Relatório do trabalho da disciplina de Introdução à Visão por Computador

Fase 1 – Algoritmos de Segmentação

Henrique Azevedo - 23488

José Lourenço - 23496

Engenharia em Desenvolvimento de Jogos Digitais

Novembro de 2022

ESCOLA SUPERIOR DE TECNOLOGIA IPCA

Índice

Introdução

Contextualização

Este projeto diz respeito à proposta de trabalho da unidade curricular de Introdução à Visão por Computador (IVC), no âmbito do 2º ano da Licenciatura em Engenharia em Desenvolvimento de Jogos Digitais, no Instituto Politécnico do Cávado e Ave, realizado durante o período letivo de 2022/2023.

A partir da utilização do código jogo *Breakout*, previamente dado pelo docente, objetiva-se a criar um algoritmo de segmentação, que com o uso da câmara seja possível detetar um objeto e com o mesmo, realizar as ações necessárias para o jogo.

Nesta primeira fase do projeto foi proposto a colocação do *paddle* do jogo *Breakout* a mover, utilizando assim, alguns métodos de algoritmia de segmentação que foram lecionadas ao longo do decorrer das aulas.

Concluindo, este relatório vem a fazer explicação profunda de todos os métodos adotados que contribuíram para criação da proposta elaborada para a fase 1, com conceitos teóricos, problemas e resultados.

Desenvolvimento

Para a boa realização desta fase do projeto, foi necessária a revisão de vários conceitos teóricos sobre processamento de imagem que foram previamente lecionados e assim criar um melhor entendimento e encontrar a melhor forma de resolver o projeto proposto.

Conceitos Teóricos

Espaço de Cor HSV

O espaço de cor HSV proporciona um método intuitivo de especificar a cor. Neste espaço de cor, cada cor é representada por três componentes:

- Tonalidade ou Matiz (*Hue*);
- Saturação (*Saturation*);
- Valor (*Value*).

No espaço HSV é possível selecionar a tonalidade desejada, e posteriormente realizar ajustes na saturação e intensidade.

Esta separação entre a componente da luminância (luz) e da crominância (cor) traz vantagens relativamente ao espaço RGB quando se pretende realizar operações sobre cores.

Filtragem

Aparte da deteção do objeto que se pretende utilizar, através da câmara, é habitual que também se detete ruído na imagem, que pode fazer com que o resultado que queremos alcançar não seja o mais limpo.

A filtragem é utilizada para remover o ruído de uma imagem ou para salientar algum atributo em específico. Podendo se utilizar filtros passa-baixo e passa-alto, conforme o objetivo final.

Para a remoção de tal ruído, utilizou-se um filtro, passa-baixo, que procede á remoção de componentes de alta frequência da imagem, assim obtendo uma imagem mais suave e um pouco desfocada.

Quanto maior for a máscara aplicada melhor será o efeito de suavização.

No projeto foi utilizado um filtro gaussiano, aplicando assim a função *"cv.GaussianBlur"*, que multiplica cada pixel da imagem pelo valor da máscara

Para se obter a máscara do filtro gaussiano usa-se a seguinte formula:

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Figura 1 - Fórmula *Kernel Gaussiano*

O *x* e *y* representam o índice da localização, o *x* a distância do pixel central no eixo horizontal e o *y* a distância no eixo vertical.

O σ representa o desvio padrão, controlando a dimensão do efeito de suavização em torno de um pixel.

Segmentação

A segmentação de imagem tem como objetivo isolar regiões de pixéis de uma imagem, que pertencem a determinados tipos de objetos, para posterior extração de atributos e cálculo de parâmetros descritivos.

Do processo de segmentação resultam assim dois tipos de pixéis:

- De primeiro plano;
- De plano de fundo.

Assim, como resultado de um processo de segmentação, podemos obter uma imagem binária.

As regiões formandas por pixéis de primeiro plano definem assim os objetos sobre os quais se pretende obter dados ou atributos relevantes:

- Número de objetos;
- Dimensões dos objetos;
- Luminosidade dos objetos;
- Etc.

