Optical Flow-based Video Completion in Spherical Image Sequences

Binbin Xu, Sarthak Pathak, Hiromitsu Fujii, Atsushi Yamashita and Hajime Asama

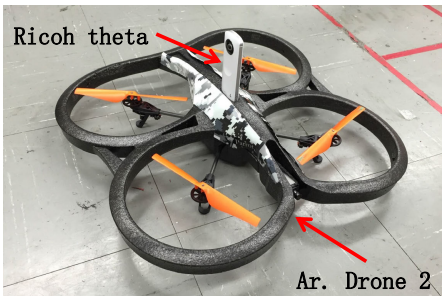
***Abstract*— Spherical cameras are widely used for robot perception because of their full 360 degree fields of view. However, the robot body is always seen in the view, causing occlusions. In this paper, we propose a video completion method which is able to remove occlusions fast and recover the occluded background accurately. We first estimate motion in a 2D dense spherical optical flow field. Then we interpolate the motion in the occlusion regions by solving least square minimization problems using polynomial models. Based on the interpolated motion, we recover the occluded regions by tracing the optical flow trajectory to find the corresponding pixels in other frames and warping them back to fill in the occlusions. We also provide a simple, yet fast solution to effectively remove occlusions in all regions of the image by utilizing the continuity of field of view of spherical images. In experiments, quantitative methods are conducted to demonstrate the effectiveness and efficiency of our proposed method by comparing the results with those from state-of-the-art methods.**

I. INTRODUCTION

Spherical videos have a wide variety of applications in robotics and entertainment, such as 3D environment reconstruction [1] and virtual reality [2]. They can acquire the whole scene from every direction in space simultaneously. However, a common problem existing in these applications is that anything attached to camera, such as robot body or photographer body, always appears in view, causing undesired regions. For example, when mounting a spherical camera on a drone, as is shown in Fig. 1(a), the drone body occludes some parts of field of view from the camera. This causes severe occlusions and one example of such occluded image frame can be seen in Fig. 1(b). In addition, in 3D reconstruction applications [1], spherical cameras are fixed on robot bodies and undesired robot platform occlusions have to be ignored by manually putting a mask to avoid mismatching of point correspondences. In virtual reality applications [2], the photographer's head is always in the center of videos, occluding users' view and leaving blind holes after being deleted. Furthermore, difference in motion between occlusions (photographer's head) and background causes discontinuity for visualization. Therefore, removing undesired occlusions to recover the true background information, as shown in Fig. 1(c), is necessary and desired for applications involving spherical cameras.

In this paper, a new method is proposed to complete occlusions in panoramic video captured by a freely moving camera. The proposed method is based on the assumptions that background environment is static and information behind occlusions is visible in at least one of other frames. The procedure is summarized as follows. First, the occlusion region is rotated to the center of the equirectangular image frame to minimize image distortion. Then the 2D dense spherical optical flow field is calculated from the rotated panoramic image sequences. The optical flow field in and near occlusions is removed to avoid the influence from occlusions. In addition, the continuity of field of view of spherical images is utilized here to effectively remove occlusions in all regions of the image. Based on the estimated dense optical flow field, two polynomial models are used to model

The authors are with Department of Precision Engineering, Graduate School of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, 113-8656, Japan {xubinbin, pathak, fujii, yamashita, asama}@robot.t.u-tokyo.ac.jp



(a) An example of mounting a spherical camera, Ricoh Theta, on the body of an AR.Drone 2.

(b) An original equirectangular image from the mounted Ricoh Theta. Here, the drone body severely occludes the camera vision.

(c) A completed result, in which the drone body is removed, by the proposed method.

Fig. 1. Example of removal of occlusions in spherical images by the proposed method (from 6th video frame in our experiment).

object motion and then the motion in occluded regions is interpolated based on the models. Based on the interpolated motion, corresponding pixels are found in other frames by tracing along the optical flow trajectory to fill in occlusions. Our main contributions in this paper is using the continuity of spherical images in estimating and interpolating optical flow robustly and warping image frames iteratively to find corresponding pixels in further frames accurately.

The remainder of this paper is organized as follows. Section II discusses the related work of video completion on panoramic and normal videos. Section III explains the spherical camera model and the motion estimation method we propose. Section IV describes how we interpolate motion and use that to inpaint occlusions. Section V shows the quantitative experiments and compares our results with prior methods, followed by the conclusion in the last section.

II. Related work

Some video completion methods have been proposed for occluded regions in panoramic videos before. Kawai *et al.* proposed methods to fill in missing regions by searching for similar exemplars in projected image planes from other frames [3] and by aligning multiple frames based on a reconstructed 3D model [4]. However, the former can only work under the assumption that occluded region is planar in all image frames, which limits the applicability. The latter severely relies on a previously reconstructed 3D model and thus is not feasible without 3D reconstruction model. Paredes *et al.* [5] rectify omnidirectional images to perspective ones and then implement an image inpainting method [6] on each frame separately, without using information from other frames. Thus, such methods lose temporal coherence and lead to wrong background recovery .

