A SPATIALLY RECURSIVE OPTICAL FLOW ESTIMATION FRAMEWORK USING ADAPTIVE FILTERING

Teahyung Lee and David V. Anderson

School of Electrical and Computer Engineering, Georgia Institute of Technology
Atlanta, Georgia 30332–0250 Email: taehyung,dva@ece.gatech.edu

ABSTRACT

In this paper, we propose a spatially recursive optical flow estimation (OFE) framework using adaptive filtering. One of most successful OFE algorithms is a gradient-based least-squares (LS) within a local image window because of high performance and low-complexity. However, it has some redundancies for calculating successive LS among adjacent pixels. Therefore, we suggest an efficient framework using recursive least-squares (RLS) and adaptive filtering to improve the computational efficiency. The performance and computational complexity are compared to least-squares OFE and spatially recursive OFE algorithms. Based on these results, we conclude that our proposed algorithm framework under proper window size can reduce computational complexity especially as the number of motion modeling parameters increases by using the property of RLS and adaptive filtering.

Index Terms— Motion analysis, least squares methods, recursive estimation, machine vision, image processing

1. INTRODUCTION

In computer vision and image processing, optical flow fields are often used for processing a sequence of images. Applications of optical flow estimation range from video compression to three-dimensional (3-D) surface structure estimation to object tracking.

Based on simulation results in [1] and [2], a gradient-based optical flow estimation (OFE) algorithm using a least-squares (LS) technique is one of most successful solutions in terms of performance and the number of operations. However, it has some redundancies for calculating successive LS among adjacent pixels. As video frame rate and/or video format size per each frame increase, it is important to find more efficient ways of estimating motion information.

Previous work regarding efficient optical flow estimation algorithms are reported in [3], [4]. and [5]. Liu, *et al.* improved the performance using an adaptive structure tensor and an affine parametric model. However, considering the computational complexity for embedded real-time system design, a gradient-based OFE using a LS technique can be a better choice because of low-complexity with comparable performance. Rav-Acha and Peleg proposed the efficient way to implement Lukas-Kanade algorithm, which is an iterative form of LS-type algorithm by using warping. Therefore, it is hard to expect to improve the computational complexity of non-iterative LS-type algorithms. Fleet and Langley employed recursive temporal filtering using infinite impulse reaponse (IIR) filtering for a LS technique of local first-order constraints to reduce the number of temporal frames and filtering operations [5]. Even though Fleet and Langley achieved an efficient algorithm using temporal recursive filtering with comparable performance [1], a potential problem is caused from IIR filtering. IIR approach is more sensitive to finite precision arithmetic, as is commonly required for efficient implementations.

In this paper, we show how to design a spatially recursive optical flow estimation (OFE) framework using adaptive filtering and sliding window RLS techniques and simulation results for constant and affine motion models.

The rest of the paper is organized as follows. The introduction of a gradient based OFE using local optimization is described in section 2. The proposed sliding window RLS-based OFE framework using adaptive filtering is explained in section 3. In section 4, the performance and computational complexity are compared. The conclusions are presented at the last section.

2. GRADIENT-BASED OFE USING LOCAL OPTIMIZATION

Gradient-based optical flow estimation is based on the constraint equation (Eq. (1)) using first derivatives with respect to spatial and temporal domains [1].

$$\vec{I_s} \cdot \vec{v} + I_t = 0, \tag{1}$$

where $\vec{I_s} = (I_x, I_y)$ and I_x, I_y, and I_t are spatial and temporal derivatives of an image I respectively, and $\vec{v} = (v_x, v_y)^T$ is a motion vector for a dense motion field. Equation (1) is derived from the Taylor series expansion of the translational brightness consistency between successive image sequences. The brightness consistency can be described in Eq. (2).

$$I(x, y, t) = I(x + v_x, y + v_y, t + 1), \tag{2}$$

where $I(x, y, t)$ is the intensity of the image at a point (x, y) at a time t.

Since Eq. (1) has two unknowns in one linear constraint equation, it is an under-determined system. To find a unique solution for the motion vector, more constraint terms must be incorporated. Local or global optimization techniques are often used to calculate motion vectors for additional local or global smoothness constraints. This transforms the problem into an estimation problem based on the derivative values of image, such as regularization or least squares.

