

Depth-from-Trajectories for Uncalibrated Multiview Video

Paul A. Ardis, Amit Singhal, Christopher M. Brown

Copyright 2009 Society of Photo-Optical Instrumentation Engineers.

This paper was published in the Proceedings of the SPIE, vol. 7252, and is made available as an electronic reprint with permission of SPIE. One print or electronic copy may be made for personal use only. Systematic or multiple reproduction, distribution to multiple locations via electronic or other means, duplication of any material in this paper for a fee or for commercial purposes, or modification of the content of the paper are prohibited.

Depth-from-Trajectories for uncalibrated multiview video

Paul A. Ardis^{a,†}, Amit Singhal^b, Christopher M. Brown^a

^aUniversity of Rochester, Rochester, NY, USA 14627

^bResearch Laboratories, Eastman Kodak Company, Rochester, NY, USA, 14650-2103

ABSTRACT

We propose a method for efficiently determining qualitative depth maps for multiple monoscopic videos of the same scene without explicitly solving for stereo or calibrating any of the cameras involved. By tracking a small number of feature points and determining trajectory correspondence, it is possible to determine correct temporal alignment as well as establish a similarity metric for fundamental matrices relating each trajectory. Modeling of matrix relations with a weighted digraph and performing Markov clustering results in a determination of emergent depth layers for feature points. Finally, pixels are segmented into depth layers based upon motion similarity to feature point trajectories. Initial experimental results are demonstrated on stereo benchmark and consumer data.

Keywords: Depth-from-X, Multiview, Stereo, Video Processing

1. INTRODUCTION

Computer vision methods restricted to purely laboratory use are like skyscrapers without elevators: intriguing, but impractical. For instance, while it is possible to precisely calculate depth maps in overlapping views given sufficient scene and camera information, most practical situations do not allow for such precision and prior knowledge. We desire a robust method for producing depth maps for uncalibrated video while minimizing computational expense, for application in fields such as mobile robotics and consumer multiview imaging.

Zhang *et al.*¹ are well known for having posed a solution to precisely this problem, relying on epipolar constraints to solve for intercamera and interframe motion parameters. However, producing depth maps from their results requires first calibrating all inputs and then solving for pixelwise stereo (or rendering from captured 3D geometry), a process that is computationally expensive regardless of scene content and assumes that temporal alignment is already solved. Likewise, structured light approaches like those of Furukawa and Kawasaki² also require a large number of computations per pixel no matter the complexity of the captured scene, and additionally require specialized projection hardware. Depth-from-focus methods like the work of Levin *et al.*³ introduce additional capture constraints (as objects with discernable depth must be located beyond the plane of maximum focus) and only guarantee correct results with the introduction of specialized optical equipment (*i.e.*, coded apertures). Finally, single-camera layer methods⁴⁻⁶ do not provide a means for determining consistent multiview depth, ignore temporal alignment between videos, and additionally make use of extremely expensive methods for pixel depth computation (*e.g.*, pixelwise mixture-of-Gaussians fitting using Expectation Maximization), fail to group disparate moving objects that are located at the same depth, or make further assumptions about scene content (*e.g.*, piecewise constant velocity).

We propose a novel method for depth map recovery from overlapping video under minimal capture assumptions, where computational complexity is primarily dependent upon scene complexity (rather than resolution) and without the assistance of specialized hardware. By observing feature trajectories that appear in multiple videos (formed by salient features undergoing unconstrained combinations of scene and camera motion), we are able to determine the temporal alignment necessary to rectify each pair of videos and to identify features that are located at approximately the same depth relative to the camera configuration. An extremely small number of computations are then required to identify the depth layer that best classifies each pixel, with this number growing linearly with the number of trajectories with a naïve implementation (reduced to linear growth with the number of feature depth layers upon adoption of suggested optimizations).

This paper explains the method in detail (Section 2) and presents initial experimental results (Section 3) and our conclusions (Section 4).

[†] ardis@cs.rochester.edu; Dept. of Comp. Science, Univ. of Rochester, P.O. Box 270226, 734 Comp. Studies Bldg., Rochester, NY 14627-0226

2. METHOD

Processing can be divided into three stages: identifying, matching, and relating feature trajectories (Section 2.1); modeling and clustering trajectory relations using a complete weighted digraph (Section 2.2); and finally determining pixel depth assignments using motion similarity with feature trajectories (Section 2.3). The process is summarized in Figure 1.

