Computer Vision - EE 00046746
HW2

Tomer Raz Adi Hatav

August 1, 2024

Part 1: Keypoint Detector

See Lowe (2004).

1.1 Images

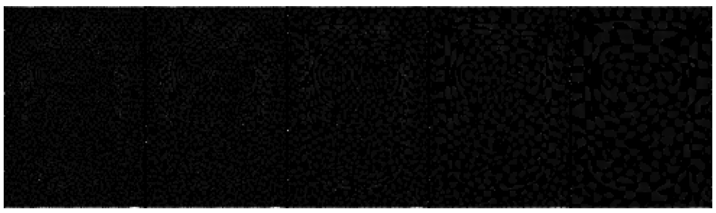
The image of interest.

1.2 Gaussian Pyramids

1.3 Difference of Gaussians (DoGs)

1.4 Edge Suppression

Principal curvature plot (normalized elementwise using $\frac{1}{5}10$ for visualization).

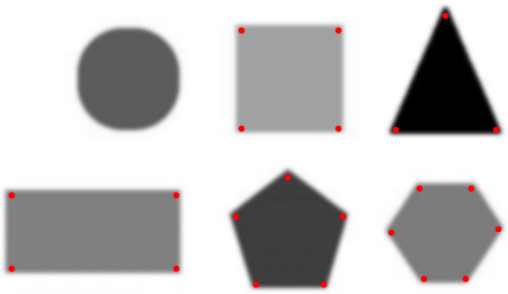


1.5 Detecting Extrema

```
def is_local_extremum(DoGPyramid: np.ndarray, x: int, y: int, i: int , shape_size:
    Tuple[int, int] = (256, 256)) -> bool:
    """
    Returns a boolean for whether the pixel at hand is a local extremum relative to up
        to 10 bordering pixels:
    up to 8 bordering pixels in the same level of the pyramid and 2 adjacent pixels in
        the level above and below.
    """
```

1.6 Putting it Together: DoG Detector

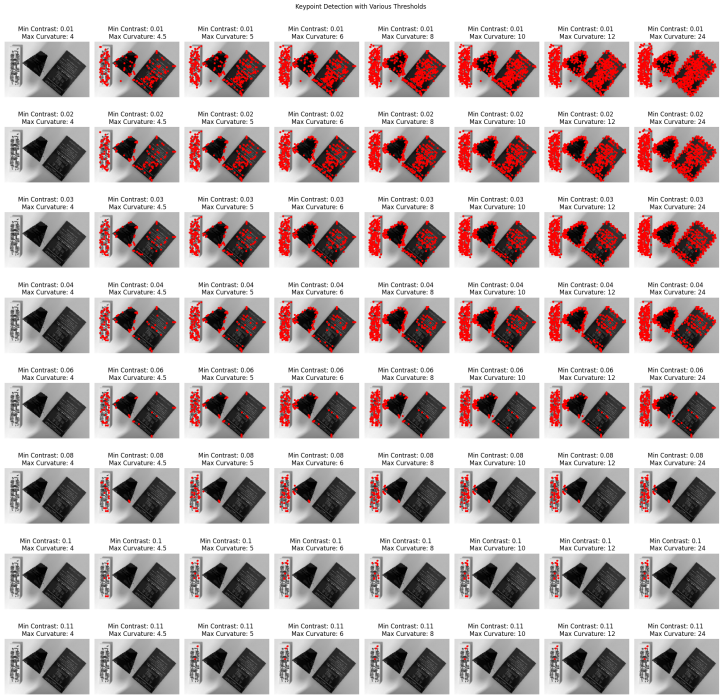
DoG detector results using a minimum contrast threshold of 0.03 and maximum curvature threshold of 12.



Some custom images:

Tests for different thresholds (on **custom** images): We found that local extrema tend to have a curvature greater than 4 (hence points only appear for max curvature threshold > 4) and for contrasts of < 0.06 Nalla!

