Cs512- Computer Vision

Kinjal Kachi

Hawk id: A20449343

Abstract

The aim of this assignment coding question is to let students understand the insights of Corner Detection, Localization and Feature Matching and Implementation of Harris Corner Detection Algorithm from scratch.

Problem Statement:

1.1. Load and display two images containing similar content. In each image perform:
- Estimate image gradients and apply the Harris corner detection algorithm.
- Obtain a better localization of each corner.
- Compute a feature vector for each corner point.
- Display the corners by drawing empty rectangles over the original image centered at locations where corners where detected.

1.2. Using the feature vectors, you computed match the feature points. Number corresponding points with identical numbers in the two images.

1.3. Interactively controlled parameters should include: the variance of the Gaussian (scale), the neighborhood size for computing the correlation matrix, the weight of the trace in the Harris corner detector and a threshold value.

2. Proposed solution

2.1. Harris Córner Deteción Proposed Solution:

Use the Below Formula for Implementation:

Measure of corner response:

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(k is an empirically determined constant; $k = 0.04 - 0.06$)

2.2. Feature Vector Calculation and Feature Matching.

Basics of Brute-Force Matcher
Brute-Force matcher is simple. It takes the descriptor of one feature in first set and is matched with all other features in second set using some distance calculation. And the closest one is returned.

2.3. Feature Localization:

Formula: $P = C^{-1} \sum \nabla I(x_i) \; \nabla I(x_i)^T x_i$

C – correlation matrix

$\nabla I(x_i)$ – gradients of x_i

Given that there is a corner in a window find it's location. To determine if P is the corner connect each point xi to P and project the gradient at xi into (Xi – P)
The best P will minimize the sum of all the projections.

3. Implementation details

Harris Corner Detection Implementation:

1. Read the Image.
2. Find the height and weight of the image to iterate over the image.
3. Calculate the elements of the correlation Matrix:
   Sx2, Sy2, Sxy
   a. Compute the derivate ox and y of an image.(Ixx, Iyy, Ixy)
   b. Compute product of Derivative at each pixel. (Sx1, Sy1, Sxy1)
   c. Compute the sum of the product of the derivate at each pixel. (Sx2, Sx2, Sxy)
4. Calculation the Deteminant Det(C) and the Trace trace(C) of the Correlation Matrix
5. Calculate the Response using the Formula: Det(C) - k*(Trace trace(C))**2
   Where k = coefficient of trace. (range: 0.04 to 0.15)
6. Compare if the Value of the Response is greater than the user defined threshold. If Response > threshold then corner detected and being highlighted,

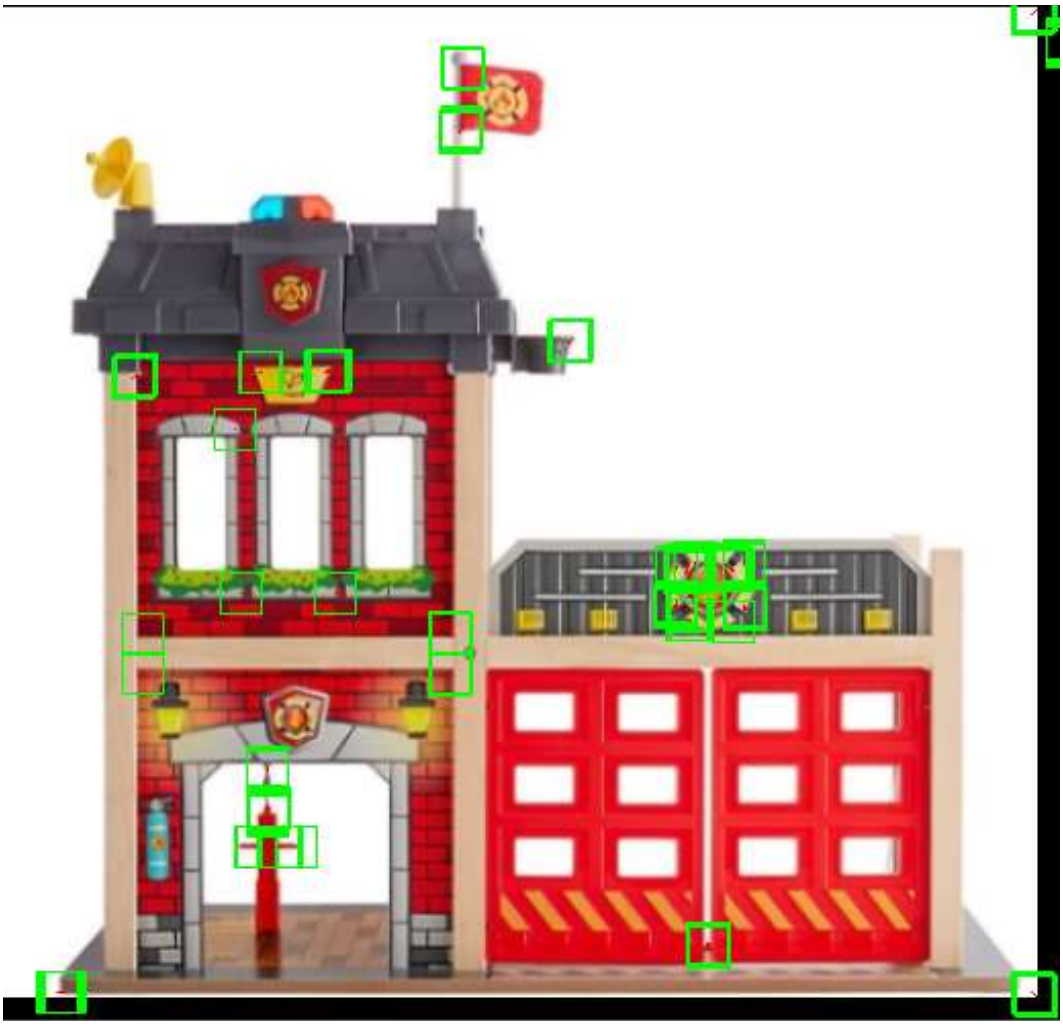
Feature Vector Calculation and Feature Matching.

