UE : VISION

Implementation of two optical flow methods

SCIENCES SORBONNE UNIVERSITÉ

20 January 2024

Contents

1 Horn-Schunck method 3
1.1 gradhorn(I1, I2) . . . 3
1.2 horn(I1, I2, alpha, N) . . . 3

2 Lucas-Kanade method 3
2.1 lucas_kanade(I1, I2, window_size) . . . 3
2.2 lucas_kanade_gaussian(I1, I2, window_size, sigma) . . . 3

3 Test and evaluation 3
3.1 square images . . . 4
3.1.1 Horn-Schunck method . . . 4
3.1.2 Lucas Kanade method . . . 5
3.1.3 Lucas Kanade with Gaussian window . . . 7
3.1.4 Conclusion . . . 9
3.2 mysine images . . . 9
3.2.1 Horn Schunck method . . . 9
3.2.2 Lucas Kanade method . . . 11
3.2.3 Lucas Kanade with Gaussian window . . . 12
3.2.4 Conclusion . . . 14
3.3 rubberwhale images . . . 14
3.3.1 Horn Schunck method . . . 14
3.3.2 Lucas Kanade method . . . 16
3.3.3 Lucas Kanade with Gaussian window . . . 18
3.3.4 Conclusion . . . 19

1 Horn-Schunck method

1.1 gradhorn(I1, I2)

The $gradhorn(I1, I2)$ function inputs two consecutive two-frame images and outputs the gradients of adjacent pixels in the X, Y and t directions, which is the average change intensity.

1.2 horn(I1, I2, alpha, N)

The $horn(I1, I2, alpha, N)$ function inputs two consecutive two-frame images, the smoothing term coefficient $alpha$ and the number of iterations N, and outputs the calculated optical flow matrix w by implementing the Horn-Schunck algorithm.

2 Lucas-Kanade method

2.1 lucas_kanade(I1, I2, window_size)

Here, at first, the color image is converted into a gray image. On this basis, the Lucas-Kanade method is applied to calculate the motion optical flow of each pixel between two consecutive image frames. The input values are two consecutive images $I1$ and $I2$, and the window size $window\_size$. The output results are the horizontal component u and vertical component v of the optical flow.

2.2 lucas_kanade_gaussian(I1, I2, window_size, sigma)

Here, we introduce the Gaussian window as the weights, which is to change the uniform window in the Lucas Kanade method. Gaussian windows assign each pixel within the window a weight based on its distance from the center of the window. This makes the pixels near the center have a greater impact on the calculation of optical flow, while the pixels at the edge of the window have less impact.

We add a parameter $sigma$ as the input, and have the same output u and v with the $lucas\_kanade(I1, I2, window\_size)$ function.

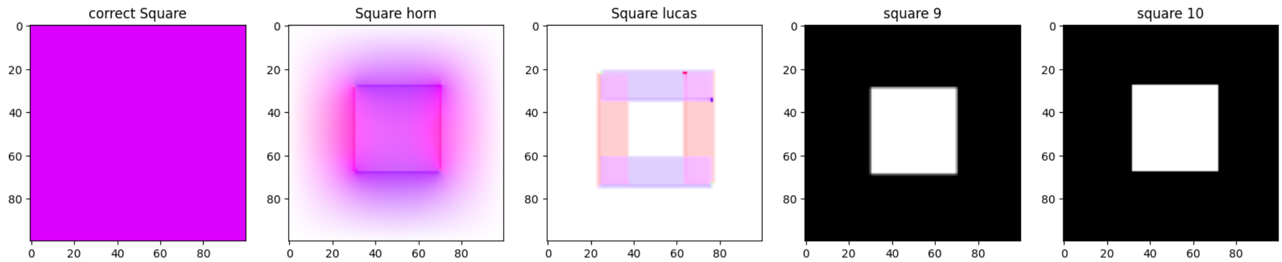
3 Test and evaluation

First, we define three error functions to evaluate the performance of the optical flow algorithm. By comparing the relationship between the reference optical flow field and the optical flow field obtained by the Horn Schunck or Lucas Kanade method, the mean and standard deviation are calculated.

And then, we test with the difference images. For each image set, we find the best result and compare the two optical flow methods.

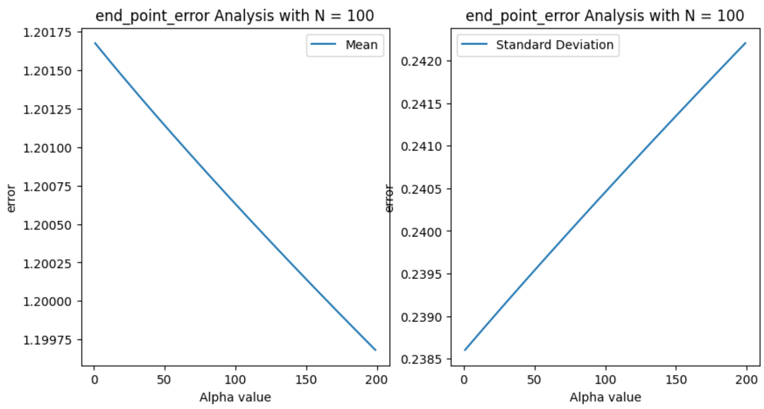
3.1 square images

As follows, the visualization of the square images in the dataset:



3.1.1 Horn-Schunck method

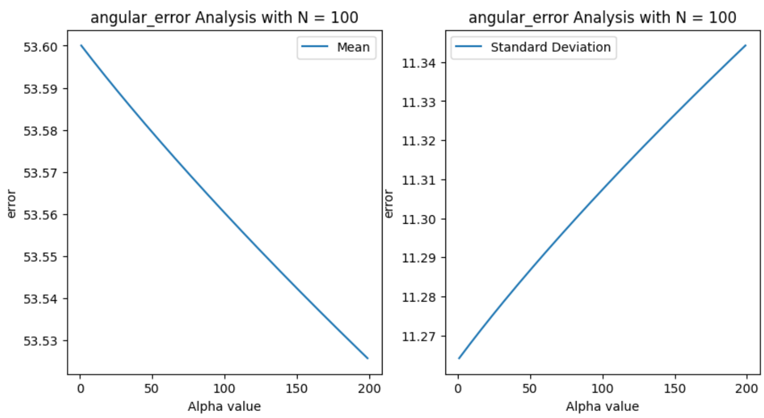
- end point error



We set the range of alpha from 0 to 199. From the result graph, we can see that as alpha continues to increase, the error becomes lower.

