ESTUDO DE TECNOLOGIAS ASSISTIVAS PARA DEFICIENTES VISUAIS APLICADAS A PLATAFORMA FURBOT

Caroline Batistel, Dalton Solano Reis – Orientador

Curso de Bacharel em Ciência da Computação
Departamento de Sistemas e Computação
Universidade Regional de Blumenau (FURB) – Blumenau, SC – Brasil

cbatistel@furb.br, dalton@furb.br

Resumo: *O seguinte artigo apresenta o desenvolvimento de um protótipo da plataforma Furbot que visa criar acessibilidade para pessoas cegas ou com baixo grau de visão. O protótipo desenvolvido utiliza o RT-Voice Pro, uma ferramenta text-to-speech, para situar os usuários, e uma série de alterações e novas funções para tornar o nível de dificuldade no jogo o mesmo para todos. O desenvolvimento foi realizado em Unity, utilizando como base a versão 3.0 da plataforma Furbot. Foram realizados testes com cinco usuários, sendo um deles com baixa visão. No geral os usuários testados aprovaram o protótipo desenvolvido e acharam as novas funções uteis. O objetivo do protótipo foi alcançado, pois foi possível gerar um ambiente acessível sem perder a essência da plataforma.*

Palavras-chave: *Deficientes visuais. Acessibilidade. Text-to-speech. RT-Voice Pro. Unity.*

1 INTRODUÇÃO

A inclusão social tem assumido grande importância nos dias de hoje, e com a popularização de ferramentas computacionais e da Internet, também é necessário focar na inclusão digital. Segundo dados do Censo demográfico de 2010, existem 6,5 milhões de pessoas no Brasil que possuem deficiências visuais, sendo 582 mil cegas e 6 milhões com baixa visão (IBGE, 2011).

Muitos métodos foram criados para auxiliar pessoas não-videntes (que não enxergam) em seu cotidiano, entre eles se encontram as tecnologias assistivas. Tais tecnologias, podem ser descritas como uma série de equipamentos, estratégias, práticas e serviços concebidos para minimizar problemas funcionais enfrentados por pessoas portadoras de necessidades especiais (Cook; Polgar, 2014).

Grande parte dos dados passados diariamente às pessoas, seja qual for o ambiente, se dá através de imagens e apelos visuais, o que cria barreiras para pessoas cegas ou de baixa visão (Nunes; Machado; Vazin, 2011). Para proporcionar a essas pessoas o devido acesso aos conteúdos visuais, foi desenvolvida uma tecnologia assistiva chamada audiodescrição. Sendo um recurso amplamente utilizado, a audiodescrição tenta traduzir em palavras todo o conteúdo que pessoas não-videntes não tem acesso por sua condição. Em meios digitais a audiodescrição se adaptou a leitores de tela, que permitem a pessoas não videntes a navegação na internet, e a utilização com autonomia de celulares e computadores. Além da audiodescrição, existem diversas outras tecnologias assistivas direcionadas a pessoas não-videntes, aplicativos que utilizam reconhecimento de imagem, o sistema de escrita em braille, entre outros.

Pensando neste meio de inclusão social e digital, chega-se à plataforma Furbot, criada na Universidade Regional de Blumenau (FURB) em um projeto que, segundo Mattos *et. al.* (2019a, p. 1), "busca promover a inclusão digital cidadã por meio de oficinas de programação que permitam o desenvolvimento de aptidões em pensamento computacional [...]".

Tendo em vista este cenário, este trabalho se propôs a estudar tecnologias assistivas para pessoas cegas e de baixa visão, e viabilizar um protótipo da plataforma Furbot com acessibilidade para tais pessoas, de modo que o uso da plataforma possa ser feito tanto por pessoas videntes quanto não videntes igualmente. Onde o objetivo deste trabalho é viabilizar um protótipo da plataforma Furbot com acessibilidade para pessoas cegas ou com baixo nível de visão, assim podendo fazer o uso da plataforma. Os objetivos específicos são:

a) disponibilizar um módulo de audiodescrição integrado ao Furbot, que permita a pessoa se localizar, sem interferir na forma de encontrar a solução das atividades apresentadas;
b) criar estratégias para auxiliar os usuários a se localizarem e atravessarem uma fase de teste.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo traz informações sobre os aspectos da fundamentação teórica utilizados para a construção do projeto. Na primeira seção deste capítulo serão apresentados os conceitos utilizados como base para o desenvolvimento, a segunda seção apresenta três trabalhos correlatos com o trabalho realizado neste artigo.

2.1 ACESSIBILIDADE DIGITAL

O conceito de acessibilidade nasceu ligado a assuntos físicos, como facilidade de acesso a locais, e começou a ser difundido no mundo por volta de 1981, declarado pela Organização das Nações Unidas (ONU) como o "Ano Internacional dos Portadores de Deficiência". A acessibilidade é classificada por permitir acesso a todo e qualquer espaço, físico ou não, deixando viável a entrada de diferentes tipos de pessoas, com ou sem necessidades especiais aos locais que elas frequentam, garantindo-lhes qualidade de vida através da Lei nº. 10.098/2000 (BRASIL, 2000).

Para Granollers (2004), acessibilidade digital significa propiciar a flexibilidade para adaptações decorrentes das necessidades de cada usuário segundo suas preferências e/ou limitações. Segundo Dias (2007), a acessibilidade é a propriedade de um produto que permite atender pessoas, sendo compatível com tecnologias assistivas. Portanto, um software é tido como acessível quando qualquer pessoa, portadora de deficiência ou não, consegue executar as mesmas funções, e alcança os mesmos resultados ao utilizá-lo.

Passerino *et. al.* (2007) afirma que a acessibilidade digital só pode ser proporcionada mediante uma combinação entre hardware e software, oferecendo os mecanismos físicos para superar barreiras de percepção e o acesso a funções e informações. Passerino *et. al.* (2007) acredita que é possível promover a inclusão digital através de três grandes áreas, sendo elas as tecnologias assistivas, incluir portadores de necessidades (PNE) especiais na concepção dos softwares e dar mais acesso à internet para PNE, flexibilizando a informação e as interações.

2.2 TECNOLOGIAS ASSISTIVAS – AUDIODESCRIÇÃO

O termo tecnologia assistiva foi proposto por Sassaki (1997) no Brasil, como "coisa que assiste ou coisa que ajuda, que auxilia". Em 16 de novembro de 2006 foi instituído no Brasil, pela Portaria nº 142 (BRASIL, 2006), o Comitê de Ajudas Técnicas (CAT) apresentou o conceito de Tecnologia Assistiva como uma área do conhecimento que engloba produtos, recursos, metodologias, estratégias, práticas e serviços. A apresentação de tal conceito "visa promover a autonomia e independência de pessoas portadoras de deficiência, gerando assim qualidade de vida e inclusão social" (BRASIL, 2009).

