Exercise 1 - Features

Computer Vision

Alberto Montes (malberto@student.ethz.ch) October 6, 2016

1 Harris Corner Detector

The implementation of the Harris Corner Detector starts with the computation of the image's gradient. This step presented a problem as it was computed using a convolution, because corner detection appeared at the margins of the image. This was solved padding symmetrically before convolving as is shown in Listing 1

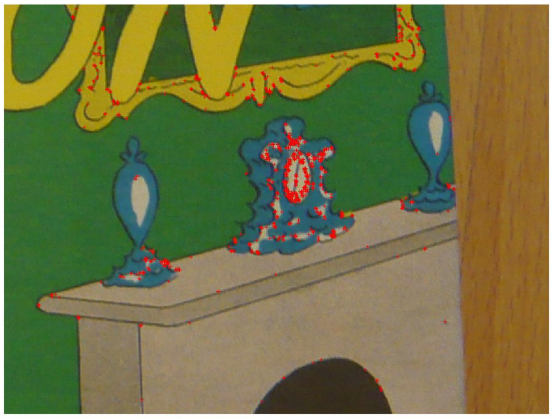
```
% Define the gradients steps
dx = [-.5 0 .5];
dy = dx';
% Compute the gradient of the image at each axis
IX = conv2(padarray(img, [0, 1], 'symmetric'), dx, 'valid');
IY = conv2(padarray(img, [1, 0], 'symmetric'), dy, 'valid');
```

Listing 1: extractHarrisCorner.m

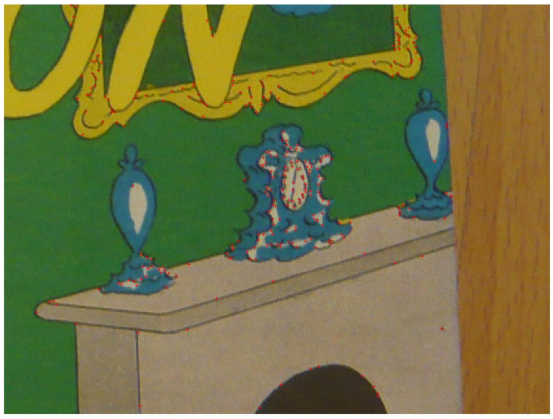
Then each component of the Harris matrix was computed with this image gradient and a gaussian filter to finally compute the Harris response mesure as:

$$K = \frac{det(H)}{trace(H)} = \frac{I_x^2 \cdot I_y^2 - (I_x I_y)^2}{I_x^2 + I_y^2} \tag{1}$$

Once the Harris response mesure has been computed, a Non-Maximum-Suppression operation must be performed in order to obtain corners too close each one to the others. This can be easily see on the following figure where the threshold was set to $3 \cdot 10^{-4}$ and it compares the use or not of the Non-Maximum-Suppression operation.



(a) Before applying Non-Maximal-Suppression



(b) After applying Non-Maximal-Suppression

Finally the global images applying the Harris Corner Detector with a Non-Maximum-Suppression of 3 pixels of radius are displayed on Figure 2

Figure 2: Corners detected with the implementation of the Harris Corner Detector

2 Feature Matching

After computing the corners, the SSD matching has been implemented. This did not present any major issue except the threshold to set. Finally, this threshold was set to 0.8 to get the maximum of good matching and not so many wrong ones. In the Figure 3 can be seen the matching done using the Harris features extracted previously.

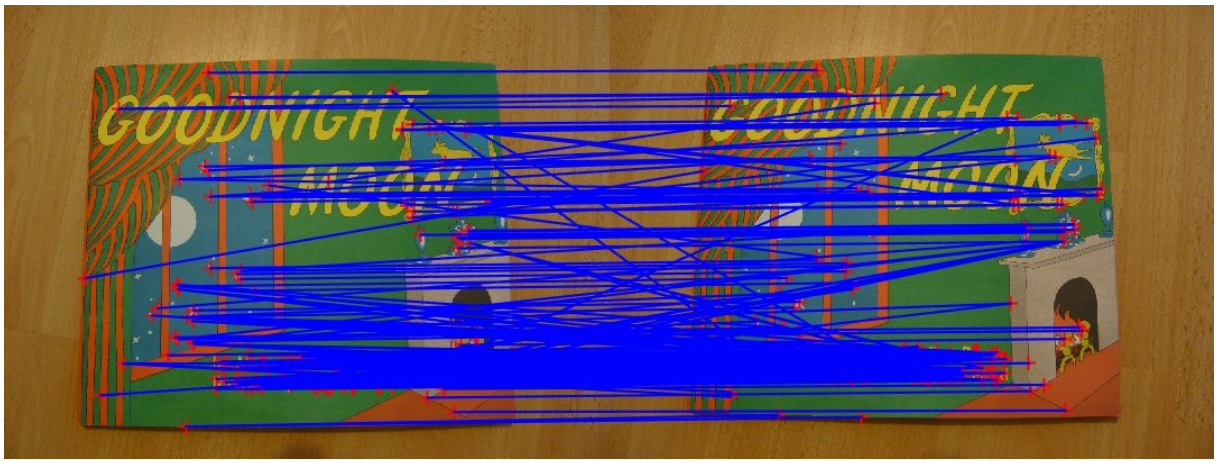


Figure 3: Matching of the Harris Corner features using the SSD.

