Touchscreen User Motion Anticipation for Usability Improvement

Tetsuyou Watanabe and Sawako Furuya
School of Mechanical Engineering
Kanazawa University
Kanazawa, Japan
emails: {te-watanabe@ieee.org, Sawakofuruya14@gmail.com}

***Abstract*— This paper proposes a method for improving touchscreen usability by anticipating user motions. Reaction speed and simple structure are important for good touchscreen usability. With this in mind, we present a system that can estimate a desired position for user motion by anticipating a motion several time steps ahead, using only sensors attached to the touchscreen. User motion that changes suddenly may not have the Markov property. We present here a novel methodology based on an Auxiliary Particle Filter (APF) with parameter estimation to deal with this issue. User motion is anticipated by regarding the motion in terms of parameters. We demonstrate the validity of our approach through experiments.**

Keywords-Auxiliary particle filter; anticipation; user intention; touchscreen; table device; user interface

I. INTRODUCTION

Many machine products are currently available, most of which require human operation. A machine that requires human operation does not always run as intended. Touchscreen manipulation is a typical example. A user can control items displayed on a touchscreen using fingers in an intuitive manner. However, a system sometimes cannot react to rapid finger motion, resulting in the display of an unintended image. Anticipating user motion to produce an output corresponding to user intention can be expected to result in improved work efficiency and reduced human error.

The important issues in user interface design are product downsizing (a hardware issue) and reaction speed (a software issue). A large user interface system can have reduced mobility, difficult set up, and therefore limited use. A late system response might not always produce the output that a user wants. Developing a small system for a user interface with rapid reaction can be expected to enhance convenience and usage in various fields, such as medicine. With this in mind, this paper targets the development of a convenient user interface system for touchscreens. By predicting intended user motion, namely, the desired position for the motion, we speed up the reaction speed of a system and enhance its usability. We use only regular sensors originally attached to a touchscreen to construct a compact system.

There are some research on predicting user intention. Chen et al. [1] proposed a method for predicting user intention/action in web applications using a user intention model based on extracted linguistic features. Armentano et al. [2] developed a method in which interface agents can detect user intention based on a variable-order Hidden Markov Model (HMM). Carrasco et al. [3] presented a method for predicting user hand movement using information from a camera attached to the wrist. Surf features extracted from the image were used to predict hand motion using an HMM. However, their target was *sequential* user intention or action expressed using discrete state variables, and the presented methods are difficult to apply to a continuous process such as user touchscreen motion. However, the predicted motion is a manually discrete event. There are several methods using a Particle Filter (PF) [4] for tracking humans and objects. Zahidul et al. [5] developed a PF-based method for tracking objects in a video scene using color and shape information. Wang et al. [6] presented a methodology for dealing with occlusion by combining the information from multiple cameras. The purpose of such research is not the anticipation of user motion (estimation of user intention), but its detection. Anticipation is required in situations in which a user motion suddenly changes, and its tracking is temporarily lost. A method that can deal with sudden changes in user motion is proposed by Oka et al. [7]. They developed a system that estimates the three-dimensional posture of a user's head in real time using PF from multiple camera images. However, this method did not provide an adequate solution. Enlarging the search area for a state at the next time step results in sudden changes in user motion from a discrete state change to a non-discrete (continuous) state change. This method thus requires a large number of particles and cannot be expected to work on touchscreens in real time.

On the other hand, research dealing with touchscreens has targeted mainly the evaluation of systems to improve designs. Wobbrock et al. [8] analyzed gestures used with touchscreens. Lin [9] evaluated the usability of mobile maps running on touchscreens. There is some research on developing systems for improving touchscreen usability. Dohse et al. [10] presented a method that combines hand tracking and touch detection to detect the movement of each user when multiple users access one touch display. However, a camera was used as an additional sensor for touch detection.

With this in mind, this paper presents a methodology for estimating user intention (desired position for user motion) for touchscreens. The key features are as follows. (1) Only regular sensors originally attached to a touchscreen are used. (2) We improve the reaction speed of a system by estimating user intention continually, using an Auxiliary Particle Filter (APF) [11]. APF is a kind of PF. PF is a Bayes filter [12] that approximates posterior density using a number of particles. Each particle has a weight, and the particles are recursively updated/sampled according to their weights. When computing weights in APF, observation results are taken into account. Particles closer to the actual value than in PF can be generated in APF. APF has been used for such tasks as robot localization [13], anomaly detection in spacecraft [14], and human tracking [15]. APF was extended to a filter that can deal with parameter estimation by West [16] and Liu et al. [17]. User intention (user-desired position) is usually continuous, but sometimes discontinuous. For example, a user's finger moves from left to right (in this case the motion is continuous), but the direction of motion changes suddenly from right to upward (in such a moment the motion is discontinuous). Then, user intention may lack the Markov property. This indicates that we cannot handle user intention estimation similarly to state estimation. In this connection, this paper presents a novel methodology that regards user intention in terms of parameters, and applies APF with parameter estimation.

The remainder of this paper is organized as follows. In Section II, the target system is described using a basic strategy. APF is introduced in Section III. We present our APF-based methodology for estimating user intention in Section IV. The experimental results are shown in Section V, and we conclude in Section VI.

