Video Stabilization using Epipolar Geometry

Amit Goldstein
and
Raanan Fattal
Hebrew University of Jerusalem, Israel

We present a new video stabilization technique that uses projective scene reconstruction to treat jittered video sequences. Unlike methods that recover the full three-dimensional geometry of the scene, this model accounts for simple geometric relations between points and epipolar lines. Using this level of scene understanding, we obtain the physical correctness of 3D stabilization methods yet avoid their lack of robustness and computational costs. Our method consists of tracking feature points in the scene and using them to compute fundamental matrices that model stabilized camera motion. We then project the tracked points onto the novel stabilized frames using epipolar point transfer and synthesize new frames using image-based frame warping. Since this model is only valid for static scenes, we develop a time-view reprojection that accounts for non-stationary points in a principled way. This reprojection is based on modeling the dynamics of smooth inertial object motion in three-dimensional space and allows us to avoid the need to interpolate stabilization for moving objects from their static surrounding. Thus, we achieve an adequate stabilization when both the camera and the objects are moving. We demonstrate the abilities of our approach to stabilize hand-held video shots in various scenarios: scenes with no parallax that challenge 3D approaches, scenes containing non-trivial parallax effects, videos with camera zooming and in-camera stabilization, as well as movies with large moving objects.

Categories and Subject Descriptors: I.4.3 [**Image Processing and Computer Vision**]: Enhancement; I.4.9 [**Image Processing and Computer Vision**]: Applications; I.3.8 [**Computer Graphics**]: Applications

Additional Key Words and Phrases: video stabilization, novel view synthesis, image warping, epipolar geometry

1. INTRODUCTION

Shooting pleasing smooth video sequences from a moving camera is a non-trivial task which is often dealt with using rather involved solutions. Film-makers and broadcasting studios mount their cameras on wheeled dollies that roll on tracks or use steadicams, which are hand-held mechanical stabilizing systems. Often, the dolly tracks need to be customized to the particular scene and take time to set up. There are various scenarios where both solutions are inapplicable or do not provide enough stabilization. This motivates the development of computer-based video stabilization algorithms.

Existing stabilization techniques can be roughly divided into two categories based on the degree of complexity of the model they use to interpret the input video. The first class consists of two-dimensional (2D) methods that do not attempt to recover any explicit three-dimensional (3D) geometry of the scene. These methods explain the observed motion in the video using 2D transformations in the image plane and search for their parameters. Stable motions are obtained by fixating or smoothing these parameters. The excess jittered movement is removed by the same type of transformation. In general, 2D stabilization is computationally efficient, robust and is often found in still and video cameras [Lukac 2008]. However, when the video contains non-planar geometry and the camera shake is not purely rotational, this level of scene modeling is insufficient to account for parallax effects due to shake.

The second type of methods tackles these difficulties using scene modeling of a greater complexity where various 3D quantities are recovered. These 3D stabilization methods [Buehler et al. 2001; Bhat et al. 2007; Liu et al. 2009] track a sparse set of feature points along the video and use the correspondences to recover the 3D camera pose and the 3D location of every point using structure-from-motion (SFM) [Hartley and Zisserman 2000]. Stabilization is then achieved by computing smoother camera paths and reprojecting the 3D points to construct the novel frames. Liu et al. [2009] demonstrate stabilization of scenes with highly-challenging geometry by combining this approach with a content preserving warping. Nevertheless, the use of a richer model introduces various weaknesses. SFM is a non-linear problem which is typically solved using bundle-adjustment [Triggs et al. 2000] and becomes costly at moderately large numbers of tracked points. It is also unstable when the scene complexity drops below the model complexity; small camera translation or approximately planar scene geometry (far objects) make it hard to determine the 3D location variables.

In view of these shortcomings, Liu et al. [2011] smooth the tracked point trajectories in 2D and do not recover any 3D information. This corresponds, however, to a very loose model that can generate inconsistent point configurations. Therefore, Liu et al. restrict the 2D point trajectories to a low-dimensional subspace, which is known to be approximately valid for projections of static 3D scenes in short time intervals [Irani 2002]. This approach is shown to be more robust to degenerate camera motions than 3D approaches and, at the same time, handles videos containing parallax better than the existing 2D stabilization. Since the low-dimensionality assumption holds for static scenes, Liu et al. discard feature points that correspond to moving objects. As a result, moving objects are stabilized using interpolated warp constraints computed for static background points. Furthermore, this method as well as the 3D techniques mentioned above operate on a limited number of points that allow reliable tracking. In general these points are not necessarily spread evenly across the frame, leaving large portions of it with no close and relevant warping constraints.

In this paper we present a new video stabilization technique that uses a mid-level scene modeling, known as *projective reconstruction* [Hartley and Zisserman 2000]. This model does not recover explicit 3D point locations or 3D camera pose, rather it accounts for simple geometric relations between points and epipolar lines. This level of scene understanding is sufficient for obtaining the physical correctness of 3D stabilization methods while avoiding their associated drawbacks. More specifically, our method starts by tracking points along the video and uses them to compute *fundamental matrices* that encapsulate the epipolar relations between successive frames. We use these relations to generate virtual point trajectories that last long enough to define the stabilized views by filtering them in time with large smoothing kernels. We then use the correspondences between the jittered input and these smoothed trajectories to compute another set of fundamental matrices that model the stabilized camera views. Finally, the output frames are computed using the *epipolar point transfer* of Laveau and Faugeras [1994] that uses these matrices to reproject the input points to the stabilized novel views. This level of scene modeling allows us to construct physically-correct warp constraints, applies to scenes of variable depth complexity, it is valid for arbitrary uncalibrated perspective cameras and allows camera zooming and in-camera stabilization.

The classic epipolar point transfer applies only to points that correspond to static objects in 3D space. Therefore, similarly to existing methods, this approach cannot stabilize points that belong to non-stationary objects. To overcome this limitation, we derive a novel time-view point reprojection that allows us to deal with moving objects in a principled way that models the dynamics of smooth inertial object motion in space. Thus, we avoid the need to interpolate stabilization for moving objects based on their static surrounding and achieve adequate stabilization when both the camera and the object are moving. Furthermore, we describe a point tracking scheme that uses the epipolar point transfer to predict, rather than search, additional corresponding points. This use of the epipolar relations considerably reduces the aperture problem and allows us to obtain correspondences at more challenging regions of the frames. We use these additional correspondences to define warping constraints that are spread more uniformly across the frame and hence better capture the scene shape and aid the stabilization of non-planar scenes.

We demonstrate the abilities of our approach to stabilize hand-held video shots in various scenarios: scenes with no parallax that challenge 3D approaches, scenes containing non-trivial parallax effects, videos with camera zooming, as well as movies with large moving objects.

2. BACKGROUND

Here we review the existing work on video stabilization as well as provide some background on view interpolation methods that are relevant to our work.

