# Multi-body Depth-Map Fusion with Non-intersection Constraints

Bastien Jacquet[1], Christian Häne[1], Roland Angst[2], and Marc Pollefeys[1]

[1] ETH Zürich, Switzerland
[2] Stanford, USA

**Abstract.** Depthmap fusion is the problem of computing dense 3D reconstructions from a set of depthmaps. Whereas this problem has received a lot of attention for purely rigid scenes, there is remarkably little prior work for dense reconstructions of scenes consisting of several moving rigid bodies or parts. This paper therefore explores this multi-body depthmap fusion problem. A first observation in the multi-body setting is that when treated naively, ghosting artifacts will emerge, ie. the same part will be reconstructed multiple times at different positions. We therefore introduce non-intersection constraints which resolve these issues: at any point in time, a point in space can only be occupied by at most one part. Interestingly enough, these constraints can be expressed as linear inequalities and as such define a convex set. We therefore propose to phrase the multi-body depthmap fusion problem in a convex voxel labeling framework. Experimental evaluation shows that our approach succeeds in computing artifact-free dense reconstructions of the individual parts with a minimal overhead due to the non-intersection constraints.

**Keywords:** Multi-view stereo, multi-body structure-from-motion, depthmap fusion, convex optimization, Dynamic Scene Dense 3D Reconstruction.

## 1 Introduction

While there exists a large body of work for *rigid* multi-view stereo and depth-map fusion methods, there is remarkably little prior work for dense, entirely image-based 3D reconstructions of multi-body scenes where multiple rigid parts move with different transformations between two frames. However, this is a highly relevant setting since many man-made scenes or objects actually contain multiple moving rigid parts, for example pieces of furniture, adjustable screens, doors, etc. A dense 3D reconstruction together with a dense segmentation of the scene into functional parts is interesting for applications where a user can interact with the reconstructed objects, eg. by opening a door or by pulling out a drawer. This paper therefore addresses this multi-body depth-map fusion problem, and as a byproduct also provides a dense segmentation into differently moving parts. We note that sparse multi-body structure-from-motion (SfM) has received some attention in previous work, see eg. [1]. Since our paper clearly focuses on the

D. Fleet et al. (Eds.): ECCV 2014, Part VI, LNCS 8694, pp. 735–750, 2014.

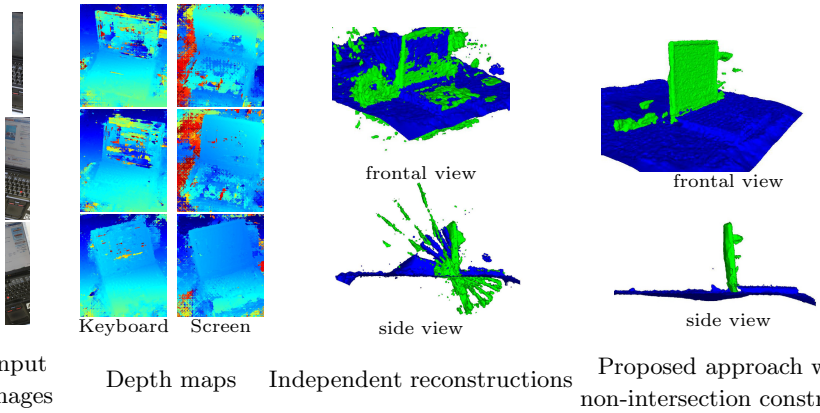


**Fig. 1.** In multi-body depth-map fusion, we are trying to compute a dense reconstruction from a set of input depth-maps of a scene containing multiple moving rigid objects or parts (in the visualized example, the screen rotates w.r.t. the keyboard). A straight-forward approach which treats the reconstruction of each rigid part independently leads to severe artifacts, like a part being reconstructed multiple times at slightly different poses (especially visible in the side view where features of the desk, keyboard, and screen get reconstructed multiple times). The approach proposed in this paper introduces non-intersection constraints which enable complex interactions between the individual reconstruction problems thereby eliminating almost all those artifacts.

dense reconstruction, which requires known camera poses, we assume that those camera poses are provided by such a sparse multi-body SfM component. For the sake of completeness, we will nonetheless highlight several particular issues with the sparse multi-body setting in Sec. 4. Our multi-body depth-map fusion algorithm only requires depth-maps and camera poses registered to a common references frame. Recent depth-cameras could therefore be used as well, even though in this paper we only consider depth-maps computed from RGB images. As described in detail in Sec. 5, we are building on top of a convex voxel labeling framework for dense rigid reconstructions. In a nutshell, those frameworks label a voxel as either 'occupied' or 'freespace' based on a spatial regularization term together with a data fidelity term derived from depth-maps. Those methods provide accurate results for rigid scenes. As multi-body or articulated scenes are assembled from multiple rigid parts, an intuitive idea is to instantiate a separate voxel grid for each part and solve multiple labeling problems independently.

However, this leads to severe ghosting artifacts in the reconstructions: described in more detail in Sec. 3, the same part will be reconstructed multiple times, maybe even in a voxel grid associated to another part (see Fig. 1). In order to prevent this from happening, the grids must be allowed to interact with each other. Indeed, if we consider whether the reconstructions of all the parts are intersection free, we see that ghosting artifacts actually lead to intersections with other parts. Hence, our physically-motivated method is based on a remarkably simple, yet powerful idea: a moving rigid part can not occupy a point in space which is already occupied by another rigid part. These

*non-intersection constraints* can be formulated as linear inequality constraints which link the previously independent labeling problems together. This results in complex interactions of regularization and data terms between different grids and will resolve the ghosting artifacts. It is important to note that despite rather weak data terms due to noisy depth maps, the non-intersection constraints result in a strong mutual exclusion principle which provides a clear part-based 3D segmentation. Interestingly, those non-intersection constraint also enable to 'carve out' regions which have never been directly observed in the input images. For example, observing how a drawer moves in and out of the drawer casing together with the non-intersection constraints immediately lead to the conclusion that the drawer casing must be hollow, a fact which is reproduced by our algorithm.

