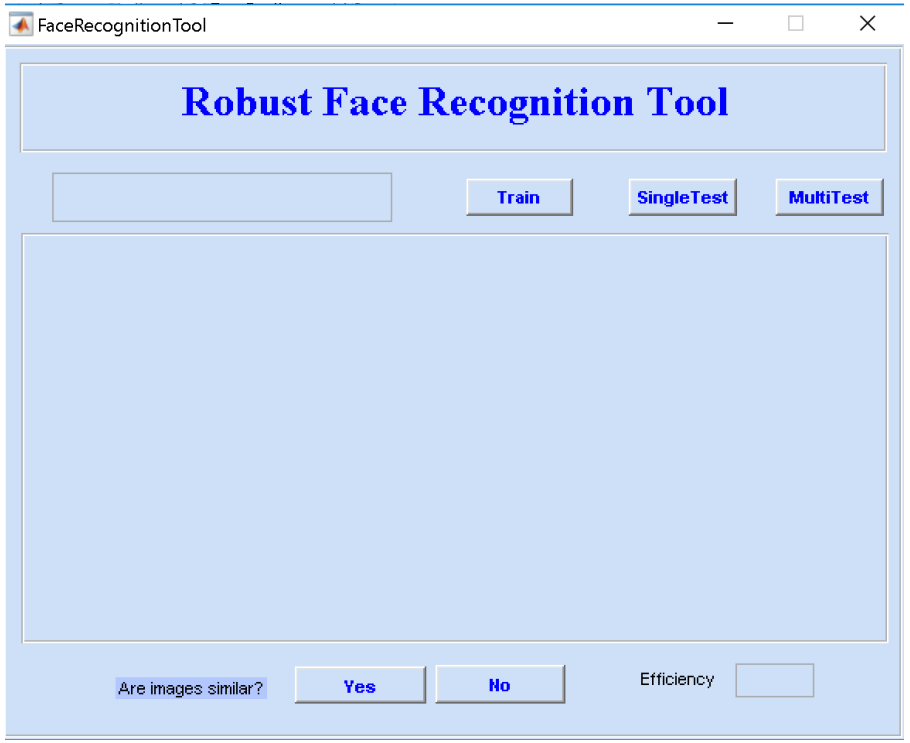
Face recognition Tool - Brief

Paulo Henrique de Castro Oliveira

Universidade de Brasília
Laboratório de Processamento de Sinais em Arranjos
departamento de engenharia elétrica

Face Recognition Tool



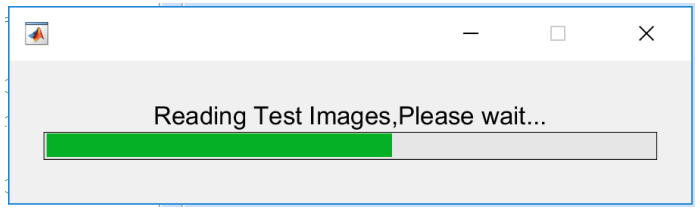
1o - Clicar no botão Train

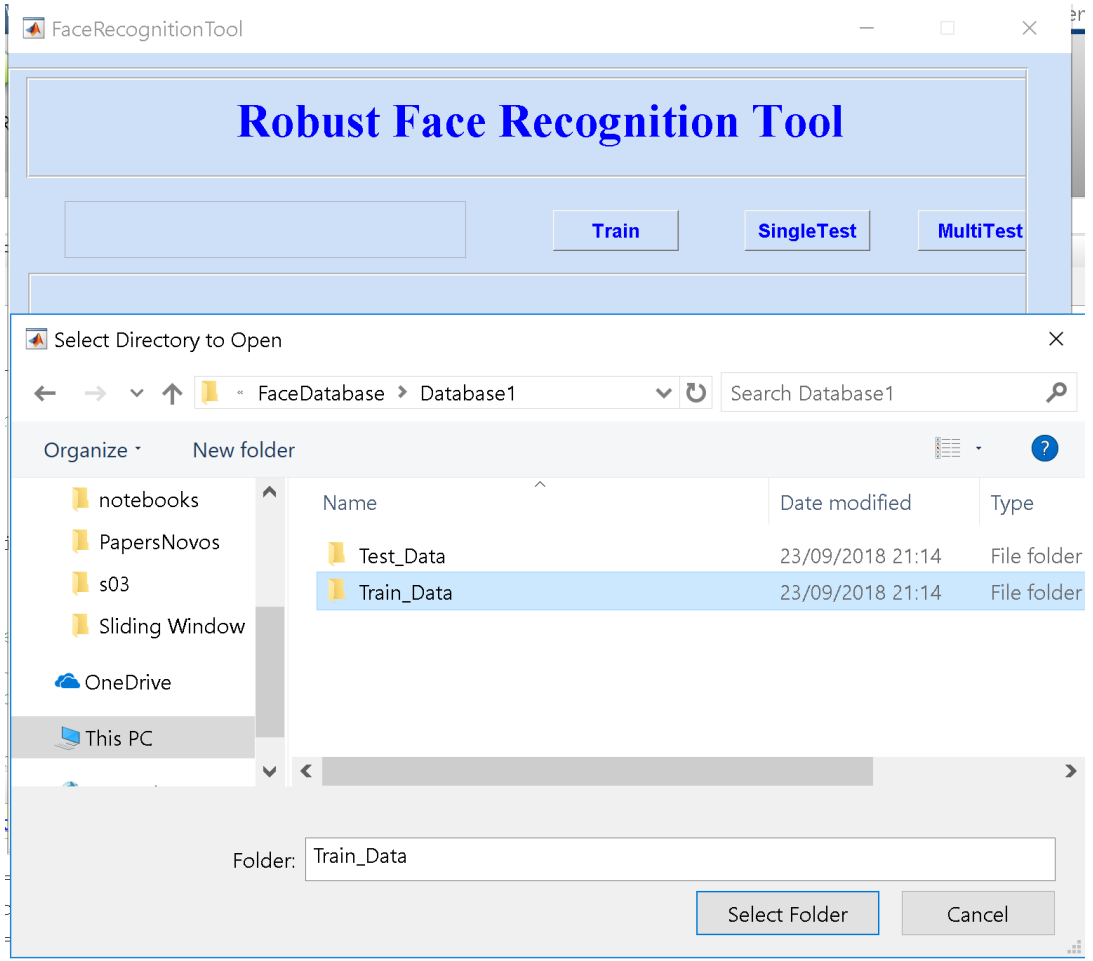
2o - Selecionar a pasta FaceDatabase\Database1\Train_Data