For square images, as *alpha* increases, the error decreases, that is to say, the smoothness is increased, because the motion of the two frames of square images is smooth and the area changes little.

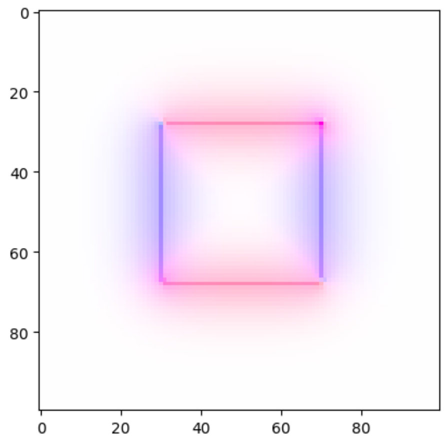
- angular error



The same as the test with the end-point-error.

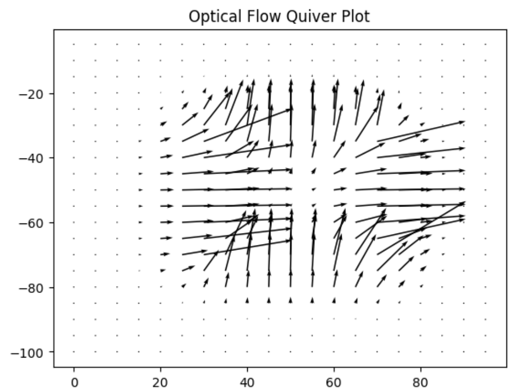
As the alpha value increases, the average value of the error continues to decrease, and the standard deviation of the error continues to increase. Because a higher alpha value prompts the algorithm to emphasize the smoothness of the optical flow field rather than the data fidelity of local pixels, there is Helps reduce the overall error in optical flow estimation. For the standard deviation, the range of changes becomes wider, the consistency of different areas in the image decreases, and the estimation error becomes larger.

- result image



We output the color result map, the results are as follows, here I set alpha to 100, because the larger the $alpha$, the better the effect, and $N = 100$ iterations.

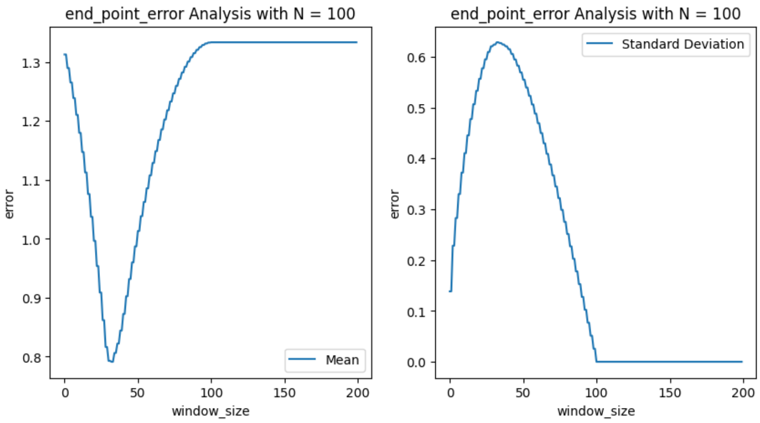
- quiver imgae



Here, each arrow represents the movement direction and speed of each pixel. The pointing direction is the direction of movement, and the length represents the speed of movement, which is the size of the optical flow.

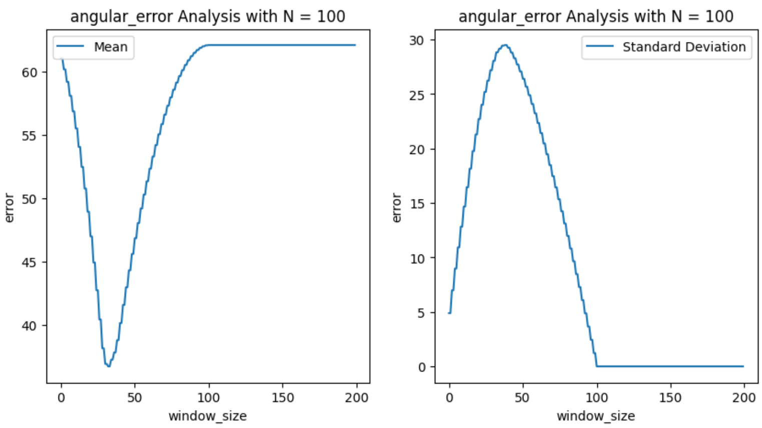
3.1.2 Lucas Kanade method

- end point error



The best alpha value is 32.

- angular error

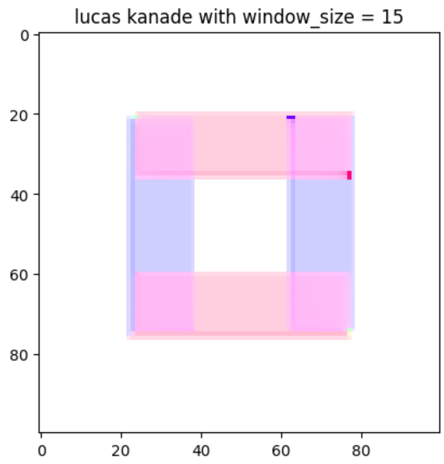


The best alpha value is 32.

