VIDEO-BASED RENDERING USING FEATURE POINT EVOLUTION

Wende Zhang[*] and Tsuhan Chen

Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA
{wendez, tsuhan}@andrew.cmu.edu

ABSTRACT[1]

We propose a novel video-based rendering algorithm with a single moving camera. We reconstruct a dynamic 3D model of the scene with a feature point set that "evolves" over time. As the scene's appearance changes due to camera and object motions, some existing feature points dynamically disappear while some new feature points dynamically appear relative to the camera. The newly generated feature points' 3D positions and motions are initialized using nearby existing feature points' positions and motions. Our feature evolution, when incorporated into standard tracking and 3D reconstruction algorithms, provides for robust and dense 3D meshes, and their corresponding motions. Consequently, the evolution-based, time-dependent 3D meshes, motions, and textures render good-quality images at a virtual viewpoint and at a desired time instance. We also extend the proposed video-based rendering algorithm from using one single moving camera with one reconstructed depth map to using multiple moving cameras with multiple reconstructed depth maps to avoid occlusion and improve the rendering quality.

Index Terms — stereo vision, motion analysis, 3D reconstruction, rendering, feature extraction

1. INTRODUCTION

With one or more cameras capturing a scene, the 3D modeling uses the captured images to reproduce the scene's 3D structure, or geometry [1]. Rendering an image from another viewpoint is then possible with this knowledge of the underlying 3D structure of the objects in the images [2]. In this paper, we study the rendering of dynamic scene using feature point evolution. We first overview how to model dynamic scenes. We then discuss dynamic 3D model building as well as our contribution of using dynamically appearing, evolving feature points applied to dynamic scenes in Section 2. The experimental results are discussed in Section 3. An experimental visualization using novel multi-depth-map video-based rendering is also presented. Finally, concluding remarks are given in Section 4.

Video-based rendering typically models the dynamic scene with multiple synchronized video cameras [3][4]. [3] uses a color segmentation-based stereo algorithm to estimates the scene depth with the smoothness constraint between the segments and the spatial consistence between the synchronized images. The dynamic modeling can be enhanced with multiple time frames through finding the optical flow and addressing the temporal consistence. Optical flow can be used to recover 3D scene flow and shape from multiple sparsely distributed cameras for high-quality modeling [4]. By enforcing the temporal consistence, [5] shows better reconstructed geometry for a time-varying scene surface.

To improve the standard tracking techniques, for example, the KLT tracker [6], the type of dynamic modeling of particular interest to our own work is time series modeling, such as the Kalman filter [7], the extended Kalman filter (EKF) [8], and particle filters [9] to enforce the temporal consistence. Kalman filtering has been used to track feature points in video frames for reconstructing the scene [10]. EKF allows for more complex modeling than the Kalman filter for estimating the structure and motion of a rigid object, assuming smooth motion [10]. Its use in vision has been refined, for example, by recursively recovering motion and geometry [11] and by assuming that all tracked points lie on a plane [12]. Particle filters provide an approximation to the tracking and were used by [13] in template-based tracking which adapted the number of particles used in the observation and velocity models.

For modeling scenes, the time-series-modeling-based tracking algorithms typically use a static set of feature points/patches, which may not remain reliable as the scene evolves. [13] does provide a framework for changing the size of the particle set over time; however, they do not make provisions for how to incorporate the particles/feature points which dynamically appear. [14] incorporates the new feature points which best discriminate between the changing foreground object and background to improve the object tracking performance in the 2D image. As their goal is 2D tracking and not the more demanding 3D reconstruction, their correspondences are not very precise. [15][16][17] provide different mechanisms for generating and deleting feature points as they appear or leave the scene. However, all of them focus on tracking a sparse feature point set only of dominant features for scene modeling. [15] initializes the state of each new feature point typically using the state of its single nearest neighbor.

In contrast, for high-quality rendering, our work not only tracks a dense 3D point set of both dominant and subtle features, but also initializes their underlying states (especially for subtle features) using the existing prior knowledge: the tracking results of their multiple neighboring states. The final result is the improved modeling and rendering of the 3D geometry and motions of dynamic scene from a single moving camera.

