LAP-BASED VIDEO FRAME INTERPOLATION

Tejas Jayashankar[1], *Pierre Moulin*[1], *Thierry Blu*[2], *and Chris Gilliam*[3]

[1]Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign
[2]Department of Electronic Engineering, The Chinese University of Hong Kong
[3]School of Engineering, RMIT University, Melbourne, Australia

ABSTRACT

High-quality video frame interpolation often necessitates accurate motion estimation, which can be obtained using modern optical flow methods. In this paper, we use the recently proposed Local All-Pass (LAP) algorithm to compute the optical flow between two consecutive frames. The resulting flow field is used to perform interpolation using cubic splines. We compare the interpolation results against a well-known optical flow estimation algorithm as well as against a recent convolutional neural network scheme for video frame interpolation. Qualitative and quantitative results show that the LAP algorithm performs fast, high-quality video frame interpolation, and perceptually outperforms the neural network and the Lucas-Kanade method on a variety of test sequences.

Index Terms— Optical flow, Convolutional neural network, Lucas-Kanade algorithm, Video interpolation, Splines

1. INTRODUCTION

High-end video display systems ranging from TVs and laptops to smartphones, aim at faithfully rendering high-resolution image sequences. In particular, it is often desired to display the video at a higher temporal rate than the input video provides for. Then some mechanism for video interpolation is needed, providing high visual quality and often requiring real-time operation. [1] Clearly there is a tradeoff between these two requirements.

Advanced video interpolation methods often require estimation of the displacement field between frames. For compressed video this has been done by exploiting the motion vectors [1]. However higher-quality displacement fields can be potentially obtained using optical flow methods [2, 3]. While a variety of optical flow estimation algorithms can be used, e.g, the pyramidal version of the Lucas-Kanade algorithm [4], we focus in this paper on the Local All-Pass (LAP) algorithm proposed by Gilliam and Blu [5]. A strong motivation for this choice is the high approximation order of the LAP algorithm [6], which is typically quadratic while traditional optical flow methods are only first order. Moreover the LAP method is fast, which also provides another strong motivation for applying it to video interpolation, especially at high spatio-temporal resolutions, when real-time operation is needed and the computational requirements of the optical flow estimation algorithm are critical.

It has also been recognized that displacement fields need not be estimated explicitly for video interpolation. In particular, excellent results have recently been achieved using convolutional neural networks (CNNs) [7][8][9] for computing the interpolated frames. Unfortunately, these deep networks require extensive training and have very high computational and storage requirements on test sequences as well. We compare our method with a CNN method proposed by Niklaus *et al.*, which runs on a GPU and learns 1-D kernels that can be convolved with the input frames [10]. We choose to use this method because the code is publicly available online.

[1] In applications such as slow motion rendition, real-time operation may not be needed.

2. VIDEO FRAME INTERPOLATION METHODS

We formulate the problem as interpolating a frame $I_{1/2}$ temporally between two video frames I_0 and I_1. Optical flow methods generate this intermediate frame using the optical flow estimates between I_0 and I_1. The adaptive separable convolution approach performs interpolation using a forward-backward filtering approach. The neural network learns the forward and backward filters that reduces the perceptual loss.

2.1. LAP Algorithm

The LAP algorithm [5] is based on the observation that spatial shifting is an instance of all-pass filtering. Consider two images I_1 and I_2 that are related locally at $\mathbf{x} = (x_1, x_2)^\top$ by

$$I_2(\mathbf{x} + \mathbf{u}) = I_1(\mathbf{x}), \tag{1}$$

where $\mathbf{u} = (u_1, u_2)^\top$ is the displacement field at $\mathbf{x}$. In the frequency domain this relation is equivalent to

$$\hat{I}_2(\boldsymbol{\omega}) = \hat{I}_1(\boldsymbol{\omega}) e^{-j\boldsymbol{\omega}^\top \mathbf{u}}, \tag{2}$$

where $\hat{I}$ represents the 2-D Fourier Transform of I and $\boldsymbol{\omega} = (\omega_1, \omega_2)$. The right hand side of (2) can be thought of as filtering $\hat{I}_1$ with the all-pass filter $h(\boldsymbol{\omega}) = e^{-j\boldsymbol{\omega}^\top \mathbf{u}}$. Since any real digital all-pass filter can be written as $h(\boldsymbol{\omega}) = p(\boldsymbol{\omega})/p(-\boldsymbol{\omega})$, we can now write $\hat{I}_2(\boldsymbol{\omega})p(-\boldsymbol{\omega}) = \hat{I}_1(\boldsymbol{\omega})p(\boldsymbol{\omega})$. In the discrete domain with pixel coordinates $\mathbf{k} = [x, y] \in \mathbb{Z}^2$, this results in the following forward-backward filter relation:

$$I_2[\mathbf{k}] * p[-\mathbf{k}] - I_1[\mathbf{k}] * p[\mathbf{k}] = 0 \tag{3}$$

where $*$ denotes 2-D convolution. Gilliam and Blu [5] represent the filter p using a basis of known real filters p_n:

$$p_{app}[\mathbf{k}] = \sum_{n=0}^{N-1} c_n p_n[\mathbf{k}]. \tag{4}$$

Without loss of generality, let $c_0 = 1$. The remaining coefficients c_n are estimating by solving the following quadratic minimization problem at each pixel $\mathbf{k}$:

$$\min_{\{c_n\}} \langle |I_2[\mathbf{k}] * p_{app}[-\mathbf{k}] - I_1[\mathbf{k}] * p_{app}[\mathbf{k}]|^2 \rangle_{\mathcal{W}}, \tag{5}$$

where $\langle J \rangle_{\mathcal{W}} = \sum_{\mathbf{k}} J[\mathbf{k}]$ and $\mathcal{W}$ is window around each pixel. The solution is found by solving a system of $N - 1$ equations with $N - 1$ unknowns using Gaussian elimination.

In each iteration of the algorithm, an optical flow estimate is obtained and I_1 is warped towards I_2. The estimated flow vectors undergo a round of kernel based image inpainting to remove erroneous values followed by a round of mean and median filtering to smoothen the flow. Instead of using an image pyramid, a 3-level filter pyramid is used. In each iteration the filter width is reduced by a factor of two to refine the flow estimates and account for varying amplitudes. We use $N = 4$ to obtain second-order approximation order. Implementation details can be found in [11].

2.2. Adaptive Separable Convolution

This CNN method [10] performs video interpolation by solving a forward-backward problem similar to (3). Given two frames I_1 and I_2, the CNN finds two separable kernels K_1 and K_2 at each pixel such that:

$$\hat{I}[\mathbf{k}] = P_1[\mathbf{k}] * K_1[\mathbf{k}] + P_2[\mathbf{k}] * K_2[\mathbf{k}], \tag{6}$$

where $\hat{I}$ is the ground truth interpolated frame, and P_1 and P_2 are patches centered around the pixel of interest in images I_1 and I_2 respectively. Each sample of the training set is comprised of three consecutive images. The middle one represents the ground truth for interpolation.

Rather than using 2D filters, Niklaus *et al.* designed the neural network to learn four 1D filters per pixel. This reduces the space complexity from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ per pixel. The neural network uses convolutional, averaging pooling and ReLU layers. Skip connections are used to improve the stability of the otherwise unstable contracting network. The network is trained using AdaMax and a perceptual loss function based on the `relu4_4` layer of the VGG-19 network. The authors mention that it took 20 hours to train on an NVIDIA Titan X (Pascal). As shown in Table 1, computation of the interpolated frames is extremely heavy due to the size of the network. For this reason the authors ran their interpolation algorithm on a GPU. The operations count is more than 300 times higher than it is for LAP.

Table 1: Number of operations (multiply-adds) and execution time (CPU) required to interpolate a frame of size 960x540x3. Execution time is in seconds, measured on a machine with Intel Core i7-8750H processor and 32 GB RAM. The execution time for the CNN was extrapolated based on the time required to interpolate 200 pixels.

	CNN	LAP	LK
Operation Count	4.9×10^{11}	1.5×10^9	6.1×10^7
Execution Time	76204	4.3	0.2

2.3. Lucas-Kanade Method

The basic Lucas-Kanade (LK) method [4] solves for the motion vector $\mathbf{v}$ using local weighted least-squares matching in a neighbourhood around the pixel under consideration. In our experiments, we use a modern version of the Lucas-Kanade method available in Piotr's Computer Vision Toolbox [12]. This version uses the weighted version of the Lucas-Kanade method and image pyramids. The algorithm uses a 7-layer image pyramid with the highest level computing the flow on the image subsampled by 64 and refining the estimate in each level by warping the first image closer to the second. Best results were obtained by using a neighborhood of radius 10 pixels.

2.4. Performance Evaluation Criteria

In addition to computational speed, it is desirable to employ an objective performance metric to evaluate the match between the interpolated frame and the ground truth. The standard criterion is the ubiquitous Mean-Squared Error (MSE) [13]. Unfortunately MSE is a particularly bad criterion for measuring the quality of interpolated images, as slight misalignments can be quite acceptable perceptually yet cause large squared errors [14]. We have experimented with other metrics such as SSIM [15] and CW-SSIM [16] but they suffer from similar artifacts. Therefore we rely extensively on visual evaluation to assess performance of competing algorithms. The discrepancy between MSE and perceptual quality is often striking.

