Color-Based Fingertip Tracking Using Modified Dynamic Model Particle Filtering Method

Thesis

Presented in Partial Fulfillment of the Requirements for
the Degree Master of Science in the
Graduate School of The Ohio State University

By

Ting Zhu, B.S.

Graduate Program in Electrical & Computer Engineering

The Ohio State University

2011

Abstract

Various kinds of algorithms have been developed for object tracking, which can be divided into two categories, probabilistic and non-probabilistic, respectively. In either case, existing algorithms sometimes need to be improved to meet the challenges of a particular application, such as tracking abrupt motions of the target, changing lighting conditions of the environments, existing objects with similar appearance in the background, and etc. A good algorithm has to be robust for a particular application usually resulting in a trade-off between robustness and efficiency.

In our research topic, we have developed a system to efficiently track the motion of the tip of the index finger, for the purpose of replacing the mouse and pad of a computer for HCI. We call this setup Finger Mouse implementation. The fingertip is marked by red using an electrical tape, and the background is the surface of the desk where the computer lays. We have developed a modified priori motion model for the particle filtering algorithm based on the analysis of natural motion of human fingertip movement. Our high-order autoregressive model combined with temporal velocity performs more accurately and efficiently for fingertip tracking, compared with the existing methods.

The results of this research will be very useful. In addition to providing an alternative to healthy individuals, it is particularly suitable for disabled people who cannot mechanically move the mouse but use fingertip to express his/her intention of moving the cursor.

This is dedicated to my beloved parents.

Acknowledgments

I would like to give my sincerest gratitude to my advisor, Professor Yuan F. Zheng, for all the guidance he has given me throughout my years of study at The Ohio State University with his patience and knowledge. He was always there to offer inspirational discussions and precious opinions to me. He was always there to boost my confidence and encourage me to exploit anything of interest. He taught me how to approach the research related problems, and I always learned from his innovative solutions to many research problems. This thesis would not have been possible without his guidance and persistent help.

A special thanks to my committee member, Professor Bradley D. Clymer, for his valuable technical feedbacks and constructive insights.

I would also like to express my appreciation to my colleagues in The Multimedia and Robotics Laboratory at The Ohio State University. Special thanks should go to Mr. Junda Zhu, Mr. Liang Yuan, Mr. Chih-hsiang Chang, and Mr. Jatan Pandya to help me with many technical problems. My research has been a wonderful journey because of them.

I dedicate this thesis to my parents, who never doubted my dreams and have always been there for me. Without their constant encouragement and undying love, it would not have been possible for me to achieve my goals.

Vita

September 25, 1989	Born – Anqing, China
July, 2009	B.S. Electronic Eng. & Information Sci., Univ. of Sci. and Tech. of China
Sep 2009 to present	University Fellowship, Graduate Teaching Associate, Electrical & Computer Engineering, The Ohio State University, Columbus, OH

Fields of Study

Major Field: Electrical and Computer Engineering

Studies in:

Communication and Signal Processing

Image Processing

Semiconductor Devices

Table of Contents

Abstract ii

Acknowledgments v

Vita vi

List of Figures x

Chapter 1: Introduction 1

1.1 Purpose of the Research 2

1.2 Literature Review on Previous Work 2

1.3 A Brief Introduction of the Current Work 3

1.4 Thesis Overview 5

Chapter 2: The Approaches Used in Fingertip Tracking 7

2.1 Literature Review on Hand and Fingertip Tracking Algorithms 8

2.2 Principle of Particle Filtering 9

2.2.1 The Problem Description 9

2.2.2 The Kalman Filter 11

2.2.3 The Particle Filter 13

2.3 Discussion on the Performance of PF Algorithm.. 16

Chapter 3: Natural Motion of Human Fingertip Movement.. 19

3.1 Literature Review on Human Hand Motion Analysis.. 19

3.2 Natural Motion of Fingertip Movement... 22

3.3 Our Modified Dynamic Model in the PF Algorithm.. 24

3.3.1 Related Work... 24

3.3.2 Description of Our Approach 26

Chapter 4: Experimental Results 29

4.1 Tracking Setup and Initialization.. 29

4.2 Experimental Results.. 31

4.2.1 Comparison between Our Method and General PF.. 31

4.2.2 Comparison between Our Method and MS 43

4.2.3 Control Mouse Cursors.. 47

Chapter 5: Contributions and Future Work 51

5.1 Contributions.. 51

5.2 Future Works.. 52

Bibliography 53

List of Figures

Figure 1: A system overview 5

Figure 2: The process of the PF algorithm.. 11

Figure 3: Three steps of PF method 17

Figure 4[49]: An example to analyze writing of letter A (Continued) 23

Figure 5: Physical setup 30

Figure 6: Tracking initialization 30

Figure 7: Comparison of the tracking performance using three different PF algorithms. Frames 5, 50, 100, 150, 200, 250, 300 and 350 are shown. The trajectory of the fingertip is a number “518”. (Continued) 32

Figure 8: Quantitative comparison of three different PF algorithms of the first video sequence, 50 particles 36

Figure 9: First-order with velocity model, 200 particles 37

Figure 10: Comparison of the tracking performance using three different PF algorithms. Frames 5, 100, 200, 300, 400and 500 are shown. The trajectory of the fingertip is a circle, and the finger moves faster than normal speed. (Continued) 39

Figure 11: Quantitative comparison of three different PF algorithms of the second video sequence 42

Figure 12: Comparison between MS and our method. Frames 40, 41, 42 and 43 are shown. (Continued) 44

Figure 13: Quantitative comparison between MS and our method. 46

Figure 14: The PF can track the fingertip well in the whole sequence. The frames 100, 150, 200 and 250 are shown. 47

Figure 15: An example trajectory (“OSU”) of the mouse cursor 49

Figure 16: Desktop display of the cursor movement 50

Chapter 1: Introduction

In recent years, computer vision has played a significant role in Human-Computer Interaction (HCI). With the help of the availability and popularity of video cameras in everyday life, a huge volume of video data can be recorded and processed to computers to achieve HCI. There have been a large number of researches [1-5] conducted to substitute conventional devices currently used for HCI such as mouse and keyboard with vision-based natural interface according to the motion performed by human operators such as moving a finger in the air instead of the mouse. Such achievements rely largely on the technique of video tracking, or so-called object tracking of the motion of the finger while the video camera is assumed to be the sensor. Only when the motion is accurately tracked and the intended information of the operator is learned can we use vision-based technologies to interact with computers. As a result, the method of video tracking becomes the major component of a successful vision-based-HCI approach. This thesis is devoted to a new technology for visual tracking of a fingertip, which is to replace the mouse and pad for moving the cursor of a computer.

In the following we will first formally state the purpose of the research, which is followed by the literature review of the previous work, the introduction of the current work, and the overview of this thesis.

1.1 Purpose of the Research

Various kinds of algorithms have been developed for object tracking, and they can be mainly divided into probabilistic methods and non-probabilistic methods. In either case, existing algorithms sometimes need to be improved to meet the challenges of a particular application, such as tracking abrupt motions of the target, changing lighting conditions of the environments, existing objects with similar appearance in the background, and so on. A good algorithm has to be robust for a particular application usually resulting in a trade-off between robustness and efficiency.

In our research topic, we have developed a system to efficiently track the motion of the tip of the index finger, for the purpose of replacing the mouse and pad of a computer for HCI. We can call this setup Finger Mouse implementation. The fingertip is marked by red using an electrical tape, and the background is the surface of the desk where the computer lays. (See Figure 5 for the setup). The results of this research will be very useful. In addition to providing an alternative to healthy individuals, it is particularly suitable for disabled people who cannot mechanically move the mouse but use fingertip to express his/her intention of moving the cursor. On the other hand, such fingertip tracking results can be used to identify sign languages when deaf people want to express their feelings through the hand gestures.

1.2 Literature Review on Previous Work

One of the earliest system in HCI device can be traced back to “DigitEyes” [1] proposed by Rehg and Kanade, where they built a model-based hand tracking system using two cameras to recover the state of a hand. Kjeldsen and Kender [2] conducted a simple visual gesture system to manipulate Graphic User Interface (GUI) of a computer. Andrew [6] presented a method for tracking the 3D position of a finger, using a single camera which is placed several meters away from the user. Among these three applications, DigitEyes is based on a 3D model of a human hand and computationally very expensive. The performances of the other two approaches are not robust to complex backgrounds.

Jiyoung and Juneho [7] have proposed a similar idea as ours, where they designed a working system that visually recognize hand gestures of the user for the control of a window based user interface. Their tracking method is based on CAMSHIFT algorithm [8] which can track well particular hand poses in complex backgrounds. They also described how the location of the fingertip is mapped to the cursor location on the computer. However, one big defect of CAMSHIFT algorithm is that they cannot handle fast motion pattern, and an assumption must be made that the system is evolved relatively slowly and the displacement of the target between two consecutive frames is relatively small. Such challenges have attracted us to promote ourselves into the research studies presented in this thesis.

1.3 A Brief Introduction of the Current Work

Based on the previous works, we designed our own system to track the fingertip movement and replace the mouse with the index finger in a computer, whose main features can be stated as follows.

Firstly, we recorded videos of human finger movement and analyzed the natural motion of fingertip trajectory, getting the conclusion that the fingertip moves nonlinearly and changes its velocity and direction irregularly, so that it’s impossible to represent fingertip movement using a unified motion pattern.

Secondly, we proposed a modified-model particle filtering (PF) method which is particularly tuned to track the fingertip according to pre-learning of the natural motion of human finger movement. As we know, PF is a technique for implementing recursive Bayesian filtering by Monte Carlo sampling [10], and the basic idea of PF method is to represent the posterior density by a set of random particles with associated weights, and compute fingertip state estimation based on the samples and weights. In order to realize the purpose, we need to define the priori motion model for the fingertip state first, which is very important and directly affect the accuracy and efficiency of the tracking results. If the priori model fits the real fingertip movement closely, the particles can distribute according to a small covariance, and we can get the weighted mean value which is close to the real fingertip state, so that the particle numbers can be reduced accordingly; otherwise, if the priori model is not chosen properly, the particles will have a large covariance, increasing the tracking errors inevitably, and in order to prevent tracking performance degrading, the only way is to employ a large number of particles, which will cause large computational effort. Through the learning of natural motion of fingertip movement, we performed a high-order autoregressive priori model by combining temporal velocity information, which improved the tracking performance greatly as well as saving the computational time.

Thirdly, we computed the mapping between the camera coordinates and monitor coordinates and smoothed the cursor path, to give the user a natural feeling of moving a cursor on the screen.

Figure 1 gives a system overview.

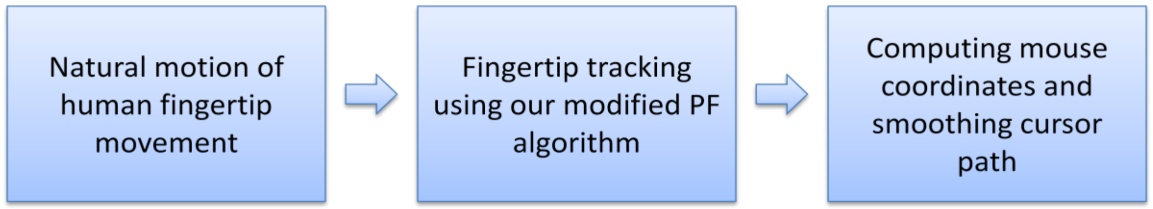

Natural motion of human fingertip movement
Fingertip tracking using our modified PF algorithm
Computing mouse coordinates and smoothing cursor path

Figure 1: A system overview

1.4 Thesis Overview

The rest of the thesis is organized as follows. In Chapter 2, we first give a brief review on the current different approaches used in finger tracking, which can be divided into probabilistic and non-probabilistic methods. We then focus on the introduction of the preliminary background of the algorithm we used, the PF method, and the probabilistic frameworks that our research is built on. At the end of this chapter we concluded the three steps of PF algorithm and illustrated the possible improvements we can make on the algorithm, so that the tracking results will be more robust and computationally effective.

In Chapter 3, we move on to the analysis of natural motion of human hand and fingertip movement. Since the finger can move arbitrarily in real applications, it’s impossible to define a unified motion pattern for the trajectory, though we can still make the assumption that the movement of human finger will follow smooth path and cover small distance in small amount of time. We still use the idea of PF method in our human fingertip tracking application. For the purpose of establishing a dynamic motion model of the fingertip state which can provide more accurate priori information, we redefined the PF algorithm by modifying the priori motion model from the general first-order model with zero-mean Gaussian distribution, to a high-order autoregressive model combining with the temporal velocity information in the Gaussian distribution, and tested the new algorithm in our video sequences.

Chapter 4 describes the various experiments we conducted to test our modified dynamic model embedded in PF method. We also made comparisons between our new method with the general PF method as well as with the mean shift algorithm. Experimental results show that such new PF model works more accurately and efficiently for our case when the fingertip moves arbitrarily and quickly. Then the fingertip locations are smoothed to be used for the control of mouse cursor on the monitor.

Chapter 5 concludes this thesis and provides some ideas on future improvements of the Finger Mouse application.

Chapter 2: The Approaches Used in Fingertip Tracking

How to track objects robustly and efficiently in complex environment has become a challenging issue in the field of computer vision. Overall, there are mainly two kinds of successful approaches in the pursuit of robust tracking, one is probabilistic (statistical) methods, and the other is deterministic methods. [9] The probabilistic methods explicitly take the object measurement and take uncertainties into consideration to establish correspondence. Two typical probabilistic methods are Kalman Filter, tracking objects which have linear state and the noise process is Gaussian, and Particle Filter, used more general in cases where object state is nonlinear and noise process is not assumed to be Gaussian. The deterministic methods define a cost of associating each object in the previous frame to the current frame using a set of motion constraints. Minimization of the correspondence cost is formulated as an optimization problem. Mean Shift [54] is such a typical method which finds the best correspondence between the target candidate and target model based on the density gradient estimation. In general, the former is stochastic and model-driven while the latter is deterministic and data-driven.

In this chapter, we first give a brief review on the current different approaches used in fingertip tracking, and then move on to give a detail representation of the particle filtering method we used, which is followed by a conclusion of the three steps of the particle filtering algorithm and illustration of the possible improvements we can make on the algorithm, so that the tracking results will be more robust and computationally effective.

2.1 Literature Review on Hand and Fingertip Tracking Algorithms

Hand tracking has been an active area of research in computer vision community for a long time, mainly for the purpose of HCI and sign language recognition. Researchers have developed various methods in hand tracking field. One of the original tracking systems to focus on articulated hand motion was presented in [1] by Rehg and Kanade. In their “DigitEyes” system, a 27 degree-of-freedom (DOF) human hand can be tracked at the rate of 10 Hz by extracting point and line features from gray scale images. However, it has difficulty tracking in occluded and complex backgrounds, as well as computationally expensive.