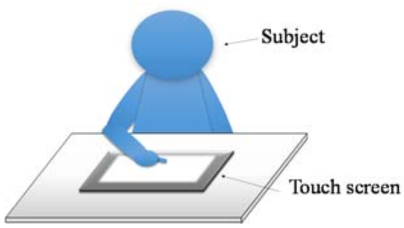

Figure 1. Schematic view of the target system

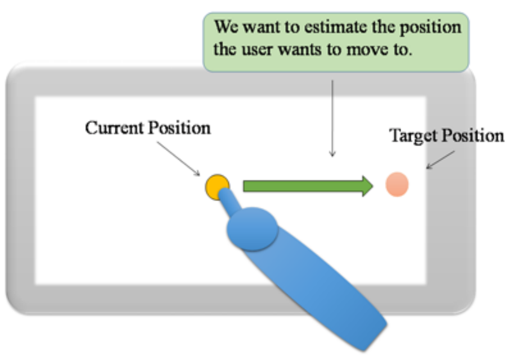

Figure 2. User operation on touch screen

II. TARGET SYSTEM AND BASIC STRATEGY

Fig. 1 shows a schematic view of the target system. We suppose that the user uses the touchscreen with one finger and moves object images on the touchscreen. Our purpose is to anticipate the position to which the user wants to move the target, namely, the user intention (Fig. 2).

The user finger position is measured by sensors attached to the touchscreen. This measurement is the observation value. We denote the observation value at time t by $\boldsymbol{y}_t \in R^2$. Let the user finger position be the state of the system: $\boldsymbol{x}_t \in R^2$. Ideally, $\boldsymbol{x}_t = \boldsymbol{y}_t$, but owing to (for example) noise and reaction delay, this does not always hold in practice. We regard user intention as the desired target position of user motion. As mentioned previously, the desired target position may lack the Markov property. We then regard the desired target position in terms of parameters: $\boldsymbol{\theta}_t \in R^2$. This is the key idea for estimating user intention.

In this case, state and observation equations can be represented by

$$p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, \boldsymbol{\theta}_t)\ , \tag{1}$$

$$p(\boldsymbol{y}_t|\boldsymbol{x}_t)\ , \tag{2}$$

with the aim of estimating user intention $\boldsymbol{\theta}_t$ while estimating the state $\boldsymbol{x}_t$:

$$p(\boldsymbol{x}_t, \boldsymbol{\theta}_t|\boldsymbol{y}_t)\ . \tag{3}$$

We use APF with parameter estimation [16][17] for this for this purpose. The original method estimates time-invariant parameters, but the target parameter $\boldsymbol{\theta}_t$ is not always time-invariant. Therefore, we extend the original APF with parameter estimation for estimating time-variant parameters.

III. AUXILIARY PARTICLE FILTER WITH PARAMETER ESTIMATION

We briefly introduce APF with parameter estimation as presented by West [16] and Liu et al. [17]. APF can be regarded as an extended version of PF based on the Sampling Importance Resampling (SIR) algorithm [18][19][20]. APF is explained after a brief introduction to PF.

A. Particle Filter based on SIR

PF is a Bayesian filter for estimating the state $\boldsymbol{x}_t$ at time t from observation $\boldsymbol{y}_t$. It corresponds to deriving the posterior distribution $(\boldsymbol{x}_t|\boldsymbol{y}_t)$. From Bayes' theorem, we have

$$p(\boldsymbol{x}_{t+1}|\boldsymbol{y}_{t+1}) \propto p(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1})p(\boldsymbol{x}_{t+1})\ . \tag{4}$$

Here, the prior distribution can be expressed by

$$p(\boldsymbol{x}_{t+1}) = \int p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t)p(\boldsymbol{x}_t|\boldsymbol{y}_t)d\boldsymbol{x}_t\ . \tag{5}$$

PF is for approximating the posterior distribution $p(\boldsymbol{x}_t|\boldsymbol{y}_t)$ at each time step t by $i = 1, \ldots, N$ particles $\boldsymbol{x}_t^{(i)}$ with corresponding weights $\omega_t^{(i)}$. Then, (5) is rewritten as

$$p(\boldsymbol{x}_{t+1}) = \textstyle\sum_{j=1}^{N} \omega_t^{(j)} p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t^{(j)})\ , \tag{6}$$

after which (4) is rewritten as

$$p(\boldsymbol{x}_{t+1}|\boldsymbol{y}_{t+1}) \propto p(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1}) \textstyle\sum_{j=1}^{N} \omega_t^{(j)}\, p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t^{(j)})\ . \tag{7}$$

We sample from $p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t^{(j)})$ for the j^{th} particle and evaluate the corresponding value of $\omega_t^{(j)}\, p(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1})$. Let this value be $\omega_{t+1}^{(j)}$ (at the next time step).

Copyright (c) IARIA, 2014. ISBN: 978-1-61208-325-4

$$\omega_{t+1}^{(j)} \propto \omega_t^{(j)} p\left(\boldsymbol{y}_{t+1} \middle| \boldsymbol{x}_{t+1}^{(j)}\right) = \omega_t^{(j)} \frac{p(y_{t+1}|x_{t+1}^{(j)})p(x_{t+1}^{(j)}|x_t^{(j)})}{q(x_{t+1}^{(j)}|x_t^{(j)},y_t)} . \quad (8)$$

The second term is the general form from the viewpoint of importance sampling. In importance sampling, the state is expressed by the weighted sum of proposal distribution $q(\cdot)$. SIR-based PF can be regarded as the filter assuming that the proposal distribution $q(\cdot)$ is the prior distribution $p(\boldsymbol{x}_t)$.