3. RESULTS

We now evaluate the methods discussed in Section 2 on three standard datasets: Middlebury [2], EPIC Kitchens [17] and Derf's Media Collection [18]. All results were obtained on a machine with an Intel Core i7-8750H processor with 32

Table 2: MSE Evaluation on the Middlebury dataset (high-speed camera samples). **Bold** values indicate best results.

	Beanbags	DogDance	MiniCooper	Walking	Backyard	Basketball	Dumptruck	Evergreen
LAP	339.0	240.1	176.7	67.9	163.5	198.9	209.3	268.6
CNN	**196.9**	**159.4**	**80.5**	**57.2**	**102.6**	**105.8**	**88.3**	**102.8**
LK	454.7	223.9	233.2	97.3	273.5	157.2	281.7	276.8

GB RAM. The CNN was executed on an NVIDIA GTX 1050Ti GPU with 4GB RAM. The interpolation results can be viewed at `https://bit.ly/2WqXbKR`. Sequences should be downloaded else the media player will drop the intermediate frames!

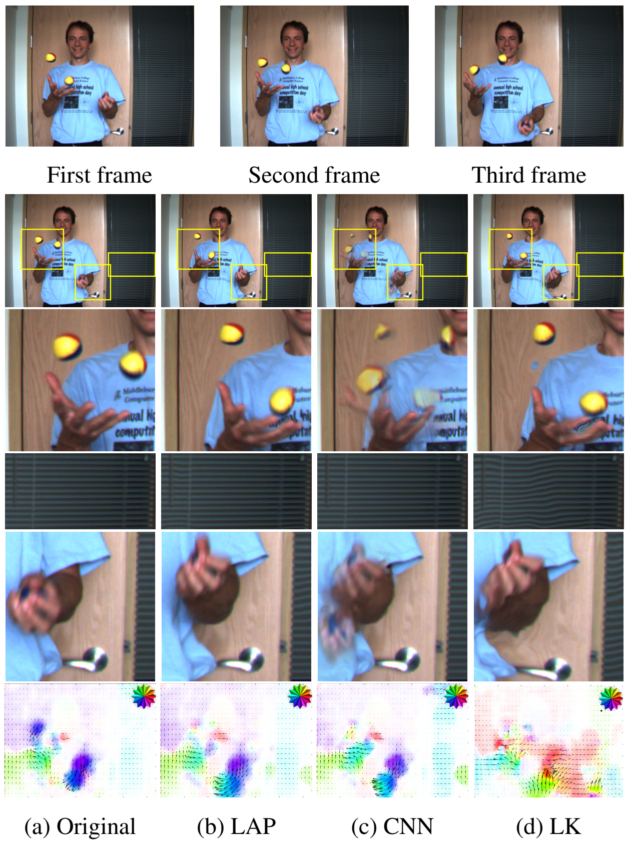

Fig. 1: Beanbags sequence: The original sequence is shown at the top. The original second frame is shown in (a), the LAP-interpolated second frame is shown in (b), the CNN-interpolated second frame is shown in (c), the LK-interpolated second frame is shown in (d). LAP was used to compute the optical flow in (a), (b) and (c).

3.1. Middlebury Dataset

MSE results for the Middlebury dataset are given in Table 2. Although the CNN seems to perform the best interpolation in terms of MSE, we observe that actually the LAP consistently interpolates frames with the highest perceptual quality. This is evident from Figure 1: Notice how the palms are smudged and how the balls are distorted in the CNN interpolated frame. Yet, since the imprints of the middle two balls are slightly closer to the actual position of the balls, the CNN interpolated frame has a lower MSE. There is also a greater intensity match between the pixels in the original image and the CNN-interpolated image in comparison to the LAP-interpolated images.

The Lucas-Kanade method performed poorly on all sequences. The flow field across the juggler's torso is incorrect. The fingers are deformed in Figure 1, and the blinds are skewed due to inexact optical flow estimates. This is evident from the patch of blue on the door in the zoomed in image of the balls. Also notice how the left side of the juggler's torso has been warped inwards.

The displacement field for the LAP algorithm is the closest to the flow between the original frames. The optical flow between the original frame and CNN-interpolated frame is not as smooth in the blue-yellow region boundary in comparison to the flow between the original frame and LAP-interpolated frame. This results in the juggler's palms being blurred in the CNN-interpolated frame.