From an interaction perspective, most of the hand tracking applications has focused on 2D interfaces. In [11], the authors used low-cost web cameras to track a finger across a visual panel to manipulate a traditional graphic interface without using a mouse or keyboard. The concepts of virtual mouse and virtual keyboard came up for the first time. They used the method of Kalman filtering to accomplish the task of local tracking of fingertip, and detected the tip by fitting a conic to rounded feature.

Similarly, in [12], infrared cameras were used to segment skin regions from background pixels for the purpose of tracking two hands for interaction on a 2D desktop display. They used a template matching approach to recognize a small set of hand gestures that can be interpreted as interface commands. While in their system, no precise fingertip location information was obtained.

Our fingertip tracking system is primarily based on the particle filtering algorithm, which is robustly and widely used in dealing with nonlinear and uncertain motion. We will provide detail information of the algorithm in the coming subsection.

2.2 Principle of Particle Filtering

We begin in this section with a description of nonlinear tracking problem and its optimal Bayesian solution. The basic goal is to estimate a stochastic process given some noisy observations. When certain constraints hold, the optimal solution is tractable. And in this case, Kalman Filter can be used to solve the problem. More often is the case when the optimal solution is intractable. That's why we need to talk about Particle Filter.

2.2.1 The Problem Description

In order to track a fingertip in real applications, we need at least two models. The first is a motion model, which describes the evolution of the state of the fingertip with time. The second is a measurement model relating the noisy measurements to the fingertip state.

Let us first consider the evolution of the state sequence of the fingertip given by

$$x_k = f_{k-1}(x_{k-1}, v_{k-1}) \tag{2.1}$$

where f_{k-1} is a known, possibly nonlinear function of the state x_{k-1} and v_{k-1} is referred to as a process noise sequence. Process noise includes any mismodeling effects or unforeseen disturbances in the fingertip motion model.

The objective of nonlinear filtering is to recursively estimate x_k from measurements z_k. The measurement model can be represented as

$$z_k = h_k(x_k, w_k) \tag{2.2}$$

where h_k is a known, possibly nonlinear function and w_k is a measurement noise sequence.

The noise sequences v_{k-1} and w_k are assumed to be white, with known probability density functions and mutually independent. The initial fingertip state is assumed to have a known pdf $p(x_0)$ and also to be independent of noise sequences.

We denote $Z_k \triangleq \{z_i, i = 1, \cdots k\}$ as the sequence of all available measurements up to time k. From a Bayesian perspective, the problem is to construct the posterior pdf $p(x_k|Z_k)$, and estimate the state x_k at time k based on Z_k. Then in principle, the pdf $p(x_k|Z_k)$ is obtained recursively in two stages: prediction and update.

Suppose that the required pdf $p(x_{k-1}|Z_{k-1})$ at time $k-1$ is available. The prediction stage involves using the motion model (2.1) to obtain the prediction density at time k via the Chapman-Kolmogorov equation [10]:

$$p(x_k|Z_{k-1}) = \int p(x_k|x_{k-1})\, p(x_{k-1}|Z_{k-1}) dx_{k-1} \tag{2.3}$$

Here we write $p(x_k|x_{k-1}, Z_{k-1}) = p(x_k|x_{k-1})$ due to the fact that (2.1) describes a Markov process of order one, by which we mean that the current fingertip state is only related to the previous state and has nothing to do with the earlier states.

At time k when a measurement z_k becomes available, the update stage is used, which involves an update of the prediction pdf via the Bayes' rule:

$$p(x_k|Z_k) = p(x_k|z_k, Z_{k-1})$$

$$= \frac{p(z_k|x_k, Z_{k-1})p(x_k|Z_{k-1})}{p(z_k|Z_{k-1})}$$

$$= \frac{p(z_k|x_k)p(x_k|Z_{k-1})}{p(z_k|Z_{k-1})}$$

$$= \frac{p(z_k|x_k)p(x_k|Z_{k-1})}{\int p(z_k|x_k)p(x_k|Z_{k-1})\,dx_k} \tag{2.4}$$

where $p(z_k|x_k)$ is the likelihood function defined by the measurement model (2.2) and the known statistics of w_k. We can use the following Figure 2 to represent the whole process:

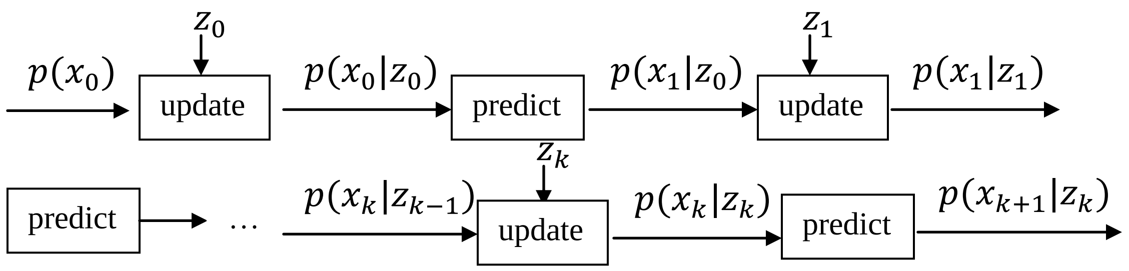


Figure 2: The process of the PF algorithm

2.2.2 The Kalman Filter

The Kalman filter is proposed to solve problems when the posterior density at every time step is Gaussian and the two model functions are linear. Certain assumptions are hold: v_{k-1} and w_k are distributed according to Gaussian densities of known parameters; $f_{k-1}(x_{k-1}, v_{k-1})$ is a known linear function of x_{k-1} and v_{k-1}; $h_k(x_k, w_k)$ is a known linear function of x_k and w_k.

Thus, (2.1) and (2.2) can be written as

$$x_k = F_{k-1}x_{k-1} + v_{k-1} \tag{2.5}$$

$$z_k = H_k x_k + w_k \tag{2.6}$$

where F_{k-1} and H_k are known matrices defining the linear functions. The covariances of random sequences v_{k-1} and w_k are Q_{k-1} and R_k, respectively. Then (2.3) and (2.4) can be derived as

$$p(x_{k-1}|Z_{k-1}) = \mathcal{N}(x_{k-1}; m_{k-1|k-1}, P_{k-1|k-1}) \tag{2.7}$$

$$p(x_k|Z_{k-1}) = \mathcal{N}\left(x_k; m_{k|k-1}, P_{k|k-1}\right) \tag{2.8}$$

$$p(x_k|Z_k) = \mathcal{N}(x_k; m_{k|k}, P_{k|k}) \tag{2.9}$$

where

$$m_{k|k-1} = F_{k-1}m_{k-1|k-1} \tag{2.10}$$

$$P_{k|k-1} = Q_{k-1} + F_{k-1}P_{k-1|k-1}F_{k-1}^T \tag{2.11}$$

$$m_{k|k} = m_{k|k-1} + K_k(z_k - H_k m_{k|k-1}) \tag{2.12}$$

$$P_{k|k} = P_{k|k-1} - K_k S_k K_k^T \tag{2.13}$$

where $\mathcal{N}(x; m, P)$ is a Gaussian density with argument x, mean m and covariance P, which is

$$\mathcal{N}(x; m, P) \triangleq |2\pi P|^{-1/2} exp\,\{-\frac{1}{2}(x-m)^T P^{-1}(x-m)\} \tag{2.14}$$

and

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{2.15}$$

is the covariance of the term $z_k - H_k m_{k|k-1}$, and

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{2.16}$$

is the Kalman gain. In the above equations, M^T denotes the transpose of a matrix M.

The Kalman filter is the optimal solution to the tracking problems holding linear and Gaussian environment.

2.2.3 The Particle Filter

In many practical situations, the linear and Gaussian assumptions made above do not hold. Researchers have proposed many approximate nonlinear Bayesian filters [23], such as extended Kalman Filter (EKF), approximate grid-based methods and Particle Filter (PF).

Generally speaking, PF is a technique for implementing recursive Bayesian filtering by Monte Carlo sampling. The basic idea is to represent the posterior density by a set of random particles with associated weights, and compute estimation based on these samples and weights. The reason to calculate the posterior density is to get the mean of the state using Monte Carlo integration.

Suppose we want to evaluate an integral

$$I = \int g(x)dx \tag{2.17}$$

Monte Carlo (MC) methods for numerical integration factorize $g(x) = f(x) \cdot \pi(x)$ such that $\pi(x)$ is interpreted as a probability density satisfying $\pi(x) \geq 0$ and $\int \pi(x)dx = 1$. In this sense, we can draw $N \gg 1$ samples $\{x^i; i = 1, \cdots, N\}$ distributed according to $\pi(x)$. The MC estimation of the integral

$$I = \int f(x) \cdot \pi(x)dx \tag{2.18}$$

is the sample mean

$$I_N = \frac{1}{N}\sum_{i=1}^{N} f(x^i) \tag{2.19}$$

In the Bayesian estimation, density $\pi(x)$ is the posterior density. Suppose we can only generate samples from a density $q(x)$, which is similar to $\pi(x)$. We call this $q(x)$ as importance or proposal density. Then we can generate $N \gg 1$ independent samples $\{x^i; i = 1, \cdots, N\}$ distributed according to $q(x)$ and form the weighted sum

$$I_N = \frac{1}{N}\sum_{i=1}^{N} f(x^i)\,\tilde{w}(x^i) \tag{2.20}$$

where

$$\tilde{w}(x^i) = \frac{\pi(x^i)}{q(x^i)} \tag{2.21}$$

are the importance weights.

Such importance sampling method is a general MC integration method, and the resulting sequential importance sampling (SIS) is known widely as bootstrap filtering [13], the condensation algorithm [14][15], particle filtering [16], interacting particle approximations [17][18], and survival of the fittest [19].

As indicated in the previous section, here we define $X_k = \{x_j, j = 0, \cdots, k\}$, which represents all the fingertip states up to time k. In order to derive the equation for weight update, we can use Bayesian principle to get

$$p(X_k|Z_k) = \frac{p(z_k|X_k, Z_{k-1})p(X_k|Z_{k-1})}{p(z_k|Z_{k-1})}$$

$$= \frac{p(z_k|X_k, Z_{k-1})p(x_k|X_{k-1}, Z_{k-1})p(X_{k-1}|Z_{k-1})}{p(z_k|Z_{k-1})}$$

$$= \frac{p(z_k|x_k)p(x_k|x_{k-1})p(X_{k-1}|Z_{k-1})}{p(z_k|Z_{k-1})}$$

$$\propto p(z_k|x_k)p(x_k|x_{k-1})p(X_{k-1}|Z_{k-1}) \tag{2.22}$$

and we choose the importance density to factorize such that

$$q(X_k|Z_k) \triangleq q(x_k|X_{k-1}, Z_k)q(X_{k-1}|Z_{k-1}) \tag{2.23}$$

furthermore, the weights can be represented as

$$w_k^i \propto \frac{p\left(X_k^i \middle| Z_k\right)}{q\left(X_k^i \middle| Z_k\right)}$$

$$\propto \frac{p\left(z_k \middle| x_k^i\right)p\left(x_k^i \middle| x_{k-1}^i\right)p\left(X_{k-1}^i \middle| Z_{k-1}\right)}{q\left(X_k^i \middle| Z_k\right)}$$

$$= w_{k-1}^i \frac{p\left(z_k \middle| x_k^i\right)p\left(x_k^i \middle| x_{k-1}^i\right)}{q\left(x_k^i \middle| X_{k-1}^i, Z_k\right)} \tag{2.24}$$

If $q\left(x_k^i \middle| X_{k-1}^i, Z_k\right) = q\left(x_k^i \middle| x_{k-1}^i, z_k\right)$, and we choose the importance density function be equal with the transition priori, $q\left(x_k^i \middle| x_{k-1}^i, z_k\right) = p\left(x_k^i \middle| x_{k-1}^i\right)$, then the weight can be simplified as

$$w_k^i \propto w_{k-1}^i p\left(z_k \middle| x_k^i\right) \tag{2.25}$$

A common problem with the PF is the degeneracy phenomenon, where after a few iterations, all but few particles will have negligible weight. It has been show in [20] that the variance of the importance weights can only increase over time. This degeneracy will cause a large computation cost, which means that much computational effort is dedicated to updating particles whose contribution to the approximation to $p(X_k|Z_k)$ is almost zero. In this case, authors in [21] and [22] introduced a suitable measure of degeneracy of the algorithm according to the effective particle sample number N_{eff} according to the following equation

$$N_{eff} = \frac{1}{\sum_{i=1}^{N}(w_k^i)^2} \tag{2.26}$$

We can notice than when $w_k^i = 1/N$, that is, all particles are equally weighted, $N_{eff} = N$. In the common cases, $N_{eff} < N$. The ratio $\alpha = N_{eff}/N$ can be defined as the threshold to decide whether to resample or not. If α is too small, which means the effective particles are very few, then the tracking accuracy will be negatively affected and severe degeneracy came up.

In [23], the authors proposed two possible methods to solve the degeneracy problem. The first method involves choosing the importance density to maximize N_{eff}. As we stated before, we often choose $q\left(x_k^i \middle| x_{k-1}^i, z_k\right) = p\left(x_k^i \middle| x_{k-1}^i\right)$, since it is intuitive and simple to implement. The second method is to use resampling whenever a significant degeneracy is observed. The basic idea of resampling is to eliminate particles that have small weights and concentrate on particles which have large weights. We usually implement resampling by assigning equal weights to all particles, that is, $w_k^i = 1/N$.

