Vol.2015-CVIM-195 No.54
2015/1/23

View interpolation for omnidirectional-based immersive virtual systems

Abdoulaye Maiga[1,2,a)] Naoki Chiba[2,b)] Tony Tung[2,c)] Hideo Saito[1,d)]

Abstract: We present a novel method for interpolating views captured with an omnidirectional camera. The proposed method relies on a piece-wise transition-based image interpolation model that produces smooth local transitions between images by selecting optimal transition points. Hence, ghosting effects in standard interpolation methods can be reduced, and occlusion issues can be handled. The technique can be used to reduce the number of capturing images, which is the burden when constructing an image-based virtual system, while preserving the immersive realism. We show early results using real-world images as a proof of concept.

1. Introduction

To date, to realize a virtual reality (VR) system for visiting real world scenes, there are mainly three popular techniques: 1) computer-based generated world, 2) 3D reconstruction from images, and 3) Image-Based Rendering (IBR) techniques. First, fully computer-based generated virtual worlds can nowadays look very realistic and display great level of details, in particular thanks to the progress of computer graphics technologies (GPU). However, the main issue is still the time and expertise level required to be able to generate a compelling real world scene using computer graphics software (e.g., week to months for complex scenes using MAYA). Second, image-based 3D reconstruction techniques have continuously shown significant evolution since the past decade as in [3] and [2], which are able to generate full 3D models of real-world (indoor) environments automatically. However, the accuracy of the results is still lacking in some respect to be fully incorporated into a virtual environment where the users can freely inspect detailed elements. Finally, on the other hand, IBR techniques are very suitable for generating virtual worlds, simply due to the fact that resolution is higher when using High Definition images. Nevertheless, it is still necessary to capture a considerable amount of images or video frames to obtain an accurate world, such as in [8].

We present a method to interpolate views between two images representing the same scene observed from different positions or angles. Hence, the actual number of images to be taken can be greatly reduced. As observed different parts of the image should be interpolated differently (e.g., foreground objects versus background), and therefore transition points should be spatially anisotropic and occur independently with respect to image regions. Hence, we propose a piece-wise transition-based image interpolation model in order to reduce ghosting effect and handle occlusion issues. First, visual structure of images are characterized by planar meshes built from visual features. Then, each mesh face content is derived from the two observed images while transition point is estimated geometrically, and optimized using an energy minimization framework. Moreover, by segmenting the images into regions we are able to handle various types of camera motions within the image (i.e., rotation and translation) and therefore we can apply our method for a full 360 degrees interpolation of omnidirectional images.

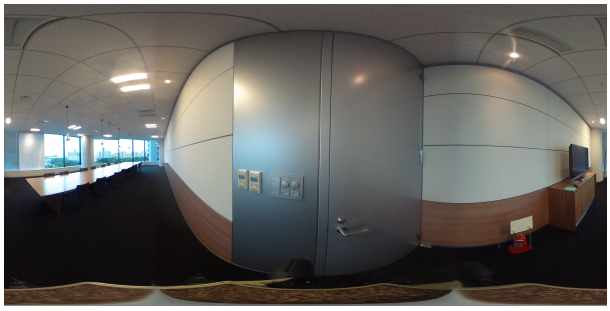


Fig. 1: Example of input omnidirectional image (360 degree view) in equirectangular format.

In comparison to a pixel-based approach (see details in the next section), our method defines transitions per region (where pixels will behave almost identically), hence reducing the computation to the number of regions instead of the number pixels, and better preserving image content structure (such as contours of non-natural objects). One of the possible application, as we will explain in further sections, is to obtain high quality interpo-

[1] Keiko University, Yokohama, Kanagawa 223–8522, Japan
[2] Rakuten, Inc., Shinagawa, Tokyo 140–0002, Japan
[a)] abdoulaye.m@hvrl.ics.keio.ac.jp
[b)] naoki.a.chiba@mail.rakuten.com
[c)] tony.tung@mail.rakuten.com
[d)] saito@hvrl.ics.keio.ac.jp

lation which can be directly used in a virtual reality system. In this work, experiments were made on real-world omnidirectional images (see example of input image in Fig. 1).

2. Related Work

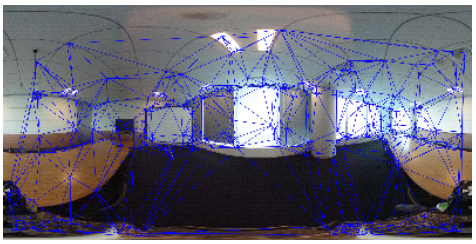
There is much research that has been done on both 3D reconstruction and IBR within the past decades [1][2][3][8]. However the presented work will focus mainly on the in-between image interpolation using accurate matches.

