Sistema de Rastreamento Ocular Usando Técnicas de Vídeo-Oculografia (VOG) como Tecnologia Assistiva

Davi de Alencar Mendes, João Paulo Sanches Guimarães

***Resumo*—Vídeo-Oculografia (VOG) refere-se ao processo de estimar a trajetória ocular e determinar a localização do olhar do usuário usando sequências de imagens coletadas. Nesse sentido, o presente trabalho propõe um sistema controlado a partir dos movimentos oculares para pessoas com deficiências físicas e pode ser considerado uma tecnologia assistiva. O sistema consiste em uma *Raspberry Pi 3 Model B+* equipada com uma câmera e um monitor. A interação do usuário com o sistema se dá a partir de opções segmentadas no monitor em formato de grade e a respectiva seleção de cada opção é efetuada de acordo com o movimento ocular efetuado pelo usuário.**

***Index Terms*—Eye-tracking, Vídeo-Oculagrafia (VOG), Tecnologia, Assistiva.**

I. Introdução

NA última década observa-se o vasto desenvolvimento de ferramentas que ampliam habilidades funcionais de pessoas com deficiência e consequentemente promovem qualidade de vida e independência pessoal [1]. Essas inovações receberam o nome de tecnologias assistivas. Embora essas tecnologias estejam presentes na mídia de maneira frequente ainda apresentam alto custo de aquisição e não estão amplamente difundidas. O desenvolvimento de um sistema de rastreamento ocular de maneira não invasiva baseado em uma *Raspberry Pi* pode trazer avanços ao desenvolvimento de tecnologias assistivas de menor custo e de maior portabilidade e que ainda assim sejam capazes de prover ao usuário independência e conforto por meio de opções interativas de comunicação [2].

II. Desenvolvimento

A. Panorama do Protótipo Funcional

A partir da concepção inicial do projeto foram realizadas discussões com uma pessoa com deficiência para propor adequações da interface para melhor atender um cenário real. Nesse sentido, foram elaboradas novas necessidades e requisitos funcionais para o projeto. As necessidades e requisitos funcionais do projeto são listadas a seguir na qual as adições aparecem em **negrito**.

- [NE01] Detectar faces
- [NE02] Detectar olhos
- [NE03] Estimar trajetória ocular
- [NE04] Interface gráfica para interação com usuário
- [NE05] Adaptação à diferentes condições de iluminação
- [**NE06**] **Controlar aparelhos ar-condicionado e televisão por meio da interface.**
- [**NE06**] **Enviar mensagem SMS padronizada em caso de emergência por meio da interface.**

Para o ponto de controle 2 é esperado que seja obtido um protótipo funcional que atenda os principais requisitos do projeto.

- [RF01] O sistema deverá detectar faces frontais usando classificador em cascata Haar
- [RF02] O sistema deverá detectar olhos usando classificador em cascata Haar
- [RF03] O sistema deverá processar imagem dos olhos para obter posição da iris (*Hough Circles* ou *Blob detection*)
- [RF05] O sistema deverá ser controlado a partir da trajetória ocular do usuário
- [RF06] O sistema deverá exibir uma interface de navegação para o usuário
- [**RF07**] **O sistema deverá possuir um controle infravermelho (IR) configurável para diferentes dispositivos**
- [**RF08**] **O sistema deverá enviar mensagens SMS padronizadas em situações emergenciais**

Quanto aos requisitos não funcionais, temos:

- [RNF01] O sistema deverá ser adequado para o usuário com deficiência motora dos membros superiores
- [RNF02] O sistema deverá permitir procedimento de calibração para adequar-se à diferentes condições de uso e a diferentes usuários

A partir do desenvolvimento dos requisitos citados podemos metrificar o andamento do projeto rumo aos objetivos. Como resultado do desenvolvimento desses requisitos apresentamos uma descrição de *software* e de *hardware* nas seções seguintes. Finalmente, apresentamos na seção de resultados comentários a respeito do progresso obtido para o presente ponto de controle.

