ACVM - Lucas Kanade and Horn Schunck Optical Flow

Sebastijan Trojer

I. Introduction

In this project we were dealing with computing the optical flow in consecutive frames. Optical flow is a velocity field in the image which transforms one image into the next in the sequence. Therefore we are trying to determine where each pixel in the first image is in the second. We implemented two well-known algorithms – Lucas Kanade (LK) [1] and Horn Schunck (HS) [2], evaluated and compared them, and considered some improvements.

II. Experiments

A. Lucas-Kanade and Horn-Schunck Optical Flow

We implemented both methods as described in the instructions. With LK, the only deviation from the given equations was that the temporal derivative was smoothed with a small gaussian to reduce noise. Results of the rotated noise for both methods, along with the visualization of angle and magnitude of the optical flow are shown in Figure 1. Both methods performed well, however the HS output is smoother and more accurate. The roughness of the LK can be seen in the artifacts in the color-coded angle-magnitude image on the bottom left, and in some artifacts at the corners of the visualized flow. The reason in the discrepancy is that LK only considers a small neighborhood and does not assume global smoothness, while HS enforces it.

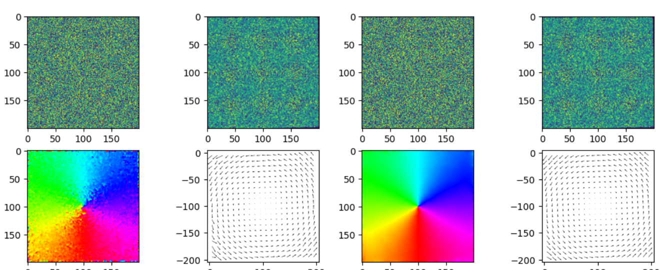

Figure 1. Rotated noise flow detected by LK (left) and HS (right).

The methods were also tested on 3 other real-life images. The results are shown in Figure 2. In these cases we can see more clearly what assumptions each approach makes. The LK tries to solve a local neighborhood problem and is sensitive to larger changes. The outcome is more sensitive to high frequency details, especially where there is low texture or edges, which is reflected by a lot of noisy vectors in the grassy area. Since it also assumes only local changes the directions of the output in that area are highly inconsistent. We can notice similar problems on the other examples as well. LK also had trouble detecting the movement of the building in the first example, even though it is a fairly structured texture with plenty of edges, which stems from the fact that LK has trouble in scenarios where the image gradients are disproportional to each other, making the flow in such regions unreliable. On the other side, the HS output resembles the true flow much better. That is because HS avoids the eigenvalue disproportionality issue by iteratively solving the problem globally.

B. Lucas Kanade Improvements

Since we are able to identify areas where LK flow cannot be reliably estimated we implemented that using the Harris detector. The Harris detector measures the quality of the local gradient structure and identifies corners. That leaves us with a mask, that indicates high and low quality features/areas, but this adds a new threshold parameter that has to be determined. Therefore, using the mask allows LK to focus only on areas where we can reliably estimate the flow. To confirm this, we evaluated LK with and without the Harris detector and the comparison is shown in Figure 3. The top row shows the angle and the magnitude of the flow, while the bottom row shows the flow field. Notice that the filtered methods have much less saturated responses and the Harris detector successfully suppressed the majority of the noise, which enabled LK to focus operate on the quality points and better estimate the flow in areas with corners, such as the building and the bookshelf.

C. Parameters

Both LK and HS introduce some parameters. An overview is shown in Table I. During evaluation we considered multiple values for each but eventually we decided on some optimal values. As mentioned, the sigma in the LK determines the amount of detail we keep in an image so setting it too high produces bad results, as there is not enough texture in the image. The neighborhood size is a harder parameter to optimize, as it again resembles a trade-off between detail and robustness to noise. It also determines the magnitude of motion we are able to detect. We use a small neighborhood for small motions and relatively non-noisy images with rich texture, and a large one for the opposite use case.

Parameter	**Effect on Performance**
Neighborhood size	Larger values smooth flow but lose fine details – small values improve detail but increase noise. We set this value to 3.
Smoothing (σ)	Higher σ reduces noise but can blur motion edges; low σ retains details but is sensitive to noise. We set this value to 1.
Number of iterations	Determines a balance between speed and flow estimation quality. We set this value to 1000.
Regularization weight	Enforces smoothness in optical flow and how fine are the details we consider. We set this value to 0.5.

Table I

Key parameters and their effects of LK and HS. The first two rows describe LK parameters and the second two the HS parameters.

For HS, the interesting parameter is the regularization weight, which controls how much we penalize abrupt changes. Figure 4 shows the difference in flow with parameter values set to 0.1, 0.5 and 3 respectively with 1000 iterations. Notice how there is less noise, the larger the weight, but with it being set too high we also start losing on valuable information. There seems to also be a correspondence with how much flow we detect in the left area, where there are branches, which are