Video Stabilization. In case of approximately planer scenes or cases were the camera shake is strictly rotational, unwanted jitters can be effectively reduced based on two-dimensional reasoning of the video. In these cases the camera jitter can be explained and removed by a homography transformation [Hartley and Zisserman 2000]. Irani et al. [1994] treat more complex scenes by computing the homography that stabilizes a dominant large planar region in a video. The stabilized motion is computed by either fully canceling or smoothing the camera rotation component. In [Gleicher and Liu 2008] the stabilized camera motion is computed by interpolating between homography transformations using matrix logarithms. Matsushita et al. [2006] discuss additional important aspects of video stabilization, such as extending the stabilized frames to become full-frames and reducing the motion blur.

Assuming the scene geometry and camera motion do fall into these categories, such 2D stabilization methods are robust, operate on the entire frame, require a small number of tracked points and consume minimal computing efforts. In fact, this type of stabilization became very common in still and video cameras where it is implemented via mechanical means, either in the lens or the camera sensor [Lukac 2008]. However, most scenes do contain objects at arbitrary depths and in many scenarios, such as hand-held camera shoots, it is virtually impossible to avoid any translational component in the camera shake. In these cases parallax effects cannot be ignored and 2D modeling is insufficient for video stabilization.

In order to cope with general scenes and camera jitter, three-dimensional modeling of the scene is used. Buehler et al. [2001] compute SFM and recover the 3D camera pose and the 3D locations of the tracked feature points. This is formulated in a general uncalibrated camera setting and solved using the bundle-adjustment method [Triggs et al. 2000]. Stabilization is then achieved by computing new regularized camera projection matrices that produce smooth trajectories and reprojecting the recovered (static) 3D point locations. The output novel stabilized frames are computed using image-based rendering technique, similar to [Buehler et al. 2001], that blends pixels from several frames. In [Bhat et al. 2007] the averaging is replaced by coherent patch-based image synthesis. Here again, the pixels come from multiple images. Constructing each stabilized frame from multiple frames requires the corresponding source pixels to be consistent and hence these techniques are inadequate for non-static scenes.

Liu et al. [2009] construct each stabilized frame individually and therefore operate better on videos containing moving objects. They suggest that for the purpose of video stabilization and given the limited amount of motion that is needed to be removed, the goal of synthesizing physically-correct frames can be relaxed to seeking perceptual plausibility. Liu et al. propose a *content-preserving warping* thats attempts to meet two objectives: displace every tracked feature coordinate to its regularized reprojected location and, at the same time, minimize the warping distortion at content-rich regions.

In a more recent work, Liu et al. [2011] avoid the computational cost and the various sensitives of 3D models by directly operating on the 2D point trajectories. This is done by restricting the interpretation and synthesis of the trajectories to a low-dimensional linear subspace, according to a previous result in computer vision [Irani 2002]. This assumption holds for static 3D points being projected by a moving camera at short time intervals. This work also uses the content-preserving warps to generate the output frames by displacing every tracked point to its smoothed position in the frame. On top of handling parallax effects, this approach can cope with effects due to camera zoom, in-camera stabilization and rolling shutter. In our work we propose a mid-level scene modeling that uses epipolar geometry to handle 3D scenes with moving objects.

Image-Based View Synthesis. Here we mention another relevant line of work that deals with synthesizing physically-correct novel views without knowing or recovering explicit 3D scene geometry. Laveau and Faugeras [1994] show that accurate novel view synthesis does not necessitate full 3D point or camera recovery, rather it can be carried out under a setting they call ‘weakly-calibrated’ that relies only on epipolar relations. The *fundamental matrix* governs these relations and matches points in one image to their corresponding epipolar lines[1] in a second image of the same scene (both taken by general uncalibrated perspective cameras). This matrix can be estimated from very few point correspondences [Hartley and Zis-

[1]The epipolar line is the projection of an optical ray that passes through a point in one image, as seen in the second image, see [Hartley and Zisserman 2000].

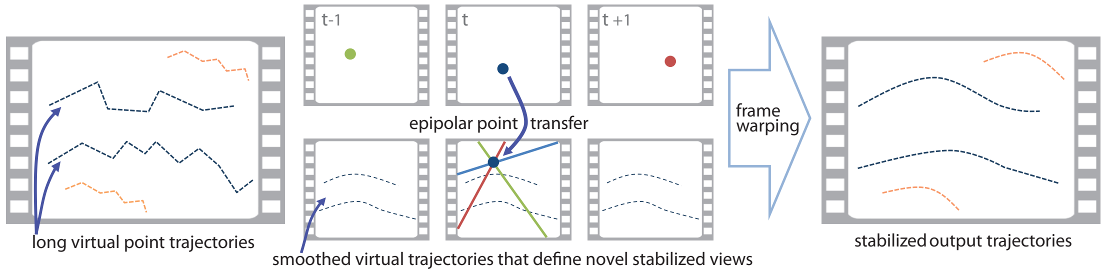
t-1
t
t+1
epipolar point transfer
frame warping
long virtual point trajectories
smoothed virtual trajectories that define novel stabilized views
stabilized output trajectories

Fig. 1. A schematic overview of the new video stabilization algorithm. Given a set of tracked point trajectories, we generate a few long virtual trajectories which we smooth using large filters. The smoothed paths are used to model the motion of a stabilized camera, expressed by fundamental matrices. Stabilized point coordinates are then obtained by intersecting the epipolar lines of corresponding points in the stabilized views. Finally, these point coordinates are used to create stabilized frames by warping the input frames.

serman 2000]. Laveau and Faugeras generate novel views by defining fundamental matrices that relate the input images to the new view using three user-specified corresponding points. The new image is synthesized by mapping corresponding pixels from the input views to the intersection point of their epipolar lines in the new image. Werner et al. [1995] address visibility issues of the interpolated points in this image-based framework.

Our purposed image stabilization method also operates at this level of scene modeling and does not explicitly estimate any 3D shape of the scene. Faugeras et al. [1993] show that this model can also be used for computing denser correspondences by restricting the search to epipolar lines in rectified image coordinates. Avidan and Shashua [1997] propose to use a trilinear epipolar relation, previously derived in [Shashua 1995], for synthesizing novel views. This formulation, as well, does not recover 3D structure and allows specifying the virtual camera intuitively. Moreover, it is robust to the singular configuration where the positions of three cameras are collinear.

Seitz and Dyer [1995] describe and analyze linear view interpolation methods that operate in rectified coordinates. They show that such interpolations produce physically-valid views under a general affine viewing model. In their interpolation algorithm, they also compute dense correspondences and exploit a monotonicity property that reduces the search into matching uniform intervals along scanlines. Finally, the *unstructured lumigraph* method of Buehler et al. [2001] assumes a partial knowledge of the plenoptic function and constructs novel frames by blending rays from existing ones.

3. NEW METHOD

The scene model used for video stabilization dictates, according to its level of complexity, to what scenes the method is applicable and its robustness to ambiguities and inaccuracies in the tracked data. As we discussed in Section 2, 2D methods are efficient and robust to noise, yet they are too limited in terms of the scene geometry and camera shake they can handle. 3D methods, on the other hand, can cope with more general scenes yet involve heavier and more delicate computations that break down when scene complexity drops. One alternative, discussed by Liu et al. [2009], is to switch on-line between models, based on the scene complexity [Torr et al. 1999]. Liu et al. [2011] propose a mid-level scene model that does not use explicit 3D geometry but relies on the approximation that corresponding point coordinates lie in a low-dimensional linear subspace. We suggest to use a projective reconstruction of the scene which is also an intermediate level of modeling between 2D and full 3D scene models. This is a more monolithic alternative compared to model-switching and it still provides the non-approximate physically-correctness that full 3D scene model provides. We proceed by giving a brief description of this model and then explain how we use it for video stabilization.

