Stereo Reconstruction : Comparison and Analysis

Chang Luo
chang.luo@tum.de

Sebastian Bauer
ge86joc@tum.de

Abstract

Inspired by the human visual system, the contemporary computer vision solutions raised their performance highly. One of the important examples is stereo reconstruction which defines variety of methods in order to create a precise 3D visual understanding of the environment. In this paper, we implement different ways of extracting feature points and using different dense matching methods to reconstruct a 3D scene and the performance of them will be compared and analyzed.

1. Introduction

The reconstruction of 3D scenes on the basis of 2D images represents an important building block for numerous applications, such as autonomous driving or robotics. Stereo vision is now a field where the goal is to reconstruct those 3D scenes based on two images that represent this scene in different views with a transitional baseline. Although great progress has been made in the past years, it is still the subject of active research.

In our project, we analyzed various the performance feature detector and stereo matching methods. For this purpose, we first applied different feature descriptor to both views. For checking the similarity of descriptors, the FLANN-based matcher is used to deriving the best corresponding point set.

As the matches in two views are given, we can compute the rotation and translation between two frames by using the 8-point algorithm. After that, it is possible to rectify the images and thus transform them on a common plane. Subsequently, it is possible to generate a disparity map using different block matching methods. Afterwards, we can compute a depth map which is then used in the last step for the mesh reconstruction.

2. Related Work

Feature Detector. There are several mature feature detection algorithm available and well-packaged in the OpenCV library. The very common one of the algorithms is SIFT (Scale Invariant Feature Transform) which extracts the local feature and detect it in Gaussian scale space [9]. Speeded Up Robust Features (SURF) can be seen as an improved version of SIFT, which reduces the computational effort by reducing feature descriptor dimensions and introducing an integral image [4]. While ORB concentrates on algorithm speed-up and get rid of creating the scale space like SIFT [13], Binary Robust invariant scalable keypoints (BRISK) aims to identifying the characteristic direction of features for pose retrieving [8]. Besides, there are also two KAZE descriptor included in OpenCV library, KAZE and the accelerated version AKAZE. KAZE detector is considered as a variant of SIFT, the distinct feature is the detection will happen in non-linear scale space [1, 2].

Correspondence finding. The fetched feature descriptors should be matched into pairs for getting the correspondence in two views. For this purpose, fast nearest neighbour search method by FLANN library performs more efficient for high dimensional vectors [11].

Sparse matching. Given at least five corresponding point matches and camera intrinsic parameters, using epipolar constraint to derive the essential matrix [12]. Decomposition of the resulting essential matrix derives four combinations of potential rotation matrix and translation vector [3]. After validation, only an unique combination should represent the real motion. Since the camera motion is given, Bundle Adjustment algorithm can be further applied for tuning the derived pose. The re-projection error between two frames will be minimized for a preciser result [15].

Dense matching. Global dense matching methods not only take into account the immediate neighborhood of a pixel but also all information available in an image [6]. Thus, they're expensive to compute and not suitable for

