

Trabalho de Avaliação

Deteção e Reconhecimento de Baralho de Cartas Clássico










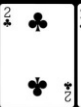
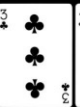














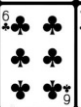
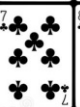
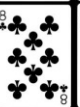













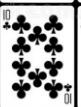






















1 Introdução

O baralho de cartas foi criado na China por volta do século 10, e foi distribuído por todo o mundo. Além de ser usado em vários tipos de jogos de cartas, também tem sido usado para ilusionismo, adivinhação e instrução.


Cada carta de um baralho é um retângulo de papel com um lado estampado com cores e símbolos, geralmente números e naipes, chamado de face. Para esconder o valor da face, o outro lado é estampado com um padrão comum para todas as cartas do jogo.

O baralho inglês de 52 cartas, ao qual podem ser adicionadas uma ou duas cartas chamadas jokers é a quantidade de cartas mais utilizadas em diferentes jogos. É dividido em quatro naipes franceses, cada um contendo cartas de 2 a 10 e as figuras valete, dama e rei.

Para além dos números e figuras, o baralho principal de 52 cartas contém 13 cartas de cada um dos quatro naipes franceses: paus (♣), ouros (♦), copas (♥) e espadas (♠), como se pode observar nas duas figuras seguintes.

															
															
															
															
Copas				Ouros				Paus				Espadas			

1	2	3	4
5	6	7	8
9	10	Valete	Dama
Rei	JOKER		



Objetivos

1.1 Níveis do Trabalho

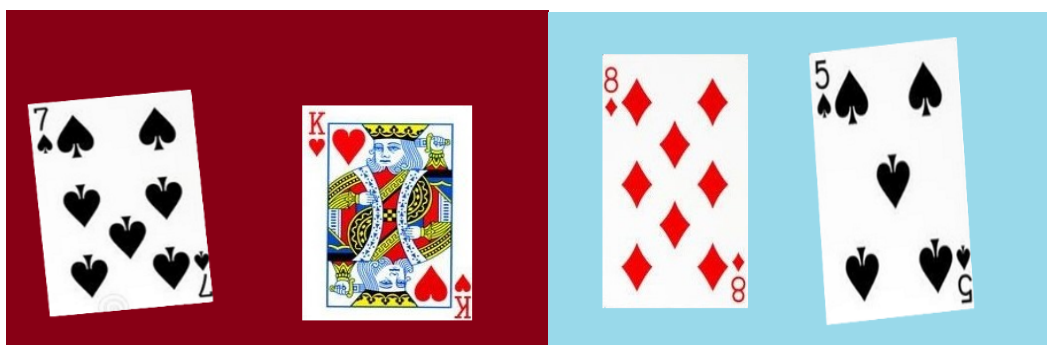
O trabalho consiste na detecção de cartas (até 4 cartas) que estão na mão de um jogador, onde é necessário detetar o tipo de cada carta e o seu correspondente naipe.

Este trabalho é constituído por 3 níveis:

- Nível 1 – Uma carta num fundo uniforme onde o ângulo é sempre igual a 45°.



- Nível 2 – Mais do que 1 carta (de 2 até 4 cartas) com fundo uniforme com ângulo que pode ser diferente de 45°.



- Nível 3 – 1 a 4 cartas num fundo não-uniforme, com ângulos que podem ser diferentes de 45°.



1.2 Requisitos da Aplicação a desenvolver

O programa deverá:

- disponibilizar ao utilizador:
 - uma interface que permita aplicar à imagem sobre a qual se está a trabalhar todas as funções desenvolvidas nas aulas práticas; e
 - uma interface adequada à deteção e reconhecimento das cartas do baralho.
- identificar as cartas que são em cada imagem (o efetivo);
- identificar o número da carta e o motivo (sua localização, dimensão e ângulo);
- tolerar alguma variação de:
 - dimensão entre imagens;
 - iluminação entre imagens;
 - deformidade nas imagens fotográficas (cartas deformadas ou oval - nível 3).

