Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition Letters 28 (2007) 842–853
Pattern Recognition Letters
www.elsevier.com/locate/patrec

A recursive camera resectioning technique for off-line video-based augmented reality

Jong-Sung Kim, Ki-Sang Hong *

Electrical and Computer Engineering Division, POSTECH, San 31 Hyoja-Dong, Pohang 790-784, Republic of Korea

Received 27 July 2005; received in revised form 8 August 2006
Available online 23 January 2007

Communicated by T.K. Ho

Abstract

In this paper, we propose a new recursive framework for camera resectioning and apply it to off-line video-based augmented reality. Our method is based on an unscented particle filter and an independent Metropolis–Hastings chain, which deal with nonlinear dynamic systems without local linearization, and lead to more accurate results than other nonlinear filters. The proposed method has some desirable properties for camera resectioning: Since it does not rely on erroneous linear solutions, initialization problems do not occur, in contrast to the previous resectioning methods. Jittering error can be reduced by considering consistency and coherency between adjacent frames in our recursive framework. Our method is fairly accurate comparable to nonlinear optimization methods, which in general have higher levels of computation and complexity. As a result, the proposed algorithm outperforms the standard camera resectioning algorithm. We verify the effectiveness of our method through several experiments using synthetic and real image sequences comparing the estimation performance with other linear and nonlinear methods.

Keywords: Augmented reality; Camera resectioning; Unscented particle filter

1. Introduction

The fundamental process of augmented reality system is to merge virtual objects with images of a real environment for giving more information or making an augmented environment. The transformation between world space and image plane is represented by a camera matrix, and the augmented reality system utilizes the camera matrix for merging virtual objects with images. Camera tracking, which sequentially estimates the camera matrix of a viewing system, is an essential step of the augmented reality system. Since the accuracy of the camera tracking algorithm significantly affects the perceived accuracy of augmented environment, developing an optimal camera tracking algorithm is one of the most important issues in the augmented reality community.

In an on-line augmented reality system, camera tracking should be conducted as rapidly as possible. To reduce the computational complexity of the camera tracking algorithm in the on-line system, intrinsic parameters of the camera, i.e., focal length, aspect ratio, and skew, are estimated a priori by using any calibration process, and the estimate of the camera motion is sequentially updated by using any high-speed camera tracking algorithm (Davison, 2003; Pupilli and Calway, 2005). Camera tracking algorithms often make specific assumptions on the nature of camera motion or scene structure, e.g., Simon et al. (2000), and the flexibility of the on-line augmented reality system is limited by these assumptions. Contrary to on-line systems, in off-line systems, the accuracy of augmented environment is more important than the high-speed processing, and any assumption on the nature of camera

* Corresponding author. Tel.: +82 54 279 2216; fax: +82 54 279 5586.
E-mail addresses: kimjs@postech.ac.kr (J.-S. Kim), hongks@postech.ac.kr (K.-S. Hong).

0167-8655/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.patrec.2006.11.012

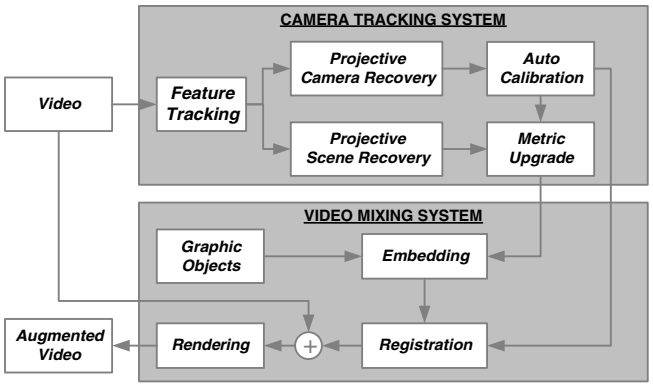

Fig. 1. An abstract framework for the off-line video-based augmented reality system. In this paper, we focus on the camera tracking module, and the frame-sampling based image sequence analysis and camera resectioning are accounted for the exact camera tracking.

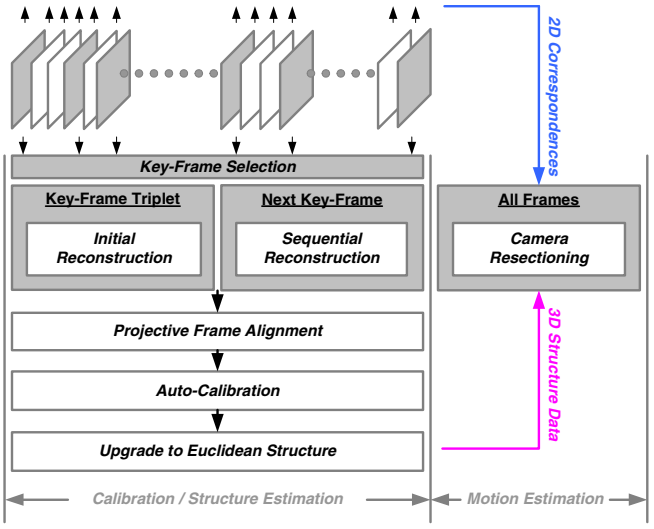

Fig. 2. Frame-sampling based auto-calibration, structure and motion analysis system in uncalibrated image sequences. It can be divided by two part; Euclidean scene structure estimation from key-frames and motion estimation by camera resectioning.

motion and scene structure is not utilized for the flexibility of the system.

An overview of the off-line video-based augmented reality system is illustrated in Fig. 1. Video-based off-line augmented reality is commonly used as a post-process for television or film special effects. In the off-line system, there is no need for the camera motion to be updated in real-time, and *auto-calibration techniques* (Pollefeys et al., 1999, 2002; Hartley and Zisserman, 2000, chapter 18) are used for camera calibration and metric upgrade with an unknown projective basis. *Bundle adjustment* (Triggs et al., 2000), a nonlinear method simultaneously optimizing camera motion and scene structure, is commonly used in the camera tracking module to distribute errors as evenly over video sequence as possible in order to acquire the high perceptual accuracy of an augmented environment. Although the bundle adjustment itself is well-developed, it has an essential problem; requiring an initial solution with small residual. Several approaches have been proposed for this problem, such as *Frame-sampling based approaches* (Fitzgibbon and Zisserman, 1998; Georgescu and Meer, 2002; Seo et al., 2003), adopted in this work, originated from considering this problem. In our system, illustrated in Fig. 2, video frames are classified into *key-frames* and *intra-frames*, and camera intrinsic parameters and scene structure are computed from the correspondence information between key-frames by using several computing steps. Camera motion over the entire video frames is computed based on camera intrinsic parameters and scene structure, estimated in the previous step. During this process, we attempt to acquire the camera motion that is consistent with the recovered camera geometry and scene structure. The main focus of this paper is on this process, which has never been directly considered in previous works.

