Learning-based Homography Matrix Optimization for Dual-fisheye Video Stitching

Mufeng Zhu
Rutgers University
Piscataway, NJ, USA
mz526@rutgers.edu

Yang Sui
Rutgers University
Piscataway, NJ, USA
yang.sui@rutgers.edu

Bo Yuan
Rutgers University
Piscataway, NJ, USA
bo.yuan@soe.rutgers.edu

Yao Liu
Rutgers University
Piscataway, NJ, USA
yao.liu@rutgers.edu

ABSTRACT

In this paper, we propose a novel feature-based video stitching algorithm for stitching back-to-back fisheye camera videos into one omnidirectional video in a video live streaming scenario. Our main contribution lies in a learning-based approach that refines the homography matrix in an online manner via gradient descent. The homography matrix is updated by training on a rolling dataset of feature points that are extracted and matched as new video frames are captured. Experimental results show that our method can create stitched images that better align matching features with lower mean squared error (MSE) than traditional feature-based stitching method. Furthermore, compared to vendor-supplied software (VUZE VR Studio) that uses calibration-based stitching, our method also produces visibly better results.

CCS CONCEPTS

• **Information systems → Multimedia streaming**.

KEYWORDS

fisheye, omnidirectional video stitching, feature extraction, homography matrix optimization

ACM Reference Format:
Mufeng Zhu, Yang Sui, Bo Yuan, and Yao Liu. 2023. Learning-based Homography Matrix Optimization for Dual-fisheye Video Stitching. In *Workshop on Emerging Multimedia Systems (EMS '23), September 10, 2023, New York, NY, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3609395.3610600

1 INTRODUCTION

Virtual reality (VR) technologies have rapidly developed in recent years, providing immersive virtual experiences for various applications such as entertainment, education, training, medicine, etc. An essential element of virtual reality is omnidirectional images and videos that can be viewed in any directions. To record omnidirectional content, cameras with at least two lenses are needed to capture the omnidirectional visual information from various directions. For example, a consumer-grade 360-degree camera includes two fisheye lenses. A further step, stitching, is needed to combine visual content captured by different lenses into one omnidirectional image/video frame. Stitching video content recorded by multiple lenses is particularly challenging. This is because objects can move between the boundaries of different lenses' field-of-view (FoV) over time, which makes visual artifacts even more prominent.

In this paper, we focus on stitching videos captured by dual-fisheye cameras such as Samsung Gear 360 and VUZE XR. Two main methods can be used for stitching back-to-back fisheye images: i) calibration-based stitching, and ii) feature-based stitching. Due to the property of rigid connection, a series of research studies focus on how to accurately calibrate two fisheye lenses and build correlated coordinates between them [12, 19]. Since calibration only needs to be performed once, the quality of stitched omnidirectional video highly depends on the accuracy of parameters obtained from calibration. If the calibration result is not sufficiently accurate, obvious misalignments can persist in the stitched video because of using a fixed transformation matrix during the stitching process, as shown in Figure 1. In addition, this misalignment cannot be eliminated since camera calibration parameters cannot be updated during video stitching.

Compared to calibration-based methods, feature-based methods are more flexible. Traditional feature-based methods extract feature points from the images and use approaches such as KNN [15] to match corresponding features. Then, through RANSAC [7], we can calculate the homography matrix that yields the highest proportion of inliers among matched features. With homography matrix, pixels in one image can be transformed to corresponding pixels in another image. With this method, the quality of stitched omnidirectional content depends on the accuracy of homography matrix. Traditional feature-based stitching methods, however, may not produce well-stitched results for dual-fisheye cameras. This is due to the limited overlapping region between two back-to-back lenses' FoVs. With limited overlapping region, only a limited number of features can be extracted, which leads to poor performance of the RANSAC algorithm. Additionally, extracting and matching features and calculating homography matrix on a per-frame basis will not only increase the computational requirements and take a long time, but

