Efficient Lookup Table Based Camera Pose Estimation for Augmented Reality

Abstract

Until now, existing camera pose estimation methods for the widely used square marker-based augmented reality (AR) are either highly sensitive to noise or much time consuming, and developers have to work hard to find the proper trade-off between computational speed and quality in mobile AR applications where computational resources are limited. The major difficulty is that only the 4 corner points of the square AR marker are available, and no redundant point correspondences can be used for a stable estimation.

To solve this problem, an efficient lookup table (LUT)-based non-iterative solution is presented in this paper that achieves high stability in the presence of noise better than the most robust and accurate iterative solutions in the field, with the same level of accuracy and a much lower computational complexity. Our central idea consists of extracting a key parameter β from the camera pose and creating a LUT for β by

taking the symmetrical structure of the square marker into account, thereby exploiting additional information.

Keywords: Augmented reality, camera pose estimation, lookup table.

Introduction

Augmented reality (AR) is an important technology in the computer graphics field, which attempts to integrate digital contents, such as computer graphics, annotations and texts, into the physical scene seamlessly in real-time [1, 2, 3, 4]. It has been predicted that, the trend of AR in mobile devices is one of the technologies most likely to alter industries, fields of research and the ways we live [5, 6].

However, it is not easy to find the optimal trade-off between computational speed and quality in mobile AR applications, because the computational resources of mobile devices are limited. In particular, the only current way to eliminate the significant instability introduced by the image noise [7] involves time consuming iterative scheme in the camera pose estimation procedure.

The pose of the AR camera must be estimated reliably to align digital contents exactly with the physical objects in users' visual field [8]. For example, the great work of the Handheld Augmented Reality project [9] at Graz University can estimate camera pose from deformable markers, and the exciting work of Klein and Murray [10, 11] can deal with marker-less registration. Nevertheless, the square marker method [12, 13, 1] is one of the most widely used pose estimation methods in current AR applications because the square marker can be reliably distinguished from a cluttered background [14], it is effectively encoded [15, 16], and it is easily implemented. The well-known AR systems, such as ARToolkit [12], Handheld AR project [9], ARTag [13] and ARStudio [17], all use square

markers for pose estimation. The pose estimation based on square marker is a specific kind of perspective-4-point problem (P4P) that aims at determining the pose of a calibrated camera from 4 corresponding 3D/2D point sets. Its solution can be divided into 2 classes: non-iterative and iterative solutions.

Non-iterative solutions

In theory, the camera pose can be uniquely determined from 4 coplanar yet non-collinear points. Abidi and Chandra [18] proposed a non-iterative solution from four coplanar points for a perspective camera with an unknown focal length. Bujnak et al. [19] generalized this solution to four arbitrary points. The most widely used non-iterative solution in AR is the homography method [17], which takes the first two columns of the homography matrix as the rotation matrix, and augments the third column by the cross product of the first two columns. However, these methods are very sensitive to image noise and lead to significant jitter phenomenon in AR applications. In the presence of noise, more robust and accurate solutions can be achieved by introducing redundancy as additional information [20, 21]. However, when using square AR marker with only 4 corner points, no redundant points can be used to enhance the robustness and accuracy against image noise, because at least 4 coplanar points are needed to determine the camera pose.

Iterative solutions

More accurate results can be achieved by iterative schemes based on the minimization of non-linear cost functions defined either in the image space [22], or in the target space [23]. Moreover, non-iterative methods are ordinarily used as the initial guess for the iteration. Lu et al. [23] provided one of the most widely used algorithms, which minimizes the error expressed in the 3D target space. Schweighofer and Pinz [7] enhanced the robustness of the Lu et al. algorithm by taking local minimums into account, thereby making one of the most robust and accurate algorithms for camera pose estimation from planar targets. The primary advantage of these iterative algorithms is high-accuracy, but the drawbacks are as follows: (1) a highly dependence on the initial guess, and (2) a high computational complexity.

Why to use a LUT

The increasing use of AR in mobile devices [24, 25, 26] requires efficient, robust and accurate solutions. The LUT approach is efficient and suitable for devices with limited computational resources, such as mobile devices. In today's mobile AR applications, LUTs are used for accelerating details as much as possible [24].

In this paper, we propose a simple LUT approach for estimating the camera pose from square AR marker. The robustness of our method is slightly better than one of the most robust iterative algorithms for planar targets given by Schweighofer and Pinz [7], with a much lower computational complexity (see Fig. 8). When there is image noise, the accuracy

of our algorithm is significantly better than the current non-iterative methods and as accurate as the state-of-the-art iterative algorithms (see Fig. 9 and Fig. 10). Our central idea for providing more robust and accurate results is to create a LUT for a key parameter β by taking the symmetrical structure of the square AR marker into account, thereby exploiting additional information. The LUT is created using perspective projection, not orthographic or other approximation, which has not been reported previously.

To the best of our knowledge, no existing solution for the P4P problem can be both efficient and accurate because of the lack of redundant points. Although the square marker is a specific kind of target, our contribution is to provide the first non-iterative solution that outperforms the time consuming iterative ones in both efficiency and accuracy in the difficult non-redundant case. It can be seen as a useful extension of the best iterative [21] and non-iterative [7] solutions in AR field.

Camera Pose Estimation from Square AR Marker

Fig. 1 shows the orthogonal camera coordinate system $O_cX_cY_cZ_c$, the normalized image plane, and the orthogonal world coordinate system $O_wX_wY_wZ_w$. The 4 corner points p_i $(i = 1, 2, 3, 4)$ of AR marker are projected onto the normalized image plane as $m_i = (u_i, v_i)$, and the marker's center p_0 is projected as m_0.

The unknown depths of corner points p_i are referred to as t_i, and the distance from O_c to O_w is denoted as t_0. The projection angle of m_i is denoted as $\theta_i = \angle m_i O_c m_0$ which can be

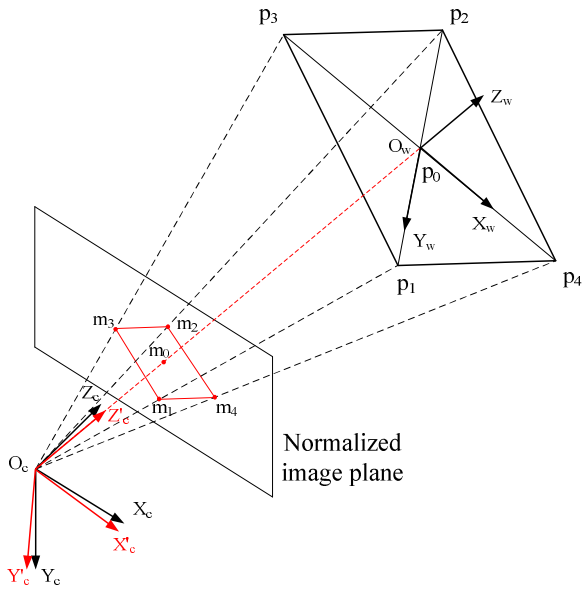

Figure 1: The projection of the AR marker onto the image plane. A new orthogonal camera frame $O_cX_c'Y_c'Z_c'$ is introduced, with axis Z_c' points at the center of the AR marker.

measured by the coordinates of m_i. The corner point with the largest projection angle θ_i is denoted as m_1, the point opposite is called m_2, and the remaining corner points are m_3 are m_4.