Scene reconstruction up to a projective transformation is known as projective reconstruction [Hartley and Zisserman 2000]. This model accounts for intrinsic geometric relations between views of general uncalibrated cameras. These relations are encapsulated by the *fundamental matrix* $\mathcal{F}$ which is a 3-by-3 rank-two matrix that relates a point in one view to its epipolar line in a second view. More formally, if q is a point (expressed in homogeneous coordinates) in one image, then $l = \mathcal{F}q$ is its corresponding epipolar line in the image of the second view. Therefore, any point q' that belongs to l, i.e., $l^\top q' = 0$, obeys $q'^\top \mathcal{F} q = 0$. Similarly to Faugeras et al. [1993] who discuss general novel view synthesis, we show how this level of scene modeling is sufficient for modeling stabilized views and projecting jittered points in the input video to these views. In fact, we show that this model allows us to match most of the capabilities 3D stabilization methods have and, at the same time, since it does not explicitly estimate 3D quantities, such as point coordinates and camera pose and inner-parameters, it suffers from less ambiguous configurations and involves lighter computations.

3.1 Overview

We start with an overview of our proposed video stabilization technique and in the subsequent sections we explain and discuss at greater depth the different components we mention here. Similarly to existing techniques, we start by tracking an initial set of feature points along the video sequence using standard 2D KLT point tracking algorithm [Shi and Tomasi 1994]. We denote the 2D coordinates of the tracked points by p_t^i where the super-index i specifies the point and the sub-index t the frame. We use these point correspondences to compute the fundamental matrices $F^{s,t}$ between every frame s and its close-in-time frames t. These matrices are estimated using the standard eight-point algorithm and RANSAC estimation [Hartley 1997], under the assumption that the majority of the points p_t^i belongs to a static background. To avoid overfitting these matrices to a particular region in the frame, we use the following stratification; we divide each frame into blocks, ranging between 16-by-16 to 32-by-32 pixels depending on the video resolution, and limit the number of points we take from each block into the RANSAC estimation. We use these matrices for several purposes: constructing the novel stabilized camera views, reprojecting the tracked points in time, and for computing additional point correspondences.

Views of a stabilized camera motion are computed from a set of virtual points trajectories v_t which we do not track but generate using the *epipolar point transfer* [Faugeras et al. 1993; Hartley and Zisserman 2000]. As we detail in Section 3.2, this is done by intersecting the epipolar lines casted by corresponding points in past frames in future frames. If these points were viewed by a stabilized cameras, their 2D trajectories in the video would be smoother. Therefore, we filter these trajectories in time by convolving their coordinates (independently) with a Gaussian kernel, and denote the result by $\tilde{v}_t^i$. In general we denote smoothed quantities with tilde. We then use the correspondence between the jittered v_s^i and smoothed $\tilde{v}_t^i$ coordinates to compute another set of fundamental matrices $\tilde{F}^{s,t}$ that relate the jittered views of the input frames and the output stabilized views. Finally, we use these matrices to transfer every tracked point p_t^i in the jittered input frames to its new coordinates in the stabilized frame. This pipeline is illustrated in Figure 1.

The epipolar transfer is limited to static points, therefore in order to handle non-static scenes we derive, in Section 3.3, a novel time-view point reprojection. This procedure estimates the projections of points on moving objects in nearby frames, as if their motion was *frozen in time*. Since these new correspondences do belong to static points in 3D, the epipolar point transfer can be used. This allows us to compute the stabilized locations of smoothly moving objects in the same way we stabilize static background points. Another issue that we address in our work is the non-uniform distribution of the tracked points p_t^i across the frame. We use $F^{s,t}$ and the epipolar point transfer to compute, rather than search for, additional correspondences (on top of p_t^i). This allows us to find matches at regions where the initial 2D KLT failed to find suitable feature points due the aperture problem, which is greatly reduced by the epipolar constraints. As we explain in Section 3.4, we perform this search in a stratified manner that achieves more uniform frame coverage.

3.2 Static Scene Stabilization

In this section we explain in more detail how we use the epipolar point transfer in practice and how we compute the stabilized views.

Epipolar Point Transfer. As mentioned earlier, the epipolar relations can be used for transferring points between different views. Since we use this mechanism extensively and extend it to non-stationary scenes in Section 3.3, we review here the epipolar transfer of Faugeras et al. [1993]. Assume q^t are projections of a static 3D point in different views t, e.g., camera views of different frames in our context. The epipolar lines casted by the points q^t in a novel view are given by $l_t = F^{t,\text{novel}} q_t$ where $F^{t,\text{novel}}$ are the fundamental matrices relating the different views t to the novel view, e.g., a stabilized or future frame in our application. Since these epipolar lines share the same 3D point, they intersect at a *single* point q in the novel view. This point is given by $l_t \times l_{t'}$ for every $t \neq t'$ and it is the projection of the 3D point onto the novel view. Hence, based solely on epipolar relations, one can transfer two or more corresponding points to new views without extracting their 3D locations or any camera information. We use this mechanism for computing long virtual trajectories that are needed for modeling the stabilized views, map tracked jittery points to their new coordinates in the stabilized views and for finding additional corresponding points in the input video. Figure 2 illustrates this epipolar point transfer.

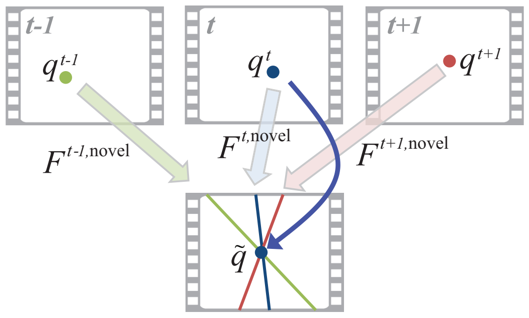

Fig. 2. Epipolar point transfer. Corresponding points at different views (frames) cast epipolar lines that intersect at a single point in a novel (stabilized) view. This point corresponds to the projection of the 3D point onto the novel view.

While two corresponding points are sufficient for using this transfer, noise and tracking and modeling errors undermine the accuracy of the transferred point location. In order to obtain a more accurate mapping, we use correspondences from more than two views that provide many possible intersections and use robust averaging to compute a more reliable estimate. Degenerate camera configurations may lead to nearly parallel epipolar lines whose intersection is less accurate. Therefore, we discard intersections between lines whose difference in orientation falls below 1.5 degrees[1]. Points that are close to the epipole[2] are algebraically close to the kernel of the fundamental matrix and hence the vector norm of the respective epipolar lines is small, indicating they are more sensitive to noise. Therefore, we discard lines with norm smaller than 10^{-3} (in normalized pixel coordinates)[1] and use the average of the remaining intersection points as the transferred point. We declare that the corresponding points are unreliable for transfer when the distance between the median and average intersection points are more than 5 pixels apart (for 640-by-480 videos) since it indicates tracking error. In such cases, the use of this trajectory is terminated.

