Master ST Computer Science option ISM **University Bordeaux**
Faculty S&T
UE VNAC academic year 2016 – 2017

Part 1. Lab 2
(4 hours, evaluated)

Motion estimation in videos

Purpose: Code and compare various motion estimation algorithms using C++ API of OpenCV.

Work environment: Linux, C++, OpenCV, ffmpeg, Gnuplot.

Get video files in lower resolution from:
*h**ttp://dept-info.labri.fr/~mansenca/VNAC2016/MotionEstimationDataSet/***

1. Motion Estimation with block-matching algorithm

The algorithm to implement is the full-search block-matching method. The motion will be computed with pixel accuracy. The direction of estimation is **backward**: between the current video frame I(t) and a previous video frame I(t-Δt).
The criterion to minimize will be MSE. We consider square blocks (of size 4, 8 or 16).

Pseudo-code for algorithm:
Input: (x,y), F, I(t), I(t-Δt)
Output: (dx, dy)

For a block (at position (x;y)):

```
E0 = MAX_INT
dx0 = -F/2
dy0 = -F/2
For dy from -F/2 to F/2 with step1
        For dx from -F/2 to F/2 with step1
                E=CRITERION(dx, dy, x, y, I(t), I(t-Δt))
                if E < E0
                        dx0 = dx
                        dy0 = dy
                        E0 = E
                endif
        endfor
endfor
```

Δt, block size and F should be parameters of your program.

You will compute the motion compensated image:
$$\tilde{I}(\vec{p},t) = I(\vec{p}+\vec{d},t-\Delta t)$$
and the motion compensated error image:
$$E(\vec{p},t) = min(255, max(0, I(\vec{p},t) - \tilde{I}(\vec{p},t) + 128))$$

You will have:

- to compute the FD image and the motion compensated error image
- to display the motion vectors

- to compute the MSE between the current image and the compensated image, the PSNR, the entropy of the current image, the entropy of the error image, and plot the curves of these measures
- to display the motion vectors
- to test with at least $\Delta t=1$, block size of 8, F=32, on Birds_720.MOV
- to write a small report with these curves and your analysis of these results

Code is provided to compute MSE, PSNR, entropy and to display the motion vectors.

2. Motion Estimation with Horn&Schunck algorithm

You have to use OpenCV Horn&Schunck implementation to compute the motion estimation.
Motion estimation must be done:
2.1) in mono-resolution: motion field is initialized to 0
2.2) in multi-resolution: motion field at one level is initialized by the projection of the motion field at the previous level

OpenCV (version 2.4.x) provides the function *cvCalcOpticalFlowHS()* in the *legacy* module.
Warning: the legacy module does not work in OpenCV version 3.x.
Warning: this function uses the old C API of OpenCV.
You can convert a *cv::Mat* to an *CvArr** with the following code:

```
cv::Mat m;
IplImage im(m);
CvArr *am=&im;
//they all share the same underlying data
```

You must send a report with the same work than for the block-matching algorithm and compare your results between mono and multi-resolution, and also with the block-matching results.

3. [Optional] Motion Estimation with Farneback method

OpenCV provides an implementation of Farneback's algorithm, via the *calcOpticalFlowFarneback()* function in the *video* module.

Compute the motion estimation:
3.1) in mono-resolution
3.2) in multi-resolution

Compare your results to the previous methods.

Send an archive with your code and report (with curves and discussions), named MYNAME_Lab2.tar.bz2, to : boris.mansencal@labri.fr