Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 150541, 10 pages
http://dx.doi.org/10.1155/2015/150541

Research Article

A Multisource Heterogeneous Data Fusion Method for Pedestrian Tracking

Zhenlian Shi, Yanfeng Sun, Linxin Xiong, Yongli Hu, and Baocai Yin

Beijing Key Laboratory of Multimedia and Intelligent Software Technology, College of Metropolitan Transportation, Beijing University of Technology, Beijing 100124, China

Correspondence should be addressed to Yanfeng Sun; yfsun@bjut.edu.cn

Received 25 September 2014; Accepted 25 February 2015

Academic Editor: Shueei M. Lin

Traditional visual pedestrian tracking methods perform poorly when faced with problems such as occlusion, illumination changes, and complex backgrounds. In principle, collecting more sensing information should resolve these issues. However, it is extremely challenging to properly fuse different sensing information to achieve accurate tracking results. In this study, we develop a pedestrian tracking method for fusing multisource heterogeneous sensing information, including video, RGB-D sequences, and inertial sensor data. In our method, a RGB-D sequence is used to position the target locally by fusing the texture and depth features. The local position is then used to eliminate the cumulative error resulting from the inertial sensor positioning. A camera calibration process is used to map the inertial sensor position onto the video image plane, where the visual tracking position and the mapped position are fused using a similarity feature to obtain accurate tracking results. Experiments using real scenarios show that the developed method outperforms the existing tracking method, which uses only a single sensing dataset, and is robust to target occlusion, illumination changes, and interference from similar textures or complex backgrounds.

1. Introduction

Visual pedestrian tracking is an important research subject in computer vision. Many object tracking methods for visual pedestrian tracking have been developed recently. In typical methods, time series information, such as the Kalman filter [1] and the particle filter [2], is used for tracking. Recently, sparse representation and compressed sensing theories have also been introduced to represent target features in visual tracking [3, 4]. However, traditional visual tracking methods perform poorly in the presence of challenges such as target occlusion, objects with similar appearances, background changes, and illumination changes. To solve these problems, researchers are increasingly fusing various sensors for target tracking [5, 6]. These methods can be generally divided into two types. The first type is the passive method in which the target does not participate in the tracking process. For example, Ros and Mekhnacha developed a Bayesian occupancy filter for human tracking by fusing different types of sensing information [7]. Kang et al. developed a robust body tracking method by fusing visual and thermal images [8]. The second type is the active tracking method in which the target participates in the tracking process via wearable or carry-on sensors, such as MEMS inertial sensors, which can be used to locate the target from acceleration and gyroscope sensing information. The passive and active pedestrian tracking methods have their respective advantages for different application scenarios. The passive method is often used in public field surveillance where the target is unaware of being tracked. The active method can be used for self-navigation. However, some applications demand robust tracking performance, such as elder care, child protection, and criminal surveillance, for which neither the passive nor the active tracking methods produce satisfactory performance. For these cases, using wearable sensors combined with passive visual sensors is an ideal tracking solution. In this study, we developed a pedestrian tracking method that integrates the passive and active tracking methods. We used heterogeneous sensors, including Android smartphones, Kinect cameras, and video cameras, to implement pedestrian tracking and positioning

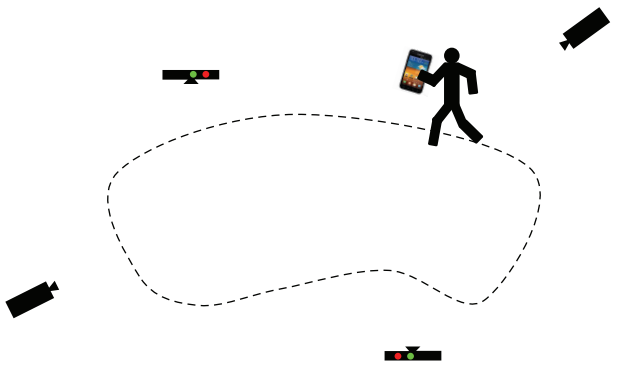


Figure 1: Multisensor pedestrian tracking scenario.

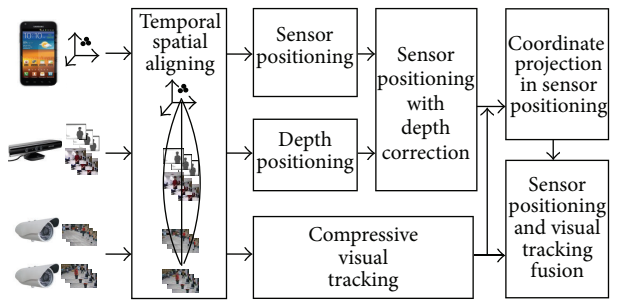


Figure 2: Multisource heterogeneous data fusion for pedestrian tracking.

over a large range (see Figure 1). Different sensing data were fused in the tracking procedure to obtain more robust results.

In principle, using different sensors for tracking should improve tracking efficiency and accuracy. However, many challenges must be met to realize these improvements, such as the spatial and temporal alignment of different sensing data, developing data and feature representations of different types of data, and the use of a proper fusion method for heterogeneous data. In this study, these issues were explored using video, depth, and inertial sensing data for pedestrian tracking. Figure 2 shows how multisource heterogeneous sensing data were aligned, processed, represented, and fused at different levels using different methods. The two primary contributions of this study are as follows: (1) RGB-D data that were captured by the depth sensor were used to eliminate the cumulative error from the inertial sensor positioning, which is a critical issue in using inertial sensors for long-term tracking; and (2) the resulting inertial sensor positions were fused with visual tracking results to solve target occlusion, illumination changes, and other difficult problems in traditional visual tracking. The developed method was tested by constructing a multisource sensing platform that was verified using data captured from real scenarios. The experimental results showed that the developed method exhibited good position and tracking performance.

