

POMDP Models for Continuous Calibration of Interactive Surfaces

¹Bastian Migge*, ²Tim Schmidt*, ¹Andreas Kunz

¹Innovation Center Virtual Reality - ETH Zurich and ²Palo Alto Research Center

¹CLA G 19.1, Tannenstrasse 3, CH-8092 Zurich

²23333 Coyote Hill Rd Palo Alto, CA-94304

bastian.migge@iwf.mavt.ethz.ch, tim.schmidt@parc.com, kunz@iwf.mavt.ethz.ch

* corresponding author

Abstract

On interactive surfaces, an accurate system calibration is crucial for a precise user interaction. Today, geometric distortions are eliminated by a static calibration. However, this calibration is specific to a user's posture, and parallax distortions occur if this changes (i.e. if the user moves or if multiple users take turns).

Within this paper, we describe an approach to model automatic online re-calibration to cope with changing viewpoints by using Partially Observable Markov Decision Processes (POMDP). Hereby, the viewpoint is stochastically deduced from the precision of user interactions on the surface. To enable the implementation on embedded systems, a small model is defined using states and observations, which are formulated relative to the current assumed viewpoint. We show the structure of a family of models, that can be generated automatically based on the user's position probability and pointing accuracy.

Introduction

Interactive surfaces become more and more state-of-the-art as human-computer interaction technology. Large electronic whiteboards and electronic tables are already widely spread for distributed collaborative development or research teams, and allow multiple users to interact on the same board at a time [Kunz2009]. Also smaller interfaces with a thick glass plane (for safety reasons) are common for ATMs and kiosk applications. In any case, the close coupling between in- and output devices allows a highly intuitive operation and a high integration into the user's task.

To enable this intuitive operation, these systems depend on a good alignment between the displayed image and the involved tracking system. This is done initially by a so-called static calibration (see Figure 1). However, even

distances of a few millimeters between image plane and interaction plane can cause enough optical distortion to significantly impair the user interaction (see Figure 2). This also holds true for back-projection systems. The possibility of distortion increases with the size of the screen and the distance between image and interaction plane (Vz) as shown in Figure 3. The distortion depends on the distance between the viewpoint and the point of interaction, as well as on the viewing angle against perpendicular.

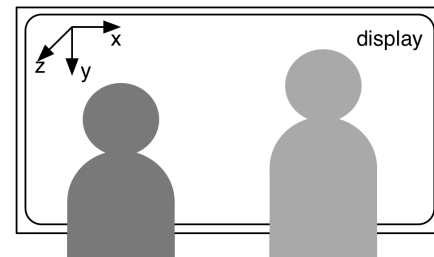


Figure 1: Static calibration is biased by a user's viewpoint.

The optical distortion is strongly biased by the user's characteristics like height, viewpoint, motion, arm length, etc. (see Figure 1). Even for the same user, the static calibration does not take into account his behavioral change over time.

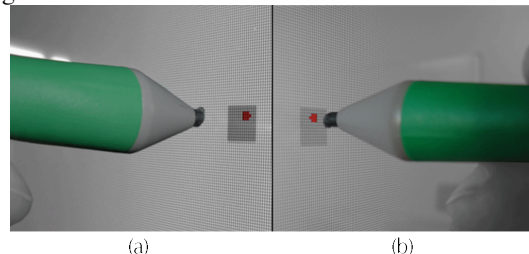


Figure 2: (a) 15 mm offset, (b) 7 mm offset¹.

¹ In both cases, the tip of the pen contacts the interactive surface (a 6 mm glass pane), while the dot in the gray rectangle appears in the image plane underneath.

This so-called Parallax Distortion can be derived from the following geometry (Figure 3):

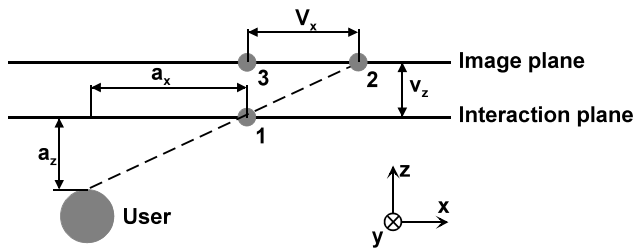


Figure 3: Parallax distortion in x direction.