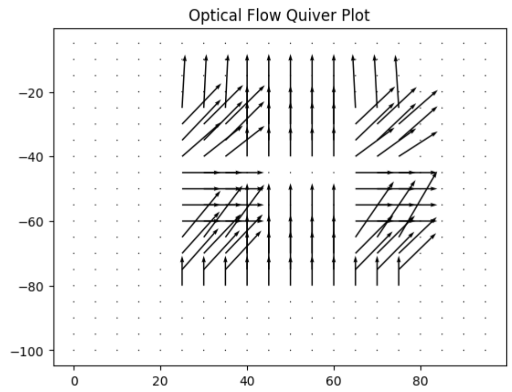
The error mean first decreases, then increases and becomes stable, while the error standard deviation is the opposite. So here I take the best window size as 15.

- result image

So here, we set $alpha = 15$ for the test, and the result image is as follows:



- quiver imgae

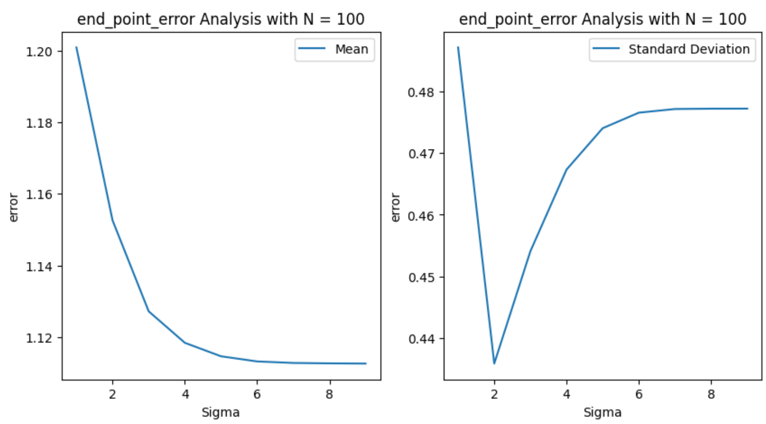


3.1.3 Lucas Kanade with Gaussian window

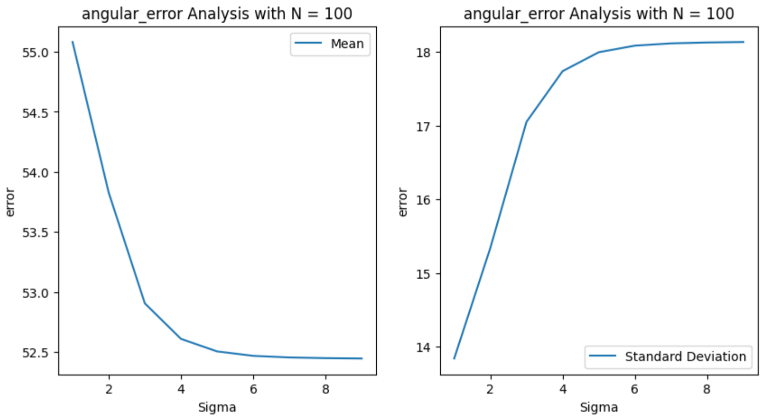
Here, we change the uniform window to the Gaussian window.

- sigma

At first, we set the range of sigma from 1 to 10 by using end point error function.



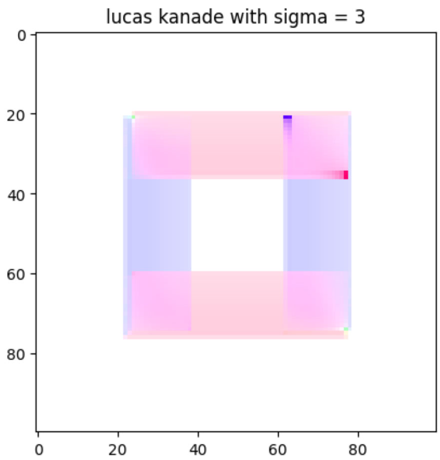
The same thing by using the angular error function.



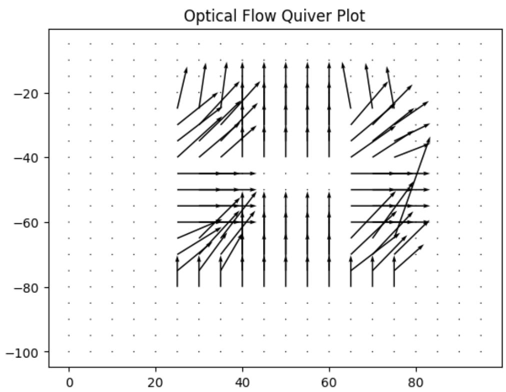
From these two error function, we can see that as *sigma* increases, the average error continues to decrease. For the two standard deviation result images, the details are a little different, but in general, the standard deviation continues to increase as sigma increases. Therefore, the best *sigma* I choose is 3.

