Spring 2017

Fusion of non-visual and visual sensors for human tracking

Wenchao Jiang

FUSION OF NON-VISUAL AND VISUAL SENSORS FOR HUMAN TRACKING

by

WENCHAO JIANG

A DISSERTATION

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

2017

Approved by

Dr. Zhaozheng Yin, Advisor
Dr. Wei Jiang
Dr. Dan Lin
Dr. Maggie Cheng
Dr. Ruwen Qin

ABSTRACT

Human tracking is an extensively researched yet still challenging area in the Computer Vision field, with a wide range of applications such as surveillance and healthcare. People may not be successfully tracked with merely the visual information in challenging cases such as long-term occlusion. Thus, we propose to combine information from other sensors with the surveillance cameras to persistently localize and track humans, which is becoming more promising with the pervasiveness of mobile devices such as cellphones, smart watches and smart glasses embedded with all kinds of sensors including accelerometers, gyroscopes, magnetometers, GPS, WiFi modules and so on. In this thesis, we firstly investigate the application of Inertial Measurement Unit (IMU) from mobile devices to human activity recognition and human tracking, we then develop novel persistent human tracking and indoor localization algorithms by the fusion of non-visual sensors and visual sensors, which not only overcomes the occlusion challenge in visual tracking, but also alleviates the calibration and drift problems in IMU tracking.

ACKNOWLEDGMENTS

Firstly, I would express my greatest gratitude to my advisor, Dr. Zhaozheng Yin, for his meticulous guidance, patient advice and continuous encouragement throughout my entire PhD career. His expertise, sincere and valuable guidance and encouragement always enlightens my path to success in PhD studies. Without his help, I could never reach the accomplishment I have so far.

I am also grateful to all the professors in Missouri University of Science and Technology, especially my committee member, Dr. Wei Jiang, Dr. Dan Lin, Dr. Maggie Cheng and Dr. Ruwen Qin. It is their help and support that led me into the field of Computer Science.

My sincere thankfulness also goes to my colleagues in Dr. Zhaozheng Yin's research team, including Mingzhong Li, Yunxiang Mao and Haohan Li. They are my best friends in my PhD life, my comrades in arms who accompanied me in the last 4 years of study.

Last but not the least, a deep sense of gratitude goes to my family for their endless support for my endeavors on the road to pursue my dreams.

TABLE OF CONTENTS

Page

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

Human tracking, including long-term tracking and instant localization, attracts a lot of attention because consistently tracking and accurately localizing people have many applications in the field of surveillance. However, human tracking is still tricky and challenging because the visual information may be in low quality (e.g., low illumination, low resolution) or sometimes we totally lose the visual signal. For instance, indoor localization system may be cheated by the unified decoration of a building and reports wrong positions to users. Vision-based tracking system may be unable to consistently track the target when the target is totally occluded for long time and changes its direction meanwhile.

These problems are difficult to be solved merely by visual signal, but we can combine information from other sensors to mitigate the human tracking problems. These non-visual sensors include accelerometers, gyroscopes, magnetometers, WiFi modules and so on, which are pervasively embedded in portable devices such as smartphones. For example, in the indoor localization system, the WiFi module is able to decrease the searching space considering the WiFi strength can be regarded as a useful feature to discriminate different positions. In tracking, the magnetometer can be used to estimate the target's proceeding direction when the target is totally occluded. In this paper, we are interested in the fusion of visual and non-visual signal for long-term human tracking and instant localization. More specifically, this paper is composed of four subtopics. The first two subtopics solve problems merely by non-visual sensors, which serves as the base for the last two subtopics, the true fusion of visual and non-visual information for human tracking.

(1). Wearable sensors for human activity recognition (Section 2): We firstly investigate human physical activity recognition merely based on wearable sensors (i.e., ac-

celerometers, gyroscopes). Human physical activity recognition based on wearable sensors has applications relevant to our daily life such as healthcare. How to achieve high recognition accuracy with low computational cost is an important issue in the ubiquitous computing. Rather than exploring handcrafted features from time-series sensor signals, we assemble signal sequences of accelerometers and gyroscopes into a novel activity image, which enables Deep Convolutional Neural Networks (DCNN) to automatically learn the optimal features from the activity image for the activity recognition task. The proposed approach is evaluated in five publicly available datasets and it outperforms five state-of-the-art methods on a wide variety of activity categories.

(2). Wearable sensors for people tracking (Section 3): Then we study human tracking with wearable sensors such as Inertial Measurement Unit (IMU), which is of great significance for ubiquitous computing and ambient applications. We propose a novel Dead Reckoning based tracking algorithm using IMUs placed in the pocket. The contribution of our approach lies in three-folds: 1. Precise steps are detected according to people's repetitive moving patterns. 2. In each step, heading direction is estimated by the principle frequency of filtered acceleration. 3. Rather than inferring the heading direction of each step independently, we compute a vector in the IMU coordinates which can be transformed to the world coordinates to represent the heading direction, by solving an optimization problem with all historical tracking data considered. The proposed tracking algorithm is tested on a public dataset and outperforms five state-of-the-arts. We also apply it to real scenarios where our IMU tracking algorithm successfully assists visual tracking to overcome the challenging visual occlusion problems.

(3). Combing visual and IMU sensors for people tracking (Section 4): We attack the problem of persistently tracking cooperative people such as children, the elderly or patients by combining passive tracking and active tracking techniques. Passive tracking

uses visual signals from surveillance cameras, but vision based people tracking becomes a hard problem in challenging scenarios such as long-term/heavy occlusion, people changing their movement patterns during occlusion, or people temporarily moving out of the visual field. Active tracking uses sensor signals from Inertial Measurement Unit (IMU) carried by targets themselves. IMU-based tracking is independent of visual signals, so it keeps working when people are visually occluded and offers clues where the target could be, helping the visual tracking to reidentify the target. Meanwhile, when visual signals on people are available, visual tracking can calibrate IMU-based tracking to avoid sensor drift. The experimental results show that the IMU and visual tracking are complementary to each other and their combination performs robustly on tracking cooperative people in many challenging scenarios.

(4). Indoor localization by signal fusion (Section 5): Indoor localization based on image matching faces the challenges of clustering large amounts of images to build a reference database, costly query when the database is large and indistinctive image features in buildings with unified decoration style. We propose a novel indoor localization algorithm using smartphones where WiFi, orientation and visual signals are fused together to improve the localization performance. The reference database is built as a signal tree with less computational cost as WiFi and orientation signals pre-cluster the reference images. During localization, WiFi and orientation signals not only offer more context information, but also prune impossible reference images, improving the accuracy and efficiency of image matching. In addition, images are described by multiple-level descriptors recording both global and local image information. The proposed method is compared with other methods in terms of localization accuracy, localization efficiency and time cost to build the reference database. Experimental results on four large university buildings show that our algorithm is efficient and accurate for indoor localization.

2. WEARABLE SENSORS FOR HUMAN ACTIVITY RECOGNITION

2.1. RELATED WORK

Human physical activity recognition has emerged as a critical problem to human-computer interaction, robot learning and ubiquitous computing [1, 2, 3]. When people carry portable devices (e.g., smart phones, smart watches), their physical activities, defined by bodily states such as walking, standing, opening doors, can be recognized based on sensors such as accelerometers embedded in these portable devices.

The critical factor attributed to the success of accelerometer-based activity recognition is to seek an effective representation of the time-series accelerometry signal. The most widely used approaches fall into two categories: handcrafted feature design and automatic feature learning. It is intuitive to manually pick statistical attributes (e.g., means) or quantity distributions (e.g., magnitude histograms) from accelerometry signals, mostly driven by the domain knowledge, prior experience and experimental validation. For example, [4] designed as many as 341 features from 3-axis accelerometry signals while [5] preserved the statistical characteristics of accelerometry data using their empirical cumulative distributions. However, it is hard to extract all useful features in a hand-crafted manner. In addition, a pre-defined feature extraction mechanism trained on a specific scenario might not work well on other scenarios with different sets of activities to be recognized, i.e., the hand-crafted features might not be transferrable to new application domains.

The drawbacks of handcrafted features motivate researchers to explore automatic feature learning [6]. *Deep Convolutional Neural Network* (DCNN), as one of the most effective deep learning models, attracts attentions in the mobile sensing domain considering it has achieved the superior performance in other research fields such as computer vision [7]

and speech recognition [8]. To improve the accuracy of sensor-based activity recognition, [9] designed a tri-source DCNN architecture with the three inputs corresponding to the tri-axis accelerometry data. [10] and [11] took as input the two-dimensional matrix obtained by simply stacking accelerometry signals. [12] looked into this problem in a practical way and showed the application of deep learning to mobile sensing domain is hardware-efficient and can scale to a large number of inference classes.

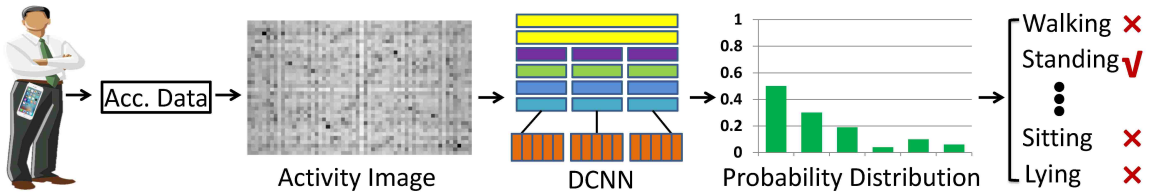

Figure 2.1. Overview of recognizing query signals.

In short, the input to the deep model and the architecture of the deep model itself are two key factors to the success of automatic feature learning. In the previous work, the accelerometry signals are directly fed into the DCNN architecture and this simple input restricts the depth of the deep model, limiting the capability to find discriminative features. For instance, the input in [13] is a small 3×30 matrix and there are only two convolutional layers in the architecture. Additionally, the tri-axis accelerometry signals are convolved with one-dimensional kernels in the deep model independently, thus the correlation among different signals is not taken into enough consideration.

2.2. OUR PROPOSAL

We propose to fuse the tri-axis accelerometry data and explore the hidden relationship between different axes of signals, for the purpose of which, we design a more complex input and a deeper DCNN architecture. A novel 36×60 *activity image* is designed to be

the Fourier transformation of the stacked signals. Taking as the input the low frequency part, the high frequency part and the whole of the activity image, we design a multi-source DCNN architecture with as many as 16 convolutional layers, largely enhancing the complexity and depth of the deep model. Figure 2.1 shows the flowchart to recognize query signals. With the activity image generated from accelerometry data as the input, DCNN outputs a probability distribution over activity categories, based on which the human activity is determined.

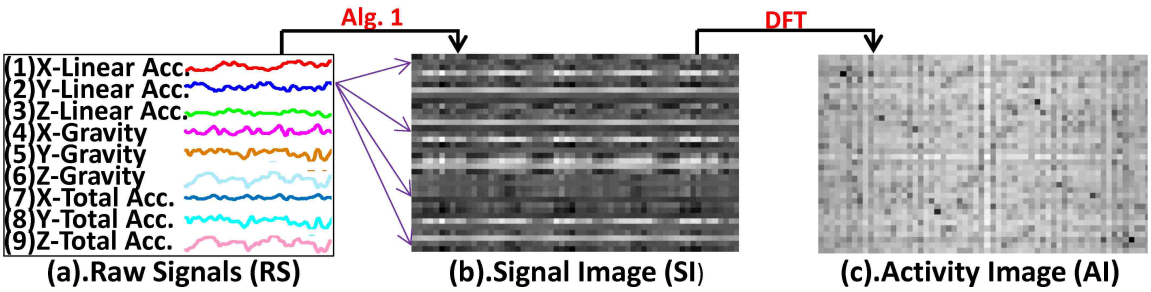

Figure 2.2. Flowchart to generate an activity image.

The section is organized as follows. Subsection 2.2 and 2.3 introduce the activity image and DCNN architecture, respectively. Experimental results on five public datasets are described in subsection 2.4, including comparison with five state-of-the-art methods, and the evaluation of the activity image and the deep architecture.

2.3. ACTIVITY IMAGE

The original accelerometry data from smart devices measure acceleration forces. Such forces may be static such as the continuous force of gravity, or dynamic to sense the movement or vibration of the device (linear acceleration). To make full analysis of the acceleration, we separate the accelerometry data into the low frequency component and

high frequency component, corresponding to the static and dynamic forces which measure the gravity and body motion, separately. Although the gravity is a relatively constant vector, the partition of which into three axes is able to indicate the static direction of the accelerometer while the body motion shows the moving trend of the user. With the separation of accelerometry data, we extend the tri-axis signals to 9 different accelerometry signals (Figure 2.2(a)), which is beneficial to explore the relationship among the acceleration signals. The 9 acceleration signals shown in Figure 2.2(a) are identified by its serial number (e.g., the 5th signal is Y-Gravity).

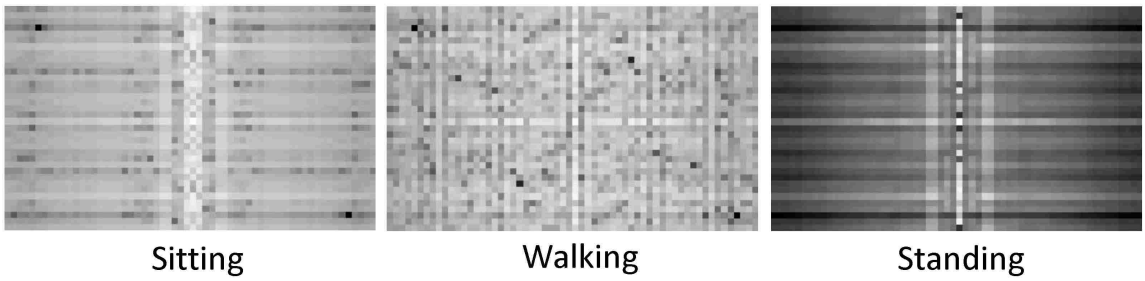

Figure 2.3. Samples of activity images.

As shown in Figure 2.2 (b), a novel *signal image* is established following Algorithm 1. There are two critical rules in Algorithm 1: **(a)** The linear acceleration, gravity and total acceleration signals are stacked row-by-row into the signal image. Algorithm 1 produces a sequence order (the sentence between asterisk lines in Algorithm 1) which tells the signal order in the signal image. For example, signal 2 (i.e., Y-axis linear acceleration signal) is in the 2nd, 15th, 25th and 35th (colored as red) of the sequence order. Correspondingly, Y-axis linear acceleration signal is in the 2nd, 15th, 25th and 35th rows of the signal image, to make which clearer, we display the position of Y-axis linear acceleration in the signal image by the purple arrows in Figure 2.2; **(b)** Algorithm 1 ensures that each signal has one and only one chance to be adjacent to each of the other signals. For instance, in the

sequence order, signal 2 (colored as red) has one and only one opportunity to be adjacent to each of all the other eight acceleration signals (colored as green), similarly do the other signals. This mutual adjacency enables signals to be fully fused with each other.

Finally, 2D Discrete Fourier Transform (DFT) is applied to the signal image and its magnitude is chosen as our *activity image* (Figure 2.2 (c)). Figure 2.3 shows a few activity images corresponding to different activities. It is easy to visually observe their differences. Compared with the signal image (Figure 2.2 (b)) which is a set of permutated accelerometry signals, each pixel in the activity image (Figure 2.2 (c)) clearly indicates the frequency magnitude at that position and the difference between the activity images for various activities can be identified in the pixel level, which enhances the potential for DCNN to extract discriminative features for the activity recognition.

Before introducing the DCNN architecture, we point out some engineering details. Given 9 signals, the size of the signal image is $37 \times L_s$ according to Algorithm 1, where L_s is the time length of signal sequences. Since different devices may have different sampling rates, we interpolate all raw signals in the same time length (e.g., 1 second) with 60 signal samples. Note that, $L_s = 60$ is an acceptable value considering most accelerometers collect signals using the sampling rate between 50HZ and 100HZ. In addition, we delete the last row of the signal image generated by Algorithm 1, so each signal occurs equally 4 times in the signal image. The size of the signal image and the consequent activity image is finalized as 36×60 pixels.

2.4. DCNN ARCHITECTURE

One direct approach to design the DCNN architecture is to take the activity image as the input and build a single-source DCNN, but this approach treats every part of the activity image equally. To emphasize the local context of the activity image, we introduce a multi-

Algorithm 1 Raw Signals (Figure 2.2(a))→Signal Image (Figure 2.2(b)).

Notations:

- Raw Signals (Figure 2.2(a)): the 9 signal sequences shown in Figure 2.2(a). Each signal is identified by its serial number. *//e.g., The 5th signal is Y-Gravity.*
- Signal Image (Figure 2.2(b)): a 2D array stacked by raw signals in a row-by-row manner.
- "the i-th and j-th signal are adjacent in the signal image": in the signal image, there exist two adjacent rows with one row being the i-th signal and the other one being the j-th signal.

Intput:

Raw signals, the number of which is N_s. *//In Figure 2.2(a),* $N_s = 9$.

Loops:

$i = 1; j = i + 1;$
Signal image is initialized to be the i-th signal;
while $i \neq j$ **do**
  if $j > N_s$ **then**
    $j = 1;$
  else if the i-th and j-th signal are **NOT** adjacent in the signal image **then**
    Append the j-th signal to the bottom of the signal image;
    $i = j; j = i + 1;$
  else
    $j = j + 1;$
  end if
end while

Output: Signal Image.

****

The final signal image has 37 rows. From top to down, the signal order is 123456789135792468147158259369483726 1.

****

source DCNN architecture such that multiple sources of information generated from the activity image are taken as the input and they are integrated at a certain level of the deep network. Considering the activity image is actually a frequency map, as shown in Figure 2.4, we divide the activity image into two components: the high frequency part and the low frequency part. Note that the two parts are with the same size and the high frequency image is stitched by the left and right high frequency parts. The whole activity image is utilized

to model human activity globally, while the low frequency part and high frequency part are used for activity recognition in the meticulous level. The three inputs are integrated into the multi-source deep learning framework for activity recognition, and the three modelings are jointly optimized.

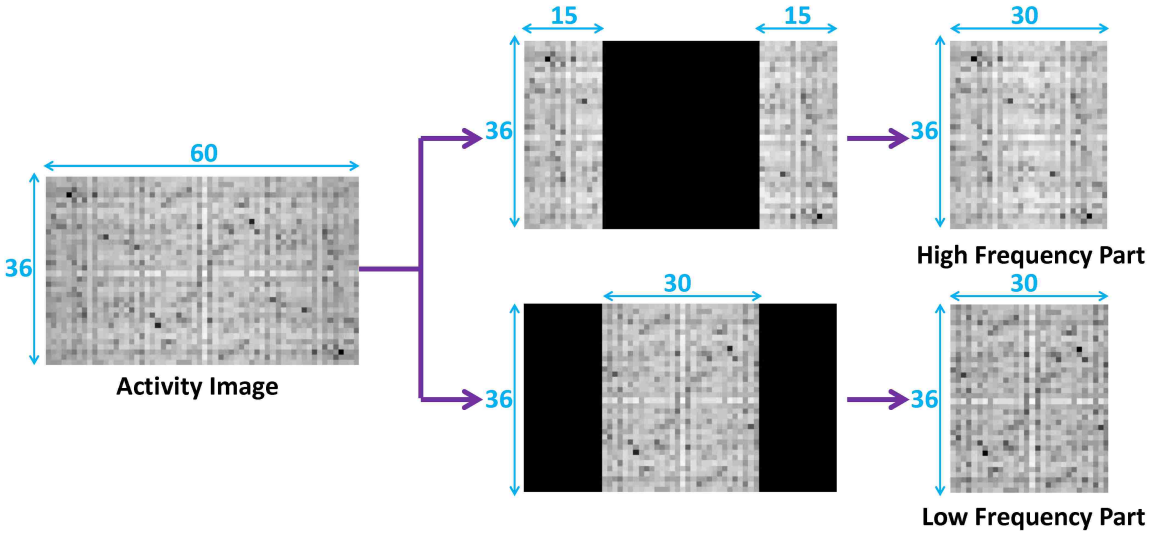

Figure 2.4. Frequency separation of activity image.

Thanks to the larger and more complex input images, we are able to design a deeper DCNN architecture for automatic feature learning. Figure 2.5 shows the multi-source network structure of our deep model. The network contains three branches where the high and low frequency components have the same structure. The network is 25 layers deep when counting only layers with parameters. Deeper depth typically means a larger number of parameters, which makes the network more prone to overfitting, thus we add the dropout layer to each of the full connected layers [14]. We use shorthand notations to represent parameters in the networks: (1) Conv(K@$M\times N\times T$ + S) for convolutional layers with K outputs, T inputs, kernel size $M\times N$ and stride size S, (2) MaxPool($M \times N$ + S) for max pooling layers with kernel size $M\times N$ and stride size S, (3) Dropout(K) for dropout layers

with the dropout rate K. Note that the full connected layer can be regarded as a special convolutional layer, thus we denote the full connected layer with the same format as the convolutional layer. In the testing phase, the softmax operator is adopted to derive the probability distribution over activity classes. In the training phase, the loss function of the network is the softmax log-loss as in Equation 2.1

$$E = \sum_{i}^{I}[(log\sum_{d=1}^{N_a} e^{x_{id}}) - x_{ig}] \tag{2.1}$$

where N_a denotes the number of output neurons, which equals to the quantity of activities to be recognized. i is the index of training samples. Thus x_{id} is the d-th output neuron of the i-th training sample. x_{ig} is the output neuron corresponding to the ground truth of the i-th training sample. The first term of Equation 2.1 ($log\sum_{d=1}^{N_a} e^{x_{id}}$) minimizes all output neurons while the second term (x_{ig}) maximizes the output neuron corresponding to the ground truth, thus Equation 2.1 maximizes the difference between the output neuron corresponding to the ground truth and the other output neurons.

