3D Motion Segmentation of Articulated Rigid Bodies based on RGB-D Data - Supplementary Material

Urbano Miguel Nunes
um.nunes@imperial.ac.uk
Yiannis Demiris
y.demiris@imperial.ac.uk

Personal Robotics Laboratory
Imperial College London
London, United Kingdom

1 Influence of the Number of Frames

In the present work, the data matrix, as in Eq. (1), depends on the number of frames F of the sequence of RGB-D images, since $\mathbf{X} \in \mathbb{R}^{3F \times N}$. In the general case where the optimization program (2) must be solved, each frame induces three constraints that must be satisfied. Thus, it is natural to assume that the complexity and processing time of the algorithm increase with the number of frames to be considered.

However, experimentally that was not observed, as shown in Fig. 3 and Table 3, where the number of frames considered was $2F$ (*i.e.* double the original number of frames F, which corresponds to watching the sequence forward and then backward). As illustrated in Fig. 3, the proposed method's time required to process a sequence of F or $2F$ frames is identical. The average processing time per point is also similar, as presented in Table 3 (*e.g.* for F frames the average processing time per point is 1.32 msec, whereas for $2F$ frames it is 1.37 msec).

The reason for this is related to which method is used to solve the sparse subspace optimization problem (2). Since the problem is convex, one could use any generic convex solver. However, as suggested in [1], an Alternating Direction Method of Multipliers (ADMM) was adopted for an efficient implementation of the proposed problem. In the present work, since we assume complete point trajectories (*i.e.* without the $\mathbf{E}$ term, since we are not considering outliers), Eq. (2) becomes

$$\begin{aligned} &\min \|\mathbf{C}\|_1 + \frac{\lambda_z}{2}\|\mathbf{Z}\|_F^2 \\ &\text{s. t.} \quad \mathbf{X} = \mathbf{X}\mathbf{C} + \mathbf{Z}, \quad \mathbf{C}^\top \mathbf{1} = \mathbf{1}, \quad \operatorname{diag}(\mathbf{C}) = \mathbf{0}. \end{aligned} \tag{9}$$

By introducing auxiliary variables and Lagrange multipliers into the optimization program, we can obtain the following associated Lagrangian function of Eq. (9) (please refer to [1] for more details):

$$\begin{aligned} \mathcal{L}(\mathbf{C},\mathbf{A},\delta,\Delta) = \|\mathbf{C}\|_1 + \frac{\lambda_z}{2}\|\mathbf{X}-\mathbf{X}\mathbf{A}\|_F^2 + \frac{\rho}{2}\|\mathbf{A}^\top\mathbf{1}-\mathbf{1}\|_2^2 + \frac{\rho}{2}\|\mathbf{A}-\mathbf{C}+\operatorname{diag}(\mathbf{C})\|_F^2 \\ +\delta^\top(\mathbf{A}^\top\mathbf{1}-\mathbf{1}) + \operatorname{tr}(\Delta^\top(\mathbf{A}-\mathbf{C}+\operatorname{diag}(\mathbf{C}))), \end{aligned} \tag{10}$$

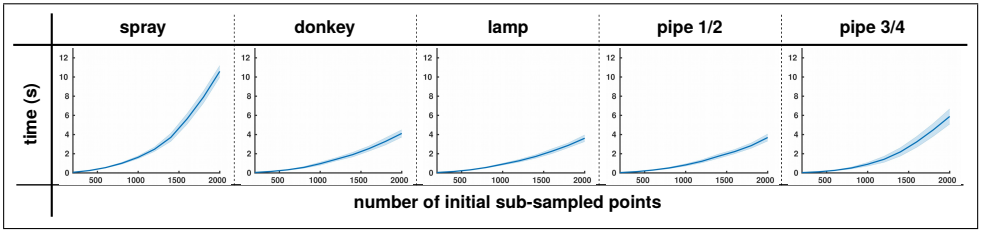

Figure 3: Quantitative results concerning the influence of the number of frames (*i.e.* $2F$ frames) on processing time. These results were obtained by varying the number of initial randomly sub-sampled points across one hundred trials per number. The mean is represented by the solid line and the standard deviation is represented by the surrounding shaded area.

	Donkey	Lamp	Pipe 1/2	Pipe 3/4	Spray	Average
Time per point (msec)	1.14	1.03	1.00	1.32	2.36	1.37

Table 3: Summary of average processing time per point obtained.

where $\mathbf{A} \in \mathbb{R}^{N\times N}$ is an auxiliary matrix, $\rho > 0$ is a penalty parameter, $\delta \in \mathbb{R}^N$ and $\Delta \in \mathbb{R}^{N\times N}$ are a vector and matrix of Lagrange multipliers, respectively, $\mathrm{tr}(\cdot)$ denotes the trace operator of a matrix and we abuse the notation of $\mathrm{diag}(\mathbf{C})$ throughout this section to denote a vector containing the diagonal elements of matrix $\mathbf{C}$, as well as a diagonal matrix whose elements correspond to the entries of the diagonal of $\mathbf{C}$. This function can be iteratively solved via an ADMM approach, until convergence or the number of iterations reaches a certain number:

- Update $\mathbf{A}^{(k+1)}$ as

$$(\lambda_z \hat{\mathbf{X}} + \rho\mathbf{I} + \rho\mathbf{1}\mathbf{1}^\top)\mathbf{A}^{(k+1)} = \lambda_z \hat{\mathbf{X}} + \rho(\mathbf{1}\mathbf{1}^\top + \mathbf{C}^{(k)}) - \mathbf{1}\delta^{(k)\top} - \Delta^{(k)}, \tag{11}$$

  where $\hat{\mathbf{X}} = \mathbf{X}^\top\mathbf{X} \in \mathbb{R}^{N\times N}$ and $\mathbf{I} \in \mathbb{R}^{N\times N}$ denotes the identity matrix;

- Update $\mathbf{C}^{(k+1)}$ as

$$\mathbf{C}^{(k+1)} = \mathbf{J} - \mathrm{diag}(\mathbf{J}), \quad \mathbf{J} = \mathcal{T}_{\frac{1}{\rho}}(\mathbf{A}^{(k+1)} + \frac{\Delta^{(k)}}{\rho}), \tag{12}$$

  where $\mathcal{T}_\eta(v) = (|v| - \eta)_+ \mathrm{sgn}(v)$ and $(\cdot)_+$ returns its argument if it is non-negative or zero otherwise;

- Update $\delta^{(k+1)}$ as

$$\delta^{(k+1)} = \delta^{(k)} + \rho(\mathbf{A}^{(k+1)\top}\mathbf{1} - \mathbf{1}); \tag{13}$$

- Update $\Delta^{(k+1)}$ as

$$\Delta^{(k+1)} = \Delta^{(k)} + \rho(\mathbf{A}^{(k+1)} - \mathbf{C}^{(k+1)}). \tag{14}$$

We can now verify that every matrix involved in the optimization procedure is of size $N \times N$, thus not depending on the number of frames. Moreover, the auxiliary matrix $\hat{\mathbf{X}}$, which is the product of two matrices that depend on the number of frames, can be computed once before the actual iterative computation. In other words, the performance of the method is only limited by one initial matrix multiplication that depends on the number of frames considered, which in practice is negligible, since we may assume $N > F$. Therefore, the number of frames does not influence significantly the performance of the algorithm in terms of processing time, as corroborated experimentally.

References

[1] Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11): 2765–2781, November 2013.