Cinematic-L1 Video Stabilization with a Log-Homography Model

Arwen Bradley, Jason Klivington, Joseph Triscari, Rudolph van der Merwe

Apple, Inc.

{arwen_bradley, klivington, jtriscari, vandermerwe}@apple.com

Abstract

We present a method for stabilizing handheld video that simulates the camera motions cinematographers achieve with equipment like tripods, dollies, and Steadicams. We formulate a constrained convex optimization problem minimizing the ℓ_1-norm of the first three derivatives of the stabilized motion. Our approach extends the work of Grundmann et al. [9] by solving with full homographies (rather than affinities) in order to correct perspective, preserving linearity by working in log-homography space. We also construct crop constraints that preserve field-of-view; model the problem as a quadratic (rather than linear) program to allow for an ℓ_2 term encouraging fidelity to the original trajectory; and add constraints and objectives to reduce distortion. Furthermore, we propose new methods for handling salient objects via both inclusion constraints and centering objectives. Finally, we describe a windowing strategy to approximate the solution in linear time and bounded memory. Our method is computationally efficient, running at 300fps on an iPhone XS, and yields high-quality results, as we demonstrate with a collection of stabilized videos, quantitative and qualitative comparisons to [9] and other methods, and an ablation study.

1. Introduction

In recent years, individuals have increasingly been shooting video with smartphones and other handheld devices. Unlike professional cinematographers, however, they usually do not carry stabilization equipment such as tripods, dollies, or Steadicams, and they may not have the time or training to shoot with cinematic intent. We want to stabilize these casually-shot, handheld videos to clean up the shaky camera motion into the smooth motions characteristic of professional cinematography to create a more aesthetically pleasing video that allows the viewer to better focus on the content. Further, we may wish to guide the stabilization based on salient objects, for example to try to keep them centered or at least within the stabilized frame.

Video stabilization methods can generally be categorized as 2D, 2.5D, or 3D. 3D methods model the full 3D scene
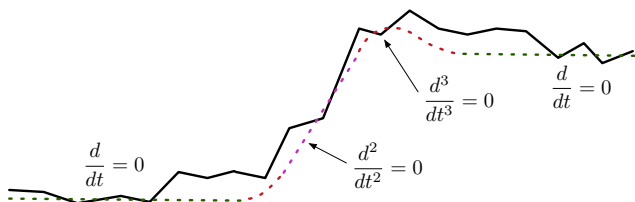

Figure 1: We seek a stabilized trajectory that mimics the camera motions of professionally-shot video, namely stationary, panning, and constant acceleration motions that might be shot with a tripod, dolly, or Steadicam. These motions correspond to zero first, second, and third derivatives.
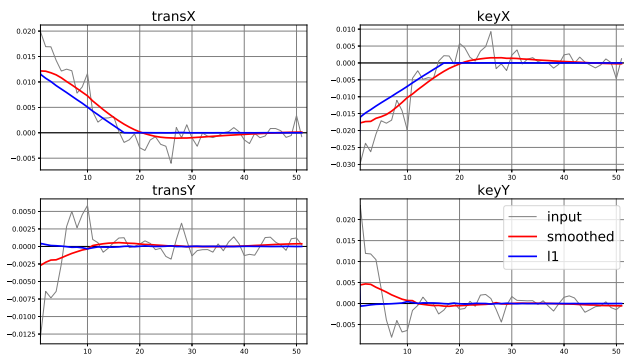

Figure 2: Comparison of Cinematic-L1 and Gaussian smoothing (corresponding to video *fountain-tripod-smooth-l1.mp4* in supplementary material). Each subplot shows the value over time of a single translation or keystone homography element (where homographies map between consecutive frames). Gaussian smoothing just removes high-frequency motion, while L1 stabilization yields a constant-velocity pan followed by a 'tripod' segment. (For corresponding derivative plots, see supplement B.)

structure, often via Structure from Motion (SFM) [13, 23], and stabilize using full 3D reprojections or intelligent approximations like Content Preserving Warps [15]. 3D methods are the most powerful, but tend to be slower (SFM performance is improving but still costly) and less robust than simpler methods. 2.5D methods seek to handle depth vari-

ation in the scene without full 3D scene reconstruction and tend to be faster than 3D: [16] exploit the fact that motion trajectories of rigid scenes tend to lie in low-dimension subspaces; [18] uses multiple bundled homographies to handle scene depth variations; [17, 19] perform spatial smoothing of an optical flow field and nonparametric mesh, respectively; and [7] use an epipolar geometry model.

Our approach belongs to the 2D class, which do not handle scene depth variation and can suffer from parallax, but have advantages of robustness and speed. In performance-critical contexts like capture-time video stabilization where inertial sensor (gyro/accel) data is available but pixel-based analysis is too costly, the scene structure cannot be determined and 2D models are the only choice. An important class of 2D methods focus on rolling-shutter correction [8, 10, 21]; such methods are complementary to ours, and can be run as a preprocessing step (supplement A.5).

The basic stages of video stabilization are 'analysis': determining frame-to-frame transforms that describe the original camera trajectory, and 'correction': computing transforms to apply to each frame plus a final crop rectangle to generate a stabilized video. 2D analysis generally relies on either pixel-based image registration techniques [2], or leveraging hardware motion sensor data [12]. There are many possible strategies for the correction phase. One is 'virtual tripod': mapping each frame back to a reference by chaining and inverting the analysis transforms; but this will fail for videos with too much motion (crop will be empty). Another is 'smoothing' by applying a fixed or variable kernel (Gaussian, for example) to the input trajectory to remove the high-frequency motion [10]; this is fairly robust, though it never achieves perfect tripod-stabilization or constant-velocity pan. Neither tripod nor fixed-kernel smoothing offer control over the amount of crop, or the possibility of guidance based on saliency.