In order to compute the feature vector, a brute-force matching with ORB descriptors using SIFT is used. For Feature Matching, similar pixels with less distance are matched and numbered accordingly.

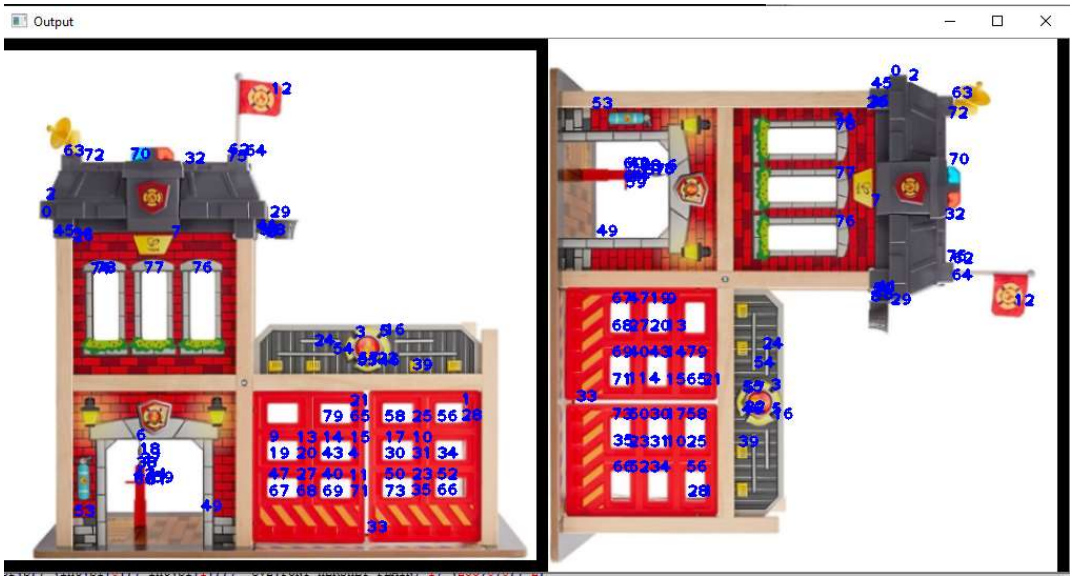
4. Results and discussion

Harris Corner Detection Output:
For k =0.2
Variance = 4
Threshold = 5000000
Neighborhood size = 5
Gives Output:

The k in range of 0.04 to 0.15 ensure corner detection
Variance is a blurring parameter, when the image is more blurred it is hard to detect corners.
Threshold is used to compare with the response fron the Harris detection function.



Feature Matching:
As seen below the one image is straight and other is tilted but the features are matched correctly and numbered:



Feature Localization:

As it can be seen that the enlarged image has Corners Localized and highlighed as follows: