Interfaces prestativas baseadas em visão computacional e informação de contexto

Thiago Schumacher Barcelos

Dissertação apresentada

ao

Instituto de Matemática e Estatística

da

Universidade de São Paulo

para

obtenção do grau

de

Mestre em Ciência da Computação

Área de Concentração: **Ciência da Computação**

Orientador: **Prof. Dr. Carlos Hitoshi Morimoto**

São Paulo – 2005

Interfaces prestativas baseadas em visão computacional e informação de contexto

Este exemplar corresponde à redação final da dissertação devidamente corrigida e defendida por Thiago Schumacher Barcelos e aprovada pela Comissão Julgadora.

São Paulo, 12 de julho de 2005.

Banca Examinadora :

- Prof. Dr. Carlos Hitoshi Morimoto (Orientador) - IME-USP
- Prof. Dr. Flávio Soares Correa da Silva - IME-USP
- Profa. Dra. Maria Cecília Calani Baranauskas - UNICAMP
- Profa. Dra. Nina Sumiko Tomita Hirata (Suplente) - IME-USP
- Profa. Dra. Maria da Graça Campos Pimentel (Suplente) - ICMC-USP

Agradecimentos*

Inicialmente gostaria de agradecer ao Prof. Dr. Carlos Hitoshi Morimoto, meu orientador, pela oportunidade de fazer pesquisa em uma área pela qual sempre me interessei muito e por sua paciência e preocupação com a vida e planos futuros de seus orientados.

Devo um agradecimento especial aos colegas do Laboratório de Fisiologia do Comportamento do Instituto de Ciências Biomédicas da USP, coordenado pelo Prof. Dr. Ronald Ranvaud, pelo inestimável apoio na definição e condução de experimentos que foram fundamentais para a conclusão deste trabalho. Em especial agradeço aos amigos Edgard, Adhemar e Luiz por manterem sempre as portas do seu laboratório abertas e pela constante disposição em conhecer e discutir os rumos deste trabalho.

Agradeço aos grandes amigos que fiz no curso de Processamento de Dados da Escola Técnica Estadual de São Paulo; mesmo que hoje nossas áreas de atuação profissional não sejam mais as mesmas, cada vez mais nossos sonhos, anseios e afinidades são. A todos o meu muito obrigado pelo apoio e carinho que foram fundamentais para que eu chegasse "são e salvo" até aqui!

Aos amigos do BCC '99 do IME-USP deixo o meu agradecimento pela companhia nessa longa e às vezes exaustiva caminhada, que em muitos momentos se tornou mais leve graças a nossos bate-papos, estudos e trabalhos em grupo. E, fora da universidade, foi muito importante a convivência com os amigos e irmãos escoteiros do Grupo Escoteiro Ararigboia – 123SP que tanto me ajudaram a ser uma pessoa mais ativa e mais feliz.

Finalmente, agradeço a meus pais, Raul e Ivonete, por sempre acreditarem no meu potencial e pela ajuda e motivação que me guiaram até este ponto e me levarão mais longe no futuro.

*Este trabalho foi financiado pela FAPESP - Fundação de Amparo à Pesquisa do Estado de São Paulo, através do processo 03/03227-4

Resumo

O surgimento de dispositivos computacionais em formatos diversos e a integração digital de pessoas que possuem limitações físicas motivam a criação de formas alternativas de interagir com computadores que não estejam mais restritas somente à operação de dispositivos como o teclado e o mouse. Uma possibilidade é incorporar ao processo de interação a informação trazida por sistemas de rastreamento do olhar. Estimando a posição observada pelo usuário, é possível obter evidências sobre qual é o objeto de seu interesse no momento. Além disso, o olho é capaz de modificar o seu foco de atenção muito rapidamente, trazendo a possibilidade de incorporar essa velocidade ao diálogo com o computador. Neste trabalho discutimos o estado-da-arte no uso do rastreamento do olhar em sistemas interativos; então descrevemos a construção de um sistema de rastreamento do olhar e propomos uma nova técnica de interação, denominada SAPOC – Sistema de Apontamento Prestativo Orientado pelo Contexto, que associa o olhar com informação sobre o contexto da interação para facilitar a execução de tarefas de apontamento. Partindo do conceito introduzido pelo MAGIC Pointing, uma técnica de interação desenvolvida anteriormente, incorporamos informação sobre o posicionamento dos alvos selecionáveis na interface para auxiliar na realização dessas tarefas em menor tempo e com maior conforto. Descrevemos um experimento com a participação de usuários para comparar a nossa técnica com o MAGIC Pointing e o uso apenas do mouse na execução de uma mesma tarefa de apontamento e seleção, e a partir de seus resultados discutimos em que situações essa nova técnica é mais apropriada.

Abstract

The development of new computer devices in different formats and the necessity to allow the use of computers by people with special needs motivate the development of new ways of interacting with computers, not restricted anymore to the use of regular devices like keyboards and mice. A possible strategy is to incorporate information brought by eye gaze trackers to the interaction process. By estimating eye gaze position, it becomes possible to infer which object is the focus of the user's visual attention. As the eye is able to shift its focus of attention very quickly, we would like to incorporate this speed to the dialog with computers. In this work we discuss the use of eye gaze trackers to interact with computers and describe the development of an eye gaze tracking system, which we use to propose and develop an interaction technique that combines eye gaze with information about the task context to facilitate the execution of pointing and selection tasks. We use the concept introduced by MAGIC Pointing, a previously developed technique, to propose this new technique that uses information about the position of selectable targets in order to help the user execute pointing and selection tasks faster and more comfortably. Then, we compare performance and usability aspects of our technique, the MAGIC Pointing and the mouse based on experimental results with users to discuss the advantages and disadvantages of this new interaction technique.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Nas décadas de 80 e 90, se consolidou nos computadores de uso pessoal o uso de uma arquitetura de periféricos de entrada e saída padronizada: entrada de dados através de teclados alfanuméricos e do mouse; saída de dados visual através de um monitor, e eventualmente complemento sonoro através de uma placa sintetizadora de sons. Essa arquitetura se popularizou como o paradigma de interação WIMP (Windows, Menus, Icons and Pointing), usado em sistemas como o MacOS, da Apple e o Windows, da Microsoft.

Esse paradigma de interação se popularizou por simplificar o diálogo com o computador: nele, a interface apresenta representações de objetos sobre os quais o usuário pode atuar. Quando deseja atuar sobre um objeto em particular, o usuário indica essa intenção *apontando* o objeto com um cursor que aparece na tela de vídeo e é controlado manualmente através da manipulação de um mouse, *trackball* ou *touchpad*. A idéia é que os objetos possam ser manipulados através de um gesto comum da comunicação humana - o apontamento.

Entretanto, apesar de ter introduzido uma relativa simplicidade de uso, essa forma de interagir ainda exige que o usuário seja treinado a usar um dispositivo, cuja manipulação não é natural, para realizar o apontamento de forma indireta. Essa forma de interagir também impõe restrições ao uso de computadores por pessoas que, por alguma limitação física, não possam operar um dispositivo de apontamento. Além disso, estamos assistindo ao surgimento de dispositivos computacionais em formatos alternativos, tais como computadores de mão e *tablet PCs*, bem como à incorporação de recursos computacionais em equipamentos que antes

não os possuíam, como automóveis e eletrodomésticos. A entrada de dados através do uso de teclados e dispositivos de apontamento pode não ser adequada para o diálogo com esses novos dispositivos de formatos tão diversos e voltados para contextos de utilização igualmente diversos.

Uma proposta que vem sendo discutida é a da multimodalidade, ou seja, incorporar ao diálogo formas alternativas de entrada, tais como a voz, o olhar, ou movimentos corporais, de forma a criar diferentes formas de diálogo para usuários de diferentes faixas etárias, níveis de instrução e limitações físicas e motoras [OC00], além de possibilitar a criação de formas de interação que aproveitem outras habilidades de comunicação naturalmente adquiridas pelos usuários no convívio social [Tur98]. Através de uma maior variedade de modalidades de comunicação, abrem-se novas possibilidades para que os computadores não fiquem mais o tempo todo passivamente esperando por uma ação do usuário e possam, a partir de um maior conhecimento do estado do usuário, tomar a iniciativa de auxiliá-lo na execução de suas tarefas – convencionou-se chamar as interfaces humano-computador criadas segundo esse princípio de *interfaces prestativas* [Tur98, Ver02].

Em particular, a incorporação de técnicas de rastreamento do olhar em interfaces tem sido considerada há vários anos [SA00, ZMI99, GEN95, Jac90, Bol81]. Através de diferentes abordagens, o objetivo de tais técnicas é a determinação do ponto observado por um indivíduo em uma superfície 2D, ou a estimação da linha de visão do indivíduo, quando não é considerada a observação de uma superfície em particular. Como nós usamos nossos olhos para explorar constantemente o ambiente à nossa volta, potencialmente essa informação pode colaborar na construção de um canal de comunicação natural e rápido em comparação com os métodos de entrada tradicionais, tais como o teclado e o mouse.

Porém, alguns fatores devem ser levados em consideração no desenvolvimento de técnicas de interação que incorporem o rastreamento do olhar. Restrições inerentes à tecnologia atual introduzem uma quantidade considerável de ruído nos dados oriundos do processo de rastreamento [GEN95]. Além disso, pela sua própria estrutura física, o olho não é uma "ferramenta" que usamos para "manipular" o ambiente à nossa volta como as mãos, por exemplo.

Assim, interpretar corretamente os dados fornecidos pelo processo de rastreamento do

olhar é um desafio no projeto de tais técnicas de interação.

Nosso trabalho se desenvolve a partir de uma técnica de interação desenvolvida por Zhai, Morimoto e Ihde [ZMI99], denominada MAGIC (Manual and Gaze Input Cascaded) Pointing, que usa a informação da posição da tela observada pelo usuário, associada ao controle motor de um dispositivo de apontamento tradicional para a execução de tarefas de apontamento em interfaces WIMP. Essa técnica permite que o cursor usado para o apontamento se mova de forma mais "prestativa", ou seja, apareça na vizinhança da posição que recebe a atenção visual do usuário.

Neste trabalho propomos associar ao MAGIC Pointing informações sobre o contexto atual da interação, criando uma interface mais prestativa, a fim de investigar como essa combinação pode produzir uma interface melhor tanto em aspectos objetivos, como a velocidade ou precisão na realização de tarefas, quanto em aspectos subjetivos como a satisfação dos usuários. Para tanto, desenvolvemos uma versão para Linux do sistema de rastreamento do olhar proposto por Morimoto, Koons, Amir e Flickner [MKAF00], que, juntamente com informações obtidas a partir do contexto da interação, é usado pelo MAGIC Pointing e por suas extensões a serem propostas por nós. Além disso, projetamos e conduzimos experimentos com a participação de usuários operando o sistema para a obtenção de medidas que permitam avaliar as vantagens e desvantagens dessas extensões.

1.1 Organização deste trabalho

No Capítulo 2 descrevemos algumas características da estrutura e dos movimentos realizados pelo globo ocular humano relevantes para compreender as dificuldades encontradas no desenvolvimento de uma técnica de rastreamento do olhar. Também introduzimos e comparamos diferentes técnicas para rastreamento do olhar descritas na literatura, discutindo as vantagens e desvantagens de cada uma.

No Capítulo 3 damos uma definição para *técnica de interação*, e descrevemos como o rastreamento do olhar foi empregado até o momento na área de Interação Humano-Computador através da comparação entre algumas técnicas de interação já implementadas; exibimos também alguns exemplos da associação do rastreamento do olhar ao uso de informação de

contexto que tem sido propostos.

No Capítulo 4 descrevemos a implementação do nosso rastreador do olhar e de nossa técnica de interação, a partir da discussão do estado-da-arte feita nos capítulos anteriores. No Capítulo 5 exibimos os resultados da avaliação do nosso sistema obtidos a partir de experimentos com a participação de usuários. No Capítulo 6 concluimos o trabalho e discutimos alternativas para avanços futuros.

Capítulo 2

Visão e rastreamento do olhar

A visão é um dos sentidos mais importantes na percepção do ambiente à nossa volta; a todo momento a usamos para obter informações e guiar os nossos movimentos. Para começar a compreender como a visão pode participar do processo de interação, neste capítulo descrevemos brevemente a estrutura do olho humano (seção 2.1), e então discutimos as diferentes técnicas para rastrear os movimentos do olho descritas na literatura, comparando suas vantagens e desvantagens em relação ao uso em sistemas interativos (seção 2.2).

2.1 Estrutura do olho humano

A parte externamente visível do olho é composta pela esclera (a parte branca), pela íris (a parte colorida) e pela pupila. A íris é protegida e coberta pela córnea, uma membrana transparente. A pupila é a abertura circular no centro da íris que controla a entrada de luz através do aumento ou diminuição do seu tamanho. Atrás da pupila localiza-se o cristalino, uma lente transparente que capta a luminosidade, focando a imagem observada no fundo do olho, onde encontra-se a retina (Figura 2.1).

A retina é responsável por converter a luz em impulsos nervosos que possam ser posteriormente processados. Ela tem dois tipos de células fotossensíveis: os *cones* e os *bastonetes*. Os bastonetes compõem cerca de 94% desse conjunto, e são células que tem alta sensibilidade à luz, permitindo a visão mesmo em ambientes com pouca luminosidade, porém não têm a capacidade de distinguir diferentes cores. Já os cones respondem somente sob uma maior

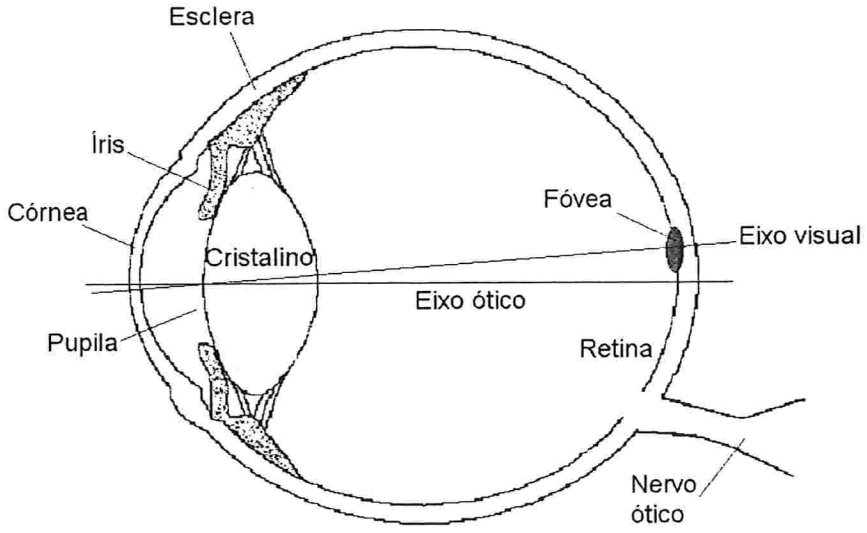


Figura 2.1: Estrutura do olho humano

intensidade luminosa; diferentes cones são sensíveis a diferentes comprimentos de onda da luz, permitindo assim a percepção das cores. [Wik05, GEN95, Lan90].

Os cones estão concentrados no centro da retina; nessa área há uma depressão, chamada *fóvea*, constituída completamente por cones, e responsável pela visão central detalhada. A fóvea não se localiza exatamente no eixo ótico do olho, definido como a semi-reta que passa pelo centro do globo ocular e o centro da pupila; o eixo que realmente define para onde está direcionada a atenção visual do indivíduo é definido pela semi-reta com origem no centro da fóvea e que passa pelo centro da pupila (Figura 2.1). A fóvea cobre uma ângulo visual de aproximadamente um grau.

A cada momento, o objeto que recebe atenção visual deve ser focado na fóvea; este evento é chamado de *fixação*, e dura entre 200 e 600 milissegundos [Jac90]; o olho muda de posição entre uma fixação e outra através de um movimento denominado *sacada*. A sacada é um movimento voluntário, que dura entre 30 e 120 milissegundos, porém sua natureza é dita *balística*, ou seja, uma vez iniciado não é possível mudar sua direção ou a posição final [GEN95]. Mesmo durante uma fixação, o globo ocular não permanece estático o tempo todo. Ele sofre pequenas mudanças de direção, corrigidas por movimentos de ajuste, chamados *movimentos microssacádicos* [Jac95]. Além disso, o globo ocular pode realizar outros mo-

vimentos, que acontecem para o ajuste do foco visual a objetos em movimento, ou a largos movimentos da cabeça [BG90, YS75].

2.2 Técnicas de rastreamento do olhar

Um conjunto de diferentes técnicas desenvolvidas ao longo do século XX podem ser classificadas como técnicas de rastreamento do olhar [YS75]. Inicialmente, tais técnicas foram empregadas nos campos da psicologia experimental, da oftalmologia, da neurologia e áreas correlatas, como ferramenta para compreender como a busca visual participa de processos cognitivos, e também no estudo e detecção de anormalidades nas características motoras do olho.

O contínuo aumento da capacidade de processamento e diminuição de custo dos microprocessadores, aliado à ampla disponibilidade de dispositivos que permitem a captura de dados para o processo de rastreamento de forma não intrusiva (por exemplo, câmeras de vídeo), permitiu o desenvolvimento de técnicas de rastreamento do olhar razoavelmente robustas e que operam em tempo real.

Na literatura encontramos vários trabalhos [MM05, Duc02, GEN95, YS75] que descrevem em detalhes as características e o funcionamento de diferentes técnicas de rastreamento do olhar; nosso objetivo não é repetir tal descrição, mas apresentar um panorama geral das diferentes alternativas existentes que nos permita avaliar o seu uso em sistemas interativos.

As técnicas de rastreamento do olhar podem ser agrupadas em duas categorias:

intrusivas: requerem o contato físico de algum tipo de instrumento com o indivíduo;

não-intrusivas: são baseadas na obtenção de imagens de vídeo do olho, da cabeça ou de ambos, e posterior aplicação de algoritmos de visão computacional para estimar a posição observada pelo usuário.

Nas seções seguintes descreveremos e comentaremos implementações de rastreadores de olhar que pertencem a essas duas categorias.

2.2.1 Rastreamento do olhar intrusivo

Young e Sheena [YS75] descrevem algumas técnicas de rastreamento do olhar que exigem o contato de instrumentos de medição com o indivíduo. Um exemplo é a técnica conhecida como *eletrooculografia*, que se baseia em pequenas diferenças de potencial elétrico existentes entre a córnea e a retina. Essa técnica exige a colocação de eletrodos ao redor dos olhos para a medição, e é sensível à variações na luminosidade ambiente: um ambiente escuro pode induzir a diminuição das diferenças de potencial, impossibilitando a medição. Em boas condições, a precisão do método é de 1,5 a 2 graus do ângulo visual. Os cuidados necessários com a aparelhagem de medição e a necessidade do uso dos eletrodos tornam problemática o uso dessa técnica fora do campo médico [GEN95].

Robinson [Rob63] descreve uma técnica que utiliza uma lente de contato especial que reage à campos eletromagnéticos de alta freqüência gerados ao redor do indivíduo. A lente é fixada através de uma pequena sucção no olho do indivíduo, permitindo assim determinar a linha de visão com uma boa precisão, cerca de $0,08$ graus do ângulo visual. Porém, essa é uma das técnicas mais invasivas, além da desvantagem do desconforto gerado pelo uso de uma lente especial, e de questões de saúde a respeito da geração de campos de alta freqüência ao redor do indivíduo [GEN95].

Apesar de poderem trazer uma medição mais precisa, as técnicas intrusivas de rastreamento do olhar têm em comum o desconforto e a pouca praticidade causados pelo contato com os instrumentos de medição. Esses fatores tornam sua aplicação ao controle de interfaces praticamente impeditivo [Jac90, GEN95], restringindo sua aplicação à execução de medições experimentais.

2.2.2 Rastreamento do olhar não-intrusivo

A partir da obtenção de imagens de vídeo, as técnicas não intrusivas estimam a posição observada pelo indivíduo através do reconhecimento de alguma característica notável do olho. Uma delas é a fronteira entre a esclera e a íris; como, em condições normais de saúde ocular, a esclera é branca e a íris é de uma cor mais escura, essa fronteira pode ser detectada e rastreada facilmente. Mas na maioria dos indivíduos a íris é parcialmente coberta pelas

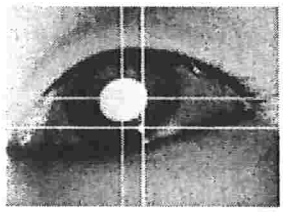


Figura 2.2: Pupila iluminada; as linhas indicam o centro da pupila e do brilho ocular

pálpebras, limitando bastante a precisão vertical do rastreamento [GEN95, YS75].

Várias técnicas [OMY02, MKAF00, MSS91] são baseadas na identificação da pupila, que é menos sujeita a variabilidades individuais do que a fronteira entre a esclera e a íris. O contraste entre a pupila e a íris é, por outro lado, muito menor. Uma estratégia para minimizar esse problema é usar uma fonte de luz alinhada com o eixo ótico da câmera, de forma a gerar uma imagem iluminada da pupila, como o efeito de "olho vermelho" que se nota em fotografias tiradas com *flash* disparado em alinhamento muito próximo ao eixo ótico (Figura 2.2). Esse efeito é devido à reflexão da luz pelo fundo do olho. Com uma câmera adequada, é possível usar uma fonte de luz com comprimento de onda próximo ao infravermelho, que por não ser visível pelo olho humano, não incomoda o usuário.

Uma vez detectada a posição da pupila, é desejável que seus movimentos possam ser estimados em relação à outra característica do olho, cuja posição possa ser considerada fixa em relação a algum referencial. Uma possibilidade usual é a detecção de uma *imagem de Purkinje* (Figura 2.3), que é uma das reflexões de luz que aparecem em diferentes camadas do olho. A primeira imagem de Purkinje é a reflexão da luz pela córnea, e é a mais clara e simples de ser identificada, seguida pela quarta imagem, que é a reflexão da luz pela parte de trás do cristalino [YS75].

Baluja e Pomerlau [BP94] desenvolveram uma técnica alternativa que não tenta identificar nenhuma característica do olho em particular, mas usa uma rede neural para identificar a posição observada. O usuário observa um cursor em movimento na tela por cerca de três minutos, e durante esse tempo são obtidas imagens de um ângulo mais aberto, captando toda a cabeça do usuário. A partir dessas imagens são segmentadas imagens do olho, que então são usadas juntamente com as respectivas posições do cursor na tela como dados de treinamento para a rede neural. A técnica permite algum movimento da cabeça, porém sua

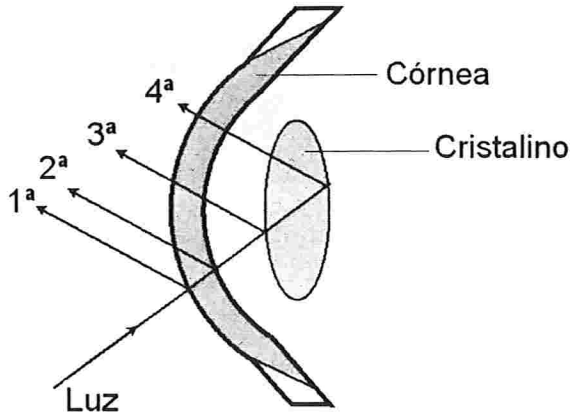


Figura 2.3: Imagens de Purkinje

precisão é pior do que a obtida por outras técnicas, ficando em torno de 2 graus do ângulo visual.

Independentemente da característica do olho que é reconhecida e rastreada, há a necessidade de se mapear a posição de tal característica na imagem de vídeo em relação à posição da linha de visão do indivíduo. Para tal, as técnicas de rastreamento não-intrusivas exigem a execução de algum procedimento de calibragem. O procedimento usual consiste em solicitar que o usuário olhe para alguns pontos na tela, enquanto efetua-se a correspondência desses pontos com os sucessivos posicionamentos da característica do olho na imagem. Então, os dados gerados posteriormente pelo rastreamento são ajustados a partir dos parâmetros calculados no processo de calibragem. Morimoto *et al.* [MKAF00] descrevem um cálculo da posição observada baseado em uma transformação de segunda ordem. Ohno *et al.* [OMY02] utilizam modelos geométricos do globo ocular associados a propriedades de refração da luz para calcular a posição observada.

Como a posição do olho em relação à posição observada varia com a movimentação da cabeça, outra limitação exigida é que a cabeça do indivíduo permaneça estática durante e após o processo de calibragem, em geral com o uso de um apoio para o queixo. Algumas adaptações propostas permitem porém alguma movimentação da cabeça; por exemplo Morimoto *et al.* [MKAF00] implementam um servo-mecanismo na câmera que reorienta a captura de imagens de acordo com pequenas mudanças de posição da cabeça.

As técnicas de rastreamento do olhar não-intrusivas têm em comum algumas limitações: os usuários tem que passar por um procedimento de calibragem cada vez que usam o sistema, além de não poderem mexer livremente a cabeça. Porém, mesmo com essas limitações, o desconforto ainda é menor do que o causado pelas técnicas intrusivas, e por isso rastreadores de olhar baseados na captura de imagens em vídeo têm sido usados com freqüência no desenvolvimento de sistemas interativos baseados em rastreamento do olhar. As técnicas de interação expostas na literatura que utilizam-se de rastreamento do olhar devem- contornar as limitações inerentes à tecnologia atual de rastreamento, como veremos no próximo capítulo.

