Tecnologia Assistiva: Tornando jogo de mesa acessível para cegos com auxílio de aplicativo móvel de reconhecimento de imagem

Aluno(a): Ronan Guimarães Kraemer

Orientador: Dalton Solano dos Reis

FURB
UNIVERSIDADE DE BLUMENAU

Roteiro

- Introdução;
- Objetivos;
- Fundamentação teórica;
- Requisitos;
- Especificação;
- Implementação;
- Resultados e discussões;
- Conclusão e sugestões.

Introdução

- Crescimento acentuado da tecnologia;
- Maior foco em inclusão social;
- Comitê de ajudas técnicas e Tecnologias assistivas;
- Board Games e Munchkin;
- Aplicação.

Objetivos

- Desenvolver o aplicativo de forma que possa ser utilizado ao menos em duas plataformas diferentes (Android e iOS) utilizando a ferramenta Ionic;
- Adaptar o jogo físico para aumentar a jogabilidade para o deficiente visual sem alterar as características do próprio jogo;
- Fazer com que a aplicação seja de acesso gratuito.

Munchkin

- Steve Jackson Games e Galápagos Games;
- Exploração e RPG;
- Comunicação;
- Cartas.

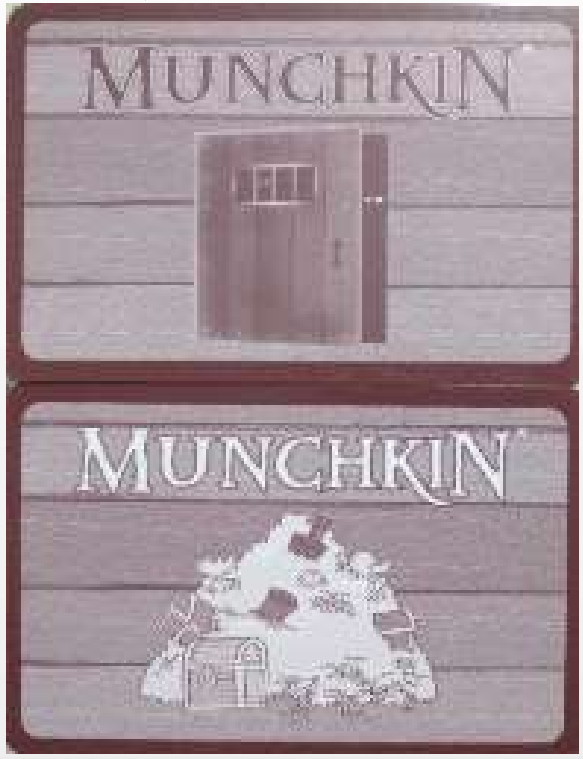


Figura 1


Figura 2

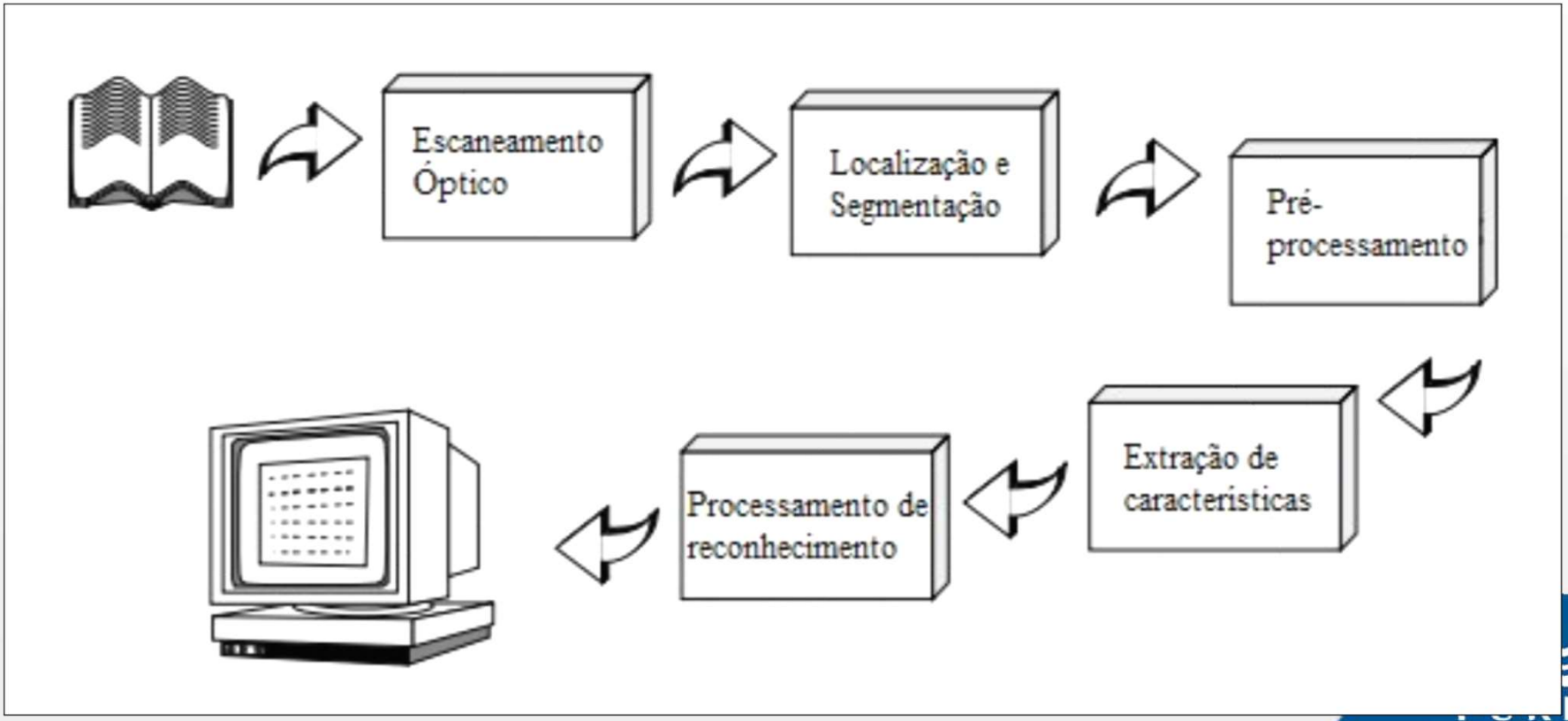
Ionic Framework

- HTML 5;
- Aplicações híbridas;
- Plugins e recursos nativos.

