OPTIMIZATION OF VARIATIONAL METHODS VIA MOTION-BASED WEIGHT SELECTION AND KEYPOINT MATCHING

Botao Wang, Qingxiang Zhu, Hongkai Xiong

Department of Electronic Engineering
Shanghai Jiao Tong University
Shanghai, 200240, P. R. China

ABSTRACT

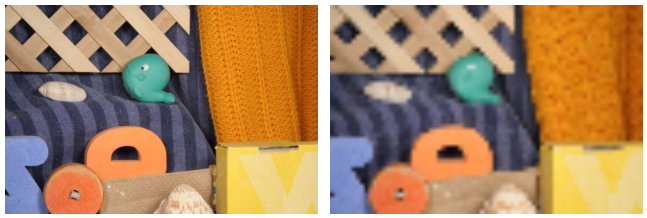
Variational method is a well-established technique that solves for a dense field, which is widely adopted in the estimation of optical flow field and remains the most accurate technique to date. However, one of the problems in variational method lies in that it is optimized in an iterative manner towards a single objective, but local details may be compromised owing to the "big picture". In this paper, we address this problem in an optical flow framework by introducing two sparse local rectifications to the global numerical scheme, i.e., motion-based weight selection and keypoint matching. The selection of the weighting parameter in a self-adaptive and content-aware manner provides a more accurate estimation of the optical flow field near motion boundaries, and motion details and small structures are preserved in the optical flow field by keypoint matching in the initialization of the optical flow field. Experimental results using the Middlebury dataset show that the proposed algorithm achieves higher accuracy compared to the original TV-ℓ^1 optical flow algorithm and many state-of-the-art methods.

Index Terms— Variational method, optical flow, total variation, weight selection, keypoint matching

1. INTRODUCTION

Variational methods are widely used in computer vision applications, especially optical flow field estimation. Variational methods solve for a dense field by minimizing an energy functional which consists of a data term, which penalizes the data fidelity, and a regularization term, which enforces smoothness constraints to the field. This energy functional is transformed into a series of convex minimization problems after introducing some auxiliary variables and solved in an iterative numerical scheme. However, as the functional is targeted for a single objective, local details may be lost. In this paper, we will illustrate this phenomenon of variational methods in the optical flow framework, and propose two novel local constraints to address this problem.

Optical flow is to estimate the displacement field $\boldsymbol{u}$ of two images I_1 and I_2. Dense optical field estimation has many practical applications, such as video coding, 3D scene reconstruction and disparity map computation. Variational methods for optical flow, proposed by Horn et al. [1], yield highly accurate results compared to other contemporary optical flow algorithms. Zach et al. proposed an optical flow algorithm based on total variation data term and the ℓ^1 norm regularization term in [2] and Wedel et al. improved it in [3]. The advantage of TV-ℓ^1 optical flow is to preserve discontinuities in the optical flow field and offer an increased robustness against illumination changes, occlusion, and noise. To get large displacement, TV-ℓ^1 optical flow algorithm is embeded into a scale-space approach in which the optical flow field is computed in an iterative coarse-to-fine warping procedure. However, the most severe problem of the scale-space approach is the inclination to diminish small motion structures when spatially significant and abrupt change of the displacement exists [4]. In Fig. 1, an example is given to compare the details of images at different scales. We can see that some detailed structures in the coarse scale are blurred or even disappeared, which is hard to recover in finer scales because the optical flow field of the current scale fully depends on the coarser scales.



(a) (b)

Fig. 1. Image details are lost in coarse scales. (a) Original image; (b) 1/8 scale of original image

There are two contributions of this paper. The first contribution is to propose a weight selection algorithm based on motion boundary estimation. In the majority of the variational optical flow literatures, the weight parameter of the data term and regularization term is a constant, which implies the same degree of smoothness in every location for the optical flow field. However, in reality, the motion variation may be

large near object boundaries if the objects have different motion directions or velocities, or it can be small in textureless regions which have consistent motions. The proposed method generates a weight matrix that assigns large weight values to pixels in motion boundaries to get better data fidelity and small weight values to pixels in textureless regions to enforce smoother optical flow field.

