Algorithms in multimedia and machine learning in the Python environment

Yakir Menahem

Harris Corner Detector

Idan Montekyo & Yarin Shlomo

Summer 2021 – שנה"ל תשפ"א, סמסטר קיץ

מכון טכנולוגי חולון
Holon Institute of Technology

Introduction

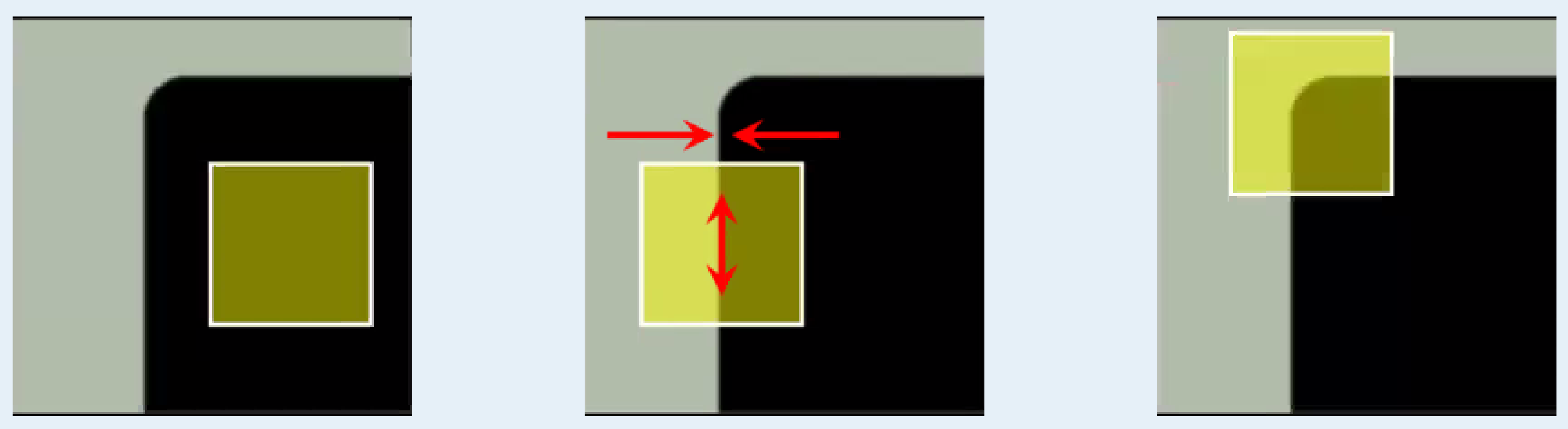
Development environment: PyCharm
Programming language: Python 3.8
Open-source libraries: OpenCV, Numpy, Pyplot, tk.filedialog

Motivation:
Harris corner detector is an algorithm to find corners in an image.
Corner detection is an approach used within computer vision systems to extract certain kinds of features and infer the contents of an image.
Corner detection is frequently used in motion detection, video tracking, panorama stitching, 3D reconstruction and object recognition. Corner detection overlaps with the topic of interest point detection.

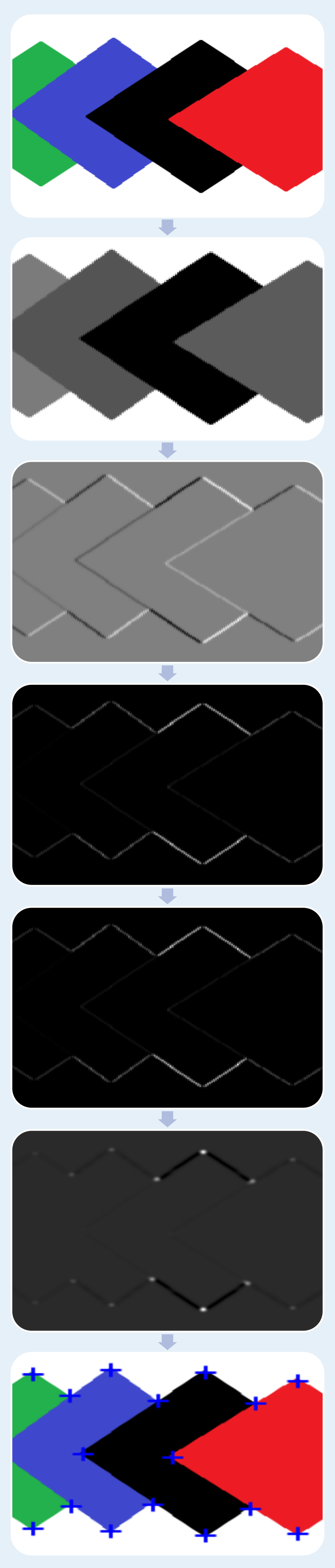
The Algorithm

Mathematical background:
Mathematical identification of corners using image gradients by looking at the pixel's neighbors.
We can classify the area into 3 region types by looking at the local gradients:

- **Flat –** no significant change in all directions of the gradients.
- **Edge –** a significant change in a single direction.
- **Corner –** a significant change in at least 2 directions.


