Improved Cost Computation with Local Binary Features in a Multi-View Block Matching Framework

K. Valentín, S. Štolc, R. Huber-Mörk

AIT Austrian Institute of Technology GmbH, Intelligent Vision Systems
2444 Seibersdorf, Austria
Email: kristian.valentin.fl@ait.ac.at

Abstract. *We propose a methodological improvement to multi-view stereo / light-field block matching algorithms based on local binary features such as Census Transform (CT), Local Binary Pattern (LBP), etc. Instead of interpolating individual binary descriptors before matching, we propose to carry out the necessary interpolations involved in cost estimation, only after matching in the cost domain. This methodological twist offers a substantial performance increase for both CPU as well as GPU architectures, while delivering the same output quality. The proposed algorithm is analyzed from theoretical as well as practical viewpoints and its performance is compared to a naïve approach with binary interpolations.*

Keywords: Multi-view stereo matching, 3-D reconstruction, Light fields, Computational imaging

1. Introduction

Various methods to measure depth information by computer vision systems exist, e.g., conventional stereo vision, light-field/multi-view stereo, laser triangulations, time-of-flight sensors, etc. In this paper, we focus on 3-D sensing using a multi-view imaging system realized by a *multi-line scan light-field camera* [1].

In general, a *light field* is defined as a 4-D radiance function that describes the intensity of light passing through every point in space (free from occluders) in every direction [2]. Among other things, such light fields can be exploited for computational imaging, such as refocusing, changing depth of field, correcting optical aberrations, all-in-focus imaging, noise reduction, as well as 3-D reconstruction. Without loss of generality, in this paper we restrict ourselves to 3-D light fields, however, our proposed method can be used equally well in algorithms dealing with complete 4-D light fields.

As for the 3-D reconstruction approach, we consider a multi-view stereo matching algorithm described in [1] that operates in the EPI domain [3]. This algorithm requires a number of interpolation steps between image pixels in order to obtain cost values for tested disparity hypotheses. Due to involvement of multiple views, interpolations become necessary even when the disparity steps between the two extreme views are restricted to integer values.

In order to make stereo matching robust against local instabilities in the image intensity domain, many methods make use of *local binary features* (LBFs) which are afterwards compared by means of the *Hamming distance* [4]. LBFs are a class of texture operators used in computer vision that describe local image neighborhoods by means of binary vectors comprised of bits obtained by individual pixel comparisons within that neighborhood.

Well-known examples of LBFs are *Census Transform* [5] and *Local Binary Pattern* [6] operators. There are many applications of these descriptors in computer vision as they have certain computational and performance advantages over other matching approaches. On one hand they represent very compact and efficient approach to describing local image neighborhoods, and on the other hand they are exceptionally robust against many common image perturbations (e.g., locally non-constant illumination).

Nevertheless, the stated computational advantage of LBFs becomes less prominent when used in a multi-view stereo setup, where sub-hypothesis matching is an inevitable step requiring interpolation between descriptors.

In this paper, we propose a methodological improvement for efficient interpolation in the aforementioned domain. Considering a large number of interpolations that have to be performed in the course of 3-D reconstruction, we show that the proposed improvement has a significant impact on the overall performance of the matching algorithm.

2. Subject and Methods

When an object is acquired by the multi-line scan light-field camera, it is transported in front of the camera at a constant speed and direction. In each time instance, multiple lines are rapidly extracted from a CMOS area-scan sensor, which are then used to construct different views of the acquired object. Throughout this process a 3-D light field is being constructed.

The acquired 3-D light field is used to derive the depth information using a multi-view block matching algorithm [1] that makes use of LBFs. A very important part of this matching algorithm is the computation of cost estimates for each pixel of the reference view w.r.t. tested disparity hypotheses. Let d be the number of tested hypotheses, v the number of light-field views and w and h width and height of each light-field view, respectively, then in the course of all cost computations the upper bound N for the number of operations that require interpolations between descriptors is given as:

$$N = d\,(v-1)\,wh. \tag{1}$$

In practice, N can be lower because there are cases where the interpolation is in fact not necessary (e.g., when testing the zero disparity or when the tested disparity slope produces an integer shift in a certain view). Despite there is only a linear relationship between N and the other variables, the dependence on the number of pixels causes N to rise quickly (e.g., for 9 views of 512×512 px and 20 tested disparity hypotheses, N is already as high as 41.9 million). Thus, an effective cost computation may significantly improve the performance of the whole matching algorithm.