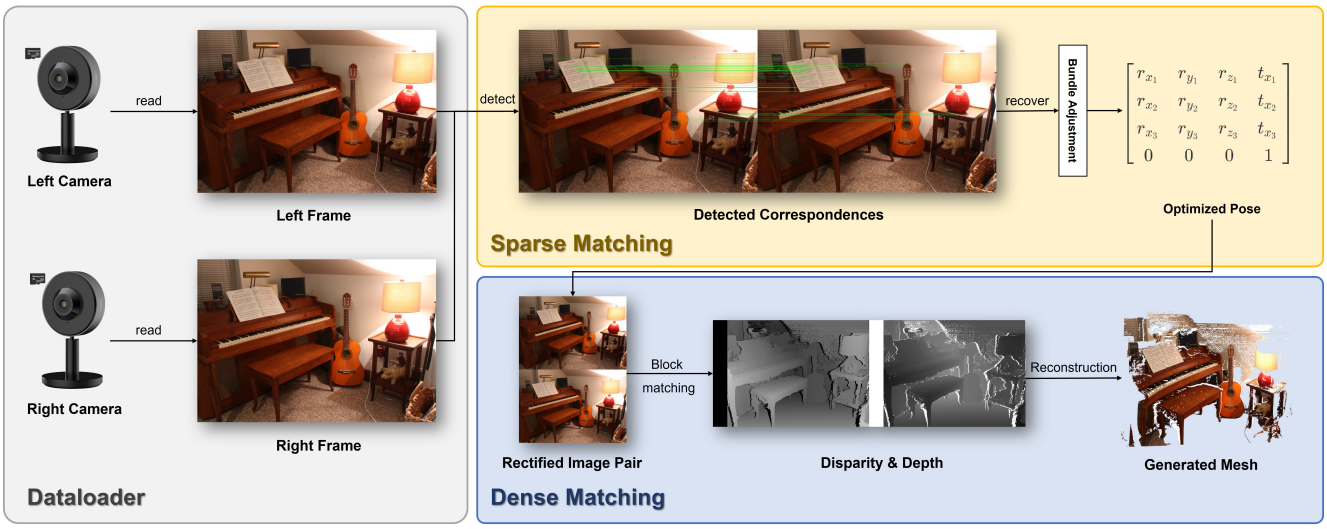

Figure 1. The overview of the stereo reconstruction pipeline.

real-time applications. To speed up the calculation, simple local block matching algorithms can be applied. However, the disadvantage here is that these often achieve poor results. SGBM [7] tries to ensure a fast calculation by taking semi-global properties into account.

3. Method

3.1. Dataset

The Middleburry-2014 dataset [14] is used in this project, which provides high-resolution rectified images and highly accurate ground truth disparity maps. We select the Piano-perfect image pair as the basis for the comparison work.

3.2. Feature Descriptor

There are several feature descriptors choosen from OpenCV library, including the aforementioned SIFT, ORB, SURF, KAZE, AKAZE, and BRISK. The descriptors which perform detection on scale space are all initialized with four octave layer for sake of fairness, other parameters are not changed and the default suggestion value from OpenCV documentation is used for initialization. Corresponding points are then matched using the FLANN-based matcher which internally conducts an optimized k-nearest neighbour search.

By sorting the found matches by distance also can be interpreted as the similarity of two points, we are able to get a specific number of good matches and averaging the distance between pair match, which as be taken as the precision metric for performance analysis. Besides, the elapsed running time and number of found matches will be also considered as a criterion in comparison.

3.3. Sparse Matching

Multiplication of inverse intrinsic camera parameters with corresponding points in image space of each images gives points in camera space. This process also called deprojection. Considering epipolar constraint,

$$0 = q_2^T E q_1 \tag{1}$$

Since (1) can be rewrited as a product of E and kronecker product of $q1$ and $q2$, denoted as $\tilde{q}$, and stack E as a vector, denoted as $\tilde{E}$. Hence,

$$0 = \tilde{q}^T \tilde{E} \tag{2}$$

We can use SVD-decomposition or QR-factorisation to derive the four stacked vectors $\tilde{X}, \tilde{Y}, \tilde{Z}, \tilde{W}$ which spans the null space of equation (2). The four vectors can be restructured to 3×3 matrices which form essential matrix by the combination of some scalars a, b, c, d in (3).

$$E = aX + bY + cZ + dW \tag{3}$$

In the case of more that five points, the four singular vectors that correspond to the four smallest singular values are considered.

$$0 = EE^T E - \frac{1}{2} trace(EE^T)E \tag{4}$$

Putting the essential matrix equation derived by the four vectors and four scalars into constraint (4), we are able to obtain the essential matrix.

Project onto essential space. Compute the SVD $E = Udiag(\sigma_1, \sigma_2, \sigma_3)V^T$. Since in the reconstruction, E is only defined up to a scalar, we project E onto the

