Quotienting Impertinent Camera Kinematics for 3D Video Stabilization

Thomas W. Mitchel Christian Wüelker Jin Seob Kim
tmitchel@jhu.edu christian.wuelker@jhu.edu jkim115@jhu.edu
Sipu Ruan Gregory S. Chirikjian
ruansp@jhu.edu gchirik1@jhu.edu
Laboratory for Computational Sensing and Robotics
Johns Hopkins University

Abstract

With the recent advent of methods that allow for real-time computation, dense 3D flows have become a viable basis for fast camera motion estimation. Most importantly, dense flows are more robust than the sparse feature matching techniques used by existing 3D stabilization methods, able to better handle large camera displacements and occlusions similar to those often found in consumer videos. Here we introduce a framework for 3D video stabilization that relies on dense scene flow alone. The foundation of this approach is a novel camera motion model that allows for real-world camera poses to be recovered directly from 3D motion fields. Moreover, this model can be extended to describe certain types of non-rigid artifacts that are commonly found in videos, such as those resulting from zooms. This framework gives rise to several robust regimes that produce high-quality stabilization of the kind achieved by prior full 3D methods while avoiding the fragility typically present in feature-based approaches. As an added benefit, our framework is fast: the simplicity of our motion model and efficient flow calculations combine to enable stabilization at a high frame rate.

1. Introduction

Video stabilization algorithms depend on some measure of the original camera motion to remove unwanted artifacts and produce smooth videos. Most existing 2D and 3D stabilization techniques recover inter-frame camera motion by either tracking feature trajectories \[8, 15, 16\] or matching image features between frames \[24, 19, 21\]. Image features are brittle and break down in videos with fast camera motions or scene occlusions, both of which are common in consumer videos. In contrast, dense flows (such as 2D optical flow or 3D scene flow) provide more robust descriptions of scene motion and remain accurate in challenging videos where feature-based approaches can fail \[27, 2, 32, 35\]. As seen in Figure 1, flows can contain information about both rigid camera motion and non-rigid artifacts, such as those produced by zooms or distortion effects. Some recent 2D stabilization methods \[22, 18, 9\] have incorporated the use of flows into camera motion estimation. However, each still rely on image features in some capacity and thus retain the weakness inherent in feature-based approaches.

Our contribution is a technique for full 3D video stabili-

Figure 1: ‘Signatures’ of different camera motion artifacts in flows as seen from the image plane.
lization that relies only on dense scene flow and as a result, is able to take full advantage of the benefits provided by dense flows. Our approach is based on a novel motion model which allows for the recovery of 3D camera motions from dense scene flow alone. Camera motion is represented by global transformations in a Lie group, and we exploit a powerful geometric relationship between 3D motion fields and twists to recover camera poses from scene flow via a closed-form expression. This model gives rise to a natural formulation of the optimal camera path, and together they form a robust framework for 3D video stabilization which we call Quotienting Impertinent Camera Kinematics or QuICK for short.

QuICK models camera motion by the action of 3D transformations on the real-world camera pose. These transformations belong to the Lie groups SE(3) (rigid-body motion), SIM(3) (similarities), SA(3) (special affine transformations – analogous to homographies and translations), and GA+(3) (general affine transformations). Fast scene flow [12] is used to produce the 3D motion field from which the transformations describing inter-frame motion are extracted. The transformations are then used to compute an optimal camera path that minimizes inter-frame camera motion subject to desired end-constraints. This framework results in robust video stabilization regimes which achieve high-quality stabilization while avoiding many of the pitfalls commonly encountered by other 3D methods. QuICK is quick. The use of fast scene flow compliments QuICK’s closed-form solution to the camera motion, enabling stabilization at a high frame-rate. QuICK is exceptionally flexible, able to handle challenging camera motions such as rapid rotations and zooms in videos where other 3D stabilization methods typically fail.

