IEICE TRANS. INF. & SYST., VOL.E95–D, NO.2 FEBRUARY 2012

PAPER

Efficient Topological Calibration and Object Tracking with Distributed Pan-Tilt Cameras

Norimichi UKITA[†a)], *Senior Member*, Kunihito TERASHITA[†], *Nonmember, and* Masatsugu KIDODE[†], *Fellow*

SUMMARY We propose a method for calibrating the topology of distributed pan-tilt cameras (i.e. the structure of routes among and within FOVs) and its probabilistic model. To observe as many objects as possible for as long as possible, pan-tilt control is an important issue in automatic calibration as well as in tracking. In a calibration period, each camera should be controlled towards an object that goes through an unreliable route whose topology is not calibrated yet. This camera control allows us to efficiently establish the topology model. After the topology model is established, the camera should be directed towards the route with the biggest possibility of object observation. We propose a camera control framework based on the mixture of the reliability of the estimated routes and the probability of object observation. This framework is applicable both to camera calibration and object tracking by adjusting weight variables. Experiments demonstrate the efficiency of our camera control scheme for establishing the camera topology model and tracking objects as long as possible.

key words: *topology of fields of view, object tracking, pan-tilt cameras, efficient camera control*

1. Introduction

Object tracking is one of the fundamental problems in recent computer vision research. In particular, tracking among widely distributed cameras has become a popular research issue. If the fields of view (FOVs) of the cameras are overlapped and their extrinsic parameters are known, object tracking can be simplified by analyzing consistencies in 3D positions of the observed objects (e.g. using fixed cameras [1]–[4], omnidirectional cameras [5], and active cameras [6]–[9]). The extrinsic parameters of the distributed cameras can be obtained from the observation results of moving objects; for example, calibration of synchronized [10] and asynchronized [11] cameras, and improving the initial calibration results [12]. For all of these calibration methods, the cameras must be positioned so that each object moves through the FOVs of the cameras without going outside the FOVs. That is, the FOVs of the cameras must be overlapped. The assumption that the FOVs are overlapped makes it practically impossible to employ a number of cameras for observing wider areas.

Accordingly, camera configuration without overlapping FOVs, namely with blind spots, is necessary for wide-area surveillance. In this challenging problem, the topology of the FOVs and its probabilistic information (e.g. presence of a route, transit times, and object transit probabilities between FOVs) can improve object identification; see [13], [14], for example. Previously, the camera topology was given manually [13], [14]. Novel algorithms, however, provide us the camera topology automatically. Automatic calibration is desired because as the observation area grows and the number of cameras increases, the topology becomes drastically more complex. In [15], the camera topology is estimated from the results of object tracking among and within FOVs. However, object tracking between isolated FOVs only based on image cues (e.g. face/object recognition) is very difficult. For reliable identification among the FOVs, in [16], it is assumed that only one object moves in an environment. Otherwise, the trajectory of a moving landmark that is easily tracked (e.g. LED) [17] can be obtained even if other objects exist, or robust identification between isolated FOVs can be achieved also by employing information only of easy-identifiable objects as proposed in [18]. With these approaches, however, object transit probabilities between FOVs cannot be estimated because it can be acquired only from a number of real object trajectories. On the other hand, [19] and [20] acquire the probabilistic-topology of FOVs from a large amount of real object data, which is represented only by first and last detection results in each image; no tracking among isolated cameras is needed.

One of the next steps in calibrating widely distributed cameras is efficient utilization of pan-tilt cameras instead of fixed cameras. As proposed in [21], object tracking with pan-tilt cameras is crucial for efficient high-resolution observation. Similar to tracking with fixed cameras, the camera topology of the pan-tilt cameras is useful for object identification. This paper 1) shows that probabilistic topological calibration of pan-tilt cameras can be achieved by the same way as that of fixed cameras and 2) proposes a camera control scheme for efficient calibration. We also show that the camera control for efficient calibration can be used also for efficient object tracking with pan-tilt cameras.

2. Probabilistic Topological Camera Calibration

As defined in [20], our topology model of cameras is defined by the routes of moving objects. The route topology model is represented by a set of points of entrance and exit in FOVs. For simplicity, entrance and exit events are denoted by IN and OUT events, respectively. Two points observed at temporally consecutive events (i.e. IN-then-IN, IN-then-OUT, OUT-then-IN, and OUT-then-IN events) of

Manuscript received April 19, 2010.
Manuscript revised April 16, 2011.

[†]The authors are with the Graduate School of Information Science, Nara Institute of Science and Technology, Ikoma-shi, 630–0192 Japan.
a) E-mail: ukita@is.naist.jp

DOI: 10.1587/transinf.E95.D.626

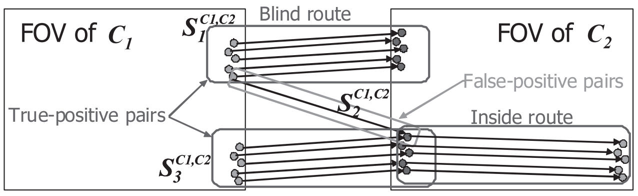


Fig. 1 True-positive route detection by vector quantization and thresholding. Small circles and lines between them indicate IN and OUT positions and their pairs, each of which connects the beginning and end points, respectively.

the same moving object compose a route. The earlier of the two points is called a *beginning point*. The other is called an *end point*. More specifically, (1) each route is defined only by its beginning and end points that are represented by 2D image coordinates, (2) routes are categorized into those within a FOV (e.g. "Inside route" in Fig. 1) and those in blind spots such as outside FOVs and behind obstacles (e.g. "Blind route" in Fig. 1), and (3) object trajectories between the beginning and end points (e.g. straight or curve trajectories) are not represented by the route information. The data observed at IN and OUT events (i.e. image coordinates in the observed image of each camera and the observation time) are called *IN data* and *OUT data*, respectively. IN data and OUT data are collectively called *IN/OUT* data; a single "IN/OUT data" means an IN data or an OUT data, and a set of "IN/OUT data" means a mixture of IN data and OUT data.

Unlike accurate 3D reconstruction, complete camera synchronization using synchronization signals is not have to be achieved in the proposed method; target positions (denoted by $\boldsymbol{P}(C)$ in the observed image of camera C) can be compared among the cameras even if the positions are captured at slightly different moments (e.g. in our experiments, the camera captured images 1 sec intervals). In distributed cameras, however, time-stamp synchronization is required for comparing the multiview images that have the closest time-stamps. In our experiments, the internal clocks of all computers connected to the cameras were synchronized by NTP [22].

The basic algorithm for probabilistic topological calibration [20] is as follows:

Step1 For finding pairs of IN/OUT data, each of which potentially consists of the beginning and end points of a route, IN/OUT data are paired with each other.

- A new IN data is paired with other IN and OUT data, and a new OUT data is also paired with other IN and OUT data.
- However, two IN/OUT data observed at quite different moments must not be temporally consecutive data of a moving object. Therefore, each new IN/OUT data is paired with other IN/OUT data, excepting those have been observed T_{max} sec[†] or more before the new data is detected.
- This pairing is executed among FOVs as well as within FOVs for finding both inside and blind routes.

