Dense Multibody Motion Estimation and Reconstruction from a Handheld Camera

Anastasios Roussos Chris Russell Ravi Garg Lourdes Agapito
School of Electronic Engineering and Computer Science
Queen Mary University of London, UK[*]

Abstract

Existing approaches to camera tracking and reconstruction from a single handheld camera for Augmented Reality (AR) focus on the reconstruction of static scenes. However, most real world scenarios are dynamic and contain multiple independently moving rigid objects. This paper addresses the problem of simultaneous segmentation, motion estimation and dense 3D reconstruction of dynamic scenes. We propose a dense solution to all three elements of this problem: depth estimation, motion label assignment and rigid transformation estimation directly from the raw video by optimizing a single cost function using a hill-climbing approach. We do not require prior knowledge of the number of objects present in the scene – the number of independent motion models and their parameters are automatically estimated. The resulting inference method combines the best techniques in discrete and continuous optimization: a state of the art variational approach is used to estimate the dense depth maps while the motion segmentation is achieved using discrete graph-cut based optimization. For the rigid motion estimation of the independently moving objects we propose a novel tracking approach designed to cope with the small fields of view they induce and agile motion. Our experimental results on real sequences show how accurate segmentations and dense depth maps can be obtained in a completely automated way and used in marker-free AR applications.

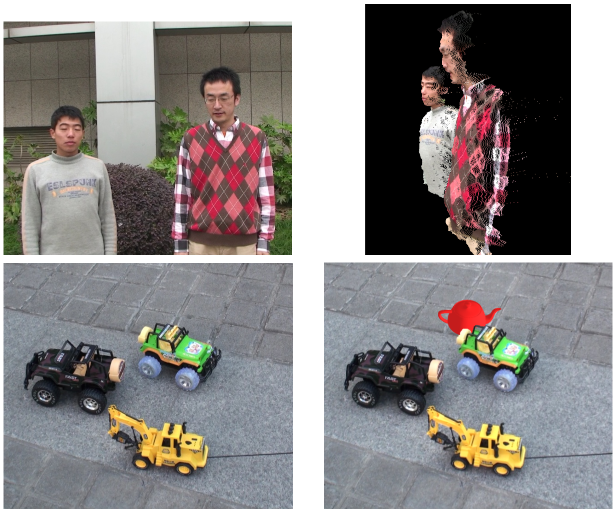


Figure 1: 3D reconstruction and Augmented Reality application on two different input sequences.

1 Introduction

Recent advances in marker-less vision-based tracking methods for Augmented Reality (AR) have resulted in reliable, real-time systems [16, 23] that can provide impressive performance on mobile platforms [37, 24, 22] or even provide live dense reconstructions that allow scene augmentation with dense occlusion reasoning [20]. However, a common drawback of all these approaches is that they require a static scene and treat moving objects as outliers. Attempts to deal with multiple independently moving objects are few and typically require prior knowledge of the number of objects present in the scene and can only reconstruct sparse feature points [21, 18]. To the best of our knowledge, an automated system that can simultaneously provide dense segmentation, motion estimation and 3D reconstruction of independently moving objects in a dynamic scene is currently missing in the literature.

In computer vision, structure from motion (SfM) algorithms for rigid scenes have made significant progress by providing completely dense detailed 3D models that estimate the 3D location of every pixel in the image. Dense approaches to *multi-view stereo* (MVS) [11, 29] aim at acquiring accurate models from a collection of fully calibrated images (where both the camera motion and internal calibration are known in advance). Although computationally expensive they have produced impressive results. More recent solutions are capable of near real-time performance [40] – as the parallel formulation of these algorithms allows the use of GPUs to dramatically speed reconstruction.

In contrast to these MVS batch approaches that treat all the images at once after the acquisition, real-time camera tracking and mapping systems, also referred to as *monocular SLAM* (Simultaneous Localization and Mapping) approaches, have also been developed and applied with great success to small AR workspace applications e.g. PTAM [16]. While PTAM can only reconstruct a sparse cloud of points these *real-time* SfM systems have recently been extended to recover dense 3D structure and motion from live video data coming from a hand-held camera [20, 33] or reconstruct urban environments from camera rigs mounted on a car [25]. One of the clear advantages of reconstructing dense surfaces for AR applications is that more accurate occlusion reasoning can be achieved.

The fundamental assumption underpinning all these algorithms is that the scene is static and the objects making up the scene do not move relative to one another. This is an overly strong assumption – real world scenes are not completely static and generally contain multiple moving objects. The area of multibody structure and motion from video is far less advanced than its rigid counterpart and, with very few multi-camera based exceptions [42, 13], existing approaches act on a sparse set of features. Dynamic scenarios are much more challenging, as they bring in the additional problem of *motion segmentation*: each pixel in the image must be assigned to an independently moving rigid body. An additional practical issue arises when recovering the rigid motion of each of the independent objects in the scene. Tracking small objects in the foreground is more error prone than tracking the camera motion with respect to a large background since: *i)* objects occupy only a small number of image pixels that induce a small aperture angle; *ii)* regions can

[*] e-mails:{troussos,chrisr,rgarg,lourdes}@eecs.qmul.ac.uk

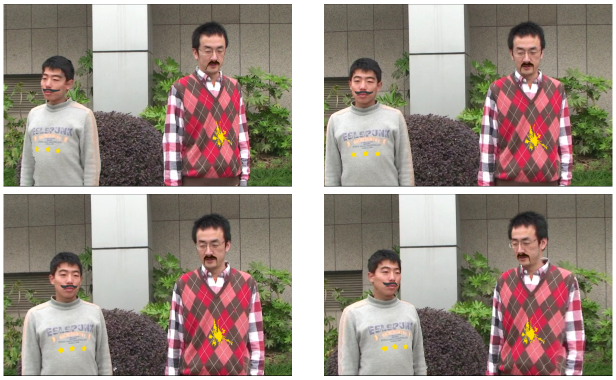


Figure 2: Augmented reality application, using our system of Dense Multibody Reconstruction, using a Handheld Camera.

move more freely, inducing faster, harder to track, motion. It is clear that in such scenarios, a dense pixel-based tracking algorithm will provide more reliable tracking than a feature-based one.

The majority of approaches to dynamic scene analysis work in a three step fashion [6, 10, 28, 21] – first sparse image features are tracked or optical flow maps are estimated; second motion segmentation is performed to separate the tracks into different objects using model selection or a RANSAC type approach; and finally S*f*M is applied to each of the segments to perform 3D reconstruction and motion estimation. However, these pipeline approaches are inherently fragile, as a failure in any of the intermediate steps need not be apparent until the final reconstruction fails, and cannot be recovered from.

Clearly structure and motion estimation and scene segmentation are two sides of the same problem and should be solved simultaneously. The novelty of our approach lies in posing the problems of scene segmentation into multiple rigidly moving objects, the estimation of their 3D shape and motion relative to the camera as the optimization of a single unified cost function. Moreover, our approach is dense, it reconstructs each pixel in 3D, and assigns it a label indicating which rigid body it belongs to. Our approach combines the best techniques of discrete and continuous optimization – a state of the art variational approach closely related to [20] is used to estimate the dense depth maps while the motion segmentation is achieved using discrete graph-cut based α-expansion with an MDL prior [15, 19]. Additionally, we propose a novel algorithm to track the rigid motion of the independent bodies that uses all the pixels for dense image registration. As with the rigid live dense reconstruction systems of [20, 33] we directly estimate the depth field by minimizing a photometric error cost without the need to use either sparse feature tracking or dense optical flow estimation. However, unlike these real-time online systems ours is batch, allowing the use of future data to resolve ambiguities in the current frame.

2 Related Work

Early approaches to multibody segmentation and reconstruction were traditionally pipeline methods in which a sparse set of input feature trajectories were separated into independently moving rigid bodies, and subsequently reconstructed in 3D using multi-view S*f*M approaches. The first of such methods, developed by [6] was based on rigid factorization [34] and applied to the simplified scenario of affine viewing conditions. A wealth of motion segmentation algorithms followed in the literature including the GPCA algebraic framework [36] and more recent approaches that can deal with noise in the measurements, outliers in the correspondences and missing data [26]. Tron and Vidal [35] created a benchmark data set to evaluate these sparse approaches on given sets of point tracks. Ozden et al. [21] proposed an algorithm for simultaneous tracking, segmentation and reconstruction capable of tackling realistic sequences and achieve 3D augmentation. However, their approach was sparse.

