S. Boll, S. Maaß & R. Malaka (Hrsg.): Mensch & Computer 2013
München: Oldenbourg Verlag, 2013, S. 261– 270

Dynamic Gaussian Force Field Controlled Kalman Filtering For Pointing Interaction

Florian van de Camp[1], Rainer Stiefelhagen[2]

Fraunhofer IOSB, Karlsruhe[1]
Karlsruhe Institute of Technology, Karlsruhe[2]

Abstract

As human computer interaction is extending from the desk to the whole room, modalities allowing for distant interaction become more important. Distant interaction however, is inherently inaccurate. Assisting technologies, like force fields, sticky targets, and target expansion have been shown to improve pointing tasks. We present a new variant of force fields that are modeled using Gaussian distributions, which makes placement and configuration as well as overlap handling straight forward. In addition, the force fields are dynamically activated by predicting intended targets, to allow for natural and fluent movements. Results from a user study show, that the dynamic Gaussian fields can speed up the time needed to click a button with a pointing gesture by up to 60%.

1 Introduction

Over the last few years, many new input technologies have emerged, not only in research but also in commercial applications. One new input modality is pointing gesture control, which allows moving interaction away from the screen to the whole room. Technologies like the Kinect already made this kind of interaction popular but there is still much room for improvement. While the detection of pointing gestures will get more accurate with better sensors, there is a limit to the achievable accuracy due to the human physique. If simply adjusting user interfaces by, for example, increasing the size of interface elements is not a feasible option, we need to find other ways to improve the accuracy. Traditionally, input modalities are unidirectional, meaning they provide input data without knowledge about the system they are connected to. However, what is currently displayed directly influences the user in his or her actions and, therefore, strongly influences the input data. Taking context information into account is useful for improving the accuracy of input modalities [1] [2] [3] [4].

To evaluate the use of this information we focused on one of the most essential pointing tasks: clicking a button. The key contributions of this paper are:

- Gaussian modeling allows optimal force field strengths
- Automatic overlap handling allows straight forward placement of force fields
- A method for dynamic force field activation for fluid cursor motions.
- Easy use of force fields because of direct integration into existing filtering process.

The paper is organized as follows. We review related work on pointing interaction and utilizing context information in section 2. In section 3 we describe the proposed design and algorithms. Results from a user study are presented in section 4. We conclude with a summary and an outlook on future work.

2 Related Work

Even so the mouse is a rather accurate input device, it is the subject of most research on assisting technologies for pointing interaction. Cockburn et al. [2] have shown that especially for more challenging user interfaces with smaller targets, feedback and enhancements can improve interaction. Especially for the sticky icons technique [5], which actively influences the cursors positioning, they report good results. Kabbash et al. [6] and Grossman et al. [7] present similar techniques that rely on increasing the area on which the cursor acts dynamically, depending on the surrounding interface elements. Instead of increasing the size of a single target, the systems by Jansen et al. [8] and Findlater et al. [9] magnify a certain area around the cursor to assist in the selection of targets. While the effect might be similar, the magnification has the advantage of not having to know the exact placement of all targets. Instead of increasing the area of influence of the cursor Brock et al. [10] take an opposing approach where the targets are increased in size instead. While this leads to similar effects, they also show the increasing size of the targets as visual feedback. Their results show however that while the performance is increased, this can also be a source of irritation for the users. Ahlström et al. [1] present force fields which actively manipulate the cursor position to move towards the center in a certain area with a fixed strength. Unlike the sticky icons technique the size of force fields can be larger than the target sizes and unlike the expanding targets the size is fixed and the area in which manipulation occurs not visible. Guiard et al. [11] demonstrate an extreme variant of force fields called "Object Pointing". In this system the cursor jumps from button to button whichever is closest. While this results in very fast and easy selection of the targets it would be hard to combine this with an application that requires both, target selection and free placement or movement of objects. Few publications utilize assisting technologies for other input devices than the mouse, but in all cases improvements by using the context information have been reported. One is a technique by Yin et al. [4] that is tailored towards digital pen input and uses visible beams from nearby buttons to guide the user in selecting them by simply lifting the pen. This technology takes both the button locations as well as the pen position into account. Specht at al. [3] adapt the force fields to deal with overlaps by using the direction in which a target was entered, to decide which force field should be used. The system is one of the few that do not use a mouse. The joystick used, however, is similar to a mouse as it is an indirect pointing device. It is im-

portant to note that knowledge of the location of interface elements can be a drawback, however, there are techniques to extract the locations from any application [12].