Computing camera matrix from known or reconstructed scene structure and correspondences is called *camera calibration* or *camera resectioning* in the vision community (Faugeras, 1993; Hartley, 1995; Hartley and Zisserman, 2000; Forsyth and Ponce, 2002). Camera resectioning is frequently utilized in frame-sampling based structure and motion analysis (Fitzgibbon and Zisserman, 1998; Nister, 2001; Georgescu and Meer, 2002), and off-line video-based augmented reality (Gibson et al., 2002; Seo et al., 2003; Kim and Hong, 2004). Frame-sampling based approaches in general are composed of two parts; camera calibration and scene structure reconstruction from selected key-frames, and camera motion estimation over intra-frames or all frames based on the reconstructed scene structure, i.e., camera resectioning. In the projective space, a camera matrix can be linearly estimated up to an unknown projective transform, but in the Euclidean space the estimation of intrinsic parameters, i.e., focal length, skew, aspect ratio and optical center, and extrinsic parameters, representing the rotational and translational motion of camera, should be carefully considered. Linear methods, based on *singular value decomposition* (Golub and Van Loan, 1983), give reasonable solutions for projective camera resectioning if the data are correctly pre-processed (Hartley, 1995). Several linear methods have been proposed for Euclidean camera resectioning (Faugeras, 1993; Forsyth and Ponce, 2002). These methods attempt to calibrate cameras frame by frame without considering consistency and coherency between adjacent frames, and the camera parameters do not frequently have consistent values over all frames. This is one reason behind the creation of jittering errors in augmented reality. Without loss of generality, most off-line video-based augmented reality systems assume that the viewing geometry of camera is not changed. Technically, only a focal length change of the video camera is rendered by generic graphic machine, and other intrinsic parameters are required to be constant over all frames. A general remedy for this problem in previous methods was to fix the intrinsic parameters with known values and then re-estimate the unfixed intrinsic and extrinsic parameters

for the consistency of estimation results. For this purpose, a nonlinear error cost function, e.g., the weighted sum of squared error cost function (Triggs et al., 2000, p.10), is minimized by using iterative nonlinear optimization techniques (Pollefeys et al., 1999; Forsyth and Ponce, 2002). However, the optimization-based approach, called the nonlinear method in this work, has a drawback in that it only works when initial solutions are close to true ones as described in the previous paragraph. Besides, the nonlinear method is computationally expensive in general. Our research is motivated by this observation. We would like to develop a recursive framework to calculate camera parameters with coherency and consistency between every adjacent frames.

Contrary to other strategies, we consider a recursive framework which efficiently uses the latest information for prediction and update (Azerbayejani and Pentland, 1995; Chiuso et al., 2002). Our camera resectioning algorithm is based on the *unscented particle filter* (Julier and Uhlman, 1997; Wan and Merwe, 2000; Merwe et al., 2000; Merwe and Wan, 2003), which was presented as an alternative to the particle filter (Liu and Chen, 1998; Isard and Blake, 1998; Doucet et al., 2001) and the extended Kalman filter (Anderson and Moore, 1979), with the aim of achieving a better level of accuracy at a comparable level of complexity. The stability and accuracy of unscented particle filter have been proved by many researchers.

Our problem for camera resectioning is formulated with a dynamic state space model, where the camera intrinsic parameters are included into system parameters, and the extrinsic parameters are regarded as unknown system states. The unscented particle filter is applied to recursively estimate the states of system through two steps, predicting and updating the mean and the covariance of system states. The proposal distribution of our unscented particle filter is modelled with a mixture of Gaussian. To efficiently generate samples from the proposal distribution, we combine the *unscented Kalman filter* (Wan and Merwe, 2000) and the *independent Metropolis–Hastings chain* (Gilks et al., 1995) into our sampling procedure. In our framework, initialization problems do not occur, and the camera intrinsic parameters are not changed by the resectioning process. The consistency and coherency in the camera parameters are successfully accounted by using a probabilistic inference in our framework. In the statistical viewpoint, our method is a maximum a posterior estimator while the previous linear and nonlinear methods are maximum likelihood estimators. Several experiments are conducted with synthetic and real image sequences to demonstrate the effectiveness of our approach. The estimation performance of our approach is compared with several related approaches. It is illustrated that our method outperforms previous linear methods and other recursive methods, and the accuracy of our method is comparable with the accuracy of iterative nonlinear optimization methods.

The organization of this paper is as follows. Section 2 introduces our image sequence analysis system. Section 3 presents our dynamic state space model for the camera motion estimation problem. Section 4 describes our recursive framework for camera resectioning by using the unscented particle filter. Experimental results are given in Section 5 and our conclusions are drawn in Section 6.

2. Uncalibrated image sequence analysis

Our system for auto-calibration, structure and motion analysis in uncalibrated image sequences is presented in Fig. 2. Similar systems have been studied by Nister (2001) and Georgescu and Meer (2002). Our system can be divided into two parts: Euclidean scene structure estimation from key-frames and motion estimation by camera resectioning. In our system, selecting and tracking image features from image sequences are conducted by the *Kanade–Lucas–Tomasi algorithm* (Shi and Tomasi, 1994). We can automatically select key-frames by using *frame decimation algorithm* (Nister, 2001) or *key-frame selection algorithms* (Gibson et al., 2002; Seo et al., 2003). These automatic algorithms often create a redundancy in the key-frame selection according to the parameter setting, and the computation complexity of the reconstruction algorithm is increased by the redundancy in the frame selection process. In our experiments, we have selected a minimal number of key-frames in order to reduce the computation complexity of the scene structure estimation. From the firstly selected three key-frames, we estimate the initial scene structure and three projective cameras by using *trifocal tensor technique* (Hartley and Zisserman, 2000). Within the projective space, we sequentially merge the next key-frame to the first three key-frames by sequentially reconstructing the projective structure of the new key-frame based on the previously reconstructed scene structure. After completing the projective reconstruction from all key-frames in image sequences, we distribute the estimation error evenly over key-frames by using the projective bundle adjustment, and then we upgrade the projective reconstruction to the Euclidean reconstruction by using the linear auto-calibration based on the absolute dual quadric and the nonlinear auto-calibration refinement of uncertainty (Pollefeys et al., 2002). Then, we apply the Euclidean bundle adjustment to the upgraded Euclidean scene structure and camera parameters of key-frames in order to minimize the error in the metric-upgrade process.

From the correspondences and the scene structure computed in the structure estimation part, computing the motion of moving camera over all video frames is done by camera resectioning. The estimated camera parameters are used in the off-line augmented reality system developed in our laboratory. Linear methods for the camera intrinsic and extrinsic parameter estimation have been introduced in many vision materials (Faugeras, 1993; Hartley and Zisserman, 2000; Forsyth and Ponce, 2002). However, linear solutions are not adequate for the augmented reality system where estimated cameras should satisfy some requirements, i.e., zero skew, unit aspect ratio, and fixed optical

center. In the previous works this problem has never been directly considered. Our camera resectioning algorithm tries to solve this problem by using a dynamic state space model and a recursive filter. This approach has never been used for camera resectioning in the previous works.

3. Problem formulation

We adopt a dynamic state-space model with parameters to represent the dynamic motion of a camera acquiring video data. The global rotation $\Omega \in SO(3)$ and the global translation T of the video camera are defined as the dynamic states in the camera resectioning problem, and written as

$$\boldsymbol{x} = \{\Omega, T\}. \tag{1}$$

Associated to each global motion Ω and T, there are time-varying parameters, i.e., angular velocity ω, linear velocity v, angular acceleration $\dot{\omega}$ and linear acceleration $\dot{v}$, and static parameters, i.e., covariance matrix of angular acceleration $\Sigma_{\dot{\omega}}$, covariance matrix of linear acceleration $\Sigma_{\dot{v}}$, camera focal length f and N points of 3-D scene structure $S^{(1)}, \ldots, S^{(N)}$, reconstructed in the previous structure estimation module. We denote θ for the notation of all the above system parameters, and define it as

$$\theta = \{f, \omega, \dot{\omega}, v, \dot{v}, \Sigma_{\dot{\omega}}, \Sigma_{\dot{v}}, S^{(1)}, \ldots, S^{(N)}\}. \tag{2}$$

