Robotics and
Visual Intelligence Lab

Digging Into Self-Supervised Monocular Depth Estimation (Monodepth2)

Clement Godard[1] Oisin Mac Aodha[2] Michael Firman[3] Gabriel Brostow[3,1]
[1]UCL [2]Caltech [3]Niantic

ICCV 2019

Gyeongsu Cho @UNIST

2022.07.28 (Thu)

Introduction

Method

Experiments

Conclusion

Introduction

Monocular Depth Estimation

Introduction

Monocular Depth Estimation

RGB

Depth

Introduction

Monocular Depth Estimation

RGB

Depth

Why Monocular Depth?

Introduction

Monocular Depth Estimation

RGB

Depth

Why Monocular Depth?

Introduction

Monocular Depth Estimation

RGB

Depth

Why Monocular Depth?

Introduction

Monocular Depth Estimation

RGB

Depth

Why Monocular Depth?

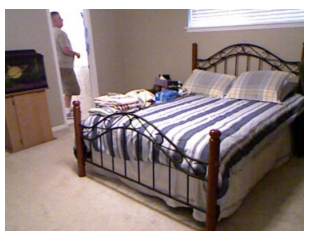
Introduction

Supervised Monocular Training

Self-supervised Monocular Training

Introduction

Supervised Monocular Training [1]



RGB

→ Model →

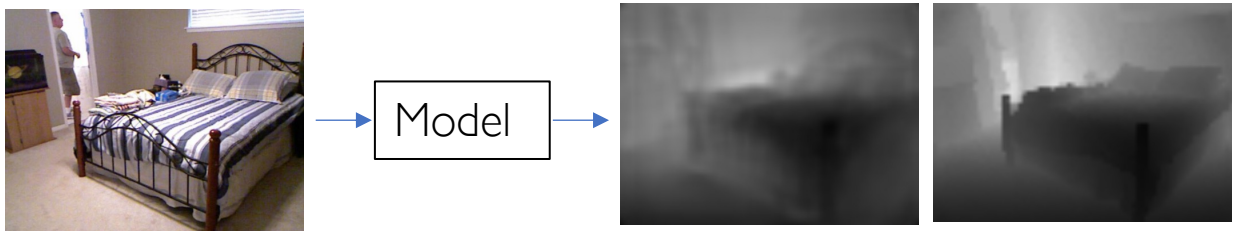
Depth prediction

Self-supervised Monocular Training

[1] Eigen, et al, Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In NIPS, 2014

Introduction

Supervised Monocular Training [1]



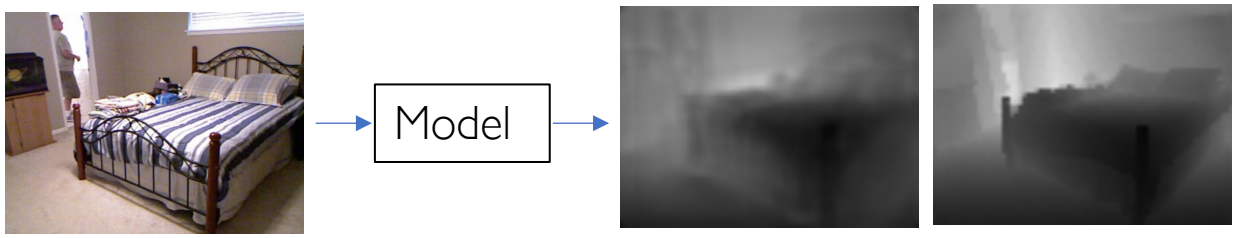
RGB Depth prediction GT

Self-supervised Monocular Training

[1] Eigen, et al, Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In NIPS, 2014

Introduction

Supervised Monocular Training [1]



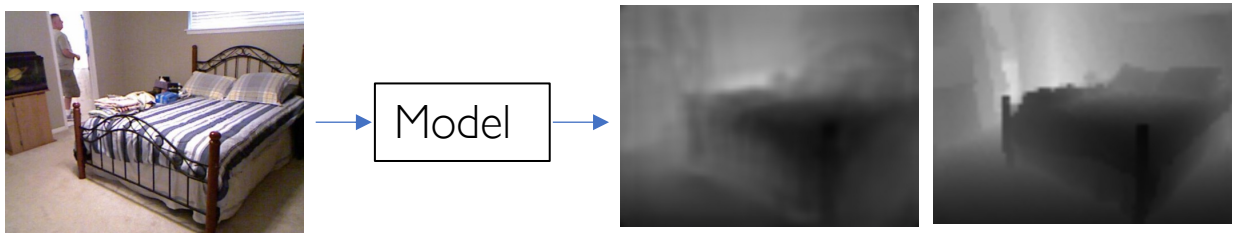
RGB Depth prediction GT

Self-supervised Monocular Training

[1] Eigen, et al, Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In NIPS, 2014

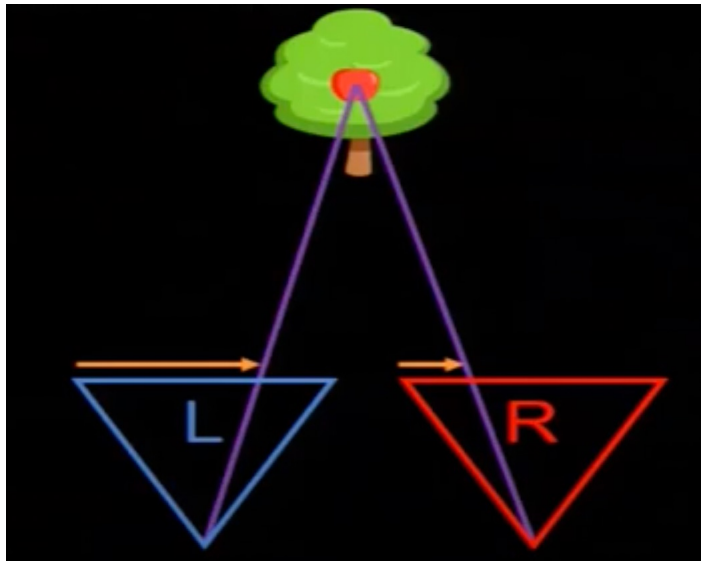
Introduction

Supervised Monocular Training [1]



RGB Depth prediction GT

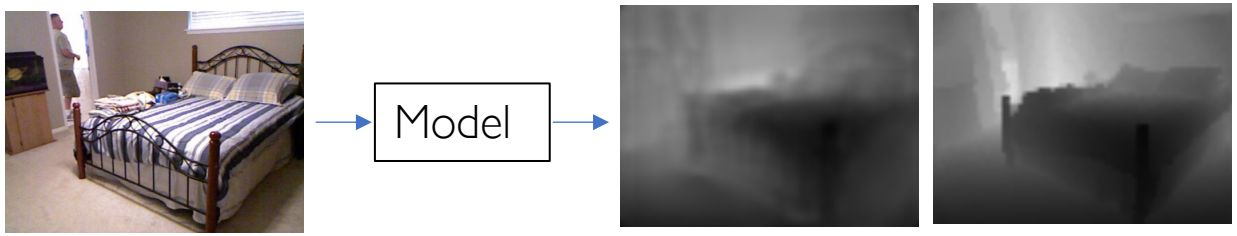
Self-supervised Monocular Training



[1] Eigen, et al, Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In NIPS, 2014

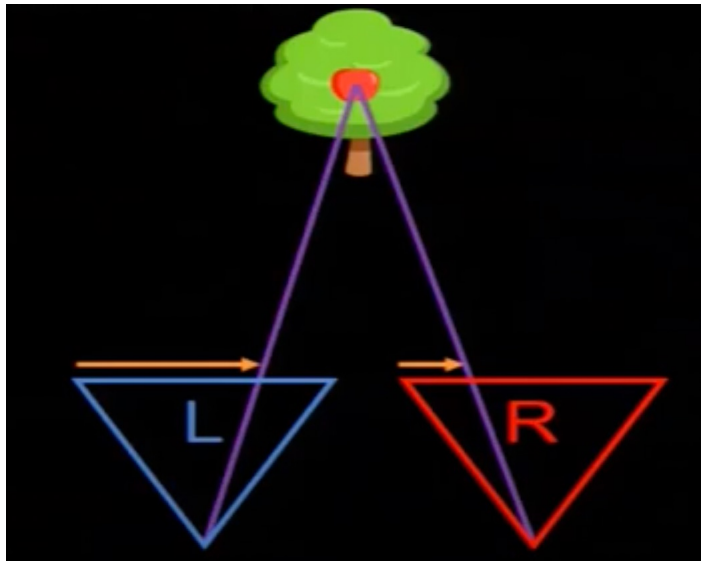
Introduction

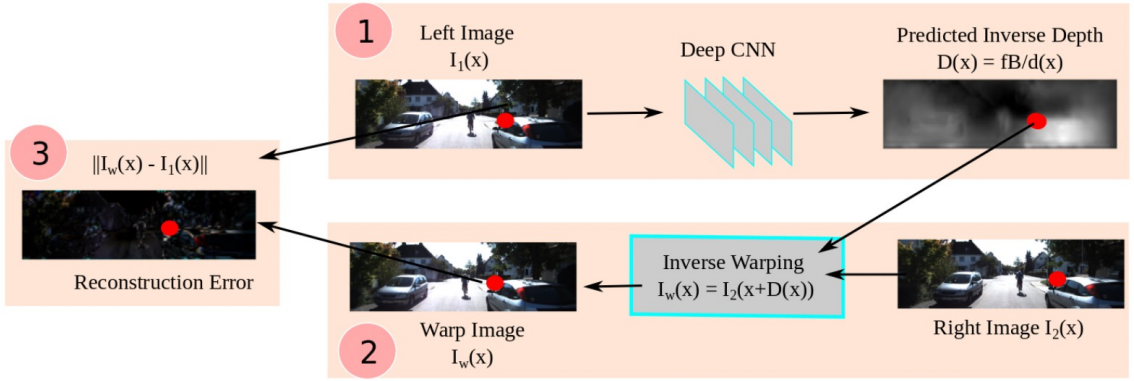
Supervised Monocular Training [1]



