Dispositivo para auxiliar deficientes visuais na leitura de documentos

Davi Antônio da Silva Santos
Graduando em Engenharia Eletrônica
Universidade de Brasília
Gama, Brasil
Email: antoniossdavi@gmail.com

Victor Aguiar Coutinho
Graduando em Engenharia Eletrônica
Universidade de Brasília
Gama, Brasil
Email: victor.a.coutinho@gmail.com

***Resumo*—Devido a problemas de deficientes visuais ao acesso a informação de livros e documentos que não vem com áudio do conteúdo, propõe-se desenvolver um dispositivo que faça a leitura do documento e transforme em áudio para o usuário. Esse sistema deve ser de fácil utilidade e portátil, e com uma câmera com resolução suficiente para a leitura do documento. Ele ajudará na inclusão de alunos nas suas escolas e os ajudará nos estudos.**

I. Introdução

Cerca de 3,46% da população brasileira possui deficiência visual severa[1]. A deficiência visual severa é a que tem maior quantidade de afetados dentro do quadro de deficiências severas [1].

Com base nos dados supracitados, propõe-se um sistema que auxilie a leitura de livros e documentos através da síntese de voz feita a partir do resultado do reconhecimento ótico de caracteres de texto (OCR) de baixo custo utilizando o computador *single-board* Raspberry Pi 3B. A abordagem utilizada é uma adaptação simplificada da adotada por ARRAHMAH, *et al.* [2].

De modo a facilitar a portabilidade do projeto, planeja-se usar o software de reconhecimento ótico de caracteres *Tesseract* e o de conversão de texto para fala *Espeak-ng*, pois possuem suporte a diversas línguas.

II. Proposta do projeto

A. Objetivos

Tem-se com objetivo desenvolver um dispositivo que auxilia estudantes com graus avançados de deficiência visual na compreensão de documentos.

B. Requisitos

Dado o grupo específico de usuários para o qual o dispositivo é voltado, pessoas com baixa visão, elegeu-se como requisitos básicos para a operação ótima:

- sistema portátil e fácil de montar;
- interface fácil de operar;
- baixa necessidade de manutenção;
- câmera com resolução suficiente para os caracteres serem reconhecidos pelo software de OCR;
- câmera com suporte aos *drivers* nativos incluídos no Raspbian;
- dado que o software de OCR possui alto custo computacional, a conversão de texto para voz deve usar o mínimo possível de recursos.

C. Benefícios

Auxiliar a inclusão de alunos com deficiência visual, auxiliando os alunos a lerem documentos que não estejam disponíveis em braille ou audiolivros.

III. Desenvolvimento

Os softwares usados, Tesseract para o reconhecimento ótico de caracteres, e Espeak-ng para conversão de texto para voz e as vozes Mbrola permitem o suporte a diversos idiomas, mas para fins de teste será usado, em primeiro momento, somente o português do Brasil.

O sistema operacional a ser usado no projeto será o Raspbian GNU/Linux, um sistema operacional livre baseado no Debian GNU/Linux otimizado para o Raspberry Pi [4]. O uso do Raspbian GNU/Linux 9.4 (stretch) possibilita o uso de uma ampla biblioteca de pacotes já compilados e que a administração do sistema seja feita de maneira semelhante a de um sistema operacional baseado em Debian Stretch.

O software de reconhecimento ótico de caracteres (OCR) escolhido foi o Tesseract pois este já foi pensado para reconhecer caracteres em situações adversas, principalmente em imagens inclinadas [3]. O dito software já estava compilado e empacotado para o sistema operacional utilizado na versão 3.04.01. Foi necessária a instalação do pacote principal, *tesseract*, o qual já inclui o reconhecimento de caracteres para o inglês, e o para português, o *tesseract-ocr-por*.

A conversão de texto para fala é feito por meio do Espeak-ng, escolhido por manter um baixo consumo de recursos, possuir uma interface altamente configurável na linha de comando e não depender de conexão com a Internet. Tal programa também já estava previamente compilado e foi instalado a partir do pacote *espeak-ng*.

