Dense Batch Non-Rigid Structure from Motion in a Second

Vladislav Golyanik Didier Stricker
Department of Computer Science, University of Kaiserslautern
Department Augmented Vision, DFKI Kaiserslautern
{Vladislav.Golyanik, Didier.Stricker}@dfki.de

Abstract

In this paper, we show how to minimise a quadratic function on a set of orthonormal matrices using an efficient semidefinite programming solver with application to dense non-rigid structure from motion. Thanks to the proposed technique, a new form of the convex relaxation for the Metric Projections (MP) algorithm is obtained. The modification results in an efficient single-core CPU implementation enabling dense factorisations of long image sequences with tens of thousands of points into camera pose and non-rigid shape in seconds, i.e., at least two orders of magnitude faster than the runtimes reported in the literature so far. The proposed implementation can be useful for interactive or real-time robotic and other applications, where monocular non-rigid reconstruction is required. In a narrow sense, our paper complements research on MP, though the proposed convex relaxation methodology can also be useful in other computer vision tasks. The experimental part providing runtime evaluation and qualitative analysis concludes the paper.

1. Introduction

Template-free deformable surface reconstruction from monocular image sequences, referred to as Non-Rigid Structure from Motion (NRSfM) experienced significant advances during the last ten years. Previously being able to retrieve sparse structures under small non-rigid deformations, NRSfM methods nowadays can recover dense surfaces exhibiting large deformations. While entering the realm of dense reconstructions, the computational time of NRSfM methods has increased significantly due to the inherent ill-posedness of the problem, higher complexity of the models, and high computational complexity of the optimisation methods. This tendency is aside from the increased pre-processing time for the dense correspondence establishment, which the vast majority of NRSfM methods rely on. At the same time, many robotics and medical application require not only dense reconstructions but also impose harsh timing constraints — either new reconstructions need to be obtained upon arrival of a new frame (real-time requirement), or reconstructions are required within an arguably reasonable time interval after the image sequence is acquired (requirement of interactivity).

Among NRSfM algorithms, Metric Projections (MP) [26] possesses properties which make it considerable for interactive, real-world applications such as high reconstruction accuracy, feasible computational complexity, availability of efficient and fast optimisation methods, notable scalability with the number of points as well as robustness to noisy and missing data. An important for this paper aspect is that MP requires solving a quadratic optimisation problem on the set of orthonormal matrices of the following form:

$$\min_{\mathbf{q}=vec(Q)} \mathbf{q}^\top \mathrm{E}\mathbf{q}, \tag{1}$$

where $\mathrm{E} \in \mathbb{R}^{6\times6}$ and $Q \in \mathbb{R}^{3\times2}$ is an orthonormal matrix. According to [26], the problem in Eq. (1) can be convex-relaxed and approximated as

$$\min_{\mathbf{q}=vec(Q)} \mathrm{tr}(\mathbf{E}\mathbf{q}\mathbf{q}^\top) = \min_{\mathbf{X}\in co(S)} \mathrm{tr}(E\mathbf{X}), \tag{2}$$

where $co(S)$ is approximated by a set of real symmetric matrices $X \in \mathbb{R}^{6\times6} = \begin{pmatrix} A & B \\ B^\top & C \end{pmatrix}$, whereby $A \in \mathbb{R}^{3\times3}$, $X \succcurlyeq 0$, $\mathrm{tr}(A) = \mathrm{tr}(C) = 1$, $\mathrm{tr}(B) = 0$, with an additional constraint on the combined matrix Y:

$$Y = \begin{pmatrix} I_3 - A - C & \mathbf{w} \\ \mathbf{w}^\top & 1 \end{pmatrix} \succcurlyeq 0, \tag{3}$$

with $\mathbf{w} = \begin{pmatrix} b_{23} - b_{32} \\ b_{31} - b_{13} \\ b_{12} - b_{21} \end{pmatrix}$ (see Sec. 3 for further details).

The problem in Eq. (2) is a Semidefinite Programming (SDP) problem. In an SDP, a linear objective function with linear constraints over a set of Positive Semidefinite (PSD) matrices needs to be optimised [21]. SDP is a field of convex optimisation which finds applications in various areas

of science and engineering ranging from the approximation of NP-hard combinatorial problems (max-cut, graph bisection, min-max eigenvalue problems) [21] and quantum complexity [7], to the theory of automatic control [27]. An SDP problem can be solved using a Matlab SDP solver, e.g., SeDuMi [30] which allows specifying constraints on a high level of abstraction directly in terms of traces, block matrices and values of particular elements. However, for an efficient C++ SDP solver such as CSDP [9], an SDP problem must be specified in a general form as

$$\begin{aligned} &\max\ \mathrm{tr}(C'U), \\ &\text{subject to } \Big\{\mathcal{A}(U) = a;\ U \succcurlyeq 0\Big\}, \end{aligned} \tag{4}$$

where

$$\mathcal{A}(U) = \begin{bmatrix}\mathrm{tr}(A_1U) & \mathrm{tr}(A_2U) & \dots & \mathrm{tr}(A_mU)\end{bmatrix}^\top. \tag{5}$$

In the optimisation model given by Expr. (4), the matrices $C', A_1, A_2, \dots, A_m$ as well as the vector a are known, and the operator $\mathcal{A}$ maps the unknown PSD matrix U to a vector.

Thus, the existing form of the convex relaxation as presented in Eq. (2) does not allow to employ CSDP and implement MP in C++ efficiently. A drawback is that robotic and medical applications do not often allow to combine Matlab and C++ source code on an embedded platform due to performance and memory issues. Thus, research on MP algorithm can not be considered as accomplished yet. Moreover, both academia and industry are interested in an efficient monocular NRSfM which supports real-time frame rates. Accordingly, our contributions in this paper are as follows:

- We show how to formulate the convex optimisation problem given in Eq. (2) in terms of the standard SDP problem as accepted by an efficient SDP solver (Expr. (4)). The shown methodology can also be useful in a broad range of algorithms taking advantage of SDP. In this paper, it is applied to an optimised CPU-only implementation of MP which we refer to as Accelerated Metric Projections (AMP)[1].
- We demonstrate experimentally that AMP is suitable for dense factorisation of long image sequences with tens of thousands of points in seconds, and compare runtimes of AMP with an optimised implementation of the Variational NRSfM Approach (VA) [17].

