2011 12th International Conference on Computer-Aided Design and Computer Graphics

A 2D-3D Hybrid Approach to Video Stabilization

ZhiYong Huang
Computer School
Wuhan University
Wuhan, China
cadcg@whu.edu.cn

FaZhi He
Computer School
Wuhan University
Wuhan, China
fzhe@whu.edu.cn

Xiantao Cai
Computer School
Wuhan University
Wuhan, China
caixiantao@whu.edu.cn

Yuan Chen
Computer School
Wuhan University
Wuhan, China
cynthia@whu.edu.cn

Xiao Chen
Computer School
Wuhan University
Wuhan, China
84299889@qq.com

***Abstract*—In this paper, we introduce a novel 2D-3D hybrid video stabilization method which combines virtues of 2D and 3D video stabilization methods in one routine. It attempts to achieve high-quality camera motions and to retain full frame coherence in each frame, at while, ensure that local regions undergo a similarity transformation. We solve the stabilization problem by integrating 3D and 2D video stabilization methods into one routine. It smooths camera motions and explicitly employs local motion information which constraints video frames to be temporal coherent, and achieves high-quality video stabilization. Experiments show that our method not only can achieve high-quality camera motion on good 3D reconstructed scene, but also can deal with complicated videos containing near, large moving objects.**

***Keywords*-video stabilization; video retargeting;**

I. Introduction

The most obvious difference between professional and amateur level video is the quality of camera motion; hand-held amateur video is typically shaky and undirected, while professionals use careful planning and equipment such as dollies or steadicams to achieve ideal motion [1]. These hardware are very expensive and impractical for personal requirements, so video stabilization software is a widely-used and important tool for improving casual video. In this paper, we introduce a technique for software video stabilization that is robust, yet provides high quality results over various kinds videos.

Prior techniques for software video stabilization can be divided into two categories, 2D video stabilization and 3D video stabilization. The most common approach is 2D stabilization [2], which is widely implemented in industry products. This approach applies 2D motion models, such as affine or projective transforms, to each video frame. Though 2D stabilization is robust and fast, the amount of stabilization it can provide is very limited because the motion model is too weak; it cannot account for the parallax induced by 3D camera motion. In contrast, 3D video stabilization techniques [3], [4], [5], [6] can perform much stronger stabilization, and even simulate 3D motions such as linear camera trajectories. In this approach, a 3D model of the scene and camera motion are reconstructed using structure-from-motion (SFM) techniques [7], and then new views are rendered from smooth 3D camera path. The problem with 3D stabilization is the opposite of 2D: the motion model is too complex to compute quickly and robustly. 2D approaches enforce plausibility by limiting changes to 2D transformations, which is simple but too limiting. 3D approaches reconstruct a 3D model of the scene and use it to enforce the validity of synthesized views. However, performing 3D reconstruction is error-prone and overkill for the stabilization problem.

In this paper, we propose a novel video stabilization technique that combines the advantages of 2D and 3D video stabilization. That is, our method achieves the strongly stabilized, high-quality appearance of 3D stabilization and the robustness of 2D methods. Our video stabilization method can be summarized by three steps. First, it recovers the 3D camera motion and a sparse 3D point cloud, static scene points using a structure-from-motion(SFM) system tool. Second, this method compute local motion for each frame. Finally, it performs a least-squares optimization that computes a spatially-varying warp for each input video frame into an output frame. The first and second step is well studied in computer vision. The third step is the key challenge of 3D video stabilization: it must synthesize new, smooth camera motion that respects geometric relationships between points, so that they appear as the motion of a plausible, non-distorted view of the scene. Our method can achieve the high quality stabilization results seen in 3D stabilization. In videos where SFM performs well, our results are comparable to Liu *et al*. [4]. In where SFM fails, our methods will degenerate to similar with full-frame video stabilization method, and be much more robust, can produce more reasonable result with fewer distortion in frames.

