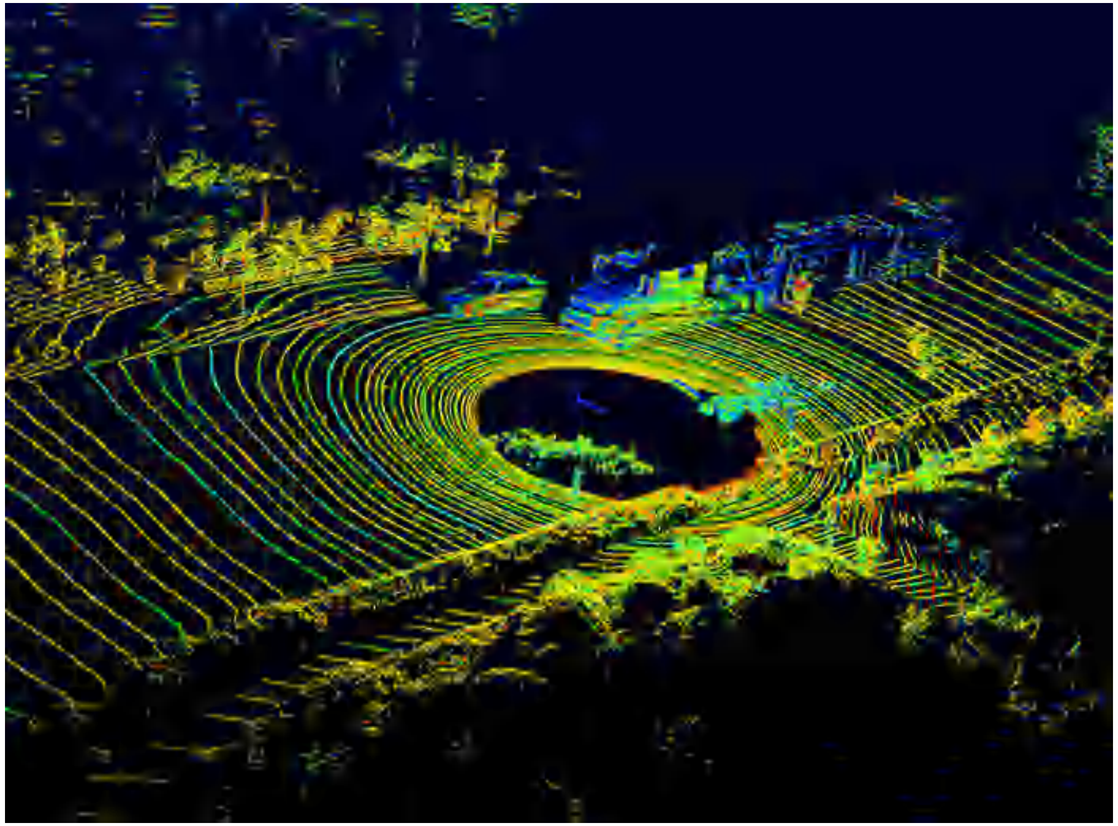
Learning to Reconstruct 3D CAD Models from A Single Image

Zihan Zhou

The Pennsylvania State University

Are Point Clouds Universal Representation?

A Gear Wheel Scanned by eviXscan 3D Pro



Streets Scanned by Velodyne Lidar

Image source: Velodyne website
Image source: “Diagnostics of machine parts by means of reverse engineering procedures”

Data Representation for Man-made Scenes

An Italian Villa

Wireframe (line drawing)

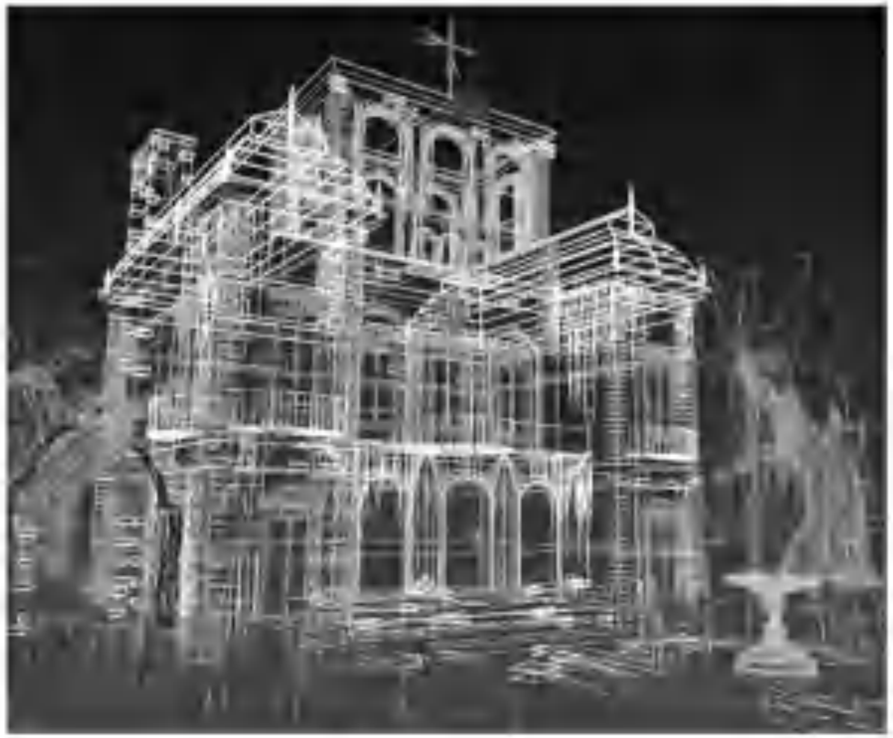
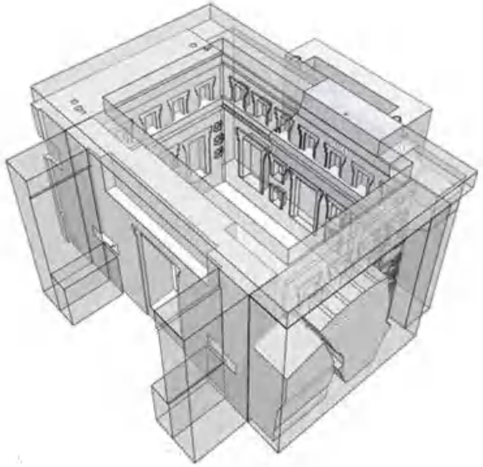


Image Source: Internet

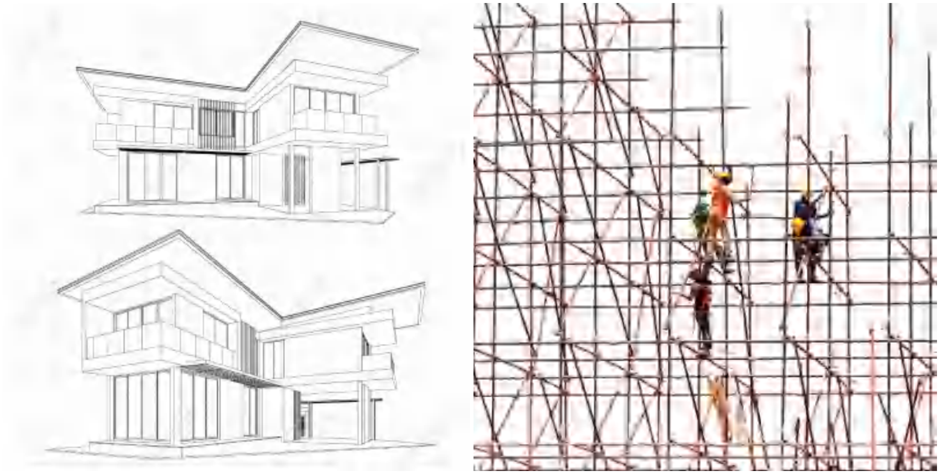
CAD Models for Architecture



- A CAD/Wireframe model encodes critical **structural information** among points, lines, patches, etc.

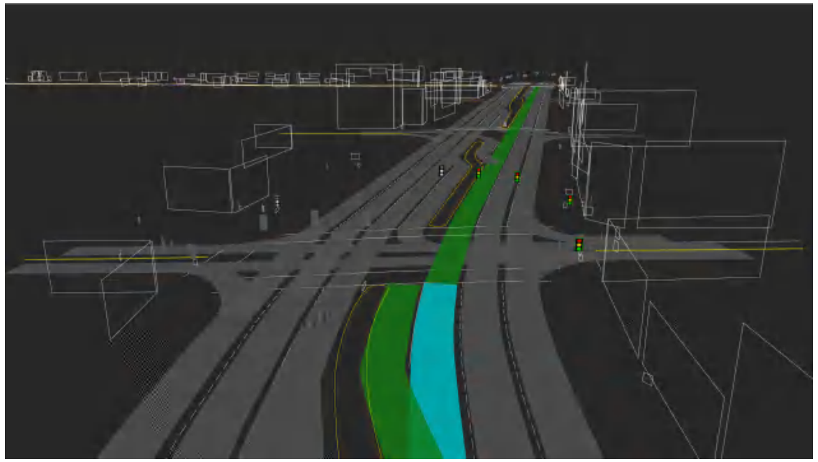
Image Source: Internet

Potential Applications



Architecture design & engineering

VR/AR



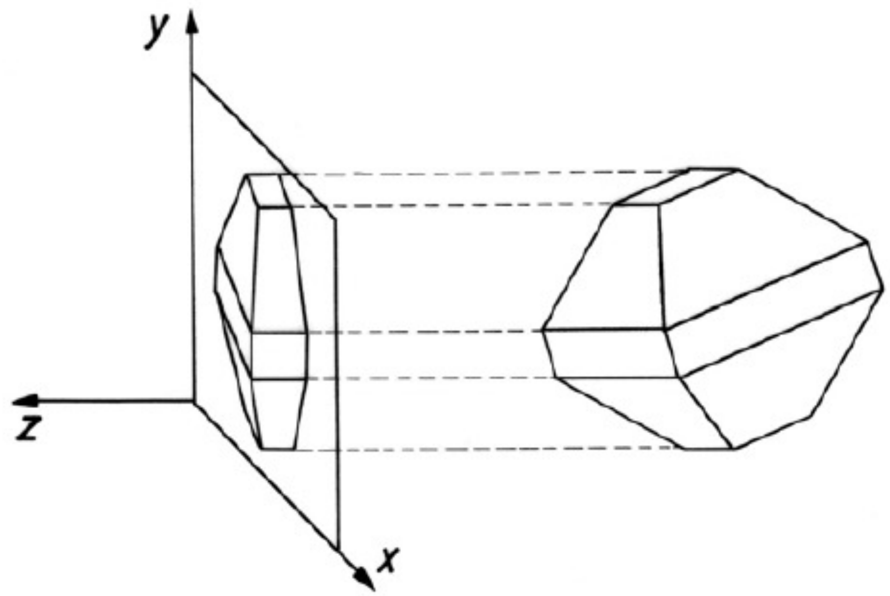
Autonomous driving

Human-robot interaction

Image Source: Internet

Early Work on the Interpretation of Line Drawings

- Consider a picture which is obtained as the orthographic projection of a polyhedral object
  - i-th vertex: (x_i, y_i, $\mathbf{z_i}$)
  - j-th face: ($\mathbf{a_j}$, $\mathbf{b_j}$, $\mathbf{c_j}$)
- 3D reconstruction can be formulated as estimating the unknowns $\mathbf{z_i}$, $\mathbf{a_j}$, $\mathbf{b_j}$, $\mathbf{c_j}$ **…**



K. Sugihara. Mathematical structures of line drawings of polyhedrons-toward man-machine communication by means of line drawings. IEEE TPAMI, 1982

Early Work on the Interpretation of Line Drawings

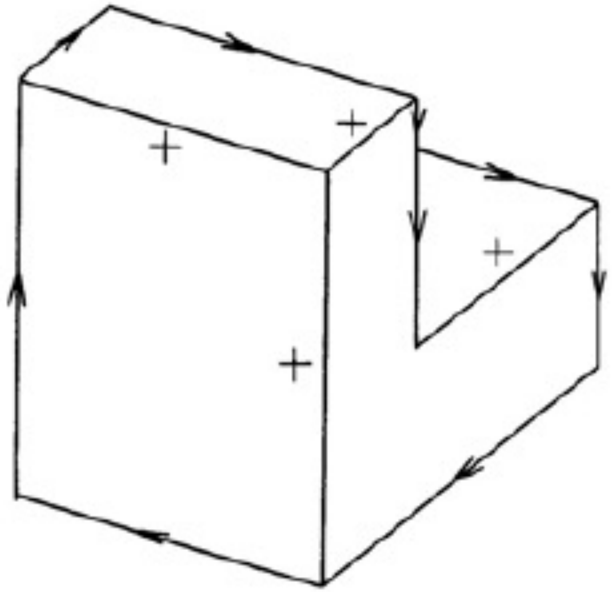
Line label assignments to the picture provide two forms of constraints:

1. Vertex i should be on the j-th face:

$$a_j x_i + b_j y_i + z_i + c_j = 0$$

2. Vertex t should be nearer than the k-th face

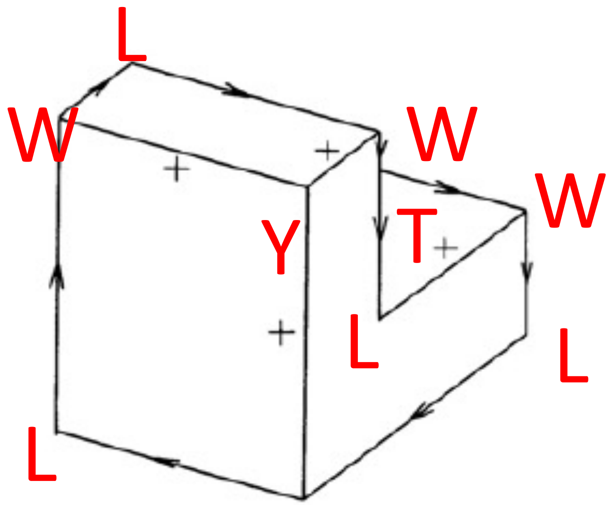
$$a_k x_t + b_k y_t + z_t + c_k > 0$$



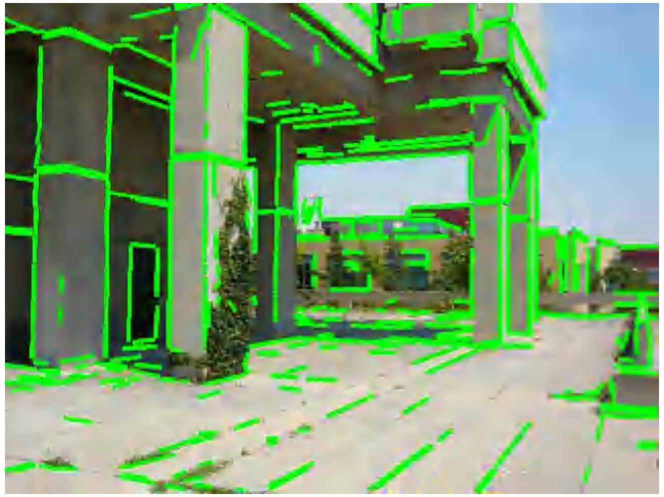
Solve the linear programming problem to get the 3D model

K. Sugihara. Mathematical structures of line drawings of polyhedrons-toward man-machine communication by means of line drawings. IEEE TPAMI, 1982

Complexity of Line-based Reconstruction in Real Images



Line drawings

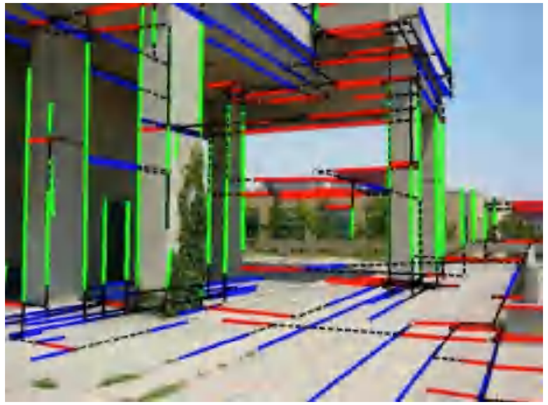


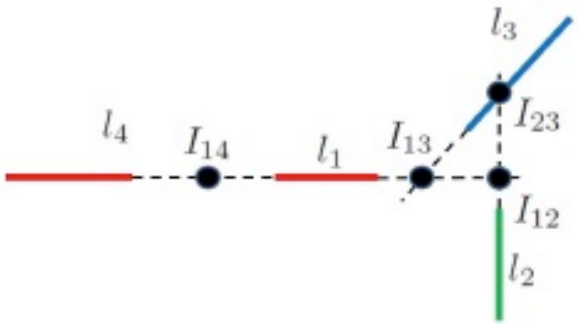
Line segments detected by LSD

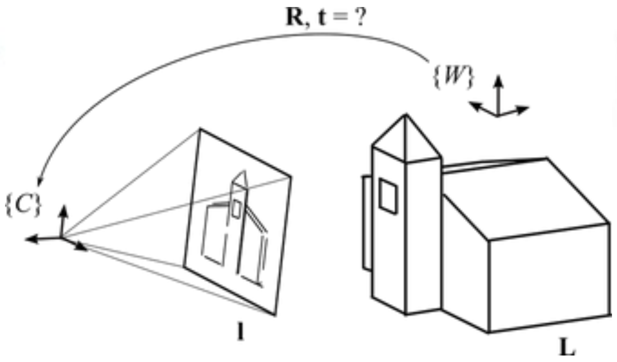
1. Real scenes are not polyhedral
2. Missing, incomplete, and spurious lines
3. No connectivity (junction) information
4. Perspective distortion & unknown camera pose

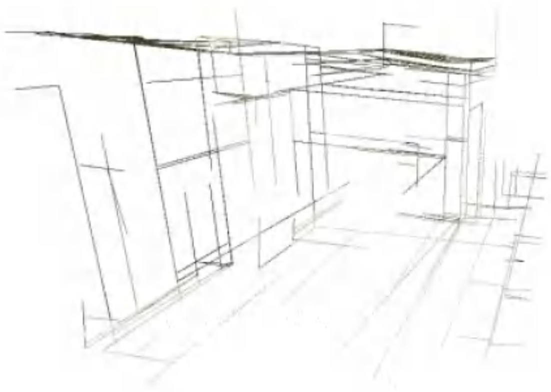
Classical (Geometric) Methods to Line-based Reconstruction

Line and vanishing point (VP) detection → Building constraint graph (junction detection) → VP-based camera pose estimation → Reconstruction via linear programming



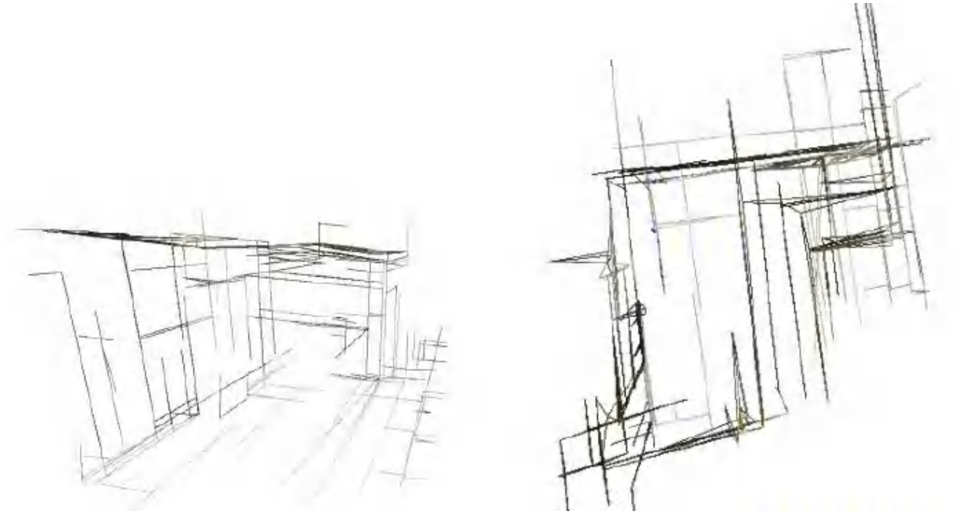




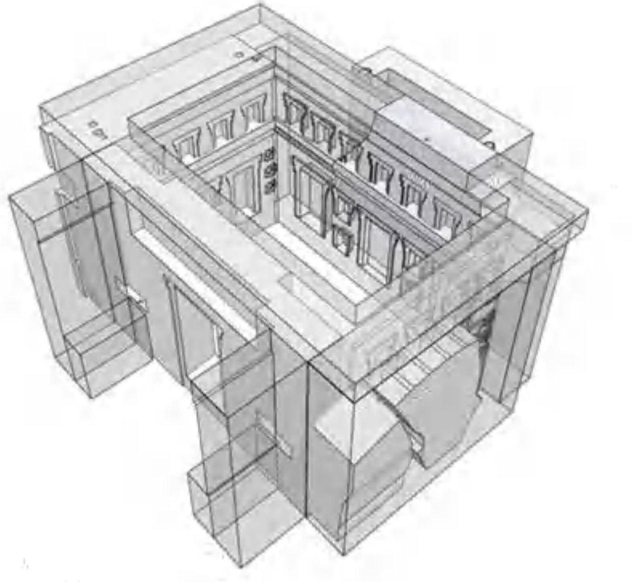


Ramalingam and Brand. Lifting 3D Manhattan Lines from a Single Image. ICCV 2013.

Problems with Multi-stage Methods



Current results



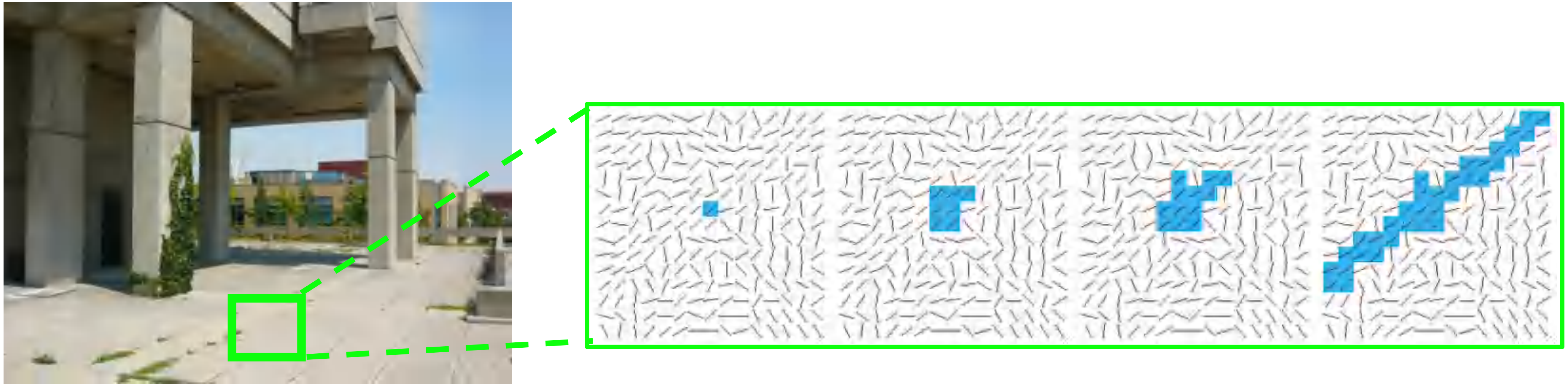
Ideal reconstruction

1. Errors accumulate over stages
2. Early stage does not exploit structure information revealed in later stages

Ramalingam and Brand. Lifting 3D Manhattan Lines from a Single Image. ICCV 2013.

