Overview of Depth Estimation from Single Image

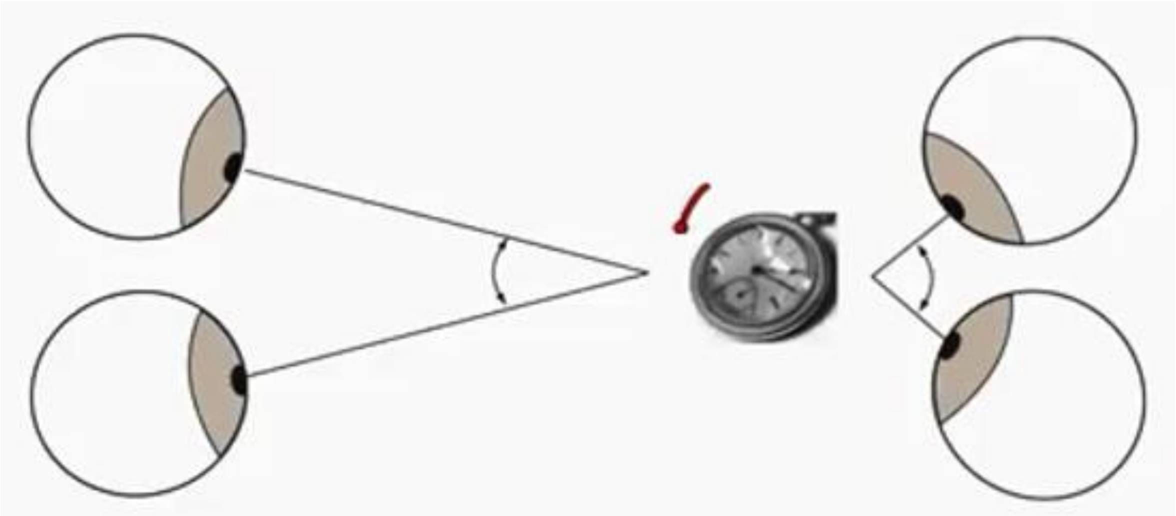
刘环宇 2017/05/12

Outline

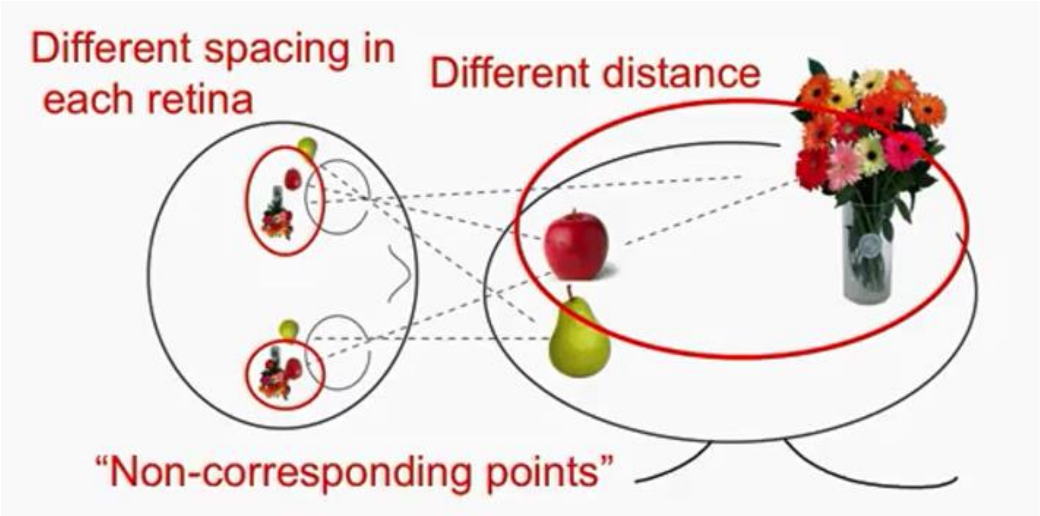
- Background
- Research Methods
- Future work

Background

Convergence angle



Stereo vision



Background

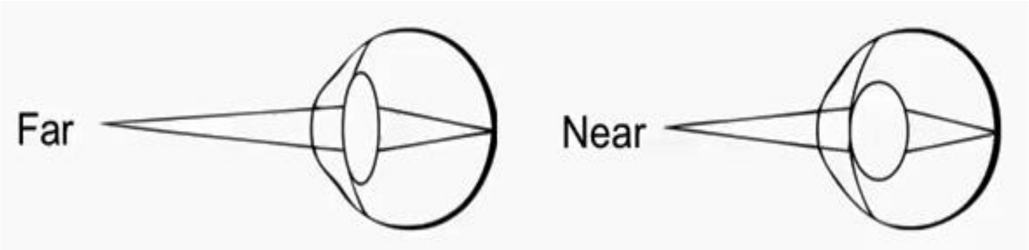
The Blank Check Rene Magritte
Occlusion Blur/Haze Size

Linear perspective

Perceived size and
perceived distance interact

Background

Focus/Accommodation



Blur/Haze

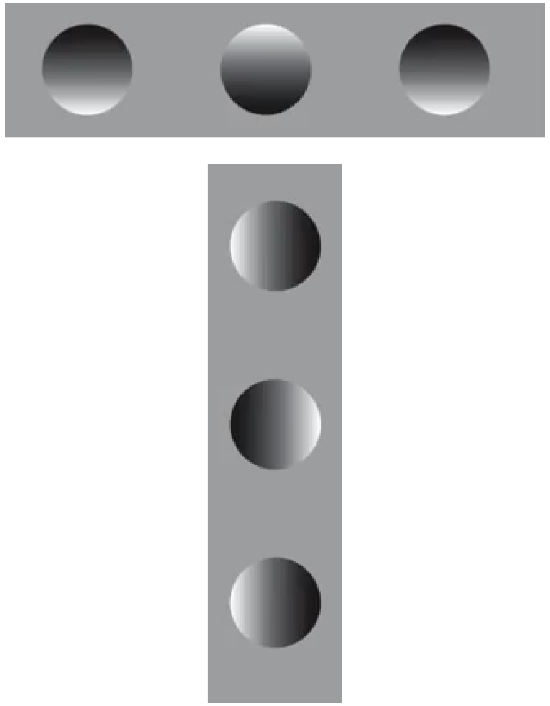
Both involve image clarity

Blur/Haze also involves color

Focus/Accommodation involves “knowing” the lens “setting”
- what did you need to do to focus the image

Background

Shape from Shading



Motion Parallax

Background Summary

Binocular Distance Cues

- Convergence angle
- Stereovision

Monocular Distance Cues

- Occlusion
- Relative size
- Blur/haze
- Linear perspective
- Focus/accommodation
- Shape from shading
- Motion parallax

Research Methods

Direct regression

- **Depth Map Prediction from a Single Image using a Multi-Scale Deep Network_NIPS2014**
- Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-Scale Convolutional Architecture_ICCV2015
- Deeper Depth Prediction with Fully Convolutional Residual Networks_3DV2016
- Joint Semantic Segmentation and Depth Estimation with Deep Convolutional Networks_3DV2016

Constraint regression

- **Deep Convolutional Neural Fields for Depth Estimation from a Single Image_PAMI2015**（ **Constraint in Loss function**）
- **Indoor Scene Structure Analysis for Single Image Depth Estimation_CVPR2015**（**solve constrained optimization problems**）
- Direction Matters - Depth Estimation with a Surface Normal Classifier_CVPR2015（**regularization**）
- Depth and surface normal estimation from monocular images using regression on deep features and hierarchical CRFs（**regularization**）

Ordinal Relationships：

- **Learning ordinal relationships for mid-level vision_ICCV2015**（**solve constrained optimization problems**）
- **Single-Image Depth Perception in the Wild_NIPS2016**（ **constraint regression** ）

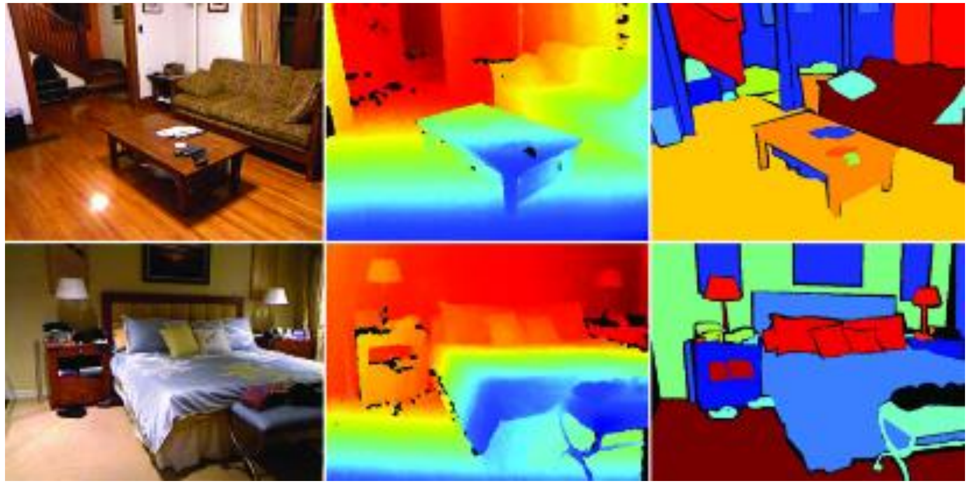
Depth dataset：

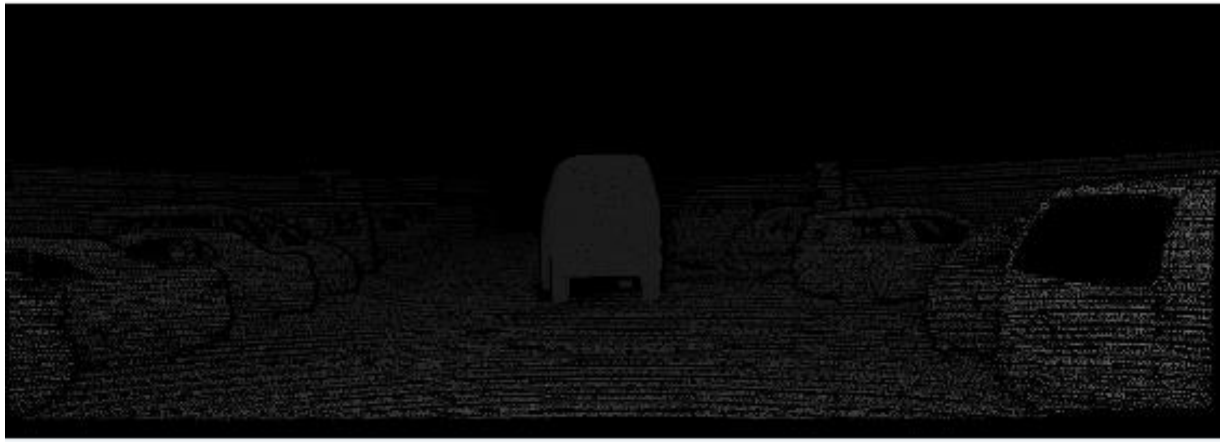
NYU Depth dataset(Indoor)：

- 1449 densely labeled pairs of aligned RGB and depth images
- 464 new scenes taken from 3 cities
- 407,024 new unlabeled frames

The KITTI dataset(Outdoor)：

- use 56 scenes from the “city,” “residential,” and “road” categories of the raw data
- 800 images per scene

