POLITECNICO DI MILANO

FACOLTA’ DI INGEGNERIA DELL’INFORMAZIONE

Corso di Laurea Magistrale in INGEGNERIA DELL'AUTOMAZIONE

Human walking speed estimation based on a surveillance camera system for safety assessment in a robotic cell

Relatore:	Prof. Paolo Rocco
Correlatori:	Dr. Luca Bascetta

Tesi di Laurea Magistrale di:

Leifeng WEI

Matr. 763635

Index

ABSTRACT

The co-work of human and industrial robots in future manufacturing industries is predicted to be the promising mode of production, which can exploit the highest flexibility of the manufacturing system. A main problem involved in this human-robotic interaction scenario is the tracking of human locomotion. Inspired by the research project ROSETTA, an algorithm to estimate the human walking speed, based on a surveillance camera system, is proposed in this thesis. The core of the algorithm is an extended Kalman filter modified to be adaptable to the human detection and tracking (HDT) system developed by ROSETTA. The main functionality of the algorithm is, by integrating the kinematic model of human walking with the HDT system, to estimate the velocity (including the direction) of the tracked human. A piece-wise filtering with velocity threshold and an information fusion technique are innovated in this thesis to improve the algorithm. Off-line data was utilized to validate the algorithm.

SOMMARIO

La collaborazione di operatori umani e robot industriali nell'industria manifatturiera è una modalità di produzione promettente, che può sfruttare la più alta flessibilità del sistema manifatturiero. Un importante problema connesso a questo scenario di interazione uomo-robot è il riconoscimento e l'inseguimento della camminata delle persone. Basandosi sui risultati del progetto ROSETTA, in questa tesi si propone un algoritmo per stimare la velocità di un essere umano nell'atto di camminare, ripreso da un sistema di telecamere di sorveglianza. Il nucleo dell'algoritmo è un filtro di Kalman esteso modificato per adattarsi al sistema di rilevazione e inseguimento (HDT) sviluppato in ROSETTA. La principale funzionalità dell'algoritmo consiste, integrando il modello cinematico della camminata umana con il sistema HDT, nello stimare la velocità (inclusa la direzione) della persona inquadrata dalle telecamere. Nella tesi si propongono un filtro con soglia sulla velocità minima e una tecnica di fusione dell'informazione per migliorare l'algoritmo, validato offline con dati ottenuti dal sistema HDT.

Chapter 1 Introduction

1.1 Overview

In the modern massive production, goods are produced in a more customized way to satisfy the markets. A demand of quicker updating and more variety products prefigures the future production model, which requires manufacturers to have a more flexible and faster manufacturing system. The industrial robotics is always a good choice of the automation methods for the factories. Nevertheless, human workers are still the most flexible productive forces. To achieve both the productivity and the flexibility, a co-work of industrial robots and human workers is a promising way (Figure 1.1).

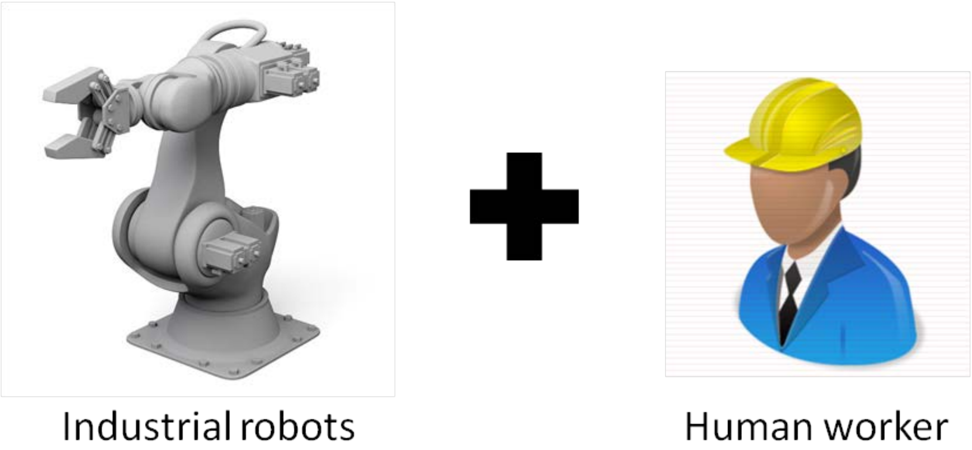

Figure 1.1 Promising manufacturing style, co-existence of industrial robots and human worker

A European Union funded project, “ROSETTA” (the acronym for “RObot control for Skilled ExecuTion of Tasks in natural interaction with humans; based on Autonomy, cumulative knowledge and learning”), aims at supporting industry through developing such technologies that facilitate to utilize and integrate industrial robots into otherwise manual assembly lines.

The project addresses the challenges in developing methods and tools to engineer and program robot stations in more intuitive ways, while still staying less specific to the installation. This requires a higher level of autonomy from the robot in terms of adapting to a changing environment and interacting with the human workers (Figure 1.2).

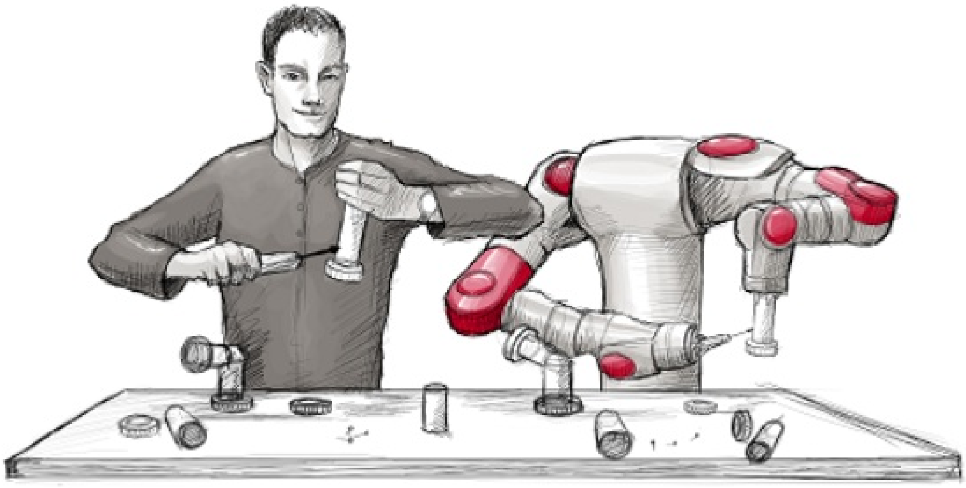

Figure 1.2 A possible preview of a human worker co-work with an intrinsically safe concept robot

In the ROSETTA concept the robot is not necessarily separated from humans by any physical safety fences or barriers. This implies that the robot needs to be safe towards the human either by being intrinsically safe or by employing active safety systems (Figure 1.2).

Regarded as a very important objective of ROSETTA, a safety system is being developed. It exploits the knowledge acquired from the sensor system developed in the project and improving skills for both co-existence and co-operation of robots and humans. This will allow safe intentional interactions while protecting from dangerous collisions. To endow the system with such capabilities, a “safety controller” is established.

The conception of this safety controller is to track the human with a visual tracking system based on commercial surveillance cameras, which are easy to be deployed in any robotic cell, and to predict the locomotion of the tracked human with the information provided by the visual tracking system. Then, by implementing a safety strategy developed in ROSETTA, the potential collision should be avoided with the prediction of human locomotion and the motion plan from the robot side.

Politecnico di Milano participates in the ROSETTA project. As one of the partners, the Department of Electronics and Information of Politecnico di Milano is assigned to work on the safe human-robot interaction control. An experimental environment is set up in one of the department laboratory. It imitates the co-existence working conditions which are specified in the ROSETTA project (Figure 1.3).

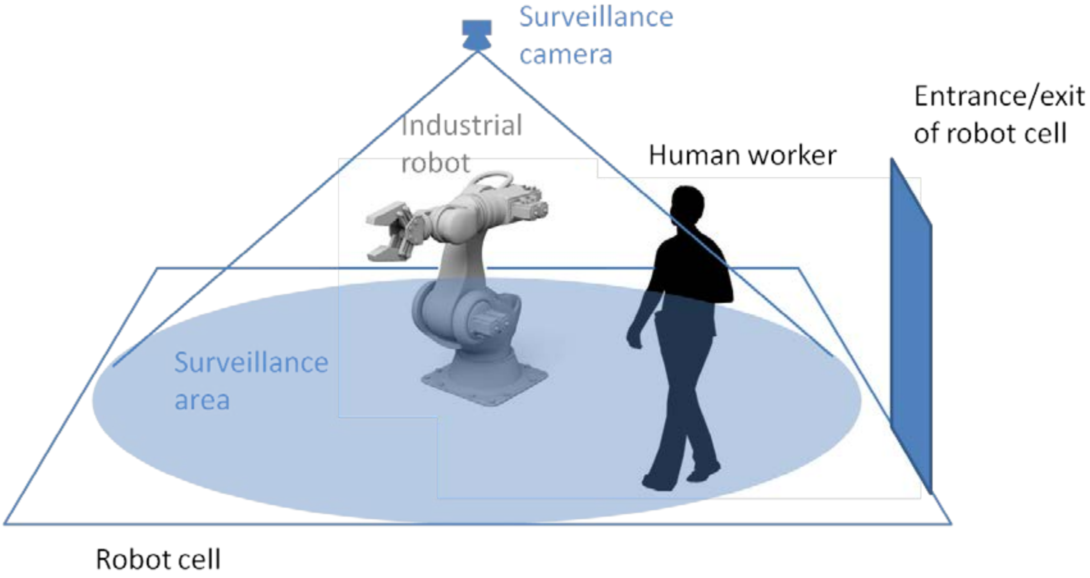

Figure 1.3 Experimental environment set up to simulates the co-existence of robot and human

The conceptual structure of the safety controller is composed of three main modules: Human Detection and Intention Estimation (HD&IE), Safety Watcher and Interaction Override Control (Figure 1.4).

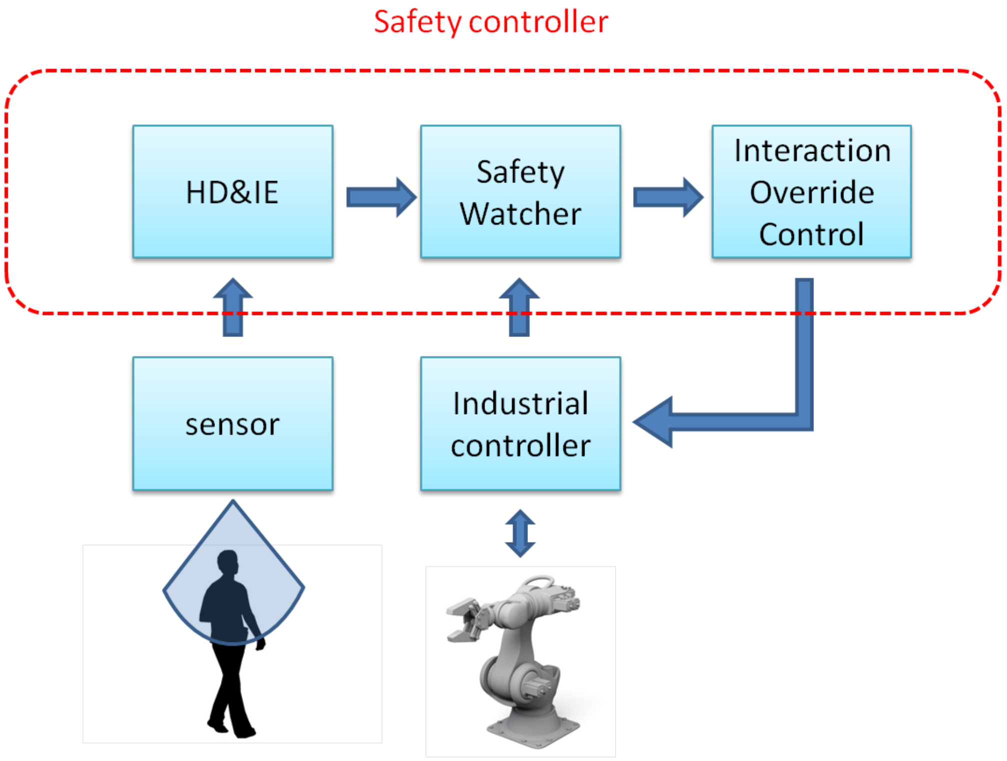

Figure 1.4 Conceptual structure of the safety controller

The control loop of the safety controller is illustrated in (Figure 1.4). The "Human Detection and Intention Estimation" module acquires the motional information of human captured by the sensor, which in the ROSETTA is commercial surveillance cameras mounted on the ceiling of the robot cell. The images sampled by the camera will be analyzed by a computer image processing software, which is developed for and embedded in the "HD&IE" module. The "HD&IE" module also contains other sorts of software supporting this human tracking system, which are going to be introduced with details in Chapter 3. The output of the "HD&IE" is the possible trajectories predicted by the module. The "Safety Watcher" module will integrate it with the information provided by the industrial control unit of the robot. Based on the locomotion-map of the human and the robot, the "Safety Watcher" will assess the risk of dangerous collisions between the human and the robot. Then the "Safety Watcher" will tell the "Interaction Override Control", whether an action should be taken to avoid the potential danger. In case a potential dangerous collision is detected, the "Interaction Override Control" will communicate with the industrial controller of the robot, and override the preconcert path plan of the robot with an evasive action.

1.2 Objective of the thesis

The “Safety Controller” conceived in the “ROSETTA” project needs the information of the human locomotion to evaluate the risk of the collision against the robot. It is critical, that the intension of the tracked human shall be estimated and predicted as soon and reliable as possible. To achieve this requirement, a specific human detecting and tracking system is established.

This thesis addresses the estimation of the human walking speed as the objective, which is an important feature of the human trajectory. It will be integrated in the human detecting and tracking system. Also as a primitive passive safety strategy, when an impact of human against the robot is predicted, the relative velocity of the human and the robot is one of the crucial criteria to assess the risk of injury [1].

1.3 Results achieved

Based on the simplified kinematic model of walking human, a modified version is proposed to reinforce the similarity of the model with the velocity-curvature dependence. Then an extended Kalman filter is applied to integrate the model with the “front end” filter, a particle filter. The algorithm is coded in MatLab®, and validated by off-line data acquired by the human detection and tracking system of the project ROSETTA.

1.4 Organization of documents

List of Figures

Chapter 2 Model of human walking

For a Gaussian-Bayesian filter, a precise state space mathematical model is preferred. However, free walk of human is a very complex and highly unpredictable locomotion, and it is not easy to model. In this chapter, a 6-states model of human walking is conceived, and it will be implemented in the extended Kalman filter.

2.1 5-states model

In this thesis, we are only interested in the trajectory of the human body, walking on the ground. Apart from neuro-ergonomic aspects, and also neglecting the limbs of human, the trunk of human body is simplified to a rectangular box with a proper size respect to the human itself. The position of the human is approximately represented by the geometric center of the rectangular box (Figure 2.1). Thus the trajectory of a walking human can be described in a Cartesian coordinate frame, which is fixed with the area we are interested in.

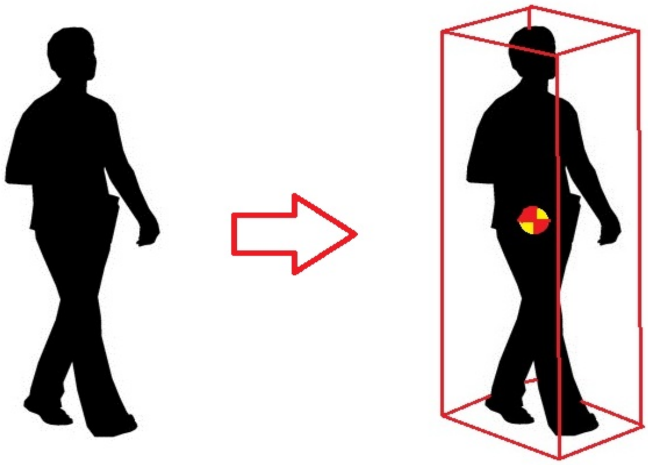

Figure 2.1 A walking human represented by a rectangular box.

It is assumed that the human is walking in an area with flat terrain, and we are not interested in the vertical position of the human body. The Cartesian coordinate frame can actually be set as a two-dimensional frame, with only horizontal axes “x” and “y”. The human’s center of gravity is projected on the ground (“x-y” plane) as the geometric center of the rectangle at the bottom of the box, named as point “P”. Then, a human walking trajectory is defined as the path followed by the point “P” through the ground plane (Figure 2.2).

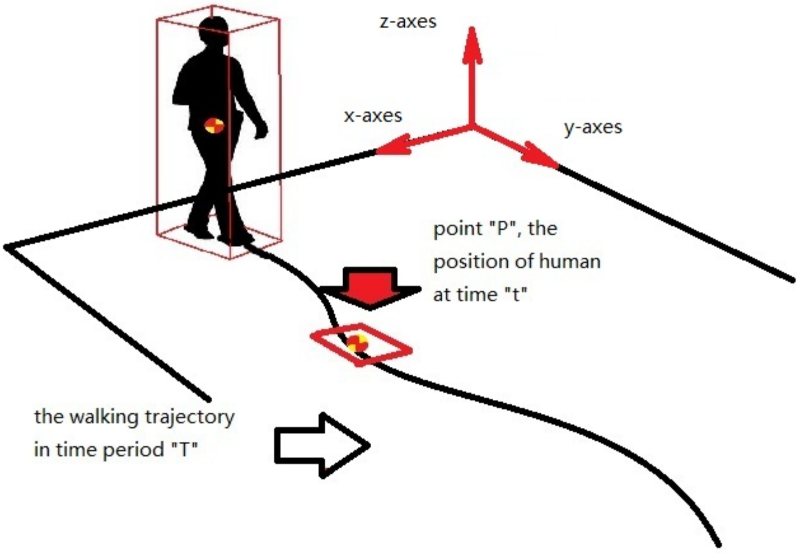

Figure 2.2 Human walking trajectory on the ground plane

Based on the previous representation of the human body and its positioning, some main kinamtic features can be observed intuitively:

- x(t) and y(t) are the human position coordinates at time “t” on the Cartesian frame fix on the ground.
- v(t) is defined as the magnitude of the horizontal linear velocity at time “t” of human’s center of gravity.
- θ(t) is defined as the horizontal orientation at time “t” of human walking direction, it is also the angle between the tangent of the trajectory and x-axes. θ is positive when turning from x-axes to y-axes anti-clockwise, which follows the right-hand law.