A seguinte figura representa o processo de segmentação:

$$\texttt{dst}(x,y) = \begin{cases} \texttt{maxval} & \text{if } \texttt{src}(x,y) > \texttt{thresh} \\ 0 & \text{otherwise} \end{cases}$$

Figura 2 - Representação do processo de segmentação por *threshold*

Depois da utilização deste método obteve-se esta imagem:

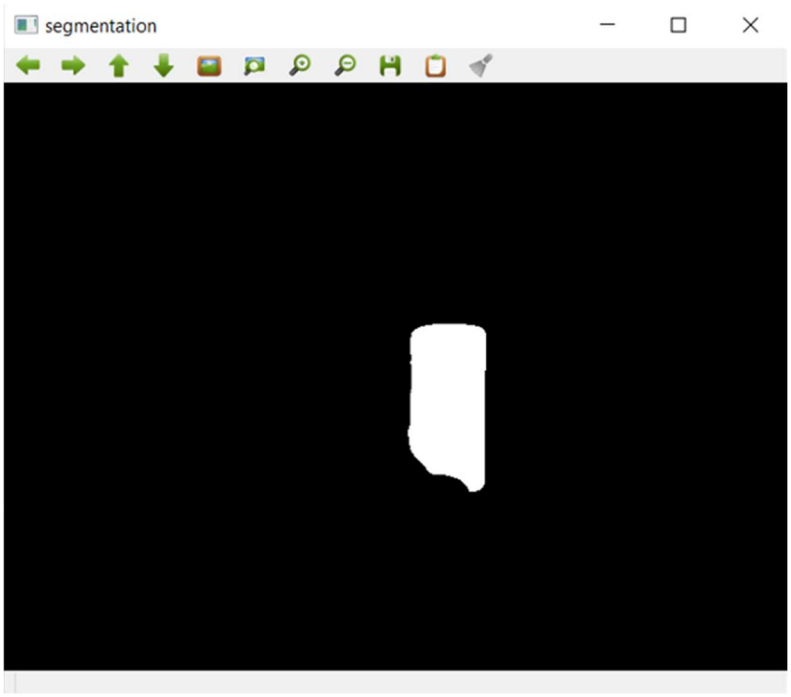

Figura 3 - Imagem segmentada do objeto

Contornos

Contornos são utlizados para reconhecimento e deteção de objetos, mostrando as áreas com maior variação de intensidade.

O *OpenCV* dá a possibilidade de utilizar duas funções para realizar este processo, sendo elas *"cv.findContours"* e *"cv.drawContours".*

Com a primeira função é possível detetar os contornos na imagem e com a seguinte, desenhar o contorno, tendo de saber os pontos de fronteira.

Descrição

Ao início foi criada uma classe, de nome *"camInput"*, contida no construtor do jogo, que permitirá captar vídeo, através da camara, assim captando o objetivo escolhido pelo grupo, tendo sido um pedaço de cartão amarelo.

Na mesma, foram implementados métodos de segmentação de imagem para, assim, ao usar a variável *"cam"* que será assignada à classe *"cv.VideoCapture()"*, para obter a imagem da câmara, seguida de um método chamado *"showcam"* que tem como objetivo mostrar a imagem captada num janela, com o uso da função *"cam.open(0)"*.

De seguida temos o método "*object_detection*" que tem como propósito final a deteção do objeto. Assim, este deteta em primeiro, a *frame* da imagem e procede a inverte-la, para esta ficar a par com a visão do jogador. Neste método faz-se ainda, a deteção da posição do objeto.

Após a obtenção do *frame*, o mesmo será transformado numa uma imagem *blur*, através da função de *Kernel Gaussiano* ("*cv.GaussianBlur*"), para tornar a imagem mais suave reduzindo o ruido, tendo sido aplicado uma convulsão de 25x25. Depois da transformação em imagem *blur*, é necessário transformar a imagem para HSV, que nos dará a possibilidade de definir um *threshold* para a deteção do objeto.