Recurso	Plugin	Link
câmera	cordova-plugin-camera	github.com/apache/cordova-plugin-camera
text to speech	cordova-plugin-tts	github.com/vilic/cordova-plugin-tts
speech to text	speech-recognition-plugin	github.com/macdonst/SpeechRecognitionPlugin
file transfer	cordova-plugin-file-transfer	github.com/apache/cordova-plugin-file-transfer

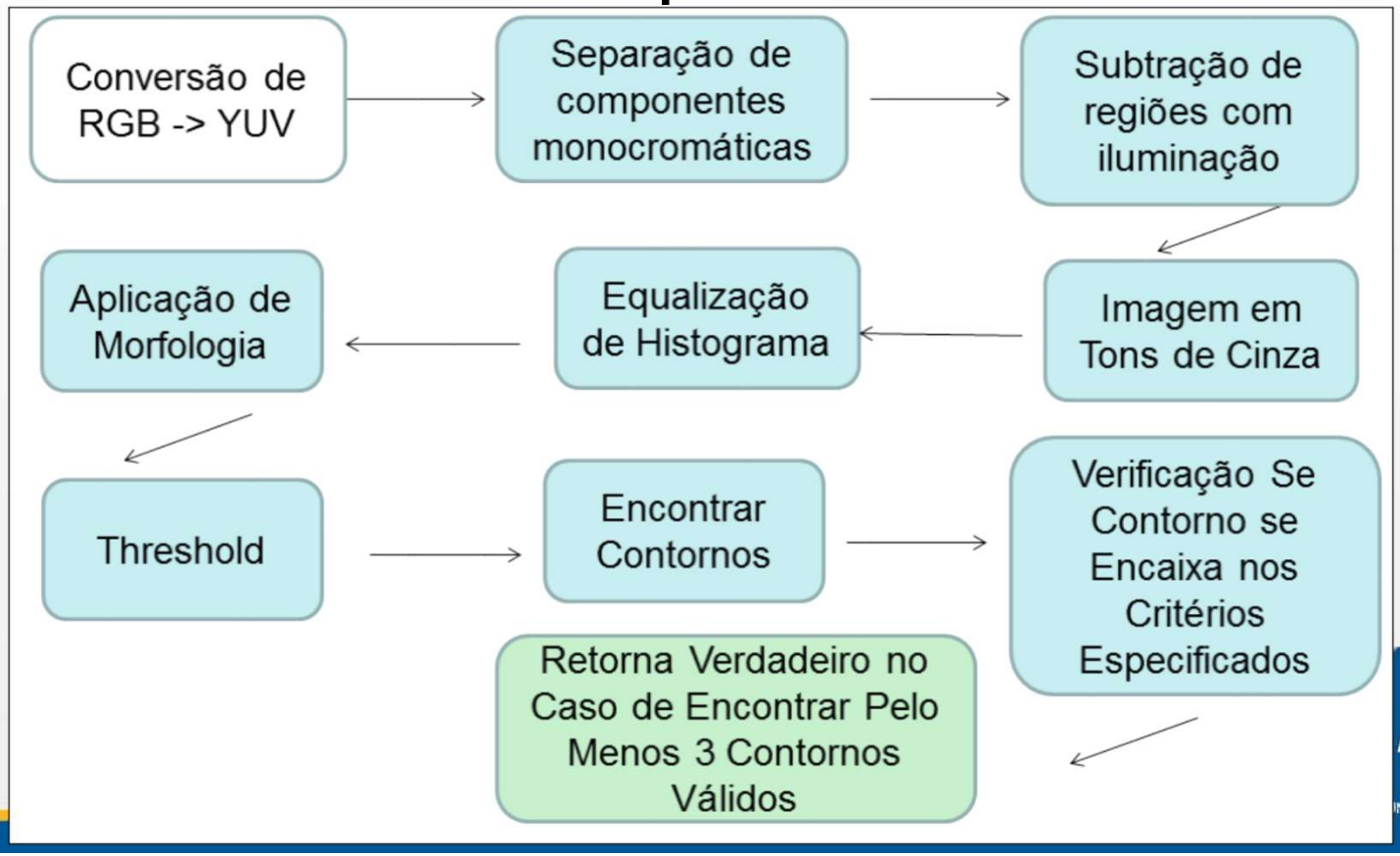
Optical Character Recognition

- Execução automática de identificação;
- Necessidade;
- Funcionamento.



Uso de visão computacional em dispositivos móveis para reconhecimento de faixas de pedestre

- Aplicativo;
- OpenCV, HOG, SVM e SVOX;
- Funcionamento do aplicativo.



Read4Blind

- Aplicativo;
- Tesseract e GoogleTTS;
- Funcionamento do aplicativo.

