LOCALITY SENSITIVE HASHING BASED DEEPMATCHING FOR OPTICAL FLOW ESTIMATION

*Chen Wang, Zongqing Lu, Qingmin Liao**

Department of Electronic Engineering/
Graduate School at Shenzhen,
Tsinghua University, China

Danyi Li

State Key Laboratory of Information Security,
Institute of Information Engineering,
CAS, Beijing, China

ABSTRACT

DeepMatching (DM) is one of the state-of-art matching algorithms to compute quasi-dense correspondences between images. Recent optical flow methods use DeepMatching to find initial image correspondences and achieves outstanding performance. However, the key building block of DeepMatching, the correlation map computation, is time-consuming. In this paper, we propose a new algorithm, LSH-DM, which addresses the problem by employing Locality Sensitive Hashing (LSH) to DeepMatching. The computational complexity is greatly reduced for the correlation map computation step. Experiments show that image matching can be accelerated by our approach in ten times or more compared to DeepMatching, while retaining comparable accuracy for optical flow estimation.

Index Terms— Image correspondence, DeepMatching, Locality Sensitive Hashing, Optical Flow

1. INTRODUCTION

Optical flow estimation plays a fundamental role in many computer vision and image processing tasks, such as object detection and tracking, super-resolution and video de-noising. Over the last several years, great improvement of optical flow methods handling small displacement is witnessed by the Middlebury benchmark[1]. However, estimating optical flow accurately from real-world scenarios, where large displacements and occlusions often occur, still remains a challenging problem[2, 3]. Traditional approaches[4, 5] minimize the optical flow energy functional using a coarse-to-fine scheme. The optimization procedure often gets stuck in local minima and leads to over-smoothing the estimated flow. To overcome such problem, LDOF [6] proposed by Brox and Malik successfully integrates sparse descriptor matching in the energy minimization framework. The key idea is to use descriptor correspondences to guide the variational optimization. Since then, several works incorporate feature matching into their approach[7, 8, 9, 10, 11, 12, 13]. LDOF uses sparse HOG descriptors matched by nearest neighbor search. Another few works use dense approximate nearest neighbor field (ANNF), instead of sparse descriptors [11, 13] and produce remarkable results on the Middlebury dataset. More recently, Revaud et al.[14] proposed a new matching algorithm called DeepMatching. The DeepMatching algorithm produces quasi-dense correspondences for image pairs, and is able to deal with non-rigid deformation and large displacement. Many top-ranked methods on MPI-Sintel dataset is base on DeepMatching to obtain initial image correspondences, either for energy minimization[15] or for interpolation[8]. In [8], the authors show that DeepMatching outperforms ANNF and sparse feature matching in their optical flow estimation framework. The main drawback of DeepMatching is the high computational complexity. DeepMatching possesses a $O(N^2)$ complexity, where N is the total pixel number of one input image. As the image size grows, the computation cost explodes. As a consequence, DeepMatching is usually performed on a down-sampled version of the original images.

Hashing-based approximate nearest neighbor search methods received significant attention in the past years[16, 17]. With a well designed hashing function, all features can be converted into binary hashing codes. Computing the Hamming distance of hashing codes is by conducting one bitwise XOR operation, which is very fast on modern CPUs. In this paper, we propose to employ hashing to accelerate DeepMatching. The utilized hashing method, Locality Sensitive Hashing, is an unsupervised and data-independent method, therefore training-free. Experiment demonstrate that DeepMatching can be accelerated by our approach in ten times or more, while retaining comparable accuracy for optical flow estimation.

The rest of this paper is organized as follows. Section 2 briefly summarizes the DeepMatching and LSH algorithms. Section 3 introduces the proposed LSH-DM algorithm. Section 4 evaluates the proposed method with a number of experiments. Finally, Section 5 concludes this paper with remarks on our future work.

*Corresponding Author, E-mail: liaoqm@tsinghua.edu.cn. This work was supported by Shenzhen STP(JCYJ20150331151358150).

2. DEEPMATCHING AND LOCALITY SENSITIVE HASHING

In this paper, we propose a hashing based method for the correlation map computation in DeepMatching, to speed up image matching for optical flow estimation. In our method, a simple hashing algorithm, Locality Sensitive Hashing (LSH), is adopted to generate binary code for SIFT descriptor extracted from image patches. Consequently, we will give a brief introduction to the DeepMatching and LSH algorithms in this section.

