Machine Vision and Applications (2014) 25:1067–1076
DOI 10.1007/s00138-014-0605-6

ORIGINAL PAPER

Temporal synchronization in mobile sensor networks using image sequence analysis

Darlan N. Brito · Flávio L. C. Pádua · Guilherme A. S. Pereira

Received: 28 May 2013 / Revised: 11 December 2013 / Accepted: 27 February 2014 / Published online: 27 March 2014

Abstract This paper addresses the problem of estimating the temporal synchronization in mobile sensors' networks, by using image sequence analysis of their corresponding scene dynamics. Unlike existing methods, which are frequently based on adaptations of techniques originally designed for wired networks with static topologies, or even based on solutions specially designed for ad hoc wireless sensor networks, but that have a high energy consumption and a low scalability regarding the number of sensors, this work proposes a novel approach that reduces the problem of synchronizing a general number N of sensors to the robust estimation of a single line in $\mathbb{R}^{N+1}$. This line captures all temporal relations between the sensors and can be computed without any prior knowledge of these relations. It is assumed that (1) the network's mobile sensors cross the field of view of a stationary calibrated camera that operates with constant frame rate and (2) the sensors trajectories are estimated with a limited error at a constant sampling rate, both in the world coordinate system and in the camera's image plane. Experimental results with real-world and synthetic scenarios demonstrate that our method can be successfully used to determine the temporal alignment in mobile sensor networks.

Keywords Mobile sensor networks · Temporal alignment · Visual object tracking · Random sample consensus

D. N. Brito
Department of Natural and Applied Sciences,
Universidade Federal de Ouro Preto, João Monlevade,
MG 35931-026, Brazil
e-mail: darlan@decea.ufop.br

F. L. C. Pádua
Department of Computing, Centro Federal de Educação
Tecnológica de Minas Gerais, Belo Horizonte,
MG 30510-000, Brazil
e-mail: cardeal@decom.cefetmg.br

G. A. S. Pereira (✉)
Department of Electrical Engineering, Universidade Federal
de Minas Gerais, Belo Horizonte, MG 31270-010, Brazil
e-mail: gpereira@ufmg.br

1 Introduction

Mobile sensor networks have been identified as a key technology in several applications such as, military sensing, physical security, industrial and manufacturing automation, distributed robotics and environment monitoring [2,15,18]. In order to retrieve accurate semantic and geometric information from the monitored scene, all those applications demand on synchronized data, that is, the sensors data samples must be placed onto a single, global timeline [23].

Typically, the temporal misalignment between the sensors occurs when they have different sampling rates, or when there is a time shift between them. This generally happens because the internal clocks of the several sensors differ. In fact, even when these clocks are initially set to be the same (e.g. the sensors are activated simultaneously), in real hardware they will differ after some amount of time due to clock drift, caused by clocks counting time at slightly different rates [9,23].

Although synchronization can be manually performed, this approach is prone to human error, especially when there are several sensors. Moreover, it could not be used in real-time applications. Alternatively, the temporal alignment may be estimated using synchronization hardware or network connections [8]. Unfortunately, special hardware is not a practical solution for remote and wireless applications. Moreover, it is very complex to specify special hardware for synchronizing sensors of different technologies and vendors. On the other hand, the use of a network

connection to synchronize the sensors requires the application of special methods to deal with the nondeterminism in the network dynamics, such as propagation time or physical channel access time, which makes the synchronization task a very challenging problem in many scenarios [16,23].

In this paper, we present a novel alternative solution for temporally aligning multiple mobile sensors in a network using images from a single camera. Unlike existing techniques, which assume that each sensor is equipped with a video camera that can detect and track a target [12,22], our solution is based on a single camera that, for a given time slot, is able to observe the mobile sensor trajectory. Suppose, for example, a scenario where a group of mobile robots equipped with several sensors perform an inspection task. To have the data from the sensors synchronized in time, it would be sufficient that each robot, not necessarily at the same time, enter the field of view of a stationary camera and transmit its estimated position to a computer attached to this camera. Our method was inspired by the methods presented by our group in [1,13], which introduced the concept of *timeline*. This concept is also used in this paper, but in a very distinct application. Consider, for example, the scenario illustrated in Fig. 1 where several mobile robots transport the sensors to be synchronized and a camera observe the movements of the robots. In this scenario with N sensors, the timeline is a straight line in $\mathbb{R}^{N+1}$ that completely describes all temporal relations between the sensors and the stationary calibrated camera. Notice that the space considered has an additional dimension, indistinguishable from the other N, which refers to the camera. To compute the timeline, the trajectories of the moving sensors are assumed to be known, being related to a fixed reference frame and estimated with a limited error at a constant sampling rate. Moreover, it must be assumed that the sensors cross the camera's field of view during a given time interval. Importantly, that time interval should be only enough to apply our technique, which means that the sensors do not need to remain in the camera's field of view during their entire operations.

An interesting characteristic of the timeline is that even though its knowledge implies knowledge of the sensors' temporal alignment, we can compute points on the timeline without knowing this alignment [13]. Using this property as a starting point, the temporal alignment problem for N sensors is reduced to the problem of estimating a single line of $N+1$ dimensions from a set of appropriately-generated points in $\mathbb{R}^{N+1}$.

The remainder of this paper is organized as follows: Section 2 presents the problem definition. Section 3 covers our temporal synchronization algorithm. Experimental results and discussions are presented in Sect. 4, followed by the conclusions and suggestions for future work in Sect. 5.

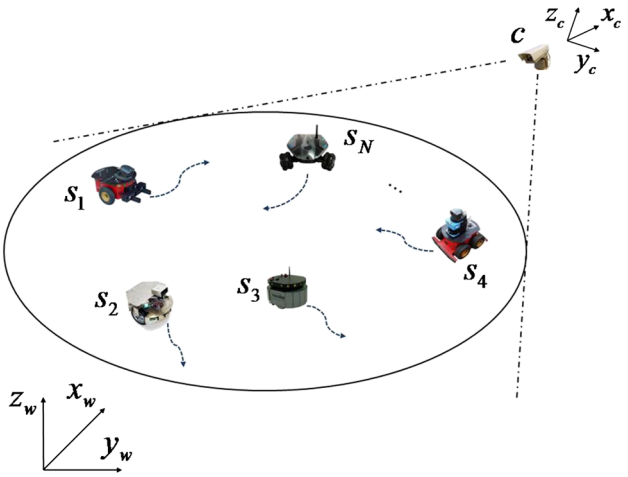

Fig. 1 A 3D scene is monitored by a set S of mobile sensors, for $S = \{s_1, \ldots, s_N\}$. The scene dynamics is captured by a stationary calibrated camera c