3 Design and Implementation

It has been shown that force fields are a good way of modeling context information for clicking tasks [1]. The idea is to move the cursor towards the button's center with a certain force by adding the difference in both x and y directions to the current cursor position. Since this would make the cursor jump very quickly, it is only moved a partial amount of the total offset. That amount is the strength of the force field. However, it is hard to find good trade-offs for both the strength and the size of force fields. While large force fields make it very easy to click a button, it also severely restricts the interface design as traditional force fields can not overlap, so each field has to be defined individually depending on the current display layout. Section 3.1 describes how force Fields can be modeled by Gaussian distributions to overcome this problem. Section 3.2 describes how context knowledge can be integrated into a Kalman filter, to create force fields. Section 3.3 describes how the intend to click a button can be predicted to dynamically activate individual force fields to allow for smooth movement.

3.1 Gaussian Force Fields

As a user moves the cursor closer to a target, the more likely it becomes that he or she intends to click on the target. So instead of defining a certain distance from the center that acts as a border where force is abruptly applied with a fixed strength, it is more natural to gradually increase the force as the cursor gets closer to the button's center. To achieve this, we model the force field strength as a two dimensional Gaussian distribution around the button's center. With a direct pointing technique, strong force fields do not cause as many problems as it is the case for indirect pointing techniques but strength still has great impact. If a cursor is on a button and the user intends to click it, a maximum strength should be used so clicking is accurate. If further away, however, such strength can lead to jumpy behavior and erroneous clicks. Modeling the strength of the force field as a Gaussian distribution solves these problems. The offset x_w^n, y_w^n applied by the n-th force field to the current cursor position (c_x, c_y) is calculated using Equation 1 with f being the normalized value at the current cursor position of the bivariate normal distribution around the button's center (b_x, b_y) with standard deviation(s_x, s_y). Normalization means that the maximum strength is 1.0, which would force a maximum correction.

$$x_w^n = (c_x - b_x) f(c_x, c_y, s_x, s_y) \qquad \textit{Equation 1}$$

$$y_w^n = (c_y - b_y) f(c_x, c_y, s_x, s_y) \qquad \textit{Equation 2}$$

An ideal force field heavily depends on the user interface and its targets sizes and arrangements. Large force fields make it easy to click the corresponding target but also influence the cursor movement and can cause clicks when they are not intended. Without explicit handling

of overlaps the placement of force fields has to be done manually for a specific layout for best results. The proposed Gaussian modeling of force fields is used to merge overlapping force fields by balancing multiple influences. Since the forces of each force field are directional corrections with a strength based on the distance to the button's center, adding these directions will eliminate forces in opposing directions or reduce the force in one direction if other buttons are nearby. The final offset (x_w, y_w) for a cursor within the range of N force fields is then calculated as shown in Equations 3 and 4.

$$x_w = \sum_{n=0}^{N} x_w^n \qquad y_w = \sum_{n=0}^{N} y_w^n \qquad M = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad s_t = M\, s_{t-1} + w$$