Unlike these works, we provide a dense approach to joint segmentation and 3D reconstruction in the multibody case. Perhaps the work closest to ours is the recent solution to simultaneous dense segmentation and reconstruction of dynamic scenes recently proposed by Zhang et al. [42] who optimize an energy based on photometric and geometric error and layer constraints. The method extended multi-view stereo to scenes containing multiple moving objects. Their optimization couples depth and segmentation labels which are fixed and predetermined at the point of the optimization. Crucially, in contrast with our approach, the number of different rigid bodies present in the scene and their motion parameters with respect to the camera must be pre-estimated in an initial step that relied upon sparse data. After this, their labels are fixed and the motion parameters are never updated or re-estimated within their dense energy optimization framework. Guillemaut and Hilton [13] proposed a method for using multiple cameras to jointly segment the image into background/foreground layers and to estimate the depth of each pixel of the foreground in challenging outdoor sports scenes. This approach is intended for multiple synchronized cameras with known relative orientation, and explicitly relies upon multi-view consistency.

Our approach is also related to the work of Fayad et al. [9] who proposed a unified solution to motion segmentation and 3D reconstruction for articulated objects, i.e. objects that do not move independently but share a common joint or axis of rotation. Although their inference method is closer to ours than that of [42], their approach is sparse and takes fixed point correspondences as an input. Instead, our approach optimizes a photometric cost based directly on the image intensities implicitly incorporating the estimation of correspondences into the overall system.

In 2-view stereo reconstruction, the work of [3] proposed an algorithm to reconstruct a scene by decomposing it into a set of b-splines. This approach shares some similarities with our work in that it alternates between a continuous optimization of fitting splines to points in their case, or assigning the depth to points in our case, and a discrete graph-cut based step that assigns points to b-splines or camera parameters. However, unlike our approach, theirs can only be applied to two frames of a static scene, where the camera motion is known in advance.

2.1 Contributions

The main contribution of this work is to provide the first algorithm in the literature to estimate simultaneously all three elements involved in the multibody structure and motion problem: depth map, motion label assignment and rigid transformations for every object in every frame, in a **dense** way i.e. for every pixel in the scene by optimizing a single energy.

Additionally, we propose an inference strategy that combines state of the art methods in continuous and discrete optimization to optimize a single energy. Our optimization alternates between a variational approach [20] to depth map estimation directly from photometric information; a discrete energy based multiple model fitting technique [15] to segment the scene into moving regions; and a novel dense tracking algorithm.

At the core of our optimization method is the energy based multiple model fitting approach to segmentation PEARL [15] that reformulates it as a labeling problem where both the labels (model parameters) and their assignment to data points are computed simultaneously using graph-cut based discrete optimization initialized with an excess of models.

Our novel dense tracking method estimates a 6-DOF transformation for each independently moving object with respect to the camera given a dense depth map. Our algorithm is most closely related to the 6-DOF camera tracking thread of DTAM [20] and the RGB-D visual odometry system proposed in [31]. However, it differs from both of these methods in various aspects. First, we propose to use a robust error function in the data term, instead of the more fragile $\mathbf{L}^2$-norm favored by [20] and [31]. This strengthens our depth-based warp estimation allowing us to cope with occlusions and changes in illumination. Secondly, in order to be able to cope with the large motions we expect from small agile objects moving in the foreground, we do not make any assumption of small rotations and we use accurate linearizations that are updated in every iteration.

3 Problem Statement

Given a sequence of images of a dynamic scene containing an unknown number of independently moving objects and given a choice of reference frame, the aim of this work is to segment the scene into the different rigid bodies, to obtain a depth estimate for each pixel in the reference frame and to track the 3D motion of the objects with respect to the camera throughout the sequence. Once the dense depth map and the motion of the segmented objects have been estimated, we use them to augment the original video sequence to provide an automatic marker-free AR approach for dynamic scenes, Figures 1 and 2.

3.1 Notation and Preliminaries

The input to our algorithm is an F frame sequence of color images $I_1(\mathbf{x}),\ldots,I_F(\mathbf{x})$ where $I_r(\mathbf{x})$ is chosen to be the reference frame. We use $\Omega \subset \mathbb{R}^2$ to refer to the continuous domain that contains all valid coordinates within the reference frame. The 2D coordinates of a point in the reference image are denoted as $\mathbf{x} = (x,y)^T \in \Omega$ and its homogeneous coordinates as $\dot{\mathbf{x}} = \left(\mathbf{x}^T,1\right)^T$. The perspective projection of a 3D point expressed in the canonical camera coordinate frame $V_c = (x,y,z)$, is given by $\pi(V_c)$ where $\pi : \mathbb{R}^3 \to \mathbb{R}^2$ $\pi(V_c) = (x/z, y/z)^T$. We now consider a calibrated camera, i.e. its 3×3 intrinsic calibration matrix K is known in advance.

We denote the inverse depth map to be the function $d(\mathbf{x}) : \Omega \to \mathbb{R}$ that assigns its inverse depth value d to each point $\mathbf{x} = (x,y)^T \in \Omega$ in the reference frame. The back-projection of a point $\mathbf{x}$ in the reference frame given its inverse depth d can now be expressed as $\pi^{-1}(\mathbf{x},d) = \left(\frac{1}{d}K^{-1}\dot{\mathbf{x}}\right) \in \mathbb{R}^3$. Finally, to describe the relative pose of the camera at the time instant m with respect to the reference time instant r we define a rigid transformation $T_m(\cdot)$ that encodes its parameters. In other words, $T_m(V)$ is the 3D rigid transform $T_m(V) = RV + t$ where R and t are the rotation matrix and translation vector that align the point V in the reference frame to its position at time m.

As we are interested in dynamic scenes, the reference image is considered to be composed of a set of N regions or segments, corresponding to the projection of the independently moving objects onto the reference frame I_r. More precisely, for every region ℓ, we define the set of rigid transformations $\mathscr{T}_\ell = \{T_{\ell 1},\ldots,T_{\ell F}\}$ that map points on rigid body ℓ from its position at reference time r to every time $m \in \{1,\ldots,F\}$.

We can now define the problem of dense multibody structure and motion as the joint estimation of the following three sets of variables:

- $d(\mathbf{x})$: a dense inverse depth map that assigns an inverse depth value $d \in \mathbb{R}$ to each pixel x in the reference frame.

- A segmentation of the reference image domain Ω into disjoint regions which correspond to the N independently moving rigid bodies. We represent this segmentation via the labeling function $L(\mathbf{x}) : \Omega \to \{1,\ldots,N\}$ that assigns to each point $\mathbf{x}$ a label $\ell \in \{1,\ldots,N\}$ indicating which object it belongs to. Also, we denote by Ω_ℓ the segment that corresponds to a label ℓ: $\Omega_\ell = \{\mathbf{x} \in \Omega : L(\mathbf{x}) = \ell\}$.

- The set of rigid motion transformations for each independently moving body $\mathscr{T} = \{\mathscr{T}_1,\ldots,\mathscr{T}_N\}$. Each set $\mathscr{T}_\ell$ encodes the rigid transformations that align the pose of the region with label ℓ from its position at reference time r to its position at every time $m \in \{1,\ldots,F\}$ in the sequence as defined above.

Our approach simultaneously solves for the above three sets of variables. Note that the number of models N is initially set to a high value (over-segmentation), but the number of active (non-empty) models can be much smaller than N and is not fixed – it can decrease from one iteration to the next.

4 Energy Formulation

The structure of our framework is simple. We design an appropriate cost function in the form of an energy and we iteratively minimize it by re-estimating each parameter set (depth map, rigid motion and object labels) while keeping the other parameters fixed.

