EUROGRAPHICS 2021 / N. Mitra and I. Viola
(Guest Editors)

Volume 40 (2021), Number 2

Supplementary for RigidFusion: RGB-D Scene Reconstruction with Rigidly-moving Objects

Yu-Shiang Wong[1] Changjian Li[1] Matthias Nießner[2] Niloy J. Mitra[1,3]

[1]University College London [2]Technical University of Munich [3]Adobe Research

Abstract

In this supplementary, we introduce our evaluation metrics, visualize our dataset samples, summarize the system parameters as well as the detailed pseudocode for our system's sub-modules, and an example using CoFusion's real-world dataset.

1. Video

Please refer to the supplementary video to see a live recording on real-world data and comparison of results against other methods.

2. Benchmark Tasks and Evaluation Metrics

We provide two example scenes, from our synthetic benchmark dataset, in Figure 1 and several of high and low scoring results, using the employed evaluation metrics, in Figure 2 and Figure 3.

2.1. Reconstruction

A good reconstruction metric should handle the following issues: (a) different 3D representation, (b) lack of correspondences between ground truth surface and an output surface, (c) the model space may be different than canonical world space (dependent on the implementation).

In our benchmark, we handle different 3D representation by converting them into a point-based representation (i.e., point cloud) and conducting evaluation with the ground truth meshes' vertices. Specifically, for the volumetric based methods, we use the vertices of a reconstructed mesh as an output point set, and, for surfel-based methods, we use the centers of each surfel as an output point set. With this shared representation, we tackle the correspondences issues and report **Precision** and **Recall** by employing a bi-directional *Chamfer* distance. To estimate Recall, we calculate the squared distance between every point in the ground truth to the corresponding nearest point in the output point set. Then, we define a distance threshold (set to 3cm) to determine whether a ground truth point is successfully captured. To estimate Precision, we compare the output point set to the ground truth point set. The error threshold is also used to determine whether an output point is an outlier. Finally, we evaluate this metric in the camera space by transforming both the ground truth and the estimated model to the model's first detected frame so that we do not assume the model space is in the world space. In Figure 2, we show several output examples and the corresponding scores.

2.2. Tracking

To evaluate the quality of foreground detection and tracking, a benchmark should show the following information: (a) the percentage of good tracked frames in a video sequence, (b) the accuracy of the foreground detection.

We employ multiple objects tracking metric [BS08], including MOTA and MOTP, and enforce one-to-one mapping between the ground truth and the output by calculating MOTA on each trajectory independently and select the best one for evaluating both tracking and reconstruction. This penalizes the duplicate reconstruction or detection on the same object. Specifically, for calculating MOTA for the foreground tracking, the center of the ground truth mesh is used as a landmark. We transform the ground truth center using the estimated poses and compare the transform positions with the ground truth foreground positions. We introduce a distance threshold (5cm) to define whether the foreground object is tracked or not. The failed tracked frames are marked as BAD frames. If a method only outputs a partial trajectory due to insensitive object detection or tracking lost, those missing frames will be marked as MISS frames. Moreover, the precision (MOTP) of the good tracked frames, which excludes BAD and MISS frames, is reported using mean L2 norm distance over the matched positions. Note that we can use this metric to evaluate camera tracking as well. The MISS ratio will be zero, and the ground truth center is set to the origin so that MOTP is equal to the absolute trajectory error metric (ATE-RMSE). In Figure 3, we show several output examples and the corresponding tracking scores.

3. System Parameters and Comparison

In Table 1, we summarized the parameters used in RigidFusion as well as the system parameters of the state-of-the-art meth-