In the performed experiments, AMP finishes up to 20 times faster than an optimised implementation of VA. Considering the achieved performance, AMP can be useful as a building block in a broad range of real-time and interactive applications including medical and robotic ones which require monocular non-rigid reconstruction. AMP can also be used for initialisation of more accurate but computationally expensive, perhaps sequential NRSfM algorithms.

The remainder of this paper includes an overview of previous and related works (Sec. 2), proposed convex relaxation (Sec. 3), implementation details (Sec. 4), experiments, discussion (Sec. 5) and conclusion (Sec. 6).

2. Previous and related work

SDP was extensively applied in computer vision [23, 1, 22, 24] including NRSfM [26, 13, 12]. We noticed, on the one hand, that a detailed consideration of SDP problems and their efficient solutions for computer vision tasks were rather sparsely discussed in the literature. On the other hand, research in the area of NRSfM was mainly focused on algorithmic improvements which were not always accompanied by an appropriate consideration of efficient optimisation methods. As a result, it is difficult to find high-performance implementations able to handle dense data in a few seconds among those shown in the literature, although several algorithms are potentially capable of it [28, 17, 3].

Several algorithms were proposed for sparse and dense NRSfM over the recent years. Most NRSfM methods require point correspondences as an input which are obtained in a pre-processing step by sparse point trackers or dense optical flow methods. In contrast to sparse NRSfM, the dense counterpart allows reconstructing all visible points of interest when considering reconstruction from image sequences (either visible in a reference frame and tracked through the sequence, or visible at any moment in time). Sparse NRSfM was initially proposed by Bregler *et al*. in 2000 [10], and the first dense implementation emerged a dozen years later [28]. Previous work was focusing on algorithmic improvements which allowed to handle more complex deformations [32] and the higher number of points, reduce 3D RMS error (see Sec. 5 for the definition), perform reconstructions in an online manner [2] and introduced new models for NRSfM [4]. Efficient implementations of the techniques remained in the background. Though, an efficient implementation is often not possible without changes in underlying optimisation methods. We discover that this is the case with the MP algorithm. The optimisation proposed in this paper allows to fully unfold the strength of the approach regarding the runtime. Whereas [26] reports around 30 seconds for sequences with 37 points tracked throughout 74 frames, AMP allows reconstructing $5 \cdot 10^4$ points tracked throughout 202 frames in 30 seconds. Despite the original MP implementation was performed in Matlab, the difference with our implementation can not be barely explained by a re-implementation in another programming language (C++ in our case) or an improved hardware during the last 48 months between [26] and our submission (recall that an efficient implementation in C++ was not earlier possible). Various runtimes for dense NRSfM were reported for other

[1] we provide an AMP executable for academic use upon request.

algorithms in the literature so far. Thus, [28] reports 600 seconds for 90 frames with $7.8 \cdot 10^4$ points per frame. [34] reports 720 seconds for 50 frames with 540 points in every frame. The sequential algorithm of Agudo *et al.* [3] achieves 62 seconds per frame for the dense flag sequence [18] with $\sim 10^4$ points. An efficient implementation of this method could potentially run in real-time. Nevertheless, the runtimes reported in the literature are still at least two orders of magnitude higher than the runtime reported in this paper, considering comparable number of points and frames (e.g., for the synthetic flag sequence [18]).

3. Accelerated Metric Projections

First, we summarise the core MP method. The input of MP is a measurement matrix $\mathbf{W}$ which combines image coordinates of the tracked points for all frames:

$$\mathbf{W}_{2f\times p} = [\mathbf{W}_1 \mathbf{W}_2 \dots \mathbf{W}_f]^\top, \tag{6}$$

where $\mathbf{W}_i = [\mathbf{w}_{ij}] = [(u_{ij} v_{ij})^\top]$, $i \in \{1, \dots, f\}$ is a frame index and $j \in \{1, \dots, p\}$ is a point index. Values in $\mathbf{W}$ are registered to the centroid of the scene. MP adopts the low-rank shape model introduced in [10] so that every non-rigid shape $\mathbf{S}_i$ observed in frame i can be expressed as

$$\mathbf{S}_i = \sum_{d=1}^{k} l_{id} \mathbf{B}_d, \tag{7}$$

i.e., a linear combination of the basis shapes $\mathbf{B}_d$ with the weights l_{id}, $d \in \{1, \dots, k\}$. Since $\mathbf{W}_i = \mathbf{R}_i \mathbf{S}_i$ defines a 2D scene observed by an orthographic camera $\mathbf{R}_i$ for every frame, $\mathbf{W}_i$ can be written as

$$\mathbf{W}_i = \begin{bmatrix} l_{i1}\mathbf{R}_i & \dots & l_{ik}\mathbf{R}_i \end{bmatrix} \begin{bmatrix} \mathbf{B}_1 & \dots & \mathbf{B}_k \end{bmatrix}^\top, \tag{8}$$

and for the whole $\mathbf{W}$ as

$$\mathbf{W} = \mathbf{M}\mathbf{B} = \begin{bmatrix} \mathbf{M}_1 \\ \vdots \\ \mathbf{M}_f \end{bmatrix} \begin{bmatrix} \mathbf{B}_1 \\ \vdots \\ \mathbf{B}_k \end{bmatrix}. \tag{9}$$