The rest of the paper is structured as follows. In Section 2, the inertial sensor positioning with depth correction is presented. The fusion of inertial sensor positioning and visual tracking is described in Section 3. The experimental results are presented in Section 4. The paper is concluded in Section 5.

2. Inertial Sensor Positioning with Depth Correction

2.1. Target Positioning with Inertial Sensor. The inertial sensor consisted of an acceleration sensor and a gyroscopic sensor. The acceleration sensor was used to obtain a sequence of changes in the three acceleration components. We determined the relationship between the step length and its duration from a set of sequence samples. We used the gyroscopic sensor to obtain the angular rotation speed around three axes: roll, pitch, and yaw. The inertial sensor positioning process in this study consisted of the following primary steps: gait detection from the acceleration data, calculating the step length according to the time, obtaining steering information from the gyroscope data, and calculating the location coordinates.

2.1.1. Gait Detection from Acceleration Data. First, we defined a coordinate system for the walking motion of the test subject. The positive x-axis was taken to be the direction of the forward motion of the test subject, and the y-axis was defined as the line that was perpendicular to and in the same plane as the x-axis. The z-axis was defined as being perpendicular to the x-y plane. The values of all three coordinates changed with motion of the test subject. The Z-component of the acceleration was found to be periodic and the most significant one of the three components. The variation in the acceleration is shown in Figure 3.

The plot of the three components of the acceleration shows that the Z-component of the acceleration was clearly more periodic than the other X- and Y-components. Thus, a gait could be detected from the variation in the vertical acceleration during the walking motion of the test subject, which was in turn used to determine the number of steps taken by the test subject. The gait detection results are shown in Figure 4.

2.1.2. Calculation of Step Length. Walking speed varies among individuals; thus, different test subjects were trained over different time periods to calculate the step length.

The following training procedure was used. Many sessions were conducted in which the test subject was instructed to walk with different step lengths, where the step length and time were measured for each step. We used the results of many training sessions to construct a graph of the time-dependent step length, which is shown in Figure 5. Longer steps took more time, and shorter steps took less time. A program was used to fit the data points with a straight line. The equation of this straight line was the time-length formula for the test subject.

When the acceleration sensor was used to locate the target, we used the duration of each step and the time-step length formula to calculate the step length. Although significantly more accurate methods are available for calculating the step

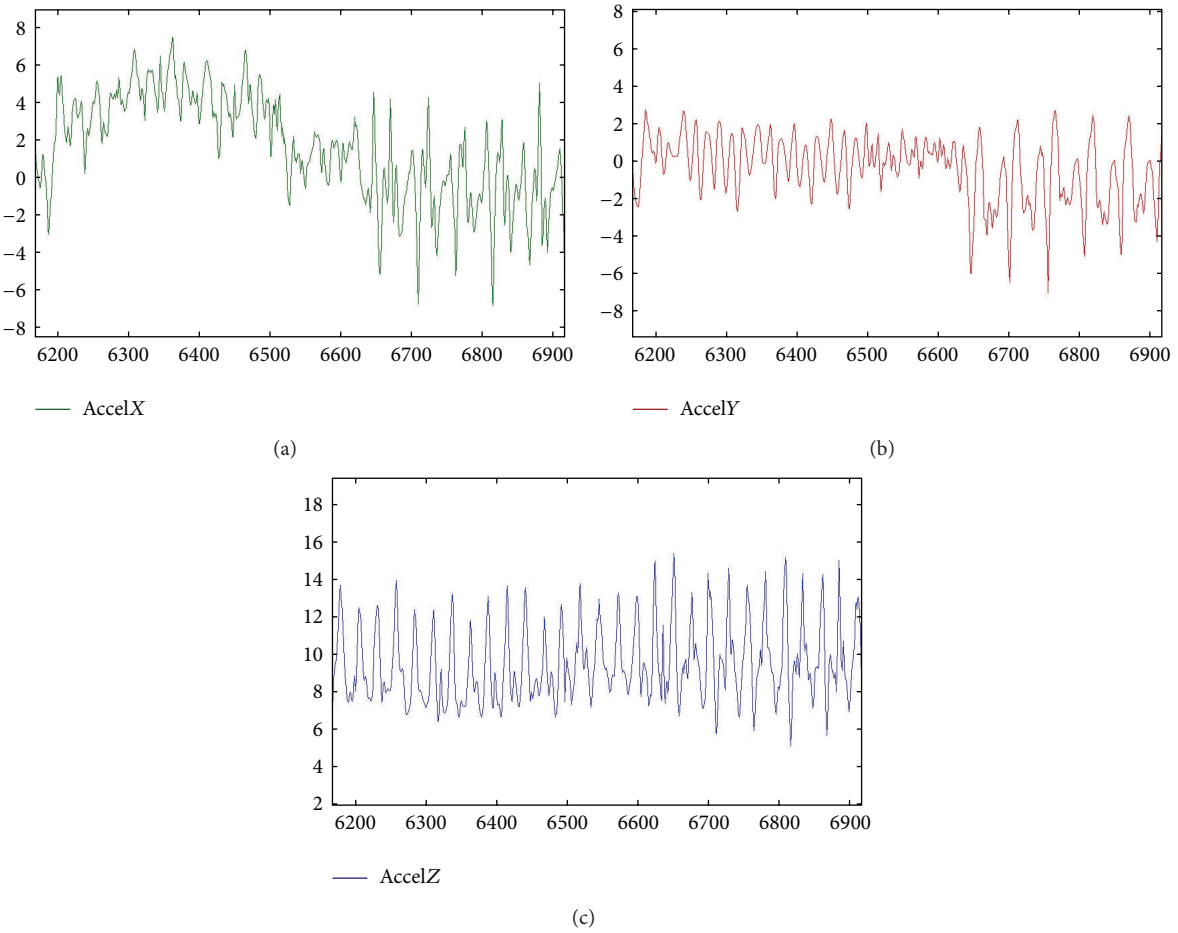


Figure 3: X-, Y-, and Z-components of acceleration.