RGB Depth prediction GT

Self-supervised Monocular Training [2]

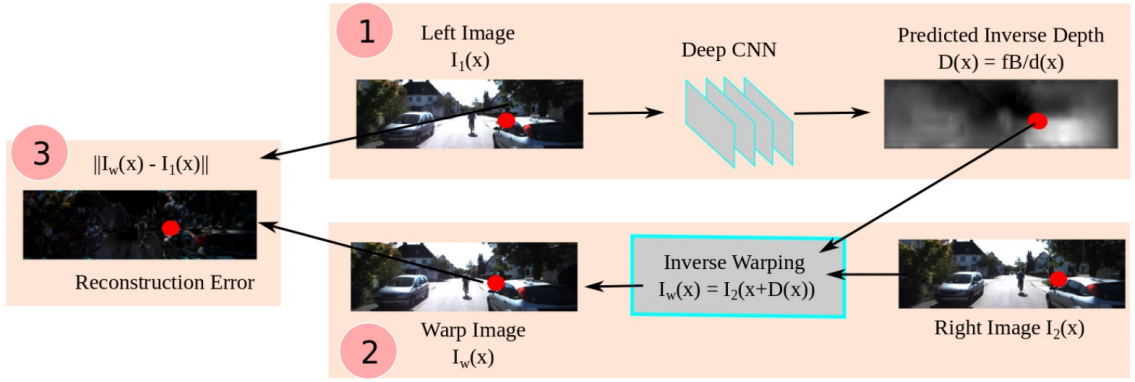




[1] Eigen, et al, Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. In NIPS, 2014
[2] Grag, et al, Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue, In ECCV, 2016

Introduction

Stereo pairs [2]

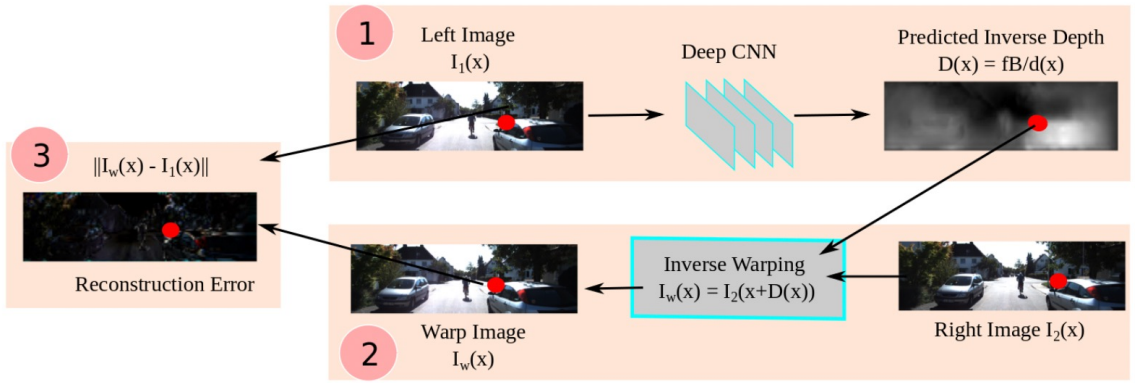


Monocular videos

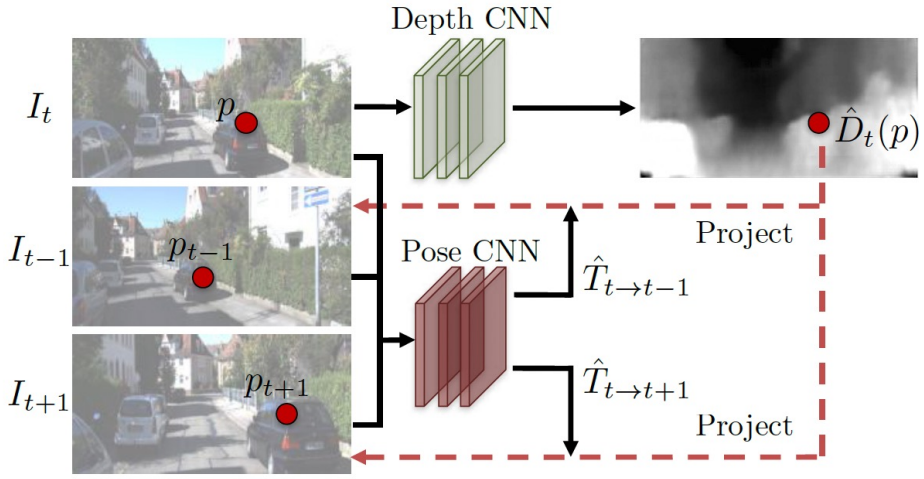
[2] Grag, et al, Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue, In ECCV, 2016

Introduction

Stereo pairs [2]



Monocular videos [3]



[2] Grag, et al, Unsupervised CNN for Single View Depth Estimation: Geometry to the Rescue, In ECCV, 2016
[3] Zhou, et al, Unsupervised Learning of Depth and Ego-Motion from Video, In CVPR, 2017

Introduction

Contribution

1. A **novel appearance matching loss** to address the problem of **occluded pixels** that occur when using monocular supervision

2. A novel and simple **auto-masking approach** to ignore pixels where no relative camera motion is observed in monocular training

3. A **multi-scale appearance matching loss** that performs all image sampling at the input resolution, leading to a reduction in depth artifacts