It is:

V_z : Distance (offset) between image plane and interaction plane

V_x : Resulting parallax distortion in x-direction

a_x : Distance from user to interaction point in x-direction

a_z : Distance from user to interaction point in z-direction

Point 1 in Figure 3 represents the touch point, i.e. the point of contact on the interaction plane. Point 2 is the one the user aims at, while point 3 is the registered point resulting from the parallax distortion. Once the user's viewpoint is known, the distortion V_x can be easily calculated by:

$$V_x = \frac{V_z \cdot a_x}{a_z}$$

(for the x-direction).

An online calibration system could overcome parallax distortion by applying a real-time automatic error correction by continuously adapting to the varying viewpoint. This improves the efficiency of interactions where precise positioning is required, such as commenting, editing drawings and working with a widget based UI.

Previous Work

In order to show the disturbing effect of the parallax distortion on the user's interaction accuracy, we performed a user study on two interactive screens with different V_z . The study was conducted following the standard ISO 9241-9:2000 [ISO2000]. We used Fitt's Law [Fitts1954] to model the pointing process on computer displays. Note that the approach only models one-directional, rapid, aimed human movements. In order to perform measurements based on a multi-directional input, the ISO standard suggests applying Fitts' Law to all dimensions of the input space. Since a detailed description of this approach is missing in the ISO standard, the two multi-directional input tests performed with the two setups were implemented following the suggestions of Douglas et al. [Douglas1999]. The study compares the user's

performance on two interactive surfaces with an offset between interaction plane and image plain of 7 and 15 mm.

The statistical analysis shows that the participants achieved a significant higher performance, i.e. a higher throughput with a reduced input offset.

However, so far this reduced offset can only be reached by modifying the underlying hardware. These modifications are limited, as soon V_z equals the thicknesses of the protective glass of the LC-screen and the glass of the tracking overlay.

The new approach aims to overcome this disturbance due to parallax distortion, which will enhance the accuracy on interactive surfaces. This interaction accuracy is increased by a user-adaptive online re-calibration based on observations about his interaction performance. Due to the noisiness of these observations, the user's continuously changing viewpoint, and due to the possibility of expressing this uncertainty in a discretized model, the problem can be treated as Partially Observable Markov Decision Processes (POMDPs) [Kaelbling1999]. In this model, the agent can choose to change the calibration for a new viewpoint, which is superimposed on the static calibration matrix of the system taking into account the user's height as well as the fact that he might continuously change his viewpoint in front of the digital whiteboard during writing and sketching.

Contribution

In this section, we present a POMDP model for adapting the parallax correction to a user's viewpoint. The viewpoint, which is not directly observable by the system, is inferred from detected interactions. This allows a user-dependent and self-adapting error correction by a stochastic controller to increase the pointing accuracy on the interaction system without any additional hardware.

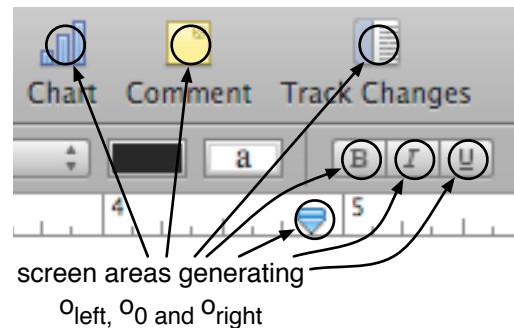


Figure 4: Detection zones around the center of small area widgets.

The main aspect of modeling the problem is the interpretation of user interactions. Contemporary graphical user interfaces like shown in Figure 4 usually provide widgets with a small hit box such as check boxes or radio buttons. The interactions with such elements indicate the parallax error for the user's current viewpoint (see Figure 3). Their centers provide a reference to allow a noisy

measurement of the pointing error, from which we infer the user's relative position to the screen.