Inspired by the proposal of Grundmann *et al.* in [9], we focus on a powerful, flexible 'L1' correction strategy. The goal is to create a stabilized trajectory consisting of stationary, constant-velocity pan, and constant-acceleration segments, so the video looks as though it had been shot with professional camera equipment (tripod, dolly, Steadicam, respectively). (This type of cinematography-inspired stabilization criteria was first proposed by [6].) [9] formulate the video stabilization problem as a linear program, use the ℓ_1-norm heuristic to encourage sparsity [22] in the derivatives of the smoothed trajectory (see figure 1), and add constraints for crop, distortion, and possibly salient points. The convex formulation is key to good performance, since it allows for the use of interior-point methods, which efficiently solve large, sparse, convex problems [3]. Subsequently, convex 'L1' type approaches have been explored by others: [5] create multiple clips focusing on different actors with a convex formulation involving saliency constraints

and an L1 penalty, [1] propose a windowing approach that allows for online processing, and [20] add an ℓ_2 term encouraging closeness to the input.

[9], as well as [1, 5, 20], formulate the optimization problem using an affine model for the analysis and correction transforms. Affinities can capture translation, rotation, scale, and shear, but cannot model perspective (keystone), which limits the quality of stabilization. The more general class of homographies accounts for perspective, and fully captures the projective relationship between two images of a plane [11]. Homographies are represented as 3x3 nonsingular *homogeneous* matrices (that is, only the ratio of the elements is significant, or put another way, homographies are equivalent up to normalization). Homographies are a powerful tool for stabilization, but the nonlinearity poses many technical difficulties in the convex problem formulation. Although homographies themselves are convex [3], compositions or other complex expressions involving homographies quickly become intractable. [9] refine the affine solution with homographies in a post-process 'residual motion suppression' step, which helps correct the residual keystone, but this is less principled than solving directly with homographies and can have undesirable side-effects like consuming significantly more crop.

1.1. Contributions

Our strategy is to follow [9] in formulating the video stabilization problem as a convex optimization problem, wherein we encourage sparsity in the first three derivatives of the stabilized motion by minimizing the ℓ_1-norm, subject to crop and other constraints. We build upon their approach by solving with homographies rather than affinities (by introducing a log-homography transform), modeling the problem as a quadratic rather than linear program so we can add ℓ_2 terms to discourage large excursions from the input trajectory, improving upon the crop constraints, addressing various distortion concerns, in particular some that arise from homographies as opposed to affinities, and proposing new ways to deal with salient objects with both constraints and centering objectives. Finally, we describe a windowing strategy to approximate the global L1 solution in linear time and bounded memory. Figure 2 compares L1 stabilization to Gaussian smoothing. Figure 3 shows L1 stabilization results for a longer, more complex, video.

- *Log-homography transform.* Using homographies rather than affinities allows for much more precise, high-quality stabilization, but the nonlinearity introduced by the perspective transform creates serious technical difficulties in the convex formulation. In order to handle homographies while preserving convexity, we introduce a log-homography transform based on Lie theory that allows us to express the problem in terms of linear and quadratic operations.

- *Crop constraints preserving field-of-view.* We develop crop constraints that not only ensure that the crop window samples valid pixels, but also that the transforms preserve field-of-view, via approximations in terms of area and sidelength. We show how to linearize these constraints using the Jacobian.
- *Fidelity objective term and variable stabilization strength.* We add an ℓ_2 objective term to encourage fidelity to the input path in order to respect intent, reduce distortion, and optionally provide granular control over the stabilization strength. (This is similar to the ℓ_2 term used in [20].)
- *Distortion mitigation.* Beyond the fidelity term, we also address distortion issues via constraints and objectives applied to the affine diagonal elements, off-diagonal elements, and homography keystone/translation ratios.
- *Saliency.* We propose two strategies for handling salient objects: constraints to keep salient objects from being cropped out of the stabilized frame, and centering objective terms to attempt to keep salient objects close to the center of the stabilized frame.
- *Windowed L1.* In order to handle arbitrary-length videos, we describe an overlapping-window strategy that runs in linear time and bounded memory and approximates the global solution given sufficient lookahead. A Markov property allows us to capture the complete history via a three-frame window initialization. (Both [1] and [20] describe windowing approaches for affine models, but neither exploits the Markov property: [1]'s CineConvex formulation adds constraints on overlapping segments rather than just 3 frames, and [20] compute independent solutions on overlapping windows and use a weighted average.)

2. L1 stabilization with log-homographies

As discussed in the introduction, video stabilization consists of two phases, 'analysis' and 'correction', where the analysis phase seeks to estimate homographies mapping consecutive frames of the input, and the correction phase determines correction transforms to be applied to each frame, along with an overall crop. The focus of this work is on the second phase, so we assume from now on that the analysis homographies are known constants, denoted by $\{F_t\}$. Let $\{P_t\}$ denote the homography corrections to be applied to a fixed centered *crop window* to stabilize the video. (Note that we would apply $\{P_t^{-1}\}$ to the original video frames, then crop, to render the stabilized video.)