In our framework, the time evolution model for the states and parameters of the video camera is given by

$$f_{t+1} = f_t \tag{3}$$

$$S_{t+1}^{(i)} = S_t^{(i)} \quad i = 1, \ldots, N \tag{4}$$

$$\Omega_{t+1} = \log_{SO(3)}(\mathrm{e}^{\hat{\omega}_t}\mathrm{e}^{\Omega_t}) \tag{5}$$

$$T_{t+1} = \mathrm{e}^{\hat{\omega}_t} T_t + v_t \tag{6}$$

$$\omega_{t+1} = \omega_t + \dot{\omega}_t \quad \dot{\omega}_t \sim N(0, \Sigma_{\dot{\omega}}) \tag{7}$$

$$v_{t+1} = v_t + \dot{v}_t \quad \dot{v} \sim N(0, \Sigma_{\dot{v}}), \tag{8}$$

where $\hat{\omega}$ is the skew symmetric matrix of angular velocity ω and $\log_{SO(3)}$ the inverse of *Rodrigues' formula* (Chiuso et al., 2002). At time t, the measurement equation for the camera state $\boldsymbol{x}$ and the camera parameter θ is given by

$$\boldsymbol{y}_t = (x_t^{(1)}, y_t^{(1)}, \ldots, x_t^{(N)}, y_t^{(N)}) = \boldsymbol{h}(\boldsymbol{x}_t, \theta_t) + \boldsymbol{n}_t, \tag{9}$$

where the measurement noise $\boldsymbol{n}_t \sim N(0, \Sigma_{\boldsymbol{n}_t})$ and $\boldsymbol{h}(\cdot)$ is the $2N$-dimensional vector of corresponding nonlinear equation of the perspective camera projection, defined by

$$(x_t^{(i)}, y_t^{(i)})^{\mathrm{T}} = \left(f\frac{[Z_t^{(i)}]^1}{[Z_t^{(i)}]^3}, f\frac{[Z_t^{(i)}]^2}{[Z_t^{(i)}]^3} \right)^{\mathrm{T}}, \tag{10}$$

where $Z_t^{(i)} = \mathrm{e}^{\Omega_t} S_t^{(i)} + T_t$ and $[\cdot]^j$ denotes jth element. In our problem formulation, camera intrinsic parameters such as focal length, skew, aspect ratio and principal points are fixed with known values. Contrary to the previous approaches, this assumption makes no significant error in our recursive camera resectioning algorithm.

4. Recursive camera resectioning

A recursive framework is used in our resectioning algorithm in order to efficiently use the latest information for consistency and coherency in the camera motion. Our algorithm is based on the unscented particle filter, which was presented as an alternative to the extended Kalman filter and achieves a better level of accuracy at a comparable level of complexity. The unscented particle filter is applied to recursively estimate the global motion of the video camera through two steps, predicting and updating the mean and the covariance of system state. Consistency and coherency in the camera motion is successfully accounted for by using the probabilistic inference capability of the unscented particle filter.

4.1. Propagating mean and covariance

Unscented transform (Julier and Uhlman, 1997) is a method for propagating mean and covariance with second order accuracy in a nonlinear system. This transform was applied to the extended Kalman filter and called as the unscented Kalman filter (UKF) by Wan and Merwe (2000). Fig. 3 is included to visually illustrate the idea of the unscented transform and to support the understanding of the UKF-based part of our algorithm. In this algorithm, the mean and the covariance of the n-dimensional state are represented with $2n+1$ weighted samples, called *sigma points*.

The mean and the covariance of the camera system state $\boldsymbol{x}$ in Eq. (1), predicted by using the UKF algorithm, are denoted $\bar{\boldsymbol{x}}^{\mathrm{UKF}}$ and Σ^{UKF}, respectively, in our framework. At time t our UKF-based prediction steps can be described in the following way:

Step 1: *Calculate* $2n+1$ sigma points $\{X_{t-1}^{(0)}, \ldots, X_{t-1}^{(2n)}\}$ of the camera system state by using $\boldsymbol{x}_{t-1}$ and Σ_{t-1}, the mean and the covariance of the camera system state at time $t-1$, respectively:

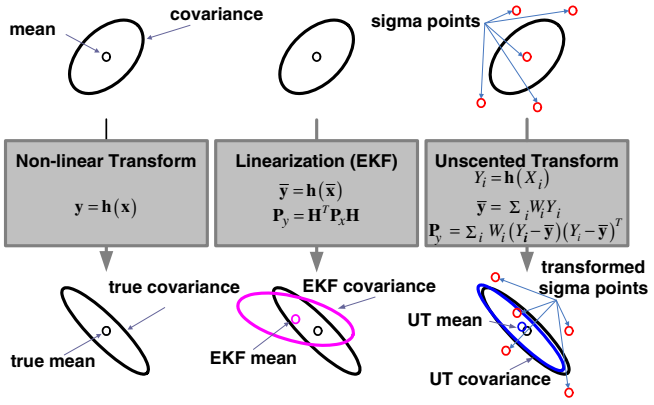

Fig. 3. Mean and covariance propagation: nonlinear transform (left), extended Kalman filter (center), unscented transform (right) (Julier and Uhlman, 1997).

$$X_{t-1}^{(i)} = \begin{cases} \bar{\boldsymbol{x}}_{t-1}^{\mathrm{UKF}} & i = 0 \\ \bar{\boldsymbol{x}}_{t-1}^{\mathrm{UKF}} + \left(\sqrt{(n+k)\Sigma_{t-1}^{\mathrm{UKF}}}\right)^{(i)} & 1 \leqslant i \leqslant n \\ \bar{\boldsymbol{x}}_{t-1}^{\mathrm{UKF}} - \left(\sqrt{(n+k)\Sigma_{t-1}^{\mathrm{UKF}}}\right)^{(i)} & n+1 \leqslant i \leqslant 2n \end{cases} \tag{11}$$

and corresponding weights are computed by

$$W^{(i)} = \begin{cases} k/(n+k) & i = 0 \\ 1/2(n+k) & o.w., \end{cases} \tag{12}$$

where $\left(\sqrt{A}\right)^{(i)}$ is ith singular vector of the matrix A, $k = 2$ is used as a scaling factor for weights, and n is the dimension of the state vector.

Step 2: *Predict* the mean and the covariance of the camera system state from $2n+1$ sigma points $\{X_{t-1}^{(0)}, \ldots, X_{t-1}^{(2n)}\}$ using the time evolution model in Eqs. (3)–(8) (we denote a symbol g to represent that evolution model), and the measurement equation in Eq. (9):

$$X_{t|t-1}^{(i)} = g(X_{t-1}^{(i)}, \theta_t) \quad i = 0, \ldots, 2n \tag{13}$$

$$\bar{\boldsymbol{x}}_{t|t-1} = \sum_{i=0}^{2n} W^{(i)} X_{t|t-1}^{(i)} \tag{14}$$

$$\Sigma_{t|t-1}^{xx} = \sum_{i=0}^{2n} W^{(i)} (X_{t|t-1}^{(i)} - \bar{\boldsymbol{x}}_{t|t-1})(X_{t|t-1}^{(i)} - \bar{\boldsymbol{x}}_{t|t-1})^{\mathrm{T}} \tag{15}$$

$$Y_{t|t-1}^{(i)} = \boldsymbol{h}(X_{t|t-1}^{(i)}, \theta_t) + \boldsymbol{n}_t \tag{16}$$

$$\bar{\boldsymbol{y}}_{t|t-1} = \sum_{i=0}^{2n} W^{(i)} Y_{t|t-1}^{(i)}. \tag{17}$$

Step 3: *Update* the predicted mean and the predicted covariance by innovation information:

$$\Sigma_{t|t-1}^{yy} = \sum_{i=0}^{2n} W^{(i)} (Y_{t|t-1}^{(i)} - \bar{\boldsymbol{y}}_{t|t-1})(Y_{t|t-1}^{(i)} - \bar{\boldsymbol{y}}_{t|t-1})^{\mathrm{T}} \tag{18}$$

$$\Sigma_{t|t-1}^{xy} = \sum_{i=0}^{2n} W^{(i)} (Y_{t|t-1}^{(i)} - \bar{\boldsymbol{y}}_{t|t-1})(X_{t|t-1}^{(i)} - \bar{\boldsymbol{x}}_{t|t-1})^{\mathrm{T}} \tag{19}$$

$$K_t = \Sigma_{t|t-1}^{xy} \left(\Sigma_{t|t-1}^{yy}\right)^{-1} \tag{20}$$

$$\bar{\boldsymbol{x}}_t^{\mathrm{UKF}} = \bar{\boldsymbol{x}}_{t|t-1} + K_t(\boldsymbol{y}_t - \bar{\boldsymbol{y}}_{t|t-1}) \tag{21}$$

$$\Sigma_t^{\mathrm{UKF}} = \Sigma_{t|t-1}^{xx} - K_t \Sigma_{t|t-1}^{yy} K_t^{\mathrm{T}}. \tag{22}$$

The propagated mean and covariance $\bar{\boldsymbol{x}}_t^{\mathrm{UKF}}$ and Σ_t^{UKF} are used to represent the modes of the proposed distribution, which is called *UKF proposal distribution* (Merwe et al., 2000), generating particles for the probabilistic inference of the system state. The procedure for the probabilistic inference is described in the following section.