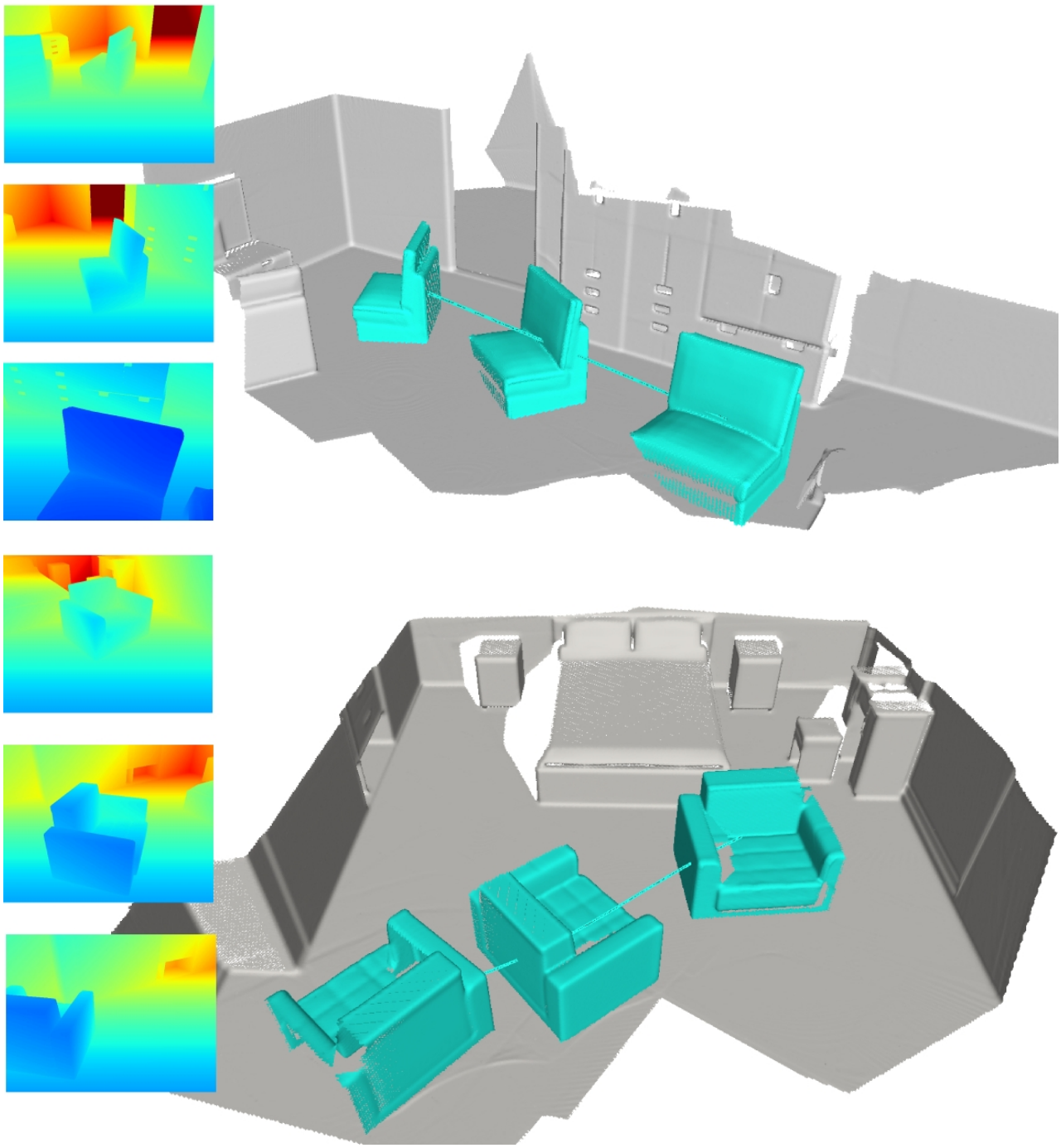
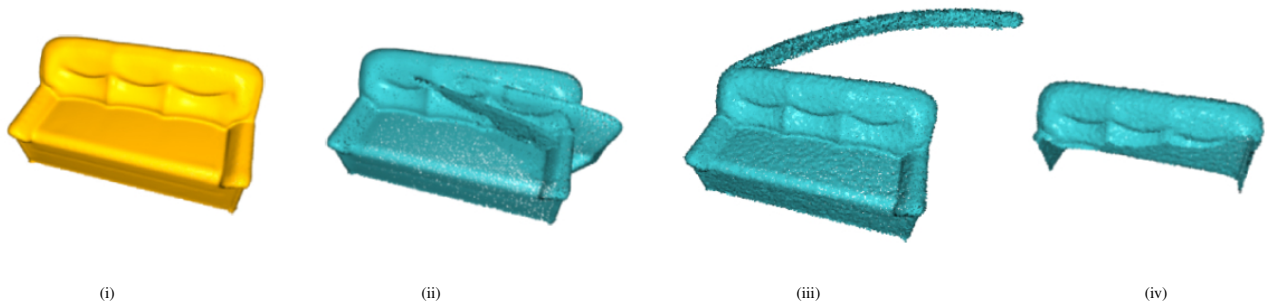


