Virtual Object Placement in Video for Augmented Reality

Jong-Seung Park¹, Mee Young Sung¹, and Sung-Ryul Noh²

¹ Department of Computer Science & Engineering, University of Incheon, 177 Dohwa-dong, Nam-gu, Incheon 402-749, Republic of Korea
{jong, mysung}@incheon.ac.kr

² SIRIUS, IVC 3F, 169-1 Juan-dong, Nam-gu, Incheon 402-201, Republic of Korea
noh@sirius.co.kr

Abstract. This article describes a method to insert virtual objects into a real video stream based on feature tracking and camera pose estimation from a set of single-camera video frames. To insert or modify 3D shapes to target video frames, the transformation from the 3D objects to the projection of the objects onto the video frames should be revealed. It is shown that, without a camera calibration process, the 3D reconstruction is possible using multiple images from a single camera under the fixed internal camera parameters. The proposed approach is based on the simplification of the camera matrix of intrinsic parameters and the use of projective geometry. The method is particularly useful for augmented reality applications to insert or modify models to a real video stream. Several experimental results are presented on real-world video streams, demonstrating the usefulness of our method for the augmented reality applications.

Keywords: video editing, metric reconstruction, texture blending, feature tracking.

1 Introduction

A flexible synthesis of real environments with virtual objects is interested in wide range of augmented reality applications. Methods of virtual view generation can be classified into two categories: In the first category, a full 3D structure of the scene is constructed and then reprojected in order to generate a virtual view[1]. The main issue in this approach is the problem of generating a full 3D model from 2D information. Though several novel methods have been presented based on multiple view geometry[2], the 3D reconstruction problem is still an ill posed problem. In the second category, virtual views are directly generated without having to estimate the scene structure[3]. This approach reconstructs virtual views from a set of reference views without concerning the geometric structure. However, these approaches require a considerable amount of computational cost to compute dense correspondences and also to generate virtual views from a moving user-defined viewpoint.
This paper describes a method to generate real-time virtual views in which real objects observed by a camera are replaced with virtual objects. Our approach to the virtual view generation falls into the first category which requires relatively cheap computational cost. To avoid numerical instability of 3D reconstruction, our method finds and tracks only a moderate number of apparent feature points.

Two main subtasks are camera pose estimation relative to the real object and seamless video blending. Camera pose estimation is to find the camera positions and orientations relative to the target object. In typical augmented reality applications, there are two types of camera pose estimation: marker-based approaches and motion-based approaches: Marker-based approaches utilize a simple black and white marker for easy detection and tracking of the marker area. In these kind of applications, the marker plays a key role for the camera pose estimation. On the other hand, motion-based approaches track many feature points and recover camera motion parameters from the point correspondences. The current state of the art technology provides solutions that can be applied only under strict conditions. A common constraint is to assume that all the images are taken by exactly the same camera, without change of focus or zoom. Existing theory and algorithms have been restricted to constant camera parameters. The image axes can be assumed orthogonal and often the aspect ratio is known.

In the past few years some preliminary research results have been presented on the 3D construction using single-camera images. Recently, Kahl presented a method to model smoothness constraints about the random camera motion such as the motion of hand-held video cameras. With developing a maximum a posteriori (MAP) estimator, a way to estimate both Euclidean structure and motion was proposed. Heyden proposed a method to improve the estimation quality of the absolute quadric refined by a nonlinear technique and more accurate Euclidean reconstruction.

A video blending task is related to the overlay of virtual objects, replacing real objects with virtual objects, or removing real objects. Blending approaches should consider the scene depth structure in composition to become accepted the real and virtual objects for augmented reality applications. Moreover, it is preferred to figure out and reconfigure the illumination conditions so that the virtual objects and the real objects share a common lighting environment. Though there have been some works on the geometry of the light sources, determining reflectance and light position is still a difficult problem in the image analysis field.