In addition to spherical videos, many video completion methods have been proposed on perspective videos. They can be categorized into two classes: ones searching for the most similar exemplars in all video frames [7][8] and others using motion information to find corresponding pixels in other frames [9][10].

The former search-based methods [7][8] define the similarity between exemplars, which are spatio-temporal cubes, and then search for the most similar exemplars in all video frames. The most similar exemplars are used to replace the occluded ones. While these methods can generate plausible results, the occluded background cannot always be found as the most similar patch, even if it does appear in other frames. In addition, such rigid cubes make it difficult to capture appearance changes resulting from the distortion of spherical images. As a result, these methods can lead to wrong output and unavoidably take much processing time for searching.

On the other hand, the latter motion-based methods estimate motion information in occlusion regions and find the corresponding pixels of occlusions in other frames. Some methods, like Yamashita *et al.* [9], estimate the camera motion and then complete missing regions by searching along the movement trajectories in neighboring frames. However, these methods require a fixed camera center to find corresponding pixels in other frames properly and thus cannot be applied to a freely moving camera. To deal with occlusions in moving cameras, some other methods, like You *et al.* [10] estimate motion in occluded regions by interpolating optical flow and then find the correspondences in other frames using interpolated optical flow. However, these methods rely on a linear approximation of optical flow in adjacent frames and thus are not suitable for the motion pattern in spherical images. Moreover, at the borders of images, the interpolation turns into an extrapolation situation due to the luck of information beyond the borders. The extrapolation may lead to falsely estimated motion, as will be seen later in Fig. 7(b). In summary, it is difficult to successfully remove occlusions in panoramic videos from a freely moving camera. We solve the above problems by considering the continuity of spherical images and image warping instead of using linear approximations.

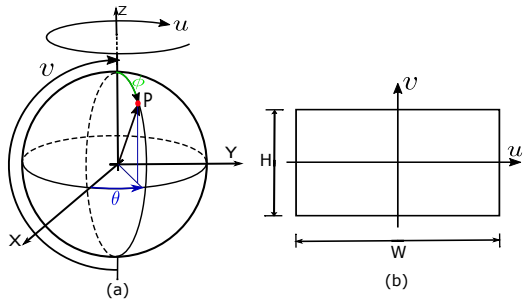


Fig. 2. Spherical image in equirectangular projection. (a) Spherical coordinates. (b) Equirectangular projection

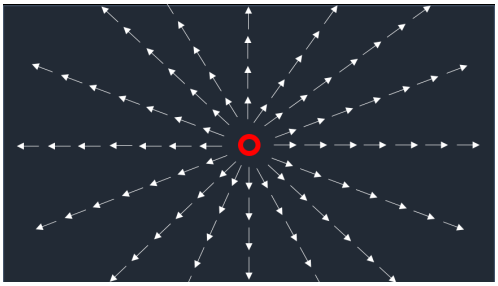
III. Motion Estimation

A. Properties of spherical images

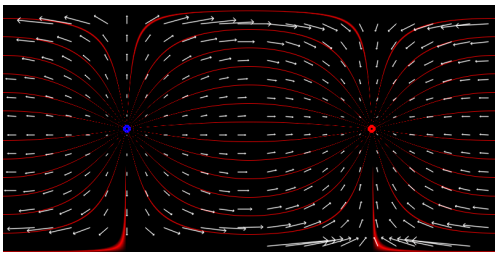
Here we first formulate the problems caused by spherical imaging. An image captured by a spherical camera is a projection to a spherical image plane (Fig. 2(a)) and usually shown as an equirectangular form for 2D visualization (Fig. 2(b)). Here we describe pixels on spherical image plane in the spherical coordinate, (r, θ, ϕ), with unit radius r, polar angle θ, and azimuthal angle ϕ. Since all pixels move on an unit sphere, the expression in equirectangular images can avoid defining 3D calculations in Cartesian coordinates. Each point in spherical images can be expressed on equirectangular images

$$\theta = \frac{2\pi}{W}u, \quad \phi = \frac{\pi}{H}v, \tag{1}$$

where W and H are the width and height of equirectangular image, respectively, and u and v are the corresponding 2D Cartesian coordinates on the equirectangular images. They have the relationship $W = 2H$. The evenly spaced horizontal and vertical space in equirectangular image leads to distortion. The distortion also varies according to the v coordinate. Regions near the poles have more severe distortion than regions near the equators. This distortion phenomenon increases the difficulty in estimating the motion near the poles.