One of the most popular gradient-based OFE algorithms using local optimization is an OFE algorithm using a local least-squares (LS) technique for solving a system of constraint equations [1]. This algorithm, LS-OFE, assumes constant motion consistency within a local window area. Mathematically this can be described as follows:

$$\arg\min_{\vec{v}} \sum_{\vec{x}\in\Omega} W(\vec{x},t)[\vec{I_s}\cdot\vec{v}+I_t(\vec{x},t)]^2, \tag{3}$$

where $W(\vec{x},t)$ is a weighting window, Ω is a region of interest for current position, and $\vec{x}=(x,y)$.

The solution for Eq. (3) can be described as follows:

$$\vec{v}=[A^TWA]^{-1}A^TW\vec{b}, \tag{4}$$

where

$$A=[\vec{I_s}(\vec{x_1},t),\cdots,\vec{I_s}(\vec{x_n},t)]^T, \tag{5}$$
$$W=diag[W(\vec{x_1},t),\cdots,W(\vec{x_n},t)], \tag{6}$$
$$\vec{b}=-[I_t(\vec{x_1},t),\cdots,I_t(\vec{x_n},t)]^T. \tag{7}$$

When $[A^TWA]^{-1}$ exists, we can solve for $\vec{v}$ using Eq. (4).

$$A^TWA=\begin{bmatrix}\sum WI_x^2 & \sum WI_xI_y\\ \sum WI_xI_y & \sum WI_y^2\end{bmatrix} \tag{8}$$

$$A^TW\vec{b}=\begin{bmatrix}-\sum WI_xI_t\\ -\sum WI_yI_t\end{bmatrix} \tag{9}$$

Equation (4) is a LS solution for calculating an optical flow.

This gradient-based OFE using local optimization can be generalized using a LS concept to incorporate parametric motion models since it employs a block-wise constant motion model. Based on the parametric motion-model, some parts of equations of LS-OFE can be changed as follows:

$$\vec{I_s}(\vec{x},t)=D,\quad \vec{v}=M\cdot\vec{p}, \tag{10}$$

where

$$D=[I_x,I_y],\ M=\begin{bmatrix}1&0\\0&1\end{bmatrix},\ \vec{p}=[p_1,p_2]^T. \tag{11}$$

By changing M and $\vec{p}$ depending on the parametric motion model, we can implement LS-OFE algorithms with different motion models (See Table 2). Equations (10) and (11) are for the LS-OFE algorithm with the constant model model. We use the notation of Eq. (10) for other parametric cases.

3. THE PROPOSED FRAMEWORK USING ADAPTIVE FILTERING AND RLS

Even though the gradient-based optical flow estimation (OFE) algorithm using a least-squares (LS) technique is one of most successful solutions in terms of performance and computational complexity, it has some redundancies for calculating successive LS among adjacent pixels. Therefore, recursive least-squares (RLS) and adaptive filtering techniques can be applied for improving the computational efficiency.

We can model LS-OFE algorithm as an adaptive filtering problem using a sliding window RLS technique. First we model the least-squares error function.

$$\varepsilon=\sum_{\vec{x}\in\Omega}|e(\vec{x},t)|^2, \tag{12}$$

where Ω is a region of interest and

$$e(\vec{x},t)=d(\vec{x},t)-y(\vec{x},t)=d(\vec{x},t)-\vec{f}^T\cdot\vec{w}. \tag{13}$$

In Eq. (13), $d(\vec{x},t)$ is the desired signal, $\vec{f}$ is the input signal for the FIR adaptive filter, and $\vec{w}$ is the filtering coefficient of the FIR adaptive filter. This set-up can transform the LS problem into adaptive filtering framework.

Based on Eq. (3), (10), and (13), we can model ε as follows:

$$\begin{aligned}\varepsilon &= \sum_{\vec{x}\in\Omega}W(\vec{x},t)[\vec{I_s}\cdot\vec{v}+I_t(\vec{x},t)]^2=\sum_{\vec{x}\in\Omega}|e(\vec{x},t)|^2\\ &= \sum_{\vec{x}\in\Omega}|d(\vec{x},t)-\vec{f}^T\cdot\vec{w}|^2,\end{aligned} \tag{14}$$

where $d(\vec{x},t)=-I_t(\vec{x},t)$, $\vec{f}^T(i)=\vec{I_s}$, and $\vec{w}=\vec{p}$. We use $W=I$ to match the adaptive filtering framework with a LS solution in Eq. (14). The gradient-based optical flow estimation (OFE) algorithm using a least-squares (LS) technique can be mapped into a sliding window RLS algorithm with Eq (14). We simplify the notation of a spatial and temporal axis format into an 1-D spatial axis form for illustration in Fig. 1 and Eq (15). Sliding window RLS is composed of two steps. The first step is a growing window RLS. This can be called the '+' step in Fig. 1 because this stage adds a new data point for least-squares (LS). To perform LS for the same number of data points, we need to remove the effect of the oldest

