DeepStereo: Learning to Predict New Views from the World's Imagery

Example video

Deep networks

- Successful in:
  - Recognition problems
  - Classification problems
- Limited in:
  - Graphics problems

Deep networks

- Traditional approaches
  - Multiple complex stages
  - Careful tuning
  - Can fail in unexpected ways
- DeepStereo
  - Trained end-to-end
  - Pixels from neighboring views of a scene are presented to the network
  - Network produces pixels of the unseen view

DeepStereo

- Benefits
  - Generality: only requires posed image sets and can easily be applied on different domains
  - High quality results (on difficult scenes)
- Generate pixels (automatically from training data) acording to
  - Color
  - Depth
  - Texture priors

New view synthesis

- Form of image-based rendering
- Used in:
  - Cinematography
  - Virtual reality
  - Teleconferencing
  - Image stabilization
  - 3-dimensionalizing monocular film footage

New view synthesis

- Is challenging and underconstrained
  - Exact solution requires full 3D knowledge of all visible geometry
  - Visible surfaces may have ambiguous geometry due to a lack of texture
- Good approaches to IBR typically require use of strong priors to fill pixels where:
  - Geometry is uncertain
  - Target color is unknown due to occlusions

New view synthesis

- New approach
  - Uses deep networks to regress directly to output pixel colors given the posed input images
  - Is able to interpolate between views separated by a wide baseline
- Exhibits resilience to traditional failure models
  - Graceful degradation in presence of scene motion and specularities
  - Maybe because of end-to-end

New view synthesis

- Minimal assumptions about the scene being rendered
  - Scene should be static
  - Scene should exist within a finite range of dephts
- In case requirements are violated
  - Resulting images degrade gracefully
  - Often remains visually plausible
- When uncertainty cannot be avoided
  - Blur details (much more visually pleasing results compared to tearing or repeating, especially when animated)

New view synthesis

Training data

- Abundance of readily available training data
- Set of posed images can be used (leaving one image out)
- Data mined from Google's Street View
- Variety of scenes
  - System is robust
  - System generalices to indoor and outdoor imagery

Related work

Learning depth from images

- Problem of view synthesis strongly related to problem of predicting depth or 3D shape from imaginery
- Automatic single-view methods
  - Make3D system (Saxena et al)
    - Trained data: aligned photos and laser scans
  - Automatic photo po-up (Hoiem et al)
    - Trained data: images with manually annotated geometric classes
  - Other methods:
    - Kinect data for training
    - Deep learning methods for single view depth or surface normal prediction
- Very challenging: gathering sufficient training data dificult and time-consuming

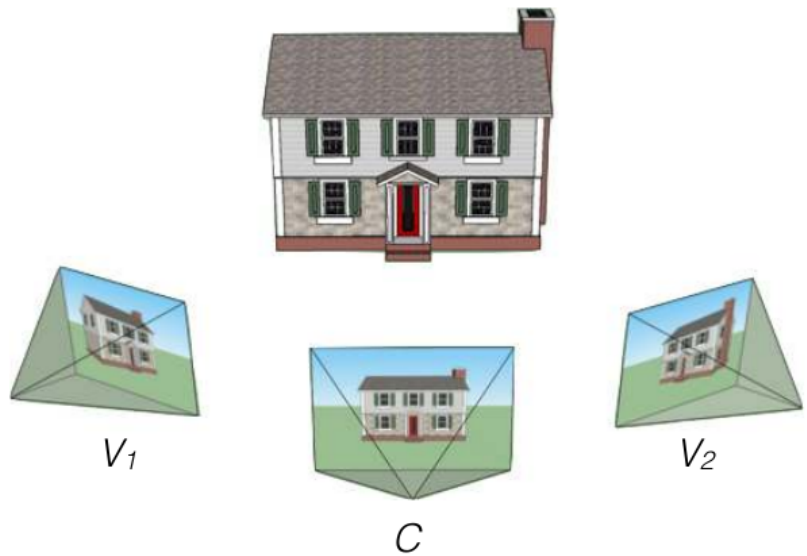
Related work

View interpolation

- Much of the recent work in this area has used a combination of 3D shape with image warping and blending
- DeepStereo uses image-based priors (inspired by Fitzgibbon)
- Goal: faithfully reconstructing the actual output image to be the key problem to be optimized
- Opposed to: reconstructing depth or other intermediate representations. Metric for stereo algorithms: image prediction error (Szeliski)

DeepStereo

- Input images: $I_1, \ldots, I_n$
- Poses: $V_1, \ldots, V_n$
- Target camera: C



DeepStereo

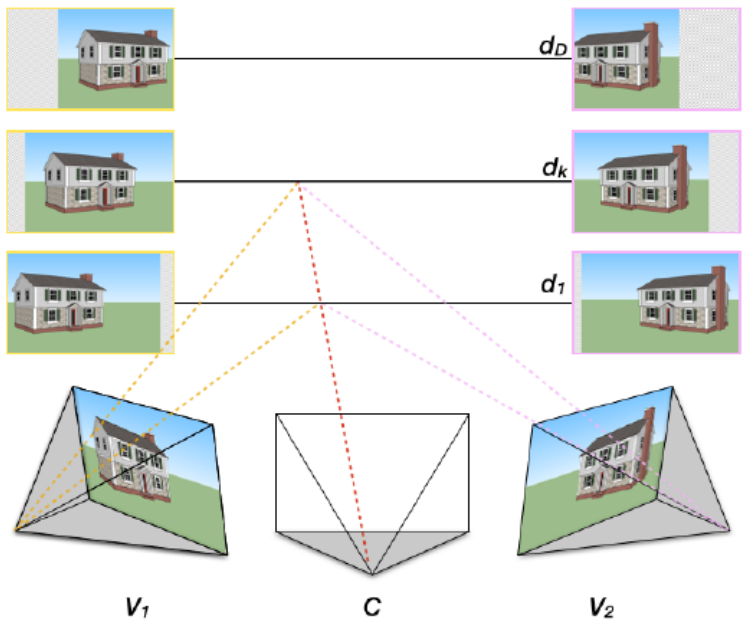
Synthesizing a new view

- Network would need to compare and combine potentially distant pixels in the original source images
  - Very dense, long-range connections.
  - Many parameters
  - Slow to train
  - Prone to overfitting
  - Slow to run inference on

DeepStereo

Plane sweep volumes

- Stack of images reprojected to the target camera C
- Depths: $d_1, \ldots, d_D$
- $V_C^k = \{P_1^k, \ldots, P_D^k\}$
- P_i^k: reprojected image I_k at depth d_i.
- $v_{i,j,z}^k$: voxel
  - R,G,B
  - A: inside or outside the field



DeepStereo

Model: two towers

- Selection tower
- Color tower
- $p_{i,j}$: pixel
- P_z: plane
- $s_{i,j,z}$: selection probability
- $c_{i,j,z}$: color probability
- Output color:

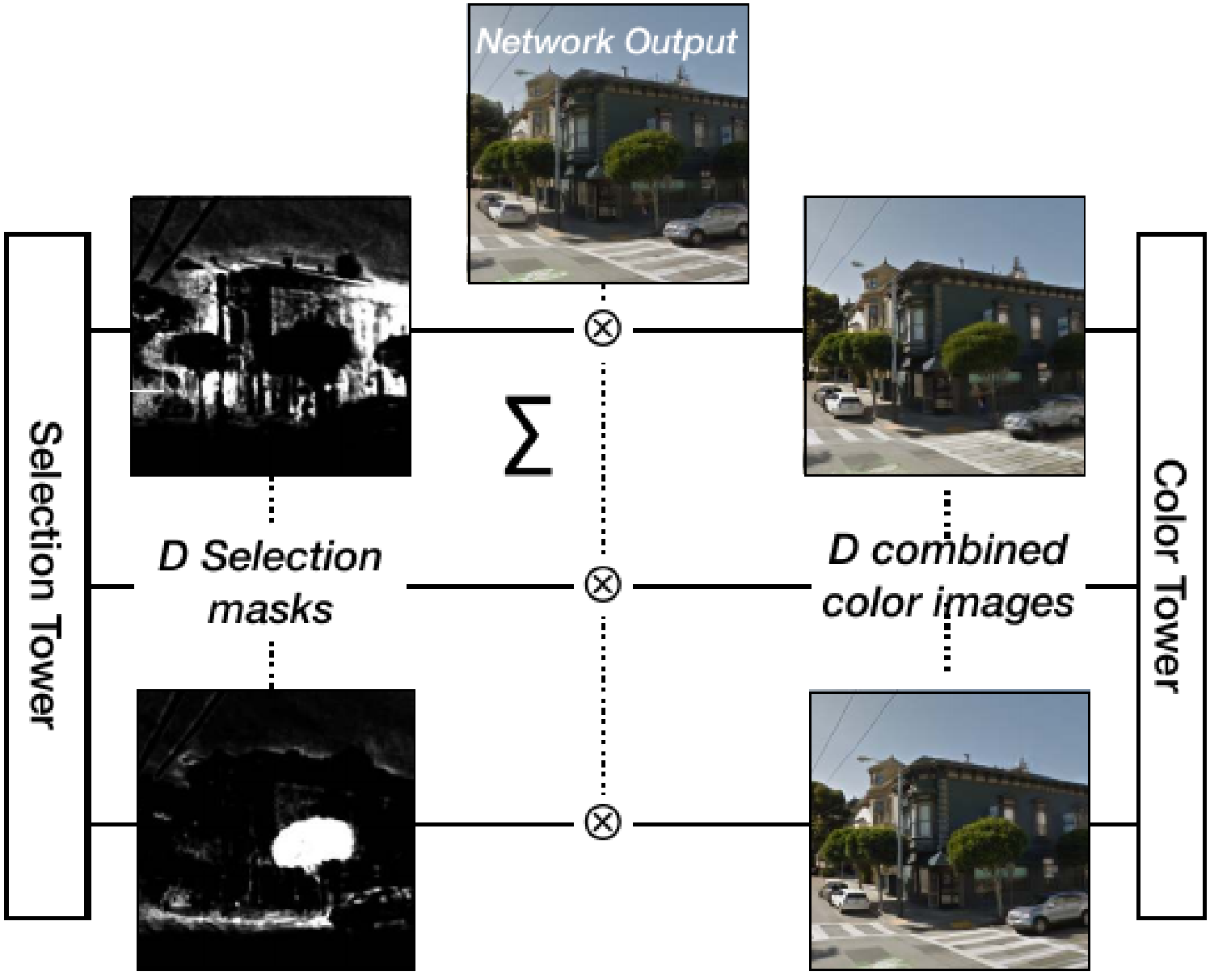
$$c^f_{i,j} = \sum s_{i,j,z} \times c_{i,j,z}$$

DeepStereo

Selection Tower

- First stage of layers
  - 2D convolutional rectified linear layers that share weights across all planes
  - Early layers compute features that are independent of depth (pixel differences)
  - Often "shut down" certain depth planes1 and never recover
- Second stage of layers
  - Connected across depth planes
  - Model interactions between depth planes (occlusion)
  - Using a tanh activation for the penultimate layer gives more stable training than the more natural choice of a linear layer
- Third stage of layers
  - Per-pixel softmax normalization transformer over depth
  - Encourages the model to pick a single depth plane per pixel
  - Ensures that the sum over all depth planes is 1
- Output: $s_{i,j,z}$

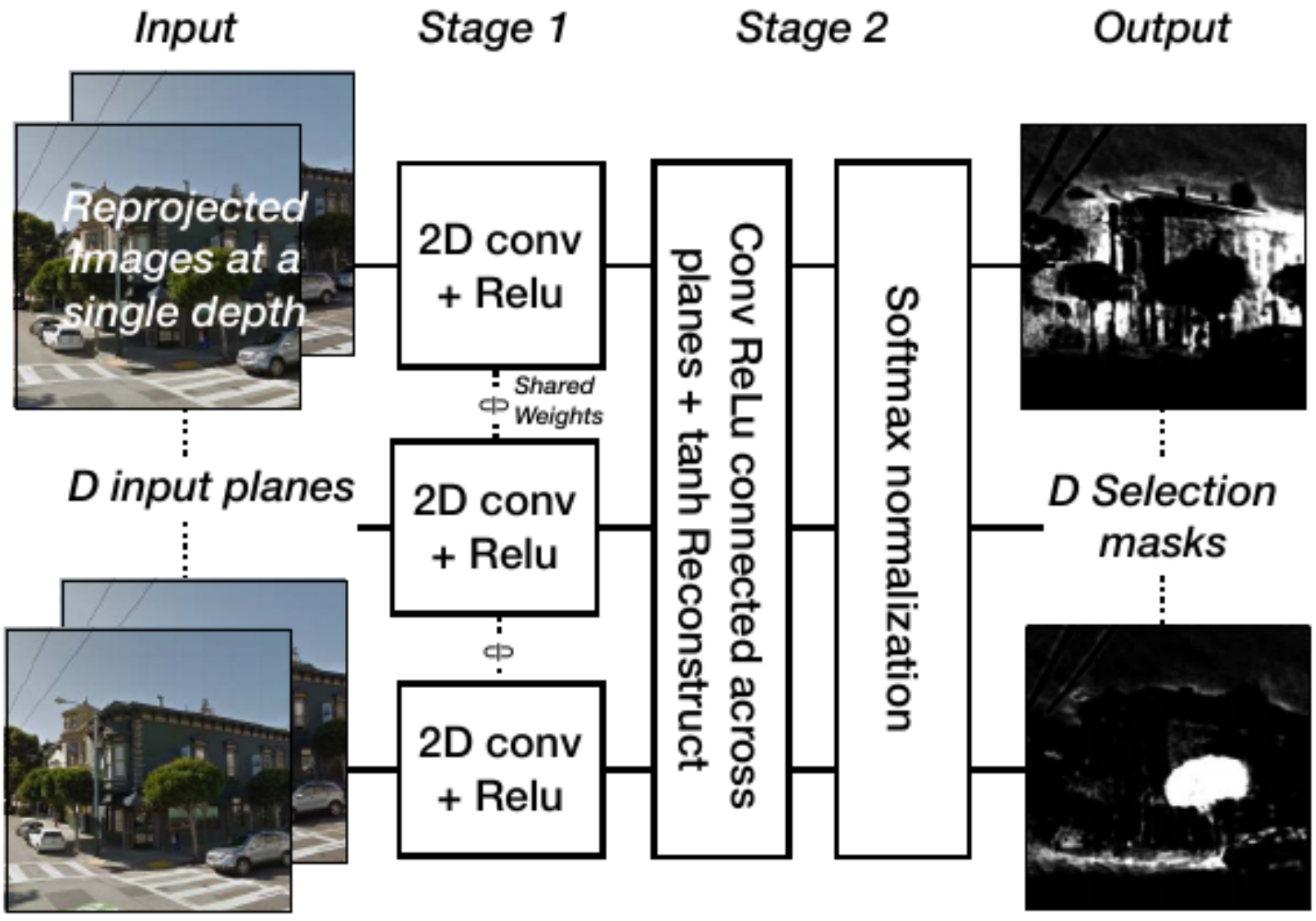
$$\sum_{z=1}^{D} s_{i,j,z} = 1$$


Network Output
Selection Tower
D Selection masks
D combined color images
Color Tower

DeepStereo

Color Tower

- 2D convolutional rectified linear layers that share weights across all planes
- Linear reconstruction layer
- No across-depth interaction is needed (occlusion effects not relevant)
- Output: 3D volume of nodes $c_{i,j,z}$ (channels R, G, B).


Input
Stage 1
Stage 2
Output
Reprojected Images at a single depth
D input planes
2D conv + Relu
Shared Weights
2D conv + Relu
2D conv + Relu
Conv ReLu connected across planes + tanh Reconstruct
Softmax normalization
D Selection masks

DeepStereo

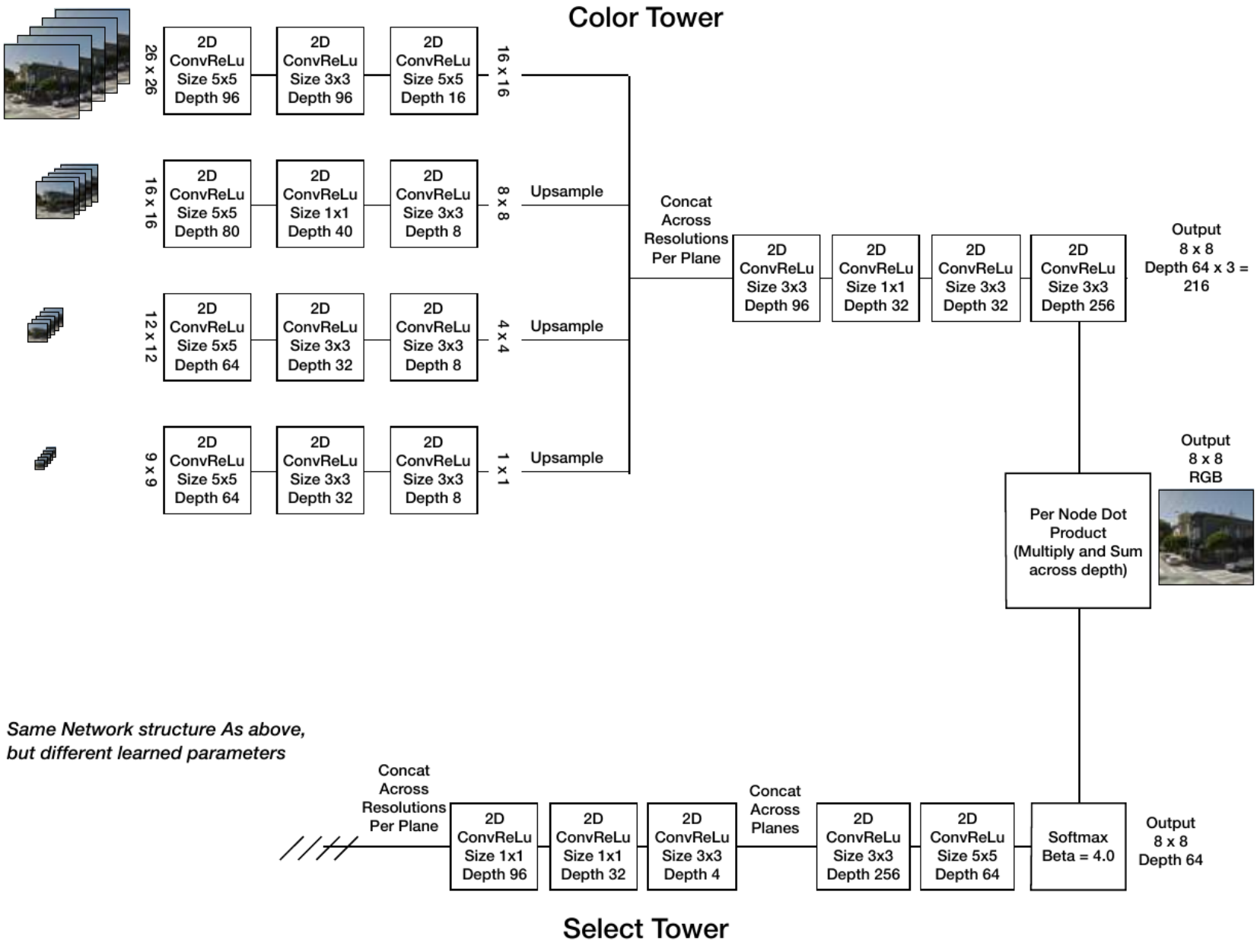
- Output image c^f produced by multiplying outputs from selection tower and color tower.
- During training the resulting image is comparedwith the known target image I^t using a per-pixel L_1 loss.
- Total loss:

$$L = \sum_{i,j} |c^t_{i,j} - c^f_{i,j}|$$

- $c^t_{i,j}$: target color at pixel i, j.

DeepStereo

- Patch-by-patch output image prediction (instead of full image at a time)
- Passing in a set of lower resolution versions of successively larger areas around the input patches helped improve results by providing the network with more context
- 4 different resolutions each of them is:
  - Processed independently by several layers
  - Upsampled (using nearest neighbor interpolation) and concatenated
  - Enters final layers


Color Tower
26 x 26
2D ConvReLu Size 5x5 Depth 96
2D ConvReLu Size 3x3 Depth 96
2D ConvReLu Size 5x5 Depth 16
16 x 16
16 x 16
2D ConvReLu Size 5x5 Depth 80
2D ConvReLu Size 1x1 Depth 40
2D ConvReLu Size 3x3 Depth 8
8 x 8
Upsample
12 x 12
2D ConvReLu Size 5x5 Depth 64
2D ConvReLu Size 3x3 Depth 32
2D ConvReLu Size 3x3 Depth 8
4 x 4
Upsample
9 x 9
2D ConvReLu Size 5x5 Depth 64
2D ConvReLu Size 3x3 Depth 32
2D ConvReLu Size 3x3 Depth 8
1 x 1
Upsample
Concat Across Resolutions Per Plane
2D ConvReLu Size 3x3 Depth 96
2D ConvReLu Size 1x1 Depth 32
2D ConvReLu Size 3x3 Depth 32
2D ConvReLu Size 3x3 Depth 256
Output 8 x 8 Depth 64 x 3 = 216
Output 8 x 8 RGB
Per Node Dot Product (Multiply and Sum across depth)
Same Network structure As above, but different learned parameters
Concat Across Resolutions Per Plane
2D ConvReLu Size 1x1 Depth 96
2D ConvReLu Size 1x1 Depth 32
2D ConvReLu Size 3x3 Depth 4
Concat Across Planes
2D ConvReLu Size 3x3 Depth 256
2D ConvReLu Size 5x5 Depth 64
Softmax Beta = 4.0
Output 8 x 8 Depth 64
Select Tower

Training

- Images of street scenes captured by a moving vehicle
- Posed using a combination of odometry and traditional structure-from-motion techniques
- vehicle captures a set of images (rosette), from different directions for each exposure
- Capturing camera uses a rolling shutter sensor
- Used approximately 100K image sets

Training

- Used a continuously running online sample generation pipeline
- Selects and reprojecs random patches from the training imagery
- 8×8 patches from overlapping input patches of size 26×26
- 96 depth planes
- To increase the variability of the patches that the network sees during training patches from many images are mixed together to create mini-batches of size 400
- Network trained with Adagrad (initial learning rate of 0.0005)

Training

- Training data augmentation was not required
- Training data selected by first randomly selecting two rosettes that were captured relatively close together (30cm)
- Then found other nearby rosettes that were spaced up to 3m away
- Selected one of the images in the center rosette as the target and train to produce it from the others

Results

Model evaluation on view interpolation

- Generated novel image from the same viewpoint as a known image captured by the Street View camera
- Despite the fact that model was not trained directly for this task, it did a reasonable job at reproducing the input imagery and at interpolating between them

Results

- Images rendered in small patches (expensive in RAM)
- 512×512 pixel image in 12 minutes on a multi-core workstation (could be reduced by a GPU implementation)

(a) Our result.

(b) Reference image.

(a) Our result.

(b) Reference image.

(a) Our result.

(b) Reference image.

Results

- Model can handle a variety of traditionally difficult surfaces (trees and glass)
- Although the network does not attempt to model specular surfaces, the results show graceful degradation in their presence
- Slight loss of resolution and the disappearance of thin foreground structures
- Partially occluded objects tend to appear overblurred
- Model is unable to render surfaces that appear in none of the inputs
- Moving objects appear blurred in a manner that evokes motion blur
- Violating the maximum camera motion assumption significantly degrades the quality of the interpolated results

Discussion

- Pros
  - It is possible to train a deep network end-to-end to perform novel view synthesis
  - DeepStereo is general and requires only sets of posed imagery
  - Results are competitive with existing image-based rendering methods, even though DeepStereo's training data is considerably different than the test sets
- Drawbacks
  - Speed (network not optimized)
  - Inflexibility of number of input images
  - Reprojecting each input image to a set of depth planes limits the resolution of the output images
  - Method requires reprojected images per rendered frame (rather than just once)

Discussion

Future work

- Explore pre-computing parts of the network and warping to new views before running the final layers
- Explore different network architectures
  - Recurrent network
  - Potential to offer real-time performance on a GPU
- Change the number of input views after training
  - Idea: choose the set of input views per pixel (may introduce discontinuities at transicion between views)
  - A more complex recurrent model may handle arbitrary numbers of input views (would likely complicate training)
- Explore the outputs of the intermediate network layers of the network
- Similar network could likely be applied to the problem of synthesizing intermediate frames in video

Thank you!