Para Cook e Polgar (2014), as tecnologias assistivas podem ser expostas como uma série de equipamentos, estratégias, práticas e serviços concebidos para tornarem mínimos os problemas funcionais enfrentados por pessoas portadoras de necessidades especiais. Nunes *et. al.* (2011) explica que grande parte dos dados passados diariamente às pessoas, seja qual for o ambiente, se dá através de imagens e apelos visuais, o que cria barreiras para pessoas cegas ou de baixa visão, por este motivo tecnologias assistivas para cegos são muito necessárias no acesso à informação.

Neste ponto, a audiodescrição segundo Franco *et. al.* (2010), permite que pessoas cegas tenham o acesso aos conteúdos visuais em qualquer tipo de mídia, por se tratar de uma tradução em palavras de toda informação relevante para o entendimento de uma mensagem apresentada de forma visual. Diferindo de outras tecnologias assistivas, a audiodescrição não é um meio que possa ser obtido isoladamente, sendo utilizado apenas quando o usuário desejar, tratando-se de um recurso disponibilizado junto com os produtos.

2.2.1 RTVoice Pro

Em meios digitais a audiodescrição geralmente vem através de leitores de tela, ferramentas também conhecidas como *text-to-speech* (TTS) que traduzem texto em fala. Existem diversas interfaces de programação (APIs) que possibilitam criação de conteúdo digital integrado com TTS. Um exemplo dessas APIs é o `RTVoice Pro`, desenvolvido pela Crosstales (2020) que utiliza vozes TTS já integradas no sistema ou plataforma para pronunciar qualquer texto podendo ser utilizado em tempo de execução.

Segundo a Crosstales (2020), o `RTVoice` conta com compatibilidade em várias plataformas, integração com ferramentas de TTS online, configurações de entonação, velocidade e volume de fala fornecendo ao usuário uma maior flexibilidade para atender suas necessidades e preferências. A ferramenta também permite alterações como sussurros, implementação de sentimento ou sensação, como alegria, tristeza, tédio, entre outras.

2.3 JOGOS EDUCACIONAIS

Para Savi e Ulbricht (2008), jogos desenvolvidos para fins educacionais podem ser denominados como jogos educativos ou educacionais, jogo de aprendizagem ou ainda jogos sérios. Segundo Gros (2003), jogos digitais são uma das principais formas de acesso ao mundo da tecnologia para crianças e jovens, afinal geralmente o primeiro contato com equipamentos eletrônicos é por meio de um vídeo game.

Prieto *et. al.* (2005) destaca que para serem utilizados como métodos educacionais, qualquer software seja ele jogo ou não, deve conter objetivos pedagógicos e a sua utilização deve estar num contexto e situação de ensino baseados em uma metodologia que oriente o processo, utilizando-se da interação, motivação e descoberta para facilitar o aprendizado de um determinado conteúdo.

2.3.1 Furbot

Entre os diversos jogos educacionais existentes, Mattos *et. al.* (2019a) descreve o Furbot como um projeto que atua no desenvolvimento cognitivo infantil através de atividades de programação de computadores, utilizando jogos de estratégia para simplificar o aprendizado da programação e desenvolver o raciocínio lógico, e a capacidade de resolução de problemas, colocando o pensamento computacional em ação.

> "A ferramenta apresenta um robô denominado Furbot como personagem principal da história o qual possui missões diferenciadas durante cada cena do jogo. O jogo possui um enredo para contextualizar o jogador. O enredo conta que a Terra foi invadida por alienígenas sendo que estes deixaram pistas no planeta. A missão do robô é salvar o planeta da invasão alienígena, coletando as pistas deixadas por eles pelo caminho." (MATTOS *et. al.*, 2019b, p. 1260)

Segundo Mattos *et. al.* (2019b) o jogo se passa em um cenário composto inicialmente pelo robô, um drone (que no jogo é chamado de S-223) e um caminho a ser percorrido. No decorrer das fases o jogador precisa programar o robô para percorrer o caminho, evitando obstáculos, como pedras, árvores e arbustos, e capturando objetivos no mapa, que são distribuídos entre tesouros, vidas, energia, rastros dos alienígenas ente outros.

Durante as fases, Mattos *et. al.* (2019b) explica que o jogador deve programar o robô para deixar sua posição inicial no mapa e atingir a última posição do caminho, onde há uma marca deixada por um alienígena. O drone S-223 é uma ajudante que fornece algumas dicas durante todo o jogo.

Conforme Mattos *et. al.* (2019b), o jogo se dá em dois momentos, no primeiro momento o jogador cria um código fonte que é composto por comandos como `andar(DIRECAO);`, que permite que o robô se mova uma casa na direção informada, podendo ser direita, esquerda, abaixo ou acima. No segundo momento o jogador executa o código criado, podendo atingir o objetivo, ou não. Caso exista algum erro de compilação no código criado pelo jogador, a S-223 informa ao jogador que há algo errado no código.

Mattos *et. al.* (2019b) ilustra que quando o Furbot atinge seu objetivo final da fase, esta se encerra, mostrando a pontuação adquirida pelo jogador e o encaminhando para a próxima fase. O autor também afirma que as fases são constituídas por cenários que contem objetos a serem desviados e coletados, e o robô deve ser programado para realizar tais ações.

2.4 TRABALHOS CORRELATOS

Nesta seção serão apresentados três trabalhos correlatos que possuem características semelhantes a proposta deste trabalho. O Quadro 1 descreve o trabalho de Kraemer (2017), que desenvolveu uma aplicação multiplataforma para dispositivos móveis para garantir a acessibilidade de deficientes visuais a um jogo de cartas. No Quadro 2 o trabalho de Costa (2013) detalha a criação de um jogo de tiros em primeira pessoa (*First-Person Shooter* - FPS) para deficientes visuais, utilizando som 3D e sistemas hápticos. No Quadro 3 o trabalho de Sobral *et. al.* (2017) detalha o desenvolvimento de um jogo digital no estilo *Role Playing Game* (RPG) que auxilia no ensino de conceitos da área de Língua Portuguesa e Matemática, para crianças e adolescentes com deficiências visuais.

