Interaction Prediction for Content Synchronization of Net-based Shared Workspaces

Bastian Migge, Andreas Kunz
Institute for Machine Tools and Manufacturing
ETH Zurich
Zurich, Switzerland
{*miggeb,kunz*}*@ethz.ch*

***Abstract*—Digital collaborative environments enable spatially separated users to access and modify shared data over network. However, transmission delays of the network lead to inconsistent data and reduce the efficiency of collaboration due to interaction conflicts.**

In this paper, we present a predictive screen-locking algorithm to avoid interaction collisions on net-based shared interactive screens. A model-based predictor calculates the user's next interaction given his past one. The algorithms locks critical objects to the remote station which is less likely to interact with the object. Although the predictor continuously adapts to the user's interaction behavior, an initial interaction model is needed when the collaboration session is started. Hence, we deduce a reasonable, probabilistic interaction model from a large screen collaboration user study.

***Keywords*-Networked collaboration, Human-computer interfaces, User modeling**

I. Introduction

In today's global enterprises, net-based collaboration becomes increasingly important, since it enables global distributed teams to increase working performance [1] without increasing travel expenses. However, for a long time there was a diffuse rejection of tele-collaboration systems, since they did not provide the functionalities and immersion as collocated meetings. Today, digital collaboration systems such as large interactive whiteboards and tables provide workspace sharing, which is intuitively operated with touch interaction and TUI (Tangible User Interfaces) manipulation. Additionally, audio and separate video transmission is widely used, so that net-based collaboration gains popularity. However, when users are collaborating over network, the transmission latency affects the synchronization of the collaborative space and the immersion for the user. Although common round-trip latency lays around 100 ms [2], high variance leads to significantly higher delays up to multiple seconds [3].

Unavoidable network delays lead to synchronization errors: On a shared workspace, a user A interacts with an object based on its current state (e.g. position, shape). If, however, a remote user B is also manipulating the same object at the same time, the changes of user B are not visible to user A due to the network delay. As a result, the data of the shared workspace gets inconsistent. This problem get even more critical since digital shared workspaces allow that user A and B interact at the exact same position on the virtual workspace a time, which would not be possible in collocated collaboration due to haptic collision of the physically present users.

Commonly, the synchronization inconsistency is either been resolved by resetting both station to the last consistent setup, or by defining a single master station, which controls the workspace content. A new way to overcome the latency problem is by interaction prediction in order to avoid simultaneous interaction on the same object. We introduce a prediction software that calculates the future interaction position of the user based on his last one. The prediction enables to efficiently lock only the digital objects on the remote station, which will be effected by the local user in the next time step. Hence, the locking prevents collisions that stem from network latency. To improve the locking mechanisms, the prediction model of the user interaction continuously adapts to the user's behavior while the user works with the system. Nevertheless, the controller needs an initial interaction model to start with. We deduce a model for predicting the user's next interaction area from a user study on two synchronized large interactive stations. The results show that the interaction position is not - as initially assumed - simply Gaussian distributed around the last interaction position, since it depends on the use case as also proposed in [4].

II. Related Work

Common collaboration software, i.e. SMART Notebook [5], avoid the problem of synchronization by transmitting a painted stroke after the local user has finished drawing ('lift up'). As a result, the remote user does not see the new content until it is done. In case of painting applications it is, moreover, reasonable to assume that strokes are not manipulated afterwards by the remote station. On real time synchronized systems like *ClearBoard* [6], *CollaBoard* [7], or *VideoArms* [8] it is, however, crucial to keep the digital content synchronized, since the strokes are shown in real time. The digital content must already be exchanged while the user is sketching, since a video of the user is shown on top if the digital content transferring deictic gestures. Keeping the workspace data synchronized to enable immersion gets even more important if the collaboration software

978-0-7695-4814-2/12 $26.00 © 2012 IEEE
DOI 10.1109/CW.2012.42

supports manipulating digital objects (i.e. pictures) in terms of size, shape and position.

A. User Behavior Models

Most of the existing work in motion prediction addresses to overcome the network latency in the research field of digital virtual environments of distributed 3D applications, such as computer games and virtual walkthroughs. Chan et al. propose prediction methods for human hand motion [9] and 2D computer mouse motion [10], [11] to navigate in virtual environments.