2.3 Discussion on the Performance of PF Algorithm

Above all, a typical PF method is consisted of three steps as shown in Figure 3:

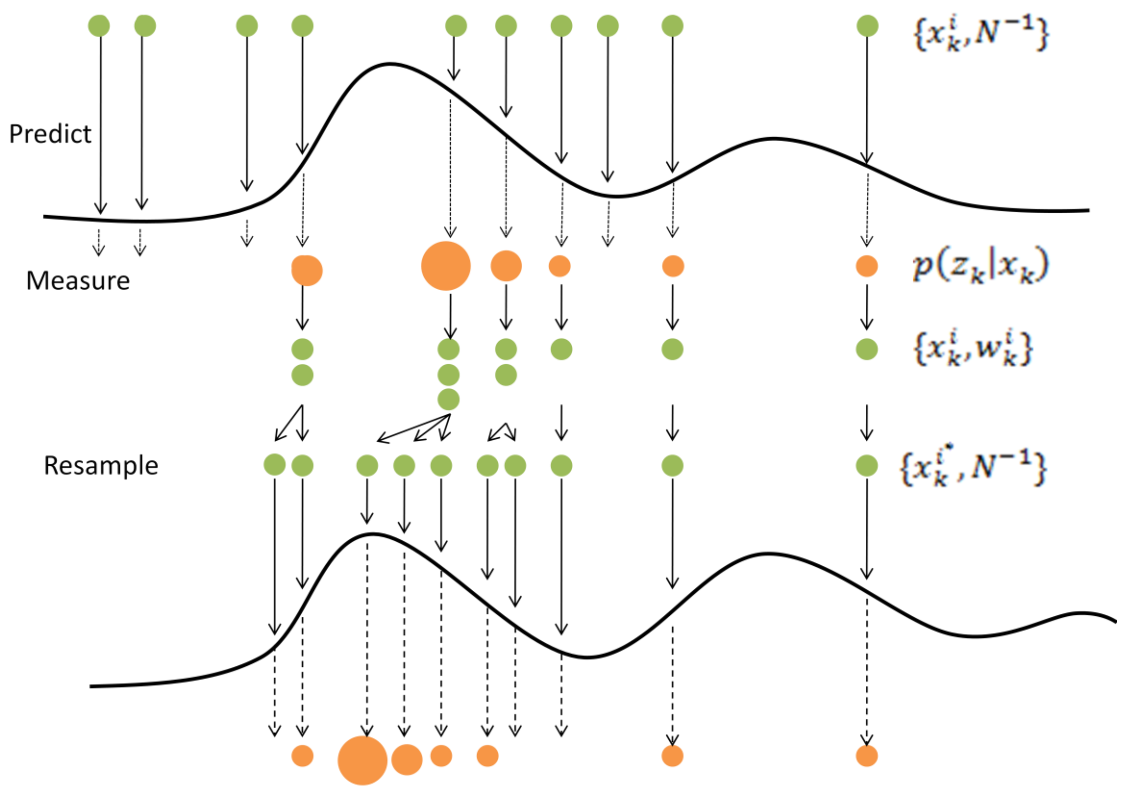


Figure 3: Three steps of PF method

As we can tell from Figure 3, based on the principle of PF algorithm, we need to estimate the posterior probability of the fingertip state based on the priori information and the measurement result.

In order to perform the algorithm, we first define the priori motion model for the fingertip state, which is very important and directly determines the region size that particles will spread, thus affecting the accuracy and efficiency of the tracking results. If the priori model fits the real fingertip movement closely, the particles can distribute according to a small variance, and we can get the weighted mean value which is close to the real fingertip state, so that the particle numbers can be reduced accordingly; otherwise, if the priori model is not chosen properly, the particles will have a large variance, increasing the tracking errors inevitably, and in order to prevent tracking performance degrading, the only way is to employ a large number of particles, which will cause large computational effort. In this sense, an adaptive priori model is preferred to accommodate more flexibility. Therefore, we proposed a high-order autoregressive priori model combining with the temporal velocity information in the Gaussian distribution, based on the pre-learning information of the natural motion pattern of the fingertip. The detail information is described in Chapter 3.

On the other hand, a good choice of importance density $q\left(x_k^i \middle| x_{k-1}^i, z_k\right)$ is a key component, and it is possible to construct suboptimal approximations to the optimal importance density by using local linearization techniques [24]. Due to convenience and simplicity, the prior motion model is selected as the importance density to propagate particles [23], which assumes that the particles in the previous frame can be efficiently moved to the current frame.

Measurement model is another key component. When we analyze a particular application, we usually need to select a good measurement model which has a high confidence on how discriminative this selected model will behave in the specific environment. As mentioned before, we mark the fingertip using a red electrical tape so that the area of the fingertip can be separated from the background (the desktop) according to the color information. This color-histogram-based measurement model works well when there is no color disturbance from the background environment and the mark is easy to implement.

Chapter 3: Natural Motion of Human Fingertip Movement

The PF algorithm mentioned in Chapter 2 is a robust method used in object tracking, and it’s also widely used in the hand tracking field by many researchers [25][26]. In our research topic, we also adopt PF in our red-colored fingertip tracking due to its robustness. As mentioned in section 2.3, the choice of priori model is a key component that we can make improvement on. If the priori model can be selected to fit the real fingertip movement well, the particles can distribute according to a small variance, so that the particle numbers can be reduced accordingly; otherwise, if the priori model is not chosen properly, the particles will have a large variance, increasing the tracking errors inevitably, and in order to prevent tracking performance degrading, the only way is to employ a large number of particles, which will cause large computational cost. In this sense, an adaptive priori model is preferred to accommodate more flexibility. Since the finger movement is non-rigid and arbitrary, in order to make the priori model more accurate, we need to do some training on the natural motion of human finger movement.

3.1 Literature Review on Human Hand Motion Analysis

In current virtual environments (VE) applications, keyboards, mice, and joysticks are the most fundamental, dominant and popular devices for controlling and navigation. However, they are unnatural and inconvenient. The use of hand gestures is a natural way for communication and many researchers have been dedicating their efforts on the development of intelligent human computer interaction systems [27] [28], where human gesture commands can be recognized by computers and computers can react to humans through diverse ways like synthesizing sign languages. For instance, HCI may facilitate the use of bare hands for direct control virtual objects in some virtual environments. [29] [30]

Several products, like glove-based devices [31], are used to capture human hand motion by attaching sensors to measure the spatial positions of users' hands in order to realize HCI. Though such methods have gained many prosperous results in hand tracking and motion detection, they are expensive and cumbersome. That's why researchers turned to non-contact vision-based technique as one of the promising alternatives to capture human hand motion. [32] Usually affordable camera settings will be needed as basic equipment for research in the modeling, analyzing, and recognition of hand gestures.

Capture hand motion from video sequences includes the estimation of the rigid global hand pose as well as the non-rigid finger articulation. One of the bottlenecks in hand motion tracking is that the fingers are articulated, which induces the high degrees of freedom (DoF). The human hand has roughly 27 DoF adding the rigid global hand motion. [33] Fortunately, the natural human motion is highly constrained and the motions among the joints are related. As a result, although the DoF is large, the hand motion can be constrained in a lower-dimensional subspace. While some simple and closed forms have been found and applied to hand motion analysis [34][35][36], the authors of [33] presented a novel approach to capturing articulated hand motion by learning and integrating natural hand motion priors. The approach consists of three significant components, the divide-and-conquer strategy, capturing the non-rigid finger articulation, and determining the rigid hand pose.

Generally speaking, there are two approaches exploded to capture the hand articulation. The first is the 3D model-based approach, taking advantage of 3D hand models and the second is the appearance-based approach, which directly associates 2D image features with hand configurations.

The 3D model-based approach obtains the hand motion parameters by aligning a 3D model and the observed images, and minimizing the difference between them. The problem becomes complicated when the dimension is high, and different image observations need to be made in order to construct correspondence between the model and the images. For example, the fingertips [36] [37] can be used as a feature to construct these correspondences, while at the same time the performance of fingertip detection becomes the determining source of accuracy and robustness. The use of line features was proposed in [38] to improve the performance.

The appearance-based approach estimates hand states directly from observed images by learning the mapping from image features to hand configurations. Some appearance-based methods were presented in [39] [40] to recover body postures. On the other hand, the method motion capture can be integrated into machine learning methods in human tracking area [41] [42]. Such appearance-based approach usually involves the collection of large sets of training data.

3.2 Natural Motion of Fingertip Movement

Human finger movement analysis is widely used in the design of robotic arms as part of bionics, and researchers also contributed a lot in this field. Robotics, especially personal robots [43], proposed with the increasing interest in health-care robotics and service applications [44] [45] [46], can be used in everyday life to serve, assist, and work together with human beings to make our life more convenient, effective and enjoyable. In order to operate the personal robots more effectively, authors in [47] put up with the modeling of primitive motion, which has a unified pattern and powerful enough so that the multiple primitive motions can form a complex path, and the pattern is similar to human motions. Fortunately, authors in [48] have already proved that human hand has a fundamental motion pattern, called the minimum-jerk (the rate of change of acceleration) model, which can be represented as:

$$x(t) = x(0) + [x(0) - x(T)](15\tau^4 - 6\tau^5 - 10\tau^3) \tag{3.1}$$

$$y(t) = y(0) + [y(0) - y(T)](15\tau^4 - 6\tau^5 - 10\tau^3) \tag{3.2}$$

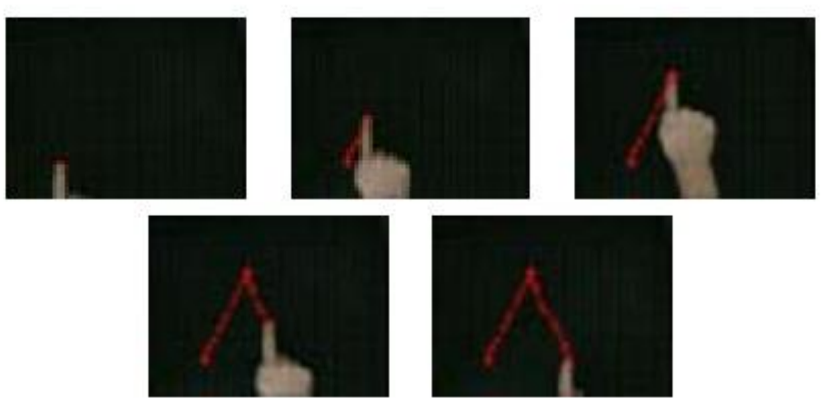
where $x(0)$ and $y(0)$ are the initial coordinates, $x(T)$ and $y(T)$ are the final coordinates, and $\tau = t/T$. Also the velocity of the primitive motion can be described by

$$v(t) = \frac{1}{T}[x(T) - x(0)](1 - cos2\pi\tau) \tag{3.3}$$

Such primitive motion works very well and a complicated path can be combined by the simple motion patterns. For example, a curved motion may be completed by moving right first followed by moving up and then moving right again.

Sometimes in the real applications we don’t need to analyze the whole hand movement, and we pay more attention to some specific finger’s trajectory. For instance, with the fast development of finger-touch electronic products, like iPhone, iPad, iTouch, and so on, we use one finger more often instead of the whole hand. In this sense, we don’t need to take the articulated fingers into consideration; we can only concern about fingertip trajectories.

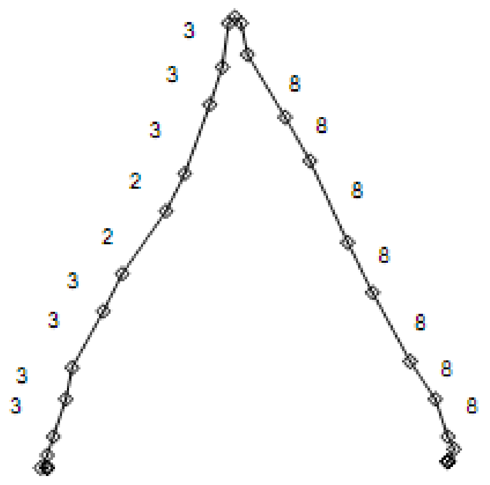
Taking advantage of the thoughts of modeling primitive motion in personal robots, we can divide human fingertip trajectory into several unit motion patterns although the fact that human finger can move in any arbitrary trajectory in real applications. Let’s take the handwriting of letters for instance. Authors in [49] proposed the method to use Hidden Markov Models (HMM) in sign language interpretation. They divided the writing of letters into several states. For example, letter A can be written as shown in Figure 4(a) and sectored as Figure 4(b), where the numbers represent different states.



(a) 5 images from a sequence where letter A is written

Figure 4[49]: An example to analyze writing of letter A (Continued)

Figure 4[49]: Continued



(b) Sectors associated to letter A's plotting

We can tell from the above figure that the writing of letter A can be divided into two sectors, and the connecting point of the two sectors can be called the turning point, where the direction of the trajectory is changed sharply. One of the great differences between these two sectors is that the curvature of the letter sketching is different. The change in curvature of the fingertip trajectory can be used as a standard in defining priori motion model, which is illustrated in the next subsection.

3.3 Our Modified Dynamic Model in the PF Algorithm

3.3.1 Related Work

From the above examples, we can conclude that the path of the fingertip cannot be modeled in one specific function, due to the fact that the fingertip may change its direction and velocity irregularly. As a result, we may not add any constraint on the path like the authors in [50] did.

In order to track the movement of a red-color-marked fingertip more accurately, we can include both the position and the velocity of the fingertip into the state component. Suppose the fingertip is represented by a rectangle window (the particle) which is initialized in the first frame, and the size of the window is constant since the distance from the finger to the camera can be kept as fixed. Then the dynamic state of the fingertip can be defined as

$$S_t = \{x_t, y_t, \dot{x}_t, \dot{y}_t\} \tag{3.4}$$

where $\{x_t, y_t\}$ is the position of the center of the rectangle in the image coordinate system at time t, and $\{\dot{x}_t, \dot{y}_t\}$ are the corresponding velocities. The width and height of the particles are determined in the first frame. As proposed in [51], the dynamic model could be learned from a set of pre-labeled training sequences, and previous researchers often adopt the weak constant velocity model as

$$x_t = Ax_{t-1} + v_{t-1} \tag{3.5}$$

$$y_t = Ay_{t-1} + w_{t-1} \tag{3.6}$$

where A defines the deterministic component of the model, and v_{t-1} and w_{t-1} are zero mean, Gaussian stochastic processes, added as the noise component in each coordinate which generates particles.

We can notice that if using the above model, we didn't include the component of the velocities into the update of the fingertip state. So authors in [50] put up with the idea to express the dynamic state model in the following form:

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{pmatrix} x_{t-1} \\ y_{t-1} \end{pmatrix} + \mathcal{N}[\begin{pmatrix} \dot{x}_t \Delta t \\ \dot{y}_t \Delta t \end{pmatrix}, \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}] \tag{3.7}$$

where $\mathcal{N}$ is a normal distribution, Δt is the time difference between two consecutive frames, and σ_x, σ_y are the variances of the distribution of x_t and y_t, which are selected empirically. Now the particles are generated according to a Gaussian process whose mean is related to the latest updated velocity.

