M2 E3A MMVAI
Lab : TP2 optical flow

Image and video processing - M2 E3A MMVAI - UEVE/Université Paris-Saclay

Exercise 1 : Optical flow

1. Study carefully the paper : "Determining optical flow" by Horn and Schunk.
2. Explain how spatial gradients are calculated.
3. Explain how the time gradient is calculated.
4. What Laplacian approximation of u and v is used ?
5. Explain the principle of the optical flow calculation method as well as its limitations.
6. Explain how the different parameters of the method can be chosen.
7. Program the algorithm in Matlab.
8. Test your algorithm on the sequence of your choice :
   — First on the Road sequence (without ground truth).
   — Then on one of the Middlebury sequences provided (**SEQUENCES_GT** directory).
9. Display the optical flow (see appendix) : 1/ as a grayscale image (one image per component), 2/ by drawing vectors using the function **quiver_uv** (see appendix) and 3/ in color (**computeColor** function in the **CODES** directory). In order to visualize the ground truth (directory **FLOTS_GT**), you have, in the directory **CODES**, a certain number of functions allowing the manipulation of the .flo format.
10. In order to evaluate the precision of the estimated optical flow without having ground truth, it is possible to realign the image $t + dt$ using the estimated vectors and compare the new image thus obtained with the image t . Write a function calculating the residual error thus obtained. Where are the biggest errors ? Conclude. Residual error at each point :

$$varepsilon_{x,y} = |I(x, y, t) - I(x + dx, y + dy, t + dt)|$$

11. Suggest another way to calculate spatio-temporal gradients. Compare the results obtained with the original method.
12. The discontinuities of the optical flow should correspond to those of the image and of the borders separating the different objects. Propose a way to determine the boundaries between different movements and compare with the real boundaries of objects (result of contour detection).
13. When we have a ground truth, the evaluation consists in comparing the optical flows obtained with those of the ground truth based on the AEPE and AAE indicators. Set up this type of assessment on the images in the Middlebury database (https ://vision.middlebury.edu/flow/).
    — Colored representation of Middlebury. Hue : orientation, Saturation : norm.
    — $\epsilon_{AE} = \arccos\left(\frac{\mathbf{w}^T\tilde{\mathbf{w}}}{\|\mathbf{w}\|\tilde{}\mathbf{w}\|}\right)$ with $\mathbf{w}$: ground truth flow and $\tilde{\mathbf{w}}$: estimated flow.
    — $\epsilon_{EPE} = \|\mathbf{w} - \tilde{\mathbf{w}}\|$
    — These are then the average of these errors which are calculated on the whole image to obtain AEPE (Average End Point Error) and AAE (Average Angular Error).

14. In the original Horn and Schunck method, the iterative process starts from an estimate of the velocity vectors equal to zero. Modify the algorithm so that the estimate is equal to the solution found at the previous time at the same point.

Exercise 2 : Temporal and spatial post-filtering of the optical flow

1. Propose an algorithm to filter out erroneous vectors using spatial filtering.
2. Propose an algorithm to filter out erroneous vectors using temporal filtering.
3. Compare the two algorithms.

Appendix : Notes on the display of the optical flow

— To view the results as vectors, you can use the matlab "quiver" function as shown below.

```
function quiver_uv(u,v)

% Resize u and v so we can actually see something in the quiver plot
scalefactor = 50/size(u,2);
u_=scalefactor*imresize(u,scalefactor,'bilinear');
v_=scalefactor*imresize(v,scalefactor,'bilinear');

% Run quiver taking into account matlab coordinate system quirkiness
% and scaling the magnitude of (u,v) by 2 so it is more visible.
quiver(u_(end:-1:1,:),-v_(end:-1:1,:),2);
axis('tight');
```

— It is also possible to visualize the component u and the component v separately by considering them each as images in levels of gray with some approximations and rounding.

— Finally, a common representation is to display the result in color, taking into account the modulus and the orientation of the vector estimated at each point. Use the **computeColor** function for that.

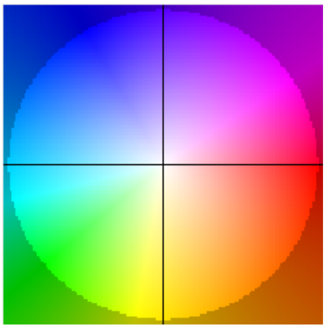

Figure 1 – Color code used for each module and each orientation of the estimated velocity vectors (in polar coordinates) for visualization on a single color image.