In the context of overcoming the parallax error on interactive screens [12], Migge et al. present a Markovian model to predict the user's position in front of large interactive screens between interactions [4]. They presented an interaction position model for single user office applications [4]. Lewis et.al. propose a synchronization method to support collaborative visualization [2]. However, the literature lacks for interaction prediction on multi-user, net-based collaboration environments.

B. Model Based Prediction

Prediction models describe the behavior of a system over time (system dynamics). The system state is defined over a continuous or discrete state space. With respect to the time t, which also can be defined either discrete or continuous, the system dynamics is modeled as transition function from one state to another: $T_t(sourceState) = sinkState$. This enables a predictor to calculate the future system state based on the current one. However, if the system dynamics can not be expressed with certainty, i.e. the user's interaction behavior on interactive screens, the system dynamics is modeled probabilistically. A probabilistic system model describes the probability of the system developing from one state to another within a certain time: $T_t(sourceState, sinkState) = P_t(sinkState|sourceState)$. This enables the predictor to take uncertainly into account. If the system dynamics depends only on the last state in time, the system is called *Markovian*. A Markov chain is a mathematical framework to describe memoryless stochastic processes. Estimators, i.e. Maximum-Likelihood, deduce a single state from the probability distribution over the future states.

Constructing Markov chains from probability distributions is done with Markov chain Monte Carlo algorithms. The key idea of this family of algorithms is to deduce a Markov chain that represents the desired distribution by sampling against the original. Measuring the user's behavior, i.e. the interaction on interactive screens, provide such a probability distribution.

In this paper, we will deduce a time invariant Markov chain from measurement data that models the interaction position on interactive surfaces probabilistically. The model is applied to lock screen workspace regions on net-based collaboration systems preemptively based on predicting the user's future interaction. The resulting lock controller overcomes synchronization errors that stem from unavoidable network latencies.

III. Contribution

We present a collision avoidance controller to overcome synchronization issues of network collaboration systems, that allow multiple users to remotely share a digital workspace. Collisions occur, if two participants manipulate the same object at a time. Due to the network latency of the shared systems, the stations loose synchronization and the workspace data differs at both stations. To overcome this problem, we introduce a predictive controller that locks objects at the remote station based on the next interaction in time of the local user.

To anticipate the next user interaction position, a prediction model is developed. Since the interaction of the user strongly depends on the setup of the GUI, which differs for each application, the correction controller starts with a standard model and learns the user's behavior while the user works with the system. The learning data is automatically transformed into the prediction model. Hence, the model is adaptable to arbitrary applications.

A. Controller Working Principle

The goal of the controller is to avoid interaction collisions. W.l.o.g. we introduce a bidirectional collaboration system, with one controller implemented at each remote side.

The local controller blocks objects (i.e. strokes, pictures, or screen regions), which might collide within the time horizon of the network latency. The network latency is measured as half of the network connection round-trip-time. Colliding objects are defined as objects on the shared workspace, which might be used by the local and the remote users at the next time step. To lock colliding objects w.r.t. the network latency, the controllers predict the next interaction position of the users based on the last interaction and a probabilistic model. Both controller continuously update the model to presume the interaction position of the user from *adaptProbabilityModel()* and get asynchronous updates from the remote station in *getPredictionRemote()*. If a station detects a probable collision, which means that both users will interact with the same object at next time step with a probability greater than *T*, the corresponding object is locked for the user with less interaction probability. To avoid deadlocks, all locks are freed initially. The predictive locking algorithm, shown in Algorithm 1, is triggered by a user interaction.

B. Prediction Model

The prediction model is defined as probabilistic interaction model. It defines the local **I**nteraction **P**osition in the next time step give the last interaction position

Algorithm 1 Predicitve locking algorithm

```
const T
while IP_local = getLocalInteraction() do
    unlockLocalScreen()
    adaptProbabilityModel(IP_local)

    p_local = P(*|IP_local)            ▷ Query local model
    p_remote = getPredictionFromRemote()
    for obj in screenObjects do
        p_match = p_local(obj) · p_remote(obj)
        if (p_match > T) then
            if (p_local(obj) < p_remote(obj)) then
                lockLocalScreen(obj)
            end if
        end if
    end for
end while
```

$P(IP\_local_{t+1}|IP\_local_t)$. For simplicity reasons, we assume the interaction probability to be time invariant and Markovian. The model is expressed as probability matrix $\mathcal{A}$ of the size $|partition| \times |partition|$. Each row defines the probability of the next interaction, given by the column index j, with respect to the current interaction given by the row index i: $\mathcal{A}(i,j) = P(IP\_local_{t+1} = j|IP\_local_t = i)$.