2 Problem definition

Consider a dynamic scene composed by N mobile sensors, as illustrated in Fig. 1. Suppose that this scene is viewed by a stationary calibrated camera c. By calibrated camera we mean a camera with known extrinsic (position and orientation) and intrinsic parameters (focal distance, lens distortion, pixel size, etc.) [10]. The camera operates with a constant frame rate and its field of view is crossed by all sensors. Importantly, each sensor s_i crosses the camera's field of view during a certain time interval, which is not necessarily identical for all of them. Therefore, the sensors may not be simultaneously viewed by the camera and may not remain in the camera's field of view during their entire operations. Assume that the trajectories of the moving sensors may be estimated at constant sampling rates (albeit not necessarily identical) in a fixed reference frame.

Analogously, we consider that each mobile sensor captures samples with a constant, unknown sampling rate and that the camera as well as the sensors are unsynchronized, i.e., they began capturing samples and frames at a different time with possibly-distinct sampling rates. In Fig. 2, for example, we illustrate the temporal misalignment between a camera and N sensors. In that example, the frame 458 of the camera corresponds in time to the samples 27, 323 and 159 of sensors s_1, s_2 and s_N, respectively. Therefore, the temporal misalignments between the camera and those sensors are $\Delta T_1 = 431$, $\Delta T_2 = 135$ and $\Delta T_N = 299$, respectively. Similarly, the temporal misalignments between those sensors are $\Delta T_{12} = 296$, $\Delta T_{1N} = 132$ and $\Delta T_{2N} = 164$. Our goal is to determine a global timeline that recovers the temporal alignment between the sensors, by using the synchronization offsets between the camera and those sensors.

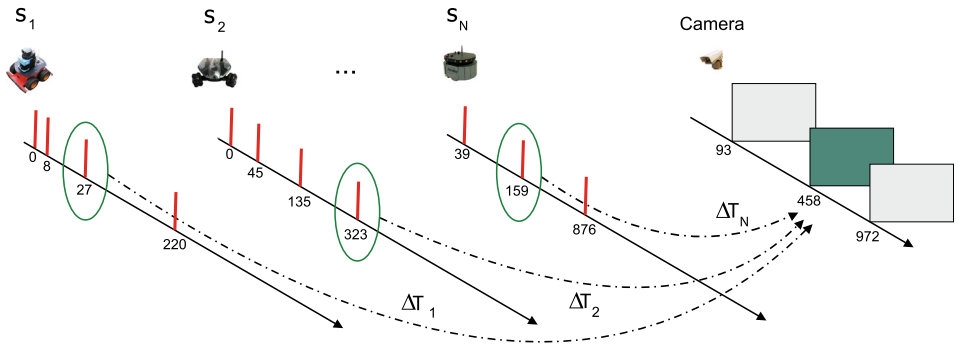

Fig. 2 Temporal misalignment between a camera and N mobile sensors. The frame *458* of the camera corresponds in time to the samples *27*, *323* and *159* of sensors s_1, s_2 and s_N, respectively. Our goal is to determine a global timeline that recovers the temporal alignment between the sensors by using the synchronization offsets $\Delta T_1, \Delta T_2, \ldots, \Delta T_N$, between the camera and the sensors

Since the sampling rate is assumed to be constant for both camera and sensors, the temporal coordinates (timestamps) of the sensors samples and the temporal coordinates (frame numbers) of the video sequence can be related by a one dimensional affine transformation [13]:

$$t_i = \alpha_i \, t_r \; + \; \beta_i \,, \tag{1}$$

where t_i and t_r denote the temporal coordinates of the ith sensor and the temporal coordinates of the camera, respectively. Parameters $\alpha_i \in \mathbb{R}$ and $\beta_i \in \mathbb{R}$ are unknown constants describing the temporal dilation and temporal shift, respectively, between the camera and the ith sensor [13].

Equation (1) represent pairwise temporal relations that induce a global relationship between the sample numbers of the sensors and the frame numbers of the camera. This relationship may be represented by a line $\mathcal{L}$ of $N+1$ dimensions, that is called *timeline*:

$$\mathcal{L} = \left\{ [\alpha_1 \;\; \cdots \;\; \alpha_{N+1}]^\top t_r + [\beta_1 \;\; \cdots \;\; \beta_{N+1}]^\top \,\middle|\; t_r \in \mathbb{R} \right\} \tag{2}$$

It is important to say that Eq. (2), which is a parametric representation of a straight line in a $N+1$-dimensional space is nothing but a composition of N lines in two-dimensions (2D). Actually, each of these lines, represented by Eq. (1), is a projection of the timeline in the plane formed by the camera time axis and the axis correspondent to sensor s_i time. Thus, the time misalignment among the N mobile sensors and the camera may be represented, indifferently, by a set of N lines in 2D as in Eq. (1) or by a single timeline in $N+1$-dimensions, as in Eq. (2). In this paper we prefer to use the timeline in $N+1$-dimensions for the sake of compactness of the notation.

The problem addressed in this work consists in to obtain an accurate estimate of the timeline based on images of the mobile sensors' workspace. To do that it is assumed that the sensors move along smooth 3D trajectories, which can be captured in the image plane of the camera by using the projection matrix obtained during its calibration [10], as well as by using standard trackers [6,7] that output trajectory segments as parametric curves. In this work, it is considered that the mobile sensors have the ability to localize themselves relatively to the world reference frame. Examples of mobile sensors include mobile robots and human beings carrying out sensor devices, such as, a global positioning system (GPS). Although the accurate localization of mobile robots and moving entities is still an open problem, which has been a major research topic in the past few years [3,14,20,24], our experiments in Sect. 4 show that the proposed methodology is robust to relatively large localization errors, thus requiring fairly simple localization techniques.

3 Methodology

In this section we present the proposed methodology to solve the problem posed in the previous section. To understand the basic idea behind this methodology consider the example in Fig. 3. In this figure, two sensors move along the trajectories $\mathbf{Q}_1(\cdot)$ and $\mathbf{Q}_2(\cdot)$ in a 3D scene, viewed by a camera. Suppose that these trajectories in the world coordinate system may be estimated by combining localization devices or using a GPS receiver. Therefore, $\mathbf{Q}_1(t_1)$ and $\mathbf{Q}_2(t_2)$ represent the sensors instantaneous positions at the temporal coordinates t_1 and t_2, respectively.