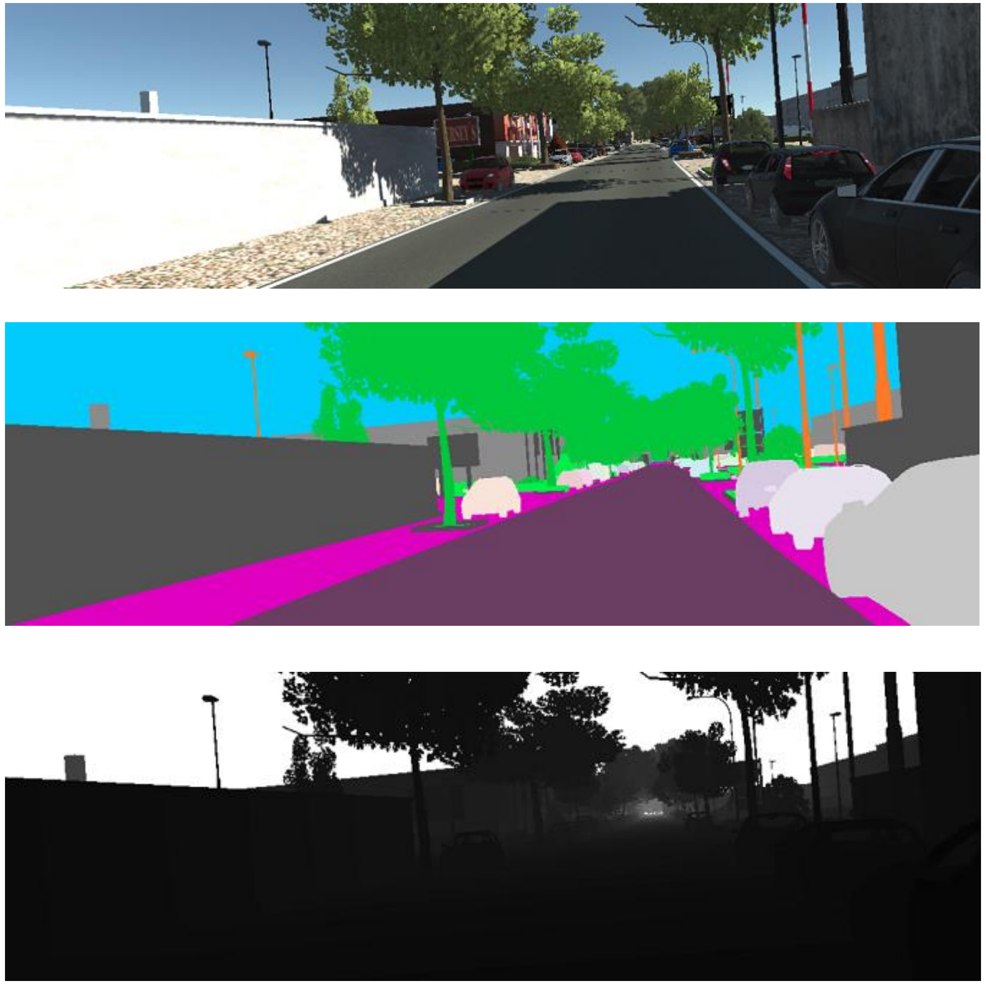




Depth dataset：

Virtual KITTI： 21,260 frames

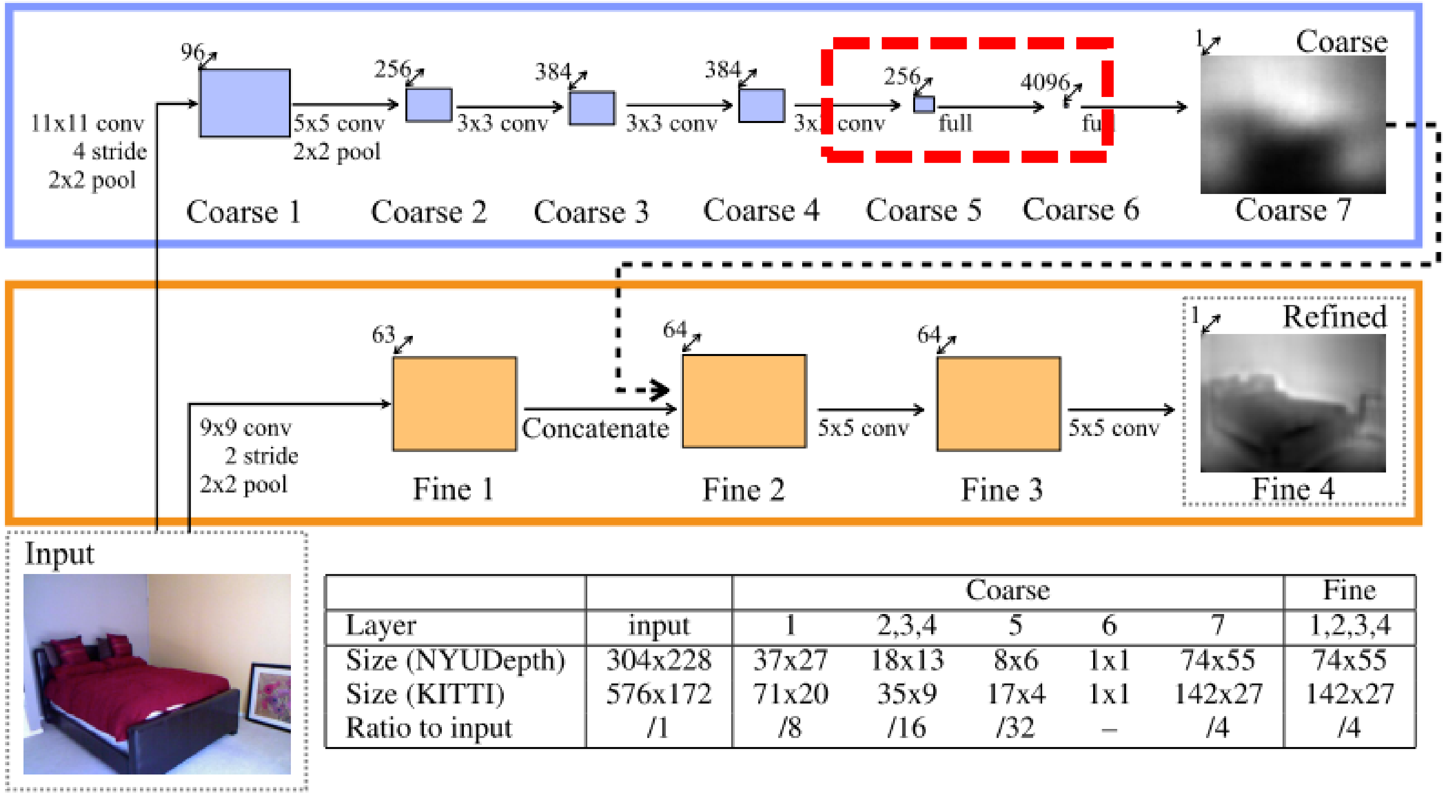
- photo-realistic synthetic video dataset
- object detection and multi-object tracking
- scene-level and instance-level semantic segmentation
- optical flow, and depth estimation



Scale-Invariant Error: measure the **relationships between points** in the scene, **irrespective of the absolute global scale**

$$D(y,y^*) = \frac{1}{2n^2}\sum_{i,j}\left((\log y_i - \log y_j) - (\log y_i^* - \log y_j^*)\right)^2$$
$$= \frac{1}{n}\sum_i d_i^2 - \frac{1}{n^2}\sum_{i,j} d_i d_j = \frac{1}{n}\sum_i d_i^2 - \frac{1}{n^2}\left(\sum_i d_i\right)^2$$

$$L(y,y^*) = \frac{1}{n}\sum_i d_i^2 - \frac{\lambda}{n^2}\left(\sum_i d_i\right)^2$$



		Coarse					Fine
Layer	input	1	2,3,4	5	6	7	1,2,3,4
Size (NYUDepth)	304x228	37x27	18x13	8x6	1x1	74x55	74x55
Size (KITTI)	576x172	71x20	35x9	17x4	1x1	142x27	142x27
Ratio to input	/1	/8	/16	/32	–	/4	/4

Figure 1: Model architecture.

Train process:

- First, **train the coarse network** for 2M samples using SGD with batches of size 32
- Then, **hold it fixed and train the fine network** for 1.5M samples

Time costs:

- Training took 38h for the coarse network and 26h for fine, for a **total of 2.6 days** using a NVidia GTX Titan Black.
- Test prediction takes **0.01s/image**

Threshold: % of y_i s.t. $\max(\frac{y_i}{y_i^*}, \frac{y_i^*}{y_i}) = \delta < thr$

Abs Relative difference: $\frac{1}{|T|}\sum_{y\in T}|y - y^*|/y^*$

Squared Relative difference: $\frac{1}{|T|}\sum_{y\in T}||y - y^*||^2/y^*$

RMSE (linear): $\sqrt{\frac{1}{|T|}\sum_{y\in T}||y_i - y_i^*||^2}$

RMSE (log): $\sqrt{\frac{1}{|T|}\sum_{y\in T}||\log y_i - \log y_i^*||^2}$

RMSE (log, scale-invariant): The error Eqn. 1

(root-mean-square error)

	Mean	Make3D	Coarse	Coarse + Fine	
threshold $\delta < 1.25$	0.556	0.601	0.679	**0.692**	higher is better
threshold $\delta < 1.25^2$	0.752	0.820	0.897	**0.899**	
threshold $\delta < 1.25^3$	0.870	0.926	**0.967**	**0.967**	
abs relative difference	0.412	0.280	0.194	**0.190**	lower is better
sqr relative difference	5.712	3.012	1.531	**1.515**	
RMSE (linear)	9.635	8.734	7.216	**7.156**	
RMSE (log)	0.444	0.361	0.273	**0.270**	
RMSE (log, scale inv.)	0.359	0.327	0.248	**0.246**	

Table 2: Comparison on the KITTI dataset.

	Mean	Make3D	Ladicky&al	Karsch&al	Coarse	Coarse + Fine	
threshold $\delta < 1.25$	0.418	0.447	0.542	–	**0.618**	0.611	higher is better
threshold $\delta < 1.25^2$	0.711	0.745	0.829	–	**0.891**	0.887	
threshold $\delta < 1.25^3$	0.874	0.897	0.940	–	0.969	**0.971**	
abs relative difference	0.408	0.349	–	0.350	0.228	**0.215**	lower is better
sqr relative difference	0.581	0.492	–	–	0.223	**0.212**	
RMSE (linear)	1.244	1.214	–	1.2	**0.871**	0.907	
RMSE (log)	0.430	0.409	–	–	**0.283**	0.285	
RMSE (log, scale inv.)	0.304	0.325	–	–	0.221	**0.219**	

Table 1: Comparison on the NYUDepth dataset

- Dataset **Mean image error is low**
- Coarse and Fine **difference is small**

Train and Test:

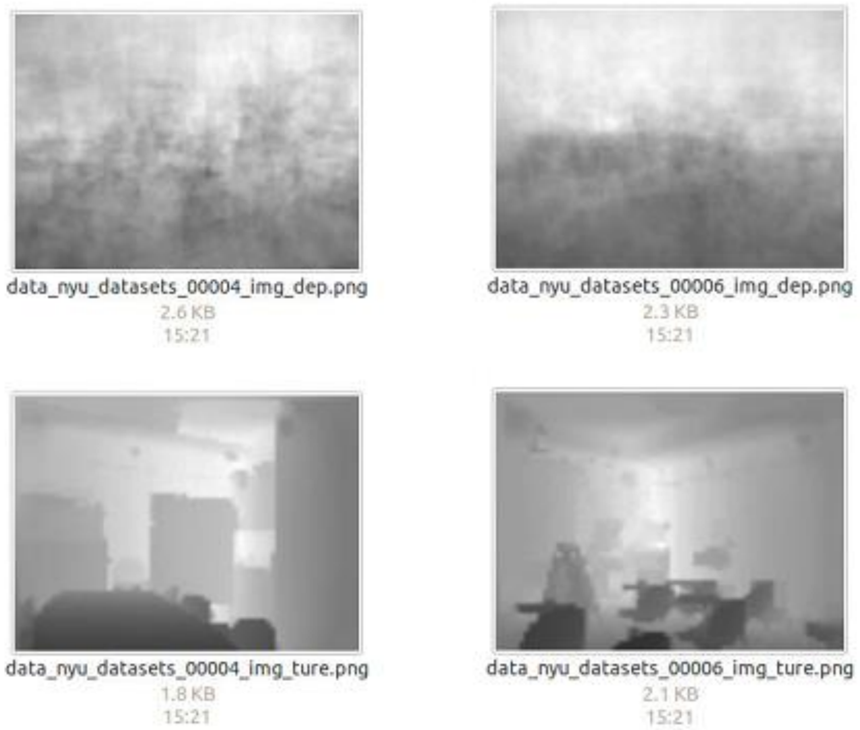
threshold $\delta < 1.25$	Coarse Accuracy	Refine Accuracy
Train Dataset： Café、 classrooms dataset： 2000 images Test Dataset： 190 images	0.878	0.414
Train Dataset： Café, classrooms, furniture, home, playrooms, reception, studies, study rooms， 5814 images（Training Long） Test Dataset： 600 images	0.895	

Conclusion：

- Network has being **overfit with the scene** and perform bad in other untrained scene
- Network has **poor explanatory**
- Network often get **wrong estimation at margin**

Conclusion：

- Network has being **overfit with the scene** and perform bad in other untrained scene
- Network has **poor explanatory**
- Network often get **wrong estimation at margin**