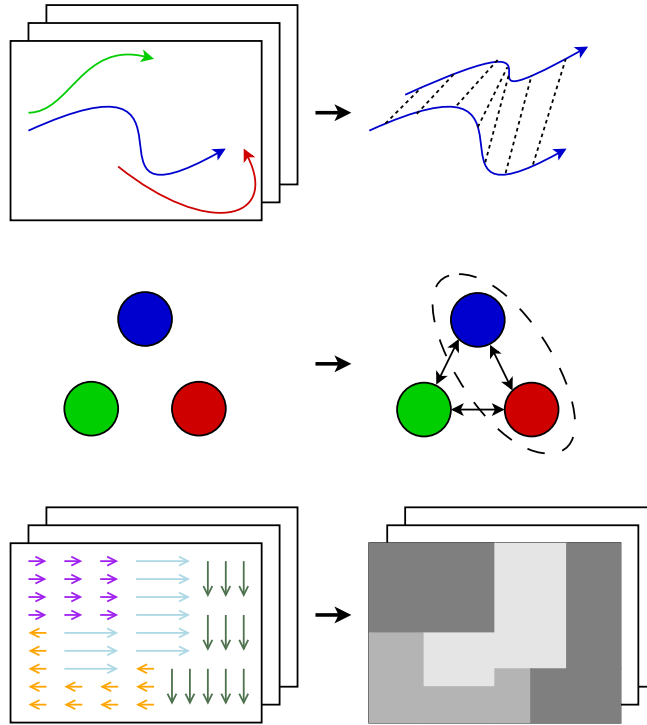


Fig. 1. Stages of Processing: Trajectories are identified from feature tracks (**top left**), matched across views, and studied to determine temporal and spatial alignment parameters that best describe their appearance in each view (**top right**); spatial alignment information is used to graphically model the pairwise relationships between trajectories (**middle left**), whereupon Markov Clustering is performed in order to identify trajectories that are located in the same depth plane (**middle right**); pixelwise motion is approximated by color match (**bottom left**) in keeping with interframe trajectory motion to determine the trajectory (and, therefore, the depth plane) that best classifies each pixel in the final depth map (**bottom right**).

2.1 Feature trajectories

The first step toward depth map construction is the identification and tracking of salient image features in all frames of each input video. The method of feature detection used may depend upon the realm of application (as with trained bags of features from classification data) or aim to identify general points of interest (as with difference-of-Gaussian extrema), although scenario-unspecific approaches are encouraged for unconstrained (consumer) video. It is important to note that features must be repeatably detectable in each video as well as localizable to within one pixel (to reduce error present in later calculations), although detection itself may be performed using a different method for each video; for example, one input video may be captured using an infrared device while another may be captured using a visible light sensor, as long as a number of the features detected from each view are also detected in the other. The number of observable depth layers in each video is at most equal to the number of unique features shared with other videos, so it is most important that features be as robust as possible to varying capture conditions.

Once features have been detected, their position over time is stored as a “feature trajectory” (a sequence of homogenous row vectors of the form $(x, y, 1)$, as in the motivating paper by Caspi and Irani⁷). These trajectories are then matched across views, either by feature match (*e.g.*, small Euclidean distance between feature appearance vectors) or by curve match (*e.g.*,

number and spacing of curvature extrema). Feature matching is preferred, as changes in viewpoint may make curvature identification difficult and prone to aliasing error.

Next is to determine the temporal relation between each pair of videos - that is, to determine the parameters α and β such that the n -th frame of the first video corresponds temporally to the $(\alpha n + \beta)$ -th frame of the second video.[†] For each trajectory, we solve the following minimization:

$$\operatorname{argmin}_{\alpha, \beta} \sum_{n=1}^{\min(|N_1|, |N_2|)} \vec{p}_1(n) \mathbf{F}_{\alpha, \beta}(\vec{p}_1, \vec{p}_2) \vec{p}_2^T(\alpha n + \beta) \quad (1)$$

where N_1 is the set of all integral frame indices present in the first video (indexed starting at 1), N_2 is the set of all integral frame indices present in the second video (indexed starting at 1), $\vec{p}_1(n)$ is the position vector of the trajectory in the first video at frame n , $\vec{p}_2(\alpha n + \beta)$ is the position vector of the trajectory in the second video at frame $(\alpha n + \beta)$, and $\mathbf{F}_{\alpha, \beta}(\vec{p}_1, \vec{p}_2)$ is the fundamental matrix relating \vec{p}_1 and \vec{p}_2 given frame-rate ratio α and initial offset β to determine correspondence; the position of the trajectory in each view at nonintegral frame entries is found by interpolation from neighboring values.[‡] α and β may be discretized coarsely to further reduce the cost of computing this minimization, at the added risk of selecting incorrect temporal parameters for some trajectories. Furthermore, nonexhaustive approaches such as that of Carceroni *et al.*⁸ are recommended if it is feasible to assume enough stationary points to recover epipolar geometry.