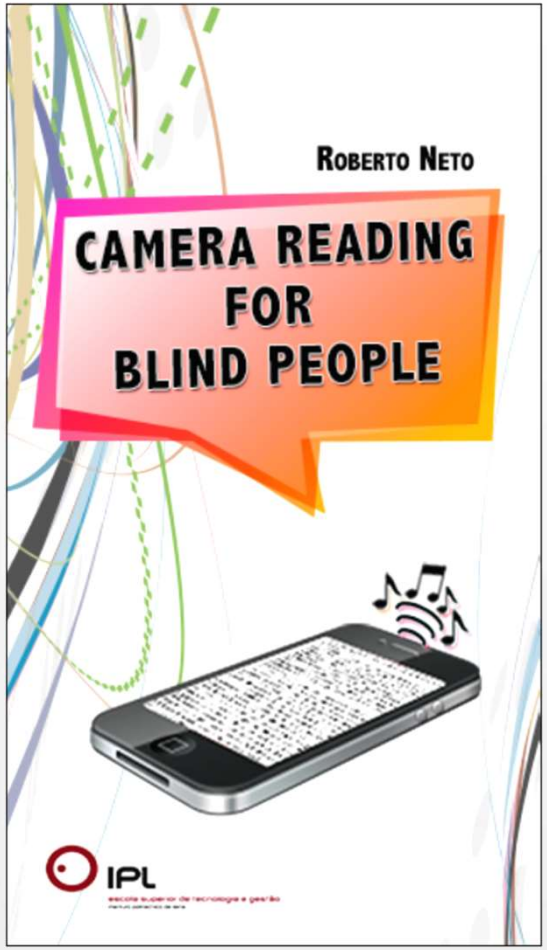


Figura 1

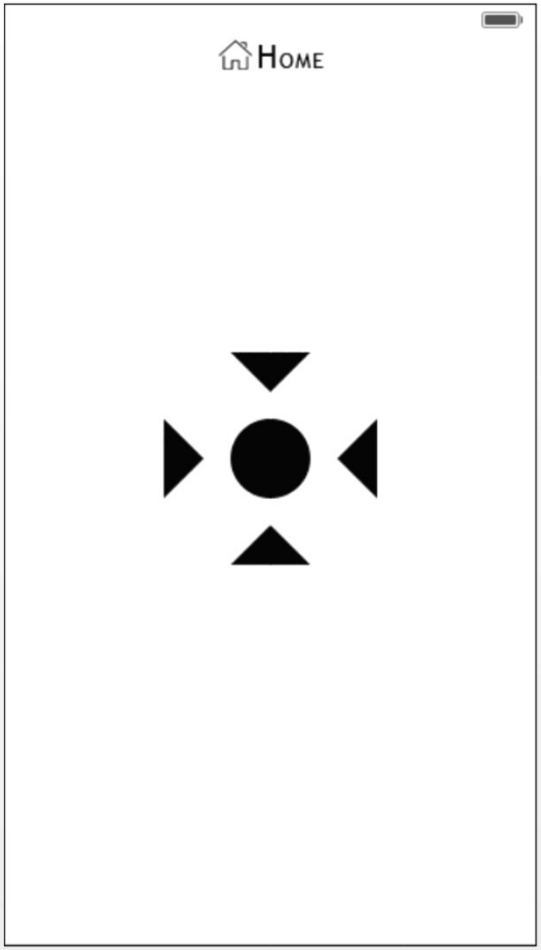


Figura 2

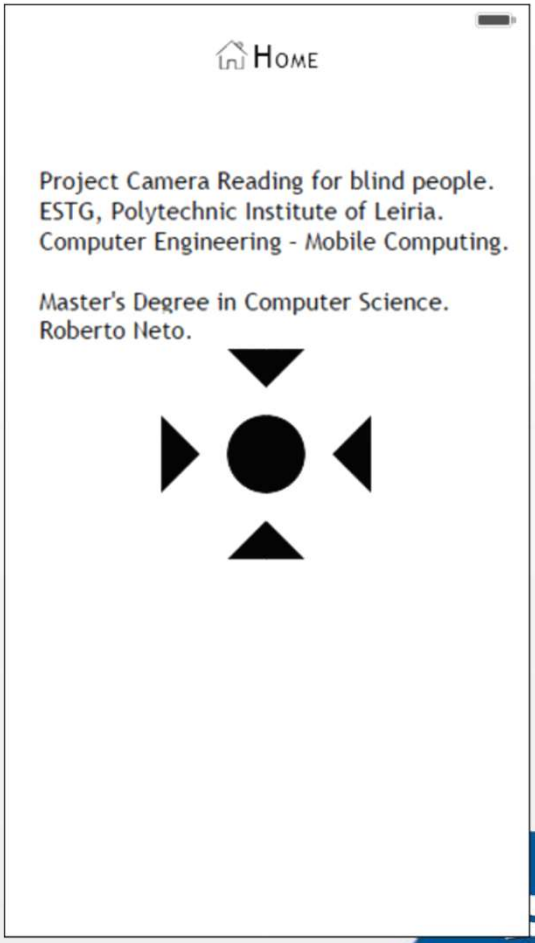


Figura 3

FURB
UNIVERSIDADE DE BLUMENAU

Aipoly

- Aplicativo;
- Inteligência artificial;
- Funcionamento do aplicativo.