In SIR-based PF, the sampled points depend on the current prior $\boldsymbol{x}_{t+1}$ and do not take $\boldsymbol{y}_{t+1}$. into account. Consequently, we need a large number of particles for accurate estimation. This is the issue in PF. APF is a filter that resolves this issue by taking the likelihood distribution $p(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1})$ into account.

B. Auxiliary Particle Filter

APF resolves the issue as follows.

$$p(\boldsymbol{x}_{t+1}|\boldsymbol{y}_{t+1}) \propto \sum_{j=1}^{N} \omega_t^{(j)} p(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1})\, p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t^{(j)}) . \quad (10)$$

The difference in the position of summation in (10) from its position in (7) indicates that in (10), $\boldsymbol{y}_{t+1}$, derived/estimated from the observation function, is counted when sampling particles in (10), but is not counted when sampling in (7). As a result, APF counts only meaningful particles, while PF has to count meaningless particles as well. This means that fewer particles are required in APF compared to PF. Improved accuracy is also expected in APF.

In practice, it is difficult to know $\boldsymbol{x}_{t+1}$ in $p(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1})$. directly. Then, for the j^{th} particle, $\boldsymbol{\mu}_{t+1}^{(j)}$ is considered instead of $\boldsymbol{x}_{t+1}$ in $(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1})$ **,** where $\boldsymbol{\mu}_{t+1}^{(j)}$ is a representative value, such as the mean of $p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t^{(j)})$. We evaluate the following weight for the j^{th} particle, instead of (8).

$$g_{t+1}^{(j)} \propto \omega_t^{(j)} p(\boldsymbol{y}_{t+1}|\boldsymbol{\mu}_{t+1}^{(j)}) , \quad (11)$$

where $\omega_t^{(j)}$ is updated at time $t + 1$ as follows.

$$\omega_{t+1}^{(j)} = \frac{p(y_{t+1}|x_{t+1}^{(j)})}{p(y_{t+1}|\mu_{t+1}^{(j)})} . \quad (12)$$

This indicates that the weight $\omega_t^{(j)}$ is updated such that for every particle, the posterior distribution given in (4) can be expected to be a maximum. Control of the diffusion of particles is also expected.

C. APF with Parameter Estimation

APF is extended by West et al. [16] to a filter that can estimate (time-invariant) parameters. Here the filter is introduced briefly.

Let $\boldsymbol{D}_t$ be all of the information that the system has at time t. We consider the case in which the parameters $\boldsymbol{\theta}_t$ are time-invariant: $\boldsymbol{\theta}_t = \boldsymbol{\theta}$. In this case, the posterior function is expressed by

$$p(\boldsymbol{x}_{t+1}, \boldsymbol{\theta}|\boldsymbol{D}_{t+1}) \;\propto\; p(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1}, \boldsymbol{\theta}) p(\boldsymbol{x}_{t+1}, \boldsymbol{\theta}|\boldsymbol{D}_t)$$

$$\propto p(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1}, \boldsymbol{\theta}) p(\boldsymbol{x}_{t+1}|\boldsymbol{\theta}, \boldsymbol{D}_t) p(\boldsymbol{\theta}|\boldsymbol{D}_t) . \quad (13)$$

$\boldsymbol{\theta}$ cannot be treated in the same way as in state $\boldsymbol{x}_t$ estimation. Thus, we add small random disturbances and evolve artificially

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \boldsymbol{\zeta}_{t+1} , \quad (14)$$

$$\boldsymbol{\zeta}_{t+1} \sim N(0, \boldsymbol{W}_{t+1}) , \quad (15)$$

where $N(0, \boldsymbol{W}_{t+1})$ is the normal distribution with mean 0 and variance matrix $\boldsymbol{W}_{t+1}$, and $\boldsymbol{\zeta}_{t+1}$ provides small random disturbances. Then, $\boldsymbol{\theta}$ becomes $\boldsymbol{\theta}_t$ and can be treated in a manner similar to that used for state estimation.

Here, supposing that $\boldsymbol{\theta}_t^{(j)}$ is the parameter for the j^{th} particle with corresponding weight $\omega_t^{(j)}$, the following equation is considered for the parameter estimations.

$$p(\boldsymbol{\theta}_t|D_t) \approx \sum_{j=1}^{N} \omega_t^{(j)} N(\boldsymbol{\theta}_t|\boldsymbol{m}_t^{(j)}, h^2\boldsymbol{V}_t) , \quad (16)$$

where $N(\boldsymbol{\theta}_t|\boldsymbol{m}_t^{(j)}, h^2\boldsymbol{V}_t)$ is a multivariate normal density function with mean $\boldsymbol{m}_t^{(j)}$ and variance matrix $h^2\boldsymbol{V}_t$. h (> 0) is the smoothing parameter. $\boldsymbol{V}_t$ is the weighted variance matrix for $\boldsymbol{\theta}_t$.

$$\boldsymbol{V_t} = \sum_{j=1}^{N} \omega_t^{(j)} (\boldsymbol{\theta}_t^{(j)} - \overline{\boldsymbol{\theta}}_t)(\boldsymbol{\theta}_t^{(j)} - \overline{\boldsymbol{\theta}}_t)^{T} , \quad (17)$$

where

$$\overline{\boldsymbol{\theta}}_t = \sum_{j=1}^{N} \omega_t^{(j)} \boldsymbol{\theta}_t^{(j)} \quad (18)$$

