LivePose: Online 3D Reconstruction from Monocular Video with Dynamic Camera Poses

Noah Stier[1,2] Baptiste Angles[1] Liang Yang[1] Yajie Yan[1] Alex Colburn[1] Ming Chuang[1]

[1]Apple [2]University of California, Santa Barbara

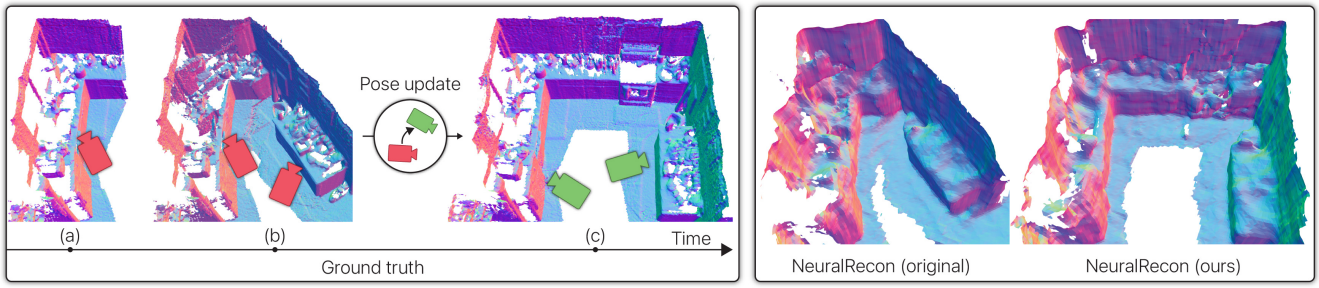

Figure 1. In online 3D reconstruction, poses obtained from a SLAM system (a, b) may be updated (c, red→green). Ignoring these updates leads to inconsistent geometry, while our pose update handling strategy produces accurate and globally consistent reconstructions.

Abstract

Dense 3D reconstruction from RGB images traditionally assumes static camera pose estimates. This assumption has endured, even as recent works have increasingly focused on real-time methods for mobile devices. However, the assumption of a fixed pose for each image does not hold for online execution: poses from real-time SLAM are dynamic and may be updated following events such as bundle adjustment and loop closure. This has been addressed in the RGB-D setting, by de-integrating past views and re-integrating them with updated poses, but it remains largely untreated in the RGB-only setting. We formalize this problem to define the new task of dense online reconstruction from dynamically-posed images. To support further research, we introduce a dataset called LivePose [1] containing the dynamic poses from a SLAM system running on ScanNet [6]. We select three recent reconstruction systems and apply a framework based on de-integration to adapt each one to the dynamic-pose setting. In addition, we propose a novel, non-linear de-integration module that learns to remove stale scene content. We show that responding to pose updates is critical for high-quality reconstruction, and that our de-integration framework is an effective solution.

[1]https://github.com/apple/ml-live-pose

1. Introduction

RGB-only reconstruction using a monocular camera has seen great progress using learned priors to address the difficulties associated with low-texture regions and the inherent ambiguity of image-based reconstruction. In particular, there has been strong interest in methods that are practical for real-time execution, a key component required for interactive applications on mobile devices. However, there is an additional requirement that has not been addressed in recent state-of-the-art reconstruction systems: a successful method must not only be real-time but also *online*.

Online operation implies that an algorithm must produce accurate incremental reconstructions at the time of image acquisition, using only past and present observations at each time point. This problem setting violates a key assumption made by existing works: **the availability of an accurate, fully-optimized pose estimate for each view**. Instead, in a real-world scanning scenario, a Simultaneous Localization and Mapping (SLAM) system suffers pose drift, resulting in a stream of *dynamic* pose estimates, where past poses are updated due to events such as pose graph optimization and loop closure. As detailed in Section 5, such pose updates from SLAM are ubiquitous in online scanning. It is critical for the reconstruction to stay in agreement with the SLAM system by respecting these updates, as we have illustrated in Figure 1.

Recent works on dense RGB-only reconstruction, however, have not addressed the dynamic nature of camera pose estimates in online applications [2, 8, 18, 19, 23, 24]. Although these efforts have made great progress in reconstruction quality, they have preserved the traditional problem formulation of statically-posed input images, providing no explicit mechanism for handling dynamic poses. In contrast, we acknowledge the presence of these updates, and we propose a solution by which existing RGB-only methods can incorporate pose update handling. We draw inspiration from BundleFusion [7], an RGB-D method that integrates new views into the scene with a linear update rule, such that past views can be de-integrated and re-integrated when an updated pose becomes available.

In this paper, we propose to use de-integration as a general framework for handling pose updates in online reconstruction from RGB images. We study three representative RGB-only reconstruction methods that have assumed static poses [8, 18, 24]. For each method, we apply the de-integration framework as detailed in Section 4 to address its limitations with respect to the online setting. In particular, we develop a novel, non-linear de-integration method based on deep learning to support online reconstruction for methods such as NeuralRecon [24] that use a learned, non-linear update rule. To validate this methodology and support future research, we release a novel and unique dataset called LivePose, featuring complete, dynamic pose sequences for ScanNet [6], generated using BundleFusion [7]. In our experiments (Section 6) we demonstrate the effectiveness of the de-integration approach, showing qualitative and quantitative improvement among three state-of-the-art systems with respect to key reconstruction metrics.

