ELEN-0016 – Computer Vision
Student projects 2018-2019

Prof. M. Van Droogenbroeck, P. Latour, A. Cioppa, and M. Fonder

Version 1.0

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 4 or 5 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 on **Friday, the 12 October 2018, at 12h30**.

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

Obviously, the very first step, after receiving your material, is to configure your NVIDIA Jetson TX2 system.

To do so, you will need a keyboard, a mouse and a screen (with a HDMI input).

Do it very quickly to be sure that there is no problems with the material.

Configuring the NVIDIA Jetson TX2

You will find some documentation concerning the NVIDIA Jetson TX2 here:

NVIDIA Jetson TX2 Documentation

Group	Members	Registered
1	KEUTGEN Pierre	Yes
	MASSART Lara	Yes
	PROBST Jérémy	Yes
	VALLOT Arnaud	Yes
2	BINDELS Quentin	Yes
	GRODENT Antoine	Yes
	MOUREAU Céline	Yes
	PIRE Renaud	Yes
3	DERKENNE Donovan	Yes
	NOIRHOMME Maxime	Yes
	POUMAY Judicaël	Yes
	SACRÉ Loïc	Yes
	VANDENBERGH Julien	Yes
4	BAJOIT Romain	Yes
	DE COOMAN Julien	Yes
	HOUART Robin	Yes
	SADZOT Antoine	Yes
5	FOLON Nora	Yes
	HENRY Maxim	Yes
	HORMAN Maxime	Yes
	LECOCQ Noémie	Yes
	STAPPERS Michaël	Yes
6	BEN MARIEM Sami	Yes
	CHIODO Adrien	Yes
	DERROITTE Natan	Yes
	GIRINEZA Guy	Yes
	TESTOURI Mehdi	Yes
-	BOURDOUXHE Alexandre	Yes
-	SONG Xuzheng	Yes

Table 1: Table of the groups

1. Powering-up the device:

   Connect a USB keyboard and mouse to the device (using the micro-usb adaptor for one of the two device or a USB hub)

   Connect a screen to the device via the HDMI port.

   Connect the AC adapter to the power connector of the device.

   Press the power-up button (fourth button from the left with the name POWER BTN)

   A basic terminal will be displayed on the screen

2. If needed, configuring the development card

   cd "${HOME}/NVIDIA-INSTALLER"

   sudo ./installer.sh

   password : nvidia (attention "nvidiq" for azerty keyboards)

   reboot

3. If needed, instaling JetPack

   Install JetPack 3.3 following the instructions provided on: https://developer.nvidia.com/embedded/downloads#

   You will need a second computer with a linux distribution to do so. We also recommend using a hub to connect the machine and the Jetson to the internet and one another.

4. Updating the device

   sudo apt-get upgrade

   sudo apt-get update

   sudo apt install python3-pip python3-dev

   pip3 install numpy

5. Install keras with tensorflow-gpu

   pip3 install –extra-index-url=https://developer.download.nvidia.com/compute/redist/jp33 tensorflow-gpu

   sudo apt-get install libblas-dev liblapack-dev libatlas-base-dev gfortran

   sudo apt-get install libhdf5-dev

   sudo apt-get -y install libhdf5-serial-dev hdf5-tool

   sudo ln -s /usr/lib/aarch64-linux-gnu/libhdf5_serial.so.10 /usr/lib/aarch64-linux-gnu/libhdf5.so $

   sudo ln -s /usr/lib/aarch64-linux-gnu/libhdf5_serial_hl.so.10 /usr/lib/aarch64-linux-gnu/libhdf5_hl.so

   sudo pip3 install h5py

   sudo pip3 install scipy

   sudo pip3 install keras

6. Install OpenCV:

   cd ~

   git clone https://github.com/jetsonhacks/buildOpenCVTX2.git

   cd buildOpenCVTX2

   bash buildOpenCV.sh (duration +- 1 hour)

7. Install jupyter notebook

   sudo pip3 install pyzmq==17.0.0

   sudo pip3 install jupyter

8. Sanity checks

   Check that openCV and keras are correctly installed. To do so, open a python terminal and run the appropriate import commands, example:

   python3

   import keras

   import cv2

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 Motion detection / estimation with compensated camera motion and performance assessment

Each task will give rise to a report and a presentation.

Schedule

Date	Time	Milestones	Room
Friday, 21/09/2018	8h30-12h30	Theory	2.93
Friday, 28/09/2018	8h30-12h30	Theory + Student project presentation	2.93
Friday, 5/10/2018	8h30-12h30	Theory + Groups for the project	2.93
Friday, 12/10/2018	8h30-10h30	Exercises	2.93
Friday, 19/10/2018	8h30-12h30	Theory + Exercises	2.93
Friday, 26/10/2018	8h30-12h30	Theory + Exercises	2.93
Friday, 02/11/2018			
Friday, 09/11/2018	8h30-12h30	Theory + Project	2.93
Friday, 16/11/2018	8h30-12h30	Theory + Project	2.93
Wednesday, 21/11/2018	17h00	Report of the first part of the project	R87a
Friday, 23/11/2018	8h30-12h30	Presentation of the first part of the project	2.93
Friday, 30/11/2018	8h30-12h30	Theory	2.93
Friday, 07/12/2018	8h30-12h30	Theory + Project	2.93
Friday, 14/12/2018			
Wednesday, 19/12/2018	17h00	Report of the second part of the project	R87a
Friday, 21/12/2018	8h30-12h30	Presentation of the second part of the project	2.93

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 python 3, based on `OpenCV` library (linux) and/or Keras deep learning 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 python 3, based on `OpenCV` library (linux) and/or Keras deep learning 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.
- Concerning the whole project, we do not expect you to use deep learning techniques if you consider that they are unnecessary to solve the problem, completely unsupervised techniques can have lots of interest as well. Be careful that deep learning often comes at the cost of a high computation time.
- 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.

