Exercise set: 5 **Student:** Tommaso Carraro **Student number:** 015147592

Exercise set 5 Computer Vision

Question 5.1

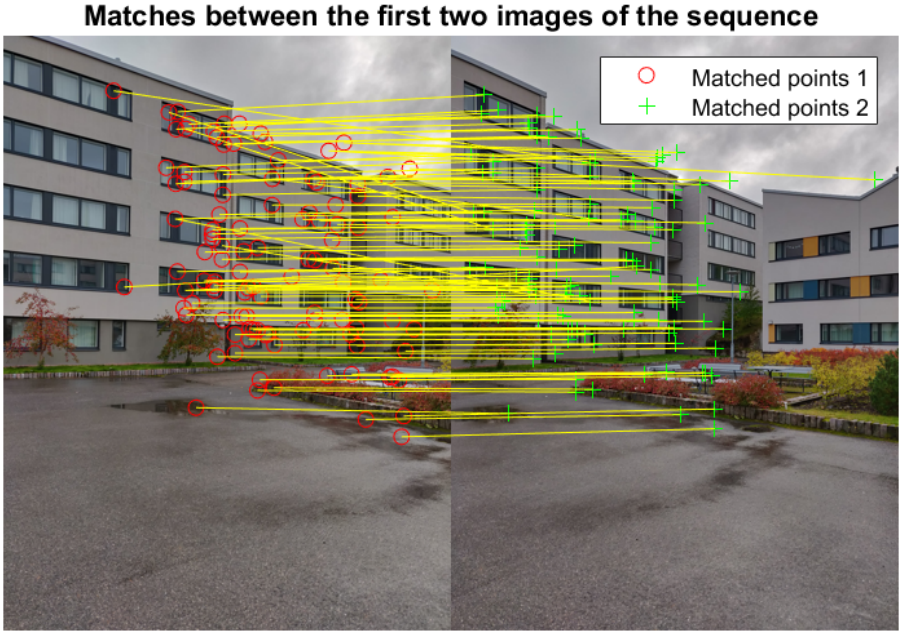
For this exercise I took eight overlapping photos of my dormitory by only rotating the camera to the right. In the following image I show the photos I have taken.

From this point I will explain my method for the first two images in the sequence because it is the same for the all sequence.

Having the first two images, we want to stitch them using a feature based image stitching, so the first thing it is necessary to do is to extract the features of the two consecutive images and then matching them.

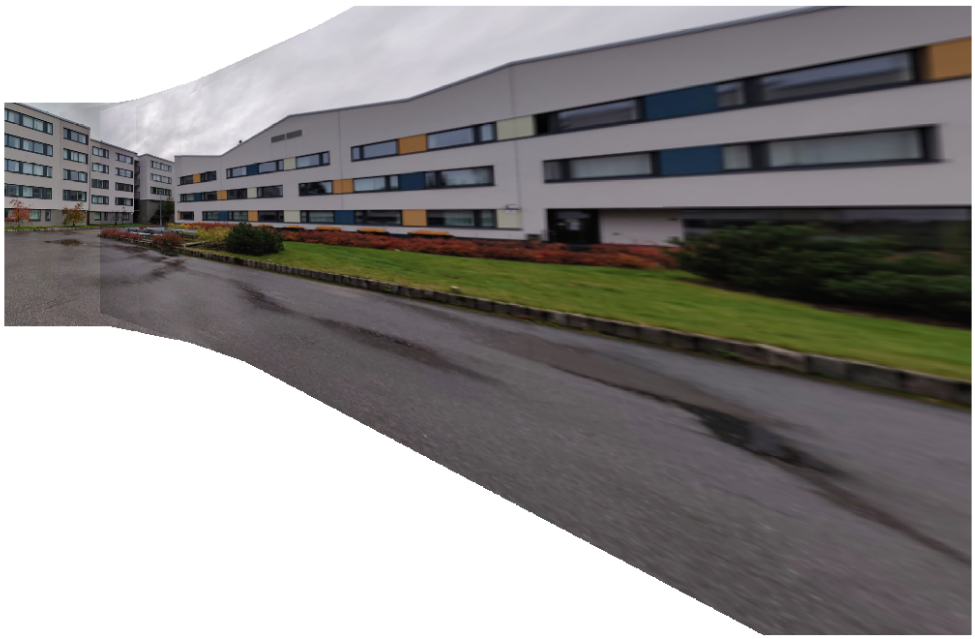
To find the key points in the images I've detected the SURF features. In the following image it is possible to see the best 40 SURF features in the first two images of the sequence.

After feature detection it is possible to match the features. In the following image I show the matched features between the first two images of the sequence.