Each pair is classified to a set, $\boldsymbol{S}^{B,E}$, where B and E denote the cameras in which the beginning and end points are observed, respectively. In the example shown in Fig. 1, each IN/OUT data is depicted by a circle. A line between two circles mean a pair of IN/OUT data. All the pairs between cameras C_1 and C_2 are classified to $\boldsymbol{S}^{C1,C2} = \boldsymbol{S}_1^{C1,C2} + \boldsymbol{S}_2^{C1,C2} + \boldsymbol{S}_3^{C1,C2}$,

Step2 Let $\{\boldsymbol{V}_1, \cdots, \boldsymbol{V}_{N^{B,E}}\}$ be a set of vectors, where r-th pair's $\boldsymbol{V}_v = (x_v^B, y_v^B, x_v^E, y_v^E, t_v)$ denotes a 5D vector comprising the image coordinates of the beginning and end points and the transit time between them. $N^{B,E}$ denotes the total number of the pairs in $\boldsymbol{S}^{B,E}$. Elements in each vector are normalized between 0 and 1. $x_v^B, y_v^B, x_v^E, y_v^E$ are normalized with the size of an image, and t_v is normalized with T_{max}.

Step3 $\{\boldsymbol{V}_1, \cdots, \boldsymbol{V}_{N^{B,E}}\}$ are divided into several subsets, $\boldsymbol{S}_i^{B,E}$ (e.g. $\boldsymbol{S}_1^{C1,C2}, \boldsymbol{S}_2^{C1,C2}, \boldsymbol{S}_3^{C1,C2}$ in Fig. 1), based on similarity; LBG algorithm [24] is used.

Step4 The number of the vectors in each subset is counted. Then its mean and standard variation of all subsets (denoted by μ^n and σ^n) are computed. If the number is less than $(\mu^n - 2.5\sigma^n)$[††], this subset is regarded as a set of false-positive pairs (e.g. $\boldsymbol{S}_2^{C1,C2}$ in Fig. 1), each of which consists of a temporally inconsecutive IN/OUT data of different moving objects, and then removed.

Step5 Each remaining subset corresponds to one route. In each route (denoted by r-th route), the mean and variance of $x_v^B, y_v^B, x_v^E, y_v^E, t_v$ and the number of the pairs of IN/OUT data, $N^{B,E}$, are computed. The mean (x, y) coordinates of the beginning and end points are denoted by $\boldsymbol{\mu}_r^B$ and $\boldsymbol{\mu}_r^E$, respectively, and their covariance matrices are denoted by $\boldsymbol{\Sigma}_r^B$ and $\boldsymbol{\Sigma}_r^E$, respectively. The mean and variance of the transit time are denoted by μ_r^t and $(\sigma_r^t)^2$, respectively.

Using these statistical data, the following two kinds of probabilistic values are computed in the following steps:

- $Pr_p(\boldsymbol{P}^B(C^B), \boldsymbol{P}^E(C^E))$: Probability that an object is detected in $\boldsymbol{P}^E(C^E)$ after it is last observed in $\boldsymbol{P}^B(C^B)$.
- $Pr_t(T_r)$: Probability that an object spends $T_r = T^E - T^B$ going through $\overrightarrow{\boldsymbol{P}^B(C^B) \cdot \boldsymbol{P}^E(C^E)}$, where T^B and T^E denote the time when the object was observed in $\boldsymbol{P}^B(C^B)$ and $\boldsymbol{P}^E(C^E)$, respectively.

Step6 Let $\boldsymbol{R}^{\cdot,E} = R_1^{\cdot,E}, \cdots, R_{N^{\cdot,E}}^{\cdot,E}$ be all routes with the end point in C^E, where $N^{\cdot,E}$ is the number of these routes. The probability that the end point of r-th route, $R_r^{\cdot,E}$, is

[†] T_{max} should determined so that the transit time of every existing route is less than T_{max}. In our experiments, T_{max} was determined by hand so that it was longish.

[††] While the threshold was determined by hand, it is not a sensitive one. Actually the finally obtained routes did not change in our experiments even if the threshold was $(\mu^n - 2.0\sigma^n)$ and $(\mu^n - 3.0\sigma^n)$.

$\boldsymbol{P}^E(C^E)$ is calculated by substituting $\boldsymbol{\mu}_r^E$ and $\boldsymbol{\Sigma}_r^E$ of $R_r^{\cdot,E}$ and $\boldsymbol{P}^E(C^E)$ for the equation of the Gaussian below:

$$Pr_E(\boldsymbol{P};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{2\pi|\boldsymbol{\Sigma}|^{\frac{1}{2}}}\exp\left((\boldsymbol{P}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{P}-\boldsymbol{\mu})\right) \quad (1)$$

Let S be the total sum of $Pr_E(\boldsymbol{P};\boldsymbol{\mu},\boldsymbol{\Sigma})$ multiplied by the number of pairs, namely,

$$S = \sum_{r=1}^{N^{\cdot,E}} Pr_E(\boldsymbol{P}^E(C^E);\boldsymbol{\mu}_r^E,\boldsymbol{\Sigma}_r^E)Nv_r,$$

where Nv_r denotes the number of pairs classified into r-th route. Then, $\left(Pr_E(\boldsymbol{P}^E(C^E);\boldsymbol{\mu}_r^E,\boldsymbol{\Sigma}_r^E)Nv_r\right)/S$ can be considered to be the probability that the end point of route $R_r^{\cdot,E}$ is regarded as the position of new detection at $\boldsymbol{P}^E(C^E)$ (denoted by $Pr_N(\boldsymbol{P}^E(C^E),R_r^{\cdot,E})$).

Step7 Let $\boldsymbol{R}^{B,E} = R_1^{B,E},\cdots,R_{N^{B,E}}^{B,E}$ be a subset of $\boldsymbol{R}^{\cdot,E}$, which has the beginning point at the FOV of C^B. The probability that the beginning point of $R_j^{B,E}$ is $\boldsymbol{P}^B(C^B)$ (this probability is denoted by $Pr_E(\boldsymbol{P}^B(C^B);\boldsymbol{\mu}_j^B,\boldsymbol{\Sigma}_j^B)$) is calculated by Eq. (1).

Step8 The total sum of $Pr_E(\boldsymbol{P}^B(C^B);\boldsymbol{\mu}_j^B,\boldsymbol{\Sigma}_j^B)$ multiplied by $Pr_N(\boldsymbol{P}^E(C^E),R_r^{\cdot,E})$ of the same route is the probability, Pr_p, that an object was lastly observed at $\boldsymbol{P}^B(C^B)$ before it is newly detected at $\boldsymbol{P}^E(C^E)$:

$$\begin{aligned} &Pr_p(\boldsymbol{P}^B(C^B),\boldsymbol{P}^E(C^E)) \\ &= \sum_{x=1}^{N^{B,E}} Pr_N(\boldsymbol{P}^E(C^E),R_x^{\cdot,E})Pr_E(\boldsymbol{P}^B(C^B);\boldsymbol{\mu}_j^B,\boldsymbol{\Sigma}_j^B) \end{aligned}$$

Step9 Pr_t, which denotes the probability that an object spends T_r for crossing route r, is also calculated by the Gaussian equation:

$$Pr_t(T_r) = \frac{1}{\sqrt{2\pi}\sigma_r^t}\exp\left(-\frac{(T_r-\mu_r^t)^2}{2(\sigma_r^t)^2}\right), \quad (2)$$

where μ_r^t and $(\sigma_r^t)^2$ denote the mean and variance of the transit time of r-th route, respectively, as described before.

Step10 $Pr_p(\boldsymbol{P}^B(C^B),\boldsymbol{P}^E(C^E))Pr_t(T_r)$ is considered to be the probability that the object is detected in $\boldsymbol{P}^E(C^E)$ after it leaves for route r from $\boldsymbol{P}^E(C^E)$ and the transit time is T_r.

