Video Motion for Every Visible Point

Susanna Ricco[1,2]
ricco@google.com
[1]Google Research

Carlo Tomasi[2]
tomasi@cs.duke.edu
[2]Duke University

Abstract

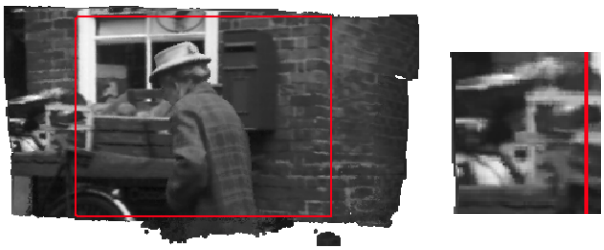
Dense motion of image points over many video frames can provide important information about the world. However, occlusions and drift make it impossible to compute long motion paths by merely concatenating optical flow vectors between consecutive frames. Instead, we solve for entire paths directly, and flag the frames in which each is visible. As in previous work, we anchor each path to a unique pixel which guarantees an even spatial distribution of paths. Unlike earlier methods, we allow paths to be anchored in any frame. By explicitly requiring that at least one visible path passes within a small neighborhood of every pixel, we guarantee complete coverage of all visible points in all frames. We achieve state-of-the-art results on real sequences including both rigid and non-rigid motions with significant occlusions.

1. Introduction

The goal of long-range, high-density motion estimation in video analysis is to compute the life of every point in a dense sampling of the visible surfaces in the scene. The image projection of a scene point moves along a *path* in the image plane. Sometimes the point is visible, and sometimes it is occluded by some object in the world or by the boundaries of the image. In a *dense* motion estimate, at least one path passes through every pixel of the sequence.

Dense, long-range motion estimation supports a number of applications. The computed paths can propagate to multiple frames any annotations or edits made in a single frame, thereby easing video labeling and editing. If visible paths can be extrapolated into regions where they are occluded, the occluding object can be removed from the video by painting the pixels it occupies with the extrapolated colors. Videos can be segmented into separate objects by clustering paths into coherent groups. The shapes and appearance of the resulting tube-like regions can support the detection and recognition of objects and activities.

Image motion information is either poor or altogether unavailable where the scene has little or no visual texture—the so-called *aperture problem*. As a consequence, regularization—or priors in probabilistic parlance—must be employed to extrapolate motion information from textured to poorly textured regions. To this end, we assume that (i) image paths live in a low-dimensional space, (ii) appearance remains approximately constant along the visible portion of a path, and (iii) exactly one world point is visible at every image point. The first assumption is exactly satisfied



(a) Our approach.

(b) Lagrangian motion [16].

Figure 1: Top: Result of transporting all gray levels in the 25-frame *marple7* sequence to frame 13 by the image motion computed with our method. The camera pans to follow Miss Marple as she walks from right to left. Pixels inside the red rectangle are native to frame 13. We find motion in all regions visible in any frame. Lagrangian motion (b) only computes paths for points visible in the first or last frame. The missing crate under the window is behind Miss Marple in the last frame and off-screen in the first. The missing portion of the wall to the right of the mailbox is behind Miss Marple in the first frame and off-screen in the last. Details on the right highlight regions where incorrect estimates in Lagrangian motion create artifacts which we avoid.

with rigid motion, and approximately satisfied in many circumstances. The second assumption is pervasive in motion analysis, and the third excludes semi-transparent objects.

Summary of Contributions

Our formulation is related to the concept of Lagrangian Motion Estimation (LME) proposed by Ricco and Tomasi [16]. Like them—and several others—we assume that paths belong to a low-dimensional subspace. We also anchor each path to a single pixel in the sequence, so as to keep paths from bunching up. Similarly to LME, we also describe path visibility with a binary, per-frame flag, and cast motion estimation as energy minimization.

However, our method differs in important ways from LME. First, we do not have fixed "reference frames" to anchor paths into. By default, LME selects the first and last frame as reference frames and estimates paths for only those scene points that are visible in one of those frames. Figure 1 illustrates this limitation; LME misses large regions in an intermediate frame because the surfaces are not visible in either reference frame. To guarantee that all visible surfaces are associated with paths, LME would have to select every frame as a reference frame, an approach that quickly becomes computationally infeasible for long videos. We find paths wherever they are visible.

