UNIVERSITÀ DI BOLOGNA

Caps Inspection

Davide Angelani

November 26, 2022

1 INTRODUCTION

The project consists in developing a software capable of detecting defect in plastic caps. The software will be developed in Python and will be able to detect defects in caps by using Computer Vision techniques. Both task will be developed using opencv library. The project will be divided in two task and the software should be able to determine:

- **Task 1**:
  - The position of the center of the cap;
  - The diamiter of the cap mouth;
  - Answering the questions: "Is the liner missing?", "Is the liner incomplete?";
- **Task 2**:
  - The position of the center of the liner;
  - The diamiter of the liner;

The report will be divided as follow: First I will introduce the theory behind the techniques used in the project, then I will describe the first task and the results obtained and finally, I will describe the second task and the results obtained.

2 THEORY

2.1 Canny Edge Detection

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect edges in a image. The algorithm is composed by 5 steps:

1. *Noise reduction*: The first step is to reduce the noise in the image. This is done by using a 5x5 Gaussian filter.
2. *Gradient calculation*: The second step is to find the intensity gradients of the image. This is done by calculating the first derivatives in both the x and y directions.
3. *Non-maximum suppression*: In this step, we suppress the non-maximum pixels of the gradient magnitude image.

4. *Double threshold*: In this step, we determine the threshold of the image. Any gradient value larger than the maxVal is considered to be an edge. Any value below the minVal is considered not to be an edge. Values in between the minVal and maxVal are considered edges or non-edges based on their connectivity. If they are connected to "sure-edge" pixels, they are considered to be part of edges. Otherwise, they are also considered not to be edges.
5. *Edge tracking by hysteresis*: This step completes the edge detection by suppressing all the other edges that are weak and not connected to strong edges.

2.2 Hough Transform

The Hough Transform is a technique used to detect lines, circles and ellipses in an image. So it enables us to outline the the cap and detect defects. It is based on the projection into a parameter space known as Hough space.

2.2.1 Hough Transform for lines

Given the equation of a line in the form:

$$y - mx + c = 0 \tag{2.1}$$

usually we fix the m and c and we have a mapping from the parameter space to the image space. If we fix y and x, we have a mapping from the image space to the parameter space, so we get all the lines passing through a point (x, y).

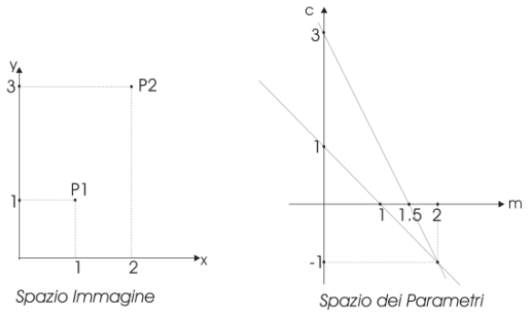


Figure 2.1: Mapping from the parameter space to image space and viceversa

So given a set of points, the parameter space is composed of many intersections as there are lines that can be drawn through the points. If we consider a set of collinear points, we obtain that all lines associated to such points intersect at the same point in the parameter space. Higher will be the number of intersection, higher will be the probability that a lines will be present in the image. The only problem is that m can be infinite, so we need to discretize the parameter space. To do so, we can use the polar representation of the line:

$$r = x\cos\theta + y\sin\theta \tag{2.2}$$

where r is the distance from the origin and θ is the angle between the line and the x axis. This representation map point (x, y) into sinusoidal curves in the parameter space.

2.2.2 Hough Transform for circles

The Hough Transform for circles is similar to the one for lines, but the equation will be:

$$(x-a)^2 + (y-b)^2 = r^2 \tag{2.3}$$

So it is a mapping from an image point (x, y) to a point in the parameter space (a, b, r). Since we have 3 parameters our parameter space will be 3 dimensional, but if we fix r we will have a 2 dimensional parameter space. So in the parameter space we get a circle which is the intersection of all the circles passing through the point and which have radius r.

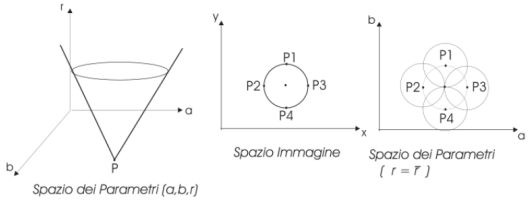


Figure 2.2: Mapping from the parameter space to image space and viceversa

3 TASK 1

3.1 Outline the cap

The first task is to outline the cap by generating a circle that fits the cap mouth. To do so we will use the Hough Transform for circles. opencv provides a function that does exactly that.