2. Related work

Existing video stabilization methods can be categorized as 2D, 2.5D, and 3D regimes. Most 2D methods rely exclusively on matching image features [28] between consecutive frames. Matsushita et al. [24] represented the camera path through a series of homographies. Grundmann et al. [8] constructed cinematographic [6] L1 optimal camera paths piecewise, utilizing a combination of similarity and homography transformations. Liu et al. [21] introduced a nonlinear image warping motion model used by many recent 2D methods. Recently, a series of 2D stabilization methods have been developed that utilize 2D motion fields to recover camera motions. Liu et al. [22] developed the first such method in which feature tracking and dense optical flows were used to produce smoothed motion fields. Liu et al. [18] and Guo et al. [9] then produced similar regimes for monocular and stereoscopic stabilization, respectively. Liu et al. also developed a ‘codingflow’ regime, using a 2D motion field comprised of smoothed motion vectors employed in video coding. With the exception of ‘codingflow’, all of these methods still rely on image features to some degree.

Virtually all stabilization methods that incorporate 3D information use feature-based approaches. Long feature tracks represent a middle ground between 2D and 3D scene descriptions. Liu and Gleich [16] exploited the partial 3D information contained in smoothed 2D feature trajectories to develop the first 2.5D method. Subsequently, both Goldstein and Fattal [7] and Wang et al. [36] introduced methods based on improving the length and quality of tracked feature trajectories. Recently, Liu et al. [20] developed a hybrid 2.5D regime which combines warping and homography motion models using both 2D and 3D features.

3D stabilization involves the use of 3D scene information to estimate the change in the 3D pose of the camera between frames. Liu et al. [15] developed the first full 3D stabilization regime by smoothing trajectories of 3D points recovered using structure-from-motion (SFM) [11] and guiding ‘content preserving’ image warps with their projections. Other successful 3D methods rely on supplemental cameras to avoid the need for brittle scene reconstruction. Smith et al. [29] and Liu et al. [19] used light field and depth cameras, respectively, to project matched features into 3D and recover the 3D camera pose. However, by relying on sparse feature matches these regimes remain sensitive to occlusions, dynamic scene content, and fast camera motions under which feature matching can break down.

In contrast to prior 3D stabilization methods, QuICK can use dense scene flow alone to recover 3D camera poses. The result is a framework that produces high-quality stabilization without the accompanying fragility present in prior 3D regimes.

3. The QuICK motion model

QuICK is built upon a novel method for recovering 3D camera motion directly from scene flow. To motivate our approach, we begin by discussing the advantages of dense flows over image features in the context of video stabilization. Then we introduce the QuICK motion model and describe how 3D camera motion can be estimated from dense flows. Though we demonstrate our method with RGB-D videos, we are convinced that our motion model can be applied to any type of videos where 3D scene structure can be estimated and dense scene flow computed, such as stereoscopic or light field video.

3.1. Dense scene flow for 3D motion estimation

In contrast to sparse flows, which are calculated by matching features, dense flows rely on a variational framework to give a complete description of the motion field covering an image. The most important consequence of this approach is that dense flows are much more robust than feature matches and by extension, sparse flows. They remain
Figure 2: The geometry of the induced twist $\xi_{ij}$ at the camera frame corresponding to the point $x_{ij}$ with velocity $v_{ij}$.

3.2. Rigid-body motion: SE(3)

3D camera motion can be described by global parametric transformations belonging to a matrix Lie group. We begin by considering camera motion consisting of 3D translations and rotations, i.e., rigid-body motions. These motions belong to the Lie group SE(3), which is the semi-direct product of $\mathbb{R}^3$ with the group of orientation-preserving 3D rotations, SO(3), i.e., $\text{SE}(3) = \mathbb{R}^3 \rtimes \text{SO}(3)$.

