Consistent Depth Video Segmentation using Adaptive Surface Models

Farzad Husain, Babette Dellen, and Carme Torras

***Abstract*—We propose a new approach for the segmentation of 3-D point clouds into geometric surfaces using adaptive surface models. Starting from an initial configuration, the algorithm converges to a stable segmentation through a new iterative split-and-merge procedure, which includes an adaptive mechanism for the creation and removal of segments. This allows the segmentation to adjust to changing input data along the movie, leading to stable, temporally coherent, and traceable segments. We tested the method on a large variety of data acquired with different range imaging devices, including a structured-light sensor and a time-of-flight camera, and successfully segmented the videos into surface segments. We further demonstrated the feasibility of the approach using quantitative evaluations based on ground-truth data.**

***Index Terms*—Motion, Range data, Segmentation, Shape, Surface fitting.**

I. Introduction

RECENT advances in 3-D sensing have revived the interest in range data segmentation and extended their possible field of application to depth video segmentation. Time-of-flight (ToF) cameras and commercial structured-light devices (Kinect) allow acquiring depth images in real-time at a quality suitable for feature extraction, object recognition, 3-D reconstruction, tracking, mobile robot navigation and parts identification.

However, range image segmentation is an area in computer vision that is less mature than, e.g., color segmentation, and if we extend the idea of segmentation to maintain *consistency* in a range image sequence, then this field is almost unexplored. By consistency we mean that the same surface should get the same label in subsequent frames. Very little work has been done in this regard [1], [2], [3].

If we consider segmentation of a single depth image then most segmentation algorithms focus on either particular surface shapes [2], [4], or segment the depth data using local surface descriptors measuring surface orientation [5], planarity [6], curvature [7] or depth probability distribution [1]. Other algorithms rely on statistical inferences for making decisions during segmentation [8]. This makes these algorithms slower and unfeasible for real-time applications. Comparisons of different range image segmentation algorithms can be found in [9], [10], [11].

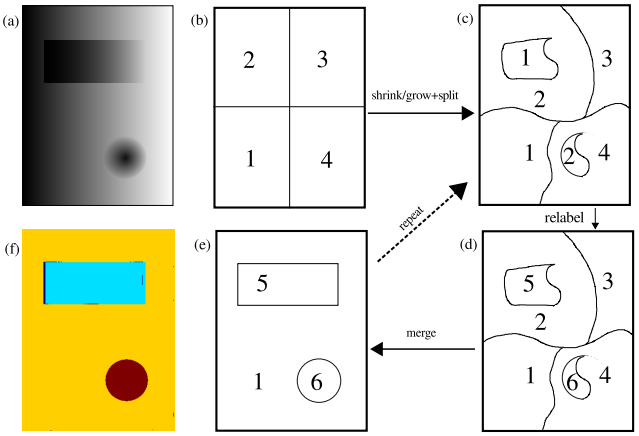

Fig. 1. Schematic illustrating the main mechanisms driving the segmentation procedure. (a) Input range data. (b) Initial configuration of segments. (c) Segments after growing/shrinking/splitting. (d) Segments after relabeling. (e) Segments after merging. Processes (c $\rightarrow$ d $\rightarrow$ e) are iterated until the final result is obtained. (f) Actual segmentation result obtained using our algorithm.

Depth video segmentation poses different demands than single depth image segmentation. Even small changes in camera position, or motions within a scene, can cause significant changes in the position and orientation of the surfaces from one frame to the next. For this reason, it is disadvantageous to work with a precisely defined segmentation based on local descriptors such as surface normals. Instead, we draw the segmentations from globally defined quadratic surface models which compete for label assignments within a local neighborhood, thus depending on the relative configuration of surfaces in the scene. In a related work, Leonardis *et al.* used superquadric models for segmenting a scene using a recover-and-select paradigm [12]. First, the data is partitioned into a predefined number of small areas (splitting), which are then merged depending on the superquadric fitting error. While this approach yields satisfactory results for single images, it cannot be easily extended to video because the solution is obtained through progressive merging. Hence decisions cannot be revoked. This however is a fundamental requirement when searching for a method that can adapt to changing data, which fails to be met by standard split-and-merge approaches [13], [14].

This shortcoming motivated us to develop a split-and-merge method in which splitting and merging mechanisms are active at all times during the application of the procedure. This way, wrong splits can be revoked, and merges be undone, giving rise to two competing processes. Convergence is characterized

Manuscript received xxx; revised xxx; accepted xxx. Date of publication xxx; date of current version xxx. This research is partially funded by the EU project IntellAct (FP7-269959), the Grup consolidat 2009 SGR155, the project PAU+ (DPI2011-27510), and the CSIC project CINNOVA (201150E088). B. Dellen acknowledges support from the Spanish Ministry of Science and Innovation through a Ramon y Cajal program.

F. Husain and C. Torras are with the Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Llorens i Artigas 4-6, 08028, Barcelona, Spain, (e-mail: {shusain, torras}@iri.upc.edu).

B. Dellen is with the RheinAhrCampus der Hochschule Koblenz, Joseph-Rovan-Allee 2, 53424 Remagen, Germany, (email: dellen@hs-koblenz.de).

by a constant number of segments, which can thus be easily determined. However, for this strategy to be successful, the competitive decision process for label assignment during the split-and-merge process has to be made dependent, at least partly, on relative values instead of absolute thresholds. The main mechanisms driving the segmentation of the proposed approach are illustrated in Fig. 1, allowing for the dynamic growing, shrinking, removing, and adding of seed points instead of only growth and removal of predefined regularly spaced seeds on an image grid as in [12].

Furthermore, our algorithm does not follow the conventional tracking approaches where a predefined motion model (constant velocity/acceleration model) is used. Segments are solely analyzed based on their extrinsic surface geometry and temporal coherence. We use an adaptive model fitting approach for clustering unlabeled points, which makes the algorithm flexible enough to accommodate non-rigid transformations as well. In our method, the shape and position of the segments are continuously adapted to the data such that the surface model fitting error is minimized. During the video segmentation, we enforce the following supporting constraints in addition to the already introduced split-and-merge mechanisms shown in Fig. 1:

1) Each segment has to stay connected.
2) A segment can only be merged with those segments that have been in its proximity since the time of its creation.
3) A segment cannot be smaller than a minimum size (predefined in 'number of pixels').
4) A segment cannot move further than a maximum distance from one frame to the next.

These constraints play a central role in inducing consistency between segmented surfaces with the passage of time.

Compared to a preliminary version of this work [15], the main enhancement is the inclusion of an adaptive mechanism that allows creating new segments and eliminating segments that are subthreshold. As a consequence, the method automatically adjusts the number of segments along the image sequence. This method further permits finding an initial segmentation from scratch by converging to a stable solution. Moreover, extensive quantitative results are reported for a wide range of datasets.

II. Related work

A. Video Segmentation

To the authors' knowledge, little work has been done using depth information as the primary vision cue for joint segmentation and tracking. Parvizi and Wu (2008) performed multiple object tracking using an adaptive depth segmentation method [1]. Time-of-flight depth was used to segment each frame independently by finding the connected components based on an absolute depth distance measure. The segments of adjacent frames were then associated with each other using a depth histogram distribution. However, this depth segmentation method is rather simple and does not partition the data into distinct surfaces. As a consequence, boundaries defined by changes in 3D shape (curvature) cannot be detected, which constitutes a major difference in comparison to our method. In addition, each movie frame is segmented from scratch. In the case of surface segmentation, this can be rather costly. Furthermore, the temporal consistency of the segmentations will degrade with increasing clutter in the scene.

Seeding and subsequent growing of segments has previously been proposed in [3]. An erosion of the previous segmentation is applied to generate seeds for the new frame. For region growing, orthogonal distance of a point to a plane is checked. The method has been tested with planar surfaces only, because the initial segmentation is restricted to planar surfaces. In [16], seeds are obtained using a positive motion capture method followed by region growing.

In [17], upper body tracking of a human using a range sensor (Microsoft Kinect) is performed. The technique is limited to human beings only, as they use a prior model of the human body. The RGB-D data from Microsoft Kinect is also used in [18] for object detection and tracking. Planar surfaces are detected in the depth image and afterwards, surfaces connected to the planes are discovered by computing the differences in color and depth values. To detect the same object along time, a tracker that uses histogram of HSV values is employed. In [19], model based segmentation and tracking of rigid objects using cues from both color and depth data is proposed.

Joint segmentation and tracking has previously been performed mostly for color image sequences [20], [21], [22], [23], [24], [25]. Many methods for color-video segmentation usually perform independent segmentations of each frame and then try to match segments [21], [22], [23], [25]. This is problematic because segmentations have to be computed from scratch for every frame, which has consequences on both the computational cost and the temporal consistency of the results. For cluttered scenes, the partition of the segmentation tends to change from one frame to the next, and temporal coherence of the segmentations is prone to be impaired because of this effect.

In another work, segmentation and multi-object tracking were performed simultaneously using graphical models [24]. Observed and hidden variables of interest describing the appearance and the states of objects are jointly considered and used to formulate the objective as a Markov random field energy minimization problem. Different from our method, depth measurements do not enter the framework, and objects are defined based on their 2D appearance alone. Also, objects of interest are defined in the first frame and are then tracked along the sequence. While the method delivers convincing results, energy minimization is computationally expensive and efficient optimizations would have to be developed to make the approach more practical.

