UNIVERSIDADE FEDERAL FLUMINENSE

EYRE BRASIL BARBOSA MONTEVECCHI

PROVENDO INTERAÇÕES OCULARES EM APLICAÇÕES GINGA-NCL

NITERÓI

2022

EYRE BRASIL BARBOSA MONTEVECCHI

PROVENDO INTERAÇÕES OCULARES EM APLICAÇÕES GINGA-NCL

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: CIÊNCIA DA COMPUTAÇÃO.

Orientadora:
DÉBORA CHRISTINA MUCHALUAT SAADE, D.Sc.

NITERÓI

2022

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

M772p Montevecchi, Eyre Brasil Barbosa
Provendo Interações Oculares em Aplicações Ginga-NCL / Eyre Brasil Barbosa Montevecchi ; Débora Christina Muchaluat Saade, orientadora. Niterói, 2022.
103 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Niterói, 2022.

DOI: http://dx.doi.org/10.22409/PGC.2022.m.07649609626

1. Multimídia interativa. 2. NCL (Linguagem de marcação de documento). 3. Sistemas multimídia. 4. Televisão digital. 5. Produção intelectual. I. Muchaluat Saade, Débora Christina, orientadora. II. Universidade Federal Fluminense. Instituto de Computação. III. Título.

CDD -

Bibliotecário responsável: Debora do Nascimento - CRB7/6368

EYRE BRASIL BARBOSA MONTEVECCHI

PROVENDO INTERAÇÕES OCULARES EM APLICAÇÕES GINGA-NCL

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: CIÊNCIA DA COMPUTAÇÃO.

Aprovada em abril de 2022.

BANCA EXAMINADORA

Profa. DÉBORA CHRISTINA MUCHALUAT SAADE - Orientadora, UFF

Profa. FLÁVIA COIMBRA DELICATO, UFF

Prof. MARCELO FERREIRA MORENO, UFJF

Niterói

2022

Agradecimentos

Agradeço muito ao meu marido, André Montevecchi, por me incentivar, se interessar e sempre me apoiar durante todo o mestrado! Nossas conversas foram fundamentais para eu iniciar essa caminhada e sem você não seria possível a conclusão desta pesquisa! Agradeço por toda a compreensão, companheirismo e motivação!

Gostaria de agradecer aos meus familiares pelo apoio, torcida e todo o interesse demonstrado pela minha pesquisa. Obrigada por me ajudarem a construir a base necessária para chegar até aqui!

Um muito obrigado à minha maravilhosa orientadora Débora Saade! Sempre muito animada e compreensiva, me motivou e me inspirou muito! Obrigada pela paciência e muito estimada orientação! Sinto muito orgulho de ter tido você como orientadora e do seu trabalho como pesquisadora e professora!

Agradeço aos maravilhosos amigos que fiz durante essa caminhada do mestrado Fábio Barreto, Marina Josué e Raphael Abreu. Sem vocês, não seria possível concluir esta pesquisa! Muito obrigada pela amizade, discussões, apoio e motivação!

Agradeço aos professores que me incentivaram a iniciar o mestrado Fábio Barreto, Débora Saade, João Fernandes, Diego Passos e Mário João. Um agradecimento especial ao Fábio Barreto que me incentivou, inspirou e acompanhou desde a graduação, se transformando em colega de pesquisa e amigo durante o meu mestrado!

Gostaria de agradecer também à CAPES, CNPq e FAPERJ pelo apoio financeiro parcial deste trabalho.

Resumo

Sensores de rastreamento ocular (*eye tracking*) permitem que os usuários interajam pelo olhar em aplicações multimídia, proporcionando uma nova modalidade de interação para aqueles que não querem ou não podem usar os dispositivos tradicionais como mouse, teclado e controle remoto. No contexto das aplicações de TV digital, o *middleware* Ginga-NCL não suporta ainda esse tipo de interação. Esta dissertação de mestrado propõe uma extensão do Ginga-NCL para proporcionar interação baseada em fixação ocular usando a tecnologia *eye tracking* e permitir o uso de um novo tipo de evento para autoria de aplicações multimídia interativas na linguagem NCL (*Nested Context Language*), denominado *EyeGaze*. Além disso, esta dissertação propõe uma arquitetura de integração, que pode ser utilizada na integração de sensores *eye trackers* e se adapta à definição atual de Ginga e de NCL, para facilitar sua adoção. Esta dissertação apresenta também como estruturar os dados das mídias associadas ao evento *EyeGaze* para enviá-los ao módulo de interação ocular, denominado *Gaze Recognition Module*, utilizando objetos JSON (*JavaScript Object Notation*). O paradigma GQM (*Goal Question Metric*) foi usado para estruturar a avaliação da solução desenvolvida nesta dissertação, em que o evento *EyeGaze* para interação com a TV digital foi analisado em relação à experiência dos usuários, aceitação da tecnologia e desempenho. Os objetivos definidos foram alcançados ao final da etapa de avaliação, observando-se bons resultados em relação à experiência do usuário, considerada excelente para todas as escalas e classificações do UEQ (*User Experience Questionnaire*). Além disso, teve avaliações positivas em afirmações relacionadas a conforto, utilidade percebida e percepção da facilidade de uso. Um bom desempenho também foi observado sob ponto de vista do sistema e dos usuários.

Palavras-chave: Rastreamento ocular, Ginga, NCL, Interação por fixação ocular, Aplicação Multimídia, TV Digital.

Abstract

Eye tracking sensors allow users to interact by eye gaze with multimedia applications, providing a new interaction modality for those who do not want or cannot use traditional devices such as a mouse, keyboard, and remote control. Considering digital TV applications, middleware Ginga-NCL does not support this type of interaction yet. This dissertation proposes a Ginga-NCL extension to provide eye-gaze interaction using eye tracking technology and enable a new event type for the NCL (Nested Context Language) multimedia authoring language, called EyeGaze. Furthermore, this dissertation proposes an integration architecture, which can be used to integrate eye tracker sensors and adapts to the current definition of Ginga and NCL to facilitate its adoption. This dissertation also presents how to structure the media data associated with the EyeGaze event to send them to the eye gaze interaction module, called the Gaze Recognition Module, using JSON (JavaScript Object Notation). The GQM (Goal Question Metric) paradigm was used to structure the evaluation of the solution developed in this dissertation. The EyeGaze event for interaction with digital TV was analyzed concerning user experience, technology acceptance, and performance. The defined objectives were achieved at the end of the evaluation stage, observing good results about the user experience, which is considered excellent for all scales and classifications of the UEQ (User Experience Questionnaire). In addition, it had positive evaluations in statements related to comfort, perceived usefulness, and perceived ease of use. A good performance was also observed from the point of view of the system and users.

Keywords: Eye tracking, Ginga, NCL, Eye Gaze Interaction, Multimedia Application, Digital TV.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

AAC Augmentative and Alternative Communication

API Application Programming Interface

ELA Esclerose Lateral Amiotrófica

GQM Goal Question Metric

HTML5 HyperText Markup Language

IPTV Internet Protocol Television

IR Radiação Infravermelha

JSON JavaScript Object Notation

NCL Nested Context Language

NCM Nested Context Model

PEOU Percepção de Facilidade de Uso

PU Utilidade Percebida

SBTVD Sistema Brasileiro de Televisão Digital

SDK Software Development Kit

SMIL Synchronized Multimedia Integration Language

TAM Technology Acceptance Model

UEQ User Experience Questionnaire

URI Uniform Resource Identifier

XML Extensible Markup Language

Sumário

1 Introdução

Em geral, aplicações multimídia interativas permitem que o usuário interaja com a informação transmitida, seja através da seleção de algum conteúdo na tela, ou até mesmo no controle de reprodução das mídias em apresentação (e.g. avançar, retroceder, pausar). Ao adicionar interação em aplicações multimídia, é possível oferecer diferentes tipos de serviços, como jogos, aplicações educacionais, simulações, ou mesmo aplicações de turismo, por *tours* virtuais. Em sistemas multimídia tradicionais, a interação normalmente se dá por dispositivos como *mouse*, teclado, controle remoto ou *joysticks* para jogos. Entretanto, existem outras formas de interação como interação por voz, por gestos ou por meio do olhar, que podem proporcionar maior sensação de imersão, visto que são mais naturais no cotidiano dos usuários.

De acordo com Tanriverdi e Jacob (2000), a interação por meio do movimento dos olhos oferece potencial como uma maneira mais fácil, natural e rápida de interagir em ambientes virtuais. Isto porque o uso do olhar para apontar é mais rápido do que apontar usando gestos com as mãos. Com isso, essa tecnologia encontra diversos usos em multimídia, por exemplo, para interação em jogos (SUNDSTEDT, 2010). Além disso, sensores rastreadores oculares (*eye trackers*) proveem uma nova forma de interação para usuários que não querem ou, devido à alguma deficiência motora, não podem utilizar as modalidades tradicionais por meio do mouse, teclado ou controle remoto, por exemplo.

As aplicações multimídia estão disponíveis em várias plataformas como *smartphones*, computadores, *tablets* e TVs digitais. Para que essas aplicações sejam desenvolvidas, é necessária a utilização de uma linguagem de autoria hipermídia, possibilitando a especificação das propriedades temporais e espaciais das mídias que compõem essas aplicações (SOARES; RODRIGUES; MUCHALUAT SAADE, 2000).

Existem diferentes linguagens de autoria para desenvolver aplicações multimídia. Parte dessas linguagens é declarativa e baseada em *Extensible Markup Language (XML)* (W3C, 2008a) e especificam aplicações multimídia focando na definição da sincronização

de objetos de mídia, independentemente do conteúdo deles. Exemplos de linguagens de autoria multimídia são a *Synchronized Multimedia Integration Language (SMIL)* (W3C, 2008b), *HyperText Markup Language (HTML5)* (W3C, 2014) e *Nested Context Language (NCL)* (ANTONACCI et al., 2000).

A linguagem NCL é padrão para desenvolvimento de serviços interativos no Sistema Brasileiro de Televisão Digital (SBTVD) (ABNT, 2011), sendo utilizada para especificação de conteúdo interativo para a TV digital e para serviços *Internet Protocol Television (IPTV)* conforme o padrão ITU-T H.761 (ITU-T H.761). Essa linguagem é interpretada pelo *middleware* Ginga-NCL (ABNT, 2011) e usa o paradigma de sincronização baseado em eventos (SOARES; RODRIGUES; MUCHALUAT SAADE, 2000), permitindo que a relação temporal entre as mídias seja especificada de forma relativa, e não considerando o tempo absoluto em uma linha temporal (HARDMAN et al., 2000). As relações de sincronização são definidas no documento NCL, e os seus momentos de início e fim são determinados em tempo de execução. Em NCL, um evento pode estar relacionado à exibição de um determinado conteúdo (e.g., início e fim da apresentação), ou alguma forma de interação do usuário (e.g., a seleção de um botão do controle remoto). Para permitir novas modalidades de interação com o usuário, seria interessante oferecer outros tipos de eventos em NCL. A versão atual do Ginga-NCL assim como a linguagem NCL não suportam a interação utilizando o olhar, embora Barreto (2021), Barreto, Abreu, Montevecchi et al. (2020) e Fábio Barreto, Montevecchi et al. (2019b) tenham proposto a extensão da linguagem e a criação de um evento para interação baseada na tecnologia de *eye tracking*.

Existem diferentes tipos de interação por meio do olhar, tais como apontar, gestos oculares, seleção por tempo de permanência, arrastar e soltar, selecionar seguindo um objeto em movimento com o olhar e leitura (FEIT et al., 2017). Essas interações se baseiam em eventos oculares, como, por exemplo, as fixações e sacadas. Uma fixação ocular indica um período (de cerca de 100 ms até vários segundos) durante o qual a localização do *gaze point* permanece em uma região (FEIT et al., 2017). Já o evento de sacada refere-se à movimentação rápida dos olhos para a próxima posição visual (SILVA; AMORIM; SANTOS, 2019) (com duração de 30 a 80 ms) (FEIT et al., 2017).

No contexto da linguagem NCL e do Ginga-NCL, cuja principal plataforma é a TV digital, a interação ocular do tipo seleção por tempo de permanência — baseada no evento de fixação ocular por um tempo determinado — pode ser uma escolha adequada, dado que se assemelha ao evento *selection* já existente na linguagem NCL e já suportado

pelo Ginga-NCL. O evento *selection* representa a seleção feita pelo usuário geralmente por meio do *mouse*, teclado e controle remoto. Assim, a inclusão de uma interação ocular similar ao evento *selection*, mas que não depende desses dispositivos de entrada vai ao encontro da motivação de proporcionar uma nova forma de interação para usuários que não podem utilizar as modalidades tradicionais devido à deficiência motora. Outra vantagem da interação ocular do tipo seleção por tempo de permanência é que ajuda a evitar problemas inerentes do uso da tecnologia *eye tracking*, como o do Toque de Midas, Jacob (1990).

Segundo Duchowski (2002), Jacob (1990) foi um dos primeiros trabalhos a usar a tecnologia de *eye tracking* para interação e a identificar esse importante problema em sistemas interativos por meio do olhar: o Toque de Midas. Se os olhos são usados de maneira similar a um *mouse*, é difícil diferenciar a ativação intencional de um comando da não intencional, i.e., o usuário ativa comandos para onde quer que olhe (voluntária ou involuntariamente). Para evitar o problema do Toque de Midas, Jacob (1990) discutiu várias soluções possíveis, incluindo piscadas, finalmente promovendo o uso do tempo de permanência para atuar como um mecanismo de seleção (DUCHOWSKI, 2002). Nesta dissertação, a solução adotada para reduzir esse problema foi o uso do tempo de permanência, i.e., fixação ocular por um tempo determinado, visto que o uso de um dispositivo auxiliar (e.g., controle) para realizar seleções não poderia ser utilizado por pessoas com deficiência motora.

1.1 Definição do Problema

Essa dissertação define as seguintes questões de pesquisa a serem estudadas e solucionadas por este trabalho:

Problema 1. *Como fazer com que o ambiente de TV digital possa oferecer interação por meio do olhar?*

Problema 2. *Uma vez implementada a modalidade de interação ocular no ambiente de TV digital, qual a percepção dos usuários sobre este novo tipo de interação com a TV?*

1.2 Objetivos e Contribuições

Esta dissertação tem como objetivo geral:

- Incorporar uma modalidade de interação por meio do olhar, baseada em fixação ocular, ao ambiente de TV digital que utiliza o *middleware* Ginga-NCL.

Destacam-se as seguintes contribuições da dissertação:

- Proposta de evento para interação ocular *EyeGaze* no modelo Nested Context Model (NCM) e na linguagem Nested Context Language (NCL).
- Proposta de arquitetura que permita integração de novos dispositivos *eye trackers* ao Ginga-NCL.
- Implementação de prova de conceito da proposta, estendendo o *middleware* Ginga-NCL para suportar autoria e execução de documentos NCL que usam o evento de interação ocular *EyeGaze*.
- Avaliação da proposta para interação com aplicações de TV digital baseada no paradigma Goal Question Metric (GQM), analisando experiência dos usuários, aceitação da tecnologia e desempenho (sob ponto de vista dos usuários e do sistema).

1.3 Estrutura da Dissertação

O restante desta dissertação está estruturada da seguinte forma. O Capítulo 2 apresenta o modelo NCM, os principais elementos da linguagem NCL, o trabalho de extensão do Ginga-NCL e a linguagem NCL 4.0, além de um contexto geral sobre a tecnologia *eye tracking*. No Capítulo 3 são discutidas pesquisas relacionadas a este trabalho.

O Capítulo 4 apresenta a proposta de interação usando a fixação ocular no Ginga-NCL. O Capítulo 5 descreve os detalhes de implementação da proposta. O Capítulo 6 descreve a etapa de avaliação, explicando a metodologia utilizada, experimentos realizados e análise dos resultados obtidos.

O Capítulo 7 conclui o trabalho, apresentando também suas limitações e assinalando trabalhos futuros.

2 Fundamentação Teórica

Este capítulo contextualiza os principais conceitos utilizados nesta dissertação. A Seção 2.1 apresenta os conceitos mais relevantes do modelo *Nested Context Model (NCM)*. Na Seção 2.2 são descritas as principais estruturas da linguagem *Nested Context Language (NCL)*. A Seção 2.3 descreve conceitos da linguagem NCL 4.0[1] e a Seção 2.4 o trabalho de extensão do Ginga-NCL (BARRETO, 2021; BARRETO; ABREU; MONTEVECCHI et al., 2020), usados como base para esta dissertação. A Seção 2.5 descreve o funcionamento da tecnologia *eye tracking* e os diferentes tipos de dispositivos sensores oculares.

2.1 Modelo NCM

O *Nested Context Model (NCM)* (SOARES; RODRIGUES, 2005) é um modelo conceitual de dados para representar e manipular documentos hipermídia. Segundo os autores, dentre outras funcionalidades, o NCM permite a estruturação lógica do documento através do uso de composições aninhadas, oferece a possibilidade de definição de relacionamentos causais e de restrição entre os seus componentes, permite especificar o comportamento temporal e espacial do documento de forma flexível e oferece suporte à autoria cooperativa.

O NCM utiliza o conceito de nós e elos, em que um nó representa uma informação de mídia. Um elo define relacionamentos entre nós, que por sua vez, podem ser estendidos em dois tipos principais: nó de composição e nó de conteúdo. Um nó de composição representa um conjunto de nós e elos, criando uma estrutura em grafo aninhado. Descritores permitem especificar como e onde os nós serão exibidos e os conectores permitem a criação de regras para que os elos iniciem ações sobre os nós que ele relaciona.

Nós de conteúdo são estendidos para representar diversos tipos de mídia, por exemplo, texto, imagem, áudio, vídeo ou itens como o *ApplicationNode* que pode ser utilizado para disparar a execução de um *script*, e o *SettingsNode*, que simboliza todas as variáveis

[1] https://forumsbtvd.org.br/tv3_0/

controladas pelo formatador, responsável pela interpretação e execução do documento. Já o aninhamento é realizado pelos nós de composição ou nós de contexto, que representam um conjunto de nós e elos.

O NCM permite a especificação espaço-temporal por meio de relacionamentos entre nós e elos, uma vez que é possível determinar uma duração para apresentação de um nó, quando será exibida a mídia que ele representa e suas repetições, e a posição dentro de uma região da tela do dispositivo de exibição. Por ser baseado em eventos, esse modelo considera ocorrências de eventos, por exemplo, o início da exibição de um nó, uma interação com o usuário ou a atribuição de uma variável para definição dos elos (relacionamentos entre nós). O NCM define um grupo de tipos básicos de eventos, que pode ser estendido se necessário (MUCHALUAT-SAADE, 2003). A Tabela 1 mostra tipos de eventos básicos do NCM e suas descrições (SOARES; RODRIGUES, 2005; SOARES; BARBOSA, 2011; BARRETO, 2021).

Tabela 1: Tipos de eventos existentes do modelo NCM.

Evento	**Descrição**
presentation	É o evento de apresentação ou exibição. Representa a exibição de uma âncora de conteúdo (segmento de mídia).
composition	É o evento de composição. Representa a apresentação da estrutura de um nó de composição (organização da composição).
selection	É o evento de seleção. Representa a seleção de uma âncora de conteúdo de um nó feita pelo usuário (por meio de um dispositivo de entrada, e.g., *mouse*, teclado e controle remoto).
attribution	É o evento de atribuição. Refere-se a mudança de valor de uma âncora de atributo de um nó.
hovering	É o evento de superposição do dispositivo apontador. Representa a superposição do *mouse* sobre uma âncora de conteúdo de um nó.
drag	É o evento de arraste. Representa o arraste sobre uma âncora de conteúdo de um nó.
focus	É o evento de foco. Representa o ganho de foco de uma âncora de conteúdo de um nó.
preparation	É o evento de preparação. Representa a preparação da exibição de um objeto de mídia, realizando uma busca antecipada do conteúdo a ser exibido até que todo ele seja carregado ou até que o*buffer* do dispositivo receptor esteja cheio. Além disso, esse evento inclui a instanciação do *player* que vai exibir o objeto de mídia como parte de sua ocorrência. (JOSUÉ; MUCHALUAT-SAADE; MORENO, 2018; IVANOV PEREIRA JOSUÉ, 2021).

Cada evento define uma máquina de estados (SOARES; RODRIGUES, 2005; SO-

ARES; BARBOSA, 2011) gerenciada pelo formatador, conforme Rodrigues (2003). A Figura 1 mostra a máquina de estados do modelo NCM. Um evento NCM pode estar em um dos seguintes estados: *sleeping*, *occurring* ou *paused*. Usando um evento de apresentação (i.e., *presentation*) como exemplo, seu estado inicial é igual a *sleeping*. Se é iniciada uma ação de *start*, indicando que a exibição de suas unidades de informação foi iniciada, o evento tem seu estado alterado para *occurring*. Se essa exibição for temporariamente suspensa (ação *pause*), o estado é alterado para *paused*, permanecendo até que uma ação *resume* ocorra, voltando então para o estado *occurring*, ou a apresentação seja interrompida (ação de *abort*). O evento de apresentação passará do estado de *occurring* para *sleeping* com o término natural da exibição das unidades de informação (*natural end)* ou como consequência de uma ação que force seu término (ação de *stop* ou *abort*). A duração do evento é dada pelo tempo em que ele permanece no estado de *occurring*.

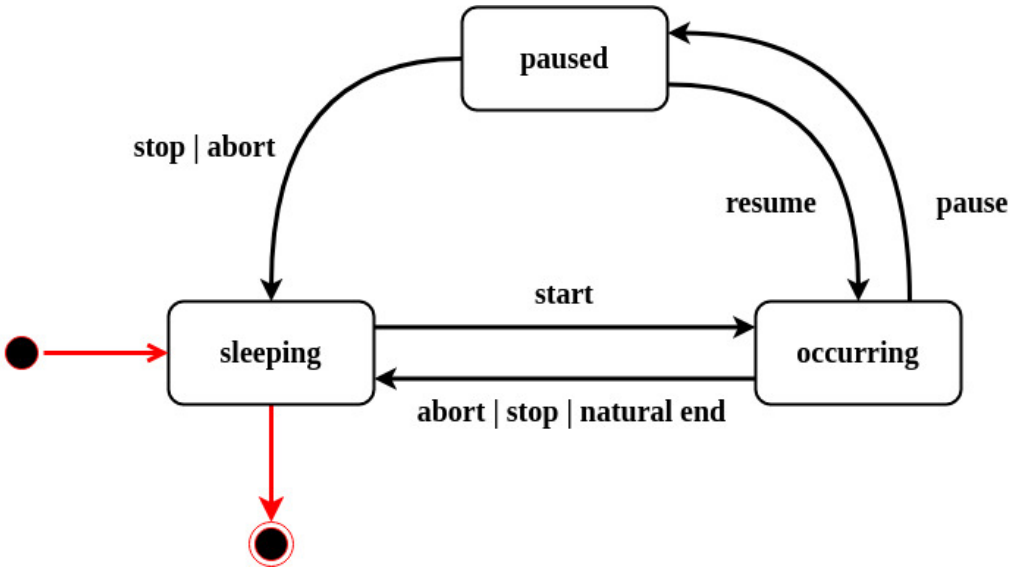


Figura 1: Máquina de estados do modelo NCM.

2.2 Linguagem NCL

Nested Context Language (NCL) (ITU-T H.761) é uma linguagem declarativa, baseada no modelo conceitual NCM. Como citado anteriormente, a linguagem NCL é padrão para desenvolvimento de serviços interativos no Sistema Brasileiro de Televisão Digital (SBTVD), utilizada para especificação de conteúdo interativo para TV digital e para serviços *IPTV* conforme o padrão ITU-T H.761 (ITU-T H.761).

Na linguagem NCL, os conectores e links definem a sincronização e interação entre os objetos que compõem o documento NCL (SOARES; BARBOSA, 2011). Consequen-

temente, ambos também definem a interação entre a aplicação e seu usuário. Conectores (elemento *<causalConnector>*) definem relações de causalidade, i.e., condições sobre eventos que devem ser satisfeitas para que ações sejam realizadas. Essas condições são especificadas nos conectores por meio de papéis de condição, descritos na Tabela 2, elaborada conforme ITU-Recommendation (ITU-T H.761), Soares e Barbosa (2011) e Josué, Muchaluat-Saade e Moreno (2018). Os elos (elemento *<link>*), por sua vez, usam os conectores para relacionar objetos em um documento NCL.

Tabela 2: Papéis de condição da linguagem NCL.

Papel	**Descrição (quando o elo será ativado)**
onBegin	Quando a apresentação for iniciada.
onEnd	Quando a apresentação for finalizada (naturalmente ou por uma ação *stop*).
onAbort	Quando a apresentação for abortada.
onPause	Quando a apresentação for pausada.
onResume	Quando a apresentação for retomada após uma pausa.
onSelection ou *onBeginSelection*	Quando uma tecla (a ser especificada) for pressionada enquanto o objeto ligado a esse papel estiver sendo apresentado; ou quando a tecla OK for pressionada enquanto o objeto ligado a esse papel estiver com o foco; ou quando um dispositivo apontador (.e.g, mouse) selecionar o objeto em apresentação ligado a esse papel.
onEndSelection	Quando uma tecla (a ser especificada) terminar de ser pressionada enquanto o objeto ligado a esse papel estiver sendo apresentado; ou quando a tecla OK terminar de ser pressionada enquanto o objeto ligado a esse papel estiver com o foco; ou quando um dispositivo apontador (.e.g, mouse) terminar a seleção do objeto em apresentação ligado a esse papel.
onBeginAttribution	Logo antes que um valor (a ser especificado) seja atribuído a propriedades ligadas a esse papel.
onEndAttribution	Logo após um valor (a ser especificado) ter sido atribuído a propriedades ligadas a esse papel.
onAbortAttribution	Quando a atribuição for abortada.
onPauseAttribution	Quando a atribuição for pausada.
onResumeAttribution	Quando a atribuição for retomada após uma pausa.
onBeginPreparation	Quando a preparação for iniciada.
onEndPreparation	Quando a preparação for finalizada.
onAbortPreparation	Quando a preparação for abortada.
onPausePreparation	Quando a preparação for pausada.
onResumePreparation	Quando a preparação for retomada.

Um objeto de mídia de um documento NCL deve ser associado a um descritor para definição de como será sua apresentação. Além disso, o descritor associa um objeto a uma região, especificando assim onde ele será exibido. As regiões em NCL são compostas por atributos espaciais (e.g., *top*, *left*, *width* e *height*) que representam o local em que a mídia será apresentada no dispositivo de exibição. A definição dos atributos espaciais de uma região pode ser feita tanto de forma absoluta (em *pixels*), como de forma relativa, considerando uma hierarquia de regiões. Por exemplo, ao atribuir o valor 50% para os atributos *width* e *height* de uma região que tem como pai a tela inteira do dispositivo, a mídia que utiliza essa região será apresentada com metade do tamanho da tela de exibição. A especificação de regiões utilizando valores relativos é uma abordagem interessante para permitir que a aplicação se adapte a diferentes dispositivos com tamanhos de telas variados.

Na fase de autoria da aplicação, o autor deve especificar regiões para mídias visuais que serão exibidas na tela. Neste processo, pode ocorrer que duas ou mais regiões se sobreponham total ou parcialmente (RANDELL; CUI; COHN, 1992). A fim de permitir a especificação de uma ordem de precedência, para o caso de regiões sobrepostas, a linguagem NCL define o atributo *zIndex* que deve ser especificado na região. O atributo *zIndex* deve receber um valor do tipo inteiro, e os objetos de mídia com valores maiores de *zIndex* são posicionados a frente das mídias que possuem valores menores de *zIndex*.

A Listagem 2.1 mostra um exemplo de documento NCL, especificado de forma que, ao ser executada, a aplicação NCL apresenta um vídeo de ondas do mar durante 5 segundos. Quando o vídeo é iniciado, uma imagem de uma praia deve ser exibida à frente do vídeo, em tamanho menor, no canto superior direito. Quando o vídeo termina, a imagem deve ter sua apresentação finalizada e a aplicação será encerrada. Desta forma, o documento possui 2 nós de mídia, um do tipo vídeo (linha 27) e um do tipo imagem (linha 28). O elemento <*port*> (linha 26) indica por onde começar a exibição do documento, sendo neste caso, pela mídia de vídeo. Para descrever onde as mídias devem ser apresentadas, foram especificadas 2 regiões (linhas 4 a 9): a região de id "rgVideo" (linhas 5 e 6) determina que o objeto deve ocupar toda a tela e define o atributo *zIndex* igual a zero; a região de id "rgImagem" (linhas 7 e 8) estipula que o objeto de mídia deve ser exibido no canto superior direito da tela, em tamanho reduzido e com *zIndex* igual a 1 (i.e., à frente do objeto associado à região "rgVideo"). Para estipular como as mídias devem ser exibidas, existem 2 descritores (linhas 10 a 13) vinculados às regiões existentes no documento. O descritor de id "dsVideo" define uma duração de 10 segundos para a mídia associada por meio da propriedade *explicitDur*.