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

- You have to write and put into place a program written in python 3, based on `OpenCV` library (linux) and/or Keras deep learning 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.
- 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 `OpenCV` library. We expect you to understand not only the theory behind the algorithms but also their advantages, drawbacks and their internals.

Figure 1: Panoramic image with the current image highlighted.

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 × 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 than 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 `6`). 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_?_?_0.jpg', you must pad the right part of the four 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).

1.3 Reporting

The report is due for **Wednesday, the 21nd of November no later than 17h00** in the R87a room.

The report will contain:

- A short presentation of the whole task.
- The contribution of each student of the team with respect to the 5 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.
- The code of the applications/modules produced by the four/five students in the team (on a USB key).
- A description of how to use the code (install, run and test).
- A description and a few examples of your image database and the corresponding annotated camera and person motion.
- 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 report to **6 pages maximum** (including title page, figures, images, tables, graphics, etc.). Pages beyond 6 pages will be discarded.

1.4 Presentation and demo

The presentation will take place on **Friday, the 23th of November starting at 8h30** in the 2.93 room.

You are strongly encouraged to demonstrate your applications/modules during the presentation.

You will have no more than 20 minutes to present the project, including the time for a visual demonstration of your application.

All the members in the team have to present a part of the project.

The schedule for the presentation is

Group	Members	Time
1	KEUTGEN Pierre, MASSART Lara, PROBST Jérémy, VALLOT Arnaud	8h30-8h50
2	BINDELS Quentin, GRODENT Antoine, MOUREAU Céline, PIRE Renaud	9h00-9h20
3	DERKENNE Donovan, NOIRHOMME Maxime, POUMAY Judicaël, SACRÉ Loïc, VANDENBERGH Julien	9h30-9h50
4	BAJOIT Romain, DE COOMAN Julien, HOUART Robin, SADZOT Antoine	10h15-10h35
5	FOLON Nora, HENRY Maxim, HORMAN Maxime, LECOCQ Noémie, STAPPERS Michaël	10h45-11h05
6	BEN MARIEM Sami, CHIODO Adrien DERROITTE Natan, GIRINEZA Guy, TESTOURI Mehdi	11h15-11h35

2 Task 2: Scene description, performance assessment, and modules integration

The following part is a first draft that will be reviewed and possibly modified.

2.1 Description

2.1.1 Sub-task 2.1: Motion detection

- You have to write and put into place a program written in python 3, based on `OpenCV` library (linux) and/or Keras deep learning 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 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 reference video sequences will be used later, 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.

2.1.2 Sub-task 2.2: Enhanced panoramic image - Second version

- You have to write and put into place a program written in python 3, based on `OpenCV` library (linux) and/or Keras deep learning library, running on the Jetson TX2 platform.
- Based on the moving objects detection (see 2.1.1), 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.

2.1.3 Sub-task 2.3: Person detection

- You have to write and put into place a program written in python 3, based on `OpenCV` library (linux) and/or Keras deep learning library, running on the Jetson TX2 platform.
- You have to distinguish (classify) moving persons 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. 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.

2.1.4 Sub-task 2.4: 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.
2. the moving object detection module.
3. the panoramic image construction module.
4. the person detection module.

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 first three sub-tasks and it should be a quantitive assessment for the fourth one.

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.

2.2 Reporting

The report is due by Wednesday, the **19th of December**, no later than **18h00** in the R87a room.

The report could contain:

- A short presentation of the whole task.
- The contribution of each student of the team.
- A clear explanation of the assessment methodology and choices.
- Your results and discussion concerning the performances of:
  - the camera motion estimation module.
  - the object motion detection module.
  - the panoramic image construction module.
  - the person detection module.
- Your conclusions, comments and ideas for further enhancements of the application.
- The final code of your application integrating the work of the four/five students in the team, with a quick description of the algorithms implemented (you may make references to your previous reports) and how to use the code (install, run, and test the application on the NVIDIA Jetson TX2).

Please limit the size of your report to a strict **10 pages maximum**.

2.3 Presentation and exam

The presentation will take place on **Friday, the 21th of December starting at 8h30** in the R.157 room.

It will be followed by the theoretical test (each group on turn and separately), in the R.21 room.

You are strongly encouraged to demonstrate your final application during the presentation. The driving simulator will be available for your live demo.

You will have no more than 20 minutes to present the project, including the time for a visual demonstration of your application.

All the members in the team have to present a part of the project.

The schedule for the presentation is:

Group	Members	Project, R.157	Exam, R.21
6	BEN MARIEM Sami, CHIODO Adrien DERROITTE Natan, GIRINEZA Guy, TESTOURI Mehdi	8h30-8h50	9h00-9h25
5	FOLON Nora, HENRY Maxim, HORMAN Maxime, LECOCQ Noémie, STAPPERS Michaël	9h00-9h20	9h30-09h55
4	BAJOIT Romain, DE COOMAN Julien, HOUART Robin, SADZOT Antoine	9h30-09h50	10h00-10h25
3	DERKENNE Donovan, NOIRHOMME Maxime, POUMAY Judicaël, SACRÉ Loïc, VANDENBERGH Julien	10h15-10h35	10h45-11h10
2	BINDELS Quentin, GRODENT Antoine, MOUREAU Céline, PIRE Renaud	10h45-11h05	11h15-11h40
1	KEUTGEN Pierre, MASSART Lara, PROBST Jérémy, VALLOT Arnaud	11h15-11h35	11h45-12h10