The second contribution of this paper is to propose a keypoint matching algorithm for the initialization of the optical flow field to recover small structures, like corners. Original scale space approaches initialize the optical flow field of the current level of the image pyramid by resizing the optical flow field generated from the coarser level and scaling it by the scale factor. In our approach, we refine the initial optical flow field propagated from the coarser level with a sparse set of high-confidence optical flow vectors generated by keypoint matching, thus the initial optical flow field is refined with image details of the current level, which do not exist or are not clear in coarser levels.

The rest of the paper is organized as follows: Section 2 introduces the overall framework of the proposed algorithm; Section 3 describes the proposed motion-based weight selection algorithm; Section 4 describes the proposed keypoint matching algorithm tailored for optical flow; Section 5 shows the experimental results of the proposed algorithm and the comparison with the original TV-ℓ^1 optical flow algorithm and other optical flow algorithms; Section 6 summarizes the paper.

2. SYSTEM OVERVIEW

The proposed algorithm is based on the framework of the total variation ℓ^1 norm (TV-ℓ^1) optical flow proposed by Zach et al. [2] [3]. Given two images $I_1(\boldsymbol{x}), I_2(\boldsymbol{x}) : \boldsymbol{x} \in \Omega$, where $\boldsymbol{x} = (x, y)$ is the two dimensional coordinate and Ω is the set of all possible positions of the image. The optical flow field $\boldsymbol{u}(\boldsymbol{x}) = (\boldsymbol{u}_x(\boldsymbol{x}), \boldsymbol{u}_y(\boldsymbol{x}))$ of the original TV-ℓ^1 optical flow is the minimizer of the following objective function:

$$\min_{\boldsymbol{u}} \sum_{\boldsymbol{x}} |\nabla \boldsymbol{u}(\boldsymbol{x})| + \lambda \left| I_1(\boldsymbol{x}) - I_2(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) \right| \quad (1)$$

where $|I_1(\boldsymbol{x}) - I_2(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x}))|$ is the ℓ^1 norm data term that penalizes the fidelity of corresponding pixels in I_1 and I_2, $|\nabla \boldsymbol{u}(\boldsymbol{x})|$ is the total variation regularization term that imposes smoothness constraint to the optical flow field, and λ is a constant that balances two terms.

As opposed to the traditional approach, we replace the constant weight parameter λ in Eq. (1) with a matrix $\lambda(\boldsymbol{x})$, which is determined based on motion boundary estimation. The objective function of the proposed optical flow algorithm is therefore

$$\min_{\boldsymbol{u}} \sum_{\boldsymbol{x}} |\nabla \boldsymbol{u}(\boldsymbol{x})| + \lambda(\boldsymbol{x}) \left| I_1(\boldsymbol{x}) - I_2(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x})) \right| \quad (2)$$

We will elaborate details for the computation for $\lambda(\boldsymbol{x})$ in Section 3. Note that once $\lambda(\boldsymbol{x})$ is determined, the solution of Eq. (2) can be directly derived from the solution of Eq. (1).

We linearize $I_2(\boldsymbol{x} + \boldsymbol{u}(\boldsymbol{x}))$ near $I_2(\boldsymbol{x} + \boldsymbol{u}_0(\boldsymbol{x}))$. Here $\boldsymbol{u}_0(\boldsymbol{x})$ is a given optical flow field:

$$I_2(\boldsymbol{x} + \boldsymbol{u}) = I_2(\boldsymbol{x} + \boldsymbol{u}_0) + \langle \nabla I_2(\boldsymbol{x} + \boldsymbol{u}_0), \boldsymbol{u} - \boldsymbol{u}_0 \rangle \quad (3)$$

where we simplify $\boldsymbol{u}(\boldsymbol{x})$ and $\boldsymbol{u}_0(\boldsymbol{x})$ as $\boldsymbol{u}$ and $\boldsymbol{u}_0$. The residual in the data term is denoted as $\rho(\boldsymbol{u}) = I_1(\boldsymbol{x}) - I_2(\boldsymbol{x} + \boldsymbol{u}) = I_2(\boldsymbol{x} + \boldsymbol{u}_0) - I_1(\boldsymbol{x}) + \langle \nabla I_2(\boldsymbol{x} + \boldsymbol{u}_0), \boldsymbol{u} - \boldsymbol{u}_0 \rangle$. Subsequently, Eq. (2) is transformed into the equivalent form

$$\min_{\boldsymbol{u}} \sum_{\boldsymbol{x}} \sum_{d=x,y} |\nabla \boldsymbol{u}_d| + \sum_{d=x,y} \frac{(\boldsymbol{u}_d - \boldsymbol{v}_d)^2}{2\theta} + \lambda(\boldsymbol{x}) \left| \rho(\boldsymbol{v}) \right| \quad (4)$$

An auxiliary variable $\boldsymbol{v}(\boldsymbol{x}) = (\boldsymbol{v}_x(\boldsymbol{x}), \boldsymbol{v}_y(\boldsymbol{x}))$ is introduced as a close approximate of $\boldsymbol{u}$, and θ is a small constant.