3o - Clicar no botão SingleTest

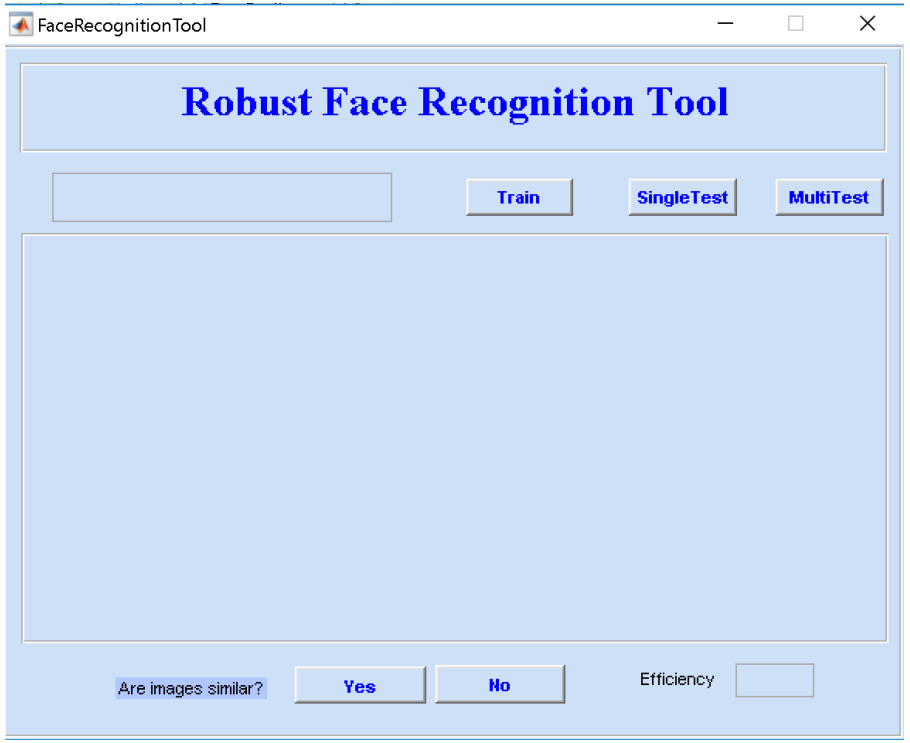
4o - Selecionar alguma imagem dentro de alguma pasta no diretório FaceDatabase\Database1\Test_Data