$$
\begin{aligned}
F_t \quad & \text{maps input } t \to t-1 \\
\tilde{F}_t = F_0 \dots F_t \quad & \text{maps input } t \to 0 \\
S_t = \tilde{F}_t P_t \quad & \text{maps stabilized } t \to 0
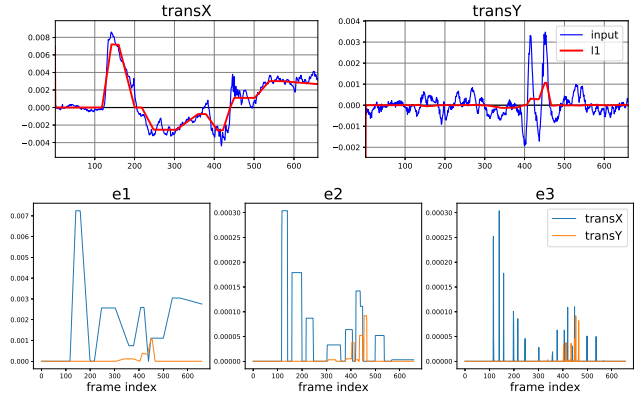\end{aligned} \tag{1}
$$

Figure 3: L1 stabilization results for supplementary video *river-l1.mp4*. **Top**: Translation elements of homographies over time (where homographies map between consecutive frames). **Bottom**: Sparse absolute values of first (e^1), second (e^2), and third (e^3) derivatives of L1 stabilized path (per equation 4) (only translation elements are shown; see supplement B for plots of all elements.)

We formulate the stabilization problem as a convex optimization problem in the variables P_t, with constraints and objectives that achieve the following goals:

- encourage sparse first, second, and third derivatives of the stabilized motion S_t by minimizing the ℓ_1-norm (as proposed by [9]) – these zero derivatives correspond to tripod, panning, and smoothly accelerating motions.
- encourage fidelity of the stabilized to the original camera path, by keeping corrections small in an ℓ_2 sense.
- respect crop constraints by ensuring both valid pixels in the crop window and preservation of field-of-view.
- minimize distortion via objective terms and constraints that control the relationships between elements of the correction homographies.
- optionally, keep 'salient' objects either approximately centered or in frame.

In order to support homographies (as opposed to affinities) while maintaining convexity, we linearize the problem by transforming to the vector space of log-homographies. In this section, we describe the log-homography transformation (with a brief introduction to the Lie theory that supports it) and show how to express or approximate each of the items above in the resulting framework.

2.1. Log homographies

The basic motivation for the log-homography transform is to turn complex nonlinear operations (composition and inverses of homographies) into simple linear operations (addition and negation of log-homographies, respectively). These properties are analogous to those of the standard
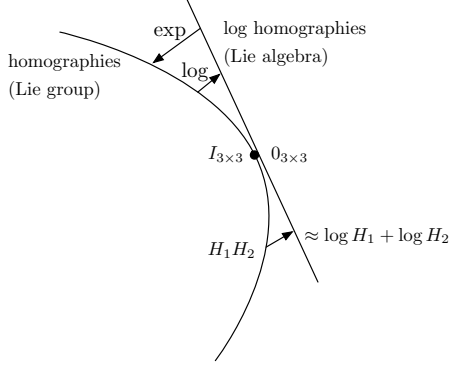
Figure 4: The space of homographies is a Lie group, and its tangent space at the identity forms a Lie algebra (a vector space). The matrix exponential maps from the Lie algebra to the Lie group; its inverse is the matrix logarithm. We will formulate the L1 stabilization problem in terms of log-homographies (elements of the Lie algebra).

logarithm on real numbers: $\log(xy) = \log(x) + \log(y)$, $\log(x^{-1}) = -\log(x)$. Lie theory allows us to formalize this notion. We will briefly outline the key points relevant to our application here; for more detail see for example [14]. The group of homographies is isomorphic to $SL(3, \mathbb{R})$, the group of real invertible square matrices with determinant 1, since homography matrices differing only by a scale factor are equivalent and we can normalize to determinant 1. $SL(n, \mathbb{R})$ is a Lie group, which is both a group and a manifold. As a manifold, $SL(n, \mathbb{R})$ is a complex surface where addition is undefined (think of the surface of a sphere), but by working in the tangent space we can recover the notion of addition. In general, the tangent space at the identity of a Lie group forms a Lie algebra (see figure 4), which unlike the group itself is a vector space: in particular, we have addition and scalar multiplication. The Lie algebra of $SL(n, \mathbb{R})$, denoted $\mathfrak{sl}(3, \Re)$, consists of 3×3 matrices with zero trace; the identity of $SL(n, \mathbb{R})$ is the usual identity matrix I and the identity of $\mathfrak{sl}(3, \Re)$ is the zero matrix. The exponential map from the Lie algebra to the manifold is given by the matrix exponential $\exp(X) = \sum_{k=0}^{\infty} \frac{X^k}{k!}$. (This exponential is related to but different from the Riemannian exponential used for example in [25] for geodesic interpolation between key frames.) In a neighborhood of the identity, the exponential map has an inverse: the matrix logarithm. That is, for a homography H:

$$H = \exp(h) \in SL(3, \mathbb{R}), \quad h = \log(H) \in \mathfrak{sl}(3, \Re). \quad (2)$$

There are a variety of methods for computing exp and log available in many scientific computing libraries. Our implementation perturbs the matrix until it is diagonalizable and exploits the fact that if $\exp(A) = U e^D U^{-1}$ with $D = \mathrm{diag}(d)$ then $e^D = \mathrm{diag}(e^d)$. Other implementations

use the Pade approximation to the infinite series. Two important properties for our application are that, to first-order in a neighborhood of the identity, we have:

$$\log(H^{-1}) \approx -h;$$
$$\log(H) \approx \log(H_1) + \log(H_2). \quad (3)$$

Both properties follow from the fact that group multiplication in a Lie group is reflected to first order in the vector space structure of its Lie algebra [14]. Neither can hold exactly since homographies do not commute in general, so these are approximations, but nevertheless useful. The linearity of log-homographies enables us to formulate L1 stabilization as a convex problem, which would be difficult or impossible with homographies.