We trained our deep model using the stochastic gradient descent where the gradient is obtained by the back-propagation. We use batches of 50 examples, momentum of 0.9 and weight decay 0.0005. The model is initialized with learning rate of 0.001 and this value is further reduced manually whenever the validation error stops improving.

2.5. EXPERIMENTAL RESULTS

As summarized in Table 5.1, we selected five publicly available datasets for the method validation. These datasets are collected in various contexts including different sensor positions on the human body, different sampling rates, and different numbers of vol-

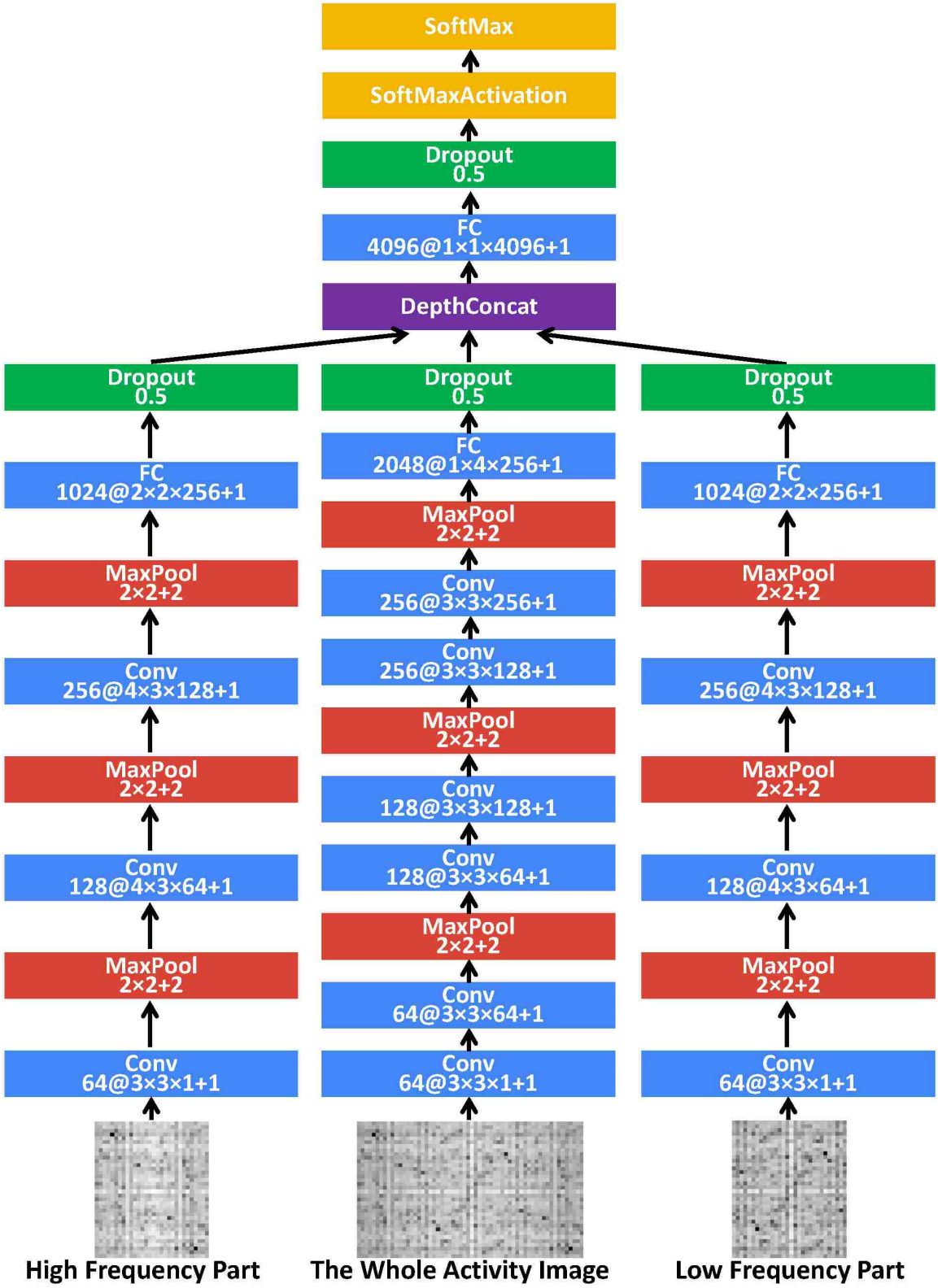

Figure 2.5. The proposed multi-source DCNN architecture. Note that the Rectified Linear Unit (ReLU) layer is not depicted here which follows each convolutional layer and full connected layer.

unteers. In addition, the five datasets include activities with different levels of classification difficulties, for example, the relatively more discriminative activities such as walking, sitting (USC [15], UCI [4], WISDM [16]), and the challenging fine-grained activities such as closing a closet vs. closing a door, cleaning a table vs. moving a cup (Oppor [17]). Complex activities in special scenarios such as the manipulative gestures performed in a car maintenance workshop (Skoda [18]) are also used in our experiments. By leveraging these five different datasets, we want to test the effectiveness and robustness of our activity image and DCNN architecture.

Table 2.1. Dataset information.

Dataset	Number of Activity	Sensor Position	Sampling Rate	Number of Volunteers	Number of Training Samples	Number of Testing Samples
USC[15]	12	Hip	100HZ	14	7675	3307
UCI[4]	6	Waist	50HZ	30	7166	3133
WISDM[16]	6	Pocket	20HZ	36	14453	6134
Skoda[18]	10	Wrist	96HZ	1	3811	1661
Oppor[17]	11	Arm	33HZ	4	5352	2303

2.5.1. Evaluating the Design of Activity Image. The design of activity image can affect the effectiveness of the proposed method. Table 5.2 shows the performance of activity recognition with various designs of input images where *accuracy* is defined as the number of correctly recognized samples divided by the total number of testing samples. The proposed activity image (Row 3) achieves the highest recognition accuracy. The accuracy decreases when we use the signal image directly (Row 1) or replace the Discrete Fourier Transform with the Discrete Cosine Transform (Row 2). The proposed activity image is symmetrical with each pixel explicitly indicating the frequency magnitude at that position. This pixel-level difference among different activity categories facilitates the deep model to learn discriminative features.

Table 2.2. Accuracy vs. input images.

Input Image Design	DCNN Accuracy (%)				
	USC[15]	UCI[4]	WISDM[16]	Skoda[18]	Oppor[17]
RS $\xrightarrow{\text{Alg.1}}$ SI	89.73	97.48	96.76	92.49	71.03
RS $\xrightarrow{\text{Alg.1}}$ SI $\xrightarrow{\text{DCT}}$ AI	91.58	97.76	96.26	92.39	68.08
RS $\xrightarrow{\text{Alg.1}}$ SI $\xrightarrow{\text{DFT}}$ AI (Fig.2)	**92.17**	**98.52**	**97.11**	**94.21**	**71.95**

*Raw Signals (RS); Signal Image (SI); Activity Image (AI).

2.5.2. Evaluating the Design of DCNN Architecture. The design of the deep model is a critical factor to the proposed method. In this section, we design a multi-source deep model which takes the whole activity image and its high and low frequency parts as the inputs. We compare our model with other alternatives that are built by discarding one or more branches. For example, a network only taking the high and low frequency parts can be designed by omitting the branch of the whole activity image.

Table 2.3. Accuracy vs. architecture alternatives.

Dataset	Accuracy (%)						
	H+W+L	W+L	W+H	H+L	H	L	W
USC[15]	92.36	92.70	92.26	91.61	64.67	90.47	92.17
UCI[4]	98.54	98.49	98.46	98.05	61.43	98.22	98.52
WISDM[16]	97.25	96.77	97.47	96.88	74.86	97.36	97.11
Skoda[18]	94.50	94.76	95.17	93.10	47.95	91.34	94.21
Oppor[17]	74.51	74.03	72.26	70.93	44.10	70.41	71.95

*W: the whole activity image; H: the high frequency part. L: the low frequency part.

Table 2.3 shows the recognition accuracy with different model alternatives, from which we have the following observations: (1) The comparison between the first and the last column indicates that the performance can be improved by adding the high and low frequency parts to the deep model with the whole activity image only. (2) The first three columns shows that the performance is comparable when we discarding either the high or low frequency part, because the deep model itself is capable of retuning the parameters

and balancing the branches when the high or low frequency part is discarded. (3) The first and fourth column indicate when the branch of the whole activity image is deleted, the performance significantly decreases, which means the whole activity image is important to improve the recognition accuracy. (4) The large performance gap shown in the fifth and sixth column illustrates that the low frequency part is of much more importance than the high frequency part, because the low frequency part contains more principle information while the high frequency part is more likely to be influenced by noise.

Table 2.4. Performance comparison.

Dataset	Accuracy (%)					
	The Proposed	JBY[13]	MZ[9]	SD[10]	SVM[4]	ECD[5]
USC[15]	**92.36**	88.89	85.74	81.50	91.93	90.81
UCI[4]	**98.54**	97.57	95.16	94.34	96.55	92.08
WISDM[16]	**97.25**	95.91	95.63	93.92	97.06	96.09
Skoda[18]	**94.50**	92.15	91.16	84.28	92.23	86.69
Oppor[17]	**74.51**	66.40	67.09	60.36	72.99	64.74

2.5.3. Quantitative Comparison with State-of-the-Arts. As shown in Table 5.3, we compare the proposed method with five other state-of-the-arts which can be classified into two categories: (1) Automatic feature learning. The three methods labeled by green in Table 5.3 are based on deep convolutional model, which differ with our proposed method on the input image and the DCNN architecture; (2) Handcrafted feature design. The two methods labeled by blue in Table 5.3 extract statistical attributes and empirical cumulative distributions to recognize activities, respectively.

Our proposed method achieves higher accuracy than the other three deep model based approaches, which is attributed to two factors: a more complex input image intensively establishing the relationship between accelerometry signals and a deeper DCNN architecture extracting discriminative features. Our method also outperforms the two ap-

proaches based on handcrafted features. The manual design of features largely relies on researchers' domain knowledge and experiences, thus it is possible to omit important hidden features.

2.6. SUMMARY FOR SECTION 2

In this section, we focus on going deeper to achieve the higher performance on human activity recognition. The word "deep" not only corresponds to the deep learning method adopted in this section, but also has the following meanings. First of all, we go deeper to process the accelerometry signals by separating them into gravity and linear acceleration, then stacking the signals to have them adjacent to each other. DFT further simplifies the difficulty to learn discriminative features. In addition, we design a deeper and more complex network model for feature learning. In the experiments, we not only evaluate various designs of input images and different network alternatives, but also compare our method with state-of-the-arts on five public datasets. The proposed method achieve the superior performance in terms of recognition accuracy.

3. WEARABLE SENSORS FOR PEOPLE TRACKING

3.1. RELATED WORK

This section is revised based on the paper published in IEEE 2015 Global Conference on Signal and Information Processing [19].

Human tracking based on Inertial Measurement Units (IMU) has a wide range of applications in the ubiquitous and wearable computing [20, 21, 22, 23]. IMU-based human tracking (abbr. IMU tracking) is able to achieve a high accuracy in a short or middle term [24, 25]. In addition, because of its low battery consumption, low cost and the advantage of requiring no extra infrastructures, IMU tracking can be applied to bridge the gap in sensor-denied areas [26, 27], such as scenarios without visual, GPS or WiFi signals.

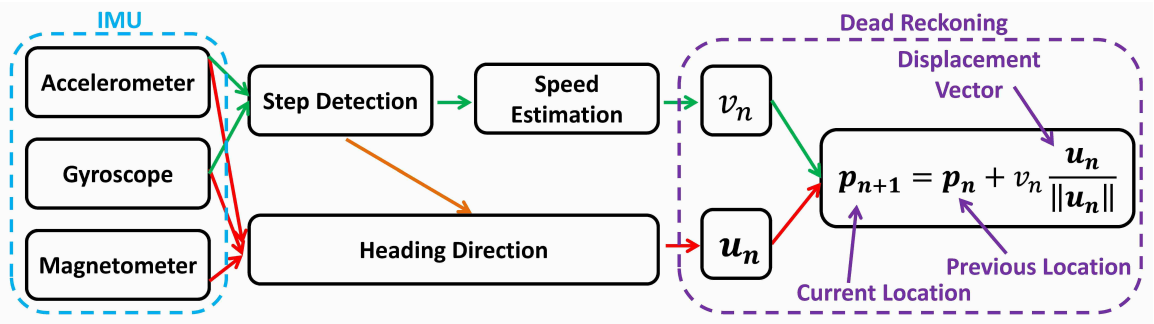

Figure 3.1. Overview of our IMU tracking. In the Dead Reckoning framework, the current location $\mathbf{p_{n+1}}$ is obtained by adding the estimated displacement vector, $v_n \frac{\mathbf{u_n}}{\|\mathbf{u_n}\|}$, to the previously estimated location $\mathbf{p_n}$. v_n and $\mathbf{u_n}$ are the speed and heading direction in step n, respectively.

With IMUs embedded in mobile devices, their carriers can be tracked based on the Dead Reckoning (DR) framework [25, 28, 29, 30, 31, 32], which consists of three components: step detection, speed estimation and heading direction determination. Previ-

ously, steps are detected by setting a threshold on the value of acceleration [28, 30, 32]. However, the threshold is person-specific and is dependant on the speed, so it needs to be pre-determined and manually changed according to a target's speed. Heading direction estimation is another challenging part. Acceleration is often used to determine the forward moving direction [29, 31]. However, when a person walks, the movement pattern of an IMU on the body is complicated, including noise which is unrelated to the heading direction, thus further calculation on the IMU signals is needed to infer the forward moving direction. Overall, DR is subject to cumulative errors if we treat steps individually, so it is necessary to consider the historical tracking data and reach a globally-optimal trajectory.

3.2. OUR PROPOSAL

This section presents a novel Dead Reckoning (DR) based human tracking algorithm using IMUs embedded in wearable devices. Here we concentrate on the scenario when wearable devices are placed in the pockets of people's trousers because it is a common location to carry mobile devices especially for smartphones. Figure 3.1 shows the overview of our IMU tracking. IMU consists of an accelerometer, a gyroscope and a magnetometer, measuring tri-axis acceleration, tr-axis angular velocity and the strength of magnetic field, respectively. In this section, steps are detected using accelerometer and gyroscope by Discrete Fourier Transformation and no person-specific thresholds are needed. Speed is estimated by the maximal difference of angular velocities. Heading direction is obtained from the principle frequency of the filtered acceleration. Additionally, the heading direction of each step will be further rectified by considering historical tracking data. In the rest of the section, we introduce the three components of DR in order, then we summarize our algorithm and present the experimental study.

3.3. STEP DETECTION

Speed and heading direction require to be computed on a complete step period, so the accurate beginning and end of each step are needed. Unlike previous work which relies on person-specific thresholds [28], we detect step in the frequency domain of acceleration and angular velocity using the repetitive moving pattern when people walk. Magnetic field is not suitable for step detection because step detection should be direction invariant.

Figure 3.2(a) shows the magnitude results after applying Discrete Fourier Transformation (DFT) to the six signals of accelerometer and gyroscope over a time sliding window L of 400 samples (4 seconds). As we know, the frequency components related to the step number should have high magnitude. We compute the principle frequency of all six signals, f^*, by

$$f^* = \arg\max_f \sum_{i=1}^{6} |F_i(f; L)| \tag{3.1}$$

where $|F_i(f; L)|$ denotes the magnitude of frequency component f of the ith signal within the time sliding window L. Note DFT only detects integral frequency here, thus f^* is an integer, equalling to 4 in Figure 3.2(a), but if there are 3.8 or 4.3 steps in the sliding window L, the corresponding principle frequency will be rounded to 4. To find the step precisely, we observe that the principle frequency should have a large difference compared to its neighboring frequencies. Therefore, we propose a new metric M_L, the magnitude variance of the principle frequency compared with its neighboring frequencies, to search the accurate steps:

$$M_L = \sum_{i=1}^{6} var([|F_i(f^*\text{-}1; L)|, F_i(f^*; L)|, F_i(f^*\text{+}1; L)|]) \tag{3.2}$$

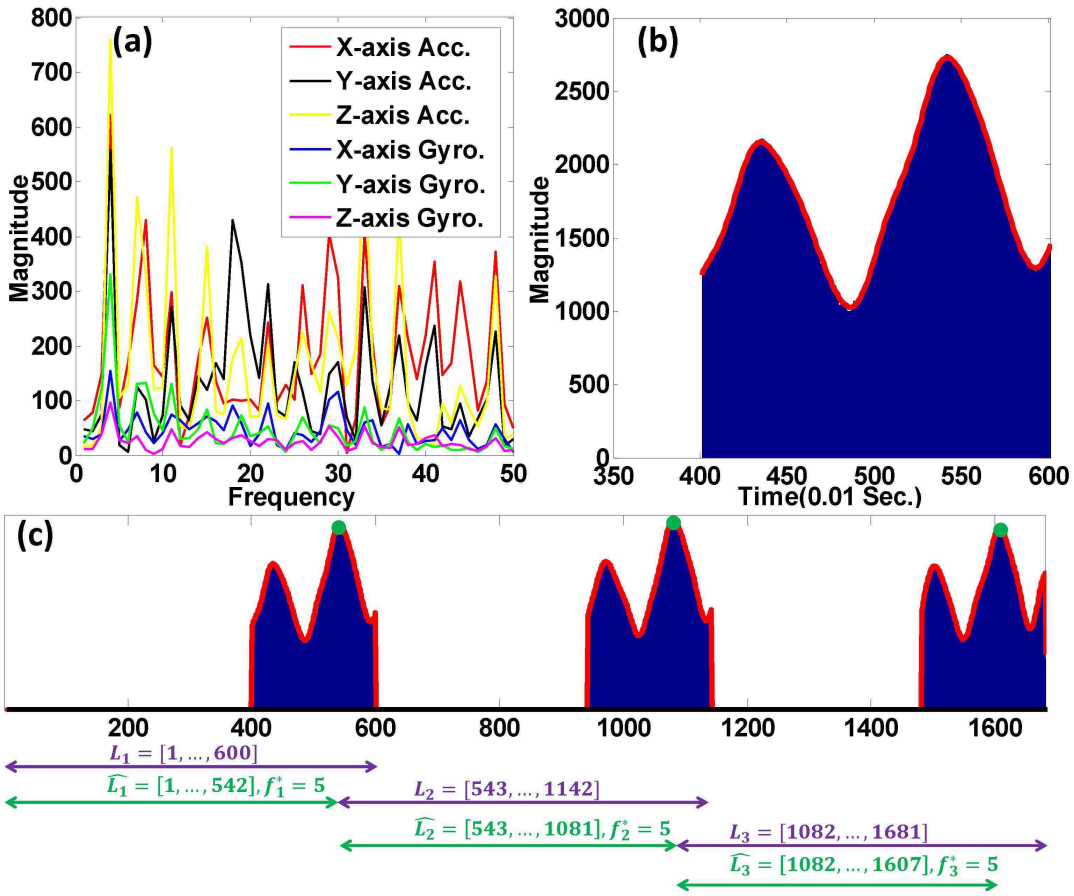

Figure 3.2. Step detection. (a) DFT results of signals with 400 samples. (b) The variance metric M_L vs. the length of time sliding window L. Red curve is the curve fitting result. (c) Illustration of the precise step detection procedure.

The blue part of Figure 3.2(b) shows the plot of M_L when we gradually increase L from 400 samples to 600 samples. The first peak and second peak are around 432 and 540, respectively, which means that there are 4 steps in 432 samples (4.32 seconds) and 5 steps in 540 samples, i.e., each step period is about 108 samples. The peaks in Figure 3.2(b) indicate that at these points, the magnitude of the principle frequency has the locally largest difference compared to its neighboring frequencies. The raw curve of M_L may not be smooth, so we fit it by the least square method using the sum of sines model (i.e., $M_L = \sum_j a_j sin(b_j s + c_j)$, where a_j, b_j and c_j are parameters to be estimated. s is the

sample sequence number). The red curve in Figure 3.2(b) is the curve fitting result whose local maximum can be detected according to the parametric formula.

Figure 3.2(c) illustrates how we determine the beginning and end of each step precisely. We firstly define a coarse sliding window L (purple interval, with 600 samples or 6 seconds). For general walking, 6 seconds are able to include several steps. We then compute the M_L metrics in the last 200 samples of interval L. From the peak (green point) in the M_L curve, a precise time sliding window $\hat{L}$ (green interval) can be calculated, which includes exactly f^* steps.