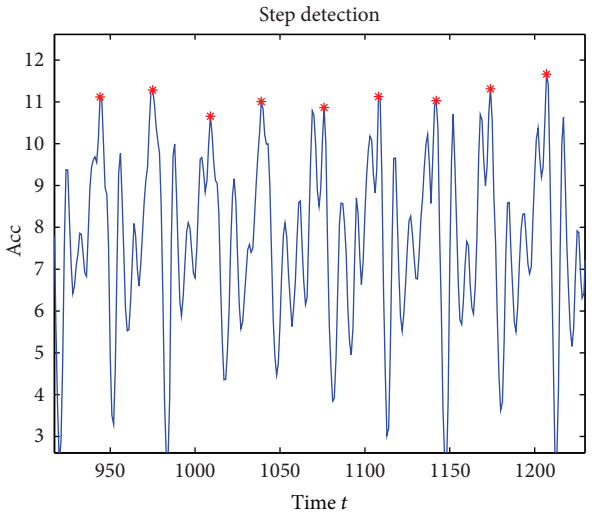


Figure 4: Gait detection.

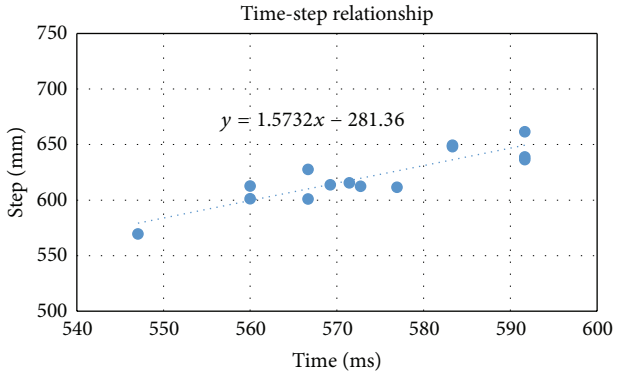


Figure 5: Step length as a function of time.

length, our method was fast, highly efficient, and sufficiently precise for most cases, thereby satisfying our experimental needs.

2.1.3. Obtaining Steering Information from Gyroscope Data. The steering angle of the human body has three components that correspond to rotation around three axes, namely, horizontal roll, vertical pitch, and perpendicular yaw, as shown in Figure 6.

We primarily used the yaw angle to determine steering information for the test subject. We used the gyroscope to directly obtain the angular velocity in all three directions,

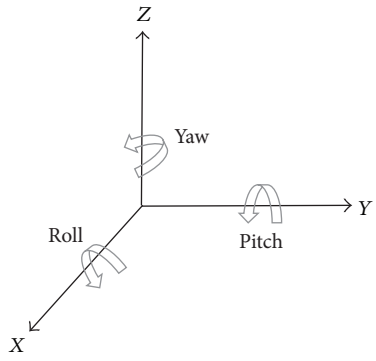


Figure 6: Three components of steering angle.

from which we calculated the steering angle. However, the sensor was not placed very accurately in most cases, and we could not ensure that the three axes of the sensor were coincident with those of the test subject. Thus, we had to correct the yaw value to increase the accuracy of the results. The steering angle of the target was calculated as follows:

$$\boldsymbol{\Psi}_{k+1} = \boldsymbol{\Psi}_k + \mathbf{E}_{b2n,k} \cdot \left(\widetilde{\boldsymbol{\omega}}_{b,k} - \widehat{\boldsymbol{\omega}}_{b,k}^{\text{bias}}\right) \cdot \Delta t,$$

$$\mathbf{E}_{b2n,k} = \begin{pmatrix} 1 & \sin\theta\tan\phi & \cos\theta\tan\phi \\ 0 & \cos\theta & -\sin\theta \\ 0 & \dfrac{\sin\theta}{\cos\phi} & \cos\theta\cos\phi \end{pmatrix}, \tag{1}$$

where k denotes the timestamp of the current measurement, $\boldsymbol{\Psi}$ denotes the Euler angle matrix (consisting of the three angles of the roll, pitch, and yaw), $\mathbf{E}_{b2n}$ denotes the rotation rate correction matrix between the body and navigation frame, $\widetilde{\boldsymbol{\omega}}_b$ denotes the vector of the measured angular rate, $\widehat{\boldsymbol{\omega}}_b^{\text{bias}}$ denotes the intrinsic error estimation of the angular rate, Δt denotes the time interval between two measurements, and θ and ϕ denote the roll and pitch, respectively.

2.1.4. Calculation of Location Coordinates. We calculated the location coordinates using

$$\begin{aligned} x_t &= x_{t-1} + L_t \cdot \cos\theta_t, \\ y_t &= y_{t-1} + L_t \cdot \sin\theta_t, \end{aligned} \tag{2}$$

and the known values of the length and steering angle of each step: $t > 0$ denotes the current step number, L_t denotes the length of the tth step, and θ_t denotes the direction at the tth step, which represents the accumulated angle from step 1 to t.

We calculated the subsequent position of the target from its current coordinates using the step distance and the steering angle of each step. However, a very large error accumulated as the tracking time increased, representing a critical problem with the inertial sensor positioning method. There were many potential sources of error. The primary source of error may have been the inaccurate estimation of the intrinsic error in the gyroscopic sensor's angular speed $\widehat{\boldsymbol{\omega}}_b^{\text{bias}}$. Other sources of this cumulative error were the installation of the sensors and the individual walking postures. Although various methods are available to correct the cumulative positioning error produced by inertial sensors, these methods perform poorly for long-term tracking. We overcame this problem by correcting the cumulative error using the RGB-D data captured by the depth camera.

2.2. Positioning Using the RGB-D Sequence. A depth camera, such as a Kinect, can simultaneously capture visual and depth information, namely, the RGB-D sequence, making the three-dimensional position of the target easy to obtain. The primary task of RGB-D sequence-based tracking is to construct a dynamic tracking model. Here, we implemented dynamic tracking using a particle filter tracking framework. In the visual particle filter model, RGB-D data are used to represent the target features. We described the features of the test subject in both the visual and depth domains and then defined the fusion features of the RGB-D data.