EMS '23, September 10, 2023, New York, NY, USA

ACM ISBN 979-8-4007-0303-4/23/09…$15.00
https://doi.org/10.1145/3609395.3610600

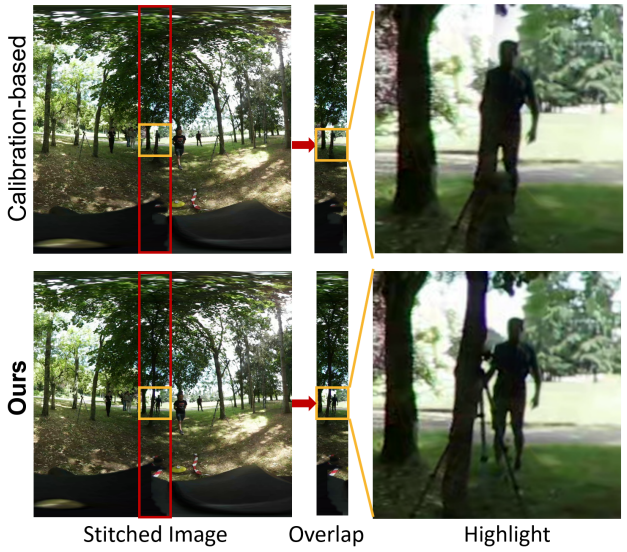

Figure 1: Visualization of the omnidirectional image stitched with a fixed extrinsic provided by the previous method, as outlined in FTV360 [14]. Despite capturing only a single person under a tree in the real world, the processed omnidirectional images unexpectedly blend tree and person, demonstrating misalignment and ghosting effect from the previous method.

can also lead to worse visual experience when there is a significant difference between homography matrices calculated from adjacent frames. As a result, misalignments and continuous jitter can be observed from stitched videos.

To address the issues with traditional feature-based stitching methods, in this paper, we propose a novel feature-based video stitching algorithm for back-to-back dual fisheye cameras in a low-latency live streaming scenario. Our main contribution lies in a learning-based approach that refines the homography matrix in an online manner via gradient descent. Our algorithm first unwarps raw fisheye frame, extracts, and matches feature points in the overlapping region, and uses RANSAC to calculate an initial homography matrix – as with traditional feature-based methods. We then create a rolling dataset of extracted and matched feature points that are considered "inliers" by RANSAC. As more dual-fisheye frames are captured over time, we update the dataset with inliers from newer frames. Periodically, with the updated dataset, we refine the homography matrix via a learning-based approach by training with the Adam optimizer [9] and mean squared error (MSE) loss. We implemented our proposed algorithm and compared it against both traditional feature-based method and calibration-based method used by the camera vendor-supplied software. Results show that our method can both visibly and quantitatively improve the quality of stitching compared to these approaches.

2 RELATED WORK

Fisheye Lens Projection Model. Before stitching, the first step is to unwarp the fisheye image/frame. Unwarping includes two steps: projection from fisheye image coordinates to 3D spherical coordinates and projection from 3D spherical coordinates to 2D rectangular image coordinates [2]. In this work, we choose the projection model proposed by Ho et al. [8]. First, for each point $P(x, y)$ in the raw fisheye image, we calculate its corresponding projected 3D coordinate as $P_{3D}(\cos\varphi_s \sin\theta_s, \cos\varphi_s \cos\theta_s, \sin\varphi_s)$ in the unit sphere. Here, θ_s and φ_s can be calculated as below:

$$\theta_s = f\frac{x'}{W} - 0.5, \qquad \varphi_s = f\frac{y'}{H} - 0.5$$

where f is the fisheye lens' field-of-view, and W and H are the width and height of the image. We can then transform 3D spherical coordinates to 2D rectangular image coordinates through the equirectangular projection. We calculate ρ and θ as below:

$$\rho = \frac{H}{f}\tan^{-1}\frac{\sqrt{(\cos\varphi_s \sin\theta_s)^2 + (\sin\varphi_s)^2}}{\cos\varphi_s \cos\theta_s}$$
$$\theta = \tan^{-1}\frac{\sin\varphi_s}{\cos\varphi_s \sin\theta_s}$$

Then the point on the equirectangular projection $P'(x', y')$ can be obtained as:

$$x' = 0.5W + \rho\cos\theta, \qquad y' = 0.5H + \rho\sin\theta$$

Feature Extraction. With good feature descriptors, high-quality matching pairs can be obtained. For example, ORB [17], SIFT [13] and SURF [3] are three feature descriptors widely used in image stitching. ORB combines FAST [16] feature and BRIEF feature descriptor [5]. It is an order of magnitude faster than SURF and over two orders of magnitude faster than SIFT [17]. Given our goal to achieve low-latency omnidirectional video stitching in the live video streaming setting, we use the ORB feature in our algorithm.

