原创 小李不会编程啊 已于 2022-04-01 18:32:54 修改 阅读量1k 收藏 14 点赞数 2

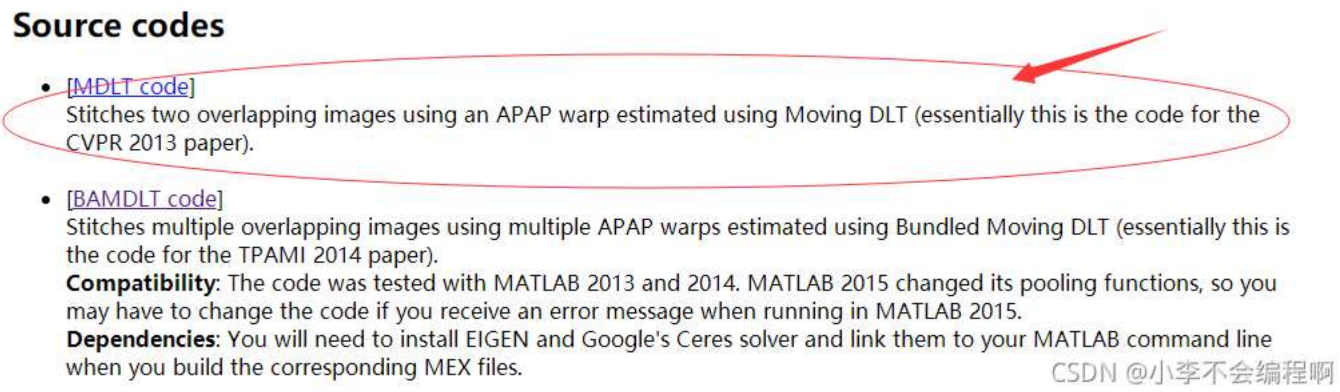
分类专栏： 代码调试记录 文章标签： matlab 图像处理 c++

代码调试记录 专栏收录该内容 3 订阅 17 篇文章

代码见：https://cs.adelaide.edu.au/~tjchin/apap/

详细代码论文

此次实验选用的代码是2013年的版本：



Source codes

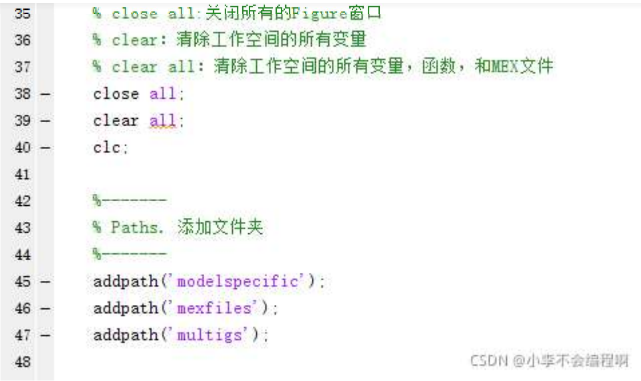
- [MDLT code]
  Stitches two overlapping images using an APAP warp estimated using Moving DLT (essentially this is the code for the CVPR 2013 paper).
- [BAMDLT code]
  Stitches multiple overlapping images using multiple APAP warps estimated using Bundled Moving DLT (essentially this is the code for the TPAMI 2014 paper).
  Compatibility: The code was tested with MATLAB 2013 and 2014. MATLAB 2015 changed its pooling functions, so you may have to change the code if you receive an error message when running in MATLAB 2015.
  Dependencies: You will need to install EIGEN and Google's Ceres solver and link them to your MATLAB command line when you build the corresponding MEX files.

由于文档中的代码块截图不一定清楚，需要的可以去上面的网址下载代码对照着看。

文章目录

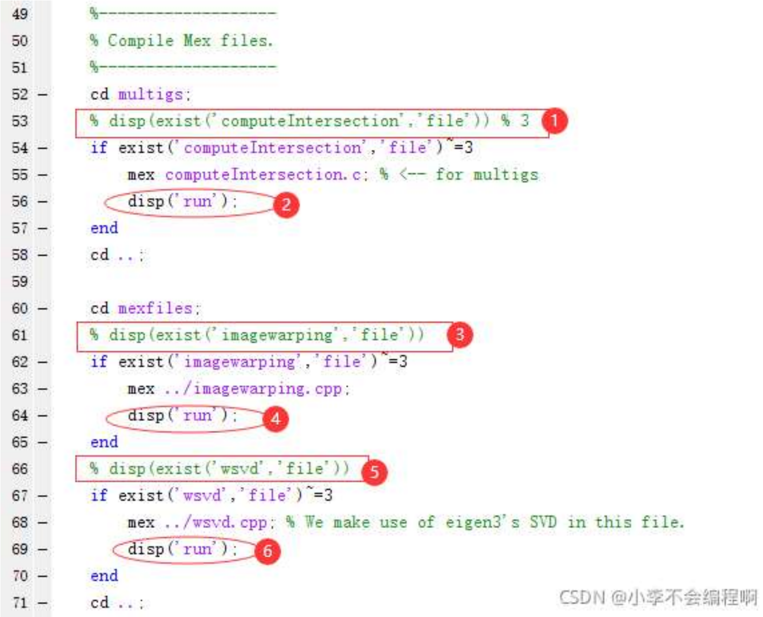
一、代码

1.1、加载文件



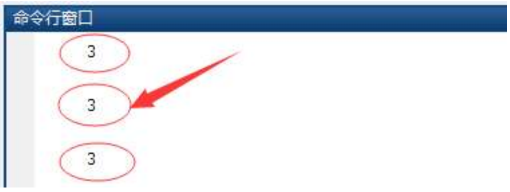
```
% close all：关闭所有的Figure窗口
% clear：清除工作空间的所有变量
% clear all：清除工作空间的所有变量，函数，和MEX文件
close all;
clear all;
clc;

%-------
% Paths. 添加文件夹
%-------
addpath('modelspecific');
addpath('mexfiles');
addpath('multigs');
```

1.2、编译Mex文件



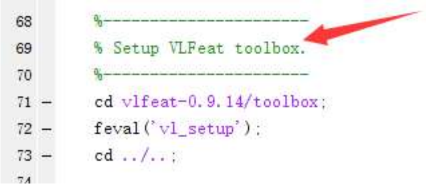
```
%-------------------
% Compile Mex files.
%-------------------
cd multigs;
% disp(exist('computeIntersection','file')) % 3
if exist('computeIntersection','file')~=3
    mex computeIntersection.c; % <-- for multigs
    disp('run');
end
cd ..;

cd mexfiles;
% disp(exist('imagewarping','file'))
if exist('imagewarping','file')~=3
    mex ../imagewarping.cpp;
    disp('run');
end
% disp(exist('wsvd','file'))
if exist('wsvd','file')~=3
    mex ../wsvd.cpp; % We make use of eigen3's SVD in this file.
    disp('run');
end
cd ..;
```

经过对代码块添加测试代码，证明了此处的代码块并未执行，在 命令行 仅仅输出了对文件的判断，而未输出if-end代码块中的测试代码。结果显示三
是3，如下图所示：



```
3
3
3
```

1.3、设置VLFeat工具箱



```
%----------------------
% Setup VLFeat toolbox.
%----------------------
cd vlfeat-0.9.14/toolbox;
feval('vl_setup');
cd ../..;
```

1.3.1 什么是VLFeat工具?

