École Centrale Lyon

ELC C04
Second Tutorial
Report

Template Matching and Image Transformations

Students :
Corentin Triboulet
Antoine Missue

Teacher :
Shaifali Parashar

11 avril 2023

Introduction

During this tutorial, we solve two different problems. One is about finding the location of Waldo in a amusement scene using template matching method, especially the SSD method of comparison. The second one is about transforming two images. We have an image of a bus with an advertisement and we want to replace this ad into an image of Simpson.

1 Template Matching

In this problem, we will play *'Where is Waldo ?'*. We have a scene of an amusement park in witch we have to find Waldo. In order to find him, we have access to two different templates witch represent the face of Waldo with and without noise.

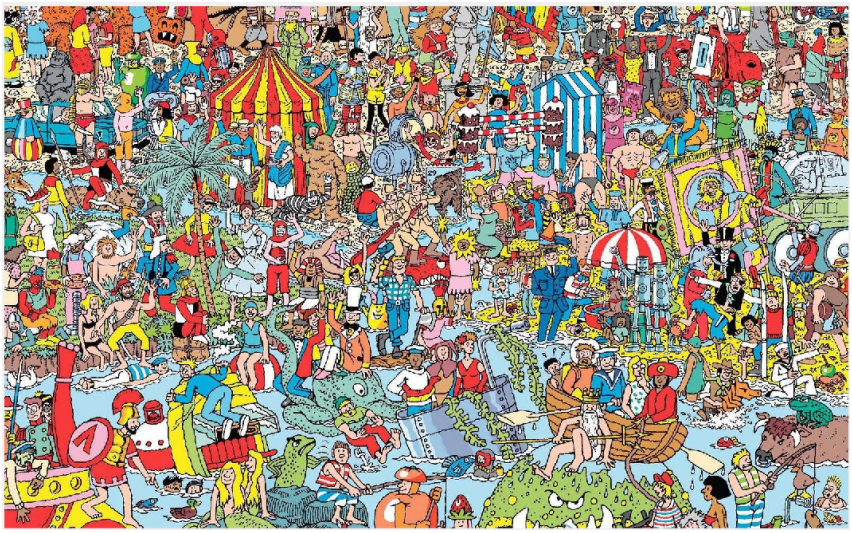

FIGURE 1 – Image of the scene

FIGURE 2 – Image of the Waldo's face without and with noise

To find Waldo in the reference image, we will use template matching method. First, we convert the image to grayscales. Then, we iterate over all pixel locations of the reference image and we compare the local patch with the template (the face of Waldo) using the sum of square distance (SSD). Finally, we find the location (x, y) of the pixel where the SSD is minimum. Normally, if everything wet well, Waldo is at the place of the patch

starting with this pixel. We displayed the image of the scene with a blue rectangle in the position where Waldo should be and we got this result.

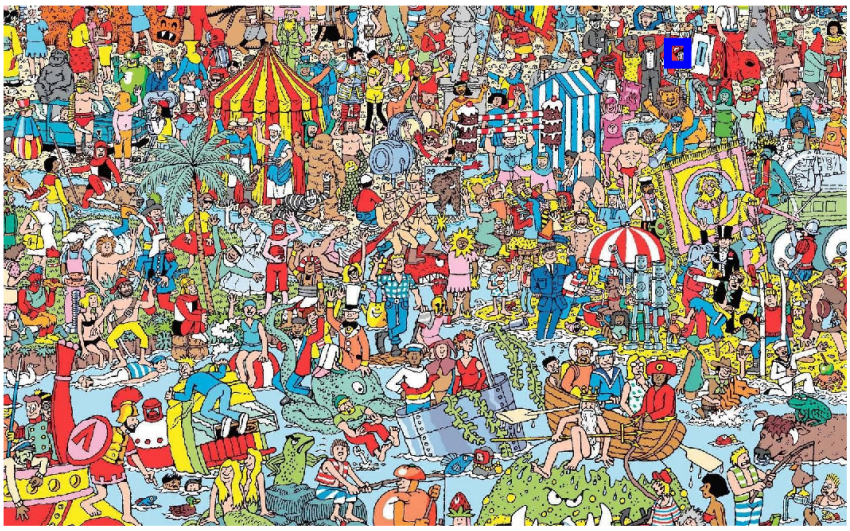

FIGURE 3 – Image of the scene with the rectangle on Waldo for the template 1

When we used the template of Waldo without any noise, we obtained a value of 0 for the minimum of the SSD. That means that the template corresponds exactly to the patch.

We repeated this process with the noisy template of Waldo. We also found where was Waldo but the minimum value of the SSD isn't 0 and is worth 293546. This is due to the difference between the noisy representation of Waldo's face in the second template and the real representation of it in the scene. We still found the location of Waldo because, despite the minimum of the SSD isn't zero, the patch of Waldo in the reference image is still the closest patch of template 2.

