ESCOLA POLITÉCNICA DA UNIVERSIDADE DE SÃO PAULO
DEPARTAMENTO DE ENGENHARIA DE SISTEMAS ELETRÔNICOS

Carlos Eduardo Castilho Marques
Gustavo Shimabukuro Marchini
Marcelo Victor Bomfim Gomes

PopEye – Rastreamento do olhar

São Paulo
2011

Carlos Eduardo Castilho Marques
Gustavo Shimabukuro Marchini
Marcelo Victor Bomfim Gomes

PopEye – Rastreamento do olhar

Trabalho de Formatura apresentado à Escola Politécnica da USP como parte dos requisitos da disciplina PSI2594-Projeto de Formatura II.

Área de Concentração:

Sistemas Eletrônicos

Orientador:

Prof. Fuad Kassab Junior

São Paulo
2011

RESUMO

Este projeto tem como objetivo o desenvolvimento de um sistema de rastreamento do olhar. Este sistema será dotado de um óculos devidamente equipado com uma câmera captando a imagem do olho do usuário. Por meio de um processamento de imagem e utilizando a técnica de iluminação infravermelha do olho será possível determinar para onde o usuário está olhando em tempo real. Com esta informação em mãos, é possível realizar diversas aplicações voltadas para áreas como neurociência, interface homem-máquina, usabilidade, etc. No caso deste projeto, o rastreamento do olhar será utilizado para aplicação de interface homem-máquina em que, através do olhar, o usuário conseguirá controlar o cursor do *mouse*.

Palavras-Chave: Processamento de imagem; Interface homem-máquina; *Eye-Tracking*.

ABSTRACT

This project aims to develop an eye tracking system. This system will have glasses equipped with a camera capturing the image of the user's eye. Through image processing and using the technique of infrared illumination of the eye will be possible determine where the user is looking in real time. With this information in hand, it's possible perform various applications related to areas such as neuroscience, human-machine interface, usability, etc. For this project, eye tracking will be used for application of human-machine interface in which, through the eyes, the user will be able to control the mouse cursor.

Key-words: Image Processing; Human-Machine Interface; Eye-Tracking.

LISTA DE ABREVIATURAS

ASL	*Applied Science Laboratories.*
DAQ	*Data Aquisition.*
DSP	*Digital Signal Processing.*
EEPROM	*Electrically-Erasable Programmable Read-Only Memory.*
ELA	Esclerose Lateral Amiotrófica
EOG	Eletro-oculografia.
FPGA	*Field Programmable Gate Array*
HTML	Hiper Text Mark-up Language
LED	*Light Emitting Diode.*
MMC	*Multi Media Card.*
OpenCV	*Open Source Computer Vision.*
RANSAC	*Random Sample Consensus*
TID	Transtorno Invasivo de Desenvolvimento.
USB	*Universal Serial Bus.*

LISTA DE FIGURAS

Figura 1 – Heat map (mapa de calor) incluindo os primeiros 20 segundos de olhar de oito participantes [18] 12
Figura 2 – Diagrama do olho [11] 15
Figura 3 – Densidade de distribuição de cones e bastonetes na superfície da retina [11] 16
Figura 4 - Desenho do olho com os eixos óptico e visual [11] 17
Figura 5 - Formação das 4 imagens de Purkinje [11] 18
Figura 6 - Distribuição aproximada da velocidade do olho em graus por segundo para cada tipo de movimento [11] 19
Figura 7 - Exemplo de lentes de contato utilizadas para rastreamento [8] 20
Figura 8 - Exemplo de uma medição de eletro-oculograma [16] 21
Figura 9 - Componentes do sistema de eye tracking: (1) bracelete com uma bolsa, (2) microprocessador, (3) óculos, (4) eletrodos, (h) e (v) eletrodos verticais e horizontais respectivamente, (l) sensor de luz e (a) acelerômetro. [3] 22
Figura 10 - Arquitetura do hardware (chamado de Pocket) [3] 22
Figura 11 - Movimentos elaborados para o jogo [3] 23
Figura 12 - Óculos com sensores e LEDs utilizado para rastreamento [20] 24
Figura 13 - Diagrama de blocos do sistema de rastreamento [20] 25
Figura 14 - Interface de aplicação par rastreamento [20] 25
Figura 15 - Montagem utilizada no método Dark pupil e imagem do olho captada pela câmera [1] 28
Figura 16 - Montagem utilizada no método Bright pupil e imagem do olho captada pela câmera [1] 28
Figura 17 - Modelo H6 Head Mounted Optics da empresa ASL [2] 29
Figura 18 - Modelo de eye tracker da empresa Tobii [19] 30
Figura 19 – Imagem de faces depois de realizado o rastreamento do olhar para crianças com TID e crianças com desenvolvimento normal [15] 34
Figura 20 – Câmera modelo Creative Webcam Vista [4] 38
Figura 21 – Webcam modelo Microsoft VX-1000 utilizada no projeto [10] 39
Figura 22 – Webcam já desmontada e desconectada [10] 40
Figura 23 - Detalhes da lente e dos filtros, o original e o novo [10] 40
Figura 24 – Pinos de alimentação correspondentes da conexão USB [10] 41

Figura 25 - Montagem concluída do óculos [10] 41
Figura 26 - (a) Imagem original. (b) Imagem com redução de ruído. (c) Imagem com a reflexão removida [6] 43
Figura 27 - (a) Candidato a centro da pupila em amarelo e realização da primeira etapa do algoritmo. (b-c) Realização da segunda etapa do algoritmo a partir de (a). (d) Resultado após uma iteração completa com o novo centro geométrico em vermelho. (e) Resultado após a segunda iteração. (f) Evolução do algoritmo através do centro geométrico. [6] 45
Figura 28 - (a) Pontos encontrados na etapa anterior. (b) Elipse ajustada com o método de mínimos quadrados. (c) Pontos em vermelho descartados pelo RANSAC. (d) Elipse gerada a partir de (c). [6] 46
Figura 29 – Óculos de proteção utilizado 52
Figura 30 - Espaguete Termo Retrátil 52
Figura 31 – Tela principal do Gaze Tracker [9] 53
Figura 32 – Exemplo de calibrações [9] 54
Figura 33 – Tela de calibração com 9 pontos fornecida pelo ITU Gaze Tracker 55
Figura 34 – Ponto de calibração com offset pequeno e desvio grande 56
Figura 35 - Ponto de calibração com offset médio e desvio pequeno 56
Figura 36 - Ponto de calibração com offset grande 56
Figura 37 – Tela do modo de configuração do eye tracking 59
Figura 38 - Tela do modo de calibração do eye tracking 59
Figura 39 - Tela do modo de teste do eye tracking 60
Figura 40 - Imagem da calibração obtida para os testes com o ITU Gaze Tracker 63
Figura 41 - Foto do protótipo em uso 64
Figura 42 - Região de interesse no Eye Writer 71
Figura 43 - Foto do filtro hot mirror adquirido 72

LISTA DE TABELAS

Tabela 1 - Resultados obtidos para o rastreamento com sensor [20] 26
Tabela 2 - Síntese das técnicas com os pontos favoráveis e desfavoráveis 31
Tabela 3 - Componentes do hardware openEyes [21] 37
Tabela 4 - Testes utilizando mouse convencional com blocos pequenos 65
Tabela 5 - Testes utilizando Touchpad de computadores portáteis com blocos pequenos 65
Tabela 6 - Testes utilizando ITU Gaze Tracker com blocos pequenos 66
Tabela 7 - Testes utilizando Eye Writer com blocos pequenos 66
Tabela 8- Testes utilizando mouse convencional com blocos grandes 67
Tabela 9 - Testes utilizando Touchpad de computadores portáteis com blocos grandes 68
Tabela 10 - Testes utilizando ITU Gaze Tracker com blocos grandes 68
Tabela 11 - Testes utilizando Eye Writer com blocos grandes 69
Tabela 12 – Preço dos componentes do sistema final projetado 73
Tabela 13 - Preços dos itens do hardware openEyes 74
Tabela 14 - Etapa 1: Determinação da tecnologia 77
Tabela 15 - Segunda Etapa: Estudo detalhado da tecnologia 77
Tabela 16 – Terceira Etapa: Definições sobre os softwares utilizados 78
Tabela 17 - Desenvolvimento do protótipo 79
Tabela 18 - Divisão de tarefas: Testes 79
Tabela 19 - Aplicações 80
Tabela 20 - Monografia Final 80

SUMÁRIO

1 INTRODUÇÃO **12**

1.1 Contextualização e Motivação *12*

1.2 Objetivos *13*

1.3 Organização do Trabalho *13*

2 REFERENCIAL TEÓRICO **15**

2.1 Considerações Iniciais *15*

2.2 Conceitos básicos *15*

2.2.1 Fisiologia do olho 15

2.2.2 Movimentos do olho 18

2.2.3 Eye Tracking x Gaze Tracking 19

2.3 Técnicas para rastreamento *19*

2.3.1 Rastreamento por lentes de contato 20

2.3.2 Rastreamento por eletro-oculografia (EOG) 20

2.3.3 Rastreamento por sensores 23

2.3.4 Rastreamento por vídeo 26

2.3.4.1 Rastreamento sem iluminação infravermelha 26

2.3.4.2 Rastreamento com iluminação infravermelha 27

2.4 Técnica escolhida *30*

2.5 Aplicações do rastreamento *32*

2.5.1 Usuabilidade e pesquisa de mercado 32

2.5.2 Interface homem-máquina 33

2.5.3 Neurociência 33

2.6 Considerações finais *34*

3 DESENVOLVIMENTO DO TRABALHO **35**

3.1 Considerações Iniciais *35*

3.2 Projeto openEyes *35*

3.2.1 *Hardware* original 36

3.2.2 Modificações no *hardware* original 38

3.2.3 *Software* original 42
3.2.3.1 Filtragem, detecção, localização e remoção da reflexão da córnea 42
3.2.3.2 Detecção dos candidatos a pontos característicos de maneira iterativa 43
3.2.3.3 Aplicação do RANSAC e determinação do melhor ajuste de elipsóide 45
3.2.3.4 Calibração e determinação do ponto do olhar 46
3.2.4 Modificações no *software* original 46
3.2.4.1 Software Starburst 47
3.2.4.2 Software cvEyeTracker 48
3.2.5 Desenvolvimento de *Softwares* auxiliares 49
3.2.6 Conclusões acerca do *openEyes* 50
3.3 Projeto ITU Gaze Tracker *51*
3.3.1 Hardware 51
3.3.2 Software 53
3.4 Projeto Eye Writer *57*
3.4.1 *Hardware* 57
3.4.2 Software 58
3.6 Considerações Finais *60*
4 RESULTADOS **62**
4.1 Considerações iniciais *62*
4.2 Metodologia de testes *62*
4.3 Resultados utilizando os blocos pequenos *64*
4.3 Resultados utilizando blocos grandes *67*
4.4 Análise dos resultados *69*
4.5 Dificuldades e Limitações *70*
4.6 Otimizações *72*
4.7 Orçamento *73*
4.8 Considerações finais *75*
5 GERENCIAMENTO DO PROJETO **76**

6 CONCLUSÃO **81**

6.1 Resultados e Contribuições/Inovações Esperadas *81*

6.2 Considerações sobre o Curso de Graduação *81*

REFERÊNCIAS **83**

APÊNDICE A – Carta de Gannt elaborada em Junho **86**

APÊNDICE B – Carta de Gannt elaborada em Setembro **87**

1 INTRODUÇÃO

1.1 Contextualização e Motivação

Desde o século 19 o movimento ocular é objeto de estudo de muitos pesquisadores. O que eles talvez não soubessem é como esses movimentos são de grande utilidade para as mais diversas aplicações. Atualmente, o rastreamento do olhar é utilizado, por exemplo, para pesquisas relacionadas à neurociência, pesquisas para linguagem cognitiva, usabilidade de *websites*, interação humano-computador, etc. Na figura 1 pode-se ver um exemplo de aplicação voltado para usuabilidade de *websites*, em que foi realizado um rastreamento da *homepage* de um *website*. Na figura as manchas vermelhas/amarelas/verdes mostram onde os usuários focaram sua atenção visual durante os 20 segundos iniciais de visualização da página.

Figura 1 – *Heat map* (mapa de calor) incluindo os primeiros 20 segundos de olhar de oito participantes [18]

Apesar da grande aplicabilidade, um fator que limita o uso do rastreamento do olhar é o alto preço de sistemas como esse. Por isso, para estimular o uso desses equipamentos, é interessante procurar soluções com menores custos sendo que para isso, é importante conhecer cada técnica de rastreamento e temas como processamento de sinal, já todas as técnicas existentes utilizam isto.

1.2 Objetivos

Este projeto tem como objetivo o desenvolvimento de um sistema para rastreamento do olhar composto de um óculos devidamente equipado e o *hardware* necessário. O equipamento do qual o óculos contará será composto de uma câmera do tipo *webcam* captando a imagem do olho do usuário. Esta imagem servirá para determinar a posição do olho, sendo que isto é feito através de um algoritmo próprio para isso. A câmera estará conectada a um computador que terá a função de realizar todo o processamento necessário em tempo real.

Quanto à aplicação do rastreamento do olhar, o objetivo é criar uma interface em que o usuário, através do movimento ocular, seja capaz de controlar o cursor de um mouse. Esta aplicação exige que o rastreamento do olhar seja feito em tempo real. A motivação para a escolha desta aplicação é a possibilidade de criar uma nova ferramenta de interação entre homem e máquina, diferente das usuais, como *mouse*, teclado, interações através do toque (*touchscreen*) e outras. Além disso, controlar o mouse através do movimento ocular pode ser uma ótima ferramenta de acessibilidade para portadores de esclerose lateral amiotrófica, por exemplo, que, como conseqüência da doença perdem a capacidade de se movimentar, restando muitas vezes apenas o movimento ocular.

1.3 Organização do Trabalho

Este trabalho está estruturado em oito capítulos. O segundo capítulo fornece um referencial teórico, explicando as técnicas existentes para o rastreamento e o detalhamento da escolha da técnica a ser utilizada.

O terceiro capítulo fornece todos os detalhes existentes no projeto. Para facilitar o entendimento, a descrição do funcionamento do equipamento está dividida em duas principais seções, uma contendo detalhes do *hardware* outra contendo detalhes do *software*.

No quarto capítulo é apresentado um cronograma com todas as atividades feitas para o atendimento dos objetivos propostos.

O quinto capítulo apresenta o orçamento do projeto, levando em conta o projeto original e o efetivamente construído.

No sexto capítulo é realizada uma análise de risco a respeito da execução do projeto, apontando os principais fatores que poderiam prejudicar o andamento do trabalho.

O sétimo capítulo trata do gerenciamento do projeto e descreve as atividades de cada integrante do grupo de acordo com os objetivos estabelecidos.

O último capítulo deste trabalho refere-se às conclusões e algumas considerações sobre o curso de graduação.

No apêndice A é possível ver a carta de Gantt feita no mês de junho de acordo com as atividades na época estabelecida.

No apêndice B é possível ver a carta de Gantt feita no mês de setembro de acordo com as atividades na época estabelecida.

2 REFERENCIAL TEÓRICO

2.1 Considerações Iniciais

O rastreamento do olhar pode ser feito de diversas maneiras e para as mais variadas aplicações. Por isso, neste capítulo será descrito em mais detalhes como cada técnica funciona e as respectivas pesquisas feitas para cada uma. Ao final será justificada a escolha da técnica para este projeto.

Além disso, neste capítulo será feita também uma descrição de três possíveis aplicações para o projeto, sendo que estas foram escolhidas pelo grupo como as mais importantes a serem pesquisadas.

2.2 Conceitos básicos

Antes de caracterizar cada uma das técnicas é importante detalhar alguns conceitos básicos sobre o olho humano, que serão importantes mais adiante e sobre diferenças entre nomenclaturas no rastreamento.

2.2.1 Fisiologia do olho

Na figura 2 é possível ver um esquema do olho humano, com todas as partes constituintes mais importantes.

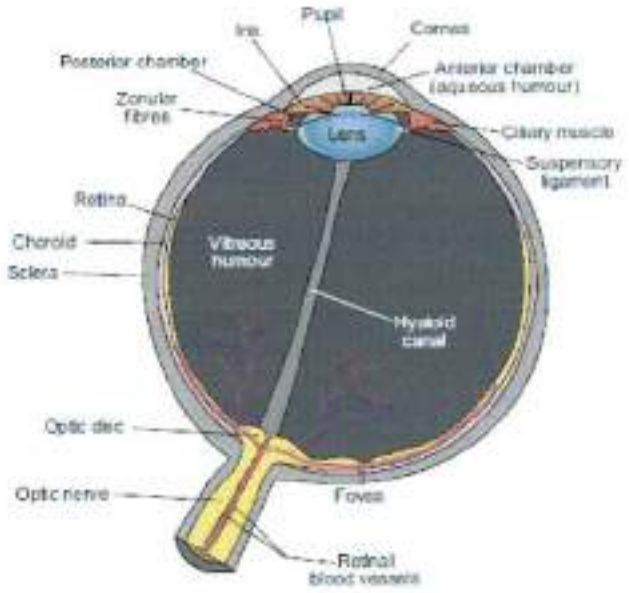


Figura 2 – Diagrama do olho [11]

O globo ocular é aproximadamente esférico e com um raio de aproximadamente 12mm. Há 3 estruturas que podem ser identificadas externamente: a íris (parte colorida do olho), a pupila (parte preta que fica no centro da íris) e a esclera (parte branca do olho). A pupila e a íris são cobertas pela córnea, que nada mais é do que uma camada transparente que reflete a luz antes que ela entre no olho. A fronteira entre a córnea e a esclera é conhecida como limbo. A pupila é a abertura por qual a luz entra no olho, sendo que a quantidade de luz pode ser regulada pela contração e expansão da mesma. Quando a luz entra pelo olho ela é refratada pelo cristalino, uma estrutura transparente localizada atrás da pupila. O formato do cristalino se altera de modo a focar os raios de luz para a retina, uma superfície localizada na parte de trás do olho e que contém fotorreceptores. Os fotorreceptores são um tipo de neurônio sensível a luz que convertem a luz em impulsos elétricos, sendo que esses impulsos são enviados ao cérebro através do nervo óptico. Pode-se dividir os fotorreceptores em dois principais grupos: cones e bastonetes. Os bastonetes são responsáveis pela visão monocromática e são muitos sensíveis a luz. Também são responsáveis pela visão noturna. Já os cones são menos sensíveis a luz e reconhecem as cores, além de serem mais sensíveis às mudanças rápidas de cena. Na figura 3 é possível ver a distribuição de cones e bastonetes (*rods*) na superfície da retina. [11]

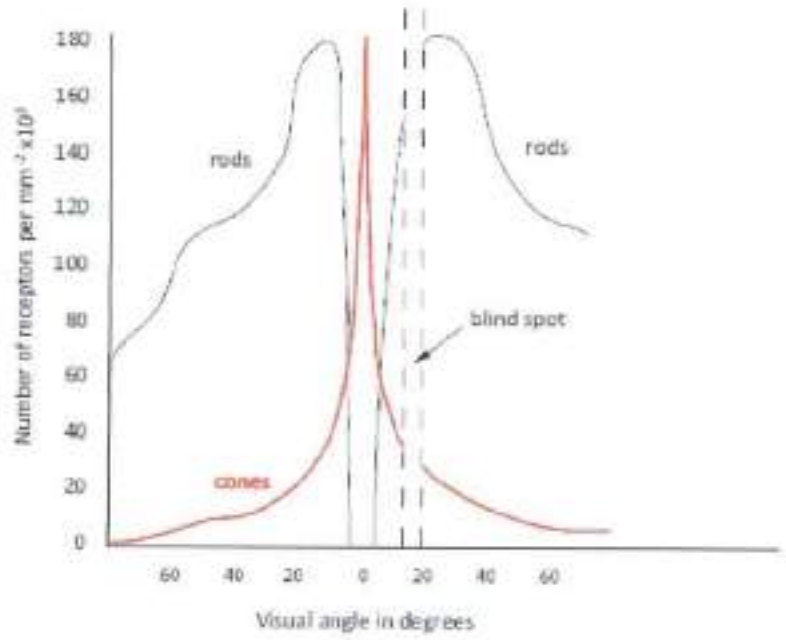


Figura 3 – Densidade de distribuição de cones e bastonetes na superfície da retina [11]

A maior concentração de cones ocorre na fóvea que apresenta um ângulo visual de aproximadamente 2°, condizente com a distribuição apresentada na figura 3.

