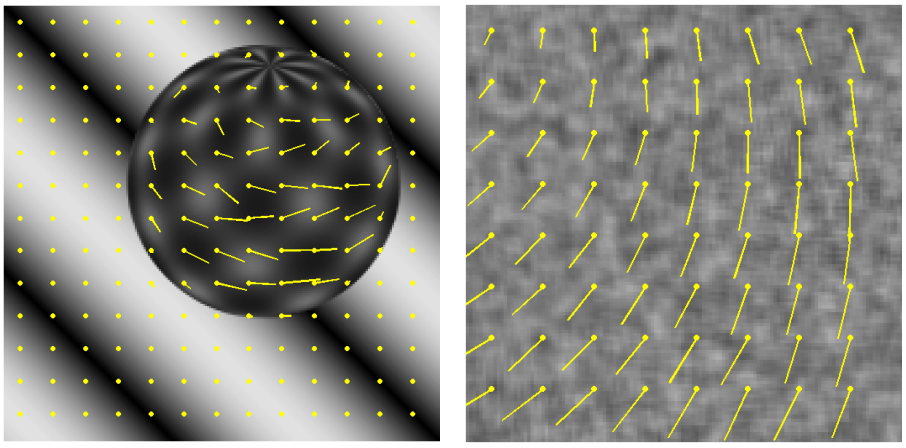
Assignment 4: Optical Flow and Structure from Motion

Robrecht Jurriaans (5887380), Taco Cohen (6394590)

December 2, 2012

1 Optical Flow

We have implemented Lucas-Kanade optical flow for a single scale on both synthetic images and on a sequence of a model house. For the synthetic images, optical flow was computed on a grid of non-overlapping 15×15 windows, whilst a set of points was tracked over the model house sequence. We created two separate functions for this: `LucasKanade.m` for the non-overlapping windows and `LK_M.m` for the tracking.



(a) Optical flow for the sphere images (b) Optical flow for the synth images

Figure 1: Optical flow for non overlapping 15×15 windows, circle denotes origin of optical flow vector

1.1 Sphere and Synthetic Images

In figure 1 the resultant optical flow vectors can be seen for both the *sphere* images as the *synth* images. The sphere is turning to the right, which can be seen in the optical flow vectors on the sphere. Some of the flow vectors seem to be a bit off, but this is mainly because the windows were not centred at good features which would improve results. The synthetic image rotates slightly, and

this result can be seen on the right with optical flow vectors with a relatively small magnitude.

1.2 Optical flow on the model house sequence

The model house sequence consist of 101 frames with 215 given points which can be tracked. In figure 2 the 100th frame can be seen with the ground truth and the corresponding points as tracked by the algorithm. The final sum of squared errors is 337.3183 and the median squared error is 17.3528. The main contributor to the error are points which drift into "flat" regions in which no motion can be observed causing the points to further drift away from the ground truth. Another contributor is the *jpg* compression of the sequence, causing noise in the data which result in less precise optical flow vectors.

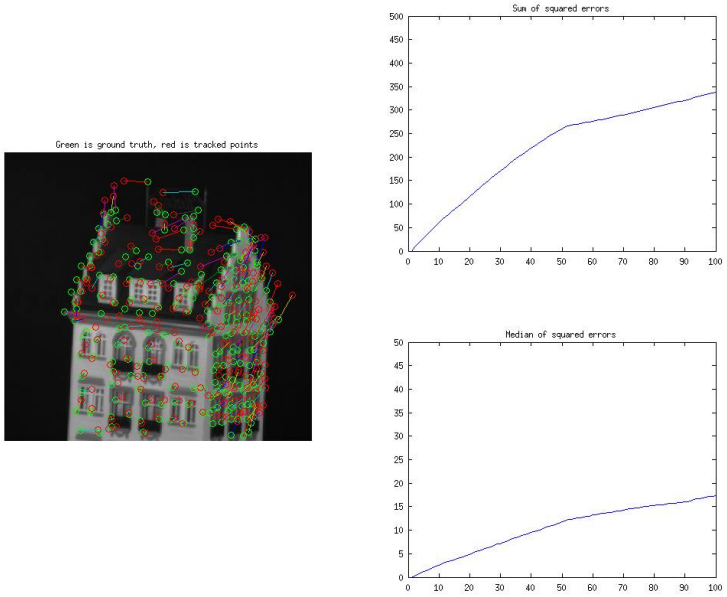


Figure 2: Frame 100 from the model house sequence, green circles represent ground truth, red circles are the points tracked over all 100 frames, lines connect corresponding points, the graphs represent the sum of squared errors and the median of squared errors over the sequence

A movie was created of the output frames: `opflow.avi` in which this effect can be clearly seen. This is especially the case for the points on the left of the house which drift into the dark background.