Equation 3 *Equation 4* *Equation 5* *Equation 6*

3.2 Kalman Filter Integration

Since the pointing data is noisy, it has to be filtered to provide a smooth user experience. We use a Kalman filter, which models all known aspects of the system that transform the actual user movements into the observations that can be made. Since the user interacts with the user interface, its contents and layout are aspects of the complete system. However this kind of context information is not typically used for Kalman filters. We will first describe the set up of the Kalman filter for the pointing gesture recognition system in general and then describe how context information can be included using the control matrix. The measurements are the x and y positions on the screen produced by the pointing gesture. The system state is modeled using the position (x, y) and the velocity (v_x, v_y). Using estimates for the measurement error w the current system state s_t can be derived from the previous state s_{t-1} according to the Kalman filter as shown in *Equation 6* with M as the state transition model. To account for both, the previous position and the effect of the velocity on the position M is set up as shown in *Equation 5*. We use the often neglected control input of the Kalman filter to integrate the context information into the system as it allows to integrate additional forces. We calculate the cursor offsets for the current position as described above (Equation 3 and 4) from all force fields that affect the cursor. Since the targets are static buttons we assume that the user will try to stop the cursor when getting closer to the button so the velocity is eliminated by making the opposing control equal to the current velocity. Given the total cursor offset (x_w, y_w), the control input vector b_t to calculate state s_t is set up as in *Equation 7*. As the control vector consist of the same parameters as the system state, the control matrix C is identical to the transition model M. This inclusion of the context information via the control matrix C and the control vector b_t results in a Kalman filtering step (*Equation 8*). If the cursor is not within the range of any force field, the control matrix is set to zero and has no influence on the filtering process.

$$b_t = \begin{pmatrix} x_w^t \\ y_w^t \\ -v_x^{t-1} \\ -v_y^{t-1} \end{pmatrix}$$

Equation 7

$$s_t = M\, s_{t-1} + C\, b_t + w$$

Equation 8

3.3 Dynamic Force Fields

While force fields as well as other assisting technologies like manipulating the control-display ratio or sticky targets can be helpful, they are only helpful when needed. When they are not needed, however, they can interfere with the interaction and introduce sources of irritation or even errors. A best case scenario would be that the system knows when which button is targeted and activate only the target buttons force field. To gain this knowledge we developed an algorithm that analyzes the users' behavior to predict which force fields should be activated. After observing the cursor trails and accompanying parameters of the Kalman filter for click interactions, we observed the expected pattern of human motion [13], which always includes an acceleration as well as a deceleration phase. In all cases there was a deceleration phase before the click, caused by the user trying to place the cursor right on the button. This phase can be more or less distinct but a complete stop from full speed is debarred. Using this knowledge, the deceleration phase D can be detected:

$$D = \begin{cases} 1, & (v_x^t - v_x^{t-1} < 0) \vee \left(v_y^t - v_y^{t-1} < 0\right) \\ 0, & else \end{cases}$$

Continually observing the movements, only when a deceleration phase is detected force fields are activated. They stay activated until a click occurred or an acceleration is detected.

4 Evaluation

To evaluate the use of force fields for a direct interaction technology in general, as well as the effects of the dynamic extensions described in the previous section we conducted a user study. For comparison, three techniques were implemented for the participants to try. The first technique, "PLAIN" uses no context knowledge and is a direct mapping of the detected pointing direction with Kalman filtering not using any control input. "SFF" are static force fields that are always active and apply a fixed strength which is the same at any position in the force field as described in [1]. Finally, the third technique is "DGFF", force fields that are dynamically turned on or off depending on the systems prediction described in section 3.3 and model the strength of the force fields using a Gaussian distribution.

4.1 Apparatus and Participants

The experiment was conducted using a 4m x 1.5m (4096px × 1536px) back projection video wall in conjunction with a computer vision based pointing system. The pointing gesture system [14], creates a 3D reconstruction of a person using overlapping views of multiple

calibrated cameras. A skeleton model is fitted into the reconstruction and the pointing direction of an extended arm is derived. Using the calibration data of the cameras and with knowledge of the location of the display, a 2D intersection point can be found and converted to pixel coordinates. Since this two dimensional position on the display is what the user perceives, the goal is to get those movements as smooth as possible. Therefore Kalman filtering on the 2D data is used to compensate for measurement errors and sensor noise. With this system, users can control a cursor on the wall by simply pointing to the desired position. The position is updated with 30Hz and a click is triggered using a dwell timer. A dwell timer triggers a click if the cursor is kept stationary for a specified duration. For this experiment keeping the cursor stationary meant less than 5 pixels offset to the previous position for 15 consecutive position updates (=0.5sec) to trigger a click. Users got visual feedback about the click progress as shown in Figure 1.