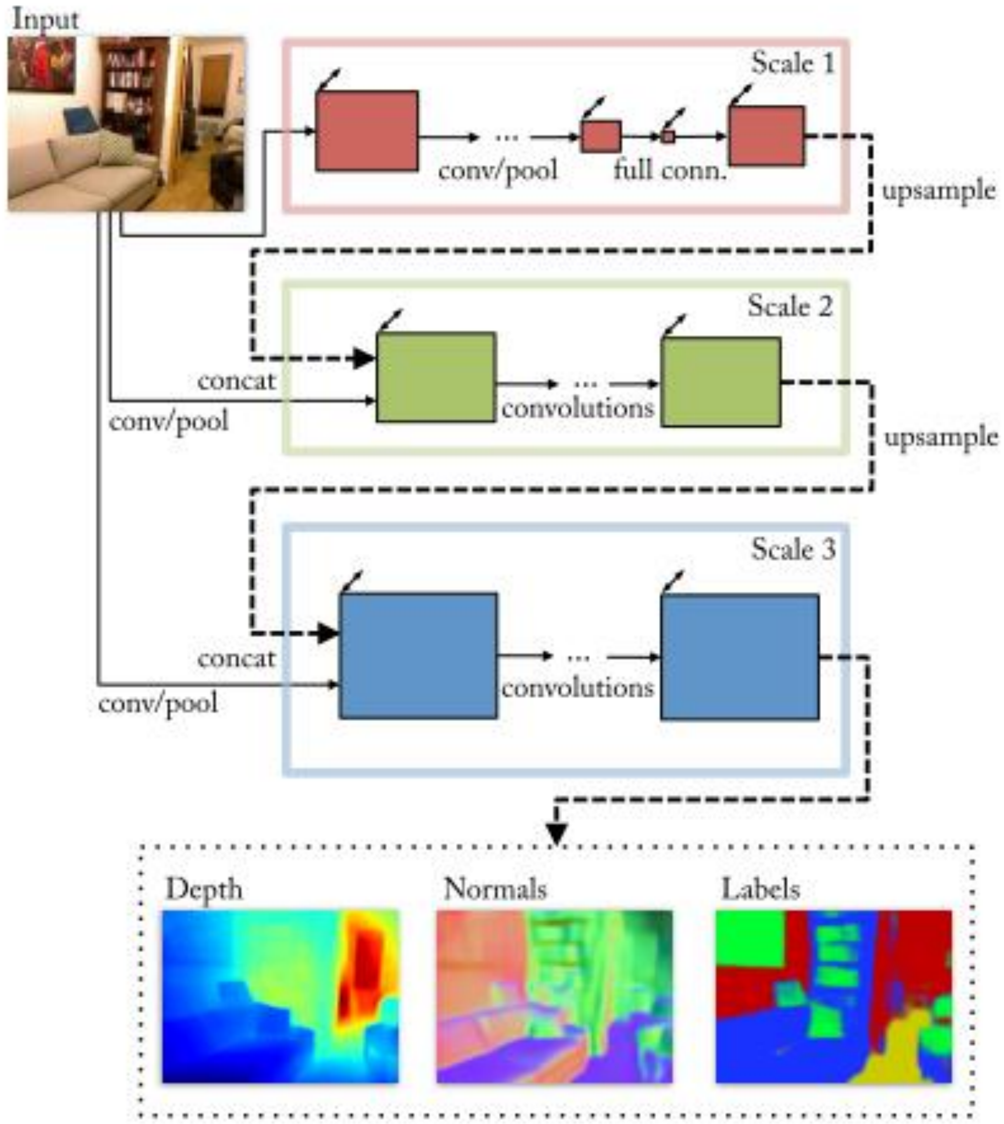
perform bad in other untrained scene

perform good in trained scene

Conclusion：

- Network has being **overfit with the scene** and perform bad in other untrained scene
- Network has **poor explanatory**
- Network often get **wrong estimation at margin**

Large Improvement as the network grow



Depth Prediction	Ladicky[20]	Karsch[18]	Baig [1]	Liu [23]	Eigen[8]	Ours(A)	Ours(VGG)
$\delta < 1.25$	0.542	–	0.597	0.614	0.614	0.697	**0.769**
$\delta < 1.25^2$	0.829	–	–	0.883	0.888	0.912	**0.950**
$\delta < 1.25^3$	0.940	–	–	0.971	0.972	0.977	**0.988**
abs rel	–	0.350	0.259	0.230	0.214	0.198	**0.158**
sqr rel	–	–	–	–	0.204	0.180	**0.121**
RMS(lin)	–	1.2	0.839	0.824	0.877	0.753	**0.641**
RMS(log)	–	–	–	–	0.283	0.255	**0.214**
sc-inv.	–	–	0.242	–	0.219	0.202	**0.171**

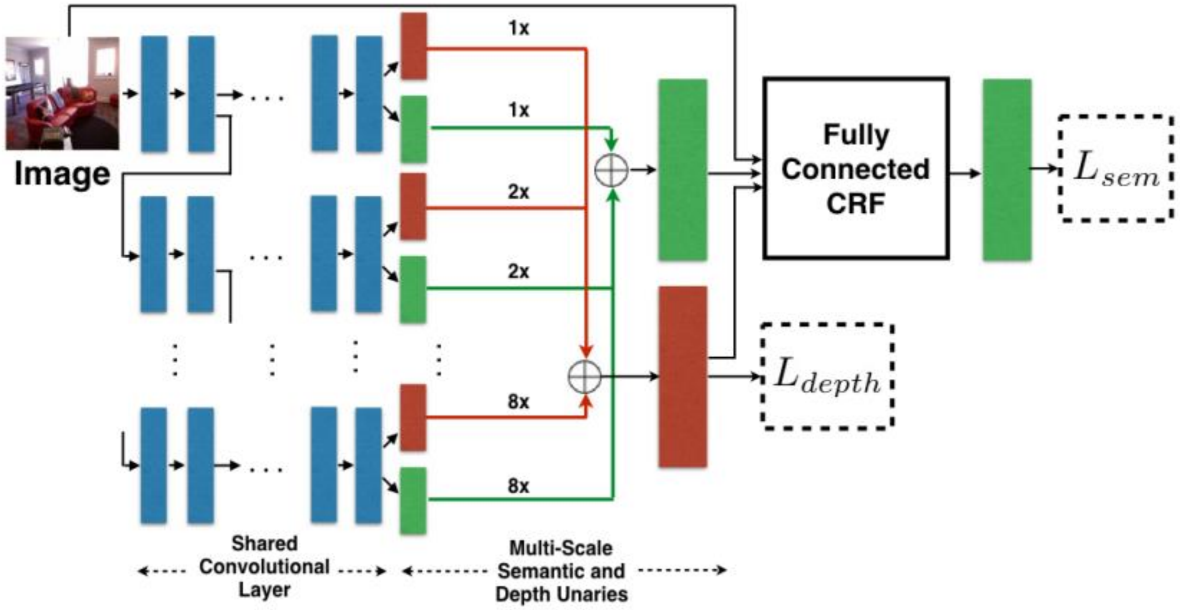
Table 1. Depth estimation measurements. Note higher is better for top rows of the table, while lower is better for the bottom section.

Pascal VOC Semantic Segmentation	2011 Validation				2011 Test	2012 Test
	Pix. Acc.	Per-Cls Acc.	Freq.Jacc	Av.Jacc	Av.Jacc	Av.Jacc
Dai *&al.*[7]	–	–	–	–	–	61.8
Long *&al.*[24]	90.3	75.9	83.2	62.7	**62.7**	62.2
Chen *&al.*[5]	–	–	–	–	–	**71.6**
Ours (VGG)	90.3	72.4	82.9	62.2	62.5	62.6

Table 5. Semantic labeling on Pascal VOC 2011 and 2012.

Table 1. Details of multi-scale network for computing depth and semantic unaries. Dimensions of each layer shown in the number of output channels and the kernel size.

Branch	Input								
Branch1	RGB	conv1-1 64x3x3	conv1-2 64x3x3	conv1-seg 40x1x1	conv1-depth 50x1x1				
Branch2	RGB	conv2-1 64x3x3	conv2-2 64x3x3	pool2 64x3x3	conv2-3 128x3x3	conv2-seg 40x1x1	conv2-depth 50x1x1	upsample x2	
Branch3	pool2	conv3-1 128x3x3	conv3-2 128x3x3	pool3 128x3x3	conv3-3 128x3x3	conv3-4 128x3x3	conv3-seg 40x1x1	conv3-depth 50x1x1	upsample x4
Branch4	pool3	conv4-1 256x3x3	conv4-2 256x3x3	pool4 256x3x3	conv4-3 128x3x3	conv4-4 128x3x3	conv4-seg 40x1x1	conv4-depth 50x1x1	upsample x4
Branch5	pool4	conv5-1 512x3x3	conv5-2 512x3x3	pool5 512x3x3	conv5-3 1024x3x3	conv5-4 1024x1x1	conv5-seg 40x1x1	conv5-depth 50x1x1	upsample x8

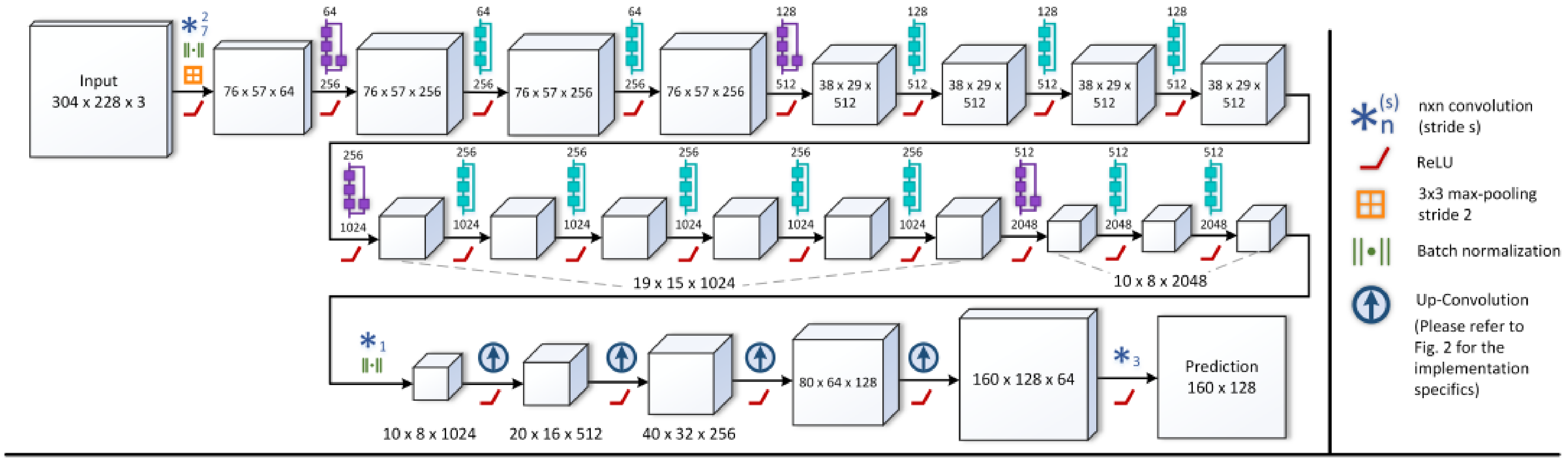


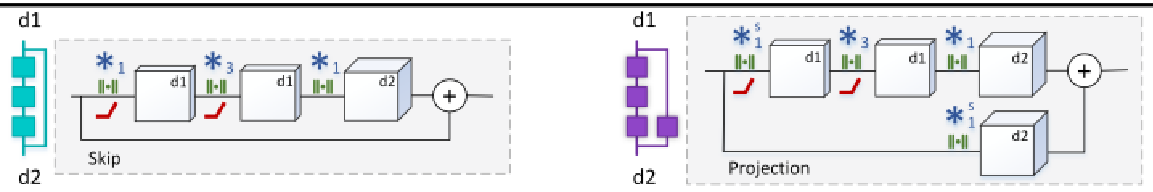
Perference is not so good

Table 2. Quantitative Evaluation of Depth Estimation

	Eigen et al.[7]	Liu et al [15]	Ours	
threshold $\delta < 1.25$	**0.769**	0.614	0.568	higher
threshold $\delta < 1.25^2$	**0.950**	0.883	0.856	is
threshold $\delta < 1.25^3$	**0.988**	0.971	0.956	better
abs relative distance	**0.158**	0.230	0.200	
sqr relative distance	**0.121**	-	0.301	lower
RMSE (linear)	**0.641**	0.824	0.816	is
RMSE (log)	**0.214**	-	0.314	better
RMSE (log. scale invariant)	0.171	-	**0.061**	

Joint Semantic Segmentation and Depth Estimation with Deep Convolutional Networks_3DV2016

