SoftwareX 26 (2024) 101669

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original Software Publication

ECPC – versatile multicamera system calibration framework for immersive video applications

Błażej Szydełko [a], Dawid Mieloch [a,*], Adrian Dziembowski [a], Jakub Stankowski [a], Dominika Klóska [a], Jun Young Jeong [b], Gwangsoon Lee [b]

[a] Poznań University of Technology, 60-965 Poznań, Poland
[b] Electronics and Telecommunications Research Institute, Daejeon, 34129 Republic of Korea

ARTICLE INFO

Keywords:
Immersive video
Multicamera system
Extrinsic parameters
Multicamera system calibration

ABSTRACT

Accurate extrinsic parameters calibration is crucial particularly in immersive video, where camera calibration plays a significant role, as its quality is essential for accurate reconstruction and efficient compression of three-dimensional scenes. While methods for intrinsic parameters calibration, color correction, and depth estimation are publicly available, there is a lack of versatile techniques for estimating extrinsics in the context of immersive video. The proposed Extrinsic Camera Parameters Calibration (ECPC) software addresses these limitations by proposing an extrinsic parameters estimation method and a framework for testing its accuracy. The software is compatible with MPEG Immersive Video framework, allowing for seamless integration and evaluation. The proposed method contributes to the advancement of immersive video applications by providing a reliable and comprehensive approach for estimating and evaluating extrinsic parameters.

Metadata

Nr	Code metadata description	
C1	Current code version	*V1*
C2	Permanent link to code/repository used for this code version	https://github.com/bszydelko/ecpc
C3	Permanent link to reproducible capsule	–
C4	Legal code license	*GNU General Public License (GPL)*
C5	Code versioning system used	*Git*
C6	Software code languages, tools and services used	*C++, Python, Blender, OpenCV, FFmpeg*
C7	Compilation requirements, operating environments and dependencies	*Python 3.8+, Blender 4.0+*
C8	If available, link to developer documentation/manual	https://github.com/bszydelko/ecpc/blob/main/README.md
C9	Support email for questions	blazej.szydelko@put.poznan.pl

1. Motivation and significance

Accurate camera calibration is fundamental for a wide range of applications, including measurements, accurate 3D reconstructions, scene understanding [1], camera networks and surveillance systems setup [2]. Intrinsic parameters calibration (estimating the optical characteristics of cameras) for machine vision applications poses no issues in terms of availability of open-source implementations [3,4] and comparative accuracy evaluation [5,6]. However, the extrinsic parameters calibration (estimating their relative positioning) can be still very challenging, particularly in the context of immersive video applications.

The estimation of extrinsic parameters is essential for achieving accurate 3D reconstruction from multiple views [7]. By determining the relative positions and orientations of cameras, it becomes possible to triangulate corresponding image points and reconstruct the 3D structure of the scene. Therefore, the accuracy of calibration is crucial for depth estimation, as it relies on precise knowledge of positions and orientations of cameras within the system [7], and, as a result, on an efficient compression and transmission of scene information [8]. Therefore, without proper calibration, depth estimation and rendering of virtual views in immersive video would be compromised. Immersive video requires calibration methods that are versatile enough to accommodate a wide range of camera configurations and placements. Available methods often cannot be used with all camera arrangements [9] or require the use of a kind of calibration marker that can be hard to use with large distances between cameras [10].

While methods for intrinsic parameters calibration, color correction [11], depth estimation [12,13], and final view quality assessment [14]

* Corresponding author at Dawid Mieloch, Institute of Multimedia Telecommunications, Poznan University of Technology, 60-965 Poznań, Poland.
E-mail address: dawid.mieloch@put.poznan.pl (D. Mieloch).

https://doi.org/10.1016/j.softx.2024.101669
Received 3 January 2024; Received in revised form 1 February 2024; Accepted 18 February 2024

have already been developed and made publicly available, as well as well-established frameworks for their comparison [15–17], the evaluation of extrinsic parameters quality lacks a standardized approach. Existing evaluation frameworks often rely on data specifically collected for assessing the performance of designed algorithms [10], which poses challenges in conducting meaningful comparisons within the research community. Moreover, many of the current datasets may not fully capture the distinctive characteristics of immersive video, as some frameworks use shifted images to generate pseudo-multiview images [18,19], or can be used only for a single stereo-pair image [20].

