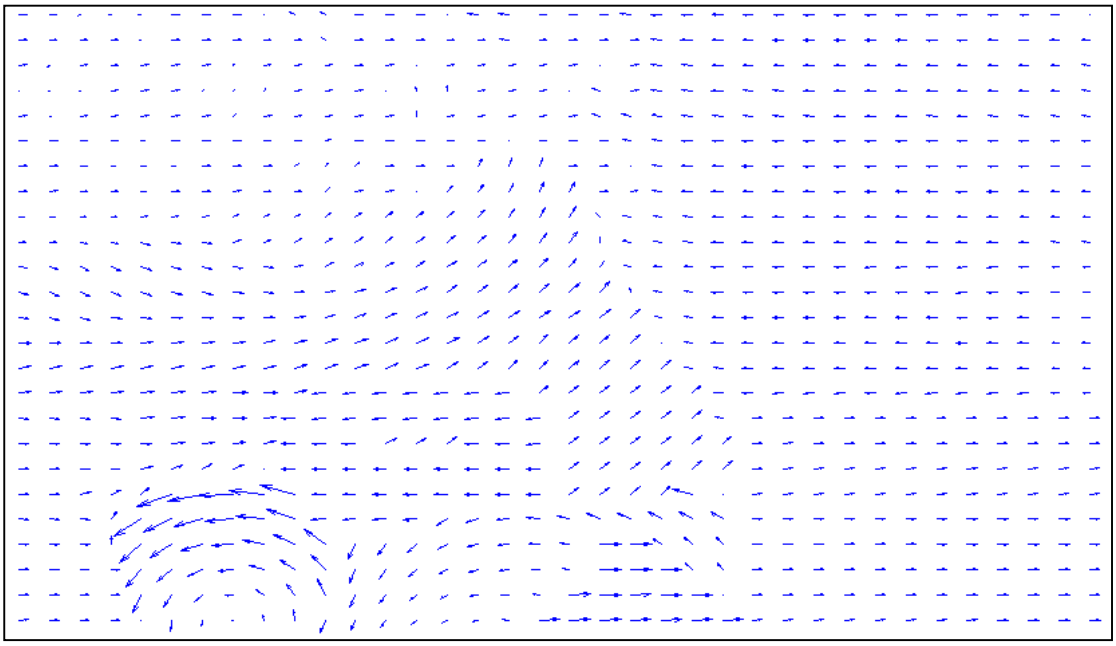


Figure 3.43: Rubberwhale sequence: ground truth optical flow.

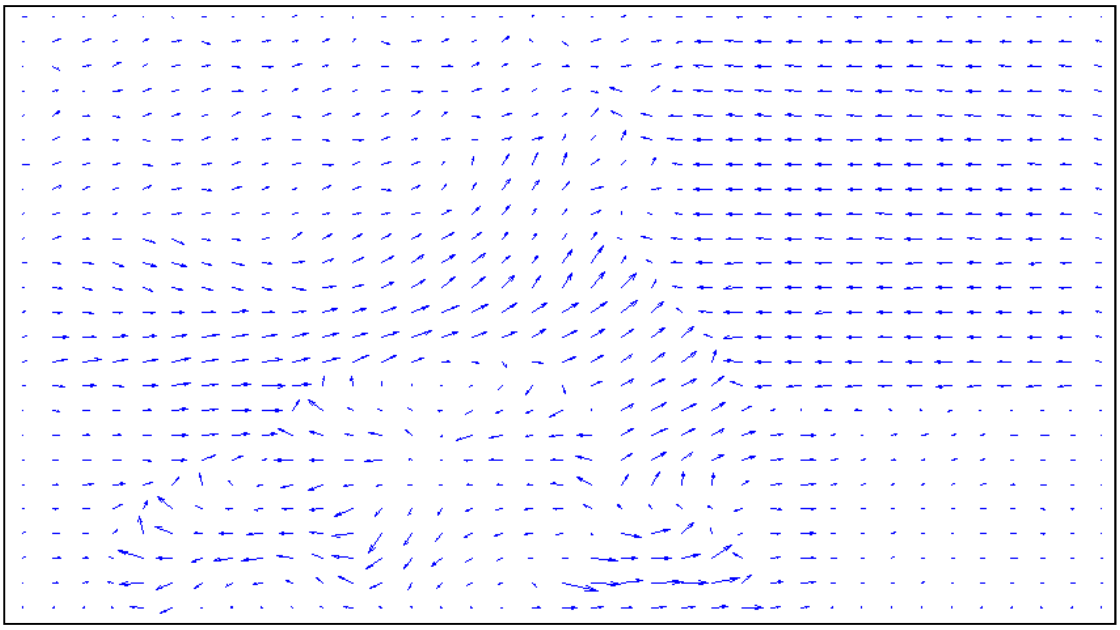


Figure 3.44: Rubberwhale sequence: optical flow using the JLK method [7].

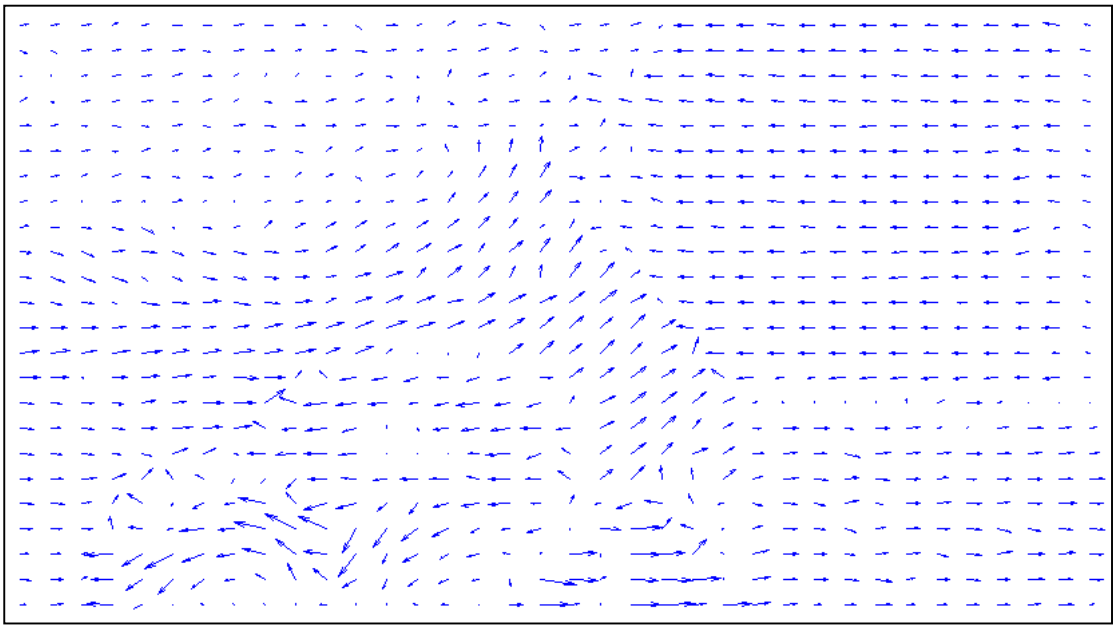


Figure 3.45: Rubberwhale sequence: optical flow using the method of Nagel et al. [32].

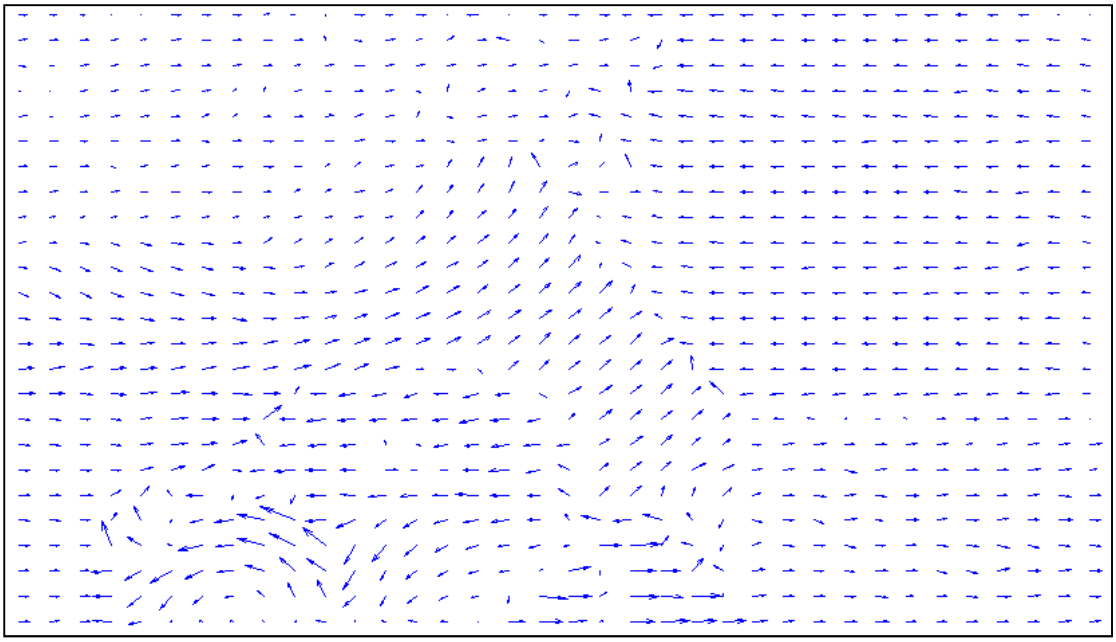


Figure 3.46: Rubberwhale sequence: resulting optical flow of the proposed method of section 3.3.

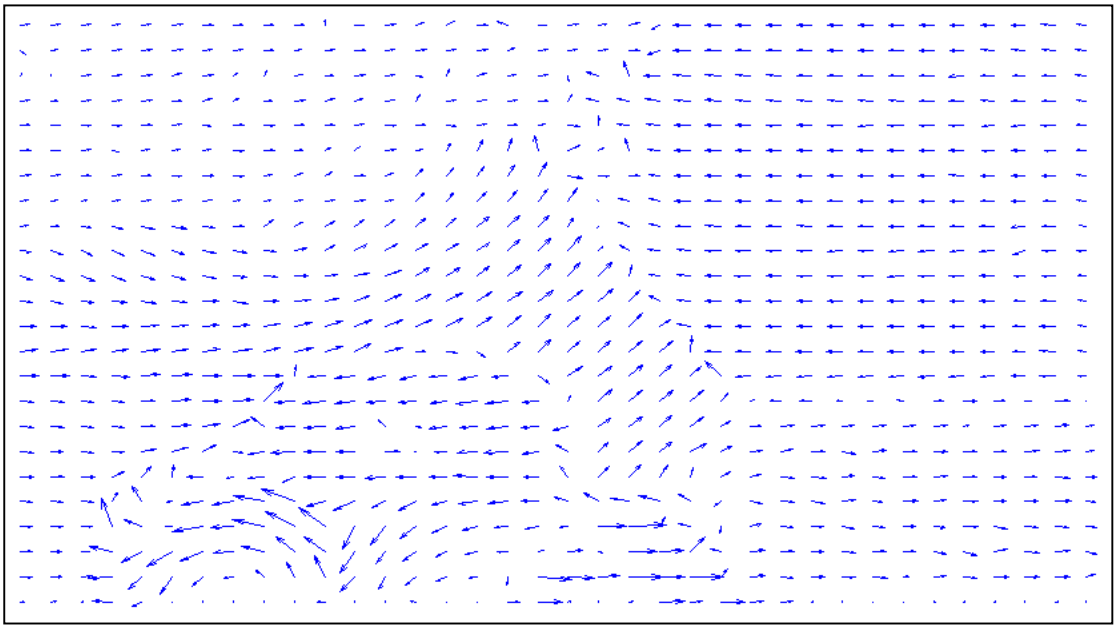


Figure 3.47: Rubberwhale sequence: resulting optical flow of the proposed method of section 3.4.

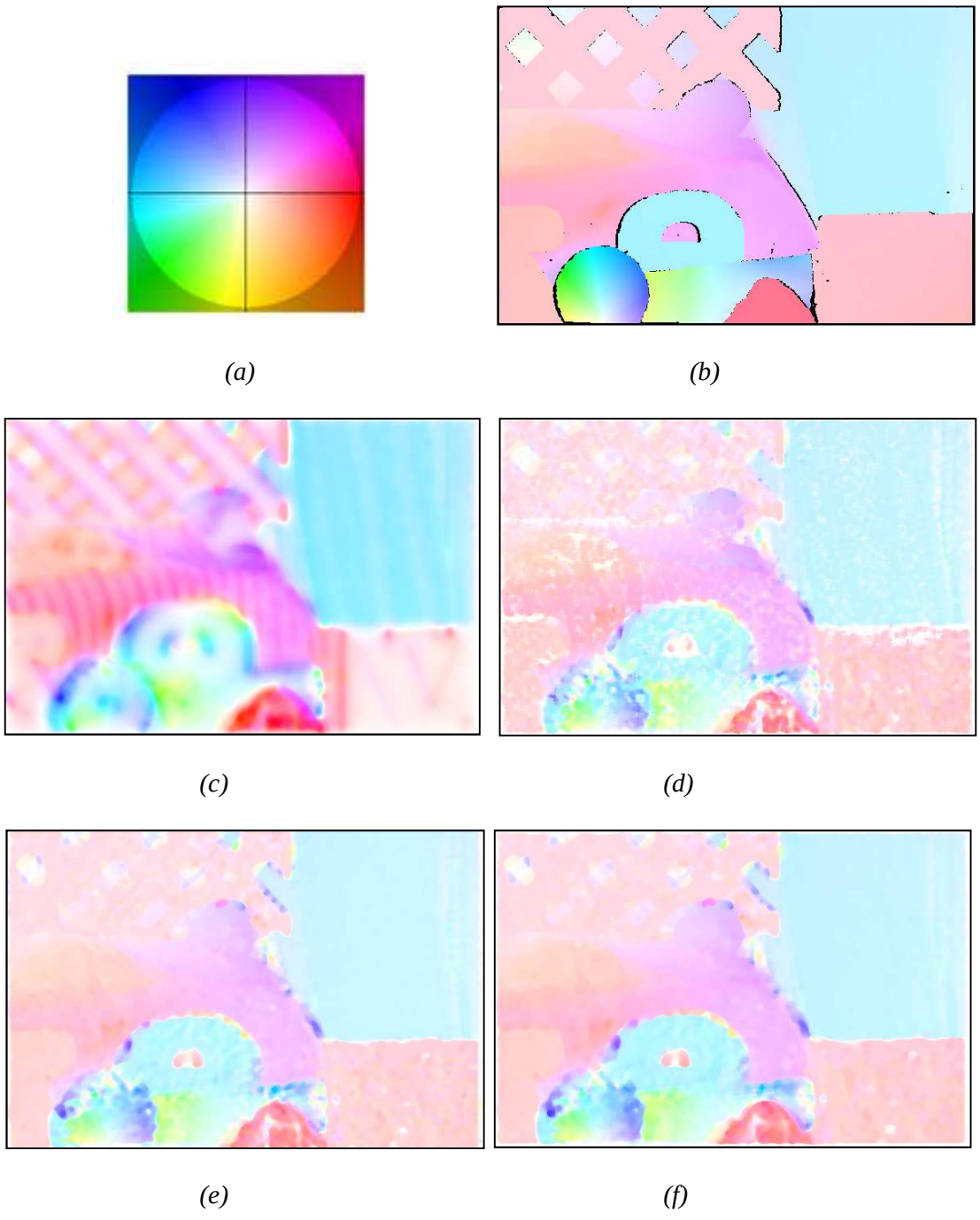
Figure 3.48: Rubberwhale's Angular Error (AE) of the JLK method [7].

(a)

(b)

(c)

Figure 3.49: Rubberwhale's Angular Error (AE) of the compared methods. (a) Method of Nagel et al. [32], *(b) JLK with adaptive smoothing (section 3.3) and (c) guided optical flow using image segmentation (section 3.4).*



(a) *(b)*

(c) *(d)*

(e) *(f)*

Figure 3.50: Rubberwhale sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) flow field using the JLK method [7], *(d) flow field using the method of Nagel et al.* [32], *(e) flow field using the method proposed in section 3.3 and (f) flow field using the method proposed in section 3.4.*

Table 3.7: Average error metrics for the Rubberwhale sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	9.59	0.22	0.29
Horn-Schunck [25]	8.75	0.22	0.25
Joint Lucas-Kanade [7]	18.44	0.43	0.50
Nagel et al. [32]	11.87	0.29	0.33
Method of section 3.3	8.35	0.21	0.25
Method of section 3.4	**8.17**	**0.21**	**0.24**

As we can see from table 3.7 our approaches are better than Nagel's *et al.* method [32], Joint Lucas-Kanade [7], LK [28] and HS [25], for all the error metrics. In order to obtain those results, we used 100 super-pixels and a 9x9 window, representing the neighborhood. See appendix E for more combinations between the number of the super-pixels and the window size.

3.6. Partial Conclusion

In this chapter we studied the methods proposed in [7], [32] but also we proposed two variations of them. As we can see from the previous section, our suggestions manage to achieve significantly better result than the JLK method [7] and the approach of Nagel e*t al.* [32].

Furthermore, we conclude that for the same window size, as the number of the super-pixels increases, we obtain worse results. Additionally, it is understood that the window size has a greater role than the number of the super-pixels, which was expected, since super-pixels have an effect only on the pixel belonging to motion boundaries or to edges in the image, who are the minority of the image canvas.

CHAPTER 4. VARIATIONAL BAYESIAN OPTICAL FLOW

4.1. Introduction

This work resulted from the combination of the method proposed in [14] with the well-known Horn–Schunck (*HS*) method [25]. More specifically, the main difference between our approach and the method of *HS* is that we don't employ a deterministic parameter to control the strength of the smoothness constraint. More specifically, we impose stochastic parameters, one for each pixel, similar in spirit with [14], which are updated at each iteration. Moreover, we impose Gaussian noise

statistics in order to capture the missing information due to the linearization by Taylor expansion series.

4.2. A Prior for the Motion Vectors

We assume that $\mathbf{u}_x(i)$, $\mathbf{u}_y(i)$ for $i = 1, \ldots, N$ are i.i.d zero mean Student's-t distributed (see Appendix B for details), with parameters λ_x, v_x and λ_y, v_y, respectively:

$$\begin{cases} \mathbf{u}_x(i) \sim St(0,\ \lambda_x, v_x) \\ \mathbf{u}_y(i) \sim St(0,\ \lambda_y, v_y) \end{cases}, \tag{4.1}$$

The Student's t-distribution implies a two-level generative process [8]. More specifically, $\boldsymbol{\alpha}_x(i)$ and $\boldsymbol{\alpha}_y(i)$ are first drawn from a Gamma distribution

$$\begin{cases} \boldsymbol{\alpha}_x(i) \sim Gamma\left(\frac{v_x}{2}, \frac{v_x}{2}\right) \\ \boldsymbol{\alpha}_y(i) \sim Gamma\left(\frac{v_y}{2}, \frac{v_y}{2}\right) \end{cases}, \tag{4.2}$$

At this step, the probability density function (4.1) may be written as an integral

$$\begin{aligned} p(\mathbf{u}_k(i)) = St(0,\ \lambda_k, v_k) &= \int_0^\infty p(\mathbf{u}_k(i), \boldsymbol{\alpha}_k(i))\ \mathrm{d}\boldsymbol{\alpha}_k(i) \\ &= \int_0^\infty p(\mathbf{u}_k(i) \mid \boldsymbol{\alpha}_k(i))\, \mathrm{p}(\boldsymbol{\alpha}_k(i))\ \mathrm{d}\boldsymbol{\alpha}_k(i), \end{aligned} \tag{4.3}$$

As v_k goes to infinity, the pdf of $\boldsymbol{\alpha}_k(i)$'s has its mass concentrated around its mean. This in turn reduces the Student's-t to a normal distribution, because all $\mathbf{u}_k(i)$, $k \in \{x, y\}$ are drawn from the same normal distribution with precision λ_k, since $\boldsymbol{\alpha}_k(i) = 1$. When $v_k \to 0$ the prior becomes uninformative. In general, for small values of v_k the probability mass of the Student's-t pdf is more "heavy tailed".

Then, $\mathbf{u}_x(i)$, $\mathbf{u}_y(i)$ are generated from two independent zero-mean normal distributions with precision $\lambda_x\ \mathbf{Q}^{\mathrm{T}}\mathbf{A}_x\mathbf{Q}$, $\lambda_y\ \mathbf{Q}^{\mathrm{T}}\mathbf{A}_y\mathbf{Q}$, respectively, where $\mathbf{Q}$ is the Laplacian operator and $\mathbf{A}_x = diag\{\ \boldsymbol{\alpha}_x(i)\ \}$, $\mathbf{A}_y = diag\{\ \boldsymbol{\alpha}_y(i)\ \}$, according to

$$\begin{cases} p(\mathbf{u}_x \mid \mathbf{A}_x) = N\left(0, \left(\lambda_x \mathbf{Q}^T \mathbf{A}_x \mathbf{Q}\right)^{-1}\right) \\ p(\mathbf{u}_y \mid \mathbf{A}_y) = N\left(0, \left(\lambda_y \mathbf{Q}^T \mathbf{A}_y \mathbf{Q}\right)^{-1}\right) \end{cases}, \tag{4.4}$$

Equation (4.4) may also be written more compactly as:

$$p(\mathbf{u} \mid \tilde{\mathbf{A}}) = N\left(0, \left(\lambda\, \tilde{\mathbf{Q}}^T \tilde{\mathbf{A}}\ \tilde{\mathbf{Q}}\right)^{-1}\right),$$

where $\lambda = \begin{bmatrix} \lambda_x \\ \lambda_y \end{bmatrix}$, $\tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_x & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_y \end{bmatrix}$, $\mathbf{0}$ is a zero matrix of size (N x N) and similarly $\tilde{\mathbf{Q}} = \begin{bmatrix} \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} \end{bmatrix}$.

Combining both components of $\mathbf{u}$ in one equation we obtain the density for the motion vectors

$$p(\mathbf{u} \mid \tilde{\mathbf{A}}) = \prod_{k \in \{x,y\}} \prod_{i=1}^{N} \left(\lambda_k \mathbf{A}_k\right)^{1/2} \exp\left\{-\frac{1}{2} \lambda_k \mathbf{u}_k^T \mathbf{Q}^T \mathbf{A}_k \mathbf{Q} \mathbf{u}_k\right\}, \tag{4.5}$$

Following (4.3), the marginal distribution $p(\mathbf{u})$ yearns for a closed form. However, this prior is analytically intractable because one cannot find in closed form its *normalization constant*. This problem stems from the fact that it is not possible to find the eigenvalues of the matrix $\mathbf{Q}_k^{\mathrm{T}}\mathbf{A}_k\mathbf{Q}$ since it is very large and it does not have a structure that is amenable to efficient eigenvalue computation. Consequently, we have to import a proper model inference scenario, which in our case is described in section 4.4.

4.3. A Probabilistic Model for Optical Flow

Let I be the first image frame (vectorized intensity values) which is commonly named as the target image, J the second image frame, which will be the source image, $\mathbf{x}$ the vector containing the 2D coordinates of the pixels in a frame, and $\mathbf{u}$ the optical flow vectors of the pixels. For convenience, but without loss of generality, we use 1D notation.

As many methods usually do, based on the brightness constancy constraint, our aim is to minimize the intensity error, $J(\mathbf{x}) - I(\mathbf{x} + \mathbf{u})$, with respect to $\mathbf{u}$. By developing the Taylor series expansion of $I(\mathbf{x} + \mathbf{u})$ around point $\mathbf{x}$ and keeping only the linear part, we come up with the following linear system:

$$\left[\mathbf{G}_x \quad \mathbf{G}_y\right]\begin{bmatrix}\mathbf{u}_x \\ \mathbf{u}_y\end{bmatrix} - \mathbf{d} = 0\,,$$

which can be written also as

$$\mathbf{d} = \mathbf{G}\,\mathbf{u} + \mathbf{w}\,, \tag{4.6}$$

where $\mathbf{d}$ is the initial intensity difference between the two frames $\mathbf{d} = J(\mathbf{x}) - I(\mathbf{x})$ in vectorized form (e.g. lexicographic ordering), $\mathbf{G}$ contains the spatial gradients $\mathbf{G} = \nabla J = [\mathbf{G}_x \quad \mathbf{G}_y]_{N\,x\,2N}$ $\mathbf{G}_x = diag\left\{\frac{\partial J}{\partial x_i}\right\}_{i=1,\ldots,N}$, $\mathbf{G}_y = diag\left\{\frac{\partial J}{\partial y_i}\right\}_{i=1,\ldots,N}$, N being the number of pixels, $\mathbf{u} = [\mathbf{u}_x\,, \mathbf{u}_y]^{\mathrm{T}}$, and $\mathbf{w}$ is additive white noise modeling the rest of the Taylor expansion terms. We also assume Gaussian statistics for the noise:

$$\mathbf{w} \sim \mathcal{N}\left(\mathbf{0}, \left(\lambda_{noise}\,\mathbf{B}\right)^{-1}\right), \tag{4.7}$$

where $\lambda_{\text{noise}}\mathbf{B}$ is the noise precision matrix, $\mathbf{0}$ is an N x 1 vector of zeros and $\mathbf{B} = diag\{\mathbf{b}(1),\ldots,\mathbf{b}(N)\}$. To make the model more flexible, we also consider that:

$$\mathbf{b}(i) \sim Gamma\left(\frac{\mu}{2}, \frac{\mu}{2}\right)\,, \tag{4.8}$$

Following the optical flow model in (4.6),

$$p(\mathbf{d} \mid \mathbf{u}) = \mathcal{N}\left(\mathbf{Gu}, \left(\lambda_{noise}\mathbf{B}\right)^{-1}\right), \tag{4.9}$$

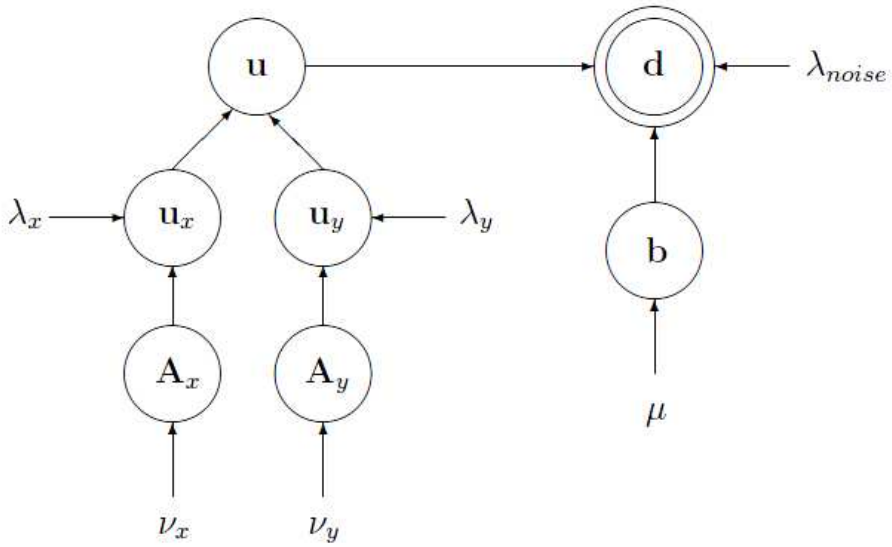


Figure 4.1: The graphical model of the method.

As it may be observed the graphical model of figure 4.1, **d** is the vector containing the observations (temporal differences), **u**, $\mathbf{A}_x$, $\mathbf{A}_y$, **b**, are the hidden variables of the model and λ_x, λ_y, λ_{noise}, ν_x, ν_y and μ are the model's parameters.

4.4. Model Inference

Working in the Bayesian framework, the complete data likelihood is

$$p\left(\mathbf{d}, \mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}; \theta\right) = p\left(\mathbf{d} \mid \mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}; \theta\right) p\left(\mathbf{u} \mid \tilde{\mathbf{A}}, \mathbf{b}; \theta\right) p\left(\tilde{\mathbf{A}}; \theta\right) p\left(\mathbf{b}; \theta\right), \tag{4.10}$$

where $\theta = [\lambda_{noise}, \lambda_x, \lambda_y, \mu, \nu_x, \nu_y]$ contains the parameters of the model.

Estimation of the model parameters could be obtained through maximization of the marginal distribution of the observations $p(\mathbf{d}; \theta)$:

$$\hat{\theta} = \arg\max_{\theta} \iiint p\left(\mathbf{d}, \mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}; \theta\right) \, \mathrm{d}\mathbf{u} \; \mathrm{d}\tilde{\mathbf{A}} \; \mathrm{d}\mathbf{b} \tag{4.11}$$

However, in the present case, this marginalization is not possible, since the posterior of the latent variables given the observations $p(\mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b} \mid \mathbf{d})$ is not known explicitly and inference via the Expectation-Maximization (EM) algorithm is not possible [5].

For this reason, we have to resort to the variational methodology [14], [8], [26] and [5]. According to this methodology, we have to maximize the following lower bound

$$L\left(\mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}; \theta\right) = \int_{\mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}} q\left(\mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}\right) \log \frac{q\left(\mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}\right)}{p\left(\mathbf{d}, \mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}; \theta\right)} \; . \tag{4.12}$$

This involves finding approximations of the posterior distribution of the hidden variables, denoted by $q(\mathbf{u})$, $q(\tilde{\mathbf{A}})$, $q(\mathbf{b})$ because there is no analytical form of the auxiliary function q for which the bound in (4.12) becomes equality. In the variational methodology, however, we employ the *Mean Field* approximation (see Appendix D for details):

$$q\left(\mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}\right) = q(\mathbf{u}) \, q\left(\tilde{\mathbf{A}}\right) q(\mathbf{b}) \tag{4.13}$$

and (4.12) becomes

$$L\left(\mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}; \theta\right) = \int_{\mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}} q(\mathbf{u}) \, q\left(\tilde{\mathbf{A}}\right) q(\mathbf{b}) \log \frac{q(\mathbf{u}) \, q\left(\tilde{\mathbf{A}}\right) q(\mathbf{b})}{p\left(\mathbf{d}, \mathbf{u}, \tilde{\mathbf{A}}, \mathbf{b}; \theta\right)} \; . \tag{4.14}$$

In our case, in the VE-step of the variational algorithm, optimization of the functional $L(q(\mathbf{x}), \theta)$ is performed with respect to the auxiliary functions. In the present case following the variational inference framework, the distributions $q(\mathbf{u}_k)$, $k \in \{x, y\}$, are normal:

$$q(\mathbf{u}) = N\left(\begin{bmatrix} \mathbf{m}_x \\ \mathbf{m}_y \end{bmatrix}, \begin{bmatrix} \mathbf{R}_x & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_y \end{bmatrix}\right) \Rightarrow \begin{cases} q(\mathbf{u}_x) = N(\mathbf{m}_x, \mathbf{R}_x) \\ q(\mathbf{u}_y) = N(\mathbf{m}_y, \mathbf{R}_y) \end{cases}$$

Therefore, this bound is actually a function of the parameters $\mathbf{R}_k$ and $\mathbf{m}_k$ and a functional w.r.t. the auxiliary functions $q(\mathbf{A_k})$, $q(\mathbf{B})$. Using (4.13), the variational bound in our problem becomes

$$L\left(q(\mathbf{u}_x), q(\mathbf{u}_y), q(\mathbf{A}_x), q(\mathbf{A}_y), q(\mathbf{B}), \theta_1, \theta_2\right) =$$
$$\int \prod_{k\in\{x,y\}} q(\mathbf{u}_k;\theta_1) q(\mathbf{A}_k) q(\mathbf{B}) \log p(\mathbf{d},\mathbf{u},\mathbf{A},\mathbf{B};\theta_2)\ d\mathbf{u}\ d\mathbf{A}\ d\mathbf{B}$$
$$-\int q(\mathbf{u}_k;\theta_1) q(\mathbf{A}_k) q(\mathbf{B}) \log \prod_{k\in\{x,y\}} p(\mathbf{u}_k;\theta_1) q(\mathbf{A}_k) q(\mathbf{B})\ d\mathbf{u}\ d\mathbf{A}\ d\mathbf{B}\ , \quad (4.15)$$

where $\theta_1 = [\mathbf{R}_x, \mathbf{R}_y, \mathbf{m}_x, \mathbf{m}_y]$ and $\theta_2 = [\mathbf{A}_x, \mathbf{A}_y, \mathbf{B}, \lambda_x, \lambda_y\ v_x, v_y]$. Thus, in the VE-step of our algorithm the bound must be optimized with respect to $\mathbf{R}_k$, $\mathbf{m}_k$, $q(\mathbf{A}_k)$ and $q(\mathbf{B})$.

Taking the derivatives of (4.15) w.r.t to $\mathbf{m}_k$, $\mathbf{R}_k$, $q(\mathbf{A}_k)$, $q(\mathbf{B})$ and setting them to zero we obtain

$$\begin{cases} \mathbf{m}_x^{(t+1)} = \lambda_{noise}^{(t)} \mathbf{R}_x^{(t)} \hat{\mathbf{B}}^{(t)} \mathbf{G}_x \left(\mathbf{d} - \mathbf{G}_y \mathbf{u}_y^{(t)}\right) \\ \mathbf{m}_y^{(t+1)} = \lambda_{noise}^{(t)} \mathbf{R}_y^{(t)} \hat{\mathbf{B}}^{(t)} \mathbf{G}_y \left(\mathbf{d} - \mathbf{G}_x \mathbf{u}_x^{(t)}\right) \end{cases}, \quad (4.16)$$

$$\begin{cases} \mathbf{R}_x^{(t+1)} = \left(\lambda_{noise}^{(t)} \mathbf{G}_x^{\mathrm{T}} \hat{\mathbf{B}}^{(t)} \mathbf{G}_x + \lambda_x^{(t)} \mathbf{Q}^{\mathrm{T}} \hat{\mathbf{A}}_x^{(t)} \mathbf{Q}\right)^{-1} \\ \mathbf{R}_y^{(t+1)} = \left(\lambda_{noise}^{(t)} \mathbf{G}_y^{\mathrm{T}} \hat{\mathbf{B}}^{(t)} \mathbf{G}_y + \lambda_y^{(t)} \mathbf{Q}^{\mathrm{T}} \hat{\mathbf{A}}_y^{(t)} \mathbf{Q}\right)^{-1} \end{cases}, \quad (4.17)$$

After some manipulation, we obtain the update equations for the model parameters which maximize over $q(\mathbf{A}_k)$, $q(\mathbf{b})$. The form of all q approximating-to-the-posterior functions will remain the same as the corresponding prior (due to the conjugate priors we employ) namely $q(\mathbf{A}_k)$, $q(\mathbf{b})$ which approximate $p(\mathbf{A}_k \mid \mathbf{u}_k, \lambda_k, \mathbf{C}_k; v_k)$, $p(\mathbf{b} \mid \mathbf{u}, \lambda_{noise}, \mathbf{F}; \mu)$ will follow Gamma distributions.

$$\begin{cases} q^{(t+1)}\left(\boldsymbol{\alpha}_x(i)\right) = Gamma\left(\frac{v_x^{(t)}}{2} + \frac{1}{2}, \frac{v_x^{(t)}}{2} + \frac{1}{2}\lambda_x^{(t)}\left(\left(\mathbf{Q}\mathbf{u}_x^{(t)}\right)_i^2 + \mathbf{C}_x^{(t)}(i,i)\right)\right), \forall i, \\ q^{(t+1)}\left(\boldsymbol{\alpha}_y(i)\right) = Gamma\left(\frac{v_y^{(t)}}{2} + \frac{1}{2}, \frac{v_y^{(t)}}{2} + \frac{1}{2}\lambda_y^{(t)}\left(\left(\mathbf{Q}\mathbf{u}_y^{(t)}\right)_i^2 + \mathbf{C}_y^{(t)}(i,i)\right)\right), \forall i, \end{cases} \quad (4.18)$$

$$q^{(t+1)}(\mathbf{b}(i)) = Gamma\left(\frac{\mu^{(t)}}{2}+\frac{1}{2}, \frac{\mu^{(t)}}{2}+\frac{1}{2}\lambda_{noise}^{(t)}\left(\left(\mathbf{G}\mathbf{u}^{(t)}-\mathbf{d}\right)_i^2+\mathbf{F}^{(t)}(i,i)\right)\right), \forall i, \qquad (4.19)$$

where $\mathbf{C}_k^{(t)} = \mathbf{Q}\mathbf{R}_k^{(t)}\mathbf{Q}^T$, $k\epsilon\{x, y\}$, $\mathbf{F}^{(t)} = \sum_{k\in\{x,y\}}\mathbf{G}_k\mathbf{R}_k^{(t)}\mathbf{G}_k^T$, $\mathbf{d} = J(\mathbf{x}) - I(\mathbf{x})$ (the intensity difference between the two initial frames) . Notice that the final estimates for $\mathbf{u_x}$ and $\mathbf{u_y}$ are $\mathbf{m_x}$ and $\mathbf{m_y}$, in (4.16) respectively.

Observing the size of matrices $\mathbf{R}_x$, $\mathbf{R}_y$ and consequently $\mathbf{C}_x$, $\mathbf{C}_y$, $\mathbf{F}$, we have to use an iterative method in order to calculate them. Hence, we recur to the Lanczos method [14, 33].

As we can see from (4.16) there is a dependency between $\mathbf{u_x}$ and $\mathbf{u_y}$, as it is the case in the standard Horn–Schunck method.

Notice that since each $q^{(t+1)}(\boldsymbol{\alpha}_k(i))$ are Gamma pdfs of the form $q^{(t+1)}\left(\boldsymbol{\alpha}_k^{(t+1)}(i)\right) = Gamma(\alpha,\ \beta)$, their expected values are

$$\begin{cases} \left\langle \boldsymbol{\alpha}_x(i)\right\rangle_{q^{(t+1)}(\boldsymbol{\alpha}_x(i))} = \dfrac{\alpha}{\beta} = \dfrac{v_x^{(t)}+1}{v_x^{(t)}+\lambda_x^{(t)}\left(\left(\mathbf{Q}\mathbf{u}_x^{(t)}\right)_i^2+\mathbf{C}_x^{(t)}(i,i)\right)} \\ \left\langle \boldsymbol{\alpha}_y(i)\right\rangle_{q^{(t+1)}(\boldsymbol{\alpha}_y(i))} = \dfrac{\alpha}{\beta} = \dfrac{v_y^{(t)}+1}{v_y^{(t)}+\lambda_y^{(t)}\left(\left(\mathbf{Q}\mathbf{u}_y^{(t)}\right)_i^2+\mathbf{C}_y^{(t)}(i,i)\right)} \end{cases}, \qquad (4.20)$$

and the same stands for the expected value of $\mathbf{b}(i)$:

$$\left\langle \mathbf{b}(i)\right\rangle_{q^{(t+1)}(\mathbf{b}(i))} = \frac{\alpha}{\beta} = \frac{\mu^{(t)}+1}{\mu^{(t)}+\lambda_{noise}^{(t)}\left(\left(\mathbf{G}\mathbf{u}^{(t)}-\mathbf{d}\right)_i^2+\mathbf{F}^{(t)}(i,i)\right)}, \qquad (4.21)$$

where $\langle\cdot\rangle_{q(\cdot)}$ denotes the expectation w.r.t. an arbitrary distribution $q(.)$. These estimates are used in (4.16) and (4.17), where $\hat{\mathbf{A}}_k^{(t)}$ and $\hat{\mathbf{B}}^{(t)}$ are diagonal matrices with elements

$$\hat{\mathbf{A}}_k^{(t)}(i,i) = \left\langle \boldsymbol{\alpha}_k(i)\right\rangle_{q^{(t)}(\boldsymbol{\alpha}_k(i))} \quad \text{and} \quad \hat{\mathbf{B}}^{(t)}(i,i) = \left\langle \mathbf{b}(i)\right\rangle_{q^{(t)}(\mathbf{b}(i))},\ i = 1,\ldots, N.$$

At the variational M-step, the bound is maximized with respect to the model parameters:

$$\text{VM-step:}\quad \theta_2^{(t+1)} = \arg\max_{\theta_2} L\left(q^{(t+1)}(\mathbf{u}_k), q^{(t+1)}\left(\hat{\mathbf{A}}_k\right), q^{(t+1)}\left(\hat{\mathbf{B}}\right), \theta_1^{(t+1)}, \theta_2\right), \text{ where}$$

$$L\left(q^{(t+1)}(\mathbf{u}_k), q^{(t+1)}\left(\hat{\mathbf{A}}_k\right), q^{(t+1)}\left(\hat{\mathbf{B}}\right), \theta_1^{(t+1)}, \theta_2\right) \propto \left\langle \log p\left(\mathbf{d}, \mathbf{u}, \hat{\mathbf{A}}_k, \hat{\mathbf{B}}; \theta_2\right)\right\rangle_{q\left(\mathbf{u}_k;\theta_1^{(t+1)}\right), q^{(t+1)}\left(\hat{\mathbf{A}}_k\right), q^{(t+1)}\left(\hat{\mathbf{B}}\right)}$$

is calculated using the results from (4.16) – (4.19).

The update for λ_{noise} is obtained after taking the derivative of $L\left(q^{(t+1)}(\mathbf{u}_k), q^{(t+1)}\left(\hat{\mathbf{A}}_k\right), q^{(t+1)}\left(\hat{\mathbf{B}}\right), \theta_1^{(t+1)}, \theta_2\right)$ in (4.15) and setting it to zero:

$$\lambda_{noise}^{(t+1)} = \frac{N}{\sum_{i=1}^{N} \mathbf{b}^{(t+1)}(i)\left(\left(\mathbf{G}\mathbf{u}^{(t)} - \mathbf{d}\right)_i^2 + \mathbf{F}^{(t+1)}(i,i)\right)}, \tag{4.22}$$

By the same procedure we obtain:

$$\begin{cases} \lambda_x^{(t+1)} = \dfrac{N}{\sum_{i=1}^{N} \boldsymbol{\alpha}_x^{(t+1)}(i)\left(\left(\mathbf{Q}\mathbf{u}_x^{(t)}\right)_i^2 + \mathbf{C}_x^{(t+1)}(i,i)\right)} \\ \lambda_y^{(t+1)} = \dfrac{N}{\sum_{i=1}^{N} \boldsymbol{\alpha}_y^{(t+1)}(i)\left(\left(\mathbf{Q}\mathbf{u}_y^{(t)}\right)_i^2 + \mathbf{C}_y^{(t+1)}(i,i)\right)} \end{cases}, \tag{4.23}$$

The "degrees of freedom" parameter ν_k of the Student's *t*-distribution is also computed by setting the derivative of (4.15) equal to zero with respect to ν_k:

$$\begin{aligned} \frac{1}{N}\left(\sum_{i=1}^{N} \log\left\langle \boldsymbol{\alpha}_k(i)\right\rangle_{q^{(t+1)}(\mathbf{A}_k)} - \sum_{i=1}^{N}\left\langle \boldsymbol{\alpha}_k(i)\right\rangle_{q^{(t+1)}(\mathbf{A}_k)}\right) + \psi\left(\frac{\nu_k^{(t)}}{2} + \frac{1}{2}\right) \\ - \log\left(\frac{\nu_k^{(t)}}{2} + \frac{1}{2}\right) - \psi\left(\frac{\nu_k}{2}\right) + \log\left(\frac{\nu_k}{2}\right) + 1 = 0 \end{aligned}, \tag{4.24}$$

for ν_k, $\forall k \in \{x, y\}$, where

$$\psi(x) = \frac{d}{dx}\log\Gamma(x) = \frac{\Gamma'(x)}{\Gamma(x)},$$

is the digamma function and $\nu_k^{(t)}$ is the value of ν_k at the previous iteration (t) used to evaluate the expectations in (4.20) during the VE-step.

Finally, by the same procedure we obtain estimates for the parameter μ of the noise distribution

$$\frac{1}{N}\left(\sum_{i=1}^{N}\log\langle\mathbf{b}(i)\rangle_{q^{(t+1)}(\mathbf{b}(i))}-\sum_{i=1}^{N}\langle\mathbf{b}(i)\rangle_{q^{(t+1)}(\mathbf{b}(i))}\right)+\psi\left(\frac{\mu^{(t)}}{2}+\frac{1}{2}\right)$$
$$-\log\left(\frac{\mu^{(t)}}{2}+\frac{1}{2}\right)-\psi\left(\frac{\mu}{2}\right)+\log\left(\frac{\mu}{2}\right)+1=0 \quad , \qquad (4.25)$$

where $\mu^{(t)}$ is the value of μ at the previous iteration (t) used to evaluate the expectations in (4.21) during the VE-step.

In our implementation, we solve (4.16), (4.24) and (4.25) iteratively. For equations (4.24) and (4.25), we employ the bisection method, as also proposed in [27] and [14]. For equation (4.16) we employ a method based on the Lanczos process [5], [33].

To resume, the steps of the Variational EM – algorithm are presented in fig. 4.2.

Algorithm: Variational – Bayesian optical flow method

1: Initialize $\mathbf{u}_x$, $\mathbf{u}_y$ by the *Horn-Schunck* optical flow.
2: **DO** *until* convergence
3: **VE-step:**
4: Compute the expectations $\boldsymbol{\alpha}_{\mathbf{x}}(i)$, $\boldsymbol{\alpha}_{\mathbf{y}}(i)$ using (4.20).
5: Compute the expectation of $\mathbf{b}(i)$ using (4.21).
6: **VM-step:**
7: Compute λ_{noise} using (4.22).
8: Compute λ_x , λ_y using (4.23).
9: Solve for v_x, v_y equation (4.24), using the bisection method.
10: Solve for μ equation (4.25), using the bisection method.
11: Update the mean vectors using (4.16).
12: Update matrices $\mathbf{C}_x$, $\mathbf{C}_y$, $\mathbf{F}$ and $\mathbf{R}_x$, $\mathbf{R}_y$ using (4.17) and the *Lanczos* method.
13: Solve (4.16) to obtain the values of $\mathbf{m}_x$, $\mathbf{m}_y$.
14: Set $[\mathbf{u}_x, \mathbf{u}_y] := [\mathbf{m}_x , \mathbf{m}_y]$.
15: **END DO**

Figure 4.2: The steps of the proposed method.

4.5. Experimental Results and Discussion

The proposed method was tested on image sequences including both synthetic and real scenes. Some of the synthetic images were synthesized from us and other were taken from publicly available data sets as for example the Middlebury public flow dataset [3], [4, 29] in order to guarantee the objectiveness of our evaluations.

More specifically, we tested our method on three synthetic sequences. The first sequence is showing a *Textured Square* moving from the center to the top left corner by one pixel. The second one is showing two *Textured Triangles*. The upper left triangle moves about one pixel to the bottom left corner, while the bottom right triangle moves about one pixel to the bottom right corner. The third one is showing two *Textured Triangles* with the only difference from the second synthetic sequence that here the bottom right triangle moves about 2 pixels to the bottom right corner. The background color for all the previously mentioned sequences is black, without loss of generality.

Additionally, we tested our method on the well-known *Yosemite* sequence without clouds and the *Dimetrodon* sequence [3] (which contains hidden texture). We compared our approach with the algorithms of Horn-Schunck [25], and Lucas-Kanade [28]. For the evaluation of our method we used the error metrics described in section 2.4.

4.5.1. Textured-Square Sequence

This is a simple 256 x 256 example, which consists of a textured square located at the center of the first frame, while at the second frame it moves by one pixel towards the top left corner. Figure 4.3 shows a frame of the image and figure 4.4 shows optical flow estimations from the compared methods along with the ground truth. Figure 4.5 shows the angular error and figure 4.6 presents the flow by using color coding [3]. We do not show the end-point error for each pixel as it has too small values (but we are showing the average end-point error, which is equivalent and more meaningful).

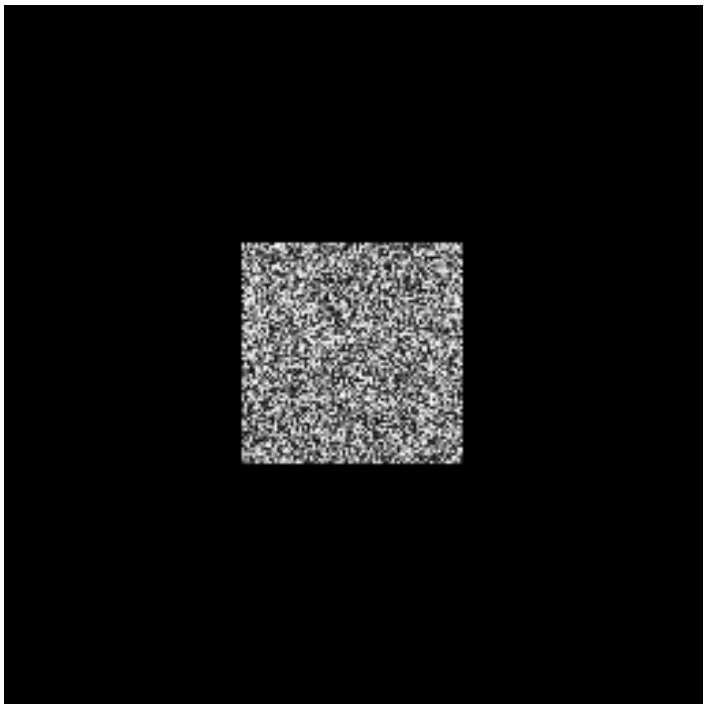
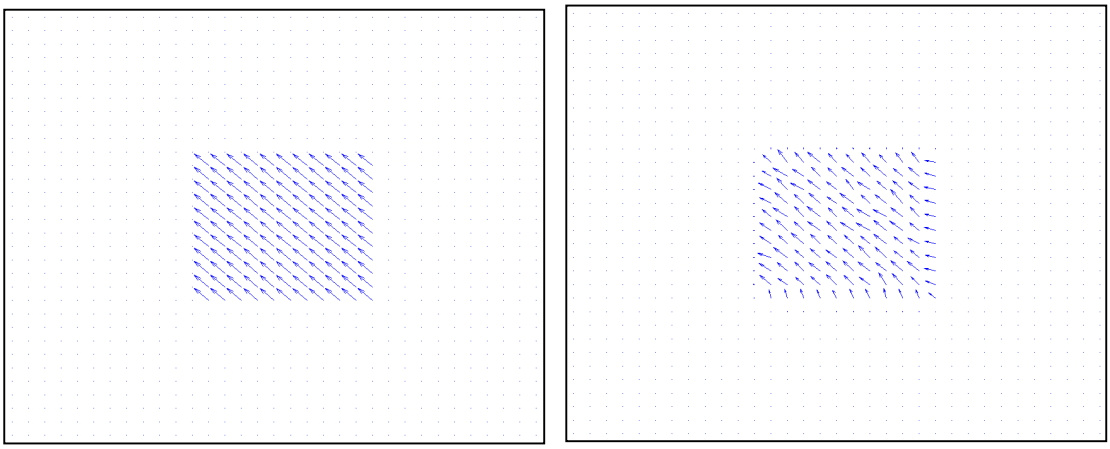
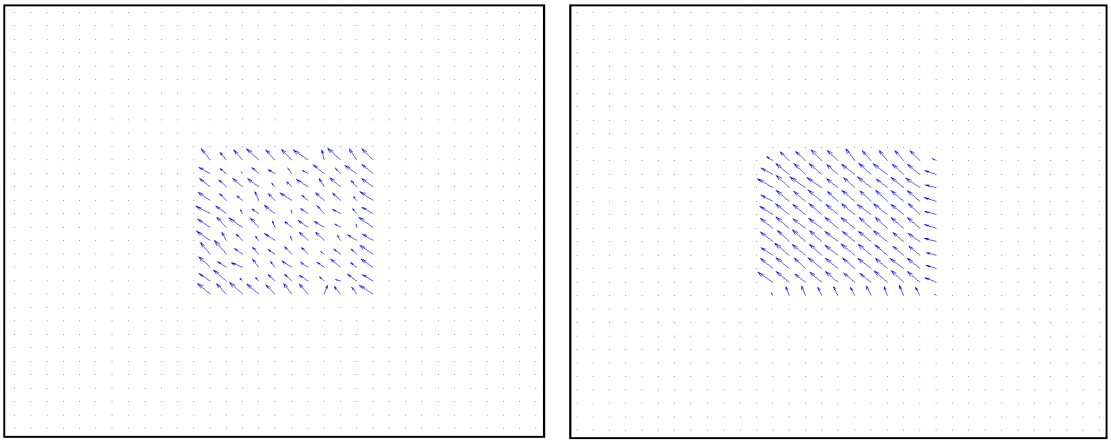


Figure 4.3: Textured-square sequence: first frame of the sequence.

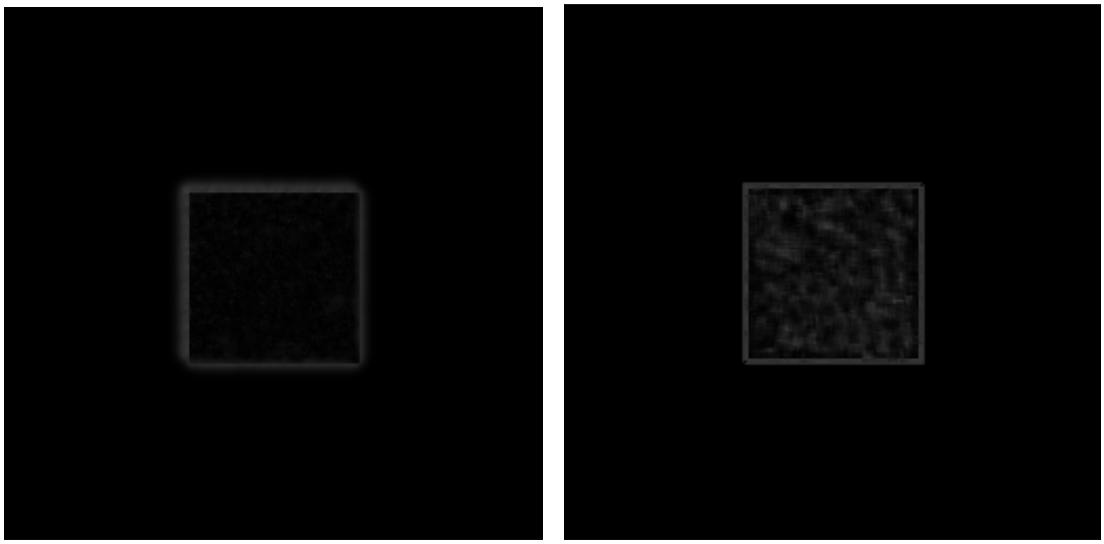


(a) *(b)*

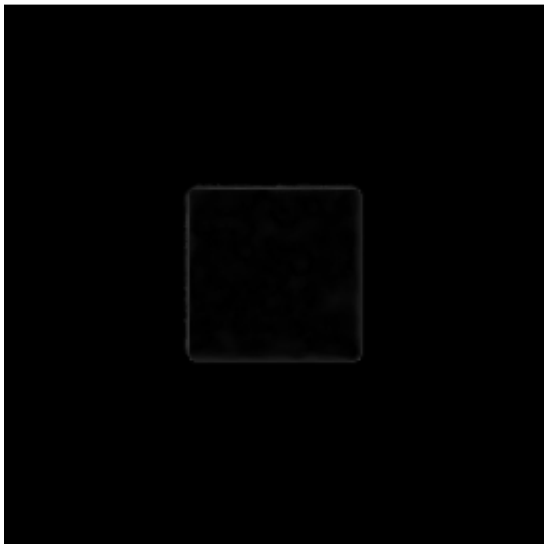


(c) *(d)*

Figure 4.4: Textured-square sequence: (a) ground truth optical flow, (b) optical flow initialization using the method of Horn-Schunck [25], *(c) optical flow using the method of Lucas-Kanade* [28] *and (d) resulting optical flow of the proposed method.*