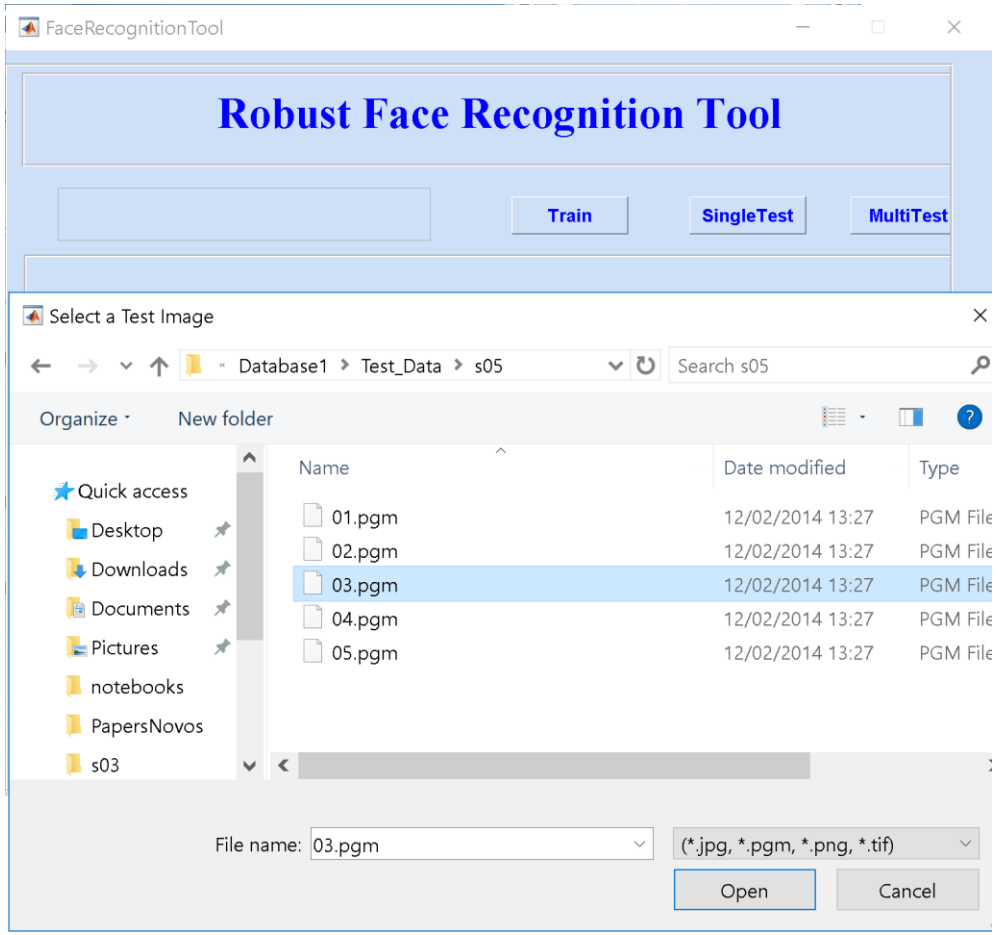
Face Recognition Tool



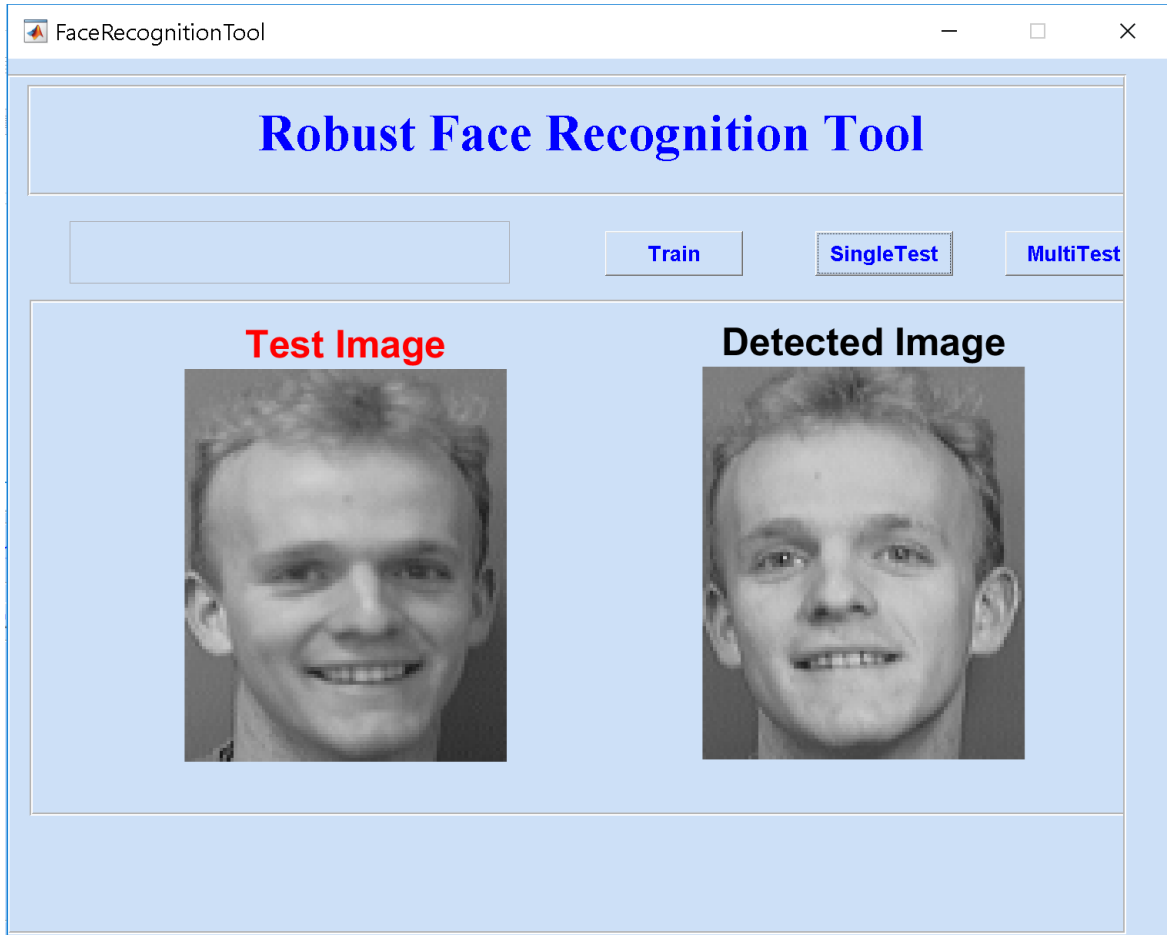


Face Recognition Tool





Face Recognition Tool



Face Recognition Tool – Funções

```
function FaceRecognitionTool_OpeningFcn(hObject, eventdata, handles,
varargin)
```

```
function varargout = FaceRecognitionTool_OutputFcn(hObject,
eventdata, handles)
```

```
function pushbutton4_Callback(hObject, eventdata, handles)
```

Função responsável pelo treinamento do dicionário

```
function pushbutton5_Callback(hObject, eventdata, handles)
```

Função responsável pelo singletest – Identificação de uma única imagem após o treinamento do dicionário

```
function pushbutton6_Callback(hObject, eventdata, handles)
```

Função responsável multitest do dicionário – Identificação de múltiplas imagens após o treinamento do dicionário

```
function togglebutton3_Callback(hObject, eventdata, handles)
```