Quadro 1 – Tecnologia assistiva: Tornando jogo de mesa acessível para cegos com auxílio de aplicativo móvel de reconhecimento de imagem

Referência	Kraemer (2017)
Objetivos	Criar uma aplicação móvel multiplataforma para permitir que pessoas cegas possam participar de uma partida do jogo de cartas Munchkin
Principais funcionalidades	Conforme a Figura 1 (a) o usuário permite o uso do microfone e se comunica com o aplicativo através de comandos de voz, após ter a permissão concedida, o usuário pode tirar uma foto da carta que deseja reconhecer, como demonstrado na Figura 1 (b). Após ter tirado a foto, deve ser feita a confirmação de que a foto que foi tirada será realmente utilizada para o reconhecimento, caso queira o usuário pode optar por tirar uma nova foto. Depois do usuário realizar a confirmação, o sistema irá reconhecer a carta e sintetizar por voz a descrição e informações que foram interpretadas. Estas informações são apresentadas na Figura 1 (c) através do modo desenvolvedor do aplicativo. Depois de realizar este processo, o usuário pode escolher ouvir novamente a descrição da carta ou reconhecer outra carta.
Ferramentas de desenvolvimento	Ambiente de desenvolvimento Visual Studio Code, framework Ionic, Google Cloud Vision API, HTML5, CSS, Javascript e AngularJS
Resultados e conclusões	Nos testes individuais, o aplicativo demonstrou os melhores resultados em fotos com boa iluminação e utilizando uma rede wi-fi para o envio da foto para a API externa. Nos testes em grupo, ao aumentar os níveis de ruído do ambiente, a aplicação passou a distorcer comandos e eram necessárias novas tentativas. A utilização de fones de ouvido com microfone amenizou o problema do ruído

Fonte: elaborado pelo autor.

Elaborado por Kraemer (2017) o projeto apresenta o desenvolvimento de uma ferramenta de tecnologia assistiva para garantir acessibilidade de pessoas cegas e de baixa visão ao jogo de cartas Munchkin, utilizando um aplicativo multiplataforma para dispositivos móveis. O autor explica que é necessário fornecer ao aplicativo a permissão de acesso a câmera para que o Munchkin Recognizer funcione.

Figura 1 – Funcionamento do Munchkin Recognizer

(a) Tela do Munchkin Recognizer (b) Execução em um Moto G 2 com Android 6.0 (c) Resultado sendo exibido em modo desenvolvedor

Fonte: Kraemer (2017).

Em todo o processo, o Kraemer (2017) identifica algumas possibilidades de melhoria, como tornar a ferramenta off-line, implementar o reconhecimento em tempo real, eliminando a necessidade de tirar uma foto da carta. Bem como eliminar a necessidade de um botão para realizar os comandos de voz, e adaptar a ideia para outros tipos de jogos de mesa.

Quadro 2 – Blind Counter-Strike: um jogo FPS para deficientes visuais

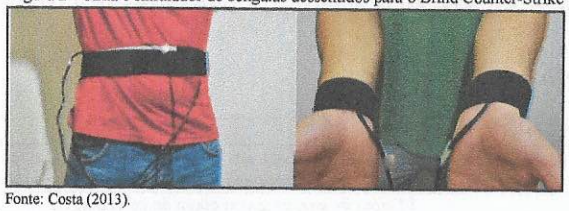
Referência	Costa (2013)
Objetivos	Criar um jogo estilo *fist person shooter* (FPS) acessível para pessoas cegas, utilizando técnicas para eliminar a necessidade de recursos visuais
Principais funcionalidades	O jogo conta com um menu sintetizado por voz que narra cada opção ao usuário, bem como, localiza o mesmo sobre em qual menu ele se encontra (menu inicial, tela de pause etc.). Além destas, Costa (2013) afirma que existem mais narrações no jogo com a intenção de contextualizar o jogador, como as informações sobre o objetivo do jogo, como jogar, quantidade de vida e de munição durante as fases, posição e direção do personagem no mapa, e uma lista com o significado de cada efeito sonoro do jogo. Existem respostas hápticas para localizar o jogador quanto a estar sendo atingido, estar alinhado a um inimigo entre outras situações e som 3D para ajudar o jogador a se localizar. A jogabilidade do Blind Counter-Strike, se dá através de um controle de Xbox360 (Microsoft), utilização de fones de ouvido para ter acesso ao som 3D em que o jogo se baseia e dos acessórios desenvolvidos por Costa (2013). Tais acessórios são mostrados na Figura 2, sendo eles: uma cinta com vibradores que indica quando o jogador está recebendo tiros e um simulador de bengalas, que consistem em duas munhequeiras também com vibradores
Ferramentas de desenvolvimento	Foi desenvolvido para a plataforma Windows 7 utilizando a *engine* gráfica XNA juntamente com a linguagem de programação C#, utilizando o ambiente Visual Studio. A comunicação com os hardwares se deu através do microcontrolador Arduino
Resultados e conclusões	Na fase de testes o jogo se saiu bem, porém Costa (2013) identificou que no início do jogo muitas informações eram dadas em um curto período de tempo dificultando a assimilação do jogador, as sintetizações por voz de informações triviais foram provadas importantes, porém estas devem ser dosadas corretamente. As respostas hápticas foram pouco mencionadas nos testes, porém no que foi mencionado estas não tiveram um resultado tão bom por conta de explicações confusas e do excesso de informações ao início do jogo.

Fonte: elaborado pelo autor.

Segundo Costa (2013), o objetivo do jogo Blind Counter-Strike é que o jogador passe por cinco fases sozinho, encontrando e matando inimigos em diferentes níveis de dificuldade. O jogador conta com duas armas, uma mais fraca, porém com mais munição disponível e outra mais forte, mas sem tanta munição. Mesmo tendo sido criado para pessoas

cegas ou de baixa visão, Costa (2013) desenvolveu uma parte gráfica para o jogo, principalmente para questões de depuração durante o desenvolvimento.

Figura 2 - Cinta e simulador de bengalas dessentidos para o Blind Counter-Strike



Fonte: Costa (2013).

