1. Materiais e Metodos

Neste capitulo serão abordados todos os conhecimentos necessários para o desenvolvimento do projeto, como as características da Carteira Nacional de Habilitação, as bibliotecas e linguagens utilizadas para o desenvolvimento e principais algoritmos utilizados na implementação.

* 1. Ambiente de desenvolvimento

Uma linguagem de programação é uma linguagem artificial projetada para comunicar instruções a uma máquina, especialmente um computador. Linguagens de programação podem ser utilizadas para criar os programas que controlam o comportamento de uma máquina (AABY, 2004).

A descrição de uma linguagem de programação é geralmente dividida em dois componentes da sintaxe (forma) e semântica (significado). Alguns idiomas são definidos por um documento de especificação, como por exemplo, a linguagem de programação C é especificada por um padrão ISO. (ISO/IEC, 2011).

* + 1. Linguagem C#

C# é uma linguagem de programação orientada a objeto desenvolvida pela Microsoft em meados de 1999 com base na linguagem C++ que permite criar uma grande variedade de aplicativos seguros e robustos que são executados no .NET Framework. A intenção da Microsoft foi criar uma linguagem de uso geral simples, robusta, orientada objetos e fortemente tipada. É possível usar C# para criar aplicativos cliente do Windows, Web Services, aplicativos cliente-servidor, entre outros.

* + 1. Biblioteca OpenCV e EmguCV
       1. OpenCV

OpenCV (Open Source Computer Vision Library) é uma biblioteca livre ao uso acadêmico e comercial, para o desenvolvimento em linguagem C e C++, de aplicativos na área de visão computacional (OPENCV, 2013). Esta biblioteca possui mais de 2500 algoritmos otimizados, desde os mais simples até os mais modernos, tais como os de Machine-Learning.

O OpenCV pode ter ser utilizado no desenvolvimento de aplicativos com as mais diversas aplicações, desde programas simples como colagem de imagens até programas complexos como auxilio na navegação robótica. Segundo (Rui Miguel, 2013 - Superfícies Interativas com Kinect)

O OpenCV foi projetado especialmente para eficiência computacional e têm enorme foco em aplicações em tempo real, que utilizam processamento de visão por computador. Foi desenvolvido em C/C++ otimizado e permite tirar partido de processamento multi-core. Confere, ainda, um enorme grau de abstração da programação que requer este tipo de processamento.

* + - 1. EmguCV

EmguCV tem como principal função adaptar o código na biblioteca OpenCV para que possa ser utilizado em plataformas e linguagens compatíveis com o .NET Framework, como C#, VB, VC++, entre outros. Dessa forma, o EmguCV permite a implementação de funcionalidades do OpenCV atrás do Visual Studio em linguagens de programação como o C#.

* + 1. Tesseract

O Tesseract é a biblioteca opensource responsável pelo reconhecimento ótico dos caracteres, desenvolvida pela HP entre 1985 e 1995 e a partir de 2006 o projeto foi continuado pela Google. Atualmente o Tesseract é considerado a melhor ferramenta OCR opensource (Bhaskar, 2012).

* + 1. Microsoft Visual Studio

Visual Studio é o ambiente de desenvolvimento (IDE) da Microsoft para construção de aplicações em C#, Visual Basic, Visual C#, C++, JavaScript, entre outras linguagens. Com esta ferramenta é possível criar as mais diversas aplicações desktop, aplicativos móveis, serviços Web, dentre outros. A versão do Visual Studio utilizada para o desenvolvimento deste trabalho é a 2015 com .NET Framework 4.

* 1. Modelagem do sistema
     1. Conceito

A leitura automática de documentos consiste na aquisição e interpretação da informação contida no formato físico do mesmo. Para este processo é utilizado tecnologias para digitalizar os documentos, tais como câmeras e scanners, e software para o reconhecimento de caracteres, o OCR. Dessa forma, ao se scanear um documento, será possível permitirá não somente a transformação para o formato digital como também obter os dados para o preenchimento de um cadastro pessoal em um sistema de informação de uma empresa, por exemplo.

Dessa forma, percebe-se a importância e utilidade desses sistemas de leitura automática de documentos, uma vez que reduz o trabalho manual para interpretar e digitar os dados do documento, reduzindo o tempo e os custos referentes a estas atividades.

Para interpretação dos dados é necessário definir um modelo de identificação do documento. Neste trabalho, será utilizada a localização das regiões ou segmentos de interesse para atribuir sentido ao dado lido. Por exemplo, para a leitura do nome completo é necessário definir as coordenadas (x,y), a largura (L) e altura (A) do campo no documento, como pode ser visto na imagem XXX.

[imagem CNH desenhada]

Outro fator importante para este sistema é a independência em relação à digitalização e arquivamento dos dados. Dessa forma, é possível alterar o design da tela de cadastro, ou a forma de armazenamento dos dados sem que seja necessária uma atualização do sistema de leitura automática. Ou seja, a leitura deve acontecer de forma o sistema a leitura é transparente para o desenvolvedor do SI, como pode ser visto na imagem XX.

[imagem SI]

* 1. Leitura Automática de Documento
     1. Comunicação entre processos

Como foi explicado na seção 3.2.1, o sistema foi projetado para ser independente da interface gráfica e do modelo de dados do usuário. Dessa forma, o protótipo do sistema desenvolvido para este trabalho é constituído de dois projetos executáveis.