Thus, MP factorises $\mathbf{W}$ into the motion matrix $\mathbf{M}$ (it contains camera poses $\mathbf{R}_i$ multiplied with weights l_{ik}) and a set of the basis shapes $\mathbf{B}$ using alternating least squares, i.e., either $\mathbf{B}$ or $\mathbf{M}$ is fixed while for the other one is being optimised. Further, the motion matrix $\mathbf{M}$ is projected onto the motion manifold (i.e., a Lie group with augmented scalar weights) which guarantees the correct block structure of the motion matrix and camera pose matrices contained in it. The projection is performed for each $l_{id}\mathbf{R}_i$ submatrix of $\mathbf{M}_i$ individually and can be written in a form of an optimisation problem as

$$\min_{\mathbf{R}_i, l_{i1}, \dots, l_{ik}} \|\mathbf{M}_i - [l_{i1}\mathbf{R}_i | \dots | l_{ik}\mathbf{R}_i]\|_{\mathcal{F}}^2 \tag{10}$$

or, after reordering, writing out the squared Frobenius norm and eventually bringing the expression to a quadratic form:

$$\min_{\mathbf{R}_i} \mathbf{r}_i^\top \left[-\sum_{d=1}^{k} \mathbf{m}_{id} \mathbf{m}_{id}^\top \right] \mathbf{r}_i \quad \text{so that} \tag{11}$$

$$\mathbf{R}_i \mathbf{R}_i^\top = \mathbf{I}_{2\times 2}, \tag{12}$$

with $\mathbf{r}_i = \text{vec}(\mathbf{R}_i^\top)$ and $\mathbf{m}_{id} = \text{vec}(\mathbf{M}_{id}^\top)$. The quadratic form in Eq. (11) has the form as in Eq. (1) and is convex-relaxed as stated in Sec. 1, Eqs. (1)–(3). A detailed derivation of the convex relaxation is given in the recent work by Dodig *et al.* [15].

Solving Eq. (11) leads to a provably optimal $\mathbf{M}_i$ update. As proposed in [26], we use a warm-start strategy in combination with a Newton-like iterative optimisation approach [16], i.e., we compute only $\mathbf{M}_0$ using the convex-relaxed Eq. (11) is each iteration, and the remaining $\mathbf{M}_i$ matrices are obtained based on the optimal $\mathbf{M}_0$ estimate. Despite this strategy is theoretically suboptimal, it was empirically shown to converge to a local minimum in multiple experiments while significantly reducing the overall runtime [26].

Once recovered, the $\mathbf{R}_i^\top$ are used to update the weights:

$$l_{id} = \frac{1}{2} \text{tr}\left[\mathbf{M}_{id}^\top \mathbf{R}_i\right]. \tag{13}$$

Given the weights l_{id}, projection of $\mathbf{M}$ onto the motion manifold is complete — which, in turn, allows to obtain a current estimate of $\mathbf{S}$ and an update of $\mathbf{M}$. MP is an iterative approach, and requires an initial estimate of $\mathbf{M}$. Further details on the core MP method can be found in [26].

Secs. 3.1 and 3.2 describe how Eq. (1) can be convex-relaxed and brought to the general form as accepted by an efficient C++ SDP solver. The modification given in the following leads to the proposed AMP method.

3.1. Coefficient splitting

A symmetric matrix $Z = [z_{ij}]$ equals to its own transpose, i.e., $z_{ij} = z_{ji}$. Moreover, the following property for symmetric matrices holds:

$$\text{tr}(AZ) = \sum_{j=1}^{n} \sum_{i=1}^{n} a_{ij} z_{ij}, \tag{14}$$

where $A = [a_{ij}]$ is a real square matrix. Suppose a constraint is given in the form

$$c_{11}z_{11} + c_{12}z_{12} + \dots + c_{1n}z_{1n} + \dots + c_{nn}z_{nn} = b. \tag{15}$$

The key observation is that the constraint in Eq. (15) can be written with a *coefficient splitting* as

$$\sum_{i=1}^{n} c_{ii} z_{ii} + \sum_{i\neq j} \frac{c_{ij} + c_{ji}}{2} (z_{ij} + z_{ji}) = b \tag{16}$$

(again, this holds for a symmetric matrix $Z = [z_{ij}]$). Considering Eq. (14), Eq. (16) can be written as $\mathrm{tr}(AZ) = b$, where

$$A = \begin{pmatrix} c_{11} & \frac{c_{12}+c_{21}}{2} & \frac{c_{13}+c_{31}}{2} & \dots \\ \frac{c_{12}+c_{21}}{2} & c_{22} & \frac{c_{23}+c_{32}}{2} & \dots \\ \frac{c_{13}+c_{31}}{2} & \frac{c_{23}+c_{32}}{2} & c_{33} & \dots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}. \quad (17)$$

In Eq. (17), only the divisor value 2 preserves matrix symmetry, i.e., an arbitrary re-distribution of the coefficients is not possible. In particular, if $c_{ii} = 1$, then $a_{ii} = 1$; if $c_{ij} = 1$, then $a_{ij} = a_{ji} = 0.5$ $(i \neq j)$; if $c_{ij} = 0$, then $a_{ij} = a_{ji} = 0$ $(i \neq j)$. Note that a constraint can also be formulated in terms of an upper/lower triangular submatrix of X. In this case, if c_{ij} defines a coefficient, then c_{ji} always equals to 0 (no coefficient splitting is required).

3.2. Constraints in the unified form

Now, using the properties of the matrix trace (Sec. 3.1), we can advance the convex relaxation of the quadratic optimisation problem given in Eq. (2) into the standard form of an SDP problem given by Expr. (4). The main matrices — on which elements (submatrices or particular values) constraints are imposed — are X and Y. In the following, we will use the proposition (see App. A for the proof):

Proposition 1 *If a matrix* $U = \begin{pmatrix} U_1 & 0 \\ 0 & U_2 \end{pmatrix}$ *has a block structure, then* U *is PSD if both* U_1 *and* U_2 *are PSD. Conversely, if* U *is PSD, then the principal submatrices*[2] U_1 *and* U_2 *are PSD.*

Using Prop. 1 we can join X and Y matrices into a block matrix U and demand its positive semidefiniteness:

$$U = \begin{bmatrix} X_{6\times6} & 0 \\ 0 & Y_{4\times4} \end{bmatrix} = \begin{bmatrix} A_{3\times3} & B & 0 \\ B^\top & C_{3\times3} & 0 \\ 0 & 0 & Y_{4\times4} \end{bmatrix} \overset{!}{\succcurlyeq} 0. \quad (18)$$

Thus, if the matrix U will be found, X and Y will be guaranteed to be PSD.