VLFeat开源库实现了很多著名的机器视觉算法，如HOG、SIFT、MSER、hierarchical、k-means, agglomerative information bottleneck、SLIC supe
quick shift，专门用于图像理解和局部特征提取和匹配。VLFeat开源库是用C语言写的，以确保其效率和兼容性，同时VLFeat还提供了MATLAB接口和
档，可以在windows, Mac, 和Linux上使用。下载地址是：http://www.vlfeat.org/

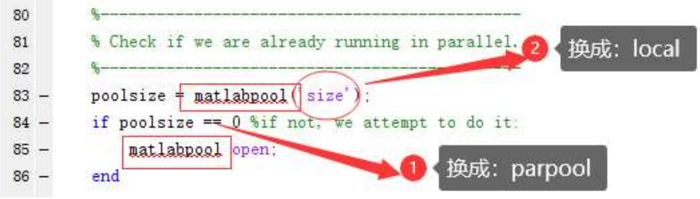
如何在MATLAB中配置VLFeat请参考文章：matlab配置vlfeat工具箱（https://blog.csdn.net/u013531940/article/details/86758215?
ops_request_misc=%257B%2522request%255Fid%2522%253A%2522163117581216780265435350%2522%252C%2522scm%2522%253A%252
.130102334.pc%255Fall.%2522%257D&request_id=163117581216780265435350&biz_id=0&utm_medium=distribute.pc_search_result.none-task-
2allfirst_rank_ecpm_v1~rank_v29_ecpm-2-
86758215.pc_search_result_control_group&utm_term=
E4%BD%9C%E7%94%A8&spm=1018.2226.3001.4187

1.4.1 matlab并行运算函数

在matlab较老的版本，如2013，2014版本中，使用的并行运算函数是：matlabpool函数

在matlab较新的版本，如2017，2018版本中，换成了：parpool函数

我在这次实验中使用的是2017版本的，需要对作者的代码进行一定的修改，如下图所示：



```
%-----------------------------------------------
% Check if we are already running in parallel.
%-----------------------------------------------
poolsize = matlabpool('size');
if poolsize == 0 %if not, we attempt to do it:
    matlabpool open;
end
```

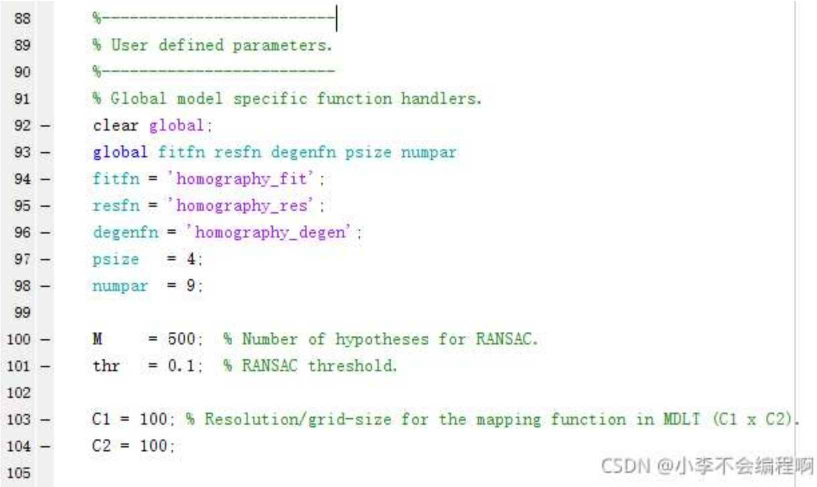
1.4.2 matlab并行原理

Matlab的并行计算实质还是主从结构的分布式计算。当你初始化Matlab并行计算环境时，你最初的Matlab进程自动成为主节点，同时初始化多个（具设定，详见下文）Matlab计算子节点。Parfor的作用就是让这些子节点同时运行Parfor语句段中的代码。Parfor运行之初，主节点会将Parfor循环程序递给计算子节点。子节点运算过程时互不干扰，运算完毕，则应该有相应代码将各子节点得到的结果组合到同一个数组变量中，并返回到Matlab主节最终计算完毕应该手动关闭计算子节点。

参考：https://blog.csdn.net/luolang_103/article/details/84870735?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522163117962216780255233923%2522%252C%2522scm%2522%253A%252.130102334…%2522%257D&request_id=163117962216780255233923&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2$_{all}$baidu_landing_v2~default-3-84870735.pc_search_result_control_group&utm_term=parpool&spm=1018.2226.3001.4187

1.5、使用定义的参数

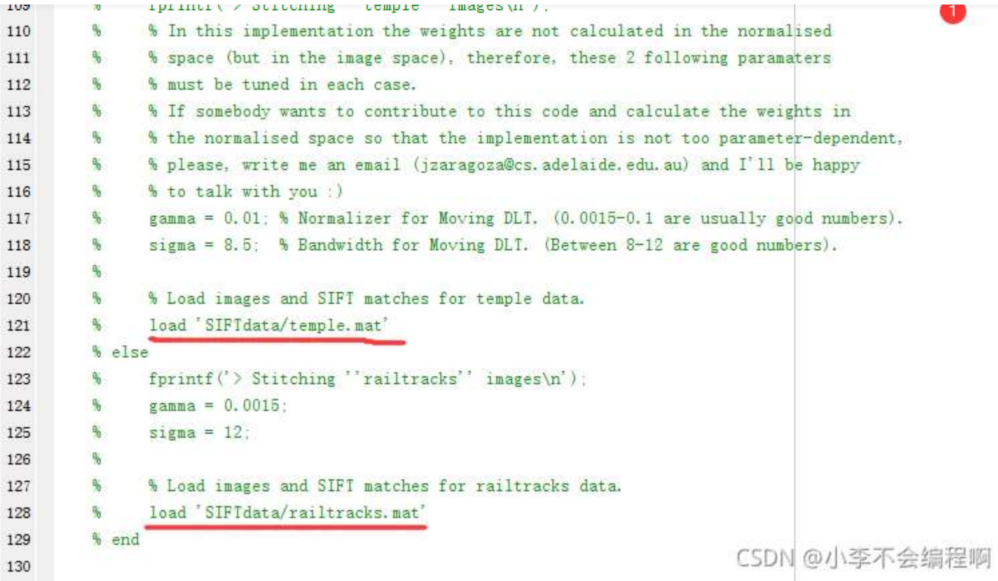
这里不做过多的解释。



```
%--------------------------
% User defined parameters.
%--------------------------
% Global model specific function handlers.
clear global;
global fitfn resfn degenfn psize numpar
fitfn = 'homography_fit';
resfn = 'homography_res';
degenfn = 'homography_degen';
psize   = 4;
numpar  = 9;

M     = 500;  % Number of hypotheses for RANSAC.
thr   = 0.1;  % RANSAC threshold.

C1 = 100; % Resolution/grid-size for the mapping function in MDLT (C1 x C2).
C2 = 100;

