# Depth Enhanced and Content Aware Video Stabilization

A. Lindner, K. Atanassov, S. Goma
Qualcomm Incorporated., San Diego, CA, USA

## ABSTRACT

We propose a system that uses depth information for video stabilization. The system uses 2D-homographies as frame-pair transforms that are estimated with keypoints at the depth of interest. This makes the estimation more robust as the points lie on a plane. The depth of interest can be determined automatically from the depth histogram, inferred from user input such as tap-to-focus, or selected by the user; i.e., *tap-to-stabilize*. The proposed system can stabilize videos on the fly in a single pass and is especially suited for mobile phones with multiple cameras that can compute depth maps automatically during image acquisition.

**Keywords:** video stabilization, depth map, stereo vision, mobile imaging

## 1. INTRODUCTION

Video stabilization is an important topic due to the omnipresence of mobile phones that enable amateur users to spontaneously capture videos without equipment such as a Steadicam stabilizer. The small form factor of phones and the competitive pricing make software solutions a priority.

Video stabilization algorithms usually have three stages [1]. First, the movement of the camera with respect to the scene is estimated based on temporal tracking of the scene content. Then the resultant trajectory is filtered such that jitter is reduced but desired motions such as smooth panning and rotation are preserved. Finally, the video sequence is retargeted for the new smoothed trajectory. Each of these stages has challenges to overcome, but in our work we focus on the first stage. We do this by exploiting depth information, which is becoming more and more available on mobile phones equipped with symmetric or asymmetric stereo cameras such as the HTC One M8 phone and the Amazon Fire phone.

In order to track camera movement, one must determine a mapping function that projects image points from one frame to the subsequent frame. Mapping approaches fall into two broad categories: 2D and 3D. The 3D approaches [2] try to reconstruct the camera pose and the scene in three dimensional space, which is to some extent related to SLAM [3]. The drawback of 3D approaches is that they are computationally expensive and prone to errors leading to unpleasant visual artifacts. Therefore, we often see 2D approaches that model the camera motion as a frame-to-frame transform, in many cases in the form of a homography.

Homographies, or specializations thereof, have been found to adequately describe camera motion. A plethora of algorithms and applications have been published [4] [5], which is out of the scope of this article to summarize. The advantage of homographies over more complex image transforms is that they can be written as a linear system in homogeneous image coordinates and that they are relatively inexpensive to estimate. This advantage comes at the price that homographies are only valid for points that lie on a planar surface in the 3D scene; e.g., a building facade. Thus, scenes with objects at different depths and depth discontinuities are not well modeled.

We propose to use selective depth information in order to compute a homography, such that only those points that have the same depth; i.e., the points that lie on a plane, are used. With this technique the user can, for example, choose to stabilize a person in the foreground independent of the potentially moving background. Or the user can stabilize on the background independent of objects in the foreground. This has two main advantages: (1) the homography estimation is more robust (points with different depths that cannot be modeled are rejected) and (2) the user can choose which part of the scene to use for stabilization.

It is not necessary that the user explicitly state which part of the scene has to be stabilized. It is, for example, reasonable to assume that those areas where the user taps to focus are also important for the stabilization. Alternatively, the algorithm can default on the closest object, which can be determined from the depth histogram.

Mobile Devices and Multimedia: Enabling Technologies, Algorithms, and Applications 2015, edited by Reiner Creutzburg, David Akopian, Proc. of SPIE-IS&T Electronic Imaging, Vol. 9411, 941106
© 2015 SPIE-IS&T · CCC code: 0277-786X/15/$18 · doi: 10.1117/12.2083443

## 2. SPARSE DEPTH MAP FROM STEREO VIDEO

We used a Sony 3D Handycam camcorder (HDR-TD10), which records a stereo video in full HD resolution; i.e., two video streams with 1920x1080 pixels each. The camera has an optical image stabilization that was disabled for the experiments. A keypoint in the scene is imaged onto the sensor plane at different locations for both of the stereo sensors as shown in Figure 1. The displacement $d = x_L - x_R$ of an imaged keypoint from one sensor to the other determines the estimated depth $Z = fB/d$.