Remark: x(t) and y(t) are the coordinates of the point "P" at time t. The box representing the model of the human inherits all these kinamtic features (Figure 2.3).

The main kinematic equations are as followed:

$$\begin{cases}\dot{x}(t) = v(t) \cdot \cos(\theta(t)) \\ \dot{y}(t) = v(t) \cdot \sin(\theta(t))\end{cases} \qquad (2\text{-}1)$$

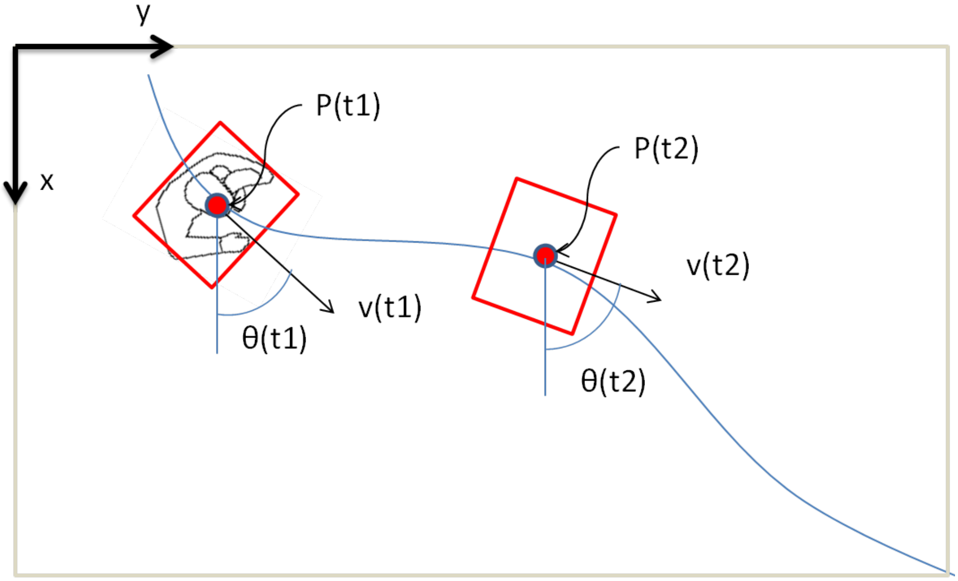

Figure 2.3 Top view of human walking trajectory with velocity and angle

Obviously, these elementary kinematic states are quite insufficient to construct the model of a walking human. To distinguish the walking model from other locomotion, we have to put on some constraints to describe the behavior of a walking human. The neuroscience approaches in modeling human motion pointed out the critical role of the curvature. To prevent curvature discontinuities, we propose to make the curvature a variable of the system [2]:

$$k = \frac{1}{R} \qquad (2\text{-}2)$$

R is the radius of curvature (m), and k is defined as the curvature (1/m), i.e. the reciprocal of the radius of curvature. According to the geometric relationships, we can establish a 5-states kinematic model of human walking:

$$\begin{cases} \dot{x}(t) = v(t) \cdot \cos(\theta(t)) \\ \dot{y}(t) = v(t) \cdot \sin(\theta(t)) \\ \dot{\theta}(t) = k(t) \cdot v(t) \\ \dot{k}(t) = \sigma(t) \\ \dot{v}(t) = a(t) \end{cases} \tag{2-3}$$

σ is the derivative of the curvature and a is the linear acceleration. With this 5-states model, the human walking model can be simulated as a monocycle [2]. This 5-states “monocycle” model constraints the point “P” (position of human) which should move along a trajectory with curvature having 1st-order continuity. In a more intuitive way, it can be interpreted as follows: when a human is walking normally, he/she can not jump sideways.

In this 5-states model, σ (the derivative of the curvature) and a (the linear acceleration) are considered as two inputs. But, they are almost impossible to be predicted or to be estimated. What we can do, is setting these two variables as normally distributed random variables. As the choice in this thesis, “σ” and “a” will be treated as process noise of the model.

2.2 Velocity-curvature dependence

As mentioned in section 2.1, the 5-states model (2-3) described the walking human as a monocycle, featured by

$$\dot{\theta}(t) = k(t) \cdot v(t) \tag{2-4}$$

However, it is easy to get a paradox in equation (2-4): when the linear speed is getting lower, the steering speed $\dot{\theta}$ is getting lower too, if we assume “k” is kept constant. On the contrary, the common situation is, when a human is walking at a lower speed, he/she is easier to redirect his/her walking. This defect of the model will lead to greater covariance of the system when the walking speed is low. And will keep affecting the filter to make results under-estimated. And for the assumption of k remaining constant, the clarification is as follows: in the 5-states kinematic model (2-3), the only constraint on it is its continuity. The linear velocity v and the curvature k are independent. The variation of k is driven just by its derivative σ, which can be considered as an input. So k is just a proportion between $\dot{\theta}$ and v, i.e. $\dot{\theta}(t) \propto v(t)$.

In order to overcome this problem, some improvements should be conceived. Previous neuroscience research [3] sheds a light upon a possible solution: the dependence of curvature on velocity.

In the literature, some curved locomotor paths are studied with the hypothesis that, also during locomotion, movement obey the so-called "two-thirds power law" stating that the angular velocity of the end effector (here the entire body) is proportional to the two-thirds root of its curvature ($\omega = k \cdot R^{2/3}$). For self-contentment in this thesis, a brief introduction of "power law" of velocity-curvature will be developed as followed.

The relationship between geometrical (radius of curvature) and kinematic (tangential velocity) properties of movement trajectories has been studied in handwriting and drawing in a 2D or 3D space. A specific rule had been found, which can be expressed as:

$$v(t) = \alpha \cdot R(t)^{\beta} \tag{2-5}$$

where v is the tangential velocity, R is the radius of curvature, β is a constant exponent, and α is the "velocity gain factor", which is a piecewise constant. For complex trajectories, the velocity gain factor α is constant within some identifiable geometrical segments and discontinuously changes at the transition points that separate two consecutive segments. This observation, of identifiable units of motor action, suggests a segmentation strategy for the description of complex movements. Although α changes from one segment to another, the β exponent was empirically found to be near 1/3 for all parts of the trajectory (derived from "two-thirds power law"). Thus the one-third power law provides a quantitative description of both movement generation and segmentation [4]. In the work by S. Vieilledent et al. [5], the power law was applied to 2D human locomotor trajectories on elliptical paths. They tested several subjects with an optoelectronic video motion capture device. Each subject followed the shape of the proposed pathways (Figure 2.4).

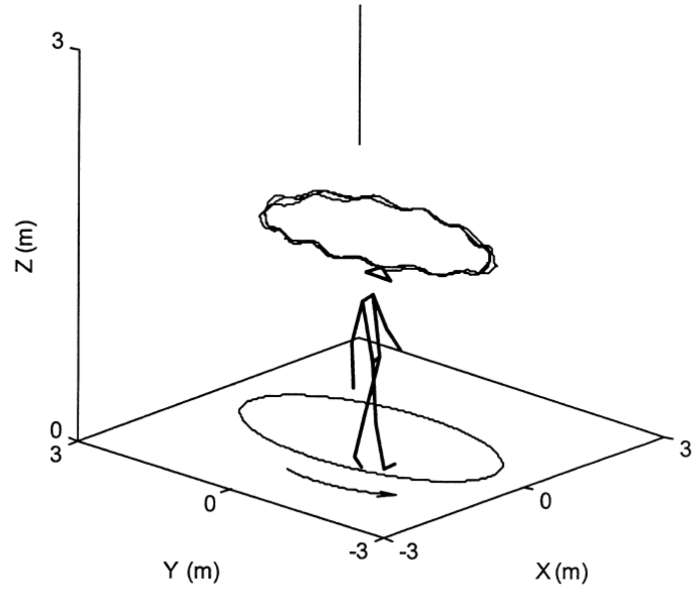

Figure 2.4 A typical subject is walking alone the ellipse shaped path drawn on the ground

By varying the dimension and shape of the paths, a set of data including velocity and curvature were recorded at a sample rate (60 Hz). To analyze the acquired data, a “log” transformation of the “power law” equation (2-5) was executed:

$$\log(\mathrm{v}(\mathrm{t})) = \log(\alpha) + \beta \cdot \log(\mathrm{R}(\mathrm{t})) \qquad (2\text{-}6)$$

Then correlation of velocity and curvature can be analyzed statistically by plotting them in log-log scales (Figure 2.5). Some observations were summarized from the plots in the literature:

- The relationship between tangential velocity and radius of curvature of a locomotor trajectory along specific elliptical pathways is presented.
- The shape of pathways has a significant effect on the correlation coefficient.

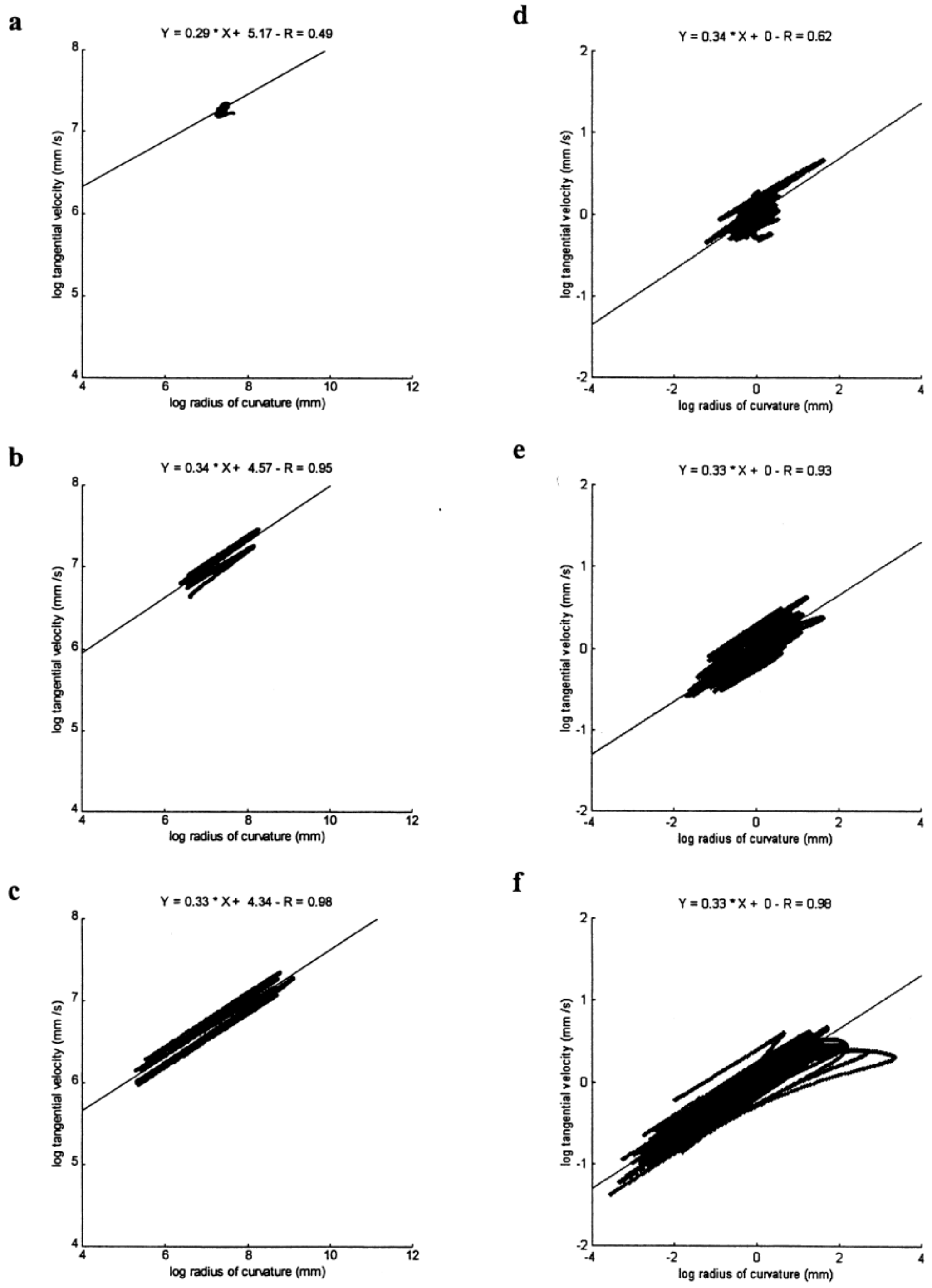

Figure 2.5 Log-log plots of velocity-curvature by varying the shape of the paths with regression line, "R" is the correlation coefficient in these plots.

Although violations of the "one-third power law" are reported in some papers (eg. [6]), this empirical hypothesis is proved to be practically helpful in this thesis in the following chapters.

Concerning the violations of the "two-third power law" and its shape-sensitivity, the authors of previously presented literatures did not provide precise values of the parameters in the equation (2-5). Only some approximate estimation can be found in their experiments. To adapt this empirical hypothesis to the walking human model of this thesis, some approximation also should be made.

Another issue shall be considered: when human is changing direction while walking, his/her tangential velocity should be constrained by the curvature he/she follows, but not vice versa. When human is slowing down the velocity, it does not predestinate a presence of steering. More intuitively speaking, when a vehicle is going to make a turn, it shall slow down its velocity. When a vehicle is slowing down, it may not change its direction.

2.3 6-states model with velocity threshold

As previously discussed, to model the dependence of velocity-curvature, a non-deterministic variable should be conceived. This variable should embody the willingness of the steering while walking at a certain speed. The original "power law" equation (2-5) can be interpreted as follow:

$$\mathrm{k(t)} = \ \upalpha_{\mathrm{rev}} \cdot \mathrm{v(t)}^{\upbeta_{\mathrm{rev}}} \cdot \mathrm{st(t)} \qquad (2\text{-}7)$$

k is the curvature, v is the tangential velocity, α_{rev} and β_{rev} are the parameters derived from equation (2-5), and "*st*" is the so-called "steering willingness". "*st*" drives the locomotion of steering like an input. The curvature of steering is constrained by the instantaneous velocity in the relation of "$\alpha_{rev} \cdot v(t)^{\beta_{rev}}$", which is considered in this thesis as the maximum curvature human can achieve at the certain linear velocity v. Thus "*st*" is suggested to be a normally distributed random variable on the interval (-1,1).

To implement this additional constraint on curvature to the state space model, the equation (2-7) should be differentiated with respect to time.

$$\frac{dk}{dt} = \alpha_{rev} \cdot \beta_{rev} \cdot v(t)^{\beta_{rev}-1} \cdot a(t) \cdot st(t) \tag{2-8}$$

a is the linear acceleration, which is assumed previously to be a normally distributed random variable. Replacing the curvature equation in 5-states model (2-3) with the constrained equation of curvature (2-8), we have the new 5-states model (2-9).

$$\begin{cases} \dot{x}(t) = v(t) \cdot \cos(\theta(t)) \\ \dot{y}(t) = v(t) \cdot \sin(\theta(t)) \\ \dot{\theta}(t) = k(t) \cdot v(t) \\ \dot{k}(t) = \alpha_{rev} \cdot \beta_{rev} \cdot v(t)^{\beta_{rev}-1} \cdot a(t) \cdot st(t) \\ \dot{v}(t) = a(t) \end{cases} \tag{2-9}$$

In this new 5-states model, we have “*a(t) • st(t)*”, a product of two random variables [7]. Implementing the theorem provided by the literature [7], the distribution of the product would be similar to the one reported in (Figure 2.6).

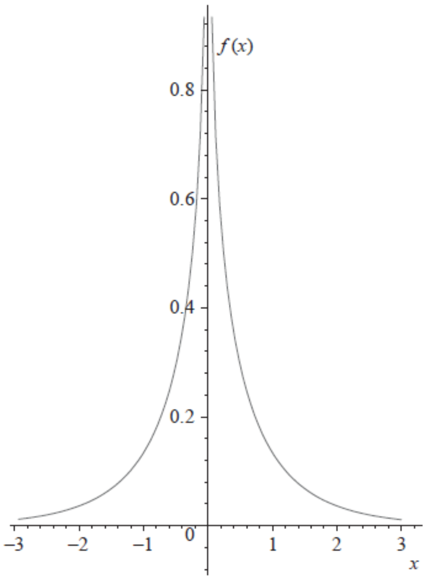

Figure 2.6 The PDF of V = XY for X~N(0,1) and Y~N(0,1)

For the variance of *a* and *st Var(a·st)*, the result can be computed according to the equation in [8]. As a matter of fact, the product of these two random variables can be approximated by a new random variable with suitable mean value and variance. In this thesis, the new random variable is called “*cn*”. To put this random variable into the system, we have to introduce it as a new state, which is a pure process noise. Thus a 6-states model (2-10) is proposed.

$$\begin{cases} \dot{x}(t) = v(t) \cdot \cos(\theta(t)) \\ \dot{y}(t) = v(t) \cdot \sin(\theta(t)) \\ \dot{\theta}(t) = k(t) \cdot v(t) \\ \dot{k}(t) = \alpha_{rev} \cdot \beta_{rev} \cdot v(t)^{\beta_{rev}-1} \cdot cn(t) \\ \dot{c}n(t) = w(t) \\ \dot{v}(t) = a(t) \end{cases} \tag{2-10}$$

w(t)~N(0,W1) is a normal distributed random variable, *a(t) ~N(0,W2)* is also a normal distributed random variable, and assuming *w(t)*$\perp$*a(t)*.