Figure 1: ***RigidFusion dataset.*** *Example scenes in RigidFusion dataset with one or more objects being rigidly moved, along the ground, across the scene. The dynamic scenes are in turn recorded from moving cameras. The full dataset comprises of 320 scenes with 92742 frames.*



(i) (ii) (iii) (iv)

Figure 2: ***Assessing reconstruction quality evaluation and types of errors.*** *(i) Ground truth reconstruction, (ii) A noisy reconstruction. In this example, precision and recall are 0.66 and 1.0, respectively, and the F1 score is 0.79. Low F1 is usually caused by tracking lost, which leads to misaligned surfaces. (iii) Another noisy reconstruction example. The precision and recall are 0.50 and 1.0, respectively, and the F1 score is 0.67. This happens when outliers are accumulated in the model over time due to the inaccurate foreground/background segmentation. (iv) A partial reconstruction example. The precision and recall are 1.0 and 0.54, and the F1 score is 0.70. This is usually caused by missed detection, which skips some views of the object.*

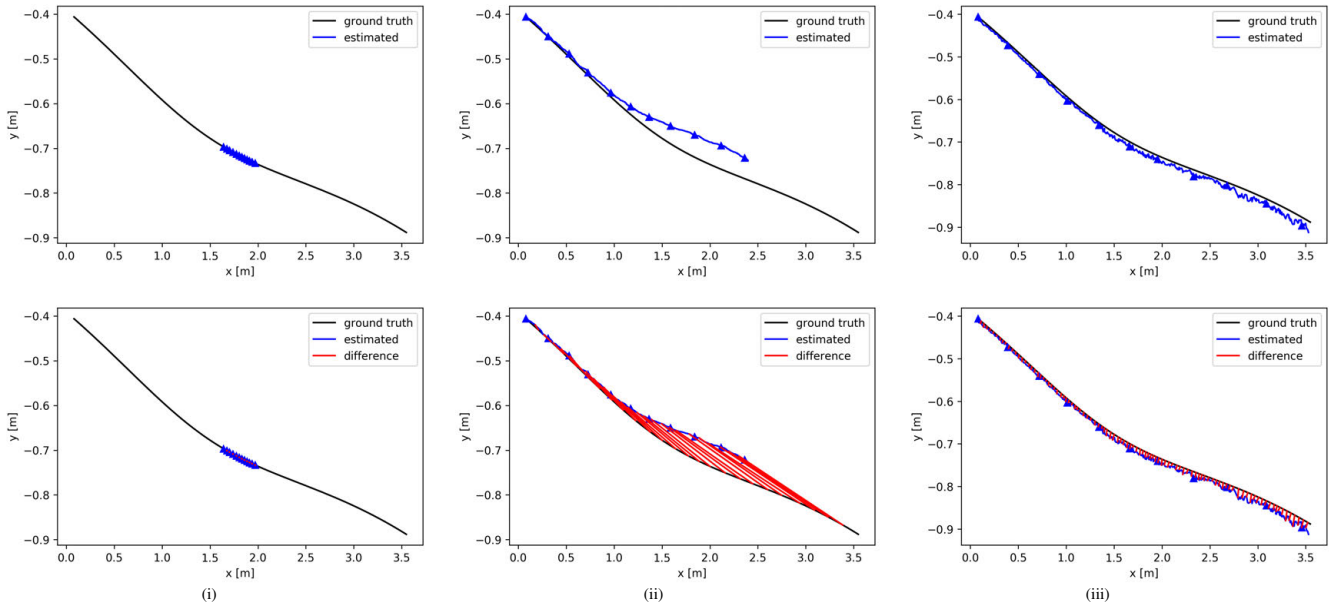


Figure 3: *Examples of tracking performance evaluation. Best viewed in color. The markers show sparse keyframes for the visualization purpose. (i) A delayed detection example. MOTA is 12%, and MISS ratio is 88%. MOTP is 0.1cm. (ii) A good detection but inaccurate tracking example. MOTA is 5%, and BAD ratio is 94%. MOTP is 2.8cm because it evaluates the precision of the good tracked frames (6% of the frames). (iii) An example with slight tracking drift. MOTA is 100%, MOTP is 1.6cm.*

ods [RA17, RBA18]. During the evaluation, we have tried our best to select comparison methods' parameters.

