TD 2: Analysis and modeling of artificial visual systems

Objective: The goal of this assignment is to learn to describe a visual system, to analyse its statistics and to build models of efficient code for it.

Components: There is a written component, where you have to describe the visual system, a code component for the statistics and modeling part and a interpretation component where you have to analyse your results.

Code Guide Lines: It is mandatory to use python, it is recommended to use ipython (but if you use spider, it is also fine). The functions are written in the files: PSpy.py, WhiteningFilterspy.py, ICApy.py, You should write your code in the file main.py and the functions files. The code should be organized and commented in such way that it is possible to run and reproduce the figures that appear in the report and understand the process.

New packages to use: You will have to use scikit-learn (https://scikit-learn.org/stable/), and h5py (https://www.h5py.org/).
HDF5 file stands for Hierarchical Data Format 5, It is an open-source file which comes in handy to store large amount of data. I have chosen to use this format because it is very efficient, it compresses the data without loss and it allows to load the data one by one in the memory of the computer program. Thus, it is very suitable for processing images one by one.
H5py is an interface to read and write numpy arrays in the hdf5 format. Below is a link to a tutorial on this package, please note that they suggest to use h5pyViewer, to view these files graphically. I have never used the viewer, but you can try and, if you have time, send me your opinion.
https://www.geeksforgeeks.org/hdf5-files-in-python/
I have also written a small tutorial on this package in the end of this document. Please, read it before starting to work on the code.

Working with large datasets: When using large datasets, it is recommended to write the code and test it with a small number of samples. Only after the code passes all existing tests, itA should be run over the entire dataset.

Submission: You should submit in the Moodle the a zip/tar file with code, report files and figures. The file should have name: MI210_TD2_GroupX, where X is your group number. If the Moodle does not work, send me an email to daniela.pamplona@ensta-paristech.fr, the subject of the email must be MI210_TD2_GroupX, where X is your group number.

Please do not forget to write in the header of the code and at the beginning of the report the name of all members of the group.
The deadline to submit the TD is April 28 at 9H. Each extra day is punished with 2 values on the TD's grade (between 0-20). This TD weights 6 values of your final grade.
Evaluation: Interpretation worth 75% of the points in questions with both code and interpretation components.

Before starting: Choose one of the autonomous systems to be analyzed from the list below:

Underwater robot	Autonomous vehicle
Domestic robot	Airport surveillance

Go to the spreadsheet
https://lite.framacalc.org/9fq8-groups-mi210-2020

and write the group number linked with the chosen system. One of the systems can be repeated, in that case, choose group 5. You should stick to this visual system in this and next assignment. Do not forget to update the spreadsheet with the group identifications.

For each artificial system, there is a dataset of its natural images available at:

https://filesender.renater.fr/?s=download&token=fe1b0928-77d6-4ac5-bcb1-179dff83a5c3

please download the one corresponding to your group. Let me know if you have problems with the download, for instance, if you have low internet connection.

The images were resized, converted to greyscale, with double values within the range 0-255.

Part I: The structure of a visual system

Q1) [3 val] Describe the functional and hardware requirements of the visual system that you are modeling. Please note that visual systems should be energetically cheap, meaning, hardware requirements should fit the system main functions. From the list below select three functions and the corresponding hardware requirements that are convenient.

List of functional requirements: face recognition, motion detection, pedestrian detection, anomalies detection, predators detection, peers recognition, vehicles detection, indoor classification, outdoor navigation, multicamera integration, pose estimation, people counter, traffic sign recognition, depth estimation, road detection, human-robot interaction, other (specify which)

List of hardware requirements: number of cameras, resolution, frame rate, the field of view size, chromatics, range, static/active cameras, other (specify which).

Part II: The second order statistics of natural images

The goal of this part is to analyze and estimate the power spectrum statistics of natural images samples for the visual system considered. The coding part is guided bellow, step by step.

The goal of this part is to analyze and estimate the power spectrum statistics of natural images samples for the visual system considered. The coding part is guided bellow, step by step.

Q2) [3 val] Estimate the average 2D power spectrum of samples of the natural images (32x32 pixels) of your visual system and compare it with the statistics of the natural images of humans/monkeys visual system (hint: the slides of the class might be useful).

I. Write a function generates randomly the top left corner position of the sample in the image (hint: NumPy.random). You should be careful and be sure that the entire image sample will be inside the image. From the top left corner position and the sample size, you can define the "safe" limits.