2. MODELING EVOLUTION SCENES

---

* Wende Zhang is currently with General Motors Corporation, Warren, MI.

[1] Work supported in part by General Motors Corporation and Industrial Technology Research Institute, Taiwan.

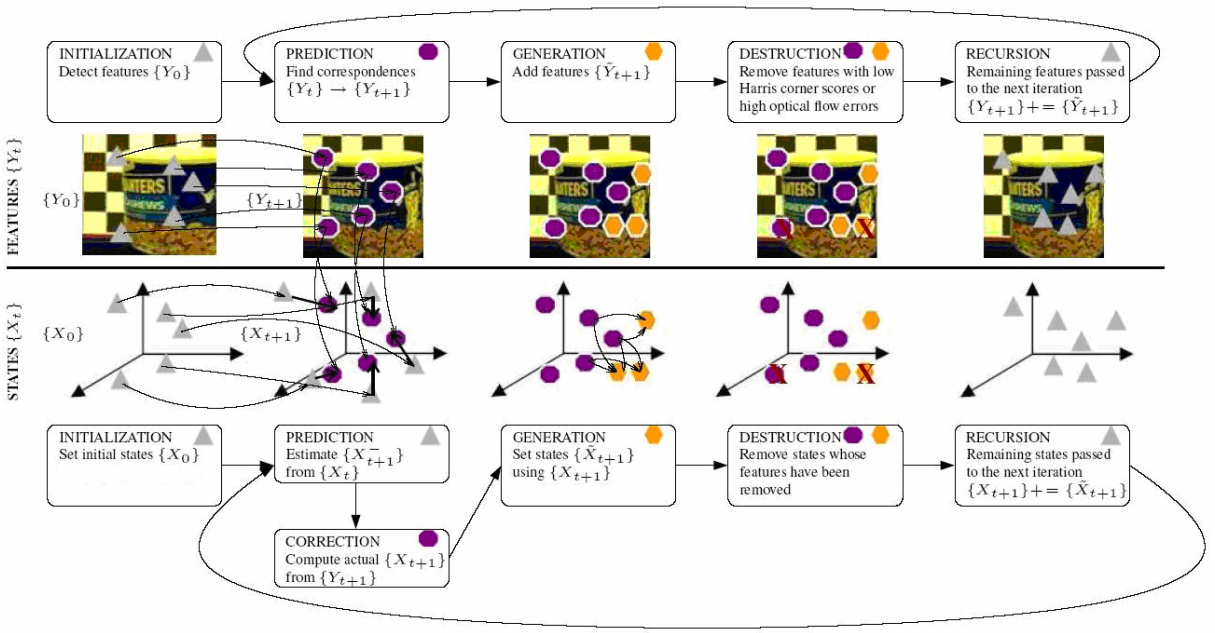


Figure 1. Evolution Flow Chart. The evolution of the features is modeled on top: images with sample feature points are marked. The respective evolution of the states is modeled on bottom: the estimated 3D positions and 3D motions of the sample feature points are plotted. INITIALIZATION, PREDICTION, and CORRECTION follow standard Kalman filtering while GENERATION and DESTRUCTION follow our proposed evolution framework.

In this section, we introduce our novel tracking algorithm for video-based rendering, and the reconstruction and rendering algorithms.

2.1. Tracking

Evolution, the core contribution of this work, is a dynamic feature point extractor embedded in standard time-series analysis (see Figure 1). As video progresses over time, certain tracked feature points (e.g., state X_t) will have noisy 2D image feature points Y_t that become difficult to track while, conversely, new feature points will appear that are robust, and easy-to-track. Hence, we only model each portion of the scene while it is easy to track. In addition to proposing which feature points (and associated states) to model at each time frame, we also propose a novel "state passing" mechanism that initializes the states of the newly generated feature points in each frame. Evolution proceeds as follows, where Steps 1, 2, & 3 below correspond to the EKF's initialization, prediction, and correction and where Steps 4 & 5 below are the key contributions of this work:

We first introduce the EKF modeling used in evolution. Let $\{X_t\}$ be the set of the states, which are 3D positions $\mathbf{p}_t$ and 3D motions $\mathbf{v}_t$, and 2D image feature point set $\{Y_t\}$ be the observations. By assuming constant velocity model for each feature point, we have

State equation: $X_{t+1} = F_t X_t + Q_t, \quad Q_t \sim N(0, q_t)$ (1)

Observation equation: $Y_t = f_t(X_t) + R_t, \quad R_t \sim N(0, r_t)$ (2)

where $X_t = \begin{bmatrix} \mathbf{p}_t \\ \mathbf{v}_t \end{bmatrix}$, $F_t = \begin{bmatrix} I & I \\ 0 & I \end{bmatrix}$, and f_t projects the feature points' position $\mathbf{p}_t$ to the current image plane with known intrinsic and extrinsic camera calibration parameters. Q_t and R_t represent the Gaussian noises in the modeling with variance q_t and r_t, respectively. In the current implementation, the noise modeling q_t and r_t are time-independent.

1. Initialization (time $t = 0$ only)
   - Find 2D feature point set $\{Y_0\}$ at time $t = 0$, using a Harris corner detector [18].
   - For each $y_0 \in Y_0$, initialize its state $x_0 \in \{X_0\}$:

$$x_0 = \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{v}_0 \end{bmatrix} \quad (3)$$

   where 3D position $\mathbf{p}_0$ is calculated by assuming a constant depth, and 3D motion $\mathbf{v}_0$ is set to zero.

2. Prediction
   - For each $x_t \in X_t$, predict x^-_{t+1} using the EKF modeling above.
   - For each $y_t \in Y_t$, find its corresponding position y_{t+1} in frame $t+1$ using the pyramid KLT tracker [6].

3. Correction
   - For the predicted state $x^-_{t+1} \in X_{t+1}$, correct its value x_{t+1} using the EKF modeling.

4. Generation
   - As in Step 1, find feature point set $\{\tilde{Y}_{t+1}\}$ using a Harris corner detector. These feature points are chosen independently of the predicted $\{Y_{t+1}\}$.
   - For each of the new feature points $\tilde{y}_{t+1} \in \{\tilde{Y}_{t+1}\}$, find its corresponding position $\tilde{y}_t$ in frame t using (reverse) pyramid KLT tracker. Let $\{\tilde{X}_{t+1}\}$ be their (un-initialized) states.
   - Initialize each new state $\tilde{x}_{t+1} \in \{\tilde{X}_{t+1}\}$. Let $\tilde{x}_{t+1}$ be initialized using weights $w_{x'_{t+1}}$ on the nearby existing states $X'_{t+1} = \{x_{t+1} \in X_{t+1} \,|\, \|y_{t+1} - \tilde{y}_{t+1}\| < Th_y\}$:

$$w_{x'_{t+1}} = \frac{\mathrm{Age}(x'_{t+1})}{\left\|\mathrm{y}'_{t+1} - \tilde{y}_{t+1}\right\| \sum_{i=t+1-\mathrm{Age}(x'_{t+1})}^{t+1} \left|E_i(x'_i)\right|^2} \quad (4)$$

$$\tilde{x}_{t+1} = \frac{\sum_{x'_{t+1} \in X'_{t+1}} w_{x'_{t+1}} x'_{t+1} + \beta X_0}{\sum_{x'_{t+1} \in X'_{t+1}} w_{x'_{t+1}}} \quad (5)$$

where $\tilde{y}_t$, y_t, and y'_t are the observations of $\tilde{x}_t$, x_t, and x'_t, respectively; Th_y is the threshold for defining a new state's neighbors in the 2D image; β is the weight on the prior state X_0; $E_t(x'_t) = \left|y'_t - f_t(x'^-_t)\right|$ is the Kalman error of the state x'_t at time t, and $\mathrm{Age}(x'_{t+1})$ is the number of frames that state x'_{t+1} has been in existence.