O relacionamento entre os nós de mídia é definido pelos 2 *links* descritos nas linhas 29 a 36. O primeiro *link* (linhas 29 a 32) faz com que, ao iniciar a apresentação da mídia de vídeo, a imagem também tenha sua exibição iniciada. Para isso, foi especificado o conector "onBegin_Start" (linhas 15 a 18), com o papel de condição *onBegin* e ação *start*, i.e., definindo a regra para esse *link* de que quando a mídia associada ao papel de condição (*role*) *onBegin* iniciar, a mídia vinculada ao papel *start* deve sofrer a ação de ser iniciada também. O segundo *link* (linhas 33 a 36) indica que ao terminar a apresentação da mídia de vídeo, a imagem também tenha sua exibição finalizada. Para isso, foi especificado o conector "onEnd_Stop" (linhas 19 a 22), com o papel de condição *onEnd* e ação *stop*, i.e., definindo a regra para esse *link* de que quando a mídia associada ao papel de condição (*role*) *onEnd* tiver sua exibição finalizada, a mídia vinculada ao papel *stop* deve sofrer a ação de ser terminada.

```
1  <?xml version="1.0" encoding="ISO-8859-1"?>
2  <ncl id="appExe" xmlns="http://www.ncl.org.br/NCL3.0/EDTVProfile">
3      <head>
4          <regionBase>
5              <region id="rgVideo" left="0%" top="0%" width="100%"
6                  height="100%" zIndex="0"/>
7              <region id="rgImagem" left="80%" top="0%" width="20%"
8                  height="20%" zIndex="1"/>
9          </regionBase>
10         <descriptorBase>
11             <descriptor id="dsVideo" region="rgVideo" explicitDur="5s"/>
12             <descriptor id="dsImagem" region="rgImagem"/>
13         </descriptorBase>
14         <connectorBase>
15             <causalConnector id="onBegin_Start">
16                 <simpleCondition role="onBegin"/>
17                 <simpleAction role="start"/>
18             </causalConnector>
19             <causalConnector id="onEnd_Stop">
20                 <simpleCondition role="onEnd"/>
21                 <simpleAction role="stop"/>
22             </causalConnector>
23         </connectorBase>
24     </head>
25     <body>
```

```
26        <port id="pVideo" component="video"/>
27        <media id="video" descriptor="dsVideo" src="mar.mp4"/>
28        <media id="imagem" descriptor="dsImagem" src="praia.jpg"/>
29        <link xconnector="onBegin_Start">
30            <bind role="onBegin" component="video"/>
31            <bind role="start" component="imagem"/>
32        </link>
33        <link xconnector="onEnd_Stop">
34            <bind role="onEnd" component="video"/>
35            <bind role="stop" component="imagem"/>
36        </link>
37    </body>
38 </ncl>
```

Listagem 2.1: Exemplo de documento NCL.

2.3 Linguagem NCL 4.0

A linguagem NCL 4.0 é a proposta de extensão da linguagem NCL para permitir interação multimodal (BARRETO, Fábio; MONTEVECCHI et al., 2019b; BARRETO; ABREU; MONTEVECCHI et al., 2020; MONTEVECCHI et al., 2020; VALENTIM; BARRETO; MUCHALUAT-SAADE, 2020; BARRETO, 2021), identificação e interação de múltiplos usuários (BARRETO, Fábio; MONTEVECCHI et al., 2019a; BARRETO, Fabio; MONTEVECCHI et al., 2019; BARRETO; ABREU; MONTEVECCHI et al., 2020; BARRETO, 2021) e efeitos sensoriais (JOSUÉ; ABREU et al., 2018; BARRETO; ABREU; SANTOS et al., 2019; IVANOV PEREIRA JOSUÉ, 2021). Essa proposta de extensão denominada NCL 4.0 foi submetida ao projeto da TV 3.0 do Sistema Brasileiro de Televisão Digital (SBTVD)[2] como extensão do *middleware* Ginga para a futura geração do sistema de TV digital do Brasil. No momento, foi aprovada na segunda fase (testes e avaliações) e se encontra na terceira fase do projeto da TV 3.0.

Para possibilitar a multimodalidade em NCL, a versão 4.0 especifica novos eventos: *Touch*, *BodyMotionRecognition*, *Pointer*, *VoiceRecognition*, *FaceRecognition*, *GestureRecognition* e *HandPoseRecognition* (BARRETO, 2021). Esta dissertação colaborou com a especificação de NCL 4.0 propondo o tipo de evento *EyeGaze*, conforme será detalhado no Capítulo 4. Em NCL, os eventos de interação possuem um atributo *key* que especifica

[2] https://forumsbtvd.org.br/tv3_0/

o que foi reconhecido na interação realizada pelo usuário. Consequentemente, foram definidos novos papeis de condição e possíveis valores para o atributo *key*, conforme Barreto (2021), descritos na Tabela 3.

Tabela 3: Papéis de condição para contemplar interações multimodais em NCL 4.0.

Papel	**Descrição**	**Atributo key**
onVoiceRecognition	Quando algo dito pelo usuário for reconhecido enquanto o objeto associado a esse papel estiver sendo apresentado ou com foco.	Trecho de voz reconhecido.
onFaceRecognition	Quando alguma expressão facial do usuário for reconhecida enquanto o objeto associado a esse papel estiver sendo apresentado ou com foco.	Tipo da expressão facial do usuário.
onHandPoseRecognition	Quando algum sinal com a mão feito pelo usuário for reconhecido enquanto o objeto associado a esse papel estiver sendo apresentado ou com foco.	Tipo do sinal feito com a mão do usuário.
onGestureRecognition	Quando algum gesto feito pelo usuário for reconhecido enquanto o objeto associado a esse papel estiver sendo apresentado ou com foco.	Tipo do gesto do usuário.
onBodyMotionRecognition	Quando algum movimento corporal feito pelo usuário for reconhecido enquanto o objeto associado a esse papel estiver sendo apresentado ou com foco.	Tipo do movimento do corpo do usuário.
onTouch	Quando alguma interação por toque feita pelo usuário for reconhecida enquanto o objeto associado a esse papel estiver sendo apresentado ou com foco.	Tipo do toque.
onPointer	Quando um apontamento realizado pelo usuário for reconhecido enquanto o objeto associado a esse papel estiver sendo apresentado ou com foco.	Posição do apontamento.

A versão 4.0 da linguagem NCL também inclui um novo módulo, para possibilitar interações multiusuário, denominado *Users* (BARRETO, 2021). Esse módulo possui os elementos *<userAgent>* e *<userProfile>* para permitir que usuários e perfis sejam representados individualmente. O *userAgent* proporciona uma representação individualizada do usuário: além de um *id* para identificação do usuário no documento NCL, seu atributo *src* recebe a *Uniform Resource Identifier (URI)* de um arquivo XML com as características do usuário, que são carregadas para o nó *userSettings* (i.e., nó de características do

usuário) correspondente. Assim, é possível criar *links* associados a essas características, e especificar, por exemplo, que determinados eventos de interação só podem ocorrer caso tenham sido disparados pelo usuário especificado na cláusula *<userAgent>*. A Listagem 2.2 mostra como especificar o elemento *userAgent* em um documento NCL.

```
<userBase>
    <userAgent id="maria" src="maria.xml"/>
</userBase>
```

Listagem 2.2: Exemplo de definição do elemento *userAgent* em um documento NCL.

O *userProfile* representa a descrição de um perfil de usuário, e também permite atribuição de um arquivo XML com as propriedades de perfil (BARRETO, 2021). O perfil serve como um filtro, indicando quais usuários encontrados no receptor de TV digital atendem ao perfil definido: somente esses usuários terão suas propriedades consideradas na aplicação. A Listagem 2.3 mostra como especificar o elemento *userProfile* em um documento NCL.

```
<head>
    <userBase>
        <userProfile id="perfil_A" src="perfil_A.xml" max="2"/>
    </userBase>
    ...
</head>
<body>
    ...
    <link xconnector="onVoiceRecognition_Start">
        <bind role="onVoiceRecognition" component="video"/>
            <bindParam name="key" value="iniciar"/>
            <bindParam name="user" value="perfil_A"/>
        <bind role="start" component="video"/>
    </link>
</body>
```

Listagem 2.3: Exemplo de definição do elemento *userProfile* em um documento NCL.

Por fim, é possível identificar a participação do usuário sem usar os elementos *userAgent* e *userProfile* usando-se o valor "*all*" no parâmetro *user* dos *links* especificados no documento. Neste caso, serão criados *links* dinâmicos para todos os usuários encontrados no receptor, sem a validação de perfil (BARRETO, 2021). A definição de perfis e usuá-

rios, assim como o uso do parâmetro *user* não é obrigatória: se não forem utilizados no documento NCL, qualquer usuário poderá interagir com a aplicação.

A nova versão 4.0 também possibilita a manipulação de efeitos sensoriais de forma similar à manipulação de outros nós em NCL, por meio do novo elemento *effect*. Desta forma, é possível a autoria de aplicações mulsemídia (GHINEA et al., 2014), i.e., aplicações que estimulam outros sentidos do usuário além da audição e visão, permitindo a sincronização de efeitos sensoriais ao conteúdo audiovisual.

2.4 Ginga-NCL Estendido

O Ginga-NCL é um subsistema lógico do *middleware* Ginga, padrão do SBTVD e utilizado para especificação de conteúdo interativo para TV digital que permite a execução de aplicações multimídia especificadas em linguagem NCL (ITU-T H.761). Segundo Carvalho, Oliveira e Pereira (2009), a finalidade de uma camada de *middleware* é oferecer um serviço padronizado para as aplicações, fazendo com que não sejam perceptíveis as peculiaridades e heterogeneidades das camadas inferiores. A Figura 2, presente na recomendação ITU-Recommendation (ITU-T H.761), apresenta a arquitetura do *middleware* Ginga. O *Ginga Common Core* centraliza os serviços utilizados pelas máquinas de execução e apresentação, abarcando os *players* de mídias que notificam o *NCL Player* da ocorrência de eventos, entregando aplicações para o Ginga-NCL, reunindo informações de metadados e disponibilizando-a por meio dos objetos de mídia NCL *settings*. O *Ginga-NCL Presentation Enviroment*, ambiente de apresentação NCL que integra os interpretadores Lua (IERUSALIMSCHY; FIGUEIREDO; FILHO, 1996) e NCL, tem como principal núcleo o *NCL Player* ou *NCL Formatter*, que recebe e controla as aplicações multimídia NCL. Ao receber uma aplicação, o *NCL Player* solicita aos *XML Parsers* e aos *Converters* que a traduzam para as estruturas de dados internas necessárias para controle da apresentação do documento NCL. O componente *Scheduler* é então responsável por orquestrar essa apresentação.

Para que o Ginga consiga dar suporte às aplicações especificadas em NCL 4.0, foi estendido, com seus subsistemas *Ginga-NCL Presentation Environment* e *Ginga Common Core* sendo modificados conforme apresentado na Figura 3, que mostra a arquitetura estendida retirada de Barreto (2021). Ao *NCL Formatter* foram inseridos por Ivanov Pereira Josué (2021) os novos componentes *Presentation Orchestrator*, *Preparation Orchestrator*, *Sensory Device Calibrator* e *Sensory Effect Renderer Manager*. Já o *Interaction Manager*

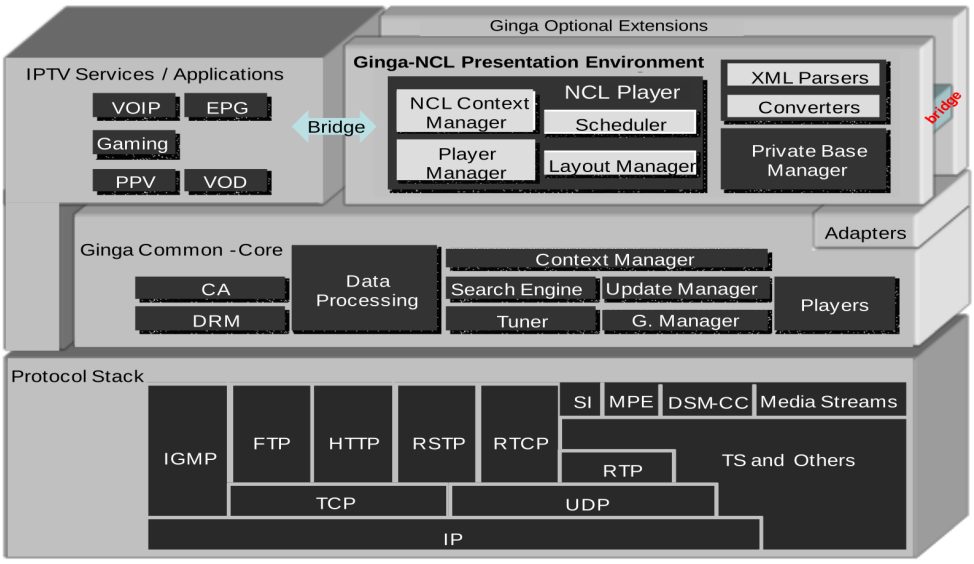


Figura 2: Arquitetura do *middleware* Ginga (ITU-T H.761).

foi inserido por Barreto (2021) ao *NCL Formatter* para dar suporte a diversas modalidades de interação por ele propostas. Ainda, no *NCL Formatter* foram modificados o *Player Manager* e *NCL Context Manager*. Ao *NCL Context Manager* foram inseridos como especializações os componentes *User Context Manager* e *Ambient Context Manager* para dar suporte ao gerenciamento de contexto de usuários e ambiente, respectivamente, propostos por Barreto (2021). O *Ginga Common Core* teve os módulos *Sensory Effect Renderer* (IVANOV PEREIRA JOSUÉ, 2021) e *Interaction Modules* (BARRETO, 2021) adicionados, além da alteração do módulo *Players*.

2.5 Eye tracking

Dispositivos rastreadores oculares, i.e., *eye trackers*, permitem capturar para onde o usuário está olhando (i.e., *gaze point* ou *point of gaze*) em tempo real, detectando atenção, foco, presença do usuário e em alguns casos, medições de tamanho das pupilas (i.e., pupilometria) (TOBII, 2020). A tecnologia de *eye tracking* também proporciona diferentes tipos de interação com os olhos a pessoas que não conseguem (devido à alguma deficiência) ou não querem interagir por meio de outros dispositivos tradicionalmente utilizados, como mouse, teclado ou controle remoto. Pode permitir o controle de uma aplicação mulsemídia seguindo a movimentação dos olhos do usuário (SILVA; AMORIM; SANTOS, 2019) ou possibilitar a criação de mapas que mostram em que os usuários mantêm sua atenção, de que forma e com o que se distraem. Combinado com outros dispositivos de entrada, pode criar novas experiências de interação em jogos. Além disso, *eye trackers* são capazes de

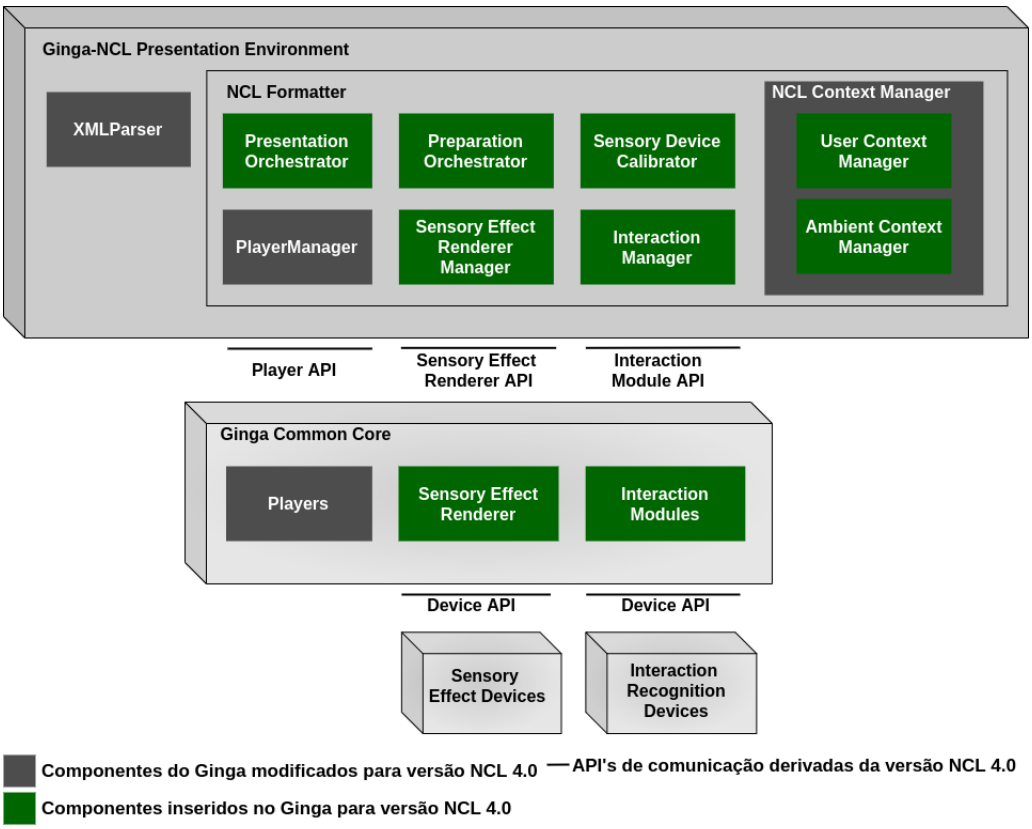


Figura 3: Arquitetura estendida do Ginga para suporte à NCL 4.0 (BARRETO, 2021)

auxiliar em pesquisas que utilizam a pupilometria para determinar situações de estresse, por exemplo. Os *eye trackers* também são dispositivos usados em *Augmentative and Alternative Communication (AAC)*, i.e., equipamentos que pessoas utilizam sem usar fala verbal, ajudando a melhorar a qualidade de vida de pessoas que têm *Esclerose Lateral Amiotrófica (ELA)*, permitindo que o movimento dos olhos seja usado para ativar uma letra, palavra ou frase (ASSOCIATION, 2022).

Dispositivos rastreadores oculares podem ser do tipo *head-mounted* (i.e., usados na cabeça), remotos (i.e., acoplados a monitores), *head-stabilized* (i.e., acoplado a uma mesa, onde o usuário fica com a cabeça imobilizada) ou estarem embarcados/integrados em outros dispositivos (MENTO, 2020). A Figura 4 mostra um exemplo de *eye tracker* do tipo *head-mounted*, o Tobii Pro Glasses 3[3]. Esse tipo de dispositivo é vestido pelo participante, possuindo a vantagem de alta mobilidade. Pode ajudar em experimentos no mundo real, como aplicações de pesquisa em esportes, communicação social, testes de dispositivos móveis e prateleiras de lojas, dentre outros (MENTO, 2020).

A Figura 5 mostra um exemplo de *eye tracker* do tipo *head-stabilized*, o EyeLink 1000

[3]https://www.tobiipro.com/product-listing/tobii-pro-glasses-3/

Figura 4: Exemplo de *eye tracker* do tipo *head-mounted*.

Plus[4]. Esse tipo se sensor restringe o movimento do usuário, sendo usados em pesquisas de alta fidelidade e em experimentos onde a precisão e exatidão são mais importantes que o conforto do usuário, já que não há liberdade de movimentação e a experiência visual possui um alto nível de controle (MENTO, 2020).

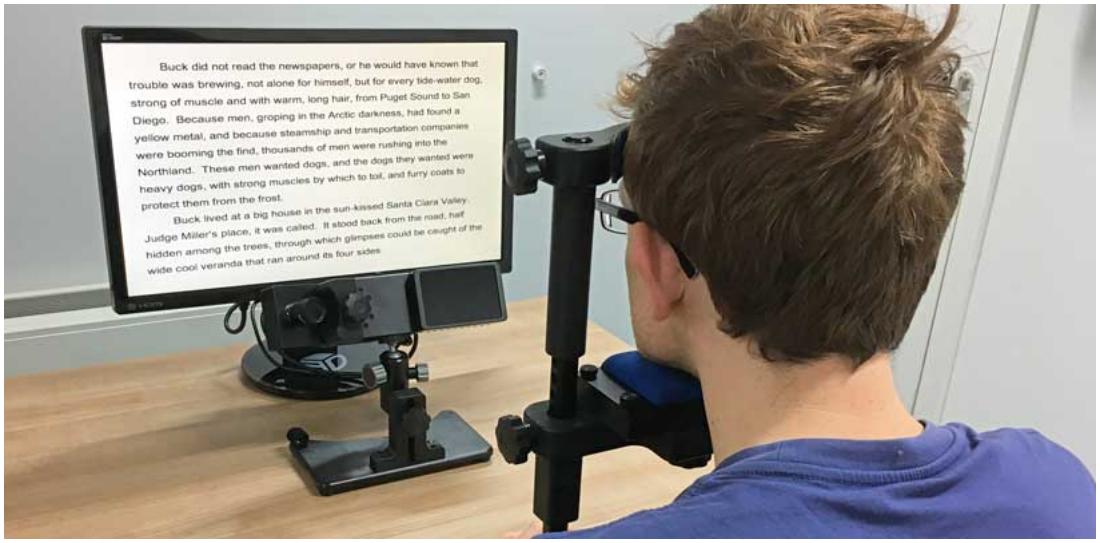


Figura 5: Exemplo de *eye tracker* do tipo *head-stabilized*.

A Figura 6[5] mostra um exemplo de *eye tracker* que está embarcado ou integrado a um óculos de realidade virtual, onde o usuário pode interagir com o conteúdo apresentado por meio de movimentos oculares (i.e., substituindo os controles). Esses sensores podem estar integrados a outras tecnologias também, como dispositivos de realidade aumentada, dispositivos de mira em sistemas de cirurgia ocular, painéis de veículos, dentre outros (MENTO, 2020).

A Figura 7 apresenta um exemplo de sensor rastreador remoto, o Tobii Pro Nano[6]

[4]https://www.sr-research.com/eyelink-1000-plus/
[5]https://www.bitbrain.com/blog/eye-tracking-devices
[6]https://www.tobiipro.com/product-listing/nano/

Figura 6: Exemplo de *eye tracker* que está embarcado/integrado em outro dispositivo.

acoplado a um *laptop*. Esse tipo de sensor geralmente fica acoplado a uma tela é não invasivo, não necessitando de contato com o usuário. Normalmente são usados para interações baseadas no que o usuário está olhando em relação à tela à qual estão acoplados.

Figura 7: Exemplo de *eye tracker* do tipo remoto.

A Figura 8 ilustra o esquema de funcionamento do *Tobii Eye Tracker 4C*[7], um sensor de rastreamento ocular remoto baseado em vídeo. Projetores de luz infravermelha formam um padrão dos olhos (i.e., reflexão córnea) e câmeras de alta resolução capturam imagens desse padrão e de movimentos do centro da pupila. Algoritmos matemáticos de processamento de imagem e de aprendizado de máquina são então utilizados nessas imagens para determinar a posição dos olhos e o *gaze point* (TOBII, 2020). O *Tobii Eye*

Tracker 4C é o modelo de sensor utilizado nesta dissertação para prova de conceito da proposta de inclusão de novo evento de interação pelo olhar no Ginga-NCL, o *EyeGaze*.

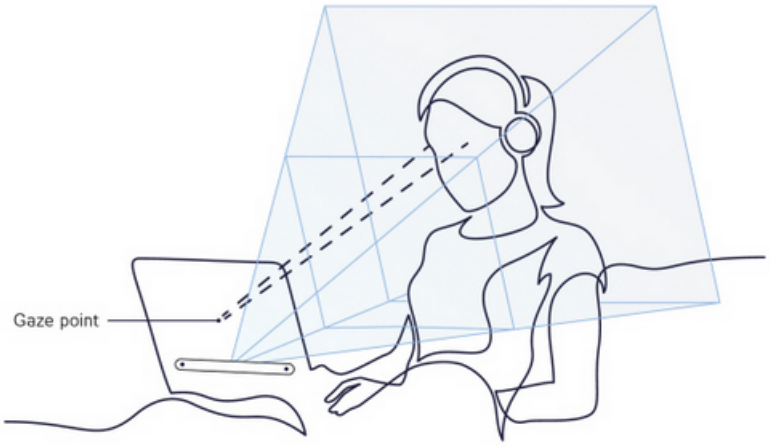


Figura 8: Ilustração do esquema de funcionamento do *eye tracker* (TOBII, 2020).

[7]https://help.tobii.com/hc/en-us/articles/213414285-Specifications-for-the-Tobii-Eye-Tracker-4C

3 Trabalhos Relacionados

Este capítulo discute na Seção 3.1 trabalhos relacionados ao uso da tecnologia *eye tracking* para realizar interações oculares em aplicações de leitura de textos (SILVA; AMORIM; SANTOS, 2019), sistemas para operação de TVs inteligentes (HENNESSEY; FISET, 2012; CHEN; LIN, 2018), em contexto de *displays* em exibição pública com interação espontânea(ZHANG et al., 2015), métodos para interação em ambientes 3D com múltiplas telas e múltiplos usuários (MARDANBEGI; HANSEN, 2011), um navegador *web* com interfaces ativáveis pelo olhar (CASARINI; PORTA; DONDI, 2020) e ferramentas para *Augmentative and Alternative Communication (AAC)* como forma de auxílio a pessoas com deficiência motora ou doenças neurológicas degenerativas (BISWAS; LANGDON, 2013; PEDROSA; PIMENTEL; TRUONG, 2015; RAMOS; SALES; TEIXEIRA, 2016; KURAUCHI et al., 2016). Na Seção 3.2 são apresentadas APIs (*Application Programming Interface*) para a integração e suporte de *eye tracking* na criação de aplicações e sistemas.

3.1 Eye Tracking para Interação Ocular

Em Silva, Amorim e Santos (2019), o *eye tracking* é utilizado para capturar a posição de leitura do usuário durante a apresentação de um texto, e disparar eventos na aplicação multimídia com base neste posicionamento. Para isso, os autores consideram dois tipos de movimentos oculares: fixação e sacada. O movimento de fixação está relacionado ao momento em que os olhos estão relativamente fixos em algum ponto, já o de sacada considera a movimentação dos olhos para a próxima posição visual. O evento de leitura proposto por Silva, Amorim e Santos (2019) possibilita inferir informações como o instante em que determinado ponto do texto foi alcançado, bem como quando a leitura do texto terminou.

Para que o *eye tracking* se comunique com o formatador multimídia, responsável pelo controle da apresentação da aplicação, Silva, Amorim e Santos (2019) propõem um componente Leitor, que é composto por um *Parser*, o *eye tracker* e um Escalonador.

Durante a execução da aplicação, o *eye tracker* fornece de forma contínua a posição do olhar do usuário para o Leitor. Já o Escalonador verifica se aquele ponto de visualização possui alguma âncora de texto associada ou não. A âncora representa um trecho do texto, que pode ser uma palavra ou um subconjunto de palavras que compõem o texto. Caso o ponto de visualização associado a uma âncora seja visualizada, o Escalonador dispara um evento de notificação de que aquela âncora foi iniciada. Desse modo, a proposta de Silva, Amorim e Santos (2019) permite que o autor multimídia dispare conteúdos de áudio ou efeitos sensoriais sincronizados com a leitura de trechos de um texto.

Diferente da proposta de Silva, Amorim e Santos (2019), que permite o disparo de eventos em resposta à visualização apenas de elementos textuais, a proposta desta dissertação possibilita associar o evento de visualização com qualquer tipo de mídia em um documento NCL, como vídeo e imagem. Dessa forma, o evento proposto neste trabalho pode ser utilizado em diferentes tipos de aplicações multimídia, não estando restrito a aplicações de leitura somente.

No contexto das TVs inteligentes, Hennessey e Fiset (2012) desenvolveram um sistema de rastreamento ocular remoto para ser utilizado no ambiente da sala de estar. Visando permitir livre movimentação por parte do espectador e conseguir captar as imagens necessárias para o rastreamento independentemente da posição do usuário, os autores usam um rastreador ocular (localizado atrás de uma placa transparente IR — Radiação Infravermelha) montado em um servomecanismo de panorâmica e inclinação que controlam a orientação do *eye tracker*. Os ângulos de panorâmica e inclinação desejados são determinados com base na localização dos rostos captados nas imagens fornecidas pelo Microsoft R Kinect. O rastreador ocular utilizado estende o trabalho anterior de Hennessey, Noureddin e Lawrence (2006), com melhorias de alcance, aumento, adição de uma lente de *zoom* automático e um aumento na potência das luzes infravermelhas. A lente oferece distâncias focais variáveis, permitindo a operação em distâncias de até 4 m do sistema ao visualizador. Embora o Kinect também use luz infravermelha, segundo os autores, não houve interferência entre os esquemas de iluminação e ambos os sistemas puderam operar ao mesmo tempo. Para a realização de experimentos e demonstração do sistema, foi desenvolvida uma aplicação onde o usuário controla a televisão usando fixação ocular pelo tempo de 1,5 segundo. O sistema de interação de TV é mostrado em operação na Figura 9. Quando o usuário deseja realizar ativar a interface *gaze-enabled*, olha acima da tela da TV por 1,5 segundo e então são exibidos os controles de seleção de canal e controle de volume sobrepostos ao conteúdo da tela. Essa interface sobreposta desaparece após o usuário voltar a observar o conteúdo na região central da tela. Os autores justificam que a

seleção realizada por fixação ocular poderia ter o auxílio de um controle remoto, mas que durante a avaliação do sistema, mesmo que o controle remoto de um botão resultasse em uma ativação mais rápida, os usuários preferiram universalmente a experiência totalmente *hands-free* ao controlar a televisão. De acordo com Hennessey e Fiset (2012), o desempenho do sistema alcançado foi mais do que suficiente para operar uma nova interface de *Smart TV* no ambiente da sala de estar. Todos os indivíduos testados foram capazes de operar a interface de TV inteligente e indicaram que a interface era fácil e intuitiva de usar.

