Relatório 2 - Visão Computacional.
Aluno: Eduardo Mafra Pereira.
Orientador: Leonardo Mejia Ricon.

Contextualização:
O objetivo do projeto é desenvolver um algortimo que conte quantas letras de mesma cor são encontradas encontradas na imagem UFSC_homography.jpg. Sendo assim, o algoritmo desenvolvido atende este objetivo.
Seu funcionamento se apresenta da seguinte maneira:
Primeiramente é realizada a homografia da imagem gerando uma nova, na sequência a matriz é transformada para a escala de cinza e finalmente a imagem é transformada em uma imagem preto e branco. Após a transformação a matriz preto e branco é percorrida e ignorando os ruídos e os apagando. Ao encontrar o primeiro pixel de uma letra uma função recursiva é chamada e esta percorre a letra até transformado seus pixels em zero e ao final contabiliza uma letra. Após perrocer toda a matriz o a quantidade de letras é atribuida a uma variável.

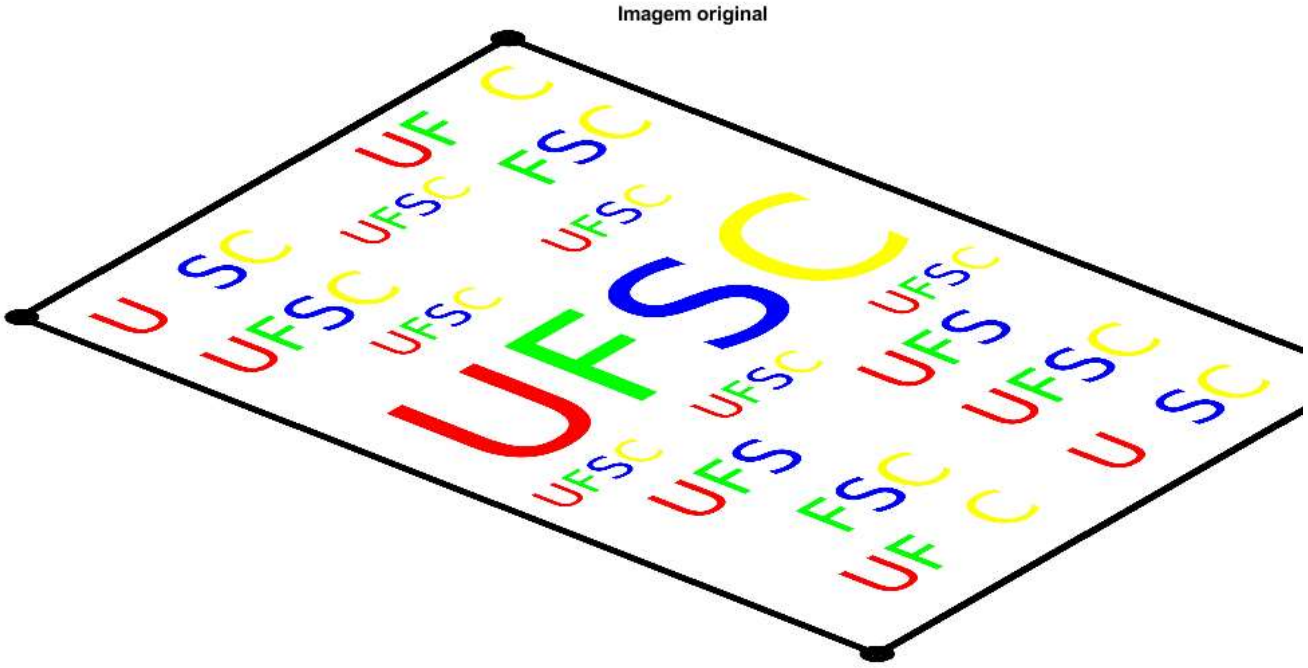
```
clc
Im1= imread('UFSC_Homography.jpg'); % Imagem Original.
figure(1),imshow(Im1);
title('Imagem original');
% Captura 4 pontos da imagem para gerar a matriz de homografia.
[i y] = ginput(4);
x1=fix(i);
y1=fix(y);
x2=[1;500;500;1];
y2=[1;1;500;500];
T=maketform('projective',[x1 y1],[x2 y2]);
T.tdata.T
% Realiza a homografia da imagem.
[Im2,xdata,ydata]=imtransform(Im1,T);
% Apresenta a imagem após a homografia.
figure(2),imshow(Im2);
title('Imagem após a homografia');
% Matrizes RGB
Im2r = Im2(:,:,1);
Im2g = Im2(:,:,2);
Im2b = Im2(:,:,3);
% Media ponderada das matrizes RGB para transformação em escala de cinza
gray = 0.299*Im2r + 0.587*Im2g + 0.144*Im2b;
% Figura em cinza
figure(3), imshow(gray);% Captura a letra a ser contabilizada