3. Panoramic Image for Efficient FOV Representation

The topology of fixed cameras is determined for a set of FOVs in [19], [20]. With pan-tilt cameras, it is possible to prepare the camera topology for FOVs corresponding to several pan tilt angles in each camera as shown in Fig. 2 (a), in which each pan-tilt angle is regarded as a virtual FOV. In this virtual FOV configuration, the number of the FOVs is increased in contrast to the number of the real cameras. Increasing the FOVs results in increasing false-positive pairs of IN/OUT data, as proved in [20]. Increasing the false-positive pairs makes removing them (i.e. Step4 in Sect. 2) more difficult. Because the method described in Sect. 2 removes them by assuming that they are fewer than true-positive pairs corresponding to real routes.

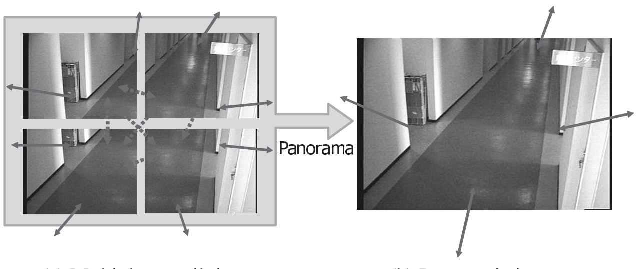


Fig. 2 Camera topology in pan-tilt cameras. Arrows indicate routes. Dotted arrows indicate routes between the pan-tilt images of a camera.

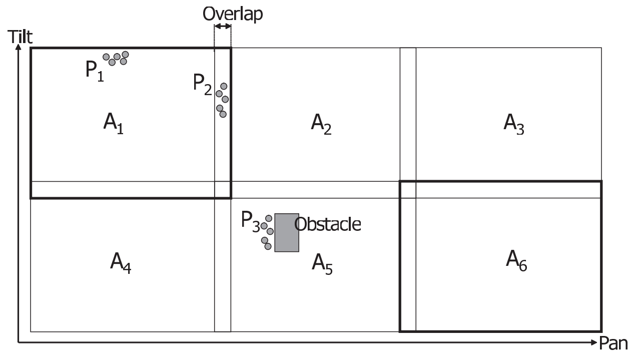


Fig. 3 Predefined pan-tilt angles for a panoramic image. $A_1,\cdots,A_6$ indicate six partial images.

In our method, therefore, images captured from all pan-tilt angles of a camera are stitched into one *panoramic image*. The image captured from each pan-tilt angle is called a *partial image*. With the panorama generation algorithm proposed in [23], a seamless panoramic image is obtained from images observed in any pan-tilt angles. With this panoramic image, only one image representation in each camera is needed for estimating the topology of pan-tilt cameras in the same way as [19], [20] as illustrated in Fig. 2 (b).

Our method controls each pan-tilt camera towards several predefined angles for synthesizing the panoramic image. That is, the camera is controlled with a stop-and-go manner (not smooth panning and tilting). In the example shown in Fig. 3, six partial images ($A_1,\cdots,A_6$) are used. For monitoring all scenes observable from a pan-tilt camera, its panoramic image must be synthesized from partial images with no gap among them. The neighboring partial images must overlap because the gap between the partial images cannot be observed at any time. This is not desired for monitoring a wider scene. The stop-and-go camera control only with the predefined angles gives us two advantages.

The first advantage is robust tracking inside the FOV of the panoramic image. Compared with object tracking while smoothly moving the pan-tilt angle, tracking in a fixed angle (i.e. tracking with a atop-and-go manner) is easier. As described later, this tracking is required both for topological calibration and tracking with the calibration results.

The second advantage is easy removal of false-positive IN/OUT data in the calibration process. The false-positive IN/OUT data is a kind of ghost IN/OUT data, which should not be detected in the panoramic image. In each partial image, IN/OUT data (e.g. $\boldsymbol{P}_2$ in Fig. 3) are detected in its border region when objects enter and exit the partial image. These IN/OUT data are not obtained if a fixed wider-FOV camera observes the whole scene (e.g. $A_1 + A_2 + A_3 + A_4 + A_5 + A_6$ in Fig. 3) with no pan-tilt rotation. In the panoramic image representation, therefore, these IN/OUT data are false-positive. Most true-positive IN/OUT data, which should be detected by the fixed wider-FOV camera, are detected in the border region of the panoramic image (e.g. $\boldsymbol{P}_1$ in Fig. 3). But true-positive IN/OUT data might be detected also inside the panoramic image. For example, true-positive IN/OUT data are detected if there is an obstacle in the FOV of this camera (e.g. $\boldsymbol{P}_3$ in Fig. 3).

To remove only the false-positive IN/OUT data, consistency between neighboring partial images are evaluated. Figure 4 shows how to discriminate between true-positive and false-positive IN/OUT data. See Fig. 4 (a). If $\boldsymbol{P}_1$ is detected in the border region of A_1 when an object enters or exits A_1, $\boldsymbol{P}_1$ is projected onto the panoramic image. In the same region of the panoramic image, no data is projected from A_2. This is because neither of the border and obstacles exist in the corresponding region of A_2. In this case, $\boldsymbol{P}_1$ in A_1 is regarded as a false-positive IN/OUT data. For this verification of false-positive data, exact correspondence between temporally successive IN/OUT data in the neighboring partial images (e.g. $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$) is not required. Similarly, $\boldsymbol{P}_2$ detected in the border of A_2 is also regarded as a false-positive IN/OUT data.

On the other hand, if IN/OUT data detected in the neighboring partial images are projected onto the same region of the panoramic image as depicted by $\boldsymbol{P}_3$ and $\boldsymbol{P}_4$ in Fig. 4 (a), the IN/OUT data are regarded as true-positive.

In order to gain the two advantages, the width of the overlap between the neighboring partial images is crucial:

- If the overlap is smaller, the camera should change its pan-tilt angle soon when its target gets into the overlap. In undesired cases (e.g. when the target turns over and gets back), the camera might lose the target.
- If the width of the overlap between the neighbors is small, false-positive data detected in the neighbors are mixed as shown in the right of Fig. 4 (b). This results in difficulty in removing the false-positive data.

To robustly track the target and remove the false-positive IN/OUT data, the overlap between the neighboring partial images should be larger. In the larger overlap, the above two advantages can be obtained as follows:

- The pan-tilt angle is controlled when the target is near the center of the overlap. In the next angle, the target can be observed in the overlap despite the unknown movement of the target (e.g. even if the target suddenly turns over), if the pan-tilt speed is fast[†].
- Since the overlap is large, IN/OUT data detected in the borders of different partial images (i.e. $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ in Fig. 4 (a)) are away from each other. The long distance between $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$ allows us to discriminate between them easily.

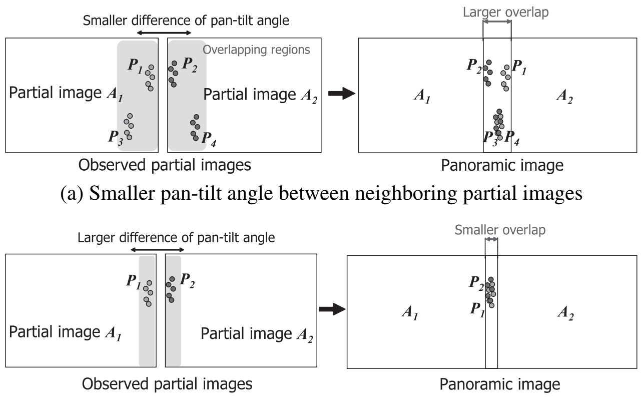
In our experiments, the width of the overlap was determined to be the maximum horizontal/vertical velocity of objects observed around the image border (i.e. the distance that the object travels between the camera capturing interval, denoted by dis, 1 sec in our experiments). With this condition, (1) a moving object stays in the overlapping region while the camera changes its angle and (2) IN/OUT data detected in the borders of the neighboring partial images are around dis away from each other.