The prediction model is continuously adapted to the actual user behavior by counting the number of interactions with respect to it's position and the last interaction while the user is working with the system. A FIFO buffer is implemented, to limit the history of considered interactions. To provide a probability matrix, the resulting distribution is normalized with the overall number of interactions.

C. User Study

In this section, we deduce a reasonable initial model from a user study. We measure the interaction position of users collaborating on large interactive screens, that are synchronized remotely. We deduce a probabilistic model of the user's interaction behavior, which is used as initial prediction model of the locking algorithm as described above.

1) Setup and Task: We set up a user study in a collaboration environment to evaluate the interaction behavior of the users, in particular the dynamics of the interaction position. The collaboration environment consists of two identical remote setups that are connected via network. Each room contains a touch sensitive large interactive screen (65" widescreen LC Display SHARP LCD PN 655E) with an interaction tracking device (DViT PA 365 from SMART Technologies [5] as shown in Figure 1. The system gathers touch interaction with the user's finger as well as with TUI's (eraser and color pens) in display resolution (1920 x 1080 [pixel]). Besides the shared workspace, a video of the remote user is shown on a small display. The video and audio link is provided using Skype [13]. The digital content on the screen is shared between the two remote stations in real time by the CollaBoard-Software [7]. The software supports sketching and displaying of image objects.

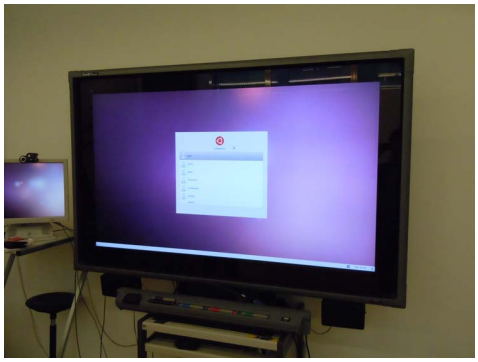

Figure 1. Remote collaboration system setup

The overall number of subjects was 12 (10 male and 2 female) with a median age of 22.5 years. In each study, a pair of users worked together remotely connected over network. The subjects were students and research assistants from our department. The participants were introduced in using the collaboration system. Afterwards, the participants were introduced in the task: Work together on the same problem in the two different rooms of the collaborative environment. They were told to solve the task as fast as possible.

The task of the study is designed to simulate a realistic scenario. The users are motivated to communicate on a everyday problem. In our study, we asked the users to design a floor plan. Basic requirements, like the position of the main door, are given to users. To enforce communication, both participants got different requirements.

The interaction with the digital whiteboard is measured to deduce the interaction model in order to predict the user's behavior. The time of the collaboration systems is synchronized at startup and both stations logged the interaction time, as well as the type of interaction ("lift up" or "press down") and the position in two-dimensional display coordinates.

2) Measurement results: We analyzed the log files from the user study, which contain the position and the type of interaction. We analyze the occurrence of interaction collisions between the two stations before we investigate the sequence of interaction locations on the interactive screen.

Interaction Collisions: Based on 594 interactions on the one and 510 interactions on the other station, we investigate the number of collisions to motivate the lock controller. We define collisions as two interactions, one at the first and the other one at the second station, which occurs at the same time and the same position. The chart in Figure 2 show the number of collisions with respect to different time and space thresholds. The thresholds define the time interval

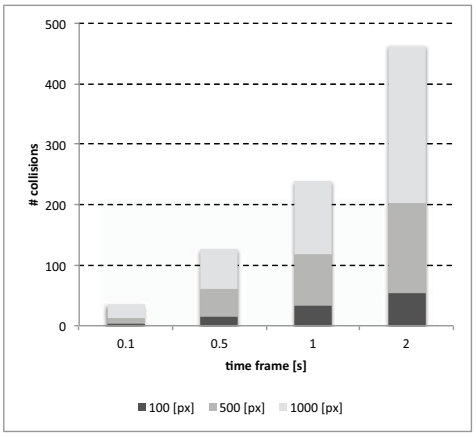

Figure 2. Interaction collisions between remote stations

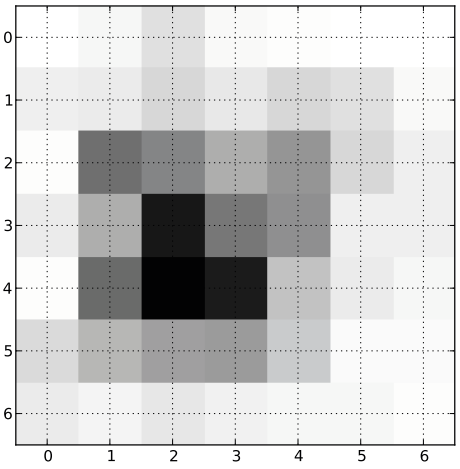

Figure 3. Cumulated interaction distribution (normalized)