Some random objects with sharp angles found around the house.



Part 2: BRIEF Descriptor

Binary Robust Independent Elementary Features. patchWidth = 9 (patch is square), n = 256 (number of patch pairs for comparison)

2.1 Creating a Set of BRIEF Tests

```
def makeTestPattern(patch_width: int, nbits: int) -> Tuple[np.ndarray, np.ndarray]:
    """
    Creates two arrays of random integers to be used as indices of patches for
        comparing pixel intensities.
    """
```

We chose to attempt method (I) of section 3.2 of (Calonder et al., 2010): randomly and uniformly choosing pairs of points to compare in the patches (9×9 pixels). The resulting matrices are saved to testPattern.mat and need to be **TO DO** in 'code' folder

2.2 Compute the BRIEF Descriptor

```
def computeBrief(im: np.ndarray, compareX: np.ndarray, compareY: np.ndarray,
                 GaussianPyramid: Union[np.ndarray, Sequence[np.ndarray]],
                 locsDoG: np.ndarray, k: float = np.sqrt(2),
                 levels: List[int] = [-1, 0, 1, 2, 3, 4], patchWidth: int = 9)
                -> Tuple[np.ndarray, np.ndarray]:
    """
    Given the keypoint locations of an image, compute the BRIEF descriptors
    for those which have patch fully contained in the image.
    Returns the valid keypoint locations and their descriptors.
    """
```

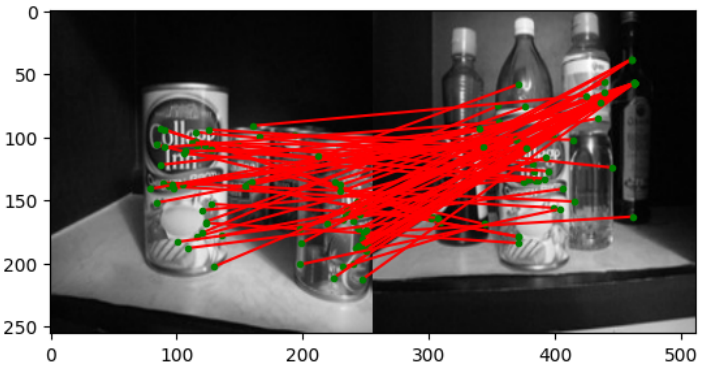
2.3 Putting it Together: Extracting Descriptors

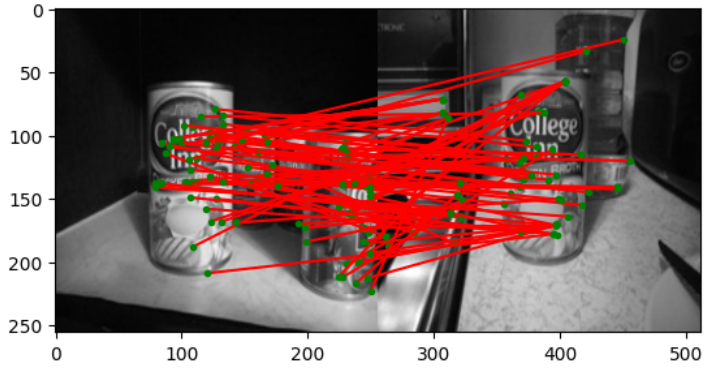
```
def briefLite(im, sigma0: int = 1, k: float = np.sqrt(2),
              levels: List[int] = [-1, 0, 1, 2, 3, 4],
              th_contrast: float = 0.03, th_r: float = 12,
              patch_width: int = 9, nbits: int = 256)
             -> Tuple[np.ndarray, np.ndarray]:
    """Returns valid keypoint locations and their descriptors."""
```

