Homework 04

Deepak Sharma
CSCI 731 Advance Computer Vision

July 17, 2017

PartA and B Key-Point Displacement.

1. **Solution**
   1. Converted input images/frames to into gray level images.
   2. Applied Bilateral filter for smoothing and image enhancement.
   3. Find Feature points using Shi-Tomasi method:

```
//Creating two point type array for storing intial and final location
//of key-points.
vector<Point2f> startPoints;
vector<Point2f> endPoints;

// Selecting key-points(Harris/Shi-Tomasi(Default)) with both high
// eigenvalue and and storing in startPoints vector.
float quality_level = 0.01; //Parameter characterizing the minimal
   accepted quality of image corners
int min_distance = 10; // Discard corner this close
int block_size = 3; //Neighbourhood used/patch size
bool useHarrisDetector = false;

goodFeaturesToTrack(gray, startPoints,
       MAX_COUNT, quality_level, min_distance, Mat(), block_size,
       useHarrisDetector);
```

   4. Converted Key-point coordinates to real valued coordinates using CornersubPix method:

```
//Creating termination critaria for Cornersubpix
//Termination critaria decide the level of accuracy while converting
//Integer corrdinates to real valued coordinates using subPix
TermCriteria termcrit(TermCriteria::COUNT|TermCriteria::EPS,20,0.03);
// Window sizes for Cornersubpixel method and optical flow patch size
Size subPixWinSize(20,20), winSize(31,31);
//convert corner coordinates to real coordinates,
//performe inplace conversion(output will be store in startPoints)
```

```
cornerSubPix(gray, startPoints, subPixWinSize,
        Size(-1,-1), termcrit);
```

5. Find association between feature points using Lucs Kanade optical flow method:

```
needToInit=0;
vector<uchar> status; // For each point found=1 else 0
vector<float> err; //For storing error measure for found points
if(prevGray.empty())
    gray.copyTo(prevGray);
int npramindLayers = 3; // Number of layers for LK method
//Calculate optical flow of keypoints stored in start points
//provide associated points in endPoints
calcOpticalFlowPyrLK(prevGray, gray, startPoints, endPoints,
        status, err, winSize, npramindLayers, termcrit, 0, 0.001);
```

6. For the Part B repeated the above process for first 30 set of consecutive frames.

2. **Output for Part A:**

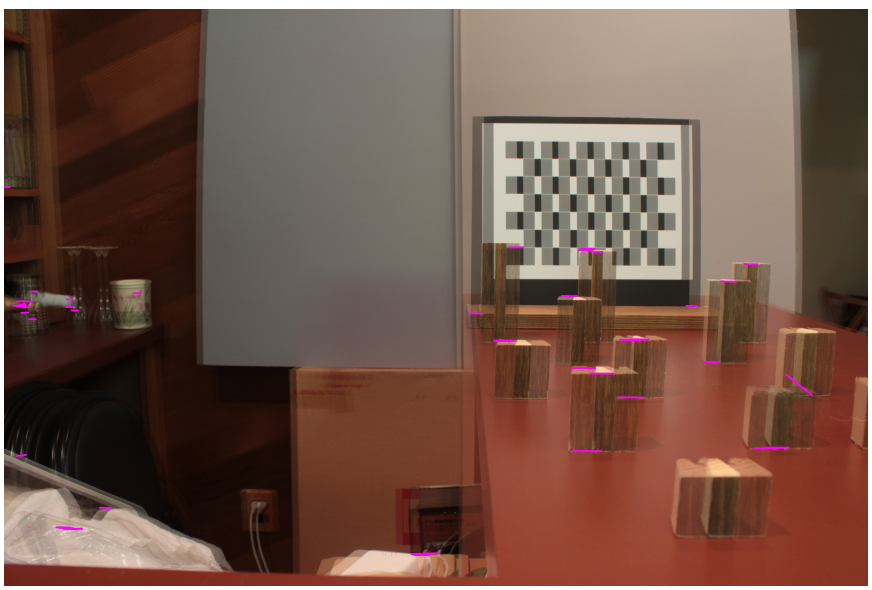

Figure 1: Part A, Combined the first and Second Image, for understanding the displacements of corner points.