3.4. SPEED ESTIMATION

Integration on acceleration to obtain speed accumulates errors very fast, making it impractical for speed estimation. Observing that intensity of movement is approximately proportional to speed, we propose to use the maximal difference of angular velocity to measure the movement speed. The speed for step n is defined as

$$v_n = K(max_{s\in[s_b^{(n)},\cdots,s_e^{(n)}]} \parallel \mathbf{e_s} \parallel - min_{s\in[s_b^{(n)},\cdots,s_e^{(n)}]} \parallel \mathbf{e_s} \parallel) \tag{3.3}$$

where $s_b^{(n)}$ and $s_e^{(n)}$ denote the beginning and end of the nth step and $\parallel \mathbf{e_s} \parallel$ computes the magnitude of angular velocity at sample s. K is a normalization factor calibrated using a small fraction of the IMU data (about 20 seconds).

3.5. HEADING DIRECTION

The 3D acceleration in the IMU coordinates during a step can be projected to the horizontal plane in the world coordinates. Principle Component Analysis (PCA) is applied to the transformed acceleration to infer the heading direction [25, 28], but this is likely

to be influenced by noise. As shown in Figure 3.3(a), the estimated heading direction by applying PCA onto the projected raw acceleration data is not accurate.

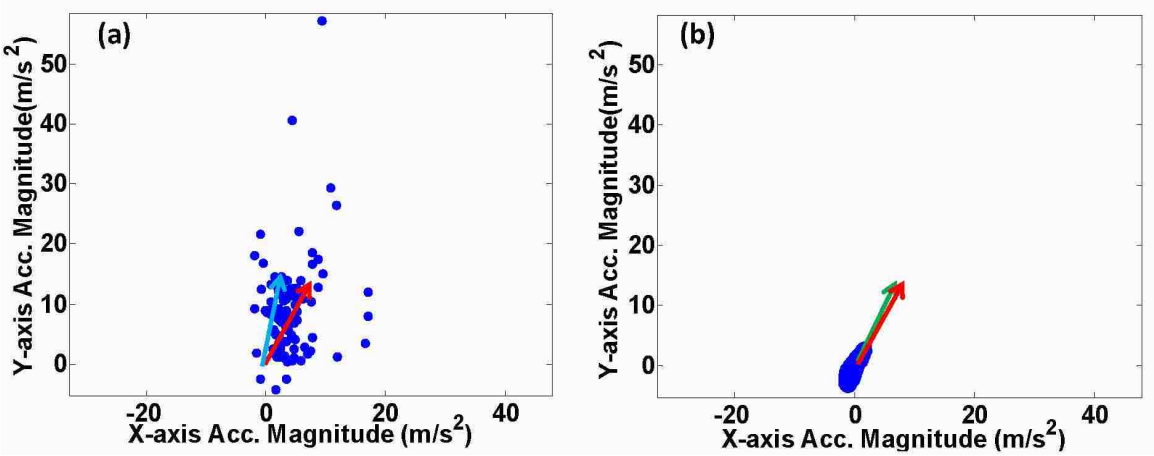

Figure 3.3. Determination of the heading direction. Red arrows are the ground truth. (a) The raw acceleration during a step is projected to the horizontal plane of the world coordinates and the blue arrow is the estimated heading direction, which is inaccurate. (b) The acceleration corresponding to the principle frequency is projected and the green arrow is the estimated heading direction, which is much more accurate.

In this subsection, we transform the time-series acceleration signals in a step into the frequency domain. We consider the frequency with the largest magnitude as the principle frequency and treat all non-principle frequency components as noise and zero out them. Then, we transform the filtered acceleration back to the time domain and project it to the world coordinates' horizontal plane. PCA is applied to the filtered acceleration to find the heading direction $\mathbf{u}_n^{(w)}$ of step n. As shown in Figure 3.3(b), the heading direction is more accurate.

When applying the above procedure (filtering, projection and PCA) repetitively to determine the heading direction for each step individually, small errors may be accumulated over time. We further rectify the heading direction by considering all available steps' heading direction globally. In Figure 3.4, if an IMU follows the forward directions

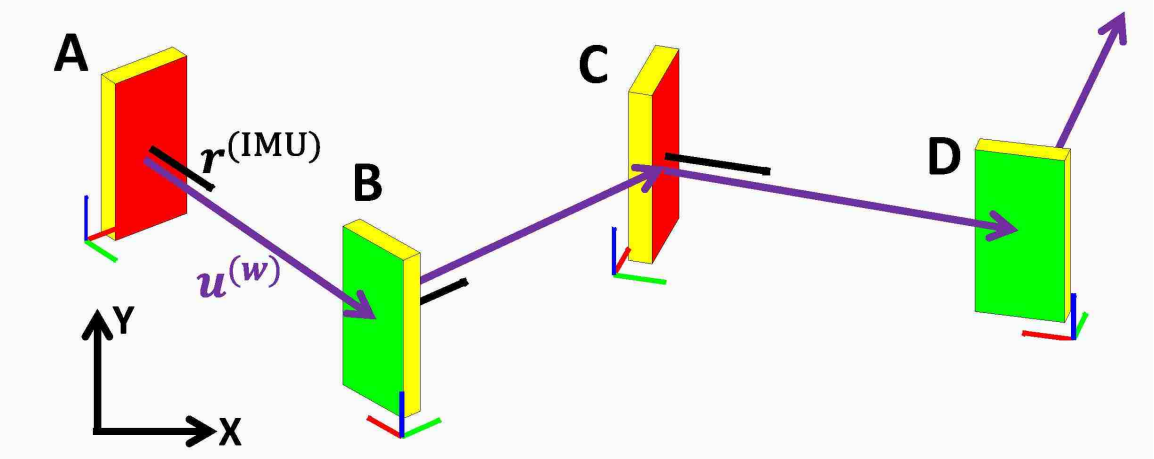

Figure 3.4. A constant in different coordinate system. A constant vector in the IMU coordinates always represents the forward direction after being projected to the world coordinates.

$A \rightarrow B \rightarrow C \rightarrow D$, the black vector that is constant in the IMU coordinates always points to the forward direction after being projected to the world coordinates. Assuming an IMU's position in the target's pocket does not dramatically change, it is possible to find a vector $\mathbf{r}_n^{(IMU)}$ in the IMU coordinates and then project it to the world coordinates to represent the heading direction. $\mathbf{r}_n^{(IMU)}$ is found by solving the following optimization problem

$$\arg\min_{\mathbf{r}_n^{(IMU)}} \sum_{s\in[s_b^{(1)},\cdots,s_e^{(n)}]} \| \mathbf{Q}_s^{(w\rightarrow IMU)} * \mathbf{u}_s^{(w)} - \mathbf{r}_n^{(IMU)} \| \tag{3.4}$$

where $\mathbf{u}_s^{(w)}$ is the heading direction of sample s in the world coordinates directly obtained from the heading direction of the step that sample s belongs to. $\mathbf{Q}_s^{(w\rightarrow IMU)}$ is the coordinate transformation matrix of sample s from the world to the IMU coordinates which may be provided by IMU model or computed by the approach proposed in [33]. $s_b^{(n)}$ and $s_e^{(n)}$ denote the beginning and end sample of step n, respectively. $\mathbf{r}_n^{(IMU)}$ is obtained by solving the optimization problem in Equation 3.4 using the least square method.

Each time we detect a new step n, the optimization problem in Equation 3.4 is solved with all historical tracking data, thus $\mathbf{r}_n^{(IMU)}$ is updated. Then we can rectify $\mathbf{u}_s^{(w)}$ to $\hat{\mathbf{u}}_s^{(w)}$ by Equation 3.5:

$$\hat{\mathbf{u}}_s^{(w)} = \mathbf{Q}_s^{(IMU \to w)} * \mathbf{r}_n^{(IMU)}, \quad s \in [s_b^{(1)}, \cdots, s_e^{(n)}] \tag{3.5}$$

Note that $\mathbf{Q}_s^{(IMU \to w)}$ is the transpose of $\mathbf{Q}_s^{(w \to IMU)}$. The rectified heading direction for each step $\hat{\mathbf{u}}_n^{(w)}$ is the average direction of all samples within this step:

$$\hat{\mathbf{u}}_n^{(w)} = mean_{s \in [s_b^{(n)}, \cdots, s_e^{(n)}]} \hat{\mathbf{u}}_s^{(w)} \tag{3.6}$$

3.6. OUR COMPLETE IMU TRACKING ALGORITHM

Algorithm summarizes the complete IMU tracking algorithm combing the three aforementioned DR components. Row 4 to row 14 presents the updating strategy. Note that algorithm 2 is designed for online tracking, so we update the trajectory $\mathbf{p}$ after each new step detection. Algorithm 2 can be easily revised to an offline version by solving the optimization problem after we obtain all data.

3.7. EXPERIMENTS AND EVALUATION

To validate the effectiveness of the proposed IMU tracking algorithm, we test it in a public dataset. In addition, we also successfully apply the IMU tracking to assist visual tracking in an occlusion-included environment.

Algorithm 2 IMU Tracking Procedure.

Inilization: $n = 0; k = 0; \mathbf{p}_0$ = predefined starting position; *// n is the step sequence number; k is sequence number of sliding window.* $\mathbf{p}$ *is the tracking trajectory.*

Loops:

while the target is walking **do**
  • $k = k + 1$;
  • Determine the precise sliding window $\hat{L}_k$; and its number of steps f_k^*.
  for $i = 1 : f_k^*$ **do**
    • $n = n + 1$;
    • Define the n_{th} step's interval $[s_b^{(n)}, \cdots, s_e^{(n)}]$; *//The interval of each step can be determined by dividing the interval of $\hat{L}_k$ by the principle frequency f_k^*.*
    • Determine the tentative heading direction $\mathbf{u}_n^{(w)}$ of step n by the filtering, projection and PCA procedure;
    • Determine the speed v_n of step n by Equation 4.3;
    • Update $\mathbf{r}_n^{(IMU)}$ by solving the optimization problem Equation 3.4;
    • Calculate the rectified heading direction $\hat{\mathbf{u}}_1^{(w)}$ to $\hat{\mathbf{u}}_n^{(w)}$ by Equation 3.5.
    for $j = 1 : n$ **do**
      $\mathbf{p}_j = \mathbf{p}_{j-1} + \hat{\mathbf{u}}_n^{(w)} \cdot v_n$;
    end for
  end for
end while

Output: $\mathbf{p}$;

3.7.1. Test on a Public Dataset. The dataset is provided by [25]. For each recording, three XSens MTx sensors are used with two being freely placed in the left and right pockets respectively for testing and another one dorsally on the back for ground truth. This dataset includes 23 traces from 8 volunteers wearing trousers ranging from jeans to slacks, thus the pockets are with variable sizes and shapes. All volunteers are required to walk with their normal patterns in two locations (a park and a garden). In the park (Figure 3.5(a), a route with 870m), volunteers first walk anti-clockwise in a rectangular-like shape and then turnaround in the opposite direction to the starting point. In the garden (Figure 3.5(c), a route with 415m), volunteers walk based on a more complex trace. Some volunteers are

required to walk several times with different trousers or at different times. Different factors of human, trousers, traces and time are all considered in this dataset and it provides a total of 30km trace length.

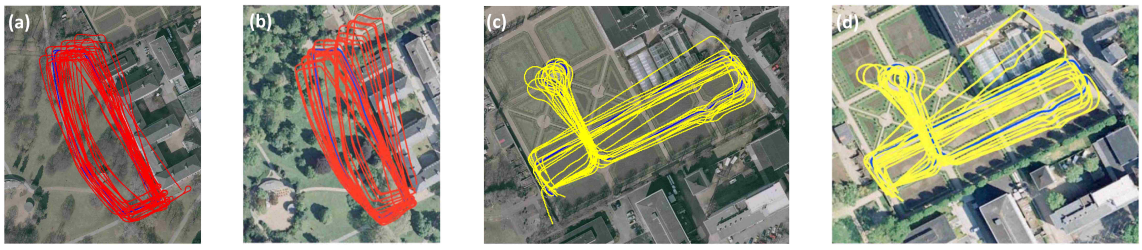

Figure 3.5. Traces comparison. The blue traces are ground truth. Red traces are the traces of all volunteers from the left pocket in the park based on the proposed algorithm (a) or PCA2df [25] (b). Yellow traces are the traces of all volunteers from the right pocket in the garden based on the proposed algorithm (c) or PCA2df [25] (d).

(1). Performance of Step Detection: Table 3.1 shows the performance of step detection by our FFT and adaptive sliding window strategy. Results from column 8 to 11 are obtained from one volunteer wearing four different trousers. The relative error is computed by the difference between the estimated step number and the ground truth divided by the ground truth. The relative error is pretty stable for different volunteers or different trousers. Without setting person-specific thresholds, the overall relative error is 2.45% which is acceptable in real scenarios especially for short-term or mid-term applications.

Table 3.1. Performance of step detection.

Volunteer	1	2	3	4	5	6	7	8a	8b	8c	8d	Total
E. S. N.	1661	2894	1625	1578	1923	1769	1609	1640	1630	1643	1685	19657
G.T.	1636	2842	1568	1528	1871	1733	1568	1601	1597	1597	1645	19186
R.E.(%)	1.53	1.83	3.64	3.27	2.78	2.08	2.61	2.44	2.07	2.88	2.43	2.45

∗Estimated Step Number(E.S.N.); Ground Truth(G.T.); Relative Error(R.E.);

(2). Performance of Heading Direction Estimation: Table 3.2 shows the heading direction estimation performance of the proposed method compared with five state-of-the-arts surveyed in [25]. The rotational approach utilizes the rotational motion of IMU. PCA2D directly projects raw acceleration to the world coordinates while PCA2Df processes the acceleration with a 5HZ mean filter. PCA3Df analyses the eigenvector in the 3D world coordinates. GyroPCA relies on PCA on the angular velocity.

The comparison is performed on the total 30km trajectories from both pockets of all volunteers in terms of two metrics: the deviation per step and the orientation error per step. Among all the approaches, our proposed method achieves the best performance. Figure 3.5 shows the comparison of resulting traces based on the proposed algorithm and PCA2Df [25]. The traces based on our method visually present less deviations than the traces based on PCA2Df [25].

Table 3.2. Performance of heading direction estimation.

	Deviation per step(cm)		Orientation error	
	median	$75\%^{ile}$	median	$75\%^{ile}$
Rotational	31.4	38.9	13.7	15.9
PCA2D	15.1	24.0	7.1	10.3
PCA2Df	12.8	18.2	5.7	7.4
PCA3Df	16.7	23.1	7.7	9.7
gyroPCA	47.6	58.2	21.0	24.4
The Proposed	**9.1**	**14.5**	**3.8**	**6.5**

3.7.2. Practical Application. We conduct a potential application of our IMU tracking in some vision-denied scenarios described in Figure 3.6 where the target is occluded by moving pedestrians at location A and moves out of visual field from location B. At locations C and D, the target is occluded by background objects frequently. The target

is cooperative to carry a Samsung Galaxy III embedded with IMU in his pocket and the tracking-by-detection [34] is adopted as visual tracking algorithm. As we know, visual occlusion is a hard problem for camera-based human tracking [35], but IMU does not rely on vision, thus it has no problem of occlusion. The tracking strategy is as follows: when the target is visible, only visual tracking works. When the target is occluded or missed, the IMU tracking works to re-identify the target. As a result, if only vision is used, it might fail in location A. However, as shown in Figure 3.6, despite the complex scenarios, the IMU tracking algorithm can successfully aids vision to track the target persistently.

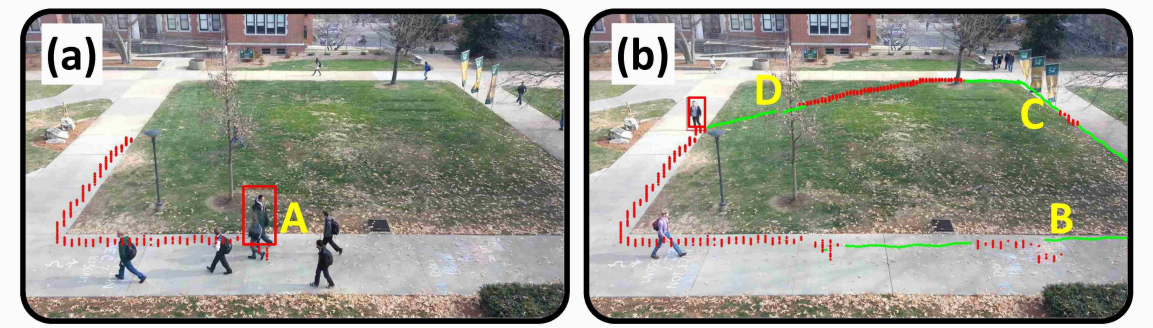

Figure 3.6. An application when the proposed IMU tracking assists visual tracking. Red curves are visual trajectories and green curves are IMU trajectories when visual tracking fails. The two kinds of trajectories form a persistent tracking trajectory.

3.8. SUMMARY FOR SECTION 3

This section presents a novel IMU tracking algorithm based on Dead Reckoning. Steps are detected using accelerometers and gyroscopes by DFT and no person-specific thresholds are needed. Heading direction is obtained from the principle frequency of the filtered acceleration. We not only test it in a large public dataset, but also apply it to a practical application where IMU tracking aids visual tracking to mitigate the challenging visual occlusion problem. The experimental study shows that the proposed IMU tracking is eli-

gible for short-term or mid-term tracking and applicable to practice especially when other sensors are combined. In the future, we will apply the proposed IMU tracking algorithm to more applications such as indoor localization and navigation.

4. COMBING VISUAL AND IMU SENSORS FOR PEOPLE TRACKING

4.1. PROBLEM

This section is revised based on the paper published in 18th International Conference on Information Fusion [36].

People tracking has a wide range of applications such as tracking people in public crowded environments for security surveillance and tracking family members to avoid losing loved ones. Typically, people tracking techniques can be classified as "passive" or "active" tracking. *Passive tracking* utilizes devices that are not carried by people, such as surveillance cameras. *Active tracking* locates targets by sensors carried by the *cooperative* targets themselves, such as the Global Positioning System (GPS), WiFi receiver and Inertial Measurement Unit (IMU). This section attacks the problem of *persistently* tracking cooperative targets (e.g., children, teens, the elderly, patients with autism/alzheimers/dementia) by combining passive and active tracking.

4.2. RELATED WORK

We firstly present the related work for passive visual tracking, then previous work on active sensor tracking is introduced. Previous research on Vision and IMU Fusion will be illustrated finally.

4.2.1. Passive Visual Tracking. Passive vision-based people detection and tracking have been studied for several decades [37, 38]. The challenges are to track people persistently through occlusion or clutter. For partial occlusion, [39] represented humans as an assemble of four body parts and combined body part detectors and human detectors

to track humans when they were occluded. [40] formed the representation of visible and occluded parts and segmented the two parts by graph cuts. [41] trained an occlusion-aware person detector, which was a joint model of detecting single person as well as pairs of persons under varying degrees of occlusion. For full occlusion, previous efforts focused on predicting targets' positions when they are occluded and this was realized by Kalman filter [42, 43, 44] or assuming the target keeps a constant velocity [45, 46].

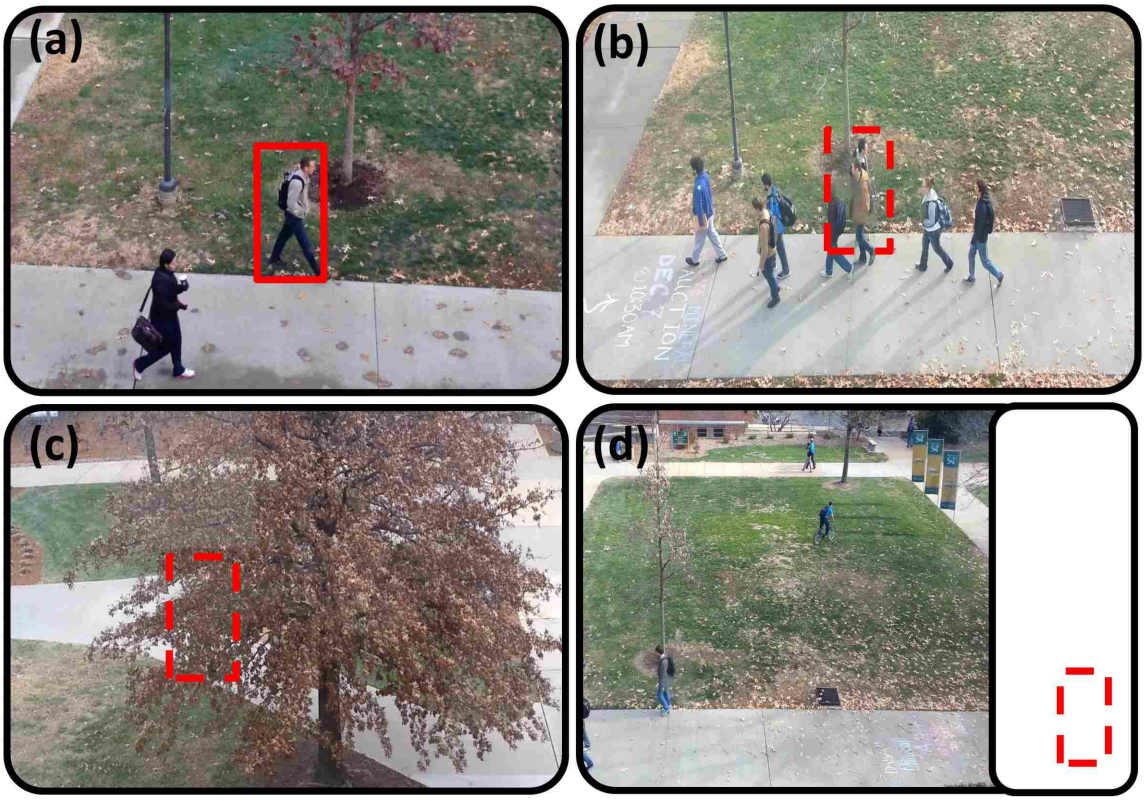

Figure 4.1. Visual people tracking and its challenges. (a) Successful tracking; (b) The target is occluded by other people; (c) The target is occluded by a tree over a long period; (d) The target moves out of the field of view temporarily.