In [20], color images were segmented by finding the equilibrium states of a Potts model. Consistency of segmentations obtained along the movie and the tracking of segments were achieved through label transfer from one frame to the next using optic flow information. This way, the equilibrium states in the current frame could be encountered more rapidly. The resulting segments represent regions of uniform color and usually do not coincide with the object surfaces in a geometric sense, which we would desire for our system. The solutions

found in [20] cannot be easily adapted to our problem, because color segmentation and depth segmentation are inherently different problems. Surfaces cannot be defined based on local properties only, which increases the difficulty of the problem considerably.

B. Single Range-Image Segmentation

Many algorithms have been proposed for single range-image segmentation [11], [26], [27], [28], [29], [30], [31], [32], [33]. Most of these methods use detection of local depth discontinuities to segment the depth data. In [34], a single-range-image-segmentation method is proposed which segments each horizontal scan line into quadratic curves. Afterwards, the longest curve segment is chosen as a seed and used to perform grouping based on quadratic functions, which also allowed handling curved surfaces. The widely used gPb/UCM hierarchical segmentation for color images [35] is applied to both color and depth images and the segmentation results are linearly combined to generate a soft segmentation mask of the scene in [36]. There are also different learning approaches using labeled 3D scan data for segmentation [37], [38]. Collet et al. [39] made some high-level shape assumptions to discover different structures in the scenes.

Leonardis et al. [12] used superquadric models for segmenting a scene using a recover-and-select paradigm. One of the drawbacks of using superquadrics is the larger number of iterations ($\sim$ 15 in this case) needed to estimate the model parameters using Levenberg-Marquardt algorithm. In our approach we use rather simple quadratic surface models for segmenting and tracking range data. Furthermore, in our method, splitting and merging mechanisms are constantly competing, which allows earlier decisions to be revoked in subsequent iterations.

However, single range-image segmentation has different demands than range-video segmentation since it does not require that partition consistency is maintained across frames. Instead, the foremost goal of single range-image segmenters is to segment the image with high accuracy and to resolve very small structures. For video segmentation, the resolution of small structures is of less importance, since the main purpose is the stabilization of the segment labels along the video.

III. ALGORITHM

We describe a method for segmenting a movie of depth images into surface patches. A surface patch should contain a smooth surface, which can be planar or curved. Segments that describe the same surface should carry the same label along the sequence, corresponding to a tracking of the objects in the scene.

In our method, a coherent segmentation of a current frame is obtained by recycling the segmentation which has been computed for the preceding frame. Only the first frame of the sequence has to be segmented from scratch, simply because it has no predecessor. To obtain the segmentation of the first frame, we need to choose a starting point. Normally, two or four segments are taken initially to cluster the entire scene. Then the algorithm is iterated over the initial frame while updating the segmentation. Afterwards, we re-use the segmentation obtained for frame F^t to find the segmentation of frame F^{t+1}, and so on, where t denotes the frame number. The main components of the algorithm are presented in Fig. 2. The algorithm consists of four main steps:

A. Seeding/growing/splitting: Segment regions, labels, and the respective surface models of frame F^t are transferred to frame F^{t+1}. For each pixel, the depth predicted by the surface model of a segment is compared with the measured depth values found in the respective (preliminary) segment region. If the depth difference is below an adaptive threshold ψ, it is considered a seed for the segment (shrinking). The remaining points are unlabeled. Using seed points only, the surface models are re-estimated. The seeds are grown by assigning non-seed points in a competitive way (growing/splitting) to the neighboring surface that predicts the depth value of the point with the smallest error.

B. Relabeling: The assignment of new labels during the growing phase does not guarantee that the segments defined by the new labeling represent connected components. This is resolved by determining all connected components for a label and relabeling all components but the largest.

C. Merging: Neighboring segments are merged if they can be described approximately by the same surface model. Because only direct neighbors are considered, segments that have been split from a larger segment during the relabeling phase cannot be merged back with the same segment during a single iteration. This way, conflicts between splitting and merging do not arise, which facilitates convergence.

D. Iteration: Steps A-C are repeated until the number of segments stabilizes, indicating convergence. During video segmentation, the depth data is updated with the new movie frame at each iteration. For obtaining the initial segmentation, the procedure is applied repeatedly for the same frame. Note that while within a single iteration revokes are not allowed, both splitting and merging decisions can nevertheless be undone in the next iteration, which allows the method to adjust to changing data.

A pseudo code of the method including all mandatory steps of the algorithm is provided in Fig. 3. Parameter values used for segmentation are provided in Table III in the Appendix. Detailed calculation of variables such as ψ and specific implementation choices - taking into account special characteristics of the depth data used (e.g. holes) - are also provided in the Appendix.

Since merging is a particularly delicate/critical operation in the process, some more details provided here as follows.

During the merging step C1, we do not test for all possible merging combinations. Instead, during the growing step A3, the second-best label assignment for the unlabeled points is determined for each segment s and the most frequent label defines the interacting segment for s, representing the candidate for a potential merge. Then, segments are merged dependent on a threshold τ_1. New segments added during step B are treated separately and tested for merges using a threshold

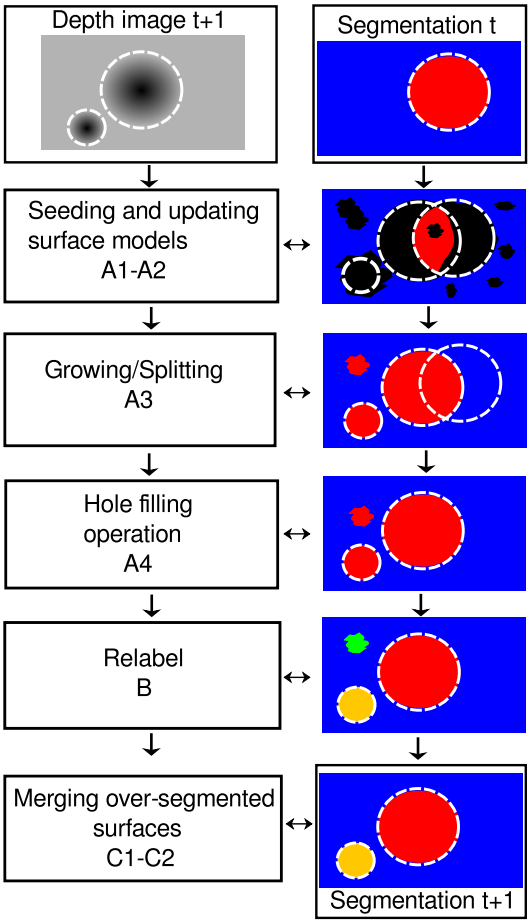

Fig. 2. Schematic illustrating the steps of our algorithm. Undefined areas are colored white, which usually correspond to the object contours. Black color represents the unlabeled regions in the segmentation.

Input: range image $x(u,v), y(u,v), z(u,v)$
assign an initial segment labeling $\rightarrow l(u,v)$
for all labels **do**
  fit a quadratic surface model
  if $|z_e - z| > \psi$ **then** (A1)
    unlabel the points (u,v) to generate seed
  end if
  update the quadratic surface model $\rightarrow f$, using seeds (A2)
end for
for all segments s_i **do**
  for all segments s_j **do**
    if s_j is neighbor with s_i **then**
      determine the model fitting error $\rightarrow \delta_{l_j}$, for unlabeled points in s_i, given the model f_j
    end if (A3)
  end for
  for all unlabeled points $(u,v) \in s_i$ **do**
    $l(u,v) \leftarrow \arg[\min_l(\{\delta_{l_1}(u,v), \delta_{l_2}(u,v), ..\})]$
    $r(u,v) \leftarrow \arg\,[\min_l(\{\delta_{l_1}(u,v), \delta_{l_2}(u,v), ..\} \setminus \{\delta_l(u,v)\})]$
  end for
  the interacting segment $I_i \leftarrow$ segment labeled mode(r)
end for (A)
label the undefined points in $l(u,v)$ (A4)
for all labels **do**
  determine connected components
  reassign a new label to each connected component except the largest one
  determine surface model parameters for the new segments
end for (B)
for all segments s_i **do**
  if model fitting error for $I_i < \tau_1$ **then**
    merge s_i with the interacting segment I_i (C1)
  end if
end for
for all newly added segments s_k **do**
  for all segments s_j **do**
    if s_k shares the largest boundary with s_j **then**
      if average distance between nearest neighbors $< \tau_2$ **then**
        merge the two segments (C2)
      end if
    end if
  end for
end for (C)

Fig. 3. Pseudo code describing a single iteration. Steps A1-C2 are in accordance with the illustration in Fig. 2. More details about the algorithm can be found in the Appendix.

τ_2 with the segment with which they share the largest boundary in step C2. In step B, new labels are assigned and at this particular moment this may represent a temporally incoherent event. This incoherence is needed in order to account for new objects that may enter or leave the scene. However if a new segment is falsely created it can be merged back during step C2 or in successive iterations during step C1.

IV. Image Acquisition

For in-house data, a Microsoft Kinect sensor along with the Kinect package of ROS (Robot Operating System) was used to acquire sequences of depth images $F^1, \ldots, F^t, F^{t+1}, \ldots F^{t+n}$ for different scenarios. Each frame contains a matrix of size $w \times h \times 3$, where w and h are the spatial dimensions of the image grid. Each point in the grid stores the values of the three coordinates (x, y, z) in the Euclidean space. The algorithm is implemented in Matlab.