Suppose we are given a sequence of intensity images, their corresponding depth maps, and that the camera is in motion throughout the video. Assuming the camera calibration is known, all pixels in a given frame with valid depth information can be mapped to their real-world 3D points as seen from the camera frame, namely $x \in \mathbb{R}^3$. Computing the scene flow gives the 3D translational velocity of each point $v \in \mathbb{R}^3$. Now, let us consider an individual point, say the one corresponding to the $ij$th pixel, $x_{ij}$, and imagine it to be rigidly fixed to the origin of the camera frame as shown in Figure 2. Since the two are fixed, the origin must undergo the same translational displacement. However, the displacement of the point also produces a corresponding angular velocity at the origin, given by

$$\omega_{ij} = \frac{x_{ij} \times v_{ij}}{x_{ij}^T x_{ij}}.$$  \hfill (1)

Together, they form a twist at the camera frame,

$$\xi_{ij} = \begin{bmatrix} v_{ij} \\ \omega_{ij} \end{bmatrix} \in \mathbb{R}^6,$$  \hfill (2)

which we call the induced twist.

Applying this process to all of the 3D points, the twist describing the motion of the camera between frames, $\xi \in \mathbb{R}^6$, minimizes

$$f(\xi) = \frac{1}{2} \sum_{ij} \| \xi - \xi_{ij} \|^2,$$  \hfill (3)

where the sum is over all pixel indices with valid depth. Letting $D$ be the number of all such pixel indices, $\xi$ is simply the average of all induced twists,

$$\xi = \frac{1}{D} \sum_{ij} \xi_{ij} = \begin{bmatrix} v \\ \omega \end{bmatrix},$$  \hfill (4)

which we call the camera twist.

Regardless of frame rate, we can interpret the camera twist $\xi$ as the displacement of the camera between frames, such that $v$ is the translation of the camera and $\omega$ is the axis of rotation in $\text{so}(3)$, the Lie algebra of $\text{SO}(3)$. Together, they are the twist coordinates that parameterize the transformation $g \in \text{SE}(3)$ representing the camera motion. Under this observation, we break from the classical structure of $\text{SE}(3)$ and consider translational and rotational motion to be decoupled. This allows for the camera motion to be expressed by the homogeneous transformation [5]

$$g(\xi) = \begin{bmatrix} \exp(\omega) & v \\ 0^T & 1 \end{bmatrix},$$  \hfill (5)

where $\hat{\cdot}$ is defined such that for any $a, b \in \mathbb{R}^3$, $\hat{a}b = a \times b$. We find that only basic filtering of the flow by excluding points with unnaturally large velocities is required for robust estimation of the camera motion.

A visualization of the recovery of different camera motions in shown in Figure 3. It is interesting to note that due to the position of the scene in front of the camera, pure translations and rotations are not always recovered as such—demonstrated by the small rotations and translations recovered in 3a and 3b, respectively. Scene symmetry exists only about and along the forward axis and corresponding motions can be recovered exactly as seen in 3c. Regardless, we find this has a negligible effect on stabilization.
3.3. Similarities: SIM(3)

Zooms are a common feature in consumer videos. QuICK’s framework can be extended to model them by considering camera motion in the Lie group SIM(3), consisting of rigid motions and isotropic scaling. The induced twist corresponding to an arbitrary point $x_{ij}$ with displacement $v_{ij}$ is of the form

$$\xi_{ij} = \begin{bmatrix} v_{ij} \\ \omega_{ij} \\ a_{ij} \end{bmatrix},$$

the key insight being that the scaling component is given by

$$a_{ij} = \frac{x_{ij}^T v_{ij}}{x_{ij}^T x_{ij}}.$$  

Again, the camera twist, $\xi = \begin{bmatrix} v^T, \omega^T, a \end{bmatrix}^T$, is simply the average of all induced twists. Decoupling the translational component and exponentiating the rotational and scaling components gives the homogeneous transformation describing the camera motion,

$$g(\xi) = \begin{bmatrix} e^a \exp (\bar{\omega}) & v \\ 0^T \\ 1 \end{bmatrix}.$$  

To our knowledge, our SIM(3) implementation of QuICK is the first full 3D video stabilization regime to explicitly consider the effects of zooms.

3.4. Affine transformations: SA(3) and GA⁺(3)

Modeling camera motion using higher dimensional Lie groups requires a general formulation of the QuICK motion model. In particular, the form of the induced twist corresponding to an arbitrary point depends on chosen basis for the Lie algebra.