II. Related Work

2D video stabilization techniques have reached such a level of maturity that they are commonly implemented in on-camera hardware and run in real time [2]. These approaches [2], [8]can be sufficient if the user only wishes to damp undesired camera shake, if the input camera motion consists mostly of rotation with very little translation, or if the scene is planar or very distant. However, in the common case of a camera moving through a three-dimensional scene,

978-0-7695-4497-7/11 $26.00 © 2011 IEEE
DOI 10.1109/CAD/Graphics.2011.39

there is typically a large gap between 2D video stabilization and professional-quality camera paths.

The idea of transforming hand-held videos to appear as if they were taken as a proper tracking shot was first realized by Gleicher and Liu [9]. Their approach segments videos and applies idealized camera movements to each. However, it is based on full-frame 2D warping, and therefore suffers (like 2D approaches) from two fundamental limitations: it cannot reason about the movement of the physical camera in 3D, and it is limited in the amount of viewpoint change for scenes with non-trivial depth complexity.

The 3D approach to video stabilization was first described by Buehler *et al*. [3]. In 3D video stabilization, the 3D camera motion is tracked using structure-from-motion [10], and a desired 3D camera path is fit to the hand-held input path. With this setup, video stabilization can be reduced to the classic image-based rendering problem of novel view interpolation: given a collection of input video frames, synthesize the images that would have been seen from viewpoints along the desired camera path. Though the novel viewpoint interpolation problem is challenging and ill-posed, recent sophisticated techniques have demonstrated high-quality video stabilization results [11], [12]. However, the limitation to static scenes renders these approaches impractical, since most of us shoot video of dynamic content.

Our technique can avoid the distortion which was introduced by inaccurate 3D recovery. But accurate 3D recovery is still important to our method, more accurate 3D recovery can produce more stable result video. To achieve high quality camera motion, our approach requires computation of video structure-from-motion. However, this step has become commonplace in the visual effects industry, and commercial 3D camera trackers are widely used. We use the free and publicly available ACTS[1] camera tracker and Voodo[2] camera tracker, which has been used in a number of recent research systems.

III. Our Approach

Our approach is based on Liu's method [4] and introduce relative motion information to constrain results to be temporal coherence. Firstly, it recover the original 3D camera motion and sparse 3D point cloud using structure-from-motion and then smooth camera motion, which similar with 3D stabilization techniques. Secondly, it compute relative motion from pairwise registration and optical flow. Lastly, rather than synthesize novel views using multiple input video frames, at the constraints of 2D motion fields, we use both the sparse 3D point cloud and the content of the video frames as a guide in warping each input video frame into its corresponding output video frame.

Specifically, we compute an output video sequence from the input video such that each output video frame $\tilde{I}_t$ is a warp of its corresponding input frame I_t. We use a sparse 3D point cloud as guidance, which we can project into both the input and output cameras, yielding two sets of corresponding 2D points: $\tilde{V}$ in the output, and V in the input. Each k pair of projected points yields a 2D displacement $\tilde{V}_k - V_k$ that can guide the warp from input to output. As soft constraints, we employ relative optical flow to retain full frame coherence in each frame. The problem remaining is to create a dense warp guided by this sparse set of displacements. This warp, which can use the displacements as either soft or hard constraints, should maintain the illusion of a natural video by maintaining temporal coherence and not distorting scene content.

We fit a full-frame warp to the sparse displacements, using a homography. This method can achieve good results if the depth variation in the scene is not large, or if the desired camera path is very close to the original. In the general case, a homography is too constrained to sufficiently fit the desired displacements. This deficiency of Liu's method [4] is that can result in undesired distortion, and temporal wobbling, but our approach can avoid these problems by employing relative optical flow to constrain temporal coherence. So, our method is the best of the alternatives we have considered up to now.

A. 2D-3D hybrid warps

Inspired by Liu's method [4] and Wang's method [13], Our 2D-3D hybrid warps integrate 3D displacement constraints and 2D relative motion constraints in one routine. It is designed with two principles in mind. First, the sparse displacements should be treated as soft constraints, since hard constraints will lead to distortions near occlusions and temporal incoherence. Instead, we wish to spread the error near occlusions slowly across the rest of the image and into areas where the eye will notice them less. Second, the warp should attempt to preserve the content of the image. This second goal is met in two ways.