Since a camera is observing the movements of the sensors, using an object tracking algorithm [6,7] it is possible to compute $\mathbf{q}_1(\cdot)$ and $\mathbf{q}_2(\cdot)$, the corresponding trajectories traced by the sensors in the image plane. Assuming that the camera is calibrated, notice that the projections of the sensors' trajectories in the image plane may be also computed by using the projection matrix P. In this case, the projections of $\mathbf{Q}_1(\cdot)$ and $\mathbf{Q}_2(\cdot)$ may be represented by $\tilde{\mathbf{q}}_1(\cdot)$ and $\tilde{\mathbf{q}}_2(\cdot)$

Given these definitions, it is important to notice that the trajectories of the sensors in the image plane may be obtained in two distinct ways: (1) by the tracker ($\mathbf{q}_1(\cdot)$ and $\mathbf{q}_2(\cdot)$) and; (2) by the projections of the trajectories in 3D using the projection matrix P ($\tilde{\mathbf{q}}_1(\cdot)$ and $\tilde{\mathbf{q}}_2(\cdot)$). By determining correspondences between these trajectories we may also determine correspondences between the temporal coordinates of the frames of the video sequence and the sample numbers of the sensors.

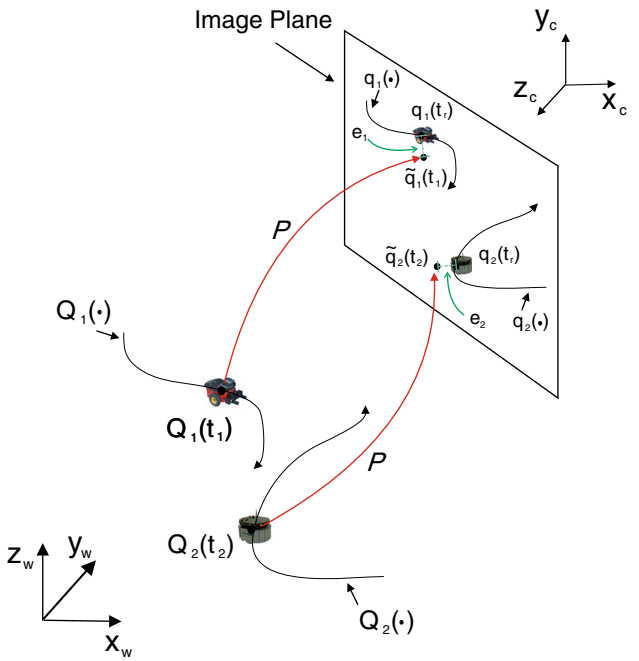

Fig. 3 Two sensors move along the trajectories $\mathbf{Q}_1(\cdot)$ and $\mathbf{Q}_2(\cdot)$ in a 3D scene, viewed by a camera. In this case, $\mathbf{Q}_1(t_1)$ and $\mathbf{Q}_2(t_2)$ represent the 3D sensors instantaneous positions at the temporal coordinates t_1 and t_2, respectively. The projections of those 3D positions in the image plane, namely, $\tilde{\mathbf{q}}_1(t_1)$ and $\tilde{\mathbf{q}}_2(t_2)$ may be computed by using the projection matrix P, obtained during the calibration of the camera

Coming back to the example in Fig. 3, consider, that $\mathbf{q}_1(t_r)$ and $\mathbf{q}_2(t_r)$ represent the sensors instantaneous positions in the image plane at frame t_r, computed by the tracker. Assuming that $\mathbf{q}_1(t_r)$ and $\mathbf{q}_2(t_r)$ correspond in time to, respectively, $\mathbf{Q}_1(t_1)$ and $\mathbf{Q}_2(t_2)$ in the world coordinate system, the projections $\tilde{\mathbf{q}}_1(t_1)$ and $\tilde{\mathbf{q}}_2(t_2)$ should coincide with $\mathbf{q}_1(t_r)$ and $\mathbf{q}_2(t_r)$ or stay at distances of e_1 and e_2 pixels caused by errors in the sensors' localization, in the camera calibration and/or in the tracking algorithms used.

Given this, it is also possible to establish correspondences between the temporal coordinates t_r, t_1 and t_2 of $\mathbf{q}_1(t_r)$, $\mathbf{q}_2(t_r)$ and $\tilde{\mathbf{q}}_1(t_1)$, $\tilde{\mathbf{q}}_2(t_2)$, respectively, since they represent the same 3D instantaneous positions of the sensors ($\mathbf{Q}_1(t_1)$ and $\mathbf{Q}_2(t_2)$). In fact, we may estimate for the camera and a general number N of sensors a set $\mathcal{V}$ of $N+1$-dimensional points with coordinates $[t_r \;\; t_1 \;\; \cdots \;\; t_n]$ that represent "candidate" temporal alignments for the camera and the sensors. Specifically, the set $\mathcal{V}$ defines a *voting space* that is built as follows:

$$\mathcal{V} = \left\{ [t_r \;\; t_1 \;\; \cdots \;\; t_n]^\top \;\middle|\; D\left(\mathbf{q}_i(t_r), \tilde{\mathbf{q}}_j(t_j)\right) \leq \varepsilon, \right\}, \qquad (3)$$

where $D(\cdot)$ denotes the Euclidean distance between the points $\mathbf{q}_i(t_r)$ and $\tilde{\mathbf{q}}_j(t_j)$, and ε denotes a tolerance in pixels.

After computing the set $\mathcal{V}$ described in Eq. (3) it is necessary to determine the most appropriate subset of candidate temporal alignments in $\mathcal{V}$ that will be used to determine the timeline that recovers the temporal alignment between the camera and the mobile sensors. Notice that this step is necessary because, in practice, set $\mathcal{V}$ will contain outliers. As in our previous work [1,13], to estimate the subset we use the RANSAC algorithm [4], which can be regarded as an algorithm for robust fitting of models in the presence of many data outliers.

