Learning to recover 3D shape

from a single image

Len Bauer

May 29, 2024

Example

Distorted Point Cloud

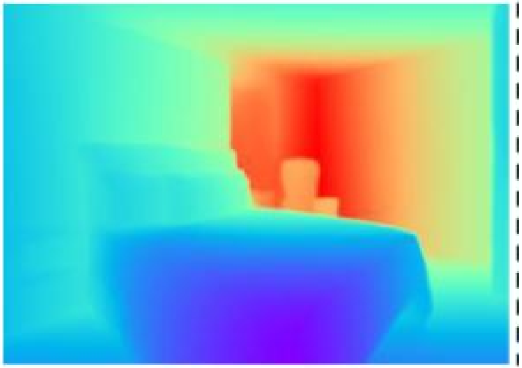


Figure: Predicted Depth

Figure: Walls clearly stretched

Example

Recovered Shift

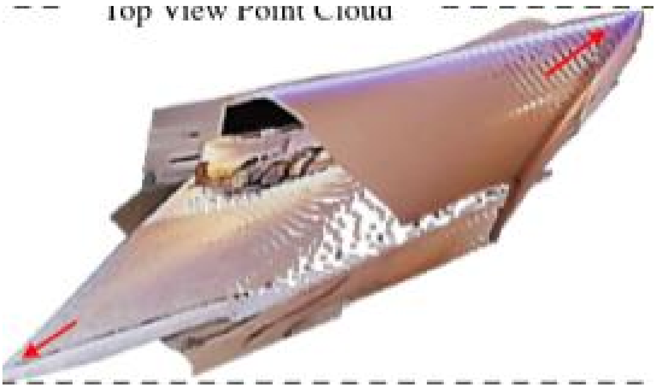


Figure: Edges are now straight

Example

Recovered Shift & Focal Length

Figure: Final result of 3d shape

DPM Training

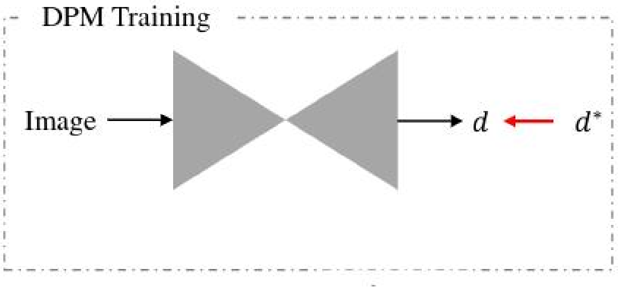


Figure: Depth prediction model

Method Pipeline

DPM Training

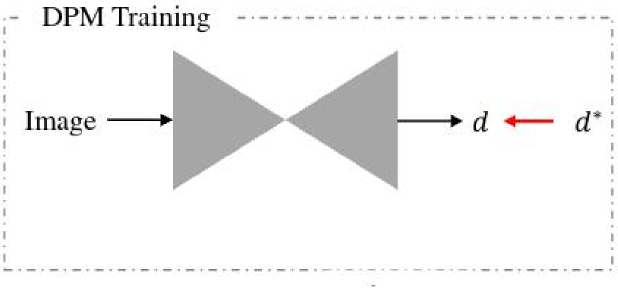


Figure: Depth prediction model

- CNN trained on mixture of existing datasets

DPM Training

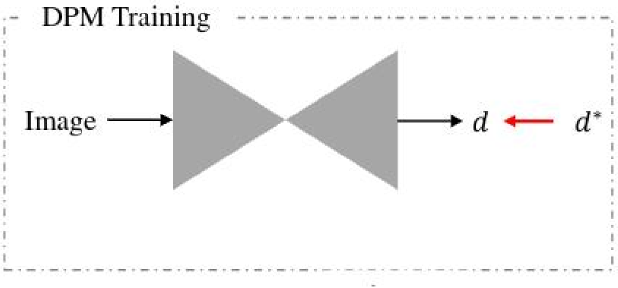


Figure: Depth prediction model

- CNN trained on mixture of existing datasets
- predicts depth maps
- fails to predict scale and shift