Once each trajectory has selected its set of parameters, the final assignment of parameters α and β for the pair of videos is found by selecting the set that is most common among the detected trajectories,[§] then discarding all trajectories that produced other alignment parameters. At this point, feature trajectories have been identified and their respective fundamental matrices computed (as part of determining temporal alignment parameters), so processing can proceed to depth layer identification.

2.2 Depth layers from graph clustering

The second major stage of processing is to produce a model of trajectory relations in order to identify natural separations into approximate depth layers. Similar to the idea of clustering features based on disparity in a traditional stereo setup (with a fixed baseline placed perpendicular to the camera focal rays and with equal focal lengths), it is possible to cluster feature trajectories in a general overlapping-view arrangement using their position data and the fundamental matrix that relates their appearance in each view. Specifically, we define an error function $Err(\vec{p}, \vec{p}')$ for any two trajectories \vec{p} and \vec{p}' (appearing in at least two input videos) such that

$$Err(\vec{p}, \vec{p}') = \sum_{n=1}^{\min(|N_1|, |N_2|)} \|\vec{p}_2(\alpha n + \beta) - (\mathbf{F}_{\alpha, \beta}(\vec{p}_1, \vec{p}_2) \vec{p}_1^T(n))^T\|^2 \quad (2)$$

where all terms are defined as for (1). Of note is that $Err(\vec{p}, \vec{p}) = 0$, and that both $Err(\vec{p}', \vec{p})$ and $Err(\vec{p}, \vec{p}')$ are strictly monotonically increasing with “shared depth,” where shared depth is defined as distance along a ray that is: 1.) originating at the intersection of the two camera focal planes, and 2.) in a direction perpendicular both to this intersection and to the line formed by joining their focal points.

Using this error function, then, we construct a graph G with the following properties:

[†]This parameterization makes the assumption that the frame rate of each video remains constant during capture, although the ratio of any two frame rates is not required to be 1. Alternate parameterizations may be substituted if capture conditions should vary.

[‡]While nonlinear interpolation may produce a better approximation of interframe trajectory motion, linear interpolation was seen to be sufficient for all blur-free motion in our experiments. Alternate interpolation methods may be substituted here depending upon specific capture scenarios.

[§]Caspi and Irani⁹ suggest a “support” voting procedure instead, although our experiments indicate that it is not feasible to determine trajectory support in a scenario-unspecific way using their method. Specifically, values of ϵ used to produce correct alignment parameters in experiments upon benchmark data varied by up to two orders of magnitude depending upon the selection of videos to be aligned. Our use of a supportless voting scheme produced correct temporal alignment and acceptable depth map results in our initial experiments (see Section 3 for details), although others may choose to replace it with the more computationally intensive “support” system if ϵ can be reasonably approximated with minimal additional cost.

- There is a vertex $V_{\vec{p}}$ for each trajectory \vec{p} that appears in both videos
- There is a directed edge between each ordered pair of vertices, including self-loops
- For all \vec{p} and \vec{p}' , the edge $E_{\vec{p}, \vec{p}'}$ (from $V_{\vec{p}}$ to $V_{\vec{p}'}$) has weight $e^{-\frac{1}{2}Err(\vec{p}, \vec{p}')^2}$

This particular assignment of weights ensures that the probability density function for depth colabelling[†] is a Gaussian distribution based upon our error function and having identity covariance. While this method may lead to a reasonable assignment of trajectories to depth layers, the use of an identity covariance matrix is unlikely to provide a good fit for the data (as it makes a strong assumption about the relative scale of error readings).[‡] To rectify this problem, we perform graph clustering to provide a better approximation of relative label assignment.

One of the most computationally simple and elegant approaches to graph clustering is that of Markov Clustering (MCL),¹⁰ performed on the adjacency matrix A . The intuition behind this approach is to normalize the columns of the adjacency matrix of G and observe natural partitions in the space of random walks of progressive length.¹¹ Clustering is performed by alternating the following three steps:

1. **Inflation** - Adjacency matrix entries are raised to a small constant power (frequently 2 for computational simplicity).
2. **Expansion** - The adjacency matrix is multiplied by itself to indicate increased walk length.
3. **Normalization** - Each column of the adjacency matrix is divided by the column's sum.

These steps are repeated until the matrix has converged to become roughly idempotent.[§] To further reduce the computational expense of this clustering step, it is possible to limit iteration to a small constant number of attempts, with a binary threshold placed upon the resulting adjacency matrix as an approximation of continued processing.