RANSAC randomly chooses a pair of candidate temporal alignments to define the timeline, and then computes the total number of candidates that fall within an δ-distance of this line. These steps are repeated for a number of iterations. Provided that sufficient repetitions are performed, RANSAC is expected to identify solutions computed from outlier-free data. Thus, the two critical parameters of the algorithm are the number k of RANSAC iterations and the distance δ. In this paper, based on the methodology proposed in [4] and previously used by our group [13], we compute k to be 1,840 iterations. This value ensures that with probability 0.99, at least one randomly-selected pair of candidates is an inlier. To compute δ, we observe that δ can be thought of as a bound on the distance between tracked sensor locations in the camera and their associated projections. Thus, the proper value of δ depends on the camera resolution and on the set of real data. In the experiments and simulations shown in the next section, δ was chosen to be 30 pixels.

The final step of the methodology is to apply, over the subset of candidate temporal alignments estimated by RANSAC, a least-squares method to compute the timeline parameters. By combining the computed equations $t_i = \alpha_i t_r + \beta_i$ with parameters α_i and $\beta_i, i = 1, \ldots, N$, we may obtain new equations that capture the temporal relation between any two arbitrary sensors, as well as the line $\mathcal{L}$ that captures the global relationship among all the existing sensors.

To summarize the methodology, Fig. 4 presents a block diagram that shows all steps of the proposed approach. Basically, a voting space (Step 3) is generated by looking for pairs of closest points in two estimates of the sensors' trajectory: (1) an estimate given by the sensor's localization system, which is projected on the image plane using the camera calibration matrices (Step 2.1), and (2) an estimate given by a tracker, which works directly on the images given by the camera (Step 2.2). These pairs of closest trajectories' points will generate pairs of sample times that compose the voting space. We then look for a straight line in this space, the timeline, using RANSAC (Step 4), which is an outlier removing algorithm, and a least-squares optimization process, which in fact estimates the timeline parameters (Step 5).

Next section will present experimental results obtained both with synthetic and real-world data that illustrate and validate the proposed methodology.

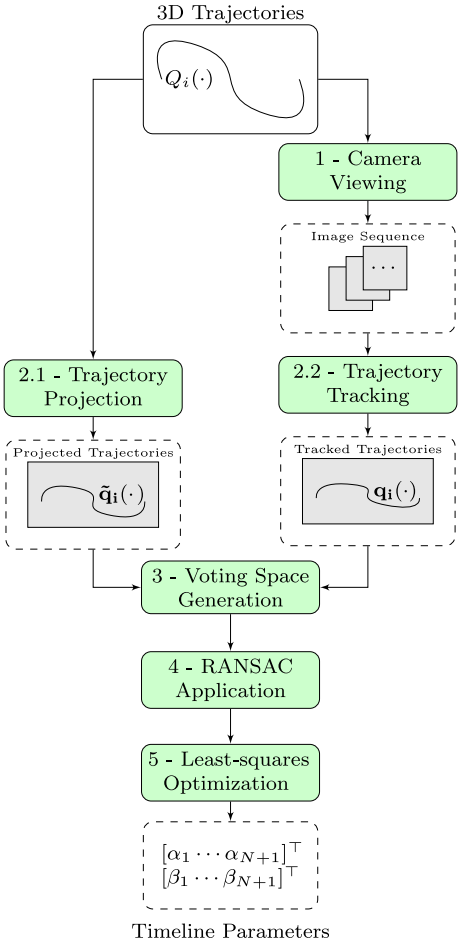

Fig. 4 Block diagram of the proposed approach

4 Experiments

In this section we evaluate, by means of simulated experiments, how our method is affected by errors: (1) in the sensors global localization, (2) in the image based tracker and (3) in the camera calibration. Also, using real-world data, we show a simple and practical application of the method on the synchronization of cameras installed on a group of mobile robots. We start presenting the results obtained in simulation.

4.1 Simulations

By using a Matlab based simulator, dynamic scenes were artificially created. These scenes contain up to 32 planar mobile sensors distributed in an ad-hoc wireless network. The movement of each mobile sensor is performed in a very simple way. Basically, a direction vector is randomly generated and the sensor moves in that direction at constant speed for a fixed time. After that, a new direction vector is generated. If the sensor reaches the workspace limit, defined as a circumference of radius 30 m, the direction vector is chosen to point normally to the limit and inside the workspace. Collisions among the sensors are not avoided. To observe the mobile sensors, we simulated a single calibrated camera whose intrinsic parameters were the same of a Sony DCR-TRV320 Digital Camcorder, used previously by the authors.

In Fig. 5 we present two typical voting spaces and their corresponding timelines. In Fig. 5a, we show a two dimensional voting space obtained in a scene with a single mobile sensor. The timeline in this case relates the camera and the sensor sampling rates. Figure 5b presents a timeline estimated in a scenario where two mobile sensors were observed by the camera. At this point, it is important to remember that the proposed algorithm does not necessarily compute the timeline using the three dimensional voting space directly. In fact, two voting spaces, one for each sensor, similar to the one in Fig. 5a may be used. By combining the timelines of each voting space it is possible to obtain the timeline that capture the temporal relationships between the sensors. Since the same procedure may be used for larger sets of mobile sensors, the complexity of the algorithm grows linearly with the number of sensors.

Note in Fig. 5 that the points in the voting spaces are dispersed around the encountered timelines. This dispersion occurs mainly because a point of the sensor trajectory projected on the image plane may be related [by Eq. (3)] to one or more points of the trajectory given by the tracker. This may happen due to three factors: (1) image tracker error, (2) camera calibration error, and (3) noise at the sensors' position estimation. To evaluate the effect of these factors in the success of the method in recovering timeline we perform a set of controlled simulations.

In the first case, we added white noise with zero mean and standard deviation varying from 1 to 10 pixels to the tracker result. The proposed algorithm was executed 100 times for each standard deviation value (we did the same for all experiments in this subsection). Figure 6a shows the percentage of success of the algorithm in recovering timelines that yields a temporal misalignment smaller than three samples. Notice that if the the tracker has a small noise ($<$4 pixels) the algorithm is able to find a good timeline in more than 90 % of the cases. Also, even for higher levels of noise, the rate of success of the algorithm is higher than 70 %. It is important to mention that these rates may decrease for about 80 and 50 %, respectively, if temporal misalignments smaller than one sample is considered. This is the need of several applications.