4. Algorithms

We provide detailed pseudocode for the sub-modules used on RigidFusion, including free-space aware TSDF fusion, segmentation by reconstruction, and re-optimization background reconstruction.

4.1. Free-space Aware TSDF Fusion

We maintain an byte array as a free-space count for the corresponding TSDF grid, named FreeGrid, using sparse voxel hashing. Each byte represents the frequency of the corresponding voxel locating in the positive truncation regions (free space). If a free-space count large than a pre-defined threshold C (set to ten), we reject the integration at the corresponding voxel. This approach prevents outliers from being integrated and alleviates the memory consumption problem of capturing multiple moving objects in a large scene.

For reconstructing a foreground object, an instance mask u is passed as input to indicate foreground pixels. For reconstructing background, this instance mask is always set to true because the foreground is unknown during background reconstruction. The pseudocode of the proposed TSDF fusion method is listed in the Algorithm 1.

4.2. Segmentation by Reconstruction

For each input frame F_j in the foreground module, we segment non-background regions by using the current background TSDF and the estimated camera pose at the time step j. The pseudocode is listed in the Algorithm 2.

4.3. Post-Processing: Re-optimization Background Reconstruction

We perform a post-refinement on background reconstruction after all input frames are processed. The reconstructed foreground models are jointly used to re-optimize camera trajectory $\left\{T_i^{(c)}\right\}$.

5. CoFusion's real-world example.

In addition, a qualitative example using CoFusion's real-world data is shown in Figure 4.

Algorithm 1: Free-space Aware TSDF Fusion

Input: a RGBD frame, TSDF, FreeGrid, instance mask u, camera frustum η

for each voxel $v \in \text{TSDF} \cup \eta$ **do**
- $c \leftarrow$ the free-space count of v in FreeGrid
- $C \leftarrow$ the free-space threshold
- $sdf \leftarrow$ the signed distance value of v
- $w \leftarrow$ the weight of v
- $v_{2d} \leftarrow$ projected image coordinates on the input frame
- $dist \leftarrow$ the signed distance from v to the back-projected depth pixel at v_{2d}
- `/* reject integration and denoising */`
- **if** $(c \geq C)$ **then**
  - **if** $w > 0$ **then**
    - remove the voxel v
  - **end**
  - continue
- **end**
- `/* integration */`
- isForeground $\leftarrow u(v_{2d})$
- **if** (isForeground AND $|dist| <$ truncation) **then**
  - `/* In truncation, do standard integration */`
  - update the w and sdf using running mean as in [CL96]
- **end**
- **else if** ($dist \geq$ truncation) **then**
  - `/* In free space */`
  - $c \leftarrow c + 1$
- **end**
- **else**
  - `/* In occluded space */`
  - continue
- **end**

end

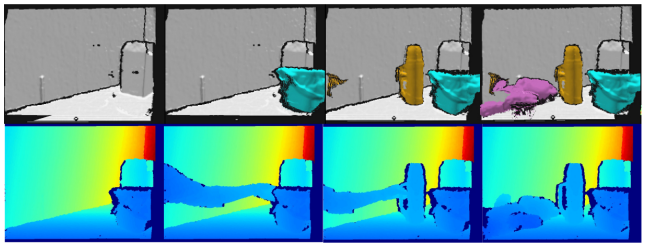


Figure 4: *Qualitative demonstration on CoFusion's real-world example. Top row: our results. Bottom row: input depth frames.*

References

[BS08] BERNARDIN K., STIEFELHAGEN R.: Evaluating multiple object tracking performance: The clear mot metrics. *EURASIP Journal on Image and Video Processing 2008* (01 2008). 1

[CL96] CURLESS B., LEVOY M.: A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (1996), ACM, pp. 303–312. 4

[RA17] RÜNZ M., AGAPITO L.: Co-fusion: Real-time segmentation, tracking and fusion of multiple objects. In *2017 IEEE International Conference on Robotics and Automation (ICRA)* (May 2017), pp. 4471–4478. 4, 5