Introduction

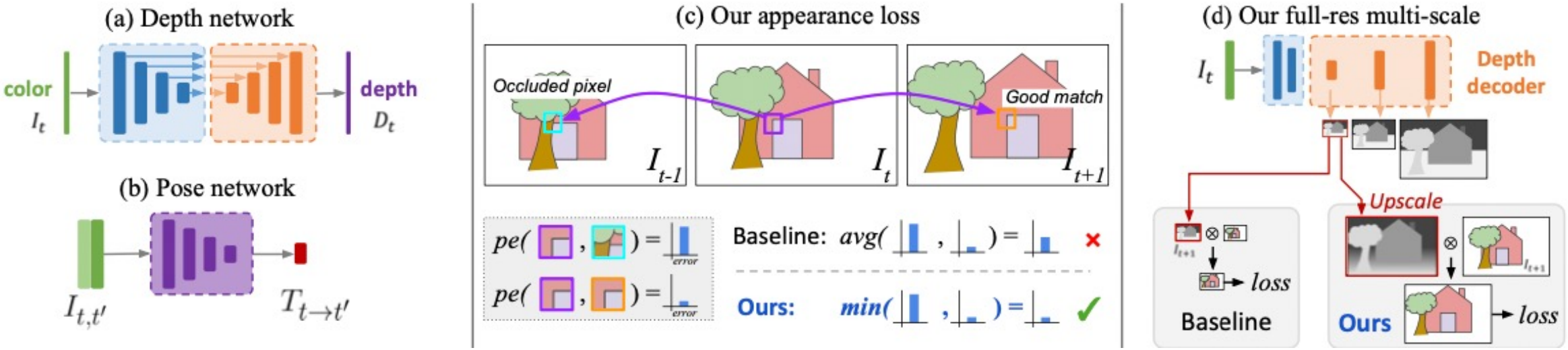
Method

Experiments

Conclusion

Method

- Overview

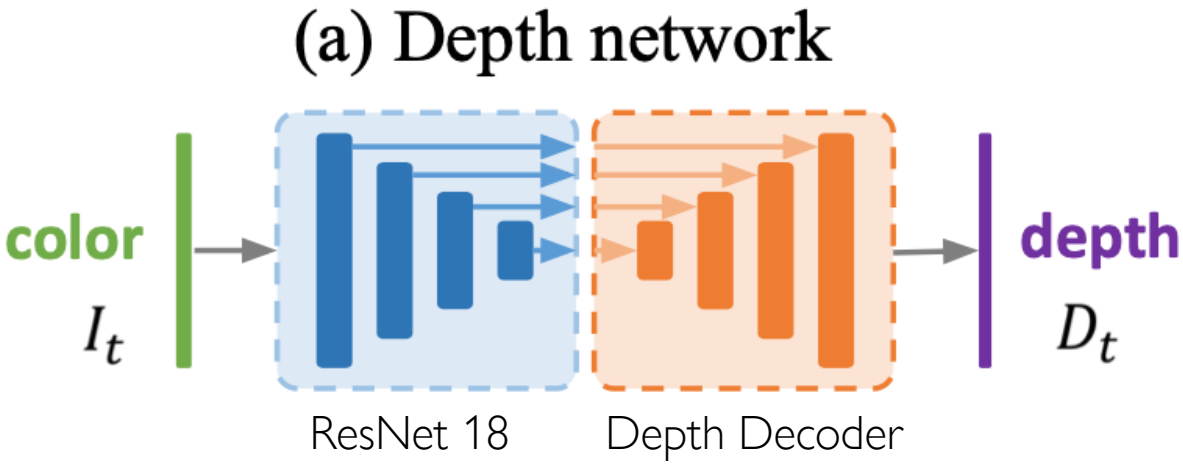


Method

- Network

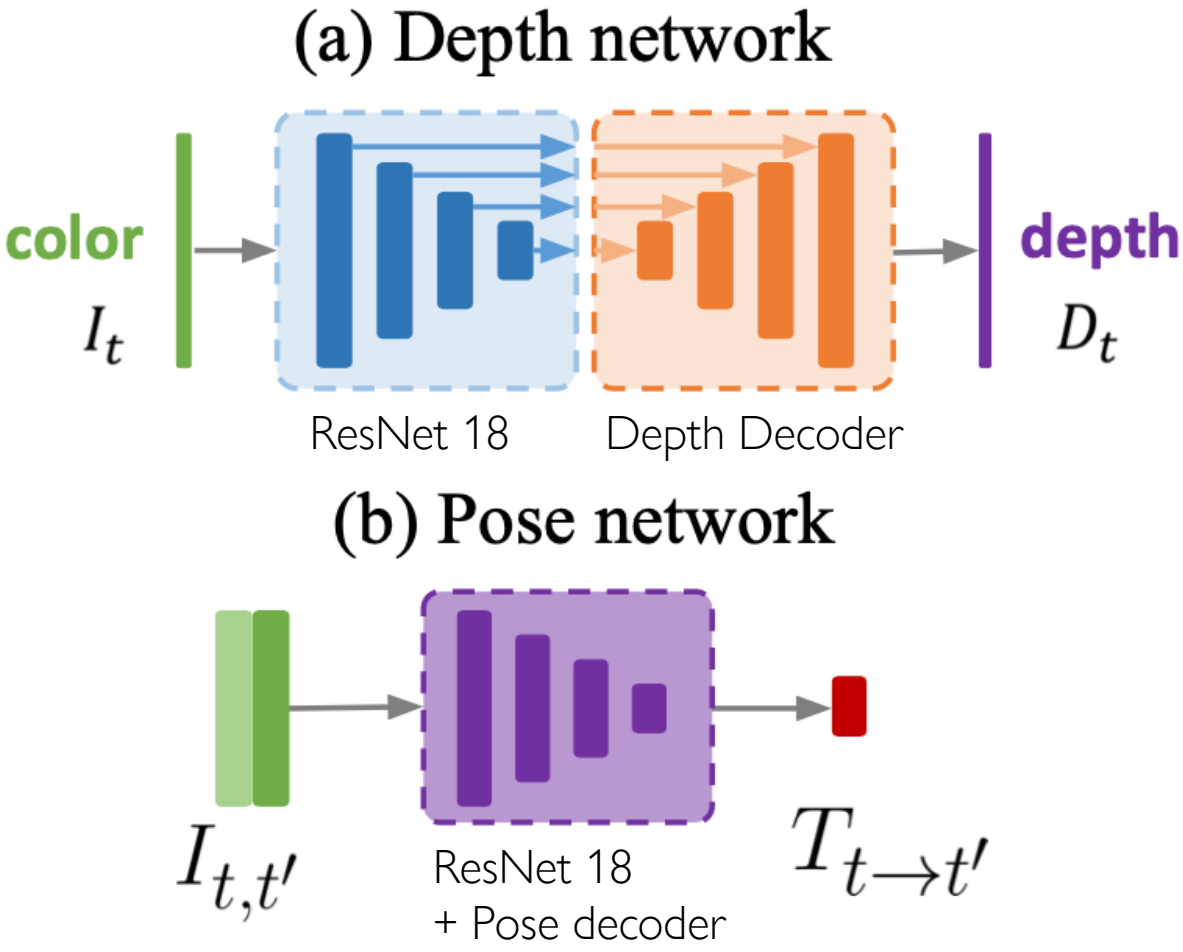
Method

- Network



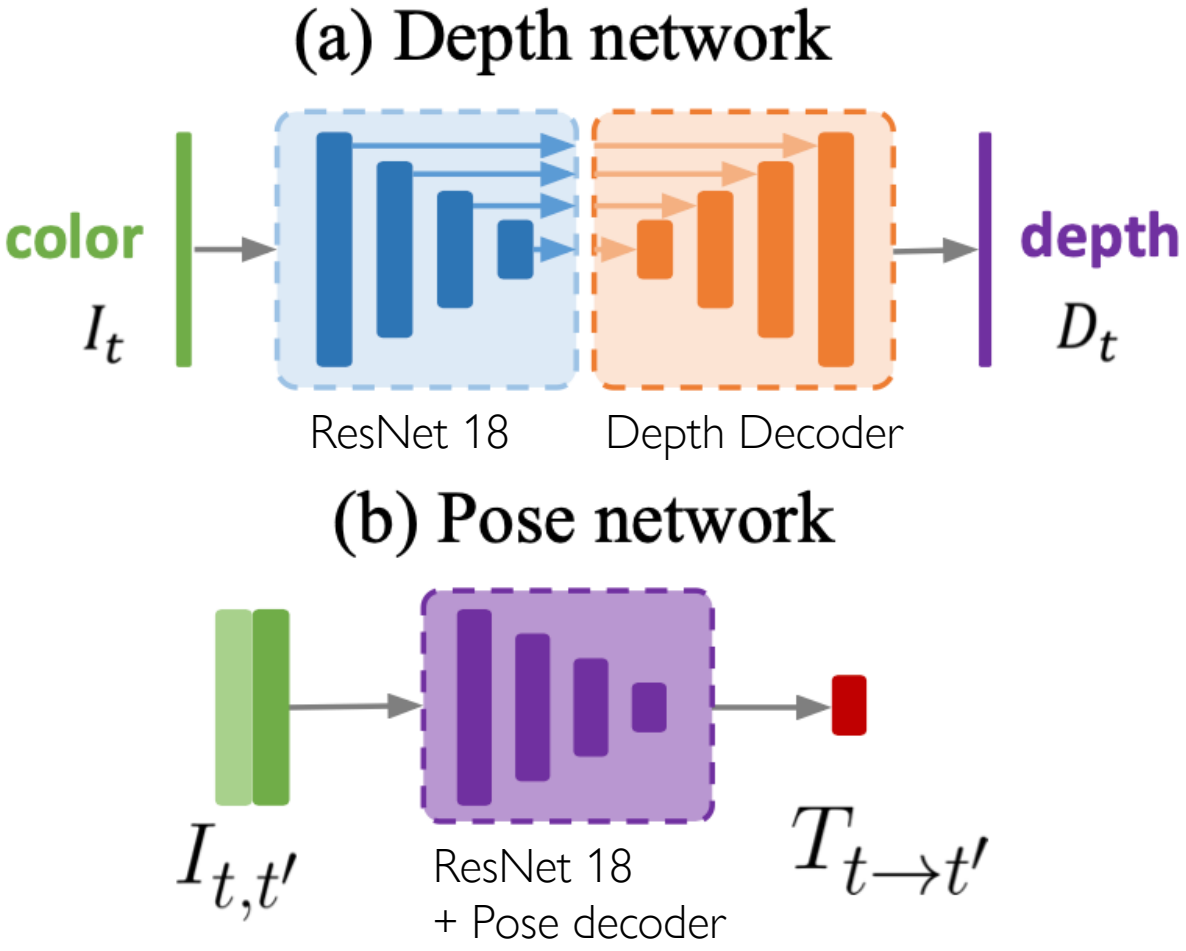
Method

- Network



Method

- Network



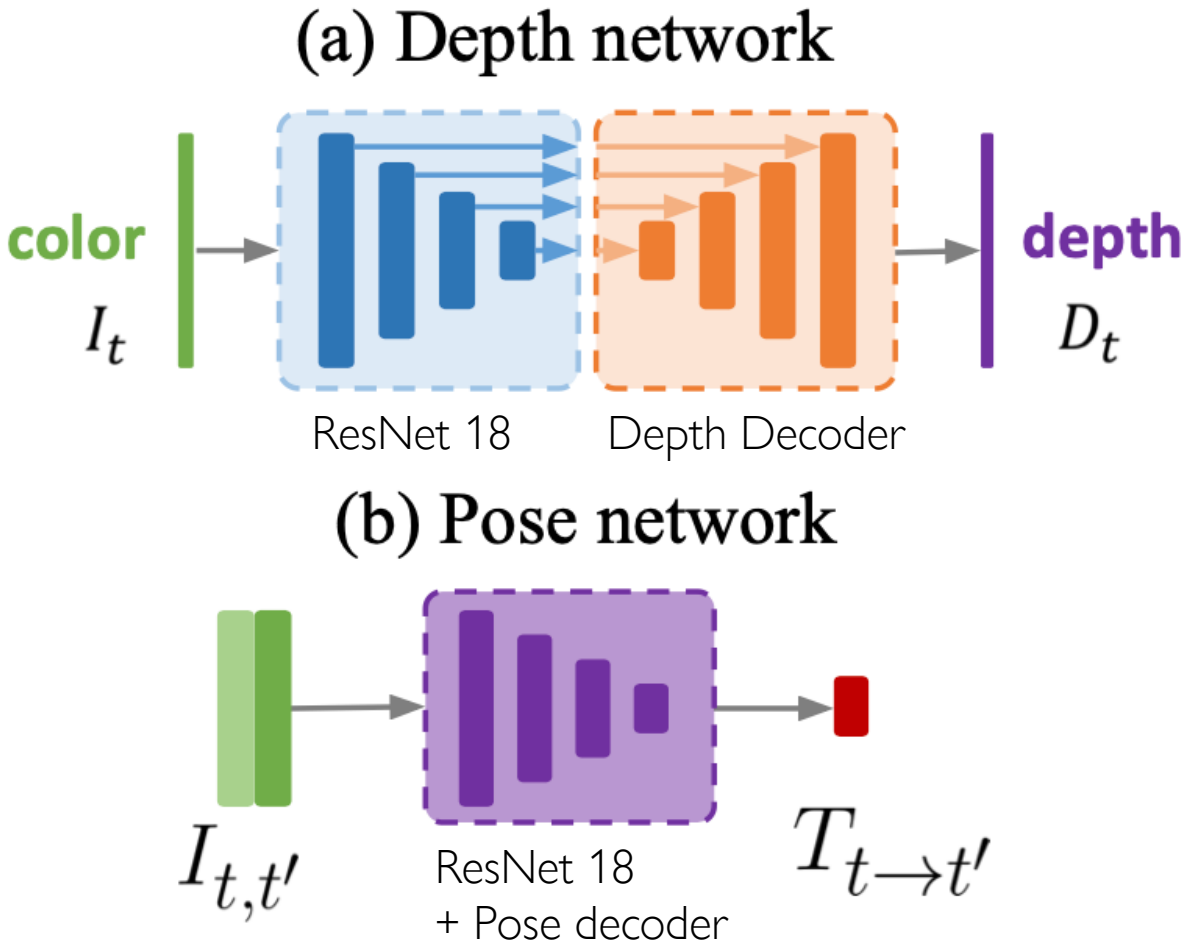
- Loss function

$$L_p = \sum_{t'} pe(I_t, I_{t'\to t}),$$

$$I_{t'\to t} = I_{t'}\left\langle proj(D_t, T_{t\to t'}, K)\right\rangle$$

Method

- Network



- Loss function

$$L_p = \sum_{t'} pe(I_t, I_{t'\to t}),$$

$$I_{t'\to t} = I_{t'}\left\langle proj(D_t, T_{t\to t'}, K)\right\rangle$$

Method

- Projection & Warp

$$
\begin{aligned}
L_p &= \sum_{t'} pe(I_t, I_{t'\to t}), \\
I_{t'\to t} &= I_{t'}\left\langle proj(D_t, T_{t\to t'}, K)\right\rangle