4. Camera Control Strategy for Efficient Trajectory Acquisition

Even if an object is within the panoramic image of a pan-tilt camera, it fails to observe the object that is outside the current partial image. To avoid this problem, efficient camera control that allows us to obtain IN/OUT data as much as possible is required. In our method, this camera control is achieved based on the following three functions:

- Function-1: Search of new objects
- Function-2: Tracking in a panoramic image
- Function-3: Tracking through blind routes

In what follows, these three functions are introduced (Sect. 4.1, 4.2, and 4.3) and then a control scheme of



(a) Smaller pan-tilt angle between neighboring partial images

(b) Larger pan-tilt angle between neighboring partial images

Fig. 4 Overlap between IN/OUT data detected in neighboring partial images. A_1 and A_2 are neighboring images. "P_1 and P_2" and "P_3 and P_4" are false-positive and true-positive IN/OUT data detected in these images, respectively.

[†]All pan-tilt cameras that have been used by the authors are fast enough for tracking between neighboring partial images; any pan-tilt camera takes less than 0.5 sec, which is less than a capturing interval (i.e. 1 sec), for changing the angle to the neighboring partial image.

the pan-tilt cameras with these functions is described in Sect. 4.4.

4.1 Function-1: Search of New Objects

If no object and OUT data is currently detected by camera C_c and cameras each of which has a route(s) with C_c, the pan-tilt angle of C_c is directed towards where a new object is likely to appear. The possibility of object detection in each partial image, A_i, is computed from the number of IN/OUT data detected in A_i. Then A_i is observed at the following interval so that the interval is proportional to the number of IN/OUT data plus a minimum duration, D_0:

$$d_i = (D - ND_0)\frac{N_i^O}{\sum_{i=1}^{N} N_i^O} + D_0, \tag{3}$$

where D denotes the total duration for observing all partial images through one cycle, and N_i^O is the number of IN/OUT data obtained in A_i, except that $d_i = D_0$ at an initial state (i.e. when no IN/OUT data is obtained).

4.2 Function-2: Tracking in a Panoramic Image

If any object is currently observed by a camera, this camera can track it by controlling the pan-tilt angles in order to detect its OUT data. Notice that only predefined pan-tilt angles shown in Fig. 3 are acceptable also for this tracking. A large number of successful methods have been proposed for object detection and tracking within a FOV (e.g. robust detection under non-stationary scenes [25] and occlusion-robust tracking [26]), unlike tracking among isolated FOVs.

4.3 Function-3: Tracking through Blind Routes

Not only for tracking with the camera topology but also for efficiently obtaining IN/OUT data for topological calibration, object transit probability estimated from the probabilistic camera topology is useful. The transit probability of object o that travels through route r is estimated as defined in [20] (i.e. Step 10 in Sect. 2). At each moment, the point in which function $Pr_p(\boldsymbol{P}^B(C^B), \boldsymbol{P}^E(C^E))Pr_t(T_r) = Pr(o,r)$ has the maximum value is regarded as the end point of route r in which object o is most likely to appear. In order to use this function for predicting the transit probability, $\boldsymbol{P}^B(C^B)$, $\boldsymbol{P}^E(C^E)$, and T_r are regarded as the position where o is lastly detected, the mean position of the end point of route r from C^B, and the transit time after o is lastly detected in $\boldsymbol{P}^B(C^B)$. If C^B connects to multiple routes, $Pr_p(\boldsymbol{P}^B(C^B), \boldsymbol{P}^E(C^E))Pr_t(T_r)$ is computed for each route.

4.4 Camera Control by Object Observability and Route Uncertainty

Our objective in the calibration period is to efficiently obtain IN/OUT data that is useful for improving the reliability of the calibration. In the calibration period, the useful IN/OUT data are detected in the beginning/end points of *uncertain* routes. The *uncertainty* of the route is determined by the number of IN/OUT data detected in the route; the uncertainty is decreased as the number of the IN/OUT data increases. After the uncertainty is decreased, namely in the tracking period, the cameras should observe as many objects as possible for as long as possible.

In both of the calibration and tracking periods, if no object and recent OUT data is not detected in camera C_c and cameras each of which has a route(s) with C_c, C_c is controlled based on function-1.

On the other hand, when one or more objects are moving through the routes of C_c, one of the objects is selected as a target. C_c is then controlled so that the selected one is observed. Our method selects the target based on the uncertainty of each route and the observation probability of IN/OUT data. This is because more IN/OUT data should be detected in uncertain routes and the camera should be directed towards the beginning/end point of the route in which any object is most likely to appear.

The normalized score of the uncertainty of each route is expressed as follows:

$$U(r) = \frac{u_r}{max}, \tag{4}$$

$$u_r = \frac{(\sigma_r^{x^B})^2 + (\sigma_r^{y^B})^2 + (\sigma_r^{x^E})^2 + (\sigma_r^{y^E})^2 + (\sigma_r^{t})^2}{Nv_r}, \tag{5}$$

where max denotes the maximum u_r during the calibration period. $(\sigma_r^{x^B})^2, (\sigma_r^{y^B})^2, (\sigma_r^{x^E})^2, (\sigma_r^{y^E})^2, (\sigma_r^{t})^2$ are variance values of $x_r^B, y_r^B, x_r^E, y_r^E, t_r$[†], which are the components of a 5D vector $\boldsymbol{V}_r$, in route r; see steps 2 and 5 in Sect. 2.

The observation probabilities are computed for all possible routes of every detected object. The probability of a previously detected object, denoted by o, moving through a route, denoted by r, is computed using collected IN/OUT data based on function-3: $Pr(o,r) = Pr_p(\boldsymbol{P}^B(C^B), \boldsymbol{P}^E(C^E))Pr_t(T_r)$. On the other hand, the observation probability of each object currently being detected is set to be a constant value.

The pan-tilt camera C_c is controlled at each moment towards the route having the maximum value of the following weighted sum of $Pr(o,r)$ and $U(r)$:

$$S(o,r) = w_p Pr(o,r) + w_u U(r). \tag{6}$$

w_p and w_u are adjusted depending on the uncertainty of the obtained probabilistic topology. During the calibration period, w_p and w_u should be low and high, respectively. In the tracking period, on the other hand, w_p and w_u should be high and low, respectively.

The method switches from the calibration period to the tracking period in accordance with the uncertainty of the routes, $U(r)$. In our method, the method switches to the tracking period when $U(r)$ of every route becomes above

[†]Namely, "$(\sigma_r^{x^B})^2, (\sigma_r^{y^B})^2$" and "$(\sigma_r^{x^E})^2, (\sigma_r^{y^E})^2$" are "1st-row-1st-column and 2nd-row-2nd-column" elements of $\boldsymbol{\Sigma}_r^B$ and $\boldsymbol{\Sigma}_r^E$, respectively.

