LABORATÓRIO NACIONAL DE ENGENHARIA CIVIL

SISTEMA DE PREVISÃO E ALERTA DE INUNDAÇÕES EM ZONAS COSTEIRAS E PORTUÁRIAS

PTDC/AAC-AMB/120702/2010

Ação de Formação

MODELAÇÃO COMPUTACIONAL EM MATLAB
DOS FUNDAMENTOS ÀS APLICAÇÕES

APLICAÇÃO DA MODELAÇÃO COMPUTACIONAL EM MATLAB
PÓS-PROCESSAMENTO DE DADOS DE LEVANTAMENTOS ESTEREOFOTOGRAMÉTRICOS EM MODELOS FÍSICOS DE QUEBRA-MARES DE TALUDES

Lisboa, Novembro de 2013

Relatório HIDRALERTA 04/2013

INDICE

INDICE DE FIGURAS

1. INTRODUÇÃO

Atualmente, as plataformas para elaboração de programas de cálculo científico as plataformas do tipo MATLAB são as que mais potenciam o desenvolvimento das competências MST (Mathematics, Science, & Technology). Assim sendo, esta ação de formação teve como principal objetivo mostrar que a programação em ambiente MATLAB é "muito confortável" e permite desenvolver aplicações computacionais de grande eficiência com boas interfaces e com saídas gráficas de grande qualidade.

O programa do curso versou essencialmente sobre:

- Fundamentos de programação em MATLAB;
- Medição e análise de vibrações num modelo físico e conceitos básicos de dinâmica de estruturas;
- Desenvolvimento de aplicações para análise de estruturas com base no MEF;
- Desenvolvimento de interfaces com o GUIDE (Graphical User Interfaces Development Environment);

O presente trabalho pretendeu aplicar os conhecimentos adquiridos ao longo do curso, à resolução de um caso concreto de automatização do tratamento de dados de levantamentos estereofotogramétricos e insere-se no âmbito do Projeto HIDRALERTA - Sistema de previsão e alerta de inundações em zonas costeiras e portuárias financiado pela Fundação para a Ciência e a Tecnologia (contrato PTDC/AAC-AMB/120702/2010), no qual os programas de cálculo científico do tipo MATLAB são muito utilizados.

Este projeto está a ser desenvolvido no LNEC - Laboratório Nacional de Engenharia Civil em conjunto com a Universidade Nova de Lisboa (Faculdade de Ciências e Tecnologia e Faculdade de Ciências Sociais e Humanas) e a Universidade dos Açores, e tem como principal objetivo desenvolver o Sistema de Previsão e Alerta de Inundações em Zonas Costeiras e Portuárias HIDRALERTA. Em especial, inclui as seguintes tarefas:

1. Criação de uma ferramenta de uso amigável que permita:

- Avaliação do risco por intermédio de mapas de risco que constituam uma ferramenta de apoio à decisão pelas entidades competentes. Estes mapas são construídos com longas séries temporais de previsões da agitação marítima ou com cenários pré-definidos associados às mudanças climáticas e/ou eventos extremos
- Avaliação em tempo real de situações de emergência e a emissão de alertas às entidades competentes sempre que se preveja estar em causa a segurança de pessoas, bens ou atividades desenvolvidas nessas zonas;

2. Desenvolvimento de um protótipo para o porto da Praia da Vitória e da zona de Lisboa-Vale do Tejo.

Em qualquer destas tarefas, o recurso à programação em MATLAB é muito frequente, pelo que o presente relatório pretende demonstrar a aplicação dos conhecimentos de modelação computacional em MATLAB adquiridos durante o curso, ao pós-processamento de dados de levantamentos.

No ponto seguinte é feito o enquadramento da aplicação da modelação computacional em MATLAB ao tratamento de dados de levantamentos estereofotogramétricos. No ponto 3 é feita uma breve descrição dos códigos desenvolvidos. Finalmente, no ponto 4 são tecidas as considerações finais acerca do trabalho desenvolvido.

2. ENQUADRAMENTO DA APLICAÇÃO DA MODELAÇÃO COMPUTACIONAL EM MATLAB AO TRATAMENTO DE DADOS DE LEVANTAMENTOS ESTEREOFOTOGRAMÉTRICOS

A quantificação da eficácia do dimensionamento prévio do manto resistente de um quebra-mar é conseguida através de ensaios em modelo físico reduzido, onde é feita a análise do comportamento hidráulico e/ou estrutural dos seus elementos constituintes.

Para avaliar o nível de dano ocorrido ao longo dos ensaios em modelo reduzido, recorre-se por vezes ao levantamento de perfis transversais da estrutura ensaiada. De modo a facilitar essa tarefa, tem vindo a testar-se recentemente, no LNEC, um método de levantamento da envolvente de taludes de quebra-mares baseado em estéreo-fotogrametria.

Este processo permite obter a representação tridimensional da realidade a partir de pares de fotografias dessa mesma realidade, ou cena, tiradas de localizações ligeiramente afastadas. Comparando levantamentos tridimensionais antes e após os ensaios, podem ser determinadas as diferenças entre perfis levantados e consequentemente a área erodida.

Esta técnica foi já testada de um modo intensivo (Lemos, 2010) e (Contente, 2012), em ensaios bidimensionais de quebra-mares de taludes, estando atualmente a ser transposta para ensaios tridimensionais.

O software existente

O método utilizado faz uso de um pacote de *software* (Ferreira, 2006), que permite corrigir a refração da luz na interface ar-água, permitindo realizar os levantamentos sem que seja necessário esvaziar o canal. O equipamento utilizado consiste em duas câmaras fotográficas montadas numa configuração fixa e aptas a disparar duas fotografias simultâneas, controladas por dois computadores portáteis. (Figura 1).

IMG_8031_l IMG_8031_r

Figura 1 Equipamento fotográfico utilizado e exemplo de um par fotogramétrico obtido durante um levantamento

O pacote de software, elaborado em MATLAB permite a reconstrução tridimensional usando pares de imagens de uma mesma cena, obtidas apenas com um pequeno desfasamento entre si. O pacote de software utilizado consiste em duas aplicações distintas, referentes à calibração das câmaras e à reconstrução tridimensional das superfícies, que se descrevem de seguida.

Calibração das câmaras:

Consiste na identificação dos parâmetros das câmaras e sua posição perante a cena observada partindo de um conjunto de pares de fotografias de um objeto padrão, a quadrícula mostrada na Figura 2.

O processo de calibração consiste em selecionar os quatro cantos internos de um padrão axadrezado, cuja dimensão da quadrícula servirá de referência para as dimensões da cena reconstruída (Figura 4). A mesma quadrícula serve também como referência para a identificação do plano de água. O primeiro canto selecionado define a origem do referencial e o segundo canto a direção do eixo dos *x*.