Eq. (4) can be solved by alternating steps updating $\boldsymbol{u}$ or $\boldsymbol{v}$ during every iteration in the following manner.

1. For $\boldsymbol{v}$ being fixed, we solve

$$\min_{\boldsymbol{u}_d} \sum_{\boldsymbol{x}} |\nabla \boldsymbol{u}_d| + \frac{1}{2\theta} (\boldsymbol{u}_d - \boldsymbol{v}_d)^2 \quad (5)$$

   where $d = x, y$

2. For $\boldsymbol{u}$ being fixed, we solve

$$\min_{\boldsymbol{v}} \sum_{\boldsymbol{x}} \frac{(\boldsymbol{u}_x - \boldsymbol{v}_x)^2 + (\boldsymbol{u}_y - \boldsymbol{v}_y)^2}{2\theta} + \lambda(\boldsymbol{x}) \left| \rho(\boldsymbol{v}) \right| \quad (6)$$

Eq. (5) is a Rudin-Osher-Fatemi total variation problem [5]. An efficient solution is based on gradient descent and subsequent reprojection using the dual-ROF model [6]. A dual variable $\boldsymbol{p} = (p_1, p_2)$ is introduced for each element of $\boldsymbol{u}$. The solution of Eq. (5) is given by

$$\boldsymbol{u}_d = \boldsymbol{v}_d + \mathbf{div} \boldsymbol{p}_d \quad (7)$$

where **div** is the discrete divergence and $\boldsymbol{p}_d$ can be obtained by the iterative fixed point scheme:

$$\widetilde{\boldsymbol{p}_d}^{(k+1)} = \boldsymbol{p}_d^{(k)} + \frac{\tau}{\theta} \nabla \boldsymbol{u}_d \quad (8)$$

$$\boldsymbol{p}_d^{(k+1)} = \frac{\widetilde{\boldsymbol{p}_d}^{(k+1)}}{\max\{ \left| \widetilde{\boldsymbol{p}_d}^{(k+1)} \right|, 1\}} \quad (9)$$

where k denotes the number of iterations and $\tau \leq 1/4$.

The solution of Eq. (6) is given by the thresholding step:

$$\boldsymbol{v} = \boldsymbol{u} + \begin{cases} \lambda(\boldsymbol{x}) \theta \nabla I_2, & \text{if } \rho(\boldsymbol{u}) < -\lambda(\boldsymbol{x}) \theta \left| \nabla I_2 \right|^2 \\ -\lambda(\boldsymbol{x}) \theta \nabla I_2, & \text{if } \rho(\boldsymbol{u}) > \lambda(\boldsymbol{x}) \theta \left| \nabla I_2 \right|^2 \\ -\dfrac{\rho(\boldsymbol{u}) \nabla I_2}{\left| \nabla I_2 \right|^2}, & \text{if } |\rho(\boldsymbol{u})| \leq \lambda(\boldsymbol{x}) \theta \left| \nabla I_2 \right|^2 \end{cases} \quad (10)$$

As the linearization of $I_2(\boldsymbol{x}+\boldsymbol{u})$ near $I_2(\boldsymbol{x}+\boldsymbol{u}_0)$ is only valid for a small displacement, to obtain a large displacement, the numerical scheme is embeded into a scale-space approach. At the coarsest level of the image pyramid, $\boldsymbol{u}$, $\boldsymbol{p}_x$ and $\boldsymbol{p}_y$ are initialized to be $\mathbf{0}$. At each level of the image pyramid, we solve $\boldsymbol{u}$, $\boldsymbol{p}_x$ and $\boldsymbol{p}_y$ with the iterative numerical scheme and propagate the solutions to the next finer level.

The diagram of the proposed algorithm is displayed in Algorithm 1.