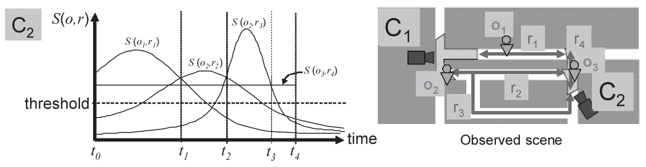


Fig. 5 Temporal history of camera control. Left: temporal histories of $S(o, r)$ in camera C_2, Right: The locations of cameras and objects in an observed scene.

a threshold. The threshold should be determined depending on the task. For example, the threshold should be lower if tracking through popular routes, whose uncertainty becomes soon lower, is important. In all of our experiments, the threshold was $U(r) = 0.05$. The threshold was determined by preliminary experiments as follows. In each camera configuration, 10 sets of IN-OUT data were obtained, and each set was analyzed by our method separately. Then in all the sets, $U(r) = 0.05$ could get routes whose number was $\pm 10\%$ of the mean of those of all the sets.

Note that $S(o, r)$ is changed with time because $Pr(o, r)$ is determined depending on the difference between the current time and the time when object o is last detected. Let $S(o_{max}, r_{max})$ be the maximum at time t. If $S(o_{max}, r_{max})$ is lower than a threshold, the camera is controlled for search based on function-1. If object o_{max} is not observed currently, the camera is controlled towards the end point of route r based on function-3. If object o_{max} is observed currently (i.e. o_{max} is moving an inside route), the camera tracks it based on function-2.

Figure 5 shows an example. Objects o_1 and o_2 have left the FOV of C_1 while object o_3, which has left the FOV of C_2 at time t_4, is currently moving an inside route, r_4, in the FOV of C_2 at time t_0. The lefthand graph indicates the temporal histories of $S(o, r)$ in possible object trajectories. As shown in this graph, C_2 is controlled towards r_1, r_2, r_3, and o_3 during time periods t_0-t_1, t_1-t_2, t_2-t_3, and t_3-t_4. Later the camera is controlled for search because no $S(o, r)$ has a value that is above a threshold. The same camera control manner is applied in every camera at every time.

5. Experiments

We conducted comparative experiments to demonstrate efficiency of our proposed method. Both for calibration and tracking, two methods were evaluated: our proposed method and a method using simple search described in 4.1 and intra-FOV visual tracking described in 4.2. To evaluate the results of different methods in the same situation in each trial, 3D simulation environments were used.

First of all, two camera configurations in a small block environment shown in Fig. 6 and 7 were calibrated. Configuration (b) was more complex than configuration (a). For example, 1) P_1 in C_1 connected to multiple positions, P_2 and P_3, in C_2, 2) P_4 in C_3 connected to multiple cameras, P_5 in C_2 and P_6 in C_4, and 3) C_3 had a blind route due to an obstacle. In the simulation environments, each object was represented by a 3D point that moved on a floor. When the point was first and last observed by a camera, the observed point was regarded as IN and OUT data, respectively. Objects entered the block environment from one of $E_1, \cdots, E_5$ randomly, selected its direction randomly at each corner, and left from one of $E_1, \cdots, E_5$. Object trajectories and velocities were fluctuated with Gaussian noise. The mean velocity was determined so that an object moved from E_2 to E_5 in 12 seconds. At each moment, two objects were simultaneously moving in the environment. With the above manners, three trials, each of whose IN/OUT dataset was analyzed by two methods, were performed.

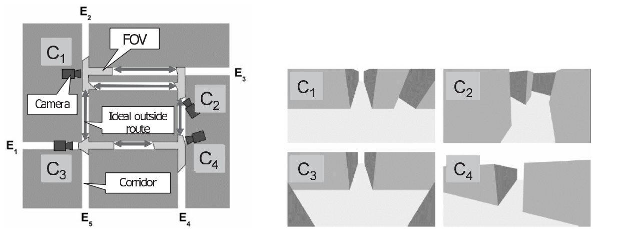


Fig. 6 Camera configuration (a). Left: bird view of the 3D simulation environment and routes between the cameras (inside routes are not illustrated for simplification). Right: panoramic images in cameras C_1, C_2, C_3, and C_4.

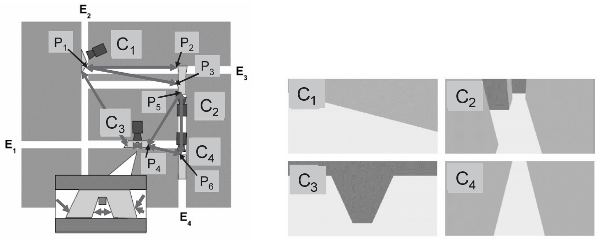


Fig. 7 Camera configuration (b). See the caption of Fig. 6 for details.

The resolution of each camera was assumed to be 640×480 pixel. The ranges of its pan-tilt angles were $\pm 30°$ and $\pm 15°$, respectively. The predefined pan-tilt angles were 12 directions (4 pan angles $\times$ 3 tilt angles). The cameras captured images at 1 sec intervals. The minimum duration for object search (i.e. D_0 in Formula (3)) was five seconds. The weight variables in Formula (6) were determined as follows: $w_p = 0.2$ and $w_u = 0.8$ for the calibration period and $w_p = 1.0$ and $w_u = 0.0$ for the tracking period.

With the above experimental environments, the topology calibration was evaluated in terms of 1) the effectiveness of removing false-positive IN/OUT data described in Sect. 3 and 2) efficiency of our proposed camera control.

1) removing false-positive IN/OUT data: Figure 8 shows one of the estimated routes in camera configuration (a) by our method without/with false-positive removal. In the figure, small dots and arrows between them indicate IN/OUT data and estimated routes, respectively. The arrows are grouped into true-positive (indicated by solid arrows) and false-positive (indicated by dotted arrows) routes. Discrimination between true-positive and false-positive was

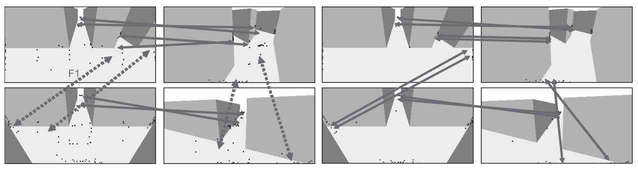


Fig. 8 Topology calibration results without/with false-positive IN/OUT data removal in camera configuration (a). Each image is a panoramic image captured from a pan-tilt camera. All arrows indicate estimated routes. Solid and dotted arrows indicate true-positive routes and false-positive routes, respectively.

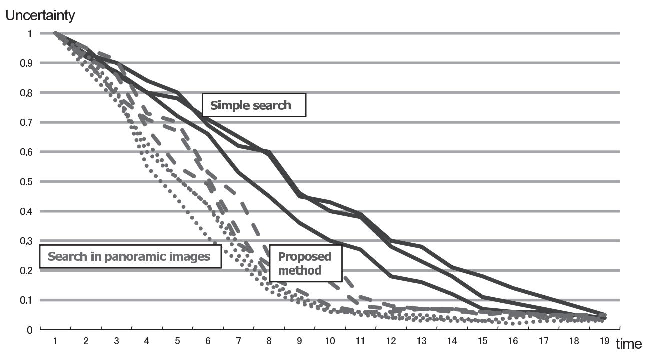


Fig. 9 Temporal histories of the uncertainty scores in camera configuration (a). Solid lines: simple search for 12 partial images, Dotted lines: panoramic images, Broken lines: proposed method for 12 partial images.

done manually. Four false-positive routes were obtained by the method without removing false-positive IN/OUT data. For example, route F1 was the false-positive because its right-side end point was far from the border of the panoramic image (i.e. actual end point of F1). The end point of F1 was shifted towards the inside of the panoramic image due to false-positive IN/OUT data detected in the border of a partial image. By comparing two results in Fig. 8, it can be seen that false-positive routes were not detected by our method.