O primeiro executável será responsável pela entrada de dados do sistema e receberá o resultado do processamento do documento, exibindo os dados em uma interface gráfica e com a possibilidade de salvar os dados em um arquivo XML. Como foi explicado anteriormente, o projeto foi desenvolvido desta forma para deixar o sistema de leitura automática de dados independente de interfaces gráficas e modelo de dados, permitindo a personalização.

O segundo executável é o sistema de leitura de dados em si. Para a comunicação entre os processos foi utilizado o protocolo de comunicação HTTP. O sistema de leitura automática é um servidor HTTP, espera uma requisição em uma porta, com a imagem da CNH passada como parâmetro. Em seguida é realizado o processamento da imagem e o resultado é devolvido na forma de json com os dados lidos.

* + 1. Tratamento da imagem e Leitura do Documento

Como pode ser visto na seção [XXX] um dos primeiros passos de sistemas de visão computacional é o processamento da imagem. No caso deste sistema, é recebida uma imagem colorida do documento da CNH como entrada. O primeiro tratamento é a conversão da mesma para escala de cinza. Em seguida, são iniciadas duas threads de processamento de imagem.

A primeira recebe a imagem e tenta reconhecer um rosto humano no documento. Caso não seja encontrado, o sistema é automaticamente invalida o mesmo, uma vez que como pode ser visto na imagem XXX a CNH possui uma foto de rosto e não encontrar esta face significa que ou a imagem recebida realmente não é do documento ou não possui qualidade suficiente para a identificação. Para a identificação de faces foi utilizado um método chamado Haar Trainning. Neste método é utilizado um arquivo XML, chamado Haar Classifier, que contém as informações do objeto que se deseja identificar, no caso uma face humana. No caso deste sistema, foi utilizado o arquivo disponibilizado pelo EmguCV.

Na segunda thread é feito um processamento a fim de segmentar a imagem nas regiões de interesse. Inicialmente é detectada as bordas da imagem, em seguida é utilizado a transformada de Hough para as linhas da imagem, para segmentar as área de dados do documento, que possui o contorno delimitado por uma linha escura. Outra abordagem possível para este problema seria identificar os contornos da imagem, identificando a área desejada. Porém esta abordagem se mostrou menos efetiva uma vez que devido à irregularidade da imagem, que possui a descrição do campo, não foi possível identificar os contornos desejados corretamente. Dessa forma optou-se pela abordagem das linhas.

O sistema aguarda o resultado destas duas threads e após obtê-lo é iniciada a busca pelas regiões de interesse. Essa busca se baseia na localização dos campos em relação à face. Ou seja, inicialmente, busca-se o nome, que é o primeiro campo detectado acima da face. Após identificar as linhas que compõe o campo, é possível extrair as informações do ponto inicial do campo (x,y), do comprimento do campo e da altura, com base nas linhas que se interceptam em um ângulo de 90º. Uma vez identificado estas componentes, é extraída a região de interesse da imagem e esta passa pelo reconhecimento de caracteres.

Foram iniciados três instancias do leitor OCR, com três dicionários possíveis. Este tratamento foi realizado para reduzir a possibilidade de erros do sistema, uma vez que cada campo possui uma determinada característica, por exemplo, o nome pessoal possui letras de A à Z enquanto campos como o CPF e datas só possuem números e alguns pontos .

Após a leitura de todos os dados, é montado um objeto e ele é retornado na requisição HTTP, que foi comentada na seção XXX. No diagrama da figura XX é possível ver um esquema do funcionamento do sistema.

<http://www.detran.pr.gov.br/modules/catasg/servicos-detalhes.php?tema=motorista&id=130>

EMGU. EmguCV. Disponível em: . Acesso em: 11 jun. 2014.

OPENCV. (2013). OpenCV: About OpenCV. Disponivel em: Acessado em: 17 Ago 2013.

<http://opencv.org/about.htm>

COUTO, Leandro Nogueira . Sistema para localização robótica de veículos autônomos baseado em visão computacional por pontos de referência . Biblioteca Digital da USP. São Carlos . 2012. Disponível em: . Acesso em: 18 Novembro 2012.

33. WATERS, D. J. Do microfilme à imagem digital: projeto conservação

preventiva em bibliotecas e arquivos. Disponível em:

<http://www.cpba.net>. Acesso em: 29 abr. 2003.

<http://www.prefeitura.sp.gov.br/cidade/upload/cpba_49_1253284217.pdf>

SHAPIRO, L. G. (2001). *Computer Vision* (1 ed., Vol. 1). Prentice Hall.

ZUECH, N. (1988). *Understanding and Applying Machihe Vision* (2 ed.). Yardley, Pennsylvania: Marcel Dekker INC.

DAVIES, R. (2005). *Machine Vision: Theory, Algorithms, Practicalities* 1 ed., Vol. 1. Morgan Kaufmann.

3.3.1 Aquisicão das Imagens

Para realizar a aquisição das imagens, utilizou-se um código disponível na *Internet* do programa *C# Webcam Capture* desenvolvido por Philip Pierce (PIERCE, 2003) que foi adicionado ao projeto do sistema biométrico no *Visual Studio* e adaptado conforme a necessidade. Esse programa captura a imagem de uma *webcam* conectada ao computador, mostrando a imagem a uma certa taxa de tempo, definida pelo usuário.

As imagens são adquiridas em uma resolução de 640 x 480 *pixels*.