Contributions. Our main contributions are as follows:

- We introduce and formalize a new vision task, dense online 3D reconstruction from dynamically-posed RGB images, that more closely reflects the real-world setting for interactive applications on mobile devices.
- We release LivePose: the first publicly-available dataset of dynamic SLAM pose estimates, containing the full SLAM pose stream for all 1,613 scans in the ScanNet dataset.
- We develop novel training and evaluation protocols to support reconstruction with dynamic poses.
- We propose a novel recurrent de-integration module that learns to handle pose updates by removing stale scene content, enabling dynamic-pose handling for methods with learned, recurrent view integration.

2. Related Work

Here we discuss the rationale for selecting representative reconstruction systems to evaluate using de-integration for live pose handling. Selected methods are in bold.

2.1. Dense visual SLAM

Although SLAM often uses additional hardware such as depth sensors [22, 28], we instead focus on visual SLAM methods with no depth input, to support our use case of RGB-only reconstruction. Online mapping has been a long-standing goal in the visual SLAM literature, and a number of methods have evolved into mature, industry-standard software tools [9, 11, 17]. A subset of visual SLAM methods also construct dense scene representations [1, 9, 26], which could be used for downstream applications. These methods typically optimize jointly over the camera pose and dense depth for each input image. However, in our target application scenario, we assume that the poses are already estimated online, using an onboard SLAM system provided by the sensor platform. This is a reasonable assumption for smartphones and other mobile devices, and it has two main benefits: 1) it eliminates the additional degrees of freedom and computational complexity that are incurred by optimizing jointly over camera pose, and 2) it renders our solutions agnostic to the particular choice of SLAM system.

2.2. Depth-based reconstruction

Image-based 3D reconstruction is traditionally posed as per-pixel depth estimation, followed by fusion into scene space to form a unified 3D model [8, 12, 13, 15, 19, 20, 21, 27]. In particular, we select **DeepVideoMVS** [8] as a representative method in this group due to its high accuracy and fast execution time. We also highlight several notable limitations of reconstruction by depth estimation: 1) it provides no guarantee of agreement between different depth maps observing the same surfaces–this can result in incoherence in the fused model, which may require strong filtering or smoothing to resolve; 2) it cannot fill holes that arise in the reconstruction due to unobserved or occluded regions; 3) it does not offer a solution for end-to-end learning of 3D outputs for auxiliary tasks such as 3D semantic segmentation.

2.3. Implicit volumetric reconstruction

Recently, a number of methods have demonstrated a new paradigm in which a unified 3D model is estimated directly in scene space [2, 4, 10, 18, 23, 24], alleviating the above limitations of depth prediction. These algorithms back-project deep feature maps extracted from the input images, forming a world-aligned feature volume over the scene. Then, they apply a 3D CNN to predict a truncated signed distance function (TSDF), and the result can be visualized by extracting an explicit mesh using marching cubes [16]. This approach guarantees multi-view consistency of the reconstruction, and it provides a natural framework for hole-filling and surface completion using learned 3D shape and scene composition priors. Moreover, these methods can be trained end-to-end for arbitrary volumetric prediction tasks, as demonstrated by Atlas [18] for semantic segmentation.

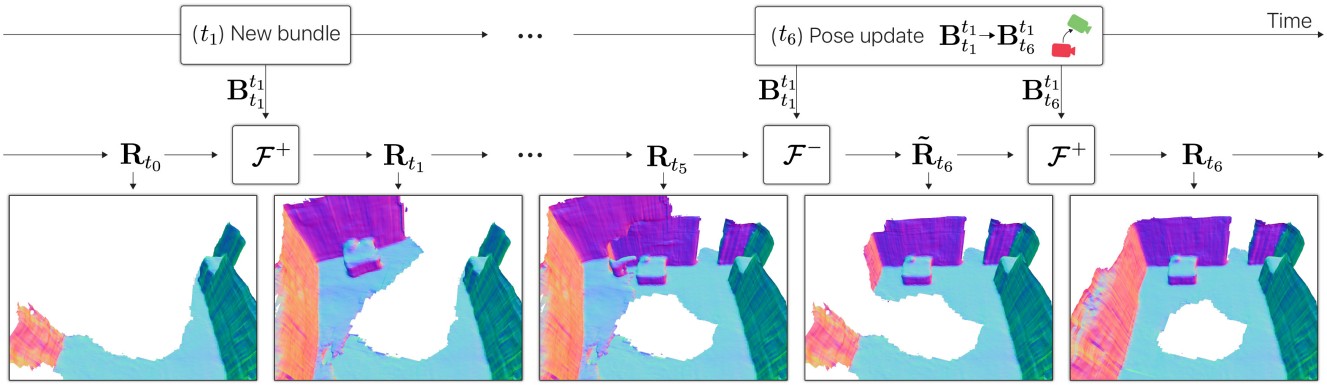

Figure 2. **Pose update handling by de-integration.** The integration module $\mathcal{F}^+$incorporates a new bundle $\mathbf{B}^{t_1}_{t_1}$ into the scene $\mathbf{R}_{t_0}$ as it is received at time t_1. Later at t_6 when the SLAM system issues a pose update, the de-integration module $\mathcal{F}^-$processes the same outdated bundle $\mathbf{B}^{t_1}_{t_1}$ to remove it from the scene. Then the updated bundle $\mathbf{B}^{t_1}_{t_6}$ with new pose estimates is re-integrated into the scene by $\mathcal{F}^+$.