O olho não é completamente simétrico. Entretanto, um eixo aproximadamente simétrico pode ser considerado como a linha que une as diferentes lentes do olho. Esta linha é chamada de eixo óptico, sendo que este não coincide com o eixo visual. Este último é formado por uma linha conectando a fóvea e o objeto para qual a pessoa está olhando. Na figura 4 é possível entender melhor a diferença entre esses eixos.

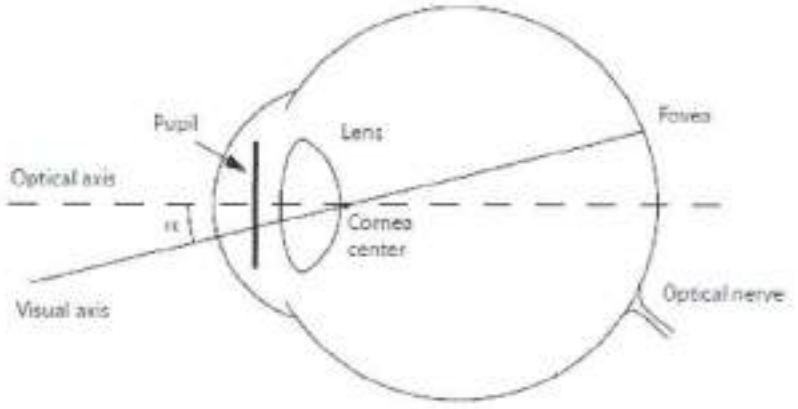


Figura 4 - Desenho do olho com os eixos óptico e visual [11]

Há duas estruturas que refletem a luz no olho: o cristalino e a córnea. Além da refração, essas duas superfícies refletem a luz de volta, formando as chamadas imagens de Purkinje. Na figura 5 é possível ver 4 dessas imagens. A primeira imagem de Purkinje é criada na superfície anterior da córnea e é a mais visível delas.

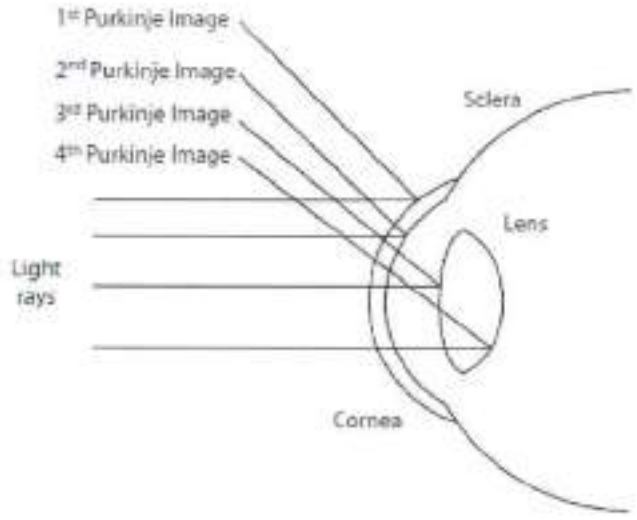


Figura 5 - Formação das 4 imagens de Purkinje [11]

2.2.2 Movimentos do olho

O sistema visual utiliza alguns movimentos para colocar objetos relevantes no eixo visual: movimentos sacádicos e de rastreio lento. Quando o olho está estável ocorre a fixação.

Fixação

A fixação ocorre quando olha-se para um objeto de interesse com uma duração de pelo menos 100ms a 150ms. Embora pareça que a imagem está completamente parada há 3 diferentes micromovimentos que ocorrem durante a fixação: microssacádicos, drifts sinuosos e tremores. O objetivo desses movimentos é manter excitados os fotorreceptores na retina e trazer o objeto alvo a ser visto para o centro da fóvea.

Movimentos sacádicos

Os movimentos sacádicos são movimentos rápidos que reposicionam o olho para que a cena visual seja projetada na fóvea. Estes movimentos ocorrem geralmente entre duas fixações.

Durante o movimentos sacádicos, o olho se movimenta a alta velocidade, chegando a picos de 700°/s. A duração do movimento não é constante mas aumenta conforme a amplitude do movimento aumenta.

Movimento de rastreio lento

O movimento de rastreio lento ocorre quando os olhos acompanham um objeto em movimento. É formado por duas diferentes componentes: uma de movimento e outra sacádica. O primeiro é utilizado para manter a fóvea estabilizada no objeto em movimento; a segunda é utilizada para menores correções e para reposicionar a fóvea no alvo visual.

Na figura 6 é possível ver a distribuição das velocidades do olho para cada movimento descrito.

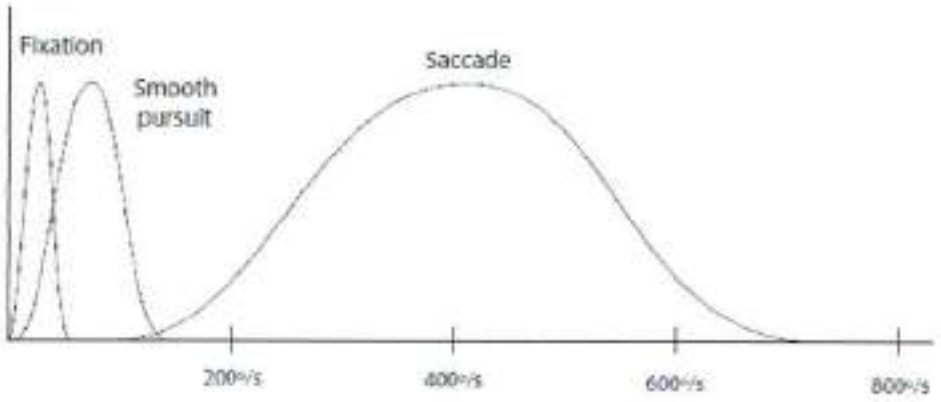


Figura 6 - Distribuição aproximada da velocidade do olho em graus por segundo para cada tipo de movimento [11]

2.2.3 Eye Tracking x Gaze Tracking

Existem dois possíveis rastreamentos, conhecidos como *eye tracking* e *gaze tracking*. O primeiro se refere aos movimentos do olho, sem qualquer relação com a posição da cabeça. Já o segundo, também mede o movimento do olho, porém com o objetivo de saber para onde a pessoa está olhando, o que depende da posição da cabeça [3]. Essa diferença entre eles será importante também para diferenciar cada técnica.

2.3 Técnicas para rastreamento

A seguir serão descritas as técnicas existentes para o rastreamento ocular.

2.3.1 Rastreamento por lentes de contato

Este método de rastreamento dos olhos é feito através do uso de uma lente de contato. A lente, nesse caso, possui duas bobinas de fio ortogonais que perturbam, de acordo com o movimento do globo ocular, um campo magnético de alta frequência que é aplicado ao redor da cabeça do usuário [13]. Na figura 7 é possível ver um exemplo de uma lente.

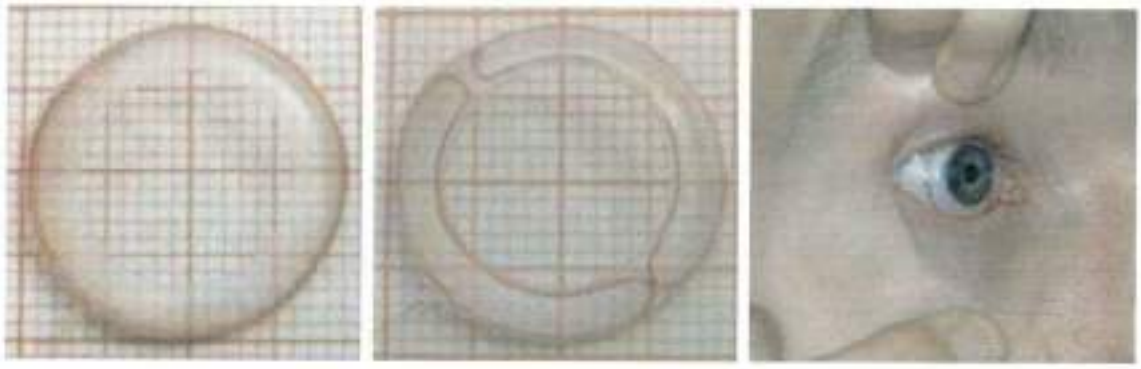


Figura 7 - Exemplo de lentes de contato utilizadas para rastreamento [8]

Apesar de possuir bastante precisão, este método é pouco utilizado na prática por ter alguns pontos negativos que são inerentes à sua concepção. Destes pontos podemos citar como principal o fato da técnica ser invasiva, já que a lente de contato é considerada um corpo estranho dentro do olho, podendo causar algum tipo de problema de saúde para o usuário, além de um eventual desconforto. Outro problema encontrado na técnica é a questão de não estarem resolvidas todas as dúvidas sobre as ameaças à saúde da pessoa que fica exposta ao campo magnético intenso.

Por ser uma técnica invasiva e pouco utilizada, este tipo de rastreamento foi eliminado inicialmente pelo grupo.

2.3.2 Rastreamento por eletro-oculografia (EOG)

Este método é resultado da medição do potencial de repouso da retina. Esta medição é feita através de eletrodos posicionados ao redor do olho, sendo que o seu princípio de funcionamento pode ser resumidamente explicado da seguinte forma: se

o olho é deslocado da sua posição central em direção a um eletrodo, este eletrodo "vê" o lado positivo da retina e o eletrodo oposto "vê" o lado negativo da retina [17]. Conseqüentemente, ocorre uma diferença de potencial entre os eletrodos. Supondo que o potencial de repouso é constante, o potencial registrado é uma medida para a posição dos olhos. A figura 8 demonstra um exemplo de uma eletro-oculografia [7].

Figura 8 - Exemplo de uma medição de eletro-oculograma [16]

Este método realiza apenas o *eye tracking* já que, independente do movimento da cabeça, a medição do potencial será a mesma. Portanto, para saber para a onde a pessoas está olhando realmente ou é necessário impedir o movimento da cabeça ou medir de alguma forma o seu movimento.

Como resultado da pesquisa sobre esta técnica foram encontrados estudos realizados por pesquisadores de Zurich, Suiça do laboratório ETH. Com em base em um dos artigos [3] pode-se fazer um resumo do sistema de *eye tracking* utilizando o sinal de EOG.

O sistema para este tipo de rastreamento consiste basicamente em óculos com eletrodos, amplificador para os sinais de EOG (que estão na faixa de 0,4mV – 1,0mV) e um microprocessador DSP (*Digital Signal Processing*). Na figura 9 é possível ver o equipamento elaborado e os componentes do sistema:

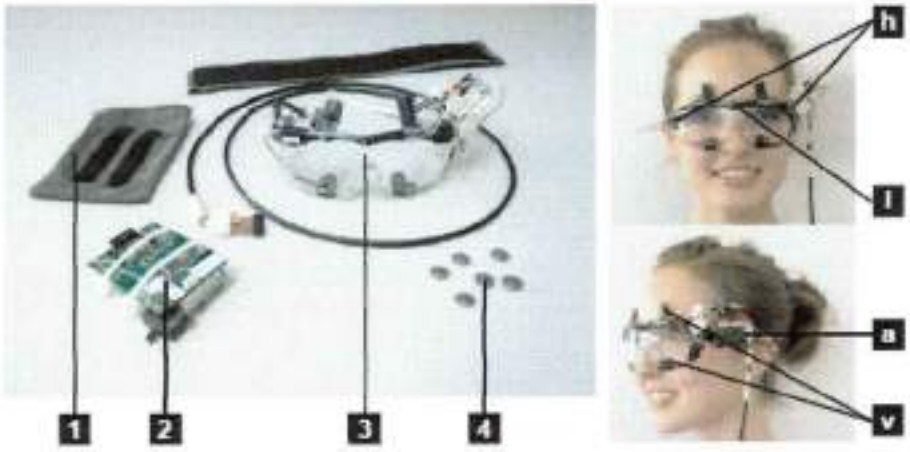


Figura 9 - Componentes do sistema de eye tracking: (1) bracelete com uma bolsa, (2) microprocessador, (3) óculos, (4) eletrodos, (h) e (v) eletrodos verticais e horizontais respectivamente, (l) sensor de luz e (a) acelerômetro. [3]

O *hardware* principal do sistema consiste no microprocessador (dsPIC), dois conversores A-D de 24 bits, Bluetooth, módulo MMC (*Multi Media Card*) e uma EEPROM. Na figura 10 é possível ver a respectiva arquitetura:

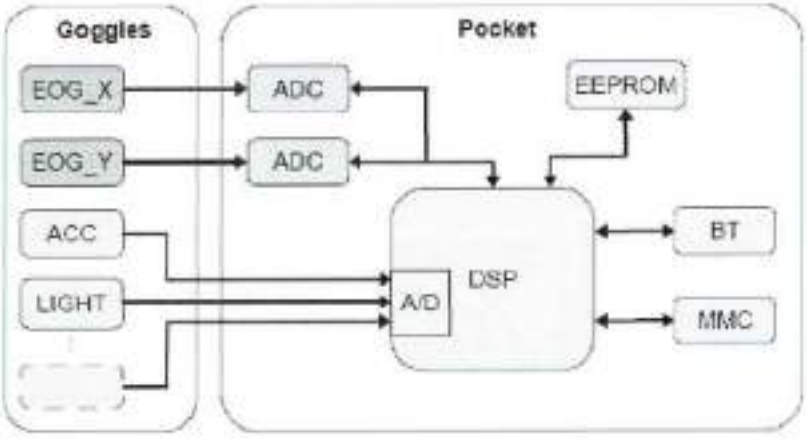


Figura 10 - Arquitetura do hardware (chamado de Pocket) [3]

Já em relação ao *software*, no dsPIC utiliza-se o freeRTOS, um sistema de operação em tempo real *open source*. Para o processamento de sinal foram elaborados alguns algoritmos, como: algoritmo para detecção de piscar, para detecção do movimento do olho e para remoção de piscar. Além disso, utilizou-se

também um filtro adaptativo para remoção de ruídos quando utilizava o sistema em movimento.

Com relação à aplicação, foi elaborado um jogo que consistia na repetição de movimentos mostrados na tela do computador. A figura 11 mostra os movimentos correspondentes. Antes do início do jogo, é feita uma calibração do sistema através de pontos de referência na tela colocados nos cantos e na borda da tela.

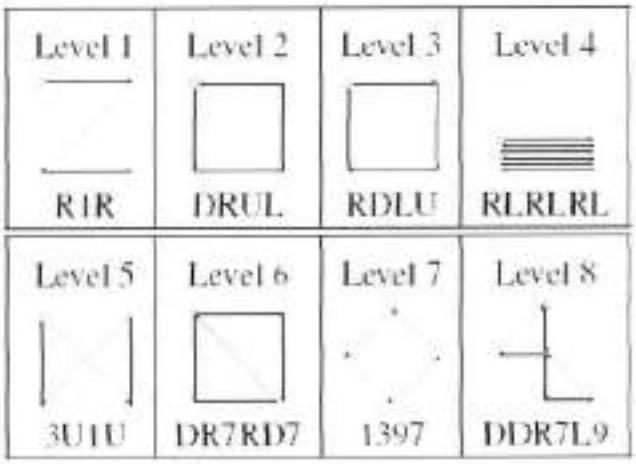


Figura 11 - Movimentos elaborados para o jogo [3]

Os pesquisadores chegaram às seguintes conclusões: o sistema construído apresenta menor computação, se comparado a outras técnicas como a de rastreamento por vídeo, permitindo o uso de um equipamento de baixo consumo energético. Os resultados para o jogo elaborado foram considerados bons (a razão mínima de movimentos que resultaram em gestos corretos pelo total de movimentos foi de 83%) e chegou-se a conclusão de que o sistema é mais robusto para variações de distância entre o usuário e a tela. Em compensação o método é menos confortável, devido a utilização dos eletrodos, e não é bom para o *gaze tracking*, ou seja, não consegue determinar com precisão para onde a pessoa está olhando.

