The Visual Computer (2023) 39:4103–4116
https://doi.org/10.1007/s00371-022-02578-z

ORIGINAL ARTICLE

Stabilization of spherical videos based on feature uncertainty

A. Luchetti[1] · M. Zanetti[1] · D. Kalkofen[2] · M. De Cecco[1]

Accepted: 6 June 2022 / Published online: 9 July 2022

Abstract
Nowadays the trend is to acquire and share information in an immersive and natural way with new technologies such as Virtual Reality (VR) and 360° video. However, the use of 360° video, even more the use of VR head-mounted display, can generate general discomfort ("cybersickness") and one factor is the video shaking. In this work, we developed a method to make the viewing of 360° video smoother and more comfortable to watch. First, the rotations are obtained with an innovative technique using a Particle Swarm Optimization algorithm considering the uncertainty estimation among features. In addition, a modified Chauvenet criterion is used to find and suppress outliers features from the algorithm. Afterward, a time-weighted color filter is applied to each frame in order to handle also videos with small translational jitter, rolling shutter wobble, parallax, and lens deformation. Thanks to our complete offline stabilization process, we achieved good-quality results in terms of video stabilization. Achieving better robustness compared to other works. The method was validated using virtual and real 360° video data of a mine environment acquired by a drone. Finally, a user study based on a subjective and standard Simulator Sickness Questionnaire was submitted to quantify simulator sickness before and after the stabilization process. The questionnaire underlined alleviation of cybersickness using stabilized videos with our approach

Keywords Video stabilization · 360° video · Particle swarm optimization · Chauvenet's criterion · Uncertainty estimation · Shaking

1 Introduction

The recent spread of cheap 360° cameras is leading to an extension of their use and therefore availability to a large number of people.

Through a spherical video, they give the most accessible way to Virtual Reality (VR). VR allows users to enter a digital world and fully immerse themselves in it. In a more general context VR is part of Mixed Reality (MR)[1]. One of the more well-known MR uses is gaming, but in recent years with its ability to immerse its users in a virtual or augmented world, MR was used for education [2], training [3], healthcare [4], tourism [5].

Some famous 360° device examples of omnidirectional camera are the GoPro Max [1], Xiaomi Mijia Mi Sphere 360 [2] and Insta360 [3]. After the explosion of immersive media technologies even social platforms, such as Facebook[4] and Youtube[5] have been providing 360° videos in the offered services. The use of spherical videos is mainly used for action sports [6] such as biking or skiing, virtual tours [7], education [8], and adult videos.

Omnidirectional camera videos contain more information because they capture data from the entire environment. Their videos allow the viewer to look in all directions. This technique allows creating what is better known as "immersive video" because the user is completely immersed in a vir-

✉ A. Luchetti
alessandro.luchetti@unitn.it

M. Zanetti
matteo.zanetti@unitn.it

D. Kalkofen
kalkofen@icg.tugraz.at

M. De Cecco
mariolino.dececco@unitn.it

1 Department of Industrial engineering, University of Trento, Sommarive, 9, 38123 Trento, Italy

2 Institute of Computer Graphics and Vision, Graz University of Technology, RechbauerstraBe 12, 8010 Graz, Austria

1 https://gopro.com [Accessed: February 2022].
2 https://www.mi.com [Accessed: February 2022].
3 https://store.insta360.com [Accessed: February 2022].
4 https://facebook360.fb.com/ [Accessed: February 2022].
5 https://support.google.com/youtube/answer/6178631?hl=en [Accessed: February 2022].

tual world presented to him and he can control the viewing direction during playback. Anyhow, the level of the viewer's immersive experience can change depending on the medium in which the 360° videos are watched.

The viewer can watch 360° videos on a standard display of a smartphone, tablet, or computer with the possibility to scroll the graphic interface with a mouse or touch to see different parts of the scene. This method is similar to watching a conventional video but with the benefits of having the possibility to decide the viewing angles thanks to the more images information acquired. This advantage is widely used by filmmakers during editing because, by having more options for angle choices, they can decide and fix the viewing angles generating a standard 2D video that emphasizes the video from the best parts of the scene. The medium that widens the immersive experience is given by Virtual Reality (VR) Head-Mounted Display (HMD) that isolates the user from the surrounding environment and makes the experience more immersive. The user can control the viewing direction with a simple and natural interaction and an effective and hyper-realistic result. The HMD is a wearable hardware device that displays video on a small head-mounted screen that updates dynamically to show different parts of a scene as the viewer turns his head.

With the growing use of immersive virtual environments also the environmental psychology research based on human–nature interactions started to study the connection between VR and cybersickness [9] including the impact of improved camera stability on cybersickness [10]. With the use of 360° VR, especially with HMDs, users can undergo general discomfort, nausea, headache, vomiting, disorientation. Causes could include the user's health conditions or too much time spent in VR but mainly is due to video shaking [11]. It can be very disorienting for a VR-user, which may make the viewer fall or get sick.

Exposure to camera vibration is affected by the support to which the camera is connected, such as our body or an external object. For example, in the case of Unmanned Aerial Vehicles (UAV) the impact of the wind field and rotor dynamics conditioned its trajectory [12]. These disturbances inevitably generate shaking in panoramic shots. To make the final experience more comfortable, the most recent omnidirectional cameras have mechanical solutions and real-time optical flow algorithms to smooth out shaky movements. Sometimes this is not enough and some advanced video editing software solutions such as Adobe Premiere Pro CC[6] and Cyberlink PowerDirector 365[7] are used. However, commercial 360° stabilization solutions are expensive, hard to use, and require intensive training before they can be used. Furthermore, it is not always clear which commercial stabilization algorithms are used in the mentioned software products, making them in many cases unsuitable for the required application.

In this work, we provide a robust and efficient solution to stabilize 360° camera motion, to make the resulting panoramic video more comfortable to view. We introduce a new approach for estimating the camera orientation among frames based on the Particle Swarm Optimization algorithm. Our approach takes into account the estimation of uncertainty among features and the suppression of outliers through a modified Chauvenet criterion.

This paper can be divided as follows:

– In the introduction, we presented the problem of stabilization for a more comfortable VR experience.
– In Sect. 2, we discussed the state of the art on stabilizing 360° video.
– In the following section, we describe the developed method.
– In the fourth section, we validate its results in a controlled environment by using simulated 360° videos, as well as on captured 360° videos.
– In the fourth section, we present the results of a user study comparing processed and unprocessed 360° videos.
– In the final section, we expose the drawn conclusions.

2 Related work

The challenge of camera motion estimation is an important research topic for computer vision, widely used especially for robotics applications such as visual servoing [13] or visual simultaneous localization and mapping [14]. Most of existing techniques to estimate camera motion are based on feature-correspondences techniques [15] or on analysis of the optical flow between consecutive video frame [16]. In recent years, the approaches based on the convolutional neural networks (CNNs) have also been exploited [17].

On the other hand the goal of the stabilization is to remove the camera shaking from the estimated camera motion. Stabilization of 360° video was studied in many works in the past decade. The proposed approaches can be divided into 2D, 3D, and a combination of the two (2.5D).

The work in [18] uses the SIFT algorithm and the 3D approach based on the Structure from Motion (SfM) process for feature extraction and obtaining the camera path, respectively. The SfM estimates a camera pose (location and orientation) and point cloud. Each output frame is generated by warping a single frame from the input video. The 3D reconstruction makes this process computationally expensive and not robust under certain situations such as the absence of

[6] https://www.adobe.com/products/premiere.html [Accessed: February 2022].

[7] https://www.cyberlink.com/products/powerdirector-video-editing-software [Accessed: February 2022].

translational motion or with a significant amount of pure rotations. In general, approaches based on 3D methods [19,20] depend highly on camera motion, limiting their practical application.

In [21], the authors through a 2.5D approach first estimate and remove all rotations; then, they employ mesh-based image warping [22] in order to compensate the remaining high-frequency jitters caused by camera translation and parallax. After that, they restore proper camera rotations. They use the Kabsch algorithm [23] to find the rotation matrix between two adjacent frames, with the disadvantage that it is less robust because it does not avoid possible local minima by limiting the search. In addition, the warp stabilizer used can lead to a warping increase and sometimes an inappropriate output frame can be generated.