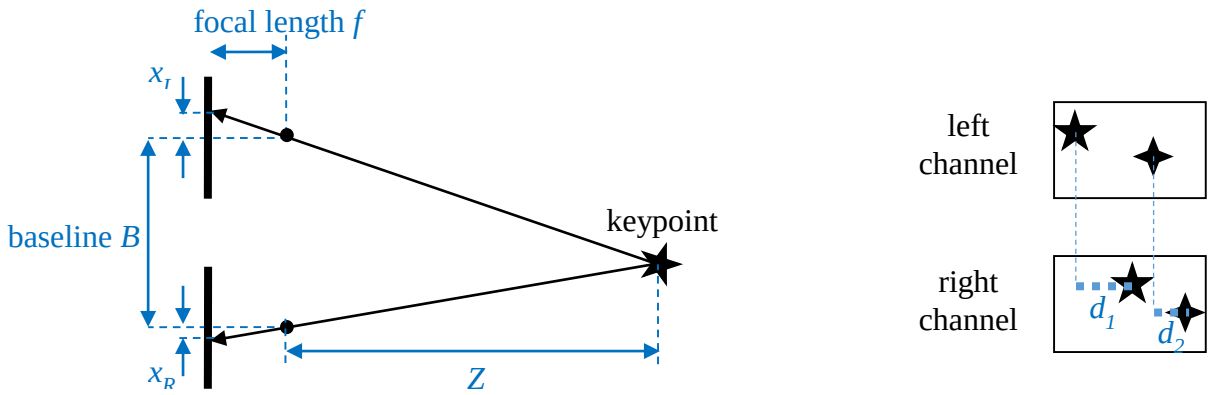



Figure 1. Schematic drawing of a stereo vision camera for depth estimation. Right: Schematic drawing of two keypoints in both video channels with displacements indicated by thick dashed lines; as $d_1 > d_2$ it follows $Z_1 < Z_2$.

In order to speed up the computation of the sparse depth map we downsampled the video frames to a processing resolution of one fourth the original size in each dimension. We used the FAST keypoint detector [6] to find corner points in the left frame and used a two-dimensional cross correlation to find the matching corner along the epipolar line in the corresponding right frame. As the stereo camera is calibrated, the epipolar line is horizontal and at the same vertical position. This process is illustrated on the right hand side of Figure 1 and in the flowchart in Figure 3. Starting with a keypoint in the left image, we go to the same location in the right channel image and find the most similar neighborhood along a horizontal line. The similarity is measured with normalized cross correlation of patches of size 11x11 pixels. The result is a sparse depth map with a displacement value for each matched keypoint location.

Figure 2 shows on the left hand side a stereo image pair where the matched keypoints are connected by a thin red line; it is visible that the horizontal displacement is larger for the keypoints on the soda can in the foreground than on the book in the background. The foreground and background result in two distinct peaks in the histogram shown on the right hand side.