As vozes disponíveis no Espeak-ng soam claras, mas soavam muito artificiais. O Espeak-ng é compatível com as vozes Mbrola, mas o repositório do Raspbian possui apenas os pacotes com as vozes mas não o programa principal, o qual está disponível apenas no formato de pacote fonte. Foi necessário compilar o pacote fonte [5] [6] para que este e as

vozes fossem instaladas. O pacote *mbrola* foi compilado, e as vozes instaladas foram *mbrola-br1* e *mbrola-br2*.

A primeira câmera testada era uma *webcam* com resolução de 640 por 480 pixels. A câmera era acessada através da interface V4L2 (Video for Linux v2) pelo programa FFmpeg, também já compilado e disponível no pacote *ffmpeg*. Os testes realizados identificaram que o equipamento utilizado possuía uma resolução muito baixa, pois a aplicação de OCR muitas vezes não identificava nenhuma palavra.

Os testes preliminares indicaram que a versão mais nova do Tesseract, a 4.0.0 disponível no repositório do Debian GNU/Linux Buster, produzia resultados muito melhores em condições adversas, como imagens de livros com páginas em perspectiva, e que uma resolução de 1280x720 pixels já era o suficiente para bons resultados, mesmo no formato JPEG.

Dado que o sistema atual não apresentava bons resultados devido a baixa resolução da câmera utilizada, esta foi trocada por uma Canon Powershot A430 modificada com o software *CHDK* 1.4.1 100b. A interface entre o Raspbian e a câmera, conectada à Raspberry por um cabo USB, foi feita pelo software *chdk-ptp* r795. Esta configuração garante imagens com resolução de 4 MP e elimina a necessidade de foco manual. Ambos softwares já se encontram como arquivos binários distribuídos pelos sites dos desenvolvedores, mas também possuem código fonte disponível [7] [8].

Outro problema observado na configuração anterior era a orientação trocada da fotografia. Para solucionar tal problema, instalou-se o *imagemagick*, disponível nos repositórios do Raspbian. Este pacote disponibiliza o comando *convert*, o qual pode rotacionar imagens antes de ser realizado o reconhecimento de caracteres por meio do *tesseract*.

Após a realização destas alterações o programa precisou ser modificado. Retirou-se a abertura da webcam usando o *ffmpeg* e a interface *v4l2*. Esta foi substituída pela conexão à câmera por meio do *chdk-ptp*. O programa também teve de ser reorganizado. Para isso, as definições dos caminhos dos binários e dos comandos usados no programa, assim como as dos protótipos das funções utilizadas no programa para inicializar a câmera, tirar a fotografia, inicializar o reconhecimento de caracteres, inicializar a conversão de texto para voz e desconectar a câmera foram colocadas em um *header*, e as definições das mesmas em um arquivo *.c* separado, o que levou à criação de um *Makefile* para facilitar o processo de compilação.

Foi adicionado a biblioteca *gpio_sysfs.h*, porém nela só havia funções para portas como saída. Adicionou então as funções *setGPIO_In* (que tem como entrada o pino desejado e o modo de detecção) e *GPIO_Read*(que tem como entrada o pino desejado). Na função *setGPIO_In* é alterado o arquivo *edge* e o modo de detecção é dado pela variável de entrada do tipo char* *modo*. A função *GPIO_Read* utiliza *polling* para a leitura do pino.

No arquivo *main_functions.c* foi adicionado a função *read_continue* que utiliza as funções criadas na biblioteca *gpio_sysfs.h* e trava o sistema enquanto não é pressionado o botão.

Neste ponto do desenvolvimento, percebeu-se que o Raspberry Pi 3B não era capaz de processar em tempo satisfatório as imagens com a resolução escolhida de 4 MP (resolução 2272x1704 pixels), portanto escolheu-se reduzir a resolução das fotos tiradas pela câmera para 1600x1200 pixels (1,92 MP).