Let $G$ be an affine matrix Lie group (i.e. the bottom row of each matrix element is filled with zeros except for the last element, which is 1) and suppose the matrices $E_1, \ldots, E_n$ form a basis for its corresponding Lie algebra, $\mathfrak{g}$. Then, the induced twist at the camera frame corresponding to a point $x_{ij}$ with velocity $v_{ij}$ is of the form

$$\xi_{ij} = \Lambda_{ij}^{-T} \begin{bmatrix} v_{ij} \\ 0 \end{bmatrix} \in \mathbb{R}^n,$$

where $\Lambda_{ij} \in \mathbb{R}^{n \times n}$ is defined with respect to $x_{ij}$ such that

$$\Lambda_{ij} = \left[ (h_{ij} E_1 h_{ij}^{-1})^\vee, \ldots, (h_{ij} E_n h_{ij}^{-1})^\vee \right],$$

where $\Lambda_{ij}$ is a particular form of the matrix representation of the Lie group’s adjoint operator [3], namely $\Lambda_{ij} = \text{Ad}(h_{ij})$. To this point, the form of the induced twist in (9) is similar to the transformation of a pure force between coordinate frames in the theory of wrenches [26]. However, normalizing the translational component $h_{ij}$ by dividing by the square of its magnitude allows us to preserve velocities instead of forces and torques.

The 3D special affine group, $\text{SA}(3)$, is the semi-direct product of $\mathbb{R}^3$ with $\text{SL}(3)$, the group of all unit determinant matrices in $\mathbb{R}^{3 \times 3}$. $\text{SL}(3)$ is of interest to the problem of video stabilization as it can be roughly thought of as the group of homographies with unit determinant [23]. Therefore, we believe that $\text{SA}(3)$ is worth examining in the context of 3D stabilization.
The Lie algebra of \(\text{SL}(3)\) is \(\mathfrak{sl}(3)\), which consists of all zero-trace matrices in \(\mathbb{R}^{3\times 3}\). A possible basis for \(\mathfrak{sl}(3)\) is

\[
\begin{align*}
\tilde{E}_1 &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\
\tilde{E}_2 &= \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\
\tilde{E}_3 &= \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \\
\tilde{E}_4 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \\
\tilde{E}_5 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\
\tilde{E}_6 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\
\tilde{E}_7 &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \\
\tilde{E}_8 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\
\tilde{E}_9 &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\
\tilde{E}_{10} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\
\tilde{E}_{11} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \\
\tilde{E}_{12} &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.
\end{align*}
\]  

where \(\cdot\) denotes the absolute value of the matrix elements. We can choose a basis for \(\mathfrak{sl}(3)\) based on \(\mathfrak{sl}(3)\), namely

\[
E_k = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad 1 \leq k \leq 3
\]

We can then define a basis for \(\mathfrak{sl}(3)\), namely

\[
E_k = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad 4 \leq k \leq 11.
\]

Expressed relative to this basis, the induced twist corresponding to an arbitrary point \(x_{ij}\) with displacement \(v_{ij}\) is of the form

\[
\xi_{ij} = \begin{bmatrix} v_{ij} \\ \omega_{ij} \\ \frac{1}{x_{ij}^T x_{ij}} |x_{ij}^T v_{ij}| \end{bmatrix} \in \mathbb{R}^{11}.
\]

All groups we have considered thus far are subgroups of the positive general affine group, \(G^+(3)\), which is the semi-direct product of \(\mathbb{R}^3\) with \(\text{GL}^+(3)\), the group of all \(3 \times 3\) matrices with positive determinant. As the most general 3D affine Lie group, it has the potential to best approximate 3D rigid camera motions and non-rigid artifacts via a single, global transformation. A possible basis for the Lie algebra of \(G^+(3)\), \(\mathfrak{g}^+(3)\) is given by the matrices

\[
\begin{bmatrix} E_k \end{bmatrix}_{mn} = \begin{cases} 1, & \text{if } 3(n-1) + m = k \leq 9, \\ 0, & \text{otherwise} \end{cases}
\]

