SIMULTANEOUS 3D OBJECT TRACKING AND CAMERA PARAMETER ESTIMATION BY BAYESIAN METHODS AND TRANSDIMENSIONAL MCMC SAMPLING

Raúl Mohedano and Narciso García

Grupo de Tratamiento de Imágenes, Universidad Politécnica de Madrid, 28040 Madrid, Spain
{rmp,narciso}@gti.ssr.upm.es; www.gti.ssr.upm.es

ABSTRACT

Multi-camera 3D tracking systems with overlapping cameras represent a powerful mean for scene analysis, as they potentially allow greater robustness than monocular systems and provide useful 3D information about object location and movement. However, their performance relies on accurately calibrated camera networks, which is not a realistic assumption in real surveillance environments.

Here, we introduce a multi-camera system for tracking the 3D position of a varying number of objects and simultaneously refining the calibration of the network of overlapping cameras. Therefore, we introduce a Bayesian framework that combines Particle Filtering for tracking with recursive Bayesian estimation methods by means of adapted transdimensional MCMC sampling. Additionally, the system has been designed to work on simple motion detection masks, making it suitable for camera networks with low transmission capabilities. Tests show that our approach allows a successful performance even when starting from clearly inaccurate camera calibrations, which would ruin conventional approaches.

Index Terms— 3D tracking, Bayesian estimation, camera network, transdimensional MCMC sampling.

1. INTRODUCTION

Most 3D tracking systems assume that an accurate calibration of the camera network is available. Unfortunately, this assumption does not match reality. The intrinsic calibration of the cameras (including lens distortion) can be accurately performed before placing the cameras (*e.g.* using calibration patterns), or can even be even provided by the manufacturer. However, the extrinsic parameters of each camera in the network depend directly on its specific geometry, and must thus be obtained once the cameras have been placed. Therefore, it is hard to achieve an accurate calibration for the camera network, specially in open scenarios such as streets or public buildings.

In addition, cameras could have been fixed using frail supports, rendering them susceptible to displacements (*e.g.* due to torsion or hits). A change in the camera position and, more importantly, orientation will result in a modification of the extrinsic parameters of the camera. It would cause a deviation from the calibration information initially provided, which could possibly ruin the 3D tracking system.

Here, we propose a multi-camera 3D tracking system which not only estimates people 3D position over time, but also the real extrinsic camera calibration of the network (assumed static). This can be performed by combining Bayesian tracking [1] and recursive Bayes estimation techniques [2] within a joint Bayesian framework.

This work has been partially supported by the Ministerio de Ciencia e Innovación of the Spanish Government under project TEC2010-20412 (Enhanced 3DTV). Also, R. Mohedano wishes to thank the Comunidad de Madrid for a personal research grant.

We assume an accurate intrinsic calibration for all cameras (a mild assumption), and an initial approximate estimation for the extrinsic calibration of the network. The 3D system will refine that approximate calibration over time, providing satisfying 3D tracking results even from poor initial extrinsic calibration information. These results could not be achieved without the calibration refining process.

To make it flexible and suitable for real camera networks (with moderate processing and transmission capabilities), the proposed 3D system works on binary motion masks reported independently by each camera. This simple input presents two advantages:

- All mono-camera processing is performed individually in each camera, as it does not represent a great computational cost. In addition, it can be done using unspecific, *off-the-shelf* binary motion detectors.
- Input data can be easily transmitted from all the cameras to the central node even in wireless network with low transmission capabilities, as binary images (and even more images with high spatial correlation such as binary motion masks) can be efficiently compressed.

Unlike in [3], the use of appearance information has been discarded, because it would be expensive to transmit and it has not been considered reliable enough for real tracking applications [4], as it changes dramatically from different points of view, and usual surveillance cameras tend to 'desaturate' colors in real illumination conditions. Instead, the presented system relies only on purely geometric reasonings, enhanced with the sequential refining of the calibration data.