Figura 9: Interface de controle para TV e sistema de *eye tracking* desenvolvidos por Hennessey e Fiset (2012) em funcionamento.

Ainda no contexto de operação de TVs inteligentes, os autores Chen e Lin (2018) propõem um sistema controlado pelo olhar para operar *Smart TVs*. Os autores analisaram as tarefas necessárias para operação da TV (e.g., mudar canais e alterar volume) e desenvolveram dois protótipos com interfaces voltadas para controle do aparelho de TV por meio interação visual. A Figura 10 mostra os protótipos A e B criados, que contam uma interface de menu (*a* e *b*), seleção de canais (*c* e *d*) e ajuste de volume (*e* e *f*). Uma *Smart*

TV da marca Panasonic foi usada nos testes em conjunto com um sensor da Tobii. Três *experts* em *design* de interfaces operaram os dois protótipos e por meio de uma avaliação heurística de interfaces de usuário (NIELSEN; MOLICH, 1990) e entrevistas, avaliaram a solução proposta, oferecendo sugestões de correção para ambas as interfaces. Em relação a qual *design* era mais funcional, para a interface de menu, o protótipo B foi considerado melhor; para a mudança de canal, o A foi considerado mais adequado; para o ajuste de volume, foi sugerida a integração de A e B. Os autores fazem considerações importantes relacionadas ao *design* de aplicações para interação visual e explicitam as modificações sugeridas pelos avaliadores. Outra questão que precisa ser melhor trabalhada na solução é a resolução do problema do Toque de Midas (JACOB, 1990), para a qual os autores sugeriram a adição de um controle remoto auxiliar.

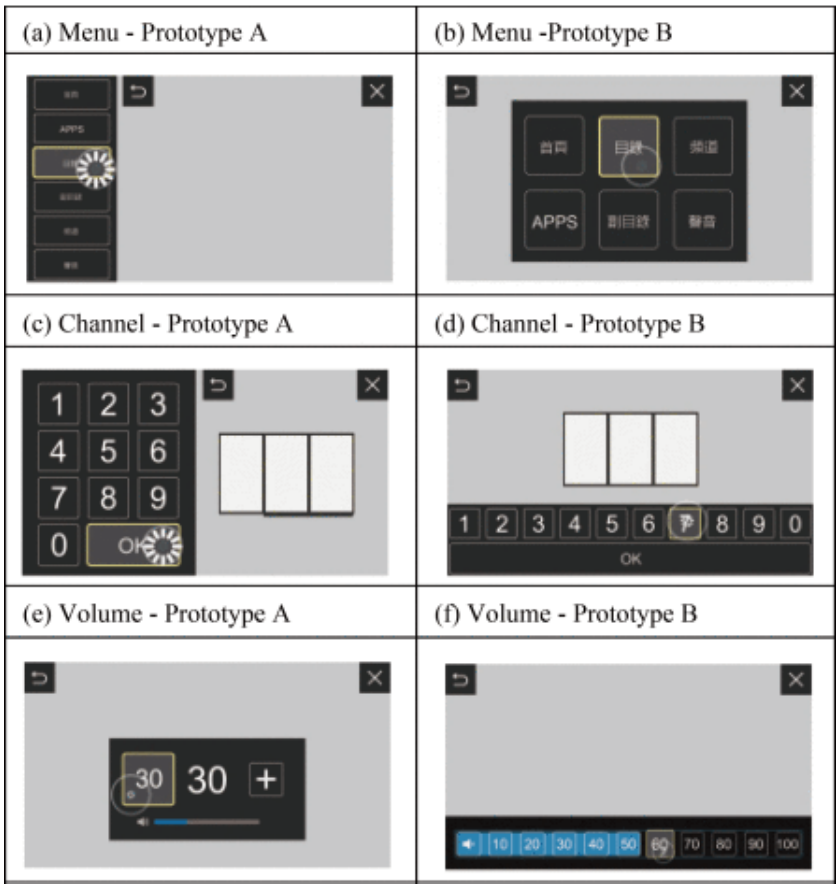


Figura 10: Protótipos criados por Chen e Lin (2018) para controlar *Smart TV* com os olhos.

Em contextos de *displays* em exibição pública, Zhang et al. (2015) apresenta o *GazeHorizon*, um sistema que demonstra a interação espontânea do olhar (usuários não estão previamente cientes do tipo de interação oferecida), permitindo que os usuários caminhem até uma tela e naveguem pelo conteúdo usando apenas os olhos, sem configuração prévia, calibração ou treinamento. O sistema rastreador *eye tracking* foi desenvolvido pelos au-

tores usando uma câmera, acoplada em cima de uma tela e uso de visão computacional para rastreamento, conforme exibido na Figura 11. A cada 30 frames, o sistema verifica a presença de usuários que estão olhando para a tela: se um grupo é detectado, somente a pessoa que está de frente para a parte mais central da tela conseguirá interagir. O sistema permite o comando de *scrolling* acompanhando a direção horizontal do olhar, i.e., quando o usuário olha para a esquerda, a tela é rolada para a direita (e vice-versa), e a velocidade de rolagem varia de acordo com a distância do centro que o usuário olha. À medida que o olhar do usuário segue naturalmente o objeto de interesse, a velocidade de rolagem diminuirá e fará com que o objeto pare no centro da tela. Se o sistema detectar que o rosto do usuário desapareceu de seu campo de visão, o sistema será redefinido para a fase inicial de detecção e rastreamento. O *GazeHorizon* foi avaliado em uma série de estudos de campo em um ambiente público durante o qual mais de uma centena de transeuntes interagiram com ele, sem solicitação e sem assistência. Os autores relatam que como os movimentos dos olhos são sutis, os usuários não podem aprender a interação do olhar apenas observando os outros e, como resultado, é necessária orientação.

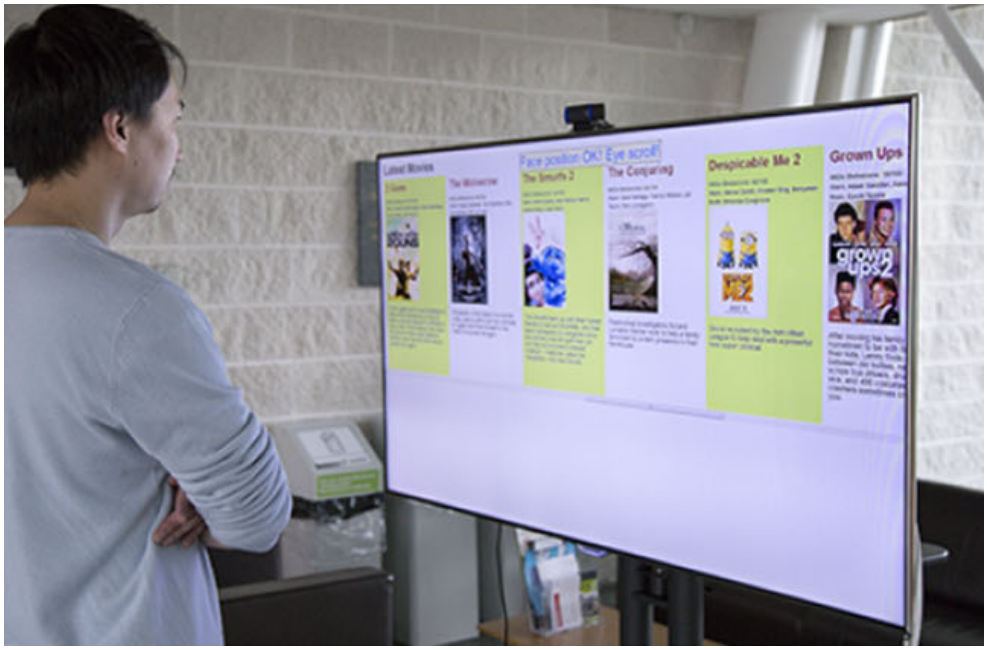


Figura 11: Sistema desenvolvido por Zhang et al. (2015) em funcionamento

Mardanbegi e Hansen (2011) apresentam um método para permitir que múltiplos usuários possam interagir com múltiplas telas planas, de maneira não simultânea na mesma tela, em um ambiente 3D usando um rastreador ocular do tipo *head-mounted*. O *eye tracker* foi desenvolvido pelos autores e usa duas câmeras: uma para captura do movimento dos olhos e outra para captar as imagens do ambiente (ou cena). Além disso, o *eye tracker* transmite os dados de imagem para um servidor *wireless* para processamento.

Dessa forma, os usuários possuem mais liberdade de movimentação e podem interagir com mais de uma tela — algo que não é possível com os *eye trackers* remotos, que estão limitados à interação com uma tela a que estão acoplados. A Figura 12 mostra um cenário 3D de uso, onde um usuário está usando o *head-mounted eye tracker* e pode interagir com três telas: celular, *laptop* e da TV. Para a identificação da tela com a qual o usuário está interagindo, foi usado um marcador visual e informações de contorno quadrilateral. Para a realização de experimentos, foi criada uma aplicação onde é possível interagir com três telas (2 TVs e 1 computador) e controlar objetos tais como ventilador e luzes. A ativação dos comandos é feita piscando duas vezes ao olhar para a região ativável pelo olhar. Os autores relatam que a acurácia foi suficiente para interação com aparelhos móveis e telas maiores, como a de TVs e que, portanto, esse tipo de sensor pode ser usado em espaços 3D para interação com telas. Uma limitação relatada é a de que para o método de mapeamento desenvolvido, a tela precisa estar inteiramente na imagem da cena (e.g., a tela está próxima demais).

Figura 12: Sistema desenvolvido por Mardanbegi e Hansen (2011) em exemplo de cenário de uso.

Em Casarini, Porta e Dondi (2020) é apresentado um navegador *web* com interfaces ativáveis pelo olhar. Essas interfaces consistem em cinco métodos para seleção de *links* (Menu, *Discrete Cursor*, *Progressive Zoom*, *Quick Zoom* e *Free Pointing* e um teclado integrado à tela, como mostra a Figura 13. Além disso, também foram desenvolvidas duas técnicas de rolagem de página e a possibilidade de salvar e recuperar favoritos. O navegador também possui os principais botões de navegação, como "voltar", "avançar",

"atualizar" e "Home", além de uma *pop-up* para minimizar ou fechar a janela. Todas es funções estão distribuídas em formato de botões e são ativáveis usando fixação ocular pelo tempo de 1,7 segundo. Como forma de *feedback* visual, os botões mudam de cor quando os usuários iniciam a fixação. Para implementação da solução foram usados a *open source engine* Gecko[1], o *Software Development Kit (SDK)* Tobii Core foi usado para a interação realizada por meio do *eye tracker* EyeX da Tobii.

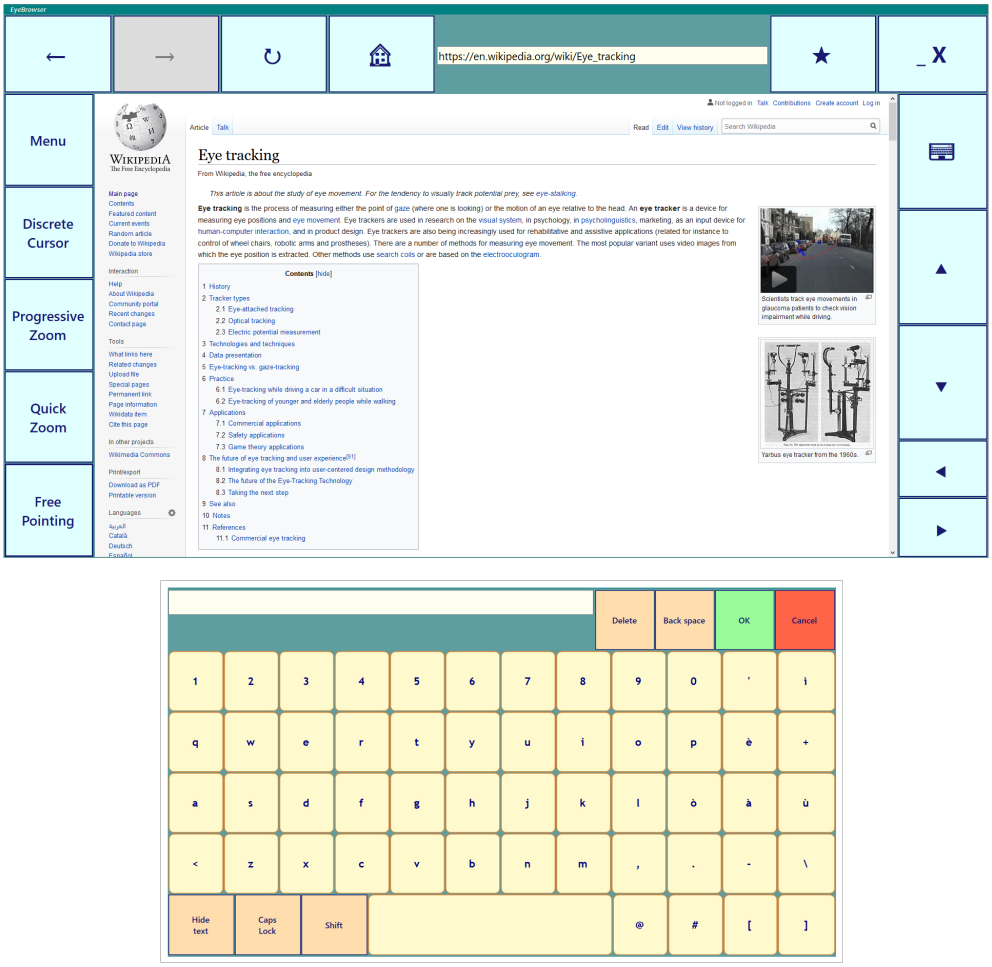


Figura 13: Interface gráfica do navegador *web* e do teclado desenvolvidos por Casarini, Porta e Dondi (2020).

Em aplicações multimídia, o usuário também pode interagir com o conteúdo através da entrada de dados textuais. Na proposta de Pedrosa, Pimentel e Truong (2015), é apresentado um método para permitir que usuário digite um texto utilizando a interação por movimento dos olhos. Para isso, Pedrosa, Pimentel e Truong (2015) propõem um abordagem baseada em filtragem de termos chaves, denominada *Filteryedping*, para oferecer suporte à digitação ocular. Dessa forma, ao invés do usuário ter que selecionar letra por letra com os olhos, o *Filteryedping* pesquisa em uma lista por possíveis palavras

[1]https://developer.mozilla.org/en-US/docs/Glossary/Gecko

que podem ser formadas considerando as letras que o usuário visualizou. O sistema então classifica as palavras candidatas com base em sua duração e frequência, e as apresenta ao usuário para confirmação.

Ramos, Sales e Teixeira (2016) apresentam uma ferramenta denominada LETRAS para auxiliar na comunicação cotidiana de pessoas que vivem com Esclerose Lateral Amiotrófica (ELA). A Figura 14 mostra a interface gráfica dessa ferramenta, onde o usuário deve realizar movimentos de olhar que desenham segmentos de linhas, triângulos e retângulos para formar palavras. O início da formação da palavra se dá ao dirigir o olhar a partir da região central da tela, para regiões laterais e/ou um dos cantos em sequência, e voltar para a região central. Dessa forma, o processo de comunicação não depende de uma temporização e sim de o paciente finalizar a movimentação do olhar correspondente ao caractere desejado. Os caracteres mais comuns foram associados aos movimentos mais curtos e rápidos. Também foi integrada à LETRAS uma ferramenta para predição de palavras, para fornecer funções de autocompletar e sugestões de palavras que são retroalimentadas pelo uso da ferramenta LETRAS. A ferramenta desenvolvida não usa um sensor *eye tracker* de alta precisão, mas usa a tecnologia de *eye tracking* por meio da combinação de câmera e *software*, indicando como trabalho futuro a utilização da própria câmera do *laptop* usado para testes, reduzindo assim ainda mais o custo da ferramenta.

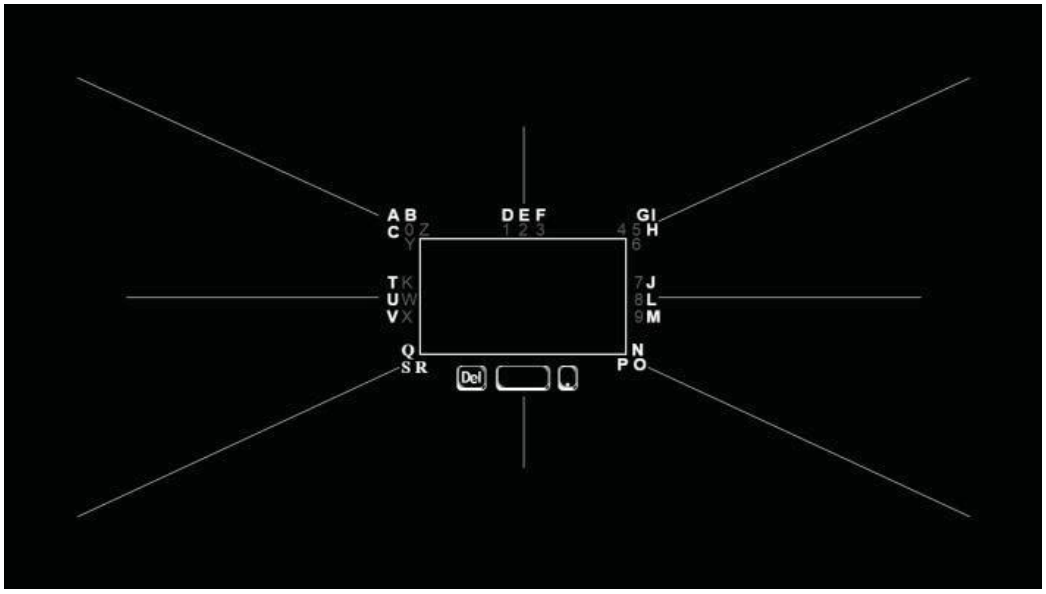


Figura 14: Interface gráfica da ferramenta LETRAS (RAMOS; SALES; TEIXEIRA, 2016).

Kurauchi et al. (2016) propõem um método de entrada de texto usando a interação baseada no olhar, denominada *EyeSwipe*, visando auxiliar na comunicação de pessoas com deficiências motoras. O sensor utilizado é o *eye tracker* remoto Tobii EyeX. Para digitar

as palavras, é usado um método chamado de seleção de cruzamento reverso: o usuário olha inicialmente para a região alvo, e um botão de ação é exibido acima dessa região após 100 milissegundos, o usuário deve olhar para esse botão de ação e então novamente para a região alvo para realizar a seleção. A Figura 15 exibe a interface gráfica do *EyeSwipe*, onde a linha em azul mostra o *gaze path* realizado para escrever a palavra "*This*". O usuário digita uma palavra selecionando inicialmente seu primeiro caractere "T" usando o cruzamento reverso e olhando pela vizinhança dos caracteres do meio em sequência. À medida que o usuário seleciona o último caractere da palavra, também com cruzamento reverso, o *EyeSwipe* calcula uma lista de palavras candidatas. O primeiro candidato é exibido no botão de ação quando o usuário olha para a última tecla, como forma de *feedback* visual, e é digitado assim que o usuário indica que o caminho do olhar foi concluído. Esse *feedback* visual serve para que o usuário possa digitar sem ter que conferir o que já foi digitado. Os espaços são adicionados automaticamente. Essa proposta também permite a adição de palavras que não estão no dicionário (e.g., nomes de pessoas). Segundo os autores, o *EyeSwipe* proporcionou taxas de digitação estatisticamente significativamente mais altas e interação mais confortável em experimentos com dez participantes que atingiram 11,7 palavras por minuto após 30 minutos de digitação com o *EyeSwipe*.

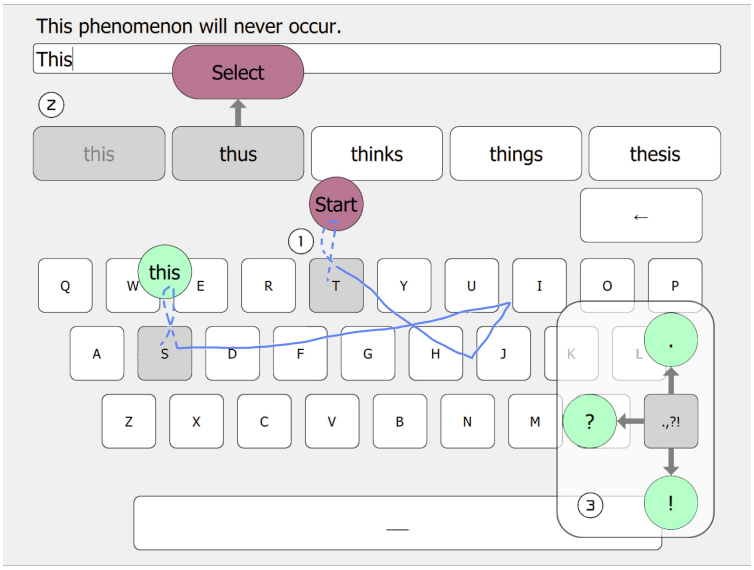


Figura 15: Interface gráfica da proposta de Kurauchi et al. (2016).

O uso da interação por meio dos olhos visando auxiliar pessoas com deficiências motoras também é abordado no trabalho de Biswas e Langdon (2013). No sistema proposto por Biswas e Langdon (2013), a interação por movimento dos olhos é utilizada para movimentação do cursor na tela, emulando a interação fornecida por um *mouse*. Conforme apresentado em Biswas e Langdon (2013), esse tipo de interação possui alguns desafios,

como a acurácia e a detecção de um clique, que geralmente é realizado por meio do piscar dos olhos. Com a realização de testes, Biswas e Langdon (2013) observaram que os usuários levaram em média 10 minutos para aprender a utilizar a nova forma de interação para movimento do cursor.

A principal contribuição desta dissertação em comparação aos trabalhos discutidos nessa seção é o foco em prover autoria de aplicações usando uma linguagem declarativa, permitindo a associação do evento de interação ocular a diferentes tipos de objetos de mídia. O trabalho de Hennessey e Fiset (2012) se concentra no desenvolvimento de um sistema de rastreamento ocular apropriado para o ambiente de sala de TV, mas o autor de aplicações multimídia ainda teria que realizar a integração do sistema à aplicação e desenvolvimento da interação ocular. O trabalho de Silva, Amorim e Santos (2019) permite o disparo de eventos em resposta à visualização apenas de elementos textuais, mas não permite associar o evento de visualização com os outros tipos de objetos de mídia em um documento NCL. Chen e Lin (2018) se preocupam em criar uma interface para controle de TVs inteligentes, mas não em prover interação com o conteúdo multimídia exibido nessas TVs. Já Mardanbegi e Hansen (2011) concentram-se no método para interação em ambientes 3D com múltiplas telas e múltiplos usuários, mas não no processo de autoria: a interface que permite a interação ocular com as telas foi feita para testes. Zhang et al. (2015), (CASARINI; PORTA; DONDI, 2020) e os trabalhos de ferramentas para *Augmentative and Alternative Communication (AAC)* (BISWAS; LANGDON, 2013; PEDROSA; PIMENTEL; TRUONG, 2015; RAMOS; SALES; TEIXEIRA, 2016; KURAUCHI et al., 2016) usam *eye tracking* para prover interações oculares nas próprias aplicações específicas, não para prover também autoria. Dessa forma, por propor uma extensão ao modelo NCM, à linguagem NCL e ao *middleware* Ginga-NCL para que os usuários possam interagir com os olhos usando o evento de fixação ocular *EyeGaze*, esta dissertação permite ao autor NCL a autoria de aplicações multimídia, especificando condições de disparo em conectores e elos em aplicações NCL. Assim, o autor não precisa se preocupar com a integração e implementação da interação ocular, além de ter o processo de autoria facilitado. Outra contribuição é de que o evento proposto independe de um sensor ou sistema rastreador específico para seu funcionamento. Finalmente, há o próprio processo de avaliação realizado nesta dissertação como contribuição. Na maior parte dos trabalhos aqui discutidos, as avaliações não foram estruturadas ou usaram paradigmas e métodos presentes na literatura, e muitas vezes não foi explicado no que se basearam para realização dos testes.

3.2 Eye Tracking APIs

Além dos trabalhos que usam *eye tracking* para prover interações oculares nas próprias aplicações desenvolvidas, por exemplo, livros multissensoriais, ferramentas para AAC e controle de TVs inteligentes, existem APIs que possibilitam a integração dessa tecnologia na autoria de aplicações e criação de sistemas, permitindo incorporar o *eye tracking* por meio de linguagens imperativas.

A *GazeCloudAPI*[2] é uma API em linguagem JavaScript que pode ser incorporada a páginas HTML5. Para a utilização em *website* há um período de teste gratuito e é necessário registro informando o domínio do *site* e assinatura da API. Para realizar o rastreamento ocular, é usada a própria *webcam* do usuário. A API permite capturar os *gaze points* em tempo real, possibilitando que sejam implementados diferentes tipos de interação ocular. Outra funcionalidade oferecida é a de calibragem dinâmica: há uma calibragem inicial onde o usuário fixa pontos que vão aparecendo na tela e depois que é terminada, é possível ajustá-la usando o *mouse*. Dessa forma, se há uma diferença entre o *gaze point* real e o indicado pela API, é possível clicar no ponto realmente fixado da tela e a API ajusta a calibragem. Há exemplos de utilização e uma demonstração[3] de utilização da ferramenta disponibilizados pelos criadores da *GazeCloudAPI*. Além dessa API, os autores disponibilizam diversas ferramentas que servem para gerar mapas de atenção do usuário como a *GazeRecorder*[4] e *GazeFlow*[5] ou usar a interação ocular "apontar" com a *GazePointer*[6]. Essa última, se usada em conjunto com a aplicação *GazeBoard*[7] pode ser usada como AAC, fornecendo um mecanismo para entrada de texto.

A biblioteca JavaScript *WebGazer* (PAPOUTSAKI; SANGKLOY et al., 2016) também permite integrar a tecnologia *eye tracking* a páginas HTML5 e usa a própria *webcam* do usuário. Diferentemente da *GazeCloudAPI*, que usa fixação ocular e movimentos com a cabeça para calibragem, a biblioteca *WebGazer* usa modelos de predição treinados a partir de cliques de *mouse* e movimentos do cursor realizados pelo usuário. Também há uma calibragem automática a partir de interações feitas usando o mouse por parte dos usuários. Existem diferentes formas de acessar os dados de *gaze points* permitindo implementação de diferentes tipos de interação oculares em aplicações. Os autores dispo-

[2] https://gazerecorder.com/gazecloudapi/
[3] https://api.gazerecorder.com/
[4] https://gazerecorder.com/gazerecorder/
[5] https://gazerecorder.com/gazeflow/
[6] https://gazerecorder.com/gazepointer/
[7] https://gazerecorder.com/gazeboard/

nibilizam a documentação das funções[8] e explicam as formas de acesso às informações dos *gaze points*. Também permite usos mais avançados da biblioteca, focados em treinamento dos modelos de predição para rastreamento ocular e coleta de dados, o que pode ser muito útil para pesquisas que envolvam *eye tracking*. Outra biblioteca também oferecida pelos autores é a *SearchGazer* (PAPOUTSAKI; LASKEY; HUANG, 2017), que tem como foco incorporar a *WebGazer* a *engines* de busca, como por exemplo *Bing* e *Google*, para permitir inferir áreas de interesse. A documentação e explicações das formas de uso também são detalhadas pelos autores[9].

Para sistemas e aplicações que usam um sensor *eye tracker* remoto da *Tobii*, a empresa disponibiliza um *Software Development Kit (SDK)* para integrar interação ocular a *games* e aplicações desenvolvidas nas *engines Unity* e *Unreal*[10]. Esses SDKs já vêm com exemplos funcionais para facilitar o entendimento por parte dos autores da aplicação. Além disso, a *Tobii* também oferece o SDK de baixo nível *Stream Engine SDK*[11], para usuários que precisam de maior controle dos recursos e tipos de interações oculares que serão desenvolvidas e integradas aos sistemas e aplicações. Esse SDK está disponível nas linguagens de programação C, C++ e C# e é compatível com o sistema operacional Windows. A biblioteca fornecida pelo *Stream Engine SDK* possui seis arquivos de cabeçalho e serve como intermediária entre a aplicação e o *eye tracker*, sendo os *headers tobii.h* e *tobii_streams.h* os mais importantes para implementação de interações oculares. Enquanto o *tobii.h* abarca as principais funções da API, tais como inicialização da API, conexão e enumeração dos sensores, o *tobii_streams.h* compreende as funções de controle dos *listeners* para leitura de *gaze points*.

As APIs apresentadas nesta seção buscam integrar *eye tracking* na autoria de aplicações usando linguagens imperativas e por meio dessas APIs é possível implementar diferentes tipos de interações oculares. Porém, para sua adoção, o autor ainda precisará realizar a integração dessas APIs e implementar o funcionamento da própria interação antes de incluí-la em sua aplicação. A contribuição desta dissertação em relação a esses trabalhos é tornar transparente para o autor as etapas de integração e desenvolvimento da interação usando o sensor *eye tracker*, oferecendo uma API de mais alto nível. Esta dissertação estende uma linguagem de autoria declarativa, a linguagem NCL, com um novo evento para representar a fixação ocular chamado *EyeGaze*. Dessa forma, o autor da aplicação que usa a interação ocular, só se preocupará em especificar os objetos de

[8] https://webgazer.cs.brown.edu
[9] https://webgazer.cs.brown.edu/search/
[10] https://gaming.tobii.com/developer/
[11] https://developer.tobii.com/product-integration/stream-engine/

mídia que são interativos através do evento ocular de fixação por período determinado, seguindo uma sintaxe XML já padronizada.

4 Proposta

Aplicações que suportam a modalidade de interação com os olhos podem usar a fixação do olhar (i.e., *eye gaze*) para indicar a seleção de um objeto. Uma fixação indica um período de tempo (de aproximadamente 100 ms até vários segundos) durante o qual a localização do *gaze point* permanece em uma região (FEIT et al., 2017). Regiões ativadas pelo olhar (i.e., *gaze-enabled regions*) (FEIT et al., 2017) são regiões espaciais que permitem que o usuário interaja ao fixar seu olhar nelas, ativando-as. Para que seja possível representar uma modalidade de interação que utiliza a fixação ocular para selecionar objetos de mídia, esta dissertação propõe a inclusão do evento *EyeGaze* no modelo NCM e, para implementação deste novo evento, a extensão da linguagem NCL, definindo o evento *EyeGaze* e consequentemente os novos nomes de papéis de condição para uso nos conectores NCL (elemento <*causalConnector*>).

A Tabela 4 mostra esses novos papéis de conectores e suas correspondentes transições de estado para o evento *EyeGaze*: *onBeginEyeGaze* terá sua condição satisfeita quando o início de uma fixação ocular for reconhecida enquanto um objeto ligado a esse papel estiver sendo apresentado, mudando o estado do evento de *sleeping* para *occurring*; *onEndEyeGaze* quando a fixação do objeto ligado ao conector for completada, sendo o estado do evento alterado de *occurring* para *sleeping*; *onAbortEyeGaze* terá sua condição cumprida quando a fixação ocular do objeto ligado ao papel do conector for interrompida e o estado do evento alterado de *occurring* para *sleeping*. Uma fixação ocular é considerada completa se o usuário olhar para determinada posição da tela por um tempo mínimo, que pode ser parametrizado por meio de um arquivo de configuração que faz parte da implementação da máquina de execução NCL (*middleware* Ginga-NCL).

A Listagem 4.1 apresenta um exemplo de como especificar a interação pelo olhar por meio do evento *EyeGaze*. O conector possui uma <*simpleCondition*> que usa o papel de condição *onBeginEyeGaze* (linha 9) e uma <*simpleAction*> que usa o papel de ação *start* (linha 10). Também é possível associar um usuário à condição no conector por meio do atributo *user* (linhas 8 e 9) e definição do usuário por meio do elemento <*userA-*

Tabela 4: Novos papéis de condição para conectores NCL.

Nome do Papel	Transição de Estado	Descrição
onBeginEyeGaze	*sleeping* para *occurring*	Quando o início de uma fixação ocular for reconhecida enquanto o objeto associado a esse papel estiver sendo apresentado.
onEndEyeGaze	*occurring* para *sleeping* (*natural end*)	Quando uma fixação ocular completa for reconhecida enquanto o objeto associado a esse papel estiver sendo apresentado.
onAbortEyeGaze	*occurring* para *sleeping* (*abort*)	Quando a interrupção de uma fixação ocular for reconhecida enquanto o objeto associado a esse papel estiver sendo apresentado.

gent> (linha 4) conforme proposto por NCL 4.0 (BARRETO, Fábio; MONTEVECCHI et al., 2019a; BARRETO; ABREU; MONTEVECCHI et al., 2020; BARRETO, 2021). Dessa forma, o elo especificado (linhas 16 a 21) usa o conector *onBeginEyeGazeStart* (linhas 7 a 11) para definir que quando a usuária "ana" iniciar uma fixação ocular da mídia "imageWinter" enquanto essa estiver sendo apresentada, será iniciada a apresentação da mídia "videoWinter". O atributo *user*, proposto e implementado por Fábio Barreto, Montevecchi et al. (2019a), Barreto, Abreu, Montevecchi et al. (2020) e Barreto (2021), é uma forma de possibilitar suporte a múltiplos usuários em NCL. É possível associar um usuário específico, como no exemplo da Listagem 4.1, ou pode ser utilizado o valor igual a "*all*", determinando que qualquer usuário pode ser o responsável pela interação e, consequentemente, ativação do evento *EyeGaze*.

```
1  <head>
2    ...
3    <userBase>
4        <userAgent id="ana" src="ana.xml"/>
5    </userBase>
6    <connectorBase>
7      <causalConnector id="onBeginEyeGazeStart">
8          <connectorParam name="user"/>
9          <simpleCondition role="onBeginEyeGaze" user="$user"/>
10         <simpleAction role="start"/>
11     </causalConnector>
12   </connectorBase>
13 </head>
```

```
<body>
  ...
  <link xconnector="onBeginEyeGazeStart">
    <bind role="onBeginEyeGaze" component="imageWinter">
        <bindParam name="user" value="ana"/>
    </bind>
    <bind role="start" component="videoWinter"/>
  </link>
</body>
```

Listagem 4.1: Exemplo de conectores e links que usam o evento *EyeGaze*.

Para que fosse possível o suporte a novas modalidades de interação utilizando outros tipos de dispositivos além do controle remoto e mouse, Barreto, Abreu, Montevecchi et al. (2020), Barreto (2021) e Montevecchi et al. (2020) estenderam e incluíram componentes no *middleware* Ginga. Para possibilitar a modalidade de interação pelo olhar utilizando um dispositivo de rastreamento ocular no Ginga-NCL, este trabalho propõe sua extensão a partir do trabalho de Barreto (2021) e Barreto, Abreu, Montevecchi et al. (2020). A Figura 16 mostra as modificações realizadas no *Ginga-NCL Presentation Enviroment*, que teve o componente *XML Parser*, o *NCL Formatter* e o *Interaction Manager* alterados para lidar com o evento *EyeGaze*. O novo componente *Gaze Recognition* foi adicionado ao *Ginga Common Core* para reconhecer e notificar a ocorrência dos eventos de interação que usam fixação ocular.

A Figura 17 mostra de forma mais detalhada, a arquitetura da solução de proposta de extensão para funcionamento do novo evento *EyeGaze* no *middleware* Ginga. O *Formatter* usa o *XML Parser* para derivar um *Document* do arquivo NCL recebido. Se durante o *parsing* do documento NCL, é identificado que existem objetos de mídia interativos por meio do evento *EyeGaze*, o módulo *Gaze Recognition Module* precisa ser iniciado. Assim, o *Interaction Manager*, criado por Barreto, Abreu, Montevecchi et al. (2020) e Barreto (2021), funciona como intermediário entre o *Formatter* e o *Gaze Recognition Module*, controlando as interações do usuário com a aplicação e consequentemente, sendo o responsável por gerenciar o *Gaze Recognition Module*. A *Interaction Module Common Interface* (BARRETO; ABREU; MONTEVECCHI et al., 2020; BARRETO, 2021) permite que sejam adicionados novos módulos de interação, usando métodos virtuais predefinidos, e serve como intermediária entre o *Gaze Recognition Module* e o *Interaction Manager*, possibilitando a inicialização do módulo e envio de informações. Ao iniciar o módulo de reconhecimento de interação ocular, o *Interaction Manager* deve informar ao

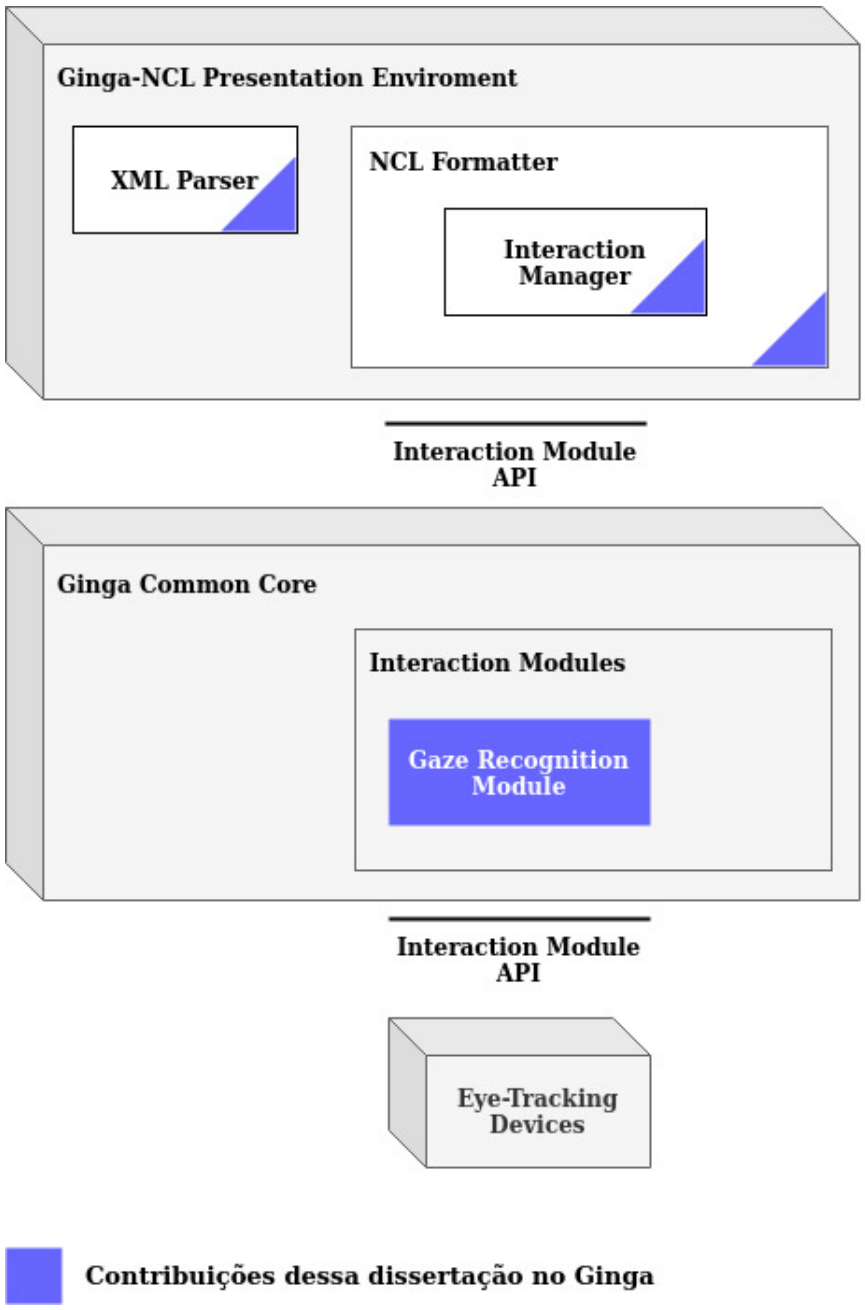


Figura 16: Arquitetura de módulos do Ginga com as modificações realizadas para dar suporte ao evento de interação ocular *EyeGaze*.

Gaze Recognition Module a lista de objetos de mídia e usuários associados que devem ser monitorados. Uma vez que o *Gaze Recognition Module* recebe essas informações, o módulo interno *Regions Converter* mapeia os atributos de região da lista de objetos (que seguem o padrão Ginga) para regiões com coordenadas normalizadas (no padrão das coordenadas dos *gaze points* recebidos do *eye tracker*). As regiões normalizadas são usadas pelo módulo interno *Gaze Point Verifier* que ao receber uma leitura de *gaze point*, verifica se esse está dentro das regiões ativáveis recebidas e se permanecem nessas regiões durante

o período de tempo parametrizado, o que caracteriza uma fixação ocular. Note que como os atributos da região são utilizados para informar as áreas que devem ser monitoradas pelo *Gaze Recognition Module*, apenas a posição inicial do objeto está sendo considerada pelo evento *EyeGaze*.

Caso uma ou mais regiões sejam ativadas ao mesmo tempo, pelo fato de serem sobrepostas, o *Gaze Recognition Module* notifica ao *Interaction Manager*, comunicando quais foram as mídias cujas regiões foram acionadas. Nesse caso, o *Formatter* irá executar a ação relacionada ao papel de condição satisfeito apenas para o elo relacionado à visualização da mídia que está sendo apresentada e que possui o maior valor de atributo *zIndex*. Para a leitura de *gaze points*, o sensor de *eye tracking* do usuário envia os dados de leitura para o *Gaze Recognition Module* por meio da *Application Programming Interface (API)* desse dispositivo.

O *Gaze Recognition Module* também usa dados de um arquivo de configuração de interações, mostrado na Figura 17 como *Interaction Config*. Esse arquivo permite ao autor parametrizar dados dos eventos de interações. No caso do *Gaze Recognition Module*, este arquivo de configuração contém os dados de parametrização do tempo de duração do evento *EyeGaze*.

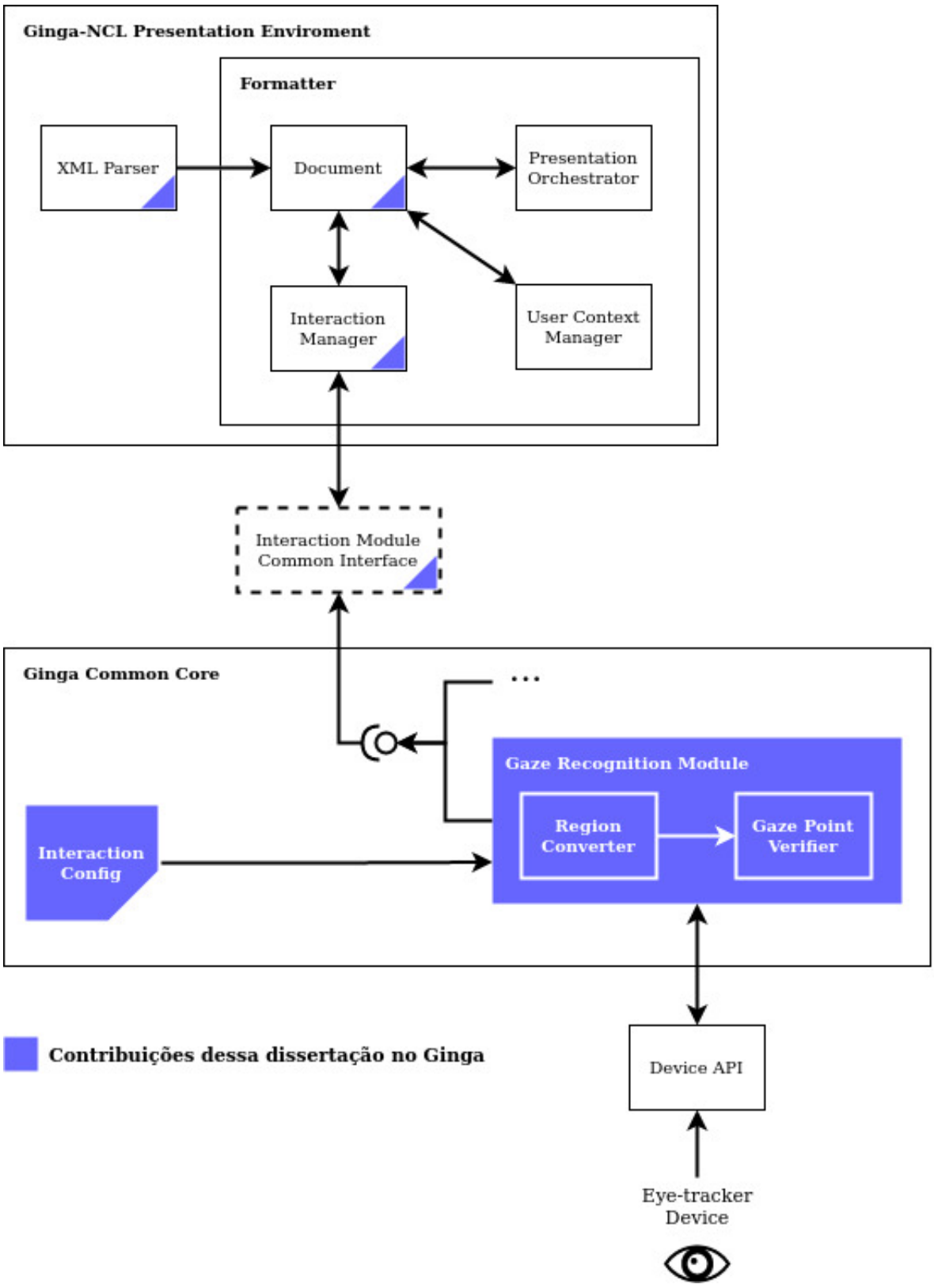


Figura 17: Arquitetura da proposta de extensão do Ginga-NCL para suporte ao evento de interação ocular *EyeGaze*.

5 Implementação

Como apresentado no Capítulo 4, para que fosse possível a modalidade de interação pelo olhar utilizando um dispositivo de rastreamento ocular no Ginga-NCL, o *middleware* foi estendido a partir da implementação[1] de Barreto, Abreu, Montevecchi et al. (2020) e Barreto (2021). Esta seção detalha essa extensão realizada pela implementação do módulo *Gaze Recognition Module*[2].

Como prova de conceito, o sensor utilizado neste trabalho é o *Tobii Eye Tracker 4C*[3], que pode ser usado em uma tela com tamanho máximo de 27 polegadas (proporção 16:9) ou 30 polegadas (proporção 21:9) e o usuário deve ficar a 50 até no máximo 95 centímetros de distância do sensor.

A biblioteca presente no *Stream Engine SDK*[4], disponiblizada pelo fabricante do sensor, a empresa *Tobii*, serve como intermediária entre o *eye tracker* e o *Gaze Recognition Module* desenvolvido. É importante lembrar que embora esta implementação da proposta apresentada no Capítulo 4 use o sensor *Tobii Eye Tracker 4C*, a arquitetura apresentada pode ser utilizada com outros modelos de *eye trackers*.

O *Formatter* é o responsável pelo ciclo de vida da apresentação e usa o *XML Parser* para derivar um *Document* do arquivo NCL recebido. Por isso, esses componentes precisaram ser alterados de forma que reconheçam o novo evento em NCL *EyeGaze*. O componente *XML Parser* foi modificado para reconhecer os papéis reservados para o evento *EyeGaze* (*onBeginEyeGaze*, *onEndEyeGaze*, e *onAbortEyeGaze*), para que durante o *parsing* do documento NCL, por meio dos elos especificados, sejam identificadas as mídias a serem "monitoradas" e quais são os usuários associados a elas. Desta forma, o *Document* foi adaptado para que comporte os dados das mídias associadas ao evento

[1] Implementação disponível em: bit.do/gingaMultimodalBranch

[2] Implementação disponível em: https://github.com/marinaivanov/ginga-mulsemedia/tree/gaze_module

[3] https://help.tobii.com/hc/en-us/articles/213414285-Specifications-for-the-Tobii-Eye-Tracker-4C

[4] O *Stream Engine SDK* compatível com o sistema operacional Linux foi instalado a partir do projeto disponível em: https://github.com/Eitol/tobii_eye_tracker_linux_installer

EyeGaze. Esses dados são utilizados em seguida pelo módulo *Interaction Manager*.

O *Interaction Manager* é o responsável pelo controle das interações do usuário com a aplicação multimídia, sendo o intermediário entre o *Formatter* e os módulos de interação. Além disso, faz com que esses módulos sejam ativados conforme especificado no documento NCL. Para adicionar novos módulos de interação, Barreto, Abreu, Montevecchi et al. (2020) e Barreto (2021) criaram uma interface chamada *Interaction Module Common Interface*, que deve ser implementada por cada módulo de interação.

A *Interaction Module Common Interface* é implementada por meio da classe *Interaction Module*, que possui os métodos virtuais pré-definidos utilizados para comunicação entre o *Interaction Manager* e os módulos de interação. Para que um módulo de interação seja adicionado, deve-se estender a classe *Interaction Module* e implementar os métodos virtuais *start()*, *setUserKeyList(JSON)* e *stop()*, que possibilitam respectivamente ao *Interaction Manager* iniciar, enviar informações e parar o novo módulo. Assim, o componente *Interaction Manager* foi alterado para gerenciar o *Gaze Recognition Module*, que por sua vez, tem implementados os métodos pré-definidos pela *Interaction Module Common Interface*.

Se durante o *parsing* do documento NCL é identificado que um ou mais objetos de mídia são interativos por meio dos papéis *onBeginEyeGaze*, *onEndEyeGaze* ou *onAbortEyeGaze*, o *Interaction Manager* ativa o módulo de reconhecimento do olhar (*Gaze Recognition Module*). Para realizar essa ativação, o *Interaction Manager* usa o método *start()*. Essa instância cria uma *thread* que executa a rotina de inicialização do dispositivo *eye tracker* e seus *listeners tobii_wait_for_callbacks()* (que coloca a em estado de espera até receber uma leitura de *gaze point*) e *tobii_device_process_callbacks()* (que recebe os *gaze points* e os envia para o método responsável pelo seu processamento).

O *Interaction Manager*, além de iniciar o *Gaze Recognition Module*, deve também informar quais são as mídias interativas associadas ao evento *EyeGaze*. Essa comunicação é feita usando o método virtual *setUserKeyList(JSON)*, que envia um objeto *JavaScript Object Notation (JSON)* (CROCKFORD, 2006) com uma lista de mídias e seus dados de regiões da tela (*keys*), resolução da tela (*screenWidth* e *screenHeight*) e um *user*. Essa estrutura é apresentada na Listagem 5.1. Cada item do *array key* é uma mídia com dados da sua região e o *user* é o usuário associado aos papéis do evento *EyeGaze* no *link* relacionado a essas mídias (BARRETO, Fábio; MONTEVECCHI et al., 2019a; BARRETO; ABREU; MONTEVECCHI et al., 2020; BARRETO, 2021). Para identificação da mídia, o atributo *id* contém o *id* do objeto de mídia. Se nenhum elo do documento

utiliza o evento *EyeGaze*, o *Gaze Recognition Module* não é iniciado. O atributo *user* foi incluído no objeto JSON como forma de possibilitar suporte a múltiplos usuários, mas não é obrigatório. Caso seja utilizado o valor igual a *all*, qualquer usuário à frente do sensor pode realizar a interação. Caso a aplicação NCL especifique mais de um usuário, sendo necessário diferenciá-los, cada um dos usuários deve possuir seu próprio sensor e associá-lo ao seu identificador de usuário, o que ainda é um trabalho de implementação futuro.

```
{ "user": "string",
  "screenWidth": "number",
  "screenHeight": "number",
  "key":
      [  {   "id":     "string",
             "left":   "number",
             "top":    "number",
             "width":  "number",
             "height": "number"
          },
          ...
      ]
}
```

Listagem 5.1: JSON para mapeamento de regiões.

A utilização e composição do objeto JSON foi pensada para evitar que alterações de dispositivos *eye trackers* prejudiquem a interoperabilidade da ferramenta. Ao receber o JSON, o *Gaze Recognition Module* usa o módulo interno *Region Converter* para realizar a conversão de mapeamento das coordenadas de região utilizadas pelo Ginga-NCL em coordenadas normalizadas (0 a 1): a borda da esquerda da tela é 0, a borda da direita é 1, a borda superior é 0 e borda inferior é 1. Conforme mostrado na Figura 18, para a conversão, são necessárias as informações de resolução da tela contidas no JSON (*screenWidth* e *screenHeight*) e atributos de região da mídia: *top*, *left*, *width* e *height*. A região convertida é determinada por dois pontos: *topLeft* e *bottomRight*.

As coordenadas x e y do ponto *topLeft* são obtidas por meio das Equações 5.1 e 5.2 respectivamente.

$$x = left/screenWidth \tag{5.1}$$

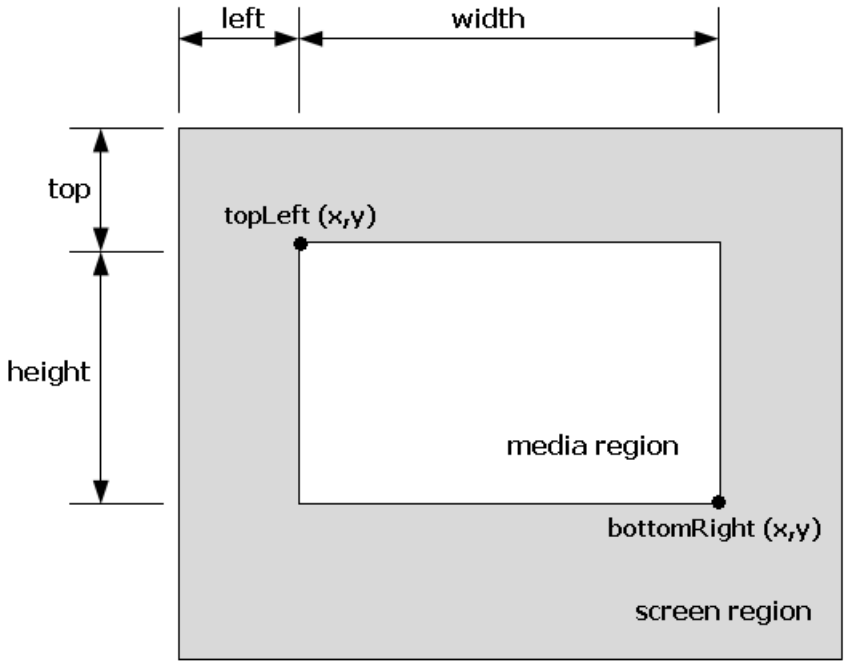


Figura 18: Representação da região de mídia em relação à tela inteira (adaptado de (SOARES; BARBOSA, 2011)).

$$y = top/screenHeight \tag{5.2}$$

As coordenadas x e y do ponto *bottomRight* são obtidas por meio das Equações 5.3 e 5.4 respectivamente.

$$x = (left + width)/screenWidth \tag{5.3}$$

$$y = (top + height)/screenHeight \tag{5.4}$$

Depois que as regiões são recebidas e normalizadas no *Region Converter*, são utilizadas pelo módulo interno *Gaze Point Verifier* cujo funcionamento é representado pelo Algoritmo 1. Cada região R possui, além dos pontos *bottomRight* e *topLeft* que a delimitam, os seguintes atributos: um *id* que armazena a identificação de sua mídia; uma flag *gazed*, que indica se a região foi olhada (assume valor *true*) ou não (assume valor *false*); a variável *startTime*, que armazena a hora em que uma região começou a ser fixada; o *regState* contém o estado da fixação da região, assumindo os valores *sleeping* (padrão), *initiated* e *finished*. Quando o *Gaze Point Verifier* recebe um ponto P do sensor, a variável *curTimeStamp* armazena o momento em que o ponto foi lido. A lista de regiões então é percorrida, e se o ponto P está em alguma região R, é necessário identificar se

a fixação acabou de começar ou já tinha iniciado anteriormente. Se a região R já tinha sido olhada, o evento *EyeGaze* ainda não foi iniciado (i.e., *regState* igual a *sleeping*) e a duração da fixação é suficiente para considerar que o evento *EyeGaze* deve ser iniciado na região R, *regState* é atualizada para o valor *initiated* e o formatador é notificado de que o evento deve sofrer a ação de *start*. Se região R já tinha sido olhada, o evento *EyeGaze* iniciado (i.e. *regState* igual a *initiated* e o tempo para completar a fixação foi alcançado, significa que o evento *EyeGaze* deve ser finalizado. Desta forma, *regState* é atualizada para o valor *finished* e o formatador é notificado de que o evento deve sofrer a ação de *stop*. Se a região R não tinha sido olhada (i.e., *gazed* igual a *false*), a fixação acabou de começar: a região é marcada como olhada (i.e. *gazed* recebe valor *true*) e *startTime* recebe o valor de *curTimeStamp*, indicando o momento em que o usuário começou a olhar para a região R em questão. Se o ponto P não está dentro da região R, é necessário verificar se o usuário a estava olhando anteriormente (i.e., variável *gazed* igual a *true*), indicando que interrompeu a fixação. Em caso afirmativo, a variável *gazed* recebe o valor *false*. Se o evento *EyeGaze* já havia sido iniciado (i.e. *regState* igual a *initiated*) precisa então ser abortado e o formatador é notificado de que o evento deve sofrer a ação de *abort*. A variável *regState* adquire então o valor *sleeping*.