After clustering is complete, any two arbitrary trajectories \vec{p} and \vec{p}' are considered to be in the same depth layer if $A(V_{\vec{p}}, V_{\vec{p}'}) > 0$ or if there exists an undirected path between $V_{\vec{p}}$ and $V_{\vec{p}'}$ (where “undirected” indicates that an edge between arbitrary nodes V_i and V_j exists if $A(V_i, V_j) > 0$ or $A(V_j, V_i) > 0$). If quantitative depth is desired, it is now possible to solve for stereo for a single feature from each layer; whether or not this has been performed, processing can now continue to final depth map creation.

2.3 Segmentation by motion similarity

Our method is concluded by identifying a depth assignment for each input pixel based upon its motion and the motion of each observed trajectory. In strictest terms, the fact that a pixel appears to move identically to a given trajectory (in a single view) does not indicate that the pixel really does match the 3D trajectory. In fact, there are an infinite number of 3-dimensional trajectories that will have the same 2-dimensional projection in the view due to projective ambiguity. However, for two arbitrary trajectories \vec{p} and \vec{p}' that have been identified as being located at different depths, projective ambiguity in an input view can only occur when

$$\forall n \in \{1, \dots, \min(|N_1|, |N_2|)\} \quad \vec{p}_1(n) - \vec{p}_1(n-1) = \vec{p}'_1(n) - \vec{p}'_1(n-1) \quad (3)$$

[†]Two trajectories are considered “colabelled” if they are considered to be at approximately the same shared depth plane within the bounds of computational precision. By using colabelling, we are able to identify natural emergent depth layers without introducing additional model parameters.

[‡]Instead of $e^{-\frac{1}{2}Err(\vec{p}, \vec{p}')^2}$, we also considered and tested situations where edge weights were strictly proportional to $Err(\vec{p}, \vec{p}')$, with correct results for fixed-baseline stereo arrangements and limited repeatability otherwise. In the case of fixed-baseline stereo, such a weighting is identical to the appearance of disparity increase due to distance, as arbitrary translation is linearly scaled according to distance from the baseline.

[§]In this case, “rough” idempotency occurs when the difference between the adjacency matrix after n iterations and the adjacency matrix after $(n+1)$ iterations is less than a small constant ρ (with this or another constant also used for thresholding the final matrix). While this threshold introduces an additional parameter into our model, we found that it required a very small number of iterations (typically less than ten) to produce changes of a scale imperceivable with floating point precision.

where all terms are defined as for (1). This means that motion at a greater depth must necessarily be of a greater magnitude in order to match the projection of the closer motion (“depth-size” ambiguity). Given trajectories of significant length (and, more frequently, of varying direction), however, such ambiguity is uncommon as even a single frame’s variation can dispel it. Therefore, we make use of the matching of single-view pixel motion to trajectory motion in order to assign pixels to depth layers quickly. Specifically, we set a pixel’s depth labelling to be the same as the trajectory whose motion it best matches. This depth can be computed for each pixel location x by solving the following minimization exhaustively:

$$\operatorname{argmin}_{\vec{p} \in \mathbb{P}} \sum_{n=2}^{\min(|N_1|, |N_2|)} \|\vec{c}_1(x) - \vec{c}_n(x + (\vec{p}_1(n) - \vec{p}_1(1)))\| \quad (4)$$

where terms are defined as for (1), \mathbb{P} is the set of all trajectories from the two input views, $\vec{c}_1(x)$ returns the color (*i.e.*, pixel vector) for location x in the first frame, and $\vec{c}_n(x + (\vec{p}_1(n) - \vec{p}_1(1)))$ returns the color for location $(x + (\vec{p}_1(n) - \vec{p}_1(1)))$ in the n -th frame.[†] This minimization will not work with very large homogenous regions if the number of frames observed is small (as internal pixels may match several trajectory motions due to failed localization), although such regions can be identified by observing pixels that have multiple near-minimizing trajectories in (4). Once these regions have been identified, their label assignment may be performed pixelwise by copying from a neighboring pixel (outside of the remaining ambiguous region) that has the smallest color difference across all frames. Given sufficiently lengthy trajectories, however, such label-confusion should be infrequent and easily rectified.

Further reductions in computational complexity can be found by restricting the set \mathbb{P} to contain a smaller set of “representative” trajectories[‡] from each depth layer. To make this computation even faster, we also suggest restricting the summation to a minimal distinguishing subset of framespans.[§] Finally, sampling may be performed on the space of frame-to-frame trajectory motions, replacing the summation in (4) with a summation over a subset of the indices; doing so introduces the possibility of incorrectly classifying pixels due to motions that appear ambiguous in the sampled space, although rejection sampling methods can eliminate this problem at the cost of acceptance checks. (4) is written prior to these optimizations, with the number of computations still less than computing pixelwise stereo via epipolar geometry as long as the number of trajectory components checked is less than the length of the visible portion of each epipolar line.

