512-AS6-REPROT

Chen Xu

A20377739

1. Problem statement

1. Estimate the optical flow vectors in a sequence of images.
2. The input to the program read from a video file.
3. The optical flow vectors should be computed and displayed on one of the images in the sequence as colored straight-line segments.
4. Pressing ‘p’ should pause/release the current image.
5. Implement one of Lucas-Kanade, affine-flow, or Horn-Schunk algorithm.
6. The spatiotemporal derivatives should be computed by extending the spatial gradient estimation technique you used in the previous assignments.

2. Proposed solution

The Horn-Schunck algorithm assumes smoothness in the flow over the whole image. Thus, it tries to minimize distortions in flow and prefers solutions which show more smoothness.

The flow is formulated as a global energy functional which is then sought to be minimized. This function is given for two-dimensional image streams as:

$$E = \iint \left[(I_x u + I_y v + I_t)^2 + \alpha^2 (\|\nabla u\|^2 + \|\nabla v\|^2) \right] \mathrm{d}x\mathrm{d}y$$

According to the contents we learned in the class, we know the detailed approach of Horn–Schunck method solution should be:

- Start with initial guess for u and v, we use L-K or affine flow iteratively refine them.

$$u^{n+1} = u^n - \frac{(I_x \bar{u}^n + I_y \bar{v}^n + I_t) I_x}{I_{x^2} + I_{y^2} + \alpha}$$

$$v^{n+1} = \bar{v}^n - \frac{(I_x \bar{u}^n + I_y \bar{v}^n + I_t) I_y}{I_{x^2} + I_{y^2} + \alpha}$$

- When max of $(u^{n+1} - u^n || v^{n+1} - v^n) < \tau$, where τ is the threshold, it stops.

3. Implement details

1. When pick up kernel we need set it to float.
2. Every time use read() function , it take one frame of the videos, we can use this to implement the pause function by calling while loop when process the frame, and use cv2.waitkey() to continue the video.
3. When process numpy array as matrix reshape() function may cause more bucket in following data processing.
4. When take parameter into some function, we need to know the type and change it into right type to compute.
5. I find out we don't have to write a help function, it also work, if I write comment and use print(__doc__) to display it at program running.
6. Use cv2.waitkey() to slow down the frame playing.

4. Results and discussion

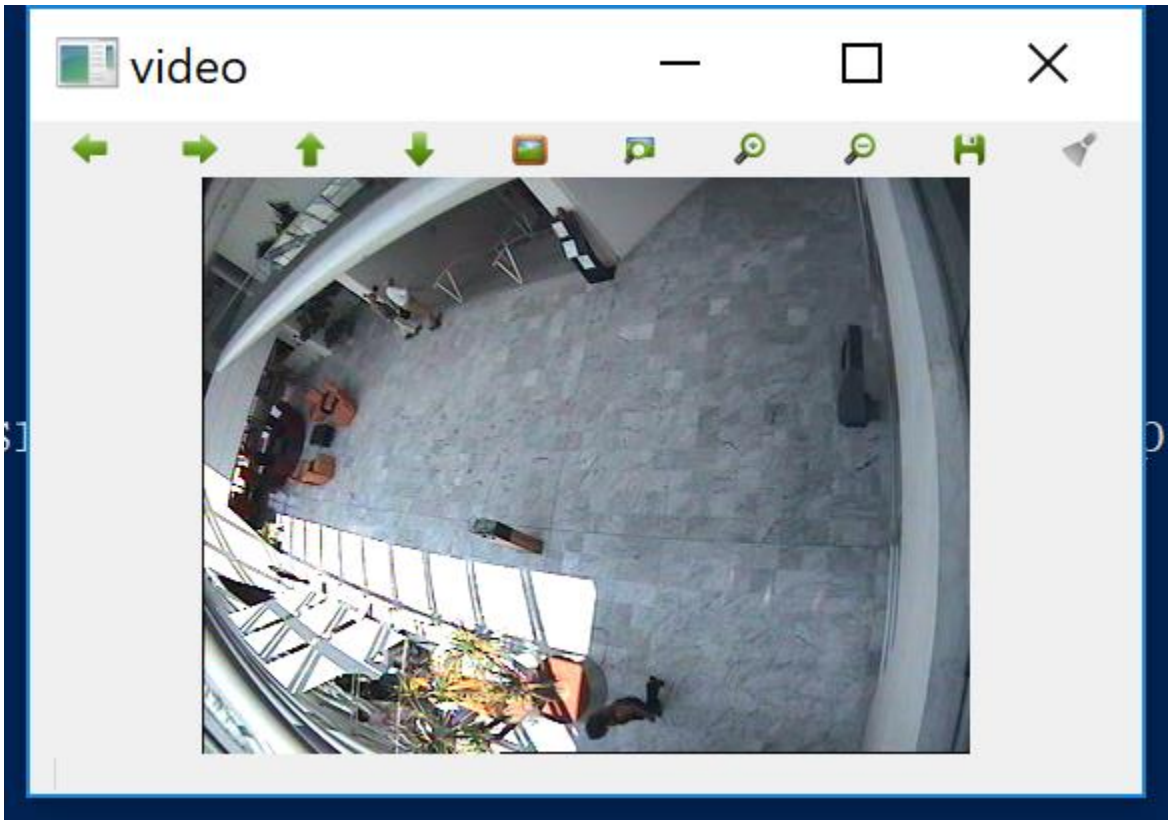
1. Read a file by using the argument when run the program.

```
src> python .\AS6.py ..\data\Walk2.mpg
```