Quadro 3 - A utilização de *role playing* games digitais como ferramenta complementar no processo de aprendizagem de crianças deficientes visuais

Referência	Sobral *et. al.* (2017)
Objetivos	Entender como ocorre o processo de interação de deficientes visuais com sistemas computacionais na aprendizagem e identificar recursos de entretenimento para esses indivíduos, e com base no conhecimento adquirido criar um jogo.
Principais funcionalidades	O jogo é um áudio game no formato *role playing* game (RPG), também conhecido como jogo de interpretação, em que o jogador assume o papel do protagonista e passa por uma história sendo que a dificuldade aumenta gradativamente junto com o nível do jogador.
Ferramentas de desenvolvimento	Foi utilizada a linguagem de programação Java, outras ferramentas não foram identificadas no artigo.
Resultados e conclusões	Os autores verificaram que os alunos com deficiência visual se interessaram pelo jogo, porém foram relatadas dificuldades de compreensão na narração da história por conta da velocidade da fala.

Fonte: elaborado pelo autor.

Sobral *et. al.* (2017) indica que a princípio foi feita uma coleta de dados, através de entrevistas na Secretaria da Educação e em instituições de atendimento a deficiente visuais. Partindo destas informações foi desenvolvido o áudio game A Cidade de Ominicron. O jogo ocorre inteiramente na tela apresentada na Figura 3, e toda a ambientação ocorre através do som. A interface desenvolvida por Sobral *et. al.* (2017) é baseada no uso de efeitos sonoros e narrações instrutivas, que levam os jogadores a fazerem suas escolhas, caso os jogadores errem muitas vezes eles recebem dicas de como prosseguir.

Figura 3 - Tela do jogo

Fonte: Sobral *et. al.* (2017).

3 DESCRIÇÃO DO PROTÓTIPO

Este capítulo pretende apresentar os detalhes de especificação e implementação do protótipo, para isso são apresentadas três seções. A primeira seção apresenta a visão geral do protótipo, de forma a ambientar o leitor quanto ao funcionamento das novas funcionalidades implementadas na plataforma Furbot. A segunda seção apresenta aspectos da implementação do protótipo. A terceira seção demonstra a atual operacionalidade do protótipo, por meio de comparações com a sua versão anterior.

3.1 VISÃO GERAL

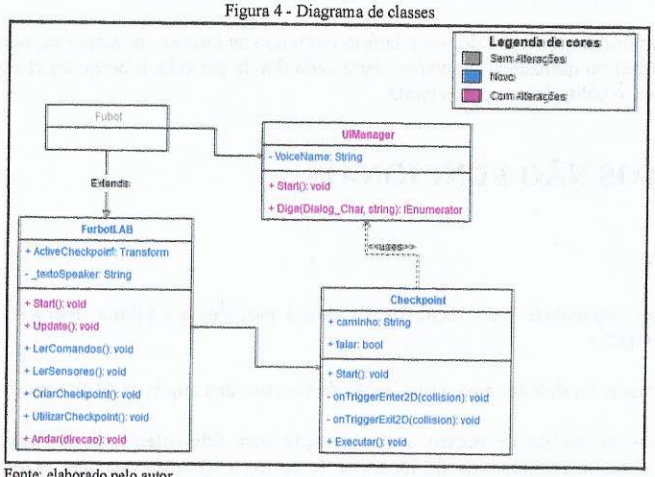
O protótipo tem como objetivo viabilizar a acessibilidade para pessoas não videntes na plataforma Furbot em sua versão Desktop (Windows / MacOS / Linux), para isso, foi criado um modo de jogo novo, chamado "modo labirinto", que utiliza descrição de tela como forma de audiodescrição, e um cenário escuro, limitando o campo de visão para usuários videntes. As modificações de dentro da fase, como a diminuição da visibilidade de cenário, foram feitas com a intenção de nivelar a dificuldade entre todos os tipos de usuários. Para a implementação do protótipo, foi utilizada a versão 3.0 do Furbot como base.

O desenvolvimento do protótipo seguiu os principais Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF) destacados a seguir:

a) permitir ao usuário se localizar no ambiente sem depender de recursos visuais (RF);
b) permitir ao usuário realizar as atividades sem se beneficiar da descrição do ambiente (RF);
c) permitir que o usuário configure a voz do TTS conforme suas preferências (RF);
d) permitir que o usuário faça uso de pontos de controle no decorrer do jogo para evitar que fique preso em *dead-ends* (RF)
e) criar um ambiente que permita que usuários videntes e não videntes joguem com o mesmo nível de dificuldade (RNF);
f) criar uma forma de orientar o usuário de um ponto de controle ao outro (RNF);
g) utilizar audiodescrição para ambientar o usuário (RNF);
h) utilizar o ambiente de desenvolvimento Unity 2D e linguagem C# (RNF).

O Furbot como um todo recebeu modificações para possibilitar a utilização do *asset* `RTVoice`, sendo que toda a plataforma recebeu o recurso de TTS desde o menu principal a fase de teste. Uma série de adaptações foram realizadas dentro da fase para que a acessibilidade fosse possível. De acordo com o diagrama de classes abstraído apresentado na Figura 4, foi criada a classe `FurbotLAB` estendendo da classe `Furbot` já existente no projeto. Dentro dela ocorreram modificações nos métodos `Start` e `Update`, para abrigar novas funcionalidades, sendo elas os `checkpoints`, a leitura de sensores, a leitura de comandos disponíveis, a movimentação pelo teclado e a leitura de status do Furbot.

Figura 4 - Diagrama de classes

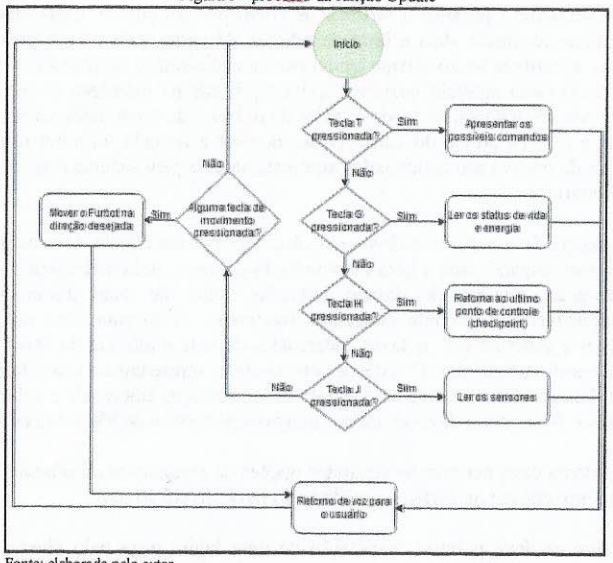


Fonte: elaborado pelo autor.

