

**CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO
ITAJAÍ - UNIDAVI**

VINÍCIUS DALASTRA

**PROTÓTIPO DE SISTEMA PARA DETECÇÃO E IDENTIFICAÇÃO DE PESSOAS
EM MULTIDÕES ATRAVÉS DE SOFTWARE PARA RECONHECIMENTO
BIOMETRICO FACIAL**

**RIO DO SUL
2018**

**CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO
ITAJAÍ - UNIDAVI**

VINÍCIUS DALASTRA

**PROTÓTIPO DE SISTEMA PARA DETECÇÃO E IDENTIFICAÇÃO DE PESSOAS
EM MULTIDÕES ATRAVÉS DE SOFTWARE PARA RECONHECIMENTO
BIOMETRICO FACIAL**

Trabalho de Conclusão de Curso a ser apresentado ao curso de Sistemas da Informação, da Área das Ciências Naturais, da Computação e das Engenharias do Centro Universitário para o Desenvolvimento do Alto Vale do Itajaí, como condição parcial para a obtenção do grau de Bacharel em Sistemas de Informação.

Prof. Orientador: Me. Marcondes Maçaneiro

**RIO DO SUL
2018**

**CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO
ITAJAÍ - UNIDAVI**

VINÍCIUS DALASTRA

**PROTÓTIPO DE SISTEMA PARA DETECÇÃO E IDENTIFICAÇÃO DE PESSOAS
EM MULTIDÕES ATRAVÉS SOFTWARE DE RECONHECIMENTO BIOMETRICO
FACIAL**

Trabalho de Conclusão de Curso a ser apresentado ao curso de Sistemas da Informação, da Área das Ciências Naturais, da Computação e das Engenharias do Centro Universitário para o Desenvolvimento do Alto Vale do Itajaí - UNIDAVI, a ser apreciado pela Banca Examinadora, formada por:

Professor Orientador: Me. Marcondes Maçaneiro

Banca Examinadora:

Prof.

Prof.

Rio do Sul, 22 de novembro de 2018.

Dedico o presente trabalho à todas as pessoas que se sempre estão empenhas no desenvolvimento e crescimento tecnológico em prol de uma sociedade mais justa e comprometida com a evolução e melhoria.

AGRADECIMENTOS

Agradeço primeira a Deus, que ilumina e guia a minha trajetória, possibilitando que eu enfrente todas as dificuldades sem me abalar.

Ao meu pai Claudemir Dalastra, minha mãe Marlene e meus irmãos Carla e Júlio César, que prestaram apoio em toda minha caminhada, dando força e também apoio nas dificuldades e fortalecendo meu caráter.

Aos professores que foram o instrumento mais importante na construção e na moldagem do meu conhecimento técnico, em especial pelo professor orientador Marcondes Maçaneiro que não mediu esforços para o auxílio e desenvolvimento deste trabalho.

A todos que de alguma forma auxiliaram, acreditando na minha capacidade e nas minhas ideias, quando muitos duvidaram e falaram que eu não conseguiria.

RESUMO

O presente trabalho propõe avaliar tecnicamente a possibilidade da criação de um protótipo de sistema para detecção e identificação de pessoas em multidões através de *software* para reconhecimento biométrico facial. Sua aplicação é voltada para a efetivação de detecção, identificação e registro de pessoas em locais que possuem um grande fluxo de pessoas. Sabendo que a identificação de pessoas não é um problema para os seres humanos, para as máquinas, porém é um processo complexo e impreciso necessitando de alto poder computacional para obter-se resultados precisos e eficazes. A estrutura aqui proposta foi testada e avaliada com base no uso de uma aplicação real considerando a realização de um experimento aplicado a um grupo de pessoas. Entre as ferramentas utilizadas no desenvolvimento considera-se o uso da linguagem *CSharp* com a ferramenta Visual Studio 2017, integrando o uso dos frameworks Microsoft .NET, AForge.NET e da API de Detecção e Identificação Facial da Azure e para o armazenamento dos registros o banco de dados PostgreSQL. Na sessão final deste estudo apresenta-se a análise do experimento realizado, tendo como intuito a verificação da aceitação, desempenho, confiança e possíveis aplicações do protótipo de detecção e identificação facial. Os resultados obtidos no estudo demonstram que é possível desenvolver um sistema para detecção e identificação de pessoas em multidões através de *software* para reconhecimento biométrico facial. Comprovou-se também que o sistema possui aceitabilidade, confiança e desempenho elevados e que possui várias possibilidades de aplicações.

Palavras-Chave: Detecção Facial, Identificação Facial, Microsoft Azure.

ABSTRACT

The present work proposes to evaluate technically the possibility of the prototype system for the detection and identification of people in crowds through biometric facial recognition software. Its application is focused on the effective detection, identification and registration of people in places that have a large flow of people. Knowing that identifying people is not a problem for humans, for machines, but it is a complex and imprecise process requiring high computational power to obtain accurate and efficient results. The structure proposed here was tested and based on the use of an actual application considering the realization of an experiment applied to a group of people. Among the tools used in the development is the use of the CSharp language with the Visual Studio 2017 tool, integrating the use of the Microsoft .NET, AForge .NET and Azure Facial Detection and Identification for the records of the PostgreSQL database. In the final session of this study we present the analysis of the experiment carried out, with the purpose of verifying the acceptance, performance, confidence and possible applications of the prototype of detection and facial identification. The results obtained in the study demonstrate that it is possible to develop a system for detecting and identifying people in crowds through software for facial biometric recognition. It has also been proven that the system has high acceptability, reliability and performance and that it has several applications possibilities.

Keywords: Facial Detection, Facial Identification, Microsoft Azure.

LISTA DE ILUSTRAÇÕES

Figura 1- Software De Identificação Biométrica Digital.....	32
Figura 2 - Mapeando Os Pontos Característicos	33
Figura 3 - Vinculando Face E Id	34
Figura 4 - Identificação Facial.....	35
Figura 5 - Interface Principal Do Software <i>VoiceAnalysis</i>	36
Figura 6 - Interface De Análise E Processamento Do Software <i>VoiceAnalysis</i>	36
Figura 7 - Sistema Óptico Experimental	37
Figura 8 - Código De Uma Íris Com Canais, Em Cima Nir E R Em Baixo G E B Respectivamente.....	38
Figura 9 - Diagrama de Atividades (Cadastro Identificador de Pessoa)	40
Figura 10 - Diagrama de Atividades (Treinamento da <i>API</i> de Face)	41
Figura 11 - Diagrama de Atividades (Detecção e Identificação de Pessoas Através de Imagem Facial)	42
Figura 12 - Modelo de Dados	43
Figura 13 - Interface Principal de Chamada das Rotinas de Consulta e Treinamento da <i>API</i> (Interface Redimensionada).....	44
Figura 14 - Método da Inicialização da Tela Principal e da Conexão com o Banco de Dados	45
Figura 15 - Método de Chamada do Treinamento da <i>API</i>	45
Figura 16 - Método de Fechamento da Conexão com o Banco de Dados da Tela Principal ...	45
Figura 17 - Interface de Chamada da Consulta de Pessoas (Interface Redimensionada).....	46
Figura 18 - Interface de Cadastro de Pessoa (Interface Redimensionada)	47
Figura 19 - Interface de Consulta do Relacionamento Pessoa com Identificador (Interface Redimensionada)	48
Figura 20 - Interface de Cadastro do Relacionamento Pessoa com Identificador (Interface Redimensionada)	49
Figura 21 - Método de Chamada da Adição de Face no Identificador de Pessoa	50
Figura 22 - Método de Chamada da Exclusão de Identificador da <i>API</i>	51
Figura 23 - Interface de LOGS de Reconhecimentos Realizados (Interface Redimensionada)	51
Figura 24 - Interface de Cadastro de LOG de Reconhecimento (Interface Redimensionada) .	52
Figura 25 - Botões da Interface do Reconhecedor (Interface Redimensionada).....	53
Figura 26 - Interface do Reconhecedor Detectando e Identificando Faces (Interface Redimensionada)	53
Figura 27 - Método de Chamada da Detecção e Identificação da Face (Interface Redimensionada)	54
Figura 28 - Atributos para Utilização da Câmera.....	55
Figura 29 - Carregamento dos Dispositivos de Entrada de Vídeo Disponíveis	55
Figura 30 - Inicialização e Finalização da Câmera.....	56
Figura 31 - Corpo da Requisição <i>PersonGroupId</i>	57
Figura 32 - Métodos para Montagem das Requisições	58
Figura 33 - Métodos de comunicação com <i>API</i> para Criação do Identificador de Pessoa.....	59
Figura 34 - Métodos de comunicação com <i>API</i> para Remoção do Identificador de Pessoa	60
Figura 35 - Métodos de comunicação com <i>API</i> para Adição de Faces na Pessoa	61
Figura 36 - Métodos de comunicação com <i>API</i> para Treinamento da <i>API</i>	62
Figura 37 - Métodos de comunicação com <i>API</i> para Detecção de Faces.....	63
Figura 38 - Métodos da <i>URI</i> para Identificação das Faces.....	64

Figura 39 - Corpo da Requisição de Identificação	64
Figura 40 - Retorno da Requisição de Identificação Facial.....	65
Figura 41 - Formatação da Imagem da Face	65
Figura 42 - Container do Histórico 1	66
Figura 43- Diagrama de Atividade do Escopo De Teste Do Protótipo	67
Gráfico 1 - Primeira Questão (Grau De Instrução).....	68
Gráfico 2 - Segunda Questão (Formação Relacionada A Tecnologia)	69
Gráfico 3 - Terceira Questão (Desempenho).....	70
Gráfico 4 - Quarta Questão (Precisão Do Algoritmo).....	70
Gráfico 5 - Quinta Questão (Interferência Do Usuário)	71
Gráfico 6 - Sexta Questão (Confiabilidade Do Protótipo)	72
Gráfico 7 - Sétima Questão (Privacidade Do Usuário)	73
Gráfico 8 - Oitava Questão (Aplicabilidade Do Protótipo No Dia-A-Dia).....	74
Gráfico 9 - Nona Questão (Biometrias De Fácil Usabilidade).....	74
Gráfico 10 - Décima Questão (Probabilidade De Usabilidade Do Protótipo No Comércio Ou Empresa).....	75
Gráfico 11 - Décima Primeira Questão (Probabilidade De Usabilidade Do Protótipo Em Eventos).....	76
Gráfico 12 - Décima Segunda Questão (Sentimento De Segurança Com Uso De Reconhecimento Facial Em Eventos).....	76
Gráfico 13 - Décima Terceira Questão (Inibição De Delitos Em Eventos Que Possuem Monitoramento De Pessoas Via Software).....	77
Gráfico 14 - Décima Quarta Questão (Presença em Eventos Com Identificação De Pessoas Através De Reconhecimento Facial)	78
Gráfico 15 - Décima Quinta Questão (Identificação De Todas As Pessoas Em Grandes Eventos).....	78
Gráfico 16 - Décima Sexta Questão (Auxílio Na Identificação De Pessoas Por Sistemas de Identificação).....	79

LISTA DE QUADROS

Quadro 1 - Funções Disponíveis na FACE API da Microsoft Azure.....	28
--	----

SUMÁRIO

1. INTRODUÇÃO	12
1.1 OBJETIVOS	13
1.2.1 Geral	13
1.2.2 Específicos	13
2. REVISÃO DA LITERATURA	15
2.1 INTELIGÊNCIA ARTIFICIAL	15
2.1.1 Redes Neurais.....	16
2.1.1.1 Processos de Aprendizado	18
2.2 PROCESSAMENTO DE IMAGENS	19
2.2.1 Visão Humana.....	19
2.2.2 Processamento de Imagens em Sistemas de Visão Computacional	20
2.3 BIOMETRIA	23
2.4 RECONHECIMENTO FACIAL.....	23
2.4.1 Sistemas de Reconhecimento Facial.....	24
2.5 AZURE.....	26
2.5.1 Serviços cognitivos.....	27
2.5.1.1 Face API	28
2.6 POSTGRESQL.....	28
3. METODOLOGIA	30
4. PROTÓTIPO DE SISTEMA PARA DETECÇÃO E IDENTIFICAÇÃO DE PESSOAS ATRAVÉS DE BIOMETRIA FACIAL	31
4.1 ESTADO DA ARTE	31
4.2 PROJETO DO PROTÓTIPO	39
4.2.1 Diagrama de Atividades.....	39

4.2.2 Diagrama de Modelo	42
4.3 DESENVOLVIMENTO DAS INTERFACES DO PROTÓTIPO.....	43
4.3.1 Interface de Chamada das Consultas e Treinamento	44
4.3.2 Interfaces de Pessoas	46
4.3.3 Interfaces de Identificador.....	47
4.3.4 Interfaces de Reconhecimento	51
4.3.5 Interface do Reconhecedor	52
4.4 DESENVOLVIMENTO DO PROCESSO DE CAPTURA DAS IMAGENS DOS DISPOSITIVOS E PROCESSOS DE COMUNICAÇÃO COM A API	54
4.4.1 Captura de Imagens dos Dispositivos	55
4.4.2 Processos de Comunicação com a API	56
4.4.2.1 Criação do PersonGroup.....	57
4.4.2.2 Geração e Remoção do Identificador de Pessoa.....	57
4.4.2.3 Adição de Faces no Identificador de Pessoa	60
4.4.2.4 Treinamento da API.....	61
4.4.2.5 Detecção Facial.....	62
4.4.2.6 Identificação de Faces Detectadas	63
4.5 EXPERIMENTO DE AVALIAÇÃO DO PROTÓTIPO DE DETECÇÃO E IDENTIFICAÇÃO DE PESSOAS ATRAVÉS DE BIOMETRIA FACIAL	67
4.6 RESULTADOS OBTIDOS	68
5. CONCLUSÕES.....	80
5.1 TRABALHOS FUTUROS	81
REFERÊNCIAS	82

1. INTRODUÇÃO

Em diversas circunstâncias, é essencial a identificação e registro de pessoas, seja para o controle de acessos, controles de segurança ou apenas, para registro de informações. O protótipo proposto neste trabalho tem o intuito de possibilitar a identificação de pessoas utilizando para tal algoritmos de reconhecimento facial.

Um exemplo de aplicação de soluções como a aqui proposta possibilitaria por exemplo o reconhecimento de pessoas em grandes eventos com grande quantidade de pessoas. As equipes de organização e segurança tem dificuldades de detectar e identificar com precisão as pessoas na multidão. Sendo que muitas vezes estas equipes não têm capacidade técnica e suporte profissional para gerir e controlar o fluxo de indivíduos em determinado ambiente. Assim, o apoio de sistemas automatizados de detecção e identificação de pessoas são extremamente importantes, uma vez que além de facilitar o trabalho desenvolvido pelas equipes, auxilia na precisão de identificação e registro das pessoas que transitam em determinado evento ou ambiente.

Sabe-se que a identificação de pessoas não é um problema para o homem, porém para as máquinas muitas vezes é um processo complexo e impreciso, devido as grandes variáveis relacionadas ao processo de identificação. A visão computacional aliada a um processo de biometria facial, acaba muitas vezes gerando a necessidade de um alto poder computacional devido a necessidade de aplicação de inteligência artificial e redes neurais.

Desta forma o protótipo implementado neste trabalho, avalia a utilização de uma *API* especializada na detecção e identificação de pessoas através da biometria facial em imagens capturadas por dispositivos de vídeo conectadas em um computador.

Abordando a utilização da *API* de Face do Azure, para os processos de criação do grupo de pessoas, criação do identificador de pessoa, relacionamento de dados faciais no identificador de pessoa, treinamento da *API* para detecção e identificação, realização de detecções faciais e identificações faciais de pessoas.

Com o intuito de validar a aceitação, desempenho, confiança e aplicabilidade do protótipo foi realizado um experimento juntamente com a aplicação de formulário a determinado grupo de pessoas.

1.1 PROBLEMATIZAÇÃO

O problema de identificar e registrar as pessoas vem causando adversidades que se tornam mais notáveis quando nos referimos a grandes multidões. Sabendo que os eventos de grande porte se popularizaram nos últimos anos, acompanhados de um crescimento exponencial do número de participantes. Expandindo do mesmo modo complexidade de identificar e registrar os participantes pelos organizadores e pelas equipes de segurança dos eventos.

Diante dos fatos apresentados anteriormente, neste trabalho pretende-se avaliar se é tecnicamente possível identificar pessoas em grupos por meio de um *software* de computador.

1.2 OBJETIVOS

1.2.1 Geral

- Implementar um protótipo que disponha de visão computacional para realizar a detecção, identificação e registro de pessoas em locais com grande fluxo de pessoas.

1.2.2 Específicos

- Implementar um protótipo de sistema para realizar a detecção e identificação de pessoas em imagens capturadas utilizando de visão computacional.
- Relatar as tecnologias utilizadas para a implementação de um protótipo de visão computacional para reconhecimento de pessoas em imagens através da face.
- Apresentar o *log* de identificação de pessoas que foram identificadas pelo protótipo de reconhecimento facial com visão computacional.
- Definir as ferramentas e as tecnologias utilizadas para o desenvolvimento do protótipo.

- Realizar a avaliação do protótipo tendo como intuito a verificação da aceitação, desempenho, confiança e possíveis aplicações do protótipo de detecção e identificação facial.

1.3 JUSTIFICATIVA

Mesmo sendo de conhecimento geral que algoritmos utilizados para a identificação de pessoas através de imagens faciais capturadas por visão computacional, tem um grau de precisão inferior comparadas a outras tecnologias, devido as capturas possuírem influência externa como variação de iluminação, expressões faciais, acessórios e posicionamento de captura, é uma tecnologia que não necessita de interação do usuário, apenas sensores para monitoramento do ambiente.

Atualmente os organizadores e equipes de segurança de eventos não conseguem realizar de forma completa a identificação e registro dos participantes quando o evento possui muitas pessoas e quando realizam necessitam de uma grande quantidade de mão de obra, aumentando o custo do evento.

A implantação de um protótipo que realiza a identificação e registro de pessoas em eventos seria vantajoso pois poderia reduzir os custos com mão-de-obra e gerar informações de forma automatizada, como um *log* com dados de data e hora e um identificador para cada participante e com devidas adaptações realizar a identificação de emoções e acessórios que estão sendo utilizados pelos participantes do evento.

2. REVISÃO DA LITERATURA

A Revisão da Literatura tem como intuito realizar a apresentação do referencial teórico que possui relevância para o estudo realizado, com base em conceitos e estudos já realizado na área ou áreas paralelas do estudo.

2.1 INTELIGÊNCIA ARTIFICIAL

Para Coppin (2012, p.3) a “Inteligência Artificial é o estudo dos sistemas que agem de um modo que a um observador qualquer pareceria ser inteligente.”. Porém como o autor descreve, a frase não abrange como um todo da Inteligência Artificial, desta forma Coppin (2012, p.3) utiliza para complementar a primeira citação a seguinte frase: “Inteligência Artificial envolve utilizar métodos baseados no comportamento inteligente de humanos e outros animais para solucionar problemas complexos”.

De acordo com Magnus (2018) a Inteligência Artificial é a seção da informática que tem como objetivo pressuposto a criação de máquinas inteligentes, que através da programação de computadores cria-se alguns traços, dentre eles estão capacidade de conhecimento, raciocínio, solução de problemas, percepção, aprendizagem, planejamento e até a capacidade de manipular e movimentar objetos. Desta forma as máquinas inteligentes conseguem optar de forma autônoma as melhores decisões a serem tomadas conforme as opções que lhe foram pré-estabelecidas.

A inteligência artificial é um ramo de pesquisa da ciência da computação que busca, através de símbolos computacionais, construir mecanismos e/ou dispositivos que simulem a capacidade do ser humano de pensar, resolver problemas, ou seja, de ser inteligente. O estudo e desenvolvimento desse ramo de pesquisa tiveram início na Segunda Guerra Mundial. (SANTOS, 2018)

Segundo Santos (2018) junto com a ascensão da computação houve avanços também na inteligência artificial, possibilitando uma grande evolução na análise computacional. No início dos estudos sobre inteligência artificial, tinha como objetivo reproduzir a capacidade humana de pensar, mas foi notado que haviam outros aspectos que podiam ser descobertos, os cientistas optaram por fazer que as máquinas tivessem capacidade de sentir, ter criatividade, auto aperfeiçoar e utilizar a linguagem humana.

2.1.1 Redes Neurais

Conforme Luger (2013) os modelos neurais ou também conhecidos como sistemas conexionistas ou processamento paralelo distribuído, não se destacam pela utilização de símbolos para resolução de problemas, mas através de processos de aprendizado ou de adaptação pela qual as conexões entre os componentes são ajustadas de forma inteligente.

Segundo Tatibana e Ketsu (2018) as redes neurais artificiais resumem-se em uma metodologia utilizada para solucionar problemas de inteligência artificial, confeccionando um sistema que simule o cérebro humano através de circuitos, simulando o comportamento do cérebro humano, assim tendo a capacidade de aprender errar e fazer descobertas, através da experiência.

De acordo com Coppin (2012) as redes de neurônios artificiais têm como base o cérebro humano e em sua composição apresentam neurônios artificiais, possuindo uma quantidade numérica inferior as do cérebro humano.

“Redes Neurais Artificiais são técnicas computacionais que apresentam um modelo matemático inspirado na estrutura neural de organismos inteligentes e que adquirem conhecimento através da experiência.”. Conforme é descrito por Carvalho (2018).

Luger (2013) define que o modelo neural utiliza padrões de um domínio cujo o mesmo é codificado como vetor numérico e as conexões entre os componentes ocorrem através de valores numéricos, gerando durante as transformações de padrões resultados de operações numéricas, na maioria dos casos multiplicação de matrizes.

Assim como o sistema nervoso é composto por bilhões de células nervosas, a rede neural artificial também seria formada por unidades que nada mais são que pequenos módulos que simulam o funcionamento de um neurônio. Estes módulos devem funcionar de acordo com os elementos em que foram inspirados, recebendo e retransmitindo informações. (TATIBANA; KETSU, 2018)

Segundo Coppin (2012) cada um dos neurônios é submetido a diversas entradas, a função de ativação é aplicada sobre os valores de entrada resultando em um nível de ativação do neurônio, sendo este o valor de saída.

Coppin (2012) ainda descreve que o *Perceptron* foi proposto por Rosenblatt no ano de 1958 e é um neurônio usado para realizar classificação das entradas em duas categorias. As

entradas são organizadas em forma de grade, que são utilizadas na classificação de imagens ou tarefas de reconhecimento simples.

Com base em Maia (2016) o *Perceptron* é uma reprodução computacional da menor unidade neural artificial, dispõe de uma arquitetura no formato entrada, processamento e saída.

Conforme Coppin (2012, p.258) “Um único *perceptron* pode ser usado para aprender uma tarefa de classificação, na qual ele recebe uma entrada e a classifica em uma de duas categorias: 1 ou 0.”.

Luger (2013, p.379) descreve que “Após tentar resolver uma ocorrência do problema, um professor fornece a ele o resultado correto. O *perceptron* modifica, então, seus pesos de modo a reduzir o erro.”.

Tatibana e Ketsu (2018) descrevem que os neurônios estão ligados por meio vínculos sinápticos e classificados em neurônios de entrada, estes recebem as informações do meio externo, neurônios internos que também são conhecidos como *hidden* ou ocultos e os neurônios de saída, que tem a função de se comunicar com o exterior. *Multilayer Perceptron* é o nome dado a arranjar os *Perceptrons* em camadas, o modelo *Multilayer Perceptron* foi criado com o intuito de resolver problemas que pelo uso do neurônio básico não seria possível. Os neurônios internos possuem grande importância na rede neural pois o uso dos mesmos é imprescindível para resolver problemas não separáveis linearmente.

Conforme Carvalho (2018) a rede neural artificial é composta por várias unidades de processamento que possuem um funcionamento muito simples. As unidades são interconectadas através de canais de comunicação que possuem alusão a um determinado peso, desta forma a unidade posteriormente a receber uma entrada pelo seu canal de comunicação realiza as devidas operações sobre essa entrada.

Arquiteturas neurais são tipicamente organizadas em camadas, com unidades que podem estar conectadas às unidades da camada posterior.

Usualmente as camadas são classificadas em três grupos:

- Camada de Entrada: onde os padrões são apresentados à rede;
- Camadas Intermediárias ou Escondidas: onde é feita a maior parte do processamento, através das conexões ponderadas; podem ser consideradas como extratoras de características;
- Camada de Saída: onde o resultado final é concluído e apresentado.

(CARVALHO, 2018)

No entender de Tatibana e Ketsu (2018) a rede neural passa por aprendizado utilizando como base casos reais conhecidos, obtendo desta forma a sistemática primordial para executar

de forma adequada o processo desejado com base nos dados fornecidos. Desta forma a rede neural consegue reproduzir as regras básicas, diferenciando-se da computação programada, onde são utilizadas regras prefixadas e algoritmos.