Stabilized Views Construction. In order to define stabilized camera views in the epipolar geometry, we have to define the fundamental matrices that relate the jittered input camera views and the stabilized camera views. This task is not equivalent to smoothing the fundamental matrices $F^{s,t}$ computed between the jittered views, since we need matrices that relate jittered camera motion to smooth motion, rather than between different views of a smoothed camera motion. We achieve this by computing trajectories of virtual points v_t^i using the epipolar transfer described above. We initiate these virtual points using coordinates taken from the tracked points, i.e., $v_t^i = p_t^i$ for their first five frames (which provide ten possible intersection points). We do not need a large number of virtual trajectories and select about a hundred trajectories that are spread evenly across the frame. We select these points by dividing the frames into bins and picking a single tracked point from every bin (assuming one can be found). Note that v_t^i are expected to be jittery since they are transferred according to the input camera motion, via $F^{s,t}$. In fact, v_t^i coincides with p_t^i as long as the latter is not occluded by an-

[1]This value was found empirically and used for producing all our output videos.

[2]The epipole is the image in one view of the camera center of the other view, see [Hartley and Zisserman 2000].

other object in the scene, in which case v_t^i will last longer. In order to define stabilized camera views, we estimate how these trajectories appear in such a camera motion by simply smoothing them in time,

$$\tilde{v}_t^i = (g * v^i)_t \qquad (1)$$

using a Gaussian blurring kernel $g_t = e^{-t^2/2\sigma^2}$, and use $\sigma = 50$. This smoothing is computed for the horizontal and vertical coordinates independently. These trajectories are computed as long as they remain relevant for the shot and drop them once they exit the frame. For this purpose we increase the frame size by adding a margin of 20% its size in each direction. Note that using small windows in time (ten frames) we construct long virtual trajectories incrementally. Hence, we do not rely on the KLT tracking to find long trajectories that are needed for strong stabilization, i.e., convolving with large Gaussians in (1).

We use these smoothed trajectories to compute another set of fundamental matrices $\tilde{F}^{s,t}$ that relate points in jittered input views s and the epipolar lines these points cast in the novel stabilized views of close-in-time frames t. Here again we use $|t - s| \leq 5$ to obtain multiple point intersections when we later use the point transfer. These matrices are computed with v_s^i and $\tilde{v}_t^i$ at the corresponding points in the eight-point algorithm (used, as before, with RANSAC estimation). We use these matrices with the epipolar point transfer described above to compute the mapping between p_t^i and their stabilized locations $\tilde{p}_i^t$ which provide us the constraints needed for warping the input frames to produce the output frames, as we explain below. In our implementation, we apply additional fine smoothing to each of the stabilized trajectories $\tilde{p}_i^t$ using a Gaussian kernel with $\sigma = 6$ before we use them. This is done to smooth-out gentle high-frequency jitters that might remain due to tracking inaccuracies and errors in the stabilization process.

Note that the trajectories $\tilde{v}_s^i$ computed in this process are smoothed independently, and in the 2D frame plane. Hence these coordinates do not necessarily correspond to a projection of geometrically-valid 3D points locations on any realizable camera configuration. However, these smoothed points locations are not the final stabilized point locations, rather they are used to define new views by defining the fundamental matrices $\tilde{F}^{s,t}$. Since every rank-two matrix defines a geometrically-valid view, the matrices $\tilde{F}^{s,t}$ model view of a physically-realizable camera. This camera is moving smoothly in space since it is optimized (via the eight-point algorithm) to relate the jittered points v_t^i to the smoothly moving points $\tilde{v}_t^i$.

Frame Warping. As we explain below, similarly to 3D stabilization methods our point transfer is physically-correct. However, occlusions in the scene prevents us from producing proper novel views and therefore we follow the approach of Liu et al. [2009] and synthesize the stabilized frames by warping the frames in a content-preserving manner. This warping attempts to displace every input point p_i^t to its stabilized location $\tilde{p}_i^t$ while minimizing the distortion at content-rich regions. In this method each frame is warped individually such that pixels from different frames do not intermix. Thus, corresponding pixels in different frames are not required to have the same color value. This property allows handling non-static scenes containing moving objects. Algorithm 1 summarizes the steps of our stabilization algorithm.

Discussion. The point transfer described here corresponds to a physically-correct novel view, i.e., there exists a perspective camera that contains the very same new point coordinates $\tilde{p}_t^i$. Hence, the fact that the scene is reconstructed only up to a projective transfor-

Algorithm 1: Epipolar Video Stabilization.

```
track feature points in the movie using standard KLT tracking ;
for every frame $t$ do
    compute fundamental matrices $F^{s,t}$ for $|s - t| \leq 10$
end
optional: add trajectories using epipolar tracking (Section 3.4);
choose $n$ trajectories uniformly spread across the first frame ;
for $i = 1..n$ do
    set $v_t^i = p_t^i$ for $t = 1..5$ ;
    compute $v_t^i$ for $t = 6..end$ using epipolar transfer using
    $F^{s,t}$ and $v_s^i$, with $t - 10 \leq s < t$ ;
end
smooth virtual trajectories by $\tilde{v}_t^i = (g * v^i)_t$ ;
for every frame $t$ do
    use $v_s^i$ and $\tilde{v}_t^i$ to compute fundamental matrices $\tilde{F}^{s,t}$ for
    $|s - t| \leq 5$ ;
end
for every trajectory $i$ do
    if trajectory belongs to a dynamic object, calculate
    time-view reprojection (Section 3.3) ;
    for every frame $t$ do
        use $\tilde{F}^{s,t}$ and $p_i^s$ to compute $\tilde{p}_i^t$ using epipolar point
        transfer for $|s - t| \leq 5$ ;
    end
end
Use correspondence between original trajectories $p_i^t$ and
stabilized trajectories $\tilde{p}_i^t$ as input for frame warping ;
```

mation, does not undermine its ability to produce physically-correct views as 3D stabilization methods do. However, since this approach requires only the estimation of fundamental matrices, it is also valid for scenarios where the camera movement is translation-free (e.g., in far scenes) or when the scene is approximately planar.

In these cases the fundamental matrices $F^{s,t}$ can be factored to the products $F^{s,t} = S^{s,t}H^{s,t}$ where $S^{s,t}$ are arbitrary skew-symmetric matrices and $H^{s,t}$ are the homography matrices relating the points between the two frames. Multiplying a homography by a skew-symmetric matrix maps the points into lines that pass through an arbitrary epipole, see [Hartley and Zisserman 2000, Section 11.9.3]. Applying the point transfer using such fundamental matrices is equivalent to mapping points using the appropriate homography regardless of the skew-symmetric matrices. While the arbitrary fluctuations in the epipole location do not undermine the point transfer, cases where the matrices $S^{s,t}$ are identical will not allow the transfer (corresponding epipolar lines will be identical). In practice this does not pose a problem as we use more than two views and robust averaging to perform the point transfer. Hence, in cases where the scene does not contain parallax effects our algorithm naturally boils-down to the proper 2D stabilization needed for such time segments.