Capítulo 3

Técnicas de interação baseadas no olhar

Uma *técnica de interação* é uma forma de usar um dispositivo físico de entrada para executar uma tarefa em um diálogo entre seres humanos e computadores [Jac90]. Com o desenvolvimento de técnicas de rastreamento do olhar razoavelmente robustas, não-intrusivas e capazes de fornecer dados em tempo real, surgiu o desafio: como o rastreamento do olhar pode participar da construção de uma técnica de interação?

O projeto de técnicas de interação que utilizam sistemas de rastreamento do olhar deve levar em consideração uma série de fatores inerentes tanto à estrutura fisiológica do processo de visão humana quanto à tecnologia empregada no rastreamento do olhar. Assim, na seção 3.1 introduzimos os problemas mais comuns que surgem a partir da aplicação de técnicas de rastreamento do olhar a interfaces, e propostas descritas na literatura para superá-los. Além disso, expomos os conceitos de interfaces prestativas e multimodalidade, amplamente discutidos em IHC, e onde se insere o rastreamento do olhar nessa discussão. Na seção 3.2 descrevemos e comentamos a implementação das propostas de algumas técnicas de interação que incorporam o rastreamento do olhar, e na seção 3.3 discutimos as vantagens e desvantagens das técnicas descritas na seção 3.2.

3.1 Conceitos

3.1.1 Direção do olhar

Uma característica da estrutura do funcionamento da visão humana impõe uma restrição fundamental à precisão das técnicas de rastreamento do olhar. Conforme citamos na seção 2.1, a fóvea é a área de maior acuidade visual da retina, e tem um área correspondente a cerca de 1 grau do ângulo visual. Não é possível saber qual o ponto exato focado pelo olho dentro desse intervalo de 1 grau; podemos considerar, como uma regra prática, que essa medida corresponde aproximadamente ao tamanho do polegar visto por um indivíduo com seu braço estendido. Além disso, os movimentos microssacádicos realizados inconscientemente durante as fixações criam um potencial fator de introdução de ruído nos dados gerados pelo rastreador.

3.1.2 Fixações

Um rastreador de olhar que opere em tempo real fornece estimativas de pontos observados pelo usuário em curtos intervalos de tempo. Suponha um sistema não-intrusivo de rastreamento do olhar que tenha capacidade de processar 30 quadros de vídeo por segundo; esse sistema forneceria a estimativa de um ponto observado pelo usuário a cada 33 milissegundos, aproximadamente. Esses dados devem passar por um processo de filtragem para obter informações semanticamente relevantes sobre o processo visual. Em particular, os dados provenientes do rastreador devem ser agrupados em unidades que identifiquem as fixações e as sacadas. Este é o primeiro passo para obter informação sobre o ponto de interesse do usuário.

A estratégia mais simples para implementar essa filtragem é o uso de um limiar espacial e outro temporal – somente pontos próximos entre si, e que ocorram dentro de um intervalo mínimo de tempo são agrupados e considerados como componentes de uma mesma fixação [Jac90, SJ00, ZMI99]. Estes valores são definidos a partir do conhecimento prévio da duração média dos eventos do processo visual, geralmente obtido através de testes empíricos. Uma estratégia mais sofisticada é apresentada por Salvucci e Goldberg em [Sal99a]; essa estratégia é baseada no uso de uma cadeia de Markov oculta (HMM - *Hidden Markov Model*) com dois estados: *fixação* e *sacada*. Essa modelagem permite a reestimação dos parâmetros

correspondentes aos limiares temporal e espacial. Em tese, essa reestimação permite o ajuste fino dos parâmetros, de forma a obter uma melhor detecção das fixações. Porém esse processo é computacionalmente mais complexo, e atualmente só é usado para analise de dados *a posteriori* [SG00].

3.1.3 Seleção e o problema do "Toque de Midas"

Uma aplicação comum do rastreamento do olhar às interfaces já existentes é usar o olhar para executar tarefas de apontamento e seleção. Essa estratégia é justificada por algumas discussões encontradas na literatura. Glenstrup e Engell-Nielsen [GEN95] argumentam sobre numerosas evidências sobre a relação entre o interesse dos indivíduos e o que eles estão olhando. Card, Moran e Newell [CMN80] apresentam o KLM (*Keystroke-Level Model*), um modelo cujo objetivo é prever o tempo T gasto para a execução de tarefas simples. A descrição dessas tarefas deve ser detalhada até o nível onde seja possível descrever a necessidade de ações como pressionar teclas ou movimentar o mouse. O KLM modela o tempo necessário para executar uma tarefa de apontamento da seguinte forma:

$$T = T_{pm} + T_v + T_{mt} + T_r \tag{3.1}$$

Nele, a subtarefa motora (isto é, efetivamente mover o dispositivo para a posição correta), que leva tempo T_{mt}, é precedida pelo processo de preparação mental para a execução da tarefa, que leva tempo T_{pm}, e por uma tarefa visual (isto é, a busca visual do alvo em questão) que leva tempo T_v. T_r é o tempo de resposta do sistema.

Para efeito de discussão e modelagens elementares, esse modelo é intencionalmente simplificado. Ele não leva em consideração de forma explícita que, por exemplo, usuários experientes podem necessitar de um menor tempo T_{pm} de preparação mental, ou que em alguns contextos a tarefa motora T_{mt} pode iniciar-se enquanto a busca visual ainda acontece. Essa última hipótese em particular é validada por Hornof e Kieras [HK99] em sua análise dos dados de um experimento envolvendo o apontamento e seleção de itens em menus. Apesar dessa simplificação, o KLM traz uma estimativa razoável e justifica os esforços no sentido de aplicar o rastreamento do olhar para a execução de tarefas de apontamento.

Porém, o mapeamento direto do olhar (mais especificamente, das fixações) a um comando de seleção do sistema cria o problema identificado por Jacob [Jac90] e chamado de *"Toque de Midas"* – uma seleção pode ser ativada em qualquer posição da tela observada pelo usuário, tenha ele a intenção de fazê-la ou não. Após a filtragem dos dados do rastreador de olhar, um desafio no projeto de uma técnica de interação é evitar o problema do Toque de Midas, implementando mecanismos para que o usuário indique quando deseja executar um comando de seleção. Uma primeira solução foi a inclusão de uma latência no tempo de resposta – a seleção de um objeto é efetuada somente após um intervalo de tempo. Alternativamente, a seleção pode ser ativada manualmente pelo pressionamento de um botão pelo usuário. O MAGIC Pointing [ZMI99] traz uma abordagem alternativa que incorpora o rastreamento do olhar ao apontamento tradicional e será descrito mais detalhadamente na seção 3.2.4.

3.1.4 Modelagem do contexto e interfaces prestativas

O estudo da informação gerada por rastreadores de olhar tem sido importante para o campo da psicologia cognitiva, pois ela fornece evidência sobre a forma com que indivíduos usam a exploração visual para resolver problemas, quanto tempo gastam para analisar essa informação e quando precisam voltar à informação previamente acessada [Sal99b].

Ao incorporar o olhar e outras informações sobre o estado do usuário no processo de interação com sistemas computacionais cria-se um potencial para que as interfaces, dispondo de mais informação sobre o estado da atenção do usuário, não fiquem mais o tempo todo passivas à espera de um comando, mas se tornem *prestativas*[1], ou seja, possam se adaptar dinamicamente às necessidades do usuário a cada instante.

Vertegaal [Ver02], em sua discussão sobre interfaces prestativas, identifica que para atingir esse objetivo tais interfaces constroem um modelo do estado passado, presente e futuro da atenção do usuário, levando também em consideração a disponibilidade de recursos do sistema – ou seja, um modelo composto por informações que fazem parte do *contexto* da interação. O conceito de interfaces prestativas vem ampliar a proposta de interfaces perceptivas de Turk [Tur98] – além da perceber o estado do usuário, a interface também modela as ações possíveis

[1]Nossa tradução do inglês *attentive interfaces*

e seleciona dentre elas a mais apropriada a cada momento.

Recentemente, sistemas não-intrusivos de rastreamento do olhar têm sido empregados no desenvolvimento de algoritmos de *inferência*[2] de movimentos do olhar. Um algoritmo de inferência toma como entrada uma seqüência de ações do usuário definidas em um certo protocolo[3] e devolve uma estimativa, baseada em um modelo previamente construído, de qual seria a intenção do usuário ao tomar essas ações.

O projeto de técnicas de interação baseadas em rastreamento do olhar pode se beneficiar com o uso de algoritmos de inferência [Sal99a, SA00, Edw98], uma vez que seja possível definir um modelo das tarefas que possam ser executadas na interface. No caso da aplicação do rastreamento do olhar a tarefas de apontamento, um algoritmo de interpretação pode ajudar a definir qual objeto o usuário tem a real intenção de selecionar, compensando as imprecisões inerentes ao olho humano (veja seção 3.1.1) e ruído gerado pelo processo de rastreamento.

3.2 Exemplos de técnicas de interação baseadas no olhar

Vamos agora descrever algumas técnicas de interação descritas na literatura. O objetivo dessa seção não é enumerar exaustivamente as aplicações do rastreamento do olhar a interfaces, mas ilustrar através de alguns dos trabalhos mais relevantes na área os conceitos definidos na seção anterior.

A partir da forma como o rastreamento do olhar é empregado podemos tentar organizar as suas aplicações em categorias. Duchowski [Duc02] as classifica em dois grandes grupos: *de diagnóstico* e *interativas*. Aplicações de diagnóstico são aquelas onde os dados provenientes do processo de rastreamento são armazenados e processados posteriormente para o estudo dos padrões de atenção do usuário em resposta a um determinado estímulo. Tais aplicações tipicamente não mudam seu estado (em particular, o estímulo apresentado) de acordo com a

[2]Nossa tradução do inglês *tracing algorithms*, usada aqui para evitar a confusão com a expressão *rastreamento* do olhar (*eye tracking*), usada para se referir ao processo de obter o ponto observado pelo usuário ou sua linha de visão

[3]A idéia de protocolo implica a filtragem semântica dos dados do dispositivo de entrada rastreador das ações; no caso do rastreamento do olhar aplicado a interfaces teríamos como componentes desse protocolo as fixações e as sacadas

mudança da atenção visual do usuário.

O segundo grupo, das chamadas aplicações *interativas*, é o foco de interesse dos exemplos desta seção. Tais aplicações reagem à posição observada pelo usuário, e a forma como reagem as divide em duas subcategorias:

seletivas: são as aplicações que usam a informação do ponto observado pelo usuário para selecionar alvos da interface, de forma análoga a um dispositivo de apontamento como o mouse. Como exemplos de sistemas desta categoria podemos citar o protótipo descrito por Jacob (seção 3.2.3) e o MAGIC Pointing (seção 3.2.4).

de contingência do olhar: são as aplicações que usam a informação do ponto observado pelo usuário para redistribuir a informação exibida de forma a otimizar a renderização de telas complexas ou o uso de um canal de comunicação com largura de banda limitada. Podem ser classificadas nesta categoria o Mundo das Janelas de Bolt (seção 3.2.1), o ambiente interativo do Pequeno Príncipe de Starker e Bolt (seção 3.2.2) e o sistema de renderização de simulações de O'Sullivan, Dingliana e Howlett (seção 3.2.5).

3.2.1 *Mundo das janelas*

Uma das aplicações pioneiras do rastreamento do olhar é o *mundo das janelas* descrito por Bolt [Bol81]. O mundo das janelas constitui-se de um *videowall* que exibe diferentes seqüências de vídeo e, de acordo com a atenção visual recebida, as janelas correspondentes às seqüências aumentam ou diminuem; o som da seqüência que recebe maior atenção visual também tem seu volume aumentado. É uma das primeiras tentativas de inferir a intenção do usuário através do olhar em um sistema interativo que se tem notícia.

3.2.2 *Pequeno príncipe*

Glenstrup e Engell-Nielsen [GEN95] descrevem em seu trabalho um sistema desenvolvido por Starker e Bolt (1990) que conta, em uma versão animada, a história "O Pequeno Príncipe", de Antoine de Saint Exupery. A história é contada em um ambiente (o pequeno planeta habitado pelo príncipe) que contém vários objetos, tais como vulcões, escadas e flores.

O espectador é monitorado por um rastreador de olhar baseado na reflexão de luz pela córnea; quando é detectada a atenção do espectador sobre algum objeto do ambiente, o sistema exibe maiores informações sobre o objeto. Essa detecção é feita através da contagem do número de fixações feitas sobre o objeto associada com o tempo de duração das fixações. É feita também uma tentativa rudimentar de inferência do contexto: os objetos são agrupados em categorias, e quando se detecta que a atenção está distribuída de forma aproximadamente igual por vários objetos de uma mesma categoria, o sistema exibe informações sobre a categoria de objetos.

3.2.3 Seleção por latência

Jacob [Jac90] descreve os protótipos de algumas técnicas de interação usando rastreadores de olhar. Seu trabalho levanta a questão do "Toque de Midas" e implementa duas alternativas para a seleção: o usuário pode pressionar um botão indicando a intenção de selecionar o objeto observado ou pode aguardar um tempo de latência razoavelmente curto (definido entre 150 e 250 milissegundos). Ambas as alternativas são oferecidas em paralelo, e é reportado que em experiências preliminares o tempo de latência se mostrou mais conveniente.

O protótipo baseia-se em uma tela onde são apresentados vários navios do lado direito, e uma área para exibição de texto do lado esquerdo (Figura 3.1). Quando um navio é selecionado da maneira descrita acima, informações sobre o mesmo são exibidas. Foi implementada também a operação de "arrastar-e-soltar" navios: uma vez selecionado um navio, o usuário pressiona um botão e olha para o lugar onde deseja posicionar o navio, soltando então o botão. A seleção usando o olhar também foi empregada no protótipo para a seleção de menus do tipo "pull-down"; nesse caso a estratégia do tempo de latência curto não se mostrou eficiente – o autor especula que isso acontece porque o tempo necessário para a leitura de itens do menu que não são familiares ao usuário é falsamente interpretado como o tempo de latência para a seleção. O aumento do tempo de latência também parece não ser razoável, pois gera a necessidade de manter a atenção visual sobre um item de menu durante um intervalo de tempo pouco natural.

Posteriormente, esse protótipo foi avaliado com maior rigor por Sibert e Jacob [SJ00]

através de experimentos com usuários. Uma tarefa simples, que consistia em selecionar círculos exibidos na tela, foi apresentada aos usuários, que a executaram usando o mouse e a técnica de interação baseada em seleção por latência. Os autores verificaram que, em média, a tarefa foi executada mais rapidamente usando a seleção por latência. A latência utilizada na configuração do experimento foi de 150 milissegundos.

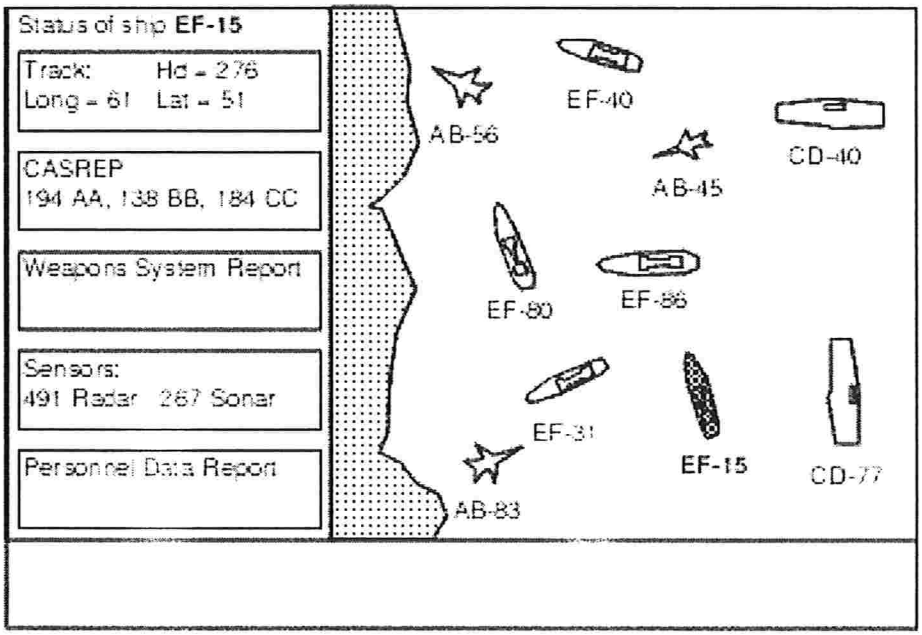


Figura 3.1: Seleção usando tempo de latência (conforme exibido em [SJ00])

3.2.4 MAGIC Pointing

O MAGIC Pointing [ZMI99, Zha03] de Zhai, Morimoto e Ihde trouxe uma nova abordagem à forma de executar tarefas de apontamento auxiliadas por rastreamento do olhar. Levando em consideração as limitações na precisão causadas pelo processo de rastreamento e a calibragem do sistema, propõem uma técnica de interação que combina o rastreamento do olhar com um dispositivo de apontamento convencional, como um mouse ou um *track-ball*. A posição da tela indicada pelo rastreamento do olhar é considerada uma estimativa da posição observada pelo usuário, sujeita a erro em algum grau. Nesse momento o usuário ajusta a posição do cursor através do controle motor do dispositivo tradicional, obtendo

maior precisão.

O cursor acompanha o olhar quando a atenção visual se volta para um alvo distante da posição atual do cursor. Duas estratégias diferentes de reposicionamento foram implementadas: na primeira (chamada estratégia *conservadora)* o cursor se move quando o usuário indica a intenção de fazê-lo, executando um movimento com o dispositivo tradicional; na segunda (chamada estratégia *liberal*) o cursor acompanha o olhar sem que seja necessário executar esse movimento.

Posteriormente, Wang, Zhai e Su [WZS01] desenvolveram uma aplicação do MAGIC Pointing a um domínio específico: a digitação de ideogramas do idioma chinês através do *pinyin*, uma representação fonética usando caracteres do alfabeto ocidental. O problema surge com a grande quantidade de ideogramas que correspondem a uma mesma representação fonética, exigindo a eliminação de ambiguidades de alguma forma. Nesse sistema isso é feito através da seleção de botões que exibem os diferentes caracteres correspondentes a uma mesma representação fonética, que acabou de ser digitada pelo usuário. A seleção dos botões é feita usando o MAGIC Pointing, e é empregada uma estratégia para atenuar o efeito da precisão limitada do rastreamento do olhar: os botões não são dispostos em filas ou colunas, mas em um formato de 'W' (Figura 3.2), de forma a maximizar a distância entre os alvos.

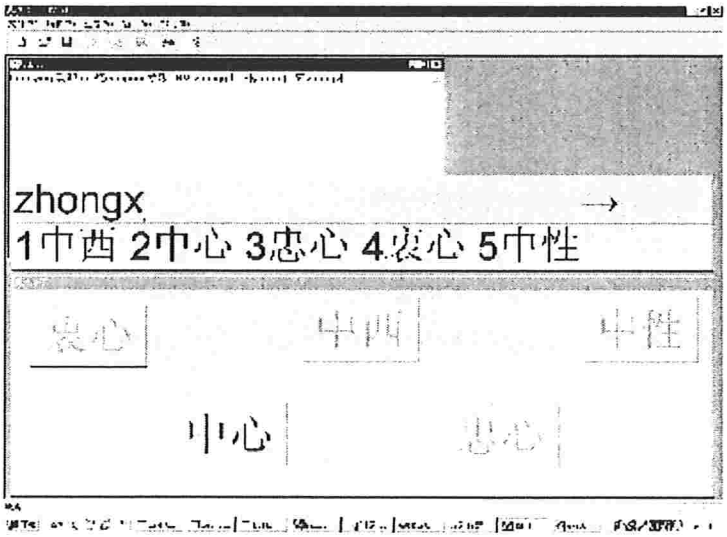


Figura 3.2: Digitação de ideogramas do idioma chinês através do *pinyin* [WZS01]

3.2.5 Renderização de simulações em tempo real

O'Sullivan, Dingliana e Howlett [ODH02] descrevem um sistema onde o rastreamento do olhar é usado para otimizar o uso de recursos computacionais aplicados à renderização de colisões entre objetos complexos.

Renderizar objetos em tempo real para aplicações como jogos e sistemas de realidade virtual (VR) é uma tarefa computacionalmente complexa, e muita vezes é necessário renderizar alguns objetos com um nível de detalhamento (LOD - *level of detail*) menor para manter uma taxa de redesenho da tela aceitável. Outra estratégia para atingir esse objetivo é manter modelos em vários níveis de detalhamento para o tratamento e renderização de colisões entre objetos. Em [ODH02] os autores utilizam o rastreamento do olhar para determinar a região da tela onde está focada a atenção visual do usuário, e a renderização de colisões nessa região terá alocada uma parcela maior de tempo dos recursos computacionais.

Um experimento foi conduzido, onde usuários classificaram seqüências de animações de colisões entre dois objetos de acordo com a qualidade da renderização, como "boas" ou "más", de acordo com critérios aprendidos em uma sessão de treinamento onde foram expostos a animações com maior ou menor nível de detalhamento. Os resultados mostraram que, quando as seqüências onde o nível de detalhamento das colisões foi modificado dinamicamente de acordo com as informações do rastreamento do olhar, a qualidade da renderização percebida pelos usuários foi maior.

3.2.6 Modelagem do contexto

Como comentado na seção 3.1.4, um método de compensar a imprecisão do processo de rastreamento que tem sido empregado recentemente é incorporar mais informação sobre o contexto do diálogo entre o usuário e o computador. Salvucci [SA00] descreve uma interface, denominada IGO (*Intelligent Gaze-added Operated Interface*) que simula uma área de trabalho comum de um ambiente WIMP (Windows, Menus, Icons and Pointing), mas que pode ser operada através do olhar. A seleção é efetuada através do pressionamento de um botão no teclado, de maneira semelhante ao descrito em [Jac90]; porém aqui a funcionalidade do apontamento tradicional é mantida, de forma que o apontamento pelo olhar seja uma forma

adicional de interagir com o sistema. As fixações identificadas pelo sistema de rastreamento são mapeadas nos objetos através de um modelo probabilístico: dada uma fixação g, o item da interface i_{best} no qual a fixação será mapeada é aquele que maximiza a probabilidade $P(i|g)$, onde i representa o evento "selecionar o objeto i". Daí temos:

$$\begin{aligned} i_{best} &= \text{argmax}_{i \in I} \left(P(i|g)\right) \\ &= \text{argmax}_{i \in I} \left(\frac{P(g|i)P(i)}{P(g)}\right) \\ &= \text{argmax}_{i \in I} \left(P(g|i)P(i)\right) \end{aligned}$$

$P(g|i)$, a probabilidade de uma fixação (x, y) dada a intenção de se selecionar o item i é modelada através de uma distribuição gaussiana bidimensional centrada no item. No momento do cálculo de $P(i)$ é incorporada a informação de contexto da tarefa em execução. Cada um dos itens da interface recebe uma pontuação (posteriormente normalizada na escala de probabilidades) de acordo com a última tarefa executada, ou de acordo com algum estado da interface. Por exemplo, o botão "X" de fechamento de uma janela tem sua pontuação aumentada após a seleção de algum item na janela, considerando que o usuário tipicamente não fecha uma janela sem efetuar pelo menos uma seleção nela. O menu que contém o comando *Esvaziar lixeira* tem sua pontuação aumentada quando há itens na lixeira, e assim por diante.

Em [Sal99a] é desenvolvida uma modelagem mais complexa do contexto de uma aplicação em que o usuário digita palavras olhando para as letras de um teclado exibido no vídeo. Para um dado conjunto de palavras pré-definido, é construída uma gramática (Tabela 3.1) que em suas regras de produção modela os passos cognitivos que podem ser tomados a partir de um dado estado. No exemplo é exibida uma gramática completa que modela os passos cognitivos para a digitação das palavras *rat*, *trap* e *part*.

Há três possibilidades para a construção da gramática, enumeradas em ordem crescente do grau de informação seqüencial que possuem:

- **gramática de primeira ordem**: representa transições de letra para letra;
- **gramática de segunda ordem**: representa transições de pares de letras para letras;

- **gramática completa**: representa cada palavra separadamente.

start	→	**word** rat
start	→	**word** trap
start	→	**word** part
rat	→	R A T end
trap	→	T R A P end
part	→	P A R T end
end	→	**out**

Tabela 3.1: Exemplo de gramática completa: passos cognitivos para digitação de palavras

O usuário inicia a tarefa selecionando o botão **word**, que mostra a palavra a ser digitada, e encerra a digitação selecionando o botão **out**, representados por seus nomes na gramática. A partir da gramática é construída uma cadeia de Markov oculta que segue a estrutura da gramática dos passos cognitivos e tem como subunidades fundamentais de construção cadeias de Markov ocultas para a identificação das fixações em cada alvo (no caso, em cada letra do teclado virtual). Uma vez montado esse modelo, um protocolo de fixações e sacadas obtido a partir do rastreador de olhar é mapeado na seqüência mais provável de estados usando o Algoritmo de Viterbi [Rab89].

