A Comparison of Viewing Geometries for Augmented Reality

Dana Cobzas and Martin Jagersand

Computing Science, University of Alberta, Canada
www.cs.ualberta/~dana

Abstract. Recently modern non-Euclidean structure and motion estimation methods have been incorporated into augmented reality scene tracking and virtual object registration. We present a study of how the choice of projective, affine or Euclidean scene viewing geometry and similarity, affine or homography based object registration affects how accurately a virtual object can be overlaid in scene video from varying viewpoints. We found that projective and affine methods gave accurate overlay to a few pixels, while Euclidean geometry obtained by auto calibrating the camera was not as accurate and gave about 7 pixel overlay error.

1 Introduction

In Augmented Reality a virtual object is registered with and visually overlaid into a video stream from a real scene[1]. In classical AR systems this is commonly achieved by a-priori geometric modeling for the registration and using external devices (e.g. magnetic) to track camera pose. Using visual tracking through the real scene camera offers several advantages. Since ideally the real and virtual camera should be the same, it avoids the calibration to an unrelated external sensor. It also allows error minimization using image measurements directly between the real scene and virtual object. Recent progress in geometric vision furthermore offers a variety of methods for auto-calibration and alignment of object without needing any a-priori information. These new methods introduce a variety of choices in building an AR system. First, under varying camera models, the scene-camera pose tracking can be done in Euclidean[11], affine[7] or projective[10] formulation. Second, the VR object is normally given as an a-priori (Euclidean) graphics model, but in recent work also captured image-based objects have also been inserted[2, 9]. Third, the transform which aligns the object can either be similarity[3], affine or homography[12].

An important consideration in designing a system is choosing the geometric representation for the above three parts so that the accuracy constraints of the task at hand are satisfied. This is perhaps particularly important in industrial applications where AR can be used e.g. to overlay geometric guides for machining and assembly. In AR a relevant way to characterize accuracy is in pixel reprojection error. Note that this is different from e.g. absolute errors in computed camera pose and scene structure, since some errors will cancel out when projected. However, AR is also different from pure re-projection. In the alignment phase the AR object and scene geometry are related, and inconsistencies

J. Bigun and T. Gustavsson (Eds.): SCIA 2003, LNCS 2749, pp. 501–508, 2003.

in either can make them impossible to align correctly. In this paper we present a study of reprojection errors when inserting and AR-rendering an object under Euclidean, affine and projective geometry.

In the next section we first define and describe the geometry of AR capture and rendering in general, then specialize this to each of the chosen geometries. In Section 3 we compare experimentally the accuracy under each geometry, and finally in Section 4 we summarize and reflect on the consequences of our study.

2 Theory

Below we will first present a general formulation of an AR system that will later be specialized for particular camera and scene models.

An augmented reality system involves inserting a virtual object into a real scene, and continue to render the scene augmented with the new object. We assume that the geometry of the object is given a-priory as a set of 3D coordinates X^o. The basic steps of an AR system, as illustrated in Figure 1, are:

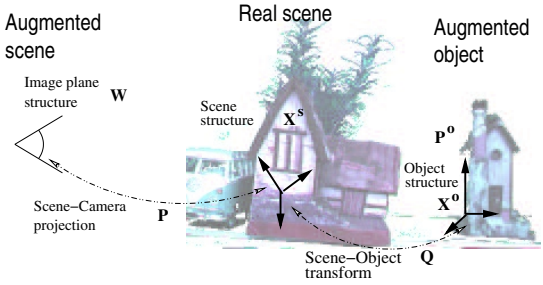

Fig. 1. Coordinate frames in the AR system

Scene structure and motion Before inserting the object, the structure of the scene X^s has to be recovered from a set of fiducial points $\mathbf{u}_{ik}, i = 1 \ldots N$ tracked in a sequence of training images $I_k, k = 1 \ldots M$. This is a common and well studied problem in computer vision, known as structure from motion (SFM). The 3D structure X^s is related to its projection u_k in image k by the projection matrix P_k:

$$u_k = P_k X^s \tag{1}$$

The projection matrix contains both camera internal parameters and camera motion parameters (external) expressed with respect to a reference frame. For our study we consider an uncalibrated camera i.e. both the internal and external parameters are initially unknown.