Most existing 2.5D approaches start by tracking the motion of feature points in the video, such as in [24]. In [24], they convert the 360° frames into cube map representation. This representation is less distorted than the equirectangular projection near the poles. On the other hand, if a tracked point falls outside its originating cube face that corresponding observation is dropped and the track is ended. With this method, it is not possible to select a Region of Interest (ROI) for the algorithm. Therefore, the method will fail if the video contains static overlays such as logos or text. Their algorithm [24] removes all rotation from the input video, and afterward, it adds back a smoothed version of rotations. It uses keyframes that are appropriately spaced, so it can reliably estimate the true rotations. To estimate the rotation among them it uses a five-point algorithm in a RANSAC procedure [25]. For the inner frames, it solves a 2D optimization that maximizes the smoothness of tracked feature point trajectories. Finally, it uses a flexible deformed-rotation motion model for handling residual jitter. The most recent approach [26] differs from [24] by using a 3D spherical warping model which is derived from a motion estimation that handles both, rotation and translation, which allows more control than [24]. In both aforementioned works, the search of the keyframes slows down the tracking operation.

Our work is similar to [27,28]. They estimate and smooth the relative rotations between consecutive equirectangular frames. Similar to [27,28], we also track feature points between adjacent frames. However, due to the similarity between each pair of frames small translations and rotations may be difficult to estimate. The limitation of their approach is that if the features tracked are always those of the first frame, less smooth results can be obtained due to the accumulation of errors.

In our work we developed a 2.5D method that estimates 3D rotations from the selected features motions without involving 3D structure from motion methods. We periodically detect new features with their uncertainty in frames taken in consecutive pairs. In order to remove only shaking while preserving intentional rotations, and to compensate the remaining jitters, we use a moving average filter and a time-weighted average filter in the pixel frame color, respectively. In this way, we can handle not only small translations and parallax effect but also videos with temporal artifacts, such as shearing and wobbling, otherwise impossible to manage with a mesh-warping approach.

3 Method

The method can be divided into three steps.

- First, we estimate the camera rotation between each pair of consecutive frames.
- Subsequently, we remove undesired rotations from each frame to generate new stabilized equirectangular images.
- Finally, we produce the stabilized 360° video.

3.1 Step 1: estimating relative camera rotations

First of all, the frames from the 360° video are read and saved as equirectangular images (2880 × 5760 pixel) in the MATLAB R2019b environment using the MATLAB $VideoReader$ function. This function returns an object that contains information about the video file and allows us to read data from the video. Equirectangular projection allows us to show 360° images in a single 2D visualization since it shows 360° × 180° angles at once. This projection, which transforms spherical coordinates into planar coordinates, creates severe geometric distortions near the poles, i.e., away from the horizontal centerline. However, it is the most popular and most widely supported format.

The possible movement of the camera involves both rotations and translations. In our method we decouple rotation from the other motions and handle them separately. First we find the rotation matrix between two consecutive frames, describing only the 3D rotations of yaw, pitch and roll, assuming zero translations. In the case of panoramic video, this is justified by the large distance of the 360° camera image from its surroundings, which makes the relative translation between frames negligible. In addition, camera rotation has a much more significant impact than translation in the final stabilization process because translations can be more tolerated. However, remaining jitters generated by the translations, not considered in this first step, is handled by a final filter added in the stabilization process.

The first input image is transformed into 2D grayscale. Then the SURF algorithm [29] is applied to detect blob features. The algorithm is used in a rectangular Region of Interest (ROI) to exclude the more distorted areas near the poles corresponding to high latitude and lower areas of the image.

A feature-tracking algorithm is then used to track points using Kanade-Lucas-Tomasi (KLT) [30] applied only to the adjacent frame. KLT is based on a small motion assumption. In our case, the accuracy is high because the difference in time and space between each two consecutive equirectangular images is small. This feature-tracking algorithm is commonly used for video stabilization, camera motion estimation, and object tracking. It has the advantage of being fast and handling RGB images.

The matched features between two consecutive frames are shown in Fig. 1, found with SURF and KLT algorithms, respectively, for the first and adjacent frames.

The problems most frequently observed with the use of the KLT algorithm, in the analysis of real sequences, concern overlapping of multiple movements, occlusions, lighting changes, and non-rigid movements. Under these conditions, some points can be lost. For this reason, to strengthen our method, we periodically reacquired points every two consecutive frames with the SURF detector.

We do not use SURF feature descriptor and feature matching because they are computationally demanding and KLT is robust enough under the assumption of small transformations, e.g., rotations and scale.

However, the KLT method does not guarantee that the corresponding points in the adjacent frame are feature points. To solve that we used the Block method [31] to extract features' descriptors in the first and consecutive frames. It consists of a simple square neighborhood extraction to identify the features. The Block method extracts only the neighborhoods fully contained within the image boundary. Therefore, only valid points in the first and the corresponding points in the adjacent frame are stored. The difference between each pair of features' descriptors is obtained by summing all the differences among the descriptors' components. This provides the uncertainty estimation values among features.

Using this parameter in the optimization process improves the robustness of our method without affecting its computational efficiency because the features' descriptor is simple and allows a smaller number of matching candidates to be used. The features are sorted according to their weights in ascending order. A high weight value means that the descriptors between a pair of features are different.

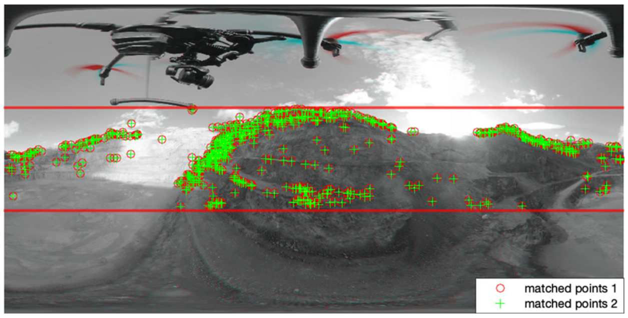

Fig. 1 Matching points of the first (in red) and second (in green) frames with ROI (960×5760 pixel)

To find and suppress possible outliers from the list of features, a modified Chauvenet criterion [32] is used. We modified Chauvenet by changing the standard deviation with the camera angular resolution term in the expression of fractional deviation from the mean. The new expression is:

$$D_i = \frac{|x_i - \overline{x}|}{Res} \tag{1}$$

where

- D_i is the fractional deviation from the mean;
- x_i is the value of suspected outlier;
- $\overline{x}$ is the mean of the data set;
- Res is the camera angular resolution value.

The set of data used to apply the modified Chauvenet criterion was obtained by taking n-times three features at a time and calculating their 3D rotations comparing them to those of the adjacent frame. The total number of previously sorted features was divided into intervals, and the calculation of 3D rotations was repeated within them. Each interval identifies a subset of features and the percentage values indicate which subset of the sorted features we are considering. In each interval, we obtained a histogram with the height of the baskets indicating the number of elements and the abscissa angle values. An example of the histograms obtained for two different intervals of features is shown in Fig. 2. The maximum spread value was extracted by adding the minimum and maximum absolute values of the angles obtained. In Fig. 2 the spread for each angle in the interval 0–5% is less than the one in the interval 95–100%. In particular the spreads of yaw, pitch, roll angles in the interval 0–5% are 0.1037°, 0.2452°, and 0.1056°, respectively; in the interval 95–100% are 0.4919°, 0.8922°, 0.5116°.

Dividing the features with 1% intervals, the results of the histogram spreads for each angle are displayed in Fig. 3. In the same figure, in red, the range selected with the modified Chauvenet criterion corresponds to the spreads of histograms with values above the Chauvenet threshold.

In Fig. 4 can be seen how, thanks to the use of the Res camera resolution in the modified Chauvenet expression Eq. (1), if the variations of the angle values are low, and therefore a low standard deviation occurs, no values are considered as an outlier. This is motivated also by the fact that, as shown in Figs. 3 and 4, the maximum value of the weights, when comparing their first plot, is different by one order of magnitude. Moreover, after the weights of the features are sorted in increasing order as in Figs. 3 and 4, the possibility to find outliers lies in the last intervals; for this reason, the research can

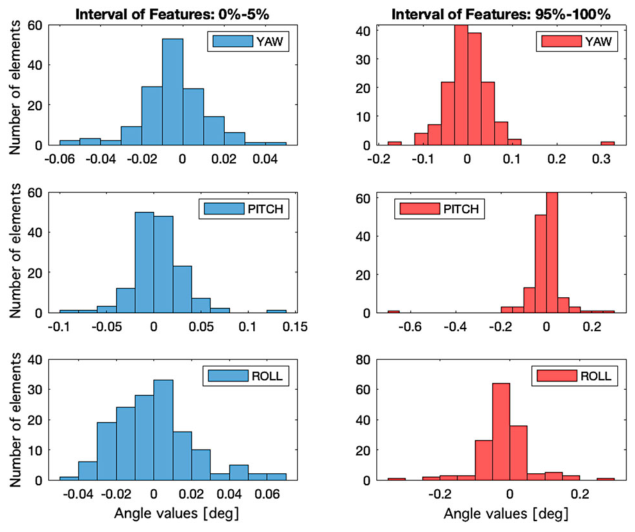

Fig. 2 Histograms of the values assumed by each 3D rotation angle by taking two different subsets of features previously sorted in ascending order according to the weight associated with their uncertainty estimate

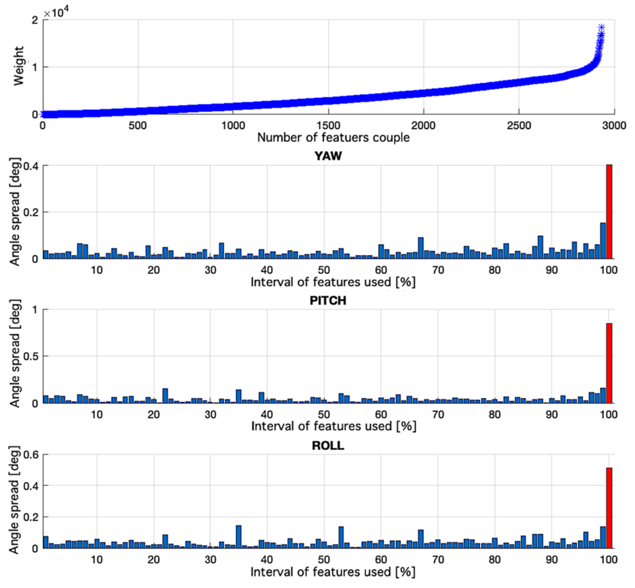

Fig. 3 Weight for each pair of features and histograms spread of each 3D angle in each interval between two random frames: in red the outliers with Chauvenet filtering

be focused in that area reducing by a lot the computational time.

The final architecture of the method used to find the valid features with their estimation of uncertainty between each pair of frames is schematically shown in Fig. 5. Each step of Fig. 5 will be validated in Sect. 4.

The valid points found in the previous steps are used as input to the optimization algorithm. The points selected in the first frame and those tracked in the adjacent frame are projected from the two equirectangular images with N rows and M columns, to the surface of two unitary radius spheres.

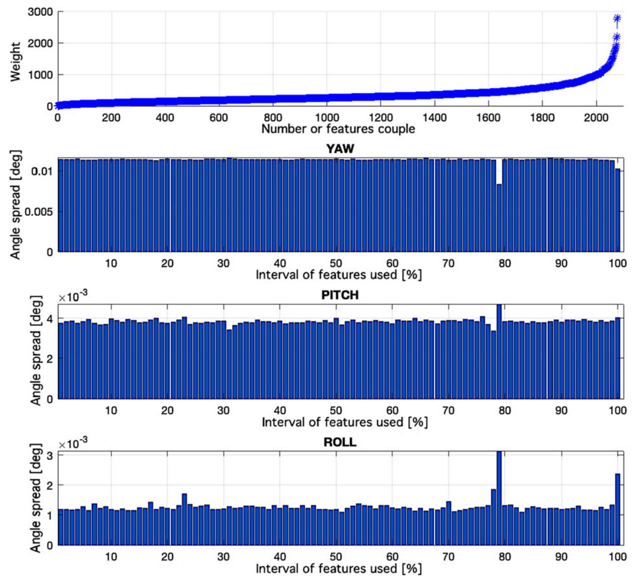

Fig. 4 Weight for each pair of features and histograms spread of each 3D angle in each interval between two random frames: no outliers with Chauvenet filtering

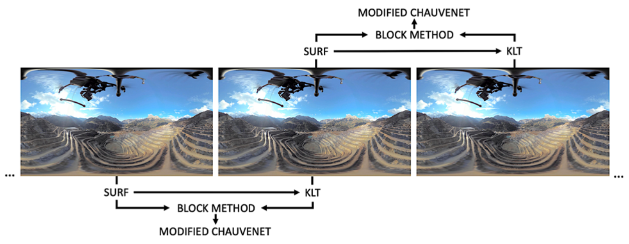

Fig. 5 Architecture for finding valid features

First of all each selected image's pixel in 2D Cartesian coordinates pixel (n, m) is transformed in spherical coordinates, computing the corresponding azimuth a and elevation e, setting the radius r equal to 1.

The relations used for the conversion are:

$$a = -\left(\frac{m}{M} - 0.5\right) \cdot 2\pi \tag{2}$$

$$e = -\left(\frac{n}{N} - 0.5\right) \cdot \pi \tag{3}$$

$$r = 1 \tag{4}$$

The purpose of the value 0.5 in Eqs. (2–3) is to have the spread of the angle between $\pm\pi$ and $\pm\frac{\pi}{2}$ for a and e, respectively.

Finally, the 3D point cloud is obtained through the mapping from spherical coordinates to 3D Cartesian coordinates:

$$x = r \cdot \cos(e) \cdot \cos(a) \tag{5}$$

$$y = r \cdot \cos(e) \cdot \sin(a) \tag{6}$$

$$z = r \cdot \sin(e) \tag{7}$$

The two "spherical" point clouds centered in the same origin are rotated to try to overlap the matching points by using a Particle Swarm Optimization (PSO) algorithm [33].

PSO algorithm is applied for each consecutive frame, trying to minimize the objective function by changing the values of the three Euler angles around the z-axis, y-axis, x-axis, which are roll (γ), pitch (β), and yaw (α), respectively.

The objective function ($\Im$) to minimize between the first image (I_1) and the adjacent one (I_2) is:

$$y(\gamma, \beta, \alpha) = \Im(P^{I_1}, P^{I_2}, R, \overline{W}) = \sum_{i=1}^{N} \left| P_i^{I_1} - P_i^{I_2} \cdot R \right|^2 \cdot \overline{W_i} \quad (8)$$

where

$$\overline{W_i} = \left(\frac{\max(\mathbb{W}) - W_i}{\max(\mathbb{W})} \right)^2$$
$$P^{I_1} = \{P_1^{I_1}, P_2^{I_1}, .., P_N^{I_1}\}$$
$$P^{I_2} = \{P_1^{I_2}, P_2^{I_2}, .., P_N^{I_2}\}$$
$$\mathbb{W} = \{W_1, W_2, .., W_N\}$$
$$R = R_z(\gamma) R_y(\beta) R_x(\alpha)$$

The objective function in Eq. (8) is a function of the two-point clouds P^{I_1} and P^{I_2}, of the first and adjacent frames, respectively, the rotation matrix R and the weights of the features' descriptors $\overline{W}$. In particular, the output of Eq. (8) is given by the sum of the squared differences between the 3D Cartesian coordinates of each ith feature in I_1, (P^{I_1}), with the corresponding one in I_2, (P^{I_2}), multiplied by their normalized weights $\overline{W}$. In the optimization loop, the position of the second point clouds P^{I_2} changes each time because it is multiplied with a rotation matrix of design variables yaw (α), pitch (β), and roll (γ) angles. In Eq. (8) the ith weight W_i is obtained through the distance between the descriptors vectors of each pair of features found with the Block method. It gives more weight to a pair of points with a higher correspondence between their descriptors given by a smaller distance between them. The max ($\mathbb{W}$) returns the maximum weight value of the array from 1 to N, with N as the total number of valid features. It is used to normalize the weight and to use the weight parameter accordingly to the minimization process.

A lower and upper bounds on the design variables x were set so that a solution is found in the range $lb \leq x \leq ub$.