The HSV color model is considered to be consistent with human color perception and robust to light changes. Thus, we used HSV color to represent the visual images of the RGB-D sequence. For a target with a rectangular image area, the histogram in HSV space was computed as the target visual feature, which is usually represented in vector form; that is, $H = [h_1, \ldots, h_i, \ldots, h_n]^T$, where n is the number of bins and h_i is the frequency of the ith grade. Compare with visual sequences, fewer features can be represented by depth data. The depth data that were captured by the current RGB-D camera exhibited poor precision; thus, we used a simple shape-based feature to represent the depth feature of the RGB-D sequence. Specifically, we transformed the depth area of the target into a binary image, where entry values of unity represented the human body and entry values equal to zero represented the background or other objects. The binary depth feature was computed using the following function:

$$B(x, y) = \begin{cases} 1, & \text{if } \left\| \text{Dp}(x, y) - \overline{\text{Dp}} \right\| \le \varepsilon, \\ 0, & \text{else}, \end{cases} \tag{3}$$

where $\text{Dp}(x, y)$ denotes the depth at (x, y) in the target area, $\overline{\text{Dp}}$ denotes the average depth of the target area, and ε is a threshold value that was assigned based on experience. Figure 7 shows the procedure for generating the binary depth image. Once the depth binary image was created, the depth feature of the target was represented as $D = [d_1, \ldots, d_i, \ldots, d_m]^T$ by concatenating the entries of the binary image.

First, we defined the similarity feature for the visual and depth features to integrate the visual and depth features into a uniform representation for tracking. Here, we adopted the Bhattacharyya distance as the visual similarity. The depth similarity is defined as the ratio of the shared area between the observation particle and the template to the template foreground. Thus, the similarity of the fused visual and depth features is defined as the product of the visual and depth similarities. The similarity of the RGB-D feature was used to implement the dynamic tracking in the RGB-D sequence using particle matching.

Figure 7: Binary image representation of depth feature.

2.3. Inertial Sensor Position with Depth Correction. Applying the inertial sensor positioning method over long tracking periods results in a critical cumulative error. The depth position and tracking were highly accurate, but the tracking scale was limited to the local area. Thus, we developed a depth correction for the inertial sensor position method. In this method, the depth camera positioning result was used to correct the cumulative error of the inertial sensor positioning results. The correction method consisted of determining a mapping that transformed the inertial sensor position trajectory into a new trajectory that was consistent with the local positioning result in the depth camera frame (Figure 8). Here, we used a thin plate spline (TPS) to construct the transformation.

The TPS is a nonrigid spline mapping function that offers advantages such as considerable flexibility, global smoothing, and ease of computation [9]. Here, the TPS was as a mapping $f : R^2 \to R^2$ that was obtained by aligning two sets of points on the reference and the target. The TPS transformation for the inertial sensor positioning reference trajectory H_1 and the corrected target trajectory H_2 was determined using the following corresponding control point sets that were simultaneously positioned by the inertial sensor and depth camera:

$$M_k = \left\{H_{kj} \mid H_{kj} = \left(x_{kj}, y_{kj}\right),\ j = 1, \ldots, M,\ k = 1, 2\right\}, \tag{4}$$

where M_1 denotes the inertial sensor positioning location and its aligned depth positioning result, M_2, when the depth camera was woken up for tracking. M denotes the number of points. The definition of TPS results implies the following interpolation condition for f:

$$f\left(M_{1j}\right) = M_{2j}, \quad j = 1, 2, \ldots, M. \tag{5}$$

At the same time, the constraint on smooth bending for the TPS required the minimization of the following bending energy function:

$$E(f) = \iint_{R^2} \left(\frac{\partial^2 f}{\partial x^2}\right)^2 + 2\left(\frac{\partial^2 f}{\partial x \partial y}\right)^2 + \left(\frac{\partial^2 f}{\partial y^2}\right)^2 dx\, dy. \tag{6}$$

The TPS was decomposed into affine and nonaffine components as follows:

$$f(P) = Pd + Kw, \tag{7}$$

where P is a point on the inertial sensor positioning reference trajectory, H_1, with the homogeneous coordinate $(1, x, y)$, and d denotes a 3×3 affine transformation matrix. The TPS kernel is denoted by K, which is a $1 \times M$ vector of the form $K = (K_1(P), \ldots, K_M(P))$, such that $K_j(P) = \|P - M_{1j}\|$, $j = 1, \ldots, M$, and w is $M \times 3$ warping coefficient matrix for the nonaffine deformation.

The d and w matrixes were determined to obtain the TPS transformation. The following energy function was minimized using the least squares method to determine the optimal solution:

$$E(d, w, \lambda) = \frac{1}{M}\sum_{j=1}^{M} \left\|M_{2j} - f\left(M_{1j}\right)\right\| + \lambda E(f), \tag{8}$$

where λ is the weight that controls the smoothness of the component. Given a fixed λ, there is a unique minimum for the energy function.

After the TPS transformation was obtained, the reference walking trajectory, H_1, was corrected to H_2. In the entire tracking procedure, the correction was performed after the target is detected and tracked by the depth cameras.

3. Fusing Visual Tracking with Inertial Sensor Positioning

The traditional visual tracking method usually becomes unstable when target occlusion and illumination changes are encountered. Thus, we developed a method to fuse the former inertial positioning results with visual tracking to obtain robust tracking. The Kinect positions the target locally (3-4 meters) by fusing the texture and depth features. The local position is then used to eliminate the cumulative error resulting from the inertial sensor positioning. A camera calibration process is used to map the inertial sensor position onto the video image plane, where the visual tracking position and the mapped position are fused using a similarity feature to obtain accurate tracking results.