The new 6-states model is featured by the constrained curvature based on the velocity-curvature dependence. The derived equation (2-7) provides the possibility to endow the curvature with a dynamic maximum variance depending on the linear velocity *v*, instead of a constant one. The functional mechanism of the dynamic maximum variance is illustrated in the following.

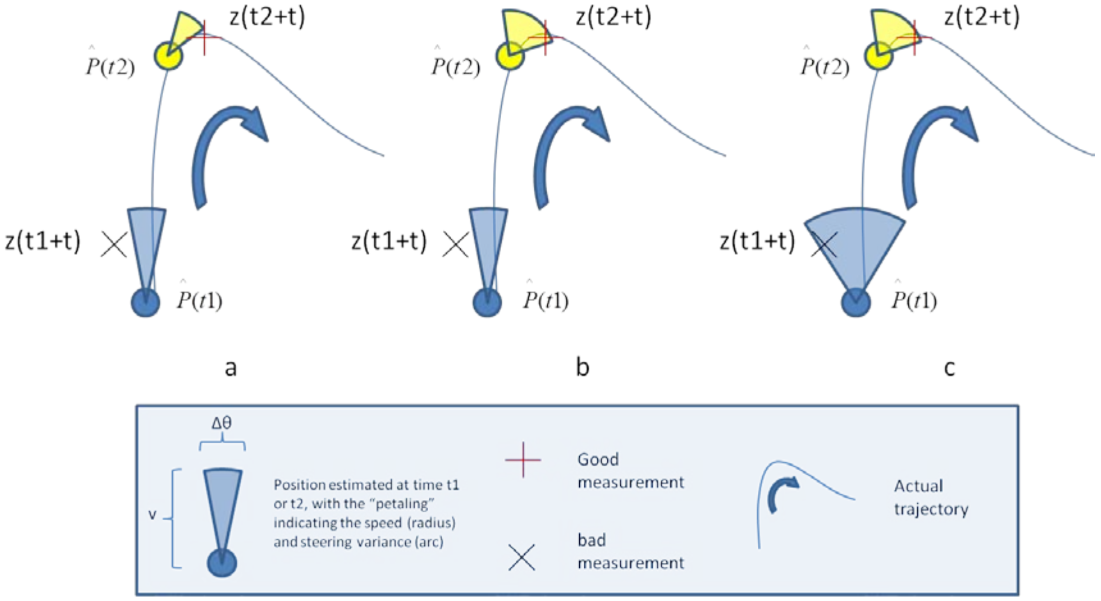
z(t2+t)
$\hat{P}(t2)$
z(t1+t)
$\hat{P}(t1)$
a
b
c
Δθ
v
Position estimated at time t1 or t2, with the "petaling" indicating the speed (radius) and steering variance (arc)
Good measurement
bad measurement
Actual trajectory

Figure 2.7 Illustration of the advantage of "dynamic maximum variance"

In this graphical demonstration (Figure 2.7), the circle point $\hat{P}$ indicates the estimated position at time *t1* or *t2* which are not necessarily correlated; the "petaling" indicates the instantaneous tangential velocity and the steering variance with radius and arc respectively. According to the model (2-10), the position at next time instance *t1/t2+*t_s , is expected inside the "petaling".

The "good measurement" means the measurement from the sensor with small deviation to the true value sampled at *t2+*t_s, and "bad measurement" means the measurement with greater deviation to the true value sampled at *t1+*t_s, t_s is the sampling time.

The graph represents how the measurements are followed by estimators. For example, in plot b, the estimator $\hat{P}(t1)$ is tracking the measurement *z(t1+*t_s*)*. Since the measurement *z(t1+*t_s*)* is a "bad measurement" and it is out of the "petaling" of $\hat{P}(t1)$, we would trust more the estimator than the measurement. Of course it is under the assumption that $\hat{P}(t1)$ is a good estimator. Thus when to predict the position at *(t1+*t_s*)*, $\hat{P}(t1 + t_s)$ is supposed to be closer to the "petaling" than measurement *z(t1+*t_s*)*, i.e. $\hat{P}(t1 + t_s)$ will be closer to the true trajectory.

In "plot a", with a smaller but constant variance of steering, the model will recognize the both "good" and "bad" measurements as corrupted ones. And in the prediction, the estimated states will be over rectified, i.e. a "good" measurement may be rectified. On the other side, in "plot c", the filter has a bigger variance, which consequently accepts both the measurements as well sampled ones. And in the prediction, the states will be under rectified, i.e. a "bad" measurement may be not rectified. Compared with constant variance, the advantage of "dynamic variance" is conspicuous. Regulated by the velocity-curvature dependence, the possibility of steering during walking is better estimated than the previous 5-states model.

However, the model has its limitations. When the linear velocity is too low, the SNR is getting worse, i.e. the amplitude of the velocity is too small to distinguish itself from the noise. The variance of the system diverges quickly. A linear velocity threshold is suggested in this thesis, to overcome the problem. When the linear velocity goes below the threshold, the model is not applied as long as the velocity stays under the threshold. Instead, the estimators will only rely on the current measurements, i.e. the estimates are simply the duplicates of the measurements. The linear velocity will be calculated as the difference of the position with respect to the sampling time. As the velocity threshold is set about 0.3 m/s, which is quite low, the period under the threshold is called “Pseudo-stationary”. When the speed is continuously above the threshold for three times, the model will be applied again. This piece wise method is conceived based on the consideration that a potential injury risk is much greater on the high speed object.

Chapter 3 Visual tracking system

In order to present a self-contained element, we include in this thesis a description of the visual tracking system, which has been developed before this thesis.

3.1 Visual tracking system based on surveillance cameras

As a task in ROSETTA project, the safety controller system should be easy to be deployed in the existing-facilities, especially for the sensor technologies applied in the system. The installation of the sensors must be effortless, low cost and without any restrictions on the objective, the human. Nevertheless, the measurements from the sensor technology should be adequate for the scenario. Compared with mechanical motion capture systems, inertial motion capture systems and magnetic motion capture systems, the visual capture system based on surveillance cameras, e.g. a visual tracking system based on multi-cameras [16], is more suitable. This visual tracking system just needs to mount commercial cameras on the ceilings of robotic cells, and calibrate once when installed. Moreover, it does not need to put any other devices on the human, which makes it more comfortable and convenient for the human.

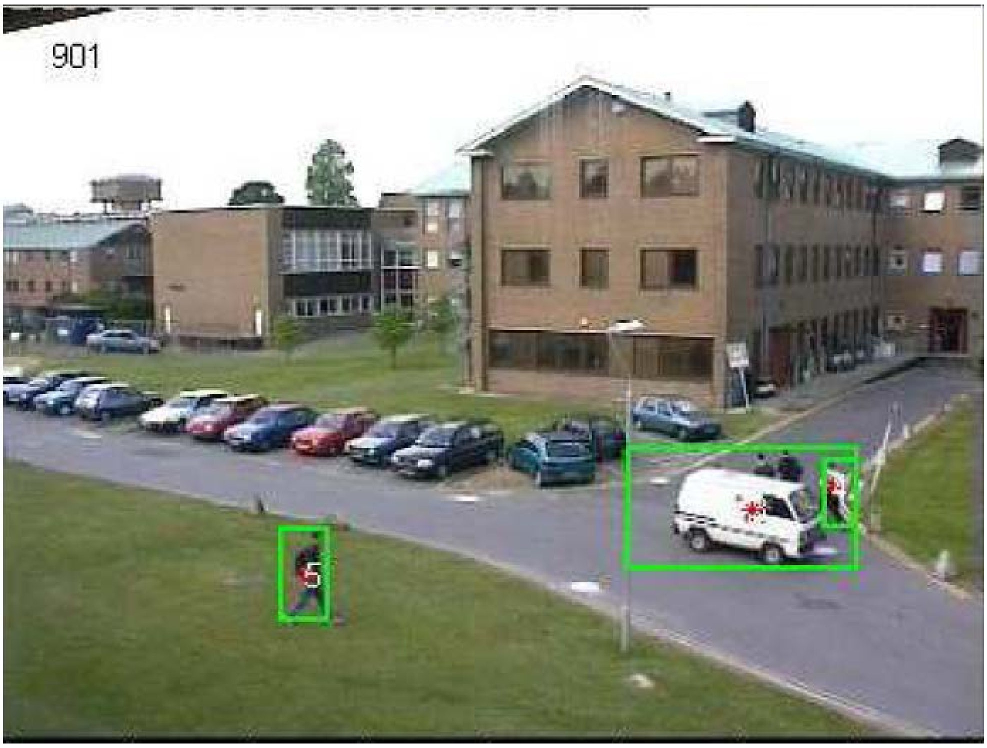

Figure 3.1 Image from visual tracking system described in [18].

Applying the computer vision techniques and a specific model of human, the visual tracking system can track the motion of human in the images captured by the surveillance cameras (Figure 3.1).

3.2 Particle filter

A particle filter is usually employed to filter the images captured from the visual tracking system [17], [18]. It provides a probability distribution over possible states rather than an exact state. The key feature of the particle filter is that the posterior is approximately represented by a set of particles, each particle including a state vector and an associated weight, which indicates the probability of the state vector.

In this thesis, the particle filter implemented in the visual tracking system is providing the possible positions of the tracked human, in the form of *x* and *y* coordinates. And each position was given a probability indicating the reliability of the states. The information generated by the particle filter will be considered as the inputs of the following extended Kalman filter.

Chapter 4 Speed estimation based on extended Kalman filter

There are many various algorithms about estimating the positioning and tracking of human, based on different tracking devices. In the previous chapter, a particle filter is applied to the visual tracking system, to filter the noise and imprecision of the image process. However, the particle filter assumes that the velocity of the human is constant. The simplified kinamtic model does not concern the tangential velocity of the human. And it focuses on the geometric characteristics of the trajectory. In this thesis, an EKF is proposed to fuse the position “measured” by the “front end” particle filter, to estimate the linear tangential velocity of the walking human.

4.1 Extended Kalman filter introduction

For system identification purpose, we consider the previously derived dynamic state-space model of walking human (2-10):

$$\begin{cases} \dot{\mathrm{x}}(\mathrm{t}) = \mathrm{v}(\mathrm{t}) \cdot \cos(\theta(\mathrm{t})) \\ \dot{\mathrm{y}}(\mathrm{t}) = \mathrm{v}(\mathrm{t}) \cdot \sin(\theta(\mathrm{t})) \\ \dot{\theta}(\mathrm{t}) = \mathrm{k}(\mathrm{t}) \cdot \mathrm{v}(\mathrm{t}) \\ \dot{\mathrm{k}}(\mathrm{t}) = \alpha_{\mathrm{rev}} \cdot \beta_{\mathrm{rev}} \cdot \mathrm{v}(\mathrm{t})^{\beta_{\mathrm{rev}}-1} \cdot \mathrm{cn}(\mathrm{t}) \\ \dot{\mathrm{cn}}(\mathrm{t}) = \mathrm{w}(\mathrm{t}) \\ \dot{\mathrm{v}}(\mathrm{t}) = \mathrm{a}(\mathrm{t}) \end{cases} \tag{2-10}$$

The state-space model can be translated into a compact form (4-1):

$$\dot{x}(t) = f(x) + w(t) \tag{4-1}$$

In which:

$\mathrm{x(t)} = \begin{bmatrix} \mathrm{x(t)} \\ \mathrm{y(t)} \\ \mathrm{\theta(t)} \\ \mathrm{k(t)} \\ \mathrm{cn(t)} \\ \mathrm{v(t)} \end{bmatrix}$ and $w(t)$ is the process noise of the model, $w(t) \sim N(0, Q)$.

The stochastic estimation of the current state is presented through the posterior probability density function (PDF) when all the measurement up to the current time instant and all the inputs up to the previous time instant are given. Bayes filter calculates the posterior PDF in two steps: 1) prediction and 2) update. Using the Bayes rule and the Markov property, that is if the current state is known, the future state is independent of the past states, the prediction and update step can be formulated as the likelihood. In order to construct the posterior PDF, the prior should be available including the initial PDF.

A Bayes filter requires integration over the state space, which is often impossible to calculate analytically. In some cases, the posterior distribution can be calculated such as the case of a linear Gaussian state-space model (i.e., KF). The well-known Bayes filter, Kalman Filter (KF), presents an optimal solution by assuming that the posterior density is Gaussian. In order for the posteriors to be Gaussian at every time step, the following conditions have to be satisfied.

1) The initial PDF is Gaussian.
2) *f(x)* (the propagation function of the states) is a linear function with added zero mean Gaussian noise.
3) The function of measurements is a linear function of *x*(t) with added zero mean Gaussian noise.

The KF assumes that the posteriors of the process noise and the measurement noise are zero-mean Gaussian distributions, and the initial state (*x*(0)), the covariance of process noise (Q), and the covariance of measurement noise are known.

Although our kinamtic model of a walking human is simplified, the state function is still non-linear. The standard Kalman filter can not be applied directly. The extended Kalman filter (EKF) is an optimal estimator in the least-square sense for estimating the states of dynamic nonlinear systems, and it is, thus, a viable and computationally efficient candidate for the online determination of position and speed in some locomotion tracking scenarios. The EKF consists of using the classical Kalman filter equations to the first-order approximation of the nonlinear model about the last estimate. A brief introduction of the EKF is necessary to help explaining the derivation of the algorithm implemented in this thesis.

For now, we consider a general non-linear dynamic system with the observation model:

$$x_k = f(x_{k-1}, u_{k-1}) + w_{k-1}$$
$$z_k = h(x_k) + v_{k-1} \qquad (4\text{-}2)$$

Where “k” is the time step, “w_k” and “v_k” are the process and observation noises which are both assumed to be zero mean Gaussian noise with covariance Q_k and R_k respectively. The function “f(•)” can be used to compute the predicted state from the previous estimate and similarly the function “h(•)” can be used to compute the predicted measurement from the predicted state. However, “f(•)” and “h(•)” cannot be applied to the covariance directly. Instead a matrix of partial derivatives (the Jacobian) is computed. At each time step, the Jacobian is evaluated with current predicted states. These matrixes can be used in the Kalman filter equations. This process essentially linearizes the non-linear function around the current estimate. Discrete-time predict and update equations are computed as follow:

Predict	
Predicted stat estimate:	$\hat{x}_{k\|k-1} = f(\hat{x}_{k-1\|k-1}, u_{k-1})$
Predicted covariance	$P_{k\|k-1} = F_{k-1} P_{k-1\|k-1} F_{k-1}^{T} + Q_{k-1}$

estimate:	
Update	
Innovation or measurement residual:	$\tilde{y}_k = z_k - h\left(\hat{x}_{k\|k-1}\right)$
Innovation (or residual) covariance:	$S_k = H_k P_{k\|k-1} H_k^T + R_k$
Local optimal Kalman gain:	$K_k = P_{k\|k-1} H_k^T S_k^{-1}$
Updated state estimate:	$\hat{x}_{k\|k} = \hat{x}_{k\|k-1} + K_k \tilde{y}_k$
Updated estimate covariance:	$P_{k\|k} = (I - K_k H_k) P_{k\|k-1}$
Jacobians	
State transition Jacobian	$F_{k-1} = \frac{\partial f}{\partial x} \|_{\hat{x}_{k-1\|k-1}, u_{k-1}}$
Observation Jacobian	$H_k = \frac{\partial h}{\partial x} \|_{\hat{x}_{k\|k-1}}$

The recursion equations given above is a first-order extended Kalman filter. Higher order EKFs may be obtained by retaining more terms of the Taylor series expansions. However, higher order EKFs tend to only provide performance benefits when the measurement noise is small, which in the case are not needed.

To implement the previously derived continuous-time state transition equations of walking human (2-10), they have to be converted into discrete-time equations. The discretization is executed with Euler-forward method.

$$\begin{cases} x(i+1) = x(i) + T \cdot v(i) \cdot \cos(\theta(i)) \\ y(i+1) = y(i) + T \cdot v(i) \cdot \sin(\theta(i)) \\ \theta(i+1) = \theta(i) + T \cdot k(i) \cdot v(i) \\ k(i+1) = k(i) + T \cdot \alpha_{rev} \cdot \beta_{rev} \cdot v(k)^{\beta_{rev}-1} \cdot cn(k) \\ cn(i+1) = w(i) \\ v(i+1) = v(i) + T \cdot a(i) \end{cases} \tag{4-3}$$

In the discrete-time state transition equations (4-3), “T” is the sampling time. It is not a constant in this case, because in the visual tracking system the acquisition time of the image and its processing time involve very heavy computation. In the demonstrations of the system, the sampling time is usually varied in a range of “20ms ~ 50ms”. As in the modeling stage “cn” is the state describing the willingness of turning, and it is assumed as a noise with a normal distribution. In another word, the previous “thought of turning” does not have an influence on the current “thought of turning”. So when “cn” is propagating, it shall not have any information about the previous state.