Também foi percebido que o sistema operaria melhor caso fosse usada uma abordagem de controle do fluxo de execução por meio de máquina de estados, de modo a facilitar a autonomia do sistema quando operando. Planejou-se também configurar o sistema para iniciar assim que o Raspberry Pi 3B fosse ligado, ou assim que acontecesse o *login* em um usuário especificado, como, por exemplo, o usuário padrão, *pi*, porém isso não foi possível devido a dificuldades de desenvolvimento.

A. Estrutura

Conforme o projeto foi se desenvolvendo, notou-se cada vez mais a importância de haver uma estrutura para estabilizar a câmera e facilitar o posicionamento dos documentos a serem lidos. Para este propósito, elaborou-se um protótipo de uma estrutura que atenda aos requisitos e seja suficiente para demonstrar o funcionamento básico do projeto.

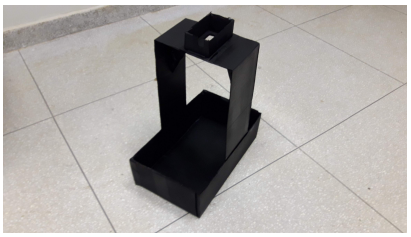

Figura 1. Estrutura montado para o sistema

A estrutura será feita em uma caixa de papelão, a qual terá um furo centralizado por onde passará a objetiva da câmera, de modo que esta consiga capturar todo o documento. Como a câmera possui foco automático e *flash*, o sombreamento e a distância não são grandes problemas. em uma das faces laterais da caixa, de preferência no lado maior caso esta seja um paralelepípedo, ficará a abertura para inserção de documentos, sendo que internamente haverá guias feitas de filetes de papelão para auxiliar na inserção correta do documento.

Outras estruturas foram analisadas antes da escolha da que será implementada, em particular as descritas em [9], as quais permitem que se escaneie maior variedade de materiais. Entretanto, tais estruturas são difíceis de operar considerando-se o público alvo, as pessoas com visão reduzida, pois exigem ajustes na posição da câmera, de luminárias auxiliares e de placas de vidro ou acrílico, e no suporte dos documentos de acordo com o tamanho e o formato do documento escaneado.

Após a escolha da estrutura, construiu-se um apoio de papelão para que fosse possível escanear documentos no formato A4 ou menores com a câmera utilizada para o sistema. Para melhorar a aparência da estrutura e a delimitação da área com texto por meio do sistema de OCR usado foi aplicada tinta em spray da cor preta nas superfícies interna e externa.

A câmera foi colocada no topo da estrutura, no qual havia furos para a objetiva, a lâmpada de flash e o LED do autofoco. O circuito com os botões, os resistores de *pull-up* e o computador Raspberry Pi ficaram no interior. Estavam conectados ao Raspberry a fonte de alimentação USB, um fone de ouvido e o cabo USB da câmera fotográfica, a qual funcionava por pilhas recarregáveis.

B. Código Geral

Foi elaborado uma máquina de estados para automatização do sistema. Essa facilitou a codificação e organização do arquivo principal *prog_main.c*.

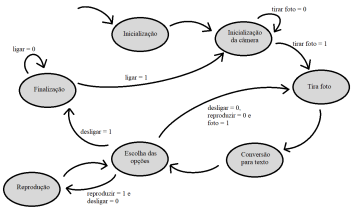

Figura 2. Máquina de estados do código *prog_main.c*

O primeiro estado é a *Inicialização*, nesse é onde é criado as três *threads* para os três botões *ligar/desligar*,*foto* e *reproduzir*. O segundo estado é o *Inicialização da câmera*, nele é executado a função *init_camera* que serve para ligar a câmera.