2.1.1.1 Processos de Aprendizado

De acordo com Luger (2013, p. 319) as pessoas ao serem questionadas sobre quais são as habilidades humanas mais complexas de serem computadorizadas a maioria menciona a linguagem e o aprendizado, tendo como razão para a complexidade a necessidade de englobar várias habilidades inteligentes humanas, como linguagem natural, raciocínio automático e o aprendizado de máquina.

Conforme Tatibana e Ketsu (2018) a habilidade de aprender é a característica mais importante das redes neurais, essa característica possibilita a melhoria no desempenho, sendo feito através de ajustes dinâmicos e iterativos em seus pesos, somente ocorre o aprendizado quando a rede neural ascender a uma solução generalizada para uma classe de problemas.

Do ponto de vista de Luger (2013, p.322) “O aprendizado envolve a generalização a partir da experiência: o desempenho deve melhorar não apenas na “repetição da mesma tarefa”, mas também tarefas semelhantes no domínio.”.

Carvalho (2018) define que um Algoritmo de Aprendizado é composto por um grupo de regras bem definidas para a explicação de um problema de aprendizado, os algoritmos diferenciam-se entre si pela forma como os pesos são alterados.

Outro fator importante é a maneira pela qual uma rede neural se relaciona com o ambiente. Nesse contexto existem os seguintes paradigmas de aprendizado:
Aprendizado Supervisionado, quando é utilizado um agente externo que indica à rede a resposta desejada para o padrão de entrada;
Aprendizado Não Supervisionado (auto-organização), quando não existe um agente externo indicando a resposta desejada para os padrões de entrada;
Reforço, quando um crítico externo avalia a resposta fornecida pela rede.
(CARVALHO, 2018).

Para Tatibana e Ketsu (2018) a correção de pesos que ocorre em um ciclo pode ser executada em dois métodos, o Padrão, onde acontece a correção dos pesos a cada comunicação de conjunto de treinamento à rede e no método *Batch* ocorre apenas uma correção por ciclo, onde é buscado o erro médio do conjunto de treinamento que se

comunicou a rede, através de cálculos e daí em diante realiza-se as correções necessárias nos pesos.

2.2 PROCESSAMENTO DE IMAGENS

Coppin (2012) diz que a percepção de um mundo visual em sistemas computacionais, robôs ou agentes é muito desejável.

O sistema de visão em mamíferos (tais como seres humanos) é um dos sistemas mais extraordinários do mundo natural. Sem visão, seria possível argumentar que seres humanos não teriam chegado aos níveis atuais de avanço tecnológico e, na verdade, que nenhum dos seres vivos atualmente teria sido capaz de evoluir com êxito sem visão. (COPPIN, 2012, p.525)

De acordo com Russell e Norvig (2013) observações visuais são impressionantemente detalhadas, fazendo que os sensores tenham que ignorar alguns aspectos, para isso estão dispostas três abordagens. A extração de características, que através de alguns cálculos simples efetuados sobre as informações obtidas através de sensores fornece dados sem maiores detalhes do objeto, o reconhecimento que é a verificação de individualidades entre as informações visuais e a reconstrução que é o desenvolvimento de um sistema geométrico do mundo, partindo das informações visuais captadas pelos sensores.

2.2.1 Visão Humana

Para Coppin (2012) compreender como os seres humanos veem é primordial para promover a capacidade de visão a computadores. Partindo deste entendimento define-se como partes essenciais para a visão do ser humano os olhos e o cérebro, principalmente a parte do córtex visual, pois essa parte está associada diretamente a capacidade de visão do ser humano.

No olho, a imagem é formada na retina, que consiste em dois tipos de células: cerca de 100 milhões de bastonetes, que são sensíveis à luz em ampla gama de comprimento de onda, e 5 milhões de cones. Cones, que são essenciais para a visão das cores, são de três tipos principais, cada um dos quais é sensível a um conjunto diferente de comprimento de onda. (RUSSEL; NORVIG, 2013, p.810)

De acordo com Coppin (2012) os bastonetes são altamente sensíveis, tendo uma boa resposta quando existe baixa taxa de iluminação, porém eles possuem baixa acuidade, transmitindo para o cérebro imagens incolores, menos detalhadas e mais “nebulosas” do que as enviadas pelos cones, os bastonetes são encontrados principalmente em torno das bordas da retina. Os cones são praticamente insensíveis, desta forma possuem uma boa resposta quando expostos a alta taxa de iluminação, eles possuem alta acuidade e capacidade de definir cores e ficam situados mais próximos ao centro da retina.

Coppin (2012) ainda complementa que os sinais que são captados pela retina são enviados para o núcleo lateral geniculado e ao colículo superior através do nervo ótico, os sinais captados pelo olho esquerdo são enviados ao lado direito do cérebro e do olho esquerdo para o lado direito do cérebro, cruzando-se no Quiasma Ótico e seguindo para o córtex visual através das radiações óticas.

2.2.2 Processamento de Imagens em Sistemas de Visão Computacional

De acordo com Coppin (2012) o processo de reconhecer imagens através de sistemas de visão computacional pode ser desdobrado nos estágios:

Captura de Imagem;

- Detecção de Bordas;
- Segmentação;
- Segmentação Tridimensional; e
- Reconhecimento e Análise.

Captura de Imagem pode ser realizada por uma simples câmera (ou par de câmeras, para ter visão estereoscópica) que converta sinais de luz, a partir de uma cena, em sinais eletrônicos, de forma bem semelhante a que faz o sistema visual humano. (COPPIN, 2012, p.528)

Segundo Russell e Norvig (2013) as arestas são linhas retas ou curvas presentes no plano da imagem, que ao longo das mesmas existe uma taxa de variação significativa no brilho da imagem. O objetivo da detecção de bordas é abstrair a imagem para uma reprodução mais compacta e abstrata, demonstrando os contornos importantes da cena.

Para Coppin (2012, p.529) “Como quase todos os objetos no mundo real têm bordas sólidas de um tipo ou de outro, detectar estas imagens é a primeira etapa no processo de determinar quais objetos estão presentes em uma cena.”.

A medição do brilho de um *pixel* em uma câmera CCD é baseada em um processo físico que envolve a absorção de fótons e a liberação de elétrons; há flutuações estatísticas de medição – ruído. O ruído pode ser modelado com uma distribuição de probabilidade gaussiana, com cada *pixel* independente dos outros. Uma maneira de suavizar uma imagem é atribuir a cada *pixel* a média de seus vizinhos. Isso tende a cancelar valores extremos. (RUSSELL; NORVIG, 2013, P.816)

Segundo Coppin (2012) as imagens reais contêm muito ruído, o método da diferenciação que se baseia na análise de cores buscando diferenciais não é muito favorável pois produz diferenciais aonde não ocorre o aparecimento de bordas, para isso foi desenvolvido o método de convolução, que utiliza de duas funções discretas para eliminar os efeitos de ruído realizando uma suavização da imagem. Após a realização da suavização *pixels* que são considerados como borda são ligados a *pixels* adjacentes com o objetivo de definir as formas e localização das bordas.

Russel e Norvig (2013, p.818) “Uma vez que tenhamos marcado os *pixels* da aresta por esse algoritmo, a próxima etapa é ligar esses *pixels* que pertencem às mesmas curvas da Aresta.”.

De acordo com Coppin (2012) após a detecção das bordas, pode-se através da informação resultante realizar a segmentação nas áreas homogêneas da imagem, definido que a área homogênea é uma área com pouca variação de cor ou intensidade de sombreado.

Russell e Norvig (2013) afirmam que a segmentação é o procedimento de desincorporar em *pixels* semelhantes uma área de uma imagem, sendo que cada *pixel* é capaz de vincular a certos atributos visuais, como brilho, cor e textura. Sendo que nas extremidades dos objetos ocorre uma variação grande de algum dos atributos, e no interior dos objetos a variação é um tanto quanto baixa.

Para Coppin (2012, p.532) um método simples de segmentar uma imagem é a “Imposição de limite envolve encontrar a cor de cada *pixel* em uma imagem e depois considerar *pixels* adjacentes como sendo da mesma área, desde que as cores deles sejam parecidas o suficiente.”.

Segundo a análise de Russel e Norvig (2013, p.821) “Não se pode esperar da segmentação baseada puramente em atributos locais de baixo nível, como brilho e cor, que entregue os limites finais corretos de todos os objetos na cena.”.

No entender de Coppin (2012) existe um método para segmentação que é conhecido por separação e junção. Separar é utilizar uma área não homogênea e separa-la em duas ou mais áreas menores, cada uma delas sendo homogêneas. Juntar é basicamente utilizar duas áreas que sejam idênticas e adjacentes e combina-las em uma área maior. Esse formato de segmentação iterativa é frequentemente mais confiável que a simples imposição de limites.

Na linguagem cotidiana, textura é a sensação visual de uma superfície – o que você vê evoca o que a superfície poderia sentir se você a tocasse (“textura” tem a mesma raiz que “têxtil”). Em visão computacional, textura refere-se a um padrão que se repete especialmente em uma superfície que pode ser percebida visualmente. (RUSSEL; NORVIG, 2013, p.818)

Coppin (2012) define que a textura é uma particularidade essencial do mundo visual. A textura nos auxilia a identificar as peculiaridades do que vemos e também nos fornece informações relevantes a formação dos materiais, formas e movimentos.

Com base em Russel e Norvig (2013) o brilho é uma propriedade de *pixels* individuais, para a ideia de textura ter fundamento é essencial que seja considerado múltiplos *pixels*. Desta forma pode-se calcular a orientação em cada *pixel* caracterizando o trecho através de um histograma de orientações. Porém em imagens de artefatos texturizados a detecção de arestas pode ter um desempenho inferior que em objetos lisos, isso ocorre devido as arestas mais importantes estarem misturadas entre os elementos da textura e não seja possível identificá-los.

Coppin (2012) descreve que um dos aspectos mais importantes presentes na visão dos mamíferos é a aptidão de detectar e reagir a movimentos. Com a grande quantidade de informação visual confusa, os animais tendem a utilizar o movimento para obter mais informações sobre o que está sendo observado. Desta forma para um sistema de reconhecimento de imagens é de suma importância a capacidade de detecção de movimentos.

De acordo com Russel e Norvig (2013, p.819) “Quando um objeto no vídeo está se movendo ou quando a câmera se move em relação a um objeto, o movimento aparente resultante na imagem é chamado de fluxo óptico.”.

É possível estimar a natureza do campo de movimento e, assim, o fluxo ótico, em uma sequência de imagens comparando as características das imagens. Primeiro, assumimos que os objetos em uma sequência de imagens não mudarão por eles mesmos e, então, quaisquer mudanças que ocorram com eles serão causadas pelo movimento da câmera. Isto claramente não se aplicará se existirem objetos em movimento (e.g., carros, pessoas, animais) na imagem, mas ainda se aplicará à maioria das características da maioria das imagens e as anomalias poderão ser tratadas separadamente. (COPPIN, 2012, p.543)

Coppin (2012) ainda explica que se pode calcular os vetores de fluxo ótico computando pontos comuns de uma determinada sequência de imagens e assim determinar a velocidade de movimento da câmera, podendo também utilizar desta técnica caso a câmera esteja fixa e capturando imagens de objetos em movimento.

2.3 BIOMETRIA

Segundo Martins (2007) os seres humanos possuem características fisiológicas e comportamentais que podem ser classificadas como biometria, mas para isso devem possuir algumas propriedades, a característica deve ser comum de todas as pessoas, porém única, imutável com o tempo e podendo ser medida de forma quantitativa. Os tipos de biometria existentes estão dispostos através da voz, impressão de digitais, íris, orelha, datilografia, assinatura, emissões acústicas e pela face.

2.4 RECONHECIMENTO FACIAL

Conforme defendido por Faria (2018) um dos processos mais utilizados para identificar pessoas é o reconhecimento facial, que permite além de identificar uma pessoa de forma rápida ainda possa definir o emocional da pessoa através de suas expressões faciais.

“O problema de análise da identidade de uma pessoa pode ser dividido em dois tipos distintos de problemas com diferentes complexidades: (i) verificação e (ii) o reconhecimento (mais conhecida popularmente como identificação).” (MARTINS, 2007).

No entender de Martins (2007) a verificação é basicamente confirmar se uma pessoa é de fato quem ela diz ser e a identificação é definir quem é a pessoa, gerando desta forma um percentual de certeza da identidade da pessoa.

2.4.1 Sistemas de Reconhecimento Facial

Conforme Faria (2018) descreve, existem diversas técnicas que podem ser usadas no desenvolvimento de um *software* de reconhecimento facial, que a partir de uma foto ou vídeo consiga realizar a extração da imagem facial da pessoa e realizar uma verificação para saber se essa pessoa pertence ou não a uma base de dados com imagens faciais previamente definidas.

Tradicionalmente, o problema de reconhecimento de faces por computador é realizado em três etapas:

(1) Detecção/segmentação das faces. (2) Extração de características das faces. (3) Identificação e/ou verificação das faces. Algumas vezes estas etapas não são completamente separadas. É possível, por exemplo, realizar a extração de características ao mesmo tempo em que se detecta uma face. (ALBERGARIA; SANTOS; ALVIM JÚNIOR, 2006)

No entender de Albergaria, Santos e Alvim Júnior (2006), o reconhecimento de faces é baseado na conciliação de análises holísticas e de características específicas, na primeira fase analisa-se o aspecto geral do rosto, fazendo assim que faces fora do padrão sejam identificadas com maior facilidade que faces comuns.

A eficiência de um sistema de reconhecimento facial pode ser medida pela porcentagem de acertos na identificação de pessoas sobre essas condições. Os sistemas de reconhecimento facial que existem atualmente não são seguros o suficiente para serem aplicados em larga escala, mas muitos avanços foram obtidos. (FARIA, 2018)

“É muito desafiador desenvolver uma técnica de reconhecimento de rosto que pode tolerar os efeitos do envelhecimento, expressões faciais, ligeiras variações na imagiologia ambiente e a posição do rosto em relação à câmera”. (MARTINS, 2018)

De acordo com Russel e Norvig (2013) o rosto por ser redondo e bastante brilhante destaca-se quando comparado com as órbitas dos olhos que são escuras por estarem submersas na face e da boca e sobrancelha que são caracterizadas por ser um talho escuro. Desta forma quando ocorrem grandes variações de iluminação pode causar mudanças no padrão, mas o intervalo de mudança é controlável.

As dificuldades com reconhecimento automático de face são numerosas. Em primeiro lugar, as condições nas quais uma face pode ser vista, como luminosidade, distância entre a câmera e face e ângulo podem alterar dramaticamente a aparência da face. Este problema é enfrentado pela maioria dos sistemas de reconhecimento de objetos. Reconhecimento facial é complicado adicionalmente pelo fato de faces humanas serem bastante flexíveis e fáceis de serem alteradas. Expressões faciais representam uma complexidade, mas pessoas também são capazes de deixar a barba crescer, cortar ou deixar o cabelo crescer, usar óculos, óculos escuros, chapéus e brincos e envelhecer, todos estes fatores podendo afetar significativamente a aparência de uma face. (COPPIN, 2012, p.545)

Segundo Albergaria, Santos e Alvim Júnior (2006), existem três formas de extrair características das faces, a extração por métodos geométricos partindo de arestas, retas e curvas, os que utilizam olhos e boca como modelo e os que partem da junção de estruturas que consideram características geométricas como limitações. Após a extração das características o reconhecimento pode ser subdividido em três grupos, métodos holísticos baseados em características e os híbridos.

“Os métodos holísticos consideram todos os *pixels* da imagem ou de regiões características da face. Nessa abordagem a dimensionalidade dos dados é igual ao número de *pixels* das imagens consideradas. Para resolver esse problema podem ser utilizados métodos estatísticos de redução de dimensionalidade[...]. Os métodos holísticos proporcionaram resultados melhores que os locais. A vantagem de abordagens globais está no fato de que pequenas variações locais não prejudicam muito o reconhecimento. A principal desvantagem está nos problemas de variação de iluminação e, em alguns casos, no custo computacional” (CAMPOS, 2000).

Albergaria Santos e Alvim Júnior (2006) descrevem que os métodos baseados em características mais satisfatórios foram baseados em *Elastic Bunch Graph Matching* também conhecido pela Sigla *EBGM*, esses sistemas aplicam-se na detecção e excisão de faces, junto a estimativa de posição, classificação sexual e reconhecimento de objetos de forma geral. O triunfo deste método se dá devido a semelhança com o sistema visual humano.

De acordo com Almeida e Bento (2018) a técnica de reconhecimento de padrões através de *Eigenfaces* se dá através da extração total das informações relevantes da imagem facial utilizada, posteriormente realiza-se a codificação da informação analisada e compara-se com faces codificadas da mesma forma, que estão presentes em uma base de dados.

Conforme Coppin (2012) o método das *Eigenfaces* ou autofaces como descreve o autor está baseado em componentes principais, tem como base que para realizar o aprendizado de reconhecimento para qualquer tipo de itens de dados a melhor forma é determinar as características que mais variam entre os dados. Se forem examinadas dez faces e observar a

posição da ponta do nariz, relativa a extremidade do queixo, é esta a característica que mais sofre alterações desta forma então ela será transformada em um componente principal e será considerada importante na detecção de faces.

Segundo Albergaria, Santos e Alvim Júnior (2006), “Métodos híbridos utilizam características tanto holísticas quanto locais. Um desses métodos é baseado em *eigenfeatures*. Este método se mostrou melhor que o baseado em *eigenfaces* para espaços de menor ordem”.

Conforme Coppin (2012, p.546) declara “As autofaces são os vetores (ou autovetores) que formam os principais componentes dos dados de treinamento e, assim, definem o espaço de face”.

Coppin (2012) ainda retrata que quando uma face é examinada ela é produzida na forma de soma de alguma combinação de autovetores e o novo vetor é comparado com os vetores construídos para as faces do treinamento, tendo como combinadora a face que estiver com o vetor similar.

2.5 AZURE

A Microsoft disponibiliza uma plataforma de infraestrutura em nuvem (Microsoft Azure), a qual possui serviços no formato de contratação de recursos com pagamento pela utilização, permitindo que sejam utilizados recursos de forma elástica e escalável, possibilitando a contratação de recursos conforme as necessidades específicas. (SANTOS; MACHADO; FIGUEIREDO. 2017. p.176)

De acordo com o TIVIT Cloud (2017) a Azure disponibiliza de forma gratuita para novos clientes, um nível de serviço para realização de experimentos e desenvolvimento de algumas aplicações, possibilitando até a criação de redes virtuais aonde só é cobrado o tráfego realizado na mesma.

O Azure é um conjunto abrangente de serviços de nuvem que os desenvolvedores e os profissionais de TI usam para criar, implantar e gerenciar aplicativos por toda a nossa rede global de datacenters. As ferramentas integradas, o *DevOps* e o Marketplace dão suporte a você para criar de maneira eficiente desde aplicativos móveis simples até soluções usadas em escala da Internet. (MICROSOFT, 2018).

BRLink (2017) declara que o Microsoft Azure foi lançado em 2010, estando distribuído em 30 regiões oferecendo 99,95% de disponibilidade mensal que em caso de reduções da disponibilidade acarreta em geração de multas.

O Azure oferece um conjunto abrangente de serviços de IA flexíveis para qualquer cenário e Infraestrutura de IA de nível corporativo que executa cargas de trabalho de IA em escala em qualquer lugar. Ferramentas de inteligência artificial modernas projetadas para desenvolvedores e cientistas de dados lhe ajudam a criar soluções de IA com facilidade e máxima produtividade. (MICROSOFT, 2018).

Conforme a Microsoft (2018) o Azure pode ser utilizado para o desenvolvimento de aplicações inteligentes e controladas por dados, partindo o reconhecimento de imagem até serviços de *BOT* através dos serviços de dados e da inteligência artificial com suporte de aprendizado profundo, simulações de HCP e análise de dados com diferentes formas e tamanho em tempo real.

2.5.1 Serviços cognitivos

Segundo a Microsoft (2018) o Azure disponibiliza serviços cognitivos com base em inteligência facial, podendo ser algoritmo de processamento de imagens que pode ser utilizado para identificar, incluir legendas e até controlar as imagens, algoritmos baseados em áudio, podendo converter áudio falado em texto, verificação e reconhecimento de voz, algoritmos de conhecimento e pesquisa que através de mapeamento e mineração de dados e informações complexas realiza tarefas como orientar decisões inteligentes e até mesmo realizar o processamento de idiomas avaliando sentimentos e reconhecer o que os usuários desejam.

A Microsoft (2018) ainda descreve que os Serviços cognitivos fazem parte da plataforma de inteligência artificial da empresa. Dentro dessa plataforma de inteligência artificial está presente a “Classificação de imagens com redes neurais convolucionais” que através do uso de redes neurais desenvolve-se um sistema de classificação de imagens com 50 camadas ocultas, treinada previamente com 350.000 imagens em um conjunto de “*imageNet*” para geração de recursos visuais por meio da remoção da última camada, que serão utilizados para treinamento de uma árvore de decisão aumentada classificando a imagem como “aprovada” ou “com falha” e a pontuação final será realizada nos computadores de borda na instalação.

2.5.1.1 Face API

Segundo a Microsoft (2018) a *API* de Face possibilita a detecção de rostos, comparação, organização em grupos com base em semelhanças e até mesmo identificar as pessoas nas imagens. As funções disponibilizadas pela *API* estão dispostas conforme o Quadro 1.

Quadro 1 - Funções Disponíveis na FACE API da Microsoft Azure

Detecção facial	Função de detectar rostos em imagens e através de características faciais como a idade, emoção, gênero, pose, sorriso, pelos faciais e outros 27 pontos de referência para cada rosto que estiver presente na imagem.
Verificação facial	Verificar a possibilidade de dois rostos pertencerem a mesma pessoa, tendo como retorno um valor entre 0 e 1 para a probabilidade de confiança de pertencimento.
Reconhecimento de emoções	Detecta através da API algumas emoções que são apresentadas como expressões interculturais e universais, retornado um valor entre 0 e 1 para a probabilidade de a face estar expressando tal emoção.
Identificação de rosto	Detectar através de pesquisa, identificação e comparação se o rosto da imagem pertence a um repositório privado com até 1 milhão de pessoas.
Pesquisa de rosto semelhante	Encontrar rostos parecidos dentro de uma coleção de imagens com uma face base para a consulta, tendo como retorno a coleção de rostos parecidos.
Agrupamento Facial	Realizar agrupamento de rostos não identificados em grupos conforme a semelhança facial presente nos grupos.

Fonte: Elaborado a partir de Microsoft (2018).

De acordo com a Microsoft (2018) “A *API* de Detecção Facial pode detectar até 64 faces humanas com localização de faces de alta precisão em uma imagem.”. Ela ainda descreve que “A *API* de Detecção Facial pode identificar pessoas com base em uma face detectada e um banco de dados de pessoas.”.

2.6 POSTGRESQL

Segundo a Dionisio (2018) “O PostgreSQL é um sistema de gerenciamento de banco de dados objeto-relacional (ORDBMS) baseado no POSTGRES versão 4.21, que foi desenvolvido na Universidade da Califórnia em Berkeley Computer Science Department.”.

De acordo com PostgreSQL (2018) o PostgreSQL é um *software* de código fonte aberto mantido pela comunidade e possui muitas funcionalidades atuais, como por exemplo chaves estrangeiras, gatilhos, visões, integridade transacional, controle de simultaneidade multiversão, tipo de dados, funções de agregação, índices, linguagens procedurais entre outros.

Ele consiste em um processo de servidor que lê e grava os arquivos de banco de dados reais, e um conjunto de programas cliente que se comunicam com o servidor. O mais comumente utilizado é o comando `psql`, que permite ao usuário executar consultas *SQL* e visualizar os seus resultados. Nenhum dos clientes acessa os arquivos do banco de dados diretamente, o que é deixado inteiramente para o servidor. (DIONISIO, 2018).

Conforme PostgreSQL (2018) “Relação é, essencialmente, um termo matemático para tabela. A noção de armazenar dados em tabelas é tão trivial hoje em dia que pode parecer totalmente óbvio, mas existem várias outras formas de organizar bancos de dados.”

Dionisio (2018) declara que as tabelas são construídas de linhas onde cada linha dispõe de um grupo de colunas nomeadas e tipificadas conforme o tipo de dado disposto nela.

3. METODOLOGIA

O método de pesquisa utilizado no presente trabalho de conclusão de curso tipifica-se como pesquisa aplicada, descritiva, que tem como propósito analisar a possibilidade de um protótipo de *software* que através de inteligência artificial tenha a capacidade de identificar e registrar as pessoas que estão presentes em um determinado ambiente de forma automática, mesmo sendo empregado frente a uma multidão de pessoas.