After recovering the scene structure, camera motion can continually be tracked from the projection of the scene points, by factoring P_k in Equation 1. If desired, the scene structure can also be improved by considering all the available frames.

Object registration The core problem in an AR system is how to relate object structure X^o with scene structure X^s so that the augmented structure $X^a = [X^s; X^{os}]$ can be rendered using the scene projection matrix P_k. This involves recovering the geometric transformation Q that registers the object reference system with the scene reference system: $X^{os} = QX^o$. Another way to solve the registration problem is to recover a virtual camera matrix P_k^o so that the projection of the object with this camera appears correct. It is easy to see that $P_k^o = P_k Q$.

Scene rendering After the virtual object has been inserted into the scene geometry, the augmented scene X^a is rendered using the current projection matrix P_k. Alternatively, a virtual camera P_k^o can computed by transforming the scene camera to the object coordinates and the virtual object is rendered separately and overlaid onto the current scene.

Different AR systems can be created depending on the chosen camera model and reconstruction method. In the next subsections we will present four examples, starting with a projective camera model and reconstructed projective structure, that is then upgraded to an Euclidean structure. Next an affine model is reconstructed assuming an affine camera, that can be upgraded to a metric model if the camera is constrained to weak perspective.

2.1 Projective structure

The most common camera model is the projective, represented by a 3×4 matrix P_k^p. We assume N points tracked in M views. Let the homogeneous coordinates of the 3D scene points and 2D image points be:

$$\mathbf{X}_i = [X_i, Y_i, Z_i, 1]^T \qquad \mathbf{u}_{ik} = [u_{ik}, v_{ik}, 1]^T, \quad i = 1 \ldots N, k = 1 \ldots M \tag{2}$$

They are related by the projection equation:

$$\rho_{ik} \mathbf{u}_{ik} = P_k^p \mathbf{X}_i \tag{3}$$

where ρ_{ik} is a nonzero scale factor, which in general is different for each point and view. It has been shown that P_k^p can be recovered up to a 3D projective transformation [5]. There are several well known estimation algorithms to recover the projective structure and motion of a scene using the fundamental matrix (2 views), the trilinear tensor (3 views) or multi view tensors for more than 3 views. In our experiments we used the algorithm developed by Urban et all [15] that estimates the trilinear tensors for triplets of views and then recovers epipoles from adjoining tensors. The projection matrices are computed at once using the recovered epipoles.

At the registration stage the projective scene structure is related by the object structure by a 3D projective transformation Q^p (4×4 matrix) that has in general 15 DOF and can be recovered from 5 corresponding points. In practice the correspondences are specified by aligning the projection of 5 fiducial points in two views from which their projective coordinates are recovered.

2.2 Euclidean structure

In many applications we are interested in recovering the Euclidean structure of the scene and the true (Euclidean) projection matrices. As mentioned before, there is a ambiguity in recovering the projective structure that cannot be solved without additional knowledge. The estimated projection matrices differ from the 'real' Euclidean ones by a 3D projective transformation (4×4 matrix):

$$P_k^e = P_k^p H \qquad X^e = H^{-1} X \tag{4}$$

The Euclidean camera has the form:

$$P_k^e = \rho K[R_k| - R_k \mathbf{t}_k] \qquad K = \begin{bmatrix} f_u & s & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

where R and t are the 3D rotation and translation, respectively, and K is a calibration matrix.

If we assume that the cameras have zero skew and aspect ratio ($f_u = f_v$ and $s = 0$) and the principal point (u_0,v_0) is approximately known, the Euclidean projections can be recovered using self-calibration [14]. There is still a scale ambiguity that cannot be recovered without additional metric measurements.

The estimated Euclidean structure of the scene is related to the Euclidean structure of the object by a similarity transformation $Q^e = \begin{bmatrix} sR & \mathbf{t} \\ O^T & 1 \end{bmatrix}$. Q^e has in general 7DOF and can be recovered from 3 or more corresponding points. Similar to the case of projective structure we specified the projection of 3 fiducial points in 2 views and recover their Euclidean coordinates.

At the stage of rendering, the estimated Euclidean projection matrix can be factored into its components (from Equation 5) and the object can be rendered though a traditional computer graphics pipeline.