(a) Motion pattern in normal images.



(b) Motion pattern on spherical images.

Fig. 3. Motion pattern difference for camera going straight forward in normal images and spherical images.

In addition to the distortion on each single spherical image, the motion pattern is also special in spherical image frames. It has two properties. First, pixels on spherical images can only move along the spherical surfaces. Such movements are projected as curves on the equirectangular images. Take a movement of camera going straight forward as an example. In normal images, as is shown in Fig. 3(a), correspondences (white vectors) are appearing as a linear trajectory from the vanishing point of the translation direction (red circle). On the contrary, in the equirectangular images, as is shown in Fig. 3(b), correspondences (white vectors) move along projected curves, from one vanishing point (blue circle) of the translation direction to another (red circle). The epipolar curves are visualized as red curves, along which the point correspondences move. Furthermore, in normal images where the field of view is limited, it is not possible to estimate the motion of points near the image borders due to the lack of information outside of the borders. On the other hand, in spherical images where the field of view is full, correspondences will not move out of the field of view and thus can be found for points in all areas. Even if the points near the equirectangular image borders, as can be seen in Fig. 3(b), the correspondences can be found in the opposite borders because of the continuity of spherical images.

B. Decreasing the occlusion distortion

Occlusions, in most cases, appear near the bottoms of images. This is caused by the fact that the bottoms of cameras are usually fixed to other things, for example, held by hand or attached to robots. One example can be seen in Fig. 1(b), where the bottom of the camera was plugged in to the drone body. However, as we mentioned before, the bottom images have more severe distortion. It is difficult to accurately measure the motion there using existing algorithms which were designed for normal images.

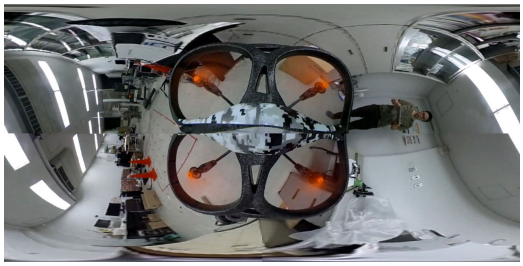


Fig. 4. Decreased distortion on occlusion regions by image rotation

To deal with this severe distortion problem, we first rotate the image to bring the occlusions to the image center, where distortion is similar to that of a perspective projection image. The rotation is performed based on the spherical image and then converted back to the equirectangular image based on Eq. (1). The image after rotation is shown in Fig. 4, where distortion in occluded region is decreased significantly by compared with image being rotated before, in Fig. 1(b).

C. Spherical Optical Flow

Here we describe how we estimate motion on spherical images using 2D optical flow fields. We use $\mathbf{V}_x$ to denote an optical flow field in the direction of x. Although a spherical image has three dimensions, it also has a constraint that radius is constant and thus its motion can be expressed in 2D spherical optical flow vectors, $\mathbf{V}_\theta$ and $\mathbf{V}_\phi$. Considering the image distortion, we first use SIFT flow [11] to obtain the dense image correspondences $\mathbf{V}_u$ and $\mathbf{V}_v$ on equirectangular images. SIFT flow is a dense optical flow method based on SIFT features [12], invariant to scale, rotation and perspective transformation and thus can find the correspondences even in distorted images. Then we obtain $\mathbf{V}_\theta$, $\mathbf{V}_\phi$ from $\mathbf{V}_u$, $\mathbf{V}_v$ using Eq. (2).

$$\mathbf{V}_\theta = \frac{2\pi}{W}\mathbf{V}_u, \qquad \mathbf{V}_\phi = \frac{\pi}{H}\mathbf{V}_v \tag{2}$$

This way, instead of calculating optical flow vectors on 3D sphere images, we can obtain a quadratic computation complexity in the two dimensions.

However, some problems may occur in the motion estimation process on equirectangular images. For example, objects which move across the right and left borders of the image appear on the corresponding opposite border due to the 360 degree field of view. It shows an apparent large displacement across the equirectangular image whereas the true displacement on a spherical image is actually small. Most existing optical flow algorithms cannot deal with such phenomenon of large displacement because the objective functions in most of them constrain the optical flow vectors to be as small as possible, which is useful for perspective images, but insufficient for capturing true movement in equirectangular images.