Referente aos procedimentos aplicados, realizou-se o uso da pesquisa documental, onde foi realizada a através do desenvolvimento do *software* de reconhecimento facial em três etapas distintas, sendo a primeira o cadastramento da face conhecida e treinamento da *API*, a segunda a captação da imagem da face e identificação das faces caso conhecidas e a terceira etapa sendo a realização da gravação das informações identificadas.

O *software* para reconhecimento facial foi desenvolvido utilizando o Microsoft Visual Studio 2017, com a linguagem de programação C#, juntamente com integração do framework Microsoft .NET, AForge.NET e da *API* de Detecção e Identificação Facial da Azure, que também foi desenvolvida pela Microsoft, tendo como pressuposto a realização da detecção e identificação das faces. Foi escolhida a *API* da Azure pela mesma disponibilizar a possibilidade de realizar a identificação com pacotes de diferentes valores, até mesmo um pacote básico com valor gratuito, porém mais limitado. Para o armazenamento dos registros utilizou-se o banco de dados PostgreSQL 10.3.

4. PROTÓTIPO DE SISTEMA PARA DETECÇÃO E IDENTIFICAÇÃO DE PESSOAS ATRAVÉS DE BIOMETRIA FACIAL

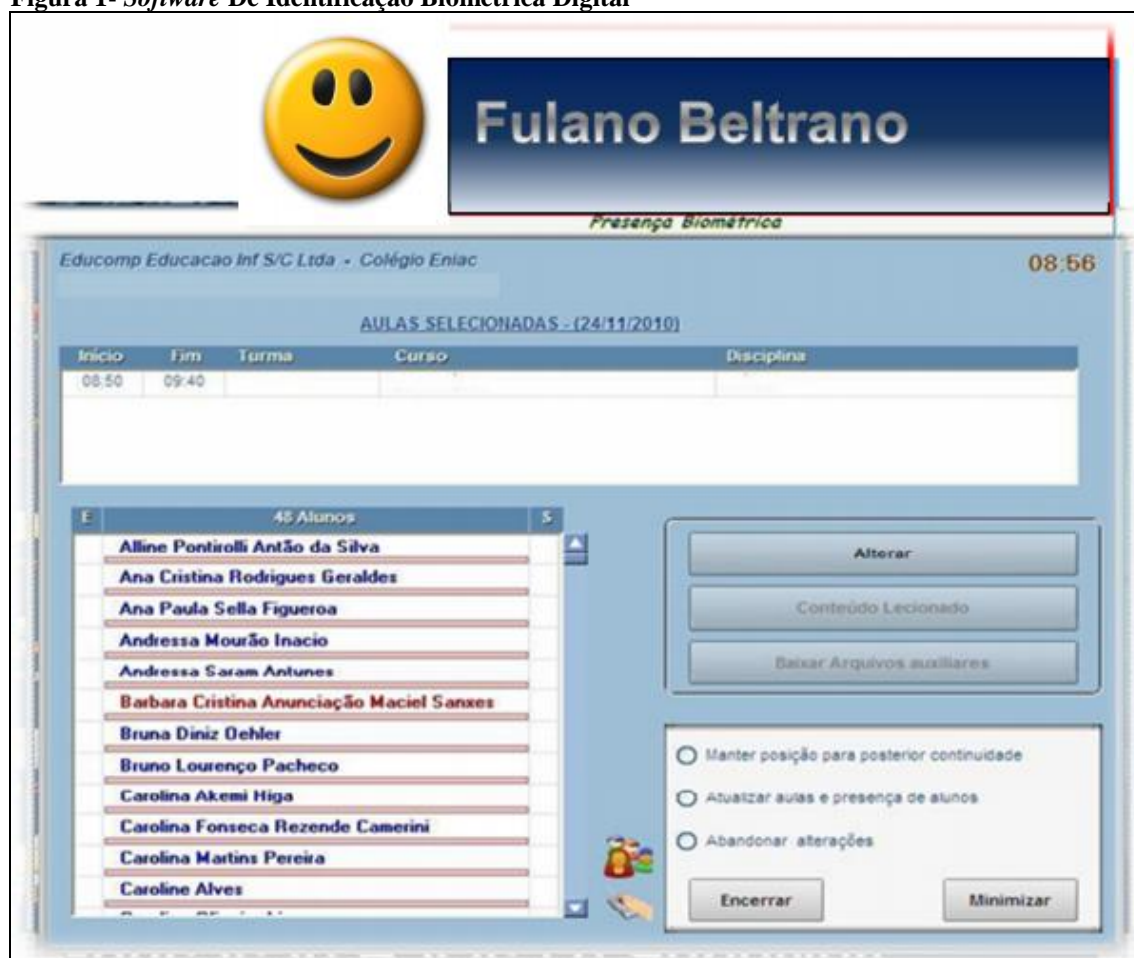
Neste capítulo, será especificado o desenvolvimento do protótipo de sistema para realizar a detecção e identificação de pessoas através do uso de biometria facial, sendo empreendido em duas etapas. A primeira caracterizou-se pela modelagem e desenvolvimento do banco de dados, desenvolvimento das interfaces de cadastro e consulta dos dados. A segunda caracterizou-se pelo desenvolvimento do processo de captura de imagens faciais através de uma câmera e nos procedimentos de comunicações realizadas na *API* da Azure, sendo elas a atribuição de um identificador gerado pela *API* a face de uma determinada pessoa, também a detecção de faces e identificação das faces detectadas na imagem capturadas pela câmera.

4.1 ESTADO DA ARTE

Atualmente foram levantadas diversas abordagens para a realização de detecção e identificação de pessoas. Há trabalhos que dispõem da análise de imagens digitalizadas originadas de scanners, fotos ou ainda de vídeos transformados em imagem e ainda outros modelos que empregam a utilização da análise do sonora.

Conforme demonstra Sanchez (2011) no desenvolvimento de um *software* que tem o pressuposto de realizar a identificação biométrica digital dos discentes de uma instituição de ensino buscando a coleta de informações referentes a frequência dos mesmos. O *software* teve que possuir um isolamento de informações para ter um desempenho superior na identificação, ou seja, antes de realizar a identificação de um discente, deve-se identificar o docente através de código, e as turmas horários de aula, para só então comparar a informação biométrica do discente, conforme demonstra a Figura 1.

Figura 1- Software De Identificação Biométrica Digital

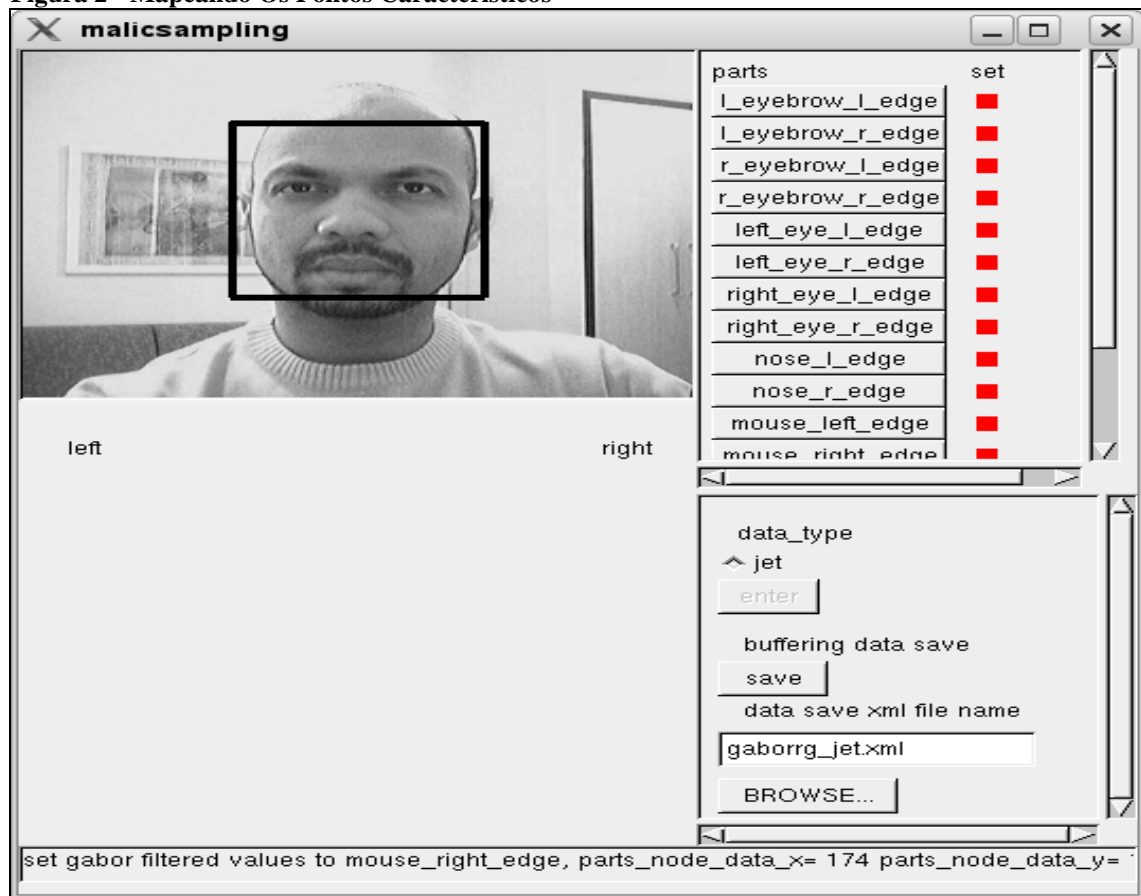


Fonte: Sanchez (2011, p.59).

Faria (2018) apresenta o projeto *Malic*, que é um protótipo de reconhecimento facial *open source* embasado nas bibliotecas *OpenCV* e *Malib* com o filtro de *Gabor*. Este trabalho tem como pressuposto realizar a identificação de pessoas através de uma câmera conectada ao computador. Neste projeto para realizar a identificação de uma pessoa é necessário realizar três etapas.

A primeira etapa apresentada por Faria (2018) é realizar o mapeamento das características da face (sobrancelha, olhos, nariz e boca), para realizar esta etapa é necessário o comando “*./malicsampling*”, abrindo uma tela conforme demonstra a Figura 2. Aonde serão informadas as características de forma manual clicando nos pontos da imagem e posteriormente salvando em um arquivo *.XML*.

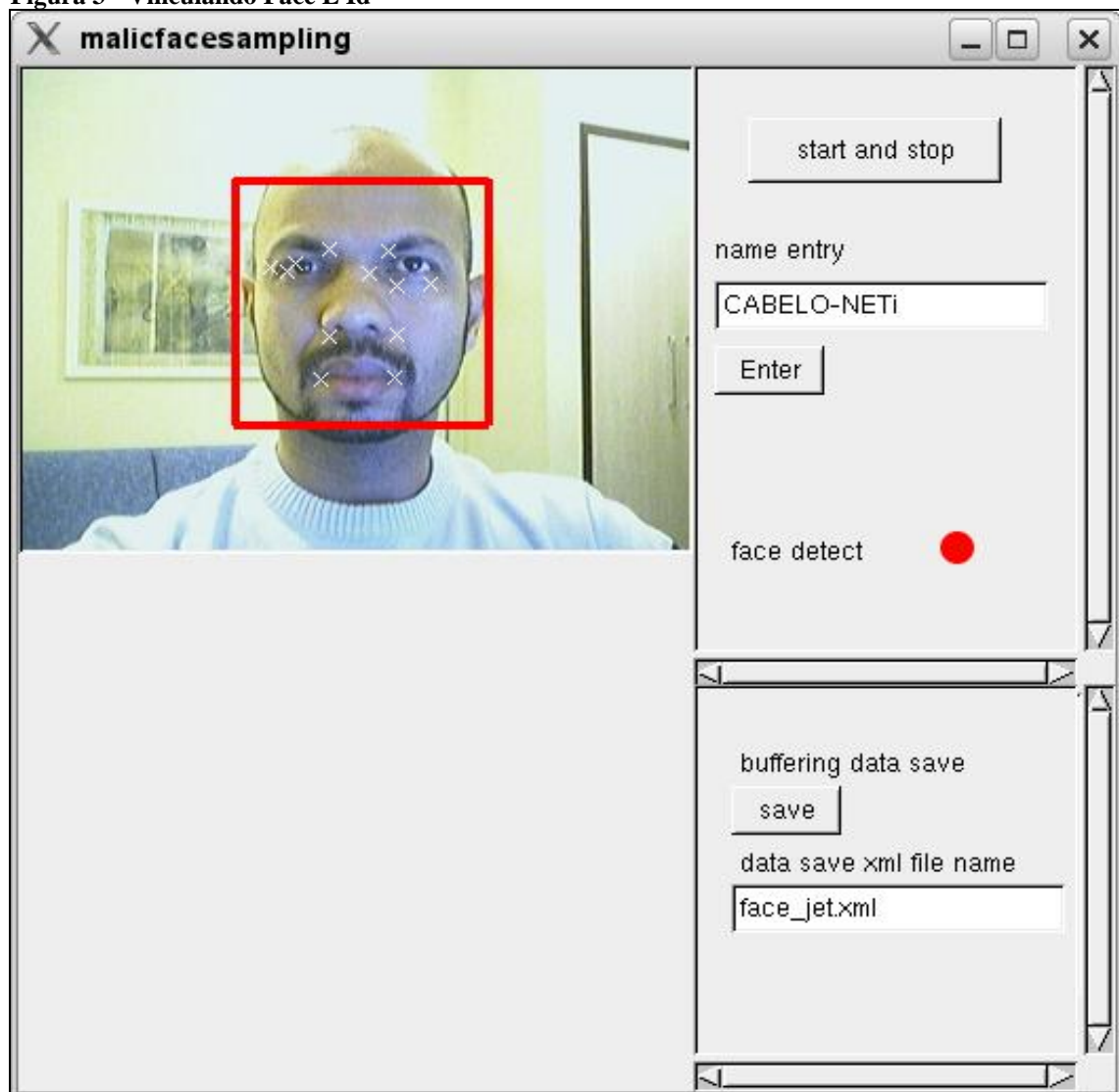
Figura 2 - Mapeando Os Pontos Característicos



Fonte: Faria (2018).

A segunda etapa apresentada por Faria (2018) é a realização do armazenamento da face e um identificador, através do comando “./malicfacesampling”, gerando outro arquivo .XML com as informações da face e do identificador, conforme demonstra a Figura 3.

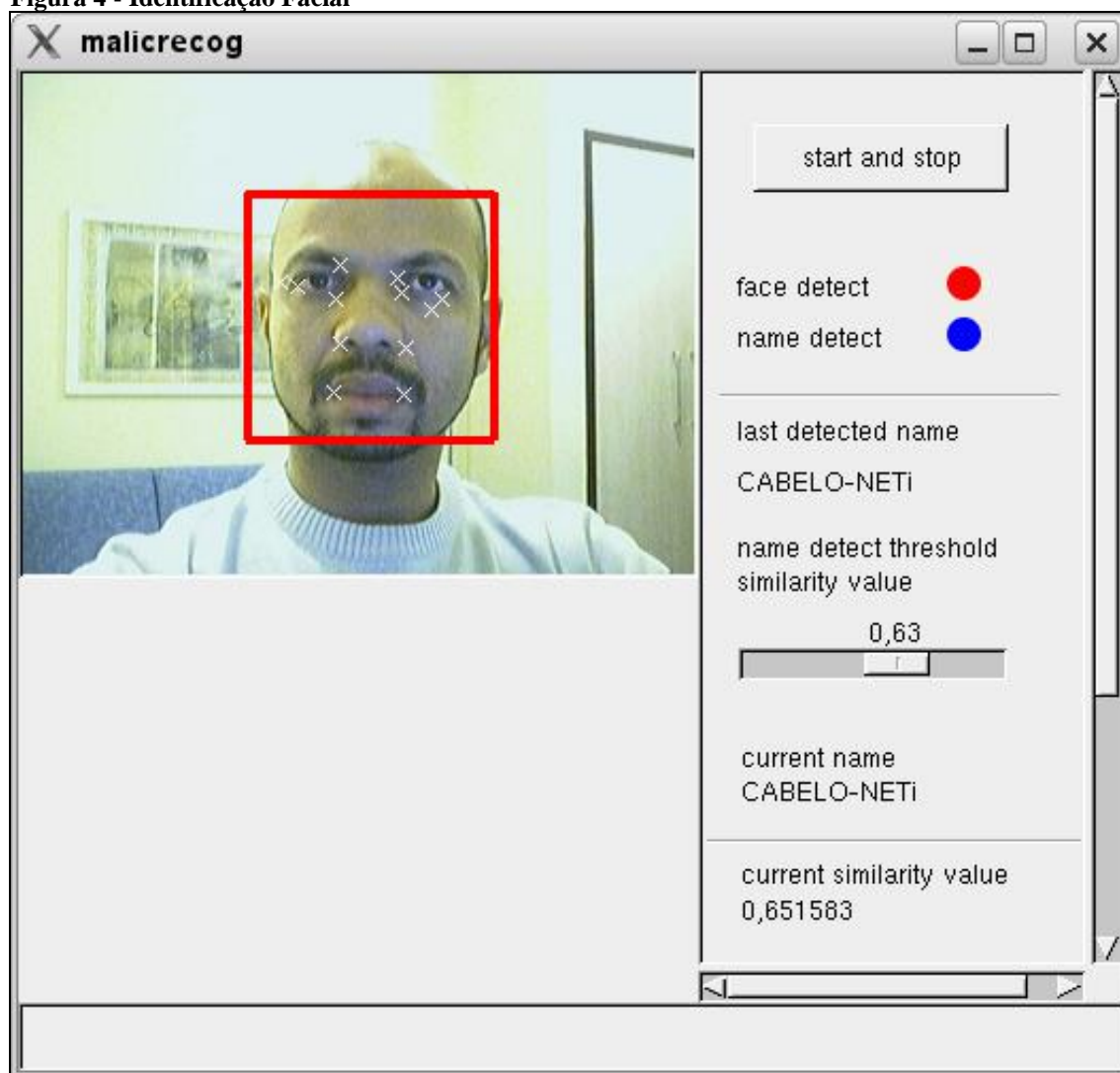
Figura 3 - Vinculando Face E Id



Fonte: Faria (2018).

A terceira e última etapa apresentada por Faria (2018) é a execução do comando “\$./malicrecog -j gaborrg_jet.xml -f face_jet.xml” baseando-se na estrutura montada nos exemplos dispostos acima. Para realizar o reconhecimento da face conforme demonstra a Figura 4, sendo o ponto vermelho para indicar que existe uma face detectada e o ponto azul indica que possui uma identificação realizada com sucesso.

Figura 4 - Identificação Facial

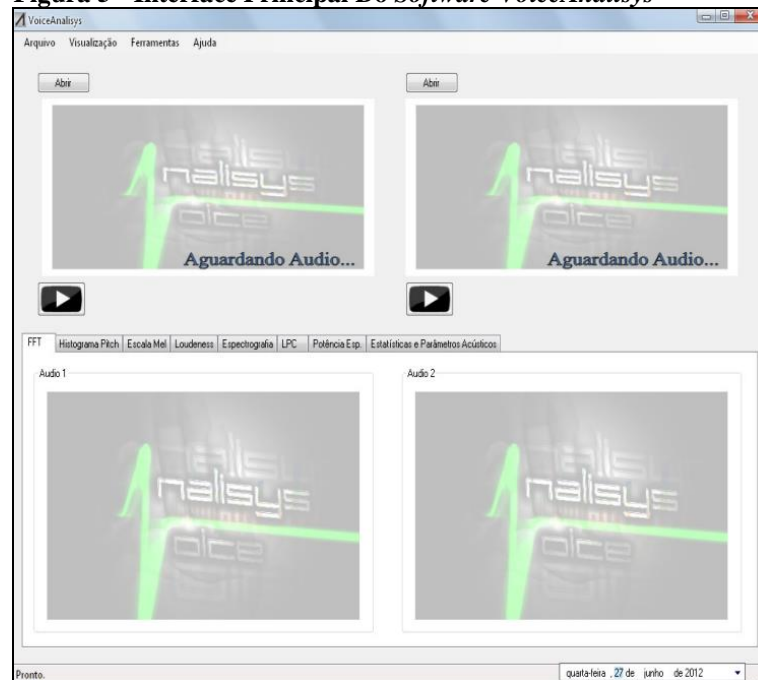


Fonte: Faria (2018).

Silva et al (2012) realizam a apresentação do *software VoiceAnalisys* que tem como função principal auxiliar o processo de perícia no reconhecimento da voz de um determinado locutor com base em procedimentos periciais.

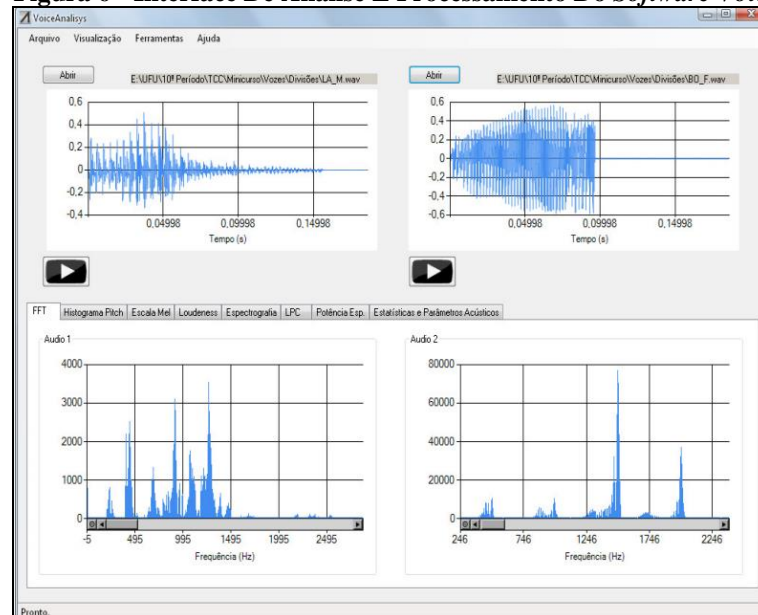
Conforme Silva et al (2012, p. 451) “O sistema é capaz de realizar a extração de parâmetros temporais e no domínio das frequências, além da comparação estatística de parâmetros quantitativos.”. Ainda descrevendo que o sistema possui interface amigável e fornecendo os parâmetros utilizados por peritos na identificação de dois áudios. Conforme demonstra a Figura 5 e a Figura 6.

Figura 5 - Interface Principal Do Software VoiceAnalysisys



Fonte: Silva et al (2012, p. 452).

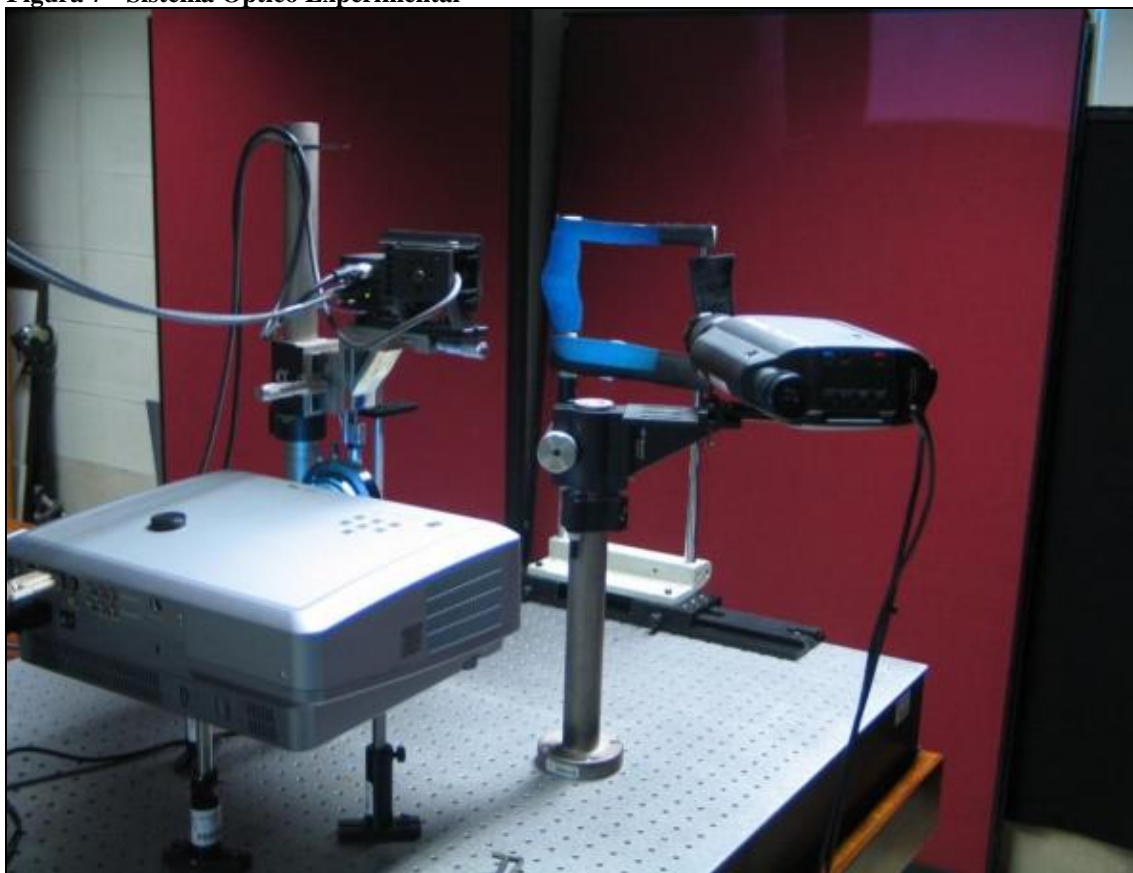
Figura 6 - Interface De Análise E Processamento Do Software VoiceAnalysisys



Fonte: Silva et al (2012, p. 453).