-result image

Here, the window size is 15 and the sigma value is 3, the result image is as follow:



- quiver imgae



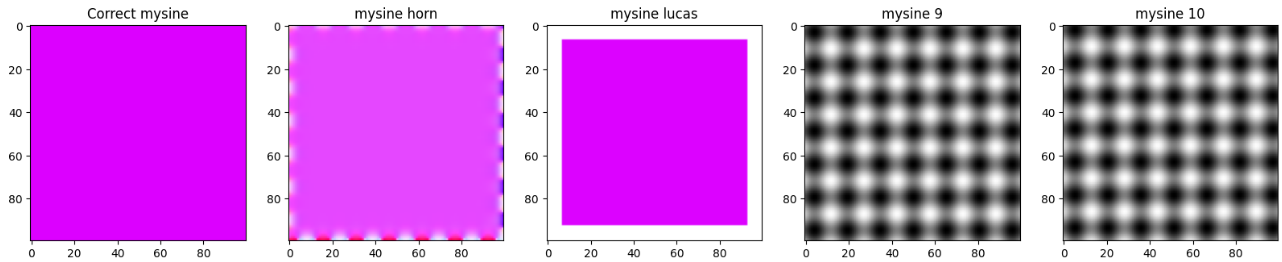
3.1.4 Conclusion

It can be seen from the results that the result map of Horn-Schunck is smoother and does not have very large color changes, because of the global smoothness of Horn-Schunck. For the Lucas-Kanade result, due to the selection of window size, it not fit the central square in the original image. And for Lucas-Kanade with Gaussian window, we get the same result, but for the detail of the quiver images, we can observe that after applying the Gaussian window, the edges are a little messier.

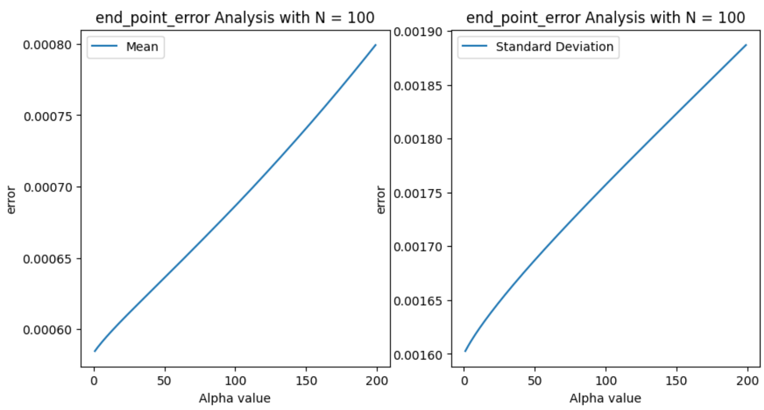
3.2 mysine images

3.2.1 Horn Schunck method

As follows, the visualization of the mysine images in the dataset:

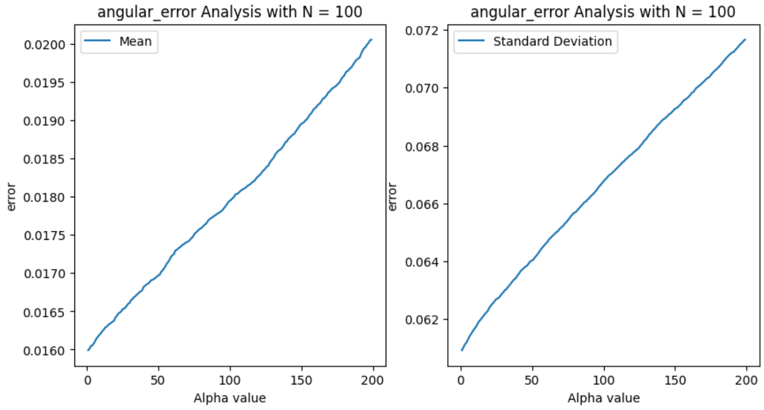


- end point error



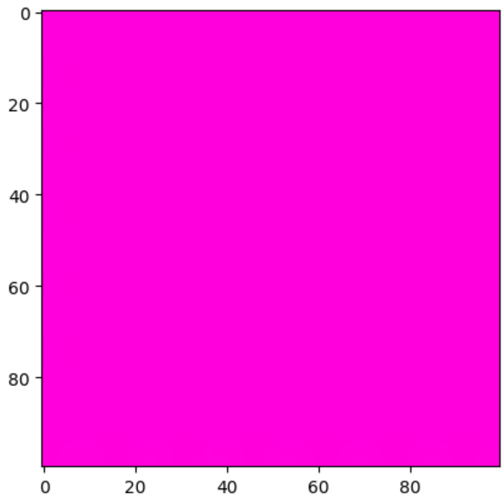
We set the range of alpha from 0 to 199. From the result graph, we can see that as alpha continues to increase, the error becomes bigger. So for $alpha = 1$, we have the best error.

- angular error



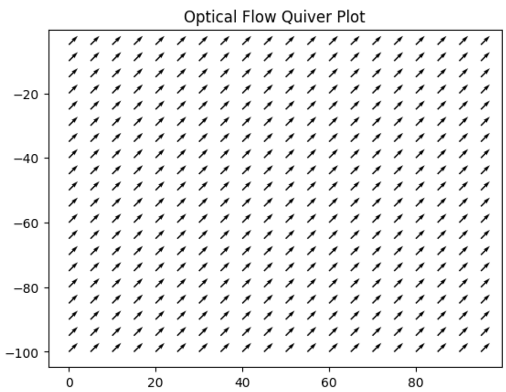
For the angular error, we have the same situation.

- result image



With the $alpha = 1$, we realize the color image.