Atlas [18] integrates the features from each view into the scene volume using a linear running average. This approach lends itself to a straightforward solution based on linear de- and re-integration. One drawback of this method is that the linear integration function may result in reduced reconstruction quality relative to subsequent methods with learned, non-linear view integration. **NeuralRecon** [24] uses a recurrent neural network (RNN) to integrate new views, yielding improved reconstruction quality relative to integration based on unweighted averaging. However, the non-linear nature of this integration function also prevents the direct de-integration of arbitrary past observations, leaving NeuralRecon with no clear strategy for handling pose updates. VoRTX [23] further demonstrates the quality improvements that can be attained with non-linear view integration, using transformers [25] to learn view selection and fusion. However, VoRTX provides no mechanism for producing incremental reconstructions as new views are added. The alternative is to reconstruct from scratch at each time point, which is prohibitively expensive for real-time applications.

TransformerFusion [2] also uses transformers to achieve increased reconstruction quality, and it introduces a view selection method that is compatible with incremental reconstruction. The proposed strategy is to maintain the top N image features per voxel, fusing them via transformer at each time step. However, storing N features per voxel, instead of a single, accumulated feature, results in an N-fold increase in the memory footprint of the feature volume, as well as added computational cost due to repeatedly reducing across views. With the proposed $N = 16$, these extra costs are significant, in particular when targeting mobile applications where power and memory are limited.

3. Problem statement

Given a sequence of images whose camera poses are estimated by an online SLAM system, we seek to compute an accurate online reconstruction of the scene. As new observations are processed, camera pose estimates are continually updated by the SLAM system. Formally, we denote the image captured at time t_1 as $\mathbf{I}^{t_1}$, and the estimate at time t_2 for the corresponding camera pose as $\mathbf{T}^{t_1}_{t_2} \in \mathbb{SE}(3)$.

In a dynamic system, poses can change dramatically. Therefore, it is unreasonable to define the ground truth for time t using the final pose estimates. Instead, the reconstruction system must respect the poses that are available at each point in time. To define the desired output, we assume the existence of a ground-truth depth image $\mathbf{D}^t$ paired with each color image $\mathbf{I}^t$. The ground truth geometry at time t is obtained by fusing all depths before and up to t:

$$\hat{\mathbf{R}}_t = \text{Fusion}(\{\mathbf{D}^{t_i}\}, \{\mathbf{T}^{t_i}_t\}), t_i \leq t, \tag{1}$$

where Fusion represents the TSDF fusion operation[5].

4. Method

4.1. Generalized pose update handling

Our solution to handle pose updates is a generalization of the strategy proposed by BundleFusion [7]. Without loss of generality, we consider a reconstruction system that processes frame bundles incrementally. Depending on the specific system, such bundles can contain multiple views or just a single one. We denote the bundle whose last image was captured at time t_1 and the corresponding pose estimates at time t_2 as $\mathbf{B}^{t_1}_{t_2}$:

$$\mathbf{B}^{t_1}_{t_2} = \{\mathbf{I}^{t_i}, \mathbf{T}^{t_i}_{t_2}\}_{t_i \in M^1}, \tag{2}$$

where M^1 is the set of image timestamps of the bundle created at time t_1.

When a new bundle is integrated, the reconstruction state produced by such a system at time t, denoted $\mathbf{R}_t$, is given by

$$\mathbf{R}_t = \mathcal{F}^+(\mathbf{R}_{t-1}, \mathbf{B}^t_t), \tag{3}$$

where $\mathcal{F}^+$ is the integration module which integrates a frame bundle to the previous reconstruction state $\mathbf{R}_{t-1}$.

When the pose estimates of a previously integrated bundle are updated by the SLAM system, we update the reconstruction state by first de-integrating the bundle's contribution and then integrating it back with the corrected pose. Therefore, we define our strategy to update the poses of a bundle created at time t_i and previously integrated at time t_j, with the new poses available at time t as

$$\mathbf{R}_t = \mathcal{F}^+(\mathcal{F}^-(\mathbf{R}_{t-1}, \mathbf{B}_{t_j}^{t_i}), \mathbf{B}_t^{t_i}), \tag{4}$$

where $\mathcal{F}^-$ refers to a de-integration operator which is responsible for un-doing the past integration of the bundle. Our strategy is illustrated in Figure 2.

These two operators should be implemented such that $\mathcal{F}^-$ is the inverse of $\mathcal{F}^+$:

$$\mathcal{F}^- = (\mathcal{F}^+)^{-1} \tag{5}$$

The details of how $\mathbf{R}_t$, $\mathcal{F}^+$ and $\mathcal{F}^-$ are implemented vary depending on the method that is applied. We discuss this topic for three classes of methods in the following subsections.