Algorithm 1 Proposed optical flow algorithm

Input: I_1, I_2
Output: $\boldsymbol{u}$

Initialization: $\boldsymbol{u}=\mathbf{0}$, $\boldsymbol{p}_x=\mathbf{0}$, $\boldsymbol{p}_y=\mathbf{0}$
for l = *LEVELS* - 1 to 0 **do**
  $\left[I_1^{(l)}, I_2^{(l)}\right]$ = Resize(I_1, I_2, $SCALE\_FACTOR^l$);
  $\boldsymbol{u}$ = KeypointMatching($I_1^{(l)}$, $I_2^{(l)}$, $\boldsymbol{u}$);
  for $w=1$ to *WARPS* **do**
    $I_2^{warp}(\boldsymbol{x}) = I_2^{(l)}(\boldsymbol{x}+\boldsymbol{u}(\boldsymbol{x}))$;
    $\lambda(\boldsymbol{x})$ = WeightSelection($\boldsymbol{I}_1^{(l)}$, $\boldsymbol{u}$);
    for $i=1$ to *OUTER_ITERATIONS* **do**
      $\boldsymbol{v}$ = ThresholdingStep($\boldsymbol{u}$, $\lambda(\boldsymbol{x})$, I_2^{warp}, $I_1^{(l)}$);
      for $j=1$ to *INNER_ITERATIONS* **do**
        $\boldsymbol{u}$ = UpdateU($\boldsymbol{v}$, $\boldsymbol{u}$, $\boldsymbol{p}$);
        $\boldsymbol{p}$ = UpdateP($\boldsymbol{p}$, $\boldsymbol{u}$);
      end for
    end for
  end for
  [$\boldsymbol{u}$, $\boldsymbol{p}_x$, $\boldsymbol{p}_y$] = Resize($\boldsymbol{u}$, $\boldsymbol{p}_x$, $\boldsymbol{p}_y$);
  $\boldsymbol{u}$ = Scale($\boldsymbol{u}$);
end for

3. MOTION-BASED WEIGHT SELECTION

In original TV-ℓ^1 optical flow algorithm [2] [3], $\lambda(\boldsymbol{x})$ is a constant for each pixel of the image, which implies that we expect the same degree of smoothness in every location of the optical flow field. However, in practice, the optical flow field usually appears non-smooth near motion boundaries and smooth in textureless regions, which is shown in Fig. 2(a). We propose a motion-based weight selection algorithm to generate the weight matrix $\lambda(\boldsymbol{x})$. The basic idea is to adaptively assign the weights based on the regions to enforce smoother optical flow field. Similar approach appears in [4], but they selected the weight parameter based on the pixel intensity $\lambda(\boldsymbol{x}) = \exp\left(-\|\nabla I_1\|^k\right)$, as illustrated in Fig. 2(b). The main disadvantage of their approach is that motion boundaries are only a subset of edges, and it may decrease the performance to assign large $\lambda(\boldsymbol{x})$ to pixels in edges which are not motion boundaries.

(a) (b) (c)

Fig. 2. (a) Ground truth motion boundary; (b) Image gradient; (c) Estimated motion boundary

In our approach, we estimate the motion boundaries based on the motion gradient calculated from the coarse initial optical flow field magnitude and edges of I_1. Our approach generates an approximate of the ground truth motion boundary. Visually inspecting, it looks more reasonable than the outputs using the algorithm based on [4], which is shown in Fig. 2(c). The diagram of our approach is displayed in Fig. 3.

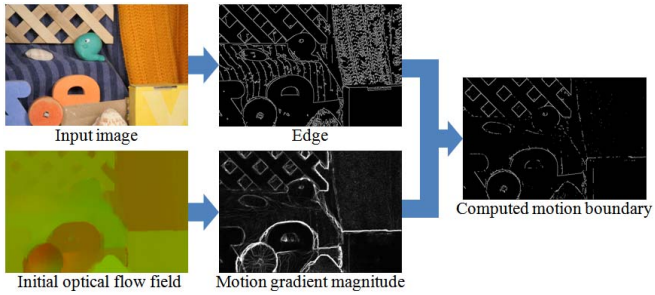


Fig. 3. The proposed method estimates the motion boundaries based on the edges of the image and the motion gradient magnitude of the coarse optical flow field

Given an optical flow field $\boldsymbol{u}$, the motion gradient magnitude is defined as:

$$M_m(\boldsymbol{x}) = \sqrt{\left|\frac{\partial \boldsymbol{u}(\boldsymbol{x})}{\partial x}\right|^2 + \left|\frac{\partial \boldsymbol{u}(\boldsymbol{x})}{\partial y}\right|^2}. \tag{11}$$