```

1.6、输入测试图片

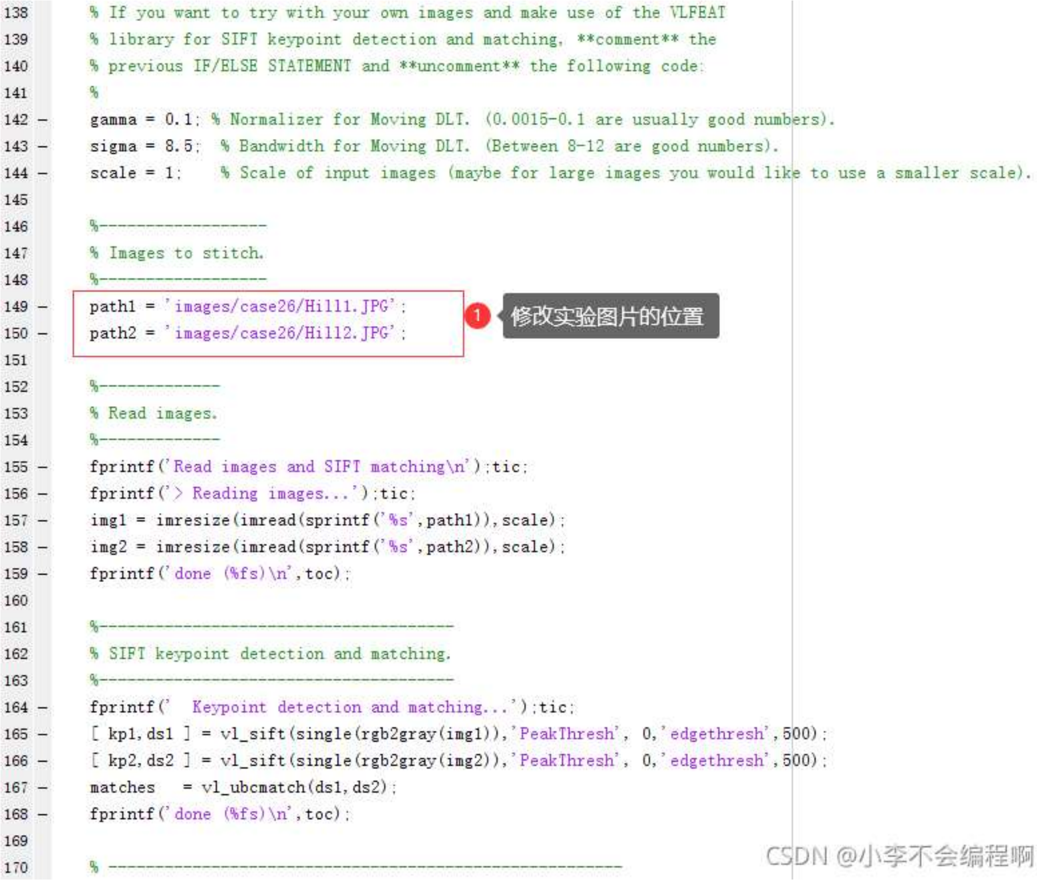
这里使用了两种方式，第一种是使用作者定义的图片，如下图所示，根据使用者输入的数字不同（如图中①所示），选用不同的图片进行实验，同时SIFTdata文件夹中的数据。



```
%     fprintf('> Stitching ''temple'' images\n');
%     % In this implementation the weights are not calculated in the normalised
%     % space (but in the image space), therefore, these 2 following paramaters
%     % must be tuned in each case.
%     % If somebody wants to contribute to this code and calculate the weights in
%     % the normalised space so that the implementation is not too parameter-dependent,
%     % please, write me an email (jzaragoza@cs.adelaide.edu.au) and I'll be happy
%     % to talk with you :)
%     gamma = 0.01; % Normalizer for Moving DLT. (0.0015-0.1 are usually good numbers).
%     sigma = 8.5;  % Bandwidth for Moving DLT. (Between 8-12 are good numbers).
%
%     % Load images and SIFT matches for temple data.
%     load 'SIFTdata/temple.mat'
% else
%     fprintf('> Stitching ''railtracks'' images\n');
%     gamma = 0.0015;
%     sigma = 12;
%
%     % Load images and SIFT matches for railtracks data.
%     load 'SIFTdata/railtracks.mat'
% end
```

CSDN @小李不会编程啊

第二种方式是实验者自己选择照片，进行实验，如下图所示。本次实验我们选用的是第二种方式，用我们自己的图片进行测试。

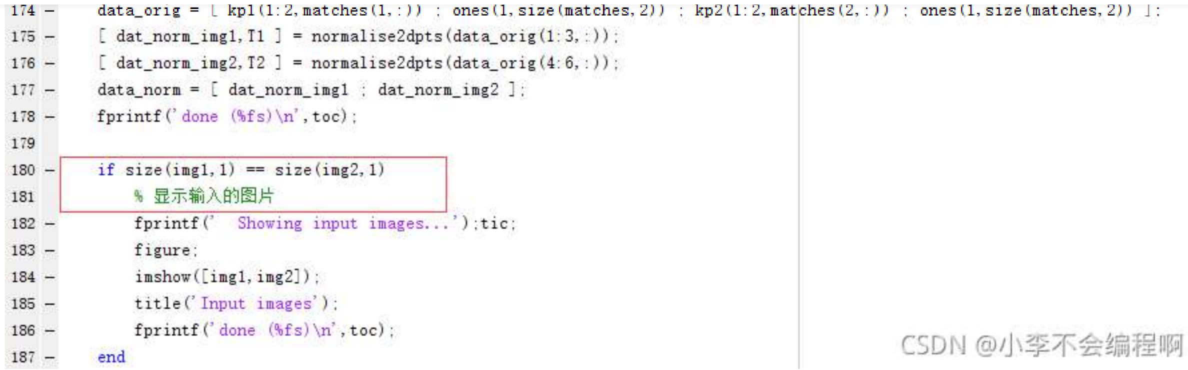


```
% If you want to try with your own images and make use of the VLFEAT
% library for SIFT keypoint detection and matching, **comment** the
% previous IF/ELSE STATEMENT and **uncomment** the following code:
%
gamma = 0.1; % Normalizer for Moving DLT. (0.0015-0.1 are usually good numbers).
sigma = 8.5;  % Bandwidth for Moving DLT. (Between 8-12 are good numbers).
scale = 1;    % Scale of input images (maybe for large images you would like to use a smaller scale).

%------------------
% Images to stitch.
%------------------
path1 = 'images/case26/Hill1.JPG';
path2 = 'images/case26/Hill2.JPG';

%-------------
% Read images.
%-------------
fprintf('Read images and SIFT matching\n');tic;
fprintf('> Reading images...');tic;
img1 = imresize(imread(sprintf('%s',path1)),scale);
img2 = imresize(imread(sprintf('%s',path2)),scale);
fprintf('done (%fs)\n',toc);

%--------------------------------------
% SIFT keypoint detection and matching.
%--------------------------------------
fprintf('  Keypoint detection and matching...');tic;
[ kp1,ds1 ] = vl_sift(single(rgb2gray(img1)),'PeakThresh', 0,'edgethresh',500);
[ kp2,ds2 ] = vl_sift(single(rgb2gray(img2)),'PeakThresh', 0,'edgethresh',500);
matches   = vl_ubcmatch(ds1,ds2);
fprintf('done (%fs)\n',toc);

% -------------------------------------------------------
```