4.2. Linear TSDF de-integration

The first reconstruction methodology we investigate is multi-view stereo (MVS) depth prediction. For this class of methods, we adopt a TSDF voxel grid for $\mathbf{R}_t$, and we use TSDF fusion to define our operators. This gives the following definitions:

$$\mathcal{F}^+(\mathbf{R}, \mathbf{B}) = \text{Fusion}(\mathbf{R}, \text{MVS}(\mathbf{B})) \tag{6}$$

$$\mathcal{F}^-(\mathbf{R}, \mathbf{B}) = \text{Fusion}^{-1}(\mathbf{R}, \text{MVS}(\mathbf{B})), \tag{7}$$

where Fusion^{-1} is the TSDF fusion operator with a negative weight, and MVS is depth prediction network which leverages existing keyframes as auxiliary frames. These choices are preferred because the TSDF fusion operator is fast and $\mathcal{F}^-$ is the exact inverse of $\mathcal{F}^+$.

We choose DeepVideoMVS [8] as a representative depth prediction method.

4.3. Linear de-integration for neural features

We now turn to the direct, volumetric TSDF estimation approaches with linear integration. In this paradigm, deep image features are extracted from each view by a network $\mathcal{E}$, and then densely back-projected into a voxel grid as $\mathbf{R}_t$, integrating features into each voxel via running average. We observe that because this integration operation is linear, it supports de-integration similar to that of depth prediction methods, but operating on the neural features instead of directly on the TSDF values. Therefore, we define our operators using a running average:

$$\mathcal{F}^+(\mathbf{R}, \mathbf{B}) = \text{Avg}^+(\mathbf{R}, \text{BackProj}(\mathcal{E}(\mathbf{B}))) \tag{8}$$

$$\mathcal{F}^-(\mathbf{R}, \mathbf{B}) = \text{Avg}^-(\mathbf{R}, \text{BackProj}(\mathcal{E}(\mathbf{B}))) \tag{9}$$

We use Atlas [18] as a representative method in this category.

4.4. Learned de-integration for neural features

The third category of approach that we demonstrate is volumetric TSDF reconstruction with *non*-linear multi-view fusion, as exemplified by NeuralRecon [24]. In this case, the 2D deep features from $\mathcal{E}$ are integrated into the volume using a convolutional gated recurrent unit (GRU [3]) $\mathcal{G}^+$.

Recent trends show that there is a significant advantage to data-driven integration approaches [2, 23, 24], allowing the network to learn to attend to the most informative image inputs. However, the recurrent, non-linear feature integration makes it intractable to de-integrate arbitrary past views because of the order-dependence. We propose to address this by developing a novel, non-linear de-integration step, using deep learning to approximate the true de-integration function. Our solution is to use a second convolutional GRU to serve as the de-integration network $\mathcal{G}^-$.

$$\mathcal{F}^+(\mathbf{R}, \mathbf{B}) = \mathcal{G}^+(\mathbf{R}, \text{BackProj}(\mathcal{E}(\mathbf{B}))) \tag{10}$$

$$\mathcal{F}^-(\mathbf{R}, \mathbf{B}) = \mathcal{G}^-(\mathbf{R}, \text{BackProj}(\mathcal{E}(\mathbf{B}))) \tag{11}$$

The inputs to $\mathcal{G}^-$ are 1) the current scene feature volume and 2) the feature volume of the stale bundle to be de-integrated. We can interpret the desired function of this module in two steps. First, it must detect the effect that the target bundle has had on the current feature volume, and second, it must undo that effect.

We train our modified version of NeuralRecon on the LivePose dataset with the time-dependent ground truth $\hat{\mathbf{R}}_t$ defined in Section 3. This dynamic ground truth provides the necessary signal for learning the weights of the integration and de-integration networks, enabling the reconstruction system to remain in agreement with the SLAM system across updates.

Supervising the network with the data from LivePose has the added benefit that it exposes the network to a realistic distribution of online pose states, which may exhibit varying degrees of local and global consistency during online scanning. We show the impact of this effect in our ablation study (Table 2, row (b) vs. (c)).

4.5. Pose update filtering

We observe that the reconstruction systems investigated in this paper require a non-negligible amount of time to perform an update (see Table 1). As a result, it is not tractable to keep the reconstruction in perfect synchrony with a SLAM system that provides high-frequency pose updates for a large number of past views. We therefore propose that an intermediate layer should filter the pose stream from SLAM, emitting an update to the reconstruction system each time that the current optimized poses have drifted

Base method	Integration	Strategy	Acc↓	Comp↓	Chamfer↓	Prec↑	Recall↑	F-score↑	Latency (ms)	
									New	Update
Atlas [18]	Linear (features)	No updates	0.084	0.183	0.133	0.584	0.489	0.531	2,481	0
		Re-integration	0.082	0.188	0.135	0.602	0.500	0.545	2,492	2,363
		Ours	0.078	0.177	0.127	0.619	0.520	0.564	2,475	2,673
NeuralRecon [24]	Non-linear (features)	No updates	0.056	0.215	0.136	0.667	0.469	0.546	223	0
		Re-integration	0.053	0.204	0.129	0.688	0.492	0.568	223	233
		Ours	0.048	0.211	0.129	0.708	0.495	0.577	211	426
DeepVideoMVS [8]	Linear (TSDF)	No updates	0.082	0.096	0.089	0.536	0.508	0.521	1,081	0
		Re-integration	0.080	0.092	0.086	0.550	0.524	0.535	1,081	143
		Ours	0.071	0.086	0.079	0.569	0.543	0.554	1,081	287