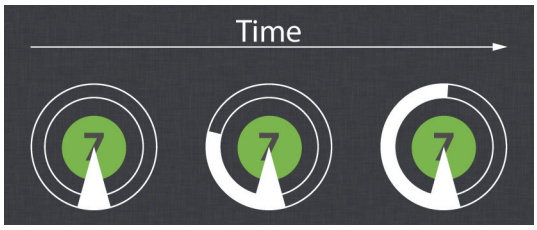

Figure 1: Users get visual feedback about the progress of the dwell timer for clicking

Figure 2: User interacting with pointing gesture.

Eleven (ten male) users aged between 21 and 32 participated. Nine of those were right handed and all used their primary arm for the entire experiment. Seven of the users had prior experience with pointing gesture interaction. All eleven participants had normal or corrected to normal sight. The average height of participants was 180.3cm, while buttons were placed at heights from 99cm to a maximum of 196cm from the ground.

4.2 Experimental Setup

The participants had to perform a multi directional pointing task following part 9 of the ISO 9241 standard [15] (See also Figure 2). Two variants for the size of the circle (a 1000px diameter and a 400px diameter) and three different button sizes were used (12px (~11.7mm), 25px (~24.4mm), and 50px (~48.8mm) diameter). In addition to these ISO standard suggestions, we made one modification. One of the major downsides of force fields is the fact that they can interfere with the interaction when they are not needed, but the suggested layout does not take this problem into account. For this reason, we placed additional buttons randomly on the screen that were not intended as targets. In Figure 2, the 15 numbered, green buttons are the targets, the 24 plain, red buttons are the additional buttons. The red buttons were randomly placed once for both circle diameters and all button sizes, and then kept the same for each user. There was no difference in how the buttons behaved but users were clearly instructed to only click on the green buttons in the labeled order. To eliminate any possible confusion about the next target, in addition to the numbering of the buttons a white indicator circle was placed around the current target (around button 6 in Figure 2). The combination of circle sizes and button sizes resulted in 6 combinations users had to perform for each technique. Users were able to try and experiment with each technique for as long as they wished

before they were presented with the 6 tasks for the current technique. The order of techniques was randomly chosen for each participant. Because pointing gesture interaction in front of a large video wall can be tiring, users were free to take breaks. With a small mark (in 74cm distance from the video wall) on the floor for users to stand on, we minimized any influence the distance might have on the accuracy of the pointing gesture recognition. Users were encouraged to select the targets as fast as possible but asked to balance speed and accuracy. For static force fields a strength had to be defined while the dynamic Gaussian force fields vary their strength depending on their size and the distance to the buttons center automatically. As suggested by Ahlström et al. in [1], a strength of 0.8 for static force fields was used. Because the pointing gesture is a direct pointing technique opposed to the indirect pointing of a mouse, we did not have to use an "escape function" despite this rather high strength. The sizes of the static force fields depended on the available space which was limited mostly because of the added non target buttons and were never larger than 3 times the size of the button with no overlaps. Since the dynamic Gaussian force fields handle overlapping automatically, a standard deviation of 30 pixels was used for all buttons.

4.3 Measurements

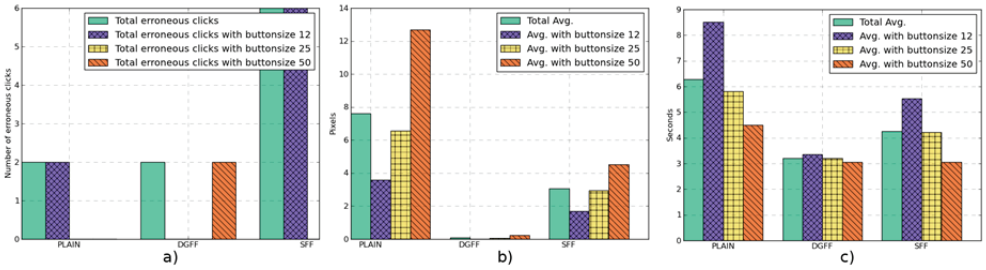

Figure 3:Results for erroneous clicks(a) ,click offset from center(b) and speed(c)