2.3.3 Rastreamento por sensores

Pelo próprio nome, percebe-se que esta técnica é similar a anterior, sendo que agora não se utiliza mais a câmera como sensor e sim um sensor óptico

sensível a infravermelho. A posição do olho é medida através das reflexões do infravermelho na superfície do olho. Como cada região apresenta uma coloração diferente, haverá um maior ou menor número de reflexões captadas pelos sensores, permitindo a localização do olho.

A pesquisa por esta técnica se baseou principalmente na leitura do artigo intitulado "A Wearable Head-Mounted Sensor-Based Apparatus for Eye Tracking Applications" [20]. Neste artigo foi proposto o sistema de *eye tracking* através de sensores ópticos e LEDs de baixo custo. Na figura 12 é possível ver o protótipo construído com esta técnica.

Figura 12 - Óculos com sensores e LEDs utilizado para rastreamento [20]

O sistema conta com, além dos óculos, com um equipamento de aquisição de dados (DAQ), um computador e uma fonte de tensão externa. Na figura 13 é possível ver o diagrama de blocos do sistema.

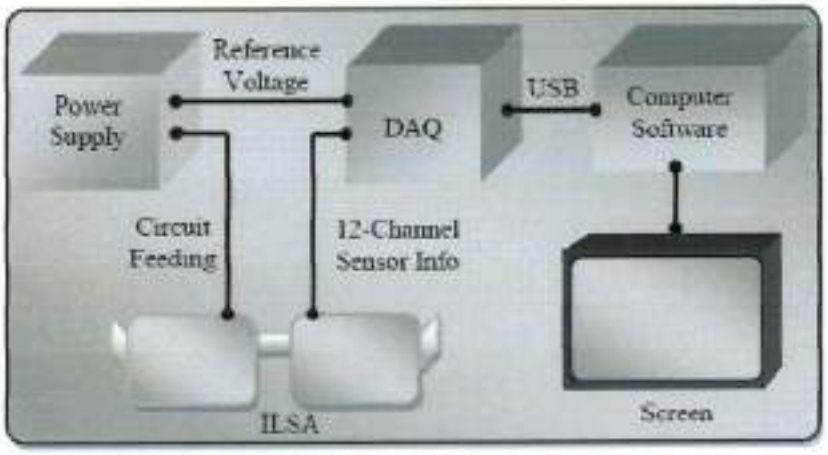


Figura 13 - Diagrama de blocos do sistema de rastreamento [20]

De acordo com os autores, futuramente, o equipamento de aquisição será substituído por um microcontrolador para permitir maior mobilidade ao sistema.

Com relação ao software projetado, ele apresenta uma interface que consiste em doze quadrados, cada um correspondente a uma região vista pelo usuário. Esta interface é apresentada na figura 14:

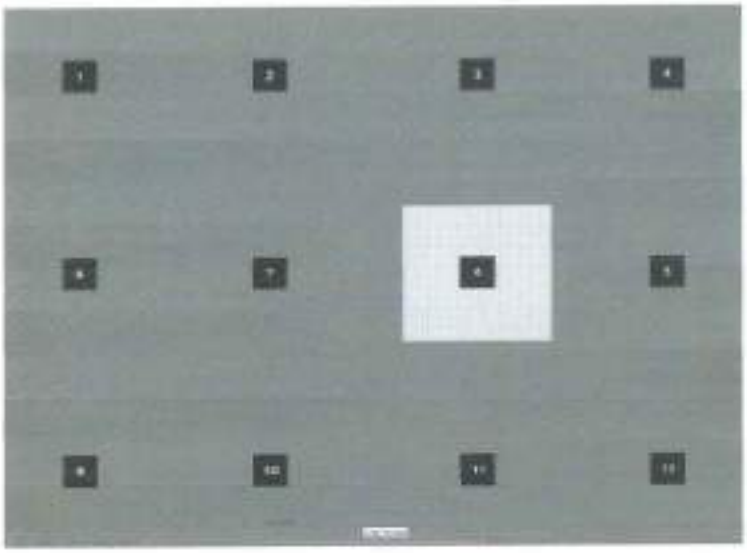


Figura 14 - Interface de aplicação par rastreamento [20]

Além disso, o *software* realiza duas funções: treinamento e rastreamento. A função de treinamento se refere a calibrar o sistema, sendo que esta calibração é feita com o usuário olhando para regiões específicas da tela. Enquanto olha, o

software coleta as informações necessárias para adaptar o sistema às propriedades físicas do olho do usuário e prepara os parâmetros do algoritmo para rastreamento. Já para a função de rastreamento, o algoritmo utiliza os dados fornecidos pelo DAQ para destacar a respectiva região da interface apresentada na figura 14. Também, o algoritmo pode utilizar os dados fornecidos para mover um cursor de *mouse*, sendo que para que o usuário selecione algum ícone na tela do computador é necessário fechar o olho por um tempo de aproximadamente 500ms.

Os resultados encontrados estão apresentados na tabela 1 abaixo. Note que a performance obtida para as regiões utilizadas na interface de aplicação podem ser consideradas boas, sendo que o menor valor apresentado foi de 89.28%.

Tabela 1 - Resultados obtidos para o rastreamento com sensor [20]

Region	Performance (%)	Region	Performance (%)
1	100	7	92.85
2	89.28	8	92.85
3	100	9	96.42
4	100	10	100
5	96.42	11	100
6	96.42	12	100

Este sistema de rastreamento é menos preciso que sistemas baseados nas técnicas de rastreamento por vídeo. A vantagem é que não é necessário utilizar uma câmera e o processamento envolvido é bem menor.

2.3.4 Rastreamento por vídeo

2.3.4.1 Rastreamento sem iluminação infravermelha

Para este tipo de rastreamento utiliza-se uma câmera para captação da imagem do olho. Obtida esta imagem, é feito um processamento de imagem sobre ela para a detecção da posição do olho. Este processamento pode tanto detectar o limbo do olho, através do contraste entre a esclera (o branco do olho) e o escuro da íris como também pode detectar a fronteira entre a íris e a pupila. Utilizando a pupila, existem algumas vantagens, já que ela é menos coberta pelas pálpebras,

permitindo um melhor monitoramento vertical, além de sua fronteira ser muitas vezes mais nítida do que a do limbo, gerando uma maior resolução [14].

A pesquisa revelou que, apesar de exigir um equipamento mais simples (apenas uma câmera focada no olho) e poder a princípio parecer mais fácil, a técnica que não utiliza infravermelho não é ideal para o desenvolvimento do projeto por questões práticas e construtivas. Este método exige um grande processamento de imagens, fato que acarreta problemas como a duração que este processamento pode levar, excluindo a possibilidade de uma aplicação em tempo real, por exemplo. Outro fator que dificulta sua implementação dentro da proposta apresentada é a sua utilização quase sempre exigir que a câmera foque os olhos de forma frontal, inviabilizando sua implantação em um óculos, como foi idealizado no projeto. Esta técnica pareceu ser viável para aplicações onde o usuário vai permanecer parado e de frente para o dispositivo utilizado, como no caso de uma pessoa trabalhando em um microcomputador, onde de fato ela é realmente mais utilizada.

2.3.4.2 Rastreamento com iluminação infravermelha

Além disso, existe outro método para detecção da pupila. Neste ilumina-se a pupila, utilizando infravermelho, sendo que existem duas maneiras para isso: *Dark pupil* e *Bright pupil*.

Dark pupil: com a iluminação a pupila aparece como um círculo negro. Desta maneira, realiza-se um processamento de imagem para localizar a pupila, utilizando o contraste entre a pupila e a íris. Este método pode ser problemático para olhos castanhos, já que o contraste entre a íris e a pupila é baixo [7]. Na figura 15 é possível ver a montagem necessária para o método e o resultado da imagem do olho.

Figura 15 - Montagem utilizada no método *Dark pupil* e imagem do olho captada pela câmera [1]

Bright pupil: este método utiliza luz infravermelha refletida da retina o que faz com que a pupila apareça com a cor branca na imagem da câmera. Para que isso aconteça, a iluminação com o infravermelho deve vir da mesma direção que a câmera. Na figura 16 é possível ver esta montagem e a imagem do olho capturada pela câmera [7]. Por fim, faz-se o processamento de imagem através do contraste entre a pupila (agora na cor branca) e a íris.

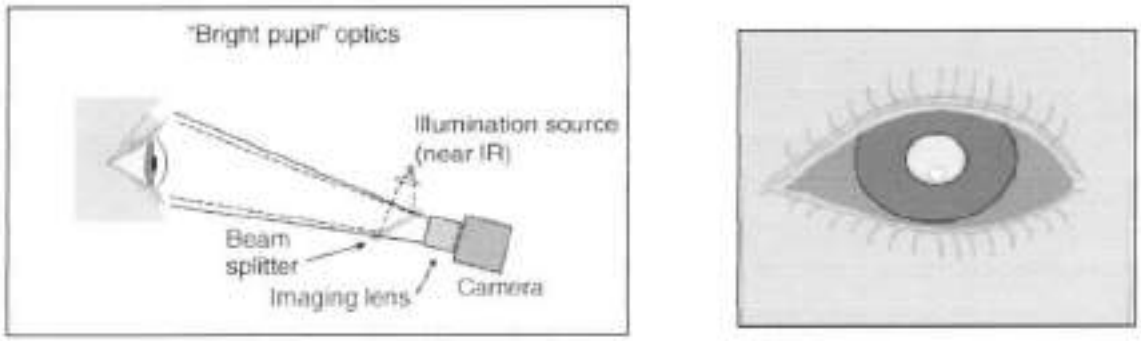


Figura 16 - Montagem utilizada no método *Bright pupil* e imagem do olho captada pela câmera [1]

A pesquisa sobre este tipo de rastreamento revelou que a técnica que utiliza a combinação de vídeo com a luz infravermelha apresenta algumas vantagens, e por essas razões se tornou a mais utilizada atualmente.

É possível afirmar que esse método é o mais utilizado atualmente não apenas pela grande quantidade de artigos científicos que foram encontrados através da pesquisa realizada, mas também por essa técnica ser implementada por algumas empresas que exploram a tecnologia de *eye track* de forma comercial. Existe não apenas uma grande quantidade de pesquisas que envolvem esse método, mas

também alguns procedimentos e *softwares* abertos (*open source*) que fazem o processamento necessário.

Para determinar para onde a pessoa está olhando com esse método, a imagem é processada seguindo alguns passos bem definidos. Primeiro ocorre uma filtragem (que pode ser feita através de uma janela gaussiana) para redução do ruído na imagem, e assim a detecção da reflexão na córnea poderá ser feita, seguida de sua localização precisa e de sua remoção [6].

O passo seguinte é a detecção do contorno da pupila na imagem, e o cálculo de um elipsóide que melhor se ajusta a ela. Uma vez que essas informações são obtidas, ocorre o confronto entre o ponto onde a pessoa está efetivamente olhando com a imagem gravada naquele momento e assim é determinado com precisão o ponto do olhar.

Durante este tempo, o grupo entrou em contato com uma empresa que desenvolve sistemas de *eye tracking* utilizando a técnica com iluminação com infravermelho, a ASL (Applied Science Laboratories). Através deste contato, a empresa forneceu um catálogo de como escolher a melhor tecnologia que eles dispõem. Na figura 17 é possível ver um dos modelos comercializados pela empresa. Nela é possível ver a presença lente, chamada "*hot mirror*", sendo que esta lente é capaz de refletir luz infravermelha, porém deixando passar a luz visível. A utilização desta lente auxilia na iluminação do olho com a luz infravermelha.

Figura 17 - Modelo H6 *Head Mounted Optics* da empresa ASL [2]

Outra empresa relevante no mercado é a Tobii. Esta empresa conta com diversos modelos, sendo que o modelo apresentado na figura 18 também apresenta uma lente do tipo "*hot mirror*". A câmera e a luz infravermelha ficam alocados no módulo à esquerda da figura.

Figura 18 - Modelo de eye tracker da empresa Tobii [19]

2.4 Técnica escolhida

Para a definição da técnica a ser utilizada foi montada uma tabela que sintetiza os pontos favoráveis e desfavoráveis de cada uma delas. A técnica utilizando lentes de contato não será adicionada, pois, como já dito, por ser uma técnica invasiva foi desconsiderada pelo grupo.

Tabela 2 - Síntese das técnicas com os pontos favoráveis e desfavoráveis

Técnica	Pontos favoráveis	Pontos desfavoráveis
Eletro-oculografia	• Menor computação para processamento de sinal, permitindo baixo consumo de energia • Robusto para variações de distância entre a tela e o usuário	• Utilização dos eletrodos pode ser desconfortável • Impreciso para *gaze tracking* • Sinais de baixa amplitude estão sujeitos a ruídos • Restrito a certas aplicações (quando envolvem *gaze tracking*)
Vídeo sem infravermelho	• Eficiente para aplicações sem óculos já que não necessita de iluminação infravermelha • Menos *hardware* (utiliza apenas a câmera)	• Sujeito a interferência da luz ambiente • Exige alto processamento de imagem
Vídeo com infravermelho	• Robustez quanto à interferência da luz ambiente • Técnica mais difundida e, portanto mais consolidada • Pode atuar em maior número de aplicações	• Exige alto processamento de imagem • Mais *hardware* (utilização da câmera com infravermelho)
Sensor óptico	• Baixo custo de implementação • Menor processamento de sinal	• Técnica menos precisa e por isso é restrita quanto a certas aplicações • Técnica menos difundida

Cada técnica apresenta vantagens e desvantagens quando comparadas entre si. Porém aquela que mais se destaca e será utilizada neste projeto é a de rastreamento por vídeo com infravermelho. Esta técnica é mais precisa e apresentar maior confiabilidade devido ao maior uso, ou seja, menor risco. Técnicas como a dos sensores ópticos e a da eletro-oculografia são menos utilizadas e menos confortáveis: a primeira apresenta um grande número de sensores que podem obstruir a visão e a iluminação infravermelha mais próxima e mais potente pode causar um desconforto na vista; a segunda exige o uso de eletrodos que devem ser bem fixados no rosto do usuário, principalmente quando se utiliza o sistema em movimento. Já a técnica de vídeo sem infravermelho é mais comumente utilizada em sistemas que não possuem óculos, como em aplicações com pouca movimentação do usuário, por exemplo, na frente de um monitor. Isto de certa forma limita o número de aplicações, diferentemente de um sistema portátil (possível com a iluminação infravermelha) que pode atender tanto aplicações com pouca movimentação ou até mesmo usuários se locomovendo.

Um ponto desfavorável da técnica é necessidade de maior processamento e, portanto mais *hadware*. Porém, projetos como o *openEyes* que será detalhado mais a frente apresenta uma configuração que exige menor custo, atenuando uma desvantagem desta técnica e confirmando-a como a melhor escolha.

2.5 Aplicações do rastreamento

2.5.1 Usuabilidade e pesquisa de mercado

Para publicidade a idéia seria trabalhar com propagandas em jornais e revistas, ou na tela de um computador. Neste caso, os usuários estariam lendo, olhando para imagens, sentados em frente ao objeto de pesquisa, parados e em ambiente fechado. A intenção é estudar o que chama a atenção da vista num texto ou imagem impressa.

O grupo entrou em contato com o responsável pela área de usabilidade na internet de uma renomada agência de publicidade (DM9DDB) para identificar eventuais problemas, ou futuros objetivos da área nesse campo de pesquisa.

2.5.2 Interface homem-máquina

Para interações de usuário com o computador, tem-se um sistema com o usuário parado em ambiente fechado também, olhando para a tela de seu monitor. O *eye track* é mais uma interface que pode ser implementada em microcomputadores, funcionando como um periférico adicional para controlar, por exemplo, o *mouse* sem o uso das mãos, ou um menu específico criado para esta interface, sendo ativado somente quando o usuário olhar para ele.

Além de auxiliar em ferramentas de uso geral o sistema de *eye track* pode ser usado de forma bastante específica por determinadas aplicações de um computador, como controlador de jogos. Para tal aplicação, é necessário identificar para onde o usuário olha em tempo real.

Além disso, temos o manuseamento de aplicativos em carro. Neste caso a estratégia deve ser parecida com a interação com o computador, com a óbvia exceção de que deve ser realizado um controle adequado para que o usuário não ative aplicações sem querer. Este é um sistema em que na maior parte do tempo, o motorista está mais preocupado com a via (pelo menos, espera-se que isso seja verdade) e então os aplicativos só devem ser acionados em momentos oportunos, e com a comprovada intenção do usuário.

Também se deve atentar ao fato de que o usuário está se movimentando neste caso, pois o corpo reage diferentemente a uma aceleração ou a uma freada brusca. Logo a preocupação com estabilidade é muito maior para esta aplicação.

Outra aplicação ligada à interface homem-máquina é a de assistência para deficientes ou portadores de doenças que prejudicam a locomoção, como portadores de esclerose múltipla.

2.5.3 Neurociência

Nesta área há inúmeros estudos relacionados à análise dos movimentos oculares. Por exemplo, há artigos que avaliam Transtornos Invasivos de Desenvolvimento (TID) (sendo o mais conhecido deles o autismo), mais especificamente comparando crianças com desenvolvimento normal com crianças com TID [15]. Neste caso, a análise foi feita utilizando imagens de faces e rastreando o movimento ocular das crianças. Os autores do artigo chegaram a conclusão que crianças com TID apresentam padrões de movimentos oculares

diferentes, sendo que essas crianças apresentam um padrão de menos fixações nas áreas do nariz e da boca, além de apresentar um maior número de fixações em pontos isolados distantes da face. Na figura 19 é possível ver o resultado do estudo.

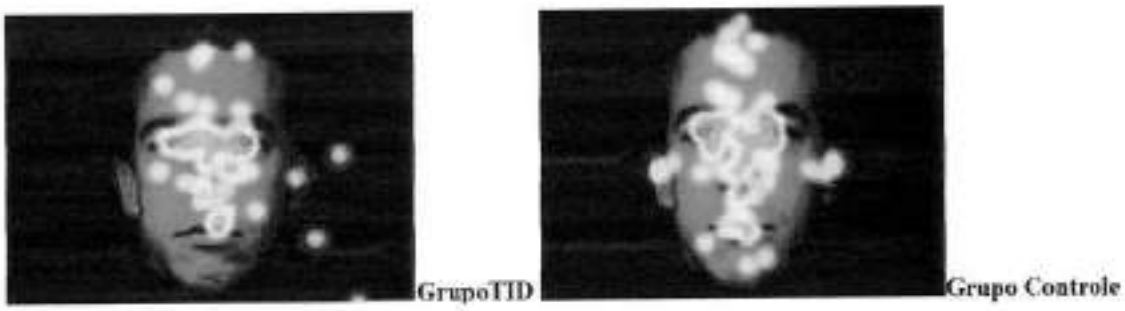


Figura 19 – Imagem de faces depois de realizado o rastreamento do olhar para crianças com TID e crianças com desenvolvimento normal [15]

Além disso, há estudos também relacionas ao processo cognitivo, como o estudo da alteração da exploração visual de imagens quando informações, como o título, são dadas antes da exibição da imagem [12].

