DCHM: Depth-Consistent Human Modeling for Multiview Detection

Supplementary Material

Algorithm 1 Multi-view Label Matching

```
$G$: Set of optimized Gaussians from final training cycle
$M^v$: Set of pedestrian masks for each camera view $v$
$V$: View configuration of the current camera
$\tau_{vis}$: Visibility threshold

1: function LABELMATCHING($G, M^v, V, \tau_{vis}$)
2:     $G_{mask} \leftarrow$ RasterizeGaussianID($G, V$)
3:     for $M_i^v$ in $M^v$ do
4:         if $\exists g \in (G_{mask} \cap M_i^v) : g$ has ID then
5:             $c\_id \leftarrow$ MostFrequentID($G_{mask} \cap M_i^v$)
6:             AssignID($G_{mask} \cap M_i^v, M_i^v, c\_id$)
7:         else
8:             $n\_id \leftarrow$ GenerateNewID()
9:             for all $g$ in $(G_{mask} \cap M_i^v)$ do
10:                $w \leftarrow$ ComputeBlendingWeight($g$)
11:                if $w > \tau_{vis}$ then
12:                    AssignID($G_{mask} \cap M_i^v, M_i^v, n\_id$)
13:                end if
14:            end for
15:        end if
16:    end for
17:    return $G, M^v$
18: end function
```

Algorithm 2 Multi-view Clustering and Localization

```
$\tilde{G}$: Set of Gaussians with assigned IDs
$\tau_{cluster}$: Threshold for minimum number of Gaussians
$\epsilon$: DBSCAN epsilon parameter
$minPts$: DBSCAN minimum points parameter

1: function CLUSTER($\tilde{G}, \tau_{cluster}, \epsilon, minPts$)
2:     $L \leftarrow \{\}$                          ▷ Initialize empty set of locations
3:     for each unique ID $i$ in $\tilde{G}$ do
4:         $\tilde{G}_i \leftarrow \{g \in \tilde{G} : g$ has ID $i\}$
5:         if $|\tilde{G}_i| > \tau_{cluster}$ then
6:             $C_i \leftarrow$ DBSCAN($\tilde{G}_i, \epsilon, minPts$)     ▷ Apply DBSCAN clustering
7:             for each cluster $c$ in $C_i$ do
8:                 $cnt_c \leftarrow$ ComputeCenter($c$)
9:                 $conf_c \leftarrow |c|$     ▷ Confidence is the number of Gaussians
10:                $L \leftarrow L \cup \{(cnt_c, conf_c)\}$
11:            end for
12:        end if
13:    end for
14:    $L \leftarrow$ NMS($L$)                 ▷ Apply Non-Maximum Suppression
15:    return $L$                              ▷ Return final predicted locations
16: end function
```

Algorithm 3 Multi-view Unsupervised Localization

```
$G$: Set of optimized Gaussians from final training cycle
$M^v$: Set of pedestrian masks for all camera views $v$
$V$: Set of view configurations for all cameras
$\tau_{vis}$: Visibility threshold
$\tau_{cluster}$: Minimum Gaussian count for each cluster
$\epsilon$: DBSCAN epsilon parameter
$minPts$: DBSCAN minimum points parameter threshold

1: for all camera $V^c$ in $V$ do                        ▷ Label matching
2:     $\tilde{G}, M^v \leftarrow$ LabelMatching($G, M^v, V^c, \tau_{vis}$)
3: end for
4: $L \leftarrow$ Cluster($\tilde{G}, \tau_{cluster}, \epsilon, minPts$)     ▷ Clustering and localization
```

Algorithm 4 Detection Compensation

```
$r$: Reference view (index) $r$
$D$: Predicted depth of camera view
$K$: Intrinsic matrix of camera view
$Rt$: Extrinsic matrix of camera view
$I$: Image of camera view
$M^v$: Foreground mask of camera view
$\tau_{pcs}$: Threshold for the number of valid projection points
$\tau_m$: Threshold for the proportion of projection points outside the source view foreground mask
$\epsilon$: DBSCAN epsilon parameter
$minPts$: DBSCAN minimum points parameter threshold

1: function DETCOMP($r, D, K, Rt, I, M^v, \tau_{pcs}, \tau_m, \epsilon, minPts$)
2:     for source view $s$ in all cameras do
3:         $u^{s\to r} \leftarrow$ Project($D^s, K^s, Rt^s, M^{sv}, K^r, Rt^r$)  ▷ Source view projections on reference view $r$
4:         $\tilde{u}^{s\to r} \leftarrow$ DBSCAN($u^{s\to r}, \epsilon, minPts$)     ▷ Filter outliers
5:         if $|\tilde{u}^{s\to r}| < \tau_{pcs}$ then Continue
6:         end if                                       ▷ Check projection count
7:         if $\frac{|\tilde{u}^{s\to r} \cap M^r|}{|M^r|} > \tau_m$ then Continue
8:         end if                           ▷ Skip on high overlap in new regions
9:         $bbox \leftarrow$ BBox($\tilde{u}^{s\to r}$)         ▷ Get projection bounds
10:        $M^{r\prime} \leftarrow$ SAM($\tilde{u}^{s\to r}, bbox$)              ▷ Initial mask
11:        $\tilde{u}^{s\to r}, bbox \leftarrow$ Sample($M^{r\prime}$)           ▷ Refine prompts
12:        $M^{r\prime} \leftarrow$ SAM($\tilde{u}^{s\to r}, bbox$)              ▷ Final mask
13:    end for
14: end function
```

6. Supervised localization

We conducted additional experiments to explore if our depth-consistent human modeling can enhance label-based methods.

Methodology. For supervised localization, we adopt the

Method	Label-based	Accuracy (*MODA*)	Speed (FPS)
UMPD [27]	✗	76.6	1.0
Ours	✗	84.2	1.2
MVDet [14]	✓	88.7	5.3
3DROM [33]	✓	93.9	2.4
MvCHM [31]	✓	95.3	2.5
Ours	✓	**95.5**	**6.1**

Table 5. Comparison of accuracy and computational efficiency.

approach from MvCHM [31], where pedestrians are represented as point clouds and an off-the-shelf detection framework is employed. We refer to this process as *label-based aggregation*. To aggregate features from multiple views, we convert point clouds into feature vectors using the network from [21], concatenating them to regress pedestrian positions on the ground plane. We discretize the point clouds into a grid on the BEV plane, creating pillars (vertical voxels [21]). We then use PointNet [32] to extract high-dimensional pillar features. Following the methodology from [53], these features are flattened into the BEV plane for the final position regression. Pedestrian occupancy is represented as Gaussian maps, and we employ focal loss [26] for position regression, defined as:

$$\mathcal{L}_{reg} = -\alpha(1-p)^{\gamma}\log(p), \tag{10}$$

where α and γ are hyperparameters specified in [26].

Comparison. We replace label-free clustering with label-based aggregation to explore the potential of our framework under supervised settings. Quantitative results on the Wildtrack and MultiviewX datasets are reported in Table 6. Our method ("**Ours**") outperforms others across key metrics. Specifically, our method achieves the highest MODA (95.5%) and MODP (90.2%), indicating superior detection accuracy and localization precision. While MvCHM [31] also models pedestrians using point clouds, it relies on the accurate detection of human standing points. In contrast, our method does not require keypoints for localization. Instead, we achieve a dense representation of pedestrians through consistent monocular depth estimation, which significantly aids downstream tasks such as multiview detection.

7. Unsupervised localization

Our multiview unsupervised localization is summarized in Algorithm 3. The main functions, such as *matching* and *clustering*, are listed in Algorithm 1 and 2.

8. Multi-view Detection Compensation

We formalize our multiview detection compensation approach in Algorithm 4.

9. Computational cost

We provide an additional comparison of accuracy (*MODA*) and computational efficiency in Table 5. Integrating proposed human modeling with supervised method outperforms existing label-based approaches, achieving both the highest accuracy and computational speed.

10. Ground depth calculation

This section explains the process of calculating ground depth in the Wildtrack dataset. Although the ground range differs in the MultiviewX dataset, the calculation method remains consistent across both datasets. The ground plane is specified in a world coordinate system with an x-range $[0, 480]$, and a y-range $[0, 1440]$. z is set to 0 by default. We uniformly sample points $\mathbf{p}^w \in \mathbb{R}^3$ on the ground within the predefined range. Given the camera c specified via the rotation matrix $\mathbf{R}^c$ and the translation vector $\mathbf{t}^c \in \mathbb{R}^3$ in the world coordinate system, the ground points $\mathbf{p}^w$ in the world coordinate system are transformed into the camera coordinate system as:

$$d^c = [\mathbf{p}^c]_z, \quad \mathbf{p}^c = (\mathbf{R}^c)^{-1}(\mathbf{p}^w - \mathbf{t}^c), \tag{11}$$

where $[\,\cdot\,]_z$ denotes the z-coordinate of the point in the camera coordinate system. This process is repeated for each camera to calculate the corresponding ground depth.

11. GS depth filtering details

This section provides additional details about the GS depth filtering process. We define $\mathbf{u}^s$ as the homogenous pixel coordinate in source view v^s, $\mathbf{K}^s$ is the source view camera intrinsic matrix. We unproject each pixel $\mathbf{u}^s$ from source view v^s into a 3D point $\mathbf{p}^{s\to w} \in \mathbb{R}^3$ in world coordinate, using predicted depth map $D^s(\mathbf{u}^s)$ and camera poses:

$$\mathbf{p}^{s\to w} = \mathbf{R}^s(D^s(\mathbf{u}^s)(\mathbf{K}^s)^{-1}\mathbf{u}^s) + \mathbf{t}^s. \tag{12}$$

Here, $\mathbf{R}^s$ is the 3×3 camera rotation matrix, and $\mathbf{t}^s \in \mathbb{R}^3$ is the camera translation vector. With reference view camera pose $\mathbf{R}^r$ and $\mathbf{t}^r$, we can project $\mathbf{u}^s$ to reference view v^r and obtain $D^{s\to r}$ via:

$$D^{s\to r}(\mathbf{u}^{s\to r}) = \left[\mathbf{K}^r(\mathbf{R}^r)^{-1}(\mathbf{p}^{s\to w} - \mathbf{t}^r)\right]_z, \tag{13}$$

$$\mathbf{u}^{s\to r} = \mathbf{K}^r(\mathbf{R}^r)^{-1}(\mathbf{p}^{s\to w} - \mathbf{t}^r)/D^{s\to r}(\mathbf{u}^{s\to r}). \tag{14}$$

12. Discussion

Sparse view setup in multi-view detection differs from existing sparse view reconstruction in several aspects. 1. *Large-scale scene*: unlike traditional indoor or object-level settings, we tackle crowded surveillance (e.g., the Wildtrack dataset) covering a $30\,\mathrm{m} \times 40\,\mathrm{m}$ plaza with over

Method	Wildtrack				MultiviewX			
	MODA	MODP	Precision	Recall	MODA	MODP	Precision	Recall
MVDet [14]	88.7	73.6	93.2	95.4	83.9	79.6	96.8	86.7
SHOT [37]	90.8	77.7	96.0	94.3	88.3	82.0	96.6	91.5
MVDeTr [13]	92.1	84.1	96.1	94.5	93.7	91.3	99.5	94.2
3DROM [33]	93.9	76.0	97.7	96.2	95.0	84.9	99.0	96.1
MvCHM [31]	95.3	84.5	98.2	97.1	93.9	88.3	98.5	94.8
Our-Sup	95.5	90.2	98.2	97.4	95.1	91.5	99.1	96.8

Table 6. Performance (%) of ***supervised*** methods on Wildtrack and MultiviewX. **Our-Sup**: our human modeling + *supervised* localization.

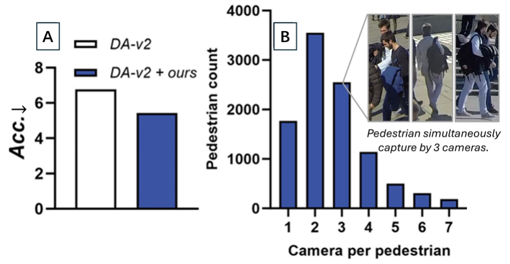

Figure 10. (**A**) Error comparison with DA-v2 on DTU. (**B**) Histogram of the number of pedestrians captured simultaneously by multiple cameras.

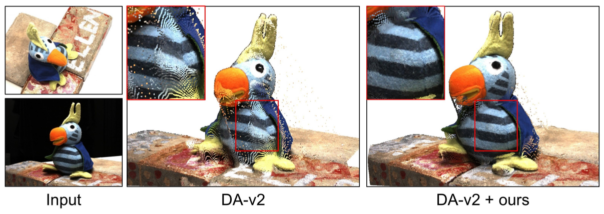

Figure 11. Visual comparison of DepthAnything-v2 (DA-v2) with and without our optimization.

30 pedestrians per frame. 2. *Severe occlusion*: despite of seven cameras, severe occlusion causes most pedestrians (78.67%) to appear in only one to three views (see Fig. 10.B) resulting in an extremely challenging sparse-view setup. Additionally, even though some pedestrians are observed in multiviews, their appearances are dramatically different across views (see Fig. 10.B), making the problem even harder. In such challenging scenarios, generating accurate, consistent depth annotations for fine-tuning is exceptionally difficult. Therefore, our proposed method addresses a meaningful and underexplored challenge.

DCHM also gains in multi-view depth estimation. To quantify its impact on multi-view depth estimation, we evaluate on the DTU benchmark using DepthAnything-v2 (DA-v2) [50] as the baseline. Under sparse-view inputs, DA-v2 predicts relative depth per view, scales it using ground-truth, and fuses results into point clouds via known intrinsics and extrinsics. We finetune DA-v2 using pseudo-depth labels from our framework. Our approach yields more coherent depth maps and smoother fusion as in Fig. 11; and lower depth error as in Fig. 10.B compared to the baseline. While designed for Multiview Detection, our method still improves multi-view depth estimation.

Comparison: Ours *vs.* Gaussian-based 3D Segmentation. Gaussian splatting provides a promising approach for 3D segmentation. Methods like Gaussian Grouping [51] use video tracking to enforce 2D mask consistency across views before projecting to 3D, while Gaga [29] leverages spatial information to associate object masks across multiple cameras. However, these methods require densely overlapping camera views, which limits their applicability in scenarios with sparse view coverage. Our task introduces greater challenges due to minimal scene overlap, making geometric reconstruction particularly demanding. Approaches relying on Gaussian splatting for reconstruction followed by segmentation assignment often struggle under these conditions. In contrast, our framework overcomes these limitations, facilitating effective scene reconstruction and extending 3D segmentation from object-centric domains to large-scale outdoor environments.

13. Limitations and future works

Impact of depth estimation on detection compensation. While multiview detection compensation improves monocular detection, its performance is highly dependent on the accuracy of depth estimation. Error in depth estimation can result in imprecise prompts for SAM, leading to noisy segmentation despite mask-guided sampling. Future work could focus on methods for assessing the validity of compensatory masks to mitigate these issues.

Lack full use of temporal information. Our method, which solely relies on multiview geometry for monocular depth fine-tuning, can lead to instability when objects are visible to only a single camera view. Incorporating temporal information could provide a more robust and consistent approach to depth estimation in such scenarios.

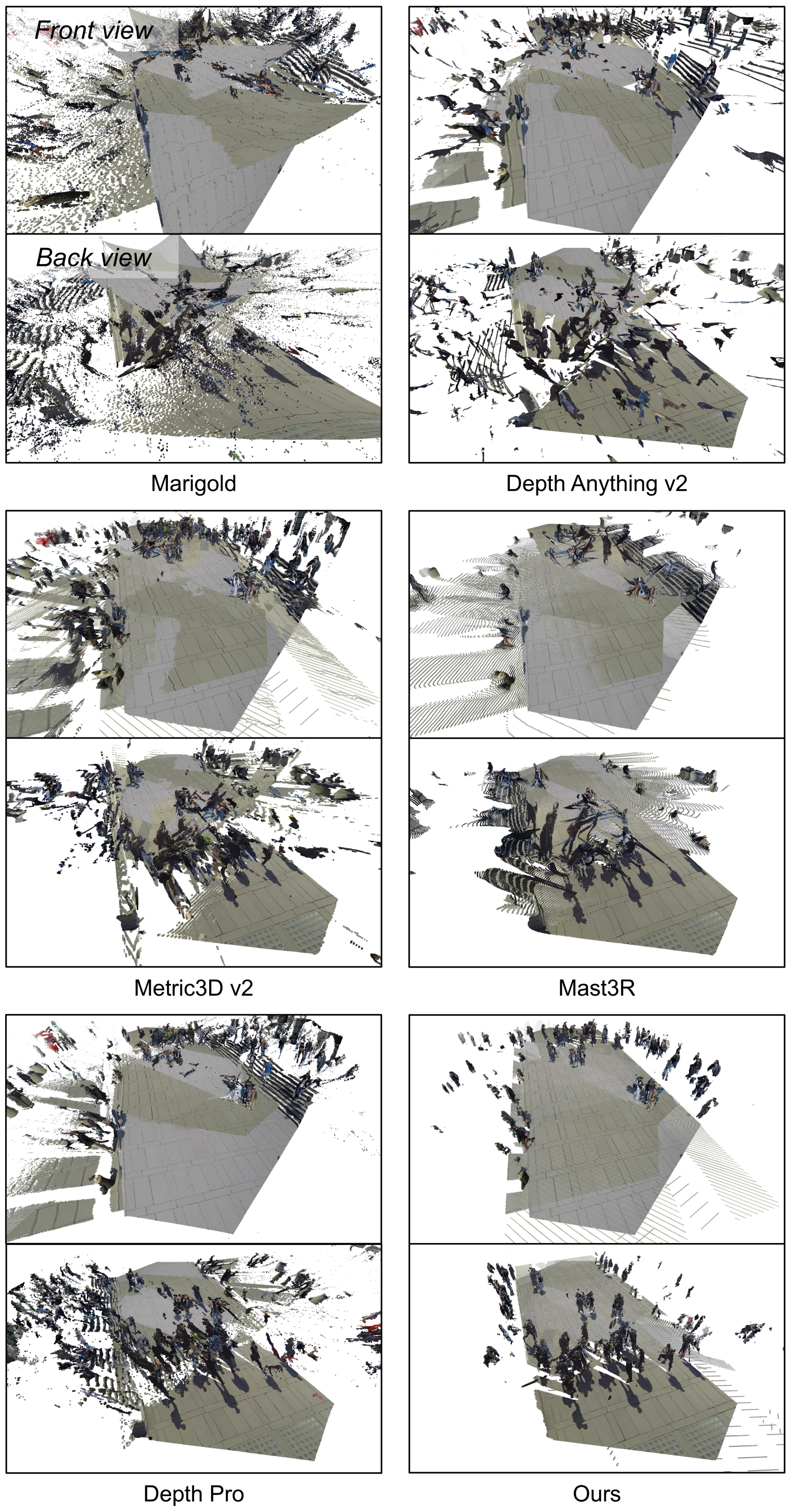

Figure 12. **Enlarged 3D reconstruction from various methods**. We present enlarged views of 3D reconstructions generated by baselines [3, 15, 17, 23, 50], showcasing both front and back perspectives for each. Our method ("Ours") produces more accurate and complete reconstructions compared to the baselines.