5. Destruction

- Define $P_t(y'_t)$ as the square patch centered at pixel y'_t in time frame t. Determine the optical flow matching error for each existing $y_{t+1} \in \{Y_{t+1}\}$ and for each new feature point $\tilde{y}_{t+1} \in \{\tilde{Y}_{t+1}\}$, respectively:

$$E_{t+1}(y_t, y_{t+1}) = \left\|P_t(y_t) - P_{t+1}(y_{t+1})\right\| \quad (6)$$

$$E_{t+1}(\tilde{y}_{t+1}, \tilde{y}_t) = \left\|P_{t+1}(\tilde{y}_{t+1}) - P_t(\tilde{y}_t)\right\| \quad (7)$$

- Define $\mathrm{HC}_t(y_t)$ as the corner score returned by the Harris corner detector for feature point y_t at time frame t.
- Destroy any existing feature point y_{t+1} or new feature point $\tilde{y}_{t+1}$ that fails either of its respective tests:

$$y_{t+1} : E_{t+1}(y_t, y_{t+1}) < Th_E \quad (8)$$

$$\mathrm{HC}_{t+1}(y_{t+1}) > Th_{HC} \quad (9)$$

$$\tilde{y}_{t+1} : E_{t+1}(\tilde{y}_t, \tilde{y}_{t+1}) < Th_E \quad (10)$$

$$\mathrm{HC}_{t+1}(\tilde{y}_{t+1}) > Th_{HC} \quad (11)$$

where Th_E is the threshold for the optical flow matching error and Th_{HC} is the threshold for the corner score.

- Let the sets of states and of feature points for the next iteration be:

$$\{X_{t+1}\} = \{X_{t+1}\} + \{\tilde{X}_{t+1}\} \quad (12)$$

$$\{Y_{t+1}\} = \{Y_{t+1}\} + \{\tilde{Y}_{t+1}\} \quad (13)$$

In summary, with evolution we detect additional new feature points in each frame. Instead of initializing the state of the newly generated feature point from scratch, we borrow information from its neighbors. Only the feature points with good 2D correspondences (large corner scores and low matching errors) can be passed on to the next iteration. Therefore, we allow those feature points with good 2D correspondence to continue in the future iterations where they would hopefully become more reliable and have a low Kalman error. We also let a feature point die if it does not have good 2D correspondence across neighboring frames since we cannot accurately reconstruct its 3D point position anyways.

2.2. Reconstruction and Rendering

Once the tracking of the evolving points is complete, the underlying states can then be used to construct 3D depth maps. First, as we are tracking the feature points using EKF, we utilize the Kalman error to remove those feature points which were poorly tracked. The remaining, reliable states (3D positions) are then used to build the time-dependent 3D mesh (depth map) using Delaunay triangulation [19] at each time instance of capturing. Finally, the meshes are deformed locally in time based on the reliable states' motions for rendering at a desired time instance. Given the deformed meshes and textures from the captured images, we were able to render the scenes at the desired time instance.

3. EXPERIMENTS

We first show the experimental results of our video-based rendering with a single moving camera. A better visualization using novel multi-depth-map video-based rendering with multiple moving cameras is also presented.

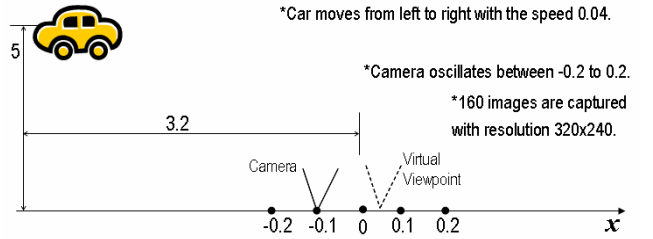


Figure 2. Experiment setup of a moving car

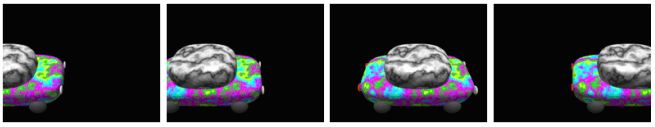
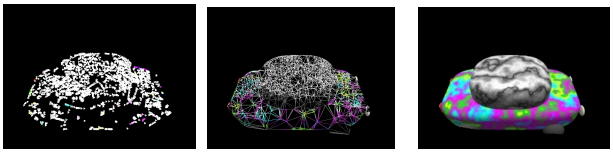
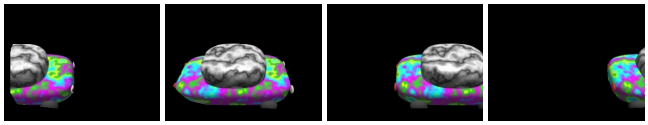