2.2. L1 stabilization with log-homographies

Now that we have the log-homography transformation tools, we show how to formulate the stabilization problem. With the notation of equation 1, let $f_t = \log F_t$ denote the input log-homographies. The problem variables are the correction log-homographies $p_t = \log P_t \in \mathbb{R}^9$, which we stack into a column vector $p \in R^{9n}$ where n is the number of frames. We also introduce auxiliary variables $e_1 \in \mathbb{R}^{9(n-1)}, e_2 \in \mathbb{R}^{9(n-2)}, e_3 \in \mathbb{R}^{9(n-3)}$ representing the first, second, and third derivatives of the smoothed path. The constants w_0, w_1, w_2, w_3 are fixed weights.

We define the optimization problem:

$$\min_{p \in \mathbb{R}^{9n}} \quad \frac{1}{2} w_0 \|p\|_2^2 + w_1 \|e^1\|_1 + w_2 \|e^2\|_1 + w_3 \|e^3\|_1$$

$$\text{s.t.:} \quad \mathrm{lb} \leq p_t \leq \mathrm{ub}, \quad \forall t$$
$$\mathrm{trace}(p_t) = 0, \quad \forall t$$
$$\text{linear crop constraints on } p_t, \quad \forall t$$

where $p = (p_1, \ldots, p_n)^T, \quad p_t \in \mathbb{R}^9$

$$e^1(t) = p_{t+1} + f_{t+1} - p_t$$
$$e^2(t) = p_{t+2} + f_{t+2} - 2p_{t+1} - f_{t+1} + p_t$$
$$e^3(t) = p_{t+3} + f_{t+3} - 3p_{t+2}$$
$$\quad - 2f_{t+2} + 3p_{t+1} + f_{t+1} - p_t \quad (4)$$

The objective term $\frac{1}{2} w_0 \|p\|_2^2$ is intended to keep the correction log-homographies close to the identity, the goal being a stabilized path that is fairly faithful to the input path (see subsection 2.4). The objective term $w_1 \|e^1\|_1$, along with the corresponding constraint defining e^1, is intended to promote sparsity of the first derivative of the smoothed path via the ℓ_1-norm sparsity heuristic. e^1 is the first derivative of the stabilized log-path, $\tilde{s}_t = \tilde{f}_t + p_t$, so, using forward-differences:

$$e_1(t) = \tilde{s}_{t+1} - \tilde{s}_t = \tilde{f}_{t+1} + p_{t+1} - \tilde{f}_t - p_t$$
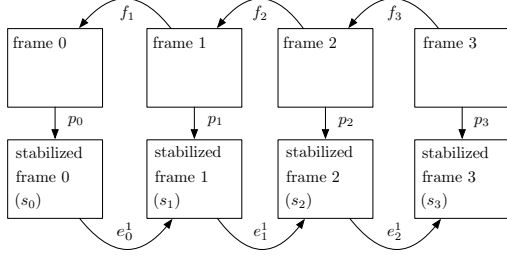$$= p_{t+1} + f_{t+1} - p_t \quad (\text{since } \tilde{f}_{t+1} - \tilde{f}_t = f_{t+1}).$$

Figure 5: The analysis log-homographies f_t map consecutive input frames. The corrections p_t are applied to a fixed crop window centered in each frame to stabilize the video. The derivatives e^i, $i = 1, 2, 3$, refer to the stabilized path.
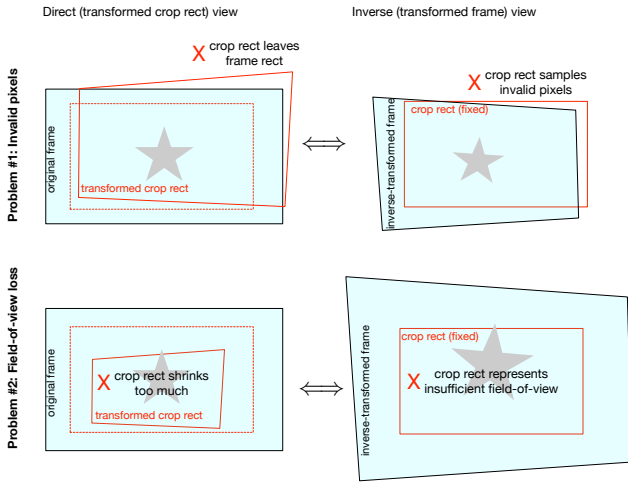

Figure 6: There are two basic problems that our crop constraints must guard against: sampling invalid pixels and loss of field-of-view. Recall that there are two equivalent views of the problem: corrections $\{P_t\}$ applied to the crop window within a fixed frame, or inverse-transforms $\{P_t^{-1}\}$ applied to the original frame, viewed through a fixed crop window. Here we show the two views on the two problems. Problem 1: the transformed crop rect leaves the frame (equivalently, the fixed crop rect samples from outside the inverse-transformed frame), and hence contains invalid pixels. Problem 2: the transform shrinks the crop rect too much (equivalently, inverse transform enlarges the frame too much), so that the crop represents insufficient field-of-view (despite a fixed size in pixels).