The Shi-Tomasi corner detection method [18] can detect robust corner features. It has similar time efficiency as ORB, which can also be used in our work. Therefore, we make comparisons between these two methods in this paper. We use the percentage of inlier points corresponding to the optimal matrix calculated by RANSAC as the metric. As illustrated in Table 1, experiment results show that the inlier percentage of ORB feature points is substantially higher than that of corner detection.

Multi-band Blending. Seamless blending is an indispensable step for both calibration-based and feature-based stitching methods. Without blending, users can observe two seamlines in the stitched image. In our work, we utilize multi-band blending [4] that can effectively avoid ghosting or unsmooth stitching. Besides, multi-band blending can be fully parallelizable, so we believe it can be successfully implemented in low-latency video stitching.

3 PROPOSED METHOD

In this paper, we propose a novel pipeline for omnidirectional video stitching, where the stitching quality can be gradually improved with the usage of Adam optimizer. The overall structure of the pipeline is illustrated in Algorithm 1. It includes three main components: i) feature extraction and matching, ii) homography matrix optimization, and iii) seamless image blending. The core idea of our pipeline lies in the second component. We describe details of these components below with a focus on our main contribution in learning-based homography matrix optimization.

Algorithm 1 Learning based dual-fisheye video stitching

```
i ← frame number
frame1 ← content recorded by one fisheye camera
frame2 ← content recorded by the other fisheye camera
for each recorded video frame do
    kps1 ← ORBdetect(frame1[overlap])
    kps2 ← ORBdetect(frame2[overlap])
    if this is the first frame then
        H, inliers ← RANSAC(kps1, kps2)
        Dataset.add(inliers)
    else if i%60 == 0 then
        if # of sampled frames in the dataset > 4 then
            Delete inliers collected from first sampled
            frame in the dataset
        inliers ← RANSAC(kps1, kps2)
        Dataset.add(inliers)
        Deduplicate(Dataset)
        Shuffle(Dataset)
        training_data, test_data ← divide(Dataset)
        H_Adam ← Optimze(Adam, H, training_data, test_data)
        loss_H ← MSE(H, test_data)
        loss_Adam ← MSE(H_Adam, test_data)
        if loss_H > loss_Adam then
            H ← H_Adam
    Stitch(H, frame1, frame2)
    Multiband_blending(frame1, frame2)
```

3.1 Feature Extraction and Matching

For each frame, we first unwarp the fisheye image. Since the FoV of fisheye lenses are known, we can calculate the overlapping regions of a pair of back-to-back fisheye images as: $W_{\text{overlap}} = \text{FoV}/180 * r$, where FoV is the field of view of the lens, r is the radius of the fisheye image, and W_{overlap} is the width of the overlapping region.

Therefore, instead of detecting feature points through the whole image (which is used in ordinary 2D image stitching where the overlapping area is unknown), we only need to detect feature points in the overlapping regions. Subsequently, we use template matching to match corresponding features. For each obtained feature point in a source image, we make it the upper left corner and create a rectangular template image with size of 60×16. We then search for the best matching (similar) part in a destination image. The similarity is defined by normalized cross-correlation, calculated as the following equation:

$$R(x,y) = \frac{\sum_{x',y'}(T(x',y') * I(x+x',y+y'))}{\sqrt{\sum_{x',y'} T(x',y')^2 * \sum_{x',y'} I(x+x',y+y')^2}}, \quad (1)$$

where $I(x,y)$ represents pixel at (x,y) in the source image, and $T(x,y)$ represents pixel at (x,y) in the template image, respectively. The part with the highest normalized cross-correlation in the destination image is considered the best matching part. Its upper left corner is then set as the correspondingly matched feature point of the feature point in the source image that generates the template. To filter out high quality matches, we consider matched pairs with normalized cross-correlation higher than 0.9 to be good matches which will be subsequently used in RANSAC.