After that it is possible to compute the homography between the two images. The homography is the projective transformation that has to be applied to the second image to be able to properly align it to the first image. It is possible to estimate the homography using RANSAC, a method we have seen to exclude the outliers from the estimation. Finally, having the homography transformation matrix we have to warp the images and composite them to make a panorama. In the following image I show the result of my algorithm for the first two images of the sequence.

In the following image I show an issue I got for the all sequence.



This issue is related to the fact that all the transformations I have computed are relative to the first image. Using the first image as the start of the panorama does not produce a pleasing panorama because it tends to distort most of the images that form the panorama, like in the image provided. A nicer panorama can be created by modifying the transformations such that the center of the scene is the least distorted. This is accomplished by inverting the transform for the center image and applying that transform to all the others (solution found online). In the following image I show the final result of this exercise.

The MATLAB code to reproduce my results is shown below.

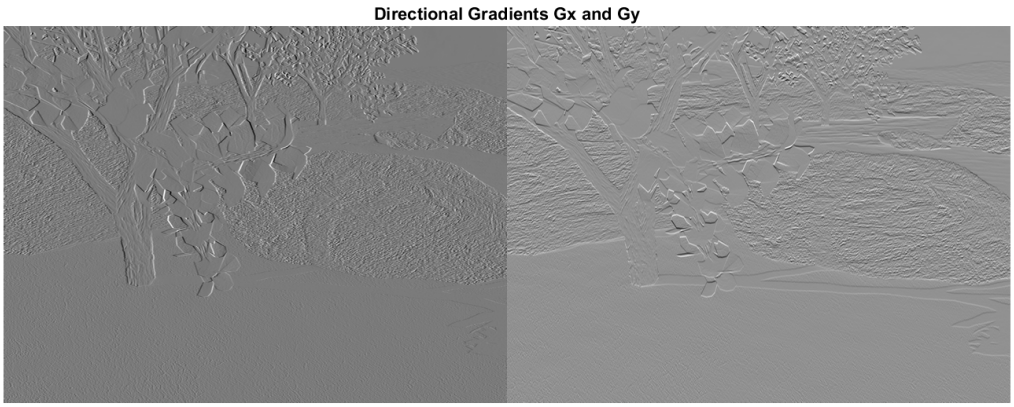
```
% this is the directory containing the images
buildingDir = fullfile('C:','Users','tomma','Desktop',
'uni','Helsinki','corsi','computerVision','esercizi',
'5','panorama','img');
buildingScene = imageDatastore(buildingDir);
% Read the first image from the image set.
I = imresize(imrotate(readimage(buildingScene, 1),
-90,'bilinear'),0.1);
% Initialize features for the first image
grayImage = rgb2gray(I);
points = detectSURFFeatures(grayImage);
[features, points] = extractFeatures(grayImage, points);
numImages = numel(buildingScene.Files);
% set a standard projective transformation for the first image
tforms(1) = projective2d(eye(3));

% Iterate over remaining image pairs
for i = 2:numImages
    % Store points and features for I(n-1).
    pointsPrevious = points;
    featuresPrevious = features;
    % Read I(n).
    I = imresize(imrotate(readimage(buildingScene, i),
    -90,'bilinear'),0.1);
    % Convert image to grayscale.
    grayImage = rgb2gray(I);
    % Detect and extract SURF features for I(n).
    points = detectSURFFeatures(grayImage);
    [features, points] = extractFeatures(grayImage, points);
    % Find correspondences between I(n) and I(n-1).
    indexPairs = matchFeatures(features, featuresPrevious);
    matchedPoints = points(indexPairs(:,1), :);
    matchedPointsPrev = pointsPrevious(indexPairs(:,2), :);
    % Estimate the transformation between I(n) and I(n-1).
```

```
    tforms(i) = estimateGeometricTransform(matchedPoints,
    matchedPointsPrev,'projective');
    tforms(i).T = tforms(i).T * tforms(i-1).T;
end

% making the center image of the scene the least distorted
Tinv = invert(tforms(4));
for i = 1:size(tforms, 2)
    tforms(i).T = tforms(i).T * Tinv.T;
end

% Prepare to warp the images
[w, h]     = deal(2000, 1300);  % Size of the mosaic
[x0, y0]   = deal(-700, -300);   % Upper-left corner of the mosaic
xLim = [0.5, w+0.5] + x0;
yLim = [0.5, h+0.5] + y0;
outputView = imref2d([h,w], xLim, yLim);
mosaic = ones(h, w, 3, 'uint8')*255;

% blender definition
halphablender = vision.AlphaBlender('Operation', 'Binary mask',
'MaskSource', 'Input port');

% Create the panorama.
for i = 1:numImages
    I = imresize(imrotate(readimage(buildingScene, i),
    -90,'bilinear'),0.1);
    % Transform I into the panorama.
    warpedImage = imwarp(I, tforms(i), 'OutputView', outputView);
    % Generate a binary mask.
    mask = imwarp(true(size(I,1),size(I,2)), tforms(i),
    'OutputView', outputView);
    % Overlay the warpedImage onto the panorama.
    mosaic = step(halphablender, mosaic, warpedImage, mask);
end

figure;
imshow(mosaic);
```