A nota do trabalho terá em conta a implementação de requisitos extra que melhorem o trabalho.

O programa será avaliado com imagens que contêm um até quatro cartas.

2 Avaliação

O trabalho será objeto de uma avaliação semi-automática que incluirá a análise de um conjunto de imagens fornecidas pelos docentes e de um outro conjunto de imagens idênticas, mas que não serão disponibilizadas aos alunos.

Na maioria dos casos não haverá discussão presencial do trabalho. Contudo caso os docentes vejam necessidade irão contactar os alunos para marcar uma discussão presencial, a realizar em dezembro, na semana de Avaliação Contínua.

A avaliação semi-automática vai ter em conta a existência ou não das funções, o seu tempo de execução e os resultados obtidos.

O trabalho (projecto, código fonte, executável e relatório em PDF) deverá ser entregue na plataforma moodle na data fixada para o efeito.

O relatório (obrigatório) deverá descrever os algoritmos utilizados para o reconhecimento das cartas, bem como das suas localizações e apresentar sucintamente todos os métodos utilizados. O relatório deverá ser estruturado de forma que o leitor compreenda como foi pensado e criado o sistema, para que no futuro, caso se pretenda, se possa a vir melhorar ou alterar o projeto realizado.

Data de entrega do trabalho: 14 de dezembro de 2024.

Data de Defesa: 16/17 de dezembro de 2024

Nota: Considera-se uma penalização na nota máxima de 2 valores por cada dia de atraso na sua entrega. Exemplo: 2 dias de atraso = nota máxima possível 16v.

A não aparência do aluno no dia defesa, mesmo submetido o trabalho no moodle, e equivalente a zero na prática.

3 Operações – Métodos *ImageClass*

3.1 Métodos obrigatórios

1	Negative	Negativo de uma imagem
void Negative(Image<Bgr, byte> img) - recebe a imagem a alterar		
2	BrightContrast	Ajuste de brilho e contraste.
void BrightContrast(Image<Bgr, byte> img, int bright, double contrast) - recebe a imagem a alterar, o valor de brilho e de contraste		
3	RedChannel	Copia a componente Red para a Green e Blue.
void RedChannel(Image<Bgr, byte> img) - recebe a imagem a alterar		
4	Translation	Desloca a imagem de (Dx,Dy) =(-10 , -10), preenchendo as partes sem pixeis a preto.
void Translation(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, int dx, int dy) - Recebe a imagem a alterar, uma cópia da imagem e o deslocamento em x e y.		
5	Rotation	Roda a imagem (angulo em radianos) , preenchendo as partes sem pixeis a preto.

void Rotation(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, float angle) - Recebe a imagem a alterar, uma cópia da imagem e o angulo de rotação (radianos).		
6	Scale	Aplica um factor de escala a partir de (x,y) = (0,0), preenchendo as partes sem pixels a preto.
void Scale(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, float scaleFactor) - Recebe a imagem a alterar, uma cópia da imagem e o fator de escala.		
7	Scale_point_xy	Aplica um factor de escala centrado em (x,y), preenchendo as partes sem pixels a preto.
void Scale_point_xy(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, float scaleFactor, int centerX, int centerY) Recebe a imagem a alterar, uma cópia da imagem, o fator de escala e o centro (x, y).		
8	Mean	Filtro de média 3x3 – solução A (analisa 9 pixels)
void Mean(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy) - Recebe a imagem a alterar e uma cópia da imagem		
9	NonUniform	Aplica um filtro não-uniforme à imagem, usando os coeficientes e pesos passados como parametro.
void NonUniform(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, float[,] matrix, float matrixWeight) - Recebe a imagem a alterar, uma cópia da imagem, uma matrix 3x3 contendo os coeficientes e o peso		
10	Sobel	Filtro de Sobel 3x3
void Sobel(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy) - Recebe a imagem a alterar e uma cópia da imagem		
11	Differentiation	Filtro de diferenciação (2x2).
void Differentiation(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy) - Recebe a imagem a alterar e uma cópia da imagem		
12	Median	Filtro de mediana a cores usando a avaliação 3D.
void Median(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy) - Recebe a imagem a alterar e uma cópia da imagem		
13	Histogram_Gray	Calcula o histograma apenas da escala de cinzentos.
int[] Histogram_Gray(Emgu.CV.Image<Bgr, byte> img) - Recebe a imagem a analisar - Devolve o histograma		
16	ConvertToBW	Binarização com valor de threshold recebido como parametro (valor = 157)
void ConvertToBW(Emgu.CV.Image<Bgr, byte> img, int threshold) - recebe a imagem a alterar e o valor de threshold		
17	ConvertToBW_Otsu	Binarização com valor de threshold calculado pelo método de Otsu
void ConvertToBW_Otsu(Emgu.CV.Image<Bgr, byte> img) - recebe a imagem a alterar		