3. Proposed Method

The main operation in the cost computation is to evaluate the dissimilarity between a reference binary descriptor $\mathbf{r} \in \{0,1\}^n$ extracted from the reference view and an interpolated real-valued descriptor calculated from two binary descriptors $\mathbf{x},\mathbf{y} \in \{0,1\}^n$ corresponding to two neighboring descriptors extracted from some other view. A conventional approach to computing this quantity would be to interpolate between $\mathbf{x}$ and $\mathbf{y}$ followed by computation of the distance between $\mathbf{r}$ and the interpolated descriptor.

We propose an alternative approach to this task, that is computationally much less expensive and, furthermore, leads to the same numerical results. The main idea behind is to compute distances between $\mathbf{r}$ and both $\mathbf{x}$ and $\mathbf{y}$ separately, followed by interpolation between the two computed distances. In this way, one avoids the explicit computation of the interpolated descriptor which is the most complex task in the entire cost calculation.

Stereo matching algorithms based on LBFs typically evaluate the dissimilarity between binary descriptors by means of the Hamming distance. The Hamming distance between two binary vectors $\mathbf{x}$ and $\mathbf{y}$ is defined as follows:

$$H(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} x_i \oplus y_i, \tag{2}$$

where $\oplus$ is the XOR operator. Allowing for the interpolation between binary vectors, the resulting vector is no longer binary but rather real-valued with elements $\in [0,1]$. In order to measure distances between interpolated vectors, a generalized Hamming distance is used:

$$\widehat{H}(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n} |x_i - y_i|, \tag{3}$$

where $\mathbf{x},\mathbf{y} \in [0,1]^n$. Note that for strictly binary vectors $H(\cdot)$ and $\widehat{H}(\cdot)$ are equal.

The main idea of the proposed improvement to LBF-based block matching algorithms is to perform a computationally cheap interpolation between two ordinary Hamming distances, instead of costly interpolating between binary vectors and calculating generalized Hamming distance afterwards. Formally this can be expressed as follows:

$$\widehat{H}(\mathbf{r},\ w\,\mathbf{x} + (1-w)\,\mathbf{y}) = w\,H(\mathbf{r},\mathbf{x}) + (1-w)\,H(\mathbf{r},\mathbf{y}), \tag{4}$$

where $\mathbf{r}$ is a reference binary vector and $\mathbf{x}$ and $\mathbf{y}$ are binary vectors to be interpolated by a weight $w \in [0,1]$. For the sake of simplicity, we restrict ourselves to linear interpolation in this paper. However, a similar approach should be applicable to other interpolation schemes as well. To show the equivalence of Eq. (4), we conduct the following reasoning:

$$\begin{aligned}
\widehat{H}(\mathbf{r},\ w\,\mathbf{x} + (1-w)\,\mathbf{y}) &= \sum_{i=1}^{n} |r_i - [w\,x_i + (1-w)\,y_i]| \\
&= \sum_{i=1}^{n} |w\,(r_i - x_i) + (1-w)\,(r_i - y_i)| \\
&= \sum_{i=1}^{n} \begin{cases} w\,(r_i - x_i) + (1-w)\,(r_i - y_i) & \text{if } r_i = 1 \\ w\,(x_i - r_i) + (1-w)\,(y_i - r_i) & \text{if } r_i = 0 \end{cases} \\
&= \sum_{i=1}^{n} \big[w\,|r_i - x_i| + (1-w)\,|r_i - y_i|\big] \\
&= w\,\sum_{i=1}^{n} |r_i - x_i| + (1-w)\,\sum_{i=1}^{n} |r_i - y_i| \\
&= w\,\widehat{H}(\mathbf{r},\mathbf{x}) + (1-w)\,\widehat{H}(\mathbf{r},\mathbf{y}) = w\,H(\mathbf{r},\mathbf{x}) + (1-w)\,H(\mathbf{r},\mathbf{y}). \quad \square
\end{aligned} \tag{5}$$

Advantages of the Proposed Approach

There are three main advantages of the proposed approach over the naïve method: (i) distance is computed between two binary vectors, which can be done more efficiently than between a binary and a real-valued vector, (ii) compared to *n* interpolations in the naïve approach, there is only one scalar interpolation required, (iii) there is no need to unpack the individual bits from the descriptors to compute either interpolation or distance as it is in the naïve approach.