title('Imagem em escala de cinza');
[x y] = ginput(1);
x = fix(x);
y = fix(y);
% Define um faixa de valores para escala de cinza
percentual = 5;
% Transformação da escala de cinza para preto e branco
bw = gray <= (gray(y,x)+gray(y,x)*percentual/100) & gray >= (gray(y,x)-gray(y,x)*percentual/100);
global bw;
% Apresenta a imagem em preto e branco
figure(4),imshow(bw);
title('Imagem em preto e branco');
% Atribui o tamanho da matriz para variáveis n e m
[n,m] = size(bw);
flag=0;
count=0;
% Laço para percorrer a imagem
for i = 1:1:n-1
    for j = 1:1:m-1
        % Seguintes condições para verificar os pontos
         if bw(i,j) == 1
            % Verifica se é ou não ruído.
            for e=-1:1:1
                for r=-1:1:1
                    if bw(i+e,j+r)==1
                    flag=flag+1;
                    end
                end
            end
            % Se flag maior que 2 o pixel pertence a letra
            if flag > 2
                % Chama a função recursiva que percorre a letra
                verifica(i,j);
                % Após letra percorrida contador soma 1
                count=count+1;
            end
            % Considera os pixels 4 a 0 para eleminar ruído
            bw(i,j)=0;
        end
    flag=0;
    end
end
Im3= imread('funcao_recursiva.jpg');
figure (6), imshow (Im3);
title('Função Recursiva');
count;
```

```
Warning: Image is too big to fit on screen; displaying at 25%
```

```
ans =

    0.1380    0.1149   -0.0000
   -0.3504    0.1994   -0.0000
  294.8471 -175.6863    1.0000

Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
Warning: Image is too big to fit on screen; displaying at 67%
```