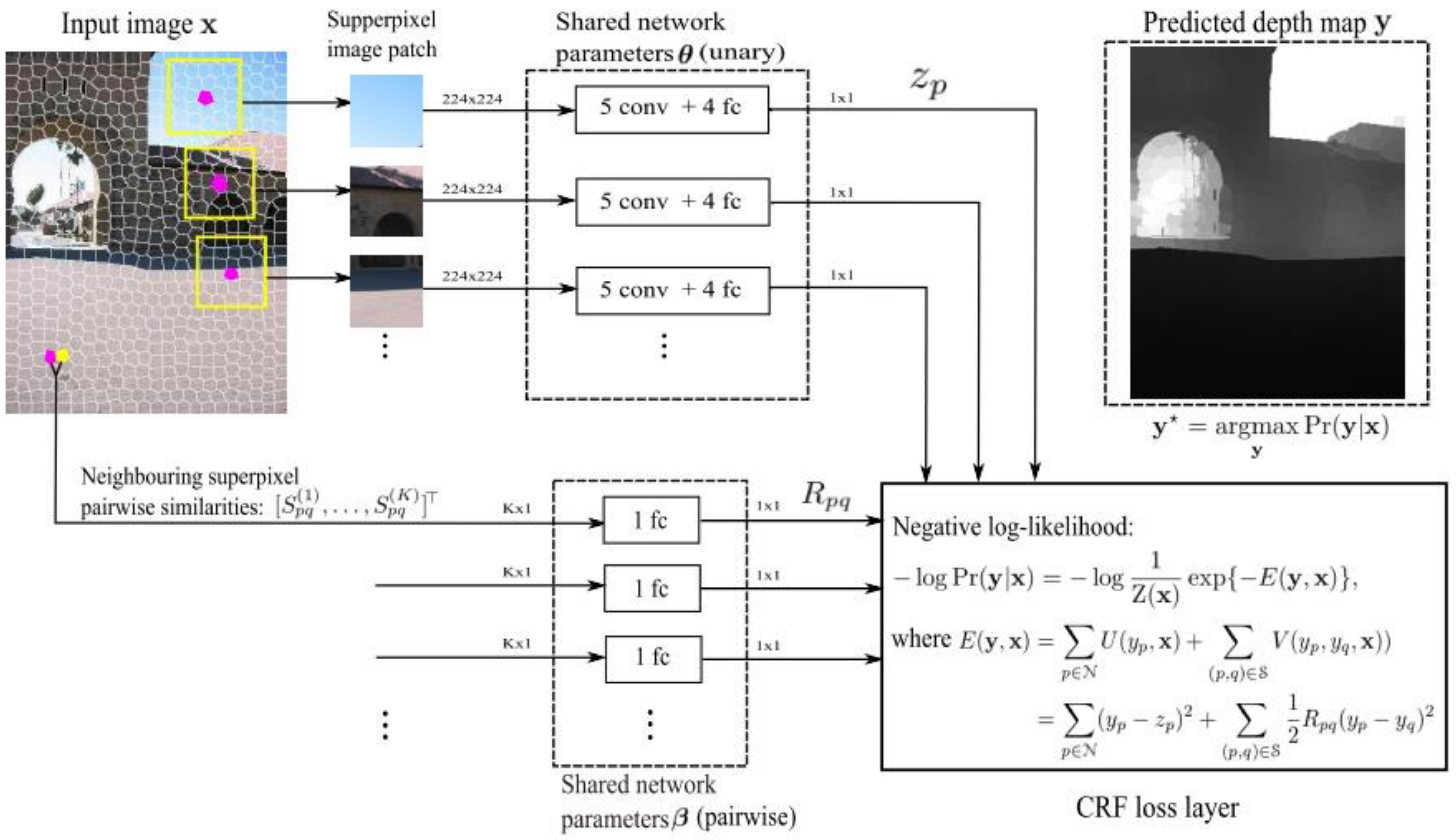




NYU Depth v2	rel	rms	rms(log)	$\log_{10}$	δ_1	δ_2	δ_3
Karsch *et al.* [10]	0.374	1.12	-	0.134	-	-	-
Ladicky *et al.* [15]	-	-	-	-	0.542	0.829	0.941
Liu *et al.* [20]	0.335	1.06	-	0.127	-	-	-
Li *et al.* [16]	0.232	0.821	-	0.094	0.621	0.886	0.968
Liu *et al.* [19]	0.230	0.824	-	0.095	0.614	0.883	0.971
Wang *et al.* [37]	0.220	0.745	0.262	0.094	0.605	0.890	0.970
Eigen *et al.* [6]	0.215	0.907	0.285	-	0.611	0.887	0.971
Roy and Todorovic [27]	0.187	0.744	-	0.078	-	-	-
Eigen and Fergus [5]	0.158	0.641	0.214	-	0.769	0.950	**0.988**
ours (ResNet-UpProj)	**0.127**	**0.573**	**0.195**	**0.055**	**0.811**	**0.953**	**0.988**

Table 2. Comparison of the proposed approach against the state of the art on the NYU Depth v2 dataset. The values are those originally reported by the authors in their respective paper

Eigen *et al.* [6]	0.215	0.907	0.285	-	0.611	0.887	0.971
Roy and Todorovic [27]	0.187	0.744	-	0.078	-	-	-
Eigen and Fergus [5]	0.158	0.641	0.214	-	0.769	0.950	**0.988**
ours (ResNet-UpProj)	**0.127**	**0.573**	**0.195**	**0.055**	**0.811**	**0.953**	**0.988**



Deep Convolutional Neural Fields for Depth Estimation from a Single Image_PAMI2015

$$\boxed{E(\mathbf{y}, \mathbf{x}) = \sum_{p \in \mathcal{N}} U(y_p, \mathbf{x}) + \sum_{(p,q) \in \mathcal{S}} V(y_p, y_q, \mathbf{x})}$$

$$\Pr(\mathbf{y}|\mathbf{x}) = \frac{1}{\mathrm{Z}(\mathbf{x})} \exp(-E(\mathbf{y}, \mathbf{x})), \tag{1}$$

where E is the energy function; Z is the partition function defined as:

$$\mathrm{Z}(\mathbf{x}) = \int_{\mathbf{y}} \exp\{-E(\mathbf{y}, \mathbf{x})\} \mathrm{d}\mathbf{y}. \tag{2}$$

$$\boxed{\mathbf{y}^{\star} = \operatorname*{argmax}_{\mathbf{y}} \Pr(\mathbf{y}|\mathbf{x}).}$$

Unary potential

$$U(y_p, \mathbf{x}; \boldsymbol{\theta}) = (y_p - z_p(\boldsymbol{\theta}))^2, \quad \forall p = 1, ..., n. \tag{5}$$

Here z_p is the regressed depth of the superpixel p parametrized by the CNN parameters $\boldsymbol{\theta}$.

Pairwise potential

$$V(y_p, y_q, \mathbf{x}; \boldsymbol{\beta}) = \frac{1}{2} R_{pq} (y_p - y_q)^2, \ \forall p, q = 1, ..., n. \tag{6}$$

Here R_{pq} is the output of the network in the pairwise part (see Fig. 1) from a neighbouring superpixel pair (p, q). We use a fully-connected layer here:

$$R_{pq} = \boldsymbol{\beta}^{\top} [S_{pq}^{(1)}, \ldots, S_{pq}^{(K)}]^{\top} = \sum_{k=1}^{K} \beta_k S_{pq}^{(k)}, \tag{7}$$

where $\mathbf{S}^{(k)}$ is the k-th similarity matrix whose elements are $S_{pq}^{(k)}$ ($\mathbf{S}^{(k)}$ is symmetric); $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_k]^{\top}$ are the network parameters. From Eq. (7), we can see that we don't

$$\boxed{E(\mathbf{y},\mathbf{x}) = \sum_{p\in\mathcal{N}} (y_p - z_p)^2 + \sum_{(p,q)\in\mathcal{S}} \frac{1}{2} R_{pq}(y_p - y_q)^2.}$$

For ease of expression, we introduce the following notation:

$$\mathbf{A} = \mathbf{I} + \mathbf{D} - \mathbf{R}, \tag{9}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix; $\mathbf{R}$ is the matrix composed of R_{pq}; $\mathbf{D}$ is a diagonal matrix with $\mathbf{D}_{pp} = \sum_q R_{pq}$. Expanding Eq. (8), we have:

$$E(\mathbf{y},\mathbf{x}) = \mathbf{y}^\top \mathbf{A}\mathbf{y} - 2\mathbf{z}^\top \mathbf{y} + \mathbf{z}^\top \mathbf{z}. \tag{10}$$

$$\Pr(\mathbf{y}|\mathbf{x}) = \frac{|\mathbf{A}|^{\frac{1}{2}}}{\pi^{\frac{n}{2}}} \exp\left\{ -\mathbf{y}^\top \mathbf{A}\mathbf{y} + 2\mathbf{z}^\top \mathbf{y} - \mathbf{z}^\top \mathbf{A}^{-1}\mathbf{z} \right\}, \tag{12}$$

Final optimization

$$\begin{aligned} \min_{\boldsymbol{\theta},\boldsymbol{\beta}\geq\mathbf{0}} & -\sum_{i=1}^{N} \log \Pr(\mathbf{y}^{(i)}|\mathbf{x}^{(i)};\boldsymbol{\theta},\boldsymbol{\beta}) \\ & + \frac{\lambda_1}{2}\|\boldsymbol{\theta}\|_2^2 + \frac{\lambda_2}{2}\|\boldsymbol{\beta}\|_2^2, \end{aligned}$$

Depth prediction

$$\begin{aligned} \mathbf{y}^\star &= \operatorname*{argmax}_{\mathbf{y}} \Pr(\mathbf{y}|\mathbf{x}) \\ &= \operatorname*{argmax}_{\mathbf{y}} -\mathbf{y}^\top \mathbf{A}\mathbf{y} + 2\mathbf{z}^\top \mathbf{y} \\ &\boxed{= \mathbf{A}^{-1}\mathbf{z}.} \end{aligned}$$

Train Details：

- An image contains ~700 superpixels
- initialize the first 6 layers of the unary part using a CNN model trained on the ImageNet
- Training the whole net-work takes around 33 hours on the NYU v2 dataset
- it takes ~ 1.1s to perform the network forward pass

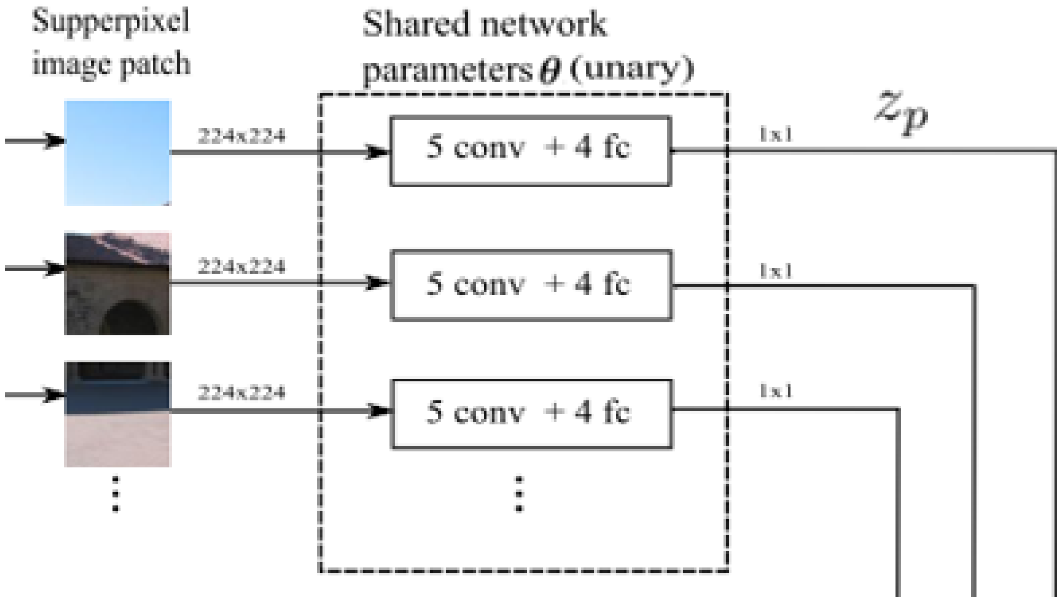
Method	Error (lower is better)			Accuracy (higher is better)		
	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
SVR	0.313	0.128	1.068	0.490	0.787	0.921
SVR (smooth)	0.290	0.116	0.993	0.514	0.821	0.943
Unary only	0.295	0.117	0.985	0.516	0.815	0.938
Unary only (smooth)	0.287	0.112	0.956	0.535	0.828	0.943
Ours (pre-train)	0.257	0.101	0.843	0.588	0.868	0.961
Ours (fine-tune)	**0.230**	**0.095**	**0.824**	**0.614**	**0.883**	**0.971**

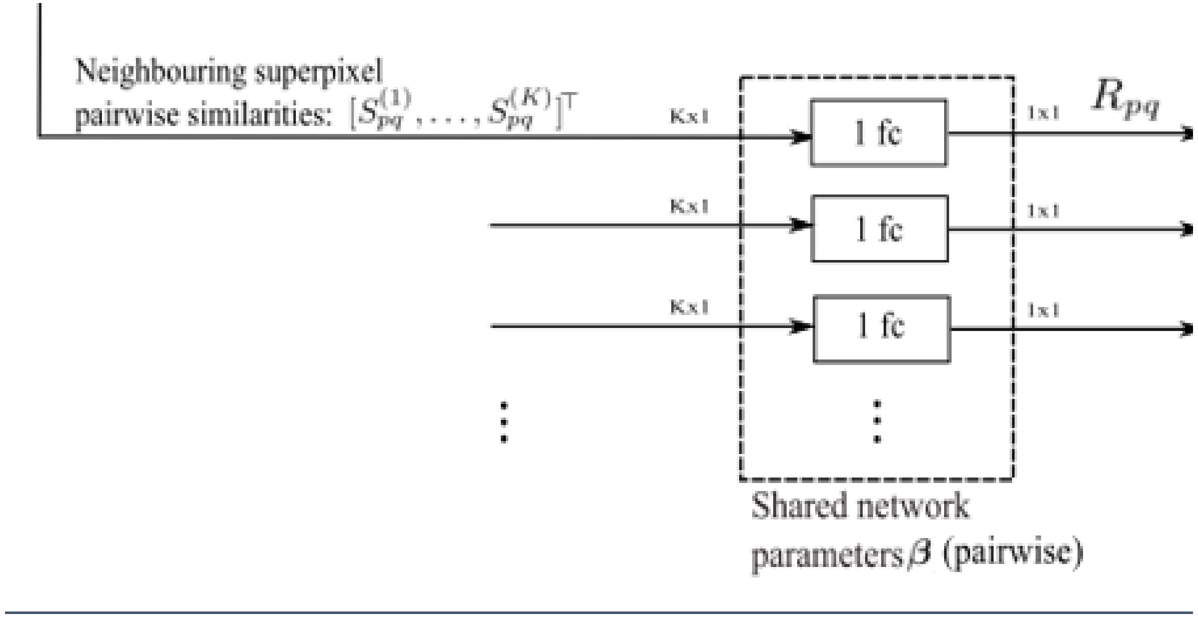
Table 2: Baseline comparisons on the NYU v2 dataset. Our method with the whole network training performs the best.

