Project on Processamento de Imagem e Visão (PIV)

A - Structure of the project

1. The project involves writing code that solves the problem described below and can be done in Matlab or Python (any other language may be accepted based on reasonable arguments). External libraries and toolboxes can be used though specific rules may be set for the submission (ex: no graphics).
2. The project code is submitted in two dates:
   I. First submission Friday, December 17-2021-17:30
   II. Second and final submission last day of 6th week (check FENIX (general info), it may change due to covid situation)

   The program will be shown running in the last week of classes
3. Grading: Grading is explained in the General Info (check fenix webpage)

B - General Description

Consider the scene depicted in the figure below. A camera is viewing a sheet of paper where somebody is writing. Sometimes people move the paper adjusting its position to a more comfortable posture or viewing angle. Likewise, the camera may move or change position. In summary, the basic setup is one camera viewing a sheet of paper where somebody is writing.

As the figure illustrates, the goal of PIV project is to detect the sheet of paper, compute the transformation between the paper and the camera, correct the perspective and generate an image as if the camera was viewing the paper from the top. Furthermore, the output image must be cropped showing nothing else than the paper.

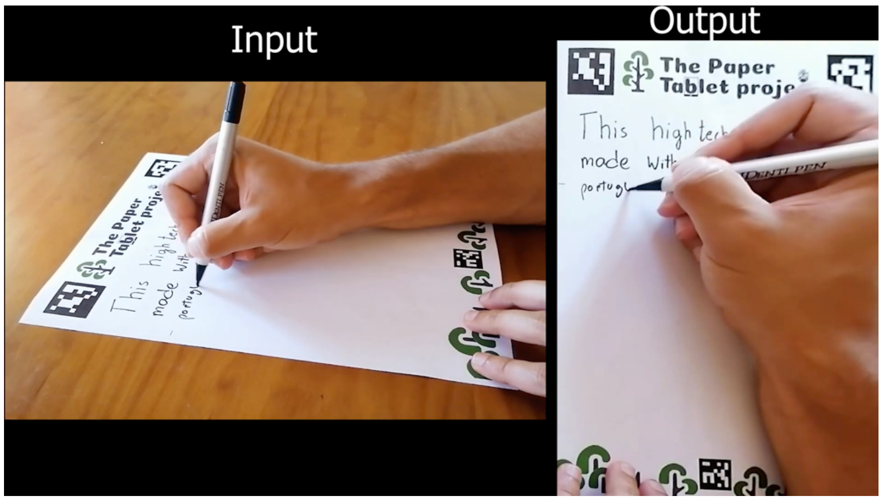


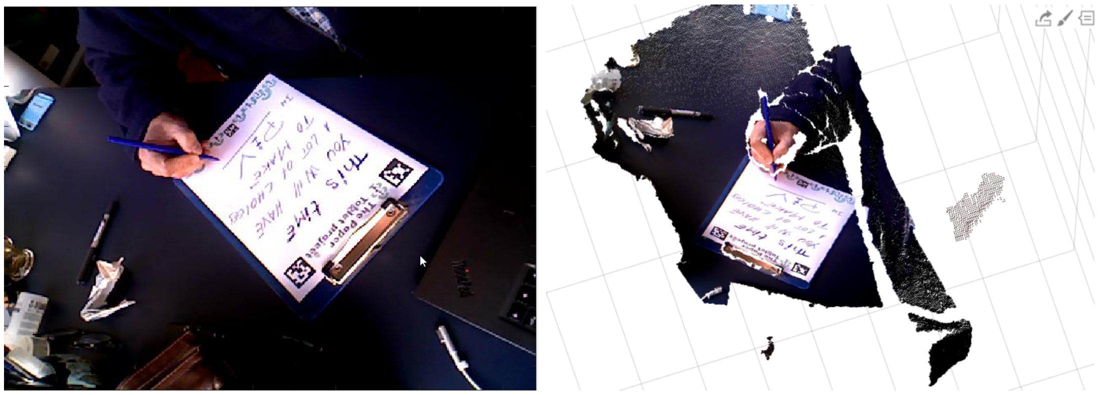
Figure 1: Left, the input from camera. Right, the output of processing the input image

We will call this project the "Paper Tablet" since the camera and the processing produce the same output of a digital tablet, for much less cost !