Acquiring ground truth parameters in natural sequences is a difficult task. Even when ground-truth extrinsic parameters are available, determining the final quality of rendered immersive video is not a straightforward process and this quality cannot be solely deduced from the measurement of the deviation from ideal parameters. The estimated extrinsics should result in consistent transformations between views, ensuring that the scene geometry aligns properly, so the rendering process can produce plausible virtual views. Therefore, a comprehensive assessment of extrinsic parameters quality necessitates measurement of the accuracy of estimated geometry.

To address these challenges, this paper proposes two main contributions. Firstly, Extrinsic Camera Parameters Calibration (ECPC) – a versatile software developed to enable the extrinsic parameters estimation in various camera configurations and placements. Secondly, a comprehensive framework prepared to evaluate different methods of extrinsic parameters estimation. This framework involves simulating a virtual Blender-rendered multiview sequence, estimating parameters from this sequence, and evaluating the quality of depth maps estimated using calibrated parameters. The provided software is compatible with the MPEG immersive video framework [21], supports JSON-based data format, omnidirectional media format (OMAF) coordinate system [22], and depth maps with normalized disparity [23], therefore, is the first publicly available toolset for evaluating the extrinsic parameters in this state-of-the-art application.

2. Software description

The ECPC framework is a set of software tools designed to simplify and improve the evaluation of extrinsic camera parameters and 3D geometry reconstruction from images or multiple views. It covers ground-truth data generation and reconstruction with evaluation phases.

2.1. Software architecture

The ECPC framework includes a range of core functionalities that are organized into five separate modules. The modules are presented in Fig. 1 and cover major aspects of immersive video production: scene setup, calibration, geometry reconstruction, and quality evaluation. Each module operates independently, providing flexibility and adaptability.

2.2. Software functionalities

The following subsections describe these fundamental functionalities in more detail, sequentially presenting an overview of the entire workflow.

2.2.1. Data generation

To calibrate the multicamera system and evaluate the accuracy of estimated camera parameters, it is essential to provide proper calibration data and reference ground-truth data for comparison. To achieve this, a virtual scene created in Blender software is used (Fig. 2), providing a controlled and constant test environment allowing for proper calibration and evaluation.

The Blender project contains a representative example of a scene captured by a multicamera system, together with scripts allowing for outputting the multiview calibration sequence in an MVD representation (multiple views and depth [24]).

The provided scene includes 10 high-definition virtual cameras arranged on an arc, modeling systems often used in immersive video applications [15]. However, as underlined earlier, all modules of the ECPC software do not have any constraints on camera arrangement, thus users of the software can arbitrarily modify it to model the arrangement of their interest.

The calibration algorithm (Section 2.2.3) estimates the positions of cameras on the basis of the positions of reference points, which are provided by tracking a calibration object ("marker", Section 2.2.2). In ECPC, this marker is an orange ball moving through the scene volume.

2.2.2. Marker tracking

The second module analyzes the calibration sequence generated in the previous step and outputs the position of the marker in each frame and view. The marker positions are stored in the output file containing the horizontal and vertical coordinates and a visibility flag indicating if

