Optical Flow Report

Meldrick Reimmer
Universite de Bourgogne
Master of Science in Computer Vision
Email: meldrick_reimmer@etu.u-bourgogne.fr

Abstract---This is a report for my course module in Autonomous Robotics. This report is based on a practical class we had to get familiarize with Optical flow problem. I implemented " Horn-Schunck" and " Lucas-Kanade" method. These methods were applied to image and video stabilization problems. I implemented this using Matlab. You will get in the references the link to my Github account for the source code and sequences images I used for this work. I tested the methods on the several sequence images I had and manipulated the parameters to get different outcomes. This report will give you detailed knowledge of the things you will need to know about optical flow and the manner in which I implemented them.

I. Introduction

OPTICAL flow is the distribution of apparent velocities of movement of brightness patterns in an image. Optical flow estimation has given rise to a tremendous quantity of works for 35 years. Optical flow provides fundamental information on the basis of various computer vision systems, in a wide range of applicative domains. The most common assumption used in optical flow estimation is the brightness constancy assumption. It states that the gray value of corresponding pixels in the two consecutive frames should be the same. Unfortunately, not every object motion yields a change in gray values and not every change in gray values is generated by body motion. Optical flow can also be a determinant feature in video indexing and retrieval.

The visualization of motion fields provides a qualitative insight on the accuracy of the estimation. The two main visualization techniques are Arrow visualization and Color visualization. Most of my results were done in both visualizations. You will see later in the report, how it represented the actual sequences of images given.

II. Objectives

The steps I took to undertake this implementation was done regarding the following :

1) Implement the Horn-Schunck algorithm where it will iterate a fixed number of times.
2) Implement the Lucas-Kanade algorithm .
3) Test my implemented methods on sequences of images given and manipulate the parameters
4) Improve the Lucas-Kanade method with a multiresolution scheme.
5) Build a fucntion to compute the affine parameters, which i will use to compute the displacement of points from one image to another.
6) Compensate the camera motion of successive images with theta to stabilize the sequence.

III. Performing Tasks

A. Horn-Schunck Algorithm

Horn-Schunck method is a classical optical flow estimation algorithm. It assumes smoothness in the flow over the whole image. Thus, it tries to minimize distortions in flow and prefers solutions which show more smoothness. To perform this task I first made a derivative mask from that, I then made a convolution with the image pixels and the mask. I used Matlab built function for this purpose after that I then computer the u and v which are the movement vector. I visualized them outputs in Arrow visualization and Color visualization.

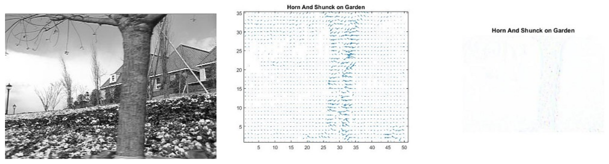


Figure 1.

B. Lucas-Kanade Algorithm

The Lucas-Kanade optical flow algorithm is a simple technique which can provide an estimate of the movement of interesting features in successive images of a scene. We would like to associate a movement vector (u, v) to every such "interesting" pixel in the scene, obtained by comparing the two consecutive images. To perform this task I first made a derivative mask from that, I then made a convolution with the image pixels and the mask. I computed the (u and v)being the movement vectors using the "Pinv" function for pseudoinverse. I visualized them outputs in Arrow visualization and Color visualization.

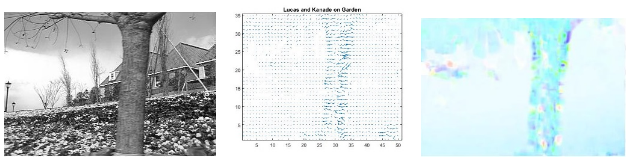
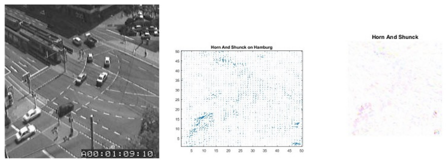
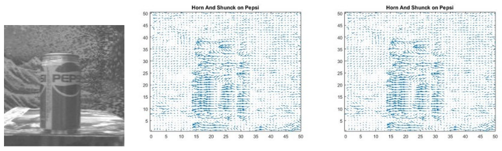


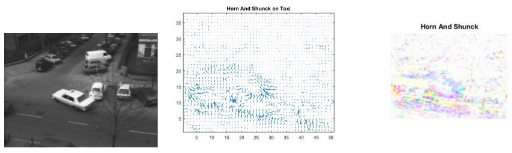
Figure 2.

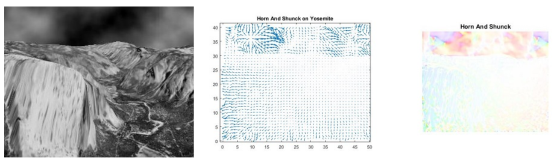
C. *Methods on Other Sequences of Images*

1) Horn-Schunck on Hamburg, Pepsi, Taxi and Yosemite Images.