The motion gradient magnitude is calculated from the initial optical flow field, which is propagated from the coarser level of the image pyramid, and it is usually blurred. We refine the motion gradient magnitude with edges that accurately coincide with the true motion boundaries. Edges are extracted by the Canny edge detector [7]:

$$E(\boldsymbol{x}) = \begin{cases} 1, & \text{if } \boldsymbol{x} \text{ is located at edges} \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

The estimated weight matrix $\lambda(\boldsymbol{x})$ is obtained by merging the motion gradient magnitude $M_m(\boldsymbol{x})$ and the edge $E(\boldsymbol{x})$:

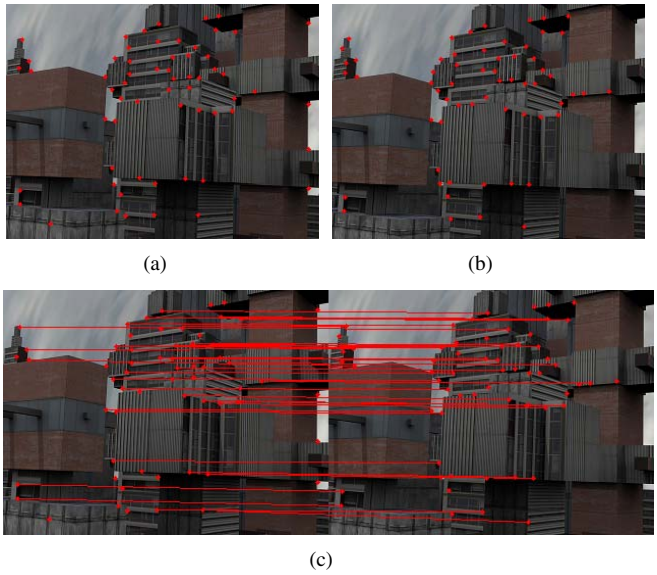
$$\lambda(\boldsymbol{x}) = \begin{cases} \lambda_b, & \text{if } M_m(\boldsymbol{x}) > s \text{ and } E(\boldsymbol{x}) = 1 \\ \lambda_s, & \text{otherwise} \end{cases} \tag{13}$$

where $\lambda_b > \lambda_s > 0$ and s is the motion sensitivity that depicts the sensitivity to different motions. In the experiments, we fix $s = 2$.

4. KEYPOINT MATCHING

There are three main differences between conventional keypoint matching methods for object detection and the proposed keypoint matching for optical flow field refinement. First, the proposed keypoint matching method extracts keypoints only from the original scale of the image while conventional methods extract keypoints from the scale space of the image. Second, geometric constraint is imposed in the proposed keypoint matching method based on the fact that two matched keypoints must be located in nearby positions. Third, many keypoint descriptors, such as SIFT[8], are rotation-invariant, which is unnecessary for the proposed keypoint matching method because two corresponding keypoints in the image pair only have very small degree of rotation.

Given two images $I_1^{(l)}$ and $I_2^{(l)}$ in the same level of the image pyramid, keypoints are extracted at a sparse set of locations by the Harris corner detector [9]. Harris corners are fast to compute, compared with DoG operator [9] and other keypoint detectors, and they are sensitive to small structures which have corners. A demonstration of keypoints extraction is shown in Fig. 4(a) and Fig. 4(b).



(a) (b)

(c)

Fig. 4. Keypoint matching for the *Urban3* sequence in Middlebury dataset. (a) Keypoints in frame 10; (b) Keypoints in frame 11; (c) Matched keypoints

The descriptor of a keypoint is computed from the 16×16 local image patch centered at the keypoint. The image patch is divided into 4×4 non-overlapping cells, and in each cell a local histogram of oriented gradients is calculated. The gradient orientation is divided into 8 bins evenly spaced between $[0°, 360°)$. Each pixel casts a weighted vote for the local histogram of oriented gradient of the cell it resides. The gradient magnitude is used as the voting weight. Hence, each keypoint generates a $4 \times 4 \times 8 = 128$ dimensional descriptor after normalization $|\boldsymbol{F}(i,j)| = 1$.