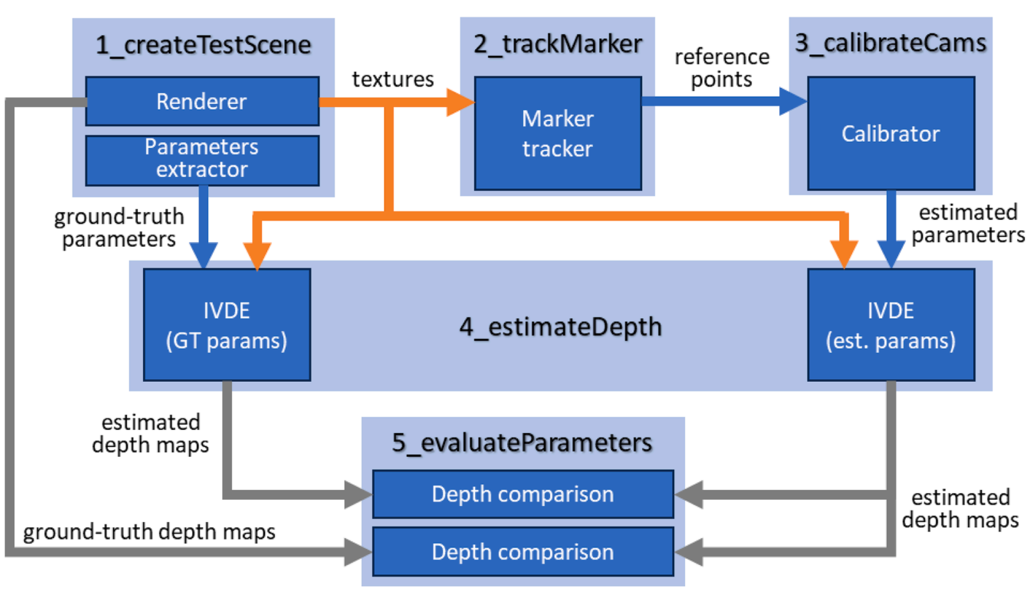

Fig. 1. The general scheme of the ECPC software.

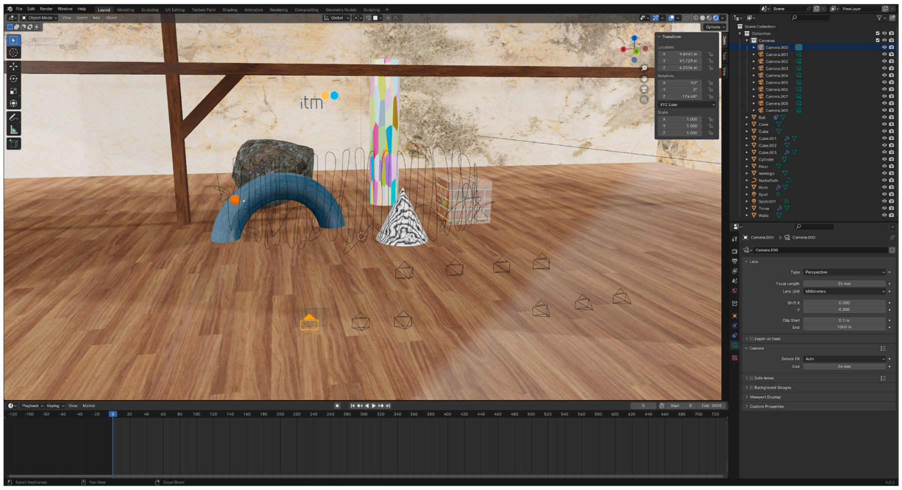

Fig. 2. The Blender project contains a test scene with 10 cameras, several static objects, and a moving calibration marker.

the marker was visible by the camera in an analyzed frame.

The marker detection algorithm is performed by analyzing both chroma components of the calibration sequence. All the pixels detected as orange (i.e., their chromas are: Cb ∈ [50, 70] and Cr ∈ [155, 195]) are in the first step segmented into clusters.

In the first frame, the largest cluster characterized by a high circularity coefficient is selected. Then, this cluster is tracked in the following frames, so the calculation of its accurate position does not require analysis of the entire frame because of the movement prediction (Fig. 3). The prediction allows for a significant decrease of the area being analyzed (blue rectangle in Figs. 3A and 3B), especially when consecutive frames of a calibration sequence are analyzed (Fig. 3A). When only every i th frame is used, prediction is less accurate (Fig. 3B) implying that a larger area of the frame has to be searched for the marker. Nevertheless, the position of the marker is detected correctly in both cases (Fig. 3C).

2.2.3. Calibration

The calibrator module serves as the core module of the ECPC software and is responsible for determining the extrinsic parameters for the multicamera system. The underlying calibration algorithm is based on minimizing the total distance between corresponding reference points from multiple views and their respective epipolar lines [25]. The optimization process is carried out by l-BFGS (limited-memory Broyden–-Fletcher–Goldfarb–Shanno [26]) algorithm, commonly used to solve problems with large numbers of variables, and minimizes an error function defined as:

$$E(\vec{t},q)=\sum_{c_r=0}^{N-1}\sum_{c_t=0\; c_t\neq c_r}^{N-1}\sum_{p=0}^{P-1} d(c_r,c_t,p),$$

where N represents the number of cameras in the system, P denotes the number of characteristic points in a camera pair (c_r,c_t), p is the index of a characteristic point, c_t and c_r are target and reference camera indices respectively, d quantifies the distance between point p and its corresponding epipolar line, $\vec{t}$ symbolizes the 3-dimensional translation vector of the camera optical center relative to a global coordinate system, while q is a quaternion-based representation of the rotation of a