2.3 Affine structure

When the scene depth is small relative to the distance from the camera to the scene, the camera can be approximated with a 2×4 affine projection matrix P^a. For the linear affine camera, structure and motion can be estimated efficiently using factorization [13]. N points tracked in M views, with the affine coordinates $\mathbf{u}_{ik} = [u_{ik}, v_{ik}]^T$, form a $2M \times N$ measurement matrix W that can is factored into M projection matrices $P_k^a, k = 1 \ldots M$ and an affine structure $\mathbf{X}_i = [X_i, Y_i, Z_i]^T, i = 1 \ldots N$ that satisfy the reprojection property:

$$u_k = P_k^a X \tag{6}$$

The affine structure of the scene is registered with the object structure using a 3D affine transformation Q^a that has in general 12DOF and can be estimated from 4 corresponding points. The object is inserted into the scene by specifying the projection of 4 fiducial points in 2 views from which their affine coordinates can be recovered. When rendering the augmented scene, new affine projection matrices consistent with the affine structure X are estimated and used for reprojection.

2.4 Scaled Euclidean structure under weak perspective projection

When the projection model is constrained to weak perspective, the affine structure of the scene can be upgraded to scaled Euclidean. The weak perspective projection matrix has the form:

$$P_k^a = [s_k R_k | t_k] \tag{7}$$

where R_k contains the components $\mathbf{i}_k$ and $\mathbf{j}_k$ along the camera rows and columns of the rotation, s_k is a scale factor and $\mathbf{t}_k$ represents the image plane translation. To constrain P_k^a in this format we align the reference coordinate system with the pixel coordinate system of camera row and column.

Let us denote $P^a = [\hat{R}|t]$ and $\hat{X}$ the projection matrices and structure estimated by the factorization algorithm ($P^a = [P_1^a; \ldots P_M^a]$ is a $2M \times 4$ matrix). The matrices $\hat{R}$ and $\hat{X}$ are a linear transformation of the metric scaled rotation matrix R and the metric shape matrix X. More specifically there exist a 3×3 matrix H such that:

$$\begin{aligned} R &= \hat{R}H \\ X &= H^{-1}\hat{X} \end{aligned} \tag{8}$$

H can be determined by imposing constraints on the components of the scaled rotation matrix R:

$$\begin{aligned} \hat{\mathbf{i}}_k^T H H^T \hat{\mathbf{i}}_k &= \hat{\mathbf{j}}_k^T H H^T \hat{\mathbf{j}}_k \quad (= s_k^2) \\ \hat{\mathbf{i}}_k^T H H^T \hat{\mathbf{j}}_k &= 0 \qquad\qquad\qquad k \in \{1..M\} \end{aligned} \tag{9}$$

The resulting scene structure is metric and is related to the object structure by a similarity transformation that can be recovered from 3 or more corresponding points (same as the structure in Section 2.2).

When rendering, the recovered 2 rows of rotation can be completed to a full 3D rotation matrix by adding a last row perpendicular to the first two. This allow traditional graphics light and shading to be applied on the virtual object. The main problem with this structure is that there is no perspective effect when rendering and that can result in distorted views.

3 Experiments

We have performed a set of experiments comparing Euclidean, affine and projective formulations. Using a uniform setup and procedure ensures that we actually contrast the models as opposed to artifacts of a particular implementation.

A scene consisting of two planes with a calibration pattern as background was built. Since scene geometry is estimated the particular scene structure and texture doesn't matter, but the high contrast corners on the pattern are easy to track (see Figure 2 top left). To allow accuracy measurements an AR structure of a cube with one corner cut off (giving four simultaneously visible planes) is introduced both as an AR virtual object and physically into the scene using a real object made closely to the geometric specifications of the virtual AR object.

The scene is viewed by a commodity machine vision camera, 640x480 pixels Basler A301fc with a Pentax 12mm lens. To allow precise point correspondences between views high contrast corner markings on the background and cube were tracked using XVision[4]. Errors relating to the physical object geometry and tracking were both less than one pixel.