3.2 Métodos Facultativos

Lista de métodos que se forem implementados poderão ser testados no Eval:

14	Histogram_RGB	Calcula o histograma das componentes B, G e R
int[,] Histogram_RGB(Emgu.CV.Image<Bgr, byte> img) - Recebe a imagem a analisar - Devolve o histograma das 3 componentes B + G + R numa matrix [3, 256]		
15	Histogram_All	Calcula o histograma das componentes Gray, B, G e R
int[,] Histogram_All(Emgu.CV.Image<Bgr, byte> img) - Recebe a imagem a analisar - Devolve o histograma das 4 componentes GRAY + B + G + R numa matrix [4, 256]		
18	Mean_solutionB	Filtro de média 3x3 – solução B (analisa 6 pixels)
void Mean_solutionB(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy) - Recebe a imagem a alterar e uma cópia da imagem		
19	Mean_solutionC	Filtro de média 7x7 – solução C
void Mean_solutionC(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, int size) - Recebe a imagem a alterar, uma cópia da imagem e a dimensão do filtro		
20	Roberts	Filtro de Roberts
void Roberts(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy) - Recebe a imagem a alterar e uma cópia da imagem		
51	Rotation_Bilinear	Roda a imagem de 30º (ângulo em radianos) , preenchendo as partes sem pixels a preto. Usa Interpolação Bilinear.
void Rotation_Bilinear(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, float angle) - Recebe a imagem a alterar, uma cópia da imagem e o ângulo de rotação (radianos).		
61	Scale_Bilinear	Aplica um factor de escala de 0.5x centrado em (x,y) = (0,0), preenchendo as partes sem pixels a preto. Usa Interpolação Bilinear.
void Scale_Bilinear(Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, float scaleFactor) - Recebe a imagem a alterar, uma cópia da imagem e o fator de escala.		
71	Scale_point_xy_Bilinear	Aplica um factor de escala de 2.2x centrado em (x,y) = (250,310), preenchendo as partes sem pixels a preto. Usa Interpolação Bilinear.
void Scale_point_xy_Bilinear (Image<Bgr, byte> img, Image<Bgr, byte> imgCopy, float scaleFactor,int centerX, int centerY) Recebe a imagem a alterar, uma cópia da imagem, o fator de escala e o centro (x, y).		

4 Eval - Descrição

4.1 Painel 1 - Description

No *Eval*, no primeiro painel (figura 1) devem colocar os vossos números de aluno e colocar na descrição uma lista de pontos contendo o que gostariam de chamar a atenção dos professores que forem avaliar o vosso projeto.

Aqui deverão referir os requisitos extras/complementares que valorizem o vosso programa, bem como referir e justificar falhas e limitações que o vosso programa possa ter.

	Student1	Student2	Student3
▶	11111	0	0

Indique aqui a lista de funcionalidades que valorizam o vosso projeto:
-

Release: 2017.10.11.1612

Load Save

Figura 1 – Primeiro painel - descrição.