Flat Edge Corner

Algorithm steps:



1. Receiving an image
2. Converting the image to gray-scale
3. Compute image gradients G_x , G_y
4. Compute 2nd order gradients G_x^2 , G_{xy} , G_y^2
5. Apply mask (filter) on 2nd order gradients
6. For each pixel (i, j) define the matrix M

$$\mathbf{M} = \begin{bmatrix} F \otimes G_x^2 & F \otimes G_{xy} \\ F \otimes G_{xy} & F \otimes G_y^2 \end{bmatrix}$$

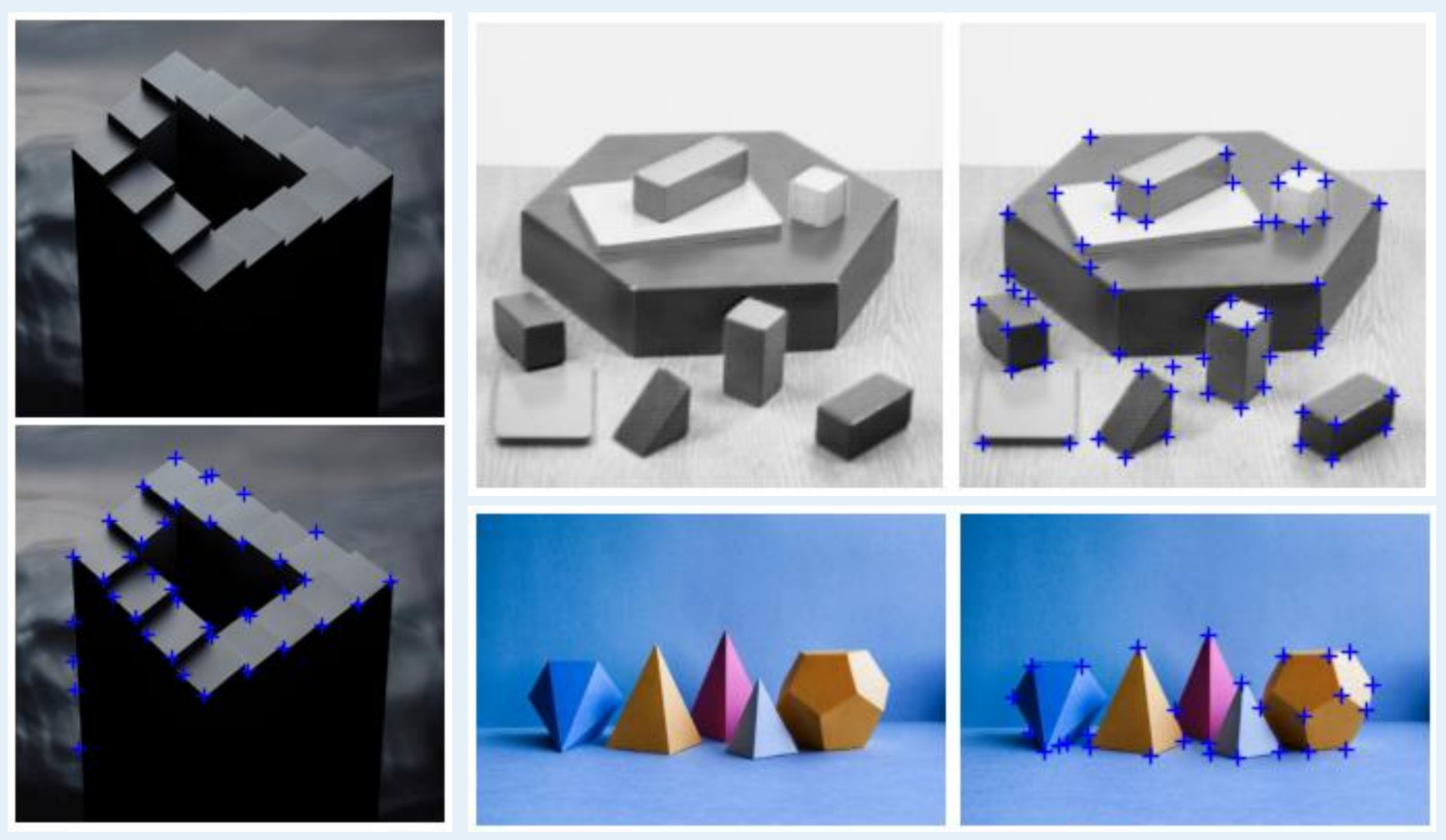
7. For each pixel (i, j) compute the score R:

$$\mathbf{R = \det(M) - k \cdot trace^2(M),\ k \in [0.04, 0.06]}$$

8. Threshold R and perform NMS (non-maxima suppression)
9. Mark the corners and print the result

Conclusion

Our implementation of Harris corner detector can successfully locate corners in images.
We managed to improve the algorithm to get better and more accurate results.
However, certain conditions may affect the accuracy.
Conditions such as noisy texture or bad resolution.
Therefore, we focused on locating object's corners.

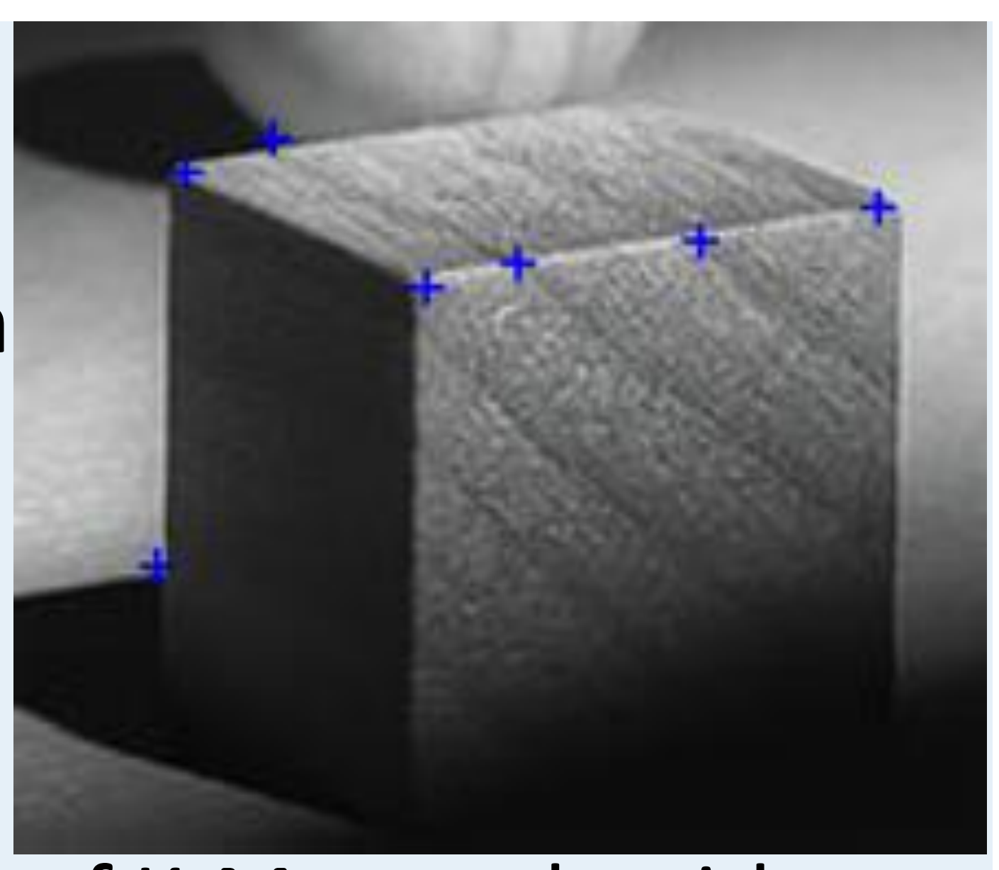


As you can infer, we managed to locate and mark the corners in the given images.
The results are still not perfect, but are better than the original Harris algorithm.

The algorithm's pros and cons:

Pros:	**Cons:**
- Uses mathematical equations in order to rate the chances of a pixel being a corner.	- Having hard time with lower contrast corners.
- Rotation invariance.	- Noisy textures might affect the accuracy.
- Illumination invariance.	- Does not always bring the ideal results.

Discussion



Although the results were good, we think our algorithm might improve.
For example we'll take this cube with noisy texture:
The results were not accurate due to the object's texture, therefore marked an edge as a corner.
For next research we suggest the use of K-Means algorithm to unite similar colors, or using Histogram equalization.
This way the noises will reduce, and the program will detect corners more efficiently.