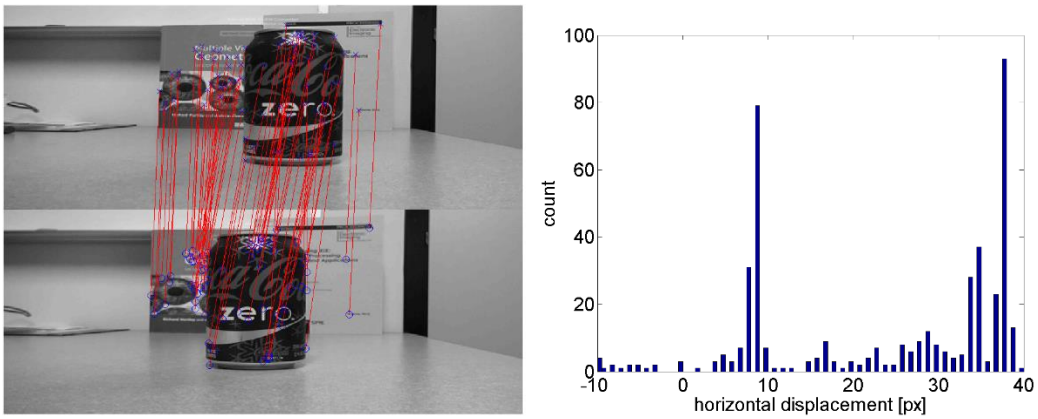



Figure 2. Left: stereo image pair with matched keypoints (connected by a red line). The horizontal displacement is larger in the foreground than in the background. For better visualization we show only 20% of the keypoints. Right: histogram of horizontal displacements showing two peaks for the foreground and background, respectively.

# 3. VIDEO STABILIZATION

Our full solution is based on the three stage approach to video stabilization introduced in [1]:

1. Motion estimation: extract frame-pair transforms to model camera motion.
2. Motion compensation: remove jitter while maintaining intentional panning and rotation of camera.
3. Video retargeting: use compensated motion parameters to produce a smooth output video.

The following subsections explain each of these steps of our implementation in detail.

### 3.1 Motion Estimation

We describe the temporal transformation between two consecutive video frames as a 2D transformation. Specifically we use a similarity transform *S*, which is a special type of homography and is defined as

$$(x' \quad y' \quad 1) = (x \quad y \quad 1)\begin{bmatrix} a & b & 0 \\ c & d & 0 \\ e & f & 1 \end{bmatrix} = (x \quad y \quad 1)\underbrace{\begin{bmatrix} k\overbrace{\begin{bmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{bmatrix}}^{R} & 0 \\ \begin{bmatrix} t_x & t_y \end{bmatrix} & 1 \end{bmatrix}}_{S} \qquad \textit{Equation 1}$$

where *x, y* and *x', y'* are the x and y coordinates before and after the transform, *k* a scaling parameter, *R* a rotation matrix of angle $\varphi$, and $[t_x \quad t_y]$ an offset in x and y directions, respectively. We use a similarity transform (4 DOF) instead of a full homography (8 DOF) because it is more robust in cases where few keypoints are available. Even with outlier rejection, high-DOF homographies can overfit to noisy data and produce visually unpleasing results as also pointed out by Grundmann et al. [5]

Under the pinhole camera model assumption a homography is valid to map projected points from one frame to the next only if they lie on a plane in the 3D world coordinate system. Depth discontinuities thus pose a problem to estimate the transform parameters because points from either side of the discontinuity cannot be modeled with the same transform. This is the reason why outlier rejection; e.g., using RANSAC, is necessary to make the model estimation more robust.

In the example depicted in Figure 2, either the foreground or the background can then be rejected depending on where the algorithm finds the stronger consensus. However, in this example it is not evident whether the foreground or the background will dominate as they have approximately equal numbers of keypoints; it is even possible that different regions become dominant throughout the video sequence. Finally, the user can be interested in a particular region even though it does not numerically dominate over the others.

We propose to use the scene's depth information to resolve this issue. For instance, if a depth of interest is known, the algorithm can estimate the transform using only keypoints that lie on the appropriate plane while rejecting all others. The depth of interest can be determined by a choice of inputs such as the camera focus, the user's tap-to-focus input, or a default to the foreground.

Given two consecutive stereo frames (see Figure 3 and Figure 4) we first find FAST [6] keypoints *kp* in the left image of frame $n-1$. We then estimate the horizontal displacements *d* in the right image of the same frame, which indicates the depth of the keypoints in frame *n-1*. Then, the keypoints are pruned according to the depth of interest $D_{interest}$ for the stabilization resulting in the final keypoints $kp_{n-1}$. We then compute the keypoints of the left image of the subsequent frame $kp_n$. We then extract FREAK features [7] of the left images of both frames at their respective keypoint locations. Finally the features $f_{n-1}$ and $f_n$ of both images are matched and MSAC [8] is used to estimate the transform $S_n$.