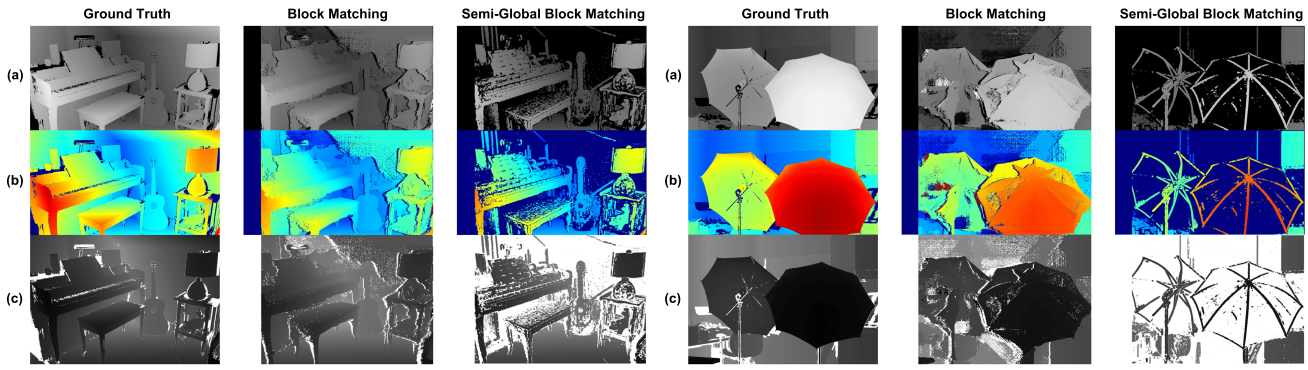

Figure 2. Disparity and depth map for the Pianno-perfect and Umbrella-perfect. (a) Disparity map, (b) JET-colored disparity map, (c) Depth map.

normalized essential space by replacing the singular values $\sigma_1, \sigma_2, \sigma_3$ with $1, 1, 0$. Afterwards, we are allowed to calculate the pose R and $\hat{T}$ like equation (5).

$$R = UR_Z^T(\pm\frac{\pi}{2})V^T, \quad \hat{T} = UR_Z(\pm\frac{\pi}{2})\Sigma U^T \tag{5}$$

The obtained two potential rotation matrices and two translation vectors are validated by cheirality constraint [12] which fundamentally retain the only true configurations which corresponds to positive depth values. After the validation operation we end up with a true rotation matrix and a true translation vector.

Due to the found corresponding points could be vague or mismatched, we will apply the Bundle Adjustment algorithm to optimize the recovered pose. In equation (6), x_i refers to the observerd pixel position in left frame, and x_i' for right frame. $\hat{x_i}$ and $\hat{x_i'}$ represents to the re-projected position of 3D point into both frame. In our case, the 3D point are initialized with backprojection of left frame, and the depth value is set to 1 for convenience. The g2o library provides a well-defined non linear optimization which is used in our project for carrying out the optimized pose.

$$E_{re-proj} = \sum_{i}^{N} d(x_i, \hat{x_i}) + d(x_i', \hat{x_i'}) \tag{6}$$

Considering the Middlebury 2014 dataset are all rectified views, the result should reflect a identity rotation matrix $R \in SO(3)$, for translation a baseline shift in x-direction should be observed. As the ground truth of $[R|T]$ now is clear, we can use it for further metric analysis.

3.4. Dense Matching

After stereo rectification we performed stereo correspondence matching, i.e. dense matching, that is to find similar regions between the left and right rectified images. From this, one can calculate a disparity map with:

$$\text{disp} = x_l - x_r \tag{7}$$

where x_r is the best fit in the right image for the pixel x_l in the left image. After that, the depth for a pixel is given by

$$d_j = \frac{b + f}{\text{disp}_j} \tag{8}$$

where b is the baseline and f the focal length. For finding the best match we use two matching methods: one is operating only locally around a neighbourhood of the pixel (BM). The other matching algorithm SGBM applies not only in local area, but it will also take global effect into account when trying to find the best match [7].

3.4.1 Local Block Matching (BM)

In local Block Matching one has to define a block B with size $w \times w$ over which the similarity of two parts of the left and right image are measured. Then an error function needs to be defined which evaluates the corresponding matching costs for two blocks. In our case we used the sum of absolute differences (SAD) which is for two B_1, B_2 defined as

$$SAD(B_1, B_2) = \sum_{x=1}^{w}\sum_{y=1}^{w} |B_1(x, y) - B_2(x, y)|. \tag{9}$$

For each pixel x_l in the left image and its corresponding block B_l, the best match x_r^* is the one which yields the smallest SAD value in the right image in the same line as x_l.

3.4.2 Semi-global Block Matching (SGBM)

The disadvantage of local matching methods is that due to the small search space, discontinuities can occur (e.g. at

edges), which can significantly worsen the calculation of the disparity map [10]. In SGBM, the disparity values are now calculated in a similar way to BM. For a given pixel in the left image with coordinates (x, y), we calculate disparities d up to a maximum value d_{max}. Here, also a cost function is used which determines how good a match is:

$$C(d) = |I_l(x, y) - I_r(x - d, y)| \tag{10}$$

In the original SGBM paper [7] a special matching cost is used which takes the illumination change into account and is more robust than e.g. SAD. This is also the matching cost we use in our implementation. However, in contrast to local block matching, not only the disparity in the respective row is considered, but also the disparity value of the surrounding pixels. In order to reduce the computation time, however, SGBM only considers one dimensional paths in different directions.