2 Structure from Motion

We implemented Structure from Motion (SfM) by the Tomasi-Kanade Factorization method. We first generate a measurement matrix D, either by reading the ground truth from disk or using the outputs of our Lucas-Kanade point tracker. This matrix has the image (x, y) coordinates of point j in view i at $D_{2*i,j}$ and $D_{2*i+1,j}$. We construct the Shape matrix S and Motion matrix M by using SVD on this matrix (distributing diagonal matrix W equally among the two).

The points can be plotted now (they are simply the columns of S), but the resulting figure is affinely skewed from the ground truth, see figure 3. The reason is that we have so far only imposed affine constriants. To retrieve the true structure, we further impose Euclidean constraints as follows.

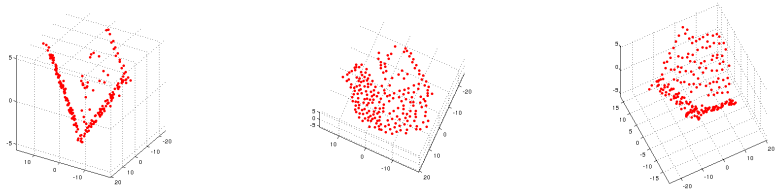
We can split M into chunks A_i, that multiply each 3D point to produce its 2D projection on image i. The rows of this matrix correspond to the image axes, which we will constrain to be of unit-length and orthogonal to each other: $a_1^T a_2 = 0$, $a_1^T a_1 = 1$ and $a_2^T a_2 = 1$. If we transform these by a matrix Q, we get the transformed constraint $(Qa_1)^T(Qa_2) = a_1^T(Q^TQ)a_2 = 0$, and similarly for the others.

Let L be the symmetric matrix $L = QQ^T$, then we have three linear equations in L for each camera. We could write this out, but it is a big 3×6 matrix with entries that are quadratic in a_i. So instead we refer the reader to `TomasiKanadeFactorization.m`, where this is implemented.

We stack each of these equations in a large $3m \times 6$ matrix (3 equations for each camera, 6 unknowns in a 3×3 symmetric matrix), and solve for the parameters of L using least-squares. We then factorize $L = QQ^T$ using Cholesky decomposition, and update $M = MQ$ and $S = Q^{-1}S$. The result is show in figure 4.

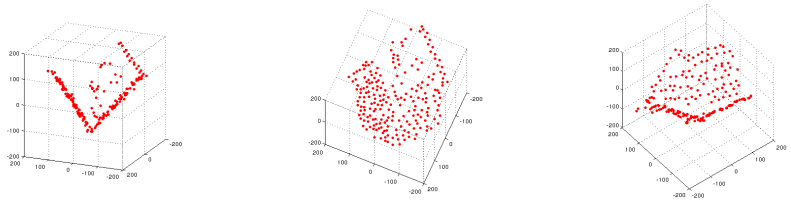
Finally, we applied the Tomas-Kanade algorithm to the output of the Lucas-Kanade algorithm. The results are a bit more noisy, as can be seen in figure 5.

The code can be tested by running `SFMdemo.m`, which plots Affine structure from ground-truth, Euclidean structure from ground-truth, Affine structure from LK tracker, and Euclidean structure from LK tracker. When viewing results, please select the rotation tool, then right-click and select Rotate Options/Fixed Aspect Ratio.



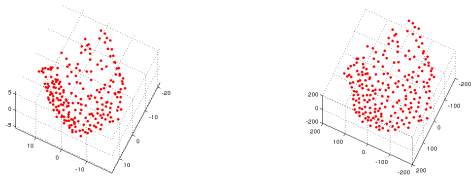
(a) View 1 (b) View 2 (c) View 3

Figure 3: Affine structure from motion (using groud-truth points as input). Notice the affine skew, which is particularly obvious in view 1.



(a) View 1 (b) View 2 (c) View 3

Figure 4: Euclidean structure from motion (using ground-truth points as input). Notice that the true Euclidean structure has been recovered: the walls are now orthogonal.



(a) Affine (b) Euclidean

Figure 5: Affine and Euclidean structure from motion, using the result of the LK tracker as input.