Method	Error (lower is better)			Accuracy (higher is better)		
	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Make3d [15]	0.349	-	1.214	0.447	0.745	0.897
DepthTransfer [5]	0.35	0.131	1.2	-	-	-
Discrete-continuous CRF [16]	0.335	0.127	1.06	-	-	-
Ladicky *et al.* [8]	-	-	-	0.542	0.829	0.941
Eigen *et al.* [1]	**0.215**	-	0.907	0.611	**0.887**	**0.971**
Ours (pre-train)	0.257	0.101	0.843	0.588	0.868	0.961
Ours (fine-tune)	0.230	**0.095**	**0.824**	**0.614**	0.883	**0.971**

Conclusion：

- **Unary part：** single superpixel depth estimation does not use the **information of global context**
- **Pairwise part：** only compare the **similarity of adjacent superpixels**, not use **other cues**, such as occlusion, linear perspective





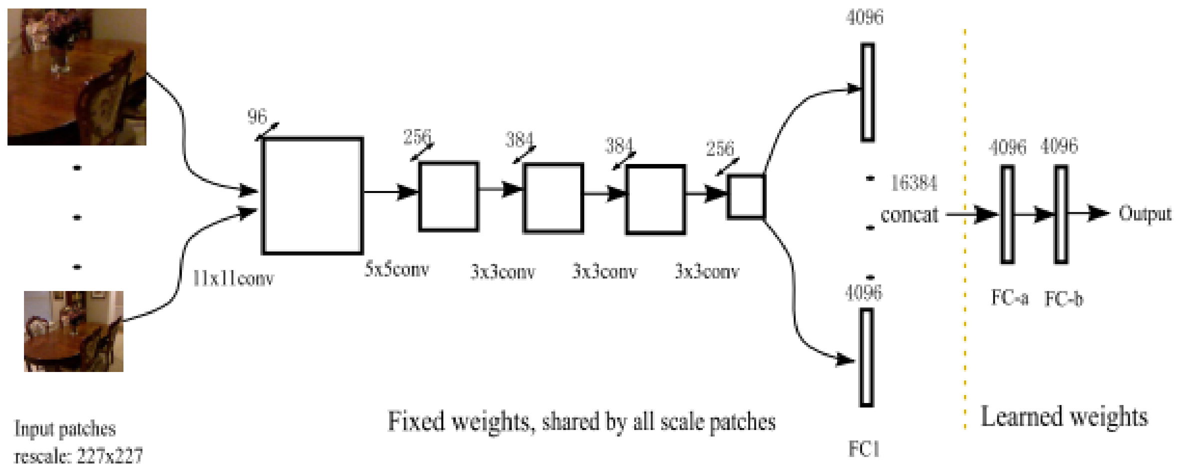


Figure 1: *Visualization of our multi-scale framework. Each patch goes through five convolutional layers and the first fully-connected layer (here transferred from AlexNet). The features are concatenated before they are fed to two additional fully-connected layers. We then refine the predictions from the CNN by inference of a hierarchical CRF (not shown here; see text for details).*

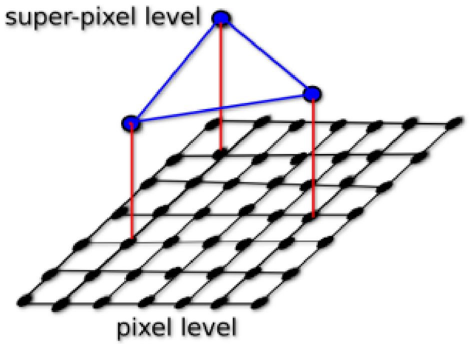


Figure 2: *Illustration of our hierarchical CRF. Two layers are connected via region hierarchy. The blue nodes represent the super-pixels, where the depth is regressed by the proposed CNN. The blue edges between the nodes represent the neighborhoods at the super-pixel level; and the black edges represent the relation at the pixel level and the red edges represent the relation between these two levels which is forced to be equal.*

Method	$\delta < 1.25$ $\delta < 1.25^2$ $\delta < 1.25^3$	rel	$\log_{10}$	rms
Depth transfer [13]*	- - -	0.374	0.134	1.12
Liu *et al.* [11]*	- - -	0.335	0.127	1.06
Our method*	**63.95%** **90.03%** **97.41%**	**0.223**	**0.091**	**0.759**
Ladicky *et al.* [18]	54.22% 82.90% 94.09%	-	-	-
Eigen *et al.* [19]	61.1% **88.7%** **97.1%**	**0.215**	**0.094**	0.871
Regression only	59.94% 87.20% 96.30%	0.243	0.098	0.851
Our method	**62.07%** 88.61% 96.78%	0.232	**0.094**	**0.821**

Table 2: *Depth estimation errors on the NYU v2 data set, ∗*

Normal Classifier Based Regularization

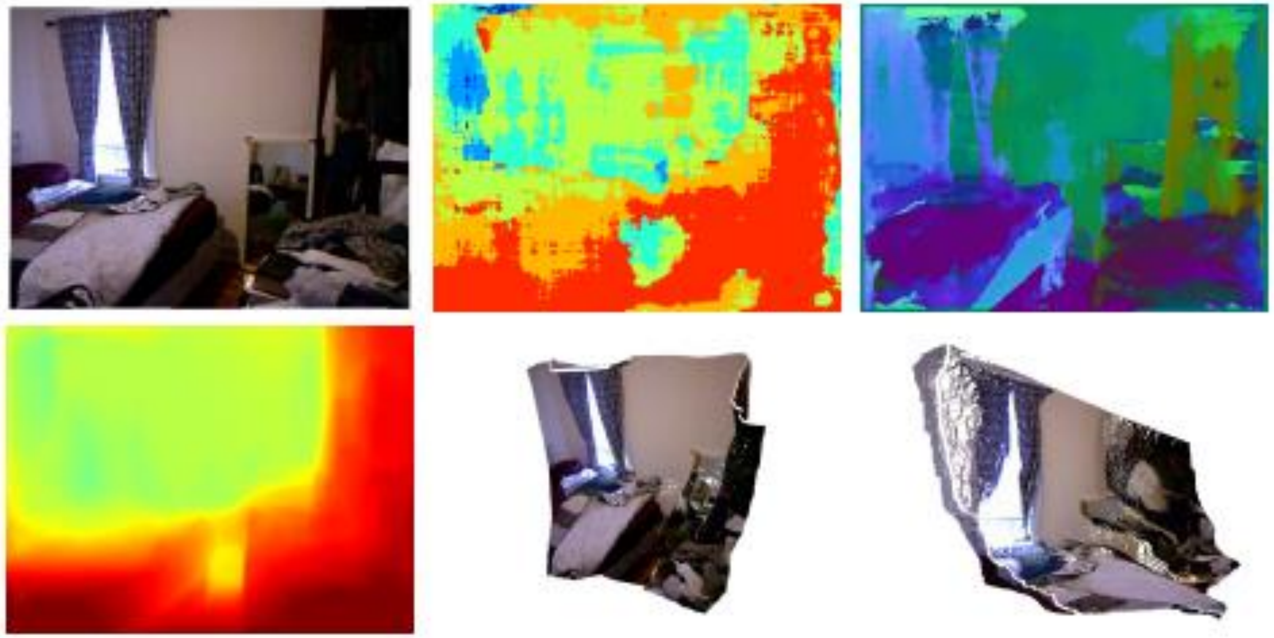
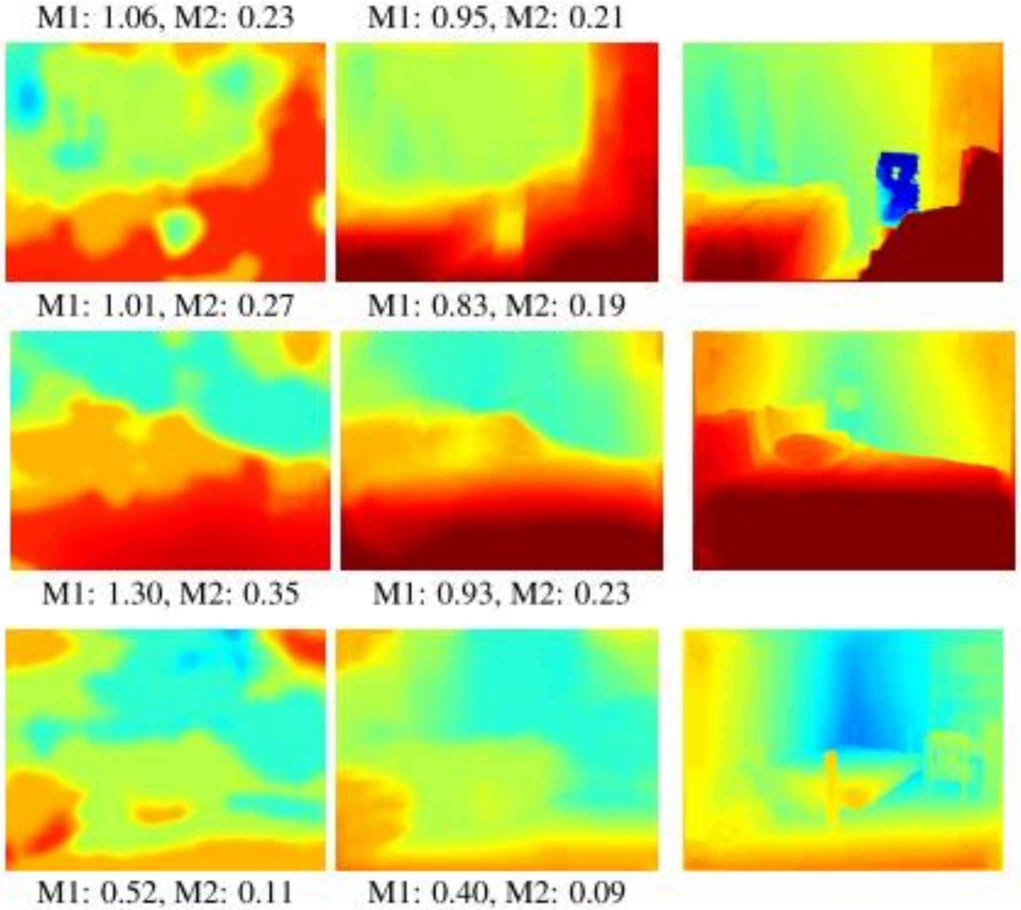


Figure 1. Overview of our method. Top Row: The input to our method is depicted in the top row. On a single input image (left) two classifiers are evaluated, single view depth estimation (middle) and surface normal directions (right). Bottom Row: On the bottom the obtained depth map by our surface normal direction based regularization is shown (left) together with two renderings of the obtained dense point cloud (middle and right).



No compare test table

Direction Matters - Depth Estimation with a Surface Normal Classifier_CVPR2015

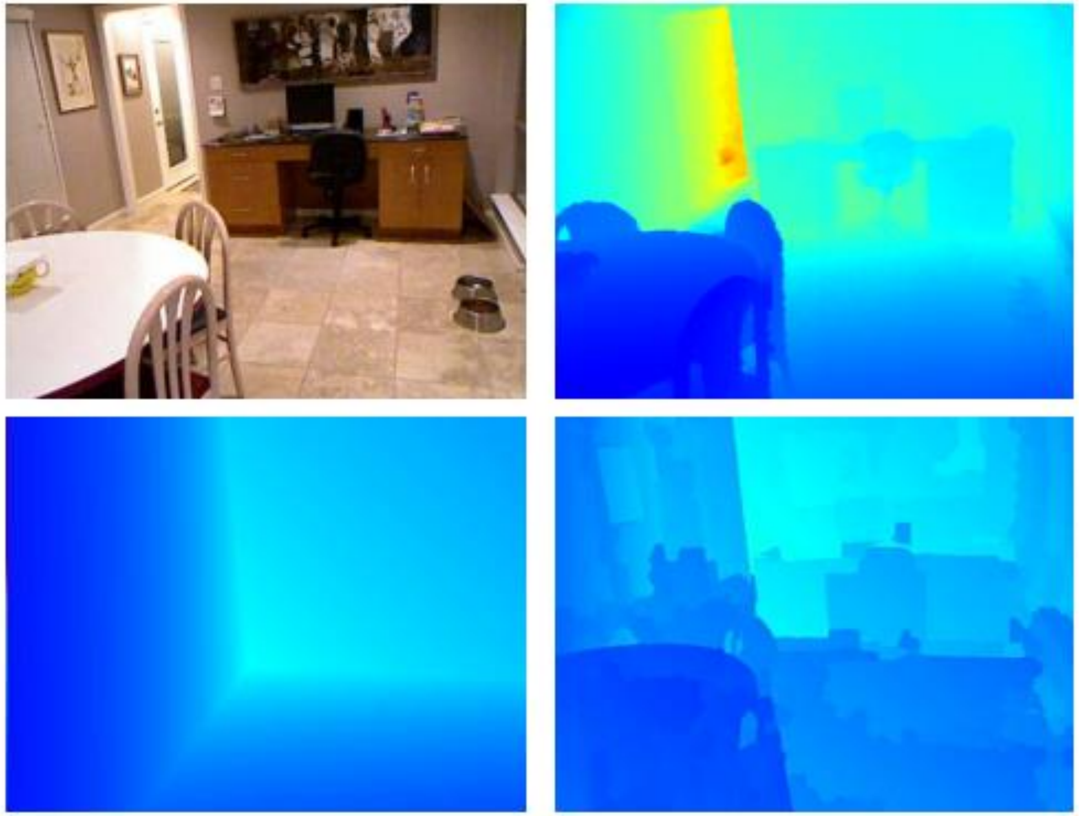


Figure 1. **Depth estimation from a single image:** (Top) Image and ground-truth depth map. (Bottom) Estimated layout and detailed depth map. Color indicates depth (red is far, blue is close).