When there is heavy occlusion, large appearance change, nearby clutter or pedestrians temporarily moving out of the field of view, as shown in Figure 4.1, it is challenging

for a merely vision-based tracking algorithm to persistently track people without any failure. This problem becomes worse when people change their moving patterns (e.g., speed, direction) when occluded, which voids the linear filtering based prediction approaches.

4.2.2. Active Sensor Tracking. It is intuitive to track people with GPS considering its wide application in navigation. However, the accuracy of common GPS modules isn't high enough (15 meters on average as reported in http://www8.garmin.com/aboutGPS/). Furthermore, obstructions such as city canyons or tall trees outdoors and walls/ceilings indoors weaken the signals transmitted between GPS receivers and positioning satellites, making the GPS-based tracking unreliable in these GPS-denied environments. WiFi is another choice to locate people but the coverage area of most WiFi hotspots is less than 50 meters, limiting its application in people tracking outdoors.

Inertial Measurement Unit (IMU), consisting of gyroscope, magnetometer and accelerometer, is a good choice to track people by Dead Reckoning (DR) which adds the current displacement vector to the previous estimated location. DR is built upon three components: step detection, speed estimation and forward moving direction determination. Previously, steps are detected by setting a threshold on the value of acceleration [25, 28], but the threshold depends on a person's movement patterns such as running and fast/slow walking. Speed equals to the product of a predefined calibrated coefficient and the amplitude of acceleration [25, 28, 47, 48], but the calibration coefficient is hard-coded and person-dependant. The orientation of acceleration is used to determine the forward moving direction [25, 28, 49], but finding the accurate transformation from sensor movement directions on human body (e.g., in a pocket) to the walking/running direction of a person in the world coordinate is difficult. Furthermore, the DR-based approach is prone to drift if small errors on each step are accumulated over a long period.

4.2.3. Vision and IMU Fusion. Previous work has explored the possibility to fuse vision and IMU for people tracking or navigation [50, 51, 52, 53]. These work usually fixed a camera on the target's body and utilized vision information for motion estimation. By involving the motion information from vision and Dead Reckoning result from IMU into the Kalman filter framework, a more accurate tracking results can be obtained. These work is single-direction fusion, that is only vision can aid IMU for people tracking. Our work sets up the stationary surveillance camera out of the target's body and investigated how IMU and vision tracking can assist each other and form a persistent people tracking system.

4.3. MOTIVATION

Visual tracking can obtain the movement trajectories/patterns of people, thus it can calibrate IMU-based active tracking to avoid sensor drift. It is challenging for visual tracking to handle heavy occlusion, but active people tracking methods have no problems of occlusion because they do not rely on visual signals, thus the occlusion problem of visual tracking can be compensated by active sensor tracking.

4.4. PROPOSAL

Since visual tracking and IMU-based active tracking are complementary, i.e., not only can IMU assist visual tracking when the target is occluded, but also the challenges of IMU tracking (calibration and drift) are alleviated when visual signals are available, we propose a novel people tracking system combining passive visual tracking and active sensor tracking. The visual signal is from stationary surveillance cameras and IMU devices from cooperative people are used for active tracking.

4.5. SYSTEM OVERVIEW

Our cooperative tracking system consists of three parts:

(1). Passive Visual Tracking (Subsection 4.2): Given a stationary surveillance camera, a scene-specific pedestrian detector is trained to improve the detection performance. An adaptive scale selection algorithm is proposed to further improve the pedestrian detection performance and reduce computational cost. Mode-seeking algorithm is applied to the detection confidence map for people tracking.

(2). Active IMU Tracking (Subsection 4.3): A Discrete Fourier Transform (DFT)-based step detection method is proposed, which does not need preset person-dependant thresholds. The calibration coefficient in speed estimation is obtained by visual tracking instead of manual setting. More accurate forward moving direction is obtained by a principle frequency component filter.

(3). Integration of Visual and IMU Tracking (Subsection 4.4): When the target is visible, its visual trajectory calibrates and adjusts its IMU trajectory. When the target is occluded, IMU tracking keeps working and offers clues for visual tracking to re-identify the missed target.

4.6. PASSIVE VISUAL TRACKING

In this subsection, a scene-specific and scale-adaptive pedestrian detector is firstly introduced, then visual people tracking based on detections is described.

4.6.1. Training a Scene-specific Pedestrian Detector. Histogram of Oriented Gradient (HOG) features along with Support Vector Machine (SVM) has been widely used to perform pedestrian detection in images. A large dataset (both positive samples and negative samples) are usually needed to train a general pedestrian detector which is very time-

consuming and the detector may not work well on scenarios different from the training dataset [37].

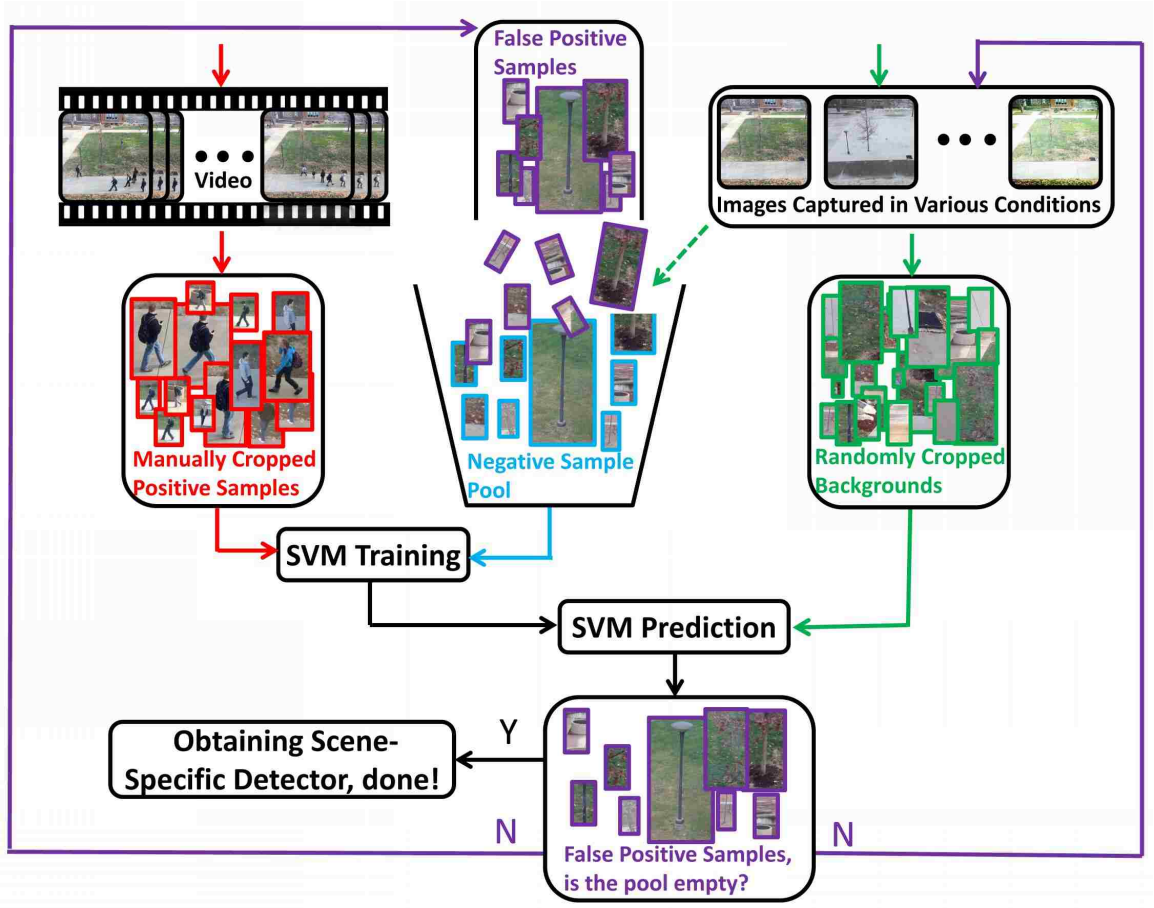

Figure 4.2. Training a scene-specific pedestrian detector.

In a fixed scene, the viewpoints from which people can be observed and the scales of people in images are limited. Moreover, the negative samples are limited (they are just the background in the scene!). The critical problem in people detection is how to classify those background samples that are very likely to be mistakenly classified as people samples. If the detector can correctly classify those background samples whose feature descriptors are near the decision boundary, it is sufficient to classify other background samples which

are largely different from people samples. In this subsection, we propose a new iterative training algorithm to deal with the problem.

As illustrated in Figure 4.2, the positive samples (images framed in red) are the manually cropped pedestrian images from videos taken on the specific scene, which include pedestrian images with different walking gaits and scales that can be seen from the specific viewpoint. The pool of positive samples is not changed during the iterative training. The negative sample pool initially consists of randomly cropped backgrounds from images taken on the specific scene in different weather and illumination conditions (images framed in blue). The negative sample pool expands gradually during the iterative training.

The iterative training algorithm is performed in the following steps: when a new pedestrian detector is available after SVM training, it will be applied to classify background images randomly cropped from images taken on the specific scene (images framed in green); the false positive samples (images framed in purple) are put into the negative sample pool and the SVM will be updated for the next iteration; training stops when the number of false positive samples is zero. Every time the SVM is updated, the detector is more robust to classify those samples that are misclassified previously.

4.6.2. Adaptive Scale Selection. In common people detection algorithms, for every input frame, different scales defined by the height and width of rectangles need to be searched in the image exhaustively to detect all pedestrians. In a fixed scene, although the same person may display different scales at different locations in the image (e.g., Figure 4.3(a)). However, if we transform the image into the top-down view by a homography matrix $\mathbf{H}_a$, the width of the pedestrian rectangle is almost a constant (red lines in Figure 4.3(b) are the warped rectangular bottoms from four detections). Thus, if we fix the ratio of the height and width of a detection rectangle and determine the standard scale S_{std} by the length of the bottom side of the warped rectangle, people's scales in every region of the

specific scene can be estimated in advance, i.e., we know which scale in the original image we should use to detect pedestrians rather than performing the exhaustive scale search.

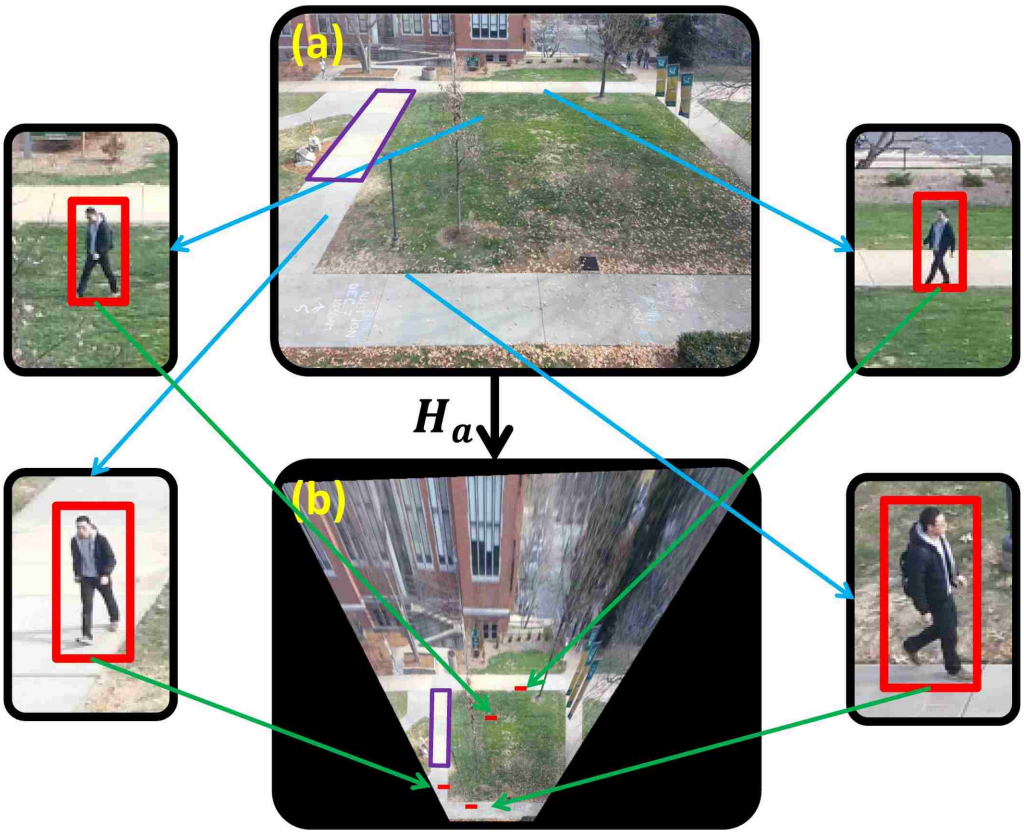

Figure 4.3. Adaptive scale selection.

$\mathbf{H}_a$ is estimated by four pairs of point correspondences (e.g., the four corners of the purple rectangle in Figure 4.3). $\mathbf{H}_a$ is a constant for a fixed scene and it only needs to be updated when the viewpoint changes.

4.6.3. Tracking by Detection. Based on the target's location in the previous frame $t-1$, we apply our scene-specific and scale-adaptive pedestrian detector within a local region around the previous location to detect the target in the current frame t. Figure 4.4 shows some pedestrian images and their confidence maps corresponding to SVM scores of

the pedestrian detector. The white in a confidence map denotes high score (confidence) of people detection. The target's location in frame t is determined by seeking the mode (the position with maximal confidence in the confidence map).

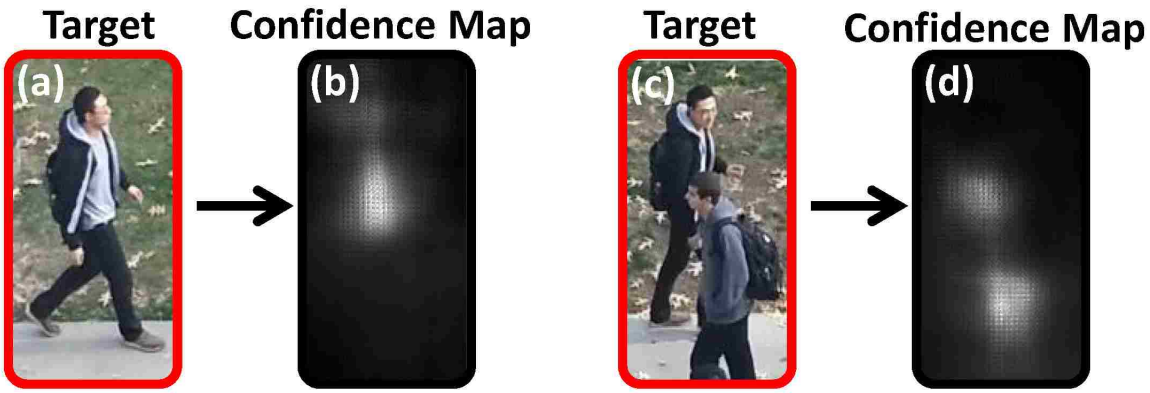

Figure 4.4. Visual tracking. (a) and (c) are the pedestrian images. (b) and (d) are their confidence maps, respectively.

If the target is not occluded (Figure 4.4(a)), there is a single global peak in the confidence map, thus the target can be correctly tracked. However, when the target is occluded by other pedestrians (e.g., Figure 4.4(c)), there are multiple peaks in the corresponding confidence map. It is possible that the non-target pedestrian is detected and tracked mistakenly. Therefore, when occlusion, clutter and disappearance of the target happen, we refer to IMU-based active tracking to correct the visual tracker and reidentify the lost target.

4.7. ACTIVE IMU-BASED TRACKING

IMU includes accelerometer, magnetometer and gyroscope, which measures tri-axis acceleration, the strength of magnetic field and tri-axis angular velocities, respectively. As shown in Figure 4.5, our IMU tracking is based on Dead-Reckoning (DR) which adds the displacement vector, $v_n \frac{\mathbf{u}_n}{\|\mathbf{u}_n\|}$, to the previously estimated location $\mathbf{p}_n$. v_n and $\mathbf{u}_n$ are the

speed and forward moving direction in step n, respectively. Our IMU tracking approach consists of three components: step detection, speed estimation and forward moving direction determination.

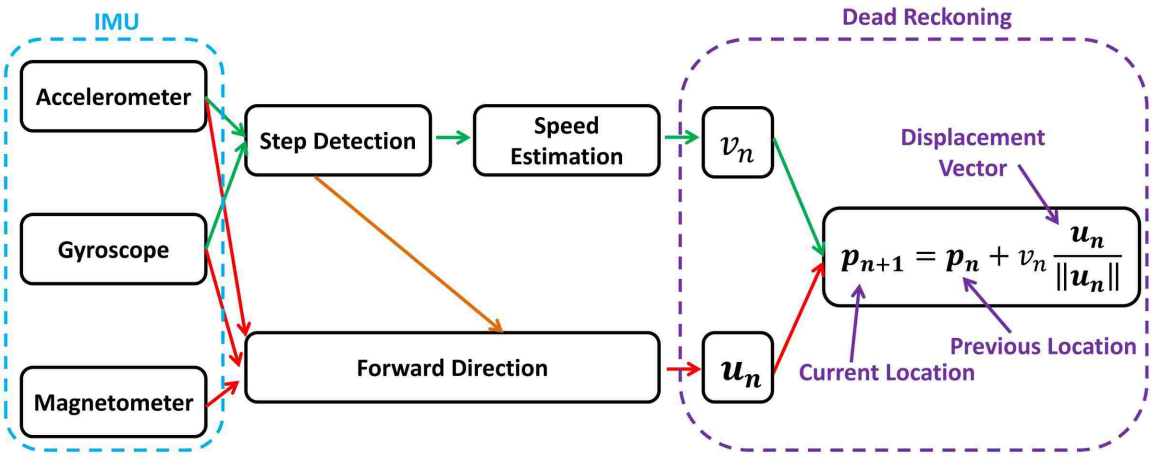

Figure 4.5. Flow chart of our IMU-based pedestrian tracking.

4.7.1. Step Detection. Speed and heading direction require to be estimated on a complete step period for DR, so the accurate beginning and end of each step is needed. In [25], step is detected in the time domain by finding local maximum and minimum of acceleration data and a threshold is set to rule out false positives. However, the threshold depends on the speed and is person-specific. When speed greatly changes, missed detection of steps increases rapidly. Different thresholds need to be chosen for different targets.

In this subsection, a step detection algorithm based on adaptive sliding window and Discrete Fourier Transform (DFT) is introduced, which is inspired by the following observations: (1) The movement pattern of a walking person is periodic. Therefore, DFT can be applied to find the number of periods (i.e., the number of steps) in a certain sliding window. (2) Magnetic field is sensitive to heading direction change, so it is not suitable for step detection. Instead, angular velocity and acceleration are ideal because they do not

depend on the forward direction. (3) Only one axis signal is not reliable for step detection. All 6 axes of gyroscope and accelerometer are considered in our step detection by DFT.