First, we make the observation that since the desired camera will not be very far from the original camera, the local content in the original image should not be distorted significantly. So, we would like to avoid local shearing or non-uniform scaling. We therefore prefer warps that locally resemble a similarity transformation. (Recent work in image warping [14], [15], [16] explore both as-similar-as-possible and as-rigid-as-possible warps, and generally reach the conclusion that rigid transformations are better.) Second, we observe that violations of this constraint will be less noticeable in less salient regions of the image, while the shape of strong edges should be better preserved.

A solution best satisfying the above principles in a least-squares sense can be computed by discretizing the warp into a grid and minimizing an energy function of three weighted energy terms: a data term for each sparse displacement, and a similarity transformation term that measures the deviation

[1] http://www.zjucvg.net/acts/acts.html
[2] http://www.digilab.uni-hannover.de

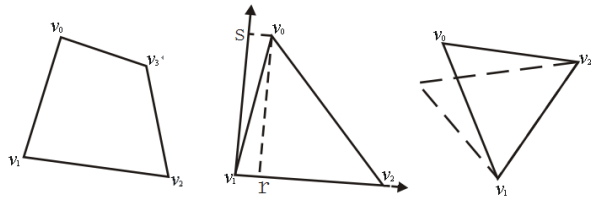


Figure 1: A triangle vertex can be expressed in the local coordinate system (r, s) of its opposite edge. The deviation from a similarity transformation of a warp can be measured as the distance between the vertex and the location it would have had under a similarity transformation(dashed lines).

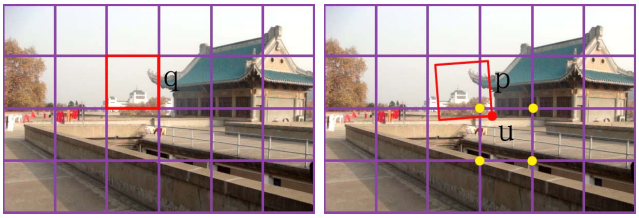


Figure 2: The corresponding quads q and p are determined based on optical flow. In this example, we represent the bottom right vertex of p, denoted by u, using linear combination of the yellow mesh vertices.

of each grid cell from a similarity transformation weighted by the salience of the grid cell, and a temporal coherence term that measures the local motion difference.

We divide the original video frame I_t into an $n \times m$ uniform grid mesh, where $v^t_{i,j}$ is the grid vertex at position (i, j) in frame t. We compute a warped version of this grid for the output video frame, where each $\tilde{v}^t_{i,j}$ is a 2D unknown to be computed.

1) Data term: Each projected point V_k in the input frame is typically not coincident with a vertex $v_{i,j}$, so we must represent each constraint with a bilinear interpolation of the four corners of the enclosing grid cell. We define v_k as a vector of the four vertices enclosing the grid cell that V_k projects to; $\tilde{v}_k$ represents the same four vertices in the output grid. The vector w_k contains the four bilinear interpolation coefficients that sum to 1, so that $V_k = w_k^T v_k$ represents a bilinear interpolation operation for a projected point. We compute w_k by finding the grid cell that V_k projects to and inverting its bilinear interpolation [17]. Then, the data term is

$$E_d = \sum_k \|w_k^T \tilde{v}_k - \tilde{V}_k\|^2 \quad (1)$$

where $\tilde{v}_k$ contains four unknowns, and w_k and $\tilde{V}_k$ are known. This term minimizes the distance between the output projected point $\tilde{V}_k$ and the interpolated location in the grid cell in the output that corresponds to the input grid cell containing V_k.

