Supplementary Material for "Feature-Based Multi-Video Synchronization with Subframe Accuracy"

A. Elhayek, C. Stoll, K. I. Kim, H.-P. Seidel, C. Theobalt

MPI Informatik
{elhayek, stoll, kkim, hpseidel, theobalt}@mpi-inf.mpg.de

This appendix presents a summary of our synchronization algorithm (Sec. A) and additional experimental results (Sec. B).

A Summary of the proposed algorithm

Our two-video synchronization algorithm is summarized in Fig. 1(b) and Tables 1, while the multi-video version of the algorithms is summarized in Tables 2. It should be noted that the first three steps of multi-video synchronization algorithm is identical to those of two-video algorithm.

Table 1. Two-video synchronization algorithm

1. Extract features from each frame
2. Construct in-sequence trajectories
3. Filter out trivial trajectories:
   - short trajectories
   - space-static trajectories
   - epipolar-static trajectories
4. Build a table of tentative matches based on epipolar geometry
5. RANSAC-based optimization:
   (a) Randomly sample two pairs of matching trajectories and estimate parameters accordingly
   (b) Compute the number of inliers
   (c) Repeat steps (5.a) and (5.b) and choose the parameters which show the highest number of inliers
6. (optionally) Refine the RANSAC estimate using continuous optimization

B Additional Results

Table 3 summarizes the results of additional two-video experiments. These results are in accordance with Table 1 of the main paper and support the introduction of filtering stages. We were not able to perform any experiments without static filtering for the case

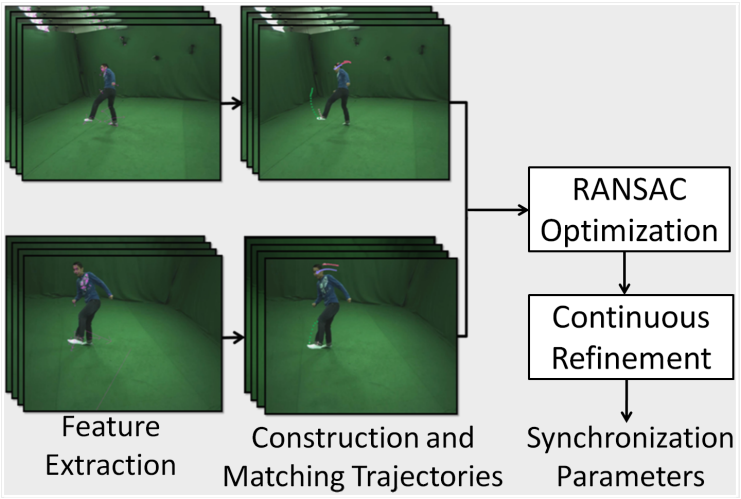

Fig. 1. Two-video synchronization overview. We extract feature-points from each frame (left), construct in-sequence trajectories by matching these features across consecutive frames, and use epipolar matching to find the corresponding trajectories between two videos (middle). Then, the synchronization parameters are estimated based on RANSAC optimization, which are refined by continuous optimization (right).

of $\mathcal{S}^2$ since the videos in this set contain many trajectories and accordingly the number of potential matches were prohibitively high.

Figure 2 shows examples of connectivity graph constructed based on our algorithm while Fig. 3 shows an example of three-video synchronization with synchronized frames.

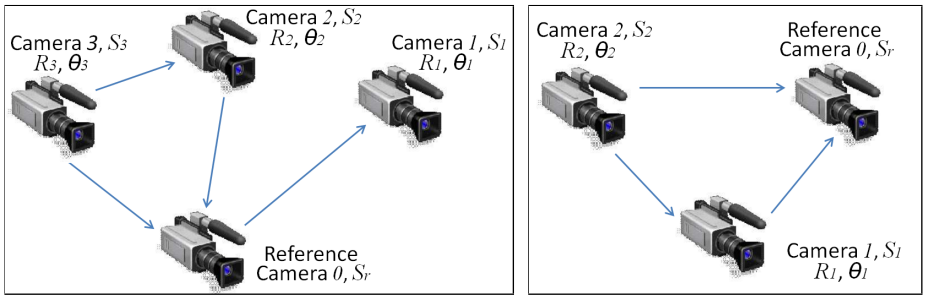

Fig. 2. Connectivity graphs resulting from our algorithm for $\mathcal{S}^1$ (left) and $\mathcal{S}^2$ (right).