The uncertainty in these measurements stems from the user's dexterity. Another source of uncertainty is the possibility that the user changes his position between measurements. The controller takes these uncertainties into account when determining whether and how to adapt the correction function.

Problem Description

We assume the user's viewpoint moves along a discretized x-y plane in front of the display. Due to symmetry, we treat the x (horizontal) and y (vertical) axis analogously in our model derivation and restrict the following discussion to movements in the x-axis. We assume an infinite screen to allow for a simpler model with fine granular tracking characteristics in the majority of possible user positions. This allows a regular model, in which the position on the discretized x-axis represents the deviation between the inferred viewpoint our system compensates for and the actual position of the user's viewpoint. Possible (discrete) interaction points on the screen are realistically limited by a user's arm length and we assume that they are symmetrically distributed around a peak of his perpendicular screen position. Within this interaction zone, each potential hit point corresponds to a correction setting that is derived by the inferred viewpoint and the actual target. These basic geometrical properties in the x-z plane are shown in Figure 5.

In the following we will show, how the compensation problem can be modeled as a POMDP.

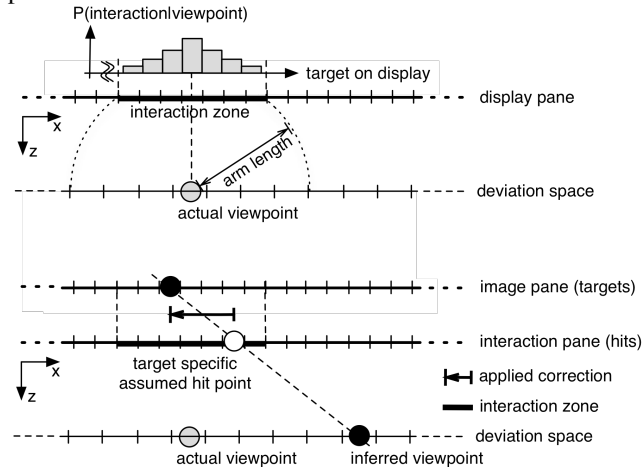


Figure 5: (a) Infinite, discretized display and deviation space, static probability of user interaction (b) applied parallax correction.

Partially Observable Markov Decision Process

A POMDP describes a discrete combinatory decision problem with uncertainty in decision outcomes and partially observable system state.

POMDPs can be described as a six tuple of States, a finite set of Actions the agent can execute and Transition probabilities for successor states conditioned on predecessor state and action. The system is in exactly one

state at any time point. This state however is not directly observable by the agent. This is modeled by a finite set of Observations and Observation Probabilities, which describe the probability getting an observation after executing an action in a certain domain state; and by a cost or Reward Function for executing an action in a certain state.

The goal of the agent is to select the optimal correction action for the current but uncertain situation. For this purpose, the POMDP planner looks for optimal actions based on its current belief states (i.e. the probability distribution over the domain states that represent the agent's current belief about the environment) by considering the accumulated rewards of future worlds. Representing all information available to the agent at a point in time, a belief state fulfills the Markov Property [Hauskrecht2000].

In this framework, all actions are considered to be applicable at each decision point. The observation space consists of discrete observations, which the agent receives through its sensors. It is assumed that the agent receives one of these observations after each decision to get a discrete sequence of correction actions and observations. The models link these discrete entities as follows. The transition model is a set of transition probabilities $P(S'|S,A)$ that represent the probabilistic behavior for all possible combinations of system states and agent decisions. The observation model is a set of probabilities $P(O|S',A)$, each representing the likelihood of the agent receiving a particular observation given that the system changed to a particular state under a particular agent decision. Finally, the reward function $(S \times A) \rightarrow R$ gives the rewards (or costs) for an agent's decision in a system state.

Model

The continuous calibration problem is defined as a POMDP as follows: The system's action space A comprises the possible adjustments to the parallax correction the agent can make at each time step. Its discrete observations stem from user interactions with specific widgets and a null observation occurs when no such interaction took place during a time slot. Finally, the state space (S) represents the deviation of the user's viewpoint to the compensated viewpoint.