2. BAYESIAN ESTIMATION FRAMEWORK

Let us denote by $\mathbf{x}_t$ the vector of parameters describing jointly the positions of the 3D objects present in the scene at time step t. Let us denote by Γ the joint extrinsic calibration information of all the C cameras of the system with respect to a certain real-world coordinate system. Note that, unlike $\mathbf{x}_t$, which represents varying phenomena that we aim to track over time, Γ does not depends on t, as it represents static parameters that we intend to iteratively refine. Let us finally denote by z_t the set of binary motion detection masks reported at time t by the C cameras of the system, $z_t = (z_t^c)_{c=1}^C$, and by Z^t the whole set of masks reported up to time t, $Z^t = (z_t, z_{t-1}, \ldots, z_1)$. Then, the conditional joint distribution of $\mathbf{x}_t$ and Γ given all the available observations Z^t can be written as

$$p\left(\mathbf{x}_t, \Gamma | Z^t\right) \propto p\left(z_t | \mathbf{x}_t, \Gamma\right)\, p\left(\mathbf{x}_t, \Gamma | Z^{t-1}\right), \qquad (1)$$

where we have assumed that the distribution of z_t is totally determined by $\mathbf{x}_t$ and Γ. The distributions $p\left(\mathbf{x}_t, \Gamma | Z^t\right)$ and $p\left(\mathbf{x}_t, \Gamma | Z^{t-1}\right)$ are not the usual *posterior* and *predicted distributions* of the Bayesian tracking framework [1], as unlike in common tracking systems they also include the time-invariant parameter Γ: this should be addressed specifically as in recursive Bayes estimation [2].

To handle this expression over time, we will use MCMC-based Particle Filtering, which has proved extremely effective for tracking multiple interacting objects. Traditional sampling methods for Particle Filtering (such as SIR [1]) draw samples from a predicted distribution which could differ drastically from the target distribution to approximate: whereas, MCMC-based methods (and his varying dimension version RJ-MCMC) sample directly from the posterior distribution. In our case, we should adapt the MCMC to our joint framework for Bayesian tracking and recursive Bayes estimation.

2.1. State-space definition

The state-space of the presented problem can be defined as the cartesian product of two spaces, one of them of variable dimensionality corresponding to object layout in space, $\mathbf{x}_t$, and the other of fixed dimensionality and corresponding to camera calibration, Γ.

We will define the object state vector as $\mathbf{x}_t = (\mathbf{x}_{i,t})_{i\in\mathcal{I}_t}$, where $\mathcal{I}_t$ is the set of indices identifying the $|\mathcal{I}_t|$ objects present in the scene and $\mathbf{x}_{i,t}$ the state of the object with identifier i. Additionally, each object will be modeled as a vertical cylinder with constant horizontal velocity: so each object will be encoded as

$$\mathbf{x}_{i,t} = (x_{i,t}, y_{i,t}, z_{i,t}, \dot{x}_{i,t}, \dot{y}_{i,t}, h_{i,t}, r_{i,t}) \ , \qquad (2)$$

where $x_{i,t}$, $y_{i,t}$ and $z_{i,t}$ are the cartesian coordinates of the central point of the cylinder base with respect to the real-world coordinate system at time t, $\dot{x}_{i,t}$ and $\dot{y}_{i,t}$ represent its horizontal velocity, and $h_{i,t}$ and $r_{i,t}$ are respectively its height and its radius. That choice for object representation is clearly oriented towards people tracking, but is not motivated by any limitation of the presented framework and could be adapted to other specific practical situations.

Whereas, the camera calibration state vector will be defined as $\Gamma = (\Gamma_c)_{c=1}^{C}$, where each Γ_c encodes both the rotation (3DoF) and the position (3DoF) of the c-th camera. Rotations will be encoded using Euler angles expressed in the ZYX convention, as this choice prevents gimbal lock problems in practical camera settings.

2.2. Continuous prediction equation using kernels

Our extended predicted distribution $p\big(\mathbf{x}_t, \Gamma | Z^{t-1}\big)$ can be written as

$$p\left(\mathbf{x}_t, \Gamma | Z^{t-1}\right) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})\, p\big(\mathbf{x}_{t-1}, \Gamma | Z^{t-1}\big)\, \mathrm{d}\mathbf{x}_{t-1}, \qquad (3)$$

where we have assumed that the *dynamic model* of the system, $p\left(\mathbf{x}_t|\mathbf{x}_{t-1}, \Gamma, Z^{t-1}\right)$, is simply $p\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right)$. Additionally, let us assume that, as in usual MCMC-based systems for tracking, we have approximated the joint tracking and estimation results at time t as a set of S equally-weighted samples $\left\{\mathbf{x}_{t-1}^{(s)}, \Gamma^{(s)}\right\}_{s=1}^{S}$, that is,

