GInX - Gaze Based Interface Extensions

Thiago S. Barcelos Carlos H. Morimoto*
Departamento de Ciência da Computação - IME/USP
Rua do Matão, 1010 - São Paulo, SP 05508-090 - Brazil
{barcelos,hitoshi}@ime.usp.br

Abstract

This paper introduces the Gaze based Interface Extensions (GInX) architecture designed for the development of eye-gaze enhanced attentive interfaces. The architecture is composed of 3 modules, the domain, user, and attentive modules. In the absence of information about the user and the domain, the attentive module controls the cursor using gaze and target position information alone. The cursor control can be refined in an attentive way [Vertegaal 2002] as more information about the application and the user are added. The system currently offers 3 different operation modes: Latency, MAGIC, and GInX default mode. In the Latency mode, the cursor position is controlled by gaze and selection is done using dwell time. MAGIC Pointing [Zhai et al. 1999] was suggested to combine the speed of eye tracking with the accuracy of manual pointing devices. GInX extends the concept of Magic Pointing by introducing information about the user and application context in order to eliminate the time required for cursor reacquisition and position adjustment inherent in the original MAGIC Pointing interface. A prototype of GInX was implemented and used to compared the performance of all these 3 modes with a mouse. Our experiments show that GInX outperforms MAGIC Pointing, although the mouse has the best performance overall.

CR Categories: I.2.10 [Artificial Intelligence]: Vision and Scene Understanding; H.1.2 [Models and Principles]: User/Machine systems

Keywords: Gaze aware interfaces

1 Introduction

Current eye tracking technology still presents a few drawbacks when used as an alternative input mode to computer interfaces [Jacob 1993]. Preliminary attempts to use eye-gaze as a pointing device were limited by the eye physiology. The one-degree pointing precision due to constant micro saccades and the fovea size show that the eye has not evolved to be a manipulation tool. Besides, most eye trackers still require a previous calibration session to be performed by each user per each session. Because of the calibration model used, one's head must be kept still during system utilization.

In spite of these problems, the use of eye-gaze as an input mode for computer interfaces is particularly attractive due to the possibility of creating faster interfaces, while reducing fatigue and potential injury caused by operating keyboards and mice, and allowing interaction during situations that prohibit the use of hands, such as while driving a car. A natural first solution is to use EGTs as control input devices to create a "what you look is what you get" interaction paradigm, where pointing is driven directly by eye-gaze, and selection by dwell time [Jacob 1993].

Pointing and selection tasks using eye-gaze data introduce a problem known as the "Midas touch", i.e., everything you look at the computer screen may become eye activated. Several mechanisms have been proposed to overcome such problems but, as described in [Zhai et al. 1999], there are two fundamental shortcomings regarding the direct use of eye-gaze data as a control mechanism. First, due to the nature and behavior of the eye and its movements, the accuracy of EGTs will hardly achieve the accuracy required by current pointing tasks in common computer interfaces. Second, the eye evolved as our primary perceptual input, subject to voluntary and involuntary movements, thus loading this primary sensory input channel with a motor control task must be done very cautiously if at all. But if eye-gaze may not be appropriate for direct manipulation tasks, could it still be used to facilitate such tasks?

The MAGIC (Manual And Gaze Input Cascaded) Pointing proposed by Zhai *et al.* [Zhai et al. 1999] presents a natural way of combining the accuracy of common pointing devices with the speed of eye movements. To understand how MAGIC Pointing works, consider a common pointing task, where the user looks at a target position, then looks for the current cursor position, and physically moves the mouse (or uses other pointing device such as a touchpad or trackpoint) to place the cursor over the target. If initially the user is not holding the mouse, it is also very common not to know the current cursor position, so time is spent finding the cursor in order to place it over the target. The initial mouse motion can be considered as a hard evidence that the user wants to move the cursor, and therefore the eye-gaze data can be used to warp the cursor instantaneously to a previous recent fixation, which in general corresponds to the target position, so the cursor "magically" appears over the target, without the need to find the cursor or move it at all across the screen. When the user is holding the mouse, the eye-gaze data can be used to warp the cursor when the eye is fixating a point distant from the current cursor position and the mouse is moving toward the target, therefore saving the time required to drag the mouse.