More precisely, we minimize the following energy with respect to the set of parameters $\{d(\mathbf{x}), L(\mathbf{x}), \mathscr{T}\}$:

$$E[d,L,\mathscr{T}] = \lambda E_{data} + \alpha E_{reg} + \beta E_{potts} + \gamma \text{MDL}\,, \tag{1}$$

where $\lambda,\alpha,\beta,\gamma$ are weighting parameters that control the balance of the different terms of the energy. Our energy contains a data term E_{data} that accounts for the sum of photometric errors over all the frames, a spatial regularization term E_{reg} for the depth map, discrete pairwise costs E_{potts} that encourage neighboring pixels to share the same model (or equivalently camera/motion parameters) and a Minimum Description Length prior MDL, that prefers compact solutions containing a small number of active regions. We now define each of the terms in detail.

4.1 Data Term E_{data}

E_{data} is the photo-consistency term:

$$E_{data}[d,L,\mathscr{T}] = \int_\Omega C(\mathbf{x}, d(\mathbf{x}), \mathscr{T}_{L(\mathbf{x})})\mathrm{d}\mathbf{x}. \tag{2}$$

The function C yields the average photometric error between every pixel $\mathbf{x}$ in the reference frame and its position in every other frame $I_m, m \in \{1,\ldots,F\}$, in the sequence. The function that maps a pixel $\mathbf{x}$ with depth d in the reference time r to its corresponding location in a different time instant of the sequence can be expressed as:

$$\mathscr{P}(\mathbf{x},d,T) = \pi\left(KT\left(\pi^{-1}(\mathbf{x},d)\right)\right) \tag{3}$$

where K is the known calibration matrix and T is the rigid transformation that aligns the two frames. We now define the photometric cost:

$$C(\mathbf{x},d,\mathscr{T}_\ell) = \sum_{m=1}^{F} \rho\left(\left|I_r(\mathbf{x}) - I_m\left(\mathscr{P}(\mathbf{x},d,T_{\ell m})\right)\right|^2\right) \tag{4}$$

where $\rho(e^2)$ is a robust norm. Note that in the above cost, we have used grayscale versions I_r, I_m of the input images, but the method can be easily extended to color. The robust norm $\rho(e^2)$ is a truncated Huber norm [14]:

$$\rho(e^2) = \begin{cases} e^2/2\sigma_\rho, \text{ if } e \leq \sigma_\rho \\ e - \sigma_\rho/2, \text{ if } \sigma_\rho < e \leq \theta_\rho \\ \theta_\rho - \sigma_\rho/2, \text{ if } \theta_\rho < e \end{cases} \tag{5}$$

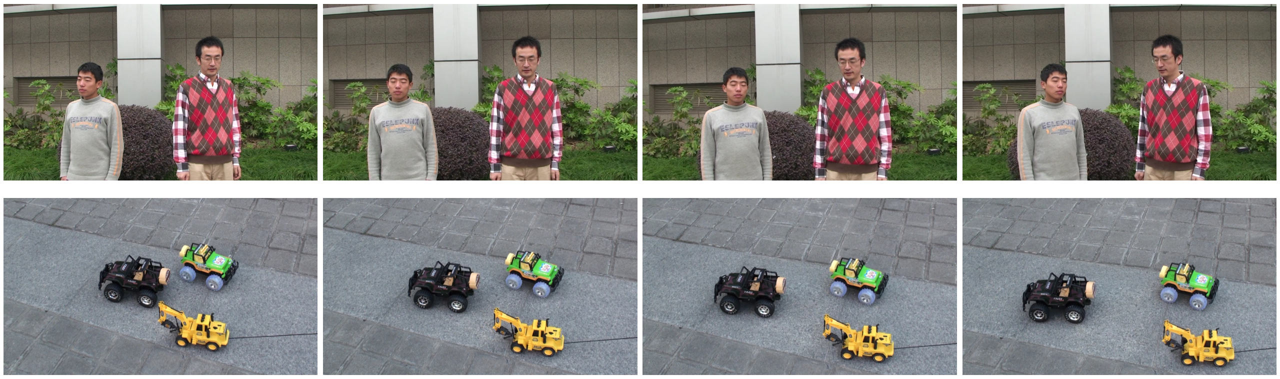


Figure 3: Four frames from each of the **two-men** and **toy-cars** input sequences.

where σ_ρ is a relatively small scale parameter that makes the derivative $\rho'(s)$ bounded near 0 and helps in the numerical stability of motion estimation. Also, θ_ρ is a saturation parameter that accounts for outliers in photometric matching (e.g. due to occlusions). Intuitively, the data term for each pixel accounts for the average photometric error of mapping that pixel to every other frame in the sequence for a given depth value d, labeling L and set of rigid transformations $\mathscr{T}$. Note that we use a spatial discretization of the term (2), in order to implement it and combine it with the purely discrete terms of motion segmentation.

4.2 Inverse Depth Regularization Term E_{reg}

The second term of our energy (1) corresponds to the spatial regularization of the inverse depth map $d(\mathbf{x})$:

$$E_{reg}[d] = \int_\Omega |\nabla d(\mathbf{x})|_\varepsilon \, \mathrm{d}\mathbf{x} \tag{6}$$

where ∇ denotes denotes the 2D gradient operator. Following [38, 20], we define the regularizer using the Huber norm $|\cdot|_\varepsilon$ [14]:

$$|\nabla d|_\varepsilon = \begin{cases} |\nabla d|^2/2\varepsilon, \text{ if } |\nabla d| \leq \varepsilon \\ |\nabla d| - \varepsilon/2, \text{ otherwise} \end{cases} \tag{7}$$

The Huber norm combines quadratic regularization for small magnitudes of the gradient with the discontinuity preserving properties of Total Variation for larger magnitudes of the gradient. Note that we consider a spatial discretization of the E_{reg} too.

4.3 Discrete Pairwise Costs E_{potts}

E_{potts} is a Potts model energy, a regularizer over the segmentation that encourages neighboring pixels to share the same label/model, or equivalently to belong to the same rigid object:

$$E_{potts}[L] = \sum_{\mathbf{x} \in D(\Omega)} \sum_{\mathbf{y} \in \mathscr{N}_\mathbf{x}} w_{\mathbf{xy}} \, \Delta(L(\mathbf{x}) \neq L(\mathbf{y})) \tag{8}$$

where $D(\Omega)$ is the discretization of the reference image domain on the image grid, i.e. the set of image pixel coordinates. $\mathscr{N}_\mathbf{x}$ is the set of pixels that neighbor the pixel $\mathbf{x}$, based on a specific pixel connectivity (we have used 4-pixel connectivity). Also, L is the assignment of rigid models to pixels and $\Delta(\cdot)$ is the discrete Dirac delta function that takes value 1 if the containing statement is false, and value 0 otherwise. $w_{\mathbf{xy}}$ is a positive weighting that takes into account the similarity between the appearance of pixels in the reference image, and more strongly encourages pixels of similar appearance to take the same label:

$$w_{\mathbf{xy}} = \exp\left(-||\mathbf{M}(\mathbf{x}) - \mathbf{M}(\mathbf{y})||_1/\sigma_w\right) \tag{9}$$

where $\mathbf{M}$ is formed by applying a 7×7 median filter across the color reference image I_r, and σ_w is a scale parameter. This use of the median filter substantially contributes to the effectiveness of our approach, and gives the crisp object boundaries required for convincing reconstruction.

It is worth mentioning that the continuous analogue of (8) is the sum of weighted lengths of the segment boundaries, an extension of the classical Mumford-Shah model that is widely used in variational methods [7]. This could be solved alongside the MDL cost using techniques such as [39].

4.4 Minimum Description Length Prior

The final term (MDL) is a sparsity inducing minimum description length prior:

$$\mathrm{MDL}[L] = \sum_{\ell=1}^{N} \Delta(\Omega_\ell \neq \emptyset) \tag{10}$$

which measures the number of non-empty segments Ω_ℓ. This term induces a fixed cost for each non-empty model to encourage compact solutions.

5 Optimization of the Energy

The optimization is a direct hill climbing approach, that iteratively minimizes the energy (1) alternating the estimation of each parameter set (depth map d, rigid motion parameters $\mathscr{T}$ and object labeling L) assuming that the other sets remain fixed: see Algorithm 1.