The inertial sensor positioning generally retrieves the real spatial location of the target, but visual tracking is implemented in the image plane. Thus, the inertial sensor positioning results were mapped onto the image plane. This mapping was obtained using the following projective translation:

$$\mathbf{x} = P\mathbf{X}, \tag{9}$$

where $\mathbf{X}$ denotes the inertial sensor positioning coordinates, $\mathbf{x} = (x, y)$ denotes the projected coordinates in the image plane, and P denotes the projective matrix. Here, we used an automatic training method to determine the projective matrix P. In the training procedure, a test subject was tracked using both the visual and inertial sensing data sequences. Then, the tracking positions from the two sequences were aligned to obtain the training set of $\mathbf{x}$ and $\mathbf{X}$. Finally,

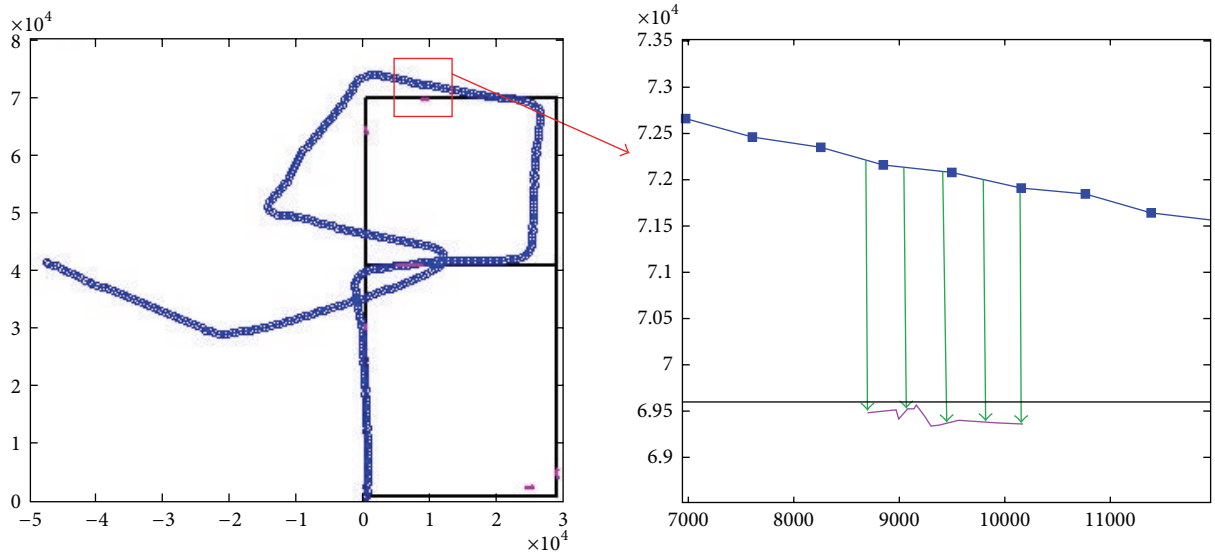


FIGURE 8: Sensor positioning with depth correction: blue line represents inertial sensor positioning trajectory, pink line represents trajectory of depth camera, and black line represents actual trajectory.

the projective matrix P was calculated using the least squares method with the training samples of $\mathbf{x}$ and $\mathbf{X}$.

For the visual sequence, we used a compressive tracking algorithm [10] to track the target in the image plane. The underlying principle of compressive tracking is to construct a dynamic template to match the target with the observations. The matching level was estimated using a Bayesian scheme to select an observation v_i from the foreground ($y = 1$) or the background ($y = 0$):

$$H(v) = \log\left(\frac{\prod_{i=1}^{n} p(v_i \mid y = 1)\, p(y = 1)}{\prod_{i=1}^{n} p(v_i \mid y = 0)\, p(y = 0)}\right) = \sum_{i=1}^{n} \log\left(\frac{p(v_i \mid y = 1)}{p(v_i \mid y = 0)}\right), \tag{10}$$

where $p(v_i \mid y = 1) \sim N(\mu_i^1, \sigma_i^1)$; $p(v_i \mid y = 0) \sim N(\mu_i^0, \sigma_i^0)$; μ^1 and σ^1 denote the mean and standard deviation of the target sample; and μ^0 and σ_i^0 denote the mean and standard deviation, respectively, of the background sample. We usually selected the target sample close to the target area and the background sample far from the target area. The candidate region that maximized $H(v)$ was the final target position. At the same time, we used the tracked target position to update the target parameter and the background parameters:

$$\mu_i^1 \longleftarrow \lambda\mu_i^1 + (1 - \lambda)\,\mu^1,$$
$$\sigma_i^1 \longleftarrow \sqrt{\lambda\left(\sigma_i^1\right)^2 + (1 - \lambda)\left(\sigma^1\right)^2 + \lambda(1 - \lambda)\left(\mu_i^1 - \mu^1\right)^2}, \tag{11}$$

where λ was a parameter used to maintain the consistency between the two positioning results.

We fused the visual tracking position $\mathbf{x}$ with the inertial sensor positioning result $\mathbf{x}'$ by defining a consistency measurement for the two positioning results using the following function:

$$\mathrm{sim}\left(\mathbf{x}, \mathbf{x}'\right) = \exp\left(-\frac{1}{2}\left(\mathbf{x} - \mathbf{x}'\right)^T \Sigma^{-1}\left(\mathbf{x} - \mathbf{x}'\right)\right), \tag{12}$$

where Σ is the covariance of the inertial sensor positioning samples. Generally, when visual tracking fails because of occlusion or illumination changes, the consistency measurement decreases dramatically. This consistency measurement was used to detect visual tracking errors using the threshold method. However, to ensure continuous tracking when errors were incurred, we developed an error recovery solution. We developed a multitemplate compressive tracking method for error recovery tracking.