2.1. DeepMatching

We begin this section by introducing the DeepMatching algorithm[14]. DeepMatching is based on feature correlation at the patch-level. Given the source image I_0 and the target image I_1, the *correlation map* is the matching scores of a single patch $\mathbf{p}$ from image I_0 at every position in image I_1. Features extracted for matching patches are based on the SIFT/HOG descriptor. The correlation of 4×4 patches $\mathbf{P}, \mathbf{P}'$ is defined by

$$\text{corr}(\mathbf{P}, \mathbf{P}') = \frac{1}{16} \sum_{i=0}^{3} \sum_{j=0}^{3} \mathbf{P}_{i,j}^{\top} \mathbf{P}'_{i,j}, \tag{1}$$

where the pixel descriptor $\mathbf{P}_{i,j}$ is defined by the histogram of oriented gradients pooled over a 4×4 neighborhood. In (1), the patches are compared using inner product and the correlation score takes value in the interval $[0, 1]$.

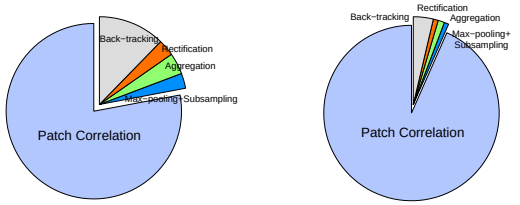
Correlation maps are computed for non-overlapping patches on regular grids in the source image. Once the bottom correlation maps are obtained, the correlation maps of larger patches are computed by aggregating correlation maps of smaller children patches. The aggregation is performed in a quadtree-like manner:

$$\text{corr}(\mathbf{P}, \mathbf{P}') = \frac{1}{4} \sum_{i=0}^{3} \max_{x_i} \text{corr}(\mathbf{P}_i, \mathbf{P}'_i(x_i)) \tag{2}$$

where $\mathbf{P}'_i(x_i)$ is the descriptor of one of the quadrant patches extracted at position x_i. A correlation map pyramid is thereby constructed from the bottom to the top. The advantage of such optimization with regard to quadrant positions is that non-rigid motions are well handled. Inspired by the deep convolutional neural network approaches [18], the aggregation step can be divided into several basic operations including max-pooling, sub-sampling, shift and rectification, as illustrated in [14].

Top correlation maps are built from sub-patches at the lower pyramid level, so one can recursively backtrack correspondences from the top to the bottom level of the pyramid. We also refer to [14] for more details of aggregation and backtracking strategy in DeepMatching.

Figure 1 shows the time cost of separate operations in the DeepMatching pipeline. It is clear that the correlation map computation step occupies most of the consuming time. As the image size grows, computing the correlation map occupies even more proportion of the total time.



(a) 0.5x image size (b) 1x image size

Fig. 1. Time cost of separate operations in the DeepMatching pipeline. As the image size grows, correlation map computation occupies more proportion of the total time.

2.2. Locality Sensitive Hashing

Given a set of d-dimensional features $X = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathbb{R}^d$, LSH relies on the existence of locality sensitive hashing functions. Let $\mathcal{F}$ be a family of hashing functions mapping the feature space $\mathbb{R}^d$ to Hamming space $\mathbb{H}$. For any two points x_i and x_j, a hashing function h is picked from $\mathcal{F}$ uniformly at random. The idea is that if two points x_i and x_j in $\mathbb{R}^d$ are close, then the probability that $h(x_i) = h(x_j)$ should be higher. In formal terms, $\mathcal{F}$ is locality sensitive if it satisfies the following conditions:

Definition 2.1 (Locality Sensitive Hashing)**.** A family $\mathcal{F}$ of functions from $\mathbb{R}^d$ to $\mathbb{H}$ is called (r_1, r_2, p_1, p_2)-sensitive for $D(\cdot, \cdot)$, if for any $x, y \in \mathbb{R}^d$ and h chosen uniformly at random from $\mathcal{F}$ satisfies the following:

- $Pr_{h \in \mathcal{F}}[h(x) = h(y)] \geq p_1$, if $D(x, y) \leq r_1$;
- $Pr_{h \in \mathcal{F}}]h(x) = h(y)] \leq p_2$, if $D(x, y) \geq r_2$,

where $D(\cdot, \cdot)$ is a distance function in the feature space $\mathbb{R}^d$. Obviously, a family $\mathcal{F}$ is valid only when $r_1 < r_2$ and $p_1 > p_2$.

