Optical flow

Radoslav Atanasoski

I. Introduction

In this assignment, two methods are used for estimating the optical flow in successive images: Lukas-Kanade and Horn-Schunck. Both algorithms follow the brightness constancy and small displacement assumptions. However, for estimating the flow vectors, Lucas-Kanade uses least-squares, looking for a single vector that minimizes an energy function in a small neighborhood of a pixel, while Horn-Schunck uses variational calculus, looking for a function that minimizes the error over the entire image. Furthermore, a pyramidal Lucas-Kanade is implemented, improving the motion flow estimation. And also, Harris corner detection is evaluated in combination with Lucas-Kanade to compare for improvements.

II. Experiments

A. Lucas-Kanade

The results we get with Lucas-Kanade highly depend on the parameters we use. For our basic implementation, we use just two:

- N - Neighborhood of the kernel (NxN).
- Sigma - Parameter that determines the amount smoothing when using Gaussian blur.

For the random noise test image, the first image is just noise randomly generated, and the second image is the first image rotated by 1 degree. Because of the small motion (rotation by 1 degree), we can use a smaller neighborhood. However, due to the noise, a larger neighborhood is required for smoother optical flow.

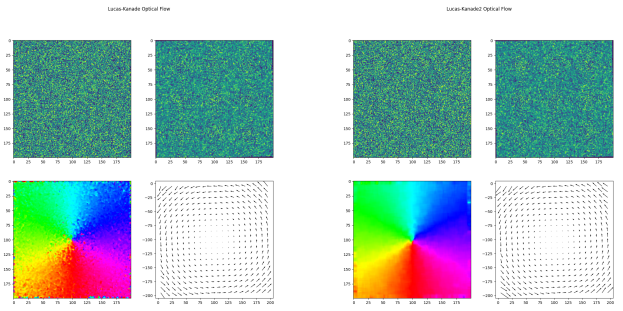

Figure 1: Lucas-Kanade on random noise image rotated by 1 degree using a neighborhood of 3 on the left image, and a neighborhood of 11 on the right image.

We can see the improvement of a larger neighborhood in figure 1. We can see that in both cases, the flow in the center of the images is smaller than the flow at the edges, as when rotating the image, the pixels at the edges move way more than those in the center.

Furthermore, different values for sigma were tested (0.5, 1, $\sqrt{2}$, 1.5, 2, 3, 5), where the most optimal one seemed to be 1. Using a higher sigma, results in over-smoothing the images, thus losing important details of the image and causing less accurate motion estimates.

1) Harris response: Because of the small displacement assumption, Lucas-Kanade will fail to accurately estimate the flow on places with large motions. Furthermore, the algorithm relies on the presence of distinctive feature points with high texture to track motion. In regions of the image with low texture, the algorithm may fail to track feature points accurately, leading to inaccurate motion estimates. Thus, by using Harris response to omit the non-edges in the image, we can get rid of noisy background flows, as shown in figure 2.

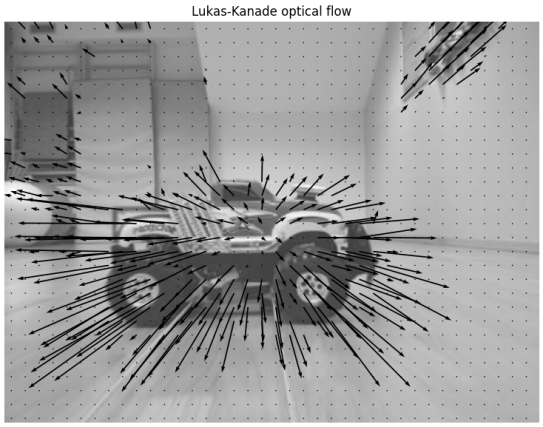

Figure 2: Lukas-Kanade with Harris response, using a neighborhood of 11.

2) Pyramidal implementation: The pyramidal implementation is used to address two key challenges: the aperture problem and the sensitivity to noise and small-scale details. It is a powerful technique that allows optical flow algorithms to take into account motion at multiple scales, which improves their accuracy and robustness.

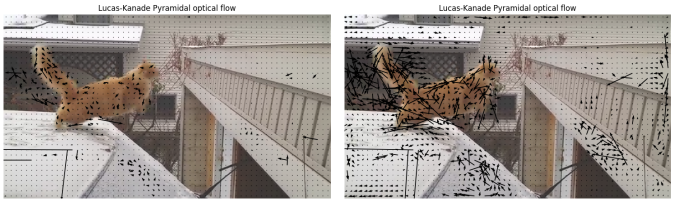

Figure 3: Lucas-Kanade Pyramidal example using a neighborhood of 3, with 2 levels on the left image, and 6 levels on the right image.

We can see in figure 3 that by increasing the number of levels, we get a more detailed flow. Furthermore, the small motion assumption is often violated, thus the pyramidal implementation is often necessary. Also, when skipping a frame, the non-pyramidal implementation will fail to estimate the flows.

B. Horn-Schunck

For this algorithm, also two parameters are used:

- Number of iterations
- Lambda - A regularization parameter that is used to control the smoothness of the estimated flow.

Using a higher lambda can result in losing small details, and using a small lambda can result in overly detailed flow with noise and artifacts. Thus after testing with different values, choosing one over the other really didn't make much difference, hence a value of 0.5 is used throughout.

