ELEN-0016 – Computer Vision
Student projects 2022-2023

Prof. M. Van Droogenbroeck, R.Vandeghen

Version 2.2

Introduction

The aim of the project is to design methods, to understand its components, and evaluate the quality of the results for an application.

The application consists to manage the image/video acquisition process with a NVIDIA Jetson TX2, to detect and segment motion in the images/video stream when the camera itself is in motion and to produce a panoramic view of the imaged background.

Organization of the working teams

Each team consists of a maximum of 5 or 6 persons.

If possible, each team should be composed of students with different orientations (electronics – computer sciences – biomedical engineering). It is believed that such a team composition will enrich your experience and lead to the best results.

The teams must be constituted and the project will start **the 5th of October 2022**.

Provided material

- NVIDIA Jetson TX2 with a camera module
- Power supply adapter
- USB Micro-B to USB A cable
- USB Micro-B to Female USB A cable
- Antennas to connect to Wi-Fi enabled devices

You should check as fast as possible that you can use python3 and `OpenCV` correctly.

Task subdivision

The project is subdivided into two tasks (described in more details in the following sections):

Task 1 Main modules (building blocks) development (image/video acquisition, camera motion estimation, panoramic view and video database)

Task 2 Ball tracking and ball position in 3D.

Each task will give rise to a report.

1 Task 1: Main modules development

1.1 Description

This task is aimed at developing the main modules required to detect and estimate the motion of the camera and to produce a first version of the panoramic view. This first task is further subdivided into 4 sub-tasks described hereafter:

1.1.1 Sub-task 1.1: Image/Video acquisition

- You will work with gray-scale and/or color images. You should preferentially use gray-scale images for performance reasons. Processing color images is much more demanding than grayscale images. Nevertheless, if you think that using color images is worth the extra cost, you are allowed to do so and to discuss the advantages and drawbacks of the color images compared to grayscale images.
- You have to write and put into place a(several) program(s) written in python3, based on `OpenCV` library (linux) or any other python library, running on the Jetson TX2 platform.
- This(these) application(s) will be able:
  - either to read (and process) from the disk/memory any video sequences.
  - or to read (and process) any video stream acquired by the camera module embedded in the Jetson TX2.
- The image processing part is under the responsibility of the other sub-tasks.

1.1.2 Sub-task 1.2: Camera motion estimation

- You have to write and put into place a program written in python3 based on the `OpenCV` library (linux) or any other python library, running on the Jetson TX2 platform.
- The principle of camera motion estimation is to provide a continuous measure describing the movement of the camera. In this project, we expect you to provide the direction (expressed as an angle) of the camera compared to its resting or initial position.
- The only allowed movement of the camera is panning from left to right or from right to left between two predefined angles.
- You are allowed to reuse any algorithms available in the `OpenCV` library. We expect you to understand not only the theory behind the algorithms but also their advantages, drawbacks and their internals.

1.1.3 Sub-task 1.3: Panoramic image construction - First version

- You have to write and put into place a program written in python3 based on the `OpenCV` library (linux) or any other python library, running on the Jetson TX2 platform.
- Based on the estimation of the camera motion (see 1.1.2), you have to build a panoramic view of the background and a visualization of where the camera is situated in this panoramic view. An example is illustrated in Figure 1. You should create this panoramic image, as much as possible, without the foreground objects in it. But, at this stage, without the detection of the moving objects, we do not expect you to completely remove the foreground (moving objects) from the panoramic view.