We perform a similar study with the camera calibration error. To generate a matrix with a given projection error e, we begin with the actual calibration matrix, add a small constant (10^{-5}) to each element, measure the projection error, and iterate until the error becomes equal to e. This method was inspired by the one proposed in [13]. Using such an approach, the result for temporal alignments smaller than

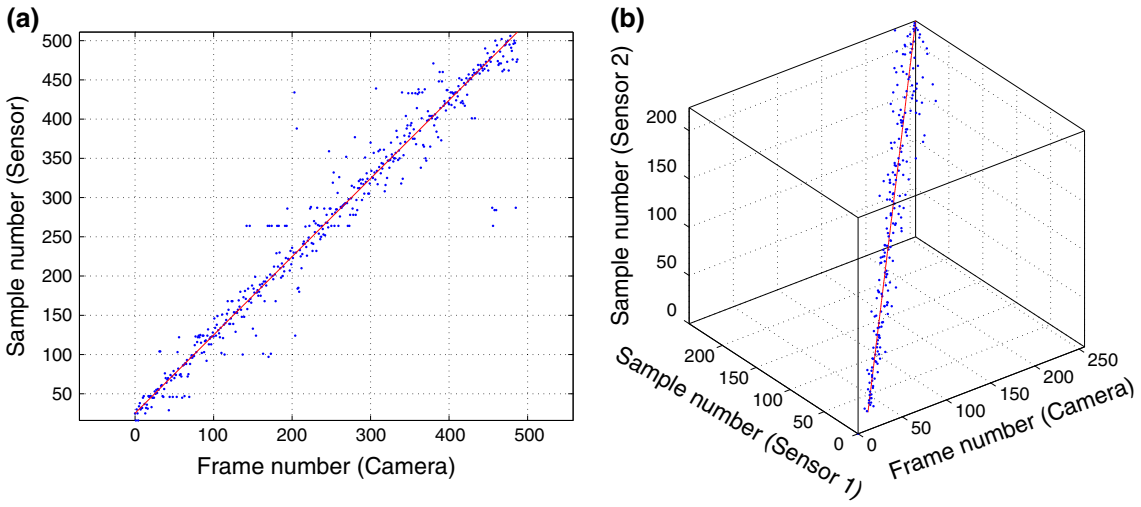

Fig. 5 Examples of 2D (**a**) and 3D (**b**) voting spaces for an artificial scenario with one and two sensors, respectively. In both cases, the computed timeline is also presented

three samples with projection errors varying from 1 to 10 pixels is shown in Fig. 6b. Notice that the rate of success in this case decreases faster with the error if compared with the previous case, indicating a stronger dependence of the method on the camera calibration.

Finally, we have performed experiments where we vary the noise in the sensor 3D positioning. In this case, we added white noise with zero mean and standard deviation varying from 1 to 10 % of the size of the workspace (30 m in our case). The result for temporal alignments smaller than three samples is shown in Fig. 6c. Observe that the method seems to be very robust to localization errors, what makes it a practical tool for synchronizing sensors with simple localization devices. However, it was observed during the simulations that this result also depends on the sensors' path. A path that crosses itself several times is more sensitive to localization errors, once it can generate more outliers in the voting space. This will be also observed in the next subsection, where we present an experiment with actual robots acting as mobile sensors.

4.2 Real-world data

We applied the proposed method to synchronize sensor data collected by three mobile robots moving in a laboratory environment. The movements of the robots are similar to the ones used in the sensors of the previous section. The mobile robots used are iRobot's Create platforms equipped with simple netbooks running Linux. The data to be synchronized are video images originated from the netbooks' embedded webcams. Each robot captures an image from the webcam and store it on the hard drive together with localization information given by a visual based "GPS like" system. Since we are not using a real time operating system, in order to keep the constant sample time required by the method, we have slowed down the program responsible for saving the data by using a *sleep()* function. With this, the sample rate was reduced to about 3 Hz. Actually, we cannot know the exact sample time, since the time needed to store the image on a file and other time delays are unknown and vary from one computer to another. Therefore, the problem that we are dealing with in this section is a practical one, which involves, not only the synchronization of the sensors, but also the estimation of their unknown sample times.

To perform the synchronization, we also recorded, with a Sony DCR-SR62 Digital Camcorder, a video from the robots' movements. As required by our method, the camera was previously calibrated in relation to the world reference frame, which is the same used by the localization system of the robots. The camcorder frame rate is 30 Hz. Using a simple particle filter based tracker, we obtained the robots trajectories on the image plane as shown in Fig. 7.

By comparing the robot trajectory projected on the image plane with the tracker output in Fig. 7, it is possible to see that only in small parts they are close to each other. This is mainly due to a limitation of the tracker used, that was not able to satisfactory handle multiple similar targets, occlusions, and light changes. This limitation generated voting spaces with several outliers, as can be seen in Fig. 8.

Each point of the voting spaces in Fig. 8 was generated by searching for the pixels in the robot trajectories projected on the image plane that are within a distant of 50 pixels of each pixel given by the tracker. This distance was chosen experimentally. In this way, it can observed that there are several points in the voting space with the same value in the vertical axis. Figure 8 also shows the "actual" timeline and the estimated timeline. The actual timeline was computed manually based on visual events introduced solely

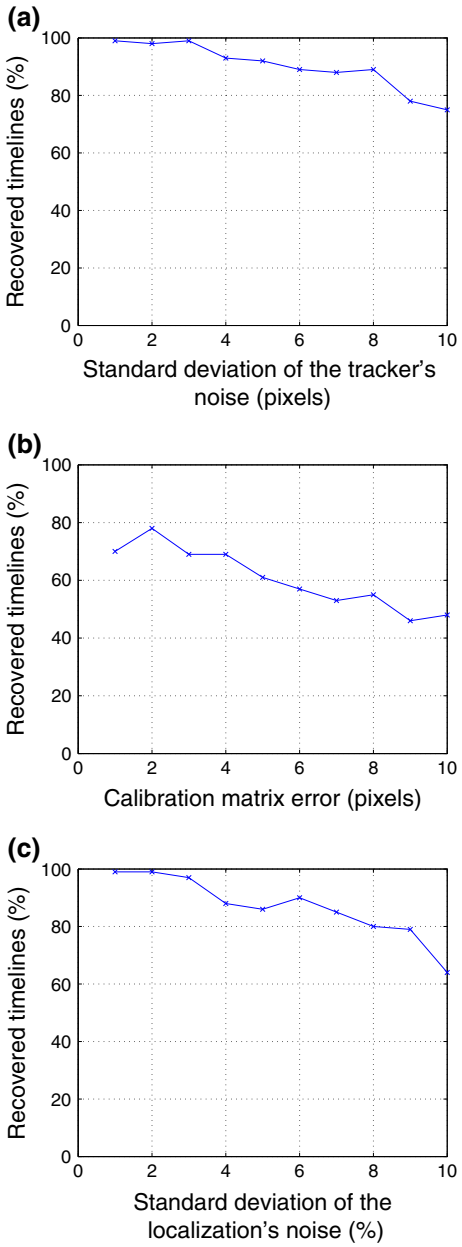

Fig. 6 Rate of success of the proposed method when it is subject to **a** noise in tracker, **b** camera calibration errors, and **c** sensor's localization noise. All figures represent the percentage of timelines recovered by the method that yields temporal alignments smaller than three samples

for this purpose. To cause these events we simply turn off the laboratory's illumination for a few seconds during the experiment.