The e^2 and e^3 terms are similar, corresponding to second and third derivatives of the smoothed path (see supplement C). The trace constraint is needed because log-homographies have zero trace, and the crop constraints are described next.

2.3. Crop constraints

Stabilization necessarily results in crop (based on the intersection of valid pixels after all frames have been transformed by their respective corrections); in general, there is a tradeoff between stabilization strength and crop severity. We provide for adjustment of this tradeoff via a maximum-crop-allowance parameter, and build crop constraints to ensure that the stabilization does not exceed this allowance. A somewhat subtle point is that the crop fraction doesn't refer directly to the *dimensions* of the stabilized and cropped video, but to the fraction of the *field-of-view* represented therein. In other words, the maximum-crop parameter is a lower bound for the preserved field-of-view.

We model the problem in terms of the corrections to be applied to the crop window (which we generally make somewhat larger than the minimum allowed dimensions for the client-specified crop fraction, since the correction transformations may reduce its field-of-view). As shown in figure 6, we need to protect against two basic problems: invalid pixels and loss of field-of-view. Hence we design constraints that ensure that (1) the transformed crop window stays within the original frame rectangle, ensuring all pixels are valid, and (2) sufficient field-of-view is preserved after transforming the crop window.

Valid-pixel constraints We want to ensure that the transformed crop window lands fully inside the frame rectangle, so that it samples only valid pixels (to avoid problem 1 in figure 6). That is, if $D(h; c^{0:4}) : \mathbb{R}^9 \to \mathbb{R}^8$ computes the change in the positions of four corners due to a log-homography h, we want

$$(-\frac{w}{2}, -\frac{h}{2}) \leq D(p_t; c_{\text{wind}}^{0:4}) + c_{\text{wind}}^{0:4} \leq (\frac{w}{2}, \frac{h}{2}),$$

for each correction log-homography p_t, where $c_{\text{wind}}^{0:4}$ are the 4 corners of the original crop rectangle and w, h are the width and height of the original frame, with origin at the center of the frame. We linearize the constraints using the approximation $D(p_t; c_{\text{wind}}^{0:4}) \approx \nabla D|_{0, c_{\text{wind}}^{0:4}} p_t$, where $\nabla D \in \mathbb{R}^{8 \times 9}$ is the Jacobian with respect to h centered at 0, which linearly approximates how the corners move due to h (see supplement D eq. 3).

Field-of-view constraints The correction transformation must also preserve sufficient field-of-view (*i.e.* avoid problem 2 in figure 6). Since we can't express that requirement as a linear constraint, we instead approximate it by requiring that the area and sidelengths of the transformed crop window are at least as large as the area and sidelengths of a rectangle with dimensions $(1 - \text{crop fraction})$ times the frame dimensions. These constraints do not necessarily guarantee field-of-view preservation; under extreme distortion – large shear, for example – they can be satisfied but field-of-view can still be lost. However, our distortion constraints (section 2.5) and objective terms generally prevent such pathologi-

cal cases from occurring. So, the approximate field-of-view constraints are:

$$\text{Area}(c(p; c_{\text{wind}})) \geq (1 - \text{crop frac})^2 \times \text{Area}(\text{frame})$$
$$\text{Sidelen}(c(p; c_{\text{wind}})) \geq (1 - \text{crop frac}) \times \text{Sidelen}(\text{frame}),$$

for each correction p, where $c(p; c_{\text{wind}})$ denotes the corners of the crop window transformed by p. As with the valid-pixel constraints, we can linearize the area and sidelength constraints via the Jacobian (details in supplement E).

2.4. ℓ_2 Fidelity objective

The quadratic term $\frac{1}{2} w_0 \|p\|_2^2$ helps keep the correction log-homographies close to the identity. This encourages fidelity to the input camera trajectory in order to respect the original intent. (For example, if the input is a deliberate pan, we should not necessarily tripod-stabilize even if the constraints would allow.) The ℓ_2-term also acts as regularization, making the problem strictly convex and the solution unique. (Without saliency constraints, [9] is not strictly convex.) The fidelity term also offers control over stabilization strength, since varying w_0 trades off fidelity versus minimizing derivatives. w_0 can even be adjusted per-frame; for example, we may want to decrease stabilization strength for frames with serious motion blur, since removing the camera motion associated with motion blur can look unnatural.

2.5. Distortion mitigation

We attempt to minimize the perceived distortion that can result from stabilization corrections. For example, shearing and perspective transforms can create the appearance of unnatural, fluid motion of rigid objects, and planar distortions of non-planar content like faces can produce unpleasant results, particularly if the distortions vary rapidly over time.

Affine diagonal and off-diagonal. Grundmann [9] encourage rigidity in the correction transforms by constraining the diagonal and off-diagonal absolute difference and sum *i.e.* $|a - d| < \epsilon_1$, $|b + c| < \epsilon_2$, where a, b, c, d are the entries of the upper left 2×2 block of the corrections. We find it more helpful to add quadratic objective terms $\|a - d\|^2$, $\|b + c\|^2$ penalizing these quantities – the quadratic choice is mild for small values but becomes gradually severe for larger ones, as opposed to a constraint which provides no discouragement until hitting a hard stop.