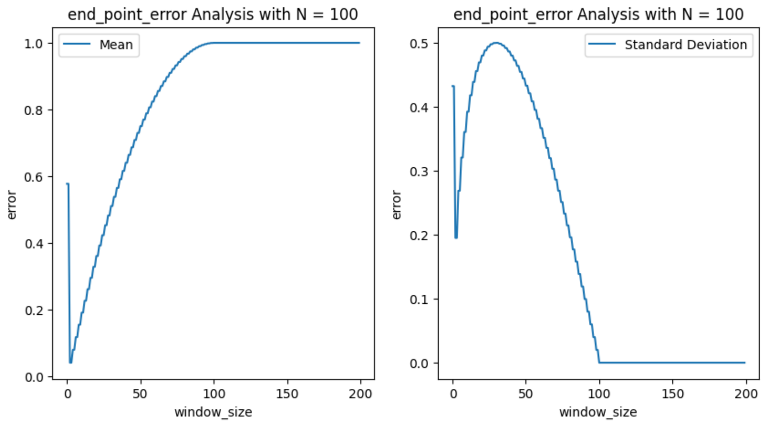
- quiver imgae



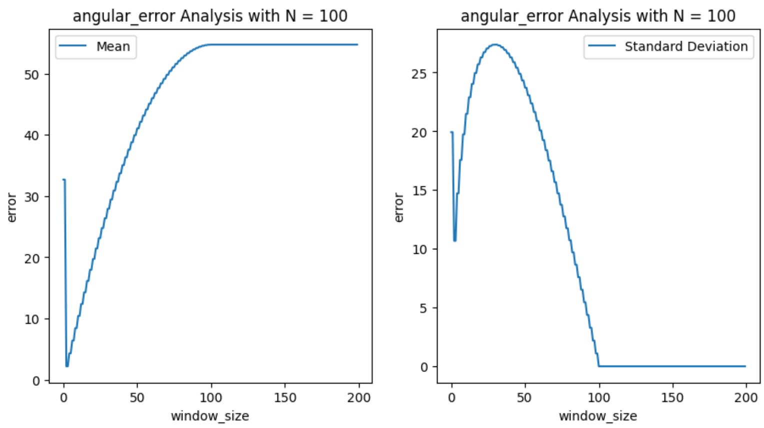
From this image, we can observe the direction and size of the movement. We can see that it moves uniformly, and for every pixel, there is the same movement.

3.2.2 Lucas Kanade method

- end point error



- angular error

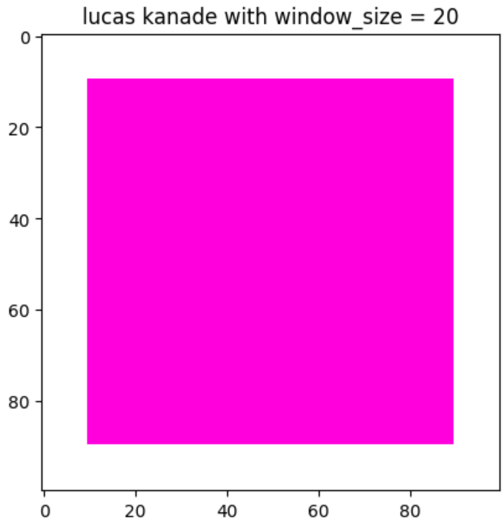


From these two error functions results, we can observe that the curve trends are the same, so we analyze them together.

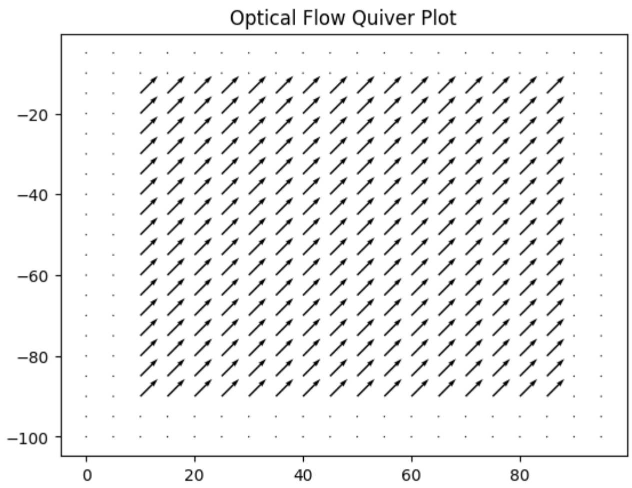
As the window size increases, the average value first decreases rapidly and then levels off, and the standard deviation first increases to a peak and then decreases. Because the small window is too small to capture enough information to estimate movement accurately and consistently, the average error is high and the standard deviation is small. As the window size increases enough large, the average error decreases and tends to be stable, which shows that when the window size is large enough, movement can be better estimated. And the standard deviation decreases, the window size is large enough to cover more motion information, making the estimation more accurate and consistent.

So we set the size of the window as 20.

- result image



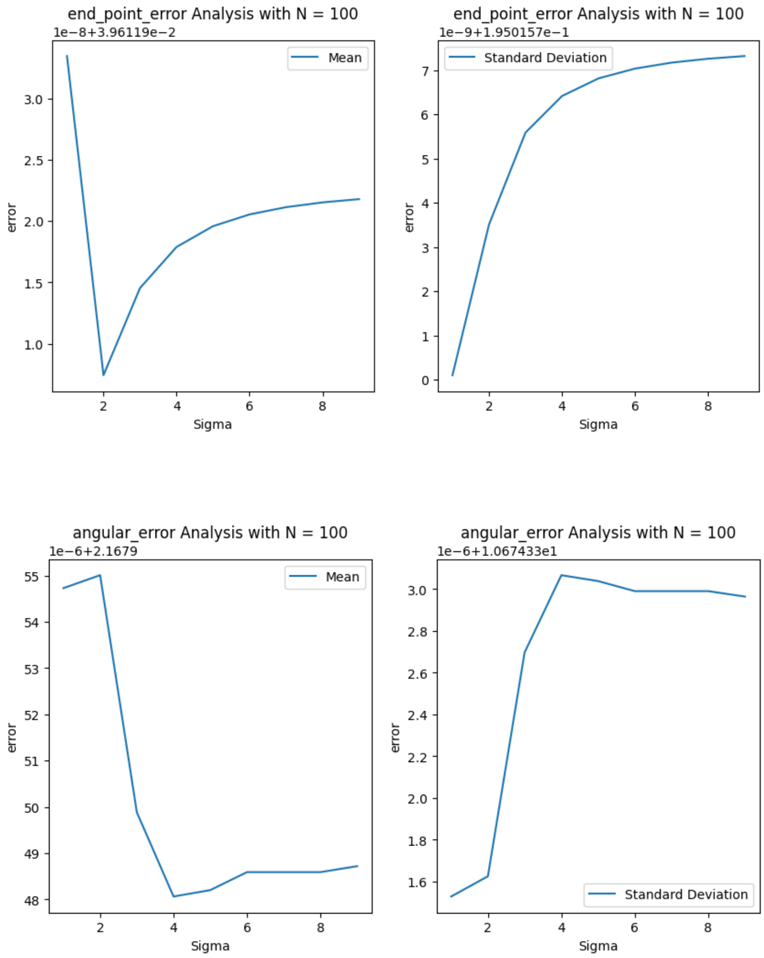
Here, we set $window\_size = 20$ and $sigma = 3$. **- quiver imgae**