A abordagem probabilística, baseada na atribuição de uma pontuação variável a cada item da interface, tem uma estrutura e uma implementação mais simples, além de possibilitar de forma imediata a modelagem da mudança de estados da interface. Ela exige, por outro lado, a codificação manual prévia das regras de pontuação que geram as probabilidades de seleção de itens. A abordagem baseada na construção de uma gramática apresenta um modelo mais preciso dos passos cognitivos, mas também precisa ser previamente codificada. Uma vantagem dessa abordagem é a maior similaridade entre a estrutura da gramática e a estrutura de outras modelagens dos passos necessários para cumprir uma tarefa, como o GOMS [CMN83]. Apesar de não terem sido feitas comparações entre as duas abordagens na modelagem de uma mesma tarefa, a avaliação experimental de ambas inclui o cálculo da taxa de acerto da inferência. Segundo o autor, na abordagem probabilística foi possível inferir corretamente a intenção do usuário em 83% das execuções de tarefas pré-definidas em um experimento com o IGO; já na abordagem baseada na construção de uma gramática,

com uma gramática completa para 1000 palavras como a da figura 3.1, o sistema foi capaz de inferir corretamente a palavra "digitada" com o olhar em 92% das situações. A construção de uma gramática dos passos de uma tarefa exige um bom conhecimento prévio do domínio da aplicação, e sua decodificação eficiente em tempo real necessita de uma representação igualmente eficiente das estruturas de dados e dos algoritmos de busca, porém esse esforço adicional parece resultar em uma inferência mais eficaz.

3.3 Análise comparativa

Várias técnicas de interação baseadas em rastreamento do olhar foram desenvolvidas considerando o olhar como uma estimativa direta do objeto de interesse. Em outras palavras, tais técnicas (pertencentes à categoria das aplicações interativas seletivas de Duchowski [Duc02]) foram projetadas considerando que, se o usuário está olhando para um objeto, essa informação é suficiente para considerar que ele deseja tomar alguma ação sobre aquele objeto. É o que acontece no sistema interativo do Pequeno Príncipe (seção 3.2.2 e posteriormente no protótipo de Jacob (seção 3.2.3). Mesmo identificando e reconhecendo a existência do problema do "Toque de Midas" e propondo a solução do tempo de latência, o trabalho de Jacob identifica que a estratégia do tempo de latência não é bem sucedida em todos os casos de uso que surgem em um protótipo mais próximo do funcionamento de interfaces reais. Por exemplo, em tarefas que exigem leitura ou exploração visual mais detalhada, o tempo de latência deve ser mantido alto para evitar seleções não intencionais. A deficiência dessas técnicas está em não considerar com a devida importância a própria natureza exploratória do processo da visão, que está em ação a todo momento, mas é mais evidente quando há a necessidade de executar tarefas complexas.

Uma técnica de interação mais recente que ainda usa o olhar diretamente para tarefas de seleção é o IGO de Salvucci [SA00]. Nesse caso a intenção de selecionar é indicada através do pressionamento de uma tecla. Porém, a interface de testes é toda projetada para minimizar a imprecisão do processo de rastreamento – os objetos selecionáveis são bem maiores do que o usual em interfaces WIMP, e a informação de contexto é empregada para eliminar eventuais ambiguidades. De qualquer forma, uma técnica de interação que associa a seleção de objetos

diretamente ao interesse visual acaba por sobrecarregar a visão com uma tarefa não natural, que é usá-la para atuar sobre o ambiente à nossa volta. Apesar dos problemas, essa estratégia parece justificável em casos particulares em que o usuário não poderia interagir com o sistema de outra forma, como o de pessoas que sofrem de limitações dos movimentos, por exemplo devido a tetraplegia ou doenças degenerativas [HJM+89, Edw98].

Em outra linha de desenvolvimento estão as interfaces que empregam o rastreamento do olhar como um canal de entrada passivo, usado para redistribuir dinamicamente a informação exibida (ou aplicações interativas de contingência de olhar [Duc02]). Nessa linha podemos classificar o Mundo das Janelas de Bolt (seção 3.2.1) e mais recentemente as técnicas de renderização de simulações (seção 3.2.5). Essas aplicações do rastreamento do olhar tem menos dificuldades a contornar quanto à precisão do rastreamento ou à tomada de decisão sobre quando o usuário deseja ou não tomar uma ação no objeto que recebe sua atenção visual – uma estimativa menos precisa da região observada já é suficiente para redimensionar a informação exibida [ODH02].

Conhecidos os problemas enfrentados e as características de ambas as linhas de desenvolvimento poderíamos concluir que as aplicações interativas de contingência de olhar tem melhor condição de se adaptar à tecnologia de rastreamento atual e às características do processo de visão. O MAGIC Pointing (seção 3.2.4), entretanto, teve sucesso em implementar uma técnica de interação para a execução de tarefas de apontamento que associa o melhor das duas linhas de desenvolvimento descritas acima: ele inova no sentido de tratar o olhar como um canal de entrada passivo, trazendo uma estimativa da região de interesse do usuário. A atuação continua tendo o componente motor, e diminui-se o esforço cognitivo de explicitamente atuar sobre o ambiente com o olhar.

Capítulo 4

Desenvolvimento de uma interface prestativa baseada no olhar

Como vimos nos capítulos anteriores, o olho é um canal para a percepção do ambiente à nossa volta que recebe informação continuamente durante boa parte de nossas atividades. A velocidade com que ocorre a mudança do foco da atenção visual sugere que, uma vez que seja possível obter informação sobre a atenção visual a cada momento, essa informação possa ser empregada no processo de interação humano-computador de forma a aumentar a velocidade do diálogo. Por outro lado, também vimos que as técnicas de rastreamento do olhar disponíveis apresentam problemas: a precisão de algumas é limitada, enquanto que outras são invasivas demais para o uso cotidiano.

Frente às vantagens potenciais e às deficiências atuais apresentadas pelo rastreamento do olhar, nossa proposta consiste em incorporar a informação sobre a atenção visual ao processo de interação com interfaces do tipo WIMP para a realização de tarefas de apontamento. A nossa implementação e testes buscam identificar as situações em que o rastreamento do olhar pode tornar o processo de interação *melhor* para o usuário – melhor tanto em aspectos objetivos, como a velocidade ou precisão na realização das tarefas, quanto em aspectos subjetivos como a sua satisfação.

Em particular, queremos avaliar a necessidade e a importância de incorporar ao processo de interação informações sobre o olhar e sobre o contexto atual da interação. A combinação

dessas duas fontes de informação, uma sobre o estado do usuário e outra sobre a tarefa em curso, aproxima as técnicas de interação que desenvolvemos do paradigma de interfaces prestativas.

A técnica de interação escolhida como ponto de partida para nosso trabalho é o MAGIC Pointing [ZMI99, Zha03], descrito na seção 3.2.4. Sua estratégia inova ao tratar o olhar como um canal de entrada mais passivo. A escolha do paradigma WIMP justifica-se pois ele é seguido pela maioria dos sistemas computacionais atualmente, permitindo avaliar com maior confiabilidade a eficiência das nossas extensões.

Na seção 4.1 descrevemos a implementação do protótipo do nosso sistema de rastreamento do olhar. A versão inicial desse sistema surgiu a partir da migração para o sistema operacional Linux de um sistema previamente implementado na plataforma Windows. A partir dessa versão aperfeiçoamos algumas funcionalidades do sistema: a capacidade de processamento de imagens foi aumentado de 30 para 60 imagens por segundo, e um novo reconhecedor de fixações e sacadas foi implementado. Na seção 4.2 apresentamos a nossa técnica de interação, desenvolvida a partir do conceito do MAGIC Pointing e que incorpora informação sobre o posicionamento dos objetos selecionáveis a cada momento.

4.1 Sistema de rastreamento do olhar

Nosso sistema de rastreamento do olhar surgiu a partir de uma implementação para Linux do sistema desenvolvido por Morimoto *et al.* [MKAF00]. Em setembro de 2003 foi concluída a primeira versão do rastreador, a partir da adaptação do código proveniente de uma versão anterior desenvolvida para Windows. O porte do sistema para Linux se insere nos planos do LaTIn - Laboratório de Tecnologias de Interação do IME-USP de distribuí-lo sob uma licença de *software* livre.

Nosso rastreador de olhar baseia-se na detecção da pupila e da primeira imagem de Purkinje (que exibimos no esquema da Figura 2.3) através da iluminação do olho por dois conjuntos de LEDs infra-vermelhos (Figura 4.1). O primeiro conjunto é posicionado próximo ao eixo óptico da câmera, de forma a obter uma imagem iluminada da pupila através da reflexão da luz pelo fundo do olho, uma técnica que mencionamos na seção 2.2.2. O segundo

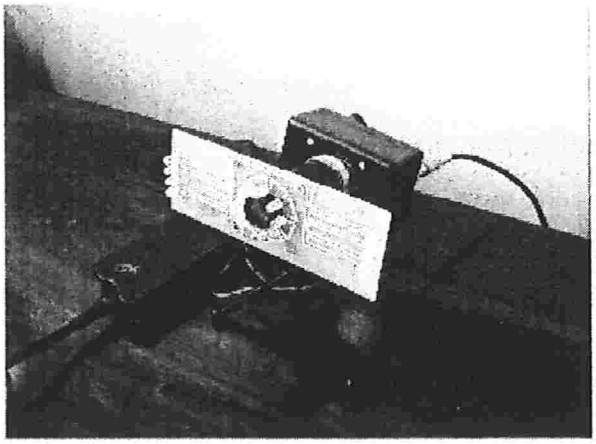
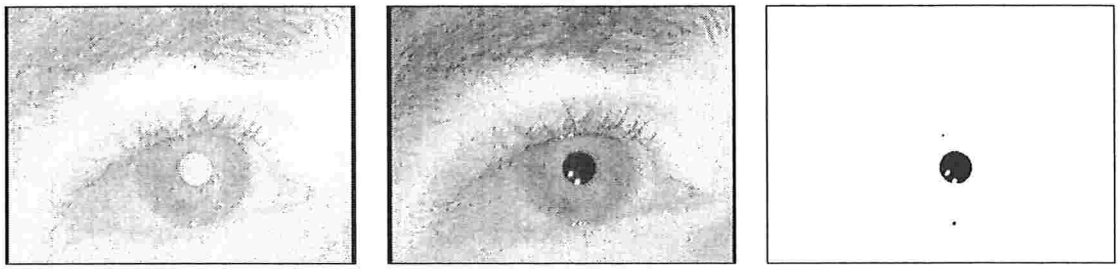


Figura 4.1: Câmera e conjuntos de *leds* infravermelhos



(a) (b) (c)

Figura 4.2: Pupila iluminada (a), pupila escurecida e 1^a imagem de Purkinje $IP1$ (b), e diferença entre as imagens (c)

conjunto, mais afastado do eixo óptico, produz uma imagem escurecida da pupila; note na Figura 4.2b que cada um dos grupos de 4 LEDs que formam o segundo conjunto produzem a primeira imagem de Purkinje $IP1$. Os conjuntos de luzes brilham alternadamente, de forma a obter, em seqüência, imagens do olho com a pupila clara e escura.

A diferença entre essas duas imagens produz uma terceira imagem (Figura 4.2c) onde a segmentação da pupila é obtida através da identificação do maior componente conexo. Na imagem da pupila escurecida, (Figura 4.2b) as imagens de Purkinje são segmentadas através da limiarização da imagem. Assim, é possível obter na imagem os pontos correspondentes ao centro da pupila e ao centro de massa de $IP1$. O próximo passo é obter um mapeamento entre a posição desses pontos na imagem do olho e a posição da tela observada pelo indivíduo; esse é o objetivo do procedimento de calibragem.

Na calibragem, que deve ser efetuada antes do uso do sistema, o usuário é solicitado a olhar

para nove pontos sobre a tela. Para cada ponto da tela, que denominaremos (x_t, y_t), calcula-se na imagem do olho o vetor entre o centro de $IP1$ e o centro da pupila, que denominaremos (x_v, y_v). Define-se então uma transformação de segunda ordem entre (x_t, y_t) e (x_v, y_v) da seguinte forma:

$$\begin{aligned} x_t &= a_0 + a_1 x_v + a_2 y_v + a_3 x_v y_v + a_4 x_v^2 + a_5 y_v^2 \\ y_t &= a_6 + a_7 x_v + a_8 y_v + a_9 x_v y_v + a_{10} x_v^2 + a_{11} y_v^2 \end{aligned}$$

O par de equações acima é gerado para cada um dos 9 pontos, produzindo finalmente um sistema sobredeterminado de 18 equações e 12 incógnitas $a_0, a_1, \ldots, a_{11}$, que são então determinadas através da minimização do erro quadrático. A cada quadro de vídeo capturado pode-se então obter um vetor (x'_v, y'_v) ao qual é aplicada a transformação de segunda ordem com os parâmetros calculados durante a calibragem para obter um ponto (x'_t, y'_t) sobre a tela. Na verdade, aproveitando uma particularidade do processo de captura da imagem do olho, é possível, a partir de uma mesma imagem, obter seqüencialmente dois pontos distintos $(x'_t, y'_t)_1$ e $(x'_t, y'_t)_2$ sobre a tela; na seção 4.1.1 descrevemos essa técnica, que foi posteriormente incorporada ao sistema.

Originalmente, o código do MAGIC e do rastreador de olhar foram desenvolvidos no ambiente Microsoft Visual C++, sendo o rastreador uma aplicação Windows e o MAGIC um módulo compilado como uma DLL (Dynamic Link Library), podendo assim ser usado em conjunto com outros sistemas de rastreamento do olhar.

Boa parte do código em C++ pôde ser migrado de uma plataforma para outra sem maiores dificuldades; porém, algumas decisões de implementação tiveram que ser tomadas no momento de substituir alguns módulos específicos ao sistema operacional Windows. Em particular, três módulos foram completamente substituídos: a captura de imagens, a exibição das imagens processadas e a interface com o usuário.

A captura de imagens usava originalmente a API DirectX, da Microsoft. Quando da migração para o Linux, optamos pela API Video4Linux [Dir04]. Atualmente, essa é a interface mais utilizada para o acesso a dispositivos de vídeo no Linux, por ser fornecida

diretamente pelo *kernel* e permitir de forma transparente o acesso a dispositivos de diferentes fabricantes, desde que os módulos do *kernel* compatíveis com a API VIDEO4LINUX tenham sido corretamente carregados.

Em linhas gerais, um programa que deseje capturar um quadro de vídeo primeiro obtém acesso ao descritor de arquivo que representa o dispositivo de captura no sistema de arquivos (usualmente um caminho como `/dev/video0`); depois configura os parâmetros da captura (profundidade de cores, dimensões do quadro, etc) e, finalmente, efetua uma chamada para indicar ao módulo do dispositivo de captura que comece a escrever os dados do quadro atualmente disponível em um espaço de memória previamente alocado. Esses dados não estão disponíveis imediatamente, ou seja, o programa precisa aguardar até que o módulo indique que o processo de escrita dos dados terminou.

Para que o nosso sistema seja capaz de capturar e processar todos os quadros de vídeo fornecidos pela câmera, seria ideal que a API fornecesse um mecanismo assíncrono de captura[1]; como isso ainda não é possível, simulamos esse modo de operação em nosso sistema através do uso de múltiplas linhas de execução (*threads*) dentro do mesmo processo. As chamadas para a captura de imagens são feitas em uma linha de execução separada. Uma aplicação que deseje utilizar o módulo de captura registra junto a ele uma função de retorno, que receberá um ponteiro para o início da área de memória que armazena os dados referentes a cada quadro de vídeo capturado. Uma segunda linha de execução no módulo de captura faz essa passagem dos dados para a função de retorno da aplicação.

O sistema de rastreamento do olhar exibe, para fins de depuração e controle, a imagem capturada do olho com duas cruzes, uma sobre o centro da pupila e outra sobre o centro de $IP1$ quando o rastreamento está ocorrendo com sucesso (Figura 4.3). Para que a renderização das imagens após o processamento não limitasse o desempenho do sistema utilizamos a extensão MIT Shared Memory do sistema gráfico XFree86 [Inc04], que cria uma área de memória compartilhada entre o sistema gráfico e a aplicação cliente. Dessa forma a transferência dos dados da imagem a ser renderizada é feita de forma mais rápida do que a renderização usual que é fornecida pela XLIB, a API convencional do XFree86.

[1] Apesar da captura assíncrona estar prevista na definição da versão 2 da VIDEO4LINUX, ela ainda não foi implementada até o momento

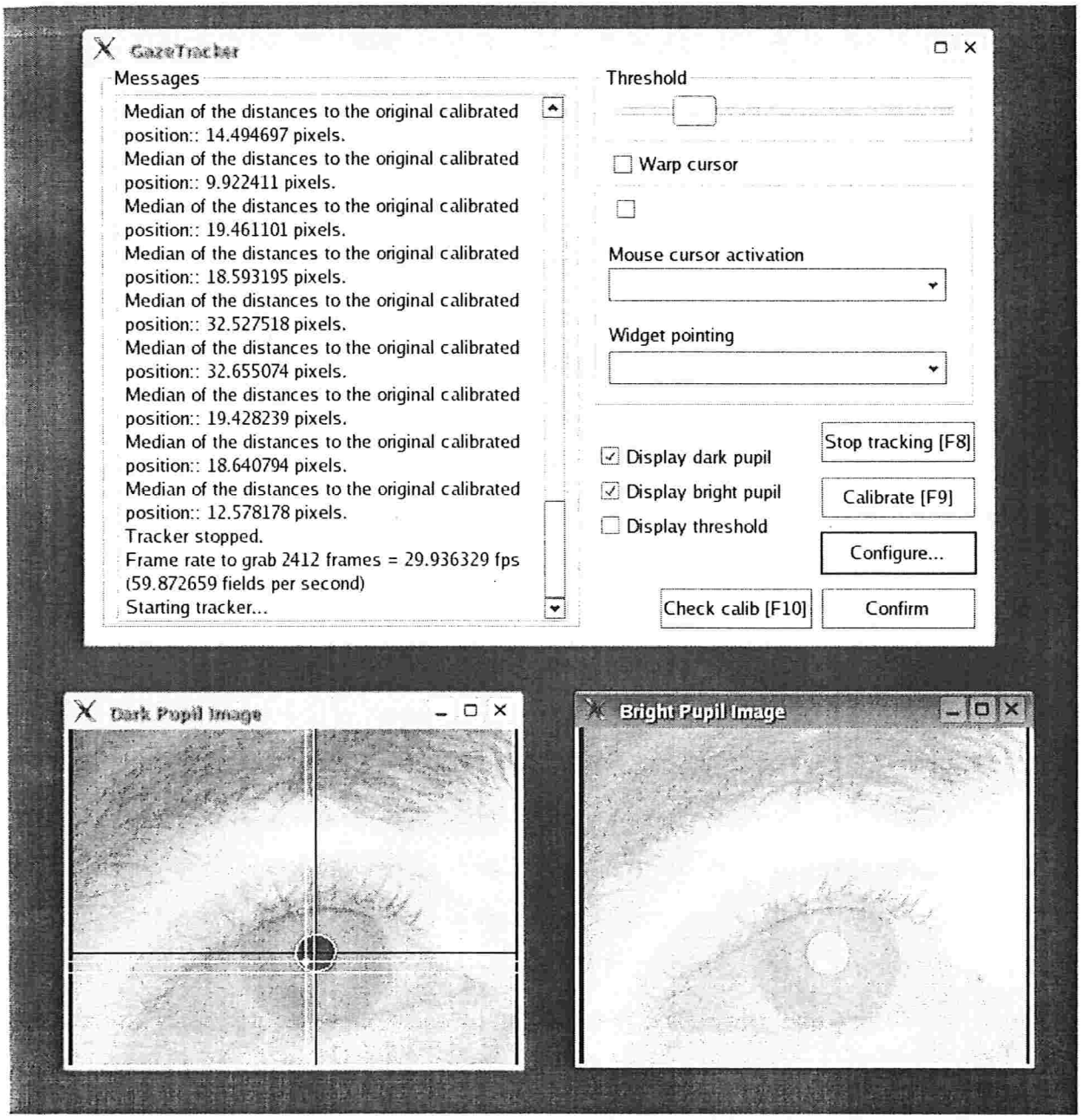


Figura 4.3: Interface do sistema de rastreamento do olhar

A interface de controle do rastreador de olhar e do MAGIC Pointing foi desenvolvida originalmente para Windows usando a MFC (Microsoft Foundation Classes) e no porte para Linux passamos a usar o *toolkit* QT [AS04], desenvolvido pela Trolltech e utilizado, por exemplo, pelo ambiente gráfico KDE. O QT provê uma API que simplifica e acelera o desenvolvimento de interfaces gráficas e pode ser distribuído livremente sob a licença GPL. O código do processo de captura e processamento de imagens é desacoplado do código da interface gráfica, permitindo, se necessário, o desenvolvimento de outra interface para o sistema, ou mesmo o uso futuro do rastreador e do MAGIC como uma biblioteca compartilhada.

4.1.1 Operação a 60 imagens por segundo

A primeira alteração efetiva na implementação do sistema após a migração para Linux foi o aumento da sua capacidade de processamento de 30 para 60 imagens por segundo; dessa forma, com uma maior resolução temporal do rastreamento do olhar, é possível reduzir a latência no reconhecimento dos principais comportamentos do olho, em especial a execução de fixações. Assim, diminui-se o tempo de resposta do sistema, que é uma característica desejável para a aplicação do rastreamento do olhar em um sistema interativo.

Utilizamos uma câmera NTSC (Figura 4.1), capaz de capturar 30 quadros de vídeo entrelaçado por segundo; em sua construção, ela é adaptada de forma que o funcionamento de cada um dos conjuntos de LEDs infra-vermelhos seja sincronizado com a captura de um dos dois campos que formam uma imagem entrelaçada, e assim cada conjunto permanece aceso por metade do tempo necessário para a captura da imagem. Efetuando-se o desentrelaçamento, de cada imagem é possível obter duas imagens do olho, correspondentes a cada campo. Em uma delas, a pupila estará iluminada e na outra a pupila estará escurecida; obtem-se a seguir a diferença entre essas imagens e a segmentação da pupila (Figura 4.2). Dessa forma é possível estimar a direção do olhar uma vez a partir de cada imagem de vídeo, ou seja, 30 estimativas por segundo.

Porém, também é possível detectar a pupila utilizando a diferença das imagens geradas por dois quadros capturados consecutivamente. Suponha que a imagem da pupila iluminada no instante t é obtida através do campo par da imagem (isto é, o campo que é armazenado nas

linhas pares da imagem original); chamemos essa imagem de P_t, e seja I_t a imagem da pupila escura no instante t, gerada pelo campo ímpar da imagem. Usualmente, a detecção da pupila é feita a partir da diferença $P_t - I_t$, enquanto que o objetivo da nossa alteração no código é detectar a pupila também através da diferença $P_{t+1} - I_t$. Assim, a cada imagem capturada é possível estimar duas vezes a direção do olhar, ocorrendo efetivamente o processamento de 60 campos por segundo.

A implementação de tal modificação no sistema baseia-se na criação de um área de armazenamento temporária para continuar armazenando I_t quando o sistema já está processando P_{t+1} e I_{t+1}. Porém, alguns cuidados adicionais são necessários. Na discussão do procedimento descrito acima, estamos sempre supondo que o conjunto de luzes próximo ao eixo óptico da câmera, que gera a imagem da pupila iluminada, está sincronizado com o primeiro campo da imagem (o campo par). Se ocorrer o inverso, ou seja, a pupila iluminada for gerada no momento da captura do segundo campo da imagem (o campo ímpar), a diferença correta a ser calculada para obter a imagem onde a pupila poderá ser segmentada não é $P_{t+1} - I_t$, mas $I_t - P_{t+1}$.

Assim, para que o sistema seja flexível o suficiente para operar com câmeras que eventualmente sincronizem os *leds* com a captura na ordem inversa, um procedimento adicional foi implementado. Esse procedimento funciona como um "estado de espera", ativo quando o sistema de rastreamento do olhar começa a funcionar. A cada instante t nesse estado, o sistema calcula as diferenças $P_t - I_t$ e $I_t - P_t$, e executa o algoritmo de detecção da pupila e do brilho ocular em cada uma delas. Se o algoritmo não foi bem sucedido em nenhuma das duas, o estado de espera continua e o procedimento é executado novamente após a captura do próximo quadro. Quando o algoritmo é bem sucedido na detecção da pupila na imagem gerada por uma das diferenças, o sistema a partir dessa informação identifica se a imagem da pupila clara poderá ser obtida no campo par ou no campo ímpar da imagem. Nesse momento o estado de espera termina e esse procedimento é adotado a partir daquele momento no processamento das imagens.

4.1.2 Reconhecimento de fixações e sacadas

Com as alterações que descrevemos na última seção, o sistema de rastreamento passa a fornecer a estimativa de um ponto observado pelo usuário a cada 16 milissegundos, aproximadamente. Como discutimos na seção 3.1.2, esse fluxo de dados deve ser analisado de forma a identificar as fixações e as sacadas, para que possamos começar a obter informação sobre o interesse visual do usuário.

A implementação inicial do MAGIC Pointing detectava fixações de forma bastante rudimentar: apenas a ocorrência de dois pontos consecutivos reportados pelo rastreador e cuja distância fosse menor que um determinado limiar definia a ocorrência de uma fixação. Com o objetivo de introduzir um método mais formal e que utilize mais informações sobre o comportamento dos dados ao longo do tempo, definimos e implementamos uma máquina de estados cuja função é identificar fixações e sacadas a partir da entrada fornecida pelo sistema de rastreamento do olhar. Além disso, essa máquina de estados também procura representar e tratar eventuais ruídos ou interrupções momentâneas de sinal ocorridas durante o processo de rastreamento.

A máquina tem os seguintes estados:

- Fixação;
- Sacada;
- Sem dados

Sabemos que os movimentos do olho humano não se restringem às fixações e sacadas; porém, distinguir os momentos nos quais a atenção visual está fixada em um ponto já fornece informação suficiente para a aplicação do rastreamento do olhar em sistemas interativos. Além disso, a resolução temporal de um rastreador operando a 60 Hz não é suficiente para identificar com precisão outros movimentos, como por exemplo os movimentos micros-sacádicos.

Dessa forma, durante seu funcionamento, a nossa máquina de estados está tipicamente nos estados *Fixação* ou *Sacada*. A permanência em um desses estados ou a transição entre eles depende da análise dos dados fornecidos pelo rastreador em um determinado intervalo de

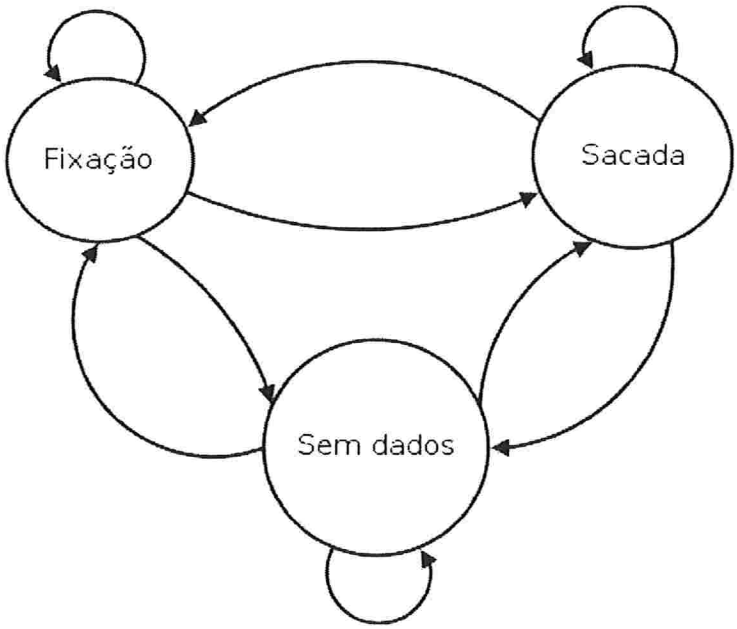


Figura 4.4: Diagrama da máquina de estados

tempo correspondente aos últimos instantes de funcionamento. Na escolha desse intervalo, dois fatores devem ser levados em consideração. Se por um lado é desejável obter uma amostra representativa do comportamento do olho nos últimos instantes, por outro lado não queremos que a espera para obter essa amostra seja muito longa de forma a introduzir um atraso na identificação de uma transição de estado que dificulte ou impossibilite a aplicação dos resultados da máquina de estados ao processo de interação.

Com esses fatores em mente, definimos o intervalo de análise como aquele necessário para a captura e processamento pelo rastreador das últimas 6 imagens não-entrelaçadas da pupila e do brilho ocular, que corresponde, aproximadamente, aos últimos 90 milissegundos, levando-se em conta o funcionamento a 60 Hz do sistema de rastreamento. Esse atraso na definição do estado pode ser considerado aceitável, já que é inferior ao tempo usual de duração de uma fixação ou da maioria das sacadas [YS75].

A transição de um estado para outro é determinada por uma análise da distribuição espacial dos pontos fornecidos pelo rastreador durante esse intervalo de tempo. Quando os pontos estão "próximos" entre si, é possível considerar que o usuário esteja efetuando uma fixação e que as pequenas variações entre pontos se devem aos ruídos introduzidos

Entrada: δ e p_i $(i = 1, \ldots, n)$

1: **para** $i = 1$ até $n - 1$ **faça**
2: Calcular $d_i = \text{dist}(p_i, p_{i+1})$
3: **fim para**
4: Calcular md, a mediana de $D = \{d_1, d_2, \ldots, d_{n-1}\}$
5: Calcular $\text{std}(D)$, o desvio padrão de D
6: Calcular $\max(D)$, o elemento máximo de D
7: **se** $md \leq \delta$ e $\max(D) \leq md + 2 \times \text{std}(D)$ **então**
8: Calcular a nova fixação $f = (x_f, y_f)$ como o centro de massa de $p_1, p_2, \ldots, p_n$
9: **senão**
10: Permaneça no estado atual
11: **fim se**

Figura 4.5: Algoritmo reconhecedor de fixações: identifica se uma amostra de dados do sistema de rastreamento corresponde à ocorrência de uma fixação quando a máquina de estados está atualmente no estado *Sacada*

pelo processo de rastreamento. Quando as distâncias entre pontos consecutivos aumenta, há a indicação que uma sacada está ocorrendo. Definimos δ, o limiar de proximidade entre pontos pertencentes a uma fixação, como o valor correspondente a um grau do ângulo visual do usuário. Esse valor corresponde ao limite da precisão obtida pelo nosso rastreador de olhar [MKAF00], e para seu cálculo é necessário que o sistema conheça, antecipadamente, a distância do usuário ao monitor e a sua resolução física. O primeiro valor deve ser fornecido manualmente pelo usuário no início da operação do sistema, e o segundo pode ser obtido através da API do sistema X-Window, sendo medido usualmente em pixels por polegada [Inc04].

Se o estado atual do sistema é *Sacada*, o critério para a transição para o estado *Fixação* usa como métrica a distância entre pontos fornecidos consecutivamente pelo rastreador. Com a amostra de dados dos últimos instantes, é executado o algoritmo da Figura 4.5.

Note que nesse algoritmo usamos duas medidas para avaliar a proximidade dos pontos potencialmente pertencentes a uma fixação: a mediana das distâncias é usada por ela ser mais robusta à eventuais ruídos do que, por exemplo, a média das distâncias. A limitação do valor máximo da distância entre pontos consecutivos na amostra através da comparação com o desvio padrão é uma das técnicas estatísticas mais simples para eliminação de valores muito discrepantes do conjunto dos dados (*outliers*). No caso da nossa máquina de estados, esse

Entrada: δ, p_i $(i = 1, \ldots, n)$ e $f = (x_f, y_f)$, o ponto da última fixação

1: Calcular $f(i)$, para $i = 1, 2, \ldots, n$, definido como a distância de p_i ao ponto da última fixação reportada.
2: Calcular para $i = 1, 2, \ldots, n-1$ a derivada discreta de $f(i)$, definida como $f'(i) = \frac{f(i+1)-f(i)}{i+1-i} = f(i+1) - f(i)$
3: Obter k tal que $f'(k) = \max_{i=1,\ldots,n-1} f'(i)$
4: **se** $f'(k) \geq \delta$ e $k \leq \lfloor n/2 \rfloor$ **então**
5: **se** $f(i), i \geq k$ é uma função crescente **então**
6: **para** $i = k$ até n **faça**
7: Calcular $\overrightarrow{v_i}$, o vetor de origem f e destino p_i
8: **fim para**
9: Calcular $\alpha = \max_{k \leq i,j \leq n} \measuredangle(\overrightarrow{v_i}, \overrightarrow{v_j})$ // Ângulo entre $\overrightarrow{v_i}$ e $\overrightarrow{v_j}$
10: **se** $\alpha \leq 15$ **então**
11: A amostra de dados corresponde a uma sacada em curso
12: **senão**
13: Permaneça no estado atual
14: **fim se**
15: **fim se**
16: **senão**
17: Permaneça no estado atual
18: **fim se**

Figura 4.6: Algoritmo reconhecedor de sacadas: identifica se uma amostra de dados do sistema de rastreamento corresponde à ocorrência de uma sacada quando a máquina de estados está atualmente no estado *Fixação*

critério funciona como uma "barreira" para que pontos que ainda correspondem aos últimos instantes de uma sacada não pertençam à amostra que será usada para o cálculo do ponto da fixação.

Se o estado atual do sistema é *Fixação*, o critério para a transição para o estado *Sacada* passa a usar como métrica a distância entre os últimos pontos obtidos pelo rastreador e o ponto da fixação atual. Essa métrica foi escolhida para possibilitar a identificação do padrão típico da ocorrência de uma sacada: à medida que o olho se desloca em direção à posição da próxima fixação, a seqüência de pontos fornecida pelo rastreador começa a apresentar distâncias crescentes em relação ao ponto da última fixação. Ainda, a velocidade de afastamento dos pontos em relação à ultima fixação cresce rapidamente. Para a identificação desse padrão, é executado o algoritmo da Figura 4.6.

Com o cálculo de $f'(i)$ na linha (2), procuramos obter uma estimativa da velocidade

instantânea dos pontos fornecidos pelo rastreador. O algoritmo busca na amostra o instante onde ocorreu a máxima velocidade de deslocamento. A condição da linha (4) garante que a amostra de dados só será analisada se, no intervalo entre o processamento de dois quadros (aproximadamente 16 ms), houve um deslocamento maior do que α. Nas linhas (5) a (9), calcula-se a direção do deslocamento dos pontos; se as distâncias dos pontos em relação ao ponto da última fixação são crescentes e esses pontos estão contidos em um cone com origem no ponto da última fixação, concluímos que os pontos obtidos pelo rastreamento realmente indicam um deslocamento em direção à uma nova fixação, e a máquina muda seu estado para *Sacada*.

A máquina de estados também leva em consideração o fato de que pode haver interrupções momentâneas no fornecimento de dados pelo rastreador de olhar. Tais interrupções podem ocorrer por alguma falha no processo de detecção da posição observada provocada pela captura de imagens ruidosas ou simplesmente quando o usuário pisca o olho, impossibilitando por alguns instantes a detecção. Quando ocorre uma interrupção de sinal em qualquer um dos estados, é feita uma transição para o estado *Sem dados*. A máquina permanece nesse estado enquanto não é restabelecido o fluxo de dados. Uma vez que isso aconteça, a transição para o próximo estado depende da duração do intervalo de tempo em que não houve fornecimento de dados – se a interrupção ocorreu por um intervalo de tempo razoavelmente pequeno, podemos considerar que o último evento registrado pela máquina de estados continua acontecendo. Definimos esse valor como 160 milissegundos após testes empíricos com usuários que utilizaram o sistema após a implementação da máquina de estados. Constatamos, em especial, que esse valor atua bem como o limiar para que, após uma interrupção do sinal provocada pelo usuário piscando os olhos, o sistema não interprete erroneamente o novo fluxo de dados como o início de uma nova fixação.

4.2 Sistema de Apontamento Prestativo Orientado pelo Contexto

Para realizar as suas tarefas rotineiras nas interfaces WIMP, os usuários se habituaram a lidar com controles, tais como caixas de texto, barras de rolagem, menus e barras de ferramentas. Quando se decompõe uma tarefa usual nessas interfaces em subtarefas, em algum ponto da decomposição invariavelmente várias subtarefas se caracterizam por "selecionar o controle X" ou "apontar o controle Y".

Se grande parte do tempo que os usuários passam interagindo com interfaces WIMP é gasto apontando e selecionando controles com um dispositivo de apontamento convencional, parece razoável incorporar à nossa técnica de interação a informação sobre o posicionamento e tamanho dos controles como parte do contexto. Assim, tendo essa informação disponível, e levando em consideração que o olho dificilmente é atraído por espaços vazios, propomos usar o mesmo conceito do MAGIC Pointing: deslocar o cursor usando o olhar juntamente com um movimento motor, porém não mais reposicioná-lo diretamente sobre a posição observada na tela, mas sobre o controle mais próximo dessa posição. Denominamos nosso sistema, que incorpora essa nova extensão, de SAPOC - Sistema de Apontamento Prestativo Orientado pelo Contexto.

Continuam existindo as estratégias liberal e conservadora de reposicionamento do cursor que vimos na seção 3.2.4. A opção de apontar para o controle mais próximo passa a ser um comportamento adicional dessas duas estratégias. Vimos que a diferença fundamental entre elas é a necessidade de efetuar ou não um movimento inicial com mouse para que o cursor seja reposicionado. Porém, o SAPOC também utiliza outros critérios para reposicionar o cursor, que são baseados em informações adicionais sobre o contexto atual da interação. Esses critérios são incorporados em uma tentativa de prever com maior segurança as situações em que o usuário deseja reposicionar o cursor.

Um critério é comum a ambas as modalidades: o cursor não pode ser reposicionado para uma vizinhança de sua posição atual, pois consideramos que essa região já representa o ponto de interesse do usuário, o local onde provavelmente alguma ação já está sendo executada, e reposicionar o cursor nessa situação pode interromper ou atrapalhar a execução dessa

ação. Definimos essa vizinhança como a área que cobre um grau do ângulo visual; esse valor corresponde aproximadamente ao foco de atenção visual do olho humano. Considerando um monitor observado pelo usuário à uma distância de 50 cm, que é a configuração usual de posicionamento da maioria das estações de trabalho, essa área corresponde a pontos que distem, no máximo, cerca de 1 cm da posição atual do cursor.

Mais dois critérios foram implementados com a finalidade de diminuir a sensibilidade do mecanismo de reposicionamento do SAPOC, evitando que o cursor seja reposicionado por qualquer esbarrão acidental no mouse. No caso da modalidade liberal, o reposicionamento acontece somente quando o mouse não foi deslocado nos últimos 50 milissegundos; dessa forma procura-se evitar o reposicionamento do cursor enquanto o usuário ainda está movimentando o mouse, potencialmente no curso da execução de uma tarefa.

Em contrapartida, na estratégia conservadora é necessário estabelecer uma medição da intensidade do deslocamento inicial do cursor. Para isso, medimos a *magnitude* do movimento inicial, definida como a distância percorrida pelo cursor sobre o monitor no intervalo mínimo de varredura de dados do mouse. Porém, note que a definição do valor mínimo da magnitude do deslocamento para que o cursor seja reposicionado depende de algumas particularidades do *hardware* do sistema que executa o SAPOC. O deslocamento do cursor sobre o monitor é medido em pixels, cuja equivalência para centímetros depende da disponibilidade da informação sobre a resolução física do monitor. Além disso, como propomos utilizar a magnitude do deslocamento como uma medida da "sensibilidade" da atuação do sistema, a definição do seu valor mínimo pode também variar em função da resolução física do mouse em uso e, em último caso, da preferência pessoal do usuário. No sistema utilizado na implementação do nosso protótipo (veja apêndice B), fixamos o valor da magnitude do deslocamento inicial como 2 pixels. Imaginamos que, futuramente, seja necessário permitir que esse valor possa ser ajustado em tempo de execução pelo usuário.

Além disso, mais uma modalidade de apontamento e seleção foi implementada: a seleção por latência (*dwell time*). O funcionamento dessa modalidade é muito similar à modalidade liberal, com uma diferença: quando o cursor é reposicionado, a seleção é efetuada automaticamente após um pequeno intervalo de tempo, através da geração de um evento de "clique" de mouse com o auxílio da extensão XTest do sistema gráfico X [Inc04]. Já discutimos ante-

Critério	Liberal	Conservadora
Movimentação do mouse	Sem eventos nos últimos 50 ms	Evento após a ocorrência da fixação, desde que magnitude do deslocamento maior que 2 pixels[1]
Posição da fixação	Distante pelo menos 1 cm[2] da posição atual do cursor	Distante pelo menos 1 cm[2] da posição atual do cursor

[1] Considerando a nossa configuração atual
[2] Monitor a 50cm

Tabela 4.1: SAPOC: Quadro resumo dos critérios de reposicionamento do cursor

riormente algumas desvantagens das técnicas de interação baseadas em seleção por latência; porém, a incluimos no nosso sistema como mais um parâmetro de comparação para as modalidades do SAPOC. A seleção por latência difere do SAPOC por usar apenas a informação do rastreamento do olhar para efetuar a seleção, sem combiná-la com nenhuma informação advinda de um dispositivo de apontamento convencional.

Na Tabela 4.1 resumimos os critérios de reposicionamento do cursor para cada uma das modalidades.

O SAPOC mantém uma lista dos controles selecionáveis a cada momento e seu respectivo posicionamento e tamanho. Essa lista deve ser atualizada sempre que uma janela é redimensionada, movida, minimizada ou maximizada. Atualmente, apenas os controles da janela ativa são armazenados nessa lista, para simplificar a implementação; usamos a hipótese que a seleção de controles fora da janela ativa é um evento raro, e que usualmente é necessário trazer a janela para o primeiro plano antes de efetuar tal ação.

Nossa implementação provê duas alternativas para a construção da lista de controles. Na primeira, o SAPOC detecta automaticamente quando uma nova janela é trazida para o primeiro plano ou a janela atual muda suas dimensões e, a partir do ID dessa janela, fornecido pelo sistema gráfico X Window [Inc04], percorre uma estrutura em árvore das chamadas *subjanelas* vinculadas a esse ID, obtendo a partir do ID dessas subjanelas o posicionamento e tamanho dos controles. Em nossos testes, verificamos que essa alternativa tem sucesso em obter os controles de programas construídos com os *toolkits* MOTIF e QT [AS04], mas não nos programas construídos com o GTK+ [Pro04]. Esse *toolkit* não faz uso dos IDs fornecidos

pelo X Window para identificar os controles que cria. Para suprir essa deficiência e prover maior flexibilidade, a segunda alternativa permite que uma aplicação construída ou adaptada para ser operada com o MAGIC Pointing informe a ele a posição dos seus controles através de um mecanismo de chamada remota de procedimentos (RPC - *Remote Procedure Calls*) denominado DCOP (Desktop COmmunications Protocol [BE04]). O DCOP é disponibilizado pelo ambiente KDE para intercomunicação entre suas aplicações, e foi desenvolvido para ser extremamente simples e portanto mais leve que outros mecanismos de aplicação mais geral como o CORBA (Common Object Request Broker Arquitecture), por exemplo.

Capítulo 5

Avaliação experimental

O desenvolvimento de uma nova técnica de interação não pode acontecer sem que haja a participação de usuários em alguns ciclos de avaliação. Apesar da existência de várias metodologias de suporte ao desenvolvimento de sistemas interativos [DFAB97], a avaliação do projeto ou de um protótipo do sistema com a participação de usuários permite uma simulação mais próxima do seu uso real, trazendo informações valiosas para o início de um novo ciclo de desenvolvimento.

No nosso caso, a avaliação do SAPOC tem dois objetivos. Primeiramente, desejamos definir e obter uma medida do desempenho do sistema na execução de tarefas de apontamento e seleção; essa medida deve permitir que o desempenho do SAPOC seja comparado com o desempenho obtido, na mesma tarefa, com o uso do mouse. Nossa comparação será baseada em um modelo do tempo necessário para a execução de tarefas de apontamento – a *Lei de Fitts*, que tem sido bastante utilizada na pesquisa em IHC [IR02, MKS01, ZMI99, ISB91] como ferramenta para análises comparativas entre dispositivos de apontamento. Além disso, também desejamos que os usuários participantes possam avaliar o sistema em aspectos subjetivos, tais como conforto e facilidade de uso.

Para atingir esses objetivos, projetamos e conduzimos um experimento, que será descrito na seção 5.2. Na seção 5.3 analisamos e comentamos os resultados obtidos.

5.1 Lei de Fitts

Uma medida usual do desempenho de dispositivos de apontamento é dada pela *Lei de Fitts*. Proposta por P.M. Fitts na década de 50 [Fit54], ela fornece uma estimativa do tempo necessário para uma pessoa executar um movimento manual em direção a um alvo em função da distância do alvo e o tamanho do mesmo. Pode ser expressa através da equação

$$T = a + b \log_2(2A/W) \tag{5.1}$$

onde T é o tempo de execução do movimento, A é a amplitude do movimento realizado e W é o tamanho do alvo. Os coeficientes a e b são constantes usualmente determinadas empiricamente através de regressão linear, podendo variar para tarefas em diferentes contextos. O termo $2A/W$ do logaritmo é por vezes chamado de *índice de dificuldade* (ID) da tarefa.

Um dos experimentos propostos originalmente por Fitts envolvia o toque de duas chapas metálicas dispostas paralelamente com uma caneta e obtenção do tempo necessário para tocar a segunda chapa após tocar a primeira: a amplitude A é tomada como a distância entre as chapas e W é tomado como a largura da chapa. Outros experimentos envolviam a transferência de discos entre pinos dispostos também paralelamente.

Um modelo validado em um contexto de aplicação à primeira vista tão diferente do contexto dos sistemas computacionais interativos foi aplicado com sucesso na previsão do tempo necessário para a execução de tarefas de apontamento com o uso de dispositivos variados. MacKenzie, Sellen e Buxton [ISB91] propuseram a seguinte variação da formulação da Lei de Fitts:

$$T = a + b \log_2(A/W + 1) \tag{5.2}$$

Essa formulação mostrou ter melhor aderência aos dados empíricos obtidos nos experimentos originais conduzidos por Fitts. Nela, o índice de dificuldade $ID = \log_2(A/W + 1)$ é maior que zero por construção, e o inverso do coeficiente b é denominado *índice de desempenho* (IP, do inglês *index of performance*), medido em bits por segundo. A unidade de medida de ID é chamada de *bits* devido ao logaritmo na base 2. O índice de desempenho pode ser usado como uma medida de comparação entre diferentes dispositivos de aponta-

mento. Assim, valores de IP maiores indicam que o tempo necessário para executar a tarefa de apontamento é menos sujeito a aumentos do índice de dificuldade ID da tarefa. Para dispositivos com índices de desempenho menores, pode-se prever que o acréscimo de tempo nas mesmas condições será maior.

5.2 Descrição do experimento

O nosso experimento foi desenvolvido com o objetivo de comparar os dispositivos tradicionais de apontamento com o MAGIC Pointing e o SAPOC, utilizando a Lei de Fitts como uma métrica de comparação. Em especial, queremos investigar qual técnica de interação, em quais situações, permite executar uma tarefa de apontamento e seleção em menor tempo.

A formulação da Lei de Fitts já começa a indicar alguns critérios para criar as situações que serão apresentadas no experimento: alvos a serem selecionados com diferentes tamanhos, e que apresentem diferentes distâncias do "alcance da mão" – no caso das interfaces WIMP, esse conceito se traduz na distância que se deve deslocar o cursor para que ele atinja o alvo a ser selecionado.

Além disso, queremos ocupar uma mesma área física do monitor com diferentes quantidades de alvos, criando assim uma variação do que poderíamos chamar de "densidade de ocupação", ou "grau de confusão" da interface. Como sabemos que a precisão do rastreamento do olhar é limitada, pretendemos verificar quando interfaces mais "densas" prejudicam o desempenho do apontamento auxiliado pelo rastreamento do olhar.

Naturalmente, a velocidade de execução de uma tarefa não basta para avaliar uma técnica de interação. Outros critérios, tanto objetivos quanto subjetivos, devem ser levados em consideração nessa análise. Assim, nessa seção definiremos a tarefa a ser executada no experimento, bem como quais dados deverão ser coletados e os procedimentos a serem adotados na condução do experimento.

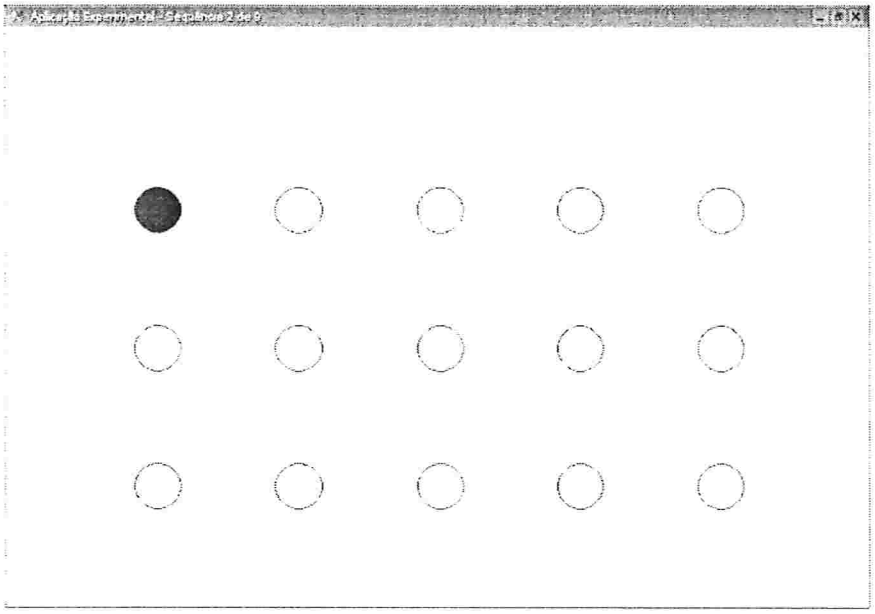


Figura 5.1: Aplicação desenvolvida para a condução do experimento

5.2.1 Tarefa

Para a execução do experimento, foi construída uma aplicação na linguagem Java (figura 5.1). Em uma janela maximizada, são exibidos círculos dispostos em uma grade. A cada momento, apenas um círculo aparece destacado em uma cor diferente; esse círculo deve ser apontado e selecionado. Uma vez selecionado, ele deixa de aparecer em destaque e um novo círculo passa a aparecer destacado e deve ser por sua vez apontado e selecionado, e assim por diante.

Os círculos são dispostos de forma a maximizar o preenchimento de uma área retangular de 30 × 20 cm no centro do monitor. Em cada tela do experimento, a disposição dos círculos corresponde a uma das possíveis combinações dos seguintes parâmetros:

Diâmetro dos círculos (W): 1 cm ($\approx 1,14°$), 2 cm ($\approx 2,29°$) ou 4 cm ($\approx 4,58°$);

Espaçamento entre os círculos (G): 0 cm, 2 cm ou 4 cm.

Consideramos como espaçamento entre os círculos a distância entre as bordas dos círculos (veja exemplo na figura 5.2). A manipulação desse parâmetro tem como objetivo variar a densidade de ocupação da tela, conforme discutimos no início desta seção. Levando em consideração que o monitor será observado a uma distância de 50 cm, os valores escolhidos

para o tamanho dos círculos e espaçamento correspondem aproximadamente ao valor em graus do ângulo visual que mencionamos em parênteses.