Second, we minimize the energy function by direct nonlinear optimization rather than by solving Euler-Lagrange PDEs. The greater flexibility of our method allows for both anchors in any frame and more realistic regularization functionals leading to more accurate paths. For example, in LME, the regularization terms only enforce consistency between paths anchored in the same reference frame; our method encourages consistency across all frames. The details in the right column of Figure 1 show an example where the reconstructed image found with our method does not exhibit the artifacts visible in the LME results.

Third, we formulate the computation of the visibility flag as a *Maximum a Posteriori* (MAP) Markov Random Field (MRF) estimation problem, for which an efficient solution method is available. This formulation allows for the explicit enforcement of the constraint that there must be some visible path at every pixel. In contrast, LME's real-valued relaxation method for this combinatorial optimization problem approximates the target function, and leaves pixels unexplained.

2. Related work

Decades of research into motion estimation has focused primarily on the computation of optical flow fields between consecutive frames. Here, we consider approaches that compute longer paths.

Sundaram *et al*. [19] concatenate flow fields found by Large Displacement Optical Flow (LDOF) [4] into longer paths, each computed independently of the others. Their paths start in regions with sufficient texture, but cover more image regions than feature trackers like KLT [14] do. Paths end at detected occlusion and motion boundaries, found by comparing flow fields computed forward and backward in time. Sand and Teller [18] start with concatenated optical flow vectors but refine these *post facto* by optimizing a cost function with multiframe data and smoothness terms. High-cost paths end at suspected occlusions, and new ones are started to fill gaps.

Early formulations for temporal regularization penalize changes in image velocity in both time and space [22]. Structure-from-motion methods regularize more globally by assuming rigid motion—a restrictive assumption—for which image paths can be proven [20] to lie in a space of low and known dimension. This work has been extended to multiple rigid motions [8] and to non-rigid motion [3, 1]. These techniques precompute paths with frame-to-frame trackers, and de-noise them *post facto* by projection into a low-dimensional subspace.

More recent methods apply subspace constraints during path estimation to track points that are hard for a frame-to-frame tracker to follow. Early approaches applied subspace constraints during optical flow estimation to improve estimates in untextured regions of rigid scenes [12] or sampled from a path subspace to improve motion estimates along intensity edges affected by the aperture problem [21]. Garg *et al*. [9] combine subspace constraints with variational techniques adapted from optical flow estimation to solve for the multiframe registration of deforming surfaces. They compute full-length paths for every point in a selected reference frame. An extension softens the subspace constraint to create a prior on image motion [10, 11]. These methods do not handle occlusions, limiting their applicability.

LME finds paths by optimizing a global energy function over the entire video. It models visibility explicitly, and reconnects paths across brief occlusions. As explained earlier, we improve upon LME by removing its reliance on reference frames, handling visibility combinatorially rather than by approximate relaxation, and minimizing energy by direct optimization rather than variational methods. Our extension has the benefits mentioned in the introduction and demonstrated by the results in Section 6.

3. Model

Let p be an index into a set of *paths* $\boldsymbol{x}_p(t) : \mathcal{T} \rightarrow \mathbb{R}^2$, where $\mathcal{T}$ is the (discrete) time domain of the video sequence. A path is visible at time t iff its *visibility flag* $\nu_p(t) : \mathcal{T} \rightarrow \{0, 1\}$ is equal to 1 at time t. Both functions $\boldsymbol{x}_p(t)$ and $\nu_p(t)$ are unknowns to be estimated for all paths in a given video sequence. To ensure at least one visible path per pixel in every frame, we *anchor* $\boldsymbol{x}_p(t)$ to point $\boldsymbol{u}_p$ in some frame τ_p by letting $\boldsymbol{x}_p(\tau_p) = \boldsymbol{u}_p$ and $\nu_p(\tau_p) = 1$.

We require (and automatically select) enough anchor points to have some path pass through every pixel in the video. In contrast with LME, τ_p is path-specific and unrestricted.

Paths are assumed to be in the space spanned by a sequence-specific *basis* of paths $\{\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_K\}$, up to a shift:

$$\boldsymbol{x}_p(t) = \boldsymbol{u}_p + \sum_{k=1}^{K} c_{pk}(\boldsymbol{\varphi}_k(t) - \boldsymbol{\varphi}_k(\tau_p)) \; . \tag{1}$$

The motion relative to the anchor point $\boldsymbol{x}_p(\tau_p) = \boldsymbol{u}_p$ is determined by the unknown coefficients $\boldsymbol{c}_p = (c_{p1}, \ldots, c_{pK})$.

Since paths in a video with F frames have F points, the standard basis over $\mathbf{R}^{2F}$ can represent any path exactly. However, for many sequences a much more compact ($K \ll 2F$) basis is adequate, and provides powerful, sequence-specific regularization.