Table 1. **Online 3D reconstruction metrics for ScanNet**, using the live pose data from LivePose. Comparison with state-of-the-art methods using the reconstruction metrics defined in Murez et al. [18], where the Chamfer distance is the mean of accuracy and completeness. Note that we report the latency per bundle (nine keyframes).

sufficiently far from their previously-integrated state. This strategy allows us to ensure a tractable pose update frequency regardless of the choice of SLAM implementation.

We propose to fill the role of the intermediate layer using the fragment generation system from NeuralRecon [24], which we extend in order to incorporate pose update dynamics. The first step in this system is to sub-sample a set of keyframes from the SLAM pose stream, adding a keyframe each time that the current camera pose differs from the previous keyframe by at least 10cm of translation or 15°of rotation. This supports real-time processing by reducing the total number of frames to be processed. Secondly, each time that K new keyframes are available, we collect them into a frame bundle, that is issued to the reconstruction system, thus increasing efficiency by batch processing. Following NeuralRecon, we set $K = 9$.

When the sum of the update distances over all views in bundle crosses a threshold d, we produce two additional bundles. The first is a de-integration bundle containing the now-stale poses that should be removed from the reconstruction. The second is a re-integration bundle containing the new optimized poses for the views in question. We set $d = 0.45$m, corresponding to an average update distance of 5cm over each of the nine keyframes.

4.6. Memory cost

For DeepVideoMVS, we require storing a depth image for each keyframe, which is tractable using our keyframe selection parameters. For Atlas and NeuralRecon, we require storing a feature map for each keyframe, although this space cost could be traded for latency by instead storing RGB images, and recomputing the feature maps when needed. The de-integration GRU in NeuralRecon incurs the memory cost of storing and running one extra neural network. However, it crucially does not require storing a second feature volume as it shares the scene state with the integration GRU.

5. LivePose dataset

In order to develop and evaluate our proposed de-integration strategies, we require access to the dynamic SLAM pose stream for a large number of RGB-D scans. However, to the best of our knowledge, there is no publicly available dataset that meets this need. The ScanNet dataset [6] contains 1,613 RGB-D scans of indoor scenes, and while it has played a significant role in recent 3D reconstruction research, its public release includes only the final, fully-optimized pose estimates for each scan.

Therefore, in order to utilize ScanNet in our dynamic-pose setting, we estimate the full, dynamic pose sequence for each scan. For consistency with the original ScanNet release, we use BundleFusion [7] as the SLAM system. In the resulting live pose streams, updates are highly prevalent:

- **98% of pose estimates receive at least one update**
- **25% of pose estimates are displaced by at least 0.5m**
- **5% of pose estimates are displaced by at least 2m**

See Figure 3 for more detailed statistics.

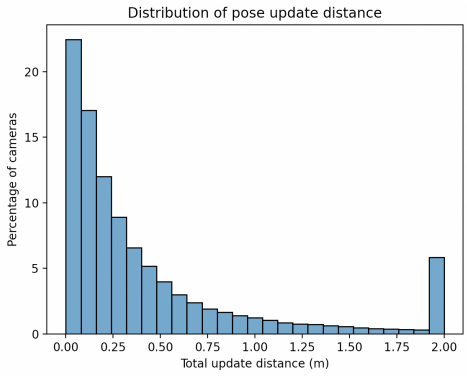

Figure 3. Histogram of the total update distance traveled by each camera as its pose estimate changes throughout the scan. Values are clipped to a maximum of 2m. Large pose updates are common with over 10% of poses being displaced by at least 1m.