Notice in Fig. 8 that, except for Robot 1, whose trajectory generated more outliers in the region close to the timeline, the estimated timelines are very similar to the actual ones. Numerically, the encountered parameters can been seen in Table 1. In this table, α_i and β_i are the parameters of the line in Eq. (1) that relates the temporal coordinates of the ith robot and the temporal coordinates of the camera.

Notice in Table 1 that for Robot 1 the difference between the temporal shift of the two timelines is about 20 samples, indicating a large synchronization error. The errors for the other robots are much smaller than this, what indicates the influence of the tracker and of the robot trajectory. Observe in Fig. 7 that the trajectory of Robot 1 is the one that has more self-intersections and the one that yields the worst tracking performance.

With the data in Table 1 it is clearly possible to make the pairwise synchronization of the sensors by combining the timeline equations. Also, it is possible to estimate the sample time of each robot. For instance, since α_2 is the slope of the timeline and the camera frame rate is 30 Hz, Robot 2 sample rate can be estimated as $30 \times 0.097 = 2.91$ Hz. In the next section we present the conclusions of the paper.

4.3 Discussion

With the simulated and real-world results of the previous subsections, we have shown some characteristics of the proposed approach. The bottom limit of 70 % of success in recovering a good timeline as noise in the tracker increases up to 10 % in its standard deviation indicates that the proposed approach is not highly dependent on good trackers. This suggests that the method could rely on simple and efficient algorithms. The results obtained with the real robots corroborate with this observation. In that case, we have used a tracker with a extremely poor performance (see Fig. 7) and, despite the large number of outliers in the voting space, could still recover the timeline. These results may also indicate that high resolution cameras are not mandatory for the method, since it seems to be robust to small variations on the position of the tracked objects on the image.

The method also seems to be robust to the sensor's localization errors. In our simulations, 80 % of timeline recovery was achieved for 2.7 m (9 % of the area) of error. This localization error can be achieved by simple GPS devices, what makes the method suitable to be used with several mobile sensors.

The quality of the calibration, however, seems to be a factor that highly interfere on the timeline recovery. This will limit the use of the method to situations where it is possible to determine the extrinsic parameters of the camera in relation to a fixed reference frame. In situations where we cannot guarantee that the camera will remain still, because of wind or other external factor, for example, we will probably have difficulties to use the proposed methodology.

Another key observation is related to the sensors' movements. In one hand, the method cannot be applied to static sensors. For the case of static sensors, network based methods such as [5,17], or other alternatives that use additional hardware, such as [11], must be used. On the other hand, we have experimented some difficulties in recovering the time-

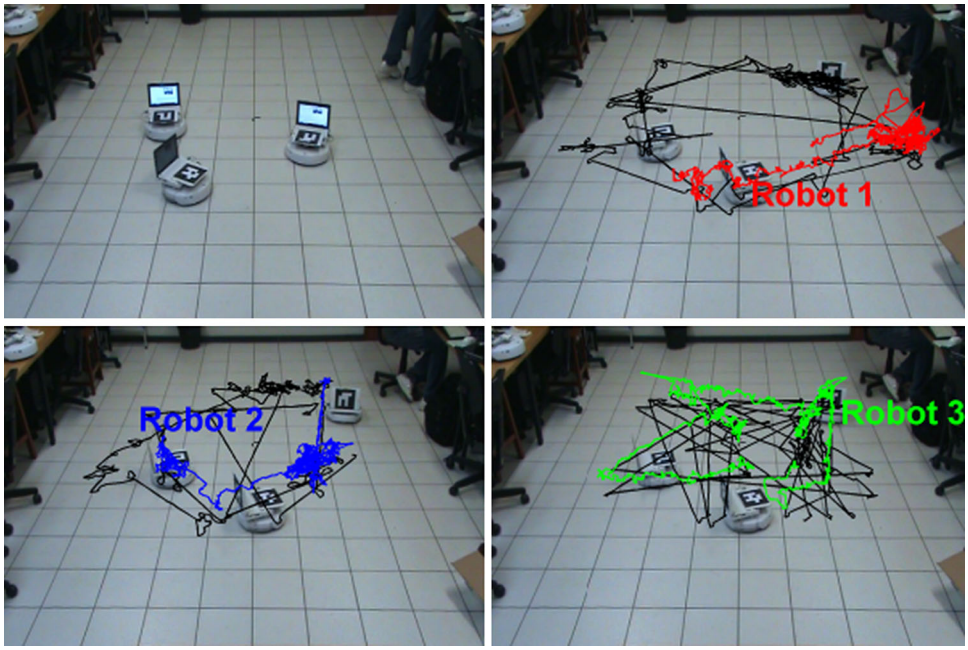

Fig. 7 Robots' trajectories obtained by the tracker (*in color*) and their respective workspace trajectory projected using the camera projective matrix (*in black*). From the *left to the right*, *top to bottom* we see a frame captured by the camcorder, and data from *robots 1*, *2*, and *3*, respectively (color figure online)

line, due to an excess of outliers in the voting space, in situations where we have random sensor movements that caused several self-intersections in the sensor's paths. This suggests that future work should consider the study of approaches for controlling the paths of the sensors so that the synchronization recovering is facilitated.

In the results presented in Sect. 4.1 we have considered that we have success in recovering a timeline when this line yielded temporal misalignment smaller than three samples. If we consider that the frame rate of a typical camera is 30 Hz, misalignments of up to 100 ms were obtained. This kind of synchronization may be useful for some applications that involve slow dynamics, such as temperature and humidity measurements and even robot localization, but cannot be used in applications that require a high level of synchronization, such as scheduling for media access control in some network protocols [21]. Since our approach is highly dependent on the camera frame rate, for these applications it cannot be used and must be replaced, for example, by the ones that rely on package exchange through the wireless network [5,17,19], where the authors report synchronization errors of about tenths of microseconds. However, whenever larger synchronization errors are acceptable, our method has a strong advantage over the previously published ones, which is energy consumption. As seen in [21], when there is a need to exchange extra packages through the network, the network based methods are generally very power consuming. Methods that reduce the communication to save energy, on the other hand, can only achieve synchronization in several minutes or even hours [19,21]. Our method do not require the transmission of extra packages, since the position of the sensor is the only information required. This is, in general, the basic information transmitted by the mobile sensors. Moreover, once the data is available, synchronization may be achieved in a few minutes.