Given basis paths and anchor points, we find paths and visibility flags by interleaving computing optimal paths given visibility with computing optimal visibility given paths. The next two sections define the optimality criteria for these computations. Section 4 shows how to find the path basis and initial anchors, and Section 5 shows how to adjust the anchors and compute optimal paths and visibility.

3.1. Optimal paths

Given a basis of paths and a set of anchors, we find the best motion coefficients for each path by minimizing an objective function that penalizes changes in appearance along a path (temporal smoothness) and differences between nearby paths (spatial smoothness):

$$\sum_{p \in \mathcal{P}} \sum_{t=1}^{F} E_D(\boldsymbol{c}_p, t) + \lambda \sum_{p,q \in \mathcal{P}} E_S(\boldsymbol{c}_p, \boldsymbol{c}_q) \; . \tag{2}$$

The terms in the first summation,

$$E_D(\boldsymbol{c}_p, t) = \nu_p(t) \Psi(\Delta I_p(t)) \; , \tag{3}$$

with $\Delta I_p(t) = I(\boldsymbol{c}_p, t) - I(\boldsymbol{c}_p, \tau_p)$, employ a robust penalty function $\Psi(s) = \sqrt{s^2 + \epsilon^2}$ to measure the difference between the image intensity $I(\boldsymbol{c}_p, t) = I(\boldsymbol{x}_p(t))$ of the path in frame t and that at the anchor $\boldsymbol{u}_p$ in frame τ_p. Multiplication by $\nu_p(t)$ ensures that this penalty is levied only on visible points. The terms in the second summation above,

$$E_S(\boldsymbol{c}_p, \boldsymbol{c}_q) = \alpha_{pq} \sum_{k=1}^{K} \Psi(c_{pk} - c_{qk}) \; , \tag{4}$$

measure the difference between the motion coefficients of pairs of paths. The multiplier α_{pq} couples nearby paths that have similar appearance, and is equal to

$$\alpha_{pq} = \exp\left(-\frac{(I(\boldsymbol{c}_p, \tau_p) - I(\boldsymbol{c}_q, \tau_q))^2}{\sigma^2} \right) \tag{5}$$

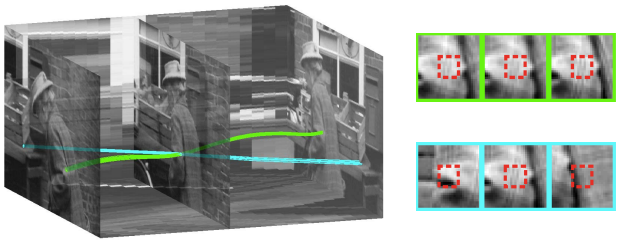


Figure 2: A spatiotemporal cube of the *marple7* sequence. Time runs from left to right. The corner of the crate (cyan) is first occluded by Miss Marple's arm (green) in frame 12. A small patch (red dashed squares) around each path in every frame is transported along the current path estimates and monitored for consistent appearance. The arm patch (top right) is most consistent, and makes this the controlling path at that point and frame. Points along paths that either coincide with or are substantially parallel to a nearby controlling path have their observed visibility flag $\hat{\nu}_p(t)$ set to 1. All other flags are set to 0. Observed flags affect the estimated visibility flags at the nodes of a MRF that enforces spatial and temporal consistency of the flags and ensures that at least one path is visible at every pixel.

if the path p is visible in the anchor frame of path q (that is, if $\nu_p(\tau_q) = 1$) and passes close enough to the anchor of q (that is, if $||\boldsymbol{x}_p(\tau_q) - \boldsymbol{u}_q|| < \Delta$). Otherwise, $\alpha_{pq} = 0$.

3.2. Optimal visibility

The binary visibility flag $\nu_p(t)$ for each path and frame is modeled as a MRF whose structure depends on the current estimates $\boldsymbol{x}_p(t)$ of the paths $p \in \mathcal{P}$. The MRF has one node for each point $\boldsymbol{v}_p(t) = (\boldsymbol{x}_p(t), t)$ along some path, for $t = 1, \ldots, F$, and one binary random variable $\nu_p(t)$ per node. The neighborhood of $\boldsymbol{v}_p(t)$ is the set of points $\boldsymbol{v}_q(t)$ with $q \neq p$ and $\|\boldsymbol{v}_p(t) - \boldsymbol{v}_q(t)\| \leq \Delta$ for some small fixed Δ (spatial neighborhood), plus the two points $\boldsymbol{v}_p(t-1)$ and $\boldsymbol{v}_p(t+1)$ that are temporally adjacent to $\boldsymbol{v}_p(t)$ along path p (temporal neighborhood).

