IN4393-16 Computer Vision - Q4
Lab Exercise 6
Chaining

May 26, 2018

Students should work on this lab exercise in groups of two people. The assignment should be completed before 4th June 2018.

In a video sequences, a single point is unlikely to remain visible in all the frames thus we need to keep track of the points that are present from one frame to another. The point view matrix is constructed for all the video frames which saves the coordinates of the correspondence between the two consecutive frames and indicates the missing points from one frame to another with zeros values. Later Tomasi-Kanade factorization can be performed on the sub-blocks of the point view matrix for a 3D reconstruction.

1 Matching

As in the previous lab exercises, we need to find the correspondence between two images as a starting step. In this part, you will write a function that takes two consecutive images as input and matches interest points. You will work with supplied teddy bear images. The overall scheme can be summarized as follows:

1. Detect interest points in each image using the software supplied with the previous lab exercise. Interest points are Harris and Hessian Affine.

2. Extract SIFT descriptors around these interest points.

3. Concatenate SIFT descriptors extracted for the same image (Harris and Hessian). Get the coordinates that matches between region descriptors of two consecutive images, i.e.,

```
feature = [feat_Harris feat_Hessian];
descriptor = [desc_Harris desc_Hessian];
```

```
matches = ...
    vl_ubcmatch(descriptor-frame(i),descriptor-frame(i+1));
```

4. Use the Normalized eight-point Algorithm with RANSAC that you implemented for previous assignment. Given two consecutive images and their matches, find fundamental matrix and inliers (best matches) of those two images that satisfies the epipolar geometry. Remove inconsistent matches from the set of previously found matches.

```
newmatches = matches(:,inliers);
```

5. You need to repeat these steps for 16 consecutive frames including the matches between the last and first frame.

2 Chaining

Construct point-view matrix with the matches found in last step of section 1 for all consecutive tedy bear images (1-2, 2-3, 3-4, ..., 15-16, 16-1). The point-view matrix has views in the rows, and points in the columns. It is constructed as follows.

1. Start from the first two consecutive image matches where the normalized eight-point Algorithm with RANSAC described in the previous section is used for finding robust inliers.

2. Add a new column to point-view matrix for each newly introduced correspondence point and a new row for each pair of frames.

3. If a point which is already introduced in the point-view matrix and another image contains that point, mark this matching on your point-view matrix using the previously defined point column.

Useful Hints:

1. For the first pair, simply add the indices of matched points to the same column of the first two rows of the point-view matrix.

```
1        point-view-matrix(1,1:size(inliers,2)) = ...
             newmatches(1,:)';
2        point-view-matrix(2,1:size(inliers,2)) = ...
             newmatches(2,:)';
```

2. For subsequent frame-pairs, use set-intersection on the sets of feature-indices of current and previous frames, in order to find already found points.

```
1
2        [¬, IA, IB] = intersect(newmatches(1,:)', ...
             matches-in-fram(i));
3        point-view-matrix(i+1,IB) = newmatches(2,IA);
```

3. You need to find new matching points that are not in the previous match set. Note that you need to grow the size of the point view matrix each time you find the a new match.

4. The very last frame-pair, consisting of the last and first frames, requires special treatment. After adding new matches found in the last frame, move the matches of points that also match the first frame over to the columns that had already been allocated to those points in the first frame.

A sample point view matrix for the teddy-bear images is provided at Brightspace for your reference. Note that we show the indices of the coordinates of matching points in different frames in this example, however, the point view matrix could be constructed in binary format that indicates the presence of a feature point in different frames with one entry and the absence with a zero value.