V. Model Fitting

A quadratic surface model $f_j(x, y)$ of the form

$$z_e = f_j(x, y) = ax^2 + by^2 + cx + dy + e \quad , \tag{1}$$

with surface parameters a, b, c, d, and e is fitted to each segment s_j by performing a Levenberg-Marquardt minimization of the mean square distance of the measured depth points $z(u, v)$ from the estimated model depth $z_e(u, v) = f_j[x(u, v), y(u, v)]$. The chosen model type allows modeling of planar and curved surfaces, e.g., cylinders and spheres. The iterative solver (Levenberg-Marquardt minimization) enables us to use the solution obtained for the preceding time step to initialize the current minimization procedure. This way, fewer iterations ($\sim$ 4) are required to reach the minimum. For the initial time step we set the starting state of the iterative solver to zero.

VI. Performance Measure

We use the segmentation covering metric, as described in [40] to determine how closely the segmentation results match the ground truth segmentation. The ground truth (column (d) of Fig. 6, 7 and column (c) of Fig. 8, 12) was created by initializing with our segmentation result and afterwards correcting the wrong labels manually. The ground truth may for this reason contain a minor bias towards our method. For one frame, the segmentation covering metric of a machine segmentation S by a human segmentation S' is defined as

$$C(S' \rightarrow S) = \frac{1}{N} \sum_{R \in S} |R| \cdot \max_{R' \in S'} O\,(R, R') \quad , \tag{2}$$

where N is the total number of pixels in the image, $|R|$ the number of pixels in region R, and $O(R, R')$ is the overlap between the regions R and R' defined as

$$O(R, R') = \frac{|R \cap R'|}{|R \cup R'|}. \quad (3)$$

Furthermore, for comparison of single range-image segmentation with other methods, we use the evaluation framework of [11].

VII. Results

The algorithm was tested with several depth movies showing human and robot manipulations of objects. A video is provided as supplementary material. Additionally, the results which are acquired and used to generate all the figures and their corresponding datasets are available at http://www.iri.upc.edu/people/shusain/datasets_segmentation.html.

A. Segmentation of initial frame/convergence

The segmentation of the first frame is obtained by iterating the algorithm over a single range image. Figure 4 shows the convergence result for an indoor scene. The numbers represent the corresponding unique label for each segment. The color image (Fig. 4(a)) is only shown for illustration but not used in the procedure. We observe the formation of stable segments after a few updates (Fig. 4(e)). The convergence plot is shown in Fig. 4(f). It can be observed that after a certain number of iterations the covering metric values became stable. For the images tested, we arrived usually at a stable configuration within 10 iterations. In order to terminate the iterative process at a stable point, we needed to define a stopping criterion. We compute a leakage factor ℓ between consecutive iterations, i.e.,

$$\ell = \text{sgn}\,(N_{t+1} - N_t) \sum_{i=1}^{k} \left| n_i^{t+1} - n_i^t \right|, \quad (4)$$

where N is the number of segments, n_i is the number of pixels in segment s_i and $k = \max(N_t, N_{t+1})$. When the leakage factor does not change anymore, the system has reached a minimum and the process can be terminated.

In Fig. 4 it can be seen that even though only a very small portion of the ball (spherical surface) is visible, its points are still clustered correctly. The algorithm further successfully resolved the depth differences at the boundaries between the table and the objects.

Figure 5 shows the segmentation result obtained with our method for a scene captured with a Laser Range Sensor (LMS-Z210 by Riegl) of size 444×1440 pixels, by the pattern theory group of Brown university [41]. The segmentation was initialized in the same way as in Fig. 4, and 50 iterations were required. Despite the complex structure of the scene, our method successfully segmented it into meaningful object surfaces. Particularly compared to [8] we obtained less over-segmentation.

B. Depth video acquired using a structured-light camera

We report results on a set of sequences recorded with the Microsoft Kinect sensor. Firstly, we present results for a human hand rolling with its fingers a green ball forward and then

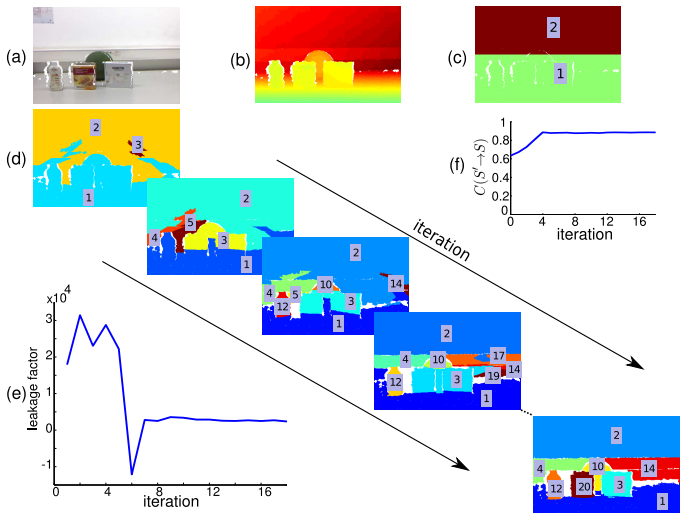

Fig. 4. Convergence of a single frame to a stable solution for a scene containing spherical and planar surfaces. (a) Color image from Kinect. (b) Depth image (Kinect). (c) Initialization with just two segments. (d) Segmentation results after iterating using our method. (e) Leakage factor after each iteration. The value of the leakage factor after convergence is ∼2366 in number of pixels. (f) Segmentation score, demonstrating convergence to a stable solution.

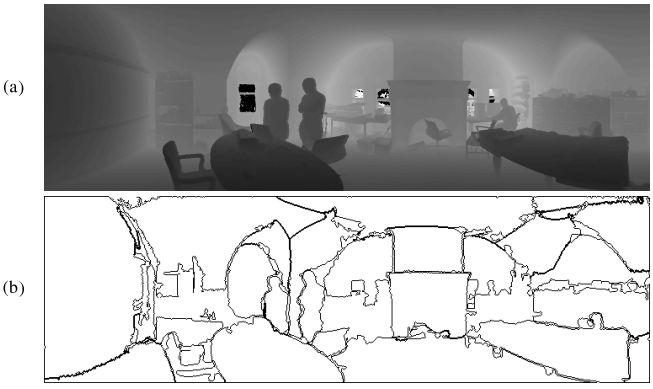

Fig. 5. Segmentation result for a scene from the Brown range dataset. (a) Range image. (b) Our segmentation result.

backwards (see Fig. 6). In Fig. 6(a) and (b), selected calibrated color and depth images acquired with the Kinect are shown, respectively. Areas for which no depth value could be acquired are marked white. In Fig. 6(c), our segmentation results are shown. Fig. 6(d) shows the ground-truth segmentation for comparison. The segments are color-coded, where each color corresponds to a unique segment label. The ball and the hand are correctly segmented and tracked along the image sequence, even though the hand is changing its shape and the ball is changing its size during the motion, so at least two different kinds of rigid transformations, i.e. translation and scaling, are taking place in this scene. Also the complete manipulator (hand plus arm) was not visible in the initial frame, but it is still segmented correctly later on.

We also show segmentation results for a movie where the robot arm grasps a paper roll and moves it to a new position (see Fig. 7). During the movement, objects in the background become occluded. Nevertheless, the sequence is correctly

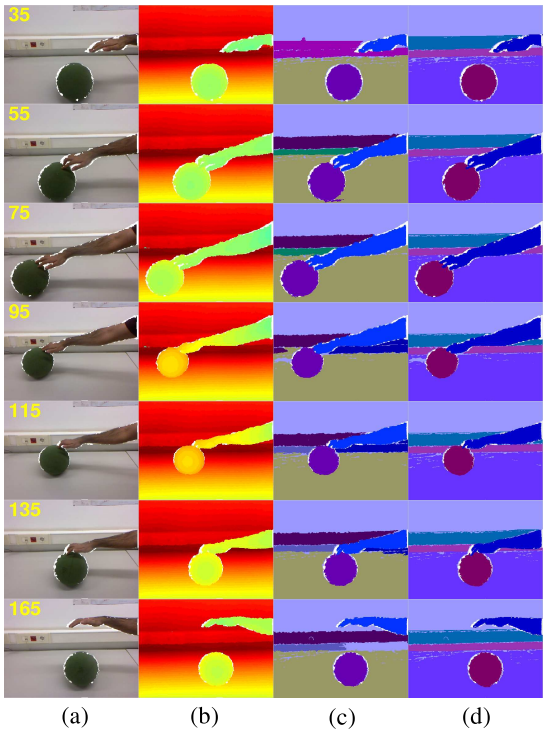

Fig. 6. Hand rolling a ball. Undefined areas are colored white. (a) Color images from Kinect. (b) Depth images (Kinect). (c) Video segmentation results using our method. (d) Human segmentation used as ground truth.

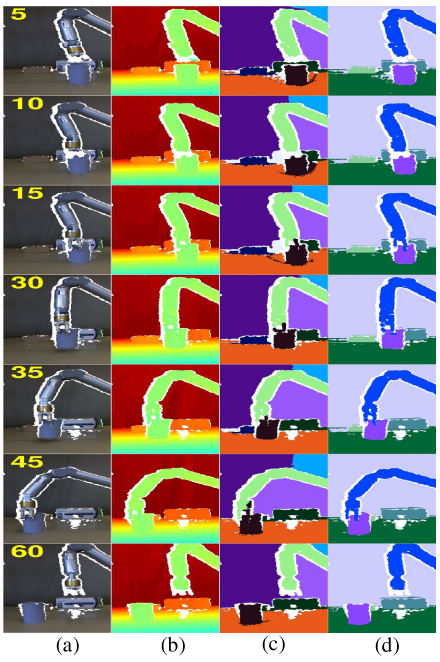

Fig. 7. WAM robotic arm grasping and displacing a paper roll. Undefined areas are colored white. (a) Color images from Kinect. (b) Depth images (Kinect). (c) Video segmentation results using our method. (d) Human segmentation used as ground truth.