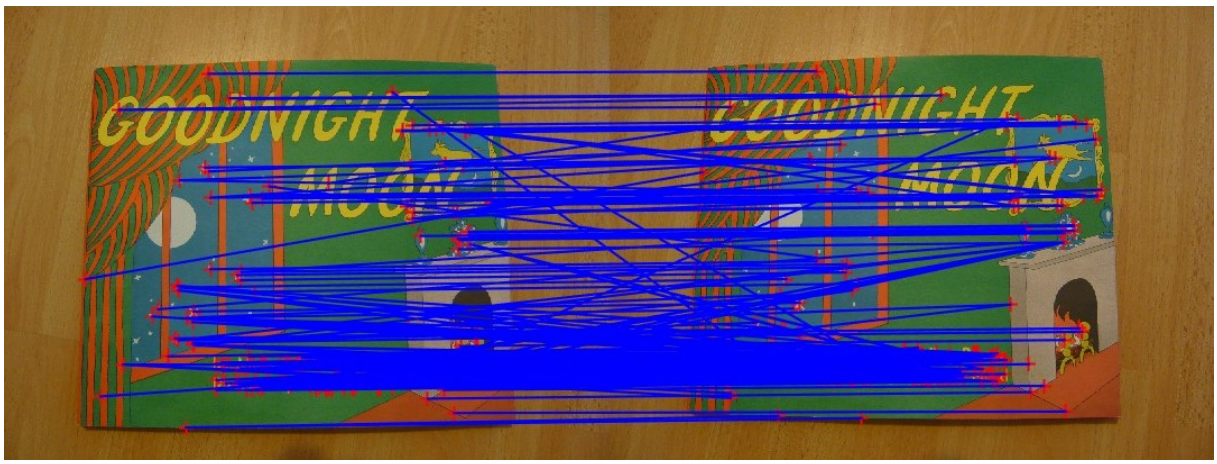
3 SIFT features

Using the library VLfeat provided, it has been possible to extract the SIFT features and compare the results with the Harris features computed with my implementation. In the following figure it is shown the Harris corner features and the SIFT features for the same image. The SIFT features was so large that it is shown only 500 selected randomly from the ones provided by the VLfeat implementation.

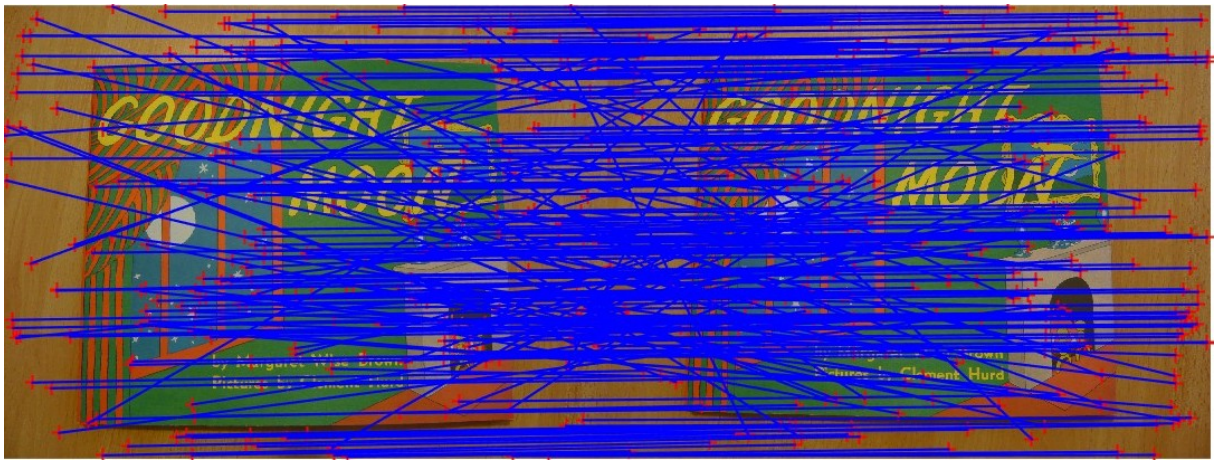
(a) Harris Corner Features (b) SIFT Features

Figure 4: Comparison between Harris and SIFT features.

As can be seen on Figure 3, the Harris Corner detector find the corners of the image and activate its detections on edges of the image. At the same time, the SIFT features seem to be more distributed along the images, activating even on flat textures, which gives more information at the time to do the matching.



(a) Harris Corner Features Matching



(b) SIFT Features Matching

Figure 5: Comparison between Harris and SIFT features matching.

In Figure 5 can be seen how there is more matchings using the SIFT features as there are more of

them even on more plain textures, where the Harris features matches the ones corresponding to more sharp textures as corners and edges.