To address this problem, we use a simple method using the continuity of the spherical image. We extend the equirectangular image on both sides by copying redundant, connected

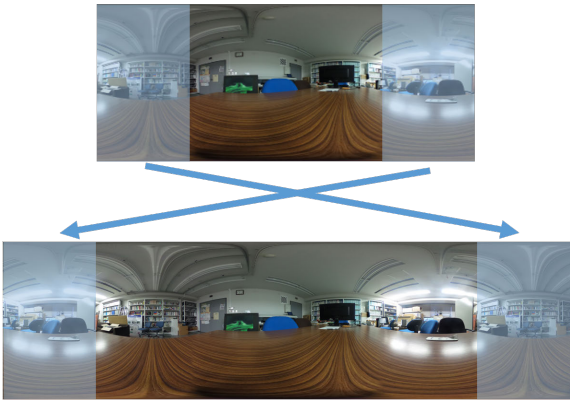


Fig. 5. Image extension to address the problem of large displacement and extrapolation

information from the corresponding opposite side to enhance a natural continuity in the image. As shown in Fig. 5, we can copy the u coordinate $(-\frac{W}{2}, -\frac{W}{2}+\triangle u)$ from the original equirectangular image to the u coordinate $(\frac{W}{2}, \frac{W}{2}+\triangle u)$ in the extended equirectangular image. Similarly, region in u coordinate $(\frac{W}{2}-\triangle u, \frac{W}{2})$ from the original equirectangular image can be copied to the u coordinate $(-\frac{W}{2}-\triangle u, -\frac{W}{2})$ in the extended equirectangular image. The parameter $\triangle u$ is chosen to be large enough to cover the possible displacements. Now, on the extended image sequences, we apply the SIFT Flow algorithm to get the optical flow fields $\mathbf{V}_u$ and $\mathbf{V}_v$. After all required computations, we remove the extended regions and revert back to the original equirectangular image.

IV. Motion Interpolation

A. Polynomial modeling

Based on the spherical optical flow calculated on the whole image, we interpolate the dense motion inside the occlusion region using two 2D polynomial models. High order polynomial models have been widely used in describing omnidirectional image distortion [13] and are able to capture the appearance change of objects caused by camera motion and environment structure [10].

First, motion in the occluded regions needs to be removed using the mask, which is given or detected in advance. However, measured motion vectors of regions near occlusions is also impaired due to the smoothness constraints in optical flow algorithms. Therefore, here we need to expand the mask region by image dilution processing in order to cover the entire influenced region. Next, we approximate the motion of pixels in each image frames using two 2D polynomial models, one for $\mathbf{V}_\theta$ and one for $\mathbf{V}_\phi$. Motion fields in the direction θ and ϕ are modeled separately based on their spherical coordinates, as is expressed in Eqs. (3) and (4).

$$\mathbf{V}_\theta(\theta,\phi) = \sum_{i+j=0,1,\ldots,m} a_{ij}\theta^i\phi^j \tag{3}$$

$$\mathbf{V}_\phi(\theta,\phi) = \sum_{i+j=0,1,\ldots,m} b_{ij}\theta^i\phi^j \tag{4}$$

where a_{ij} and b_{ij} are the polynomial model coefficients for each image frame and m is the overall order of polynomial model. All possible terms up to order m are considered in this form and give rise to p coefficients here. The polynomial model coefficients can be optimized by solving least-square minimizations. Here, we find the polynomial coefficients a_{ij} and b_{ij} by minimizing the error Eqs. (5) and (6) separately.

$$e_\theta = \sum_{k=1,2,\ldots,N} |\mathbf{v}_\theta(\theta_k,\phi_k) - \bar{\mathbf{v}}_\theta(\theta_k,\phi_k)|^2 \tag{5}$$

$$e_\phi = \sum_{k=1,2,\ldots,N} |\mathbf{v}_\phi(\theta_k,\phi_k) - \bar{\mathbf{v}}_\phi(\theta_k,\phi_k)|^2 \tag{6}$$

In these two equations, θ_k and ϕ_k represent the spherical coordinates of unimpaired pixel k and there are N unimpaired pixels on the image frame. $\mathbf{v}_\theta$ and $\mathbf{v}_\phi$ are the estimated motion vectors of unimpaired pixels using the polynomial models and $\bar{\mathbf{v}}_\theta$, $\bar{\mathbf{v}}_\phi$ are the measured motion vectors of unimpaired pixels using the optical flow algorithm. To solve for polynomial coefficients, the Eq. (5) can be further rewritten in the form of least square problems

$$\mathbf{c}^* = \arg\min_{\mathbf{c}} ||\mathbf{A}\mathbf{c} - \bar{\mathbf{v}}||_2, \tag{7}$$

where

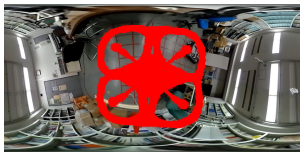
$$\mathbf{A} = \begin{bmatrix} 1 & \theta_1 & \phi_1 & \theta_1\phi_1 & \theta_1^2 & \phi_1^2 & \ldots & \theta_1^m & \phi_1^m \\ 1 & \theta_2 & \phi_2 & \theta_2\phi_2 & \theta_2^2 & \phi_2^2 & \ldots & \theta_2^m & \phi_2^m \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & \theta_N & \phi_N & \theta_N\phi_N & \theta_N^2 & \phi_N^2 & \ldots & \theta_N^m & \phi_N^m \end{bmatrix},$$

$$\mathbf{c} = \begin{bmatrix} a_{00} & a_{10} & a_{01} & a_{11} & a_{20} & a_{02} & \cdots & a_{m0} & a_{0m} \end{bmatrix}^T,$$