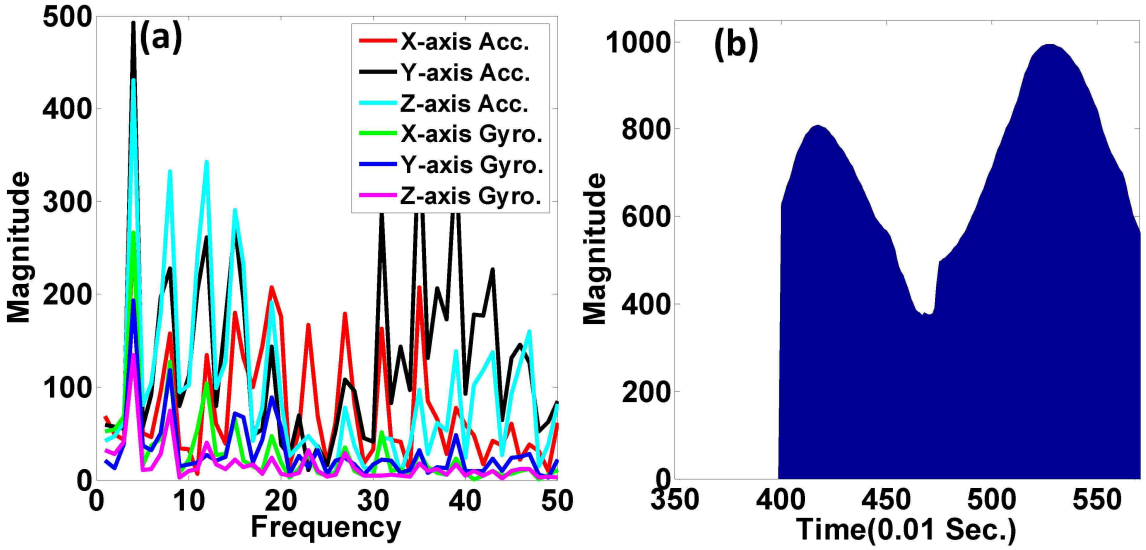

Figure 4.6. Step detection. (a) DFT results of 4-second signal with 400 samples. (b) The variance metric vs. signal length to detect the accurate step period.

Figure 4.6(a) shows the results after applying DFT to six signals of accelerometer and gyroscope over a time sliding window L of 400 samples. In our IMU device, sensor data are collected at the rate of 100 samples per second, so 400 samples of data imply data collected in 4 seconds. The horizontal axis in Figure 4.6(a) is the frequency which is related to the number of periods within the time sliding window. The vertical axis in Figure 4.6(a) is the corresponding magnitude. The frequency component related to the step number should have high magnitude while frequency components corresponding to noise should have low magnitude. We compute the principle frequency of all six signals, f^*, by

$$f^* = \arg\max_f \sum_{i=1}^{6} |F_i(f; L)| \tag{4.1}$$

where $|F_i(f; L)|$ denotes the magnitude of frequency component f of the ith signal within the time sliding window L. Note that f^* is an integer in DFT. In Figure 4.6(a), $f^* = 4$, but is there exactly 4 steps during this time sliding window (400 samples)? The answer is possibly NO. If there are 3.8 or 4.3 steps in the sliding window L, the corresponding principle frequency will be rounded to 4. We need to search the accurate beginning and end sampling moments of complete steps in the signals to estimate the speed for Dead-Reckoning (DR). Otherwise, DR will deviate from the truth quickly due to the accumulated error. Observing that the principle frequency has a large difference compared to its neighboring frequencies, we propose a new metric M_L, the magnitude variance of the principle frequency compared with its neighboring frequencies, to search the accurate steps:

$$M_L = \sum_{i=1}^{6} var([|F_i(f^*\text{-}1; L)|, |F_i(f^*; L)|, |F_i(f^*\text{+}1; L)|]) \tag{4.2}$$

We gradually increase the time sliding window L. For each L, we compute f^* by Equation 4.1 and then compute M_L by Equation 4.2. Figure 4.6(b) shows the plot of M_L versus L. We can see the first peak is around 420 with $L = [1, 420]$, which means that there are 4 steps in 420 samples (4.2 seconds), i.e., each step period is about 105 samples. If we keep increasing L, we will find another peak around 525 in $L = [1, 525]$ which means that there are 5 steps in 525 samples. The peaks in Figure 4.6(b) indicate that at these points, the magnitude of the principle frequency has the largest difference compared to its neighboring frequencies. Thus, we can detect the exact number of steps by adapting this time sliding window technique.

4.7.2. Speed Estimation. Practically, walking/running speed varies from person to person. Even for the same person, the moving speed may not be a constant over time. Integration on acceleration to obtain speed accumulates errors very fast, making it imprac-

tical for speed estimation. Observing that the magnitude of movement is approximately proportional to speed, we propose to use the maximal difference of angular velocity to measure the movement intensity. The measurement is only valid in complete movement pattern periods, which is at least one step. That is one of the reasons why we need accurate step detection and speed is calculated in the unit of step. The speed for step n is defined as

$$v_n = \alpha(max_{s\in[s_b^{(n)},\, s_e^{(n)}]} \parallel \mathbf{a}_s \parallel - min_{s\in[s_b^{(n)},\, s_e^{(n)}]} \parallel \mathbf{a}_s \parallel) \tag{4.3}$$

where $s_b^{(n)}$ and $s_e^{(n)}$ denote the beginning and end sample of the nth step and $\parallel \mathbf{a}_s \parallel$ is the magnitude of angular velocity at sample s. α is the calibration factor depending on specific persons. When visual signal is used, α is determined by a similarity warping matrix which will be introduced in subsection 4.4.

4.7.3. Forward Moving Direction Determination. The 3D acceleration vectors in the IMU coordinate during each step can be projected to the horizontal plane in the world coordinate to infer the forward moving direction [25, 28]. This method works for professional IMUs. But for low cost IMUs such as the IMU module built in smatphones which is more likely to be influenced by noise, it performs poorly. Figure 4.7 shows the results when acceleration collected by a smartphone in 4 steps is projected to the world coordinate's horizontal plane. There is no obvious forward moving direction.

Considering the 3D acceleration vectors during a time sliding window as time-series signals, we transform them into the frequency domain. Since the principle frequency during the sliding window is already detected in the step detection process (subsection 4.3.1), we treat all non-principle frequency components as noise and zero them out. Then, the filtered signal is transformed back to the time domain and is projected to the world coordinate's horizontal plane. As shown in Figure 4.7(b), the moving direction is obvious.

Ellipse-fitting (i.e., 2D Principle Component Analysis) is applied to the projected principle acceleration and the semi-major axis of the ellipse represents the forward moving direction $[u_x\ u_y]$.

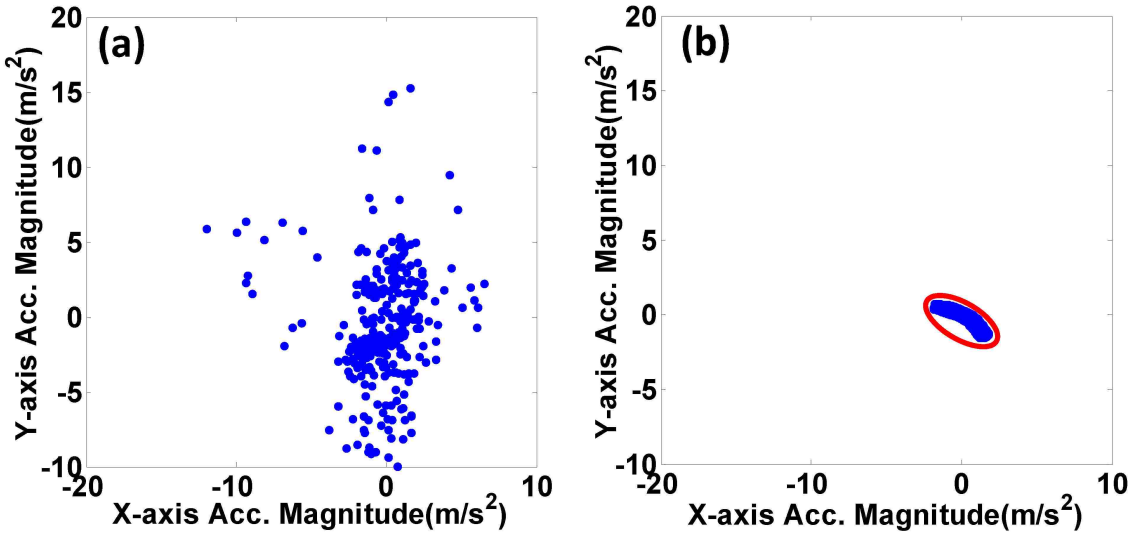

Figure 4.7. Determine the forward moving direction. (a) The acceleration in a short period (4 steps) is projected to the horizontal plane in the world coordinate. (b) The acceleration corresponding to the principle frequency in a short period is projected. The semi-major axis of the ellipse represents the forward moving direction.

4.8. INTEGRATION OF VISUAL AND IMU TRACKING

As shown in Figure 4.8, our cooperative people tracking system is divided into three parts: initialization, tracking and re-identification.

4.8.1. Initialization. The target to be tracked is initially identified by human. Figure 4.8(a) and Figure 4.8(b) show the visual trajectory (red) and IMU trajectory (green) in the first sliding window L_1, respectively. The trajectory generated by IMU tracking is in the world coordinate, so it is a 2D curve in the horizontal plane viewed from top to down. Unlike IMU trajectory, visual trajectory is in the image coordinate depending on the

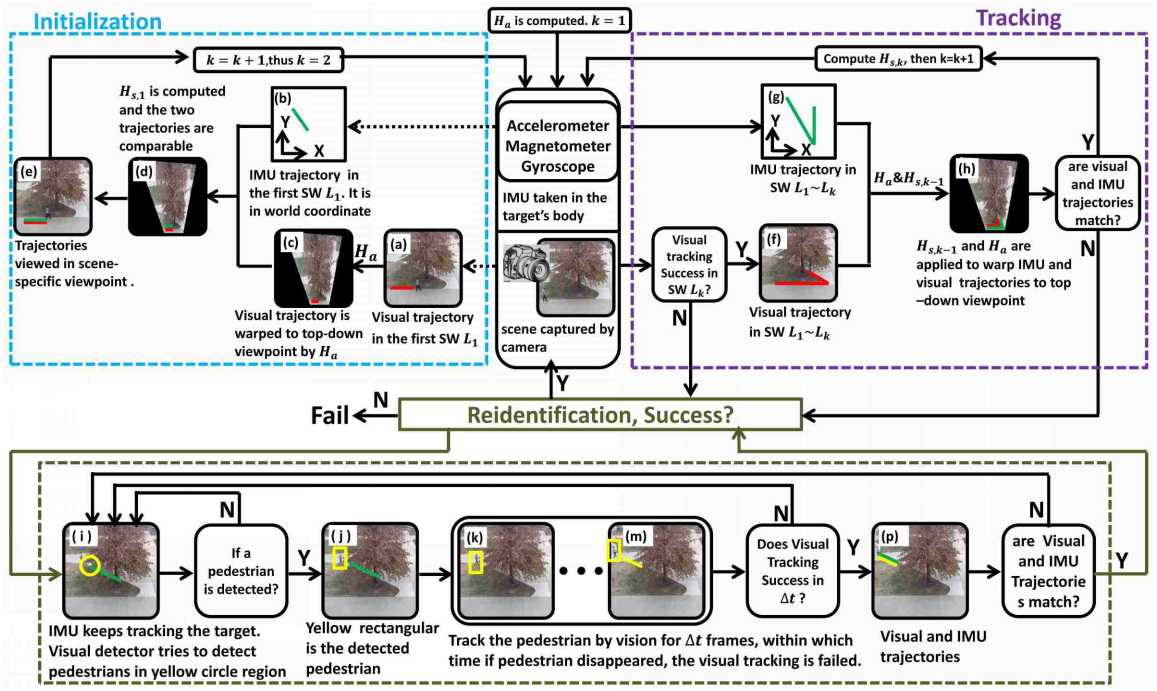

Figure 4.8. The flow chart of the cooperative tracking system. SW: Sliding Window.

specific camera viewpoint, thus they are not directly comparable. We warp the visual trajectory from scene-specific viewpoint to the top-down viewpoint by $\mathbf{H}_a$ (subsection 4.2.2), as shown in Figure 4.8(c). Since the transformation between the warped visual trajectory (Figure 4.8(c)) and IMU trajectory (Figure 4.8(b)) is just rotation, translation and scaling (i.e., similarity transformation), we match the two trajectory curves by computing the similarity transformation matrix $\mathbf{H}_{s,k}$ in sliding window L_k using the least square procedure:

$$\arg\min_{\mathbf{H}_{s,k}} \sum_{t} (\mathbf{H}_{s,k}\mathbf{T}_t^{(v,k)} - \mathbf{T}_t^{(s,k)})^2 \tag{4.4}$$

where $\mathbf{T}_t^{(v,k)}$ and $\mathbf{T}_t^{(s,k)}$ denote the uniformly sampled point t on the warped visual trajectory and sensor trajectory in sliding window L_k, respectively. The initialization step is performed in the first sliding window, so $k = 1$. Figure 4.8(d) shows the result of IMU tra-

jectories matched to visual trajectories. For better visualization, we can warp the top-down viewpoint to the scene-specific viewpoint by the inverse of $\mathbf{H}_a$. Therefore, two matrices, $\mathbf{H}_a$ (homography transformation) and $\mathbf{H}_{s,k}$ (similarity transformation), make visual and IMU trajectories compatible. $\mathbf{H}_a$ does not change unless the scene-specific viewpoint changes. $\mathbf{H}_{s,k}$ keeps being updated during each sliding window of the cooperative tracking.

4.8.2. Tracking. The initialization step only needs to be performed once, then our system goes to the normal tracking. Figure 4.8(f) and (g) show the trajectories based on visual and IMU tracking, respectively, in sliding windows $L_1 \sim L_k$. Then, $\mathbf{H}_{s,k-1}$ and $\mathbf{H}_a$ are applied to warp IMU and visual trajectories to the top-down viewpoint. The average distance d between trajectories in Figure 4.8(h) is calculated. If $d < d_{thr}$, visual and IMU trajectories are matched, then new $\mathbf{H}_{s,k}$ is computed using Equation 4.4 and we go to the next sliding window. In our tracking system, we set $d_{thr} = 80$inches.

4.8.3. Re-identification. Two cases lead to the re-identification: (1) The target disappears in visual tracking such as moving out of the visual field or being occluded by other objects; (2) Visual and IMU trajectories do not match each other, which may be caused by tracking drift (i.e., track a non-target pedestrian).

As shown in Figure 4.8(i), IMU keeps tracking the target even it is occluded (green curves). Meanwhile, visual pedestrian detector tries to detect pedestrians in a search region estimated by IMU (yellow circle in Figure 4.8(i)). If detected, the pedestrian will be visually tracked for Δt frames (Figure 4.8(k)-Figure 4.8(m)). Δt is set as 150 frames (5 seconds) here. If any visual tracking failure happens within the Δt frames, we go back to the IMU-tracking (Figure 4.8(i)). If the tracking within the Δt frames succeeds, the average distance d between IMU and visual trajectories during Δt is computed to judge if they match. If $d < d_{thr}$, the target is re-identified and we go to the normal tracking again. Otherwise, we go to the IMU-tracking (Figure 4.8(i)) for re-identification.

The above cooperative people tracking system elucidates why visual tracking and IMU tracking are "complementary". First, when visual tracking fails, IMU tracking keeps working and offers the clue where the target could be, helping visual tracking reidentify the target. Secondly, the visual trajectory corrects the bias of speed and forward direction estimation in IMU tracking by the similarity matrix $\mathbf{H}_{s,k}$. The calibration coefficient in Equation 4.3 is also compted by $\mathbf{H}_{s,k}$ once we know the length of matched visual and IMU trajectories. As we keep updating $\mathbf{H}_{s,k}$, visual tracking rebuilds the relationship with IMU tracking and rectifies the deviation of IMU-based tracking trajectory.

4.9. EXPERIMENTS

To test the effectiveness of our cooperative tracking system, we apply it for people tracking in daily environments. Consumer electronics such as smartphones embedded with IMU modules are selected as the IMU signal collector. The IMU module in a smartphone is low cost and sensitive to noise. If our system works well using smartphones, we believe it will work using expensive and professional IMU devices. In addition, the popularity of smartphones offers more possibilities of applications of our tracking system. We developed an App to collect IMU signals when the target is moving or standing. The IMU signals are transmitted back to a groundstation by GSM. Meanwhile, a stationary surveillance camera collects visual signal of the target person. The visual signal is taken at 30 frames per second and the sampling frequency of IMU signal is 100 samples per second. The data transmitted between a smartphone and the groundstation is about 13.5 MB per hour. To synchronize the two signals, for every frame of the video, the nearest IMU signal is found according to the timestamp provided by the smartphone system.

4.9.1. Evaluation. We recorded four videos in different conditions to test the performance of our cooperative people tracking system (Figure 4.9).

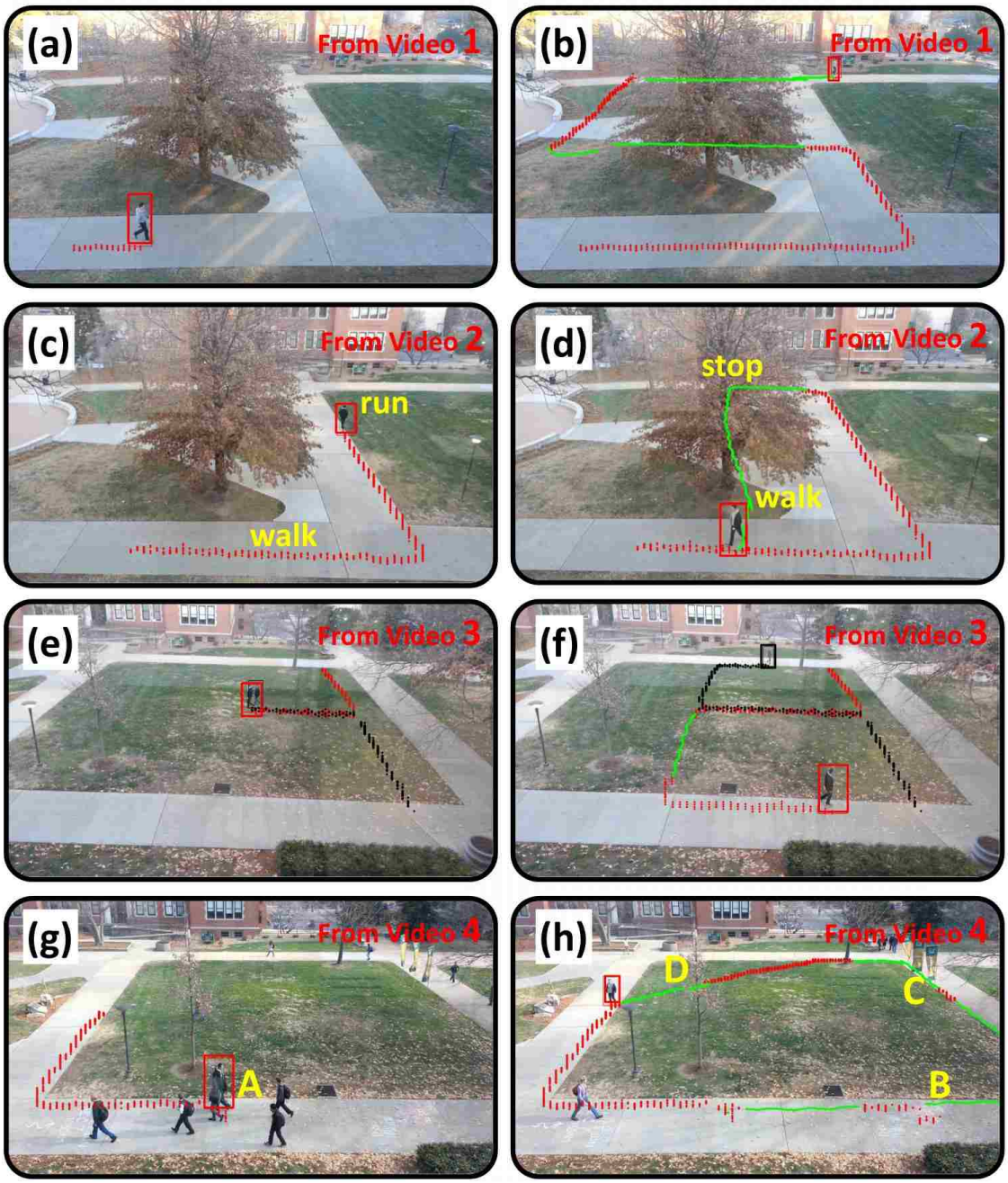

Figure 4.9. Trajectories of the target person. Red curves are visual trajectories and green curves are IMU trajectories when visual tracking fails. (a)(b) Screenshots from video 1. (c)(d) Screenshots from video 2. (e)(f) Screenshots from video 3. (g)(h) Screenshots from video 4.

Video 1 was taken in an occlusion environment with a small slope. The target person was occluded by a tree twice for 9 and 16 seconds, respectively. Figure 4.9(a)(b) show that the target is successfully tracked by our system in long term and heavy occlusions.

In video 2 (Figure 4.9(c)(d)), the target changed his speed from walking to sudden run and then stopped when hidden by the tree. Ten seconds later, the target began to walk in a direction different from his previous direction. This case is difficult for vision-based tracking algorithm because the target changes his speed and forward direction when occluded.

Video 3 (Figure 4.9(e)(f)) is an example of associating IMU trajectories with visual ones when multiple cooperative people carrying IMUs were in the scene. One target was occluded by the other. When they departed towards different directions, visual tracking failed because of the clutter of similar appearance. However, IMU tracking tracked and re-identified the target successfully.

In video 4 (Figure 4.9(g)(h)), the target was occluded by moving pedestrians at location A and moved out of visual field from location B. At locations C and D, the target was occluded by background objects frequently. Despite the complex scenarios, our cooperative tracking system can persistently track the target.