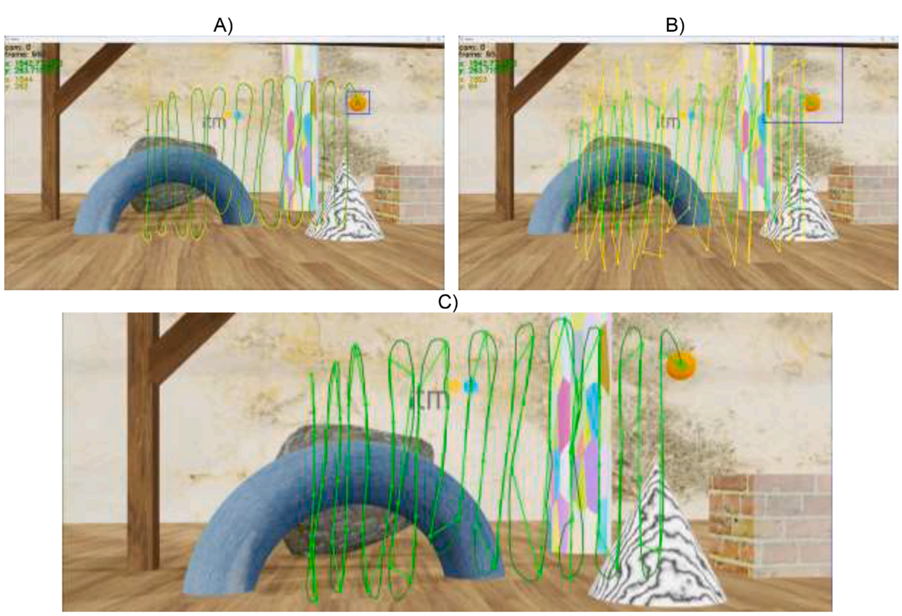

Fig. 3. Tracked marker. Yellow lines: the predicted path of a marker; green lines: the actual path measured by the marker tracker algorithm; blue rectangle: area in which the marker is being searched for; A: analysis of consecutive frames of a calibration sequence, B: analysis of every 10th frame of a calibration sequence, C: comparison of detected marker positions in A (dark green line) and B (light green line) – the curves overlap every 10th frame (light green dots).

camera. Therefore, for each camera, there are 7 variables being optimized in the process.

In the proposal, the calibration process can be performed in one or more global iterations. After each global iteration, the method excludes outliers from a list of input marker positions and repeats the process. A point is treated as an outlier if its error (i.e., its distance from the corresponding epipolar line) is tenfold larger than $E(\overrightarrow{t}, q)$. Therefore, even if some input points represent erroneously-detected objects (e.g., due to partial occlusion of calibration marker or temporary changes in the scene's lighting), they can be automatically detected and removed from the optimization.

The calibrator tool generates a .json file with sequence parameters, that will can be directly used for the depth estimation process.

2.2.4. Depth estimation

In the ECPC software, the accuracy of estimated parameters is evaluated by comparing depth reference depth maps with the ones estimated based on camera parameters received in the previous step.

To provide fast and reliable comparison, the ISO/IEC MPEG Video Coding reference software for depth estimation was used – IVDE (Immersive Video Depth Estimation, [12]), as it provides good quality depth maps for multiview systems with arbitrary camera arrangements and is publicly available on the MPEG public software repository [27].

The IVDE software, as an external component, is integrated into the ECPC repository as an independent part using the Git submodule feature.

2.2.5. Parameters evaluation

In the last step, the estimated depth maps are compared to assess the accuracy of estimated extrinsic camera parameters by means of the number of pixels with a wrong depth value ("bad pixels"). The number of bad pixels is calculated in two different ways. In the first type of evaluation, depth maps estimated using calculated extrinsic parameters are compared with ground-truth depth maps, directly rendered by Blender (first "depth comparison" block in Fig. 1). The second evaluation (second "depth comparison" block) is based on comparison of the depth maps estimated using calculated extrinsic parameters with depth maps estimated using ground-truth extrinsic parameters (extracted from Blender). We propose two evaluation paths because IVDE (as well as many other state-of-the-art depth estimation algorithms) estimates depth using texture analysis and inter-view comparison. Therefore, it is not capable of calculating the ideal geometry of a three-dimensional scene (e.g., for areas visible only in a single camera or smooth areas without edges), even if camera parameters are perfect.