The transition model describes the probability of different viewpoint changes between interactions, and the observation model represents the likelihood of receiving a particular observation based on the deviation between user and correction viewpoint. The reward model defines the costs of applying corrections and maintaining undesirable states.

These models allow the agent to maintain a probability distribution over the possible system states, and to make decisions by comparing their future expected costs.

State Space

Solving a POMDP optimally is a hard computational problem. For this reason, we restrict the state space to three elements s_{left} , s_0 , and s_{right} , where s_{left} (s_{right})

accumulates all configurations in which the user's viewpoint deviates to the left (respectively right) from the inferred viewpoint, as illustrated in Figure 6. S_0 is the system state in which both viewpoints align. We assume a fixed relationship between the accumulated states and the deviation space given by a distribution of the deviation over a finite subset of the set of possible positions. In our model, we employ ramp function for the mappings of s_{left} and s_{right} , motivated by the limited screen width and the (empirically measured) independent distribution of user viewpoint positions. This allows for a drastically simpler model, while still having a good descriptiveness for most system configurations. Figure 6 gives an example for probability distributions linking the atomic and accumulated deviation spaces with an accumulated probability mass of 1 for each POMDP state.

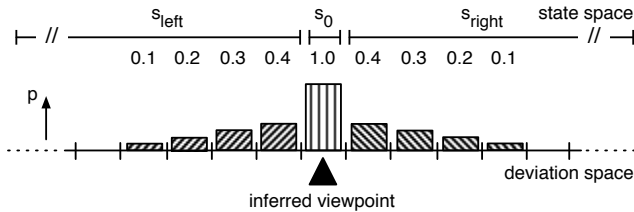


Figure 6: Transformation from deviation space to POMDP state space through fixed finite deviation distributions.

Actions

Actions are defined as changes of the parallax correction. With respect to the model's simplicity, we distinguish three actions a_{left} , a_0 , and a_{right} , where a_{left} and a_{right} adapt the parallax compensation for a user position one unit to the left or respectively to the right, while a_0 leaves the current compensation unchanged.

Observations

With respect to the state space, we define four observations o_{left} , o_{right} , o_{hit} and o_{null} . Our system is time-synchronous; hence the agent needs to make a decision about changing the parallax correction and receives an observation within each time interval. These time intervals are short enough to (reasonably) allow at least one user interaction. We only consider interactions with point targets (or where we can reasonably assume that the user aims at a clearly defined point on the screen), i.e. grid intersections, small buttons, etc. If we register a pointing event in the neighborhood of a target, we compare the hit point's x-coordinate - corrected by the current compensation - with a fixed tolerance interval (hit zone) around the target's x value in an online preprocessing step. Next, we generate o_{left} , o_{hit} , or o_{right} if the hit point is left, within or to the right of this tolerance interval (see Figure 7). If no such interaction takes place, we generate o_{null} .

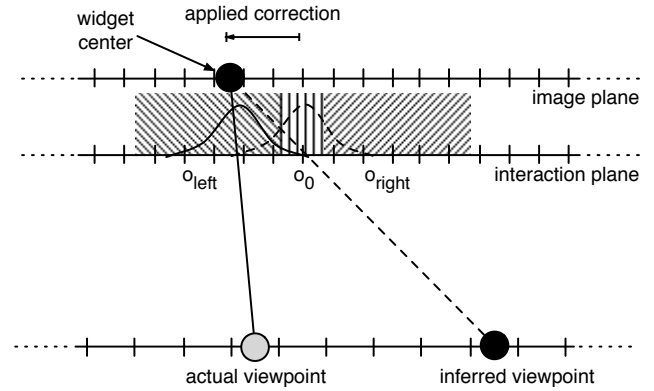


Figure 7: Deriving discrete observation probabilities based on user viewpoint deviation, the user dexterity model and tolerance interval.

Observation model

At each time step, the agent first receives an observation and then decides on a compensation action. Hence, we model them as being independent of the agent's decisions. We derive the probability distribution of getting observations in particular states $P(O|S)$ in two steps as follows.