A hashing function $h \in \mathcal{F}$ for the widely used L_2 distance is given by [19]. More specifically, a random vector r is drawn independently from a zero-mean d-dimensional Gaussian distribution $\mathcal{N}(0, I)$, with this hyperplane r, hashing function h is defined as:

$$h_r(q) = \begin{cases} 1, \text{ if } r \cdot q > 0 \\ 0, \text{ if } r \cdot q < 0 \end{cases} \tag{3}$$

known as the sign random projections (SRP)[19, 20]. The authors of [20] proved that

$$Pr(h_r(x) = h_r(y)) = 1 - \frac{\theta(x, y)}{\pi} \tag{4}$$

for any data point x and y, where

$$\theta(x,y) = cos^{-1}(\frac{x^\top y}{\|x\|\|y\|}) \quad (5)$$

is the angle between x and y. Let $D(x,y) = \frac{\theta(x,y)}{\pi}$ and $p_1 = 1 - r_1$, $p_2 = 1 - r_2$. It is easy to prove that $\mathcal{F}$ satisfies Definition 2.1. It can be seen that SRP is designed for cosine similarity, which is equivalent with L_2 distance when features are normalized.

In real cases, K concatenation of hashing functions is used to amplify the gap between p_1 and p_2. The hash function h is actually a concatenation of primitive functions: $h(x) = \langle h_1(x), h_2(x), \ldots, h_K(x)\rangle$.

3. LSH-DM

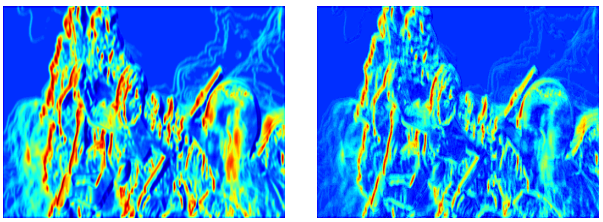
3.1. The LSH-DM Algorithm

Now we introduce our proposed method, LSH-DM. From Definition 2.1 we know that hash codes of highly correlated patches are close in the Hamming space, thanks to the nice property of Locality Sensitive Hashing. Therefore, we use Hamming distance of hash codes to represent the matching score. Small Hamming distance of the hash codes indicates high correlation of the original features. Equation (1) then becomes:

$$\mathrm{corr}(\mathbf{P}, \mathbf{P}') = \frac{1}{16}\sum_{i=0}^{3}\sum_{j=0}^{3} sD_{\mathbb{H}}(h(\mathbf{P}_{i,j}), h(\mathbf{P}'_{i,j})), \quad (6)$$

where s is a scaling parameter mapping the correlation score to $[0,1]$.

Figure 2 gives an example of the correlation map of one single patch. We can see that the Hamming distance of hash codes (Figure 2(b)) approximates the Euclidean distance (Figure 2(a)) well, especially in the high correlation score regions.



(a) Original Correlation Map (b) LSH Correlation Map

Fig. 2. Correlation map from DM and LSH-DM. Red areas represent high correlation scores. The correlation maps generated by Euclidean distance of SIFT descriptors and Hamming distance of hashing codes are similar.

The Box-Muller[21] method is used to generate a projection matrix M. Each row of M is $r_i \sim \mathcal{N}(0, I)$. Once the projection matrix is generated, patches from both source and target images are hashed using LSH (refer to Section 2.2). The hash code length K balances the trade-off between complexity and accuracy. In the experiments we observed that good trades-off were achieved at code length of $128 < K < 1024$. We use $K = 512$ in the rest of the paper. The other steps of the proposed method are the same as DeepMatching.

The proposed LSH-DM algorithm is summarized in Algorithm 1.

