Ponto de Controle 4

Leitor de Códigos de Barras para Carrinhos de Compras

Erick Antonio Correa dos Reis - 150034156
Universidade de Brasília
St. Leste Projeção A - Gama - DF, 72444-240
email: erickcorreareis@gmail.com

Tiago Avelino Ribeiro da Silva - 150022662
Universidade de Brasília
St. Leste Projeção A - Gama - DF, 72444-240
email: tiago.avelino1997@gmail.com

***Resumo* - Leitor de códigos de barras para carrinhos de compras utilizando câmera.**

***Palavras-chave* - Códigos de barras; Raspberry Pi; Processamento de imagens.**

I. INTRODUÇÃO

Algumas atividades comuns do dia-a-dia costumam demandar muito tempo, um exemplo clássico são as filas de supermercados, mesmo depois de um longo tempo escolhendo seus produtos ainda é necessário enfrentar demoradas filas para que sejam conferidos todos seus produtos.

Visando reduzir o tempo gasto nessa atividade, foi pensado em um dispositivo integrado ao carro de compras que permite analisar o preço do produto a ser comprado, ao mesmo tempo que permite acrescentar o preço do produto, aos demais do carro. Será feita a leitura do código de barras por meio de uma câmera, lendo os símbolos deste código, pela variação na largura das barras e assim interpretando o produto a ser comprado, e o valor deste.

II. OBJETIVOS

Implementar um sistema que mostre o valor total e atualizado dos itens que estão no carrinho através de um leitor de códigos de barras por câmera.

III. JUSTIFICATIVA

Estabelecimentos que não utilizam dessa técnica já são bastante comuns na Europa por exemplo, onde os próprios clientes passam seus produtos no caixa e realizam o pagamento, ainda assim é necessário certo trabalho para registrar todos os itens que já estavam no carrinho de compras.

Buscando resolver este problema, o projeto em questão visa implementar um sistema que mostra o valor total da compra em um display no próprio carrinho, esse valor também deve ser atualizado assim que um item é adicionado ou retirado do carrinho, fazendo com o pagamento ao final da compra seja muito mais rápido.

A utilização de uma câmera para a identificação do código do produto é algo que deve facilitar bastante a utilização do sistema já que o processamento de imagens em uma placa como a Raspberry Pi 3 é bastante rápido.

IV. REQUISITOS

Como requisitos do sistemas, visa-se obter:

- Decodificar com eficiência os códigos de barras.
- Registrar imagens com boa nitidez para facilitar a decodificação.
- Registrar os itens e atualizar os valor no display em um tempo limite: 2 segundos.
- Fazer a soma dos valores das compras em tempo real.

V. DESCRIÇÃO DE HARDWARE

O projeto tem como foco o processamento de imagens, dessa forma não há uma ênfase em hardware, tendo em vista que boa parte das análises são realizadas para o entendimento e o desenvolvimento em software. Dessa forma o hardware do sistema se concentra em um display de LCD Nokia 5110, que visa fazer a interface entre usuário e hardware, aliada há um conjunto de botões que realiza todas as funcionalidades do sistema.

A.Raspberry Pi 3 Modelo B

Sendo o componente principal,é o componente em que se encontra o processador, sendo este o sistema embarcado, que realizará o processamento de imagens dos códigos a serem traduzidos e a tomada de decisão no projeto.

B.Câmera para Raspberry Pi

Componente que captura várias fotos do produto contendo o códigos de barra, e que após a decisão da Raspberry Pi, analisa os frames em busca do código de barras não possui um

sistema de foco, porém apresenta uma resolução de 8MP, suficiente para a aplicação.

C.Display Nokia 5110.

Será utilizado para realizar a interface entre o usuário e o sistema embarcados, possibilitando visualizar os produtos comprados o valor da compra, e as telas de navegação para realizar as operações necessárias

D.Push Buttons.

São utilizados push buttons para navegar nas telas do display e para adicionar e remover produtos comprados, caso seja necessário.

E. Interface para o usuário.

Em hardware será feita a interface para o usuário onde este poderá colocar o produto a ser comprado no carrinho, ou caso tenha colocado um produto erroneamente retirar tal produto e ainda visualizar sua lista de compras já realizada. Essa atividade só será possível por meio do teclado matricial, que apresentará botões de avançar nas telas onde está sendo realizada a compra e caso necessário retirar o produto selecionado no display.

Foi realizada a codificação da comunicação SPI com o display LCD Nokia 5110 através da biblioteca ArmbianIO. Foram criadas algumas telas para facilitar a visualização de informações, sendo uma dessas as telas de inicialização mostrando a data e hora do momento da compra, ainda nessa tela foi realizada a chamada do "botão começar" para entrar na próxima tela.

Na tela seguinte é colocado o último produto a ser comprado sendo que nessa tela há uma mensagem que requisita o botão "adicionar", para direcionar a tela inicialização da câmera, assim a câmera só é ativada no momento de leitura do código de barras, visando poupar energia do sistema. Após a leitura do produto há o pedido da quantidade de produtos do mesmo tipo sendo comprados, para economizar tempo.

Quando é acionado o botão para realizar a compra é feita uma nova mudança de tela, e a câmera é ativada, lendo o código de barras do produto, caso em um período de tempo pré definido não seja possível realizar a leitura, o sistema alerta ao usuário com uma mensagem nesta tela que o botão seja pressionado novamente para realizar a leitura ou para voltar para tela onde o último item comprado está presente.

Caso um dos itens seja adicionado erroneamente há um botão que redireciona o usuário a lista completa de itens comprados, e nessa nova tela há um botão de selecionar o produto, subindo ou descendo na lista, com o produto selecionado bastará apertar o botão de retirar o produto.