1 修改实验图片的位置

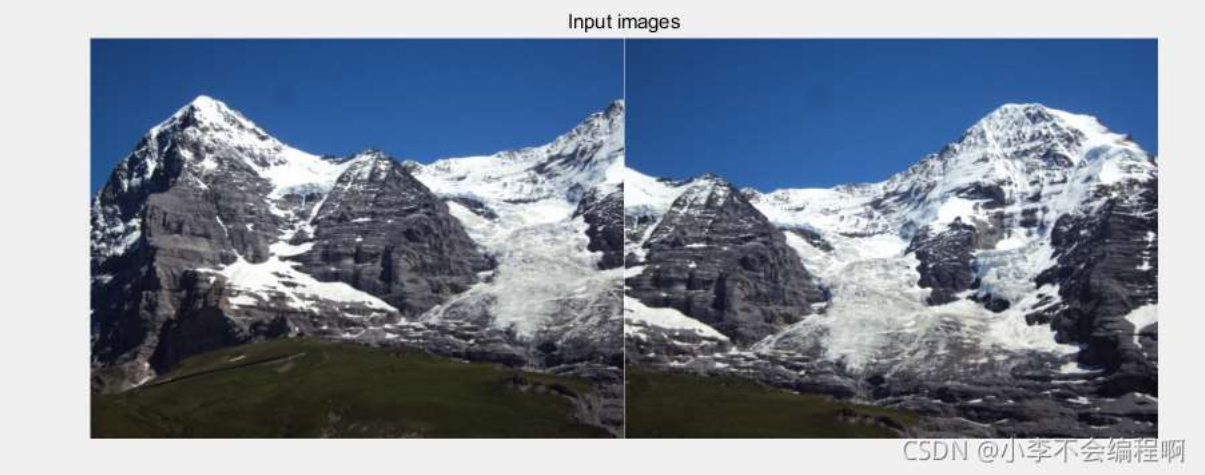
CSDN @小李不会编程啊

1.7、正常化点分布及显示图片

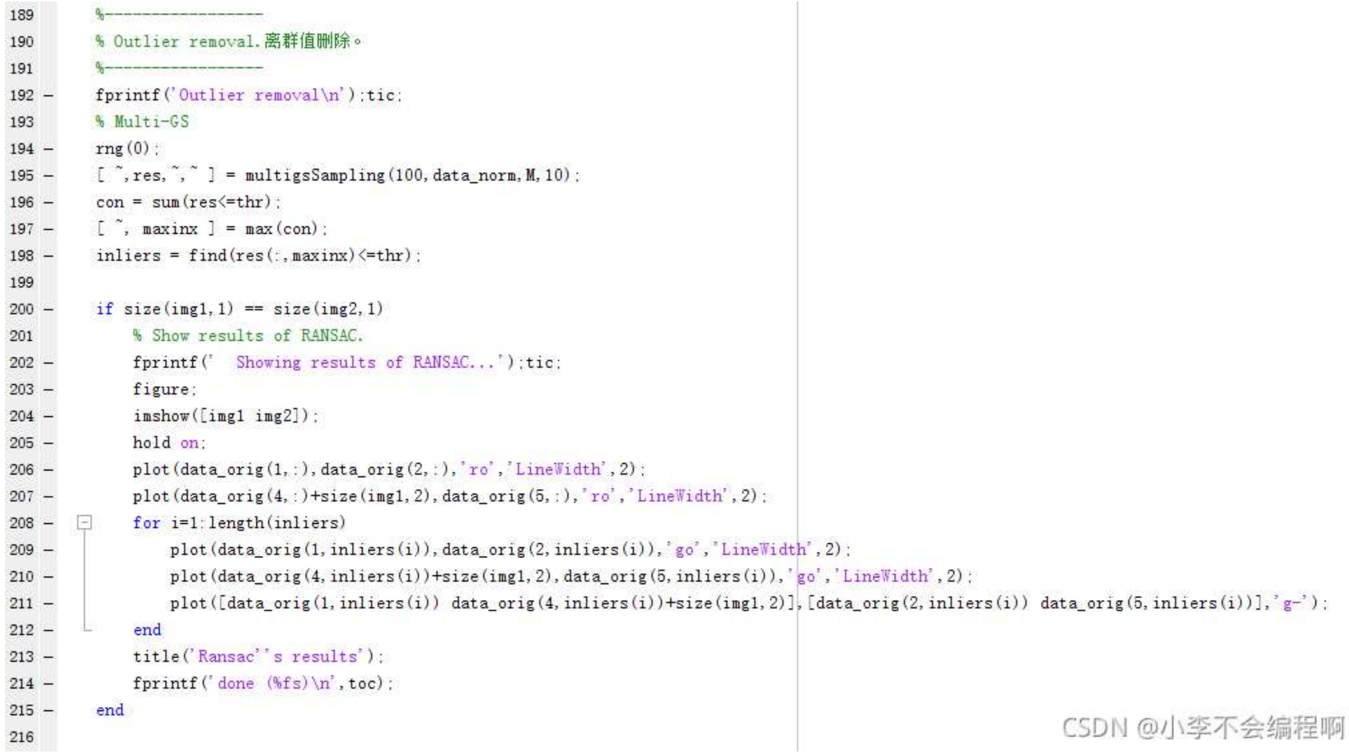


```
    data_orig = [ kp1(1:2,matches(1,:)) ; ones(1,size(matches,2)) ; kp2(1:2,matches(2,:)) ; ones(1,size(matches,2)) ];
    [ dat_norm_img1,T1 ] = normalise2dpts(data_orig(1:3,:));
    [ dat_norm_img2,T2 ] = normalise2dpts(data_orig(4:6,:));
    data_norm = [ dat_norm_img1 ; dat_norm_img2 ];
    fprintf('done (%fs)\n',toc);

    if size(img1,1) == size(img2,1)
        % 显示输入的图片
        fprintf('  Showing input images...');tic;
        figure;
        imshow([img1,img2]);
        title('Input images');
        fprintf('done (%fs)\n',toc);
    end
```