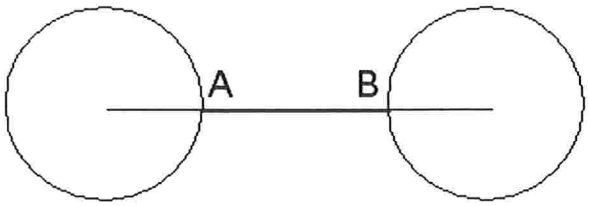


Figura 5.2: Espaçamento entre círculos no experimento: a medida do segmento de reta $\overline{AB}$ entre círculos adjacentes é uma das variáveis manipuladas

As combinações possíveis de tamanhos de círculos e espaçamento entre os círculos geram nove telas diferentes, sendo que em cada uma delas devem ser efetuadas 16 seleções de alvos. Os alvos que farão parte dessa seqüência são escolhidos de acordo com o algoritmo que descreveremos na seção 5.2.2. Daqui em diante, esse conjunto de nove telas será referenciado como um *bloco*.

A cada seleção efetuada, é registrado o tempo de execução da tarefa, computado desde o instante da seleção do último círculo destacado até a seleção do círculo atualmente destacado. Se a a seleção foi feita de forma incorreta, a ocorrência de um erro é registrada e o próximo alvo da seqüência é exibido. A causa do erro também é registrada, dentre duas alternativas: seleção de um alvo não destacado, ou seleção feita com o cursor posicionado fora de qualquer alvo.

Adicionalmente, também registramos a distância entre o centro do círculo atualmente destacado e a última posição para a qual o cursor foi reposicionado antes da seleção. Para as modalidades do SAPOC, esperamos obter assim uma medida do desempenho da inferência do sistema. Note que, em condições ideais, desejaríamos que essa medida fosse igual a zero em todos os reposicionamentos; pretendemos então analisar para quais configurações essa medida de desempenho se aproxima do ideal.

5.2.2 Definição da seqüência de seleções

Para cada uma das telas, a seqüência de círculos a serem selecionados é previamente definida, e essa mesma seqüência é apresentada a todos os participantes do experimento. Para auxiliar a nossa análise posterior dos resultados, desejamos que a distância A entre os centros de dois círculos consecutivos da seqüência também seja um parâmetro a ser manipulado no experimento. Mesmo que não seja possível controlar o valor exato dessa distância por causa da disposição dos círculos em uma grade, gostaríamos de garantir que as seleções efetuadas durante o experimento pudessem ser agrupadas em classes correspondentes, ao menos, a alvos "próximos" e "distantes"

Assim, construímos um algoritmo para a definição dessa seqüência de círculos, que funciona da seguinte maneira: em cada tela, definimos que a seqüência sempre se inicia com o círculo localizado no canto superior esquerdo. A seleção desse círculo serve para iniciar a medição do tempo necessário para o apontamento e seleção de alvos pareados. Seja C o centro desse círculo, que denominaremos $\mathcal{C}_\mathsf{C}$. Sorteamos um ângulo α no intervalo $[0; 2\pi]$ e uma distância referencial d_r no conjunto $\{5\,\text{cm}, 10\,\text{cm}, 20\,\text{cm}\}$, e então definimos $\overrightarrow{v}$ como o vetor de origem em C e módulo d_r, com angulação α considerando a orientação do círculo trigonométrico. O círculo cujo centro esteja mais próximo da extremidade de $\overrightarrow{v}$ é definido como o próximo elemento da seqüência.

Na figura 5.3 exibimos um exemplo de execução de um passo desse algoritmo. Seja $\mathcal{C}_\mathsf{C}$ um círculo intermediário na seqüência de seleções. A partir de $\mathcal{C}_\mathsf{C}$, é sorteado $\alpha = \pi/4$ e $d = 5$. O círculo cujo centro está mais próximo da extremidade do vetor $\overrightarrow{v}$ é $\mathcal{C}_\mathsf{B}$, e ele será o próximo elemento da seqüência. Na definição da seqüência de cada tela, cada distância pode ser sorteada no máximo cinco vezes; assim o usuário selecionará uma seqüência de 16 círculos, contando com a seleção inicial no canto superior esquerdo.

Com essa forma de definir a seqüência de seleções, asseguramos que as distâncias entre alvos consecutivos possam ser agrupadas em três classes, que correspondam, aproximadamente, a deslocamentos "pequenos" (5 cm), "médios" (10 cm) e "grandes" (20 cm). Juntamente com a variação no seu diâmetro, uma maior variabilidade das distâncias entre os círculos da seqüência permite produzir resultados que descrevam o funcionamento da técnica

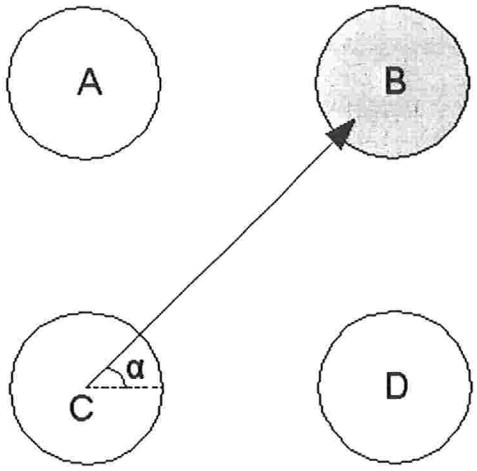


Figura 5.3: Exemplo do algoritmo de sorteio dos círculos: a partir do círculo C_C, sorteando-se $\alpha = \pi/4$ e $d = 5$, o próximo círculo da seqüência será C_B

de interação em uma maior gama de situações.

5.2.3 Procedimento

O usuário participante do experimento é inicialmente recepcionado e orientado sobre o experimento e sobre a tarefa que deverá ser realizada. Nesse momento começa uma fase inicial de treinamento, onde o usuário deve executar um bloco do experimento para cada uma das seguintes modalidades de apontamento e seleção, nessa ordem:

- mouse;
- modalidade liberal do MAGIC Pointing;
- modalidade liberal do SAPOC;
- modalidade conservadora do MAGIC Pointing;
- modalidade conservadora do SAPOC;
- seleção por latência.

O bloco do treinamento é elaborado especialmente para essa finalidade; nele, em cada uma das 9 telas descritas na seção 5.2.1 o usuário efetua 7 seleções. A configuração do bloco usado para o treinamento é o resultado da execução do algoritmo de definição da seqüência de círculos que descrevemos na seção 5.2.2, com a alteração que cada distância no conjunto {5cm, 10cm, 20cm} é sorteada no máximo duas vezes. Esse procedimento tem como finalidade garantir que a duração do bloco de treinamento seja aproximadamente a metade da duração do bloco do experimento. No treinamento de cada modalidade, o bloco pode ser repetido quantas vezes o usuário desejar, mas é solicitado que ele o execute pelo menos uma vez. Essa configuração tem como objetivo permitir que o usuário possa experimentar as diferentes situações de apontamento e seleção que encontrará a seguir no experimento. Durante o treinamento, ele é encorajado a explorar a interface para que possa a seguir executar a tarefa com o melhor desempenho possível.

Após a fase de treinamento, o usuário executa um bloco para a coleta de dados em cada uma das modalidades. Esse bloco volta a ter as 16 seleções que descrevemos anteriormente. Nessa fase, a ordem das telas dentro de cada bloco é aleatorizada, bem como a ordem em que as modalidades de apontamento e seleção são utilizadas.

Utilização do mouse

A maioria dos ambientes gráficos atuais permite ajustar alguns parâmetros para personalizar o comportamento do cursor ao estilo de movimentação de cada usuário. O principal desses parâmetros é chamado de *aceleração* do cursor. Esse recurso permite que cada unidade de medida do deslocamento físico do mouse corresponda a mais de uma unidade de medida no deslocamento do cursor na tela, diminuindo o esforço necessário para obter deslocamentos mais amplos do cursor. O deslocamento físico do mouse é medido em pontos e sua resolução física, em pontos por polegada (dpi - *dots per inch*). O deslocamento em pontos é usualmente mapeado diretamente para o deslocamento em pixels na tela.

Podemos ver que alterações na aceleração do cursor podem provocar alterações nos resultados do nosso experimento, assim surge a necessidade de padronizar esse ajuste. Para obter uma comparação mais fiel com o uso real dos sistemas interativos, decidimos ativar a aceleração do mouse de acordo com os valores da configuração padrão do sistema gráfico

X-Window; nele, dois parâmetros controlam a aceleração do mouse. O primeiro deles é um fator multiplicativo linear entre o deslocamento do mouse e o número de pixels que o cursor será efetivamente deslocado, cujo valor padrão é 2. O segundo indica o deslocamento mínimo do mouse entre duas interrupções para que a aceleração seja ativada, e esse valor é configurado como 6 pontos. Uma configuração similar é usada como padrão pelo Windows, da Microsoft [Cor05].

Cuidados adicionais com a calibragem

Tanto na fase de treinamento quanto na fase de coleta de dados, o uso do MAGIC Pointing e do SAPOC dependem do uso do nosso sistema de rastreamento do olhar. Vimos nas seções 4.1 e 2.2.2 que limitações da tecnologia disponível atualmente exigem que seja executado um procedimento de calibragem a cada vez que se inicia o uso do sistema e que a cabeça permaneça estática durante o uso.

Por esse motivo, consideramos a necessidade de minimizar o desconforto dos usuários durante a condução do experimento. Mas, ao mesmo tempo, manter a precisão do sistema de rastreamento do olhar durante o seu funcionamento é necessário para garantir o bom funcionamento da nossa técnica de interação e, por conseqüência, a qualidade dos dados obtidos no experimento.

Durante o uso do sistema de rastreamento, os usuários foram orientados a manter a cabeça em um apoio construído especialmente para a condução do experimento (figura 5.4) como uma forma de manter a qualidade da calibragem. Para minimizar a fadiga muscular normalmente causada por manter a cabeça e o pescoço parados por vários minutos, os usuários foram encorajados a fazer pausas entre os blocos sempre que desejassem.

Nos testes piloto do nosso experimento, verificamos que a adoção desses procedimentos de forma sistemática já contribuía bastante para evitar problemas com o funcionamento do sistema devido à movimentações acidentais da cabeça dos usuários. Porém, para assegurar o bom funcionamento do sistema decidimos implementar um procedimento adicional de verificação da calibragem. Esse procedimento atua durante a execução da tarefa do experimento: durante o uso de uma das modalidades do MAGIC Pointing ou do SAPOC, quando o usuário seleciona o círculo destacado, o programa do experimento monitora a posição onde

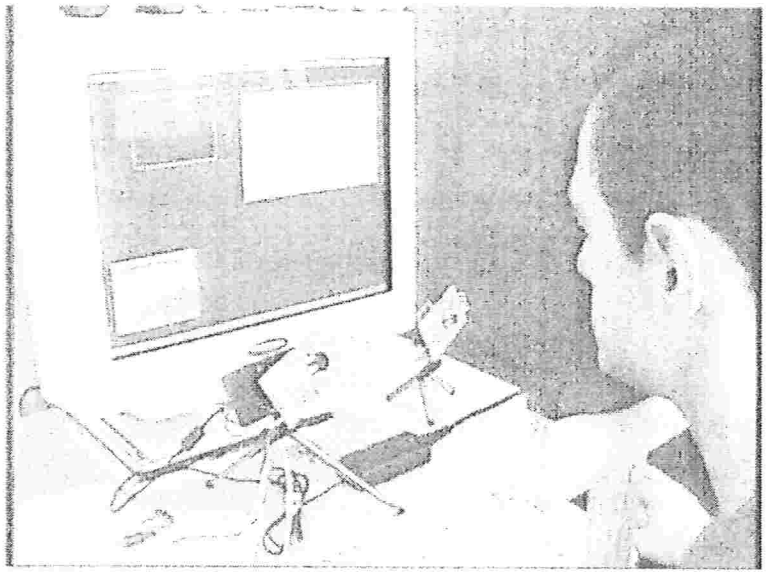


Figura 5.4: Apoio para queixo usado no experimento

foi efetuada a última fixação pelo usuário antes da seleção, conforme reportada pelo sistema de rastreamento. Sabemos que pelo funcionamento dessas modalidades, o usuário deve olhar para o círculo para deslocar o cursor do mouse; assim, se a última posição reportada pelo rastreador está muito distante do círculo destacado, podemos assumir que o erro introduzido pela calibragem é excessivo.

Para definir o limiar do erro aceitável, usamos novamente como referência o valor de um grau do ângulo visual, correspondente ao limite de precisão do sistema de rastreamento. Seja o alvo atual um círculo de raio r, d_f a distância entre a posição da última fixação reportada antes da seleção e o centro do alvo atual, e δ o valor correspondente a um grau do ângulo visual (calculado da mesma forma que descrevemos na seção 4.1.2). Enquanto $d_f \leq r + \delta$, consideramos que a calibragem está aceitável; se, para alguma seleção, $d_f > r + \delta$, a execução da tarefa naquela tela é interrompida, e o programa solicita que a calibragem seja feita novamente. O experimento se reinicia a partir do início da tela onde foi detectado o problema na calibragem.

Ilustramos um exemplo da aplicação desse critério na figura 5.5. Supondo que o monitor é observado à distância de 50 cm, a área hachurada mostra a região em torno de um círculo de 2 cm de diâmetro onde pode ocorrer a última fixação antes da seleção do alvo e a calibragem ainda ser considerada aceitável. Note que nosso critério busca contemplar uma situação limite

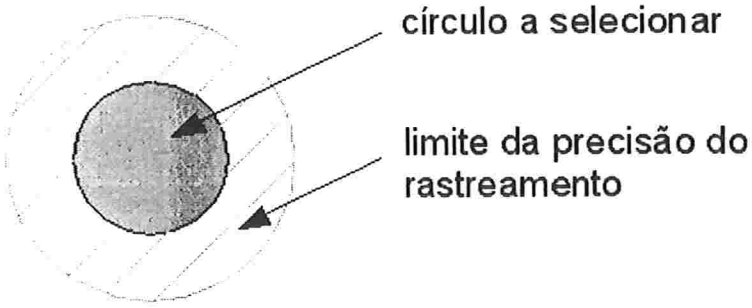


Figura 5.5: Exemplo do limite da precisão do sistema de rastreamento do olhar

em que uma fixação é feita sobre a borda do círculo.

Avaliação subjetiva

Após o término de todos os blocos, os participantes do experimento preenchem um questionário onde podem opinar sobre as modalidades em relação a três aspectos: facilidade de uso, rapidez e conforto. Esse questionário tem como objetivo coletar a opinião dos usuários sobre algumas características do nosso sistema que não podem ser mensuradas diretamente. Dessa forma, desejamos estimular que os usuários opinem sobre *quanto* a interação se torna melhor ou pior no uso das modalidades que usam o rastreamento do olhar em relação ao uso apenas do mouse.

Para obter essa comparação desejada, no preenchimento do nosso questionário (que está reproduzido no apêndice A), cada participante do experimento dá uma nota de -3 a +3 em cada quesito para cada uma das cinco modalidades que usam o rastreamento do olhar. A nota atribuída é uma comparação de cada modalidade no quesito com o uso somente do mouse, de forma que notas próximas de -3 significam avaliações piores e notas próximas de +3 significam avaliações melhores.

Buscando uma maior fidelidade nos resultados dessa avaliação, o preenchimento do questionário é feito de forma a garantir o anonimato dos participantes. O questionário não traz nenhum campo para a identificação do participante, e após preenchido ele é diretamente depositado em uma urna.

5.3 Resultados

Participaram do experimento 26 voluntários, dentre alunos do Instituto de Matemática e Estatística da USP. Foi solicitado a cada usuário que informasse há quanto tempo usa computadores com mouse; a nossa amostra de usuários apresentou uma experiência média de 10,3 anos.

Inicialmente realizamos uma análise de variância (ANOVA) multifatorial de medidas repetidas [MdL99, Lan05, Car05], a fim de investigar o efeito dos fatores (variáveis manipuladas no experimento) no tempo de apontamento e seleção de pares de círculos, bem como o efeito da interação entre os fatores combinados dois a dois, três a três e assim por diante. Nessa análise foram incluídos como fatores a modalidade, o diâmetro do círculo (W), a distância de apontamento (A) e o espaçamento entre círculos (G).

O objetivo de uma análise de variância é avaliar se alterações dos fatores manipulados no experimento influenciam a variável dependente medida – o tempo de apontamento e seleção, no nosso caso. Nossa análise é dita de *medidas repetidas* porque obtemos repetidas medições de tempo para cada indivíduo participante do experimento, uma para cada variação do valor dos fatores. Porém, de acordo com a definição da seqüência de seleções (seção 5.2.2), em cada tela gerada pela variação do diâmetro W e do espaçamento G o usuário executa cinco apontamentos e seleções para cada classe de distâncias A (5, 10 ou 20 cm); assim, os tempos obtidos em todas as execuções sem ocorrência de erros dentre essas cinco são agrupados em uma média para efeito da nossa análise.

Para cada combinação de s fatores testados, é calculada a estatística F_{obs} para testar a hipótese nula H_0 : *A combinação de fatores não influi na variável dependente* contra a hipótese alternativa H_a : *A combinação de fatores influi na variável dependente.* Sob a hipótese nula, F_{obs} segue a distribuição estatística $F_{a_1.a_2}$ de Fisher-Snedcor com os parâmetros a_1 e a_2. Se os fatores da combinação testada assumem valores $k_1, k_2, \ldots, k_s$ diferentes e a amostra tem tamanho n, então $a_1 = \prod_{i=1}^{s}(k_i - 1)$ e $a_2 = (n-1)\prod_{i=1}^{s}(k_i - 1)$. A partir dos valores da distribuição de $F_{a_1.a_2}$, e da estatística F_{obs} obtida, calculamos o valor p, correspondente à probabilidade do *erro do tipo I*, que significa rejeitar a hipótese nula dado que ela é verdadeira. Quanto maior for o valor obtido para F, maior é a probabilidade de rejeitarmos

a hipótese nula. Assim $p = P(F \geq F_{obs}), F \sim F_{a_1,a_2}$, ou seja, a probabilidade de, sob a hipótese nula, obter um valor tão grande quanto ou maior que F_{obs}. Se essa probabilidade p for menor que um dado valor limite, concluímos que a hipótese nula é falsa. Optamos por usar o valor usual de 1% para esse limiar.

Note que, deste ponto em diante, usaremos a notação $F_{a_1,a_2} = f$ para denotar que o valor calculado de F_{obs} é igual a f para o teste em questão, e que sob a hipótese nula ele segue a distribuição de Fisher-Snedcor com parâmetros a_1 e a_2. Em geral apresentaremos também o valor de p que levou à aceitação ou rejeição da hipótese nula.

Isoladamente, a modalidade afeta o tempo de apontamento de forma significativa ($F_{5,125} = 45,46$, $p < 0,001$), bem como o espaçamento entre círculos G ($F_{2,50} = 59,42$, $p < 0,001$). A interação Modalidade $\times$ G também foi significativa ($F_{10,250} = 11,60, p < 0,001$). Essa interação significa que o efeito da alteração no espaçamento entre os círculos no tempo da tarefa não é o mesmo para todas as modalidades.

Esse resultado pode ser visualizado no gráfico da figura 5.6, que representa o tempo médio de execução da tarefa em cada configuração de espaçamento entre os círculos para cada modalidade. Note que a variação do tempo entre as configurações é semelhante para o mouse e as modalidades do MAGIC Pointing, enquanto que as modalidades do SAPOC apresentam uma notável degradação do desempenho quando o espaçamento é igual a 0 cm, em especial a modalidade liberal.

Analisando a interação Modalidade $\times$ $A \times W$, notamos que ela só é significativa quando é incluída a seleção por latência na análise ($F_{20,500} = 3,81, p < 0,001$, com a inclusão da seleção por latência, contra $F_{16,400} = 0,04, p = 0,64$). Ou seja, não há diferença estatística no efeito combinado desses dois fatores sobre o tempo de execução da tarefa nas diferentes modalidades, com a exceção da seleção por latência. A e W conjuntamente participam da composição do índice de dificuldade ID da Lei de Fitts; dessa forma, esse resultado sugere que o desempenho do mouse e das modalidades do SAPOC e do MAGIC Pointing sejam similares, e que a seleção por latência apresente alguma diferença em função da variação do índice de dificuldade.

Para verificar esse fato, agrupamos para cada modalidade os dados obtidos no experimento em pares ordenados (ID, T), onde T é o tempo gasto para o apontamento e seleção do círculo

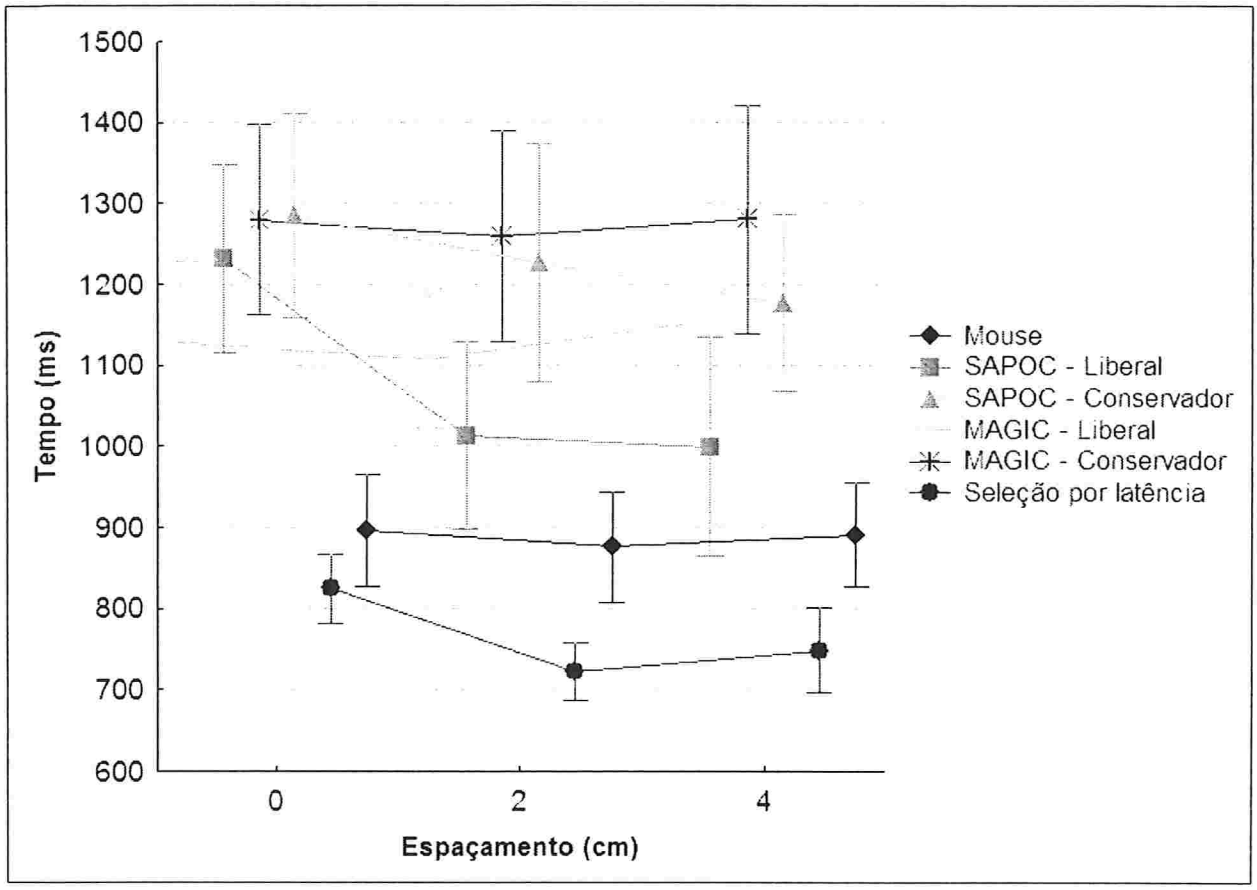


Figura 5.6: Gráfico: Tempo de execução da tarefa em função do espaçamento entre círculos. As barras verticais indicam o intervalo de confiança de 95% para a média em cada modalidade.