Primary Rotation Angle β

The camera pose is estimated by extracting a key parameter named "primary rotation angle β". By using β as key parameter, the ambiguous poses, which significantly affect the stability of the solution [7], can be effectively expressed and determined, and the other unknown elements of the camera pose can be easily figured out by a few geometric constraints immediately after β is determined.

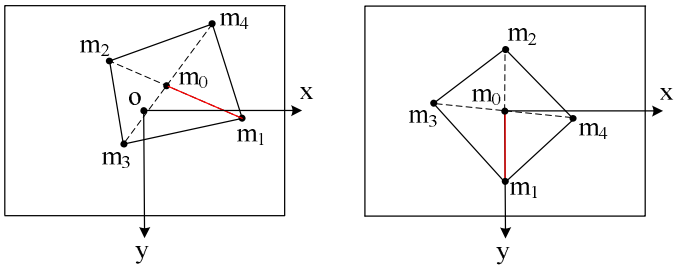

Figure 2: By rotating the camera from the original coordinate frame $O_cX_cY_cZ_c$ to a new frame $O_cX_c'Y_c'Z_c'$, the marker's center m_0 is projected onto the center of the image plane, and the line m_0m_1 is aligned with the y-axis.

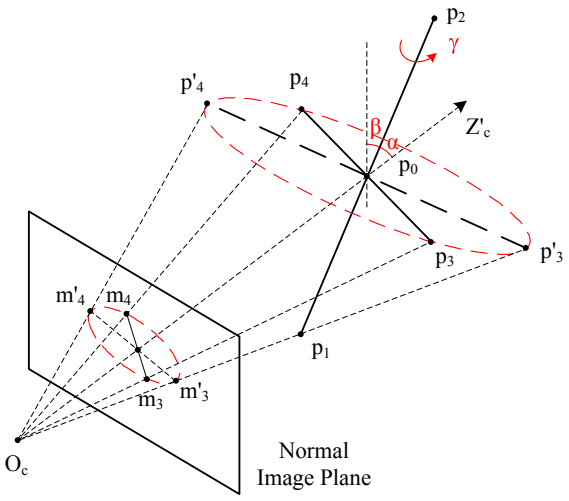

Figure 3: Let α denote the angle between p_1p_2 and the optical axis Z_c' of the camera $O_cX_c'Y_c'Z_c'$, and the primary rotation angle β denotes the complementary angle of α. The rotation angle γ denotes the rotation around p_1p_2. The circular path of p_3 and p_4 is projected on the image plane as an ellipse.

The camera pose T is expressed by the transformation of the target $O_wX_wY_wZ_w$ in the camera coordinate frame $O_cX_cY_cZ_c$, and it can be decomposed into a rotation matrix R' and 3 parameters (β, γ, t_0) as follow:

$$T = \begin{bmatrix} R' & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathrm{rot}(X,\beta)\mathrm{rot}(Y,\gamma) & t \\ 0 & 1 \end{bmatrix} \tag{1}$$

1) R' represents the rotation transform on the camera from $O_cX_cY_cZ_c$ to $O_cX'_cY'_cZ'_c$, with axis Z'_c points at the marker's center p_0, axis Y'_c on the plane $\Pi O_cm_0m_1$ and perpendicular to the axis Z'_c. By rotating the camera, p_0 is projected onto the center of the image plane as m_0, and the line m_0m_1 is aligned with the y-axis of the image coordinate system(see Fig. 2).

2) $\mathrm{rot}(X, \beta)$ is a rotation of β degrees about the marker's X-axis, $\mathrm{rot}(Y, \gamma)$ represents a rotation of γ degrees about the marker's Y-axis, and $t = [0\ 0\ t_0]^T$ refers to the translation of an AR marker in the coordinate $O_cX'_cY'_cZ'_c$.

Because $\angle m_1O_cm_2 = \theta_1 + \theta_2$ can be determined by the coordinates of m_1 and m_2, the position of the camera is constrained to a circle whose radius $R = d/sin\theta$ (see Fig. 4). As a result, there exists a geometric constraint between β and t_0, whose algebraically expression is as follows:

$$t_0 = d\,\frac{\cos\beta + \sqrt{\cos^2\beta + \tan^2\theta}}{\tan\theta} \tag{2}$$

where d denotes the distance from p_0 to p_1, which equals half of the diagonal length of the marker, and θ denotes $\angle m_1O_cm_2$. A brief proof of (2) is as follow:

Let $|p_1p_2| = 2d$ and $\angle p_1O_cp_2 = \theta$, then O_c lies on a circle with radius $R = d/sin\theta$ (see Fig. 4). We create a 2D Cartesian coordinate system at the center of the circle with its x-axis parallel to p_1p_2. As $p_0 = (0, -d/\tan\theta)$, $\overrightarrow{p_0O_c} = (-t_0\sin\beta, t_0\cos\beta)$ and $O_c = p_0 + \overrightarrow{p_0O_c}$, the 2D coordinate of O_c can be expressed by β, θ and t_0 as

$$\begin{cases} x = -t_0\sin\beta \\ y = -d/\tan\theta + t_0\cos\beta \end{cases} \tag{3}$$

By substituting (3) into the equation of the circle $x^2 + y^2 = (d/\sin\theta)^2$, we obtain

$$t_0^2 - 2\cos\beta\frac{d}{\tan\theta}t_0 - d^2 = 0 \tag{4}$$

The positive solution of (4) is t_0, and then (2) is proven. □

Let the known 3D coordinate of p_i in $O_wX_wY_wZ_w$ be $p_i^w = (x_i^w, y_i^w, z_i^w)$ which is transformed to $O_cX_c'Y_c'Z_c'$ as

$$\begin{bmatrix} x_i^{c'} \\ y_i^{c'} \\ z_i^{c'} \\ 1 \end{bmatrix} = \begin{bmatrix} \mathrm{rot}(X,\beta)\mathrm{rot}(Y,\gamma) & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_i^w \\ y_i^w \\ z_i^w \\ 1 \end{bmatrix} \tag{5}$$