Furthermore, camera zooming while shooting the video or using the in-camera stabilization does not interfere with our scene modeling. In fact, any changes in the internal parameters of the camera that retain its perspective nature do not undermine the ability to model the scene using fundamental matrices as well as apply the point transfer. We test such scenarios in Videos 5-8.

We should note that our approach is not entirely free from degeneracies; in cases where the frames differ by a pure linear motion in the camera projection plane, the epipolar lines will all be parallel

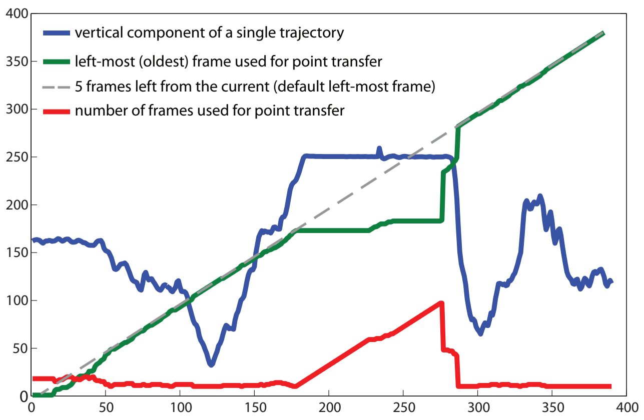

Fig. 3. Dynamic window of frames used for the point transfer. Blue plot shows the vertical component of a single trajectory, shown to indicate whether the camera is moving or not. Normally we use eleven frames for the point transfer yet when the number of valid intersections decrease (due to camera fixation in this example) the oldest frame used does not advance in time (green plot) and the number of frames used increases (red plot). Dashed gray plot shows a normal progression where eleven frames are used at every frame.

and hence the point transfer would not be possible. Another degenerate scenario is when the camera motion is restricted to an arbitrary plane in which case points that lie in this plane will also produce parallel lines. It is virtually impossible that such motions will be obtained by a jittery hand-held camera, besides the case where the camera is held still. Nevertheless, such scenarios can be easily avoided by considering more frames, that do not correspond to this degenerate camera motion, when computing the point transfer. This can be implemented in various different ways, for example, we simply extend the time window (which normally contain eleven frames, since we use $|s-t|\leq 5$) until enough intersections are left in the point transfer after discarding the degenerate intersections. The result of this strategy is shown in Figure 3 and in Video 15.

3.3 Time-View Reprojection of non-Static Scenes

When a tracked point corresponds to a moving object in 3D space, the point transfer we described above ceases to be valid. Indeed, points of moving objects are usually discarded by stabilization methods [Liu et al. 2009; Liu et al. 2011] and these regions of the frames receive their stabilization from other points that belong to the static surrounding. When this distance is large or when the depth difference between the object and its background is not negligible, the object will not receive an adequate stabilization.

We propose a novel time-view point reprojection that exploits the smoothness of the motion trajectories real-world objects typically follow. Consider a point p^s, defined in *view and time* of frame s, and the epipolar lines it casts on the *view* of frame t still *at time* s, i.e., $l^s = F^{s,t}p^s$. Assuming this is the projection of a moving point, we cannot not expect p^t to lie on this epipolar line. Nevertheless, we can consider the location $q^{s,t}$ of this point at *time* s in the *view* of frame t. In other words, we consider how this point is seen at time s by a stationary camera coinciding with the camera of frame t. Now, for any s the projections $q^{s,t}$ must lie on the the epipolar line l^s, i.e., $l^\top q^{s,t}=0$, as these are correspondences of a point that is 'frozen' in time s. While this does not provide us with enough geometrical constraints to determine $q^{s,t}$, we estimate $q^{s,t}$ based on its dynamical behavior. The points $q^{s,t}$ as a functions of s, is a projected trajectory of a moving point in 3D space viewed by a fixed camera. Assuming this object moves smoothly, or more accurately, that its motion results from the minimal acting forces needed to create it, then according to the law of dynamics its velocity vectors $u^{s,t} = q^{s+1,t} - q^{s,t}$ must change as little as possible. Figure 4 illustrates this dynamical scenario.

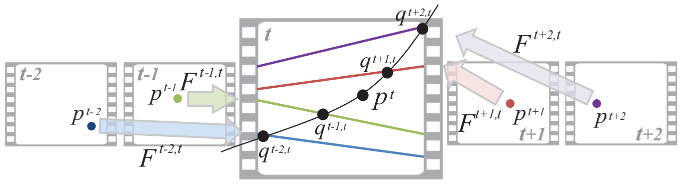

Fig. 4. Dynamic point reprojection. An inertial moving point in 3D space projects a smooth trajectory onto a fixed view (of frame t). This constraint is used to find its locations along each of the epipolar lines it casts from views of different times.

This additional assumption makes it sufficient to pose enough constraints over $q^{s,t}$ which we now derive. Denote by D_u the following time-differentiation matrix

$$\begin{pmatrix} -1 & 1 & 0 & .. \\ 0 & -1 & 1 & 0 \\ \vdots & & \ddots & \vdots \\ .. & 0 & -1 & 1 \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ q^{s-1,t} \\ q^{s,t} \\ q^{s+1,t} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ u^{s-1} \\ u^{s} \\ u^{s+1} \\ \vdots \end{pmatrix}. \tag{2}$$

This matrix has one row less than columns as every derivative is computed from two instants of time. Denote by D_a the matrix D_u with its last row and column removed. When operating on the velocities vector $u^{\vec{s}}$, this time-differentiation matrix produces the acceleration values $a^{\vec{s}} = D_a u^{\vec{s}}$, where $\vec{s}$ is a vector containing the considered times s. Therefore, in order to obtain a trajectory resulting from minimal forces such that every point $q^{s,t}$ lies on its corresponding epipolar line l^s, we have to solve the following constrained system

$$\min_{q^{\vec{s},t}} \|Lq^{\vec{s},t}\|^2 \quad \text{s.t.} \quad \forall s \neq t \;\; (l^s)^\top q^{s,t} = 0, \quad \text{and} \quad q^{t,t} = p^t, \tag{3}$$

where L is the second-order derivative matrix $D_a D_u$. Applying Lagrange multipliers rule reduces this problem to solving the following linear system

$$\begin{pmatrix} L^\top L & C^\top \\ C & 0 \end{pmatrix} \cdot \begin{pmatrix} q^{\vec{s},t} \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix}, \tag{4}$$

where C and b are the matrix and vector expressing the linear constraints in (3). Note that these equations are formulated and solved independently over the horizontal and vertical coordinates of each point $q^{s,t}$ and the homogeneous (last) coordinates are set to one.

In practice, errors prevent the points from falling exactly on their respected epipolar lines and hence we relax the hard constraints in (3) to soft constraints and minimize $\|Lq^{\vec{s},t}\|^2 + \alpha\|Cq^{\vec{s},t} - b\|^2$ with $\alpha = 10^{-4}$. Upon differentiation, this optimization is solved by

$$\left(L^\top L + \alpha C^\top C\right) \cdot q^{\vec{s},t} = \alpha C^\top \cdot b. \tag{5}$$

We compute this for $|s-t| \leq 5$ which is what we normally use for transferring points and therefore the system above consists of a 22-by-22 matrix operating on the coordinates of $q^{s,t}$.