3.5. Mesh Reconstruction

After computation of the depth map, a point cloud was calculated and afterwards triangulation was used to compute the final output mesh.

4. Results

4.1. Sparse Matching

Figure 3 shows the matching result with different detectors, only best 40 points are drawn. We can find that the result of

Figure 3. Good matches with various feature descriptors on Piano-perfect image pair.

ORB and SURF are concentrated on a specific area, which may cause a performance degradation in sequential steps.

4.2. Dense Matching

Figure 2 presents the result of BM (Block Matching) and SGBM (Semi-Global Block Matching), since the dataset is already rectified hence we can get rid of influence taken in Sparse Matching stage. The ground truth is also included for comparison. While the BM disparity and depth maps show large gaps and incontinuities, especially around edges, SGBM shows a smoother and more stable result. This can be traced back to the non-local locals calculations that SGBM does.

4.3. Reconstruction

Figure 4. Reconstructed mesh with BM and SGBM.

Figure 4 shows the final reconstruction mesh from depth map calculated by BM as well as SGBM. In case of BM, One can clearly see that due to the bad depth map computation, large parts of the mesh are missing. Compared with BM one, the one generated by SGBM is much better.

5. Comparison

5.1. Comparison: Feature descriptor

Table 1 shows the comparison of different feature descriptors, we can observe the KAZE and SURF outperforms other descriptors on precision, but KAZE shows a severe drawback at running speed. Considering all these three metrics we decided to use SURF as the feature descriptor to perform the description task. When the matches is given, we now start to use them as the basis of pose recovering, which will yield the estimated R and T for next step. We take ground truth rotation R_g and translation T_g, and deduct the derived one for computing errors.

$$R_g = I(3), \quad T_g = [baseline, 0, 0] \tag{11}$$

Rotation R_g can be converted to Euler angles $\alpha, \beta, \gamma = 0$ for sake of error computation. We hence use the derived pose subtract with the ground truth to get the error values.

	SURF	SIFT	BRISK	KAZE	AKAZE	ORB
N	16817	15064	6750	3662	3520	500
D	0.02	26.98	75.11	0.026	72.99	123.03
T	1.012	1.896	0.294	4.697	0.828	0.351

Table 1. Feature descriptor comparison on Piano-perfect, **N** = Number of Matches, **D** = averaged Distance and **T** = Elapsed running Time.

	SURF	SIFT	BRISK	KAZE	AKAZE	ORB
R / T	0.0041 / 0.14638	0.0016 / 0.14637	0.1610 / 0.14852	0.0048 / 0.14637	0.0017 / 0.14637	0.5294 / 0.15237
R / T (BA)	0.0013 / 0.14637	0.0007 / 0.14637	0.1807 / 0.14798	0.0003 / 0.14637	0.0020 / 0.14637	0.3418 / 0.15097

Table 2. Error in Pose R and T with Bundle Adjustment and without Bundle Adjustment on Piano-perfect image pair, error of R in euler angles and T in meters.

Table 2 shows the error of 5-point method calculated R and T, and compared with the optimized one with BA algorithm. It's clearly to see that BA algorithm has almost no effect with transition, but helps the rotation correction. Concerning about feature descriptor, we can find that SIFT and KAZE descriptor has a impressive performance. Besides, BA algorithm fails to optimize with some descriptors like BRISK or AKAZE, we assume it could be caused by some deteriorated point pairs.