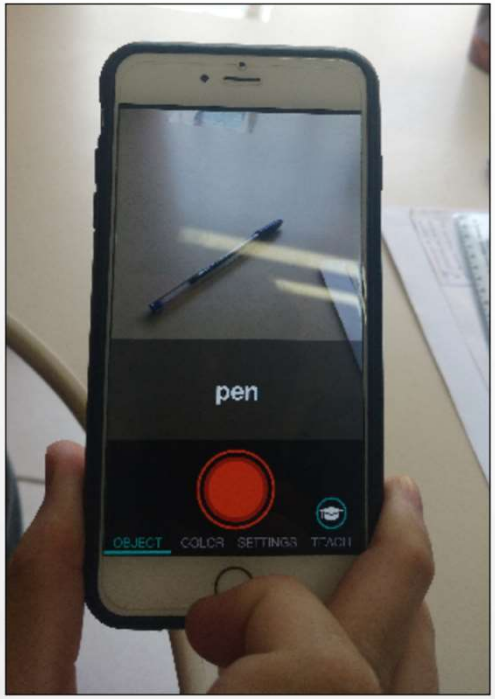


Figura 1

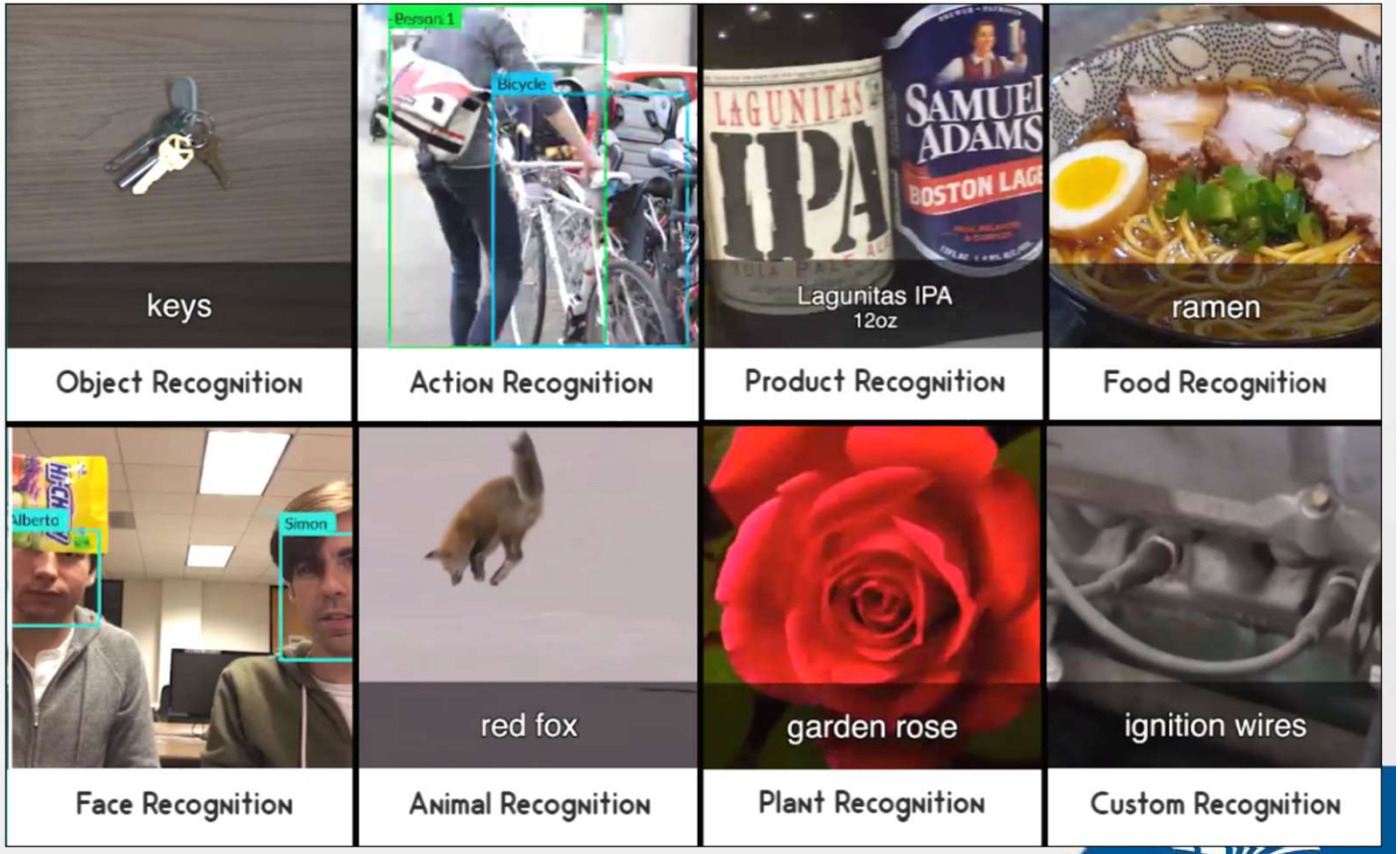


Figura 2

Requisitos funcionais

- O aplicativo deve permitir que o usuário tire foto da carta;
- O aplicativo deve efetuar o reconhecimento do texto existente na foto da carta;
- O aplicativo deve utilizar o texto reconhecido da foto para encontrar a carta correta;
- O aplicativo deve sintetizar a descrição da carta de forma clara e em português;
- O aplicativo deve permitir que o usuário possa repetir a descrição da carta;
- O aplicativo deve aceitar entradas de comando de voz.

Requisitos não funcionais

- O aplicativo deve ser implementado utilizando a ferramenta multiplataformas Ionic;
- O aplicativo deve ser desenvolvido com tecnologias web (HTML5, CSS e Javascript) no ambiente de desenvolvimento Visual Studio Code;
- O aplicativo deve utilizar a API Cloud Vision da Google para realizar o reconhecimento de texto.