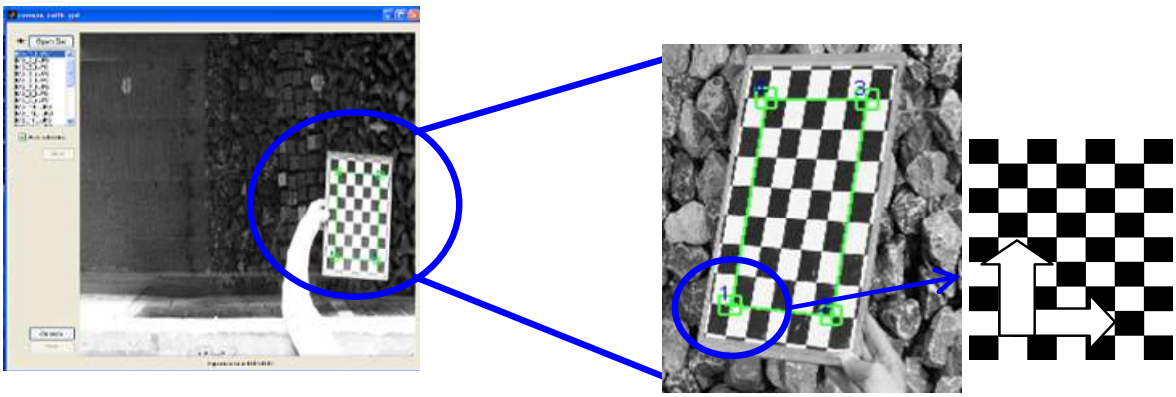


Figura 2 - Processo de seleção dos cantos internos do alvo

Reconstrução tridimensional das superfícies:

Consiste na identificação da profundidade a partir das duas vistas ligeiramente desfasadas do mesmo cenário (Figura 3). Com este software é possível reconstruir a parte emersa, submersa ou ambas as partes, uma vez que este é capaz de retificar o efeito da refração existente no plano da água.

O *software* existente inclui uma aplicação para obtenção de perfis, através da definição dos pontos do corte no écran. No entanto, este *software* obriga à seleção das partes emersa e submersa do perfil, separadamente, conduzindo a seleções pouco rigorosas.

Além disso, quando se pretende tratar dados de numerosos levantamentos onde, para cada levantamento, é necessário caracterizar entre 5 a 10 perfis, a tarefa torna-se extremamente morosa e pouco precisa.

Surgiu, assim, a necessidade de analisar os ficheiros com as matrizes x, y e z das reconstruções de um modo mais expedito. Para isso, recorreu-se a um código de pós-processamento de dados elaborado e MATLAB, o qual tem como output os ternos (x,y,z) do cenário reconstruído, tanto para parte emersa como submersa, extraídos simultaneamente. A partir dos ficheiros com os ternos, é possível extrair perfis (importando os dados para o *Excel*) e superfícies com o auxílio do *Golden Software Surfer* (Figura 3).

Apesar de eficaz, este procedimento não estava devidamente automatizado, especialmente no que diz respeito à importação de dados para as folhas de cálculo e para a geração de grelhas no *Golden Software Surfer.*

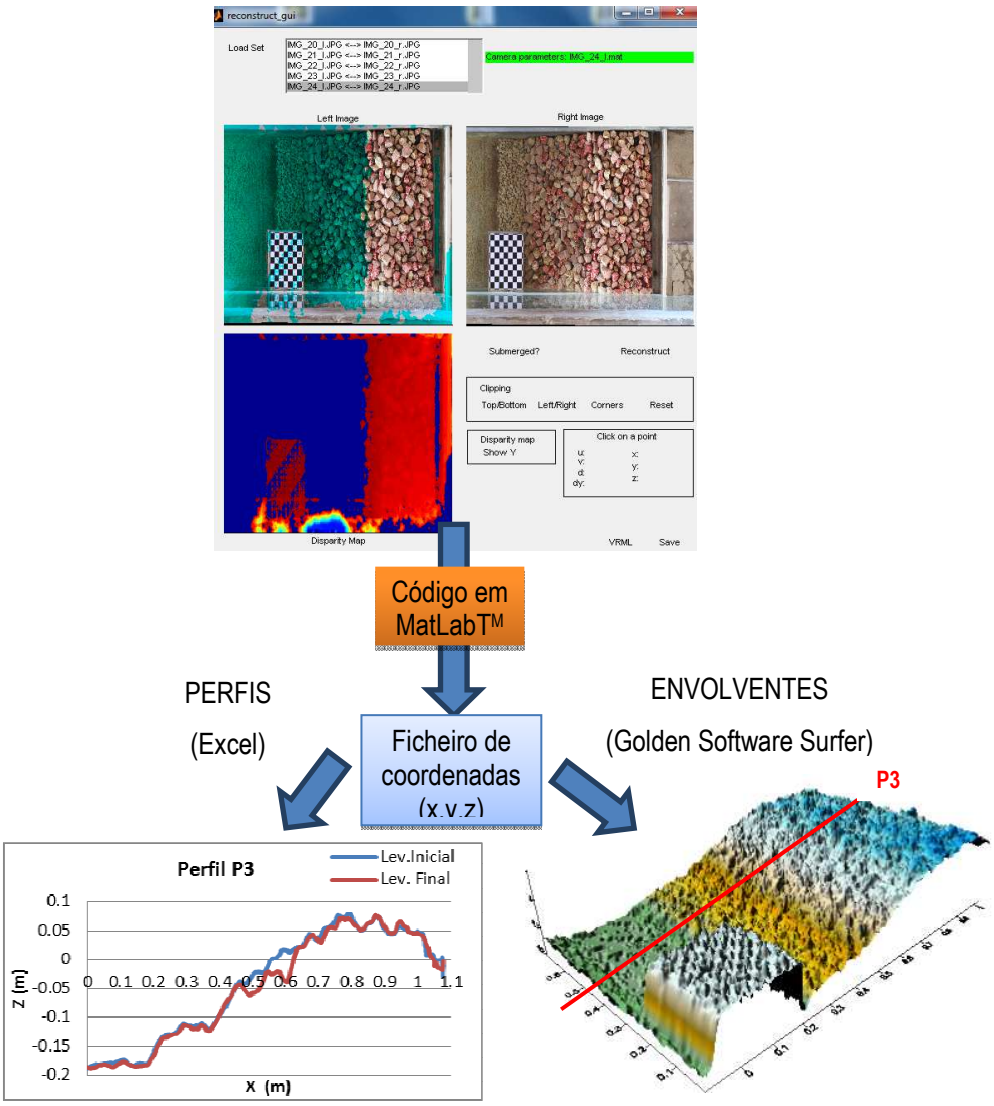


Figura 3 - Extração de perfis e superfícies através da análise das partes submersa e emersa de uma reconstrução.

São descritos nos pontos seguintes, os três códigos que contribuíram para a automatização das tarefas acima referidas. Estes códigos foram desenvolvidos com recurso a instruções aprendidas durante a ação de formação e posteriormente agregados numa interface, de modo a tornar a sua utilização mais amigável.

3. CÓDIGOS PRODUZIDOS EM MATLAB

3.1. Automatização da extração das envolventes

Pretendeu-se, com a criação deste código, designado "*TratamentoAutomatico_Envolv.m*", eliminar o processo intermédio de utilização do *Golden Software Surf para a* criação de grelhas e superfícies individualmente para cada levantamento.

O código para a extração das envolventes foi criado partindo do bloco de código já desenvolvido anteriormente, respeitante à obtenção dos ficheiros com os ternos resultantes dos ficheiros da reconstrução fotogramétrica. Dado funcionar em ciclo, permite fazer essa extração, automaticamente, para os *n* levantamentos realizados.