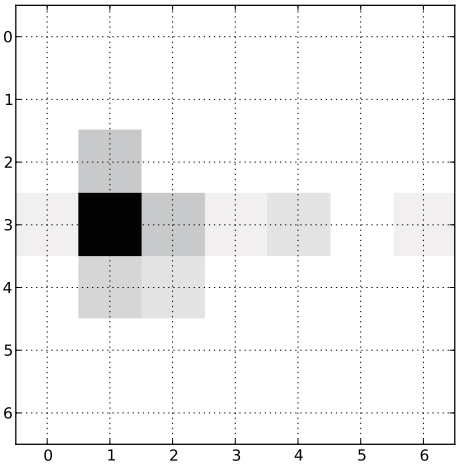

Figure 4. $P(IP\_local_{t+1}|IP\_local_t = (3,1))$

and spacial (euclidean) distance, in which we detect the two interactions as collision. The chart shows the accumulated number collisions for 4 time frames between $\pm$ 0.1 and 2 seconds, which respect to potential network delay. We found that the number of collisions increase with increasing the threshold in time and space. For a time interval of $\pm$ 1 second, the the two users did interact 119 times within 500 px (app. 350 mm) and even 33 times within a distance of 100 px (app. 70 mm). Although we did not measure if the collision did influence the work flow, since we did not measure the context of the interaction, the results indicate that collisions do occur on shared screens.

Interaction location: In total, 1104 interactions were considered. An interaction is defined as placing the finger or the TUI on the screen. The cumulated distribution of the measurements in Figure 3 denotes that the user's interaction tends to be located in the mid left region of the screen. It shows the interaction distribution $P(IP\_local_t)$ on 49 equally sized screen partitions independent of the last interaction. The cumulated distribution is the composition of all distributions, which depend on the previous interaction. The axis are normalized to $[0, 1]$, and the black color indicates 110 interactions and plain white fields indicate zero interactions.

The results confirm the findings in [4]. They showed that the user interacts mainly in the left partition of the screen in office environments. This also holds true for the sketching use case of our study on large interactive screens.

Figures 4 and 5 show the probability of the next interaction position IP_local_{t+1} given that the previous interaction IP_local_t took place in partition (3,1) respectively (3,3). The gray-scale color indicates the probability weight: Black $p = 1$ and white $p = 0$. Each matrix is linearized to a vector and represents a row in the prediction model matrix $\mathcal{A}$.

Similarly to the interaction behavior shown in Figure 4, most of the results show that the user interacts mostly close to the last interaction. The distribution differs in vertical and horizontal dimension: The variance of the horizontal position is higher than the vertical variance, which indicates a stable vertical position but a user's movement in the horizontal direction. Although the horizontal mean lies within the screen partition of the previous interaction, the distribution clearly indicates that the user tends to interact in the mid left region of the screen, as already shown in Figure 3. Since the cumulated interaction probability (in the last section) is not equally distributed, the underlying dependent interaction distributions is also likely to differ.

D. Interaction Prediction Model

To model the interaction dynamics independent of the application, we refer to interaction positions in screen coordinates instead of digital objects. Due to the high resolution of today's displays, the interaction dynamics is not defined for each pixel, but for n areal partitions on the screen. The resulting function of the interaction dynamics is expressed