O processo do método `Update` da classe `FurbotLAB` foi um dos mais modificados, tendo como base o método `Update` da classe `Furbot` (a Figura 5 ilustra o processo atualmente). O método aguarda alguma das teclas de comando ser pressionada, para então fazer alguma ação determinada por aquela tecla. A tecla `T` é utilizada caso o jogador queira ouvir todos os comandos disponíveis. Nesse caso é informado que para mover o Furbot podem ser utilizadas as setas direcionais, ou a combinação `W, A, S, D`, bem como a tecla `H` para utilizar o "portal" para o último `checkpoint` visitado. Já tecla `G` para fazer a leitura dos sensores e informar o jogador do que se encontra em torno do Furbot. E a tecla `J` para executar o diagnóstico, que vai informar ao jogador a quantidade de energia e vidas restantes. Quando um `checkpoint` é utilizado, o Furbot retorna totalmente ao estado salvo, exceto pelo código final, ou seja, a vida e energia do Furbot voltam ao estado que foi salvo no checkpoint, independentemente do jogador ter encontrado vidas extras ou energia.

diagrama de atividades

Figura 5 - processo da função Update



Fonte: elaborado pelo autor.

3.2 IMPLEMENTAÇÃO

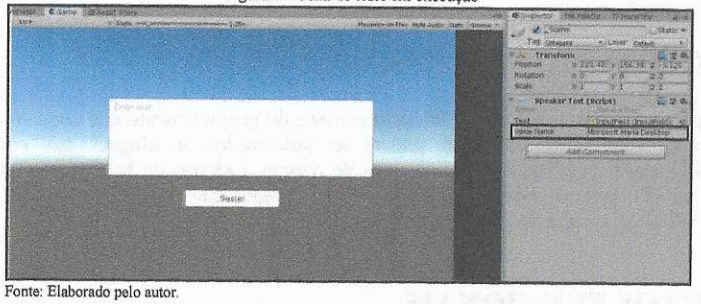
Esta seção traz os aspectos da implementação do protótipo, dividido em duas partes. Na primeira é descrito o processo de utilização do *asset* `RTVoice` e a criação de um projeto de teste para o *asset*. Na segunda parte, são apresentadas as novas funcionalidades integradas na fase de teste do protótipo.

3.2.1 Utilização do `RTVoice`

Para a função de TTS foi escolhido o *asset* `RTVoice`, distribuído pela Crosstales (2020), pois se adaptava melhor as necessidades do projeto. Este *asset* já vem com uma cena de demonstração, porém é uma cena um tanto complexa que não permite o entendimento completo de como utilizar o *asset* em código fonte. Para obter esse entendimento foi criado um projeto de teste para o *asset*, seguindo um tutorial também disponibilizado pela Crosstales (2017).

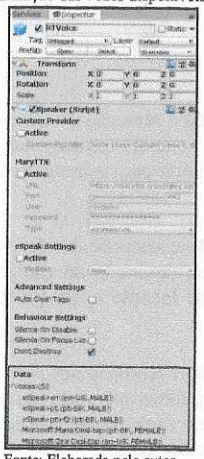
Nesta aplicação de teste foi utilizado um campo de texto que vai ser utilizado como fonte da fala e um botão para ativar o teste. A Figura 6 mostra a tela criada no projeto de teste, a variável `VoiceName` controla qual voz será utilizada para o TTS, que neste caso foi utilizado a voz da `Microsoft Maria Desktop`. Para saber quais as vozes estão disponíveis no dispositivo, é necessário selecionar o *asset* do `RTVoice` na cena, e na seção `Data`, é exibido um vetor com as vozes instaladas no dispositivo (Figura 7).

Figura 6 - Cena de teste em execução



Fonte: Elaborado pelo autor.

Figura 7 - Localização das vozes disponíveis no dispositivo



Fonte: Elaborado pelo autor.

Ao pressionar o botão o método `Speak()` da classe `SpeakerTest` é acionado, através do método `Speaker.Speak()` o *asset* transmite o texto para o TTS nativo do dispositivo, informando também a voz que deve ser utilizada, e dando um retorno praticamente em tempo real (Crosstales, 2020). É possível verificar o código da classe de testes do `RTVoice` no Quadro 4.

Quadro 4 - Classe `SpeakerTest`

```
using Crosstales.RTVoice;

public class SpeakerTest : MonoBehaviour
{
  public InputField Text;
  public string VoiceName;
  public void Speak()
  {
    Speaker.Speak(Text.text, null, Speaker.VoiceForName(VoiceName));
  }
}
```

Fonte: Elaborado pelo autor.

inserir na descrição do Furbot

Para o correto funcionamento do método é necessário a utilização do `Crosstales.RTVoice` nos fontes onde o `Speaker` for necessário. Também é necessária a utilização do *asset* `RTVoice` em uma das cenas do projeto. No Furbot o *asset* foi incluído na cena do menu principal, sem necessidade de ser replicado em outras cenas do projeto.

3.2.2 Novas funcionalidades

Para realizar a adaptação do Furbot foram necessárias algumas modificações no funcionamento inicial do jogo. No protótipo o jogo não se dá mais em dois momentos, a programação ocorre ao mesmo tempo que a execução, e as linhas de código são produzidas de forma mais automática, não dependendo do jogador. Além disto, como já mencionado anteriormente, foram realizadas alterações visuais no Furbot, escurecendo todo o cenário e permitindo que jogadores videntes tenham acesso através da visão a mesma área que os jogadores não videntes terão ao utilizar os sensores.

Foram criados dois botões no menu principal do jogo, permitindo que o jogador acesse o modo labirinto, e as configurações de usuário. Tais configurações alteram a voz, velocidade de fala e tonalidade utilizadas pelo TTS. As configurações ficam salvas em `PlayerPrefs` possibilitando que o jogo seja fechado e reaberto sem perder as seleções do usuário. Ao entrar nas configurações são carregadas as vozes em português disponíveis no dispositivo em um `Dropdown`, conforme o código exibido no Quadro 5.

Quadro 5 - Função que carrega as vozes disponíveis para a seleção do usuário

```
void LoadVoices()
{
  voicesConteiner.ClearOptions();
  if(Speaker.isTTSAvailable && Speaker.areVoicesReady && voicesConteiner
!= null)
  {
    foreach (Crosstales.RTVoice.Model.Voice v in Speaker.Voices)
    {
      Dropdown.OptionData data = new Dropdown.OptionData();
      data.text = v.Name;
      voicesConteiner.options.Add(data);
    }
  }
  else if(voicesConteiner == null)
  {
    throw new System.Exception("Conteiner de vozes null");
  }
}
```

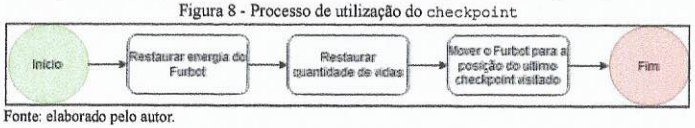
Fonte: Elaborado pelo autor.