Our 3D reconstruction system consists of three steps: feature tracking, metric reconstruction, and object insertion. Fig. 1 illustrates the overall steps of our method. First, good features are selected and tracked for input video frames. Good features include most corner points of real objects and any other points that can be reliably tracked. Second, camera poses are estimated using the feature correspondences. Finally, virtual objects are inserted to target positions according to the camera poses. The target positions are identified from the user initialization of object corners at the first frame.
Fig. 1. Steps of our video synthesis system

This paper is organized as follows. A feature tracking and camera parameter estimation method is described and the algorithm for 3D reconstruction is detailed in Section 2. In Section 3, we explain the object insertion and texture blending process. The last two sections give experimental results and conclusion.

2 Object Pose Estimation

The video analysis process to insert a virtual object into a video stream is one of formidable tasks in augmented reality applications. The difficulty comes mainly from the fact that little information is available at the beginning process. Except for a few restrict assumptions such as a basic pinhole camera model, rigid scene, and diffuse reflectance characteristics of the surface, neither the world geometry nor the camera geometry is known.

To insert a virtual object, we must know where the real world objects lies. The camera pose tracking is a problem of obtaining the relative camera-to-object position and orientation. With a prior knowledge of model geometry, it can be done by simply tracking feature points. In our approach, the model geometry is assumed to be unknown. The tracking features do not have to lie on a specific real object. The only assumption is that initial reference positions are assigned by the user at the first image frame to indicate the target place of object insertion. For each frame, the camera pose is computed only from feature correspondences without using any model geometry.

2.1 Feature Tracking

We compute the object pose relative to the camera coordinate system using feature correspondences between two images. However, reliable pixel correspondences are difficult to obtain, especially over a long sequence of images: Region tracking using motion segmentation methods often fails to detect the motions of
low textured region\cite{11}. Feature-tracking techniques often fail to produce correct matches due to large motions, occlusions, or ambiguities. Furthermore, errors in a frame are likely to propagate to all subsequent frames. However, there are some clues to break through the difficulties. Outlier rejection techniques can reduce these problems. Knowledge of camera parameters or epipolar geometry can simplify the correspondence problem.

A valid and powerful assumption is the fact that the motion between frames is generally small. In this case, the feature coordinates and the intensity distribution around the features will be alike in two images. Besides, if the feature to be matched is remarkably different from its surroundings, it is allowed to reduce the search complexity and to match features using intensity cross correlation. Cross correlation is a well known method to estimate how two patterns are correlated. For each feature point \( m_1 \) in the first image \( I_1 \), we use a correlation window of size \((2N + 1) \times (2M + 1)\) and perform a correlation operation between \( m_1 \) and all features \( m_2 \) lying within a search area in the second image \( I_2 \). The correlation score is evaluated as

\[
s(m_1, m_2) = \sum_{i=-N}^{N} \sum_{j=-M}^{M} \left[ g_{ij}(m_1) - \bar{g}(m_1) \right] \left[ g_{ij}(m_2) - \bar{g}(m_2) \right] / dan{1}
\]

where \( d = (2N + 1)(2M + 1) \sqrt{\sigma^2(m_1)\sigma^2(m_2)} \), \( g_{ij}(m_k) \) is the intensity at point \( m_k + (i, j) \) in image \( I_k \), \( \bar{g}(m_k) \) is the average intensity at point \( m_k \) of \( I_k \), and \( \sigma^2(m_k) \) is the standard deviation of \( I_k \) in the neighborhood of \( m_k \). Based on the correlation score and spatial consistency, we obtain a large enough number of accurate feature correspondences between two consecutive image frames.