2) camera control: Figure 9 shows the temporal histories of the mean value of normalized uncertainty scores ($U(r)$ in Formula (4)) in each trial in camera configuration (a). Blue and red lines indicate the scores obtained by camera control without the camera topology, which is equal to the initial search scheme of our method described in Sect. 4.1, and our proposed method. The horizontal axis indicates a transit time in the simulation environment (not a computational time). Our method could obtain a sufficient amount of IN/OUT data, which got less than $U(r) = 0.05$ in all routes, 1.4 times as fast as the simple search scheme. For comparison, the scores obtained by IN/OUT data acquisition in one panoramic image at each camera are also shown by green lines in the figure. Since all IN/OUT data can be obtained in the panoramic image, the uncertainty score decreased most rapidly. It can be seen that the proposed method got closer to data acquisition in the panoramic image.

For evaluating the effect of the number of partial im-

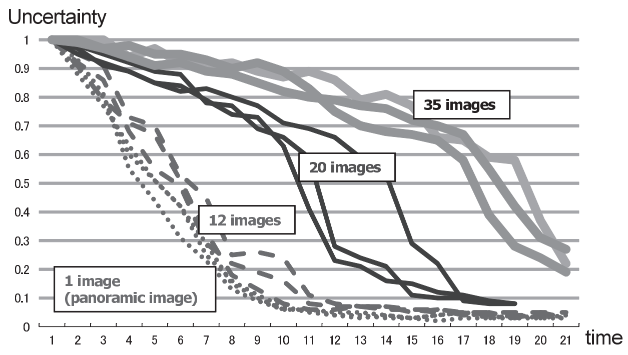


Fig. 10 Temporal histories of the uncertainty scores of our method in camera configuration (a). Thick line: 35 partial images, Solid line: 20 partial images, Broken line: 12 partial images, Dotted line: panoramic images.

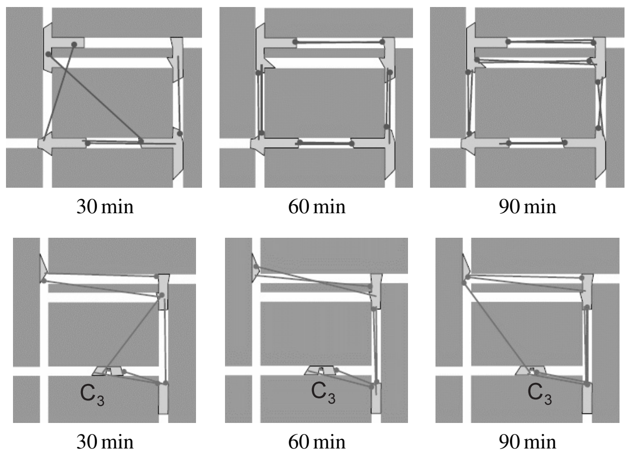


Fig. 11 Temporal histories of estimated routes. Arrows indicate the estimated routes. Upper: configuration (a), Lower: configuration (b).

ages on efficient camera control, experiments with different numbers of partial images (1, 12, 20, and 35 images) were also done in camera configuration (a). While the size of the panoramic image was not changed in these experiments, each partial image shrank as the number of the partial images grew. Figure 10 shows the temporal histories of the mean value of normalized uncertainty scores. Roughly speaking, it can be seen that the score decreased drastically between around 0.6 and 0.5 even if the number of the partial images was larger. That might be happened because collected IN/OUT data was then able to sometimes estimate correct probabilistic information about routes.

Figure 11 shows the histories of routes estimated by our method in camera configurations (a) and (b) in one of the trials. It can be seen that the estimated routes were refined over time. In (b), however, several false-positive and false-negative routes remained around the obstacle in the FOV of C_3. The magnified images around the obstacle were shown in Fig. 12. These errors were caused because tracking within a panoramic image was disturbed by the obstacle. Assume that an object is moving from left to right in the panoramic image of C_3. Its OUT data might be detected at the left boundary of the obstacle. After this OUT data detection, the

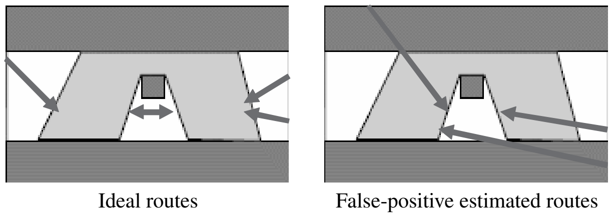


Fig. 12 False-positive routes in C_3 of camera configuration (b).

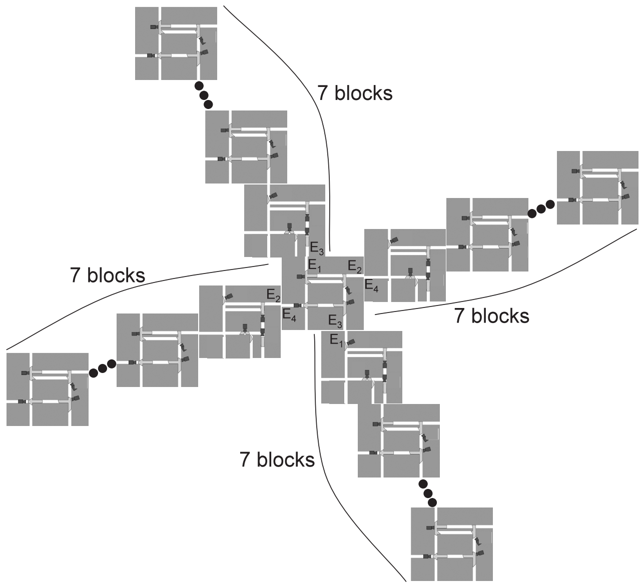


Fig. 13 Bird view of the huge simulation environment.

pan-tilt angle of C_3 was controlled based on function-1 or function-3. Unfortunately, based on function-1 (i.e. search) in a raster scan manner, the pan-tilt angle was directed towards the top left of the panoramic image. Therefore, the route between the obstacle could not be detected in our experiments. To solve this problem, camera control in a search scheme should be determined not only in a routine manner but also in a random manner as similar to Monte Carlo simulated annealing.

Next, a number of cameras in a huge environment were calibrated. The huge environment was prepared by connecting the small block environments with camera configuration (a). The outline of the huge environment is illustrated in Fig. 13. Specifically, this huge environment was generated by connecting E_1, E_2, E_3, and E_4 of camera configuration (a) to E_3, E_4, E_1, and E_2 of camera configuration (b), respectively, and vice versa, recursively. A block with camera configuration (a) was located in the center and then seven small blocks with camera configuration (b) were connected to each of E_1, E_2, E_3, and E_4 of the center block; in total 116 cameras in 29 small blocks. Figure 14 shows the temporal histories of uncertainty scores. Similar to the scores in camera configuration (a), our method could speed up camera topology calibration in contrast to the simple search scheme.