Dentro da fase foram acrescentadas novas funcionalidades ao Furbot, o método `Andar` recebeu um retorno sonoro, dizendo ao jogador para que direção ele andou. Além da alteração para que os movimentos executados pelo Furbot sejam em tempo de execução. Para possibilitar que um usuário não vidente se localize quanto a obstáculos, como paredes, foi criada a função `LerSensores`, que busca as informações dos sensores do Furbot e as traz ao jogador de forma sonora.

A função `LerComandos` dá ao jogador todas as teclas e comandos disponíveis, informando como utilizá-los, como um breve tutorial. A função de leitura de status retorna ao jogador a quantidade de vidas disponível e a quantidade de energia que o robô ainda possui.

Para garantir os avanços durante a fase foi criado um sistema de `checkpoints`, ou seja, o jogador atinge pontos específicos do mapa, aos quais ele pode retornar caso se perca no labirinto. Ao entrar em um `checkpoint` o jogador recebe as coordenadas para o próximo. Essas informações são passadas apenas na primeira vez que o jogador chega ao `checkpoint`. Caso o jogador se perca, ele pode optar por voltar ao último checkpoint visitado, e neste processo ele também restaura os status do robô salvos no último checkpoint, ignorando caso o jogador tenha encontrado uma vida ou energia antes de se perder. O processo de utilização do checkpoint é ilustrado pela Figura 8.

Figura 8 - Processo de utilização do `checkpoint`



Fonte: elaborado pelo autor.

A drone S-223 foi utilizada para reproduzir as falas, sendo mantida como uma auxiliar para o Furbot, causando uma pequena alteração no método `Diga` da classe `UIManager`, que recebeu o trecho de código referente ao `Speaker` do `RTVoice` conforme o código no Quadro 6. Neste caso foram utilizados 3 parâmetros opcionais do método `Speak()`, que controlam se o retorno em audio deve ser dado automaticamente ou não, a velocidade de fala e a tonalidade do TTS

Quadro 6 - Função `Diga` com implementação do `RTVoice`

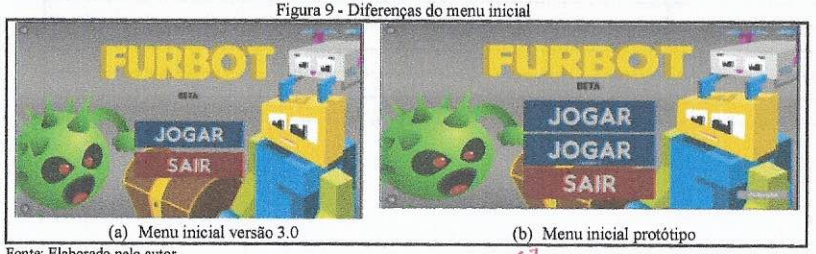
```
public IEnumerator Diga(Dialog_Char personagem, string texto)
{
  //debug.SetActive(true);
  ifCodigo.readOnly = true;
  dialogPanel.MostrarTexto(personagem, texto);
  Speaker.Speak(texto, null,
Speaker.VoiceForName(PlayerPrefs.GetString("VoiceName")), true,
PlayerPrefs.GetFloat("Rate"), PlayerPrefs.GetFloat("Pitch"));

   yield return new WaitForSeconds(3.0f);

   debug.SetActive(false);
#if !UNITY_ANDROID && !UNITY_IOS
   ifCodigo.readOnly = false;
#endif
}
```

Fonte: Elaborado pelo autor.

3.3 OPERACIONALIDADE

Nesta seção serão abordadas as diferenças entre a versão 3.0 do Furbot e o protótipo criado no decorrer deste trabalho. A primeira grande diferença entre as duas versões está na remoção da música de fundo que havia na versão 3.0 para dar lugar ao TTS. O menu inicial recebeu alterações já mencionadas, ganhando dois novos botões e a navegação através das setas direcionais do teclado e da tecla Enter. As diferenças do menu inicial do protótipo e da versão 3.0 são exibidas na Figura 9.

Figura 9 - Diferenças do menu inicial

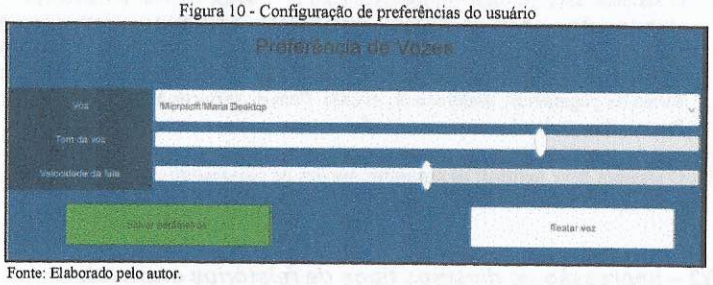


(a) Menu inicial versão 3.0 (b) Menu inicial protótipo

Fonte: Elaborado pelo autor.

Quando um modo de jogo é escolhido no protótipo e não existe uma configuração de voz salva, o jogo remete o usuário a uma tela de parametrização de voz (Figura 10). Nessa tela, é possível definir o tipo da voz dentre as já existentes no dispositivo do usuário, a velocidade de fala e a tonalidade da voz. É possível perceber que nessa tela os tipos de voz estão limitados a cultura `pt-BR`, não existindo a possibilidade de utilizar um idioma estrangeiro, salvo caso do usuário não ter nenhuma voz em português instalada em seu computador. Neste caso a biblioteca `RTVoice` utiliza a primeira voz disponível para a fala, geralmente no idioma inglês. Após salvas, as preferências de voz podem ser alteradas a partir do menu inicial do Furbot no botão `Configurações`.

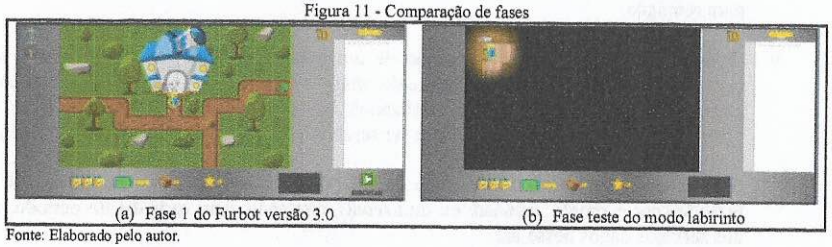
Figura 10 - Configuração de preferências do usuário



Fonte: Elaborado pelo autor.