Figure 5 illustrates the horizontal offset $t_x$ of a video sequence (see Figure 2) as a function of the frame number *n*. The sequence was processed twice, once for the background and once for the foreground. The two different curves show that the foreground and background objects move along different trajectories, which is due to the parallax. Consequently, it is impossible to stabilize both the foreground and background at the same time, so a choice has to be made. This is especially the case with scenes that have strong depth discontinuities and camera trajectories with large translational components.

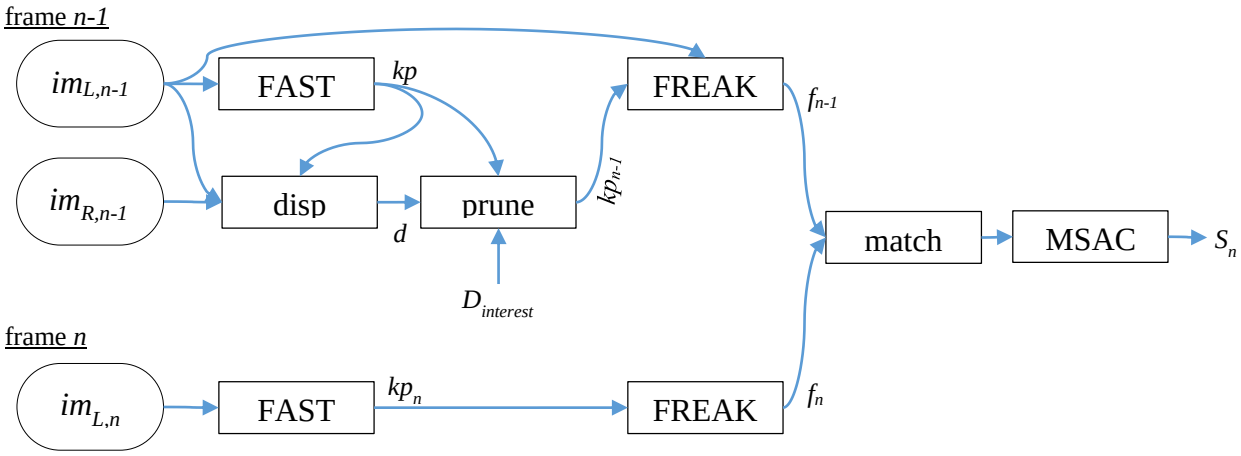



Figure 3. Flowchart of the algorithm to estimate the depth-aware similarity transform $S_n$ from images of two consecutive video frames.

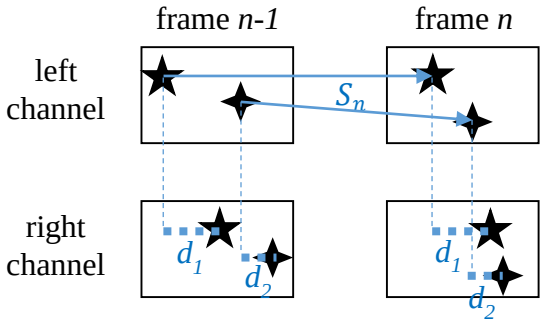



Figure 4. Two consecutive frames with left and right channel images. The mapping of the keypoints from the left to the right channel image of the same video frame are independent horizontal displacements. The mapping of the keypoints between consecutive frames is a common similarity transform $S$ (or more generally a homography).

