The Intro to Computer Vision labs will be run in Google Colaboratory, a Jupyter notebook environment that runs entirely in the cloud, you don't need to download anything. To run these labs, you must have a Google account.

Step 1: click on the assignment invite link -> **Accept this assignment**. Refresh page -> individual repo for the specific assignment is created automatically

Project 2, week 1: https://classroom.github.com/a/ntQlx132

Step 2: Navigate to http://colab.research.google.com/github -> Click the **Include Private Repos** checkbox -> **select the correct repo** (SistemeDeVedereArtificiala/assignment_name-student_name) -> Click on the jupyter notebook of the current assignment

Step 3: [GitHub sign-in window] In the popup window, sign-in to your Github account and authorize Colab to read the private files.

Step 4: [in colab] **File** -> **Save a copy to GitHub**. Select the correct repository for the SPECIFIC assignment -> Click the **Include Colab Link** -> Click **OK**

Step 5: [in colab] Navigate to the **Runtime** tab --> **Change runtime type**, under **Hardware accelerator** select **GPU/TPU** (tensor processing unit) according to your needs.

Read the suggestions and accomplish all tasks marked with **#TODO**.

!!! At the end of each laboratory **REPEAT step 4 in order to SAVE** the answers to your private repository (individual for each assignment)

Project 2: License plate text recognition

Label:B
Prediction output: B

1. Introduction to object descriptors

Objects are represented as a collection of pixels of an image. For recognising objects we need a method to describe the properties of such

groups of pixels. The object description is realised using a specific set of numbers called descriptors. Object recognition is realised by matching descriptors computed for objects in the image with the ones of known objects.

Object description	Shape boundary	Chain codes	
		Fourier descriptors	Cumulative angular function
			Elliptic descriptors
Object description	Region	Basic	Area Perimeter Compactness Dispersion
		Moments	First order Centralised Zernike

Table 1: Methods for Object Descriptors

Shape descriptors characterise the arrangement of contour pixels (shape boundary, perimeter, margin) while **region descriptors** define pixels within the object contour.

2. Fourier Descriptors

Fourier Descriptors use the properties of Fourier theory in order to describe a shape. The main idea is to characterise a contour using a set of numbers which represent the frequency content of the shape. Based on the frequency analysis a small number of Fourier coefficients are selected to describe the shape.

Let ***x[m]*** and ***y[m]*** be the coordinates of the m^{th} pixel on the boundary of a given 2D shape containing ***N*** pixels, a complex number can be formed as ***z[m] = x[m] +j*y[m]***, and the *Fourier Descriptor (FD)* of this shape is defined as the DFT of ***z[m]***:

$$Z[k] = DFT[z[m]] = \frac{1}{N}\sum_{m=0}^{N-1} z[m]e^{-j2\pi mk/N} \quad (k = 0, \cdots, N-1)$$

FD can be used as a representation of 2D closed shapes **independent of its location, scaling, rotation and starting point**. For example, we could use ***M<N*** FDs corresponding to the low frequency components of the boundary to represent the 2D shape. The reconstructed shape based on these FDs approximate the shape without the details (corresponding to the high frequency components susceptible to noise). However, note that since the Fourier transform is a complex transform, the frequency spectrum has negative frequencies as well as positive frequencies, with the DC component in the middle. Therefore the inverse transform with ***M<N*** components needs to contain both positive and negative terms:

$$\hat{z}[m] = \sum_{k=-M/2}^{M/2} Z[k]e^{j2\pi mk/N} \quad (m = 0, \cdots, N-1)$$

To compute the FD:

1. Find object contour (see cv2.findContours)
2. Create complex numbers using the (x, y) coordinates of the contour points: z = x + j*y
3. Compute the Fourier transform for every complex number