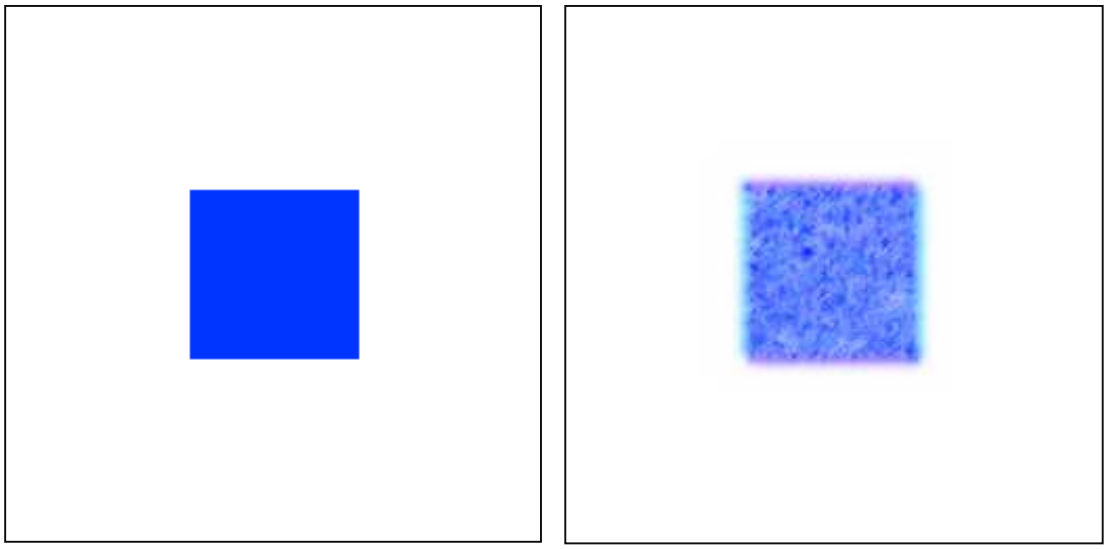
(a) *(b)*



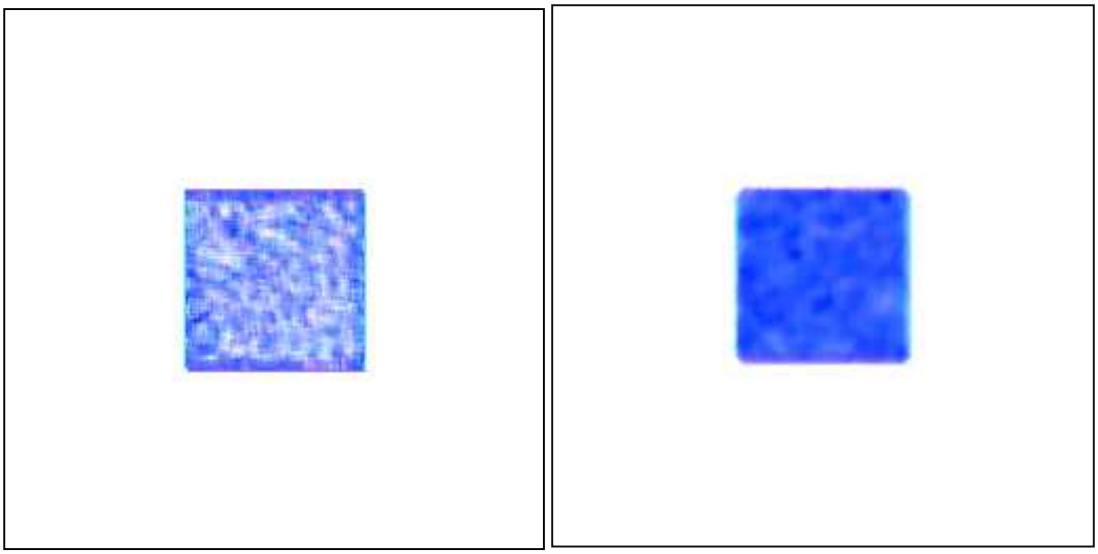
(c)

Figure 4.5: Textured-square's Angular Error (AE) of the compared methods. (a) Initial AE using the method of Horn-Schunck [25]*, (b) AE using the method of Lucas-Kanade* [28] *and (c) AE of the proposed method.*

(*a*)



(b) *(c)*



(d) *(e)*

Figure 4.6: Textured-square sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) initial flow field using the method of Horn-Schunck [25]*, (d) flow field using the method of Lucas-Kanade* [28] *and (e) flow field of the proposed method.*

Table 4.1: Average error metrics for the Textured-square sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	3.09	0.08	0.08
Horn-Schunck [25]	1.84	0.04	0.04
Joint Lucas-Kanade [7]	2.67	0.04	0.05
Nagel et al [32]	1.60	0.04	0.04
Method of section 3.3	1.50	0.05	0.04
Method of section 3.4	1.46	0.04	0.04
Proposed method (Chapter 4)	**0.76**	**0.02**	**0.02**

As we can see from table 4.1 our approach achieves smaller errors than Horn-Schunck and Lucas-Kanade method, for all the error metrics.

4.5.2. Textured-Triangles with equal in Norm Moves

This is a slightly more complicated 256 x 256 example, which consists of two textured triangles located at the top left corner and at the bottom right corner of the first frame, while at the second frame the upper left triangle moves by one pixel to the bottom left corner, while the bottom right triangle moves by one pixel to the bottom right corner. Figure 4.7 shows the image and figure 4.8 the estimated optical flow. Figure 4.9 shows the angular error and figure 4.10 presents the flow by using color coding [3]. We do not show the end-point error for each pixel as it has too small values (but we are showing the average end-point error, which is equivalent and more meaningful).

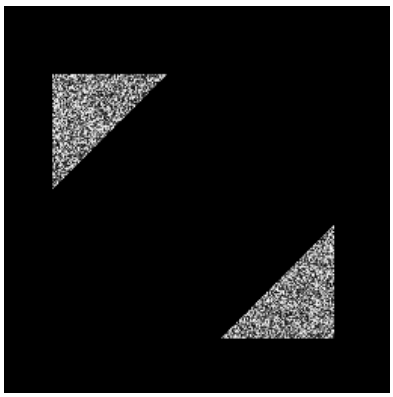
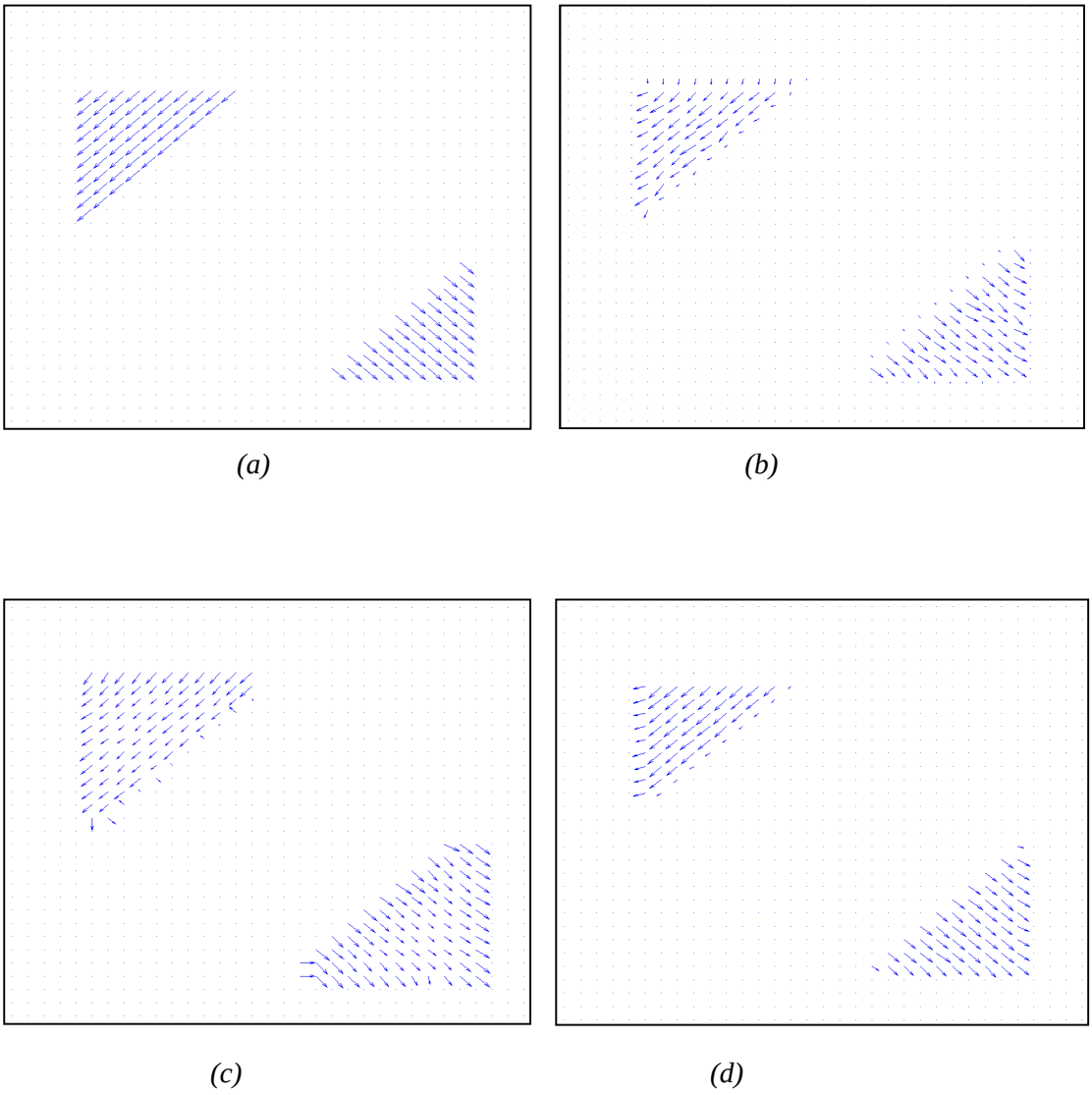


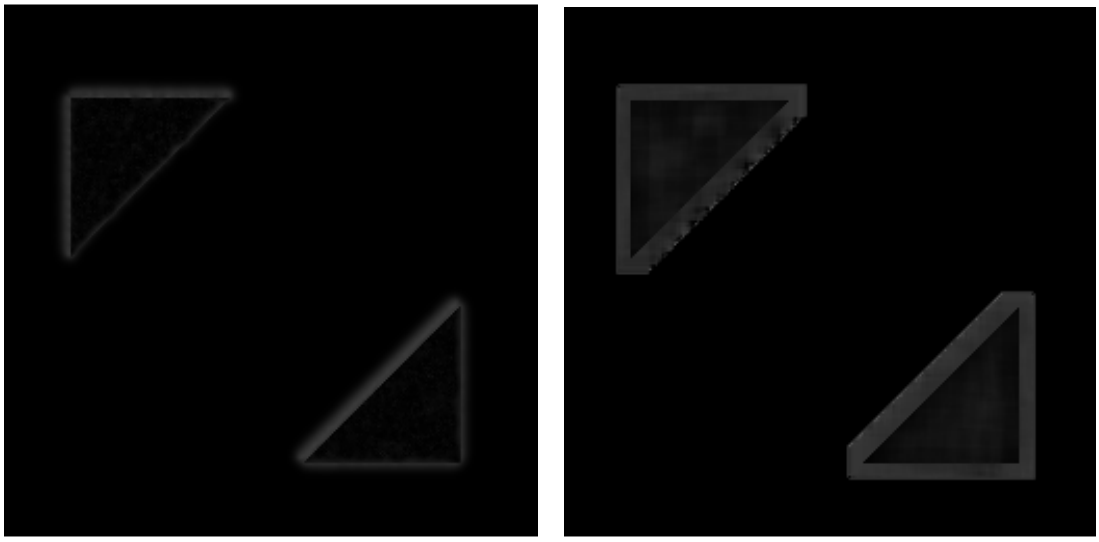
Figure 4.7: Textured-triangles (with equal in norm moves) sequence: first frame of the sequence.



(a) *(b)*

(c) *(d)*

Figure 4.8: Textured-triangles (with equal in norm moves) sequence: (a) ground truth optical flow, (b) optical flow initialization using the method of Horn-Schunck [25]*, (c) optical flow using the method of Lucas-Kanade* [28] *and (d) resulting optical flow of the proposed method.*

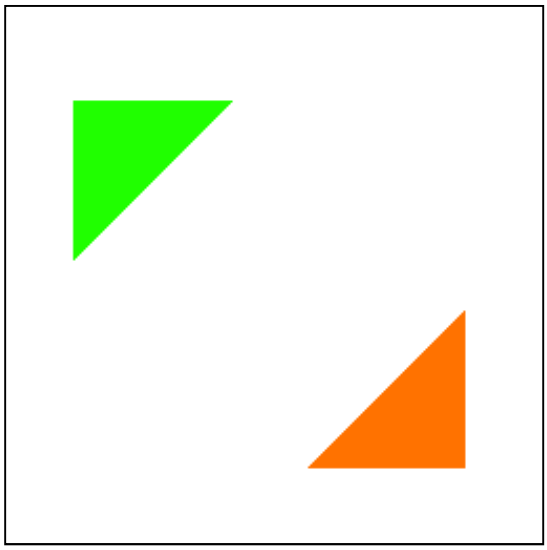


(a) *(b)*

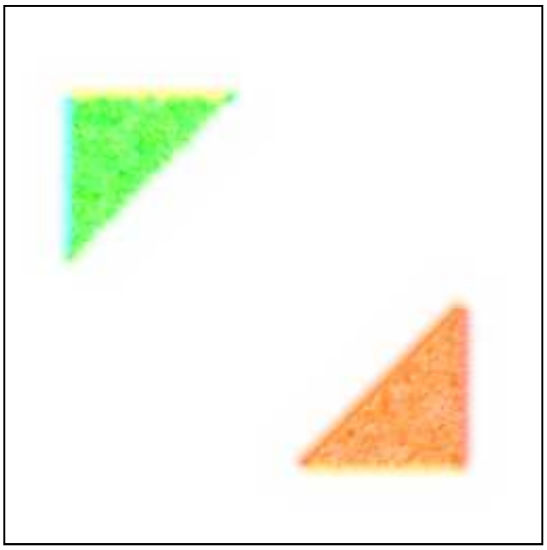
(c)

Figure 4.9: Textured-triangles' (with equal in norm moves) Angular Error (AE) of the compared methods. (a) Initial AE using the method of Horn-Schunck [25], *(b) AE using the method of Lucas-Kanade* [28] *and (c) AE of the proposed method.*

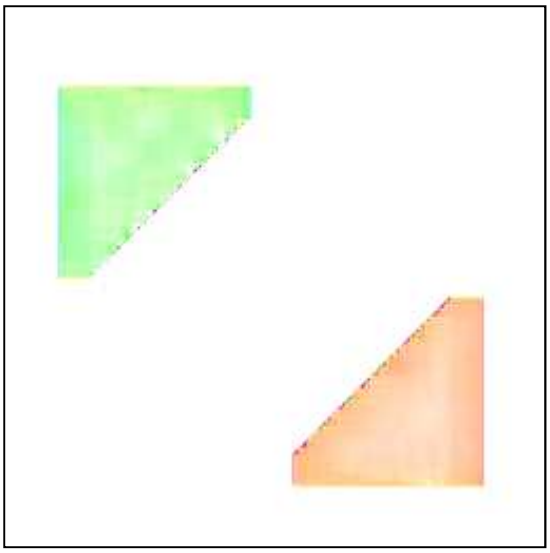
(*a*)



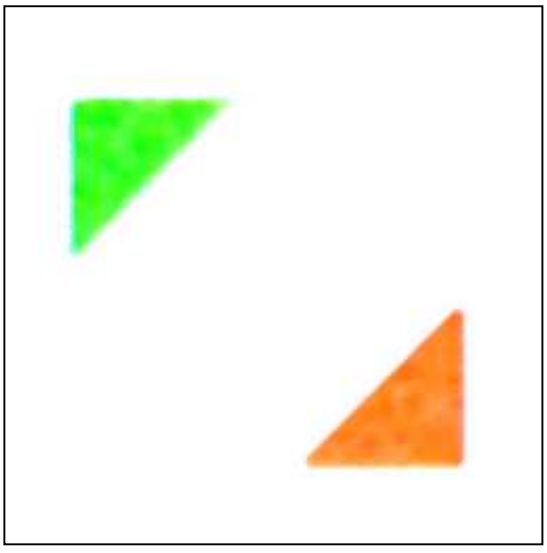
(*b*)



(*c*)



(*d*)



(*e*)

Figure 4.10: Textured-triangles (with equal in norm moves) sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) initial flow field using the method of Horn-Schunck [25], *(d) flow field using the method of Lucas-Kanade* [28] *and (e) flow field of the proposed method.*

Table 4.2: Average error metrics for the Textured-triangles (with equal in norm moves) sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	5.91	0.15	0.14
Horn-Schunck [25]	2.47	0.05	0.05
Joint Lucas-Kanade [7]	4.10	0.07	0.08
Nagel et al [32]	2.25	0.06	0.05
Method of section 3.3	2.33	0.08	0.05
Method of section 3.4	2.26	0.07	0.05
Proposed method (Chapter 4)	**1.06**	**0.02**	**0.03**

As we can see from table 4.2 our approach achieves smaller errors than Horn-Schunck and Lucas-Kanade method, for all the error metrics. Additionally, by observing figure 4.10(d), we understand that Lucas-Kanade method have problems estimating the motion vectors at the edges of the objects.

4.5.3. Textured-Triangles with unequal in Norm Moves

A next experiment consists in increasing the difficulty of the previous configurations. We have a 256 x 256 example, which consists of two textured triangles located at the top left corner and at the bottom right corner of the first frame, while at the second frame the upper left triangle moves by one pixel to the bottom left corner, while the bottom right triangle moves by two pixel to the bottom right corner. Figure 4.11 shows the image and figure 4.12 the estimated optical flow. Figure 4.13 shows the angular error and figure 4.14 presents the flow by using color coding [3]. We do not show the end-point error for each pixel as it has too small values (but we are showing the average end-point error, which is equivalent and more meaningful).