One possible operation to remove camera shake is to completely remove all camera rotations. This is possible if the positions and intentional rotations of the camera are fixed or almost fixed in time. In this case, for each frame, we could go back to the orientations of the first frame [21]. The rotation matrix applicable to each consecutive frame is a cumulative rotation matrix that sums all the contributions of the previous orientations. If the camera object has an imposed rotation movement in space, it will not be possible to relate all the frames with the first one. For this reason, a possible solution could be the use of a moving average. The length of the sliding window depends on the total number of frames and mainly on the context in which the camera is used [27]. In Fig. 6 is shown an example of the differences in yaw, pitch, and roll angles among 50 consecutive frames taken two at a time with the PSO algorithm. Figure 7 shows the cumulative rotations of angles of Fig. 6. In particular, for each angle in red, there are the cumulative rotations given by the sum of each previous rotations; in blue, there is a smooth curve found by using an average mean of length 12; in green, there is a difference between the previous two curves that allow to pass from the original to the smoothed trajectory.

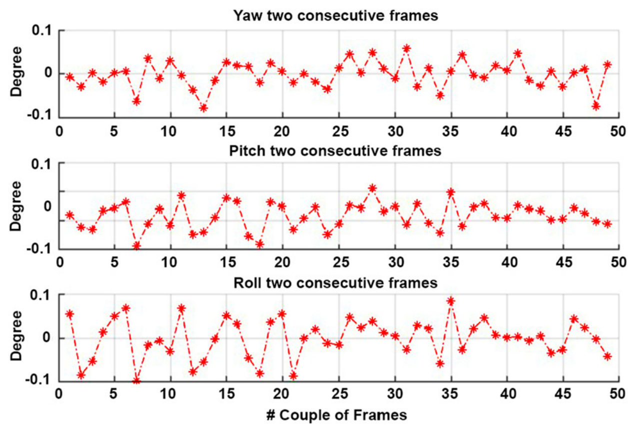

Fig. 6 Camera orientations between each consecutive frame

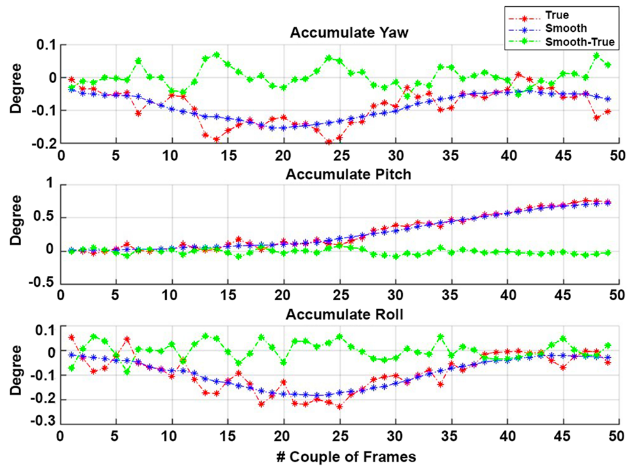

Fig. 7 Camera smoothed and cumulative rotations between each consecutive frame

In Fig. 8 the two original point clouds are displayed, in blue the key points related to the first frame, in red those related to the consecutive frame, plus the green points represent the new point cloud found by applying the rotation matrix with Euler angles, found through the PSO algorithm, to the consecutive point cloud. In the zoom area of Fig. 8

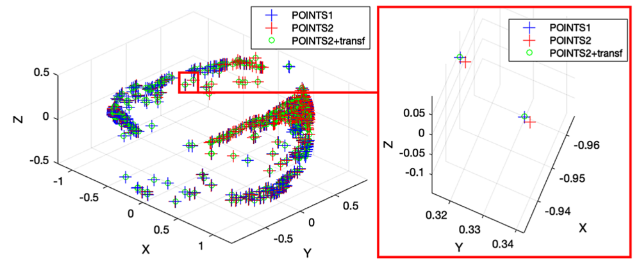

Fig. 8 Example of two point clouds related to the 3D feature positions of two consecutive frames (in red and blue) and the point cloud resulting from the PSO algorithm (in green). A zoom of a specific section is added on the right side

can be seen the result. The blue points that are related to the consecutive frame generate the green ones that are close to the points of the first frame.

3.2 Step 2: stabilized equirectangular images

After the estimations of the relative camera orientation were found, we apply the inverse rotation matrix to each image. This generates the new equirectangular images by rotating all pixels around the ZYX axis with the values found in the previous step.

Before applying the 3D rotation to each equirectangular image, we project all image pixels with their colors on the surface of a unitary radius sphere, see Eqs. (2–7). Due to the same size of each frame, the projection of all the equirectangular image's pixels to the surface of the sphere is done only the first time. For all the consecutive frames only the different pixel colors are updated. This helps to reduce the computational cost.

Rigid transformation, rotations are applied to each spherical frame keeping the position of the center of the sphere fixed. After that, the new 3D Cartesian coordinates of each pixel are converted to spherical coordinates by using the MATLAB $cart2sph$ function.

Between this new set of spherical coordinates and the 2D Cartesian coordinates of the equirectangular image to be generated there could not be an exact correspondence since some rotations could be lower than the angular resolution of the camera.

To avoid this problem the spherical coordinates of each pixel were generated from a colorless image, Eqs. (2–4). The color of each colorless pixel was then reconstructed by applying a weighted average among neighboring pixels of the previous rotated image, which on a color matrix corresponds to 8 pixels if the pixels near the corners are also taken into account.

To increase the relation and smooth between two consecutive frames a further filter was applied over time. It generates new frames updating the pixels colors from the average between two previous and two subsequent frames using an overall weight of 0.7 for the two frames nearby and 0.3 for those immediately after. These weights, although not critical values, were set with the highest value for the neighboring frames, because the farther away from the selected frame, the easier it is for the pixel color values to mismatch. In addition, to have the selected frame with the same weight between the previous and following frames, the sum of the weights was set to 1.

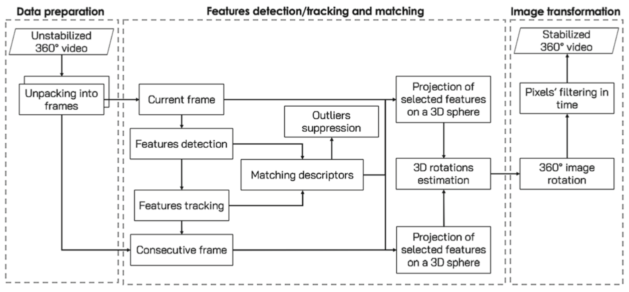

Fig. 9 General outline of the developed method

3.3 Step 3: 360° video production

After all the new equirectangular images were saved the $VideoWriter$ function in MATLAB was used to create a video file with MPEG-4 container format and a frame rate of 25 Hz.

Finally, the metadata was added to the movie in *.mp4 file extension to include 360° information with an external tool called Spatial Media Metadata injector v2.1[8]. Adding the metadata enables online platforms, such as YouTube, to warp into a sphere when watching the video.

The general outline of the developed method is schematized in the flowchart of Fig. 9.

4 Validation

A VR environment instead of a real environment was used initially as a gold standard as far as we know the result to achieve. It has allowed us to test different algorithms and approaches by verifying their results and robustness. In this section, we will show the results obtained after applying the stabilization process to virtual and real videos with our method.

4.1 Virtual videos

Each step of the previous chapter was validated within a VR environment in Unity 3D platform. A script was written to simulate a 360° camera. The 360° capture technique

[8] https://github.com/google/spatial-media/releases [Accessed: February 2022].