No final da execução deste primeiro bloco, são obtidos ficheiros de ternos (x,y,z), tanto para a parte emersa, como para a parte submersa do modelo. No final deste bloco, foi criado um segundo bloco de código que lê esses os ficheiros de ternos, automaticamente, seguindo um ciclo, importando os dados e ordenando-os segundo a coluna dos *"x".*

No terceiro bloco de código, foi utilizada a função *griddata* com o objetivo de criar uma grelha em x e *y* à qual se irão ajustar, por interpolação, as cotas *z.* A função *surf* concretiza a representação das superfícies obtidas para cada levantamento (Figura 4).

Através da função *print*, foi possível guardar as figuras em ficheiro, neste caso do tipo "jpg".

No Anexo I deste relatório é apresentado, a sombreado, o código relativo à automatização da extração de envolventes.

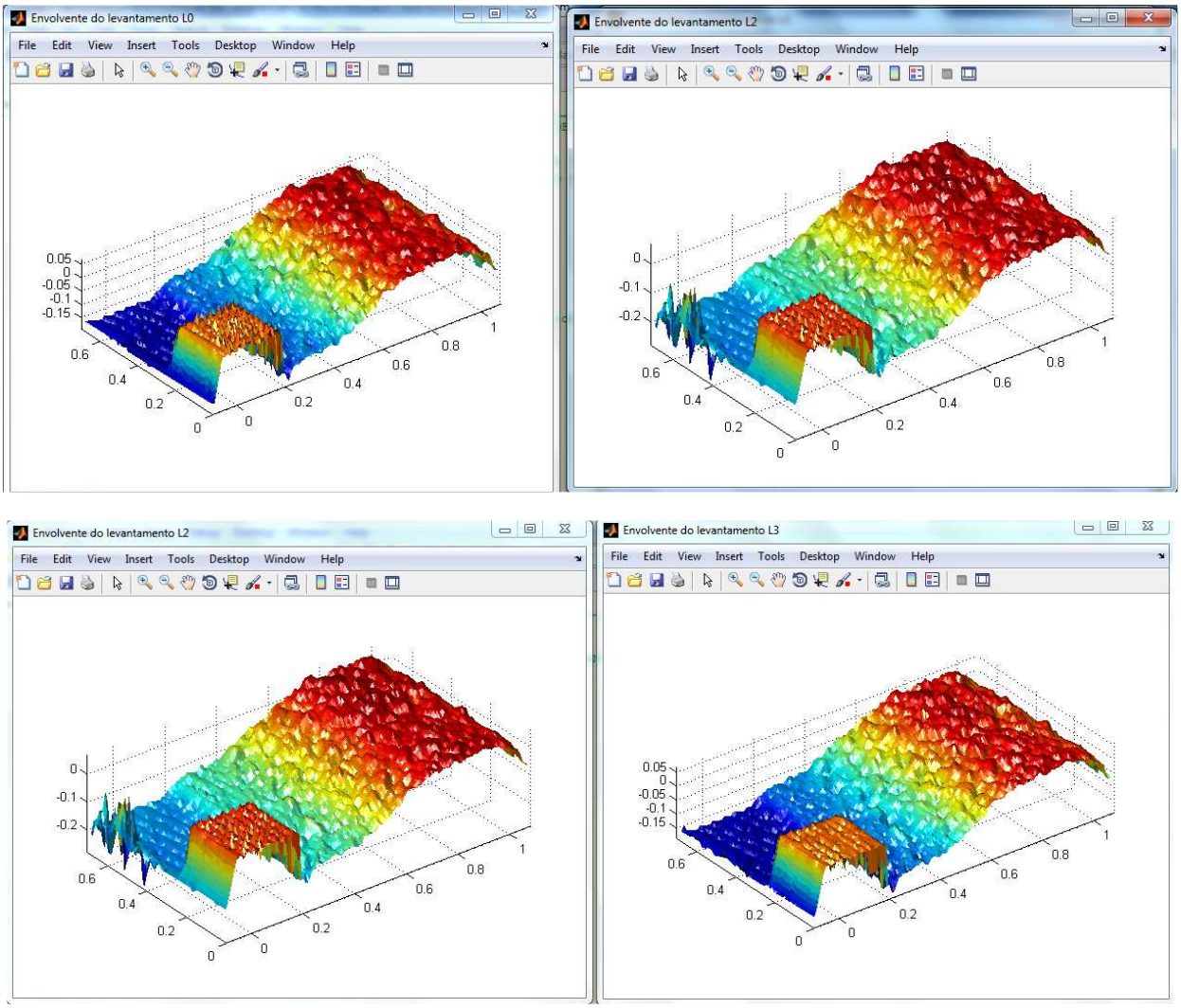


Figura 4 - Representação das superfícies obtidas para cada levantamento

No Anexo III é apresentada uma pequena variante *(EscolherLevantam_Envolv.m),* o qual, introduzindo a função *uigetfile* permite, a partir de ficheiros de ternos já existentes, escolher qual o levantamento do qual se pretende extrair a envolvente (Figura 5).

