CSLP – 2021/22

COMPUTER VISION:

CARACTERIZAÇÃO DE LEGOS

GRUPO 1

98134 – Tiago Matos

98396 – Vitor Dias

universidade de aveiro

ÍNDICE

INTRODUÇÃO

Este projeto foi realizado no âmbito da unidade curricular de CSLP (Complementos Sobre Linguagens de Programação). Com uma temática envolvendo *Computer Vision*, o projeto consistia em desenvolver com *OpenCV* e *C++* um programa capaz de identificar peças de Lego e a respetiva cor. O programa resultante seria então executado num *Raspberry Pi* com uma câmara e testado em tempo real.

Objetivos

Definimos os seguintes objetivos:

- Detetar a cor da peça de Lego em diferentes condições de luz natural
- Detetar o maior número possível de cores da *Color Palette* oficial
- Determinar o tamanho e formato da peça (e.g.: uma peça 2x2) contando os pinos da peça

Requisitos

Para atingir estes objetivos, estipulámos também os seguintes requisitos:

- A cor tem de ser convertida do espaço de cor RGB para HSV
- A distância da peça à câmara deve ser fixa
- A câmara deve captar a peça de Lego vista por cima, apanhando os pinos da mesma
- As peças devem ser de formatos retangulares
- A peça deve ser fotografada numa superfície branca

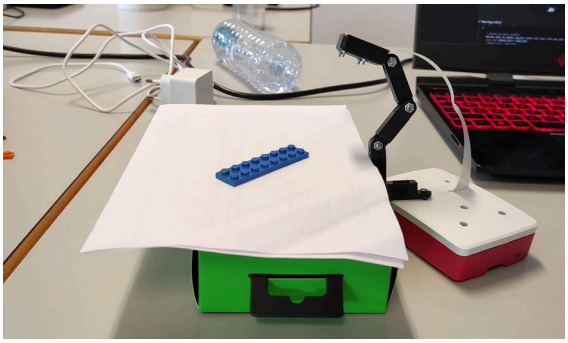

Figura 1 – Setup usado para captura da peça

IMPLEMENTAÇÃO

Análise da cor

Tendo em conta que um dos nossos objetivos era detetar o máximo de cores possíveis no espaço HSV, começámos por recolher os valores de cores da *Color List.* A partir dessa lista, numa folha de Excel aplicámos as seguintes transformações:

1. Filtrar as cores sólidas, excluindo cores transparentes, metálicas, entre outras.
2. Filtrar as cores que apresentavam valores CMYK da cor e um nome de referência.
3. Conversão dos valores CMYK para HSV, para a qual utilizámos duas funções: uma primeira para converter os valores CMYK em RGB e outra para converter os valores RGB em HSV.
4. Ordenar os resultados por H, S e V.

Deste processo conseguimos uma lista de 93 cores HSV com um nome descritivo.

Pré-processamento da imagem

Para cumprir o objetivo de detetar a cor do Lego em diversas condições de luz natural, precisámos de aplicar algumas transformações nas imagens capturadas. O resultado desejado era uma imagem que se aproximasse o máximo possível da mesma fotografia numa situação de iluminação ideal.

Tendo em conta que as nossas imagens têm um fundo branco, começámos por fazer um *threshold* dos pixéis mais claros, utilizando o método de Otsu. Para cada um dos canais RGB, calculámos a média dos valores para esses pixéis. De seguida, considerando os valores de média, determinamos as transformações lineares necessárias para levar esses valores a 255. Aplicamos estas transformações a toda a imagem RGB.

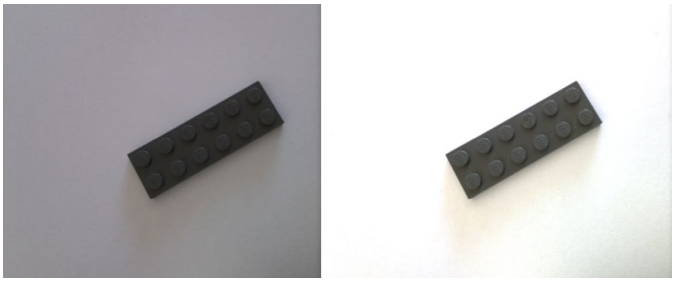

Figura 2 *– Fotografia de uma peça cinzenta antes (imagem da esquerda) e depois (imagem da direita) da transformação linear.*