Para que seja possível ao usuário definir os tempos de início e término do evento *EyeGaze*, foi criado um arquivo JSON do *middleware* para parametrização das interações, o "*interaction-config.json*". Nele, é possível informar valores de parametrização que definem o tempo de duração da fixação ocular, conforme exibido na Listagem 5.2. Para o evento *EyeGaze*, é permitido definir os valores de *duration* e *constant*, utilizadas no Algoritmo 1. A variável *duration* armazena o tempo total da duração, ou seja, por quanto tempo o usuário deve olhar para a região para que o evento *EyeGaze* esteja completo. É especificada em segundos e possui valor padrão igual a 1. A variável *constant* armazena a constante usada para definir após quanto tempo depois de começada a fixação pelo usuário, o evento *EyeGaze* deve ser iniciado pelo formatador (i.e. *gazeStartTime*), de acordo com a Equação 5.5. Essa constante deve estar entre 0 e 1, com valor padrão igual a 0.33.

```
1 { "eyeGaze":
2   { "duration": 1, "constant": 0.33 }, ...
3 }
```

Listagem 5.2: JSON com parametrização do tempo de duração do evento *EyeGaze*.

$$gazeStartTime = duration * constant \tag{5.5}$$

Algoritmo 1 Gaze Point Verifier.

Data:

P : gaze point.

$curTimeStamp$: tempo corrente do sistema.

$regionList$: lista de regiões.

$gazed$: indica se a região foi olhada.

$startTime$: o momento em que a região começou a ser fixada.

$regState$: indica o estado da fixação ocular na região. Pode assumir os valores $sleeping$, $initiated$, ou $finished$.

$duration$: tempo necessário para considerar que o evento de fixação ocular EyeGaze está completo.

$gazeStartTime$: tempo necessário para considerar que o evento de fixação ocular EyeGaze foi iniciado.

```
Ler um gaze point P
curTimeStamp ← tempo corrente do sistema
for cada R de regionList do
    if P ∈ R then
        if gazed then
            if (regState = sleeping) and ((curTimeStamp − startTime) ≥
             gazeStartTime) and (curTimeStamp − startTime) < duration then
                Notificar o formatador de que o evento EyeGaze deve ser iniciado
                regState ← initiated
            else
                if (regState = initiated) and (curTimeStamp − startTime) ≥ duration
                 then
                    Notificar o formatador de que o evento EyeGaze deve ser finalizado
                    regState ← finished
                end
            end
        else
            startTime ← curTimeStamp
            gazed ← true
        end
    else
        if gazed then
            if regState = initiated then
                Notificar o formatador de que o evento EyeGaze deve ser abortado
            end
            gazed ← false
            regState ← sleeping
        end
    end
end
```

É importante notar que vários objetos de mídia podem ocupar a mesma região de forma parcial ou total, conforme mostra a Figura 19: a imagem "deserto.png" está sobreposta à imagem "inverno.png". Essa sobreposição entre as regiões das mídias é definida por meio da propriedade *zIndex*. Quanto maior o *zIndex* de uma região, mas à frente ela está. Dessa forma, na Figura 19 a região da imagem "deserto.png" possui *zIndex* de maior valor que a região da imagem "inverno.png". Supondo que ambas as imagens exibidas estão associadas ao evento *EyeGaze* por meio do papel de condição *onBeginEyeGaze*, se o usuário iniciar uma fixação ocular na imagem "deserto.png", o *Gaze Recognition Module* notificará que é necessário iniciar o evento *EyeGaze* para ambas as mídias. Além disso, o *Gaze Recognition Module* não sabe quais mídias estão sendo apresentadas no momento (i.e., estado *occurring*). Então se a mídia "deserto.png" não estivesse sendo apresentada, mas fosse fixada de forma inconsciente pelo usuário, ainda seria notificada para ter o evento iniciado. Para solucionar esse problema, ao receber a lista de mídias que foram fixadas, o *Formatter* primeiro elimina da lista as mídias que não estão em estado *occurring*. Em seguida, verifica qual a mídia cuja região possui maior *zIndex*: essa é a mídia para qual o evento é acionado. Desta forma, a solução proposta se beneficia da capacidade do formatador Ginga-NCL de apenas ativar relações temporais de mídias que estão em execução.

Figura 19: Exemplo de sobreposição de regiões de mídias.

6 Avaliação

Este capítulo apresenta a análise do novo evento *EyeGaze* para interação com a TV digital. A Seção 6.1 discute os métodos utilizados para elaborar e estruturar a etapa de avaliação. Na Seção 6.2 são detalhados os experimentos realizados. Na Seção 6.3 os resultados obtidos são analisados e discutidos.

Para realização da avaliação do evento *EyeGaze*, foram efetuados experimentos com usuários e experimentos quantitativos, nos quais foram usados como base o paradigma *Goal Question Metric (GQM)* (BASILI, 1992), o *User Experience Questionnaire (UEQ)* (LAUGWITZ; HELD; SCHREPP, 2008; SCHREPP; HINDERKS; THOMASCHEWSKI, 2017) e o modelo *Technology Acceptance Model (TAM)* (DAVIS, 1989). Nas próximas seções, serão apresentados mais detalhes sobre a metodologia seguida, testes e os resultados obtidos com os experimentos efetuados.

6.1 Metodologia

O paradigma GQM é um mecanismo do tipo *top-down* para definição de objetivos mensuráveis (CALDIERA; ROMBACH, 1994). O GQM e sua aplicação no trabalho de Mattos (2021) foram utilizados como métodos para orientar e estruturar os experimentos realizados e resultados obtidos nesta pesquisa. Como apresentado em Caldiera e Rombach (1994), a aplicação da abordagem GQM resulta em um modelo hierárquico de três níveis: conceitual (*goals*), operacional (*questions*) e quantitativo (*metrics*). Para definir esse modelo, primeiro são estabelecidos os *goals* (i.e., objetivos). Cada *goal* definido é decomposto em *questions* (i.e., questões) que buscam respondê-los, tonando-os quantificáveis. Por sua vez, as *questions* são refinadas em *metrics* (i.e., métricas) objetivas e/ou subjetivas.

A Tabela 5 apresenta os *goals* definidos para avaliação do evento *EyeGaze* no contexto de interação com a TV digital. Para cada *goal* na coluna "Objetivo", a coluna "Descrição" contém o objeto de análise (e.g., evento *EyeGaze*), um propósito (e.g., avaliação), uma questão ou aspecto a ser mensurado (e.g., para o G1 é a experiência do usuário) e o

ponto de vista pelo qual a medição será considerada (e.g., para o G1 é o dos usuários). As próximas seções discutem as questões e métricas estabelecidas para cada objetivo definido. Para o G1, o questionário UEQ foi usado em conjunto com questões adicionais específicas. Para o G2, o modelo TAM foi empregado. Os objetivos G3 e G4 também possuem questões elaboradas especificamente para eles.

Tabela 5: *Goals* determinados para os experimentos realizados com o evento *EyeGaze*.

Objetivo	Descrição
G1	Analisar o evento *EyeGaze* para interação com a TV digital com o propósito de avaliação da experiência dos usuários.
G2	Analisar o evento *EyeGaze* para interação com a TV digital com o propósito de avaliação da aceitação dessa tecnologia sob ponto de vista dos usuários.
G3	Analisar o evento *EyeGaze* para interação com a TV digital com o propósito de avaliação do desempenho sob ponto de vista dos usuários.
G4	Analisar o evento *EyeGaze* para interação com a TV digital com o propósito de avaliação do desempenho sob ponto de vista do sistema.

6.1.1 G1 - Questões e Métricas

Para alcançar o objetivo G1, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação da experiência dos usuários*, foram usadas as questões descritas na Tabela 6 e as questões do UEQ (LAUGWITZ; HELD; SCHREPP, 2008). Na Tabela 6, as questões Q1 a Q5 usam a escala Likert (LIKERT, 1932), onde é possível escolher o nível de concordância ou discordância em relação a uma afirmação. Neste caso, o valor 1 é dado para indicar discordância total e 5 para concordância total.

Tabela 6: Questões do objetivo G1.

Questão	Descrição
Q1	Gostei de interagir usando meus olhos.
Q2	É confortável usar o Eye Gaze para interagir com a TV digital.
Q3	SENTI FADIGA nos ombros ao usar o Eye Gaze para interagir.
Q4	SENTI FADIGA no pescoço ao usar o Eye Gaze para interagir.
Q5	SENTI FADIGA nos olhos ao usar o Eye Gaze para interagir.

A Tabela 7 apresenta as métricas criadas para responder às questões Q1 a Q5 do *goal* G1. A métrica Atratividade (AT) é usada para responder à questão Q1 e o Conforto (CF) para responder às questões Q2 a Q5.

Tabela 7: Métricas para questões do G1.

Métrica	Descrição	Questões
AT	Atratividade (SCHREPP; HINDERKS; THOMASCHEWSKI, 2017), i.e., se o usuário gosta ou não, se a tecnologia em questão é atraente e agradável. Será medido usando resposta do usuário em escala Likert de 1 a 5.	Q1
CF	Conforto, i.e., o quanto o usuário sente que é confortável utilizar a tecnologia. Será medido usando resposta do usuário em escala Likert de 1 a 5.	Q2, Q3, Q4 e Q5

O UEQ também foi usado para alcançar o *goal* G1. O UEQ tem como objetivo ser uma avaliação direta e rápida da experiência do usuário (SCHREPP; HINDERKS; THOMASCHEWSKI, 2017). A Tabela 8 contém uma visão geral desse questionário, que consiste em 26 itens compostos por palavras antônimas entre si. Esses itens, contidos na coluna "Antônimos", podem ser avaliados com um número de 1 a 7 na escala Likert. Metade dos itens começa com o termo de sentido negativo e a outra metade começa com o termo de sentido positivo, listados no questionário de forma randomizada. As respostas representam valores de -3 (total concordância com o termo **negativo**) a 3 (total concordância com o termo **positivo**). Por exemplo: se um participante respondeu 7 para o item "Desagradável / Agradável", está expressando total concordância com o termo positivo "Agradável" (valor 3); se respondeu 7 para o termo "Bom / Ruim", está expressando total concordância com o termo negativo "Ruim" (valor -3).

Como discutido em Schrepp, Hinderks e Thomaschewski (2017), os 26 itens estão agrupados em 6 escalas, explicadas na coluna "Descrição" da Tabela 8: atratividade, transparência, eficiência, controle, estimulação e inovação. As escalas também podem ser divididas conforme os aspectos avaliados nas seguintes classificações: atratividade, pragmática (transparência, eficiência e controle) e qualidade hedônica (estimulação e inovação). O UEQ não produz uma nota geral como resultado, ao invés disso, as escalas e classificações são interpretadas como métricas e são, portanto, utilizadas para responder ao *goal* G1.

6.1.2 G2 - Questões e Métricas

Para alcançar o objetivo G2, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação da aceitação dessa tecnologia sob ponto de vista*

Tabela 8: Questionário UEQ.

		Descrição	Antônimos
Atratividade		Impressão geral do produto. Os usuários gostam ou não? É atraente, agradável ou atrativo?	Desagradável / Agradável, Bom / Ruim, Desinteressante / Atrativo, Incômodo / Cômodo, Atraente / Feio, Simpático / Antipático
Pragmática	**Transparência**	É fácil se familiarizar com o produto? É fácil de aprender? O produto é fácil de entender e claro?	Incompreensível/Compreensível, De fácil aprendizagem / de difícil aprendizagem, Complicado / Fácil, Evidente / Confuso
	Eficiência	Os usuários podem resolver suas tarefas sem esforço desnecessário? A interação é eficiente e rápida? O produto reage rapidamente à entrada do usuário?	Rápido / Lento, Ineficiente / Eficiente, Impraticável / Prático, Organizado / Desorganizado
	Controle	O usuário se sente no controle da interação? Ele ou ela pode prever o comportamento do sistema? O usuário se sente seguro ao trabalhar com o produto?	Imprevisível / Previsível, Obstrutivo / Condutor, Seguro / Inseguro, Atende as expectativas / Não atende as expectativas
Hedônica	**Estimulação**	É empolgante e motivador usar o produto? É divertido usar?	Valioso / Sem valor, Aborrecido / Excitante, Desinteressante / Interessante, Motivante / Desmotivante
	Inovação	O produto é inovador e criativo? Captura a atenção dos usuários?	Criativo / Sem criatividade, Original / Convencional, Comum / Vanguardista, Conservador / Inovador

dos usuários, foram elaboradas as questões descritas na Tabela 9 a partir do Technology Acceptance Model (TAM). O TAM é um modelo utilizado para entender se uma tecnologia será aceita e utilizada pelos usuários ou não. A utilização do sistema ou tecnologia é determinada pela intenção de uso, que é afetada pela utilidade percebida e pela facilidade de uso (DAVIS, 1989). As questões foram adaptadas dos questionários construídos e discutidos nos trabalhos de Davis (1989) e Hu et al. (1999).

Tabela 9: Questões do TAM para objetivo G2

Utilidade Percebida (PU)	
Questão	**Descrição**
PU1	Usar o Eye Gaze para interagir com aplicações de TV digital me permitiria interagir mais rapidamente.
PU2	Usar o Eye Gaze melhoraria meu desempenho para interagir com aplicações de TV digital.
PU3	Usar o Eye Gaze poderia tornar minha interação com aplicações de TV digital mais eficaz.
PU4	Usar o Eye Gaze facilitaria minha interação com aplicações de TV digital.
PU5	O Eye Gaze seria útil para eu interagir com aplicações de TV digital.
PU6	O Eye Gaze me ajudaria a interagir com aplicações de TV digital.
Percepção da facilidade de uso (PEOU)	
Questão	**Descrição**
PEOU1	Aprender a usar o Eye Gaze para interação com aplicações de TV digital seria fácil para mim.
PEOU2	Seria fácil usar o Eye Gaze para interagir com aplicações de TV digital.
PEOU3	Eu consideraria o Eye Gaze flexível para interagir com aplicações de TV digital.
PEOU4	Seria fácil para eu tornar-me hábil no uso do Eye Gaze para interagir com aplicações de TV digital.
PEOU5	Usar o Eye Gaze exigiria pouco esforço físico para interagir com a TV digital.
PEOU6	Usar o Eye Gaze exigiria pouco esforço mental para interagir com a TV digital.

Como modelo, o TAM não estabelece uma métrica padrão para a avaliação das questões elaboradas, mas, como as questões foram agrupadas em Utilidade Percebida (PU) e Percepção de Facilidade de Uso (PEOU), essa divisão foi considerada para a criação das métricas, descritas na Tabela 10. A métrica PU foi usada para as questões PU1 a PU6 e a métrica PEOU para as questões PEOU1 a PEOU6.

Tabela 10: Métricas para questões do G2.

Métrica	Descrição	Questões
PU	Utilidade Percebida (do inglês, *Perceived Usefulness*) (DAVIS, 1989), i.e., o quanto uma pessoa acredita que a tecnologia em questão melhorará seu desempenho em uma atividade. Será medida usando resposta do usuário em escala Likert de 1 a 5.	PU1, PU2, PU3, PU4, PU5 e PU6
PEOU	Percepção da Facilidade de Uso (do inglês, *Perceived Ease of Use*) (DAVIS, 1989), i.e., o quanto uma pessoa acredita que usar a tecnolgia em questão será livre de esforço. Será medida usando resposta do usuário em escala Likert de 1 a 5.	PEOU1, PEOU2, PEOU3, PEOU4, PEOU5 e PEOU6

6.1.3 G3 - Questões e Métricas

Para alcançar o objetivo G3, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação do desempenho sob ponto de vista dos usuários*, foram elaboradas as questões Q6 e Q7 descritas na Tabela 11. Ambas usam a escala Likert (LIKERT, 1932), em que é possível escolher o nível de concordância ou discordância em relação a uma afirmação. Nesse caso, o valor 1 é dado para indicar discordância total e 5 para concordância total.

Tabela 11: Questões do objetivo G3.

Questão	Descrição
Q6	O Eye Gaze tem boa precisão durante a interação com a TV digital.
Q7	O Eye Gaze tem boa velocidade durante a interação com a TV digital.

A Tabela 12 contém as métricas usadas para responder às questões Q6 e Q7. A métrica Precisão (PR) é usada para responder à questão Q6 e a métrica Velocidade (V) é usada para a questão Q7. Ambas são métricas subjetivas, baseadas na percepção do usuário sobre o quão assertivas são a velocidade e precisão do evento *EyeGaze* para interação com a TV Digital.

6.1.4 G4 - Questões e Métricas

Para alcançar o objetivo G4, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação do desempenho sob ponto de vista do sistema*, foram elaboradas as questões Q8 e Q9 descritas na Tabela 13. A mesma métrica Tempo (T) foi

Tabela 12: Métricas para questões do G3.

Métrica	Descrição	Questões
PR	Precisão (FEIT et al., 2017), i.e., se há uma variação aceitável da diferença entre a região fixada pelo usuário e a região que está sendo ativada na tela. Será medida usando resposta do usuário em escala Likert de 1 a 5.	Q6
V	Velocidade, i.e., se a interação está mais lenta ou mais rápida do que deveria. Será medida usando resposta do usuário em escala Likert de 1 a 5.	Q7

usada para responder às questões Q8 e Q9. Essa métrica é apresentada na Tabela 14.

Tabela 13: Questões do objetivo G4.

Questão	Descrição
Q8	Quanto tempo leva para a interação ser iniciada, uma vez que foi identificada?
Q9	Qual o tempo máximo de processamento de cada gaze point à medida que o número de mídias aumenta?

Tabela 14: Métricas para questões do G4.

Métrica	Descrição	Questões
T	Tempo, medido em milissegundos.	Q8 e Q9

6.1.5 Modelo GQM

A Figura 20 sumariza e ilustra a estrutura do modelo GQM construído nesta dissertação. A seguir, serão apresentados os experimentos realizados para a coleta de dados.

6.2 Experimentos

Nesta dissertação foram realizados experimentos práticos com usuários e experimentos quantitativos adicionais como forma de coleta de dados para as métricas estabelecidas e discutidas nas seções anteriores. Esta seção detalha esses testes. A Seção 6.2.1 apresenta os experimentos com usuários e a Seção 6.2.2 mostra como foram realizados os experimentos quantitativos.

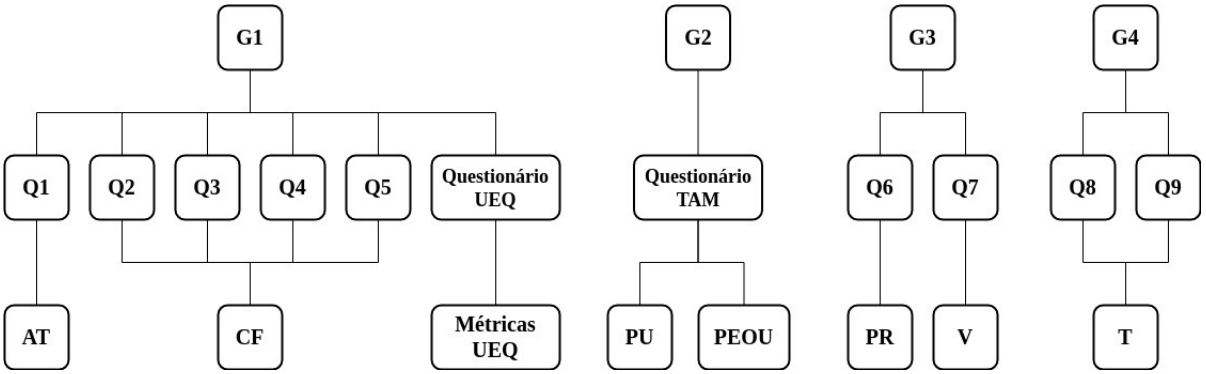


Figura 20: Estrutura hierárquica do modelo GQM desenvolvido nesta dissertação.

6.2.1 Experimentos com Usuários

Os *goals* G1, G2 e G3 buscam analisar o evento *EyeGaze* sob ponto de vista dos usuários de TV digital. O conjunto de *questions* elaboradas a partir desses *goals* configura o questionário usado no experimento com usuários. Nesse experimento, a cada participação, foi explicado ao usuário sobre esta pesquisa, o funcionamento da modalidade de interação usando o *EyeGaze*, as etapas do teste, e o que era esperado que fosse feito. Em seguida, o *eye tracker* Tobii 4C foi calibrado especificamente para o participante, usando o *software Tobii Pro Eye Tracker Manager*[1], da Tobii. Posteriormente, três aplicações[2] foram executadas em sequência, com as quais o usuário interagiu. Por fim, o questionário foi preenchido pelo participante. Esse questionário foi criado usando o aplicativo *Google Forms*[3] e disponibilizado *on-line*. O Anexo A contém todo o conteúdo do questionário, que foi preenchido por 11 participantes (5 homens e 6 mulheres), com idades variando de 37 a 64 anos. Nenhum deles tinha experiência prévia com *eye trackers*. Desses, 63,6% têm pós-graduação, 18,2% o ensino médio completo, 9,1% superior completo e 9,1% fundamental incompleto.

A Figura 21 apresenta a aplicação NCL "Escolhe Vídeo"[4], a primeira executada durante o teste. Essa aplicação exibe inicialmente três imagens à direita. Cada uma dessas imagens pode ser ativada com o olhar por meio do evento *EyeGaze*, fazendo com que um vídeo correspondente seja apresentado. Simultaneamente ao vídeo, duas imagens de botões de "Play" e "Pause" são apresentados. Esses botões também são mídias *gaze-enabled* e, quando o usuário fizer uma interação usando o evento *EyeGaze* nessas mídias, o vídeo que está sendo apresentado retomará sua exibição (i.e., ação "resume" do botão "Play") ou

[1] https://www.tobiipro.com/product-listing/eye-tracker-manager
[2] As aplicações se encontram disponíveis em: https://github.com/eyrebrasil/NCL-Apps-EyeGaze
[3] Sobre o Google Forms: https://www.google.com/forms/about
[4] Vídeo Demo disponível em: https://1drv.ms/u/s!AvKXFJ6ipz46hqgcEWo39TPNhFC4JQ?e=xyetJ1

será pausado (i.e., ação "pause" do botão "Pause"). Durante o teste, o participante deveria olhar por 1 segundo para uma das imagens iniciais à direita (i.e., satisfazendo a condição *onEndEyeGaze*).Depois que o vídeo correspondente iniciasse, deveria então olhar por 1 segundo para o botão de "Pause", pausando o vídeo, e em seguida, deveria fazer o mesmo com o botão de "Play", retomando o vídeo. Além disso, o evento *EyeGaze* também foi usado para destacar com borda amarela as mídias *gaze-enabled* em foco, i.e., as mídias para as quais o usuário olhava (i.e., condição *onBeginEyeGaze* satisfeita), com o objetivo de facilitar a identificação das mídias que possibilitavam interação com o olhar.

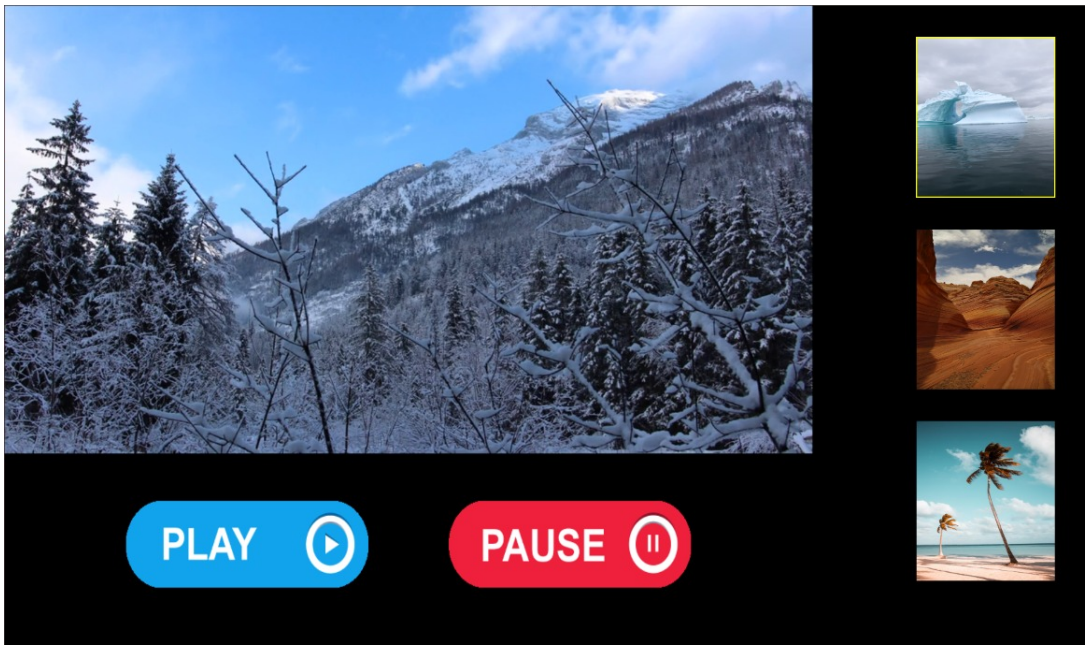


Figura 21: Aplicação Escolhe Vídeo em funcionamento, com vídeo correspondente à imagem escolhida com o olhar e botões para pausá-lo e retomá-lo.

A segunda aplicação exibida ao participante era o "Jogo da Memória". Inicialmente era exibida uma imagem da Figura 22, pedindo que o participante olhasse atentamente a imagem. Essa imagem era exibida por dez segundos e os animais deveriam ser memorizados o melhor possível. Então, uma era exibida a tela da Figura 23, pedindo a identificação do animal que não se encontrava na imagem anterior. Cada uma das imagens disponíveis para escolha era ativável com o evento *EyeGaze*. O usuário deveria identificar a resposta e olhar para a imagem correspondente por 1 segundo para responder à pergunta exibida (i.e., condição *onEndEyeGaze* satisfeita). Em seguida, uma mensagem indicando o acerto ou erro do participante era exibida e a aplicação era encerrada. Nessa aplicação também foi usado o evento *EyeGaze* para destacar com borda amarela as mídias *gaze-enabled* em foco para que o usuário soubesse qual resposta estava selecionando.

A Figura 24 representa a terceira aplicação exibida, uma adaptação do programa inte-

Figura 22: Aplicação Jogo da Memória em funcionamento, com imagem inicial para memorização.

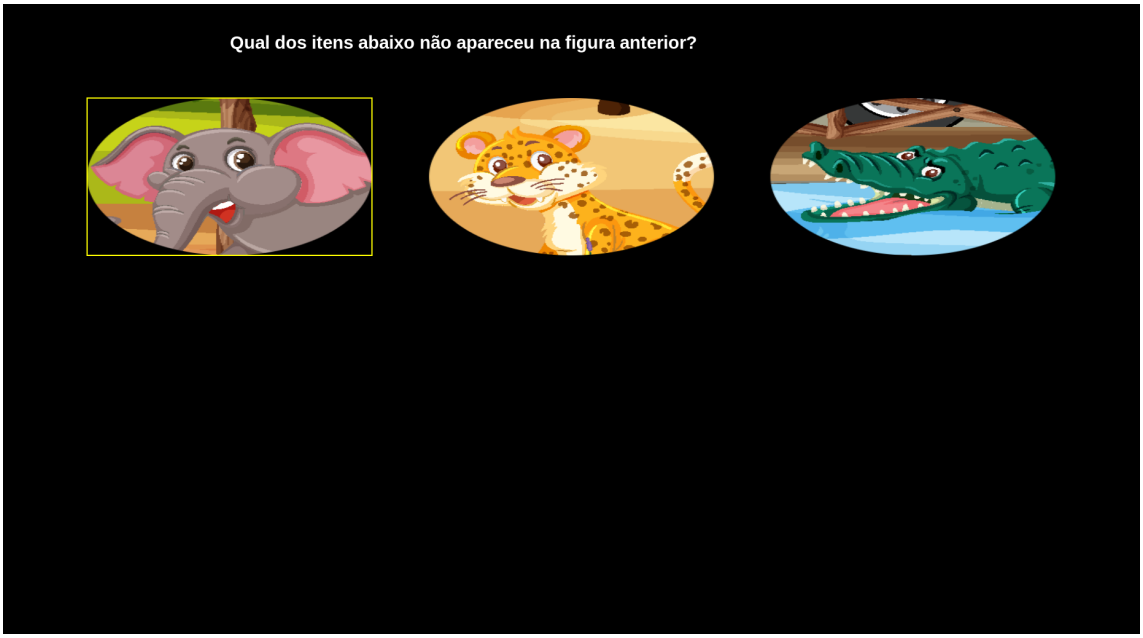


Figura 23: Aplicação Jogo da Memória após o tempo inicial para memorização da imagem, com as opções a serem escolhidas com o evento *EyeGaze*.

rativo "Roteiro do Dia"[5], criado para TV digital interativa. Um vídeo é exibido com uma apresentadora, que em determinado momento do vídeo pede ao espectador que escolha uma opção de local do Rio de Janeiro para visitar a seguir. Nesse momento, na parte inferior direita da tela, surge um quadro com duas opções: Praia de Copacabana e Cen-

[5]Publicado no Clube NCL e disponível em: http://clube.ncl.org.br/node/61

tral do Brasil. Ambas são ativáveis pelo olhar e o usuário deve fixar uma delas durante 1 segundo (i.e., condição *onEndEyeGaze* satisfeita). Um vídeo sobre o local escolhido é então apresentado, e a aplicação pode ser encerrada. Para que o participante saiba qual opção está fixando, cada uma delas é destacada com borda amarela quando em foco (i.e., condição *onBeginEyeGaze*).