As can be seen in Fig. 1, the 3D coordinates of the corner points in $O_wX_wY_wZ_w$ are $p_1^w = (0, d, 0)$, $p_2^w = (0, -d, 0)$, $p_3^w = (-d, 0, 0)$, $p_4^w = (d, 0, 0)$. According to (5), the corner points are transformed to $O_cX_c'Y_c'Z_c'$ as

$$
\begin{aligned}
p_1^{c'} &= (0, d\cos\beta, t_0 + d\sin\beta) \\
p_2^{c'} &= (0, -d\cos\beta, t_0 - d\sin\beta) \\
p_3^{c'} &= (-d\cos\gamma, -d\sin\beta\sin\gamma, t_0 + d\cos\beta\sin\gamma) \\
p_4^{c'} &= (d\cos\gamma, d\sin\beta\sin\gamma, t_0 - d\cos\beta\sin\gamma)
\end{aligned} \tag{6}
$$

and then the 2D coordinates of m_3 and m_4 on the image plane of $O_c X_c' Y_c' Z_c'$ are

$$
\begin{aligned}
u_3 &= \frac{x_3^{c'}}{z_3^{c'}} = \frac{-d\cos\gamma}{t_0 + d\cos\beta\sin\gamma} \\
v_3 &= \frac{y_3^{c'}}{z_3^{c'}} = \frac{-d\sin\beta\sin\gamma}{t_0 + d\cos\beta\sin\gamma} \\
u_4 &= \frac{x_4^{c'}}{z_4^{c'}} = \frac{d\cos\gamma}{t - d\cos\beta\sin\gamma} \\
v_4 &= \frac{y_4^{c'}}{z_4^{c'}} = \frac{d\sin\beta\sin\gamma}{t - d\cos\beta\sin\gamma}
\end{aligned} \tag{7}
$$

A) When the primary rotation angle β is known and $\beta \neq 0$, according to (7), the angle γ can be determined algebraically as

$$
\tan\gamma\sin\beta = \frac{v_3 - v_4}{u_3 - u_4} \tag{8}
$$

B) In a special case where $\beta = 0$, the angle γ can be determined by t_0, d and θ_{34}, where θ_{34} denotes the angle $\angle p_3 O p_4 (= \theta_3 + \theta_4)$ which can be measured from the coordinates of m_3 and m_4. The geometric constraint among $(\gamma, t_0, d, \theta_{34})$ is similar to the constraint among (β, t_0, d, θ) expressed in (4),

$$
t_0^2 - 2\cos\gamma\frac{d}{\tan\theta_{34}}t_0 - d^2 = 0 \tag{9}
$$

As a result, when $\beta = 0$, the angle γ can be determined algebraically as

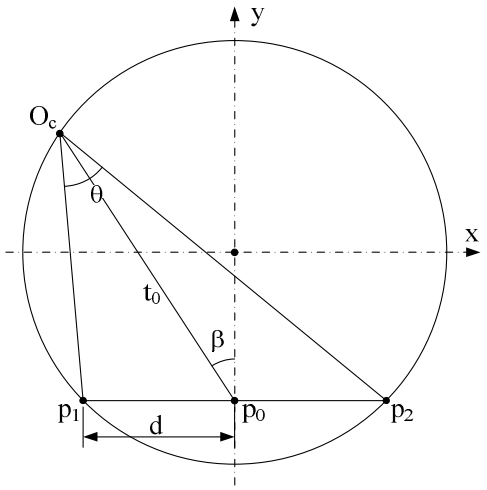

Figure 4: The relationship between β and t_0 when θ is determined.

$$\cos\gamma = \tan\theta_{34}(t_0^2 - d^2)/(2dt_0) \tag{10}$$

As analyzed above, the camera pose T can be decomposed into 4 parts, in which R' can be easily calculated (please refer to the pseudo-code in section 2.4), while t_0 and γ can be expressed in terms of β. Thus, the main task is to solve for the primary rotation angle β.

The LUT for β

The primary rotation angle β is looked up from LUT, which is denoted as an operation $\beta = \mathrm{LUT}(\theta, u, v)$ with 3 input parameters. Details of the LUT is as follow.

The corner points p_3 and p_4 are on a circle around the line p_1p_2 due to the symmetrical structure of the square marker. As a result, m_3 and m_4 are on an ellipse which is the projection of the circle on the image plane (see Fig. 3). The ellipse can be expressed analytically

as follow:

$$\frac{t_0^2 - d^2\cos^2\beta}{d^2}u^2 + \frac{[(t_0^2 - d^2\cos^2\beta)v - d^2\sin\beta\cos\beta]^2}{d^2 t_0^2\sin^2\beta} = 1 \tag{11}$$

For a given θ (namely $\angle m_1 O_c m_2$) and marker size d, the elliptic path is uniquely determined by β. Because θ_1 is defined as the largest angle in θ_i, the possible values of β range from 0 to -45 degrees. It can be briefly proved as follow:

If $90^o > \beta > 0^o$, according to (6) we have that $\theta_1 = \arctan(\frac{d\cos\beta}{t_0 + d\sin\beta})$ and $\theta_2 = \arctan(\frac{d\cos\beta}{t_0 - d\sin\beta})$, $\theta_1 < \theta_2$.

If $-90^o < \beta < -45^o$, according to (6) we have that $\theta_3 = \arctan(\frac{d\sqrt{\cos^2\gamma + \sin^2\gamma\sin^2\beta}}{t_0 + d\cos\beta\sin\gamma})$ and $\theta_4 = \arctan(\frac{d\sqrt{\cos^2\gamma + \sin^2\gamma\sin^2\beta}}{t_0 - d\cos\beta\sin\gamma})$. Let θ_m denote the bigger angle of θ_3 and θ_4, we have $\theta_m = \arctan(\frac{d\sqrt{\cos^2\gamma + \sin^2\gamma\sin^2\beta}}{t_0 - |d\cos\beta\sin\gamma|})$. When $\gamma = 90^o$, θ_m reachs its minimal and $\theta_m \geq \arctan(\frac{d\sin\beta}{t_0 - d\cos\beta}) > \arctan(\frac{d\cos\beta}{t_0 + d\sin\beta}) = \theta_1$.

As θ_1 is defined as the largest angle in θ_i, we can say that $\beta \in [-45,\ 0]$. □

We plot all elliptic paths on the image plane (see Fig. 5), and thus every possible position of m_3 and m_4 is mapped to a color that refers to possible β values. We sample the ecliptic figure on the image plane in the u and v directions to form a sub-table for β, and a resolution of 150×150 points will be enough for accurate pose estimation in practice.

Furthermore, we can plot a series of sub-tables corresponding to the possible values of θ. For example, θ can be sampled from 2 to 50 degrees with half-degree intervals (see Fig. 6).

1) Required Memory Space

The value of β can be encoded to 8-bits data. Because the LUT for β is symmetrical about the y-axis (see Fig. 6), a sub-table for a given θ of dimensions 150×150 can be reduced to 150×75. In our experiment, the LUT size is $97 \times (150 \times 75) \approx 1MB$, in which "$97''$ denotes the number of θ sampled from 2 to 50 degrees with half-degree intervals, and "$150 \times 75''$ denotes the quantization resolution of the LUT in the u and v directions.

The size of LUT can be further reduced to $296KB$ without losing accuracy by using nested LUTs which contain a low-resolution LUT for the whole region and a high-resolution LUT nested in the sub region of the former one. Firstly, a LUT with resolution $25 \times (100 \times 50) \approx 125KB$ is sampled for the whole region. And then, in the dangerous region (see Fig. 5) where β is sensitive corresponding to (u, v, θ), the LUT is more densely sampled with resolution $97 \times (42 \times 42) \approx 171KB$. The nested LUTs remain the same level of accuracy as the uniformly sampled LUT with resolution $97 \times (150 \times 75)$.