Algorithm 1 LSH-DM

Input: I_0, I_1

1: **Extract patch features** $\mathbf{p}, \mathbf{p}'$
2: **Generate Projection Matrix** $\mathbf{M}$
3: **Hash source and target patches** (Eq. 3)
4: **Compute correlation maps** (Eq. 6)
5: $N \leftarrow 4$
6: **While** $N < \max(W, H)$ **do**
  Aggregate patches (Eq. 2)
  $N \leftarrow 2N$
7: **Backtracking correspondences to the bottom level**

3.2. Complexity Analysis

DeepMatching computes a correlation map for every single patch from the source image. Suppose that the source image and the target image have the same size of pixel N. The correlation maps consist of $O(N^2)$ feature correlation scores, which are the Euclidean distances of features in $\mathbb{R}^d$ with d denoting the descriptor length. The time cost is approximately $O(N^2T)$, where T is the consuming time for computing a d-dimensional inner product (dot product).

As for LSH-DM, we first generate a random $d \times K$ projection matrix, which has negligible time cost. Then we hash both the source and target image patches with $O(KNT)$ complexity. The correlation scores in LSH-DM are computed by $O(N^2)$ Hamming distances. Note that in modern CPUs, computing Hamming distance for 128-bit hash codes in Hamming space costs only one CPU cycle, which is hundreds of times faster than computing Euclidean distance. In fact, it is easy to realize ten times or more acceleration with LSH-DM for high-resolution image pairs, without much loss of accuracy.

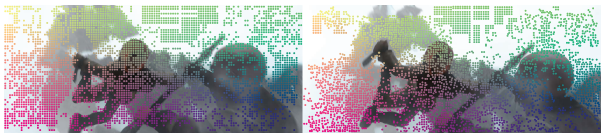
4. EXPERIMENT

The proposed LSH-DM algorithm was evaluated using the MPI-Sintel dataset[2] and the KITTI 2012 dataset[3]. All experiments are carried out on a desktop PC with i7-4790K CPU and 128 GB memory, Windows 7 64-bit operating system. No parallel computation is used. For DeepMatching, we slightly modified the code from the author's website to adapt for our operating system. The LSH-DM algorithm is also implemented using C++ for fair comparison. For the optical flow

algorithm, we use the Epic-flow method[8], which is a state-of-art method for interpolating matches to obtain dense flow fields.

4.1. Qualitative Evaluation

Figure 3 shows a visual comparison of the DeepMatching and LSH-DM matching result. Both methods produces quasi-dense matches. The matches from LSH-DM are generated at different positions, yet the density and accuracy of the matches are similar compared with DM.



(a) DM Result

(b) LSH-DM Result

Fig. 3. Example of DM and LSH-DM results on consecutive frames from the MPI-Sintel dataset. LSH-DM generate different, yet accurate matches with similar density.

Figure 4(c) and (d) further shows the optical flow results by *DM+Epic* and *LSH-DM+Epic*, respectively. Both methods first obtain image matches, either by DM or LSH-DM, then perform Epic[8] to compute the optical flow fields. We can see that both methods generate high quality flow fields. The influence of different matches and matching accuracy on optical flow estimation is beyond the scope of this paper. LSH-DM speeds-up the matching procedure by ten times or more, while the flow field by *LSH-DM+Epic* has comparable accuracy.

(a) Source Image

(b) Target Image

(c) DM+Epic Result

(d) LSH-DM+Epic Result

Fig. 4. Example of DM+Epic and LSH-DM+Epic optical flow results on MPI-Sintel dataset.

Method	matches	Acc@10	AEE	Speed-Up
DM[14]	5920	0.892	3.686	-
LSH-DM	5508	0.866	3.709	**12.1063x**

Table 1. Experimental results on MPI-Sintel test dataset

Method	matches	Acc@10	AEE	Speed-Up
DM[14]	5357	0.856	3.334	-
LSH-DM	5084	0.827	3.361	**11.5569x**

Table 2. Experimental results on KITTI 2012 dataset

4.2. Quantitative Evaluation

Following the DeepMatching framework, we use an objective measure "accuracy@T" defined as the proportion of correctly matched pixels from the first image with respect to the total number of pixels. A pixel is considered correctly matched if its match in the second image is close enough to (i.e. less than T pixels away from) ground-truth. In the experiments, we use a threshold of $T = 10$. For evaluating optical flow results, we use the average end-point error (AEE).