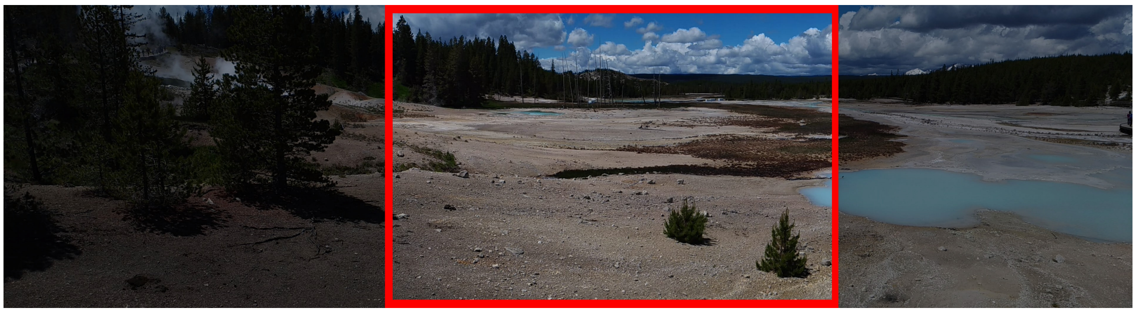

Figure 1: Panoramic image with the current image highlighted.

- As explained later (see 1.2), you will acquire two different types of sequence; the "working" sequences with moving objects and the corresponding "reference" sequences without any moving objects. You have to produce a panoramic view for both types of sequence (see 1.2 for details).
- You are allowed to reuse any algorithms available in the `OpenCV` library. We expect you to understand not only the theory behind the algorithms but also their advantages, drawbacks and their internals.

1.1.4 Sub-task 1.4: Image database

- In this sub-task, you will build a video database containing objects, persons and camera motion for different scene configurations.
- The videos will be acquired with the camera module of the Jetson TX2.
- The details concerning the videos and their formats are described hereafter.

1.2 Images database

Image size: The image size must be 1280×720 (width $\times$ height) pixels. But, for performance reasons, you are allowed to process reduced versions of the images.

Frame rate: You will choose the frame rate of 25 Frames Per Second (FPS) and the duration of the sequences will be 1 min, so 1500 frames exactly in total.

Color: You will record colored images, even if you prefer to process grayscale images.

Number of working images/sequences: The video database must contain 2 video sequences saved as a sequence of images. You should make the scenes as dynamic as possible with up to three persons in total in the sequence. One sequence should be recorded indoors and one outdoors. In addition to displaying moving persons, the 2 sequences have to contain some other moving objects that are not human beings (cars, animals, balls, ...)

Number of references images/sequences: For each video sequence, you have to provide a companion sequence, which will serve as a reference to validate your motion detection algorithm and background panoramic view. These sequences will be acquired at the same time (before or after) and in the same conditions than the corresponding video sequences with moving objects. During these companion sequences, the camera will pan from left to right then back from right to left. The reference sequences must not contain any moving objects (just the background). The duration of these reference sequences must be 20 sec, acquired at the same frame rate and resolution than the regular sequences.

Image file name format: The video sequences will be stored as sequences of individual image files with the following syntax: `"img_%1d_%1d_%04d.jpg"`. The first '`%1d`' one-digit number is the number of your group (from `1` to `7`). The second '`%1d`' one-digit number is the number of the sequence (from `1` to `2`). The '`%04d`' four-digits number is the number of the recorded image within the sequence, padded to the left with '0' (from `'0000'` to `'1499'`). So the file names for the second sequence of the group 3 will be : `img_3_2_0000.jpg`, ..., `img_3_2_1499.jpg`.
The naming convention for the reference sequences is the same as for the regular sequences excepted that the 'img' part of the name has to be replaced by 'ref'. So, the reference images for the second video of the third group will be named `ref_3_2_0000.jpg`, ..., `ref_3_2_0499.jpg`.

Panoramic view for the reference sequences: For each reference video sequence, you must provide the corresponding panoramic image. This panoramic image must be saved in four non overlapping parts. Each part is a 1280×720 image named `"pan_%1d_%1d_%1d.jpg"`. The first '`%1d`' one-digit number is the number of your group (from `1` to `6`). The second '`%1d`' one-digit number is the number of the sequence (from `1` to `2`). The third '`%1d`' one-digit number is the part number in the panoramic image (from 0 to 3). As an example, the 4 images of the panorama for the second sequence of the group 3 will be : `pan_3_2_0.jpg`, ..., `pan_3_2_3.jpg`.
If the range of your panoramic view doesn't fill completely the four images, the panorama must start with the image `"pan_%1d_%1d_0.jpg"` you must pad the right part of the last images panorama with zero (black) and you must provide the exact width of your panorama in a 'README' file (must not be larger than 5120).