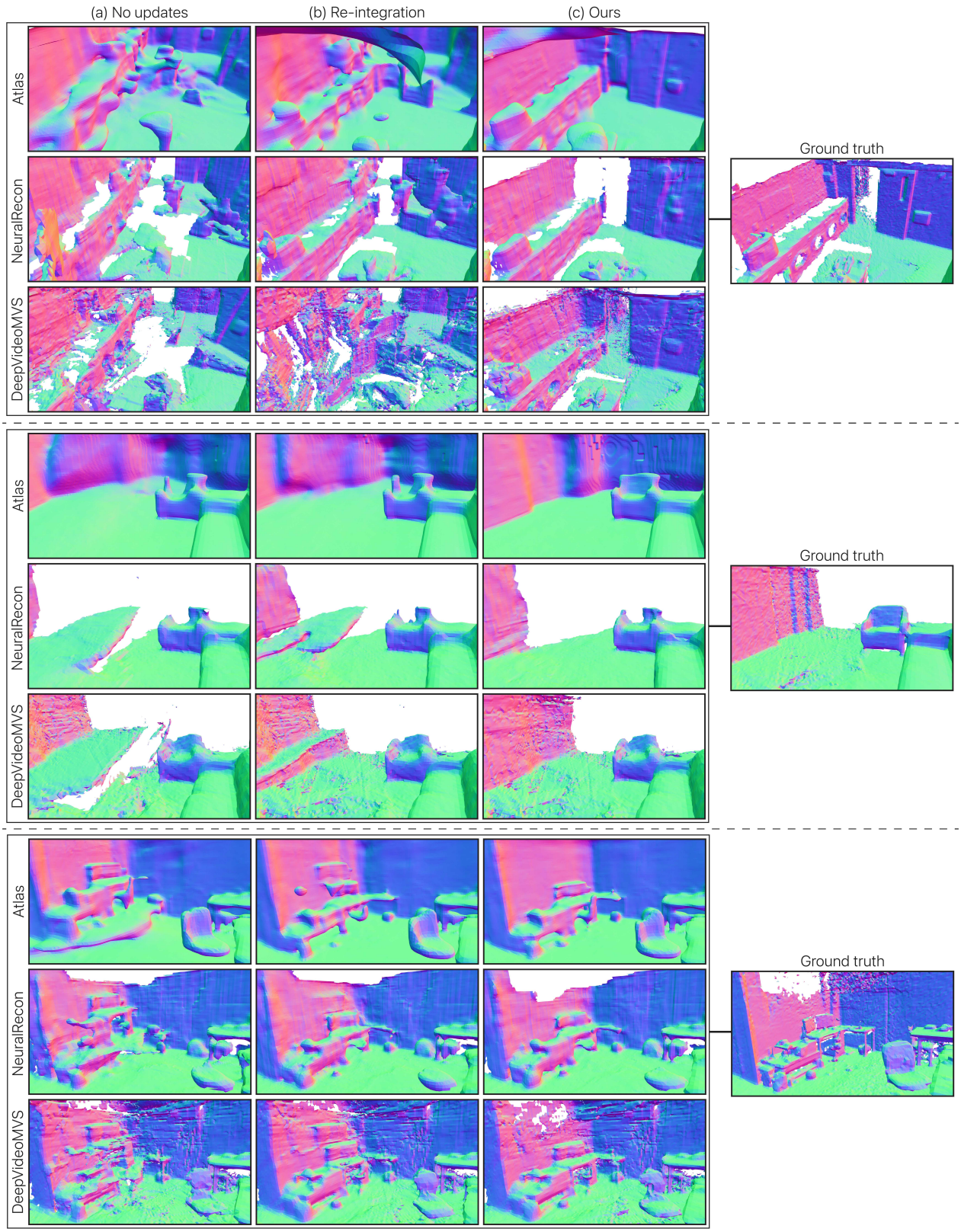

Figure 4. **Qualitative results with LivePose.** Camera poses from real-time SLAM can exhibit significant drift, leading to incoherent reconstruction states. (a) When pose updates are issued, failing to respond to them leads to significant discrepancies with respect to the ground truth reconstruction. (b) Re-integrating with updated poses can recover missing geometry, but (c) de-integration is necessary in order to remove stale scene content and achieve a coherent reconstruction.

We register our estimated trajectories to the official ground-truth poses following Horn [14], to facilitate comparison and to ensure that the gravitational axis is oriented consistently across scenes. To validate the accuracy of our pose data, we compare our BundleFusion pose estimates at the final time step of each scene to the official ground-truth poses, after registration. On average, we observe a 5cm mean absolute error (MAE) in camera position over all scenes. Upon qualitative investigation, we find that scenes with less than 15cm MAE typically do not have visually salient tracking errors, and our pose estimates fall below this threshold in 97% of scenes. In our dynamic pose data, we have valid pose estimates for 97.7% of total frames, compared to 97.9% in the official ground truth.

We release the dynamic pose sequences in the form of the publicly available LivePose dataset. This data is a key component of our training and evaluation, and we hope it will be a useful tool and point of reference for future work.

6. Experiments

We validate our solutions on ScanNet with dynamic pose estimates from LivePose, using the 100 scenes of the official test set to compute reconstruction metrics. In the tables, Best and Second -best are highlighted.

6.1. Baselines

For Atlas and DeepVideoMVS, we use the ScanNet-trained weights provided by the original authors in all experiments. For NeuralRecon, we re-train to incorporate our learned de-integration module, and for consistency we therefore use our trained weights for the NeuralRecon baselines. For DeepVideoMVS, we relax the pose update strategy by computing the depth for each view only once, and de-/re-integrating that same depth when the poses are updated. This reduces the update time significantly, and we observe no loss in accuracy due to high internal pose consistency within bundles across updates.

No updates To demonstrate the impact of failing to respond to pose updates, we evaluate each method without any modifications for handling dynamic pose. In this setting, the input sequence consists of each new bundle at the time that it first becomes available, with no subsequent updates. However, the ground-truth sequence does receive updates, and this allows us to quantify the discrepancy created between the SLAM system and the reconstruction system.

Re-integration only As a naive baseline for handling pose updates, we perform re-integration at each update without first performing de-integration. Stale scene content thus lingers in the reconstruction, and the results allow us to directly observe the contribution of de-integration.