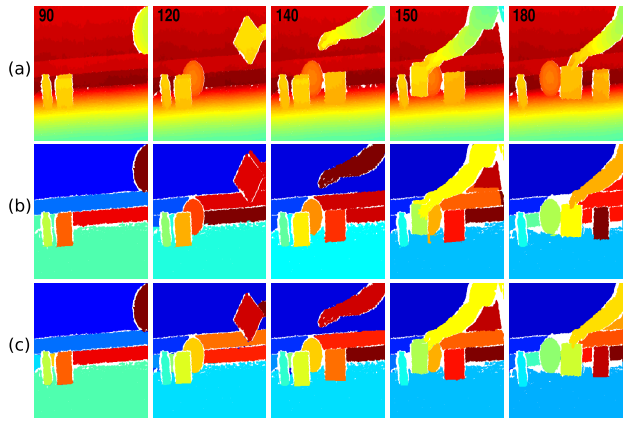

Fig. 8. (a) Depth images. (b) Machine segmentation. (c) Human segmentation.

segmented and both the robot arm and the paper roll are tracked along the movie. The background gets oversegmented because it got disconnected. The method can further handle the non-rigid transformation of the robot arm. Since our method does not rely on any kind of predefined shape models, it allows accommodating non-rigid behavior of objects. Also it can be observed that the robot arm is cylindrical and the tissue role as well, hence the depth values at the point of their contact become very similar, but due to the enforcement of constraint 2 (see Section I), the algorithm does not merge these two surfaces.

Figure 8 shows the depth images along with the corresponding machine and human segmentations of selected frames from a depth movie. The sensor pose was static, while different objects were manipulated. In this scenario, the human hand was entering and leaving the scene, displacing the objects rapidly, leaving little or no overlap of corresponding segments between consecutive frames. This naturally led to some temporal instabilities, so only the segmentation performance for single images can be evaluated, not the consistency along the movie. This movie will be used to compare the performance of the algorithm with those of different color-based video-segmentation algorithms in Section VIII-B.

Another scenario in which multiple segments are tracked simultaneously is shown in Fig. 9. Here, a plant is manually displaced on top of a cluttered table. It can be observed that as the plant is being displaced, multiple segments are tracked jointly through the scene. The advantage of using depth over color in real world is quite obvious in this scenario, since segmenting and tracking plant leaves using only color with no a priori information becomes impossible.

We also applied our approach to a typical sequence from the RGB-D Object Dataset [42]. The authors only provided the ground-truth mask of the object (the pitcher) placed on the turntable. Figure 10 shows selected frames of the segmented video sequence. It can be seen that the object on the turntable was successfully segmented while being rotated.

C. *Depth video acquired using a laser range finder*

The algorithm was tested on a scene of cluttered lab environment captured using odetics LADAR (Laser Detection

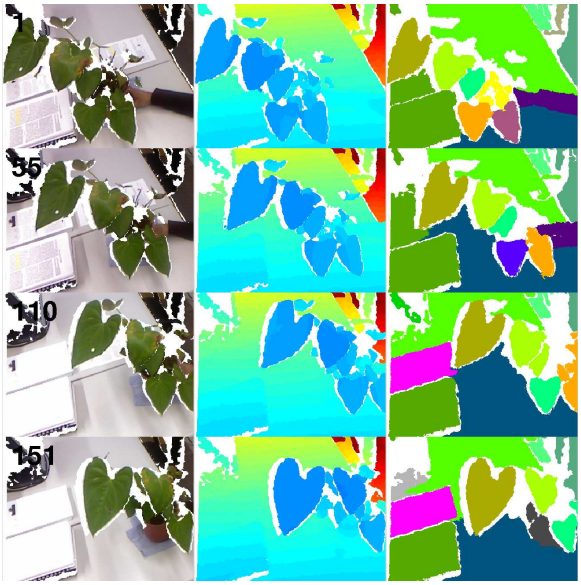

Fig. 9. Plant being displaced. Undefined areas are colored white. (a) Color images from Kinect. (b) Depth images (Kinect). (c) Video segmentation results using our method.

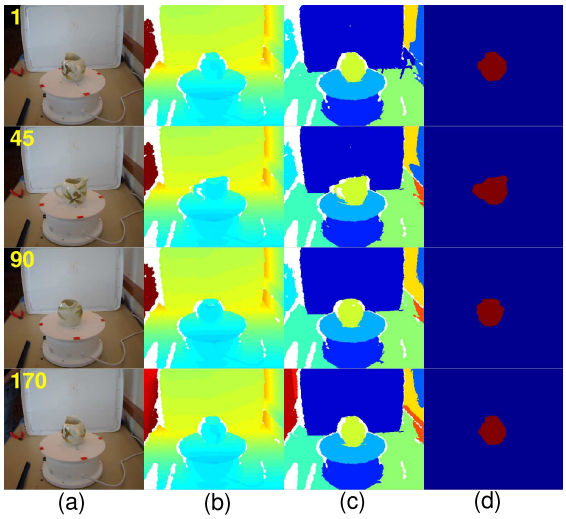

Fig. 10. Pitcher rotating on a turntable from the RGB-D Object Dataset [42]. Undefined areas are colored white. (a) Color images from Kinect. (b) Depth images (Kinect). (c) Video segmentation results using our method. (d) Ground truth of the segmented pitcher.

and Ranging) camera. The dataset was obtained from the USF range image database [43]. The robot moved quite rapidly in the scene leaving little overlap between consecutive frames. We had to iterate the algorithm three times over each frame to get consistent results. Fig. 11 shows selected typical intensity and depth images along with our segmentation results. The prism shaped surface and the ground floor kept the same label throughout the movie.

D. Depth video acquired using a time-of-flight camera

Figure 12 shows the depth images along with the corresponding machine and human segmentations of selected

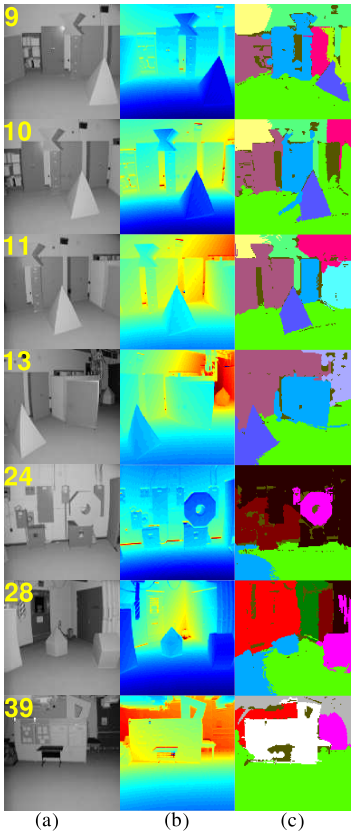

Fig. 11. Data acquired with a mobile robot in a cluttered lab environment, using odetics LADAR camera. From the USF range image database. (a) Intensity images. (b) Range images. (c) Video segmentation results using our method.

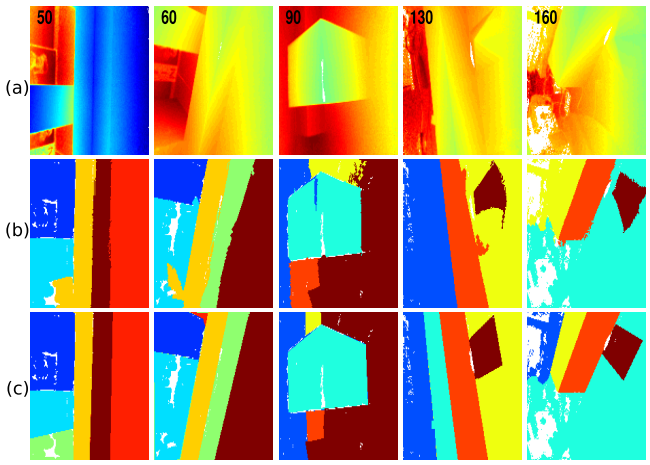

Fig. 12. (a) Depth images. (b) Machine segmentation. (c) Human segmentation.

frames from a depth movie. The data was obtained from the publicly available dataset of the Department of Robotics Systems at DLR [44]. In this scenario, the scene is static and the camera is rotating 2° at each time step. We iterated three times over each frame in order to get more consistent results along the movie.

VIII. Performance Evaluation and Comparison with other Methods

A. Comparative evaluation of single range image segmentation

We report segmentation performance of our method for the ABW and the Perceptron datasets [43] using the evaluation framework of [11] and compare it to various single-range-image segmenters [11], [26], [27], [28], [29], [30], [31], [32], [33], [7] (see Table I). For these datasets, we first used the provided training sequence (10 samples) to determine the optimal parameter values for ρ, τ_1 and τ_2. The datasets (ABW and Perceptron) contain only planar surfaces and the comparison has been done for planar segmentation only, hence we replace eq. 1 with the equation of a plane. We used a genetic algorithm to determine the parameter values which minimize the cost function,

$$cost = \frac{1}{M} \sum_{i=1}^{M} \left[1 - C_i(S' \rightarrow S)\right], \quad (5)$$

where M is the number of training samples. The remaining parameters were set as described in the Appendix. Afterwards, we evaluate the performance with the provided test sequence (30 samples).