We encourage adopters of our method to determine if further depth map constraints (such as border smoothness or minimum size of connected components) are necessary for their applications, and to compare their results against those in the next section.

3. RESULTS

Initial experiments were performed against short videos produced with frames from the *Tsukuba* dataset^{††} and the Middlebury *Aloe* dataset¹² as well as sample “consumer” videos we produced, where the correct spatial and temporal alignment of the two views was known throughout. Resolution was set at 384 by 288 pixels for novel videos and 550 by 640 pixels for benchmark videos, with frame rates set to 30 fps. Frame offset was varied over each experiment in order to confirm all observations.

[†]Given a reasonable approximation of the capture geometry, calculable via a variety of well-established methods, it is possible to introduce a second term in (4) to capture the matching of pixel motion in both views rather than just the first. This adds a significant computational overhead (likely to swamp the other benefits of our method), although it ensures that all trajectory matches will be guaranteed unambiguous if the two views are not parallel.

[‡]Ideally, near-identical trajectories should be culled from each layer (so as to maximize the information present in the motion variations from frame to frame). Identifying the optimal subset of trajectories, however, would introduce an unacceptable computational penalty, although reasonable approximations based on general trajectory properties (*e.g.*, piecewise average direction) are possible.

[§]Rather than searching for motion similarity between each pair of frames, only a small number of these framespans are really needed, as they indicate the highest-magnitude variations between trajectories. As with selecting representative trajectories, however, the process of identifying important framespans may prove too computationally expensive (although approximations may prove effective).

^{††}The *Tsukuba* stereo benchmark dataset, produced by the University of Tsukuba (Tsukuba, Ibaraki Prefecture, Japan), comprises a series of static images taken at fixed horizontal displacements with all other camera parameters (including resolution and focal length) remaining fixed. Two videos were created by assigning left-shifted images as frames in the first view and right-shifted images as frames in the second view, with unambiguous nonzero aperiodic motion (Figure 4).

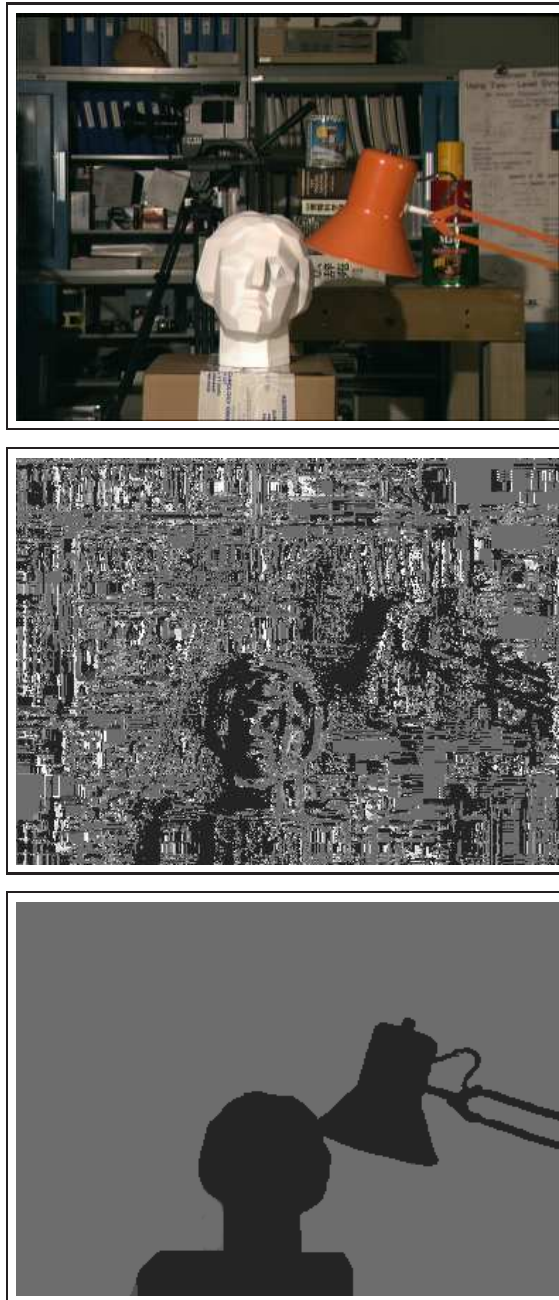


Fig. 2. *Tsukuba* Depth Map Results: an original frame is shown here (**top**), along with its calculated qualitative depth map (**middle**) and ground truth (**bottom**). Incorrect pixel classification along the right portions of the statue and lamp fixture are due to motion confusion arising from homogenous coloration and small linear motion. White pixels indicate regions where motion is exactly ambiguous between foreground and background trajectories. These misclassifications and ambiguities would be dismissed by the presence of longer trajectories (thereby making the projective difference more noticeable) or nonparallel motion (thereby making pixel matching less prone to aliasing and problems arising from color homogeneity).

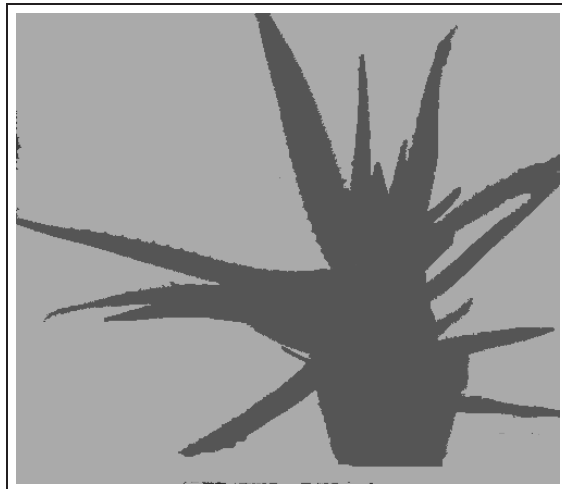
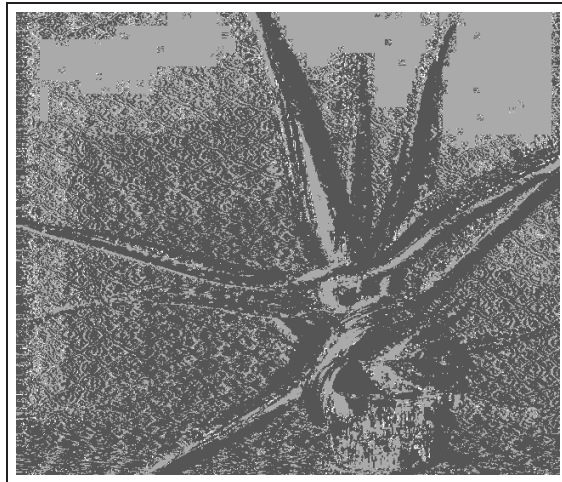


Fig. 3. *Aloe* Depth Map Results: an original frame is shown here (**top**), along with its calculated qualitative depth map (**middle**) and ground truth (**bottom**). White pixels, as in Figure 2, represent those regions where layer assignment is exactly ambiguous. Incorrect labeling due to color homogeneity, as noted with the *Tsukuba* dataset, is made much worse by miniscule purely-horizontal camera motion, MPEG artifacting, and the presence of repeating texture (resulting in poorer trajectory availability due to feature uniqueness constraints). We suggest the use of traditional stereo approaches under such circumstances, as depth map quality is significantly degraded.

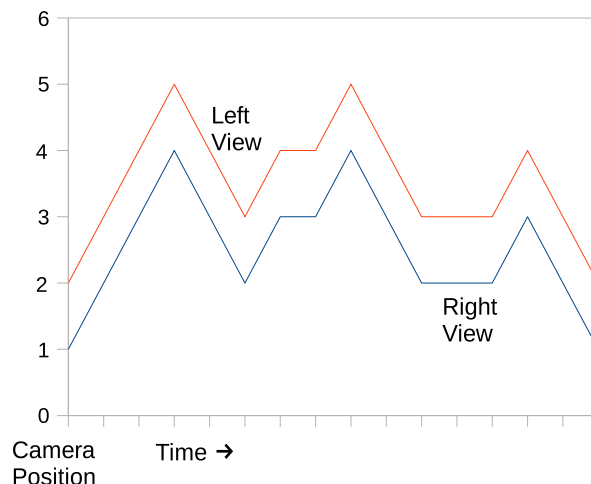


Fig. 4. Video Construction: still images from the *Tsukuba* and *Aloe* image datasets, taken in parallel directions at known locations and with constant fixed distance between consecutive positions, were used to produce stereo videos with fixed baseline and a single optimal temporal alignment. For one such video, the plot of camera position over time is shown above.

Figure 2 illustrates features from two depth layers in a frame taken from the *Tsukuba* dataset as well as an example of how the final depth map compares to the input data. Labeling noise is attributed to poor motion matching due to nearly (but not completely) color-homogeneous regions and the absence of nonparallel scene motion. Similar results are shown for the *Aloe* dataset in Figure 3, and the construction of the videos used is explained in Figure 4. Finally, Figure 5 provides an example of depth layer identification in low-resolution consumer video, where all foreground objects (*i.e.*, the person’s hand, the pen, the small pendulum) are identified as belonging to a foreground depth layer.

A variety of modern feature detection methods were implemented for trajectory detection. The SIFT approach¹³ (with difference-of-Gaussians keypoint detector and weighted local descriptor) produced the largest number of repeatable, localizable, cross-view features (approximately 800 features per *Tsukuba* or *Aloe* frame, with approximately 150 matched trajectories for one-second videos), although the Harris-Affine¹⁴ and SURF¹⁵ methods also produced a reasonable number of features. We attempted to remove the “2NN” ratio restriction (in order to increase the number of detected features) without introducing the low-velocity assumption necessary for compensatory local search,¹⁶ although all attempts resulted in an unacceptably high number of trajectory mismatches and a general failure to identify correct temporal alignment parameters, so the restriction was included for all of our experiments. These findings were repeatable for a small number of consumer videos with restricted capture conditions (*i.e.*, static or free-moving camera, static background, one to three moving foreground objects), although videos with fast camera motion or large numbers of small moving objects (*e.g.*, spilling marbles) frequently did not produce enough matchable trajectories for our method due to motion blur or ambiguity of coloration in cluttered areas. These problems could be eliminated by making use of capture devices with higher spatiotemporal resolution and improved temporal aliasing.

As stated before, the number of simple algebraic computations required for processing varies strongly with scene complexity, with the fundamental matrices (used to relate trajectories between views) already necessarily calculated for the determination of temporal alignment. Markov Clustering has its complexity tied to adjacency matrix multiplication, so the number of computations performed to achieve emergent layer identification is (with a naïve implementation) cubic in the number of trajectories, typically negligible compared to the fundamental matrix determination itself despite the low overhead of computing the eight variable values for each matrix using modern methods. Each final pixel assignment is linear in the number of trajectories (reduced to the number of layers if representative trajectories are used) and the number of frames that they span; as noted before, this is nearly always less than for pixelwise stereo computation and less prone to appearance ambiguity given trajectories of sufficient length. These relations were confirmed in our studies on real data; for experiments involving videos taken from the feature-rich *Tsukuba* dataset, time spent identifying temporal parameters and performing graph clustering was approximately eight times that spent on the same tasks for the feature-poor consumer