Line Segment Detection

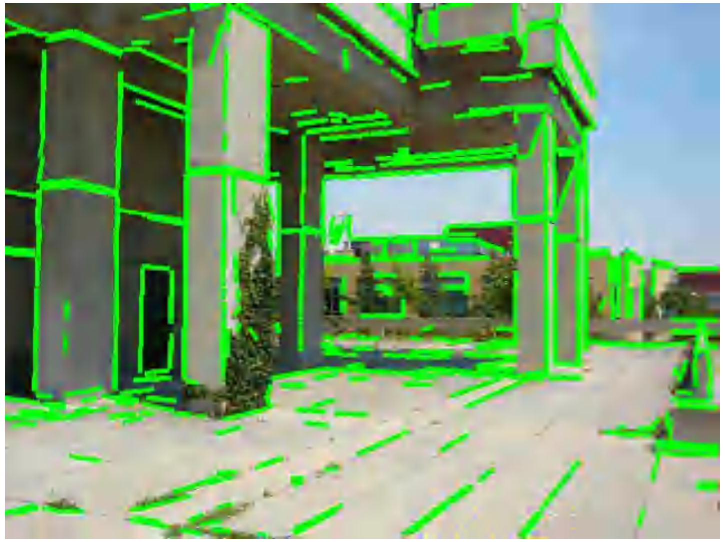
- Traditional approaches detect each line segment **independently** by grouping pixels with similar gradient orientations.



LSD: A Fast Line Segment Detector with a False Detection Control. von Gioi et. al., 2010.

Line Segment Detection

- Traditional approaches detect each line segment **independently** by grouping pixels with similar gradient orientations.

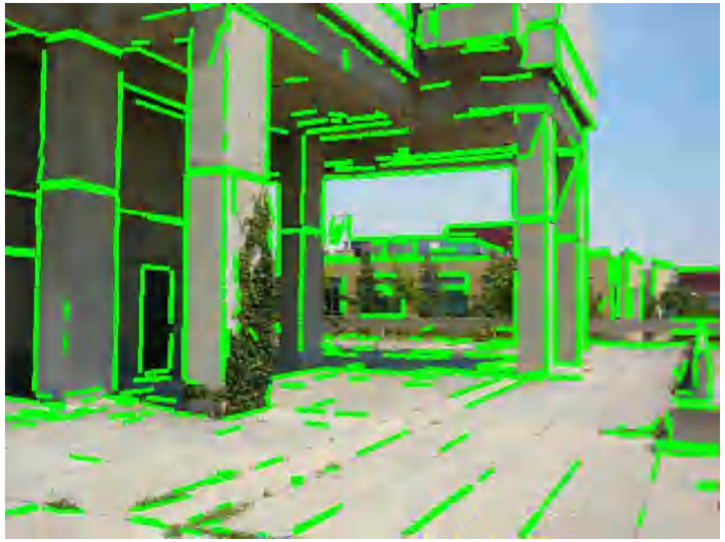
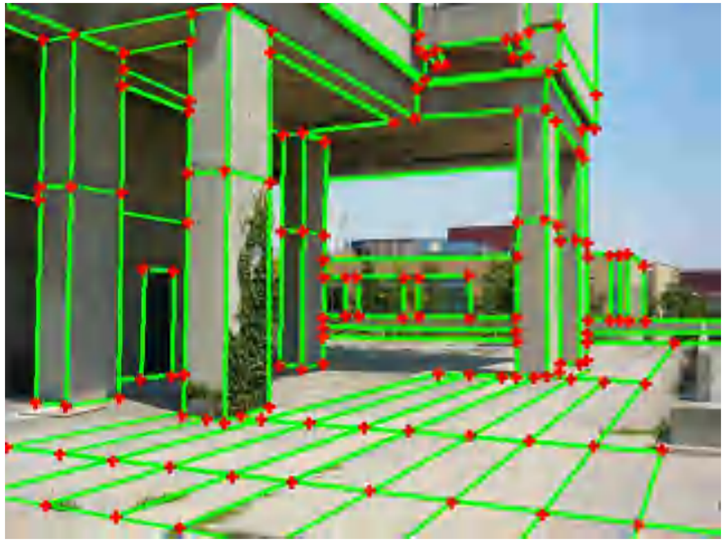
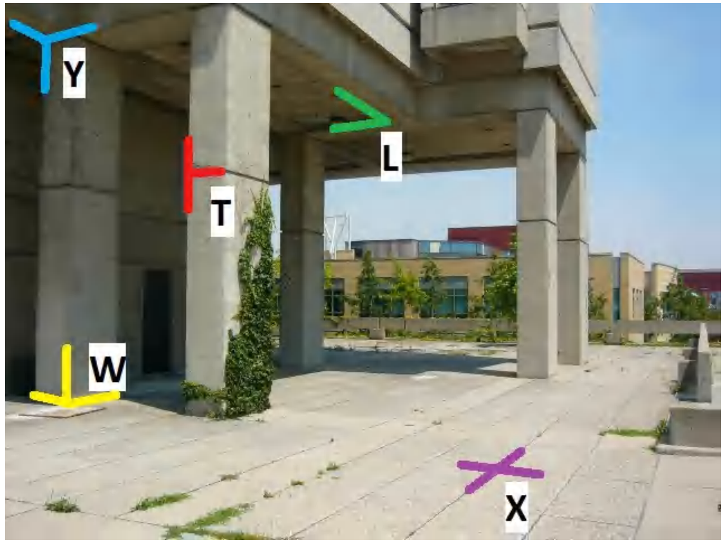


Results often contain many broken or missing line segments, as well as spurious detections.

LSD: A Fast Line Segment Detector with a False Detection Control. von Gioi et. al., 2010.

Can We Fix the Problems?

- **Yes …** If we take advantage of the incidence / intersection relationships among multiple lines (i.e., **junctions**).
- **But how?**

Google

what is the most popular ai method

All | News | Images | Videos | Shopping | More | Settings | Tools

About 262,000,000 results (0.99 seconds)

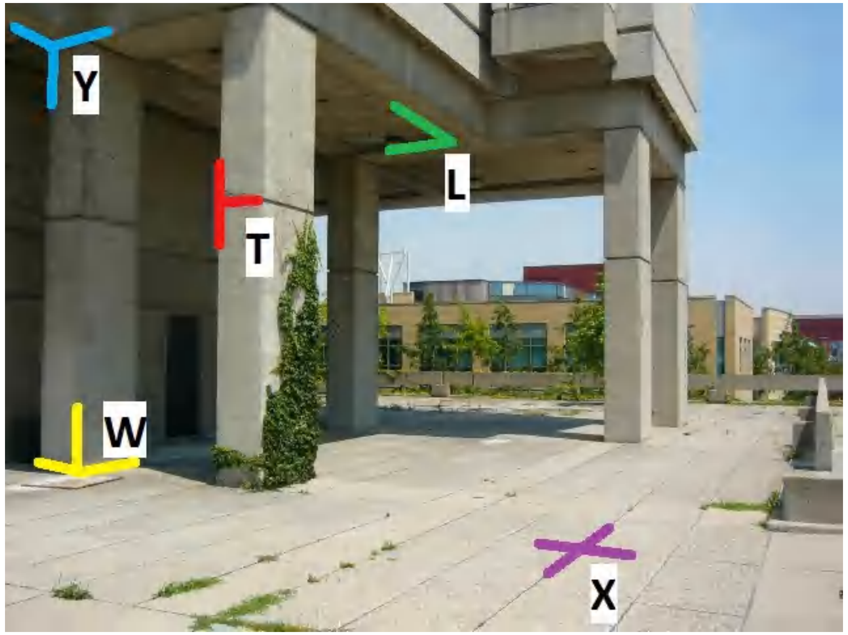
Deep Neural Networks

DNNs are among the **most** widely used **AI** and ML algorithms. Nov 8, 2018

Top 10 Most Popular AI Models - DZone AI

https://dzone.com › articles › top-10-most-popular-ai-models

Now, Some Real Inspiration

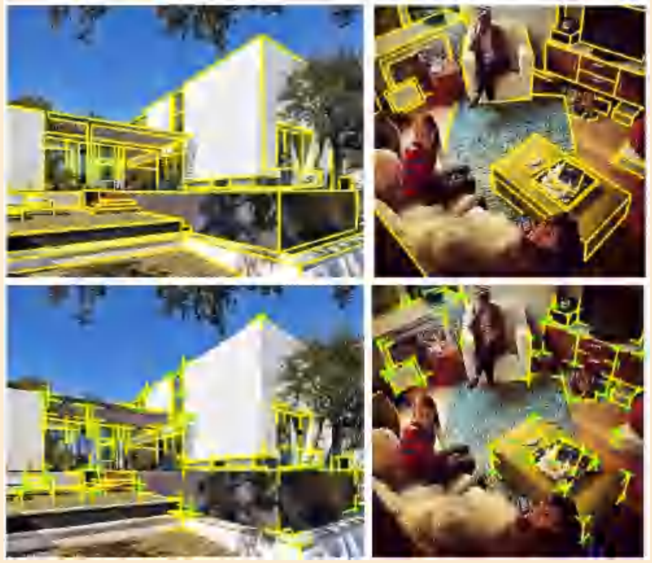


Junction detection

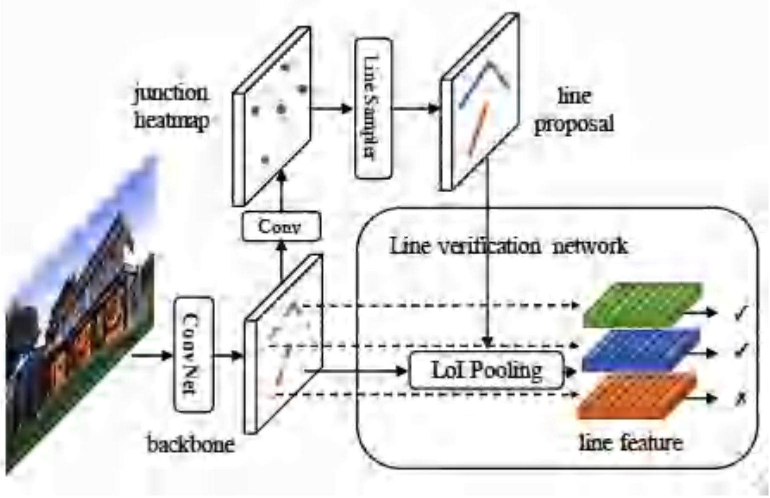
Deep learning-based text detection

Learning to Reconstruct 3D CAD Models – Three Pillars

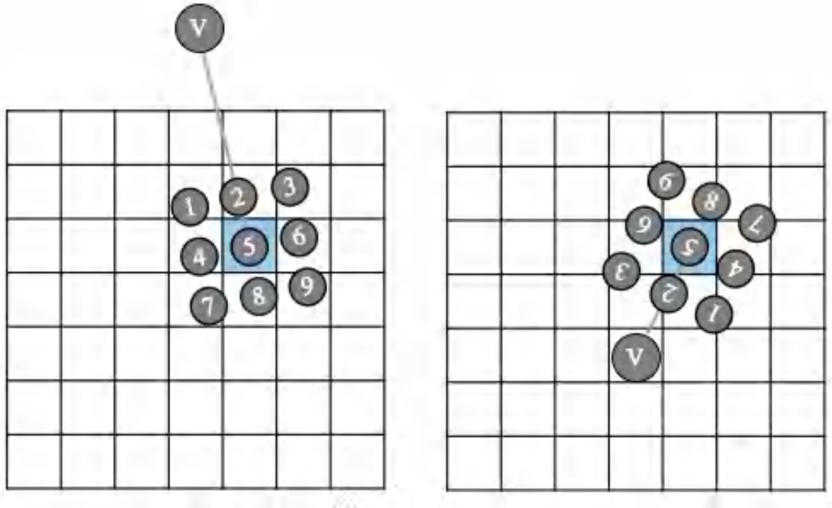
Data



Learning Scheme

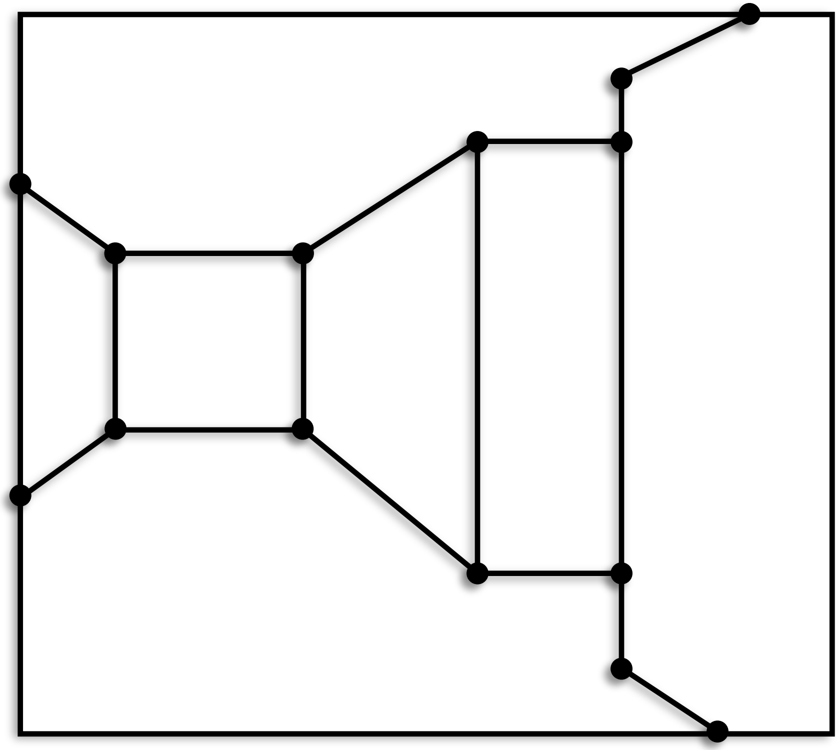


Network Design



2D Wireframe Representation

- Let $W = (V, E)$ be a wireframe
- V is the set of junctions
- $E \subseteq V \times V$ is the set of lines
- For each $\forall i \in V$
  - $\boldsymbol{p}_i$ represents its coordinate in image space

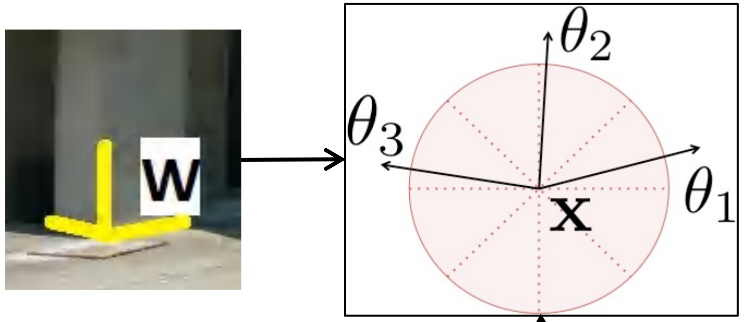


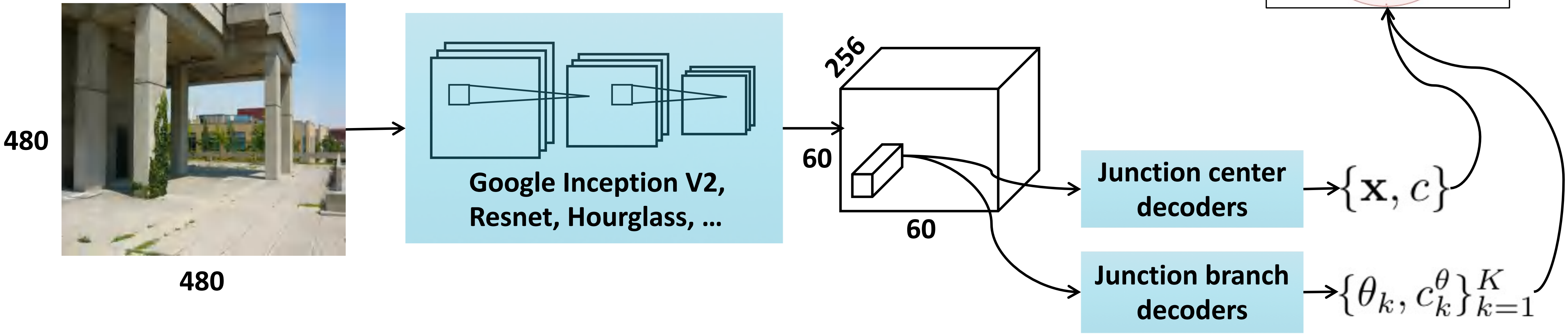
2D Wireframe Dataset

- Over 5,000 images with wireframes (junctions and lines) thoroughly labeled by humans

Junction Detection Network

- Represent a junction by its **center location** and **branch angles**.





Results: Junction Detection

Missed detections

Repeated detections

MJ (Ramalingam et. al., 2013)

ACJ (Xia et. al., 2014)

WireframeParser

False prediction in textured area

Results: Junction Detection

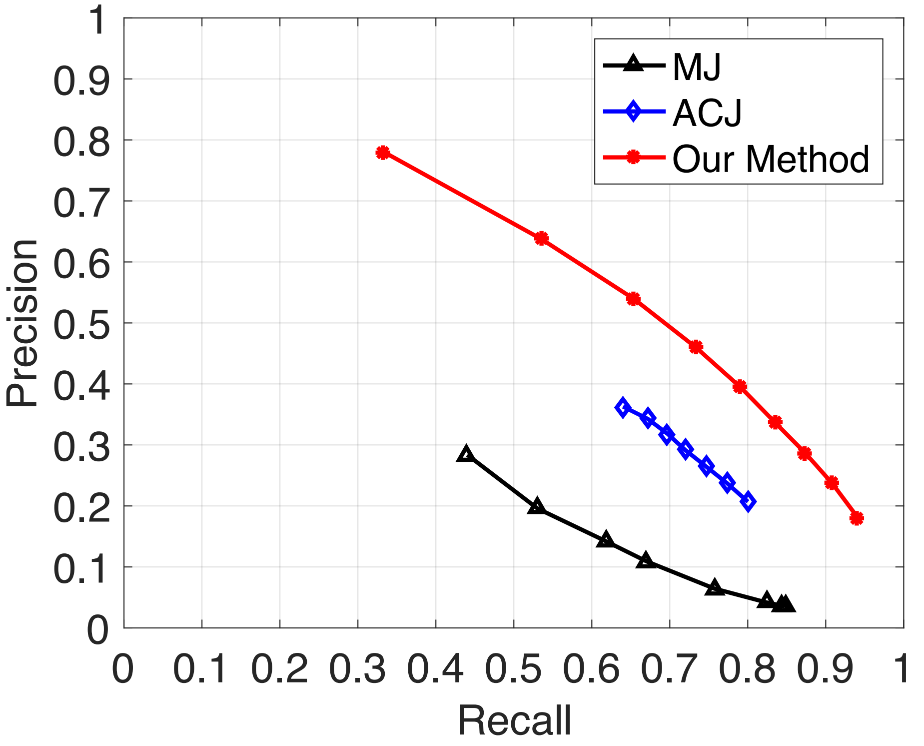
MJ (Ramalingam et. al., 2013)

ACJ (Xia et. al., 2014)

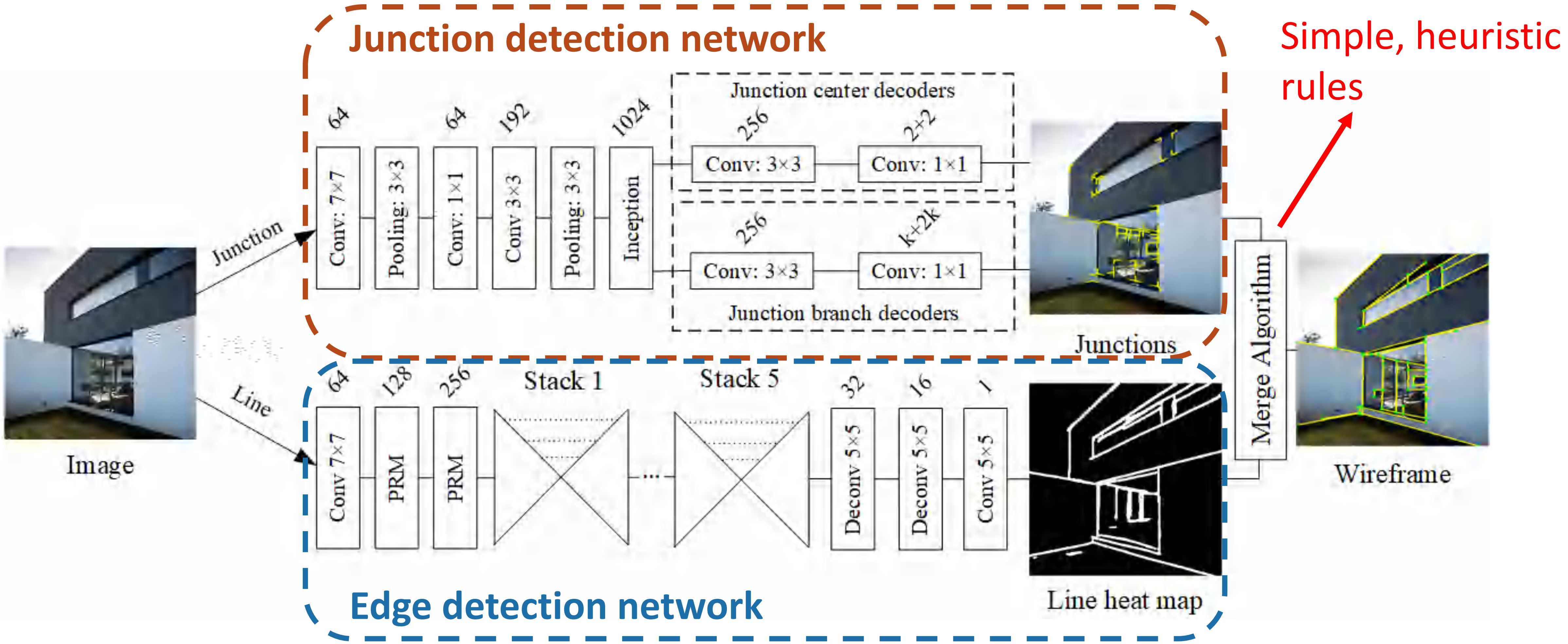
WireframeParser

Results: Junction Detection

- **Learning-based** method significantly outperforms existing **geometric** methods

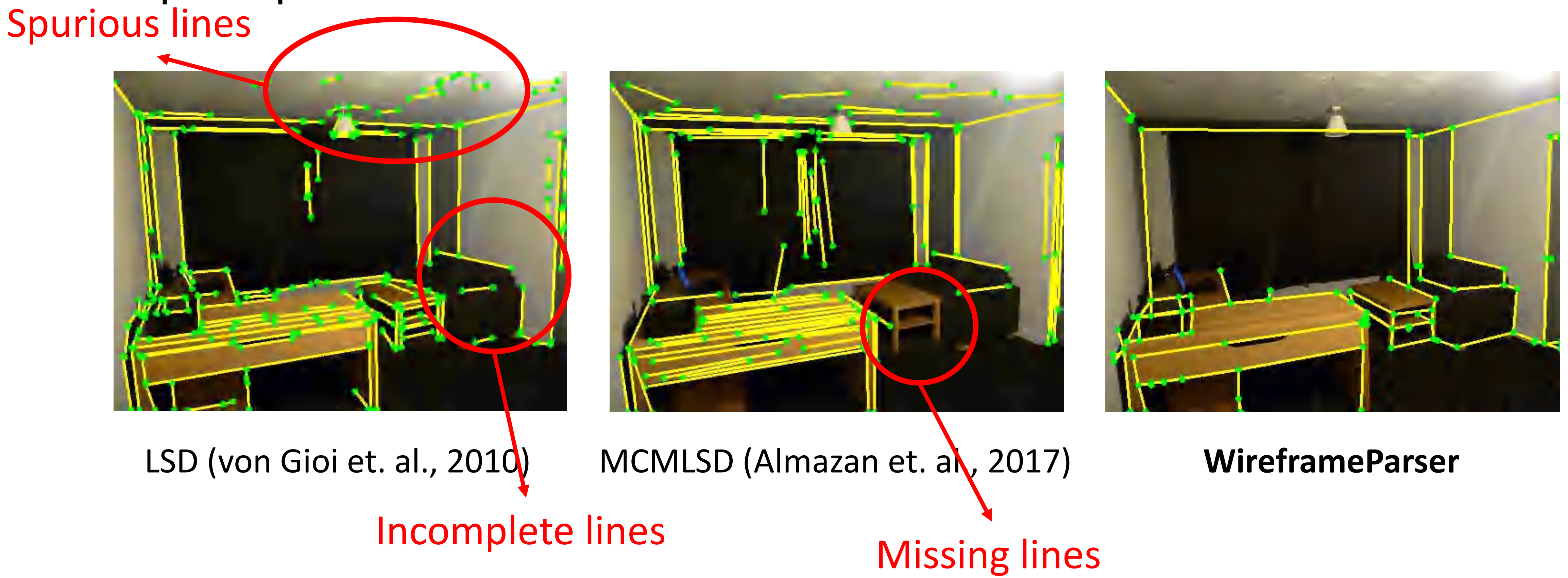


2D Wireframe Detection -- Overall Pipeline



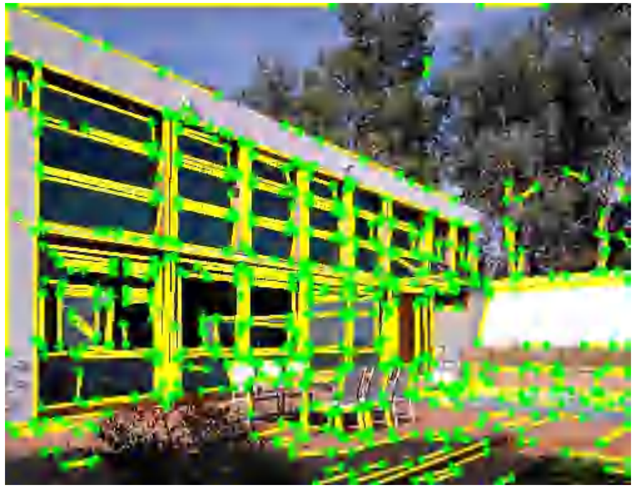
Results: Wireframe Detection

- **Learning-based** method produces results that better match with human perception of the room structure.