Inference is achieved by **maximizing the joint distribution of our CRF**, or equiva-lently minimizing the energy

$$E(Y, R, L) = E_l(Y) + E_m(Y, R) + E_g(Y, L)$$

Main Idea：

- Based on features, **retrieve candidate training images** and calculate superpixel depth, region depth, layout depth
- Based on **attribute of superpixel, region and layout**, set up the loss or energy function
- make use of the **Distributed Convex Belief Propagation** (DCBP) method of to perform inference in CRF

$$E(Y,R,L) = E_l(Y) + E_m(Y,R) + E_g(Y,L)$$

$$E_l(Y) = \sum_p \phi_p(y_p) + \sum_{p,q} \phi_{p,q}(y_p, y_q)$$

$$\phi_p(y_p) = \frac{1}{N_p} \sum_{i=1}^{N_p} \left(d_p^i(y_p) - d_{r,p}^i\right)^2$$

$$\phi_{p,q}(y_p, y_q) = w_l \cdot \begin{cases} 0 & \text{if } o_{pq} = 1 \\ g_{pq}\|\mathbf{n}_p(y_p) - \mathbf{n}_q(y_q)\|^2 + \\ \frac{1}{N_{pq}} \sum_{j=1}^{N_{pq}} (d_p^j(y_p) - d_q^j(y_q))^2 & \text{if } o_{pq} = 0 \end{cases}$$

$$E_m(Y,R) = \sum_\gamma \phi_\gamma(r_\gamma) + \sum_{\gamma,p} \phi_{\gamma,p}(r_\gamma, y_p)$$

$$\phi_\gamma(r_\gamma) = w_m \cdot (\max(P_{dn}(d(r_\gamma), \mathbf{n}(r_\gamma))) - P_{dn}(d(r_\gamma), \mathbf{n}(r_\gamma)))$$

$$\phi_{\gamma,p}(r_\gamma, y_p) = \frac{w_{m,l}}{N_p} \sum_{i=1}^{N_p} \left(d_p^i(y_p) - d_\gamma^i(r_\gamma)\right)^2$$

$$E_g(Y,L) = \sum_p \phi_{L,p}(L, y_p)$$

$$\phi_{L,p}(L, y_p) = \frac{w_g}{N_p} \sum_{i=1}^{N_p} (1 - P_c^i) \cdot (d_p^i(y_p) - d_L^i(L))^2$$

Method	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	mean	median	$\theta < 11.25$	$\theta < 22.5$	$\theta < 30$
DepthTransfer	0.374	0.134	1.12	49.81%	79.46%	93.75%	**43.0**	**40.5**	6.9%	23.2%	34.9%
DC-Depth	0.335	0.127	1.06	51.55%	82.32%	95.00%	45.7	42.2	19.7%	25.7%	35.4%
SemanticDepth	-	-	-	**54.22%**	82.90%	94.09%	-	-	-	-	-
Ours	**0.305**	**0.122**	**1.04**	52.50%	**83.77%**	**96.16%**	46.7	41.9	**21.1%**	**35.2%**	**41.7%**

Method	rel	log10	rms	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Ours-local	0.334	0.128	1.05	50.35%	82.31%	95.44%
Ours-mid	0.312	0.123	**1.03**	52.08%	**83.92%**	96.13%
Ours-global-only	0.325	0.128	1.07	50.38%	82.06%	95.35%
Ours	**0.305**	**0.122**	1.04	**52.50%**	83.77%	**96.16%**

Table 2. **NYU v2: Ablation study.** We evaluate the influence of the different components of our model. These results confirm that each parts of our model contributes to the final results, with a strong influence of the mid-level structures.

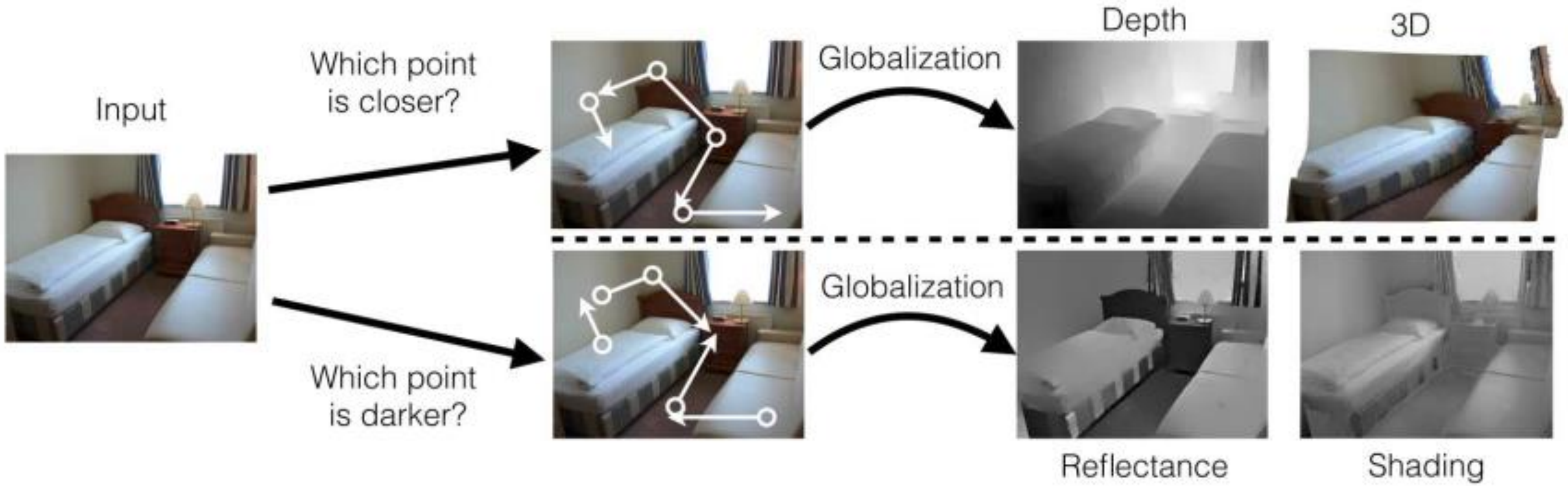
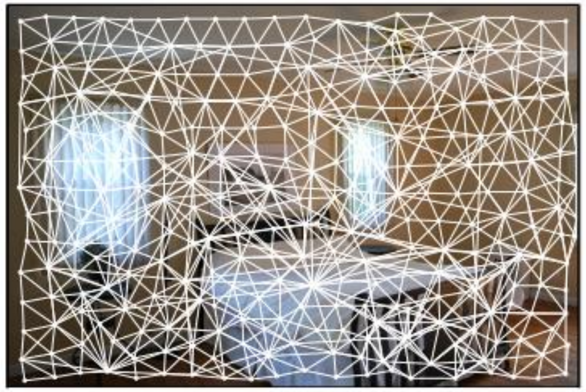


Figure 1: Framework overview: given an input image we choose points on which to make ordinal estimates (*e.g.* Which of two points is closer to the camera? Which has darker surface color?). We train models to perform these estimations. We then globalize these estimates to produce metric estimates of reflectance, shading and depth. Our approach has multiple benefits over direct metric estimation.

① From input image to point pairs

- choose N points
- edge structure



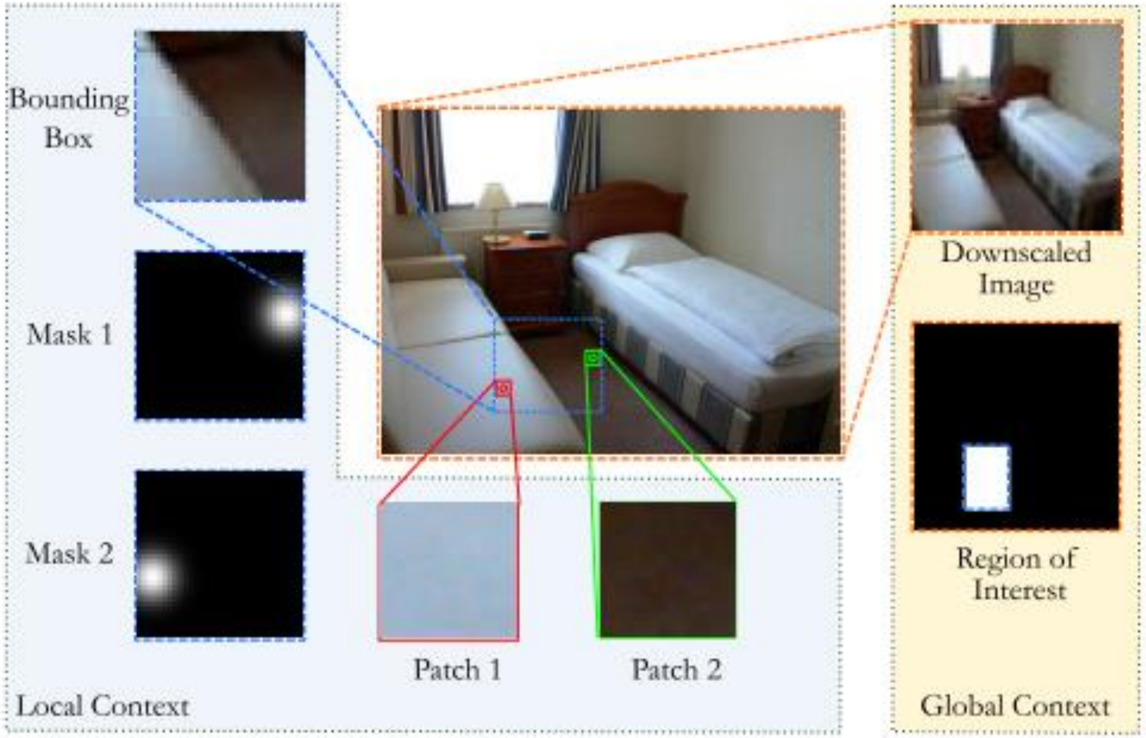
② From point pairs to ordinal

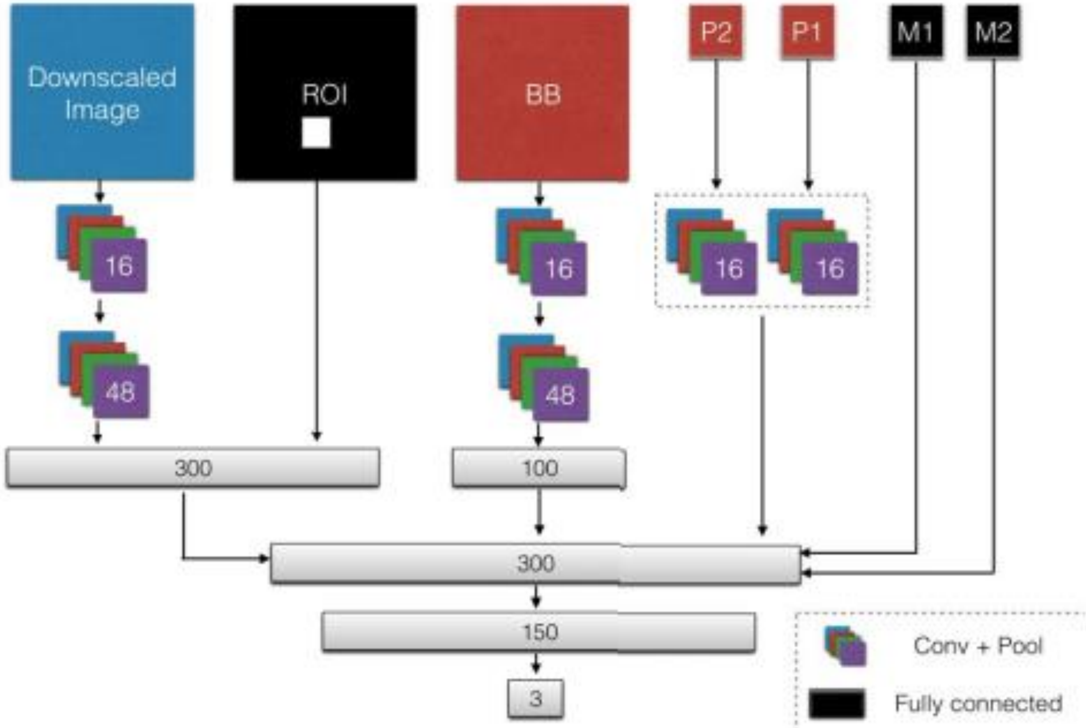
Network input：

- local appearance of the two points
- two points local surroundings
- global context

Output： Ordinal classes

- Equality between the points
- point i being larger
- point j being larger





③ From ordinal to metric

$$\mathcal{L}_{eq}(\mathbf{x}, \mathbf{R}^{eq}) = \sum_{ij} w_{ij}^{eq}(x_i - x_j - R_{ij}^{eq})^2$$