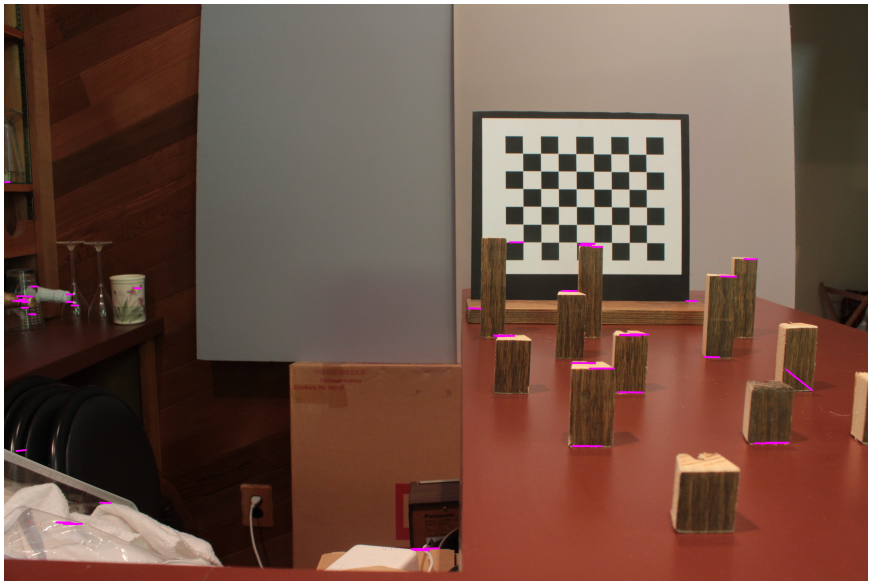

Figure 2: Part A, Displacement of the corner points, shown on second Image.

3. **Observations:**
   1. All corners of the cubes were not detected by Shi-Tomasi methods.
   2. Though the camera was moving from Left to Right but one of the displacement vector is visible in 'Z' direction. Here feature matching/ key-points association was reported incorrect by Lucus Kanade method.
   3. In part B, Motion vector away from the road was more visible due to movement in left to right or visa versa direction. Because points away from the camera was showing more distance travel in 2D projected image of the 3D world.

4. **Source code credit**
   1. Lucus Kanade Optical flow [6].
   2. Draw Lines and Arrow [5].
   3. Write Video [4].
   4. Codding style and guidelines [2].
   5. Assertion of a file path [7].
   6. Method definitions [1, 3]

References

[1] Gary Bradski Adrian Kaehler. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library 1st*. O'Reilly Media, Inc., 2 edition, 2016. ISBN 1491937998 978149193799.

[2] Errin Fulp. C++ coding guidelines, 2006. URL `http://csweb.cs.wfu.edu/~fulp/CSC112/codeStyle.html`.

[3] *The OpenCV Reference Manual*. Itseez, 3.2 edition.

[4] Karl Phillip. how-to-write-video-file-in-opencv-2-4-3, 2014. URL `source : https : / / stackoverflow . com / questions / 13623394 / how-to-write-video-file-in-opencv-2-4-3`.

[5] David Stavens. The opencv library: Computing optical flow, 2005. URL `source:http://robots.stanford.edu/cs223b05/notes/CS%20223-B%20T1%20stavens_opencv_optical_flow.pdf`.

[6] RafaRuiz ValeryTyumen, sturkmen72. Demo code for lucus kanade, 2013. URL `https://github.com/opencv/opencv/blob/master/samples/cpp/lkdemo.cpp`.

[7] Pherric Oxide Vincent, Pevik. Fastest way to check if a file exist using standard c++/c++11/c?, 2012.