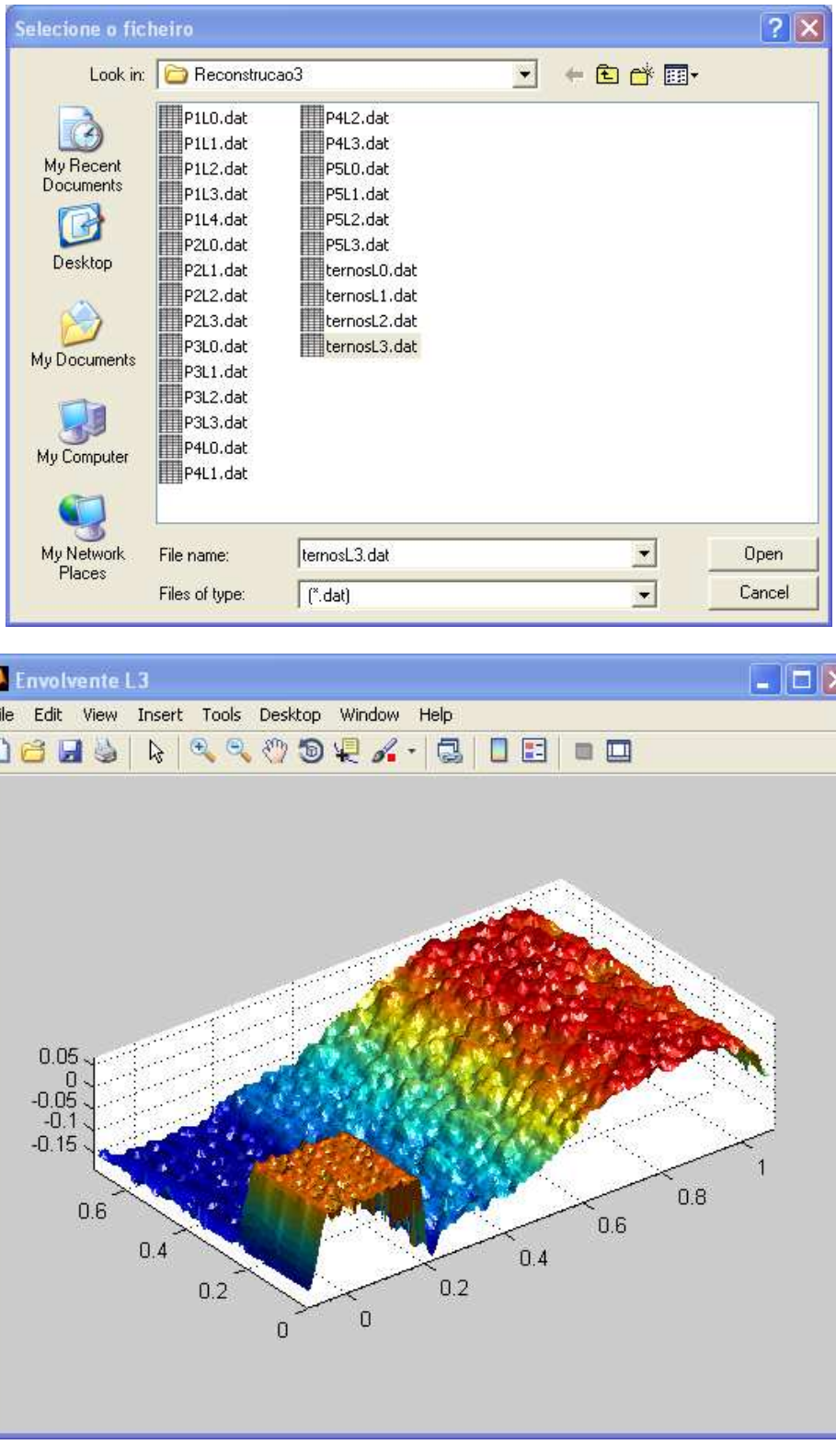


Figura 5 - Escolha do ficheiro de ternos para o qual se pretende extrair a envolvente

3.2. Automatização da extração de perfis

A obtenção de um determinado perfil para cada um dos levantamentos é, talvez, a tarefa mais morosa do pós-processamento dos ficheiros de reconstrução fotogramétrica, dado o rigor necessário à determinação dos pontos por onde se pretende efetuar o corte. Essa tarefa havia já sido anteriormente automatizada, embora o desenho dos perfis dependesse ainda da importação dos pontos coordenados obtidos para folhas de cálculo, onde as cotas eram sujeitas a um alisamento por média móvel de 20 pontos.

Com a inclusão de algumas funções no código *"ExtrairPerfis_P5.m*", após a obtenção dos ficheiros de ternos resultantes da definição dos pontos do corte, estes são lidos, seguindo um ciclo, importando os ternos e ordenando-os segundo a coluna dos *"x".*

Após o alisamento das cotas através da função *smooth*, é utilizada a função *plot* no desenho dos perfis. Recorreu-se à função *colormap* para definir o mapa de cores a utilizar.

A Figura 6 ilustra a evolução do Perfil P5 entre quatro levantamentos consecutivos.

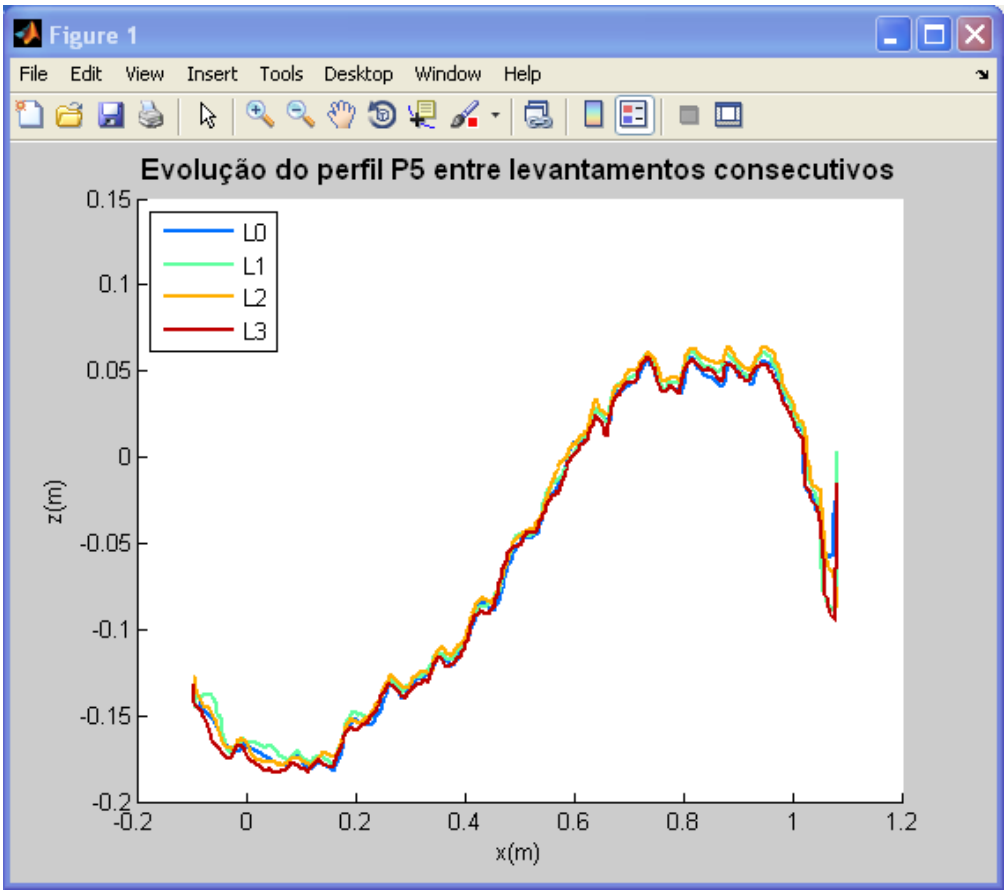


Figura 6 - Evolução do Perfil P5 entre quatro levantamentos consecutivos

3.3. Interface gráfica

De modo a facilitar a utilização dos três blocos de código foi criada uma interface em GUIDE (GRAPHICS USER INTERFACE DEVELOPMENT ENVIRONMENT). A estrutura desta interface é de grande simplicidade, tendo sido criados apenas três *push buttons* relativos aos três códigos desenvolvidos. (Figura 7). Dado funcionarem em ciclos, os códigos *"Obter as envolventes de todos os levantamentos"* e "*Extrair um determinado perfil para todos os levantamentos"* foram dotados de uma *waitbar.*

O código *"Interface.m"* é apresentado no Anexo IV deste relatório.