Figura 24: Aplicação Roteiro do Dia em funcionamento.

6.2.2 Experimentos Quantitativos

Para responder às questões Q8 e Q9, derivadas do *goal* G4, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação do desempenho sob ponto de vista do sistema*, foram realizados dois tipos de experimentos quantitativos, detalhados nesta subseção. Esses testes foram feitos em uma máquina com as seguintes configurações: Intel® Core™ i5-7500T CPU 2.70GHz; 8GB RAM; 1TB HD e Sistema Operacional Ubuntu 18.

O primeiro experimento foi realizado para responder à Q8, i.e., *Quanto tempo leva para a interação ser iniciada, uma vez que foi identificada?*. O *Gaze Recognition Module* recebe uma lista de mídias associadas ao evento *EyeGaze* para monitorar e, uma vez que o usuário fixa uma dessas mídias por tempo suficiente para atender às condições

especificadas (i.e., o papel de condição usado no conector associado à mídia), o *Gaze Recognition Module* deve notificar que identificou uma interação. Foi considerado que a interação identificada foi iniciada no momento em que a ação estabelecida no conector da mídia é executada. Desta forma, foi criada uma aplicação em NCL[6] executada cem vezes via *bash script*. Nessa aplicação, mostrada na Listagem 6.1, há uma mídia do tipo imagem "img" (linha 3) que deve ter sua apresentação encerrada (i.e., ação de *stop*, linha 8) assim que uma fixação ocular é iniciada nessa mídia (i.e., condição *onBeginEyeGaze*, linhas 5 a 7). A aplicação em execução é mostrada na Figura 25. Para ser possível a execução via *bash script*, foi incluída uma função *FakeNotification()* que notifica a identificação de início da fixação ocular após dois segundos de início da aplicação. Quando o componente *Media* executa a ação de *stop* da imagem, ocorre o momento que representa o início da interação.

```
1  <body>
2    <port id="pImg" component="img"/>
3    <media id="img" descriptor="dsMain" src="image/gelo.jpg"/>
4    <link xconnector="onBeginEyeGaze_Stop">
5      <bind role="onBeginEyeGaze" component="img">
6        <bindParam name="user" value="all"/>
7      </bind>
8      <bind role="stop" component="img"/>
9    </link>
10 </body>
```

Listagem 6.1: Link usado no documento NCL para o primeiro experimento quantitativo.

O segundo experimento foi realizado para responder à Q9, i.e., *Qual o tempo máximo de processamento de cada gaze point à medida que o número de mídias aumenta?*. A cada *gaze point* recebido do dispositivo *eye tracker*, o *Gaze Recognition Module* precisa verificar se o *gaze point* está em uma região pertencente à lista de regiões de mídias recebida para monitoramento. Desta forma, a quantidade de mídias associadas ao evento *EyeGaze* em um documento NCL impacta no tempo de processamento dos pontos recebidos durante a execução da aplicação e consequentemente, no tempo de identificação da interação. Para calcular o tempo máximo, o pior cenário foi considerado, i.e., quando o usuário não está olhando para nenhuma das mídias associadas ao evento *EyeGaze*, sendo necessário verificar todas as regiões da lista de regiões de mídias recebida a cada ponto

[6]Disponível em: https://github.com/eyrebrasil/NCL-Apps-EyeGaze/tree/main/teste1

Figura 25: Aplicação usada para medir o tempo necessário para executar a interação, uma vez que foi identificada a fixação ocular em uma mídia *gaze-enabled*.

lido. Assim, foram criadas quatro aplicações NCL[7], com cinco, vinte, cinquenta e cem mídias respectivamente. A Listagem 6.2 mostra um trecho da aplicação NCL com cem mídias. O funcionamento é o mesmo para todas as quatro aplicações, variando somente o número de mídias. Inicialmente, uma mídia de vídeo (linha 3) é exibida na parte superior esquerda da tela, conforme mostra a Figura 26, de forma que o participante preste atenção ao vídeo, e não às imagens ativáveis pelo olhar localizadas na região inferior direita (linhas 4 a 6). As imagens não são iniciadas (i.e., permanecem até o final da execução em estado *sleeping*), mas por estarem associadas ao evento *EyeGaze* por meio do link que usa o papel *onEndEyeGaze* (linhas 8 a 20), são monitoradas pelo *Gaze Recognition Module* durante a execução. O vídeo de cada uma das aplicações possui duração de dez segundos, o que permitiu o processamento de cerca de oitocentos *gaze points*. Cada uma quatro aplicações foi executada uma vez via *bash script*, e foi necessário que uma pessoa assistisse aos vídeos durante o experimento.

```
1 <body>
2   <port id="pMain" component="video"/>
3   <media id="video" descriptor="dsMain" src="video/pesadelo.mp4"/>
4   <media id="img1" descriptor="dsHid" src="image/praia.jpg"/>
5   ...
6   <media id="img100" descriptor="dsHid" src="image/praia.jpg"/>
7
```

[7]Disponíveis em: https://github.com/eyrebrasil/NCL-Apps-EyeGaze/tree/main/teste2

```
8   <link xconnector="onEndEyeGaze_Stop">
9     <bind role="onEndEyeGaze" component="img1">
10      <bindParam name="user" value="all"/>
11    </bind>
12    <bind role="stop" component="img1"/>
13  </link>
14  ...
15  <link xconnector="onEndEyeGaze_Stop">
16    <bind role="onEndEyeGaze" component="img100">
17      <bindParam name="user" value="all"/>
18    </bind>
19    <bind role="stop" component="img100"/>
20  </link>
21 </body>
```

Listagem 6.2: Trecho do documento NCL de 100 mídias para o segundo experimento quantitativo.

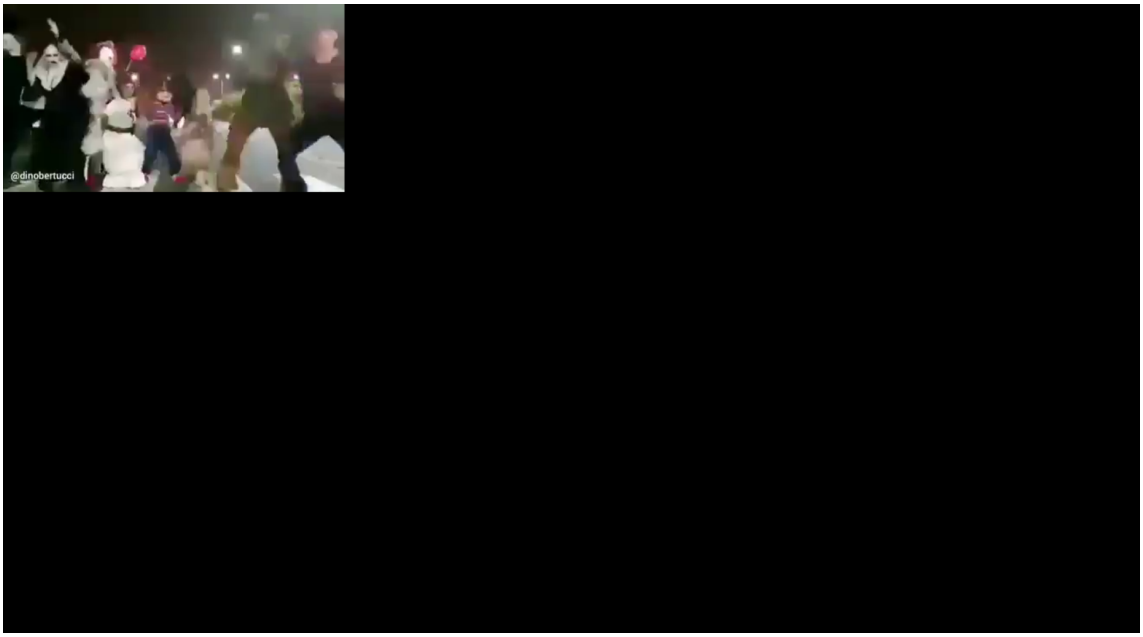


Figura 26: Aplicação com 100 mídias *gaze-enabled*, utilizada para verificar o tempo máximo de processamento dos *gaze points* a cada leitura realizada pelo sensor *eye tracker*.

6.3 Resultados

Esta seção analisa e discute os dados obtidos dos questionários e experimentos realizados para responder aos *goals* definidos nesta dissertação.

Para o G1, foram usados os valores médios das respostas às questões e a metodologia já estabelecida para o UEQ, que usa a ferramenta *data analysis tool*[8] para análise dos dados. Nos *goals* G2 e G3 foram usados os valores médios das respostas às questões. Para responder ao G4 foi usado o tempo médio obtido nos experimentos quantitativos.

6.3.1 G1 - Análise dos Resultados

Para alcançar o *goal* G1, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação da experiência dos usuários*, foram elaboradas questões e métricas especificamente para esta dissertação, que em conjunto com o questionário UEQ, indicam se o objetivo G1 foi alcançado.

Na Seção 6.1.1, foram definidas duas métricas para responder às questões Q1 a Q5: atratividade (AT) e conforto (CF). A métrica Atratividade foi medida usando-se a média aritmética das respostas dadas em escala Likert para a questão Q1. Conforto foi medida a partir da média aritmética para as respostas dadas em escala Likert para as questões Q2 a Q5. A Figura 27 mostra as médias obtidas para as questões Q1 a Q5, apresentadas na Tabela 6 e seus respectivos intervalos de confiança de 95%. A questão Q1 obteve média de 4,8, indicando que os participantes gostaram de interagir usando os olhos. Os usuários que concordaram de forma parcial com a Q1, relataram algum nível de fadiga dos olhos, pescoço ou ombros (Q3, Q4 e Q5), indicando que esse pode ter sido um fator influenciador. Em relação à métrica conforto, a questão Q2 obteve média 4,7, apontando que os participantes consideraram confortável usar o *EyeGaze* para interagir com a TV digital. Ainda, as questões Q3 e Q4 obtiveram 1,3, indicando que os usuários tendem a discordar totalmente da afirmação de que ao usar o *EyeGaze* para interagir, sentiram fadiga nos ombros e pescoço. A questão Q5, porém, de média 1,6, indica que os usuários somente discordam parcialmente sobre sentirem fadiga nos olhos.

Conforme apresentado na Seção 6.1.1, o UEQ não produz um *score* geral, e como métricas são usadas as escalas (i.e., atratividade, transparência, eficiência, controle, estimulação e inovação) e as classificações (i.e., atratividade, pragmática e qualidade hedônica). A Figura 28 mostra a média de valores obtidos para cada uma das escalas, assim como

[8]Disponível em: https://www.ueq-online.org/Material/Short_UEQ_Data_Analysis_Tool.xlsx

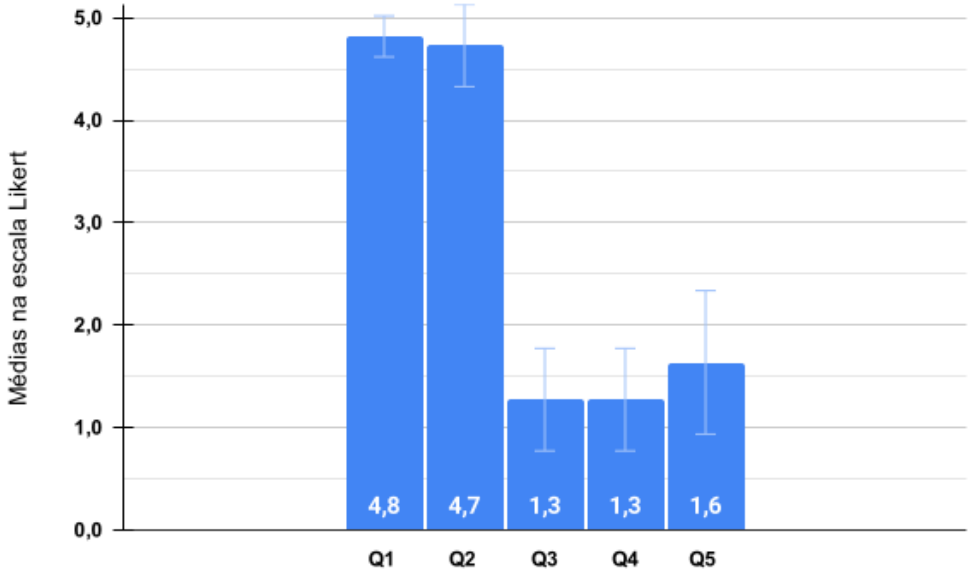


Figura 27: Médias obtidas para as questões Q1 a Q5 do *goal* G1.

suas respectivas variâncias. Os valores da média variam de -3 (horrivelmente ruim) a 3 (extremamente bom), sendo que valores maiores que 0,8 representam uma avaliação positiva. Essas médias são definidas em relação a um *benchmark*(SCHREPP; HINDERKS; THOMASCHEWSKI, 2017), que contém dados de 21.175 pessoas de 468 estudos sobre diferentes produtos. Todas as escalas obtiveram valor considerado excelente em comparação ao *benchmark*, i.e., os valores obtidos estão dentro da margem de 10% dos melhores resultados entre os estudos que compõem o *benchmark*. Os resultados estão listados na Figura 28, variando de excelente a ruim. A escala atratividade obteve valor 2,88 e a transparência 2,78, apontando que os participantes gostaram de usar o produto e o consideraram de fácil aprendizado. A eficiência obteve valor 2,90 e o controle 2,48, indicando que os participantes consideraram que o evento *EyeGaze* pode ser utilizado sem esforço desnecessário, de forma eficiente e rápida, se sentindo no controle da interação. A escala estimulação obteve média 2,88 e a inovação 2,08, assinalando uma avaliação positiva em relação a ser empolgante usar o *EyeGaze* e considerá-lo inovador.

Conforme apresentado na Seção 6.1.1, as escalas podem ser agrupadas em classificações. Os valores obtidos para esses grupos estão expostos na Figura 29. Atratividade obteve média 2,88, pragmática 2,72 e qualidade hedônica 2,48, considerados excelentes segundo o *benchmark* (SCHREPP; HINDERKS; THOMASCHEWSKI, 2017), e indicando que os usuários tiveram uma boa impressão geral do produto, e também consideraram excelentes seus aspectos pragmáticos e hedônicos.

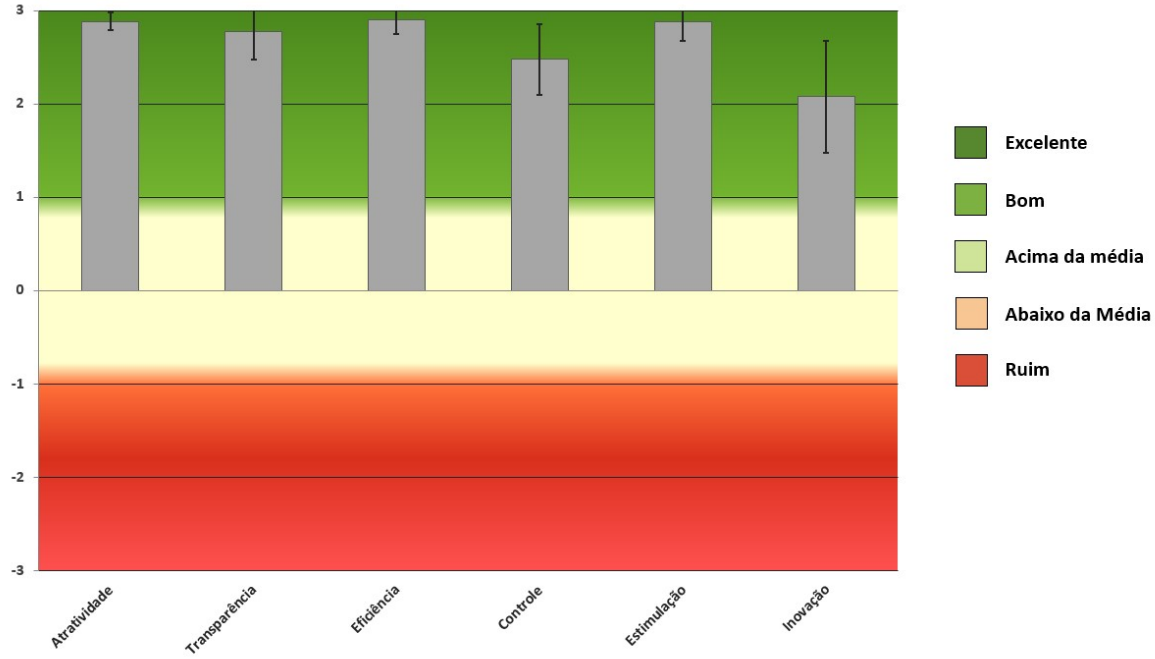


Figura 28: Valores obtidos para as escalas do UEQ.

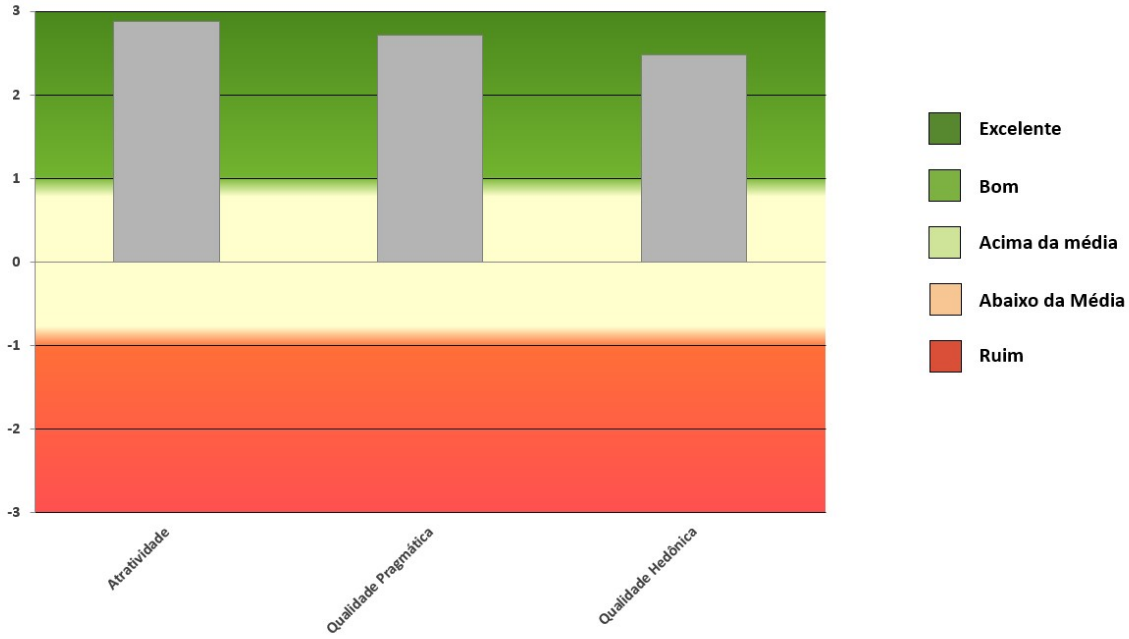


Figura 29: Valores obtidos para as classificações do UEQ.

Antes da análise dos dados coletados das respostas do UEQ, foram eliminadas as respostas de um dos participantes, consideradas suspeitas (i.e., não respondida seriamente ou de forma randômica) pela heurística usada pela ferramenta *data analysis tool*[9] do UEQ.

[9]Disponível em: https://www.ueq-online.org/Material/Short_UEQ_Data_Analysis_Tool.xlsx

Nessa heurística, a ideia é verificar se a melhor e pior avaliação dos itens de uma escala têm diferença maior que 3. Se isso ocorre para 3 escalas ou mais, é recomendado remover o conjunto de avaliações do participante em questão, para evitar erros na interpretação das métricas. No caso do participante em questão que teve suas respostas removidas para o UEQ, ocorreu diferença maior que 3 em 5 das 6 escalas.

A partir dos resultados obtidos por meio das métricas de atratividade e conforto para as questões Q1 a Q5, e da análise das métricas estabelecidas para o UEQ, que apontou resultado excelente para avaliação da experiência do usuário com o evento *EyeGaze* é possível concluir que o objetivo G1 foi alcançado.

6.3.2 G2 - Análise dos Resultados

Para alcançar o *goal* G2, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação da aceitação dessa tecnologia sob ponto de vista dos usuários*, foi utilizado questionário baseado no modelo TAM.

Conforme apresentado na Seção 6.1.2, as questões definidas para o *goal* G2 segundo o TAM estão agrupadas em utilidade percebida (PU) e percepção da facilidade de uso (PEOU), sendo ambos os grupos interpretados como métricas para o G2. Desta forma, a PU foi calculada usando a média aritmética das respostas às questões PU1 a PU6. A PEOU foi medida usando a média das respostas às questões PEOU1 à PEOU6.

A Figura 30 mostra a média obtida para as questões PU1 a PU6, apresentadas na Tabela 9 e seus respectivos intervalos de confiança. Para a PU5, que obteve média 5, todos os participantes concordaram totalmente com a afirmação de que o *EyeGaze* seria útil para interagir com aplicações de TV digital, portanto, não é possível calcular intervalo de confiança. A maior variabilidade foi obtida na questão PU1, com média de 4,6, em que 9 dos 11 participantes concordaram totalmente com a afirmação de que o *EyeGaze* permitiria interagir mais rapidamente com a TV digital, mas um deles concordou apenas de forma parcial e por último, um deles discordou parcialmente dessa afirmação. A média geral para a utilidade percebida foi de 4,8, indicando que de maneira geral os participantes concordaram que o *EyeGaze* melhorará o desempenho deles para interagir com a TV digital.

A Figura 31 mostra a média calculada para as questões PEOU1 a PEOU6, descritas na Tabela 9 e seus respectivos intervalos de confiança. A maior variabilidade foi obtida na questão PEOU5, com média de 4,5, em que 9 dos 11 participantes concordaram total-

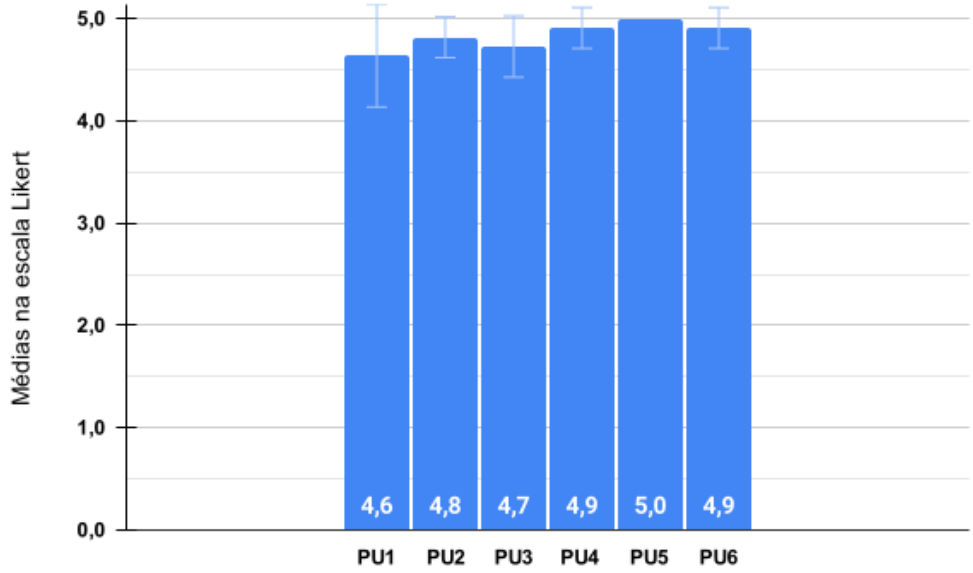


Figura 30: Médias obtidas para as questões PU1 a PU6 do *goal* G2.

mente com a afirmação de que o *EyeGaze* exigia pouco esforço físico para interagir com a TV digital, mas um deles concordou apenas de forma parcial e por último, um deles discordou totalmente dessa afirmação. O participante que discordou totalmente, não relatou nenhuma fadiga nos olhos, pescoço ou ombros, i.e., discordou também totalmente das afirmações descritas na Tabela 6 sob forma das questões Q3, Q4 e Q5 do *goal* G1, o que parece ser uma inconsistência. O mesmo usuário também concordou totalmente com a PEOU6, i.e., de que o *EyeGaze* exigiria pouco esforço mental para interagir. A PEOU4 obteve a menor média (4,5), uma vez que 6 dos 11 participantes concordaram apenas de forma parcial com a afirmação de que seria fácil tornar-se hábil no uso do *EyeGaze*. O fato de que nenhum dos participantes possuía familiaridade com um dispositivo *eye tracker* pode ter influenciado nessas respostas. Mesmo assim, a média de 4,5 para a PEOU4 e a média geral de 4,7 para a percepção da facilidade de uso indicam uma avaliação positiva em relação à facilidade de aprendizado e do quanto os participantes acreditam que usar o *EyeGaze* seria livre de esforço.

A partir dos resultados obtidos para as métricas de utilidade percebida (4,8) e percepção da facilidade de uso (4,7), que apontaram avaliações positivas em relação à aceitação do evento *EyeGaze* para interação com a TV digital sob ponto de vista dos usuários, é possível concluir que o objetivo G2 foi alcançado.

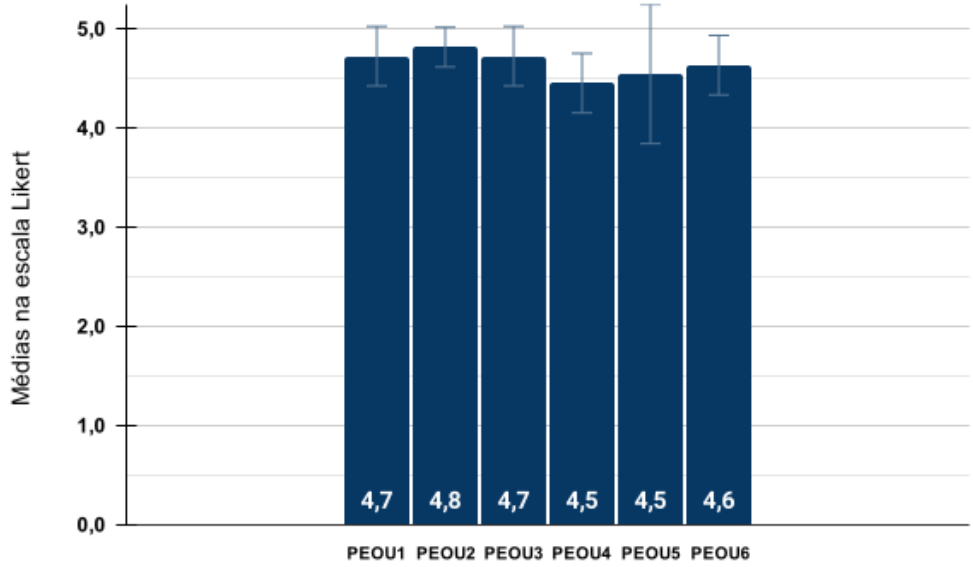


Figura 31: Médias obtidas para as questões PEOU1 a PEOU6 do *goal* G2.

6.3.3 G3 - Análise dos Resultados

Para alcançar o *goal* G3, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação do desempenho sob ponto de vista dos usuários*, foram definidas duas métricas de avaliação, apresentadas na Seção 6.1.3: precisão (PR) e velocidade (V). A precisão é calculada a partir da média aritmética das respostas dadas à questão Q6 e a velocidade, a partir da média das respostas à Q7. As questões Q6 e Q7, descritas na Tabela 11, obtiveram média igual a 5, i.e., todos os participantes concordaram totalmente com as afirmações de que o evento *EyeGaze* tem boa precisão e velocidade durante a interação com a TV digital. É interessante observar que embora a PU1 (*Usar o Eye Gaze para interagir com aplicações de TV digital me permitiria interagir mais rapidamente*, analisada na Seção 6.3.2, tenha a maior variabilidade dentre as questões de utilidade percebida, em que um dos participantes discordou parcialmente dessa afirmação, todos os usuários consideraram que o *EyeGaze* tem boa velocidade durante a interação. Desta forma, é possível concluir que na percepção dos usuários, o *EyeGaze* tem um bom desempenho para interação com a TV digital e o objetivo G3 foi alcançado.