Diagrama de casos de uso

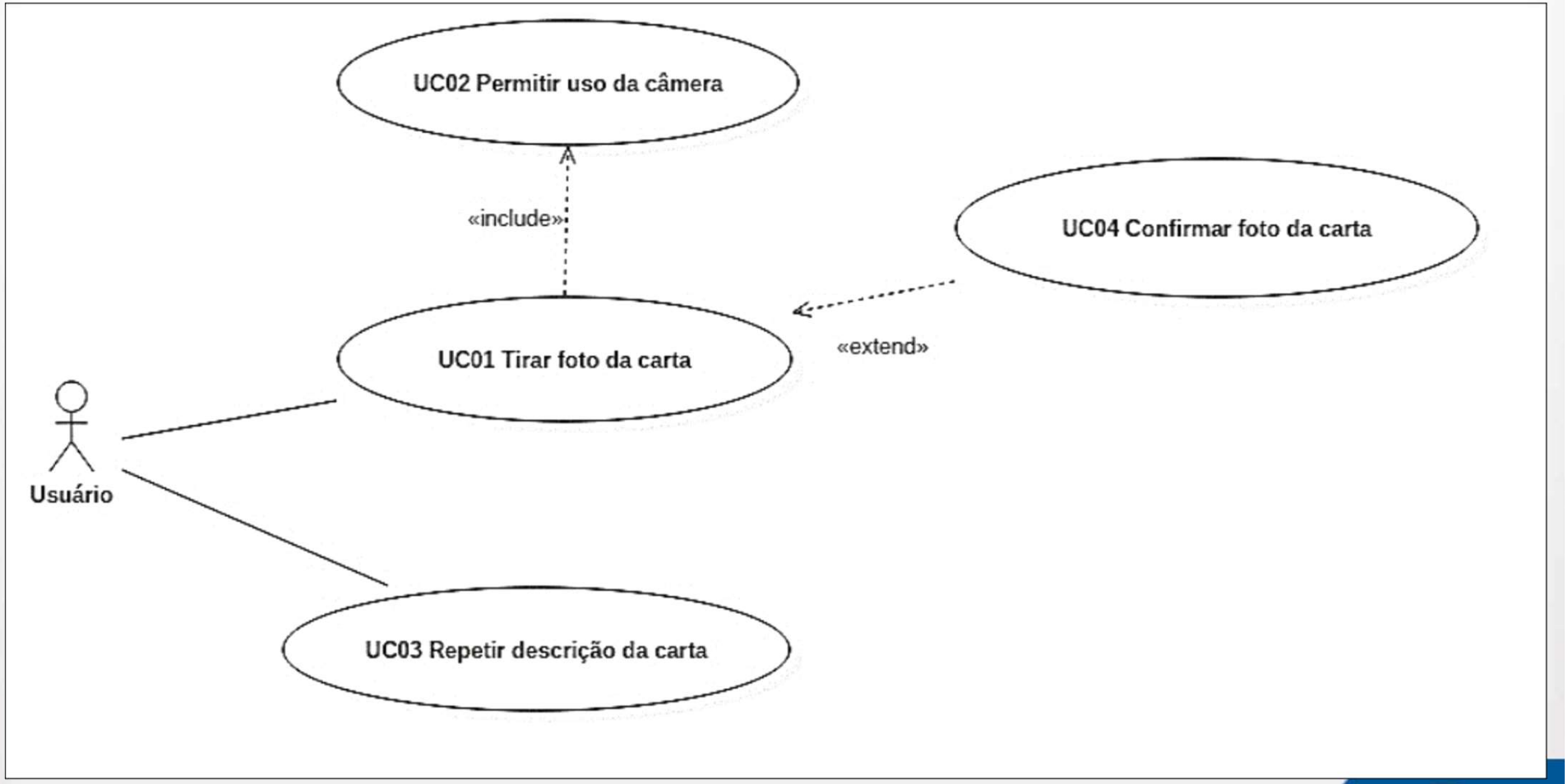


Diagrama de classes

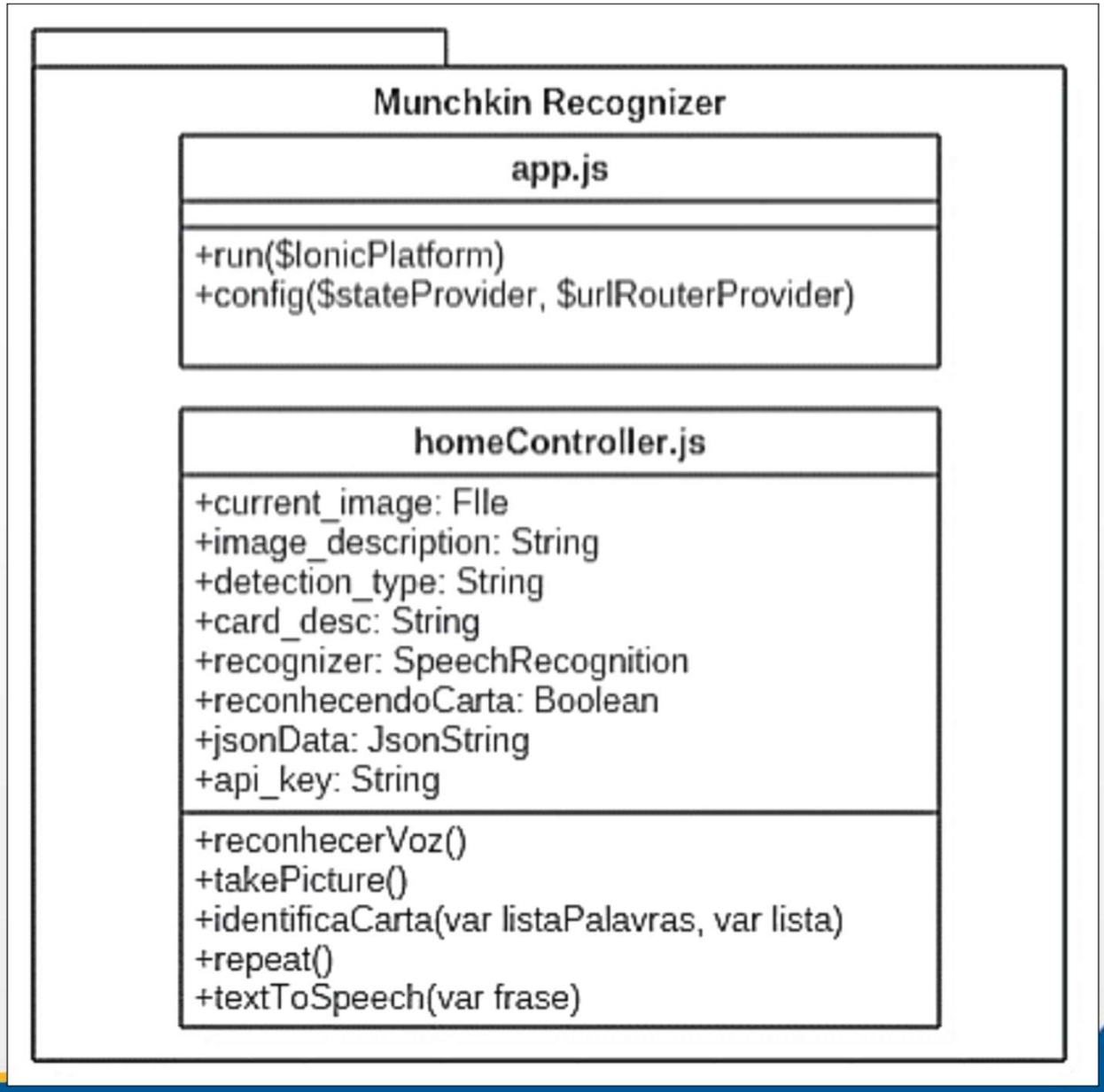
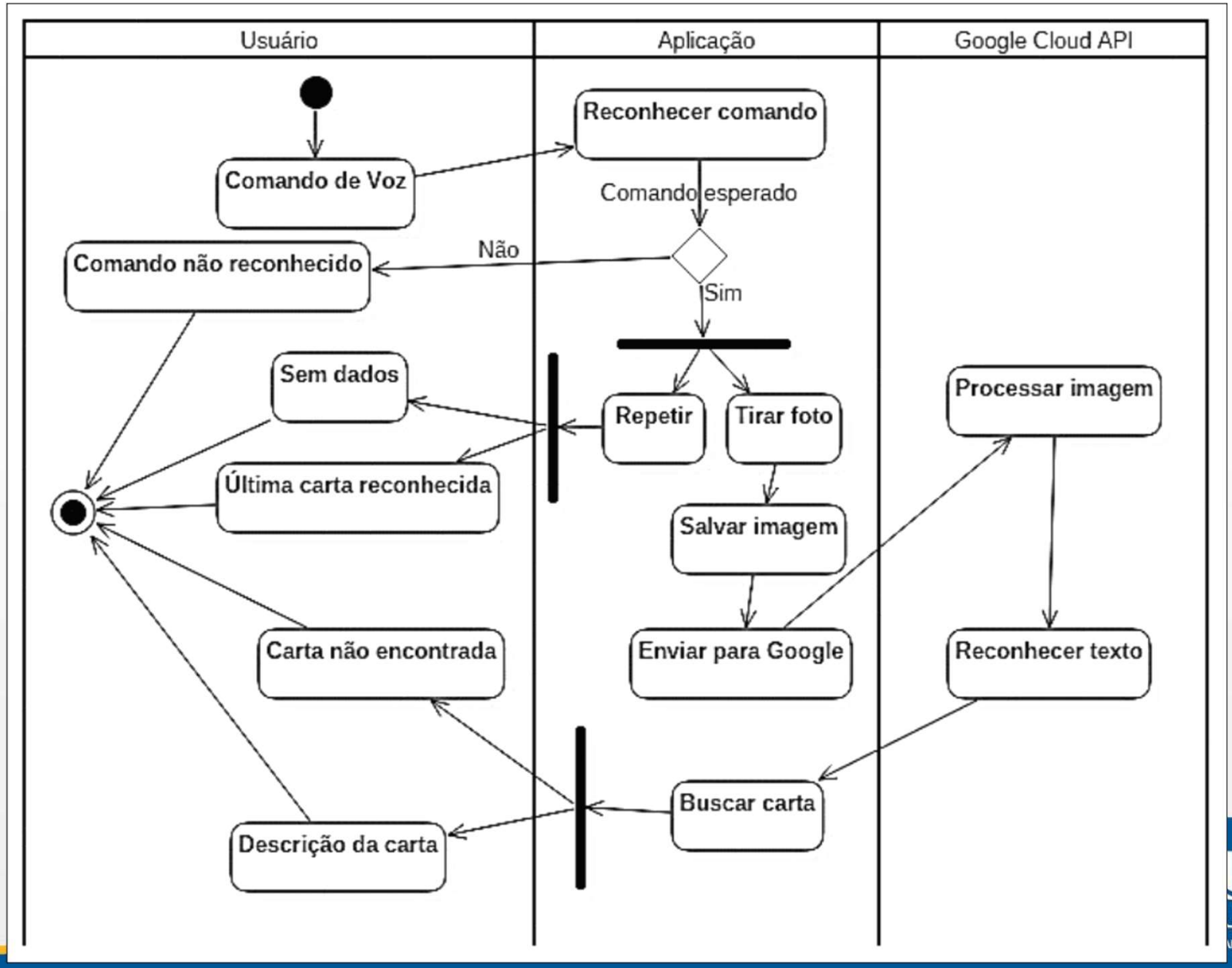


Diagrama de atividades



Arquitetura

Figura 1

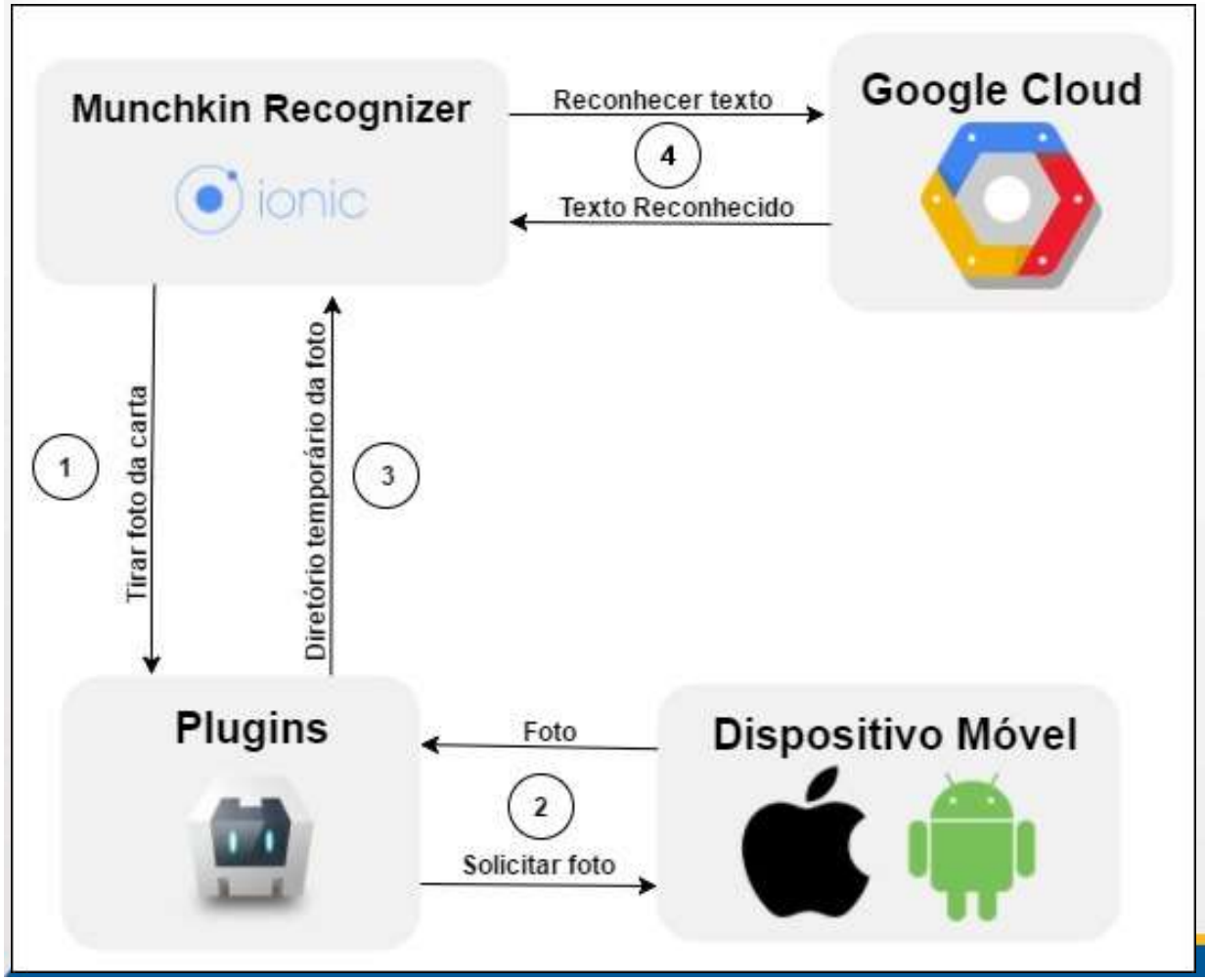


Figura 2

Acessando a câmera

```
69          $scope.takePicture = function(){
70              var options = {
71                  destinationType: Camera.DestinationType.DATA_URL,
72                  sourceType: Camera.PictureSourceType.CAMERA,
73                  targetWidth: 500,
74                  allowEdit: false,
75
76                  targetHeight: 500,
77                  correctOrientation: true,
78                  cameraDirection: 0,
79                  encodingType: Camera.EncodingType.JPEG
80              };
81
82              $scope.textToSpeech("Abrindo a câmera. Tire uma foto da carta para realizar o reconhecimento.");
83
84              $cordovaCamera.getPicture(options).then(function(imagedata){
85                  me.reconhecendo = true;
86                  me.card_desc = '';
87                  me.current_image = "" + imagedata;
88                  me.image_description = '';
89                  me.locale = '';
90
91                  //....
```

Sintetização de voz

```
226     $scope.textToSpeech = function(frase) {
227         TTS.speak({
228             text: frase,
229             locale: 'pt-BR',
230             rate: 1.2
231         }, function () {
232             // Do Something after success
233         }, function (reason) {
234             // Handle the error case
235         });
236     }
```