$$\tilde{p}\left(\mathbf{x}_{t-1}, \Gamma | Z^{t-1}\right) = \frac{1}{S}\sum_{s=1}^{S} \delta\Big((\mathbf{x}_{t-1}, \Gamma) - (\mathbf{x}_{t-1}^{(s)}, \Gamma^{(s)})\Big) \ , \qquad (4)$$

where $\delta(\cdot)$ represents the Dirac delta centered at the coordinate origin (see Fig. 1.a). Applying (4) into (3) we would obtain

$$\tilde{p}\left(\mathbf{x}_t, \Gamma | Z^{t-1}\right) \approx \frac{1}{S}\sum_{s=1}^{S} p\Big(\mathbf{x}_t|\mathbf{x}_{t-1}^{(s)}\Big)\, \delta\Big(\Gamma - \Gamma^{(s)}\Big) \ . \qquad (5)$$

Although the 'input' is a set of purely discrete samples, the result is a mixed probability distribution: continuous on $\mathbf{x}_t$ but discrete on Γ. This is so because the dynamic model $p\left(\mathbf{x}_t|\mathbf{x}_{t-1}\right)$ acts as a sort of interpolating kernel in the $\mathbf{x}_t$ dimensions [2], while there is no such an interpolating behavior in the Γ dimensions (see Fig. 1.b). However, to sample from the continuous density function $p\left(\mathbf{x}_t, \Gamma | Z^{t-1}\right)$ using MCMC we would need to construct an analytical continuous approximation for it. For that purpose, we will apply kernel density estimation techniques on the Γ dimensions.

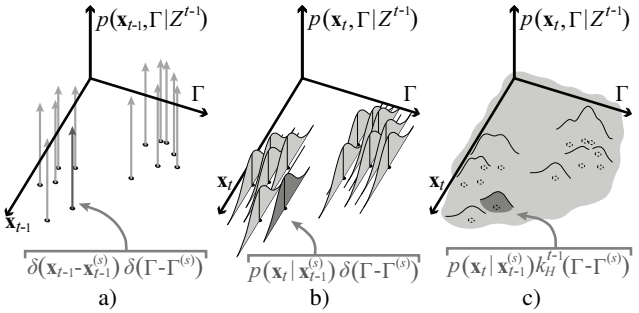

Fig. 1. Effect of the interpolating kernels $k_H^{t-1}(\Gamma)$ on the extended predicted distribution $p\left(\mathbf{x}_t, \Gamma | Z^{t-1}\right)$. a) Particle-approximated posterior at time $t-1$, and corresponding predicted *pdf* b) without kernels and c) with kernels.

Let us suppose now that we apply on (4) the kernel function $k_H^{t-1}(\Gamma)$, which, for simplicity, will be common for all the the S samples. Then, we could approximate the posterior distribution at time $t-1$ using the mixed distribution

$$\hat{p}\Big(\mathbf{x}_{t-1}, \Gamma \,|\, Z^{t-1}\Big) = \frac{1}{S}\sum_{s=1}^{S} \delta\Big(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^{(s)}\Big)\, k_H^{t-1}\Big(\Gamma - \Gamma^{(s)}\Big) \ . \qquad (6)$$

Then, applying this into the prediction equation (3), we obtain

$$\hat{p}\left(\mathbf{x}_t, \Gamma | Z^{t-1}\right) \approx \frac{1}{S}\sum_{s=1}^{S} p\Big(\mathbf{x}_t|\mathbf{x}_{t-1}^{(s)}\Big)\, k_H^{t-1}\Big(\Gamma - \Gamma^{(s)}\Big) \ , \qquad (7)$$

which is a purely continuous approximation for the predicted distribution (see Fig. 1.c). Along with the *observation model* $p\left(z_t|\mathbf{x}_t, \Gamma\right)$, this will allow us to use MCMC to approximate the posterior distribution: we will obtain again a set of S equally-weighted samples $\left\{\mathbf{x}_t^{(s)}, \Gamma^{(s)}\right\}_{s=1}^{S}$, on which we will apply again kernel density estimation to perform the operation of the algorithm at time $t+1$.