The set of keypoints extracted from I_1 is denoted as $KP_1 = \{p_i^1\}$, $i = 1 \ldots n_1$ and the set of keypoints extracted from I_2 is denoted as $KP_2 = \{p_i^2\}$, $i = 1 \ldots n_2$. Each keypoint is represented as $p = (\boldsymbol{x}, \boldsymbol{F})$, where $\boldsymbol{x}$ is the 2 dimensional vector denoting the location of the keypoint and $\boldsymbol{F}$ is the 128 dimensional descriptor.

For each keypoint p_i^1 in KP_1, the best matched keypoint in KP_2 is found by solving

$$\begin{aligned} p_j^2 = \underset{j}{\text{argmin}} \ & \left|\boldsymbol{F}_i^1 - \boldsymbol{F}_j^2\right| \\ \text{s.t.} \ & \left|\boldsymbol{x}_i^1 - \boldsymbol{x}_j^2\right| \leq R \\ & \left|\boldsymbol{F}_i^1 - \boldsymbol{F}_j^2\right| < C \end{aligned} \tag{14}$$

where R is the radius of the neighborhood and C is the minimum matching cost. That is, the best matched keypoint for p_i^1 in KP_2 is the one in its neighborhood of radius R with the lowest matching cost. In our experiments, we set $R = 10$ pixels and $C = 0.1$. Finally, for each keypoint in KP_1, we can find either one best matched keypoint in KP_2 or report no matched keypoint because there is no keypoints in its neighborhood or the lowest matching cost is still larger than C.

The matching vector for KP_1 is denoted as $\mathcal{M}_1$. $\mathcal{M}_1(i) = j$ if the best matched keypoint for p_i^1 is p_j^2 and $\mathcal{M}_1(i) = 0$ if there is no matched keypoint for p_i^1. To improve the accuracy and robustness of the final matching results, we use bidirectional matching strategy. p_i^1 and p_j^2 matches if and only if $\mathcal{M}_1(i) = j$ and $\mathcal{M}_2(j) = i$, namely, the best matched keypoint for p_i^1 in KP_2 is p_j^2 and the best matched keypoint for p_j^2 in KP_1 is p_i^1. The pseudocode of keypoint matching is shown in Algorithm 2.

Algorithm 2 Keypoint matching

Input: I_1, I_2
Output: $\mathcal{M}_1$

$\{\boldsymbol{x}_i^1\}_{i=1,\ldots n_1}$ = HarrisCornerDetector(I_1);
$\{\boldsymbol{x}_j^2\}_{j=1,\ldots n_2}$ = HarrisCornerDetector(I_2);
$\{\boldsymbol{F}_i^1\}$ = Descriptor$(\{\boldsymbol{x}_i^1\}, I_1)$;
$\{\boldsymbol{F}_j^2\}$ = Descriptor$(\{\boldsymbol{x}_j^2\}, I_2)$;
for $j = 1$ to n_2 **do**
  $\mathcal{M}_2(j) = \text{Matching}(p_j^2, \{p_i^1\})$;
end for
for $i = 1$ to n_1 **do**
  $\mathcal{M}_1(i) = \text{Matching}(p_i^1, \{p_j^2\})$;
  if $\mathcal{M}_2(\mathcal{M}_1(i)) \neq i$ **then**
    $\mathcal{M}_1(i) = 0$;
  end if
end for

After keypoint matching, a sparse set of high-confidence optical flow vectors are obtained:

$$\boldsymbol{u}_M(\boldsymbol{x}_i^1) = \boldsymbol{x}_{\mathcal{M}(i)}^2 - \boldsymbol{x}_i^1 \tag{15}$$

where the subscript M in $\boldsymbol{u}_M$ indicates that the optical flow vector is obtained through keypoint matching, $\boldsymbol{x}^2_{\mathcal{M}(i)}$ is the matched keypoint in KP_2 with the i-th keypoint in KP_1 and their difference is the displacement or the optical flow vector. At each level of the image pyramid, we replace the corresponding optical flow vectors in $\boldsymbol{u}$, which is propagated from the coarser level of the image pyramid, with the high-confidence optical flow vectors $\boldsymbol{u}_M$ before the iterative numerical scheme. Therefore, the initial optical flow field of current scale is not fully dependent on the result of coarser scales, but also includes the distinctive detailed structures of the current scale, so that motion details can be recovered in the coarse-to-fine approach.