In contrast to previous convex depth-map fusion formulations for rigid scenes, our formulation contains two entirely different types of constraints: the non-intersection constraints and the standard variational formulation of the total variation regularizer. In order to efficiently handle this large optimization problem, we propose to use a preconditioned primal-dual proximal method [2]. Moreover, the set of non-intersection constraints can be quite large. Fortunately, only a small fraction of those constraints will be active at the globally optimal solution, lending itself to an efficient constraint generation procedure. An experimental evaluation shows that our algorithm achieves its goals of a dense multi-body reconstruction. In summary, the contributions of this paper are:

i) Introduction of the problem of dense consistent multi-body depth-map fusion and presentation of a solution based on convex non-intersection constraints which not only prevents ghosting artifacts but also carves out voxels in unseen regions which are occupied by another part.
ii) Description of a convex voxel labeling algorithm which is solved with a preconditioned primal-dual proximal method and with an efficient constraint generation procedure.

## 2 Related Work

Image based 3D reconstruction methods have made large progress in recent years. Structure-from-motion (SfM) [3] is nowadays a well-established technique to accurately compute camera poses and sparse 3D point clouds. While originally assuming an entirely rigid scene, sparse 3D reconstruction methods have been extended to deal with multiple rigid bodies [1] or with articulated objects [4,5], and even to more general non-rigidly deforming objects [6,7]. Each of those classes of objects or scenes comes with its own challenges. For example, finding a globally consistent scale for all the objects in a multi-body rigid scene is a non-trivial task, for which no solution exists in the most general case of entirely independently moving rigid objects [8]. Notably, parts of an articulated rigid objects are not allowed to move freely, a fact which manifests itself for example in a rank-constraint for a feature trajectory matrix which can be used to infer joint locations and a global scale [9]. In order to constrain the solution space,

reconstruction methods for deformable objects require a prior for regularization, such as local rigidity [10], physically-inspired models [11], or a low-dimensional subspace model [12]. Even though non-rigidly deformable objects certainly represent an important class of objects, in this paper we focus on reconstructing multi-body rigid and articulated scenes, mainly because these scenes can be broken down into multiple moving rigid parts, whose interaction with each other can be captured in a mathematically exact way, as will be described later.

For multi-body rigid scenes, recent efforts in multi-body structure-from-motion led to the simultaneous computation of motion, segmentation and depth-maps from images [13,14]. These approaches estimate the different incremental motions from video with an iterative alternating scheme, and output a single [14] or a sequence [13] of depth-maps, with a segmentation mask and the associated motion for each object. One could feed those into a separate per-object depth-map fusion. However, deferring a reasoning in 3D can lead to ghosting artifacts where parts intersect or get reconstructed multiple times. This happens when a part remains photoconsistent w.r.t. another part, eg. for object shadows on the ground plane, as can be seen in [13,14]'s results. Our goal differs from theirs as we aim at volumetric, non-intersecting and time-consistent 3D models from unordered depth-maps. [13,14] can therefore be used to generate input depth-maps for our method.

In the Computer Graphics community, Chang and Zwicker's work [15] also addresses the problem of registering data from an articulated object in different configurations, mainly in order to infer the articulation chain and the assignment of points to parts. However, their focus differs from ours: while we are interested in a dense volumetric reconstruction from image data, their approach requires high-quality outlier-free point clouds as input so that no optimization over the 3D locations of the points is required. Unfortunately, image based methods require a SfM stage and point clouds acquired by SfM techniques tend to be very noisy and contaminated with many outliers, the same being true for RGBD-sensors. Those outliers are not handled by their method. Moreover, non-intersection constraints are not considered at all in [15].

Once camera poses are known from a sparse reconstruction method, dense reconstruction algorithms can be used to compute and extract 3D surfaces. Also known as multi-view stereo, this field contains a large amount of previous work. We refer the reader to [16] and its excellent accompanying website for further related work in this area. Carving out regions in 3D which project to non-occupied areas in the images have lead to the well-understood concept of the visual-hull [17]. Phrasing the multi-view stereo problem as a binary labeling problem of a 3D voxel grid with labels 'occupied' or 'freespace' and adding regularization, [18] has used graph-cuts to optimize the resulting energy minimization problem. Shortly after that, this discrete graph-cut formulation has been rephrased in the continuous setting as a convex optimization problem [19], where an efficient, highly parallelizable algorithm has been introduced. More generally, convex relaxations of binary or multi-label problems can be used to combine data evidence from images with a regularization term, usually a total-variation term, in an efficient,

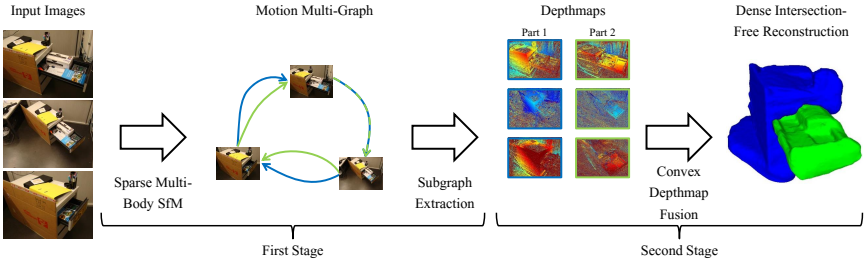


**Fig. 2.** The overall dense reconstruction pipeline takes an unordered set of images as input. As a preliminary stage, a sparse multi-body SfM component extracts a motion multigraph: a node represents an image and each edge represents a relative rigid transformation (of the camera or a rigid part) between two images. After selecting the appropriate subgraph for each part, depthmaps can be computed for that part. In the main stage, those depthmaps are fused in a volumetric convex voxel labeling framework where each part is reconstructed in a separate voxel grid. However, non-intersection constraints between the parts enable a complex interaction between all those grids, thereby carving out voxels in unseen areas which are occupied by another part and avoiding reconstruction artifacts such as 'ghost parts'.

highly parallelizable framework [20]. Our multi-body depth-map fusion approach builds on top of such convex relaxations. Recent approaches with RGBD cameras [21] only consider the unary data term and ignore any pairwise or higher-order regularization term, thereby sacrificing accuracy in favour of speed. Such a fusion approach has recently been extended for articulated model-based tracking [22]. Analogously to our approach, the model is represented by instantiating a separate voxel grid for each rigid part. An iterative-closest point algorithm between a new RGBD frame and the model is used to update the camera position, and given this camera position the new data evidence is fused into the grids. Hence in direct comparison to our approach, [22] is much faster but also less accurate, ignores the physical non-intersection constraints, and requires a RGB-D camera together with a continuous stream of images for incremental computation.