II. Write a function that, using a top-left corner randomly generated and a fixed size, extract an image sample

III. Write a function that calculates the power spectrum of an image sample (hint: NumPy.fft).

Write a function that averages the power spectrum of N random image sample

Q3) [3 val] Estimate the RADIAL average 2D power spectrum of samples of the natural images (32x32 pixels) of your visual system and compare it with the statistics of the natural images of humans/monkeys (hint 1: the slides of the class might be useful, hint 2: it is already implemented a function that returns the radial frequency for a 2D frequency space, you just have to use it to calculate the average)

Q4) [3 val] Estimate the local statistics of the images. For that consider a grid of 3x3 that defines 9 regions as shown in Figure 1. Thereafter, repeat Q2 in each region. Compare the local results with the position invariant results. Do you think that the radial average is the correct step for the analysis? Why?

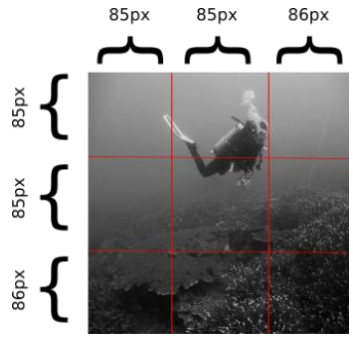

Figure1: Example of grid 3x3 dividing the image into 9 regions. Note that the length in pixels is figurative, please adapt it to your image size.

Part III: Power Spectrum whitening of natural images

Q5) [3 val] Estimate and analyze the power spectrum whitening filters of your visual system for various levels of noise and compare them with the monkey ganglion cells (hint1: in the slides of the class there is the equation. Hint 2: If you used NumPy.fft.fftshift to visualize and save the average power spectrum do not forget to use NumPy.fft.ifftshift before and after the use of the (inverse) Fourier transform. hint 3: By default, the result of NumPy.fft.ifft2 is a NumPy array of complex values. In our case, the imaginary part of the filter is zero. Convert the array of complex to an array of doubles using NumPy.real).

Part IV : Independent components of natural images

Q6) [2 val] Estimate and analyze the independent components of image samples of your dataset and compare it with the monkey V1 cells.

(Hint1: Tutorial on blind source separation using PCA and ICA in sckit:
http://scikit-learn.org/stable/auto_examples/decomposition/plot_ica_blind_source_separation.html

Hint 2: To avoid that you estimate new whitening filters you can use the option of FastICA to whiten the data in the fly.
Hint 3: To use the algorithm presented in the class you should set the option *algorithm='parallel'* in FastICA.
Hint 4: Given the complexity of the algorithm, it is better to estimate the independent components of small image samples, say 12x12 and a larger dataset, say 50000 image samples.)

Q7) [3 val] Estimate the sources of the image samples and test if they are sparse, by comparing them with a standard Gaussian distribution.

I. Write a function that, given observations and its independent components estimates the independent sources of the observations

II. Write a function that calculates the kurtosis of each observation. (hint: scipy.stats)

III. Compare the sources kurtosis with the kurtosis of standard Gaussian. Make as many plots and histograms as necessary to show your point.

Appendix: the HDF5 file format

HDF5 is a file format to store data in a binary compress way. Furthermore, it is readable from many languages like python, Matlab, c++... H5py is a package to interface the file with python, which is very simple and user friendly.

For this TD, the data are stored in the dataset 'images', which are in the root group. Below is some code to read and load the images from the hdf5 file. Do not load all images on your computer at the same time, it consumes too much memory or every freeze your computer. Instead, you can load the images one by one, or create a NumPy array of image samples.

Example 1: How to read the image 3 from your dataset.

```
import h5py
fileName = './greyImages.hdf5'
f = h5py.File(fileName,'r')
img = f.get('images')[3]
f.close()
```

Example 2: How to read a patch of 15 x 15 pixels with left corner at 200, 51 of the image 7

```
fileName = './greyImages.hdf5'
f = h5py.File(fileName,'r')
patch = f.get('images')[7, 200:215,51:66]
f.close()
```

Example 3: How to extract and apply the function f on 4 patches of size 15 x 15 pixel from 4 different images:

```
fileName = './greyImages.hdf5'
tbl = h5py.File(fileName,'r')
for i in range(4):
    idxImage = getImageIndex()
    tlf = getTopLeftCorner()
    patch = tbl.get('images')[idxImage, tlf[0]:tlf[0]+15,tlf[1]:tlf[1]+15]
    y = f(patch)
f.close()
```

Note: in this case, the file is opened and closed only once, thus the file remains open while you iterate over it. However, only 4 patches were loaded in the program memory.