The number of bad pixels with assumed properness threshold δ is calculated as a percentage of the total number of pixels:

$$BP_\delta = \frac{100\%}{W \cdot H} \sum_{y=0}^{H-1} \sum_{x=0}^{W-1} E_\delta(x, y),$$

where W and H denote the width and height of the depth map, and $E(x, y)$ is calculated as:

$$E_\delta(x,y) = \begin{cases} 1 \ if \ \left|D_{ref}(x,y) - D_{est}(x,y)\right| > \delta \\ 0 \ if \ \left|D_{ref}(x,y) - D_{est}(x,y)\right| \le \delta \end{cases},$$

where D_{est} and D_{ref} denote depth (i.e., normalized disparity) values in (x, y) pixel of depth map obtained using estimated params, and reference depth map, respectively.

By default, the number of bad pixels is calculated for several δ threshold values representing 1 %, 2 %, and 4 % of the range represented by a 16-bit depth map.

3. Illustrative examples

The software package includes a multistage example allowing for evaluation of the functionality and features of the provided software. The time required to run all stages (from rendering test sequence to the evaluation of camera parameters) is around 30 min.

Before running the software examples, the user needs to compile and build the necessary modules: the marker tracker (see Section 2.2.2.) and IVDE [12] – the depth estimator. For convenience, the building process can be done by executing the *0_buildSoftware.py* script.

In the first step of the provided example, the test scene (Fig. 2) is rendered from the provided Blender project. The scene is provided as an example and with little effort users can use their scene or modify the attached example.

Rendering of textures and ground-truth depth maps is done using Blender. Next, to provide compatibility with MPEG Immersive Video environment, views and depth maps are converted into "yuv" format. This step can be invoked by executing the 1_createTestScene.py script.

Next, the marker tracker has to be executed in order to extract the marker position from rendered scenes. If the user wants to use the software to calibrate a real system with a camera-captured calibration scene, the marker tracker can be used for camera-captured views. In the provided example, this step is invoked by the 2_trackMarker.py script. The result of marker tracking is visualized by a green line in Fig. 3.

The main part of the proposed software – the calibration of extrinsic camera parameters, can be started by invoking the 3_calibrateCameras.py script. The calibration script generates an output files estimatedParams_itN.json (where N is the number of an iteration) containing calibrated camera parameters.

Further steps of the provided example evaluate the accuracy of calibrated camera parameters through the quality of depth maps estimated on their basis. The estimation of depth maps is started by running the 4_estimateDepth.py script. This script executes IVDE [12] and calculates two sets of depth maps: using reference parameters and using calibrated camera parameters. The estimation of depth maps takes ~4 min for each set (on Ryzen 5900X CPU). The exemplary depth maps are shown on Fig. 4.

The 5_evaluate.py script evaluates the precision of calibrated camera parameters through measuring the quality of estimated depth maps. The evaluation results for a provided scene are presented in tables shown in Table 1.

The acquired results show that when the calibrated camera parameters are used, the quality of estimated depth maps is very high. For 2 % of acceptable difference threshold, only 4.55 % of depth maps samples are erroneous when compared to ground-truth rendered depth maps (left side of the table). To provide more context to this value, it should be mentioned that such accuracy is achieved only by the top 10 depth estimators in the Middlebury database [16] (https://vision.middlebury.edu/stereo/eval3/).

Moreover, when we compare depth maps estimated using calibrated parameters with depth maps estimated using ground-truth camera parameters (right side of the table), we can decrease the influence of errors introduced by depth estimation itself. In this case, only 0.64 % of depth map samples are erroneous (2 % threshold). It indicates that the proposed calibration method can be used to provide camera parameters of state-of-the-art quality, which are very close to ground truth ones.

The proposed framework provides the opportunity to test each step of the processing. For example, if can be used to test the accuracy of marker tracking. In the conducted experiment, the ground-truth positions of the marker were exported from the Blender scene and used in the calibration. The final result showed that in this case only 0.4 % of depth maps samples were erroneous when compared to depth maps estimated using exact camera parameters.

The ECPC software may also be successfully used for calibration of the real multicamera system capturing a natural scene. An example of calibration of the 20-camera system capturing a scene is shown in Fig. 5.

Fig. 4. Depth maps for view 6: A) rendered by Blender, B) estimated using exact camera parameters, C) estimated using calibrated camera parameters. The contrast of images was increased to highlight differences. Note that the quality of B is worse than for A because of imperfection of a depth estimation algorithm which cannot perfectly estimate depth for some parts of the scene (especially for areas acquired by a single camera (e.g., left top part of the view).