Fig. 3. An example of synchronization for three videos ($\mathcal{S}^2$). Each frame (taken from each synchronized video) has the same global time-coordinate. The average video length is 280 frames, while the frames resolution is 1296 x 968.

C Epipolar Filter

This filter removes all trajectories which are nearly parallel to their epipolar lines defined by the fundamental matrix of the camera pair, as its not possible to distinguish any motion along that line in the other camera. Fig. 4(a) shows an example of such a trivial trajectory where we check the angles between the tangent to the trajectory at point p_2 (i.e. the line defined by p_2 and p_3) and the epipolar line for p_2 (the line defined by p_2 and the epipole e_x). Note that, for each point in this trajectory, the sum of the angles is too small which may lead to many incorrect matches with trajectories along L_y in the other sequence. Therefore, we filter this trajectory before the epipolar matching step of our algorithm. On the other hand, Fig. 4(b) shows an example of a non-trivial trajectory where the sum of the angles is large (e.g. the angle corresponding to p_4). We reject a trajectory if the score $\sum_{t_i \in sup(T_i)} 1 - cos(angle)$ is less than 0.32.

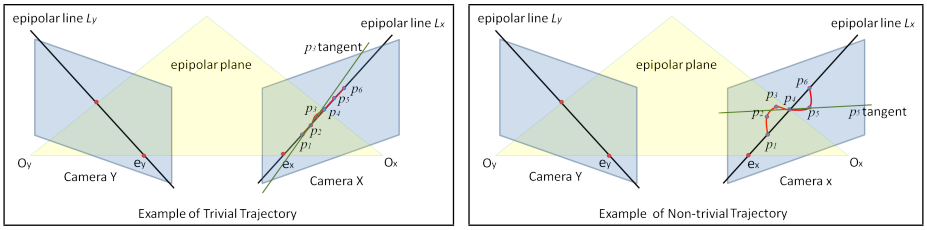

Fig. 4. Epipolar trivial trajectory filter. **(a) Left:** Example of trivial trajectory in camera X which is nearly parallel to its epipolar line L_x defined by the fundamental matrix of this camera pair. This trajectory may match any trajectory along L_y. However, it can be filtered by checking the angles between the tangents of each point along the trajectory (e.g. the green line for point p_2) and the epipolar line of these points (e.g. L_x for p_2). **(a) Right:** Example of non-trivial trajectory where the angles between the tangents of the trajectory points and there epipolar lines are large.

Table 2. Multi-video synchronization algorithm

1. Extract features from each frame
2. Construct in-sequence trajectories
3. Filter out trivial trajectories
4. Build connectivity graph:
   (a) Build a table of tentative matches for each pair of cameras
   (b) Remove edges between distant cameras (i.e. the camera pairs with low number of tentative matching trajectories)
5. RANSAC-based optimization:
   (a) Estimate the synchronization parameters based on a random spanning tree of the graph
   (b) Compute the number of inliers from the table of every tentative matches
   (c) Repeat steps (5.a) and (5.b) and choose the parameters which show the highest number of inliers
6. (optionally) Refine the RANSAC estimate using continuous optimization

Table 3. The results of additional two-video synchronization experiments.

Video pairs	Experimental setup description	Ground truth	Residual error (θ_i/R_i)	Average frame error	Maximum frame error
S_0^1,S_3^1	Without static filtering	80.00/2	0.93/0.020	1.27	3.03
	Without epipolar filtering		5.42 / 0.179	13.22	30.17
	Complete algorithm		0.71 / 0.005	0.29	0.711
	With given R_i		1.50 / 0.000	1.50	1.50
S_2^2,S_3^2	Without epipolar filtering	29.11/1	0.95/ 0.001	1.09	1.24
	Complete algorithm		0.52 / 0.004	0.24	0.52
	With given R_i		0.89 / 0.000	0.89	0.89
S_0^2,S_4^2	Without epipolar filtering	27.38/1	3.70 / 0.066	3.93	8.56
	Complete algorithm		0.46 / 0.026	3.65	6.83
	With given R_i		1.08 / 0.000	1.08	1.08
S_1^1,S_2^1	Without static filtering	60.00/1	1.61 / 0.026	1.46	3.52
	Without epipolar filtering		1.29 / 0.022	1.31	3.16
	Complete algorithm		1.32 / 0.023	1.35	3.26
	With given R_i		2.13 / 0.000	2.13	2.13