Constraints on the traces. The SDP problem we are interested in, contains three constraints on the matrix trace. Constraint $\mathrm{tr}(A) = 1$ can be rewritten as $\sum_i^3 a_{ii} = 1$ or equivalently as

$$\mathrm{tr}(A_1 U) = \mathrm{tr}\left(\begin{bmatrix} \mathbf{I}_3 & 0 \\ 0 & \mathbf{0}_{7\times7} \end{bmatrix} U\right) = 1. \quad (19)$$

[2] a principal submatrix is a square submatrix obtained by removing k rows and columns with the same indexes (whenever i-th row is removed from U, the i-th column is also removed).

In the same manner, we obtain constraint expressions for $\mathrm{tr}(C) = 1$:

$$\mathrm{tr}(A_2 U) = \mathrm{tr}\left(\begin{bmatrix} \mathbf{0}_{3\times3} & 0 & 0 \\ 0 & \mathbf{I}_3 & 0 \\ 0 & 0 & \mathbf{0_{4\times4}} \end{bmatrix} U\right) = 1 \quad (20)$$

and $\mathrm{tr}(B) = 0$:

$$\mathrm{tr}(A_3 U) = \mathrm{tr}\left(\begin{bmatrix} 0 & \frac{\mathbf{I}_3}{2} & 0 \\ \frac{\mathbf{I}_3}{2} & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} U\right) = 0. \quad (21)$$

In total, we obtain three constraints on the traces of the submatrices of X, and, accordingly, first three A_i matrices.

Constraints on Y. To formulate constraints on Y, we consider all its unique elements separately. Since Y is symmetric, it has ten non-repeating elements. We denote a_{ij} and c_{ij} elements of A and C in Eq. (2) and (3) respectively. Consider matrix Y in Eq. (3) element-wise. We see that the element y_{11} is constrained as

$$y_{11} = 1 - a_{11} - c_{11}. \quad (22)$$

Therefore, A_4 must fetch correct elements and equate them to 1. We obtain a sparse matrix with the elements $[1;1]$, $[4;4]$ and $[7;7]$ being equal to 1, and zeros otherwise. In a similar way, A_5 and A_6 for the elements y_{22} and y_{33} can be obtained, with non-zero elements shifted along the diagonal by 1 and 2 positions respectively. Accordingly, A_7 (element y_{44}) has only a single non-zero entry in the position $[10;10]$. We proceed further with the elements of matrix Y. Thus, $y_{12} = -a_{12} - c_{12}$. As these elements are not on the diagonal of U, coefficient splitting is required, i.e.,

$$\frac{u_{12}+u_{21}}{2} + \frac{u_{45}+u_{54}}{2} + \frac{u_{78}+u_{87}}{2} = 0, \quad (23)$$

and the constraint matrix A_8 reads as

$$A_8 = \begin{bmatrix} 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (24)$$

Note that since A_8 is symmetric, it expresses constraints on both elements y_{12} and y_{21}. For the elements y_{13}, y_{31}, $y_{23}, y_{32}, y_{14}, y_{41}, y_{24}, y_{42}$ and y_{34}, y_{43} we proceed in a similar manner and obtain A_9 – A_{13} respectively. For completeness, the remaining A_i matrices are given in our supplementary material. In total, we obtain 13 constraints $\mathrm{tr}(A_i U) = a_i$ with

$$a = (1\,1\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0)^\top. \quad (25)$$