## 3 Problem Description and System Overview

The input to our algorithm consists of a set of depth-maps taken from a scene with multiple moving rigid objects or parts. Furthermore, the camera is also allowed to move. The number of parts $G$ is assumed to be known.

Our reconstruction pipeline contains a preliminary depth-map generation stage, and a main fusion stage, see Fig. 2. The depth-map fusion stage is based on a volumetric fusion of depth-maps in a voxel grid. In contrast to previous approaches which consider a single rigid object and thus only a single voxel grid, we propose to use a separate voxel grid for each part in order to tackle the dense multi-body depth-map fusion problem. This second stage will be described in detail in Sec. 5. As we will see, in order to define a data term for a certain image and part, the depth-maps must be registered to those voxel grids, each of

which is rigidly attached to a part. This registration step obviously requires the camera position of this depth-map with respect to that part. The preliminary stage (outlined in Sec. 4) therefore performs a multi-body sparse reconstruction in order to compute all the camera poses and the motions of the parts. Once all the camera poses are known, depth-maps can be generated to be used in the main fusion stage.

At this point it is important to highlight a problem particular to the multi-body setting. Let us consider a simple setup with two parts and look at sets of depth-maps generated in two different cases. In the first case, the two parts move with respect to each other between any two images. The resulting depth-maps for each part will contain highly confident measurements in regions where that part is visible and contain random measurements with low confidence in regions of the other part because the motions of the two parts are different and thus inconsistent for computing a single depth-map. In the second case however, only the camera moves and the two parts stay rigid with respect to each other. The depth-maps for those images will be the same for both parts and contain highly confident measurements for *both* parts. Fusing all those depth-maps from the first and second case for each part independently will lead to *ghost parts*: the highly confident measurements of the second case provide consistent evidence for both parts in each one of the two voxel grids, whereas the inconsistent regions with low confidence in depth-maps of the first case would be simply treated as outlying measurements for example due to occlusions. In summary, the two parts would get reconstructed in both voxel grids leading to ghost parts which would occupy the same points in space once a voxel in one grid is mapped to the other grid.

Therefore, our proposed fusion process allows the grids to interact with each other in a complex way so that parts are guaranteed to not intersect with each other at any point in time. This does not only avoid ghost parts, it also allows to carve out regions in space which have never been observed directly in any of the images.

## 4 Sparse Multi-body Structure-from-Motion

As mentioned in the introduction, the main contribution of this paper is the dense depth-map fusion formulation with non-intersection constraints. To compute the required camera poses, iterative approaches similar to those used in [13,14] could be used. We preferred a non-iterative one that we quickly outline in this section explain, for the sake of completeness. This first stage mostly uses existing building blocks for sparse multi-body SfM.

### 4.1 Extracting Relative Transformations

Since each part is allowed to move with respect to other parts and the camera, we initially treat the reconstruction of each part as a separate rigid SfM problem. Sparse feature point correspondences are therefore fed into a sequential

RANSAC which extracts all the essential matrices with sufficient inlier support. While being a fairly simple and straight-forward approach, sequential RANSAC has proven to be sufficient for our purposes. As two-view epipolar geometry might lead to many spurious inliers (eg. due to repetitive structure), a three-view verification is performed to eliminate inaccurately detected essential matrices. An additional benefit of three-view verification is that the scale for relative transformations within a three-view verified cluster of images is fixed (two-view relations only provide the translation up to scale) and a consisten reference coordinate frame can be chosen for this cluster. It is important to note that such a cluster can nonetheless contain relative transformations from multiple different parts: Whenever two parts do not move with respect to each other between two images (as visualized by the dashed arrow in Fig. 2), their three-view verified relative transformations which link to those two images will be assigned to the same cluster thereby fixing a consistent scale between the two parts. If a part is never static with respect to another part, additional heuristics such as a common ground plane or motion constraints can be used [8,23]. For simplicity, we decided to take two pictures with a moving camera where no part is moving. This provides a common reference frame and scale for all the relative transformations. Conceptually, at this point, a multigraph with a node for each image and an oriented edge for each extracted relative rigid transformation captures the geometric relations between the images.

### 4.2 Motion Segmentation

In order to get the rigid motion of part $g$, the correct subset of edges needs to be selected from the multigraph. Note that an edge can be selected multiple times: this happens if two or more parts do not move with respect to each other between two images. This is a special instance of the motion segmentation problem for which a fair amount of previous work exists [1,24,5,8,13,14]. If a feature point on a certain part can be tracked throughout all the images, the motion segmentation becomes trivial for that part. If this is not the case, heuristics based on color similarity, spatial proximity, etc. can be used to 'stitch' partial feature trajectories together and read off the correct segmentation from the result. Since motion segmentation is not the focus of our work, we ask for user input to manually correct erroneously stitched trajectories.