Lucas (2011) desenvolve a uma base de dados com imagens de íris adquiridas e registradas em condições controladas, criando dois modelos de reconhecimento da íris, um com forma linear e outro embasado em redes neurais. O *software* utilizou para realizar a captura das imagens de íris do olho humano um sistema óptico experimental, conforme demonstra a Figura 7, com linguagem base *Matlab*.

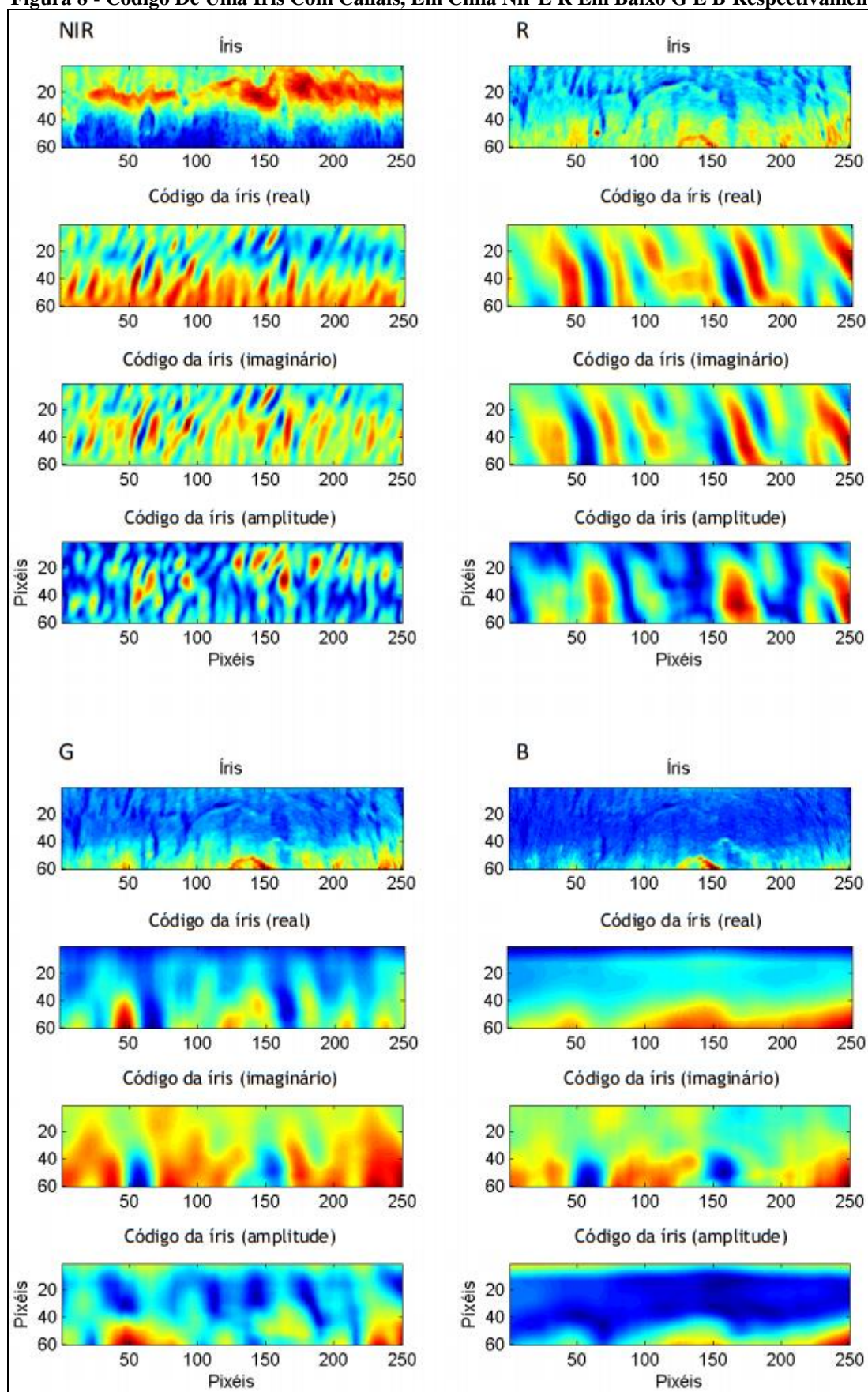
Figura 7 - Sistema Óptico Experimental



Fonte: Lucas (2011, p. 17).

A base que foi criada por Lucas (2011), utilizava imagens capturadas simultaneamente nas regiões do visível (RGB) e do infravermelho próximo do espectro (NIR), depois que foi executada a segmentação e a normalização das imagens capturadas, realiza-se a aperfeiçoamento dos parâmetros dos filtros de Gabor, conseguindo dados relativos aos quatro canais: vermelho, verde, azul (do inglês *Red, Green e Blue - RGB*) e ao canal do infravermelho próximo (*Near Infrared - NIR*). Conforme Demonstra a Figura 8.

Figura 8 - Código De Uma Íris Com Canais, Em Cima Nir E R Em Baixo G E B Respectivamente



Fonte: Lucas (2011, p. 35).

4.2 PROJETO DO PROTÓTIPO

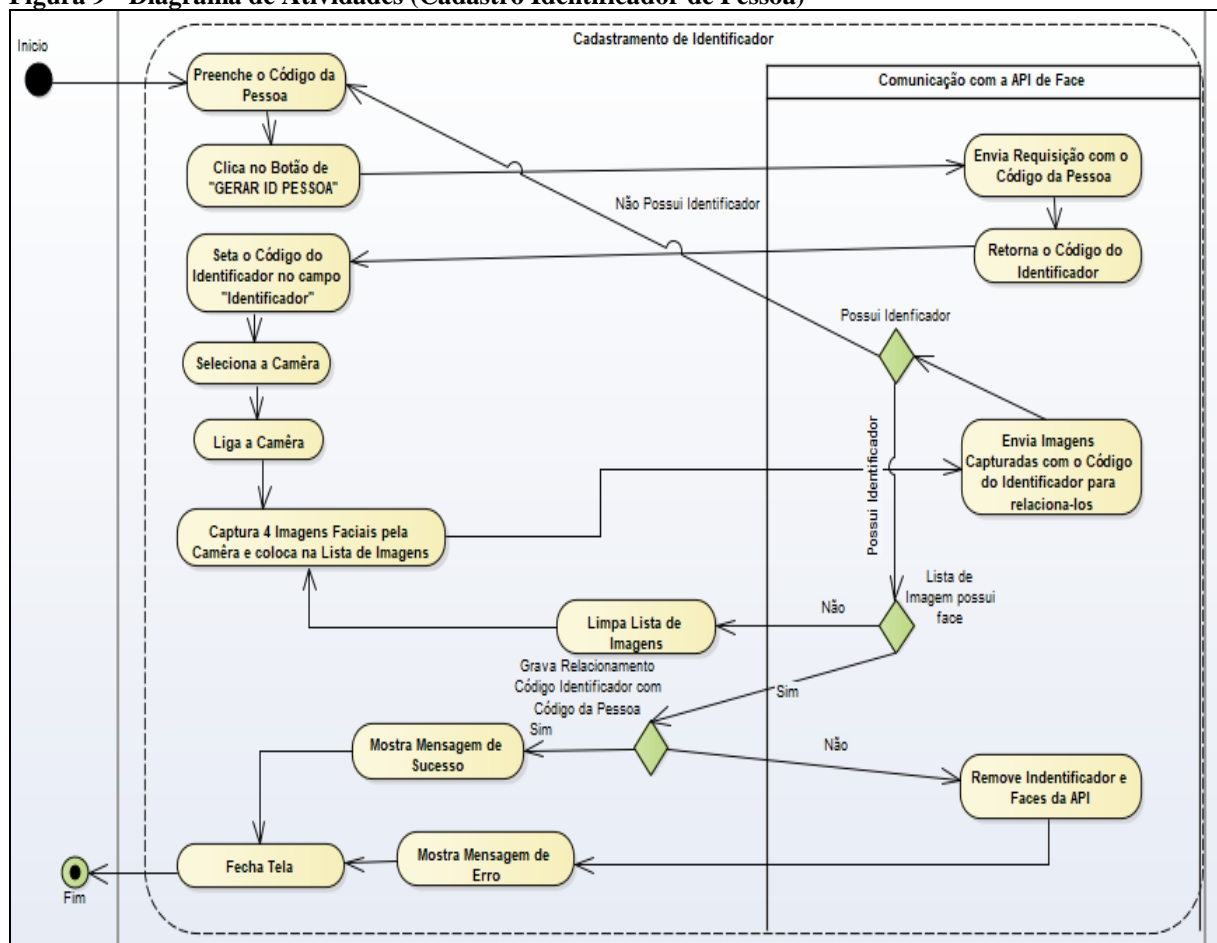
Para o possibilitar um desenvolvimento favorável foram executadas múltiplas análises viabilizando conseguir parâmetros para iniciar o desenvolvimento do protótipo, verificando como estaria disposta a modelagem do banco de dados e também como o sistema desenvolvido iria comunicar-se com a *API* da Azure, evitando comunicações desnecessárias e gerando assim menor tráfego na rede.

4.2.1 Diagrama de Atividades

Nos diagramas de atividades foram demonstradas as iterações realizadas pelo protótipo com a *API* de Face da Azure, demonstrando os procedimentos de cadastramento de um identificador de pessoa, adição de faces no identificador de pessoa, treinamento do algoritmo de identificação, detecção de faces, identificação de faces e geração de *LOG* da identificação.

A Figura 9 demonstra a etapa cadastro de identificador de pessoa, iniciando pela geração do identificador e posterior adição de faces ao identificador da pessoa, sendo utilizado posteriormente para realizar a identificação da pessoa cadastrada.

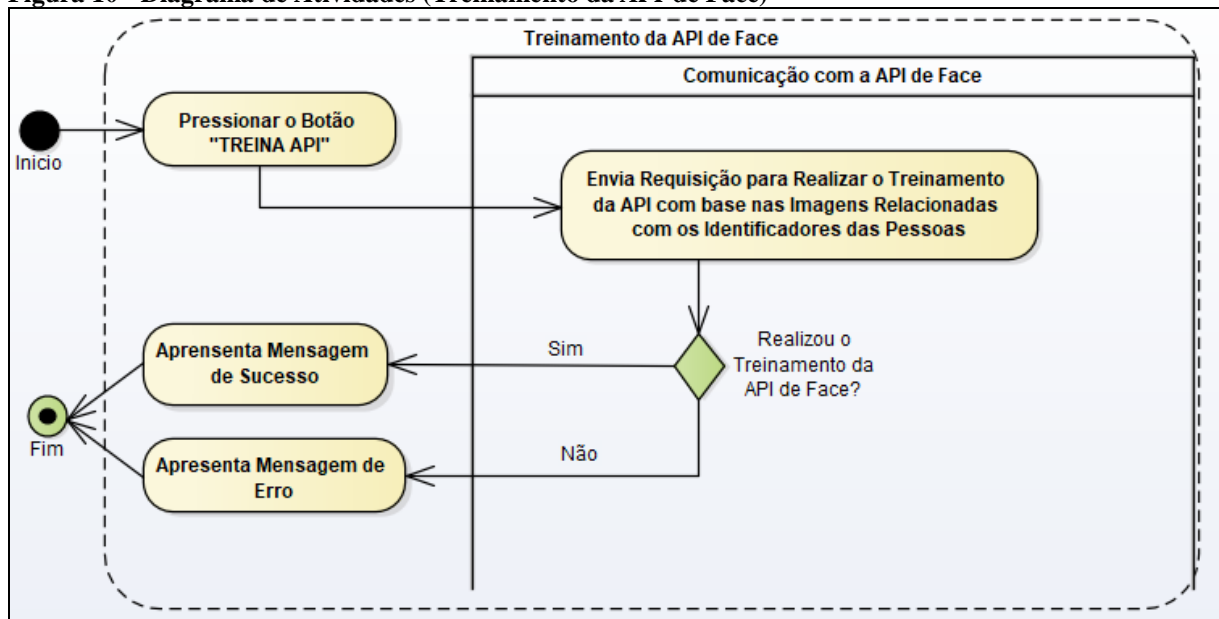
Figura 9 - Diagrama de Atividades (Cadastro Identificador de Pessoa)



Fonte: acervo do autor (2018).

O diagrama de atividade exposto pela Figura 10, demonstra como é realizado o processo de treinamento da API para possibilitar a identificação das faces cadastradas.

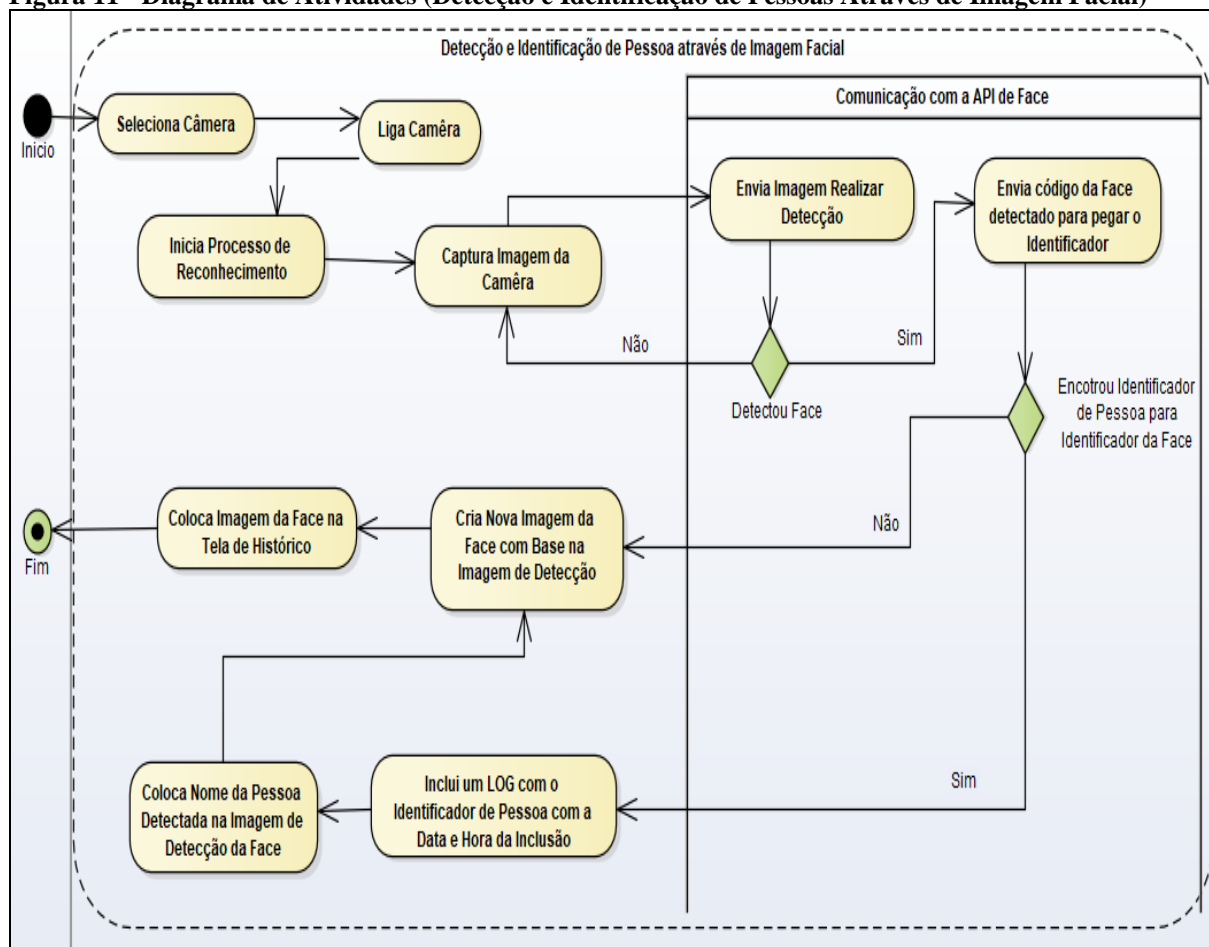
Figura 10 - Diagrama de Atividades (Treinamento da API de Face)



Fonte: acervo do autor (2018).

A Figura 11 apresenta o processo de detecção e identificação de uma face e da geração do *LOG* para a face detectada. Observa-se que pode ocorrer detecções sem identificações da face, desta forma foi previsto para apenas é demonstrar a imagem da face detectada na tela de históricos, não gerando um registro de *LOG* para essa informação.

Figura 11 - Diagrama de Atividades (Detecção e Identificação de Pessoas Através de Imagem Facial)



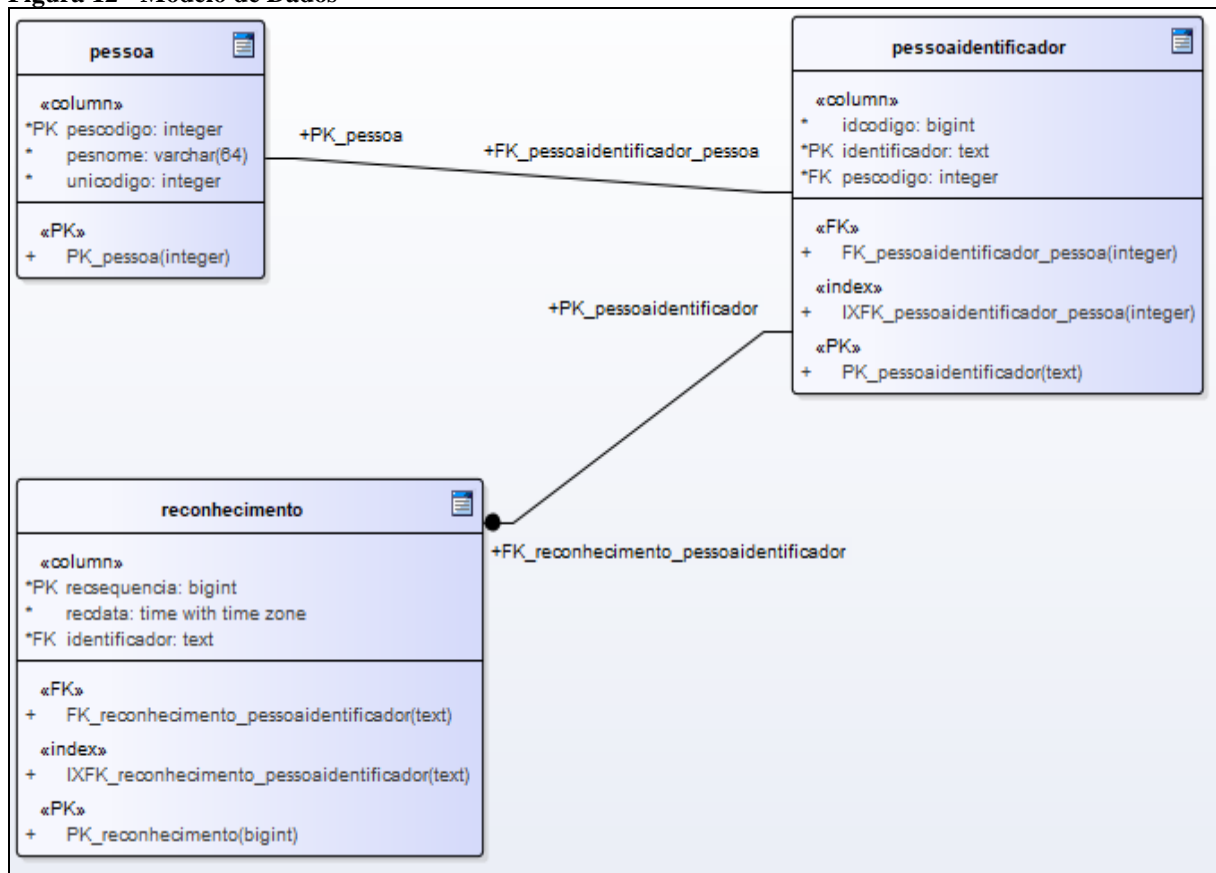
Fonte: acervo do autor (2018).

Após o desenvolvimento dos fluxos de processos que realizariam comunicações com a *API* foi possível iniciar o desenvolvimento do protótipo com o menor número de requisições para a *API*.

4.2.2 Diagrama de Modelo

Após a verificação das comunicações com a *API* da Azure e das informações geradas pelos processos cadastrais de pessoa, identificador e *LOG* de identificações, foi realizado a análise de modelagem de dados, dispondo de como as informações seriam armazenadas. Sendo que o banco de dados utilizado pelo protótipo foi o PostgreSQL versão 10.3. A Figura 12 demonstra o modelo de dados definido pela análise.

Figura 12 - Modelo de Dados



Fonte: acervo do autor (2018).

Pode-se observar que o modelo elaborado foi desenvolvido no formato mais simples possível, apenas armazenando as informações necessárias para a identificação da pessoa. Não foi permitido realizar a exclusão pelo protótipo de um registro da tabela “pessoaidentificador” nem da tabela de pessoa que exista um relacionamento com identificador cadastrado.

4.3 DESENVOLVIMENTO DAS INTERFACES DO PROTÓTIPO

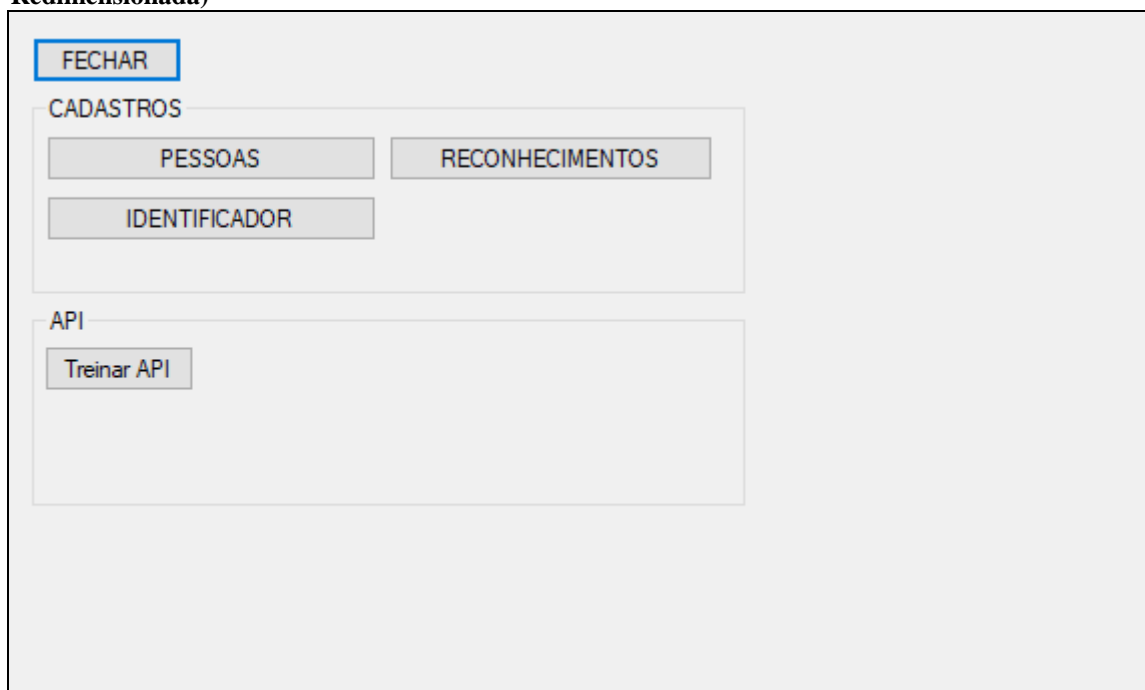
Depois de finalizado a projeção do protótipo, pode-se iniciar o desenvolvimento partindo da interface de chamada das consultas e chamada do treinamento da *API* e posteriormente as interfaces das consultas e de cadastro. Das interfaces de consulta foi iniciado pelo desenvolvimento da interface de pessoas, interface de identificador, interfaces de reconhecimentos e posteriormente a interface do reconhecedor e histórico de reconhecimentos.