2.2.1. Automatic selection of kernel scale

As kernel function $k_H^{t-1}(\Gamma)$ for the camera-related dimensions, we will use a Gaussian kernel function with diagonal scale matrix H_{t-1} which will be estimated automatically from $p\left(\mathbf{x}_{t-1}, \Gamma | Z^{t-1}\right)$, or more specifically, from the samples $\{\mathbf{x}_{t-1}^{(s)}, \Gamma^{(s)}\}_{s=1}^{S}$.

Automatic scale selection is extremely important for recursive Bayes estimation. The use of a kernel yields a continuous function from discrete samples drawn from a certain *pdf* $p(x)$. However, the resulting function does not approximate $p(x)$, but its convolution with the (reflected) kernel function instead [2]. As we expect the estimation for Γ to be more and more precise as more evidence is gathered, that is, its posterior distribution will present lower and lower variance, H_{t-1} should be related to $p\left(\mathbf{x}_t, \Gamma | Z^{t-1}\right)$ to provide an acceptable continuous approximation for it.

For the automatic calculation of the diagonal scale matrix H, let us assume that the marginal posterior distribution of Γ is approximately Gaussian, and that the different camera components composing Γ are independent. Then we can estimate the variance increase caused by the kernel $k_H(\Gamma)$ independently for each component. Let us focus only on the r-th component, and let us assume that its true standard deviation is σ_r, whereas the scale of the kernel for the same component is h_r. Then, the deviation σ_r^* of that component after applying the kernel would be $\sigma_r^* = \sqrt{\sigma_r^2 + h_r^2}$. So, if we aim to limit the increase in the standard deviation to a certain low ratio ξ (such that $\sigma_r^* \leq \sigma_r(1+\xi)$), we should set

$$h_r \leq \sigma_r \sqrt{\xi(\xi+2)} \, . \qquad (8)$$

In our experiments we set ξ to 0.05, obtaining $h_r \approx 0.32\sigma_r$. However, we have observed that moderate deviations from that value do not have visible consequences on the performance of the system. The value σ_r will be estimated as the marginal standard deviation of the r-th component of the corresponding samples, computed using the basic unbiased estimator [2].

2.2.2. Dynamic model for multiple object interaction

The definition of an object state vector $\mathbf{x}_t$ encoding jointly all the 3D tracked targets allows us to use a dynamic model taking into account possible interactions between objects. The 3D dynamic model designed for our system draws on the one defined in [5] for 2D tracking, and can be factorized into simple terms so as to facilitate consecutive iterations of the MCMC algorithm.

First, although for clarity we have been denoting only by $\mathbf{x}_t$ the state vector for objects, it is convenient in practice to explicitly express as well the set $\mathcal{I}_t$ of object identifiers at time t. Obviously, $|\mathcal{I}_t|$ represents the number of currently objects tracked. Additionally, $\mathcal{I}_t \subset \mathcal{I}$, where $\mathcal{I} = \{1, \ldots, N\}$ is the set of possible identifiers considered, and N is the maximum number of objects. Including indices explicitly, the object dynamic model will be written as

$$p(\mathcal{I}_t, \mathbf{x}_t|\mathcal{I}_{t-1}, \mathbf{x}_{t-1}) = P(\mathcal{I}_t|\mathcal{I}_{t-1}, \mathbf{x}_{t-1})\, p(\mathbf{x}_t|\mathcal{I}_t, \mathcal{I}_{t-1}, \mathbf{x}_{t-1}) \ . \quad (9)$$

At this point, we state the following partition for $\mathcal{I}$, which will be very useful for subsequent definitions:

- $\mathcal{I}_S = \mathcal{I}_t \bigcap \mathcal{I}_{t-1}$: objects that *stay*
- $\mathcal{I}_D = \mathcal{I}_{t-1} \backslash \mathcal{I}_t$: objects that *disappear* at time t
- $\mathcal{I}_A = \mathcal{I}_t \backslash \mathcal{I}_{t-1}$: objects that *appear*
- $\mathcal{I}_R = \mathcal{I} \backslash (\mathcal{I}_S \cup \mathcal{I}_D \cup \mathcal{I}_A)$: rest of potential identifiers