We can then define a basis for \(\mathfrak{g}^+(3)\), the Lie algebra of \(G^+(3)\), as in (16). In this basis, the form of the induced twist at the camera frame is

\[
\xi_{ij} = \begin{bmatrix} v_{ij} \\ \omega_{ij} \\ \frac{1}{x_{ij}^T x_{ij}} x_{ij}^T \varepsilon_1 v_{ij} \\ \frac{1}{x_{ij}^T x_{ij}} x_{ij}^T \varepsilon_2 v_{ij} \\ \frac{1}{x_{ij}^T x_{ij}} x_{ij}^T \varepsilon_3 v_{ij} \end{bmatrix} \in \mathbb{R}^{12}.
\]

The standard matrix exponential does not provide a surjective mapping from either the \(\mathfrak{sl}(3)\) or \(\mathfrak{gl}^+(3)\) to their corresponding Lie groups [37]. However, the mapping

\[
\phi(U) = \exp \left( U^T \right) \exp \left( U - U^T \right)
\]

is surjective for both \(\mathfrak{sl}(3)\) and \(\mathfrak{gl}^+(3)\) [1, 34]. Assuming the general form of the camera twist to be \(\xi = [v^T, u^T]^T\), the camera motion is

\[
g(\xi) = \begin{bmatrix} \phi(U) & v \\ 0^T & 1 \end{bmatrix},
\]

where \(U\) is the element in either \(\mathfrak{sl}(3)\) or \(\mathfrak{gl}^+(3)\) with twist coordinates \(u\).

4. Video stabilization

When depth information is known, the QuICK motion model can be applied to estimate the 3D inter-frame camera motion over the course of a video in terms of the chosen affine Lie group, \(G\). Taking \(H\) to be the Lie group such that \(G = \mathbb{R}^3 \rtimes H\), the result is the camera twist as a function of time,

\[
\xi(t) = \begin{bmatrix} v(t) \\ u(t) \end{bmatrix} \in \mathbb{R}^n,
\]

where \(u\) is a vector of Lie algebra twist coordinates corresponding to \(H\). As \(\xi\) measures relative displacement between frames, we integrate on the right to recover the estimated 3D camera path as seen from the initial camera pose, \(g_c(t) \in G\), e.g.

\[
g_c(0) = \mathbb{I}_4,
\]

\[
g_c(t_{i+1}) = g_c(t_i) g(\xi(t_i)).
\]

Conversely, \(g_c^{-1}(t)\) maps the camera back to its initial pose. When camera travels only a small distance or is intended to remain static, \(g_c^{-1}\) can be used to stabilize the video. However, the applications of this approach are limited.
4.1. Optimal camera paths

Let $\sigma$ be the mapping which takes the twist coordinates $u$ to their corresponding element in the Lie group $H$, such that $\sigma (u(t)) = A(t) \in H$. To formulate a robust description of an optimal camera path, we consider the twists

$$\zeta (t) = \left[ \begin{array}{c} -A^{-1}v \\
\sigma^{-1} (A^{-1}) \end{array} \right] \in \mathbb{R}^n. \quad (25)$$

If $G = \text{SE}(3)$ or $G = \text{SIM}(3)$, $\sigma^{-1}$ is computed via the matrix logarithm, i.e.