In the tracking period, the percentage of detected

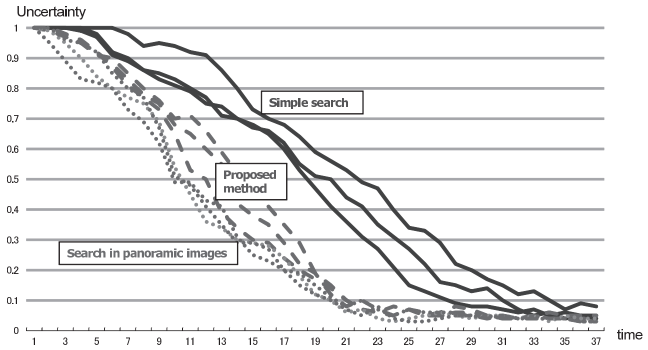


Fig. 14 Temporal histories of the uncertainty scores in the huge environment. Solid lines: simple search with partial images, Dotted lines: panoramic images, Broken lines: proposed method.

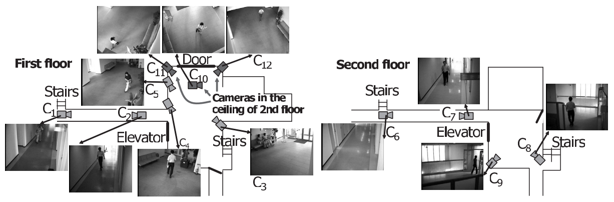


Fig. 15 View from above of the scene and image examples.

IN/OUT data was evaluated. Total IN/OUT data was known from the simulation data. In our proposed method, the mean percentages of three trials were 97% and 85% in camera configurations (a) and (b), respectively. Without the probabilistic camera topology, on the other hand, the percentages were 58% and 51%.

The comparative experiments were conducted with real cameras. To evaluate the results of different methods in the same object trajectories, first of all, a long image sequence was captured by each camera. Then each real image was regarded as a panoramic image and images segmented from the real image were regarded as its partial images. In the experiments, six partial images (3 pan divisions $\times$ 2 tilt divisions) were prepared. Assume that only one partial image could be observed at each moment.

Figure 15 shows the experimental environment and the examples of the images captured by cameras. Twelve 640×480 pixel cameras were used. Object detection was implemented with a simplified version of [25]. All detected pixels were then grouped into each object region based on connectivity of the detected pixels. The centroid of the connected pixels is regarded as the position of the object. For tracking in the partial image, each detected object region is identified with object regions in the previous frame based on proximity.

- When an object was first detected in any partial image, the object was tracked. If tracking was successful during three frames or more, an IN data was generated from the object data detected first.
- When tracking an object was unsuccessful for three frames (i.e. no object was detected at time t, $t-1$, and

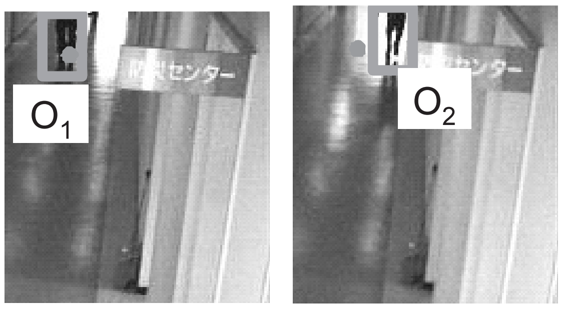
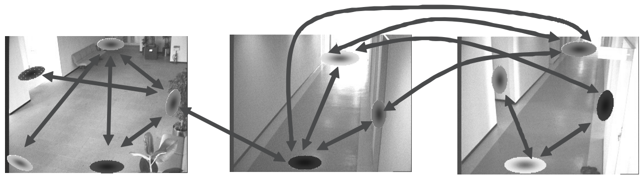
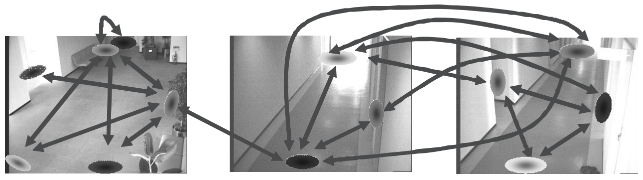


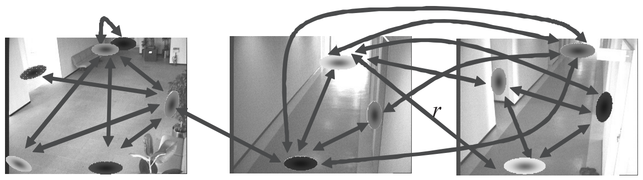
Fig. 16 Results of object detection and tracking. These images are the partial images of C_1 (right-upper partial images). Rectangles indicate detected objects. The centroids of objects O_1 and O_2 were regarded as the positions of IN/OUT data. Such varying positions are merged into the end point of a route (i.e. the mean position, $\boldsymbol{\mu}_r^B$ or $\boldsymbol{\mu}_r^E$, of route r). The end point is indicated by an orange circle in the figure.



(a) Results of simple search without the camera topology



(b) Results of our proposed method with the camera topology



(c) Results of fixed camera calibration with panoramic images

Fig. 17 Examples of detected routes. Arrows and ellipses indicate the detected routes and their beginning and end points, respectively. Only route r could not be detected in the proposed method.

$t-2$ near the object detected at $t-3$), an OUT data was generated from the object data detected at $t-3$.

Figure 16 shows the examples of detected objects around the border of a partial image. The centroids of detected object regions O_1 and O_2 were regarded as the positions of IN/OUT data. For reference, the mean position of the end point of a route, which was established by O_1, O_2, and IN/OUT data detected near them, is depicted by an orange circle in the figure. The positions of O_1, O_2, and the end point of the route in the panoramic image were (442, 18), (484, 20), and (449, 21), respectively.

Figure 17 (a) and (b) show the results of the topological calibration. For comparison, the result obtained by [20] from the original-size images are also shown in Fig. 17 (c) in which all routes were verified by hand. Note that the routes

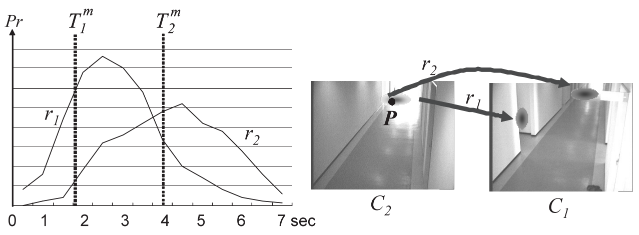


Fig. 18 Probability of object detection, $Pr(o,r)$.

that are close to each other are merged in the figures for viewability. With and without the camera topology model, 387 and 603 minutes were needed until the calibration was finished, respectively.

To demonstrate probabilistic information obtained by our method, examples of $Pr(o,r)$ are shown in Fig. 18 (a). The graph shows the temporal histories of $Pr(o,r_1)$ and $Pr(o,r_2)$ when object o left $\boldsymbol{P}$ in C_2; r_1, r_2, and $\boldsymbol{P}$ are depicted in Fig. 18 (b). The statistics of r_1 and r_2 were as follows: mean and variance values of $(\boldsymbol{x}_1^B, \boldsymbol{y}_1^B, \boldsymbol{x}_1^E, \boldsymbol{y}_1^E, t_1)$ were $(408, 39, 212, 187, 2.4)$, and $(272, 167, 296, 116, 0.30)$, respectively. Those of $(\boldsymbol{x}_2^B, \boldsymbol{y}_2^B, \boldsymbol{x}_2^E, \boldsymbol{y}_2^E, t_2)$ were $(356, 44, 393, 26, 4.4)$ and $(351, 205, 259, 289, 1.84)$. It can be seen that $Pr(o,r_1)$ and $Pr(o,r_2)$ were determined so that their peaks were located around the transit time measured by hand (indicated by T_1^m and T_2^m in Fig. 18).