As for the distribution of the identifiers, we assume that the presence of object $i \in \mathcal{I}$ in the scene does not depend on their position at time t and is also conditionally independent of the presence of other objects. Then, defining the probabilities P_S and P_A, we will write $P(\mathcal{I}_t|\mathcal{I}_{t-1}) = \prod_{i \in \mathcal{I}} P(i \in \mathcal{I}_t|\mathcal{I}_{t-1})$, where

$$P(i \in \mathcal{I}_t|\mathcal{I}_{t-1}) = \begin{cases} P_S, \text{ if } i \in \mathcal{I}_S\,; & (1-P_S), \text{ if } i \in \mathcal{I}_D \\ P_A, \text{ if } i \in \mathcal{I}_A\,; & (1-P_S), \text{ if } i \in \mathcal{I}_R \end{cases} \ . \quad (10)$$

Whereas, we will define the distribution of the positions of objects at time t (conditioned to the present indices $\mathcal{I}_t$) as

$$p\Big(\mathbf{x}_t|\mathcal{I}_t, \mathcal{I}_{t-1}, \mathbf{x}_{t-1}\Big) \propto \\ \propto \prod_{\substack{\forall i,j \in \mathcal{I}_t \\ i \neq j}} \phi(\mathbf{x}_{i,t}; \mathbf{x}_{j,t}) \prod_{\forall i \in \mathcal{I}_S} p(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}) \prod_{\forall j \in \mathcal{I}_A} p_A(\mathbf{x}_{j,t}) \ . \quad (11)$$

Object interaction is defined pairwise by means of a function involving only the current static positions of objects, and is defined as $\phi(\mathbf{x};\ \mathbf{y}) = d_{hn}^{\beta}/(1 + d_{hn}^{\beta})$, where d_{hn} is the horizontal distance between the axes of cylinders $\mathbf{x}$ and $\mathbf{y}$ normalized by the mean of their radii, and $\beta \geq 2$ to correctly discourage overlapping cylinders.

The other two last factors of (11) represent the *a priori* distribution for the position of objects present at time t, that is, $i \in \mathcal{I}_t$:

- Objects already present at t–1 ($i \in \mathcal{I}_S$) will depend on their previous state, and thus $p(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1})$. In that case, the components have been considered independent and Gaussian, all of them centered at their value at t–1 except for the horizontal coordinates $x_{i,t}$ and $y_{i,t-1}$ of the cylinder, which will be centered respectively at $(x_{i,t-1} + \dot{x}_{i,t-1})$ and $(y_{i,t-1} + \dot{y}_{i,t-1})$.
- Whereas, the distribution $p_A(\mathbf{x}_{j,t})$ for appearing objects ($j \in \mathcal{I}_A$) will be common for all objects. As we do not assume any previous information about entry areas in the scene, $p_A(\mathbf{x}_{j,t})$ will be defined as a uniform distribution for x, y and z. Velocity, radius and height distributions have been assumed Gaussian, centered respectively at $\vec{0}$, $r_0 = 0.35$m and $h_0 = 1.70$m.

2.3. Multi-camera motion detection observation model

Our multi-camera observation model draws on that in [6], adapted to multi-camera input and suppressing its training phase. Assuming that the observations performed by the C cameras, $z_t = (z_t^c)_{c=1}^C$, are conditionally independent given the true state of the system ($\mathbf{x}_t$ and Γ), we simply write $p(z_t|\mathbf{x}_t, \Gamma) = \prod_{c=1}^C p(z_t^c|\mathbf{x}_t, \Gamma_c)$, where Γ_c is the calibration of the c-th camera.

The factor corresponding to each camera will be a function of the precision ν_t^c and recall ρ_t^c of two binary masks, the observed one, z_t^c, and the expected one, $e_t^c = e_t^c(\mathbf{x}_t, \Gamma_c)$, which will be calculated as the projection of the objects/cylinders onto the image plane of the corresponding camera. The proposed distribution is $p(z_t^c|\mathbf{x}_t, \Gamma_c) = s(\nu_t^c, k_\nu) s(\rho_t^c, k_\rho)$, where $s(x, k_0)$ is a sigmoid-like function (specifically, a raised-cosine function) defined over the range $[0, 1]$ and with transition from 0 to 1 centered at the value k_0.