O protótipo foi desenvolvido para ser apresentado em telas com resolução 1920x1080 *pixels*, sendo que todas as interfaces foram implementadas em modo de Tela Cheia.

4.3.1 Interface de Chamada das Consultas e Treinamento

A primeira etapa do desenvolvimento foi a realização do desenvolvimento de uma interface para realizar as chamadas das consultas e do processo de treinamento da *API* de Face. Conforme demonstra a Figura 13.

Figura 13 - Interface Principal de Chamada das Rotinas de Consulta e Treinamento da *API* (Interface Redimensionada)



Fonte: acervo do autor (2018).

Ao inicializar a aplicação é invocado automaticamente o método `FormManutencao()` realizado a definição de apresentação em Tela Cheia e abertura simultânea de uma conexão com o banco de dados conforme pode ser observado na Linha 24 da figura, sendo utilizado posteriormente nas rotinas de consulta e cadastro chamadas pelas telas invocadas, conforme demonstra a Figura 14.

Figura 14 - Método da Inicialização da Tela Principal e da Conexão com o Banco de Dados

```
19 public FormManutencao()  
20 {  
21     InitializeComponent();  
22     FormBorderStyle = FormBorderStyle.None;  
23     WindowState = FormWindowState.Maximized;  
24     this.conexao = Conexao.getConexao();  
25     this.manutencao = true;  
26 }
```

Fonte: acervo do autor (2018).

Os botões do container “CADASTROS”, quando pressionados apresentam a interface de consulta do botão passando a conexão como parâmetro. O botão “Treina *APP*” do container “*APP*” realiza a invocação do método `button3_ClickAsync(object, EventArgs)` de Treinamento assíncrono da *API*, conforme demonstra a Figura 15.

Figura 15 - Método de Chamada do Treinamento da *API*

```
76 private async void button3_ClickAsync(object sender, EventArgs e)  
77 {  
78     if(await Treinar.treinarApiAsync())  
79     {  
80         MessageBox.Show("API Treinada!!");  
81     }  
82     else  
83     {  
84         MessageBox.Show("Erro ao Treinar API");  
85     }  
86 }
```

Fonte: acervo do autor (2018).

Ao realizar o fechamento desta tela ocorre a invocação do método `FormManutencao_FormClosing(object, FormClosingEventArgs)` também a finalização da conexão com o banco de dados, conforme demonstrado pela Figura 16.

Figura 16 - Método de Fechamento da Conexão com o Banco de Dados da Tela Principal

```
89 private void FormManutencao_FormClosing(object sender, FormClosingEventArgs e)  
90 {  
91     Conexao.setFecharConexao(conexao);  
92 }
```

Fonte: acervo do autor (2018).

4.3.2 Interfaces de Pessoas

A primeira interface implementada foi de consulta de pessoas cadastradas, essa consulta apresenta os botões para fechar a tela, cadastrar, alterar e excluir um registro selecionado, também é apresentado uma lista de filtros com diferentes formatos de filtragem e as colunas de código da pessoa, nome da pessoa e código universitário, conforme demonstra a Figura 17.

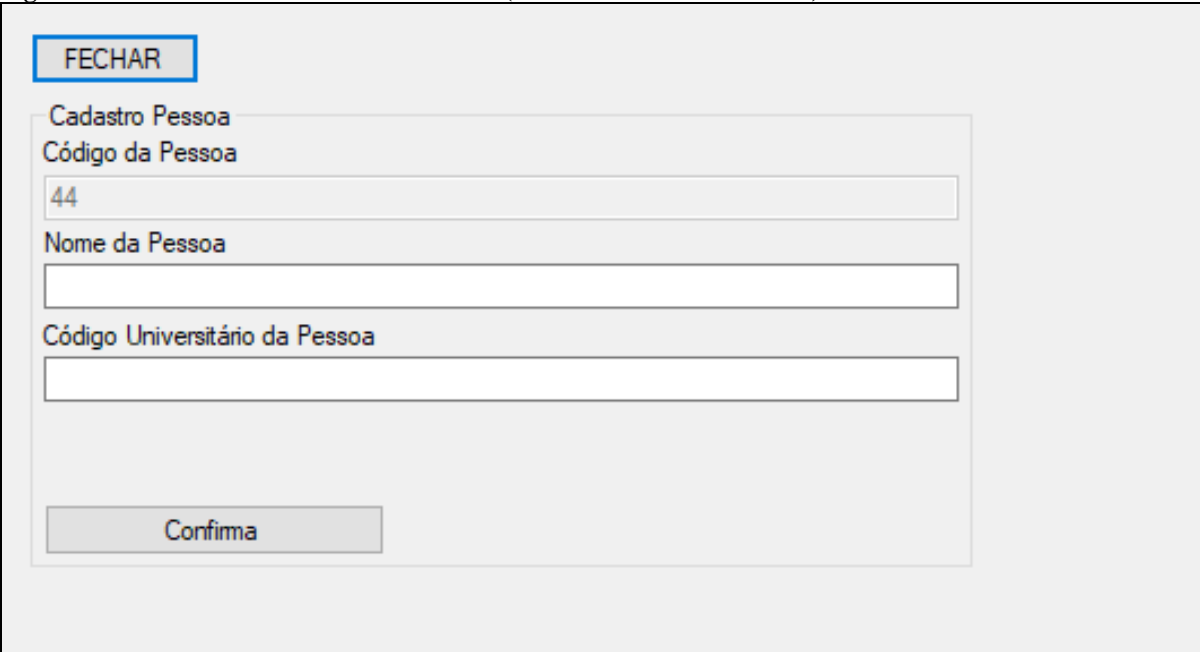
Figura 17 - Interface de Chamada da Consulta de Pessoas (Interface Redimensionada)

	Código da Pessoa	Nome da Pessoa	Código Universitário
▶	1	vinicius dalastra	40357
	2	Ricardo Fronza	41311
	3	Vitor Bellini	40720
	4	Lucas Ledra	28743
	5	Murilo Wippel	40967
	6	Andrey Longen R	41068
	7	Luis da Biz	41184
	8	Fabiano Gabardo...	40717
	9	Douglas da Silva	19494
	10	Carlos Henrique	40665
	11	Guilherme C. Lam	33747
	12	Leonardo Desch...	1005440
	13	Leonardo Pessati	40657

Fonte: acervo do autor (2018).

As ações alterar e excluir funcionam conforme os valores apresentados na tabela de dados (também conhecido como grid de dados), não possibilitando a alteração do código da pessoa. Já o botão de Cadastro realiza a invocação de uma tela de cadastro de pessoa, aonde informa-se o nome e código universitário da pessoa que está sendo cadastrada, conforme demonstra a Figura 18. Ao realizar a confirmação é cadastrada uma pessoa e fechada a tela, voltando para a tela de consulta e atualizando os dados do grid de consulta

Figura 18 - Interface de Cadastro de Pessoa (Interface Redimensionada)



FECHAR

Cadastro Pessoa

Código da Pessoa

44

Nome da Pessoa

Código Universitário da Pessoa

Confirma

Fonte: acervo do autor (2018).

4.3.3 Interfaces de Identificador

Nesta sessão apresenta-se as interfaces do relacionamento da pessoa com o identificador, sendo realizada primeiramente a implementação da rotina de consulta e posteriormente o cadastro de identificador, pois o cadastro de identificador necessitava de comunicação com a *API* da Azure, para a geração de um identificador de pessoa.

A Figura 19 demonstra a interface de consulta dos relacionamentos de pessoa com os identificadores cadastrados, juntamente com as ações de fechar a tela e cadastrar um novo identificador. Nota-se que as ações de alteração e exclusão estão bloqueadas nesta consulta.

Figura 19 - Interface de Consulta do Relacionamento Pessoa com Identificador (Interface Redimensionada)

AÇÕES

FECHAR
CADASTRAR
ALTERAR
EXCLUIR

FILTRO

Campo

Código do Identificador

Tipo

Contém

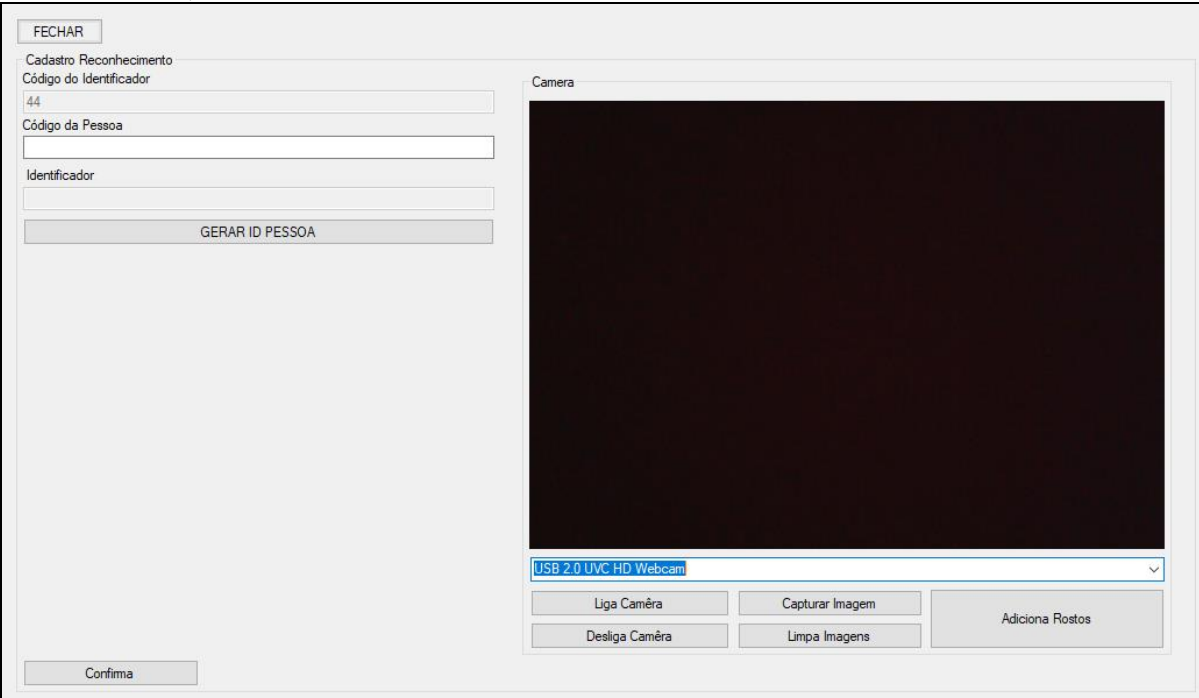
Descrição

	Código do Identificador	Código Universitário	Identificador	Nome da Pessoa	Código da Pessoa
▶	1	40357	f92e755d-6a03-4...	vinicius dalastra	1
	2	41311	b1b1a420-3ee6-...	Ricardo Fronza	2
	3	40720	fb24effe-64e4-4f...	Vitor Bellini	3
	4	28743	0f280326-3fa4-4...	Lucas Ledra	4
	5	40967	276ffc45-d270-4...	Murilo Wippel	5
	6	41068	43bd4b74-46f3-4...	Andrey Longen R	6
	7	41184	cddb77d-1db8-...	Luis da Biz	7
	8	40717	bc8053d8-cf08-4...	Fabiano Gabardo...	8
	9	19494	dc42d5c2-7e55-...	Douglas da Silva	9
	10	40665	b4de4cc2-8d6f-4...	Carlos Henrique	10
	11	33747	528e9f46-67e5-4...	Guilherme C. Lam	11
	12	1005440	77e3b8ac-d0e2-...	Leonardo Desch...	12
	13	40657	ae683ef6-d07a-4...	Leonardo Pessati	13

Fonte: acervo do autor (2018).

Já a Figura 20 demonstra como é a interface de cadastro de um relacionamento de pessoa com identificador da API de Face.

Figura 20 - Interface de Cadastro do Relacionamento Pessoa com Identificador (Interface Redimensionada)



Fonte: acervo do autor (2018).

A Interface de cadastro apresenta o botão GERAR ID PESSOA, que é responsável pela geração de um identificador na *API* com o código da pessoa informado.

Também possui uma tela com imagens de vídeo capturada da câmera selecionada pela lista e os botões para ligar e desligar a câmera, capturar a imagens que está aparecendo na câmera e limpar as imagens capturadas e ainda o botão de Adicionar Rostos, que é responsável por adicionar as imagens capturadas pela câmera no identificador gerado, conforme demonstra a Figura 21.

Figura 21 - Método de Chamada da Adição de Face no Identificador de Pessoa

```
108 private async void button5_Click(object sender, EventArgs e)
109 {
110     if (imagens.Count < 3)
111     {
112         MessageBox.Show("Favor adicionar mais " + (3 - imagens.Count) + " imagens");
113     }
114     else if (textBox4.Text == "" || textBox4.Text == null)
115     {
116         MessageBox.Show("Gerar ID de PESSOA!!");
117     }
118     else
119     {
120         MessageBox.Show("Cadastrando Imagens Rostos, aguarde mensagem de conclusão para Continuar!!!!");
121         Boolean cadastrouRosto = false;
122         for (var i = 0; i < imagens.Count; i++)
123         {
124             cadastrouRosto = await RostoPessoa.adicionaFaceAsync(textBox4.Text, imagens[i]);
125         }
126
127         if (cadastrouRosto)
128         {
129             MessageBox.Show("Imagens Cadastradas com Sucesso [Favor Treinar API! ]");
130         }
131         else
132         {
133             MessageBox.Show("Erro ao Cadastrar Imagens");
134         }
135
136         MessageBox.Show("Lista de Imagens Limpa");
137         imagens = new ArrayList();
138     }
139 }
```

Fonte: acervo do autor (2018).

Pode-se notar no método `button5_Click(object sender, EventArgs e)`, que na linha 110 é realizado a verificação se possui 3 ou mais imagens cadastradas para uma maior precisão do algoritmo posteriormente e se o identificador da pessoa foi gerado, pois sem o identificador não é possível relacionar as imagens. As imagens são enviadas uma por vez até a *API*, para não perder as demais imagens enviadas.

Caso ocorra uma exceção ou a tela seja fechada, devem ser removidos os registros enviados, pois caso seja cadastrado posterior uma mesma face a *API* retorna apenas a primeira identificada a qual ela tem um maior percentual de certeza. Desta forma, quando é fechada a tela invoca-se o método `button1_ClickAsync(object sender, EventArgs e)` que verifica se o campo Identificador está preenchido, caso estiver ele remove da *API* o identificador, conforme demonstra a Figura 22.

Figura 22 - Método de Chamada da Exclusão de Identificador da API

```

33 private async void button1_ClickAsync(object sender, EventArgs e)
34 {
35     if (textBox4.Text != "" && textBox4.Text != null)
36     {
37         await Pessoa.removeIdPessoa(textBox4.Text);
38     }
39     this.Close();
40 }

```

Fonte: acervo do autor (2018).

4.3.4 Interfaces de Reconhecimento

Nesta sessão é apresentada a interface de consulta dos *LOGS* de reconhecimentos realizados e também a interface do cadastro manual do *LOG* de reconhecimento. Deve-se notar que a interface de consulta dos *LOGS* apresenta botões de fechamento da interface, cadastro e exclusão, também apresentando a possibilidade de filtrar registros conforme os campos apresentados no grid de dados.

O grid de dados da interface de consulta apresenta os campos padrões de sequência, data e identificador do reconhecimento, também apresenta os campos de código da pessoa, código universitário e nome da pessoa que foi identificada, conforme demonstra a Figura 23.

Figura 23 - Interface de LOGS de Reconhecimentos Realizados (Interface Redimensionada)

AÇÕES

FECHAR CADASTRAR EXCLUIR

FILTRO

Campo Tipo Descrição

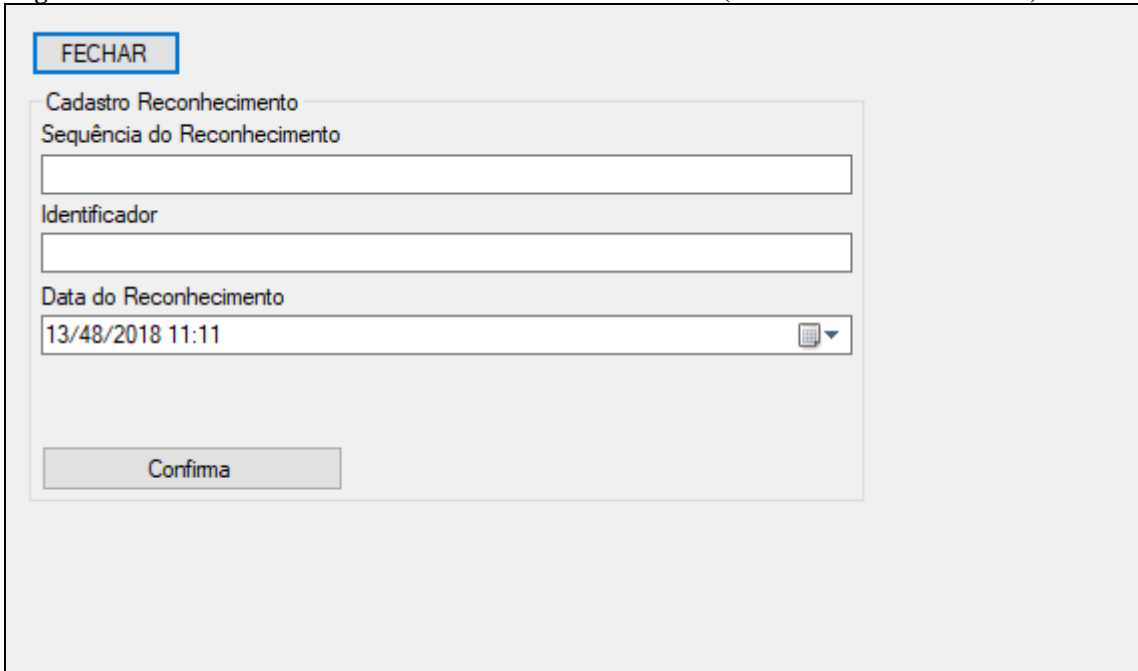
Sequência Contém

	Sequê	Data	Identificador	Código Pessoa	Código Universitária Pessoa	Nome Pessoa
▶	1	29/10/2018 19:58	43bd4b74-46f3-441d-a664-85ff2adeae2b	6	41068	Andrey Longen Rode
	2	29/10/2018 19:58	0f280326-3fa4-4e7-bc95f001a645e6ed	4	28743	Lucas Ledra
	3	29/10/2018 19:58	fb24effe-64e4-4f61-be97-15df977e6b78	3	40720	Vitor Bellini
	4	29/10/2018 19:58	43bd4b74-46f3-441d-a664-85ff2adeae2b	6	41068	Andrey Longen Rode
	5	29/10/2018 19:59	43bd4b74-46f3-441d-a664-85ff2adeae2b	6	41068	Andrey Longen Rode
	6	29/10/2018 19:59	fb24effe-64e4-4f61-be97-15df977e6b78	3	40720	Vitor Bellini
	7	29/10/2018 19:59	0f280326-3fa4-4e7-bc95f001a645e6ed	4	28743	Lucas Ledra
	8	29/10/2018 19:59	fb24effe-64e4-4f61-be97-15df977e6b78	3	40720	Vitor Bellini
	9	29/10/2018 19:59	ae683ef6-d07a-47c3-995d-5f07674aa763	13	40657	Leonardo Pessati
	10	29/10/2018 19:59	0f280326-3fa4-4e7-bc95f001a645e6ed	4	28743	Lucas Ledra
	11	29/10/2018 19:59	ccdbb77d-1db8-4820-afd8-25f06d6fb514	7	41184	Luis da Biz
	12	29/10/2018 19:59	ccdbb77d-1db8-4820-afd8-25f06d6fb514	7	41184	Luis da Biz
	13	29/10/2018 19:59	f92e755d-6a03-4c8b-9c14-0186038931a2	1	40357	vinicius dalastra

Fonte: acervo do autor (2018).

Caso queira ser cadastrado um registro de forma manual, deve-se saber o código do identificador da *API*, sequência do reconhecimento e a data do reconhecimento. A Figura 24 demonstra como é a interface de cadastro de *LOG* de reconhecimento.

Figura 24 - Interface de Cadastro de *LOG* de Reconhecimento (Interface Redimensionada)

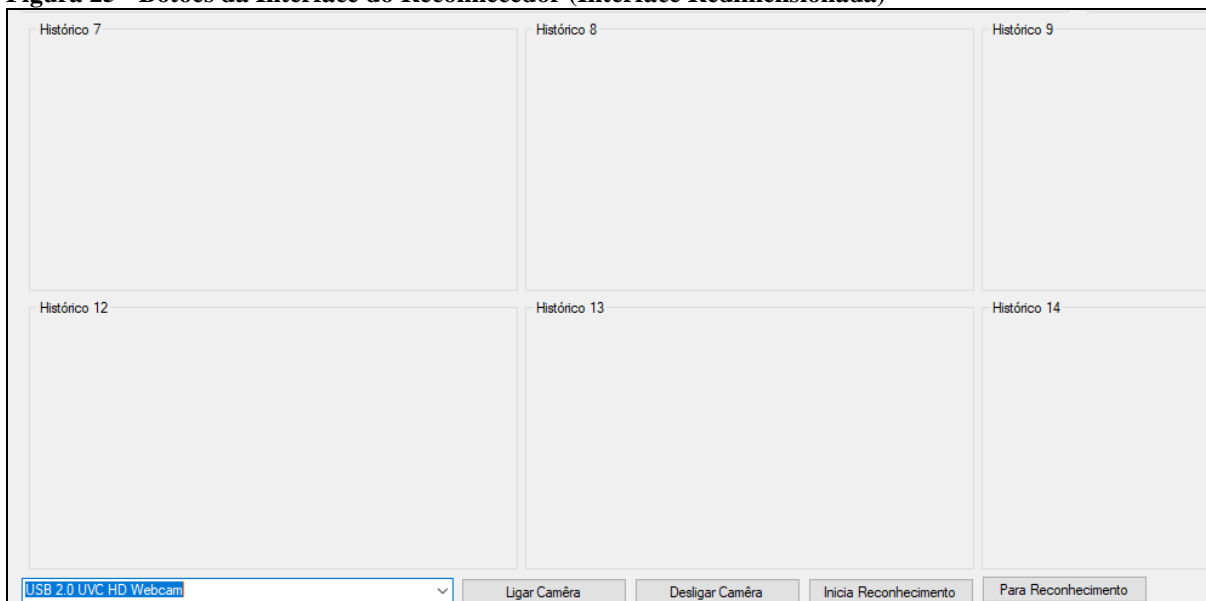
A interface de cadastro de LOG de reconhecimento é uma janela com um fundo cinza claro. No topo esquerdo, há um botão azul com o texto "FECHAR". Abaixo dele, o título "Cadastro Reconhecimento" é exibido em uma fonte menor. Seguem três campos de entrada: "Sequência do Reconhecimento" (um campo de texto simples), "Identificador" (um campo de texto simples) e "Data do Reconhecimento" (um campo de data e hora com um ícone de calendário e uma seta para baixo). O campo de data e hora contém o texto "13/48/2018 11:11". No canto inferior esquerdo da janela, há um botão cinza com o texto "Confirma".

Fonte: acervo do autor (2018).

4.3.5 Interface do Reconhecedor

Nesta sessão é apresentada a interface do reconhecedor, esta interface possui um campo de lista com as câmeras disponíveis para capturar imagens, botão de ligar a câmera, botão de desligar a câmera, botão de iniciar o reconhecimento e ainda um botão para parar o reconhecimento, conforme demonstra a Figura 25.