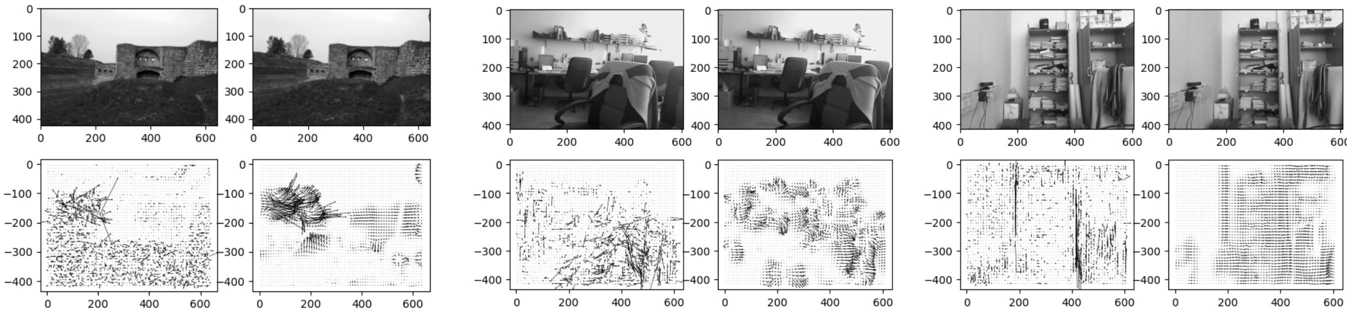

Figure 2. Input images (top row), LK (bottom left) and HS (bottom right) optical flow estimations.

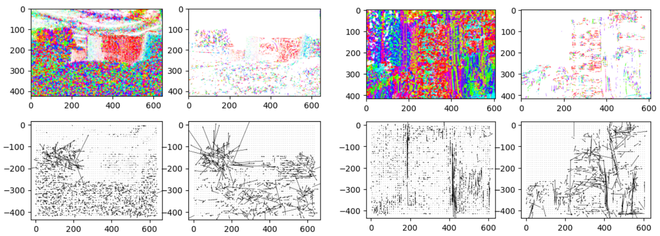

Figure 3. LK without the Harris detector (left) and with Harris detector (right) on images from example 1 and 3.

pretty monotone, so that might be why the noise is amplified with the regularization weight.

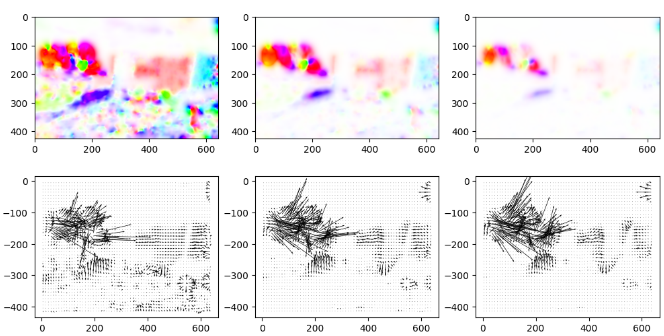

Figure 4. HS flow detection with regularization weight set to 0.1 (left), 0.5 (middle) and 3 (right) on the image from example 1.

D. HS optimization using LK

Due to requiring multiple iterations HS can sometimes be a bit slow. We usually start the process by initializing U and V to zero but we wanted to find out if initalizing them to the values returned by LK could speed up the convergence. To detect convergence we computed MSE between the new and old values of U and V and set a stopping threshold. The result was a somewhat convergence, however the speed is highly dependent on the threshold selection, which, if set too low, will cause the result to be of poor quality, or it will not even be reached. Threshold selection is also troublesome because it has to be considered for each image pair separately. Initializing with LK outputs usually caused the algorithm to stop up to 100 iterations faster, meaning we could get between 10 and 20% faster computation time, depending on what number of iterations we use (in our case it was between 500 and 1000). To really leverage this method it would need more fine tuning and a form of normalization to remove the need to set the threshold for each image pair separately.

E. Pyramidal LK

To address larger movements we implemented pyramidal LK (PLK). It works by constructing downscaled images, causing the 3x3 neigborhood at level 1 to consider 6x6 neighboorhood at level 2, etc. In the process we also warp the second image to better align with the first one. Figure 5 shows the performance of PLK compared to the original LK with Harris. We can see some significant improvement in detail capturing and noise reduction - all the bookshelf edges are better defined and consistent. We also tested if multiple iterations of LK at each level of the pyramid further improves the result, which is shown in the last column. We can notice that some of the magnitudes are more pronounced, but if we inspect the flow diagram, there is not a significant change in quality.

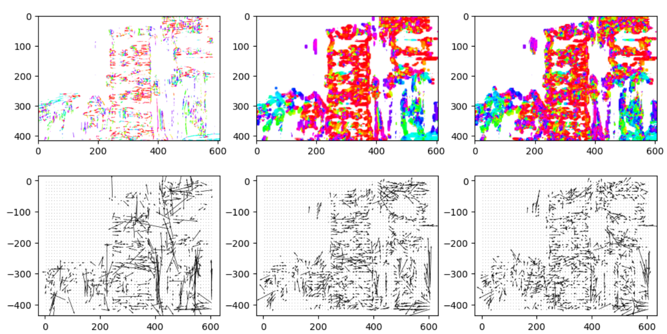

Figure 5. Pyramid LK on images from example 3. LK with Harris (left), PLK with Harris (middle), PLK with harris and 5 iteration per layer (right)

III. Conclusion

We implemented, evaluated and improved 2 optical flow estimation algorithms. We discussed some of the weaknesses of each and determined which method is more suitable in which scenario. LK works well in a scenario with minimal, local movements, and is computationally efficient, but is very sensitive to noise, which we addressed by using the Harris detector. HS is overall a stable approach and provides better results than LK, but is more computationally expensive.

References

[1] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *IJCAI’81: 7th international joint conference on Artificial intelligence*, vol. 2, 1981, pp. 674–679.

[2] B. K. Horn and B. G. Schunck, “Determining optical flow,” *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.