6.3.4 G4 - Análise dos Resultados

Para alcançar o *goal* G4, i.e., *Analisar o evento EyeGaze para interação com a TV digital com o propósito de avaliação do desempenho sob ponto de vista do sistema*, foi definida

a métrica tempo (T), descrita na Seção 6.1.4. Essa métrica foi usada para responder às questões Q8 e Q9, presentes na Tabela 13, a partir da média aritmética dos tempos de execução dos experimentos descritos na Seção 6.2.2.

Para a questão Q8, i.e., *Quanto tempo leva para a interação ser iniciada, uma vez que foi identificada?*, foi obtido o valor médio de 0,190 milissegundos, com Desvio Padrão de 0,394 e confiança de 0,077 para 100 execuções. Desta forma, em média, leva menos de 1 milissegundo para a interação ser iniciada, o que é um ótimo resultado.

Para a questão Q9, i.e., *Qual o tempo máximo de processamento de cada gaze point à medida que o número de mídias aumenta?*, os resultados de tempo médio, assim como os intervalos de confiança estão exibidos na Figura 32. As 4 aplicações foram executadas durante 10 segundos cada e a média foi calculada a partir da média dos tempos de processamento de 800 *gaze points* lidos durante a execução dessas aplicações. Para uma aplicação com 5 mídias usando o evento *EyeGaze* o tempo máximo médio foi de 0,149 milissegundos. Ao quadruplicar o número de mídias para 20, o tempo máximo assumiu valor médio de 0,192 milissegundos. Para a aplicação com 50 mídias, o tempo máximo foi de 0,414 milissegundos. Com 100 mídias associadas, o tempo máximo médio encontrado foi de 0,701 milissegundos. Os resultados mostram que mesmo com um número grande de 100 mídias associadas ao evento *EyeGaze*, no pior cenário possível, i.e., onde *Gaze Recognition Module* precisa verificar todas as 100 mídias associadas a cada *gaze point*, o tempo máximo de processamento de cada *gaze point* ainda é menor que 1 milissegundo.

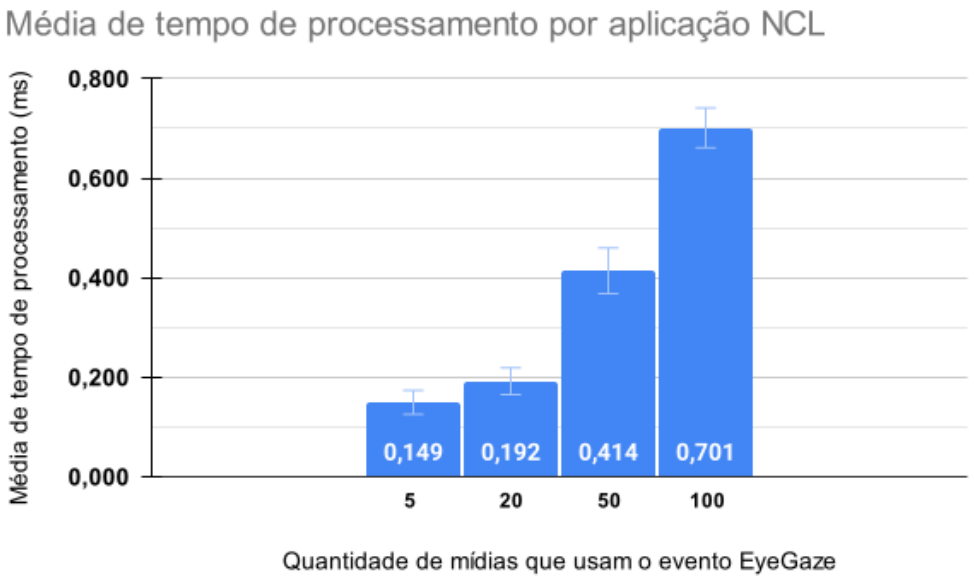


Figura 32: Médias de tempo máximo de processamento para cada *gaze point* em aplicações NCL com 5, 20, 50 e 100 mídias associadas ao evento *EyeGaze*.

Desta forma, com os resultados obtidos a partir de experimentos quantitativos para responder às questões Q8 e Q9, é possível concluir que o *EyeGaze* tem um bom desempenho para interação com a TV digital e o objetivo G4 foi alcançado.

6.3.5 Ameaças à Validade dos Experimentos

Há alguns fatores que devem ser considerados como ameaças à validade dos experimentos realizados, uma vez que podem ter influenciado nos resultados obtidos.

O primeiro fator é o espaço amostral pequeno. Devido ao cenário pandêmico, e à natureza do sensor, que exige a presença do participante no local dos experimentos, o espaço amostral conta com apenas 11 participantes, o que impediu algumas observações, e.g., associação entre idade e fadiga, e pode ter deixado os resultados mais favoráveis.

O segundo fator é experiência prévia dos participantes. Embora nenhum deles tenha relatado experiência prévia com *eye trackers*, não sabemos qual a bagagem de cada um em relação a outras tecnologias de interação multimodal ou imersivas, que caso existente, pode ter influenciado na aceitação do novo evento de maneira positiva.

Por fim, o terceiro fator é o *setup* utilizado para os experimentos quantitativos, usados para responder ao *goal* G3, relacionado ao desempenho do sistema. O *setup* usado é distante do ambiente real de TV digital, que é muito mais restritivo, o que influenciou nos resultados positivos obtidos.

6.3.6 Considerações Finais

Para a avaliação da solução proposta nesta dissertação foi planejada uma avaliação segundo o paradigma GQM. Foram estabelecidos quatro objetivos (ou *goals*) e para cada um deles, conjuntos de questões para respondê-los. Para cada grupo de questões, foram definidas métricas para respondê-las e desta forma, identificar se os objetivos estipulados foram alcançados. Portanto, o evento *EyeGaze* para interação com a TV digital foi analisado em relação à experiência dos usuários, aceitação da tecnologia e desempenho.

Para efetuar a coleta de dados e permitir o cálculo das métricas definidas para avaliação do *EyeGaze*, foram realizados experimentos com usuários e experimentos quantitativos. O experimento com usuários recolheu respostas de 11 participantes (5 homens e 6 mulheres), com idades variando de 37 a 64 anos. Todos sem experiência prévia com *eye trackers* e sem deficiências relacionadas a dificuldade de movimento dos olhos ou de

membros superiores. Embora todos os participantes tenham protestado a respeito do tamanho excessivo do questionário, todos se mostraram bastante empolgados com a nova possibilidade de interação com os olhos para a TV digital e a elogiaram bastante como sendo muito interessante, inovadora, eficaz, prática e um excelente avanço.

Foi verificado se havia alguma relação entre idade e nível de fadiga percebidos durante os experimentos. Não foi possível identificar uma ligação entre idade e fadiga. Apenas um dos participantes, de 39 anos relatou algum nível de fadiga nos ombros, pescoço e olhos. Porém, 72,3% dos participantes têm mais de 39 anos. Apenas uma pessoa com mais de 39 anos relatou fadiga nos olhos, indicando ser necessário um maior espaço amostral para investigação.

Todos os objetivos definidos foram alcançados ao final da etapa de avaliação, observando-se bons resultados em relação à experiência do usuário, considerada excelente para todas as escalas e classificações do UEQ. Além disso, teve avaliações positivas em afirmações relacionadas a conforto, utilidade percebida e percepção da facilidade de uso. Um bom desempenho também foi observado sob ponto de vista do sistema e dos usuários.

7 Conclusão

Esta dissertação teve como objetivo geral "*Incorporar uma modalidade de interação por meio do olhar, baseada em fixação ocular, ao ambiente de TV digital que utiliza o middleware Ginga-NCL*". Para isso, foram estabelecidas duas questões de pesquisa, que foram estudadas e solucionadas. Para responder a primeira questão de pesquisa "*Como fazer com que o ambiente de TV digital possa oferecer interações por meio do olhar?*" esta dissertação apresentou uma extensão do *middleware* Ginga-NCL para fornecer interação por meio do olhar usando a tecnologia de rastreamento ocular (*eye tracking*). Para isso, foi implementado o evento de fixação ocular de regiões *EyeGaze* e os papéis de condição *onBeginEyeGaze*, *onEndEyeGaze* e *onAbortEyeGaze*, permitindo seu uso em documentos NCL. Além disso, esta dissertação propôs uma arquitetura de integração, que pode ser utilizada na integração de sensores *eye trackers* e se adapta à definição atual do Ginga e de NCL, de forma a facilitar sua adoção. O evento proposto neste trabalho pode ser utilizado em diferentes tipos de aplicações multimídia, e permite que interações sejam realizadas por meio da fixação do olhar.

Esta dissertação apresentou também como estruturar a troca de dados utilizando objetos JSON, e como realizar o mapeamento de regiões de aplicações do Ginga-NCL para regiões compostas por pontos normalizados (0 a 1). Por ser um formato de troca de dados rápida entre sistemas de maneira universal, a utilização de objeto JSON evita que alterações de dispositivos *eye trackers* prejudiquem a interoperabilidade da ferramenta. O objeto JSON também foi estruturado de forma a permitir que múltiplos usuários sejam especificados na aplicação NCL.

Para responder à segunda questão de pesquisa "*Uma vez implementada a modalidade de interação ocular no ambiente de TV digital, qual a percepção dos usuários sobre este novo tipo de interação com a TV?*", foi realizada uma avaliação estruturada segundo o paradigma GQM da solução proposta nesta dissertação. O evento *EyeGaze* para interação com a TV digital foi analisado em relação à experiência dos usuários, aceitação da tecnologia e desempenho. Todos os objetivos definidos foram alcançados ao final da

etapa de avaliação, observando-se bons resultados em relação à experiência do usuário, considerada excelente para todas as escalas e classificações do UEQ. Além disso, teve avaliações positivas em afirmações relacionadas a conforto, utilidade percebida e percepção da facilidade de uso. Um bom desempenho também foi observado sob ponto de vista do sistema e dos usuários.

Com a solução proposta nesta dissertação, também espera-se facilitar a criação de aplicações multimídia com suporte a interação por meio do olhar e com isso, prover uma nova forma de interação para usuários que não querem, ou principalmente, não podem utilizar modalidades tradicionais, devido à alguma deficiência. Além disso, o evento *EyeGaze* permite não só uma alternativa à utilização dessas modalidades, mas a especificação de aplicações especificamente para o uso da fixação ocular (e.g., jogos e atividades educacionais).

Os seguintes trabalhos foram publicados durante o desenvolvimento desta dissertação:

- MONTEVECCHI, Eyre Brasil B.; JOSUÉ, Marina I. P.; BARRETO, Fábio; ABREU, Raphael S. de; SAADE, Débora C. M. Providing Eye Gaze Interaction for Ginga-NCL Applications. In: PROCEEDINGS of the Brazilian Symposium on Multimedia and the Web. São Luis, Brazil: Association for Computing Machinery, 2020. (WebMedia '20), p. 297–303.

- BARRETO, Fábio; ABREU, Raphael S. de; MONTEVECCHI, Eyre Brasil B.; JOSUÉ, Marina I. P.; VALENTIM, Pedro A.; SAADE, Débora C. M. Extending Ginga-NCL to Specify Multimodal Interactions With Multiple Users. In: PROCEEDINGS of the Brazilian Symposium on Multimedia and the Web. São Luis, Brazil: Association for Computing Machinery, 2020. (WebMedia '20), p. 281–288.

- IVANOV, Marina I. P.; MONTEVECCHI, Eyre Brasil B.; ABREU, Raphael S. de; BARRETO, Fábio; SANTOS, Joel A. F. dos; SAADE, Débora C. M. Ambientes Multissensoriais Aplicados à Saúde: Desenvolvimento de Aplicações e Tendências Futuras. Capítulo 2 do Livro de Minicursos do SBCAS 2020. Sociedade Brasileira de Computação, 2020.

- BARRETO, Fábio; MONTEVECCHI, Eyre Brasil B.; ABREU, Raphael S. de; SANTOS, Joel A. F. dos; SAADE, Débora C. M. Providing Multimodal User Interaction in NCL. In: ANAIS Estendidos do XXV Simpósio Brasileiro de Sistemas Multimídia e Web. Florianópolis: SBC, 2019. p. 203–204.

- BARRETO, Fábio; MONTEVECCHI, Eyre Brasil B.; ABREU, Raphael S. de; SANTOS, Joel A. F. dos; SAADE, Débora C. M. Providing multi-user in NCL with userAgent and UserProfile. In: ANAIS Estendidos do XXV Simpósio Brasileiro de Sistemas Multimídia e Web. Florianópolis: SBC, 2019. p. 205–206.

- BARRETO, Fabio; MONTEVECCHI, Eyre Brasil B.; ABREU, Raphael S. de; SANTOS, Joel A. F. dos; SAADE, Débora C. M. NCL: Storing user settings in media node. In: ANAIS Estendidos do XXV Simpósio Brasileiro de Sistemas Multimídia e Web. Florianópolis: SBC, 2019. p. 201–202.

- BARRETO, Fabio; MONTEVECCHI, Eyre Brasil B.; ABREU, Raphael S. de; SANTOS, Joel A. F. dos; SAADE, Débora C. M. NCL: Storing ambient settings in media node. In: ANAIS Estendidos do XXV Simpósio Brasileiro de Sistemas Multimídia e Web. Florianópolis: SBC, 2019. p. 199–200.

- FARIAS, Flávio; MONTEVECCHI, Eyre Brasil B.; BOKEHI, José Raphael; SANTANA, Rosimere F.; SAADE, Débora C. M. Memo-VR: Exercício Cognitivo para Idosos Utilizando Realidade Virtual e Interface com as Mãos. In: ANAIS do XX Simpósio Brasileiro de Computação Aplicada à Saúde. Evento Online: SBC, 2020. p. 434–439.

Além disso, o trabalho intitulado "*Providing Multimodal and Multi-User Interactions for Digital TV Applications*" foi aprovado pela *Multimedia Tools and Applications* e aguarda publicação.

A especificação do novo tipo de evento de fixação ocular *EyeGaze*, assim como seus papéis de condição *onBeginEyeGaze*, *onEndEyeGaze* e *onAbortEyeGaze*, compõem a parte de interação multimodal na proposta de nova versão do NCL, o NCL 4.0[1], submetida ao projeto da TV 3.0 do Sistema Brasileiro de Televisão Digital (SBTVD) como extensão do *middleware* Ginga para a futura geração do sistema de TV digital do Brasil. A proposta NCL 4.0 foi aprovada na segunda fase (testes e avaliações) e se encontra na terceira fase do projeto da TV 3.0.

7.1 Trabalhos Futuros

Para aplicações multiusuário, consideradas na arquitetura e implementação, a diferenciação de múltiplos usuários simultâneos no contexto da aplicação NCL e a associação desses

[1] https://forumsbtvd.org.br/tv3_0/

usuários aos seus respectivos sensores é um trabalho em andamento e que deve ter sua integração investigada. Desta forma, a implementação aqui apresentada (usando o sensor *eye tracker* Tobii 4C) é mais adequada para uso por apenas um usuário.

Como trabalho futuro, a contribuição de *eye trackers* no processo de autoria deve ser avaliada. Com a extensão da linguagem NCL para que seja possível especificar documentos NCL que usam o evento de interação ocular *EyeGaze*, o processo de autoria de aplicações multimídia em NCL teoricamente é simplificado, uma vez que não será necessário implementar e adequar o funcionamento do sensor *eye tracker* à aplicação.

Uma avaliação da própria modalidade de interação por meio da fixação ocular também deve ser realizada com usuários com deficiência motora, que podem se beneficiar do trabalho aqui apresentado. Além disso, interações por meio do olhar não se limitam a fixação ocular e também podem usar gestos oculares (i.e. sequências de fixações) ou comandos rápidos (i.e. piscada) para selecionar e arrastar, por exemplo. Desta forma, essas outras formas de interação podem futuramente ser também adicionadas à linguagem NCL e suportadas pelo Ginga-NCL.

A avaliação de desempenho do sistema, relacionada ao *goal* G3, precisa ser novamente executada. Os experimentos quantitativos usaram um *setup* distante do ambiente real de TV digital (mais restritivo). É necessário realizar essa avaliação em um ambiente similar ao do receptor de TV digital, utilizando por exemplo um Raspberry PI como plataforma de execução do *middleware* Ginga-NCL.

Ainda como contribuição futura, os sensores *eye trackers* podem ser melhor explorados em sua capacidade de realizar medições do tamanho da pupila (e.g. para detecção de estresse do usuário) e de criação de mapas baseados nas leituras de *gaze points*, permitindo a análise de atenção e distração do usuário, assim como detecção de áreas de maior interesse de mídias apresentadas.

Embora a arquitetura proposta possa ser utilizada para diferentes modelos de sensores *eye trackers*, o funcionamento da implementação atual está atrelado ao modelo *eye tracker* Tobii 4C. Como trabalho futuro, a implementação deve permitir a separação entre o *Gaze Recognition Module* e o que é específico para funcionamento do sensor, permitindo a integração de diferentes modelos de sensores ao Ginga-NCL de maneira facilitada. Além disso, por suas especificações, o modelo *eye tracker* Tobii 4C pode não ser o mais adequado para uso em ambiente de TV, uma vez que limita a uma distância relativamente curta o quão longe do sensor o usuário pode ficar para realizar a interação. Para esse tipo de ambiente, modelos de *eye tracking* do tipo *head-mounted* (e.g. acoplados a óculos) seriam

mais indicados que os remotos (tipo do *eye tracker* Tobii 4C).

Outro trabalho de implementação futuro está relacionado ao uso dos dados de propriedade de mídia ao invés dos dados de região para estabelecer as regiões ativáveis pelo olhar. O *Gaze Recognition Module* recebe os dados de região da mídia advindos do processo de *parsing* e não está ciente do que ocorre em tempo de execução da aplicação NCL. Consequentemente, se as propriedades de uma mídia mudam informações de sua região durante a execução do documento NCL, o *Gaze Recognition Module* terá uma lista obsoleta de regiões e não funcionará de maneira apropriada. Isto é uma limitação da implementação atual e precisa ser refinada futuramente.

REFERÊNCIAS

ABNT. **Digital terrestrial television - Data coding and transmission specification for digital broadcasting - Part 2: Ginga-NCL for fixed and mobile receivers - XML application language for application coding**. [S. l.: s. n.], 2011. ABNT NBR 15606-2:2011 standard.

ANTONACCI, Meire Juliana et al. NCL: Uma Linguagem Declarativa para Especificação de Documentos Hipermıdia na Web. **VI Simpósio Brasileiro de Sistemas Multimıdia e Hipermıdia-SBMıdia2000, Natal, Rio Grande do Norte**, 2000.

ASSOCIATION, ALS. **Augmentative and Alternative Communication**. [S. l.: s. n.], 2022. Acessado em: 24 de março de 2022. Disponível em: <https://www.als.org/navigating-als/living-with-als/therapies-care/augmentative-alternative-communication>.

BARRETO, Fábio. **Uma Proposta de Extensão do Middleware Ginga-NCL para Interação Multimodal e Suporte Multiusuário em Ambientes Hipermídia**. 2021. Tese (Doutorado) – Instituto de Computação, UFF, Niterói, RJ, Brasil.

BARRETO, Fabio; MONTEVECCHI, Eyre Brasil B. et al. NCL: Storing user settings in media node. In: ANAIS Estendidos do XXV Simpósio Brasileiro de Sistemas Multimídia e Web. Florianópolis: SBC, 2019. p. 201–202. DOI: 10.5753/webmedia_estendido.2019.8166. Disponível em: <https://sol.sbc.org.br/index.php/webmedia_estendido/article/view/8166>.

BARRETO, Fábio; ABREU, Raphael S. de; MONTEVECCHI, Eyre Brasil B. et al. Extending Ginga-NCL to Specify Multimodal Interactions With Multiple Users. In: PROCEEDINGS of the Brazilian Symposium on Multimedia and the Web. São Luıs, Brazil: Association for Computing Machinery, 2020. (WebMedia '20), p. 281–288. ISBN 9781450381963. DOI: 10.1145/3428658.3431076.

BARRETO, Fábio; ABREU, Raphael S.; SANTOS, Joel A. F. dos et al. Authoring sensory effects in NCL. In: ANAIS Estendidos do XXV Simpósio Brasileiro de Sistemas

Multimídia e Web. Florianópolis: SBC, 2019. p. 191–192. DOI: 10.5753/webmedia_estendido.2019.8162. Disponível em: <https://sol.sbc.org.br/index.php/webmedia_estendido/article/view/8162>.

BARRETO, Fábio; MONTEVECCHI, Eyre Brasil B. et al. Providing multi-user in NCL with userAgent and UserProfile. In: ANAIS Estendidos do XXV Simpósio Brasileiro de Sistemas Multimídia e Web. Florianópolis: SBC, 2019. p. 205–206.

_____. Providing Multimodal User Interaction in NCL. In: ANAIS Estendidos do XXV Simpósio Brasileiro de Sistemas Multimídia e Web. Florianópolis: SBC, 2019. p. 203–204. DOI: 10.5753/webmedia_estendido.2019.8167.

BASILI, Victor R. Software Modeling and Measurement: The Goal Question Metric Paradigm, CS-TR-2956. Computer Science Technical Report Series, University of Maryland, College Park, MD, (UMIACS-TR-92-96), set. 1992.

BISWAS, Pradipta; LANGDON, Pat. A New Interaction Technique Involving Eye Gaze Tracker and Scanning System. In: PROCEEDINGS of the 2013 Conference on Eye Tracking South Africa. Cape Town, South Africa: Association for Computing Machinery, 2013. (ETSA '13), p. 67–70. ISBN 9781450321105. DOI: 10.1145/2509315.2509322. Disponível em: <https://doi.org/10.1145/2509315.2509322>.

CALDIERA, Victor R Basili1 Gianluigi; ROMBACH, H Dieter. The Goal Question Metric Approach. **Encyclopedia of Software Engineering**, p. 528–532, 1994.

CARVALHO, Nuno A; OLIVEIRA, José P; PEREIRA, José. Evaluating Throughput Stability of Protocols for Distributed Middleware. In_____. **On the Move to Meaningful Internet Systems: OTM 2009**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. Springer, p. 600–613. ISBN 978-3-642-05148-7. DOI: 10.1007/978-3-642-05148-7_44.

CASARINI, Matteo; PORTA, Marco; DONDI, Piercarlo. A Gaze-Based Web Browser with Multiple Methods for Link Selection. In: ACM Symposium on Eye Tracking Research and Applications. Stuttgart, Germany: Association for Computing Machinery, 2020. (ETRA '20 Adjunct). ISBN 9781450371353. DOI: 10.1145/3379157.3388929.

CHEN, Chun-Ching; LIN, Yi-Sin. Study on The Interactive Interface Design of Gaze Input Smart TV. In: 2018 IEEE International Conference on Applied System Invention (ICASI). [S. l.: s. n.], 2018. p. 196–199. DOI: 10.1109/ICASI.2018.8394566.

CROCKFORD, Douglas. **The application/json Media Type for JavaScript Object Notation (JSON)**. [S. l.], 2006.

DAVIS, Fred D. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. **MIS Quarterly**, Management Information Systems Research Center, University of Minnesota, v. 13, n. 3, p. 319–340, 1989. ISSN 02767783. DOI: 10.2307/249008.

DUCHOWSKI, Andrew T. A Breadth-First Survey of Eye-tracking Applications. **Behavior Research Methods, Instruments, & Computers**, Springer, v. 34, n. 4, p. 455–470, 2002. DOI: 10.3758/BF03195475.

FEIT, Anna Maria et al. Toward Everyday Gaze Input: Accuracy and Precision of Eye Tracking and Implications for Design. In: PROCEEDINGS of the 2017 CHI Conference on Human Factors in Computing Systems. Denver, Colorado, USA: Association for Computing Machinery, 2017. (CHI '17), p. 1118–1130. ISBN 9781450346559. DOI: 10.1145/3025453.3025599.

GHINEA, Gheorghita et al. Mulsemedia: State of the Art, Perspectives, and Challenges. **ACM Trans. Multimedia Comput. Commun. Appl.**, Association for Computing Machinery, New York, NY, USA, v. 11, 1s, out. 2014. ISSN 1551-6857. DOI: 10.1145/2617994.

HARDMAN, Lynda et al. The link vs. the event: Activating and deactivating elements in time-based hypermedia. **New Review of Hypermedia and Multimedia**, Taylor & Francis, v. 6, n. 1, p. 89–109, 2000. DOI: 10.1080/13614560008914719.

HENNESSEY, Craig; FISET, Jacob. Long Range Eye Tracking: Bringing Eye Tracking into the Living Room. In: PROCEEDINGS of the Symposium on Eye Tracking Research and Applications. Santa Barbara, California: Association for Computing Machinery, 2012. (ETRA '12), p. 249–252. ISBN 9781450312219. DOI: 10.1145/2168556.2168608.

HENNESSEY, Craig; NOUREDDIN, Borna; LAWRENCE, Peter. A Single Camera Eye-Gaze Tracking System with Free Head Motion. In: PROCEEDINGS of the 2006 Symposium on Eye Tracking Research amp; Applications. San Diego, California: Association for Computing Machinery, 2006. (ETRA '06), p. 87–94. ISBN 1595933050. DOI: 10.1145/1117309.1117349.

HU, Paul J. et al. Examining the Technology Acceptance Model Using Physician Acceptance of Telemedicine Technology. **Journal of Management Information Systems**, Routledge, v. 16, n. 2, p. 91–112, 1999. DOI: 10.1080/07421222.1999.11518247.

IERUSALIMSCHY, Roberto; FIGUEIREDO, Luiz Henrique de; FILHO, Waldemar Celes. Lua — An Extensible Extension Language. **Software: Practice and Experience**, v. 26, n. 6, p. 635–652, 1996. DOI: https://doi.org/10.1002/(SICI)1097-024X(199606)26:6<635::AID-SPE26>3.0.CO;2-P.

ITU-RECOMMENDATION. Nested Context Language (NCL) and Ginga-NCL, nov. 2014.

_____. Nested Context Language (NCL) and Ginga-NCL for IPTV Services, abr. 2009.

IVANOV PEREIRA JOSUÉ, Marina. **Preparação de Objetos de Mídia e Efeitos Sensoriais para Formatação de Documentos Mulsemídia**. 2021. Tese (Doutorado) – Instituto de Computação, UFF, Niterói, RJ, Brasil.

JACOB, Robert J. K. What You Look at is What You Get: Eye Movement-Based Interaction Techniques. In: PROCEEDINGS of the SIGCHI Conference on Human Factors in Computing Systems. Seattle, Washington, USA: Association for Computing Machinery, 1990. (CHI '90), p. 11–18. ISBN 0201509326. DOI: 10.1145/97243.97246.

JOSUÉ, Marina; ABREU, Raphael et al. Modeling Sensory Effects as First-Class Entities in Multimedia Applications. In: PROCEEDINGS of the 9th ACM Multimedia Systems Conference. Amsterdam, Netherlands: Association for Computing Machinery, 2018. (MMSys '18), p. 225–236. ISBN 9781450351928. DOI: 10.1145/3204949.3204967. Disponível em: <https://doi.org/10.1145/3204949.3204967>.

JOSUÉ, Marina; MUCHALUAT-SAADE, Débora; MORENO, Marcelo. Preparation of Media Object Presentation and Sensory Effect Rendering in Mulsemedia Applications. In: PROCEEDINGS of the 24th Brazilian Symposium on Multimedia and the Web. Salvador, BA, Brazil: Association for Computing Machinery, 2018. (WebMedia '18), p. 45–52. ISBN 9781450358675. DOI: 10.1145/3243082.3243098. Disponível em: <https://doi.org/10.1145/3243082.3243098>.

KURAUCHI, Andrew et al. EyeSwipe: Dwell-Free Text Entry Using Gaze Paths. In: PROCEEDINGS of the 2016 CHI Conference on Human Factors in Computing Systems. San Jose, California, USA: Association for Computing Machinery, 2016. (CHI '16), p. 1952–1956. ISBN 9781450333627. DOI: 10.1145/2858036.2858335.

LAUGWITZ, Bettina; HELD, Theo; SCHREPP, Martin. Construction and Evaluation of a User Experience Questionnaire. In: v. 5298, p. 63–76. ISBN 978-3-540-89349-3. DOI: 10.1007/978-3-540-89350-9_6.

LIKERT, Rensis. A technique for the Measurement of Attitudes. **Archives of psychology**, v. 22 140, p. 55–55, 1932.

MARDANBEGI, Diako; HANSEN, Dan Witzner. Mobile Gaze-Based Screen Interaction in 3D Environments. In: (NGCA '11). ISBN 9781450306805. DOI: 10.1145/1983302.1983304.