However, the number of iterations drastically affect the flow of the algorithm. The higher the number of iterations, the smoother and more accurate the flow is, but the slower the algorithm. Hence, this parameter determines the trade-off between accuracy and speed.

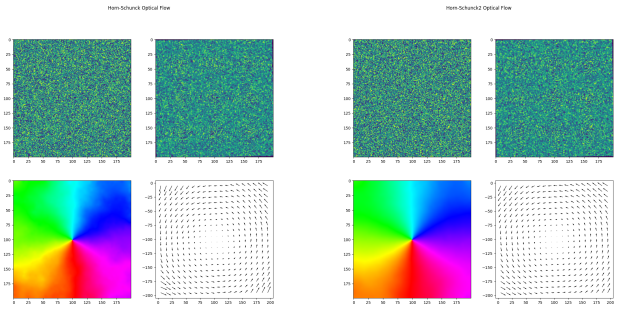

Figure 4: Horn-Schunck on random noise image rotated by 1 degree using 1000 iterations on the left image and 3000 iteration on the right image.

C. Testing the algorithms on other examples and time comparison

By initializing Horn-Schunck with Lucas-Kanade output, we can decrease the number of iterations and get a good estimate flow in less time.

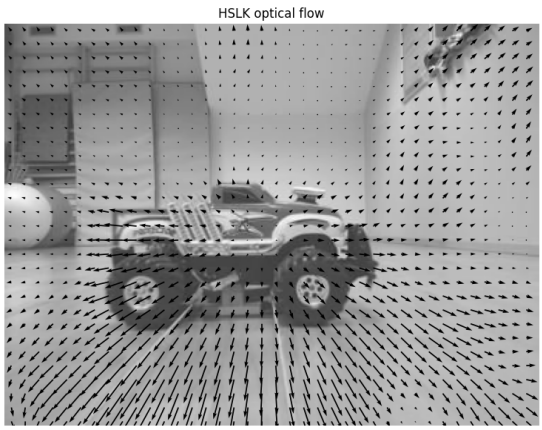

Figure 5: Horn-Schunck initialized with Lukas-Kanade output using 500 iterations.

In figure 5, we can see that in less time, we can get a better flow estimate just by initializing Horn-Schunck with the output of Lucas-Kanade. Below is a table of speed comparisons on images with size 288x384.

- LK: Lucas-Kanade
- LKP: Lucas-Kanade Pyramidal
- HS: Horn-Schunck
- HSLK: Horn-Schunck initialized from Lucas-Kanade output

Algorithm	Neighborhood	Iterations	Levels	Seconds
LK	11	/	/	0.03
LKP	11	/	6	0.05
HS	/	500	/	0.77
HS	/	1000	/	1.45
HSLK	/	500	/	0.84
HSLK	/	1000	/	1.36

Table I: Execution time comparison between algorithms and parameters

The basic implementations of the two algorithms were tested on other 3 sets of images, as can be seen in figure 6, where the images on the left column are produced from Lucas-Kanade, and the images on the right column are produced by Horn-Schunck.

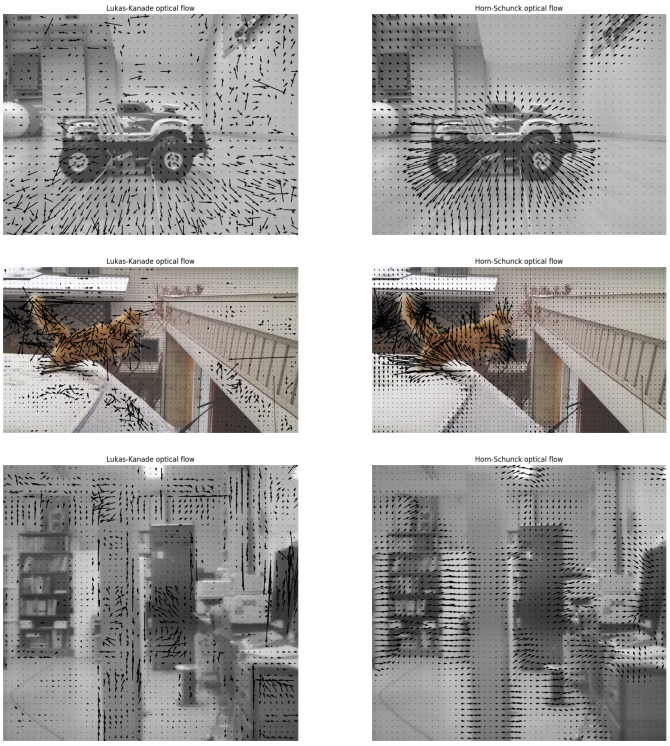

Figure 6: Lucas-Kanade examples using a neighborhood of 11.

We can see that Horn-Schunck estimates the flow more reliably and accurately, while Lucas-Kanade suffers from background noise.

III. Conclusion

In this assignment, two basic and traditional algorithms were implemented: Lucas-Kanade and Horn-Schunck. These algorithms are very fast and can be efficient depending on the taks at hand. However, they can fail on certain scenarios violating the small motion assumption. Because of it, a pyramidal approach can be used for Lucas-Kanade, estimating the flow at lower levels, giving better and more accurate flows. Also, in regions with low texture, the flow will not be accurately estimated. For this, Harris response can be used, estimating the flows around the detected edges, and returning relevant flow vectors.