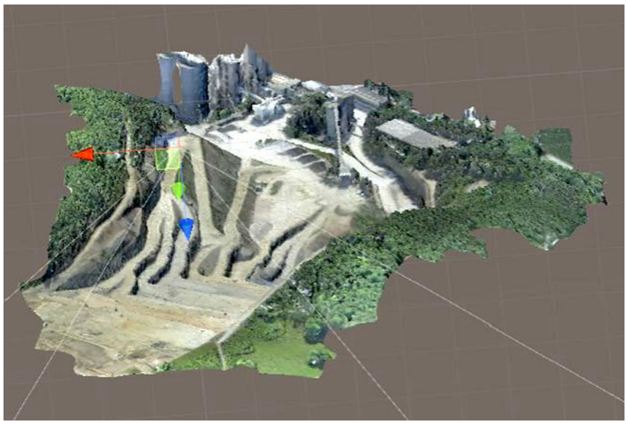

Fig. 10 Virtual scenario with the virtual mine model and the simulated 360° camera

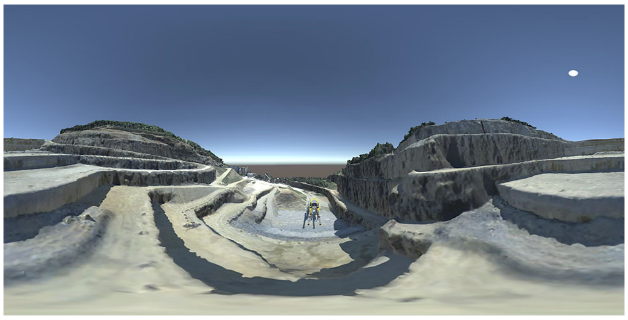

Fig. 11 Example of an equirectangular image acquired by the simulated 360° camera within the virtual scenario

is based on Google's Omni-directional Stereo (ODS) technology using cubemap rendering [34]. After the cubemap is generated, it is possible to convert this cubemap to an equirectangular map which is a projection format used by 360° video players. Placing the simulated camera inside the scene, Fig. 10, allows us to acquire an equirectangular image, Fig. 11.

The scene acquired, Fig. 11, is the one corresponding to a Wavefront 3D Object File (OBJ file extension) of the 3D high-resolution virtual environment of a mine previously imported in Unity.

The first validation was related to whether the uncertainty estimation of the feature's descriptors in Eq. (8) provides better results in terms of found rotations. To test it, 100 images were acquired in Unity by rotating the camera with random rotations in yaw, roll, and pitch between a ±0.2° interval while keeping the surrounding environment unchanged. The results of the PSO algorithm with these images are the same whether the cost function contains weight terms related to the uncertainty estimation or not. This happens because the SURF and the KLT algorithms always find robust features for small rotations in the absence of other disturbances in the environment. To make the images more realistic in the Unity platform, lighting changes have been applied to each frame. To simulate it in the Unity scene a directional light was applied. The directional light's rays are parallel and infinite in a specific direction making them suitable to simulate outdoor lighting as the real sun. This light influences all objects in the scene regardless of their distance, no matter how it is positioned. As well as the lights, the shadows were approximated with the command $Soft\ Shadows$, with softened edges. For shadows Unity uses a method called shadow mapping: in practice, it creates a depth map from the point of view of the light and uses it to decide where to cast the shadows. The color of the light was presumably fixed; instead, its intensity parameter was changed randomly, between 1 and 2, for each new camera's orientations. The intensity parameter was changed to simulate lighting changes and to force the PSO algorithm to increase variability. The swarm size of PSO was set to 30. Usually, the iterations process of the optimization ended because the relative change in the objective value over the last options of the maximum number of stall iterations (default 20) is less than the fixed-function tolerance (default 1e-6). The values of the SURF and KLT algorithm properties are chosen by default. In the Block method, the Block size of a local square neighborhood, centered at each interest point, was set to 11 by default.

Table 1 Errors in the use or non-use of the uncertainty estimation of features descriptors in the cost function: error sum without descriptors (A), error sum with descriptors (B)

Angle	A	B	A-B
YAW	0.1215°	0.1182°	−2.72%
PITCH	0.1241°	0.1224°	−1.37%
ROLL	0.1104°	0.1070°	−3.08%

Table 1 shows the values of the sum of the errors obtained using or not in the cost function the weights of the uncertainty estimation of the features descriptors on images with environmental disturbances previously discussed. How can be seen in Table 1, the results now change with the cost function; in particular, if we consider the term related to descriptors in cost function, we make a minor error on all the final rotations. Moreover, this difference in error sum values will become greater with real environments because they are subjected to more sources of disturbance.

The second validation concerns how the result changes with the suppression of outliers among the features through the modified Chauvenet criterion before using the PSO algorithm. The modified Chauvenet relation used is explained in Sect. 3.1. The validation was done with the same set of 100 images acquired in Unity in the simulated case where the environment is affected by disturbances. Table 2 shows that the sum of errors is lower with outliers suppression before the PSO. This means that the presence of outliers contributes to the generation of an error in the final angles found and it has a negative influence on the weight of a normalization

Table 2 Errors with the outliers suppression or not: error sum with descriptors (B), error sum with descriptors and with outliers suppression (C)

Angle	B	C	C-B
YAW	0.1182°	0.1157°	−2.12%
PITCH	0.1224°	0.1207°	−1.39%
ROLL	0.1070°	0.1066°	−0.37%

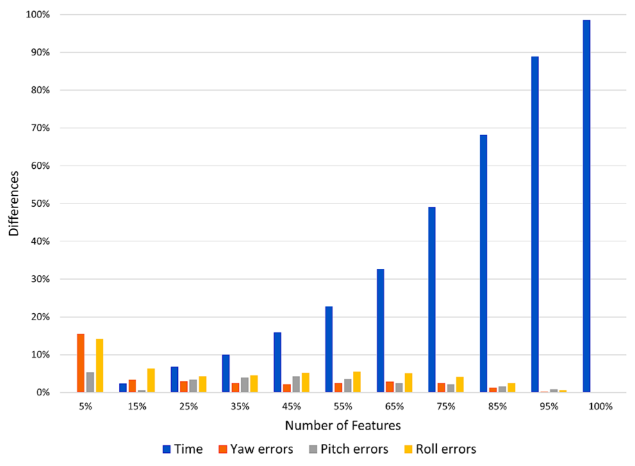

Fig. 12 Percentage differences in the final execution time of the 3D rotations estimation process and angle estimation errors as the number of features used increases

process because it refers to the maximum weight value. The expression of the cost function used to extract all the data in Table 1 always contains the weights of the estimated uncertainty of the feature descriptors because in this case we just want to quantify how the errors in rotations change if the outliers are suppressed using the Chauvenet criterion or not.

Figure 12 shows the percentage difference in time and yaw, pitch, and roll errors by changing the number of features used for the optimization process. Before the selection, the features are sorted with decreasing weight in such a way the first set of 5% concerns the best and gradually all the others are added with their weights. Among the sorted weights, there are no outliers discarded with the previous method. From the results of Fig. 12 it is clear how using all the features with their weights, from the point of view of angles' errors, is better than considering only a small set at the expense of increasing final optimization time. Although 15% of the features' number has a smaller error value for the pitch, the best result of all angles is at 100%. This means that for an offline stabilization process, where time can be longer, it is better to consider all the features.

To generate the frames used for the previous analyses, we randomly rotated the simulated 360° camera in a controlled virtual environment and then verified the effectiveness of our method by checking the errors on the 3D rotation results. Since the real videos we had available for further analysis in

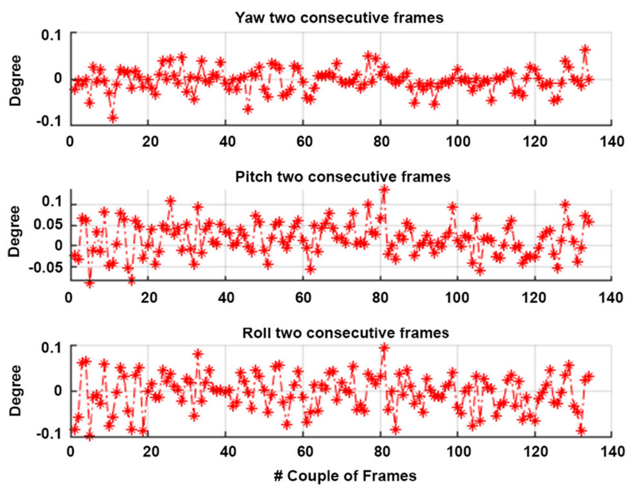

Fig. 13 Camera orientations between each consecutive frame estimated from a real video acquired with a drone. These 3D rotations are reproduced and used as ground truths in the simulated virtual video

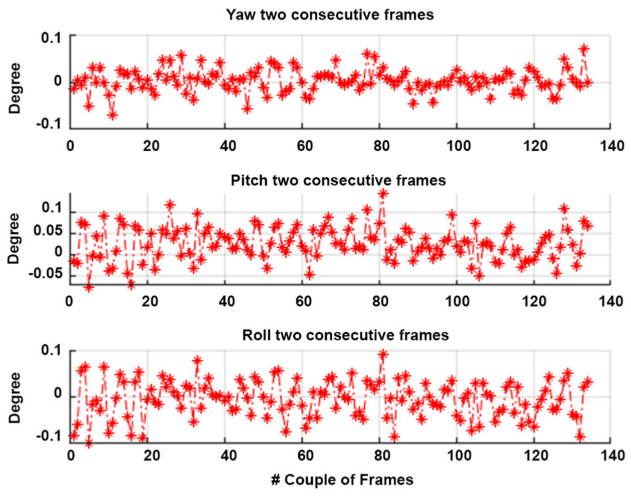

Fig. 14 Camera orientations estimated with our method from the generated virtual video with translation

Sect. 4.2 are based on acquisitions made with a real drone equipped with a 360° camera, we tried to estimate the drone shaking to replicate it in a virtual environment as well. We initially estimated the 3D rotations on the frames of the real video by applying our method and assumed them to be true. Then we verified their correctness and thus of our algorithm by replicating these rotations in a simulated environment and estimating the same data another time but now in a controlled environment. In Fig. 13 are shown the 3D rotations estimated from a real video acquired from a drone.