Table 4.1 summarizes the quantitative evaluation on our cooperative tracking system. The two trajectories are synchronized by comparing their timestamps. The trajectory error is computed by the average difference between a tracked trajectory and its ground truth. The ground truth is labelled by a human annotator in each frame of the videos. The average errors of vision and IMU tracking in our cooperative people tracking are 37 inches and 44.3 inches, respectively. Visual tracking fails when the target is heavily occluded in the first time in each video, but our proposed tracking system can persistently track the target by combining visual and IMU tracking. The experiments validate that visual track-

ing combined with IMU tracking can achieve both accuracy and persistency. Here we did not provide the quantitative IMU-only tracking results, which is because IMU-only tracking needs manually set parameters such as the speed coefficient α in Equation 4.3 and the orientation offset [25]. These handcrafted parameters are person-specific and will largely affect the tracking performance. In the next subsection, we compare the *shape* of IMU tracking results which does not need manual parameters.

Table 4.1. People tracking results. FV: number of frames successfully tracked only by Vision. FS: number of frames successfully tracked by our cooperative people tracking system. VTAE: Visual Trajectory Average Error. ITAE: IMU Trajectory Average Error.

Video	#frames	FV	FS	VTAE(in)	ITAE(in)
1	2009	580	2009	50.3	66.0
2	1940	717	1940	35.15	45.2
3	1063	625	1063	22.4	15.8
4	1857	392	1857	33.1	36.1
avg				37.0	44.3

4.9.2. Comparison. We have seen IMU tracking can assist visual tracking in subsection 4.5.1. We use video 1 as an example to compare different IMU tracking methods and shows the benefit of visual tracking to help IMU tracking. Figure 4.10(a) is the ground truth of the target trajectory in video 1, which is obtained by warping the target's trajectories in the scene-specific viewpoint to the top-down viewpoint using $\mathbf{H}_a$. All trajectories in Figure 4.10 are in the horizontal plane of the world coordinate. Figure 4.10(b) is based on the PCA2D (i.e., 2D Principle Component Analysis) method introduced in [25] which detects step in the time domain. There are many misdetections on step and direction by this approach and the tracked trajectory drifts away from the ground truth largely. Figure

4.10(c) is the result by our IMU tracking method without any assistance from the visual tracking, which is better but still drifts away from ground truth a little. Figure 4.10(d) shows the trajectory results of IMU tracking assisted by the visual tracking. When visual tracking is combined, visual trajectories constantly adjust the orientation and scale of IMU trajectories with $\mathbf{H}_{s,k}$. The IMU trajectory in Figure 4.10(d) is very close to the ground truth.

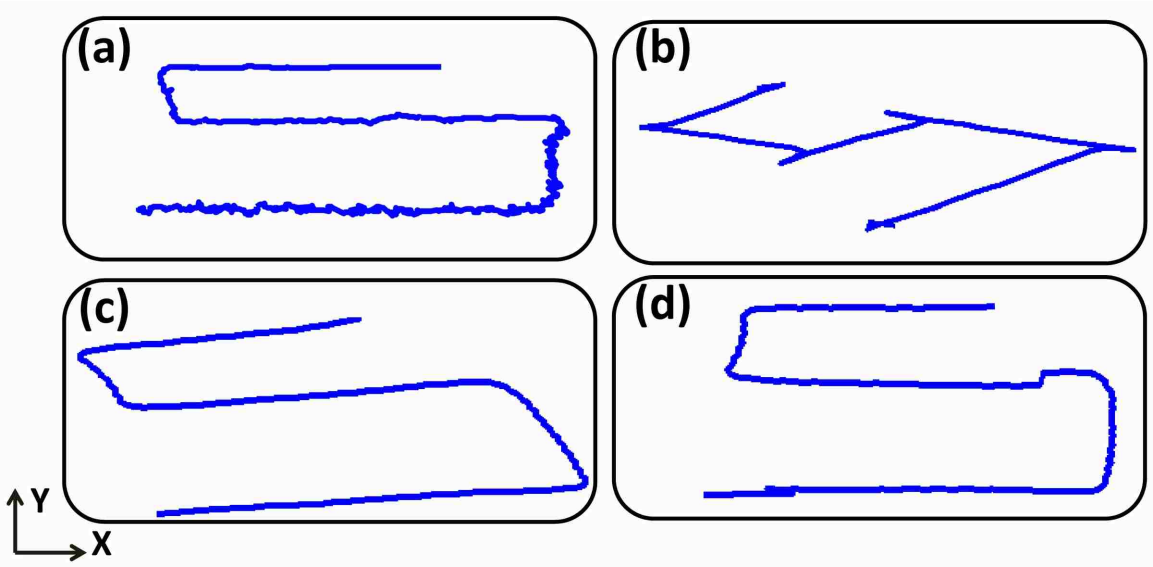

Figure 4.10. IMU trajectories generated by three different approaches. (a) Ground truth trajectory; (b) Trajectory by time-domain step detection [25]; (c) Trajectory by our DFT approach; (d) Trajectory by our DFT approach assisted by the visual signal.

4.10. SUMMARY FOR SECTION 4

To persistently track cooperative people such as children and patients in challenging scenarios, we present a novel tracking system combining the visual and Inertial Measurement Unit (IMU) signals, obtained from surveillance cameras and IMU devices carried by the targets themselves, respectively. Not only can IMU assist visual tracking when the tar-

get is occluded, but also the challenges of IMU tracking (calibration and drift) are alleviated when visual signals are available. Experimental results show that visual and IMU tracking are complementary to each other and their integration achieves very good performance on persistent people tracking under challenging daily environments.

5. INDOOR LOCALIZATION BY SIGNAL FUSION

5.1. PROBLEM STATEMENT

This section is revised based on the paper published in 18th International Conference on Information Fusion [54].

Indoor geo-location is an important component of smart buildings, which can be divided into two categories: indoor navigation and indoor localization. Indoor navigation provides the route to the user's destination while indoor localization tells the user where she/he is. This section focuses on indoor localization because it has many daily applications in different scenarios such as hospitals, shopping malls, museums and office towers. In addition, indoor localization lies the basis for further navigation. The technology of indoor localization of human can also be applied to robots in a building.

Visual signal is intuitively useful for indoor localization as people generally know where they are according to what they see. A typical vision based indoor localization algorithm consists of two stages: building a reference database and online localization by image matching. The database is built by the feature representation of geo-tagged images taken within a building. When positioning, a new image around a user's location is taken and it is compared with the database to estimate her/his location.

Although vision based indoor localization has been studied for several years [55, 56, 57, 58, 59], there are still several unsolved challenges when practical implementations are considered: (1). In a common building, thousands of images can be recorded as references and millions of visual features can be detected and extracted from the images. An efficient way to build the reference database is needed. (2). In online localization, the query image will be compared with the whole database, which decreases the efficiency when the

database is huge. (3). A building may have unified decoration style, so similar scenes exist in different positions, which is hard to be visually classified.

The pervasiveness of smartphones offers the opportunities to assist visual indoor localization with WiFi and orientation signals and mitigates the challenges described above. The WiFi module collects WiFi signals and inertial sensors (e.g., accelerometer and magnetometer) can be used to measure the orientation of a smartphone when its user takes photos. In this section we fuse the visual signal and other contextual information offered by WiFi and inertial sensors to make the *energy-saving, efficient and accurate* indoor localization possible.

5.2. PREVIOUS WORK

WiFi and inertial sensors can be individually applied to indoor localization. [60] and [61] extracted sophisticated features from the raw Received Signal Strength Indication (RSSI) of WiFi signals to describe a location. However, it is possible that some hotspots are shut down or the RSSI value of a specific hotspot is changed because of the device update, which dramatically decreases the localization accuracy of merely WiFi-based approaches. [62] and [63] utilized inertial sensors such as accelerometers and magnetometers to perform step detection, speed estimation and heading direction determination and the three components can be built in the Dead Reckoning framework to obtain the user's trajectory. The trajectory can then be matched with the floor plan to infer the user's location. Inertial sensors based approaches do not need to collect reference data in the building except the floor plan, but Dead Reckoning suffers from cumulative errors or drift, making the trajectory estimation inaccurate.

For vision-based indoor localization, in [64], local affine invariant points were extracted from images. These points were quantized into visual words by K-means. Each

image can be represented as a vector and each dimension of the vector represented a count of the occurrence of a visual word. The feature descriptor in terms of visual words was used for image/object matching. [55] proposed a coarse-to-fine localization system where several candidate images were obtained by comparing the similarity of a query vector with reference vectors in the database, then a keypoint voting algorithm was adopted to determine the final matched image. Although online localization is reliable, the database is still computationally costly to be built. [58] considered the global features, including a weighted gradient orientation histogram and color histogram to localize people in indoor environment, but merely global information can not distinguish two different locations with similar decoration. To reduce the cost to build database, [65] improved the traditional K-means by compressing and removing patterns at each iteration which are unlikely to change their membership thereafter. To improve the localization accuracy, [66] adopted epipolar geometry constrains to refine the location.

Previous work also investigated signal fusion methods for indoor localization. [67] and [68] utilized WiFi signal to rectify the trajectory obtained by inertial sensors. [69] combined Radio Frequency and WiFi to improve the localization accuracy. [70] combined more signals (e.g., WiFi, sound, motion, color) to build the localization algorithm. These signal fusion methods just concatenate several sensors together to improve the localization accuracy but the problems of feature selection on signals and how to efficiently combine signals are often overlooked.

5.3. PROPOSED METHOD AND ALGORITHM OVERVIEW

In this section, we propose a novel tree-based indoor localization algorithm in which the WiFi, orientation and visual signals from smartphones are integrated into a signal tree. In the proposed algorithm, WiFi is used for coarse positioning, thus the problem of WiFi

environment change is mitigated since WiFi is only used for coarse localization instead of fine localization. Inertial sensor is not used to estimate the trajectory, but to obtain the orientation towards which the user takes photos. The problem of image scenes caused by unified decoration can also be alleviated because similar scenes may have different WiFi and orientation information. This algorithm consists of two stages: building the signal tree and online localization (Figure 5.1).

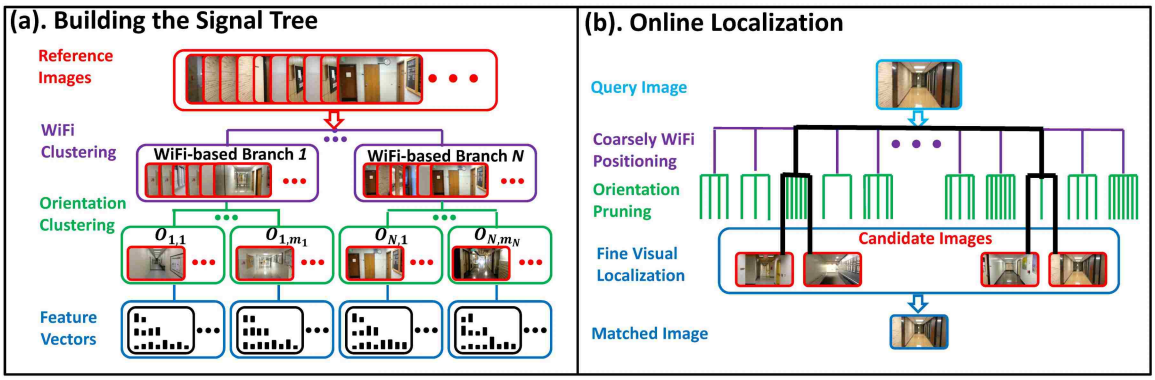

Figure 5.1. Overview of the propose indoor localization algorithm.

Building the Signal Tree (Figure 5.1(a)): WiFi signals are collected in a building, tagged with hotspots' Received Signal Strength Indication (RSSI) and the positions where the signals are collected. Reference images are densely captured in a building and labeled with the orientation and location information. Essentially, the construction of a signal tree is the process of clustering and describing reference images with the aid of WiFi and orientation signals. Locations are described by WiFi fingerprints and then all WiFi fingerprints are clustered into branches. All reference images are partitioned into the WiFi branches based on their spatial distance to WiFi fingerprints' positions (purple part in Figure 5.1(a)). Then, images in the same WiFi branch are further classified according to their orientation similarity (blue part in Figure 5.1(a)). Images in one leaf node share the same WiFi and

orientation labels. Given a leaf node, each image is described by multiple level descriptors (blue part in Figure 5.1(a)).

Online Localization (Figure 5.1(b)): When a user takes a photo to localize herself/himself, WiFi and orientation signals are recorded automatically and synchronously. The signal tree is then searched to find the best matched image that indicates the user's location. The query WiFi fingerprint coarsely determines which WiFi branches the matched image belongs to. Orientation information further rules out impossible reference images. Then, every searched leaf node gives a candidate image best match to the query image within a leaf node. Finally, these candidate images are compared to decide the final matched image. The matched image's tagged position indicates the user's location.

Our proposed tree-based indoor localization algorithm does not bring extra work to users. A user only needs to take a photo while the WiFi and orientation signals are automatically recorded. Then, the user can discover her/his location in the building based on the signal tree. In addition, when building the signal tree, images are naturally clustered into groups sharing similar WiFi and orientation environments. Combined with parallel computing, the time needed to build the database can be remarkably reduced. In online localization, WiFi and orientation can not only offer more context information to refine the matched location, but also rule out impossible reference images, decreasing computational cost and increasing localization accuracy.

In the rest of this section, building the reference signal tree is described in subsection 5.3. Detailed search strategies for localization are introduced in subsection 5.4. Then, experimental results are presented with comparisons and evaluations.

5.4. BUILDING THE SIGNAL TREE

This subsection presents the algorithm to build the signal tree. The surrounding sensor environment (WiFi and orientation) and image attributes of a position are fused together in the hierarchical signal tree to describe that location.

5.4.1. Building WiFi Branches. WiFi signals are sparsely collected in a building. A location is described by the WiFi fingerprint, which is a vector with each dimension equaling to the processed RSSI of a certain hotspot. To better describe the WiFi environment of a building, all fingerprints are clustered into groups.

It is reported in [71] that WiFi signal gets less reliable when its RSSI is lower, so we normalize the raw RSSI by an exponential distribution

$$f^*_{i,j} = \lambda exp[\,\lambda \frac{f_{i,j} - f_{min}}{f_{max} - f_{min}}\,] \tag{5.1}$$

$$\lambda = \frac{f_{max} - f_{min}}{f_{mean} - f_{min}} \tag{5.2}$$

where $f_{i,j}$ is the raw RSSI of WiFi hotspot j at location i. $f^*_{i,j}$ is the normalized RSSI. f_{max}, f_{min} and f_{mean} are the maximal, minimal and average RSSI of all $f_{i,j}$. λ is the rate parameter. Then, the WiFi fingerprint at location i, $\mathbf{f}_i$, is defined as

$$\mathbf{f}_i = [f^*_{i,1}, \ \ldots\ , \ f^*_{i,j}, \ \ldots, \ f^*_{i,N_j}] \tag{5.3}$$

where N_j is the number of WiFi hotspots in a building.

WiFi Clustering: Treating WiFi fingerprints individually is not robust to environment changes such as shutdown of some hotspots. Thus WiFi fingerprints are clustered

into groups based on their *WiFi fingerprint similarity* and *spatial distance*. The clustering procedure is divided into two steps (Figure 5.2): Bottom-Up clustering by WiFi fingerprint similarity and Top-Down cutoff by spatial distance.

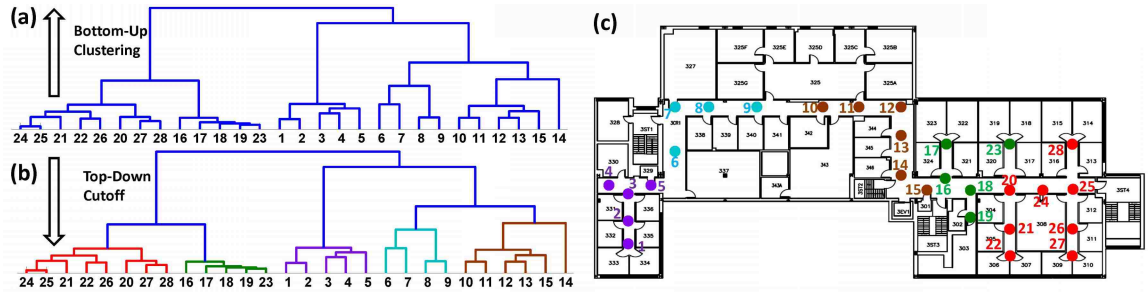

Figure 5.2. WiFi fingerprints clustering. (a) Button-Up WiFi Clustering Dendrogram; The number in the leaves are indices of WiFi fingerprints. (b) Top-Down Cutoff Dendrogram. Leaves sharing the same color belong to the same WiFi cluster. (c) The floor plan where the WiFi fingerprints are collected. Dots indicate where the WiFi signals are collected and surrounding numbers are the corresponding indices of WiFi fingerprints. Dots sharing the same color belong to the same WiFi cluster corresponding to (b).

As shown in Figure 5.2(a), WiFi fingerprints are firstly hierarchically clustered from bottom to up. Initially, each WiFi fingerprint is a cluster. Then two clusters most similar to each other are merged into a bigger cluster. This agglomerative mergence operation is performed iteratively and stops when all WiFi fingerprints are in one cluster. The similarity metric of two clusters is defined by Ward's method [72].

$$S(A,B)=\sum_{k\in A\cup B}\|\mathbf{f}_k-\bar{\mathbf{f}}_{A\cup B}\|-\sum_{k\in A}\|\mathbf{f}_k-\bar{\mathbf{f}}_A\|-\sum_{k\in B}\|\mathbf{f}_k-\bar{\mathbf{f}}_B\| \tag{5.4}$$

where $\mathbf{f}_k$ denotes a WiFi fingerprint. $\bar{\mathbf{f}}_A$, $\bar{\mathbf{f}}_B$ and $\bar{\mathbf{f}}_{A\cup B}$ are the centroids of cluster A, B and $A \cup B$, respectively. $\|\cdot\|$ is Euclidean distance.

The WiFi hierarchical tree in Figure 5.2(a) only shows a multi-branch hierarchy rather than a set of clusters. It is partitioned into several groups based on WiFi fingerprints' spatial distances. As shown in Figure 5.2(b), from top to down of the WiFi hierarchy, every node is checked if the maximal value of spatial distance between all pairs of WiFi fingerprints belonging to this node is less than a predefined threshold d_{thr}^{WiFi} (e.g., d_{thr}^{WiFi}=20 meters). When the maximal value is actually less than d_{thr}^{WiFi}, the WiFi fingerprints belonging to this node will be considered to be the same group.

Note that the number of WiFi clusters is automatically defined by the fingerprint similarity and spatial distance instead of presetting by human. Figure 5.2(c) shows the final clustering results of WiFi fingerprints in a university building. The clustering result accurately reveals the actual WiFi environment of this building. Then, each reference image is clustered to the nearest WiFi group based on spatial distance.

5.4.2. Building Orientation Branches. Inertial sensors, including accelerometer and magnetometer, are equipped in most smartphones. When smartphones are stable (taking photos), accelerometer measures the gravity while magnetometer measures the earth's magnetic field. Gravity and magnetic field set up a world coordinate system. Thus, every point in the phone's coordinate system can be converted to the world coordinate system by a transformation matrix.

Let $\mathbf{Q}_{p\rightarrow w}$ denote the transformation matrix from phone coordinate system to the world coordinate system, which can be obtained by the algorithm described in [33]. Figure 5.3 shows the scenario when a user takes a photo, the yellow vector $\mathbf{c}_p$ represents the orientation that the camera is towards. Note that $\mathbf{c}_p$ is a constant vector in phone's coordinate system. $\mathbf{c}_p$ is transformed to the world coordinate by

$$\mathbf{c}_w = \mathbf{c}_p \times \mathbf{Q}_{p\rightarrow w} \tag{5.5}$$

Denoting $\mathbf{c}_w = [c_{wx}\ c_{wy}\ c_{wz}]^T$, we project the orientation to the horizontal plane in the world coordinate, i.e, vector $O = [c_{wx}\ c_{wy}]^T$ is the orientation on the floor plan which the photo is taken towards.

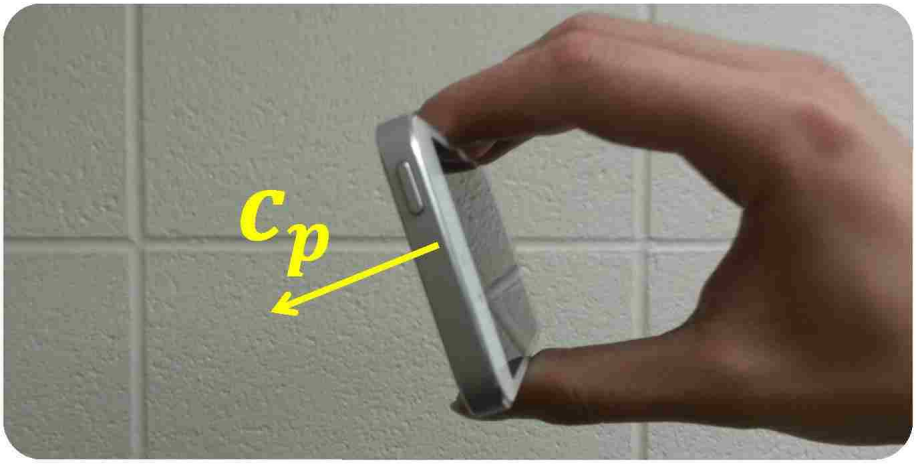

Figure 5.3. The scenario when a user takes a photo for localization. $\mathbf{c}_p$ is a constant vector in photo's coordinate system, pointing outside the back of the phone.