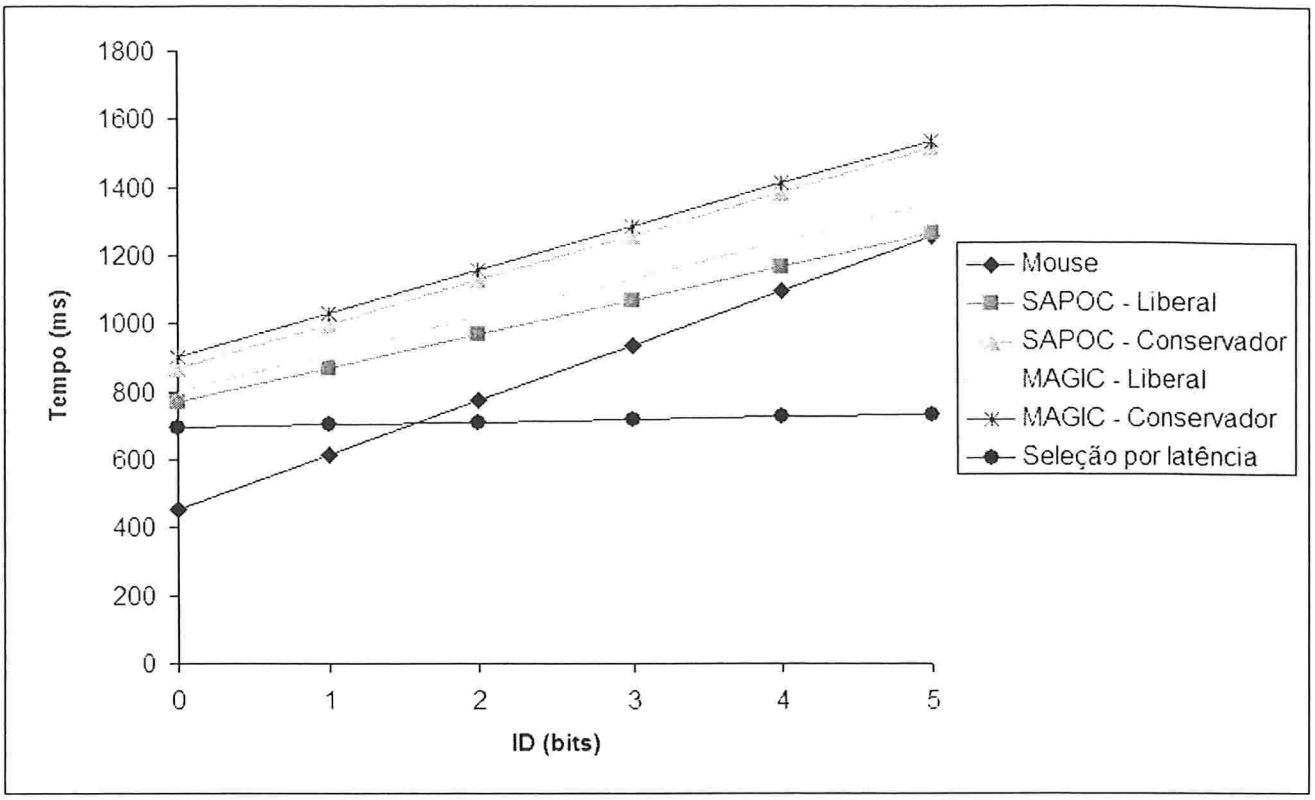


Figura 5.7: Gráfico: retas $T = a + b.ID$ para cada modalidade

com índice de dificuldade ID. Através desses pares, calculamos através de regressão linear os coeficientes a e b da reta que representa a Lei de Fitts (equação 5.2).

Esses coeficientes são exibidos na tabela 5.1, e as retas ajustadas estão no gráfico da figura 5.7. Vemos que, conforme já sugerido pelos resultados da análise de variância, as retas para o MAGIC Pointing, o SAPOC e o mouse apresentam valores próximos. O mouse apresentou o pior índice de desempenho dentre as modalidades testadas, com 6.2 bits/s. Em

Modalidade	a (ms)	b (ms/bit)	IP (bits/s)
Mouse	455	160	6.2
SAPOC Liberal	769	99	10.1
SAPOC Conservador	868	129	7.7
MAGIC Liberal	800	110	9.1
MAGIC Conservador	901	127	7.8
Seleção por latência	694	8	125

Tabela 5.1: Coeficientes de regressão linear para $T = a + b.ID$, onde $ID = \log_2 A/W + 1$

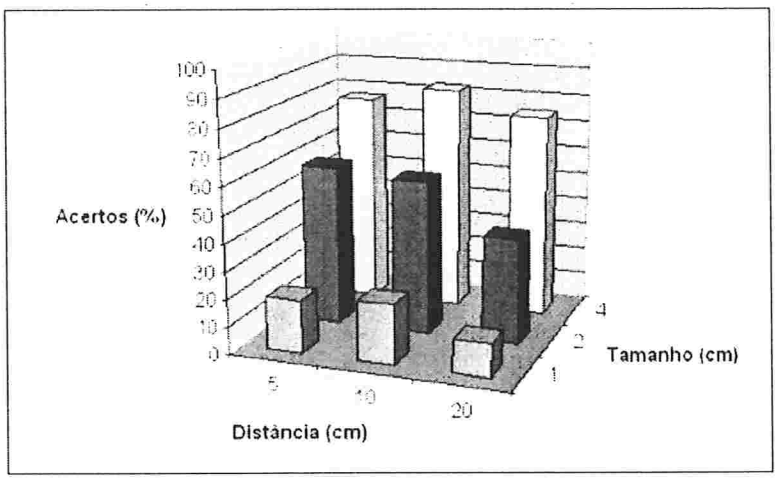
experimentos similares [ZMI99, ISB91] esse resultado foi ainda pior devido à ausência da aceleração do cursor. Para uma verificação adicional da validade desse resultado, solicitamos posteriormente que 6 dos usuários repetissem o bloco do experimento realizado com o uso do mouse, mas dessa vez desativando a aceleração do cursor. Obtivemos a reta $T = 520+210.ID$, com $IP = 4,7$ bits/s, que é um resultado similar aos outros estudos citados, onde foram obtidos índices de desempenho para o mouse entre 4 e 5 bits/s.

A reta ajustada para a seleção por latência apresentou uma inclinação pequena, e portanto um índice de desempenho alto em relação às outras modalidades (125 bits/s). Esse resultado mostra que, nessa modalidade, o tempo necessário para apontamento e seleção mantém-se praticamente constante em função do aumento do índice de dificuldade do alvo. Esse resultado é esperado, já que a seleção por latência baseia-se no movimento balístico do olho para efetuar o apontamento (veja seção 2.1), e o tempo gasto para seleção é constante. As modalidades do MAGIC e do SAPOC apresentaram índices de desempenho próximos, sendo a modalidade conservadora do SAPOC aquela que apresentou pior desempenho (7,7 bits/s) e a modalidade liberal do SAPOC a que apresentou maior desempenho (10,1 bits/s).

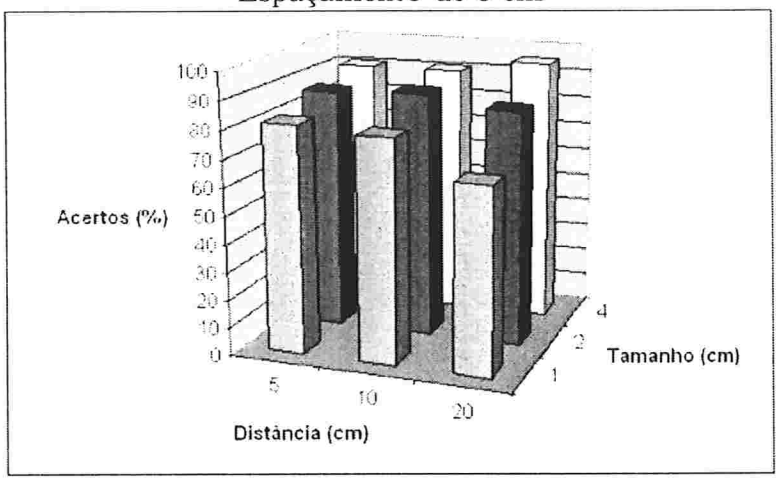
Entretanto, com o uso das informações sobre o olhar nas modalidades do SAPOC esperávamos que seus resultados, no que se refere ao índice de desempenho, exibissem um comportamento mais próximo ao da seleção por latência do que ao comportamento do mouse. O resultado que obtivemos é uma evidência que, apesar de nossos esforços de eliminar uma parte considerável do tempo gasto com o componente motor da tarefa, esse componente ainda está presente.

Para compreender esse fato nas modalidades do SAPOC analisaremos, para cada seleção efetuada, a distância d_r entre a posição do último reposicionamento do cursor antes da seleção e o centro do círculo destacado. Quando a seleção foi efetuada corretamente e $d_r > 0$, sabemos que antes de efetuá-la houve a necessidade de corrigir manualmente a posição do cursor, ou seja, mais um componente motor é adicionado à tarefa. Para identificar quando esse reposicionamento foi necessário, calculamos a porcentagem de seleções onde $d_r = 0$, e esse valor será interpretado como a *taxa de acerto* das modalidades do SAPOC.

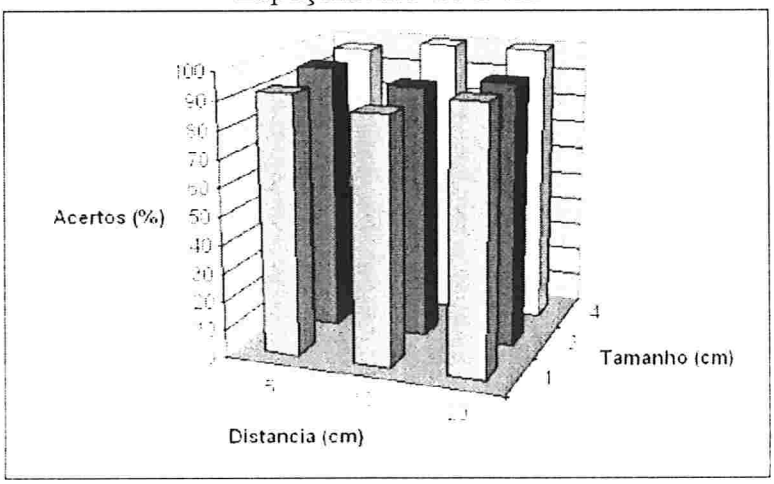
Nas figuras 5.8 e 5.9 exibimos a taxa de acerto das modalidades liberal e conservadora do SAPOC para cada combinação das distâncias do apontamento e tamanhos de círculos



Espaçamento de 0 cm

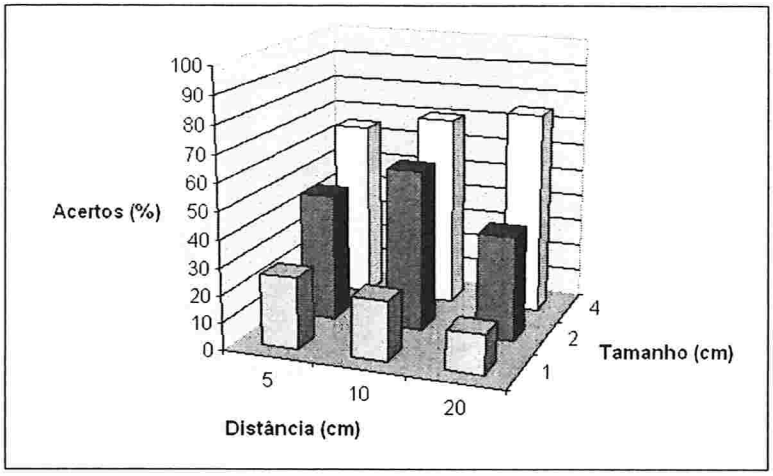


Espaçamento de 2 cm

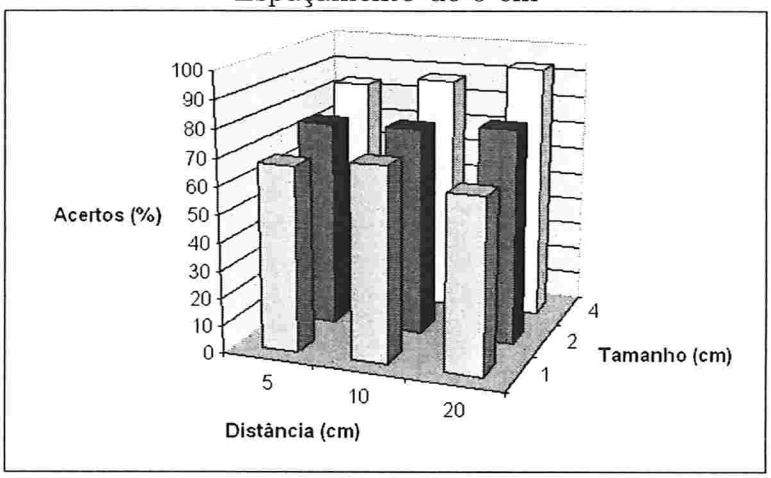


Espaçamento de 4 cm

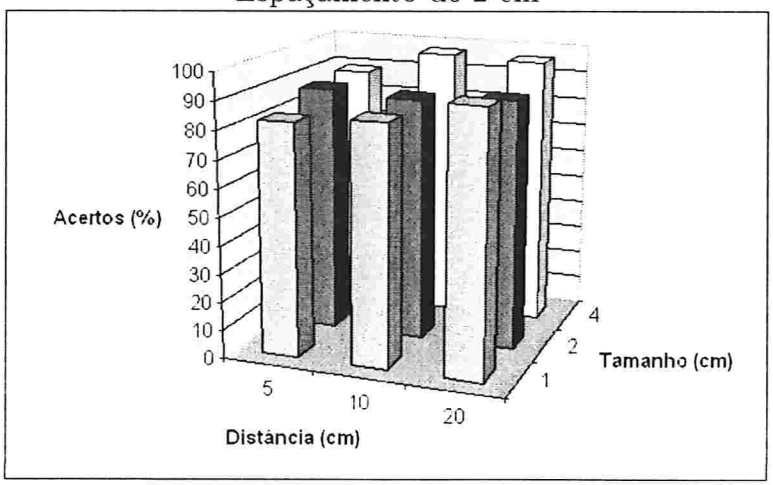
Figura 5.8: Gráfico: Modalidade liberal do SAPOC - taxas de acerto no reposicionamento do cursor para espaçamento 0, 2 e 4 cm



Espaçamento de 0 cm



Espaçamento de 2 cm



Espaçamento de 4 cm

Figura 5.9: Gráfico: Modalidade conservadora do SAPOC - taxas de acerto no reposicionamento do cursor para espaçamento 0, 2 e 4 cm

utilizados no experimento. Note que para melhor visualização agrupamos as distâncias do apontamento nas três classes (5, 10 e 20 cm) correspondentes à medida originalmente usada a cada passo do algoritmo que descrevemos na seção 5.2.2.

A taxa de acerto do reposicionamento do cursor se mantém abaixo de 50% na maioria das configurações quando o espaçamento entre os círculos é de 0 cm; nas configurações em que o espaçamento é de 2 cm, a eficácia do reposicionamento aumenta, chegando a no mínimo 60% das seleções, e a no mínimo 80% das seleções quando o espaçamento é de 4 cm. Note também que, à medida que o espaçamento G entre os círculos diminui, a taxa de acerto para os círculos menores ($W = 1$) cai rapidamente, chegando a menos de 30% das seleções quando $G = 0$ cm. Em contrapartida, quando o espaçamento aumenta, o tamanho do alvo passa a ter menor influência na taxa de acerto; para $G = 4$ cm, as taxas de acerto são semelhantes em todas as situações apresentadas no experimento.

Sabemos que o limite da precisão do sistema de rastreamento é de cerca de um grau do ângulo visual; levando em consideração essa limitação, vemos que um espaçamento menor entre os alvos aumenta a chance de erros no reposicionamento do cursor. Isso ocorre porque o cursor é reposicionado sobre o círculo mais próximo da última fixação identificada, mas essa identificação está sujeita ao erro do processo de rastreamento. Quando mais alvos estão presentes na vizinhança da margem de erro do rastreador de olhar, maior a chance do reposicionamento ser feito sobre o círculo incorreto, introduzindo aí a necessidade do ajuste manual antes de efetuar a seleção.

Esse componente motor, introduzido em uma parcela das execuções da tarefa, influencia os valores para o índice de desempenho obtido para as modalidades do SAPOC. Para verificar esse fato, recalculamos os coeficientes da Lei de Fitts dessas modalidades. Para cada configuração de espaçamento entre os círculos, fizemos o cálculo considerando tanto o conjunto dos dados quanto apenas as seleções onde o reposicionamento do cursor foi correto ($d_r = 0$), ou seja, quando o cursor foi reposicionado sobre o círculo que o usuário efetivamente desejava selecionar. Esses resultados estão expressos na tabela 5.2 e nas figuras 5.10, 5.11 e 5.12 onde vemos que, quando consideramos as seleções onde o reposicionamento foi feito corretamente, o desempenho do SAPOC melhora, agora destacando-se especialmente em relação ao desempenho do MAGIC Pointing. Notamos que os índices de desempenho chegam a 71.4 bits/s na

	Espaçamento = 0 cm					
	Conjunto dos dados			Reposicionamentos corretos		
Modalidade	a (ms)	b (ms/bit)	IP (bits/s)	a (ms)	b (ms/bit)	IP (bits/s)
SAPOC Liberal	700	170	5,9	668	85	11,8
SAPOC Conservador	887	144	6,9	897	52	19,2

	Espaçamento = 2 cm					
	Conjunto dos dados			Reposicionamentos corretos		
Modalidade	a (ms)	b (ms/bit)	IP (bits/s)	a (ms)	b (ms/bit)	IP (bits/s)
SAPOC Liberal	730	66	15,2	751	28	35,8
SAPOC Conservador	837	116	8,7	827	80	12,5

	Espaçamento = 4 cm					
	Conjunto dos dados			Reposicionamentos corretos		
Modalidade	a (ms)	b (ms/bit)	IP (bits/s)	a (ms)	b (ms/bit)	IP (bits/s)
SAPOC Liberal	805	37	27,0	778	14	71,4
SAPOC Conservador	880	76	13,2	856	75	13,3

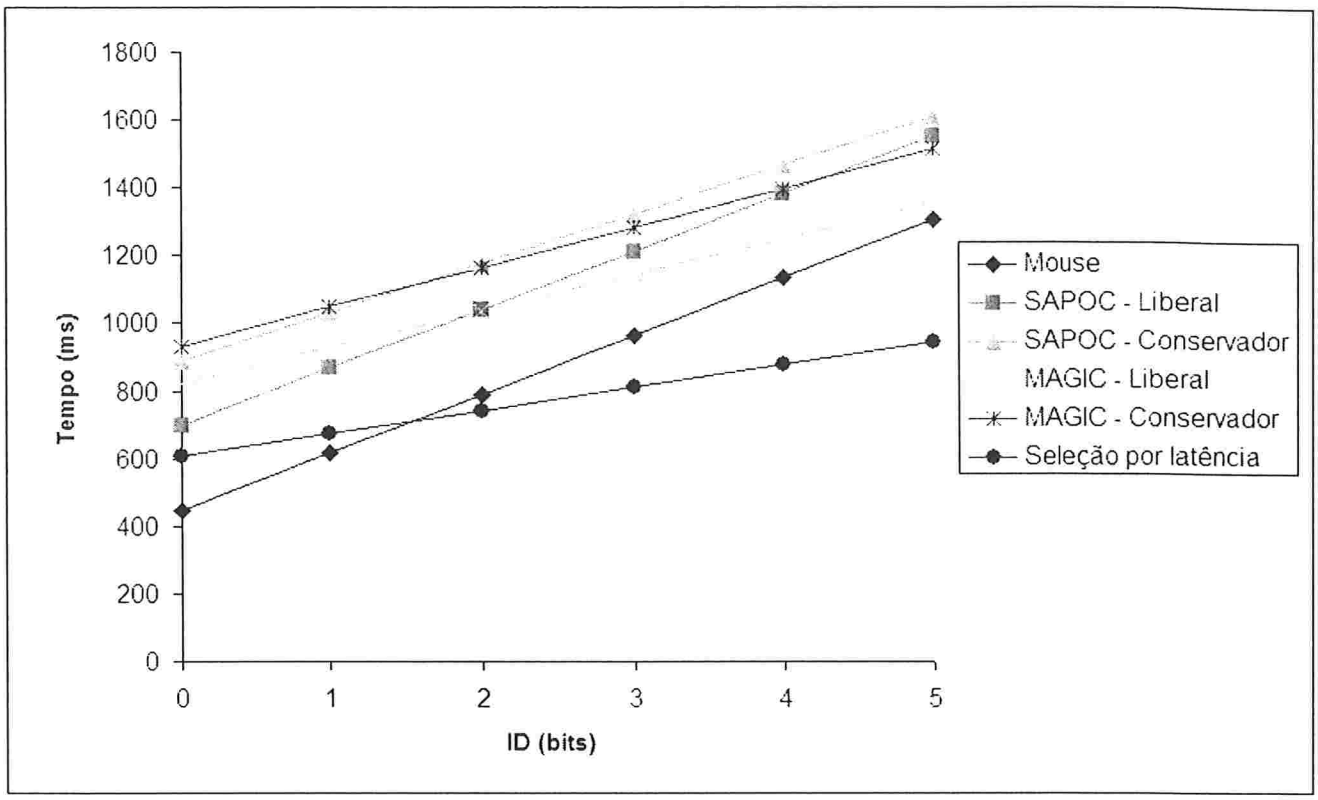
Tabela 5.2: Coeficientes de regressão linear para o SAPOC

modalidade liberal para alvos com espaçamento de 4 cm entre alvos.

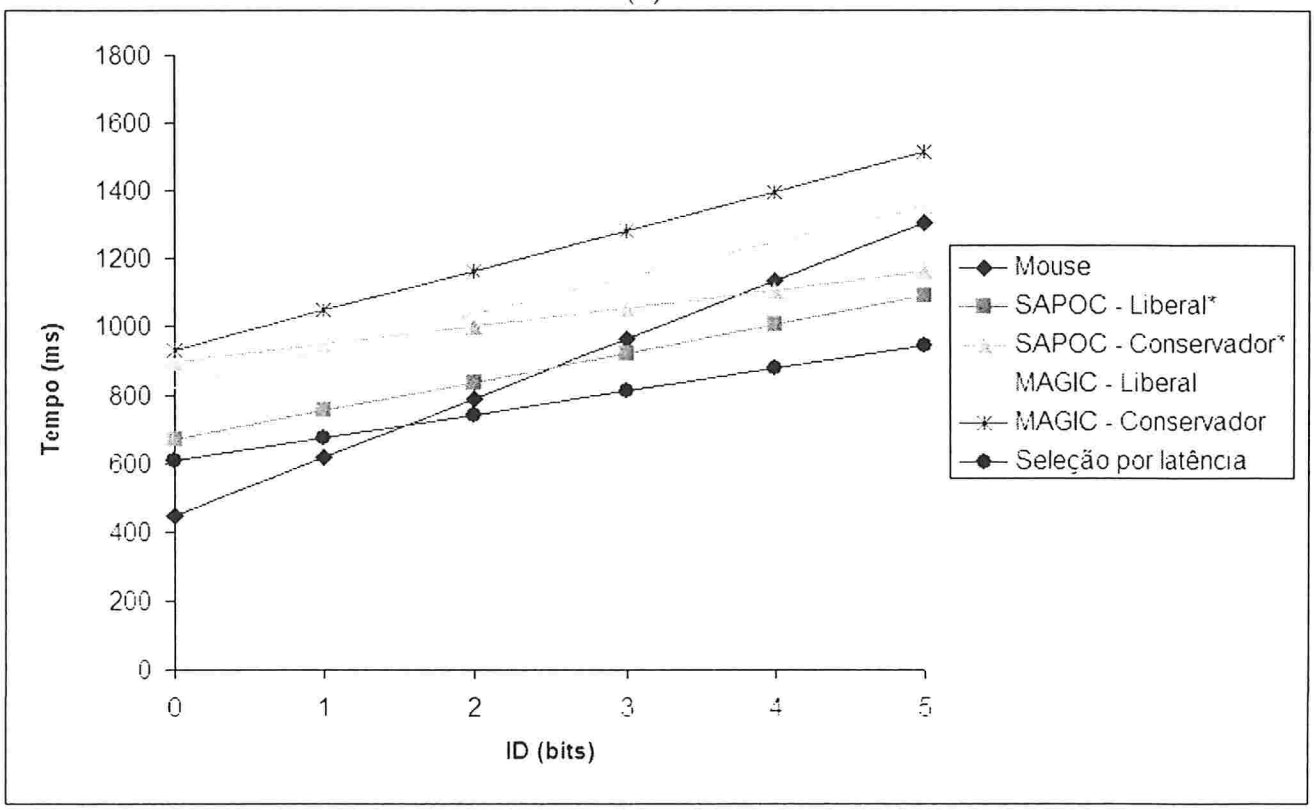
Evidentemente, como esse resultado reflete uma parcela das execuções da tarefa, ele deve ser avaliado em conjunto com a taxa de acerto do reposicionamento para cada configuração. Enquanto que, para configurações com $G = 0$ cm, esse cálculo descreve o funcionamento em menos de 50% das situações, para configurações com $G = 4$ cm o aumento do índice de desempenho reflete os resultados de pelo menos 80% das execuções da tarefa.

Quando a disposição dos alvos na interface permite contornar a limitação imposta pela precisão do rastreamento do olhar, a inferência para o reposicionamento do cursor atinge uma taxa de acerto mais elevada, reduzindo mais ainda a necessidade do ajuste manual do cursor. Dessa forma, o SAPOC explora essa condição favorável, minimizando a distância do cursor ao alvo, ao contrário do MAGIC Pointing. Como este não implementa nenhum mecanismo de inferência, o cursor é reposicionado sempre na vizinhança do alvo e esse reposicionamento fica, em todas as situações, sujeito à imprecisão do rastreamento.