$R_{ij}^{eq} \sim \mathcal{N}(0, \sigma_{eq}^2)$ Scalar slack variable

$$\mathcal{L}_{eq}(\mathbf{x}, \mathbf{R}^{eq}) = [\mathbf{x} \ \ \mathbf{R}^{eq}]^T \mathbf{A}_{eq}^T \mathbf{W}_{eq} \mathbf{A}_{eq} \begin{bmatrix} \mathbf{x} \\ \mathbf{R}^{eq} \end{bmatrix}$$

$$\mathcal{L}_{gt}(\mathbf{x}, \mathbf{R}^{gt}) = \sum_{ij} w_{ij}^{gt} \left(x_i - x_j - R_{ij}^{gt}\right)^2$$

$$\mathcal{L}_{gt}(\mathbf{x}, \mathbf{R}^{gt}) = [\mathbf{x} \ \ \mathbf{R}^{gt}]^T \mathbf{A}_{gt}^T \mathbf{W}_{gt} \mathbf{A}_{gt} \begin{bmatrix} \mathbf{x} \\ \mathbf{R}^{gt} \end{bmatrix}$$

$$\mathcal{L}_s(\mathbf{x}) = \sum_{ij} w_{i,j}^s (x_i - x_j)^2 + \sum_i b_i x_i$$

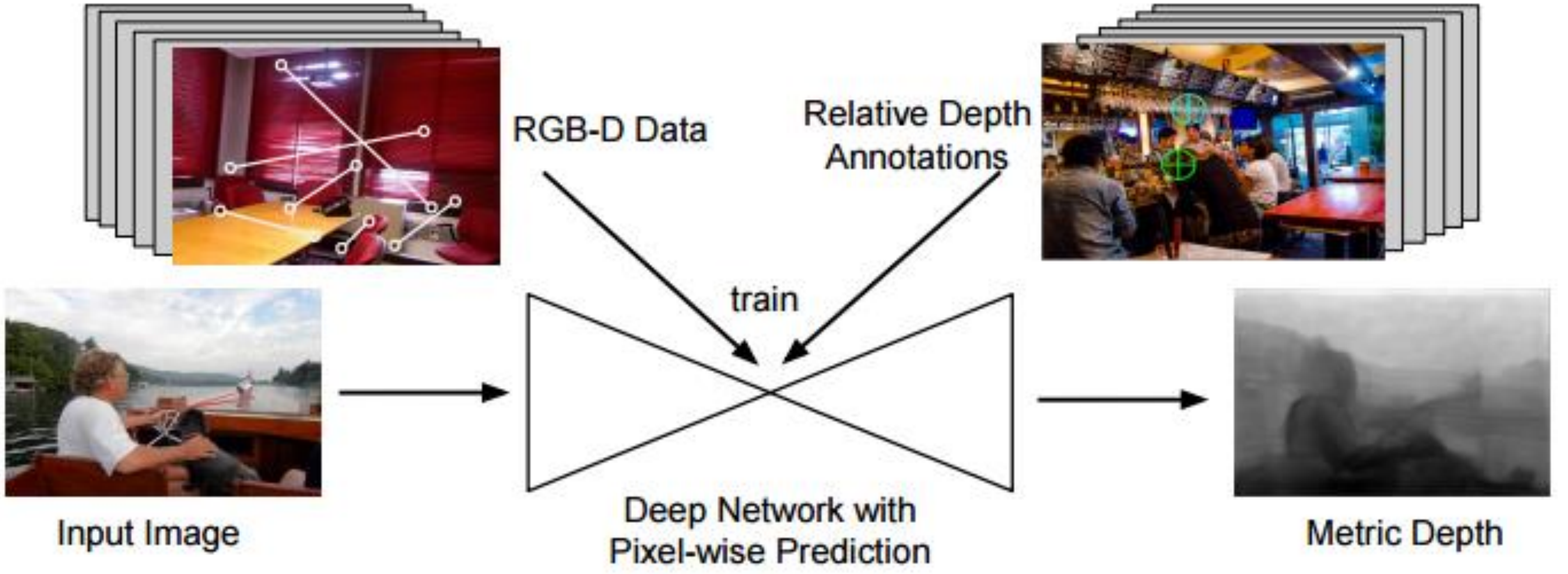
$$\mathcal{L}_s(\mathbf{x}) = \mathbf{x}^T \mathbf{A}_s^T \mathbf{W}_s \mathbf{A}_s \mathbf{x} + \mathbf{x}^T \mathbf{b}_s.$$

constrained quadratic problem

$$\min_{\mathbf{x}, \mathbf{R}^{eq}, \mathbf{R}^{gt}, \mathbf{R}^{lt}} \lambda_{eq}\mathcal{L}_{eq}(\mathbf{x}, \mathbf{R}^{eq}) + \lambda_{gt}\mathcal{L}_{gt}(\mathbf{x}, \mathbf{R}^{gt}) + \lambda_{lt}\mathcal{L}_{lt}(\mathbf{x}, \mathbf{R}^{lt}) + \lambda_s\mathcal{L}_s(\mathbf{x}) + \sum_{ij}\left(\frac{(R_{ij}^{eq})^2}{\sigma_{eq}^2} + \frac{(R_{ij}^{gt} - \mu_{gt})^2}{\sigma_{gt}^2} + \frac{(R_{ij}^{lt} - \mu_{lt})^2}{\sigma_{lt}^2}\right)$$

s.t $\mathbf{x} > \mathrm{L}, \mathbf{x} < \mathrm{U}, \mathbf{R}^{eq} > 0, \mathbf{R}^{gt} > 0, \mathbf{R}^{lt} > 0$

Figure 5: Example images and annotations. Green points are those annotated as closer in depth.



Single-Image Depth Perception in the Wild_NIPS2016

How to train the network using only ordinal annotations?

A loss function that **encourages the predicted depth map to agree with the ground-truth ordinal relations**

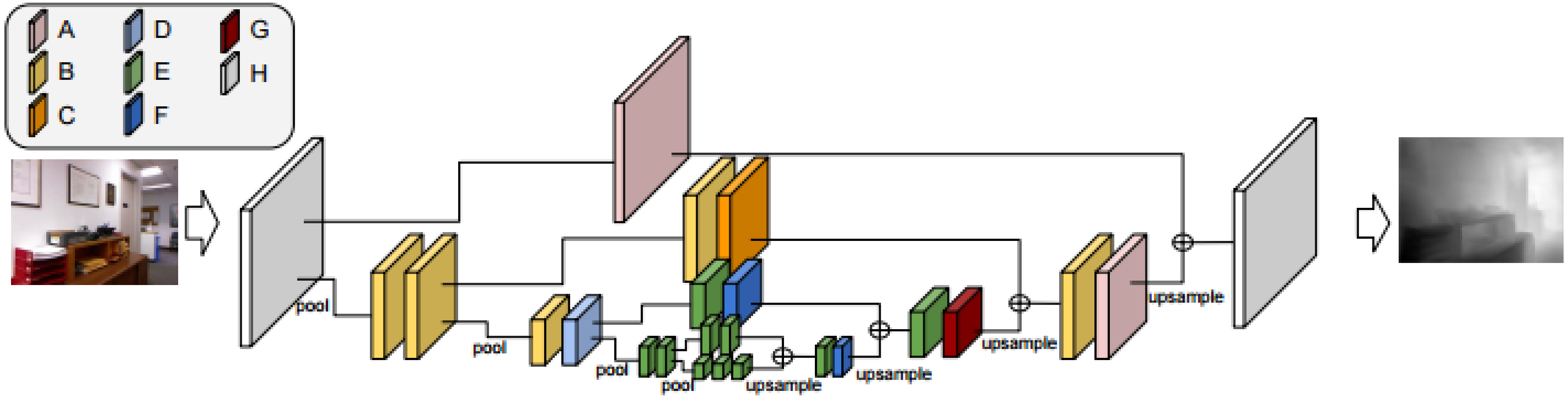
$$L(I, R, z) = \sum_{k=1}^{K} \psi_k(I, i_k, j_k, r, z),$$

r k ∈ {+1, −1, 0}，ground-truth depth relation between i k and j k : closer (+1) further (−1) equal (0)

$$\psi_k(I, i_k, j_k, z) = \begin{cases} \log\left(1 + \exp(-z_{i_k} + z_{j_k})\right), & r_k = +1 \\ \log\left(1 + \exp(z_{i_k} - z_{j_k})\right), & r_k = -1 \\ (z_{i_k} - z_{j_k})^2, & r_k = 0. \end{cases}$$

- it encourages a **small difference between depths** if the **ground-truth relation is equality**
- otherwise it **encourages a large difference**

“hourglass” network： has been used to achieve state-of-the-art results on human pose estimation



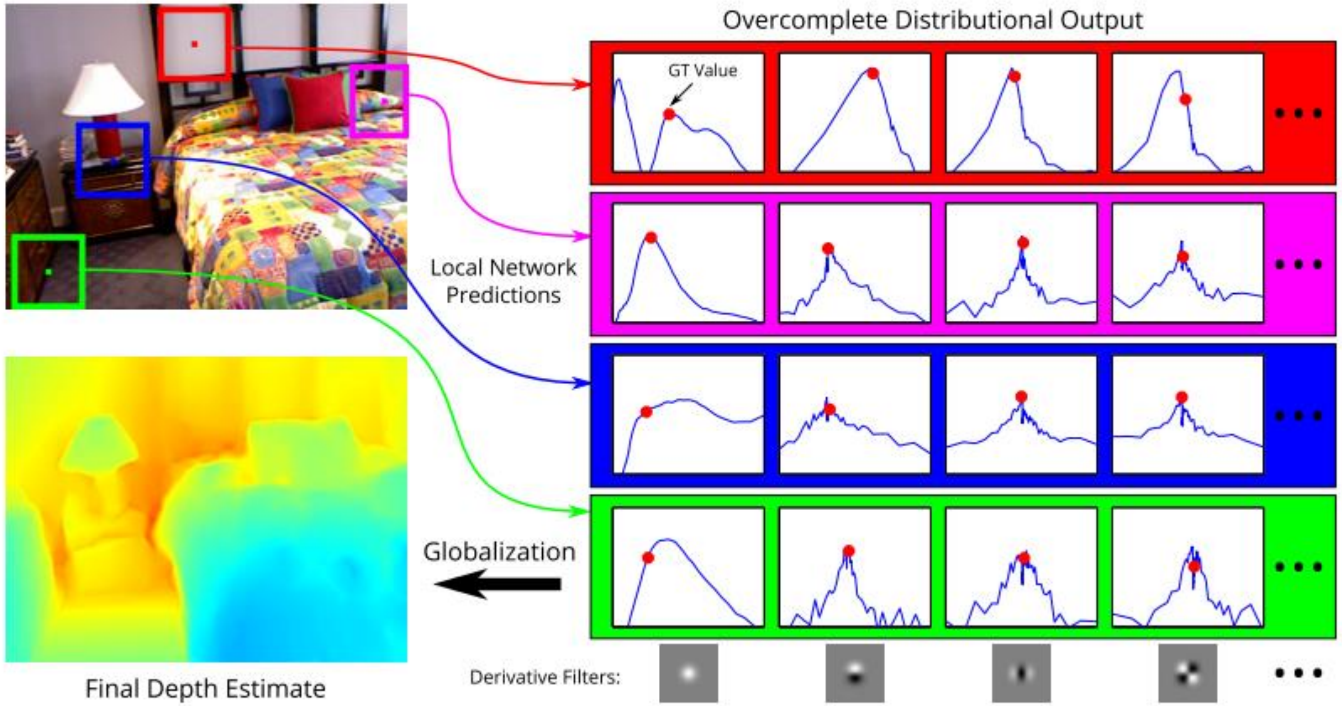
ordinal error measures

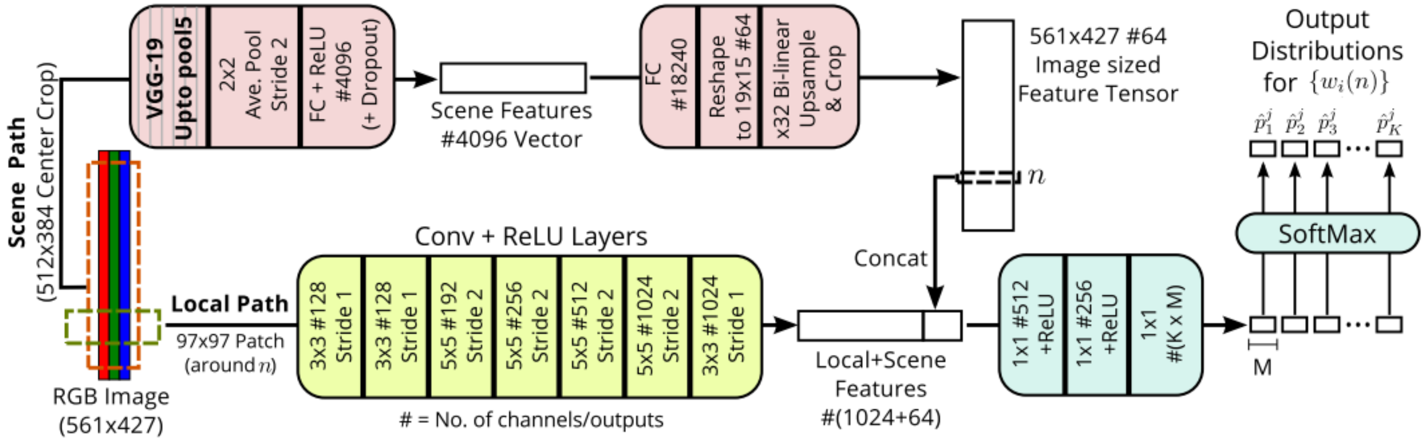
Method	WKDR	$WKDR^{=}$	$WKDR^{\neq}$
Ours	**35.6%**	**36.1%**	**36.5%**
Zoran [14]	43.5%	44.2%	41.4%
rand_12K	**34.9%**	**32.4%**	**37.6%**
rand_6K	36.1%	32.2%	39.9%
rand_3K	35.8%	28.7%	41.3%
Ours_Full	**28.3%**	**30.6%**	**28.6%**
Eigen(A) [8]	37.5%	46.9%	32.7%
Eigen(V) [8]	34.0%	43.3%	29.6%