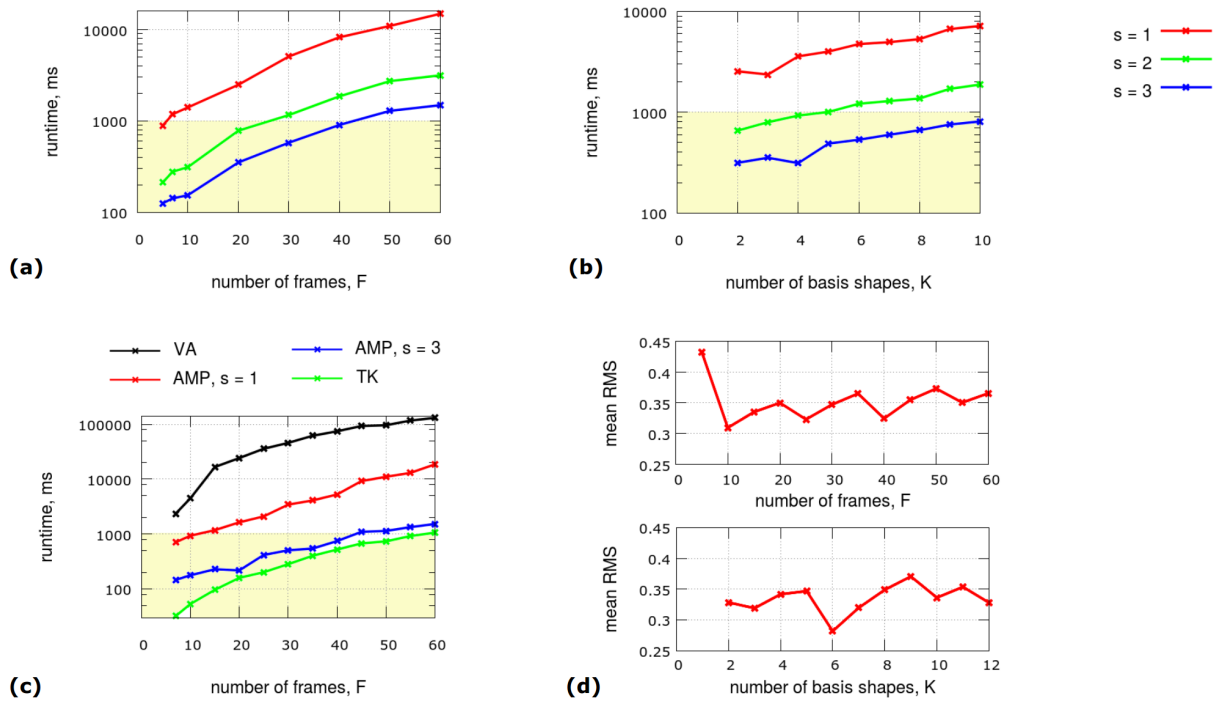

Figure 1: Results of the AMP runtime measurements on the synthetic flag sequence [18] for different subsampling factors (1 or w/o subsampling, 2 and 3) : (a) the number of basis shapes K is fixed to 3, the number of frames is varying; (b) the number of frames is fixed to 20, the number of basis shapes is varying. (c) runtime functions in the number of frames for VA [17], rigid factorisation [33] and AMP on the Xeon E5-1650 platform; AMP is an order of magnitude faster than VA; AMP achieves a comparable to rigid factorisation runtime if the number of points equals to approx. 10% compared to the TK case; (d) mean RMS error for AMP on the synthetic flag sequence as a function of the number of frames for $K = 4$ (top) and a function of the number of basis shapes for $F = 30$ (bottom). In (a-c), light yellow colour marks configurations which require less than a second to complete.

So far, we have all components required for the approximation of Eq. (1) formulated in the general form[3].

4. Implementation

In the AMP implementation, we use several lightweight libraries (e.g., *eigen3* [20] for linear algebra, *lapack* [5] especially for svd decomposition) as well as the CSDP library as an SDP solver. Our implementation is single core CPU only which enables compilation and execution on mobile platforms (though, a GPU can be used if available).

CSDP solver. CSDP is a C/C++ library for solving SDP problems [9, 8]. It is built upon the *lapack* [5] library and provides an efficient implementation of the Primal-Dual Interior Point (PDIP) algorithm for SDP proposed by Helmberg *et al*. [21]. PDIP has polynomial complexity. The *Interior point* means that the method converges to an optimum through the interior of the polyhedron (which is the solution space), whereas the optimum always lies on its surface. *Primal-dual* refers to the property of the algorithm to solve both the primal and the dual SDP problems simultaneously. This property together with additional feasibility constraints on the solution space brings the advantage of the increased stability as well as a faster convergence. CSDP enables SDP programs to be solved in polynomial time and the constraints are provided in the form as stated above in Expr. (4). In computer vision, CSDP was used for camera calibration [1], image clustering [24] and dimensionality reduction of image data [35]. The constraints as derived in Sec. 3.2 are provided to the CSDP library as a configuration file containing non-zero elements of A_i matrices.

5. Experiments

MP was extensively evaluated on sparse data sets [26], and played a role of the baseline in several works [17, 4]. In this section, we describe runtime evaluation of AMP in the dense setting and show some qualitative results. We test AMP on a mobile platform with 6 GB RAM and Intel i5-2410M processor running at 2.30 GHz. We also report runtimes on a more powerful desktop machine with 32 GB RAM, Intel Xeon E5-1650 and NVIDIA Titan X GPU for comparing AMP with rigid factorisation [33] and a heterogeneous implementation of VA [17]. We choose several sequences for the experiments: the *synthetic flag* [18], *music notes* [19], *monkaa* [25], *shaman2* [11], *barn*

[3] matrix C' in Expr. (4) equals to identity in our case.

Figure 2: Qualitative evaluation of AMP: (a) selected frames from *music notes* (top), *monkaa* (middle) [25] and *barn owl* (bottom) [14] sequences; (b) reconstructions of the above-mentioned sequences; the surfaces are shown pairwise as frontal and side views; (c) a selected reference surface reconstruction from the *shaman2* sequence [11] obtained by VA (top) and an AMP reconstruction shown in cyan overlayed with the reference shown in purple (bottom). The mean RMS error for this 30-frame long sequence amounted to 0.42.

owl [14], *face* [17] and *heart* [29] sequences.

Quantitative evaluation. For quantitative evaluation, we use the ground truth optical flow of the synthetic flag sequence available from [18], and run AMP in different modes. In the first mode, the number of basis shapes K is fixed to 3 and the number of frames is varied. In the second mode, the number of frames F is fixed to 20 and the number of basis shapes K is varied. In both modes, we perform the experiment for three different values of the correspondence subsampling factor s. For instance, if $s = 2$, correspondences are decimated so that every second point in both image directions is included into the input measurement matrix $\mathbf{W}$. When undecimated, reconstructions contain $8.2 \cdot 10^4$ points. During the execution, the influence of the operating system workloads is minimised through launching the executable file without a graphics environment in a command line terminal. We report average runtimes for 10 runs for every mode and every value of the respective variable. Fig. 1-(a),(b) illustrates results of the experiment for both modes.

As can be observed in Fig. 1-(a), runtime grows as a superlinear function of the number of frames. Depending on the number of points, a shift along the time axis is observed, whereas the function graph preserves its form. A similar effect if observed in Fig. 1-(b) for the case of varying number of basis shapes. Allowing more complex deformations results in a higher computational complexity for a fixed number of frames. However, in certain cases ($K = 3$ for $s = 1$ or for $K = 4$ for $s = 3$), deviations from the monotonous growth may occur. The deviations are explained by a faster convergence of the CSDP solver, due to better initialisations, and are possibly related to an optimal number of basis shapes for a given image sequence. From Fig. 1-(a),(b) it can also be seen that for many configurations, the execution time is below one second. Thus, for $2 \cdot 10^4$ points and 10 frames, the execution time amounted to 311.7 milliseconds which allows building a window-based approach on top of AMP and save processing time for other workloads. An optimal window size depends on an application and may vary from 5 to 15 frames.