Face Recognition Tool

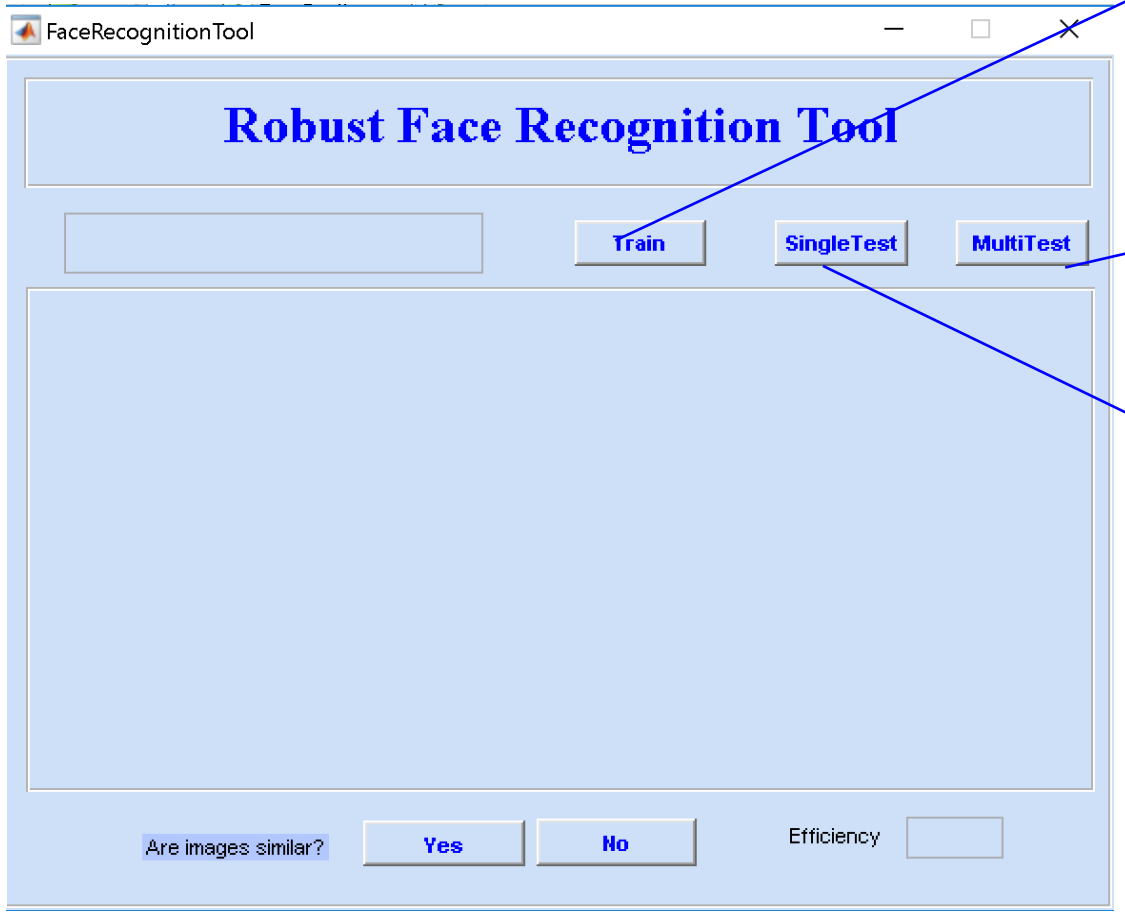
```
function togglebutton3_Callback(hObject, eventdata, handles)

function togglebutton4_Callback(hObject, eventdata, handles)

function edit1_CreateFcn(hObject, eventdata, handles)

function edit2_CreateFcn(hObject, eventdata, handles)
```

Face Recognition Tool



pushbutton4_Callback

pushbutton5_Callback

pushbutton6_Callback

Face Recognition Tool

```
function togglebutton3_Callback(hObject, eventdata, handles)

function togglebutton4_Callback(hObject, eventdata, handles)

function edit1_CreateFcn(hObject, eventdata, handles)

function edit2_CreateFcn(hObject, eventdata, handles)
```

Face Recognition Tool – Train Block

```
% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global A m1 n1 No_Files_In_Class_Folder Class_Count Training_Set_Folder

Training_Set_Folder = [uigetdir(''),'\'];
m1=6
n1=3
TS_Vector = dir(Training_Set_Folder);
No_Folders_In_Training_Set_Folder = length(TS_Vector);
File_Count = 1;
Class_Count = 1;
h = waitbar(0,'Reading Test Images,Please wait...');
for k = 3:No_Folders_In_Training_Set_Folder
    waitbar(k/(No_Folders_In_Training_Set_Folder-2))
    Class_Folder = [Training_Set_Folder '\' TS_Vector(k).name,'\'];
    CF_Tensor = dir(Class_Folder);
    No_Files_In_Class_Folder(Class_Count) = length(CF_Tensor)-2;
    %     strr = sprintf('Reading Test Images...!, # of Classes = %d, Now Reading %d ',No_Folders_In_Training_Set_Folder-2
    %     set(handles.edit3,'String',strr);
    drawnow;
```

Face Recognition Tool – Train Block

```
    for p = 3:No_Files_In_Class_Folder(Class_Count)+2
        Tmp_Image_Path = Class_Folder;
        Tmp_Image_Name = CF_Tensor(p).name;
        Tmp_Image_Path_Name = [Tmp_Image_Path,Tmp_Image_Name];
        if strcmp(Tmp_Image_Name,'Thumbs.db')
            break
        end
        test = imread(Tmp_Image_Path_Name);
        if length(size(test))==3
            Tmp_Image = rgb2gray(test);
        else
            Tmp_Image = test;
        end
        Tmp_Image_Down_Sampled = double(imresize(Tmp_Image,[m1 n1]));
        Image_Data_Matrix(:,File_Count) = Tmp_Image_Down_Sampled(:);
        File_Count = File_Count+1;

    end
    Class_Count = Class_Count+1;

end
close(h)
A = Image_Data_Matrix;
A = A/(diag(sqrt(diag(A'*A))));
```

Verifica se a imagem é colorida. Variável possui 3 dimensões.