FIGURE 4 – Image of the scene with the rectangle on Waldo for the template 2

2 Image Transformations

2.1 Question 1

This code is an implementation of homography between two images, where one Simpsons image (`S`) is warped and overlaid onto another image of a bus (`BUS`).
The code starts by reading two input images (`BUS` and `S`) using the `imread()` function. Then, the advertising region of the `BUS` image is selected by specifying four points, and the pixels in the four corners of this region are set to red using nested for-loops. Therefore, one can see the correct rectangle that we going to replace.

FIGURE 5 – 4-corners Image

2.2 Question 2

Next, the homography is computed using the eight corresponding points in the two images. The homography matrix `M` is computed by solving the system of equations using the least squares method, where `T` is a matrix formed by the corresponding points in the `BUS` and `S` images, and `A` is the solution matrix. Here is how it works :

Coordinates of the image where one wants to put the second image are $\boldsymbol{x} = (x, y)$ and the coordinates of the second image (the image that will be modified) are $\boldsymbol{x'} = (x', y')$. The goal is to find the matrix corresponding to the equation (1).

$$c \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{bmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = M_{3x3} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{1}$$

Then one established the point matching equation, here they are our four corners. And one stores them in an 8x8-matrix as written in the equation (2). But our `M`-matrix has 9

coefficients. Now we assume that $a_9 = 1$.

$$\begin{bmatrix} & & & & \vdots & & & \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i x_i' & -y_i x_i' \\ 0 & 0 & 0 & x_i & y_i & 1 & -x_i y_i' & -y_i y_i' \\ & & & & \vdots & & & \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{bmatrix} = \begin{bmatrix} \vdots \\ x_i' \\ y_i' \\ \vdots \end{bmatrix} \tag{2}$$

Or :

$$T_{8x8} a_{8x1} = B_{8x1}$$

Then one applies the Linear Least squares :

$$a = (T^T T)^{-1} T^T B$$

And finally, we obtain $\boldsymbol{x'}$ by using the equation number (1) with $z = 1$ and $c = \frac{1}{z'}$ because one needs to normalize the solution. And it comes :

$$x' = \frac{a_1 x + a_2 y + a_3}{a_7 x + a_8 y + 1}$$
$$y' = \frac{a_4 x + a_5 y + a_6}{a_7 x + a_8 y + 1}$$

Finally, the warped image (`SIMPSONS_BUS`) is created by applying the homography matrix `M` to each pixel in the `S` image and mapping it to its corresponding location in the `BUS` image. This is achieved using nested for-loops, where each pixel in the `S` image is multiplied by the homography matrix `M`, and then the corresponding pixel in the `BUS` image is replaced with the pixel from the `S` image.
The resulting `SIMPSONS_BUS` image shows the original `BUS` image with the selected region of `S` overlaid and warped to match the perspective of the `BUS` image.

FIGURE 6 – Resulting Image (`SIMPSONS_BUS`)

2.3 Question 3

To find a transformation between the bus and Simpsons image, one can simply swap the input images in the original code. That is, one uses the Simpsons image as the reference image and the bus image as the image to be transformed. The resulting transformation matrix can then be used to transform every pixel in the bus image to its corresponding location in the Simpsons image.
The first image is the Simpson's base image (figure 7) and the image shown in figure 8 is created from the `SIMPSONS_BUS` image (figure 6).
The two results are not identical. In fact, Homographic transformations involve approximations because they are not exact mappings of the image pixels. Instead, homographic transformations use a set of corresponding points in two images to estimate a transformation matrix that can map one image onto the other. This transformation matrix can be used to warp the original image to align with the target image, but the resulting warped image may not perfectly match the target image due to the inherent approximations in the transformation matrix.
The third image represents the error between the base image and the flattened image. One can see that the homographic transformation is not good to reproduce edges. This is understandable because transformation involves stretching and warping the original image to align with the target image. This can cause the edges of the original image to be distorted or lost during the transformation, leading to mismatches between the corresponding pixels in the two images.

FIGURE 7 – Simpsons base image

FIGURE 8 – Image retrieved from the bus

FIGURE 9 – Simpsons's error image

In terms of computation, the two processes are largely the same. However, if one wants to use the second method to wrap the Simpsons image on the bus advertisement, one needs to interpolate for a second time. This would deteriorate even more the image retrieved on the bus (figure 8). Thus, results are different in terms of representation of the Simpsons image. One observes an error of $8 * 10^9$ between the two images.
Therefore, the transformation the more appropriate is the first one. One goes directly to the objective without a lot of errors.

In the code of question 3, one doesn't know why by simply swapping images the code works. It shows the image of the Simpsons that correspond quite perfectly but not in the bottom left corner. The 3 others corners are correctly fitting. One was forced to modify the coordinate of the bottom left corner point to $[379, 878]$. Here is the retrieved image without changing the fourth point, i.e. with those coordinates $[468, 808]$.

FIGURE 10 – Image without changing the fourth point

To modify the coordinate one uses the 4-corners image (figure 5) to obtain a correct fitting.

Conclusion

In this tutorial, we were able to apply two fundamental image processing techniques - template matching and homography transformation - to locate a target object in an image and align two images, respectively. These techniques demonstrate the power and versatility of image processing and highlight the potential for future applications and advancements in the field.