The next section introduces GInX, the Gaze based Interface Extensions, which is a flexible 3 module architecture designed to facilitate the development of gaze enhanced applications, and we show that this architecture fits well within the paradigm of attentive interfaces. We have designed an experiment to compare the performance of the 3 operation modes available in GInX using a Fitt's pointing task. Section 3 defines the experiment and Section 4 presents the results obtained from 26 subjects. Section 5 concludes the paper.

2 Gaze Based Interface Extensions

The idea behind MAGIC Pointing is to enhance manual devices with the speed of eye movements, combining the strengths of these two input modes, and minimizing each other's weaknesses. The cursor is controlled by the manual input device for fine accurate movements, and by the EGT for rapid large cursor displacements that might not be very accurate. One limitation of MAGIC Pointing

*We would like to thank FAPESP (Fundação de Amparo à Pesquisa do Estado de São Paulo) for their financial support.

ETRA 2008, Savannah, Georgia, March 26–28, 2008.
© 2008 ACM 978-1-59593-982-1/08/0003 $5.00

is that the cursor can be warped anywhere, even into empty areas, so that the user must spend some time reacquiring the cursor before moving it towards the target.

A typical computer application in a WIMP interface requires the user to point at and click on selected targets, such as buttons, icons, scroll bars, etc, to complete a task. Because MAGIC Pointing warps the cursor to a screen coordinate defined by the point of regard as defined by the EGT, with or without an offset, the user may be required to manually adjust the cursor position quite often.

2.1 Context Information and Target Acquisition

When analysing the task of acquiring a target, one of its sub tasks is the visual search for the target. At this point, the gaze position information provided by eye trackers can be used to speed up the task. Glenstrup and Engell-Nielsen [Glenstrup and Engell-Nielsen 1995] used the keystroke-level model developed earlier by Card, Moran and Newell to argue that some amount of time necessary to complete the task can be saved. The following equation shows a simplified version of the model, where $T_{execute}$ is the time necessary to complete a pointing task with a mouse:

$$T_{execute} = T_{MT} + T_{Eye} + T_M + T_R \quad (1)$$

Where T_{MT} stands for the time required for mental operations, T_{Eye} represents the time necessary for visually searching the target, T_M is the time necessary for manual operations (moving the mouse and performing mouse clicks) and T_R is the elapsed time for the system response. In this model, if eye gaze tracking is used to help the pointing task, T_M can be in theory eliminated and T_R can be replaced by the $T_{R'}$, the time it takes for the system to respond to target selection.

We know that this model is not precise. Skilled users can start the mouse movement in the right direction even before they look at the target. Besides, the anatomical properties of our eyes give us indication that completely eliminating the manual operations can overload the eyes with a manipulation task they are not prepared to. Apart from this limitations, the model can provide a reasonable approximation to selection time and justify our efforts in applying eye tracking to accelerate target acquisition tasks.

Because eye tracking technology is still far from perfect, due to calibration errors for instance, a user can be looking at a target and the eye tracker can report a gaze position that is not on the target, but (hopefully) close enough to it. At this point, when target position is available to the eye tracking system, the target can be used as a reference point to "attract" the reported gaze point. In [Sibert and Jacob 2000] Sibert and Jacob described a proprietary test application for evaluating eye gaze as an object selection method. Raw data from the eye tracker was filtered to identify useful events, such as eye fixations and saccades. Even if the eye tracking precision is affected by input noise or calibration imprecision, their system reports a target selection if the given gaze point is close enough to a target.

These ideas can be applied to a real WIMP interface, which are the standard in desktop systems today. From everyday interaction with these interfaces, users got used to dealing with *widgets*, or controls. All accomplished tasks involve dealing with text boxes, icons and buttons. Therefore, when decomposing a task in a sequence of sub tasks, it is highly likely that one of these sub tasks will be "select widget X", or "point to widget Y". Users are constantly operating their pointing devices to acquire targets, and most of the time those targets are the widgets.

So, when dealing with WIMP interfaces, incorporating information about widgets (for instance, widget position information) in a gaze-assisted pointing system is a reasonable strategy to enhance interaction speed and comfort. We can identify the following *potential* advantages of such a technique:

- *Decrease of manual cursor adjustment.* When the user looks at a widget, warping the mouse cursor over a convenient position of the widget can reduce the necessity of moving the mouse to the exact clicking position.
- *More comfortable interaction.* When the mouse is warped to a exact position, users feel that they really "hit the target", or, in other words, they feel that the eye tracking system is really precisely responding to their commands.
- *Increased predictability.* After a few times using the new system, users may get used to the fact that the mouse will appear over the widget. When users get to predict the system behavior in advance, their performance can be even better.