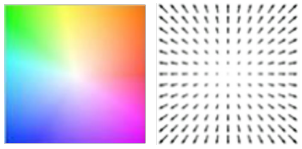
$$\bar{\mathbf{v}} = \begin{bmatrix} \mathbf{V}_\theta(\theta_1,\phi_1) & \mathbf{V}_\theta(\theta_2,\phi_2) & \cdots & \mathbf{V}_\theta(\theta_N,\phi_N) \end{bmatrix}^T.$$

Matrix $\mathbf{A}$ is the 2D polynomial expression in spherical coordinates, $\mathbf{c}$ is coefficient vector and $\bar{\mathbf{v}}$ is the measured optical flow in the θ direction. Here we use QR decomposition to solve Eq. (7) for its fast computation and higher stability as compared to calculating the pseudo-inverse. Briefly speaking, after decomposing $\mathbf{A}$ into an orthogonal matrix $\mathbf{Q}$ and an upper triangular matrix $\mathbf{R}$, the coefficient matrix $\mathbf{E}$, which minimizes Eq. (5), can be obtained as $\mathbf{R^{-1}Q^TV}$. The coefficients b_{ij} can also be solved in the same way. After obtaining the polynomial coefficients a_{ij} and b_{ij}, we can estimate the spherical optical flow in the occluded regions using Eqs. (3) and (4) according to their spherical coordinates.

In the case of high resolution spherical image sequences, the size of image is very big, which causes pixel number N to be much bigger than coefficient number p. This causes the matrix $\mathbf{A}$ as well as the decomposed matrix $\mathbf{Q}$ and $\mathbf{R}$ to be very sparse and cost high computation time. Therefore, before we solve the least square problem, we first downsample the image matrix and the corresponding optical flow matrix. Since the optical flow matrix is very dense and smooth in most regions, the downsampling process will not affect the accuracy of solution but could hasten the solution. In addition, here we choose an unweighted least square minimization on each single image frame to interpolate the



(a) Input image frame with artificial drone-shape occlusion. Red region is cut manually.



(b) Visualization of flow field.

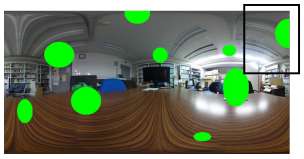
(c) Ground truth motion estimated by SIFT flow [11].

(d) Impaired motion on this image frame due to occlusions.

(e) Interpolated motion by the proposed method without downsampling.

(f) Interpolated motion by the proposed method with downsampling.

Fig. 6. Motion interpolation on an example frame. As can be seen, it is not affected by downsampling.



(a) Input image. Green regions mark occlusions and one appears at the right border.

(b) Direct interpolation causes extrapolation near upper right corner, producing incorrect motion.

(c) Our interpolation result after performing image extension.

(d) Ground truth of optical flow on this frame.

Fig. 7. Extrapolation problem for occlusions near borders

motion to ensure the spatial coherence within two frames. Because the assumption of linear approximation for motion trajectory does not hold in spherical images, we do not choose a temporally weighted least square minimization [10], which calculates average speed in multiple frames to enforce temporal coherence. Instead, the temporal coherence is enforced by image warping later when doing video completion.

Figure 6 gives an example of doing motion interpolation on an input image frame, Fig. 6(a). In this paper we follow the paper [14] to visualize the motion flow field in color, in which the hue and saturation are transformed from the orientation and magnitude of motion, respectively, as is shown in Fig. 6(b). The ground truth motion on this input image frame without red occlusions can be estimated by SIFT flow [11] and is shown in Fig. 6(c). If we manually cut a drone-shape occlusion, which is shown in red in Fig. 6(a), the motion then is impaired and unknown in occlusion regions, as can be seen in Fig. 6(d). In Fig. 6(e) and Fig. 6(f), the motions in occluded regions are both interpolated successfully, with or without performing downsampling to fasten the interpolation, and show no differences. Hence, we can increase the speed without sacrificing performance.