### 2.2 Obtaining Camera Parameters

When a camera moves, the camera coordinates of scene points are also changed\cite{12}. A camera movement causes unknown camera extrinsic parameters to be considered: \( R \) for camera rotation and \( t \) for camera translation. The reconstruction problem is directly related to the computation of these camera parameters. When all the parameters of the camera are given, the reconstruction is straightforward. Many of the previous works assume that the intrinsic camera parameters are known. Computing camera motion with known intrinsic parameters is the well-known relative orientation problem and several effective methods are available\cite{12}. In many practical augmented reality applications, however, assuming fixed intrinsic parameters or off-line calibration is not suitable. Hence, we are going to focus on the case of a single camera with unknown intrinsic parameters and unknown motion. The only available information is the video sequence.

Before computing metric calibration, an affine calibration is computed first, or equivalently, the plane at infinity is located first in the projective space, which is observed to be the most difficult step. Armstrong et al.\cite{13} obtained the affine calibration using some views of pure translations. With additional views of general motions, a metric calibration can be easily obtained. Hartley\cite{14} proposed an algorithm by using dense search. Pollefeys and Gool\cite{15} show
a nonlinear algorithm based on modulus provides one polynomial equation of degree four in the coefficients of the infinity plane.

A 3D point \( M \) is projected into an image point \( m \) which is the intersection of the retinal plane with the line passing \( M \) and the optical center. Let \( X = (X,Y,Z) \) be the coordinates of \( M \) in the world coordinate system and \( x = (u,v) \) the pixel coordinates of \( m \). Let \( \tilde{X} \) and \( \tilde{x} \) be the homogeneous notations of \( X \) and \( x \), respectively. Then, the transformation from \( \tilde{X} \) to \( \tilde{x} \) is given by \( \lambda \tilde{x} = \tilde{P} \tilde{X} \) where \( \lambda \) is an arbitrary nonzero scalar. The \( 3 \times 4 \) matrix \( \tilde{P} \) is called the camera matrix. Using a QR factorization, the camera matrix \( \tilde{P} \) can be decomposed into the product form: \( \tilde{P} = K[R|t] \). The \( 3 \times 3 \) matrix \( K \) has five unknowns: the focal lengths in two image directions \((\alpha_x, \alpha_y)\), the principal point in terms of pixel dimensions \((x_0, y_0)\), and the skew parameter \((s)\) which is close to zero in most cases. The matrix \( K \) depends on the intrinsic parameters only. The extrinsic parameters represent the rigid transformation that aligns the camera reference frame and the world reference frame and they are encoded by the rotation matrix \( R \) and the translation \( t \).

Auto-calibration is the process of determining both internal parameters and external parameters directly from multiple uncalibrated images. The general approach has two steps: First obtain a projective reconstruction \((P^i, X^j)\). Then determine a rectifying homography \( H \) from auto-calibration constraints, and transform to a metric reconstruction \((P^i H, H^{-1} X^j)\). We assume that there are \( m \) cameras with projection matrices \( P_M^i \), \( i = 1, \ldots, m \). The coordinates of \( 3D \) points in Euclidean world frame are denoted by \( X_M^j \), \( i = 1, \ldots, n \). Then, the \( i \)'th camera projects \( X_M^j \) to an image point \( x^i = P_M^i X_M^j \). The calibrated cameras may be written as \( P_M^i = K[iR[i|t[i] \) for \( i = 1, \ldots, m \). The projective cameras \( P^i \) are related to \( P_M^i \) by \( P^i = P_M^i H \) where \( H \) is an unknown homography. The absolute dual quadric \( \Omega^* \) is the symmetric \( 4 \times 4 \) rank 3 matrix. In a Euclidean frame, \( \Omega^* \) has the form \( \bar{I} = \text{diag}(1,1,1,0) \) and, in a projective frame, \( \Omega^* \) has the form \( \Omega^* = H\bar{I}H^T \). The absolute dual quadric \( \Omega^* \) projects to the dual image of the absolute conic \( \omega^* \) and we have \( \omega^* = P^i \Omega^* P^iT \). The matrix \( \Omega^* \) may be determined in the projective reconstruction from constraints on intrinsic parameters. Then the homography \( H \) is also determined by decomposing \( \Omega^* \) as \( H\bar{I}H^T \). Since \( \Omega^* \) is a real symmetric matrix a decomposition of \( \Omega^* \) is easily computed using Jacobi’s eigenvalue decomposition algorithm. \( H^{-1} \) is a homography that takes the projective frame to a Euclidean frame. We get a metric reconstruction by applying \( H^{-1} \) to the points and \( H \) to the cameras.