2.2 GInX

Gaze based Interface Extensions (GInX) extends the MAGIC Pointing idea by combining context information about the interface widgets, task, and the user. For instance, GInX might use information about current active widgets to warp the cursor, potentially saving the time required to reacquire the cursor and fine positioning it with the manual pointing device.

Another contribution of our work is the modular design of the extensions that can be turned on/off to change the behavior of the module, or trained to create new behaviors in different tasks. For example, the user or application can select the prefered operation mode for each particular context. Figure 1 shows a block diagram of the GInX architecture. The domain module contains information about the task being performed by the user which is application dependent. The user module keeps information about the user state, preferences and behaviors. The attentive module receives gaze data and information from the other modules to compute an expected target. If this target differs from the user selected target, the user model can be refined.

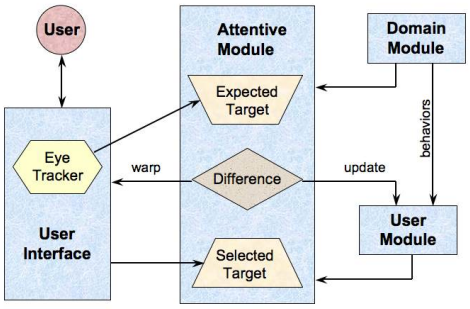


Figure 1: *Block diagram of GInX architecture.*

This 3 module architecture is quite flexible and can be used as an attentive interface [Vertegaal 2002]. An attentive interface dynamically prioritizes the information it presents to its users according to context information, such that information processing resources of both user and system are optimally distributed across a set of tasks. Vertegaal [Vertegaal 2002] suggests the classification of different types of attentive interfaces based on their ability to monitor the user's attentive state, the kind of measurements used by the sensing technology, and the way in which an attentive system may increase

or decrease the load of the user, system or network. GInX architecture does not require the explicit definition of the user and domain modules, so that it can benefit current non-gaze enhanced applications as well. When such modules are available, they may overwrite the default behavior of GInX, optimizing it to their needs.

3 Experimental Design

The performance of the MAGIC Pointing system was compared to a touchpad using a Fitts' pointing task. Subjects were asked to point and click at targets appearing in random order. To test the performance of the 3 operation modes of GInX, we use a similar test with a grid of targets. The task is to click at the highlighted target on the grid (see Fig. 2).

Fitts law establishes that the time to accomplish a pointing task increases with the amplitude of the motion and decreases with the size of the target according to the following equation:

$$T = a + b \log_2(2A/W) \quad (2)$$

where T is the time to complete the task, A is the amplitude of the movement, W the size of the target, and a and b are empirical constants that can be computed using linear regression techniques. The term $2A/W$ is usually called index of difficulty of the task, and b is known as the index of performance. In [Mackenzie et al. 1991] it is shown that using $A/W + 1$ as the index of difficulty results in a better fit to the data.

In the original MAGIC Pointing experiment the size of the targets (W) and the distance (A) were varied. In our experiment, three target sizes were used. Because the accuracy of the EGT is around 1 degree, our smallest target was chosen to be about 1 degree, the medium size target 2 degrees, and the large target to be about 4.5 degrees of visual angle. The amplitude was varied by selecting a target within the grid that was at an appropriate distance. Three distances was used for testing, 5cm, 10cm, and 20cm, distributed within a 20cm $\times$ 30cm display area. One last parameter that highly influences the error rate of the pointing task is the grid size, i.e., the distance between targets in the grid. Three grid sizes was also used in our experiments, 0cm, 2cm, and 4cm. Figure 2 shows an example of a screen used during testing.

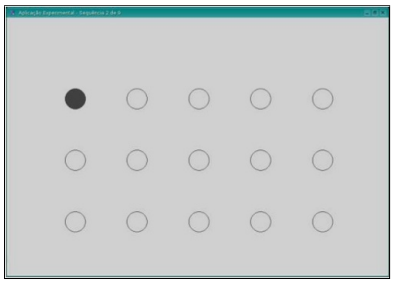


Figure 2: *Sample screen used during testing*

The typical distance of the user to the monitor screen was about 50cm. Each subject was asked to click on the highlighted target as fast as possible. By combining the target size, grid size and amplitude, 9 different screens were generated. For each screen, 15 random targets were selected.