For the observe model the discretization is trivial.

$$\begin{cases} z_1(i) = x(i) + v_1 \\ z_2(i) = y(i) + v_2 \end{cases} \tag{4-4}$$

“z_i” is the measurements feeding the extended Kalman filter, which are composed with the Cartesian coordinates and the measurement noise “v_1” and “v_2”.

The Jacobians are computed as follow:

$$\frac{\partial f}{\partial x} = \begin{bmatrix} 1 & 0 & -T\cdot v\cdot \sin\theta & 0 & 0 & T\cdot\cos\theta \\ 0 & 1 & T\cdot v\cdot\cos\theta & 0 & 0 & T\cdot\sin\theta \\ 0 & 0 & 1 & T\cdot v & 0 & T\cdot k \\ 0 & 0 & 0 & 1 & T\cdot\alpha/v & -T\cdot\alpha\cdot cn/v^2 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

For the parameter α in the Jacobian, it can be approximated from the original equation of “power law”.

4.2 Fusion of particle filter and extended Kalman filter

The particle filter implemented in the system will generate a set of particles bearing their weights. Each particle represents the possible position at a certain time, with the possibility indicated by its weights. As in this particle filter, some approximations have been already executed, such as the assumption of the constant linear tangential velocity. The possibility calculated by the particle filter associated to the particles is derated by its impreciseness. To increase the reliability of the output of the particle filter, instead of choosing the most possible particle as the result, several subordinate particles should be chosen as well. Thus “the measurements” vector will increase its dimension, including a new variable: “possibility” of the position.

To exploit this “possibility” of the position, the particle filter at “front end” is considered as a multi-sensor with “variant accuracy”, which will provide measurements to the extended Kalman filter. However, the so-called “variant accuracy” can not be directly interpreted into the variance matrix (R). Moreover, the “accuracy”, which is expressed by the possibility of the particle, is not time invariant. In this scenario, a fusion technique could be used to bridge the particle filter and the extended Kalman filter.

The information fusion of Kalman filtering has been studied and widely applied to integrated navigation systems for maneuvering targets, such as airplanes, ships, cars and robots. When multiple sensors measure the states of the same stochastic system, generally we have two different types of methods to process the measured sensor data. The first method is the centralized filter [9], where all measured sensor data are communicated to a central site for processing. The advantage of this method is that it involves minimal information loss. However, it can result in severe computational overhead due to overloading of the filter with more data than it can handle. Consequently, the overall centralized filter may be unreliable or suffer from poor accuracy and stability when there is severe data fault.

The second method is the decentralized filter where the information from local estimators can yield the global optimal or suboptimal state estimate according to certain information fusion criterion. The advantage of this method is that the requirements of communication and memory space at the fusion center are broadened, and the parallel structures would lead to increase in the input data rates. Furthermore, decentralization leads to easy fault detection and isolation. However, the precision of the decentralized filter is generally lower than that of the centralized filter when there is no data fault.

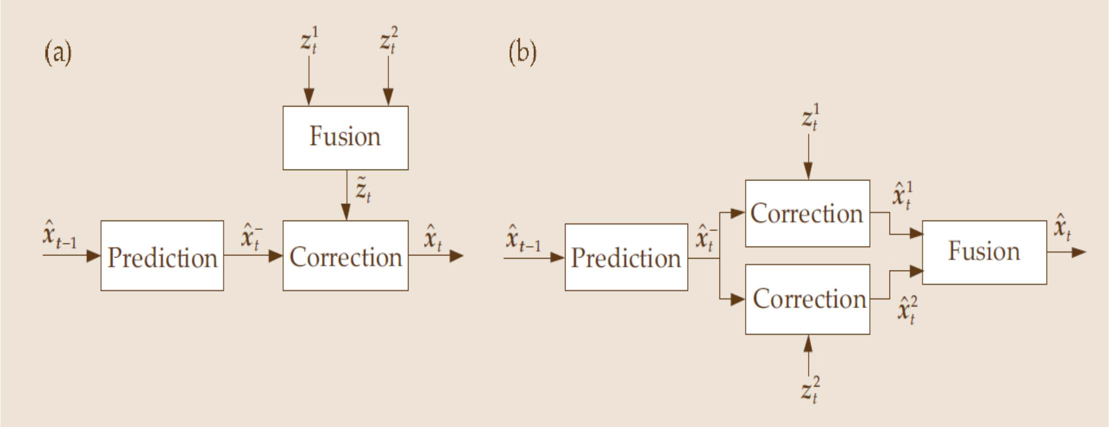

Figure 4.1 Block chart of two different fusion algorithms, (a) is the centralized filter, (b) is the decentralized filter.

In the paper of Shu-Li Sun et al. [10], the two different algorithms have been discussed. Based on the decentralized fusion algorithm, the author proposed a new multi-sensor optimal information fusion criterion weighted by matrices in the linear minimum variance sense. It is equivalent to the maximum likelihood fusion criterion under the assumption of normal distribution. Based on this optimal fusion criterion, a general multi-sensor optimal information fusion decentralized Kalman filter with a two-layer fusion structure is given for discrete time linear stochastic control systems with multiple sensors and correlated noises (Figure 4.2). The theorem of the optimal fusion decentralized filter gives the relationship between variance of the optimal information fusion estimator (P), multi-estimators' weights ($\mathrm{A_i}$) and their variance ($\mathrm{P_i}$)

respectively.

$\hat{x} = A_1 \cdot \widehat{x_1} + A_2 \cdot \widehat{x_2} + \dots + A_l \cdot \widehat{x_l}$ (4-5)

Let $\widehat{x_1}$, i = 1, 2, …, *l* be unbiased estimators of an n-dimensional stochastic vector x. The variance and cross covariance matrices are denoted by Pii (i.e. Pi) and Pij , respectively. Then the optimal fusion (i.e. linear minimum variance) estimator with matrix weights is given as

$\hat{x} = A_1 \cdot \widehat{x_1} + A_2 \cdot \widehat{x_2} + \dots + A_l \cdot \widehat{x_l}$ (4-6)

where the optimal matrix weights A_i, i = 1, 2, …, *l* are computed by

A = $\Sigma^{-1}\, e\, (e^T\, \Sigma^{-1}\, e)^{-1}$ (4-7)

where Σ=(P_{ij}), I, j=1, 2, …, *l* is an nl×nl symmetric positive definite matrix, A = $[A_1, A_2, \dots, A_l]^T$ and e=[I_n, … , $I_{n,}$]^T are both nl × n matrices. The corresponding variance of the optimal information fusion estimator is computed by

P = $(e^T \Sigma^{-1} e\,)^{-1}$ (4-8)

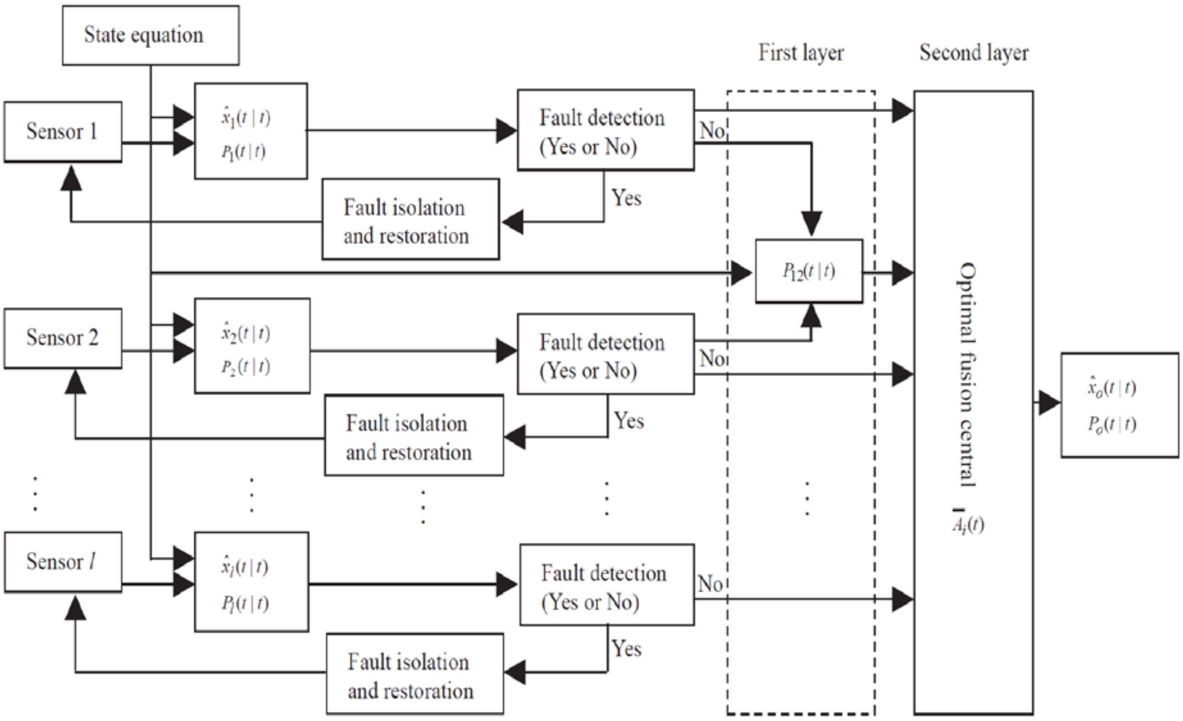

Figure 4.2 Multi-sensor optimal information fusion decentralized Kalman filter with a two-layer fusion structure [10]

According to the equations given above, a relationship can be drawn to compute the variance of each “sensor”, i.e. each particle, provided the possibility of its measurement.

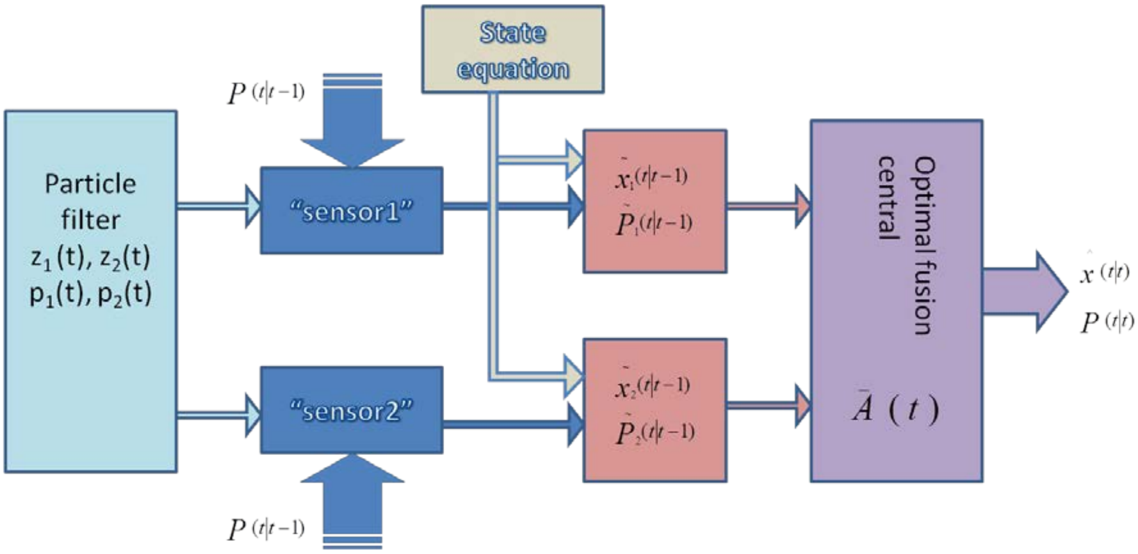

Figure 4.3 Propagation of “measurements” of each “sensor”

In (Figure 4-3), the flow chart is presenting the propagation scheme of particles generated by the particle filter, to the optimal fusion central. Before deriving the iterative equations, a main assumption shall be made: for the simplicity of the computation, the particles are not correlated to each other. And to demonstrate the derivation, two particles are considered as the circumstance. Under the assumption, the matrix Σ = (P_{ij}), I, j=1, 2, ... , *l,* can be constructed as a block diagonal matrix:

$$\Sigma = \begin{bmatrix} P_1 & 0 \\ 0 & P_2 \end{bmatrix} \qquad (4\text{-}9)$$

P_1, P_2 are covariance matrix of each particle, in this thesis, it is the covariance of “x, y” (Cartesian coordinates) of human at each time for each particle. Practically “x” and “y” are uncorrelated, thus P_1 and P_2 are diagonal themselves, too. So “Σ” is a diagonal matrix, and it is easy to get the inverse by computing the reciprocal of the elements on the diagonal.

$$\Sigma^{-1} = \begin{bmatrix} P_1{}^{-1} & 0 \\ 0 & P_2{}^{-1} \end{bmatrix} \qquad (4\text{-}10)$$

On the other hand, the probability of the particle should be converted to the weight correspond to the optimal matrix weights A$_i$. A most intuitive way is to normalize the possibility to estimate the weight.

$$a_1 = \frac{p_1}{p_1 + p_2}\ ;\ a_2 = \frac{p_2}{p_1 + p_2}\ ;$$

And construct weight matrix A as follow:

$$A = \begin{bmatrix} a_1 & 0 \\ 0 & a_1 \\ a_2 & 0 \\ 0 & a_2 \end{bmatrix}$$

Derived from equation (4-7) and (4-8), we can have the equation:

$$A\,P^{-1} = \Sigma^{-1}\,e \qquad \text{(4-11)}$$

“P” is assumed to be positive definite, so its inverse should exist. Expand the matrix:

$$\begin{bmatrix} a_1 & 0 \\ 0 & a_1 \\ a_2 & 0 \\ 0 & a_2 \end{bmatrix} \begin{bmatrix} pp_1{}^{-1} & 0 \\ 0 & pp_2{}^{-1} \end{bmatrix} = \begin{bmatrix} P_1{}^{-1} & 0 \\ 0 & P_2{}^{-1} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad \text{(4-12)}$$

From equation (4-12), we can solve Σ^{-1} element-wise.

$$a_1 P_1 = P\ ;\ \ a_2 P_2 = P$$

The weight is time dependent as the possibility of the particle, at each time instance the weight is updated by the particle filter. So the variance of each channel is calculated recursively on the previous system variance.

$$P_1(t|t-1) = \frac{P(t-1)}{a_1(t)},\ P_2(t|t-1) = \frac{P(t-1)}{a_2(t)} \qquad \text{(4-13)}$$

To implement this optimal information fusion technique in the extended Kalman filter, we have to structure two channels for the incoming data. Each channel is fed by a common prediction step, and then has its own correction steps. For the update step, a unique estimator and its variance will be computed based on the fusion central.

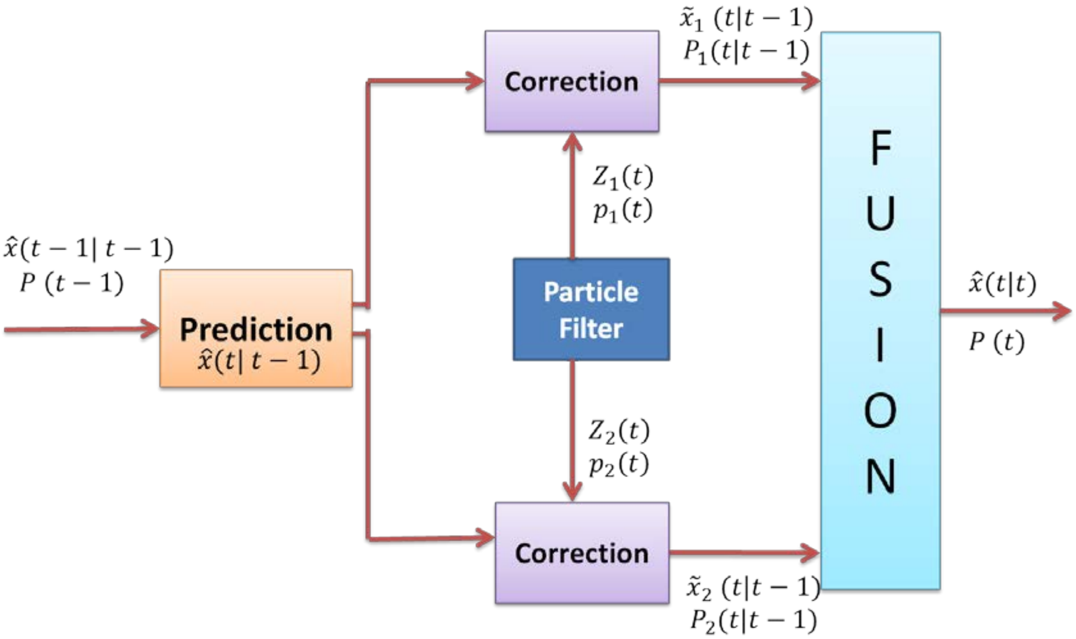

Figure (4.4) Block diagram of extended Kalman filter with the fusion technique

In Figure (4.4), the prediction step is executed with the standard EKF equations. The recursion is a first-order Taylor expansion based on the estimator at time (t-1). Then the predicted state $\hat{x}$(t|t-1) is sent via the parallel channels to two individual correction steps. Each correction step is fed by the particle filter with selected particles. In this thesis, the particles are selected by the most 2 possible ones. The variance of each channel is calculated according to the equation (4-14) and (4-13). And use the variance to compute the Kalman gain for each channel. The new estimator at time "t" will be the weighted sum of these two channels outputs. And the variance for the system is computed by equation (4-8).

In the case the dimension of measurements is not consistent with the dimension of model states, the channel variance could be computed either as the prediction variance or as the innovation variance. The weight can be treated as a scaling variable.

Under the assumption of uncorrelation of the particles and uncorrelation of the variance at each time instance, the fusion algorithm implemented in this thesis is an approximate way to estimate the individual variance of each particle. As a matter of fact, the particles are actually propagating inside the particle filter, the uncorrelation of the particle can not be ensured.