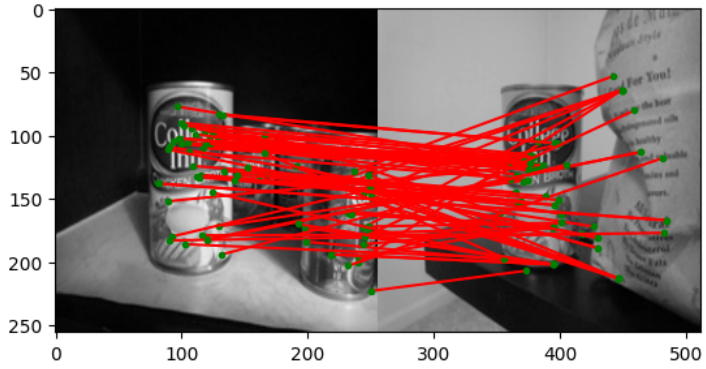
2.4 Checkpoint: Descriptor Matching

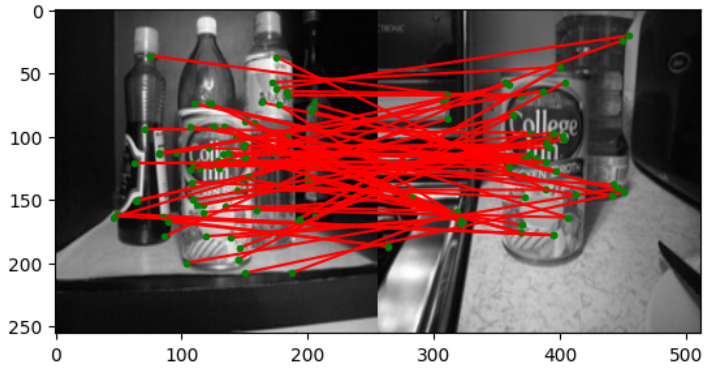
```
def testMatch(im1_path: str, im2_path: str):
    """Given two images, returns the matches between their keypoints."""
```

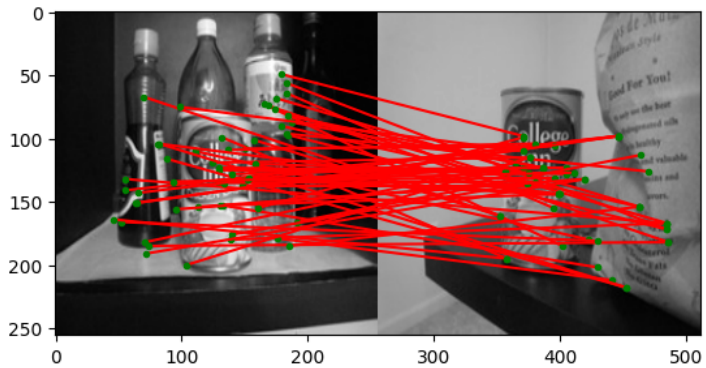
Some results on different images of the can of chicken broth:

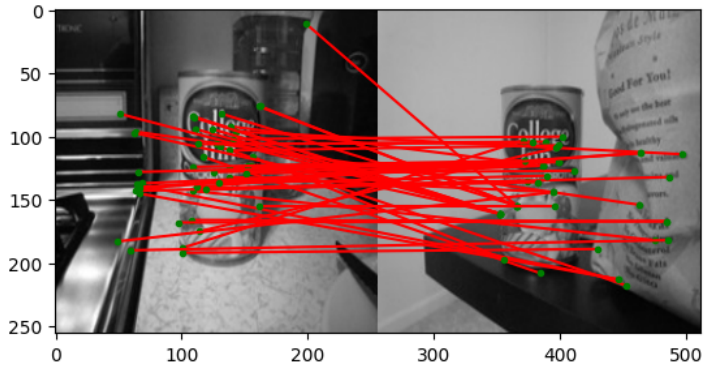



0
50
100
150
200
250
0
100
200
300
400
500


0
50
100
150
200
250
0
100
200
300
400
500


0
50
100
150
200
250
0
100
200
300
400
500

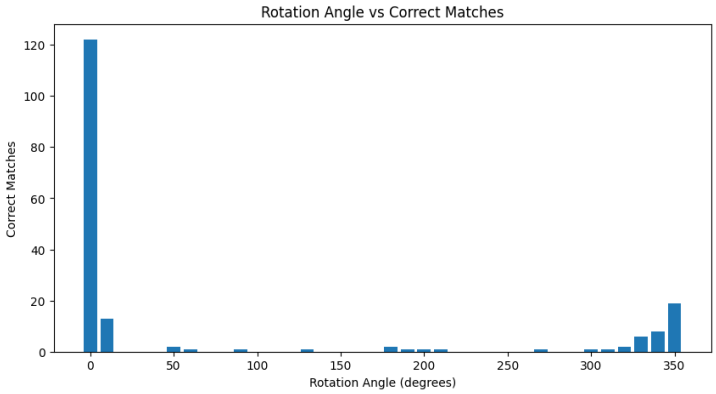

0
50
100
150
200
250
0
100
200
300
400
500


0
50
100
150
200
250
0
100
200
300
400
500

2.5 BRIEF Properties

- Is BRIEF invariant to scale? No, we could use the pyramid technique from the SIFT algorithm to make point matching more resilient to different scales. The precise implementation requires more thought.
- Is BRIEF invariant to illumination changes? Is seems to do "reasonably well" in different lighting conditions. But there is no pair of images for which keypoints seem (to our human eyes) to match well.

Below there is a bar plot presenting how BRIEF behave when we rotate the image in X degrees. as we can see, when we rotate the image, the BRIFE performance significantly decreases. For the default thresholds of 0.03 (contrast) and 12 (curvature). The BRIEF keypoint matcher does not manage to reliably identify keypoints for angles greater than $\sim \pm 10$



degrees.

2.6 Oriented Fast and Rotated BRIEF (ORB)

Include the resulting bar plot in your PDF and explain the key differences that make ORB rotation-invariant compared to BRIEF. Which components in ORB are borrowed from SIFT?

ORB enhances BRIEF by making it rotation-invariant through the following key differences:

- **Orientation Assignment**: ORB assigns an orientation to each keypoint using the intensity centroid method, which is used to rotate the BRIEF descriptor accordingly.
- **Rotation-Invariant BRIEF**: The BRIEF descriptor is rotated according to the keypoint's orientation, ensuring invariance to image rotations.
- **FAST Keypoint Detector**: ORB uses the FAST detector for efficient and robust keypoint detection, with keypoints selected based on a Harris corner measure.

Components Borrowed from SIFT

- **Orientation Assignment**: Like SIFT, ORB assigns an orientation to each keypoint to achieve rotation invariance, though it uses the intensity centroid method instead of gradient histograms.
- **Multi-Scale Pyramid**: ORB can be applied across multiple scales using an image pyramid, similar to SIFT, making it robust to scale changes.