Orientation Clustering: When building the visual database, previous work [57, 58, 59, 73] mostly took thousands of photos manually, which is pretty time consuming. Instead, *we collect continuous videos and orientation information simultaneously. Every frame of these videos is a reference image.* Without loss of generality, we make the explanation with a simple floor plan. For example, eight video clips were recorded in a building following the eight routes defined in Figure 5.4(a). Each frame in the videos is tagged with its corresponding orientation. Each video clip was recorded following the same direction, therefore the orientations of all frames in a video are similar, naturally forming a cluster of orientation.

Figure 5.4(b) shows the distributions of eight orientation clusters corresponding to the eight routes in Figure 5.4(a). The orientation distribution of each video clip is not a

constant impulse distribution due to noise. The orientation distribution of a video clip q is modeled by a Gaussian distribution $N(\mu_q, \sigma_q)$. Suppose the entire floor plan in Figure 5.4(a) is in one WiFi cluster, overlapped orientation clusters can be further merged into a bigger cluster. The similarity of two distributions q_1 and q_2 is defined as:

$$S_{q_1,q_2} = \frac{\sigma_{q_1}^2 + \sigma_{q_2}^2}{|\mu_{q_1} - \mu_{q_2}|} \tag{5.6}$$

If the centroid of two distributions are close to each other and their inter-distribution variance ar small, then they can be merged into a bigger cluster. In Figure 5.4(b), the eight orientation distributions can be clustered into four clusters. Each of the four orientation clusters is one orientation subbranch within the same WiFi branch.

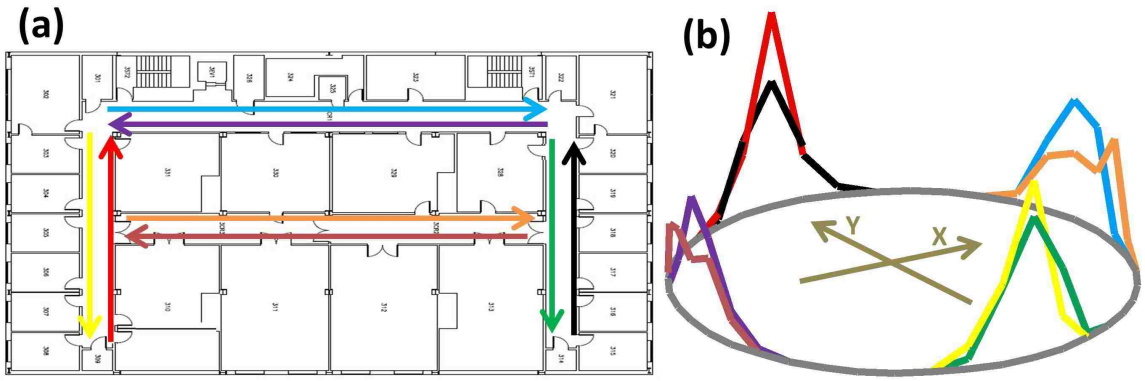

Figure 5.4. Orientation clustering. (a) Floor plan of a building with 8 routes to record videos and sensor information. (b) Orientation distributions calculated by the data collected according to (a).

5.4.3. Building Image Leaf Nodes. In this subsection, we propose a Multiple Level Image Description (MLID) method to describe images in leaf nodes of the signal tree (Figure 5.1). MLID is based on Term Frequency Inverse Document Frequency(TF-IDF)

[64], but we improve it in three-folds: (1) Dense Scale Invariant Feature Transform(SIFT) keypoints are extracted in the low texture areas. (2) Divisive hierarchical clustering is adopted rather than K-means. (3) Each image is described as multiple vectors, thus both global and local information of an image is recorded. As shown in Figure 5.6, MLID consists of four steps:

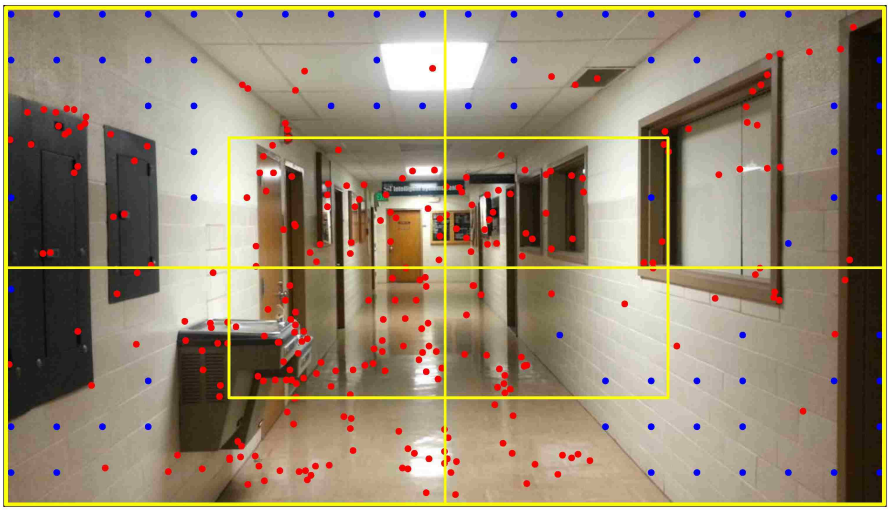

Figure 5.5. SIFT points in a user-taken image. The red points are the salient SIFT keypoints and the blue points are the dense SIFT keypoints. The image is equally divided into five subimages.

(1). Feature Extraction: In Figure 5.5, salient SIFT keypoints (red points) are firstly extracted from an image. Dense SIFT keypoints (blue points) are then extracted in the low texture areas ignored by salient SIFT such as some parts of the ceiling and walls. Features of keypoints extracted from all reference images in a leaf node are collected into a large feature pool (represented as purple circles in Figure 5.6(a)).

(2). Feature Clustering: Divisive hierarchical clustering is applied to partition SIFT features in the feature pool. In Figure 5.6(b), SIFT features are firstly clustered into t groups ($t = 2$ in Figure 5.6(b)) at level 1 ($l = 1$) based on Euclidean distance. Then each

cluster in the first level are clustered into t groups. The process is performed repeatedly until every leaf of the feature clustering tree has a small set of SIFT feature descriptors (e.g., less than 100 SIFT features on the leaves). The symbol around each node represents *the mean of SIFT feature vectors in that subtree* (called visual word) and the visual words at each level forms the visual codebook for that level. In Figure 5.6(b), the symbols in each dotted rectangle belong to one visual codebook.

(3). Feature Interpretation: As shown in Figure 5.6(c), SIFT features in an image can be interpreted into visual words hierarchically based on the visual codebooks at different levels. A SIFT feature is interpreted as the visual word which is the closest to the SIFT feature based on Euclidean distance. For example, at level 1 of Figure 5.6(c), 15 SIFT descriptors are close to visual word 1 (red star) and 10 descriptors are close to visual word 2 (green circle). The interpreted visual words at level 1 are finely interpreted at following levels.

(4). Image Description: Based on the hierarchical feature interpretation, an image can be described by multiple vectors. In each level, the dimension of the vector is the same as the number of visual words and each dimension is the count of the occurrence for corresponding visual word. For example, in level 1 of Figure 5.6(d), 15 SIFT descriptors belong to visual word 1 (red star) and 10 descriptors belong to visual word 2 (green circle), the description vector is $[15, 10]$, normalized as $[0.6, 0.4]$. The feature descriptors are finely computed in the subsequent levels according to more and more detailed visual codebooks.

Spatial information is also considered when formulating the feature description of an image. As the yellow lines in Figure 5.5 illustrate, the image is first equally divided into four subimages and the fifth subimage is in the center of the image with the same size of other four subimages. Multi-level feature vectors are calculated based on individual subimages and then they are concatenated to form long vectors to describe the whole image.

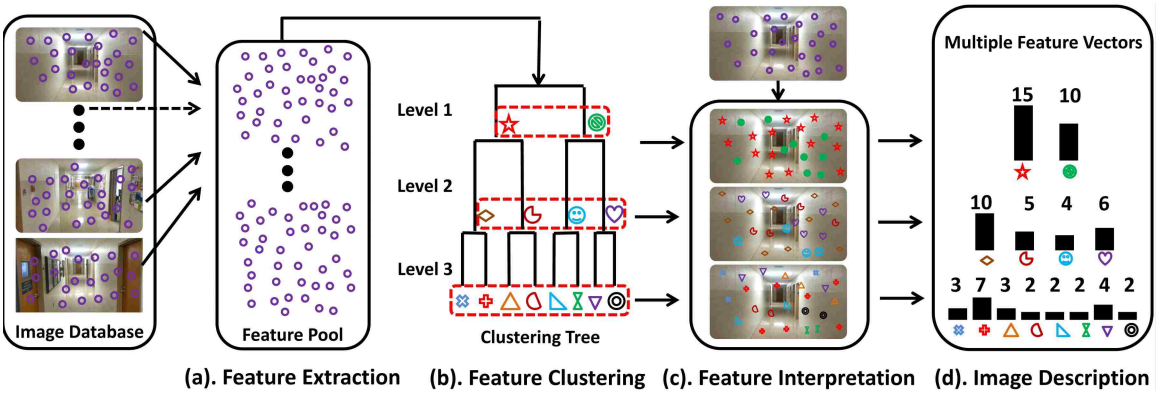

Figure 5.6. Flow chart of the Multiple Level Image Descriptions (MLID) method.

The proposed MLID algorithm keeps both global and local information of images. At the top level, SIFT descriptors are coarsely clustered and the dimension of feature vector is low, so the global information of the image is reflected. As the descriptors are finely clustered, dimension of feature vector gets larger and more detailed information is recorded. Note that, compared with K-means, there is no need to predefine how many groups we should cluster the SIFT descriptors, which is another advantage of the MLID method to handle different unknown scenes.

5.5. ONLINE LOCALIZATION

When a user takes a photo to localize herself/himself, WiFi and orientation signals are recorded synchronously. This subsection presents the search strategy to find the best matched reference image to identify a user's location, which consists of three stages: coarsely WiFi positioning, orientation pruning and fine visual localization.

5.5.1. Coarsely WiFi Positioning. Let $\mathbf{f}_0$ be the WiFi fingerprint submitted by the user and can be computed by Equation 5.3. Assume there are N_{WiFi} WiFi clusters in the signal tree. The centroid of WiFi clusters are denoted as $\mathbf{f}_n(n = 1...N_{WiFi})$. The distance between $\mathbf{f}_0$ and any WiFi cluster $\mathbf{f}_n$ is computed by Euclidean distance, denoted as $d_{0,n}$.

Only the top h WiFi clusters with the smallest distance will be searched in the next level, other WiFi clusters as well as their subbranches are skipped over. In the experiments, h is set to 2 which works well in our campus buildings. If the WiFi environment is complex, h can be larger such that more WiFi branches can be searched. In the following steps, branches are searched independently.

5.5.2. Orientation Pruning. Several hundred orientation samples can be collected when a user is taking photo. The query orientations O_0 can be modeled as a Gaussian distribution $N(\mu_0, \sigma_0)$. The similarity between O_0 and any orientation cluster can be computed by Equation 5.6. Top h orientation clusters with the smallest similarity to O_0 will be searched in the next level, other subbranches are skipped. As shown in Figure 5.1(b), the black branches indicates the search routes. Only parts of the leaf nodes need be searched, greatly increasing the efficiency.

5.5.3. Fine Visual Localization. Within each searched leaf node, the most similar reference image needs to be found. Algorithm 3 shows the search strategy within a leaf node. The best reference image is searched from top to down of multiple vectors. As the level goes deeper, the number of reference images to be compared becomes less and less, which decreases the computational cost. Meanwhile, the dimension of feature vector increases as the level goes deeper, images are compared with more and more local details.

If only one leaf node is searched, the candidate image selected from that leaf node is the final matched reference image. Otherwise, every searched leaf node gives one candidate image, we need to compare which candidate image is the best match. As shown in Figure 5.7, without loss of generality, only two candidate images are discussed here. A new visual codebook is built by concatenating the codebooks from the outputs of Algorithm 3. This new codebook is specialized to the two candidate images, therefore it is more discriminative than either of the single codebook. Then, feature vectors of the query image and candidate

images are calculated based on the new codebook. The candidate image that has the largest similarity with the query image is considered as the final matched image. The matched image's labeled position is reported as the user's location.

Algorithm 3 Search algorithm in a leaf node.

Notations:

- B: the totally number of reference images in a leaf node
- L: the number of clustering levels in a leaf node

Intput:

- Multiple feature vectors of query image: $V_{0,l}(l = 1...L)$;
- Multiple feature vectors of reference images in a leaf node: $V_{b,l}(b = 1...B, l = 1...L)$;
- Codebooks: $M_l(l = 1...L)$;
- A predefined threshold d_{thr}. It is set to 3 meters in this system;
- Comparison Pool (CP): all reference images in a leaf node;

Loops:

for $l = 1 : L$ **do**

- Compute the similarity between query image and images in CP: $S^l_{0,b} = \frac{V_{0,l} \cdot V_{b,l}}{|V_{0,l}||V_{b,l}|}$
- Compute the average similarity $\overline{S^l_{0,b}} = \frac{\sum_{b \in CP} S^l_{0,b}}{\sum_{b \in CP}}$
- Reference images satisfying $S^l_{0,b} < \overline{S^l_{0,b}}$ are deleted from CP
- Compute the maximum of pairwise spatial distance of images in CP, denoted as d_{max}

if $d_{max} < d_{thr}$ **then**

return M_l and reference images with the largest $S^l_{0,b}$

break

end if

end for

Output:

M_l and the candidate image which is the reference images with the largest $S^l_{0,b}$ in CP

5.6. EXPERIMENTS

To validate the effectiveness of our proposed indoor localization algorithm, we developed an application in the platform of Android Operation System to record the WiFi, inertial and visual signals. Figure 5.8(a) is a screenshot of the application with a simple in-

terface. This application has the capability of collecting reference signals as well as query signals.

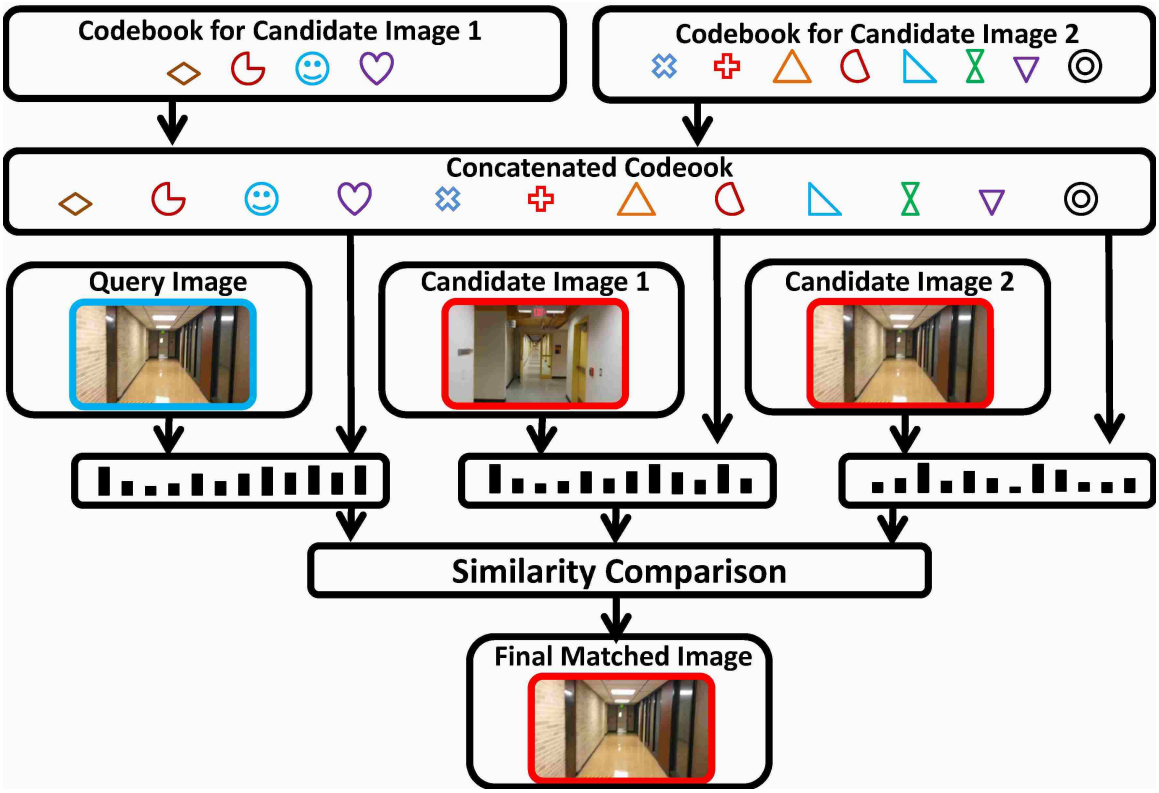

Figure 5.7. Determine final matched image from candidate images.

Figure 5.8(b) shows how we collect signals. WiFi, orientation and visual signals are collected by smartphones. A laser distance measurer is utilized to identify the actual location. When we collect reference signals, WiFi signals are collected uniformly and sparsely in the available regions of a building such as the hallway and public lounge. The distance of two adjacent WiFi collection positions is about 5 meters. As aforementioned, visual signals are recorded in videos. The frame rate of each video is 30fps. We keep walking with a constant speed when recording the videos. Thus, the position tagged to each frame can be interpolated by the positions of the start and end of each video recording.

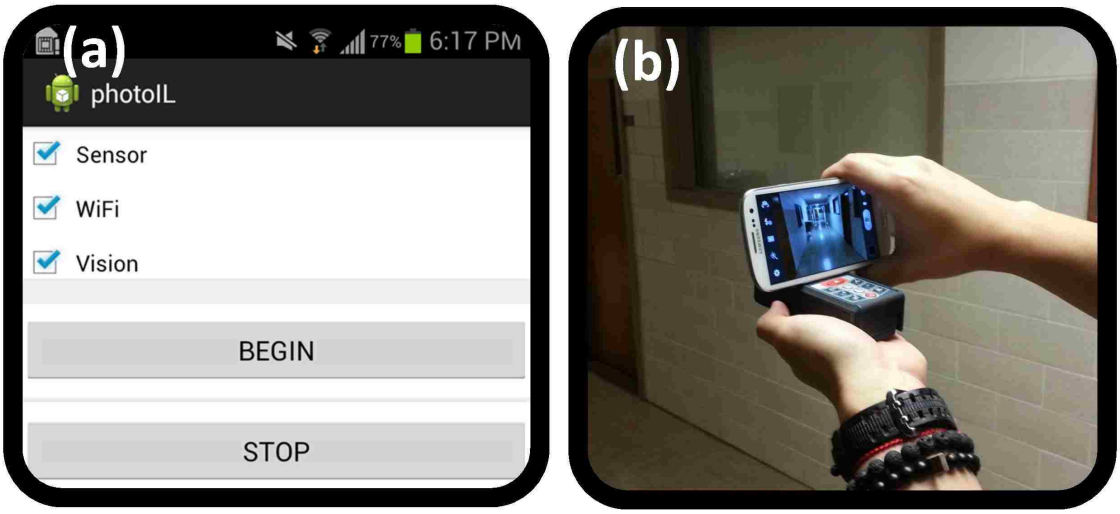

Figure 5.8. Experimental configuration. (a) The data collection App. (b) A laser distance measurer is used to identify the ground truth of a user's position.

5.6.1. Testing Environment. The proposed indoor localization algorithm is tested in 4 campus buildings whose floor plans are shown in Figure 5.9. Table 5.1 summarizes the information of signal trees of the 4 buildings which are used in experiment evaluations.

5.6.2. Comparison. Figure 5.10 shows some localization samples of our approach, which demonstrates the proposed localization algorithm is robust to crowded people, illumination change, scene changes and orientation shifts. Our proposed indoor localization algorithm is compared with three other approaches. (1) Multi-Level Image Description (MLID) method that only uses visual signals in the localization. (2) WiFi-based method. (3) The localization algorithm proposed by [55], which did not consider dense SIFT keypoints and multi-level feature vectors. The comparison is in terms of localization accuracy, localization efficiency and time used to build the reference database.

Localization Accuracy: Figure 5.11 summarizes the localization accuracy of 4 approaches in the 4 buildings. Our approach achieves the highest accuracy compared to the other 3 methods. The comparison of the approach described in [55] and MLID proposed

in this section shows it is more effective to describe images with multiple vectors, thus images' global and local information are both recoded and utilized for localization.

Table 5.1. Information about the signal trees of 4 buildings. NWB: Number of WiFi Branches; NOB: Number of Orientation Branches; NRI: Number of Reference Images; NQI: Number of Query Images.