Optical flow methods, although not traditionally used for interpolation, are used to calculate various motions between input images. Using the forward flow (from image A to image B) and backward flow, one can linearly blend images to create interpolation. The main issue is that such blending due to flow errors can be inaccurate and superpose pixels. Moreover occlusion handling is non trivial for such techniques. Another technique is to use video processing to achieve plausible interpolation. The idea of using transition points has previously been used to solve issues related to video texture . Transition points were defined as points where the video could be looped with very minimal obstruction to the global structure of the video. In that sense, it differs somewhat from our current definition.

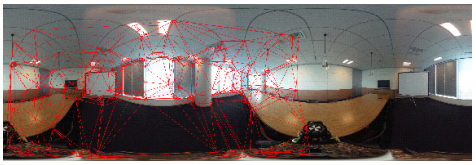
In our attempt to create a fully interactive VR system where the user can navigate through the pictures using a head-mounted display (HMD), omnidirectional images taken apart are used (e.g., at $0.8m$). Therefore, depending on the user position the system chooses which image to display. By doing so, we are able to simulate a walking experience through the images. Using an HMD and the images as environment maps, users can have a sense of virtual exploration. However, in order to enhance the experience, images would need to be taken at a very high frequency. A similar approach is presented in [13], although not using an HMD, a walkthrough system for virtual tourism is presented. Using 3 images and supposing that the 3D point matches are given, the notion of blending weight is introduced. The weight is determined depending on the position of the viewpoint.

In [9] an interactive system for browsing photos is presented. Using a sparse 3D model, a smooth transition is established between different images. Similarly, once accurate keypoints have been matched the authors compute interpolated images between 2 input images. However, 3D information using structure from motion (SFM) is used to place the images. Because of the application (browsing hundreds of photos from the Internet), extremely precision is not required and therefore some artifacts remain visible without any noticeable discomfort to the users. Our aim is to create seemingly no artifacts in order for the system to be used for a VR system where small misalignments may prove to be very bothersome to the end users.

Finally in [11] a pixel path based method is presented. This technique consists in interpolating between two images by computing a plausible path for each pixel. This path is obtained using the gradient domain. Once a plausible path is computed a Poisson reconstruction from the gradients is used to obtain the interpolated frames. The notion of arbitrary transition point is defined and used in a similar fashion. The difference between their approach and ours is mainly the need to minimize for each pixel, and the explicit process to handle occlusions. Therefore entire regions where pixels share the same path are not clearly defined as in our case. By doing so, we can reduce drastically the number of transition points that actually need to be calculated. This makes our technique adaptable to images with big disparities where optical flow would require closer images.



(a) Image A



(b) Image B

Fig. 2: Example of images with triangulated mesh built from matching keypoints (here, SIFT features).

3. Algorithm overview

This section presents our method in more details. The algorithm consists in 3 essential steps: 1) identifying suitable visual keypoints across both images (detection and matching) and generating Delaunay triangle mesh, 2) computing affine transformation between mesh triangles, 3) and finally determining the appropriate transition point for each triangle.

The first step is to correctly identify and match keypoints across images. Keypoint matching is the basis to many algorithms in computer vision and yet remains a fatal part of the process and we need a robust method in order to maintain only accurate keypoints. To avoid overpartition of complex scenes, we adopt a coarse-to-fine strategy. We begin by creating 2 reduced resolution images of the input images (i.e., to half and one forth the original sizes). By extracting keypoints from scaled images we can still keep accurate keypoints by recalculating original positions in the full size image. For each image, SIFT [12] keypoints and contours (i.e., line segments) are detected and extracted. We use SIFT as it is invariant to image transformations and provides a feature vector that can be used for many purposes (but actually

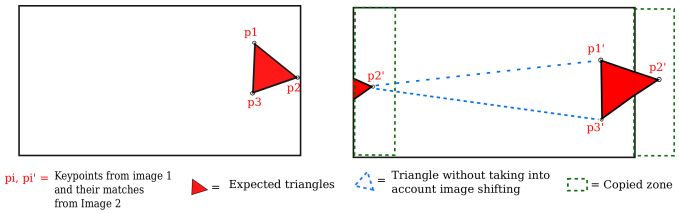


Fig. 3: Equirectangular images A (left) and B (right). Keypoints (p_1, p_2, p_3) are respectively matched to (p'_1, p'_2, p'_3). Due to camera motions, point p'_2 has shifted to the left side. Forming a triangle using this would generate many artifacts. Hence, the image is extended in order to obtain consistent triangulation.

any robust feature detector would also be acceptable). We then perform a nearest neighbour matching in the feature space, and apply a symmetry test (for robustness) by matching points from image A to image B then from image B back to image A. In order to select regions which are as homogeneous as possible, we define a threshold and keep only keypoints that are within that distance to the closest line segment. This ensures that we only keep points close to contours. By doing so, we are able to eliminate significant amount of triangles that lie on inhomogeneous regions (e.g., texture representing a part of table and wall), and which can potentially be a problem for smooth interpolation.