Listing 1: Outline the cap

```
circles = cv2.HoughCircles(img, cv2.HOUGH_GRADIENT, 1, 20, param1=50,
    param2=30, minRadius=0, maxRadius=0)
```

The function implements a detection method which is slightly different from the Hough Transform. The *Hough Gradient* is made op of two stages:

1. *Edge detection*: The first stage is to detect the edges in the image. This is done by using the Canny edge detector.
2. *Circle detection*: The second stage is to detect the circles in the image. This is done by using the Hough Transform.

3.2 Detect defects

There are two types of defects:

- **Defect 1**: the liner is not present in the cap.
- **Defect 2**: the liner has some cracks.

3.2.1 Defect 1

The first defect we want to detect is if the liner is not present in the cap. To do so we will use the image masked by the circle generated in the previous step. Then we can notice that the mean of the grayscale image is similar in all the images except for the ones in which the liner is not present. So we can use this information to detect if the liner is present or not.

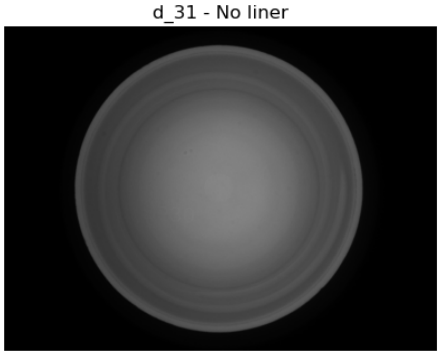


Figure 3.1: example of caps without liner

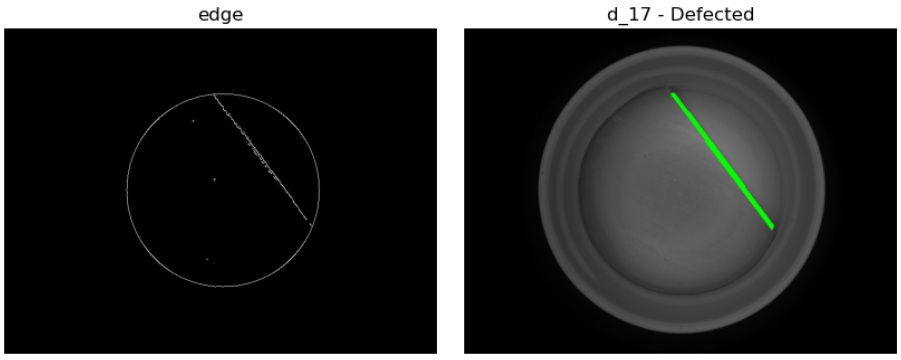
3.2.2 Defect 2

The second defect we want to detect is if the liner has some cracks. This time since we are interested only in the liner, we can mask out everything else. Then we perform a canny edge detection to find the edges of the liner. Finally we can use the Hough Transform for lines to detect the cracks.

Listing 2: Detect defects

```
lines = cv2.HoughLinesP(edges, 1, np.pi/180, 100, minLineLength=100,
    maxLineGap=10)
```

The function implemented in opencv is a probabilistic version of the Hough Transform for lines. Unlike the Hough Transform it returns the extremes of the lines instead of the equation of the lines. If the function returns a line, it means that there is a crack in the liner.



(a) Edges (b) Defect detected

4 TASK 2

The second task is to outline the liner of the cap. Given that the liner is present, we can use again the Hough Transform for circles to detect the circle that fits the liner. This time to prevent finding the circle that fits the cap mouth, we can define the maximum radius of the circle smaller than the one that fits the cap mouth.

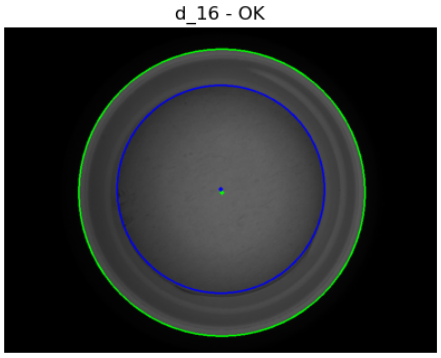


Figure 4.1: Example of an image with all the circles detected