2) The LUT Can Be Shared By Different Size of AR Marker

The generation of the LUT for β takes several minutes, but the user need not repeat the process for different AR markers. Assuming an arbitrary square marker with size $d^{(a)} = k\,d$, its corresponding distance parameter $t_0^{(a)} = k\,t_0$ according to (2). Substituting $d^{(a)}$ and $t_0^{(a)}$ into (11), we find that the analytical expression of the new elliptic path remains the same as the original. As a result, AR markers of different sizes can share the same LUT.

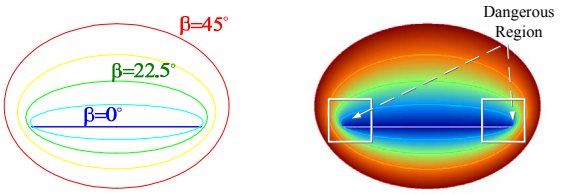

Figure 5: For a given θ, all elliptic paths are drawn on the image plane as a sub-table.

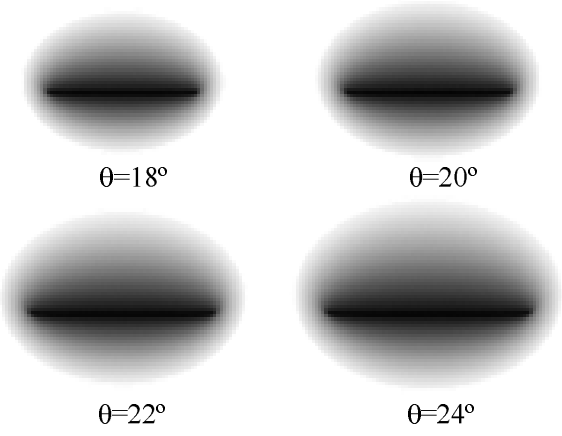

Figure 6: The LUT consists of a series of sub-tables corresponding to the values of θ sampled.

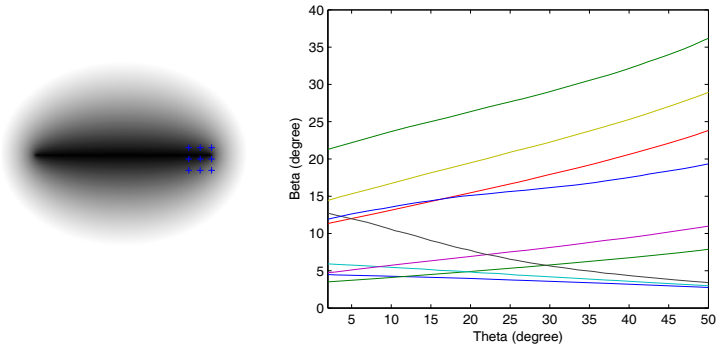

Figure 7: Left: A series of points (u,v) are picked in the LUT to study the relationship between β and θ. Right: For each point, the value of β in the LUT is plotted as a function of θ from 2 to 50 degrees.

Camera Pose Estimation

1) Lookup β in the LUT

Assuming that $\beta_i = \mathrm{LUT}(\theta, u_i, v_i)$ $(where\ i = 3\ or\ 4)$ denotes the operation that lookups the value of β in the LUT using θ, u_i and v_i as inputs, where θ represents $\angle m_1 O_c m_2$, and u_i and v_i are the normalized image coordinates of m_i. Theoretically, β_3 equals β_4 because of the symmetrical structure of the marker, but in practice they are not equal due to noise. To enhance the robustness and accuracy of our procedure, we use the average of β_3 and β_4 as the value of β.

2) Retrieve Ambiguous Pose β'

Schweighofer and Pinz [7] proved that the predicted camera pose suffers from an ambiguity that results in significant instability when using planar targets. Assuming a rotation

of β about the X-axis, there may exist a second, different angle β' that also leads to a local minimum. In contrast to computing β' by solving a fourth order polynomial in [7], the ambiguous pose can be retrieved more efficiently in our method by looking it up in the table.

Because the ambiguous pose β' is in the opposite direction of β, the LUT for β' is mirror symmetric with the LUT for β about the x-axis. It is easy to find β' in the LUT for β by flipping the y coordinates of m_3 and m_4, namely, to use θ, u_i and $-v_i$ as the input of the LUT operation.

3) Retrieve Camera Pose

Although the camera pose can be calculated by (1) when β and β' are looked up in the LUT, a more accurate result can be achieved by first estimating the depths of the corner points and then retrieving the camera's position and orientation as the Euclidean motion that aligns these points. The depths of marker's 4 corner points t_i $(i = 1 \dots 4)$ can be determined as follows:

$$\begin{aligned} t_1 &= \sqrt{d^2 \cos^2 \beta + (t_0 + d \sin \beta)^2} \\ t_2 &= \sqrt{d^2 \cos^2 \beta + (t_0 - d \sin \beta)^2} \\ t_3 &= \sqrt{d^2 \cos^2 \gamma + d^2 \sin^2 \beta \sin^2 \gamma + (t_0 + d \cos \beta \sin \gamma)^2} \\ t_4 &= \sqrt{d^2 \cos^2 \gamma + d^2 \sin^2 \beta \sin^2 \gamma + (t_0 - d \cos \beta \sin \gamma)^2} \end{aligned} \tag{12}$$

When the depths of corner points are solved, the camera pose can be easily retrieved by the standard method [27].

Pseudo-code

The source code of the LUT solution can be downloaded from "http://xuchi.weebly.com/poselut.html", and the pseudo-code is as follow:

1. Compute the rotation matrix R':

   $m_0 \leftarrow \text{intersection\_of\_lines}(\overline{m_1 m_2}, \overline{m_3 m_4})$

   $Z' \leftarrow \text{normalize}(\overrightarrow{O_c m_0})$

   $X' \leftarrow \text{normalize}(\overrightarrow{m_0 m_1}) \times Z'$

   $Y' \leftarrow Z' \times X'$

   $R' \leftarrow [\ X' \quad Y' \quad Z'\ ]$

   for $i = 1 \ldots 4$

   $$\begin{bmatrix} u_i w_i \\ v_i w_i \\ w_i \end{bmatrix} \leftarrow R' \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix}$$

2. Lookup β and β' in the LUT:

   $\theta \leftarrow \text{measure}(\angle m_1 O_c m_2)$

   $\beta \leftarrow -[\text{LUT}(\theta, u_3, v_3) + \text{LUT}(\theta, u_4, v_4)]/2$

   $\beta' \leftarrow [\text{LUT}(\theta, u_3, -v_3) + \text{LUT}(\theta, u_4, -v_4)]/2$

3. Compute (γ, t_0) and (γ', t_0'):

   $t_0 \leftarrow (\cos\beta + \sqrt{\cos^2\beta + \tan^2\theta}) * d/\tan\theta$ (please refer to (2))

   $t_0' \leftarrow (\cos\beta' + \sqrt{\cos^2\beta' + \tan^2\theta}) * d/\tan\theta$

   if $\beta \neq 0$

$\gamma \leftarrow \arctan[(v_3 - v_4)/(u_3 - u_4)/\sin\beta]$ (please refer to (8))

$\gamma' \leftarrow \arctan[(v_3 - v_4)/(u_3 - u_4)/\sin\beta']$

else (when $\beta = 0$, $\beta' = 0$ either)

$\theta_{34} \leftarrow \text{measure}(\angle m_3 O_c m_4)$

$\gamma \leftarrow \arccos[\tan\theta_{34} * (t_0^2 - d^2)/2d/t_0]$ (please refer to (10))

$\gamma' \leftarrow -\gamma$

4. Compute the camera pose T:

$[t_1, t_2, t_3, t_4] \leftarrow \text{cal\_depths}(\beta, \gamma, t_0)$

$T \leftarrow \text{cal\_camera\_pose}(t_1, t_2, t_3, t_4)$

$[t'_1, t'_2, t'_3, t'_4] \leftarrow \text{cal\_depths}(\beta', \gamma', t'_0)$

$T' \leftarrow \text{cal\_camera\_pose}(t'_1, t'_2, t'_3, t'_4)$

if $\text{residual}(T') < \text{residual}(T)$

$T \leftarrow T'$

"intersection_of_lines" denotes the function that retrieves the intersection point of two lines, "normalize" denotes the function that normalizes a vector, and "measure" denotes the function that measures the angle θ. "LUT" denotes the function that looks up β in the table as mentioned in the previous subsection. "cal_depths" denotes the function that calculates the depth of the corner points by (12). "cal_camera_pose" denotes the standard method that retrieves the camera pose by aligning two 3D point sets. "residual" denotes the function that calculates the projection residual of the AR marker using an estimated pose.

RESULTS

We compared the efficiency, robustness and accuracy of our approach (referred to as LUT) against that of state-of-the-art ones. The following algorithms were compared:

(1) HOMO: Homography method [17].

(2) LMF: A non-iterative solution of Lepetit, Moreno-Noguer and Fua [21], which is one of the most accurate state-of-the-art non-iterative methods. In redundant point cases where $n \geq 6$, the result of LMF can be as accurate as that of the iterative methods.

(3) LHM: A widely used iterative algorithm of Lu, Hager and Mjolsness [23] initialized with a weak perspective assumption.

(4) SP: An iterative algorithm of Schweighofer and Pinz [7] initialized with a weak perspective assumption, which is one of the most robust and accurate solutions for pose estimation from a planar target. SP can deal with arbitrary $n \geq 4$ planar cases.

Synthetic Experiments

We randomly generated 1000 poses for 11 different noise levels from 0 to 5 pixels in a 640×480 image using a virtual pinhole camera with a focal length of $f = 800$ and a principal point at $(u_c, v_c) = (320, 240)$. The size of square marker is $60 \times 60mm$, and the area of the projected target ranges from approximately 600 to 25600 $pixels^2$. The camera orientation is randomly generated, while the angle from the target's Z_w-axis to the camera's Z_c-axis is limited to the range of -82 to 82 degrees.

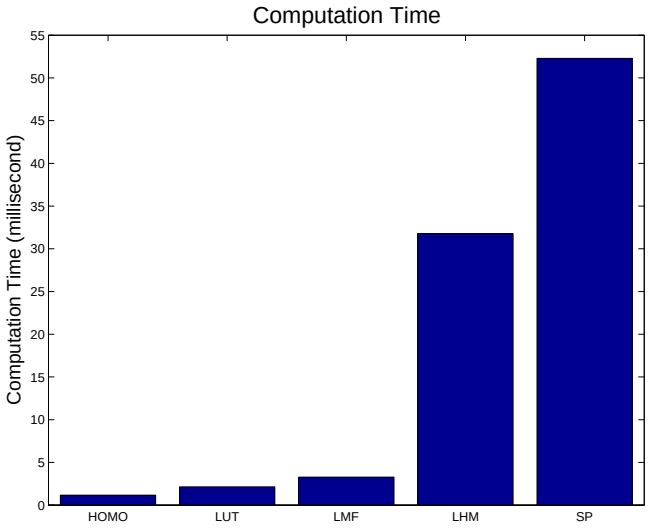

Figure 8: The average computation time of our method against that of state-of-the-art methods.

Given the true camera rotation R_{true} and translation t_{true}, we computed the error of the estimated rotation matrix R by $E_{rot} = max(\Delta\alpha_x, \Delta\alpha_y, \Delta\alpha_z)$ using the unit $degrees$, where $\Delta\alpha_i$ denotes the angle between the i-axis of R_{true} and that of R. The relative error of the estimated translation t is determined by $E_{trans} = ||t_{true} - t||/||t||$.

1) Efficiency: 11,000 random experiments were performed in MATLAB, and the average computational times of each method were plotted in Fig. 8. HOMO takes 1.178 ms, LUT takes 2.159 ms, LMF takes 3.244 ms, LHM takes 33.840 ms, and SP takes 54.118 ms. The computational time per execution of non-iterative solutions (HOMO, LUT and LMF) is almost constant, whereas that of iterative methods (LHM and SP) is highly dependent on the initial guess. The minimal and maximal computational time of LHM was 5.189 ms and

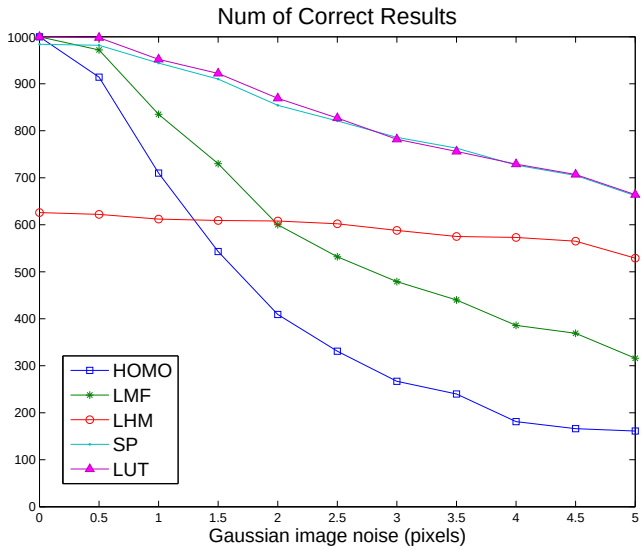

Figure 9: The robustness of our method was compared to that of the state-of-the-art methods by plotting the number of correct results of each method as a function of the noise-level from 0 to 5 pixels. A result with rotation error less than 15 degrees is considered correct.

251.928 ms, respectively, std. 26.282 ms; the minimal and maximal computational time of SP was 18.175 ms and 284.606 ms, respectively, std. 46.542 ms. Compared with iterative methods, the efficiency of our solution is approximately 15 times better than LHM and about 25 times better than SP, while maintaining the same level of accuracy and stability. LUT is slightly less efficient than the simple but inaccurate method HOMO because two ambiguous poses are considered to achieve robust and accurate results.

2) Robustness: We consider the results with a rotation error of less than 15 degrees as correct. The robustness of each method is evaluated by the number of correct results, which are plotted in Fig. 9. Our method outperforms other methods in robustness. Non-iterative methods (HOMO and LMF) achieve a correct rate 100% when the noise-level is 0, but their robustness decreases significantly as the noise-level increases. The correct rate of the iterative method LHM is approximately 60% regardless of noise-levels. The iterative algorithm SP, one of the most robust and accurate solutions for the AR marker problem, tends to achieve stable results at each noise-level that are significantly better than HOMO, LMF and LHM. SP can achieve all possible ambiguous poses in theory, although it sometimes fails to obtain the correct result because the structure of the image points degenerates due to noise whenever false poses have a smaller projection residual than the correct pose. By using the symmetric structure as additional information, our method, LUT, gets slightly more correct results than SP does, because our method is simpler, less complex and more numerically stable. LMF is not stable in the AR marker case because of the lack of redundant points, however it is one of the most accurate, state-of-the-art non-iterative solutions that can be

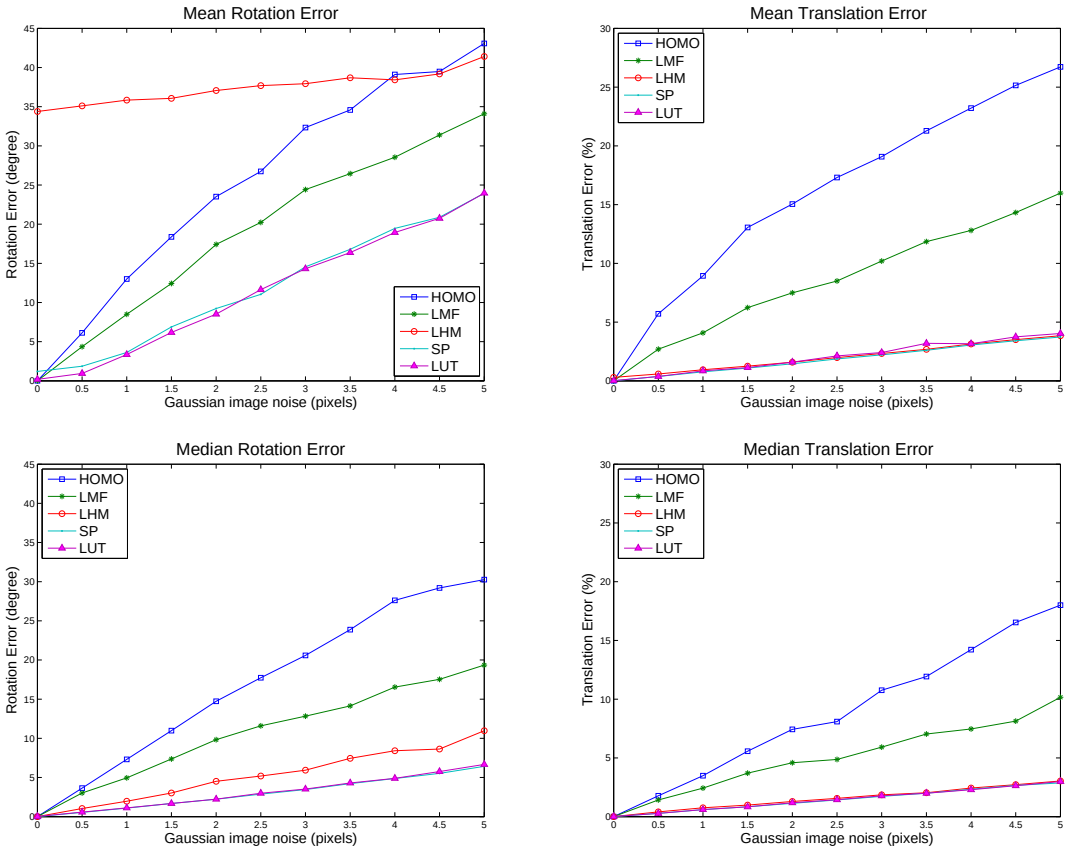

Figure 10: The first row is the mean rotation and translation errors of each of the methods. The second row is the median error.

achieved in this field. To the best of our knowledge, no existing solution can be both efficient and accurate in camera pose estimation from the widely used AR markers. Our LUT method can be seen as a useful extension of SP and LMF in the AR field.

3) Accuracy: The mean and median errors of each method are plotted as a function of noise-level from 0 to 5 pixels in Fig. 10. HOMO and LMF can retrieve exact results when there is no noise, but their errors increase significantly with noise-level. The accuracy of

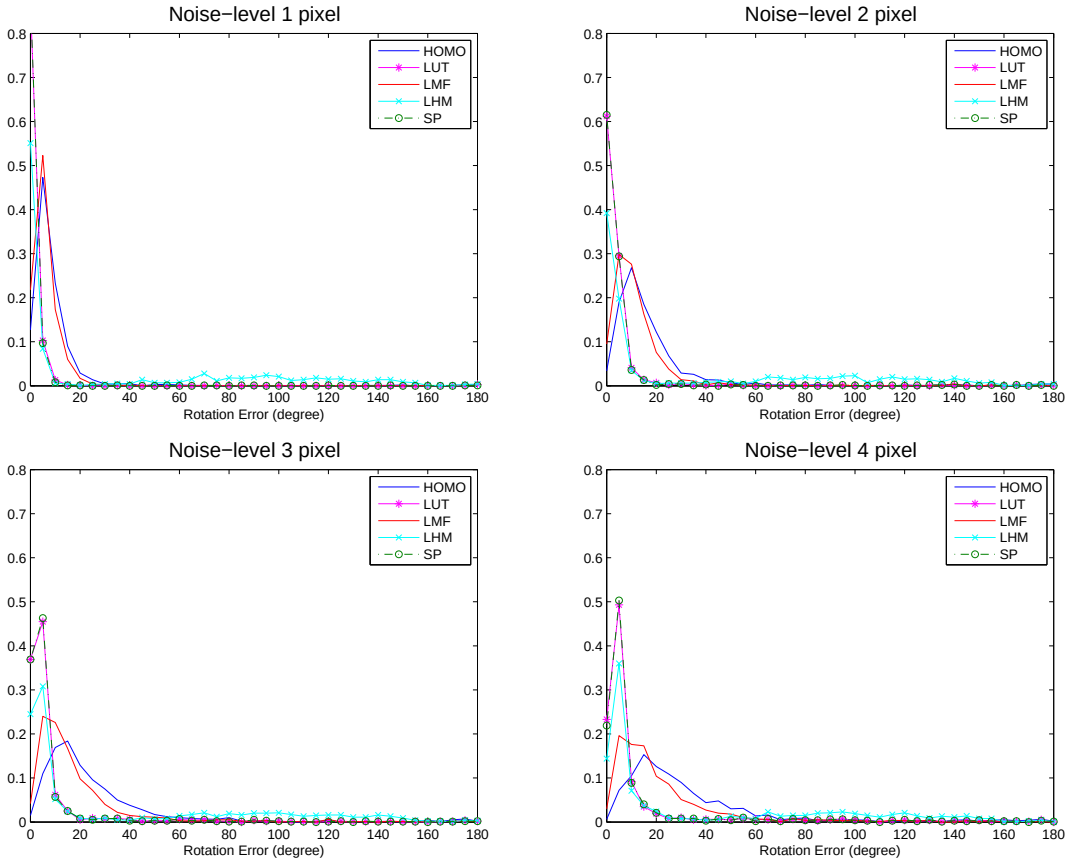

Figure 11: The histograms of the rotation error E_{rot} of the compared methods. The vertical axis of the figures denotes the percentage of the results.