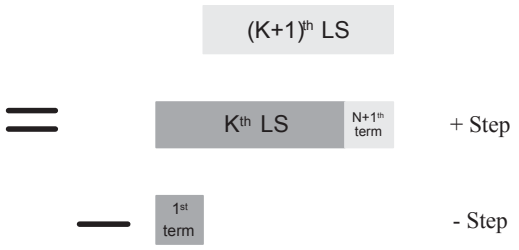


Fig. 1. Two-step sliding window algorithm description.

data point. We can call this step the '-' step in Fig. 1 because this step reduces the window. Mathematically the sliding window algorithm can be regarded as two-step LS based on the previous local-window LS result. If we assume that the LS of previous pixel position is already calculated, the new LS can be calculated as follows (See Figure 1):

$$\min_{\vec{w}(K)} \sum_{1}^{N} |e(i)|^2 \xrightarrow{'+'step} \min_{\vec{w^+}(K+1)} \sum_{1}^{N+1} |e(i)|^2$$

$$\xrightarrow{'-'step} \min_{\vec{w}(K+1)} \sum_{2}^{N+1} |e(i)|^2, \quad (15)$$

1. Signal modeling:
   $d(i) = -I_t(\vec{x}, t)$, $\vec{f}^T(i) = \vec{I_s}$, and $\vec{w} = \vec{p}$
   $e(i) = d(i) - y(i) = d(i) - \vec{f}^T(i) \cdot \vec{w}$
   $\mathbf{R}_f(n) = \sum \vec{f}(i) \cdot \vec{f}^T(i)$,
   $\vec{r}_{df}(n) = \sum d(i)\vec{f}(i)$, $\mathbf{P}(n) = \mathbf{R}_f^{-1}(n)$
2. Algorithm
   For $n = 0...N$
   Do growing window RLS
   a) $\vec{z}(n) = \mathbf{P}(n-1)\vec{f}(n)$
   b) $\vec{g}(n) = \frac{1}{1+\vec{f}^T(n)\cdot\vec{z}(n)}\vec{z}(n)$
   c) $\alpha(n) = d(n) - \vec{w}^T(n-1) \cdot \vec{f}(n)$
   d) $\vec{w}^+(n) = \vec{w}(n-1) + \alpha(n)\vec{g}(n)$
   e) $\mathbf{P}^+(n) = \mathbf{P}(n-1) - \vec{g}(n) \cdot \vec{z}^T(n)$
   Do reducing window RLS
   f) $\vec{z}^+(n) = \mathbf{P}^+(n)\vec{f}(n-L-1)$
   g) $\vec{g}_+(n) = \frac{1}{1-\vec{f}^T(n-L-1)\cdot\vec{z}^+(n)}\vec{z}^+(n)$
   h) $\alpha^+(n) = d(n-L-1) - \vec{f}^T(n-L-1)\cdot\vec{w}^+(n)$
   i) $\vec{w}(n) = \vec{w}^+(n) - \alpha^+(n)\vec{g}^+(n)$
   j) $\mathbf{P}(n) = \mathbf{P}^+(n) + \vec{g}^+(n) \cdot \vec{z}^{+T}(n)$

Table 1. A spatially recursive optical flow estimation framework using adaptive filtering and sliding window RLS techniques. We simplify the notation of the spatial and temporal axis format into a 1-D spatial axis form for illustration

where the '+' step is a growing window RLS step and the '-' step is a reducing window RLS step. The sliding window algorithm can turn $O(n^3)$ of LS into $O(n^2)$, where n is the 1-D size of a square matrix.

By using the sliding window RLS algorithm and the relationships in Eq (14), we can make a spatially recursive optical flow estimation framework. The algorithm is described in Table 1.