Interestingly, the dense reconstruction framework can handle some potential errors in the motion segmentation. This robustness is mainly due to the following observations. (i) Each subgraph is built from robust, three-view verified relations because the original multigraph is built that way as well. The motion subgraph for part $g$ is further refined by running a rigid bundle adjustment on that subgraph. This further eliminates wrongly included or inaccuratly estimated relative transformations. (ii) Remaining inaccurately registered views in a subgraph will be handled by two components in our framework: Firstly, inaccurate views lead to depthmaps with low confidence scores and hence they will contribute less to the data term. Secondly, the depth map fusion algorithm uses a robust data cost and a regularization term which inhibit outlying measurements to some extent.

### 4.3 Generating Cost Volumes

Once the subgraph and thus relative transformations for each part is known, depthmaps for that part can be generated for each node (ie. camera view) in the subgraph. We are currently using a CUDA-implemented plane-sweep algorithm similar to [25] for the computation of the depth maps. The depth maps of a specific part $g$ are all registered into a separate cost volume for that part. Specifically, given the positions of the grid $g$ and camera, each depth map is converted into a truncated signed distance field and these values are aggregated in a voxel grid $\mathbf{d}_g$. The contribution to the signed distance field of a depth measurement at a certain pixel in a depthmap is weighted with a pixel confidence score based on the disparity matching cost for that pixel. These voxel grids are then used as data evidence in the dense reconstruction step. The orientation and extents of the voxel grid for a part are given by a well-aligned and tightly fitted bounding box around the sparse point cloud associated to that part. This bounding box is robustly estimated by considering the distributions of the projection of this point cloud onto normal directions of dominant scene planes contained in that point cloud. Note that in this way, the grids are best aligned with respect to each part and are not axis-aligned to each other.

## 5 Dense Multi-body Depth-Map Fusion

### 5.1 Preconditioned Primal-Dual Proximal Method

**Continuous Formulation.** Motivated by widely-used convex formulations for binary segmentation problems [19,20], our formulation is based on a separate 'occupancy indicator function' $x_g : V_g \to [0,1]$ for each grid $g \in \{1,\dots,G\}$. $V_g$ denotes a spatial volume around part $g$ and a value of $x_g(v) = 1$ denotes that this point $v \in V_g$ is occupied by part $g$ whereas $x_g(v) = 0$ denotes unoccupied freespace. The local data evidence $d_g(x_g)$ from the depth maps is combined with a regularization term $r_g(x_g)$ in a continuous energy function. Specifically, we follow a convex fusion framework with the widely-used total-variation regularization $r_g(x_g) = \int_{V_g} \|\nabla x_g(v)\| dv$ and a unary data term $\langle d_g, x_g \rangle = \int_{V_g} d_g(v) x_g(v) dv$ [19]. Hence, the continuous energy functional looks like

$$E(x) = \sum_g r_g(x_g) + \mu \langle d_g, x_g \rangle, \tag{1}$$

where $x = \{x_1,\dots,x_G\}$ denotes the set of occupancy densities and $\mu \in \mathbb{R}$ is a parameter balancing data fidelity with the regularization. For later reference, we note the variational representation of the total-variation $\int_{V_g} \|\nabla x_g(v)\| dv = \max_{p_g : \|p_g(v)\| \le 1} \int_{V_g} \langle p_g(v), \nabla x_g(v) \rangle dv$, which makes use of a dual vector-valued function $p_g : V_g \to \mathbb{R}^3$.

Since each rigid part $g$ is allowed to move in the scene, the volumes $V_g$ actually move rigidly with respect to each other and thus also with respect to a fixed global coordinate system. The non-intersection constraints impose that at any

point in time, no point in space can be occupied by more than one part. Formally, let $T_{t,g_0\to g}$ denote the rigid transformation which maps a point $v \in V_{g_0}$ from grid $g_0$ at time $t$ to the local coordinates of grid $g$. The transformations $T_{t,g_0\to g}$ are computed as described in Sec. 4. Hence, we have access to transformations at points in time $t$ for which we have image observations. The non-intersection constraints then become

$$\forall v \in V_{g_0} : \sum_{g\in\{1,\dots,G\}} x_g(T_{t,g_0\to g}(v)) \le 1. \tag{2}$$

Note that for known $T_{t,g_0\to g}$, the non-intersection constraints are linear inequality constraints with respect to the unknown functions $x_g$. This observation is important since this allows us to include those non-intersection constraints in a convex optimization framework.

**Discretization.** In the discrete setting, the volumes $V_g$ are represented by appropriately sized and aligned voxel grids around each part. The set of indices for voxels in grid $g$ is also denoted by $V_g$, and the number of voxels in grid $g$ equals $n_g = |V_g|$. $\mathbf{x}_{v_g} \in [0,1]$ denotes the occupancy indicator of a voxel $v_g \in V_g$, whereas $\mathbf{x}_g = \left(\Downarrow_{v_g\in V_g} \mathbf{x}_{v_g}\right) \in [0,1]^{n_g}$ is the vertical concatenation of labels for voxel grid $g$. Similarly, the complete vector of occupancy indicator variables is given by $\mathbf{x} = (\Downarrow_{g\in G} \mathbf{x}_g) \in [0,1]^{\sum_g n_g}$. An analogous notation is used for dual variables $\mathbf{p}_v$ and the unary data costs $\mathbf{d}_g$. Using the variational form of the total variation, the discrete version of Eq. (1) is

$$E(\mathbf{x}) = \sum_g \sum_{v_g\in V_g} \max_{\mathbf{p}_{v_g}} \langle \mathbf{p}_{v_g}, \nabla \mathbf{x}_{v_g}\rangle - i(\mathbf{p}_{v_g} \le 1) + \mu \mathbf{d}_{v_g}\mathbf{x}_{v_g} + i(\mathbf{x}_{v_g} \in [0,1]), \tag{3}$$

where $i$ denotes the indicator function (0 if condition is met, $\infty$ if not) and $\nabla \mathbf{x}_{v_g}$ is computed with finite forward differences. As we will see in Sec. 5.2, the non-intersection constraints can be represented as a linear inequality $\mathbf{A}\mathbf{x} \le \mathbf{b}$. After introducing Lagrange multipliers $\mathbf{z}$ in order to handle this linear constraint set, the primal-dual formulation of the resulting optimization problem looks like