6.2. Evaluating with dynamic poses

Previous works have not used dynamic poses for evaluation. In addition, they have not evaluated the quality of the incremental reconstructions at intermediate time steps. One challenge in performing this evaluation is that the official ScanNet data only contains ground-truth reconstructions for the final time step of each sequence. A second challenge is that in order to compare across methods, each one must produce reconstructions at the same set of fixed time points.

We address these challenges by first generating and publishing the dynamic pose sequences of LivePose. We then define a set of common, incremental reconstruction checkpoints: we set a checkpoint for every time step at which a bundle is integrated. This naturally produces a variable-rate series of checkpoints, more frequently sampling the reconstruction during periods of rapid updates. We generate a ground-truth mesh for each checkpoint by online TSDF fusion and marching cubes [16], at a resolution of 4cm.

6.3. Pose update handling by de-integration

6.3.1 3D reconstruction evaluation

Figure 4 shows qualitative results, using time points chosen before and after a series of pose updates. These results illustrate the significant incoherence that can result from pose drift, as well as the ability of our de-integration solutions to remain in agreement with the SLAM system as it recovers. We also observe that re-integration alone, without de-integration, is not enough to remove the stale scene content.

As shown in Table 1, de-integration leads to the best outcome on all metrics. This supports our claim that direct de-integration is a practical, effective framework for handling pose updates, and we show how it can be generalized across multiple scene representations and integration strategies.

6.3.2 Online reconstruction efficiency

Timing results are computed as an average over the ScanNet test set on an NVIDIA V100 GPU (Table 1). Atlas averages 2,673ms to process an update event for a bundle, comparable to its time for new bundle integration. In the case of NeuralRecon, handling an update requires running the 3D CNN twice, thus doubling the execution time for updates relative to new bundles. However, each update still requires less than 500ms, which is permissible in many interactive applications. For DeepVideoMVS, the update time is much faster than the new bundle integration time, primarily because we opt not to re-compute depth for each update; the timing breakdown for a processing a new bundle (1,081ms total) is on average 104ms for depth estimation and 16ms for TSDF fusion, for each of the nine frames.

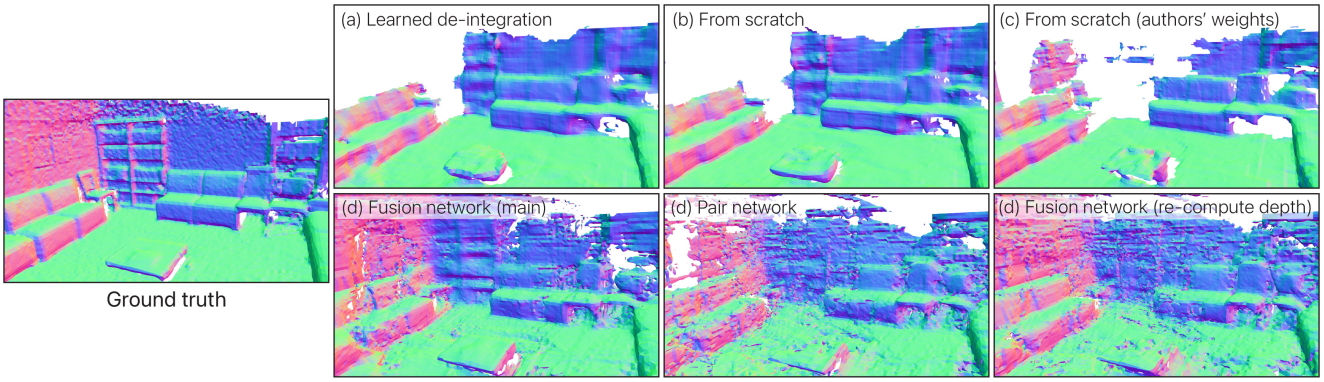

Figure 5. **Visual results for ablation study (Table 2).** For NeuralRecon, our solution (a) approaches the quality of our upper bound based on reconstructing from scratch (b). The improvement from (c) to (b) shows the benefit of live-pose training. For DeepVideoMVS, (d, e) show results from the fusion vs. pair network, and (f) shows the effect of recomputing depth upon re-integration. Details in Section 6.4.

Method	F-score↑	Latency (ms)	
		New	Update
NeuralRecon [24]			
(a) Learned de-integration (ours)	0.577	211	426
(b) Reconstruct from scratch	0.581	233	5,487
(c) Reconstruct from scratch (authors' weights)	0.544	204	4,724
DeepVideoMVS [8] with our de-integration			
(d) Fusion network	0.554	1,031	287
(e) Pair network	0.515	919	284
(f) Fusion network (re-compute depth)	0.545	1,031	1,233

Table 2. **Ablation study.** For NeuralRecon, we compare our learned de-integration (a) with an upper bound obtained by reconstruction from scratch for each update (b). In (c) we repeat (b) using the model weights provided by the original authors, showing the positive impact of our training protocol with live ground truth TSDF. For DeepVideoMVS, we show the effects of using the pair network (e), and re-computing depth before re-integration (f). See Section 6.4 for details and discussion.

6.4. Ablation studies

We further characterize our results using a series of ablation studies, with metrics reported in Table 2 and qualitative results shown in Figure 5.