B. Descrição de Hardware

1) Controle IR: Visando cumprir o requisito funcional [RF07], foi projetado um *hardware* capaz de se comunicar com aparelhos televisivos e de condicionamento de temperatura. Para ambos os casos, a comunicação via ondas na faixa do

Davi de Alencar Mendes é estudante de Graduação em Engenharia Eletrônica pela Universidade de Brasília - UnB, Brasília, Brasil. Email institucional: dmendes@aluno.unb.br - Matrícula: 16/0026415

João Paulo Sanches Guimarães é estudante de Graduação em Engenharia Eletrônica pela Universidade de Brasília - UnB, Brasília, Brasil. Email institucional: sanches.joao@aluno.unb.br - Matrícula: 16/0031923

infravermelho (IR) se mostrou a mais interessante, tendo em vista sua ampla utilização nesses setores.

Visando atingir um leque de aparelhos maior, o projeto escolhido para esse sub-sistema foi um que pudesse copiar uma onda recebida por um controle remoto por meio de um receptor IR e armazenar os dados de um dado comando, podendo replicá-lo a qualquer momento a partir de um emissor IR.

Pode-se visualizar na Figura 1 o esquemático representativo do *hardware* montado para cópia dos sinais IR. O arranjo com o transistor com os LEDs emissores se dá pela intenção de aumentar o alcance do sinal emitido, onde uma corrente vinda da base do transistor relativamente pequena proporciona um alto ganho de corrente que entra pelo coletor [7].

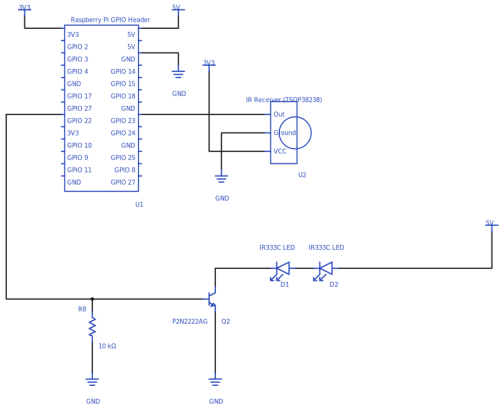

Figura 1. Esquemático do *hardware* do emissor e receptor IR

C. Descrição de Software

1) LIRC: O Linux Infrared Remote Control (LIRC) é um pacote que permite a decodificação e o envio de sinais IR da maioria dos controles remotos comumente utilizados. Sua parte mais importante é a *lirc daemon*, que decodifica sinais IR recebidos pelos *drivers* de dispositivos e disponibiliza a informação em um *socket*. Além disso, o LIRC aceita comando para envio de sinais IR, se o *hardware* assim permitir [8].

O LIRC oferece uma base de dados de diversos controles que podem ser utilizados, mas, mesmo que não seja encontrado o específico que desejarmos, poderemos escolher um semelhante (normalmente da mesma empresa) e assim usá-lo como um *template* e modificá-lo por meio de um *script* que nos permite inserir o nome de um botão (há uma lista de nomes possíveis para os comandos) e inserir o sinal por meio do receptor IR e assim gravá-lo para o controle personalizado [7].

2) Nexmo: Com a finalidade de atender ao requisito funcional [RF08], pesquisou-se opções de envio de SMS via *Raspberry Pi*. Surgiram opções que envolviam *hardware* adicional, como um moldem USB ou um módulo GSM, que utilizariam um chip SIM para enviar mensagens. Porém, há opções como Nexmo, que são plataformas online que fornecem esse serviço, facilitando a recarga de créditos, bem como reduzindo o custo do projeto, visto que só é necessário o acesso à internet, sem adição de *hardware*.

Em particular, a plataforma Nexmo permite o envio e gerenciamento de mensagens via *script*, possibilitando sua integração com a *Raspberry Pi* e com os outros subsistemas do projeto.

3) QtCreator: A fim de atender ao requisito funcional [RF06], pesquisou-se opções para auxilio da implementação de uma interface interativa. Encontro-se o QtCreator, um aplicativo que possibilita o projeto de uma interface de forma gráfica.

Essa abordagem facilita o desenvolvimento do projeto, gerando um código fonte em C++ (ou python) correspondente à interface gráfica, possibilitando a inserção de funcionalidades a partir de ações tomadas nas entidades da interface, proporcionando assim a interação desejada para acionar os outros subsistemas.

