May 03, 2018

Intent Preserving 360 Video Stabilization Using Constrained Optimization

Chetan Arora

Vivek Kwatra

INTENT PRESERVING 360 VIDEO STABILIZATION USING CONSTRAINED OPTIMIZATION

ABSTRACT

A system and method are disclosed, that solve for rotational updates in 360 videos by removing camera shakes, while preserving user intended motions. The method uses a constrained nonlinear optimization approach in quaternion space. At first, optimal 3D camera rotations are computed between key frames. 3D camera rotations between consecutive frames are then computed. The first, second, and third derivatives of the resulting camera path are minimized, to stabilize the camera orientation path. The computation strives to find a smooth path, while also limiting its deviation from the original path. The system keeps the orientations close to the original, for example, even when the videographer takes a turn. Each frame is then warped to the stabilized path, which results in a smoother video. The rotational camera updates may be applied to the input stream at source or added as metadata. The technology may influence standards by making rotational updates metadata a component of 360 videos.

KEYWORDS: 360 degree video, camera rotation, removing camera shake, computing camera rotation

BACKGROUND

360 videos capture the entire 360-degree field of view (FOV) around a camera. These videos are typically viewed inside a VR headset. Cameras that allow 360 capture are becoming commonplace. While there exist high-end multi-camera systems that allow capturing full 3D (stereo) 360 videos, more affordable and convenient handheld devices are gaining in popularity. One problem with handheld cameras is the camera shake caused during capture. Any shake in 360 videos is highly undesirable because besides being aesthetically displeasing, it can cause

motion sickness to the viewer when viewing these videos inside a VR headset. Hence stabilization of 360 videos is extremely important for making them watchable. Stabilization of 360 videos is different from regular video stabilization. Predominantly it is the rotational shake that is of concern in 360 videos, since that is the major factor interfering with the user's head motion. The entire field of view around the camera that can be used to aid and improve the motion estimation and stabilization quality is captured. When viewing a 360 video, the user is usually presented a small portion (viewport) of the entire 360 view. This viewport can be changed by the user using his/her head motion, when viewing the video in VR or using a mouse or touch-based UI when watching the video on a desktop or mobile. To the last point above, if the user does not change the viewport during the video, then any apparent motion in the viewport is caused by the camera motion, which could either correspond to the camera shake or an actual intentional motion of the camera by the person capturing the video.

DESCRIPTION

A system and method are disclosed that solve for rotational updates in 360 videos by removing camera shakes, while preserving user intended motions. The method uses a constrained nonlinear optimization approach in quaternion space to solve for rotational updates. The method as shown in FIG. 1 includes stabilizing the input frames around the input trajectory. In step A, the optimal 3D camera rotations R_i, are computed between key frames. This represents the rotations with respect to the first frame using quaternions. In step B, 3D camera rotations between consecutive frames B_i, are computed. The first, second, and third derivatives of the resulting camera path are minimized in step C, to stabilize the camera orientation path. The first order smoothness is given as

$$\square_{\square} = \sum_{\square \epsilon \square} \square(\square_{\square}\square_{\square} - \square_{\square-1}\square_{\square-1})\ldots\ldots\ldots(1)$$

where f is a robust loss function penalizing non-smoothness of trajectories.

The second order smoothness is given as

$$\square_{\square} = \sum_{\square\epsilon\square} \square(\square_{\square}\square_{\square} + \square_{\square-2}\square_{\square-2} - 2\square_{\square-1}\square_{\square-1})\text{.........(2)}$$

In step D, each frame is warped to the camera path to keep the output video view point near to original, even when the videographer takes a turn. The overall optimization is given as

$$\square^{*} = \square\square\square\ \square\square\square_{\square_{\square}|\square\epsilon\square}\square_{\square}\square_{\square} + \square_{\square}\square_{\square}\ \text{........(3)}$$