3.2.3 Lucas Kanade with Gaussian window

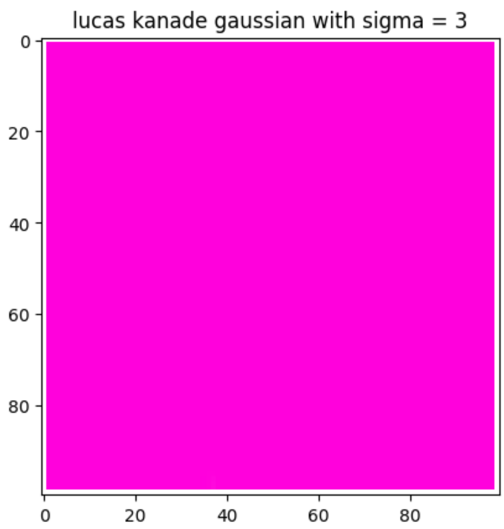
- sigma

By using end point error and angular error, the curve trends are generally the same, but the details are different. But we can observe together to find the best *sigma*.

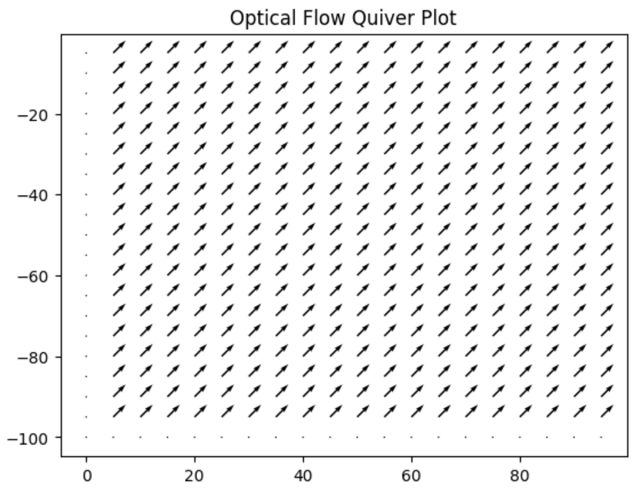


We should choose points with a low mean error and a relatively low error standard deviation. Therefore, our selection range is $2 - 4$, and in testing, we used $sigma = 3$.

- result image



- quiver image

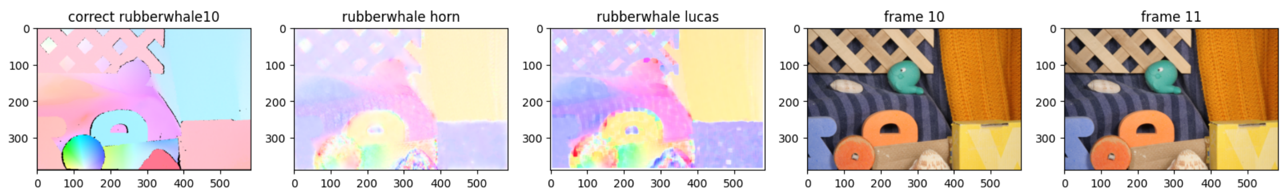


3.2.4 Conclusion

For the Horn Schunck method, no convergence is observed in the range of *alpha*, but for error, we choose $alpha = 1$ corresponding to the smallest error, but from the result image, we do not observe a color change, that is to say the movement corresponding to the original image. For the fixed window and Gaussian window of the Lucas Kanade method, both images show areas of a single color, and the optical flow algorithm does not detect significant motion, or the estimated motion is very small. And due to the introduction of *sigma*, the results are also different.

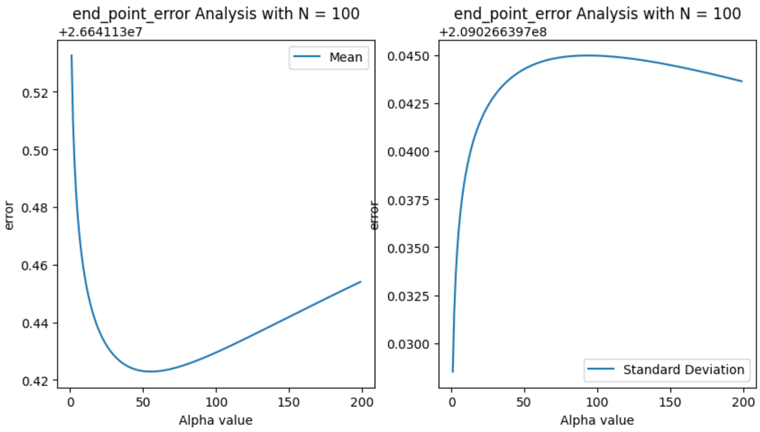
3.3 rubberwhale images

As follows, the visualization of the rubberwhale images in the dataset:

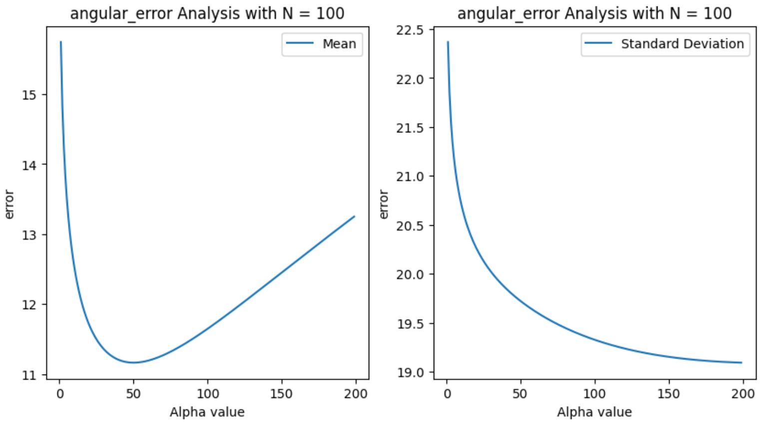


3.3.1 Horn Schunck method

- end point error



- angular error

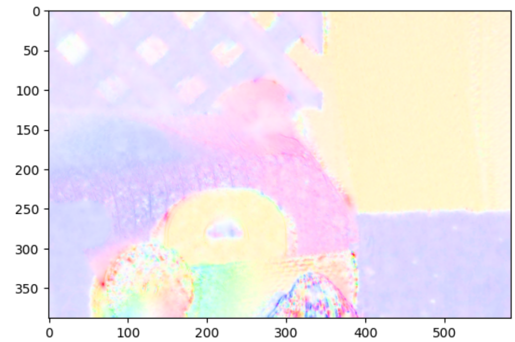


From the results of the two error functions, we can see that the trend of the mean curve is roughly the same, but for the standard deviation, it is completely opposite.

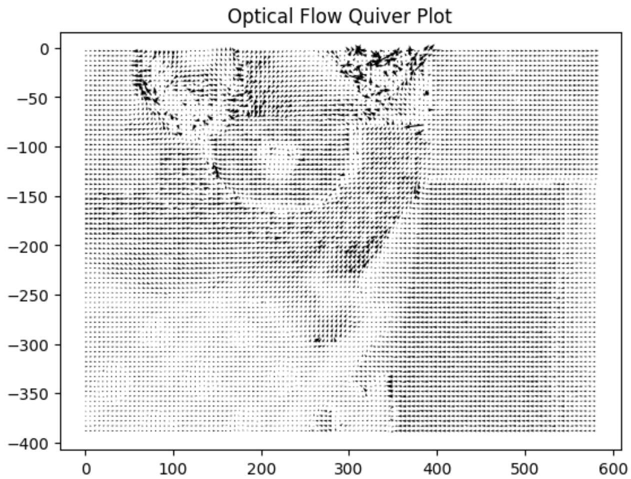
End point error focuses on measuring differences in magnitude of optical flow vectors, while angular error focuses on measuring differences in direction. Therefore, there will be different changes corresponding to different optical flows.

We choose alpha as 50.