3.3.2 Description of Our Approach

One defect of the above dynamic model in (3.7) lies in the fact that the movement of the fingertip is assumed to be linear, and the current fingertip state is only related to the previous one state. While as it was mentioned before, the fingertip often moves non-rigid and arbitrarily, so that such a first order model can no longer fit the moving trajectory well. The current fingertip state is not only related to the previous one state, but may also be related to several previous states, especially when there is a turning point in the movement. As a result, we expand the motion model to a high order, autoregressive one as follows:

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = A_n \begin{pmatrix} x_{t-n} \\ y_{t-n} \end{pmatrix} + \cdots + A_2 \begin{pmatrix} x_{t-2} \\ y_{t-2} \end{pmatrix} + A_1 \begin{pmatrix} x_{t-1} \\ y_{t-1} \end{pmatrix} + \mathcal{N}[\begin{pmatrix} \dot{x}_t \Delta t \\ \dot{y}_t \Delta t \end{pmatrix}, \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix}] \tag{3.8}$$

where (x_{t-n}, y_{t-n}) is the coordinate of the center of the rectangle window at time $t-n$. A_1 , $A_2 \dots$ A_n are the deterministic parameters in the model, and the current state is predicted based on the previous n states plus the Gaussian noise whose mean is related to the temporal velocity of the fingertip. The number of n depends largely on the curvature of the movement trajectory. If the fingertip moves in an asymptotically linear path, a first-order model will be enough; in this case we can set $A_1 = 1$ and all the other parameters to be zero. Then if the fingertip changes its direction to be parabolic, we can adaptively change to $A_1 = -1$, $A_2 = 2$ and all the other parameters to be zero. The key component is to know when there is a turning point and a new parameter combination should be used accordingly. As we stated before, we calculate the curvature of the trajectory in every two consecutive frames and compare it with the previous value to determine whether to change the parameters or not.

The velocities $\dot{x}_t$ and $\dot{y}_t$ are estimated numerically as

$$\dot{x}_t = (x_{t-1} - x_{t-2})/\Delta t \tag{3.9}$$

$$\dot{y}_t = (y_{t-1} - y_{t-2})/\Delta t \tag{3.10}$$

Using such modified dynamic motion model, we can get the motion priori information $p(S_t|S_{t-1}, S_{t-2}, \dots S_{t-n})$. Then the next step is to use the local observation of color histogram as the measurement model to update the fingertip state.

According to [52], the observation likelihood $p(z_t|S_t)$ must favor candidate color histogram $h(S_t)$ close to the reference histogram h_0. We therefore need to choose a distance D on the color distributions. Such a distance is widely used in the deterministic techniques [53] as the criterion to be minimized at each time slot. By gathering statistics on a number of sequences obtained from successful tracking runs, authors of [52] observed a consistent exponential behavior for the squared distance D^2. Thus the observation likelihood can be represented as

$$p(z_t|S_t) \propto e^{-\lambda D^2(h_0, h(S_t))} \tag{3.11}$$

where distance D is defined by the similarity function and has the following expression

$$D\big(h_0, h(S_t)\big) = \sqrt{1 - \rho[h_0, h(S_t)]} \tag{3.12}$$

There are many standards of histogram similarity function, such as the Kullback-Leibler divergence (KL), the Jeffrey-divergence (JD), Sum of Square Difference (SSD), and etc. Here we select the Bhattacharyya coefficient [54] defined as

$$\rho[h_0, h(S_t)] = \sum_{l=1}^{m} \sqrt{h(l; S_0) h(l; S_t)} \tag{3.13}$$

where $h(l; S_0)$ and $h(l; S_t)$ represent the color histogram value in lth bin of the reference model and the particle, respectively. λ is an empirical coefficient and usually fixed to the same value $\lambda = 20$. Each particle is weighted by its likelihood and a weighted sum of all the particles represents the fingertip state at time t:

$$\hat{S}_t = \int S_t \ p(S_t | z_{1:t}) dS_t \cong \sum_{i=1}^{N} w_t^i \, S_t^i \tag{3.14}$$

where w_t^i is the weight of the ith particle at time t, and N is the number of particles.

Chapter 4: Experimental Results

We have tested our color-based particle filtering method with modified dynamic motion model on some video sequences. In this section we compare the accuracy and efficiency of tracking results between our method and general PF method using different number of particles, and we also compare PF method with mean shift (MS) algorithm to illustrate the advantage of using PF method in finger tracking applications.

4.1 Tracking Setup and Initialization

The tester’s index is marked with a red-color electrical tape at first. Then we fix the background to be the desk with a single color like white. A Logitech Webcam C905 is used as the video camera to track the motion of the fingertip in real time, and mounted on the top of the computer monitor or hooked on the laptop screen. Our setup is shown in Figure 5.

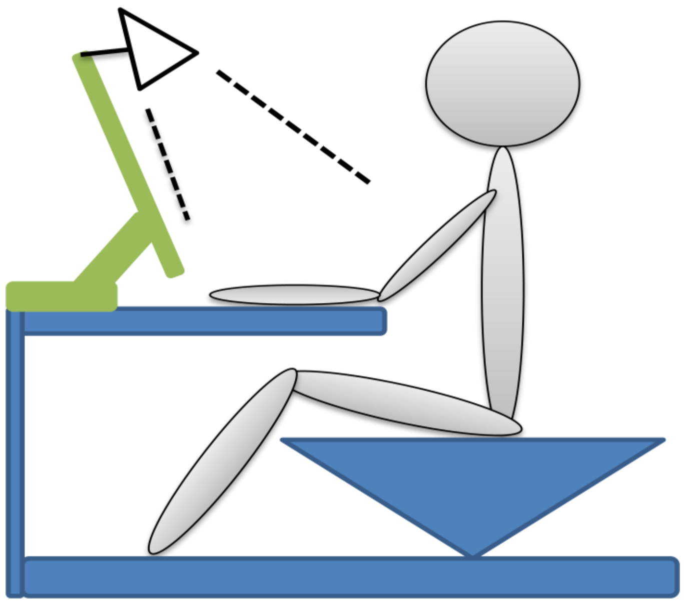


Figure 5: Physical setup

And the tracking initialization is shown in Figure 6.

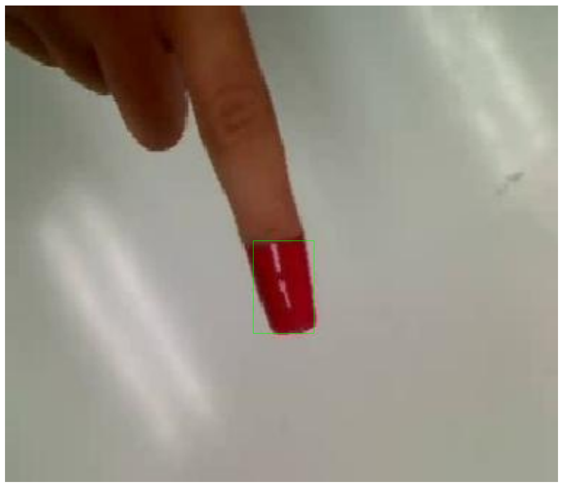


Figure 6: Tracking initialization

The image size is 352×240 pixels, and we use color histogram as the measurement model the fingertip state. In order to obtain the reference model, which defines the color histogram of the fingertip, we manually initialized the tracking window in the first frame by hand, usually 1-2 pixels wider than the fingertip.

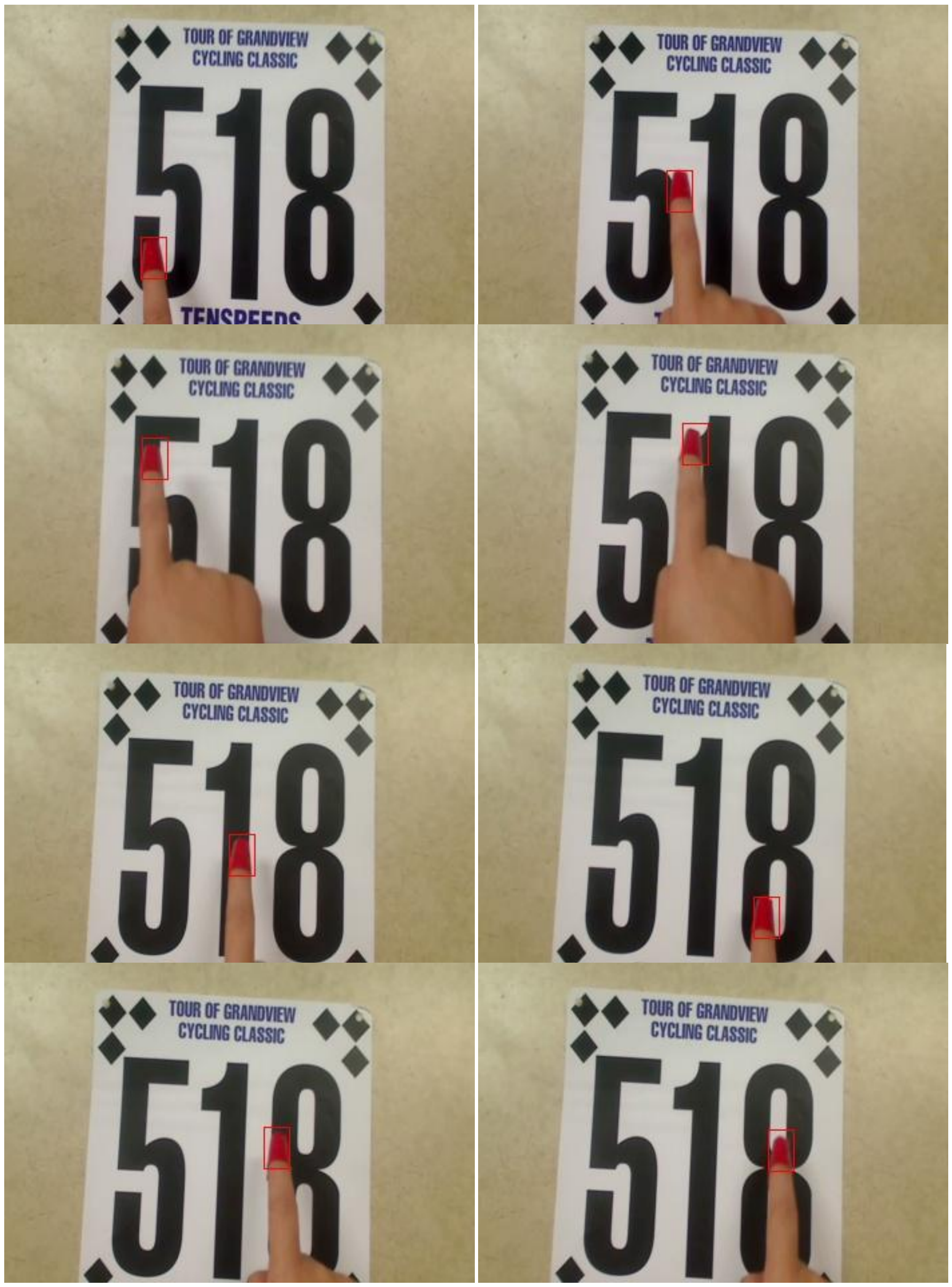
4.2 Experimental Results

We conducted general PF method and our modified model PF method in some video sequences, and compared and evaluated their performances. Then we compared PF with MS to analyze the advantages and disadvantages of these two widely-used tracking algorithms. Finally we performed some specific finger trajectories to illustrate the accuracy of our tracking results and controlled the mouse cursor in the monitor.

4.2.1 Comparison between Our Method and General PF

All of the following comparisons are consisted of three parts: the first is using our modified high-order motion model considering velocity update in particle generation; the second is using the first-order motion model with velocity update; and the third is the general PF using the first-order motion model without velocity update.

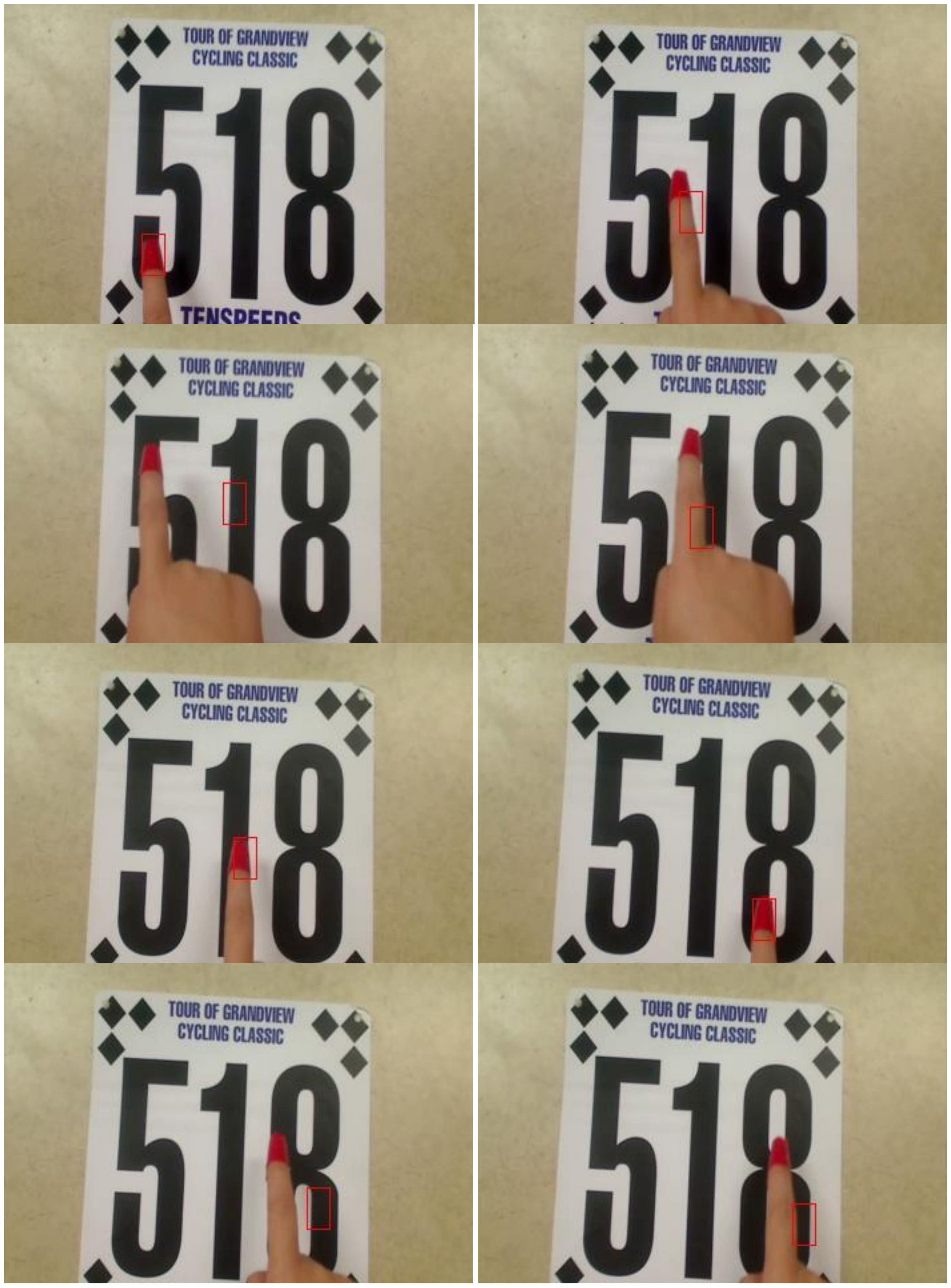
Figure 7 shows the tracking result of the movement of the fingertip whose trajectory is the writing of number “518”. Here we purposely move the fingertip very slowly. Firstly we use the same particle numbers and can tell that only our method can accurately track the fingertip location. The only way to increase the robustness of method (b) and (c) is to increase the particle numbers.



