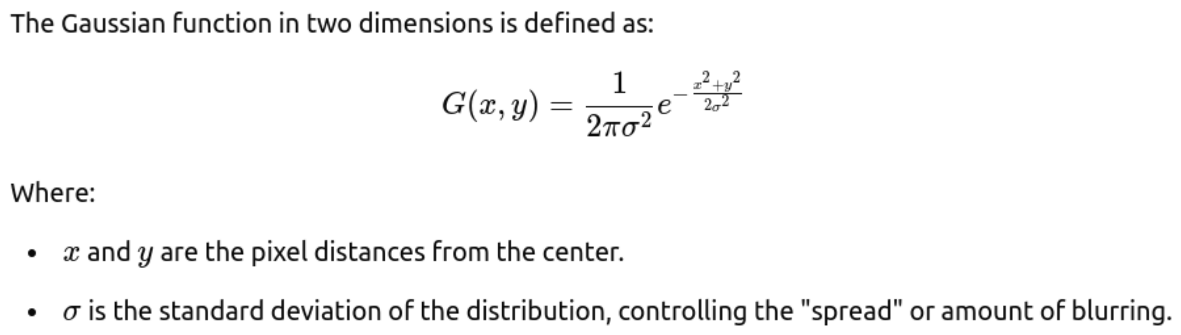
Finding Object

1. Convert the image into GrayScale

Objects in an image often have distinct boundaries or shapes that are identifiable by their intensity changes (e.g., between light and dark regions). By focusing on the intensity (without color), we can better highlight these boundaries, which is important for edge detection.

2. Applying Gaussian Blur



The Gaussian function in two dimensions is defined as:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Where:

- x and y are the pixel distances from the center.
- σ is the standard deviation of the distribution, controlling the "spread" or amount of blurring.

Noise Reduction: Real-world images can contain noise (small intensity variations) that might interfere with edge detection. By blurring the image, we can smooth out noise, making it easier to detect the true edges of objects.
Edge Detection Stability: Blurring reduces the influence of small intensity changes that do not correspond to meaningful object boundaries, making edges more stable and continuous.
Cleaner Edges: Object boundaries become clearer and less noisy after applying a Gaussian blur. This makes edge detection (Canny) more reliable.
Prevents False Positives: Blurring ensures that tiny, irrelevant intensity variations are not mistakenly identified as edges.

3. Perform Edge Detection

Using Canny Edge Detection

4. Perform Template Matching

Compare the similarity between the template and the image at each location.

5. Filtering Locations

Filtering locations that are lower than threshold, keep only locations that have value greater or equal than threshold. Draw a rectangle surrounding the objects.

Counting Object

1. Convert the image into GrayScale

2. Applying Thresholding

To create a high-contrast binary image where objects (rabbits) are white against a black background, facilitating easier detection

3. Performing Morphological Opening to Clean the Image

4. Find Contours

Retrieve only the external contours (external contours are the contours that are on the edge of the image, internal contours are the contours that are inside the image)

5. Filtering Contours

Filter out very small areas that are unlikely to be objects

6. Iterating Over Filtered Contours to Count Objects

Example:

```
for contour in contours:
    area = cv2.contourArea(contour)
    if area > 500:  # Filter out very small areas that are unlikely to be rabbits
        rabbit_count += 1
```