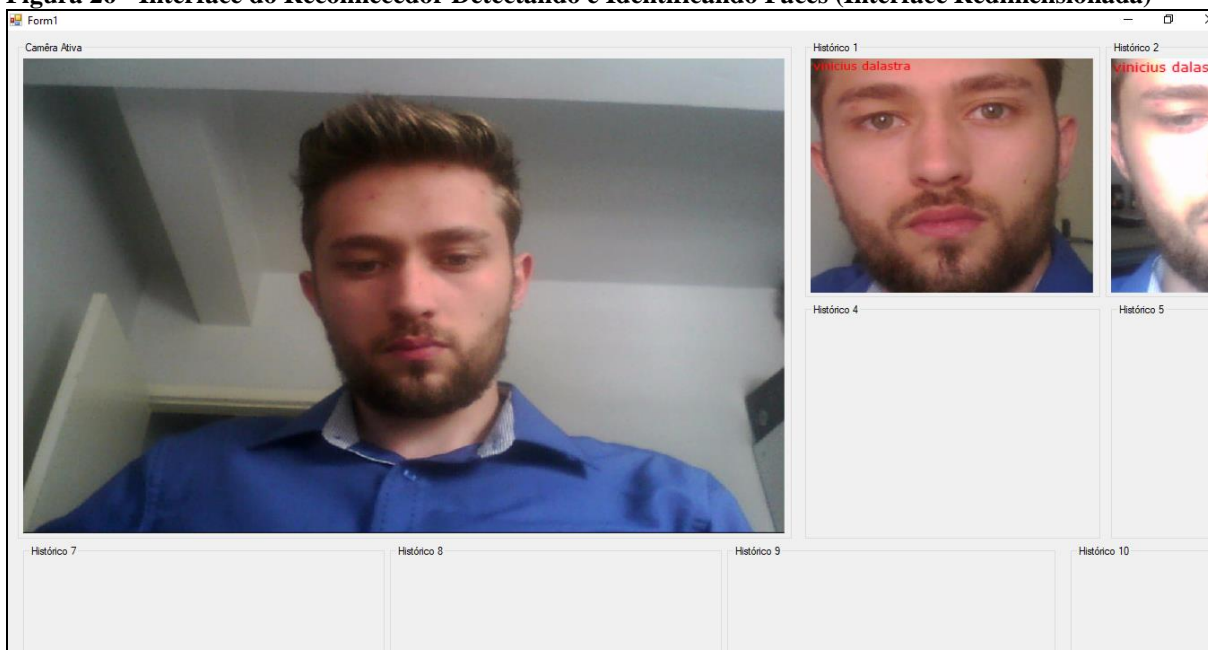
Figura 25 - Botões da Interface do Reconhecedor (Interface Redimensionada)



Fonte: acervo do autor (2018).

Ainda é possível notar acima os containers que são responsáveis para armazenar as imagens detectadas, não somente as identificadas. Ainda nesta interface apresenta-se a imagem em vídeo que está sendo capturada pela câmera, e enviada para detecção e identificação das faces conforme demonstra a Figura 26. As imagens apresentadas nos históricos de 1 a 16 são apenas as faces detectadas na sessão.

Figura 26 - Interface do Reconhecedor Detectando e Identificando Faces (Interface Redimensionada)



Fonte: acervo do autor (2018).

Os processos de detecção de identificação ocorrem separados e de forma assíncrona, quando pressionado o botão inicia reconhecimento, cria-se um thread para em segundo plano realizar o envio das imagens para a *API* realizar a detecção (linha 83), caso ocorra erro na detecção o tratamento é pular a instrução atual do laço de repetição e iniciar com outra imagem capturada, caso seja detectada uma face realiza-se o envio do *Response* retornado pela detecção para ser realizado o processo de identificação (linha 90), geração do *LOG* de reconhecimento e para apresentar no histórico a face detectada. Conforme demonstra a Figura 27.

Figura 27 - Método de Chamada da Detecção e Identificação da Face (Interface Redimensionada)

```
78 private async void realizaReconhecimento()
79 {
80     while(!button2.Enabled)
81     {
82         object imagem = pictureBox1.Image;
83         string sResponse = await Reconhecedor.reconhece(imagem);
84
85         if (sResponse.Contains("error"))
86         {
87             continue;
88         }
89
90         Identificador.reconhece(sResponse, imagem, this);
91
92         await Task.Delay(5000);
93     }
94 }
```

Fonte: acervo do autor (2018).

Ainda pode-se notar o *Delay* implementado, para controlar o tempo de envio de requisições, buscando manter a quantidade baixa de requisições evitando a geração de um custo elevado para o protótipo.

4.4 DESENVOLVIMENTO DO PROCESSO DE CAPTURA DAS IMAGENS DOS DISPOSITIVOS E PROCESSOS DE COMUNICAÇÃO COM A API

Esta sessão tem por objetivo relatar como foi realizado o processo de captura das imagens da câmera e também como foram realizados os processos de comunicação com a *API* de Face da Azure.

4.4.1 Captura de Imagens dos Dispositivos

O processo de captura das imagens nas rotinas de cadastro de identificador de pessoa e na realização dos reconhecimentos faciais foram realizados no mesmo formato, utilizando a framework AForge.NET.

Para iniciar o processo foi realizada a importação das referências no projeto e então nas interfaces que iria ser utilizada a câmera criado os atributos demonstrados na Figura 28.

Figura 28 - Atributos para Utilização da Câmera

```
19 private FilterInfoCollection device;  
20 private VideoCaptureDevice image;
```

Fonte: acervo do autor (2018).

Ao carregar o formulário foi introduzido no método de invocação, o carregamento dos dispositivos de entrada de vídeo (linha 32) e depois percorrido as informações capturadas definindo os em uma caixa de seleção para o usuário selecionar (linha 35), informando o primeiro encontrado como valor padrão selecionado (linha 38), conforme demonstra a Figura 29.

Figura 29 - Carregamento dos Dispositivos de Entrada de Vídeo Disponíveis

```
32 device = new FilterInfoCollection(FilterCategory.VideoInputDevice);  
33 foreach (FilterInfo capture in device)  
34 {  
35     comboBox1.Items.Add(capture.Name);  
36     ...  
37 }  
38 comboBox1.SelectedIndex = 0;  
39 ...  
40 image = new VideoCaptureDevice(device[comboBox1.SelectedIndex].MonikerString);
```

Fonte: acervo do autor (2018).

Os botões responsáveis por ligar a câmera, criam os frames do dispositivo de vídeo selecionado (linha 48) e inicializam a apresentação das capturas de vídeo (linha 54), definido no *pictureBox* da interface. Já o botão de desligar apenas para a criação dos frames (linha 62), caso a captura de imagens da câmera esteja ocorrendo (linha 60). Conforme demonstra a Figura 30.

Figura 30 - Inicialização e Finalização da Câmera

```
43      /*Botão Ligar Camera*/
44      private void button1_Click(object sender, EventArgs e)
45      {
46          image = new VideoCaptureDevice(device[comboBox1.SelectedIndex].MonikerString);
47          image.NewFrame += new NewFrameEventHandler(camera);
48          image.Start();
49      }
50
51      void camera(object sender, NewFrameEventArgs e)
52      {
53          Bitmap bit = (Bitmap)e.Frame.Clone();
54          pictureBox1.Image = bit;
55      }
56
57      /*Botão Desligar Camera*/
58      private void button3_Click(object sender, EventArgs e)
59      {
60          if (image.IsRunning)
61          {
62              image.Stop();
63          }
64      }
```

Fonte: acervo do autor (2018).

Caso o protótipo sofresse uma falha causando exceção ou o fechamento da tela sem o desligamento da câmera, chama-se o método `button3_Click(object sender, EventArgs e)` para realizar o desligamento do dispositivo de forma correta evitando que o mesmo fique ligado em segundo plano.

4.4.2 Processos de Comunicação com a API

Nesta sessão são apresentados os processos de comunicação realizados com a *API* de Face da Azure, que tiveram seu desenvolvimento em partes separadas, sendo realizadas via requisição *POST* e recebendo retornos de objetos em *JSON*.

A *API* ficou responsável pelos processos de Inteligência Artificial aplicados na Detecção e Identificação de Pessoas através da biometria facial, realizando cálculos e aplicando processos de aprendizado, visto que a mesma utiliza informações aprendidas em iterações de outras aplicações.

4.4.2.1 Criação do *PersonGroup*

Nesta sessão abordaremos o *PersonGroup*, que é basicamente o agrupador de pessoas que foram cadastradas, possibilitando o cadastro de até 10 mil pessoas, o Azure (2018) explica que “Uma pessoa é uma unidade básica de identificação. Uma pessoa pode ter uma ou mais faces conhecidas registradas. No entanto, um *PersonGroup* é uma coleção de pessoas, e cada pessoa é definida dentro de um determinado *PersonGroup*.”. Portanto antes de ser realizada qualquer outra etapa deve-se criar um *PersonGroup* e posteriormente criar pessoas nele.

A criação do *PersonGroup* do protótipo, foi realizado de forma manual no site de referências da API de Face da Azure, informando o *personGroupId*, como “undavi1” e a *Ocp-Api-Subscription-Key* com a chave do Azure, gerando a URL <<https://brazilsouth.api.cognitive.microsoft.com/face/v1.0/persongroups/undavi1>>, com a corpo da requisição no formato JSON, conforme demonstra a Figura 31.

Figura 31 - Corpo da Requisição *PersonGroupId*

```
1 {  
2   "name": "undavi1",  
3   "userData": "Grupo de Testes do experimento na Unidavi."  
4 }
```

Fonte: acervo do autor (2018).

4.4.2.2 Geração e Remoção do Identificador de Pessoa

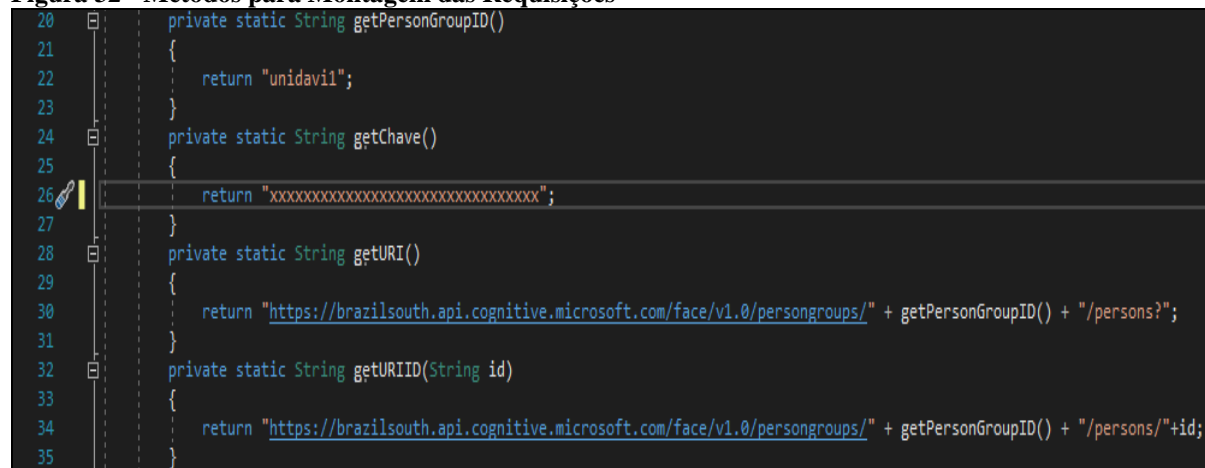
Nesta sessão será apresenta a forma que é criado o Identificador de Pessoa dentro do Grupo de Pessoas, que posteriormente será utilizado para adicionar as faces de uma pessoa e também será o retorno da identificação de um rosto.

A criação do *PersonId* ou Identificador de Pessoa foi realizado na rotina de cadastro do identificador, que utiliza do código da pessoa que foi informado para a criação do identificador, e ao pressionar o botão de “GERAR ID PESSOA” conforme apresenta a Figura 20, é realizado um envio de requisição para a API do Azure que realiza a codificação de um Identificador único para aquela pessoa do grupo.

Para realizar a comunicação com a API foi criado uma classe chamada Pessoa, que possui alguns métodos padrões retornando algumas informações importantes como o

PersonGroupId que foi criado (linha 20), a chave (*Ocp-Apim-Subscription-Key*) para realizar autenticação com a *API* (linha 24), a chave foi ocultada por segurança de acesso, sendo que será utilizada posteriormente em outras comunicações com a *API*, também a montagem de um *URI* (linha 28) e outro com o parâmetro do identificador da Pessoa sendo recebido (linha 32), conforme demonstra a Figura 32.

Figura 32 - Métodos para Montagem das Requisições



```
20 private static String getPersonGroupID()
21 {
22     return "unidavi1";
23 }
24 private static String getChave()
25 {
26     return "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
27 }
28 private static String getURI()
29 {
30     return "https://brazilsouth.api.cognitive.microsoft.com/face/v1.0/persongroups/" + getPersonGroupID() + "/persons?";
31 }
32 private static String getURIID(String id)
33 {
34     return "https://brazilsouth.api.cognitive.microsoft.com/face/v1.0/persongroups/" + getPersonGroupID() + "/persons/" + id;
35 }
```

Fonte: acervo do autor (2018).

Foram desenvolvidos outros métodos para a criação do Identificador e para a remoção do Identificador caso necessário.

O método de Criação é o *CriaIdPessoa(String nome)* sendo apresentado na Figura 33, a linha 41 é responsável pela adição da chave de comunicação no cabeçalho da requisição criada na linha 40, posteriormente na linha 44 é adicionado id da pessoa no corpo da requisição, sendo este corpo uma *String* em formato *JSON*, depois de montada a requisição é enviada (linha 51) e o retorno é recebido também em formato *JSON*, sendo retornado pelo método o valor do identificador gerado, caso não foi gerado retorna uma *String* vazia.

Figura 33 - Métodos de comunicação com API para Criação do Identificador de Pessoa

```
37 public static async Task<String> CriaIdPessoa(String nome)
38 {
39     var client = new HttpClient();
40     // Request headers
41     client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", getChave());
42     var uri = getURI();
43     HttpResponseMessage response;
44     String body = "{\"name\": \"" + nome + "\"}";
45     byte[] byteData = Encoding.UTF8.GetBytes(body);
46
47     using (var content = new ByteArrayContent(byteData))
48     {
49         content.Headers.ContentType = new MediaTypeHeaderValue("application/json");
50         response = await client.PostAsync(uri, content);
51         Stream responseStream = await response.Content.ReadAsStreamAsync();
52         responseStream = await response.Content.ReadAsStreamAsync();
53         using (StreamReader responseReader = new StreamReader(responseStream))
54         {
55             string sRetorno = responseReader.ReadToEnd();
56
57             JObject oRetorno = JObject.Parse(sRetorno);
58
59             foreach (JProperty propriedade in oRetorno.Properties())
60             {
61                 if (propriedade.Name == "personId")
62                 {
63                     return (string)propriedade.Value;
64                 }
65             }
66         }
67     }
68     return "";
69 }
```

Fonte: acervo do autor (2018).

O método de remoção do Identificador da API é o `removeIdPessoa(String id)` sendo demonstrado pela Figura 34, aonde é montado uma requisição com os métodos mencionados anteriormente, neste método de remoção não foi esperado retorno, afinal o usuário não precisa saber do retorno da remoção, pois o mesmo ocorre em segundo plano.

Figura 34 - Métodos de comunicação com *API* para Remoção do Identificador de Pessoa

```
77 public static async Task removeIdPessoa(String id)
78 {
79     var client = new HttpClient();
80
81     // Request headers
82     client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", getChave());
83
84     var uri = getURIID(id);
85
86     HttpResponseMessage sRetorno = await client.DeleteAsync(uri);
87 }
```

Fonte: acervo do autor (2018).

4.4.2.3 Adição de Faces no Identificador de Pessoa

Nesta sessão será demonstrada como foi realizado o processo de comunicação entre o protótipo e a *API* do Azure, quanto a adição de faces no identificador da pessoa. O processo ocorreu pela captura das imagens inicialmente na tela de cadastro do identificador e posteriormente o envio da imagem capturada, juntamente com o identificador da pessoa e do grupo da pessoa para a *API*.

Conforme demonstra a Figura 35, o método `adicionaFaceAsync(String idPessoa, object imagens)` realizada a adição da face no identificador de pessoa. Neste método é realizado a montagem do *URI* (linha 28) com o parâmetro do Identificador de Pessoa e Identificador do Grupo e depois realizada a adição na requisição (linha 50), também é realizado a adição da chave no cabeçalho da requisição (linha 39) e a conversão da imagem a ser enviada para *API* de objeto para o formato de um vetor do tipo *byte* (linha 45), retornado para o local da chamada um valor booleano se adicionou ou não o rosto.

Figura 35 - Métodos de comunicação com API para Adição de Faces na Pessoa

```
26 private static String getURI(String idPessoa)
27 {
28     return "https://brazilsouth.api.cognitive.microsoft.com/face/v1.0/persongroups/" +
29         getPersonGroupID() +
30         "/persons/" +
31         idPessoa +
32         "/persistedFaces";
33 }
34
35 public static async Task<Boolean> adicionaFaceAsync(String idPessoa, object imagens)
36 {
37     var client = new HttpClient();
38     // Request headers
39     client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", getChave());
40
41     var uri = getURI(idPessoa);
42
43     HttpResponseMessage response;
44     ImageConverter converter = new ImageConverter();
45     byte[] byteData = (byte[])converter.ConvertTo(imagens, typeof(byte[]));
46
47     using (var content = new ByteArrayContent(byteData))
48     {
49         content.Headers.ContentType = new MediaTypeHeaderValue("application/octet-stream");
50         response = await client.PostAsync(uri, content);
51         Stream responseStream = await response.Content.ReadAsStreamAsync();
52         responseStream = await response.Content.ReadAsStreamAsync();
53         using (StreamReader responseReader = new StreamReader(responseStream))
54         {
55             String sResponse = responseReader.ReadToEnd();
56             return sResponse.Contains("persistedFaceId");
57         }
58     }
59 }
```

Fonte: acervo do autor (2018).

4.4.2.4 Treinamento da API

Nesta sessão é abordado a comunicação do protótipo com a API do Azure para realizar o treinamento da API, de acordo com o Azure (2018) “O *PersonGroup* deve ser treinado antes que se possa realizar a identificação utilizando-o. Além disso, ele precisará ser treinado novamente após a adição ou remoção de qualquer pessoa ou se a face registrada de qualquer pessoa for editada.”.

Desta forma foi criado uma classe específica para realizar o treinamento, conforme demonstra a Figura 36. No método `treinaApiAsync()` é realizado o treinamento da API de forma assíncrona retornando ao final um valor booleano para o sucesso do processo. Neste método é adicionada a chave da API (linha 37), depois a montagem da URI (linha 29) e envio da requisição (linha 47) e no final se o retorno for de sucesso é retornado um valor verdadeiro para o booleano.

Figura 36 - Métodos de comunicação com API para Treinamento da API

```
27 private static String getURI()
28 {
29     return "https://brazilsouth.api.cognitive.microsoft.com/face/v1.0/persongroups/" +
30         getPersonGroupID() +
31         "/train?";
32 }
33 public static async Task<Boolean> treinarApiAsync()
34 {
35     var client = new HttpClient();
36     // Request headers
37     client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", getChave());
38
39     var uri = getURI();
40
41     HttpResponseMessage response;
42     byte[] byteData = Encoding.UTF8.GetBytes("{}");
43
44     using (var content = new ByteArrayContent(byteData))
45     {
46         content.Headers.ContentType = new MediaTypeHeaderValue("application/json");
47         response = await client.PostAsync(uri, content);
48         return response.StatusCode == HttpStatusCode.Accepted;
49     }
50 }
```

Fonte: acervo do autor (2018).

4.4.2.5 Detecção Facial

Nesta sessão é apresentada a forma que foi realizada a comunicação do protótipo com a API para detecção das faces nas imagens transmitidas, através do método reconhece(object image) (linha 25).

Neste método é realizado a montagem da requisição com o valor da chave no cabeçalho (linha 29) e com a transformação da imagem em um vetor do tipo de *byte* (linha 35), e posterior envio da requisição (linha 40) com base endereço de *URI* montado pelo método getURI(), finalizando o método com o retorno da resposta da API, essa resposta contém no caso do protótipo o retorno de identificadores de múltiplas faces detectadas, cada uma com um identificador específico, conforme demonstra a Figura 37. Para realizar a identificação das faces deve-se realizar outro processo que é descrito na sessão 4.4.2.6.

Figura 37 - Métodos de comunicação com API para Detecção de Faces

```
21 private static String getURI()
22 {
23     return "https://brazilsouth.api.cognitive.microsoft.com/face/v1.0/detect?returnFaceId=true";
24 }
25 public static async Task<String> reconhece(object image)
26 {
27     var client = new HttpClient();
28     // Request headers
29     client.DefaultRequestHeaders.Add("Ocp-Apim-Subscription-Key", getChave());
30
31     var uri = getURI();
32
33     HttpResponseMessage response;
34     ImageConverter converter = new ImageConverter();
35     byte[] byteData = (byte[])converter.ConvertTo(image, typeof(byte[]));
36
37     using (var content = new ByteArrayContent(byteData))
38     {
39         content.Headers.ContentType = new MediaTypeHeaderValue("application/octet-stream");
40         response = await client.PostAsync(uri, content);
41         Stream responseStream = await response.Content.ReadAsStreamAsync();
42         responseStream = await response.Content.ReadAsStreamAsync();
43         using (StreamReader responseReader = new StreamReader(responseStream))
44         {
45             String sResponse = responseReader.ReadToEnd();
46
47             return sResponse;
48         }
49     }
50 }
```

Fonte: acervo do autor (2018).

4.4.2.6 Identificação de Faces Detectadas

Nesta sessão é apresentada a forma em que o protótipo se comunicou com a API do Azure para realizar a identificação das faces detectadas no processo mencionado na sessão anterior.

A face de teste precisa ser detectada usando as etapas anteriores e, em seguida, a ID da face será passada para a API de identificação como um segundo argumento. Várias IDs de face podem ser identificadas ao mesmo tempo, e o resultado conterá todos os resultados de identificação. Por padrão, a identificação retornará somente uma pessoa que melhor corresponde à face de teste. Se preferir, você poderá especificar o parâmetro opcional *maxNumOfCandidatesReturned* para permitir que a identidade retorne mais candidatos. (AZURE, 2018).

Para a etapa de detecção é realizado a montagem de uma requisição com a chave de autenticação do Azure, a URI montada conforme demonstra a Figura 38 e ainda o corpo da requisição em formato *JSON*.

Figura 38 - Métodos da URI para Identificação das Faces

```
26 private static String getURI()  
27 {  
28     return "https://brazilsouth.api.cognitive.microsoft.com/face/v1.0/identify";  
29 }
```

Fonte: acervo do autor (2018).

O corpo da requisição é montado com um vetor de identificadores das faces detectadas no formato de *String* (*faceIds*), o identificador do grupo que está sendo identificada a pessoa (*PersonGroupId*), o número de retorno de candidatos da face (*maxNumOfCandidatesReturned*) e ainda o valor mínimo de confiança na detecção (*confidenceThreshold*) que varia de 0 a 1, conforme demonstra a Figura 39.

Figura 39 - Corpo da Requisição de Identificação

```
66 String body = "{ \"PersonGroupId\": \"\" + getPersonGroupID() + "\", " +  
67     "\"faceIds\" : [\" + sFaceId + \"], " +  
68     "\"maxNumOfCandidatesReturned\": 1, " +  
69     "\"confidenceThreshold\": 0.55}";
```

Fonte: acervo do autor (2018).

Após o envio da requisição para a *API*, aguarda-se o retorno em formato *JSON*. Caso o retorno da requisição conter erro, o método retorna um valor falso. E caso contrário ele retorna um vetor de objetos com o identificador da face em forma de *String*. E retorna também o valor de confiança em um ponto flutuante de duas casas decimais, conforme demonstra a Figura 40 com a identificação de duas faces.

Figura 40 - Retorno da Requisição de Identificação Facial

```
[
  {
    "faceId": "c5c24a82-6845-4031-9d5d-978df9175426",
    "candidates": [
      {
        "personId": "25985303-c537-4467-b41d-bdb45cd95ca1",
        "confidence": 0.92
      }
    ]
  },
  {
    "faceId": "65d083d4-9447-47d1-af30-b626144bf0fb",
    "candidates": [
      {
        "personId": "2ae4935b-9659-44c3-977f-61fac20d0538",
        "confidence": 0.89
      }
    ]
  }
]
```

Fonte: adaptado de Azure (2018).

Após capturado o retorno, percorre-se o vetor de inserindo os identificadores de pessoa como registro de *LOG* de Reconhecimento, com data e hora da inclusão. Posteriormente realiza-se a criação de imagens com o nome da pessoa que pertence o identificador e o rosto redimensionado da imagem capturada, conforme demonstra a Figura 41.