\end{aligned}
$$

Method

- Projection & Warp

$$
\begin{aligned}
L_p &= \sum_{t'} pe(I_t, I_{t'\to t}), \\
I_{t'\to t} &= I_{t'}\left\langle proj(D_t, T_{t\to t'}, K)\right\rangle
\end{aligned}
$$

$$
p_{t'} \sim K T_{t\to t'} D_t(p_t) K^{-1} p_t
$$

Method

- Projection & Warp

$$L_p = \sum_{t'} pe(I_t, I_{t'\to t}),$$

$$I_{t'\to t} = I_{t'}\left\langle proj(D_t, T_{t\to t'}, K)\right\rangle$$

$$p_{t'} \sim K T_{t\to t'} D_t(p_t) K^{-1} p_t$$

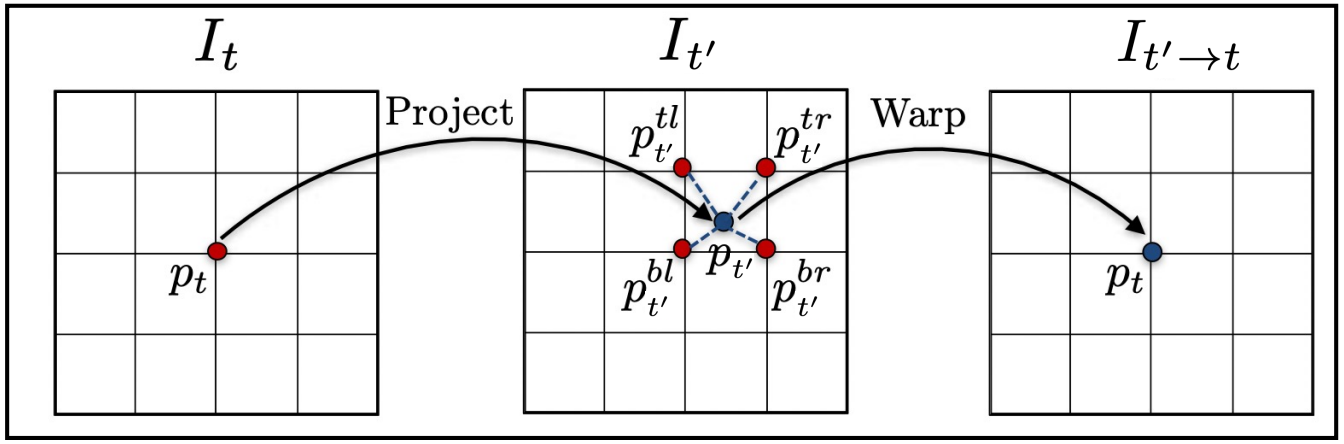
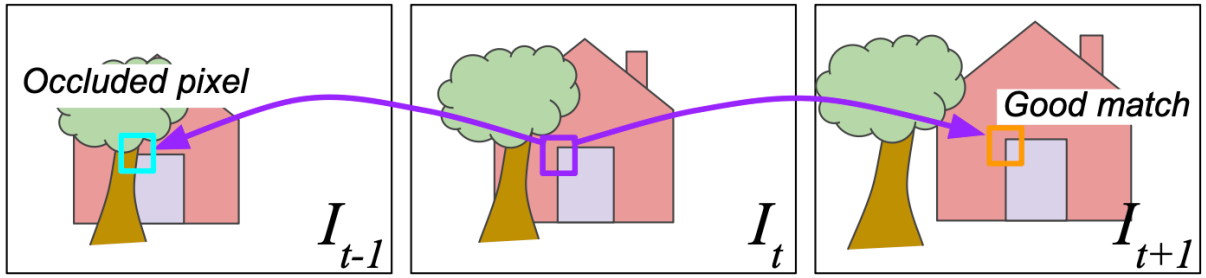


Illustration of the differentiable image warping process.
For each point p_t in the target view, we first project it onto the source view based on the predicted depth and camera pose, and then use **bilinear interpolation** to obtain the value of the warped image $I_{t'}$ at location p_t