Keystone-translation ratio. Supplement G shows that keystone and translation have a constant linear relationship under certain common circumstances. When this relationship exists in the input we should preserve it in the stabilized trajectory to avoid distortion. Given a keystone-translation ratio R in the input: *i.e.* $k \approx Rt$, where k is keystone and t is translation, we can add an objective term: $\|k_x - Rt_x\|^2 + \|k_y - Rt_y\|^2$ to encourage the desired relationship in the corrections. If the input and correction transforms preserve the ratio, the stabilized path will also.
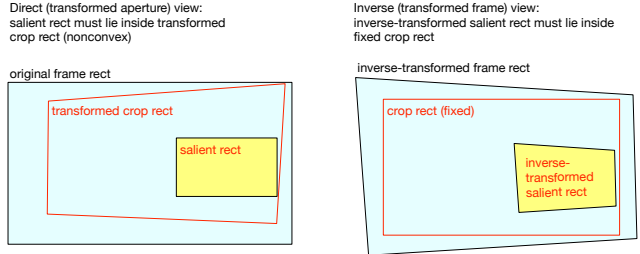

Figure 7: To ensure that a salient rectangle remains visible in the stabilized video, we must keep it inside the transformed crop window, or equivalently, make sure that the inverse-transformed salient rect is contained in the fixed crop rect. We can approximate the latter viewpoint as a convex constraint in terms of log-homographies.
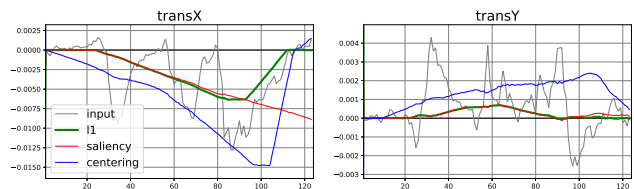

Figure 8: Stabilization with saliency constraints or centering objective (please see supplementary video *dog-l1-saliency-comparison.mp4*). Plots show value over time of translation homography elements (where homographies map between consecutive frames). Each plot compares default L1 (green), L1-with-saliency-constraint (red), and L1-with-centering-objective (blue).

2.6. Saliency constraints and centering objective

Many videos contain one or more salient objects or regions (people's faces, pets, or players at sports events [4], for example). We may wish to direct the stabilization path to 'focus' on, or at least avoid cropping out, these objects. In this section we present a few approaches to this problem.

Inclusion constraints. One approach is to add hard or soft constraints requiring that salient points are included in the cropped, stabilized video. This is similar in spirit to the proposal of Grundmann, but our log-homography transform simplifies the formulation considerably. This approach can handle any collection of salient points, arbitrary salient regions (by computing the convex hull and placing a point at each vertex), and bounding boxes (by placing a point at each corner). Suppose we wish to ensure that a point $s_t \in \mathbb{R}^2$ remains visible in frame t of the stabilized video. That is, s_t should be inside in the crop window after it has been corrected by $P_t = \exp(p_t)$. We can't express this directly as a convex constraint, so we instead note that it is equivalent to $P_t^{-1} s_t \in$ crop rect. (P_t^{-1} is the correction that would be applied to the original frame to stabilize it.)
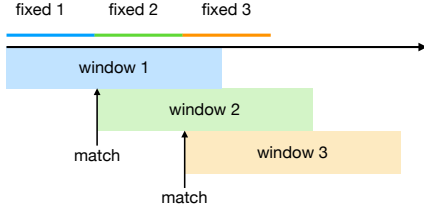
Figure 9: Basic windowed L1 strategy: iteratively solve within overlapping windows subject to constraints requiring beginning of current window to match previous fixed solution.

Using our earlier notation and linearization strategy from the crop constraints, $d(h; x)$ represents the delta in point x due to the corresponding homography $H = \exp(h)$, and so, using equation 3 ($\log H^{-1} = -h$), we have

$$P_t^{-1} s_t = s_t + d(-p_t, s_t) \approx s_t - \nabla d|_{0, s_t} p_t,$$

Hence we can approximate the constraint by the linearization: $s_t - \nabla d|_{0, s_t} p_t \in$ crop rect. [9] make a careful distinction between 'feature path' and 'camera path' formulations, (where corrections are applied to the video frame or crop window, respectively) – the corrections in the two cases are inverses of each other. The former choice makes the saliency constraints easy to formulate but the crop constraints difficult, and they cannot express some constraints in terms of P_t and others in terms of P_t^{-1} since the inverse operation is non-convex. But log-homographies avoid that issue, since the inverse of p_t is just $-p_t$.

Centering objective. We can also add an objective term that attempts to center a salient object in the stabilized video, creating a stronger sense of 'focus' than constraints alone. In general, we can guide the stabilized trajectory by adding an objective term to keep the corrections close to a sequence of target log-homographies: $\|p - p_{\text{target}}\|_2^2$ (we choose an ℓ_2 penalty to strongly discourage large deviations while accepting small ones). In this case, we can set the target as the pure-translation corrections that would center the salient object in each frame. An advantage of objective terms versus constraints is that infeasibility is not an issue – cases where the object cannot be centered are handled gracefully. Some disadvantages are that pulling toward targets far from the identity can introduce distortion, and such targets cannot be achieved without a large crop budget. For examples, please see figure 8 and the referenced video.