3.2. Derf's Media Collection

This dataset consists of video sequences which are generally used to evaluate compression algorithms. Table 3 shows the interpolation results on seven 60 fps videos with 704x576 spatial resolution. We dropped every other frame in the video sequence and interpolated between the remaining pairs using the three methods discussed in Section 2.

Table 3 shows MSE results, and Fig. 2 shows frames for the soccer sequence. While the LAP method yields good visual quality, a visible artifact is the position of the running player in the LAP-interpolated frame, which is slightly to the left of his position in the original frame. The player's leg is also slightly lower and farther away from his body in comparison to the original frame. Still, when viewed as a sequence, the LAP-interpolated video looked natural and smooth.

The Lucas-Kanade method did not compute a very precise flow field. Notice how the displacement vectors are pointing south across the running player. This results in the fence being highly deformed in the interpolated frames. When the inter-

Table 3: Average interpolation MSE on Derf's Media Collection. **Bold** values indicate best results.

	LAP	CNN	LK
city	**62.1**	79.8	111.7
crew	522.1	**406.1**	806.5
harbour	112.2	**94.8**	117.1
ice	129.6	**57.4**	109.9
soccer	848.3	**141.6**	399.5
stockholm	**54.6**	60.4	77.6
riverbed	**361.7**	409.6	445.8

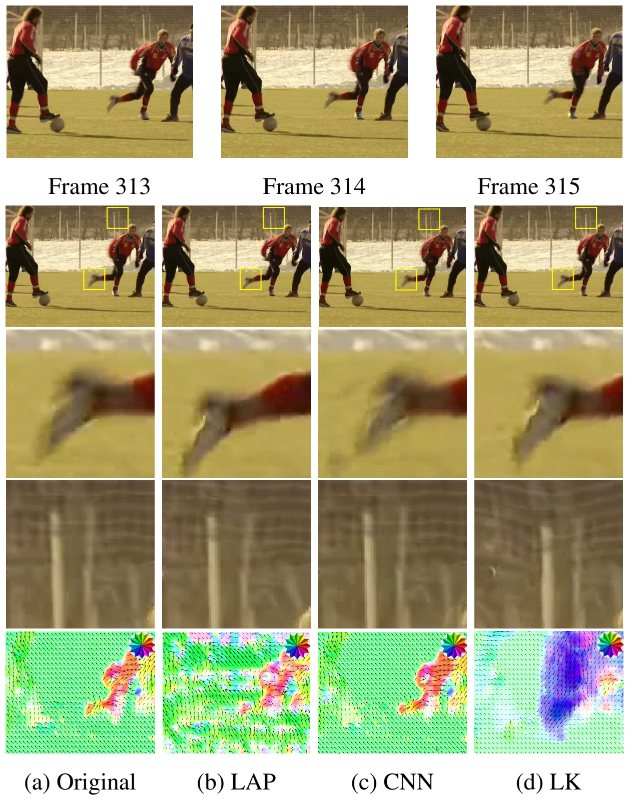

Fig. 2: Soccer sequence: The original sequence is shown at the top. Original frame 314 is shown in (a), LAP-interpolated frame 314 is shown in (b), CNN-interpolated frame 314 is shown in (c) and the LK interpolated 314 is shown in (d). The optical flow between frame 314 and frame 314 is shown for the original sequence in (a) and for the three methods in the other columns. LAP was used to compute the optical flow in (a), (b) and (c).

polated frames are played at 60 frames per second, this causes flickering and strenuous visual artifacts. The neural network almost aligns the interpolated frame with the original frame. However, it blurs the limbs and leaves an imprint of the previous position of the limbs. This appears as a shadow around the limbs when viewed as a sequence.

Table 4: Average interpolation MSE on EPIC Kitchens dataset. **Bold** values indicate best results.

	LAP	CNN	LK
P01_11	163.8	**130.2**	149.5
P01_12	**151.2**	199.0	204.1
P01_14	**152.6**	234.6	251.0
P01_15	**163.2**	273.4	301.9
P02_13	518.4	**458.9**	541.6
P03_21	**193.2**	433.1	491.9
P03_22	**307.1**	494.6	565.8
P03_23	**266.9**	462.1	537.9

3.3. EPIC Kitchens Dataset

Finally, we perform interpolation on eight full-HD sequences from the EPIC Kitchens dataset. Since our GPU has insufficient memory to interpolate a 1920x1080 image, the spatial resolution of the video sequences was reduced to 960x540. The video sequences consist of body cam footage of people navigating around kitchens. The sequences consist of crisp fast motion such as bending down to open cabinets, cutting vegetables and sudden changes in direction. The LAP algorithm performs very well in preserving these motions. The CNN produces artifacts and jerkiness in the frames when there is a sudden change in direction or fast motion. The Lucas-Kanade method also produces similar artifacts which are more pronounced than the CNN interpolated frames.