Table 1: *Summary of System Parameters*

Our			CoFusion [RA17]			MaskFuion [RBA18]		
Parameters	**Values**	**Explanation**	**Parameters**	**Values**	**Explanation**	**Parameters**	**Values**	**Explanation**
Δ	60	the size of delay window	confO	0.01	initial surfel confidence threshold for objects	confO	0.01	initial surfel confidence threshold for objects
S	225*20	the minimum size of new object segments	confG	1.0	initial surfel confidence threshold for scene	confG	1.0	initial surfel confidence threshold for scene
segth	5	freespece count threshold	segMinNew	0.015	the minimum size of new object segments	segMinNew	0.015	the minimum size of new object segments
detth	1.00E-04	foreground de-activation threshold	segMaxNew	0.4	the maximum size of new object segments	segMaxNew	0.4	the maximum size of new object segments
cknum	0.5 $\cdot\Delta$	foreground de-activation check frame numbers	thNew	5.5	the threshold of initilizing a new model	thNew	5.5	the threshold of initilizing a new model
bvxsize	0.03	background TSDF voxel size	offset	22	offset between creating models	offset	22	offset between creating models
fvxsize	0.01	foreground TSDF voxel size	or	1	outlier rejection level	or	1	outlier rejection level
btrunc	10*voxel size	background truncation	crfRGB	10	the parameters for the conditional random field	filter_classes	-	filter instance segmentation by semantic labels
ftrunc	15*voxel size	foreground truncation	crfDepth	0.9	the parameters for the conditional random field	icpWeight	20	ICP weight
			crfPos	1.8	the parameters for the conditional random field	frameQ	30	the size of frame-queue
			crfAppearance	15	the parameters for the conditional random field			
			crfSmooth	4	the parameters for the conditional random field			
			icpWeight	10	ICP weight			

Algorithm 2: Segmentation by Reconstruction

Input: an input depth frame D, a human detection mask h, background model, camera pose $\mathbf{T_j^{(c)}}$
Output: instance mask u_j

$u_j \leftarrow$ Initialize a 2D mask with false values
$C \leftarrow$ the free-space threshold
set the human segments' depth to zero values in D using h
$d_{max} \leftarrow$ the maximum depth value
FreeGrid $\leftarrow$ background model's free-space grid
for each pixel at $(x,y) \in$ input depth D **do**
  `/* skip invalid depth */`
  if ($D(x,y)$ is 0 or $D(x,y) > d_{max}$) **then**
    continue
  end
  `/* back-project depth and transform to the world space */`
  $p \leftarrow \mathbf{T_j^{(c)}} \cdot$ backproject($D(x,y)$)
  `/* query the background model */`
  $c \leftarrow$ FreeGrid (p)
  `/* valid check */`
  if ($c > C$) **then**
    $u_j(x,y) \leftarrow$ true
  end
end

[RBA18] RÜNZ M., BUFFIER M., AGAPITO L.: Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (Oct 2018), pp. 10–20. 4, 5

Algorithm 3: Re-optimization Background Reconstruction

Input: All input RGBD frames, all human detection masks, all foreground models, all optimized foreground trajectories $\left\{T_j^{(k)}\right\}$
Output: re-optimized $\left\{T_i^{(c)}\right\}$, background TSDF

N $\leftarrow$ the number of input frames
K $\leftarrow$ the number of foreground models
allocate a new background TSDF
for $i \leftarrow 0$ to N step 1 **do**
  `/* pre-processing */`
  I $\leftarrow$ input RGB at the frame i
  $D \leftarrow$ input depth at the frame i
  h $\leftarrow$ human detection mask at the frame i
  set the human segments' depth to zero values in D using h
  $z \leftarrow$ a floating-point image, initialized to inf values
  `/* ray-casting foreground depth images */`
  for $k \leftarrow 0$ to K step 1 **do**
    if the instance k is active at the frame i **then**
      $start_k \leftarrow$ the detected frame index of the instance k
      $w_{th} \leftarrow \min(20, 0.2(i - start_k))$
      $d_k \leftarrow$ ray-casting a depth image from the instance k's model and filtering low weight voxels using the threshold w_{th}
      $visibleMask \leftarrow (d_k <$ z) AND $(d_k != 0)$
      $z(visibleMask) \leftarrow d_k$
    end
    `/* generate a foreground mask` Ψ `*/`
    $\Psi \leftarrow (|z - D| < 0.1)$ AND $(z! = 0)$
    set the foreground depth pixels to zero values in D using Ψ
    `/* background tracking and reconstruction */`
    re-optimize $\mathbf{T_i^{(c)}}$ using depth d and I
    update the background TSDF
  end
end