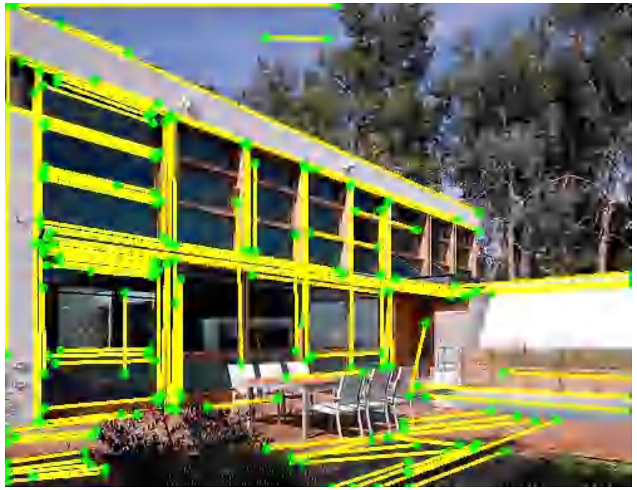


Results: Wireframe Detection

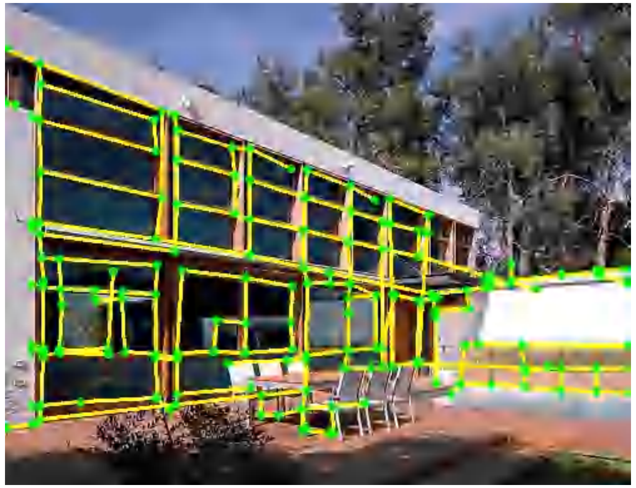
- **Learning-based** method produces results that better match with human perception of the room structure.



LSD (von Gioi et. al., 2010)



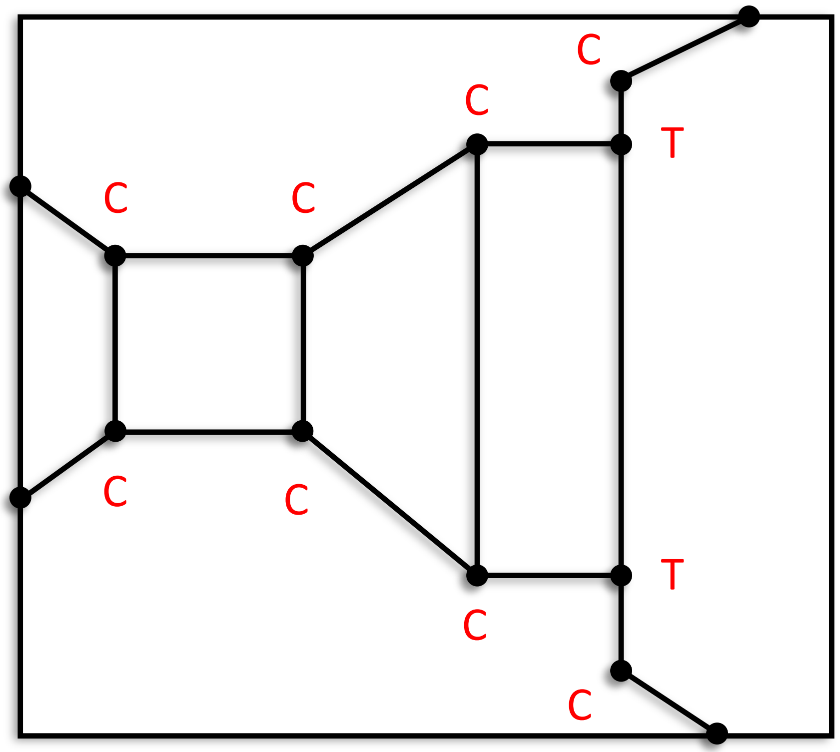
MCMLSD (Almazan et. al., 2017)



WireframeParser

Lifting Wireframe to 3D

- Let $W = (V, E)$ be a wireframe
- V is the set of junction indices
- $E \subseteq V \times V$ is the set of lines
- For each $\forall i \in V$
  - $\boldsymbol{p}_i$ represents its coordinate in image space
  - $\boldsymbol{z}_i$ represents its depth in camera space
  - $\boldsymbol{t}_i \in \{C, T\}$ represents its type

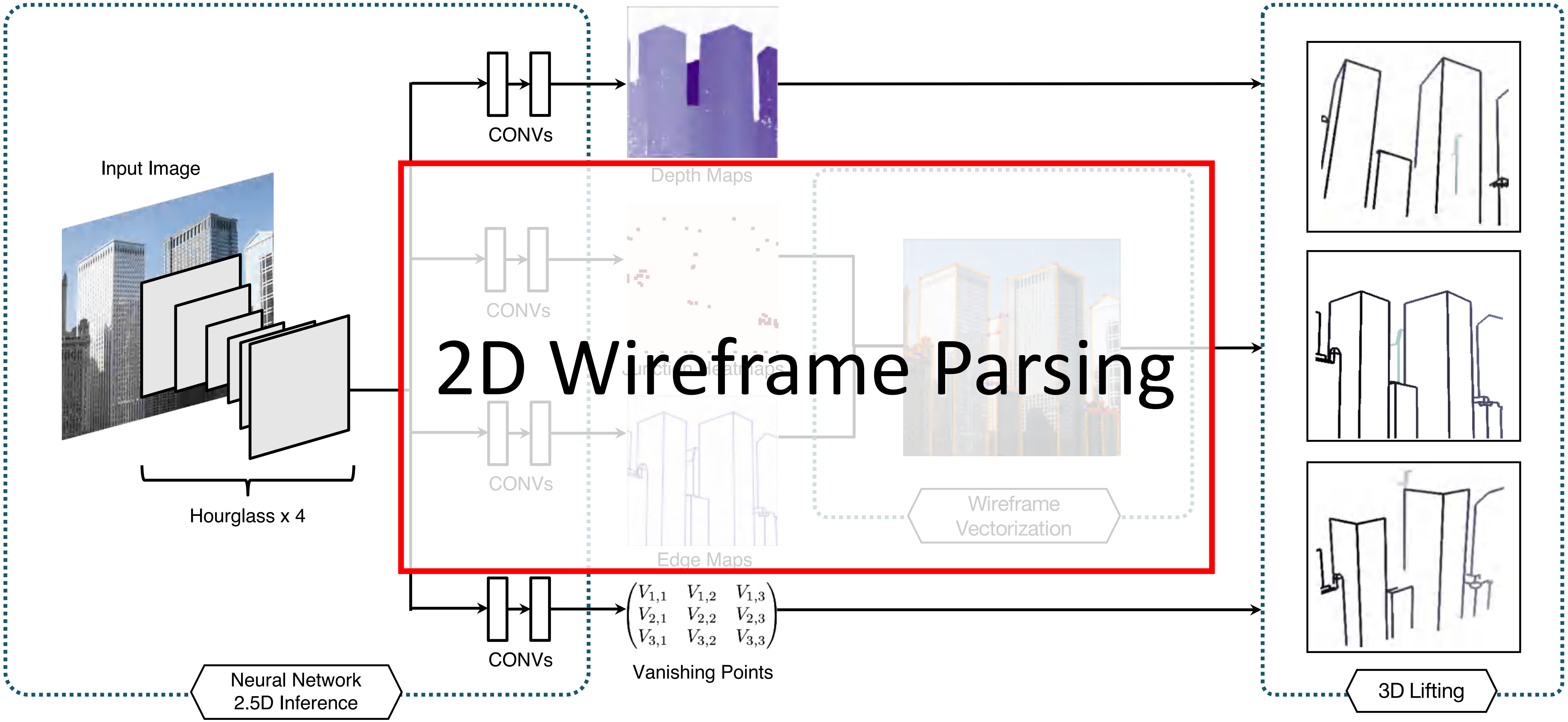


3D Wireframe Datasets

- Required information
  - Junctions
    - Location x, y and depth z
    - Corner C-Junction (Red)
    - Occlusional T-Junction (Blue)
  - Lines (Junction Connectivity)

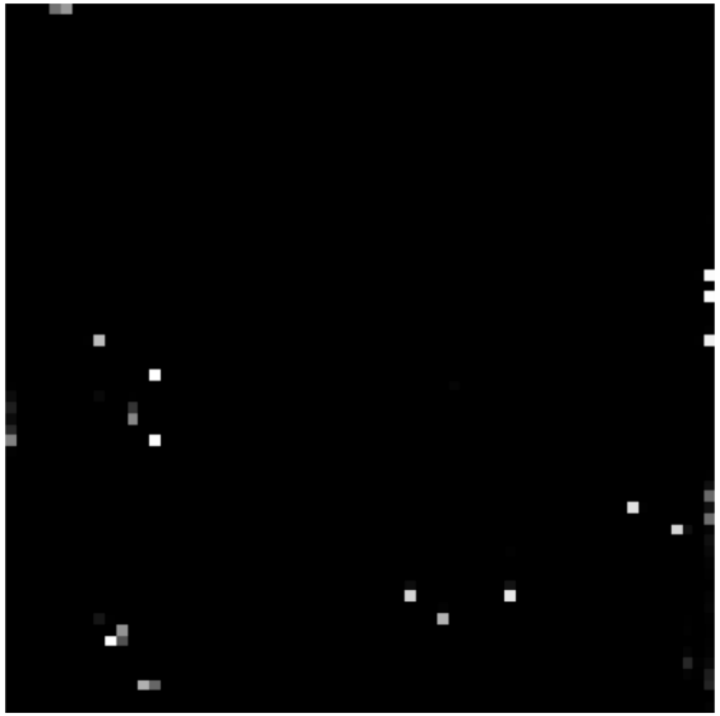
- Synthetic Urban 3D Dataset (23k Images)
  - Procedural building generation
  - Photo-realistic rendering using Blender
  - Ground truth wireframes from OpenGL and computation geometry

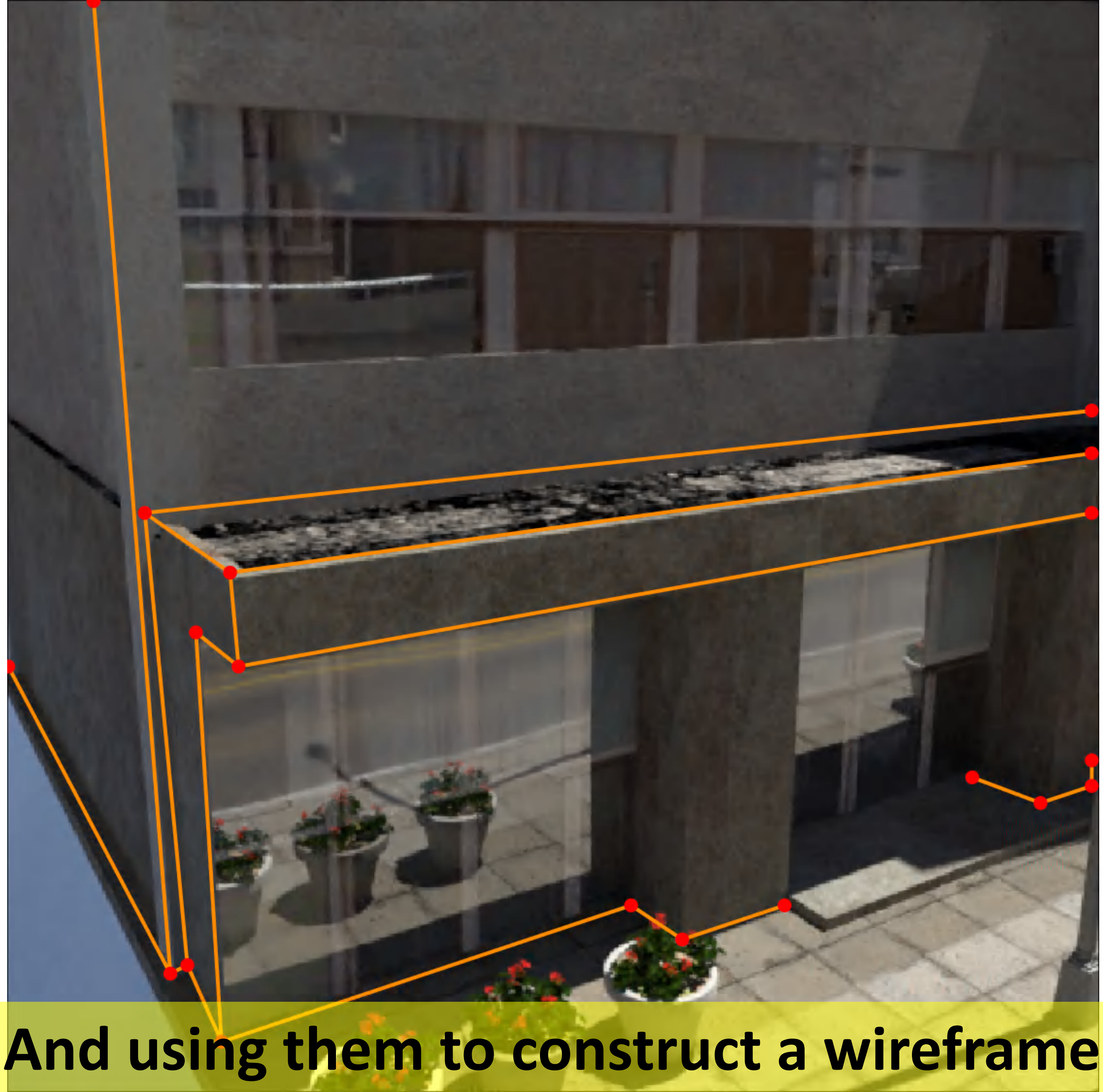
3D Wireframe Detection – Overall Pipeline