Tendo sido a cor escolhida o amarelo, foi necessária a criação de um *threshold* com intervalo de valores que correspondam a esta cor, tendo sido o intervalo [20, 30] para a variação do *Hue* e [100, 255] para a variação de saturação, para não detetar pixéis demasiado cinzentos.

Após a deteção do objeto, usando o método *"positionside"* dá se a indicação do movimento para o *paddle*, analisando a posição do objeto, assim, caso o objeto esteja na parte direita da imagem o *paddle* move-se para a direita e vice-versa, havendo também uma posição neutra no centro da imagem, que faz com que o *paddle* não se mova. Caso nenhum objeto seja detetado, o programa interpreta que o objeto está no meio, ou seja, o *paddle* não move.

Para o melhor uso do método anterior foi elaborado outro método de nome *"contourcenter"* que dá a posição central do objeto para uso em outros métodos. Outro método criado foi o *"countoridx"*, que serve para encontrar a maior área de um contorno, assim se houver mais do que um objeto detetado aquele que apresentar uma maior área será o usado para o jogo.

No final, foi elaborado o método *"destroywindow"* que tem como propósito fechar a janela da imagem quando o jogo não está a decorrer.

Problemas de Codificação

O maior problema foi com a correção do ruído, tendo em conta que ele atrapalhava muito na suavidade da leitura e deteção de objetos. Por isso tivemos de usar um filtro gaussiano, filtro passa-baixo, com convulsão de 25x25 e assim reduzir o ruído e melhorar a deteção dos objetos maiores.

Um dos problemas encontrados foi a deteção correta das cores, onde foi necessário a realização de vários testes para encontrar o valor correto para obtermos a cor desejada

Outro pequeno problema foi a correta deteção do centro da imagem, onde os nossos valores não nos colocaram exatamente o centro de deteção no real centro da imagem.

Com a pesquisa e a retirada de dúvidas com o docente fomos capazes de superar estes problemas e elaborar o trabalho como esperado.

Resultados

No final da implementação de todo o algoritmo no código, conseguiu-se obter o resultado pedido no enunciado da fase 1.

Ao iniciar o programa, surge uma janela que apresenta o jogo *Breakout*, como ser pode ver imagem seguinte

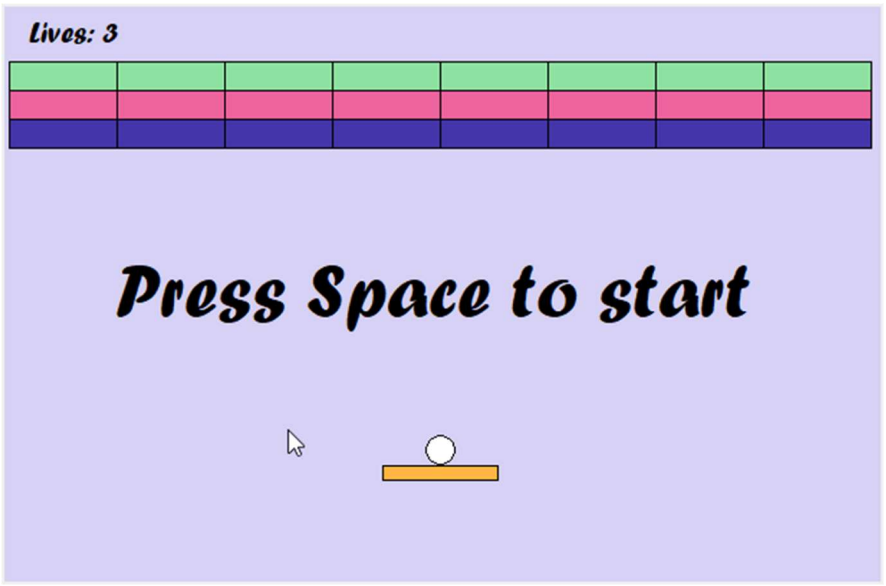

Figura 4 - Jogo *Breakout*

Assim para iniciar o jogo é necessário carregar a tecla espaço, o que fará com que abra uma janela com a camara se abra.