### 2.3 Metric Reconstruction

With some constraints on intrinsic parameters we can obtain an initial guess by a linear method. Assume that the principal point is known. We change the image coordinates so that the origin corresponds to the principal point: \( x_0 = y_0 = 0 \). Moreover we assume that the skew is zero: \( s = 0 \). These two constraints significantly simplify the problem. The dual image of the absolute conic becomes

\[
\omega^* = \text{diag}(\alpha_x^2, \alpha_y^2, 1)
\]

and the three equations follows from the zero entries:

\[
(P^i \Omega^* P^iT)_{1,2} = 0, \quad (P^i \Omega^* P^iT)_{1,3} = 0, \quad (P^i \Omega^* P^iT)_{2,3} = 0
\]  
(2)

Virtual Object Placement in Video for Augmented Reality 17
For $m$ views, there are $3m$ constraints. The estimation of $\Omega^*$ is a problem of the linear system. Since the matrix is symmetric, it is parametrized linearly by a 10D vector $x$:

$$x = [q_{11}, q_{12}, q_{13}, q_{14}, q_{22}, q_{23}, q_{24}, q_{33}, q_{34}, q_{44}]^T$$

Rearranging (2) into a matrix equation of the form $Ax = 0$ where $A$ is a $3m \times 10$ coefficient matrix from $P_i$, we get a usual linear system. A least-squares solution of $Ax = 0$ is obtained using the SVD. From three images 12 equations are available and a least-squares solution is obtained.

The homography $H$ is obtained by decomposing $\Omega^*$. Then, by applying $H^{-1}$ to the points and $H$ to the cameras, we get a metric reconstruction. The rank 3 constraint does not be enforced and the absolute dual quadric $\Omega^*$ computed by the linear method will not be rank 3 in general. A rank 3 matrix can be obtained by setting the smallest eigenvalue to zero in the eigenvalue decomposition. The rank 3 matrix can be used as an initial value for an iterative method.

The calibration matrix $K_i$ of each camera may be computed directly by computing $\omega^i = P_i \Omega^* P_i^T$ and by Cholesky factorization from the equation $\omega^i = K_i K_i^T$. However, there is a difficulty in enforcing the condition that $\Omega^*$ is positive semi-definite. If $\Omega^*$ is not positive semi-definite, $\omega^*$ would not be positive-definite and $\omega^*$ cannot be decomposed using Cholesky factorization to compute the calibration matrix.

### 3 Object Image Blending

Once we recover the scene structures and camera parameters using the method described in the previous section, it is possible to insert virtual objects into the video frames. Since virtual objects are provided with their local coordinates and the recovered scene is represented with world coordinates, we should transform the virtual objects to the world coordinate system. A user manually assigns four corresponding points of a virtual object and the recovered scene structure by clicking their image points. Using the correspondences, the local to world transformation matrix is computed. The texture blending process is shown in Fig. 2. Now the virtual object is projected to each video frame using the recovered camera projection matrix and the image regions for the object are found. For each projected face, the texture pixels are computed from the textures of the virtual object. For each inner position $(x, y)$ of the projected face, the color $f'(x, y)$ of the pixel is determined by the color $f(u, v)$ at the input frame and also by the color $t(x, y)$ at the virtual 3D object. The weight for $t(x, y)$ is increased when the position is close to a face edge.