For the ABW dataset, our method showed better results in terms of

1) correct detection than WSU, OU, PPU and UA,
2) over-segmentation than WSU, UB, UE, UBP, UBham, RCA and UFPR/OSU,
3) under-segmentation than PPU, UA and UMel,
4) missed regions than WSU, OU, PPU and UA and
5) noise than WSU, UB, OU, PPU, UA and RCA.

For the Perceptron dataset, our method showed better results in terms of

1) correct detection than WSU,
2) under-segmentation than WSU and UBP,
3) missed regions than WSU and
4) noise than WSU and UBham.

Our method has a performance that is overall comparable to other segmenters, but cannot outperform highly accurate segmenters such as the USF method. This is because the resolution of small image structures is of less importance in video segmentation, since fine details cannot be stabilized in the video anyway due to noise and changes in the temporally varying data.

However, the use of adaptive surface models, which are hooked on global structures rather than local ones such as jump edges or local changes in surface orientation, makes our method less sensitive to small changes in the depth data, and therefore more robust to noise. We demonstrate this by comparing it to a standard range segmenter from USF [11], for different levels of noise (see Fig. 13) added to the 30 images of the ABW sample test data using the segmentation coverage metric for evaluation. As can be seen in Fig. 13, for low levels of noise ($\sigma < 1.2$ mm) the USF segmenter performs better as it is able to delineate even very small image structures. However, for noise levels larger than 1.3 mm, the performance of the USF segmenter decreases rapidly, while our method only shows a slight decay, demonstrating its robustness.

TABLE I
Average results of 16 segmenters on ABW and 11 segmenters on perceptron test set at 80% compare tool tolerance.

Algorithm	ground truth	correctly detected	over-segmented	under-segmented	missed	noise
ABW 30 test images						
USF [11]	15.2	12.7 (83.3%)	0.2	0.1	2.1	1.2
WSU [11]	15.2	9.7 (63.8%)	0.5	0.2	4.5	2.2
UB [11]	15.2	12.8 (84.2%)	0.5	0.1	1.7	2.1
UE [11]	15.2	13.4 (88.1%)	0.4	0.2	1.1	0.8
OU [26]	15.2	9.8 (64.4%)	0.2	0.4	4.4	3.2
PPU [26]	15.2	6.8 (44.7%)	0.1	2.1	3.4	2.0
UA [26]	15.2	4.9 (32.2%)	0.3	2.2	3.6	3.2
EG [27]	15.2	13.5 (88.8%)	0.2	0.0	1.5	0.6
UBP [28]	15.2	13.0 (85.5%)	0.6	0.3	1.0	1.3
UBham [29]	15.2	13.4 (88.1%)	0.4	0.3	0.8	1.1
RCA [30]	15.2	13.0 (85.5%)	0.8	0.1	1.3	2.1
UFPR/OSU [31]	15.2	13.4 (88.1%)	0.4	0.1	1.2	1.7
GoD [32]	15.2	13.2 (86.8%)	0.3	0.2	1.1	1.8
UBonn [33]	15.2	11.1 (73.0%)	0.2	0.7	2.2	0.8
UMel [7]	15.2	12.8 (84.2%)	0.0	1.5	0.8	N.A
Our Approach	15.2	10.0 (66%)	0.3	0.8	3.0	1.9
Perceptron 30 test images						
USF	14.6	8.9 (60.9%)	0.4	0.0	5.3	3.6
WSU	14.6	5.9 (40.4%)	0.5	0.6	6.7	4.8
UB	14.6	9.6 (65.7%)	0.6	0.1	4.2	2.8
UE	14.6	10.0 (68.4%)	0.2	0.3	3.8	2.1
EG	14.6	10.5 (71.9%)	0.0	0.2	3.6	1.6
UBP	14.6	10.6 (72.6%)	0.2	0.6	2.6	2.0
UBham	14.6	11.2 (76.7%)	0.1	0.2	2.9	5.2
RCA	14.6	9.6 (65.8%)	0.7	0.2	3.7	3.6
UFPR/OSU	14.6	10.8 (74.0%)	0.1	0.1	3.4	2.0
GoD	14.6	10.7 (73.3%)	0.4	0.1	3.6	4.4
Our Approach	14.6	7.23 (49.5%)	1.9	0.3	4.7	4.4

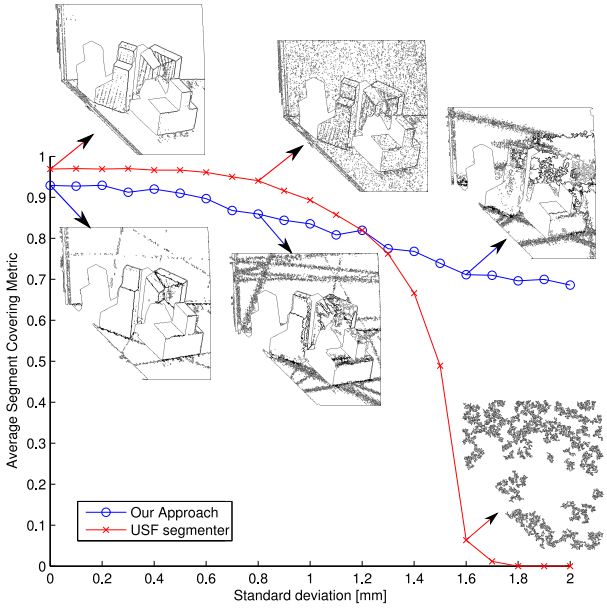

Fig. 13. Average segment covering metric for the 30 images of the ABW test dataset, along with example results of range image abw.test.8 for both our approach and the USF segmenter at different noise levels.

We further evaluated our segmentation approach on the NYU depth dataset [37] by calculating the F-measure using the boundary-detection evaluation criterion as proposed in [35]. It can be seen in Table II that we obtained a similar F-measure value as the approach of [36]. Figure 14 shows results for

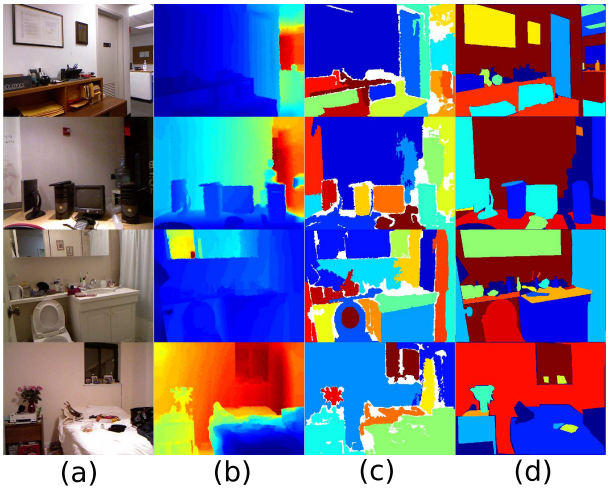

Fig. 14. Selected images from the NYU depth dataset [37]. (a) Color images. (b) Depth images. (c) Segmentation results using our method. (d) Human segmentation.

TABLE II
COMPARISON OF F-MEASURE EVALUATION FOR BOUNDARY DETECTION ON NYU DEPTH DATASET.

Algorithm	Range image segmentation from Table 1 in [36]	Our Approach
F-measure [35]	0.421	0.422

selected images of the dataset.

We additionally tested our approach on the more recent Cornell (77 samples) [45] and UBonn (30 samples) [33] datasets and obtained an average segmentation covering metric of around 60%. The ground truth labeling for the aforementioned datasets was supplied by the respective authors.

B. Comparative evaluation of video segmentation

Since there are no available benchmarks on depth video, we compare our method to two available color-video segmentation methods. Fig. 15 shows a plot of the segment covering metric for movies corresponding to Fig. 6 and Fig. 7. The metric is computed for our segmentation results and the ones obtained with a video segmentation algorithm based on color as described in [25]. Not surprisingly, our method outperforms [25] for those scenes where object surfaces contain multiple colors, and obtain similar results if the surfaces possess a unique color.

Figure 16 shows the metric for the movies shown in Figs. 8, 10 and 12 computed for frames chosen at fixed time intervals. For comparison Fig. 16(a) also shows the metric for two different video segmentation algorithms based on color as described in [46], [47] and [25]. It can be seen that our method outperforms the color-video segmenters. To obtain Fig. 16(b), we computed the covering metric for the pitcher only, since the ground truth was provided only for this object [42]. Overall, we observed that the proposed method allows segmenting videos while maintaining performance.

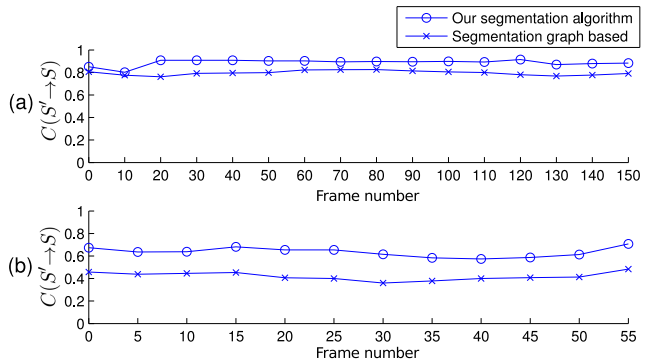

Fig. 15. Segmentation score with respect to human segmentation for our segmentation and for color segmentation using [25], corresponding to (a) Fig. 6 and (b) Fig. 7.