While the users performed the tasks, all cursor positions and clicks were recorded. From this data we extracted the number of erroneous clicks, the time it took to perform clicks, and the accuracy with which buttons were hit. One of the concerns was that force fields of any kind might lead to accidental clicks. However, only a negligible number of such failed clicks occurred with any technique (Figure 3a). The second property considered was the offset of the buttons center to the final cursor position when the click occurred (Figure 3b). The technique without any cursor warping (PLAIN) has the largest offset. Considering that both other techniques pull the cursor towards the center this is right in line with our expectations. Here, the difference in the modeling of force fields becomes apparent. While the offset for the static force fields is significantly smaller than that of the PLAIN technique, the offset for the dynamic Gaussian force fields (DGFF) is close to zero. The ability to use up to full force at the very center but lower strengths further away allows to pull the cursor to the very center without getting unusably strong force fields. To compare the time it took to click a button with each technique, the time from leaving the previous target to the successful click on the current target was measured. To be able to compare these durations despite the different circle layouts, we normalized the values to a distance of 1000 pixels (Figure 3c). The high values especially for the smaller buttons for the PLAIN technique show that this button size would not be usable in a real application. The much smaller values for the force field tech-

niques (SFF are 32%, DGFF even 48% faster) show that these assisting technologies are enabling the use of such small interface elements. While there are improvements for all button sizes using the dynamic Gaussian force fields over the static force fields, they are especially apparent for the smaller buttons. For the larger buttons (25px and 50px) there are still significant improvements when utilizing force fields but these bigger buttons are at least somewhat usable without assisting technologies as well. While the large 50px buttons are hard to miss, in case of the smaller buttons the overlap handling of the Gaussian force fields and the larger force field size this permits, leads to significant improvements over the PLAIN and also the SFF technique. The table below shows the durations in seconds (standard deviation given in braces) for each technique by button size as well as the average over all button sizes respectively. The t-test p-value for DGFF over PLAIN and SFF is given in the fifth and sixth column.

Button size	*PLAIN*	*SFF*	*DGFF*	**PLAIN*	**SFF*
Avg.	6.27 (2.50)	4.26 (0.83)	3.20 (1.42)	**0.0020**	**0.0451**
12px	8.50 (2.67)	5.52 (0.76)	3.35 (1.17)	**0.0001**	**0.0001**
25px	5.81 (1.47)	4.21 (0.96)	3.20 (1.03)	**0.0001**	**0.0274**
50px	4.50 (1.03)	3.05 (0.72)	3.04 (0.79)	**0.0013**	0.9756

4.4 Heatmaps

In addition to the results discussed above, we created heat maps from the cursor positions over time to analyze the movement properties of the different techniques. To avoid clutter only the data from the experiments with the 1000px circle layout are shown in Figure 4. While the heat map for the technique PLAIN (Figure 4a) shows similarities to that generated by the DGFF (Figure 4c) technique, the heat map for the SFF (Figure 4b) technique shows that the static attraction force of the buttons create hotspots at those locations, which influence the cursor even though buttons were never targeted. The result is a “jumpy” cursor that several participants criticized. The automatic activation of force fields for the dynamic Gaussian case creates a heat map that reflects the smooth movement over non-target buttons, resembling the heat map in Figure 3a, where no cursor manipulation is used at all. The lack of cursor manipulation results in much more scattered cursor positions around the targets.

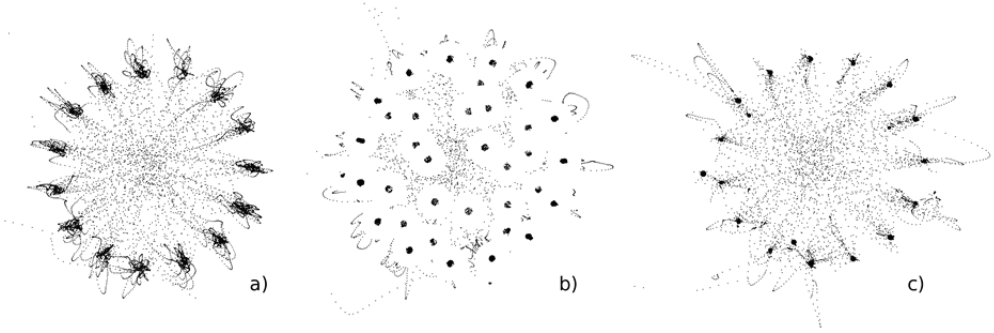

Figure 4: Cursor positions over time for the three techniques PLAIN (a), SFF(b) and DGFF(c)