B. Solution to extrapolation problem near image borders

As is mentioned before, normal images suffer from estimating motion near borders due to lack of information outside of image. This is caused by the limited field of view in normal cameras. It will also cause the failure of accurately interpolating motion near image borders. To illustrate this issue, we use Fig. 7 as an example, where the image frame was under almost a purely yaw rotation. The ground truth motion between Fig. 7(a) and its next frame is shown in Fig. 7(d). However, if we cut some regions (marked as green) in Fig. 7(a), and directly interpolate the motion inside these regions using the algorithm above, motion interpolation near borders, such as the black rectangular region, fails as is shown in Fig. 7(b). The reason for failure is that the lack of information outside of the border causes interpolation to extrapolation and leads to false motion interpolated.

To deal with this information, we again utilize the continuity of spherical images. Different from normal images, spherical images are continuous and the information actually exists on the opposite border. Before doing polynomial interpolation, we first extend the equirectangular image to ensure there is enough information outside of border, as is done in Fig. 5. Besides, for the top and bottom border of equirectangular images they are projected from two single pixels, top and bottom one, in the spherical image. Therefore, optical flow field can be set as approximately zero value at these places. After image extension, we can do motion interpolation on the extended image and the result can be seen in Fig. 7(c). The result shows the extrapolation problem is solved properly and the motion can be interpolated well near borders.

V. Video Completion Using Image Warping

Based on the motion that we estimate by polynomial interpolation, we complete the occlusion region by image warping. Image warping is an image transformation process using a vector field $\mathbf{V}$: $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ that translates each points in one image to another one. In this case, the vector field consists of the interpolated spherical optical flow vectors and points are spherical coordinates of pixels on image frames. For a particular pixel (θ_s, ϕ_s) in the source image, the corresponding coordinates (θ_w, ϕ_w) of the warped pixel on the next frame can be calculated as

$$(\theta_s, \phi_s) \leftarrow (\theta_s + \mathbf{V}_\theta(\theta_s, \phi_s), \phi_s + \mathbf{V}_\phi(\theta_s, \phi_s)). \quad (8)$$

To find corresponding pixels in the further frames, we use the chain of optical flow by iteratively warping the source image

Fig. 8. Experimental setting to record ground truth image frames. A spherical camera, Ricoh Theta, is hung using a thin string to take ground truth image sequences.

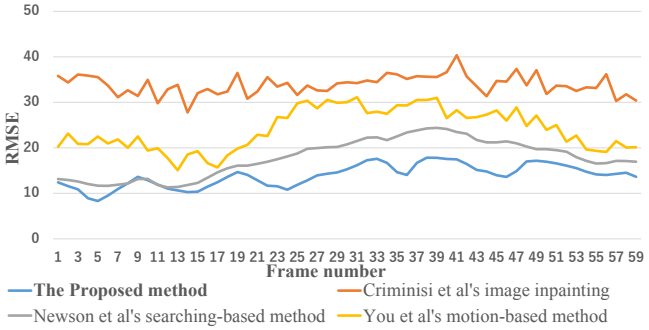


Fig. 9. Comparison of RMSE on all video frames. The proposed method shows the best accuracy in almost all image frames, compared to other three methods.

further, instead of directly multiplying the frame difference, since the linear approximation assumption does not hold in panoramic image sequences. For any given image frame i, it can be warped to another image frame j, frame by frame from $i, i+1, ...,$ to j, according to the motion of each two consecutive frames between frame i and frame j. Based on the motion, we can find the correspondences in other image frames. When pixels move out of the left or right borders, we find their warped position near the opposite border based on the continuity of the spherical image.

To utilize the information in temporally far frames, we iteratively alternate the warping sequence in the forward and reverse direction to scatter information instead of one single direction. In addition, to trace the occluded region and prevent holes from appearing in the warped images due to interpolation in pixel coordinates, we warp the source image and mask together to the target frame. Useful information for completing the target image is that which appears in the target mask yet outside of source mask. Since the motion between the target image and the closer source image is more accurate and more coherent, we choose the corresponding pixels in the temporally closest frame's unoccluded regions and use those pixels to complete the occlusions in the source image by copying. For large occlusions, in which the correspondences cannot be found, we choose to believe in the completed regions and propagate the inpainting, which are then used to complete the remaining occlusions.

VI. Experiments

We conducted quantitative experiments to measure the accuracy and efficiency of our proposed video completion algorithm. The motivation of our approach is to complete the occlusions using real background regions in limited time. Therefore, it is important to check the execution time and whether our approach is able to recover the occluded information, instead of just filling in some plausible information. We use the root mean squared error (RMSE) metric to calculate the error on the 8 bit (R; G; B) values between the ground truth and the completed video frames. The RMSE value is calculated on each frame and the average RMSE value of the whole image sequences is also calculated.