A fase criada no protótipo tem um cenário escuro muito diferente do cenário apresentado na fase 1 da versão 3.0, conforme a Figura 11. No protótipo o jogador deve encontrar a pista que o leva a finalizar a fase seguindo as orientações dadas pela assistente S-223 ao longo do percurso. Essas orientações são dadas conforme o jogador atinge os `checkpoints` no labirinto.

Figura 11 - Comparação de fases

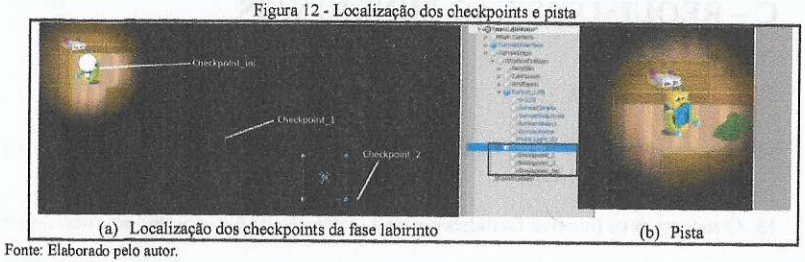


(a) Fase 1 do Furbot versão 3.0 (b) Fase teste do modo labirinto

Fonte: Elaborado pelo autor.

Na fase de teste do protótipo existem três `checkpoints` (representados na Figura 12 (a) por legendas que não existem no jogo em si) ao todo, sendo um deles o ponto de partida da fase. Caso o jogador saia de um `checkpoint` do meio da fase e retorne a um `checkpoint` anterior, o progresso do jogador é salvo no `checkpoint` anterior novamente, pois o jogador perdeu energia e aumentou o tamanho do código final do Furbot. O modo labirinto acaba quando o jogador encontrar a `pista` que o Furbot está procurando, que pode ser vista na Figura 12 (b), ou quando a energia ou vidas do Furbot se esgotarem.

Figura 12 - Localização dos checkpoints e pista



(a) Localização dos checkpoints da fase labirinto (b) Pista

Fonte: Elaborado pelo autor.

Toda ação executada no modo labirinto tem um retorno em áudio, caso o jogador bata em uma parede, a assistente S-223 irá informar que o caminho está incorreto (Figura 13). Caso o jogador se movimente, ela irá informar que o Furbot andou na direção informada. Porém para diminuir o processo este retorno é apenas pelo TTS, não envolvendo a caixa de diálogo.

Figura 13 - Diálogo de assistência S-223

Fonte: Elaborado pelo autor.

4 RESULTADOS

Para criar um protótipo com mais objetividade e melhor adaptação para não videntes, foi realizada uma reunião em maio de 2020 com as especialistas Fernanda (Pacheco, 2020) e Leia (Andrade, 2020) do Centro Municipal de Educação Alternativa (CEMEA) de Blumenau. Durante a entrevista foi feita uma demonstração do protótipo desenvolvido para as especialistas, que deram algumas sugestões de melhorias a serem aplicadas ao protótipo.

A versão utilizada para demonstração na reunião possuía a possibilidade de criação de `checkpoints` pelo jogador, sem `checkpoints` fixos, nenhum tipo de orientação sobre os caminhos a serem seguidos, e sem a possibilidade de configurações da voz do TTS. Estas modificações foram aplicadas conforme as orientações dadas pelas especialistas. Além das modificações feitas, a especialista Fernanda sugeriu o uso de som 3D próximo a objetivos ou obstáculos, que acabou não sendo implementado.

Após as alterações o protótipo foi testado por cinco usuários, destes um usuário realizou o teste vendado e um usuário possui baixa visão. Para a coleta de informações destes testes foi aplicado um formulário on-line via Google Forms. Do total de usuários testados, todos utilizam o computador com frequência e costumam jogar no computador, 60% já havia tido algum contato com a plataforma Furbot, e apenas um deles estava familiarizado com leitores de tela.

Nenhum dos usuários teve dificuldades em navegar pelo menu inicial. Na tela de configuração do TTS o usuário que testou vendado relatou que precisou de ajuda com as configurações, pois a tela não possui um guia em TTS. Durante a fase teste do jogo, nenhum usuário relatou dificuldades com as novas funções. Porém um usuário identificou que ocorreram algumas falhas ao informar os status do Furbot, que estava com 72% de energia e ao ler os sensores foi lido 76%. Apenas o usuário que testou vendado precisou de auxílio para interagir com a plataforma, e todos os usuários classificaram as instruções dadas durante a fase como de fácil compreensão. No geral, os usuários se sentiram bem motivados a concluir a fase e acharam o sistema de `checkpoints` de fácil uso.

Um usuário classificou a aplicação do TTS como ruim, pois ao pular falas ou executar ações muito rápido, as falas se sobrepõem, o que gera confusão. Algumas sugestões de melhorias foram dadas pelos usuários, como cortar a fala do TTS em execução ao rodar a próxima evitando que as falas se sobreponham. Ajustar a quantidade de textos por caixa de diálogo ou criar uma forma de elas serem *scrollable*, e ajustar o calculo das posições, pois dependendo a resolução selecionada ao andar o Furbot acaba se perdendo, encontrando paredes onde não existem entre outras situações do gênero.

5 CONCLUSÕES

Com base nos resultados, o protótipo cumpriu seus objetivos, mostrando que é possível aplicar um leitor de tela a plataforma Furbot em sua versão desktop para torná-la acessível a pessoas não videntes, sem perder a essência da plataforma, e manteando uma faixa de dificuldade similar para todos os usuários. Contudo foram encontradas algumas limitações na implementação. No menu inicial, caso os botões percam o foco, não é possível controlar o menu pelas setas do teclado. As caixas de diálogo ficam muito pequenas para textos maiores, e não foi encontrado uma forma de apresentar os diálogos em caixas separadas que saem em sequência, pois a caixa de diálogo se trata de uma co-rotina. O jogo deve ser rodado em resoluções especificas (1360x768 e 1600x1024) ou a plataforma se perde quanto as posições. Esta limitação já veio da versão 3.0 do Furbot, salvo o foco dos botões.