(a) Our high-order with velocity model, 50 particles

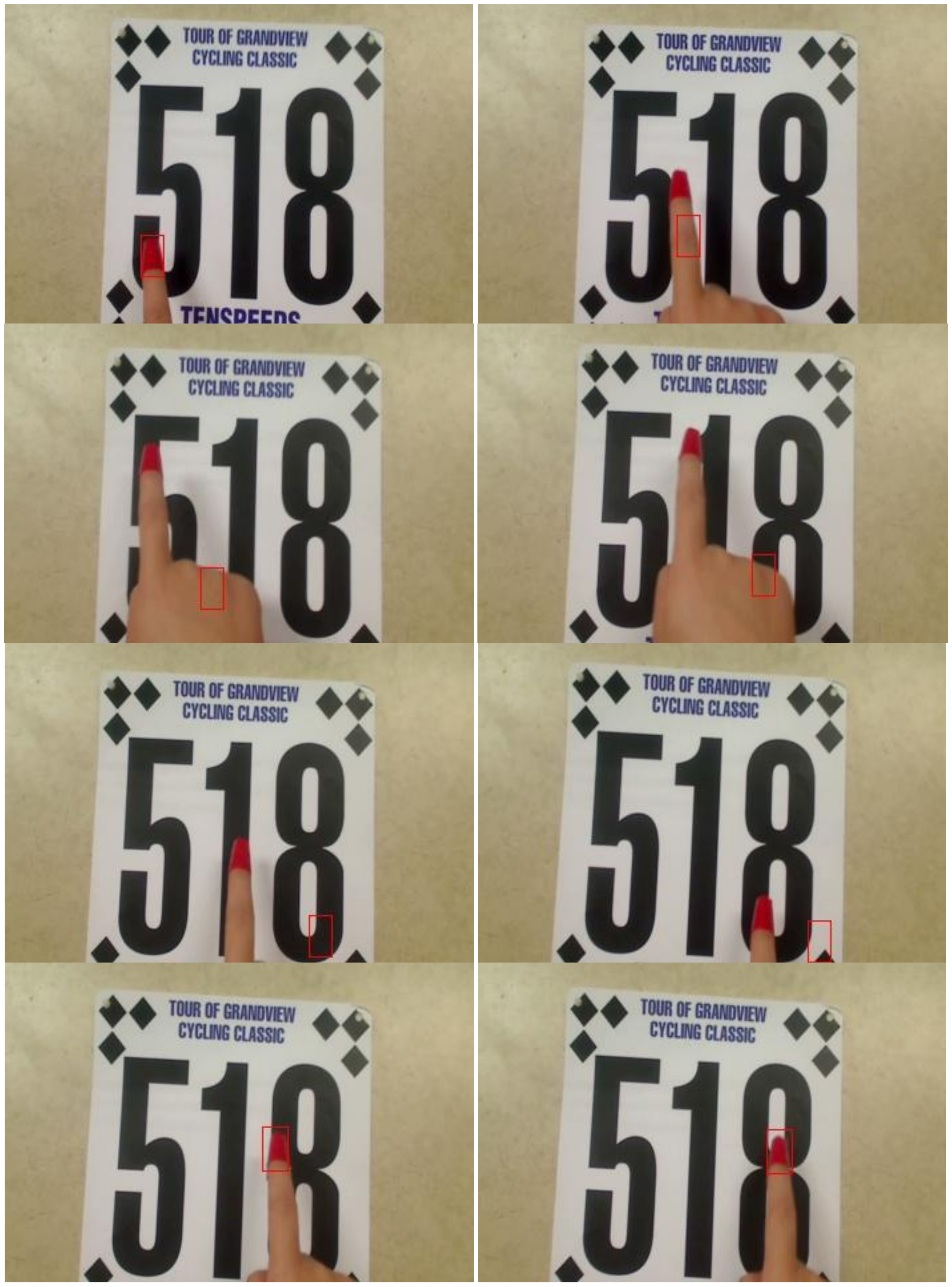
Figure 7: Comparison of the tracking performance using three different PF algorithms. Frames 5, 50, 100, 150, 200, 250, 300 and 350 are shown. The trajectory of the fingertip is a number “518”. (Continued)

Figure 7: Continued



(b) First-order with velocity model, 50 particles (Continued)

Figure 7: Continued



(c) General linear model, 50 particles

To further analyze the performance of our high-order with velocity model, the first-order with velocity model and the traditional first-order without velocity model, we repeated each of the three methods 50 times in the above tracking experiment and computed mean value of the pixel location of the center of tracking window in each frame. Here we conducted so many times due to the fact that PF algorithm is a stochastic method and the result obtained every time has the random characteristic, while getting a mean value of all the results makes sense from a probability perspective. In order to compare the accuracy of the tracking results, we first manually tracked the position of the fingertip in each frame, and then defined the tracking error as the pixel distance from the actual location and the estimated location. Figure 8 shows the comparison of the tracking error using these three methods.

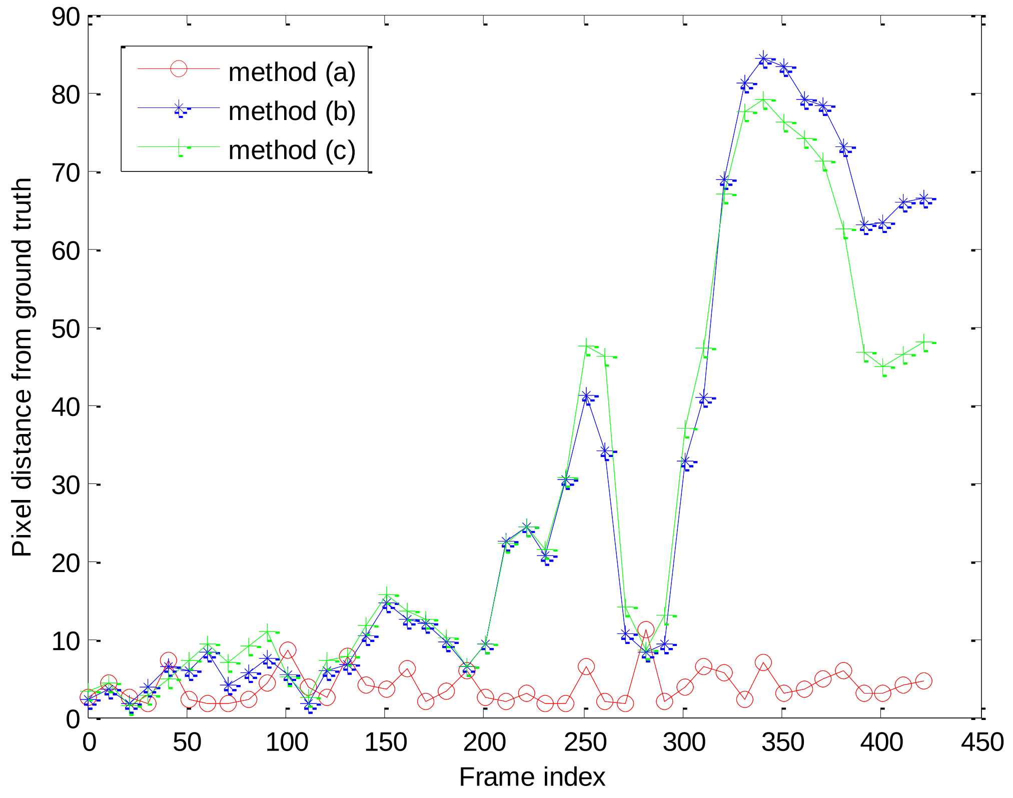


Figure 8: Quantitative comparison of three different PF algorithms of the first video sequence, 50 particles

We can tell from the above figure that our high-order model (method (a)) has much less error than the first-order models (method (b) and (c)), when the number of particles is the same. Only when we increase the particle numbers to 200 using method (b), can we get accurate tracking result as shown in Figure 9.

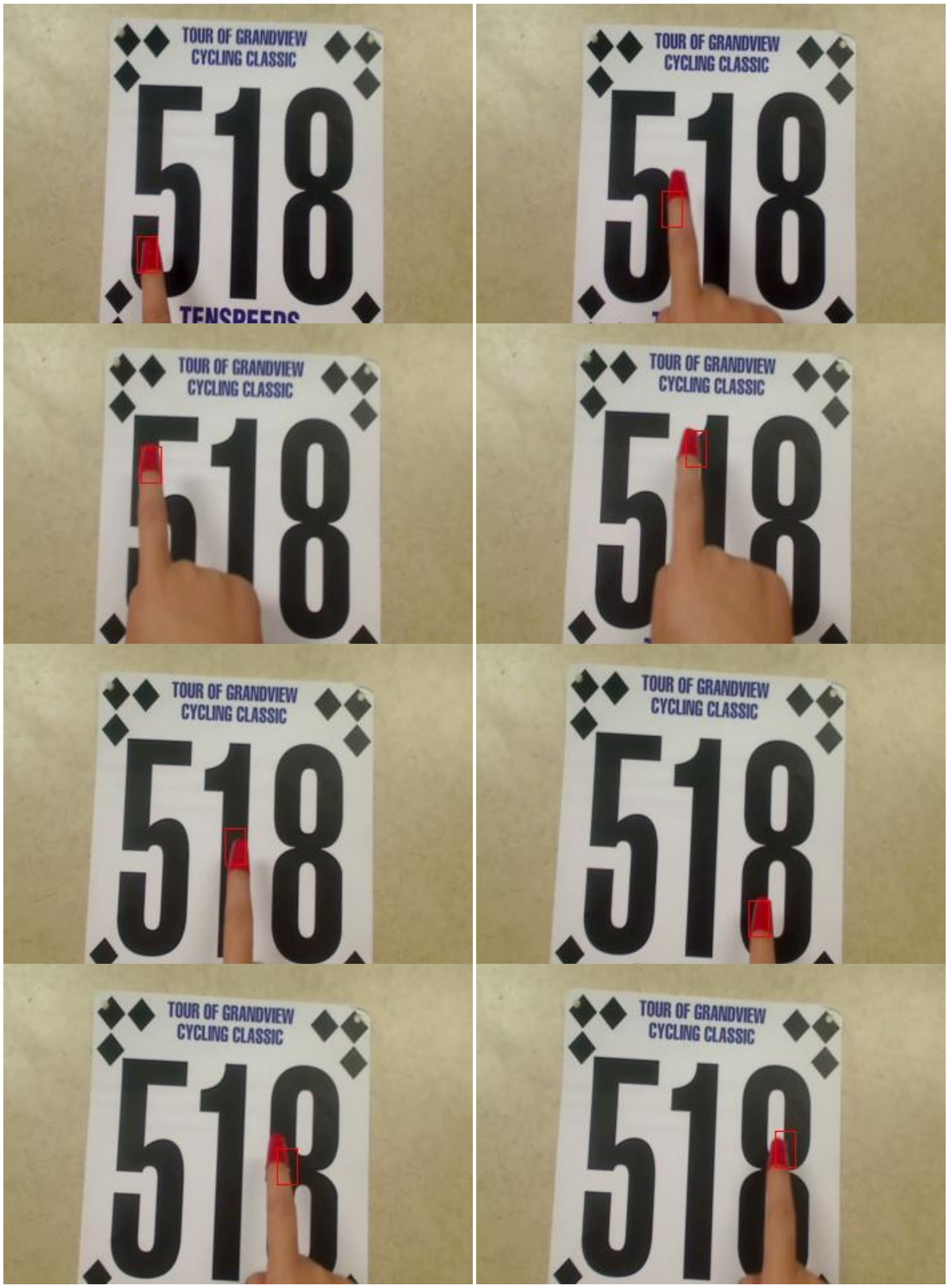
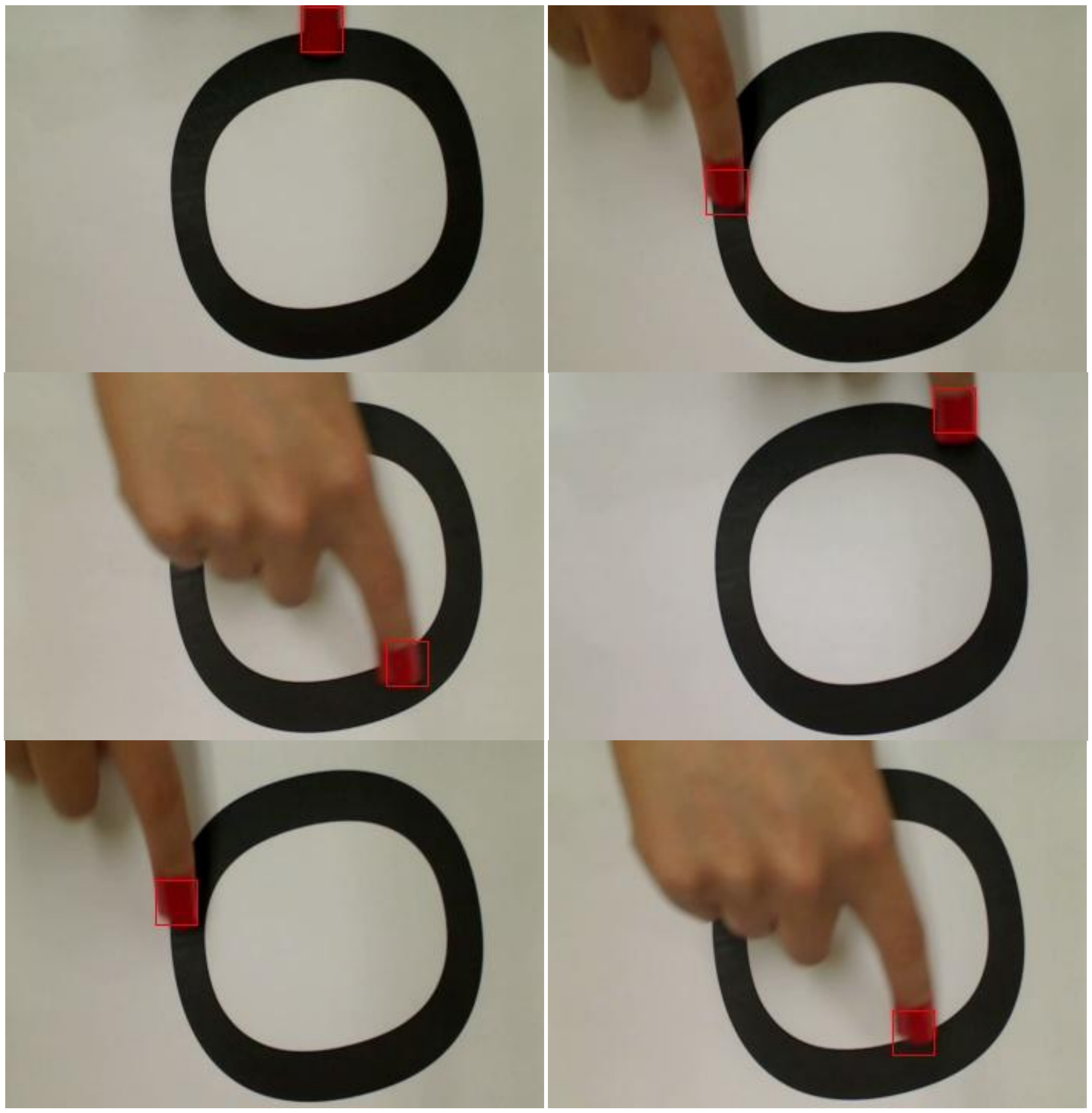


Figure 9: First-order with velocity model, 200 particles

We can tell that tracking with the first-order motion model obtains its robustness at the cost of large number of particles, which means a large computational cost. Our high-order with velocity motion model can save much computational time and get accurate tracking result at the same time.