- result image

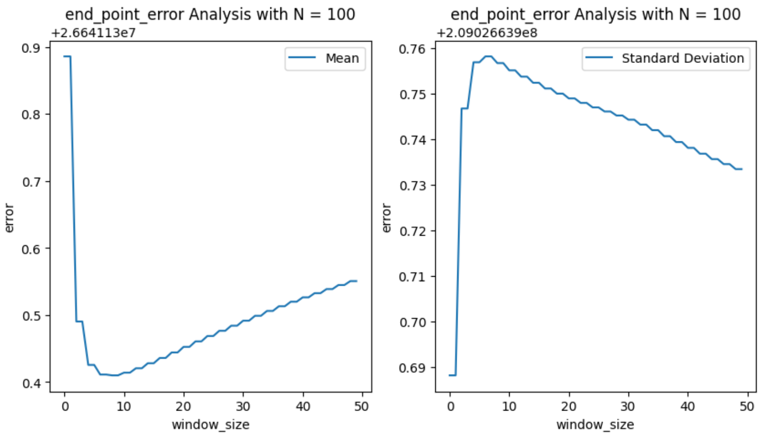


- quiver imgae

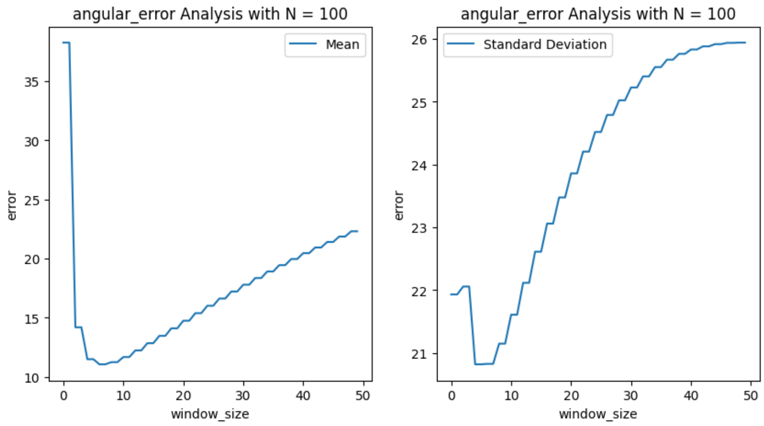


3.3.2 Lucas Kanade method

- end point error



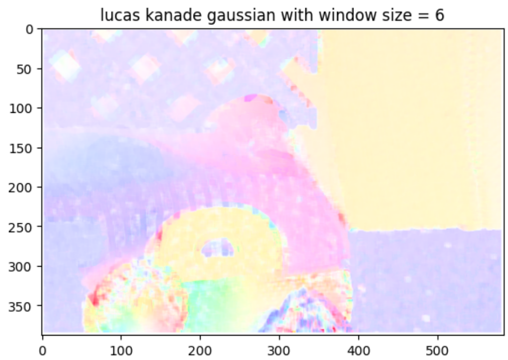
- angular error



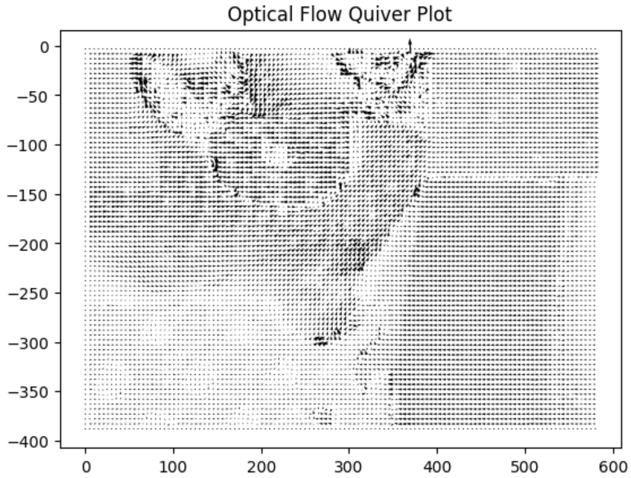
When the window size first increases, the average error decreases sharply. After the lowest point, as the window size gradually increases, the average error also increases.

For curved changes in standard deviation, for endpoint errors, the consistency of the errors improves with increasing window size, but for angular errors, larger windows may start to include many different directions of motion, causing the consistency of the error changes to decrease. So here I choose window size as 6.

- result image

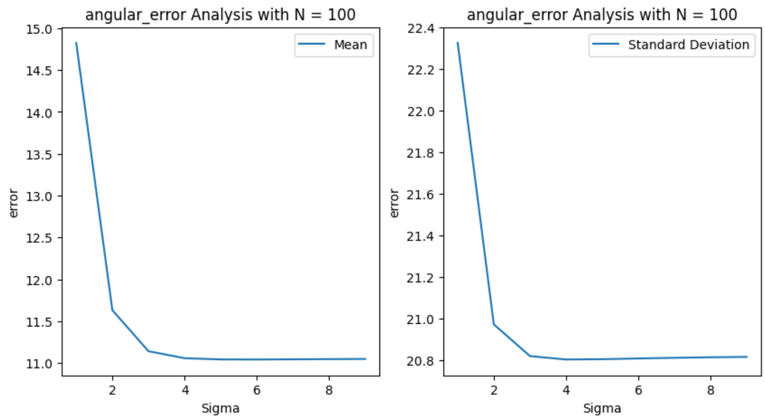


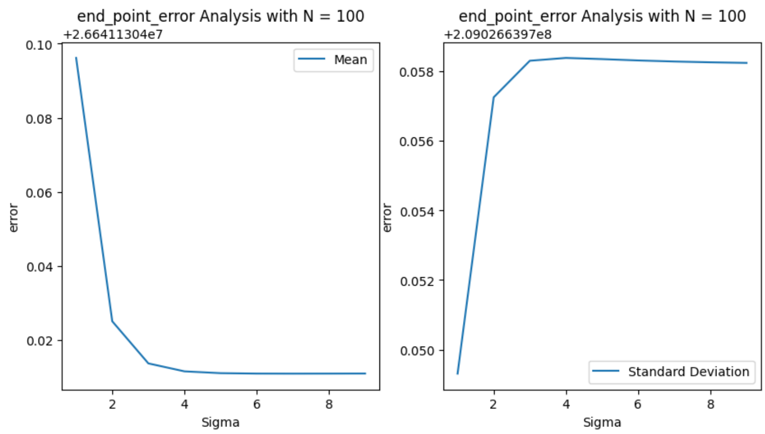
- quiver image



3.3.3 Lucas Kanade with Gaussian window

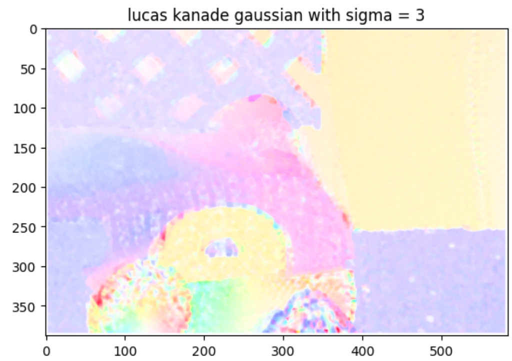
- sigma



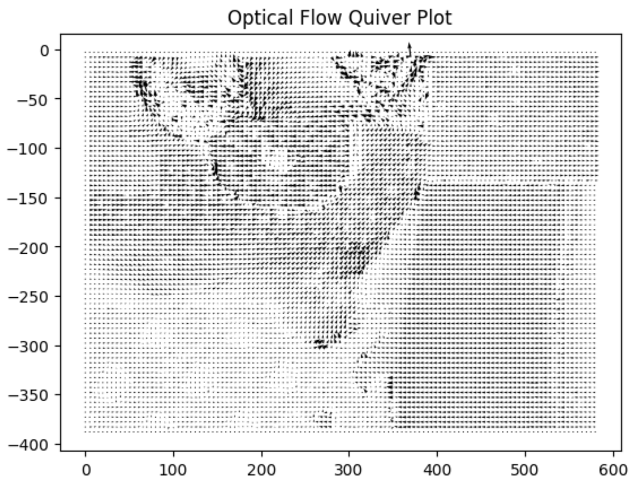


For both error functions, as sigma increases, the error average decreases rapidly and levels off. For the standard deviation, the results of the two error functions are different. Therefore we choose a sigma between 2 and 4, where the mean and standard deviation of the error are relatively low.

- result image



- quiver image



3.3.4 Conclusion

It can be seen from the quiver images of the three attempts that this movement is more complicated and the image size is larger than before.