To have ground truth image frames to compare with, we need to take image frames, in which there are no occlusions existing. As is shown in Fig. 8, we used a thin string to hang a spherical camera, Ricoh theta, to capture image sequences. This captured videos with almost zero occlusions and thus were used as the ground truth. To simulate motion of a drone with a spherical camera, we put a drone-shaped mask on the video and manually delete the mask region, which made the video severely occluded. We also took real videos captured by a Ricoh Theta on the a moving AR Parrot Drone 2.0 to test the applicability of our algorithm. The input data is in 960 $\times$ 480 resolution. Our method runs on a laptop with an 8GB memory with an Intel i7 CPU. On the same hardware and conditions, we compared our method with Criminisi *et al.*'s image inpainting method [6], Newson *et al.*'s search-based video completion method [8], and You *et al.*'s motion-based method [10]. Other methods [3], [4], [9] require a multiple-camera system, a pre-reconstructed environment model or a pan-tilt camera and thus were not considered for comparison. All programs were run in a MATLAB environment.

As shown in Table I, all four methods take relatively long time to process each frame due to the large occlusions, but our proposed method is significantly faster than all other methods since we do not involve any similarity searching methods. Besides, the proposed method also shows better results in recovering true background information. The comparison of four methods on each frame can be seen in Fig. 9. The results show that our method outperforms others on almost all frames. Fig. 10 shows a comparison on the 50th frame. To help see the difference better, we rotated the occlusion to the center of image. As the outputs and the RMSE values show, our method can efficiently recover background information by tracing the optical flow trajectory to find the corresponding information, even across the image borders. In comparison, the search-based method [8] gives a smooth output, in which most occlusions are replaced by textureless regions. These textureless regions were chosen as the most similar patches with the surrounding information. This makes the outputs look visually nice, but does not fit our purpose of recovering true background information. Image inpainting method [6], as well as the previous motion-based method [10], which calls method [6] when correspondences in neighboring frames cannot be found, fails to recover the true information since they only utilize the information on the same frame itself.

TABLE I

AVERAGE RGB RMSE AND COMPUTATION TIME COMPARISON PER FRAME

	Proposed	**Image inpainting [6]**	**Searching-based [8]**	**Motion-based [10]**
Average RMSE	**13.8719**	33.8339	17.7644	24.0618
Time (Seconds)	**126**	890	261	172