3. Windowed L1 stabilization

To handle arbitrary-length videos, we describe a windowing strategy to approximate the solution in linear time and bounded memory. L1 stabilization models the problem as a sparse quadratic program that scales linearly with the frame count, so the memory footprint grows unboundedly as the frame count increases. Also, the processing time grows superlinearly in the number of frames (the majority of L1 time is spent in an interior-point solver bottlenecked by an LDL factorization; dense LDL is in $O(n^3)$, while sparse LDL depends on the sparsity pattern: for this problem, it is roughly quadratic). Therefore, in order to solve in linear time with bounded memory, we propose a windowed strategy. Consider a window length l_w and stride length $l_s < l_w$ (in frames). As shown in figure 9, the strategy is:

1. Set window start to zero: $s_w = 0$.
2. Solve global problem in window $[s_w, s_w + l_w)$.
3. Fix the solution for the first l_s frames of the window.
4. Move window forward by $l_s - 3$ frames: $s_w \mathrel{+}= l_s - 3$.
5. Solve global problem in window $[s_w, s_w + l_w)$ subject to initialization constraint forcing first 3 frames of current window to match previous fixed solution: $p_t = p_t^{\text{fixed}}$ for $s_w \leq t < s_w + 3$.
6. Repeat (3)-(5) until the current window reaches end of the video, truncating the final window as necessary.

Step (5) is the crux: with the solution fixed up to frame $s_w + 3$, we actually wish to find the optimal solution in the entire range $[0, s_w + l_w)$ that agrees with the fixed solution in the range $[0, s_w + 3)$. A third-order Markov property lets us achieve this by initializing only the first 3 frames of the current window. The intuition is that the past and future are linked only by derivatives up to third order, so the effect of all previous frames is only propagated through derivatives that depend on the previous 3 frames. Details are in supplement F. With the windowed strategy, the memory footprint is bounded and equal to the memory required for a single window; the wall time for an input with $n > l_w$ frames is roughly $n \times \frac{\text{L1 time for } l_w \text{ frames}}{l_s}$. See figure 10.

4. Results

Quantitative comparisons: We follow the quantitative evaluation strategy proposed by [25], comparing *stability*, *distortion*, *cropping ratio*, and *running time* metrics as defined in [25] against five representative stabilization methods. Results are shown in figure 11 and details are in appendix A.1. Our method has by far the best *running time* and is competitive in the other metrics across all categories.
Qualitative comparisons: We provide a collection of L1-stabilized videos in the supplementary material; details are in supplement A. One of our main contributions is solving with homographies (versus affinities as used in [9]), which allow for keystone (perspective) correction. Videos described in A.2 demonstrate the superiority of homographies over affinities in removing residual keystone wobble. Next, we provide specific comparisons to [9], detailed in
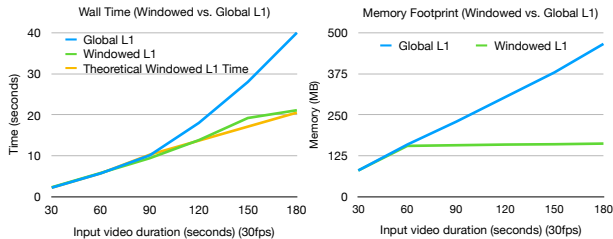
Figure 10: Time and memory of global versus windowed L1 stabilization measured on an iPhone XS, using window length $l_w = 1800$ (= 60s@30fps) and stride $l_s = 1500$ (= 50s@30fps). Global L1 time grows superlinearly and memory grows linearly in the number of input frames, while windowed L1 time grows linearly and memory is bounded. The measured windowed L1 time agrees well with the theoretical time, which is $n \times \frac{\text{L1 time for } l_w \text{ frames}}{l_s}$.
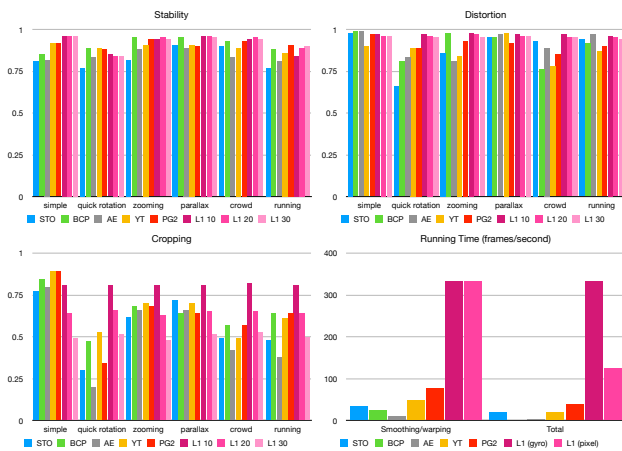

Figure 11: *Stability*, *distortion*, *cropping*, and *running time* metrics proposed in [25]. Comparisons of our method (L1), with a 10%, 20%, or 30% crop budget, against 'spatially and temporally optimized' (STO) [24], 'bundled camera paths' (BCP) [18], 'geodesic video stabilization' (homography version) (PG2) [25], Adobe After Effect (AE) warp stabilizer (based on [16]), and YouTube stabilizer (based on [9]). We use the dataset provided in [18] consisting of 174 videos divided into several categories. For *running time*, we consider both the total time and smoothing/warping time: L1 total time depends on the input motion analysis method: pixel-based analysis increases the total time while inertial sensor analysis (gyro) adds negligible time.

supplement A.3. Since [9] refine the affine solution with homographies in a 'residual motion suppression' step, they do not suffer from the severe residual keystone discussed above, but the post-processing step can consume significant additional crop on top of the fraction allowed in the optimization. Finally, supplement A.5 discusses stabilization of very shaky videos like *wakatipu-l1-affine.mp4*. The local nature of the log approximation might raise concern when

aggressive stabilization is required, but since we control the correction magnitude through the fidelity objective and crop and distortion constraints, we avoid making corrections severe enough to invalidate the approximation. A.5 also includes examples showing the effects of windowing.

Ablation study: An ablation study in supplement A.4 isolates the effects of individual objective terms and constraints. We evaluate the effect of the derivative terms, compare different crop allowances, and specifically assess key innovations relative to [9].