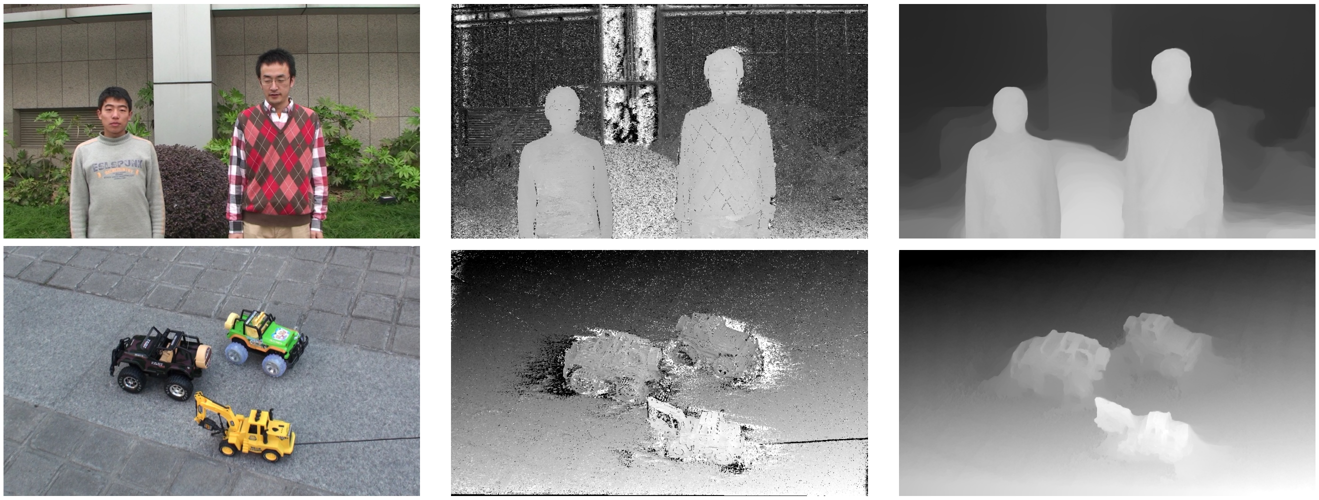
Algorithm 1: Multi-rigid depth estimation

```
Initialize $L$;
Initialize $\mathscr{T}$;
for $alternation = 1, .., N_{alt}$ do
    Fix $L$ and $\mathscr{T}$. Update $d$ by minimizing (1) w.r.t. $d$;
    Fix $L$ and $d$. Update $\mathscr{T}$ by minimizing (1) w.r.t. $\mathscr{T}$;
    Fix $\mathscr{T}$ and $d$. Update $L$ by minimizing (1) w.r.t. $L$;
```

We will discuss the initialization in Section 6. Note that in the above algorithm, N_{alt} denotes the number of alternations. In our experiments, we have used $N_{alt} = 2$, since we observed that this value was in all the cases sufficient to get an accurate result.

5.1 Step 1: Depth Map Estimation

The first step of the optimization of the energy (1) involves the estimation of the dense inverse depth map d given a segmentation of the scene into regions L and the set of motion matrices relative to



Reference frames

Initialization of depth maps

Final regularized depth maps

Figure 4: Estimated dense inverse depth maps $d(\mathbf{x}))$ for the **two-men** sequence (top row) and the **toy-cars** sequence (bottom row). The left column shows the reference image. The middle column shows the initialization of $d(\mathbf{x}))$, which is yielded via point-wise minimization of the photometric cost E_{data}. The right column shows the final regularized result that minimizes $\lambda E_{data} + \alpha E_{reg}$.

the camera $\mathscr{T}$. Since only the data and regularization terms depend on the depth map, in this step we seek to minimize:

$$\lambda E_{data} + \alpha E_{reg} = \int_{\Omega} \{\lambda C(\mathbf{x}, d(\mathbf{x})) + \alpha \left|\nabla d(\mathbf{x})\right|_{\varepsilon}\} \, \mathrm{d}\mathbf{x} \tag{11}$$

with respect to $d(\mathbf{x})$. Note that for the sake of simplicity, we write in this subsection $C(\mathbf{x}, d)$ instead of $C(\mathbf{x}, d, \mathscr{T}_{L(\mathbf{x})})$, since the mapping $\mathscr{T}_{L(\mathbf{x})}$ is fixed. This energy is almost equivalent to the one optimized in the 3D reconstruction thread of the real-time dense tracking and mapping system DTAM [20]. The only differences are that we use the truncated Huber norm and not the $\mathbf{L}^1$-norm as the robust function $\rho(\cdot)$ in the data term and we do not make use of a space varying weighting for the regularizer. Furthermore, while the optimization in DTAM is carried out online for a number of key frames at a time, our approach is batch i.e. we treat all the images in the sequence at once after the acquisition.

However, we can adopt the same variational approach to solve for the inverse depth map. An auxiliary function $d'(\mathbf{x}) : \Omega \to \mathbb{R}$ is used to form a quadratic relaxation of the cost (11):

$$\int_{\Omega} \{\lambda C(\mathbf{x}, d) + \frac{1}{2\theta_n}(d - d')^2 + \alpha \left|\nabla d'\right|_{\varepsilon}\} \, \mathrm{d}\mathbf{x} \tag{12}$$

where $\theta_n > 0$ is a varying inverse weight of the quadratic term, which is used to bring the variables d and d' close together. We follow an iterative approach, where the relaxation tightens with each iteration i.e. θ_n is a decreasing sequence that tends to zero, as in [20] (up to a scaling): $\theta_{n+1} = \theta_n(1 - C_n n)$, where $C_n = 10^{-3}$, if $\theta_n \geq 0.005\theta_0$ and $C_n = 10^{-4}$, otherwise.

Duplication of the optimization variable via relaxation decouples the linearized data and regularization terms, decomposing the optimization problem into two, each of which can be solved efficiently. We follow the paradigm of the large displacement optical flow method [32], which avoids the loss of detail inherent to common coarse-to-fine approaches [41, 5]. We alternate between solving for the auxiliary and original functions assuming the other fixed. The optimization of the energy with respect to the auxiliary function d' involves the sum of the quadratic and the regularization terms: $\frac{1}{2\theta_n}(d - d')^2 + \alpha \left|\nabla d'\right|_{\varepsilon}$. This is a convex energy and is optimized in a similar way to the ROF denoising model of [27] using a primal-dual approach [5]. On the other hand, minimizing the cost with respect to the original function d involves the sum of the quadratic relaxation and data terms: $\lambda C(\mathbf{x}, d) + \frac{1}{2\theta_n}(d - d')^2$. Although non-convex, this term is point-wise independent and is easily solved via exhaustive search over a discretized set of values for the inverse depth. Following [20] and in order to refine this quantized result, we afterwards apply a Newton step using numerical derivatives. This optimization can be parallelized with the use of GPUs and solved very efficiently.

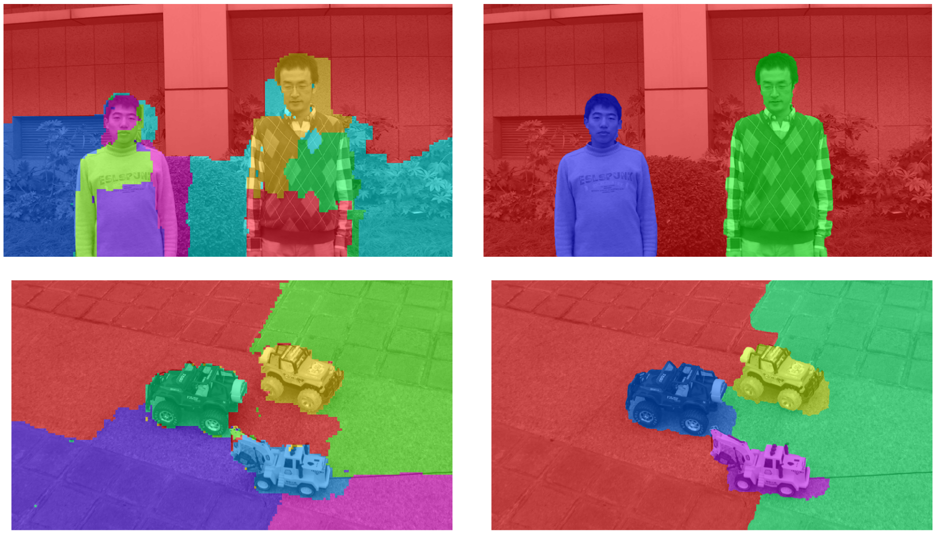
Note that when $n = 0$, i.e. the first internal iteration of Step 1, $d(\mathbf{x})$ is initialized by minimizing the term $C(\mathbf{x}, d)$ (without the quadratic relaxation term), via point-wise exhaustive search (see Figure 4).