4.3 Tuning of extended Kalman filter

The Kalman filtering equations provide an estimate of the state $\hat{x}(t|t)$ and its error covariance P(t|t) recursively. The estimate and its quality depend on the system parameters and the noise statistics fed as inputs to the estimator. This section analyzes the effect of uncertainties in the statistical inputs to the filter. In the absence of reliable statistics or the true values of noise covariance matrices Q_t and R_t, the expression:

$$P_{t|t} = (I - K_t\,H_t)\,P_{t|t-1}(I - K_t\,H_t)^T + \,K_t\,R_t\,{K_t}^T \tag{4-14}$$

no longer provides the actual error covariance. In other words $P_{t|t}$ is not equal to the expected value of $\left(x_t - \hat{x}_{t|t}\right)\left(x_t - \hat{x}_{t|t}\right)^T$. In most real time applications the covariance matrices that are used in designing the Kalman filter are different from the actual noise covariance matrices.

Tuning of Kalman Filter is very important to reach a small and reasonable delay in tracking. The tuning process includes identification of the measurement and process noise covariance matrices. The measurement noise covariance matrix R_t can be determined by off-line sample measurements as explained in [11]. However, determination of process noise covariance matrix Q_{t-1} is not straightforward.

The determination of the process noise covariance is generally more difficult as we typically do not have the ability to directly observe the process we are estimating. Sometimes a relatively simple (poor) process model can produce acceptable results if one “injects” enough uncertainty into the process via the selection of Q. Certainly in this case one would hope that the process measurements are reliable.

In either case, whether or not we have a rational basis for choosing the parameters, often times superior filter performance (statistically speaking) can be obtained by tuning the filter parameters Q and R. The tuning is usually performed off-line, frequently with the help of another (distinct) Kalman filter in a process generally referred to as system identification. We note that under conditions where Q and R are in fact constant, both the estimation error covariance and the Kalman gain will stabilize quickly and then remain constant. If this is the case, these parameters can be pre-computed by either running the filter off-line.

It is frequently the case however that the measurement error (in particular) does not remain constant, which is the case in this thesis. Also, the process noise is sometimes changed dynamically during filter operation—becoming Q_t —in order to adjust to different dynamics. For example, in the case of tracking the head of a user of a 3D virtual environment we might reduce the magnitude of if the user seems to be moving slowly, and increase the magnitude if the dynamics start changing rapidly. In such cases might be chosen to account for both uncertainties about the user's intentions and uncertainty in the model.

In this thesis, not all the states are observable. And the covariance can not be estimate off-line statically due to the high dynamic system. The best we can do is choose simple parameterizations for P(0), Q and R (typically identity matrices multiplied by a scalar to tuned). In most cases the EKF matrices are designed and tuned by trial-and-error procedures. By varying the matrix elements in a range of several decades, we try to get the best fit for the specific application, using either simulations or experimental data acquired in the demos.

Other than the covariance matrixes, the initial states also have a great influence in the performance of the filter. To suppress the variance, the initial value should be estimated properly. The closer to the mean value, the less iteration steps are needed to rectify the states.

Chapter 5 Validations

The algorithm of the human walking speed estimation is validated with offline data, which was acquired from the human detection and tracking system of the project ROSETTA in the experimental scenario.

5.1 Set up of the experimental scenario

An experimental scenario (Figure 5.1) has been set up to simulating the circumstance of the co-existence of human and robot. An environment of approximately 3m×2.5m, with a single entrance has been selected. An industrial robot is placed in the center of this space, surrounded by an interference area (red area marked with number 3 in Figure 5.1) that covers the workspace of the selected task. Two tables, representing workstations, have been also added, at the left and right side of the robot, each one with a corresponding cooperation area (blue areas marked with number 2 in Figure 5.1).

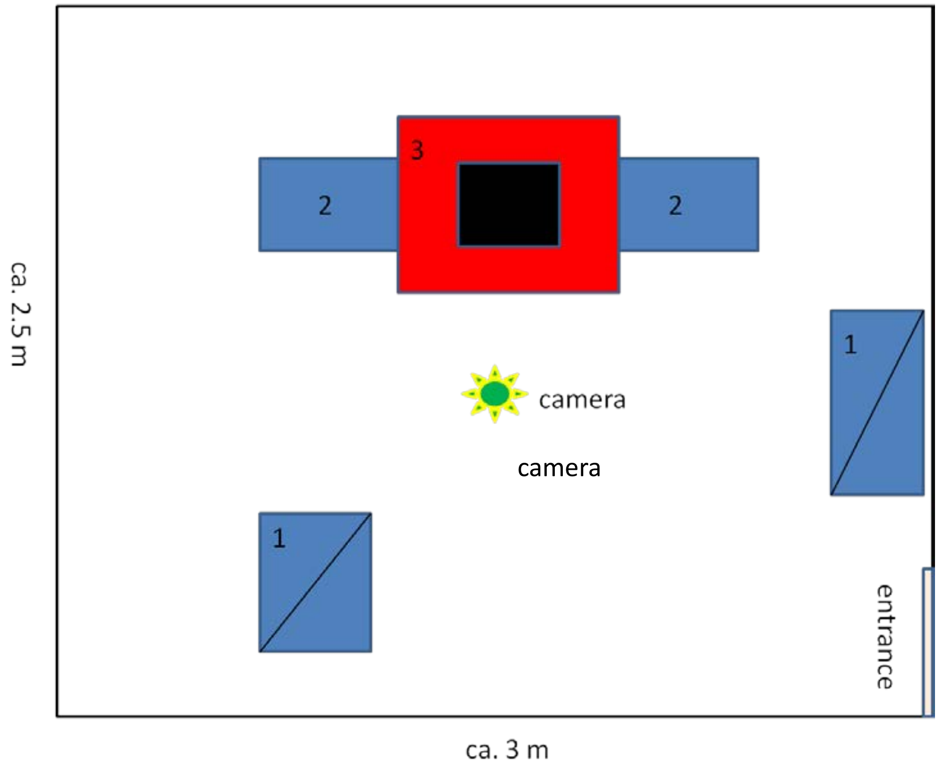

Figure 5.1 Layout of the experimental scenario

Two more areas, which are far from the robot and have to be intended as coexistence area (blue areas crossed with a line and marked with number 1 in Figure 5.1), have been added as well. The robotic cell has been equipped with two ceiling mounted surveillance cameras. The two cameras have been suspended at about 3m and located at a distance that ensures a complete overlap on the interference area.

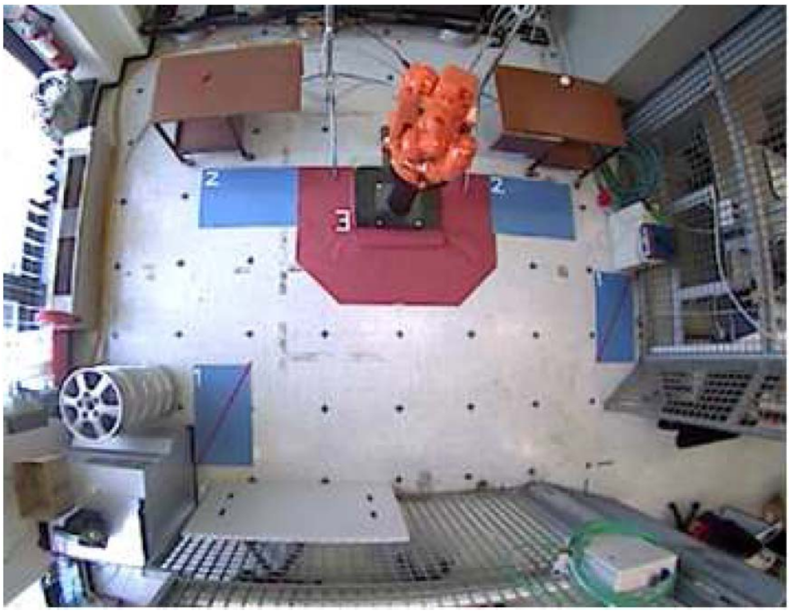

Figure 5.2 A snapshot of the experimental scenario

A preliminary software architecture (Figure 5.3), including the human detection and tracking and the intention estimation software, has been implemented to demonstrate the HDT-IE functionality.

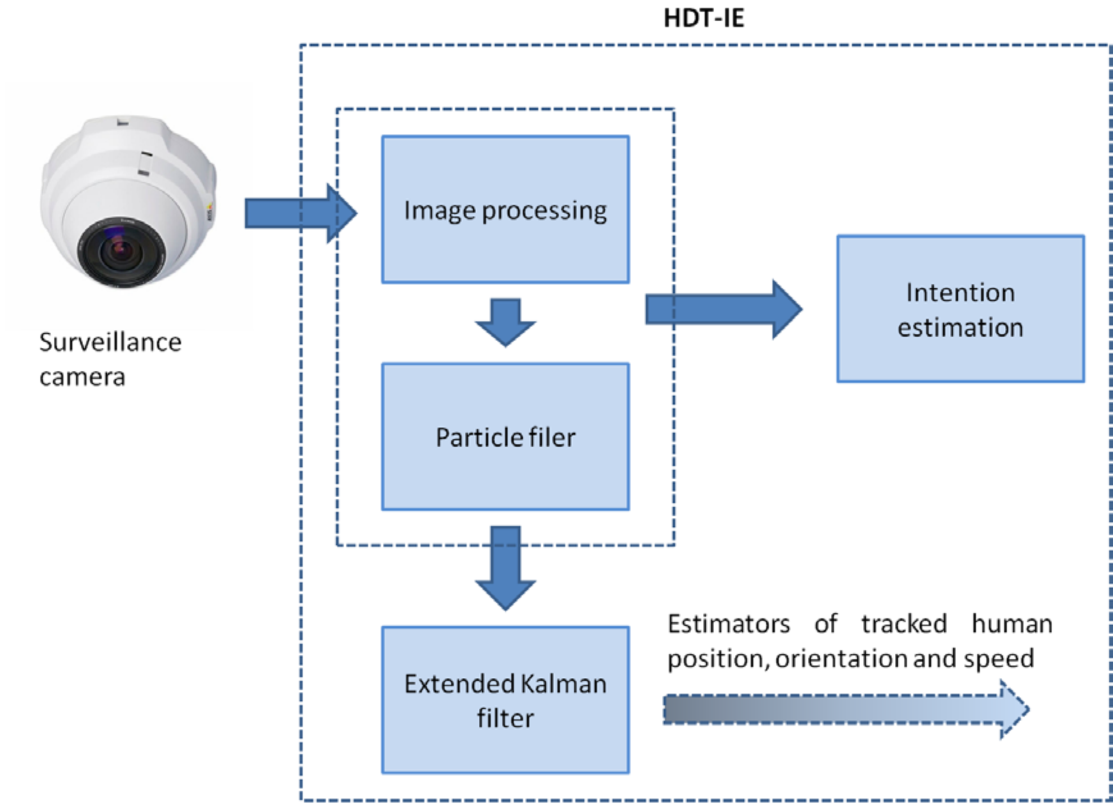

Figure 5.3 Software architecture of HDT-IE

A set of five volunteers was selected to perform five different experiments of human detection and intention estimation in a robotic cell. The experiments are focused on walking humans. Each experiment is structured according to the following steps:

1. the volunteer enters the robotic cell from a door located in the bottom-right corner;
2. the volunteer steps towards a preconceived destination;
3. the volunteer stops at the destination and performs a simple task;
4. the volunteer comes back to the entrance door;
5. the volunteer leaves the robotic cell.

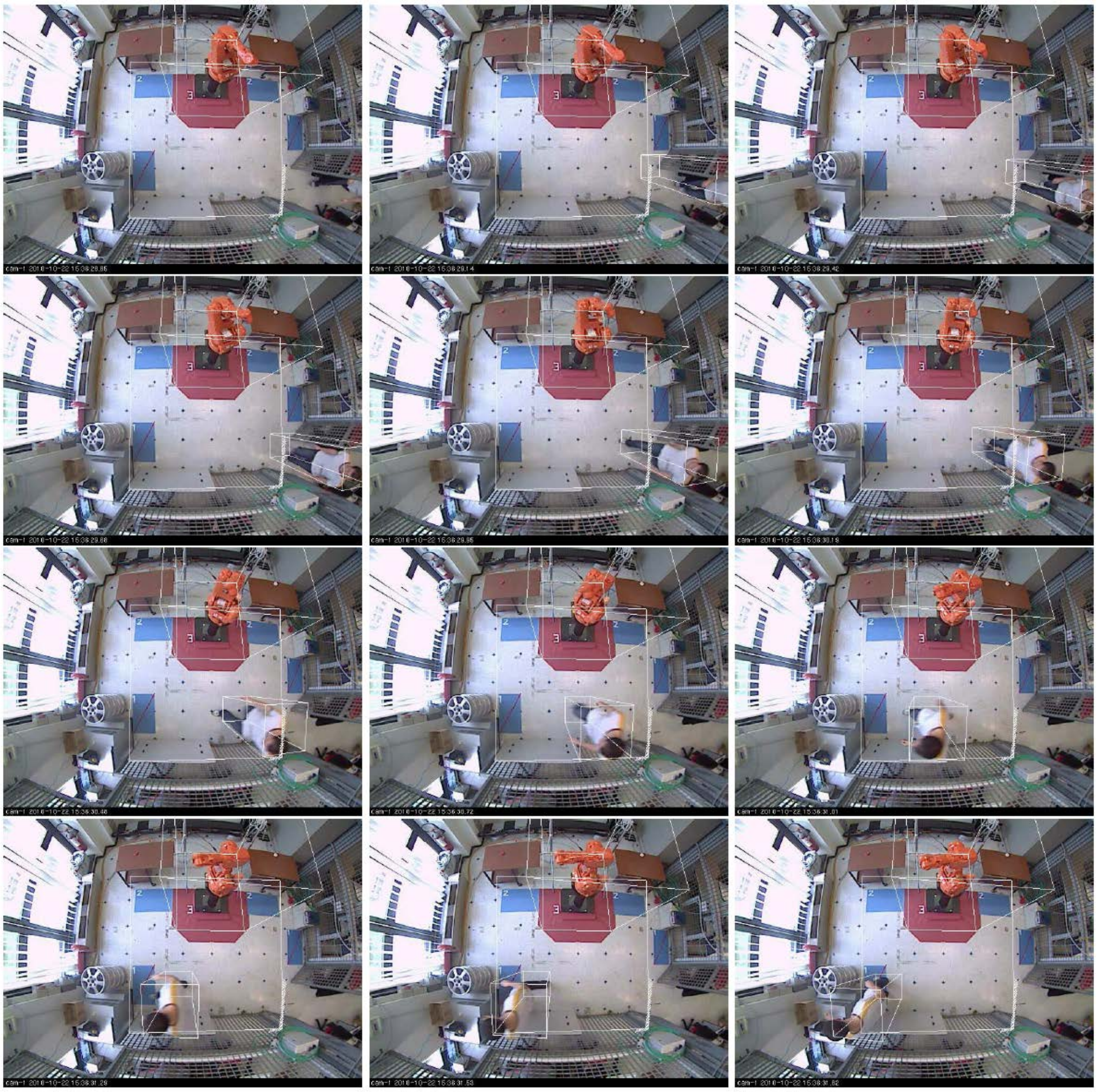

Figure 5.4 A sequence of frames extracted from a coexistence experiment (the box around the volunteer is created by the HDT algorithm).

Experiments have been following the protocol described above. The sequence of frames (Figure 5.4) illustrating the functionality of the HDT system, is to be read from left to right and from top to bottom. Also notice that the robot was moving during the experiments, however its motion is masked by the system in order to avoid any misinterpretation of this motion as coming from a moving person. These experimental data of walking trajectories were then used to validate the algorithm.

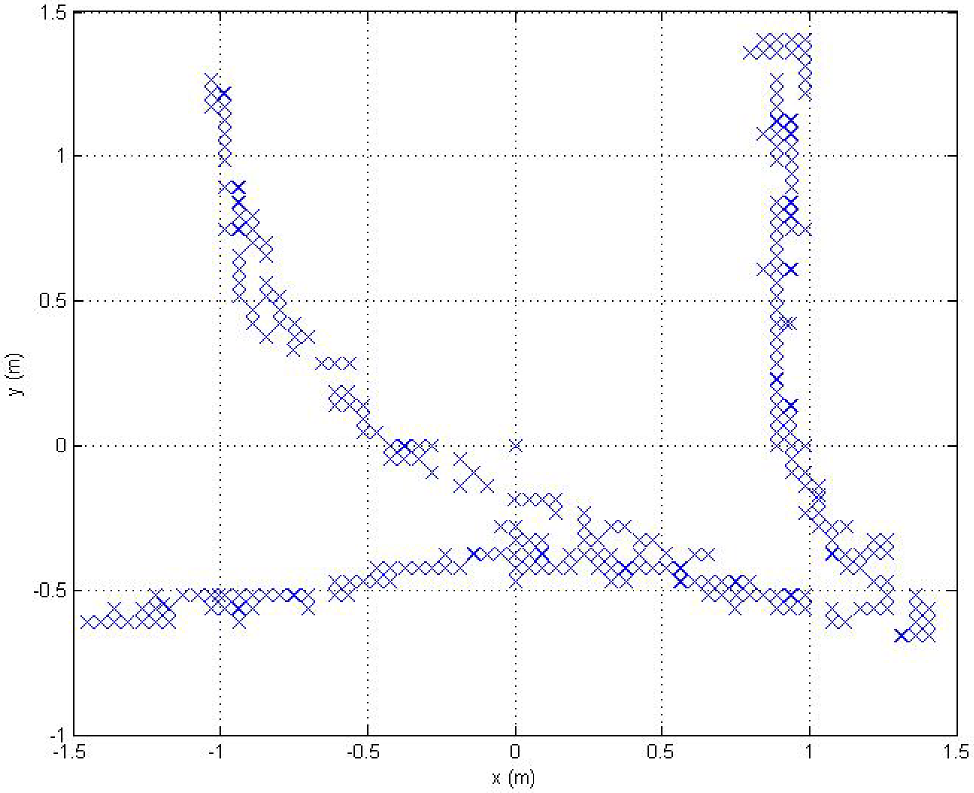

Figure 5.5 Examples of trajectories

The HDT computes the states associated to the tracked people and forwards them to the extended Kalman filter. The algorithm computes the predicted state in terms of Cartesian position and the velocity of the moving person. 2D plots of the estimated trajectories are shown in Figure 5.5, superimposed to the picture of the robotic cell already shown in Figure 5.2, for the same trajectories as in Figure 5.6.