Are we close to optimal de-integration? In Table 2 (b-c) we simulate perfect non-linear de-integration performance by starting the reconstruction from scratch each time an update is received. This strategy requires over 6s to process an update on average. While it is not a practical solution, it provides a useful upper bound on the accuracy of the de-integration module. Our ablation results are illustrated in Table 2, where (a) represents our novel solution for learned, non-linear de-integration and (b) is the perfect de-integration. In (c) we repeat the perfect de-integration experiment using the NeuralRecon weights provided by the original authors, instead of our re-trained weights. The gain in F-score from (c) to (a) indicates the importance of training with dynamic poses, which helps the network to perform accurate reconstruction when the pose estimates have not yet been fully optimized. These results show that the learned de-integration achieves the fastest processing rate while maintaining comparable reconstruction quality.

Should we re-compute depth? For DeepVideoMVS, updating the pose involves more than just integrating and de-integrating the per-view geometry. It also requires a re-evaluation of the depth computation due to potential changes in inter-frame relative poses. We perform experiments with vs without re-computing depth upon updates, and results are presented in the lower part of Table 2. Row (d) shows our main result for DeepVideoMVS with de-integration. In (e) we use the pair network instead of the fusion network, showing a slight reduction in F-score, consistent with our findings. In (f) we additionally re-compute the depth for each reference view before each re-integration. In theory, this could be necessary in the case of significant relative pose updates within the bundle, but in practice, we find that the results are similar, indicating high internal pose consistency across updates.

7. Conclusions and future work

We have presented a new problem formulation for online monocular 3D reconstruction, in which a dense reconstruction system must account for pose update events such as loop closures issued by the SLAM system, and we have demonstrated the importance of handling these updates to create accurate reconstructions. To address this problem, we have developed and validated solutions based on de-integration for three representative state-of-the-art methods, and we have introduced a novel de-integration module that enables pose update handling for methods using non-linear view integration. We have prepared a unique and novel dataset of online SLAM pose estimates, which we hope will facilitate future research in this area.

References

[1] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2560–2568, 2018. 2

[2] Aljaž Božič, Pablo Palafox, Justus Thies, Angela Dai, and Matthias Nießner. Transformerfusion: Monocular rgb scene reconstruction using transformers. *Proc. Neural Information Processing Systems (NeurIPS)*, 2021. 2, 3, 4

[3] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 4

[4] Jaesung Choe, Sunghoon Im, Francois Rameau, Minjun Kang, and In So Kweon. Volumefusion: Deep depth fusion for 3d scene reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16086–16095, 2021. 2

[5] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996. 3

[6] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 1, 2, 5

[7] Angela Dai, Matthias Nießner, Michael Zollöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration. *ACM Transactions on Graphics 2017 (TOG)*, 2017. 2, 3, 5

[8] Arda Duzceker, Silvano Galliani, Christoph Vogel, Pablo Speciale, Mihai Dusmanu, and Marc Pollefeys. Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15324–15333, 2021. 2, 4, 5, 8

[9] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017. 2

[10] Ziyue Feng, Leon Yang, Pengsheng Guo, and Bing Li. CVRecon: Rethinking 3d geometric feature learning for neural reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 2

[11] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. Svo: Semidirect visual odometry for monocular and multicamera systems. *IEEE Transactions on Robotics*, 33(2):249–265, 2016. 2

[12] Yasutaka Furukawa, Carlos Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015. 2

[13] Silvano Galliani, Katrin Lasinger, and Konrad Schindler. Massively parallel multiview stereopsis by surface normal diffusion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 873–881, 2015. 2

[14] Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions. *Josa a*, 4(4):629–642, 1987. 7

[15] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[16] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 2, 7

[17] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 2

[18] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-to-end 3d scene reconstruction from posed images. In *European conference on computer vision*, pages 414–431. Springer, 2020. 2, 3, 4, 5

[19] Alexander Rich, Noah Stier, Pradeep Sen, and Tobias Höllerer. 3dvnet: Multi-view depth prediction and volumetric refinement. In *2021 International Conference on 3D Vision (3DV)*, pages 700–709. IEEE, 2021. 2

[20] Mohamed Sayed, John Gibson, Jamie Watson, Victor Prisacariu, Michael Firman, and Clément Godard. Simplerecon: 3d reconstruction without 3d convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2

[21] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European conference on computer vision*, pages 501–518. Springer, 2016. 2

[22] Thomas Schops, Torsten Sattler, and Marc Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 134–144, 2019. 2

[23] Noah Stier, Alexander Rich, Pradeep Sen, and Tobias Höllerer. Vortx: Volumetric 3d reconstruction with transformers for voxelwise view selection and fusion. In *2021 International Conference on 3D Vision (3DV)*, pages 320–330. IEEE, 2021. 2, 3, 4

[24] Jiaming Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. Neuralrecon: Real-time coherent 3d reconstruction from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15598–15607, 2021. zju3dv.github.io/neuralrecon. 2, 3, 4, 5, 8

[25] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[26] Nan Yang, Lukas von Stumberg, Rui Wang, and Daniel Cremers. D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1281–1292, 2020. 2

[27] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018. 2

[28] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. 2