We generated two virtual videos: for the first video, the position of the camera was fixed and only the rotations of Fig. 13 were applied; for the second virtual video also a camera motion in the straight forward direction with a velocity of 0,5 m/s was applied. From the two arrays of frames saved, 360° virtual videos were produced by using a *VideoWriter* object in MATLAB.

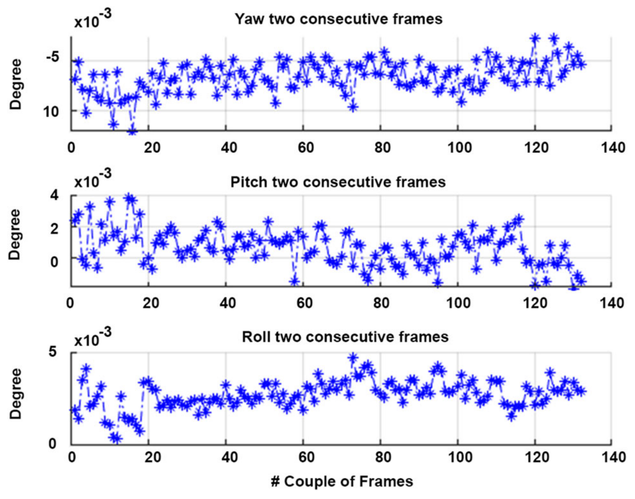

Fig. 15 Camera orientations estimated with our method on the stabilized virtual video affected by rotations and translation

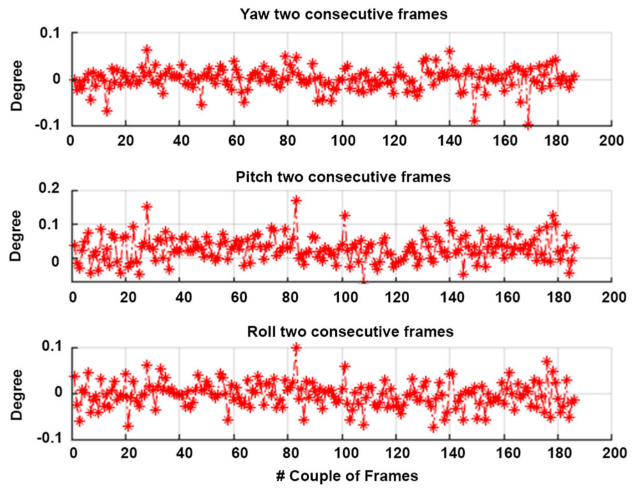

Fig. 17 Camera orientations estimated with our method on non-stabilized real video acquired with a drone

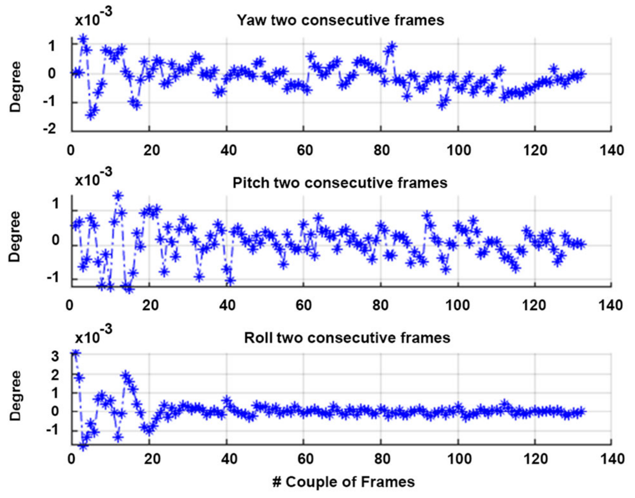

Fig. 16 Camera orientations estimated with our method on the stabilized virtual video affected only by pure rotations

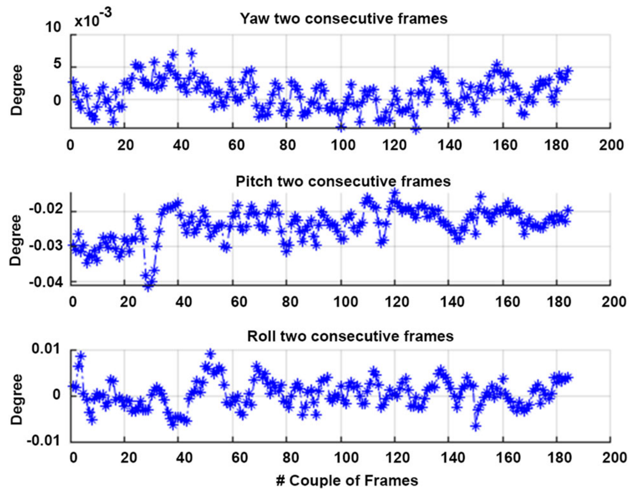

Fig. 18 Camera orientations estimated with our method on stabilized real video with our developed approach

The aforementioned method of Sect. 3 was applied to both videos. The first step of our method is to look for camera orientations between each pair of frames. The camera orientations found from the first video where the camera position is fixed perfectly match the input 3D angles of Fig. 13 provided to the camera in Unity. It shows the goodness of the PSO algorithm.

Figure 14, which represents the camera orientations estimated with our method for the second video, shows a similarity with the pure rotations of Fig. 13. It proves the robustness of the PSO algorithm in estimating 3D rotations even in panoramic videos where small translations are present.

Once the equirectangular image sets of the two input videos were stabilized, we produced the two new stabilized videos.

In addition, to quantify how stabilized the new videos are, we applied the PSO algorithm to find the new camera rotations. The result for the first stabilized video is shown in Fig. 16 and for the second one in Fig. 15. How it can be seen in Figs. 16 and 15, the oscillations due to the stabilization process have been reduced by an order of magnitude, so that the final video results much smoother.

4.2 Real videos

In a real environment compared to a virtual one, other variables come into consideration, such as the multiple movements, occlusions, lighting changes, and non-rigid movements. In this section, we want to verify the correct performance of the implemented method with real 360° videos.

Table 3 Total method execution time

Stage	Time/frame
Image saving:	150 ms
Camera relative rotations:	
Features detection (SURF)	150 ms
Points tracking (KLT)	71 ms
Block method	10 ms
Modified Chauvenet criterion	4800 ms
Point cloud generation	1,5 ms
Particle Swarm Optimization (PSO)	3300 ms
Stabilize equirectangular images:	
Equirectangular to spherical image	5000 ms
Image rotation	40 ms
Pixels coloring	20,000 ms
Pixels' filtering in time	16,000 ms
	Time
Video input reading:	290 ms
Image to video:	7500 ms
Adding metadata file:	100 ms

We tested our method on a large number of spherical panorama videos captured with the DJI Matrice 210 RTK drone[9] equipped with an Insta360 camera.

To show the goodness and robustness of the method proposed in this paper we analyzed an example of a short real video in the worst-case undergoing high shaking. The camera yaw, pitch, and roll trajectories before and after the stabilization process can be seen, respectively, in Figs. 17 and 18.