For each particular choice of geometry and alignment transform the following experimental procedure was followed:

1. The structure and motion of the scene was obtained from an image sequence of 10 frames, with an angular camera variation of 30° pan and tilt w.r.t. the scene. (Because scene planes were 90° apart it was difficult to move more without loosing track of points on one of the planes.)
2. The scene and virtual AR object was registered using a varying number of corresponding points in two views to establish a basis transform between real and virtual frames. Here the actual tracked points of the real cube was used in place of the user clicking to avoid user induced alignment errors. We ensured that the fiducial points were not coplanar and the insertion frames were about 7° apart.
3. The virtual AR object was reprojected into 128 different views (again with an angular camera variation of about 30° to maintain simultaneous tracking on all planes). The errors were computed by comparing the point reprojections with the tracked points on the real version of the same object physically inserted into the scene.

The most important aspect of an AR system is how precisely the virtual object is rendered in new views. Over all views the average reprojection error is a few pixels for all models except the Euclidean. The Euclidean model has a constant error of about 10 pixels. We think is due to the auto-calibration giving an erroneous scene structure, which is incompatible with the object. Hence it cannot be aligned with the few freedoms of the similarity transform. Further evidence for this is that when instead using a projective homography, the alignment was comparable to the other types of geometric structure.

The reprojection error is not constant, but increases as the camera viewpoint becomes more distant from the views where the object was registered. We found that the effects of camera translations were minor, while rotations mattered. In Figure 2 (bottom) we can see that for the projective and affine models, the re-projection error starts at 2-4 pixels near the insertion view and rises to about 6 pixels at farther views for the affine model. This behaviour was similar for all angular directions on a viewing sphere.

Another consideration is the number of points used in aligning the AR object and scene structure. Most published systems have used the minimum number needed for a particular transform (e.g. 3 for similarity, 4 for affine and 5 for projective). However, doing a least squares fit over more points can even out errors as seen in Fig. 2 (top right).

Overall, we found that for views close to the insertion view, the affine model had a slight advantage of the projective. The affine model is linear and can be accurately estimated for small viewpoint variations. Over all views the affine and

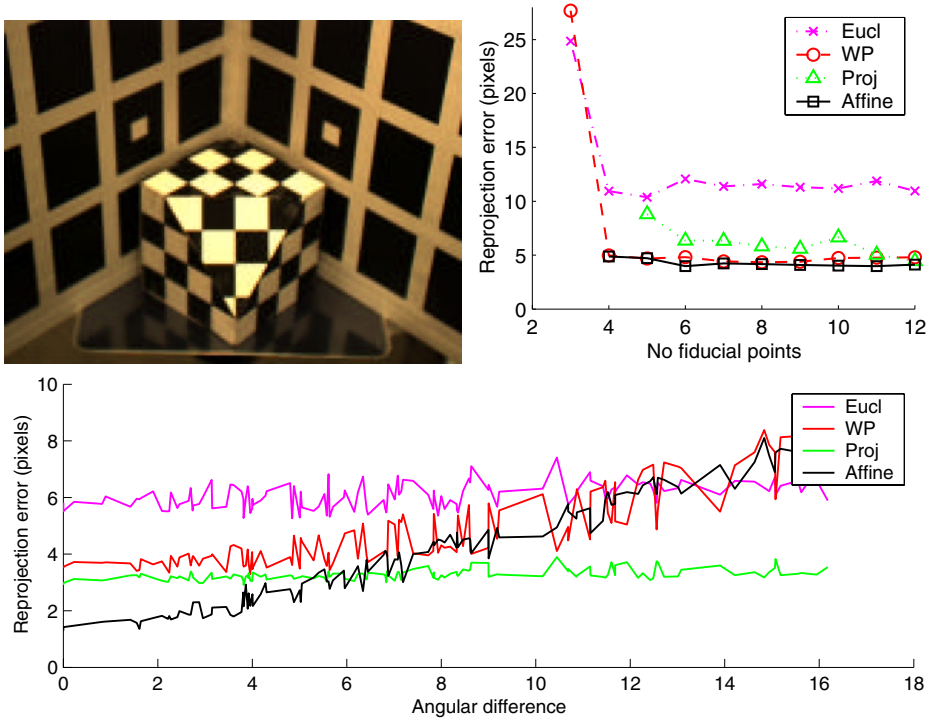

Fig. 2. Experiment scene (top left) and results when varying the number of points used to align the AR object (top right) as well as a function of the angular difference between viewpoint and closest insertion view (bottom)

projective methods are about equal. The projective method obviously is better than the affine at viewpoints distant from the insertion point, where the effects of the linear affine camera approximation are more pronounced. Practically in AR, accuracy in views far away from the registration views often matter less, since the most salient views would be picked out for the object registration and insertion. In distant views the projection of the object is also smaller, and unlikely to be used by an human for precision judgments.