Converte para cinza

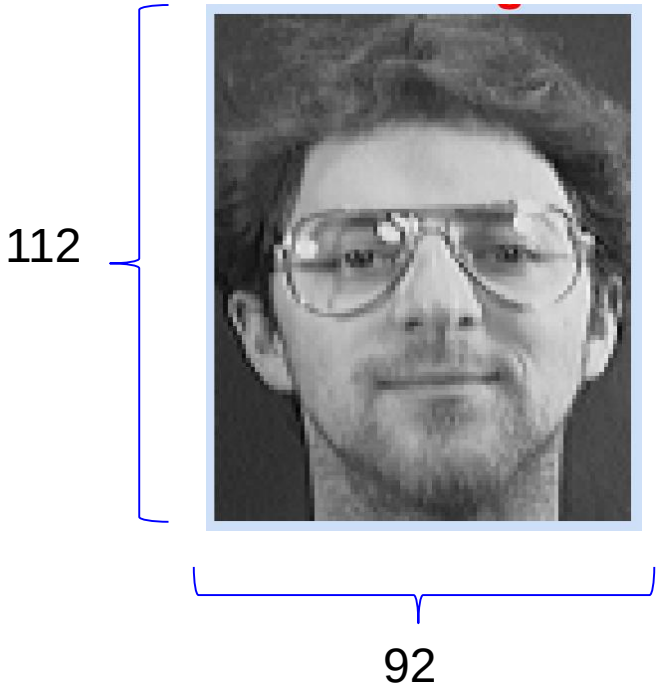
Feature Extraction – Transforma em uma matrix 6x3

Vetorização – transforma a matriz acima em um vetor 18x1

Extrai a norma quadrática de cada coluna da matriz

Face Recognition Tool – Train Block

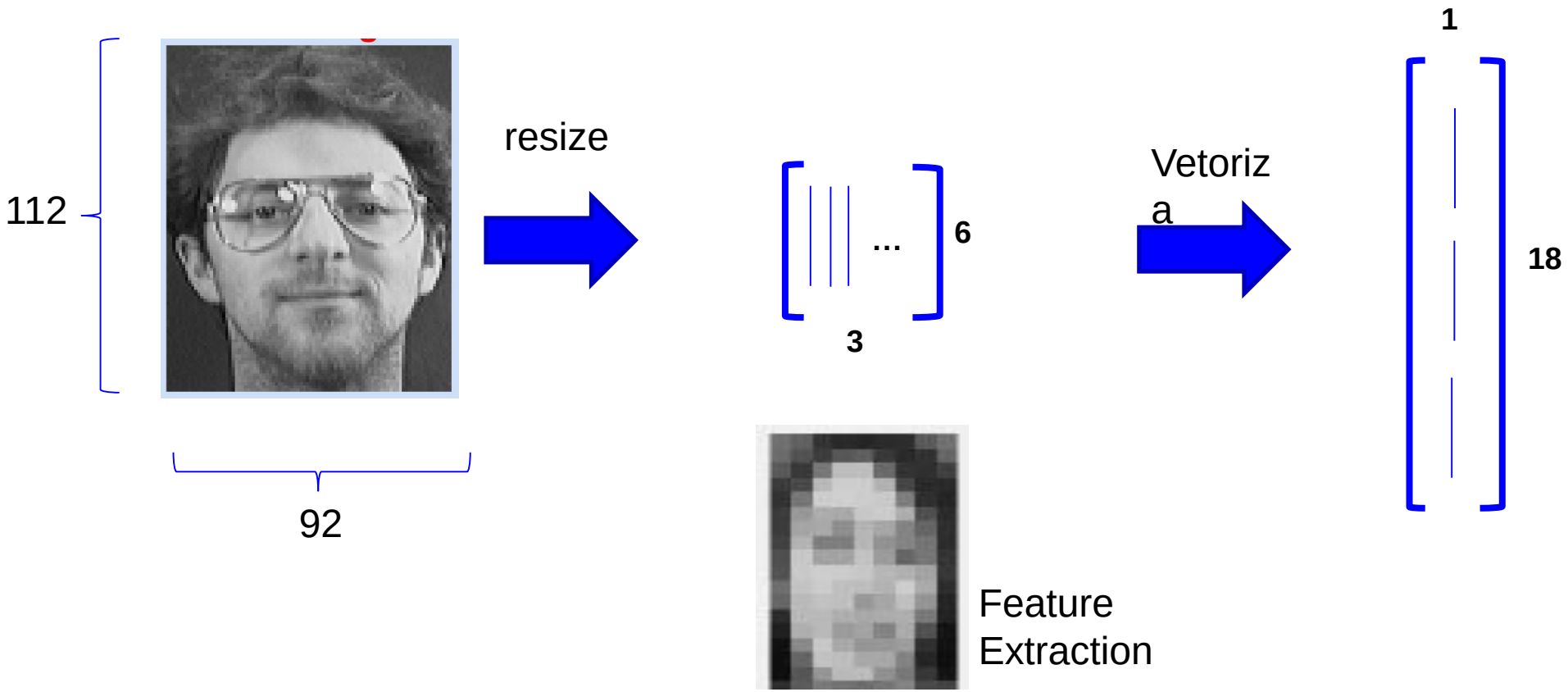
Cada imagem possui dimensões **112 x 92**



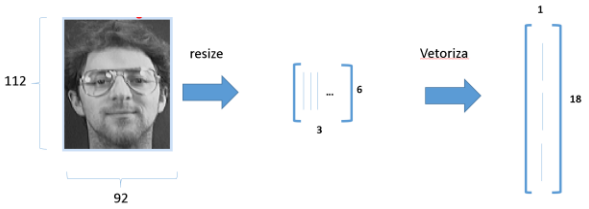
Train Data = possui 40 pastas
- Cada pasta possui 5 imagens.
- Total 200 imagens

Test Data = possui 40 pastas
- Cada pasta possui 5 imagens.
- Total 200 imagens

Face Recognition Tool – Train Block



Face Recognition Tool – Train Block



$$A = \begin{bmatrix} | & | & | & \cdots & | \end{bmatrix}_{18 \times 200}$$

A = Dicionário de imagens

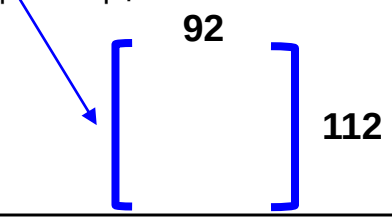
Face Recognition Tool – Test Block

```
for k=1:length(TestFiles)
    if ~strcmp(TestFiles(k,1).name(1),'.')
        Imgfiles=dir([FullPath '\' TestFiles(k).name]);
        for m=1:length(Imgfiles)
            if ~strcmp(Imgfiles(m,1).name(1),'.')
                axes(handles.axes3)
                Test_File = [FullPath '\' TestFiles(k,1).name '\' Imgfiles(m,1).name];
                set(handles.edit1,'string',[TestFiles(k,1).name '\' Imgfiles(m,1).name]);
                imshow(Test_File)
                drawnow;
                test = imread(Test_File);
                if length(size(test))==3
                    Test_Image = rgb2gray(test);
                else
                    Test_Image = test;
                end
```

Imgfiles = string com diretório completo de cada arqivo de imagem.