Next, we compare runtimes of AMP, VA [17] and rigid Tomasi-Kanade (TK) factorisation approach [33] on the Intel Xeon platform with a GPU. We use own heterogeneous C++/CUDA C implementation of VA and an optimised C++ version of the rigid factorisation. VA was shown to outperform MP in terms of reconstruction accuracy for dense cases due to the Total Variation (TV) term [17]. However, due to the same reason — the TV term as well as the proximal splitting — it is computationally expensive. The runtime of TK factorisation serves as a lower runtime bound for every NRSfM approach. The runtimes as a function of number of frames for the comparison are plotted in Fig. 1-(c) (runtime comparison as functions of number of basis shapes for VA is omitted, since the number of basis shapes is determined in VA automatically as proposed by Dai *et al.* [13]). For VA, we set 10 iterations with 10 primal-dual alternations each, unless the algorithm converges in early iterations. Despite AMP is a single core CPU implementation, it finishes in average one order of magnitude faster than VA. On the other hand, TK finishes approximately ten times faster than AMP. Using the decimation factor of 3, the runtimes of AMP are approaching the runtimes of TK. Thus, by decreasing the number of points by the factor of nine, we are able to achieve a runtime comparable to TK for the synthetic flag sequence.

Figure 3: Results on the face sequence [17]: (a) exemplary sequence frames with the reference frame highlighted by yellow colour; (b) corresponding reconstructions, two different perspectives respectively.

In the next experiment on synthetic data, we perform measurement of mean RMS error for the dense case. The mean RMS error is defined as $e_{3D} = \frac{1}{F}\sum_{f=1}^{F}\frac{\|\mathbf{s}_f^{ref}-\mathbf{s}_f\|_{\mathcal{F}}}{\|\mathbf{s}_f^{ref}\|_{\mathcal{F}}}$, where $\mathbf{S}_f^{ref}$ are ground truth 3D shapes. We take the 3D motion capture data [36] and create measurement matrix by projecting it using an orthographic camera (which is identity $\mathbf{I}_{2\times 3}$ — the third coordinate is omitted). Note that in the reference frame, the object is not observed frontally but is rotated by approx. $30°$. This makes reconstruction more challenging. Due to the orthographic camera, ambiguities in the initial rotation may occur, and reconstructions are rigidly pre-aligned to the ground truth using a transformation estimation algorithm (Procrustes analysis). Fig. 1-(d) shows the measured mean RMS as a function of the number of frames (top) and basis shapes (bottom). Starting from 10 frames, the 3D error periodically increases with the number of frames. Since the synthetic flag sequence represents motion with strong deformations, AMP accumulates an error and becomes less accurate locally. This example shows that the 3D error can vary considerably with the number of basis shapes. An optimal parameter K depends on the type of the observed motion and deformations.

Qualitative results on real and rendered image sequences. We show results of dense reconstructions for several real-world image sequences. The experiment shows that AMP is able to generate spatially smooth results for the dense cases if correspondences are estimated accurately and context-aware, e.g., by multi-frame optical flow (MFOF) approaches such as [18, 31]. Some exemplary results of AMP on several image sequences are given in Fig. 2-(a), (b). The shown reconstructions are smooth and detailed. Bending of the music notes sheet is reasonably conveyed through non-rigid deformations. Head movements of the owl are realistically explained by combined rotational effects and non-rigid deformations. Results on the processed sequences are best viewed in the supplementary video. Fig. 2-(c) shows an example of reconstruction of the *shaman2* sequence by VA (top) as well as AMP, given point reprojections as an input. This experiment shows that AMP can reconstruct an overall appearance of the scene, despite a relatively high mean RMS error of 0.42. In many applications, this accuracy can be sufficient. An excerpt from the 120 frames long *face* sequence together with exemplary reconstructions (two different perspectives) are shown in Fig. 3. Every shape contains $\sim 2.8 \cdot 10^4$ points and the complete reconstruction is accomplished within 10 seconds. Note how well the mouth expressions are reflected in the reconstructed point clouds.

The experiment on the *heart* bypass surgery sequence demonstrates the performance of AMP in the medical context. This sequence contains 80 frames and has several distinctive attributes. First, self-occlusion effects are almost not presented (except, perhaps, the areas of drifting specular effects). Second, point displacements from every frame to every other frame are comparably small due to the periodicity of heart beating and the fixed camera. These circumstances create conditions for application of a two-frame optical flow algorithm to compute correspondences from a key frame to every other frame in the sequence, instead of the computationally expensive MFOF. Thus, we compute correspondences using the real-time capable TV-L^1 optical flow approach of Zach *et al.* [37] in 21 seconds (computation is sequential). Based on the established dense point correspondences, AMP computes realistic reconstructions with $68 \cdot 10^4$ points each in 11.5 seconds. Only several frames exhibit insignificant surface fluctuations due to the highest point displacements (e.g., see Fig. 4, frame 37). This experiment demonstrates that an accurate dense non-rigid reconstruction, 80 frames in length, can be obtained in combination with AMP within a half minute after the acquisition; adaptation for a real-time window-based operation is conceivable.

Discussion. Due to the runtimes achieved in the experiments, AMP is suitable for building interactive and real-time applications (e.g., robotic or medical ones) on CPU only, possibly in combination with other methods. We believe that AMP, being suitable for mobile platforms, can at the same time provide initialisation for other, more accurate NRSfM methods. We also observe that AMP scales well with the number of points. Either 10^3 or 10^6 points, AMP is running reliably; we have not observed any side effects as the number of points increases unless there is not enough memory in the system.