Building No.	NWB	NOB	NRI	NQI
1	9	4	10117	241
2	6	4	4335	283
3	11	4	19313	202
4	12	4	18825	278

Localization Efficiency: Table 5.2 summarizes the comparison of the average time cost of online localization. During all the experiments, we notice that all query signals can be localized in less than 6.5 seconds with our method. The comparison of column 2 (Our signal tree method) and column 3 (Multi-Level Image Description, image-only method) proves that WiFi and orientation signals are capable to rule out impossible reference images and largely speed up the online localization.

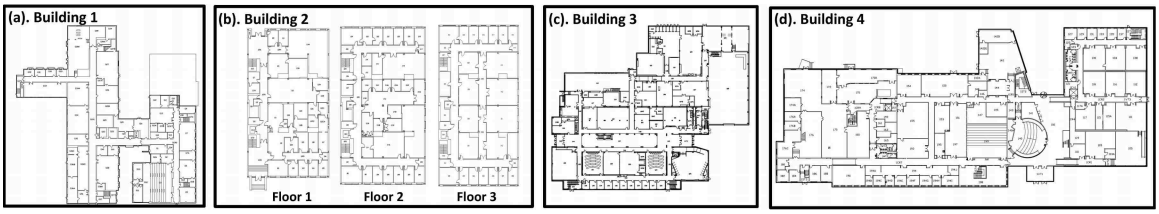

Figure 5.9. Floor plans of the test buildings.

Our method is slightly slower than [55]. We analyzed the average time cost of every step in our method and found out that computing dense SIFT keypoints which is not

required in [55] consumes 58.06% (about 3.35s) of the total time while searching the signal tree only takes 7.75% (about 0.45s) in our method. The SIFT key detection and extraction can be speeded up with GPU parallel computing. For example, it only needs 0.07 second to detect and extract SIFT keypoints from a 1024×768 image by a GPU [74]. We leave this as our future work. The WiFi-only method is the fastest, but its localization accuracy is very low (Figure 5.11).

Table 5.2. Average time used for localization (Seconds).

Building	Ours	MLID	[55]	WiFi
1	5.77	10.42	5.48	0.0094
2	5.63	9.63	5.22	0.006
3	5.80	10.79	5.11	0.0050
4	6.20	11.23	4.91	0.0014

Time Used to Build the Database: Table 5.3 summarizes the time cost of the 4 approaches to build the reference database. Except WiFi-only method, The proposed signal tree takes the least time to build the database (about one-tenth of the time cost of the image-only(MLID) method). Note that building or updating a reference database including thousands of images for a skyscraper can be a very time-consuming task. However, in our signal tree method, WiFi and orientation signals pre-cluster reference images into several leaf nodes, thus a complex problem is divided and conquered by small problems.

5.6.3. Discussion. From the experimental results, our method takes more time to build the database compared to WiFi-only method. In addition, our method takes more time for query compared to WiFi-only and the method proposed in [55], but the accuracy of our method is far better than all the other methods. Considering the evaluation metrics com-

prehensively, our proposed method is competitive to other methods and its effectiveness is multi-folds.

For a fingerprint based algorithm, it is time-consuming to collect a complete reference dataset to satisfy the high accuracy requirement. In this section, we just uniformly and sparsely collect WiFi fingerprints in a building. We collect reference images in the format of videos (subsection 5.3.2), which largely speed up the data collection and updating.

Table 5.3. Time used to build the database (Hours).

Building	Ours	MLID	[55]	WiFi
1	1.75	15.5	12	0.000866
2	2.5	23.75	22	0.001178
3	2	28	27.75	0.000948
4	2.25	27	26	0.00145

The proposed algorithm deals with the problem of WiFi environment change in two ways. As discussed in subsection 5.3.1, WiFi fingerprints are clustered into groups, thus our tolerance to WiFi environment change is getting higher. In the scenario that WiFi environment is largely changed, we can increase the number of search branch h described in subsection 5.4.1 to allow more WiFi branches to be searched.

5.7. SUMMARY FOR SECTION 5

In this section, we propose a novel signal-tree based indoor localization algorithm by fusing WiFi, inertial and visual signals. Our proposed algorithm is accurate as well as efficient because it makes full use of the advantages of three signals and finds the matched sig-

nal in a hierarchy manner. The proposed Multi-Level Image Description (MLID) method is very effective to describe and compare images with coarse-to-fine image descriptors.

Figure 5.10. Samples of our indoor localization. Top row: query images. Bottom row: matched reference images. (a). People occlusion. (b). Illumination changes. (c). Orientation shifts. (d). Scene slightly changes. (e). Low texture scene.

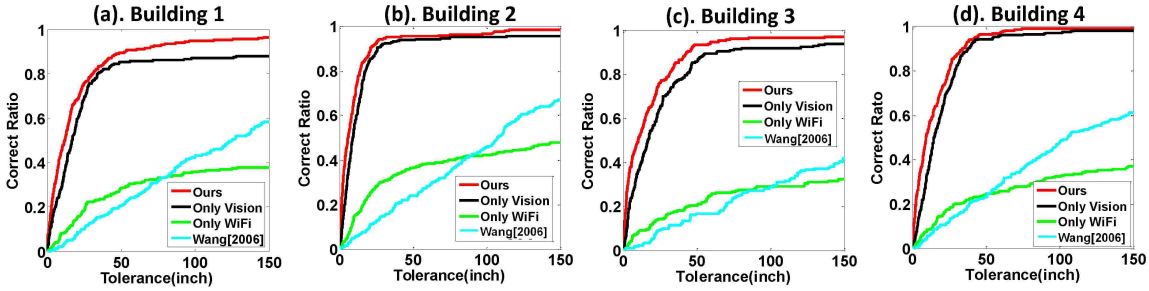

Figure 5.11. Accuracy comparison. Horizontal-axis is the distance between ground truth and estimated user's position. Vertical-axis is the proportion of query signals that have the accuracy within the distance labeled in horizontal-axis. MLID: Multi-Level Image Description method that only uses visual signals; Ours: signal tree (MLID + WiFi + Inertial sensor).

6. CONCLUSION AND FUTURE WORKS

6.1. CONCLUSION

In this paper, we investigate fusing visual and non-visual sensors for human tracking. To deeply understand the principle of non-visual sensors, we firstly study the application of wearable sensors to human physical activity recognition and human tracking. These two subtopics do not involve visual signal, but they are necessary technical foundation for the actual fusion. Then we fuse the visual signals from cameras and non-visual signals from IMU, WiFi model and so on to improve the accuracy of indoor localization and human tracking. Experimental results of each method are evaluated and shown in corresponding chapters.

6.2. FUTURE WORKS

The research work in this paper serves as a solid foundation for future investigation of fusion. Considering the proposed long-term tracking algorithm only utilizes one camera, we aim to update this tracking algorithm where multiple cameras (i.e., camera network) are adopted and we would like to seamlessly track the target. In addition, for the instant indoor localization, we plan to provide intelligent guidance to the user allowing a second localization in the extremely challenging cases when the first localization is not reliable.

BIBLIOGRAPHY

[1] Pierluigi Casale, Oriol Pujol, and Petia Radeva. Human activity recognition from accelerometer data using a wearable device. In *Pattern Recognition and Image Analysis*, pages 289–296. 2011.

[2] Oscar D Lara and Miguel A Labrador. A survey on human activity recognition using wearable sensors. *IEEE Communications Surveys & Tutorials*, 15(3):1192–1209, 2013.

[3] Muhammad Shoaib, Stephan Bosch, Ozlem Durmaz Incel, Hans Scholten, and Paul JM Havinga. Fusion of smartphone motion sensors for physical activity recognition. *Sensors*, 14(6):10146–10176, 2014.

[4] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN*, 2013.

[5] Nils Y Hammerla, Reuben Kirkham, Peter Andras, and Thomas Ploetz. On preserving statistical characteristics of accelerometry data using their empirical cumulative distribution. In *International Symposium on Wearable Computers*, pages 65–68, 2013.

[6] Wenchao Jiang and Zhaozheng Yin. Human activity recognition using wearable sensors by deep convolutional neural networks. In *the 23rd Annual ACM Conference on Multimedia Conference*, pages 1307–1310, 2015.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[8] Abdel-rahman Mohamed, Dong Yu, and Li Deng. Investigation of full-sequence training of deep belief networks for speech recognition. In *INTERSPEECH*, pages 2846–2849, 2010.

[9] Ming Zeng, Le T Nguyen, Bo Yu, Ole J Mengshoel, Jiang Zhu, Pang Wu, and Juyong Zhang. Convolutional neural networks for human activity recognition using mobile sensors. In *6th International Conference on Mobile Computing, Applications and Services*, pages 197–205, 2014.

[10] Stefan Duffner, Samuel Berlemont, Grégoire Lefebvre, and Christophe Garcia. 3d gesture classification with convolutional neural networks. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5432–5436, 2014.

[11] Earnest Paul Ijjina and C Krishna Mohan. One-shot periodic activity recognition using convolutional neural networks. In *International Conference on Machine Learning and Applications*, pages 388–391, 2014.

[12] Nicholas D Lane and Petko Georgiev. Can deep learning revolutionize mobile sensing? In *the 16th International Workshop on Mobile Computing Systems and Applications*, pages 117–122, 2015.

[13] Jian Bo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. In *the 24th International Joint Conference on Artificial Intelligence*, pages 25–31, 2015.

[14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[15] Mi Zhang and Alexander A Sawchuk. Usc-had: a daily activity dataset for ubiquitous activity recognition using wearable sensors. In *ACM Conference on Ubiquitous Computing*, pages 1036–1043, 2012.

[16] Jennifer R Kwapisz, Gary M Weiss, and Samuel A Moore. Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, 12(2):74–82, 2011.

[17] Ricardo Chavarriaga, Hesam Sagha, Alberto Calatroni, Sundara Tejaswi Digumarti, Gerhard Tröster, José del R Millán, and Daniel Roggen. The opportunity challenge: A benchmark database for on-body sensor-based activity recognition. *Pattern Recognition Letters*, 34(15):2033–2042, 2013.

[18] Thomas Stiefmeier, Daniel Roggen, Georg Ogris, Paul Lukowicz, and Gerhard Tröster. Wearable activity tracking in car manufacturing. *IEEE Pervasive Computing*, (2):42–50, 2008.

[19] Wenchao Jiang and Zhaozheng Yin. Human tracking using wearable sensors in the pocket. In *IEEE Global Conference on Signal and Information Processing*, pages 958–962, 2015.

[20] Nisarg Kothari, Balajee Kannan, Evan D Glasgwow, and M Bernardine Dias. Robust indoor localization on a commercial smart phone. *Procedia Computer Science*, pages 1114–1120, 2012.

[21] Zhuoling Xiao, Hongkai Wen, Andrew Markham, and Niki Trigoni. Robust pedestrian dead reckoning (r-pdr) for arbitrary mobile device placement. In *International Conference on Indoor Positioning and Indoor Navigation*, volume 27, 2014.

[22] Liu Meng, Sui Hulin, Wang Jun, and Ma Qingbo. Firefighter indoor dead-reckoning system with inertial sensors. *International Journal of Digital Content Technology and Its Applications*, 5(12):259–265, 2011.

[23] Ya Tian, Hongxing Wei, and Jindong Tan. An adaptive-gain complementary filter for real-time human motion tracking with marg sensors in free-living environments. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 21(2):254–264, 2013.

[24] Seon-Woo Lee and Kenji Mase. Activity and location recognition using wearable sensors. *IEEE pervasive computing*, 1(3):24–32, 2002.

[25] Ulrich Steinhoff and Bernt Schiele. Dead reckoning from the pocket-an experimental study. In *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 162–170, 2010.

[26] Qiang Wang, Yan Liu, and Juan Chen. Accurate indoor tracking using a mobile phone and non-overlapping camera sensor networks. In *IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, pages 2022–2027, 2012.

[27] M Schussel and Florian Pregizer. A comparison of sensorfusion methods for localization on mobile phones. In *IEEE Third International Conference onãĂĂConsumer Electronics and Berlin*, pages 397–401, 2013.

[28] Yunye Jin, Hong-Song Toh, Wee-Seng Soh, and Wai-Choong Wong. A robust dead-reckoning pedestrian tracking system with low cost sensors. In *2011 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 222–230, 2011.

[29] Haitao Bao and Wai-Choong Wong. Improved pca based step direction estimation for dead-reckoning localization. In *IEEEãĂĂInternational Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pages 325–331, 2013.

[30] Jiuchao Qian, Jiabin Ma, Rendong Ying, Peilin Liu, and Ling Pei. An improved indoor localization method using smartphone inertial sensors. In *IEEE International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pages 1–7, 2013.

[31] Aleksandr Mikov, Alex Moschevikin, Alexander Fedorov, and Axel Sikora. A localization system using inertial measurement units from wireless commercial hand-held devices. In *IEEE International Conference on Indoor Positioning and Indoor Navigation*, pages 1–7, 2013.

[32] Agata Brajdic and Robert Harle. Walk detection and step counting on unconstrained smartphones. In *Proceedings of the ACM International Joint Conference on Pervasive and ubiquitous computing*, pages 225–234, 2013.

[33] Sebastian OH Madgwick, Andrew JL Harrison, and Ravi Vaidyanathan. Estimation of imu and marg orientation using a gradient descent algorithm. In *IEEE International Conference on Rehabilitation Robotics*, pages 1–7, 2011.

[34] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. People-tracking-by-detection and people-detection-by-tracking. In *Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[35] Jamie Sherrah. Occluded pedestrian tracking using body-part tracklets. In *IEEE International Conference on Digital Image Computing: Techniques and Applications*, pages 314–319, 2010.

[36] Wenchao Jiang and Zhaozheng Yin. Combining passive visual cameras and active imu sensors to track cooperative people. In *18th International Conference on Information Fusion*, pages 1338–1345, 2015.

[37] Piotr Dollar, Christian Wojek, Bernt Schiele, and Pietro Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):743–761, 2012.

[38] Markus Enzweiler and Darieu M Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2179–2195, 2009.

[39] Bo Wu and Ram Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. *International Journal of Computer Vision*, 75(2):247–266, 2007.

[40] Nicolas Papadakis and Aurelie Bugeau. Tracking with occlusions via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):144–157, 2011.

[41] Siyu Tang, Mykhaylo Andriluka, and Bernt Schiele. Detection and tracking of occluded people. *International Journal of Computer Vision*, 110(1):58–69, 2014.

[42] Bao Hong Yuan, De Xiang Zhang, Kui Fu, and Ling Jun Zhang. Video tracking of human with occlusion based on meanshift and kalman filter. In *Applied Mechanics and Materials*, volume 380, pages 3672–3677, 2013.

[43] Mahmoud Mirabi and Shahram Javadi. People tracking in outdoor environment using kalman filter. In *International Conference on Intelligent Systems, Modelling and Simulation*, pages 303–307, 2012.

[44] BZ De Villiers, WA Clarke, and PE Robinson. Mean shift object tracking with occlusion handling. IAPR, 2012.

[45] Jinfeng Yan, Qiang Ling, Yicheng Zhang, Feng Li, and Feng Zhao. A novel occlusion-adaptive multi-object tracking method for road surveillance applications. In *IEEE Chinese Control Conference*, pages 3547–3551, 2013.

[46] Yang Hua, Karteek Alahari, and Cordelia Schmid. Occlusion and motion reasoning for long-term tracking. In *ECCV*, pages 172–187. 2014.

[47] Masakatsu Kourogi and Takeshi Kurata. Personal positioning based on walking locomotion analysis with self-contained sensors and a wearable camera. In *International Symposium on Mixed and Augmented Reality*, page 103, 2003.

[48] Raúl Feliz Alonso, Eduardo Zalama Casanova, and Jaime Gómez García-Bermejo. Pedestrian tracking using inertial sensors. *Journal of Physical Agents*, 3(1):35–42, 2009.

[49] Kai Kunze, Paul Lukowicz, Kurt Partridge, and Bo Begole. Which way am i facing: Inferring horizontal device orientation from an accelerometer signal. In *International Symposium on Wearable Computers*, pages 149–150, 2009.

[50] Guillem Casas Barcelo, Ghazaleh Panahandeh, and Magnus Jansson. Image-based floor segmentation in visual inertial navigation. In *Instrumentation and Measurement Technology Conference*, pages 1402–1407, 2013.

[51] Ghazaleh Panahandeh, Dave Zachariah, and Magnus Jansson. Exploiting ground plane constraints for visual-inertial navigation. In *Position Location and Navigation Symposium*, pages 527–534, 2012.

[52] Ghazaleh Panahandeh, Magnus Jansson, and Seth Hutchinson. Imu-camera data fusion: Horizontal plane observation with explicit outlier rejection. In *International Conference on Indoor Positioning and Indoor Navigation*, pages 1–9, 2013.

[53] Chris Hide, Tom Botterill, and Marcus Andreotti. Low cost vision-aided imu for pedestrian navigation. In *Ubiquitous Positioning Indoor Navigation and Location Based Service*, pages 1–7, 2010.

[54] Wenchao Jiang and Zhaozheng Yin. Indoor localization by signal fusion. In *18th International Conference on Information Fusion*, 2015.

[55] Junqiu Wang, Hongbin Zha, and Roberto Cipolla. Coarse-to-fine vision-based localization by indexing scale-invariant features. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 36(2):413–422, 2006.

[56] Rainer Mautz and Sebastian Tilch. Survey of optical indoor positioning systems. In *Indoor Positioning and Indoor Navigation*, pages 1–7, 2011.

[57] Jason Zhi Liang, Nicholas Corso, Eric Turner, and Avideh Zakhor. Image based localization in indoor environments. In *International Conference on Computing for Geospatial Research and Application*, pages 70–75, 2013.

[58] Hong Liu, Xiaojia Yu, and Haitao Yu. Combining color histogram and gradient orientation histogram for vision based global localization. In *International Conference on Systems, Man and Cybernetics*, pages 4043–4047, 2009.

[59] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *IEEE computer society conference on Computer vision and pattern recognition*, volume 2, pages 2161–2168, 2006.

[60] Joydeep Biswas and Manuela M Veloso. Wifi localization and navigation for autonomous indoor mobile robots. 2010.

[61] Krishna Chintalapudi, Anand Padmanabha Iyer, and Venkata N Padmanabhan. Indoor localization without the pain. In *international conference on Mobile computing and networking*, pages 173–184, 2010.

[62] Ionut Constandache, Romit Roy Choudhury, and Injong Rhee. Towards mobile phone localization without war-driving. In *Infocom*, pages 1–9, 2010.

[63] Shohei Koide and Masami Kato. 3-d human navigation system considering various transition preferences. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 1, pages 859–864, 2005.

[64] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *IEEE International Conference on Computer Vision*, pages 1470–1477, 2003.

[65] Ming-Chao Chiang, Chun-Wei Tsai, and Chu-Sing Yang. A time-efficient pattern reduction algorithm for k-means clustering. *Information Sciences*, 181(4):716–731, 2011.

[66] Hamid Sadeghi, Shahrokh Valaee, and Shahram Shirani. A weighted knn epipolar geometry-based approach for vision-based indoor localization using smartphone cameras. In *Sensor Array and Multichannel Signal Processing Workshop*, pages 37–40, 2014.

[67] Shizhe Zhang, Yongping Xiong, Jian Ma, Zheng Song, and Wendong Wang. Indoor location based on independent sensors and wifi. In *International Conference on Computer Science and Network Technology*, volume 4, pages 2640–2643, 2011.

[68] Anshul Rai, Krishna Kant Chintalapudi, Venkata N Padmanabhan, and Rijurekha Sen. Zee: zero-effort crowdsourcing for indoor localization. In *international conference on Mobile computing and networking*, pages 293–304, 2012.

[69] Yin Chen, Dimitrios Lymberopoulos, Jie Liu, and Bodhi Priyantha. Fm-based indoor localization. In *international conference on Mobile systems, applications, and services*, pages 169–182, 2012.

[70] Martin Azizyan, Ionut Constandache, and Romit Roy Choudhury. Surroundsense: mobile phone localization via ambience fingerprinting. In *international conference on Mobile computing and networking*, pages 261–272, 2009.

[71] Eladio Martin, Oriol Vinyals, Gerald Friedland, and Ruzena Bajcsy. Precise indoor localization using smart phones. In *international conference on Multimedia*, pages 787–790, 2010.

[72] Rui Xu and Don Wunsch. *Clustering*, volume 10. IEEE Press, 2008.

[73] Martin Werner, Moritz Kessel, and Chadly Marouane. Indoor positioning using smartphone camera. In *Indoor Positioning and Indoor Navigation*, pages 1–6, 2011.

[74] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M Seitz. Multicore bundle adjustment. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3057–3064, 2011.

VITA

Wenchao Jiang was born in Chongqing, China. In July 2012, he received his B.S. in University of Electronic Science and Technology of China. Then, he joined the Computer Science Department of Missouri University of Science and Technology (formerly the University of Missouri-Rolla) as a Ph.D. student in February, 2013. In May, 2017, he received his Ph.D. degree in the Computer Science Department of Missouri University of Science and Technology.