5.2 Step 2: Rigid Motion Estimation

Given a new estimate of the inverse depth map, the second step of the iteration involves the estimation of the set of rigid motion transformations $\{T_{\ell m}\}$ that align the pose of each region ℓ in each time m with the camera coordinate frame. This energy can be decoupled and optimized independently for each region and frame:

```
for every $\ell \in \{1, \ldots, N\}$ with $\Omega_\ell \neq \emptyset$ (active region) do
    for every $m \in \{1, \ldots, F\}$ do
        minimize w.r.t. $T_{\ell m}$ :
        $J = \int_{\Omega_\ell} \rho\left(\left|I_r(\mathbf{x}) - I_m(\mathscr{P}(\mathbf{x}, d, T_{\ell m}))\right|^2\right) \mathrm{d}\mathbf{x}$    (13)
```

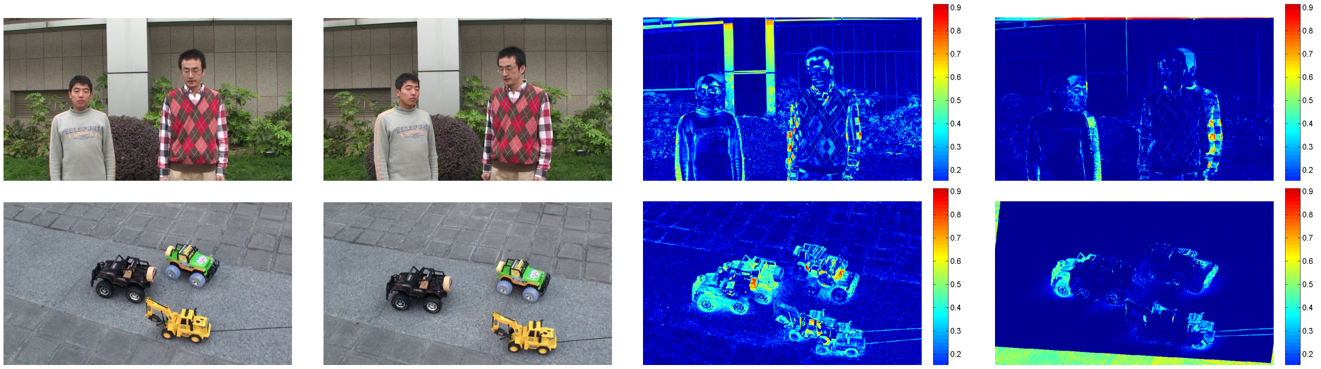
Our optimization strategy in this stage is most closely related to the 6-DOF camera tracking thread of DTAM [20] and the RGB-D visual odometry system of [31]. However, it differs from both of these methods in various aspects. First, we use a robust error function in the data term, instead of the non-robust $\mathbf{L}^2$-norm favored by [20] and [31]. This improves the robustness of our depth-based warp estimation allowing us to cope with occlusions and changes in illumination; it also means that the motion estimation step minimizes the same energy (1) as the rest steps, which is required to guarantee convergence of our batch method. Further, we do not make any assumption of small rotations, which increases our robustness to



Initialization

Final segmentation

Figure 5: **Segmentation Results:** The left image shows the results after the initialization and the right column shows the final segmentation given by our algorithm in both test sequences. Shadows are captured as belonging to the object as this results in a lower photometric error.



(a) Reference

(b) Current frame

(c) Photometric error before

(d) Final photometric error

Figure 6: Depth-based image warping estimated during the rigid motion estimation step. The estimation for each segment is done separately. (a) Reference frames. (b) Current frames. (c-d) Photometric error between the reference and the current image, (c) before the application of any warping and (d) after the application of the depth-based image warping using the. The regions where the errors in (d) are high correspond to occluded regions. Thanks to the robust photometric error, these regions are treated as outliers.

large camera motions and improves agile camera tracking. In this way, we can directly optimize (13) even for large motions (given a decent initialization), without needing to synthesize projections of the scene in a virtual camera (as e.g. done in [20]) or to constrain ourselves to pairs of frames that are close together (as e.g. done in [31]). In addition, we get accurate results without the need for temporal smoothness or any other prior. With this novel optimization for motion estimation, it became possible to bind the estimations of the different parameters (depth map, rigid motions and object labels) into a global minimization framework.

More precisely, we parametrize the rigid transform $T_{\ell m} = T_{\ell m}(p)$ with the 6-dimensional vector $p = [q, t]$, where q is the axis-angle parametrization of the rotation matrix and t is the translation vector. We derive the rotation matrix $R(q)$ from q using the Rodrigues formula:

$$R(q) = \mathrm{I}_3 + \sin(|q|)[\hat{n}(q)]_\times + (1 - \cos(|q|))[\hat{n}(q)]_\times^2 \qquad (14)$$

where I_3 is the 3×3 identity matrix, $\hat{n}(q) = q/|q|$ is the rotation axis and $[\hat{n}]_\times$ is the operator of cross product with $\hat{n}$ in the form of a 3×3 matrix: $\hat{n} \times v = [\hat{n}]_\times v$, for any $v \in \mathbb{R}^3$. Note that under this parametrization, $|q|$ is equal to the rotation angle. It is important to mention that the above formula does not assume any linearization.

Given the depth map and the rigid motion parameters of each object, each point $\mathbf{x} \in \Omega_\ell$ in the reference that belongs to it can be mapped to its corresponding position in the current frame using the transform (3), which can be viewed as a warp function $W(\mathbf{x}; p) = \mathscr{P}(\mathbf{x}, d, T_{\ell m}(p))$. The motion estimation can thus be viewed as an image registration problem. We derive our minimization algorithm by following the paradigm of Lucas-Kanade, as revisited by Baker and Matthews [2]. Since we want to cope with large camera motions, we do not simplify the warp $W(\mathbf{x}; p)$ beforehand. This means that, in our formulation, the composition of two warps as well as the inverse warp have not any simple analytical expression. Therefore, following the terminology of [2], we apply a forwards additive optimization method with a robust function.

Our motion estimation algorithm is iterative and finds a local minimum of the energy $J(p)$ (13) as a function of the motion parameters p. It starts from some initial estimate of p, given from the overall initialization or from the previous alternation of Algorithm 1. Afterwards, in every iteration, the previous estimate of p is updated to $p+\Delta p$. The value Δp is specified by minimizing $J(p+\Delta p)$. For this minimization, in every point $\mathbf{x}$ of the current region, the composition $I_m(W(\mathbf{x};\cdot))$ is linearized around p and afterwards the robust function $\rho(s)$ is linearized around $|I_r(\mathbf{x}) - I_m(W(\mathbf{x};p))|^2$. Note that these linearizations are not done beforehand, but during each iteration. In addition they are done around a different point in every iteration, which increases their accuracy. These approximations result to an energy that is quadratic w.r.t. Δp and the optimum is computed by solving a linear system. This requires to compute in every iteration the jacobian of the warp $\frac{\partial W(x;p)}{\partial p}$ evaluated at the current estimate p. The corresponding analytic expressions can be derived by using the equations (3),(14) and applying the chain rule.

5.3 Step 3: Multibody Segmentation

Given new estimates of the inverse depth map and rigid motions, the final step of the optimization of the energy (1) is the labeling step consisting on assigning each pixel in the reference image a label indicating which region it belongs to.