As can be seen in Fig. 18 after the stabilization process, the camera rotations are consistently less. As in the case of a virtual environment, the result is smooth enough to make viewing comfortable, even with a headset. Both videos have a frame rate of 25 Hz.

Given the short duration of this example and the type of panoramic shot with slow intentional rotations, camera rotations are removed by comparing them to the first frame.

The average times per frame spent in each step of the proposed method are shown in Table 3. The tests were run on a PC with an Intel(R) Core(TM) i7-9700KF CPU @ 3.60GHz processor and 64 GB of RAM, on 2880x5760 equirectangular video frames.

5 User study

We tested the impact of our approach to 360° videos presented in VR. We were interested in the capability of our method to reduce symptoms of cybersickness.

[9] https://www.dji-store.it/prodotto/dji-matrice-210-rtk-g/ [Accessed: February 2022].

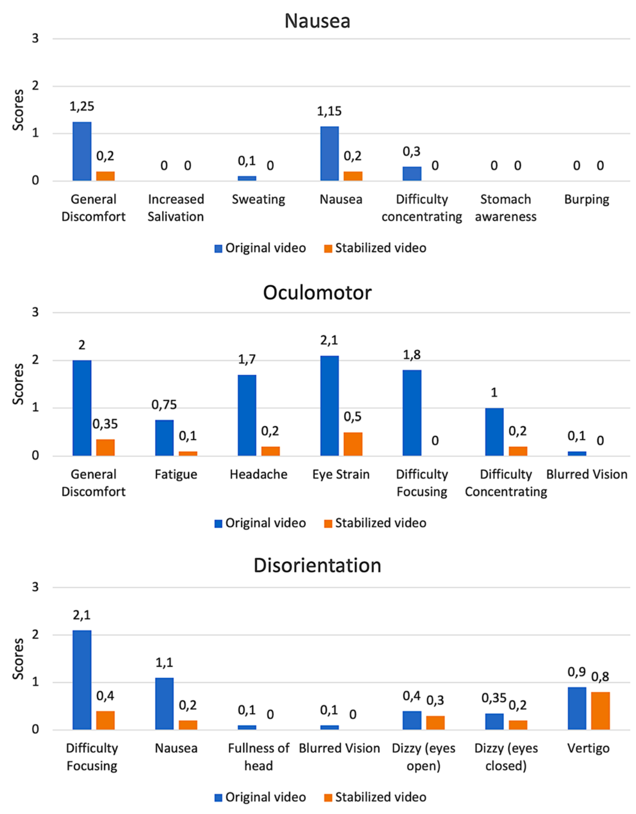

Fig. 19 Simulator Sickness Questionnaires results

Task We designed a task, in which a user watches several 360° videos in a VR environment. In particular, each participant was asked to watch a total of five minutes of 360° unprocessed videos, followed by watching a total of five minutes of 360° video that is the result from the proposed stabilization process. We scheduled a one-hour break between each session to avoid compound effects between both conditions.

Apparatus We used an Oculus Quest 1 VR Headset[10] in both conditions. After completing a consent form and demographics questionnaire, the participant was introduced to the system. We recruited twenty (20) participants for the experiment. Participants include 3 females and 17 males: 12 people with ages between 18 and 24 years, and 8 people with ages between 25 and 35 years; 10 people reported to have experience with VR. All participants were free from any known neurological disorders, as verified by self-report.

Procedure After completing each session, the participants filled out a Simulator Sickness Questionnaire (SSQ) [35]. Participants were asked to add notes wherever they feel it is necessary.

Results Figure 19 shows the SSQ results in terms of the scores for symptoms related to their specific aspects of nausea, oculomotor, disorientation. The values range from 0 to

[10] https://www.oculus.com/quest [Accessed: February 2022].

3 accordingly to the effect of each item: None (0), Slight (1), Moderate (2), Severe (3).

Furthermore, we compute a total score (TS) to represent the overall severity of cybersickness experienced by the users. We follow the approach of Walter et al. [36] for calculation TS as the weighted sum of nausea, oculomotor, and disorientation. We measure a total score of **64.70** for the raw 360° video condition and a score of **13.65** for the stabilized 360° video.

Discussion From the SSQ scores it can be noticed how the trend of all the symptoms is decreased in the sessions with stabilized videos when compared to the ones with original videos. In particular, we believe that reduced camera shaking affects the oculomotor and disorientation aspects.

Vertigo's score remains almost unchanged in both sessions due to the videos showing a panoramic video shot with a drone. The remaining general discomfort score in the stabilized video is connected with the 360° videos which slowly rotate and translate even if the participants do not move. This is confirmed by the participants' notes. Their notes also show how all these symptoms grow over time when using the VR Headset.

6 Conclusion

The stabilization of a 360° video is increasingly required because viewers, especially when it comes to immersive visual environment, are dizzy or nauseous due to shaky scenes. This paper presents a new approach to stabilize the 360° videos affected by the problem of shaking to alleviate cybersickness. Our approach is 2.5D because it estimates 3D rotations without involving 3D structure from motion methods.

The method's validation was achieved using real 360° videos captured by a drone equipped with an omnidirectional camera during a mine overflight. Instead, the virtual videos were simulated and produced in a VR environment using Unity 3D platform. Working in a simulated virtual environment allowed us to have a reference to test the goodness of each step of our approach knowing in this context the result to achieve.

With the method proposed in this paper, it is possible to remove shaking from videos without knowing the initial camera position and its motion. The possibility to select the ROI area where to analyze the video and to define the length of the window to interpolate differently the camera's orientations in time makes possible the use of this method in different contexts. The selection of the ROI area allows us to stabilize also videos with static overlays, logos.

The periodical selection of two frames at a time avoids the possibility of accumulation errors in the estimation of camera rotation angles. The detection and tracking of interest points, respectively, with the SURF and KLT algorithm, the use of a PSO algorithm, for their matching using their descriptors similarity in the cost function and the outliers suppression through the modified Chauvenet's Criterion, makes our method accurate and more robust than other works. Finally, the time-weighted color filter applied to each frame gives the possibility to handle videos with small translational jitter, rolling shutter wobble, parallax, and lens deformation.

Through the user study, we proved how a good stabilization can reduce all the simulator sickness symptoms that can be summarized with the final TS value. From the user study, this value was 64.70 before the stabilization and 13.65 after.

One main drawback of our approach is the computation time, which makes our method suitable for non-real-time applications. In addition, our algorithm is unable to stabilize videos in which there is a predominance of translations over camera rotations. Furthermore, it is based on features tracking between frames to estimate 3D camera rotations; if light is scattered or flared in the camera lens, it can generate severe lens flare that generates inaccurate estimates leading to poor stabilization.[11]:

Acknowledgements This paper was developed inside the European project MiReBooks Mixed Reality Handbooks for Mining Education, a project funded by EIT Raw Materials.

Funding Open access funding provided by Università degli Studi di Trento within the CRUI-CARE Agreement.

Declarations

Conflict of interest The authors declare to have no potential conflict of interest. This work was enabled by a research grant of the European Institute of Innovation and Technology (EIT) Raw Materials in the project MiReBooks: Mixed Reality Handbooks for Mining Education (18060).

[11] https://eitrawmaterials.eu/project/mirebooks-mixed-reality-handbooks-for-mining-education/, [Accessed February 2022].