5. EXPERIMENTAL RESULT

We evaluate the proposed optical flow algorithm on the sequences with public ground truth in the Middlebury dataset [10]. The test image pairs are shown in Fig. 5. Average angular error (AAE) and average endpoint error (AEE) are used to measure the accuracy of the optical flow algorithms. In the experiments, the parameters of the proposed algorithm are set as: $\lambda_b = 40$, $\lambda_s = 20$, *SCALE_FACTOR* = 0.5, $\theta = 0.3$, $\tau = 0.25$, *WARPS* = 5, *OUTER_ITERATIONS* = 5, *INNER_ITERATIONS* = 2 for both accuracy and computational complexity considerations. The calculated optical flow fields and ground truth are shown in Fig. 6.

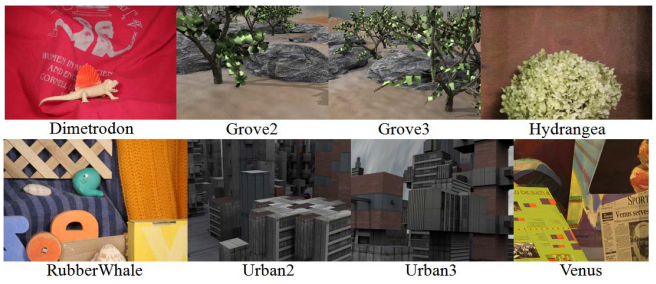
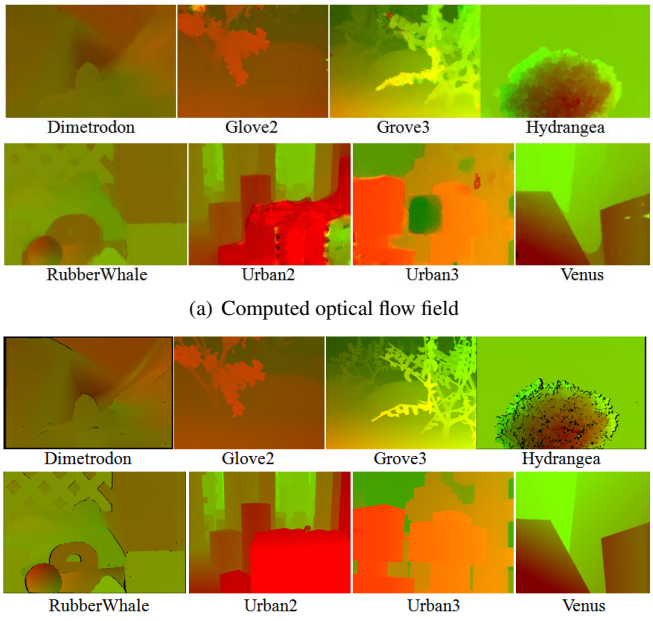


Fig. 5. Test images in Middlebury dataset

First, we compare the performance of the proposed algorithm with the original TV-ℓ^1 optical flow algorithm [3]. The parameters for the original TV-ℓ^1 optical flow algorithm are the same as the proposed algorithm and $\lambda = \lambda_s = 20$. The result is shown in Table 1 and some local motion details are compared in Fig. 7.

The proposed method has better performance than the original TV-ℓ^1 algorithm in *Dimetrodon*, *Grove2*, *Grove3*, *Urban2* and *Urban3* sequences, because these sequences have clear motion boundaries and object structures, which is suitable for the proposed keypoint matching and motion-based weight selection algorithms. However, in *Hydrangea* and *Venus* sequences, the proposed algorithm has worse performance than the original TV-ℓ^1 algorithm, mainly because *Hydrangea* has no clear motion boundaries and the dense letters in *Venus* may cause mismatches of keypoints.



(a) Computed optical flow field

(b) Ground truth optical flow field

Fig. 6. Visual comparison for computed and ground truth optical flow fields

Table 1. Comparison with the original TV-ℓ^1 algorithm

Image pair	Original		Proposed	
	AAE	AEE	AAE	AEE
Dimetrodon	3.838	0.199	2.618	0.143
Grove2	2.809	0.192	2.315	0.156
Grove3	7.403	0.761	6.573	0.677
Hydrangea	2.602	0.219	2.701	0.230
RubberWhale	5.230	0.167	5.522	0.169
Urban2	3.917	1.062	3.337	0.399
Urban3	16.088	1.642	5.407	0.709
Venus	6.130	0.370	7.993	0.530