Note que valores de G superiores a 2 cm correspondem, na nossa configuração, a mais de

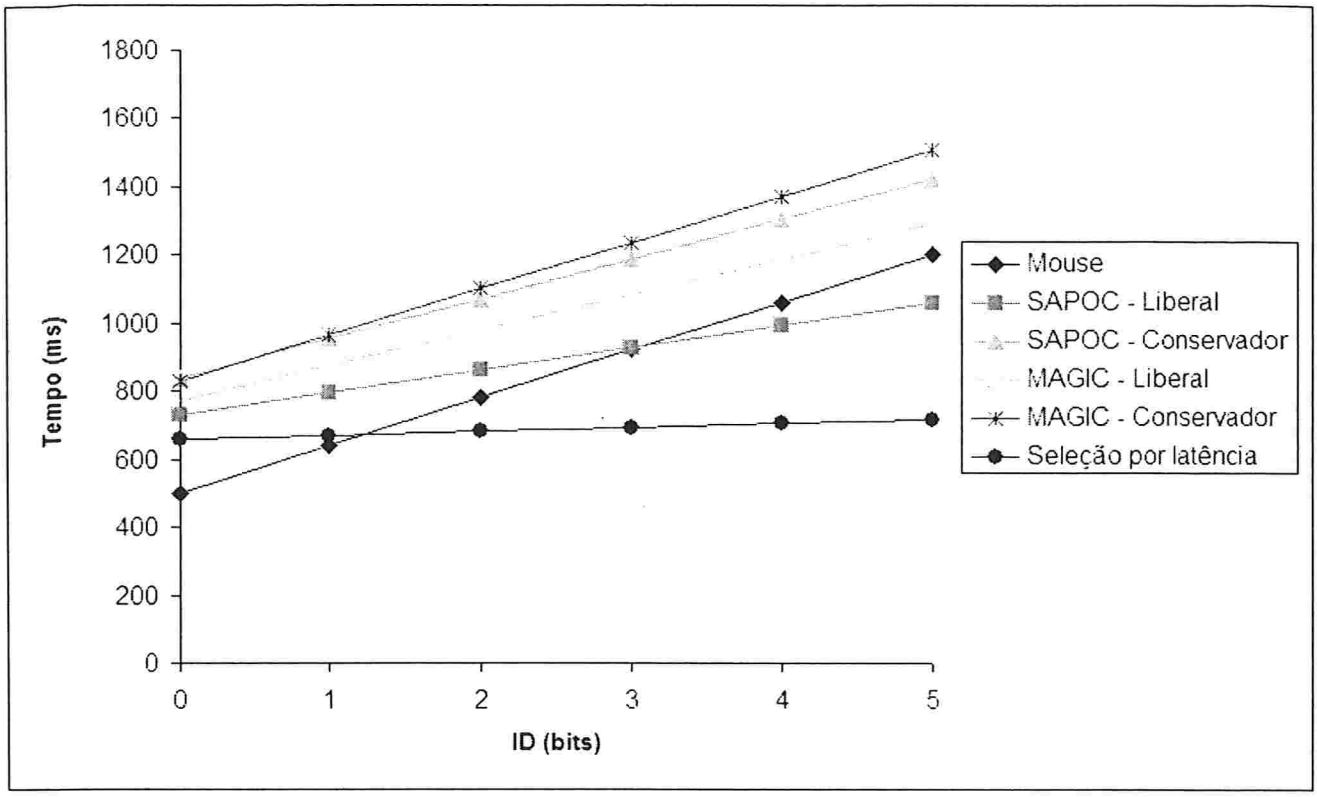


(a)

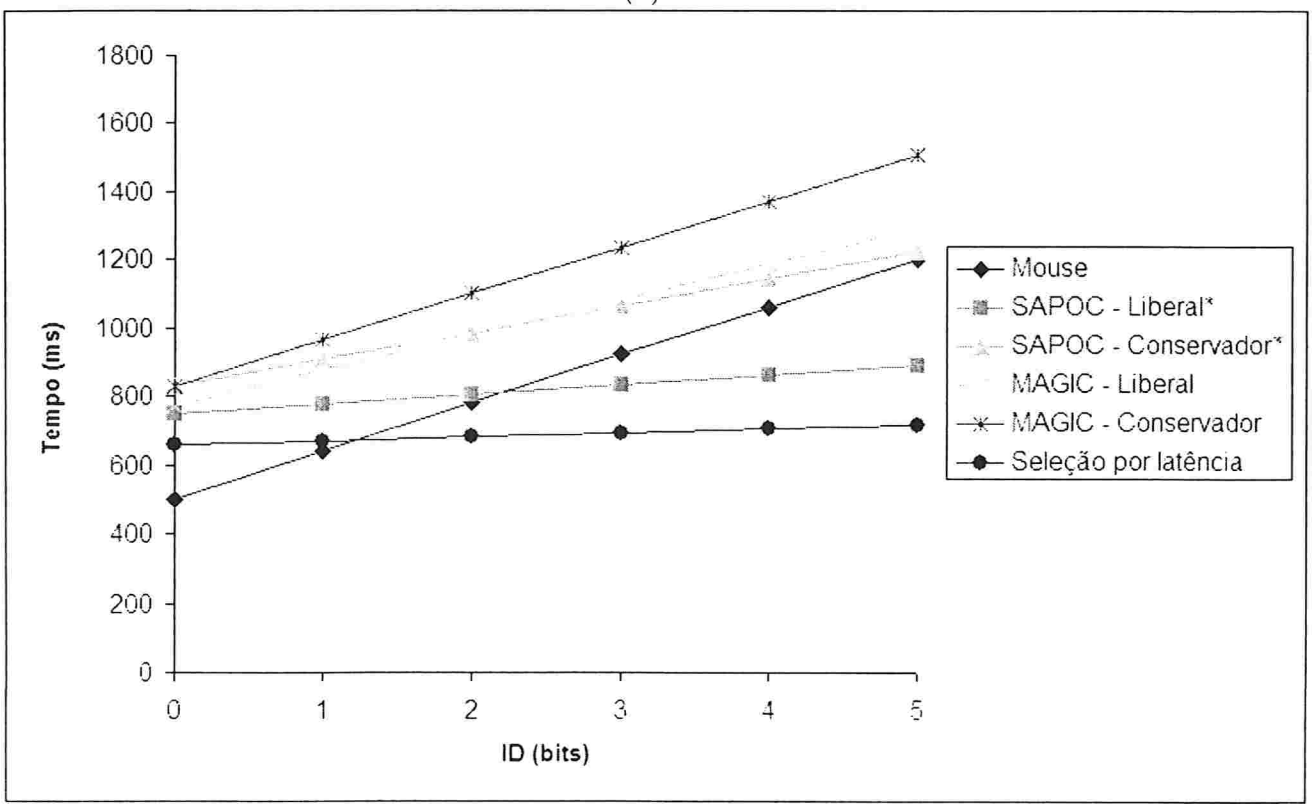


(b)

Figura 5.10: Gráfico: retas $T = a + b.ID$ para cada modalidade, para círculos com espaçamento 0 cm, considerando: (a) todas as seleções feitas com o SAPOC; (b) apenas as seleções feitas com o SAPOC a partir de reposicionamentos corretos

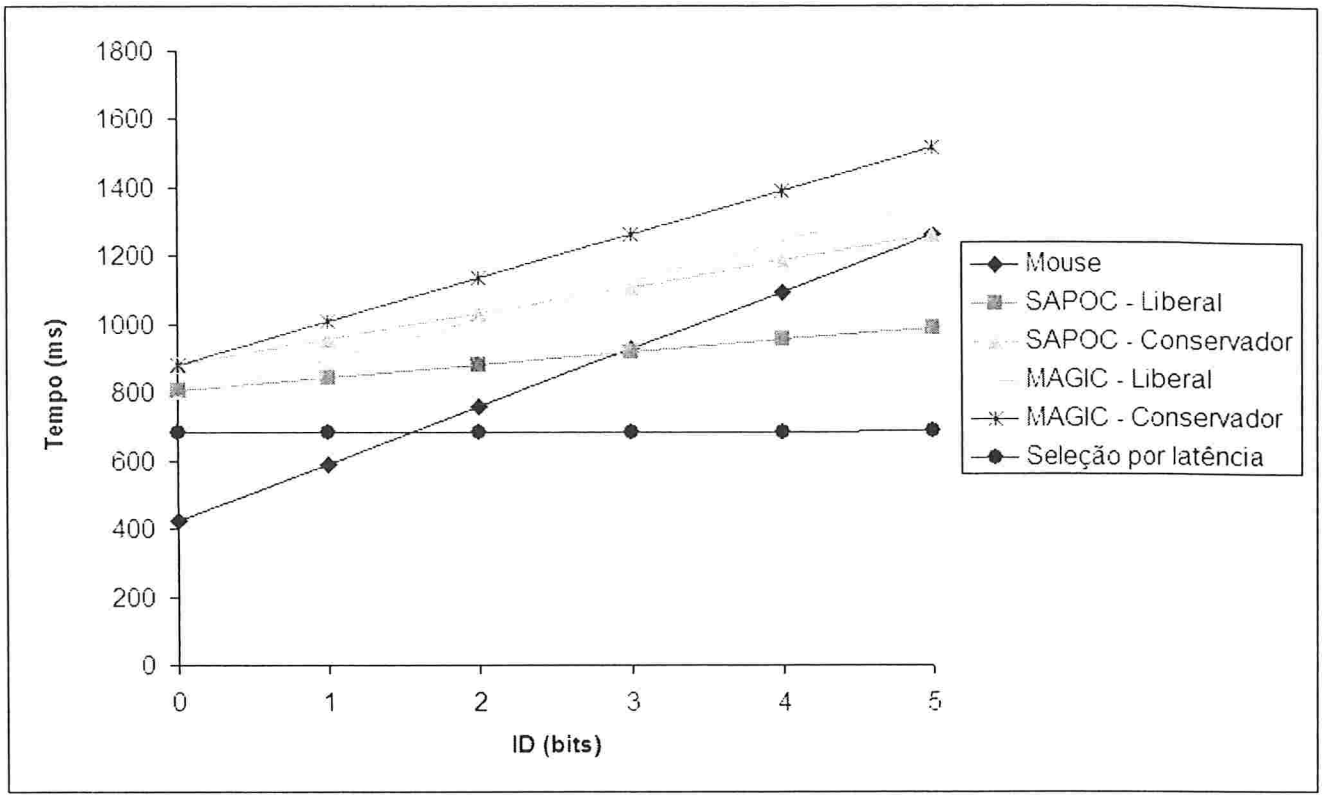


(a)

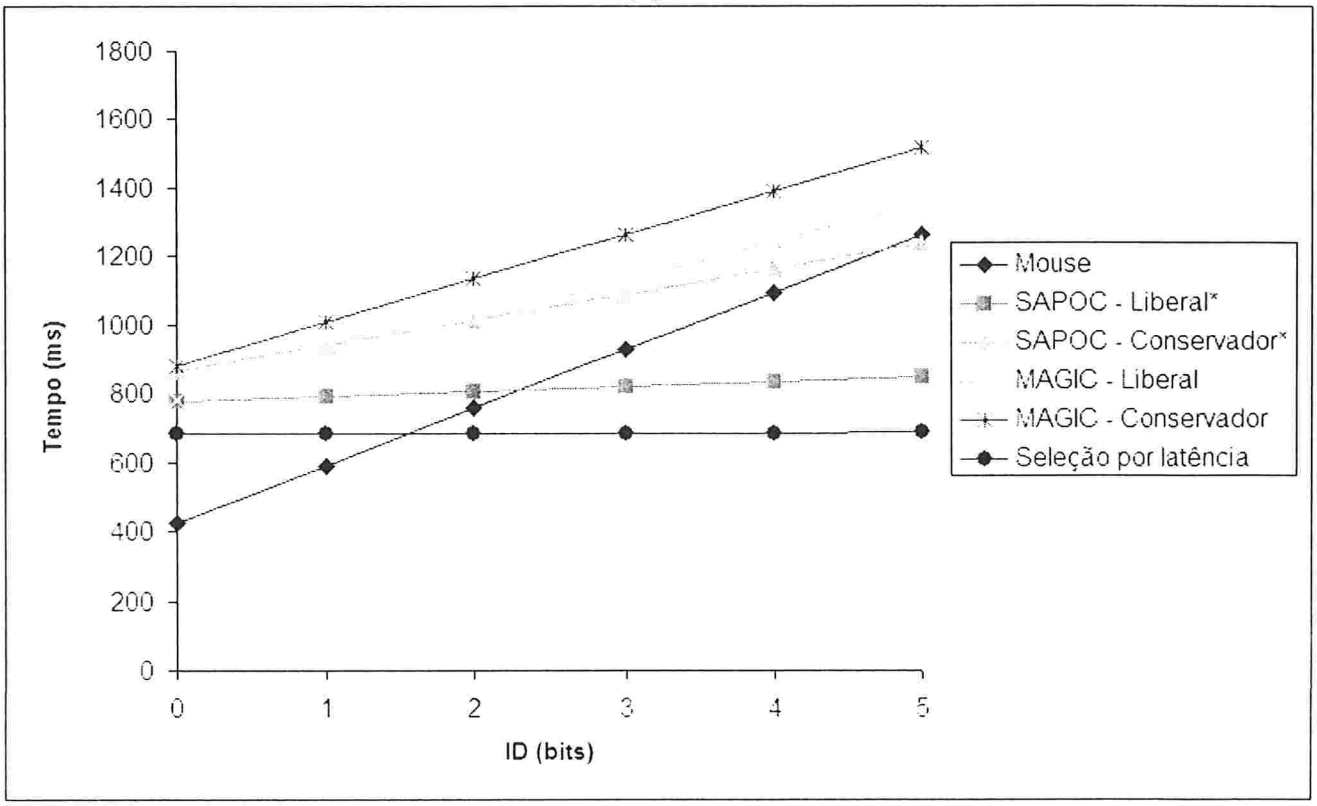


(b)

Figura 5.11: Gráfico: retas $T = a + b.ID$ para cada modalidade, para círculos com espaçamento 2 cm, considerando: (a) todas as seleções feitas com o SAPOC; (b) apenas as seleções feitas com o SAPOC a partir de reposicionamentos corretos



(a)



(b)

Figura 5.12: Gráfico: retas $T = a + b.ID$ para cada modalidade, para círculos com espaçamento 4 cm, considerando: (a) todas as seleções feitas com o SAPOC; (b) apenas as seleções feitas com o SAPOC a partir de reposicionamentos corretos

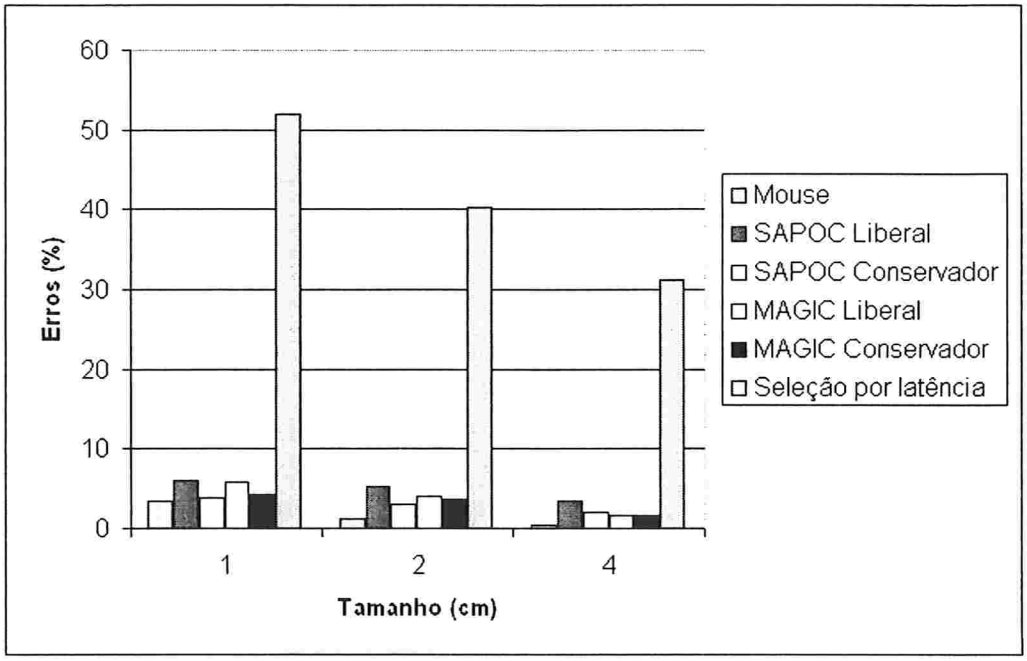


Figura 5.13: Gráfico: porcentagem de seleções incorretas em função do tamanho do círculo

2 graus do ângulo visual, que é ligeiramente superior ao valor da precisão do rastreador. O mesmo raciocínio se aplica ao tamanho dos círculos: quando $W = 1\,\text{cm}$, esse valor é próximo a um grau do ângulo visual, e vemos que as taxas de acerto do SAPOC diminuem em todas as configurações de G.

O tamanho do alvo também influi na ocorrência de seleções incorretas. Na figura 5.13 exibimos a porcentagem de seleções incorretas feitas por todos os usuários em relação ao total de seleções de círculos de tamanho 1, 2 e 4 cm. Vemos que, em todas as modalidades, há uma redução na quantidade de erros à medida que o tamanho do alvo aumenta. Mesmo com o mouse essa redução acontece, o que sugere que a quantidade de erros está relacionada principalmente com a dificuldade de apontar alvos menores, e não somente com a precisão do sistema de rastreamento.

Destaca-se também a grande ocorrência de erros nas seleções feitas com o uso da seleção por latência, chegando a mais de 50% das tentativas para alvos de tamanho 1 cm. Esse resultado mostra mais um aspecto do "Toque de Midas" dessa modalidade. Quando a precisão do rastreador provoca posicionamentos incorretos do cursor, o curto tempo de latência faz com que o círculo incorreto seja selecionado. A ausência do ajuste manual do cursor impede

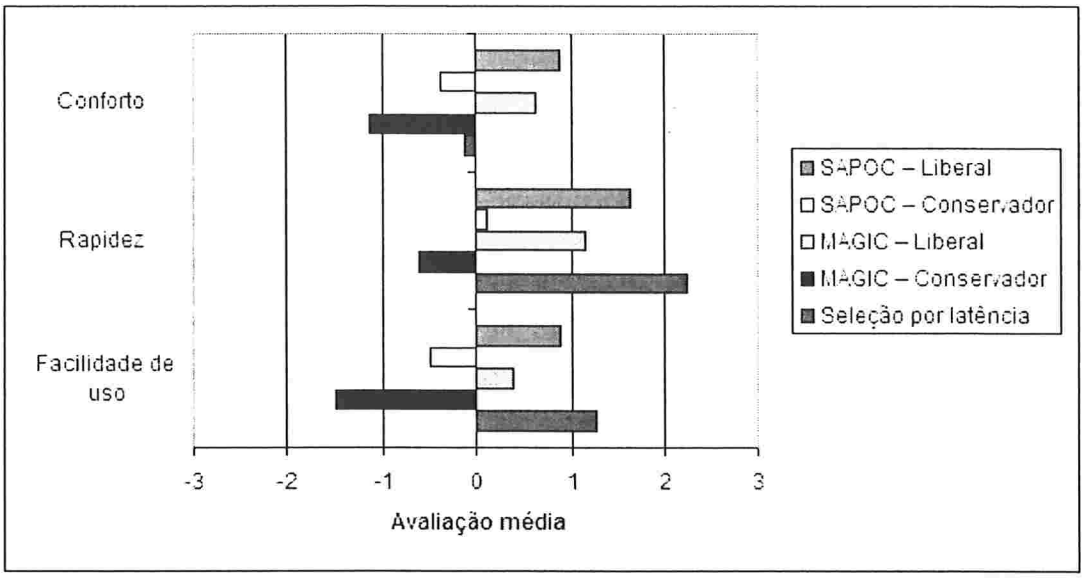


Figura 5.14: Gráfico: avaliação subjetiva

a possibilidade de corrigir o erro do reposicionamento, resultando assim nesse grande número de seleções incorretas.

5.3.1 Avaliação subjetiva

Na figura 5.14 exibimos a avaliação feita pelos usuários de todas as modalidades, em relação ao mouse, nos quesitos *Facilidade de uso*, *Rapidez* e *Conforto*. As maiores notas da avaliação foram dadas à seleção por tempo de latência. Essa modalidade foi a mais bem avaliada em facilidade de uso em relação ao mouse, obtendo uma avaliação média de 1,25, e sua melhor avaliação foi em rapidez (média de 2,25). Os tempos baixos obtidos nessa modalidade foram percebidos pelos usuários e se refletiram em sua avaliação; além disso, evidentemente para usá-la não é necessário aprender nenhuma habilidade motora adicional – basta olhar para o alvo. Porém, a avaliação no quesito conforto foi negativa. Durante o experimento, vários usuários se queixaram de aspectos dessa modalidade relativos ao "Toque de Midas", como a impossibilidade de corrigir o posicionamento incorreto do cursor antes do final do tempo de latência e da conseqüente seleção de um círculo incorreto. Esse problema se refletiu no grande número de erros registrados com o uso dessa modalidade, como já vimos anteriormente.

A modalidade liberal do SAPOC e a modalidade liberal do MAGIC Pointing receberam avaliações positivas nos três quesitos, com uma pequena vantagem para a primeira. As modalidades conservadoras receberam avaliações negativas; a única avaliação ligeiramente positiva foi para a modalidade conservadora do SAPOC no quesito rapidez. No uso dessas modalidades, vários usuários comentaram que sentiram dificuldades em coordenar a ação de olhar para o alvo com o movimento do mouse para indicar a intenção de efetivamente deslocar o cursor. Em particular, a avaliação da modalidade conservadora do MAGIC Pointing foi a pior no quesito conforto (-1,13). No quesito facilidade de uso, enquanto que as modalidades liberais apresentaram avaliações médias positivas (0,38 e 0,88 respectivamente para o MAGIC e o SAPOC), as modalidades conservadoras apresentaram avaliações médias negativas (-1,5 e -0,5).

A política de reposicionamento do cursor na modalidade liberal, na qual o cursor sempre se desloca para a vizinhança da posição observada, foi vista como mais fácil de usar e, ao contrário do que poderia se esperar, não foi avaliada como um fator gerador de desconforto. Os usuários demonstraram sua preferência pelo ajuste do cursor para o centro do círculo mais próximo: as modalidades do SAPOC foram sempre melhor avaliadas, em todos os quesitos, do que as respectivas modalidades do MAGIC Pointing.

5.3.2 Conclusões e investigações pós-experimento

A nossa análise dos resultados do experimento através da Lei de Fitts mostrou que, quando o reposicionamento do cursor ocorre corretamente, o SAPOC apresenta melhores índices de desempenho do que o mouse e do que o MAGIC Pointing. A partir da proporção de reposicionamentos corretos feitos pelo SAPOC, vimos que a distribuição espacial dos alvos na interface tem influência direta sobre esse resultado: o desempenho do sistema é melhor para alvos maiores, e para alvos com maior espaçamento para os alvos ao seu redor.

Apesar do resultado positivo para o SAPOC em relação ao índice de desempenho, esse fato não implicou em tempos inferiores aos do mouse na execução da tarefa. Tomemos os valores do coeficiente a, que modela a parte da tarefa cuja duração é constante, ou seja, cuja duração independe da variação de ID: enquanto o mouse apresenta um valor de 455

milissegundos para a constante, as modalidades do MAGIC Pointing, do SAPOC e a seleção por latência tem valores entre 700 e 870 milissegundos.

Que fatores relacionados à realização da tarefa com uma técnica de interação baseada em rastreamento do olhar podem explicar esse resultado? Após a execução do nosso experimento, investigamos dois desses prováveis fatores. Inicialmente, sabemos que ao desenvolver uma nova técnica de interação, todos os usuários são novatos. Apesar dos participantes terem passado por uma fase inicial de treinamento de duração máxima livre, ainda devemos considerar que trata-se da sua primeira experiência com a nova técnica, e esse fator pode influenciar no desempenho na realização da tarefa. Além disso, há uma latência envolvida no processo de detecção de fixações pela nossa máquina de estados que não existe no uso apenas do mouse. Na seção 4.1.2 mencionamos a nossa decisão de projeto de usar o histórico das últimas 6 imagens capturadas e processadas pelo rastreador de olhar para reconhecer a ocorrência de uma fixação, de forma a obter uma amostragem representativa do comportamento do sistema nos últimos instantes.

Após obter os resultados do experimento, decidimos testar a influência desses fatores através de duas alterações, uma no experimento e outra no sistema de rastreamento. Em primeiro lugar, decidimos reduzir o intervalo de análise da detecção de fixações para as últimas 4 imagens capturadas e processadas. Sacrificando uma parte do tamanho da amostra disponível, buscamos chegar à quantidade mínima de dados necessária para manter o funcionamento do reconhecedor de fixações estável e ao mesmo tempo diminuir a latência do sistema.

Além disso, após essa alteração no sistema, solicitamos a 6 usuários que já haviam participado do experimento que fizessem novamente o bloco da modalidade liberal do SAPOC, por ter sido a modalidade que apresentou os melhores índices de desempenho dentre as modalidades do SAPOC. O procedimento do experimento foi o mesmo, incluindo o bloco de treinamento livre antes do bloco de coleta de resultados. Essa segunda sessão foi realizada cerca de 3 semanas após o término da coleta de dados da primeira sessão. A participação de usuários que já haviam tido uma experiência anterior com o sistema teve como objetivo avaliar se há uma melhora de desempenho na execução da tarefa nessa situação, após uma segunda fase de treinamento. O perfil dessa segunda amostra se assemelhou à dos primeiros

	Primeira sessão			Segunda sessão		
Modalidade	a (ms)	b (ms/bit)	IP (bits/s)	a (ms)	b (ms/bit)	IP (bits/s)
SAPOC - Liberal	689	68	14,7	510	68	14,7
SAPOC - Liberal[1]	691	26	38,4	545	28	35,7

[1] Apenas reposicionamentos corretos

Tabela 5.3: Modalidade liberal do SAPOC: coeficientes de regressão linear para a primeira e segunda sessões do experimento

participantes, com uma experiência média de uso do mouse de 9,8 anos.