To make the blended view photo-realistic, we fully implemented the entire steps of transformations and texture mapping. For the visible faces, the corresponding textures are projected to the image space with a suitable interpolation scheme. For the fast rendering, we trace the projected area on the frames and determine the pixel color for each pixel position.
Texture fusion process should consider the two important aspects: edge blending and illumination changes. The blended textures should change accordingly across the projected edges. Abrupt changes cause seams which make unnatural blending. To replace the original frame textures into virtual object textures without seams, near edge pixels should reflect original frame colors. Though the illumination of the virtual object textures and that of the real frame textures could possibly be significantly different, adjusting overall brightness of virtual object textures to that of the real frame textures decreases the difference of the illumination and hence increases the output quality.

### 4 Experimental Results

The proposed feature tracking and 3D recovery method has been implemented and experiments have been performed on a desktop computer. For an initial frame, a number of corner points are extracted. Among them, the user selects target positions which indicate the locations of corresponding corners of a virtual object to be inserted. The feature tracker tracks all the extracted corner points including the selected points. Using tracking results metric reconstruction is performed. The features should be tracked along at least three consecutive frames. The target virtual object is projected to each frame using the reconstructed camera projection matrix. There would be small errors between the tracked points and the projected points. Fig. 3 shows an example of such errors. The black circles are the user clicked positions and the red lines are the projection of a rectangle of the recovered structure. From the four selected corner positions, we obtained stable tracking and reconstruction results which are shown in Fig. 4. For each position, the reprojection error was at most 3 pixels. The tracking accuracy is very critical since the metric reconstruction is directly influenced by
Fig. 3. The tracked points and projected target image region

Fig. 4. Distances between projection points and tracked points

Fig. 5. Test images from the house frames (upper) and the village frames (lower)

the tracking accuracy. Fig. 5 and Fig. 6 show two test results of metric reconstruction. The actual size of the structure needs not be considered since only the relative depths of object faces are used in the blending stage. Once a metric structure is available, a wide range of video synthesis applications are possible.
Fig. 6. The recovered structures from the house frames (upper) and from the village frames (lower)

Fig. 7. The chessboard frames (upper) and the blended frames (lower)

Fig. 7 shows the input video frames with tracked points and the modified video frames using a simple external texture. The gap between the projected boundary and the region of the checkerboard rectangle is indistinguishable. In Fig. 8, a virtual desktop calendar is placed at the real desktop calendar. The tracking of the corners is robust and the virtual calendar is not oscillated even when the camera moves or rotates abruptly. Fig. 9 shows results of seamless edge blending applied to the desktop calendar test set. The upper left figure is an input frame. We placed a rectangular object using a simple texture replacement (upper middle) and using a seamless edge blending (upper right). The lower figures show the magnified views of the upper right corner areas. Similar textures could be replaced to give better image quality or to enhance the visibility. In Fig. 10, a used tissue box (upper) is replaced by a new tissue box with a brighter texture.
Several other experiments also verified that our system works well for real video streams in which objects are static and the motion of the camera is dominant.

5 Conclusion

We proposed a method of 3D structure reconstruction and blending using a single-camera video stream. Our approach is based on a simplification of camera parameters and the use of projective geometry without camera calibration.

The proposed method can be applied to many practical applications. For examples, the method can be used as a core module for reconstruction systems of
Fig. 10. Three sample images from the tissue box video frames (upper) and the blended video frames (lower)

architecture buildings, real-world environment modelers from videos, measurement applications, camera pose control systems, and other applications. The method is particularly useful to augmented reality applications to insert virtual 3D objects to a real video stream.

Future work still remains. It has been not an easy problem to detect occlusion and fill the holes on the reconstructed surfaces. The high order surface equations should be approximated to substitute appropriate surface patches. Noisy points prohibit accurate estimation of structure geometry and we need to develop a more reliable method which is robust to outliers.

Acknowledgement. This work was supported by the Ministry of Commerce, Industry and Energy (MOCIE) through the Incheon IT Promotion Agency (IIT).

References