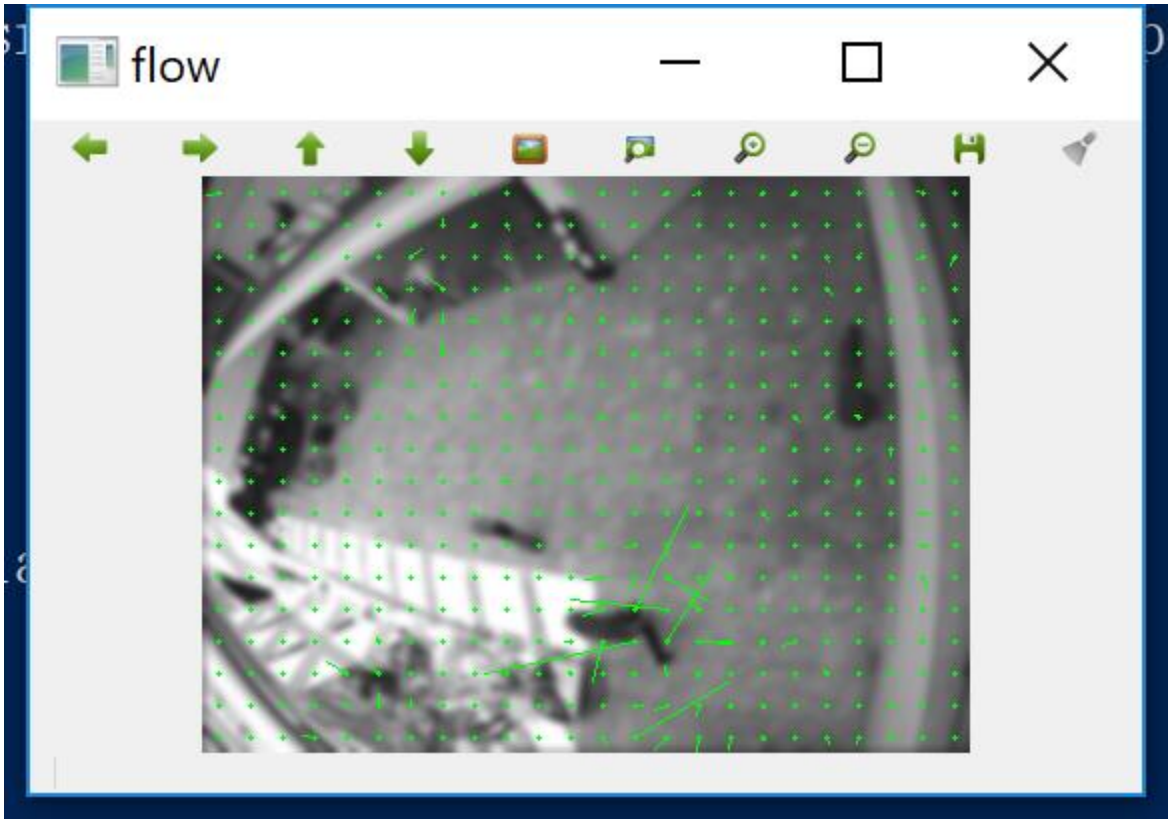
2. When running the program it will display the help doc.

```
Optical Flow Estimation
USAGE:
    AS6.py [<video_source>]
    e.g. .\AS6.py ..\data\Walk2.mpg
Keys:
    p ..... pause/release current image
    ESC ... exit <<press q twice to exit when video is playing>>
```