3.2 Rolling Dataset Generation

As we mentioned above, good matches are used to calculate the optimal homography matrix through RANSAC. Meanwhile, according to the principle of RANSAC algorithm, we can obtain inliers which consist of matches that best satisfy the current optimal homography matrix. In our learning-based optimization, we treat the homography matrix as a fully connected layer with inputs and outputs both of length 3 in homogeneous coordinates (these homogeneous coordinates are converted feature point coordinates), and we construct the dataset using homogeneous coordinates of inliers. We set feature points in source image areas as training data and set their matched feature points in the destination image as label correspondingly. Instead of collecting matches on a per-frame basis, we do so every 60 frames.

Another challenge in dataset generation is data duplication. In some videos, the background and foreground objects in the overlapping regions do not change much. As a result, we may detect duplicate features. In addition, feature points in the source image may correspond to two or more different feature points in the destination image. Thus, it is necessary to deduplicate the dataset so that one feature point in the source image only corresponds to one feature point in the destination image, and there is no duplicated feature points.

Furthermore, in order to avoid continuous increase of the dataset size during live streaming video processing, data is only collected from four adjacent sampled frames. That is, as a new frame is sampled, the matched features from the oldest of the four sampled frames are removed from the dataset. We thus refer to our dataset as a **rolling dataset**. Finally, to evaluate the performance of our trained homography matrix, we divide dataset into a training set (80%) and a testing set (20%), during the live streaming video processing.

3.3 Homography Matrix Optimization

With RANSAC, the optimal homography matrix must be the one that results in the highest proportion of inliers and calculated from four points in the dataset. However, this is also the limitation of RANSAC. It is possible that the optimal solution may not be calculated by the existing data.

In this paper, we propose a learning-based method to continuously optimize the homography matrix initially calculated by RANSAC as new video frames are captured over time. Work [11] proposes a method to refine the homography matrix for a single pair of images through batch gradient descent with a learning rate of 0.01. However, this method is unable to converge due to the unique properties of back-to-back fisheye camera: with rigid connection, a pair of back-to-back fisheye lenses are nearly parallel, meaning that there is no large angle or scale transformation. Our proposed method creates a rolling dataset of matching features that can quickly adapt to changing scenes. It regards the homography matrix as a single fully connected layer with 3 inputs and 3 outputs without any other hidden layers, and uses the Adam optimizer for refining this fully connected layer. We reduce the learning rate to 10^{-6} according to the characteristics of the fisheye camera.

We use mean squared error (MSE) as the loss function, which is calculated as: $MSE = \frac{1}{n}\sum_{k=0}^{n}((x_k, y_k) - H(x'_k, y'_k))^2$, where n is the

Table 1: Inlier Percentage Comparisons between Shi-Tomasi Corner Detection and ORB on VZ videos.

Method	VZ_117	VZ_118	VZ_119	VZ_120	VZ_121	VZ_122
Shi-Tomasi	26.4%	25.7%	19.5%	32.4%	41.5%	38.5%
ORB	**69.4%**	**48.5%**	**45.8%**	**69.2%**	**57.0%**	**68.7%**

number of the feature points in the rolling dataset, $H(x_k, y_k)$ is the corresponding coordinates after homography transformation. Multiplying homogeneous coordinates by the homography matrix may result in a non-standard homogeneous coordinate, as interpreted below:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ w \end{bmatrix} \tag{2}$$

Since the label in the generated rolling dataset is standard homogeneous coordinate, we need to transform the result to standard homogeneous coordinate by Equation 3 below before applying the MSE loss in the optimization process.