In order to further prove that our method make a better performance when the fingertip moves fast, we also test a video sequence in which the fingertip moves faster than the previous one, and the fingertip changes its moving direction irregularly from clockwise to anticlockwise and vice visa. The results are shown in Figure 10.

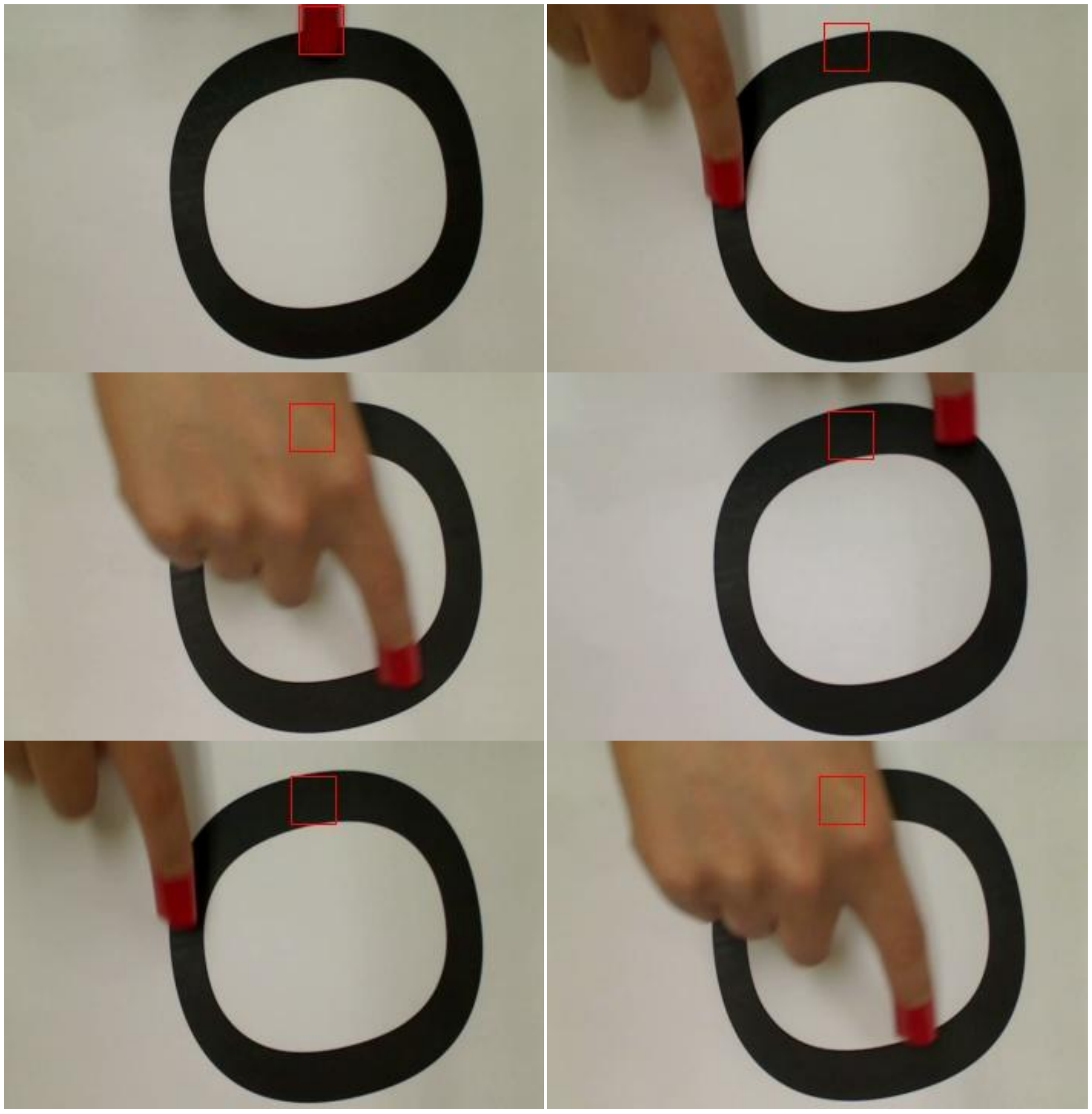
We can easily tell from Figure 10 that our method can accurately track the location of fingertip when it moves very fast and changes the moving direction. The first-order model performs very bad even with more particle numbers, and loses the fingertip once it starts moving fast and cannot track it back any more.



(a) Our high-order with velocity model, 50 particles

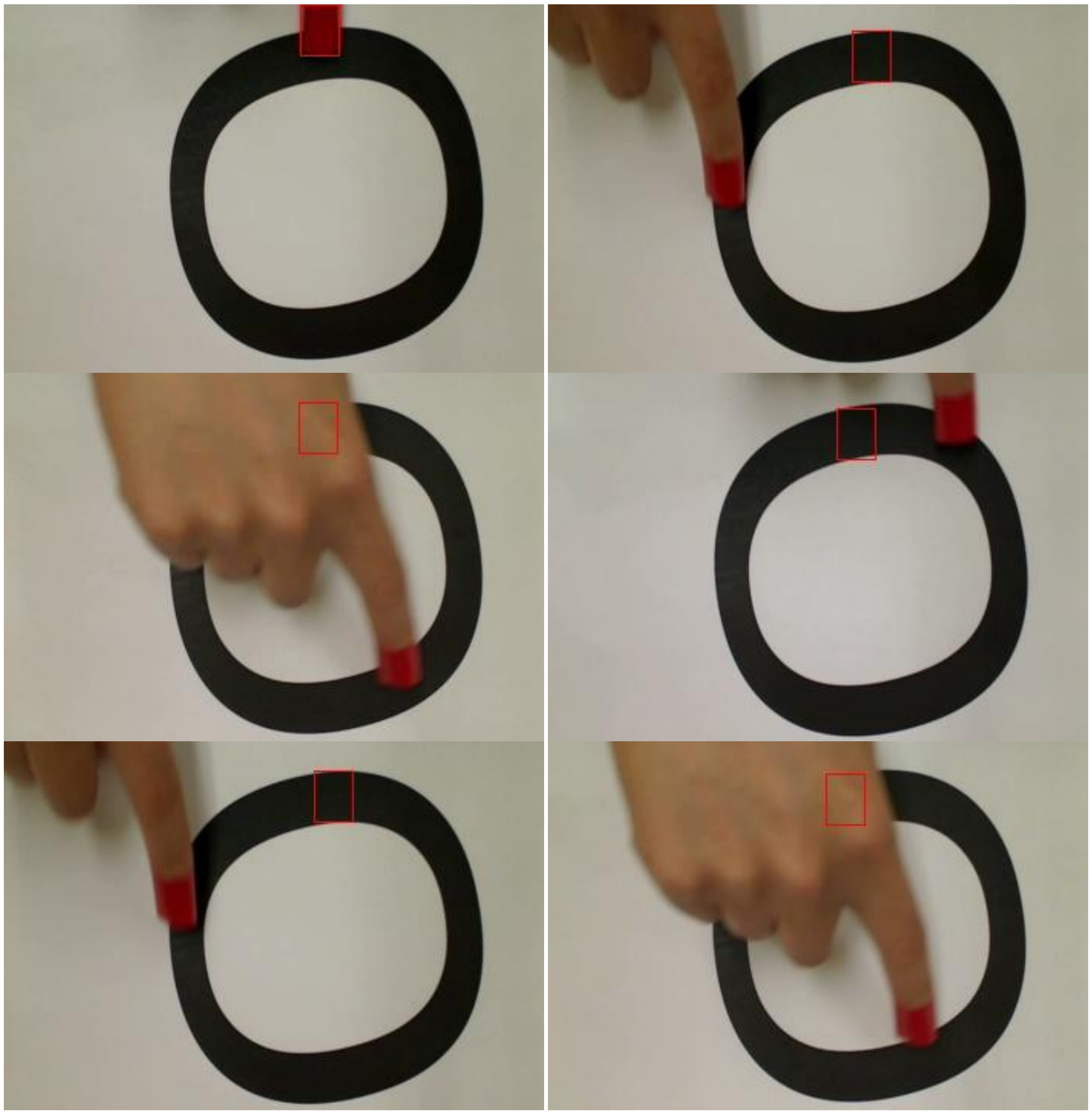
Figure 10: Comparison of the tracking performance using three different PF algorithms. Frames 5, 100, 200, 300, 400and 500 are shown. The trajectory of the fingertip is a circle, and the finger moves faster than normal speed. (Continued)

Figure 10: Continued



(b) First-order with velocity model, 200 particles (Continued)

Figure 10: Continued



(c)General linear model, 200 particles

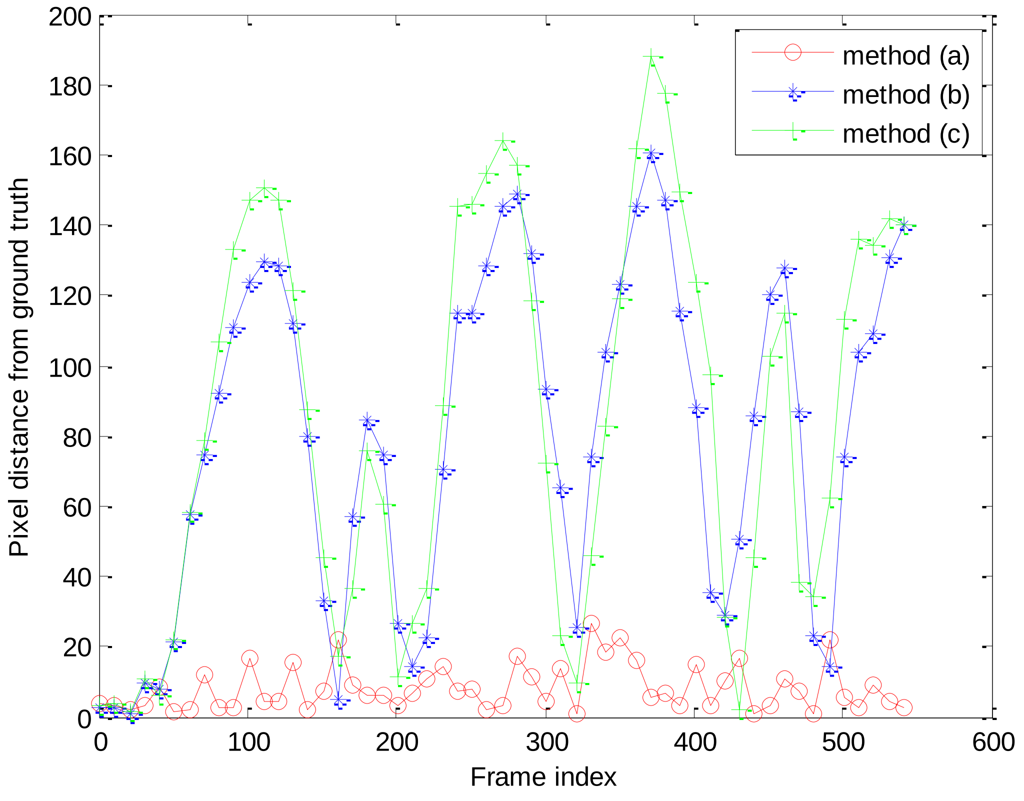


Figure 11: Quantitative comparison of three different PF algorithms of the second video sequence

The tracking error is calculated the same way as in the first video sequence. This time the fingertip moves very fast and changes its moving direction irregularly. We can tell our method performs much better with only 50 particles, compared with method (b) and (c) which both used 200 particles. Method (b) and (c) even totally loses the track once the finger starts to move along the circle trajectory. We reduce the particle numbers so that the computation cost will be degraded greatly.

4.2.2 Comparison between Our Method and MS

Fundamentally speaking, Mean Shift (MS) algorithm is used to calculate the density gradient estimation instead of the density estimation. It is a deterministic method which finds the best correspondence between the target candidate and the target model. Here the assumption that the object won't move too fast in the consecutive frames of the image sequences needs to be made, so that the target candidates will fall at least partly inside the model region of the previous frame.