Implementation of the Proposed Approach

The ordinary Hamming distance between two binary vectors can be efficiently computed as a combination of a vector XOR operation followed by a set-bit-count function (also called *popcount* or *sideways sum*). Both these operation are usually available on most common CPU and GPU architectures. In the naïve approach, there is no such architectural support for binary vector interpolation and so each bit has to be algorithmically unpacked using a combination of bit-wise shift and AND operations. On physical platforms, some of the operations listed in Tab. 1 can be implemented using vector instructions, usually as multiples of 32/64-bit variants. Thus, the performance does not scale proportionally to the descriptor length in practice.

Table 1: Number of operations by type in tested implementations of the cost computation for one pixel and one disparity, where n is the length of the descriptor. These numbers hold for cases where $n \leq$ size of processor vector operations (typically 32 or 64 bits).

	Add	Sub	Mul	Abs	Bitshift	`AND`	`XOR`	Popcount
Naïve approach	$2n$	$2n$	$2n$	n	$3n$	$3n$	–	–
Proposed approach	1	1	2	–	–	–	2	2

4. Results and Conclusions

In this study, we proposed a methodological improvement to the light-field / multi-view stereo block matching algorithms based on local binary features. The proposed as well as the naïve cost computation methods were implemented on CPU and GPU architectures. For the Hamming distance computation, we exploited intrinsic `XOR` and set-bit-count operations. An overview of the executed operations are listed in Tab. 1. In the case of naïve method, the interpolated descriptors were computed at the same time with the distance computation in order to save memory. In both cases, we considered 32-bit descriptors.

The implementation for CPU architecture was done in C++, compiled using Microsoft Visual Studio 2013 and ran on a single core of a quad-core Intel Xeon E5 processor. We used the `std::bitset` class to represent LBF descriptors. As a result for we obtained a speedup factor as high as 20 between the proposed and the naïve methods. On the other hand, the GPU implementation was done in CUDA C and compiled using the CUDA 6.5 compiler. In this case, 32-bit unsigned integers were used to represent LBF descriptors. On a GeForce GTX TITAN graphics card we achieved a speedup factor of 13. Although these numbers do not scale as much as might have been expected from Tab. 1, they still represent a very high performance improvement for both tested architectures and therefore have great potential for practical applications.

As for the future research, we plan to investigate into LBF descriptors whose length is not strictly dependent on the size of the matching window. Such descriptors might prove useful for better conformity with the specifics of particular hardware architectures (i.e., vector operations, memory alignment, etc.), which is essential for further performance improvements as well as for enabling this type of algorithms on more limited embedded platforms.

References

[1] Štolc S, Soukup D, Holländer B, Huber-Mörk R. Depth and all-in-focus imaging by a multi-line-scan light-field camera. *Journal of Electronic Imaging* 23(5):053020, 2014.

[2] Levoy M., Hanrahan P. Light field rendering. In: *Proc. of Conference on Computer Graphics and Interactive Techniques*. New York, NY, USA, pp. 31–42, 1996.

[3] Bolles RC, Baker HH., David, Marimont H. Epipolarplane image analysis: an approach to determining structure from motion. *International Journal of Computer Vision* 1(1): 7–55, 1987.

[4] Hamming R. Error detecting and error correcting codes. *The Bell System Technical Journal* 29(2): 147–160, 1950.

[5] Zabih R, Woodfill J. Non-parametric local transforms for computing visual correspondence. In: Eklundh, J.-O. (ed.) *Computer Vision – ECCV 94*, Vol. 801 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, pp. 151–158, 1994.

[6] Pietikäinen M, Ojala T. Texture analysis in industrial applications, 1996.