Performance: Timing data is shown in Figure 10. Our methods runs at about 300fps on an iPhone XS (assuming inertial sensor measurements for the input path). With windowing, runtime scales linearly in the number of frames. In A.1.1 table 1, we compare our performance with that of several alternatives (a superset of those in figure 11); our method is orders of magnitude faster than the 2.5-3D methods, and only comparable in performance to 2D methods; of the 2D methods our goals are most similar to [9]'s.

5. Conclusions, Limitations, and Future Work

Building on [9], we develop a convex optimization-based video stabilization framework that can create stationary, panning, and constant-acceleration motions characteristic of professionally shot video, accurately satisfy a crop constraint, stay close to the original camera path, and handle salient objects. Our contributions include a log-homography transform enabling homography modeling without loss of convexity, a principled linearization strategy for crop and saliency constraints, quadratic objective terms to encourage fidelity to the input path and possibly center salient objects, and a windowing strategy that approximates the solution in linear time and bounded memory. Limitations of this method include its 2D nature, which cannot handle depth/parallax but is essential for speed, and the fact that the model weights must currently be hand-tuned. In future work, we would like to study data-driven tuning of the model parameters using professionally-shot versus handheld videos as labeled data for supervised learning.

References

[1] Sudheer Achary, Syed Ashar Javed, Nikita Shravan, KL Moorthy, Vineet Gandhi, and Anoop Namboodiri. Cinefilter: Unsupervised filtering for real time autonomous camera systems. *arXiv preprint arXiv:1912.05636*, 2019.

[2] Sebastiano Battiato, Giovanni Gallo, and Salvatore Scellato. SIFT features tracking for video stabilization. In *14th International Conference on Image Analysis and Processing (ICIAP 2007)*, pages 825–830. IEEE, Sept 2007.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[4] Jianhui Chen and James J Little. Where should cameras look at soccer games: Improving smoothness using the over-

lapped hidden markov model. *Computer Vision and Image Understanding*, 159:59–73, 2017.

[5] Vineet Gandhi, Remi Ronfard, and Michael Gleicher. Multiclip video editing from a single viewpoint. In *Proceedings of the 11th European Conference on Visual Media Production*, pages 1–10, 2014.

[6] Michael L Gleicher and Feng Liu. Re-cinematography: Improving the camerawork of casual video. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 5(1):2, 2008.

[7] Amit Goldstein and Raanan Fattal. Video stabilization using epipolar geometry. *ACM Transactions on Graphics (TOG)*, 31(5):1–10, 2012.

[8] Matthias Grundmann, Vivek Kwatra, Daniel Castro, and Irfan Essa. Calibration-free rolling shutter removal. In *2012 IEEE international conference on computational photography (ICCP)*, pages 1–8. IEEE, 2012.

[9] Matthias Grundmann, Vivek Kwatra, and Irfan Essa. Autodirected video stabilization with robust l1 optimal camera paths. In *CVPR 2011*, pages 225–232. IEEE, 2011.

[10] Gustav Hanning, Nicklas Forslöw, Per-Erik Forssén, Erik Ringaby, David Törnqvist, and Jonas Callmer. Stabilizing cell phone video using inertial measurement sensors. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 1–8. IEEE, 2011.

[11] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[12] Alexandre Karpenko, David Jacobs, Jongmin Baek, and Marc Levoy. Digital video stabilization and rolling shutter correction using gyroscopes. Technical Report Tech. Rep. CSTR 2011-03, Stanford University Computer Science Department, Sept 2011.

[13] Johannes Kopf, Michael F Cohen, and Richard Szeliski. First-person hyper-lapse videos. *ACM Transactions on Graphics (TOG)*, 33(4):1–10, 2014.

[14] John M Lee. Smooth manifolds. In *Introduction to Smooth Manifolds*, pages 1–31. Springer, 2013.

[15] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving warps for 3d video stabilization. *ACM Transactions on Graphics (TOG)*, 28(3):1–9, 2009.

[16] Feng Liu, Michael Gleicher, Jue Wang, Hailin Jin, and Aseem Agarwala. Subspace video stabilization. *ACM Transactions on Graphics (TOG)*, 30(1):1–10, 2011.

[17] Shuaicheng Liu, Ping Tan, Lu Yuan, Jian Sun, and Bing Zeng. Meshflow: Minimum latency online video stabilization. In *European Conference on Computer Vision*, pages 800–815. Springer, 2016.

[18] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Bundled camera paths for video stabilization. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013.

[19] Shuaicheng Liu, Lu Yuan, Ping Tan, and Jian Sun. Steadyflow: Spatially smooth optical flow for video stabilization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4209–4216, 2014.

[20] Hui Qu and Li Song. Video stabilization with l1–l2 optimization. In *2013 IEEE International Conference on Image Processing*, pages 29–33. IEEE, 2013.

[21] Erik Ringaby and Per-Erik Forssén. Efficient video rectification and stabilisation for cell-phones. *International journal of computer vision*, 96(3):335–352, 2012.

[22] Robert Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[23] Shimon Ullman. The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 203(1153):405–426, 1979.

[24] Yu-Shuen Wang, Feng Liu, Pu-Sheng Hsu, and Tong-Yee Lee. Spatially and temporally optimized video stabilization. *IEEE transactions on visualization and computer graphics*, 19(8):1354–1361, 2013.

[25] Lei Zhang, Xiao-Quan Chen, Xin-Yi Kong, and Hua Huang. Geodesic video stabilization in transformation space. *IEEE Transactions on Image Processing*, 26(5):2219–2229, 2017.