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} \Rightarrow \begin{bmatrix} \frac{x'}{w} \\ \frac{y'}{w} \\ 1 \end{bmatrix} \tag{3}$$

Overall, we calculate the homography matrix H with the first video frame and set it as the initial weight of fully connected layer. For newly captured frames, we first determine whether it is a sample frame (line 10 in Algorithm 1). If not, we directly stitch the unwarped fisheye image into one omnidirectional image using the existing homography matrix. Otherwise, we determine whether the size of dataset exceeds our constraints. If so, we remove the features collected from the oldest sampled frame in the dataset. Then we extract new features, use RANSAC to obtain inliers, and add them to the rolling dataset. The training frequency is the same as the frame sampling frequency, e.g., every 60 frames in our setup. Before training, we deduplicate the dataset and shuffle the data. We then separate test data and training data from the dataset. We train the fully connected layer using Adam and MSE loss for 40 epochs with a learning rate of 10^{-6}. After training, we obtain a refined homography matrix H'. We compare the MSE loss of H' and H on the same testing dataset. If the MSE loss of H' is smaller than that of H, we update homography matrix for the stitching process of upcoming frames and set H' as the initial weight of next training process. Otherwise, we keep the same H for the next frames stitching. We implement multi-band-blending in the overlapping part for each frame to efficiently reduce ghosting effect.

4 EXPERIMENT AND RESULTS

4.1 Feature Comparison

In our experiment, we first compare the quality of feature points obtained from Shi-Tomasi corner detection and ORB. For this comparison, we recorded six videos in both indoor and outdoor environments using VUZE XR. Each video contains 600 frames. We collected feature points every 60 frames and used RANSAC to obtain an optimal homography matrix and its corresponding percentage of inliers. To measure the quality of feature points, we record the percentage of inliers corresponding to 10 optimal homography matrices in total and calculate their average for each video. Higher average of inliers percentage indicates better quality of detected features. Table 1 shows that ORB consistently achieves much higher inlier percentages than Shi-Tomasi corner detection. For some videos, the inlier percentage of ORB feature points is almost 2 times or more than that of Shi-Tomasi corner detection. This shows that for fisheye image stitching with limited overlapping area, ORB can find high quality feature points that can improve the accuracy of video stitching.

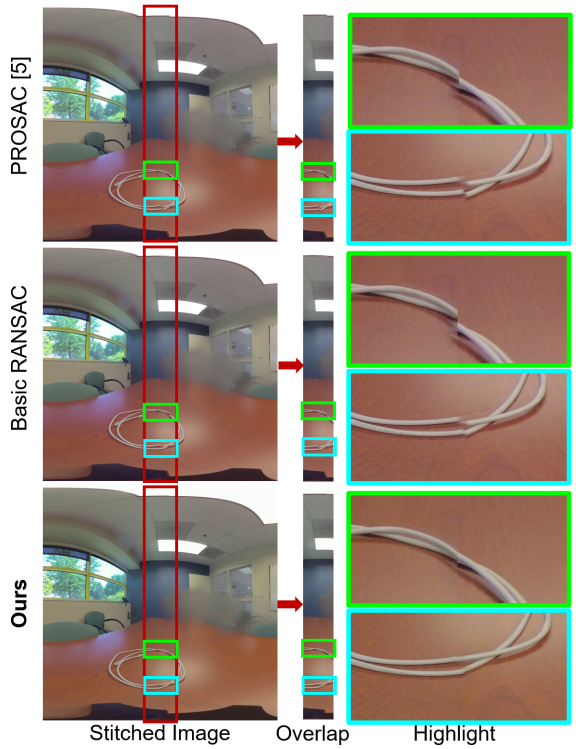

Figure 2: Example visual comparisons between Basic RANSAC, PROSAC, and our proposed method. We can observe that with our method, cables are aligned better.

4.2 Mean Square Error Comparisons

In this part, we compare a basic traditional feature-based method that uses RANSAC for finding the homography matrix (we denote these methods as "Basic RANSAC"), PROSAC [6] that is an improved-RANSAC with our proposed method on raw fisheye videos with 600 frames. We performed the comparison on three different datasets: i) six videos recorded by ourselves using the VUZE XR 360 camera with the resolution of 5760 × 2880, ii) 120 outdoor scenes downloaded from the FTV360 dataset [14] with the resolution of 3840 × 1920, and iii) two 360 videos downloaded from a fisheye video dataset [1] with resolution of 2560 × 1280. Since our method collects new features and optimizes homography matrix every 60 frames, the Basic RANSAC and PROSAC methods also collect new features every 60 frames and then run RANSAC and PROSAC on the collected features to update the homography matrix currently in use for stitching the next 60 frames. To measure the accuracy of these two methods, we calculated Mean Square Error (MSE), which is used in APAP [20], on test dataset we have