Assim que o usuário aperta o botão *foto* então do estado *Inicialização da câmera* vai para *Tira foto* na qual executa a função *take_foto*. Após esse estado vai para *Conversão para texto* e depois para *Escolha de opções*. Nesse último, o usuário decide se desliga o sistema, reproduz o texto ou tira outra foto. Caso queira que desliga, o botão *ligar/desligar* deve ser apertado, mas queira que seja reproduzido o texto o botão *reproduzir* deve ser apertado. Se o botão *foto* for apertado então o próximo estado é o *Tira foto*.

O estado *Reprodução* reproduz o texto que está salvo no arquivo *saida.txt* que foi criado no estado *Conversão para texto*. Ao fim da reprodução, retorna-se para o estado *Escolha de opções*.

No estado *Finalização*, é desligado a câmera. Caso o usuário queira ligar novamente, deve apertar o botão *ligar/desligar* e então o próximo estado seria o *Inicialização da câmera*.

IV. Resultados

Notou-se que depois que a Respberry é reiniciada, é necessário executá-lo três vezes antes da execução funcione. O erro que aparece é que não é possível acessar o arquivo *direction* dos pinos GPIOs. Em cada execução dá erro em um botão. Na quarta execução o sistema o sistema funciona com esperado.

Após colocar a câmera na estrutura, o arquivo texto começou a apresentar mais falhas, mas sem comprometer a compreensão. Isso foi ocasionado pela altura entre o local da folha de papel e onde a câmera ficou. Quando testou novamente fora da estrutura e um pouco mais perto do papel, os erros diminuiu drasticamente.

O programa *tesseract* tem um tempo de execução considerável, cerca de 3 minutos, porém ao reduzir a imagem de 1.9 MP para 0.7 MP, o tempo de execução aumentou, o que não era esperado.

V. Conclusão

É necessário fazer mais algumas alterações na estrutura para ficar adequado a pessoas com eficiências visuais, como posicionar os botões de forma adequada para uma melhor percepção do usuário e ajustar o local onde fica a câmera para que assim fique mais estável e evitando erros.

Em suma, o sistema funcionou como esperado exceto as ressalvas já citadas. Porém é necessário fazer adequações para melhorar a utilização do sistema de leitura de documentos e assim pessoas com deficiências visuais possam ter proveito do sistema

Referências

[1] Luiza Maria Borges Oliveira, *Cartilha do Censo de 2010 - Pessoas com eficiência*, 1a ed. Brasília : SDH-PR/SNPD, 2012.

[2] A. I. Arrahmah, A. Rahmatika, S. Harisa, H. Zakaria e R. Mengko, *Text-to-Speech device for patients with low vision*, 2015 4th International Conference on Instrumentation, Communications, Information Technology, and Biomedical Engineering (ICICI-BME), Bandung, 2015, pp. 214-219.

[3] Ray Smith. *An Overview of the Tesseract OCR Engine*, Ninth International Conference on Document Analysis and Recognition (ICDAR 2007), Curitiba, Brasil, 2007.

[4] Raspbian. *Welcome to Raspbian*. Disponível em: https://raspbian.org/FrontPage. Acesso em 02 de maio de 2018.

[5] NixCraft. *How to: Recompiling / Rebuild Debian / Ubuntu Linux Binary Source File Packages*. Disponível em: https://www.cyberciti.biz/faq/rebuilding-ubuntu-debian-linux-binary-package/. Acesso em 02 de maio de 2018.

[6] Raphaël Hertzog. *Howto to rebuild Debian packages*. Disponível em: https://raphaelhertzog.com/2010/12/15/howto-to-rebuild-debian-packages/. Acesso em 02 de maio de 2018.

[7] CHDK. *Canon Hack Development Kit - Instalation Guide*. 2010.

[8] CHDKPTP. *About chdkptp*. Disponível em: https://app.assembla.com/spaces/chdkptp/wiki. Acesso em 05 de junho de 2018.

[9] DUERIG, Johnatann. *DIY Book Scanner Introduction*. Disponívem em: https://diybookscanner.org/en/intro.html. Acesso em 19 de junho de 2018.