4) Detecção de Faces: A detecção de faces constitui a primeira necessidade do projeto ([NE01]) englobando os requisitos [RF01] e [RNF02]. Como já havia sido pesquisada um método de implementação por meio de classificadores em cascata de *Haar* estes foram implementados por meio da biblioteca de visão computacional de tempo real *OpenCV Python* 3.4.6-dev. Ademais, o sistema permite somente a utilização de um usuário por vez, filtrando somente a maior face presente no vídeo obtido. Além disso, para melhorar a detecção de faces usando a respectiva imagem em escala de cinza foi realizada uma equalização de histograma para ajustar o contraste da imagem melhorando a performance do classificador utilizado. A etapa subsequente a detecção de faces consiste na detecção de olhos a partir da imagem da face detectada conforme pode ser visto na figura a seguir.

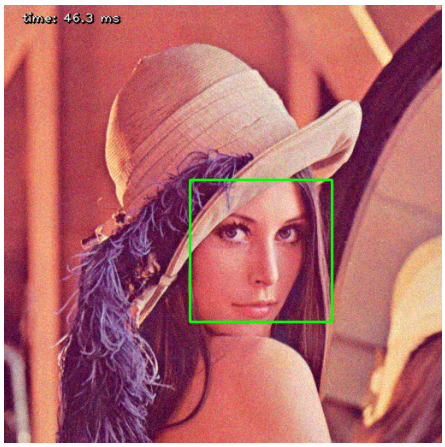

Figura 2. Detecção de face com *Haar Cascade*

5) Detecção de Olhos: A detecção de olhos constitui a segunda necessidade do projeto ([NE02]) englobando os requisitos [RF02] e [RNF02]. Novamente, foi utilizado um classificador em cascata de *Haar* para detectar os olhos presentes em uma face. Após a extração dos olhos de uma face, a imagem é cortada parcialmente na parte superior para evitar a presença das sobrancelhas nas etapas subsequentes de processamento dos olhos.

Inicialmente, a imagem em escala de cinza da face detectada é binarizada de acordo com um limiar que pode

ser configurado por meio de uma *trackbar*, que auxilia o sistema a funcionar em diferentes condições de iluminação conforme previsto em [RNF02]. Em seguida, são aplicadas operações morfológicas para que o resultado final se assemelhe com um circulo que compõe a íris do olho. Essas etapas de processamento podem ser vistas nas figuras a seguir para ambos os olhos.

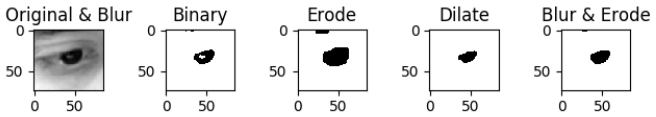

Figura 3. Processamento do Olho Esquerdo

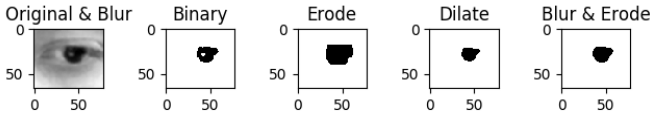

Figura 4. Processamento do Olho Direito

O resultado desta etapa é a última imagem vista em 3 e 4. Esta imagem binária é, posteriormente, inserida em um *blob detector*. Este tipo de detector busca *blobs* que são regiões na qual alguma propriedade da imagem é mantida aproximadamente constante como forma, área, brilho ou outros se comparada às regiões vizinhas. Para extrair a iris podemos supor uma área média e uma inércia alta (já que a íris se assemelha com um círculo). Nesse sentido, esse parâmetros são utilizados no detector para ajustar sua performance. Além disso, os *blobs* detectados são filtrados para evitar falsos positivos. Para o principal *blob* é desenhado um circulo e são marcadas as coordenadas do centro do *blob* para referência. O resultado desta etapa conforme foi descrito pode ser visto na figura a seguir.

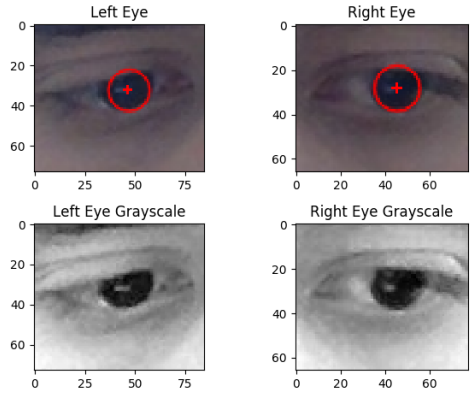

Figura 5. Detecção de Olhos e Iris

Em suma, até a presente etapa está implementada a VOG ([RF01] - [RF02] - [RNF02]) completando as 2 primeiras necessidades do projeto. Com a localização aproximada obtida para a pupila pode-se estimar a trajetória ocular.