In this case, the terms of the energy that depend on the labeling are the data term, pairwise costs and the MDL prior. Therefore, we minimize:

$$\lambda E_{data} + \beta E_{potts} + \gamma \mathrm{MDL} \; = \lambda \int_{\Omega} C(\mathbf{x}, d(\mathbf{x}), \mathscr{T}_{L(\mathbf{x})})\, \mathrm{d}\mathbf{x} +$$
$$\beta \sum_{\mathbf{x}} \sum_{\mathbf{y}\in\mathscr{N}_{\mathbf{x}}} w_{\mathbf{xy}}\, \Delta(L(\mathbf{x}) \neq L(\mathbf{y})) + \gamma \sum_{\ell=1}^{N} \Delta(\Omega_\ell \neq \emptyset)$$

with respect to $L(\mathbf{x})$. After the necessary spatial discretization, this cost can be efficiently solved using a variant on the graph-cut [17] based algorithm α-expansion [4], that handles MDL costs [8, 19].

6 Initialization

To initialize our overall algorithm we require initial estimates for the labeling L (i.e. the segmentation of the scene into regions) and the rigid motion transformations for every region $\mathscr{T} = \{\mathscr{T}_1, ..., \mathscr{T}_N\}$.

6.1 Initialization for the Labeling $L(\mathbf{x})$

We initialize the segmentation of the scene into independently moving rigid objects by performing simultaneous motion segmentation and reconstruction on dense optical flow data. First we extract dense optical flow from video using a multi-frame optic flow algorithm [12] specifically tailored for dynamic sequences (multi-rigid or deformable). This algorithm returns the flow, or pixel-wise deformation, of a reference frame to every other frame in the sequence.

Given these dense tracks we use the energy-based multiple model fitting algorithm of [15] for motion segmentation. Much like the multibody segmentation of section 5.3, this method assigns labels representing a choice of model parameters to every pixel in the image using graph-cuts. Parameters and the assignment of pixels to a model are chosen to minimize the sum of algebraic errors that correspond to the cost of assigning a 6-DOF orthographic model to each pixel, and to respect soft spatial constraints which say that neighboring points should normally belong to the same model. In this step, the considered models are orthographic rigid motion models and the cost of assigning a point to a rigid body with motion parameters $\mathscr{T}_\ell$ is its average image re-projection error throughout the sequence. This simplifying assumption of an orthographic camera model is used only for the initialization. The initial set of proposed models is formed by fitting a rigid model to each point and its 8 nearest neighbors. Our initialization is almost identical to the whole method of [9], however, we substitute the model overlap constraints with pairwise costs that encourage neighboring pixels to share the same label.

The result of our initialization on two example sequences can be seen in the left column of Figure 5. Its intended behavior is to provide an over-segmentation of the scene where the motion boundaries are respected.

6.2 Initialization of Rigid Transformations $\mathscr{T}$

Given a segmentation of the scene into independently moving regions $\{\Omega_\ell\}$ we can estimate the set of rigid transformations $\{\mathscr{T}_\ell\}$ associated with each rigid body $\ell \in \{1,\dots,N\}$ in every time $m \in \{1,\dots,F\}$. For this purpose, given the segmentation and the dense tracks computed from optical flow in the previous section, we use an off the shelf rigid S*f*M algorithm (such as bundler [30] or ACTS [43]) independently on each of the regions to initialize the motion matrices.

7 Towards Online Reconstruction

While the method presented is batch, it can be modified to an online (and on future hardware, real-time) algorithm. The principal bottleneck in our current code, preventing real-time online reconstruction is the re-estimation of camera parameters for past frames, and the subsequent recalculation of the cost volumes used for depth and object assignment. Fortunately, DTAM [20] has shown that such re-estimation is unnecessary, and that a cost volume for depth estimation can be maintained by assuming that the camera parameters for previous frames were correct, and never revisiting them. We will briefly outline two approaches based on this for extending our work to an online setting:

Multi-volume based approach The most direct online implementation of our method would maintain online one copy of the entire cost volume (i.e. the data term $C(\mathbf{x},\cdot,\mathscr{T}.)$ of eq. (2)) for all choices of camera parameters $\mathscr{T}$ associated with any object label, and updating the decision to assign pixels to particular models by performing efficient variants of dynamic α-expansion at keyframes [1].

Two-volume based approach A computationally less intensive approach would be to maintain only two sets of costs online. The first cost volume being the cost of assigning a pixel a particular depth under the assumption that previous camera parameters, and the current assignment of points to objects are correct (i.e. the data term $C(\mathbf{x},\cdot,\mathscr{T}_{L(\mathbf{x})})$), and this volume will be used to refine depth estimates. The second set of costs $C(\mathbf{x}, d(\mathbf{x}), \mathscr{T}.)$ will be used to update $L(\mathbf{x})$, the assignment of pixels to camera parameters, under the assumption that both camera parameters and pixel depths are accurate. This is substantially more efficient than the previous approach but carries the disadvantage that some costs will be estimated incorrectly. However, both [16], and [20] show that maintaining two threads in this way (corresponding to online estimation of camera parameters, and depth) is possible, as would be the addition of a third thread corresponding to segmentation. This second approach, should be feasible on contemporary hardware and it is an active area of research.

8 Experiments

Our experiments have been carried out on real video footage of dynamic scenes acquired with a hand-held digital camera provided by [42]. The size of the images is 960 × 540 pixels. We show results on two sequences. The **toy-cars** sequence is 27 frames long and shows three toy cars moving on the ground plane in different directions while the camera moves. The 30 frames long **two-men** sequence shows two people turning around rigidly while the camera also moves. The sequence contains multiple texture-less regions and displays significant occlusions. In Figure 3 we show four frames from each of the sequences.

8.1 Parameter Choice

To make the choices of ideal parameters stable over multiple sequences and to allow a direct comparison between weights (i.e. a weight of 1 for normalized versions of both α and β of eq. (1) should indicate that they have similar importance), we make several normalizations. First, the intensities of the input images $I_k(\mathbf{x})$ are normalized in the range $[0,1]$. The normalized parameters, denoted via a hat above the corresponding symbol, are defined as follows:

$\lambda = \hat{\lambda}/(FP)$ (see eq. (1)) where P is the number of pixels in a frame; Also, $\alpha = \hat{\alpha}/(P\,\Delta_d)$ where Δ_d is an estimate of the range of values of inverse depth computed in the initialization. Since only the relative values of the weights in (1) affect the result, we always set $\hat{\alpha} = 1$. Similarly, $\beta = \hat{\beta}/(P\,N_{con})$, where $N_{con} = 4$ is the number of neighbors of each pixel. In eq. (7), we set $\varepsilon = \hat{\varepsilon}\ \Delta_d$ and in eq. (12), $\theta_n = \hat{\theta}_n \Delta_d^2$. Note, the only free parameter for the sequence $\hat{\theta}_n$ is the initial value $\hat{\theta}_0$ that controls its scale. In the exhaustive search of Step 1, we use N_s quantized levels of $d(\mathbf{x})$, equally spaced within its estimated range.

After normalization, for almost all the parameters, the same values work across multiple sequences. We observe that the results vary smoothly with the choice of parameters and a wide range of parameters give convincing results.

In the results we show of the **two-men** and **toy-cars** sequences, we used the following parameters: $\hat{\lambda} = 1.3$, $\hat{\beta} = 1.1$, $\hat{\theta}_0 = 5.6$, $\hat{\varepsilon} = 3 \cdot 10^{-4}$, $\sigma_p = 6.4 \cdot 10^{-3}$, $\theta_p = 0.04$, $\sigma_w = 0.033$ and $N_s = 64$. The only parameter varied was γ, which we set to $1.6 \cdot 10^{-3}$ in the case of **two-men** sequence and $3.2 \cdot 10^{-7}$ in the case of **toy-cars**.

8.2 Results

Figure 4 shows results from the estimation of the depth maps. The images in the middle column show the intermediate results obtained after running only the exhaustive search over the discretized values of depth. The right column shows the final depth maps obtained for both sequences after the regularization step. Note the sharp depth boundaries and the accuracy of the result.

The left column in Figure 5 shows the segmentation results obtained using our initialization approach described in Section 6 based on the segmentation of dense optical flow. Our initialization results in an over-segmentation of the reference frame. The final results of our alternating approach are shown on the right column of Figure 5. Our iterative algorithm for multibody segmentation, tracking and reconstruction improves this initial estimate substantially.