Verifica se a imagem é colorida. Variável possui 3 dimensões.

Converte para cinza



Face Recognition Tool – Test Block

FaceRecognitionTool.m | test.m

```
1 —   f =[-20000, -15000,-16000];
2 —   A=[50,30,30;2,3,2;1,1,1];
3 —   b=[2000,70,30];
4 —   Aeq=[];
5 —   beq=[];
6 —   lb=[0,0,0];
7 —   ub=[];
8 —   [X , Z] = linprog (f, A, b, Aeq, beq, lb, ub);
```

Command Window

New to MATLAB? See resources for Getting Started.

```
X =

   30.0000
    0.0000
    0.0000


Z =

   -6.0000e+05

>> test
Optimization terminated.
```

Usando função linprog Matlab

max $20000x_1 + 15000x_2 + 16000x_3$ => **min** $-2000x_1 - 15000x_2 - 16000x_3$

$50x_1 + 30x_2 + 30x_3 <= 2000$

$2x_1 + 3x_2 + 2x_3 <= 70$

$x_1 + x_2 + x_3 <= 30$

$x_1, x_2, x_3 >= 0$

```
f =[-20000, -15000,-16000]
A=[50,30,30;2,3,2;1,1,1]
b=[2000,70,30]
Aeq=[]
beq=[]
lb=[0,0,0]
ub=[]
[X , Z] = linprog (f, A, b, Aeq, beq, lb, ub)
```

Linprog example: https://www.youtube.com/watch?v=kavYLZatz44

Face Recognition Tool – Test Block

FaceRecognitionTool.m | test.m

```
f =[-20000, -15000,-16000];
A=[50,30,30;2,3,2;1,1,1];
b=[2000,70,30];
Aeq=[];
beq=[];
lb=[0,0,0];
ub=[];
[X , Z] = linprog (f, A, b, Aeq, beq, lb, ub);
```

Command Window

New to MATLAB? See resources for Getting Started.

```
X =

   30.0000
    0.0000
    0.0000


Z =

   -6.0000e+05

>> test
Optimization terminated.
```

linprog

Solve linear programming problems

Linear programming solver

Finds the minimum of a problem specified by

$$\min_x f^T x \text{ such that } \begin{cases} A \cdot x \le b, \\ Aeq \cdot x = beq, \\ lb \le x \le ub. \end{cases}$$

f, *x*, *b*, *beq*, *lb*, and *ub* are vectors, and *A* and *Aeq* are matrices.

Syntax

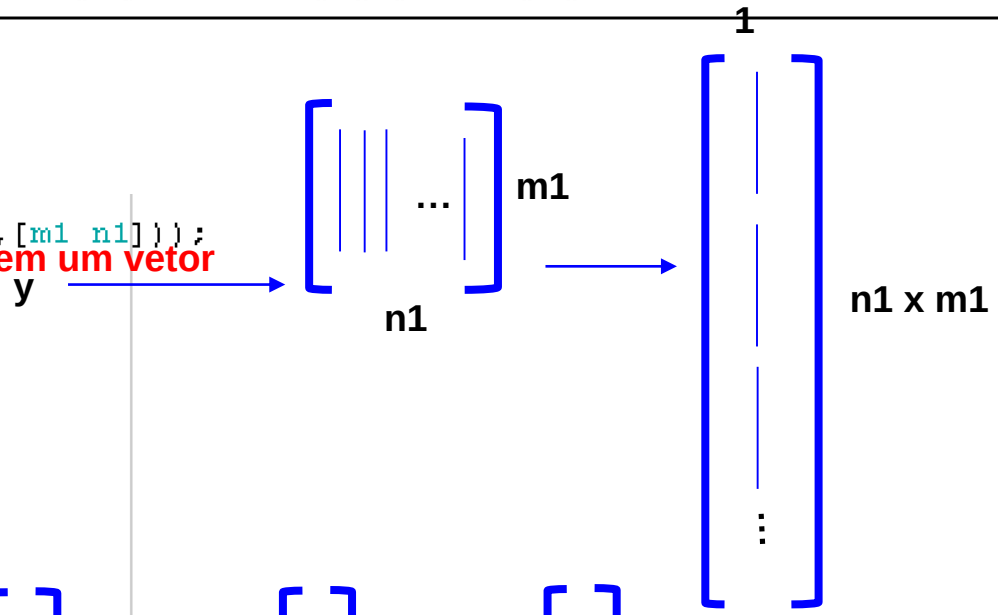
```
x = linprog(f,A,b)
x = linprog(f,A,b,Aeq,beq)
x = linprog(f,A,b,Aeq,beq,lb,ub)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0,options)
x = linprog(problem)
[x,fval] = linprog(__)
[x,fval,exitflag,output] = linprog(__)
[x,fval,exitflag,output,lambda] = linprog(__)
```