5 Conclusions and future work

This paper presented a new methodology for synchronization of data gathered by networked mobile sensors. The method is based on the observation of the sensors' movements by an external camera and in the determination of a *timeline*, a function that captures all temporal relationships among the sensors. In some specific scenarios, the proposed method may be an interesting alternative for standard synchronization methodologies based on wireless communication and specific hardware.

The main requirements of our method are: (1) each sensor node must be mobile and able to estimate (and communicate) its position in relation to a global reference frame; (2) the movement of each sensor must be observed for a given and limited time by a video camera calibrated in relation to the same global frame; (3) there is a tracker able to estimate the sensors movements in the image plane; the tracker does not need to make a distinction among the sensors, although this would simplify the voting space and facilitate the timeline recovery.

We presented synthetic and real-world experiments that showed some important characteristics of the methodology. First, the method is scalable, since it depends on the pairwise synchronization of each sensor with the camera. Therefore, one could think on the synchronization of N sensors by estimating N timelines in 2D instead of estimating a single

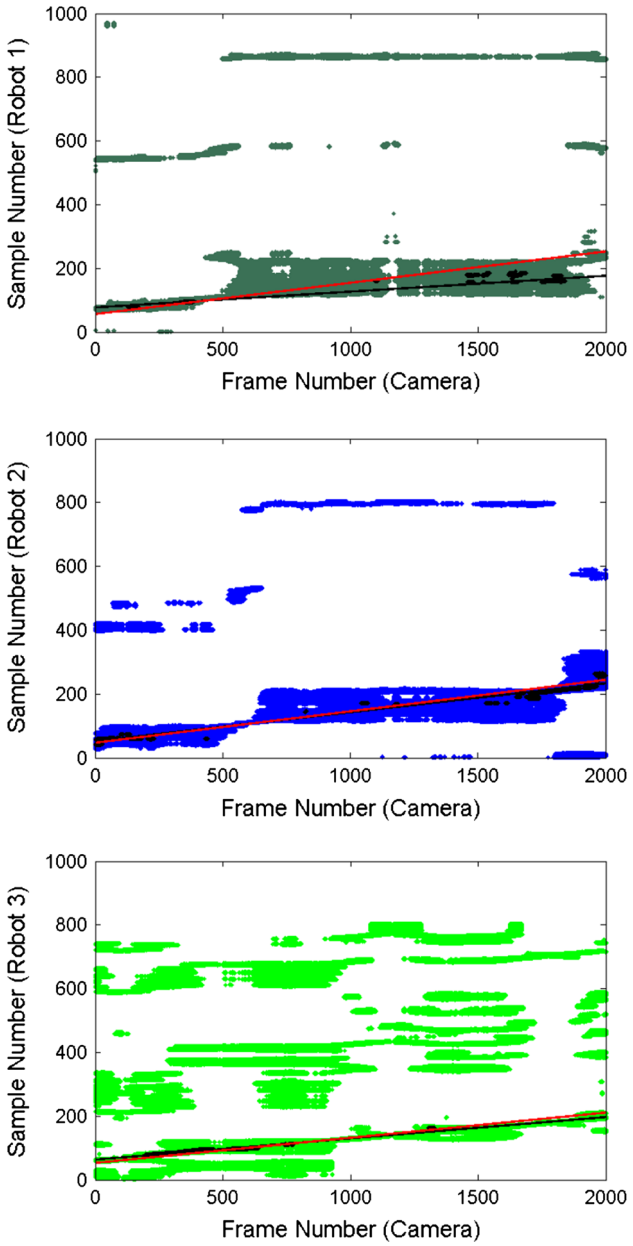

Fig. 8 Voting spaces obtained for the trajectories in Fig. 7. From the *top to the bottom*, the figures represent the 2D voting spaces that relates the camcorder frames with the samples of robots 1, 2, and 3, respectively. The *color dots* represent the points of each voting space. The *black dots* represent the points obtained after RANSAC. The *black lines* represent the timeline obtained using those points and the *red lines* represent the actual align between the robot's webcam and camcorder (color figure online)

Table 1 Parameters of the timelines in Fig. 8

Robot	Real		Estimated	
	α_i	β_i	α_i	β_i
1	0.098	55.99	0.045	76.83
2	0.098	46.70	0.097	51.65
3	0.079	52.65	0.067	62.79

timeline in a $N + 1$-dimensional space. Second, the method present some robustness to noise on tracker and localization data but, on the other hand, is somewhat sensible to errors in the camera calibration matrix. Finally, the method is dependent on the sensor's trajectory. Good trajectories would be the ones that do not intersect themselves for several times. Each intersection point generates at least two points in the voting space, being one of them an inlier and the others outliers. Since a large number of outliers can make more difficult the timeline estimation, a small number of intersections on the sensor trajectories would facilitate the computation.

Future work includes the proposition of a navigation strategy for the mobile sensors, so that the timeline recovery is improved. Another interesting direction for future research is to think on how to use a moving camera to synchronize the sensors. This comes in consonance with recent advances in the area of unmanned air vehicles, which are generally equipped with cameras and could fly above the sensors.

Acknowledgments This work would not be possible without the financial support of FAPEMIG/Brazil, CEFET-MG, and CAPES/Brazil. Guilherme Pereira holds a scholarship from CNPq/Brazil.