3. RJ-MCMC MOVES AND PRACTICAL OPERATION

Combining all the partial distributions previously discussed, the extended posterior distribution can be written as

$$p\big(\mathbf{x}_t, \Gamma|Z^t\big) \propto \prod_{c=1}^{C} p(z_t^c|\mathbf{x}_t, \Gamma_c) \prod_{\substack{\forall i,j \in \mathcal{I}_t \\ i \neq j}} \phi(\mathbf{x}_{i,t}; \mathbf{x}_{j,t}) \times \\ \times \sum_{s=1}^{S} \Bigg(\prod_{\forall i \in \mathcal{I}_S^{(s)}} p\Big(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}^{(s)}\Big) \prod_{\forall j \in \mathcal{I}_A^{(s)}} p_A(\mathbf{x}_{j,t}) \times \\ \times \prod_{i \in \mathcal{I}} P\Big(i \in \mathcal{I}_t|\mathcal{I}_{t-1}^{(s)}\Big)\ k_H^{t-1}\Big(\Gamma - \Gamma^{(s)}\Big) \Bigg) \quad (12)$$

where the interaction has been extracted outside the summation as it does not depend on the distribution at t–1. We will approximate this $p\big(\mathbf{x}_t, \Gamma|Z^t\big)$, defined over a space of variable dimensionality, by drawing discrete samples from it using RJ-MCMC techniques [5, 6]. More specifically, we will use the Metropolis-Hastings algorithm, which requires the definition of different reversible *moves* to iteratively draw candidate samples that will be accepted or not according to their acceptance ratio α [5]. An accurate representation of the posterior requires numerous iterations and consequently numerous evaluations of $p\big(\mathbf{x}_t, \Gamma|Z^t\big)$. To minimize the operations required by each possible move, we will store individually the projected mask of each of the (potentially $|\mathcal{I}|$) objects for each camera, and maintain intermediate calculations in specific matrix and vector structures.

Diffusion moves

Diffusion moves does not involve dimensionality changes. We define two moves: modification of one single camera, and modification of one single object (the rest of objects and cameras unchanged).

The modification of the c-th camera will be performed by drawing a new Γ_c^* from a Gaussian distribution centered at the Γ_c', the value of the previous iteration of Metropolis-Hastings. Reevaluating the posterior at the new hypothesis means reprojecting all objects on that specific camera only.

Analogously, the modification of the i-th object will be performed by drawing a new $\mathbf{x}_{i,t}^*$ from a Gaussian distribution centered at $\mathbf{x}_{i,t}'$. In that case, we will need to reproject that specific object in all the cameras to calculate the resulting posterior density.

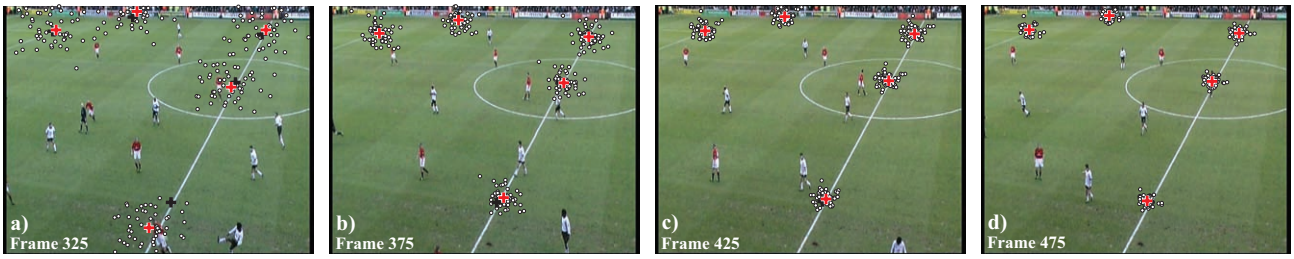

Fig. 2. Sequential refining of the calibration for a single camera, showed by projecting five 3D reference points using each calibration hypothesis. Black crosses: projection using the actual calibration data. White dots: projection using samples drawn with MCMC to approximate the distribution at time t. Red crosses: projection using the calibration estimation (mean of samples) at time t.

Transdimensional jumps

Jumps represent a dimensionality change of the state space. We define two jumps, *birth* and *death*, both related to appearance and disappearance of objects, and one the reverse of the other.

