CSE 559A: Computer Vision

Fall 2018: T-R: 11:30-1pm @ Lopata 101

Instructor: Ayan Chakrabarti (ayan@wustl.edu).
Course Staff: Zhihao Xia, Charlie Wu, Han Liu

http://www.cse.wustl.edu/~ayan/courses/cse559a/

November 27, 2018

GENERAL

- Reminder: Problem Set 5: Deadline Extended to Dec 4th.

- Recitation this Friday in Lopata 103: 10:30AM - Noon

- Project Reports Due Dec 9th

NEURAL NETWORKS FOR PHYSICAL TASKS

- So far, we've talked about using neural networks for semantic tasks
  - Classification
  - Segmentation
  - Object Detection
- The argument is, semantic tasks have no model to invert, so must rely on learning.
- But remember, for a lot of low and mid-level vision tasks, we also had to use "regularization" or priors.
- These regularizers and priors were hand-crafted. Why not use the expressive capacity of neural networks here ?

DENOISING

- Simplest Case
- Traditional Methods
  - Smoothing with a Gaussian Filter
  - Bilateral Filtering
  - Solving some optimization problem
- Instead, just learn a neural network that predicts a clean image from a noisy image.
- Create training samples of noisy and clean pairs (y_i, x_i)
  - Cheap to create. Just get regular images x_i
  - Synthetically add Gaussian noise with a certain variance to get y_i.
- Train a neural network f to get $\hat{x} = f(y)$ and set loss as $||\hat{x} - x||^2$.
- Learn by back-propagation !

DENOISING

State-of-the-art: IRCNN. Zhang et al., CVPR 2018.

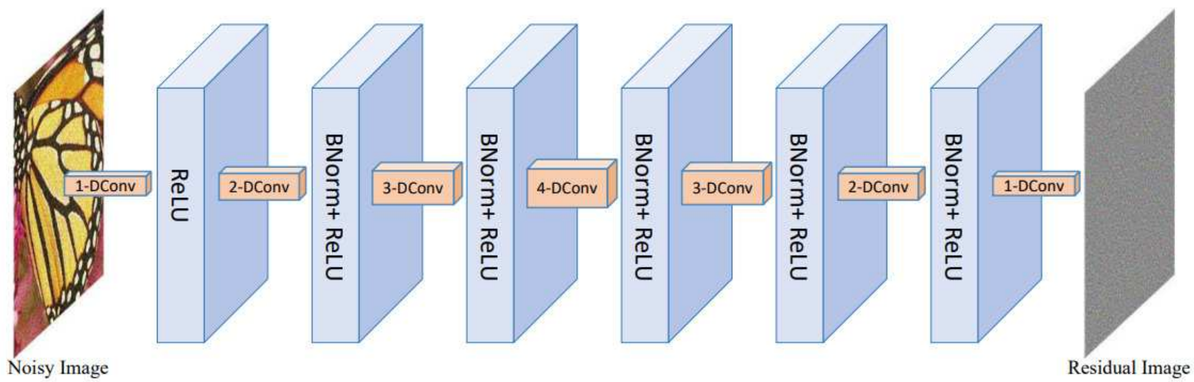


Figure 1. The architecture of the proposed denoiser network. Note that "s-DConv" denotes s-dilated convolution [63], here s = 1, 2, 3 and 4; "BNorm" represents batch normalization [32]; "ReLU" is the rectified linear units ($\max(\cdot, 0)$).

- Learns a network to predict the noise instead of the clean image
- $\hat{x} = y + f(y; \theta)$

DENOISING++

- Similarly, networks have been proposed for other image restoration tasks
  - Deblurring
  - Super-resolution
  - In-painting (fill in values in masked out pixels)
- But IRCNN proposes a clever way of training networks only for denoising (at different noise levels), and using them for a variety of other tasks---without re-training.

DENOISING++

Plug-and-Play Priors

- Problem setup
  - You observe $y = Ax +$ noise.
  - You know A. Given y, estimate x.
  - A may not be invertible: the problem is ill-posed
- Optimization Setup

$$\hat{x} = \arg\min_x \|y - Ax\|^2 - \log p(x)$$

Here $p(x)$ is a prior on natural clean images x. But we want this to be a complex prior: say something a CNN learns "implicitly".

- You can re-write this by "splitting" the variable x into x and z:

$$\hat{x} = \arg\min_{x,z} \|y - Ax\|^2 + \beta(x - z)^2 - \log p(z)$$

- Equivalent when $\beta \to \infty$

DENOISING++

Plug-and-Play Priors

$$\hat{x} = \arg\min_{x,z} \|y - Ax\|^2 + \beta(x - z)^2 - \log p(z)$$

- Equivalent when $\beta \to \infty$
- Basic Idea: Optimize x, z while slowly increasing the value of β
- Begin with some initial estimate of x. For a given value of β, alternatingly minimize this objective wrt x and z while keeping the other fixed.
- When z is fixed and you minimize wrt x, $\log p(z)$ isn't involved.

$$x = \arg\min_{x} \|y - Ax\|^2 + \beta(x - z)^2$$

This is a linear least-squares problem that we know how to solve !

- When x is fixed, and you minimize wrt z:

$$z = \arg\min_{z} \beta(x - z)^2 - \log p(z)$$

- This is Denoising: with noise variance $\sigma^2 = 1/(2\beta)$! Use a CNN for this.

DENOISING++

Plug-and-Play Priors

Algorithm

- Initialize x and pick an initial value of β
- At each iteration
  - Set z by "denoising" x with a network that assumes noise variance $= 1/(2\beta)$.
  - Set x by minimizing $|y - Ax|^2 + \beta(x - z)^2$ as a linear least squares problem.
  - Increase β according to some schedule.

STEREO

- Remember the first part of Stereo is building a cost volume.
- Each element of this cost volume $C[x, y, d]$ is the "distance" between the region or patch around (x, y) in the left image, and $(x - d, y)$ in the right image.
- Distance function must account for similarity between non-matching patches (in smooth regions), and variance between matching patches (specular highlights).
- What did we do in traditional stereo ?
  - Picked a representation for the patches: Census transform
  - Then used a logical distance measure between the representations for the two patches: Hamming
- But these are hand-crafted. Let's try to "learn" them instead.

STEREO

- Let's say we have a training set of left-right pairs and ground truth disparity.
- We want to learn a distance function $f(p, q; \theta)$:
  - For p a patch for left image, and q and q' patches from the right image.
  - $f(p, q) < f(p, q')$ when q is the correct match to p and q' is not.

Siamese Networks

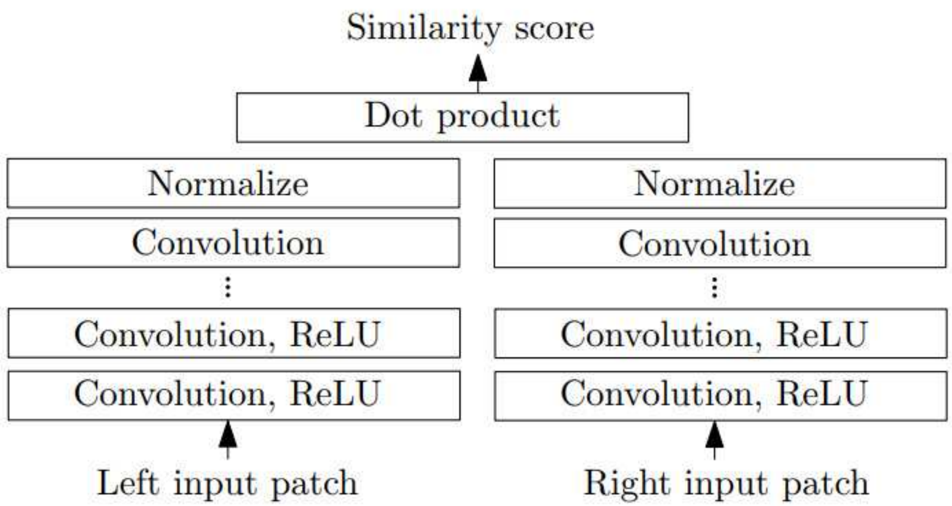
- Learn a feature representation for patches $\tilde{p} = g(p; \theta)$.
- Given two patches, let their distance $f(p, q)$ as:

$$f(p, q) = -\frac{\langle \tilde{p}, \tilde{q} \rangle}{\|\tilde{p}\| \|\tilde{q}\|}$$

- Called Siamese Network because we use the same $g(\cdot; \theta)$ to compute both $\tilde{p}$ and $\tilde{q}$.
- Loss is $f(p, q) - f(p, q')$ for "triplet" of example training pairs.

STEREO

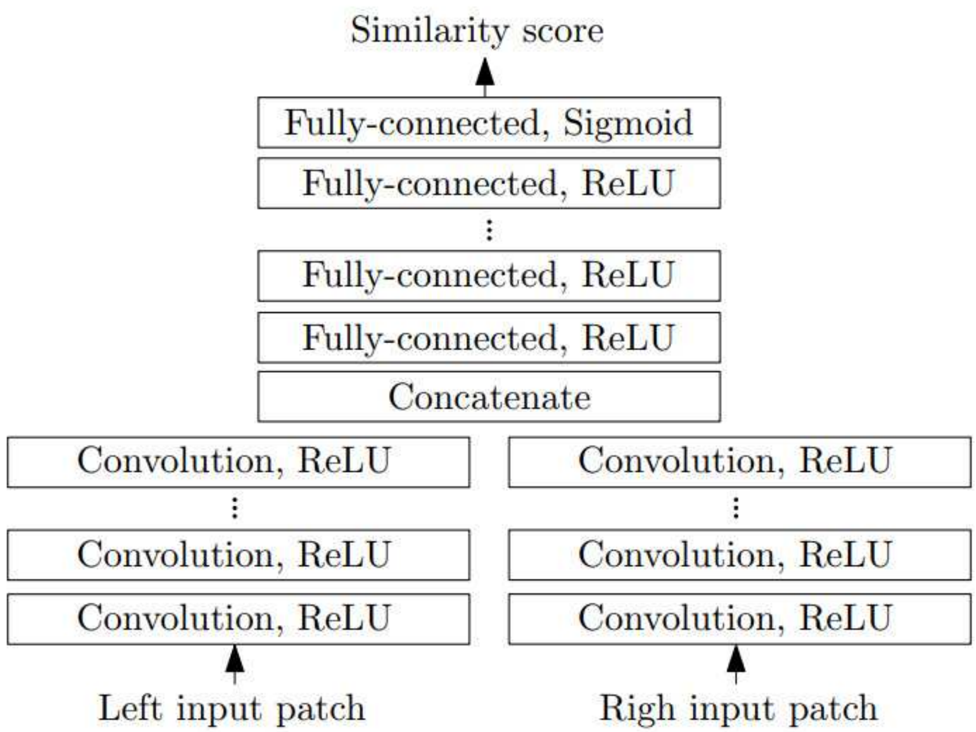
Zbontar and LeCun, CVPR 2015 / JMLR 2016.



- Layers on the left and right are "siamese twins": share weights.
- Far more accurate than census, etc. Efficient because you run the feature extraction layers only once on the left and right image. Only the dot product computed on all candidate pairs to construct the cost volume.
- But, the feature representation function gets to look only at its own patch.
- Simple dot-product on top is not sufficient to get discriminability.
- Might want to reason about: p and q match if they both at least one of two groups of common features, etc.

STEREO

Zbontar and LeCun, CVPR 2015 / JMLR 2016.



- Ver 2: Concatenate the features, and have a separate network to compute the distance function.
- More accurate, but more computationally expensive. Layers on top need to be re-computed for all pairs.

STEREO

- Zbontar & Lecun only used CNNs to compute matching cost to build cost-volume.
- Then, they simply used SGM on this cost volume to compute the disparity map.
- But even SGM uses some hand-crafted notion of smoothness, and an approximate optimization algorithm to enforce that smoothness.
- Can we do this "globalization" step also by neural networks ?

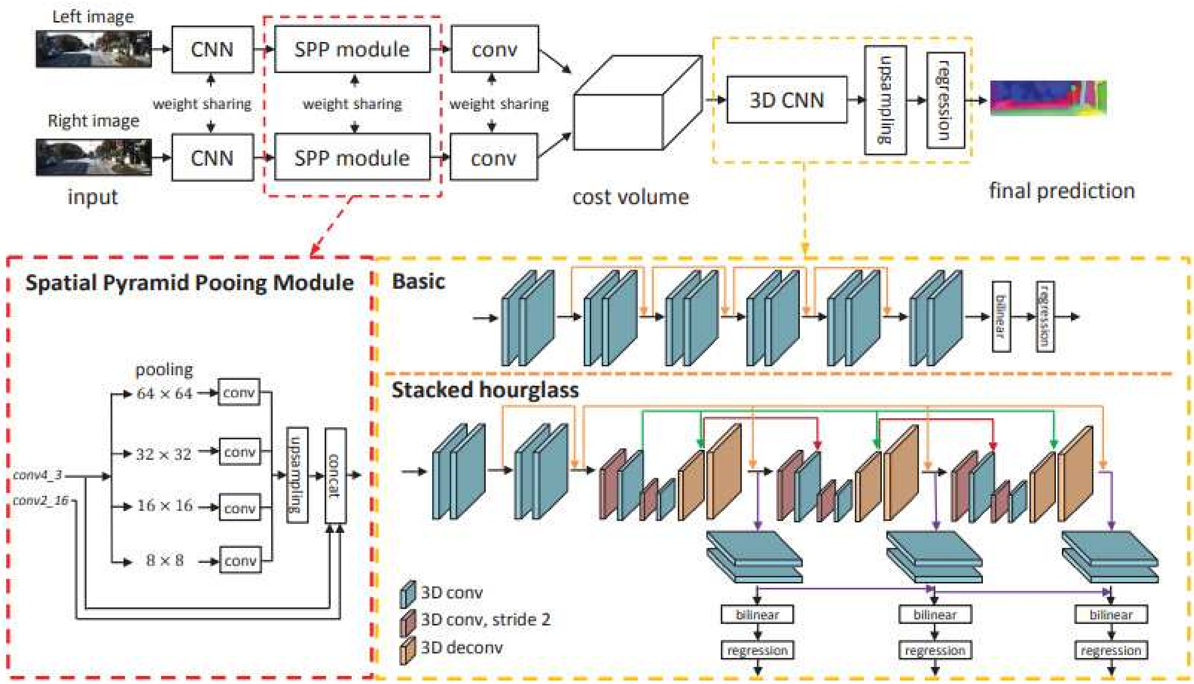
STEREO

- Modern methods compute matching costs *and* do globalization using CNNs.
- For globalization, remember you need to operate on costs at neighboring disparity values.
- Use 3D convolutions.
- Cost Feature Volume is now a 5-dimenional tensor: B x H x W x D x F
- For each of B images, have an F-dimensional vector for each (y, x, d): representing information about the match of (x, y) to $(x - d, y)$.
- Convolve with a 5-D kernel $K_y \times K_x \times K_d \times F_1 \times F_2$

$$y[b, y, x, d, f_2] = \sum_{k_x} \sum_{k_y} \sum_{k_d} \sum_{f_1} x[b, y - k_y, x - k_x, d - k_d, f_1] K[k_y, k_x, k_d, f_1, f_2]$$

STEREO

Example: Chang and Chen, Pyramid Stereo Matching Network, CVPR 2018.



- Now trained based on accuracy of final disparity map.

STEREO

- We have assumed so far that we were given a ground truth disparity map.
- But ground truth depth data (aligned with stereo images) is hard to compute.
- Can we instead learn matching costs from left-right images alone ?
- Key Ideas: Use the implicit constraints from epipolar geometry to generate labels
- Match has to lie on the same "epipolar" line.
  - For a reference patch p in left image, get $q_1, q_2, \ldots$ patches from same line in the right image, and $q'_1, q'_2, \ldots$ from a different line.
  - Let your loss be that the distance of all $q'_1, q'_2, \ldots$ should be higher than the min cost of $q_1, q_2, \ldots$
- Other constraints: Ordering constraint, smoothness, etc. in the predictions

STEREO

Example: Tulyakov et al., ICCV 2017

(E) **Epipolar constraint.** Every non-occluded reference patch has a matching positive patch [19][239-241p].

(D) **Disparity range constraint.** The offset of the reference patch index with respect to the matching positive patch index is bounded by a maximum disparity d_{max}. This comes from the stereo system parameters (focal length, pixel size, baseline) and the distance range of the scenes.

(U) **Uniqueness constraint.** The matching positive patch is unique [33].

(C) **Continuity (smoothness) constraint.** The offsets of the reference patches indices with respect to the matching positive patch indices are similar for nearby reference patches everywhere except on depth discontinuities [33].

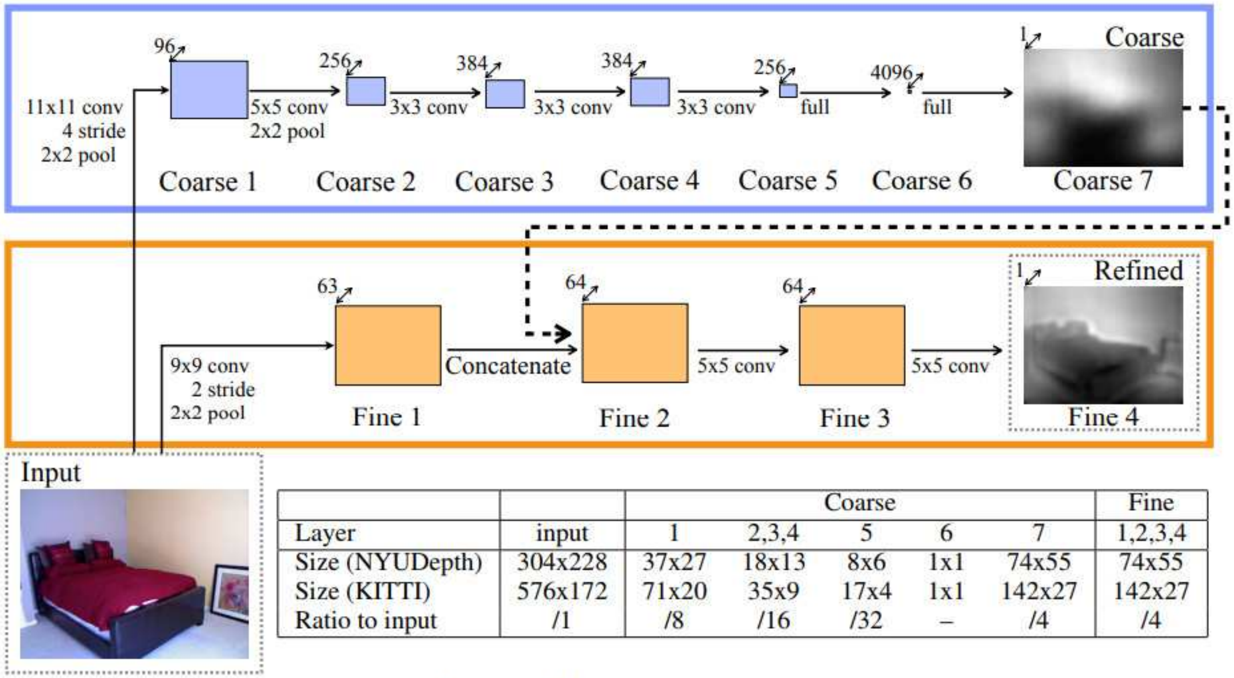
(O) **Ordering constraint.** The reference patches are ordered on their lines as the matching positive patches on theirs.

MONOCULAR DEPTH ESTIMATION

- Estimate depth from a single RGB image
- Highly ill-posed problem, no geometric or photometric information
- But, humans are able to do it
- By using cues like shading (surface normals), contours (which tell us about depth discontinuities), priors (most regions are planar), familiar objects (for which we know true size).
- Again, trained with datasets of RGB, depth pairs.

MONOCULAR DEPTH ESTIMATION

- Eigen et al., NIPS 2014.



		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.

- Architecture carefully chosen to break up tasks across scale.
- Intermediate supervision from getting a good low-resolution depth map as intermediate output.

MONOCULAR DEPTH ESTIMATION

- Wang et al., CVPR 2015.

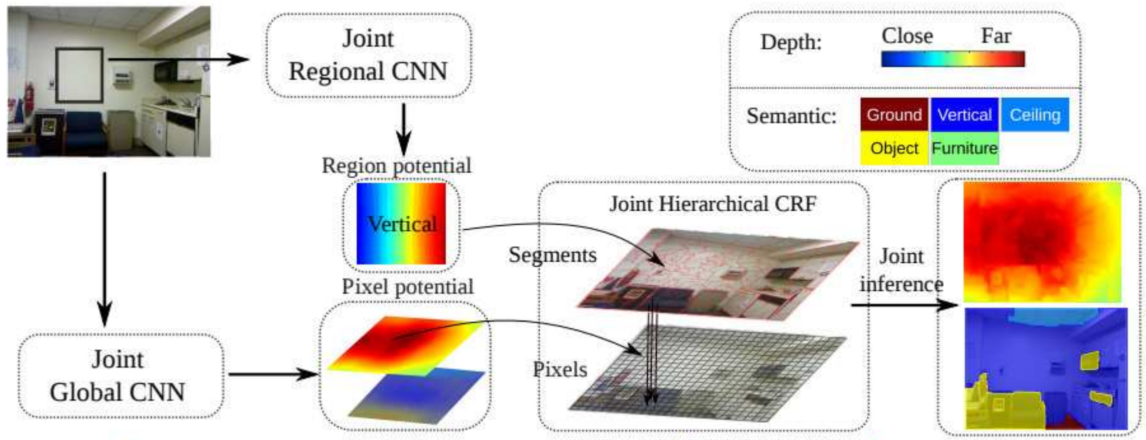


Figure 1. Framework of our approach for joint depth and semantic prediction. As described in Sec. 1, given an image, we obtain region-wise and pixel-wise potential from a regional and a global CNN respectively. The final results are jointly inferred through the Hierarchical CRF. We keep the color legend consistent in the paper.

- Predict pixel-wise probabilities of depth (as well as semantic information).
- Then use a graphical model on top to get final depth map.

MONOCULAR DEPTH ESTIMATION

- Chakrabarti et al., NIPS 2016.

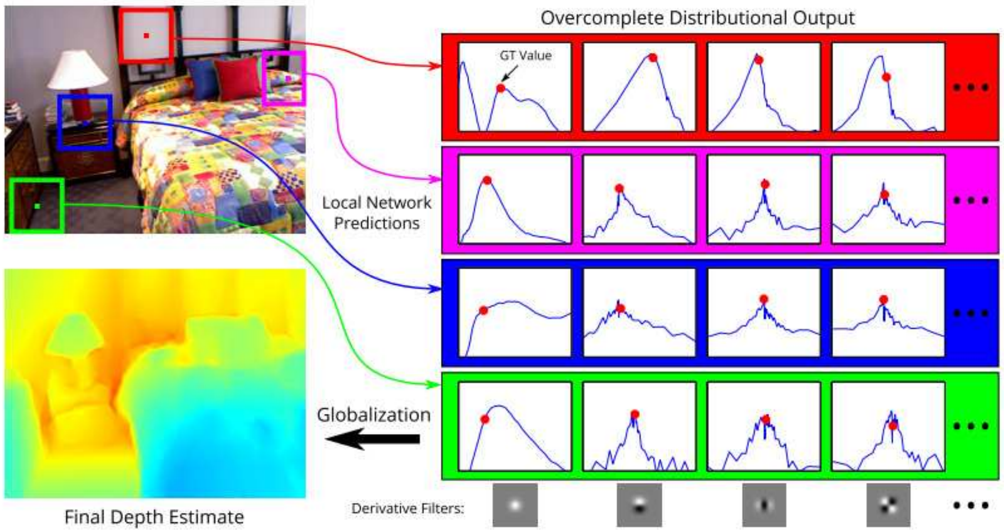


Figure 1: To recover depth from a single image, we first use a neural network trained to characterize local depth structure. This network produces distributions for values of various depth derivatives—of different orders, at multiple scales and orientations—at every pixel, using global scene features and those from a centered local image patch (top left). A distributional output allows the network to determine different derivatives at different locations with different degrees of certainty (right). An efficient globalization algorithm is then used to produce a single consistent depth map estimate.