Diferente do trabalho de Sobral *et. al.* (2017), o protótipo desenvolvido permite que o usuário acompanhe mais facilmente as instruções e as falas, pois permite que o usuário configure parâmetros da voz utilizada conforme suas preferências. Da mesma forma que Costa (2013), existiram algumas dificuldades na dosagem de informações, com alguns diálogos ficando muito extensos, e um pouco repetitivos. Diferindo do trabalho de Kraemer (2017), não foi implementada nenhuma forma de reconhecimento de voz, sendo que todas as ações são realizadas através do teclado.

Durante a fase de testes foram encontrados alguns pontos de melhoria, como melhorar a tela de configuração da voz possibilitando que um usuário não vidente configure o TTS sozinho, estender o uso do TTS aos dispositivos móveis e web, aplicar o som 3D sugerido pela especialista Fernanda (Pacheco, 2020), melhorar a quantidade de diálogos para orientar melhor os usuários, criar um sistema de *checkpoints* e orientações de direção dinâmico, entre outras melhorias.

REFERÊNCIAS

ANDRADE, L. P. de; PACHECO, F. J. **Entrevista sobre demonstração do protótipo**. Entrevistador: Caroline Batistel. Blumenau. 2020. Entrevista feita em reunião pelo Hangouts – não publicada.

BRASIL. Lei nº 10.098, de 19 de dezembro de 2000. Lei de promoção a acessibilidade, garoante direitos as pessoas com deficiência. **Diário Oficial da União**: seção 1, Brasília, DF, p. 2, 20 dez. 2000.

BRASIL Portaria nº 142, de 16 de novembro de 2006. Institui o Comitê de Ajudas Técnicas (CAT). **Diário Oficial da União**: seção 2, Brasília, DF, p. 3, 16 nov. 2006.

BRASIL. Subsecretaria Nacional de Promoção dos Direitos da Pessoa com Deficiência. Comitê de Ajudas Técnicas. **Tecnologia Assistiva** – Brasília: CORDE, 2009. 138 p.

COOK, A. M.; POLGAR, J. M. **Assistive Technologies: Principles and Practice**. 4 ed. St. Louis, Missouri: Elsevier Health Sciences, 2014.

COSTA, D. **Blind Counter-Strike: Um jogo de FPS para deficientes visuais**. 2013. 74 f. Monografia (Bacharelado em Ciência da Computação) - Curso de Ciência da Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre.

CROSSTALES. **RT-Voice: Basic Tutorial**. 2017. Disponível em: https://www.youtube.com/watch?v=OJyVgCmX3wU. Acesso em: 10 abr. 2020.

CROSSTALES. **RT-Voice PRO**: Hearing is understanding - Documentation. 2020. Disponível em: https://www.crosstales.com/media/data/assets/rtvoice/RTVoice-doc.pdf. Acesso em: 21 jun. 2020.

DIAS, C. **Usabilidade na WEB**. Rio de Janeiro: Alta Books, 2007.

FRANCO, E. P. C.; SILVA, M. C. C. C. da. Audiodescrição: Breve Passeio Histórico. In MOTTA, L. M. V. M.; FILHO, P. R. (org.). **Audiodescrição. Transformando Imagens em Palavras**, São Paulo: Secretaria de Estado dos Direitos da Pessoa com Deficiência, 2010, p. 23-42.

GRANOLLERS, T. **MPlu_a Una metodologia que integra la ingenieria del software, la interacción persona-ordenador y la accesibilidad en el contexto de equipos de desarrollo multidisciplinares**. 2004. 77 f. Tese de doutorado, Universidade de Lérida, Lérida.

GROS, B. **The impact of digital games in education**. First Monday, v. 8, n. 7, jul. 2003.

IBGE – Instituto Brasileiro de Geografia e Estatística. **Censo demográfico 2010**: Características da população e dos domicílios – Resultados do Universo. Rio de Janeiro. 2011.

KRAEMER, R. G. **Tecnologia Assistiva: Tornando Jogo de Mesa Acessível para Cegos com Auxílio de Aplicativo Móvel de Reconhecimento de Imagem**. 2017. 63 f. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Curso de Ciência da Computação, Universidade Regional de Blumenau, Blumenau

MATTOS, M.; SANTOS, B. F. F.; TRIDAPALLI, J. G.; ZUCCO, F.; WUO, A. FURBOT - Desenvolvimento cognitivo infantil através de atividades de programação de computadores. In: Seminário de Extensão Universitária da Região Sul, 37 ed., 2019, Florianópolis. **Anais...** Florianópolis: UFSC, 2019a. p. 112

MATTOS, M.; KOHLER, L. P. A.; ZUCCO, F. D.; FRONZA, L.; BIZON, A.; UGARTE, H.; SANTOS, B. F. F.; GIOVANELLA, G. C. Ambiente de programação para a introdução da lógica de programação **Anais do Workshop de Informática na Escola** [S.l.], ISSN 2316-6541, p. 1259-1264, nov. 2019b.

NUNES E. V.; MACHADO F. O.; VANZIN T. Audiodescrição como tecnologia assistiva para o acesso ao conhecimento por pessoas cegas. In: ULBRICHT V. R.; VANZIN T.; VILLAROUCO V. (org.). **Ambiente virtual de aprendizagem inclusivo,** Florianópolis: Padion, 2011, p. 191-232.

PASSERINO, L. M.; MONTARDO, S. P. Inclusão social via acessibilidade digital: proposta de inclusão digital para Pessoas com Necessidades Especiais. **E-Compós**, v. 8, n 11, abril 2007

PRIETO, L. M.; TREVISAN, M. C. B.; DANESI, M. I.; FALKEMBACH G. A. M.; Uso das Tecnologias Digitais em Atividades Didáticas nas Séries Iniciais. **Renote: revista novas tecnologias na educação**, Porto Alegre, v. 3, n. 1, p.1-11, maio 2005.

SASSAKI, R. K. **Inclusão: construindo uma sociedade para todos.** Rio de Janeiro: WVA, 1997.

SAVI, R.; ULBRICHT, V. R. Jogos digitais educacionais: benefícios e desafios. **Renote: Revista novas tecnologias na educação**, Porto Alegre, v. 6, n. 2, 2008.

SOBRAL, F. V.; UMERES, L. F.; SCHANOSKI, W.; BARTELMEBS, C. R.; ASSIS, M. V. O. de. A Utilização de Role Playing Games Digitais como Ferramenta Complementar no Processo de Aprendizagem de Crianças Deficientes Visuais. In: Simpósio Brasileiro de informática na Educação, 6 ed., 2017, Recife. **Anais**... Recife: UFRPE, 2017. p. 635 – 644.