4 Discussion

In this paper we presented a study of the accuracy of AR object registration in Euclidean, affine and projective geometries. We found that for most situations either and affine or projective method work well. The affine gives the lowest error, 2-3 pixels for small viewpoint variations, while the projective is better for large viewpoint variations. The object insertion transform also matters. A transform with more freedoms can "stretch" the object to fit the scene structure better. In this respect metric methods obtained by upgrading projective or

affine structure to Euclidean showed problems in aligning scene structure (with estimation errors) with the predefined AR object.

However a Euclidean metric structure can be valuable. Non-Euclidean insertion requires the user to click on a number of point (4-5) in two images to define a basis transform. This has been argued to be unintuitive. Using an Euclidean similarity the user can specify the object insertion in familiar concepts of metric translations and rotations. Another reason for a metric model is to allow standard graphics calculations of light and shading. Hybrid AR systems could be built using non-Euclidean methods for accurate alignment, but metric calculations for e.g. lighting.

We studied the most common method of using one global registration between the scene and AR object. Variations of this are possible For instance, insertion by local feature alignment have been used in the case of planar transfer[12]. Explicit minimization of image error is also used in visual servoing where instead of a virtual object real objects are aligned in real scenes. These methods can achieve subpixel errors[6], and have recently been transferred to the AR application[8].

References

1. R. Azuma. A survey of augmented reality. *Presence: Teleoperators and Virtual Environments*, 6(4):355–385, 1997.
2. D. Cobzas, M. Jagersand, and K. Yerex. Editing real world scenes: Augmented reality with image-based rendering (poster). In *IEEE Virtual Reality*, 2003.
3. K. Cornelis, M. Pollefeys, M. Vergauwen, and L. V. Gool. Augmented reality using uncalibrated video sequences. *Lecture Notes in Computer Science*, 2018, 2001.
4. G. D. Hager and K. Toyama. X vision: A portable substrate for real-time vision applications. *Computer Vision and Image Understanding: CVIU*, 69(1):23–37, 1998.
5. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2000.
6. M. Jagersand, O. Fuentes, and R. Nelson. Experimental evaluation of uncalibrated visual servoing for precision manipulation. In *Int. Conf. on Robotics and Automation*, 1997.
7. K. N. Kutulakos and J. R. Vallino. Calibration-free augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):1–20, 1998.
8. E. Marchand and F. Chaumette. Virtual visual servoing: A framework for real-time augmented reality. In *Eurographics*, 2000.
9. Y. Seo, M. H. Ahn, and K. Hong. Video augmentation by image-based rendedring under the perspective camera model. In *International Workshop on Advanced Image Technology*, pages 147–153, 1998.
10. Y. Seo and K. Hong. Calibration-free augmented reality in perspective. *IEEE Transactions on Visualization and Computer Graphics*, 6(4), 2000.
11. A. Shahrokni, L. Vachetti, V. Lepetit, and P. Fua. Extended version of polyhedral object detection and pose estimation for augmented reality applications. In *International Conference on Computer Animation*, 2002.
12. R. A. Smith, A. W. Fitzgibbon, and A. Zisserman. Improving augmented reality using image and scene constraints. In *British Machine Vision Conference*, 1999.
13. C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal of Computer Vision*, 9:137–154, 1992.
14. W. Triggs. Auto-calibration and the absolute quadric. In *IEEE Conference Computer Vision and Pattern Recognition (CVPR97)*, pages 609–614, 1997.
15. M. T.Werner, T.Pajdla. Practice of 3d reconstruction from multiple uncalibrated unorganized images. In *Czech Pattern Recognition Workshop*, 2000.