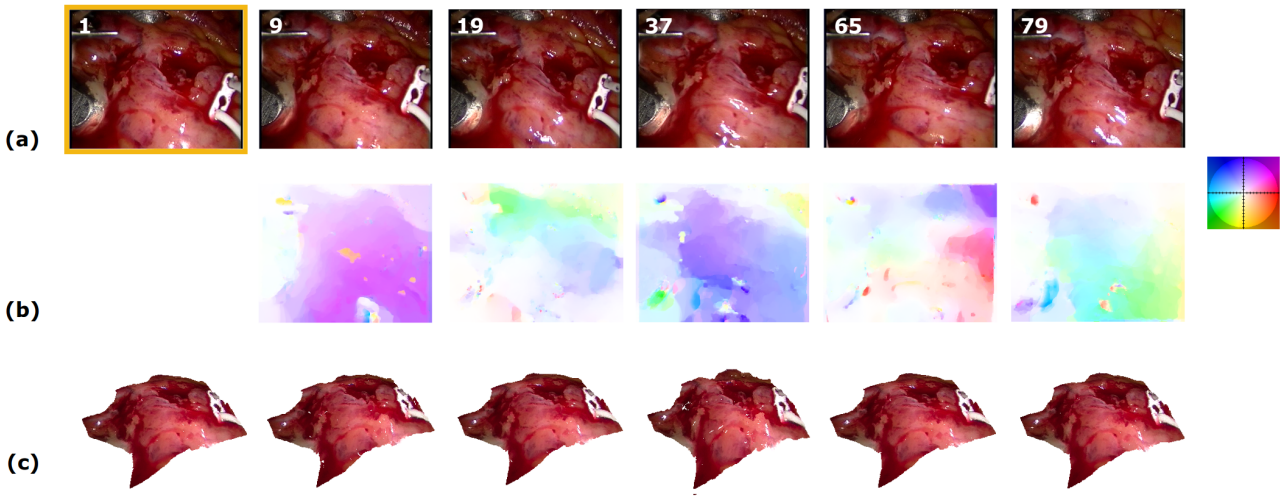

Figure 4: Results on the heart sequence [29]: (a) the input image sequence with the reference frame highlighted by yellow colour; (b) optical flow computed using TV-L^1 approach of Zach *et al*. [37], together with the colour key [6] on the right; (c) corresponding reconstructions recovered by AMP. Correspondence computation and non-rigid surface reconstruction with AMP are obtained within 33.5 seconds.

6. Conclusion

The methodology proposed in this paper enables convex relaxation problems formulated on a high level of abstraction to be solved with an efficient SDP solver such as CSDP. The feasibility of the proposed methodology is demonstrated on the example of MP algorithm, for which an efficient implementation or an implementation on a mobile platform was not previously possible. The proposed algorithm, i.e., AMP allows obtaining accurate dense reconstructions with tens of thousands of points on a mobile platform in seconds; it outperforms the more accurate VA (own optimised heterogeneous implementation) in regard to the runtime by at least the factor of 10. We discovered that AMP scales well with the number of points, the number of frames as well as the number of basis shapes. Combined with suitable methods for correspondence computation, AMP can produce spatially and temporally accurate and realistic results in challenging real scenarios. These properties suggest that AMP can be advantageously placed in the context of modern real-time and interactive applications in such areas as robotics, medicine, and many others. Readers interested in real-time and interactive applications of NRSfM may wish to further investigate the temporal smoothness constraint, as AMP reconstructions are currently independent of the frame order.

Acknowledgements

The research was supported by the projects DENSITY (01IW12001) and DYNAMICS (01IW15003) of the German Federal Ministry of Education and Research (BMBF). The authors thank Bertram Taetz for the fruitful discussions about the core MP method, Brian Borchers for helping to broaden our understanding of the CSDP library, and Marco Paladini for a review and the valuable feedback.

A. Appendix – Proof of Prop. 1

To show validness of Prop. 1, we use one of the necessary and sufficient conditions for positive semidefiniteness: a symmetric matrix U is PSD if $x^\mathsf{T} U x \succeq 0$ for all vectors x (this is also definition of PSD).

Consider symmetric matrix $\hat{U} = \begin{pmatrix} U_1 & 0 \\ 0 & 0 \end{pmatrix}$. Since U_1 is PSD, $y^\mathsf{T} U_1 y \succeq 0$ holds for all vectors y. Moreover, $\hat{y}^\mathsf{T} \hat{U} \hat{y} \succeq 0$, where $\hat{y}$ is an arbitrary vector. Hence, $\hat{U}$ is PSD.

Analogously, consider symmetric matrix $\tilde{U} = \begin{pmatrix} 0 & 0 \\ 0 & U_2 \end{pmatrix}$. Since U_2 is PSD, $z^\mathsf{T} U_2 z \succeq 0$ holds for all arbitrary vectors z. Furthermore, $\tilde{z}^\mathsf{T} \tilde{U} \tilde{z} \succeq 0$, where $\tilde{z}$ is an arbitrary vector. Hence, $\tilde{U}$ is PSD. Now consider the sum $\hat{U} + \tilde{U}$:

$$x^\mathsf{T} \hat{U} x + x^\mathsf{T} \tilde{U} x \succeq 0, \text{ or} \tag{26}$$

$$x^\mathsf{T} (\hat{U} + \tilde{U}) x \succeq 0. \tag{27}$$

Thus, Eq. (27) implies that $U = \hat{U} + \tilde{U}$ is PSD.

To show positive semidefiniteness of U_1, we choose a vector $\tilde{x}$ with non-zero entries at the first $k = \dim(U_1)$ positions. Since U is PSD, $\tilde{x}^\mathsf{T} U \tilde{x} \succeq 0$. Hence, $x_{1\times k}^\mathsf{T} \, U_1^{k\times k} \, x_{k\times 1} \succeq 0$ for all vectors $x_{k\times 1}$, i.e., U_1 is PSD. Analogously, we choose a vector $\tilde{x}$ with non-zero entries at the last $l = \dim(U_2)$ positions, and $x_{1\times l}^\mathsf{T} \, {U_2}^{l\times l} \, x_{l\times 1} \succeq 0$ for all vectors $x_{l\times 1}$, i.e., U_2 is PSD. □

References

[1] M. Agrawal and L. S. Davis. Camera calibration using spheres: a semi-definite programming approach. In *International Conference on Computer Vision (ICCV)*, pages 782–789 vol.2, 2003. 2, 5