In the original compressive tracking method, the tracked target sample is always used to update the foreground sample parameters μ^1 and σ^1 and the background sample parameters μ^0 and σ^0 even when errors are incurred in the tracking procedure. In our revised compressive tracking method, we employed a mechanism to terminate the updating of the parameters when an error occurred and constructed a template buffer to save the optimal template in the tracking history. For convenience, the compressive tracking parameters are denoted by $\Psi = \{\mu^1, \sigma^1, \mu^0, \sigma^0\}$. The history template parameter sets are denoted by $\Psi_1, \Psi_2, \ldots, \Psi_N$, where N is the number of history templates. Different parameters yield different current consistency measurements. Thus, the consistency measurement of the original method was also revised using $H(v, \Psi)$. When no errors in the tracking procedure were incurred, in addition to updating the parameter Ψ as in the aforementioned operation, we updated the history parameter sets if the current parameter Ψ was a better match to the target, that is, if there was a Ψ_i such that

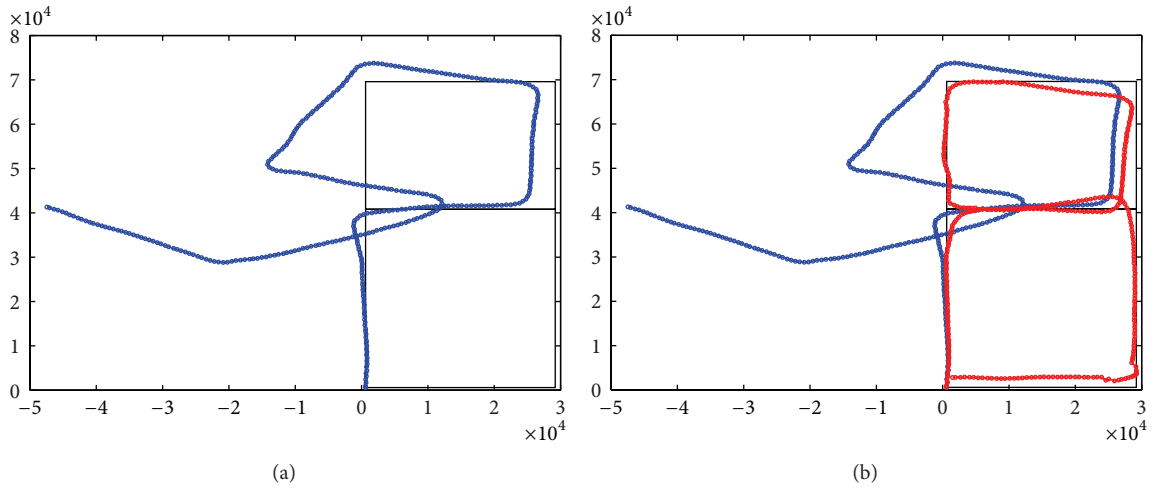


Figure 9: (a) Original inertial sensor positioning result and (b) result obtained using inertial sensor positioning with depth correction.

$H(v, \Psi) > H(v, \Psi_i)$. Thus, the following template parameter was replaced by Ψ:

$$\Psi^r = \arg \min_{1 \le i \le N} H(v, \Psi_i). \tag{13}$$

When the consistency measure of the current sample was below the given threshold, a tracking error was considered to have occurred. In this case, we used the projected coordinate of the inertial sensor positioning result to reset the position of the visual tracking position. At the same time, we stopped updating the parameter Ψ and selected the following optimal template in the history parameter sets to restore the tracking procedure:

$$\Psi^* = \arg \max_{1 \le i \le N} H(v, \Psi_i). \tag{14}$$

4. Experiments and Results

We verified the developed multisource heterogeneous data fusion method by conducting several experiments using data captured from real scenarios. We used an Android smartphone to collect inertial sensor data, including three-dimensional accelerometer and gyroscope data. The collection frequency was 50 Hz. The smartphone was installed in the test subject's pocket and fixed on the subject's body with a belt. Before the experimental tracking began, the test subject stood steadily in the same position for over 10 seconds, and the inherent noise in the sensor measurements was estimated. At the same time, the RGB-D sequence data were captured by several Kinect sensors. A visual video was also captured by a high-resolution camera. Three experiments were performed in this study.

In the first experiment, the performance of the inertial sensor positioning with the depth correction was tested. This experiment was performed in a corridor that was shaped like a figure eight in a teaching building with dimensions of 70 m × 30 m. The test subject walked along the corridor. Figure 9(a) shows the original inertial sensor positioning results. The first half of the positioning results were more accurate than the second half for which a growing error was observed. We used correction data from the Kinect camera to improve the positioning result. Then, we used the TPS-based method to correct the positioning result. Figure 9(b) shows that the correction method produced an accurate positioning result. During the experiment, we used the difference in the durations to calculate the average deviation before and after the correction. A 20-second gap was used in this study, and the average deviation among the different time gaps is shown in Table 1. Before correction, the average deviation in the results increased sharply in time, and, after correction, the average deviation declined considerably. Thus, the method lowered the average deviation from 17.120 meters to 2.119 meters, which was a sufficiently accurate result. The average errors of the two methods were also computed. The developed method decreased the average error from 17.120 m to 2.119 m.