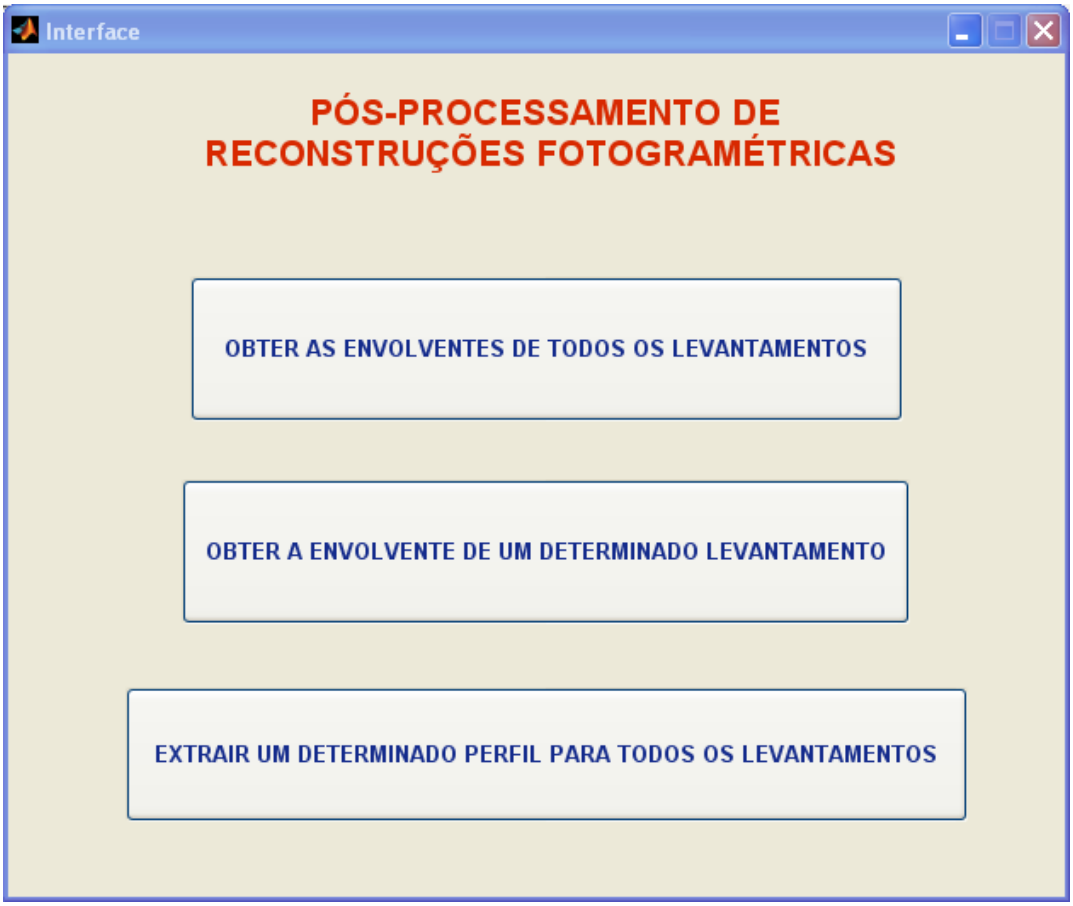


Figura 7 - Aspeto da interface referente aos códigos de pós-processamento das reconstruções fotogramétricas

4. CONSIDERAÇÕES FINAIS

Do trabalho desenvolvido resultou:

- A automatização da obtenção das envolventes de todos os levantamentos;
- A escolha, interativa, da envolvente de um determinado levantamento para analisar;
- A automatização da obtenção de um perfil com uma localização exata, para todos os levantamentos;
- A criação de uma interface que agrupa todas estas aplicações de modo a tornar o seu uso mais amigável.

Desde a importação dos dados em bruto, resultantes dos ficheiros de reconstrução, à sua representação gráfica e exportação para ficheiros "*jpg*", foram conseguidas melhorias consideráveis na precisão e no tempo de processamento, resultantes da eliminação de passos intermédios no seu pós-processamento.

BIBLIOGRAFIA

LEMOS, R. (2010). "Verificação de fórmulas para a evolução da erosão em taludes de quebra-mares". Tese de Mestrado em Engenharia Civil. Instituto Superior de Engenharia de Lisboa.

CONTENTE, J. (2012). "Desenvolvimento de uma Técnica Fotogramétrica, Aplicada à Evolução do Dano em Ensaios em Modelo Reduzido de Quebra-mares de Taludes". Estágio de final de curso. Faculdade de Ciências e Tecnologia

FERREIRA, R., COSTEIRA, J.P., SILVESTRE, C., SOUSA, I. e SANTOS, J.A. (2006). "Using stereo image reconstruction to survey scale models of rubble-mound structures". 1st CoastLab 2006 - International Conference on the application of physical modelling to port and coastal protection. Porto, Portugal, pp.107-116.

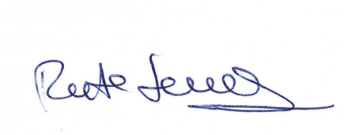
MOORE, H. (2012). "MATLAB for engineers", E source.

Folhas do curso (Powerpoint)

http://www.mathworks.com/help/matlab/

Lisboa, Novembro de 2013

Autor:



Rute Lemos

Técnico superior

ANEXO I

Código “TratamentoAutomatico_Envolv.m” para o pós-processamento automático dos ficheiros resultantes das reconstruções e obtenção de superfícies

```
% OBTER FICHEIROS COM OS PONTOS COORDENADOS A PARTIR DOS FICHEIROS DE
% RECONSTRUÇÃO FOTOGRAMÉTRICA
% ABRE OS FICHEIROS .mat QUE CONTÉM AS MATRIZES X, Y e Z E EXTRAI OS TERNOS
% (x,y,x), escrevendo-os em ficheiros do tipo ".dat". O PROCEDIMENTO É
% REALIZADO TANTO PARA A PARTE EMERSA COMO PARA A SUBMERSA

clear;
maxsize = 2000000; %maior número de pontos a amostrar

ti = 24*60*60*now;

% Criação de uma "waitbar" devido a este cálculo ser bastante demorado

h1=waitbar(0,'Cálculo em execução...');

%Início do ciclo para os "n" levantamentos realizados

for n=0:3 %nº de levantamentos

waitbar(n/4) % Barra a indicar o tempo de cálculo
p=['ReconstrucaoL' int2str(n) '.mat'];
load (p);

    imax = size(reconstruction_b.X,1);
    jmax = size(reconstruction_b.X,2);
    di = ceil(((imax*jmax)/maxsize)^(1/2));

    for s = [1,2,4,8,16]
        d = ceil(s^(1/2)*di)
        k = 1;
        XYZ = [1,1,1];
        for i = 1:d:imax
            for j = 1:d:jmax
                if (reconstruction_b.X(i,j) > -0.10) &
(reconstruction_b.X(i,j) <= 1.08) & (reconstruction_b.Y(i,j) > 0) &
(reconstruction_b.Y(i,j) <= 0.7)
                    XYZ(k,1) = reconstruction_b.X(i,j);
                    XYZ(k,2) = reconstruction_b.Y(i,j);
                    XYZ(k,3) = reconstruction_b.Z(i,j);
                    k = k + 1;
                end
            end
            ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
                    0.5 + ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end

    dlmwrite(['ternosL' int2str(n) '.dat'], XYZ);

 load (p);

    imax = size(reconstruction.X,1);
    jmax = size(reconstruction.X,2);
    di = ceil(((imax*jmax)/maxsize)^(1/2));

    for s = [1,2,4,8,16]
        d = ceil(s^(1/2)*di)
```

```
        k = 1;
        XYZ = [1,1,1];
        for i = 1:d:imax
            for j = 1:d:jmax
                if (reconstruction.X(i,j) > -0.10) & (reconstruction.X(i,j)
<= 1.08) & (reconstruction.Y(i,j) > 0) & (reconstruction.Y(i,j) <= 0.7)
                    XYZ(k,1) = reconstruction.X(i,j);
                    XYZ(k,2) = reconstruction.Y(i,j);
                    XYZ(k,3) = reconstruction.Z(i,j);
                    k = k + 1;
                end
            end
            ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
                    0.5 + ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
        dlmwrite(['ternosL' int2str(n) '.dat'],XYZ,'-append');

end
tf = 24*60*60*now;
tf-ti

clc; clear all; close all
```

```
%% LÊ OS FICHEIROS DE TERNOS ANTERIORMENTE CRIADOS, AUTOMATICAMENTE, SEGUINDO
UM CICLO

for n=0:3 %nº de levantamentos L0, L1, L2,..., Ln

Fich_dados=['ternosL' int2str(n) '.dat']; % Ficheiros de ternos dos
levantamentos
%obtidos das reconstruções (reconstrucaoL0.mat, .... reconstrucaoLn.mat)


delimiterIn=','; %Define qual o delimitador de colunas do ficheiro

xyztotal=importdata(Fich_dados,delimiterIn);% matriz com o ficheiro de dados
totais(com as partes submersa e emersa)
xyztotal_ord=sortrows(xyztotal,1); % Ordena o ficheiro total, pela 1ª coluna

x=xyztotal_ord(:,1);%coluna com o eixo dos xx
y=xyztotal_ord(:,2);%coluna com o eixo dos xx
z=xyztotal_ord(:,3); %coluna dos ZZ (cotas levantadas, mas muito irregulares)


%% UTILIZAÇÃO DA FUNÇÃO GRIDDATA (DATA GRIDDING)

dh=0.005; % Controla a discretização da malha regular para a interpolação
xx=-0.1:dh:1.08;
yy=0:0.01:0.7;
[XI YI]=meshgrid(xx,yy);

ZI = griddata(x,y,z,XI,YI);

leg=['Envolvente do levantamento L',num2str(n)];

figure('Name',leg,'NumberTitle','off');

set(gcf,'Color','white'); %Põe o fundo branco na figura
```

```
surf(XI,YI,ZI,'edgecolor','none') % UTILIZAÇÃO DA FUNÇÃO SURF PARA CRIAÇÃO
DAS SUPERFÍCIES

axis equal
camlight('left')

drawnow;

eval(['print -djpeg Surface_L',num2str(n)]);


%% Funções alternativas para a criação de grelhas e superfícies dos
levantamentos

% Utilização da função TriScatteredInterp (Interpolate scattered data)
%
% F=TriScatteredInterp(x,y,z);
% ZII=F(XI,YI);
% %
% figure(2)
% surf(XI,YI,ZII,'edgecolor','none')
% axis equal
% camlight('left')
% drawnow;
% %
% figure(3)
% mesh(XI,YI,ZII)
% axis equal
% camlight('left')
% %
% figure(4)
% surfc(XI,YI,ZII) % Contour plot under a 3-D shaded surface plot
% axis equal
% camlight('left')
% %
% figure(5)
% surfl(XI,YI,ZII) % Surface plot with colormap-based lighting
% shading interp
% colormap winter % gray, bone, jet, hsv, bone, cooper....etc
% axis equal
% camlight('left')

end
```