4.2. Sequential probabilistic inference

In Fig. 4, we show the block diagram of the unscented particle filter for the recursive filtering of the global camera motion state. The unscented particle filter in Fig. 4 is

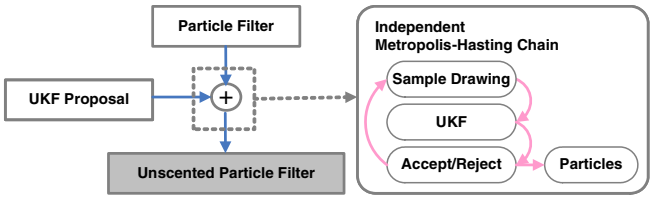

Fig. 4. Block diagram of the unscented particle filter with independent Metropolis–Hastings chain.

achieved by combining *sequential importance sampling* (Liu and Chen, 1998) and the unscented Kalman filter in order to improve the performance of the basic particle filter and overcome the drawbacks of other nonlinear filtering methods.

For dynamic systems, the importance proposal can be modelled with a mixture of Gaussian distributions and are obtained by a bank of unscented Kalman filters, which was discussed in the paper of Merwe and Wan (2003). Because a moving video camera is a dynamic system, we adopt the Gaussian mixture proposal distribution to represent the multi-modal probability distribution. This approach has another advantage to allow the reduction of the number of samples. Our importance proposal distribution, a mixture of M Gaussian distributions, is given by

$$g_t(\boldsymbol{x}_t | \boldsymbol{x}_{0:t-1}) = \sum_{j=1}^{M} w_{t-1}^{(j)} N\left(\boldsymbol{x}_t | \bar{\boldsymbol{x}}_t^{(j),\mathrm{UKF}}, \Sigma_t^{(j),\mathrm{UKF}}\right), \tag{23}$$

where the mixing weight of jth mode, $w_{t-1}^{(j)}$, is sequentially updated through the updating rule of the unscented particle filter. We introduce an auxiliary random variable J, which takes values from the set $\{1, \ldots, M\}$. Drawing a sample (J, X_t) according to the proposal in Eq. (23) can be realized with the *rejection algorithm* (Liu and Chen, 1998). However, it is a well-acknowledged fact that the rejection algorithm is restrictive and inefficient, because the sampling procedure should be iterated until the generated sample is accepted. To improve the efficiency and the convergence of the sampling procedure, we adopt the *independent Metropolis–Hastings chain* (IMHC) (Gilks et al., 1995). We define k as the index of IMHC iteration. At kth iteration, our *UKF & IMHC sampling procedure* has the following seven steps:

Step 1: *Select* $J_k = j$ with probability proportional to the weighting factor $w_{t-1}^{(j)}$, which is represented with the cumulative distribution function $C(J)$, given by

$$C(J) = \sum_{j=1}^{J} w_{t-1}^{(j)}. \tag{24}$$

Step 2: *Propagate* the mean $\bar{\boldsymbol{x}}_{t-1}^{(j),\mathrm{UKF}}$ and the covariance $\Sigma_{t-1}^{(j),\mathrm{UKF}}$ with UKF, described in Section 4.1.

Step 3: *Generate* $X_t = \boldsymbol{x}_{t,k}$ from the proposal distribution written by

$$g_t^{(j)}(\boldsymbol{x}_t|\boldsymbol{x}_{0:t-1}) = N\left(\boldsymbol{x}_t|\bar{\boldsymbol{x}}_t^{(j),\mathrm{UKF}}, \Sigma_t^{(j),\mathrm{UKF}}\right). \tag{25}$$

Step 4: *Generate* a uniform (0, 1) random variable U.
Step 5: *Compute* the acceptance probability with

$$p_a(\boldsymbol{x}_{t,k}, \boldsymbol{x}_{t,k-1}) = \min\left[1, \frac{f(\boldsymbol{y}_t|\boldsymbol{x}_{t,k}, \theta_t) w_{t-1}^{(J_{k-1})}}{f(\boldsymbol{y}_t|\boldsymbol{x}_{t,k-1}, \theta_t) w_{t-1}^{(J_k)}}\right], \tag{26}$$

where $f(\boldsymbol{y}_t|\boldsymbol{x}_t, \theta_t)$ is a likelihood function, given by

$$f(\boldsymbol{y}_t|\boldsymbol{x}_t, \theta_t) = \exp\{-(\boldsymbol{y}_t - \tilde{\boldsymbol{y}}_t)^{\mathrm{T}} \Sigma_{\boldsymbol{n}_t}^{-1} (\boldsymbol{y}_t - \tilde{\boldsymbol{y}}_t)\}. \tag{27}$$

Step 6: *Accept* $\boldsymbol{x}_{t,k}$, if $U \geqslant p_a(\boldsymbol{x}_{t,k}, \boldsymbol{x}_{t,k-1})$. Otherwise, set $\boldsymbol{x}_{t,k}$ equal to $\boldsymbol{x}_{t,k-1}$.
Step 7: *Stop* the iteration if $k \geqslant k_{\mathrm{th}}$. Otherwise, increase k and go to Step 1.

This sampling procedure has many preferable properties. It achieves re-sampling effect automatically and also avoids weight estimation. Re-sampling is necessary to evolve the system state for time t to $t+1$ and to prevent the proposal distribution from becoming skewed. The flow chart of our sampling procedure is illustrated in Fig. 5. The number of IMHC iterations is limited by k_{th} for practical realization. This procedure is repeatedly applied to acquire M samples, $\boldsymbol{x}_t^{(1)}, \ldots, \boldsymbol{x}_t^{(M)}$, corresponding to the modes of the probability distribution of camera system state at time t. The estimation of camera motion can be simply computed with the weighted average of M samples. Our recursive camera resectioning algorithm can be summarized in the following way:

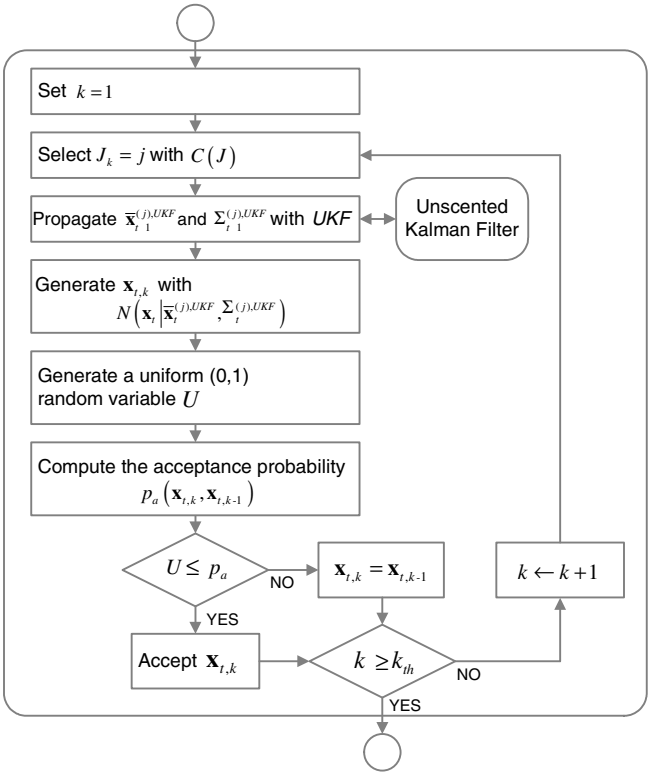

Fig. 5. Our sampling procedure based on unscented Kalman filter and independent Metropolis–Hastings chain algorithm.