2.6 Considerações finais

Neste capítulo foi possível conhecer as técnicas de rastreamento do olhar bem como alguns projetos já existentes. Ao final dessa caracterização foi feita uma síntese com os pontos mais importantes de cada técnica e também foi feita uma discussão acerca da escolha da técnica e as justificativas para isso. Por fim, foram caracterizadas também algumas aplicações do rastreamento do olhar.

No capítulo seguinte será detalhado o funcionamento do sistema de rastreamento do olhar a ser desenvolvido de acordo com a técnica escolhida anteriormente.

3 DESENVOLVIMENTO DO TRABALHO

3.1 Considerações Iniciais

Neste capítulo será descrito em detalhes o funcionamento do sistema que foi desenvolvido, tanto em software quanto em hardware. Como referências para o trabalho foram utilizados alguns projetos *open-source* que tratam da questão de rastreamento do olhar, entre os quais podemos destacar dois deles, o *openEyes* e o projeto desenvolvido pelo *Gaze Group* da Universidade de Copenhagen, o ITU *Gaze Tracker*.

O *openEyes* é um projeto que oferece uma solução completa, tanto para a questão do *hardware* quanto a do *software*, para o rastreamento por vídeo utilizando iluminação infravermelha. Por se tratar de um sistema robusto e eficiente, ele se tornou a primeira opção de escolha para se tornar base do trabalho, tendo seu algoritmo amplamente estudado.

Já o ITU *Gaze Tracker*, acabou se tornando a base para esse projeto por um tempo, conforme descrito no decorrer do capítulo. Fornece soluções apenas para o *software* de rastreamento, embora a estrutura de *hardware* necessária seja semelhante à do primeiro sistema.

Além disso, serão dados mais detalhes também do projeto *Eye Writer* que fornece tanto *software* quanto *hardware*. Este último é muito parecido com o *hardware* já feito para o projeto anterior, enquanto que o *software* apresenta algumas diferenças básicas.

Nas seções subseqüentes serão fornecidos detalhes dos três projetos citados, bem como algumas das alterações implementadas pelo grupo. Essas alterações são na verdade otimizações propostas que não fazem parte de nenhum dos projetos citados acima.

3.2 Projeto *openEyes*

Sabendo da importância discutida no item anterior da utilização de *softwares open source* para o projeto, foi encontrada uma solução que à primeira vista pareceu ser a mais adequada, o Projeto *OpenEyes*. Este projeto disponibiliza de forma livre um sistema completo para o rastreamento do olhar, sendo que para isso ele fornece

tanto o *hardware* necessário para tal aplicação, como o *software* com o algoritmo capaz de detectar o movimento através de imagens captadas por uma câmera.

Nos tópicos a seguir são apresentados em detalhes todas as principais partes do sistema de *software* e *hardware* originais desse projeto, que a princípio foi considerado como sendo a mais adequada base para a implementação das melhorias esperadas com esse trabalho.

3.2.1 *Hardware* original

O *hardware* proposto pela plataforma *openEyes* tem como princípio o baixo custo dos componentes, e consequentemente do sistema como um todo. A proposta para desenvolvê-lo é construir através dos itens listados na Tabela 3, um sistema que irá ser integrado com o *software* descrito na próxima seção, disponibilizando para ele as entradas necessárias para realizar o procedimento de *eye tracking*.

Tabela 3 - Componentes do *hardware openEyes* [21]

Aluminum Wire 14 gauge
Aluminum Wire 9 gauge
2 10' DB15 Male to Female cable
2 Aluminum Project Enclosures(13.3 x 7.6 x 5.4cm)
2 DB15 connectors
Fire-I Board Camera (black and white)
Fire-I Digital Camera (color)
5.7mm lens OR 12.0mm telephoto lens (eye lenses)
1.9mm wide angle lens OR 3.6mm medium angle lens (scene lenses)
17mm M12x0.5 Lens holder
13.5mm M12x0.5 Lens holder 13.5mm
940nm Infrared LED
4 14 pin dip sockets (do not use low profile sockets)
33 Ohm Resistor (0.75W or greater)
Safety Glasses
USB connector
Zip Ties
Standard Rosin-Core Solder
Electrical Tape
Plastic CD case
Nylon Spacers
8 Screws 2-56 x ½
Wratten IR Filter
Shrink wrap tubbing
2 DB9 connectors

As duas câmeras utilizadas, que são basicamente a parte principal do *hardware* são câmeras com interface FireWire já que como relatado em [21], houve uma dificuldade em encontrar câmeras baratas com interface USB compatíveis com a largura de banda do USB 2.0. O inconveniente desta câmera é o fato do sensor estar alocado diretamente na placa, exigindo que se retire para realizar a correta

montagem do sistema e o fato de elas apresentarem um alto custo (que será detalhado mais adiante na seção Orçamento).

3.2.2 Modificações no *hardware* original

O hardware original se baseia no uso de câmeras com interface FireWire já que, na época em que foi desenvolvido o projeto, poucas câmeras com interface USB do tipo 2.0 estavam disponíveis. Como este impedimento já não existe mais, foi considerada a utilização de câmeras do tipo *webcam* que, além de mais baratas, facilitam a montagem por permitir conexão direta com um computador.

Porém, para a utilização de *webcams*, é necessário realizar algumas modificações. A primeira delas é mais simples e consiste em apenas desmontá-la para ficar com um tamanho compatível para um sistema *head-mounted*. A segunda modificação é um pouco mais trabalhosa. Ela consiste em tornar a câmera em uma que capte imagens apenas no espectro do infravermelho. Para isso, é necessário remover o filtro IR, presente na maioria dos modelos de câmeras atuais, e substituí-lo por um filtro que permita a passagem apenas da luz no espectro infravermelho. Para este filtro podem ser utilizados filmes fotográficos negativos, sendo uma opção de baixo custo.

Inicialmente, utilizou-se a câmera modelo Creative WebCam Vista. Na figura 20 é possível ver o modelo.

Figura 20 – Câmera modelo Creative Webcam Vista [4]

Com este modelo, foi possível realizar apenas a segunda modificação mencionada anteriormente, de transformá-la em uma câmera infravermelho, já que não é possível desmontá-la adequadamente, já que a peça em que se encontra a lente não é acoplada ao circuito com sensor e sim à tampa frontal. Mesmo assim, com apenas uma das modificações possíveis já foi possível gravar vídeos que seriam posteriormente processados pelo *software* de rastreamento do projeto *openEyes*.

Em busca de outros modelos de câmeras, foi encontrado um artigo intitulado "How to build low cost eye tracking glasses for head mounted system" [10] que demonstra como construir um sistema *head mounted* utilizando *webcams*. A câmera especificada no artigo, modelo Microsoft VX-1000, permite as duas modificações necessárias, ou seja, é possível desmontá-la e deixá-la de um tamanho mais apropriado como também é possível transformá-la em uma câmera infravermelha. A seguir serão detalhadas as modificações feitas.

Inicialmente, desmonta-se a câmera. Na figura 21 é possível ver o modelo da câmera utilizada.

Figura 21 – *Webcam* modelo Microsoft VX-1000 utilizada no projeto [10]

Retirando o microfone, que não será necessário, e desconectando o cabo que se liga ao conector USB, tem-se o circuito da câmera com sua lente, como é possível ver na figura 22.

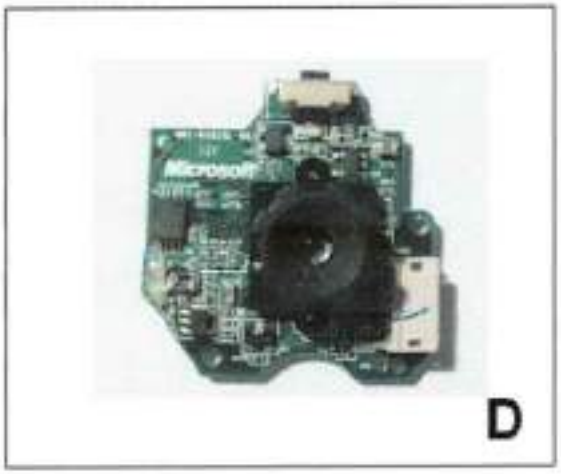


Figura 22 – *Webcam* já desmontada e desconectada [10]

Feito isto, retira-se a lente da câmera e substitui o filtro IR por um pedaço de filme fotográfico negativo de mesmo tamanho. Este procedimento, já feito na câmera anterior, pode ser visto na figura 23.

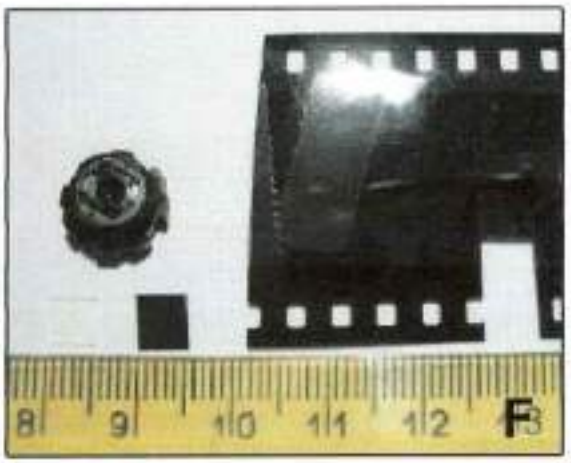


Figura 23 - Detalhes da lente e dos filtros, o original e o novo [10]

Com isto, finaliza-se a câmera para captação da imagem do olho. A câmera para captura da imagem referente a visão do usuário também é desmontada, porém não é necessário alterar seu filtro.

Além das modificações na câmera, também é necessário adicionar a iluminação infravermelha e montar a câmera no óculos. Para o primeiro, basta ligar o

LED infravermelho com um resistor nos pinos de alimentação correspondentes da conexão USB. Na figura 24 é possível ver detalhes destes pinos.

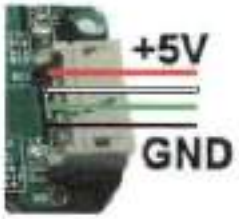


Figura 24 – Pinos de alimentação correspondentes da conexão USB [10]

Para finalizar, através de uma haste metálica monta-se a câmera junto ao óculos. Na figura 25 é possível ver o resultado final do óculos retirado do artigo. Note que neste modelo não está incluída a câmera para captura da imagem da visão do usuário, já que no artigo esta não é importante segundo suas necessidades.

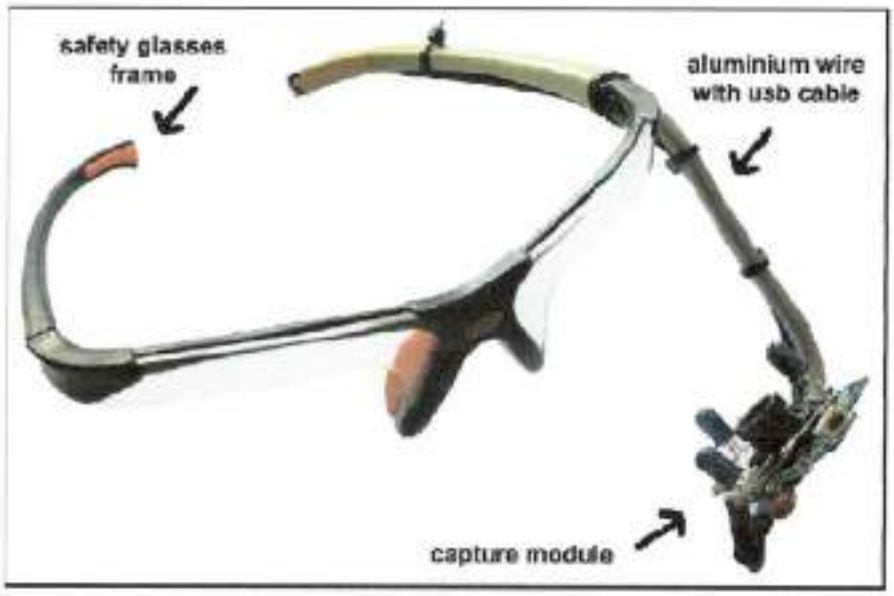


Figura 25 - Montagem concluída do óculos [10]

Com o óculos pronto, é possível utilizar os *softwares* disponíveis para o rastreamento do olhar, sendo eles de tempo real ou não.

3.2.3 *Software* original

Para realizar o rastreamento do olhar o sistema *openEyes* utiliza um algoritmo intitulado "*Starburst*", que integra as duas técnicas mais utilizadas para realizar esse tipo de detecção: ajustes baseados em pontos característicos que são detectados na imagem e aproximações baseadas em modelos. Por esta razão o *Starburst* é considerado um algoritmo híbrido e bastante eficiente [6].

Além da qualidade no resultado final este algoritmo realiza o processamento completo que é necessário, já que ele recebe como entrada a imagem do olho obtida da câmera de vídeo e retorna na saída o ponto para onde a pessoa está olhando. O algoritmo pode ser descrito de maneira simplificada em quatro itens:

- Filtragem, detecção, localização e remoção da reflexão da córnea.
- Detecção dos candidatos a pontos característicos de maneira iterativa.
- Aplicação do RANSAC e determinação do melhor ajuste de elipsóide.
- Calibração e determinação do ponto do olhar.

3.2.3.1 Filtragem, detecção, localização e remoção da reflexão da córnea

Assim como em qualquer sistema, a imagem que vem da câmera é colocada na entrada do algoritmo possui uma quantidade significativa de ruído, o que atrapalha consideravelmente o processamento. Para reduzir esta interferência e aumentar a eficiência, o primeiro passo do algoritmo é realizar uma filtragem nessa imagem, que no caso é feita através de um Filtro Gaussiano utilizando uma janela 5x5 e desvio padrão de dois pixels.

Depois de ter o ruído reduzido, como pode ser observado na Figura 26 (a) e (b), é iniciada a etapa de detecção da córnea. Como o *openEyes* utiliza a técnica de *dark-pupil*, a reflexão da córnea pode ser obtida através da detecção de limiar entre o único ponto brilhante (que é a reflexão em si) e o resto da imagem. Para agilizar a localização, o algoritmo realiza essa procura apenas na região onde é possível que haja esse reflexo, ganhando alguma economia de processamento.

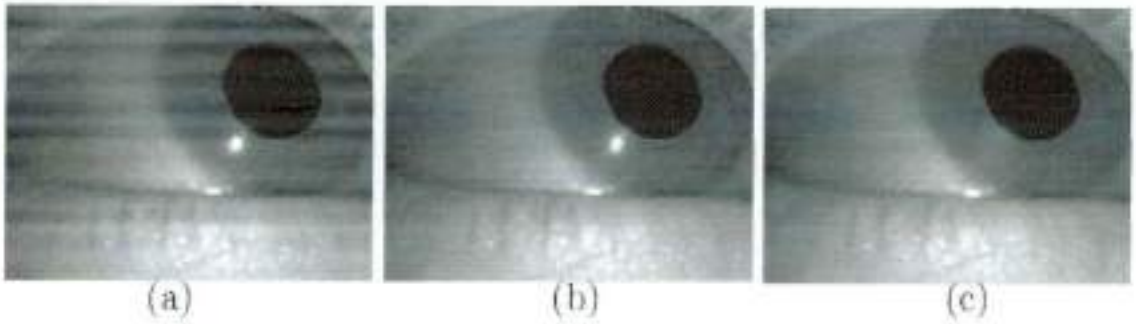


Figura 26 - (a) Imagem original. (b) Imagem com redução de ruído. (c) Imagem com a reflexão removida [6]

Por fim, para a remoção da reflexão da córnea é utilizada uma interpolação radial a partir do pixel central da reflexão. Para determinar o valor do pixel central na imagem corrigida, é calculada a média dos valores de contorno da região de reflexão e atribuído a ele. Depois é realizada uma interpolação linear simples entre este novo valor de centro e cada ponto de contorno, corrigindo a imagem, como pode ser observado na Figura 26 (b) e (c).

3.2.3.2 Detecção dos candidatos a pontos característicos de maneira iterativa

A técnica que utiliza a detecção de pontos característicos basicamente analisa se determinada propriedade está presente ou ausente na imagem. A característica que é analisada varia bastante entre os diversos algoritmos, mas na maioria das vezes são utilizados níveis de intensidade ou gradientes. Um exemplo disso é o caso de imagens infravermelhas onde através da variação de intensidade podem-se detectar os contornos presentes no olho, sendo possível separar elementos e assim determinar sua posição. No caso do *Starburst*, essa técnica é utilizada para detectar o contorno da pupila.

O algoritmo da função utilizada para realizar a detecção desses pontos recebe como entrada a imagem do olho já com o reflexo da córnea removido e o melhor candidato a ponto central da pupila, e retorna os pontos característicos. Esta função iterativa é dividida em duas partes.

A primeira etapa consiste em analisar raios que saem do ponto de partida e calcular a variação de intensidade entre cada ponto. Se esta variação for maior que

certo limiar, ele irá marcar aquele ponto e parar de analisar aquele raio, partindo para o próximo. A segunda etapa é idêntica à anterior, porém realizada para cada um dos pontos encontrados no estágio predecessor, e terminada determinando um novo ponto de partida como sendo o centro geométrico de todos os pontos marcados. Estas duas etapas são repetidas até que os pontos de partida convirjam. Na Figura 27 podemos observar melhor o funcionamento desta função. A primeira imagem, Figura 27 (a), mostra o ponto de partida fora da pupila, representado pelo circulo amarelo. Partindo dele temos os raios em vermelho terminados pelos candidatos a contorno da pupila, encontrados na primeira etapa do algoritmo. Já nas Figuras 27 (b) e (c), é realizada a segunda etapa, ou seja, a partir de um ponto encontrado anteriormente temos uma procedimento semelhante ao anterior. A Figura 27 (d) mostra o resultado final da realização da primeira e da segunda etapa com os candidatos a contorno em verde, o ponto de partida em amarelo e o novo ponto de partida para a próxima iteração em vermelho, obtido através do centro geométrico dos pontos. Uma segunda iteração do algoritmo é mostrada na Figura 27 (e), enquanto na Figura 27 (f) é possível observar a evolução na determinação dos candidatos a centro da pupila, cada vez se aproximando mais do centro real, e consequentemente do contorno mais adequado.