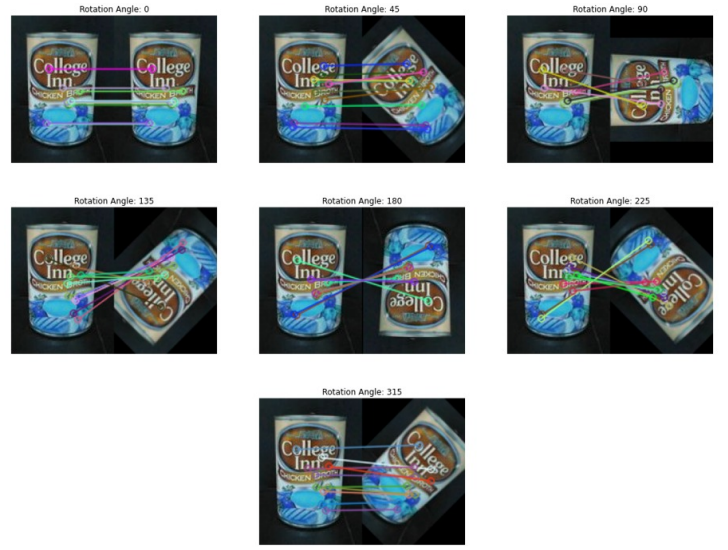
Performance Under Rotation

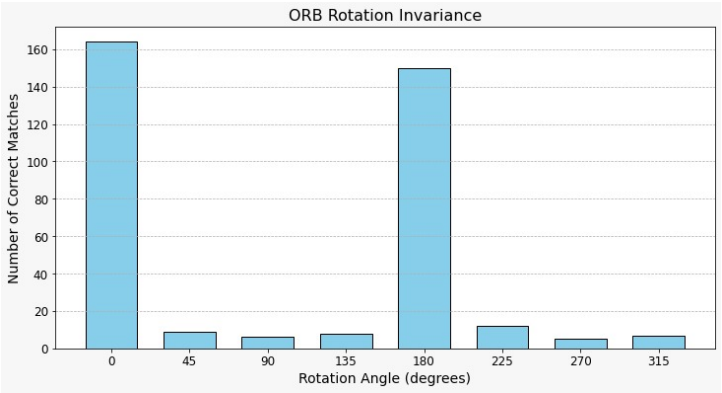
The image and bar plot showing us ORB performed well in rotation of 180 degrees but it not so well on other angles comparatively. Regardless, it performs much better than the DoG detector implemented in the exercise.

Part 3: Dry Questions

3.1 Semantic vs Ordinary Image Segmentation

Both are ways of segmenting an image, meaning classifying pixels. Semantic segmentation is segmentation that utilizes an understanding of the objects in the image to classify pixels into predefined classes. Pixels in different areas geometrically can be classified similarly. On the other hand, ordinary segmentation classifies pixels based on geometrical aspects of the image, by identifying different objects. No "world knowledge" regarding relations between objects is needed. All that is needed is the image since only geometric relations between pixels are used. To exemplify, a semantic segmentation algorithm may use

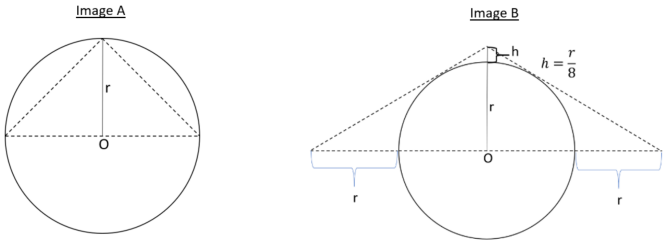




a foundation model trained on both images and language while an ordinary segmentation algorithm could rely only on image training.

An example for each: U-net architecture for semantic segmentation, R-CNN (R for region) for regular segmentation (explained in tutorial 5 on deep semantic segmentation).

3.2 IoU examples



- Image A:

$$\text{IoU}(A) = \frac{\text{Area}_{\triangle}}{\text{Area}_{\circ}} = \frac{2 \cdot \frac{1}{2} r^2}{\pi r^2} = \frac{1}{\pi} \approx 0.32$$

- Image B:

$$\text{IoU}(B) = \frac{\text{Area}_{\text{half circle}}}{\text{Area}_{\triangle} + \text{Area}_{\text{half circle}}} = \frac{\frac{1}{2}\pi r^2}{\frac{1}{2}\pi r^2 + 2 \cdot \frac{1}{2} \cdot 2r \cdot (r+h)} = \frac{\frac{1}{2}\pi r^2}{\frac{1}{2}\pi r^2 + 2 \cdot \frac{1}{2} \cdot 2r \cdot (\frac{9}{8})r} = \frac{\frac{1}{2}\pi}{\frac{1}{2}\pi + \frac{9}{4}} \approx 0.41$$

References

M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 778–792. Springer, 2010.

D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.