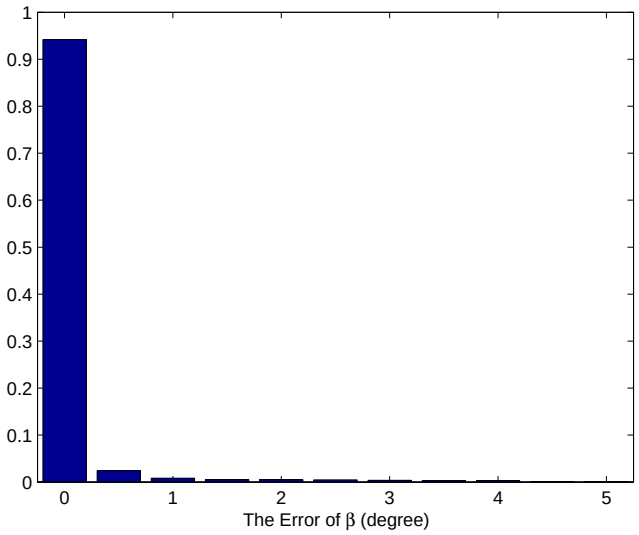

Figure 12: The histogram of the error of β caused by quantization approximation of the LUT.

the iterative method SP is much better than HOMO and LMF, whereas our approach can be as accurate as SP. The histograms of the rotation errors of each method are shown in Fig. 11. The histogram of our method is almost the same as that of SP and much better than the others.

4) Quantization approximation: 10,000 random tests were performed in noise-free cases to study the effect of the quantization approximation of the LUT. The histogram of the error of β looked up in the LUT is shown in Fig. 12. The average error is 0.1139 degrees, 94.5% of the errors are less than 0.25 degrees, and the maximum error is 5.658 degrees. The large errors are mainly located in the dangerous region shown in Fig. 5. This region is also dangerous for existing methods in noisy cases, in which a small noise in the coordinates of

the input points lead to a big change in the predicted camera pose, therefore lead to unstable result. In practice, noise is inevitable, and the effect of the quantization of LUT is negligible compared to the effect of noise.

Real images

We also tested our methods on real images captured from a common USB camera (image size 320×240) previously calibrated by the approach of Zhang [28]. The pose of the AR markers can be reliably estimated by LUT to enable a seamless alignment of virtual and real scenes. As shown in Fig. 13, a sub-table of LUT was selected and the 3D surface of its matrix data was augmented in the AR scene.

As shown in Fig. 14, each AR marker was augmented with a colored frame box corresponding to it. The AR markers were encoded using an approach similar to [15] and the camera poses were robustly estimated by our LUT method.

Following methods can be used to extract the square target in the scene: 1) Typically, the ARToolKit marker detector [12, 29] is a good choice for AR applications. 2) In the real environment, the target detector of Claus and Fitzgibbon [14] is one of the best choices because the targets can be reliably detected over a large range of scales, lighting and scene clutter, even when it is difficult for humans to identify the target. 3) If there are square textured patches in the scene, the well-known SURF [30] or FAST [31, 32] detectors can be used to identify the target for initialization, and then the edges of the box can be efficiently

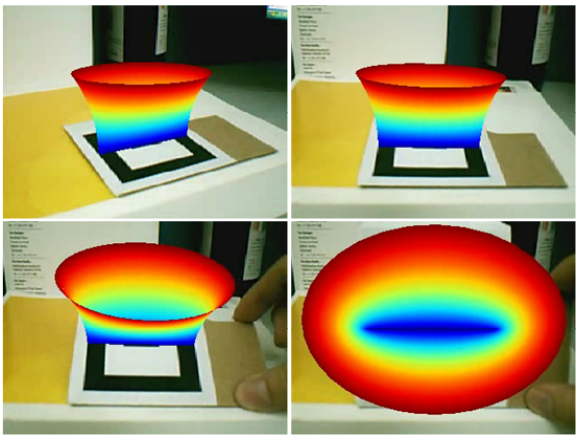

Figure 13: The 3D visualization of the LUT is augmented in AR scene. We selected the sub-table $\theta = 20$ and plotted the 3D surface of its matrix data. The height of the 3D surface over the underlying coordinates (u,v) equals the value of β stored in LUT.

tracked by using moving edge algorithm [33].

Conclusion

We have proposed an efficient LUT solution to estimate the pose of a calibrated camera from a square AR marker, which performs significantly more robustly and more accurately than existing non-iterative methods whenever there is noise. Compared to state-of-the-art iterative methods, ours is more robust and much more efficient with the same level of accuracy. The proposed LUT solution is suitable for applications that estimate camera poses from square markers, and is particularly suitable for mobile AR applications that require both high efficiency and high accuracy.

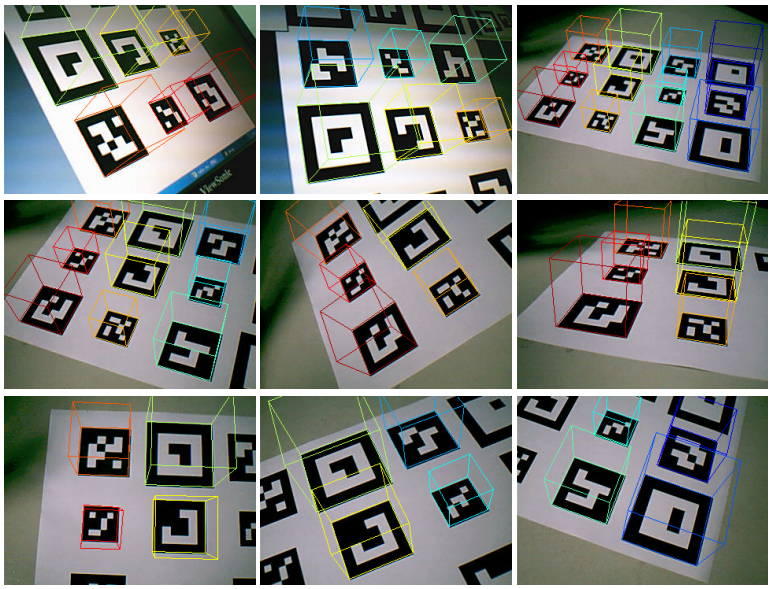

Figure 14: Some frames of the video sequence captured from a common USB camera (image size 320×240). The AR markers were augmented with colored frame boxes, and the camera poses were robustly estimated by our LUT method.