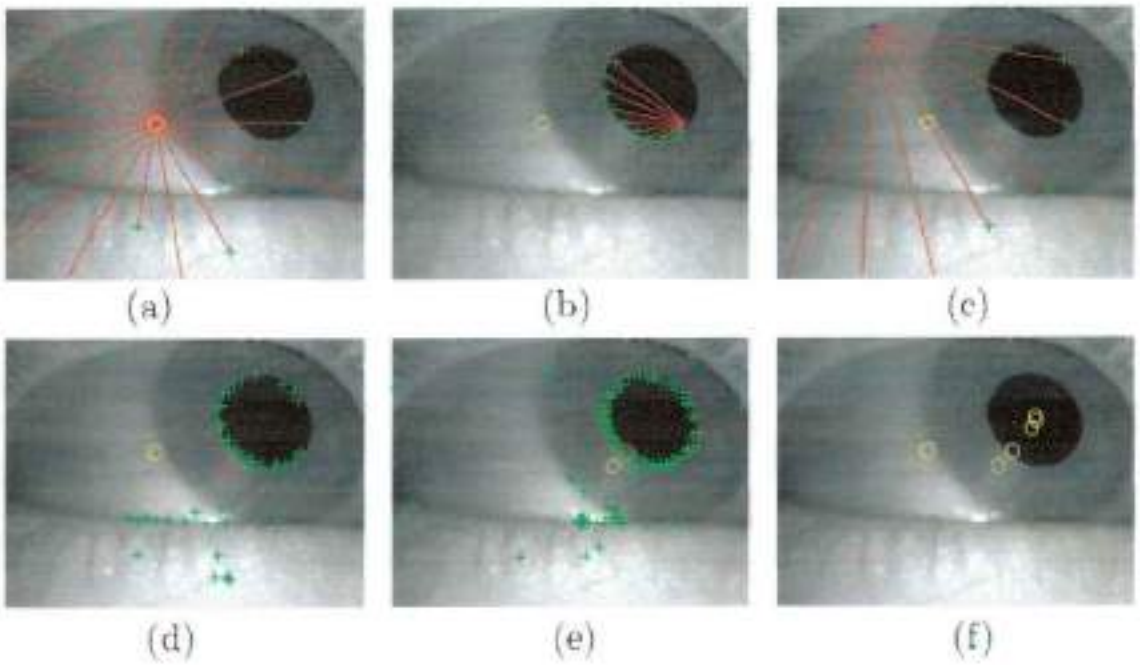


Figura 27 - (a) Candidato a centro da pupila em amarelo e realização da primeira etapa do algoritmo. (b-c) Realização da segunda etapa do algoritmo a partir de (a). (d) Resultado após uma iteração completa com o novo centro geométrico em vermelho. (e) Resultado após a segunda iteração. (f) Evolução do algoritmo através do centro geométrico. [6]

3.2.3.3 Aplicação do RANSAC e determinação do melhor ajuste de elipsóide

Para determinar o melhor elipsóide que irá se ajustar aos pontos de contorno da pupila é utilizado o método RANSAC. Essa técnica é utilizada nesse caso ao invés de um comum ajuste de mínimos quadrados porque ela é bastante adequada para o caso em que se sabe que entre os pontos que vão ser ajustados existem muitos que são inconsistentes, chamados de *outliers*.

Enquanto o ajuste de mínimos quadrados considera que todos os pontos fazem efetivamente parte do conjunto que deve ser ajustado, o RANSAC considera a existência de *outliers* e compensa isso. Para realizar essa tarefa, o método RANSAC consiste em um procedimento iterativo que seleciona subconjuntos muito pequenos de dados aleatórios, e usa cada subconjunto para ajustar um modelo, e encontrar um modelo que esteja mais de acordo com o conjunto de dados como um todo. Para o caso em questão, os *outliers* são os pontos marcados na etapa anterior que conhecidamente não fazem parte do contorno da pupila.

A Figura 28 (a) mostra com pontos em verde os candidatos a contorno da pupila obtidos anteriormente. Na imagem é possível observar que existem dois pontos *outliers*, que são responsáveis pelo ajuste ruim da elipse da Figura 28 (b),

obtido através do método de mínimos quadrados. Na Figura 28 (c) os dois pontos *outliers que* estão marcados em vermelho são descartados pelo RANSAC, possibilitando assim um ajuste mais adequado, mostrado na Figura 28 (d).

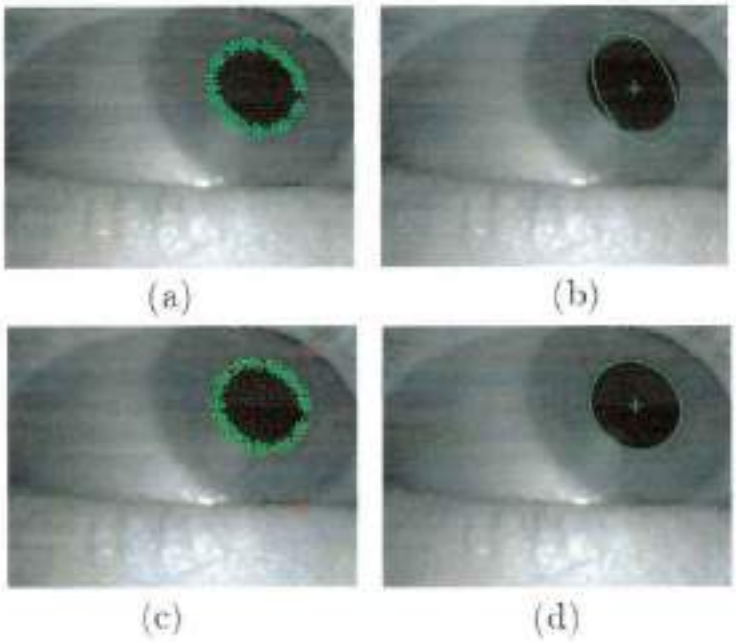


Figura 28 - (a) Pontos encontrados na etapa anterior. (b) Elipse ajustada com o método de mínimos quadrados. (c) Pontos em vermelho descartados pelo RANSAC. (d) Elipse gerada a partir de (c). [6]

3.2.3.4 Calibração e determinação do ponto do olhar

Para determinar o ponto onde a pessoa está olhando sobre a imagem do ambiente, é preciso realizar um mapeamento entre as coordenadas da posição do olho e da imagem que veio da câmera que está gravando o ambiente. Isso é obtido no caso deste algoritmo através do vetor diferença entre centro da pupila e o centro do reflexo da córnea, e também se realizando uma calibração onde o usuário olha para pontos conhecidos e pré-determinados (uma grade 3x3 neste caso), quando então a posição do olho é medida. Desta maneira o ponto onde o olho está é determinado, e o seu correspondente ponto de olhar na imagem ambiente.

3.2.4 Modificações no *software* original

Originalmente, o projeto *openEyes* apresenta dois softwares para o rastreamento: o primeiro, de nome *Starburst*, feito em MATLAB que realiza processamento com vídeos gravados anteriormente e o segundo, intitulado

cvEyeTracker, que foi programado em linguagem C e realiza rastreamento em tempo real. Para os dois softwares foram necessárias modificações para seu uso e elas serão explicadas nos itens a seguir.

3.2.4.1 Software Starburst

O software *Starburst* foi desenvolvido em MATLAB, porém feito na plataforma Linux. Seu funcionamento se baseia na execução de 4 *scripts*, sendo que ele devem ser executados em ordem. Os arquivos a serem executados estão escritos abaixo, já na ordem de execução, e uma breve explicação é dada de cada um:

- *extract_images_from_video* – cria uma série de imagens a partir do vídeo fornecido. Isto é feito amostrando o vídeo com uma taxa já pré-determinada.
- *extract_calibration_from_video* – função responsável pela calibração do rastreamento.
- *calculate_pupil_and_gaze* – com base na calibração, processa o vídeo com a imagem do olho, estimando para onde o usuário olha.
- *plot_pupil_and_gaze* – gera um novo vídeo com o resultado obtido da função anterior.

Como todo o trabalho estava sendo feito no sistema operacional *Windows*, foram necessárias algumas mudanças para que o *software* funcionasse adequadamente, mais especificamente em dois dos arquivos: "extract_image_from_video" e "extract_calibration_from_video".

Em "extract_image_from_video" as mudanças foram:

1) Mudança no comando de execução do ffmpeg – Na plataforma *Linux*, o programa ffmpeg já vem disponível, enquanto que para *Windows* foi necessário adicioná-lo na mesma pasta em que estava localizado o código.
2) Adição de outras extensões de vídeo – O *software* original aceitava apenas a extensão mp4, relativa ao arquivo de formato *MPEG-4 Part 14*. Foram adicionadas as seguintes extensões: mpeg, mpg, avi e wmv, possibilitando trabalhar com outros formatos de vídeos.

Em "extract_calibration_from_video" as mudanças foram:

1) Mudança na função "get_first_or_last_frame" devido à utilização do comando ls que possui diferentes resultados em *Linux* e *Windows*.
2) Geração de elipses em torno da pupila na imagem do olho do usuário – O programa original não desenhava as elipses em volta da pupila, o que impossibilitava que o usuário pudesse ver o resultado final da elipse estimada pelo algoritmo que melhor definia o contorno da pupila. A construção pode ser feita através das equações paramétricas da elipse (equações 1 e 2):

$$x(t) = x_c + a\cos(t)\cos(\varphi) - b\sin(t)\sin(\varphi) \quad \text{Eq. 1}$$

$$y(t) = y_c + a\cos(t)\sin(\varphi) + b\sin(t)\cos(\varphi) \quad \text{Eq. 2}$$

Em que:

x_c e y_c – centro da elipse

t varia de 0 a 2π

φ - ângulo entre o eixo principal da elipse e o eixo x

a – distância do semieixo maior

b – distância do semieixo menor

Basicamente estas foram as mudanças realizadas. Mesmo com todas elas, o *software* ainda apresentava problemas, mais especificamente no último dos arquivos, o "plot_pupil_and_gaze". Após diversos testes, não foi possível solucionar este problema, sendo que o grupo partiu para outra versão do *Starburst*, escrita por Joel Clemens e destinada à plataforma *Windows* [5]. Esta nova versão funcionou corretamente, porém, é destinada a vídeos gravados sem iluminação infravermelha, ou seja, não atende aos objetivos requeridos do projeto.

Com estes problemas e somando o fato de que o *software* é muito pouco usual, principalmente para a realização da calibração, partiu-se à procura de outros programas que atendessem os objetivos do projeto.

3.2.4.2 Software cvEyeTracker

O software *cvEyeTracker* também faz parte do projeto *openEyes*, mas, diferentemente do *Starburst*, faz o processamento em tempo real. Escrito na linguagem C, utiliza os mesmos algoritmos do *Starburst*, e, além disso, utiliza a

biblioteca OpenCV. Assim como o *software* anterior, também foi programado na plataforma Linux.

Para execução do código foi necessário utilizar o sistema operacional baseado em *Linux*, e no caso foi escolhido o Ubuntu 11.04. Mesmo assim alguns problemas foram encontrados. Um deles se relaciona com a biblioteca OpenCV pois, a última versão do código data do ano de 2006 e muitas das instruções da biblioteca estão ultrapassadas. Sendo assim, todas as respectivas instruções tiveram que ser modificadas. Além disso, outra biblioteca a "libraw1394-0.10.1" foi retirada do arquivo original já que ela é utilizada para conexões *FireWire* que não será utilizado neste projeto.

Após diversas mudanças, ainda não foi possível compilar o código. Alguns dos erros ainda podem estar relacionados com instruções que estão ultrapassadas, como ocorreu com as instruções da biblioteca OpenCV. Sendo assim, os esforços acabaram se voltando para outro *software*, o *Gaze Tracker*, que será detalhado mais adiante.

3.2.5 Desenvolvimento de *Softwares* auxiliares

Uma das adaptações que foi realizada com relação ao projeto original do *openEyes* foi a substituição das câmeras com conexão *FireWire* por câmeras de conexão USB comum. Esta mudança, apesar de trazer uma economia no projeto, introduziu um novo desafio, já que as câmeras originais possuíam a propriedade de se conectarem entre si e assim os dois sinais já chegavam para o processamento de maneira sincronizada, o que não acontecia no caso das câmeras USB.

Para contornar esse problema foi necessário desenvolver um software que realizasse essa função de fazer com que o sinal de ambos os vídeos chegassem juntos no computador. Foram estudas e levantadas três opções para desenvolver esse *software*, as linguagens C, C# e o ambiente computacional Matlab.

O desenvolvimento utilizando linguagem C foi cogitado devido ao fato de existir a biblioteca *open source* "OpenCV", para processamento utilizando visão computacional. Essa biblioteca também é utilizada pelo projeto *openEyes*, nesse caso para processamento em tempo real. Seu uso foi descartado devido ao fato de

que sua complexidade excedia as necessidades do projeto, principalmente quando comparada com as outras opções.

A segunda opção foi o uso do *software* Matlab, que, além de o grupo estar mais familiarizado, possui um conjunto de ferramentas específicas para a aquisição e o processamento de imagens, o que facilitaria bastante a integração com o resto do projeto, uma vez que ele também seria processado nesse mesmo ambiente. Foi escrito um algoritmo preliminar para aquisição de imagens nesse ambiente que era capaz captar os vídeos, no entanto ainda assim ele não foi tão eficaz quanto a outra alternativa, que era desenvolvida em paralelo.

O *software* final para sincronização das câmeras foi escrito utilizando a linguagem de programação C# que assim como as outras possui uma biblioteca própria para trabalhar com aquisição e processamento de imagens, a *DirectShowNet*. Através dele é possível escolher, por exemplo, a taxa de quadros por segundo que queremos captar, a resolução das imagens e também o formato de compressão do vídeo através de qual CODEC utilizar, o que trás ao projeto uma grande flexibilidade com relação ao tamanho dos arquivos de vídeo gravados.

3.2.6 Conclusões acerca do *openEyes*

O projeto *openEyes* apresenta mais de uma solução para o rastreamento do olhar: uma com processamento em tempo real e outra com processamento de vídeos gravados anteriormente. Infelizmente diversas dificuldades apareceram e, mais importante, foi possível tirar algumas conclusões que corroboraram para a desistência em utilizar os *softwares* disponíveis no projeto *openEyes*. Por exemplo, para o *Starburst*, a dificuldade na calibração foi um importante fator. Como esta calibração é feita manualmente, o usuário deve informar ao algoritmo para onde o usuário está olhando e após isso, identificar o centro da pupila do olho do usuário. Se não bastasse a imprecisão deste método, caso o resultado da elipse sobre o contorno da pupila for errado, o usuário é obrigado a repetir o processo. Para o *cvEyeTracker*, o fato de o código estar desatualizado e com erros que impedem sua compilação fazem com que ele não seja a melhor escolha para a realização do rastreamento.

Mesmo que o *openEyes* não seja mais utilizado neste projeto, todos os conceitos que foram aprendidos para o entendimento dos algoritmos servirão como uma ótima bagagem ao longo da finalização do projeto.

3.3 Projeto ITU Gaze Tracker

Após vários testes e modificações com o projeto *openEyes*, foi necessário pesquisar por outros *softwares* similares, de preferência *open source*, para substituir os *softwares* disponíveis no projeto *openEyes*.

Através do orientador do projeto, foi apresentado um aluno de graduação de que, coincidentemente, está trabalhando com rastreamento do olhar no ICB (Instituto de Ciências Biomédicas) da USP, no Laboratório de Fisiologia do Comportamento. Apesar de seu trabalho ter outra finalidade ele apresentou outro *software open source* muito interessante.

Este *software* é o *GazeTracker* da ITU-DK. Muito mais prático nas questões de calibração e interface, este código foi escolhido pelo grupo como candidato a substituto do projeto *openEyes*. Abaixo serão dados mais detalhes deste novo projeto.

3.3.1 Hardware

O *hardware* necessário para o projeto ITU *Gaze Tracker* é basicamente o mesmo do projeto *openEyes*. Porém, serão demonstrados aqui o óculos especifico que foi utilizado, e como conseguiu-se prendê-lo à câmera.

No começo do projeto, utilizava-se um óculos de proteção de plástico, que apresentou uma dificuldade muito difícil de corte da lente para o posicionamento da câmera. Além disso, a lente ficava apoiada diretamente no nariz. Para garantir maior estabilidade na cabeça, utilizou-se um óculos de proteção especifico, com lentes que são facilmente removidas e um apoio mais robusto para o nariz. Na figura 29, há uma foto deste material utilizado.

Figura 29 – Óculos de proteção utilizado

Para prender a câmera ao óculos, depois de desmontá-la, utilizou-se uma haste de alumínio acoplada ao chamado espaguete termo retrátil, objeto que se fixa a outro quando submetido à alta temperatura (o material pode ser visualizado na figura 30.). Além disso, foram utilizados braceletes para fixar a haste no óculos, dando maior rigidez ao sistema. Foram utilizados também pequenos parafusos para prender a haste à câmera.

Figura 30 - Espaguete Termo Retrátil

A haste de alumínio junto com a câmera foi fixada na armação do óculos, ficando a câmera a uma distância conveniente.

3.3.2 Software

Este *software* é compatível com qualquer plataforma com *Microsoft .Net Framework 3.5* com pacote de serviços SP1 do *Windows XP*. A seguir, demonstra-se o funcionamento do *software*.

A tela principal possui três opções, como apresentado na figura 31: instalação, calibração e início. Quando o programa é carregado, a única opção disponível é a de instalação, em que o usuário configurará as opções de calibração e as configurações da câmera. Logo após, a calibração é feita com as configurações selecionadas anteriormente, e assim que finalizada, o botão de início é habilitado.

Figura 31 – Tela principal do Gaze Tracker [9]

Abaixo, maiores detalhes sobre essas as duas partes iniciais, necessárias para a execução do programa.