Depth error measures

Method	RMSE	RMSE (log)	RMSE ^{a} (s.inv)	absrel	sqrrel
Ours	1.13	0.39	0.26	0.36	0.46
Ours_Full	1.10	0.38	0.24	0.34	0.42
Zoran [14]	1.20	0.42	-	0.40	0.54
Eigen(A) [8]	0.75	0.26	0.20	0.21	0.19
Eigen(V) [8]	**0.64**	**0.21**	**0.17**	**0.16**	**0.12**
Wang [28]	0.75	-	-	0.22	-
Liu [6]	0.82	-	-	0.23	-
Li [10]	0.82	-	-	0.23	-
Karsch [1]	1.20	-	-	0.35	-
Baig [40]	1.0	-	-	0.3	-

Single-Image Depth Perception in the Wild_NIPS2016





Depth from a Single Image by Harmonizing Overcomplete Local Network Predictions_NIPS2016

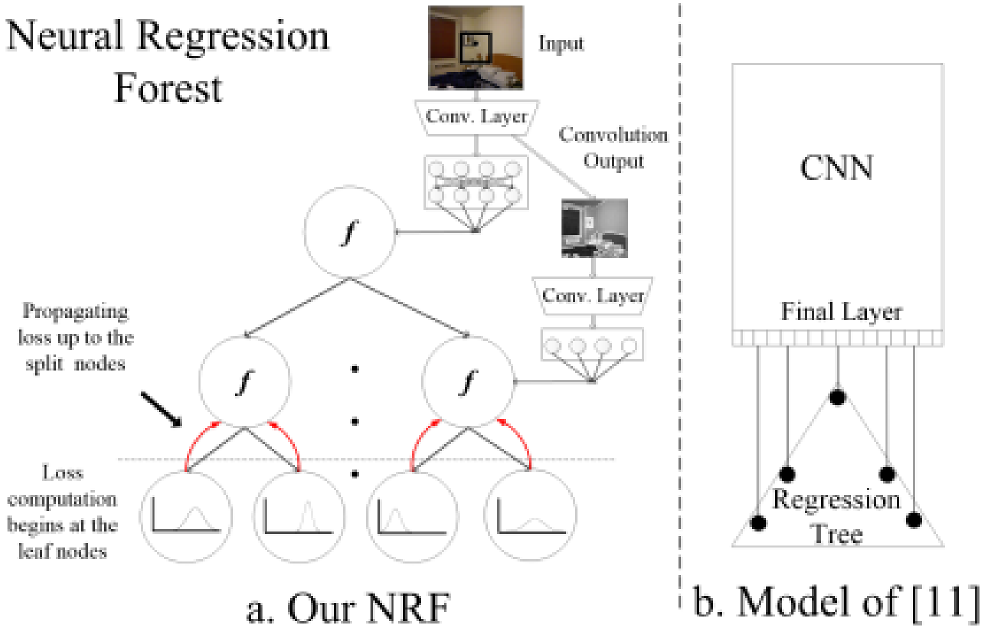
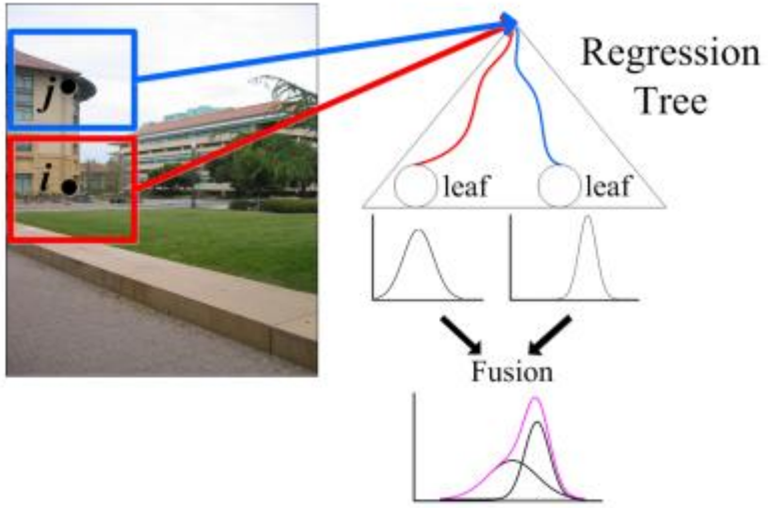


Figure 2. Neural Regression Forest. (a) A CNN is associated with every node of a binary Convolutional Regression Tree (CRT) for performing the convolutional processing of data samples. The CNN's output is passed to the left and right children nodes with a Bernoulli probability. (b) While our CNNs process data samples as they pass down the CRT, the related deep architecture of [11] uses a single deep CNN to fully process the data before passing them through a decision tree.



	Make3D			NYU v2		
	rel	log10	rms	rel	log10	rms
[22]	0.370	0.187	-	0.349	-	1.214
[1]	0.362	0.168	15.8	-	-	-
[16]	0.338	0.134	12.60	0.335	0.127	1.06
[10]	0.361	0.148	15.10	0.35	0.131	1.2
[12]	0.364	0.148	-	-	-	-
[14]	0.379	0.148	-	-	-	-
[6]	-	-	-	0.215	-	0.907
[15]	0.307	0.125	12.89	0.230	0.095	0.824
[27]	-	-	-	0.305	0.122	1.04
Ours	0.26	0.119	12.40	0.187	0.078	0.744

Table 2. Comparison with the state of the art on Make3D and NYU v2 datasets.

Future work

The information of linear perspective and geometry is abundant!

Figure 2. 3d reconstruction of a corridor from single image presented in Figure 1 using our autonomous algorithm.

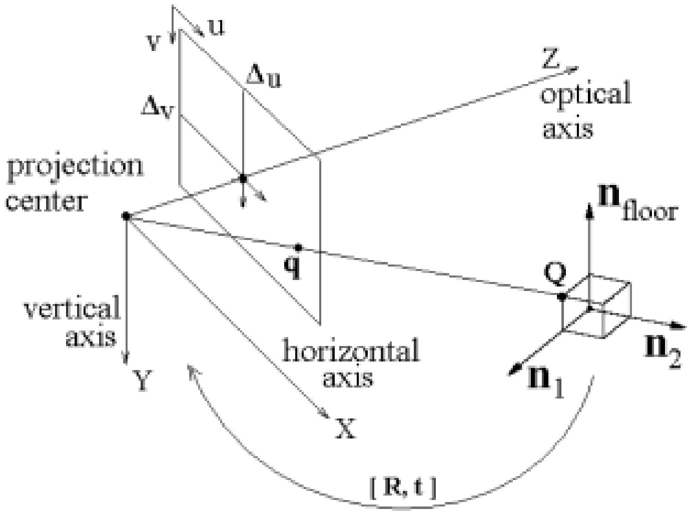


Figure 3. Coordinate systems involved in 3d reconstruction.

Future work

The information of linear perspective and geometry is abundant!

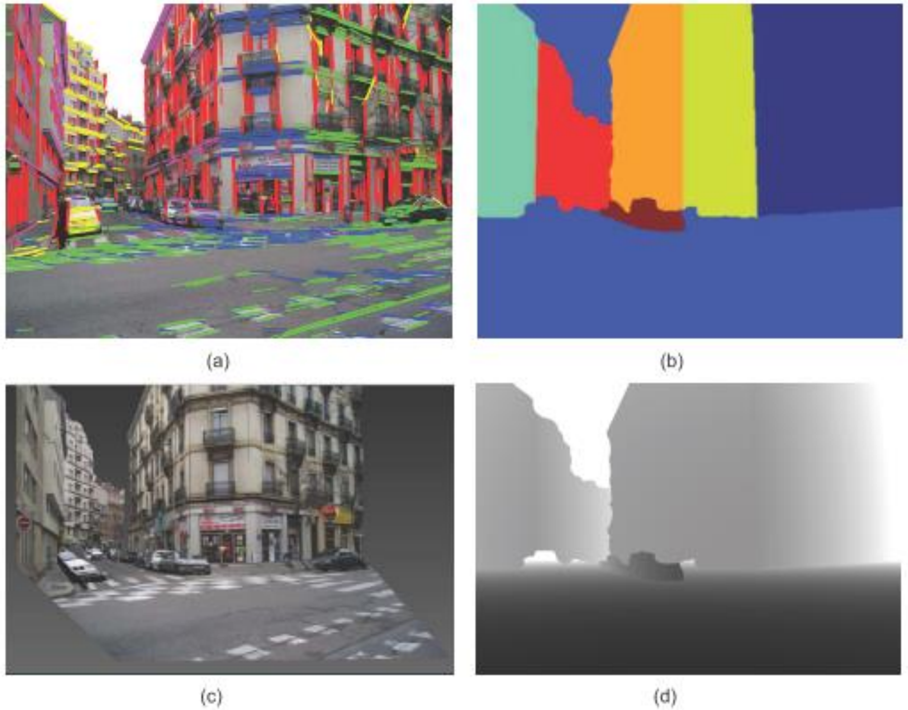
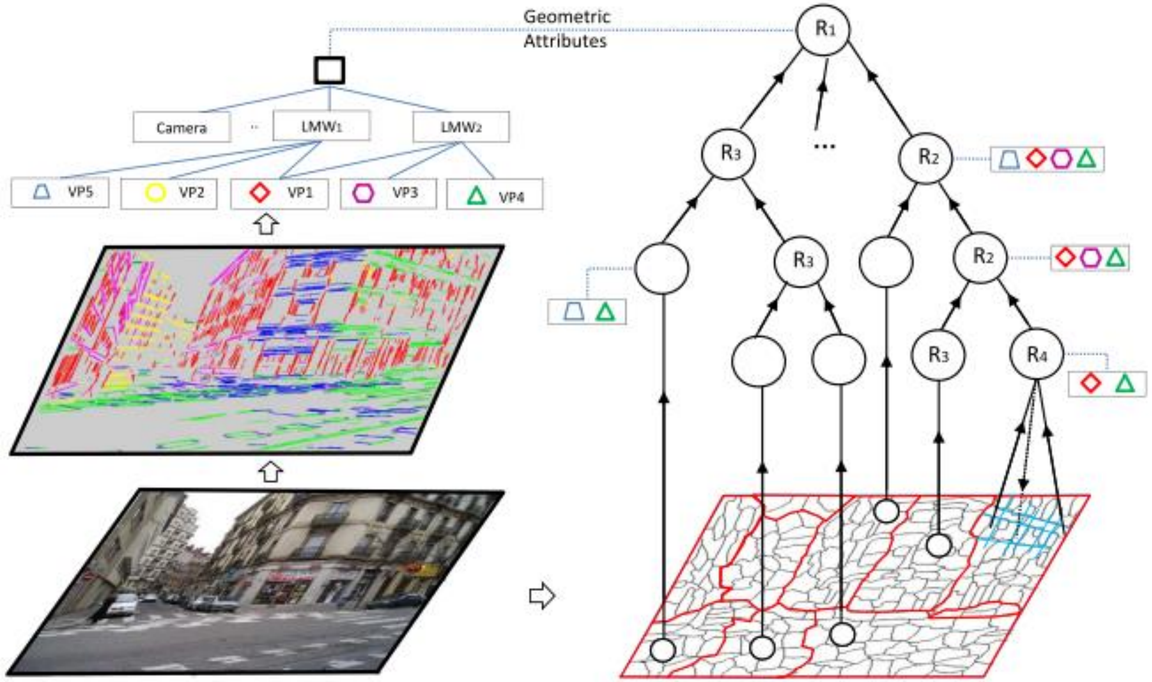


Fig. 1. A typical result of our approach. (a) Input image overlaid with detected parallel lines; (b) surface normal map where each color indicates a unique normal orientation; (c) synthesized images from a novel viewpoint; and (d) depth map (darker is closer).



Future work

Almost all human environment scene images contains straight line and linear perspective

- **Depth estimation or optimization**

  Try to make use of the geometry and linear perspective to constraint depth estimation

- **Ordinal relationship and space relationship**

  Try to measure the ordinal relationship using linear perspective

  Try to quantificat the ordinal relationship, not only use the tag of near/far/equality

Finding straight lines