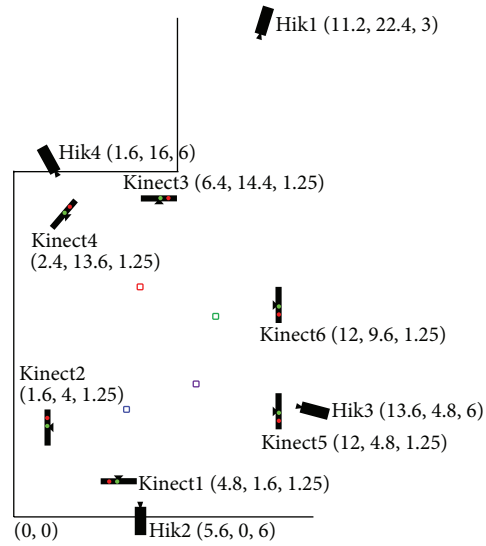
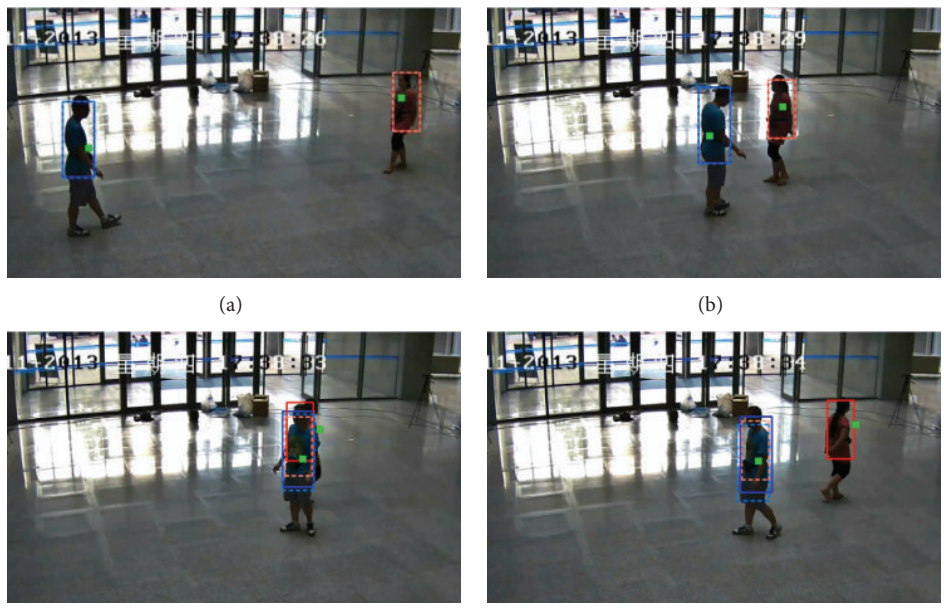


Figure 10: Layout of experimental scenario.



(a) (b) (c) (d)

Figure 11: Comparison of visual tracking method and inertial sensor positioning fusion method for a simple scenario: green dots show inertial sensor positioning results projected onto image plane; blue and red boxes show tracking boxes for different targets; dashed boxes show results from visual tracking only; and solid boxes show results from fusion of inertial sensor positioning and visual tracking.

Table 1: Average error at different times.

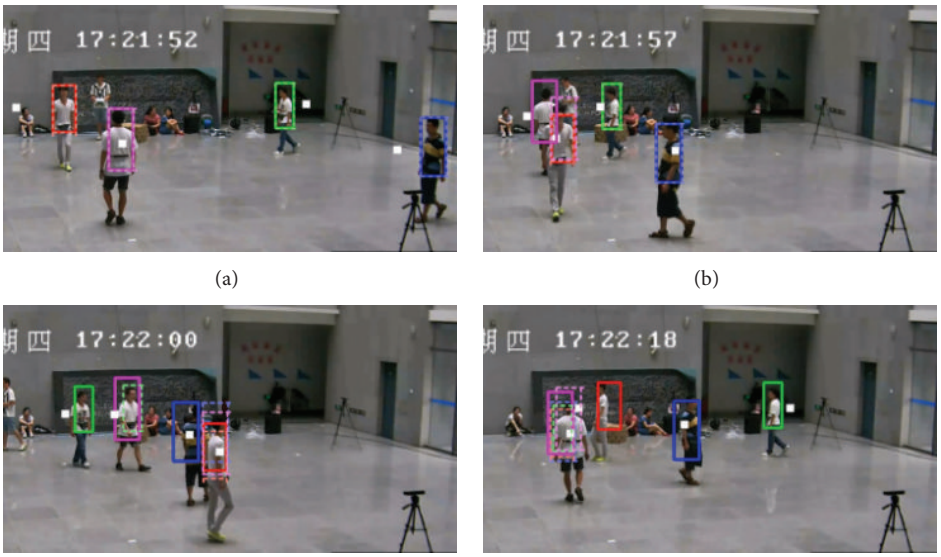
Time duration (s)	Average deviation before correction (m)	Average deviation after correction (m)
20	0.591	0.538
40	1.332	0.849
60	2.206	1.361
80	2.443	1.842
100	2.713	1.777
120	4.329	1.639
140	5.935	1.573
160	8.706	1.650
180	13.622	2.034
200	17.120	2.119

In the second experiment, the visual tracking performance was tested by fusing the inertial sensor positioning when occlusions were encountered. This experiment was performed in the lobby of a building with dimensions of 15 m × 12 m. The layout of the experimental scenario is shown in Figure 10. Four high-definition cameras and six Kinect cameras were located around the center and their corresponding location values are provided in parentheses. This experiment was first performed for a simple scenario in which two persons walked facing each other, met, and then returned to their original positions (Figure 11). Using the visual tracking method alone could have shifted the tracking box to the same target. However, we obtained an accurate tracking result by fusing the inertial sensor positioning with visual tracking. The experiment was then performed for a complex scenario. In this scenario, four persons stood at different locations and then walked along specified routes. Many occlusions occurred during the walking periods (see Figure 12). The target is often lost using visual tracking when occlusions occur. However, the inertial sensor positioning and visual tracking fusion method yielded stable tracking results because of the error detection and correction scheme of the developed tracking method. Figure 13 shows that the developed method detected the occurrence of tracking errors and quickly corrected the positions of the target tracking boxes.

In the third experiment, the visual tracking performance was tested by fusing the inertial sensor positioning when the illumination changed. This experiment was performed for an indoor scene in a room with dimensions of 10 m × 8 m. The illumination was changed by either switching the lights in the room on or off. Illumination changes often cause the visual tracking method to lose the tracked target. The developed inertial sensor positioning and visual tracking fusion method detected and corrected the position of the target tracking box even for severe illumination changes (Figure 14).