显示的输入图片：

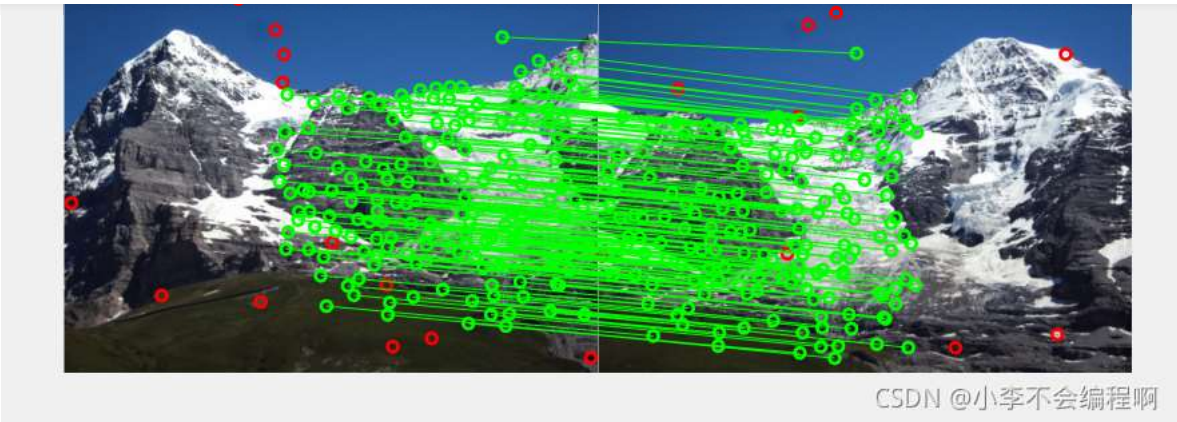

Input images

1.8、删除离群点，同时利用RANSAC算法得到匹配图

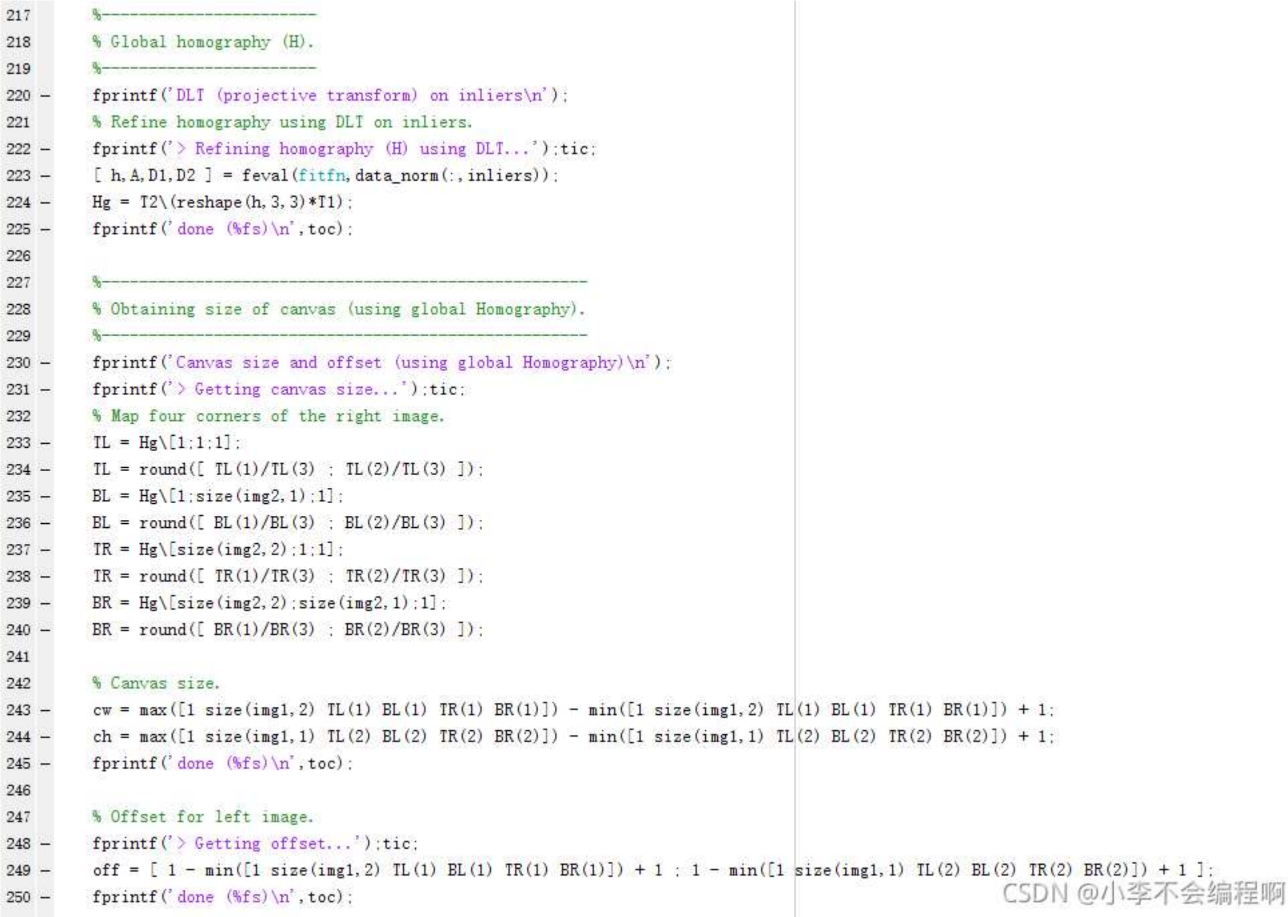


```
    %-------------------
    % Outlier removal.离群值删除。
    %-------------------
    fprintf('Outlier removal\n');tic;
    % Multi-GS
    rng(0);
    [ ~,res,~,~ ] = multigsSampling(100,data_norm,M,10);
    con = sum(res<=thr);
    [ ~, maxinx ] = max(con);
    inliers = find(res(:,maxinx)<=thr);

    if size(img1,1) == size(img2,1)
        % Show results of RANSAC.
        fprintf('  Showing results of RANSAC...');tic;
        figure;
        imshow([img1 img2]);
        hold on;
        plot(data_orig(1,:),data_orig(2,:),'ro','LineWidth',2);
        plot(data_orig(4,:)+size(img1,2),data_orig(5,:),'ro','LineWidth',2);
        for i=1:length(inliers)
            plot(data_orig(1,inliers(i)),data_orig(2,inliers(i)),'go','LineWidth',2);
            plot(data_orig(4,inliers(i))+size(img1,2),data_orig(5,inliers(i)),'go','LineWidth',2);
            plot([data_orig(1,inliers(i)) data_orig(4,inliers(i))+size(img1,2)],[data_orig(2,inliers(i)) data_orig(5,inliers(i))],'g-');
        end
        title('Ransac''s results');
        fprintf('done (%fs)\n',toc);
    end