Algorithm 1. Recursive Camera Resectioning Algorithm

(1) Iterate for $i = 1, \ldots, M$
(1.1) Draw $(J, X_t) = (j, \boldsymbol{x}_t^{(i)})$ with UKF & IMHC sampling procedure.
(1.2) Compute the incremental weight as

$$u_t^{(i)} = f(\boldsymbol{y}_t|\boldsymbol{x}_t^{(i)}, \theta_t) g_t(\boldsymbol{x}_t^{(i)}|\boldsymbol{x}_{0:t-1}^{(j)}). \tag{28}$$

(1.3) Let $w_t^{(i)} = u_t^{(i)} w_{t-1}^{(j)}$.
(2) Normalize so that $\sum_j w_t^{(i)} = 1$.
(3) Estimate camera motion as

$$E[X_t|Y_t] \approx \sum_{i=1}^{M} w_t^{(i)} \boldsymbol{x}_t^{(i)}. \tag{29}$$

The accuracy of our method depends on the number of samples, M, in the importance proposal distribution, modelled with a mixture of Gaussian distributions and obtained by a bank of UKF. As M is increased, the estimation performance of our algorithm can be significantly improved. Although we have a computational burden in the bank of UKF, where the 6×6 covariance matrix of each mode should be decomposed to acquire sigma points, our method dramatically reduces the number of particles and acquire a higher level of accuracy in the estimation results. Our method corresponds to a maximum a posterior estimator, statistically superior to maximum likelihood estimators adopted in linear and nonlinear resectioning methods. Consequently, initialization problems do not occur in our recursive framework, and the camera intrinsic parameters, representing the camera geometry, are not changed, which is important for stabilizing the perceptual accuracy of the video-based augmented reality.

5. Experimental results

Numerical experiments were conducted with synthetic and real image sequences to demonstrate the effectiveness of our recursive camera resectioning algorithm. All computations were carried out on a desktop PC with a 2.4-GHz Intel P4 CPU. In order to show the effectiveness of our method, we compared it with several different nonlinear filters available for recursive camera resectioning: the particle filter (PF), the particle filter with independent Metropolis–Hastings chain (PF-IMHC), the extended Kalman filter (EKF), and the unscented Kalman filter (UKF), as well as with the linear and nonlinear methods for camera resectioning. For comparison, the system model in Section 3 was used in other nonlinear filters as in our method. Without much loss of generality, we assumed that $\Sigma_{\dot{\omega}} = \sigma_{\dot{\omega}}^2 I$, $\Sigma_{\dot{v}} = \sigma_{\dot{v}}^2 I$ and $\Sigma_{\boldsymbol{n}} = \sigma_{\boldsymbol{n}}^2 I$. In all experiments,

the initial state vector was set with $\boldsymbol{x}_0 = \mathbf{0}$, and the initial system covariance with $\Sigma_0 = \mathrm{diag}(\Sigma_{\dot{\omega}}, \Sigma_{\dot{v}})$. The IMHC iteration number, k_{th}, was set to five in both PF-IMHC and our method, this is, the IMHC procedure iterates five times to generate a single sample from a proposal distribution. In some experiments, we tested a single Gaussian distribution, i.e., $M = 1$, for the proposal distribution of our method, instead of the mixture of M Gaussian distributions. When $M = 1$, the iterative IMHC sampling was bypassed to remove unnecessary computations. In this case, our method is equal to UKF.

The different methods were implemented as follows. The linear and nonlinear methods for camera resectioning were implemented by following the process for geometric camera calibration (Faugeras, 1993, chapter 3; Forsyth and Ponce, 2002, chapter 3). In the linear method, we first estimated the projective camera matrix using the linear least-squares method, and then estimated the intrinsic and extrinsic parameters from the projective camera matrix using the formulas proposed by Faugeras (1993). In the nonlinear method, we optimized the entries of the projective camera matrix before estimating the camera parameters. Then, we estimated the intrinsic and extrinsic parameters from the optimized projective camera matrix. Lastly, we set the intrinsic parameters with known values as we stated in Section 1, and then optimized the remaining extrinsic parameters. In the numerical implementation of the nonlinear method, we employed a weighted sum of squared error cost function (Triggs et al., 2000, p. 10) to formulate the cost function for camera resectioning, and then searched a minimum of the cost function using the Gauss–Newton and Levenberg–Marquard algorithm while controlling the step size using the line search method to guarantee convergence to a minimum (Triggs et al., 2000, pp. 14–17). The generic algorithms for EKF (Welch and Bishop, 2001) and PF (Merwe et al., 2000) were used in our experiments.

In order to quantitatively evaluate the performance of camera resectioning, we have computed the reprojection error, i.e., the Root-Mean-Squares (RMS) error in the unit of pixel, defined by

$$\text{RMS error} = \sqrt{\frac{1}{N}\|\boldsymbol{y}_t - \boldsymbol{h}(\hat{\boldsymbol{x}}_t, \theta_t)\|^2},$$

where $\|\cdot\|$ denotes the l^2-norm of a vector. To examine the estimation error of each method, we have computed the relative l^2 error of the rotation estimate, e_r, defined by

$$e_r = \frac{\|\hat{\Omega}_{1:t} - \Omega_{1:t}\|}{\|\Omega_{1:t}\|},$$

where $\hat{\Omega}_{1:t}$ and $\Omega_{1:t}$ denote the estimate and the ground truth of the global rotation, respectively, from times *1* to t. In like manner, we have also computed the relative l^2 error of the translation estimate, e_t, from the estimate and the ground truth of the global translation. Note that ground truth is only available for simulation.

Through the simulation and real experimental results, it is validated that our method outperforms other nonlinear filters as well as the linear method, and the estimation performance of our method is comparable to the nonlinear method.

5.1. Simulation results

In the first experiment, we show that in the camera resectioning problem, our method is more accurate and cost-efficient than the other particle filters. Our method was compared to PF and PF-IMHC in terms of the error and computation time relative to the number of samples, M. Note that our method uses the UKF & IMHC sampling procedure described in Section 4.2, but PF and PF-IMHC use the random sampling and the independent Metropolis–Hastings chain, respectively. We synthesized a camera motion sequence of 512×512 pixels and 100 frames simulating a freely moving camera viewing a 3-D object. We set the focal length of the camera to 1.0, corresponding to 53° field of view and 512 in the unit of pixel, the skew to zero, and the optical center equal to the image center, i.e., $(256, 256)$. The initial motion was set to zero. The 3-D object was composed of 100 points on a sphere. The radius of the sphere was set to 1.0, and the distance from the camera to the center of the sphere was set to 4.0, where we can figure out that the object is half the size of the image at the first frame. Gaussian random noise was added to each coordinate of image point. The standard deviation of noise was set to 0.1 pixel. In all methods, the system parameters in Eq. (2) were set with known values.