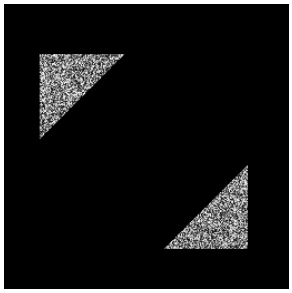
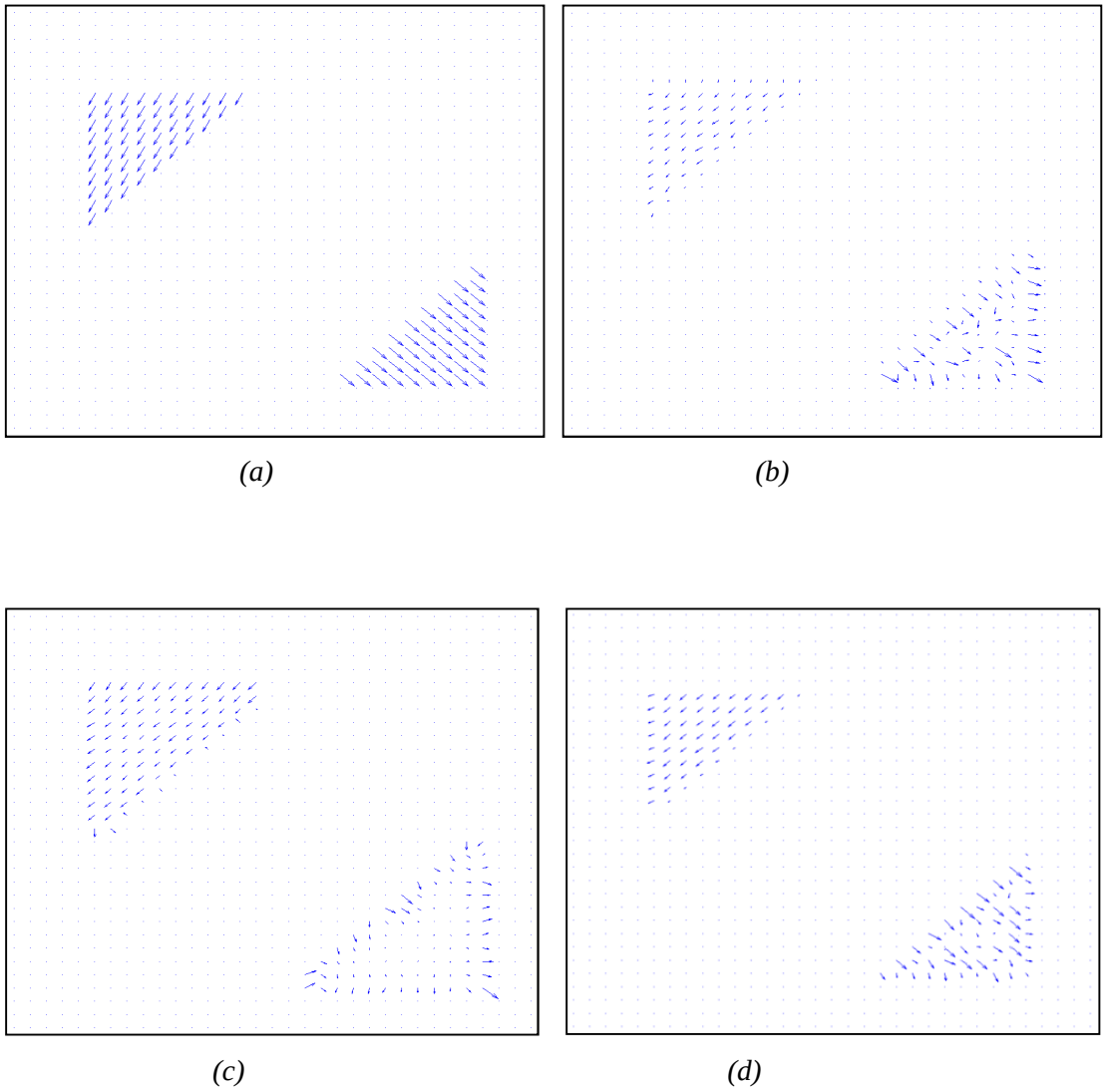


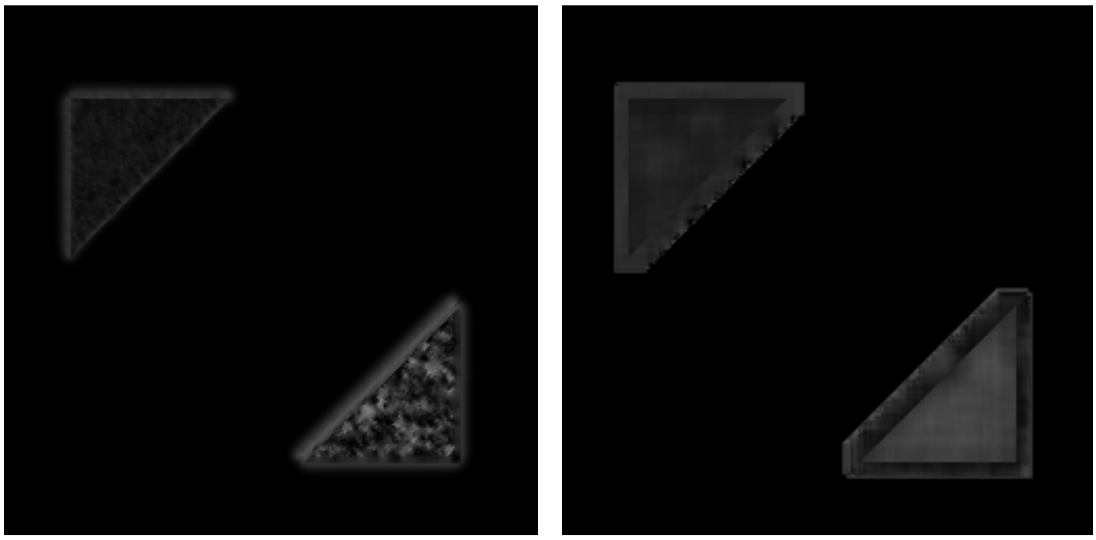
Figure 4.11: Textured-triangles (with unequal in norm moves) sequence: first frame of the sequence.



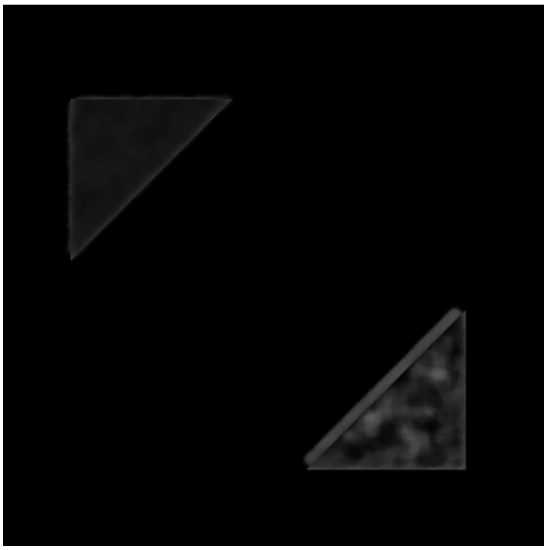
(a) *(b)*

(c) *(d)*

Figure 4.12: Textured-triangles (with unequal in norm moves) sequence: (a) ground truth optical flow, (b) optical flow initialization using the method of Horn-Schunck [25], *(c) optical flow using the method of Lucas-Kanade* [28] *and (d) resulting optical flow of the proposed method.*



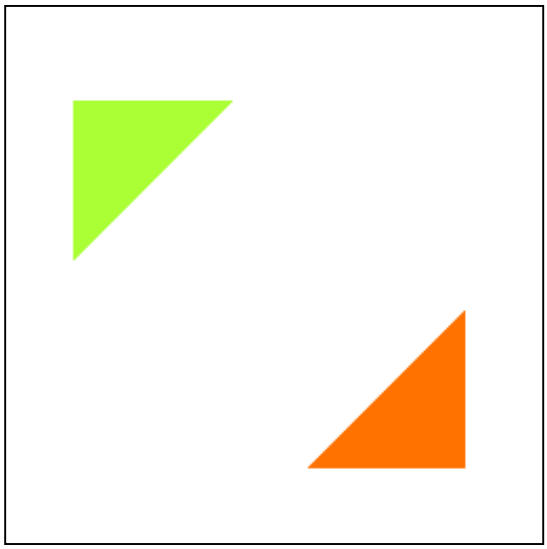
(a) *(b)*



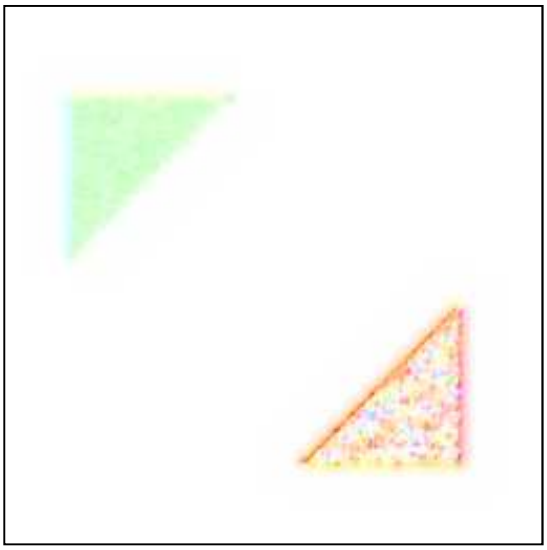
(c)

Figure 4.13: Textured-triangles' (with unequal in norm moves) Angular Error (AE) of the compared methods. (a) Initial AE using the method of Horn-Schunck [25], *(b) AE using the method of Lucas-Kanade* [28] *and (c) AE of the proposed method.*

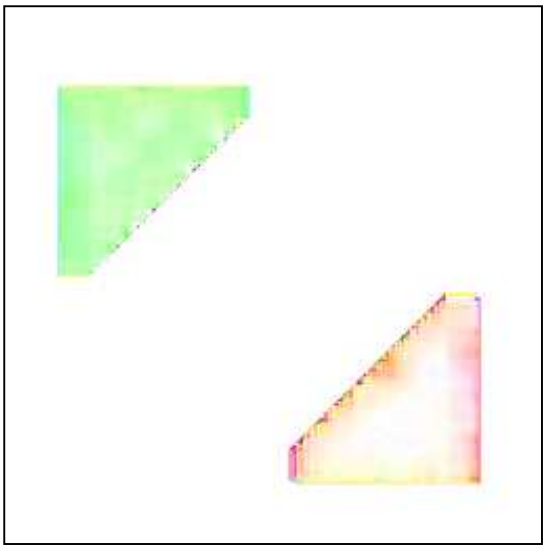
(*a*)



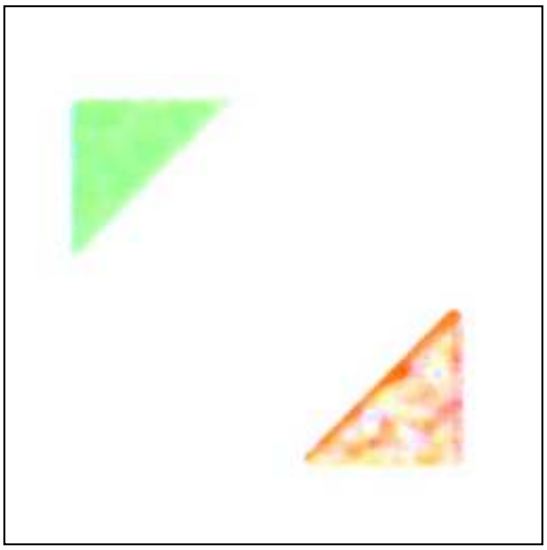
(b)



(c)



(d)



(e)

Figure 4.14: Textured-triangles (with unequal in norm moves) sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) initial flow field using the method of Horn-Schunck [25]*, (d) flow field using the method of Lucas-Kanade* [28] *and (e) flow field of the proposed method.*

Table 4.3: Average error metrics for the Textured-triangles (with unequal in norm moves) sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	8.58	0.17	0.26
Horn-Schunck [25]	5.57	0.14	0.19
Joint Lucas-Kanade [7]	6.95	0.18	0.22
Nagel et al [32]	4.79	0.13	0.18
Method of section 3.3	4.67	0.17	0.17
Method of section 3.4	4.78	0.16	0.18
Proposed method (Chapter 4)	**3.93**	**0.10**	**0.16**

As we can see from table 4.3 our approach achieves smaller errors than Horn-Schunck and Lucas-Kanade method, for all the error metrics, although all methods did not have perfectly estimations. Additionally, by observing figure 4.14(d), we understand that Lucas-Kanade method have problems estimating the motion vectors at the edges of the objects.

4.5.4. Yosemite Sequence without Clouds

The *Yosemite* sequence without clouds, is available at `http://www.cs.brown.edu/people/black/images.html`.

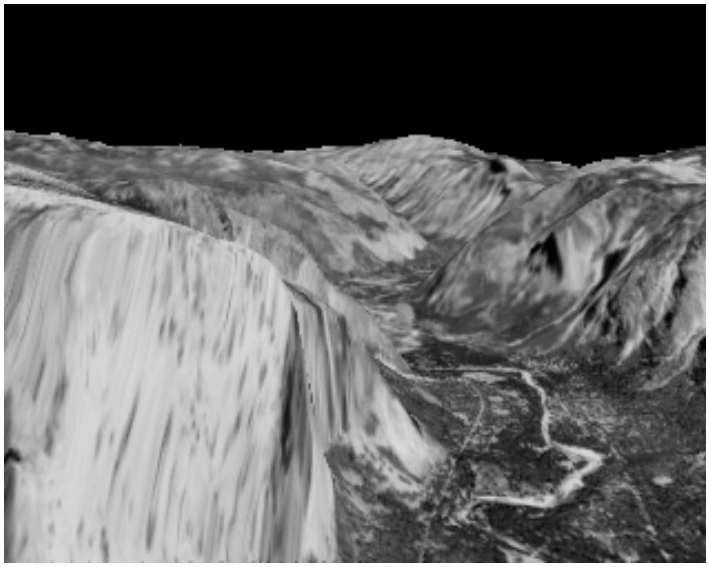


Figure 4.15: Yosemite sequence without clouds: first frame of the sequence.

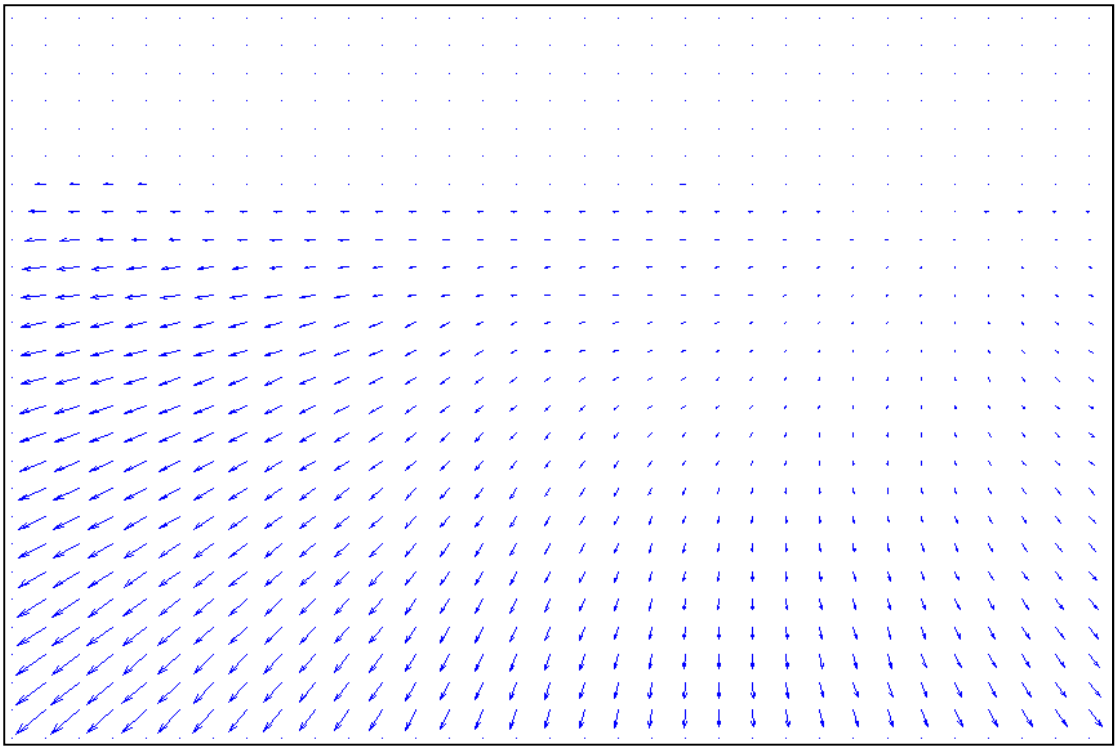


Figure 4.16: Yosemite sequence without clouds: ground truth optical flow.

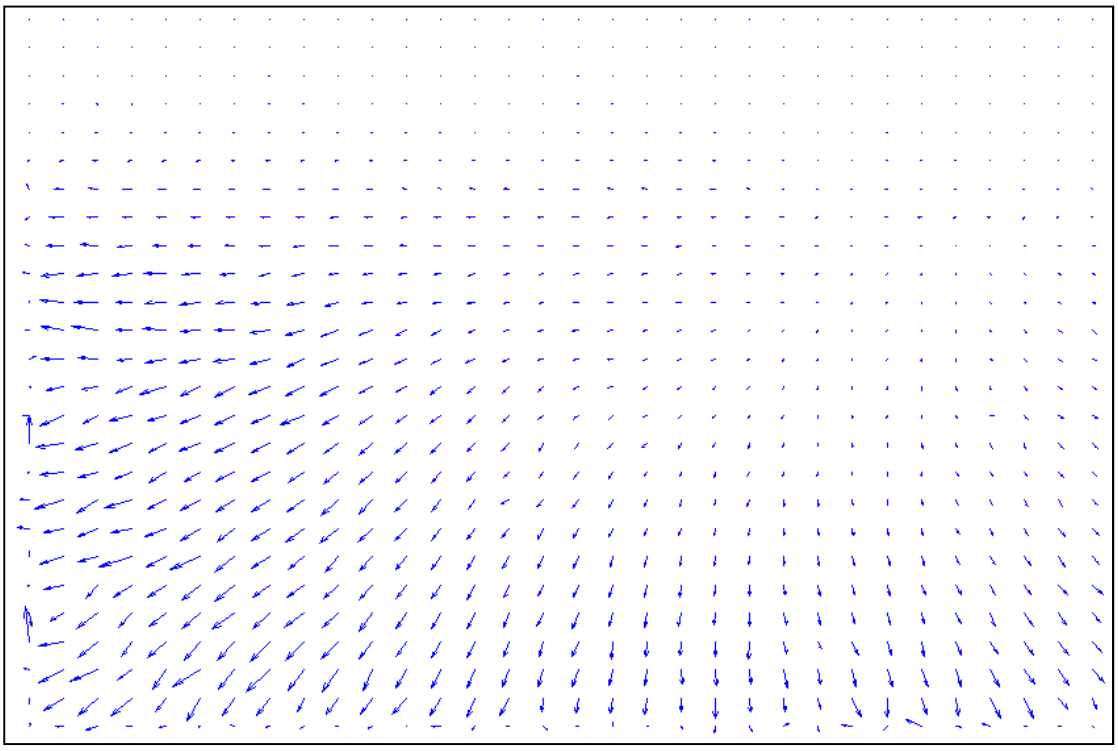


Figure 4.17: Yosemite sequence without clouds: optical flow initialization using the method of Horn-Schunck [25].

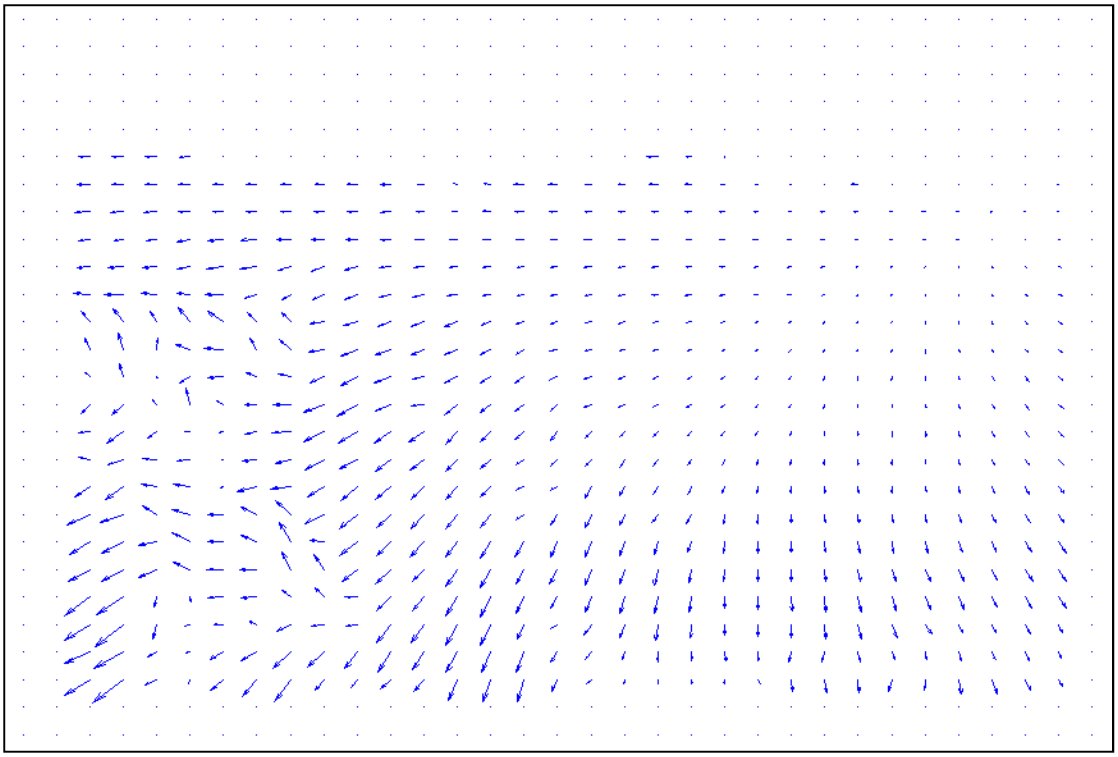


Figure 4.18: Yosemite sequence without clouds: optical flow using the method of Lucas-Kanade [28].