```

匹配点结果图：



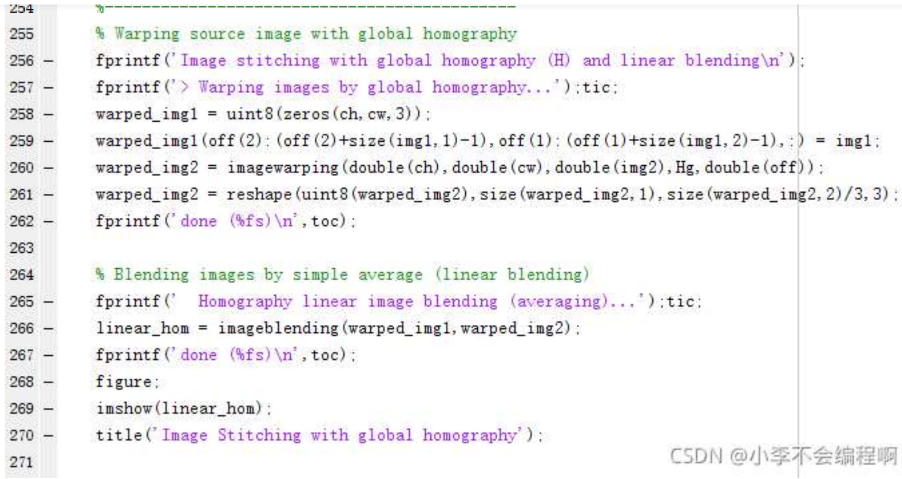
1.9、求全局单应性，DLT



```
%-------------------------
% Global homography (H).
%-------------------------
fprintf('DLT (projective transform) on inliers\n');
% Refine homography using DLT on inliers.
fprintf('> Refining homography (H) using DLT...');tic;
[ h,A,D1,D2 ] = feval(fitfn,data_norm(:,inliers));
Hg = T2\(reshape(h,3,3)*T1);
fprintf('done (%fs)\n',toc);

%-------------------------------------------------------------
% Obtaining size of canvas (using global Homography).
%-------------------------------------------------------------
fprintf('Canvas size and offset (using global Homography)\n');
fprintf('> Getting canvas size...');tic;
% Map four corners of the right image.
TL = Hg\[1;1;1];
TL = round([ TL(1)/TL(3) ; TL(2)/TL(3) ]);
BL = Hg\[1;size(img2,1);1];
BL = round([ BL(1)/BL(3) ; BL(2)/BL(3) ]);
TR = Hg\[size(img2,2);1;1];
TR = round([ TR(1)/TR(3) ; TR(2)/TR(3) ]);
BR = Hg\[size(img2,2);size(img2,1);1];
BR = round([ BR(1)/BR(3) ; BR(2)/BR(3) ]);

% Canvas size.
cw = max([1 size(img1,2) TL(1) BL(1) TR(1) BR(1)]) - min([1 size(img1,2) TL(1) BL(1) TR(1) BR(1)]) + 1;
ch = max([1 size(img1,1) TL(2) BL(2) TR(2) BR(2)]) - min([1 size(img1,1) TL(2) BL(2) TR(2) BR(2)]) + 1;
fprintf('done (%fs)\n',toc);

% Offset for left image.
fprintf('> Getting offset...');tic;
off = [ 1 - min([1 size(img1,2) TL(1) BL(1) TR(1) BR(1)]) + 1 ; 1 - min([1 size(img1,1) TL(2) BL(2) TR(2) BR(2)]) + 1 ];
fprintf('done (%fs)\n',toc);
```

具体的分析步骤见参考1。

1.10、使用单应矩阵拼接图像



```
%------------------------------------------
% Warping source image with global homography
fprintf('Image stitching with global homography (H) and linear blending\n');
fprintf('> Warping images by global homography...');tic;
warped_img1 = uint8(zeros(ch,cw,3));
warped_img1(off(2):(off(2)+size(img1,1)-1),off(1):(off(1)+size(img1,2)-1),:) = img1;
warped_img2 = imagewarping(double(ch),double(cw),double(img2),Hg,double(off));
warped_img2 = reshape(uint8(warped_img2),size(warped_img2,1),size(warped_img2,2)/3,3);
fprintf('done (%fs)\n',toc);

% Blending images by simple average (linear blending)
fprintf('  Homography linear image blending (averaging)...');tic;
linear_hom = imageblending(warped_img1,warped_img2);
fprintf('done (%fs)\n',toc);
figure;
imshow(linear_hom);
title('Image Stitching with global homography');
```