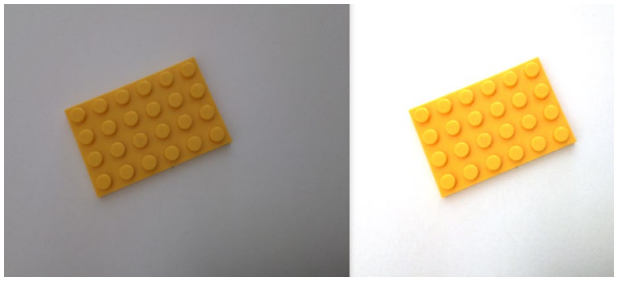

Figura 3 *– Fotografia de uma peça amarela antes (imagem da esquerda) e depois (imagem da direita) da transformação linear.*

Deteção dos pinos

Para detetar os pinos, é usada a função *HoughCircles()* do OpenCV aplicada a uma imagem onde foi aplicado um *Threshold* e *Histogram Equalization*. Os parâmetros são limitadores em relação à distância da peça e da luminosidade.

Cálculo das Dimensões

O cálculo das dimensões começa por encontrar os pontos com X menor, Y maior e Y menor, sendo estes os extremos considerados. De seguida são formadas duas retas imaginárias, de X menor a Y maior e de X menor a Y menor. Depois são percorridos todos os círculos encontrados. Assim, os círculos que contiverem o valor da reta no ponto cujo seu X é igual ao X do centro do círculo em questão pertencerão à lateral da peça. Dois contadores vão sendo incrementados durante o processo dando no final as dimensões da peça.

Cálculo da zona provável da peça

Com o intuito de analisar a cor apenas na zona da peça evitando possíveis zonas com cor, sombras por exemplo, selecionamos apenas a zona da imagem onde foram detetados mais círculos, que em princípio será o Lego.

Deteção da cor

A partir da imagem resultante do pré-processamento, aplicamos mais algumas funções de filtro como, por exemplo, a *pyrMeanShiftFiltering()* que ajuda a remover as texturas do Lego. Para encontrar a cor efetiva, percorremos a lista de cores e aplicamos um *inRange()* para cada uma delas e vamos guardar a cor que abrange o maior número de pixéis, que é a nossa solução.

LIMITAÇÕES

Deteção da cor

Tendo em conta que considerámos uma lista tão extensa de cores, cujos valores HSV nem sempre são tão divergentes, tornou-se difícil caracterizar a cor das peças de forma precisa. As transformações aplicadas no pré-processamento das imagens ajudaram a tornar os resultados mais próximos da realidade, mas ainda assim encontrámos problemas a distinguir cores de gamas semelhantes. Por exemplo, ao executar a imagem da *Figura 2* o resultado é a cor *Dark Curry* em vez da cor *Bright Yellow*.

Cálculo das dimensões

O método para o cálculo das dimensões utilizado não é funcional para muitas condições de luminosidade, sendo que esta afeta a deteção dos pinos e, portanto, o respetivo cálculo da dimensão.

Outro aspeto importante é a necessidade de a peça se localizar de forma oblíqua (conforme observado nas figuras acima), sendo que, caso a peça esteja horizontalmente ou apenas ligeiramente inclinada, não é certa a deteção verídica das dimensões.

Finalmente, o programa atual está destinado a distâncias peça-câmara semelhantes ao observado na Figura 1.

Tempo de execução

Devido à elevada complexidade computacional do programa, este leva algum tempo a ser executado, com aplicações de filtros, etc. O tempo médio é 11.8514 segundos.

Assim, optou-se por manter o programa a captar uma fotografia ao ser executado e terminar, ao invés de ser contínuo e resposta em vídeo.

UTILIZAÇÃO

Instalação de Software

De forma a conseguir executar o nosso programa, é necessário primeiramente instalar-se a livraria OpenCV no Raspberry Pi. O nosso foi instalado seguindo o seguinte tutorial:

- https://solarianprogrammer.com/2019/09/17/install-opencv-raspberry-pi-raspbian-cpp-python-development/

Compilação do Código

Para compilar o nosso código ou outro desejado com o método de instalação que utilizámos:

```
$ g++ main.cpp -o readLego `pkg-config --cflags --libs opencv
```

- main.cpp : ficheiro com código a compilar
- readLego : nome para o ficheiro a executar para correr o programa

Execução do Código

Aqui já deve ter tudo preparado para executar o programa, incluindo câmara do Raspberry Pi pronta e setup preparado. Para executar o programa:

```
$ ./readLego
```

- readLego : nome dado no processo de compilação

Figura 4 - Resultado da execução do programa