Enviando a imagem para processamento

```
111  $cordovaFile.writeFile(
112      cordova.file.cacheDirectory,
113      'file.json',
114      file_contents,
115      true
116  ).then(function(result){
117
118      var headers = {
119          'Content-Type': 'application/json'
120      };
121
122      options.headers = headers;
123
124      var server = 'https://vision.googleapis.com/v1/images:annotate?key=' + api_key;
125      var filePath = cordova.file.cacheDirectory + 'file.json';
126
127      if (me.developMode) {
128          me.tempo = 0.0;
129      }
130
131      me.tempo = new Date();
132
133      $cordovaFileTransfer.upload(server, filePath, options, true)
134      .then(function(result){
135
136          // ...
```

Reconhecimento de voz

```
41          $scope.reconhecerVoz = function() {
42              var texto = "";
43              me.recognizer = new window.SpeechRecognition();
44              me.recognizer.lang = "pt-BR";
45              me.recognizer.continuous = true;
46
47              me.recognizer.onresult = function(event) {
48                  if (event.results.length > 0) {
49                      texto = event.results[0][0].transcript;
50
51                      if (!me.reconhecendo) {
52                          if (texto == "tirar foto") {
53                              $scope.takePicture();
54
55                          } else if (texto == "repetir") {
56                              $scope.repeat()
57
58                          } else {
59                              $scope.textToSpeech('Comando inválido.');
60                          }
61                      }
62
63                  }
64              };
65
66              me.recognizer.start();
67          }
```

FURB
UNIVERSIDADE DE BLUMENAU

Reconhecimento da carta

```
177             for (i=0; i < listaPalavras.length; i++) {
178                 for (j=0; j < listaCartas.length; j++) {
179                     tempList = listaCartas[j].nome.split(" ");
180 
181                     for (k=0; k < tempList.length; k++) {
182                         if ($filter('uppercase')(tempList[k]) == $filter('uppercase')(listaPalavras[i])) {
183                             novaLista.push(listaCartas[j]);
184                             break;
185                         }
186                     }
187                 }
188 
189                 if (novaLista.length > 0) {
190                     if (novaLista.length == 1) {
191                         break;
192 
193                     } else {
194                         listaCartas = novaLista;
195                         novaLista = [];
196                     }
197                 }
198             }
```

