IPBMA. Exercice 7.

Building a basic CT system. Sinogram detection.

Built an industrial CT scanner using python functions. Unlike the CT scanner for medical purposes, in the industrial CT scanner, the x-ray tube and detector remain fixed, and the object. There are two basic models. One is based on fan-beam x-rays and linear detectors, where the x-ray tube and detector move up and down to cover the entire object. The second is based on cone-beam X-rays and full-field detectors. There is no need to move the source and detector in this second case. This exercise aims to simulate a CT scanner based on fan-beam x-rays and linear detectors (see figure 1) and obtain a sinogram from the cubic phantom using such a device.

The python functions to be implemented will be called from a main program and will be: ***source()*** (already implemented)***, cube_phantom_cc()*** (already implemented)***, interactor_CT(), detectSinogram(), detector_1D(), and process_CT*()** and will include the following parameters:

interactor_CT(N0, Object, zPos, nProjections).- function that simulates the interaction between the x-ray beam and the object. This interaction depends on the linear attenuation coefficient and the object's shape. Both quantities are included in the object itself. Also, for CT, the interaction depends on the height at which we shoot (zPos) and the number of angles we use (nProjections) for those shots. These angles are assumed to be equispaced for a total distance of 360 degrees.

i) N0 → Number of photons generated per unit area
ii) Object → Subject to X-ray
iii) zPos → The height of the object where the shot is taken
iv) nProjectios → number of angels (projections) used at the acquisition

Output→ Numpy array (2D), whose values represent the sinogram of the CT in terms of quantum image.

detectorSinogram(qImage, nProjections, nDetectors).- function that simulates the process of detecting the sinogram using a linear detector. Inside this function, you have to call the function detector_1D(), which you have to implement, also.

i) Image → Quantum image captured
ii) nProjections → number of angels (projections) used at the acquisition
iii) nDetectors → number of detector cells of the linear detector used

Output→ Numpy array (2D), whose values represent the pixel values of the sinogram acquired.

detector_1D(qImage, angle, nDetectors).- function that simulates the 1D detector used in the acquisition process. Suppose the detector is ideal. No noise considerations are made, and each cell's size is equal to the size of each pixel of the quantum image that represents the sinogram. Thus, nDetectors take control of the FOV of the system.

i) Image → Quantum image captured
ii) angle → Acquired angle. In terms of quantum imaging, the row's height to be detected.
iii) nDetectors → Number of detector cells of the linear detector used

Output→ Numpy array (1D), whose values represent the pixel values of the sinogram detected.

process_CT(image, n0).- which represents the image transformation that the detector electronics make to compensate for the exponential distribution of the components of the captured quantum image. Therefore, this function calculates the neperian logarithm of each element of the incident radiation. Use n0 to normalize the number of detected photons, represented by the sinogram.

i) Image → detected image of the sinogram
ii) N0 → Number of photons generated by the source per unit area

Output→ Numpy array (2D), whose values represent the pixel values of the sinogram, compensated.

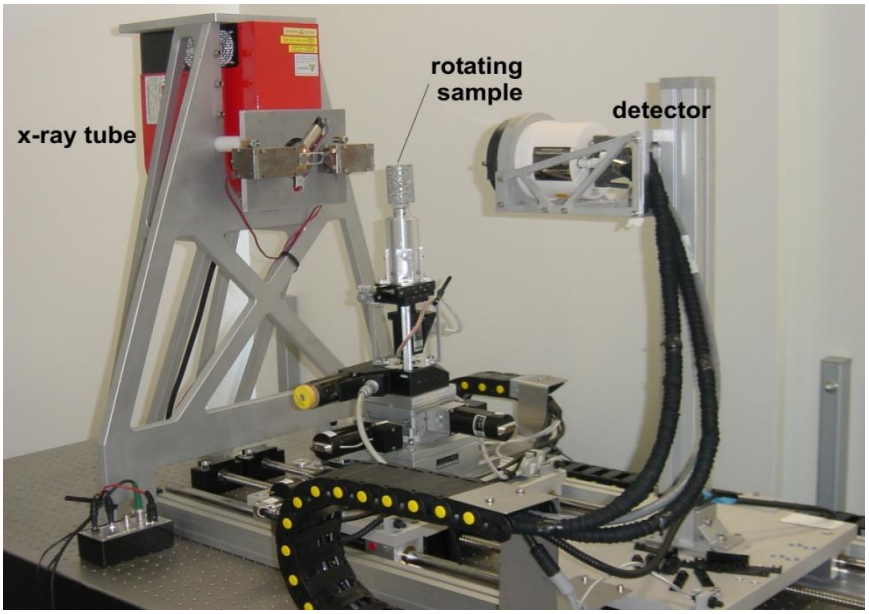

Figure 1. Industrial CT scanner based on fan-beam x-rays and linear detectors. The X-ray tube and detector have to move up and down to cover the entire object.

Note.- each student has to bring a zip file called *lastName_Name_P7.zip*, to the following address: *pablogtahoces@gmail.com.* The subject of the e-mail should be: IPBMA_P7. Inside the zip should be included:

- A jupyter notebook, showing how the software works (see the example).
- All the necessary files to verify the correct operation of the application.
- Maximum score: 25 points.
- The deadline will be: Sunday, January 30, 20:00.