Method Pipeline

PCM Training

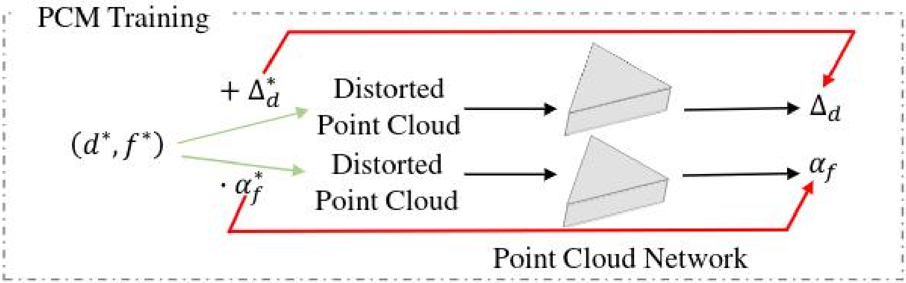


Figure: Point cloud module

Method Pipeline

PCM Trainin

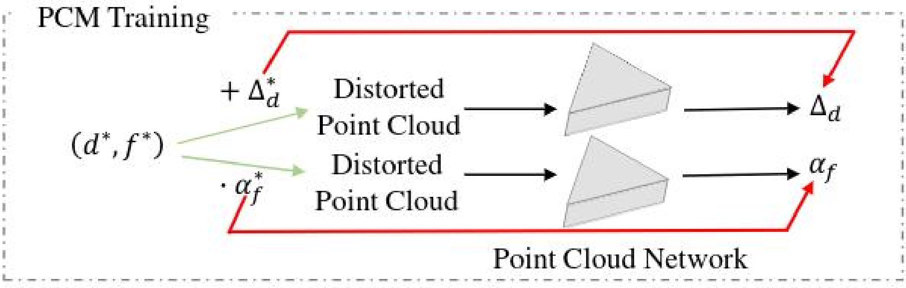


Figure: Point cloud module

- Point cloud encoder

Method Pipeline

PCM Training

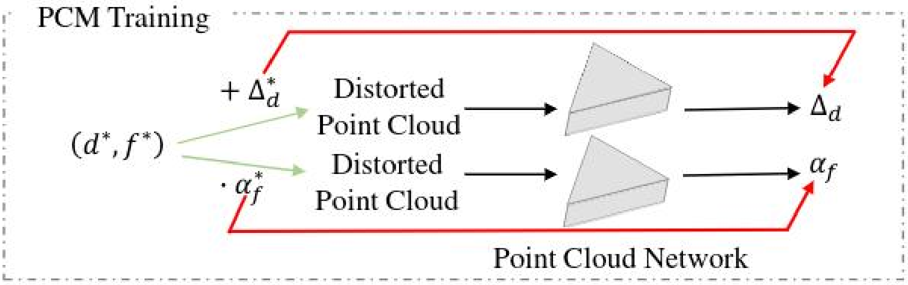


Figure: Point cloud module

- Point cloud encoder
- takes initial guess
- predicts shift and focal length adjustment factors

Method Pipeline

Inference

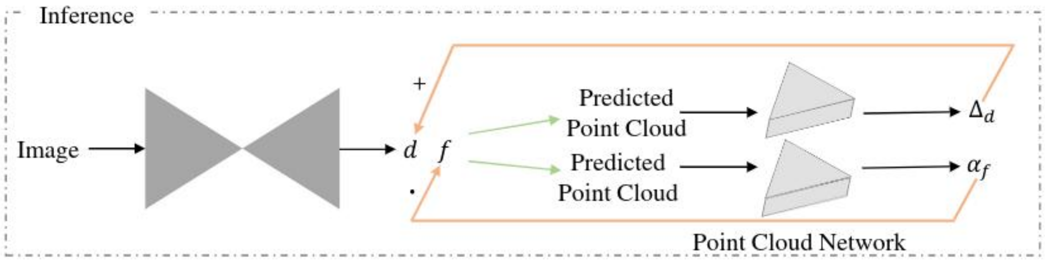


Figure: Both models combined together!

Key Observation

PCM training

Operates on point clouds derived from depth maps, not images.

Key Observation

PCM training

Operates on point clouds derived from depth maps, not images.

- Hence we can train models to learn 3D scene shape **priors** using:

Key Observation

PCM training

Operates on point clouds derived from depth maps, not images.