Table 2: Average MSE achieved by Basic RANSAC, PROSAC and our method on different datasets. "VZ" represents video dataset captured by VUZE XR. "April-fools-day", "hide-and-seek", and "race" are datasets recorded from different scenarios in the FTV360 dataset [14]. In particular, note that video "360_0080" is recorded with a moving camera. Our method can obtain much lower max test loss and variance compared to the Basic RANSAC and PROSAC [6] method.

	Basic RANSAC			PROSAC [6]			**Ours**		
Video_dataset	Max	Mean	Var	Max	Mean	Var	Max	Mean	Var
VZ	0.565	0.151	0.042	0.615	0.192	0.053	**0.284**	**0.134**	**0.011**
april-fools-day	1.052	0.171	1.708	13.724	1.714	190.815	**0.148**	**0.079**	**0.018**
hide-and-seek	0.491	0.102	0.109	37.241	4.435	807.354	**0.163**	**0.067**	**0.013**
race	0.101	0.124	0.406	117.573	11.846	29590.611	**0.126**	**0.052**	**0.006**
360_0087	0.026	0.014	0.000	0.031	0.021	0.000	**0.011**	**0.015**	**0.000**
360_0080 (moving camera)	2.100	0.505	0.313	1.848	0.808	0.181	**0.669**	**0.258**	**0.030**

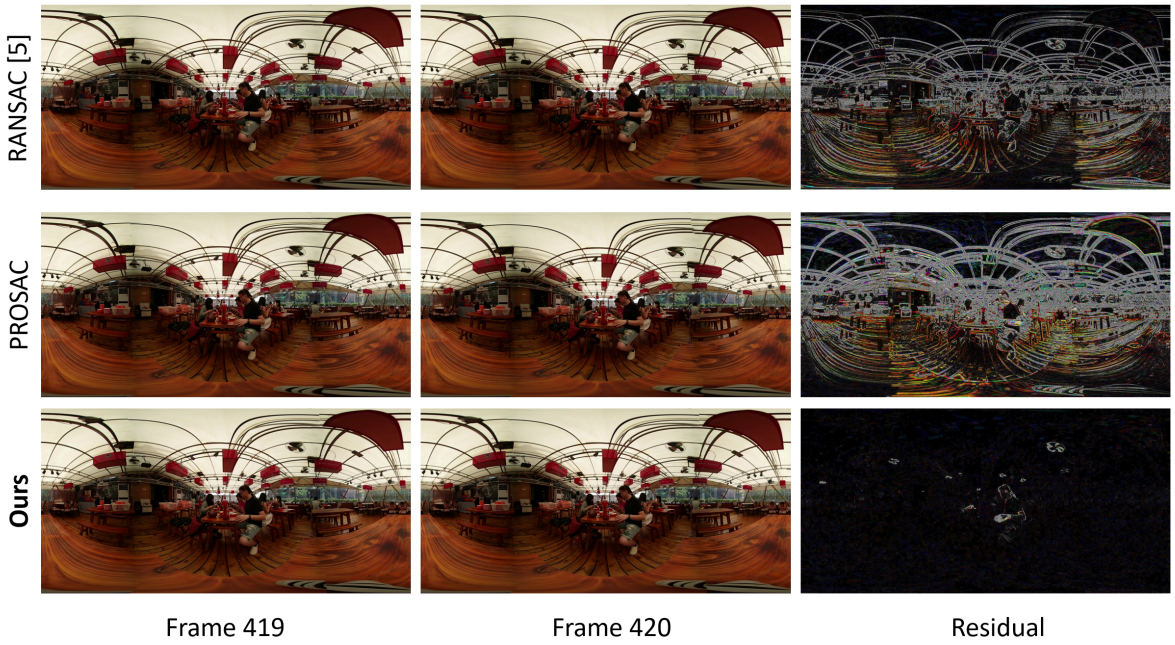

Figure 3: Visualization of "residual" represents the change from frame 418 to frame 419 (consecutive frames). Our method shows less change and jitter.