拼接的结果：

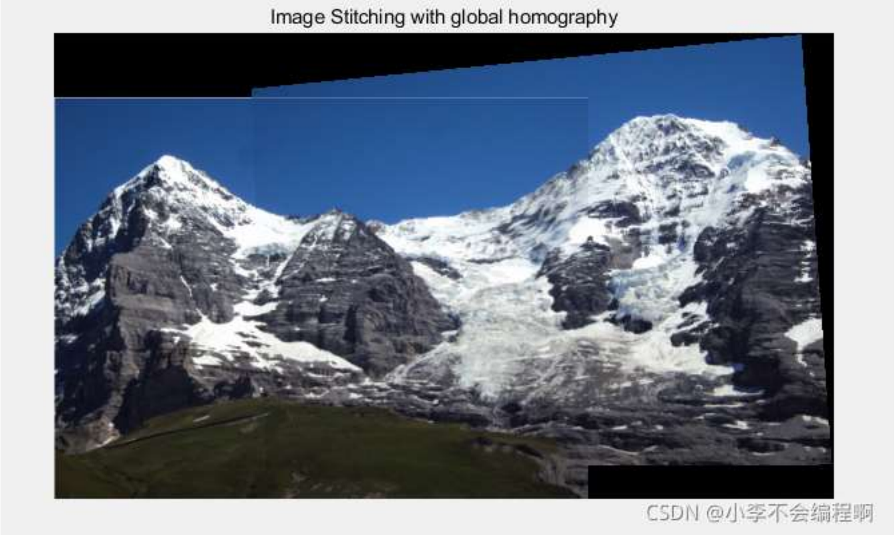
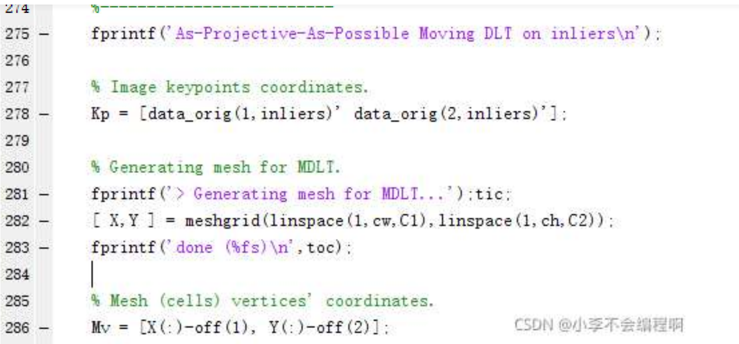
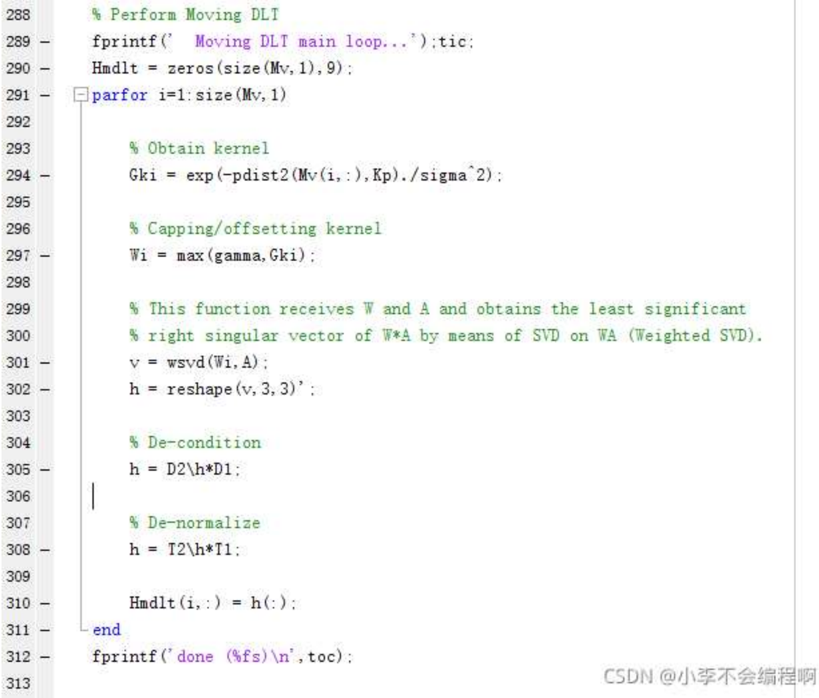


Image Stitching with global homography

1.11、APAP,Moving DLT

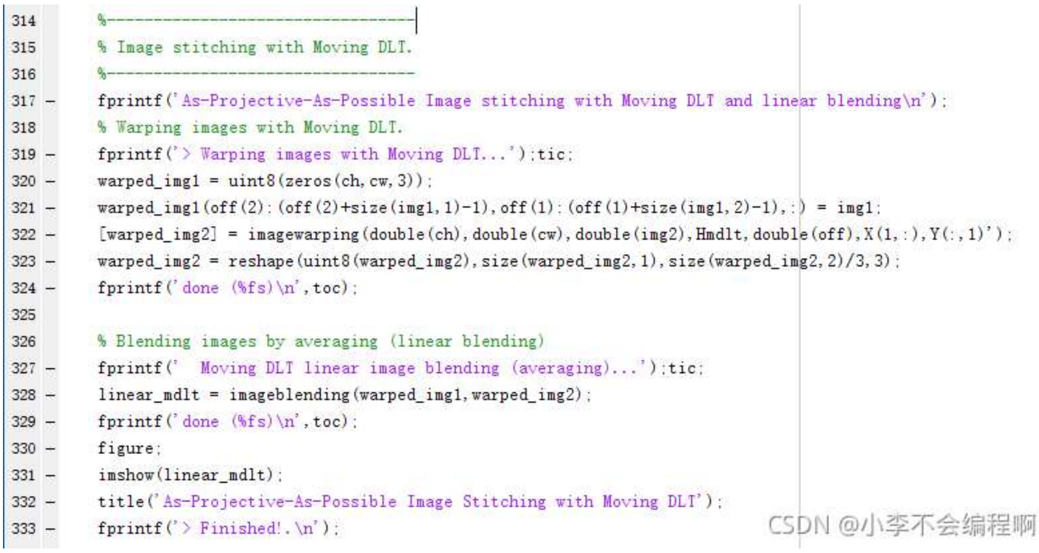


```matlab
%--------------------------
fprintf('As-Projective-As-Possible Moving DLT on inliers\n');

% Image keypoints coordinates.
Kp = [data_orig(1,inliers)' data_orig(2,inliers)'];

% Generating mesh for MDLT.
fprintf('> Generating mesh for MDLT...');tic;
[ X,Y ] = meshgrid(linspace(1,cw,C1),linspace(1,ch,C2));
fprintf('done (%fs)\n',toc);

% Mesh (cells) vertices' coordinates.
Mv = [X(:)-off(1), Y(:)-off(2)];
```



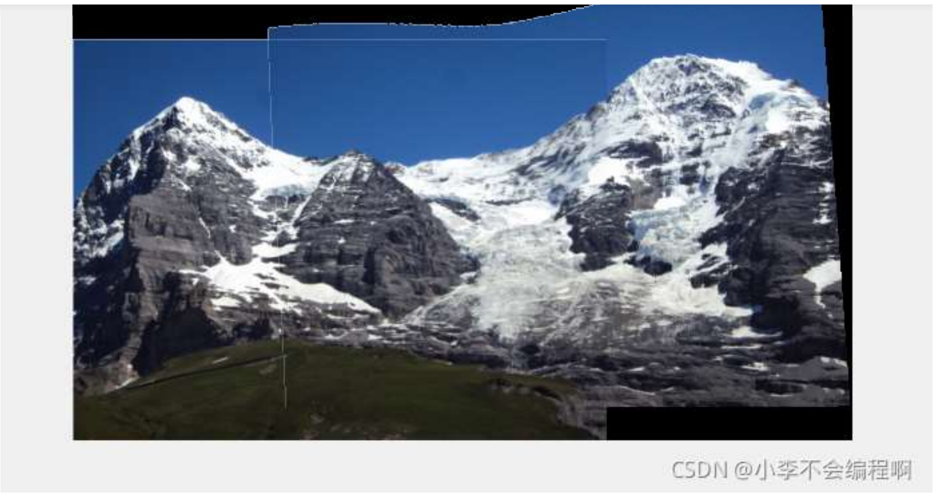
```matlab
% Perform Moving DLT
fprintf('  Moving DLT main loop...');tic;
Hmdlt = zeros(size(Mv,1),9);
parfor i=1:size(Mv,1)

    % Obtain kernel
    Gki = exp(-pdist2(Mv(i,:),Kp)./sigma^2);

    % Capping/offsetting kernel
    Wi = max(gamma,Gki);

    % This function receives W and A and obtains the least significant
    % right singular vector of W*A by means of SVD on WA (Weighted SVD).
    v = wsvd(Wi,A);
    h = reshape(v,3,3)';

    % De-condition
    h = D2\h*D1;

    % De-normalize
    h = T2\h*T1;

    Hmdlt(i,:) = h(:);
end
fprintf('done (%fs)\n',toc);
```