$$\sigma^{-1} (A^{-1}) = \log^\vee (A^{-1}) = -u. \quad (26)$$

However, if the group is $\text{SA}(3)$ or $\text{GA}^+(3)$, the inverse of (20) has no closed form expression and $\sigma^{-1}$ must be computed numerically using gradient descent [37]. The twists $\zeta (t)$ correspond to the inverses of the motions parameterized by the camera twist $\xi (t)$. With this in mind, we consider the functional

$$J [\zeta] = \frac{1}{2} \int_{t_0}^{t_f} ||\zeta(t) - \zeta(t)||^2_W dt$$

$$= \frac{1}{2} \int_{t_0}^{t_f} [\zeta^T W \zeta - 2b^T \zeta + c] dt, \quad (27)$$

where $W = W^T \in \mathbb{R}^{n \times n}$ and

$$b = W \zeta$$

$$c = \zeta^T W \zeta. \quad (28)$$

Subject to desired end-constraints, the twist that minimizes $J$, $\zeta^* (t)$, parameterizes the transformations that map the camera to the optimal path. Hence, QuiCK gets its name — imagining $S$ as the space on which videos temporally evolve, applying these transformations to the video reduces the ‘problem’ of viewing the video from $S$ to the quotient space, $G \backslash S$.

The variational minimization of (27) can be directly addressed by applying the coordinate-free generalization of the Euler-Lagrange equation known as the Euler-Poincaré equation [4], through which the differential equation describing the minimal solution can be recovered:

$$\ddot{\zeta} = W^{-1} \left[ b + \text{ad} (\zeta)^T [W \zeta - b] \right]. \quad (30)$$

The matrix $[\text{ad} (\zeta)]$ is defined with respect to the chosen Lie algebra basis for $g$, $\{E_i\}$, such that

$$[\text{ad} (\zeta)] = \left[ \left( [\zeta, E_1]\right)^\vee, \ldots, \left( [\zeta, E_n]\right)^\vee \right], \quad (31)$$

where for any $A, B \in \mathbb{R}^{n \times n}$, $[A, B] = AB - BA$. The matrix $W$ can be used to set the relative weights of different motions along the path, however, we do not explore this in detail and set $W = I_n$ in our implementations.

Videos are stabilized piecewise. Key-frames are selected and the video is split into segments bounded by adjacent key-frames. In our implementations, key-frames are set at 30 frame intervals or when large changes in the direction of the velocity occur along the estimated camera path. The formulation of (27) means that the chosen end-constraints for the path are in fact the transformations that map the estimated camera poses at the bounding key-frames to their desired, stabilized poses. If no corrections are desired at the key-frames, then the end-constraints are just the identity transformation. Since consecutive sequences share the same final and initial camera poses, the global camera path is smooth and continuous in the first derivative.

We use a shooting method similar to that proposed in [13] to solve (30) subject to the desired end-constraints. We find that this method easily generalizes to handle the affine Lie groups we consider here. An example of the solution process is shown in Figure 4. The differential equation is first integrated on the right with respect to an initial guess for the value of $\zeta_0 = \zeta (0)$. Applying these transformations to the estimated camera path produces a stabilized path, shown in dark red. An artificial trajectory is defined between the distal correction transformation produced by integrating the guess and the desired correction transformation given by the end-constraints. In Figure 3, this is visualized in terms of the camera path: the artificial trajectory, represented by the dashed line, is between the camera pose at the end of the initial path and the desired pose, highlighted in blue. The initial guess is then updated to track the trajectory [10] and the process is iterated until convergence. Figure 5 shows an example of an estimated camera trajectory (in red) and the corresponding stabilized path (in green).

The end result of this process is a series of stabilizing transformations, $g_s (t) \in G$, that map the camera to its stabilized path. We accept a greater degree of global distortion in exchange for increased local image quality in and around occluded regions in the depth map, and use ‘content-preserving’ warps [15] to synthesize stabilized frames. All pixels with valid depth form the control points. We use a
We run QuICK in C++ on a system with a 3.8 GHz CPU, 32GB of RAM, and an NVIDIA RTX 2080 GPU. Only scene flow calculations [12] are implemented on the GPU. We plan to publish our code online for public use. For a 30-frame RGB-D sequence at 640 × 480 resolution, our SE(3) implementation of QuICK takes approximately 121 ms to process a frame. Specifically, computing the scene flow, recovering the camera motion, and computing the optimal path take 16 ms, 0.3 ms, and 4.2 ms, respectively. The main bottleneck is the rendering of the stabilized frame which takes 103 ms. For a 320 × 240 resolution sequence, we achieve a processing speed of 35 ms per frame (29 fps).