References

1. Milgram, Paul, Kishino, Fumio: A taxonomy of mixed reality visual displays. IEICE TRANSACTIONS on Information and Systems **77**, 1321–1329 (1994)
2. Bell, J.T., Fogler, H.S.: The investigation and application of virtual reality as an educational tool, In: Proceedings of the American Society for Engineering Education Annual Conference, pp. 1718–1728 (1995)
3. Luchetti, A., Tomasin, P., Fornaser, A., Tallarico, P., Bosetti, P., De Cecco, M.: The human being at the center of smart factories thanks to augmented reality, In: 2019 IEEE 5th International Forum on Research and Technology for Society and Industry (RTSI), pp. 51–56 (2019)
4. Butaslac, I.III., Luchetti, A., Parolin, E., Fujimoto, Y., Kanbara, M., De Cecco, M., Kato, H.: The feasibility of augmented reality as a support tool for motor rehabilitation, In: International Conference on Augmented Reality, Virtual Reality and Computer Graphics, pp. 165–173, Springer (2020)
5. Raptis, G.E., Fidas, C., Avouris, N.: Effects of mixed-reality on players behaviour and immersion in a cultural tourism game: A cognitive processing perspective. International Journal of Human-Computer Studies **114**, 69–79 (2018)
6. Hebbel-Seeger, Andreas: 360 degrees video and VR for training and marketing within sports. Athens Journal of Sports **4**, 243–261 (2017)
7. Argyriou, L., Economou, D., Bouki, V.: 360-degree interactive video application for Cultural Heritage Education, In: 3rd Annual International Conference of the Immersive Learning Research Network, Verlag der Technischen Universität Graz (2017)
8. Guervós, E., Ruiz, J.J., Pérez, P., Muñoz, J.A., Díaz, C., García, N.: Using 360 VR video to improve the learning experience in veterinary medicine university degree, In: Electronic Imaging, pp. 217–1, Society for Imaging Science and Technology (2019)
9. Stanney, K.M., Kennedy, R.S., Drexler, J.M.: Cybersickness is not simulator sickness, In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol 41, pp. 1138–1142, SAGE Publications Sage CA: Los Angeles, CA (1997)
10. Litleskare, S., Calogiuri, G.: Camera stabilization in 360° videos and its impact on cyber sickness, environmental perceptions, and psychophysiological responses to a simulated nature walk: a single-blinded randomized trial. Front. Psychol. **10**, 2436 (2019)
11. Bonato, Frederick, Bubka, Andrea, Palmisano, Stephen: Combined pitch and roll and cybersickness in a virtual environment. Aviation, Space Environ. Med. **80**(11), 941–945 (2009)
12. Perozzi, Gabriele: Efimov, Denis, Biannic, Jean-Marc, Planckaert, Laurent: Trajectory tracking for a quadrotor under wind perturbations: sliding mode control with state-dependent gains. Journal of the Franklin Institute **355**, 4809–4838 (2018)
13. Zhang, Xuebo, Fang, Yongchun, Liu, Xi.: Motion-estimation-based visual servoing of nonholonomic mobile robots. IEEE Transactions on Robotics **27**(6), 1167–1175 (2011)
14. Li, S., Wang, F., Shi, T., Kuang, J.: Probably secure multi-user multi-keyword searchable encryption scheme in cloud storage, In: 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), pp. 1368–1372 (2019)
15. Huang, T.S., Netravali, A.N.: Motion and structure from feature correspondences: A review, In: Advances in Image Processing and Understanding: A Festschrift for Thomas S Huang, pp. 331–347 (2002)
16. Nguyen, Nhat-Tan., Laurendeau, Denis, Branzan-Albu, Alexandra: A robust method for camera motion estimation in movies based on optical flow. International Journal of Intelligent Systems Technologies and Applications **9**(3–4), 228–238 (2010)
17. Kamranian, Zahra, Sadeghian, Hamid, Mehrandezh, Ahmad Reza Naghsh Nilchi Mehran.: Fast, yet robust end-to-end camera pose estimation for robotic applications. Applied Intelligence **51**(6), 3581–3599 (2021)
18. Kamali, M., Banno, A., Bazin, J.-C., Kweon, I.S., Ikeuchi, K.: Stabilizing omnidirectional videos using 3d structure and spherical image warping, IAPR MVA, 1, 2, Citeseer (2011)
19. Liu, Feng, Gleicher, Michael, Jin, Hailin, Agarwala, Aseem: Content-preserving warps for 3D video stabilization. ACM Transactions on Graphics (TOG) **28**, 1–9 (2009)
20. Buehler, C., Bosse, M., McMillan, L.: Non-metric image-based rendering for video stabilization, In: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, 2, II–II (2001)
21. Shen, L.-C., Huang, T.-K., Chen, C.-S., Chuang, Y.-Y.: A 2.5 d approach to 360 panorama video stabilization, In: 2018 25th IEEE International Conference on Image Processing (ICIP), pp. 3184–3188 (2018)
22. Liu, Shuaicheng, Yuan, Lu., Tan, Ping, Sun, Jian: Bundled camera paths for video stabilization. ACM Transactions on Graphics (TOG) **32**, 1–10 (2013)
23. Kabsch, W.: A solution for the best rotation to relate two sets of vectors. Acta. Crystallograph. Sect. A Cryst. Phys. Diffract. Theor. General Crystallograph. **32**, 922–923 (1976)
24. Kopf, Johannes: 360 video stabilization. ACM Transactions on Graphics (TOG) **35**, 1–9 (2016)
25. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**, 381–395 (1981)
26. Tang, C., Wang, O., Liu, F., Tan, P.: Joint stabilization and direction of 360° videos. ACM Trans. Graph. (TOG) **38**, 1–13 (2019)
27. Kasahara, S., Nagai, S., Rekimoto, J.: First person omnidirectional video: System design and implications for immersive experience, In: Proceedings of the ACM International Conference on Interactive Experiences for TV and Online Video, pp. 33–42 (2015)
28. Lai, W.-S., Huang, Y., Joshi, N., Buehler, C., Yang, M.-H., Kang, S.B.: Semantic-driven Generation of Hyperlapse from 360 Video Supplementary Material
29. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features, In: European Conference on Computer Vision, pp. 404–417, Springer (2006)
30. Tomasi, C., Kanade, T.: Detection and Tracking of Point Features. Carnegie Mellon Univ. Pittsburgh, School of Computer Science (1991)
31. Bradski, G., Kaehler, A.: Learning OpenCV: Computer Vision with the OpenCV Library, O'Reilly Media, Inc. (2008)
32. Blaedel, W.J., Meloche, V.W., Ramsay, J.A.: A comparison of criteria for the rejection of measurements. J. Chem. Edu. **28**, 643 (1951)
33. Trelea, I.C.: The particle swarm optimization algorithm: convergence analysis and parameter selection. Inform. Process. Lett. **85**, 317–325 (2003)
34. Google Inc.: Rendering Omni-directional Stereo Content https://developers.google.com/vr/jump/rendering-ods-content.pdf, [Accessed: 14/10/2020]
35. Kennedy, R.S., Lane, N.E., Berbaum, K.S., Lilienthal, M.G.: Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness. Int. J. Aviation Psychol. **3**, 203–220 (1993)
36. Walter, H., Li, R., Munafo, J., Curry, C., Peterson, N., Stoffregen, T.: APAL Coupling Study (2019)

Alessandro Luchetti received his BSc in Mechanical Engineering at the Polytechnic University of Marche (Italy) and specialized in Mechatronics Engineering at the University of Trento (Italy) in both cases with honors. His research activity based on Mixed Reality technologies was partly undertaken at the NAIST Institute, Japan. After a period of three years as Research Fellow he is now enrolled in the PhD school at the Department of Industrial Engineering of University of Trento. His PhD focuses on the study of how innovative technologies can augment human capabilities, well-being and safety in different fields.

Matteo Zanetti received the MS degree in Mechatronic Engineering in 2014, and the Ph.D. degree in Materials, Mechatronics, and System Engineering in 2019, both from the University of Trento. During his PhD, he was involved, within the AUSILIA project, in the development of motion capture and stress measurement algorithms for subjects with physical and/or cognitive disabilities, contributing to the creation of a virtual reality interface for therapists. Currently, his research activity focuses mainly on the development of algorithms for 3D vision systems and technologies in Mixed Reality related to the European project MiReBooks.

Denis Kalkofen is currently an assistant professor with the Institute of Computer Graphics and Vision (ICG), Graz University of Technology, Austria. Before joining ICG, he was a member of the Virtual Reality Laboratory at the University of Michigan, Ann Arbor, Michigan. In 2019, he joined the Wearable Computer Laboratory at the University of South Australia, Australia, as visiting researcher and the Computational Imaging Laboratory at Stanford University, Stanford, California, as visiting assistant professor. His research interest includes visual computing for developing visualization, interaction, display and authoring techniques for mixed reality environments.

Mariolino De Cecco is Associate Professor at the Department of Industrial Engineering—University of Trento, and is Head of the MIRo laboratory and coproposer and member of the steering committee of the autonomous province of Trento project, AUSILIA. His research fields are about Measurements, Robotics and Space. He is referee of different international papers and congress, and of different national and international research projects and author of more than 190 papers on international journal with referee, book chapters, papers on national or international congress with referee, H-index 33.