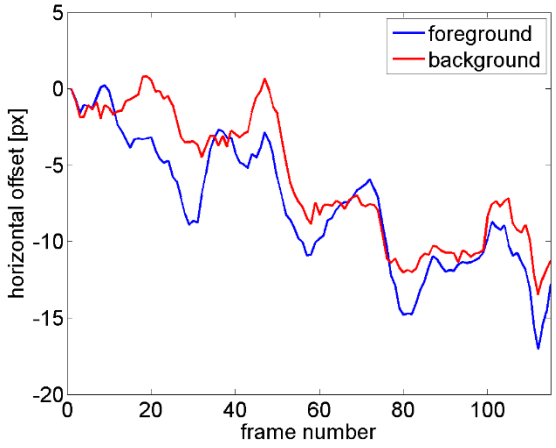



Figure 5. Horizontal offset $t_x$ extracted from a single video sequence (see Figure 2) with depth of interest at foreground and at background, respectively. The fact that the two curves are different shows that the foreground and background move along different trajectories. The stabilization can thus be done for either of them, but not for both at the same time.

### 3.2 Motion Compensation

The similarity transforms $S_n$ from the previous step describe the mapping of the image between consecutive frames *n-1* and *n*. The absolute movement starting at the beginning of the sequence can thus be expressed with the cumulative transform $C_n = S_1 S_2 \ldots S_n$ where $S_1$ is initiated as the unity transform. $C_n = C_{n-1} S_n$ can be computed recursively for $n > 1$.

We decompose the cumulative transform at frame *n* into its scaling parameter *k*, rotation angle $\varphi$, and horizontal and vertical offsets $t_x$ and $t_y$, respectively. We then use a simple IIR filter to compute the smoothed translation. Let $p_n$ one of the four parameters at frame *n*, then the smoothed parameter $\widetilde{p_n} = \alpha p_{n-1} + (1-\alpha) p_n$ where $\alpha$ controls the smoothening effect. In our implementation we found $\alpha = 0.9$ to yield good results. Figure 6 shows the cumulated vertical offset parameter $t_y$ of an example sequence. The red dotted line shows the raw parameter extracted from the frame-pair transforms, and the blue solid line shows the parameter after IIR smoothing.

There exist more elaborate ways to smooth frame-pair transforms [5] [4] [9], but we found this simple IIR approach to produce good results, and it can be computed at much lower computational costs on the fly while the sequence is being processed.

After filtering, the smoothed cumulative transforms $\widetilde{C_n}$ are reassembled according to the smoothed parameters $\widetilde{p_n}$. This is done by simply inserting the four parameters into the *S* matrix on the right hand side of Equation 1.

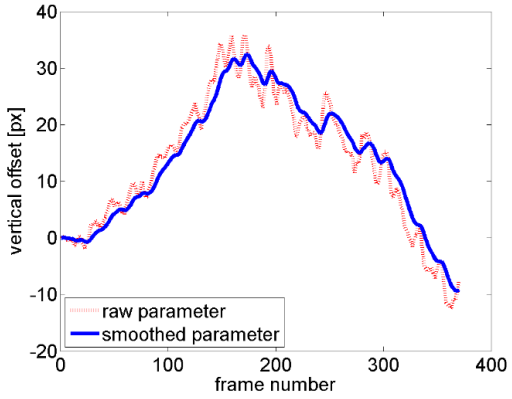



Figure 6. Vertical offset $t_y$ of the cumulative transforms before and after smoothing with an IIR filter.

### 3.3 Video Retargeting

Given the cumulative transforms $C_n$ and their smooth versions $\widetilde{C_n}$ the retargeting transform can be simply computed as $T_n = {C_n}^{-1} \widetilde{C_n}$, since the first frame of the original and the smoothed sequence are linked with an identity transform *I*.

This is illustrated in Figure 7.