6) Estimação da Trajetória Ocular: A estimação da Trajetória Ocular constitui a terceira necessidade do projeto ([NE03]) e estabelece que o sistema deverá ser controlado a partir da trajetória ocular ([RF06]).

Ao longo das pesquisas iniciais e do desenvolvimento dos requisitos citados foi constatado uma certa escassez de material a respeito da estimação de trajetória e os métodos testados não mostraram bons resultados. Para a presente etapa do projeto esse requisito está sendo implementado e requer testes posteriores. Alguns fatores dificultam o procedimento de estimação como movimento da cabeça, perda dos *blobs* entre outros.

III. Resultados

Visando os objetivos do Ponto de Controle 2, fez-se testes separadamente para cada um dos subsistemas, com intuito de efetuar uma validação prática destes. Em particular, para validação do Controle Infravermelho, foi construído o esquemático mostrado na Figura 1, foi criado um arquivo com base no *template* de um controle e testado diferentes comandos personalizados, cumprindo suas funcionalidades sem falhas.

Quanto ao envio de mensagem, foi testado um *script* inicial na plataforma Nexmo e foi validado seu funcionamento com o envio de mensagens de testes via SMS. A criação de uma interface gráfica de fase inicial foi efetuado, bem como testes que validassem a possibilidade de interação com o código fonte em C++, o que mostra que será possível a futura alteração para interação com os outros subsistemas.

No que concerne a implementação da VOG - detecção de faces e olhos e posterior processamento dos olhos - observa-se um progresso satisfatório já que os requisitos foram bem implementados e testados, estando prontos para serem implementados em C++ utilizando a mesma biblioteca para aumento de performance e robustez. Entretanto, a estimação da trajetória ocular ainda não está implementada e será foco até o próximo ponto de controle do projeto.

IV. Conclusão

Tendo em vista os resultados alcançados, vemos que a grande maioria dos subsistemas já estão em pleno funcionamento e estão prontos para serem sintetizados em um grande projeto. Visa-se também re-implementar os *scripts* da VOG de Python para C++, principalmente por esperar-se uma melhora significativa na performance dos algoritmos.

Temos boa indicação de que isso será possível visto que a principal biblioteca utilizada é o OpenCV, que tem sua versão tanto em Python quanto em C++. Vale lembrar que o código criado pelo QtCreator já está em C++, bem como o Controle IR pode ser operado com comandos no terminal, que serão implementados via código C/C++.

Espera-se efetuar um refinamento principalmente na estimativa de posicionamento relativo do olho na VOG, visando uma implementação robusta da estimação da trajetória ocular. Esta que irá permitir a manipulação visual do usuário por meio do controle da interface gráfica.

REFERÊNCIAS

[1] Singh, Hari. (2012). Human Eye Tracking and Related Issues: A Review. International Journal of Scientific and Research Publications. 2. 1-9.

[2] Sadeghian, Amir & Yim, Chol-Ho & Chan, Adrian & Green, James. (2007). Eye-Interact: A Low-Cost Eye Movement Controlled Communication System. 10.13140/2.1.2943.9046.

[3] Phung, Manh Duong & Vinh, Tran & Hara, Kenji & Inagaki, Hirohito & Abe, Masanobu. (2008). Easy-setup eye movement recording system for human-computer interaction. 292 - 297. 10.1109/RIVF.2008.4586369.

[4] Speech generating device with eye tracker (Acessado em 26 de março de 2019) Retirado de https://www.tobiidynavox.com/devices/eye-gaze-devices/I-12/#Overview

[5] Eye-driven tablet communication system (Acessado em 26 de março de 2019) Retirado de https://eyegaze.com/products/eyegaze-edge/

[6] Pohl, Klaus. Requirements engineering: fundamentals, principles, and techniques. Springer Publishing Company, Incorporated, 2010.

[7] Automated IR Remote Control Using a Raspberry Pi and LIRC (Acessado em 2 de maio de 2019) Retirado de https://www.fad-diyit.com/automated-ir-remote-control-using-a-raspberry-pi-and-lirc/2/

[8] Linux Infrared Remote Control (Acessado em 2 de maio de 2019) Retirado de http://www.lirc.org/