The performance of our novel dense tracking algorithm, that uses all the pixels in the image to track each independently moving region, is assessed in Figure 6. The two left-most columns of Figure 6 show the reference frame and another input frame in the sequence. The two right-most columns show the residual errors (color coded using a heat-map) of warping all the pixels in the input frame back to the reference frame. The images in Figure 6(c) show the residual errors before estimating the warp and the images in Figure 6(d) show the errors after the estimation of the warp function. The errors are seen to go down significantly which accounts for the correct estimation of the rigid transformation T that aligns the two frames. The regions where the errors in Figure 6(d) are high correspond to occluded regions. Thanks to the robust photometric error, these regions are treated as outliers.

Figure 7 shows four frames of the dynamic 3D model of the **two-men** sequence from two different view points. The 3D renderings show how the rotating motion of the two men is recovered accurately, as well as the detail of the 3D structure. The bottom row shows closeups of the 3D models of the men where an accurate reconstruction of the faces is achieved with the facial features preserved. Note also the detailed structure on the men's knitted sweaters reconstructed with our algorithm. We show 3D models of the reconstructed **toy-cars** sequence in Figure 8 from a novel viewpoint. This sequence is particularly challenging due to the small size of the tracked objects. This induces a small aperture angle in each of the tracked regions, making them hard to track. The 3D reconstructions of the moving objects show accurate 3D models. The small imperfection on the cars are caused by the shadows having been segmented as part of the cars.

Figures 2, 9 show frames of the video sequences after augmentation. In Figure 2, we have augmented the faces of the men with a mustache and the knitted sweaters with splashes. The augmentation seems almost perfect. Figure 9 shows an AR application on the **toy-cars** sequence that illustrates how the availability of a dense depth map allows high quality occlusion reasoning. The teapot situated on the pavement is plausibly occluded by one of the cars in some of the frames.

For better inspection of the quality of our dense 3D models and the AR sequences, we provide demo videos in the supplementary material as well as on the following URL:
`http://vision.eecs.qmul.ac.uk/humanis/dense_multibody` .

8.3 Runtime Performance Analysis

As the runtime of our algorithm depends on the number of frames used for depth estimation and the number of rigid segments proposed at every iteration of Algorithm 1, we here report the runtime per frame, per rigid region proposal.

Our current implementation is in unoptimized Matlab code on a 64 bit i5-2500K machine, with NVIDIA GTX590 GPU, and makes use of the Matlab Parallel Computing Toolbox. We have a runtime of **0.736** seconds per frame per region for **two-men** sequence and **0.682** seconds for **toy-cars** sequence. The total runtimes for these two sequences were 265 and 221 seconds respectively, whereas the runtimes per frame were 8.83 and 8.19 seconds per frame respectively. Note that these runtimes do not include the steps of the overall initialization (Section 6).

9 Conclusions

In this paper we show a novel approach to joint dense multibody segmentation, tracking and 3D reconstruction from sequences taken with a single handheld camera. The strength of our approach comes from showing how dense depth maps, the segmentation of the scene into rigid bodies and the rigid transformations describing their motion can be simultaneously estimated by optimizing a single cost function using a hill-climbing approach. We show detailed and accurate 3D reconstructions and apply our approach to Augmented Reality applications that allows dense occlusion reasoning.

Our method has several limitations. First, our current system is batch and treats all the images at once after the acquisition. However, in Section 7 we have described two different approaches to convert it into an online method which is currently an active area of research. An additional limitation of our approach is that it can only reconstruct the pixels visible in the reference frame, since it is effectively 2.5D approach (the 3D reconstruction relies on a depth map), and therefore holes may appear in the background for long sequences. This limitation would be mitigated once the system becomes online and the depth maps estimated for different key-frames are merged. Similarly to other motion segmentation algorithms, our current system copes poorly with shadows and often pixels in shadow are labeled as belonging to the moving object. Finally, for 3D reconstruction, the independent scale ambiguity between segments is currently fixed manually. However, this could be easily solved with the strategies described in [21].

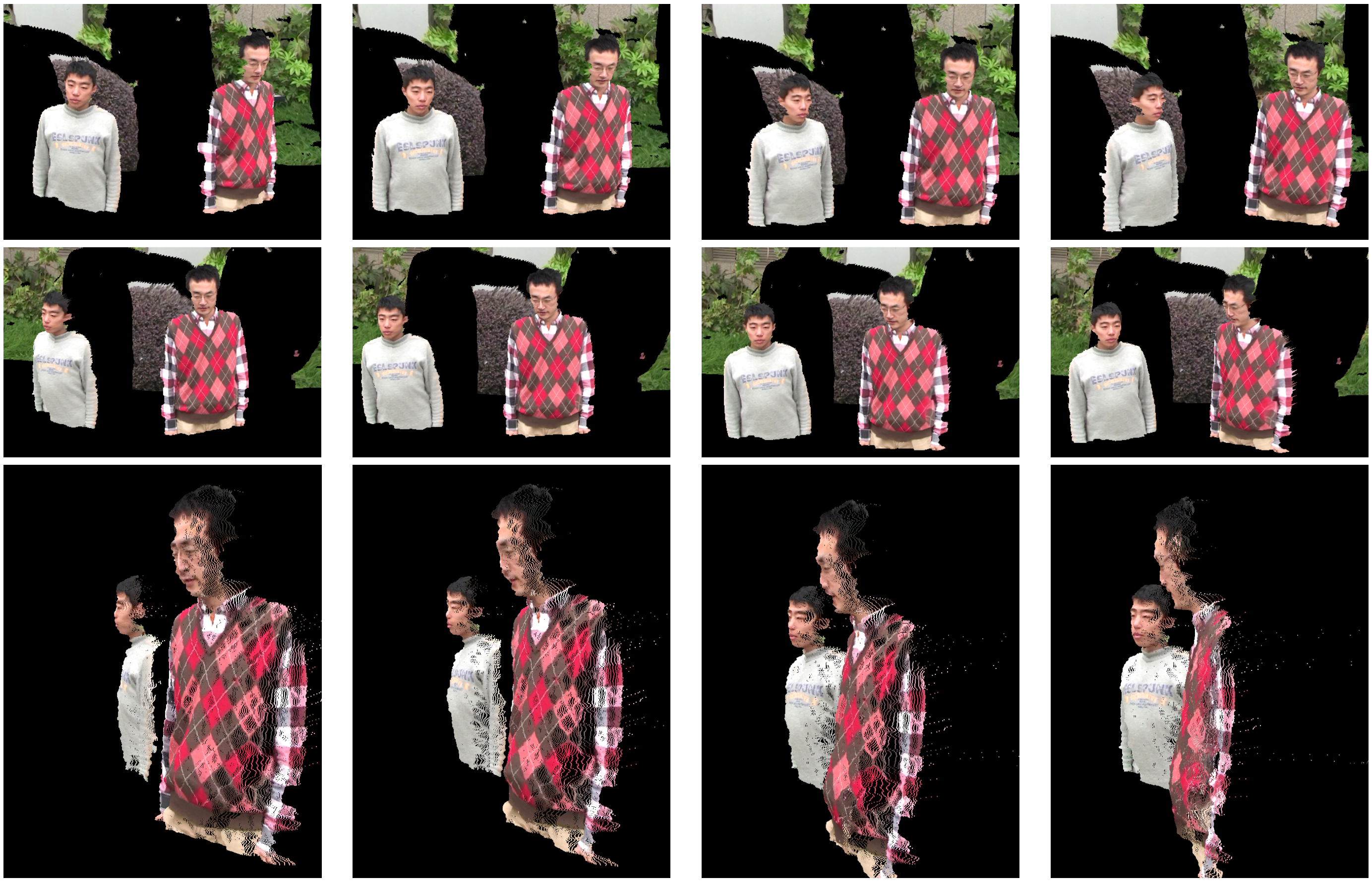


Figure 7: Rendering of the estimated dense 3D models from different viewpoints for **two-men** sequence.