References

1. Brito, D.N., Pádua, F.L.C., Pereira, G.A.S., Carceroni, R.L.: Temporal synchronization of non-overlapping videos using known object motion. Pattern Recognit. Lett. **32**, 38–46 (2011)
2. Chong, C., Kumar, S.: Sensor networks: evolution, opportunities, and challenges. Proc. IEEE **91**(8), 1247–1256 (2003)
3. Ding, Y., Krislock, N., Qian, J., Wolkowicz, H.: Sensor network localization, euclidean distance matrix completions, and graph realization. Optim. Eng. **11**, 45–66 (2010)
4. Fischler, M., Bolles, R.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**, 381–395 (1981)
5. Huang, G., Zomaya, A.Y., Delicato, F.C., Pires, P.F.: An accurate on-demand time synchronization protocol for wireless sensor networks. J Parallel Distrib. Comput. **72**(10), 1332–1346 (2012)
6. Isard, M., MacCormick, J.: BraMBLe: a bayesian multiple-blob tracker. In: IEEE International Conference on Computer Vision, pp. 34–41 (2001)
7. Jepson, A., Fleet, D., El-Maraghi, T.: Robust on-line appearance models for visual tracking. IEEE Trans. Pattern Anal. Mach. Intell. **25**, 1296–1311 (2003)
8. Kitahara, I., Saito, H., Akimichi, S., Onno, T., Ohta, Y., Kanade, T.: Large-scale virtualized reality. In: IEEE Conference on Computer Vision and Pattern Recognition (2001)
9. Lamport, L.: Time, clocks, and the ordering of events in a distributed system. Commun. ACM **21**, 558–565 (1978)
10. Ma, Y., Soatto, S., Kosecka, J., Sastry, S.S.: An Invitation to 3-D Vision—From Images to Geometric Models, 1st edn. Springer, New York (2003)
11. Medina, C., Segura, J.: Accurate time synchronization of ultrasonic TOF measurementsin IEEE 802.15.4 based wireless sensor networks. Ad Hoc Netw. **11**(1), 442–452 (2013)
12. Messom, C.H.: Synchronisation of vision based sensor networks with variable frame rates. Int. J. Comput. Appl. Technol. **39**, 153–158 (2010)

13. Pádua, F.L.C., Carceroni, R.L., Santos, G.A.M.R., Kutulakos, K.N.: Linear sequence-to-sequence alignment. IEEE Trans. Pattern Anal. Mach. Intell. **32**, 304–320 (2010)
14. Pereira, G., Kumar, V., Campos, M.: Localization and tracking in robot networks. In: IEEE International Conference on Advanced Robotics, pp. 465–470 (2003)
15. Pottie, G.J., Kaiser, W.J.: Wireless integrated network sensors. Commun. ACM **43**, 51–58 (2000)
16. Sivrikaya, F., Yener, B.: Time synchronization in sensor networks: a survey. IEEE Netw. **18**, 45–50 (2004)
17. Sommer, P., Wattenhofer, R.: Gradient clock synchronization in wireless sensor networks. In: International Conference on Information Processing in Sensor Networks, pp. 37–48. Washington, DC (2009)
18. Stankovic, J.A., Wood, A.D., He, T.: Realistic applications for wireless sensor networks. In: Nikoletseas, S., Rolim, J.D. (eds.) Theoretical Aspects of Distributed Computing in Sensor Networks, pp. 835–863. Springer, Berlin (2011)
19. Swain, A., Hansdah, R.: An energy efficient and fault-tolerant clock synchronization protocol for wireless sensor networks. In: International Conference on Communication Systems and Networks, pp. 1–10 (2010)
20. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics, 1st edn. The MIT Press, USA (2005)
21. Wang, Y., Nunez, F., Doyle, F.J.: Energy-efficient pulse-coupled synchronization strategy design for wireless sensor networks through reduced idle listening. IEEE Trans. Signal Process. **60**(10) (2012). doi:10.1109/TSP.2012.2205685
22. Wolf, W., Velipasalar, S., Schlessman, J., Chen, C.Y., Lin, C.H.: Real-time distributed tracking. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 1389–1392 (2007)
23. Wu, Y.C., Chaudhari, Q., Serpedin, E.: Clock synchronization of wireless sensor networks. IEEE Signal Process. Mag. **28**, 124–138 (2011)
24. Zhang, H., Shen, G., Jin, D.: An accurate localization method for video sensor network nodes. Adv. Mater. Res. **225–226**, 531–535 (2011)

Darlan N. Brito received the Bachelor degree in physics from the Universidade Federal de Minas Gerais (UFMG), Brazil, in 2005, and the M.Sc. degree in mathematical and computational modeling from the Centro Federal de Educacação Tecnológica de Minas Gerais (CEFET-MG), Brazil, in 2008. He has been an assistant professor of computer engineering at the Universidade Federal de Ouro Preto (UFOP) since 2010. His research interests include computer vision, pattern classification and content-based image and video retrieval.

Flávio L. C. Pádua received the Bachelor degree in Electrical Engineering from Universidade Federal de Minas Gerais (UFMG), Brazil, in 2000, and the M.Sc. and Ph.D degrees in Computer Science from the same university, in 2002 and 2005, respectively. From 1998 to 1999, he studied at Technical University of Berlin in Germany, where he has fulfilled one academic year of his undergraduation in the scope of an fellowship program established by the governments of Brazil (CAPES) and Germany (DAAD). During that period, he worked as a research assistant at the Institute for Machine Tools and Factory Management (IWF). He has been working, since 2005, as an Associate Professor at the Department of Computing of CEFET-MG, specifically, on the Graduate Program in Mathematical and Computational Modeling and on the Undergraduate Program in Computer Engineering. In 2010, he worked as a Visiting Professor at Institut Universitaire de Technologie 1 (IUT1) and the laboratory Grenoble Images Parole Signal Automatique (GIPSA-LAb) in Grenoble, France. His research interests include Computer Vision, Content-Based Image and Video Retrieval and Automated Visual Inspection in Industry.

Guilherme A. S. Pereira received the B.S. and M.S. degrees in electrical engineering and the Ph.D. degree in computer science from the Federal University of Minas Gerais (UFMG), Belo Horizonte, Brazil, in 1998, 2000, and 2003, respectively. Dr. Pereira received the Gold Medal Award from the Engineering School of UFMG for garnering first place among the electrical engineering students in 1998. He was, from November 2000 to May 2003, a Visiting Scientist at the General Robotics, Automation, Sensing and Perception (GRASP) Laboratory, University of Pennsylvania, Philadelphia. Since July 2004, he is an Associate Professor of the Electrical Engineering Department at the Federal University of Minas Gerais (DEE/UFMG), where he is the director of the Computer Systems and Robotics (CORO) Laboratory, one of the laboratories that compose the Group for Reaserch and Development of Autonomous Vehicles (PDVA) at UFMG. His research interests include cooperative robotics, robot navigation, autonomous vehicles development, computer vision, and distributed sensing. Dr. Pereira is a Member of Sociedade Brasileira de Automática and a Senior Member of IEEE.