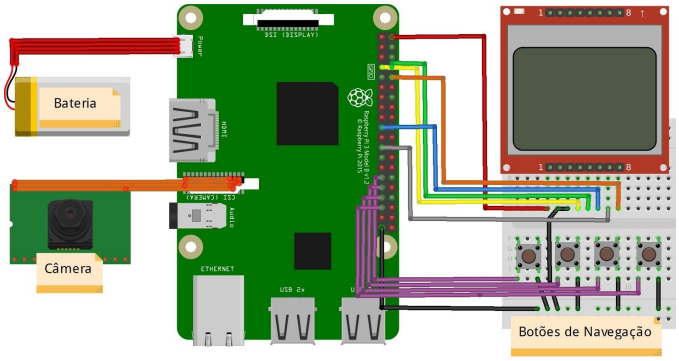

Figura 1. Esquemático do hardware

Tabela 1: Materiais utilizados.

Quantidade	Materiais
1	Raspberry Pi 3 Modelo B
1	Display Nokia 5110
4	Push Button sem trava
1	Câmera 8.0 MP
1	Bateria
1	Placa perfurada

VI. DESCRIÇÃO DE SOFTWARE

Para implementar o projeto é utilizada a biblioteca OpenCv[3] - *Open Source Computer Vision Library*, essa biblioteca tem como propósito auxiliar na construção de projetos na área de visão computacional, possuindo módulos para o tratamento de fotos e vídeos, utilizando a linguagem Python. Foi implementado inicialmente o projeto em linux para verificar a funcionalidade desta em seguida os blocos de códigos da biblioteca a serem utilizados serão implementados na Raspberry Pi. Serão explicados os blocos de códigos utilizados.

A. Threads

A utilização de threads é uma maneira de otimizar a execução do código, sendo elas responsável por dividir o códigos e partes que são executadas quase que de forma paralela. O programa não trabalha totalmente em paralelo pois que define a ordem e prioridade dos processo é o próprio sistema operacional.

Neste projeto a utilização das threads se tornou necessário quanto optamos por utilizar múltiplos botões além da tela LCD. Como a atualização de telas é um processo relativamente lento, criamos uma thread para cada botão de

forma a esperar independentemente o pressionamento de cada um deles.

B. Identificador de qrcode e barcode.

Para resolver esse requisito foi utilizada uma biblioteca, a *pyzbar*, que por sua vez trabalha em conjunto com o OpenCV. Esta biblioteca funciona em *python* e consegue obter o código seja ele de um *qrcode* ou *barcode* e armazenar em uma struct que contém todas as informações a respeito da imagem analisada.

C. Integração C e python.

O desenvolvimento do código seguiu-se em duas frentes, uma delas responsável por controlar todo o display e a outra para tirar as fotos e decodificar o *qrcode* ou *barcode*, entretanto a primeira parte foi feita em *C* enquanto para a segundo parte a linguagem utilizada foi o *Python*.

Essa integração foi feita utilizando a função *system()* em *C*, esta função executa um linha de comando no terminal no dispositivo. Com isso foi possível executar o código em *python*, que por sua vez armazena o resultado em arquivo que fica disponível posteriormente para o restante do código em *C*.

D. Cadastro dos produtos.

Através do código anterior, será feito a união do codigo de barra relacionando este com o valor de mercado do produto, caso não haja esse produto será separada uma área de cadastro, prévio, realizado inicialmente pelos integrantes do projeto ou pelo usuário que tiver acesso a essa área (gerente ou funcionário da loja).

E. Valor efetivo.

Será realizada a soma de todos os valores adicionados ao carrinho e esta soma será mostrada no display LCD, assim como será gerada toda a lista dos produtos correspondentes à compra realizada.

VII. RESULTADOS

Para o ponto de controle atual foram obtidos os seguintes resultados:

- Um código em *python* que tira fotos e decodifica o *qrcode* ou *barcode* presentes nelas fornecendo o resultado final em um arquivo.
- Um código em *C* que controla todas as funcionalidades do display e além disso executa o código em *python* e verifica os dados deixados no arquivo.

Para a fase final do projeto espera-se o protótipo totalmente funcional e em um local adequado que facilite sua utilização.

VIII. CONCLUSÃO

Depois das inúmeras dificuldades em utilizar o OpenCV e a câmera a dupla conseguiu integrar as funcionalidades da linguagem *C* e do *Python* para obter um código eficiente e capaz de executar a proposta do projeto.

A decodificação dos códigos é bastante precisa entretanto notou-se que a leitura destes é feita com maior velocidade quando o código é posicionado centralizado com a câmera.

REFERÊNCIAS

[1] Contagem de objetos em movimento com OpenCV e Python usando Raspberry Pi. Dispnível em: <https://www.embarcados.com.br/objetos-opencv-e-python-raspberry-pi/>. Acesso em 04 abr. 2018.

[2] SIMÕES, Eduardo Dusanoski. DESENVOLVIMENTO DE SISTEMA PARA LEITURA DE CÓDIGO DE BARRAS COM ”FEEDBACK” PARA AQUISIÇÃO E SEGURANÇA DE PRODUTOS EM SUPERMERCADOS. 2015. 53 f. Disponível em: <http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/6789/1/CT_COEAU_2015_1_10.pdf>. Acesso em: 04 abr. 2018

[3]OpenCv library. Acesso em 27 abr.2018 <http://projectabstracts.com/list-of-projects-on-image-processing>.

[4] ArbianIO library. Acesso em 26 mai.2018 ás 9:00 horas <https://github.com/bitbank2/ArmbianIO>.