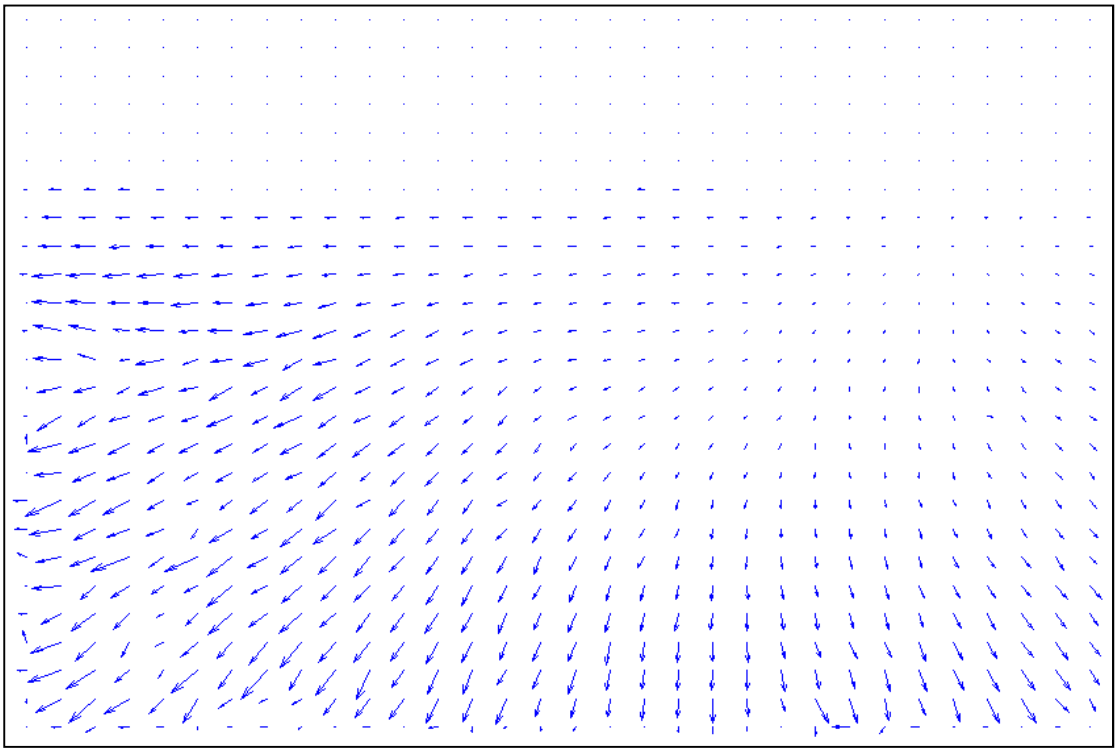
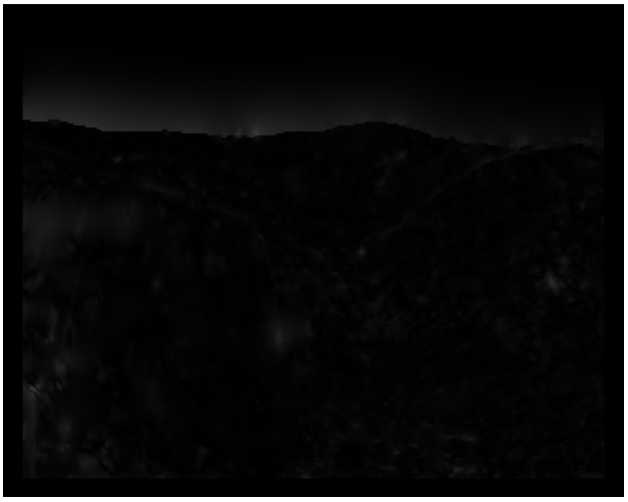
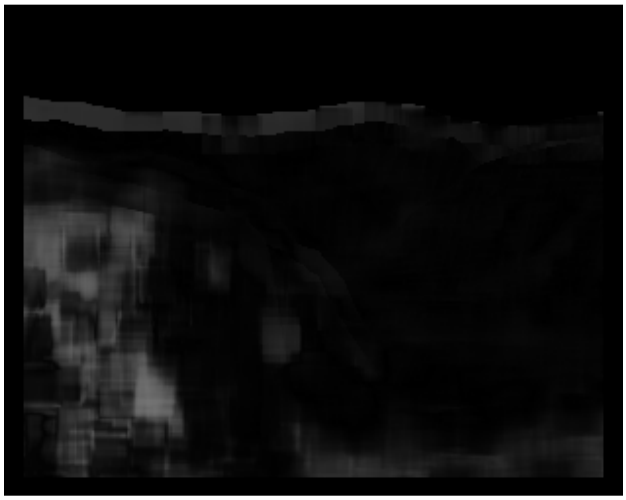


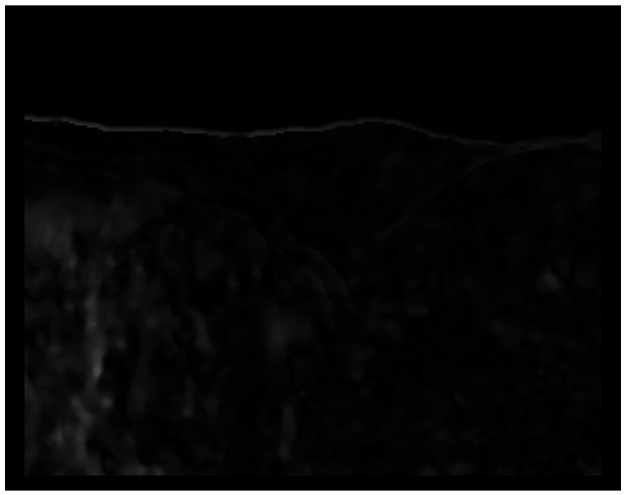
Figure 4.19: Yosemite sequence without clouds: resulting optical flow of the proposed method.



(a)



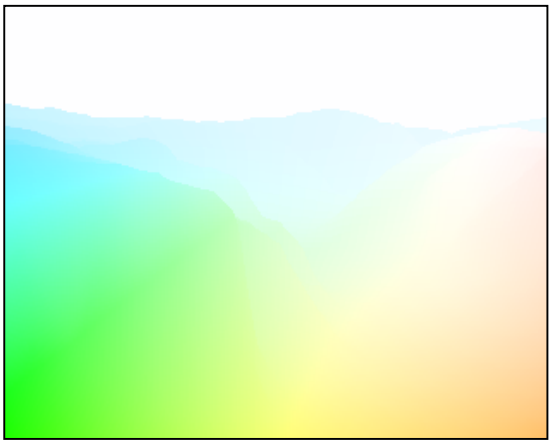
(b)



(c)

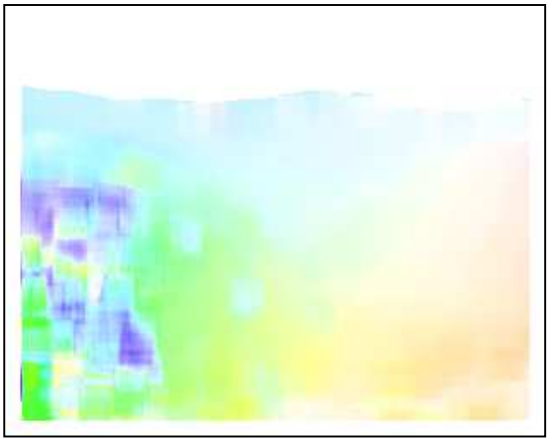
Figure 4.20: Yosemite's Angular Error (AE) of the compared methods. (a) Initial AE using the method of Horn-Schunck [25]*, (b) AE using the method of Lucas-Kanade* [28] *and (c) AE of the proposed method.*

(a)



(b)

(c)



(d)

(e)

Figure 4.21: Yosemite sequence without clouds: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) initial flow field using the method of Horn-Schunck [25]*, (d) flow field using the method of Lucas-Kanade* [28] *and (e) flow field of the proposed method.*

Table 4.4: Average error metrics for the Yosemite without Clouds sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**EP** (in pixels)
Lucas-Kanade [28]	11.65	0.23	0.48
Horn-Schunck [25]	5.43	0.10	0.20
Joint Lucas-Kanade [7]	7.97	0.17	0.35
Nagel et al [32]	9.15	0.19	0.36
Method of section 3.3	5.12	0.12	0.22
Method of section 3.4	**3.79**	**0.09**	**0.15**
Proposed method (Chapter 4)	4.45	0.11	0.24

As we can see from the table 4.4 our approach is better than Horn-Schunck method at the average angular error metric (which is the most important), slightly worse for the average magnitude error (difference 0.01) but *HS* must know the exact value of the deterministic parameter and also slightly worse for the average end-point error (difference 0.04).

4.5.5. Dimetrodon Sequence

The Dimetrodon sequence is obtained from the Middlebury database [3]. It contains nonrigid motion and large areas with little (hidden or not) texture.

Figure 4.22: Dimetrodon sequence: first frame of the sequence.

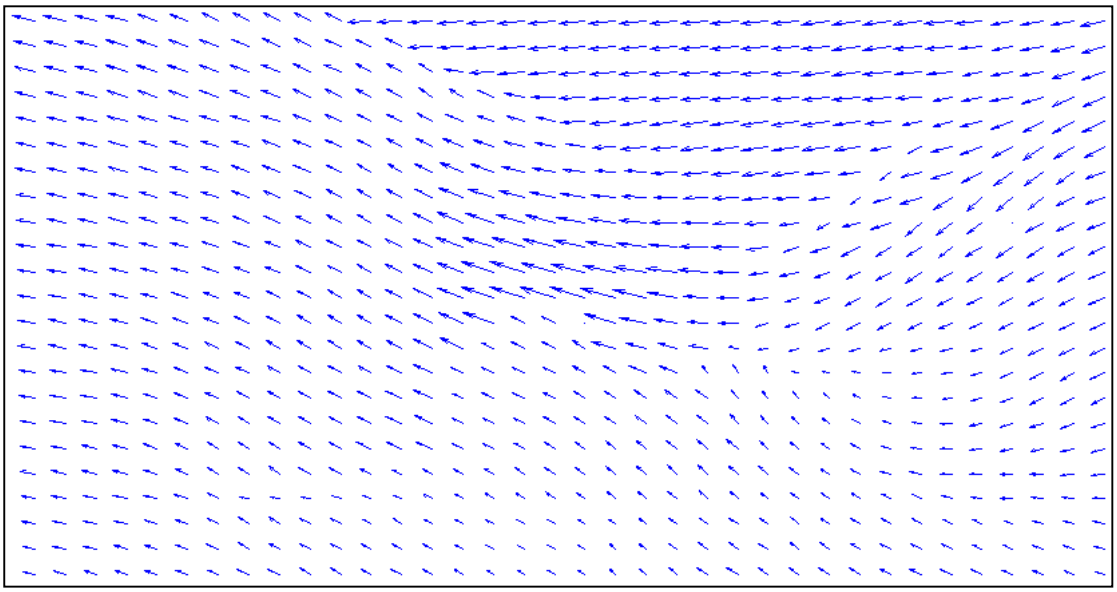


Figure 4.23: Dimetrodon sequence: ground truth optical flow.

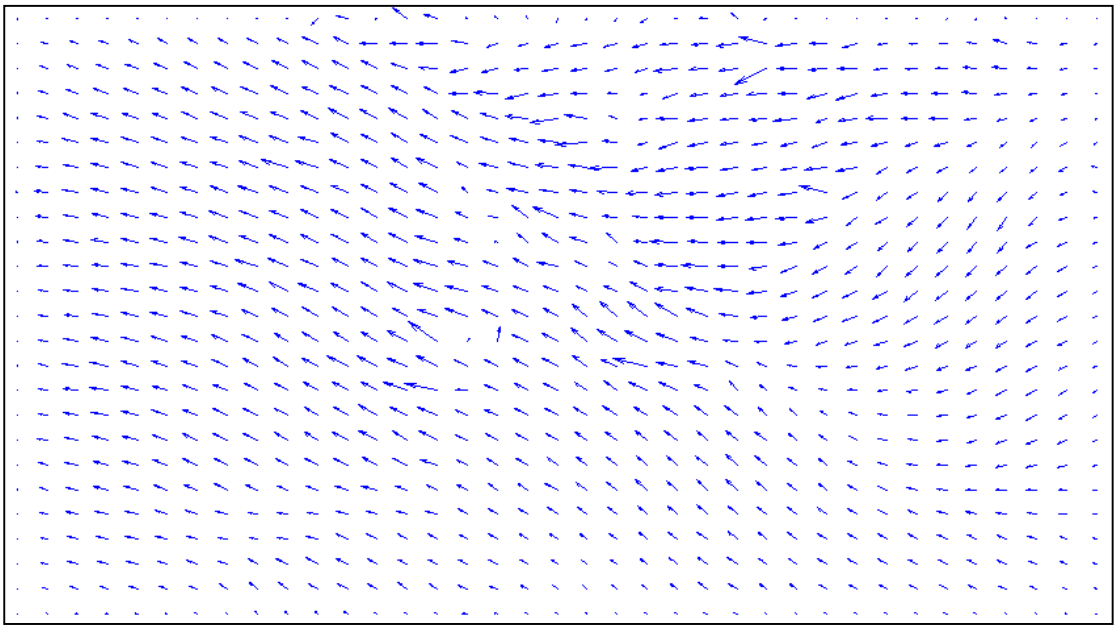


Figure 4.24: Dimetrodon sequence: optical flow initialization using the method of Horn-Schunck [25].

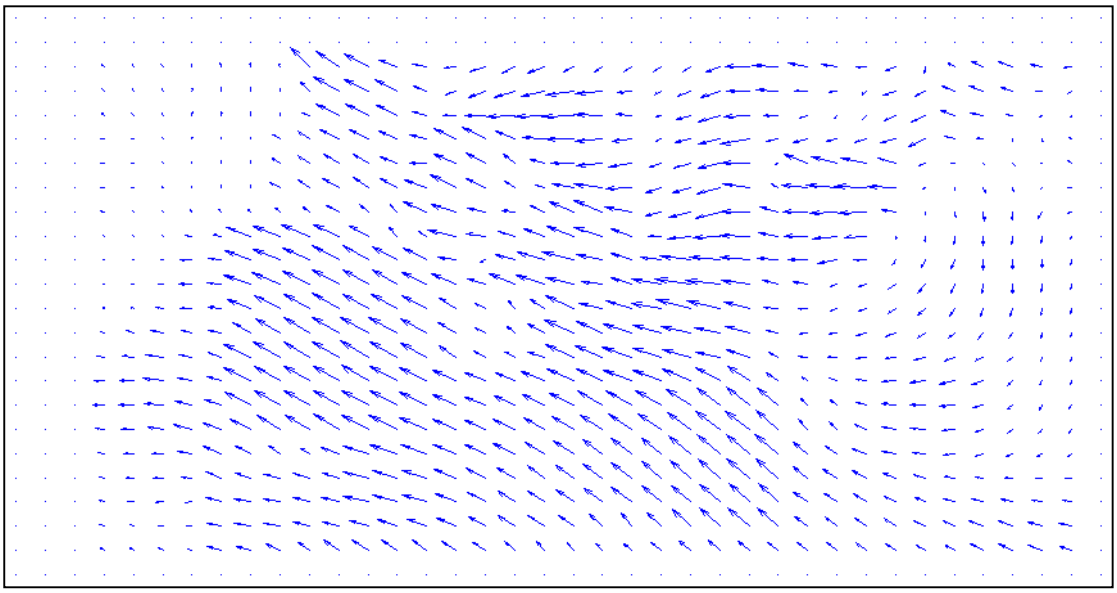


Figure 4.25: Dimetrodon sequence: optical flow using the method of Lucas-Kanade [28].

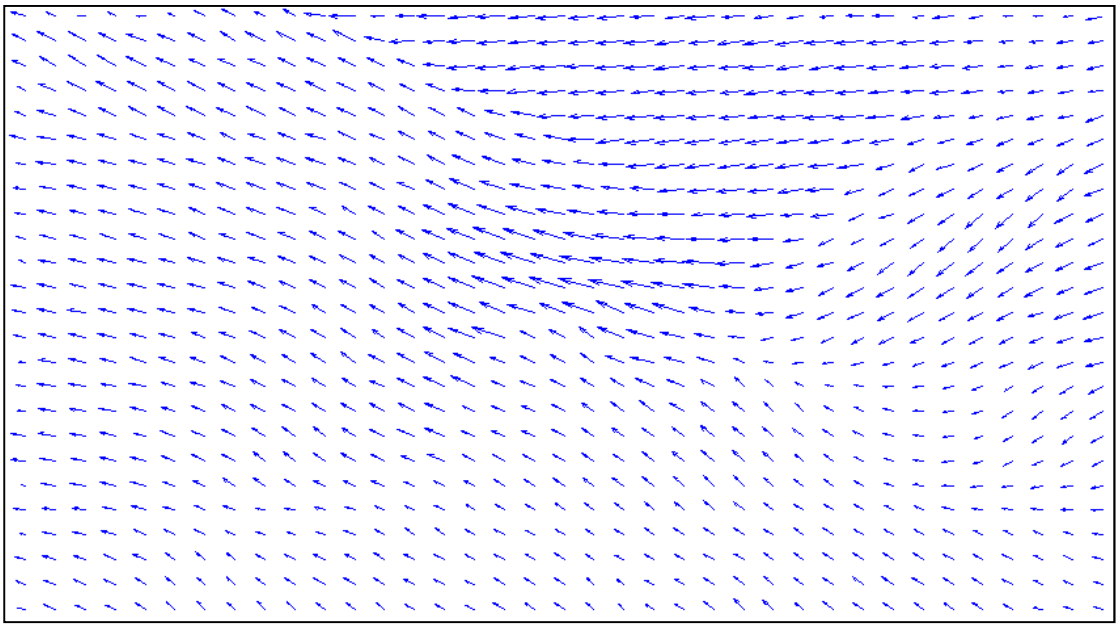
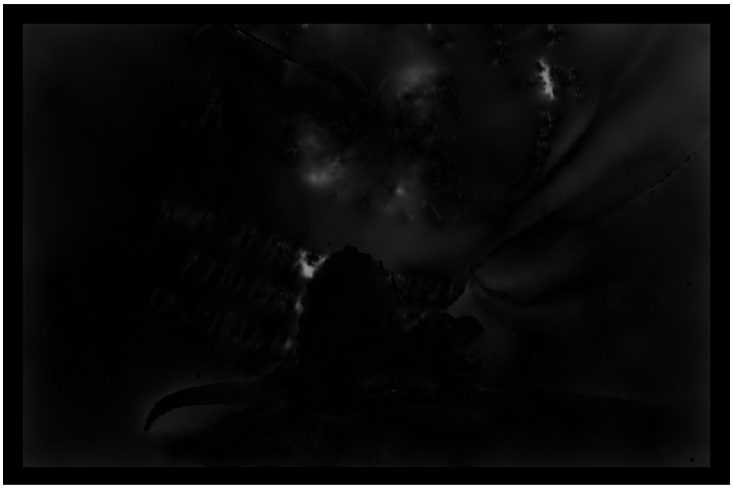


Figure 4.26: Dimetrodon sequence: resulting optical flow of the proposed method.