Death will be simply performed by randomly choosing an index $i \in \mathcal{I}'$ and removing the corresponding object. Whereas, birth will be performed by choosing randomly an identifier $i^* \in \mathcal{I} \backslash \mathcal{I}'_t$ and then following a data-driven proposal distribution [5] for $\mathbf{x}^*_{i^*,t}$. That distribution will be a sum of Gaussians centered at the 3D positions reported by an auxiliary 3D object detector, performed by processing pairwise those motion mask 2D blobs that cannot be justified by currently considered objects.

4. RESULTS

The proposed system has been tested on different scenarios monitored with several semi-overlapped cameras, modifying the extrinsic parameters of some of them. In the observation model, we have set the $k_\nu = k_\rho = 0.6$ to take into account that cylinders cannot perfectly approximate real objects. As for RJ-MCMC parameters, we draw 1000 total samples, out of which 25% will be considered part of the *burn-in* stage and thus discarded [5]. All the rest of particles will be used for kernel scale estimation, but only 50 of them, randomly chosen, are used to approximate the posterior distribution, reducing so the computational cost of each Metropolis-Hastings iteration.

The experiment displayed here corresponds to the SCEPTRE football database [7], where a match is monitored using 8 semi-overlapped cameras. Extrinsic parameters of cameras 2, 3 and 7 were deliberately deviated from their actual values: cartesian coordinates in 2m, and Euler angles in 2° (each). Fig. 2.a shows the uncertainty and the offset of the calibration estimation for camera 2 during the initial time steps of the algorithm by displaying the projection of several reference 3D points for the 50 selected particles $\Gamma_2^{(s)}$. Fig. 2b-d show how the estimated camera calibration approaches its actual value over time, reducing also the uncertainty on the estimation. Fig. 3 displays a virtual bird's-eye view of the trajectories (during the last 3 seconds for clarity) of 22 detected players, showing the tracking performance of the system in situations with several objects and close interactions between them.

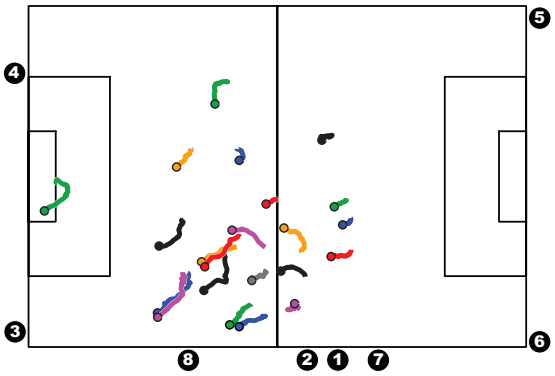

Fig. 3. Bird's eye view with tracked object trajectories (last 3 seconds). True camera positions have been schematically indicated.

5. CONCLUSIONS

We have presented a multi-camera system for jointly tracking a varying number of 3D objects and recursively refining the camera network calibration initially provided. So, it eases the need for accurate camera calibrations of most 3D tracking algorithms, which is often difficult to satisfy in real scenarios. This is achieved by combining Bayesian tracking and sequential Bayesian estimation through kernel-based techniques and transdimensional MCMC sampling.

Conducted tests show the capability of the presented approach to satisfactorily track the 3D positions of several interacting objects even when the initial calibration of certain cameras is not correct, yielding also an improved calibration estimation for them.

6. REFERENCES

[1] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[2] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, Wiley, New York, second edition, 2001.

[3] R. Mohedano and N. García, "Robust multi-camera 3D tracking from mono-camera 2D tracking using bayesian association," *IEEE Trans. Consumer Electronics*, vol. 56, no. 1, pp. 1–8, 2010.

[4] X. Zou, B. Bhanu, and A. Roy-Chowdhury, "Continuous learning of a multilayered network topology in a video camera network," *EURASIP Journal on Image and Video Processing*, vol. 2009, Article ID 460689, 2009.

[5] Z. Khan, T. Balch, and F. Dellaert, "Mcmc-based particle filtering for tracking a variable number of interacting targets," *IEEE Trans. PAMI*, vol. 27, no. 11, pp. 1805–1918, 2005.

[6] K. Smith, D. Gatica-Perez, and J. M. Odobez, "Using particles to track varying numbers of interacting people," in *Int. Conf. CVPR*, 2005, pp. 962–969.

[7] Kingston University, "Sceptre database (Service to Evaluate the Performance of Tracking and Recognition of Events)," 2008, `http://sceptre.king.ac.uk/sceptre/default.html`.