Figure 3. Sample input images of a moving car



(a) 3D points (b) 3D mesh (c) Rendered image

Figure 4. The reconstruction of dynamic scene



(a) Frame 35 (b) Frame 55 (c) Frame 115 (d) Frame 155

Figure 5. The proposed video-based rendering at virtual viewpoint

We first ran experiments on synthetic data with a single moving camera. A moving toy car was captured by an oscillating camera simulated by POV-ray [20]. The car moved with the speed of 0.04 from left to right. The camera oscillated between -0.2 to 0.2, and was 5 away from the car center vertically. We rendered the scene activities at the virtual viewpoint x = 0.05 as shown in Figure 2. As illustrated in Figure 3, a total of 160 images were captured at a low resolution of 320x240 pixels with the known intrinsic and extrinsic camera calibration parameters as the input image sequence. As shown in Figure 4, we first reconstructed the feature points of the scene using our proposed evolution algorithm of Section 2.1. Based on the reconstructed feature points' positions and motions in space, we then built the triangle mesh of the scene at the desired time instance. Finally, we rendered the scene at a novel virtual viewpoint (see Figure 2) using the time-dependent meshes and textures from the captured images, as explained in Section 2.2. We showed that evolution had a good rendering quality in the experiment, for example, the car's geometry and motion were well rendered as shown in Figure 5. A video with both capturing, rendering and detail parameter settings can be downloaded at

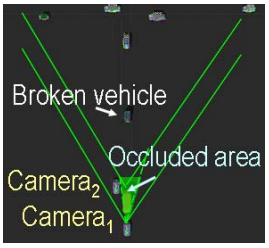
ftp://amp.ece.cmu.edu/Outbox/ICIP2006/SynMoveCar.zip.

The rendering results in Figure 4 and Figure 5 will be bad without the feature point evolution involved, since the object appearance relative to the camera changes dramatically between the first frame and the last frame as shown in Figure 3.

We also extended our video based rendering to using multiple moving cameras to reconstruct multiple depth-maps for better driver visualization. We can attach one camera to vehicle's moving wind-shield wiper. POV-ray [20] is used again to simulate the oscillating camera to capture the street scene as shown in Figure 6(b). The driver can visualize the dynamic street scene at a higher virtual viewpoint using the proposed video-based rendering algorithm with a single moving camera. However, due to the occluded area blocked by the school bus ahead, the rending result Figure 6(c) at a higher virtual viewpoint did not render the broken vehicle existing in the scene as shown in Figure 6(a), which is important information to avoid a potential accident.

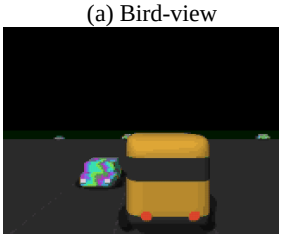
(a) Bird-view (b) Input image (c) Rendered image

Figure 6. Visualization with a single camera

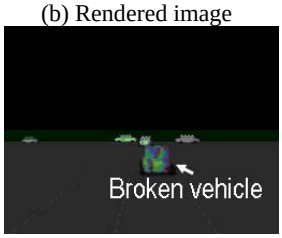


(a) Bird-view

(b) Rendered image



(c) Captured from Camera$_1$



(d) Captured from Camera$_2$

Figure 7. Visualization with two cameras

Since vehicles can communicate between each other in future, with the multiple-depth-map modeling, we can have a better visualization using neighboring vehicles' cameras. In this example, we simply composed the textures, the resulting depth maps reconstructed by current vehicle (Figure 7(c)), and those reconstructed by the school bus ahead (Figure 7(d)). As shown in Figure 7(a), we had less occluded area compared to Figure 6(a). Our rendering result at a higher viewpoint rendered the broken vehicle as shown in Figure 7(b), which was a better visualization. The dynamic rendering results (video) can be downloaded at ftp://amp.ece.cmu.edu/Outbox/ICIP2006/SynMultiCam.zip.