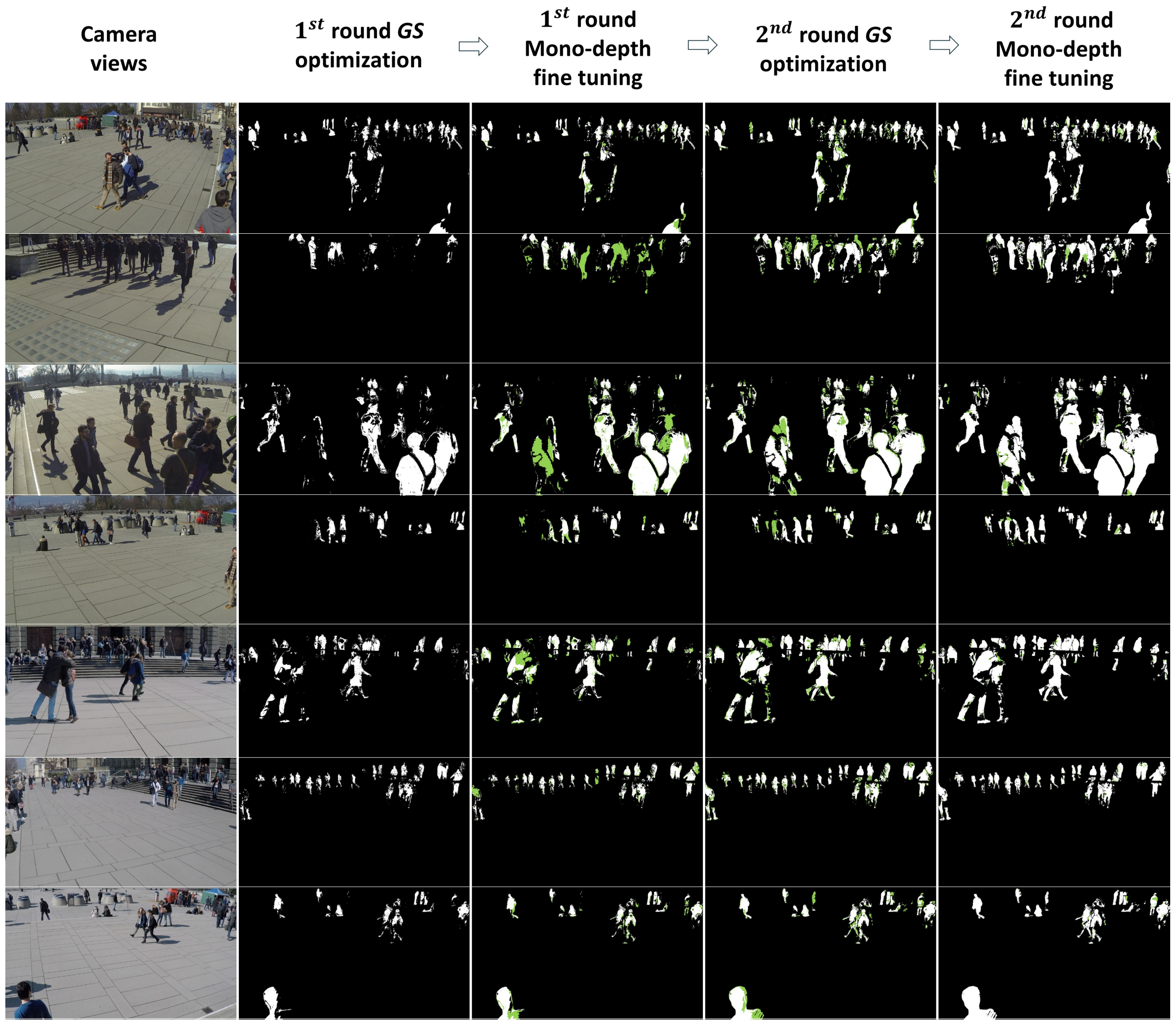
Camera views
1^{st} round *GS* optimization
1^{st} round Mono-depth fine tuning
2^{nd} round *GS* optimization
2^{nd} round Mono-depth fine tuning

Figure 13. **Visualization of increasing valid pseudo-depth across all cameras**. We illustrate the progression of valid depth regions for all viewpoints during each round of optimization. This serves as a supplement visualization for Figure 7 in the main text.