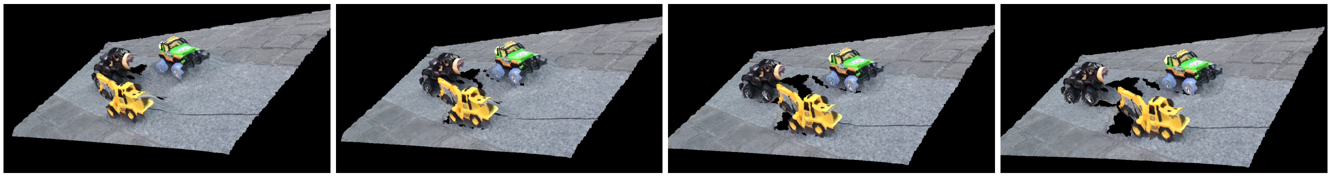


Figure 8: Rendering of the estimated dense 3D models for **toy-cars** sequence.

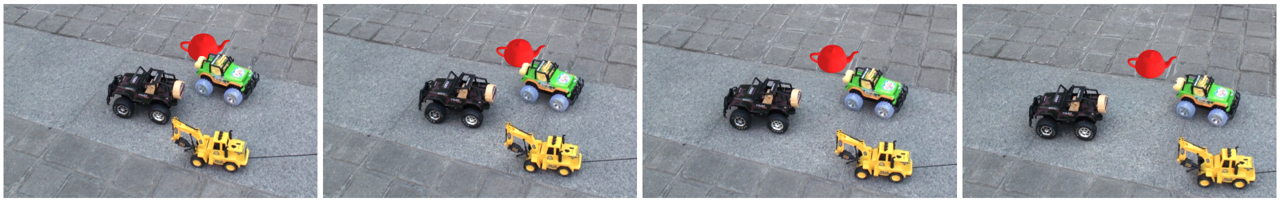


Figure 9: Augmented reality application for **toy-cars** sequence.

10 Acknowledgements

This work was funded by the European Research Council under ERC Starting Grant agreement 204871-HUMANIS. We thank G. Zhang for providing the input sequences (**two-men** and **toy-cars**) and for helping us in the usage of the ACTS software. We also thank S. Lovegrove for helping us with his open-source Pangolin code, which we used as core-code in our 3D visualizations and AR applications. Finally, we are grateful to R. Newcombe and A. Davison for fruitful discussions on the DTAM system.

References

[1] K. Alahari, P. Kohli, and P. H. S. Torr. Dynamic hybrid algorithms for map inference in discrete mrfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(10):1846–1857, 2010.

[2] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221 – 255, March 2004.

[3] M. Bleyer, C. Rother, and P. Kohli. Surface stereo with soft segmentation. In *CVPR*, pages 1570–1577, 2010.

[4] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, 23, 2001.

[5] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *JMIV*, 2011.

[6] J. Costeira and T. Kanade. A multi-body factorization method for motion analysis. In *ICCV*, pages 1071–1076, 1995.

[7] D. Cremers, T. Pock, K. Kolev, and A. Chambolle. Convex relaxation techniques for segmentation, stereo and multiview reconstruction. In *Markov Random Fields for Vision and Image Processing*. MIT Press, 2011.

[8] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. Fast approximate energy minimization with label costs. *IJCV*, 2010.

[9] J. Fayad, C. Russell, and L. Agapito. Automated articulated structure and 3d shape recovery from point correspondences. In *IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, November 2011.

[10] A. Fitzgibbon and A. Zisserman. Multibody structure and motion: 3-d reconstruction of independently moving objects. In *European Conference on Computer Vision (ECCV)*, Dublin, Ireland, 2000.

[11] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(8):1362–1376, 2010.

[12] R. Garg, A. Roussos, and L. de Agapito. Robust trajectory-space tv-l1 optical flow for non-rigid sequences. In *EMMCVPR*, 2011.

[13] J.-Y. Guillemaut and A. Hilton. Joint multi-layer segmentation and reconstruction for free-viewpoint video applications. *International Journal of Computer Vision*, 93(1):73–100, 2011.

[14] P. J. Huber. *Robust Statistics*. John Wiley & Sons, 1981.

[15] H. Isack and Y. Boykov. Energy-based geometric multi-model fitting. *International Journal of Computer Vision (IJCV)*, 97(2), 2012.

[16] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007.

[17] V. Kolmogorov and C. Rother. C.: Comparison of energy minimization algorithms for highly connected graphs. in: Eccv. In *In Proc. ECCV*, pages 1–15, 2006.

[18] A. Kundu, K. M. Krishna, and C. V. Jawahar. Realtime multibody visual slam with a smoothly moving monocular camera. In *IEEE International Conference on Computer Vision. ICCV*, 2011.

[19] L. Ladicky, C. Russell, P. Kohli, and P. Torr. Graph cut based inference with co-occurrence statistics. In *ECCV*, 2010.

[20] R. Newcombe, S. Lovegrove, and A. Davison. DTAM: Dense tracking and mapping in real-time. In *ICCV*, 2011.

[21] K. Ozden, K. Schindler, and L. van Gool. Multibody structure-from-motion in practice. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2010.

[22] Q. Pan, C. Arth, E. Rosten, G. Reitmayr, and T. Drummond. Rapid scene reconstruction on mobile phones from panoramic images. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[23] Q. Pan, G. Reitmayr, and T. Drummond. Interactive model reconstruction with user guidance. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2009.

[24] C. Pirchheim and G. Reitmayr. Homography-based planar mapping and tracking for mobile phones. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2011.

[25] M. Pollefeys, D. Nistér, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell, C. Salmi, S. N. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles. Detailed real-time urban 3d reconstruction from video. *International Journal of Computer Vision*, 78(2-3):143–167, 2008.

[26] S. Rao, R. Tron, R. Vidal, and Y. Ma. Motion segmentation in the presence of outlying, incomplete or corrupted trajectories. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 32(10):1832–1845, 2010.

[27] L. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[28] K. Schindler, D. Suter, and H. Wang. A model selection framework for multibody structure-and-motion of image sequences. *International Journal of Computer Vision (IJCV)*, 79(2):159–177, 2008.

[29] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, New York, USA, June 2006.

[30] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: Exploring photo collections in 3d. In *SIGGRAPH Conference Proceedings*, pages 835–846, New York, NY, USA, 2006. ACM Press.

[31] F. Steinbrücker, J. Sturm, and D. Cremers. Real-time visual odometry from dense rgb-d images. In *Workshop on Live Dense Reconstruction with Moving Cameras at ICCV*, 2011.

[32] F. Steinbruecker, T. Pock, and D. Cremers. Large displacement optical flow computation without warping. In *ICCV*, 2009.

[33] J. Stuehmer, S. Gumhold, and D. Cremers. Real-time dense geometry from a handheld camera. In *Pattern Recognition (Proc. DAGM)*, pages 11–20, September 2010.

[34] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization approach. *IJCV*, 1992.

[35] R. Tron and R. Vidal. A benchmark for the comparison of 3d motion algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Minneapolis, USA, June 2007.

[36] R. Vidal, Y. Ma, and S. Sastry. Generalized principal component analysis (gpca). In *CVPR*, pages 621–628, 2003.

[37] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmalstieg. Pose tracking from natural features on mobile phones. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2008.

[38] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof. Anisotropic Huber-L1 optical flow. In *BMCV*, 2009.

[39] J. Yuan and Y. Boykov. Tv-based multi-label image segmentation with label cost prior. In F. Labrosse, R. Zwiggelaar, Y. Liu, and B. Tiddeman, editors, *BMVC*, pages 1–12. British Machine Vision Association, 2010.

[40] C. Zach. Fast and high quality fusion of depth maps. In *International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT)*, 2008.

[41] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. In *Pattern Recognition (Proc. DAGM)*, pages 214–223, 2007.

[42] G. Zhang, J. Jia, and H. Bao. Simultaneous multi-body stereo and segmentation. In *IEEE International Conference on Computer Vision (ICCV)*, Barcelona, Spain, November 2011.

[43] G. Zhang, X. Qin, W. Hua, T.-T. Wong, P.-A. Heng, and H. Bao. Robust metric reconstruction from challenging video sequences. In *CVPR*, 2007.