First, we consider only the relation between the state space and o_{left} , o_0 , o_{right} , and then integrate o_{null} .

As shown in Figure 7, the actual hit point is converted into the discrete observation space as follows. If the hit point is measured within the tolerance interval around a specific target we assume that the respective target was aimed for. Therefore, it is mapped to the relative POMDP observation space according to the current parallax error correction. As Figure 4 illustrates, a hit point is related to a target only if it is measured next to it. Otherwise, o_{null} is generated.

Inside the detection zone, the observation space is defined around the expected hit point: Within the tolerance interval around its center, o_{hit} is generated, while left (right) shifted measurements cause o_{left} (o_{right}).

The user's pointing accuracy is given by a static probability distribution based on empirical data. Since the pointing accuracy depends on the distance between target and view point, the measurements are done with a fix viewpoint and a fix target, and the data has to be adapted with respect to the actual target and viewpoint.

We deduce the probabilities for each observation considering each possible state by first projecting the pointing accuracy distribution to the display plane based on viewpoint to target point angle and distance.

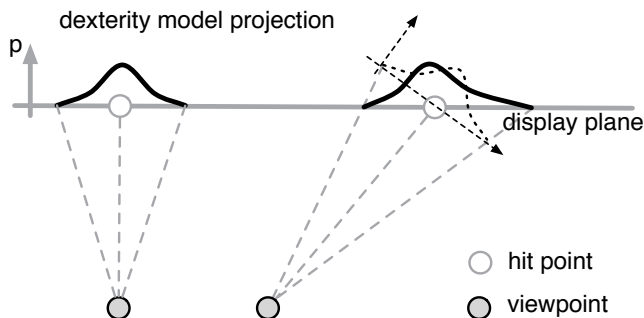


Figure 8: Derivation of angle-adjusted dexterity models by projecting the model onto the display plane.

Figure 8 gives an example of projecting the pointing accuracy distribution to the display plane under a certain viewing angle. Since the original interval width is squeezed left of the touch point and stretched on the right hand side), the observation distribution is adapted to each reachable target cell based on their static interaction probabilities. Finally, the accumulation mapping limits the number of actual viewpoint deviations we have to consider. The relative probability distribution of the accumulated states allows us to convert the observation models for the atomic deviations to models for the parent states s_{left} and s_{right} .

The fourth observation (o_{null}) represents no information gain to the agent. This is either because no interaction took place in the current time interval, or the interaction was with some widget that does not reasonably resemble a point target. In the final model, we distribute this probability to o_{left} , o_0 and o_{right} based on their relative weights for a given state. The remainder is associated with o_{null} .

Transition model

System state transitions are influenced by two distinct factors: The controller realigning the compensation function and the user changing his position in front of the screen. As in our discrete world model both could happen instantly at the same (idealized) point in time, we can reasonably assume them to be independent. In the following, we will derive both models separately. Probabilities for the agent's decisions follow straightforwardly out of our model and the specified ramp function. The a_{left} action results in deterministic transitions for s_{left} ($p(s_{left} | s_{left}, a_{left}) = 1$) and s_0 ($p(s_{left} | s_0, a_{left}) = 1$), s_{right} results in a stochastic transition to either s_0 or s_{right} determined by the ramp function (see Figure 10 for example).

State transition probabilities stemming from user movements can be treated in a similar way. Given an unconditional, relative user movement model for our time step (which we built from data of our empiric user study), we can logically compute the transition probabilities for each state, by first distributing the probability mass as given by the accumulation mapping, computing the discrete folding of this distribution with the relative movement model, and finally by accumulating the

resulting distribution back to our three states s_{left} , s_0 and s_{right} (see Figure 10). Each of these transition models is represented as a 3x3 matrix. The combined model is simply the product of both matrices.

Reward model

The rewards define the benefit of applying actions: Corrections incur a fixed cost as do the states s_{left} and s_{right} .