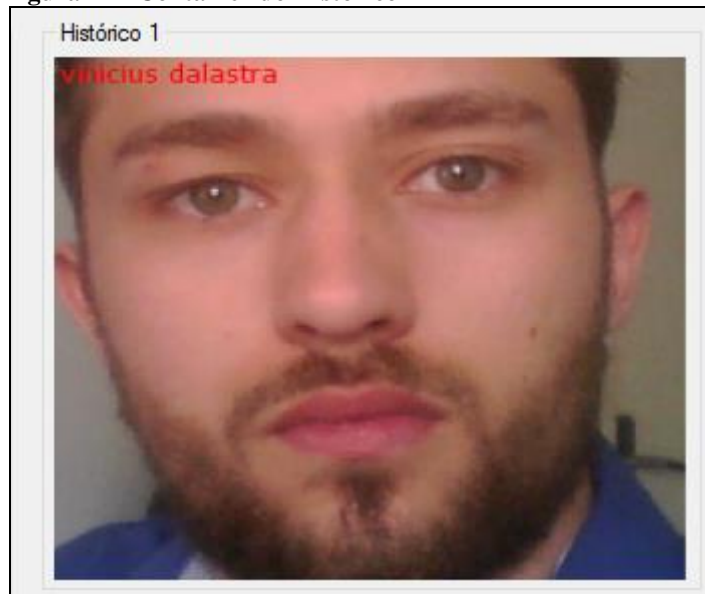
Figura 41 - Formatação da Imagem da Face

```
152 Retangulo retangulo = propriedade.Value.ToObject<Retangulo>();
153 Bitmap myBitmap = (Bitmap)imagem;
154 Rectangle cloneRect = new Rectangle(retangulo.left,retangulo.top, (retangulo.width + 20),(retangulo.height+20));
155 Bitmap BitmapCortado = myBitmap.Clone(cloneRect, myBitmap.PixelFormat);
156
157 Graphics grafico = Graphics.FromImage(BitmapCortado);
158 grafico.DrawString(nomePessoa, new System.Drawing.Font("Tahoma", 10), Brushes.Red, new PointF(0, 0));
```

Fonte: adaptado de Azure (2018).

Posteriormente a formatação da imagem da face identificada é definida para a mesma ser apresentada no histórico de detecções, conforme demonstra a Figura 42. As detecções realizadas anteriores são movidas para o quadro próximo, sendo 16 o máximo de históricos apresentados.

Figura 42 - Container do Histórico 1

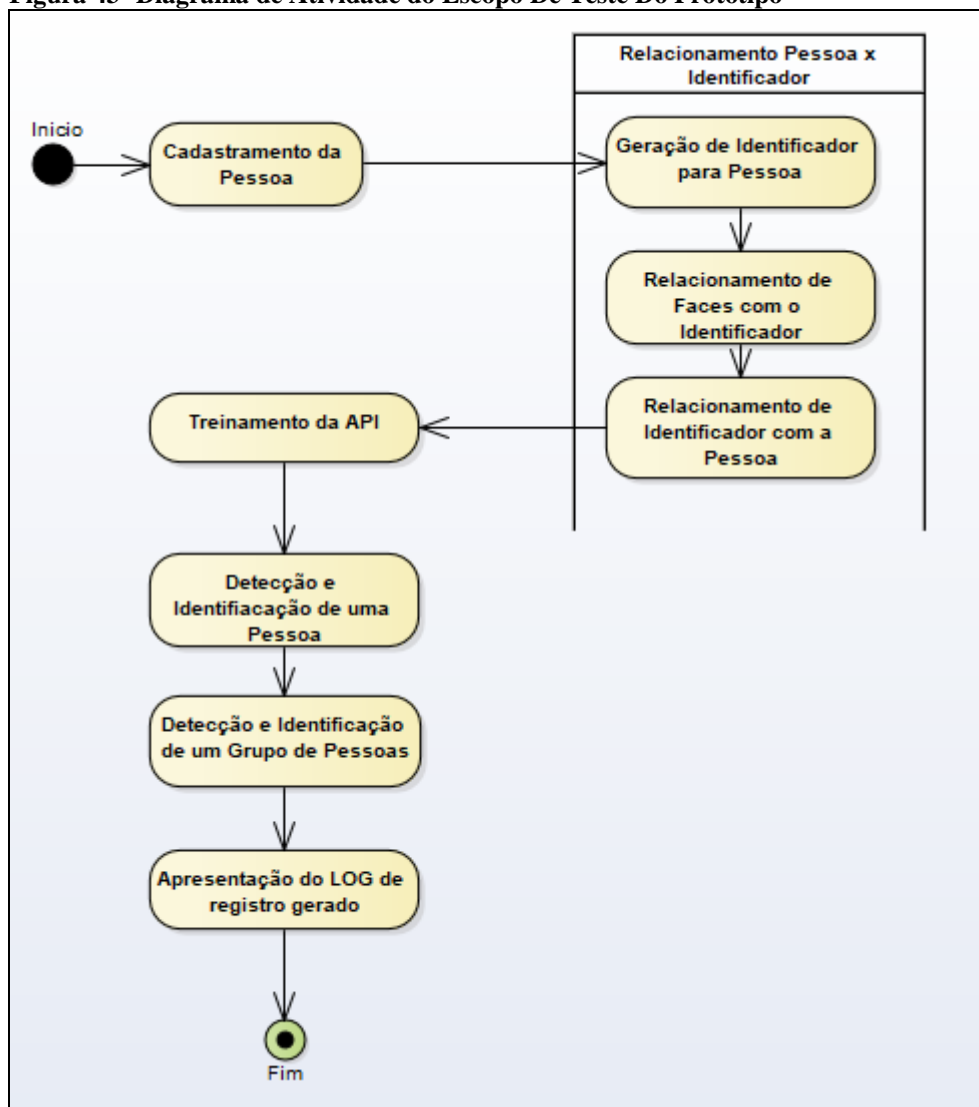


Fonte: acervo do autor (2018).

4.5 EXPERIMENTO DE AVALIAÇÃO DO PROTÓTIPO DE DETECÇÃO E IDENTIFICAÇÃO DE PESSOAS ATRAVÉS DE BIOMETRIA FACIAL

Após a implementação de todas as funcionalidades, o protótipo foi apresentado a um grupo de pessoas com o intuito de testar e analisar a real funcionalidade do protótipo. Os testes foram realizados seguindo um escopo predefinido conforme demonstra a Figura 43 abaixo.

Figura 43- Diagrama de Atividade do Escopo De Teste Do Protótipo



Fonte: acervo do autor (2018).

O experimento teve um total de 56 participantes, que foram submetidos ao cadastro pessoal da face com posterior detecção e identificação pessoal e em grupo de pessoas e no final das identificações foi apresentado os registros de *LOG's* gerados para os participantes.

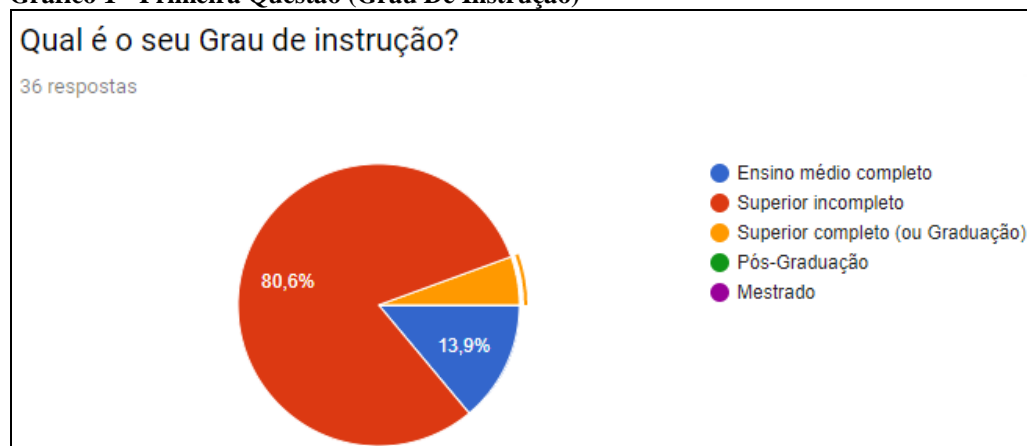
4.6 RESULTADOS OBTIDOS

Nesta seção é apresentada a avaliação dos resultados obtidos através das respostas de um formulário aplicado no período de 29 de Outubro de 2018 até 8 de Novembro de 2018. O formulário foi disposto digitalmente para 56 pessoas que participaram do experimento, das pessoas convidadas a responder o formulário 36 efetuaram a avaliação.

O formulário aplicado foi apresentado em duas partes, a primeira solicitando que as respostas informadas no formulário fossem realizadas de forma séria e comprometida com a verdade, a segunda parte foi composta de 16 perguntas com relação ao conhecimento do avaliador e a avaliação do protótipo demonstrado no experimento.

Ao avaliar os resultados obtidos, pode-se perceber que a primeira questão tinha como objetivo identificar qual o grau de instrução do avaliador. Como pode-se observar no Gráfico 1, 80,6% dos respondentes possuem ensino superior incompleto, 13,9% possuem ensino médio completo e apenas 5,6% possuem ensino superior completo (ou graduação).

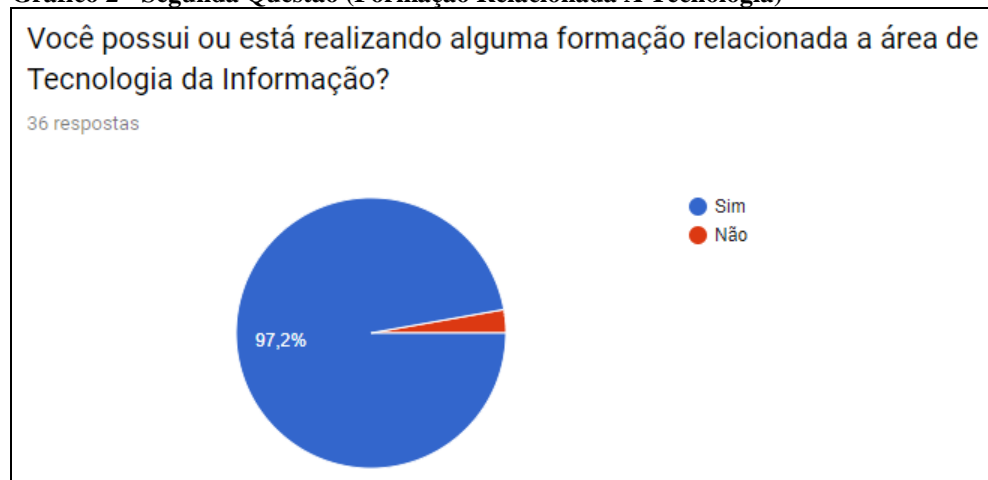
Gráfico 1 - Primeira Questão (Grau De Instrução)



Fonte: acervo do autor (2018).

A segunda questão vem com o pressuposto de saber se os respondentes possuem ou estão realizando formação relacionada a área de tecnologia da informação, sendo que apenas 2,8% dos respondentes não possuem nem realizam formação relacionada, demonstrando que a 97,2% possuem ou estão realizando, conforme demonstra o Gráfico 2.

Gráfico 2 - Segunda Questão (Formação Relacionada A Tecnologia)

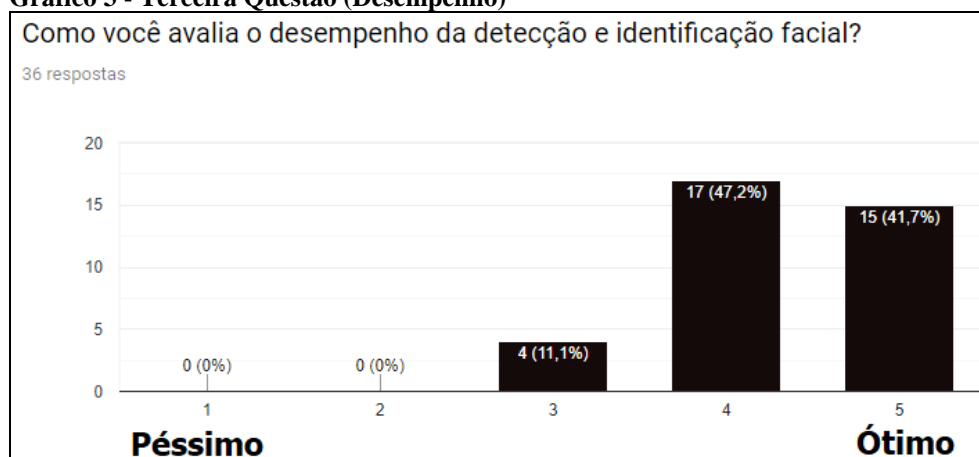


Fonte: acervo do autor (2018).

Com base na análise desempenhada nas avaliações das questões um e dois, pode-se afirmar que os respondentes estavam aptos a realizar o experimento e realizar a avaliação do protótipo.

A terceira questão busca realizar a avaliação do desempenho da detecção e identificação facial proposta no protótipo, sendo que uma representa o valor de desempenho péssimo e 5 representa o valor de desempenho ótimo, obteve-se como respostas 11,1% para nota 3 de desempenho 47,2% para nota 4 de desempenho e 41,7% para nota 5 de desempenho.

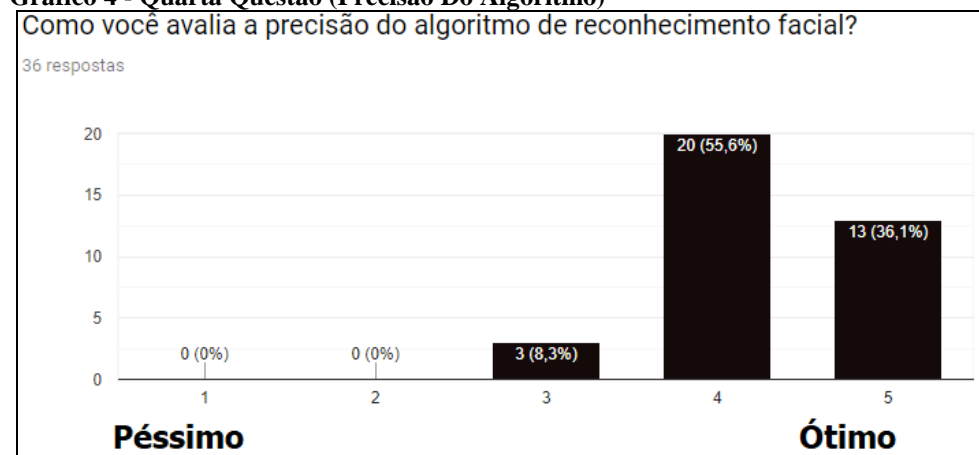
Gráfico 3 - Terceira Questão (Desempenho)



Fonte: acervo do autor (2018).

A quarta questão propõe avaliar a precisão do algoritmo utilizado no protótipo para realizar os reconhecimentos dos respondentes, sendo que 1 representa o valor péssimo e 5 representa o valor ótimo, obteve-se como resposta 8,3% para 3 como preciso, 55,6% para 4 como preciso e 36,1% como 5 para preciso, conforme demonstra o Gráfico 4. Analisando as avaliações das questões três e quatro, nota-se que o desempenho do protótipo tende a ter um desempenho e precisão muito elevados.

Gráfico 4 - Quarta Questão (Precisão Do Algoritmo)

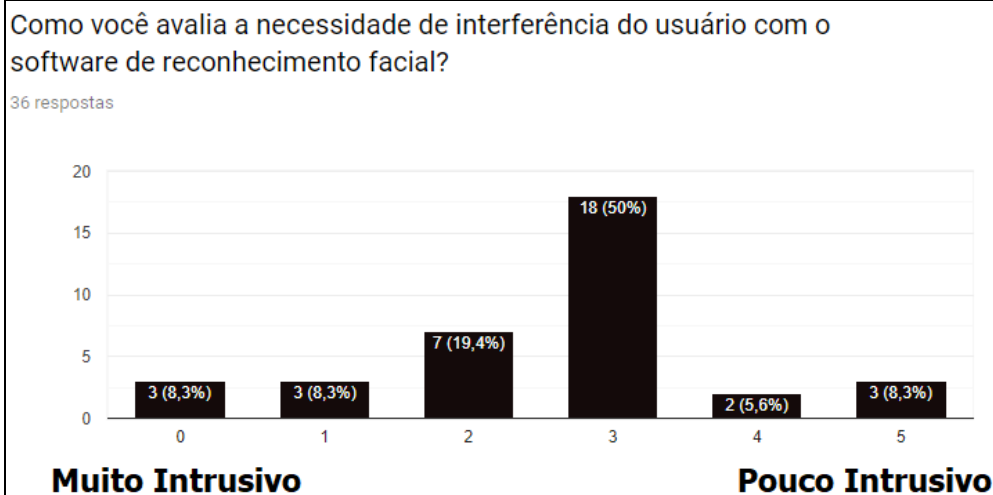


Fonte: acervo do autor (2018).

A quinta questão tem como pressuposto avaliar a necessidade de interferência que o protótipo deveria ter da pessoa para detecta-la e identifica-la, tendo 0 como valor alto para intrusão do usuário e 5 para um valor baixo de intrusão. Obtendo como resposta 8,3% para valor 0, 1 e 5 de intrusão, 19,4% para valor 2 de intrusão, 50% para valor 3 de intrusão, 5,6%

para valor 4 de intrusão, desta forma notando-se o protótipo possui uma intrusão média tendendo para pouca intrusão.

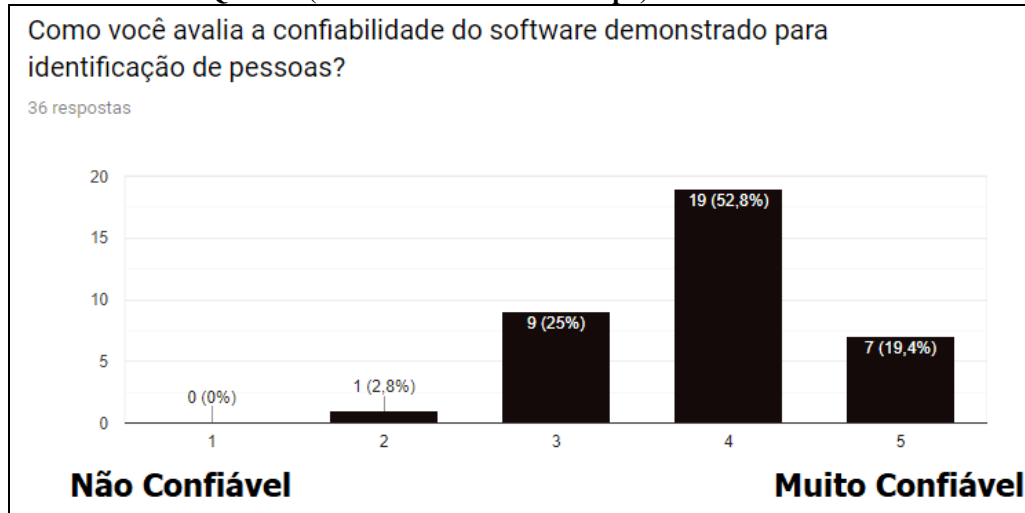
Gráfico 5 - Quinta Questão (Interferência Do Usuário)



Fonte: acervo do autor (2018).

A sexta questão tem como intenção avaliar a confiabilidade dos respondentes nas identificações realizadas pelo protótipo, definindo 1 para não confiável e 5 para muito confiável. Obtendo-se como resposta 2,8% o valor 2 de confiabilidade, 25% o valor 2 de confiabilidade, 52,8% o valor 4 de confiabilidade e 19,4% o valor 5 de confiabilidade, dispondo assim que o protótipo possui uma confiabilidade alta realização de identificação de pessoas.

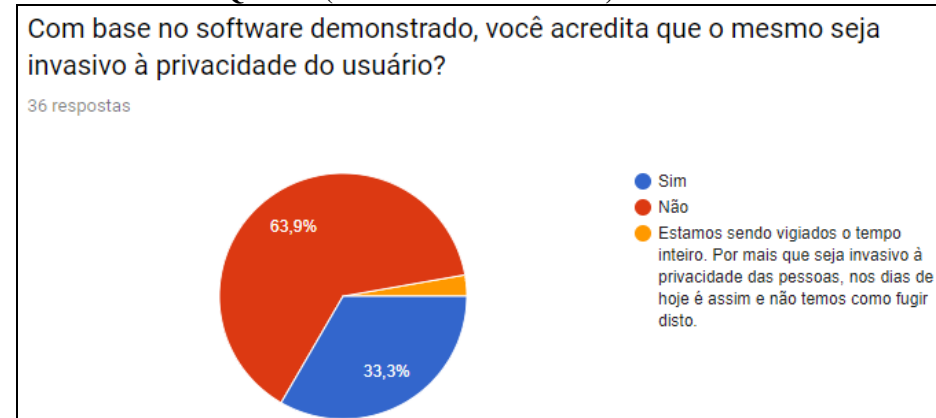
Gráfico 6 - Sexta Questão (Confiabilidade Do Protótipo)



Fonte: acervo do autor (2018).

O Gráfico 7 apresenta a opinião dos respondentes sobre sétima questão que trata da invasão da privacidade referente a detecção e a identificação de pessoas por parte do protótipo. Notando que 63,9% dos respondentes acham que o protótipo não invade a privacidade dos usuários, já 33,3% acha que o protótipo é invasivo a privacidade e apenas 2,8% dizem que “Estamos sendo vigiados o tempo inteiro. Por mais que seja invasivo à privacidade das pessoas, nos dias de hoje é assim e não temos como fugir disto.”, deixando como entendimento que também acha invasivo, porém hoje não existem maneiras de fugir das invasões de privacidade.

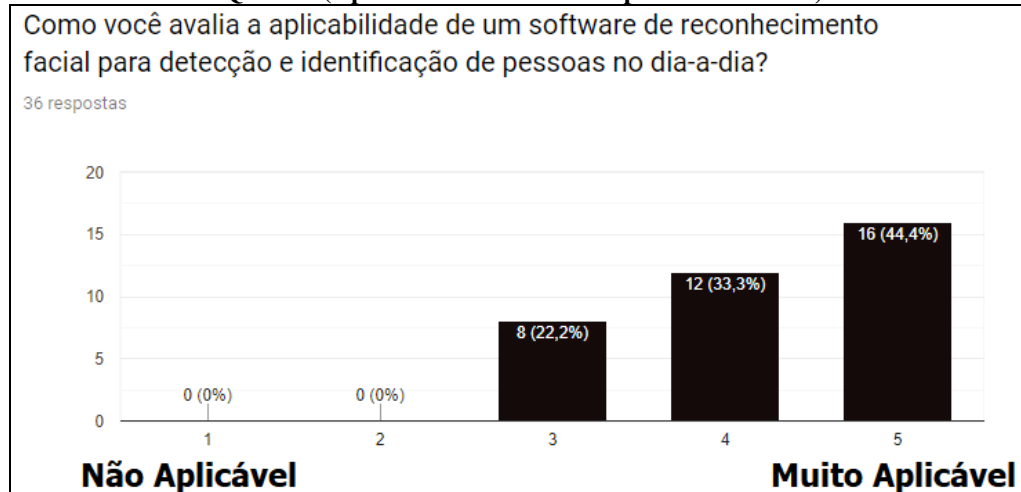
Gráfico 7 - Sétima Questão (Privacidade Do Usuário)



Fonte: acervo do autor (2018).

O Gráfico 8 apresenta a avaliação da oitava questão que se refere a aplicabilidade do protótipo de detecção e identificação de pessoas através de biometria facial no dia-a-dia, sendo que o valor 0 representa que o protótipo não é aplicável e o valor 5 representa que o protótipo é muito aplicável. Desta forma foram obtidas como resposta 22,2% para valor 3 de aplicabilidade, 33,3% para valor 4 de aplicabilidade e 44,4% para valor 5 de aplicabilidade, conforme demonstra o Gráfico 8, sugerindo que o protótipo poderia na maior parte aplicável a rotina diária de detecção e identificação de pessoas.

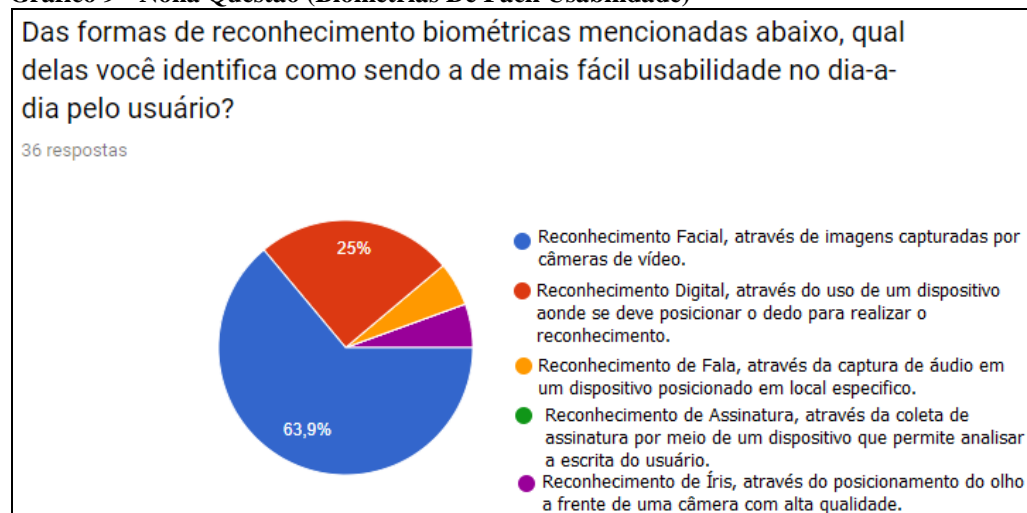
Gráfico 8 - Oitava Questão (Aplicabilidade Do Protótipo No Dia-A-Dia)



Fonte: acervo do autor (2018).