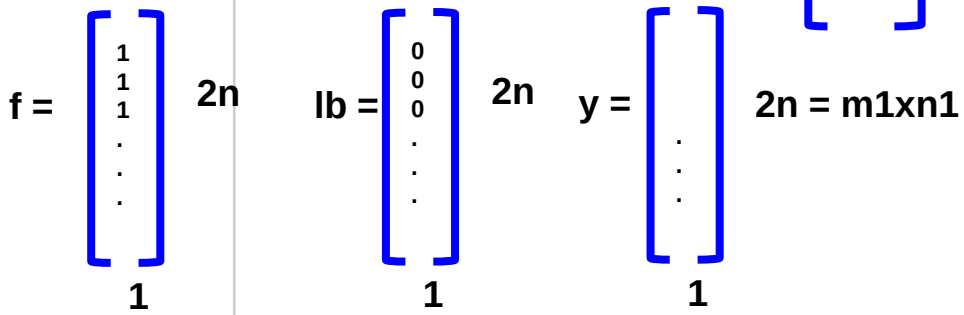
Linprog example: https://www.youtube.com/watch?v=kavYLZatz44

Face Recognition Tool – Test Block

```
Test_Image_Down_Sampled = double(imresize(Test_Image,[m1 n1]));
y = Test_Image_Down_Sampled(:);
n = size(A,2);
%                   cvx_quiet true
%                   cvx_begin
%                   variable x1(n)
%                   minimize norm(x1,1)
%                   subject to
%                   A*x1 == y;
%                   cvx_end
% figure,plot(x1);
f=ones(2*n,1);
Aeq=[A -A];
lb=zeros(2*n,1);
x1 = linprog(f,[],[],Aeq,y,lb,[],[],[]);
x1 = x1(1:n)-x1(n+1:2*n);
```

Transforma a matriz em um vetor coluna.





$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0 \ldots x_{2n} \geq 0 \Rightarrow$ lb [0,0 ...]

Face Recognition Tool – Test Block

```
n = size(A,2);
%                   cvx_quiet true
%                   cvx_begin
%                   variable x1(n)
%                   minimize norm(x1,1)
%                   subject to
%                   A*x1 == y;
%                   cvx_end
% figure,plot(x1);
f=ones(2*n,1);
Aeq=[A -A];
lb=zeros(2*n,1);
x1 = linprog(f,[],[],Aeq,y,lb,[],[],[]);
x1 = x1(1:n)-x1(n+1:2*n);
```

x_1 = vetor que armazena as variáveis que minimizam a função.

$$f = \min x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \ \dots \ x_{2n}$$

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 + a_{1n}x_n \dots - a_{11}x_{n+1} - a_{12}x_{n+2} - a_{1n}x_{2n} = y_1$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 + a_{2n}x_n \dots - a_{11}x_{n+1} - a_{12}x_{n+2} - a_{1n}x_{2n} = y_2$$

$$\dots$$

$$a_{p1}x_1 + a_{p2}x_2 + a_{p3}x_3 + a_{p4}x_4 + a_{p5}x_5 + a_{p6}x_6 \ \dots \ a_{p2n}x_{2n} = y_p$$

$$\begin{bmatrix} A & | & -A \end{bmatrix} \quad p = m1 \times n1$$

$2n$

Face Recognition Tool – Test Block

$$f = \begin{bmatrix} 400 \\ \\ \end{bmatrix}$$

$$18 \begin{bmatrix} A & | & -A \end{bmatrix} = \begin{bmatrix} y \\ \\ 400 \end{bmatrix}$$

$$Lb = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ 0 \end{bmatrix}$$

Lingprog(f,Aeq,lb,y,[],[])

linprog

Solve linear programming problems

Linear programming solver

Finds the minimum of a problem specified by

$$\min_x f^T x \text{ such that } \begin{cases} A \cdot x \le b, \\ Aeq \cdot x = beq, \\ lb \le x \le ub. \end{cases}$$

f, *x*, *b*, *beq*, *lb*, and *ub* are vectors, and *A* and *Aeq* are matrices.

Syntax

```
x = linprog(f,A,b)
x = linprog(f,A,b,Aeq,beq)
x = linprog(f,A,b,Aeq,beq,lb,ub)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0)
x = linprog(f,A,b,Aeq,beq,lb,ub,x0,options)
x = linprog(problem)
[x,fval] = linprog(__)
[x,fval,exitflag,output] = linprog(__)
[x,fval,exitflag,output,lambda] = linprog(__)
```

Face Recognition Tool – Test Block

$$f = \begin{bmatrix} 400 \\ \\ \end{bmatrix}$$

$$18 \begin{bmatrix} A & | & - \\ & | & A \end{bmatrix} = \begin{bmatrix} y \end{bmatrix}$$

$$400$$

$$Lb = \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ 0 \end{bmatrix}$$

Lingprog(f,Aeq,lb,y,[],[])

$f = \min x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \ \dots \ x_{2n}$

$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 + a_{1n}x_n \dots - a_{11}x_{n+1} - a_{12}x_{n+2} - a_{1n}x_{2n} = y_1$

$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 + a_{2n}x_n \dots - a_{11}x_{n+1} - a_{12}x_{n+2} - a_{1n}x_{2n} = y_2$

...

$a_{p1}x_1 + a_{p2}x_2 + a_{p3}x_3 + a_{p4}x_4 + a_{p5}x_5 + a_{p6}x_6 \ \dots \ a_{p2n}x_{2n} = y_p$

Face Recognition Tool – Test Block

```
nn = No_Files_In_Class_Folder;
nn = cumsum(nn);
tmp_var = 0;
k1 = Class_Count-1;
for i = 1:k1
    delta_xi = zeros(length(x1),1);
    if i == 1
        delta_xi(1:nn(i)) = x1(1:nn(i));
    else
        tmp_var = tmp_var + nn(i-1);
        begs = nn(i-1)+1;
        ends = nn(i);
        delta_xi(begs:ends) = x1(begs:ends);
    end
    tmp(i) = norm(y-A*delta_xi,2);
    tmp1(i) = norm(delta_xi,1)/norm(x1,1);
end
```

Armazena a soma acumulativa do vetor nn

norm(…, 1) => a soma dos elementos de cada coluna é 1

```
a =

     1     2     3     4     5

>> cumsum(a)

ans =

     1     3     6    10    15
```

$$6 \times 3 = 18 \quad \begin{bmatrix} y \\ 1 \end{bmatrix} - 18 \begin{bmatrix} A \\ n \end{bmatrix} \quad f = \begin{bmatrix} delta\_xi \\ 1 \end{bmatrix} n$$