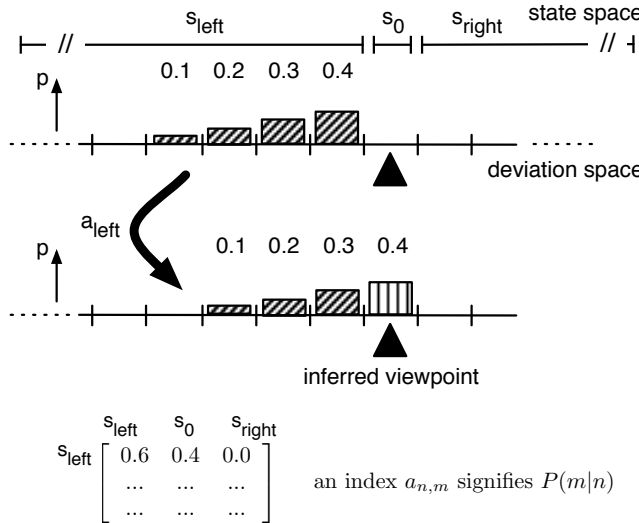


Figure 9: Deriving state transition probabilities due to agent actions. The upper diagram shows the movement of probability mass from state s_{left} when executing a_{left} . The matrix shows the corresponding row in a_{left} 's transition model.

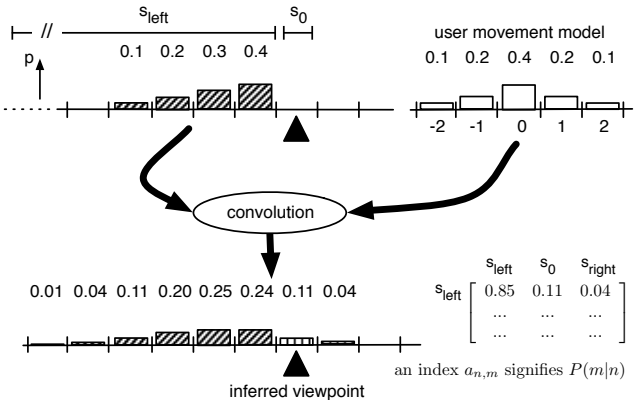


Figure 10: Deriving state transition probabilities due to user actions. The diagram shows the resulting distribution from combining an empiric user movement model with our state accumulation mapping for s_{left} (through convolution) and the resulting matrix row.

Two-dimensional Parallax Correction

So far, we developed the model only for the horizontal (x) dimension on the screen. Due to symmetry, we can apply the same ideas to the vertical direction. Assuming independence, the combined model is simply expressed as

the cross product of both POMDP models. The resulting correction steps are shown in Figure 11.

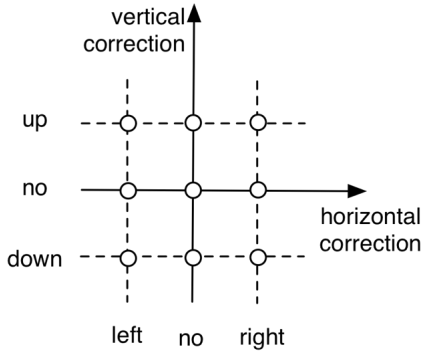


Figure 11: Two-dimensional correction matrix.

POMDP Solver

A standard approach to solve POMDPs is value iteration over the system's belief space, which represents the space of probability distributions over the system states. Such a distribution is called a belief state. The idea is to use dynamic programming to generate the set of n-timestep optimal policies out of the set of n-1-timestep optimal policies by solving the Bellman equation over the belief space. The computation terminates in a fixed-point [Bellman1957]. Hence, the optimal value for a belief state BS can be expressed as:

$$V^*(BS) = \max_{a \in A} R(BS, a) + \lambda \sum_{BS'} P(BS' | BS, a) V^*(BS')$$

with

$$R(BS, a) = \sum_{s \in S} \sum_{s' \in S} R(s, a) P(s' | s, a) BS(s)$$

that describes the one-step reward after executing action a in BS. Formulated this way, the optimal value function can be solved using dynamic programming.