- Hence we can train models to learn 3D scene shape **priors** using:
  - Synthetic 3D data

Key Observation

PCM training

Operates on point clouds derived from depth maps, not images.

- Hence we can train models to learn 3D scene shape **priors** using:
  - Synthetic 3D data
  - Data from 3D laser scanning devices

Key Observation

PCM training

Operates on point clouds derived from depth maps, not images.

- Hence we can train models to learn 3D scene shape **priors** using:
  - Synthetic 3D data
  - Data from 3D laser scanning devices
- Domain gap between datasets is less significant for point clouds than for images.

Key Observation

PCM training

Operates on point clouds derived from depth maps, not images.

- Hence we can train models to learn 3D scene shape **priors** using:
  - Synthetic 3D data
  - Data from 3D laser scanning devices
- Domain gap between datasets is less significant for point clouds than for images.
- Point cloud data sources are less diverse than internet images.

Monocular Depth Estimation

Monocular depth estimation

... relies on high-level scene priors and data-driven approaches.

- Challenges include:
  - Diversity of training data from different cameras.
  - Different image priors affecting depth estimation.
- Web stereo images and videos provide depth supervision up to a scale and shift due to unknown camera baselines.

State-of-the-art

Models use loss functions invariant to scale and shift.

- Camera focal length may not be accessible at test time, leading to 3D scene shape distortion.
- Scene shape distortion is critical for downstream tasks (e.g. 3D photography).

Point Cloud Module

We use a pinhole camera model for 3D point cloud reconstruction:

$$\begin{cases} x = \dfrac{u - u_0}{f} d \\ y = \dfrac{v - v_0}{f} d \\ z = d \end{cases} \tag{1}$$

Key points:

- (u_0, v_0): Camera optical center

Point Cloud Module

We use a pinhole camera model for 3D point cloud reconstruction:

$$\begin{cases} x = \dfrac{u - u_0}{f} d \\ y = \dfrac{v - v_0}{f} d \\ z = d \end{cases} \tag{1}$$

Key points:

- (u_0, v_0): Camera optical center
- f: Focal length
- d: Depth

Point Cloud Module

We use a pinhole camera model for 3D point cloud reconstruction:

$$\begin{cases} x = \dfrac{u - u_0}{f} d \\ y = \dfrac{v - v_0}{f} d \\ z = d \end{cases} \tag{1}$$

Key points:

- (u_0, v_0): Camera optical center
- f: Focal length
- d: Depth
- f scales x and y, not z

Point Cloud Module

We use a pinhole camera model for 3D point cloud reconstruction:

$$\begin{cases} x = \dfrac{u - u_0}{f} d \\ y = \dfrac{v - v_0}{f} d \\ z = d \end{cases} \tag{1}$$

Key points:

- (u_0, v_0): Camera optical center
- f: Focal length
- d: Depth
- f scales x and y, not z
- Shift in d

Point Cloud Module

We use a pinhole camera model for 3D point cloud reconstruction:

$$\begin{cases} x = \dfrac{u - u_0}{f} d \\ y = \dfrac{v - v_0}{f} d \\ z = d \end{cases} \tag{1}$$

Key points:

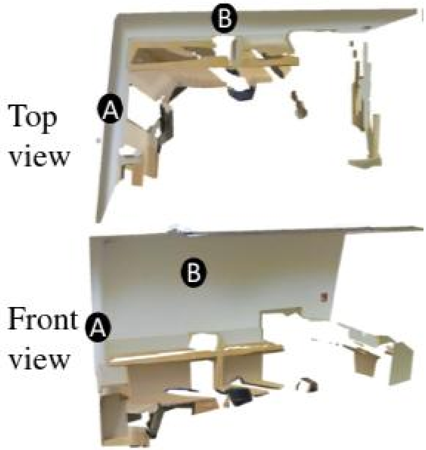
- (u_0, v_0): Camera optical center
- f: Focal length
- d: Depth
- f scales x and y, not z
- Shift in d affects x, y, and z non-uniformly, causing shape distortions