2) Similarity transformation term: The similarity transformation term measures the deviation of each output grid cell from a similarity transformation of its corresponding input grid cell. We split each grid cell into two triangles and then apply the method of Igarashi et al. [14]. As shown in Figure 1, each vertex can be represented in a local coordinate system formed by the vector between the other two vertices, and the 90 degree rotation of that vector. For example, v_0 can be defined using v_1 and v_2 as

$$v_0 = v_1 + r(v_2 - v_1) + sR_{90}(v_2 - v_1),$$

$$R_{90} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

where r, s are the known coordinates within the local coordinate system. However, if the output triangle has not undergone a similarity transformation from its input, v_0 will not coincide with the location calculated from v_1 and v_2.We therefore minimize the distance between v_0 and its desired location under a similarity transformation.

$$E_s(v_0) = w_s\|v_0 - (v_1 + r(v_2 - v_1) + sR_{90}(v_2 - v_1))\|^2$$

where w_s is saliency sum of grid cells which include vertex v_0. Notice that there are two such triangles for each of the four quads cornered by v_0. The full energy term $E_s(v)$ is formed by summing over all eight triangles of each vertex. Using all eight triangles is redundant, but avoids special handling along the grid boundaries.

3) Temporal coherence term: To achieve temporally coherent video stabilization, we use the temporal coherence preserving method proposed by Wang et al. [13] and employ an energy term to preserve the motion information, such that flickering and waving artifacts can be minimized. Our method employs Anisotropic Huber-L^1 method[18] to count optical flow. Given the optical flow, we still can not constraint the evolution of every quad directly. But, we can compute related optical flow at the foundation of prewarping with full-frame stabilization[8].

Relative optical flow determine the evolution of every quad q_i^t in the following frame, denoted as p_i^{t+1}. We find the best fitting linear transformation T_i^t such that $T_i^t(q_i^t) \approx p_i^{t+1}$ (we do not include translation in T_i^t since it will be eliminated and we are only considers the transformation of the shape of each quad, not its precise location). Our goal is to preserve this transformation in the stabilized video, so we formulate the following energy term:

$$E_c(q_i^t) = \|T_i^t(\tilde{q}_i^t) - \tilde{p}_i^{t+1}\|^2 \quad (2)$$

Note that the simple energy above encompasses full-frame stabilization motion. What remains is to properly formulate it in terms of our unknowns, the mesh vertex positions. Denote the vertices of p_i^{t+1} by u_j^{t+1}; we represent each of these vertices as a linear combination of the grid mesh

vertices v_d^{t+1} in the immediate vicinity (see Figure 2):

$$u_j^{t+1} = \sum_d w_d v_d^{t+1}, \tag{3}$$

where w_d are the barycentric coordinates w.r.t. the quad vertices v_d^{t+1}. Now we can properly reformulate equation(2) in terms of the $\tilde{v}_i^t$s:

$$E_\gamma(q_i^t) = \sum_{(j,k)\in e(q_i^t)} \|T_i^t(\tilde{v}_j^t - \tilde{v}_k^t) - (\tilde{v}_j^{t+1} - \tilde{v}_k^{t+1})\|^2,$$

where $e(q_i^t)$ is the set of edges of quad q_i^t.

B. Least square optimization

The sum of the above weighted energy terms E_d, E_c and E_s is a linear least-squares problem in the set of unknown grid vertices $v_{i,j}$. The final energy equation is

$$E = E_d + \alpha E_s + \beta E_c \tag{4}$$

where α and β are the relative weights of three terms. This energy equation is quadratic and can be minimized by solving a sparse linear system, where the matrix is narrow-banded since each vertex only shares constraints with its neighbor vertexes. We solve it efficiently using OpenNL Library which is a open source GPU-based conjugate gradient solver of [19] with multigrid strategy, and more memory- and time-efficient. While Liu's method [4] which computes the least square problem once a frame, we compute all frames vertexes in one time, since local motion result in relations between different frames. The target video can then be rendered using a standard texture mapping algorithm according to the warped mesh. This approach alone can generate good results for many sequences; we show a successful result in supplement Video Figure 2.