Although a Value Iteration [Bellman1957] converges to the optimum, in practice, the iteration is stopped after reaching an epsilon optimal solution in order to avoid high computational effort. The major problem using value iteration is the infinite number of belief states. Hence, approximations (upper and/or lower bounds) are stored as a finite linear and convex set of alpha vectors (α) [Smallwood1973]. Each of them represents the value of a policy (its assigned action) over the POMDP's belief space. The set of alpha vectors is called value function [Hauskrecht2000]. Due to linearity, it implies vice versa the - so far - best action (policy) for any given belief state BS:

$$\arg \max_{\alpha \in \Theta} \sum_{s \in S} BS(s) \alpha(s)$$

Integration and Implementation

The parallax correction system is integrated into the event chain between pointing device driver and user specific application as shown in Figure 12. Hence, the actual

correction settings are applied to the pointer device coordinates and forwarded to the application within the systems event chain.

To provide a system wide error correction, the error correction must have full access to global pointer events. Our tests showed that reading global events is easily possible whereas changing the event parameters can only be done within the system kernel space for security reasons. Hence, we implemented the parallax correction as a component of a user space application, which simplifies full access to the pointer event parameters.

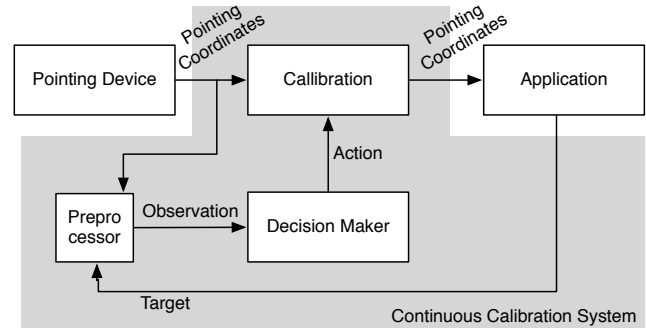


Figure 11: System components

As part of the correction component, a preprocessor is used to generate discrete observations. Therefore, it combines the actual pointing device coordinates and available targets of the application to a sequence of observations and continuously sends them to the decision maker. Based on this, the planner's decisions change the applied correction parameter. Due to computational limitations of embedded systems, the decision maker is implemented as a controller, which holds and updates its belief state (a simple Bayesian update over the 9-state distribution) and makes decision using the implicit policy of the precomputed static value function. The controller computes the dot-product of the belief state and each alpha vector and executes the action associated with the alpha vector that maximizes this product. This can be computed very efficiently for our low-dimensional models. The runtime grows linearly with the number of alpha vectors in the value function. Hence, a continuous adaption to the user specific parallax correction is implemented.

Model Parameterization

The POMDP model depends on various parameters. In this section, we describe the empirical studies from which we derived the parameters for the independent probability distribution of the relative change of a user's position in front of the screen and his pointing accuracy.

Test Setup

For the test, we used a resistive overlay (SMART Tech.) mounted on top of a 50" plasma screen (PDP-502MXE from Pioneer) and an inkless felt-tip pen as interaction device. A resolution of 1024x768 causes a pixel size of

approx. 1.080×0.810 mm. The system was calibrated beforehand (with the SMART toolbox) for orthogonal viewpoints to compensate any system error. To get application independent results, the user did not had to fulfill a certain task.

By using a head tracking system, we recorded the location of the user’s viewpoint in front of the screen within discrete time steps. Then, we discretized the horizontal position values into cells (increasing to the right) and computed a transition model. The matrix A describes the probability of moving from any source to sink cell:

$$a_{ij} = P(\text{sink} = j \mid \text{source} = i)$$

In the second step, the transition model is transformed regarding the relative POMDP state model: staying, moving to the left or right within one time step. Hence, the example probability for moving left, which is represented by the POMDP state s_{left} is given by the accumulated (and normalized) values of the lower left triangular matrix, which can be derived directly from the transition matrix A:

$$P(s_{\text{left}}) = \frac{1}{\# \text{cells}} \sum_{\substack{i,j=1,\dots,\# \text{cells} \\ j < i}} P(\text{sink} = j \mid \text{source} = i)$$

s_{right} is calculated by the upper right triangular matrix and s_0 by the diagonal values. Our tests result in the transition models shown in table 1. The user’s pointing accuracy, which influences the uncertainty of the planner’s observations, is assumed to be Gaussian distributed due to preliminary tests.