The calibration of the system starts with acquiring the calibration sequence containing a person holding the calibration object – an orange ball (attached to a stick to increase its visibility from different viewpoints, Fig. 5A). This sequence is analyzed in the marker tracking algorithm, which estimates the position of the marker in consecutive frames (Fig. 5B). Then, using the positions of the marker acquired from all cameras, the calibration algorithm estimates extrinsic parameters of the multicamera system (positions of cameras were visualized in Fig. 5C). In the end, these parameters are used for estimating depth maps (Fig. 5E) for the captured, natural multiview sequence (Fig. 5D).

4. Impact

The significance of the proposal lies in its potential to advance immersive video applications. By addressing the challenges associated with extrinsic parameters estimation and providing a standardized evaluation framework, this research enables the development of more accurate and reliable immersive video systems. This, in turn, contributes to the creation of new immersive video content of high quality. The software was already used to provide extrinsic parameters for the Martial Arts sequence [28], which became a part of MPEG Immersive Video Common Test Conditions [15]. The possibility of providing new content more easily paves the way for enhanced user experiences and applications in fields such as virtual reality, augmented reality,

Table 1
Evaluation results: a percentage of incorrect depth map samples for the threshold values measured for each view.

	Percentage of incorrect samples in depth maps estimated using calibrated camera parameters in comparison to rendered depth maps			Percentage of incorrect samples in depth maps estimated using calibrated camera parameters in comparison to depth maps estimated using exact camera parameters		
Acceptable difference threshold [% of max depth value]	1 %	2 %	4 %	1 %	2 %	4 %
Average	***5.68***	***4.55***	***4.01***	***1.61***	***0.64***	***0.38***

telepresence, and 3D mapping.

The software serves as a valuable reference point for correctly representing camera parameters and depth maps within the MIV framework, facilitating the development of new contributions and advancements in this compression standard. We also address the need for a standardized evaluation approach for extrinsic parameters, which is lacking in the current research landscape. Therefore, the framework promotes reproducibility by providing a proposal for an evaluation approach, enabling researchers to conduct comparative assessments of their work. As the examples presented in Section 3 have shown, the framework enables to test not only the accuracy of the calibration. Besides provided comparisons, future experiments that could be possible research directions are, e.g., testing the influence of input views quality (e.g., the strength of their compression or noise), or different camera arrangements on the final quality of estimated parameters. The usefulness of the framework in such a wide scope of research shows how impactful it can become.

Moreover, the proposed calibration method eliminates the need for expensive and complex calibration objects, making it an affordable and accessible solution for researchers and practitioners in the immersive video field, lowering entry barriers. ECPC was already shared with the MPEG Video Coding group and was decided by its experts to be available on the MPEG software repository to provide a reference for future experiments performed in the standardization process and to encourage further collaboration.

5. Conclusions

The paper highlights the challenges associated with extrinsic parameter calibration and the importance of this calibration for achieving accurate 3D reconstructions and depth estimation in immersive video applications. We introduce two key contributions: the versatile ECPC method for diverse camera setups and a comprehensive evaluation framework. This framework allows for the assessment of parameter quality through simulations in Blender-rendered multiview sequences and depth map evaluations. The software is compatible with the MPEG immersive video framework to promote collaboration and lower entry barriers for new researchers interested in works on this subject.