De acordo com as informações apresentadas no Gráfico 9, que tem como objetivo avaliar qual forma de reconhecimento biométrico seria a de mais fácil usabilidade no dia-a-dia pelos respondentes. As respostas obtidas para a nona questão foram de 5,6% para Reconhecimento de Iris, 5,6% também para reconhecimento de Fala, 25% para reconhecimento através de digital e 63,9% para reconhecimento facial e 0% para reconhecimento por assinatura.

Gráfico 9 - Nona Questão (Biometrias De Fácil Usabilidade)

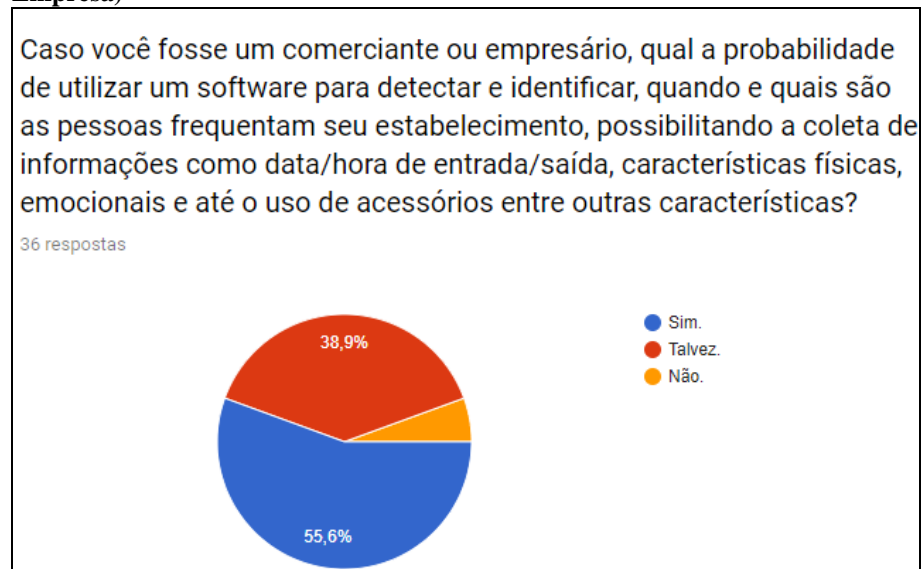


Fonte: acervo do autor (2018).

A Décima questão demonstrada no Gráfico 10, tem como objetivo avaliar a probabilidade do uso pelo respondente caso o mesmo fosse um comerciante ou empresário um *software* para detecção e identificação de pessoas por biometria facial que frequentam o

estabelecimento para coleta de informações pessoais, obtendo-se como respostas 5,6% que não teriam a probabilidade de uso, 38,9% para que talvez teriam a probabilidade de uso e 55,6% das respostas que teriam a probabilidade de uso.

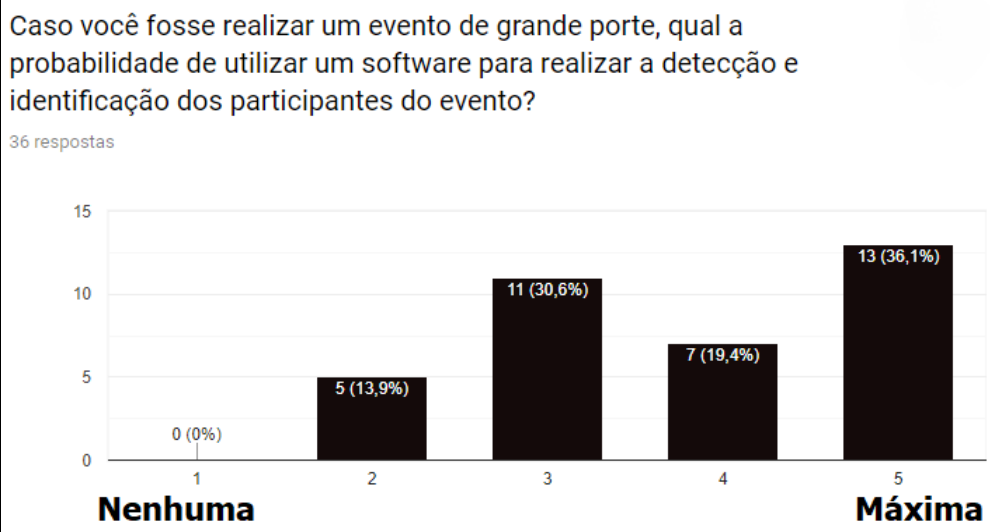
Gráfico 10 - Décima Questão (Probabilidade De Usabilidade Do Protótipo No Comércio Ou Empresa)



Fonte: acervo do autor (2018).

Já a décima primeira questão vem ao encontro da décima, porém relacionada a utilização em eventos de grande porte, sendo o valor 1 para nenhuma probabilidade e o valor 5 para máxima probabilidade, coletando como respostas 13% para o valor 2 de probabilidade de uso, 30,6% para o valor 3 de probabilidade de uso, 19,4% para o valor 4 de probabilidade de uso e 36,1% para o valor 5 de probabilidade de uso. Com base nessas informações pode-se notar que em todos os casos houve algum valor para a probabilidade de uso, pois nenhum valor 1 foi informado como resposta.

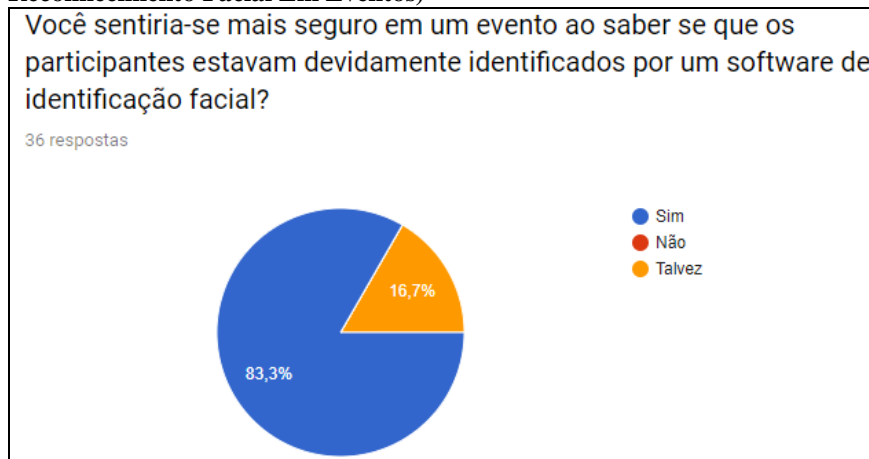
Gráfico 11 - Décima Primeira Questão (Probabilidade De Usabilidade Do Protótipo Em Eventos)



Fonte: acervo do autor (2018).

A décima segunda questão apresentada pelo Gráfico 12, tem como objetivo avaliar a sensação de segurança do usuário em um evento que aplicasse o reconhecimento facial de pessoas, obtendo como resposta 16,7% como talvez se sentiram mais seguros e 83,3% para que se sentiram mais seguros, sendo que nenhum dos respondentes respondeu que não se sentiram mais seguros.

Gráfico 12 - Décima Segunda Questão (Sentimento De Segurança Com Uso De Reconhecimento Facial Em Eventos)



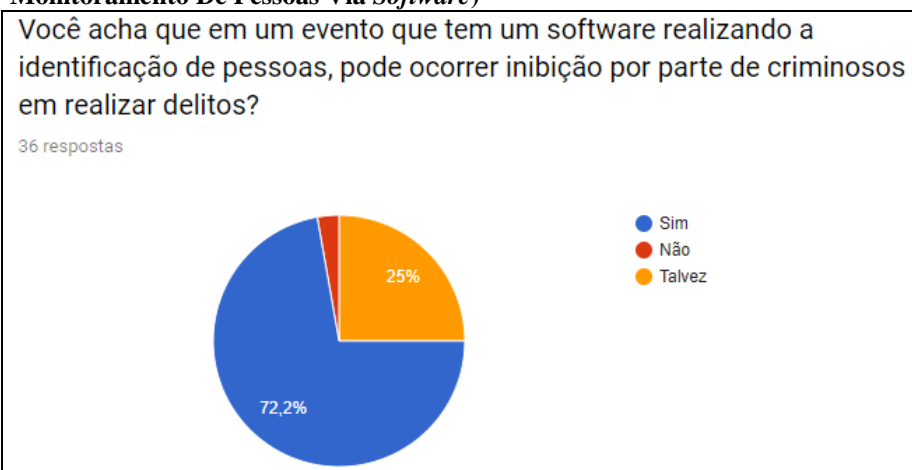
Fonte: acervo do autor (2018).

Para a décima terceira questão teve como objetivo avaliar a opinião dos respondentes sobre a possibilidade de inibição da ocorrência de delitos em eventos com identificação de pessoas por *softwares* de identificação facial, obtendo como respostas 2,8% para que não inibição de ocorrência de delitos, 25% para que talvez ocorra a inibição de ocorrência de

delitos e 72,2% para que ocorreria a inibição da ocorrência dos delitos, conforme demonstra o Gráfico 13.

Com base na décima segunda e décima terceira questão, nota-se que as pessoas tendem a sentir-se mais seguras ao frequentar eventos em que as pessoas estejam devidamente identificadas através de *software*, acreditando que os delitos seriam inibidos.

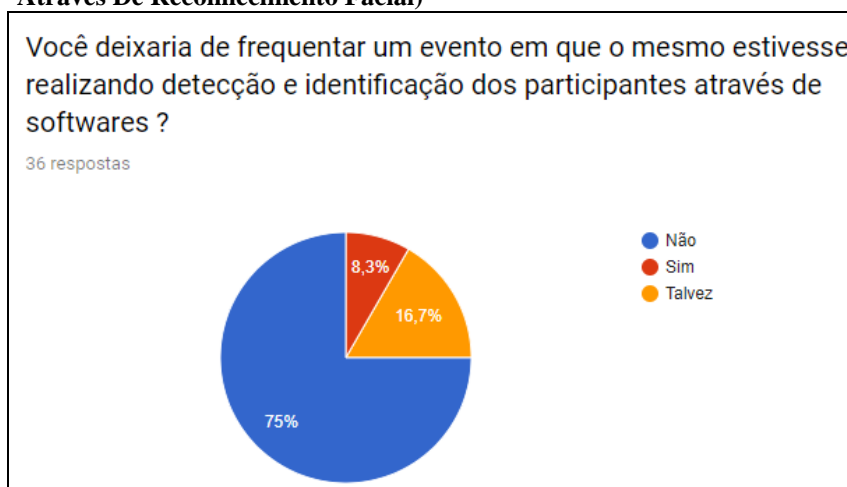
Gráfico 13 - Décima Terceira Questão (Inibição De Delitos Em Eventos Que Possuem Monitoramento De Pessoas Via *Software*)



Fonte: acervo do autor (2018).

A décima quarta questão teve o intuito de identificar se os respondentes deixariam de frequentar eventos que realizassem a identificação de pessoas através de reconhecimento facial, obtendo como resposta 8,3% para que deixaria de frequentar o evento, 16,7% para talvez deixariam de frequentar o evento e 75% não deixaria de frequentar o evento, conforme o Gráfico 14. Desta forma deixa-se claro que a maioria das pessoas não deixariam de frequentar um evento apenas por ele possuir identificação de pessoas por biometria facial.

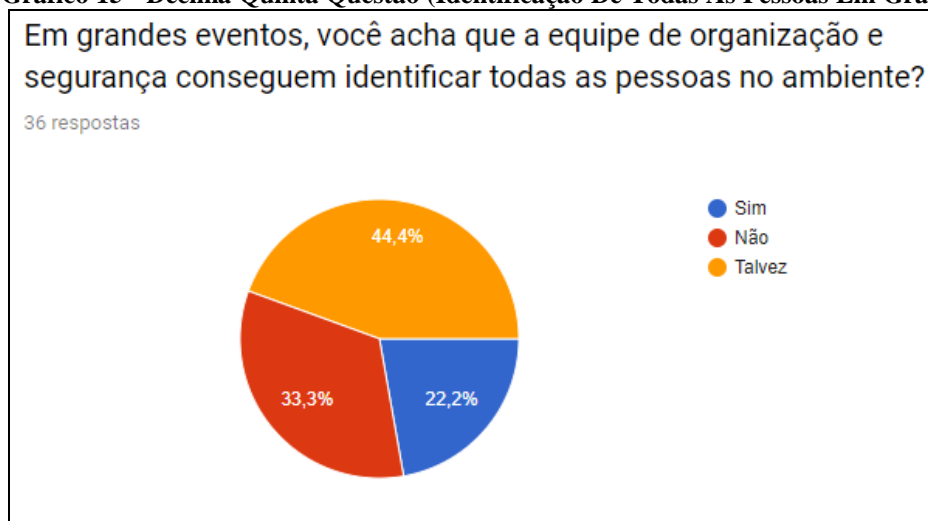
Gráfico 14 - Décima Quarta Questão (Presença em Eventos Com Identificação De Pessoas Através De Reconhecimento Facial)



Fonte: acervo do autor (2018).

A décima quinta questão apresentada no Gráfico 15, tem como objetivo avaliar a opinião dos respondentes quanto a capacidade de equipes de organização e segurança realizarem a identificação de todas as pessoas que estão presentes em um evento de grande porte, obtendo como resposta 44,4% que acham que talvez seja possível identificar todas as pessoas 33,3% que acham não ser possível identificar todas as pessoas e 22,2% que acham que é possível identificar todas as pessoas.

Gráfico 15 - Décima Quinta Questão (Identificação De Todas As Pessoas Em Grandes Eventos)

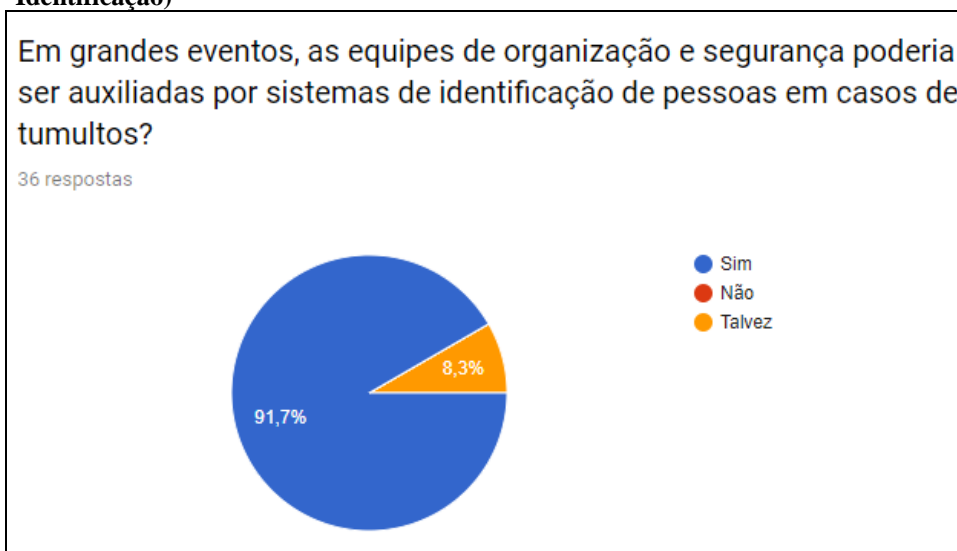


Fonte: acervo do autor (2018).

Conforme apresentado pelo Gráfico 16, que se refere a questão decima sexta buscando avaliar a possibilidade de equipes de organização e segurança de grandes eventos serem

auxiliadas na identificação de pessoas em tumultos por sistemas de reconhecimento, obteve-se como resposta 8,3% que identificaram como talvez seria possível auxiliar, 91,7% que identificaram como possível as equipes serem auxiliadas por sistemas de identificação de pessoas.

Gráfico 16 - Décima Sexta Questão (Auxílio Na Identificação De Pessoas Por Sistemas de Identificação)



Fonte: acervo do autor (2018).

5. CONCLUSÕES

O presente trabalho é apresentado com o intuito de evidenciar o desenvolvimento de um protótipo de *software* para detecção, identificação e registro de pessoas através de visão computacional e biometria facial, possibilitando um controle automatizado de multidões de pessoas presentes em um determinado ambiente.

Tecnicamente para realizar o desenvolvimento desta solução fez-se necessária a utilização da linguagem de programação C#, integrando os frameworks Microsoft .NET, AForge.NET e *API* de Detecção e Identificação Facial da Azure, juntamente com o banco de dados PostgreSQL, assim possibilitando atender o objetivo geral proposto.

O desenvolvimento do protótipo teve início com a realização do projeto geral do protótipo e suas comunicações realizadas com a *API*, seguido da implementação das interfaces de consulta e cadastros e seguido da implantação da captura de imagens por dispositivos de entrada de vídeo e então finalmente a realização das interações com a *API*, para realizar os processos de detecção e identificação baseados em Inteligência Artificial.

A *API* de Face do Azure utilizada foi o modelo de custo gratuito, sendo disponibilizada 20 transações por minuto e 30.000 transações por mês, sendo suficiente para o desenvolvimento do protótipo documentado e da execução dos experimentos relacionados ao trabalho.

Porém pode-se notar um pouco de limitações no modelo gratuito, sendo que para a aplicação em locais de grande fluxo de pessoa seria necessário a detecção e identificação a cada 2 ou 3 segundos, pois as pessoas poderiam sair do foco da câmera caso o tempo de envio entre as transações fossem limitadas.

Posteriormente ao termino do desenvolvimento do protótipo ocorreram os testes, experimentos e então a avaliação do mesmo, sendo aplicada a 56 pessoas com conhecimento na área de tecnologia da informação. O formulário aplicado possuía 16 questões elaboradas, sendo que 2 eram destinadas a avaliar o conhecimento dos respondentes e 14 questões relacionadas a utilização do protótipo apresentado durante o experimento.

O formulário teve como propósito a verificação da aceitação, desempenho, confiança e possíveis aplicações do protótipo de detecção e identificação facial. Ao realizar a análise das respostas do formulário foi possível identificar que o desempenho, confiança e aceitação tiveram uma boa avaliação e a questão da aplicabilidade tendeu a um valor de alta aplicabilidade.

Por fim com os resultados obtidos foi possível comprovar os pressupostos da pesquisa. Ratificando que é possível desenvolver um sistema para detecção e identificação de pessoas em multidões através de *software* para reconhecimento biométrico facial. Comprovou-se também que o sistema possui aceitabilidade, confiança e desempenho elevados e que possui várias possibilidades de aplicações.

5.1 TRABALHOS FUTUROS

Para realização de trabalhos futuros, propõem-se:

- Unificação da rotina de cadastro de pessoas com a geração de identificadores da pessoa.
- Criação de um cadastro para o Grupo de Pessoas.
- Criação de um armazenamento para as imagens de faces detectadas e não identificadas.
- Criação de rotina de controlador para identificação, possibilitando alterar o percentual de confiança da identificação de faces.

REFERÊNCIAS

- ALBERGARIA, Elisa Tuler de; SANTOS, Katia C. Lage; ALVIM JÚNIOR, Mário Sérgio Ferreira. **Reconhecimento de Faces Utilizando Programação Genética**. Universidade Federal de Minas Gerais, 2006. Disponível em: <<http://homepages.dcc.ufmg.br/~nivio/cursos/pa06/seminarios/seminario12/seminario12.pdf>>. Acesso em: 20 abr. 2018.
- ALMEIDA, J. M.; BENTO, H. **Reconhecimento de padrões através de Eigenfaces**, Sinfic, 2018. Disponível em: <<http://www.sinfic.pt/SinficWeb/displayconteudo.do2?numero=44666>>. Acesso em: 05 mai. 2018.
- BRLINK. **Descobrimo Cloud: Microsoft Azure ou AWS, qual é melhor?** BRLink, 2016. Disponível em: <<https://www.brlink.com.br/blog/aws/descobrimo-cloud-microsoft-azure-ou-aws-qual-e-melhor/>>. Acesso em: 25 abr. 2018.
- CAMPOS, Teófilo Emídio de. **Técnicas de Seleção de Atributos e de Classificação para Reconhecimento de Faces**, Universidade de São Paulo, 2018. Disponível em: <<http://www.vision.ime.usp.br/~teo/publications/qualificacao/>>. Acesso em: 31 mar. 2018.
- CARVALHO, André Ponce de Leon F. de. **Redes Neurais**, Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo, 2018. Disponível em: <<http://conteudo.icmc.usp.br/pessoas/andre/research/neural/>>. Acesso em: 05 mai. 2018.
- COPPIN, Ben. **Inteligência Artificial**. 1. Ed. Rio de Janeiro: LTC, 2012.
- DIONISIO, Edson José. **PostgreSQL Tutorial**, DevMedia, 2017. Disponível em: <<https://www.devmedia.com.br/postgresql-tutorial/33025>>. Acesso em: 28 abr. 2018.
- FARIA, Alessandro de Oliveira. **Biometria: Reconhecimento Facial Livre!** Linha de Código, 2018. Disponível em: <<http://www.linhadecodigo.com.br/artigo/1813/biometria-reconhecimento-facial-livre.aspx>>. Acesso em: 20 abr. 2018.
- LUCAS, Luis. **Reconhecimento Biométrico da Íris na Região de Comprimentos de Onda do Infravermelho Próximo e do Visível**, Universidade da Beira Interior, 2011. Disponível em: <<https://ubibliorum.ubi.pt/handle/10400.6/973>>. Acesso em: 10 nov. 2018.
- LUGER, George F.. **Inteligência Artificial**. 3. Ed. São Paulo: Pearson Education do Brasil, 2013.
- MAGNUS, Tiago. **Entenda o que é Inteligência Artificial e como ela pode mudar tudo o que conhecemos**, Transformação Digital, 2018. Disponível em: <<https://transformacaodigital.com/o-que-e-inteligencia-artificial/>>. Acesso em: 05 mai. 2018.
- MAIA, Hugo Leite Florenço. **Detecção e reconhecimento facial por meio de aprendizado de máquina**, Biblioteca Digital da Produção Intelectual Discente da Universidade de Brasília, 2016. Disponível em: <<http://bdm.unb.br/handle/10483/15247>>. Acesso em: 05 mai. 2018.

MARTINS, Leonardo Dias. **Biometria de Digitais**. GTA/UFRJ, 2018. Disponível em: <https://www.gta.ufrj.br/grad/07_2/leonardo/>. Acesso em: 30 mar. 2018.

MICROSOFT. **Microsoft Azure**, Microsoft, 2018. Disponível em: <<https://azure.microsoft.com/pt-br/>>. Acesso em: 25 abr. 2018.

POSTGRESQL. **O que é PostgreSQL?** PostgreSQL, 2018. Disponível em: <<http://pgdocptbr.sourceforge.net/pg82/intro-what-is.html>>. Acesso em: 28 abr. 2018.

RUSSELL, Stuart; NORVIG, Peter. **Inteligência Artificial**. 3. Ed. Rio de Janeiro: Elsevier, 2013.

SANCHEZ, Marcos Paulo. **Registro de Frequência de Descendentes por Meio de Biometria**, Universidade de Taubaté, 2011. Disponível em: <http://bdtd.ibict.br/vufind/Record/ITAU_5003197ca1828f2463cc7bac08bf07aa>. Acesso em 10 nov. 2018.

SANTOS, Júlio Sérgio Quadro dos. MACHADO, Vanessa Lago. FIGUEIREDO, José Antônio Oliveira de. **Análise da viabilidade de Reconhecimento Facial e Autenticação em aplicações mobile**, EATI, 2017. Disponível em: <<http://eati.info/eati/2017/assets/anais/Curtos/C174.pdf>>. Acesso em: 21 abr. 2018.

SANTOS, Marco Aurélio da Silva. **Inteligência Artificial**, Brasil Escola, 2018. Disponível em <<https://brasilecola.uol.com.br/informatica/inteligencia-artificial.htm>>. Acesso em: 05 mai. 2018.

SILVA, Gustavo Moreira da et al. **Interface computacional de biometria como ferramenta de apoio à perícia de confronto de voz**, Universidade Estadual Paulista, 2012. Disponível em <http://jaguar.fcav.unesp.br/RME/fasciculos/v30/v30_n4/A4_Angela.pdf>. Acesso em: 10 nov. 2018.

TATIBANA, Cassia Yuri; KAETSU, Deisi Yuki. **Redes Neurais**, Departamento de Informática da Universidade Estadual de Maringá, 2018. Disponível em: <<http://www.din.uem.br/ia/neurais/#artificial>>. Acesso em: 05 mai. 2018.

TIVIT CLOUD. **Qual é o melhor free tier? AWS, Azure ou Google Cloud**, TIVIT BLOG, 2017. Disponível em: <<http://blog.tivit.com/qual-e-o-melhor-free-tier-aws-azure-ou-google-cloud>>. Acesso em: 25 abr. 2018.