ANEXO II

Código “EscolherLevantam_Envolv.m” para obtenção da envolvente de um determinado levantamento

```
%LER O FICHEIRO DE DADOS PERMITINDO AO UTILIZADOR ESCOLHER INTERATIVAMENTE O
FICHEIRO
[Filename, Pathname]=uigetfile('*.dat','Selecione o ficheiro','.\');

%stracat - permite a concatenção de strings
Fich_dados=strcat(Pathname, Filename);

delimiterIn=',';

xyztotal=importdata(Fich_dados,delimiterIn);%matriz com o ficheiro de dados
totais(com a parte submersa e emersa
xyztotal_ord=sortrows(xyztotal,1); %ordenar o ficheiro total, pela 1ª coluna

x=xyztotal_ord(:,1);%coluna com o eixo dos xx
y=xyztotal_ord(:,2);%coluna com o eixo dos xx
z=xyztotal_ord(:,3); %coluna dos ZZ (cotas levantadas, mas muito irregulares)


%% UTILIZAÇÃO DA FUNÇÃO GRIDDATA (DATA GRIDDING)

dh=0.005; % Controla a discretização da malha regular para a interpolação
xx=-0.1:dh:1.08;
yy=0:dh:0.7;
[XI YI]=meshgrid(xx,yy);
ZI = griddata(x,y,z,XI,YI);

figure('Name','Envolvente','NumberTitle','off');
surf(XI,YI,ZI,'edgecolor','none') %UTILIZAÇÃO DA FUNÇÃO SURF PARA CRIAÇÃO DAS
SUPERFÍCIES
axis equal
camlight('right')
drawnow;
eval(['print -djpeg Fig',Filename,'.jpg']);
```

ANEXO III

Código “ExtrairPerfis_P5.m” para obtenção de determinado perfil para todos os levantamentos

```
clc; clear all; close all;
maxsize = 42000000; %maior número de pontos a amostrar
 ti = 24*60*60*now;
    for n=0:3
    p=['ReconstrucaoL' int2str(n) '.mat'];
load (p);

    imax = size(reconstruction_b.X,1);
    jmax = size(reconstruction_b.X,2);
    di = ceil(((imax*jmax)/maxsize)^(1/2));

    for s = [1,2,4,8,16]
        d = ceil(s^(1/2)*di)
        k = 1;
        XYZ = [1,1,1];
        for i = 1:d:imax
            for j = 1:d:jmax
                if (reconstruction_b.X(i,j) > -0.10) &
(reconstruction_b.X(i,j) <= 1.08) & (reconstruction_b.Y(i,j) > 0.70) &
(reconstruction_b.Y(i,j) <= 0.705)
                    XYZ(k,1) = reconstruction_b.X(i,j);
                    XYZ(k,2) = reconstruction_b.Y(i,j);
                    XYZ(k,3) = reconstruction_b.Z(i,j);
                    k = k + 1;
                end
            end
            ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
                    0.5 + ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
        dlmwrite(['P5L' int2str(n) '.dat'], XYZ);

p=['ReconstrucaoL' int2str(n) '.mat'];
        load (p);
    imax = size(reconstruction.X,1);
    jmax = size(reconstruction.X,2);
    di = ceil(((imax*jmax)/maxsize)^(1/2));

    for s = [1,2,4,8,16]
        d = ceil(s^(1/2)*di)
        k = 1;
        XYZ = [1,1,1];
        for i = 1:d:imax
            for j = 1:d:jmax
                if (reconstruction.X(i,j) > -0.10) & (reconstruction.X(i,j)
<= 1.08) & (reconstruction.Y(i,j) > 0.70) & (reconstruction.Y(i,j) <= 0.705)
                    XYZ(k,1) = reconstruction.X(i,j);
                    XYZ(k,2) = reconstruction.Y(i,j);
                    XYZ(k,3) = reconstruction.Z(i,j);
                    k = k + 1;
                end
            end
            ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
                    0.5 + ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
        dlmwrite(['P5L' int2str(n) '.dat'],XYZ,'-append');

end

tf = 24*60*60*now;
```

```
tf-ti
%% Tratar as cotas com média móvel (alisar o perfil). Fazer gráficos dos
levantamentos do perfil

for n=0:3 %nº de levantamentos L0, L1, L2,..., Ln

q=['P5L' int2str(n) '.dat']; %ficheiros de dados do Perfil 1 para os
Levantamentos 1 a n - P1L0, P1,L1...P1Ln

delimiterIn=',';

xyztotal=importdata(q,delimiterIn);%matriz com o ficheiro de dados totais(com
a parte submersa e emersa)

xyztotal_ord=sortrows(xyztotal,1); %ordenar o ficheiro total, pela 1ª coluna

ZZ=xyztotal_ord(:,3);%coluna dos ZZ (cotas levantadas, mas muito irregulares)

xx=xyztotal_ord(:,1);%coluna com o eixo dos xx

% Alisar a coluna dos ZZ (cotas)
zz=smooth(ZZ,20,'moving'); % média móvel da cota, 20 elementos

 cmap=colormap('jet'); %Mapa de 64 Cores: hsv, jet, gray, hot, cool, bone,
copper, pink, flag, prism, jet, rgbplot, hsv2rgb, rgb2hsv.
%Matriz de cores 64x3. A cada cor está associado um código rgb (valores
%entre 0 e 1). Por exemplo, o código rgb para a cor vermelha ('r') é [1 0 0]

figure(1); hold all

leg=['L',num2str(n)];

l=n+1 %nº de linhas a mostrar

plot(xx,zz,'DisplayName',leg, 'color', cmap(l*15,:), 'linewidth', 2)
 %NOTA: Ao fazer cmap(l*10,:), a cor fica a variar de acordo com a variável j
do ciclo for,
%multiplicada por uma dada constante (podemos alterar esta constante), e
assim conseguimos
%apanhar diferentes cores, ou seja:
 % l=1 --> cmap(15,:)
% l=2 --> cmap(30,:)
% l=3 --> cmap(45,:)
% l=4 --> cmap(60,:)
% À medida que o nº de linhas a plotar aumenta, o coeficiente de l tem que
% diminuir

xlabel('x(m)')
ylabel('z(m)')
title('Evolução do perfil P5 entre levantamentos consecutivos','Fontsize',12,
'Fontweight','bold')
legend('-DynamicLegend','Location','Northwest');

% legend (leg,'Location','Northwest')
% legend ('L0','L1','L2','L3','Location','Northwest')


end
```