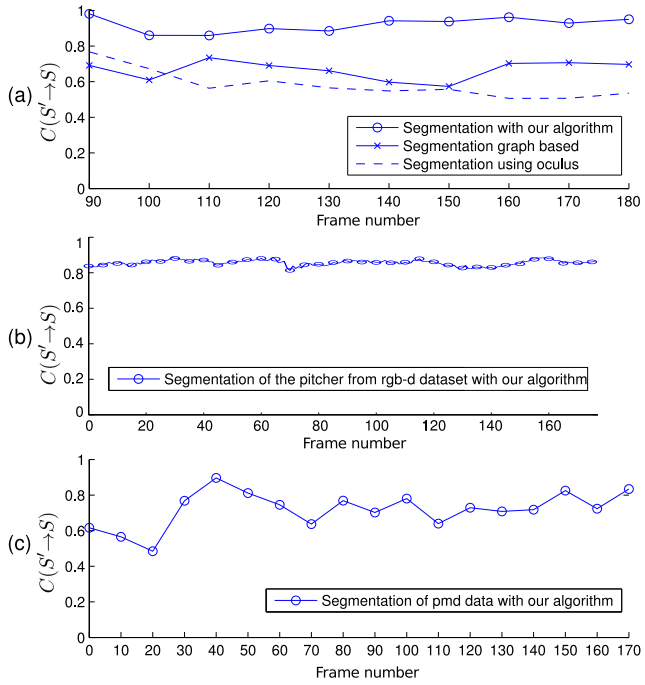

Fig. 16. Segmentation score of the depth movies corresponding to (a) Fig. 8 with our approach, graph based [25] and oculus [46], [47], (b) Fig. 10 and (c) Fig. 12 with our approach only.

IX. CONCLUSION AND FUTURE WORK

We presented a novel algorithm for joint segmentation and tracking of object surfaces, defined by their geometric shape. Segments obtained for the first frame are used to initialize the segmentation procedure of the next frame, and so on. We tested the algorithm for several movies acquired with different range sensors in cases where the sensor pose is static and others in which it is changing. Different scenarios were investigated including human and robot manipulations of objects. The algorithm allowed us to segment and track the main object surfaces in the scene, despite frequently occurring occlusions, limited resolution of the depth images, and shape changes of the hand and the robot gripper. However, some problems still remain. Occasionally, depth differences between surfaces are too small, resulting in assignment conflicts that cannot be resolved by the method as it is. For example, at

boundary regions, where a hand or robot gripper touches an object and assimilates its shape, pixels are often assigned to the wrong surface temporarily, since the local depth differences are insufficient for recovering the true boundary line. A motion model could be used to predict the most likely poses of the surfaces in the upcoming frames, for example by using a particle filter. Currently we are exploring such a mechanism to resolve assignment conflicts.

We also provide a mechanism for generating new segments in addition to the ones that have been determined in the first frame, which is important in case new objects are entering the scene.

We provided a quantitative evaluation of the method using human-annotated ground truth. Obtaining ground-truth for video is however a very tedious procedure and thus poses us limits. Since there is no available implementation of a similar algorithm performing joint segmentation and tracking in depth space, we compared our method to two standard color-video segmentation algorithms [46], [47] and [25]. We could show that our method outperformed color-video segmentation for the videos analyzed. However, this comparison may not be entirely fair, since we are using a different feature, i.e., depth, and not color.

We further evaluated the performance of our method with respect to single range image segmentation. While our method has overall a performance comparable to other range segmenters, it cannot outperform highly accurate techniques such as the USF method (see Table I). This is because video segmentation has different demands than single range-image segmentation. Our foremost goal is the stabilization of segments along the frames of the videos, for this reason, fine details have to be neglected. However, since our method uses global rather than local image criteria for segmentation, it is more robust to noise as compared, e.g., the USF method. The performance of the USF method rapidly deteriorates with the inclusion of Gaussian noise, as can be seen in Fig. 13, while the results obtained by our method are less affected by noise. We also measured our performance using the NYU depth dataset [37] and obtained a similar performance as a recent segmentation approach [36].

The video segmentation algorithm may be used to extract important contextual scene information. Fig. 17 shows a segment neighboring matrix for the entire depth movie corresponding to Fig. 12. The value of the matrix is one if the corresponding two segments are neighbors. The depth movie contained a total of 180 frames. It can be observed that segment 3 was neighbor with most segments from almost the beginning until the end of the movie. Hence it can be deduced that this segment surface is modeling a permanent surface of the scene, which can only be the ground floor as the camera was rotating and the entire scene was changing except the floor.

The computation speed depends on the frame size used. Currently, it varies from ~ 0.45 seconds to process a frame of size 143×175 pixels to ~ 1 second to process a frame of size 430×282 pixels in Matlab on Intel Xeon 3.3 GHz processor. Our implementation mostly involves arithmetic operations on matrices. With an efficient C/C++ implementation of our method using multi-threading libraries, real-time performance might be in reach, which is one of our next goals. Since color information complementary to depth is provided by the Kinect camera, optic flow obtained from color images could be used to get a better estimate of the initial segmentation for the upcoming frame [46]. This might help minimizing the segment overlapping requirement between consecutive frames, and we plan to investigate this aspect in the future.

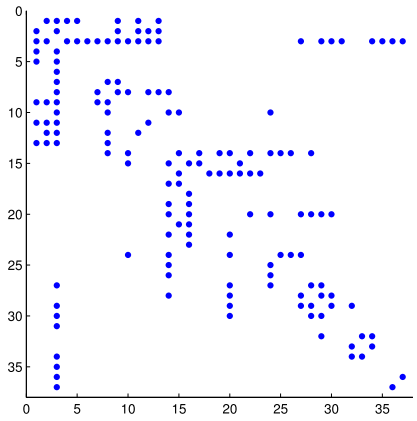

Fig. 17. Segment relations matrix corresponding to the results shown in Fig. 12.

To our knowledge, the method presented in this work is the first algorithm for joint segmentation and tracking of geometrically defined object surfaces, described by quadratic functions, using depth, not color. Because the segmentation is grounded on depth values, only the geometric shape of the surface matters, leading to invariance with respect to other object features such as color or texture. We plan to use our method in the context of a learning-from-demonstration task [48], [49]. The method may also be relevant for tackling problems frequently encountered in industry which require handling of geometric objects, as for example bin picking.

APPENDIX
ALGORITHMIC IMPLEMENTATION

The appendix headers are labeled according to the grouping A, B, C in the pseudo code in Fig. 3.

Depth segmentation of the first frame: An *initial segment labeling* $l^0(u,v)$ is defined as

$$l^0(u,v) = \begin{cases} 1 & \text{if } u \leq \lfloor h/2 \rfloor, \\ 2 & \text{otherwise} \quad , \end{cases} \tag{6}$$

where u and v are the indexes of the image grid and h is the frame height. The initial segment label matrix is multiplied with a binary mask B^0 defined as

$$B^0(u,v) = \begin{cases} 0 & \text{if } F^0(u,v) = 0, \\ 1 & \text{otherwise} \quad , \end{cases} \tag{7}$$

where the zeros (undefined points) correspond to the holes (undefined values) in F^0. Binary masks B^t for upcoming frames are defined in the same way. Steps A1-C2 (see below) are repeated for F^0 until a labeling l^t for the first frame is obtained.

In the following we assume that a labeling $l^t(u,v)$ has been already computed. Using this labeling we compute the current labeling of frame F^{t+1} as follows:

A1 Seeding: In order to update the segmentation grid, the first step should be to unlabel the points (u,v) that do not fit the surface. We achieve this by generating seeds. For each point (u,v) of frame F^{t+1}, we find the projected label $p = l^t(u,v)$ from the previous segmentation and define a seed labeling for F^{t+1} according to

$$m^{t+1}(u,v) = \begin{cases} p & \text{if } |z_e(u,v) - z(u,v)| < \psi_p, \\ 0 & \text{otherwise.} \end{cases} \quad (8)$$

with $z_e(u,v) = f_p^t[x(u,v), y(u,v)]$ and

$$\psi_p = \sum_{(u,v)\in s_p^{t+1}} |z_e(u,v) - z| \,/ (\rho n_p) \quad , \quad (9)$$

where s_p^{t+1} is the segment s_p^t projected onto the current frame F^{t+1}, n_p is the number of pixels in the area of s_p, and ρ is a constant. This defines a labeling $l^{t+1}(u,v) = p$ for all $(u,v) \in s_p^{t+1}$.

A2 Updating models: Now the surface model parameters need to be updated, so that they can model the current state of the segments. For each label j we obtain a surface model $f_j^{t+1}(x,y)$ for the current frame by applying the fitting procedure as explained in Section 5 to the seed of s_j, consisting of all the points (u,v) for which $m^{t+1}(u,v) = j$ holds.