Table 1 shows the number of samples M, the average, minimum, and maximum values of the RMS error in pixels, the relative l^2 error, and the computation time in seconds. To obtain the statistical value of the errors, all simulation results were averaged over 10 test runs. The computation time was the total computation time for estimating a complete camera motion of 100 frames. Note that we used the same system model and set the system parameters with known values in all methods. The results show that the superior performance of the UKF & IMHC sampling procedure of our method is clearly evident. Therefore, we can conclude that our method can perform the camera resectioning with higher accuracy and smaller samples at the same time than other particle filters. We could improve the estimation accuracy of PF about 2 times using IMHC. The increase of the computational cost to adopt IMHC can be supplemented by reducing the number of samples as in PF-IMHC and our method. Notice that our method uses a bank of UKFs for proposal distribution generation, which explains why our method shows the performance superior to PF-IMHC as well as PF. It is worth noting that our method could conduct the camera resectioning using a single sample. As we stated above, our method works exactly like UKF when the numbers of samples is set to one, i.e., $M = 1$. The computation time of our method with $M = 1$ was 3.9 s.

Table 1
Comparison of the RMS error, relative l^2 error and computation time relative to the number of samples on the simulation data

Method	M	RMS error (pixel)			Rel. l^2 error (%)		Time (s)
		Avg.	Min.	Max.	e_r	e_t	
PF	1000	1.09	0.44	2.08	6.30	11.25	143.7
	750	1.19	0.34	2.24	6.67	11.72	85.9
	500	1.51	0.52	2.73	8.62	15.35	42.3
	250	2.20	0.82	4.02	12.71	22.66	13.9
PF-IMHC	500	0.70	0.26	1.26	4.23	7.56	244.2
	250	0.89	0.30	1.70	5.17	9.22	85.9
	100	1.40	0.46	2.62	7.92	14.14	25.9
	50	2.07	0.66	3.68	12.14	21.48	11.6
Our method	25	0.06	0.02	0.16	0.15	0.30	96.2
	10	0.09	0.02	0.24	0.17	0.37	38.0
	5	0.12	0.02	0.34	0.20	0.48	19.7
	1	0.19	0.04	0.72	0.27	0.72	3.9

In the second experiment, different camera resectioning methods using PF, EKF, the linear method and the nonlinear method, were compared to our method in terms of the RMS error, the relative l^2 error, and the computation time. In order to analyze the estimation error with respect to the variation of measurement noise, each algorithm was tested using a camera motion sequence added measurement noise. For the noise simulation, a Gaussian noise with $N(0, \sigma_n^2 I)$ is added to each image point as in the fist experiment, where $\sigma_n = 0.1$, 0.5, 0.7, and 1.0. The error-free case, i.e., $\sigma_n = 0$, was not considered for the even comparison (in this ideal case, the linear method does not have any error, but numerical error). The other simulation parameters were set as in the first experiment. The same system model and the known system parameters were used in all methods.

Table 2 shows the camera resectioning results of each algorithm for four noise levels. To obtain the statistical value of the errors, the experiment was repeated 100 times over 100 frames for four noise levels. The computation time of each method was also listed for the comparison of computational complexities. The results demonstrate

Table 2
Comparison of the RMS error and relative l^2 error relative to measurement noise on the simulation data

Method	Noise	RMS error (pixel)			Rel. l^2 error (%)		Time (s)
	σ_n	Avg.	Min.	Max.	e_r	e_t	
PF	0.1	1.50	0.54	2.80	8.53	15.23	42.7
	0.4	1.53	0.53	2.75	8.65	15.40	
	0.7	1.53	0.54	2.82	8.71	15.49	
	1.0	1.55	0.55	2.80	8.78	15.58	
EKF	0.1	1.07	0.08	3.56	1.24	2.65	2.2
	0.4	1.07	0.11	3.48	1.29	2.75	
	0.7	1.10	0.15	3.54	1.48	2.91	
	1.0	1.10	0.18	3.58	1.63	3.13	
Linear method	0.1	0.59	0.09	1.58	0.58	1.01	0.5
	0.4	2.41	0.36	6.20	2.34	4.11	
	0.7	4.20	0.61	11.01	4.08	7.13	
	1.0	5.93	0.94	15.75	5.79	10.01	
Nonlinear method	0.1	0.02	0.01	0.04	0.11	0.19	61.0
	0.4	0.09	0.03	0.17	0.44	0.77	
	0.7	0.16	0.06	0.30	0.76	1.34	
	1.0	0.23	0.08	0.43	1.10	1.91	
Our method ($M = 1$)	0.1	0.19	0.04	0.79	0.27	0.71	3.8
	0.4	0.21	0.05	0.81	0.49	1.00	
	0.7	0.26	0.08	0.84	0.78	1.46	
	1.0	0.31	0.10	0.86	1.09	1.96	
Our method ($M = 10$)	0.1	0.09	0.02	0.25	0.16	0.35	38.0
	0.4	0.13	0.04	0.28	0.44	0.80	
	0.7	0.18	0.06	0.36	0.75	1.34	
	1.0	0.25	0.09	0.47	1.07	1.88	

the gradual degradation in the performance of our method with increased noise. The averaged RMS error of our method with $M = 10$ was 6–17 times smaller than PF, and 4–10 times smaller than EKF. As we used more samples, i.e., more UKFs for the proposal distribution, our method produced more accurate results. The computation time of our method was linearly increased with the number of samples. When we used 10 samples for our method, we could reduce the averaged RMS errors by 20–50% and the maximum RMS errors by 40–70%. The table demonstrates that the linear method is very sensitive to a small amount of measurement noise, unlike other methods. In the linear method, the intrinsic camera parameters did not satisfy the constraints; zero skew, unit aspect ratio, and fixed optical center. Besides, the extrinsic camera parameters were inaccurate even at the moderate level of noise. The RMS and relative l^2 errors of the linear method were 3–10 times larger than our method. The table shows that the nonlinear method effectively solved the problems of the linear method. The most accurate results were obtained using the nonlinear method. However, the difference in the performance between our method and the nonlinear method rapidly decreased as the measurement noise increased. Notice that only our method and the nonlinear method among test algorithms achieved sub-pixel accuracy for all noise levels, and the computation time of our method with $M = 1$ was 3.8 s.

5.2. Real sequence results

We tested our method with three real image sequences and compared our results with others. The first test sequence has 90 frames of 720×480 pixels, the second one 100 frames of 704×480 pixels, and the third one 120 frames of 704×480 pixels. The images of the first frame from the test sequences 1, 2, and 3 are shown in Fig. 6. For each test sequence, we applied the auto-calibration and structure estimation module of our system described in Section 2 to automatically calibrate the video camera and estimate the scene structure. The system was set to select the first, last, and every 10th frames as key-frame. The feature tracking algorithm was set to track less than 100 image features at each frame.

Table 3 shows the total number of selected key-frames, the focal length estimate in pixels, the number of reconstructed 3-D points in the scene structure estimate, and the average number of tracked image features per frame. The auto-calibration and structure estimation results were used as an input to the camera resectioning module of the system estimating the camera rotation and translation over all frames of corresponding sequence. For comparison, we implemented the camera resectioning module with five different methods; PF, EKF, the linear method, the nonlinear method, and our method. We used 500 samples in PF, but 1 or 10 samples in our method. Angular acceleration, linear acceleration, and measurement noise components were set to $\sigma_{\dot{\omega}} = 0.007$, $\sigma_{\dot{v}} = 0.004$ and $\sigma_{\boldsymbol{n}} = 1.0$, respectively, in common for all test sequences.