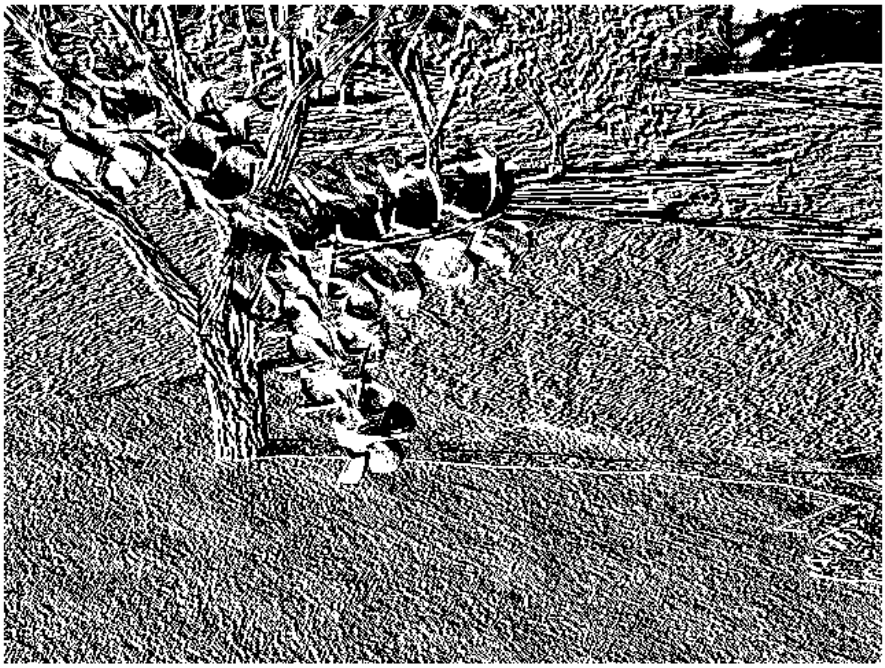
Exercise 5.2

For this exercise I have firstly computed the vertical and horizontal gradients of the first image and the temporal gradient between the two images using the frame differencing method. After that I decided to use a 5x5 patch to compute the optical flow. I moved this patch around the all image like a filter and I computed the flow velocities related to each of these patches. For each patch I made a linear system of 25 equations with two unknowns as presented in the slides and I computed the flow velocities solving the system with the least squares method.

In the following image I show the horizontal (on the left) and the vertical (on the right) gradients related to the first image.



In the following image I show the temporal gradient between the two images.



Finally, my MATLAB implementation for the optical flow is provided below.

```
I1 = rgb2gray(imread('frame10.png'));
I2 = rgb2gray(imread('frame11.png'));
% compute gradients in x and y direction at time t
% using central filter to avoid image shifting
[Gx,Gy] = imgradientxy(I1, 'central');
% compute temporal gradient between time t and t + 1
% using frame differences
Gt = I1 - I2;
Gt = double(Gt);
% move a 5x5 patch in the image and solve a 25 linear
% equations system for each patch using least square method
a1 = 0.0;
a2 = 0.0;
a3 = 0.0;
a4 = 0.0;
a5 = 0.0;
```

```
a6 = 0.0;
% variables to store the flow velocities
u = [];
v = [];
for x=1:size(I1,1)
    for y=1:size(I1,2)
        if x+4 <= size(I1,1) && y+4 <= size(I1,2)
            for i=x:x+4
                for j=y:y+4
                    a1 = a1 + Gx(i,j)^2;
                    a2 = a2 + Gx(i,j)Gy(i,j);
                    a3 = a3 + Gy(i,j)Gx(i,j);
                    a4 = a4 + Gy(i,j)^2;
                    a5 = a5 + Gx(i,j)Gt(i,j);
                    a6 = a6 + Gy(i,j)Gt(i,j);
                end
            end
            a5 = -a5;
            a6 = -a6;
            A = [a1 a2; a3 a4];
            b = [a5; a6];
            % solving the system
            uv = A\b;
            u(end+1) = uv(1);
            v(end+1) = uv(2);
        end
    end
end
```