Method

1. Per-Pixel Minimum Reprojection Loss



$pe($, $) =$ error

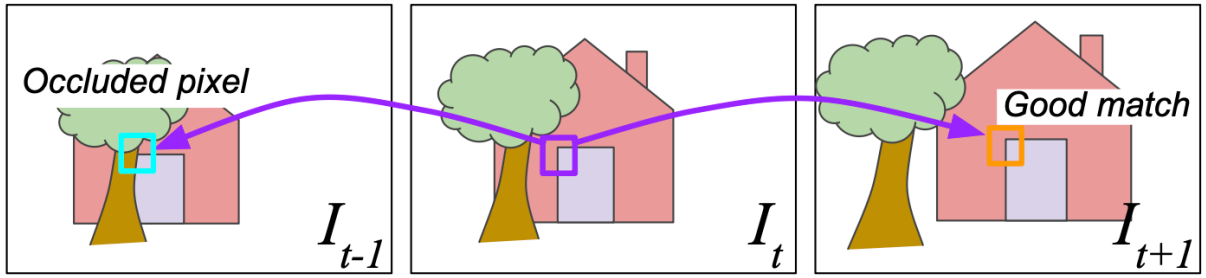
$pe($, $) =$ error

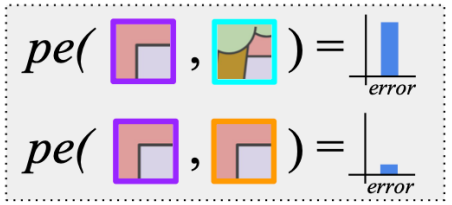
Baseline: $avg($, $) =$ ✗

Ours: $min($, $) =$ ✓

Method

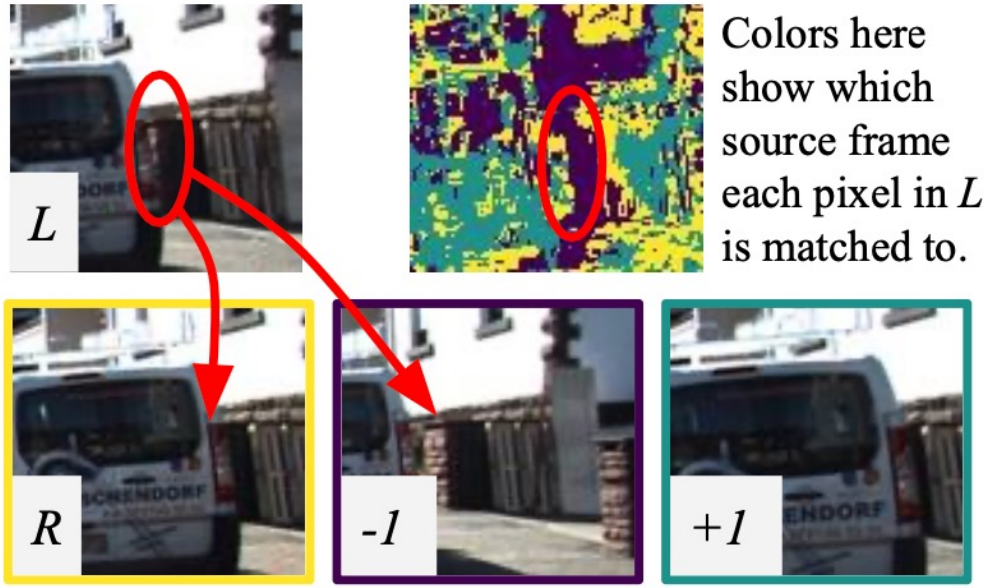
1. Per-Pixel Minimum Reprojection Loss





Baseline: $avg($, $) =$ ×

Ours: $min($, $) =$ ✓

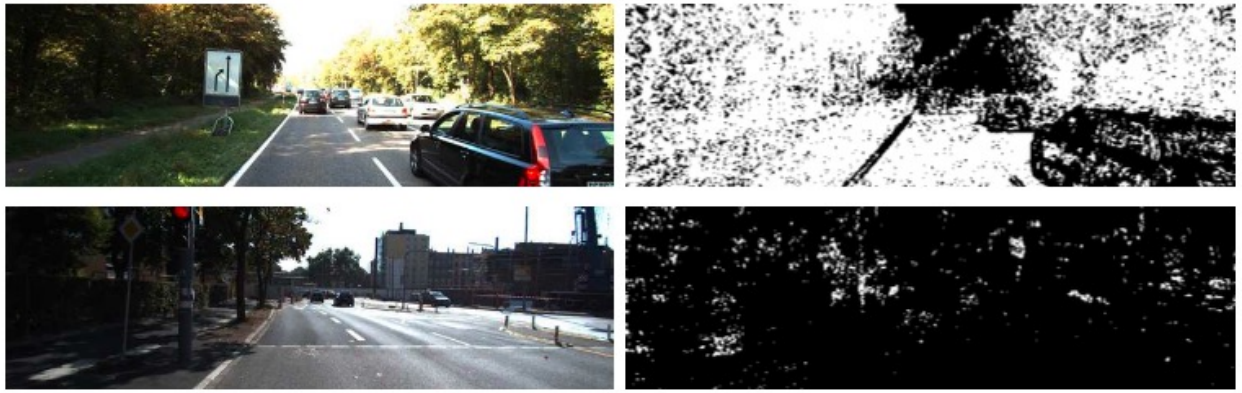


Colors here show which source frame each pixel in L is matched to.

$$L_p = \min_{t'} pe(I_t, I_{t' \to t})$$

Method

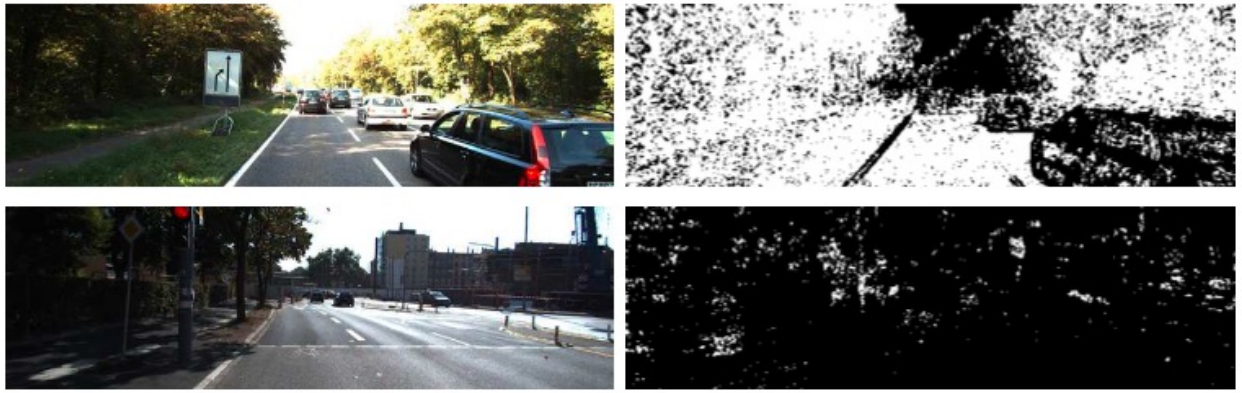
2. Auto-Masking Stationary Pixels



Black pixels are removed from the loss (*i.e.* $\mu = 0$)

Method

2. Auto-Masking Stationary Pixels



Black pixels are removed from the loss (*i.e.* μ = 0)

$$\mu = \left[\min_{t'} pe(I_t, I_{t' \to t}) < \min_{t'} pe(I_t, I_{t'}) \right]$$

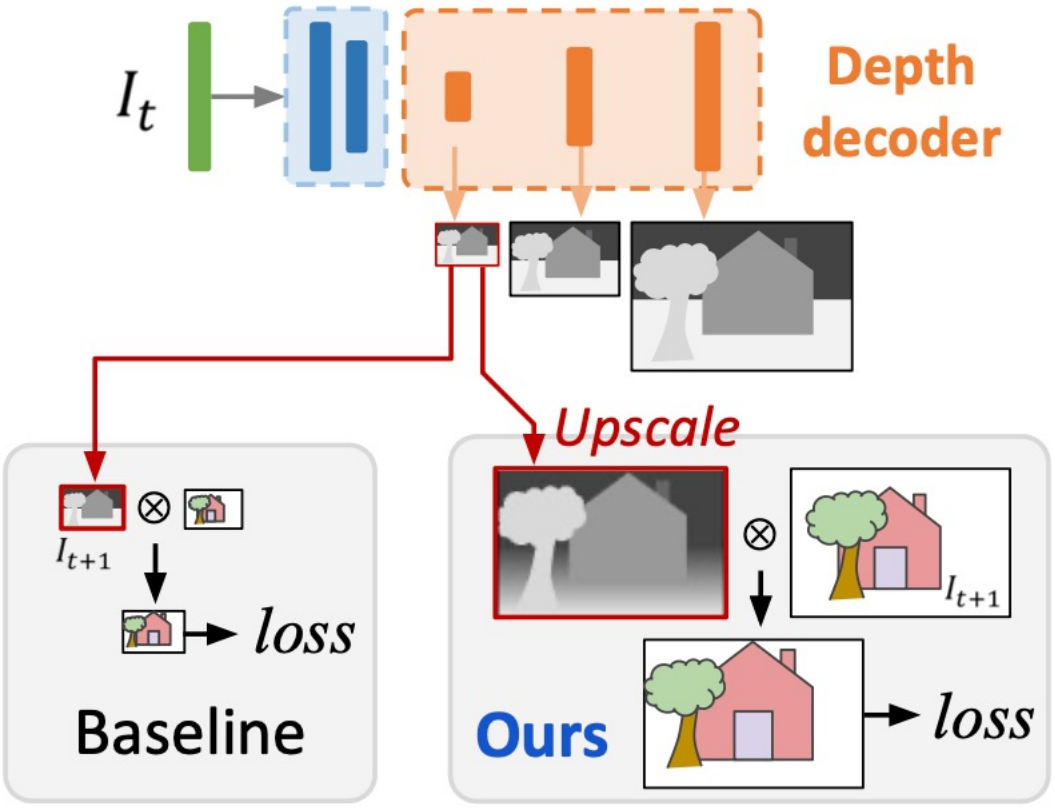
[ ] is the Iverson bracket.

$$[P] = \begin{cases} 1 & \text{if } P \text{ is true;} \\ 0 & \text{otherwise.} \end{cases}$$

This function is defined to take the value 1 for the values of the variables for which the statement is true, and takes the value 0 otherwise.