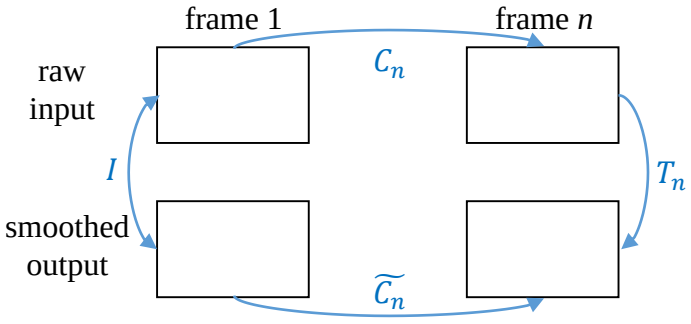



Figure 7. The retargeting transform for frame *n* is computed as ${T_n} = {C_n}^{-1} \widetilde{C_n}$

Both the left and right channels can be used with the same retargeting transforms since the two sensors do not move with respect to each other. In our experiments we use the left channel as input video. If the stereo cameras have different resolutions; e.g., with the HTC One M8, the higher resolution sequence is used.

## 4. CONCLUSIONS

We presented a system for video stabilization. Similar to other methods, we use homography transforms to model the camera motion between frame-pairs. The drawback of homographies is that they can only model keypoints that lie on a planar surface, making the model theoretically unsuitable for scenes with depth continuities. This is the reason why in practice robust techniques such as RANSAC are used to estimate reliable parameters. The problem with this approach is that the wrong region of the image yields the strongest consensus and dominates the stabilization.

Our solution to this inherent problem is to use depth information in order to find keypoints that are guaranteed to lie on a planar surface. This can be keypoints in the background or in the foreground at a specific depth of interest. This depth of interest can be determined automatically from the depth histogram, inferred from user input such as tap-to-focus, or explicitly chosen by the user; i.e., *tap-to-stabilize*.

Once the depth of interest is known, keypoints at that depth are collected and used to estimate the frame-pair homography. In our case we use a similarity transform, a specialization of a homography with fewer degrees of freedom. Low DOF transforms have less tendency to overfit to noisy data, which is important in our case as we operate on fewer keypoints due to the depth pruning. The estimation process is visualized in the flowchart in Figure 3.

The motion compensation is done with an IIR filter although alternative methods can be used within our stabilization pipeline. Finally, we compute a retargeting transform that compensates for jitter in the camera path.

## 5. REFERENCES


[1] C. Morimoto and R. Chellappa, "Evaluation of image stabilization algorithms," in IEEE International Conference on Acoustics, Speech and Signal Processing, 1998.

[2] S. Liu, Y. Wang, L. Yuan, J. Bu, P. Tan and J. Sun, "Video Stabilization with a Depth Camera," in IEEE Conference on Computer Vision and Pattern Recognition, 2012.

[3] M. Montemerlo and S. Thrun, FastSLAM: A Scalable Method for the Simultaneous Localization and Mapping Problem in Robotics, Springer, 2010.

[4] M. Gleicher and F. Liu, "Re-Cinematography: Improving the Camera Dynamics of Casual Video," in ACM Multimedia, 2007.

[5] M. Grundmann, V. Kwatra and I. Essa, "Auto-Directed Video Stabilization with Robust L1 Optimal Camera Paths," in IEEE Conference on Computer Vision and Pattern Recognition, 2011.

[6] E. Rosten and T. Drummond, "Machine Learning for High-Speed Corner Detection," in European Conference on Computer Vision, 2006.

[7] A. Alahi, R. Ortiz and P. Vandergheynst, "FREAK: Fast Retina Keypoint," in IEEE Conference on Computer Vision and Pattern Recognition, 2012.

[8] P. Torr and A. Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry," Computer Vision and Image Understanding, vol. 78, p. 138–156, 2000.

[9] B.-Y. Chen, K.-Y. Lee, W.-T. Huang and J.-S. Lin, "Capturing Intention-based Full-Frame Video Stabilization," Computer Graphics Forum, vol. 27, no. 7, pp. 1805 - 1814, 2008.


Video links:

Video 1: http://dx.doi.org/10.1117/12.2083443.1
Video 2: http://dx.doi.org/10.1117/12.2083443.2
Video 3: http://dx.doi.org/10.1117/12.2083443.3
Video 4: http://dx.doi.org/10.1117/12.2083443.4