$\boldsymbol{m}_t^{(j)}$ is the mean for $\boldsymbol{\theta}_t$ for the j^{th} particle and then, normally, $\boldsymbol{m}_t^{(j)} = \boldsymbol{\theta}_t^{(j)}$. However, in this case, the variance matrix for $p(\boldsymbol{\theta}_t|D_t)$ given in (17) becomes $(1 + h^2)\boldsymbol{V}_t$, and over-dispersion occurs. In order to resolve this issue, West [16] introduced the following equation for $\boldsymbol{m}_t^{(j)}$**.**

$$\boldsymbol{m}_t^{(j)} = a\boldsymbol{\theta}_t^{(j)} + (1 - a)\overline{\boldsymbol{\theta}}_t , \quad (19)$$

where $a = \sqrt{1 - h^2}$. In this case, the variance matrix for $p(\boldsymbol{\theta}_t|D_t)$ becomes $\boldsymbol{V}_t$ **,** and the issue is resolved.

APF with parameter estimation can be summarized as follows ([16] and [17]). Table 1 shows the nomenclature.

1. For $j = 1, \ldots, N$, we calculate the likely value associated with $\boldsymbol{x}_t^{(j)}$, $\boldsymbol{\theta}_t^{(j)}$, given by $\boldsymbol{\mu}_{t+1}^{(j)}$, $\boldsymbol{m}_t^{(j)}$

$$\boldsymbol{\mu}_{t+1}^{(j)} = E(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t^{(j)}, \boldsymbol{\theta}_t^{(j)}) , \quad (20)$$

$$\boldsymbol{m}_t^{(j)} = a\boldsymbol{\theta}_t^{(j)} + (1 - a)\overline{\boldsymbol{\theta}}_t , \quad (19)$$

where $E(X|A)$ is the conditional expectation of random variable X under phenomenon A. Note that a in (19) determines how close $\boldsymbol{\theta}_t^{(j)}$ should be to the mean $\overline{\boldsymbol{\theta}}_t$. In practice, a is set to a value extremely close to one (0.973-0.994).

2. Resample N particles according to the following weight.

$$g_{t+1}^{(j)} \propto \omega_t^{(j)} p(\boldsymbol{y}_{t+1} | \boldsymbol{\mu}_{t+1}^{(j)}, \boldsymbol{m}_t^{(j)}) \ . \tag{21}$$

Note that the index k is used for the resampled N particles.

3. Sample the parameters according to

$$\boldsymbol{\theta}_{t+1}^{(k)} \sim N(\cdot | \ \boldsymbol{m}_t^{(k)}, h^2 \boldsymbol{V}_t) \ . \tag{22}$$

4. Sample a value of the current state $\boldsymbol{x}_{t+1}^{(k)}$ from the state equation.

$$\boldsymbol{x}_{t+1}^{(k)} \sim p(\cdot \ | \boldsymbol{x}_t^{(k)}, \boldsymbol{\theta}_{t+1}^{(k)}) \tag{23}$$

5. Evaluate the following weight.

$$\omega_{t+1}^{(k)} \propto \frac{p(\boldsymbol{y}_{t+1} | \ \boldsymbol{x}_{t+1}^{(k)}, \boldsymbol{\theta}_{t+1}^{(k)})}{p(\boldsymbol{y}_{t+1} | \ \boldsymbol{\mu}_{t+1}^{(k)}, \boldsymbol{m}_t^{(k)})} \tag{24}$$

6. Repeat steps one through five.

TABLE I. NOMENCLATURE

N	The number of particles
$\boldsymbol{x}_t, \boldsymbol{y}_t, \boldsymbol{\theta}_t$	The state, the observation and the parameter vectors at t
$\boldsymbol{x}_t^{(j)}$	$\boldsymbol{x}_t$ at the j[th] particle at t
$\boldsymbol{\theta}_t^{(j)}$	$\boldsymbol{\theta}_t$ at the j[th] particle at t
$\omega_t^{(j)}$	Weight of j[th] particle at t
$\overline{\boldsymbol{\theta}_t}$	$\sum_{j=1}^{N} \omega_t^{(j)} \boldsymbol{\theta}_t^{(j)}$
$\boldsymbol{V}_t$	$\sum_{j=1}^{N} \omega_t^{(j)} (\boldsymbol{\theta}_t^{(j)} - \overline{\boldsymbol{\theta}_t})(\boldsymbol{\theta}_t^{(j)} - \overline{\boldsymbol{\theta}_t})^T$
$\overline{\boldsymbol{\theta}_t}$	$\sum_{j=1}^{N} \omega_t^{(j)} \boldsymbol{\theta}_t^{(j)}$
$\boldsymbol{\mu}_{t+1}^{(j)}$	Likely value associated with the component $p(\boldsymbol{x}_{t+1} | \boldsymbol{x}_t^{(j)}, \boldsymbol{\theta}_t^{(j)})$
$\boldsymbol{m}_t^{(j)}$	Likely value associated with $\boldsymbol{\theta}_t^{(j)}$
$g_t^{(j)}$	Weight for resampling at the j[th] particle at t

IV. User Intention Estimation Using Parameterization

Here we present a new methodology for estimating user intention. User motion can change suddenly, in which case the Markov property cannot be assumed and the usual state estimation algorithm cannot be used. The key idea for resolving this issue is to estimate the desired position for user motion in terms of parameters. We use APF with parameter estimation for user intention. The original method is intended for time-invariant parameters, while the target parameters are not always time invariant. However, we can use the same analogy, with some extensions of the algorithm, because the method treats the time-invariant parameters as time-variable parameters.