For the project, you will use the sequences of all the different groups. Therefore, your sequences are due for **the 12th of October** on the submission platform.

1.3 Reporting

The report is due for **the 26th of October** on the submission platform.
The report will contain:

- A document with:
  - A short presentation of the whole task.
  - The contribution of each student of the team with respect to the 4 sub-tasks. Several students may work on the same task and a student may work on several tasks. But we expect the work to be reasonably and fairly distributed.
  - A description of your image database.
  - For each modules:
    - A short description of the implemented algorithm, its advantages and drawbacks.
    - A short note on the implementation.
    - A short description of the validation test for the implementation.
  - Please, it is mandatory to limit the size of your document to **6 pages maximum** (including title page, figures, images, tables, graphics, etc.). Pages beyond 6 pages will be discarded.
- An optional video.
- The image database.
- The code of the applications/modules produced by the four/five students in the team **with a clear description** (README) of how to use the code (install, run and test).

2 Task 2: Motion and ball detection, panoramic background, performance assessment.

2.1 Description

2.1.1 Sub-task 2.2: New image database

- For the second part of the project, you will need to acquire new frames, where at least 2 persons are throwing a ball at each other (we will supply you the balls, that have two fixed physical sizes).
- For the database, you need to follow the same instructions as for Section 1.2. However, you are allowed to record in different places.

2.1.2 Sub-task 2.2: Motion detection

- You have to write and put into place a program written in python3 based on the `OpenCV` library (linux) or any other python library, running on the Jetson TX2 platform.
- The principle of motion detection algorithms, also called *background subtraction*, is to classify pixels in two categories, whether this pixel belongs to an object in motion, called foreground objects or the background. The goal, and the expected result, is to produce a segmentation map of the moving objects in each images of the new video sequences.
- You have to use the results of the previous sub-tasks (camera motion estimation of the working video sequence, see 1.1.2) to compensate for the movement of the background.
- In real situations, you will not have access to the reference background! So, the panoramic backgrounds you obtained with the new reference video sequences will be used only for validation purposes. Here, the motion detection must be achieved with the information you are able to extract from the working video sequences only!
- You are allowed to reuse any algorithms available in `OpenCV` library. We expect you to understand not only the theory behind the algorithms but also their advantages, drawbacks, and their internals. You can pick ideas from existing background subtraction algorithms in order to implement your panoramic background subtraction algorithm, using for example the numpy library. You can also incorporate some learning techniques using for example priors on the type of object suspected to be in motion.

2.1.3 Sub-task 2.3: Enhanced panoramic image - Second version

- You have to write and put into place a program written in python3 based on the `OpenCV` library (linux) or any other python library, running on the Jetson TX2 platform.
- Based on the moving objects detection (see 2.1.2), you have to produce a better panoramic view for the background of the working video sequences, without any moving foreground objects and a visualization of where the camera is situated in this panoramic view.
- You are allowed to reuse any algorithms available in `OpenCV` library. We expect you to understand not only the theory behind the algorithms but also their advantages, drawbacks, and their internals.

2.1.4 Sub-task 2.4: Person detection and ball detection

- You have to write and put into place a program written in python3 based on the `OpenCV` library (linux) or any other python library, running on the Jetson TX2 platform.
- You have to distinguish (classify) moving persons and balls from other moving objects. We expect you to display a rectangular bounding box around the detected persons for each image of the working video sequences. You also need to display a bounding box around the ball(s), as well as keeping track of its position (in a 3D space) for each frame of the sequence. If the segmentation mask of a human being is separated in several parts, you should try to fused them together as much as possible. On the other side, if several people are grouped together in the same segmentation mask, you should try to separate them as much as possible.
- You are allowed to reuse any algorithms available in `OpenCV` library. We expect you to understand not only the theory behind the algorithms but also their advantages, drawbacks, and their internals.