Nesta janela é possível fazer o reconhecimento do objeto de cor amarela, que será utilizado para mover o *paddle* no jogo. Assim, quando o objeto é detetado na parte direita da imagem, o *paddle* irá também para a direita e vice-versa, havendo também uma posição neutra no centro da imagem. Caso nenhum objeto de cor amarela seja detetado o *paddle* não se move.

Nas seguintes imagens é possível interpretar a deteção do objeto e deslocação do *paddle*:

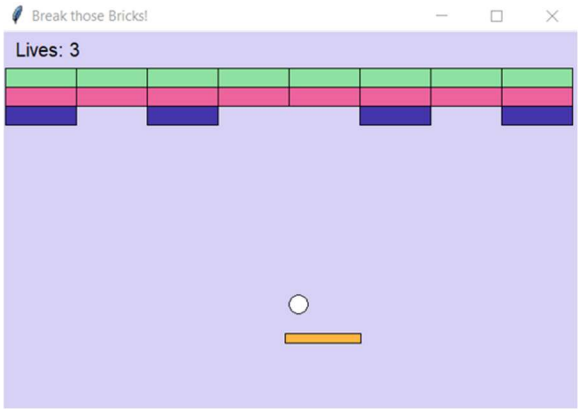

Figura 5 - Partida de Breakout

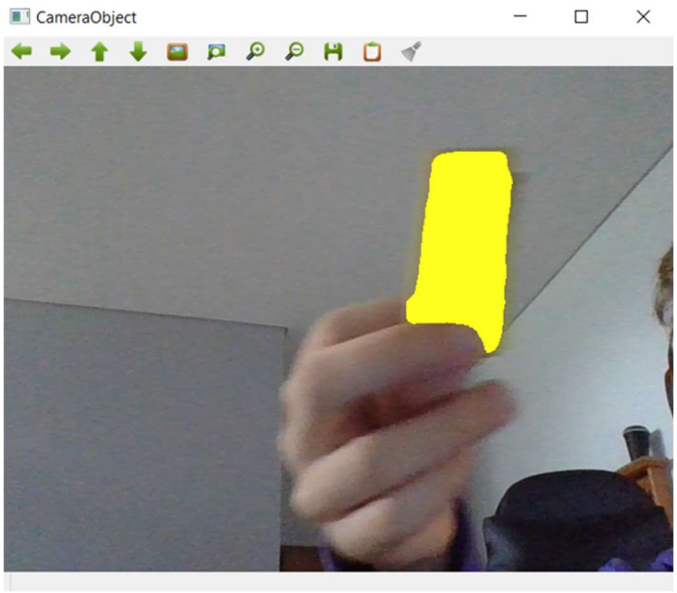

Figura 6 - Deteção do objeto

Conclusão

Com o final desta fase entende-se que foi possível a aprendizagem de vários conhecimentos sobre algoritmos de segmentação e sua respetiva implementação, tendo havido a grande contribuição dos exercícios realizados em aula pelo docente.

Sendo uma área nova foi necessário um maior empenho no estudo de informação para a melhor resolução do projeto, tal como discussões entre o grupo para decidir o melhor meio de trabalho, tendo escolhido o método que mais nos deu conforto na sua realização.

Ao longo desta fase surgiram diversas situações com dificuldades e até completos desentendimentos o que nos ajudou a ter uma evolução crítica e prática de como resolver futuros problemas.

A continuação do trabalho irá prosseguir para obtermos mais conhecimentos e prática nesta área, assim ajudando numa melhor realização das próximas fases.

Bibliografia

https://stackoverflow.com/questions/9179189/detect-yellow-color-in-opencv

https://docs.opencv.org/4.x/d4/d73/tutorial_py_contours_begin.html

https://pyimagesearch.com/2016/02/01/opencv-center-of-contour/

https://pyimagesearch.com/2015/09/14/ball-tracking-with-opencv/

https://www.geeksforgeeks.org/python-opencv-find-center-of-contour/

https://cvexplained.wordpress.com/2020/04/28/color-detection-hsv/