$$\min_{\mathbf{x}} \max_{\mathbf{p},\mathbf{z}} \left\langle \begin{pmatrix} \mathbf{z} \\ \mathbf{p} \end{pmatrix}, \mathbf{K}\mathbf{x} \right\rangle - F^*(\mathbf{p},\mathbf{z}) + G(\mathbf{x}), \tag{4}$$

$$\text{where } F^*(\mathbf{p},\mathbf{z}) = i(\mathbf{p} \le 1) + \langle \mathbf{z}, \mathbf{b}\rangle + i(\mathbf{z} \ge \mathbf{0}) \tag{5}$$

$$G(\mathbf{x}) = \mu\langle \mathbf{d}, \mathbf{x}\rangle + i(\mathbf{x} \in [0,1]) \tag{6}$$

$$\mathbf{K} = \left[\mathbf{A}^T, \nabla^T\right]^T. \tag{7}$$

The function $F^*$ (convex conjugate of $F$) acts on the dual variables and $G$ on the primal variables and the linear operator $\mathbf{K}$ connects these two sets of variables. Here, the sparse matrix $\nabla \in \mathbb{R}^{3\sum_g n_g \times \sum_g n_g}$ contains the finit forward difference coefficients for gradients $\nabla \mathbf{x}_{v_g}$.

**Optimization.** This is a large-scale convex optimization problem and we need to carefully select a suitable algorithm. We decided to use a first order primal-dual proximal algorithm [20,26], mainly because the individual update steps in this algorithm decouple and can be parallelized easily. Experimentally, we have confirmed that a preconditioned proximal algorithm [2] converges much faster than its non-preconditioned counterpart. This is due to the fact that our linear operator $\mathbf{K}$ captures two different type of constraints, namely the non-intersection constraints and the constraints due to the variational form of the total variation. Not only contain the matrices $\mathbf{A}$ and $\nabla$ a different number of rows, these rows also contain an entirely different pattern of non-zero entries with values on a different order of magnitude. Using a single step-length for all the dual variables as done in a non-preconditioned proximal method can therefore lead to slow convergence. However, as Chambolle and Pock have shown in [2], a preconditioned primal-dual proximal algorithm with a diagonal preconditioner can choose an adaptive step length per variable. For the sake of completeness and since the derivations need some attention, the required prox-steps of $F^*$ and $G$ for the preconditioned version will be derived in the following. The preconditioned prox-step replaces the standard L2-norm $\|.\|_2$ with a Mahalanobis distance $\|.\|_{\mathbf{\Sigma}}$, in our case $\mathbf{\Sigma}$ is a diagonal positive-definite matrix. Even though $F^*$ decouples into separate prox-steps for $\mathbf{p}$ and $\mathbf{z}$, the prox-step for the latter involves a sum between a linear term $\langle \mathbf{z}, \mathbf{b} \rangle$ and the indicator function $i(\mathbf{z} \geq \mathbf{0})$ leading to the following derivation

$$\begin{aligned}
\operatorname{prox}_{\langle \cdot, \mathbf{b} \rangle + i(\cdot \geq \mathbf{0}), \mathbf{\Sigma}}(\tilde{\mathbf{z}}) &= \arg\min_{\mathbf{z}} \left( \langle \mathbf{z}, \mathbf{b} \rangle + i(\mathbf{z} \geq \mathbf{0}) + \|\mathbf{z} - \tilde{\mathbf{z}}\|_{\mathbf{\Sigma}}^2 \right) \\
&= \mathbf{\Sigma}^{\frac{1}{2}} \arg\min_{\mathbf{z}' = \mathbf{\Sigma}^{-\frac{1}{2}}\mathbf{z}} \left( \langle \mathbf{\Sigma}^{\frac{1}{2}} \mathbf{z}', \mathbf{b} \rangle + i(\mathbf{\Sigma}^{\frac{1}{2}} \mathbf{z}' \geq \mathbf{0}) + \left\| \mathbf{z}' - \mathbf{\Sigma}^{-\frac{1}{2}} \tilde{\mathbf{z}} \right\|_{\mathbf{I}}^2 \right) \\
&= \mathbf{\Sigma}^{\frac{1}{2}} \arg\min_{\mathbf{z}'} \left( \langle \mathbf{z}', \mathbf{\Sigma}^{T\frac{1}{2}} \mathbf{b} \rangle + i(\mathbf{z}' \geq \mathbf{0}) + \left\| \mathbf{z}' - \mathbf{\Sigma}^{-\frac{1}{2}} \tilde{\mathbf{z}} \right\|_{\mathbf{I}}^2 \right) \\
&= \mathbf{\Sigma}^{\frac{1}{2}} \operatorname{prox}_{i(\cdot \geq \mathbf{0}), \mathbf{I}}(\mathbf{\Sigma}^{-\frac{1}{2}} \tilde{\mathbf{z}} - \mathbf{\Sigma}^{T\frac{1}{2}} \mathbf{b}) = \begin{cases} \tilde{\mathbf{z}} - \mathbf{\Sigma}\mathbf{b} & \text{if } \tilde{\mathbf{z}} - \mathbf{\Sigma}\mathbf{b} > 0 \\ 0 & \text{if } \tilde{\mathbf{z}} - \mathbf{\Sigma}\mathbf{b} \leq 0 \end{cases}.
\end{aligned} \tag{8}$$

In the second last step, we have used the property that a non-preconditioned prox-step involving a linear term $\langle \mathbf{z}', \mathbf{\Sigma}^{T\frac{1}{2}} \mathbf{b} \rangle$ is equivalent to applying the prox-step to the difference $\mathbf{\Sigma}^{-\frac{1}{2}} \tilde{\mathbf{z}} - \mathbf{\Sigma}^{T\frac{1}{2}} \mathbf{b}$ (see also Table 1 iv in [26]). Since $\mathbf{\Sigma}$ is chosen diagonal, the prox-steps for $\mathbf{z}$ still decouple. The derivation for the prox-step of the primal variable follows exactly the same steps and gives

$$\operatorname{prox}_{\mu \langle \mathbf{d}, \cdot \rangle + i(\cdot \in [0,1]), \mathbf{T}}(\tilde{\mathbf{x}}) = \max(0, \min(1, \tilde{\mathbf{x}} + \mu \mathbf{T}\mathbf{f})). \tag{9}$$