Table 3
Auto-calibration and structure estimation results for the test sequences

Sequence	Key-frame	f (pixel)	Structure	Feature
Sequence 1	10	993.2	168	87
Sequence 2	11	868.7	178	57
Sequence 3	13	904.4	237	61

Table 4 shows the RMS error and computation time for all five implementations. To obtain the statistical value of the RMS error, PF, EKF and our method iterated 10 times for each test sequence. PF showed poor performance although it used 500 samples, 50–500 times more than our method. The RMS error of PF was about 4–7 times larger than our method. EKF showed a better performance than PF, but worse than our method. The RMS error of EKF was sub-pixel in all test sequences, but the maximum errors exceeded two pixels in the sequences 2 and 3. Our method with $M = 10$ or 1 was faster than the nonlinear method in all tests. Our method with $M = 1$ was slower about 1.6 times, but more accurate than EKF. The maximum error of our method with $M = 1$ exceeded one pixel in the sequence 2. As we see in the table, our method was accurate comparable to the nonlinear method in contrast to other methods. In addition, the results show that our method makes it possible to control the tradeoff between accuracy and computation time by adjusting the number of samples.

In Fig. 7, we plotted the RMS error of EKF, the nonlinear method, and our method with $M = 10$. We did not plot the RMS error corresponding to PF and the linear method since the error values of these two methods were too large. Our method was more accurate and stable than EKF, and also as accurate as the nonlinear method. Note that our

Fig. 6. Images of the first frame from the test sequences 1, 2, and 3 in order.

Table 4
Comparison of the RMS error and computation time among methods on the three test image sequences

Sequence	Method	RMS error (pixel)			Time (s)
		Avg.	Min.	Max.	
Sequence 1	PF	5.05	0.67	7.82	38.2
	EKF	0.51	0.35	0.91	1.5
	Linear method	13.3	1.22	31.6	0.4
	Nonlinear method	0.47	0.34	0.63	81.8
	Our method ($M = 1$)	0.47	0.35	0.64	2.4
	Our method ($M = 10$)	0.46	0.34	0.62	25.4
Sequence 2	PF	2.81	1.08	7.16	42.4
	EKF	0.84	0.41	2.08	0.6
	Linear method	13.8	2.09	36.3	0.3
	Nonlinear method	0.61	0.39	1.02	81.7
	Our method ($M = 1$)	0.66	0.39	1.56	0.9
	Our method ($M = 10$)	0.61	0.38	1.00	10.1
Sequence 3	PF	3.52	0.93	13.0	52.0
	EKF	0.99	0.47	3.29	0.9
	Linear method	21.6	0.62	56.6	0.4
	Nonlinear method	0.58	0.45	0.84	105.4
	Our method ($M = 1$)	0.59	0.43	0.90	1.5
	Our method ($M = 10$)	0.57	0.42	0.82	17.2

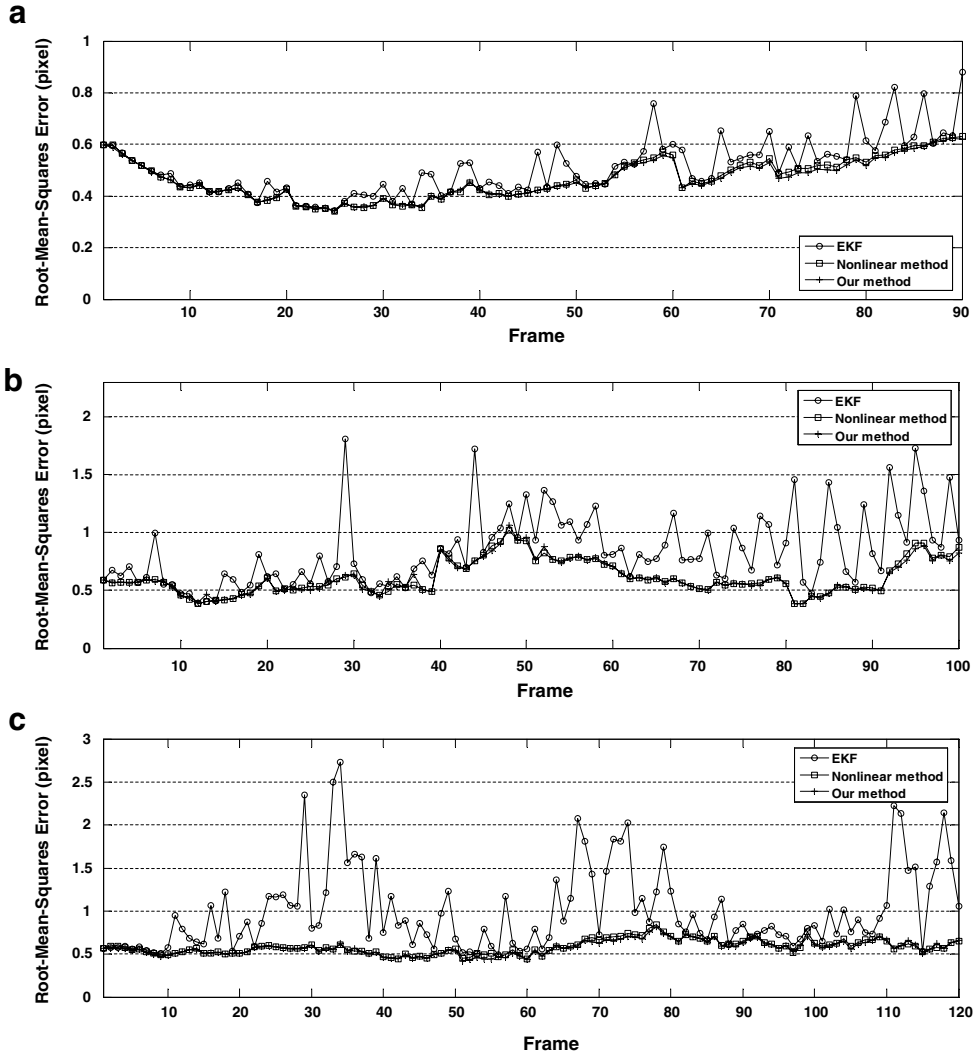

Fig. 7. Plots of the RMS error in the camera motion estimated from (a) Sequence 1, (b) Sequence 2, and (c) Sequence 3.

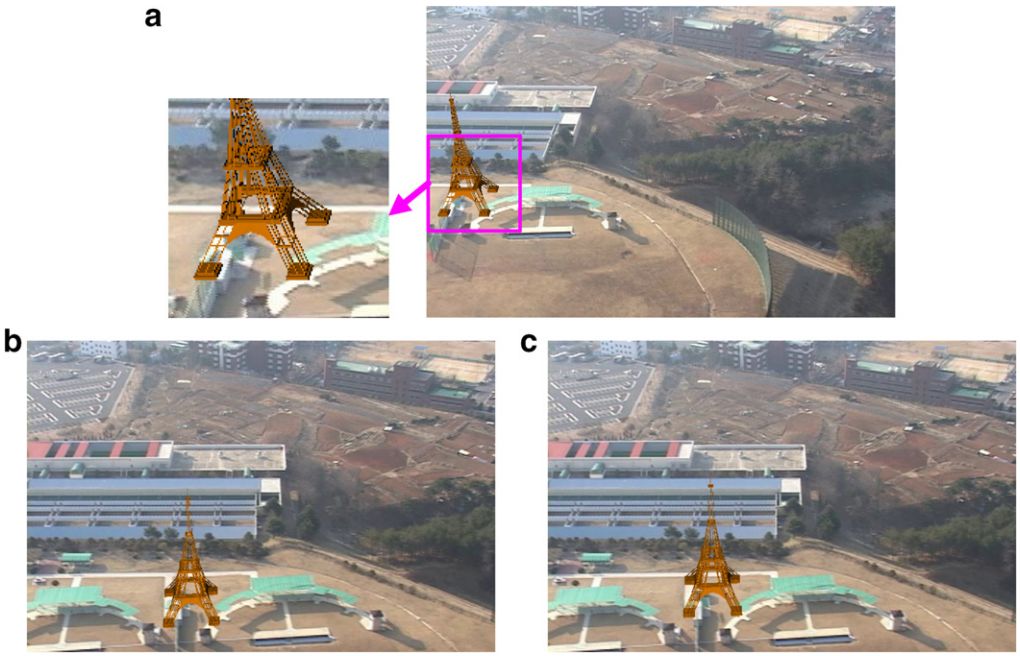

Fig. 8. Video augmentation results for Sequence 1. (a) Shows the location and pose of a virtual object at the first frame. (b) and (c) are the 90th augmented images of the video augmentation result obtained by the linear method and our method, respectively.

method was more accurate and stable than EKF even though it used only one sample.