IV. Results

We implemented our algorithm on a desktop PC with Pentium(R) Duo-Core 2.60 GHz CPU and Nvidia Geforce 9800GT graphics card. Supplement Video Figure 1 is a runtime screen recording video of our program. We tested our approach on 69 video sequences, from 170 to 1223 frames. Each video was shot hand-held, with in-camera stabilization turned off. These videos were captured by two cameras in many different scenes.

We show comparisons to other methods such as our implementation of 2D stabilization and 3D video stabilization in supplement Video Figure 2 and Video Figure 3. We compared our results to the 3D video stabilization approach of Liu *et al.*[4], and Zhang *et al.*[5]; to the 2D video stabilization algorithms of Matsushita *et al.* [8], and Liu *et al.*[1]; and to the public available 2D video stabilization software Deshaker[3]. Video Figure 4 shows more results of our method.

[3]http://www.guthspot.se/video/deshaker.htm

As the results in Video Figure 2 show, our method is able to achieve more stable camera motions than the 2D techniques while avoiding the ghosting of moving scene objects found in previous 3D techniques. Figure 3 and Video Figure 3 shows that our method can also produce better results than Liu's method [4]. It will make frames of video be distorted when the 3D information was not recovered successfully, but ours method will not, because our method not only employs 3D recovered information to constrain the result frame, but also uses 2D motion information to restrict result frames to be temporal coherence. Like Liu's method[4] that our method can simulate a range of 3D camera motions too, from simple low-pass filtering to linear camera paths with constant speed, depending on whether the user desires a more hand-held or rail-mounted camera feel. Our results are only marginally better than the 2D stabilization alternatives when the output camera path is close to the original, such as camera paths created from low-pass filtering. However, when we specify more significantly different output paths such as linear or parabolic motions, the improvements over 2D stabilization are very clear.

V. Conclusion

In this paper we described an 2D-3D hybrid video stabilization technique which combines virtues of 2D and 3D video stabilization methods in one routine. When SFM performs well, our method can achieve strongly stabilize , high quality appearance. When SFM fails, our method degenerate to full-frame 2D video stabilization which can achieve a relative stable video, and will not result in frame distort like 3D video stabilization. The 3D method can achieve more stable camera motion, and 2D motion information makes the result more temporal coherent. These two component make our method more robust than traditional 2D video stabilization and 3D video stabilization methods.

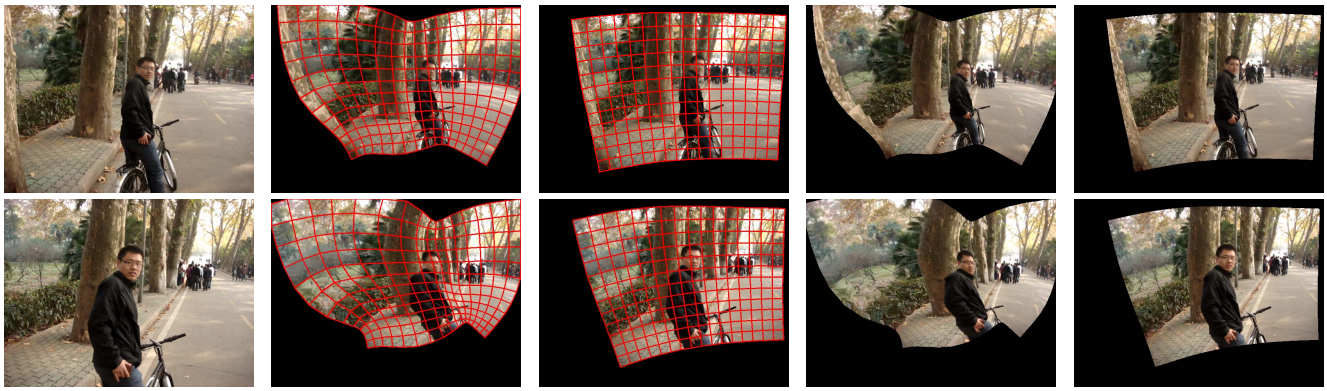
We build upon existing approaches to 3D stabilization, but are able to avoid distortion of moving scene objects and video frames by adding the constraint that each output frame be rendered as a warp of a single input frame. The key advantage of our method is that it can produce more stable video results and avoid distortion which will be introduced with Liu's method [4].