Table 1 and 2 enumerates number of matches, matching error, optical flow error and Speed-Up for LSH-DM compared with DeepMatching, on the MPI-Sintel dataset and the KITTI 2012 dataset, respectively. Experiments show that LSH-DM can effectively produce matches with high quality for optical flow estimation. A speed-up of more than ten times is easily achieved. Speed boosting by hashing methods often results in performance deterioration, which has been reported in many other applications such as nearest neighbor searching. However, our proposed LSH-DM not only significantly improves matching speed, but also ensures the matching performance, especially for optical flow estimation, with only less than 1% drop of performance.

5. CONCLUSION

Recent optical flow methods use feature matching methods to find initial image correspondences to guide optical flow. DeepMatching is one of the state-of-art matching algorithms frequently used. However, the correlation map computation step in DeepMatching is time-consuming. A new algorithm, LSH-DM, is proposed to address the problem. LSH-DM employs Locality Sensitive Hashing (LSH) to DeepMatching to efficiently compute the correlation map. The computational complexity is greatly reduced. Experiments show that DeepMatching can be accelerated by the proposed approach in ten times or more, while retaining comparable accuracy for optical flow estimation. Future work includes incorporating supervised hashing method in DM, as well as algorithm parallelization on modern GPUs.

6. REFERENCES

[1] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, 2011.

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *European Conf. on Computer Vision (ECCV)*, 2012.

[3] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 2012, pp. 3354–3361.

[4] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Computer Vision-ECCV 2004*, pp. 25–36. Springer, 2004.

[5] Christopher Zach, Thomas Pock, and Horst Bischof, "A duality based approach for realtime tv-l 1 optical flow," in *Pattern Recognition*, pp. 214–223. Springer, 2007.

[6] T. Brox and J. Malik, "Large displacement optical flow: descriptor matching in variational motion estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 500–513, 2011.

[7] Jim Braux-Zin, Romain Dupont, and Adrien Bartoli, "A general dense image matching framework combining direct and feature-based costs," in *International Conference on Computer Vision (ICCV)*. IEEE, 2013.

[8] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid, "EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow," in *Computer Vision and Pattern Recognition*, 2015.

[9] Marius Leordeanu, Andrei Zanfir, and Cristian Sminchisescu, "Locally affine sparse-to-dense matching for motion and occlusion estimation," in *Proceedings of the 2013 IEEE International Conference on Computer Vision*, 2013, ICCV '13, pp. 1721–1728.

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

[11] Zhuoyuan Chen, Hailin Jin, Zhe Lin, Scott Cohen, and Ying Wu, "Large displacement optical flow from nearest neighbor fields," in *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2013, CVPR '13, pp. 2443–2450.

[12] Radu Timofte and Luc Van Gool, "Sparse flow: Sparse matching for small to large displacement optical flow," in *WACV*, 2015.

[13] O. U. N. Jith, S. A. Ramakanth, and R. V. Babu, "Optical flow estimation using approximate nearest neighbor field fusion," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 673–6577.

[14] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid, "Deepmatching: Hierarchical deformable dense matching," *International Journal of Computer Vision*, vol. 120, no. 3, pp. 300–323, 2016.

[15] Philippe Weinzaepfel, Jerome Revaud, Zaid Harchaoui, and Cordelia Schmid, "Deepflow: Large displacement optical flow with deep matching," in *IEEE Intenational Conference on Computer Vision (ICCV)*, 2013.

[16] Aristides Gionis, Piotr Indyk, and Rajeev Motwani, "Similarity search in high dimensions via hashing," in *Proceedings of the 25th International Conference on Very Large Data Bases*, 1999, VLDB '99, pp. 518–529.

[17] Piotr Indyk and Rajeev Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, 1998, STOC '98, pp. 604–613.

[18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[19] Moses S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of the Thiryfourth Annual ACM Symposium on Theory of Computing*, 2002, STOC '02, pp. 380–388.

[20] Michel X. Goemans and David P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," *J. ACM*, vol. 42, no. 6, pp. 1115–1145, Nov. 1995.

[21] G. E. P. Box and M. E. Muller, "A note on the generation of random normal deviates," *Annals Math. Stat.*, vol. 29, pp. 610, 1958.