Example distortion

RGB

Example distortion

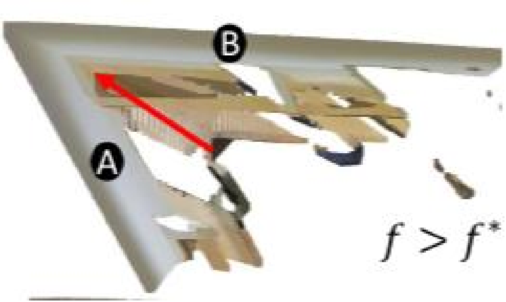
GT point cloud

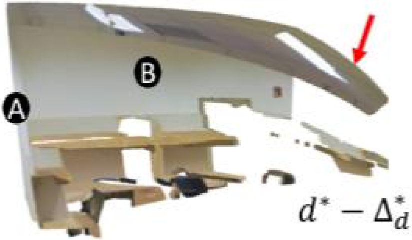


GT point cloud

Example distortion

Distorted point cloud





Training Process

Perturbed input point cloud with incorrect shift and focal length:

- Ground truth depth d^* transformed by shift Δ_d^*:

$$\Delta_d^* \sim$$

Training Process

Perturbed input point cloud with incorrect shift and focal length:

- Ground truth depth d^* transformed by shift Δ_d^*:
$$\Delta_d^* \sim \mathcal{U}(-0.25, 0.8)$$
- Ground truth focal length f^* scaled by α_f^*:
$$\alpha_f^* \sim$$

Training Process

Perturbed input point cloud with incorrect shift and focal length:

- Ground truth depth d^* transformed by shift Δ_d^*:
$$\Delta_d^* \sim \mathcal{U}(-0.25, 0.8)$$
- Ground truth focal length f^* scaled by α_f^*:
$$\alpha_f^* \sim \mathcal{U}(0.6, 1.25)$$

Objective Functions

Depth shift recovery:

Focal length recovery:

Objective Functions

Depth shift recovery:

- Perturbed 3D point cloud $\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*)$

Focal length recovery:

Objective Functions

Depth shift recovery:

- Perturbed 3D point cloud $\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*)$
- Input to shift point cloud network $\mathcal{N}_d(\cdot)$
- Objective function:

Focal length recovery:

Objective Functions

Depth shift recovery:

- Perturbed 3D point cloud $\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*)$
- Input to shift point cloud network $\mathcal{N}_d(\cdot)$
- Objective function:

$$L = \min_{\theta} \| \qquad \| \tag{2}$$

Focal length recovery:

Objective Functions

Depth shift recovery:

- Perturbed 3D point cloud $\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*)$
- Input to shift point cloud network $\mathcal{N}_d(\cdot)$
- Objective function:

$$L = \min_{\theta} \|\mathcal{N}_d(\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*), \theta) - \Delta_d^*\| \tag{2}$$

Focal length recovery:

Objective Functions

Depth shift recovery:

- Perturbed 3D point cloud $\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*)$
- Input to shift point cloud network $\mathcal{N}_d(\cdot)$
- Objective function:

$$L = \min_{\theta} \|\mathcal{N}_d(\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*), \theta) - \Delta_d^*\| \tag{2}$$

Focal length recovery:

- Point cloud $\mathcal{F}(u_0, v_0, \alpha_f^* f^*, d^*)$

Objective Functions

Depth shift recovery:

- Perturbed 3D point cloud $\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*)$
- Input to shift point cloud network $\mathcal{N}_d(\cdot)$
- Objective function:

$$L = \min_{\theta} \|\mathcal{N}_d(\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*), \theta) - \Delta_d^*\| \tag{2}$$

Focal length recovery:

- Point cloud $\mathcal{F}(u_0, v_0, \alpha_f^* f^*, d^*)$
- Input to focal length point cloud network $\mathcal{N}_f(\cdot)$
- Objective function:

$$L = \min_{\theta} \| \qquad \| \tag{3}$$

Objective Functions

Depth shift recovery:

- Perturbed 3D point cloud $\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*)$
- Input to shift point cloud network $\mathcal{N}_d(\cdot)$
- Objective function:

$$L = \min_{\theta} \|\mathcal{N}_d(\mathcal{F}(u_0, v_0, f^*, d^* + \Delta_d^*), \theta) - \Delta_d^*\| \tag{2}$$

Focal length recovery:

- Point cloud $\mathcal{F}(u_0, v_0, \alpha_f^* f^*, d^*)$
- Input to focal length point cloud network $\mathcal{N}_f(\cdot)$
- Objective function:

$$L = \min_{\theta} \|\mathcal{N}_f(\mathcal{F}(u_0, v_0, \alpha_f^* f^*, d^*), \theta) - \alpha_f^*\| \tag{3}$$

Inference Process

During inference:

- GT depth $\longrightarrow$ predicted depth d, normalized $[0, 1]$.