Each node in the MRF is associated with a binary *observed visibility* flag $\hat{\nu}_p(t)$ computed from the data as follows. Path points in each frame are scored by their *patch consistency*, which measures how little a patch around $\boldsymbol{v}_p(t)$ changes as it is transported by the current estimates of paths near $\boldsymbol{v}_p(t)$ to (i) a few frames before and after time t, and (ii) the anchor frame τ_p for path p. We use equation (11) from LME [16] to compute patch consistency and declare the *controlling path* at $\boldsymbol{v}_p(t)$ to be the most consistent path through the spatial neighborhood of $\boldsymbol{v}_p(t)$. Let now

$$\bar{d}_{pq} = \frac{1}{F} \sum_{t=1}^{F} ||\boldsymbol{x}_p(t) - \boldsymbol{x}_q(t)|| \tag{6}$$

be the average distance between two paths, and let p^* be

the controlling path at $\boldsymbol{v}_p(t)$. Then, the observed visibility $\hat{\nu}_p(t)$ is defined as follows (see also Figure 2):

$$\hat{\nu}_p(t) = \begin{cases} 1 & \text{if } \bar{d}_{pp^*} \leq 4 \text{ pixels} \\ 0 & \text{otherwise} \end{cases} . \tag{7}$$

In words, a path p is observed to be visible at $\boldsymbol{v}_p(t)$ when it either coincides with ($p = p^*$ so that $\bar{d}_{pp^*} = 0$) or is nearly parallel ($\bar{d}_{pp^*} \leq 4$) to the controlling path p^* at $\boldsymbol{v}_p(t)$. Because we require that paths must be visible in their anchor frames, we also always set $\hat{\nu}_p(\tau_p) = 1$.

The observed visibility flags $\hat{\nu}_p(t)$ influence the (hidden) visibility flags $\nu_p(t)$ through a data term in the MRF. We define the following average measure of intensity change along the visible portion of path p:

$$\Delta_p = \frac{\sum_{t=1}^{F} \hat{\nu}_p(t) \Delta I_p(t)}{\sum_{t=1}^{F} \hat{\nu}_p(t)} , \tag{8}$$

with $\Delta I_p(t)$ as defined in Section 3.1. For correctly estimated paths, this measure reflects variations of intensity caused by unmodeled effects such as image noise or global illumination changes, rather than by occlusions. The data term of the MRF is then defined as follows:

$$\begin{aligned} D(\nu_p(t) = 1) &= \Delta I_p(t) + \lambda_L (1 - \hat{\nu}_p(t)) \\ D(\nu_p(t) = 0) &= \Delta_p + \lambda_L \hat{\nu}_p(t) . \end{aligned} \tag{9}$$

The terms with multiplier λ_L bias estimated visibility values $\nu_p(t)$ toward observed values $\hat{\nu}_p(t)$. Setting a point to be visible incurs the additional charge $\Delta I_p(t)$, equal to the change in intensity between anchor and current point. Setting a point to be invisible incurs the additional charge Δ_p that accounts for the fact that intensity variations may be caused by other than occlusions.

The weights on edges between the random variables of the MRF encourage both temporal and spatial consistency among visibility values. Specifically, a penalty

$$V\Big(\nu_p(t),\, \nu_p(t+1)\Big) = \lambda_T |\nu_p(t) - \nu_p(t+1)| \tag{10}$$

is added between temporally adjacent neighbors to discourage changes of visibility along a path. The weight on an edge between spatial neighbors is

$$V\Big(\nu_p(t),\, \nu_q(t)\Big) = \lambda_S w_{pq}(t) |\nu_p(t) - \nu_q(t)| \tag{11}$$

with

$$w_{pq}(t) = \frac{e^{-\left(\frac{\Delta I_{pq}(t) + \Delta I_{pq}}{\sigma^2}\right)}}{\bar{d}_{pq} + \epsilon} \tag{12}$$

where $\epsilon > 0$ prevents division by zero. In this expression,

$$\begin{aligned} \Delta I_{pq}(t) &= (I(\boldsymbol{c}_p, t) - I(\boldsymbol{c}_q, t))^2 \\ \Delta I_{pq} &= (I(\boldsymbol{c}_p, \tau_p) - I(\boldsymbol{c}_q, \tau_q))^2 . \end{aligned} \tag{13}$$

In words, $\Delta I_{pq}(t)$ measures difference in appearance between paths in a single frame, and ΔI_{pq} measures a similar difference between anchor points. The combined effect of these two terms is to push discontinuities in visibility closer to intensity boundaries, and the division by $\bar{d}_{pq}$ reduces the spatial discontinuity penalty between unrelated paths.