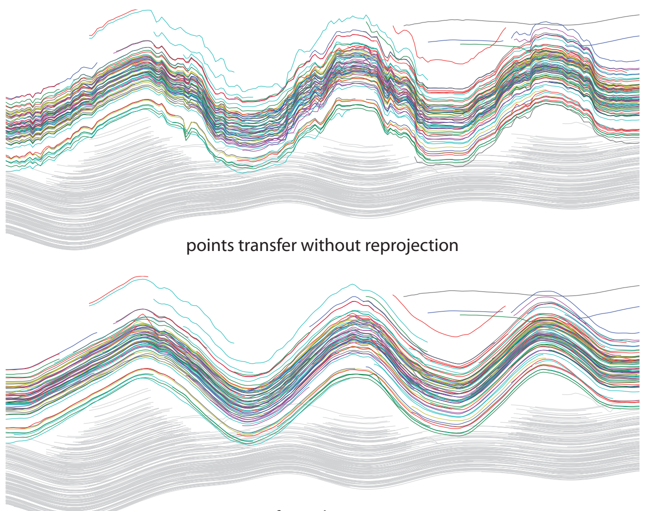

Fig. 5. Results obtained when running the epipolar point transfer with and without our time-view reprojection step. This scene contains points that belong to a static background (gray trajectories) and to a moving person in front (colored trajectories). When transferring the points without our reprojection step, there are noticeable high-frequency jitters in the output trajectories of the moving person, while the ones belonging to the static background are smooth. Applying our reprojection yields smooth trajectories for both the moving and static scene objects.

The resulting $q^{s,t}$, with s and t exchanged in their order and considered as a function of t, provide us with correspondences between the views of frames t of a *static* point (frozen at time s). This allows us to transfer the point p_t ($\approx q^{t,t}$) using the correspondences $q^{t,\vec{s}}$ to its stabilized coordinates using $\tilde{F}^{s,t}$ by applying the same epipolar transfer we described above. Note that this formulation is equally valid for stationary points, where the cost functional in (5) is expected to vanish, and hence this procedure acts as an extension to the static point transfer mechanism. However, we avoid solving these systems for static points and apply it only for points that deviate from the epipolar constraints. More specifically, we apply this time-view reprojection to points that $\min\{|p_{t-5}^{\top}F^{t,t-5}|, |p_{t+5}^{\top}F^{t,t+5}p_t|\} > 3$. The fact that this formulation applies for both dynamic and static points makes the method insensitive to misclassification of points as moving points.

Figure 5 shows the effect of using our reprojection procedure when stabilizing video containgin a moving person (taken from Video 13). Videos 12-14 show real video sequences where there are relatively large moving objects whose distance from the background is non-negligible. In such cases the stabilization needed for these objects cannot be inferred from the background, as done in [Liu et al. 2009; Liu et al. 2011]. The use of the time-view reprojection achieves proper stabilization in such scenes. In Video S1 we show a synthetic scene where the objects are moving in a linear motion that changes its direction abruptly (i.e., the motion is not differentiable). Nonetheless, our reprojection procedure did not result in a noticeable failure and conclude that the requirement of smooth motion is rather tolerant. We use this test to provide a quantitative comparison between the stabilized trajectories of the static and moving objects with and without our time-view reprojection (TVR). This is based on measuring their smoothness by computing the mean absolute time-derivative of their coordinates. As indicated in the inset above, the use of the time-view reprojection significantly improves the smoothness of the trajectories of the moving objects.

	TVR	w/o TVR
static	0.1141	0.1173
moving	0.2098	2.4173

3.4 Uniform Correspondence Maps

KLT-based feature point trackers [Shi and Tomasi 1994] identify every point with a very small window of pixels surrounding it. Matching points between frames is done by measuring the similarity of these windows. Hence, these methods require that the windows contain a sufficient amount of variation that will allow identifying movement at any direction. Straight edges, for example, do not provide this information and suffer from what is known as the *aperture problem*; an upward movement of a vertical edge does not change its surrounding pixels. Therefore, these trackers restrict their operation to unique points such as corner edges. Typically few hundreds such points are tracked at every frame in videos of NTSC resolution. While this is enough for recovering the fundamental matrices $F^{s,t}$ and $\tilde{F}^{s,t}$, in many cases these points are not spread uniformly across the frame and hence do not capture the geometry of the scene at uncovered regions.

Faugeras et al. [1993] and Seitz and Dyer [1995] use the epipolar constraints to search for matching points in rectified image coordinates. This one-dimensional search allows reducing the aperture ambiguity since now every edge which is not parallel to the epipolar line can be matched. We follow this idea combined with the epipolar point transfer in order to track more points that are evenly spread across the frame. To achieve this we divide each frame into blocks, ranging between 16-by-16 to 32-by-32 pixels depending on the video resolution, and search for one or more suitable points for tracking. We choose the points with the maximal Laplacian magnitude $|\Delta I^t(p^t)|$ in each block, where I^t denotes the frame pixel values. We then search for a matching point along the epipolar line p^t casts in the next frame, i.e., $F^{t,t+1}p^t$. We measure the matching error between the points using the L_2 norm over windows of 7-by-7 pixels around them. We normalize the pixels in each window to obtain variance of unit intensity. This search is performed along a segment of about 50 pixels, computed by intersecting the epipolar line in frame $t+1$ with a 50-by-50 pixels square around p^t. We allow a sub-pixel correction to the best match found by running a few iterations of KLT search, constrained to the epipolar line. In order to avoid ambiguities along the line, we discard points whose closest-match score does not fall below half the score of the next best match. Besides storing the matching point coordinates p^{t+1} we also store the matching error e for future reference. Starting with more than one point per block makes it more likely that we will find at least one correspondence within it. We initiate this search only in blocks that contain no tracked points (either ones found previously by this tracking mechanism or by the initial KLT tracking).

Once a correspondence is found, we *do not* continue tracking it using line searches, and instead we use the point transfer to predict its location. More specifically, we compute the epipolar lines casted at the next frame $t+1$ based on the corresponding points found in past frames (last five if available), i.e., $l^s = F^{s,t+1}p^s$ with $s = t-4, .., t$. Intersecting these lines with one another gives possible locations for p^{t+1}. As we did before, we discard intersections between lines whose difference in orientation is small or lines obtained with low-magnitude vectors. We then test which of the remaining points gives the lowest matching error (computed the same

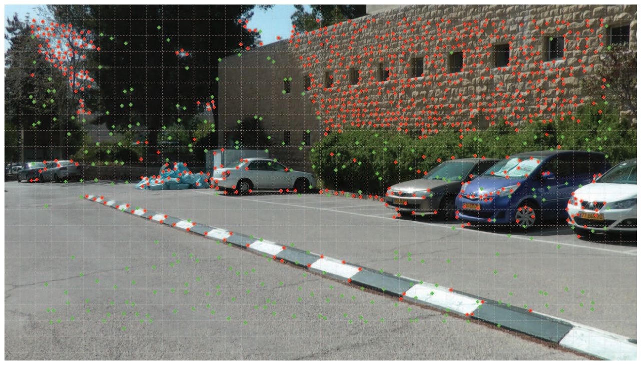

Fig. 6. Frame (taken from Video 16) showing the initial tracked points (red), the additional points obtained by the point transfer (green), and the 32-by-32 pixel blocks used (dashed white).