The desired position for user motion $\boldsymbol{\theta}_t^u$ is considered to be sometimes the same as or close to the value at the previous time step and sometimes definitely different and difficult to anticipate from that value. Therefore, we update $\boldsymbol{\theta}_t^u$, supposing the following equation

$$\boldsymbol{\theta}_{t+1}^{u}{}^{(j)} = a^* \boldsymbol{\theta}_t^{u(j)} + (1 - a^*) \boldsymbol{\theta}_{t+1}^{u*} \ , \tag{25}$$

where $\boldsymbol{\theta}_t^{u(j)}$ is $\boldsymbol{\theta}_t^u$ for the j[th] particle. The first term relates to the case in which the desired position for user motion is the same as or close to the value at the previous time step. The second term relates to the case in which user motion changes suddenly, and the desired position for user motion is far from the value at the previous time step. a^* is the parameter that represents which term more strongly affects the value at the next step. $\boldsymbol{\theta}_{t+1}^{u*}$, which is not related to $\boldsymbol{\theta}_t^{u*}$**,** can be expressed by

$$\boldsymbol{\theta}_{t+1}^{u*} = \boldsymbol{\theta}_{t+1}^{u*}\left(\boldsymbol{D}_t^{\boldsymbol{y}}\right) , \tag{26}$$

where $\boldsymbol{D}_t^{\boldsymbol{y}} = \{\boldsymbol{D}_{t-1}^{\boldsymbol{y}}, \boldsymbol{y}_t\}$ is the information set at time t with regard to observation $\boldsymbol{y}_t$. Even in this case, (13) is still valid (with $\boldsymbol{\theta}$ replaced by $\boldsymbol{\theta}_{t+1}$). Therefore, comparing (25) with (19) (and (14)), it is expected that the same methodology can be applied by replacing $\boldsymbol{m}_t^{(j)}$ by $\boldsymbol{\theta}_{t+1}^{u}{}^{(j)}$ given in (25). The two cases differ in that $\boldsymbol{\theta}_t^{(j)}$ in (19) is the main part in (19) (a is large), while $\boldsymbol{\theta}_{t+1}^{u*}$ is the main part in (25) (a^* is small). The difference appears in the variance matrix. Over-dispersion of the variance matrix must then be checked carefully. In the estimation of the desired position for user motion, the variance matrix can be expressed as

$$(h^2 + {a^*}^2) \boldsymbol{V}_t^u + (1 - a^*)^2 \boldsymbol{V}_t^{u*} \ \ , \tag{27}$$

where

$$\boldsymbol{V}_t^u = \sum_{j=1}^{N} \omega_t^{(j)} \left(\boldsymbol{\theta}_t^{u(j)} - \overline{\boldsymbol{\theta}_t^u}\right)\left(\boldsymbol{\theta}_t^{u(j)} - \overline{\boldsymbol{\theta}_t^u}\right)^T , \tag{28}$$

$$\boldsymbol{V}_t^{u*} = \sum_{j=1}^{N} \omega_t^{(j)} \left(\boldsymbol{\theta}_t^{u*} - \overline{\boldsymbol{\theta}_t^u}\right)\left(\boldsymbol{\theta}_t^{u*} - \overline{\boldsymbol{\theta}_t^u}\right)^T . \tag{29}$$

When $\boldsymbol{\theta}_t^{u*} \cong \overline{\boldsymbol{\theta}_t^u}$, letting $h^2 + {a^*}^2 = 1$ or $h^2 + {a^*}^2 < 1$, the variance matrix becomes $(h^2 + {a^*}^2)\boldsymbol{V}_t^u$ (constant or decreasing), and the diffusion of dispersion can be avoided similarly to the original APF with parameter estimation. When $\left|\boldsymbol{\theta}_t^{u*} - \overline{\boldsymbol{\theta}_t^u}\right|$ is large, letting a^* be small such that the sampled particles can be around $\boldsymbol{\theta}_t^{u*}$, the variance matrix has small magnitudes (because the second term in (25) is the same for every particle), and diffusion can be avoided. In any case, with small a^*, diffusion can be avoided. In practice, if user motion changes suddenly, a small a^* is used. If user motion is not large, a^* satisfying $h^2 + {a^*}^2 = 1$ is used and the (constant) desired position for user motion can be obtained.

 ISBN: 978-1-61208-325-4

Summarizing, for estimating desired user motion position, we will use the algorithm given in Section III.C, replacing $\boldsymbol{m}_t^{(j)}$ by $\boldsymbol{\theta}_{t+1}^{u}{}^{(j)}$ given in (25) and controlling the magnitude of a^* such that the diffusion of the variance matrix given in (27) can be avoided.