Experiments:

Inference Process

During inference:

- GT depth $\longrightarrow$ predicted depth d, normalized $[0, 1]$.
- Initial guess of focal length f.
- $\hookrightarrow$ Reconstructed point cloud $\mathcal{F}(u_0, v_0, f, d)$.

Experiments:

Inference Process

During inference:

- GT depth $\longrightarrow$ predicted depth d, normalized $[0, 1]$.
- Initial guess of focal length f.
- $\hookrightarrow$ Reconstructed point cloud $\mathcal{F}(u_0, v_0, f, d)$.
- Fed to networks $\mathcal{N}_d(\cdot)$ and $\mathcal{N}_f(\cdot)$ to predict:
  - Shift Δ_d
  - Focal length scaling factor α_f

Experiments:

Inference Process

During inference:

- GT depth $\longrightarrow$ predicted depth d, normalized $[0, 1]$.
- Initial guess of focal length f.
- $\hookrightarrow$ Reconstructed point cloud $\mathcal{F}(u_0, v_0, f, d)$.
- Fed to networks $\mathcal{N}_d(\cdot)$ and $\mathcal{N}_f(\cdot)$ to predict:
  - Shift Δ_d
  - Focal length scaling factor α_f

Experiments:

- Initial focal length with field of view (FOV) of 60°.
- Two separate networks for better performance.

Normalization

Whats the problem with Min-Max normalization?

Depth prediction module

Normalization

Whats the problem with Min-Max normalization?

- **Normalization**: Transforms each ground truth depth map to a similar numerical range.
- **Combined Methods**:
  - Tanh normalization
  - Trimmed Z-score
  - Pixel-wise mean average error (MAE)

Depth prediction module

Image-level Normalized Regression Loss

$$L_{ILNR} =$$

where

- d_i: Predicted depth
- d_i^*: Ground truth depth
- $\overline{d_i^*}$: Normalized depth
- μ_{trim}: Mean of trimmed depth map
- σ_{trim}: Standard deviation of trimmed depth map

Depth prediction module

Image-level Normalized Regression Loss

$$L_{ILNR} =$$

where

$$\overline{d_i^*} = \frac{d_i^* - \mu_{trim}}{\sigma_{trim}}$$

- d_i: Predicted depth
- d_i^*: Ground truth depth
- $\overline{d_i^*}$: Normalized depth
- μ_{trim}: Mean of trimmed depth map
- σ_{trim}: Standard deviation of trimmed depth map

Depth prediction module

Image-level Normalized Regression Loss

$$L_{ILNR} = \frac{1}{N} \sum_{i=1}^{N} \left| d_i - \overline{d_i^*} \right|$$

where

$$\overline{d_i^*} = \frac{d_i^* - \mu_{trim}}{\sigma_{trim}}$$

- d_i: Predicted depth
- d_i^*: Ground truth depth
- $\overline{d_i^*}$: Normalized depth
- μ_{trim}: Mean of trimmed depth map
- σ_{trim}: Standard deviation of trimmed depth map

Depth prediction module

Image-level Normalized Regression Loss

$$L_{ILNR} = \frac{1}{N}\sum_{i=1}^{N}\left|d_i - \overline{d_i^*}\right| + \left|\tanh\left(\frac{d_i}{100}\right) - \tanh\left(\frac{\overline{d_i^*}}{100}\right)\right| \tag{4}$$

where

$$\overline{d_i^*} = \frac{d_i^* - \mu_{trim}}{\sigma_{trim}}$$

- d_i: Predicted depth
- d_i^*: Ground truth depth
- $\overline{d_i^*}$: Normalized depth
- μ_{trim}: Mean of trimmed depth map
- σ_{trim}: Standard deviation of trimmed depth map

Pair-wise Normal Loss

Pair-wise normal loss. Normals are an important geometric property and a complementary modality to depth. Many methods use normal constraints to improve depth quality (e.g., virtual normal loss), but they often miss local geometric quality.