MATTOS, Douglas Paulo de. **An Approach for Authoring Mulsemedia Applications Based on Events**. 2021. Tese (Doutorado) – Computing Institute, Fluminense Federal University, Niterói, RJ, Brazil.

MENTO, Mark A. **Different Kinds of Eye Tracking Devices**. [S. l.: s. n.], jun. 2020. Bitbrain. Acessado em: 24 de março de 2022. Disponível em: <https://www.bitbrain.com/blog/eye-tracking-devices>.

MONTEVECCHI, Eyre Brasil B. et al. Providing Eye Gaze Interaction for Ginga-NCL Applications. In: PROCEEDINGS of the Brazilian Symposium on Multimedia and the Web. São Luis, Brazil: Association for Computing Machinery, 2020. (WebMedia '20), p. 297–303. ISBN 9781450381963. DOI: 10.1145/3428658.3430983. Disponível em: <https://doi.org/10.1145/3428658.3430983>.

MUCHALUAT-SAADE, Débora Christina. **Relations in Hypermedia Authoring Languages: Improving Reuse and Expressiveness**. 2003. Tese (Doutorado) – PhD Thesis, Informatics Department, PUC-Rio, Rio de Janeiro, Brasil.

NIELSEN, Jakob; MOLICH, Rolf. Heuristic Evaluation of User Interfaces. In: PROCEEDINGS of the SIGCHI Conference on Human Factors in Computing Systems. Seattle, Washington, USA: Association for Computing Machinery, 1990. (CHI '90), p. 249–256. ISBN 0201509326. DOI: 10.1145/97243.97281.

PAPOUTSAKI, Alexandra; LASKEY, James; HUANG, Jeff. SearchGazer: Webcam Eye Tracking for Remote Studies of Web Search. In: ACM. PROCEEDINGS of the ACM SIGIR Conference on Human Information Interaction & Retrieval (CHIIR). [S. l.: s. n.], 2017.

PAPOUTSAKI, Alexandra; SANGKLOY, Patsorn et al. WebGazer: Scalable Webcam Eye Tracking Using User Interactions. In: AAAI. PROCEEDINGS of the 25th International Joint Conference on Artificial Intelligence (IJCAI). [S. l.: s. n.], 2016. p. 3839–3845.

PEDROSA, Diogo; PIMENTEL, Maria da Graça; TRUONG, Khai N. Filteryedping: A Dwell-Free Eye Typing Technique. In: PROCEEDINGS of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems. Seoul, Republic of Korea: Association for Computing Machinery, 2015. (CHI EA '15), p. 303–306. ISBN 9781450331463. DOI: 10.1145/2702613.2725458. Disponível em: <https://doi.org/10.1145/2702613.2725458>.

RAMOS, Guilherme Matheus M.A.; SALES, Victor O.N.; TEIXEIRA, Cesar A.C. LETRAS: Communication with Lines, Triangles and Rectangles. In: PROCEEDINGS of the 22nd Brazilian Symposium on Multimedia and the Web. Teresina, Piaui State, Brazil: Association for Computing Machinery, 2016. (Webmedia '16), p. 215–218. ISBN 9781450345125. DOI: 10.1145/2976796.2988175.

RANDELL, David A; CUI, Zhan; COHN, Anthony G. A spatial logic based on regions and connection. **KR**, v. 92, p. 165–176, 1992.

RODRIGUES, Rogério Ferreira. **Formatação e Controle de Apresentações Hipermıdia com Mecanismos de Adaptação Temporal**. 2003. Tese (Doutorado) – Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brasil.

SCHREPP, Martin; HINDERKS, Andreas; THOMASCHEWSKI, Jörg. Construction of a Benchmark for the User Experience Questionnaire (UEQ). **International Journal of Interactive Multimedia and Artificial Intelligence**, v. 4, p. 40–44, jun. 2017. DOI: 10.9781/ijimai.2017.445.

SILVA, Ellen P.; AMORIM, Glauco F.; SANTOS, Joel A. F. dos. Adding Temporal Semantic to Textual Media Objects with Eye Tracking Technology. In: PROCEEDINGS of the 25th Brazillian Symposium on Multimedia and the Web. Rio de Janeiro, Brazil: Association for Computing Machinery, 2019. (WebMedia '19), p. 217–220. ISBN 9781450367639. DOI: 10.1145/3323503.3360636. Disponível em: <https://doi.org/10.1145/3323503.3360636>.

SOARES, Luiz Fernando G; RODRIGUES, Rogério F; MUCHALUAT SAADE, Débora C. Modeling, Authoring and Formatting Hypermedia Documents in the HyperProp System. **Multimedia systems**, Springer, v. 8, n. 2, p. 118–134, 2000. DOI: 10.1007/s005300050155.

SOARES, Luiz Fernando Gomes; BARBOSA, Simone Diniz Junqueira. **Programando em NCL 3.0: Desenvolvimento de aplicações para o middleware Ginga 2a. Edição Versão 2.1**. [S. l.]: Elsevier Campos, 2011.

SOARES, Luiz Fernando Gomes; RODRIGUES, Rogério Ferreira. Nested context model 3.0: Part 1–ncm core. **Relatório Técnico de Pesquisa da série de Monografias do Departamento de Informática da PUC-Rio**, v. 12, 2005.

SUNDSTEDT, Veronica. Gazing at Games: Using Eye Tracking to Control Virtual Characters. In: ACM SIGGRAPH 2010 Courses. Los Angeles, California: Association for Computing Machinery, 2010. (SIGGRAPH '10). ISBN 9781450303958. DOI: 10.1145/1837101.1837106.

TANRIVERDI, Vildan; JACOB, Robert J. K. Interacting with Eye Movements in Virtual Environments. In: PROCEEDINGS of the SIGCHI Conference on Human Factors in Computing Systems. The Hague, The Netherlands: Association for Computing Machinery, 2000. (CHI '00), p. 265–272. ISBN 1581132166. DOI: 10.1145/332040.332443.

TOBII. **What is Eye Tracking?** [S. l.: s. n.], 2020. Acessado em: 21 de maio de 2020. Disponível em: <https://www.tobii.com/group/about/this-is-eye-tracking>.

VALENTIM, Pedro Alves; BARRETO, Fábio; MUCHALUAT-SAADE, Débora C. Possibilitando o Reconhecimento de Expressões Faciais em Aplicações Ginga-NCL. In: ANAIS Estendidos do XXVI Simpósio Brasileiro de Sistemas Multimídia e Web. São Luís: SBC, 2020. p. 53–56. DOI: 10.5753/webmedia_estendido.2020.13062. Disponível em: <https://sol.sbc.org.br/index.php/webmedia_estendido/article/view/13062>.

W3C. **Extensible Markup Language (XML) 1.0 (Fifth Edition)**. [S. l.: s. n.], 2008. World-Wide Web Consortium Recommendation.

_____. **HTML5: A vocabulary and associated APIs for HTML and XHTML**. [S. l.: s. n.], 2014. https://www.w3.org/TR/html5/. World-Wide Web Consortium Recommendation.

_____. **Synchronized Multimedia Integration Language - SMIL 3.0 Specification**. [S. l.: s. n.], 2008. http://www.w3c.org/TR/SMIL3. World-Wide Web Consortium Recommendation.

ZHANG, Yanxia et al. Eye tracking for public displays in the wild. **Personal and Ubiquitous Computing**, Springer, v. 19, n. 5, p. 967–981, 2015. DOI: 10.1007/s00779-015-0866-8.

APÊNDICE A – QUESTIONÁRIO: AVALIAÇÃO DO EYE GAZE NO GINGA-NCL

Avaliação do Eye Gaze no Ginga-NCL

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

UFF – Universidade Federal Fluminense / IC - Instituto de Computação

O senhor (sra.) está sendo convidado a participar um estudo para avaliação da modalidade de interação usando a fixação ocular (Eye Gaze) no Ginga-NCL, ou seja, no contexto da TV digital. O Ginga-NCL é um subsistema do middleware Ginga, padrão do Sistema Brasileiro de TV digital, que permite a execução de aplicações multimídia especificadas em linguagem NCL. Trata-se de uma pesquisa realizada pela aluna de mestrado Eyre Brasil Barbosa Montevecchi, orientada pela professora Dra. Débora Christina Muchaluat-Saade.

Para a realização do estudo é necessário que responda a cada uma das seções deste questionário e aceite eletronicamente participar da pesquisa como voluntário. Os dados respondidos serão sigilosos e o senhor (sra.) pode se retirar do estudo a qualquer momento.

Agradecemos pela participação e colaboração.

eyre@montevecchi.com.br (não compartilhado)
Alternar conta

Rascunho restaurado.

*Obrigatório

*

Eu aceito participar do estudo.

Avaliação do Eye Gaze no Ginga-NCL

eyre@montevecchi.com.br (não compartilhado) Alternar conta

*Obrigatório

Questionário TAM (Modelo de Aceitação de Tecnologia)

Utilidade percebida (PU)

PU1: Usar o Eye Gaze para interagir com aplicações de TV digital me permitiria interagir mais rapidamente. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PU2: Usar o Eye Gaze melhoraria meu desempenho para interagir com aplicações de TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PU3: Usar o Eye Gaze poderia tornar minha interação com aplicações de TV digital mais eficaz. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PU4: Usar o Eye Gaze facilitaria minha interação com aplicações de TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PU5: O Eye Gaze seria útil para eu interagir com aplicações de TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PU6: O Eye Gaze me ajudaria a interagir com aplicações de TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

Percepção da facilidade de uso (PEOU)

PEOU1: Aprender a usar o Eye Gaze para interação com aplicações de TV digital seria fácil para mim. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PEOU2: Seria fácil usar o Eye Gaze para interagir com aplicações de TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PEOU3: Eu consideraria o Eye Gaze flexível para interagir com aplicações de TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PEOU4: Seria fácil para eu tornar-me hábil no uso do Eye Gaze para interagir com aplicações de TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PEOU5: Usar o Eye Gaze exigiria pouco esforço físico para interagir com a TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

PEOU6: Usar o Eye Gaze exigiria pouco esforço mental para interagir com a TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

Questionário de Experiência do Usuário (UEQ)

Para avaliar a modalidade de interação com os olhos utilizada, por favor preencha o seguinte questionário. É constituído por pares de opostos relativos às propriedades que a interação sendo avaliada possa ter.
As graduações entre os opostos são representadas por círculos. Ao marcar um dos círculos, você pode expressar sua opinião sobre um conceito.
Marque a sua resposta o mais espontaneamente possível. É importante que não pense em demasia na resposta porque a sua avaliação imediata é que é importante.
Por favor, assinale sempre uma resposta, mesmo que não tenha certezas sobre um par de termos ou que os termos não se enquadrem. Não há respostas "certas" ou respostas "erradas". A sua opinião é que conta!

*

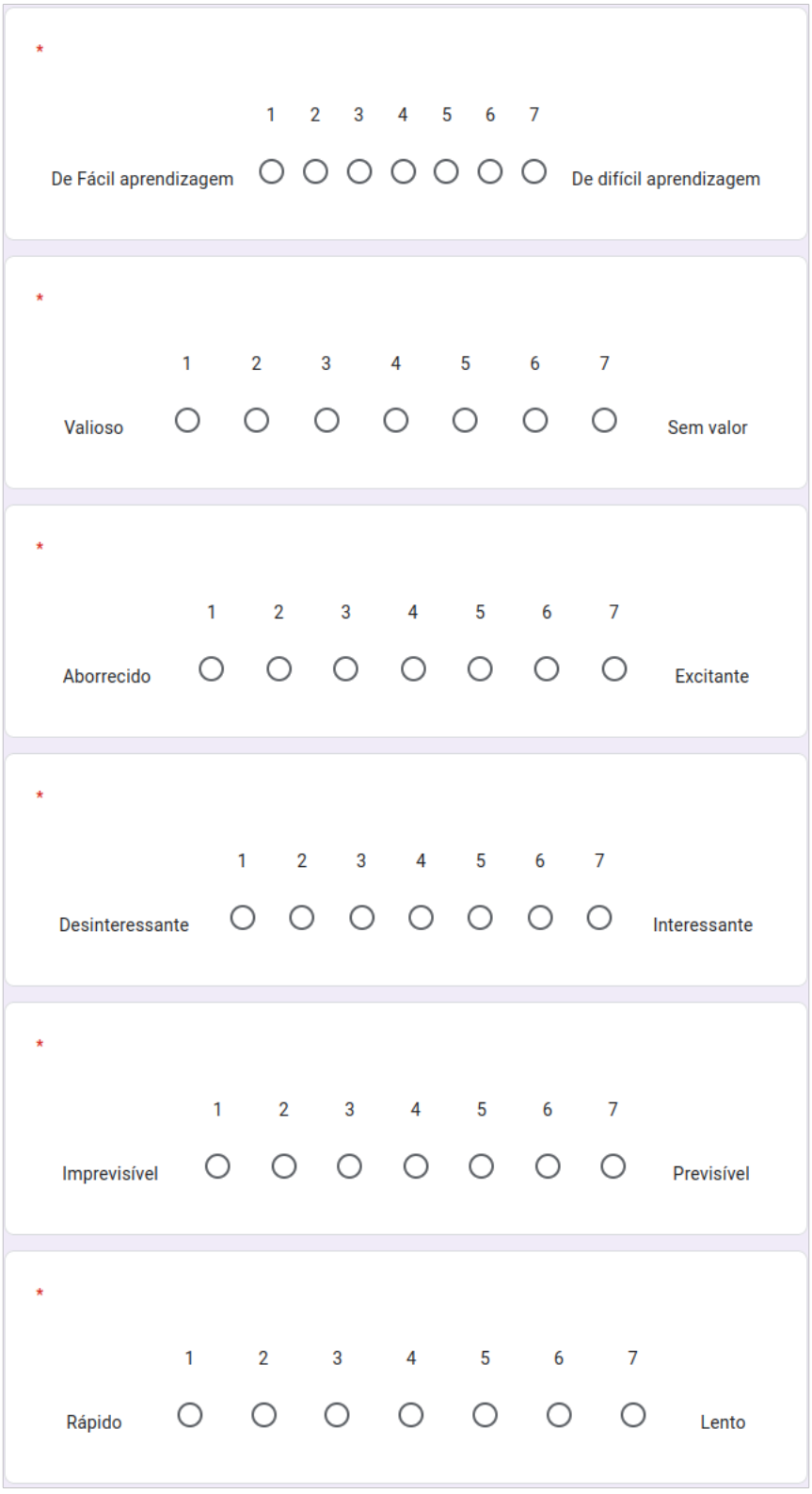
	1	2	3	4	5	6	7	
Desagradável	◯	◯	◯	◯	◯	◯	◯	Agradável

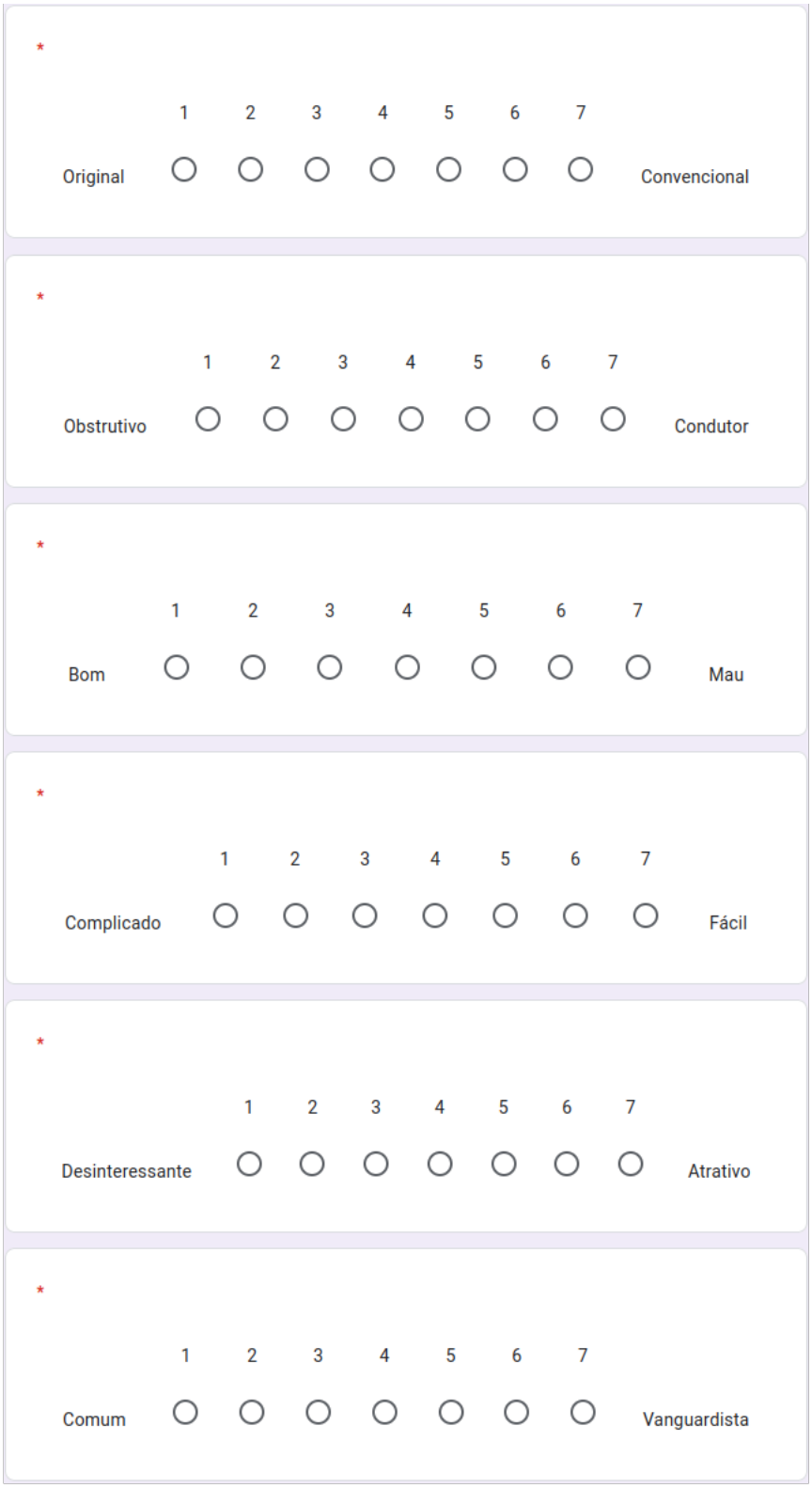
*

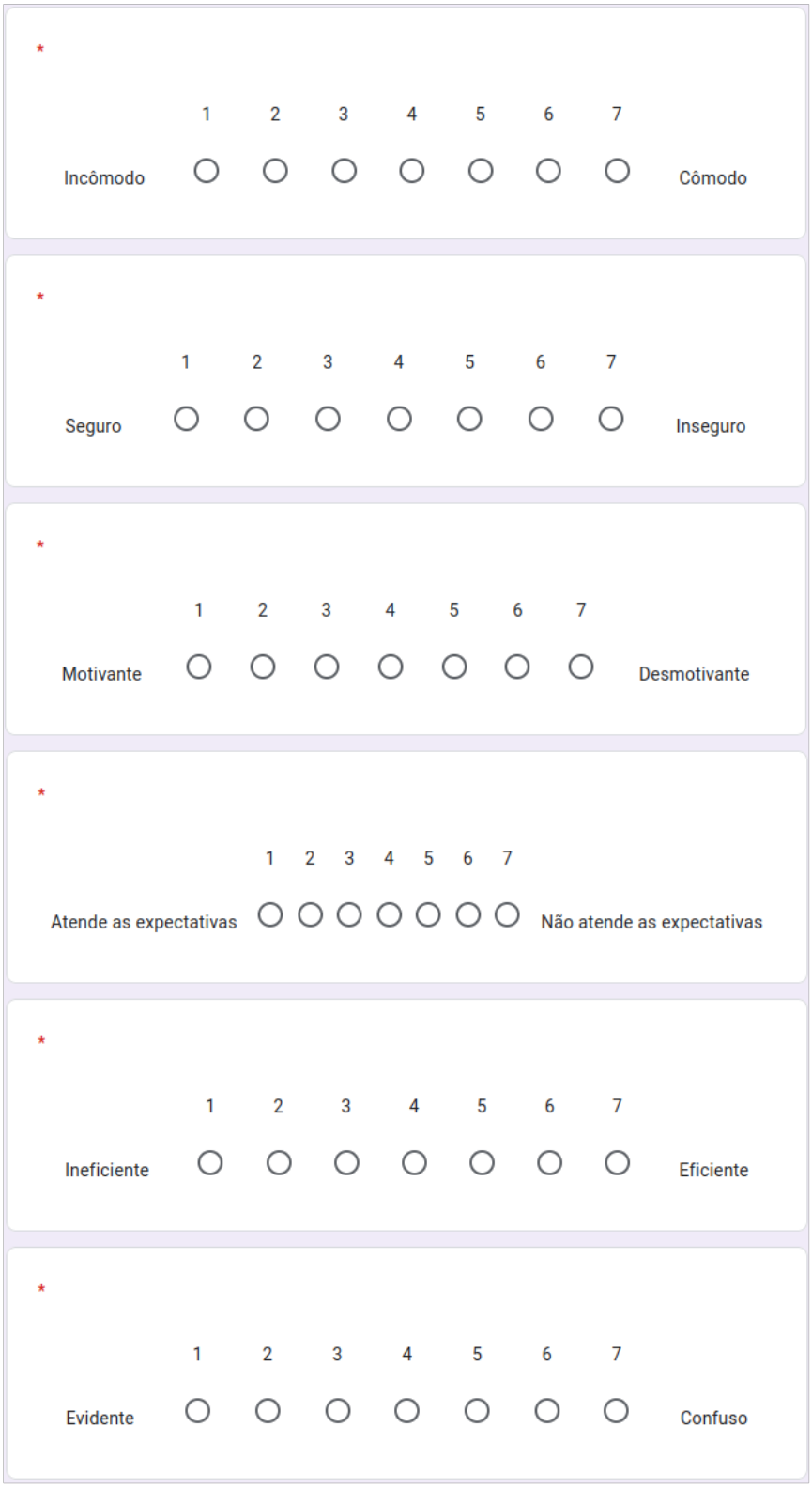
	1	2	3	4	5	6	7	
Incompreensível	◯	◯	◯	◯	◯	◯	◯	Compreensível

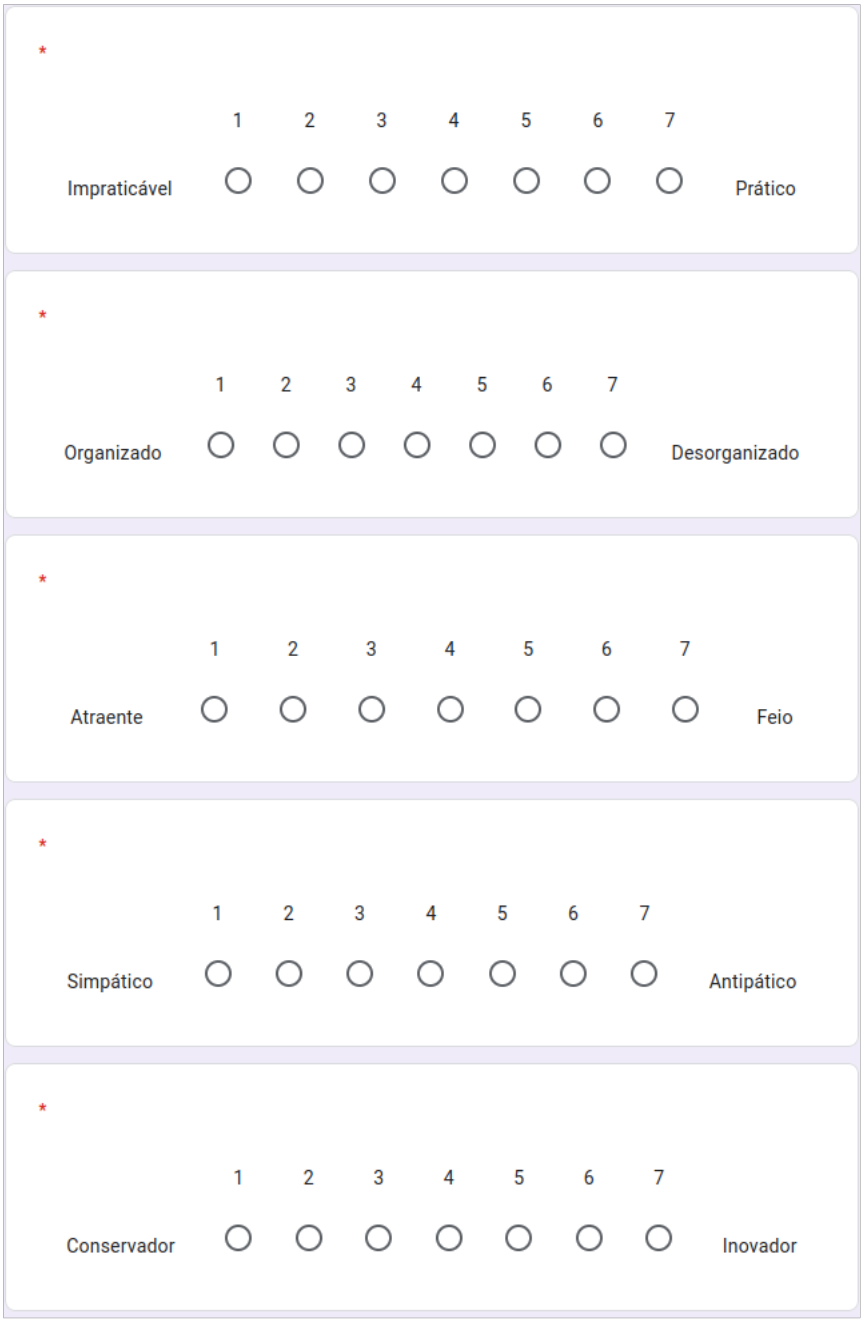
*

	1	2	3	4	5	6	7	
Criativo	◯	◯	◯	◯	◯	◯	◯	Sem criatividade


*
1 2 3 4 5 6 7
De Fácil aprendizagem
De difícil aprendizagem
*
1 2 3 4 5 6 7
Valioso
Sem valor
*
1 2 3 4 5 6 7
Aborrecido
Excitante
*
1 2 3 4 5 6 7
Desinteressante
Interessante
*
1 2 3 4 5 6 7
Imprevisível
Previsível
*
1 2 3 4 5 6 7
Rápido
Lento


*
1 2 3 4 5 6 7
Original
Convencional
*
1 2 3 4 5 6 7
Obstrutivo
Condutor
*
1 2 3 4 5 6 7
Bom
Mau
*
1 2 3 4 5 6 7
Complicado
Fácil
*
1 2 3 4 5 6 7
Desinteressante
Atrativo
*
1 2 3 4 5 6 7
Comum
Vanguardista


*
1 2 3 4 5 6 7
Incômodo
Cômodo
*
1 2 3 4 5 6 7
Seguro
Inseguro
*
1 2 3 4 5 6 7
Motivante
Desmotivante
*
1 2 3 4 5 6 7
Atende as expectativas
Não atende as expectativas
*
1 2 3 4 5 6 7
Ineficiente
Eficiente
*
1 2 3 4 5 6 7
Evidente
Confuso


*
1 2 3 4 5 6 7
Impraticável
Prático
*
1 2 3 4 5 6 7
Organizado
Desorganizado
*
1 2 3 4 5 6 7
Atraente
Feio
*
1 2 3 4 5 6 7
Simpático
Antipático
*
1 2 3 4 5 6 7
Conservador
Inovador

Questionário de Experiência Geral

Já utilizou dispositivos rastreadores oculares (eye-trackers) antes? *

- ◯ Sim
- ◯ Não

Gostei de interagir usando meus olhos. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

SENTI FADIGA nos ombros ao usar o Eye Gaze para interagir. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

SENTI FADIGA no pescoço ao usar o Eye Gaze para interagir. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

SENTI FADIGA nos olhos ao usar o Eye Gaze para interagir. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

O Eye Gaze tem boa precisão durante a interação com a TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

O Eye Gaze tem boa velocidade durante a interação com a TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

É confortável usar o Eye Gaze para interagir com a TV digital. *

- ◯ Discordo totalmente
- ◯ Discordo parcialmente
- ◯ Não concordo nem discordo
- ◯ Concordo parcialmente
- ◯ Concordo totalmente

Dados do Participante

Grau de escolaridade: *

- ◯ Fundamental incompleto
- ◯ Fundamental completo
- ◯ Médio incompleto
- ◯ Médio completo
- ◯ Superior completo
- ◯ Pós-graduação

Gênero:

- ◯ Feminino
- ◯ Masculino
- ◯ Outro

Idade:

Sua resposta

Possui alguma dificuldade de movimento dos olhos? *

- ◯ Sim
- ◯ Não

Possui alguma dificuldade de movimento dos membros superiores (braços e mãos)? *

- ◯ Sim
- ◯ Não

Deseja realizar alguma observação ou sugestão sobre a sua experiência com o Eye Gaze?

Sua resposta