5 Conclusion

In this paper we proposed a new kind of force fields, which significantly improve the accuracy of pointing gesture interaction without interfering with its natural and fluid motion characteristic even in complex interface scenarios. We demonstrated how context knowledge can be integrated into a Kalman filter system, which makes it convenient to utilize this information for inaccurate modalities which usually incorporate filtering. Extensions of static force fields were presented to deal with the real world problems of force field placement and configuration as well as to nullify the negative effect static force fields have whenever force enhanced elements are just passed instead of targeted. The resulting dynamic Gaussian force fields were evaluated in a user study along with static force fields and a baseline. The experimental setup was designed to take into account the real world situation of nearby buttons as well as buttons that were not used as targets to study the effects on the interaction. The results show, that despite the generic configuration, the dynamic Gaussian force fields improved the interaction time by 24.88% in average compared to the static force fields and 48.96% in comparison to the PLAIN technique, which does not utilize context knowledge. The effects of the proposed technologies broaden the usability of pointing interaction for real world applications. The automatic handling of overlaps makes the use of force fields feasible regardless of the application and the dynamic activation eliminates the unnatural feeling of static force fields. In the future we will take the direction of movement into account to predict targets even more accurately and try to incorporate more context knowledge to further improve interaction with pointing gestures.

References

[1] D. Ahlström, M. Hitz and G. Leitner, "An evaluation of sticky and force enhanced targets in multi target situations," *4th Nordic conference on Human Computer Interaction,* pp. 14--18, 2006.

[2] A. Cockburn and S. Brewster, "Multimodal feedback for the acquisition of small targets Multimodal feedback for the acquisition of small targets," *Human-Computer Interaction,* 2005.

[3] M. Specht, A. Söter, J. Gerken, H.-C. Jetter and H. Reiterer, "Dynamic Force Fields zur Präzisionserhöhung von Zeigegeräten," *Mensch & Computer 2010,* 2010.

[4] J. Yin, "The Beam Cursor: A pen-based technique for enhancing target acquisition," *HCI 2006,* pp. 119-134, 2006.

[5] A. Worden, N. Walker and K. Bharat, "Making computers easier for older adults to use: area cursors and sticky icons," *Proceedings of CHI,* pp. 266--271, 1997.

[6] P. Kabbash and W. Buxton, "The "prince" technique," *Proceedings of the SIGCHI conference on Human factors in computing systems,* pp. 273--279, 1995.

[7] T. Grossman, "The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area," *Proceedings of CHI,* pp. 281--290, 2005.

[8] A. Jansen, L. Findlater and J. Wobbrock, "From the lab to the world: lessons from extending a pointing technique for real-world use," *Proceedings of CHI,* pp. 1867--1872, 2011.

[9] L. Findlater, A. Jansen and K. Shinohara, "Enhanced area cursors: reducing fine pointing demands for people with motor impairments," *Proceedings of UIST,* 2010.

[10] P. Brock, "An Investigation of Target Acquisition with Visually Expanding Targets in Constant Motor-space," 2005.

[11] Y. Guiard and R. Blanch, "Object pointing: a complement to bitmap pointing in GUIs," *Proceedings of Graphics,* pp. 9--16, 2004.

[12] F. van de Camp and R. Stiefelhagen, "Applying Force Fields to Black-Box GUIs Using Computer Vision," In Proc. 1st IEEE Workshop on User-Centred Computer Vision (UCCV), Clearwater Beach, FL, USA, 2013.

[13] T. Mori and K. Uehara, "Extraction of primitive motion and discovery of association rules from motion data," *Proceedings 10th IEEE International Workshop on Robot and Human Interactive Communication,* pp. 200--206, 2002.

[14] A. Schick, F. van de Camp and J. Ijsselmuiden, "Extending touch: towards interaction with large-scale surfaces," *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces,* pp. 117--124, 2009.

[15] "ISO 9241-11: Ergonomic requirements for office work with visual display terminals (VDTs) -- Part 9: Requirements for non-keyboard input devices," ISO, 2000.

Kurzbeiträge