5.3. Video augmentation results

To show the effectiveness of our camera resectioning method compared to the linear and nonlinear methods in the off-line video-based augmented reality, we applied the camera resectioning results of the preceding subsection to the video augmentation for the three test sequences. Due to space limitations, we showed only the video augmentation results for Sequence 1, obtained by the linear method and our method.

Video augmentation is firstly accomplished by embedding the graphic coordinate system into the world coordinate system of real video. In our augmented reality system, the embedding procedure is achieved by specifying the graphic coordinate system in two images of target video as in the work of Seo and Hong (2000). This procedure is guided by the *epipolar geometry* (Forsyth and Ponce, 2002) between the two images. The estimated camera motion is used to compute the epipolar geometry for implementing the embedding procedure.

Fig. 8 shows that a virtual object were correctly embedded into the world coordinate system of Sequence 1 in our method, unlike the linear method. Here, we used the first and 90th frames in the embedding procedure of our video mixing system in Fig. 1. A virtual object was inserted at the origin of each graphic coordinate system. The 90th augmented images of the video augmentation result obtained by the linear method and our method were shown in Fig. 8(b) and (c), respectively. The result of the nonlinear method were similar to ours. If we compare the locations of the virtual object in Figs. 8(a) and (b), we can observe that the linear method reveals a misalignment of the virtual object with the real scene. As we can see in Fig. 8(c), the video augmentation result of our method dose not have such misalignment.

6. Conclusions

We proposed a new recursive framework for camera resectioning and applied it to off-line video-based augmented reality. Our method is based on unscented particle filter and independent Metropolis–Hastings chain. The proposed method enabled us to propagate the mean and the covariance of the camera system state with high accuracy. We modelled the proposal distribution of our method with a mixture of a Gaussian distribution. This makes it possible to accurately estimate variables with a small number of samples in contrary to other particle filters. To efficiently generate samples from the proposal distribution, we combined an unscented Kalman filter and the independent Metropolis–Hastings chain into our sampling procedure. As a result, the proposed method overcomes the initialization problem in camera resectioning, and recursively estimates the motion of camera with high accuracy comparable to nonlinear optimization methods. We verified the effectiveness of our method by using several experiments using some synthetic and real image sequences and compared the estimation performance with other linear and nonlinear camera resectioning methods. We also demon-

strated the video augmentation based on our recursive camera resectioning algorithm.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.patrec.2006.11.012.

References

Anderson, B.D., Moore, J.B., 1979. Optimal Filtering. Prentice-Hall, New Jersey.

Azerbayejani, A., Pentland, A., 1995. Recursive estimation of motion, structure, and focal length. IEEE Trans. Pattern Recognit. Machine Intell. 17 (6), 562–575.

Chiuso, A., Favaro, P., Jin, H., Soatto, S., 2002. Structure from motion causally integrated over time. IEEE Trans. Pattern Recognit. Machine Intell. 24 (4), 523–535.

Davison, A., 2003. Real-time simultaneous localization and mapping with a single camera. In: Proc. Internat. Conf. on Computer Vision, pp. 1403–1410.

Doucet, A., de Freitas, J.F., Gordon, N.J., 2001. Sequential Monte Carlo Methods in Practice. Springer-Verlag.

Faugeras, O., 1993. Three Dimensional Computer Vision. MIT Press, pp. 52–58.

Fitzgibbon, A.W., Zisserman, A., 1998. Automatic camera recovery for closed and open image sequences. In: Proc. European Conf. on Computer Vision, pp. 311–326.

Forsyth, D.A., Ponce, J., 2002. Computer Vision: A Modern Approach. Prentice Hall.

Georgescu, B., Meer, P., 2002. Balanced recovery of 3D structure and camera motion from uncalibrated image sequences. Lect. Note Comput. Sci. 2351, 294–308.

Gibson, S., Cook, J., Howard, T., Hubbold, R., Oram, D., 2002. Accurate camera calibration for off-line, video-based augmented reality. In: Proc. Internat. Symp. on Augmented Reality.

Gilks, W.R., Richardson, S., Spiegelhalter, D.J., 1995. Introducing Markov chain Monte Carlo. Markov Chain Monte Carlo in Practice. Chapman & Hall, pp. 1–19.

Golub, G., Van Loan, C., 1983. Matrix Computations. Johns Hopkins Univ. Press, pp. 16–20.

Hartley, R., 1995. In defence of the 8-point algorithm. In: Proc. Internat. Conf. on Computer Vision, pp. 1064–1070.

Hartley, R., Zisserman, A., 2000. Multiple View Geometry in Computer Vision. Cambridge Univ. Press.

Isard, M., Blake, A., 1998. CONDENSATION—Conditional density propagation for visual tracking. Int. J. Comput. Vision 29 (1), 5–28.

Julier, J., Uhlman, J.K., 1997. A new extension of the Kalman filter to nonlinear system. In: Proc. AeroSense: The Internat. Symp. on Aerospace/Defence Sensing, Simulation and Controls.

Kim, J., Hong, K., 2004. Recursive camera resectioning with unscented particle filter in image sequences: Application to video-based augmented reality. In: Proc. Internat. Conf. on Artificial Reality and Telexistence, pp. 114–119.

Liu, J.S., Chen, R., 1998. Sequential Monte Carlo methods for dynamic systems. J. Amer. Statist. Assoc. 93, 1032–1044.

Merwe, R., Wan, E., 2003. Sigma-point particle filters for sequential probabilistic inference in dynamic state-space models. In: Proc. Internat. Conf. on Acoustics, Speech and Signal Processing.

Merwe, R., Freitas, N., Doucet, A., Wan, E., 2000. The unscented particle filter. In: Proc. Neural Information Processing Systems.

Nister, D., 2001. Frame decimation for structure and motion. In: Workshop on 3D Structure from Multiple Images of Large-Scale Environments, Lecture Note on Computer Vision, vol. 2018, pp. 17–34.

Pollefeys, M., Koch, R., Van Gool, L., 1999. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. Int. J. Comput. Vision 32 (1), 7–25.

Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R., 2002. Visual modelling with a hand-held camera. SIGGRAPH 2002 Course Note, vol. 44, p. 54.

Pupilli, M., Calway, A., 2005. Real-time camera tracking using a particle filter. In: Proc. British Machine Vision Conf., 2005.

Seo, Y., Hong, K., 2000. Calibration-free augmented reality in perspective. IEEE Trans. Visual. Comput. Graphics 6 (4), 346–359.

Seo, J., Kim, S., Jho, C., Hong, H., 2003. 3D estimation and keyframe selection for machmove. In: Proc. Internat. Technical Conf. on Circuits/Systems.

Shi, J., Tomasi, C., 1994. Good features to track. In: Proc. Computer Vision and Pattern Recognition, pp. 593–600.

Simon, G., Fitzgibbon, A., Zisserman, A., 2000. Markerless tracking using planar structures in the scene. In: Proc. Internat. Symp. on Augmented Reality.

Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A., 2000. Bundle adjustment – a modern synthesis. Vision Algorithms: Theory and Practice, Lecture Note on Computer Science, 298–375.

Wan, E.A., Merwe, R., 2000. The unscented Kalman filter for nonlinear estimation. In: Proc. Symp. on Adaptive System for Signal Processing, Communication and Control.

Welch, G., Bishop, G., 2001. An introduction to the Kalman Filter. SIGGRAPH 2001 Course Note, p. 8.