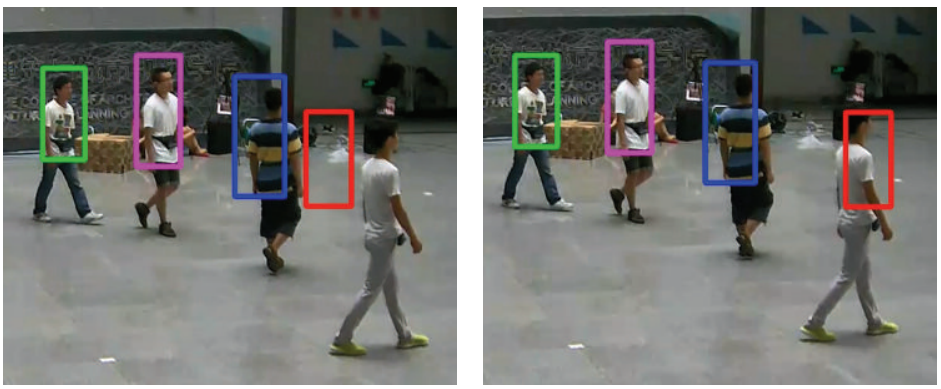
5. Conclusion

In this study, we developed a multisource heterogeneous data fusion method for pedestrian tracking. In this method, the sensing data of a depth camera were used to eliminate the cumulative positioning error of the inertial sensor, thereby making the application of this active tracking technique capable of long-term target tracking. We introduced this improved tracking method into traditional visual tracking to overcome the challenges of object occlusion and illumination



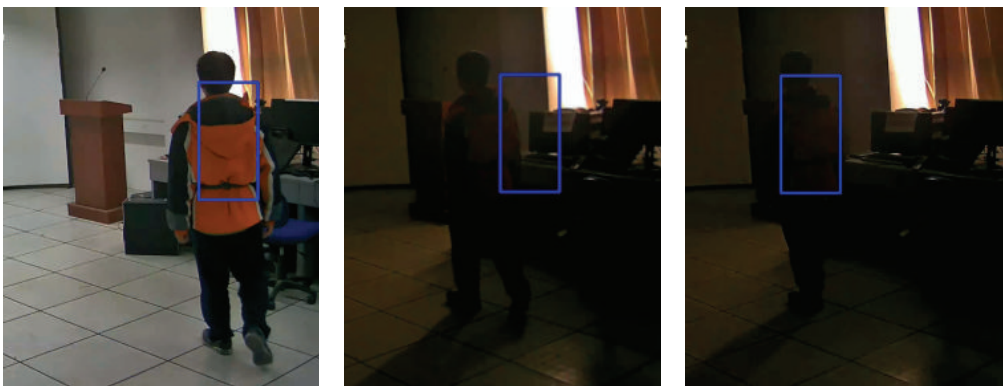
(a) (b) (c) (d)

Figure 12: Comparison of visual tracking method and inertial sensor positioning fusion method for multiperson occlusion: white dots show positions of inertial sensor positioning results projected onto image plane; boxes outlined in different colors show different target tracking boxes; dashed boxes are results obtained using only visual tracking method; and solid boxes are results obtained from fusion of inertial sensor positioning and visual tracking.



(a) (b)

Figure 13: Error detection and correction using inertial sensor positioning and visual tracking methods: (a) error incurred in visual tracking and (b) error detection and correction using fusion tracking method.



(a) (b) (c)

Figure 14: Inertial sensor positioning fusion tracking under illumination changes.

changes. Experiments were performed for different scenarios and showing that the developed method exhibited good tracking performance under critical conditions. The developed tracking method has many potential applications, such as surveillance of the elderly, children, and other specialized groups. In future studies, we will further improve various aspects of the developed method by optimizing the assignment of different sensors, increasing the accuracy of the calibration method for the sensors, and developing a fast training algorithm for large heterogeneous data. The application of the developed method to more practical fields is also a future research subject.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. 61133003, 61370119, and 61171169), Beijing Nature Science Foundation (nos. 4132013, KZ201310005006), and Beijing Science and Technology Project (no. Z141100006014032).

References

[1] S.-K. Weng, C.-M. Kuo, and S.-K. Tu, "Video object tracking using adaptive Kalman filter," *Journal of Visual Communication and Image Representation*, vol. 17, no. 6, pp. 1190–1208, 2006.

[2] Y. Song, Q. Li, and F. Sun, "Shannon entropy-based adaptive fusion particle filter for visual tracking," in *Proceedings of the Chinese Conference on Pattern Recognition (CCPR '09)*, pp. 455–459, IEEE, November 2009.

[3] H. Li, C. Shen, and Q. Shi, "Real-time visual tracking using compressive sensing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 1305–1312, Providence, RI, USA, June 2011.

[4] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2259–2272, 2011.

[5] P. Pérez, J. Vermaak, and A. Blake, "Data fusion for visual tracking with particles," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 495–513, 2004.

[6] H. Zhou, M. Taj, and A. Cavallaro, "Target detection and tracking with heterogeneous sensors," *IEEE Journal on Selected Topics in Signal Processing*, vol. 2, no. 4, pp. 503–513, 2008.

[7] J. Ros and K. Mekhnacha, "A generative model for 3D range sensors in the Bayesian Occupancy filter framework: application for fusion in smart home monitoring," in *Proceedings of the 13th Conference on Information Fusion (FUSION '10)*, IEEE, July 2010.

[8] B.-D. Kang, K.-H. Jeon, D. Kyoung, S.-H. Kim, and J.-H. Hwang, "Multiple human body tracking using the fusion of CCD and thermal image sensor," in *Proceedings of the IEEE Applied Imagery Pattern Recognition Workshop (AIPR '11)*, pp. 1–4, IEEE, Washington, DC, USA, October 2011.

[9] F. L. Bookstein, "Principal warps: thin-plate splines and the decomposition of deformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 6, pp. 567–585, 1989.

[10] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Computer Vision—ECCV 2012*, vol. 7574 of *Lecture Notes in Computer Science*, pp. 864–877, Springer, Berlin, Germany, 2012.