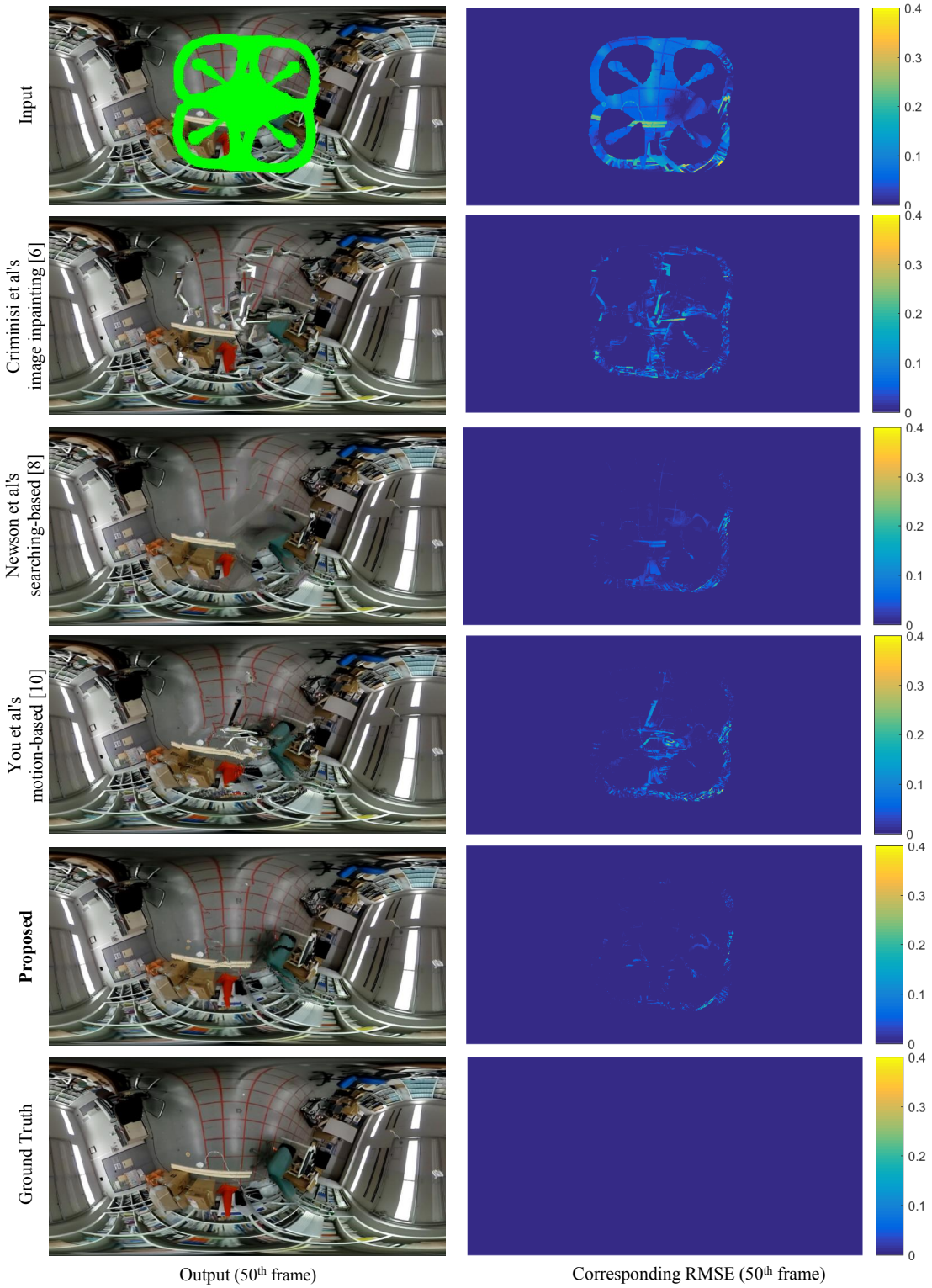


Fig. 10. Quantitative evaluation of video completion

VII. CONCLUSIONS

In this paper, we proposed a novel method to remove undesired occlusions using estimated motion in panoramic video. We newly combine polynomial fitting with spherical continuity to interpolate motion and complete occlusions along the estimated optical flow trajectories. No constraints of camera motion are included in our approach. Experimental results demonstrate the accuracy and efficiency of our proposed approach. In future work, more accurate optical flow estimation method will be considered. Besides, our current motion interpolation method gives a smooth output, in which motion of edges is not considered yet. Our future work is also to include an edge-aware interpolation into our completion method.

ACKNOWLEDGMENT

This work was in part supported by the 'Cross-ministerial Strategic Innovation Promotion Program (SIP), Infrastructure Maintenance, Renovation, and Management', Council for Science Technology and Innovation (funding agency: NEDO).

REFERENCES

[1] A. Pagani and D. Stricker, "Structure from motion using full spherical panoramic cameras," in *Proceedings of the 2011 IEEE International Conference on Computer Vision Workshop*, 2011, pp. 375–382.

[2] S. Kasahara, S. Nagai, and J. Rekimoto, "Livesphere: immersive experience sharing with 360 degrees head-mounted cameras," in *Proceedings of the Adjunct Publication of the 27th Annual ACM Symposium on User Interface Software and Technology*, 2014, pp. 61–62.

[3] N. Kawai, K. Machikita, T. Sato, and N. Yokoya, "Video completion for generating omnidirectional video without invisible areas," *IPSJ Transactions on Computer Vision and Applications*, vol. 2, pp. 200–213, 2010.

[4] N. Kawai, N. Inoue, T. Sato, F. Okura, Y. Nakashima, and N. Yokoya, "Background estimation for a single omnidirectional image sequence captured with a moving camera," *Information and Media Technologies*, vol. 9, no. 3, pp. 361–365, 2014.

[5] D. Paredes, P. Rodriguez, and N. Ragot, "Catadioptric omnidirectional image inpainting via a multi-scale approach and image unwrapping," in *Proceedings of the 2013 IEEE International Symposium on Robotic and Sensors Environments*, 2013, pp. 67–72.

[6] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1200–1212, 2004.

[7] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 463–476, 2007.

[8] A. Newson, A. Almansa, M. Fradet, Y. Gousseau, and P. Pérez, "Video inpainting of complex scenes," *SIAM Journal of Imaging Science*, vol. 7, no. 4, pp. 1993–2019, 2014.

[9] A. Yamashita, I. Fukuchi, T. Kaneko, and K. T. Miura, "Removal of adherent noises from image sequences by spatio-temporal image processing," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 2386–2391.

[10] S. You, R. T. Tan, R. Kawakami, and K. Ikeuchi, "Robust and fast motion estimation for video completion." in *Proceedings of the 2013 IAPR International Conference on Machine Vision Applications*, 2013, pp. 181–184.

[11] C. Liu, J. Yuen, and A. Torralba, "Sift flow: Dense correspondence across scenes and its applications," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 5, pp. 978–994, 2011.

[12] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the 1999 IEEE International Conference on Computer vision*, vol. 2, 1999, pp. 1150–1157.

[13] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A toolbox for easily calibrating omnidirectional cameras," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 5695–5701.

[14] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, 2011.