In fact, the paper tablet was initiated already by a former colleague of yours and the above sequence can be viewed here https://youtu.be/KQkpQ8dUKeQ

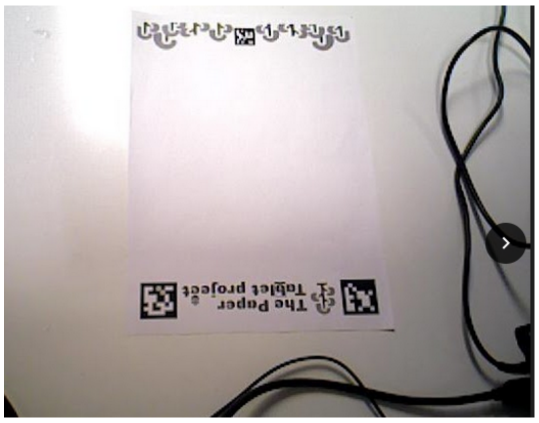
The project intent is to motivate students to apply the models and techniques taught in class and in most textbooks of the discipline. But the project is also meant to confront students with open problems whose solutions are not yet "established" and to the still infancy of computer vision. The above description describes the problem that all students must solve in order to pass the project.

The project full scenario is designed around a well structured sequence of topics that follow closely the topics explained in the lectures and that students must show some mastery. So, to the view of the above description we add another camera from a depth camera (Kinect like camera) that provides another RGB image and a depth image from which you can compute a 3D point cloud as shown in the figure below

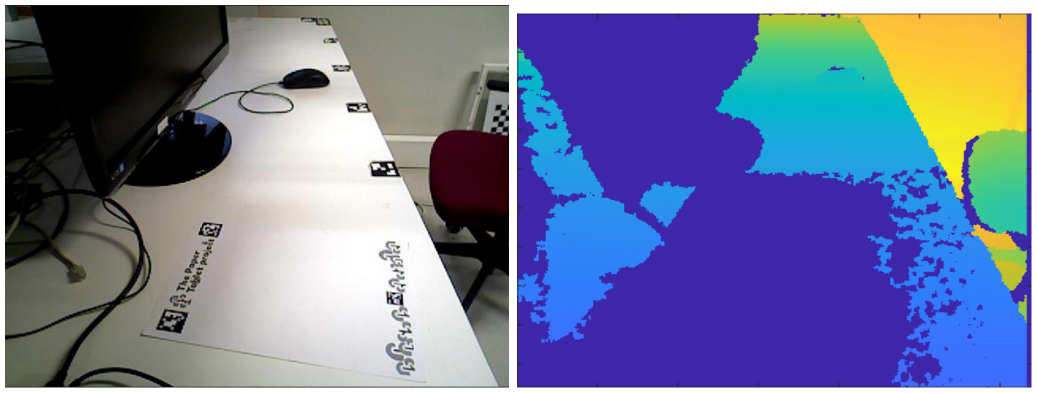


As you can see, to the rgb image on the left, the depth image allows the creation of a full 3D representation of each pixel in the image. Specifically the scenario will be as follows:

1 One RGB camera provides the picture of the tablet sheet



A second camera provides an RGB and a Depth image:



You will also have the calibration parameters from which you can compute the 3D point cloud.

Depending on the number of inputs your code must output an image of the sheet as described above using whatever images are available (one rgb camera only, one depth camera or one rgb and one depth camera).d

Summary:

Inputs: Image with the paper template, rgb or rgb and depth images

Output: a "tablet" like image with the contents of the writing

Scene: the paper can move or cameras can move or both !