1.12、使用DLT拼接图像



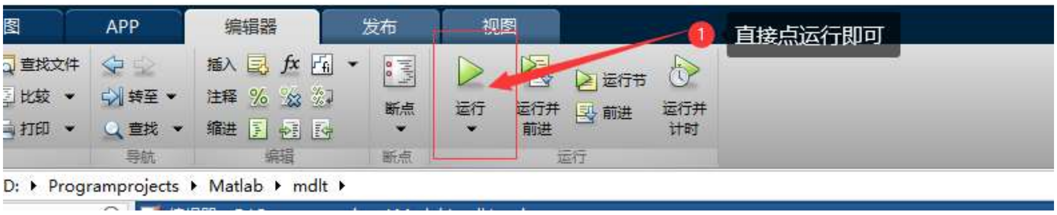
```matlab
%-----------------------------------
% Image stitching with Moving DLT.
%-----------------------------------
fprintf('As-Projective-As-Possible Image stitching with Moving DLT and linear blending\n');
% Warping images with Moving DLT.
fprintf('> Warping images with Moving DLT...');tic;
warped_img1 = uint8(zeros(ch,cw,3));
warped_img1(off(2):(off(2)+size(img1,1)-1),off(1):(off(1)+size(img1,2)-1),:) = img1;
[warped_img2] = imagewarping(double(ch),double(cw),double(img2),Hmdlt,double(off),X(1,:),Y(:,1)');
warped_img2 = reshape(uint8(warped_img2),size(warped_img2,1),size(warped_img2,2)/3,3);
fprintf('done (%fs)\n',toc);

% Blending images by averaging (linear blending)
fprintf('  Moving DLT linear image blending (averaging)...');tic;
linear_mdlt = imageblending(warped_img1,warped_img2);
fprintf('done (%fs)\n',toc);
figure;
imshow(linear_mdlt);
title('As-Projective-As-Possible Image Stitching with Moving DLT');
fprintf('> Finished!.\n');
```

拼接的结果：

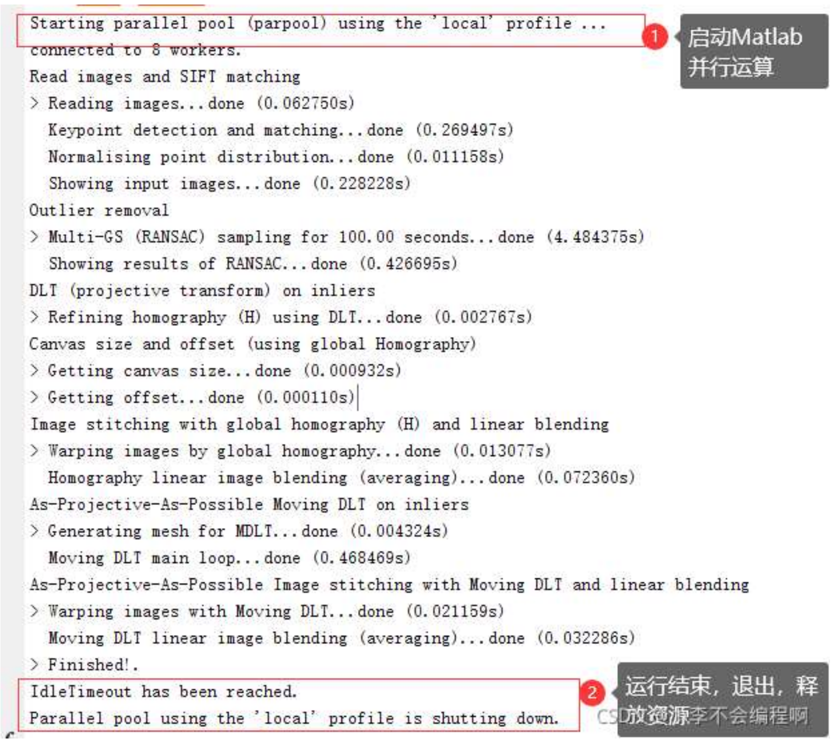


二、运行程序

找到main.m文件，将其添加到当前工作目录下，点运行即可



1 直接点运行即可



```
Starting parallel pool (parpool) using the 'local' profile ...
connected to 8 workers.
Read images and SIFT matching
> Reading images...done (0.062750s)
  Keypoint detection and matching...done (0.269497s)
  Normalising point distribution...done (0.011158s)
  Showing input images...done (0.228228s)
Outlier removal
> Multi-GS (RANSAC) sampling for 100.00 seconds...done (4.484375s)
  Showing results of RANSAC...done (0.426695s)
DLT (projective transform) on inliers
> Refining homography (H) using DLT...done (0.002767s)
Canvas size and offset (using global Homography)
> Getting canvas size...done (0.000932s)
> Getting offset...done (0.000110s)
Image stitching with global homography (H) and linear blending
> Warping images by global homography...done (0.013077s)
  Homography linear image blending (averaging)...done (0.072360s)
As-Projective-As-Possible Moving DLT on inliers
> Generating mesh for MDLT...done (0.004324s)
  Moving DLT main loop...done (0.468469s)
As-Projective-As-Possible Image stitching with Moving DLT and linear blending
> Warping images with Moving DLT...done (0.021159s)
  Moving DLT linear image blending (averaging)...done (0.032286s)
> Finished!.
IdleTimeout has been reached.
Parallel pool using the 'local' profile is shutting down.
```

1 启动Matlab并行运算

2 运行结束，退出，释放资源

三、参考：

1、https://blog.csdn.net/chentianting/article/details/88869872

2、https://blog.csdn.net/qq_32095699/article/details/80448170?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522163115436616780271522172%2522%252C%2522scm%2522%253A%252.130102334...%2522%257D&request_id=163115436616780271522172&biz_id=0&utm_medium=distribute.pc_search_result.none-task-blog-2~all~sobaiduend~default-3-80448170.pc_search_result_control_group&utm_term=APAP&spm=1018.2226.3001.4187

3、 https://blog.csdn.net/qq_32095699/article/details/80448170?ops_request_misc=%257B%2522request%255Fid%2522%253A%2522163115436616780271522172%2522%252C%2522scm%2522%253A%252.130102334...%2522%257D&request_id=16311543661
2~all~sobaiduend~default-3-80448170.pc_search_result_c