A similar derivation for the dual variable $\mathbf{p}_{v_g}$ with $\mathbf{\Sigma}_{\mathbf{p}_{v_g}} = \sigma_{v_g}^2 \mathbf{I}_3$ provides

$$\operatorname{prox}_{i(\|\cdot\| \leq 1), \mathbf{\Sigma}_{v_g}}(\tilde{\mathbf{p}}) = \begin{cases} \tilde{\mathbf{p}} & \text{if } \|\tilde{\mathbf{p}}\|_2 \leq 1 \\ \dfrac{\tilde{\mathbf{p}}}{\|\tilde{\mathbf{p}}\|_2} & \text{if } \|\tilde{\mathbf{p}}\|_2 > 1 \end{cases}, \tag{10}$$

**Algorithm 1:** First-Order Primal-Dual Algorithm

**parameters**: Time step sizes
$\tau_j = \frac{1}{\sum_{i=1}^{m} |\mathbf{K}_{i,j}|^{2-\alpha}}, \mathbf{T} = \mathrm{diag}(\tau_j)$, $\sigma_i = \frac{1}{\sum_{j=1}^{n} |\mathbf{K}_{i,j}|^{\alpha}}, \mathbf{\Sigma} = \mathrm{diag}(\sigma_i)$,
primal-dual step trade-off $\alpha \in [0, 2]$

**output** : Minimizer $\mathbf{x}$ for the multi-body depth-map fusion problem.

**while** *not converged* **do**
  `// Primal Variable Update:`
  $\tilde{\mathbf{x}} = \mathbf{x}^t - \mathbf{T}\mathbf{K}^T \begin{pmatrix} \mathbf{z}^t \\ \mathbf{p}^t \end{pmatrix}$ ;
  `// Primal Variable Prox-Step:`
  $\mathbf{x}^{t+1} = \mathrm{prox}_{\mu\langle \mathbf{d}, \cdot \rangle + i(\cdot \in [0,1]), \mathbf{T}}(\tilde{\mathbf{x}})$ ;
  `// Reflection step:`
  $\hat{\mathbf{x}} = 2\mathbf{x}^{t+1} - \mathbf{x}^t$ ;
  `// Dual Variable Update:`
  $\begin{pmatrix} \tilde{\mathbf{z}} \\ \tilde{\mathbf{p}} \end{pmatrix} = \begin{pmatrix} \mathbf{z}^t \\ \mathbf{p}^t \end{pmatrix} + \mathbf{\Sigma}\mathbf{K}\hat{\mathbf{x}}$ ;
  $\mathbf{z}^{t+1} = \mathrm{prox}_{i(\cdot \geq \mathbf{0}),}(\tilde{\mathbf{z}} - \mathbf{\Sigma}_{\mathbf{z}}\mathbf{b})$ ;
  $\mathbf{p}^{t+1} = \mathrm{prox}_{i(\|\cdot\| \leq 1), \mathbf{\Sigma}_{\mathbf{p}}}(\tilde{\mathbf{p}})$ ;
  $t = t + 1$ ;

ie. the preconditioning does not affect this prox-step. In summary, a first-order primal-dual preconditioned proximal algorithm for our non-intersection constrained multi-body depth-map fusion problem looks like outlined in Alg. 1[1]. Note again that all the prox-steps decompose into elementwise prox-steps which can be solved analytically and in parallel.

### 5.2 Non-intersection Constraints as Linear Inequalities

This section derives the discretized linear inequality constraints which enforce the non-intersection constraints. We recall that $T_{t,g \to g'}$ denotes the rigid transformation from grid (or rigid part) $g$ to $g'$ at frame $t$. Let $w_{t,v_g \cap v_{g'}}$ denote the intersection volume between voxels $v_g$ and $v_{g'}$ at time $t$. This is computed by applying the rigid transformation $T_{t,g \to g'}$ to the voxel cube $v_g \in V_g$ which yields the corresponding translated and rotated cube at time $t$ expressed in coordinates of grid $g'$. Even though there exist efficient and fast algorithms for exact voxel cube intersections [27], we opted for a simple estimation procedure of the intersection volume between two voxels. Inspired by Monte Carlo integration and efficient collision detection procedures from computer graphics, a voxel $v_g$ is subdivided into $n$ regularly spaced sample points. Each sample point is transformed into the other grid and a counter $n_{v_{g'}}$ is increased for the voxel $v_{g'}$ which

[1] The primal-dual trade-off was fixed to $\alpha = 1$ for all our experiments.

contains the transformed point. The final intersection volume is then estimated as $w_{t,v_g \cap v_{g'}} \approx \frac{n_{v_{g'}}}{n}$. In practice, we used $n = 16^3$ sampling points. With this notation in place, the discrete version of the non-intersection constraint in Eq. (2) reads like the following: For each frame $t$ and each voxel $v \in V_{g_0}$ in the grid $g_0$, it must hold that

$$\sum_{g} \sum_{v_g \in V_g} w_{t,v \cap v_g} x_{v_g} \leq 1. \tag{11}$$

Those constraints can be concisely captured in a linear matrix inequality $\mathbf{Ax} \leq \mathbf{1}$, where the entries of the matrix $\mathbf{A}$ are equal to the weights $w_{t,v \cap v_g}$. Obviously, some of those constraints are redundant (due to symmetry reasons) and therefore not required. Notably, most of the voxels are not intersecting, hence, most of the weights $w_{t,v \cap v_g}$ will be zero resulting in a highly sparse matrix. In practice, there are roughly 6.5 non-zero entries per grid in each row of the matrix $\mathbf{A}$. Nevertheless, the overall number of intersection constraints and thus rows in $\mathbf{A}$ can be very large. Fortunately, only a very small fraction of those constraints will be active at the globally optimal solution. Therefore, we follow a constraint generation approach in our implementation: after a certain number of iterations, we search for violated constraints and those constraints which are close to becoming violated and add those as additional rows to the constraint matrix. Note that when adding new rows to matrix $\mathbf{A}$, $\sigma_i$ and $\tau_j$ need to be updated.