ANEXO IV

Código “Interface.m”, relativo à criação da interface em guide

```
function varargout = Interface(varargin)
% INTERFACE MATLAB code for Interface.fig
%      INTERFACE, by itself, creates a new INTERFACE or raises the existing
%      singleton*.
%
%      H = INTERFACE returns the handle to a new INTERFACE or the handle to
%      the existing singleton*.
%
%      INTERFACE('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in INTERFACE.M with the given input arguments.
%
%      INTERFACE('Property','Value',...) creates a new INTERFACE or raises
the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Interface_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Interface_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Interface

% Last Modified by GUIDE v2.5 04-Nov-2013 22:38:22

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Interface_OpeningFcn, ...
                   'gui_OutputFcn',  @Interface_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Interface is made visible.
function Interface_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Interface (see VARARGIN)

% Choose default command line output for Interface
handles.output = hObject;

% Update handles structure
```

```
guidata(hObject, handles);

% UIWAIT makes Interface wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = Interface_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in Envolvente.
function Envolvente_Callback(hObject, eventdata, handles)
% hObject    handle to Envolvente (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

%ler o ficheiro de dados permitindo ao utilizador escolher interativamente o
ficheiro
[Filename, Pathname]=uigetfile('*.dat','Selecione o ficheiro','.\');

%stracat - permite a concatenção de strings
Fich_dados=strcat(Pathname, Filename);

delimiterIn=',';

xyztotal=importdata(Fich_dados,delimiterIn);%matriz com o ficheiro de dados
totais(com a parte submersa e emersa
xyztotal_ord=sortrows(xyztotal,1); %ordenar o ficheiro total, pela 1ª coluna

x=xyztotal_ord(:,1);%coluna com o eixo dos xx
y=xyztotal_ord(:,2);%coluna com o eixo dos xx
z=xyztotal_ord(:,3); %coluna dos ZZ (cotas levantadas, mas muito irregulares)


%% Utilização da função griddata (Data gridding)

dh=0.005; % Controla a discretização da malha regular para a interpolação
xx=-0.1:dh:1.08;
yy=0:dh:0.7;
[XI YI]=meshgrid(xx,yy);
ZI = griddata(x,y,z,XI,YI);
figure('Name','Envolvente','NumberTitle','off');
surf(XI,YI,ZI,'edgecolor','none')
axis equal
camlight('right')
drawnow;
eval(['print -djpeg Fig',Filename,'.jpg']); %Cria uma figura em formato "jpg"

% --- Executes on button press in Perfis.
function Perfis_Callback(hObject, eventdata, handles)
% hObject    handle to Perfis (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
clc; clear all; close all;
maxsize = 42000000; %maior número de pontos a amostrar

ti = 24*60*60*now;
    h1=waitbar(0,'Cálculo em execução...');
for n=0:3
    waitbar(n/3) % Barra a indicar o tempo de cálculo
    p=['ReconstrucaoL' int2str(n) '.mat'];
load (p);

    imax = size(reconstruction_b.X,1);
    jmax = size(reconstruction_b.X,2);
    di = ceil(((imax*jmax)/maxsize)^(1/2));

    for s = [1,2,4,8,16]
        d = ceil(s^(1/2)*di)
        k = 1;
        XYZ = [1,1,1];
        for i = 1:d:imax
            for j = 1:d:jmax
                if (reconstruction_b.X(i,j) > -0.10) &
(reconstruction_b.X(i,j) <= 1.08) & (reconstruction_b.Y(i,j) > 0.70) &
(reconstruction_b.Y(i,j) <= 0.705)
                    XYZ(k,1) = reconstruction_b.X(i,j);
                    XYZ(k,2) = reconstruction_b.Y(i,j);
                    XYZ(k,3) = reconstruction_b.Z(i,j);
                    k = k + 1;
                end
            end
            ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
                    0.5 + ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
        dlmwrite(['P5L' int2str(n) '.dat'], XYZ);

p=['ReconstrucaoL' int2str(n) '.mat'];
        load (p);

    imax = size(reconstruction.X,1);
    jmax = size(reconstruction.X,2);
    di = ceil(((imax*jmax)/maxsize)^(1/2));

    for s = [1,2,4,8,16]
        d = ceil(s^(1/2)*di)
        k = 1;
        XYZ = [1,1,1];
        for i = 1:d:imax
            for j = 1:d:jmax
                if (reconstruction.X(i,j) > -0.10) & (reconstruction.X(i,j)
<= 1.08) & (reconstruction.Y(i,j) > 0.70) & (reconstruction.Y(i,j) <= 0.705)
                    XYZ(k,1) = reconstruction.X(i,j);
                    XYZ(k,2) = reconstruction.Y(i,j);
                    XYZ(k,3) = reconstruction.Z(i,j);
                    k = k + 1;
                end
            end
            ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
                    0.5 + ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
```

```
        dlmwrite(['P5L' int2str(n) '.dat'],XYZ,'-append');

end
close(h1); %Fecha a waitbar
tf = 24*60*60*now;
tf-ti

%% Tratar as cotas com média móvel (alisar o perfil). Fazer gráficos dos
levantamentos do perfil

for n=0:3 %nº de levantamentos L0, L1, L2,..., Ln

q=['P5L' int2str(n) '.dat']; %ficheiros de dados do Perfil 1 para os
Levantamentos 1 a n - P1L0, P1,L1...P1Ln

delimiterIn=',';

xyztotal=importdata(q,delimiterIn);%matriz com o ficheiro de dados totais(com
a parte submersa e emersa)

xyztotal_ord=sortrows(xyztotal,1); %ordenar o ficheiro total, pela 1ª coluna

ZZ=xyztotal_ord(:,3);%coluna dos ZZ (cotas levantadas, mas muito irregulares)

xx=xyztotal_ord(:,1);%coluna com o eixo dos xx

% Alisar a coluna dos ZZ (cotas)
zz=smooth(ZZ,20,'moving'); % média móvel da cota, 20 elementos

cmap=colormap('jet'); %Mapa de 64 Cores: hsv, jet, gray, hot, cool, bone,
copper, pink, flag, prism, jet, rgbplot, hsv2rgb, rgb2hsv.