To enforce the physical constraint that there must be some visible point at every pixel, we clamp some visibility values to 1 and remove the corresponding nodes from the MRF. Specifically, we require that $\nu_p(\tau_p) = 1$ and $\nu_{p^*}(t) = 1$ with p^* a controlling path in frame t. We roll the pairwise cost for each edge incident to a clamped node into the unary cost for the other node of that edge.

4. Preliminaries

Before we solve for motion and visibility, we select basis paths and an initial set of anchors, paths, and visibility flags.

4.1. Finding the basis paths

Basis paths are obtained by first tracking a sparse set of feature points with a frame-to-frame tracker [14]. This yields several *tracks*, that is, paths that do not necessarily extend through the entire sequence. These tracks are supplemented with those formed by concatenating optical flow vectors between consecutive frames [19], as described in more detail in Section 4.2, where we do the same to initialize a dense set of paths.

For some sequences, several tracks may extend from first to last frame. PCA can then yield a basis whose size K is determined by adding principal components until the reconstruction residual for the input tracks is below, *e.g.*, 2 pixels.

In general, however, occlusions and tracking failures make tracks start late and end early, leading to a matrix of track coordinates with missing entries. We iterate between matrix factorization with missing data [6] and a compaction step that associates tracks corresponding to the same world point [17]. If needed, a user can be asked to correct mistakes in data association. We scale path coordinates so that the mean per-path motion between frames is one pixel.

4.2. Initialization

To cover every pixel in a video sequence with paths, we need to create a number of paths of the same order of the number of visible points in the sequence. Placing anchor points at every pixel in the first and last frame at worst overestimates the true number of anchors needed by a factor of two. We place additional anchors to cover visible regions that happen to be occluded in these two particular frames by following a procedure inspired by Sundaram *et al.* [19]. We first concatenate optical flow vectors into multiframe tracks, which we break when the optical flow field fails a forward-backward consistency check or when the point is too close

to a motion boundary (equations (5) and (6) from [19], respectively). To prevent merging foreground and background tracks, we create a thin empty buffer around the regions where tracks terminate. We initialize a new track at any pixel that is more than one pixel away from any track propagated from the previous frame.

Track fragments that start in the first frame are converted into paths with anchor points in the first frame. If the fragments are long enough, the initial coefficients for the path are computed by projecting the track onto the path basis. Otherwise, we select coefficients by copying from nearby track fragments, picking the coefficients that create the path with the best brightness constancy measured over a few frames. Track fragments that reach the last frame are converted into paths anchored at points in the last frame using the same procedure. The temporal extent of the track fragments provides an initial conservative estimate for the path visibility flags.

For all other track fragments, we place a (possible) anchor point in the frame in which they are initialized and convert to a path as before. We iterate through these potential paths and only include the anchor points for those that differ from already included paths by more than an average of 2 pixels per frame. Figure 3a shows the anchor points selected in this way for the *marple7* sequence. Colors other than gray are anchors, and similar colors correspond to similar sets of path coefficients.

After this initialization stage, the energy functions defined in Section 3 are minimized by the algorithms described in Section 5. This can result in the insertion of additional anchor points. Figure 3b shows the color-coded anchor points after convergence.

5. Optimization

Starting with the paths and visibility flags constructed as described in Section 4.2, we interleave two steps during optimization: a combinatorial optimization step finds visibility flags $\nu_p(t)$ for the current path estimates, and a continuous optimization step updates path coefficients $\boldsymbol{c}_p$ given the current visibility estimates. In the process, we add anchor points until every pixel in the sequence has at least one path through it, and remove anchors of invisible paths. We stop when the maximum change in every path falls below one pixel in every frame.

The initial path estimates are often poor along occlusion boundaries, because visibility is not yet accounted for. Because of this, we heuristically regroup paths between each combinatorial and continuous step to let foreground and background vie for paths between them.

We now describe the continuous step, path regrouping, combinatorial step, anchor management, and termination.

Continuous step. We update path coefficients by minimizing the energy function (2) via trust-region Newton conjugate gradients optimization [15]. This method only requires computing vectors of the form $H\boldsymbol{v}$ where H is the Hessian, rather than the very large but sparse H itself. The sparsity pattern of H changes over time because the coupling coefficients α_{pq} in equation (5) depend in turn on the path coefficients. When computing successive conjugate gradients, we treat the terms α_{pq} as constants—a good approximation for small path perturbations—and recompute them between full descent steps.