2.1.5 Sub-task 2.5: Performance assessment.

This sub-task is aimed at assessing the performances of the algorithms developed during the course of the project:

1. The camera motion estimation module (optional for Part 2).
2. The moving object detection module. To assess the performances of this module you are going to build by hand the true segmentation mask for a few images in the working image sequences. See 2.2 for the details.
3. The panoramic image construction module.
4. The person/ball detection module. To assess the performances of this module you are going to build by hand the true bounding boxes for the moving persons/ball in the annotated working images. See 2.2 for the details.

You should study, clearly design and describe how you are going to evaluate the performances (methodology, typology, kind of evaluation task, criteria/scores, reference data, ...) It can be a qualitative assessment for the third sub-task (panoramic image construction) and it should be a quantitive assessment for the three other tasks. If you come up with a binary classification problem, pay attention to correctly define and use the four quantities (TP, FP, FN, TN) of the confusion matrix.
For the evaluation sequence, you are asked to provide for each frame, in a json file:

1. The 2D coordinates (x, y) of moving persons.
2. The 3D coordinates (x, y, z) of moving balls.

The x and y coordinates are relative to each frame, and the z value is the distance between the Jetson and the ball. For each prediction, you also need to provide the score associated to the prediction (between 0 and 1).You are asked to follow the json template in the GitHub.

2.2 Image annotation

1. Each person in a group must annotate 10 images.
   (a) The groups with 5 (resp. 6) persons will provide the segmentation masks and the bounding boxes of moving persons for 50 (resp. 60) images

   (b) There must be 25 (resp. 30) images in the indoor sequence and 25 (resp. 30) images in the outdoor sequence.
   (c) The 25 (resp. 30) annotated images per sequence must be adequately distributed within the sequence and should demonstrate a maximum of different situations.
   (d) The first annotated image MAY NOT come before the image 100 of a sequence, in order for the algorithm to initialize itself properly, if needed.

2. The segmentation mask is a grayscale image containing binary information.
   (a) The background is coded with 0 and the foreground (moving object) is coded with 255.
   (b) The size of the segmentation masks is the same than the size of the images (1280 x 720).
   (c) The name of the segmentation mask for the image 'img_5_2_0523.jpg' will be 'seg_5_2_0523.png': you just have to replace the 'img' part by a 'seg' part and the 'jpg' format by a 'png' format. Pay attention to use a lossless coding for the segmentation masks.

3. The bounding boxes will be saved in one text file per sequence.
   (a) The text file corresponding to the images in the sequence 'img_2_1_XXXX.jpg' will be 'box_2_1.txt
   (b) The text files will contain ONE bounding box descriptor per line, as well as the the class of the object (person/ball).
   (c) The lines in the text files will contain 5 comma-separated fields with the following format:
   img_6_1_0783.jpg, <x1>, <y1>, <x2>, <y2>,<class>
   where <x1>, <y1>, <x2>, <y2> are numerical (integer) values, <class> is the name of the class and 'img_6_1_0783.jpg' is the name of the image of the bounding box.

4. You may use 'Gimp' or any other similar tool to build manually the segmentation masks and to obtain the parameters of the bounding boxes.

5. Each group must provide the 50 (resp. 60) segmentation masks and the two text files by mail at r.vandeghen@uliege.be by Friday, the **9th of November**, no later than **18h00.**

2.3 Poster presentation

The poster presentation will be held the last Wednesday, that is the 14th of December. More information about the schedule will follow. The presentation will last 15 minutes. You will have the time to present your poster and propose a demo of your project, followed by some questions.We will print your poster by ourselves on a A0 paper, meaning that the pdf file of your poster is due by Monday, the 12th of December before 14h00.