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant No. 61070078), and the Fundamental Research Funds for the Central Universities.

References

[1] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala, "Subspace video stabilization," *ACM Transactions on Graphics (TOG)*, vol. 30, no. 1, p. 4, 2011.



(a) Input frame (b) Liu 09 with grid (c) Ours with grid (d) Liu 09 without grid (e) Ours without grid

Figure 3: Comparison of warping results. Images in the first column are two frames of a video, in the second column the results generated using Liu's [4] method with grid, and in the third column the warping results which were produced using our method with grid. Images in the fourth and fifth columns are respect results to Liu's method and ours method without grid. The first row and second row come from two different frame of input video and result videos.

[2] C. Morimoto and R. Chellappa, "Evaluation of image stabilization algorithms," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 5. IEEE, 2002, pp. 2789–2792.

[3] C. Buehler, M. Bosse, and L. McMillan, "Non-metric image-based rendering for video stabilization," 2001.

[4] F. Liu, M. Gleicher, H. Jin, and A. Agarwala, "Content-preserving warps for 3D video stabilization," in *ACM SIGGRAPH 2009 papers*. ACM, 2009, pp. 1–9.

[5] G. Zhang, W. Hua, X. Qin, Y. Shao, and H. Bao, "Video stabilization based on a 3D perspective camera model," *The Visual Computer*, vol. 25, no. 11, pp. 997–1008, 2009.

[6] M. Pilu, "Video stabilization as a variational problem and numerical solution with the viterbi method," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1. IEEE, 2004, pp. I–625.

[7] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch, "Visual modeling with a hand-held camera," *International Journal of Computer Vision*, vol. 59, no. 3, pp. 207–232, 2004.

[8] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H. Shum, "Full-frame video stabilization with motion inpainting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1150–1163, 2006.

[9] M. Gleicher and F. Liu, "Re-cinematography: Improving the camerawork of casual video," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP)*, vol. 5, no. 1, pp. 1–28, 2008.

[10] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge Univ Pr, 2003.

[11] A. Fitzgibbon, Y. Wexler, and A. Zisserman, "Image-based rendering using image-based priors," *International Journal of Computer Vision*, vol. 63, no. 2, pp. 141–151, 2005.

[12] P. Zitnick, N. Agarwala, M. Agrawala, M. Cohen, B. Curless, and S. Kang, "Using Photographs to Enhance Videos of a Static Scene."

[13] Y. Wang, H. Lin, O. Sorkine, and T. Lee, "Motion-based video retargeting with optimized crop-and-warp," in *ACM SIGGRAPH 2010 papers*. ACM, 2010, pp. 1–9.

[14] T. Igarashi, T. Moscovich, and J. Hughes, "As-rigid-as-possible shape manipulation," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 1134–1141, 2005.

[15] S. Schaefer, T. McPhail, and J. Warren, "Image deformation using moving least squares," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 533–540, 2006.

[16] M. Alexa, D. Cohen-Or, and D. Levin, "As-rigid-as-possible shape interpolation," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 157–164.

[17] P. Heckbert, *Fundamentals of texture mapping and image warping*. Citeseer, 1989.

[18] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof, "Anisotropic Huber-L1 optical flow," in *Proceedings of the British Machine Vision Conference (BMVC)*, London, UK, September 2009.

[19] L. Buatois, G. Caumon, and B. Levy, "Concurrent number cruncher: a gpu implementation of a general sparse linear solver," *Int. J. Parallel Emerg. Distrib. Syst.*, vol. 24, pp. 205–223, June 2009. [Online]. Available: http://portal.acm.org/citation.cfm?id=1552466.1552468