transitions		x-axis			y-axis				
		s_{left}	s_0	s_{right}	s_{up}	s_0	s_{down}		
a_0	s_{left}	0.94	0.01	0.05	s_{up}	0.97	0.01	0.02	
	s_0	0.05	0.90	0.05		s_0	0.02	0.96	0.02
	s_{right}	0.05	0.01	0.94		s_{down}	0.02	0.01	0.97
$a_{\text{left/up}}$	s_{left}	0.85	0.09	0.06	s_{up}	0.01	0.01	0.98	
	s_0	0.04	0.01	0.95		s_0	0.91	0.06	0.03
	s_{right}	0.04	0.01	0.95		s_{down}	0.01	0.01	0.98
observations		o_{left} o_0 o_{right}			o_{left} o_0 o_{right}				
a_0	s_{left}	0.70	0.29	0.01	s_{up}	0.65	0.33	0.02	
	s_0	0.05	0.90	0.05		s_0	0.08	0.84	0.08
	s_{right}	0.01	0.29	0.70		s_{down}	0.02	0.33	0.65

an index $a_{n,m}$ signifies $P(m|n)$

Table 1: Transition and Observation model for a 3 step correction system (x and y axis)

Conclusion and Outlook

We primarily focused on the structure of the model and described a basic parameterization. The next step will be to refine the POMDP model. The benefit of a more accurate correction by increasing the number of states must be balanced against increased computational effort. We will enlarge the state and observation space to be able to infer

the user’s viewpoint more accurate. This will allow applying a reasonable more granular correction of the parallax error.

A more complex model requires detailed user studies for parameterization in order to realistically describe the user’s behavior. Additionally, the independent probability distribution of the target position on the screen will be integrated into the state space. Furthermore, observations with different significance have to be defined for interactive surfaces. An example could be that the significance correlates with the requirement of accuracy.

Instead of using an offline calculated and thus static policy, online planning algorithms would allow the agent to deal with redefining the model during runtime, which potentially advances the adaption to the surrounding behavior, but requires the employment of domain specific search heuristics to use the limited computing power more efficiently. Changing the underlying model would allow adapting to the user’s specific behavior and may result in a better adaption and error correction.

References

- [Bellman1957] Bellman,R. *Dynamic Programming*, Princeton University Press, Princeton, NJ (1957)
- [Cassandra2009] Cassandra, A. R. POMDP solver software.<http://www.cassandra.org/pomdp/code/index.shtml> (visited 11/2009)
- [Douglas1999] Douglas, S., Kirkpatrick, A. and McKenzie, S., *Testing Pointing Device Performance and User Assessment with the ISO 9241, Part 9 Standard*, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: the CHI is the Limit 1999, Pittsburgh, Pennsylvania, USA, pp. 215-222
- [Fitts1954] Fitts, P.M., *The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement*. Journal of Experimental Psychology, 47:6 (1954), pp. 381-391
- [ISO2000] International Organization for Standardization, *ISO 9241-9:2000 (E). Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs) – Part 9: Requirements for Non-keyboard Input Devices*. International Organization for Standardization. 2000
- [Kaelbling1998] Kaelbling, L.P. and Littman, M.L. and Cassandra, A.R., *Planning and acting in partially observable stochastic domains*, Artificial Intelligence, Elsevier (1998)
- [Kunz2009] Kunz, A. and Fjeld, M., *From Table-Systems to Tabletop: Integrating Technology into Interactive Surfaces*. To appear in: Müller-Tomfelde, C. (Eds.): Tabletops – Horizontal Interactive Displays. Springer (2009)
- [Smallwood1973] Smallwood, R. D., and Sondik, E. J., *The optimal control of partially observable Markov processes over a finite horizon*. Operations Research 21 (1973), pp.1071-1088