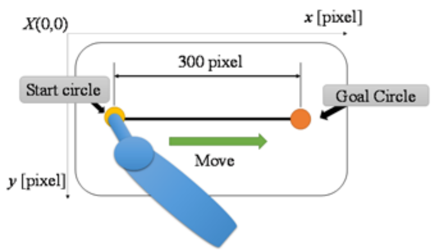

Figure 3. Overview on touchscreen for Experiment 1

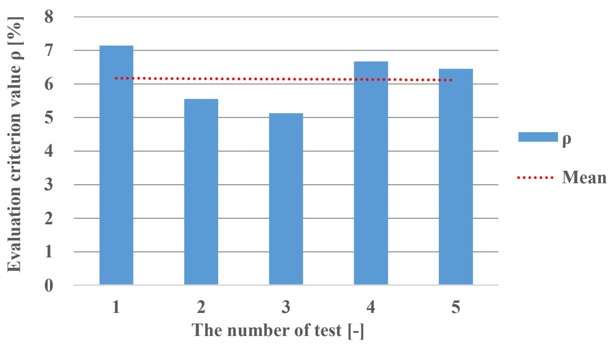

Figure 4. Evaluation criterion value ρ for every test in Experiment 1

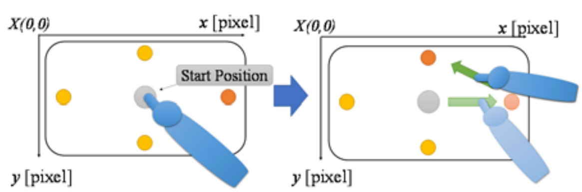

Figure 5. Schematic view of task in Experiment 2

V. EXPERIMENTS

The purpose of the presented algorithm is improvement of usability by predicting the desired position for user motion and displaying it. Three experiments were conducted to evaluate the validity of the algorithm.

A. Experiment 1

In the first experiment, we see whether the presented algorithm can actually anticipate user motion, predicting the desired position several time steps later.

Fig. 3 shows a schematic view of Experiment 1. The operator was asked first to put a fingertip on the start/yellow circle and move to the goal/orange circle without decreasing the speed (this means that the fingertip goes through the goal circle). The goal circle indicates the desired position of user motion in this case. If the anticipated state variable can reach the goal earlier than the actual fingertip gets there, then the presented algorithm can anticipate the user motion (intention). The time it takes to move from yellow/start to orange/goal circles was then investigated.

Sensors attached to the touchscreen are used for obtaining the position $\boldsymbol{y}_t$ of the user fingertip. Based on this observation value, the state and observation equations without noise terms are defined for user motion (intention) anticipation as follows:

$$\dot{\boldsymbol{x}}_t = \boldsymbol{K}_t(\boldsymbol{\theta}_t^u - \boldsymbol{x}_t) \tag{28}$$

$$\boldsymbol{y}_t = \boldsymbol{x}_t \tag{29}$$

where $\boldsymbol{K}_t$ is the gain and $\boldsymbol{\theta}_t^u$ is the desired position for user motion. Note that $\boldsymbol{\theta}_t^u$ corresponds to the estimated desired position for user motion, the user intention. As mentioned above, we investigated how far in advance $\boldsymbol{\theta}_t^u$ reaches the final goal compared to $\boldsymbol{y}_t$ (measured fingertip position). If $\boldsymbol{\theta}_t^u$ can arrive at the final goal earlier than $\boldsymbol{y}_t$, then the presented algorithm can display/predict user intention in advance of the user motion.

The ratio of the times for $\boldsymbol{\theta}_t^u$ and $\boldsymbol{y}_t$ to reach the goal gives us a quantified evaluation criterion and shows how stably the presented system can show the user intention in advance of the user motion. We define the criterion as follows:

$$\rho = \left(1 - \frac{t_r(\boldsymbol{\theta}_t^u)}{t_r(\boldsymbol{y}_t)}\right) \times 100 \ \ [\%], \tag{30}$$

$$t_r(\boldsymbol{p}) = t_e(\boldsymbol{p}) - t_s\ , \tag{31}$$

where

$t_e(\boldsymbol{p})$: The time/moment when $|\boldsymbol{p} - \boldsymbol{p}_g| < \varepsilon$ is satisfied, where $\boldsymbol{p}_g$ is the position of the goal circle and ε (= 10 pixels) is a small positive constant

t_s: The time when user motion starts

We used a Sony Duo 11 tablet PC with touchscreen (CPU: Intel(R) Core(TM) i7-3687U 2.10GHz, OS: Windows 8 Pro, Touchscreen size: 1920×1080 pixels, electrostatic capacity type touchscreen), while the screen size of the application was 900×600 pixels. The distance between the start and goal positions was set to 300 pixels. The sampling time was set to 10 msec. The number of particles was 100. The experiment was conducted five times.

Fig. 4 shows the results representing the criterion value at each test and the mean. In every case, a positive value can be obtained, and it can be seen that $\boldsymbol{\theta}_t^u$ can always precede $\boldsymbol{y}_t$; the factor was over 5% in every case, and the mean reached 6%. This shows that the system can display user intention (desired position) stably in advance of the user motion. Note that it does not matter whether the magnitude of 5~6% is large or not. What the magnitude is positive is important. These results show success in the anticipation of user motion.

B. Experiment 2

In the second experiment, we see whether the presented algorithm can actually deal with user motion that lacks the Markov property, a sudden change in user motion.

 ISBN: 978-1-61208-325-4

Fig. 5 shows a schematic view of Experiment 2. On the touchscreen, there are four circles, three yellow and one red. Initially, the user finger is positioned at the center of the screen. Then, the user was asked to move her/his fingertip to the red circle. If the user fingertip gets to the red circle, one of the yellow circles changes to red. Which circle goes to red was determined randomly. Then, we can observe the behavior of the system when user intention suddenly changes. The user tracked the red circle ten times. The other settings were the same as in Experiment 1.