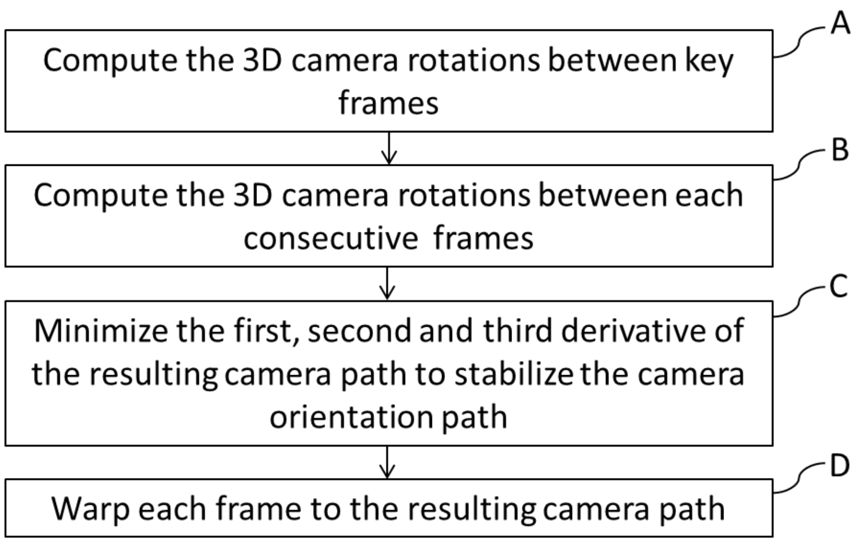

FIG. 1: Method of stabilizing the input frames around the input trajectory

To keep the update transformations small an additional constraint may be imposed on the maximum permissible angular rotation in update transformations. Since quaternions are used as representation for rotation matrices, where the first element is cos(θ/2) where θ is the angular rotation, such a constraint becomes simply a lower bound on the first element, and is easy to handle in any optimization problem solver. Rotations are composed by multiplying the 3D rotation computed from camera pose estimation with the unknown update rotation that is estimated as part of the optimization. For two rotation matrices, R_1 and R_2 represented as quaternions,

$$\square = \square_0 + \square\square_1 + \square\square_2 + \square\square_3 \ldots\ldots(4)$$

$$\square = \square_0 + \square\square_1 + \square\square_2 + \square\square_3 \ldots\ldots(5)$$

respectively, the quaternion, t, corresponding to their composition R_2R_1 is given as:

$$\square_0 = \square_0\square_0 - \square_1\square_1 - \square_2\square_2 - \square_3\square_3 \ldots\ldots(6)$$

$$\square_1 = \square_0\square_1 + \square_1\square_0 - \square_2\square_3 - \square_3\square_2 \ldots\ldots(7)$$

$$\square_2 = \square_0\square_2 + \square_1\square_3 + \square_2\square_0 - \square_3\square_1 \ldots\ldots(8)$$

$$\square_3 = \square_0\square_3 - \square_1\square_2 + \square_2\square_1 + \square_3\square_0 \ldots\ldots(9)$$

where R_2 represents the rotation from the camera pose and R_1, the update transformation to be computed. Since, camera pose, and thus r is a constant in the optimization problem, the overall expression is linear in the parameters, q. In contrast, an axis angle representation would have required complicated terms involving angle sines and cosines. From the optimization perspective, using quaternions greatly simplifies the problem. The optimization problem may also be solved using a non-linear solver such as the Ceres solver (ceres-solver.org) where the lower-bound constraint is more easily expressed. To represent rotation, q is a unit quaternion that adds up a non linear constraint given by

$$\square_0^2 + \square_1^2 + \square_2^2 + \square_3^2 = 1 \ldots\ldots\ldots(10)$$

The cos(θ/2) constraint is expressed as $\square_0 >$ *lower bound* and the quaternion q is expressed as $\square = \square_0 + \square\square_1 + \square\square_2 + \square\square_3$. FIG. 2 gives a schematic diagram describing key steps of the optimization algorithm.

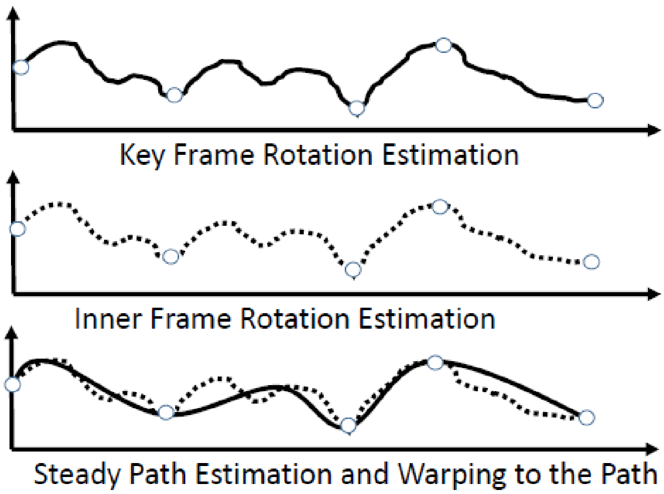

FIG. 2: schematic of steps of optimization algorithm

FIG. 3 shows stabilized orientation angles overlaid on the original orientation angles over time for two video sequences corresponding to different upper bounds on the stabilization update. This limits the degree by which the original orientation can be changed. The smoother curve after the stabilization indicates that shake in camera orientations have stabilized. At UB=6 all frames become fully registered with the first frame. FIG. 4 shows stabilization results of the optimization algorithm on a 360 degree sequence of images recorded on a jogging track.