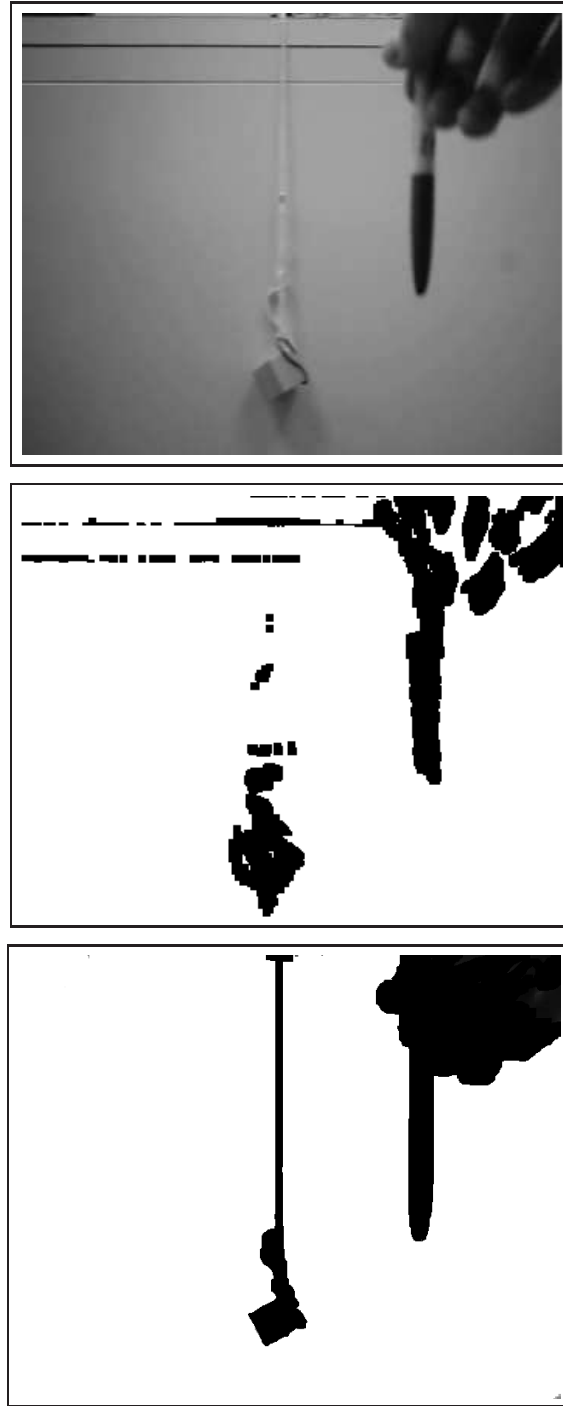


Fig. 5. Depth Layers in a Simple Consumer Video: A foreground depth layer is determined in a 5-second novel consumer video using our method, where two cameras were used with arbitrary 10° to 15° angles for relative pan and tilt and fixed 10 cm separation. Camera motion was performed by sliding the camera rig along a rough facing desk at variable speed, while scene motion included a swinging pendulum and a pen being held steady at a closer depth. A selected frame from the input video is shown (**top**), along with the corresponding foreground pixel assignment (**middle**) and ground truth (**bottom**); black pixels correspond to sections that have been labeled as foreground, while white pixels correspond to background. The misassignment of the desk creases is due to incorrect localization when performing motion matching with the pen's trajectory (where motion is almost exactly horizontal). Additional incorrect labelings are due to lighting effects (*e.g.*, shadowing, specularity, translucence) and are not addressed by our method.

videos depicted in Figure 5, where (on average) twice as many features were detected in the *Tsukuba* videos. Additionally, the ratio of time spent on other complexity-dependent tasks (*i.e.*, feature tracking, final pixel assignment) was directly proportional to the ratio of features detected, while complexity-independent tasks (*i.e.*, initial feature detection) remained approximately constant as well as stayed negligible when compared to the complexity-dependent work.

4. CONCLUSIONS

Initial results indicate that our method is indeed successful in identifying qualitative depth layers in uncalibrated multiview video, albeit with significant noise when trajectories are not sufficiently lengthy and trending towards misclassification when scene motion is purely parallel and color-homogeneous regions are uncompensated; future results will rigorously analyze the effect of increasing trajectory length and varying motion direction for benchmark and novel datasets. Computational complexity (unlike many methods in the literature) is primarily dependent upon scene complexity due to the number of feature trajectories tracked, with a number of optimizations available for consideration.

REFERENCES

- [1] Zhang, Z., Luong, Q.-T., and Faugeras, O., “Motion of an uncalibrated stereo rig: Self-calibration and metric reconstruction,” in [*Proceedings of the 12th IAPR International Conference on Pattern Recognition*], **1**, 695–697 (1994).
- [2] Furukawa, R. and Kawasaki, H., “Uncalibrated multiple image stereo system with arbitrarily movable camera and projector for wide range scanning,” in [*Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling*], 302–309 (2005).
- [3] Levin, A., Fergus, R., Durand, F., and Freeman, W., “Image and depth from a conventional camera with a coded aperture,” *ACM Transactions on Graphics* **26**(3) (2007).
- [4] Ayer, S. and Sawhney, H. S., “Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding,” in [*Proceedings of the Fifth IEEE International Conference on Computer Vision*], 774–784 (1995).
- [5] Jepson, A. D., Fleet, D. J., and Black, M. J., “A layered motion representation with occlusion and compact spatial support,” in [*Proceedings of the Fifth European Conference on Computer Vision*], 692–706 (2002).
- [6] Wang, J. Y. A. and Adelson, E. H., “Representing moving images with layers,” *IEEE Transactions on Image Processing* **3**(5), 625–638 (1994).
- [7] Caspi, Y. and Irani, M., “A step towards sequence-to-sequence alignment,” in [*Proceedings of the 2000 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*], 682–689 (2000).
- [8] Carceroni, R. L., Pádua, F. L. C., Santos, G. A. M. R., and Kutulakos, K. N., “Linear sequence-to-sequence alignment,” in [*Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*], **1**, 746–753 (2004).
- [9] Caspi, Y. and Irani, M., “Spatio-temporal alignment of sequences,” *IEEE Transactions on Pattern Analysis and Machine Intelligence* **24**(11), 1409–1424 (2002).
- [10] van Dongen, S., *Graph Clustering by Flow Simulation*, PhD thesis, University of Utrecht (2000).
- [11] van Dongen, S., “Performance criteria for graph clustering and markov cluster experiments,” Tech. Rep. INS-R0012, University of Utrecht (2000).
- [12] Scharstein, D. and Pal, C., “Learning conditional random fields for stereo,” in [*Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*], 1–8 (2007).
- [13] Lowe, D., “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision* (2004).
- [14] Mikolajczyk, K. and Schmid, C., “Scale and affine invariant interest point detectors,” *International Journal of Computer Vision* **60**(1), 63–86 (2004).
- [15] Bay, H., Tuytelaars, T., and Gool, L. V., “Surf: Speeded up robust features,” in [*Proceedings of the Fifth European Conference on Computer Vision*], **3951**, 404–417 (2006).
- [16] Hess, R. and Fern, A., “Improved video registration using non-distinctive local image features,” in [*Proceedings of the 2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*], (2007).