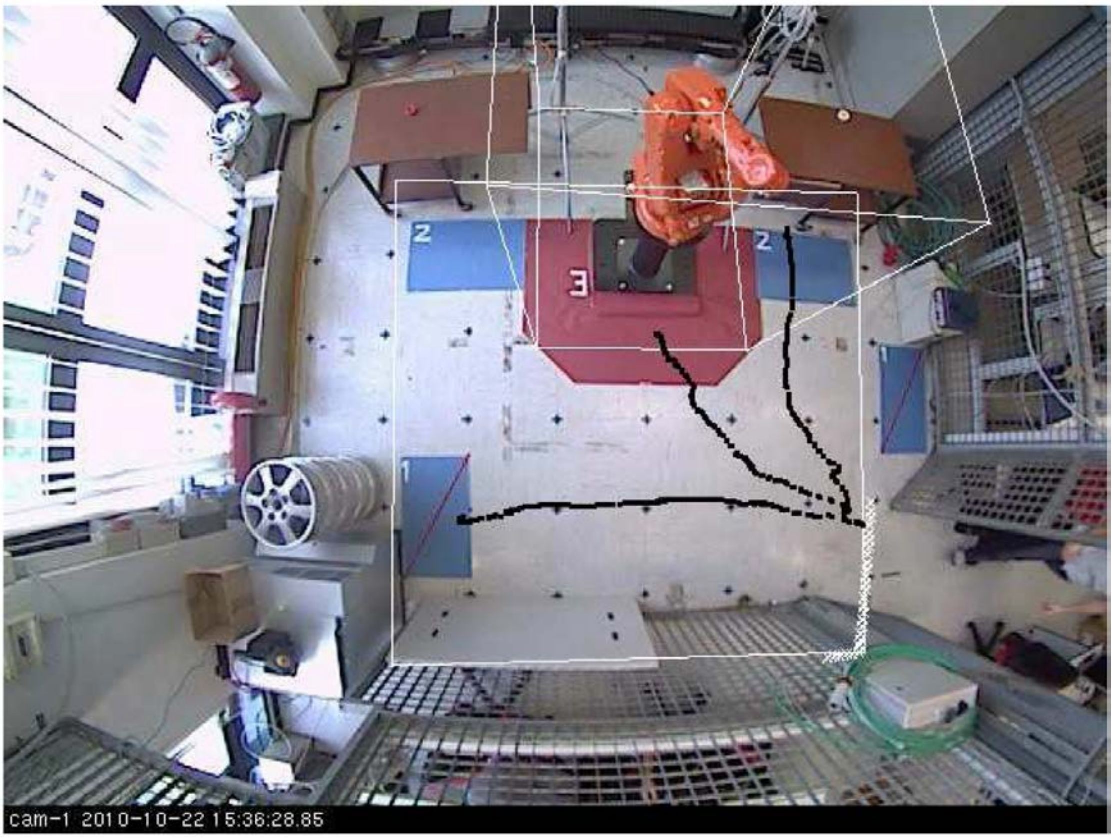

Figure 5.6 The estimated 2D paths reported into the robotic cell environment

The typical trajectories in Figure 5.6 approaching different areas represent three different interaction behaviors:

1. Coexistence areas: two areas (blue areas crossed with a red line and marked with number 1) far from the robot, where the volunteer is supposed to perform a simple manipulation task;
2. Cooperation areas: two areas (blue areas marked with number 2) near the robot, where the volunteer is supposed to perform a simple manipulation task;
3. Interference area: an area (red area marked with number 3) close to the robot, where the volunteer should not enter.

For each trajectory, the volunteers repeated the same predestinated routs for five times. For each trial a set of data was generated by the particle filter. The sets of data contain the following information:

- The time stamp for each particle.
- The ID of the particle: in case more than one person enters the robot cell. But it will be neglected because the experiment focused on one human tracking scenario.
- Coordinate of x-axes and y-axes.
- The probability represents the reliability of each particle. It is normalized, i.e. 1 indicates the best probability while 0 indicates the worst probability.

The human walking speed estimation proposed in this thesis is validated by the data sets described above.

5.2 Results and analysis

A standard algorithm has been implemented on the group of data sets. The so called “standard algorithm” is a modified extended Kalman filter specified in Chapter 4. The modified extended Kalman filter is featured by:

- The 6-states human walking model (velocity-curvature dependence).
- The data fusion technique.
- The piece wise filtering with a speed threshold

The results and analysis focus on the following problems:

1. **6-states human walking model versus 5-states model.**
2. **Best particle versus multi-particles with secondary reliability.**
3. **Algorithm with a velocity threshold versus algorithm without a velocity threshold.**
4. **Tuning of the process variance Q**
5. **The reliability of the estimated states**

1. 6-states human walking model versus 5-states model.

As discussed in Chapter 2, the difference between these two models is the velocity-curvature dependence. So the validation is to compare the two models with the same data set, to explore the ability of modeling the human walking. To have a more general conclusion, three different scenarios (coexistence, cooperation and interference) are combined together to simulate the situation that the system is working continuously, i.e. a human had entered and exited three times the robot cell in single trial (Figure 5.7 and Figure 5.8).

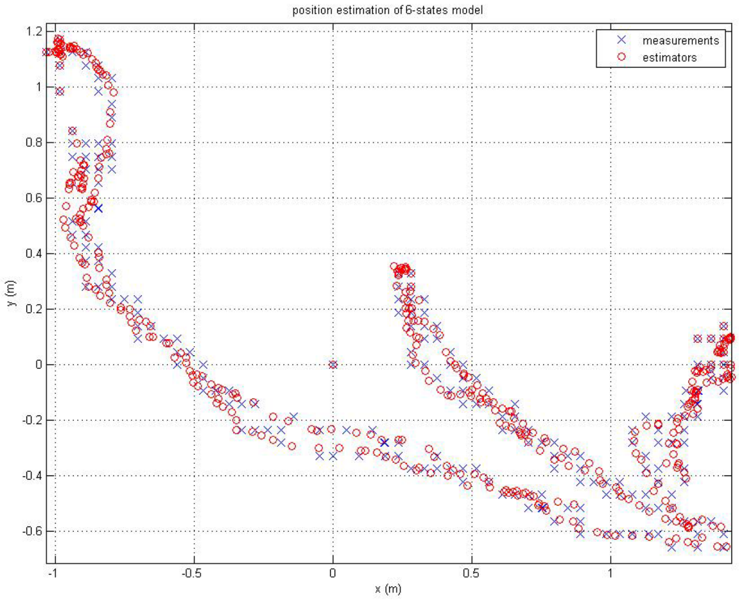

Figure 5.7 Position estimation of 6-states model

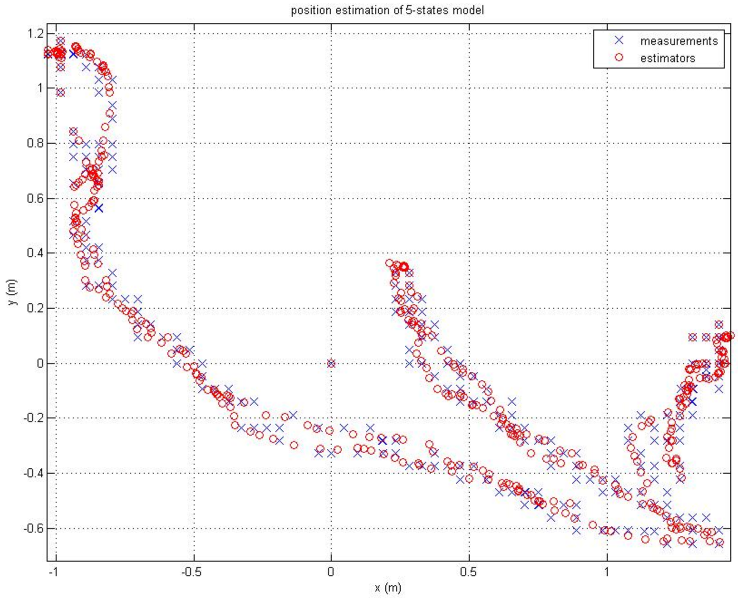

Figure 5.8 Position estimation of 5-states model

To reject other factors, the both algorithms are synchronized with their parameters, except the additional state in the 6-states model, i.e. both algorithms work with the same structure and initialization. To analyze the comparison of these two models, one of the trials is selected to demonstrate the difference. It is observed that:

- Both the models can track the measurements with the same errors (Figure 5.7 and Figure 5.8). By changing the variance matrixes of process and measurements (Q and R), the errors stayed synchronized between two models.
- The linear velocity profiles of two models are almost overlapping (Figure 5.9).
- Differences can be observed in the angle θ profiles (Figure 5.10). The θ profile of 5-states model is smoother than the one of 6-states.

Generally, the two models do not distinguish from each other obviously. From the position and linear velocity estimation, no significant impact from the 6th state can be observed. However one difference should be highlighted in this point. The variation of the angle θ shows that the 6-states model is more sensitive to the turning angle θ (Figure 5.10). Although it can not be confirmed the sensitiveness of the 6-states model indicates the better accuracy. Due to the absence of a more precise reference value to compare with, a superficial observation was left to be studied. As a probe into the causes of the ineffectiveness of this 6th state, the approximation during modeling the velocity-curvature dependence can be a factor. Also the un-modeled neural and ergonomic mechanism in the velocity-curvature degrades the behavior of this 6-states human walking model.

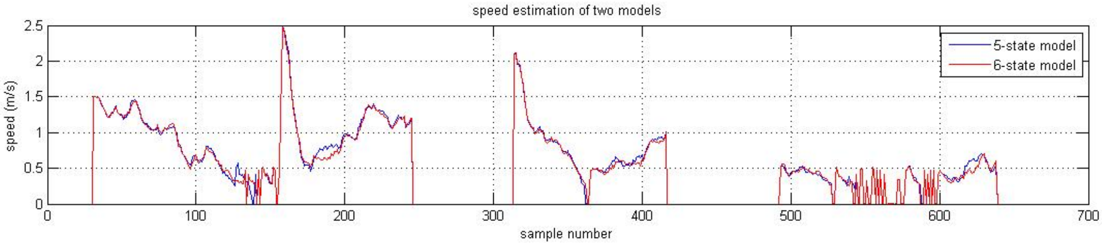

Figure 5.9 Walking speed estimated by two models

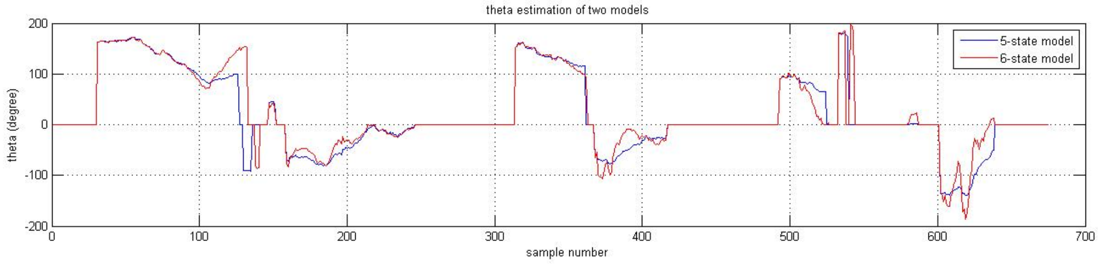

Figure 5.10 Theta estimated by two models (5-states vs. 6-states)

2. Best particle versus multi-particles with secondary reliability

This comparison is about the validation of the fusion algorithm proposed in Section 4.2. To analyze the performance of the algorithms with or without the fusion technique, we have to refer the estimations to the true value. So the validation of the fusion algorithm is based on artificial data sets, in which the “true” trajectories exit. Another reason to implement an artificial data set is that, the fusion algorithm does not involve the nature of human walking. The variety of the algorithms, different to the previous comparison about the velocity-curvature dependence, is about the algorithm itself.

Three typical trajectories are selected: straight line with a constant acceleration; curve with a constant linear velocity; curve with acceleration and deceleration (Figure 5.11). Each imaginary data set is built up with one “true” trajectory without noise and three trajectories with normal distributed noise. Each noisy trajectory is assigned a reliability associated to the variance of its noise, to simulate the output of the particle filter.

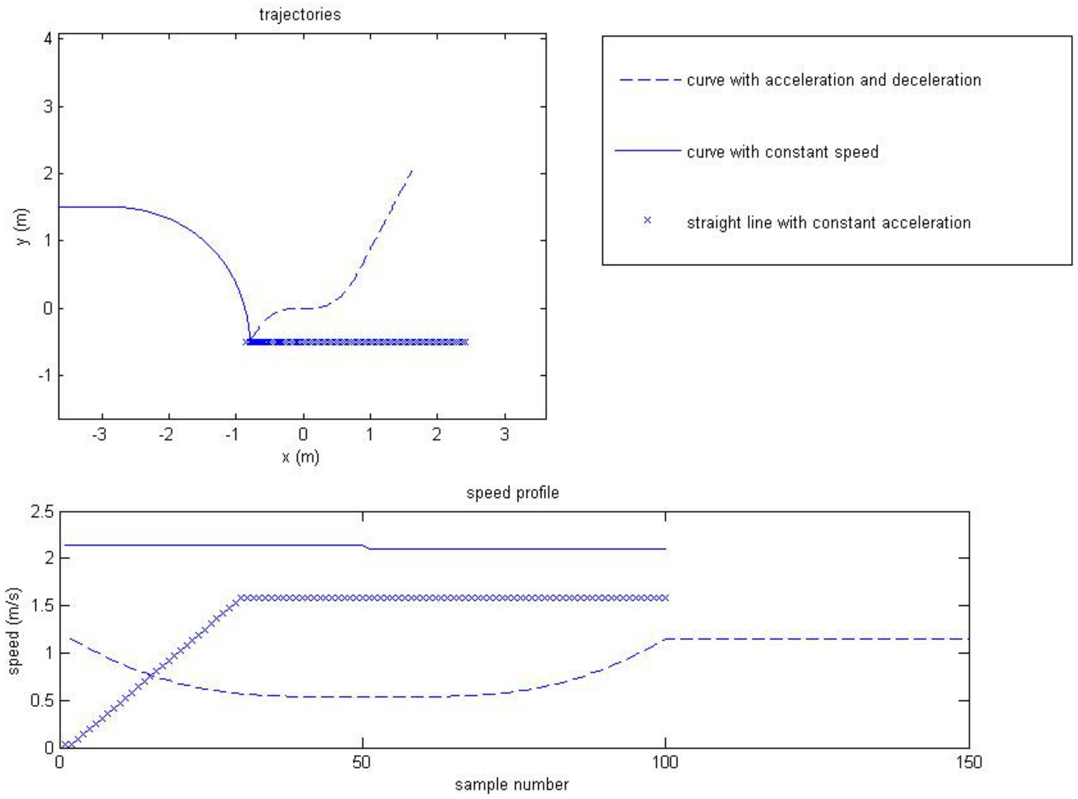

Figure 5.11 Artificial trajectories with speed profiles

All these trajectories have been applied both the algorithms with or without fusion technique. One of the three data sets is a curved line trajectory with a velocity profile described in Figure 5.11. This trajectory (Figure 5.12) is picked to illustrate the comparison. From the filtered linear velocity and angle profiles (Figure 5.13), we can observe that: with the same tracking delay, the algorithm with the fusion technique is more stable and precise. The same observation is also obtained in the other two trajectories. Thus the fusion algorithm in the extended Kalman filter is validated to be effective.