The localization procedure starts from the position of the fingertip in the previous frame (the model) and searches in the neighborhood. Suppose the search for the new fingertip location starts at the location $\hat{y}_0$ of the fingertip in the previous frame. Then the new location $\hat{y}_1$ can be obtained according to the relation

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i g(\|\hat{y}_0 - x_i\|^2)}{\sum_{i=1}^{n_h} w_i g(\|\hat{y}_0 - x_i\|^2)} \tag{4.1}$$

where $g(x) = -k'(x)$ is the derivative of the kernel function which represents the color histogram distribution of the fingertip target and candidates, and

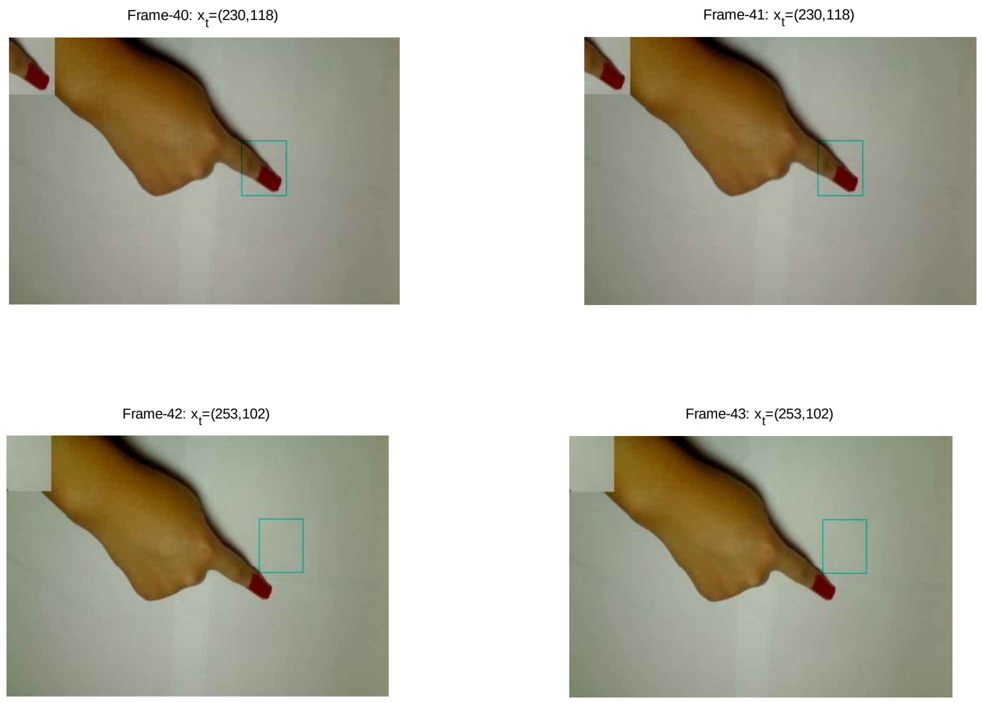
$$w_i = \sum_{l=1}^{m} \sqrt{\frac{\hat{q}_l}{\hat{p}_l(\hat{y}_0)}}\ \delta(b(x_i) - l) \tag{4.2}$$

is the weight assigned to each fingertip candidates.

Note that the kernel we use is the Epanechnikov profile [53], whose derivative is constant, so (4.1) reduces to

$$\hat{y}_1 = \frac{\sum_{i=1}^{n_h} x_i w_i}{\sum_{i=1}^{n_h} w_i} \tag{4.3}$$

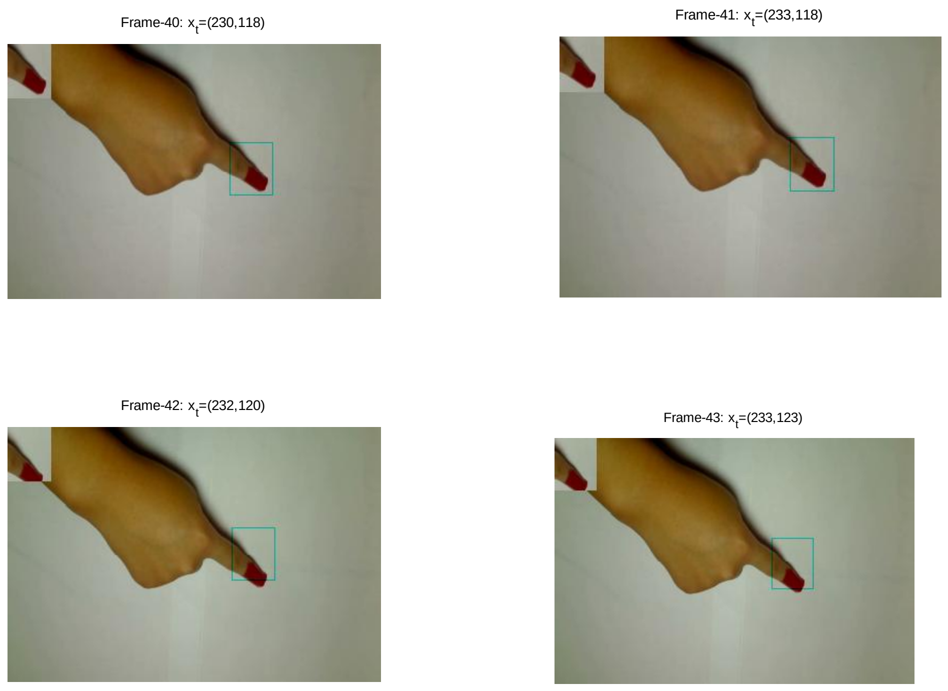
We can tell from the above equation that the basic idea of MS algorithm is to shift the target location towards the direction which leads to a local maximization in density distribution. A comparison between MS and our method is shown as Figure 12.


Frame-40: x_t=(230,118)
Frame-41: x_t=(230,118)
Frame-42: x_t=(253,102)
Frame-43: x_t=(253,102)

(a) Tracking results of the MS tracker

Figure 12: Comparison between MS and our method. Frames 40, 41, 42 and 43 are shown. (Continued)

Figure 12: Continued


Frame-40: x_t=(230,118)
Frame-41: x_t=(233,118)
Frame-42: x_t=(232,120)
Frame-43: x_t=(233,123)

(b) Tracking results of our method

In our test sequence, the finger has continuous movement, and at first it is moving very slowly, while during the time between frame 41 and 42, it suddenly moves faster, and as it can be seen from Figure 12(a) that the MS tracker loses the finger due to the fast movement. As we can tell see that in frame 42, it converges to a local maximum that corresponds to the white background, and it cannot find the red tip back after the loss. In contrast, our PF method can robustly track the tip after the fast movement, as shown in Figure 12(b). A quantitative comparison of the tracking error of these two methods is shown in Figure 13. It's obviously shown in this figure that MS loses the fingertip after the 36th frame and cannot recover it.

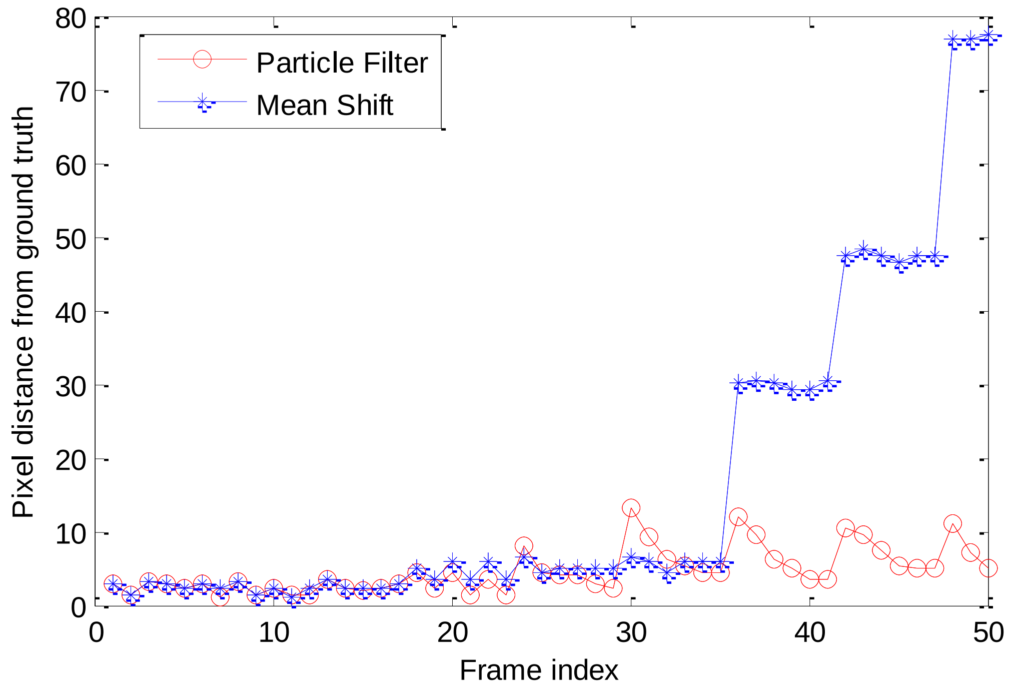


Figure 13: Quantitative comparison between MS and our method.

In order to further prove the robustness of PF, we test the whole video sequence, and some results are shown in Figure 14.

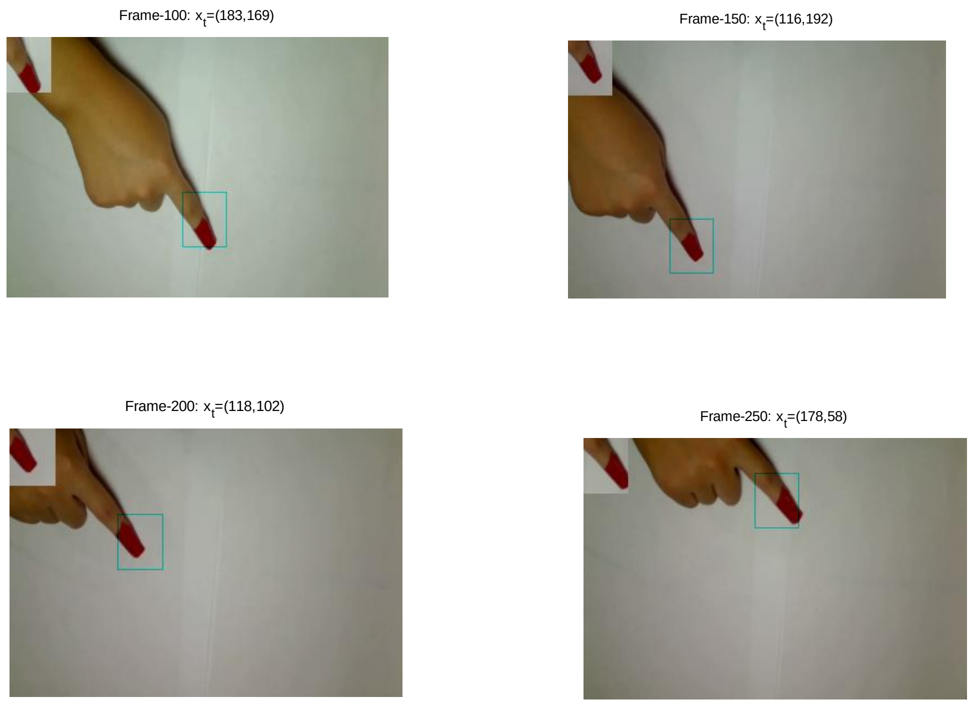

Frame-100: x_t=(183,169)
Frame-150: x_t=(116,192)
Frame-200: x_t=(118,102)
Frame-250: x_t=(178,58)

Figure 14: The PF can track the fingertip well in the whole sequence. The frames 100, 150, 200 and 250 are shown.

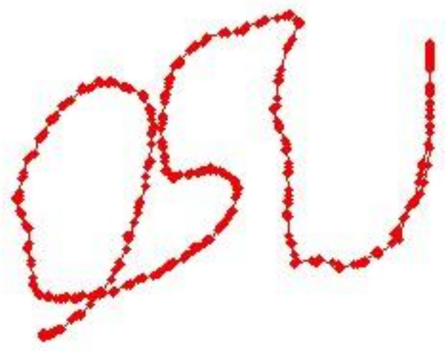
Considering the principles of the two methods, PF is a robust algorithm when dealing with fast movement and clutter environment, while MS is susceptible to converge to local maximum. In real applications of finger movement, it’s often the case when the finger suddenly moves faster, so we adopted the PF algorithm instead of MS in our research studies.

4.2.3 Control Mouse Cursors

Even if the fingertip position can be detected correctly, we might not get a natural motion of the mouse cursor due to the following reasons. Firstly, as a result of the limitation on tracking rate on commonly available PC hardware, we can merely get discontinuous coordinated of fingertip locations. [7] Accordingly, if such coordinates are directly used for the display of a mouse cursor on the monitor without any smoothing, the mouse cursor will jump around in a very unnatural fashion. Secondly, noise from the sources makes it difficult to position the mouse cursor accurately. [4] Thirdly, if the location of the fingertip is converted into the monitor coordinate directly, the difference of resolution between the input image from the camera and the monitor makes it difficult to position the mouse cursor accurately. [7]

To overcome these difficulties, we put up with a simple method to average the displacement of the detected fingertip position over a few frames and use this displacement as the mouse cursor displacement on the monitor screen. If the value of the displacement is less than the threshold value, the mouse cursor is not moved. Also we defined a valid region of the fingertip positions, only when the fingertip appears in this specific region will the cursor be moved. We used OpenCV library for our application, and $windows.h$ header file includes windows' $sendInput$ routine, which can generate mouse button events and make the mouse cursor move.

Figure 15 shows the raw tracking results as well as the smoothing result of the mouse cursor path when the tracking rate is 29Hz. We can tell from Figure 15(a) that the monitor coordinates of the mouse cursor are quite discontinuous and not smooth with the raw tracking results. The path of the mouse cursor is smoothed according to the method we proposed, and the smoothed result is shown in Figure 15 (b).