Path regrouping. After 40 descent steps, we allow paths to copy their coefficients and visibility flags from one of their neighbors if doing so improves the path's fit to data. Specifically, path p copies from q if $\tau_p \neq \tau_q$, $\nu_q(\tau_p) = 1$, $||\boldsymbol{x}_q(\tau_p) - \boldsymbol{u}_p|| < \Delta$, $\sum_{t=1}^{F} \nu_q(t) \geq F/2$, and the copy improves the data fit for p the most.

Combinatorial step. Visibility flags are updated after path regrouping by using graph cuts [2, 13] to compute the MAP estimate for the MRF in Section 3.2 . The energy function is amenable to this method as the edge costs (11) satisfy

$$V(0,0) + V(1,1) \leq V(0,1) + V(1,0)\ . \qquad (14)$$

Anchor management. When the maximum change in any path in any frame is less than one pixel, we check that every pixel in the video has a visible path through it. If not, we add new anchor points to fill voids and resume optimization. Newly inserted paths copy their initial parameters from the closest visible path.

Anchors on paths that are invisible everywhere except at the anchor itself (which is always visible) are deleted. These one-point paths occur when visibility estimation correctly identifies an outlier with an incorrect path estimate.

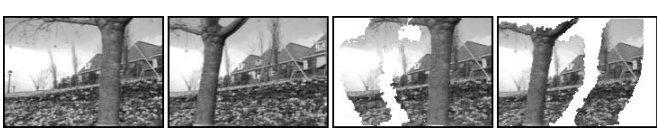
Termination. Optimization terminates when all path estimates change by less than a pixel in every frame and all pixels in the video have a path through them.

6. Results

We evaluate the performance of our technique on five real sequences of increasing complexity, all with large motions and significant occlusions. The popular *flowerbed* (29 frames) and a new sequence with a *truck* driving behind a road sign (33 frames) contain only rigid motion. The three with non-rigid motion are from the Berkeley motion segmentation dataset [5]: 60 frames from *marple1*, 72 frames from *marple8*, and 25 frames from *marple7*. The *marple7* and *flowerbed* sequences are the same as those evaluated in LME. Figure 4 shows sample frames; the full sequences and more detailed results can be found at `www.cs.duke.edu/~tomasi/video-motion`. We set $\lambda = 1$, $\sigma = 50$, and $\lambda_L = \lambda_T = \lambda_S = 0.5$ and used the same values for all five sequences. In our experiments, the results were relatively insensitive to small changes in the values of λ or σ, but were more sensitive to the values of the parameters for the occlusion detection step.

(a) (b)

Figure 3: Anchor points selected during initialization (a) and at convergence (b). Colors other than gray denote anchor points, and similar colors denote similar sets of path coefficients. Due to space constraints we display odd-numbered frames only (although anchors exist in all frames). Note the improved segmentation of Miss Marple after convergence. The initial path estimates in regions near occlusions are poor because the occlusions have yet to be considered properly. Path regrouping allows the solution to escape from local optima near the initial solution and achieve a much better final solution.



(a) Flowerbed. Two basis functions.

(b) Truck. Four basis functions.

(c) Marple1. Eight basis functions.

(d) Marple7. Five basis functions.

(e) Marple8. Eight basis functions. Miss Marple (off-screen in the first and last frames) walks across the scene, completely occluding the background. An intermediate frame can be seen in Figure 5b.

Figure 4: Results of our method. For each sequence, we show the first and last frames, followed by the last frame warped to align with the first frame, and vice versa. Regions detected as occluded in the source frame of the warp are marked in white. Full videos and comparison with LDOF trajectories and LME paths are available on the project website.

6.1. Qualitative evaluation

For a qualitative evaluation, we use our motion results to warp all frames to a selected frame. This creates a motion-compensated video that should appear static except for regions that are occluded in a particular frame (which we detect and paint white). Figure 4 shows the last frame aligned to the first frame, and vice versa, for all sequences.

We compare our results to LME paths and LDOF trajectories using implementations provided by their authors. We also ran two-frame optical flow between all pairs of frames, but even methods specifically aimed at large displacements [4] failed for distant frames. Figure 5 shows examples of mistakes made by competing methods. Videos showing comparisons for all sequences and methods are available on the project website.