4. EXPERIMENTAL RESULTS AND DISCUSSIONS

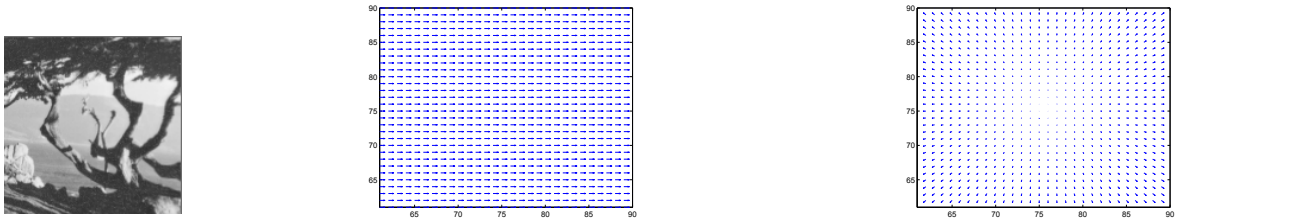
We performed simulations of our spatially recursive optical flow estimation (SR-OFE) algorithms and the gradient-based optical flow estimation algorithm using a least-squares (LS) technique for constant and affine motion models. Test images are translational and diverging tree images with size 150×150 in Fig. 2. A gaussian distribution function with $\sigma = 1.5$ is used for the 3-D smoothing kernel. For a derivative filter, we use $\frac{1}{12}(-1, 8, 0, -8, 1)$. The performance of the algorithm is tested based on the angular error between correct and estimated motion. We perform simulations to compare performance to the original gradient-based optical flow estimation algorithm using a least-squares (LS) technique - the simplified Lukas-Kanade OFE algorithm in [1]. Optical flow field results are presented in Table 3. For the constant motion model, the performance results of the SR-OFE algorithm are almost same as those of the LS-OFE algorithm for same window sizes. The difference is caused by weighting matrix. However, there is computational complexity reduction about 60% by changing the LS-OFE algorithm to the SR-OFE algorithm since we can reuse the data for LS matrix. For the

1. Constant motion modeling:
   $d(i) = -I_t(\vec{x}, t)$, $\vec{f}^T(i) = DM$, and $\vec{w} = \vec{p}$
   $\vec{I_s}(\vec{x}, t) = D$,
   $D = [I_x, I_y]$, $\quad M = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $\quad \vec{p} = [p_1, p_2]^T$
2. Affine motion modeling:
   $d(i) = -I_t(\vec{x}, t)$, $\vec{f}^T(i) = DM$, and $\vec{w} = \vec{p}$
   $\vec{I_s}(\vec{x}, t) = D$,
   $D = [I_x, I_y]$, $\quad M = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}$,
   $\vec{p} = [p_1, p_2, p_3, p_4, p_5, p_6]^T$

Table 2. The parametric motion modelings of the constant and affine models for SR-OFE algorithms.

affine motion model, the performance results of the SR-OFE with affine motion model and LS-OFE algorithms are presented in Table 3. The affine motion model [6] for SR-OFE is described in Table 2. The execution speed comparison of conventional LS-OFE and spatially recursive LS-OFE (SR-OFE) algorithms is plotted in Fig. 3. The affine parametric motion model is employed for SR-OFE algorithm.

In order to compare the number of operations of LS-OFE and SR-OFE algorithms, first we make least squares problems into normal equations (NE). In SR-OFE algorithm, RLS with adaptive filtering are applied for the NE. Cholesky factorization is used to solve the NE for LS-OFE algorithm cases if the matrix is positive definite.To save computations, some additional memory is used for the SR-OFE algorithms. By setting $N = 6$ for the affine motion model in Table 1, SR-OFE algorithm can execute about 4 times faster than LS-OFE algorithm (See Figure 3). The computational complexity of SR-OFE algorithm is $O(kn^2)$, where n is the number of motion modeling parameters and k is the number of updating point(s) or line(s) per LS. Therefore, SR-OFE algorithm can be highly efficient if the number of parameters (n) in motion modeling increase and k is small compared to the number of



(a) Basic sample image (b) Part of motion field for translating tree images (c) Part of motion field for diverging tree images

Fig. 2. The basic sample image and motion fields of translating and diverging image sequences.

motion modeling parameters. This is the reason why SR-OFE algorithm with affine motion modeling has higher reduction than constant motion modeling.