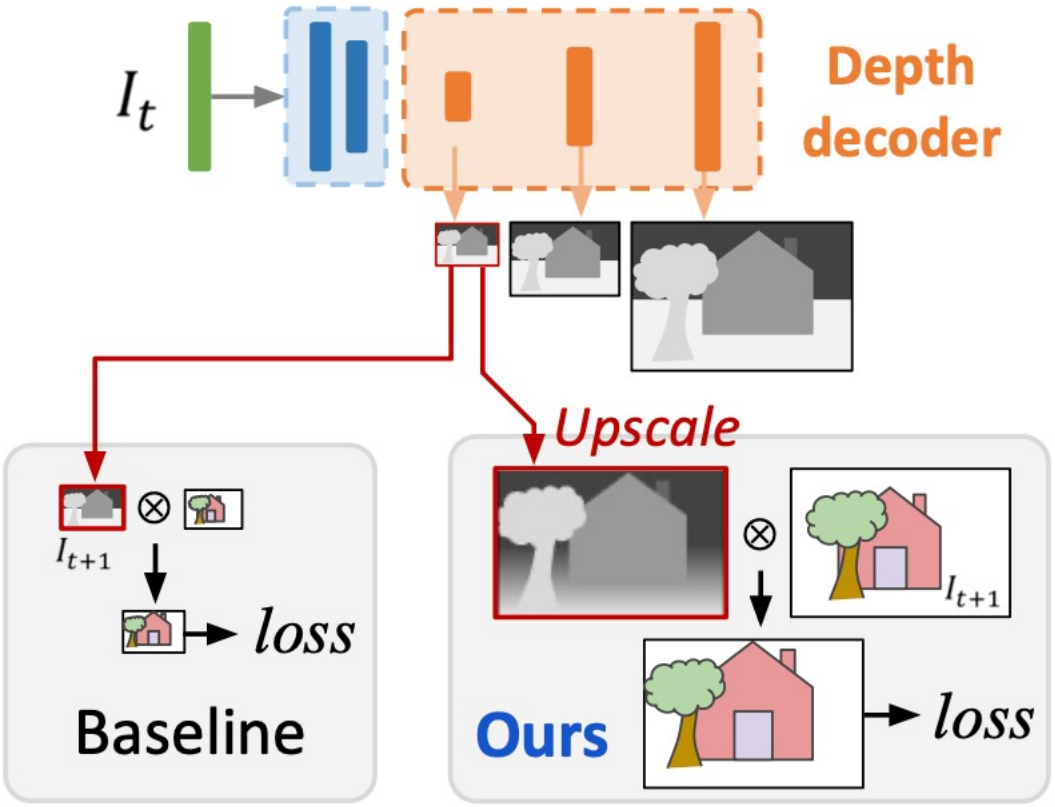
Method

3. Multi-scale Estimation



Method

3. Multi-scale Estimation

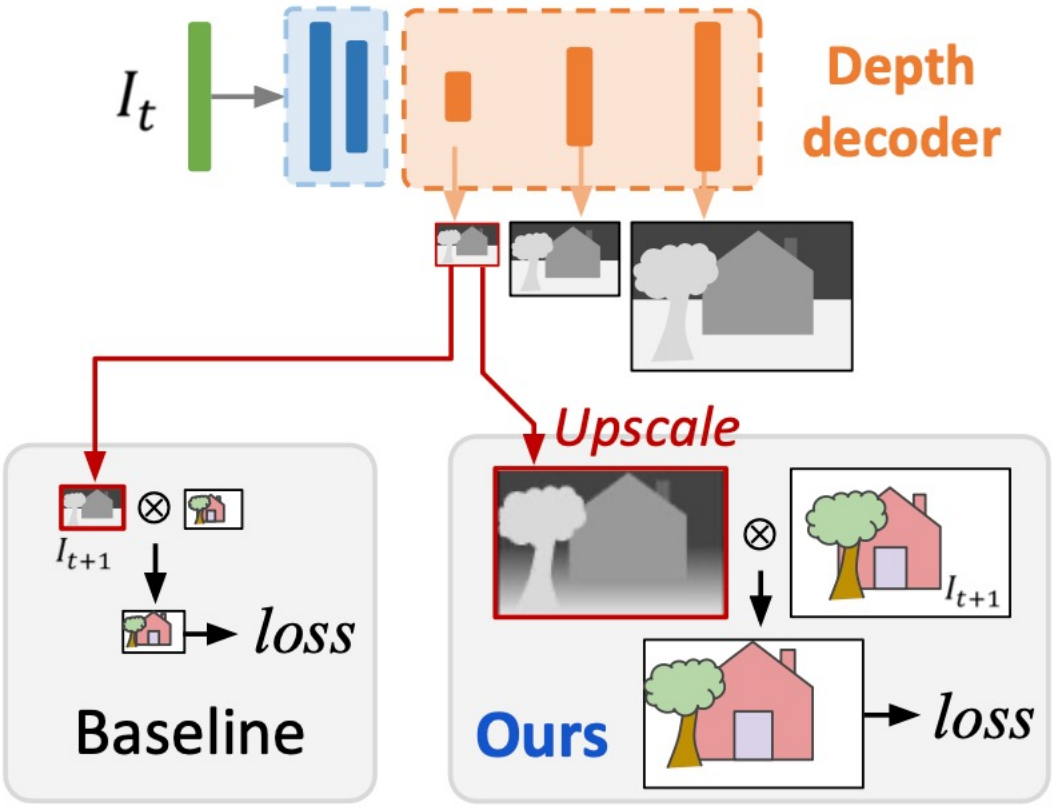


- Final Training Loss

$$L = \mu L_p + \lambda L_s$$

Method

3. Multi-scale Estimation



- Final Training Loss

$$L = \mu L_p + \lambda L_s$$

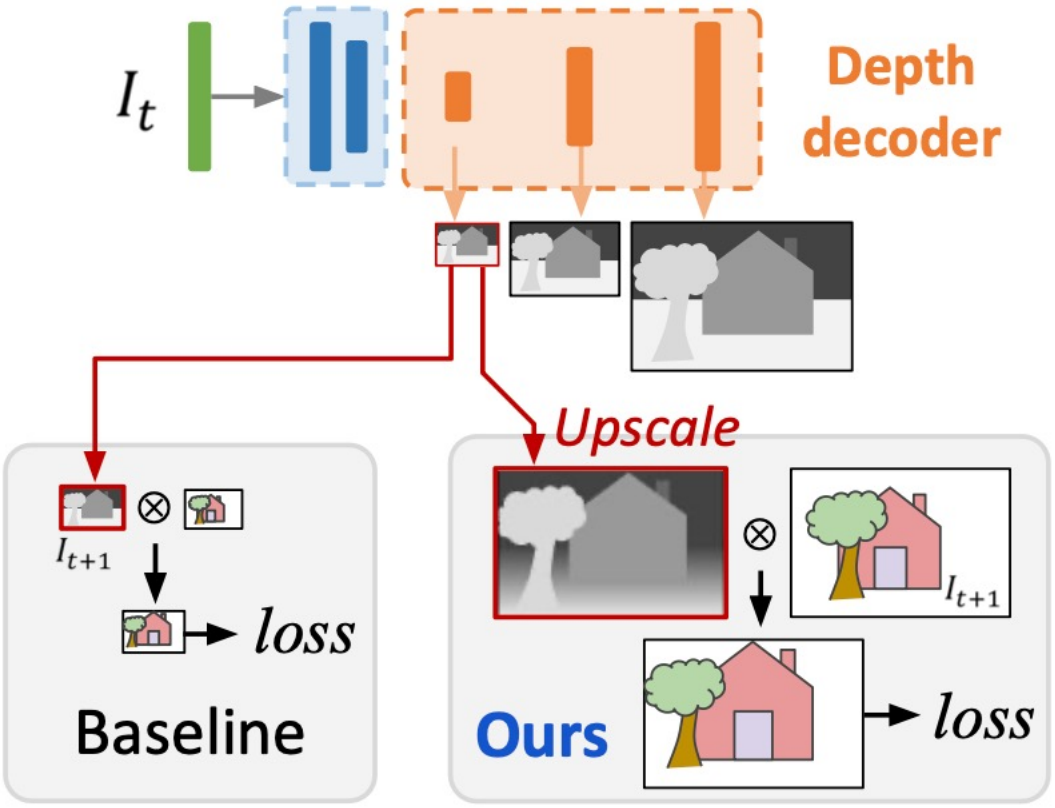
$$\mu = \left[\min_{t'} pe(I_t, I_{t' \to t}) < \min_{t'} pe(I_t, I_{t'}) \right]$$

$$L_p = \min_{t'} pe(I_t, I_{t' \to t})$$

$$pe(I_a, I_b) = \frac{\alpha}{2}(1 - \text{SSIM}(I_a, I_b)) + (1 - \alpha)\|I_a - I_b\|_1$$