The architecture offers 3 modes of operation: Latency, MAGIC and GInX. The first mode is a simple dwell time eye pointing and selection interface, where gaze data is used for pointing, and the selection is made by fixating the gaze over the desired target over a certain period of time (we used 250 ms). A 10 minute training session before the test was allowed for each user to practice with MAGIC Pointing, GInX and the Latency interaction modes.

Input Mode	a [ms]	b [ms/bit]	IP [bits/s]
Mouse	455	160	6.2
Liberal GInX	769	99	10.1
Conservative GInX	868	129	7.7
Liberal MAGIC	800	110	9.1
Conservative MAGIC	901	127	7.8
Dwell time	694	8	125

Table 1: *Coefficients of the linear regression to Fitt's Law.*

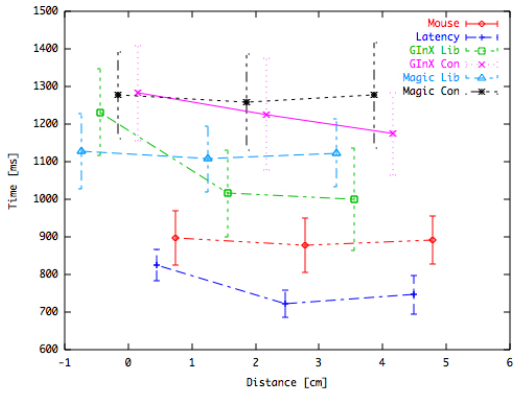


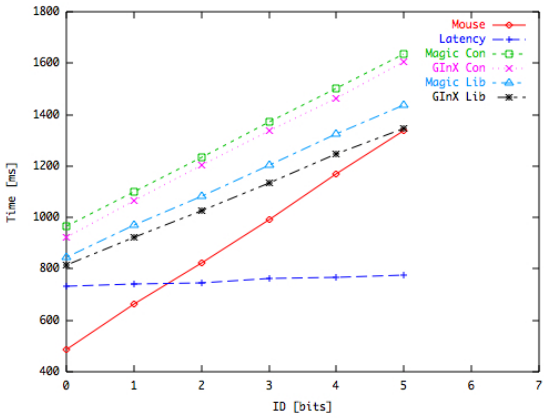
Figure 3: *Time to complete the task as a function of the grid size. The vertical bars correspond to a confidence of 95% around the mean.*

4 Experimental results

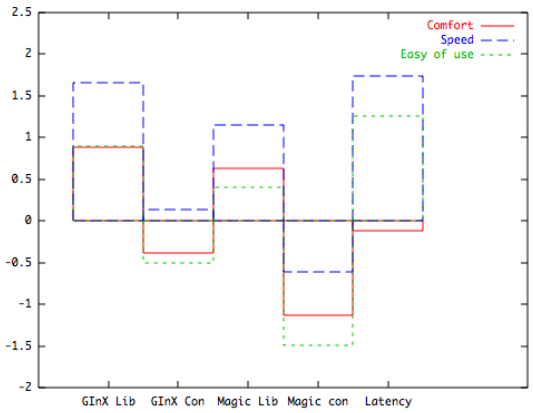
All subjects were briefed at the beginning of the session. A total of 26 college students participated in the experiments. The average computer experience of the group was about 10.3 years. Figure 3 shows the time to complete the task as a function of the grid size. Observe that the grid size has a significant impact on the completion of the task. Both MAGIC and GInX require some space between targets in order to become useful. Although GInX seems to perform a little better, this result is not statistically significant.

Figure 4a shows the lines computed from linear regression of the data to Fitts' Law. As in the original MAGIC experiment, the manual device outperforms the other modalities, although the index of performance of the MAGIC and GInX modalities are better than the mouse. The coefficients for these lines are shown in Table 1. This table shows that the performance results for MAGIC Pointing, GInX and the mouse are similar. The mouse has the worst IP (index of performance) of 6.2 bits/s. Latency has a high IP, but the number of errors is unacceptable for small grid distances. But as long as the target distances are kept within limits, the time to complete the task is almost constant. This result is expected due to the velocity of the eye-gaze. Also as expected, the IP of GInX is a little superior than that of MAGIC.

The users also answered a questionnaire at the end of the experiment for subjective evaluation, according to the criteria of Easy of Use, Speed, and Comfort. The results are presented in Fig. 4b. According to these results, the preferred mode was the dwell time, for its speed, although some subjects complained about the Midas touch problem.