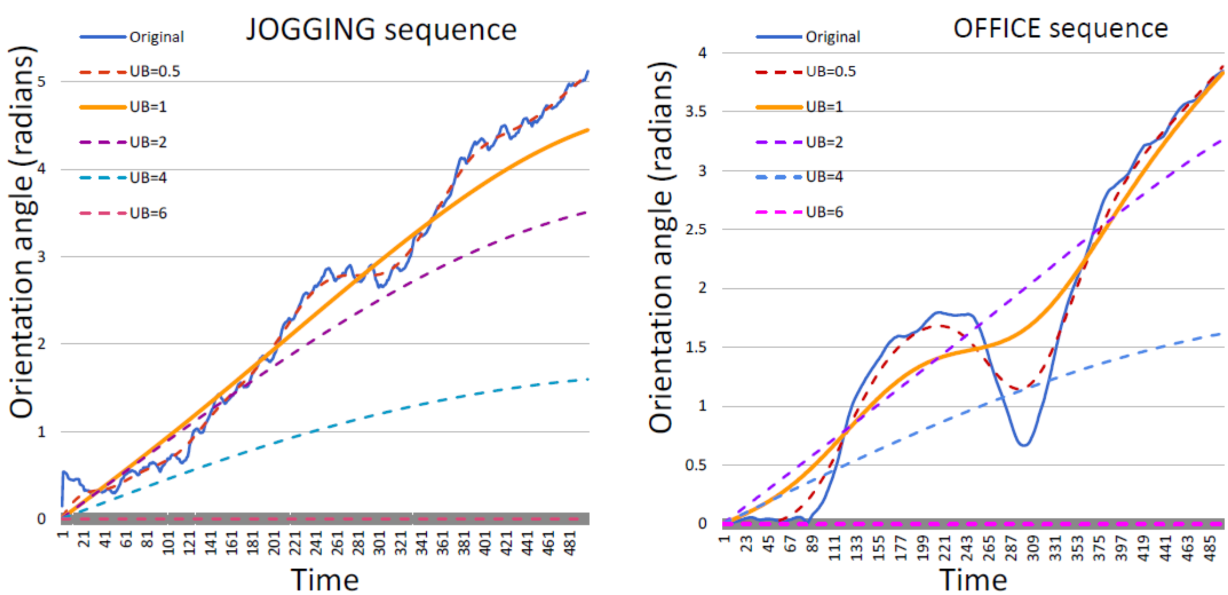

FIG. 3: Stabilized orientation angles overlaid on the original orientation angles over time for two video sequences

FIG. 4: Stabilization results on a sequence of images recorded on a jogging track

The results of stabilization on images are shown in FIG. 4, in which the 1st (top) row shows select frames from the input sequence. The 2nd row shows the stabilization result showing that frames do not deviate too far from the original, preserving the intended camera motion. The 3rd row shows jittery feature tracks computed from the input sequence, while the 4th row illustrates smooth feature tracks computed from the stabilized sequence.

The idea of computing steady optimal path based upon 3D rotations may be applied to regular narrow FOV (NFOV) videos as well. Also, determining the intent-preserving smooth camera motion may be used as an input signal for a method to evaluate the "watchability" of a certain 360 video. The method measures the amount of rotational camera motion present in the video. The constrained optimization can be used to align all frames to the "up" vector, where the "up" vector may be determined either through heuristic methods, or through machine-learning based approaches. The machine-learning based approach may learn to infer the “up” direction from appearance alone. The output rotational camera updates may be applied to the input stream at source to pre-compute the stabilized video or may be added to the video as metadata. The output rotational camera updates are picked up by the 360 video players from the metadata, and applied on the fly. To select the viewport based on user’s head position or UI actions, 360 video players may also apply rotational updates to the video through metadata. Hence, one way the technology could influence standards is by way of making rotational updates metadata a component of 360 videos. The key advantage of the method is that user intent in camera motions is preserved while undesirable camera shake is removed.