%Matriz de cores 64x3. A cada cor está associado um código rgb (valores
%entre 0 e 1). Por exemplo, o código rgb para a cor vermelha ('r') é [1 0 0]

figure(1); hold all
leg=['L',num2str(n)];

l=n+1 %nº de linhas a mostrar

plot(xx,zz,'DisplayName',leg, 'color', cmap(l*15,:), 'linewidth', 2)

%NOTA: Ao fazer cmap(l*10,:), a cor fica a variar de acordo com a variável j
do ciclo for,
%multiplicada por uma dada constante (podemos alterar esta constante), e
assim conseguimos
%apanhar diferentes cores, ou seja:

% l=1 --> cmap(15,:)
% l=2 --> cmap(30,:)
% l=3 --> cmap(45,:)
% l=4 --> cmap(60,:)
% À medida que o nº de linhas a plotar aumenta, o coeficiente de l tem que
% diminuir

xlabel('x(m)')
ylabel('z(m)')
title('Evolução do perfil P5 entre levantamentos consecutivos','Fontsize',15,
'Fontweight','bold')
legend('-DynamicLegend','Location','Northwest');
```

```
end

% --- Executes on button press in Todos_Levantamentos.

function Todos_Levantamentos_Callback(hObject, eventdata, handles)
% hObject    handle to Todos_Levantamentos (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% OBTER FICHEIROS COM OS PONTOS COORDENADOS A PARTIR DOS FICHEIROS DE
% RECONSTRUÇÃO FOTOGRAMÉTRICA
% ABRE OS FICHEIROS .mat QUE CONTÉM AS MATRIZES X, Y e Z E EXTRAI OS TERNOS
% (x,y,x), escrevendo-os em ficheiros do tipo ".dat". O PROCEDIMENTO É
% REALIZADO TANTO PARA A PARTE EMERSA COMO PARA A SUBMERSA

clc; clear all; close all

maxsize = 2000000; %maior número de pontos a amostrar

ti = 24*60*60*now;

% Criação de uma "waitbar" devido a este cálculo ser bastante demorado

h1=waitbar(0,'Cálculo em execução...');

%Início do ciclo para os "n" levantamentos realizados

for n=0:3 %nº de levantamentos

waitbar(n/4) % Barra a indicar o tempo de cálculo
p=['ReconstrucaoL' int2str(n) '.mat'];
load (p);

    imax = size(reconstruction_b.X,1);
    jmax = size(reconstruction_b.X,2);
    di = ceil(((imax*jmax)/maxsize)^(1/2));

    for s = [1,2,4,8,16]
        d = ceil(s^(1/2)*di)
        k = 1;
        XYZ = [1,1,1];
        for i = 1:d:imax
            for j = 1:d:jmax
                if (reconstruction_b.X(i,j) > -0.10) &
(reconstruction_b.X(i,j) <= 1.08) & (reconstruction_b.Y(i,j) > 0) &
(reconstruction_b.Y(i,j) <= 0.7)
                    XYZ(k,1) = reconstruction_b.X(i,j);
                    XYZ(k,2) = reconstruction_b.Y(i,j);
                    XYZ(k,3) = reconstruction_b.Z(i,j);
                    k = k + 1;
                end
            end
            ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
                    0.5 + ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
```

```
    dlmwrite(['ternosL' int2str(n) '.dat'], XYZ);

 load (p);

    imax = size(reconstruction.X,1);
    jmax = size(reconstruction.X,2);
    di = ceil(((imax*jmax)/maxsize)^(1/2));

    for s = [1,2,4,8,16]
        d = ceil(s^(1/2)*di)
        k = 1;
        XYZ = [1,1,1];
        for i = 1:d:imax
            for j = 1:d:jmax
                if (reconstruction.X(i,j) > -0.10) & (reconstruction.X(i,j)
<= 1.08) & (reconstruction.Y(i,j) > 0) & (reconstruction.Y(i,j) <= 0.7)
                    XYZ(k,1) = reconstruction.X(i,j);
                    XYZ(k,2) = reconstruction.Y(i,j);
                    XYZ(k,3) = reconstruction.Z(i,j);
                    k = k + 1;
                end
            end
            ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
                    0.5 + ((i/d)*(j/d))/((imax/d)*(jmax/d))/2
        end
        dlmwrite(['ternosL' int2str(n) '.dat'],XYZ,'-append');

end
close(h1); %Fecha a waitbar
tf = 24*60*60*now;
tf-ti

clc; clear all; close all


clc; clear all; close all
%%
% LÊ OS FICHEIROS DE TERNOS AUTOMATICAMENTE, SEGUINDO UM CICLO

for n=0:3 %nº de levantamentos L0, L1, L2,..., Ln

Fich_dados=['ternosL' int2str(n) '.dat']; % Ficheiros de ternos dos
levantamentos
%obtidos das reconstruções (reconstrucaoL0.mat, .... reconstrucaoLn.mat)


delimiterIn=','; %Define qual o delimitador de colunas do ficheiro

xyztotal=importdata(Fich_dados,delimiterIn);% matriz com o ficheiro de dados
totais(com as partes submersa e emersa)
xyztotal_ord=sortrows(xyztotal,1); % Ordena o ficheiro total, pela 1ª coluna

x=xyztotal_ord(:,1);%coluna com o eixo dos xx
y=xyztotal_ord(:,2);%coluna com o eixo dos xx
z=xyztotal_ord(:,3); %coluna dos ZZ (cotas levantadas, mas muito irregulares)
```

```
%% Utilização da função griddata (Data gridding)

dh=0.005; % Controla a discretização da malha regular para a interpolação
xx=-0.1:dh:1.08;
yy=0:0.01:0.7;
[XI YI]=meshgrid(xx,yy);

ZI = griddata(x,y,z,XI,YI); % TriScatteredInterp is the recommended
alternative to griddata as it is generally more efficient.

leg=['Envolvente do levantamento L',num2str(n)];

figure('Name',leg,'NumberTitle','off');

set(gcf,'Color','white'); %Põe o fundo branco na figura

surf(XI,YI,ZI,'edgecolor','none')

axis equal
camlight('left')

drawnow;

eval(['print -djpeg Surface_L',num2str(n)]); %Cria uma figura em formato
"jpg"


%% Funções alternativas para a criação de grelhas e superfícies dos
levantamentos

% Utilização da função TriScatteredInterp (Interpolate scattered data)
%
% F=TriScatteredInterp(x,y,z);
% ZII=F(XI,YI);
% %
% figure(2)
% surf(XI,YI,ZII,'edgecolor','none')
% axis equal
% camlight('left')
% drawnow;
% %
% figure(3)
% mesh(XI,YI,ZII)
% axis equal
% camlight('left')
% %
% figure(4)
% surfc(XI,YI,ZII) % Contour plot under a 3-D shaded surface plot
% axis equal
% camlight('left')
% %
% figure(5)
% surfl(XI,YI,ZII) % Surface plot with colormap-based lighting
% shading interp
% colormap winter % gray, bone, jet, hsv, bone, cooper....etc
% axis equal
% camlight('left')

end
```