Once keypoints are identified and matched, we then compute Delaunay triangulation using the keypoints of the first image as mesh vertices. The mesh is then replicated to the second image using the exact matches to form matched triangles (see Fig. 2). As we are using omnidirectional images in equirectangular format we have to handle the particular case where corresponding triangles have points beyond the boundaries of the image (as shown in Fig. 3). To solve cases like this, we simply extend the second image by copying a part of the same image as an extension to the same image. We then constraint triangle matching between images so that triangle edge lengths do not exceed $1/4^{th}$ of the image size (length dimension).

The second step consists in determining a mapping between triangle mesh content (i.e., texture). We then compute the affine transformation F_i between each triangle pair $\{(\Delta_i, \Delta'_i)\}$ such that:

$$\forall P_1(x_1, y_1) \in \Delta_i, \quad P_2(x_2, y_2) = F_i P_1, \tag{1}$$

where i is the triangle index in both images, P_1 is a point located in triangle Δ_i in image A, P_2 is a corresponding point to P_1 located in image B, (x_k, y_k) are pixel coordinates of point P_k in the images, (Δ_i, Δ'_i) is a pair of matching triangles from images A and B respectively, and F_i is the affine transformation between Δ_i and Δ'_i.

Finally, for each triangle i we compute transition values $\{C_i\}$ for triangle content interpolation. In the current implementation

(a) Input Image A

(b) Input Image B

(c) Interpolated image using standard blending

Fig. 4: Interpolation example without taking into account the individual triangle transition points, noticeable ghosting effect created

we use differences of triangle edge lengths $\{l_i^j\}_{j=1,2,3}$ and triangle area A_i between corresponding triangle pair and interpolated triangle. Interpolation between images is then performed linearly:

$$\Delta_i^{inter} = \frac{(X_2 - X_k) * \Delta_i + (X_k - X_1) * \Delta'_i}{X_2 - X_1}, \tag{2}$$

where Δ_i^{inter} is the triangle interpolated from triangles Δ_i and Δ'_i from images A and B respectively, X_1 and X_2 represent positions in arbitrary image space of images A and B respectively, and $X_k \in [X_1, X_2]$ is the interpolation factor corresponding to the transition point. Finally, depending on which triangle (from image A or B) the interpolated triangle is closer to, according to the criteria introduced above, pixel values from either image A or image B are mapped into the interpolated triangle. To obtain the set of interpolated triangles, we compute all minimal transition values $\{C_i = \arg\min_{\{C_i^k\}} C_i^k\}$ where C_i^k is the transition value of the interpolated triangle i at position X_k.

4. Preliminary results

Our main purpose is to create a sense of full immersion and allowing users to choose their path based on a set of predesigned paths. This allows the users to have a sense of motion while cycling through images without requiring a 3D model. However, if images are too far a part, the sense of immersion reduces because of the lack of continuity. Hence, the interpolated images need to blend naturally into the set of pre-registered images for a more realistic walkthrough experience.

We use a Ricoh Theta camera to capture input omnidirectional images. Images are captured by the camera placed on a off-the-shelf tripod at regular distance intervals such as human step (0.8m). The camera provides 3584 x 1792 pixel images representing a 360 degree omnidirectional view of a real-world scene

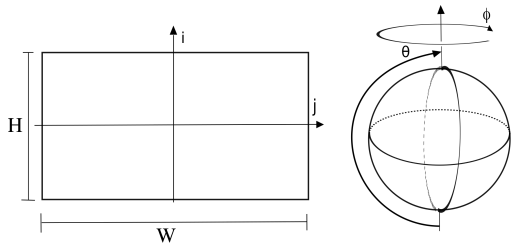
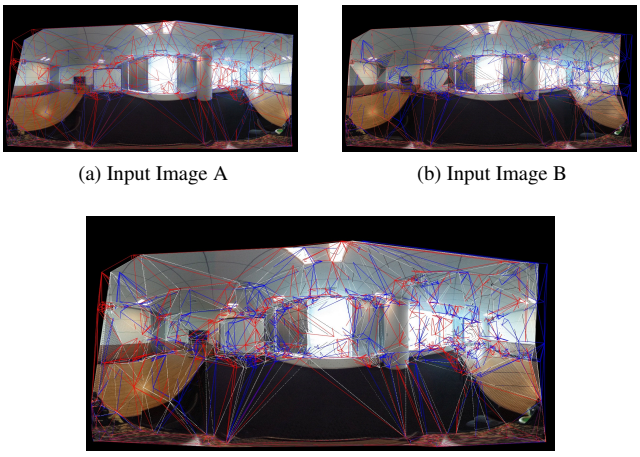


Fig. 5: Equirectangular and spherical notations of omnidirectional image.