The constraint generation procedure can be formulated with a complexity of $O(n_t n_{g_0} G)$ where $n_t$ denotes the number of images where parts have moved w.r.t. each other. The outermost loop iterates over all time instances $t \in \{1, \ldots, n_t\}$. The next loop iterates over all voxels $v \in V_{g_0}$ in a reference grid $g_0$[2]. Again motivated by collision detection procedures in computer graphics, a sequence of increasingly more complex conservative checks are employed in order to determine early on whether constraint $[t, v]$ is not violated and can be skipped. For those checks, potentially intersecting voxels in other grids $g \neq g_0$ are efficiently computed with integer index arithmetic as neighbor of $T_{t,g_0 \to g}(v)$. This constraint generation procedure incurs only a small overhead to the primal-dual method.

## 6 Results

We present an experimental evaluation on the following datasets and we also refer to the supplemental material for further results:

- **Laptop:** This data set contains 18 images of a laptop where the screen is rotating around its fixed axis w.r.t. the keyboard. The screen is not moving w.r.t. the keyboard in all the frames, ie. in some frames only the camera moved and the scene was perceived as being entirely rigid during those

[2] Here, we implicitly assume that grid $g_0$ is sufficiently large so that by iterating over its voxels, all the non-intersection constraints are considered. A better approach would be to use a hierarchical space partitioning scheme to efficiently compute the regions in which voxel grids intersect. However, we leave that for future work.

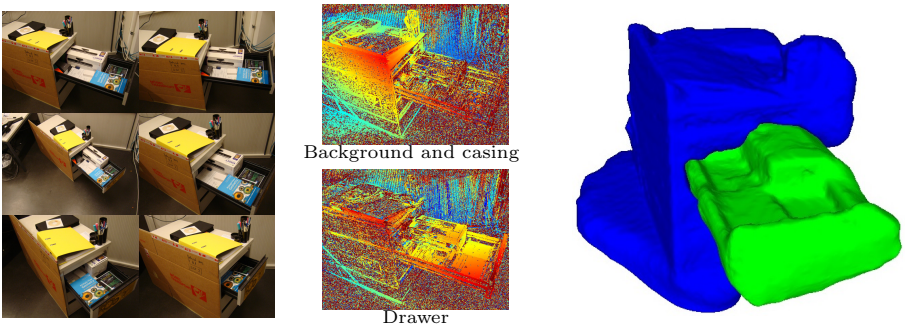


**Fig. 3.** When a drawer moves in and out of the casing, the intersection-constraints automatically carve out a tight hole in the casing in order to avoid any intersections between the drawer and the casing. Moreover, the drawer is reconstructed only once, ie. no ghosting artifacts are present. The visualized depth maps correspond to the input image on the top left. Note that this example only uses 6 input images from one side of the drawer and hence, due to absence of data evidence, the reconstruction on other sides can be inaccurate.

frames. For this example, we have used a best-K ($K = 6$) depthmap matching cost which is robust to occlusions and, more importantly for our setting, also to changes in the rigid configuration of the two parts. This choice results in depthmaps which are fairly consistent for both parts at the same time. Some of those depthmaps are shown in Fig. 1. We can clearly see that the regions of the screen and keyboard in those depthmaps will both provide consistent data evidence in the keyboard and screen voxel grid. Without any non-intersection constraints, the screen should therefore also be reconstructed in the keyboard grid (and the other way around for the keyboard in the screen grid). As the results show, the screen will indeed be reconstructed multiple times, in our case 6 times since we observed 6 different configurations between the keyboard and the screen. However, with activated non-intersection constraints, those ghosting artifacts vanish and the screen and keyboard will be reconstructed exactly once, each in its own grid.

– **Drawer:** The drawer dataset consists of 6 images. 4 different non-rigid configurations have been observed while pushing the drawer into the drawer casing. Hence, in three images, the scene was again perceived as entirely rigid. In contrast to the laptop example where we have used a best-K depthmap matching, all the views contribute to a depthmap estimate this time. Fig. 3 shows an example of the resulting depthmaps for one specific point of view. Note that even though there is a bias towards either the casing or the drawer, there is a fair amount of noise present in those depthmaps. Nevertheless, our algorithm provides an artifact free reconstruction and thus segmentation into the two parts. Moreover, the non-intersection constraints prevent intersections even in unseen or weakly observed areas, resulting in a tight hole in the casing of the drawer such that the drawer can slide in.

The non-intersection constraints obviously increase the time and memory requirements slightly compared to entirely independent reconstructions. It is difficult to exactly quantify the time overhead due to those constraints, since the additional complexity for the constraint generation largely depends on how many occupied and intersecting voxels there are in a per-part independent reconstruction (and hence on the proximity between the parts). In our experiments, the overall time overhead for the primal-dual algorithm was usually within 25 – 50%. We observed a similar overhead for the memory requirements. We currently do not remove constraints which were added to the constraint matrix at some point and got satisfied later on. Such a step could certainly be added if required.