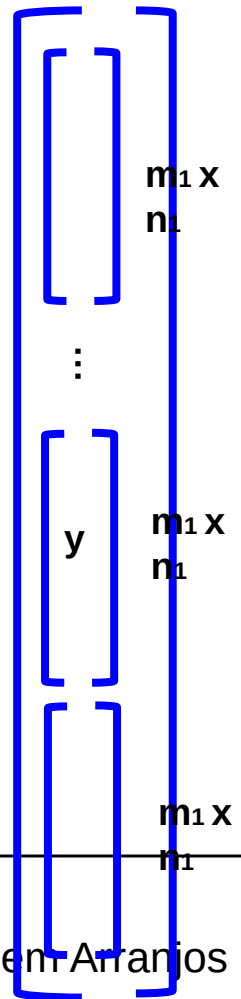
Face Recognition Tool – Test Block

```
nn = No_Files_In_Class_Folder;
nn = cumsum(nn);
tmp_var = 0;
k1 = Class_Count-1;
for i = 1:k1
    delta_xi = zeros(length(x1),1);
    if i == 1
        delta_xi(1:nn(i)) = x1(1:nn(i));
    else
        tmp_var = tmp_var + nn(i-1);
        begs = nn(i-1)+1;
        ends = nn(i);
        delta_xi(begs:ends) = x1(begs:ends);
    end
    tmp(i) = norm(y-A*delta_xi,2);
    tmp1(i) = norm(delta_xi,1)/norm(x1,1);
end
```

Armazena a soma acumulativa do vetor nn

norm(…, 1) => a soma dos elementos de cada coluna é 1

$tmp = m_1 \times n_1 \times k_1$



Face Recognition Tool – Test Block

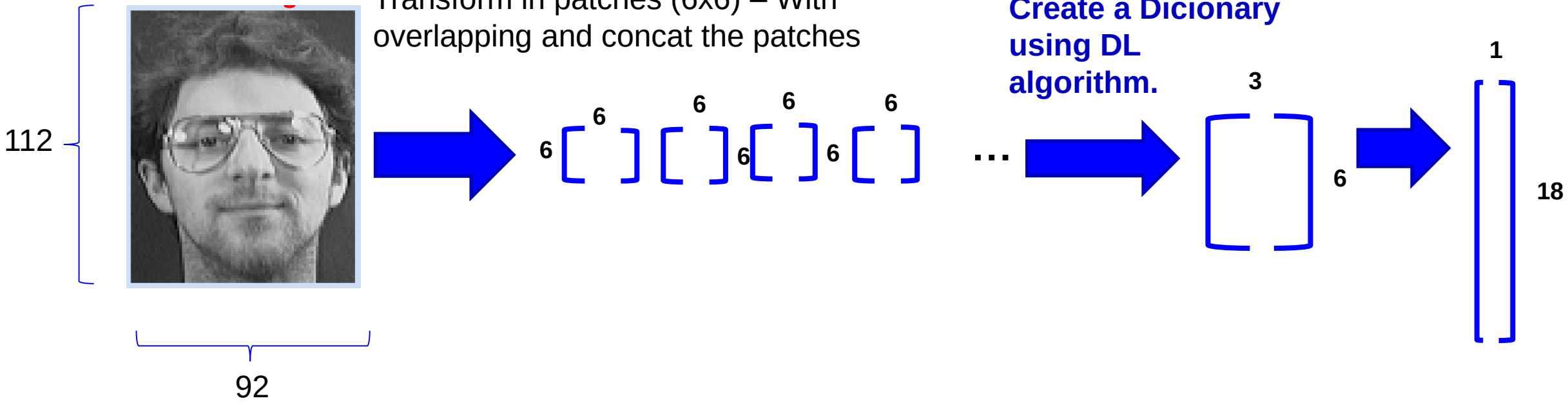
- Vetor com informações de todos os arquivos nas pastas.

- Compõe uma struct.

```
TotImg=TotImg+1;
Sparse_Conc_Index(TotImg) = (k1*max(tmp1)-1)/(k1-1);
clss = find(tmp==min(tmp));
% figure,plot(tmp)
ssttrr = sprintf('The Test Image Corresponds to Class: %d',clss);
cccc = dir([Training_Set_Folder]);
Which_Folder = dir([Training_Set_Folder,cccc(clss+2).name,'\']);
Which_Image = randsample(3:length(Which_Folder),1);
Image_Path = [Training_Set_Folder,cccc(clss+2).name,'\',Which_Folder(Which_Image).name];
Class_Image = (Image_Path);
axes(handles.axes4);
imshow(Class_Image)
```

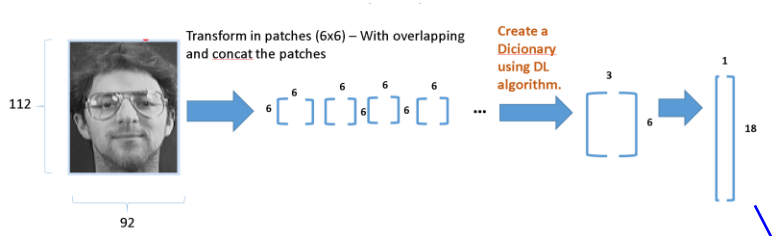
Encontra o menor vetor tmp = vetor com o menor valor para a 2a norma = vetor mais parecido

Face Recognition Tool – Our purpuse



***Resize* matlab** function provide Feature Extraction – Loss of information.

Face Recognition Tool – Our Purpuse



$$A = \left[\; | \quad | \quad | \quad \ldots \quad | \; \right]_{18} \quad \text{(200 colunas)}$$

A = Dicionário de imagens