The average interpolation MSE is reported in Table 4. Here the lower MSE for LAP correlates with the perceptual quality on this dataset. Since the videos have high spatial resolution, there is much more smoothness between consecutive frames. The LAP algorithm computes high accuracy optical flow estimates on smooth flowing sequences and as a results performs better both the CNN and Lucas-Kanade in interpolating between fast motion frames.

4. CONCLUSION

In this paper we have shown that the LAP optical flow method is an excellent candidate for video interpolation. The method has quadratic approximation order, making it exceptionally accurate when the true displacement field is smooth. The method is also very fast, compares very favorably with a recent CNN method, and generally outperforms a pyramid version of Lucas-Kanade. In all cases, MSE is a poor predictor of visual quality of the video.

We have also compared LAP against two additional optical flow methods: Deepflow2 [19] and MDP-Flow2 [20] which score highly on the Middlebury flow/interpolation rankings. Due to lack of space the results are not reported here. However LAP consistently produced interpolated frames of higher quality.

5. REFERENCES

[1] R. Li, M. Wu, Z. Gan, Z. Cui, and X. Zhu, "Joint overlapped block motion compensation using eight-neighbor block motion vectors for frame rate up-conversion," *KSII Transactions on Internet and Information Systems*, vol. 7, pp. 2448–2463, 10 2013.

[2] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, Mar 2011.

[3] R. Krishnamurthy, J. W. Woods, and P. Moulin, "Frame interpolation and bidirectional prediction of video using compactly encoded optical-flow fields and label fields," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 5, pp. 713–726, Aug 1999.

[4] 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 - Volume 2*, San Francisco, CA, USA, 1981, IJCAI'81, pp. 674–679, Morgan Kaufmann Publishers Inc.

[5] C. Gilliam and T. Blu, "Local all-pass filters for optical flow estimation," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, April 2015, pp. 1533–1537.

[6] T. Blu, P. Moulin, and C. Gilliam, "Approximation order of the LAP optical flow algorithm," in *2015 IEEE International Conference on Image Processing (ICIP)*, Sept 2015, pp. 48–52.

[7] C. Li, D. Gu, X. Ma, K. Yang, S. Liu, and F. Jiang, "Video frame interpolation based on multi-scale convolutional network and adversarial training," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, June 2018, pp. 553–560.

[8] J. R. van Amersfoort, W. Shi, A. Acosta, F. Massa, J. Totz, Z. Wang, and J. Caballero, "Frame interpolation with multi-scale deep loss functions and generative adversarial networks," *CoRR*, vol. abs/1711.06045, 2017.

[9] S. Wen, W. Liu, Y. Yang, T. Huang, and Z. Zeng, "Generating realistic videos from keyframes with concatenated gans," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1–1, 2018.

[10] S. Niklaus, L. Mai, and F. Liu, "Video frame interpolation via adaptive separable convolution," in *IEEE International Conference on Computer Vision*, 2017.

[11] C. Gilliam and T. Blu, "Local all-pass geometric deformations," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 1010–1025, Feb 2018.

[12] Piotr Dollár, "Piotr's Computer Vision Matlab Toolbox (PMT)," `https://github.com/pdollar/toolbox`.

[13] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? a new look at signal fidelity measures," *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, Jan 2009.

[14] H. Men, H. Lin, V. Hosu, D. Maurer, A. Bruhn, and D. Saupe, "Technical report on visual quality assessment for frame interpolation," 01 2019.

[15] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[16] M. P. Sampat, Z. Wang, S. Gupta, A. C. Bovik, and M. K. Markey, "Complex wavelet structural similarity: A new image similarity index," *IEEE Transactions on Image Processing*, vol. 18, no. 11, pp. 2385–2401, Nov 2009.

[17] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray, "Scaling egocentric vision: The epic-kitchens dataset," in *European Conference on Computer Vision (ECCV)*, 2018.

[18] "Derf's media collection," `https://media.xiph.org/video/derf/`.

[19] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "DeepFlow: Large displacement optical flow with deep matching," in *IEEE Intenational Conference on Computer Vision (ICCV)*, Sydney, Australia, Dec. 2013.

[20] L. Xu, J. Jia, and Y. Matsushita, "Motion detail preserving optical flow estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1744–1757, Sep. 2012.