Devido a restrições de tempo do nosso projeto, não foi possível conduzir novamente o experimento em maior escala, incluindo todas as modalidades. Porém, os dados obtidos a partir dessa segunda amostra permitem identificar uma tendência que apóia as hipóteses que levantamos. Na tabela 5.3 comparamos os coeficientes da Lei de Fitts obtidos a partir dos dados gerados pelos 6 usuários na primeira sessão e dos dados da segunda sessão. Para a segunda sessão, obtivemos a reta $T = 510 + 68.ID$, $IP = 14,7$ para o conjunto dos dados e $T = 545 + 28.ID$, $IP = 35,7$, considerando apenas os reposicionamentos corretos feitos pelo SAPOC. Na realização de duas sessões por esse grupo de usuários, verificamos que o índice de desempenho não se altera significativamente mas há uma redução da ordem de 150 ms no valor do coeficiente a.

Com a redução no tamanho do intervalo de análise do reconhecedor de fixações, podemos prever um ganho de, no máximo, 60 ms. O ganho adicional pode ser explicado através da decomposição da tarefa do nosso experimento através do KLM, que descrevemos na seção 3.1.3. Como o índice de desempenho não se altera entre as duas sessões, podemos inferir que a parcela de tempo T_{mt} que engloba o deslocamento motor, incluindo o ajuste da posição do cursor e o pressionamento do botão do mouse, não se altera. Dado o intervalo de tempo entre as duas sessões e o tamanho do bloco do experimento, também podemos descartar uma redução significativa no tempo da busca visual T_v que poderia ser causada, por exemplo, por uma eventual memorização da seqüência. Assim, além da redução no tempo de resposta do sistema T_r, avaliamos que, tratando-se de uma segunda experiência dos usuários com o sistema, o tempo de preparação mental T_{pm} para a execução da tarefa diminui.

Já havíamos identificado que, na parcela de tempo da tarefa dependente da dificuldade de adquirir o alvo, o SAPOC exibia um desempenho equiparável ao do mouse, sendo superior

quando o reposicionamento do cursor ocorre com sucesso. A partir dessa pequena alteração no sistema juntamente com uma maior experiência dos usuários, pudemos identificar dois fatores que permitam reduzir o tempo de execução das tarefas de apontamente e seleção com o SAPOC, obtendo uma vantagem de desempenho em relação ao mouse.

Capítulo 6

Conclusões e trabalhos futuros

As principais contribuições deste trabalho podem ser resumidas como:

- implementação de um sistema de rastreamento do olhar;
- desenvolvimento de uma nova técnica de interação prestativa baseada no olhar, que denominamos SAPOC, que associa o olhar com informação de contexto para facilitar tarefas de apontamento e seleção;
- análise comparativa entre o SAPOC e técnicas tradicionais, inclusive outras baseadas no olhar.

No nosso sistema de rastreamento do olhar, que apóia-se integralmente em bibliotecas de código aberto, tivemos sucesso em implementar pela primeira vez alterações propostas anteriormente por Morimoto *et al.* [MKAF00], obtendo o processamento efetivo de 60 imagens por segundo através do desentrelaçamento do sinal de vídeo NTSC. A diminuição do intervalo de amostragem dos movimentos do olho permite obter o melhor desempenho possível com o nosso *hardware* de captura e provê ao Laboratório de Tecnologias de Interação (LaTIn) do IME-USP uma plataforma versátil para o desenvolvimento de técnicas de interação baseadas no olhar e também para o estudo geral de características do processo visual.

Nós estendemos o conceito introduzido pelo MAGIC Pointing, uma técnica de interação que combina o olhar com o apontamento manual, para criar uma técnica de interação que

incorpora informação sobre o contexto da interface. Assim, o SAPOC – Sistema de Apontamento Prestativo Orientado pelo Contexto – foi implementado de forma a utilizar também a posição e tamanho dos controles selecionáveis da interface. A distância de um alvo para a posição observada, reportada pelo rastreador, é usada como critério para inferir qual alvo é mais provável de ser aquele de interesse do usuário.

Avaliamos esse sistema em um experimento controlado com a participação de usuários voluntários, que não tinham experiência prévia com sistemas semelhantes. Nesse experimento, verificamos um fator limitador do desempenho do sistema: a precisão do rastreamento do olhar, equivalente a cerca de um grau do ângulo visual. Alvos pequenos e muito próximos entre si constituem a pior condição possível. Porém, nossos experimentos sugerem que essa limitação é atenuada quando aumentamos o espaçamento entre os alvos, mantendo o seu tamanho, ou vice-versa.

Nas situações onde as limitações são atenuadas, limitações essas que podem ser contornadas no desenvolvimento das interfaces, a nossa inferência sobre o alvo de interesse do usuário funciona com sucesso em boa parte das situações. Os resultados dos nossos experimentos sugerem alguns princípios para guiar o desenvolvimento de interfaces que permitam o uso da nossa técnica de interação de forma eficiente: tais interfaces devem exibir alvos cujas dimensões ultrapassem o limite de precisão do sistema de rastreamento do olhar, e distâncias entre alvos vizinhos que respeitem esse mesmo limite.

A nossa análise através da Lei de Fitts mostrou que a parcela do tempo da tarefa dependente da dificuldade de apontar o alvo em questão cresce mais lentamente com o uso do SAPOC do que com o MAGIC ou o mouse. Posicionando o cursor sobre o alvo de interesse, o SAPOC reduz a necessidade do ajuste manual da posição do cursor devido à imprecisão do rastreamento de olhar, proporcionando assim um ganho de desempenho nessa parcela da tarefa, o que é evidenciado nos casos em que a inferência ocorre com sucesso.

Porém, a parcela do tempo da tarefa que é modelada pela Lei de Fitts como constante, ou seja, influenciada por fatores que não dependem da dificuldade de apontar o alvo, mostrou-se maior para todas as modalidades que utilizam o rastreamento do olhar em relação a essa mesma parcela de tempo na execução da tarefa com o uso do mouse. Com isso, apenas a seleção por latência apresentou tempos de execução da tarefa melhores que os do mouse de

maneira consistente.

Apontamos dois fatores que poderiam influenciar essa obtenção de valores mais altos para a parcela de tempo constante da tarefa: a latência necessária para que o rastreador de olhar reconheça a ocorrência de uma fixação e a experiência prévia com o uso do sistema. Uma investigação preliminar desses dois fatores nos forneceu alguma evidência que, acreditamos, poderá motivar estudos futuros com o objetivo de melhorar o desempenho da nossa técnica de interação.

Além disso, o desenvolvimento de técnicas de interação baseadas em rastreamento do olhar será beneficiado com a exploração em outras frentes, que procuramos expor brevemente no restante deste capítulo.

6.1 Aumento da freqüência de captura de imagens

Uma forma de diminuir a latência para o reconhecimento de fixações é incorporar ao nosso sistema uma câmera cuja freqüência de captura ultrapasse os 30 quadros por segundo convencionais do sistema NTSC. Reduzindo o intervalo entre a captura de duas imagens, que atualmente é de cerca de 30 ms, torna-se possível identificar mais rapidamente o movimento do globo ocular, buscando nos dados fornecidos pelo rastreamento um padrão como descrevemos na seção 4.1.2.

Existem sistemas comerciais de rastreamento do olhar que utilizam câmeras operando a maiores freqüências, como por exemplo aqueles desenvolvidos pela Applied Science Laboratories [asl05], que utiliza câmeras que operam a freqüências de até 350 quadros por segundo. Torna-se necessário nesse caso avaliar a viabilidade de adaptar a iluminação do nosso sistema de rastreamento a uma câmera mais rápida, ou mesmo avaliar o uso do SAPOC utilizando outros rastreadores de olhar. A influência do dispositivo de rastreamento de olhar no desempenho da nossa técnica de interação poderá ser avaliado através de novos experimentos como os descritos no capítulo 5.

6.2 Inferência dinâmica para reposicionar o cursor

O SAPOC utiliza o tamanho e o posicionamento dos controles na interface, juntamente com os dados provenientes do rastreamento do olhar, para reposicionar o cursor. O nosso critério para inferir o alvo de interesse do usuário é simples: assumimos que o alvo mais próximo da posição observada é o objeto de interesse do usuário. Em nosso experimento, vimos que em configurações com alvos maiores e mais distantes entre si, essa inferência consegue reposicionar o cursor corretamente (isto é, sobre o alvo que efetivamente o usuário desejava apontar e selecionar) em até 80% das seleções.

No uso cotidiano dessa técnica de interação, seria desejável que o reposicionamento acontecesse com sucesso em 100% das situações, ou que pelo menos a inferência não atuasse em situações que possam gerar um reposicionamento incorreto. Nossos resultados sugerem algumas formas para melhorar ainda mais a estratégia de reposicionamento do cursor.

Por exemplo, quando na vizinhança da região observada há uma grande densidade de alvos, ou quando os alvos são muito pequenos, o desempenho do sistema se degrada porque a inferência incorreta exige que o usuário precise efetuar um ajuste manual ainda maior. Nessas situações, o SAPOC poderia identificar uma situação potencialmente desfavorável e não tentar localizar o alvo mais próximo, simplesmente utilizando a posição observada para reposicionar o cursor, como o MAGIC Pointing. Outra possibilidade seria usar inicialmente a posição observada como o centro de uma espécie de "lente de aumento", ampliando momentaneamente as dimensões dos alvos na vizinhança da posição observada de forma a minimizar os problemas gerados pela falta de precisão do rastreador de olhar.

6.3 Estudo da tarefa de apontamento e seleção

Na análise dos resultados do nosso experimento, verificamos que o desempenho dos usuários na execução da tarefa de apontamento e seleção tem, em algumas situações, estrita relação com a forma de executar essa tarefa. Por exemplo, analisando o desempenho das modalidades do SAPOC em relação à Lei de Fitts, verificamos que seu desempenho mostrou-se muito semelhante ao obtido com o mouse quando consideramos tanto as seleções

em que o cursor foi reposicionado corretamente quanto aquelas em que o cursor foi reposicionado sobre um alvo incorreto. Já quando a análise incluía somente as seleções em que o reposicionamento do cursor foi correto, o desempenho do SAPOC aproximava-se mais do desempenho da seleção por latência, em que os tempos de execução da tarefa são aproximadamente constantes em função da dificuldade de apontar o alvo.

Uma hipótese razoável para esse fato é que, naquelas situações em que foi preciso corrigir a posição do cursor antes da seleção do alvo, o usuário teve que executar duas subtarefas adicionais. A primeira delas é relocalizar visualmente a posição para a qual o cursor foi reposicionado. Evidentemente, pela construção de nossa técnica de interação e pela precisão do rastreador de olhar, sabemos que o cursor estará sempre próximo do foco de atenção visual do usuário, dentro da margem de erro do rastreador. Porém, do ponto de vista do usuário, a posição do cursor nesse caso apresenta um desvio aleatório em relação ao posicionamento esperado. A segunda subtarefa consiste em reposicionar manualmente o cursor, levando-o até o alvo correto.

Podemos esperar que o tempo necessário para esse ajuste manual seja modelado pela Lei de Fitts, onde a amplitude do deslocamento é expressa em função da magnitude do desvio introduzido pela imprecisão do rastreamento de olhar e pelo reposicionamento incorreto do cursor pelo SAPOC. Porém, para uma melhor compreensão dos resultados do nosso experimento, pode-se investigar futuramente qual é o tempo gasto para localizar o cursor após um reposicionamento incorreto.

Para tanto, propomos um segundo experimento, cujo objetivo estaria mais próximo de um teste de reação psicofísica do que de um experimento de usabilidade: investigar o tempo necessário para a busca visual do cursor em função da sua distância para um alvo que recebe o foco da atenção visual. Exibe-se um alvo circular no centro do vídeo, e após o disparo de um contador de tempo, o cursor é exibido a uma distância aleatória do centro do alvo; essa distância é igualmente distribuída no intervalo de 0 ao limite empírico da precisão do nosso rastreador de olhar, ou seja, um grau do ângulo visual. Quando o usuário localiza visualmente o cursor, ele indica esse evento pressionando um botão em um dispositivo. Esse dispositivo deve reportar ao sistema o pressionamento com a menor latência possível; talvez seja suficiente o uso de um *joystick* disponível no mercado.

Os resultados de um experimento desse tipo podem ajudar a definir com maior precisão como um erro introduzido pelo SAPOC colabora com o aumento do tempo necessário para executar a tarefa, talvez até contribuindo com novas propostas para melhoramentos do sistema.

6.4 Rastreamento do olhar sem calibragem

Talvez a restrição mais impeditiva ao uso de rastreadores de olhar fora de ambientes experimentais seja o desconforto causado pela necessidade de manter a cabeça parada durante o uso e de executar um procedimento de calibragem antes de cada uso do sistema. A eliminação dessas restrições pode simplificar e abrir novas possibilidades para o uso de rastreadores de olhar em sistemas interativos.

Tem acontecido avanços bastante recentes no desenvolvimento de técnicas de rastreamento do olhar que não exigem calibragem, permitem alguma movimentação da cabeça e obtém uma precisão razoável. Dentre elas podemos citar como exemplo a descrita por Yoo *et al.* [YKLC02]. Ela destaca-se pela simplicidade, baseando-se na detecção da pupila e das reflexões geradas sobre a córnea por 4 LEDs colocados nas extremidades do monitor, que servem como referência para estimar a linha de visão. A precisão dessa técnica, segundo os autores, é de 2° do ângulo visual. Beymer e Flickner [BF03] descrevem uma abordagem baseada no uso de um modelo 3D do olho e duas câmeras, cujas posições em relação ao monitor são previamente determinadas através de um processo de calibragem. Esta técnica, apesar de mais complexa, tem uma precisão um pouco melhor: $0,6°$ do ângulo visual.

Acreditamos que a incorporação de uma dessas técnicas ao nosso sistema pode ampliar de forma significativa o seu potencial de utilização prática. Apesar da precisão de várias técnicas de rastreamento do olhar sem calibragem ainda ser inferior à obtida pelas técnicas tradicionais [MM05], o potencial da nossa técnica de interação em contornar a imprecisão do rastreamento do olhar pode permitir que, juntamente a um rastreador de olhar sem calibragem, ela participe da construção de uma interface mais eficiente e mais natural, com desempenho superior ao obtido com o uso apenas do mouse.

Apêndice A

Questionário do experimento

Apresentamos neste apêndice o questionário que foi aplicado aos usuários participantes do experimento descrito na seção 5.2.

Obrigado por participar de nosso experimento!
Pedimos um minuto de sua atenção para preencher com clareza e honestidade o questionário abaixo.
Para cada uma das modalidades de interação que você usou durante o experimento, dê uma nota em cada um dos quesitos abaixo. A nota indicará a sua avaliação da modalidade no quesito em relação ao mouse. Assim, -3 significa "muito pior do que o mouse", 0 significa "igual ao mouse" e +3 significa "muito melhor do que o mouse"

Facilidade de uso

	↙ pior				melhor ↗		
	-3	-2	-1	0	1	2	3
A1							
A2							
B1							
B2							
C							

Rapidez

	↙ pior				melhor ↗		
	-3	-2	-1	0	1	2	3
A1							
A2							
B1							
B2							
C							

Conforto

	↙ pior				melhor ↗		
	-3	-2	-1	0	1	2	3
A1							
A2							
B1							
B2							
C							

Figura A.1: Questionário respondido pelos participantes do experimento

Apêndice B

Configuração do sistema

A tabela abaixo exibe a configuração do *software* e *hardware* utilizados no desenvolvimento do protótipo do nosso sistema de rastreamento do olhar e do SAPOC.

CPU	AMD Athlon XP 2600MHz
Memória RAM	512 Mb
Monitor	NEC Accusync 90 - 19 polegadas, área vísivel de 366 × 274 mm
Placa de captura de vídeo	Osprey 100 (chipset Brooktree BT848), ligada a barramento PCI
Mouse	Microsoft Intellimouse PS/2
Sistema Operacional	Linux Fedora Core 2 executando sistema X-Window versão 6.7.0 da X.org e ambiente gráfico KDE versão 3.2.2
Resolução lógica de vídeo	1280 × 960 pixels
Resolução física de vídeo	88 pontos por polegada

Referências Bibliográficas

[AS04] Trolltech AS. Qt toolkit. Site da Web, 2004. `http://www.trolltech.com/products/qt/index.html`. Acesso em julho de 2004.

[asl05] Applied Science Laboratories. Site da Web, Maio 2005. `http://a-s-l.com`. Acesso em 25-05-2005.

[BE04] Preston Brown and Matthias Ettrich. DCOP: Desktop COmmunications Protocol. Site da Web, 2004. `http://developer.kde.org/documentation/library/kdeqt/dcop.html`. Acesso em julho de 2004.

[BF03] David Beymer and Myron Flickner. Eye gaze tracking using an active stereo head. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '03)*, pages 451–458, Madison WI, 2003.

[BG90] V. Bruce and P. R. Green. *Visual perception: physiology, psychology and ecology (2nd edition)*. Lawrence Erlbaum Associates Ltd., 1990.

[Bol81] Richard A. Bolt. Gaze-orchestrated dynamic windows. *Computer Graphics*. 15(3):109–119. august 1981.

[BP94] S. Baluja and D. Pomerlau. Non-intrusive gaze tracking using artificial neural networks. Technical Report CMU-CS-94-102. School of Computer Science. Carnegie Mellon University. Pittsburg PA. USA. 1994.

[Car05] G. David Carson. PA 765 Statnotes: An Online Textbook - North Carolina State University. Site da Web. 2005. `http://www2.chass.ncsu.edu/garson/pa765/statnote.htm`. Acesso em junho de 2005.

[CMN80] S. K. Card, T. P. Moran, and A. Newell. The keystoke-level model for user performance time with interactive systems. *Communications of the ACM*, 23(7):396–410, 1980.

[CMN83] S. K. Card, T. P. Moran, and A. Newell. *The psychology of human-computer interaction*. Erlbaum, Hillsdale, NJ, 1983.

[Cor05] Microsoft Corporation. Windows 2000 Registry Information - MouseThreshold1 key. Site da Web, 2005. `http://www.microsoft.com/resources/documentation/Windows/2000`. Acesso em fevereiro de 2005.

[DFAB97] Alan Dix, Janet Finlay, Gregory Abowd, and Russell Beale. *Human-computer interaction*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1997.

[Dir04] Bill Dirks. Video for Linux Two. Site da Web, 2004. `http://www.thedirks.org/v4l2`. Acesso em maio de 2004.

[Duc02] Andrew T. Duchowski. A breadth-first survey of eye-tracking applications. *Behavioral Research Methods, Instruments and Computers*, pages 1–16, 2002.

[Edw98] Gregory Edwards. A tool for creating eye-aware applications that adapt to changes in user behavior. In *Proceedings of ACM ASSETS 98*, Marina del Rey, California, 1998.

[Fit54] P.M. Fitts. The information capacity of the human motor systems in controlling the amplitude of movement. *Journal of Experimental Psicology*, 1954.

[GEN95] A. Glenstrup and T. Engell-Nielsen. Eye controlled media: Present and future state. Master's thesis. Institute of Computer Science. University of Copenhagen, 1995.

[HJM+89] Thomas E. Hutchinson, K. Preston White Jr., Worthy N. Martin, Kelly C. Reichert, and Lisa A. Frey. Human-computer interaction using eye-gaze input. In *IEEE Transactions on Systems, Man and Cybernetics*, volume 19, pages 1527–1534, 1989.

[HK99] Anthony J. Hornof and David E. Kieras. Cognitive modeling demonstrates how people use anticipated location knowledge of menu items. In *Human Factors in Computing Systems: CHI 99 Conference Proceedings*, pages 410–417, 1999.

[Inc04] The XFree86 Project, Inc. X Window System. Site da Web, 2004. `http://www.xfree86.org`. Acesso em julho de 2004.

[IR02] Poika Isokoski and Roope Raisamo. Speed-accuracy measures in a population of six mice. In *Proceedings of APCHI2002 (5th Asia Pacific Conference on Computer Human Interaction)*, pages 765–777, Beijing, China, 2002. Science Press.

[ISB91] MacKenzie I.S., A. Sellen, and W. Buxton. A comparison of input devices in elemental pointing and dragging tasks. In *Proceedings of the ACM Conference on Human Factors in Computing Systems - CHI '91*, pages 161–166. ACM, 1991.

[Jac90] Robert J. K. Jacob. What you look at is what you get: eye movement-based interaction techniques. In *Proceedings of the CHI'90*, pages 11–18, 1990.

[Jac95] Robert J. K. Jacob. Eye tracking in advanced interface design. In W. Barøeld and T. Furness, editors, *Advanced Interface Design and Virtual Environments*, pages 258–288, Oxford, England, 1995. Oxford Press.

[Lan90] Edwin W. Land. The retinex theory of color vision. In Irwin Rock, editor, *Readings from Scientific American - The Perceptual World*, pages 39–62, New York, 1990. W.H. Freeman and Company.

[Lan05] David M. Lane. HyperStat Online - Rice Virtual Lab in Statistics. University of Rice, TX. Site da Web. 2005. `http://www.ruf.rice.edu/~lane/rvls.html`. Acesso em fevereiro de 2005.

[MdL99] Marcos Nascimento Magalhães and Antonio Carlos Pedroso de Lima. *Noções de probabilidade e estatística*. Departamento de Estatística do IME-USP. São Paulo. 1999.

[MKAF00] C. Morimoto, D. Koons, A. Amir, and M. Flickner. Pupil detection and tracking using multiple light sources. *Image and Vision Computing*, v.18(no.4):pp. 331–336, 2000.

[MKS01] I. Scott Mackenzie, Tatu Kaupinnen, and Miika Silfverberg. Accuracy measures for evaluating computer pointing devices. In *Proceedings of the ACM Conference on Human Factors in Computing Systems - CHI 2001*, pages 9–16. ACM, 2001.

[MM05] Carlos H. Morimoto and Marcio R. M. Mimica. Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding*, 98(1):4–24, 2005.

[MSS91] G. A. Myers, K. R. Sherman, and L. Stark. Eye monitor: Microcomputer-based instrument uses an internal mode to track the eye. In *Computer*, volume 24, pages 14–21. IEEE Computer Society Press, Los Alamitos, CA, USA, 1991.

[OC00] Sharon Oviatt and Philip Cohen. Multimodal interfaces that process what comes naturally. *Communications of the ACM*, 43(3):45–53, March 2000.

[ODH02] Carol O'Sullivan, John Dingliana, and Sarah Howlett. Eye-movements and interactive graphics. In J. Hayöna, R. Radach, and H. Deubel, editors, *The Mind's Eyes: Cognitive and Applied Aspects of Eye Movement Research*, Oxford, England, 2002. Elsevier Science.

[OMY02] Takehiko Ohno, Naoki Mukawa, and Atsushi Yoshikawa. FreeGaze: a gaze tracking system for everyday gaze interaction. In *Proceedings of the Eye Tracking Research & Application Symposium 2002 (ETRA-02)*, pages 125–132, 2002.

[Pro04] The GNOME Project. GTK+ Toolkit. Site da Web, 2004. `http://developer.gnome.org/arch/gtk`. Acesso em julho de 2004.

[Rab89] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, 1989.

[Rob63] D. A Robinson. A method of measuring eye movements using a scleral search coil in a magnetic field. In *IEEE Transactions on Biomedical Engineering 10*, pages 137–145, 1963.

[SA00] Dario D. Salvucci and John R. Anderson. Intelligent gaze-added interfaces. In *Human Factors in Computing Systems: CHI 2000 Conference Proceedings*, 2000.

[Sal99a] Dario D. Salvucci. Inferring intent in eye-based interfaces. In ACM Press, editor, *Human Factors in Computing Systems: CHI 99 Conference Proceedings*, pages 254–261, 1999.

[Sal99b] Dario D. Salvucci. *Mapping Eye Movements to Cognitive Processes*. PhD thesis, Carnegie Mellon University, 1999.

[SG00] Dario D. Salvucci and Joseph H. Goldberg. Identifying fixations and saccades in eye-tracking protocols. In *Proceedings of the Eye Tracking Research and Applications Symposium*, pages 71–78, 2000.

[SJ00] Linda E. Sibert and Robert J. K. Jacob. Evaluation of eye gaze interaction. In *Proceedings of the CHI'00*, pages 281–288, 2000.

[Tur98] Matthew Turk. Moving from GUIs to PUIs. In *Proceedings of the Symposium on Intelligent Information Media*, 1998.

[Ver02] Roel Vertegaal. Designing attentive interfaces. In *Proceedings of ACM ETRA Symposium on Eye Tracking Research and Applications 2002*. ACM Press, 2002.

[Wik05] Wikipedia. Eye – Wikipedia, the free encyclopedia. Site da Web, Maio 2005. `http://en.wikipedia.org/wiki/eye`. Acesso em 27-05-2005.

[WZS01] Jingtao Wang, Shumin Zhai, and Hui Su. Chinese input with keyboard and eye-tracking – an anatomical study. In *Proceedings of the CHI'01*, pages 349–356, 2001.

[YKLC02] Dong Hyun Yoo, Jae Heon Kim, Bang Rae Lee, and Myoung Jin Chung. Non-contact eye gaze tracking system by mapping of corneal reflections. In *Proceedings*

of the Fifth IEEE Conference on Automatic Face and Gesture Recognition, pages 94–99, Washington DC, 2002.

[YS75] L. Young and D. Sheena. Methods & designs: Survey of eye movement recording methods. *Behavioral Research Methods and Instrumentation*, 7(15):397–429, 1975.

[Zha03] Shumin Zhai. What's in the eyes for attentive input. *Communications of the ACM*, 46(3):34–39, March 2003.

[ZMI99] S. Zhai, C. Morimoto, and S. Ihde. Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI'99)*, pages 246–253, 1999.