Method

3. Multi-scale Estimation



- Final Training Loss

$$L = \mu L_p + \lambda L_s$$

$$\mu = \left[\min_{t'} pe(I_t, I_{t' \to t}) < \min_{t'} pe(I_t, I_{t'}) \right]$$

$$L_p = \min_{t'} pe(I_t, I_{t' \to t})$$

$$pe(I_a, I_b) = \frac{\alpha}{2}(1 - \text{SSIM}(I_a, I_b)) + (1 - \alpha)\|I_a - I_b\|_1$$

$$L_s = |\partial_x d_t^*| \, e^{-|\partial_x I_t|} + |\partial_y d_t^*| \, e^{-|\partial_y I_t|}$$

Introduction

Method

Experiments

Conclusion

Experiments

- **RMSE** $= \sqrt{\frac{1}{|N|}\sum_{i \in N} \| d_i - d_i^* \|^2}$,
- **RMSE log** $= \sqrt{\frac{1}{|N|}\sum_{i \in N} \| \log(d_i) - \log(d_i^*) \|^2}$,
- **Abs Rel** $= \frac{1}{|N|}\sum_{i \in N} \frac{|d_i - d_i^*|}{d_i^*}$,
- **Sq Rel** $= \frac{1}{|N|}\sum_{i \in N} \frac{\|d_i - d_i^*\|^2}{d_i^*}$,
- **Accuracies:** % of d_i s.t. $\max(\frac{d_i}{d_i^*}, \frac{d_i^*}{d_i}) = \delta < thr$,

Experiments

- **RMSE** $= \sqrt{\frac{1}{|N|}\sum_{i \in N} \| d_i - d_i^* \|^2}$,
- **RMSE log** $= \sqrt{\frac{1}{|N|}\sum_{i \in N} \| \log(d_i) - \log(d_i^*) \|^2}$,
- **Abs Rel** $= \frac{1}{|N|}\sum_{i \in N} \frac{|d_i - d_i^*|}{d_i^*}$,
- **Sq Rel** $= \frac{1}{|N|}\sum_{i \in N} \frac{\|d_i - d_i^*\|^2}{d_i^*}$,
- **Accuracies:** % of d_i s.t. $\max(\frac{d_i}{d_i^*}, \frac{d_i^*}{d_i}) = \delta < thr$,

Method	Train	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Eigen [9]	D	0.203	1.548	6.307	0.282	0.702	0.890	0.890
Liu [36]	D	0.201	1.584	6.471	0.273	0.680	0.898	0.967
Klodt [28]	D*M	0.166	1.490	5.998	-	0.778	0.919	0.966
AdaDepth [45]	D*	0.167	1.257	5.578	0.237	0.771	0.922	0.971
Kuznietsov [30]	DS	0.113	0.741	4.621	0.189	0.862	0.960	0.986
DVSO [68]	D*S	0.097	0.734	4.442	0.187	0.888	0.958	0.980
SVSM FT [39]	DS	0.094	0.626	4.252	0.177	0.891	0.965	0.984
Guo [16]	DS	0.096	0.641	4.095	0.168	0.892	0.967	0.986
DORN [10]	D	**0.072**	**0.307**	**2.727**	**0.120**	**0.932**	**0.984**	**0.994**
Zhou [76]†	M	0.183	1.595	6.709	0.270	0.734	0.902	0.959
Yang [70]	M	0.182	1.481	6.501	0.267	0.725	0.906	0.963
Mahjourian [40]	M	0.163	1.240	6.220	0.250	0.762	0.916	0.968
GeoNet [71]†	M	0.149	1.060	5.567	0.226	0.796	0.935	0.975
DDVO [62]	M	0.151	1.257	5.583	0.228	0.810	0.936	0.974
DF-Net [78]	M	0.150	1.124	5.507	0.223	0.806	0.933	0.973
LEGO [69]	M	0.162	1.352	6.276	0.252	-	-	-
Ranjan [51]	M	0.148	1.149	5.464	0.226	0.815	0.935	0.973
EPC++ [38]	M	0.141	1.029	5.350	0.216	0.816	0.941	0.976
Struct2depth '(M)' [5]	M	0.141	1.026	5.291	0.215	0.816	0.945	0.979
Monodepth2 w/o pretraining	M	0.132	1.044	5.142	0.210	0.845	0.948	0.977
Monodepth2	M	**0.115**	**0.903**	**4.863**	**0.193**	**0.877**	**0.959**	**0.981**
Monodepth2 (1024 × 320)	M	**0.115**	**0.882**	**4.701**	**0.190**	**0.879**	**0.961**	**0.982**
Garg [12]†	S	0.152	1.226	5.849	0.246	0.784	0.921	0.967
Monodepth R50 [15]†	S	0.133	1.142	5.533	0.230	0.830	0.936	0.970
StrAT [43]	S	0.128	1.019	5.403	0.227	0.827	0.935	0.971
3Net (R50) [50]	S	0.129	0.996	5.281	0.223	0.831	0.939	0.974
3Net (VGG) [50]	S	0.119	1.201	5.888	0.208	0.844	0.941	**0.978**
SuperDepth + pp [47] (1024 × 382)	S	0.112	0.875	**4.958**	**0.207**	0.852	0.947	0.977
Monodepth2 w/o pretraining	S	0.130	1.144	5.485	0.232	0.831	0.932	0.968
Monodepth2	S	**0.109**	**0.873**	4.960	0.209	**0.864**	**0.948**	0.975
Monodepth2 (1024 × 320)	S	**0.107**	**0.849**	**4.764**	**0.201**	**0.874**	**0.953**	0.977
UnDeepVO [33]	MS	0.183	1.730	6.57	0.268	-	-	-
Zhan FullNYU [73]	D*MS	0.135	1.132	5.585	0.229	0.820	0.933	0.971
EPC++ [38]	MS	0.128	0.935	5.011	0.209	0.831	0.945	**0.979**
Monodepth2 w/o pretraining	MS	0.127	1.031	5.266	0.221	0.836	0.943	0.974
Monodepth2	MS	**0.106**	**0.818**	**4.750**	**0.196**	**0.874**	**0.957**	**0.979**
Monodepth2 (1024 × 320)	MS	**0.106**	**0.806**	**4.630**	**0.193**	**0.876**	**0.958**	**0.980**

Table 1. **Quantitative results.** Comparison of our method to existing methods on KITTI 2015 [13] using the Eigen split. Best results in each category are in **bold**; second best are underlined.

All results here are presented without post-processing [15]; see supplementary Section F for improved post-processed results. While our contributions are designed for monocular training, we still gain high accuracy in the stereo-only category.

We additionally show we can get higher scores at a larger 1024 × 320 resolution, similar to [47] – see supplementary Section G. These high resolution numbers are bolded if they beat all other models, including our low-res versions.

Legend

D – Depth supervision
D* – Auxiliary depth supervision
S – Self-supervised stereo supervision
M – Self-supervised mono supervision
† – Newer results from github.
+ pp – With post-processing

