**Overview:**

This project was developed during the course 'Image Processing' in Azrieli - Jerusalem College of Engineering. The project is divided to 4 parts.

**Part 1:**

Isolate footprint in image and find the ratio of the b&w rectangle of the ruler We've dealt with the issue with the following steps:

Isolate footprint:

1. Convert RGB image to Grayscale
2. Execute Canny edge detection on the Grayscaled image
3. Use HoughLines to find the width and height of an sub-image where the footprint exists

Finding the ratio of the B&W rectangles:

1. Convert from RGB to grayscale
2. Smooth
3. Threshold
4. Detect edges
5. Find contours
6. Approximate contours with linear features
7. Find "rectangles" which were structures that: had polygonalized contours possessing 4 points, were of sufficient area, had adjacent edges were ~90 degrees, had distance between "opposite" vertices was of sufficient size, etc.

**Part 2:**

Find patterns of circles in footprints. We've dealt with the issue with the following steps:

1. Convert RGB image to Grayscale
2. Find 3 types of circles - big, medium and small. Finding big circles:
   * Run morphologyEx with a morphological gradient to find the difference between dilation and erosion of an image.
   * Smooth the image with blur() using filter size 3x3
   * Run HoughCircles() to find patterns of a big circles

Finding medium circles:

* + Run morphologyEx with a closing operation to execute dilation followed by erosion.
  + Smooth the image with blur() using filter size 3x3.
  + Run HoughCircles() to find patterns of a medium circles

Finding small circles:

* + Smooth the image with blur() using filter size 5x5
  + Use HoughCircles() to find patterns of a small circles

**Part 3:**

Simulate a search engine that gets an image of a cropped and noised footprint, and finds the full footprint image of this cropped image, from the Database of footprint images.

**Important: to run this part you should run the following CLI commands: pip install opencv-python==3.4.2.16, pip install opencv-contrib-python==3.4.2.16**

We've dealt with the issue with the following steps:

1. Convert RGB cropped image to Grayscale
2. Smooth the cropped image by running morphologyEx() with a closing operation to execute dilation followed by erosion, and blur() using filter size 5x5.
3. Convert RGB full image from DB to Grayscale
4. Run SIFT (Scale-Invariant Feature Transform) algorithm on both cropped and database images to get the key-points and descriptors of them.
5. Run FlannBasedMatcher method which is used to find the matches between the descriptors of the 2 images.
6. Find the matches between the 2 images, and store those matches in the 'matches' array. ('matches' array will contain all possible matches, also false matches).
7. Apply a ratio test to select only the good matches. The quality of a match is define by the distance. The lower the distance is, the more similar the features are. By applying the ratio test we can decide to take only the matches with lower distance, so higher quality.
8. Save the image with the maximum good points from all the database images.

**Part 4:**

Find patterns of circles in real crime scene footprints. We've dealt with the issue with the following steps:

1. Convert RGB image to Grayscale
2. Smooth the image with median filter size 3x3
3. Find the difference between dilation and erosion of an image
4. Use HoughCircles to find patterns of a circles

The outcome for this part is not similar to the outcome in part 2 because the footprints are not "perfect". for example, the image is much more "dirty".