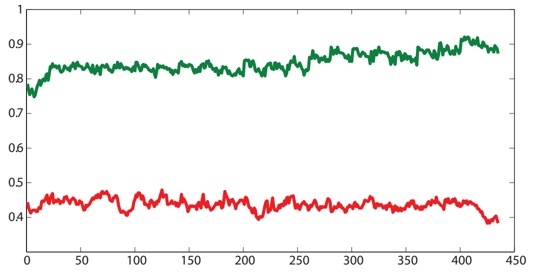

Fig. 7. Plots show the proportion of blocks containing a tracked points in every frame of Video 16. The initial KLT trajectories cover less than half the block (red) whereas our search achieves a coverage higher than 80% of the blocks (green).

way as above). We correct this match by running a few iterations of a 2D KLT search, limited to ± 1 pixel. Finally, we stop tracking a point (and discard its matches in the last 10 frames) if the matching error between its window and the corresponding one in the previous frame is higher than 20 times its initial matching error e. We use this relative criteria in order to normalize the matching quality of different windows. We do not use trajectories that last less than twenty frames.

Figure 6 shows the points tracked by the initial 2D KLT tracking and the ones obtained using the procedure described here. A better coverage is achieved at regions where no adequate feature points were found for the KLT tracking (e.g., the road). Video 16 demonstrates the more accurate frame warping achieved by these additional points. In Figure 7 we show the increased proportion of blocks containing a tracked point in this sequence.

In Videos 23b and 24b we use the initial KLT tracking only for computing the fundemental matrices $F^{s,t}$, and do not use these trajectories in the stabilizing stage. Instead, we used our epipolar transfer-based tracking to generate the all the trajectories used for stabilization. The resulting videos confirm the accuracy of this new procedure. In addition, we measure the distance between the coordinates of the trajecories found by standard KLT tracking and our predicted coodinates (before runing the sub-pixel 2D KLT correction step) in a frame-by-frame basis. The average distance found in two videos that we run this experiment on (Videos 23 and 24) fall below 0.2 pixels. In Figure 8 we provide the histrogram of these distances.

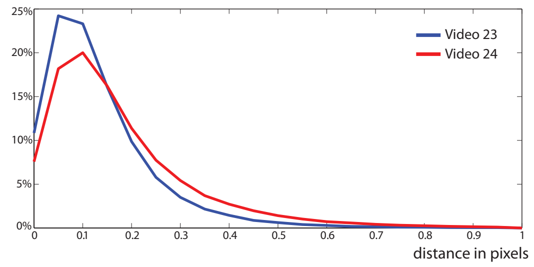

Fig. 8. Histograms of the distances between the points tracked by the KLT and the ones predicted by our epiplar transfer scheme.

4. RESULTS

We used the Voodoo camera tracker[3] software set to perform standard 2D KLT search for computing our initial point correspondences. The rest of the method was implemented in Matlab with very little optimization and executed on i7 Intel 3.07GHz machine. The running-time per frame on 1280-by-720 pixels videos are as following. The initial point tracking runs in about 5fps and our epipolar-based point tracking runs in about 4fps. Computing the fundamental matrices (both F and $\tilde{F}$) as well as solving the 22-by-22 linear systems when necessary (using LU decomposition) is done in 2fps.

Comparison. Videos 1 and 2 show non-planar scenes captured by a jittery camera. Our method handles these scenes well and matches the results of the 3D video stabilization method of Liu et al. [2009]. In Videos 3, 6 and 8-11 we compare our method with the subspace video stabilization of Liu et al. [2011]. These videos confirm our ability to match their results on scenes that: lack parallax, camera auto-stabilization turned on, camera zooming and small rolling shutter effect. Video 12 contains a moving person that covers a large portion of the frame and whose distance from the background results in a different response to the camera jitter. In such cases the stabilization needed for the object cannot be inferred from the background, as done in [Liu et al. 2009; Liu et al. 2011]. Our time-view point reprojection is designed to cope with such scenarios and manages to provide stabilization with considerably less artifacts.

Evaluation. We tested our methods on scenes with non-trivial 3D geometry (Videos 1 and 2) as well as on scenes with planar geometry or little camera translation. The lack of parallax does not introduce any difficulty to our method. This is demonstrated in shown in Videos 3 and 4 which contain distant objects and mostly rotational camera motion. Similarly, camera zooming and in-camera stabilization produce views that are valid in the epipolar geometry and hence our method performs well on such shots. Videos 5-6 were acquired while the camera zoom was changing (hence affecting the intrinsic camera parameters) and in Videos 7 and 8 the in-camera stabilization system was active and affected the camera optics.

Videos 12-14 demonstrate and compare the time-view point reprojection on scenes containing dominant moving objects. As we explained earlier, in these scenes the background's stabilization is inadequate for the foreground moving object. These tests show that the reprojection step is a critical component for treating such trajectories. Video S1 shows a synthetic scene where the objects are moving in a linear motion and changes their direction abruptly. Our

[3]`http://www.digilab.uni-hannover.de`

Table I. Summary of example videos accompanying this paper.

Video #	Description
1-2	Comparison with 3D stabilization of Liu et al. [2009]
3-4	3D stabilization failure cases: Lack of parallax
4-5	3D stabilization failure cases: Camera zoom
7-8	3D stabilization failure cases: in-camera stabilization
9-11	Comparison with subspace stabilization of Liu et al. [2011]
12	Subspace stabilization failure case: dynamic scene
13,14,S1	Time-View reprojection of scenes containing dominant moving objects.
15	Video containing degenerate segments
16	Enhanced tracking using the epipolar point transfer
17	Path fitting
18	Insufficient trajectories
19	Camera occlusions cut trajectories
20	Tracking failure due to motion blur
21	Tracking failure due to excessive camera shake
22	Warp failure due to lack of smooth regions
23,24	Stabilization using only epipolar transfer-based tracking
25-42	Additional assortment of videos

reprojection procedure, which relies on the smoothness of the motion, appears to be robust to these changes. As we described in Section 3.2 segments with no camera motion do not allow to recover the stabilizing fundamental matrices. We cope with such situations using a dynamic window containing non-degenerate frames when extracting these matrices. Video 15 shows how this strategy allows us to cope with a scenario where the camera is placed on a table and picked up a few seconds later. In Video 16 we show the benefit of extracting additional trajectories using our epipolar point transfer tracking scheme. Videos 23 and 24 show the stabilization that results by replacing all the trajectories obtained using KLT with our epipolar point transfer tracking. Finally, in Videos 25-42 we test our method on an unselected collection of videos taken by a moving person or driving car.

All our example videos, at their full resolution, are available at: `http://www.cs.huji.ac.il/~raananf/projects/stab/`.

5. CONCLUSIONS

We presented a new video stabilization method that uses epipolar geometry to model the input and synthesize the output videos. This level of modeling falls, in terms of the amount of recovered scene data, between the models 2D and 3D methods use. Despite its lower complexity, this model is sufficient for generating physically-correct stabilize views at the tracked points locations and can handle non-trivial 3D scene geometry. The advantages of this approach are its robustness to ambiguities, including planar scenes and degenerate camera motion, and that it is more computationally efficient than existing full 3D approaches. Furthermore, we presented an extension of the epipolar point transfer to for handling non-stationary points. This approach exploits the fact that the motion of 3D objects is smooth and assumes they experience minimal forces. Unlike existing solutions, this principled homogeneous approach allows us to stabilize trajectories of moving objects instead of applying them an irrelevant stabilization from their background. Finally, we described a scheme for increasing the number of tracked points efficiently using the epipolar point transfer that exploits the epipolar relation to reduce the aperture problem and achieve a more uniform distribution of tracked points across the frame.

Limitations. Video stabilization strongly depends on the number and accuracy of the tracked feature points. Scenes with little texture, excessively strong camera jitters, and cases where the static background cannot not detected, based on majority voting, will not allow successful stabilization, see Videos 20 and 21. Cases where successive frames share a small overlap (due to rapid camera motion) undermine the ability to track corresponding points as well as require stronger frame cropping when producing rectangular output frames. Strong occlusions and highly non-Lambertian surfaces are another source for failure, see Videos 18 and 19. Our method uses the epipolar geometry which relates point with lines between images of different views. Therefore, unlike the method of Liu et al. [2011], it cannot cope with strong rolling shutter effects that distort these relations. The content-preserving warp of Liu et al. [2009] that we use relies on having smooth regions with little content that will absorb the warping distortion. Texture-rich frames do not have such regions and some deformation can be visible, as seen in Video 22. Another limitation of our method, compared to 3D methods, is the inability to provide explicit 3D camera motion planning. However, as shown in [Liu et al. 2011] low-order polynomials and spline camera motions can be approximated by replacing the trajectory smoothing in (1) with fitting these models to the jittered trajectories and using them instead of the smoothed trajectories $\tilde{v}_t^i$. In Video 17 we show the result achieved by fitting a quadratic polynomial that mimics a quadratic camera path.

As future work, we intend to investigate the possibility of using the tri-focal tensor [Shashua 1995] to express the epipolar relations instead of using fundamental matrices. This formulation is known to be more robust to degeneracies as well as allowing to specify more explicit novel views which could provide a better camera path planning. We also believe that developing point trackers that can recover from occlusions will greatly benefit video stabilization.

ACKNOWLEDGMENTS

The authors would like to thank Mike Werman and Shmuel Peleg for their useful suggestions. We would like to thank Feng Liu for providing us various test video clips. Finally, we like to thank the Israel Science Foundation founded by the Israel Academy of Sciences and Humanities as well as Microsoft New Faculty Fellowship Program for supporting this research.

REFERENCES

AVIDAN, S. AND SHASHUA, A. 1997. Novel view synthesis in tensor space. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*. CVPR '97. IEEE Computer Society, Washington, DC, USA, 1034–.

BHAT, P., ZITNICK, C. L., SNAVELY, N., AGARWALA, A., AGRAWALA, M., CURLESS, B., COHEN, M., AND KANG, S. B. 2007. Using photographs to enhance videos of a static scene. In *Rendering Techniques 2007 (Proceedings Eurographics Symposium on Rendering)*, J. Kautz and S. Pattanaik, Eds. Eurographics, 327–338.

BUEHLER, C., BOSSE, M., AND MCMILLAN, L. 2001. Non-metric image-based rendering for video stabilization. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on 2*, 609.

BUEHLER, C., BOSSE, M., MCMILLAN, L., GORTLER, S., AND COHEN, M. 2001. Unstructured lumigraph rendering. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. SIGGRAPH '01. ACM, New York, NY, USA, 425–432.

FAUGERAS, O., HOTZ, B., MATHIEU, H., VIÉVILLE, T., ZHANG, Z., FUA, P., THÉRON, E., MOLL, L., BERRY, G., VUILLEMIN, J., BERTIN, P., AND PROY, C. 1993. Real time correlation based stereo: algorithm implementations and applications. *INRIA Technical Report* RR-2013.

GLEICHER, M. L. AND LIU, F. 2008. Re-cinematography: Improving the camerawork of casual video. *ACM Trans. Multimedia Comput. Commun. Appl. 5*, 2:1–2:28.

HARTLEY, R. 1997. In defense of the eight-point algorithm. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 19*, 6 (June), 580 –593.

HARTLEY, R. AND ZISSERMAN, A. 2000. *Multiple view geometry in computer vision*. Cambridge University Press, New York, NY, USA.

IRANI, M. 2002. Multi-frame correspondence estimation using subspace constraints. *Int. J. Comput. Vision 48*, 173–194.

IRANI, M., ROUSSO, B., AND PELEG, S. 1994. Recovery of ego-motion using image stabilization. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) Proceedings*, 454–460.

LAVEAU, S. AND FAUGERAS, O. 1994. 3-d scene representation as a collection of images. In *Proceedings of the 12th IAPR International Conference on Pattern Recognition*. Vol. 1. 689–691.

LIU, F., GLEICHER, M., JIN, H., AND AGARWALA, A. 2009. Content-preserving warps for 3d video stabilization. In *ACM SIGGRAPH 2009 papers*. SIGGRAPH '09. ACM, New York, NY, USA, 44:1–44:9.

LIU, F., GLEICHER, M., WANG, J., JIN, H., AND AGARWALA, A. 2011. Subspace video stabilization. *ACM Transactions on Graphics 30*, 4:1–4:10.

LUKAC, R. 2008. *Single-Sensor Imaging: Methods and Applications for Digital Cameras*. CRC Press, Inc., Boca Raton, FL, USA.

MATSUSHITA, Y., OFEK, E., GE, W., TANG, X., AND SHUM, H.-Y. 2006. Full-frame video stabilization with motion inpainting. *IEEE Trans. Pattern Anal. Mach. Intell. 28*, 1150–1163.

SEITZ, S. AND DYER, C. 1995. Physically-valid view synthesis by image interplation. In *Proceedings of the IEEE Workshop on Representation of Visual Scenes*. VSR '95. IEEE Computer Society, Washington, DC, USA, 18–.

SHASHUA, A. 1995. Algebraic functions for recognition. *IEEE Trans. Pattern Anal. Mach. Intell. 17*, 779–789.

SHI, J. AND TOMASI, C. 1994. Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*. 593 – 600.

TORR, P. H. S., FITZGIBBON, A. W., AND ZISSERMAN, A. 1999. The problem of degeneracy in structure and motion recovery from uncalibrated image sequences. *Int. J. Comput. Vision 32*, 27–44.

TRIGGS, B., MCLAUCHLAN, P. F., HARTLEY, R. I., AND FITZGIBBON, A. W. 2000. Bundle adjustment - a modern synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*. ICCV '99. Springer-Verlag, London, UK, 298–372.

WERNER, T., HERSCH, R. D., AND HLAVAC, V. 1995. Rendering real-world objects using view interpolation. In *Proceedings of the Fifth International Conference on Computer Vision*. ICCV '95. IEEE Computer Society, Washington, DC, USA, 957–.

Received May 2011; accepted January 2012