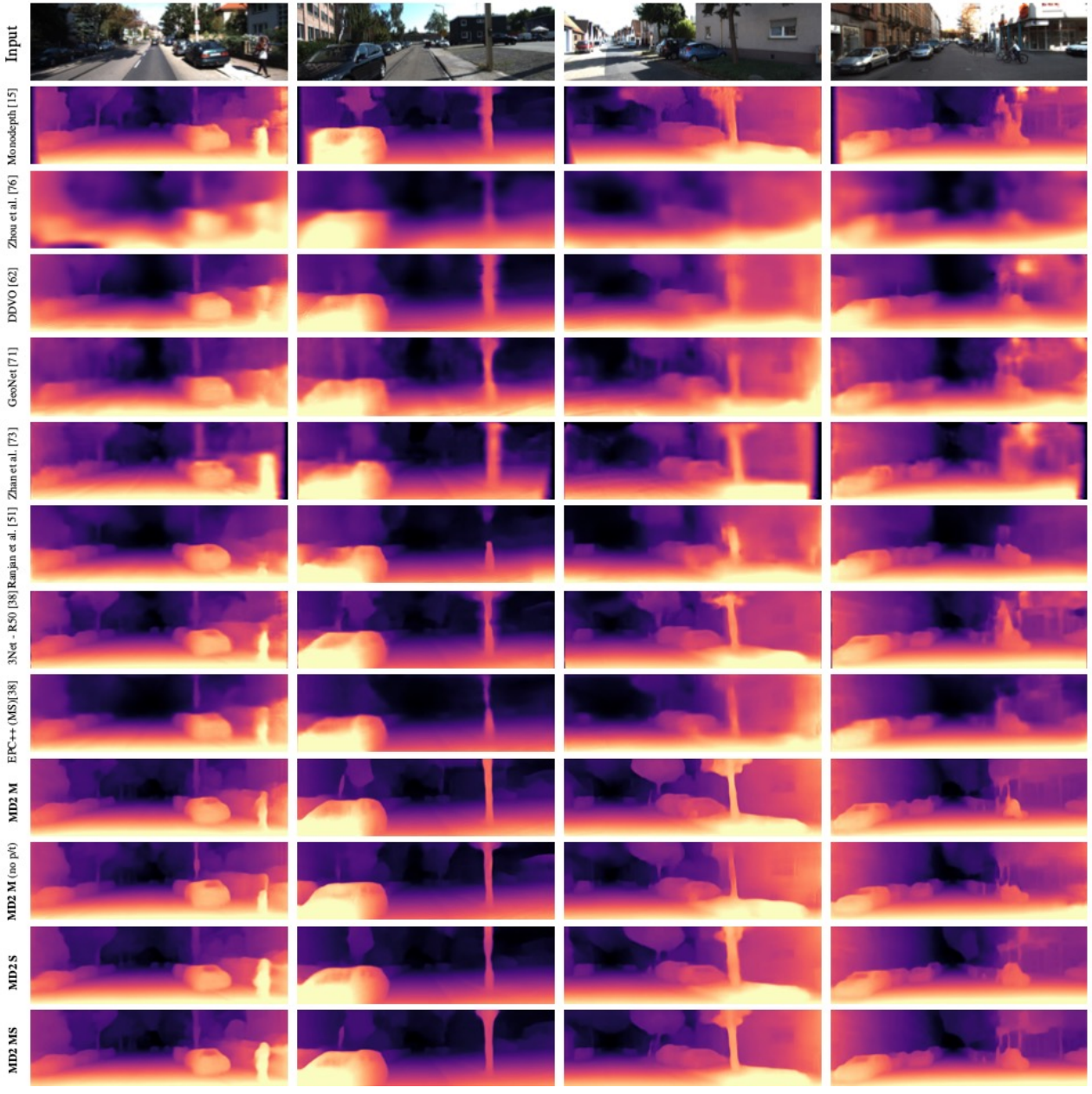
Experiments

- **RMSE** $= \sqrt{\frac{1}{|N|}\sum_{i \in N} \| d_i - d_i^* \|^2}$,
- **RMSE log** $= \sqrt{\frac{1}{|N|}\sum_{i \in N} \| \log(d_i) - \log(d_i^*) \|^2}$,
- **Abs Rel** $= \frac{1}{|N|}\sum_{i \in N} \frac{|d_i - d_i^*|}{d_i^*}$,
- **Sq Rel** $= \frac{1}{|N|}\sum_{i \in N} \frac{\|d_i - d_i^*\|^2}{d_i^*}$,
- **Accuracies:** % of d_i s.t. $\max(\frac{d_i}{d_i^*}, \frac{d_i^*}{d_i}) = \delta < thr$,

		Auto-masking	Min. reproj.	Full-res multi-scale	Pretrained	Full Eigen split	Abs Rel	Sq Rel	RMSE	RMSE log	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
(a)	Baseline				✓		0.140	1.610	5.512	0.223	0.852	0.946	0.973
	Baseline + min reproj.		✓		✓		0.122	1.081	5.116	0.199	0.866	0.957	0.980
	Baseline + automasking	✓					0.124	0.936	5.010	0.206	0.858	0.952	0.977
	Baseline + full-res m.s.			✓	✓		0.124	1.170	5.249	0.203	0.865	0.953	0.978
	Monodepth2 w/o min reprojection	✓		✓	✓		0.117	0.878	4.846	0.196	0.870	0.957	0.980
	Monodepth2 w/o auto-masking		✓	✓	✓		0.120	1.097	5.074	0.197	0.872	0.956	0.979
	Monodepth2 w/o full-res m.s.	✓	✓		✓		0.117	**0.866**	4.864	0.196	0.871	0.957	**0.981**
	Monodepth2 with [76]'s mask		✓	✓	✓		0.123	1.177	5.210	0.200	0.869	0.955	0.978
	Monodepth2 smaller (416 × 128)	✓	✓	✓	✓		0.128	1.087	5.171	0.204	0.855	0.953	0.978
	Monodepth2 (full)	✓	✓	✓	✓		**0.115**	0.903	**4.863**	**0.193**	**0.877**	**0.959**	**0.981**
(b)	Baseline w/o pt						0.150	1.585	5.671	0.234	0.827	0.938	0.971
	Monodepth2 w/o pt	✓	✓	✓			**0.132**	**1.044**	**5.142**	**0.210**	**0.845**	**0.948**	**0.977**
(c)	Baseline (full Eigen dataset)				✓	✓	0.146	1.876	5.666	0.230	0.848	0.945	0.972
	Monodepth2 (full Eigen dataset)	✓	✓	✓	✓	✓	**0.116**	**0.918**	**4.872**	**0.193**	**0.874**	**0.959**	**0.981**

Table 2. **Ablation.** Results for different variants of our model (**Monodepth2**) with monocular training on KITTI 2015 [13] using the Eigen split. **(a)** The baseline model, with none of our contributions, performs poorly. The addition of our minimum reprojection, auto-masking and full-res multi-scale components, significantly improves performance. **(b)** Even without ImageNet pretrained weights, our much simpler model brings large improvements above the baseline – see also Table 1. **(c)** If we train with the full Eigen dataset (instead of the subset introduced for monocular training by [76]) our improvement over the baseline increases.

Experiments



Introduction

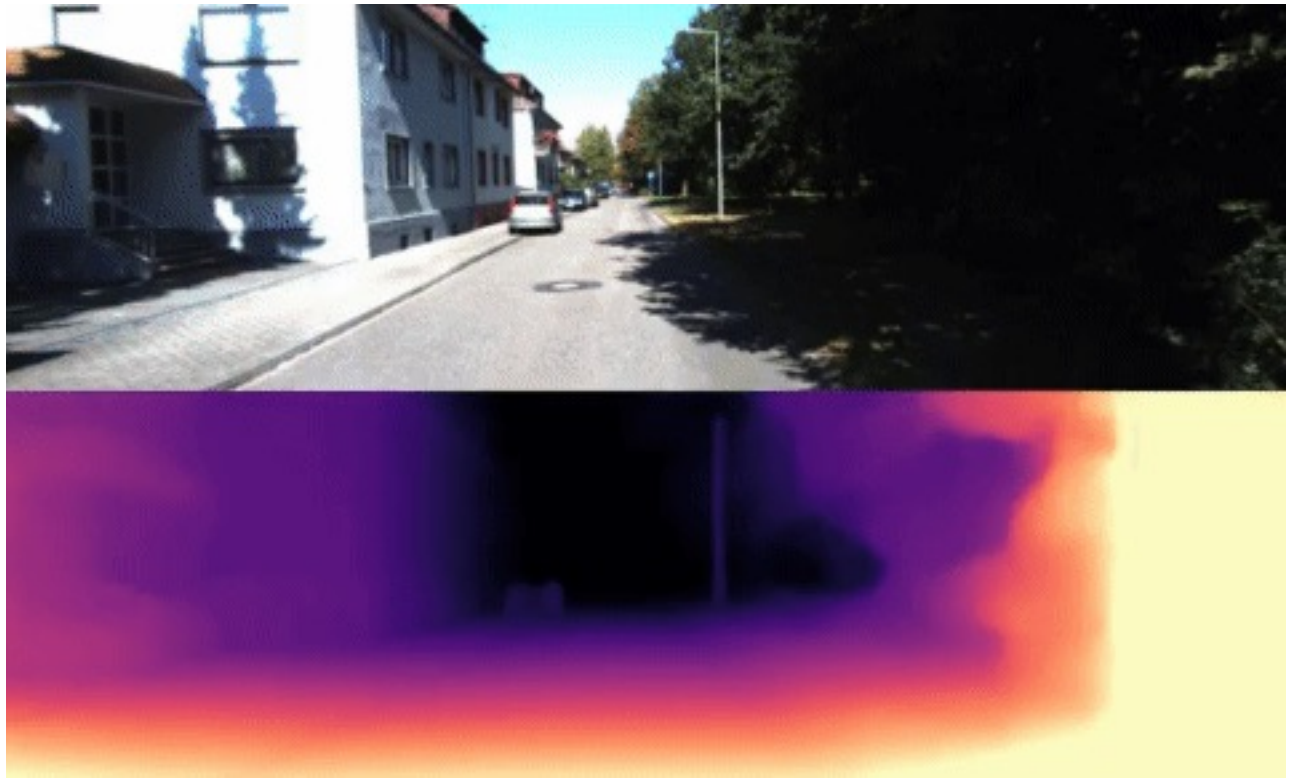
Method

Experiments

Conclusion

Conclusion

- Self-supervised Monocular Depth Estimation



Q & A