Fig. 6 shows a portion of the results, because the rest of the results showed the same tendency, and space is limited. The blue and green lines show the observation values; the blue line shows the x coordinate of the user fingertip, y_x, while the green line shows the y coordinate, y_y. The red and yellow lines show the anticipated user position/intention; the red line shows the x coordinate of the position, θ^u_{xt}, while the yellow line shows the y coordinate, θ^u_{yt}.

It can be seen that the system can stably predict the desired position for user motion even when user motion changes suddenly. The successful anticipation of user motion can also be seen. The anticipated position θ^u_{xt}, θ^u_{yt} always precedes the measured position y_x, y_y, especially for high-speed motion.

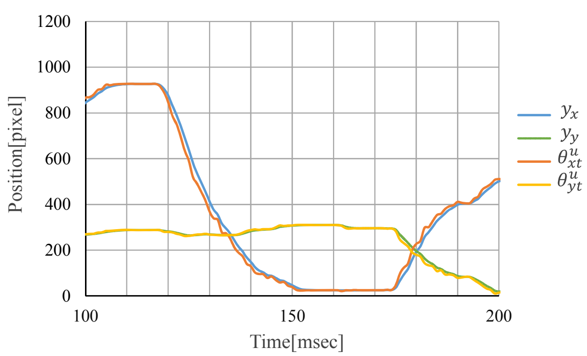

Figure 6. Time series data of measured and anticipated user motion $(\boldsymbol{y}, \boldsymbol{\theta}^{\mathrm{u}})$ in experiment 2

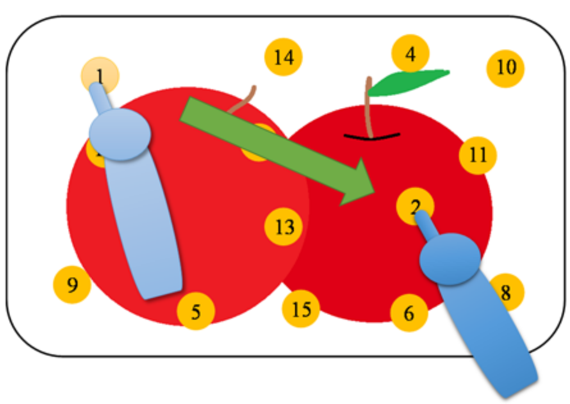

Figure 7. Schematic view of the game screen in experiment 3

C. Experiment 3

In the third experiment, we see whether anticipation by the presented algorithm can actually improve usability. For this purpose, a sensory examination method was conducted using a simple game application. Games were prepared with and without the presented algorithm, and we asked subjects to compare and evaluate the usability of the games.

Fig. 7 shows a schematic view of the game screen (1200 × 700 pixels) that the subject sees in the experiment. The numbered blue circles (from one to 15) are located randomly on the screen. The user is asked to wipe out all the circles with one fingertip touching the screen, maintaining touch throughout the game. If the fingertip is positioned around the circle, the circle disappears. However, the user has to wipe out the circles in ascending order. The following three kinds of systems were prepared for evaluation.

With both anticipation and sign: The line segment $\boldsymbol{S}_{\boldsymbol{y}-\boldsymbol{\theta}^{\mathrm{u}}}$ from $\boldsymbol{y}$ to $\boldsymbol{\theta}^{\mathrm{u}}$ was displayed as a sign, where $\boldsymbol{y}$ is the observed fingertip position and $\boldsymbol{\theta}^{\mathrm{u}}$ is the anticipated desired fingertip position determined by the presented algorithm. The condition of the erasing circle is $|\boldsymbol{S}_{\boldsymbol{y}-\boldsymbol{\theta}^{\mathrm{u}}} - \boldsymbol{p}_{ci}| < 40$ pixels, where $\boldsymbol{p}_{ci}$ is the ith circle position.

With only anticipation: The presented algorithm was used, but no sign was displayed to help users. The erasing condition is $|\boldsymbol{S}_{\boldsymbol{y}-\boldsymbol{\theta}^{\mathrm{u}}} - \boldsymbol{p}_{ci}| < 40$ pixels.

Without anticipation or sign: The presented algorithm was not used. No sign was displayed to help users. The erasing condition is $|\boldsymbol{y} - \boldsymbol{p}_{ci}| < 40$ pixels.

The sampling time was 100 msec. The other settings were the same as in Experiments 1 and 2. We asked 20 subjects (22-24 years) to conduct the games with the three kinds of systems (three games per person), and to select one game/system as having the best usability. Fig. 8 shows that the presented algorithm improved usability, because the number selecting the system with the presented algorithm is greater than the number selecting the system without the algorithm. Whether a sign should be displayed is considered to depend on individual preference, as was confirmed using a written questionnaire.