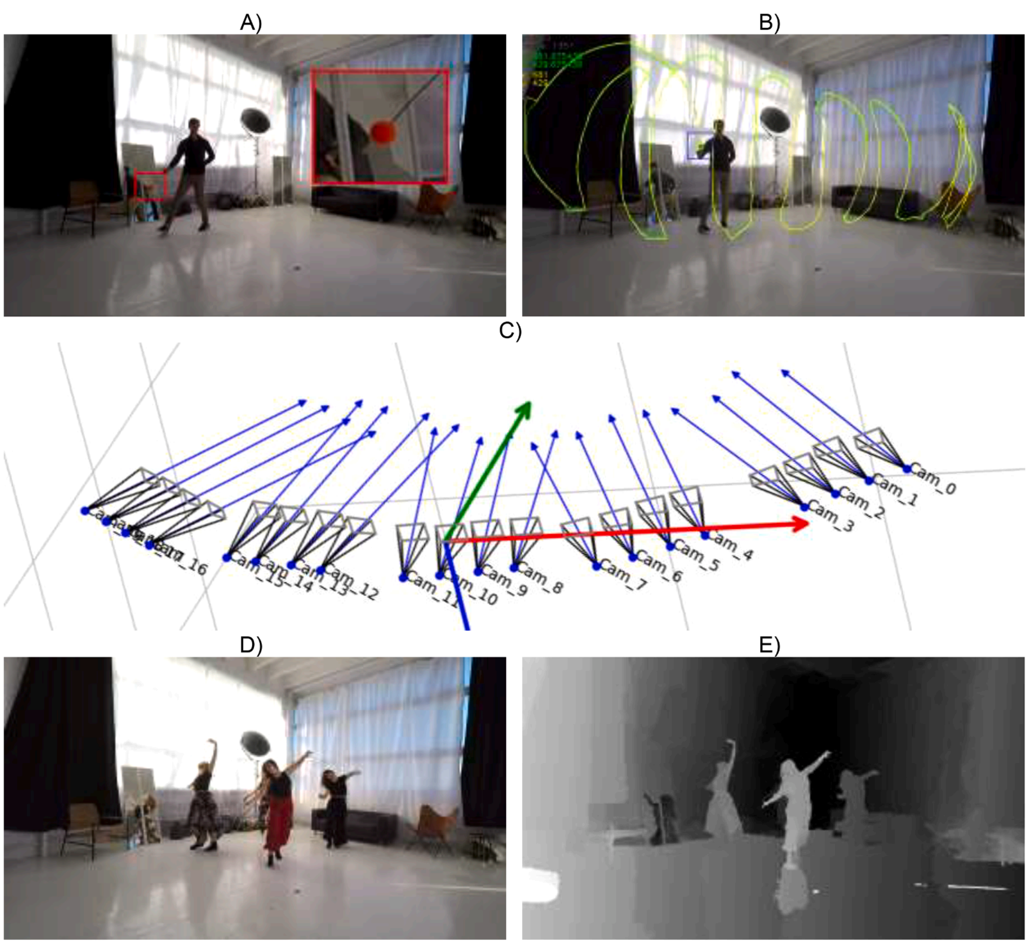

Fig. 5. Using ECPC for natural sequence. A: single frame of calibration sequence, B: tracked marker, C: visualization of multicamera system (based on estimated extrinsic parameters), D: single frame of the sequence, E: estimated depth map.

CRediT authorship contribution statement

Błażej Szydełko: Writing – original draft, Software. **Dawid Mieloch:** Writing – review & editing, Writing – original draft, Supervision, Software, Methodology, Formal analysis, Conceptualization. **Adrian Dziembowski:** Writing – review & editing, Writing – original draft, Software, Project administration, Formal analysis. **Jakub Stankowski:** Writing – review & editing, Software. **Dominika Klóska:** Writing – review & editing, Validation. **Jun Young Jeong:** Writing – review & editing, Validation. **Gwangsoon Lee:** Writing – review & editing, Validation, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgements

The work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2018–0–00207, Immersive Media Research Laboratory).