C - Details and Valuation Criteria

To give a sense of priorities, the general criteria for final evaluation will rank the code according having into consideration the following principles:

1. Geometric correctness: the output must be geometrically correct. If the output image is twisted the project will be penalized. This means the transformations and robustness of its estimation are key aspects
2. Image "goodness" and robustness to artifacts. Provide faithfull and good quality output image (ex: remove hand or filter artifacts). We can not anticipate all good features but stuff that is not related to the topics of the course (ex: OCR) will have minimal "value".
3. Using less hardware. If you can do the same job with 2 rgb cameras is better than doing with one rgb and one depth (remove the depth information).
4. Speed. The performance of the code **will not be a** differentiating criterion, unless you do take absurd processing time. In other words your program can be slow, don't put too much effort in making it fast. The reason we penalize absurd execution times (need to define absurd!), is because it is sign of algorithm ignorance (exhaustive search, absurd number of iterations - put 1000000 when 99.9% accuracy requires 100 - or infinite loops).

To help you plan your effort we suggest

1. **Task 1 –** Receiving a folder with a set of images like the one of figure 1(left), generate the corresponding corrected images (that should be saved to the output folder) as in figure 1(right) where:

a. The paper template is known (there is an image file with the original image of the template).
b. The template has a set of Aruco Markers (link here) that can be detected with available software and from which pose can be extracted. In other words, the 4 corners of each marker and their pose (Rotation and Translation) relative to the camera can be obtained from a library. Task 1 amounts to compute the transformation that maps the viewed paper to the template and render it.

Precision is a key issue here, since it influences the quality of the rendering. Hand removal or beautification of the output is not a priority, the focus is on the correctness of the perspective transformation (geometry first!).

2. **Task 2 –** The same goal as Task 1 but without using the Aruco markers.
3. **Task 3 –** The same as Task 2 with rgb and depth cameras.
4. **Task 4** - Two rgb cameras

Task1 are minimum requirements for mid-period submission. Of course you can (and should) be ahead of this stage by the end of week 3 but submission 1 will be a binary decision : “you’re on track” or “you’re bound to fail”.

Task2 sets minimum requirements for final submission.

Task 3 and 4 is for creativity: You can use anything you want, hand detectors (ex: google mediapipe), background subtraction, anything you can put your hands (and brain) on. The goal is to produce a great output. In class we will teach, give some hints and provide some links to software.

The problem is complex and our goal is not to create a product !

Since this is the first time we run a 7 week course, we may adjust objectives along the way.

Protocol for testing and submitting your code

Python:

There must be a file **pivproject2021.py** and your project must run by issuing the following command:

python pivproject2021.py task path_to_template path_to_output_folder arg1 arg2

task: integer with values 1,2,3,4 depending on the task

path_to_template: a string with the path to an image file (ex. /tmp/template.jpg or c:\temp\template.jpg)

path_to_output_folder: a string with the path to a folder where output images are to be stored

arg1, arg2:

- if task = 1,2,3 , arg1 is the **path to the image input folder** (ex: /home/jpc/piv/rgbcamera) where input images are stored. Images are named rgb_number.jpg (ex rgb_0001.png) and depth images are named depth_number.png (depth images will be in png format). For these cases arg2 does not exist.
- if task=4, arg1 is the path to images of camera 1 and arg2 the path to images of camera 2 (all rgb images).

MATLAB:

You must submit a file named **pivproject2021.m** with a function called pivproject2021 with the following calling syntax:

>>pivproject2021(task, path_to_template ,path_to_output_folder , arg1, arg2)

input arguments are the same as above.

Libraries and extra code

You can use any extra files you wish both python or matlab. Matlab code should run in Matlab 2019b (all toolboxes).

You can use any library you want !

Image formats datasets

In the Gdrive PIV repository you can find datasets. RGB images are stored as jpg or png files and depth images are stored as png. The values are unsigned integers and represent depth in millimeters (you must convert to double and meters).

You have the templates and you can create your own datasets. For Task1 and 2 you are encouraged to create your own dataset (just print a page and record a video with your cellphone!). And share it with everybody (put it in GDrive).

Link to Github: https://github.com/pivist/PIV2021