A3 Growing/Splitting: Once we have updated the model parameters, we can determine the new labels of non-seed points. All points (u,v) with $m_j^{t+1}(u,v) = 0$ are grouped into connected components. For each connected component c_i, we search for seed points in the neighborhood R_1 of all boundary points (u,v), where R_1 is the radius of the disk used as a structuring element to dilate the connected components. If a seed point is found, its label is added to the list of potential labels $L_i = \{l_1, l_2, ..\}$ for c_i. For each label $q \in L_i$, we compute the distance

$$\delta_{l_q}(u,v) = |f_q^{t+1}[x(u,v), y(u,v)] - z(u,v)|. \quad (10)$$

For all $(u,v) \in c_i$, we can set

$$l^{t+1}(u,v) = \text{arg}[\text{min}_l(\{\delta_{l_1}(u,v), \delta_{l_2}(u,v), ..\})] \quad , \quad (11)$$

defining the labeling for the non-seed points. We also find the second-best fitting label for the group of non-seed points from each segment, i.e.,

$$r^{t+1}(u,v) = \text{arg}\left[\text{min}_l\left(\{\delta_{l_1}(u,v), \delta_{l_2}(u,v), ..\} \setminus \{\delta_{l^{t+1}}(u,v)\}\right)\right]. \quad (12)$$

We then define the *interacting segment* for segment s_j as

$$I_j^{t+1} = \text{mode}(r_j^{t+1}), \quad (13)$$

where the mode function determines the value that occurs most frequently.

A4 Hole filling operation: The undefined points that are present in the segmentation $l^{t+1}(u,v)$ due to the holes B^t in the previous frame F^t need to be filled in first in order to reduce oversegmentation due to addition of new segments in Step B.
First the edges of the depth image F_z^{t+1} are computed using the Canny operator and dilated using a disk of radius R_2 as a structuring element (SE), giving

$$\begin{aligned} E^{t+1}(u,v) &= canny\left\{F_z^{t+1}\right\} \quad , \\ E_d^{t+1}(u,v) &= E^{t+1}(u,v) \oplus SE. \end{aligned} \quad (14)$$

The dilated edges are used to disconnect undefined points in F^{t+1} which extend across different surfaces. Then each connected group of undefined points is assigned the label of the segment with which the largest boundary is shared. The current labeling is updated accordingly.

B Ensuring connectedness of each segment labeled $l^{t+1}(u,v)$ and assigning new labels: The assignment of new labels in Step A3 does not guarantee that the segments defined by the new labeling represent connected components (i.e., splitting). So we enforce Constraint 1 in this step. For each label j in $1,\ldots,k$, we find all points (u,v) for which $l^{t+1}(u,v) = j$. If the respective area is disconnected, we determine the connected components in the area. All connected components which do not define the largest component in the group are unlabeled. If an unlabeled connected component has a number of pixels greater than a constant κ it is assigned a new label and model parameters corresponding to its surface are estimated. This is how new surfaces are accommodated. Else it is assigned the label of the segment with which the largest boundary is shared. The current labeling $l^{t+1}(u,v)$ is updated accordingly.

C1 Label merging of *interacting segments*: The interacting segments I_j^{t+1} as determined in Step A3 are taken into account. We define a dissimilarity measure between the two segments s_i and s_j by estimating how well the surface model of segment s_i describes the depth data of segment s_j and vice versa. Let f_i be the surface model of segment label s_i, and f_j the surface model of segment label s_j. Then, we compute the fitting errors

$$\xi_{i/j} = \sum_{(x,y,z)\in s_i} |f_j(x,y) - z| / n_i \quad , \quad (15)$$

and

$$\xi_{j/i} = \sum_{(x,y,z)\in s_j} |f_i(x,y) - z| / n_j \quad , \quad (16)$$

where n_i and n_j are the number of pixels in segment s_i and s_j, respectively. If the sum of the two errors $\xi_{i/j} + \xi_{j/i}$ is less than a threshold τ_1 and Constraint 3 (see Section I) is fulfilled, we assume that both

TABLE III
LIST OF VARIABLES AND CONSTANTS

Variable	Definition
t	Time step
F	Frame
x, y, z	Coordinates in camera reference frame
(u, v)	Location indexes in image grid
h	Image grid height
B	Binary mask
s_i, s_j	Segments
l	Segment label
p	Projected segment label
n	Number of points
m	Segment seed
a, b, c, d, e	Surface model parameters
δ	Distance between measured and estimated depth
ξ	Mean model fitting error
c	Connected component
r	Second best label
I	Set of interacting segments
E	Edge image
E_d	Dilated edge image
ψ	Adaptive threshold for seeding
$\overline{\triangle}$	Average distance between nearest neighbors
Constant	**Value**
R_1	10 pixels
R_2	3 pixels (Kinect), 1 pixel (others)
ρ	1.7
κ	1000 pixels
τ_1	0.1 meters
τ_2	0.02 meters

segments model the same surface. Then the segments are merged and the surface model parameters are updated. After a segment is merged with another, it is no longer considered as interacting with other segments.

C2 Confirming new segments: In this step only the segments that were newly added in Step B are checked against the neighboring segment with which they share the largest boundary. Let s_i be the newly added segment and s_j be the segment with which it shares the largest boundary. We translate s_i and s_j such that their centroids are at the same position yielding s'_i and s'_j. Then we compute the average distance $\overline{\triangle}$ between nearest neighbors in s'_i and s'_j. If $\overline{\triangle} \leq \tau_2$, we assume that both segments belong to the same surface, so they are merged and the surface model parameters are updated.

D Steps A1-C2 are repeated for the next frame using $l^{t+1}(u, v)$ as initial labeling.

The choice of *initial segment labeling* is not fixed and could be initialized in any other configuration. Our choice is based on the fact that our focus is on modeling indoor environments and the background is usually divided in this manner (the front facing wall and the floor). This leads to an initial convergence in fewer iterations. In our experiments we needed to adjust only the control parameter R_2 according to the depth sensor used. The remaining parameters were kept constant.

Table III summarizes the definition of variables and constants used in our algorithm.

REFERENCES

[1] E. Parvizi and Q. Wu, "Multiple object tracking based on adaptive depth segmentation," in *Canadian Conf. on Comput. and Robot Vision*, 2008, pp. 273–277.

[2] S. Ghobadi, O. Loepprich, K. Hartmann, and O. Loffeld, "Hand segmentation using 2d/3d images," in *Proc. Image and Vision Computing New Zealand*, 2007, pp. 64–69.

[3] X. Jiang, S. Hofer, T. Stahs, I. Ahrns, and H. Bunke, "Extraction and tracking of surfaces in range image sequences," in *Int. Conf. on 3-D Digital Imaging and Modeling*, 1999, pp. 252–260.

[4] J. Han, R. Volz, and T. Mudge, "Range image segmentation and surface parameter extraction for 3-d object recognition of industrial parts," in *IEEE Int. Conf. on Robotics and Automation*, vol. 4, Mar. 1987, pp. 380–386.

[5] K. Pulli and M. Pietikainen, "Range image segmentation based on decomposition of surface normals," in *8th Scandinavian Conf. on Image Anal.*, 1993.

[6] G. Hegde and C. Ye, "A recursive planar feature extraction method for 3d range data segmentation," in *IEEE Int. Conf. on Sys., Man, and Cybern.*, Oct. 2011, pp. 3119–3124.

[7] A. Bab-Hadiashar and N. Gheissari, "Range image segmentation using surface selection criterion," *IEEE Trans. on Image Process.*, vol. 15, no. 7, pp. 2006–2018, July 2006.

[8] F. Han, Z. Tu, and S.-C. Zhu, "Range image segmentation by an effective jump-diffusion method," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 26, no. 9, pp. 1138–1153, sept. 2004.

[9] J. Min, M. Powell, and K. Bowyer, "Automated performance evaluation of range image segmentation algorithms," *IEEE Trans. on Syst., Man, and Cybern., Part B: Cybern.*, vol. 34, no. 1, pp. 263–271, Feb. 2004.

[10] M. Powell, K. Bowyer, X. Jiang, and H. Bunke, "Comparing curved-surface range image segmenters," in *Int. Conf. on Comput. Vision*, 1998, pp. 286–291.

[11] A. Hoover, G. Jean-Baptiste, X. Jiang, P. Flynn, H. Bunke, D. Goldgof, K. Bowyer, D. Eggert, A. Fitzgibbon, and R. Fisher, "An experimental comparison of range image segmentation algorithms," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 18, no. 7, pp. 673–689, Jul. 1996.

[12] A. Leonardis, A. Jaklic, and F. Solina, "Superquadrics for segmenting and modeling range data," *IEEE Trans. on Pattern Anal. and Machine Intell.*, vol. 19, no. 11, pp. 1289–1295, Nov. 1997.

[13] R. Lakaemper and L. Latecki, "Using extended em to segment planar structures in 3d," in *18th Int. Conf. on Pattern Recognition*, vol. 3, 2006, pp. 1077–1082.

[14] S.-M. Rhee, Y.-B. Lee, J. D. K. Kim, and T. Rhee, "Split and merge approach for detecting multiple planes in a depth image," in *19th IEEE Int. Conf. on Image Processing*, 2012, pp. 1213–1216.

[15] B. Dellen, F. Husain, and C. Torras, "Joint segmentation and tracking of object surfaces along human/robot manipulations," in *Int. Conf. on Comput. Vision Theory and Applicat.*, 2013, pp. 244–251.

[16] C. Wang and H. Liu, "Robust visual tracking based on adaptive depth-color-cue integration using range sensor," in *IEEE Conf. on Multisensor Fusion and Integration for Intell. Syst.*, 2012, pp. 330–335.