References

[1] Olagoke AS, Ibrahim H, Teoh SS. Literature survey on multicamera system and its application. IEEE Access 2020;8:172892–922.
[2] Wang Xiaogang. Intelligent multicamera video surveillance: a review. Pattern Recognit Lett 2013;34(1):3–19.
[3] Jianzhu Huai, Yuan Zhuang, Yuxin Shao, Grzegorz Jozkow, Binliang Wang, Junhui Liu, Yijia He, Alper Yilmaz, "A Review and Comparative Study of Close-Range Geometric Camera Calibration Tools," arXiv:2306.09014, preprint, 2023.
[4] Gerum Richard C, Richter Sebastian, Winterl Alexander, Mark Christoph, Fabry Ben, Bohec Céline Le, Zitterbart Daniel P. CameraTransform: A Python package for perspective corrections and image mapping. SoftwareX 2019;10:100333.
[5] Salvi Joaquim, Armangué Xavier, Batlle Joan. A comparative review of camera calibrating methods with accuracy evaluation. Pattern Recognit 2002;35(7):1617–35.
[6] Tsoy T, Safin R, Martinez-Garcia EA, Dutta Roy S, Saha SK, Magid E. Exhaustive Simulation Approach for a Virtual Camera Calibration Evaluation in Gazebo. In: 2022 8th International Conference on Automation, Robotics and Applications (ICARA); 2022. p. 233–8.
[7] Kukolj D, Bolecek L, Polak L, Kratochvil T, Zach O, Kufa J, Slanina M, Grajek T, Samelak J, Domański M, Milovanovic DA. 3D Content Acquisition and Coding. Signals and Communication Technology 2019:41–95.
[8] Mieloch D, Dziembowski A, Domański M. Depth Map Refinement for Immersive Video. IEEE Access 2021;9:10778–88.
[9] Li B, Heng L, Koser K, Pollefeys M. A multiple-camera system calibration toolbox using a feature descriptor-based calibration pattern. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems; 2013. p. 1301–7.
[10] Rameau Francois, Park Jinsun, Bailo Oleksandr, Kweon In So. MC-Calib: a generic and robust calibration toolbox for multicamera systems. Comput Vision Image Understanding 2022;217:103353.
[11] Dziembowski A, Mieloch D, Różek S, Domański M. Color Correction for Immersive Video Applications. IEEE Access 2021;9:75626–40.
[12] Mieloch D, et al. Overview and Efficiency of Decoder-Side Depth Estimation in MPEG Immersive Video. IEEE Trans Circuits Syst Video Technol 2022;32(9):6360–74. Sept.
[13] Rogge S, et al. MPEG-I Depth Estimation Reference Software. In: 2019 International Conference on 3D Immersion (IC3D); 2019.
[14] Moina-Rivera Wilmer, Gutiérrez-Aguado Juan, Garcia-Pineda Miguel. Video quality metrics toolkit: an open source software to assess video quality. SoftwareX 2023;23:101427.
[15] "Common Test Conditions for MPEG Immersive Video", ISO/IEC JTC 1/SC 29/WG04 N0169, Jan. 2022.
[16] Scharstein D, Hirschmüller H, Kitajima Y, Krathwohl G, Nesic N, Wang X, Westling P. High-resolution stereo datasets with subpixel-accurate ground truth. In: German Conference on Pattern Recognition (GCPR 2014); 2014. p. 31–42.
[17] Marelli Davide, Bianco Simone, Ciocca Gianluigi. SfM Flow: a comprehensive toolset for the evaluation of 3D reconstruction pipelines. SoftwareX 2022;17:100931.
[18] Cao S-Y, Hu J, Sheng Z, Shen H-L. Iterative Deep Homography Estimation. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2022. p. 1869–78.
[19] Lin TY, et al. Microsoft COCO: common Objects in Context. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T, editors. Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science8693. Springer; 2014.
[20] Liu S, et al. Content-Aware Unsupervised Deep Homography Estimation and its Extensions. IEEE Trans Pattern Anal Mach Intell 2023;45(3):2849–63. 1 March.
[21] "Test Model 18 for MPEG Immersive Video," Doc. ISO/IEC JTC 1/SC 29/WG 04N 0409, Nov. 2023.
[22] Hannuksela MM, Wang Y-K. An Overview of Omnidirectional MediA Format (OMAF). Proc IEEE 2021;109(9):1590–606. Sept.
[23] "Depth Map Formats Used Within MPEG 3D Technologies," document ISO/IEC JTC1/SC29/WG11 MPEG2017/N16730, Geneva, Switzerland, Jan. 2017.
[24] Vetro A, et al. 3D-TV Content Storage and Transmission. IEEE Trans Broadcast 2011;57(2):384–94.
[25] Domański M, Dziembowski A, Mieloch D, Łuczak A, Stankiewicz O, Wegner K. A practical approach to acquisition and processing of free viewpoint video. In: 2015 Picture Coding Symposium (PCS); 2015.
[26] Zhu Ciyou, Byrd Richard H, Lu Peihuang, Nocedal Jorge. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. ACM Trans. Math. Softw. 1997;23(4):550–60. Dec. 1997.
[27] "MPEG-I Visual /IVDE – Gitlab", available online, accessed 06.11.2023: https://gitlab.com/mpeg-i-visual/ivde/-/tree/v7.0.
[28] D. Mieloch, A. Dziembowski, B. Szydełko, D. Klóska, A. Grzelka, J. Stankowski, M. Domański, G. Lee and J.Y. Jeong, "New natural content - MartialArts", ISO/IEC JTC1/SC29/WG4 MPEG 141, M61949, Jan. 2023.