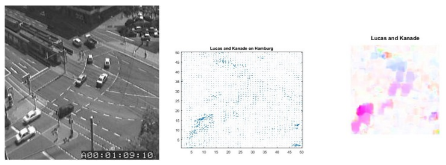


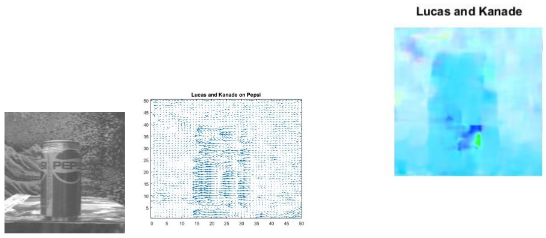


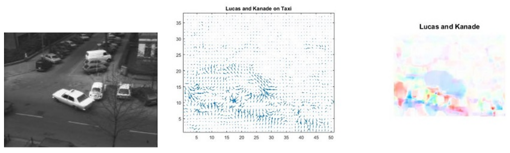


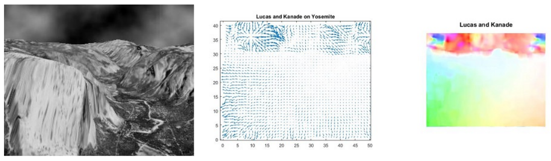
With each sequence of images, I gave different parameters, which gave our different results just like the images above. In general that HS method is a global smoothing. I visualized them outputs in Arrow visualization and Color visualization.

2) Lucas-Kanade on Hamburg, Pepsi, Taxi and Yosemite Images.





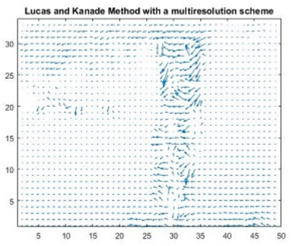




we can see that LK method is very unstable at edges and at low texture areas as compared to the HS method. It also gives stable results at corners. This usually happens when there is much noise in the image. I visualized them outputs in Arrow visualization and Color visualization.

D. *Hierarchical LK (HLK)*

The basic idea is to form a pyramid of frames and perform LK on the highest level of pyramid (low-resolution image) and propagate the optical flow to next levels. To perform this task I first made a derivative mask from that, I then made a convolution with the image pixels and the mask. I computed the (u and v)being the movement vectors using the ”Pinv” function for pseudoinverse. The results are good in the area of large motion as compared to simple LK. I visualized them outputs in Arrow visualization.



E. *Parametric motion estimation and application to image stabilization*

1) Getting equation to write the function AffineMotion

*First step:

M . θ = P

[m x θ] [θ x 1] = [m x 1]

*Second step: pseudo inverse of M

$(M^T.M).\theta$ = M^T.P

θ = $(M^T.M)^{-1}.M^T.P$

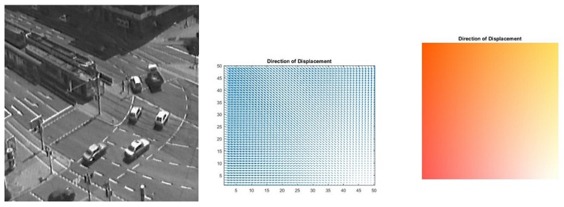
*Third step: using in (1) and (2) question

I_x.(ax +by +c) + I_y.(dx + ey + f) = $-I_t$

$(I_{xx}I_{yy}\ I_{yx}I_{yy}I_y)$. θ = $-I_t$

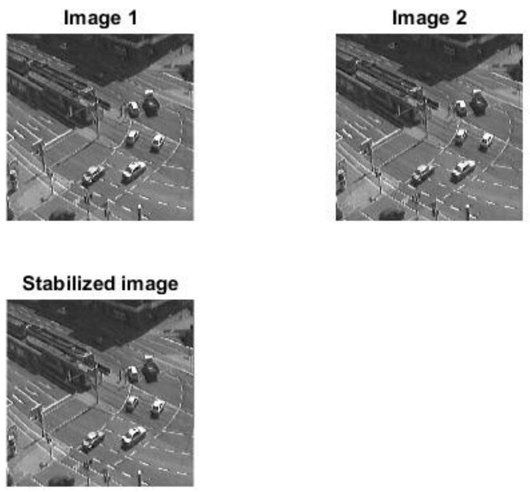
2) Getting theta to get the movement vectors u and v

After I create my function for the affine motion, I then use that to get the values of theta. With that, I can use that to calculate the value of the movement vectors (u and v). After this, I then display the direction.I visualized them outputs in Arrow visualization and Color visualization.



F. *Compensate the camera motion of successive images with theta to stabilize the sequence.*

The main idea is to take a sequence of frames and compute the affine motion between each pair and accumulate the motion fields and compensate for the motion in successive frames. I first made this using two images after which I then use a sequence of images to get a new stabilized image.



Stabilizing Image with two Sequence of Images.

Stabilizing images with 41 Sequences of Images

REFERENCES

[1] Optical Lectures By Prof. Yannick Benzeth
[2] Lucas/Kanade Meets Horn/Schunck: Combining Local and Global Optic Flow Methods http://www.mia.uni-saarland.de/Publications/bruhn-ijcv05c.pdf
[3] GitHub Account https://github.com/brown4eva