6.2. Quantitative evaluation

It is difficult to get reliable ground truth paths for realistic sequences. The synthetic datasets in [7] do not preserve associations across occlusions. Manual labeling for real sequences is painstaking and unreliable, particularly for complex motions or low-texture regions.

Instead, we measure the degree to which intensities remain constant along computed paths as a proxy for geometric accuracy. The *all-path interpolation error* (APIE) is the absolute deviation from the expected appearance of the appropriate path, averaged over all visible frames for all paths. We use the median intensity value along the estimated visible portion of each path

$$\hat{I}_p = \arg\min_a \sum_t \nu_p(t) |I(\boldsymbol{c}_p, t) - a| \,. \qquad (15)$$

Sequence	Method	APIE	Path length		Pixel distance		
			Mean	Std. dev.	50th	95th	99th percentile
Flowerbed	LDOF traj.	4.54	11.2	10.5	0.47	8.5	15.2
	LME	3.65	**23.9**	7.3	0.31	0.79	1.3
	Ours	**2.59**	23.1	7.4	**0.29**	**0.66**	**0.85**
Truck	LDOF traj.	5.40	6.8	7.5	1.2	47.0	70.0
	LME	5.97	**23.4**	7.4	0.39	1.9	4.4
	Ours	**3.56**	20.9	9.1	**0.28**	**0.67**	**0.91**
Marple7	LDOF traj.	2.50	6.7	6.4	0.47	6.9	13.3
	LME	2.64	**15.9**	7.5	0.43	5.7	9.7
	Ours	**2.27**	14.7	6.8	**0.30**	**0.68**	**0.87**
Marple1	LDOF traj.	4.57	9.4	11.4	0.51	14.7	26.7
	LME	4.11	**25.9**	19.4	0.62	9.1	18.9
	Ours	**2.61**	11.21	14.7	**0.32**	**0.84**	**1.0**
Marple8	LDOF traj.	3.70	14.9	14.7	0.47	7.4	16.4
	LME	5.17	**59.9**	15.7	0.35	1.9	6.0
	Ours	**2.79**	29.5	25.3	**0.24**	**0.65**	**0.90**

Table 1: Solution quality metrics. **APIE** measures average intensity constancy along estimated paths (smaller is better, assuming the brightness constancy assumption holds). **Path length** is the number of frames in which a path is reported as visible (longer is typically better). **Pixel distance** measures path density by reporting the distance to the nearest visible path for each pixel. We report the 50th, 95th, and 99th percentiles (smaller is better).

as the expected appearance to eliminate dependence on the choice of anchor frame. Our quality metric is then

$$\text{APIE} = \frac{\sum_p \sum_t \nu_p(t) |I(\boldsymbol{c}_p, t) - \hat{I}_p|}{\sum_p \sum_t \nu_p(t)} \,. \tag{16}$$

Note that APIE for ground truth paths would typically be non-zero as it also measures violations of the brightness constancy assumption. In general, however, lower values for APIE indicate better performance. Table 1 reports APIE for each sequence, computed with intensity values in $[0, 255]$. LDOF trajectories do not report visibility; correspondences after occlusions are simply missing. We treat missing entries as if they had $\nu_p(t) = 0$.

We could artificially decrease APIE by returning many paths, each visible in only a few frames. This runs contrary to our goal of recovering correspondences between distant frames across occlusions. We confirm that our solution is not overwhelmed by numerous short tracks by measuring the average visible length of a path. As shown in Table 1, our method and LME return significantly longer paths on average because they detect dissoclusions.

A key feature of our algorithm is the ability to compute the path for every visible point in a scene. We measure path density by computing the distance to the closest visible path for each pixel in the sequence. Table 1 reports the 50th, 95th, and 99th percentile for each method. LDOF trajectories leave many pixels unexplained because they are not initialized in low-texture areas. LME misses objects not visible in either the first or last frame of a sequence. In many sequences, these missed objects can account for a significant fraction of the scene.

In addition to the increased pixel coverage, our algorithm is faster than LME. For example, we terminated after 1.4 hours on the flowerbed sequence; LME took 3.4 hours. The marple8 sequence required the largest motion basis and contains the most frames, making it the most expensive for LME at 205.6 hours. Video motion took 89.4 hours. Because video motion paths cover every visible point, our running time scales with the quantity of motion. As a result, the marple1 sequence requires the most computation. We took 150 hours compared to 188.5 for LME. Due to its high computational cost, we ran LME for only 10 iterations instead of the suggested 20 on both sequences.