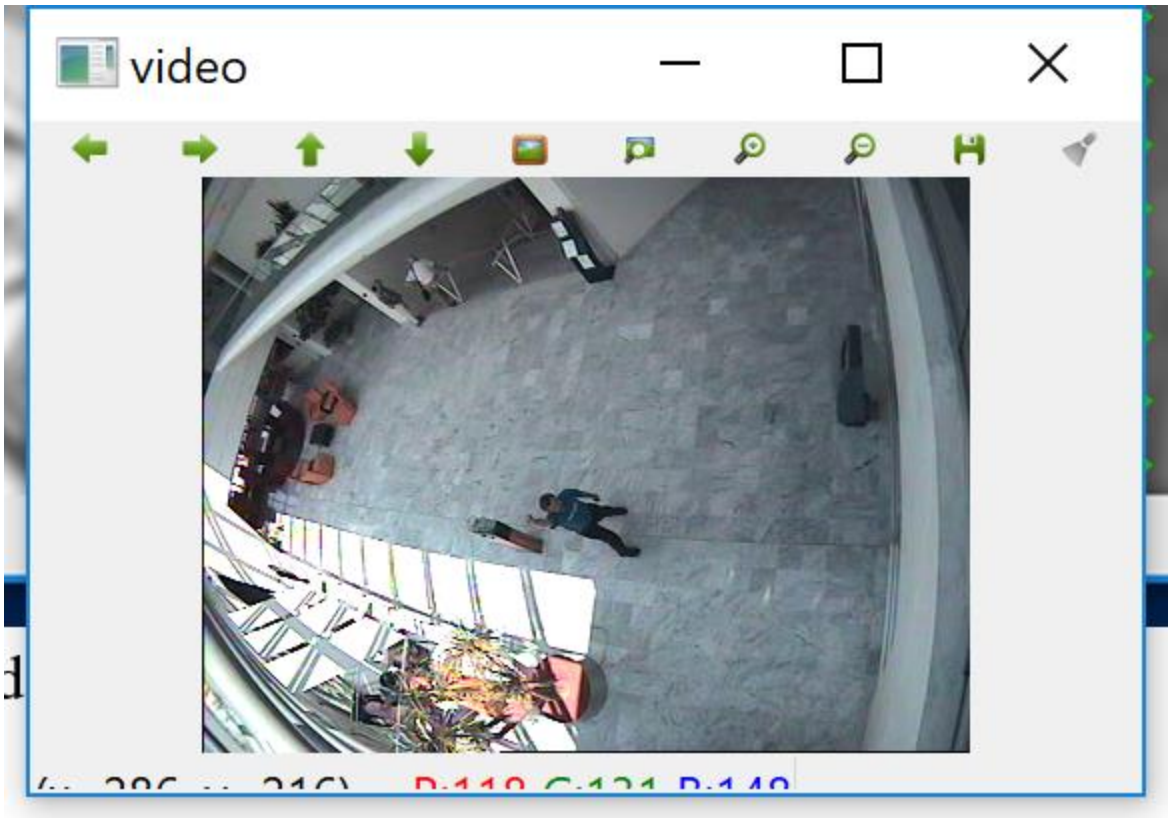
3. Video will play when running correct



4. Press ‘p’ or ‘P’ to pause the video. The Horn-Schunk algorithm implement result will display.



5. Press ‘p’ again, the video will continue playing.



6. Advantages:

   As a global technique, Horn-Schunk algorithm provides 100 percent dense flow fields, but is much more sensitive to noise.

   It takes regularization for all points and it gives solution optical vector everywhere. Compare to local methods, such as Lucas-Kanade, are less sensitive to noise, but lack the ability to produce dense optical flow fields.

5. Reference

[1] Motion detection using Horn Schunck algorithm and implementation

http://ieeexplore.ieee.org/document/5254812/

[2] OpenCV optical flow

https://github.com/opencv/opencv/blob/master/samples/python/opt_flow.py

[3] Horn–Schunck method

https://en.wikipedia.org/wiki/Horn%E2%80%93Schunck_method