[17] A. Lopez-Mendez, M. Alcoverro, M. Pardas, and J. Casas, "Real-time upper body tracking with online initialization using a range sensor," in *IEEE Int. Conf. on Comput. Vision Workshops*, Nov. 2011, pp. 391–398.

[18] A. Pieropan, C. Ek, and H. Kjellstrom, "Functional object descriptors for human activity modeling," in *IEEE Conf. on Robotics and Automation*, 2013, pp. 1282–1289.

[19] K. Pauwels, L. Rubio, J. Diaz Alonso, and E. Ros, "Real-time model-based rigid object pose estimation and tracking combining dense and sparse visual cues," in *IEEE Conf. on Comput. Vision and Pattern Recognition*, 2013, pp. 2347–2354.

[20] E. E. Aksoy, A. Abramov, J. Dörr, K. Ning, B. Dellen, and F. Wörgötter, "Learning the semantics of object action relations by observation," *The Int. J. of Robotics Research*, vol. 30, no. 10, pp. 1229–1249, 2011.

[21] Y. Deng and B. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Tran. on Pattern Anal. and Mach. Intell.*, vol. 23, no. 8, pp. 800–810, Aug. 2001.

[22] I. Patras, E. Hendriks, and R. Lagendijk, "Video segmentation by map labeling of watershed segments," *IEEE Tran. on Pattern Anal. and Mach. Intell.*, vol. 23, no. 3, pp. 326–332, March 2001.

[23] D. Wang, "Unsupervised video segmentation based on watersheds and temporal tracking," *IEEE Trans. on Circuits and Syst. for Video Technol.*, vol. 8, no. 5, pp. 539–546, Sep. 1998.

[24] C. Wang, M. de La Gorce, and N. Paragios, "Segmentation, ordering and multi-object tracking using graphical models," in *IEEE 12th Int. Conf. on Comput. Vision*, Oct. 2009, pp. 747–754.

[25] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph-based video segmentation," in *IEEE Conf. on Comput. Vision and Pattern Recognition*, June 2010, pp. 2141–2148.

[26] X. Jiang, K. Bowyer, Y. Morioka, S. Hiura, K. Sato, S. Inokuchi, M. Bock, C. Guerra, R. Loke, and J. du Buf, "Some further results of experimental comparison of range image segmentation algorithms," in *Proc. of Int. Conf. on Pattern Recognition*, 2000, pp. 877 –881.
[27] X. Jiang, "An adaptive contour closure algorithm and its experimental evaluation," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 22, no. 11, pp. 1252–1265, Nov. 2000.
[28] P. Checchin, L. Trassoudaine, and J. Alizon, "Segmentation of range images into planar regions," in *Proc. of the Int. Conf. on Recent Advances in 3-D Digital Imaging and Modeling*, 1997.
[29] K. Koster and M. Spann, "Mir: an approach to robust clustering-application to range image segmentation," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 22, no. 5, pp. 430–444, may 2000.
[30] H. Frigui and R. Krishnapuram, "A robust competitive clustering algorithm with applications in computer vision," *IEEE Trans. on Pattern Anal. and Mach. Intell.*, vol. 21, no. 5, pp. 450–465, 1999.
[31] P. Gotardo, O. Bellon, K. Boyer, and L. Silva, "Range image segmentation into planar and quadric surfaces using an improved robust estimator and genetic algorithm," *IEEE Trans. on Sys., Man, and Cybern., Part B: Cybern.*, vol. 34, no. 6, pp. 2303–2316, Dec. 2004.
[32] B. Enjarini and A. Graser, "Planar segmentation from depth images using gradient of depth feature," in *IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Oct. 2012, pp. 4668–4674.
[33] B. Oehler, J. Stückler, J. Welle, D. Schulz, and S. Behnke, "Efficient multi-resolution plane segmentation of 3d point clouds," in *4th Int. Conf. on Intell. Robotics and Applicat.*, 2011, pp. 145–156.
[34] X. Jiang, H. Bunke, and U. Meier, "High-level feature based range image segmentation," *Image and Vision Computing*, vol. 18, no. 10, pp. 817 – 822, 2000.
[35] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. on Pattern Anal. and Machine Intell.*, vol. 33, no. 5, pp. 898–916, 2011.
[36] X. Ren, L. Bo, and D. Fox, "Rgb-(d) scene labeling: Features and algorithms," in *IEEE Conf. on Comput. Vision and Pattern Recognition*, 2012, pp. 2759–2766.
[37] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012, pp. 746–760.
[38] D. Anguelov, B. Taskarf, V. Chatalbashev, D. Koller, D. Gupta, G. Heitz, and A. Ng, "Discriminative learning of markov random fields for segmentation of 3d scan data," in *IEEE Conf. on Comput. Vision and Pattern Recognition*, vol. 2, 2005, pp. 169–176.
[39] A. Collet, S. S. Srinivasa, and M. Hebert, "Structure discovery in multi-modal data: A region-based approach." in *IEEE Conf. on Robotics and Automation*. IEEE, 2011, pp. 5695–5702.
[40] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "From contours to regions: An empirical evaluation," in *IEEE Conf. on Comput. Vision and Pattern Recognition*, June 2009, pp. 2294–2301.
[41] "Pattern theory group of Brown University," http://www.dam.brown.edu/ptg/brid/range/index.html.
[42] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *IEEE Conf. on Robotics and Automation*, 2011, pp. 1817–1824.
[43] "USF range image database," http://marathon.csee.usf.edu/range/DataBase.html.
[44] "Department of robotics systems at DLR," http://www.robotic.dlr.de/242.
[45] A. Anand, H. Koppula, T. Joachims, and A. Saxena, "Contextually guided semantic labeling and search for 3d point clouds," *The Int. J. of Robotics Research*, vol. 32, no. 1, pp. 19–34, 2013.
[46] A. Abramov, K. Pauwels, J. Papon, F. Worgotter, and B. Dellen, "Real-time segmentation of stereo videos on a portable system with a mobile gpu," *IEEE Trans. on Circuits and Syst. for Video Technol.*, vol. 22, no. 9, pp. 1292–1305, 2012.
[47] J. Papon, A. Abramov, E. Aksoy, and F. Worgotter, "A modular system architecture for online parallel vision pipelines," in *IEEE Workshop on Applicat. of Comput. Vision*, Jan. 2012, pp. 361–368.
[48] A. Agostini, C. Torras, and F. Wörgötter, "Integrating task planning and interactive learning for robots to work in human environments," in *Int. Joint Conf. on Artificial Intell.*, 2011, pp. 2386–2391.
[49] L. Rozo, P. Jiménez, and C. Torras, "A robot learning from demonstration framework to perform force-based manipulation tasks," *Intelligent Service Robotics*, vol. 6, no. 1, pp. 33–51, 2013.

Farzad Husain received a Master's Degree in Electrical Engineering with emphasis on Signal Processing, in 2011, from Blekinge Institute of Technology, Sweden. He is a Ph.D. student at the Institut de Robòtica i Informàtica Industrial, Barcelona, Spain, since 2011.

He was a student researcher at the Fraunhofer Institute for Manufacturing Engineering and Automation IPA, Stuttgart, Germany, from October, 2009 to July, 2010. His current research interests include semantic analysis of depth data, acquired using different range sensing devices, 3D scene recognition and understanding.

Babette Dellen received the Diplom degree in physics from the University of Cologne, Cologne, Germany, and the Ph.D. degree in physics from Washington University, St. Louis, in 2006, where she worked on computational models of the visual system of birds and mammals. She was a Post-Doctoral Researcher and Bernstein-Fellow with the Department of Computational Neuroscience, Bernstein Center for Computational Neuroscience, University of Göttingen, Göttingen, Germany, and with the Max-Planck-Institute for Dynamics and Self-Organization, Göttingen, from 2006 to 2010, where she worked mainly on computer vision. Dr. Dellen received the Ramon y Cajal Fellowship from the Spanish Ministry for Science and Innovation in 2009 and worked on computer-vision methods for robotic applications at the Institut de Robòtica i Informàtica Industrial, Barcelona, Spain, a joint research center of the Technical University of Catalonia, Catalonia, Spain, and the Spanish Council for Scientific Research. Since April 2014 she is Professor for Mathematics in the Life Sciences at the Hochschule Koblenz, Germany.

Carme Torras (http://www.iri.upc.edu/people/torras) is Research Professor at the Spanish Scientific Research Council (CSIC). She received M.Sc. degrees in Mathematics and Computer Science from the Universitat de Barcelona and the University of Massachusetts, Amherst, respectively, and a Ph.D. degree in Computer Science from the Technical University of Catalonia (UPC). Prof. Torras has published five books and about two hundred papers in the areas of robot kinematics, computer vision, machine learning and geometric reasoning. She has been local project leader of several European projects, among which the FP6 IP project "Perception, Action and COgnition through Learning of Object-Action Complexes" (PACO-PLUS), and the FP7 STREP projects "GARdeN-Ing with a Cognitive System" (GARNICS) and "Intelligent observation and execution of Actions and manipulations" (IntellAct). She was awarded the Narcís Monturiol Medal of the Generalitat de Catalunya in 2000, and she became ECCAI Fellow in 2007, member of Academia Europaea in 2010, and member of the Royal Academy of Sciences and Arts of Barcelona in 2013. Prof. Torras is Editor of the IEEE Transactions on Robotics.