Operacionalidade da Implementação

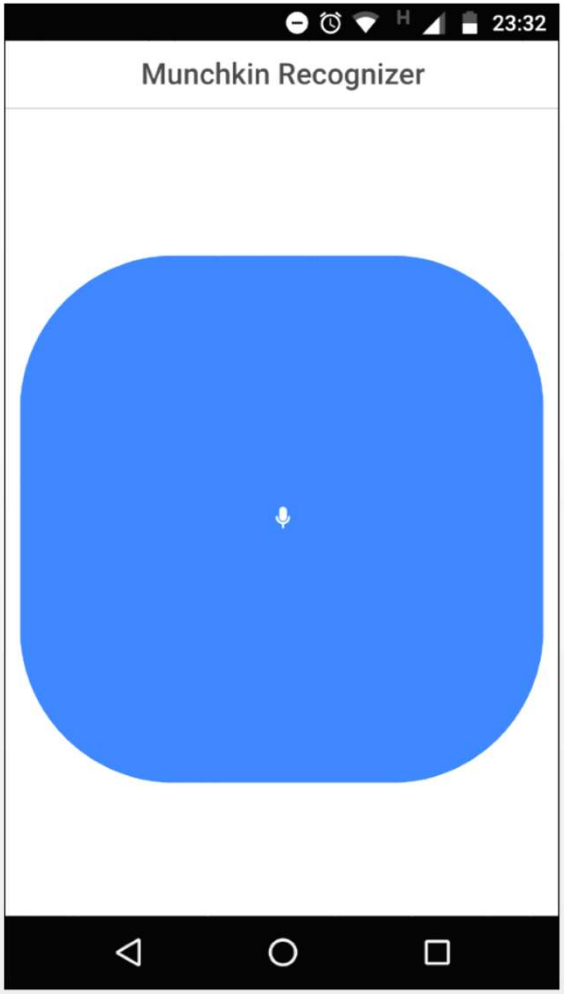


Figura 1

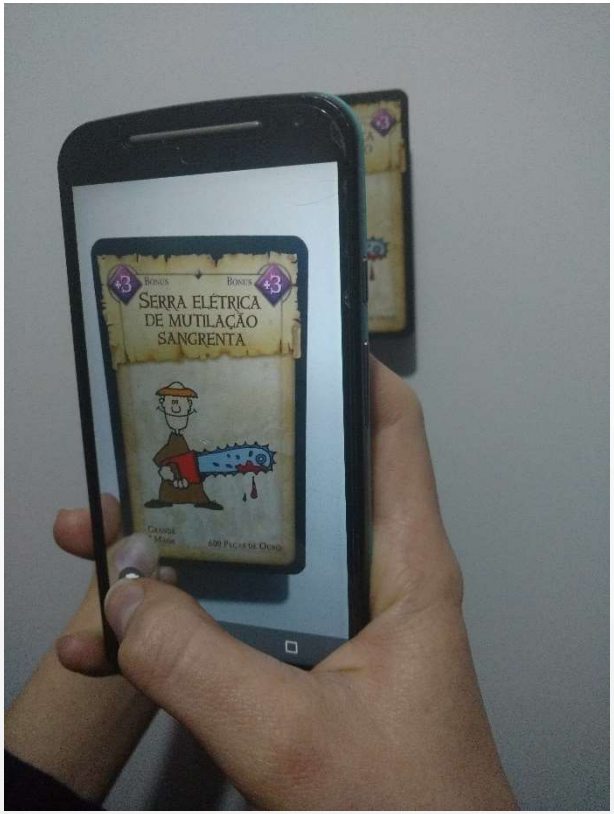


Figura 2

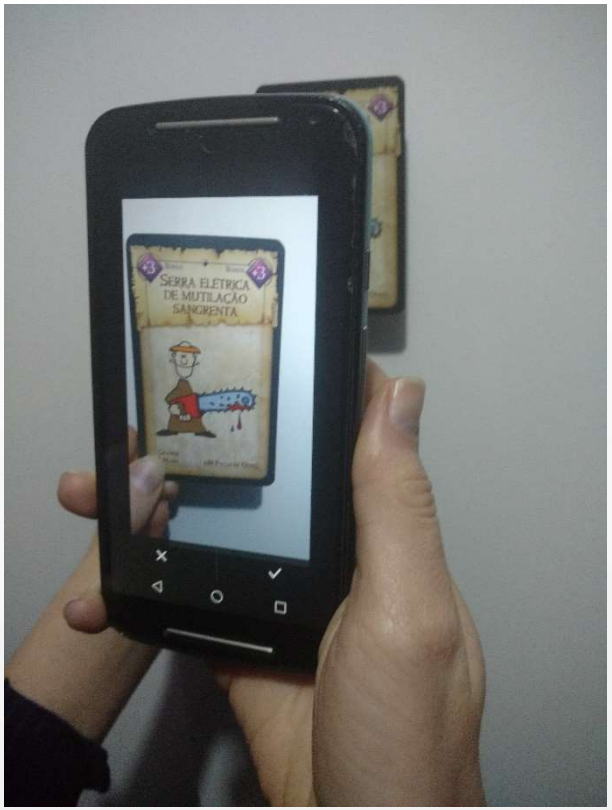


Figura 3

Testes individuais

Figura 1

Figura 2

Testes individuais

Figura 1

Figura 2

Figura 3

Resultados testes individuais

- Tempo de execução do OCR:

Conexão	Curta distância(ms)	Média distância(ms)	Longa distância(ms)
Wi-fi	2.024	2.237	2.199
Dados móveis	7.010	6.324	7.271

Tabela 1: Dados para os testes com **alto** nível de iluminação.

Conexão	Curta distância(ms)	Média distância(ms)	Longa distância(ms)
Wi-fi	1.670	1.546	1.645
Dados móveis	3.084	4.463	4.173

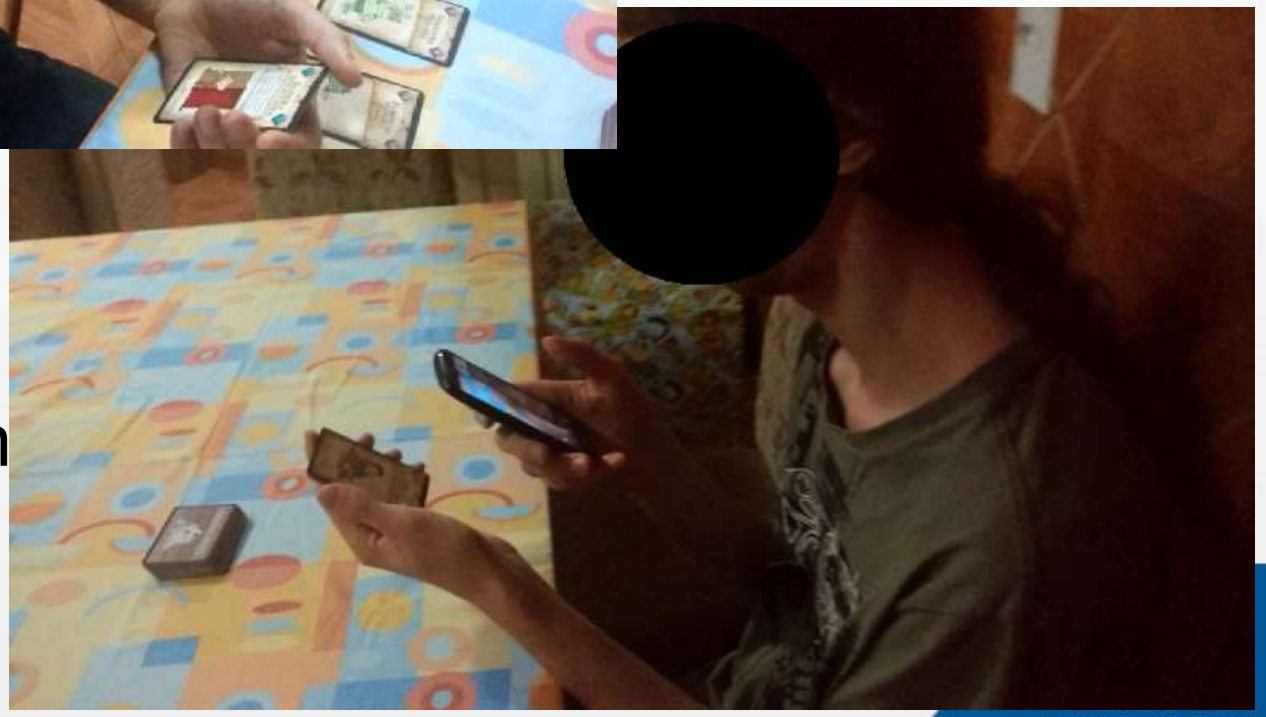
Tabela 2: Dados para os testes com **baixo** nível de iluminação.

- Precisão da aplicação:

Iluminação	Curta distância	Média distância	Longa distância
Alta	20/20	18/20	16/20
Baixa	15/20	11/20	5/20

Tabela 3: Dados para os testes de precisão do aplicativo.

Testes em grupo



Testes realizados com jogadores experientes e com olhos vendados.

Resultados testes em grupo

- Interface de uso;
- Reconhecimento de voz;
- Tirar foto;
- Confirmar foto;
- Precisão da aplicação.

Comparação com trabalhos Correlatos

	Sousa (2013)	Ferreira (2014)	Aipoly (2016)	Munchkin Recognizer
utiliza internet	Não	Não	Não/Sim	Sim
reconhecimento de texto	Não	Sim	Não	Sim
reconhecimento de imagem	Sim	Não	Sim	Não
sintetização de voz	Sim	Sim	Sim	Sim
tecnologia assistiva	Sim	Sim	Sim	Sim
disponível para uso	Não	Não	Sim	Não
plataforma suportada	Android	iOS	Hibrido	Hibrido
framework mobile	Nativo	Nativo	Nativo	Ionic

Conclusões

- Ferramenta auxiliar para jogo de Munchkin;
- Inovação;
- Testes e resultados;
- Objetivos.

Sugestões

- Criar uma aplicação nativa da câmera, afim de eliminar o problema da confirmação da foto;
- Eliminar a necessidade de um botão para realizar os comandos de voz;
- Realizar a implementação de um aplicativo nativo para utilizar as ferramentas de processamento de imagem de forma off-line;
- Criar uma página web para que a comunidade de jogadores alimente o mapeamento das cartas de forma orgânica;
- Adaptar a ideia para diferentes tipos de jogos de mesa;
- Implementar o reconhecimento em tempo real, eliminando a necessidade de tirar uma foto da carta.

Processamento de Imagens

- Definição;
- Pré-processamento e OCR;
- Tratamentos de imagens:
  - negativo,
  - binarização,
  - brilho,
  - expansão,
  - compressão.