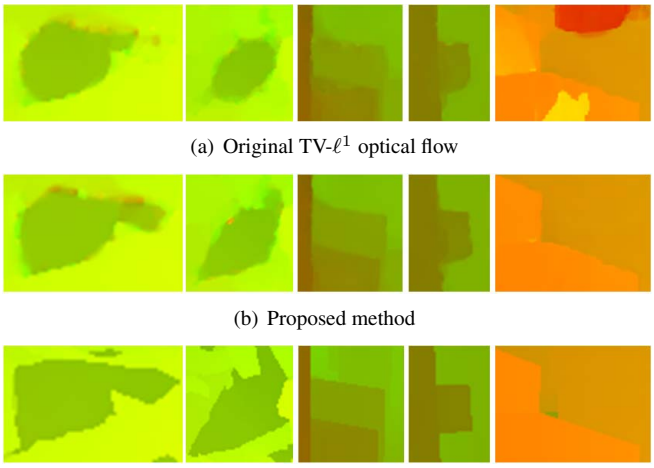
We also compare the proposed method with two classical methods [1] [1][11] and two recently proposed methods [2] [12][13]. The result is shown in Fig. 8.

6. CONCLUSIONS

In this paper, we improve TV-ℓ^1 optical flow algorithm by adopting two novel methods called keypoint matching and motion-based weight selection. Experimental results show that the proposed method has better performance than the original TV-ℓ^1 algorithm, especially in small structures of the optical flow field, and outperforms most state-of-the-art opti-

[1] http://www.cs.brown.edu/~dqsun/research/software.html

[2] http://people.csail.mit.edu/celiu/OpticalFlow/



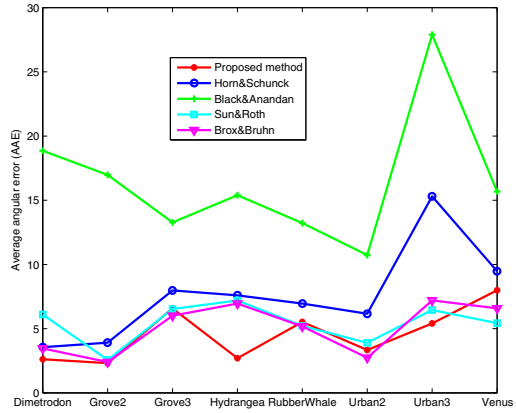
(a) Original TV-ℓ^1 optical flow

(b) Proposed method

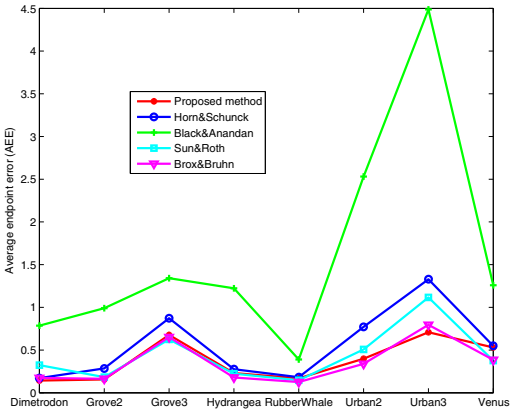
(c) Ground truth

Fig. 7. The proposed method preserves more motion details than original TV-ℓ^1 algorithm

cal flow algorithms. The next step is to improve the efficiency. In order to achieve real time computation of optical flow field, we will explore the feasibility of deploying the proposed algorithm on GPUs.



(a) AAE comparison



(b) AEE comparison

Fig. 8. Performance comparisons of different methods

References

[1] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

[2] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime tv-l1 optical flow," *Pattern Recognition (Proc. DAGM)*, pp. 214–223, 2007.

[3] A. Wedel, T. Pock, and C. Zach, "An improved algorithm for tv-l1 optical flow computation," *In Dagstuhl Visual Motion Analysis Workshop*, 2008.

[4] L. Xu, J. Jia, and Y. Matsushita, "Motion detail preserving optical flow estimation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010, pp. 1293–1300.

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

[6] A. Chambolle, "Total variation minimization and a class of binary mrf models," *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 136–152, 2005.

[7] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1994, pp. 593–600.

[10] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2007, pp. 1–8.

[11] 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.

[12] N. T. Brox, A. Bruhn, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European Conference on Computer Vision*, 2004, pp. 25–36.

[13] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2010, pp. 2432–2439.