5.2. Comparison: BM/SGBM

In order compare BM and SGBM we used different error and accuracy metrics that we calculated using a ground truth disparity map. One error function is the root mean squared,

$$RMS(D,G) = \sqrt{\frac{1}{D_w D_h}\sum_{x=0}^{w}\sum_{y=0}^{h}(D_{x,y} - G_{x,y})^2} \quad (12)$$

which measures the mean difference between the calculated disparity D and the ground truth G with size $D_w \times D_h$. Another function which does not determine the error rate but the accuracy is BAD_α with $\alpha \in \{1, 2, 5\}$ which is defined as:

$$BAD_\alpha(D_{x,y}, G_{x,y}) = \begin{cases} 1, & \text{if } |D_{x,y} - G_{x,y}| > \alpha \\ 0, & \text{else} \end{cases} \quad (13)$$

and then

$$BAD_\alpha(D_{x,y}, G_{x,y}) = \frac{1}{D_w D_h}\sum_{x=0}^{w}\sum_{y=0}^{h} BAD_\alpha(D_{x,y}, G_{x,y}) \quad (14)$$

Pair	Piano-perfect		Umbrella-perfect	
Method	BM	SGBM	BM	SGBM
BAD-1	51.8462	15.8732	76.7501	18.0213
BAD-2	52.0104	16.748	76.9073	19.1471
BAD-5	52.2857	18.9823	77.1587	21.8861
RMS	90.5088	60.1341	133.462	55.843

Table 3. BM/SGBM performance compared to ground truth when computed on Piano -perfect and Umbrella-perfect stereo image pair.

for the whole disparity map. One can see from the Table 3, that in both cases SGBM clearly outperforms BM. The number of BAD-pixels of the piano SGBM disparity map is only approximately only $\frac{1}{3}$ of the BAD-pixels of the BM disparity map. One can verify this values also by looking at the Figure 5.

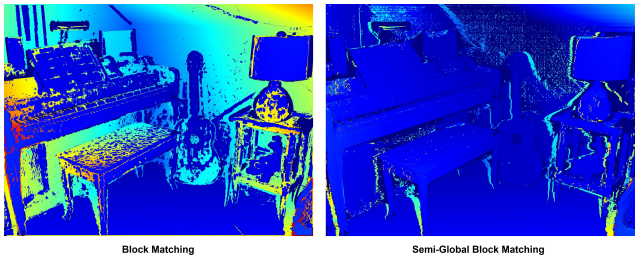

Figure 5. Error map of BM and SGBM.

6. Conclusion

In the sparse matching part, each feature descriptor has it's feature, ORB is aiming to find matches in fast manner, but the accuracy decreases. On the contrary, KAZE spend most of time and also get a precise solution. Taking all factors into consideration, SIFT seems to be a rule of thumb choice. As a For the dense matching part we compared local block matching using SAD with semi-global block matching. We concluded that even though SGBM gives much better results, its runtime is not suited for real-time applications. A further investigation into different similiarty measures for local block matching like sum of squared distances (SSD) or Normalized Cross Correlation (NCC) would yield additional insights. Also, incorporating deep learning techniques to learn the disparity map have also shown to deliver good results ([5]).

References

[1] Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J Davison. Kaze features. In *European conference on computer vision*, pages 214–227. Springer, 2012. 1

[2] Pablo F Alcantarilla and T Solutions. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *IEEE Trans. Patt. Anal. Mach. Intell*, 34(7):1281–1298, 2011. 1

[3] Alex Andrew. Multiple view geometry in computer vision20012richard hartley, andrew zisserman. multiple view

geometry in computer vision . cambridge: Cambridge university press 2000. xvi + 607 pp., isbn: 0-521-62304-9 hardback. *Kybernetes*, 30:1333–1341, 12 2001. 1

[4] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006. 1

[5] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. *CoRR*, abs/1803.08669, 2018. 5

[6] Rostam Affendi Hamzah, Rosman Abd Rahim, and Zarina Mohd Noh. Sum of absolute differences algorithm in stereo correspondence problem for stereo matching in computer vision application. In *2010 3rd International Conference on Computer Science and Information Technology*, volume 1, pages 652–657. IEEE, 2010. 1

[7] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008. 2, 3, 4

[8] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision*, pages 2548–2555. Ieee, 2011. 1

[9] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 1

[10] Matthias Michael, Jan Salmen, Johannes Stallkamp, and Marc Schlipsing. Real-time stereo vision: Optimizing semi-global matching. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 1197–1202, 2013. 4

[11] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, 2009. 1

[12] D. Nister. An eifficient solution to the five-point relative pose problem. *Proc. of CVPR*, 2:756–777, 01 2003. 1, 3

[13] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011. 1

[14] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nešić, Xi Wang, and Porter Westling. High-resolution stereo datasets with subpixel-accurate ground truth. volume 8753, pages 31–42, 09 2014. 2

[15] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999. 1