3.3.2.1 Configurações

Captura da pupila

Primeiramente, deve-se escolher o centro da pupila, com o auxílio do *mouse*. Logo após é necessário a indicação do tamanho mínimo e máximo para a pupila. Esta etapa é importante para que o rastreador classifique objetos apenas dentro deste tamanho da pupila. Na figura 32, um exemplo de certo e errado para esta calibração.

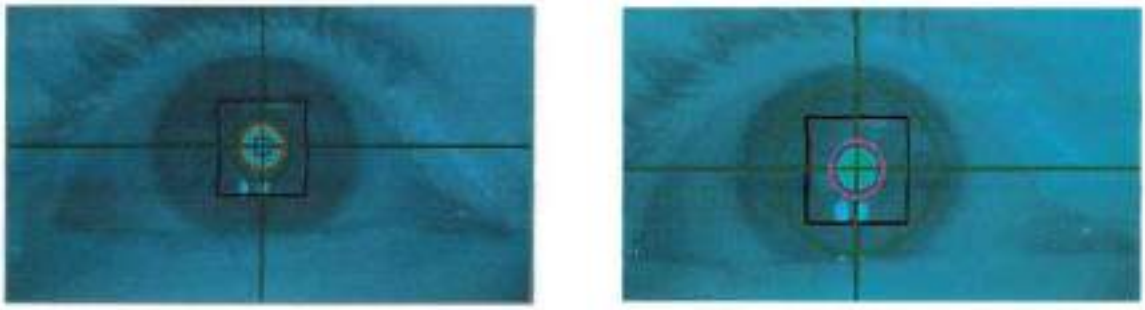


Figura 32 – Exemplo de calibrações [9]

Utilização da reflexão da córnea

O uso da câmera infravermelho cria reflexões no olho, que podem ser usadas para rebuscar o rastreamento. O rastreamento por pupila e reflexão é mais robusto, aumentando a precisão do sistema e acrescentando certa tolerância para movimentos da cabeça.

Calibração

Esta seção de configuração permite definir quantos serão os pontos de calibração, e se eles aparecerão na tela seguindo uma ordem ou aleatoriamente. Quanto maior o número de pontos escolhidos, maior a precisão da calibração, porém maior também será seu tempo de duração.

Outras configurações

Existem também outras configurações mais específicas, como, por exemplo, a escolha da resolução da câmera, ou habilitar o modo *smoothing*, que detecta a natureza do movimento do olho, como fixações do olhar.

3.3.2.2 Calibração

O processo de calibração é utilizado para o mapeamento entre uma posição da pupila e as coordenadas na tela. O *software* oferece nove, doze ou dezesseis pontos de calibração. Na figura 33 é possível ver um exemplo de uma calibração com 9 pontos. Durante a calibração, é importante manter a cabeça imóvel, e seguir os pontos na tela o tempo todo, para não criar nenhuma imprecisão.

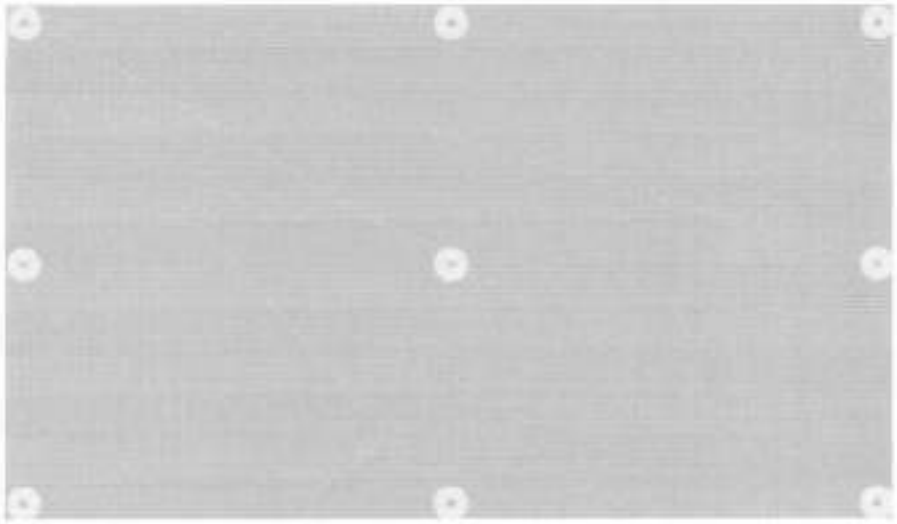


Figura 33 – Tela de calibração com 9 pontos fornecida pelo ITU Gaze Tracker

Pode-se verificar também a eficiência do processo de calibração, através da opção "*Display Visual Feedback*". Esta opção desenhará círculos na tela que mostram a precisão e o desvio em cada ponto. Nas figuras 34, 35 e 36 são mostrados três possíveis resultados de calibração. No primeiro deles, é possível ver um círculo vermelho próximo ao círculo branco. Isso significa que o usuário olhou para a região correta da tela, delimitada pelo círculo branco, porém não fixou seu olhar no centro, gerando um desvio grande. Por isso, caracteriza-se esse resultado como *offset* pequeno e desvio grande. No segundo resultado, o usuário não olhou para a região correta, gerando um *offset* maior. Porém, fixou seu olhar nesse ponto gerando um desvio pequeno, caracterizado por um círculo verde. Esse resultado é preferível em relação ao primeiro. No terceiro e último resultado, o usuário olhou

para uma região muito distante daquela que era estipulada, gerando um *offset* muito grande. Resultados assim exigem uma nova calibração.

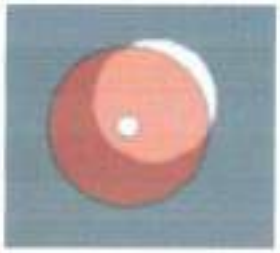


Figura 34 – Ponto de calibração com *offset* pequeno e desvio grande

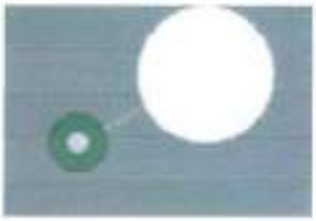


Figura 35 - Ponto de calibração com *offset* médio e desvio pequeno

Figura 36 - Ponto de calibração com *offset* grande

Também, como parte do resultado da calibração, é mostrada uma classificação de uma a cinco estrelas, caracterizando quão boa foi a calibração.

3.3.2.3 *Eye Mouse*

Para testar e verificar o rastreamento do olhar, o *software* ITU *Gaze Tracker* fornece também o *Eye Mouse*. Com esta ferramenta é possível controlar o cursor do *mouse* com os movimentos dos olhos. Através desta opção, será avaliado o

funcionamento do *software*, sendo que os resultados é o método para os testes serão abordados em tópicos mais adiante.

Vale citar também que, para completa experiência do *mouse*, ou seja, para habilitar as funções dos botões direito e esquerdo do mouse pode-se utilizar outro software, criado pelo mesmo grupo que criou o *Gaze Tracker*, intitulado *Gaze Mouse*. Nele há várias configurações que podem ser alteradas para proporcionar o melhor uso do *mouse* com o movimento do olho.

3.4 Projeto Eye Writer

O terceiro projeto abordado pelo grupo foi o projeto *Eye Writer*. Desenvolvido por membros de cinco grupos distintos, *Free Art and Technology*, *OpenFrameworks*, *The Grafitti Research Lab*, *Ebelin Group* e *Parsons Communication Design and Technology*, este projeto foi idealizado para ser utilizado em portadores de esclerose lateral amiotrófica (ELA). Basicamente, ele permite que artistas e grafiteiros com paralisias resultantes da ELA sejam capazes de desenhar utilizando apenas o movimento do olhar.

3.4.1 *Hardware*

O hardware original do projeto *Eye Writer* difere um pouco do construído para ser utilizado no ITU *Gaze Tracker*. Ao invés de utilizar uma webcam, o projeto original utilizou uma câmera para o console *Playstation* 3, a PS3 *Eye*. Este modelo, porém, se comparado com uma *webcam* comum, apresenta custos mais altos. Desta maneira o grupo optou por utilizar o mesmo hardware construído anteriormente, ou seja, com a mesma câmera. Além disso, para melhorar a precisão do rastreamento, uma imagem mais próxima do olho é necessária para uma melhor identificação de seus componentes e fronteiras (íris, limbo, pupila, etc.). A solução encontrada foi utilizar uma lente de aumento, do tipo que é utilizada em câmeras de segurança. Para o projeto, foi escolhida uma lente de 8mm.

Outra modificação veio devido a um problema encontrado com a lente de 8mm. Esta lente não é compatível com a câmera da Microsoft que estava sendo utilizada, já que o suporte para a lente era menor. Dessa maneira foi necessária a utilização de outra câmera do tipo *webcam*. Felizmente, essa mudança trouxe até

uma otimização de custo para o projeto, por se tratar de uma câmera muito mais barata (R$20,00 ao invés de R$90,00).

3.4.2 Software

O *software* do *Eye Writer* é formado por dois programas: um para *eye tracking*, com a possibilidade de testar o rastreamento com uma aplicação em que o usuário necessita olhar para um retângulo na tela, e outro para *eye drawing*, em que o usuário tem a possibilidade de criar desenhos com os movimentos dos olhos. Ambos foram desenvolvidos em C++ e utilizam a biblioteca Openframeworks que fornecem as ferramentas necessárias para projetos que envolvem expressões artísticas como no caso do projeto *Eye Writer*.

O *eye drawing* é destinado para uso em plataformas Mac, sendo que o programa foi desenvolvido no ambiente de desenvolvimento Xcode. Por isso, devido a essa limitação, apenas o programa de *eye tracking* foi utilizado, já que este último possui uma versão desenvolvida no ambiente Code::Blocks e compatível com Windows.

O *eye tracking* é formado por 3 modos distintos: configuração, calibração e teste. O primeiro deles, configuração, permite configurar parâmetros de imagem, como contraste e brilho, e também parâmetros para o rastreamento, como posição do olho e ajuste para captura da pupila. Na figura 37 é possível ver a tela para este modo.

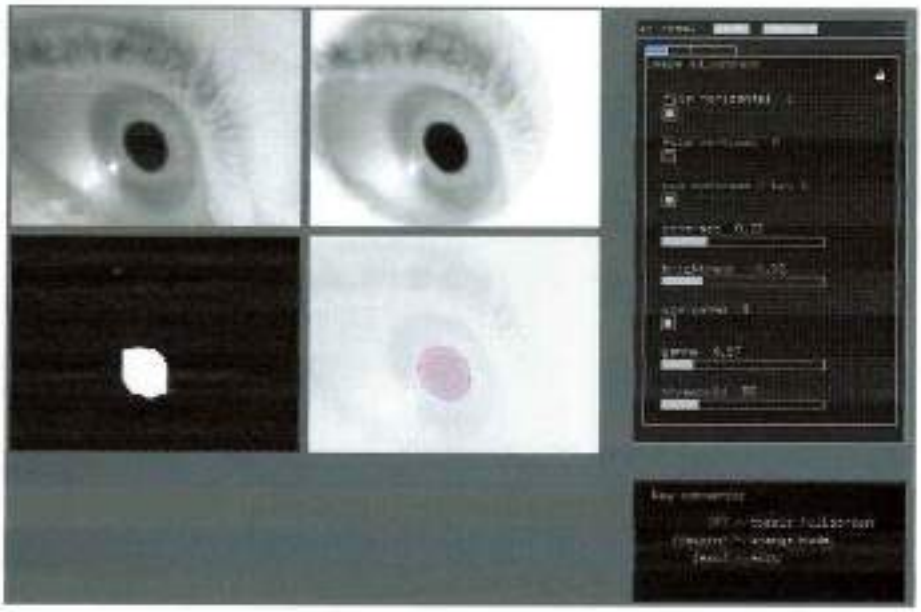


Figura 37 – Tela do modo de configuração do *eye tracking*

O modo calibração, como o próprio nome diz, é responsável pela calibração do sistema. O método é o mesmo utilizado pelo ITU Gaze Tracker, com a diferença de que não há nenhum *feedback* sobre a calibração. Nesse modo é possível também alterar o número de pontos utilizados para calibração e a velocidade com que esses pontos aparecem. Na figura 38 é possível ver a tela correspondente a este modo.

Figura 38 - Tela do modo de calibração do *eye tracking*

Por último, o modo de teste fornece uma opção de testar o rastreamento. Ele é composto por um retângulo, sendo que o usuário deve olhar para ele. Cada vez que acerta, o retângulo muda de posição ao longo da tela. Na figura 39 é possível ver a tela deste modo.

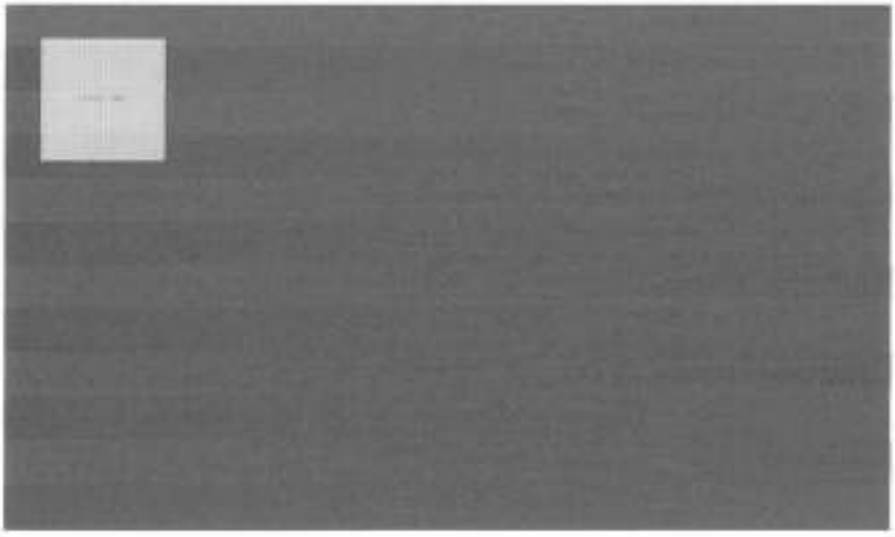


Figura 39 - Tela do modo de teste do *eye tracking*

A única limitação para o *eye tracking do projeto Eye Writer* é que o cursor fica limitado à janela do programa, diferentemente do ITU Gaze Tracker.

3.6 Considerações Finais

Neste capítulo foi possível conhecer em detalhes três projetos bastante interessantes de sistemas de rastreamento do olhar, destacar algumas otimizações que eles poderiam sofrer e algumas aplicações a que poderiam ser submetidos. Além disso, foram expostos itens fundamentais para a análise de qualquer projeto de engenharia, os resultados de uma série de testes nos quais os projetos foram submetidos.

Sobre os projetos em si, o primeiro estudado, o *OpenEyes*, foi o que mais possuía uma documentação técnica explicativa, com um artigo teórico descrevendo explicitamente cada item de seu algoritmo. Apesar disso, o *software* disponibilizado não funcionou corretamente, impossibilitando sua análise e obrigando ao grupo

descartá-lo. O ITU *Gaze Tracker*, analisado em sequência, foi o primeiro a apresentar um resultado mais interessante, mas ainda assim de forma bem abaixo da esperada, sofrendo bastante com a sensibilidade à vibrações do óculos, mas ainda assim sendo possível de se submeter a testes. Por fim, o que apresentou os melhores resultados sem dúvida foi o *software Eye Writer*, que surpreendeu positivamente com seu funcionamento.

No capítulo a seguir serão apresentados os resultados de uma análise comparativa entre dois dos projetos aqui citados e outras informações relevantes do projeto final.

4 RESULTADOS

4.1 Considerações iniciais

Para se chegar a uma conclusão da eficiência do rastreamento dos *softwares* até agora apresentados, foi realizada uma análise comparativa entre os sistemas. Para isto, adotou-se uma metodologia padrão, para que os testes fossem submetidos de maneira igual, de forma a gerar uma base que permita fazer comparações entre os *softwares*. Esses testes foram realizados utilizando o sistema de controle do cursor dos *softwares* dos projetos ITU *Gaze Tracker* e *Eye Writer*, sendo que o hardware utilizado para os testes é o mesmo. Além disso, estes testes também foram feitos com um *mouse* convencional e com o sistema *Touchad* de controle do cursor, que é bastante comum em computadores portáteis como *notebooks*. Isto permitirá comparar o controle convencional do cursor utilizando um mouse com o controle realizado pelo movimento ocular.

Por isso, neste capítulo serão apresentados os resultados encontrados bem como a metodologia adotada para estes testes. Serão abordados também alguns temas como orçamento do projeto final e as possíveis otimizações a serem feitas para melhorar a performance do sistema.

4.2 Metodologia de testes

O método idealizado para realizar os testes comparativos é bastante simples e foi baseado na saída padrão do *software* do projeto *Eye Writer*. Ele consiste basicamente em retângulos que aparecem aleatoriamente na tela, onde o objetivo do usuário é alcançá-los com o cursor da maneira mais rápida possível.

Para uma análise mais eficiente, foram feitos testes com dois tamanhos de retângulos diferentes com o objetivo de se verificar como se comportaria a diferença de velocidade e eficiência no caso da utilização de botões grandes e pequenos para cada controlador.

A realização desses testes no caso do *Eye Writer* foi possível através de um cronômetro e uma contagem direta dos retângulos que eram selecionados, além de uma modificação no *software*, para que em um segundo teste os retângulos que eram de tamanho padrão tivessem suas dimensões modificadas. Já para o caso do ITU *Gaze Tracker*, do *mouse* convencional e do *Touchpad* foi desenvolvido um

software que tivesse o mesmo efeito de exibição de quadrados aleatoriamente, já com um contador associado. No caso dos blocos grandes, no *software* desenvolvido foram utilizadas figuras de 130x130 pixels, ao passo que no caso dos blocos pequenos as dimensões eram de 67x67 pixels.

Os testes, para cada um dos quatro tipos de controlador de cursor, foram realizados vinte vezes, sendo dez para os blocos pequenos, e dez para os blocos grandes. Em cada tentativa, o objetivo era sempre levar o cursor para o maior número de blocos possível dentro de um intervalo de tempo. Vale notar também que, os testes feito com o ITU *Gaze Tracker* foram feitos em seguida e utilizando uma mesma calibração. O resultado desta calibração é apresentado na figura 40.