Figure 1: Input shape

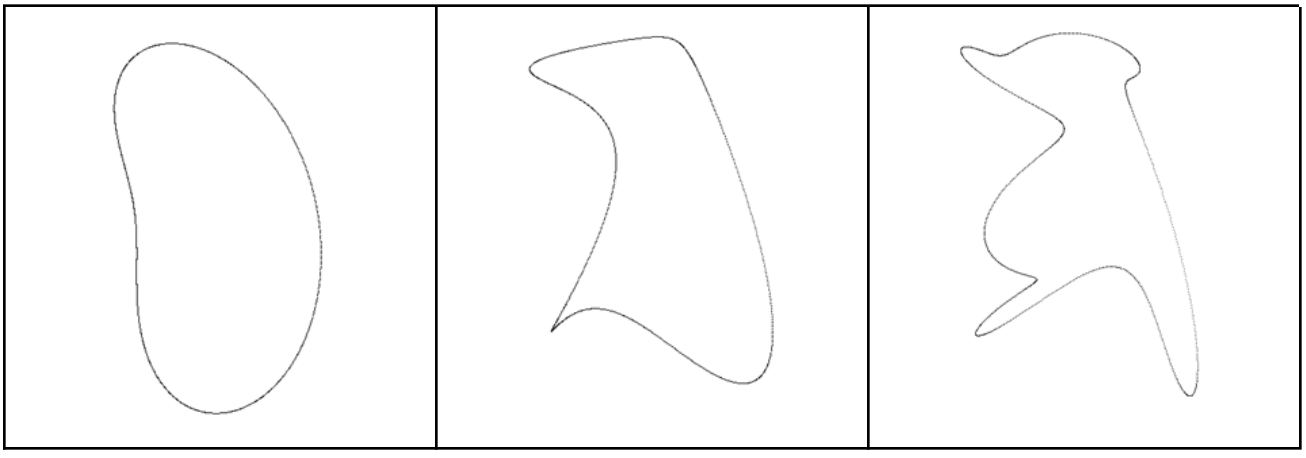

Table 2. Shape approximation for different number of Fourier Descriptors (2, 4, 6 | 8, 10, 20 | 30, 40, 50 | 100, 200, 300)

Demo: https://demonstrations.wolfram.com/FourierDescriptors/

3. Hu Moments

Moments describe the arrangement of the pixels inside a curve by combining area, compactness, irregularity and other curve descriptors. Moments represent global descriptions of a shape. In image analysis, these are statistical moments, as opposed to mechanical ones, but the two are similar. For example, the moment of mechanical inertia describes the rate of change in impulse; the second-order statistical moment describes the rate of change of the shape of the area.

Moments are often associated more with statistical pattern recognition than with vision-based models because it is assumed that the target shape does not suffer from occlusions (it is not partially covered). In simpler applications, images are often obtained by imposing a threshold and are usually a single object in the field of view. The moments produce a global description with invariant properties and with the advantage of a compact description made to avoid the effects of noise.

Hu Moments (or rather **Hu moment invariants**) are a set of 7 numbers calculated using **central moments** that are invariant to image transformations. The first 6 moments have been proved to be invariant to **translation**, **scale**, and **rotation**, and **reflection**. While the 7th moment's sign changes for image reflection.

$$h_0 = \eta_{20} + \eta_{02}$$
$$h_1 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$
$$h_2 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$
$$h_3 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
$$h_4 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$
$$h_5 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})]$$
$$h_6 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

Image	H[0]	H[1]	H[2]	H[3]	H[4]	H[5]	H[6]
	2.78871	6.50638	9.44249	9.84018	-19.593	-13.1205	19.6797
	2.67431	5.77446	9.90311	11.0016	-21.4722	-14.1102	22.0012
	2.67431	5.77446	9.90311	11.0016	-21.4722	-14.1102	22.0012
	2.65884	5.7358	9.66822	10.7427	-20.9914	-13.8694	21.3202
	2.66083	5.745	9.80616	10.8859	-21.2468	-13.9653	21.8214
	2.66083	5.745	9.80616	10.8859	-21.2468	-13.9653	-21.8214

Further study on the topic at:

1. Computer Vision: Algorithms and Applications (chapter 3.4), Richard Szeliski
2. Analysis of Hu's moment invariants on image scaling and rotation, Zhihu Huang