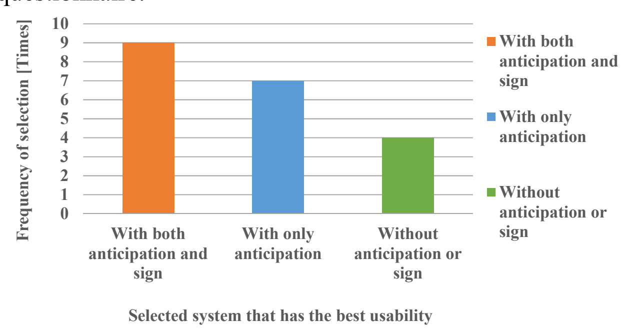

Figure 8. Frequency of selecting the game/system with the best usability

VI. CONCLUSION AND FUTURE WORK

This paper presented a methodology for anticipating user motion on a touchscreen, aimed at improving usability. In order to have simple structure, only regular sensors originally attached to a touchscreen are used. User motion on a touchscreen can change suddenly, with consequent inability of the system to react to user motion. To overcome this problem and improve usability, this paper presented a

 ISBN: 978-1-61208-325-4

methodology for anticipating user motion several time steps later, based on APF with parameter estimation. Three experiments were conducted to evaluate the effectiveness of the presented approach. In the first experiment, we showed that the desired position of user motion (user motion several time step later) can be anticipated in every case. In the second experiment, we showed that the system can predict user motion even when user motion suddenly changes. The third experiment indicated that the anticipation of user motion/intention by the proposed algorithm can improve usability. These experimental results showed the validity of our approach. It is considered that there are many other systems whose efficiency can be improved by incorporating the presented algorithm. This will be the direction of our future work.

ACKNOWLEDGMENT

This work was partly supported by JSPS KAKENHI Grant Number 24650092 and the Telecommunications Advancement Foundation.

REFERENCES

[1] Z. Chen, F. Lin, H. Liu, Y. Liu, W-Y. Ma, and L. Wenyin, “ User Intention Modeling in Web Applications Using Data Mining,” World Wide Web, 2002, pp. 181-191.

[2] M. G. Armentano and A. A. Amandi, “Recognition of User Intentions for Interface Agents with Variable Order Markov Models,” User Modeling, Adaptation, and Personalization, 2009, pp. 173-184.

[3] M. Carrasco and X. Clady, “Prediction of user’s intention based on the hand motion recognition,” Proc. of Int. Conf. of the Chilean Computer Society (SCCC), 2009.

[4] A. Doucet, N. D. Freitas, and N. Gordon, “Sequential Monte Carlo Methods in Practice,” Springer, 2001.

[5] M. Z. Islam, C. M. Oh, and C-W. Lee, “Video Based Moving Object Tracking by Particle Filter,” Int. J. of Signal Processing, Image Processing and Pattern, vol. 2, no. 1, March, 2009, pp. 119-132

[6] Y. Wang and J. Wu, “Adaptive Particle Filter for Data Fusion of Multiple Cameras,” J. of VLSI Signal Processing 49, 2007, pp. 363-76.

[7] K. Oka, Y. Sato, Y. Nakanishi, and H. Koike, “Head Pose Estimation System Based on Particle Filtering with Adaptive Diffusion Control,” MVA, 2005, pp. 586-589.

[8] J. O. Wobbrock, M. R. Morris, and A. D. Wilson, “User-Defined Gestures for Surface Computing,” Proc. of SIGCHI Conf. on Human Factors in Computer Systems, 2009, pp. 1083-1092.

[9] Y. Lin, “Evaluation of User Interface Satisfaction of Moblie Maps for Touch Screen Interfaces,” Proc. of Int. Conf. on Advances in Computer-Human Interaction , 2012, pp. 22-27.

[10] K. C. Dohse, T. Dohse, J. D. Still, and D. J. Parkhurst, “Enhancing Multi-user Interaction with Multi-touch Tabletop Displays using Hand Tracking,” Proc. of Int. Conf. on Advances in Computer-Human Interaction, 2008, pp. 297-302.

[11] M. Pitt and N. Shephard, “Filtering via simulation: Auxiliary Particle Filters,” Journal of the American Statistical Association. vol. 94, no. 466, Jun, 1999, pp. 590-599.

[12] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics. MIT Press, 2005.

[13] N. Vlassis, B. Terwiji, and B. Krose, “Auxiliary Particle Filter Robot Localization from High-Dimensional Sensor Observations,” Proc. of IEEE Int. Conf. on Robotics and Automation, 2002, pp. 7-12, vol. 1.

[14] K. Goto, Y. Kawahara, T. Yairi, and K. Machida, “Anomaly detection for spacecraft by estimating parameters with particle filter,” Trans. of the Japan Society for Aeronautical and Space Science vol. 55, no. 642, 2007, pp. 355-358.

[15] S. J. McKenna and H. Nait-Charif, “Tracking human motion using auxiliary particles filters and iterated likelihood weighting,” Image and Vision Computing , vol. 25, 2007, pp. 852-862.

[16] M. West, “Approximating Posterior Distributions by Mixtures,” Journal of Royal Statistical. Society(Ser. B), vol. 55, 1992, pp. 409-422.

[17] J. Liu and M. West, Combined parameter and state estimation in Simulation-based Filtering. Springer, 2001.

[18] N. J. Gordon, D. J. Salmon, and A. F. M.Smith, “Novel approach to nonlinear/non-Gaussian Bayesian state estimation ,” IEE Proc. of F (Radar and signal processing), vol. 140, no. 2, 1993, pp. 107-113.

[19] M. Isard and A. Blake, “Condensation – conditional density propagation for visual tracking,” Int. J. of Computer Vision, vol. 29, 1998, pp. 5-28.

[20] F. Dellaert, D. Fox, W. Burgard, and S. Thurm, “Monte carlo localization for mobile robots,” Proc. of IEEE Int. Conf. on Robotics and Automation, 1999, pp. 1322-1328, vol. 2.