(a)



(b)

Figure 4: *a) Lines computed from linear regression of the data, and (b) Results from the subjective evaluation.*

The Liberal MAGIC and GInX were also well evaluated, with a slight preference for GInX. They both were reported to be easy to use. The conservative modes received negative evaluations though. The conservative MAGIC was the worst modality according to the subjects.

The widget selection routine does not add any significant overhead to the eye tracking system, as it continues to run at approximately 30 frames per second. The user is able to point at widgets as soon as the window is shown on the screen. Still some limitations due to the current eye tracking technology precision were noticed: when widgets are too close (like, for instance, the standard *OK* and *Cancel* buttons in most dialogs) the user is not able to make the system differentiate between the two. Surprisingly, a couple of users who were allowed to freely explore the system for more than 10 minutes found a strategy to differentiate between such close buttons: they looked to one button, than fixated at a spot far away from the button area (normally the opposite side of the screen) and then fixated at the other button. Future experiments can show if this pattern could be adopted by other users in a more controlled experiment.

The preliminary tests indicated that different widget design and positioning techniques may arise to produce applications that are better suited to respond to gaze input. For example, bigger widgets and alternates between opposite screen sides seem to avoid some of the limitations of current eye tracking technology. A similar technique was used by Wang, Zhai and Su [Wang et al. 2001] in the development of a Chinese language input system that combines the keyboard with MAGIC Pointing. Buttons are displayed in two rows for the user to select one of many Chinese characters. Instead of aligning the buttons in two rows, they grouped the buttons in a 'W' shape order to minimize selection errors.

5 Conclusion

The Gaze based Interface Extensions (GInX) is a flexible architecture that facilitates the development of gaze enhanced attentive applications. The system was fully integrated with the Linux operating system with an eye-gaze tracker. The domain module is defined using the list of active widgets provided from the window manager, for the active window. The user module currently only records the fixations in time, so that the application can collect eye behaviors and act upon certain patterns such as reading, or searching. The attentive module currently offers three gaze operation modes: Latency, MAGIC pointing, and the GInX default mode. When in the Latency mode, pointing is directly controlled by the user's gaze, and selection is performed by dwell time. Magic Pointing naturally integrates the speed of eye movements with the fine precision of current pointing devices. GInX architecture allows to extend the concept of Magic Pointing (and latency as well), by introducing information about the user and application context in order to eliminate the time necessary for cursor reacquisition and position adjustment inherent in the original MAGIC Pointing interface.

This prototype allows any application under Linux to be used with any of the 3 gaze operating modes. We have designed and conducted user experiments to evaluate the performance of these modes and compare them with the mouse. The results show that GInX and MAGIC have a better index of performance than the mouse, meaning that they behave better for more difficult pointing tasks, although the mouse still outperformed both techniques. Although most subjects liked the Latency mode due to its speed, our results show that they are not appropriate for complex interfaces. Subjective results also showed that the users liked the liberal modes of MAGIC and GInX operation modes, due to their comfort. They also report that these modes seem faster than the mouse, though the results prove that they are not.

References

GLENSTRUP, A., AND ENGELL-NIELSEN, T. 1995. *Eye Controlled Media: Present and Future State*. Master's thesis, University of Copenhagen DIKU (Institute of Computer Science), Universitetsparken 1 DK-2100 Denmark.

JACOB, R. 1993. What you look at is what you get. *IEEE Computer 26*, 7 (July), 65–66.

MACKENZIE, I., SELLEN, A., AND BUXTON, W. 1991. A comparison of input devices in elemental pointing and dragging tasks. In *Proc. ACM SIGCHI - Human Factors in Computing Systems Conference*, ACM, 161–166.

SIBERT, L., AND JACOB, R. J. K. 2000. Evaluation of eye gaze interaction. In *Proc. ACM SIGCHI - Human Factors in Computing Systems Conference*, ACM, 281–288.

VERTEGAAL, R. 2002. Designing attentive interfaces. In *Proc. of the Eye Tracking Research & Applications Symposium*, ACM, 23–30.

WANG, J., ZHAI, S., AND SU, H. 2001. Chinese input with keystroke and eye tracking. In *Proc. ACM SIGCHI - Human Factors in Computing Systems Conference*, ACM, 349–356.

ZHAI, S., MORIMOTO, C., AND IHDE, S. 1999. Manual and gaze input cascaded (magic) pointing. In *Proc. ACM SIGCHI - Human Factors in Computing Systems Conference*, ACM, 246–253.