Table 1 shows the per-frame processing speed of other stabilization methods with videos of similar resolution. All data are from published results with the exception of [19] which we measure using our own implementation. It is important to note that processing speed is dependent on the system on which a method is run. Generally, QuICK is significantly faster than existing 2.5D and 3D regimes and its processing speed is on par with that of successful 2D methods.

### 5.1. Quantitative evaluation and comparisons

To demonstrate the effectiveness of QuICK, we perform experiments using publicly available real and synthetic RGB-D sequences from the TUM RGB-D SLAM [30] and SceneNet RGB-D [25] datasets, respectively. We consider four different implementations of QuICK which model 3D camera motion using the Lie groups SE(3), SIM(3), SA(3), and GA⁺(3), respectively. To evaluate the quality of stabilized videos, we adopt three objective quantitative metrics [21]: cropping, distortion, and stability defined as described in [17]. We are not aware of any publicly available implementations of prior 3D stabilization regimes, so we compare QuICK to our own implementation of the full 3D ‘depth camera’ stabilization method [19]. Additionally, we compare against the ‘Warp Stabilizer’ effect in Adobe After Effects CC 16.0 which is based on the partial 3D ‘subspace’ stabilization method [16].

To perform comparisons, we selected 12 videos from the TUM RGB-D dataset and group them into three categories based on their content: (1) Simple, (2) Fast Camera Motion, and (3) Dynamic Objects. We then randomly select 20 different 30 frame subsequences from each video for
evaluation. Similarly, we selected 12 synthetic videos from the SceneNet RGB-D dataset and selected 20 subsequences from each. Videos from this dataset are sampled at approximately 1 fps and as a result, consist exclusively of fast camera motions, many of which we consider to be highly challenging. As such, we limited the length of the subsequences to 20 frames. We accounted for failure cases (which we defined as instances in which the stabilizing transformations map at least one input frame entirely out of view of the image plane) by setting the values of all metrics to zero for the subsequence. The average performance of each method across all subsequences from each dataset is shown in Figure 6 and Figure 7.

All four implementations of QuICK consistently produce high-quality stabilization. In particular, videos stabilized via QuICK suffer from less distortion and cropping than those stabilized using either the ‘depth camera’ method or Adobe’s ‘Warp Stabilizer’ while providing comparable stability. Moreover, QuICK performs significantly better than the ‘depth camera’ method on both real and synthetic videos containing fast camera motions. This is likely a result QuICK’s reliance on dense scene flow for motion estimation instead of the feature-based approach used by the ‘depth camera’ method. In particular, many of the synthetic sequences from the SceneNet dataset contain rapid rotations that result in too few or non-existent matches between adjacent frames, causing the ‘depth camera’ method to fail. In contrast, scene flow estimation remains intact in these instances, allowing QuICK to recover the camera motion. While scene flow estimation becomes less accurate under large displacements, we find that our formulation of the optimal camera path is very robust, enabling QuICK to well-stabilize challenging sequences as long as the magnitude of the estimated flow remains relatively consistent.

Our implementations of QuICK do not explicitly handle dynamic scene content. When large, near-range dynamic occlusions occur in the scene, such as when a person walks directly in front of the camera, the synthesized view tends to ‘follow’ the dynamic object instead of focusing on the obvious point of interest in the scene. To minimize this effect, a technique similar to the one presented in [22] could be implemented to preprocess the scene flow by detecting and removing sections of the flow corresponding to dynamic content.

6. Conclusion

Our pivotal contribution is QuICK — a framework for robust, full 3D video stabilization. In particular, QuICK is based on a novel camera motion model through which camera poses can be recovered directly from 3D flows. This model can be generalized to describe non-rigid video artifacts of the type often found in consumer videos, such as those produced by zooms.

7. Acknowledgements

This work was performed under National Science Foundation grant IIS-1619050 and Office of Naval Research Award N00014-17-1-2142.
References