[2] A. Agudo, L. Agapito, B. Calvo, and J. Montiel. Good vibrations: A modal analysis approach for sequential non-rigid structure from motion. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1558–1565, 2014. 2

[3] A. Agudo, J. M. M. Montiel, L. Agapito, and B. Calvo. Online dense non-rigid 3d shape and camera motion recovery. In *British Machine Vision Conference (BMVC)*, 2014. 2, 3

[4] A. Agudo and F. Moreno-Noguer. Learning shape, motion and elastic models in force space. In *International Conference on Computer Vision (ICCV)*, 2015. 2, 5

[5] E. Anderson, Z. Bai, C. Bischof, L. S. Blackford, J. Demmel, J. J. Dongarra, J. Du Croz, S. Hammarling, A. Greenbaum, A. McKenney, and D. Sorensen. *LAPACK Users' Guide (Third Ed.)*. Society for Industrial and Applied Mathematics, 1999. 5

[6] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision (IJCV)*, 92(1), 2011. 8

[7] H. Barnum, M. Saks, and M. Szegedy. Quantum query complexity and semi-definite programming. In *Conference on Computational Complexity (CCC)*, pages 179–193, 2003. 2

[8] B. Borchers. CSDP 2.3 user's guide. *Optimization Methods and Software*, 11(1):597–611, 1999. 5

[9] B. Borchers. CSDP, a C library for semidefinite programming. *Optimization Methods and Software*, 11(1):613–623, 1999. 2, 5

[10] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3d shape from image streams. In *Computer Vision and Pattern Recognition (CVPR)*, pages 690–696, 2000. 2, 3

[11] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision (ECCV)*, pages 611–625, 2012. 5, 6

[12] A. Chhatkuli, D. Pizarro, T. Collins, and A. Bartoli. Inextensible non-rigid shape-from-motion by second-order cone programming. In *Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[13] Y. Dai, H. Li, and M. He. A simple prior-free method for non-rigid structure-from-motion factorization. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2018–2025, 2012. 2, 6

[14] P. Dinning. *Barn Owl at Screech Owl Sanctuary*. https://www.youtube.com/watch?v=xmou8t-DHh0, 2014. [online; accessed 12.05.2016; usage rights obtained]. 6

[15] M. Dodig, M. Stoi, and J. Xavier. On minimizing a quadratic function on stiefel manifold. *Linear Algebra and Its Applications*, 475:251–264, 2015. 3

[16] A. Edelman, T. A. Arias, and S. T. Smith. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications*, 20(2):303–353, 1999. 3

[17] R. Garg, A. Roussos, and L. Agapito. Dense variational reconstruction of non-rigid surfaces from monocular video. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1272–1279, 2013. 2, 5, 6, 7

[18] R. Garg, A. Roussos, and L. Agapito. A variational approach to video registration with subspace constraints. *International Journal of Computer Vision (IJCV)*, 104(3):286–314, 2013. 3, 5, 6, 7

[19] V. Golyanik, A. S. Mathur, and D. Stricker. Nrsfm-flow: Recovering non-rigid scene flow from monocular image sequences. In *British Machine Vision Conference (BMVC)*, 2016. 5

[20] G. Guennebaud, B. Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010. 5

[21] C. Helmberg, F. Rendl, R. J. Vanderbei, and H. Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2):342–361, 1996. 1, 2, 5

[22] J. Keuchel. Multiclass image labeling with semidefinite programming. In *European Conference on Computer Vision (ECCV)*, pages 454–467, 2006. 2

[23] J. Keuchel, C. Schnörr, C. Schellewald, and D. Cremers. Binary partitioning, perceptual grouping, and restoration with semidefinite programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 25(11):1364–1379, 2003. 2

[24] Z. Li, J. Liu, and X. Tang. Constrained clustering via spectral regularization. In *Computer Vision and Pattern Recognition (CVPR)*, pages 421–428, 2009. 2, 5

[25] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Computer Vision and Pattern Recognition (CVPR)*, 2016. 5, 6

[26] M. Paladini, A. Del Bue, J. Xavier, L. Agapito, M. Stošić, and M. Dodig. Optimal metric projections for deformable and articulated structure-from-motion. *International Journal of Computer Vision (IJCV)*, 96(2):252–276, 2012. 1, 2, 3, 5

[27] P. A. Parrilo and S. Lall. Semidefinite programming relaxations and algebraic optimization in control. *European Journal of Control*, 9:2–3, 2003. 2

[28] C. Russell, J. Fayad, and L. Agapito. Dense non-rigid structure from motion. In *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pages 509–516, 2012. 2, 3

[29] D. Stoyanov. Stereoscopic scene flow for robotic assisted minimally invasive surgery. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 479–486, 2012. 6, 8

[30] J. F. Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1-4):625–653, 1999. 2

[31] B. Taetz, G. Bleser, V. Golyanik, and D. Stricker. Occlusion-aware video registration for highly non-rigid objects. In *Winter Conference on Applications of Computer Vision (WACV)*, 2016. 7

[32] J. Taylor, A. Jepson, and K. Kutulakos. Non-rigid structure from locally-rigid motion. In *Computer Vision and Pattern Recognition (CVPR)*, 2010. 2

[33] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision (IJCV)*, 9:137–154, 1992. 5, 6

[34] S. Vicente and L. Agapito. Soft inextensibility constraints for template-free non-rigid reconstruction. In *European Conference on Computer Vision (ECCV)*, pages 426–440, 2012. 3

[35] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision (IJCV)*, 70(1):77–90, 2006. 5

[36] R. White, K. Crane, and D. Forsyth. Capturing and animating occluded cloth. In *ACM Transactions on Graphics (SIGGRAPH)*, 2007. 7

[37] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime tv-l1 optical flow. In *Annual Symposium of the German Association for Pattern Recognition (DAGM)*, pages 214–223, 2007. 7, 8