A. Author Contributions

All authors contributed equally.

- **Network Architecture and Model Training**
  Haoyi Zhu (Shanghai AI Lab, USTC),
  Junyi Chen (Shanghai AI Lab, SJTU),
- **Data Collection and Automatic Labeling Pipeline**
  Yifan Wang (Shanghai AI Lab, SJTU),
  Jianjun Zhou (ZJU, Shanghai AI Lab, SII),
  Wenzhang Chang (Shanghai AI Lab, USTC),
  Zizun Li (Shanghai AI Lab, USTC),
  Yang Zhou (Shanghai AI Lab, FDU),
- **Model Evaluation**
  Haoyi Zhu (Shanghai AI Lab, USTC),
  Wenzheng Chang (Shanghai AI Lab, USTC),
- **Paper (figures, visualizations, writing)**
  Haoyi Zhu (Shanghai AI Lab, USTC),
  Wenzheng Chang (Shanghai AI Lab, USTC),
  Junyi Chen (Shanghai AI Lab, SJTU),
  Jianjun Zhou (ZJU, Shanghai AI Lab, SII),
  Yifan Wang (Shanghai AI Lab, SJTU),
  Tong He (Shanghai AI Lab),
- **Leadership (managed and advised on the project)**
  Tong He (Shanghai AI Lab),
- **Consultant (provided valuable advice)**
  Chunhua Shen (ZJU),
  Jiangmiao Pang (Shanghai AI Lab)

We also want to thank Mingyu Liu and Kaipeng Zhang for the helpful discussion.

B. Robustness of Data Annotation Pipeline

Here we detail three key design choices in our methodology that were specifically implemented to enhance its robustness against common sources of uncertainty in dynamic RGB-D processing.

Robustness in Dynamic Masking Grounding SAM 2 often yields erroneous results for out-of-domain semantic inputs. To enhance the robustness of this process, we select prompts with low uncertainty and discard frames with a high mask-to-image ratio. This approach improves the reliability of our dynamic mask generation, thereby increasing the robustness of all subsequent operations.

Robustness Against Inaccurate Flow Estimation In our video slicing process, we utilize optical flow magnitude and the forward-backward error as key metrics. This approach mitigates the uncertainty inherent in flow estimation during coarse camera pose estimation, leading to more robust initial annotations.

Robustness in Points Trajectory Estimation Similarly, our video slicing is performed based on optical flow magnitude and forward-backward error. In addition, we discard frames with an insufficient number of keypoints. These steps yield a video sequence that is both rich in keypoints and temporally coherent (i.e., without frame discontinuities). Such a sequence is highly conducive to tracking estimation methods and these operations also serve to minimize the uncertainty associated with the tracking process.

Robustness in Failure Sequence Filtering As a final step, we filter out erroneous estimations using three key criteria. We discard an entire sequence if it exhibits an anomalous focal length, if its reprojection error relative to point tracking exceeds a predefined threshold, or if its geometric consistency error surpasses a specified limit.

Conclusion on Overall Robustness Our method consistently yields accurate and clean camera poses with minimal noise. Furthermore, the safeguarding operations detailed above ensure that our processed data is virtually free of failure cases. This outcome is the key to the robustness of our approach.

C. Raymap to Camera Parameters Algorithm

We adopt a direct approach to recover camera parameters from raymaps, as shown in Algorithm 1. For more details, please refer to our GitHub repository.

Algorithm 1 Raymap to camera parameters conversion.

```
# Inputs: ray_o (N,H,W,3), ray_d (N,H,W,3)
# Outputs: extrinsics (N,4,4), intrinsic (N,3,3)

# 1. Estimate Camera Position and Orientation
# -------------------------------------------
c = mean(ray_o.reshape(N,-1,3), dim=1) # camera center

# Look-at point is average of ray endpoints
p = mean((ray_o + ray_d).reshape(N,-1,3), dim=1)

# Camera coordinate frame
z = normalize(p - c) # Forward axis (N,3)
x = normalize(mean(ray_d[:,:,-1], dim=1) - mean(ray_d
    [:,:,0], dim=1)) # Right axis (N,3)
y = normalize(cross(z, x)) # Up axis (N,3)
x = normalize(cross(y, z)) # Ensure orthogonality

# 2. Construct Poses Matrix
# -----------------------------
R = stack([x, y, z], dim=2) # Rotation (N,3,3)
t = c.unsqueeze(-1) # Translation (N,3,1)
poses = eye(4).repeat(N,1,1)
poses[:,:3,:3] = R
poses[:,:3,3] = c

# 3. Construct Intrinsics Matrix
# ------------------------------
intrinsics = eye(3).repeat(N,1,1)
intrinsics[:,0,0] = norm(p - c) # Focal length
intrinsics[:,1,1] = norm(p - c) # Assume fx = fy
intrinsics[:,0,2] = W / 2 # Principal point
intrinsics[:,1,2] = H / 2 # Assume at center

extrinsics = inverse(poses)

return extrinsics, intrinsics
```

`normalize`: L2 normalization; `cross`: cross product; `eye`: identity matrix.

D. Generation Experiments Details

Prediction Validation Dataset Construction. For the validation set of prediction tasks, we collected 93 in-domain scenes and 43 out-of-domain scenes, with each scene corresponding to a synthetic video clip. The in-domain scenes are collected from the same synthetic environments used in the training dataset, while the out-of-domain scenes are sourced from entirely different synthetic environments that are not present in the training data.

Video Prediction Task Settings. For prediction tasks without action conditions, both Aether and CogVideoX take the first frame as input. However, since CogVideoX tends to generate static scenes without text prompts, we utilize GPT-4o to generate text annotations for each image. The prompt for GPT-4o is designed to:

1) Generate text labels describing the scene content 2) Predict the potential motion patterns of each object 3) Predict the most likely camera trajectory based on the image content 4) For scenes with clear subjects, predict camera movements that follow the subject 5) For scenes without prominent subjects, predict reasonable camera movements based on the scene context 6) Emphasize dynamic video generation with camera movements that closely track subjects or rapidly move to showcase the scene

The generated text labels and the first frame serve as input for CogVideoX, with a negative prompt set to "static background, static camera, slow motion, slow camera movement, low dynamic degree" and a guidance scale of 6.0. In contrast, Aether only takes the first frame as input, and sets obs guidance scale to 3.0.

Action Conditioned Video Prediction Task Settings. For action conditioned prediction tasks, Aether accepts both the first frame as observation image input and the camera trajectory of the video clip as action-conditioned input. To ensure fair comparison, we use GPT-4o to generate detailed text annotations for both the initial and final frames of each video clip. These annotations serve as text prompts for CogVideoX, providing comprehensive camera trajectory descriptions. The prompt template for GPT-4o is designed to:

1) Describe the initial frame in detail 2) Predict the video content based on both frames, including: - Object movements and interactions - Scene dynamics - Camera motion patterns 3) Analyze the differences between the start and end frames to: - Determine the precise camera movement trajectory - For scenes with clear subjects, describe how the camera follows them - For scenes without prominent subjects, predict the most probable camera movements 4) Emphasize dynamic scene generation with active camera movements

This approach provides CogVideoX with more detailed camera motion descriptions compared to the action-free setting, serving as an equivalent to Aether's explicit action conditions.

VBench Evaluation Protocol. We adopt VBench as our evaluation metric system for prediction tasks. Given the differences in input settings between Aether and CogVideoX, we evaluate the generated videos under the custom input configuration of VBench across six dimensions:

1) Subject Consistency: Evaluates the temporal consistency of main subjects

2) Background Consistency: Measures the stability and coherence of scene backgrounds

3) Motion Smoothness: Assesses the fluidity and naturalness of movements

4) Dynamic Degree: Quantifies the level of motion and activity

5) Aesthetic Quality: Measures the visual appeal and artistic merit

6) Imaging Quality: Evaluates the technical quality of video generation

The final score is computed as a weighted average of these dimensions using the official VBench weights:

- Subject Consistency: 1.0
- Background Consistency: 1.0
- Motion Smoothness: 1.0
- Dynamic Degree: 0.5
- Aesthetic Quality: 1.0
- Imaging Quality: 1.0

Based on the VBench evaluation results, as shown in Tables 3 and 4, Aether demonstrates superior overall performance compared to CogVideoX across these metrics.

Video Planning Settings. For planning tasks, we construct a validation set following a similar approach to the prediction tasks, comprising 80 in-domain scenes and 40 out-of-domain scenes from synthetic environments. For each video clip, we extract the initial and final frames as inputs for both Aether and Aether-no-depth models.

For action-conditioned tasks, we evaluate model performance using pixel-wise metrics (PSNR, SSIM, MS-SSIM, and LPIPS) as shown in Table 5. For action-free tasks, we employ the VBench evaluation metrics as presented in Table 6. Both evaluation protocols demonstrate that Aether consistently outperforms the Aether-no-depth model, validating the effectiveness of our approach.

E. Additional Losses in Stage 2 Training

In our second training stage, we decode the latent representations into image space and employ three distinct losses: MS-SSIM loss for color videos, Scale- and Shift-Invariant

(SSI) loss for depth videos, and Pointmap loss for raymaps. Each loss is tailored to the unique characteristics of the respective modality, ensuring effective supervision across all tasks.

E.1. Multi-Scale Structural Similarity (MS-SSIM) Loss for Color Videos

For color videos, we use the Multi-Scale Structural Similarity (MS-SSIM) loss to preserve perceptual quality and structural coherence across multiple scales. Unlike pixel-wise losses, MS-SSIM captures luminance, contrast, and structural differences between predicted $\hat{\mathbf{I}}$ and ground truth $\mathbf{I}$ frames. At each scale, the structural similarity index is computed as:

$$\text{SSIM}(\hat{\mathbf{I}}, \mathbf{I}) = \frac{(2\mu_{\hat{I}}\mu_I + C_1)(2\sigma_{\hat{I}I} + C_2)}{(\mu_{\hat{I}}^2 + \mu_I^2 + C_1)(\sigma_{\hat{I}}^2 + \sigma_I^2 + C_2)},$$

where $\mu_{\hat{I}}, \mu_I$ are local means, $\sigma_{\hat{I}}, \sigma_I$ are standard deviations, $\sigma_{\hat{I}I}$ is the cross-covariance, and C_1, C_2 are constants to stabilize division. MS-SSIM is computed across multiple scales by downsampling the input, with weights $\{w_i\}$:

$$\text{MS-SSIM} = \prod_{i=1}^{M} \text{SSIM}_i^{w_i}.$$

The MS-SSIM loss is defined as:

$$\mathcal{L}_{\text{MS-SSIM}} = 1 - \text{MS-SSIM}.$$

This loss is particularly effective for color videos, as it emphasizes structural similarity over pixel-wise accuracy.

E.2. Scale- and Shift-Invariant (SSI) Loss for Depth Videos

Depth predictions often suffer from scale and shift ambiguities. To address this, we use a Scale- and Shift-Invariant (SSI) loss, which aligns the predicted depth $\hat{\mathbf{D}}$ with the ground truth $\mathbf{D}$ by computing optimal scale s and shift t as follows:

$$s, t = \arg\min_{s,t} \|\mathbf{M} \odot (s\hat{\mathbf{D}} + t - \mathbf{D})\|^2,$$

where $\mathbf{M}$ is a binary mask for valid pixels, and $\odot$ is the element-wise product. The SSI loss combines a data term and a gradient regularization term:

$$\mathcal{L}_{\text{SSI}} = \mathcal{L}_{\text{data}} + \alpha\mathcal{L}_{\text{gradient}},$$

where α balances the contribution of gradient regularization. The gradient term enforces local smoothness in depth predictions, ensuring geometric consistency.

E.3. Pointmap Loss for Raymaps

Raymaps encode 3D spatial information, and their alignment requires a loss invariant to scale and translation. We transform predicted disparity and raymaps into 3D pointmaps $\mathbf{P}$ using:

$$\mathbf{P} = \mathbf{D} \cdot \mathbf{R}_d + \mathbf{R}_o,$$

where $\mathbf{D}$ is the depth, $\mathbf{R}_d$ is the ray direction, and $\mathbf{R}_o$ is the ray origin. The pointmap loss minimizes the difference between predicted and ground truth pointmaps:

$$\mathcal{L}_{\text{pointmap}} = \frac{1}{N}\sum_{i=1}^{N} w_i \|\hat{\mathbf{P}}_i - \mathbf{P}_i\|_p,$$

where w_i is a weight inversely proportional to depth, p is the norm type (e.g., L_1 or L_2), and N is the number of valid points. This loss ensures accurate 3D spatial alignment, which is critical for raymap-based tasks. Note that the pointmap loss only back-propagates gradients to raymap latents, and we stop the disparity gradients during pointmap projection.

F. More Ablation Study

Acknowledging the importance of ablation studies and working within our computational resources, we conducted a key ablation in Sec. 5.2, where the depth component was removed during training. Results presented in Tab. 5 and 6 demonstrate that excluding the 4D reconstruction target from the multi-task co-training leads to a notable degradation in visual planning performance. This finding strongly supports our paper's central claim regarding the effective integration of reconstruction and generation within a unified framework. Qualitative results further illustrating this are provided in Fig 7.

G. More analysis in Sec. 4

Our model performs well on the Sintel and Kitti datasets but is comparatively weaker on BONN. The trend is also observed in other diffusion-based methods. We suggest two primary reasons for this. First, BONN's scene type is indoor. This may be less compatible with the learned priors of video diffusion models. Second, as an older dataset, BONN exhibits lower image quality and contains artifacts such as motion blur. Diffusion models can be particularly sensitive to such image characteristics, potentially impacting their performance.

H. More Training data details

Our synthetic data collection approach directly follows DAV and TheMatrix, capturing RGB-D videos from AAA games such as Cyberpunk2077 and Horizon5. The initial raw

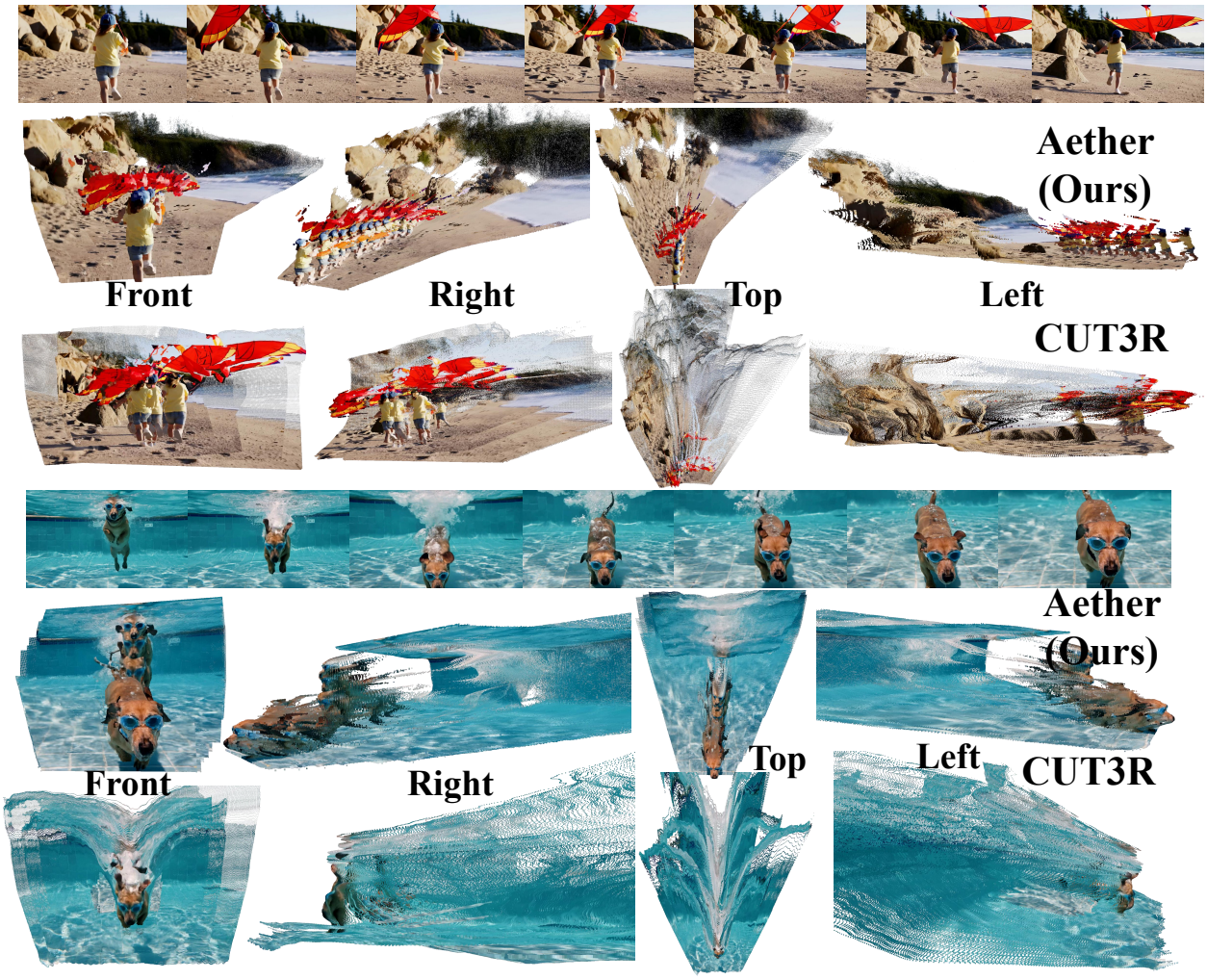

Figure 5. More reconstruction visualizations.

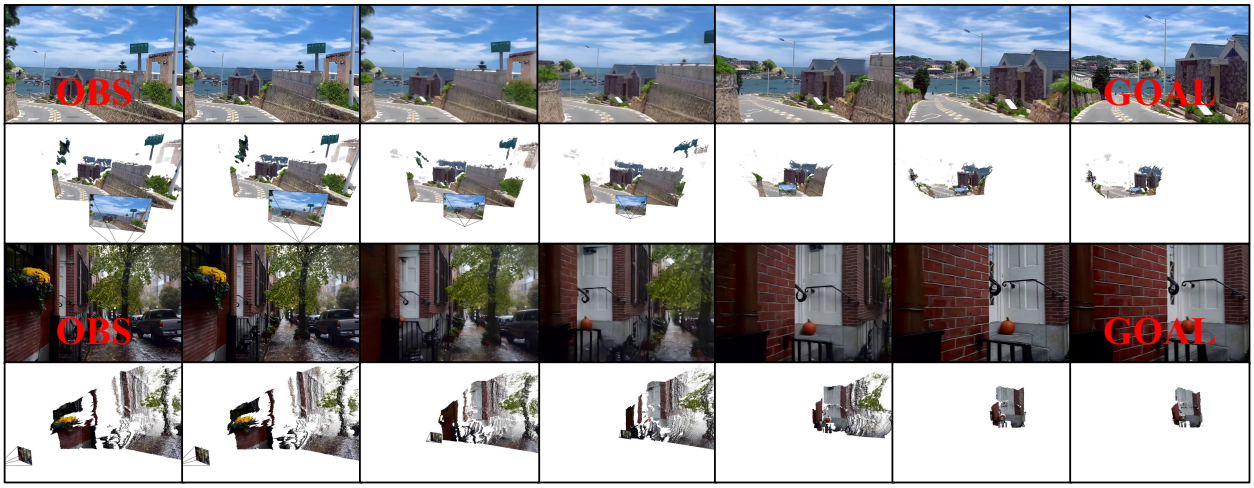

Figure 6. More visual planning examples.

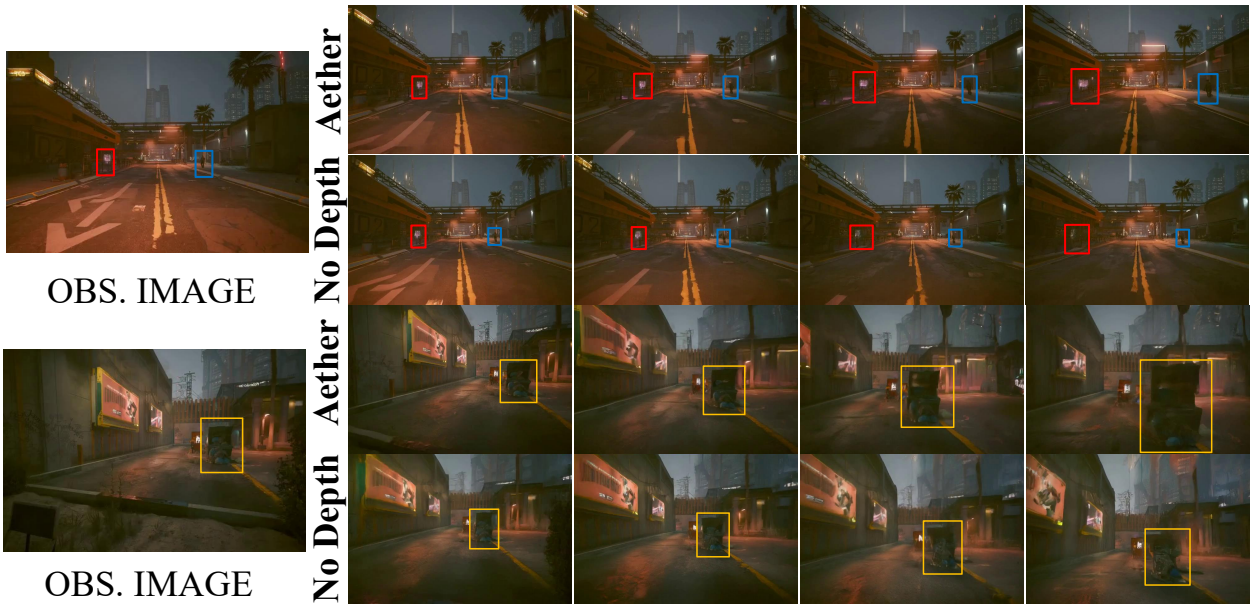

Figure 7. Qualitative results for ablation study. ***Please zoom in.***

dataset contained about 12.5 million frames. After undergoing camera pose annotation and filtering, this collection was refined to approximately 8.9 million well-annotated frames, which were subsequently used for training. Our camera pose estimation is comparable to other feed-forward methods, which typically trade the higher accuracy of optimization-based techniques for superior run-time efficiency. Reduced performance on ScanNet is likely due to the domain gap from synthetic training data, alongside ScanNet's imperfect annotations and motion blur.

I. Running time differences.

See Tab. 7.

Table 7. Reconstruction running FPS differences on A100.

Method	DUSt3R-GA	MASt3R-GA	MonST3R-GA	Aether (Ours)
Resolution	144 × 512	144 × 512	144 × 512	480 × 640
FPS	0.76	0.31	0.35	6.14