## 7 Conclusion and Future Work

In this paper, we have introduced the multi-body depth-map fusion problem. In order to prevent ghosting artifacts, we have considered the non-intersection constraints which can be formalized as linear inequalities. These inequalities have been integrated in a convex voxel labeling framework, leading to complex interactions between separate voxel grids for each rigid part. As the experimental evaluation shows, this approach eliminates ghosting artifacts. Moreover, regions which have never been observed in the images, like the interior of a drawer case, will be carved out if an intersection with another part would result otherwise. Intuitively, a part can be used to carve out unseen regions in other parts. From an optimization point of view, we verified that a preconditioned primal-dual proximal method converges must faster than its non-preconditioned counterpart. This can be attributed to the fact that our optimization problem contains two entirely different types of dual variables, a setting in which preconditioning is known to be beneficial. Since only a small fraction of the intersection constraints will be active at the globally optimal solution, a constraint generation approach has been introduced to handle the large set of non-intersection constraints without incurring a large overhead to the primal-dual method.

**Acknowledgments.** This work has been supported by the Max Planck Center for Visual Computing and Communication, by the 4DVideo ERC Starting Grant Nr. 210806 and by the Swiss National Science Foundation under Project Nr. 143422.

## References

1. Costeira, J.P., Kanade, T.: A multi-body factorization method for motion analysis. In: ICCV, pp. 1071–1076. IEEE Computer Society, Washington, DC (1995)
2. Pock, T., Chambolle, A.: Diagonal preconditioning for first order primal-dual algorithms in convex optimization. In: ICCV, pp. 1762–1769. IEEE Computer Society, Washington, DC (2011)

3. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision, 2nd edn. Cambridge University Press (2004)
4. Ross, D.A., Tarlow, D., Zemel, R.S.: Learning articulated structure and motion. IJCV 88(2), 214–237 (2010)
5. Katz, D., Brock, O.: Interactive segmentation of articulated objects in 3D. In: Workshop on Mobile Manipulation at ICRA 2011 (2011)
6. Brand, M.: A direct method for 3d factorization of nonrigid motion observed in 2D. In: CVPR, pp. 122–128. IEEE Computer Society, Washington, DC (2005)
7. Torresani, L., Hertzmann, A., Bregler, C.: Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors. In: IEEE TPAMI, pp. 878–892
8. Ozden, K., Schindler, K., Van Gool, L.: Multibody structure-from-motion in practice. In: IEEE TPAMI, pp. 1134–1141
9. Yan, J., Pollefeys, M.: A factorization-based approach for articulated nonrigid shape, motion and kinematic chain recovery from video. In: IEEE TPAMI, pp. 865–877
10. Taylor, J., Jepson, A.D., Kutulakos, K.N.: Non-rigid structure from locally-rigid motion. In: CVPR Computer Society, pp. 2761–2768. IEEE Computer Society, Washington, DC (2010)
11. Agudo, A., Calvo, B., Montiel, J.M.M.: 3D reconstruction of non-rigid surfaces in real-time using wedge elements. In: Fusiello, A., Murino, V., Cucchiara, R. (eds.) ECCV 2012 Ws/Demos, Part I. LNCS, vol. 7583, pp. 113–122. Springer, Heidelberg (2012)
12. Garg, R., Roussos, A., de Agapito, L.: Dense variational reconstruction of non-rigid surfaces from monocular video. In: CVPR, pp. 1272–1279. IEEE Computer Society (2013)
13. Zhang, G., Jia, J., Bao, H.: Simultaneous multi-body stereo and segmentation. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp. 826–833 (November 2011)
14. Roussos, A., Russell, C., Garg, R., Agapito, L.: Dense multibody motion estimation and reconstruction from a handheld camera. In: 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pp. 31–40 (November 2012)
15. Chang, W., Zwicker, M.: Global registration of dynamic range scans for articulated model reconstruction. ACM Transactions on Graphics 30(3), 26:1–26:15 (2011)
16. Seitz, S.M., Curless, B., Diebel, J., Scharstein, D., Szeliski, R.: A comparison and evaluation of multi-view stereo reconstruction algorithms. In: CVPR, pp. 519–528. IEEE Computer Society (2006)
17. Laurentini, A.: The visual hull concept for silhouette-based image understanding. In: IEEE TPAMI, pp. 150–162
18. Lempitsky, V.S., Boykov, Y.: Global optimization for shape fitting. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007, Minneapolis, Minnesota, USA, June 18-23, pp. 1–8. IEEE Computer Society, Washington, DC (2007)
19. Zach, C.: Fast and high quality fusion of depth maps. In: 3DPVT 2008, Atlanta, GA, USA, June 18-20 (2008)
20. Chambolle, A., Pock, T.: A first-order primal-dual algorithm for convex problems with applications to imaging. Journal of Mathematical Imaging and Vision 40(1), 120–145 (2011)
21. Newcombe, R.A., Davison, A.J., Izadi, S., Kohli, P., Hilliges, O., Shotton, J., Molyneaux, D., Hodges, S., Kim, D., Fitzgibbon, A.: Kinectfusion: Real-time dense surface mapping and tracking. In: IEEE Symposium on Mixed and augmented reality (ISMAR 2011), pp. 127–136. IEEE (2011)

22. Malleson, C., Klaudiny, M., Hilton, A., Guillemaut, J.Y.: Single-view rgbd-based reconstruction of dynamic human geometry. In: The IEEE International Conference on Computer Vision (ICCV) Workshops (June 2013)
23. Jacquet, B., Angst, R., Pollefeys, M.: Articulated and restricted motion subspaces and their signatures. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1506–1513. IEEE (2013)
24. Tron, R., Vidal, R.: A benchmark for the comparison of 3-D motion segmentation algorithms. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2007, Minneapolis, Minnesota, USA, June 18-23. IEEE Computer Society, Washington, DC (2007)
25. Yang, R., Pollefeys, M.: Multi-resolution real-time stereo on commodity graphics hardware. In: Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. I-211–I-217 (June 2003)
26. Combettes, P.L., Pesquet, J.C.: Proximal Splitting Methods in Signal Processing. ArXiv e-prints (December 2009)
27. Reveillès, J.P.: The geometry of the intersection of voxel spaces. Electronic Notes in Theoretical Computer Science 46, 285–308 (2001)