- Proposed method: Pair-wise normal (PWN) loss
- Benefits:
  - Enforces supervision in surface normal space
  - Includes edges and planes
  - Better constraints on global and local geometric relations

Pair-wise Normal Loss

Surface Normal Calculation

- Obtained from reconstructed 3D point cloud using local least squares fitting.
- Align predicted and ground truth depth with scale and shift factor.
- Sample 100K paired points per training sample.
- Ensure global geometric quality by sampling paired points globally.

Pair-wise Normal Loss

PWN Loss Definition

$$L_{PWN} = \frac{1}{N}\sum_{i=1}^{N} \left| n_{Ai} \cdot n_{Bi} - n^*_{Ai} \cdot n^*_{Bi} \right| \tag{5}$$

where n^* denotes ground truth surface normals.

Pair-wise Normal Loss

Multi-scale Gradient Loss

$$L_{MSG} = \frac{1}{N} \sum_{k=1}^{K} \sum_{i=1}^{N} \left| \nabla_k^x d_i - \nabla_k^x d_i^* \right| + \left| \nabla_k^y d_i - \nabla_k^y d_i^* \right| \tag{6}$$

Pair-wise Normal Loss

Loss Application

Dataset	Structure-guided ranking loss	ILNR	PWN (plane)	PWN (edge)	Multi-scale gradient loss
Taskonomy	✓	✓	✓	✓	✓
3D Ken Burns	✓	✓	✓	✓	✓
DIML	✓	✓	✓		✓
HRWSI+Holopix	✓				
Weight	1	1	1	1	0.5

Table: Losses on different datasets.

Taskonomy

Holopix

Holopix50k 7

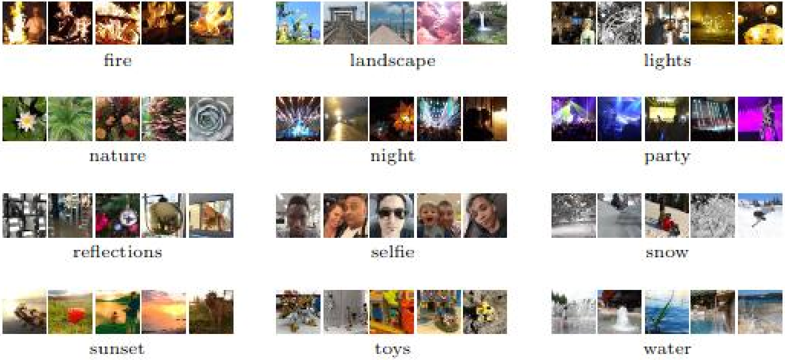
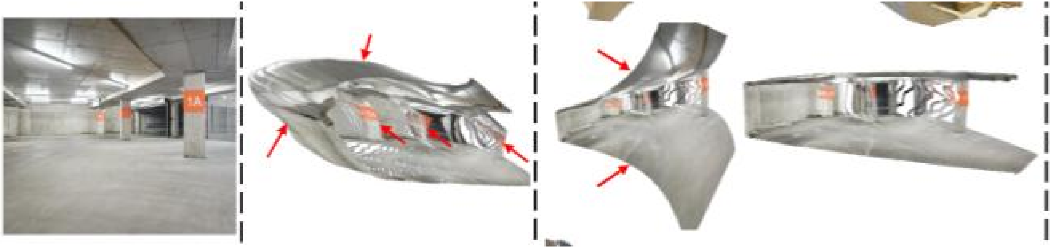


Fig. 4. Diversity of content found in the Holopix50k dataset

Limitations

- unusual camera properties
- uncommon view angle (top-to-down).
- strange angle of wall
- flat image with few geometric cues: e.g. sky
- radial distortion: e.g. fish eye lense

End of Presentation



$||RGB||$ MiDaS (another state-of-the-art 2021 tool) $||''$ *Ours − Baseline*'' $||''$ *Ours*'' $||$