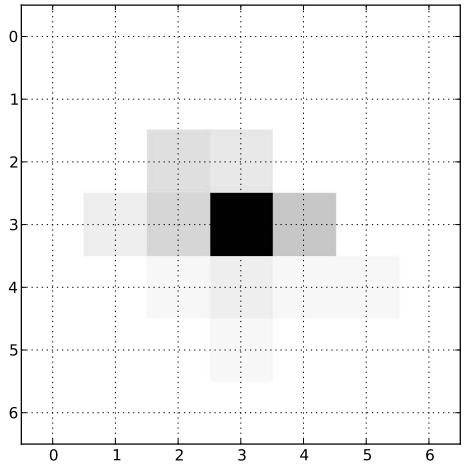

Figure 5. $P(IP\_local_{t+1}|IP\_local_t = (3,3))$

in a symmetric probability matrix $\mathcal{A}$: Each row refers to the screen partition (with partition id pid) the user interacted in. The values of the row define the probability of interacting in the corresponding partition in the next time step: $\mathcal{A}(i,j) = P(pid_{t+1} = j|pid_t = i)$. The interaction transition matrix is used to calculate the probability of the user interacting in a certain screen partition in the next time step given the last detected interaction.

IV. Conclusion and Outlook

In this paper, we introduced a prediction-based locking mechanism to overcome synchronization issues on net-based shared workspaces that stem from network latency. We showed in a user study that collisions occur if the user do collaborate closely on a net-based shared workspace.

Our solution locks digital objects based on predicting the next interaction of the remote users within the time of the network latency. This prevents the data of the remote stations to diverge. The prediction model is continuously adapted to the user's interaction behavior. An initial prediction model is deduced from a user study on large interactive screens. It shows that two sequencing user interactions do not necessarily take place in the same screen region but tend to the mid left region of the screen. Hence, the probability distribution is not symmetric.

To further improve the quality of the looking mechanism, the size of the locked screen area can be reduced and merged automatically. If the algorithm detects multiple interactions within one screen partition, the area is split up. If it detects no interaction in adjacent partitions, the areas are merged with respect to the corresponding interaction probability.

It is, moreover, reasonable to extend the interaction prediction model to depend not only on the last local but also on the last remote interaction $P(IP\_local_{t+1}|IP\_local_t, IP\_remote_t)$. This is motivated by turn taking: If the local and remote user work on the same object, it is natural to work sequentially: One user is listening, while the other explains what he is doing.

The threshold T for detecting a collision can be adapted, too. If the number of detected collisions decreases, the threshold T is increased to reduce the number of locked objects. If the number of collisions increases, T is decreased again.

References

[1] G. Mark, A. Kobsa, and V. Gonzalez, "Do four eyes see better than two? collaborative versus individual discovery in data visualization systems," in *Information Visualisation, 2002. Proceedings. Sixth International Conference on*. IEEE, 2002, pp. 249–255.

[2] R. Lau and K. Lee, "On error bound estimation for motion prediction," in *Virtual Reality Conference (VR), 2010 IEEE*. IEEE, 2010, pp. 171–178.

[3] S. Shakkottai, R. Srikant, N. Brownlee, A. Broido *et al.*, "The rtt distribution of tcp flows in the internet and its impact on tcp-based flow control," 2004.

[4] B. Migge and A. Kunz, "User model for predictive calibration control on interactive screens," in *2010 International Conference on Cyberworlds - Cyberworlds 2010*. IEEE Computer Society, 2010, pp. 32–37.

[5] "http://smarttech.com/."

[6] H. Ishii and M. Kobayashi, "Clearboard: a seamless medium for shared drawing and conversation with eye contact," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1992, pp. 525–532.

[7] T. Nescher and A. Kunz, "An interactive whiteboard for immersive telecollaboration," *The Visual Computer*, vol. 27, no. 4, pp. 311–320, 2011.

[8] A. Tang, C. Neustaedter, and S. Greenberg, "Videoarms: embodiments for mixed presence groupware," *People and Computers XX—Engage*, pp. 85–102, 2007.

[9] A. Chan, R. Lau, and L. Li, "Hand motion prediction for distributed virtual environments," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 14, no. 1, pp. 146–159, 2008.

[10] A. Chan, R. Lau, and B. Ng, "A hybrid motion prediction method for caching and prefetching in distributed virtual environments," in *Proceedings of the ACM symposium on Virtual reality software and technology*. ACM, 2001, pp. 135–142.

[11] ——, "Motion prediction for caching and prefetching in mouse-driven dve navigation," *ACM Transactions on Internet Technology (TOIT)*, vol. 5, no. 1, pp. 70–91, 2005.

[12] B. Migge, T. Schmidt, and A. Kunz, "Pomdp models for continuous calibration of interactive surfaces," in *2010 AAAI Spring Symposium, Embedded Reasoning: Intelligence in Embedded Systems*, 2010, pp. 86–92.

[13] "http://www.skype.com."