built according to Section 3.2. With our frame sampling rate of 60, we were able to record ten MSEs for each video. We then calculated the maximum, mean, and the variance of ten test MSEs. Mean MSE represents the overall accuracy of the entire video stitching. The lower the test MSE, the higher accuracy we obtain. Maximum test MSE indicates the worst situation where we may observe obvious misalignment. Variance shows the dispersion of test MSE, which can indirectly reflect the change of the homography matrix calculated from the sample frames. Lower variance represents small change of the homography matrix, where there is little jitter while updating the homography matrix.

Table 2 shows comparisons among three methods under these three metrics. It shows that our method successfully achieves lower maximum MSE, mean MSE, which demonstrates that the stitched fisheye videos based on our method are with less misalignments, as illustrated in Figure 2. Regrettably, the PROSAC method demonstrates significant limitations as it yields notably elevated maximum MSE values and variances when applied to `april-fools-day`, `hide-and -seek` and `race` datasets. This is becasue PROSAC is unable to find the right homography matrix for two or three videos in each dataset, where the collected features from the sampled frames are so limited that cannot run well with PROSAC. Except for these videos, PROSAC also obtains normal max MSE and variances.

For some videos, variance is less than 10^{-5} or approximately 0 since the Basic RANSAC and PROSAC methods almost detect the same features at each sampled frame, which results in the computed homography matrices being the same. Unfortunately, although the variances of Basic RANSAC and PROSAC are low enough, we can still observe jitters in the stitched videos, as shown in Figure 3. The residual image between consecutive frames `frame 419` and `frame 420`, where homograhy matrix is updated and there is not much difference between two frames, shows that our method can maintain lower jitter in the video. When there exists changes in the overlapping parts of the scene, our proposed method shows substantial improvements. In another example, video `360_0080` is recorded with a moving camera. Our method can obtain much lower maximum test MSE and variance compared to the Basic RANSAC and PROSAC method. In summary, compared with Basic RANSAC and PROSAC, our proposed method can obtain lower maximum test loss, mean test loss, and variance, which can reduce the occurrence

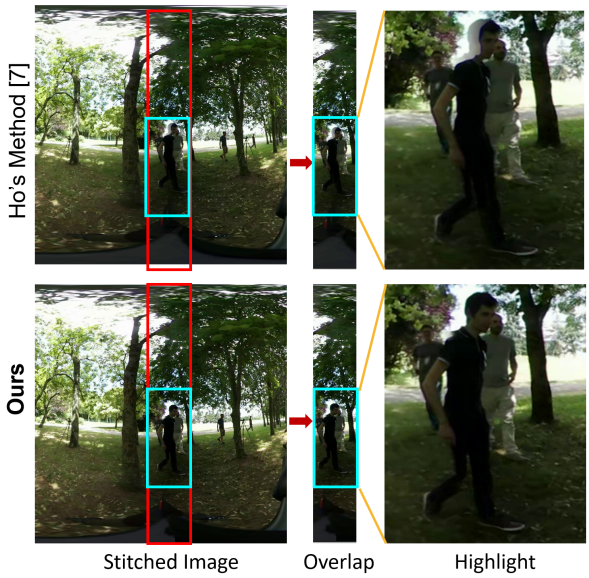

Figure 4: Compared with Ho's method, our method can avoid ghosting effect. Highlight shows a man walking across the overlapping part without ghosting effect using our method.

Figure 5: Visualized comparisons with Vuze VR Studio. Left: our results. Right: results from Vuze VR Studio. Colored rectangle highlight misalignments.

of misalignments and jitter and is with strong robustness under changing and more complex environments.

4.3 Visualized comparisons with calibration-based methods

We also compared our proposed method with calibration-based methods implemented in VUZE VR Studio, a software which can stitch two raw fisheye videos collected from VUZE XR into one omnidirectional video, and [8], proposed by Ho and Budagavi. Since we are unable to obtain the homography matrix used in these methods, we are unable to compute MSE or variance of the stitched videos from these methods. In addition, most omnidirectional image quality assessment methods [10] require ground truth, which is also infeasible in our experiment. We are thus unable to provide numerical comparisons. Instead, we compare visual results between two methods mentioned above and our method in Figures 4 and 5.