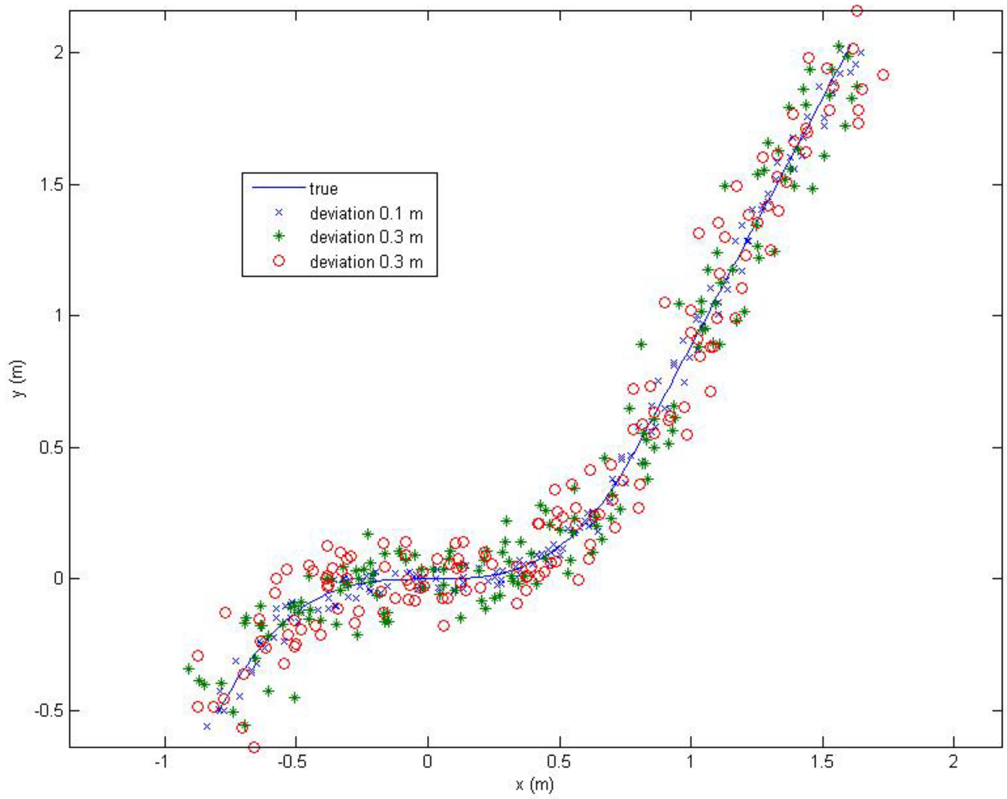

Figure 5.12 Imaginary trajectory with its noisy variations

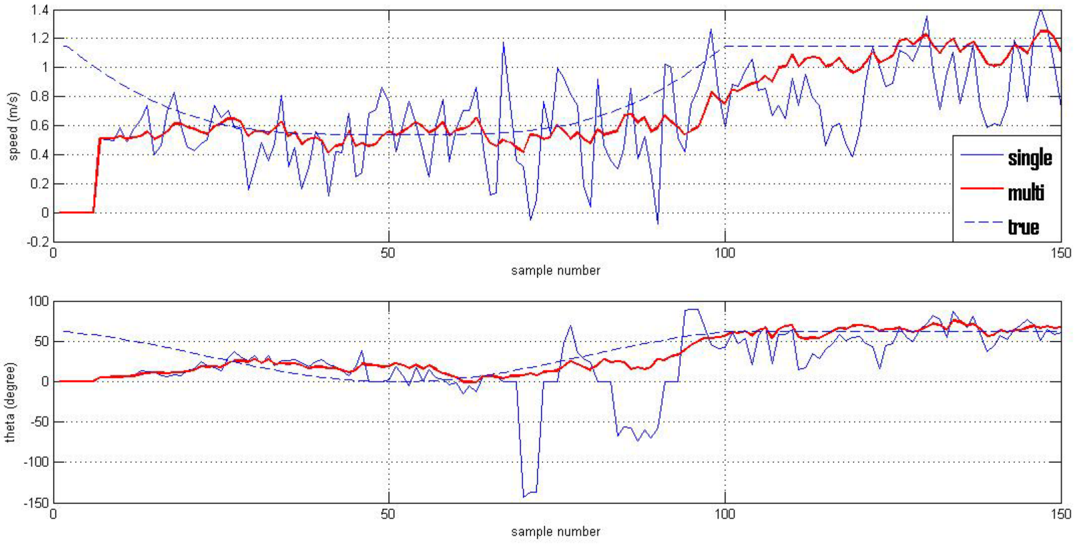

Figure 5.13 Comparison of algorithm w/o the fusion technique

3. Algorithm with a velocity threshold versus algorithm without a velocity threshold

The unicycle model described in Chapter 2 can simulate human walking properly at a relative higher speed (assumed >0.4 m/s). When the speed is low or the human is standing still, due to the limitation of the model discussed in Chapter 2, the unicycle model can no more properly track the direction the human is facing. The speed threshold is set to distinguish the high speed walking from the low speed walking (pseudo stationary). It endows the standard algorithm with the capability to filter the trajectory piece wise, and to compensate the drawback of the unicycle model.

To validate the argument above, a true data set has been selected to demonstrate the comparison of the algorithms with or without a speed threshold (Figure 5.14). A standard algorithm with the speed threshold (0.4 m/s) was applied to the trajectory. The estimations of the linear velocity, the angle θ and the velocity variance are plotted in Figure 5.15. To justify the performance of the estimation, a trajectory was reconstructed with the start point (x(0),y(0)) and the estimated velocity v and angle θ (Figure 5.16). The estimated velocity and angle can follow the shape of the trajectory without position corrections as expected.

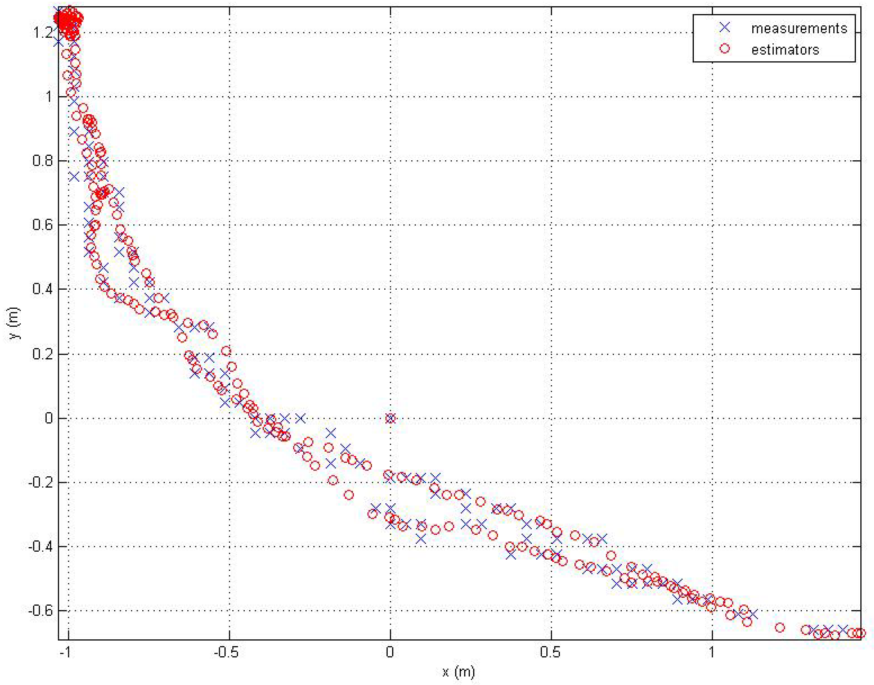

Figure 5.14 Trajectory selected and the estimation with the standard algorithm.

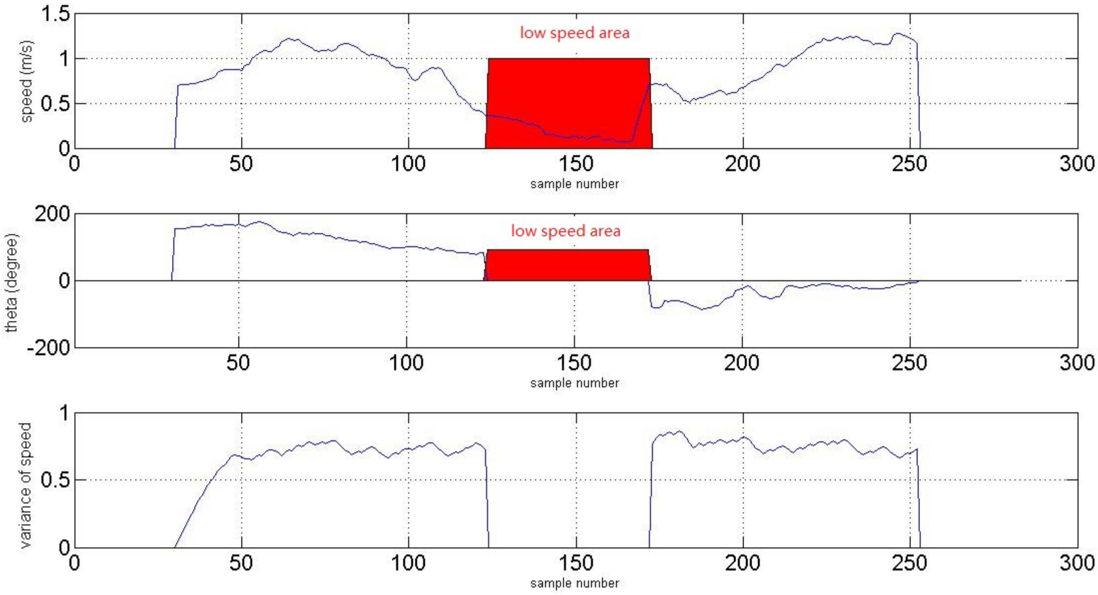

Figure 5.15 Speed, angle and speed variance profiles of algorithm with speed threshold. (With the process variance matrix Q tuned for the standard algorithm.)

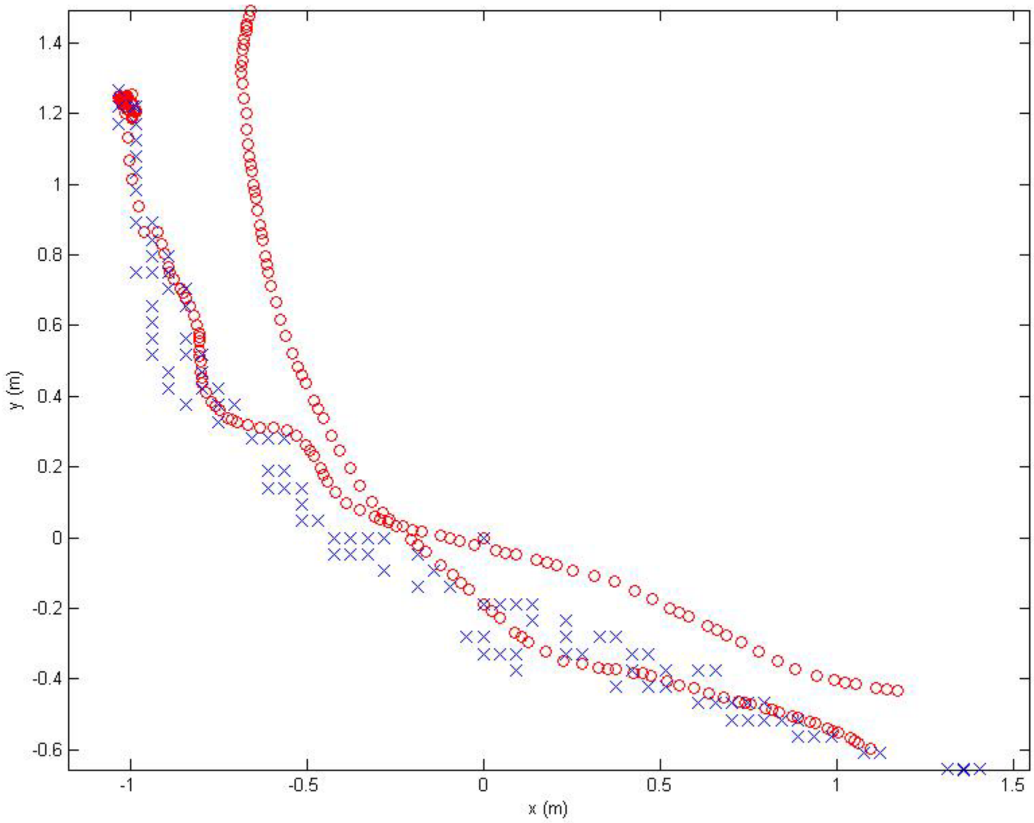

Figure 5.16 The trajectory was rebuilt based on the estimated velocity and angle.

In the comparison with the standard algorithm, the algorithm without a speed threshold provides a great variance of angle θ, which corrupts the velocity estimation via the unicycle model (2-10). The algorithm without a speed threshold can even be unstable (Figure 5.17), when started with a small tolerance, i.e. a small process variance which is tuned for the standard algorithm. In Figure 5.17, the algorithm loses the tracking of both angle θ and velocity after the first low speed area (v > 0.4 m/s). This behavior can be interpreted as: the unicycle model tries to track the states including the angle θ even in the low speed period. Thus it senses a great variation of θ. Because of a small process variance Q, the algorithm "trusts" more the model than the measurement. So the algorithm believes the great variation is caused by the measurement. In the correction step of the algorithm, the states are over estimated.

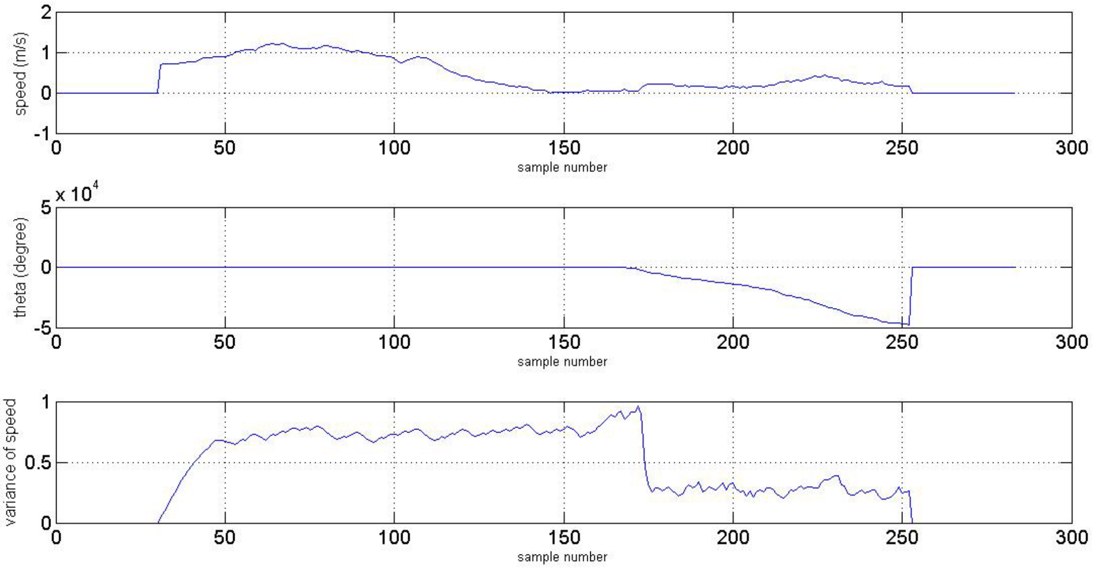

Figure 5.17 Speed, angle and speed variance profiles of algorithm without speed threshold. (With the process variance matrix Q tuned for the standard algorithm with a speed threshold.)

By enlarging the process variance matrix Q, the system can be stabilized with the cost of greater velocity variance and the inaccuracy of the estimation (Figure 5.18), i.e. the states are under estimated.

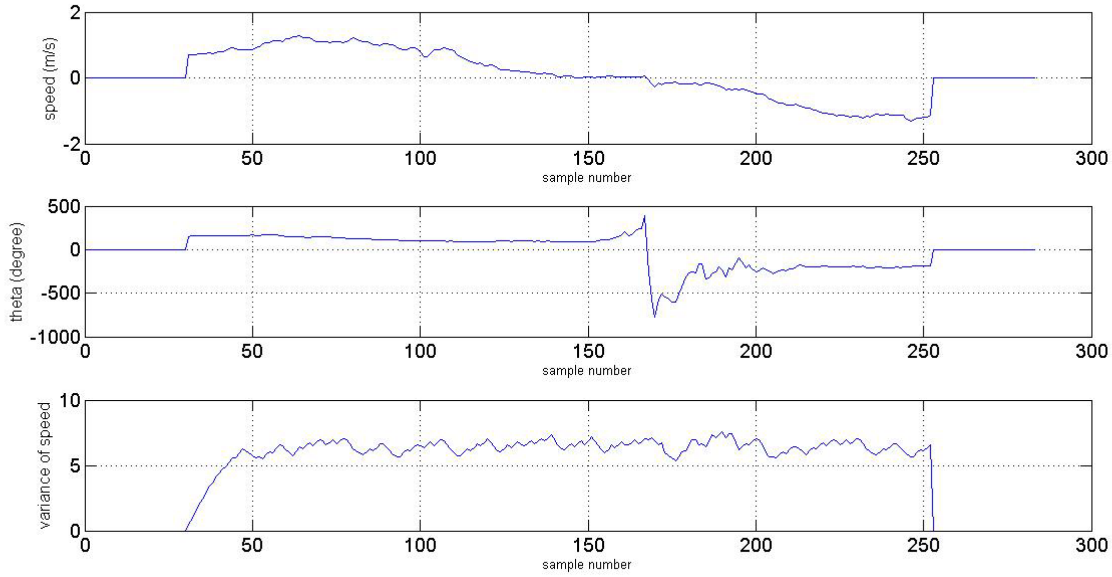

Figure 5.18 Speed, angle and speed variance profiles of algorithm without speed threshold. (With the process variance matrix Q tuned to stabilize the algorithm without a speed threshold.)

The result can be observed from the comparison, that the standard algorithm (with a speed threshold) can track the human locomotion properly even with some distortion from the low speed area. And the variance of the linear velocity is still in an acceptable range. The algorithm without a speed threshold can only track the human locomotion properly before the first time the speed is lower than approximately 0.3 m/s.

4. Tuning of the process variance Q

By the trial and error method, the process variance matrix was tuned for the standard algorithm based on the real trajectories captured by HDT. Additionally, the response speed of the extended Kalman filter has been considered. The previous described imaginary data sets were used as reference objects. Especially the variance of the linear velocity has a great impact on the response speed of the extended Kalman filter (Figure 5.19). There is a trade off in between the accuracy and the response speed of the algorithm.