In the tracking period, the percentage of the duration of object observation was evaluated. 100% objects were detected from the sequences of the original-size observed images. Object tracking was achieved for 300 minutes. In our proposed method with the camera topology model, the mean percentage of three trials was 91%. Without the probabilistic camera topology, on the other hand, the percentages was 67%. From these results, it can be demonstrated that our proposed method could improve both accuracy and efficiency compared with the simple search and tracking methods without the camera topology model.

6. Concluding Remarks

We proposed a method for estimating the topology of distributed pan-tilt cameras and its probabilistic model. To observe as many objects as possible for as long as possible, pan-tilt control is an important issue not only in a tracking period but also in a topology estimation period for efficient modeling. Our method controls the pan-tilt cameras so that every route has reliable probabilistic data and objects are detected as many as possible.

Future work includes the following aspects:

- Appearance similarity for object identification between different FOVs [27].
- Experiments with real pan-tilt cameras.

References

[1] A. Mittal and L.S. Davis, “M2Tracker: a multi-view approach to segmenting and tracking people in a cluttered scene,” Int. J. Comput.

Vis., vol.51, no.3, pp.189–203, 2003.

[2] S. Khan and M. Shah, "Consistent labeling of tracked objects in multiple cameras with overlapping fields of view," IEEE Trans. Pattern Aanal. Mach. Intell., vol.25, no.10, pp.1355–1360, 2003.

[3] K. Heath and L. Guibas, "Multi-person tracking from sparse 3D trajectories in a camera sensor network," Proc. ACM/IEEE International Conference on Didtributed Smart Cameras, 2008.

[4] H. Iwaki, G. Srivastava, A. Kosaka, J. Park, and A. Kak, "A novel evidence accumulation framework for robust multi-camera person detection," Proc. ACM/IEEE International Conference on Didtributed Smart Cameras, 2008.

[5] K.C. Ng, H. Ishiguro, M.M. Trivedi, and T. Sogo, "An integrated surveillance system-human tracking and view synthesis using multiple omni-directional vision sensors," Image Vis. Comput., vol.22, no.7, pp.551–561, 2004.

[6] R. Collins, O. Amidi, and T. Kanade, "An active camera system for acquiring multi-view video," Proc. International Conference on Image Processing, pp.517–520, 2002.

[7] N. Ukita and T. Matsuyama, "Real-time cooperative multi-target tracking by communicating active vision agents," Computer Vision and Image Understanding, vol.97, no.2, pp.137–179, 2005.

[8] V. Isler, S. Khanna, J. Spletzer, and C.J. Taylor, "Target tracking with distributed sensors: The focus of attention problem," Computer Vision and Image Understanding, vol.100, no.1–2, pp.225–247, 2005.

[9] B. Song, C. Soto, A.K. Roy-Chowdhury, and J.A. Farrell, "Decentralized camera network control using game theory," Proc. ACM/IEEE International Conference on Didtributed Smart Cameras, 2008.

[10] A. Azarbayejani and A. Pentland, "Real-time self-calibrating stereo person tracking using 3-D shape estimation from blob features," Proc. International Conference on Pattern Recognition, vol.3, pp.627–632, 1996.

[11] X. Chen, J. Davis, and P. Slusallek, "Wide area camera calibration using virtual calibration objects," Proc. IEEE Conference on Computer Vision and Pattern Recognition, vol.2, pp.520–527, 2000.

[12] L. Lee, R. Romano, and G. Stein, "Monitoring activities from multiple video streams: establishing a common coordinate frame," IEEE Trans. Pattern Aanal. Mach. Intell., vol.22, no.8, pp.758–767, 2000.

[13] V. Kettnaker and R. Zabih, "Bayesian multi-camera surveillance," Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp.253–259, 1999.

[14] B. Song and A. Roy-Chowdhury, "Robust tracking in a camera network: a multi-objective optimization framework," IEEE J. Sel. Top. Signal Process., vol.2, no.4, pp.582–596, 2008.

[15] J.E. Boyd, J. Meloche, and Y. Vardi, "Statistical tracking in video traffic surveillance," Proc. IEEE International Conference on Computer Vision, pp.163–168, 1999.

[16] O. Javed, Z. Rasheed, O. Alatas, and M. Shah, "Knight: a real time surveillance system for multiple and non-overlapping cameras," Proc. International Confernece on Multimedia and Expo, pp.649–652, 2003.

[17] J. Davis and X. Chen, "Calibrating pan-tilt cameras in wide-area surveillance networks," Proc. IEEE Internation Conference on Computer Vision, pp.144–149, 2003.

[18] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, "Tracking across multiple cameras with disjoint views," Proc. IEEE International Conference on Computer Vision, pp.952–957, 2003.

[19] D. Makris, T. Ellis, and J. Black, "Bridging the gaps between cameras," Proc. IEEE Conference on Computer Vision and Pattern Recognition, vol.2, pp.205–210, 2004.

[20] N. Ukita, "Probabilistic-topological calibration of widely distributed cameras," Machine Vision and Applications, vol.18, no.3–4, pp.249–260, 2007.

[21] Y.M. Li and B. Bhanu, "Utility-based dynamic camera assignment and hand-off in a video network," Proc. ACM/IEEE International Conference on Didtributed Smart Cameras, 2008.

[22] D.L. Mills, "Internet time synchronization: the network time protocol," IEEE Trans. Commun., vol.39, no.10, pp.1482–1493, 1991.

[23] T. Matsuyama, "Cooperative distributed vision - dynamic integration of visual perception, action and communication," Image Understanding Workshop, pp.365–384, 1998.

[24] Y. Linde, A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design," IEEE Trans. Commun., vol.28, no.1, pp.84–95, 1980.

[25] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "WallFlower: principles and practice of background maintenance," Proc. IEEE International Conference on Computer Vision, pp.255–261, 1999.

[26] J. Vermaak, A. Doucet, and P. Perez, "Maintaining multi-modality through mixture tracking," Proc. IEEE International Conference on Computer Vision, pp.1110–1116, 2003.

[27] C. Madden, E.D. Cheng, and M. Piccardi, "Tracking people across disjoint camera views by an illumination-tolerant appearance representation," Machine Vision and Applications, vol.18, no.3, pp.233–247, 2007.

Norimichi Ukita received the Ph.D. degree in Informatics from Kyoto University, Japan, in 2001. After working as an assistant professor at Nara Institute of Science and Technology (NAIST), he became an associate professor in 2007. He was a research scientist of PRESTO, Japan Science and Technology Agency (JST) from 2002 to 2006. He is now working at the Robotics Institute, Carnegie Mellon University as a visiting research scientist. His main research interests are object detection/tracking and pose/shape estimation of human body and clothing. He has received the best paper award from the IEICE in 1999.

Kunihito Terashita got his M.E. degree in engineering from Nara Institute of Science and Technology, Japan, in 2008. The theme of his master thesis was wide-area visual surceillance. He is now working at Brother Corporation.

Masatsugu Kidode received the B.E., M.E., and Ph.D. degrees in engineering from Kyoto University, Japan, in 1968, 1970, and 1973, respectively. After working at Toshiba Corpolation during 1970 and 1999, he joined at the graduate school of information science, Nara Institute of Science and Technology (NAIST), Japan in 2000 as a professor. He is a fellow of the following societies: IEEE, IAPR, and IPSJ. He served as general chair of IEEE ISWC200 and general chair of IAPR MVA2002. His research interests include image processing, robot vision, and intelligent human interfaces.