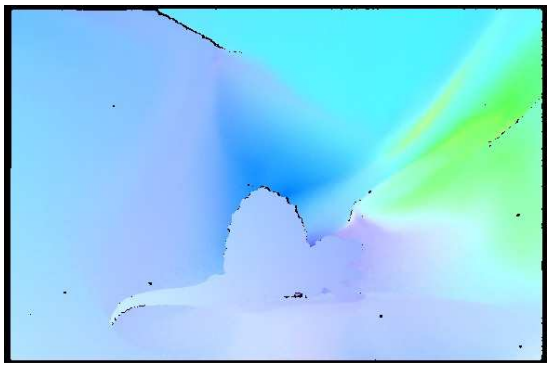
(a)

(b)

(c)

Figure 4.27: Dimetrodon's Angular Error (AE) of the compared methods. (a) Initial AE using the method of Horn-Schunck [25]*, (b) AE using the method of Lucas-Kanade* [28] *and (c) AE of the proposed method.*

(a)



(b)

(c)

(d)

(e)

Figure 4.28: Dimetrodon sequence: colorful optical flow. (a) Flow field color coding, (b) ground truth, (c) initial flow field using the method of Horn-Schunck [25]*, (d) flow field using the method of Lucas-Kanade* [28] *and (e) flow field of the proposed method.*

Table 4.5: Average error metrics for the Dimetrodon sequence.

Method	**AAE** (in degrees)	**AME** (in pixels)	**Avg EP** (in pixels)
Lucas-Kanade [28]	27.52	0.56	1.07
Horn- Schunck [25]	8.51	0.24	0.49
Joint Lucas-Kanade [7]	33.14	0.65	0.35
Nagel et al [32]	17.58	0.38	1.17
Method of section 3.3	10.17	0.24	0.52
Method of section 3.4	6.24	0.18	0.36
Proposed method (Chapter 4)	**4.31**	**0.13**	**0.22**

As we can see from table 4.5 our approach is better than both Horn-Schunck method and Lucas-Kanade, for all the error metrics.

4.6. Partial Conclusion

At the beginning, let's discuss the reason why in some experiments we do not manage better results than the HS method and LK method. Although our method is more flexible than *HS* method, since we allow every pixel to move independently in the spatial domain, it has more parameters to fix. This will be also a disadvantage, if we have to deal with sequences which contain "simple" moves.

Secondly, our method obtains better estimations when we have a variety of different in norm movements than Horn-Schunck and Lucas-Kanade methods produce (as we can see from section 4.5.4).

One issue which is worth proposing for future work, is to update a part of the equations (4.18 – 4.23) at each step, since the parameters λ_{noise}, λ_x, λ_y tend to increase their values rapidly, while $\mathbf{A}_x$, $\mathbf{A}_y$ and $\mathbf{b}(i)$ more slowly.

CHAPTER 5. CONCLUSION AND FUTURE WORK

5.1. Conclusion

In the present thesis, we studied the fundamental problem of optical flow, located in the area of computer vision, but also we proposed three methods in order to solve it.

More detail, in chapter 2 we studied three classic methods, the Lucas-Kanade (LK) method [28], the Horn-Schunck (HS) method [25] and the affine optical flow method [39]. Next, in chapter 3, we studied two variations of the LK and HS methods, firstly the Joint Lucas-Kanade [7] and secondly the method proposed from Nagel *et al.* [32] where they use adaptive smoothness constraints. Additionally, we analyze our suggestions in order to improve those methods and we show experimental result. Finally, in chapter 4, was presented a brand new approach, which was inspired from the HS method and was imposed stochastic parameters instead of stationary that were used in HS.

5.2. Future work

To begin with, one improvement for the methods proposed in chapter 3, is to find a suitable method to approximate the second order derivatives which they were

used in the linear system. Secondly, you can use another method for the image segmentation which will give more competitive results. Thirdly, you can experiment in finding the suitable type of the neighboring kernel, instead of the average kernel which we used, but also the size of it.

As for the method proposed in chapter 4, one worthy effort is to find a different algorithm instead of Lanzcos method, in order to solve the iterative system of this chapter.

REFERENCES

[1] W. F. Ames, “Numerical methods for partial differential equations”, Academic Press, New York, 1977.

[2] P. Anandan, “A computational framework and an algorithm for the measurement of visual motion”, International Journal of Computer Vision (ICCV), Vol. 2, No 3, pp. 283–310, 1989.

[3] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. Black, and R. Szeliski. “A database and evaluation methodology for optical flow”. In Proceedings of the International Conference of Computer Vision (ICCV), pp. 1-8, 2007.

[4] J. Barron, D. Fleet and S. Beauchemin. “Performance of optical flow techniques”. International Journal of Computer Vision, Vol. 12, No 1, pp. 43–77, 1994.

[5] M. Beal, “Variational algorithms for approximate Bayesian inference”, Ph.D. dissertation, The Gatsby Computational Neuroscience Unit, University College, London, 2003.

[6] S. Birchfield, “KLT: An implementation of the Kanade-Lucas-Tomasi feature tracker”, http://www.ces.clemson.edu/˜stb/klt/.

[7] S. T. Birchfield and S. J. Pundlik, “Joint tracking of features and edges”, In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2008.

[8] C. Bishop, “Pattern Recognition and Machine Learning”, Springer, 2006.

[9] M. J. Black and P. Anandan. “The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields”. Computer Vision and Image Understanding, Vol. 63, No 1, pp. 75–104, 1996.

[10] J.-Y. Bouguet, “Pyramidal implementation of the Lucas Kanade feature tracker”, OpenCV documentation, Intel Corporation, Microprocessor Research Labs, 1999.

[11] T. Brox, A. Bruhn, and J. Weickert, “Variational motion segmentation with level sets”, In Proceedings of the European Conference on Computer Vision (ECCV), pp. I: 471–483, May 2006.

[12] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, “High accuracy optical flow estimation based on a theory for warping”, In proceedings of the 8th European Conference on Computer Vision (ECCV), Vol. 4, pp. 25-36, 2004.

[13] A. Bruhn, J. Weickert, and C. Schnorr. “Lucas/Kanade meets Horn/Schunck: Combining local and global optic flow methods”, International Journal of Computer Vision, Vol. 61, No 3, pp. 211–231, 2005.

[14] G. Chantas, N. Galatsanos, A. Likas and M. Saunders, “Variational Bayesian image restoration based on a product of *t*-Distributions image prior”, IEEE Transactions on Image Processing, Vol. 17, No 10, pp. 1795-1805 October 2008.

[15] Edwin K. P. Chong and Stanislaw H. Zak, “An introduction to optimization”, Wiley, 2000.

[16] S. D. Conte and C. de Boor, “Elementary numerical analysis”, McGraw-Hill, New York, 1965, 1972.

[17] D. Cremers and S. Soatto, “Motion competition: a variational approach to piecewise parametric motion”, International Journal of Computer Vision, Vol. 62, No 3, pp. 249–265, May 2005.

[18] D. Forsyth and J. Ponce, “Computer Vision: a modern approach”, Prentice Hall, 2003.

[19] D. Gerogiannis, C. Nikou and A. Likas, “The mixtures of Student’s *t*-distributions as a robust framework for rigid registration”, Image and Vision Computing, Vol. 27, No 9, pp. 1285–1294, 2009.

[20] J. J. Gibson, “The perception of the visual world”, Riverside Press, Cambridge, 1950.

[21] J. J. Gibson, “The senses considered as perceptual systems”, Houghton-Mifflin, Boston, MA, 1966.

[22] J. J. Gibson, “On the analysis of change in the optic array”, Scandinavian Journal of Psychology, Vol. 18, No 3, pp. 161-163, 1977.

[23] R. W. Hamming, “Numerical methods for scientists and engineers”, McGraw-Hill, New York, 1962.

[24] F. B. Hildebrand, “Introduction to numerical analysis”, McGraw-Hill, New York, 1956, 1974.

[25] B. K. P. Horn and B. G. Schunck, “Determining optical flow”, Artificial Intelligence, Vol. 17, No 1-3, pp. 185–203, 1981.

[26] A. Likas and N. Galatsanos, “A variational approach for Bayesian blind image deconvolution”, IEEE Transactions on Signal Processing, Vol. 52, No 8, pp. 2222 – 2233, August 2004.

[27] C. Liu and D. B. Rubin, “ML estimation of the t-distribution using EM and its extensions”, ECM and ECME, Statistica Sinica, 5, 19-39, 1995.

[28] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision”, In Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI), pp. 674–679, 1981.

[29] B. McCane, K. Novins, D. Crannitch, and B. Galvin. “On benchmarking optical flow”. Computer Vision and Image Understanding, Vol. 84, No 1, pp. 126–143, 2001.

[30] Microsoft Corporation. Media player 9 video quality demos. http://www.microsoft.com/windows/windowsmedia/demos/video quality demos.aspx.

[31] W. E. Milne, “Numerical solution of differential equations”, Dover, New York, 1953, 1979.

[32] H. Nagel and W. Enkelmann, “An investigation of smoothness constraints for the estimation of displacement vector fields from image sequences”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 8, Issue 5, pp. 565 – 593, 1986.

[33] C. C. Paige and M. A. Saunders, “Solution of sparse indefinite systems of linear equations”, SIAM Journal on Numerical Analysis, Vol. 12, pp. 617-629, 1975.

[34] X. Ren, “Local grouping for optical flow”. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp.1-8, 2008.

[35] R. D. Richtmyer and K.W. Mortin, “Difference methods for initial-value problems”, Interscience, John Wiley & Sons, New York, 1957, 1967.

[36] D. L. Russell, “Calculus of variations and control theory”, Academic Press, New York, 1976.

[37] S. Seitz and S. Baker. “Filter Flow”, International Conference on Computer Vision (ICCV), 2009.

[38] J. Shi and J. Malik, “Normalized cuts and image segmentation”, IEEE Transaction in Pattern Analysis and Machine Intelligence, vol. 22, No 8, pp. 888-905, 2000.

[39] J. Shi and C. Tomasi, “Good features to track”, IEEE Conference on Computer Vision and Pattern Recognition, (CVPR), pp. 593-600, Seattle, June 1994.

[40] J. Shi and C. Tomasi, “Good features to track”, TR 93-1399, Cornell U., 1993.

[41] M. Sonka, V. Hlavac and R. Boyle, “Image processing, analysis and machine vision”, Brooks, Cole, 1999.

[42] W. Trobin, T. Pock, D. Cremers, and H. Bischof. “Continuous energy minimization via repeated binary fusion”, European Conference on Computer Vision (ECCV), 2008.

[43] W. Yourgau, and S. Mandelstam, “Variational Principles in Dynamics and Quantum Theory”, Dover, New York, 1968, 1979.

[44] C. L. Zitnick, N. Jojic, and S. B. Kang, “Consistent segmentation for optical flow estimation”, In Proceedings of the International Conference on Computer Vision (ICCV), Vol. 2, pp. 1308–1315, 2005.

APPENDICES

APPENDIX A. Rate of Change of Image Brightness

Consider a patch of the brightness pattern that is displaced a distance δx in the x-direction and δy in the y-direction in time δt. The brightness of the patch is assumed to remain constant so that

$$\mathrm{I}(x, y, t) = \mathrm{I}(x+\delta x,\ y+\delta y,\ t+\delta t). \tag{A.1}$$

Expanding the right-hand side about the point (x, y, t) we get,

$$I(x,y,t) = I(x,y,t) + \delta x\frac{\partial I}{\partial x} + \delta y\frac{\partial I}{\partial y} + \delta t\frac{\partial I}{\partial t} + \varepsilon. \tag{A.2}$$

Where ε contains second and higher order terms in δx, δy, and δt. After subtracting $I(x, y, t)$ from both sides and dividing through by δt we have

$$\frac{\delta x}{\delta t}\frac{\partial I}{\partial x} + \frac{\delta y}{\delta t}\frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} + O(\delta t) = 0. \tag{A.3}$$

where $O(\delta t)$ is a term of order δt (we assume that δx and δy vary as δt). In the limit as $\delta t \rightarrow 0$ this becomes

$$\frac{\delta x}{\delta t}\frac{\partial I}{\partial x} + \frac{\delta y}{\delta t}\frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} = 0. \tag{A.4}$$

APPENDIX B. Student's *t*-distribution

In what follows, we briefly present the properties of Student's *t*-distributions.

A *d*-dimensional random variable *X* that follows a multivariate t-distribution with mean μ, positive definite, symmetric and real *d* x *d* covariance matrix Σ and has $\nu \in [0, \infty)$ degrees of freedom has a density expressed by

$$p(x;\mu,\Sigma,\nu) = \frac{\Gamma\left(\frac{\nu+d}{2}\right)\left|\Sigma\right|^{-\frac{1}{2}}}{(\pi\nu)^{\frac{d}{2}}\,\Gamma\left(\frac{\nu}{2}\right)\left[1+\nu^{-1}\delta(x,\mu;\Sigma)\right]^{\frac{\nu+d}{2}}}, \tag{B.1}$$

where $\delta(x,\mu;\Sigma) = (x-\mu)^T\Sigma^{-1}(x-\mu)$ is the Mahalanobis squared distance and Γ is the Gamma function.

It can be shown that the Student's *t*-distribution is equivalent to a Gaussian distribution with a stochastic covariance matrix. In other words, given a weight *u* following a Gamma distribution parameterized by *v*:

$$u \sim \Gamma(\nu/2, \nu/2), \tag{B.2}$$

The variable X has the multivariate normal distribution with mean μ and covariance Σ/u:

$$X \mid \mu,\Sigma,\nu,u \sim N(\mu,\Sigma/u), \tag{B.3}$$

It can be shown that for $\nu \to \infty$ the Student's *t*-distribution tends to a Gaussian distribution with covariance Σ. Also, if $\nu > 1$, μ is the mean of *X* and if $\nu > 2$, $\nu(\nu\text{-}2)^{-1}\Sigma$ is the covariance matrix of *X*. Therefore, the family of *t*-distributions

provides a heavy-tailed alternative to the normal family with mean μ and covariance matrix that is equal to a scalar multiple of Σ, if $\nu > 2$ (Fig. B.1).

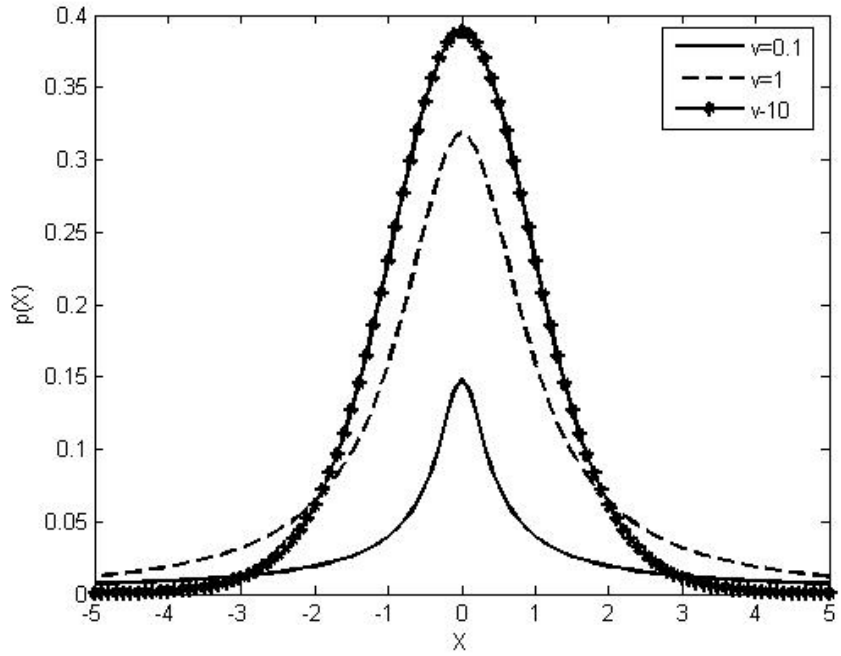


Figure B.1: A univariate Student's t-distribution ($\mu = 0$, $\sigma = 1$) for various Degrees of Freedom. As $\nu \to \infty$ the distribution tends to a Gaussian. For small values of ν the distribution has heavier tails than a Gaussian.

APPENDIX C. Calculus of Variations

C.1. Introduction

Variational methods have their origins in the 18th century with the work of Euler, Lagrange, and others on the *calculus of variations*. Standard calculus is concerned with finding derivatives of functions. We can think of a function as a mapping that takes the value of a variable as the input and returns the value of the function as the output. The derivative of the function then describes how the output value varies as we make infinitesimal changes to the input value. Similarly, we can define a *functional* as a mapping that takes a function as the input and that returns the value of the functional as the output. An example would be the entropy H[p], which takes a probability distribution $p(x)$ as the input and returns the quantity

$$\mathrm{H}[p] = \int p(x) \ln p(x) \mathrm{d}x, \tag{C.1}$$

as the output. We can introduce the concept of a *functional derivative*, which expresses how the value of the functional changes in response to infinitesimal changes to the input function (Feynman *et al.*, 1964). Many problems can be expressed in terms of an optimization problem in which the quantity being optimized is a functional. The solution is obtained by exploring all possible input functions to find the one that maximizes, or minimizes, the functional. Variational methods have broad applicability and include such areas as finite element methods (Kapur, 1989) and maximum entropy (Schwarz, 1988).

C.2. 1st Derivative in the Functional

For a given function $u(x)$: [a, b] $\rightarrow \Re$ and a functional $F(x, u, u')$ we define

$$E(u) = \int_a^b F(x, u, u') \ \mathrm{d}x, \tag{C.2}$$

and the problem is to minimize $E(u)$ with respect to $u(x)$.