From a single RGB image

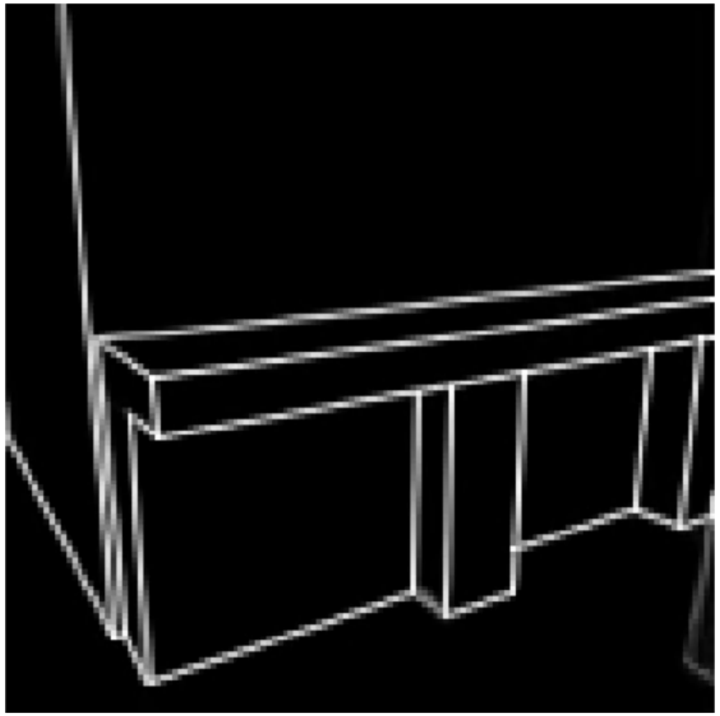
C-Junction Heat Map

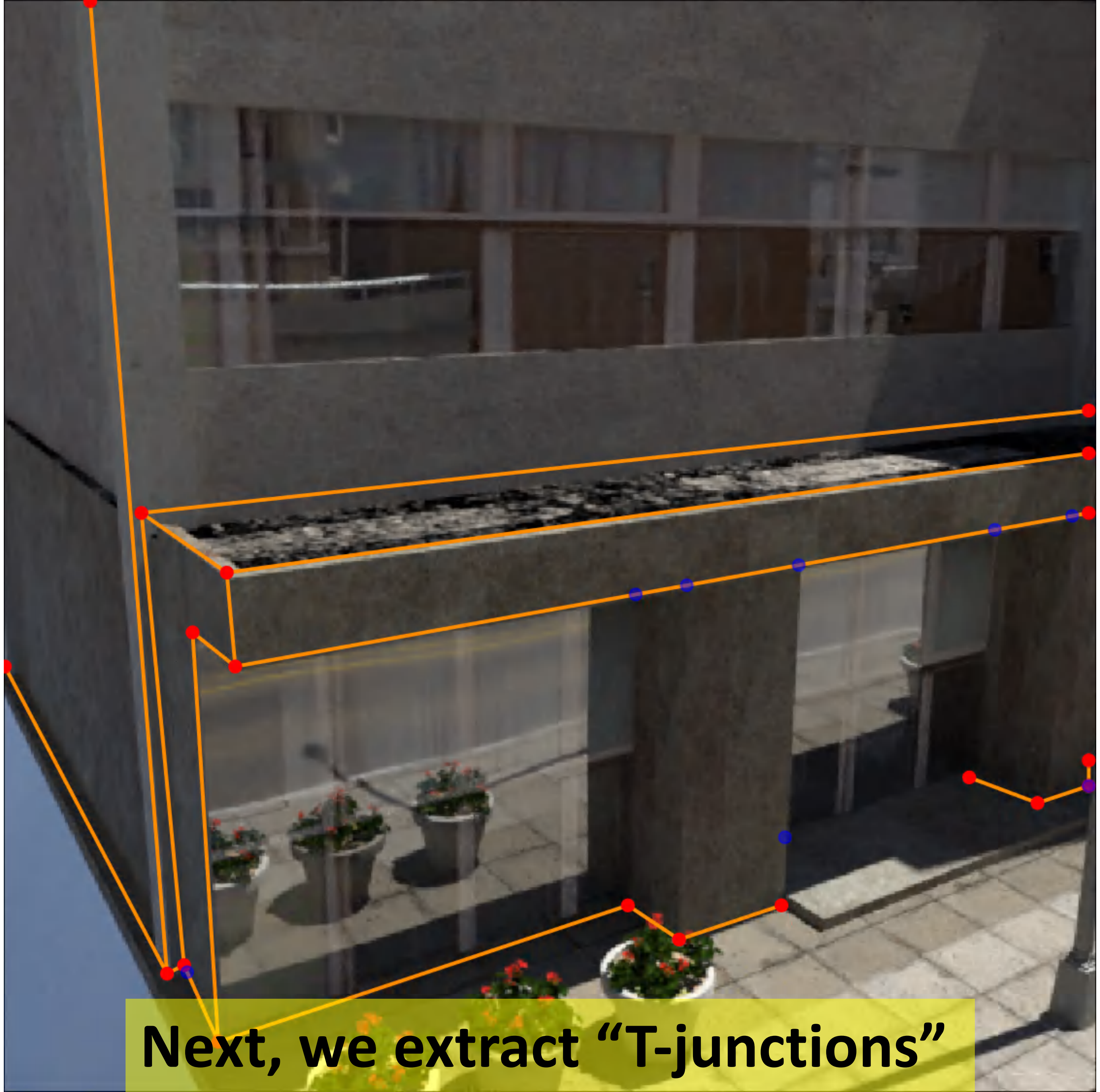




And using them to construct a wireframe

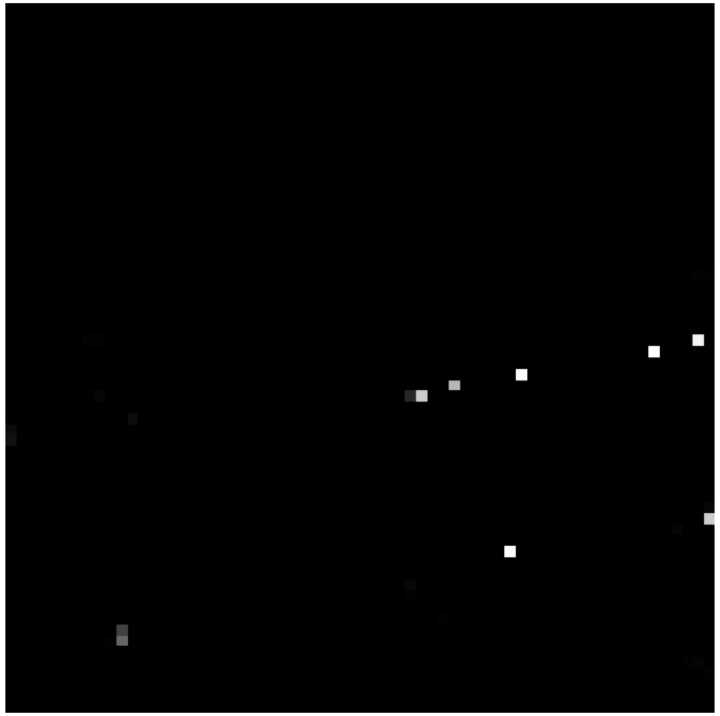
Line Heat Map

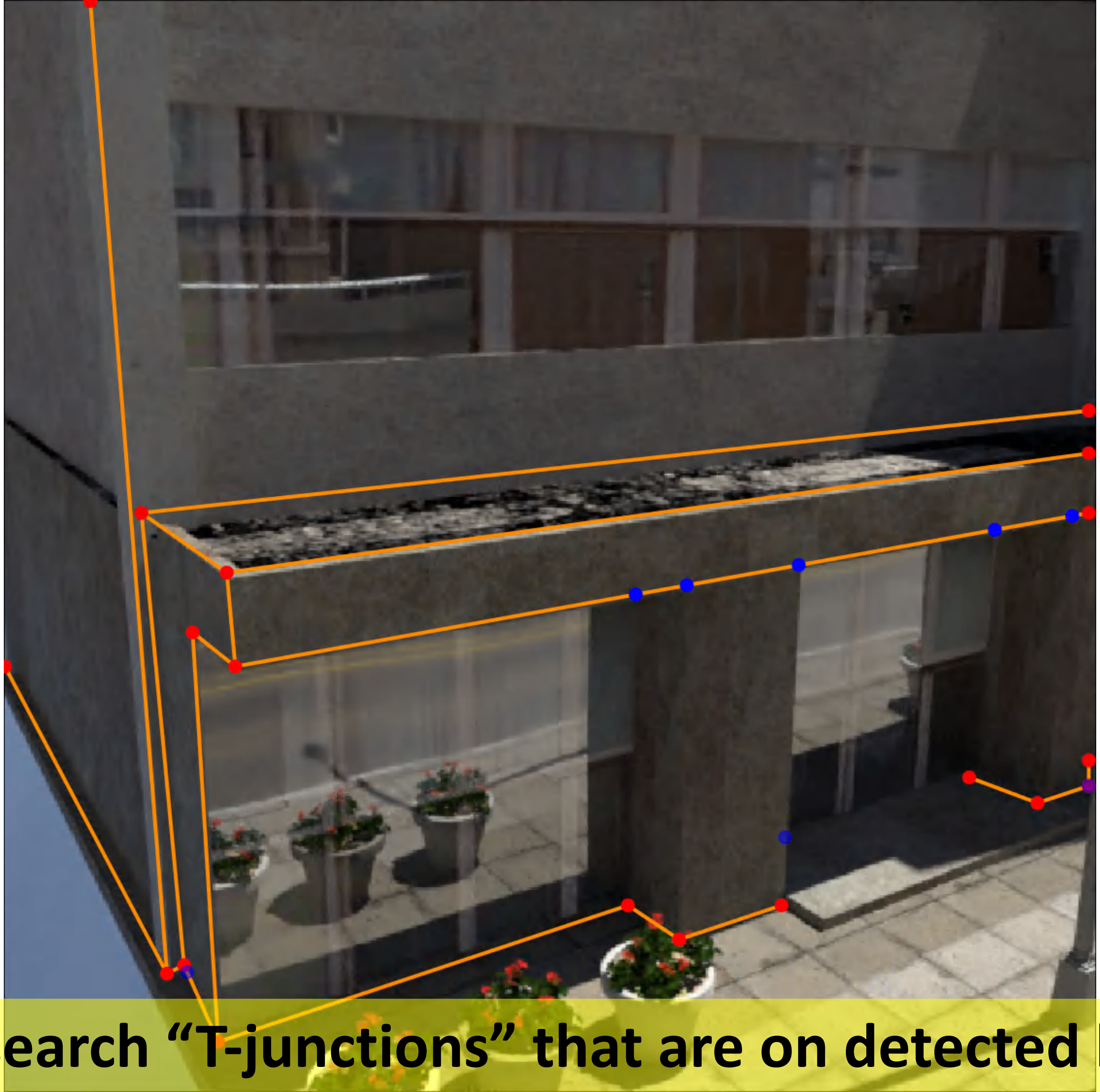




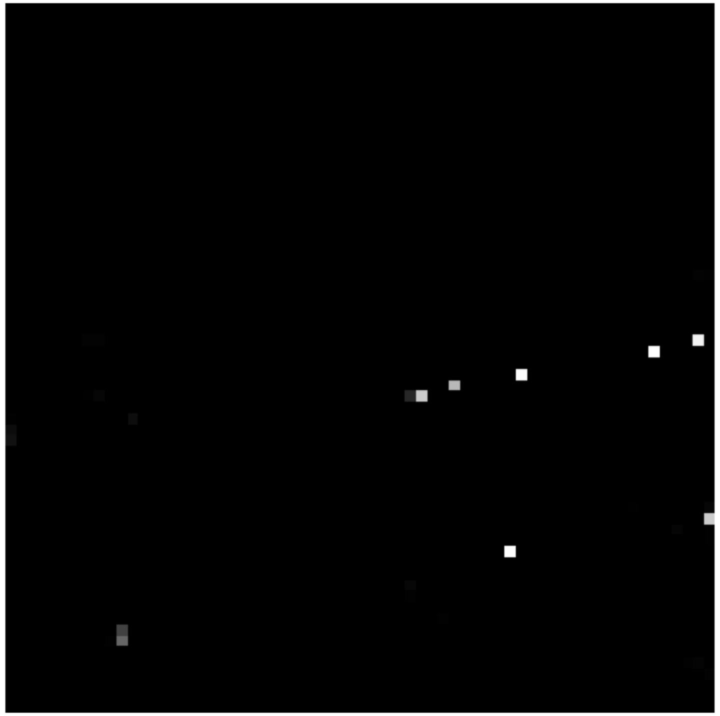
Next, we extract “T-junctions”

T-Junction Heat Map



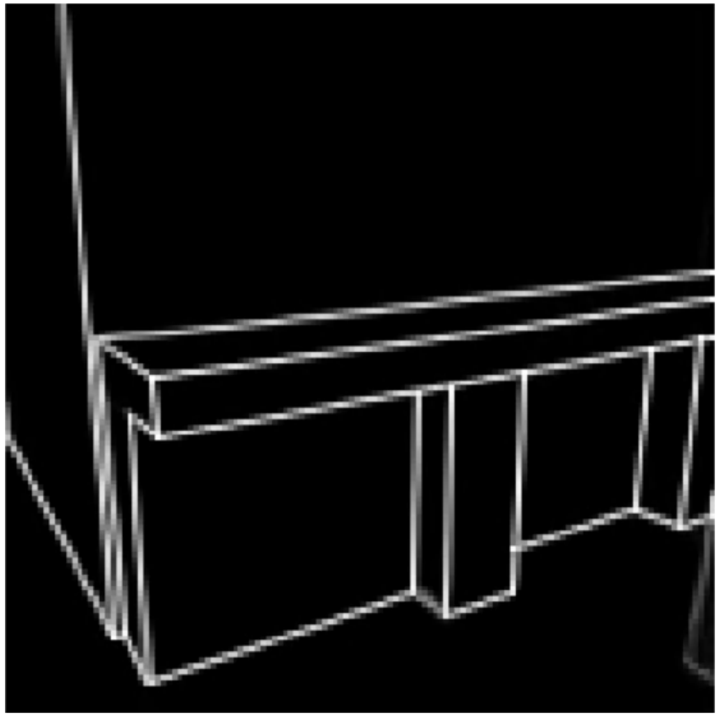


T-Junction Heat Map

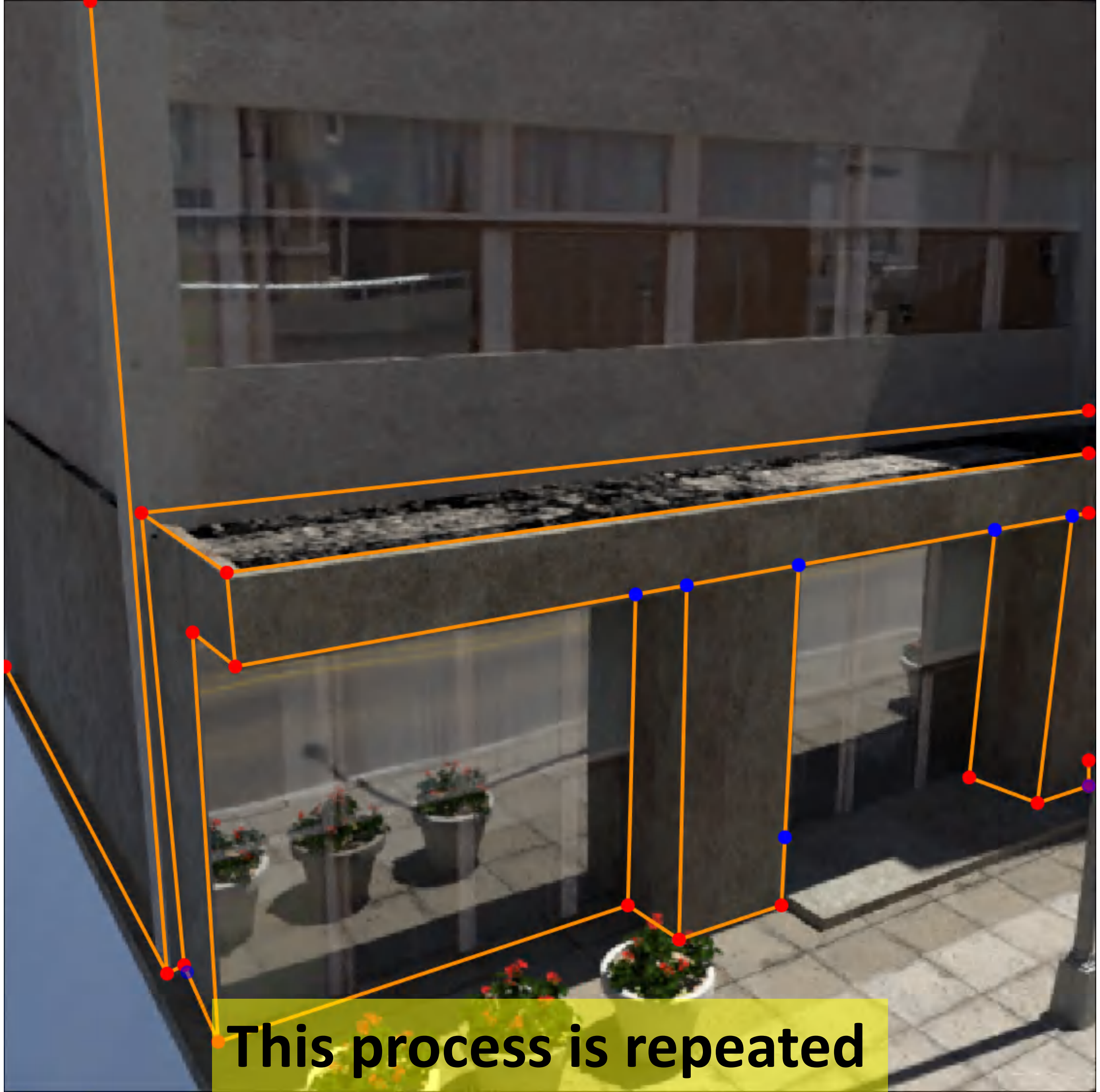


We search "T-junctions" that are on detected lines

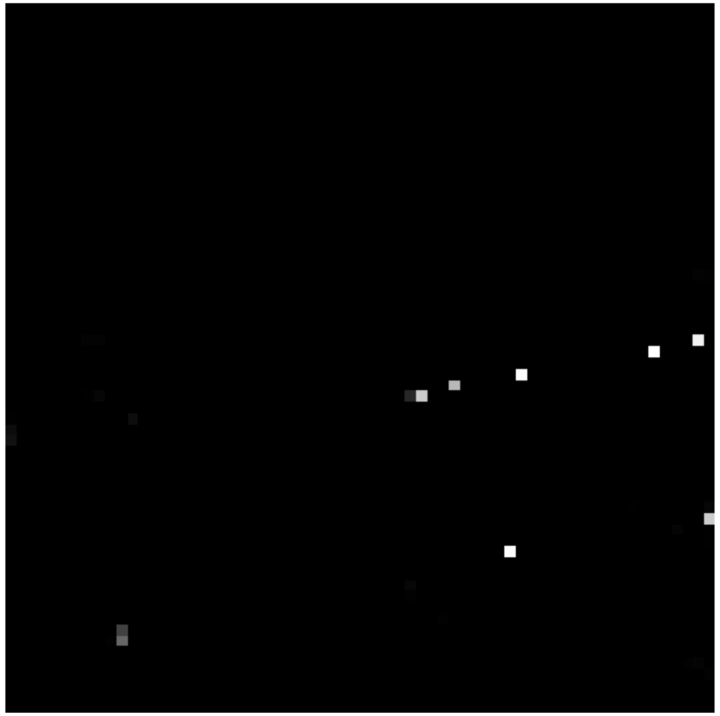
Line Heat Map



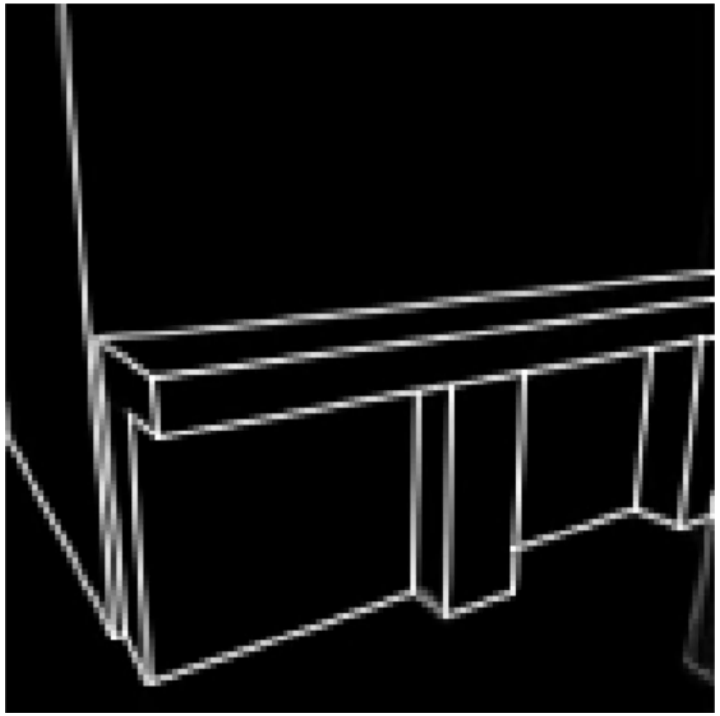
and attach them to the current wireframe



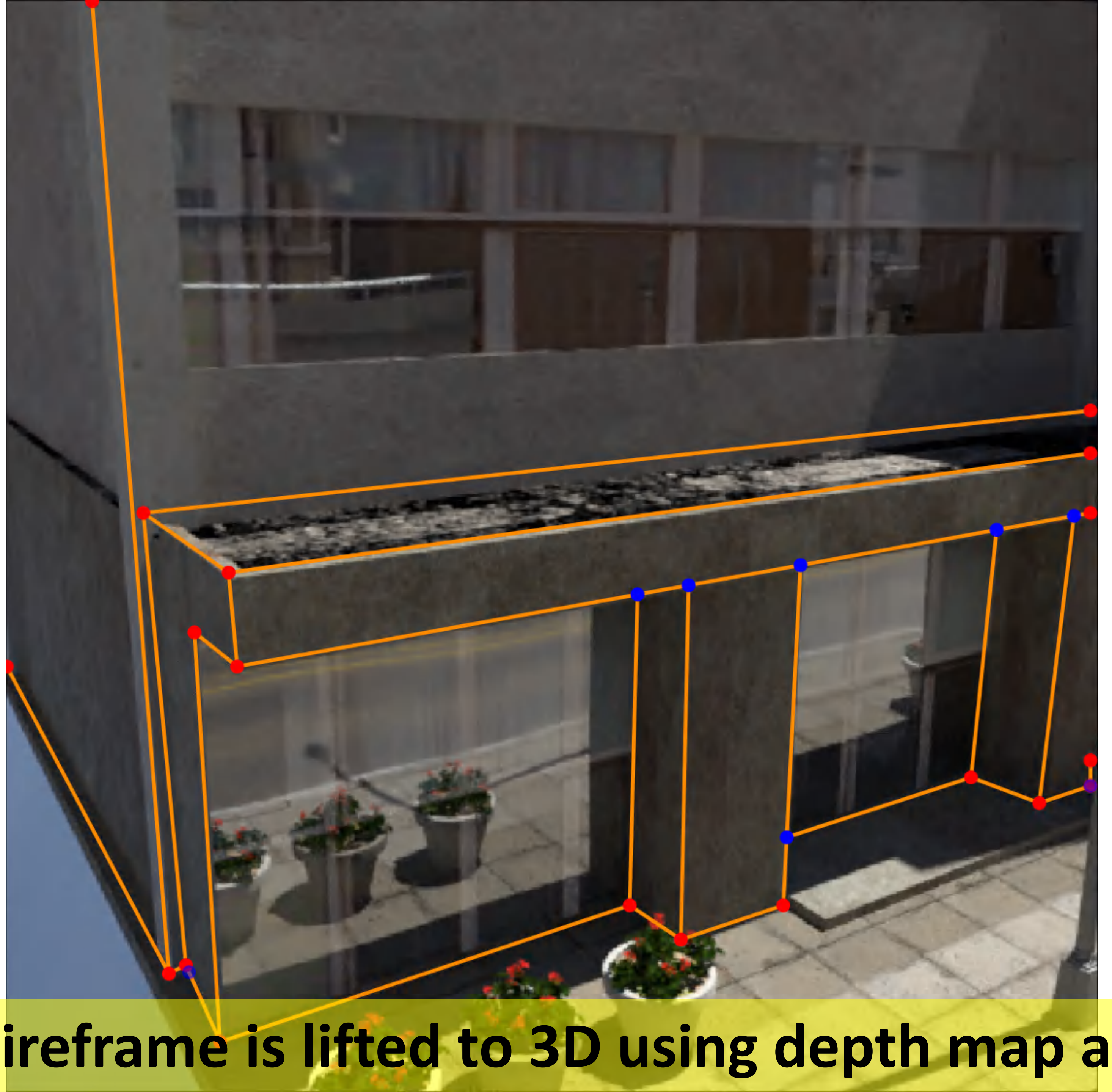
T-Junction Heat Map



Line Heat Map



until the whole wireframe is reconstructed

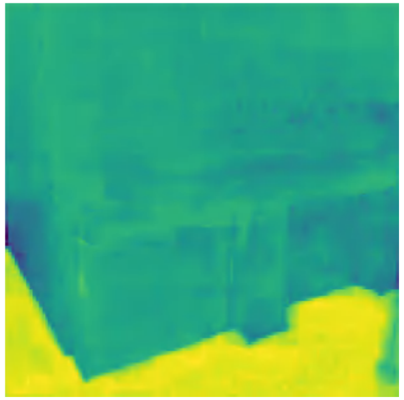


Vanishing Points

$$\begin{pmatrix} V_{1,1} & V_{1,2} & V_{1,3} \\ V_{2,1} & V_{2,2} & V_{2,3} \\ V_{3,1} & V_{3,2} & V_{3,3} \end{pmatrix}$$

+

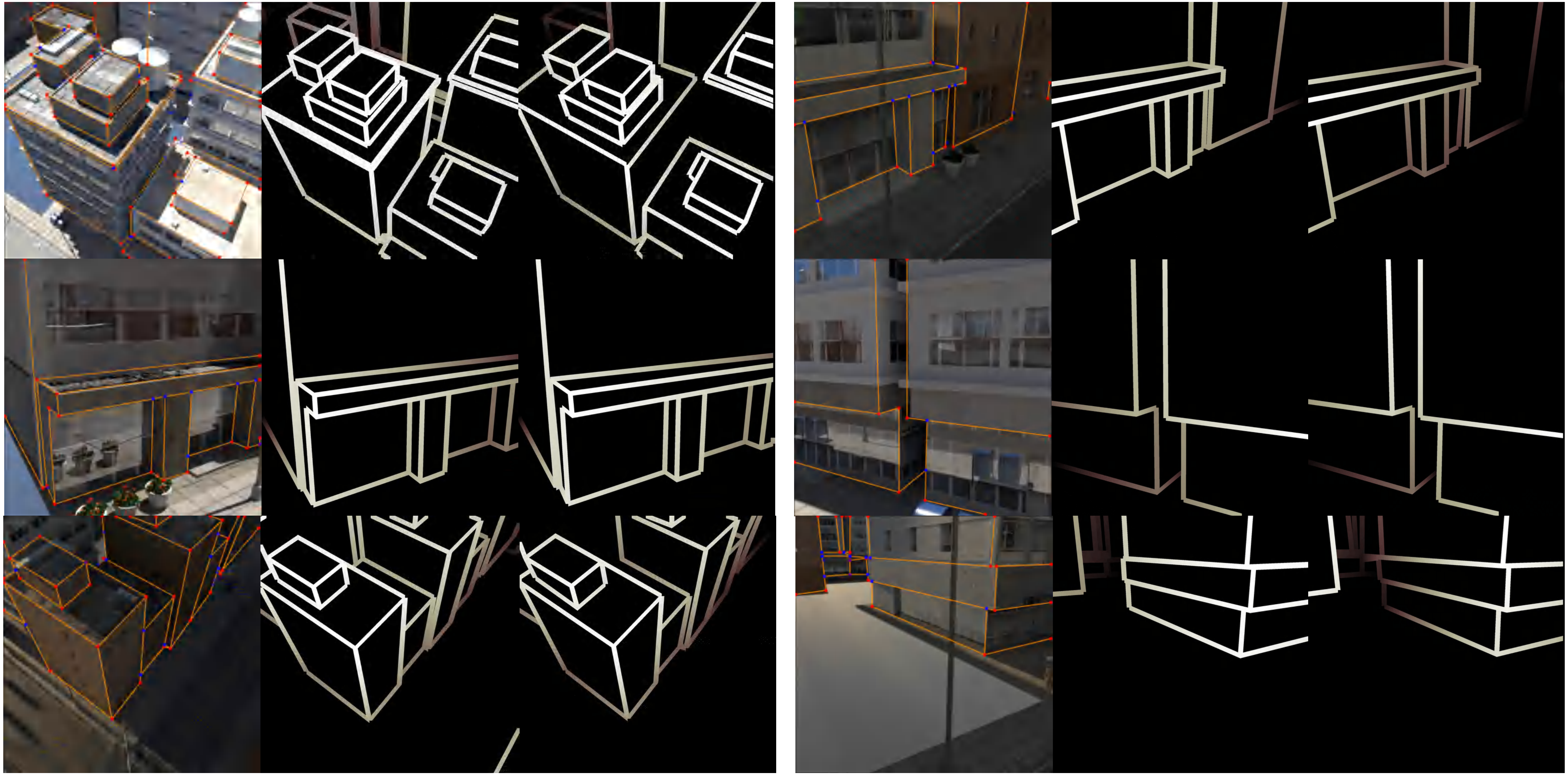
Junction Depth Maps



The wireframe is lifted to 3D using depth map and VPs

Original view (Synthetic)