(a) The raw tracking result

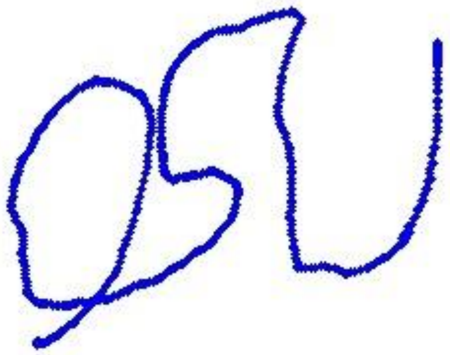
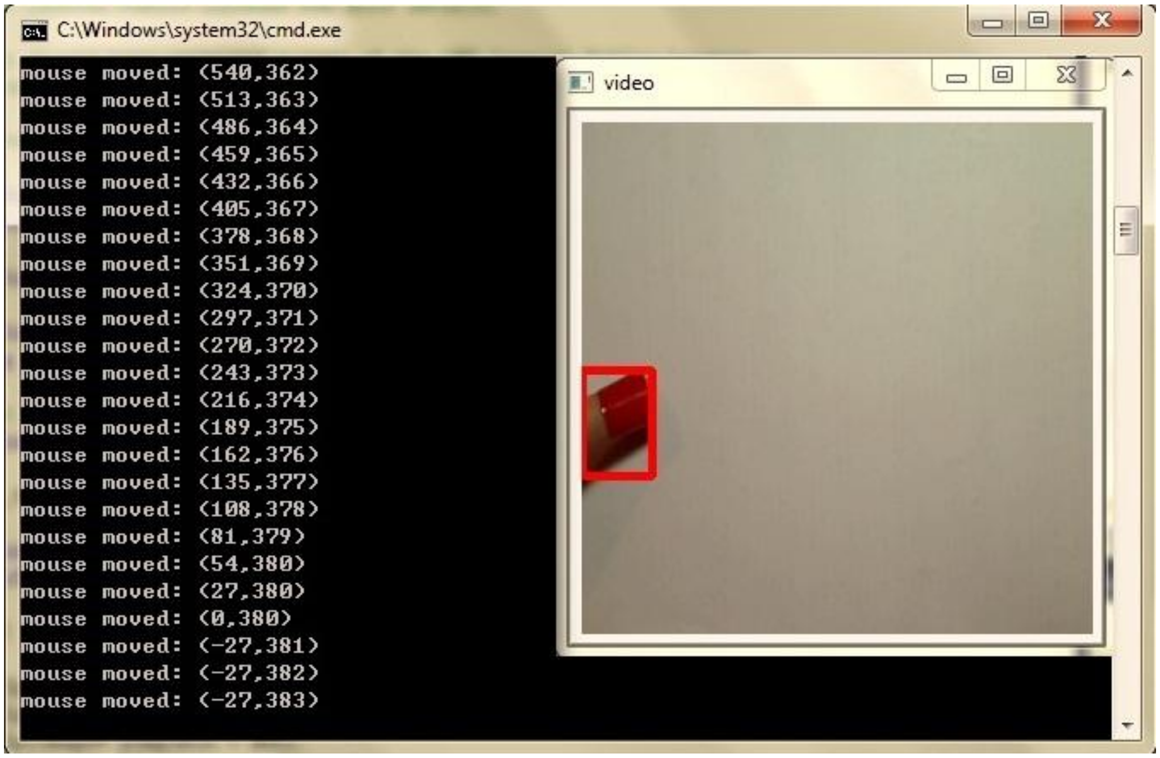

(b) The smoothed path

Figure 15: An example trajectory ("OSU") of the mouse cursor

Figure 16 shows the desktop display when the cursor is moved according to the fingertip movement.



```
mouse moved: (540,362)
mouse moved: (513,363)
mouse moved: (486,364)
mouse moved: (459,365)
mouse moved: (432,366)
mouse moved: (405,367)
mouse moved: (378,368)
mouse moved: (351,369)
mouse moved: (324,370)
mouse moved: (297,371)
mouse moved: (270,372)
mouse moved: (243,373)
mouse moved: (216,374)
mouse moved: (189,375)
mouse moved: (162,376)
mouse moved: (135,377)
mouse moved: (108,378)
mouse moved: (81,379)
mouse moved: (54,380)
mouse moved: (27,380)
mouse moved: (0,380)
mouse moved: (-27,381)
mouse moved: (-27,382)
mouse moved: (-27,383)
```

Figure 16: Desktop display of the cursor movement

Chapter 5: Contributions and Future Work

In this chapter, we first give a summary of our work and then provide a few potential directions on future improvements of the Finger Mouse application.

5.1 Contributions

The main contribution we made in this thesis is on the human fingertip tracking using a newly modified PF algorithm.

As shown is Chapter 3, we proposed a modified-model particle filtering (PF) method which is particularly tuned to track the fingertip according to pre-learning of the natural motion of human finger movement.

PF is a widely-used technique for implementing recursive Bayesian filtering by Monte Carlo sampling [10], and the basic idea of PF algorithm is to represent the posterior density of the fingertip state by a set of random particles with associated weights, and compute fingertip state estimation based on the samples and weights. In order to realize the purpose, we need to define the priori motion model for the fingertip state first, which is very important and directly affect the accuracy and efficiency of the tracking results. If the priori model fits the real fingertip movement closely, the particles can distribute according to a small covariance, and we can get the weighted mean value which is close to the real fingertip state, so that the particle numbers can be reduced accordingly; otherwise, if the priori model is not chosen properly, the particles will have a large covariance, increasing the tracking errors inevitably, and in order to prevent tracking performance degrading, the only way is to employ a large number of particles, which will cause large computational effort. Through the learning of natural motion of fingertip movement, we can get more accurate priori information on how the fingertip will move, so that we performed a second-order autoregressive priori model by combining temporal velocity information, which improved the tracking performance greatly as well as saving the computational time.

5.2 Future Works

Our system was evaluated on the basis of fingertip tracking and put effort on placing mouse to a specific location on the monitor screen. Though we got some exciting results and made improvement on the motion model in the tracking algorithm, there are still some directions we can work on.

We performed the tracking of fingertip according to the color histogram distribution, so that we required that it works in an environment free from background noise. In other words, if the background contains colors similar to the fingertip mark, when the finger moves abruptly or disappears from the camera field of view and comes back again, the tracker may lose the position of the finger.

Secondly, there is some delay to control the cursor movement when we did online implementation, and the cursor path is not smooth enough.

Thirdly, we only detected one single finger and performed cursor movement, and this can be extended to tracking multiple fingers in order to realize other mouse events, like clicking and scrolling.

Bibliography

[1] J. Rehg and T. Kanade. Visual analysis of high dof articulated object with application to hand tracking. *CMU Tech. Report CMU-CS-95-138*, Carnegie Mellon University, April, 1993.

[2] R. Kjeldsen and J. Kender. Towards the use of gesture in traditional user interfaces. *International Conference on Automatic Face and Gesture Recognition*, pages 151–156, 1996.

[3] Gary R. Bradski. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal Q2*, 1998.

[4] R. Kjeldsen and J. Kender . Interaction with on-screen objects using visual gesture recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 788–793, 1997.

[5] C. Jennings. Robust finger tracking with multiple cameras. *International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, pages 152–160, 1999.

[6] Andrew Wu, Mubarak Shah and N. da Vitoria Lobo. A virtual 3d blackboard: 3d finger tracking using a single camera. In *Proc. of the Fourth International Conference on Automatic Face and Gesture Recognition*, pages 536–543, 2000.

[7] Jiyoung Park and Juneho Yi. Efficient fingertip tracking and mouse cursor control for a human mouse. *Computer Vision Systems, Third International Conference, ICVS* 2003, Graz, Austria, April 1-3, 2003.

[8] Gary R. Bradski. Computer vision face tracking for use in perceptual user interface. *Intel Technology Journal* Q2, 1998.

[9] Aiper Yilmaz, Omar Javed, and Mubarak Shah. Object Tracking: A Survey.

[10] B. Ristic, S. Arulampalam, and N. Gordon. Beyond the Kalman Filter: Particle Filters For Tracking Applications. Artech House, 2004.

[11] Z. Zhang, Y. Wu, Y. Shan, and S. Shafer. Visual panel: Virtual mouse keyboard and 3d controller with an ordinary piece of paper. In *Proc. of Perceptual User Interfaces*, 2001.

[12] Y. Sato, Y. Kobayashi, and H. Koike. Fast tracking of hands and fingertips in infrared images for augmented desk interface. In *Proc. of IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, pages 462-467, 2000.

[13] Gordon, N. J., Salmond, D. J. and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *Proc. of IEEE on Radar and Signal Processing 140 (2):* 107–113. Retrieved 2009.

[14] A. Blake and M. Isard. The CONDENSATION algorithm - conditional density propagation and applications to visual tracking. In *Proc. NIPS*, pages 361-367, 1996.

[15] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. In *Proc. International Conference on Computer Vision*, pages 572–578, 1999.

[16] J. Carpenter, P. Clifford, and P. Fearnhead. Improved particle filter for nonlinear problems. In *Proc. Inst. Elect. Eng., Radar, Sonar, Navig.*, 1999.

[17] D. Crisan, P. Del Moral, and T. J. Lyons. Non-linear filtering using branching and interacting particle systems. *Markov Processes Related Fields*, volume 5, number 3, pages 293–319, 1999.

[18] P. Del Moral. Non-linear filtering: Interacting particle solution. *Markov Processes Related Fields*, volume 2, number 4, pages 555–580.

[19] K. Kanazawa, D. Koller, and S. J. Russell. Stochastic simulation algorithms for dynamic probabilistic networks. In *Proc. Eleventh Annu.Conf. Uncertainty AI*, pages 346–351, 1995.

[20] A. Doucet. On sequential Monte Carlo methods for Bayesian filtering. Dept. Eng., Univ. Cambridge, UK, Tech. Rep., 1998.

[21] N. Bergman. Recursive Bayesian estimation: navigation and tracking applications. Ph.D. dissertation, Linköping Univ., Linköping, Sweden, 1999.

[22] J. S. Liu and R. Chen. Sequential Monte Carlo methods for dynamical systems. *J. Amer. Statist. Assoc.*, volume 93, pages 1032–1044, 1998.

[23] S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, Feb. 2001.

[24] Rudolph van der Merwe, Nando de Freitas, Arnaud Doucet, and Eric Wan. The Unscented Particle Filter.

[25] Y. Wu and T. S. Huang. Vision-Based Gesture Recognition: A Review. *Lecture Notes in Computer Science*, volume 1739, pages 103-115, 1999.

[26] V. Pavlovic, R. Sharma, and T. S. Huang. Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review. *IEEE Trans. PAMI*, 19(7), pages 677-695, 1997.

[27] Gavrila, D. The Visual Analysis of Human Movement: A Survey. *Computer Vision and Image Understanding*, volume 73, number 1, pages82-98, January 1999.

[28] Y. Wu and T. S. Huang. Human hand modeling, analysis and animation in the context of HCI. *IEEE Intl Conf. Image Processing*, 1999.

[29] J. Lin. Visual hand tracking and gesture analysis. PhD thesis, Dept. of Electrical and Computer Eng., Univ. of Illinois at Urbana-Champaign, Urbana, 2004.

[30] A. Mulder. Design of three-dimensional virtual instruments with gestural constraints for musical applications. PhD thesis, Simon Fraser Univ., Canada, 1998.

[31] Robert Rosenberg and Mel Slater. The chording glove: a glove-based text input device. *IEEE Transactions on Systems, Man, And Cybernetics—Part C: Applications And Reviews*, volume 29, number 2, May 1999.

[32] J.J.Kuch and T.S.Huang. Vision-based hand modeling and gesture recognition for human computer interaction. *Master thesis, Univ.of Illinois at Urbana-Champaign*, 1994.

[33] Ying Wu, John Lin and Thomas S. Huang. Analyzing and capturing articulated hand motion in image sequences. I*EEE Trans. on Pattern Analysis and Machine Intelligence,* volume 27, number 12, pages 1910-1922, December 2005.

[34] E. Chao, K. An, W. Cooney, and R. Linscheid. Biomechanics of the Hand: A Basic Research Study. *Mayo Foundation, Minn.: World Scientific*, 1989.

[35] J. Lee and T. Kunii. Model-based analysis of hand posture. *IEEE Computer Graphics and Applications*, volume 15, pages 77-86, September 1995.

[36] Y. Wu and T.S. Huang. Capturing articulated human hand motion: a divide-and-conquer approach. In *Proc. IEEE International Conference on Computer Vision*, pages 606-611, September 1999.

[37] N. Shimada, Y. Shirai, Y. Kuno, and J. Miura. Hand gesture estimation and model refinement using monocular camera-ambiguity limitation by inequality constraints. In *Proc. Third Conference on Face and Gesture Recognition*, pages 268-273, 1998.

[38] R. Rosales and S. Sclaroff. Inferring body pose without tracking body parts. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 721-727, 2000.

[39] V. Athitsos and S. Sclaroff. Estimating 3D hand pose from a cluttered image. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 432-439, June 2003.

[40] C. Tomasi, S. Petrov, and A. Sastry. 3D tracking = classification + interpolation. In *Proc. IEEE International Conference on Computer Vision*, volume 2, pages 1441-1448, October 2003.

[41] M. Brand. Shadow puppetry. In *Proc. IEEE International Conference on Computer Vision*, volume 2, pages 1237-1244, 1999.

[42] C. Bregler and S. Omohundro. Nonlinear image interpolation using manifold learning. *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, eds., Cambridge, Mass.: MIT Press, 1995.

[43] P. Dario. Human-robot interface for welfare and services. Workshop on Human-Centered Robotics, *1997 IEEE International Conference on Robotics and Automation*, Albuquerque, NM, April 20-25, 1997.

[44] A. Agah and K. Tanie. Human interaction with a service robot: mobile-manipulator handling over an object to a human. In *Proc. 1997 IEEE International Conference on Robotics and Automation*, Albuquerque, NM, pages 575-580, April 20-25, 1997.

[45] S. Mascaro, I. Spano, and H. Asada. A reconfigurable homonymic unidirectional mobile bed with unified seating (RHOMBUS) for bedridden patients. In *Proc. 1997 IEEE International Conference on Robotics and Automation*, Albuquerque, NM, pages 1277-1282, April 20-25, 1997.

[46] P. Fiorini, K. Ali and H. Seraji. Health care robotics: A progress report. In *Proc. 1997 IEEE International Conference on Robotics and Automation*, Albuquerque, NM, pages 1271-1276, April 20-25, 1997.

[47] Li Xu, and Yuan F. Zheng. Real-time motion planning for personal robots using primitive motions. In *Proc. of the 2000 IEEE International Conference on Robotics & Automation*, San Francisco, CA, April 2000.

[48] T. Flash and N. Hogan. The coordination of arm movements: An experimentally confirmed mathematical model. *Journal of Neuroscience*, volume 5, number 7, pages 1688-1703, July 1985.

[49] J. Martin and J. Durand. Automatic handwriting gestures recognition using hidden markov models. In *Proc. 4th IEEE International Conference Automatic Face and Gesture Recognition (FG 2000),* IEEE Press, Piscataway, N.J., pages 403-409, 2000.

[50] Liang Yuan, Yuan F.Zheng, Junda Zhu, Lina Wang, and Anthony Brown. Object tracking with particle filtering in fluorescence microscopy images: Application to the motion of neurofilaments in axons. Submitted to *IEEE Transactions on Medical Imaging*, 2011.

[51] C. Shan, Y. Wei, T. Tan, and F. Ojardias. Real time hand tracking by combining particle filtering and mean shift. In *Proc. IEEE International Conference Automatic Face and Gesture Recognition FGR*, pages 669-674, 2004.

[52] P. Perez, C. Hue, and J. Vermaak. Color-based probabilistic tracking. In *Proc. European Conference Computer Vision*, pages 661–675, 2002.

[53] D. Comaniciu and P. Meer. Kernel-based object tracking. *IEEE Trans. Pattern Anal. Machine Intel*, 25(5):564–577, May 2003.

[54] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. In *Proc. Conf. Comp. Vision Pattern Rec.*, pages II:142–149,Hilton Head, SC, June 2000.