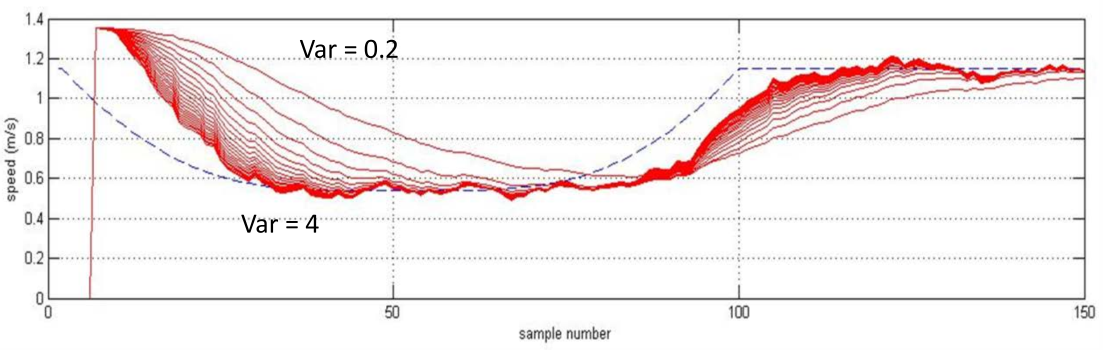

Figure 5.19 Sensitivity of the response speed with process variance of the linear velocity.

5. The reliability of the estimated states

Although the algorithm can track all the trajectories captured by HDT with a speed variance approximately $\sigma^2 \approx 0.8$, in the absence of a reference value to be compared with, the reliability of the estimated states can be validated in the sense of the consistency in between the measurements and the kinematic model. It is assumed that the kinematic model can imitate properly the behavior of human walking at a relative high linear velocity. If the estimation fits both the trajectory and the kinematic model well, i.e. in a small variance, it is considered at least as a stable version of extended Kalman filter.

Chapter 6 Conclusion and future work

Inspired by the ROSETTA project, an algorithm is proposed to estimate the human walking speed based on a visual tracking system. The core of the algorithm is an extended Kalman filter with some modification to be adapted to the scenario of ROSETTA.

From the modeling of human walking, a simplified kinematic model (unicycle) has been excerpted from previous work of ergonomic science. The velocity-curvature dependence was added to the model, which was meant to improve the model. Based on a visual tracking system developed by ROSETTA, an extended Kalman filter was applied to integrate the human walking model with the particle filter. The extended Kalman filter was modified to be adaptable to the experimental scenario. Finally the algorithm was tuned to be optimized with off-line data sets.

The algorithm was designed and tested in MATLAB®. The numerical simulations show that: the algorithm can track the human walking trajectory with a bounded process variance, and the estimation of the walking speed is reliable in the sense that the estimation fits both the measurement and the kinematic model of a walking human.

The modified extended Kalman filter has been tested to justify the modifications proposed in this thesis. The comparison demonstrates the effectiveness of the piece wise filtering with a speed threshold and the fusion technique applied in the algorithm. However, the velocity-curvature dependence embedded in the kinematic model did not give sufficient arguments during the validation to support its superiority.

Summing up, this thesis proposed a stable version of extended Kalman filter, which is dedicated to estimate the human walking velocity based on the visual tracking system subjected in ROSETTA project.

For the future work, the algorithm can be implemented into the real HDT-IE system, Combined with the motion plan of the industrial robot in the cell, an assessment of collision for safety could be obtained on-line. Surely, if it is possible, the algorithm should be validated again with reference to estimations from other tracking systems. And the velocity-curvature dependence is a promising topic in the intention estimation of human walking.

Appendix I MATLAB® script of the algorithm

```
Q = [     0.01        0          0        0         0        0
     0          0.01       0        0         0        0
     0          0          0.5      0         0        0
     0          0          0        0.5       0        0
     0          0          0        0         0.5      0
     0          0          0        0         0        0.05 ];
%process variance matrix for high speed

R = [    0.1       0
     0          0.1      ];
%measurement variance matrix for hight speed

Q_ls = [0.1        0          0          0
     0          0.1        0          0
     0          0          2          0
     0          0          0          0.5 ];
%process variance matrix for low speed

R_ls = [     0.5       0
     0          0.5      ];
%measurement variance matrix for low speed

%reserve space for estimated states and initialize
%the parameters
x_updated = zeros(9,size(data,2));
PP = zeros(6,6,size(data,2));
PP_ls = zeros(4,4,size(data,2));
TT = zeros(1,size(data,2));
entr_counter = 0;
exit_counter = 0;
present_flag = zeros(1,size(data,2)); %human presents in the area
P_buffer = zeros(6,6);
P_buffer_ls = zeros(4,4);
p_norm = zeros(1,size(data,2));
p_norm_ls = zeros(1,size(data,2));
low_flag = zeros(1,size(data,2));
fast_counter = 3;
low_counter = 0;
lowspeed = zeros(1,size(data,2));
lowspeed_counter = 0;
lost_tr = zeros(1,size(data,2));
```

```
weight = zeros(2,size(data,2));

speed_threshold = 0.4;      %     m/s     <<<<<<<<<<<<<<<<<<<<<<speed threshold



%main==========================================

for i=2:size(data,2)

    % finding entrance & exit
    %An algorithm to make sure, whether a human is in the area
    %A counter is set to count the sample number, whose reliability is beyond
    %the threshold. If 4 out 5 samples are beyond the threshold, the flag is
    %raised to indicate the present of a human.
    %The same algorithm is applied to detect the exit.
    if exit_counter >=9
        present_flag(1,i-4:i) = 0;
        exit_counter = 0;
    elseif exit_counter < 0;
        exit_counter = 0;
    end

    if entr_counter >= 5
        present_flag(1,i)= 1;
        entr_counter = 0;
    elseif entr_counter <0
        entr_counter = 0;
    end

    if present_flag(1,i-1) == 0 && data(5,i)>=0.9
        entr_counter = entr_counter+1;
    elseif present_flag(1,i-1) == 0 && data(5,i)<0.9
        entr_counter = entr_counter-1;
    end

    if present_flag(1,i-1) == 1 && data(5,i)>=0.9
        present_flag(1,i)= 1;
        exit_counter = exit_counter-1;
    elseif present_flag(1,i-1) == 1 && data(5,i)<0.9
        exit_counter = exit_counter+3;
    end


    if present_flag(1,i) == 1 %<------a human is detected, the filter is runing
```

```
% starting algorigthem

% initial guess of model

if present_flag(1,i-1) == 0
    %the first data of entrance,initialization is needed

    delta_x = data(3,i) - data(3,i-3);
    delta_y = data(4,i) - data(4,i-3);
    delta_s = sqrt(delta_x^2+delta_y^2);
    delta_t = data(1,i) - data(1,i-3);
    %The estimate of the initial states is based on the previous
    %valid data

    xp = data(3,i);
    yp = data(4,i);
    theta = atan2(data(4,i)-data(4,i-3),data(3,i)-data(3,i-3));
    k = 0.1;
    cn = 0.001;
    v = delta_s/delta_t;

    P_updated = P_buffer;
    P_updated_ls = P_buffer_ls;
end

% EKF !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

T = data(1,i) - data(1,i-1);
%time step of each sample, i.e. sampling time

t_scale = T/0.025;
%a time step scale to scale the covariance P

TT(1,i) = T;
%recording the time step
% speed threshold
%To stabilize the speed threshold, a flip flop with a tolerance is used to
%isolate the high/low speed interval

if low_flag(1,i-1) == 0 && abs(v)>= speed_threshold
    low_counter = low_counter - 1;
elseif low_flag(1,i-1) == 0 && abs(v)< speed_threshold
    low_counter = low_counter + 1;
```

```
end



if low_flag(1,i-1) == 1 && abs(v) >= speed_threshold
    fast_counter = fast_counter + 1;
elseif low_flag(1,i-1) == 1 && abs(v) < speed_threshold
    fast_counter = fast_counter - 1;
end

if fast_counter >=3 && low_flag(1,i-1) == 1
    low_flag(1,i) = 0;
    low_counter = 0;
elseif fast_counter >=3 && low_flag(1,i-1) == 0
    low_flag(1,i) = 0;
elseif fast_counter < 0;
    fast_counter = 0;
end

if low_counter >= 2 && low_flag(1,i-1) == 0
    low_flag(1,i)= 1;
    fast_counter = 0;
elseif low_counter >= 2 && low_flag(1,i-1) == 1
    low_flag(1,i)= 1;
elseif low_counter <0
    low_counter = 0;
end

if low_flag(1,i) == 0                       % highspeed
    % predict
    %prediction step
    alfa = 1;
    vv = abs(v);

    x_predicted = [ xp + T*v*cos(theta)
        yp + T*v*sin(theta)
        theta + T*k*v
        k - T*alfa/v^3*cn
        cn
        v                   ];
    %predicted estimate state

    df_dx = [1      0     -T*v*sin(theta)     0      0               T*cos(theta)
             0      1      T*v*cos(theta)     0      0               T*sin(theta)
             0      0      1                  T*v    0               T*k
```

```
          0       0     0            1     T*alfa/v^3      3*T*alfa*cn/v^4
          0       0     0            0     1               0
          0       0     0            0     0               1    ];
%Jacobian of model

P = df_dx * P_updated * df_dx' + Q        ;
P = sqrt(t_scale) * P;
%time scale to compentsate the variety of the time step

% update
dh_dx = [    1       0       0     0       0     0
     0       1       0     0       0     0       ];
%derivative of measurements respect to predicted states

z_predicted = dh_dx * x_predicted;

if data(5,i)<0.8
     outlayer = 0;
else
     outlayer = 1;
end
%checking unvalid data

meas_resi1 =     data([3,4],i) - z_predicted;
meas_resi2 =     data([6,7],i) - z_predicted;

weight(:,i) = [ data(5,i)/(data(5,i)+data(8,i))
     data(8,i)/(data(5,i)+data(8,i))];

W1 = 1/weight(1,i);
W2 = 1/weight(2,i);

K1 = P * dh_dx' / (W1*(dh_dx * P * dh_dx') + R);
K2 = P * dh_dx' / (W2*(dh_dx * P * dh_dx') + R);
K = P * dh_dx' / ((dh_dx * P * dh_dx') + R);

x_updated1 = x_predicted + outlayer * K1 * meas_resi1;
x_updated2 = x_predicted + outlayer * K2 * meas_resi2;

x_updated(1:6,i) = weight(1,i) * x_updated1 + weight(2,i) * x_updated2;

P_updated = (eye(6,6) - K * dh_dx) * P;
P_updated = P_updated / sqrt(t_scale);
xp =      x_updated(1,i);
```

```
    yp =      x_updated(2,i);
    theta = x_updated(3,i);
    k =       x_updated(4,i);
    cn = x_updated(5,i);
    v =       x_updated(6,i);

    PP(:,:,i) = P_updated;
    P_buffer = P_updated;

    % lowspeed
elseif low_flag(1,i) == 1
    %At low speed period, the states are estimated without curvature

    x_predicted_ls = [   xp + T*v*cos(theta)
        yp + T*v*sin(theta)
        theta
        v                        ];
    %predicted estimate state

    df_dx_ls = [     1         0         -T*v*sin(theta)      T*cos(theta)
        0         1         T*v*cos(theta)       T*sin(theta)
        0         0         1                    0
        0         0         0                    1                  ];
    %jacobian of model

    P_ls = df_dx_ls * P_updated_ls * df_dx_ls' + Q_ls;

    P_ls = sqrt(t_scale) * P_ls;

    % update
    dh_dx_ls = [     1         0         0      0
        0         1         0      0 ];

    z_predicted_ls = dh_dx_ls * x_predicted_ls;

    if data(5,i)<0.8
        outlayer = 0;
    else
        outlayer = 1;
    end

    meas_resi1_ls =     data([3,4],i) - z_predicted_ls;
    meas_resi2_ls =     data([6,7],i) - z_predicted_ls;
```

```
                    weight(:,i) = [ data(5,i)/(data(5,i)+data(8,i))
                        data(8,i)/(data(5,i)+data(8,i))];

                    W1 = 1/weight(1,i);
                    W2 = 1/weight(2,i);

                    K1_ls = P_ls * dh_dx_ls' / (W1*(dh_dx_ls * P_ls * dh_dx_ls') + R_ls);
                    K2_ls = P_ls * dh_dx_ls' / (W2*(dh_dx_ls * P_ls * dh_dx_ls') + R_ls);
                    K_ls = P_ls * dh_dx_ls' / ((dh_dx_ls * P_ls * dh_dx_ls') + R_ls);

                    x_updated1_ls = x_predicted_ls + outlayer * K1_ls * meas_resi1_ls;
                    x_updated2_ls = x_predicted_ls + outlayer * K2_ls * meas_resi2_ls;

                    x_updated([1:3,6],i) = weight(1,i) * x_updated1_ls + weight(2,i) * x_updated2_ls;

                    P_updated_ls = (eye(4,4) - K_ls * dh_dx_ls) * P_ls;
                    P_updated_ls = P_updated_ls / sqrt(t_scale);

                    xp =    x_updated(1,i);
                    yp =    x_updated(2,i);
                    if abs(v) >= 0.15
                        theta = x_updated(3,i);
                    elseif abs(v) < 0.15
                        theta = atan2(data(4,i)-data(4,i-2),data(3,i)-data(3,i-2));
                    end
                    %In case the speed is almost zero, the tracking of theta is
                    %totally lost. The angle have to be estimated geometrically.

                    k = 1/v;

                    v =     x_updated(6,i);

                    PP_ls(:,:,i) = P_updated_ls;
                    P_buffer_ls = P_updated_ls;


                end


            end


        end
```

Bibliography

[1]. Susanne Oberer-Treitz; Arnold Puzik; Alexander Verl; Fraunhofer-Institut für Produktionstechnik und Automatisierung IPA, Stuttgart. “Measuring the Collision Potential of Industrial Robots”, ISR / ROBOTIK , Germany, 2010

[2]. Gustavo Arechavaleta; Jean-Paul Laumond; Halim Hicheur; Alain Berthoz. “An Optimality Principle Governing Human Walking”. IEEE transactions on robotics, VOL. 24, NO. 1, February 2008

[3]. Stephane Vieilledent; Yves Kerlirzin; Stephane Dalbera; Alain Berthoz. “Relationship between velocity and curvature of a human locomotor trajectory” ,france, 2 April 2001

[4]. H. Hicheur, S. Vieilledent, M. J. E. Richardson, T. Flash, A. Berthoz. “Velocity and curvature in human locomotion along complex curved paths: a comparison with hand movements”, Experimental Brain Research Apri, Volume 162, Issue 2, page 145-154, 2005

[5]. Stephane Vieilledent; Yves Kerlirzin; Stephane Dalbera; Alain Berthoz. “Relationship between velocity and curvature of a human locomotor trajectory”, Neuroscience letters, vol. 305, no1, page 65-69 (24 ref.), 2001

[6]. S. Schaal; D. Sternad. “Origins and violations of the 2/3 power law in rhythmic three-dimensional movements”, Experimental Brain Research 136, page 60–72, 2001

[7]. Andrew G. Glena; Lawrence M. Leemisb; John H. Drewb. “Computing the distribution of the product of two continuous random variables”, Computational Statistics & Data Analysis Volume 44, Issue 3, Page 451–464, 1 January 2004

[8]. George W. Bohrnstedta; Arthur S. Goldbergerb. “On the Exact Covariance of Products of Random Variables”, Journal of the American Statistical Association Volume 64, Issue 328, pages 1439-1442, 1969

[9]. D. Willner C. B. Chang ; K. P. Dunn. “Kalman filter algorithm for a multisensor system”, Decision and Control including the 15th Symposium on Adaptive Processes, IEEE Conference, 1976

[10]. Shu-Li Sun; Zi-Li Deng. “Multi-sensor optimal information fusion Kalman filter” Automatica, Volume 40, Issue 6, June 2004, Pages 1017–1023

[11]. Wang, J.; Wilson, W.J. “3D relative position and orientation estimation using Kalman filter for robot control”, Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on 2638 - 2645 vol.3

[12]. Hakan Ardo; Kalle Agstrom. “Bayesian formulation of image patch matching using cross-correlation”. In Third ACM/IEEE International Conference on Distributed Smart Cameras, 2009.

[13]. OpenCV: http://opencv.willowgarage.com/wiki/.63

[14]. Liyuan Li; Weimin Huang; Irene Y. H. Gu; and Qi Tian. “Foreground object detection from videos containing complex background”. In Proceedings of the eleventh ACM inter-national conference on Multimedia, MULTIMEDIA ’03, pages 2–10, New York, NY, USA

[15]. P. KadewTraKuPong; R. Bowden. “An improved adaptive background mixture model for real-time tracking with shadow detection”. In Proceedings of the 2nd European Workshp on Advanced Video-Based Surveillance Systems (AVBS ’01)),NewYork, NY, USA, September 2001.

[16]. James Black; Dr. Tim Ellis. “Multi camera image tracking”, Image and Vision Computing, Volume 24, Issue 11, 1 November 2006, Pages 1256–1267

[17]. Chunhua Shen; Anton van den Hengel; and Anthony Dick. “Probabilistic multiple cue integration for particle filter based tracking”, Proc. VIIth Digital Image Computing: Techniques and Applications, Sun C., Talbot H., Ourselin S. and Adriaansen T. (Eds.), 10-12 Dec. 2003

[18]. Shaohua Kevin Zhou; R. Chellappa; B. Moghaddam. “Visual tracking and recognition using appearance-adaptive models in particle filters”, Image Processing, IEEE volume 13, issue 11, pages 1491-1506