7. Conclusion

We introduced a method to compute extended video motion paths that explain every pixel of a video sequence. We regularize the solution by projection on a low-dimensional basis of motion paths, and can follow points through brief occlusions. In contrast with previous methods, we can handle occlusions wherever they occur. This is made possible by a new, non-variational formulation that allows for more realistic visibility and appearance constraints and is also more efficient than the variational approach. While we focused on grayscale, extending to color is straightforward.

Much work remains to be done. Video with many or highly deformable moving objects such as crowds or flags [9] may require nonparametric methods, and our solution may perhaps be used for initialization. Sparsity-inducing priors on the path coefficients are an intriguing alternative, in which each path is allowed to use a small

(a) Warp from frame 60 of *marple1* (LDOF vs. ours).

(b) Warp from frame 13 of *marple8* (LME vs. ours).

Figure 5: Examples of mistakes made by other methods visible when warping to align with the first frame. Correct warps should match the first image in each row with occluded pixels replaced with white masks. The second image in each row shows the source frame for the warp. Results from video motion paths (rightmost column) are consistently higher quality; the third column shows the inferior result from a competing method. In (a), fewer LDOF trajectories survive to the last frame. Those that do suffer from drift: the man's face is misaligned in the warped image. In (b), because LME paths ignore objects not visible in the reference frames, Miss Marple is untracked. As a result, LME cannot reliably detect the occlusion she causes and erroneously includes her image in the warped frame. Video motion paths track every visible point and correctly mark the background she obscures as occluded in the warp. Full results for competing methods can be found on the project website.

number of functions from a large basis. Processing even longer sequences than the one we can handle will require models of long-term, global changes in illumination, and separate motion bases for different video segments. How to splice together solutions from these segments is an open challenge, and so is greater computational efficiency.

Acknowledgements. This work is supported by the Army Research Office under Grant No. W911NF-10-1-0387 and by the National Science Foundation under Grant IIS-10-17017.

References

[1] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade. Trajectory space: a dual representation for nonrigid structure from motion. *PAMI*, 33(7):1442–1456, 2011. 2

[2] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9):1124–1137, 2004. 5

[3] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3D shape from image streams. In *CVPR*, pages 690–696, 2000. 2

[4] T. Brox and J. Malik. Large displacement optical flow. In *CVPR*, pages 41–48, 2009. 2, 6

[5] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, pages 282–295, 2010. 5

[6] A. Buchanan and A. Fitzgibbon. Damped Newton algorithms for matrix factorization with missing data. In *CVPR*, pages 316–322, 2005. 4

[7] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, pages 611–625, 2012. 6

[8] J. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *IJCV*, 29(3):159–179, 1998. 2

[9] R. Garg, L. Pizarro, D. Rueckert, and L. Agapito. Dense multi-frame optic flow for non-rigid objects using subspace constraints. In *ACCV*, pages 460–473, 2010. 2, 8

[10] R. Garg, A. Roussos, and L. Agapito. Robust trajectory-space TV-L1 optical flow for non-rigid sequences. In *EMMCVPR*, pages 300–314, 2011. 2

[11] R. Garg, A. Roussos, and L. Agapito. A variational approach to video registration with subspace constraints. *IJCV*, 104(3):286–314, 2013. 2

[12] M. Irani. Multi-frame correspondence estimation using subspace constraints. *IJCV*, 48(3):173–194, 2002. 2

[13] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2):147–159, 2004. 5

[14] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, pages 674–679, 1981. 2, 4

[15] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer Verlag, 2nd edition, 2006. 5

[16] S. Ricco and C. Tomasi. Dense Lagrangian motion estimation with occlusions. In *CVPR*, pages 1800–1807, 2012. 1, 2, 3

[17] S. Ricco and C. Tomasi. Simultaneous compaction and factorization of sparse image motion matrices. In *ECCV*, pages 456–469, 2012. 4

[18] P. Sand and S. Teller. Particle Video: long-range motion estimation using point trajectories. *IJCV*, 80(1):72–91, 2008. 2

[19] N. Sundaram, T. Brox, and K. Keutzer. Dense point trajectories by GPU-accelerated large displacement optical flow. In *ECCV*, pages 438–451, 2010. 2, 4, 5

[20] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *IJCV*, 9(2):137–154, 1992. 2

[21] L. Torresani and C. Bregler. Space-time tracking. In *ECCV*, pages 801–812, 2002. 2

[22] J. Weickert and C. Schnörr. Variational optic flow computation with a spatio-temporal smoothness constraint. *Journal of Mathematical Imaging and Vision*, 14(3):245–255, 2001. 2