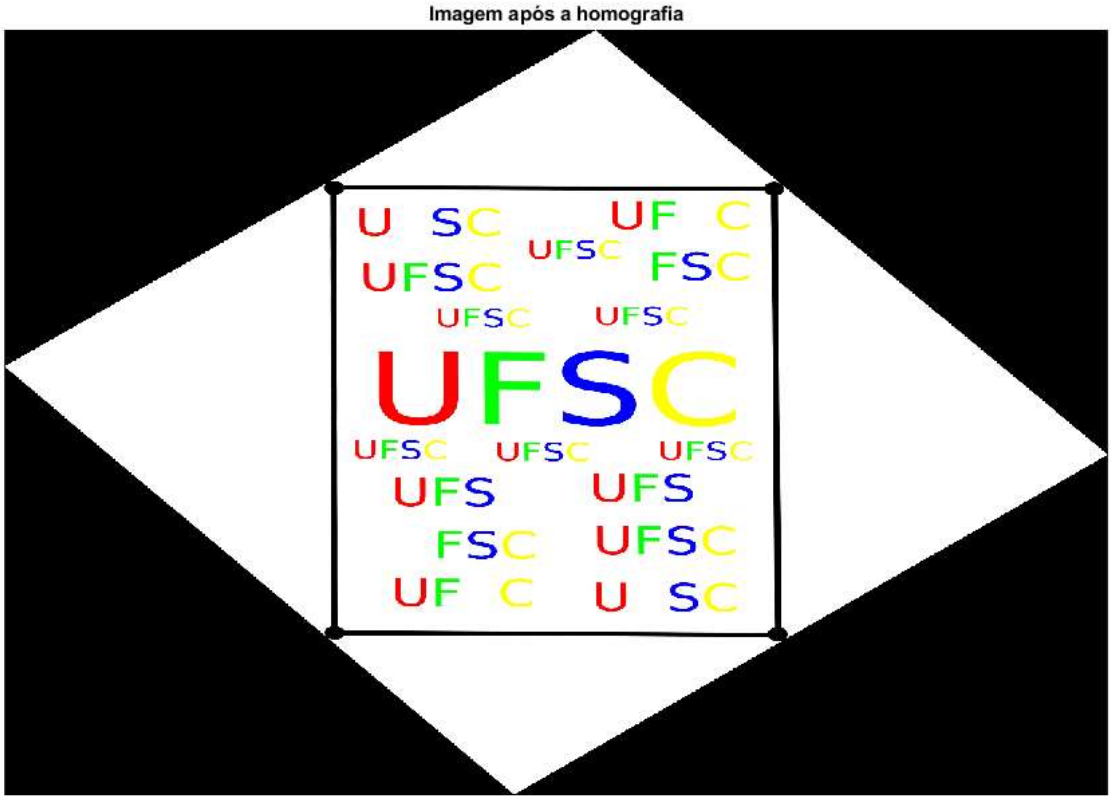


Imagem em escala de cinza

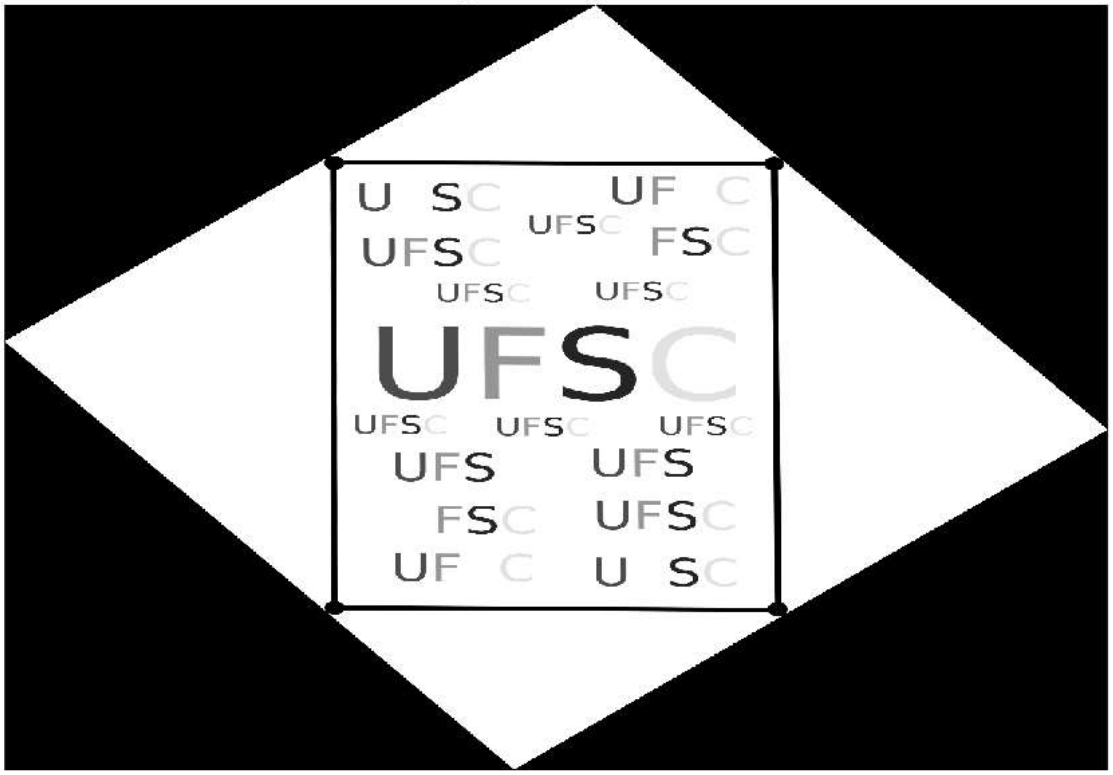
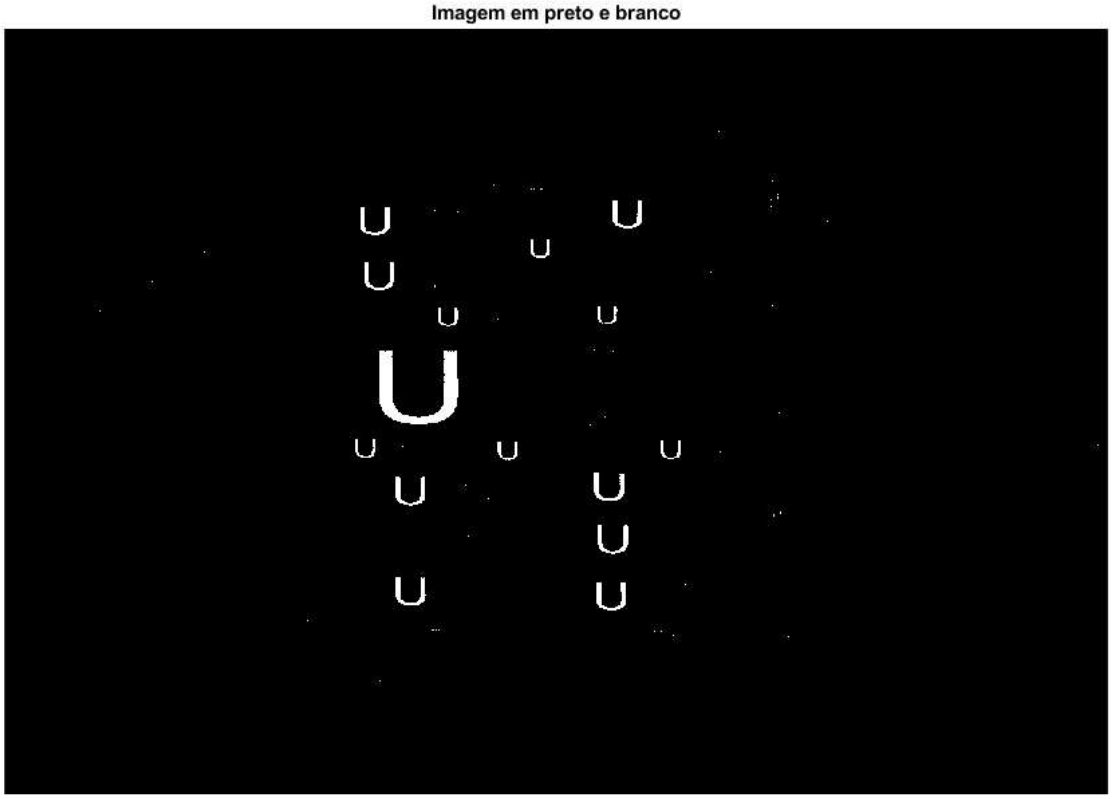


Imagem em preto e branco



Função Recursiva

Lab1_Eduardo_Mafra.m | trabalho2.m | verifica.m

```
function verifica(i,j)
global bw;
bw(i,j)=0;
for k=-1:1
    for c=-1:1
        if bw(i+k,j+c)==1
            verifica(k+i,j+c);
        end
    end
end
end
```

Published with MATLAB® R2018a