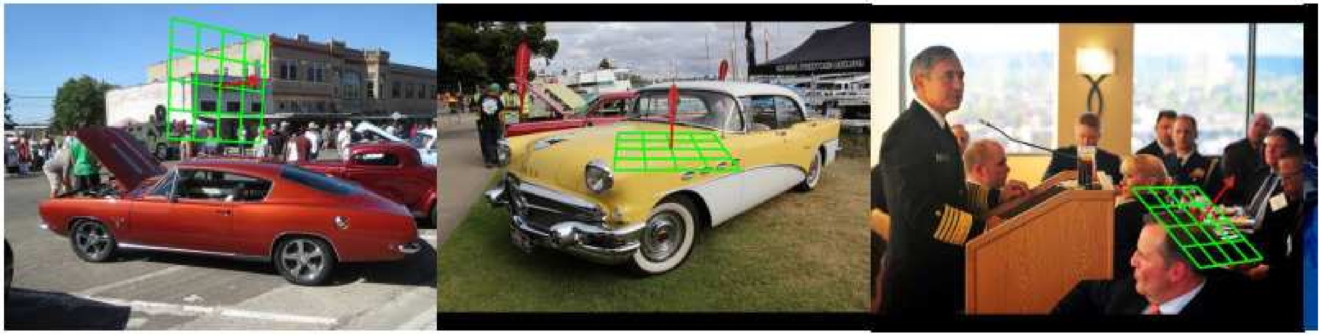
- Predict distributions over different derivatives of depths at all locations.
- At different locations, network will be sure about some derivatives but not others.
- Combine all this information to produce a depth map.

MONOCULAR DEPTH ESTIMATION

- But again, depth data is hard to collect.
- Currently, ground truth computed using depth sensors (like Kinect) that only work indoors. Or mounted on cars, to get depth for road scenes.
- These datasets are severely biased towards certain scene types.
- Want to collect larger dataset perhaps using human annotation.
- Problem: Hard to get humans to predict metric depth.
- But they can be good at estimating surface orientation.

MONOCULAR DEPTH ESTIMATION

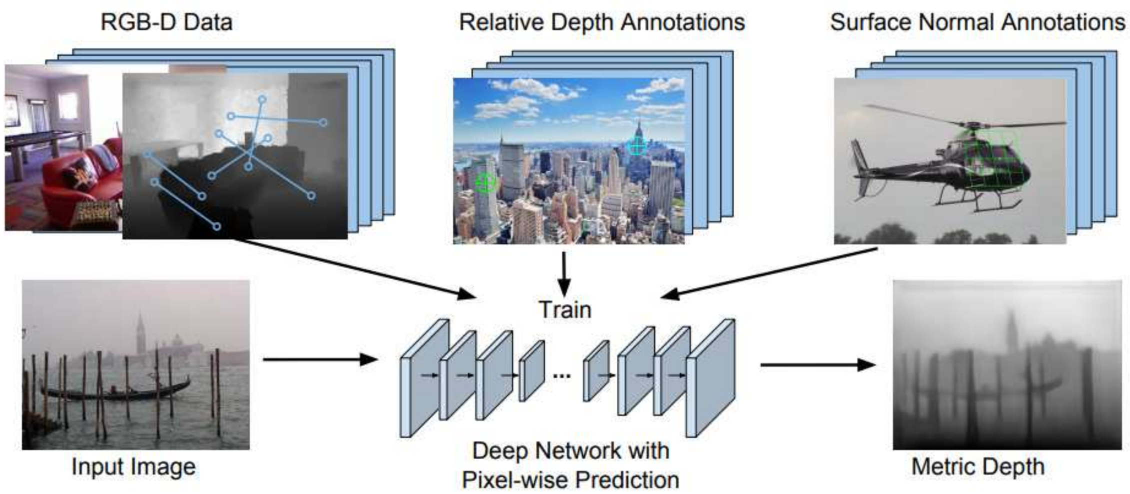
Chen et al., ICCV 2017.



- Build an interface to let user input surface orientation.

MONOCULAR DEPTH ESTIMATION

Chen et al., ICCV 2017.



- Use orientation, relative depth ordering info (behind, in-front) and actual depth supervision in combination