5 CONCLUSION

In this paper, we proposed a novel learning-based dual-fisheye video stitching method that can gradually improve the alignment accuracy with the Adam optimizer when processing 360-degree videos recorded live. The result shows that our method can achieve lower MSE loss compared to feature-based Basic RANSAC and PROSAC method. Our method also shows good robustness under changing and more complex environments. Moreover, compared to calibration-based methods, results indicate our method can improve the misalignment issue that exists with VUZE XR Studio. Our method also has limitations. For example, if the overlapping scene is relatively simple, such as only white walls, transparent glass, etc., where we cannot detect enough effective feature points, then our stitching results can be poor. Furthermore, our stitching pipeline can be further accelerated by exploiting parallelization. We believe that with the acceleration of GPU, our method can employ more sophisticated blending techniques and be applied for real-time high-quality fisheye video stitching. We plan to explore them in future work.

ACKNOWLEDGMENTS

We appreciate constructive comments from anonymous referees. This work is partially supported by NSF under grant CNS-2200042.

REFERENCES

[1] 2021. Fisheye Video. https://github.com/cynricfu/dual-fisheye-video-stitching
[2] 2022. Equirectangular Projection. http://mathworld.wolfram.com/EquirectangularProjection.html.
[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. 2006. Surf: Speeded up robust features. In *European conference on computer vision*. Springer, 404–417.
[4] Peter J Burt and Edward H Adelson. 1983. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics (TOG)* 2, 4 (1983), 217–236.
[5] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. 2010. Brief: Binary robust independent elementary features. In *European conference on computer vision*. Springer, 778–792.
[6] Ondrej Chum and Jiri Matas. 2005. Matching with PROSAC-progressive sample consensus. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Vol. 1. IEEE, 220–226.
[7] Martin A Fischler and Robert C Bolles. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* 24, 6 (1981), 381–395.
[8] Tuan Ho and Madhukar Budagavi. 2017. Dual-fisheye lens stitching for 360-degree imaging. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2172–2176.
[9] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[10] Jia Li, Kaiwen Yu, Yifan Zhao, Yu Zhang, and Long Xu. 2019. Cross-reference stitching quality assessment for 360 omnidirectional images. In *Proceedings of the 27th ACM International Conference on Multimedia*. 2360–2368.
[11] Jiahui Li, Fengshou Zhang, and Haoyang Cui. 2017. A Homography Matrix Estimation Method Based on Improved RANSAC Algorithm. *Computer Engineering and Applications* 53, 23 (2017), 6.
[12] I-Chan Lo, Kuang-Tsu Shih, and Homer H Chen. 2021. Efficient and Accurate Stitching for 360° Dual-Fisheye Images and Videos. *IEEE Transactions on Image Processing* 31 (2021), 251–262.
[13] David G Lowe. 1999. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE international conference on computer vision*, Vol. 2. Ieee, 1150–1157.
[14] Thomas Maugey, Laurent Guillo, and Cedric Le Cam. 2019. Ftv360: a multiview 360° video dataset with calibration parameters. In *Proceedings of the 10th ACM Multimedia Systems Conference*. 291–295.
[15] Leif E Peterson. 2009. K-nearest neighbor. *Scholarpedia* 4, 2 (2009), 1883.
[16] Edward Rosten and Tom Drummond. 2006. Machine learning for high-speed corner detection. In *European conference on computer vision*. Springer, 430–443.
[17] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. 2011. ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision*. Ieee, 2564–2571.
[18] Jianbo Shi et al. 1994. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*. IEEE, 593–600.
[19] Weihua Ye, Kaiwen Yu, Yang Yu, and Jia Li. 2018. Logical stitching: A panoramic image stitching method based on color calibration box. In *2018 14th IEEE International Conference on Signal Processing (ICSP)*. IEEE, 1139–1143.
[20] Julio Zaragoza, Tat-Jun Chin, Michael S. Brown, and David Suter. 2013. As-Projective-As-Possible Image Stitching with Moving DLT. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.