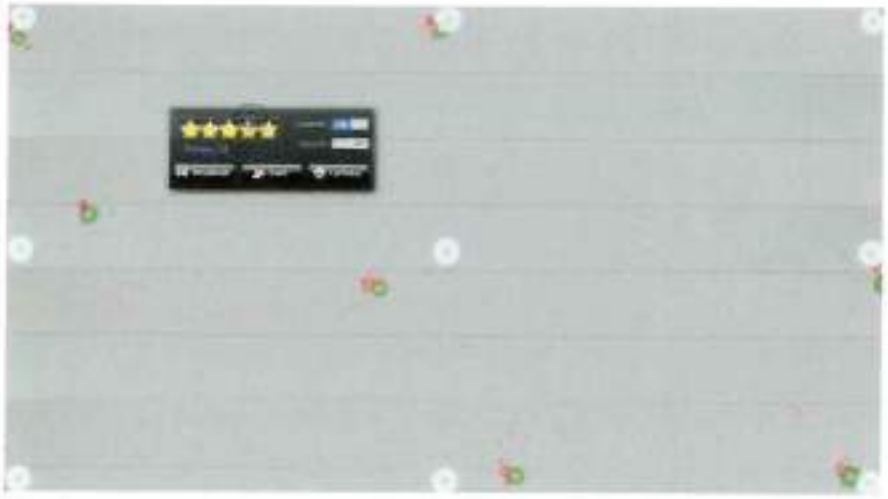


Figura 40 - Imagem da calibração obtida para os testes com o ITU *Gaze Tracker*

Os testes com os *softwares* para rastreamento foram feitos utilizando o mesmo protótipo. Na figura 41 é possível ver o equipamento em uso para um dos testes.

Figura 41 - Foto do protótipo em uso

4.3 Resultados utilizando os blocos pequenos

Utilizando os blocos de dimensão menor, para o caso de um *mouse* padrão, foi feito o levantamento de dados apresentado na Tabela 4, enquanto que os resultados para o *Touchpad* estão na Tabela 6. Nas Tabelas 7 e 8 é possível ver os resultados para o sistema de controle do *mouse* do ITU *Gaze Tracker*,e também os resultados do *Eye Writer*. Nas tabelas existem as colunas "Tempo" e "Blocos por segundo" unicamente para o critério de comparação não levar em conta algum eventual erro na contagem de tempo, já que ela é feita manualmente.

Tabela 4 - Testes utilizando *mouse* convencional com blocos pequenos

Teste	Total de blocos	Tempo (s)	Blocos por segundo
Teste 1	38	20	1,90
Teste 2	37	19	1,95
Teste 3	39	20	1,95
Teste 4	38	20	1,90
Teste 5	43	20	2,15
Teste 6	40	21	1,90
Teste 7	42	20	2,10
Teste 8	41	21	1,95
Teste 9	42	20	2,10
Teste 10	40	19	2,11
Média	**40**	**20**	**2,00**

Tabela 5 - Testes utilizando *Touchpad* de computadores portáteis com blocos pequenos

Teste	Total de blocos	Tempo (s)	Blocos por segundo
Teste 1	24	20	1,20
Teste 2	23	22	1,05
Teste 3	24	22	1,09
Teste 4	25	21	1,19
Teste 5	26	22	1,18
Teste 6	24	19	1,26
Teste 7	26	22	1,18
Teste 8	28	21	1,33
Teste 9	30	23	1,30
Teste 10	30	22	1,36
Média	**26**	**21,4**	**1,21**

Tabela 6 - Testes utilizando *ITU Gaze Tracker* com blocos pequenos

Teste	Total de blocos	Tempo (s)	Blocos por segundo
Teste 1	6	28	0,21
Teste 2	7	29	0,24
Teste 3	16	31	0,52
Teste 4	19	32	0,59
Teste 5	20	25	0,80
Teste 6	8	26	0,31
Teste 7	9	18	0,50
Teste 8	13	23	0,57
Teste 9	12	20	0,60
Teste 10	9	30	0,30
Média	**11,9**	**26,2**	**0,45**

Tabela 7 - Testes utilizando *Eye Writer* com blocos pequenos

Teste	Total de blocos	Tempo (s)	Blocos por segundo
Teste 1	10	20	0,50
Teste 2	17	24	0,71
Teste 3	7	22	0,32
Teste 4	11	23	0,48
Teste 5	18	25	0,72
Teste 6	16	30	0,53
Teste 7	12	26	0,46
Teste 8	11	28	0,39
Teste 9	13	22	0,59
Teste 10	16	24	0,67
Média	**13,1**	**24,4**	**0,54**

4.3 Resultados utilizando blocos grandes

De maneira análoga àquela que foi desenvolvida para as figuras pequenas na seção 3.7.2, as Tabelas 8, 9, 10 e 11 apresentam os resultados utilizando os blocos grandes para os sistemas submetidos ao mesmo teste.

Tabela 8- Testes utilizando *mouse* convencional com blocos grandes

Teste	Total de blocos	Tempo	Blocos por segundo
Teste 1	52	22	2,36
Teste 2	47	21	2,24
Teste 3	49	21	2,33
Teste 4	42	21	2,00
Teste 5	47	21	2,24
Teste 6	44	20	2,20
Teste 7	45	22	2,05
Teste 8	46	21	2,19
Teste 9	47	21	2,24
Teste 10	49	21	2,33
Média	**46,8**	**21,1**	**2,22**

Tabela 9 - Testes utilizando *Touchpad* de computadores portáteis com blocos grandes

Teste	Total de blocos	Tempo (s)	Blocos por segundo
Teste 1	34	21	1,62
Teste 2	34	23	1,48
Teste 3	31	22	1,41
Teste 4	37	21	1,76
Teste 5	33	21	1,57
Teste 6	34	21	1,62
Teste 7	40	23	1,74
Teste 8	33	22	1,50
Teste 9	34	22	1,55
Teste 10	34	23	1,48
Média	**34,4**	**21,9**	**1,57**

Tabela 10 - Testes utilizando *ITU Gaze Tracker* com blocos grandes

Teste	Total de blocos	Tempo	Blocos por segundo
Teste 1	14	15	0,93
Teste 2	23	30	0,77
Teste 3	17	17	1,00
Teste 4	14	16	0,88
Teste 5	11	20	0,55
Teste 6	24	20	1,20
Teste 7	31	26	1,19
Teste 8	31	27	1,15
Teste 9	15	15	1,00
Teste 10	17	22	0,77
Média	**19,7**	**20,6**	**0,96**

Tabela 11 - Testes utilizando *Eye Writer* com blocos grandes

Teste	Total de blocos	Tempo	Blocos por segundo
Teste 1	16	17	0,94
Teste 2	12	14	0,86
Teste 3	17	22	0,77
Teste 4	14	18	0,78
Teste 5	20	24	0,83
Teste 6	18	23	0,78
Teste 7	24	27	0,89
Teste 8	15	20	0,75
Teste 9	19	20	0,95
Teste 10	14	21	0,67
Média	**16,9**	**20,6**	**0,82**

4.4 Análise dos resultados

Comparando os resultados obtidos nas oito tabelas, é possível extrair uma grande quantidade de informações interessantes e pertinentes ao projeto. Dos dispositivos padrão para o controle do cursor, o *mouse* se mostrou sempre mais eficiente que o *Touchpad*, tanto para os quadrados grandes, como para os pequenos, com uma eficiência média total superior a 50%, sendo que para os quadrados menores essa superioridade passou de 60%, o que mostra que entre eles a precisão do *mouse* é bem maior que a do *Touchpad*.

Realizando uma comparação entre os dois *softwares* de rastreamento, houve certa surpresa nos resultados. Isso porque intuitivamente achava-se que o *Eye Writer* apresentaria para os dois tamanhos de quadrados resultados melhores, no entanto não foi isso que aconteceu. De fato, foi concluído que o *Eye Writer* é mais eficiente, sendo o sistema mais recomendado para esse tipo de aplicação, pois nela a precisão é fundamental, e ela pôde ser evidenciada no teste utilizando quadrados

pequenos. Ainda assim, não devem ser descartados os resultados obtidos pelo ITU *Gaze Tracker*, que evidenciam que o software traz certo controle do mouse com relativa agilidade, já que para os quadrados grandes a resposta foi consideravelmente satisfatória.

Numericamente, os resultados de ambos os *softwares* foram semelhantes, e quando comparados com o *mouse* convencional e com o *Touchpad*, é possível obter algumas informações interessantes. Para o *mouse*, o tamanho dos quadrados não influenciou muito, já para o *Touchpad*, a diferença foi semelhante à dos *softwares*. Em resumo, para tarefas que exigem precisão o *mouse* ainda é muito superior, mas para tarefas onde os botões são grandes e a agilidade não é tão necessária, os sistemas de rastreamento do olhar testados podem cumprir essa tarefa.

Dos resultados obtidos é possível concluir sobre os sistemas de rastreamento do olhar apresentados que eles podem ser úteis para mover o cursor e controlar um sistema com uma quantidade média de botões, já que eles proporcionam uma precisão razoável para essa tarefa.

4.5 Dificuldades e Limitações

O projeto teve suas principais dificuldades na montagem do hardware, devido à dificuldade do posicionamento correto da câmera e de se estabilizar o óculos.

O posicionamento da câmera é bastante complicado, já que o olho precisa estar bem centralizado na imagem que será processada, e a pupila deve ficar com um contraste grande em relação ao resto da imagem. Porém, o óculos também não pode ficar muito próximo ao olho, pois assim o campo de visão fica seriamente comprometido. Com a lente longe, aparecem algumas partes da imagem tão escuras como a pupila, como as sobrancelhas, o que causa o problema de, às vezes, o programa capturar essas outras partes da imagem como se fossem a pupila, o que é fatal para o bom funcionamento quando ocorre em momentos da calibração.

Este problema foi resolvido na etapa final do projeto, quando foi idealizada a colocação da lente de aumento para câmeras de segurança, já mencionada anteriormente.

A estabilidade do óculos também vem a ser um grande problema pois quando presa a câmera à armação, a mesma fica muito mais pesada desse lado. Além disso, apesar da haste ter ficado mais estável com o uso do termo retrátil, ainda não existe estabilidade absoluta. A câmera é bastante vulnerável às vibrações, o que degrada a precisão do protótipo.

Foram obtidas diversas limitações em relação ao planejado. A principal delas, é que era esperado não precisar imobilizar o rosto do usuário para conseguir fazer o rastreamento, mas pelo menos para a calibração, isto é necessário.

Além disso, cada pessoa que utiliza o protótipo deve modificar a câmera de acordo com seu rosto, para deixar seu olho no centro da imagem. Isto não é desejado, pois em um projeto de pesquisa, por exemplo, a intenção é testar seus resultados no maior número de pessoas. Se cada pessoa que utiliza o óculos alterar a posição do óculos, além de não ser cômodo, deixa a haste cada vez menos rígida, pois sua torção vai degradando o material, tornando-o cada vez menos rígido.

Este entrave é menor para o programa *Eye Writer*, pois nas configurações é possível a escolha de uma região de interesse dentro da imagem total. Isso faz com que o olho não precise estar necessariamente no centro da imagem, bastando somente a escolha devida da região de interesse. Isto pode ser visualizado na figura 42.

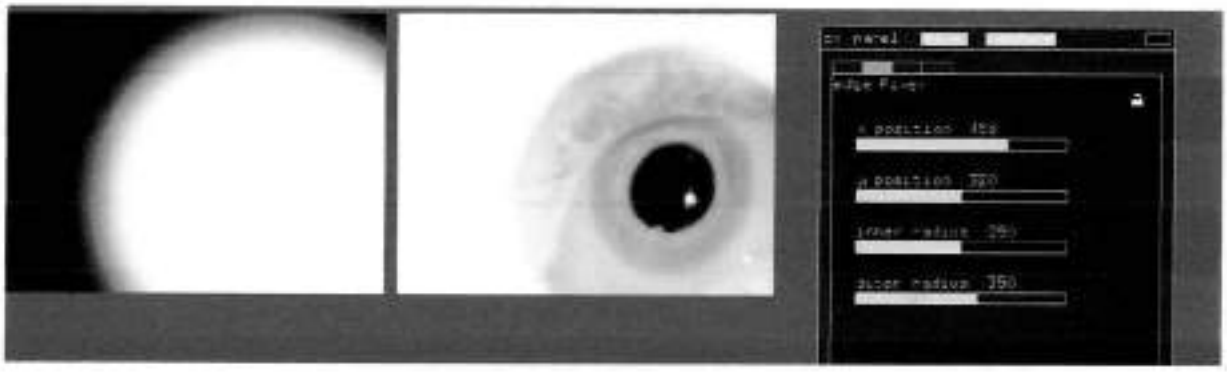


Figura 42 - Região de interesse no *Eye Writer*

4.6 Otimizações

Durante o projeto, sempre foi discutido como realizar melhorias para tornar o projeto mais robusto e inovador. Além de modificações no *software* e substituição de componentes que barateiam o projeto, também foram propostas a inserção de um filtro infravermelho (filtro *hot mirror*) e substituição do *laptop* por outra unidade de processamento menor (uma FPGA, por exemplo).

A implantação da unidade de processamento alternativa seria para dar maior mobilidade. Ela poderia ser levada em uma mochila pequena, por exemplo. Porém, não foi encontrado ferramenta com tal poder de processamento com um preço condizente com o planejado, e foi decidido que a relação custo-benefício não traria vantagens.

A inserção do filtro *hot mirror*, por sua vez, chegou mais perto de se concretizar. Inclusive, a mesma foi comprada. Na figura 43 pode-se ver o modelo do filtro adquirido. Entendia-se que seu uso seria bastante interessante, principalmente pela robustez do *hardware*, pois permitiria que a lente, se posicionada da maneira correta, não comprometeria o campo de visão do usuário. Além disso, a distância entre a câmera e o olho, apesar de passar por uma reflexão, tornaria-se menor, o que tornaria a imagem mais centrada no olho.

Figura 43 - Foto do filtro *hot mirror* adquirido

O que impediu essa otimização foi o tempo curto com que trabalhamos com a lente, devido ao tempo gasto nos testes da montagem normal. Essa montagem precisaria estar de acordo com cálculos de ótica, o que tornava-a ainda mais trabalhosa do que a primeira. Existia a necessidade de um maior processamento nesta otimização também, pois desta forma, ele deveria ser feito sobre uma imagem refletida e, consequentemente, de menor qualidade.

4.7 Orçamento

O orçamento final do projeto ficou abaixo do que esperado. A tabela 12 mostra o preço de cada componente utilizado sendo que em materiais diversos tem-se matérias como espaguete termo-retrátil, braceletes, parafusos, haste de alumínio e outros menores.

Tabela 12 – Preço dos componentes do sistema final projetado

Material	Preço
Webcam	R$20,00
Lente 8mm	R$25,00
Óculos de proteção	R$20,00
LED infravermelho	R$1,50
Materiais diversos	R$10,00

No total, projeto teve um gasto de R$ 76,50. Se comparado com outras soluções, como a fornecida pelo projeto openEyes, o preço teve um ótimo custo benefício. Apenas como comparação, a tabela 13 mostra os componentes para a construção do sistema descrito pelo projeto openEyes. Somando todos os custos, e selecionando as alternativas mais baratas para alguns itens, chega-se a um valor total de R$ 790,42, ou seja, dez vezes mais caro. Comparado com soluções oferecidas por empresas, essa diferença é muito maior. Por exemplo, algumas

empresas vendem seus produtos por preços superiores a US$10.000,00, ou seja, soluções extremamente caras.

Tabela 13 - Preços dos itens do *hardware* openEyes

Material	Preço
Aluminum Wire 14 gauge	R$9,44
Aluminum Wire 9 gauge	R$14,21
2 10' DB15 Male to Female cable	R$50,82
2 Aluminum Project Enclosures(13.3 x 7.6 x 5.4cm)	R$9,56
2 DB15 connectors	R$24,58
-I Board Camera (black and white)	R$246,09
-I Digital Camera (color)	R$174,18
5.7mm lens OR 12.0mm telephoto lens (eye lenses)	R$43,39 ou R$31,88
1.9mm wide angle lens OR 3.6mm medium angle lens (scene lenses)	R$60,72 ou R$43,39
17mm M12x0.5 Lens holder	R$9,59
13.5mm M12x0.5 Lens holder 13.5mm	R$9,59
940nm Infrared LED	R$2,86
4 14 pin dip sockets (do not use low profile sockets)	R$7,67
33 Ohm Resistor (0.75W or greater)	R$1,60
Zip Tiés	R$18,46
Standard Rosin-Core Solder	R$13,41
Electrical Tape	R$5,10
Wratten IR Filter	R$89,49
Shrink wrap tubbing	R$3,82
2 DB9 connectors	R$24,58

É importante citar também que em muitos projetos pesquisados, utiliza-se câmeras do tipo *webcam* com custos elevados. Por exemplo, é comum utilizarem a câmera modelo *Microsoft* VX-1000, com preço de R$90,00 ou mesmo a câmera PS3 Eye, com preço de US$40,00. Comparado com o modelo de câmera utilizado (preço

de R$20,00) há uma grande vantagem no custo permitindo diminuir ainda mais os gastos com o projeto.

4.8 Considerações finais

Neste capítulo foi possível observar uma análise quantitativa acerca dos *softwares* encontrados para rastreamento do olhar. Essa análise possibilitou que se fizessem comparações entre os sistemas de rastreamento e com os sistemas convencionais para controle de mouse. Com essa diferenciação, futuras aplicações, como teclados virtuais, devem levar em conta os resultados encontrados aqui para que se construa algo eficiente.

Foi possível também entender as dificuldades encontradas no projeto que, possivelmente trouxeram conseqüências para os resultados obtidos. O projeto também conta com algumas limitações que são amenizadas pelo baixo custo para construção do sistema como detalhado no orçamento.Algumas otimizações também foram sugeridas de modo a melhorar o projeto.

No capítulo seguinte será detalhado o gerenciamento do projeto ao longo de todo o tempo de execução do trabalho.

5 GERENCIAMENTO DO PROJETO

O projeto em si foi bastante extenso e trabalhoso, e para que sua execução corresse de acordo com o planejado ele foi subdividido em etapas menores que de maneira mais ampla foram ilustradas na forma de cartas de Gannt, que podem ser vistas nos Apêndices A e B. No entanto, com uma observação cuidadosa, em ambos os cronogramas é possível notar que em muitos momentos algumas das fases do projeto precisaram ser realizadas em paralelo, o que exigiu uma divisão de tarefas consideravelmente eficiente entre os membros da equipe.