Firstly, we have to define the first variation of $E(u)$, which is

$$\frac{\partial E}{\partial u} = E(u+v) - E(u) \quad \text{with } v(x) \text{ such that } v(a) = v(b) = 0 \ , \tag{C.3}$$

$$\text{therefore, } \ \frac{\partial E}{\partial u} = 0 \ \Leftrightarrow \ E(u+v) - E(u) = 0 \, , \tag{C.4}$$

Secondly, by using Taylor series expansion of $F\left(x, u+v, u'+v'\right)$ around the point (x, u, v), we get

$$F\left(x, u+v, u'+v'\right) = F\left(x, u, v\right) + v\frac{\partial F}{\partial u} + v'\frac{\partial F}{\partial u'}$$

$$\Leftrightarrow \int F\left(x, u+v, u'+v'\right)dx = \int\left[F\left(x, u, v\right) + v\frac{\partial F}{\partial u} + v'\frac{\partial F}{\partial u'} \right]dx$$

$$\Leftrightarrow E(u+v) = E(u) + \int\left[v\frac{\partial F}{\partial u} + v'\frac{\partial F}{\partial u'} \right]dx, \tag{C.5}$$

Then follows the minimization,

$$\begin{aligned}
\min_{u(x)}\left\{E(u)\right\} &\Leftrightarrow E(u+v) - E(u) = 0 \\
&\Leftrightarrow \int\left[v\frac{\partial F}{\partial u} + v'\frac{\partial F}{\partial u'} \right]dx = 0 \\
&\Leftrightarrow \int v\frac{\partial F}{\partial u}dx + \int v'\frac{\partial F}{\partial u'f}dx = 0 \\
&\Leftrightarrow \int v\frac{\partial F}{\partial u}dx + \left[v(x)\frac{\partial F}{\partial u'} \right]_{x=a}^{x=b} - \int v\frac{d}{dx}\frac{\partial F}{\partial u'}dx = 0 \\
&\overset{\text{v(a)=v(b)=0}}{\Leftrightarrow} \int v\frac{\partial F}{\partial u}dx = \int v\frac{d}{dx}\frac{\partial F}{\partial u'}dx \\
&\Leftrightarrow \int v\left[\frac{\partial F}{\partial u} - \frac{d}{dx}\frac{\partial F}{\partial u'} \right]dx = 0 \ , \ \forall v(x) \\
&\Leftrightarrow \frac{\partial F}{\partial u} - \frac{d}{dx}\frac{\partial F}{\partial u'} = 0
\end{aligned} \tag{C.6}$$

By solving the previous *Euler equation*, we obtain the solution $u(x)$ which minimizes the energy function $E(u)$.

The next point we have to stand is the case when we have second order derivatives in the functional.

C.3. 1st & 2nd Order Derivatives in the Functional

In this case the function $E(u)$ we want to minimize becomes

$$E(u) = \int_a^b F(x, u, u', u'')\ \mathrm{d}x \quad \Rightarrow \quad \min_{u(x)} \{E(u)\}\ , \tag{C.7}$$

While the differential equation becomes

$$\min_{u(x)} \{E(u)\} \Leftrightarrow \frac{\partial F}{\partial u} - \frac{d}{dx}\left(\frac{\partial F}{\partial u'}\right) + \frac{d^2}{dx^2}\left(\frac{\partial F}{\partial u''}\right) = 0\,, \tag{C.8}$$

C.4. Second order Partial Derivatives in the Functional and 2-D unknown Functions u(x, y)

For a given function $u(x, y)$: [a, b] x [c, d] $\rightarrow \Re$ and a functional $F(x, u, u', u'')$ the modified problem is

$$E(u) = \int_a^b \int_c^d F(x, u, u', u'')\ \mathrm{d}x\, \mathrm{d}y \quad \Rightarrow \quad \min_{u(x,\,y)} \{E(u)\} \tag{C.9}$$

Similarly we obtain the following differential equation,

$$\min_{u(x,y)} \{E(u)\} \ \Leftrightarrow \ \frac{\partial F}{\partial u} - \frac{d}{dx}\left(\frac{\partial F}{\partial u_x}\right) - \frac{d}{dy}\left(\frac{\partial F}{\partial u_y}\right) + \frac{d^2}{dx^2}\left(\frac{\partial F}{\partial u_{xx}}\right) + \frac{d^2}{dy^2}\left(\frac{\partial F}{\partial u_{yy}}\right) = 0\ , \tag{C.10}$$

where u_x , u_y , u_{xx} , u_{yy} are defined as the partial derivatives

$$u_x = \frac{\partial u}{\partial x}\ ,\ \ u_y = \frac{\partial u}{\partial y}\ \ ,\ u_{xx} = \frac{\partial^2 u}{\partial x^2}\ ,\ \ u_{yy} = \frac{\partial^2 u}{\partial y^2}\ .$$

APPENDIX D. Approximate Inference

In this appendix, we are going to quote some issues about the approximate inference which were taken from C. Bishop's book [8].

D.1. Variational Inference

Suppose we have a fully Bayesian model in which all parameters are given prior distributions. The model may also have latent variables as well as parameters, and we shall denote the set of all latent variables and parameters by **Z**. Similarly, we denote the set of all observed variables by **X**. For example, we might have a set of N independent, identically distributed data, for which $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_N\}$. Our probabilistic model specifies the joint distribution $p(\mathbf{X},\mathbf{Z})$, and our goal is to find an approximation for the posterior distribution $p(\mathbf{Z}|\mathbf{X})$ as well as for the model evidence $p(\mathbf{X})$. We can decompose the log marginal probability using

$$\ln p(\mathbf{X}) = L(q) + \mathrm{KL}(q \,\|\, p) \ , \tag{D.1}$$

where we have defined

$$L(q) = \int q(\mathbf{Z}) \ln\left\{ \frac{p(\mathbf{X},\mathbf{Z})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z} \ , \tag{D.2}$$

$$\mathrm{KL}(q \,\|\, p) = -\int q(\mathbf{Z}) \ln\left\{ \frac{p(\mathbf{Z}\,|\,\mathbf{X})}{q(\mathbf{Z})} \right\} \mathrm{d}\mathbf{Z} \ , \tag{D.3}$$

We can maximize the lower bound $L(q)$ by optimization with respect to the distribution $q(\mathbf{Z})$, which is equivalent to minimizing the KL divergence. If we allow any possible choice for $q(\mathbf{Z})$, then the maximum of the lower bound occurs when the KL divergence vanishes, which occurs when $q(\mathbf{Z})$ equals the posterior distribution $p(\mathbf{Z}|\mathbf{X})$. However, we shall suppose the model is such that working with the true posterior distribution is intractable.

We therefore consider instead a restricted family of distributions $q(\mathbf{Z})$ and then seek the member of this family for which the KL divergence is minimized. Our goal is to restrict the family sufficiently that they comprise only tractable distributions, while at the same time allowing the family to be sufficiently rich and flexible that it can provide a good approximation to the true posterior distribution. It is important to emphasize that the restriction is imposed purely to achieve tractability, and that subject to this requirement we should use as rich a family of approximating distributions as possible. In particular, there is no 'over-fitting' associated with highly flexible distributions. Using more flexible approximations simply allows us to approach the true posterior distribution more closely.

One way to restrict the family of approximating distributions is to use a parametric distribution $q(\mathbf{Z}|\boldsymbol{\omega})$ governed by a set of parameters $\boldsymbol{\omega}$. The lower bound $L(q)$ then becomes a function of $\boldsymbol{\omega}$, and we can exploit standard nonlinear optimization techniques to determine the optimal values for the parameters.

D.1.1. Factorized distributions

Here we consider an alternative way in which to restrict the family of distributions $q(\mathbf{Z})$. Suppose we partition the elements of $\mathbf{Z}$ into disjoint groups that we denote by $\mathbf{Z}_i$ where $i = 1, \ldots, M$. We then assume that the q distribution factorizes with respect to these groups, so that

$$q(\mathbf{Z}) = \prod_{i=1}^{M} q_i(\mathbf{Z}_i) \ , \tag{D.4}$$

It should be emphasized that we are making no further assumptions about the distribution. In particular, we place no restriction on the functional forms of the individual factors $q_i(\mathbf{Z}_i)$. This factorized form of variational inference corresponds to an approximation framework developed in physics called *mean field theory*. Amongst all distributions $q(\mathbf{Z})$ having the form (D.4), we now seek that distribution for which

the lower bound $L(q)$ is largest. We therefore wish to make a free form (variational) optimization of $L(q)$ with respect to all of the distributions $q_i(\mathbf{Z}_i)$, which we do by optimizing with respect to each of the factors in turn. To achieve this, we first substitute (D.4) into (D.2) and then dissect out the dependence on one of the factors $q_j(\mathbf{Z}_j)$. Denoting $q_j(\mathbf{Z}_j)$ by simply q_j to keep the notation uncluttered, we then obtain

$$\begin{aligned} L(q) &= \int \prod_i q_i \left\{ \ln\, p(\mathbf{X},\mathbf{Z}) - \sum_i \ln q_i \right\} \mathrm{d}\mathbf{Z} \\ &= q_j \int \left\{ \ln\, p(\mathbf{X},\mathbf{Z}) - \prod_{i\neq j} q_i \mathbf{Z}_i \right\} \mathrm{d}\mathbf{Z}_j - \int q_j \ln q_j d\mathbf{Z}_j + const \\ &= \int q_j \ln \tilde{p}(\mathbf{X},\mathbf{Z}_j) d\mathbf{Z}_j - \int q_j \ln q_j d\mathbf{Z}_j + const \ , \end{aligned} \tag{D.5}$$

where we have defined a new distribution $\tilde{\mathrm{p}}(\mathbf{X},\mathbf{Z}_\mathrm{j})$ by the relation

$$\ln \tilde{p}(\mathbf{X},\mathbf{Z}_j) = \mathbf{E}_{\mathrm{i}\neq\mathrm{j}}\left[\ln\, p(\mathbf{X},\mathbf{Z})\right] + const \ , \tag{D.6}$$

Here the notation $\mathbf{E}_{i\neq j}[...]$ denotes an expectation with respect to the q distributions over all variables $\mathbf{z}_i$ for $i \neq j$, so that

$$\mathbf{E}_{\mathrm{i}\neq\mathrm{j}}\left[\ln\, p(\mathbf{X},\mathbf{Z})\right] = \int \ln p(\mathbf{X},\mathbf{Z}) \prod_{\mathrm{i}\neq\mathrm{j}} \mathrm{q}_i d\mathbf{Z}_i \ , \tag{D.7}$$

Now suppose we keep the $\{q_{i\neq j}\}$ fixed and maximize $L(q)$ in (D.5) with respect to all possible forms for the distribution $qj(\mathbf{Z}j)$. This is easily done by recognizing that (D.5) is a negative Kullback-Leibler divergence between $q_j(\mathbf{Z}_j)$ and $\tilde{p}(\mathbf{X},\mathbf{Z}_\mathrm{j})$. Thus maximizing (D.5) is equivalent to minimizing the Kullback-Leibler divergence, and the minimum occurs when $q_\mathrm{j}(\mathbf{Z}_\mathrm{j}) = \tilde{p}(\mathbf{X},\mathbf{Z}_\mathrm{j})$. Thus we obtain a general expression for the optimal solution $q_j^*(\mathbf{Z}_\mathrm{j})$ given by

$$\ln q_j^*(\mathbf{Z}_\mathrm{j}) = \mathbf{E}_{\mathrm{i}\neq\mathrm{j}}\left[\ln\, p(\mathbf{X},\mathbf{Z})\right] + const \ , \tag{D.8}$$

It is worth taking a few moments to study the form of this solution as it provides the basis for applications of variational methods. It says that the log of the optimal solution for factor q_j is obtained simply by considering the log of the joint distribution over all hidden and visible variables and then taking the expectation with respect to all of the other factors $\{q_i\}$ for $i \neq j$.

The additive constant in (D.8) is set by normalizing the distribution $q_j^*(\mathbf{Z}_j)$. Thus if we take the exponential of both sides and normalize, we have

$$q_j^*(\mathbf{Z}_j) = \frac{\exp\left(\mathbf{E}_{i\neq j}[\ln p(\mathbf{X},\mathbf{Z})]\right)}{\int \mathbf{E}_{i\neq j}[\ln p(\mathbf{X},\mathbf{Z})]d\mathbf{Z}_j}$$

In practice, we shall find it more convenient to work with the form (D.8) and then reinstate the normalization constant (where required) by inspection.

The set of equations given by (D.8) for $j = 1, \ldots, M$ represent a set of consistency conditions for the maximum of the lower bound subject to the factorization constraint. However, they do not represent an explicit solution because the expression on the right-hand side of (D.8) for the optimum $q_j^*(\mathbf{Z}_j)$ depends on expectations computed with respect to the other factors $q_i(\mathbf{Z}_i)$ for $i \neq j$. We will therefore seek a consistent solution by first initializing all of the factors $q_i(\mathbf{Z}_i)$ appropriately and then cycling through the factors and replacing each in turn with a revised estimate given by the right-hand side of (D.8) evaluated using the current estimates for all of the other factors. Convergence is guaranteed because bound is convex with respect to each of the factors $q_i(\mathbf{Z}_i)$.

APPENDIX E. Additional Numerical Experimental Results from Chapter 3

Table E.1: Various combinations between the number of super-pixels and the window size for the Yosemite sequence, with and without clouds.

Number of super-pixels	**Window size**	*Yosemite without clouds*			*Yosemite with clouds*		
		AAE	**AeM**	**AEP**	**AAE**	**AeM**	**AEP**
40	5 x 5	5.76	0.13	0.25	13.81	0.40	0.70
70	5 x 5	5.80	0.13	0.25	13.88	0.40	0.71
100	5 x 5	5.80	0.13	0.25	13.88	0.40	0.71
200	5 x 5	5.89	0.13	0.26	13.98	0.40	0.71
1000	5 x 5	6.07	0.14	0.27	14.32	0.41	0.73
40	7 x 7	5.06	0.12	0.22	13.25	0.38	0.66
70	7 x 7	5.12	0.12	0.22	13.34	0.38	0.67
100	7 x 7	5.18	0.12	0.22	13.34	0.38	0.67
200	7 x 7	5.30	0.12	0.23	13.52	0.38	0.68
40	9 x 9	4.62	0.11	0.20	12.87	0.36	0.62
70	9 x 9	4.69	0.11	0.20	12.95	0.36	0.63
100	9 x 9	4.76	0.11	0.20	12.98	0.36	0.63
200	9 x 9	4.89	0.11	0.21	13.17	0.37	0.64
40	11 x 11	4.28	0.10	0.18	12.59	0.34	0.59
70	11 x 11	4.38	0.10	0.19	12.67	0.34	0.60
100	11 x 11	4.45	0.10	0.19	12.69	0.34	0.60
40	13 x 13	4.06	0.10	0.17	12.34	0.33	0.57
70	13 x 13	4.14	0.10	0.18	12.42	0.33	0.57
40	15 x 15	3.90	0.10	0.16	12.16	0.32	0.55
70	15 x 15	3.96	0.09	0.17	12.24	0.32	0.56
40	17 x 17	3.80	0.09	0.15	12.01	0.31	0.54
70	17 x 17	3.87	0.09	0.16	12.10	0.32	0.54
40	19 x 19	**3.79**	**0.09**	**0.15**	11.90	0.31	0.53
70	19 x 19	3.85	0.09	0.16	12.01	0.31	0.54
40	21 x 21	3.89	0.09	0.16	**11.86**	**0.30**	**0.52**
70	21 x 21	3.90	0.09	0.16	11.99	0.31	0.53
40	23 x 23	4.06	0.09	0.16	11.91	0.30	0.52
70	23 x 23	4.04	0.09	0.16	12.06	0.31	0.53
40	25 x 25	4.31	0.10	0.17	12.04	0.30	0.52
70	25 x 25	4.26	0.09	0.17	12.21	0.31	0.53
40	27 x 27	4.58	0.10	0.18	12.26	0.30	0.53
70	27 x 27	4.51	0.10	0.18	12.44	0.31	0.54
40	29 x 29	4.84	0.11	0.19	12.57	0.31	0.54
70	29 x 29	4.79	0.10	0.19	12.75	0.31	0.55
40	31 x 31	5.11	0.11	0.20	12.92	0.31	0.55
70	31 x 31	5.05	0.11	0.20	13.11	0.32	0.56

Table E.2: Various combinations between the number of super-pixels and the window size for the Dimetrodon sequence and the Rubberwhale sequence.

Number of super-pixels	**Window size**	*Dimetrodon Sequence*			*Rubberwhale Sequence*		
		AAE	**AeM**	**AEP**	**AAE**	**AeM**	**AEP**
40	5 x 5	11.31	0.26	0.57	8.45	0.22	0.26
70	5 x 5	11.34	0.26	0.56	8.44	0.22	0.26
100	5 x 5	11.36	0.26	0.56	8.46	0.22	0.26
200	5 x 5	11.46	0.26	0.56	8.50	0.22	0.26
1000	5 x 5	11.78	0.27	0.57	8.52	0.22	0.26
40	7 x 7	10.04	0.24	0.51	8.24	0.21	0.25
70	7 x 7	10.09	0.24	0.52	8.22	0.21	0.25
100	7 x 7	10.14	0.24	0.52	8.22	0.21	0.25
200	7 x 7	10.26	0.25	0.52	8.27	0.21	0.25
40	9 x 9	9.18	0.23	0.48	8.22	0.21	0.24
70	9 x 9	9.23	0.23	0.48	8.19	0.21	0.24
100	9 x 9	9.28	0.23	0.49	**8.17**	**0.21**	**0.24**
200	9 x 9	9.42	0.23	0.49	8.20	0.21	0.24
40	11 x 11	8.55	0.22	0.46	8.36	0.21	0.24
70	11 x 11	8.60	0.22	0.46	8.30	0.21	0.24
100	11 x 11	8.65	0.22	0.46	8.26	0.21	0.24
40	13 x 13	8.07	0.21	0.44	8.57	0.22	0.25
70	13 x 13	8.12	0.21	0.44	8.49	0.22	0.25
40	15 x 15	7.68	0.20	0.42	8.81	0.22	0.25
70	15 x 15	7.75	0.20	0.43	8.70	0.22	0.25
40	17 x 17	7.34	0.20	0.41	9.07	0.23	0.26
70	17 x 17	7.43	0.20	0.41	8.94	0.23	0.25
40	19 x 19	7.06	0.19	0.40	9.35	0.24	0.26
70	19 x 19	7.17	0.19	0.40	9.22	0.23	0.26
40	21 x 21	6.82	0.19	0.39	9.65	0.24	0.27
70	21 x 21	6.95	0.19	0.39	9.51	0.24	0.27
40	23 x 23	6.61	0.19	0.38	9.96	0.25	0.28
70	23 x 23	6.77	0.19	0.38	9.81	0.24	0.27
40	25 x 25	6.43	0.18	0.37	10.27	0.26	0.29
70	25 x 25	6.61	0.19	0.38	10.12	0.25	0.28
40	27 x 27	6.29	0.18	0.36	10.56	0.26	0.29
70	27 x 27	6.47	0.18	0.37	10.42	0.26	0.29
40	29 x 29	**6.24**	**0.18**	**0.36**	10.84	0.27	0.30
70	29 x 29	6.42	0.18	0.37	10.71	0.26	0.29
40	31 x 31	6.26	0.18	0.36	11.13	0.28	0.31
70	31 x 31	6.43	0.18	0.36	11.03	0.27	0.30

SHORT CV

Theodosios Gkamas was born in Ioannina in 1985. He was admitted at the Computer Science Department of the University of Ioannina in 2003. He received his BSc degree in computer science in 2008 with degree 7.32 “very good”. That year he became a postgraduate student at the same institution from where he graduated in October 2010. His main research interests lie in the field of Computer Vision, Image Processing, and Machine Learning with specific interests in Optical Flow and Image Registration.