Rotating to visualize the 3D wireframe



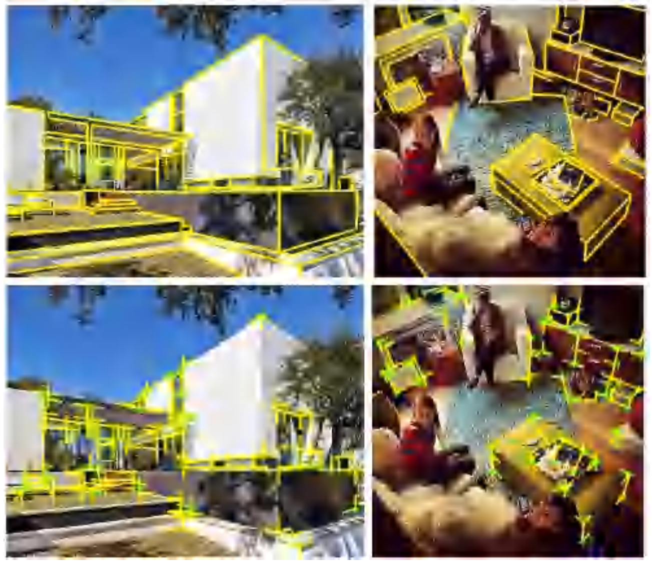
2D Wireframe Ground Truth 3D Inferred 3D 2D Wireframe Ground Truth 3D Inferred 3D

Summary

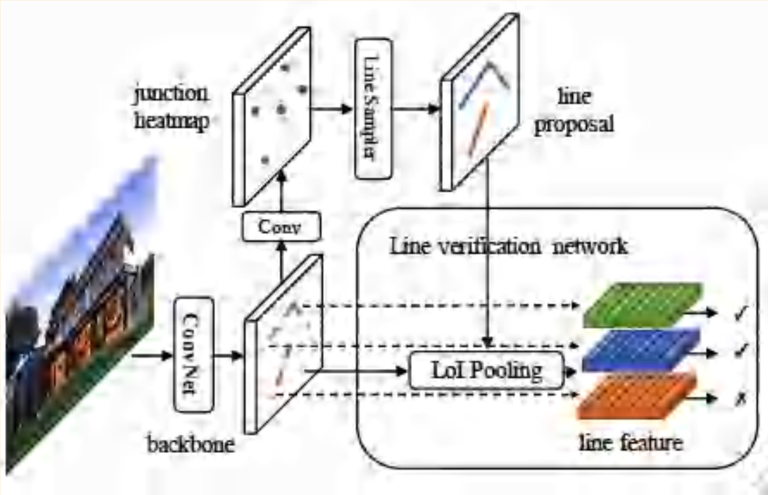
Geometric Methods	Learning-based Methods
• Detect structures by grouping local image cues in a bottom-up fashion	• Detect structures by learning from information provided by humans

Learning to Reconstruct 3D CAD Models – Three Pillars

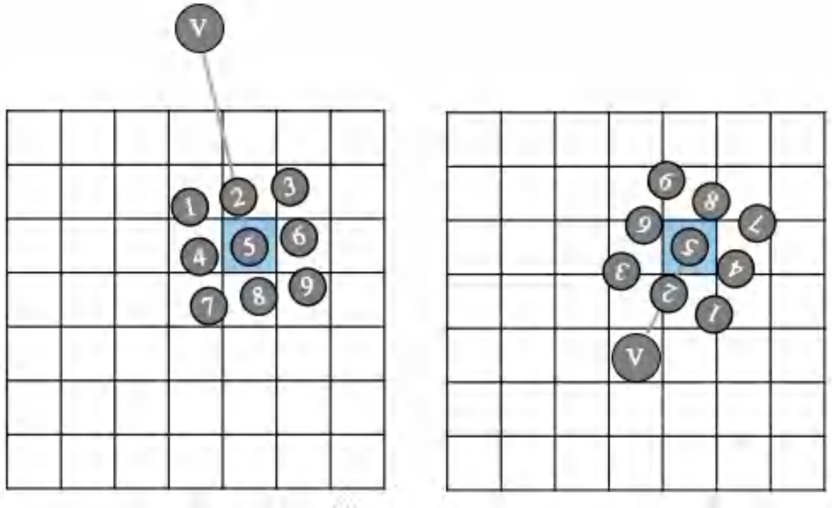
Data



Learning Scheme

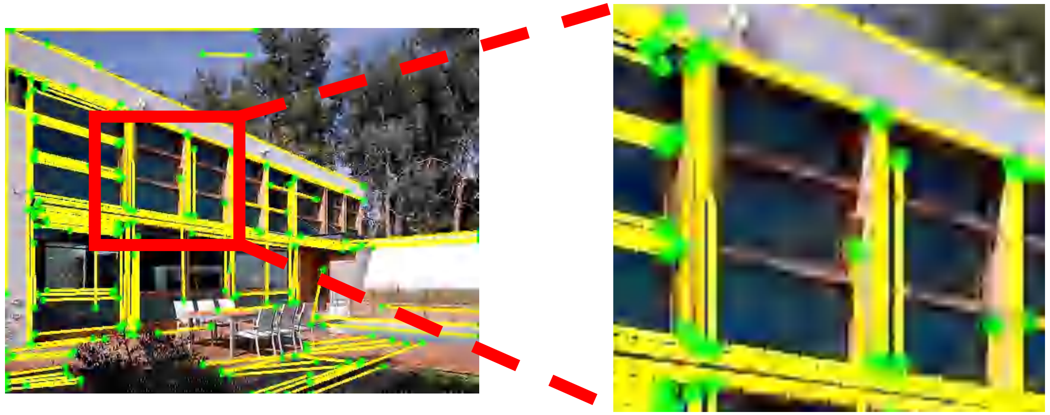


Network Design

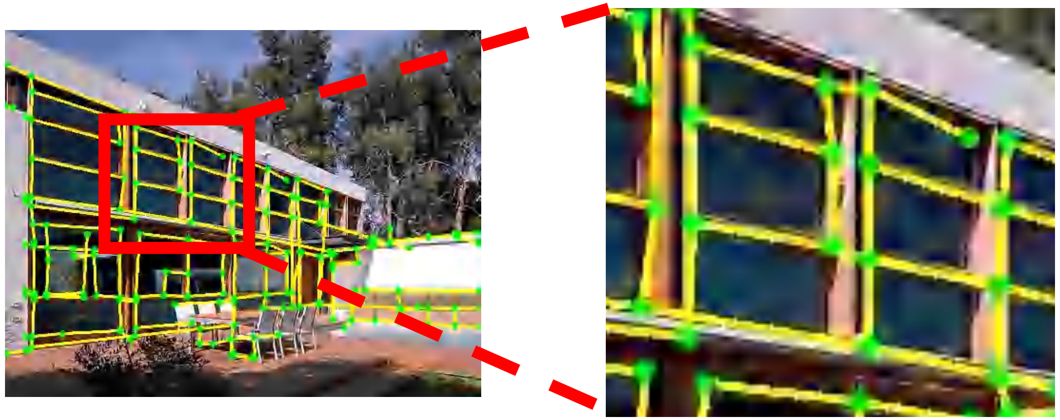


A Close Look at the Previous Results

- Sub-optimal line alignment due to small errors in the predicted junction locations



MCMLSD (Almazan et. al., 2017)



WireframeParser

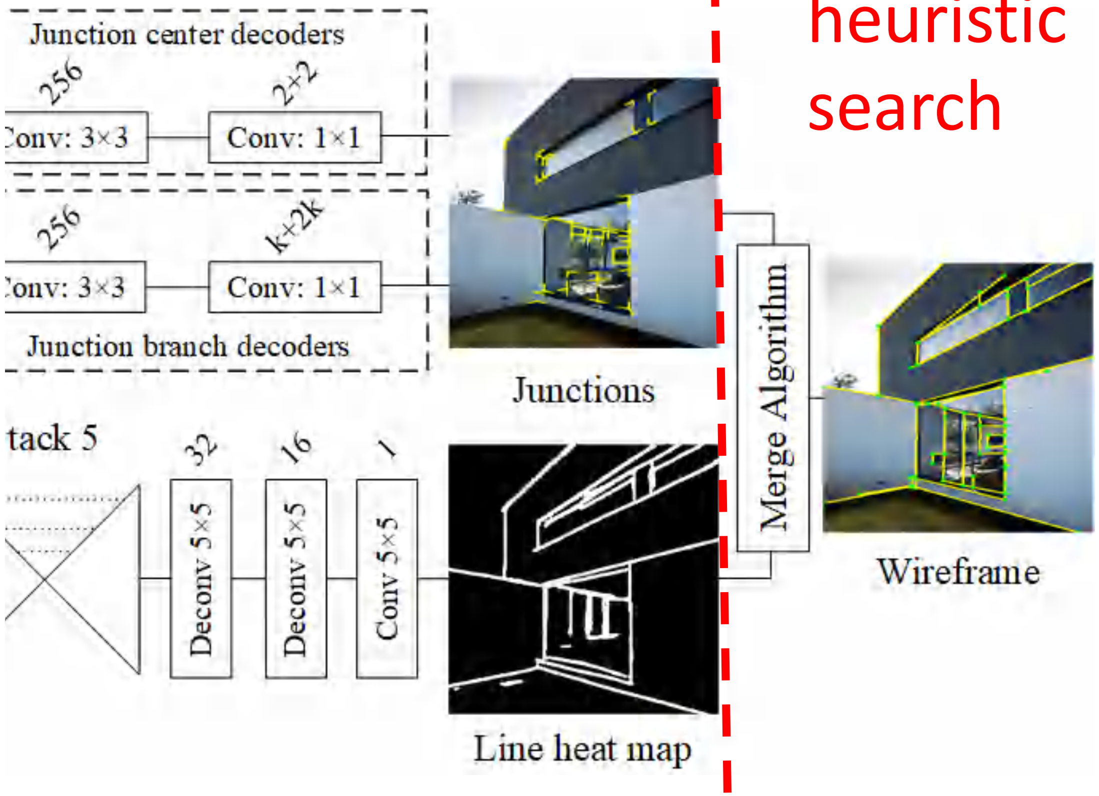
A Close Look at the Previous Results

- Sub-optimal line alignment due to small errors in the predicted junction locations

- Junction locations are not influenced by the predicted line map

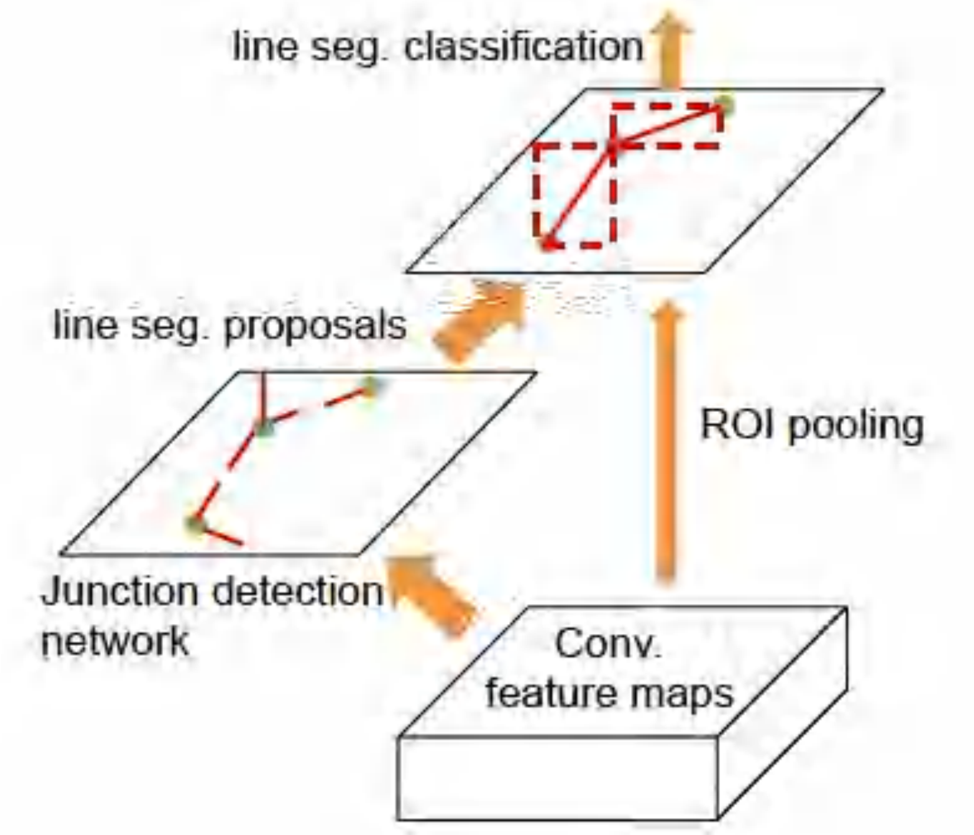
Learned

Not learned, heuristic search

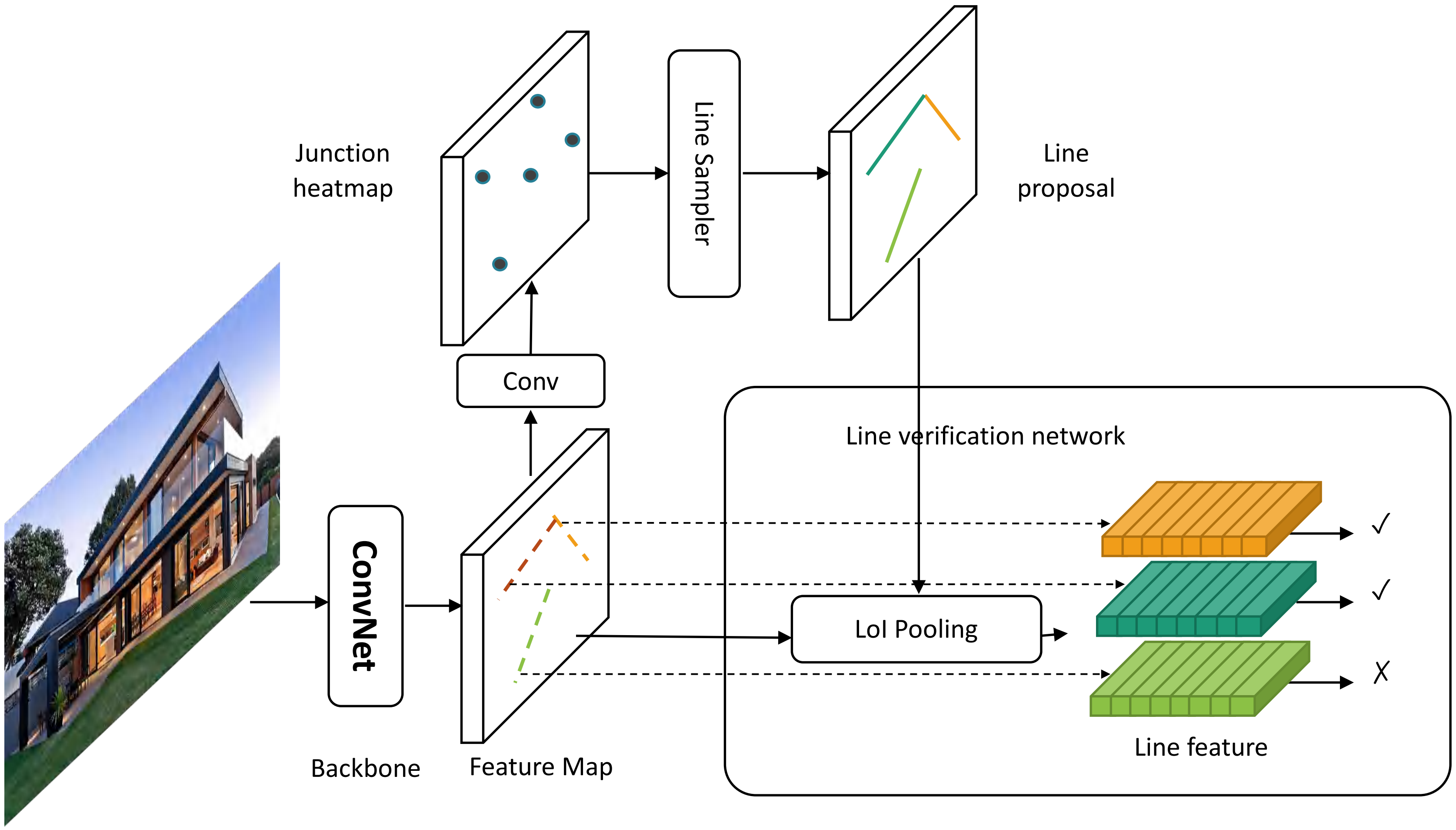


End-to-end Wireframe Parsing

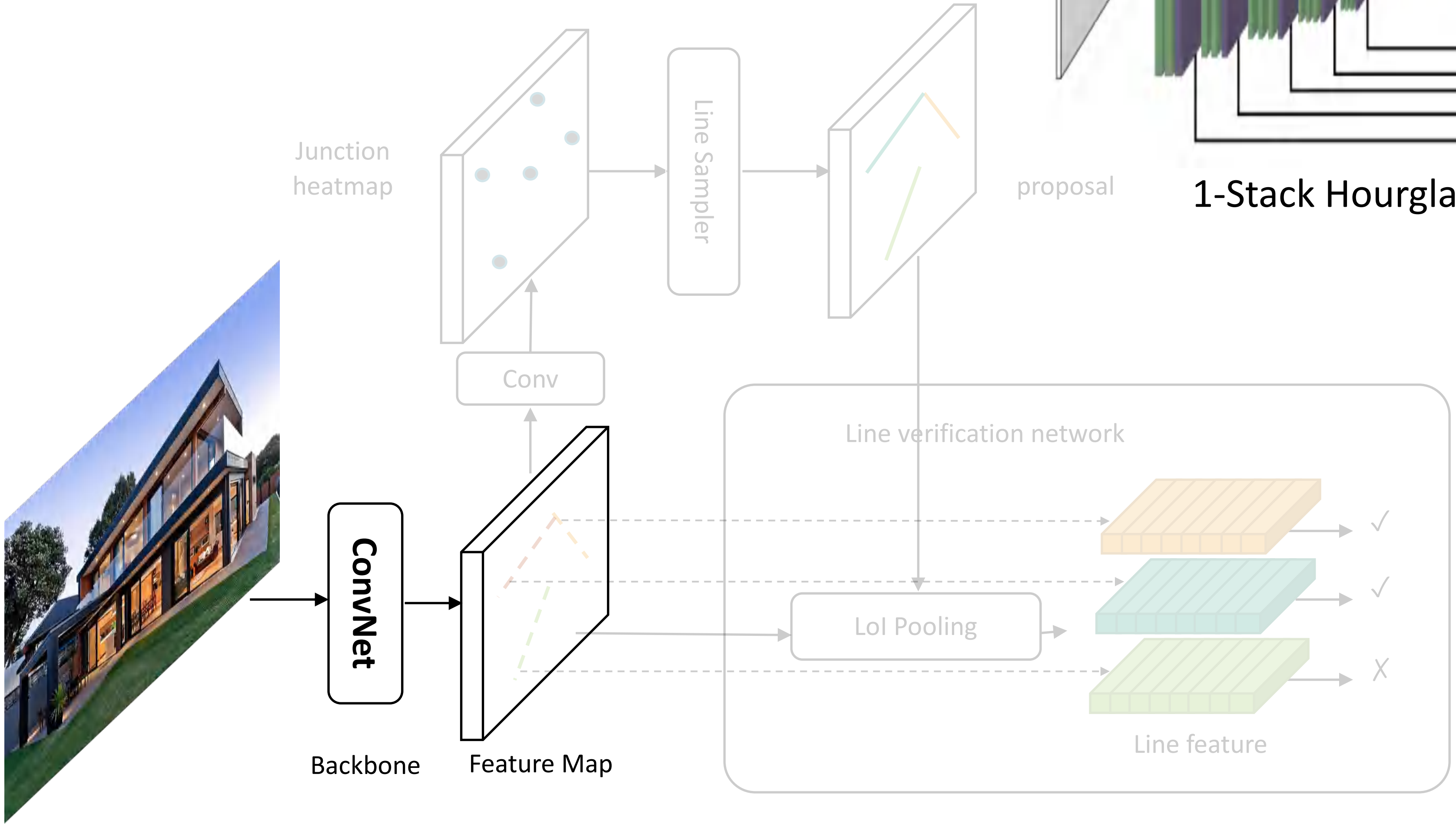
- Treat junction detection network as RPN in Faster R-CNN.
- Directly output vectorized wireframe, including junctions and lines