Efetivamente, o gerenciamento do projeto em si pode ser dividido nas seguintes fases: Determinação da tecnologia, estudo detalhado sobre a técnica escolhida, definições sobre os *softwares* utilizados, desenvolvimento do protótipo, testes de validação, aplicações e monografia final. Dentro de cada fase, individualmente os membros do grupo assumiram tarefas distintas para que cada uma delas pudesse ser realizada com o sucesso esperado. Essa divisão pode ser vista nas tabelas a seguir, onde, para cada etapa, são mostradas as funções de cada um dentro dela.

Primeira Etapa: Determinação da tecnologia

A primeira parte do projeto foi a escolha da tecnologia que aconteceu realizando-se uma pesquisa teórica onde foram abordadas as técnicas que poderiam ser utilizadas para desenvolver o trabalho. Após isso ser feito foi possível optar por uma delas através de um confronto de pontos positivos e negativos ponderados também por questões como preço e disponibilidade de componentes.

Essa fase foi dividida internamente no grupo na forma descrita pela Tabela 14.

Tabela 14 - Etapa 1: Determinação da tecnologia

Carlos	Pesquisa teórica sobre o sistema de rastreamento por vídeo (constraste da íris)
Gustavo	Pesquisa teórica sobre o sistema de rastreamento por eletro-oculografia
Marcelo	Pesquisa teórica sobre o sistema de rastreamento por vídeo (infravermelho)

Após as pesquisas individuais de cada um, todos os membros se reuniram para determinar a técnica que seria utilizada no trabalho.

Segunda Etapa: Estudo detalhado da tecnologia

Após a técnica de rastreamento ter sido escolhida, outra parte fundamental para dar sequência ao projeto foi o seu estudo mais detalhado, já que tudo que seria feito após essa fase iria depender diretamente do conhecimento da tecnologia. Maos detalhes na Tabela 15.

Tabela 15 - Segunda Etapa: Estudo detalhado da tecnologia

Carlos	Pesquisa sobre os possíveis materiais que eventualmente seriam utilizados no projeto.
Gustavo	Pesquisa teórica mais aprofundada sobre sistemas de rastreamento que utilizam processamento de imagens.
Marcelo	Pesquisa teórica mais aprofundada sobre sistemas de rastreamento que utilizam processamento de imagens.

Terceira Etapa: Definições sobre os softwares utilizados

Mais um ponto fundamental para o projeto sem dúvida é a determinação de quais softwares seriam utilizados pelo grupo para realizar o rastreamento, além de eventuais sistemas adicionais que fossem necessários, por exemplo, para alguma aplicação. No caso, o trabalho exigiu a escolha de uma plataforma principal que inicialmente seria baseada no projeto *Open Eyes*, e que depois por questões técnicas foi substituída pelo ITU *Gaze Tracker*.

Essa fase foi uma das que mais exigiu esforços individuais focados, já que além de conhecimentos de programação, era preciso trabalhar em vários sistemas ao mesmo tempo. Na Tabela 16 é possível ver como foi a divisão dessa parte do trabalho.

Tabela 16 – Terceira Etapa: Definições sobre os softwares utilizados

Carlos	• Programação e adequação do código Matlab do algoritmo "*Starburst*" • Pesquisa teórica sobre o *software* ITU *Gaze Tracker* • Estudo sobre o *software* ITU *Gaze Tracker*, envolvendo calibração e configurações.
Gustavo	• Programação e adequação do código Matlab do algoritmo "*Starburst*" • Pesquisa teórica sobre o *software* ITU *Gaze Tracker* • Estudo sobre o *software* ITU *Gaze Tracker* envolvendo codificação.
Marcelo	• Estudo teórico do algoritmo "*Starburst*" (utilizado no sistema *Open Eyes*) • Desenvolvimento de software para captura de múltiplos vídeos simultaneamente. • Desenvolvimento de software para geração de *heat maps*.

Quarta Etapa: Desenvolvimento do protótipo

Depois de ter sido escolhida a tecnologia, em paralelo com os trabalhos no *software* era preciso iniciar a construção do protótipo do projeto. Essa fase envolve a pesquisa de componentes, montagem e integração do sistema como visto na Tabela 17

Tabela 17 - Desenvolvimento do protótipo

Carlos	• Pesquisa de componentes • Aquisição de materiais • Montagem do *Hardware*
Gustavo	• Pesquisa de Câmeras alternativas • Montagem do *Hardware* • Integração entre *Software* e *Hardware*
Marcelo	• Testes de Software • Montagem do Hardware • Integração entre Software e *Hardware*

Quinta Etapa: Testes de validação do protótipo

Uma parte muito importante após a construção do protótipo é a sua validação que deve ser feita para que seja possível assegurar e quantificar a qualidade e a eficiência do projeto. Para isso, foi utilizada uma série de calibrações do sistema que serviram para tornar o sistema mais confiável para as futuras aplicações (Tabela 18).

Tabela 18 - Divisão de tarefas: Testes

Carlos	Testes e elaboração de relatório de resultados
Gustavo	Testes e elaboração de relatório de resultados
Marcelo	Testes e elaboração de relatório de resultados

Sexta Etapa: Aplicações

Nesse ponto há novamente uma pesquisa teórica relacionada ao tema, assim como uma pesquisa de algoritmos próprios que realiza tal tarefa. São levantadas e analisadas aplicações pertinentes ao projeto desenvolvido efetivamente, ou seja, são levantados possíveis usos reais para o protótipo criado. Na Tabela 19 pode-se ver as divisões.

Tabela 19 - Aplicações

Carlos	Interface homem-máquina
Gustavo	Pesquisa médica
Marcelo	*Heat maps*

Sétima Etapa: Elaboração da Monografia Final

A elaboração da monografia final é a fase do projeto que terá a maior duração devido à sua complexidade e tamanho. Ela demanda uma quantidade considerável de tempo e precisa ser levada em conta no fluxo de desenvolvimento do projeto em si. Todos os membros da equipe participarão ativamente dessa etapa, seja na escrita do texto técnico, ou em sua fase final de revisão. Algo importante a ser ressaltado é que, boa parte do conteúdo dos relatórios anteriores à monografia final fará parte do texto final, diminuindo um pouco a complexidade da tarefa. Detalhes da divisão podem ser vistos na Tabela 20.

Tabela 20 - Monografia Final

Carlos	• Elaboração da Monografia final • Revisão da Monografia
Gustavo	• Gerenciamento e divisão de tarefas na elaboração da Monografia • Elaboração da Monografia final
Marcelo	• Elaboração da Monografia final • Revisão da Monografia

6 CONCLUSÃO

6.1 Resultados e Contribuições/Inovações Esperadas

Com relação aos resultados do projeto em si, não foram alcançados os resultados projetados inicialmente, porém acredita-se que se não fosse perdido tanto tempo no primeiro *software* utilizado o projeto teria saído como esperado. Falando dos resultados efetivamente obtidos, foram considerados bons apesar de suas limitações.

A mudança de *software*, entretanto, também colaborou bastante com o projeto, pois no primeiro (*openEyes*) apesar de descartado depois de um tempo, houve a possibilidade de trabalhar com o sistema operacional *Linux*. E trabalhar com os outros dois *softwares* permitiu uma comparação entre ambos, além do estudo de dois códigos escritos em linguagens diferentes.

Considerando as expectativas iniciais, o trabalho contribuiu da maneira esperada para o complemento da formação acadêmica. Isso porque para desenvolvê-lo está foi necessário utilizar não apenas o conhecimento passado através das disciplinas, mas também buscá-lo onde era necessário para que fosse possível executar determinadas etapas do projeto. Além disso, considerando um aspecto mais amplo, o trabalho mostrou como é efetivamente um projeto de engenharia em relação a vários pontos, como ao cumprimento de cronogramas, organização e seriedade na divisão de tarefas, elaboração de orçamentos, etc.

No fechamento, acredita-se que o trabalho serviu não apenas como uma conclusão de curso, mas também como uma complementação pessoal e profissional.

6.2 Considerações sobre o Curso de Graduação

De maneira geral o Curso atendeu bem às necessidades que se mostraram relevantes quando colocado no contexto deste projeto. Em alguns aspectos técnicos é possível faltar certa profundidade em temas abordados no Curso de Graduação, no entanto, isto é algo já esperado, uma vez que o projeto exige um conhecimento mais específico de determinados pontos, que podem ser estudados especialmente para o desenvolvimento do trabalho.

Este projeto se relacionou intimamente com a base de disciplinas que formam a estrutura principal da ênfase de sistemas eletrônicos. Nele são utilizados conhecimentos de programação e processamento de sinais (MAC0122 - Princípios de Desenvolvimento de Algoritmos; PCS2478 - Tópicos de Programação; PSI2432 - Projeto e Implementação de Filtros Digitais; PSI2533 - Modelagem em Processamento de Sinais;) e de eletrônica envolvendo sensores (PSI2326/PSI2327 Eletrônica Experimental I e II; PSI2662 - Projeto em Sistemas Eletrônicos Embarcados: Sensores e Atuadores;).

Das disciplinas relacionadas, muitas delas são optativas. Uma possível contribuição para o curso seria uma reavaliação do escopo de algumas das disciplinas obrigatórias que tem conteúdos que são repetidos em excesso, fazendo com que algumas delas possam ser retiradas da grade curricular, tornando possível cursar um conjunto maior de optativas. Outro fator que pode auxiliar bastante na complementação da formação dos alunos é tornar o conjunto de disciplinas optativas eletivas mais amplo, talvez envolvendo todos os departamentos de engenharia elétrica.

REFERÊNCIAS

[1] ASL - Applied Science Laboratories. **Choosing an Eye Tracking System**. Catálogo, 2011.

[2] ASL. **H6 Head Mounted Option**, Disponível em <http://www.asleyetracking.com/site/Products/EYETRAC6Series/HeadMounted/H6HeadMounted/tabid/58/Default.aspx>. Acesso em 24 de maio 2011.

[3] BULLING, A.; ROGGEN, D.; TROSTER, G. It's in your eyes – towards context-awareness and mobile HCI using wearable EOG googles. In: **International Conference on Ubiquitous Computing (UbiComp 2008)**, p. 84-93, 2008

[4] Cirrus. **Creative Webcam Vista**, Disponível em <http://www.cirrus.ca/shop/view/creative_webcam_vista_/103/>. Acesso em 28 de setembro de 2011.

[5] CLEMENS, J. **Open source eye tracking – free**, Disponível em <http://joelclemens.colinr.ca/eyetrack/software.html>. Acesso em 10 de julho de 2011.

[6] DONGHENG, L.; PARKHURST, D. J. **Starburst: A robust algorithm for video-based eye tracking.** Human Computer Interaction Program. Iowa State Univarsity, Iowa, 2005.

[7] DREWES, H. **Eye Gaze Tracking for Human Computer Interaction**. Dissertation an der LFE Medien-Informatik der Ludwig Maximilians-Universität München. Munique, 2010.

[8] DUCHOWSKI, A. T. **Eye Tracking Methodology – Theory and Practice**. 2. ed. London: Springer, 2007.

[9] IT University of Copenhagen. **ITU Gaze Tracker – A brief users guide**. IT University of Copenhagen, 2009.

[10] KOWALIK, M. **How to build low cost eye tracking glasses for head mounted system**, The West Pomeranian University of Technology, Szczecin, 2010.

[11] LOPEZ, J. S. A. **Off-the-Shelf Gaze Interaction**. PhD at IT University of Copenhagen. Copenhagen, 2009.

[12] MAIA, M. Processos bottom-up e top-down no rastreamento ocular de imagens. **Revista Veredas,** Juiz de Fora, v.12, p. 08-23, fev. 2008.

[13] Martin tall on gaze interaction.Tobii Glasses. **A new headmounted eye tracker**, Disponível em <http://gazeinteraction.blogspot.com/2010/06/tobii-glasses-headmounted-eye-tracker.html>. Acesso em 24 de maio de 2011

[14] MERCHANT, S. **Eye Movement Research in Aviation and Commercially Available Eye Trackers Today**. Operator Performance Laboratory. Iowa, 2001.

[15] ORSATI, F. T. ; MECCA, T. ; SCHWARTZMAN, J. S. ; MACEDO, E. C. **Percepção de faces em crianças e adolescentes com Transtorno Invasivo do Desenvolvimento**. (Paidéia) USP. Ribeirão Preto, v. 19, p. 349-356, 2009.

[16] RAUDONIS, V.; SIMUTIS, R.; NARVYDAS, G.. **Discrete Eye Tracking for Medical Applications**. Kaunas University of Technology. Lituânia, 2009.

[17] RICHARDSON, D. C.; SPIVEY, M. J. **Eye Tracking: Characteristics and Methods**. Encyclopedia of Biomaterials and Biomedical Engineering Wnek. [s.l.]: G. & Bowlin, G. (Eds.), 2004.

[18] Tobii no Brasil - Líder mundial em eye-tracking e em controle pelo olhar. **Cases – FHIOS**, Disponível em <http://www.tobii.com.br/telas/fhios/>. Acesso em 25 de maio de 2011.

[19] Tobii. **Research Eye Tracking Psychology Vision Usability**, Disponível em <http://www.tobii.com/en/analysis-and-research/global/products/hardware/tobii-glasses-eye-tracker/>. Acesso em 24 de maio de 2011.

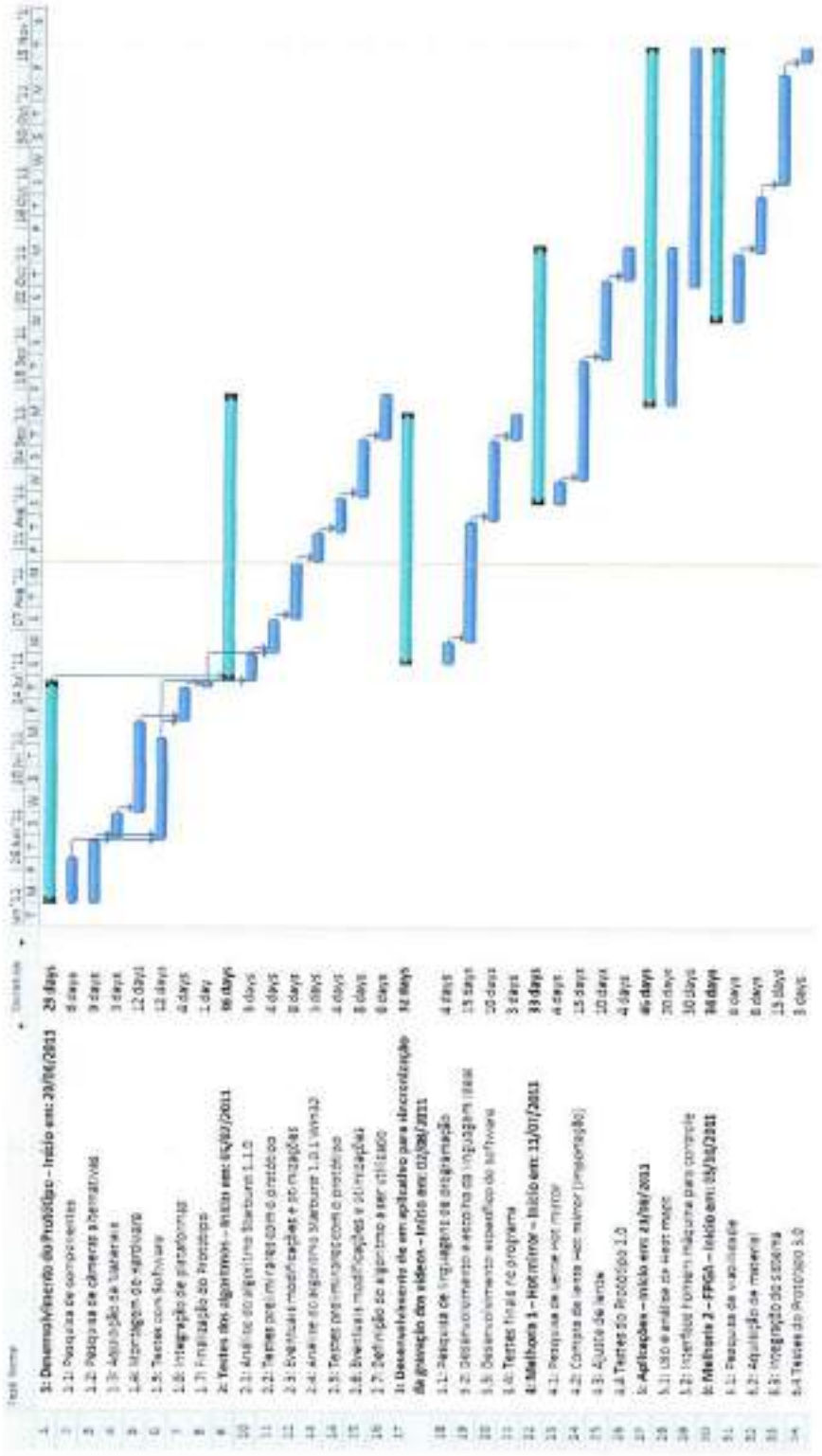
[20] TOPAL, C.; DOGAN, A.; GEREK O. N. **A Wearable Head-Mounted Sensor-Based Apparatus for Eye Tracking Applications**. Virtual Environments, Human-Computer Interfaces, and Measurement Systems. Turquia, 2008

[21] Universidade do Futebol. **Copa América de futebol 2011**. Disponível em <http://www.universidadedofutebol.com.br/Imprimir.aspx?id=11506&type=5> Acesso em 5 de outubro de 2011.

[22] WIED, P. **JavaScript Library for HTML5 canvas based *heat maps***. Disponível em <http://www.patrick-wied.at/static/heatmapjs>. Acesso em 20 de outubro de 2011.

[23] WINFIELD, D.; LI, D.; BABCOCK, J.; PARKHURST, D. J. Towards an open-hardware open-software toolkit for robust low-cost eye tracking in HCI applications. Iowa State University Human Computer Interaction **Technical Report**. Iowa State Univarsity, Iowa, 2005.

APÊNDICE A – Carta de Gannt elaborada em Junho



APÊNDICE B – Carta de Gannt elaborada em Setembro