(a) Input Image A (b) Input Image B

(c) Interpolated image using our method

Fig. 6: Interpolation example taking into account individual triangles: The blue, white and Red meshes representing triangles from Input A, Interpolated, and Input B. The White mesh morphs from the Blue towards the red

in equirectangular format. Each input image has a height H and a width $W = 2H$ since the coverage is fixed to 360 degrees horizontally and 180 degrees vertically. Spherical coordinates (θ, ϕ) can be obtained from image coordinates (i, j) by forward projection on sphere surface (see Fig. 5) for image warping.

We present preliminary qualitative results. The first set of images shown in Figure 4 display the ghosting effect that is obtained without taking into consideration transition points. Essentially we compute the affine transformation between the images, then we apply pixel blending between both images. As we can see in Figure 4, like many optical flow techniques this produces a relative ghosting effect especially for objects in close range which have large induced motion across the images. By blending triangles of different sizes into the newly formed interpolated triangle, there is a certain lost of consistency in terms of pixel values. The images therefore appear to be superposed. By doing so, there is no benefit to segmenting the image correctly.

In Figure 6 the result of our method applied to the input images is shown. In the given images, we superpose the triangle meshes for images A, B and the interpolated triangles. The interpolated mesh morphs from input image A to image B as its position gets closer to the second image.

5. Conclusion and Future Work

In this paper we presented a novel method for smooth image interpolation using anisotropic transition points. Unlike the pixel based transition, we define regions and estimate individual transitions for each. The pixels are then chosen to minimize a distance between the interpolated triangles and the original triangles from two input images.

However, by only considering individual triangles for our minimization, we are not taking into account the surroundings and global smoothness. In order to solve this, we must define a minimization function which takes into account adjacent triangles. This will allow us to handle interpolation of triangles and surroundings jointly.

References

[1] Furukawa, Yasutaka and Curless, Brian and Seitz, Steven M and Szeliski, Richard:, *Reconstructing Building Interiors from Images*, IEEE ICCV 2009.
[2] Zhang, Yinda and Song, Shuran and Tan, Ping and Xiao, Jianxiong:, *PanoContext: A Whole-room 3D Context Model for Panoramic Scene Understanding*, ECCV 2014.
[3] Cabral, Ricardo and Furukawa, Yasutaka, *Piecewise Planar and Compact Floorplan Reconstruction from Images*, IEEE CVPR 2014.
[4] A. Lippman:, *Movie-maps: An application of the optical videodisc to computer graphics*, ACM SIGGRAPH 1980.
[5] S. E. Chen, *Quicktime VR: An image-based approach to virtual environment navigation*, ACM SIGGRAPH 1995.
[6] Seitz, Steven M, Dyer Charles R., *View Morphing*, ACM SIGGRAPH 1996
[7] Saito, Hideo and Baba, Shigeyuki and Kanade, Takeo *Appearance-based virtual view generation from multicamera videos captured in the 3-D room*, Multimedia, IEEE Transactions 2003
[8] Uyttendaele, Matthew and Criminisi, Antonio and Kang, Sing Bing and Winder, Simon and Szeliski, Richard and Hartley, Richard, *High-quality Image-based Interactive Exploration of Real-World Environments*, Technical report MSR-TR-2003-61, Microsoft Research 2003.
[9] Snavely, Noah and Seitz, Steven M and Szeliski, Richard, *Photo tourism: Exploring photo collections in 3D*, ACM SIGGRAPH 2006.
[10] Kobayashi, Kenkichi and Saito, Hideo *Arbitrary viewpoint image synthesis based on light field construction from uncalibrated image sequence*, Electronics and Communications in Japan 2004
[11] Mahajan, Dhruv and Huang, Fu-Chung and Matusik, Wojciech and Ramamoorthi, Ravi and Belhumeur, Peter, *Moving gradients: a path-based method for plausible image interpolation*, ACM SIGGRAPH 2009.
[12] Lowe, David G *Distinctive image features from scale-invariant keypoints*, ICV 2004.
[13] Tomite, Kaname and Yamazawa, Kazumasa and Yokoya, Naokazu, *Arbitrary viewpoint rendering from multiple omnidirectional images for interactive walkthroughs*, Pattern Recognition, 2002.