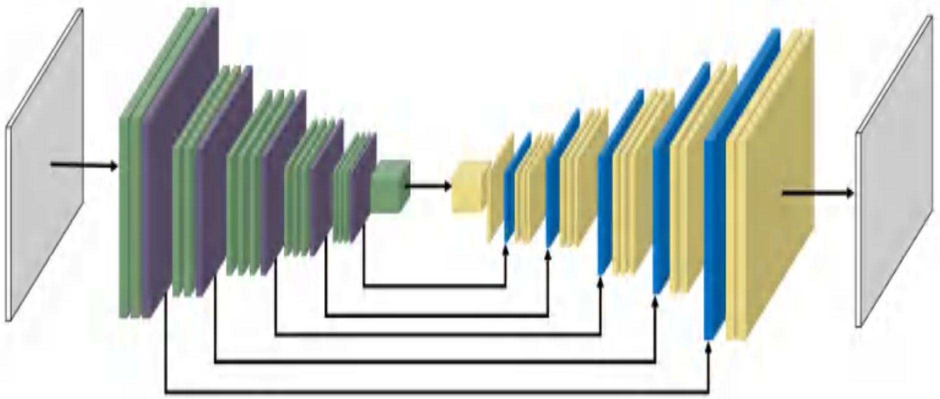


End-to-end Wireframe Parsing



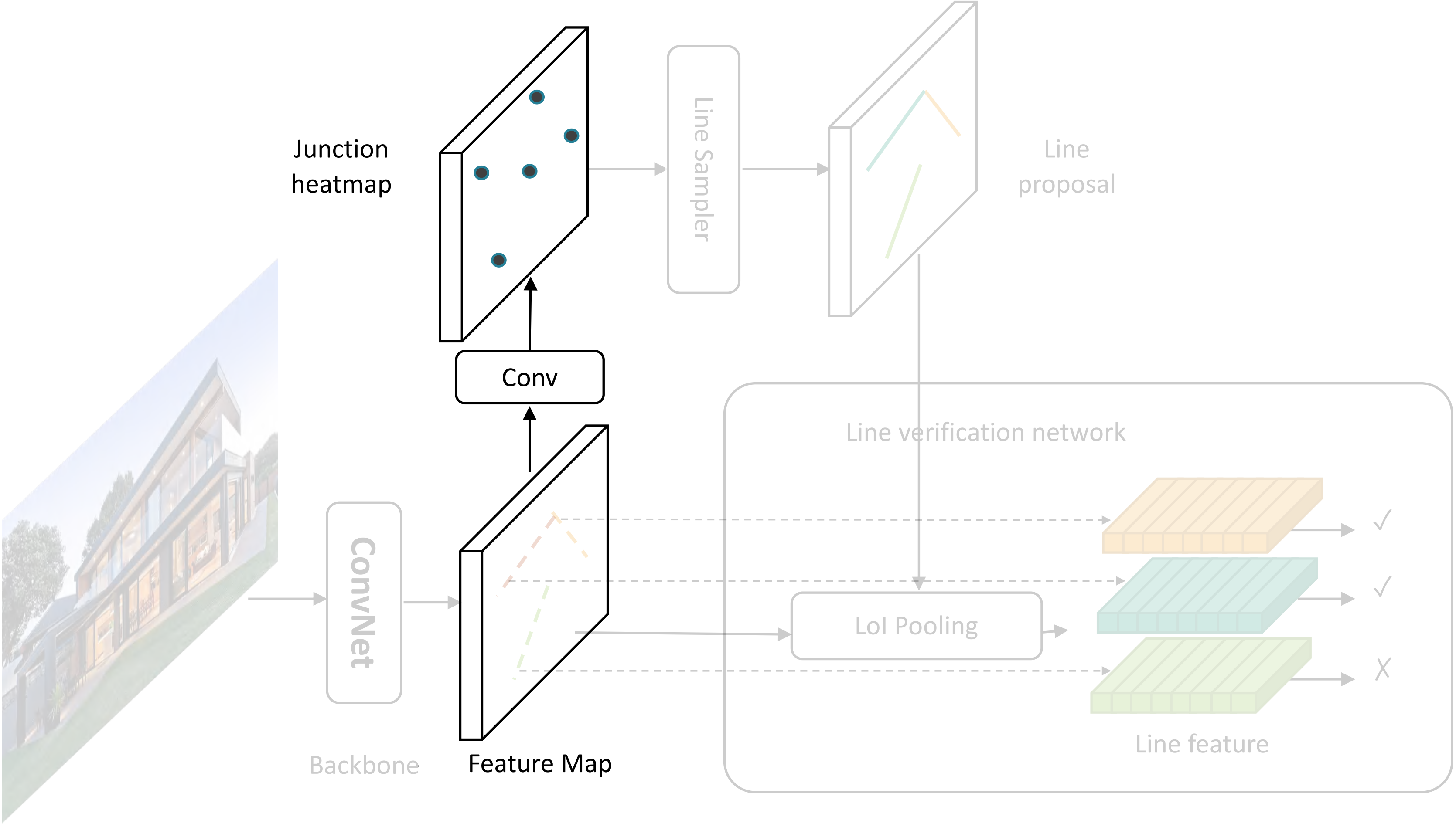
Backbone Network



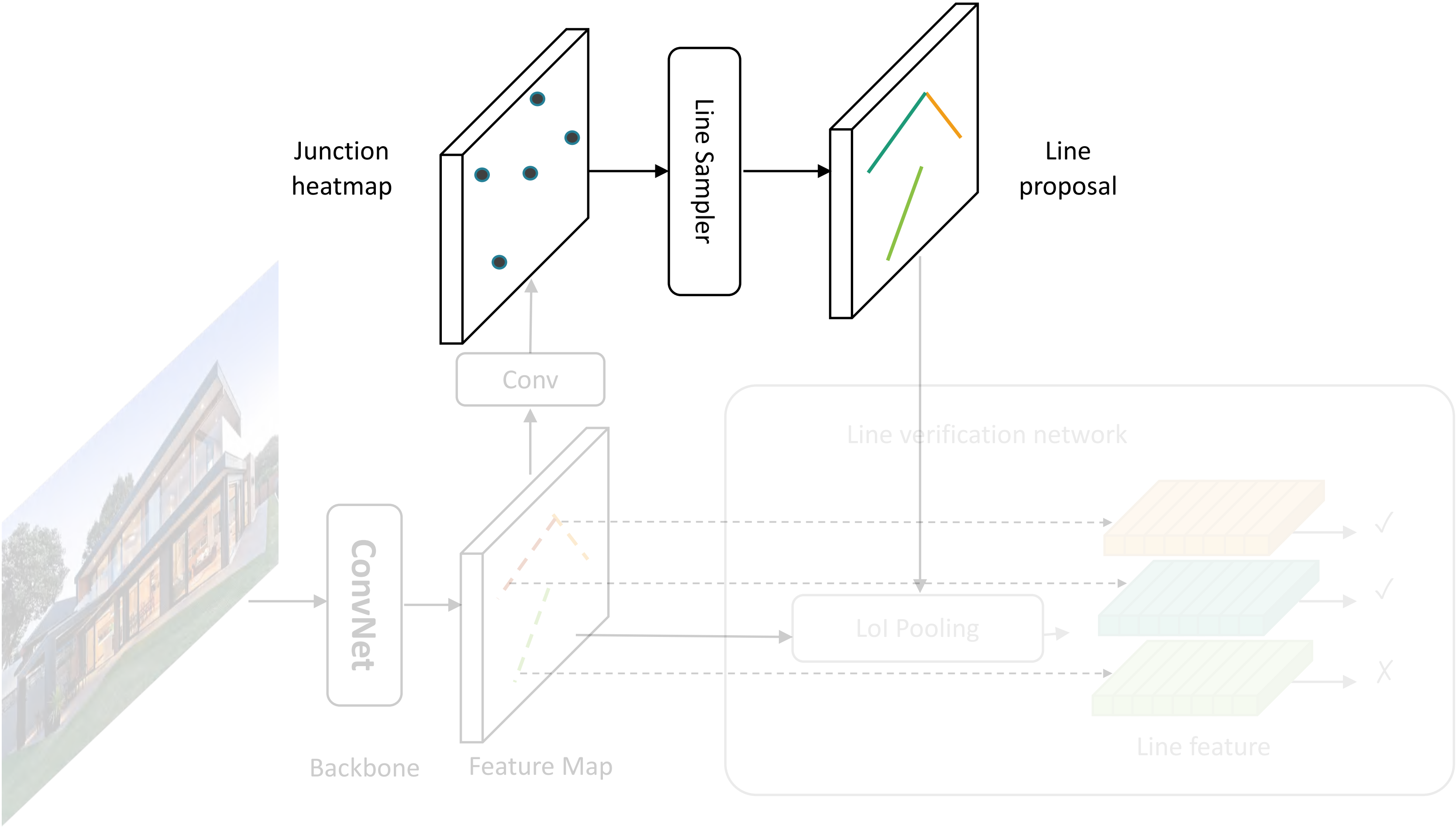


1-Stack Hourglass Network

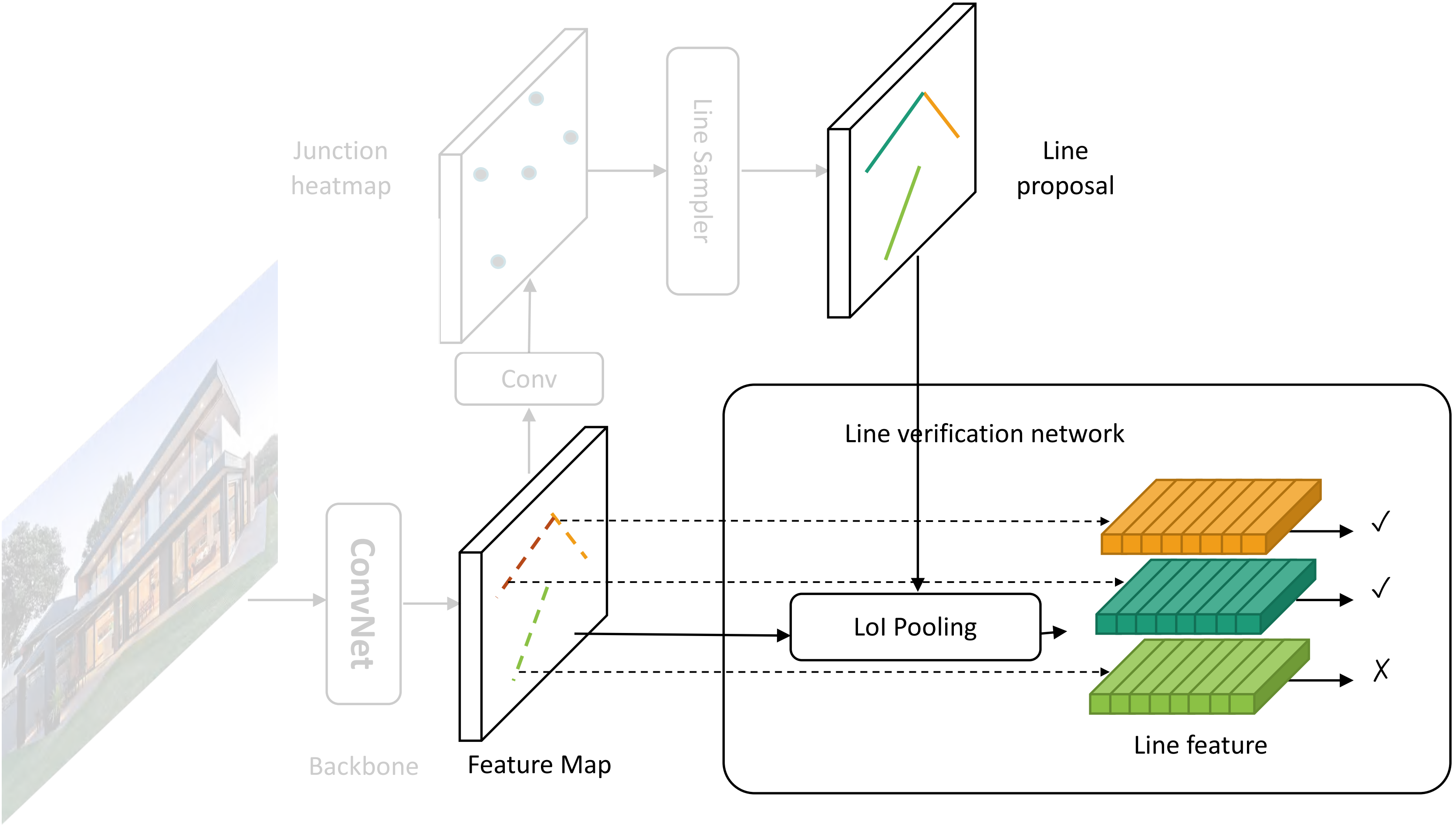
Junction Proposal Network



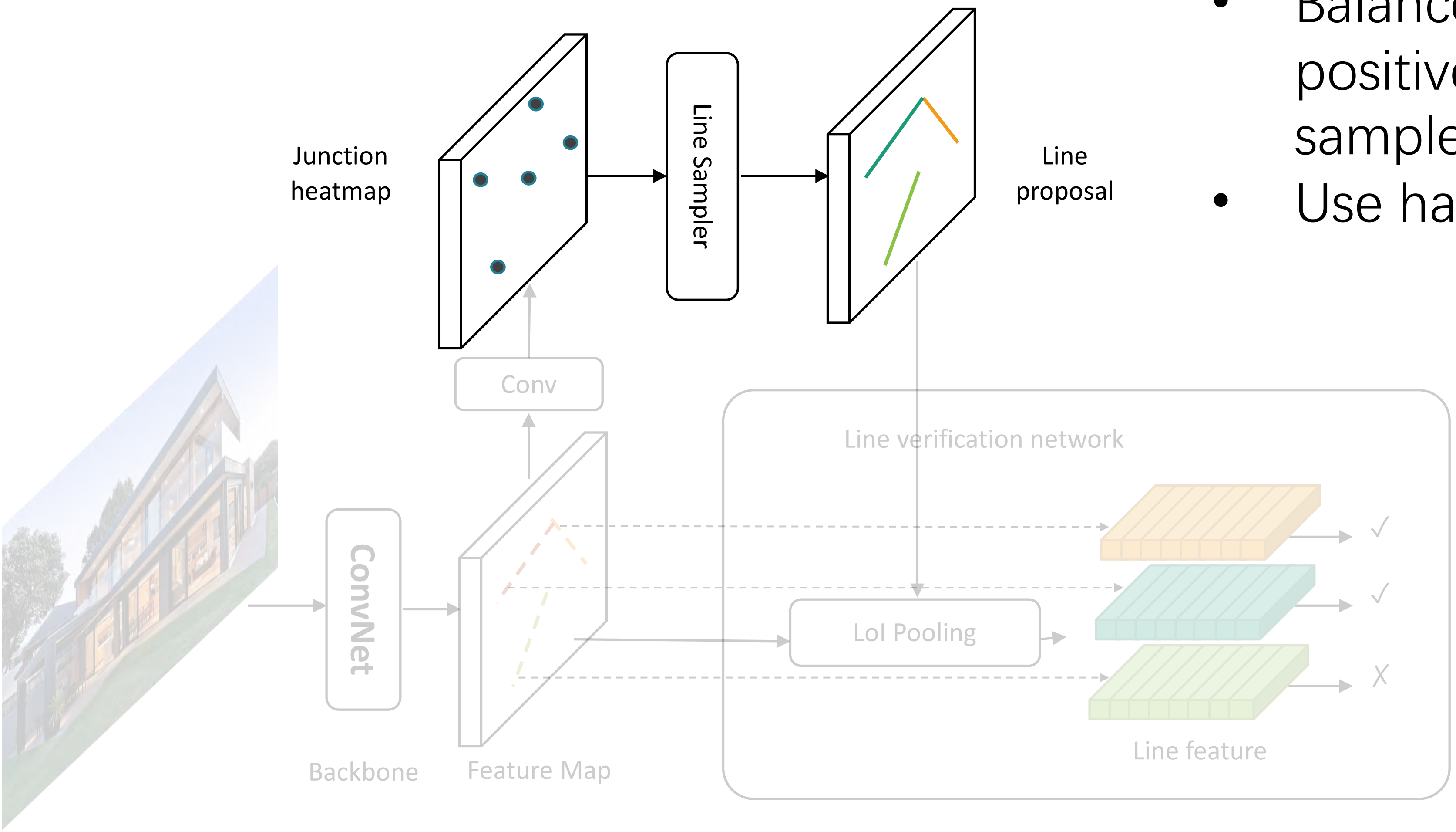
Line Sampler



Line Verification Network



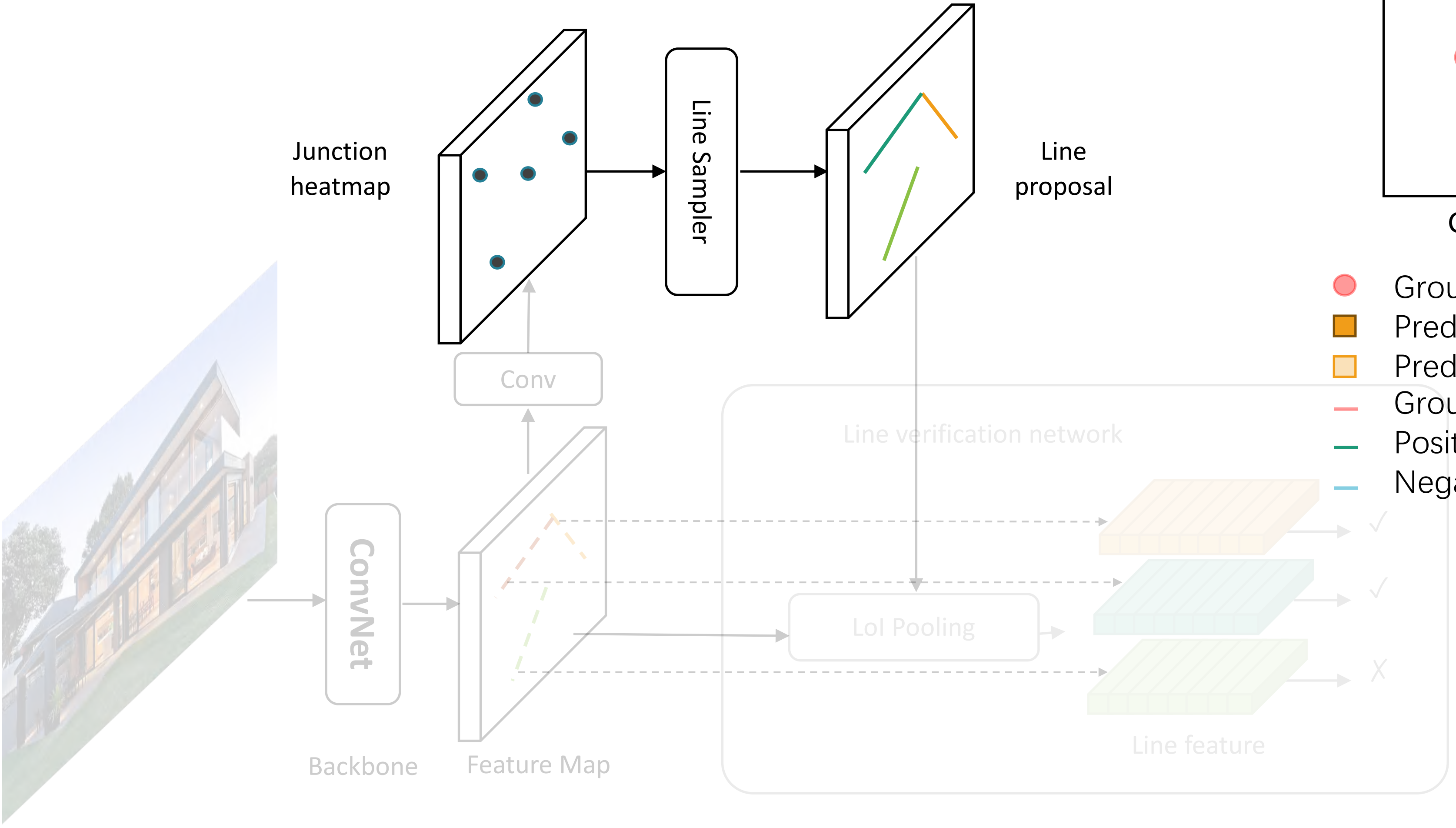
Line Sampler



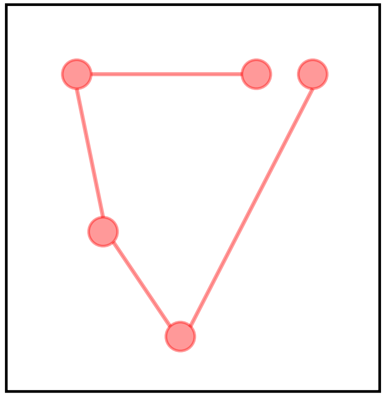
Picking lines for training

- Balance the numbers of positive and negative samples
- Use hard negative lines

Line Sampler (GT Lines)



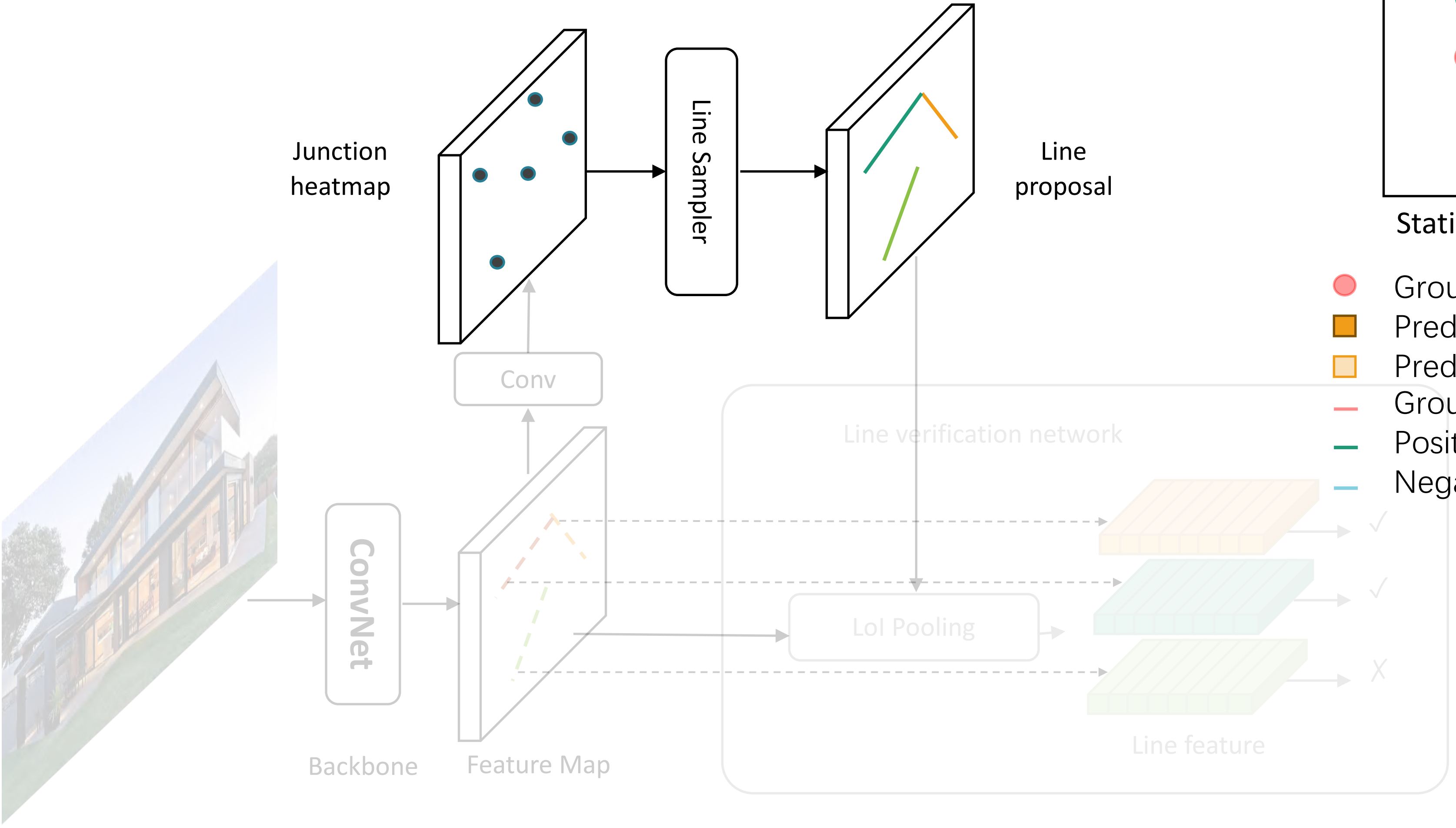
Training:



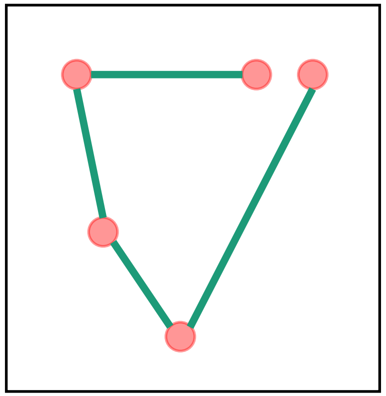
Ground truth

- Ground Truth Junctions
- Predicted Junctions (Matched)
- Predicted Junctions (Un-matched)
- Ground Truth Lines
- Positive Line Samples
- Negative Line Samples

Line Sampler (Static Sampler)



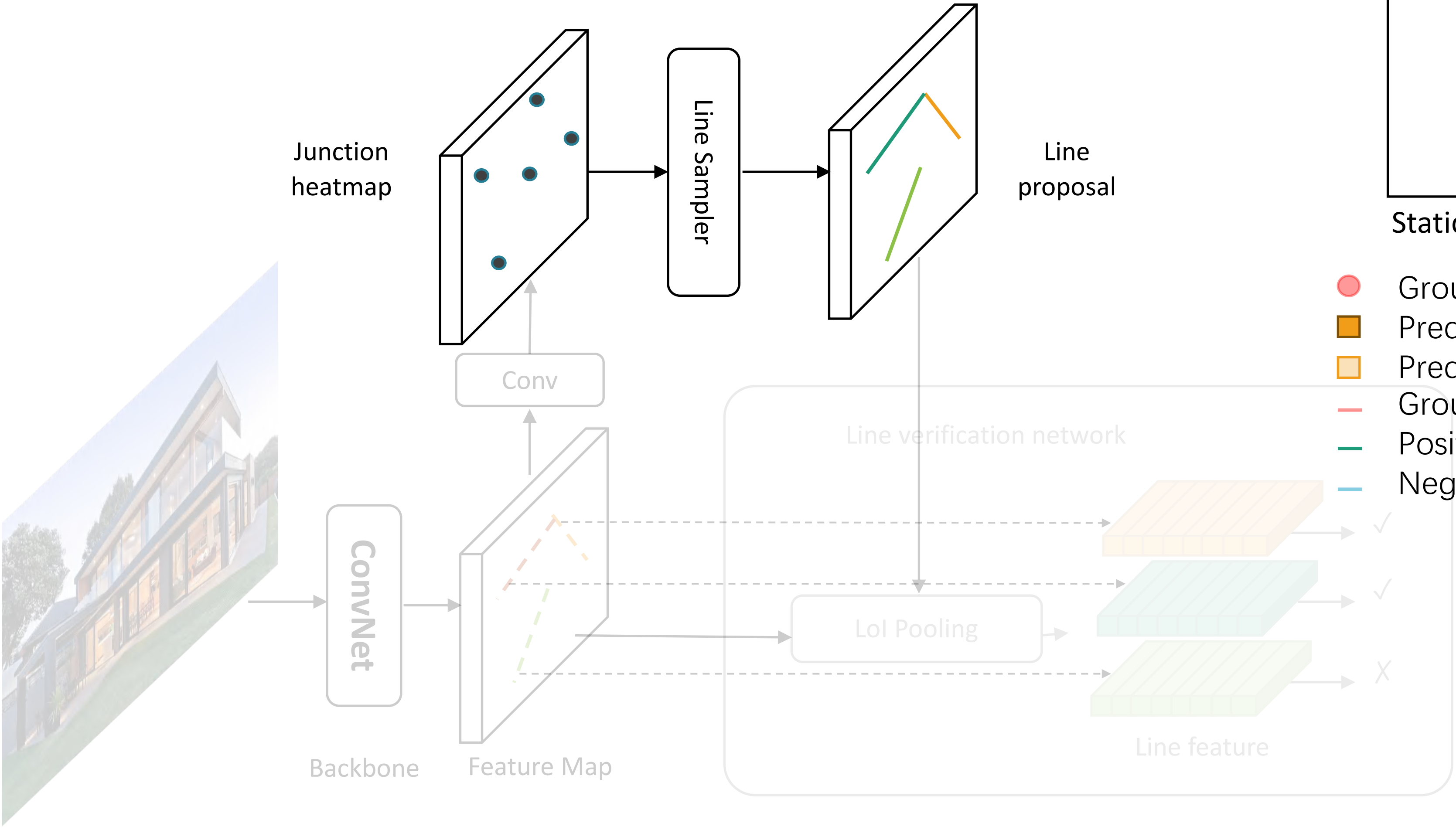
Training:



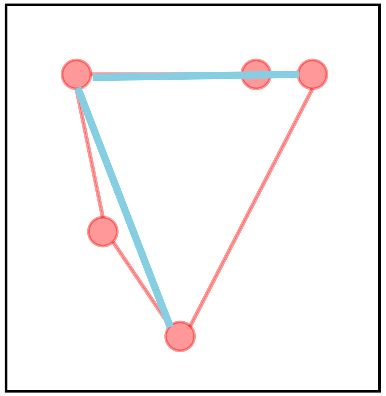
Static positive samples

- Ground Truth Junctions
- Predicted Junctions (Matched)
- Predicted Junctions (Un-matched)
- Ground Truth Lines
- Positive Line Samples
- Negative Line Samples

Line Sampler (Static Sampler)



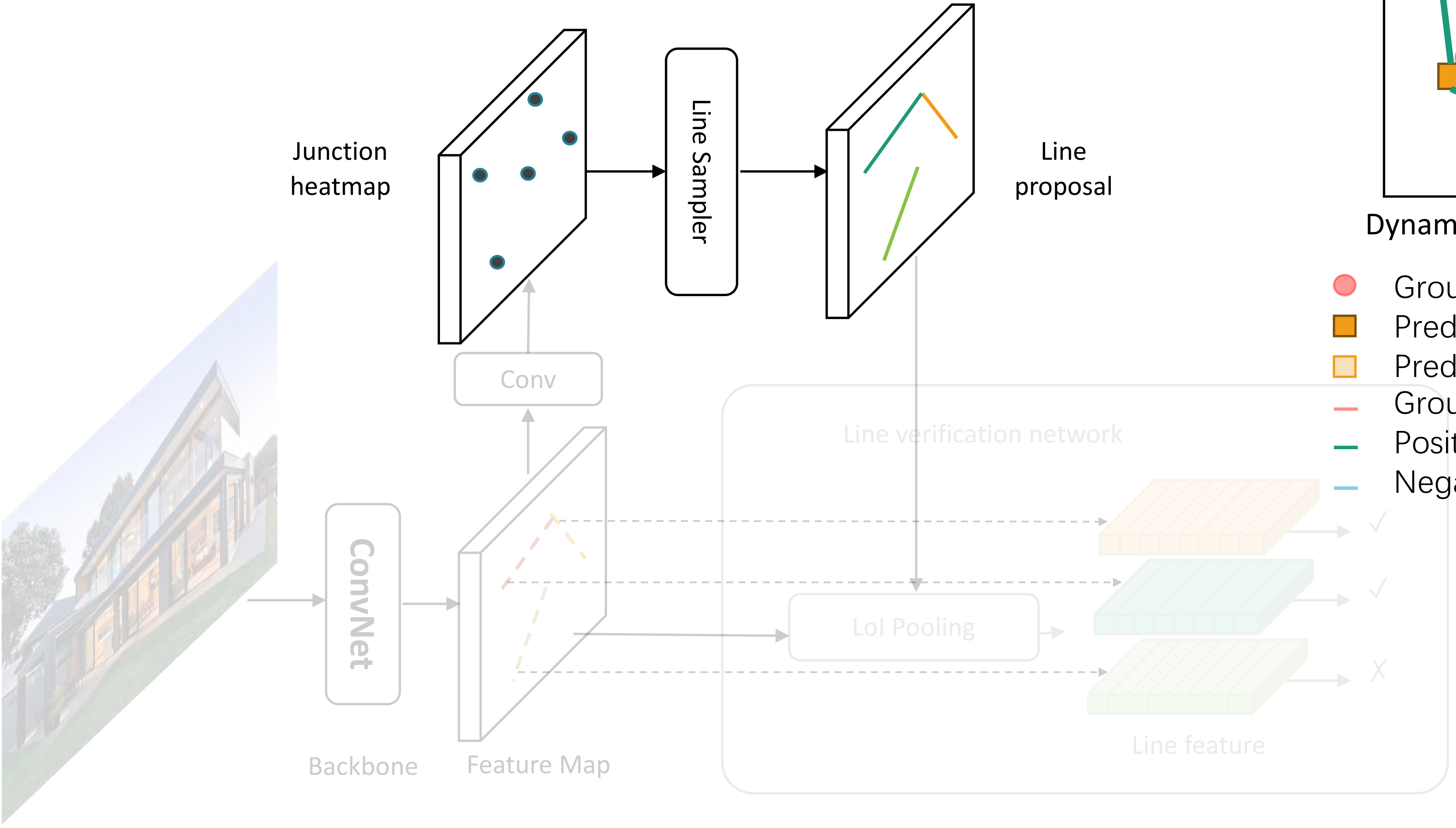
Training:



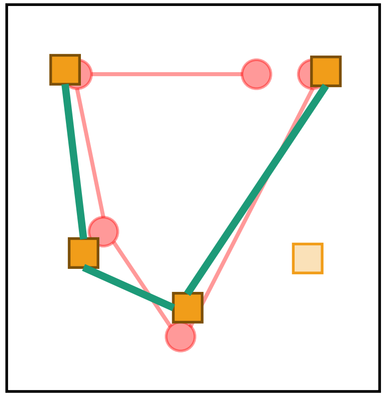
Static negative samples

- Ground Truth Junctions
- Predicted Junctions (Matched)
- Predicted Junctions (Un-matched)
- Ground Truth Lines
- Positive Line Samples
- Negative Line Samples

Line Sampler (Dynamic Sampler)



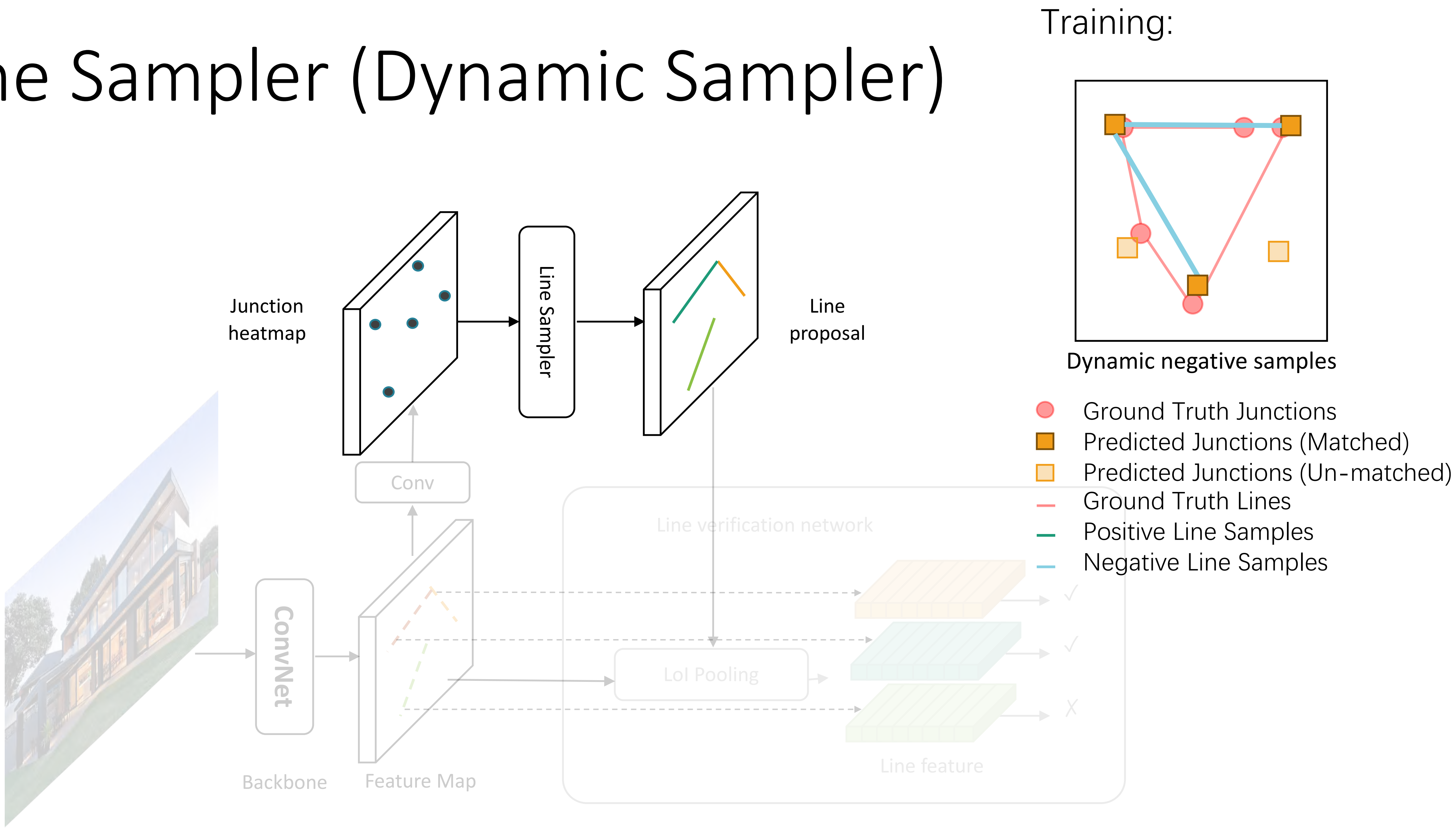
Training:



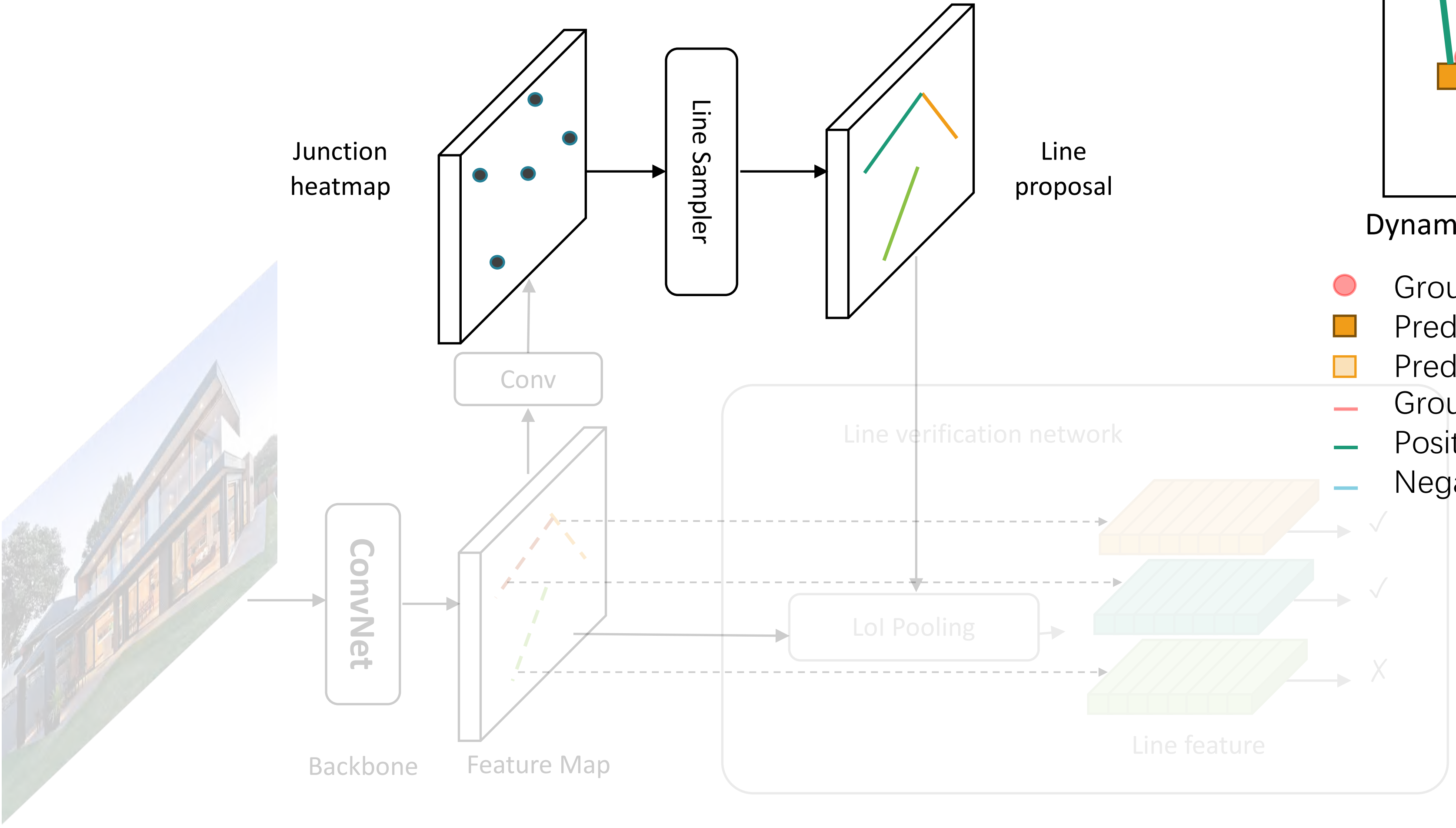
Dynamic positive samples

- Ground Truth Junctions
- Predicted Junctions (Matched)
- Predicted Junctions (Un-matched)
- Ground Truth Lines
- Positive Line Samples
- Negative Line Samples

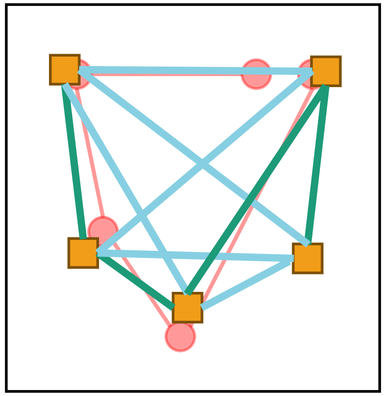
Line Sampler (Dynamic Sampler)



Line Sampler (Dynamic Sampler)



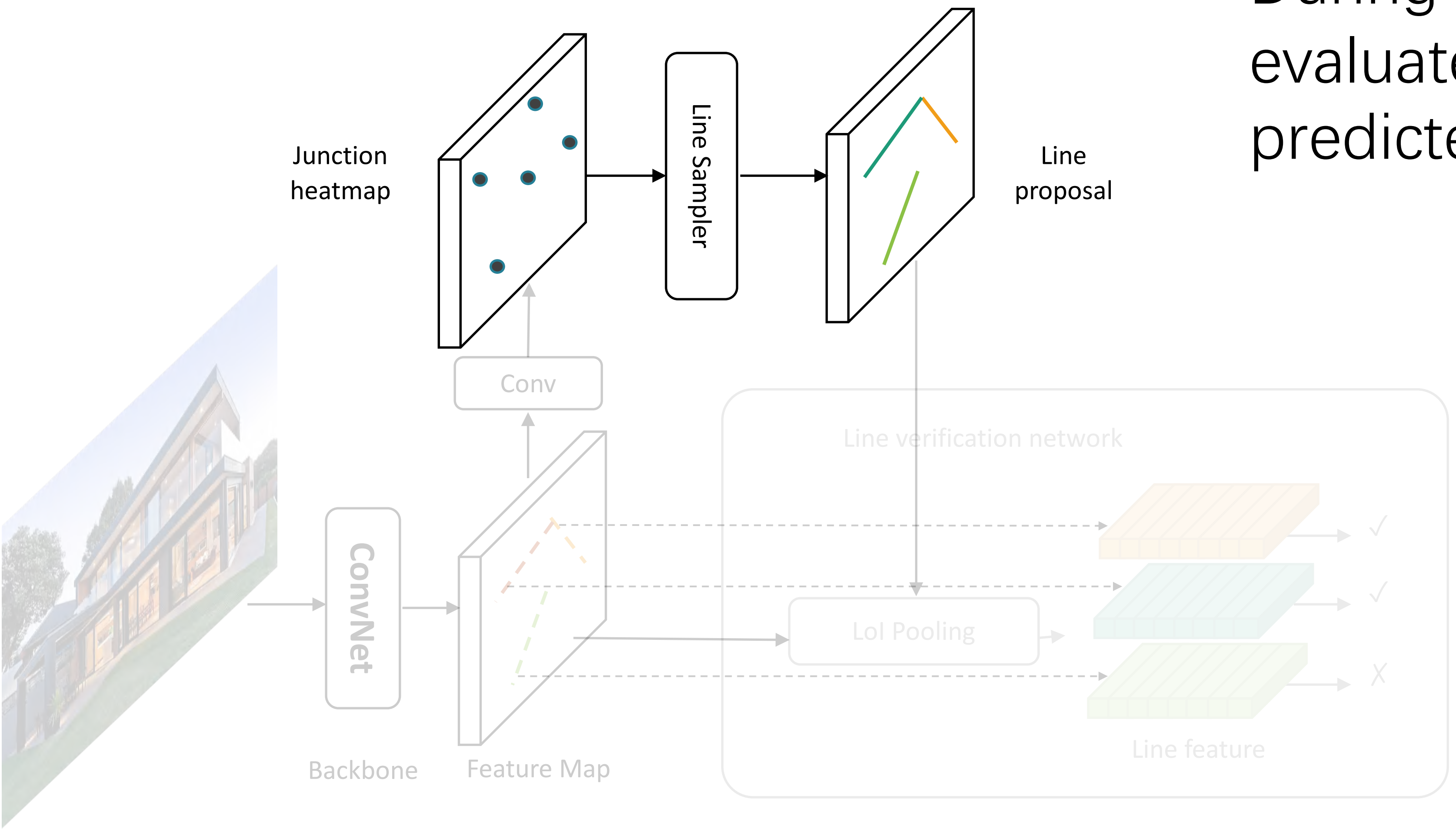
Training:



Dynamic random samples

- Ground Truth Junctions
- Predicted Junctions (Matched)
- Predicted Junctions (Un-matched)
- Ground Truth Lines
- Positive Line Samples
- Negative Line Samples