The quantization of LUT may lead to slight rotation error (about 0.1 degree) when there is no noise. Nevertheless, the noise is unavoidable in practice, and the effect of quantization approximation is negligible because the estimation error caused by quantization is always much smaller (about $1/30$) than the error caused by noise.

The main idea to improve the performance in the presence of noise lies in correctly extracting and fusing the ambiguous information from the input data. The square target can be divided into two symmetric un-overlapped triangles, and both triangles provide information of ambiguous poses. The classic non-iterative methods deal with all the input data (including information from correct pose and false pose) as a whole , so the incorrect information is fused in the final result. In this paper, the primary rotation angle β is used as a key parameter to identify the ambiguous poses, and then the information corresponding to each ambiguous pose can be extracted and fused independently without interfering with each other. As a result, the camera pose is effectively estimated without the disturbance of false information, and our method performs as accurately as the most accurate solution that can be achieved in this field.

Finally, let us consider a question: "Can non-iterative pose estimation solutions perform as accurately as iterative ones?" Ordinarily, iterative methods do much better in terms of accuracy than non-iterative ones in the presence of noise. Recently, the novel work of Lepetit, Moreno-Noguer and Fua showed that, when $n \geq 6$, non-iterative solutions can perform much faster than iterative ones with little loss of accuracy. However, when using a square AR marker with $n = 4$, the iterative methods still retains its advantage in accuracy

and stability. In this paper, we show that the well-known 4-point AR marker problem can be solved non-iteratively with high stability and high efficiency and is as accurate as the state-of-the-art iterative methods.

Acknowledgment

The authors would like to thank Dr. Vincent Lepetit and the anonymous reviewers for their helpful advice.

References

[1] D. Kalkofen, E. Mendez, D. Schmalstieg, Interactive Focus and Context Visualization for Augmented Reality, in: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007, pp. 1–10.

[2] X. Wang, P. Dunston, Compatibility issues in Augmented Reality systems for AEC: An experimental prototype study, Automation in Construction 15 (3) (2006) 314–326.

[3] J. Fischer, M. Eichler, D. Bartz, W. Straber, A hybrid tracking method for surgical augmented reality, Computers & Graphics 31 (2007) 39–52.

[4] J. Zhu, Z. Pan, C. Sun, W. Chen, Handling occlusions in video-based augmented reality using depth information, Computer Animation and Virtual Worlds.

[5] E. Jonietz, MIT Technology Review: Special Issue 10 Emerging Technologies (2007).

[6] G. Papagiannakis, G. Singh, N. Magnenat-Thalmann, A survey of mobile and wireless technologies for augmented reality systems, Computer Animation and Virtual Worlds 19 (1).

[7] G. Schweighofer, A. Pinz, Robust pose estimation from a planar target, IEEE transactions on pattern analysis and machine intelligence 28 (12) (2006) 2024–2030.

[8] M. Yuan, S. Ong, A. Nee, A generalized registration method for augmented reality systems, Computers & Graphics 29 (2005) 980–997.

[9] Graz, Handheld augmented reality project, http://studierstube.icg.tu-graz.ac.at/handheld_ar/ (2009).

[10] G. Klein, D. Murray, Parallel tracking and mapping for small AR workspaces, in: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, IEEE Computer Society, 2007, pp. 1–10.

[11] G. Klein, D. Murray, Parallel Tracking and Mapping on a camera phone, in: Proceedings of the 2009 8th IEEE International Symposium on Mixed and Augmented Reality, IEEE Computer Society, 2009, pp. 83–86.

[12] ARToolKit, Artoolkit home page, http://www.hitl.washington.edu/artoolkit/ (2003).

[13] ARTag, Artag home page, http://www.artag.net/ (2009).

[14] D. Claus, A. Fitzgibbon, Reliable Fiducial Detection in Natural Scenes, in: European Conference on Computer Vision (ECCV'04), 2004, p. 469.

[15] M. Fiala, ARTag, a fiducial marker system using digital techniques, in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), Vol. 2, 2005.

[16] X. Zhang, S. Fronz, N. Navab, Visual marker detection and decoding in AR systems: A comparative study, in: International Symposium on Mixed and Augmented Reality (ISMAR'02), 2002, pp. 97–106.

[17] S. Malik, G. Roth, C. Mcdonald, Robust 2D Tracking for Real-Time Augmented Reality, in: Proceedings of Vision Interface (VI), 2002.

[18] M. Abidi, T. Chandra, A new efficient and direct solution for pose estimation usingquadrangular targets: algorithm and evaluation, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (5) (1995) 534–538.

[19] M. Bujnak, Z. Kukelova, T. Pajdla, A general solution to the P4P problem for camera with unknown focal length, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08), 2008, pp. 1–8.

[20] A. Ansar, K. Daniilidis, Linear pose estimation from points or lines, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (5) (2003) 578–589.

[21] V. Lepetit, F. Moreno-Noguer, P. Fua, EPnP: An Accurate O (n) Solution to the PnP Problem, International Journal of Computer Vision 81 (2) (2009) 155–166.

[22] H. Araujo, R. Carceroni, C. Brown, A Fully Projective Formulation to Improve the Accuracy of Lowe's Pose-Estimation Algorithm, Computer Vision and Image Understanding 70 (2) (1998) 227–238.

[23] C. Lu, G. Hager, E. Mjolsness, Fast and globally convergent pose estimation from video images, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (6) (2000) 610–622.

[24] D. Wagner, D. Schmalstieg, Artoolkitplus for pose tracking on mobile devices, in: Proceedings of 12th Computer Vision Winter Workshop (CVWW'07), 2007, pp. 139–146.

[25] M. Rohs, Marker-based embodied interaction for handheld augmented reality games, Journal of Virtual Reality and Broadcasting 4 (5) (2007) 0009–6.

[26] G. Schall, E. Mendez, E. Kruijff, E. Veas, S. Junghanns, B. Reitinger, D. Schmalstieg, Handheld Augmented Reality for underground infrastructure visualization, Personal and Ubiquitous Computing 13 (4) (2009) 281–291.

[27] S. Umeyama, Least-squares estimation of transformation parameters between two-point patterns, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (4) (1991) 376–380.

[28] Z. Zhang, A flexible new technique for camera calibration, IEEE Transactions on pattern analysis and machine intelligence 22 (11) (2000) 1330–1334.

[29] Graz, Artoolkitplus home page, http://studierstube.icg.tu-graz.ac.at/handheld_ar/artoolkitplus.php (2006).

[30] H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (SURF), Computer Vision and Image Understanding 110 (3) (2008) 346–359.

[31] E. Rosten, T. Drummond, Machine learning for high-speed corner detection, Computer Vision–ECCV 2006 (2006) 430–443.

[32] S. Taylor, E. Rosten, T. Drummond, Robust feature matching in 2.3μs, in: Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on, IEEE, 2009, pp. 15–22.

[33] P. Bouthemy, A maximum likelihood framework for determining moving edges, IEEE transactions on pattern analysis and machine intelligence 11 (5) (1989) 499–511.