4. CONCLUSIONS

We proposed a feature point evolution algorithm for dynamic scene reconstruction that exploits the characteristics of dynamic, evolving scenes. The result is an evolution model dealing with video-based rendering using a moving camera. We also proposed the video-based rendering with the multi-depth-map modeling using multiple moving cameras to have a better rendering quality.

REFERENCES

[1] C. Zhang and T. Chen, "A Survey on Image-Based Rendering – Representation, Sampling and Compression", *EURASIP Signal Processing: Image Communication*, Vol. 19, pp. 1-28, Jan. 2004.

[2] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen, "Unstructured Lumigraph rendering", *Computer Graphics (SIGGRAPH'01)*, pp 425-432, Aug. 2001.

[3] C.L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder and R. Szeliski, "High-quality video view interpolation using a layered represenation," *Computer Graphics (SIGGRAPH)*, Aug. 2004.

[4] S. Vedula, S. Baker, and T. Kanade, "Image-Based Spatio-Temporal Modeling and View Interpolation of Dynamic Events," *ACM Transactions on Graphics*, Vol. 24, No. 2, April, 2005.

[5] B. Goldluecke, M. Magnor, "Spacetime-continuous Geometry Meshes from Multi-view Video Sequences," *Proc. IEEE International Conference on Image Processing (ICIP'05)*, Genoa, Italy, 2005.

[6] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," In *Proc. of International Joint Conf. on Artificial Intelligence (IJCAI '81)*, pages 674-679, Vancouver, April 1981.

[7] R.E. Kalman. "A new approach to linear filtering and prediction problems," *Transactions of the ASME: Series D, Journal of Basic Engineering*, 82:35.45, March 1960.

[8] H. Cox, "On the estimation of state variables and parameters for noisy dynamic systems," *IEEE Trans. on Automatic Control*, 9(1):5.12, 1964.

[9] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197-208, 2000.

[10] T. Broida, S. Chandrasekhar, and R. Chellappa, "Recursive 3-D Motion Estimation from a Monocular Image Sequence," *IEEE Transactions on Aerospace and Electronic Systems* AES-26(4): pp 639-655, 1990.

[11] A. Azarbayejani, A.P. Pentland. "Recursive Estimation of Motion, Structure, and Focal Length," *IEEE Transactions on PAMI*, vol. 17, no. 6, pp. 562-575, June 1995.

[12] J. Alon and S. Sclaroff, "Recursive Estimation of Motion and Planar Structure," in Proc. of *IEEE Conf. on CVPR*, Vol. 2, pp 550-556, 2000.

[13] S. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Transactions on Image Processing (IP)*, Vol. 11, pp. 1434-1456, 2004.

[14] R. Collins, Y. Liu, and M. Leordeanu, "On-line selection of discriminative tracking features," *IEEE Transaction on PAMI*, 27(10), pp 1631-1643, October 2005.

[15] M. Trajkovic and M. Hedley, "Robust Recursive Structure and Motion Recovery under Affine Projection", *Proc. British Machine Vision Conference-97*, Essex, UK, September 1997.

[16] Y. S. Yao and R. Chellappa, "Tracking a Dynamic Set of Feature Points," IEEE Trans. Image Processing, Vol. 4, No. 10, 1995.

[17] A. J. Davison and D. W. Murray, "Simultaneous Localization and Map-Building Using Active Vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):865—880, 2002.

[18] C. Harris and M. Stephens, "A combined corner and edge detector," In *4th Alvey Vision Conference*, pp. 189-192, 1988.

[19] A. Okabe, B. Boots, and K. Sugihara, *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams,* New York, Wiley, 1992.

[20] Persistence of vision ray tracer (POV-Ray). http://www.povray.org/.