The performance of SR-OFE algorithm from 1 to 3 points or lines update per LS operation are presented in Table 3. To compare the efficiency of the parametric motion models, SR-OFE algorithm with affine motion model, ASR-OFE algorithm, is tested. Same derivative and smoothing kernels are employed for our simulations in Table 3 as described in [1] except threshold value. The reason why we change threshold value is that the value is dependent on the number of motion modeling. Based on Table 3, we can notice that ASR-OFE has better performance for a smoothly spatial-varying motion field in diverging tree images than a constant motion field in translational tree images since a smoothly spatial-varying motion field can be well represented in affine form. As the number of updated pixels are increasing, the performance slightly decreases. This is caused by using same threshold number. If we change threshold numbers depending on the number of updated pixels, the result can be improved for the higher number of updated pixels cases. Overall the performance of ASR-OFE is comparable with that of LS-OFE with more than 4 times faster execution speed.

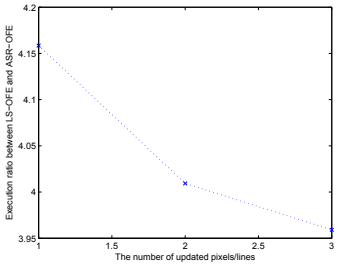


Fig. 3. The execution speed ratio between LS-OFE and ASR-OFE algorithms. The parametric motion model is affine model. LS-OFE is least-squares OFE and ASR-OFE is spatially recursive OFE with affine model. The proposed affine SR-OFE (ASR-OFE) algorithm shows computational efficiency compared with affine LS-OFE algorithm.

5. CONCLUSIONS

In this paper, we propose a spatially recursive optical flow estimation (OFE) framework using adaptive filtering. The sliding window RLS and adaptive filtering are employed to reduce the redundancies for calculating successive LS among adjacent pixels. Even though we describe about constant and affine motion models, this framework can be applied to other parametric motion models. Under authors knowledge, there are parametric motion models up to 12 parameters [6]. In addition, some algorithms such as ego-motion estimation, multi-frame OFE, and 3-D OFE, need more number of motion parameters for estimating than general 2-D OFE. Based on these results, we can conclude that our algorithm framework can improve the saving of computational complexity more as the number of motion modeling parameter increases because standard LS needs $O(n^3)$ and RLS requires $O(kn^2)$ computations, where n is the number of motion modeling parameters and k is the number of updating point(s) or line(s) per LS.

	Translating Tree		Diverging Tree	
Algorithm	Avg Error	Std	Avg Error	Std
LS-OFE	0.66°	0.67°	1.94°	2.06°
SR-OFE-1pt	0.76°	0.63°	1.64°	1.42°
SR-OFE-2pt	0.80°	0.75°	1.83°	1.87°
SR-OFE-3pt	0.87°	0.90°	1.90°	2.01°

Table 3. Angular error performance of spatially recursive OFE algorithm with affine motion model and LS-OFE algorithm in [1]. The threshold for LS-OFE algorithm is 1. Avg and Std means average and standard deviation of angle error, respectively. And pt means the number of updating point(s) or line(s).

6. REFERENCES

[1] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *International Journal of Computer Vision*, vol. 12, no. 1, pp. 43–77, Feb. 1994.

[2] H. Liu, T.-H. Hong, M. Herman, T. Camus, and R. Chellappa, "Accuracy vs efficiency trade-offs in optical flow algorithms," *Comp. Vision Image Understanding*, vol. 72, pp. 271–286, 1998.

[3] H. Liu, R. Chellappa, and A. Rosenfeld, "Accurate dense optical flow estimation using adaptive structure tensors and a parametric model," *IEEE Trans. Image Processing*, vol. 12, no. 10, pp. 1170–1180, Oct. 2003.

[4] A. Rav-Acha and S. Peleg, "Lukas-kanade without iterative warping," in *IEEE International Conference on Image Processing*, Atlanta, Oct 2006.

[5] D. J. Fleet and K. Langley, "Recursive filters for optical flow," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, pp. 61–67, Jan. 1995.

[6] J. L. Barron and M. Khurana, "Determining optical flow for large motions using parametric models in a hierarchical framework," in *Vision Interface (VI1997)*, Kelowna, B.C., May 1997.