Line Sampler

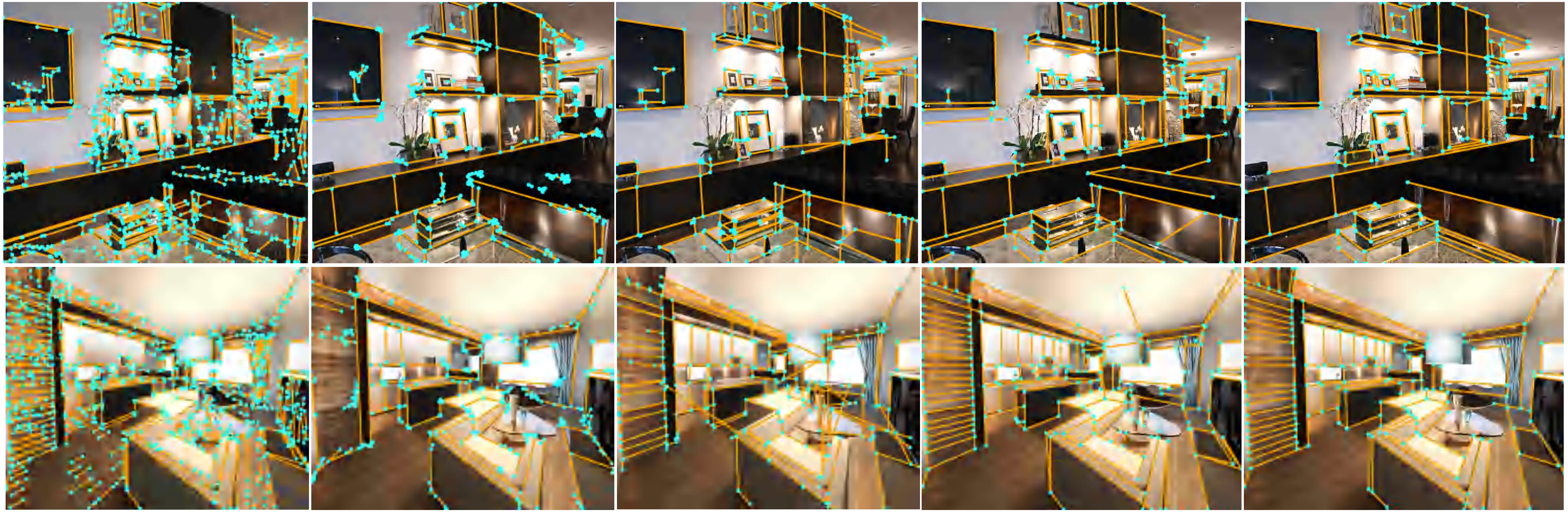


During testing, we evaluate all pairs of predicted junctions

Qualitative Results

LSD | AFM | WireframeParser | **LCNN** | Ground truth

Qualitative Results

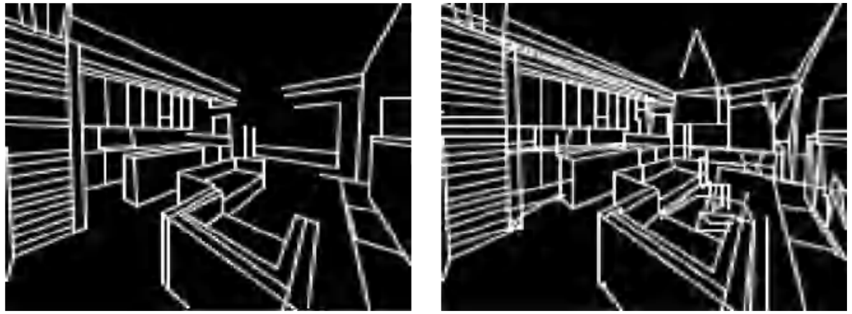


LSD AFM WireframeParser **LCNN** Ground truth

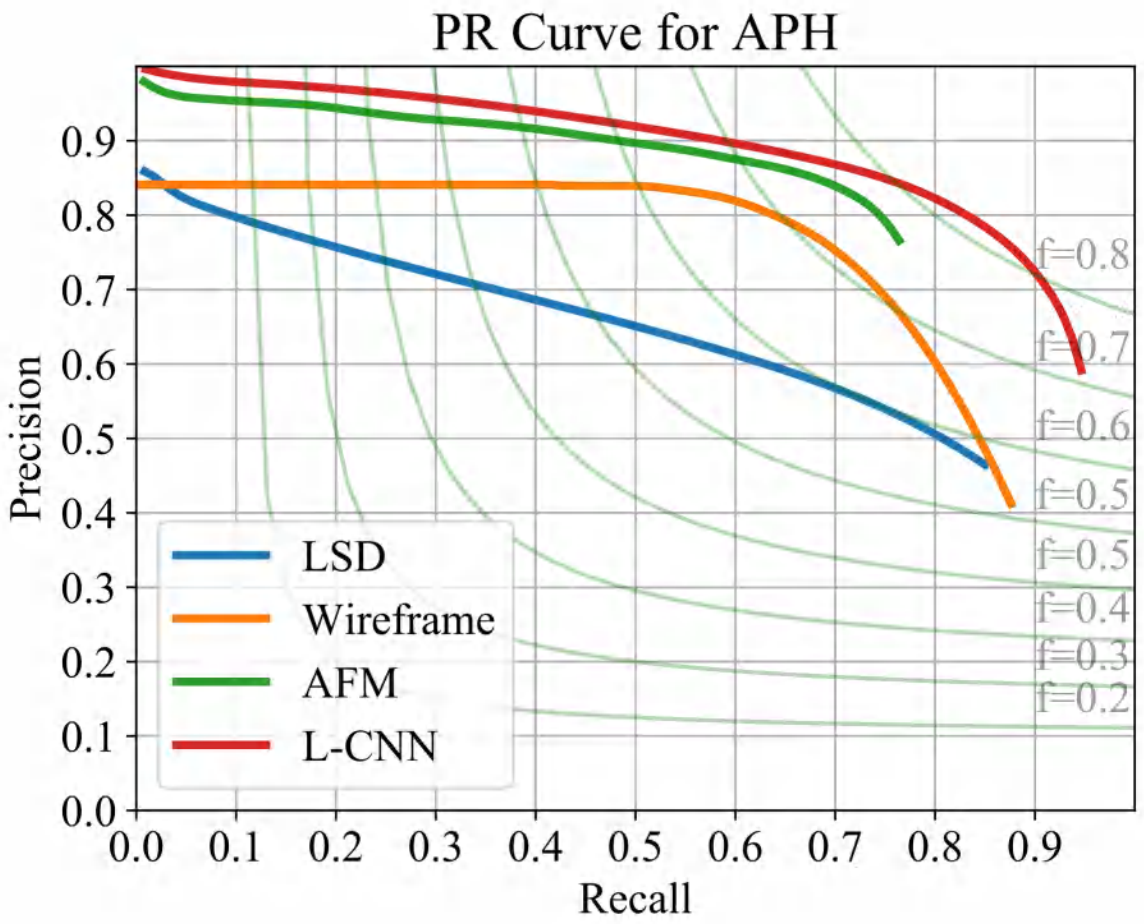
Quantitative Evaluation – Heat Map PR Curve

- Heat map-based score using pixel matching

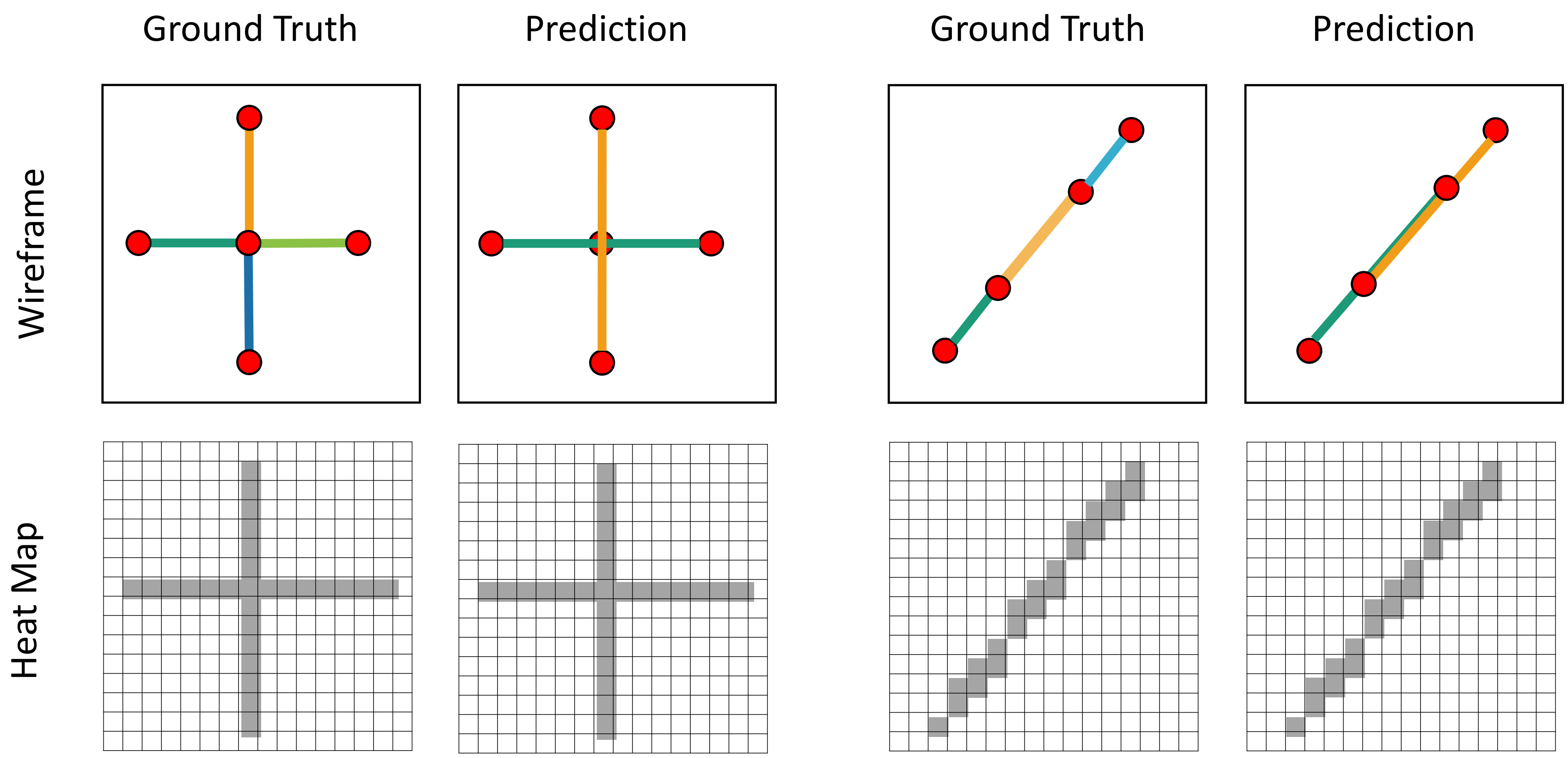
Heat Map Examples



Ground Truth   Prediction



Problem of Heat Map-based AP

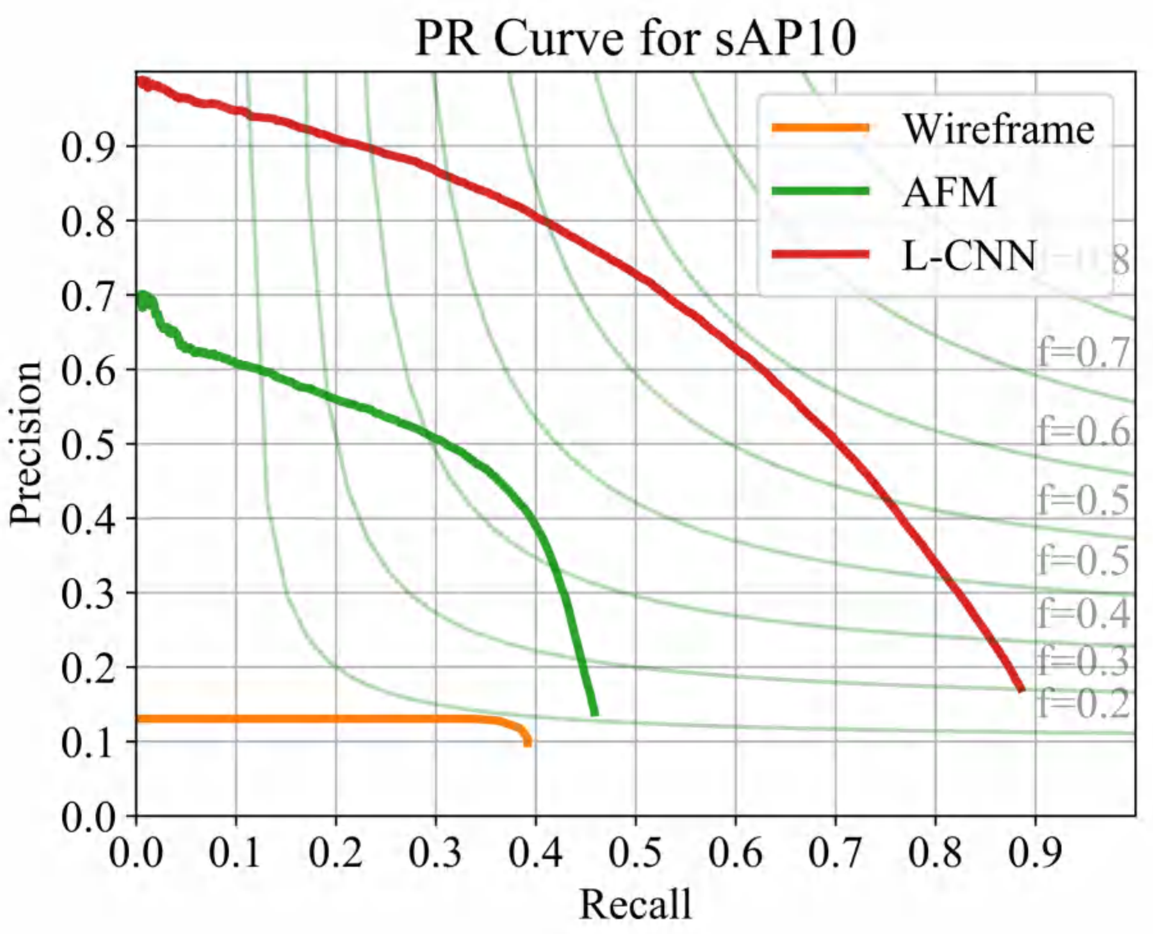


Incorrect connectivity

Overlapping lines

Quantitative Evaluation – Structural PR Curve

- **Line matching** score rather than pixel matching
- A line is considered correct iff there exists a ground truth whose end points are near the end points of this line
- Each ground truth line is allowed to be matched only once

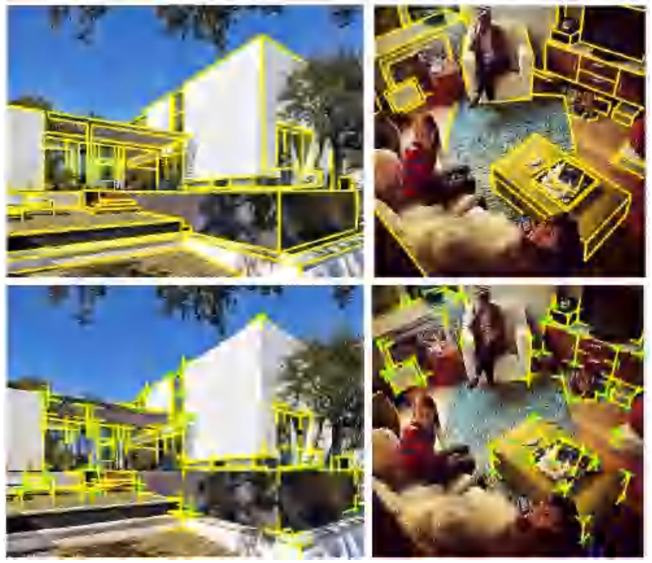


Summary

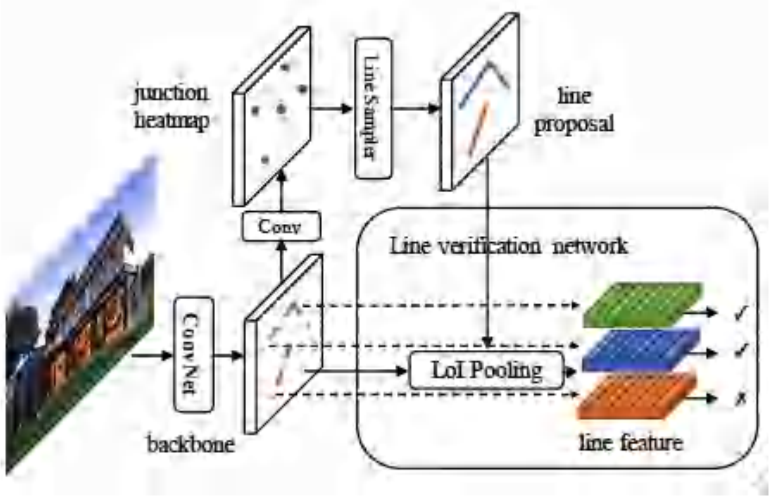
Geometric Methods	Learning-based Methods
• Detect structures by grouping local image cues in a bottom-up fashion	• Detect structures by learning from information provided by humans
• Sequential processing • Errors accumulate over stages	• End-to-end training • Backpropagate errors to all units in the network

Learning to Reconstruct 3D CAD Models – Three Pillars

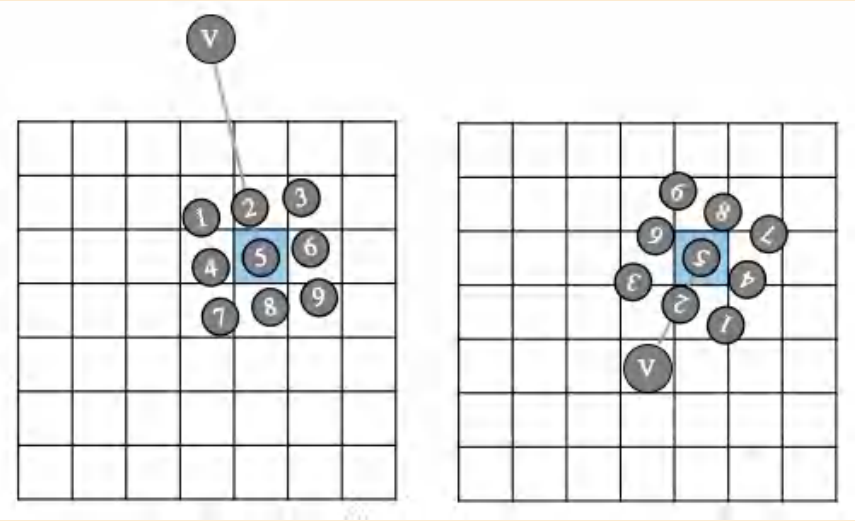
Data



Learning Scheme



Network Design



Vanishing Points

- Parallel lines in 3D intersect in one point after projection

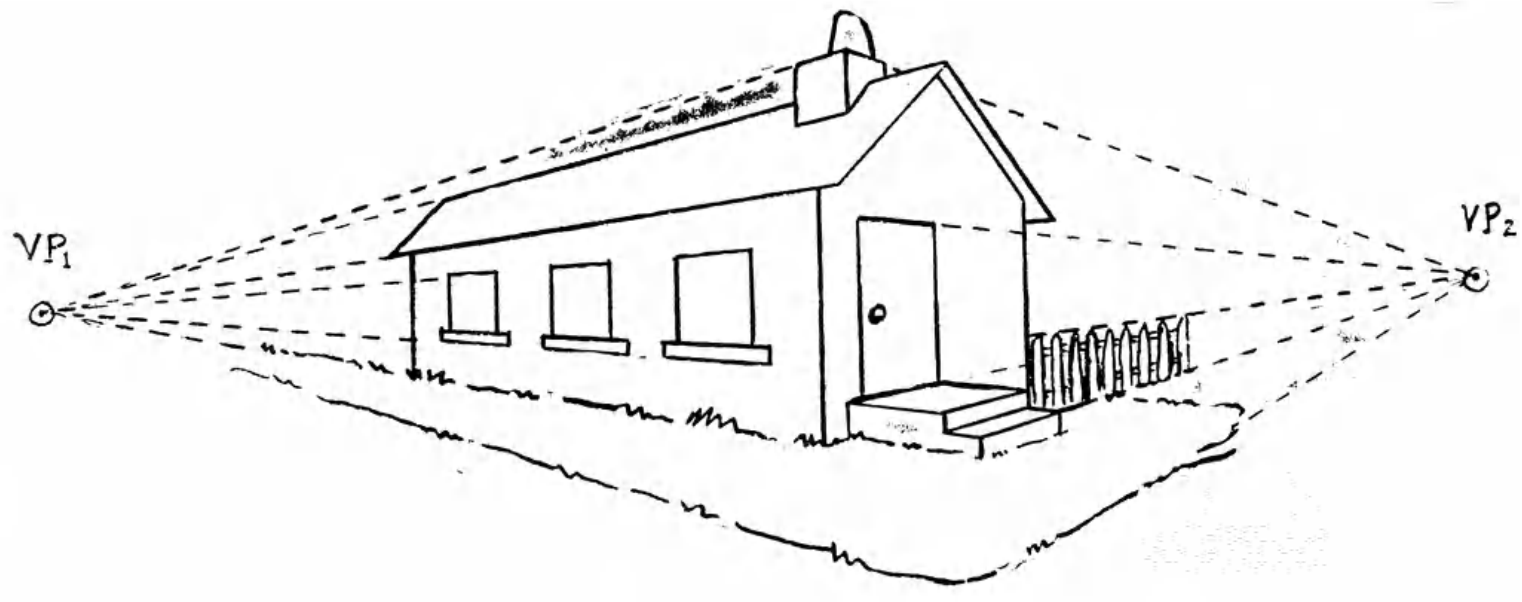
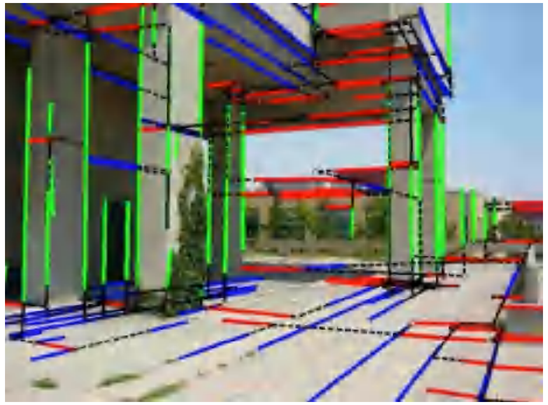
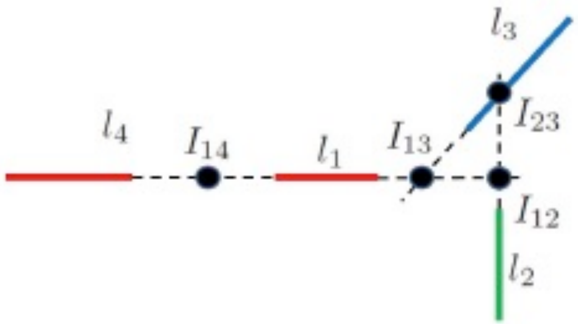


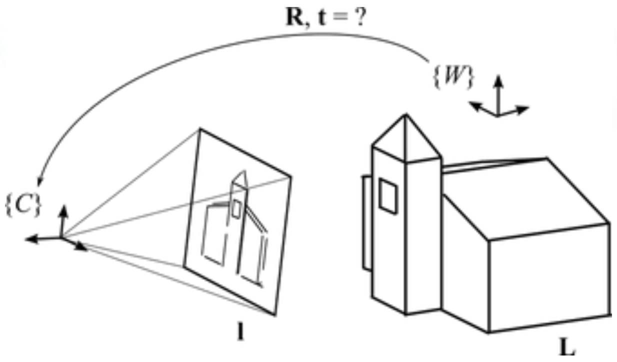
Image source: Military Science and Tactics.

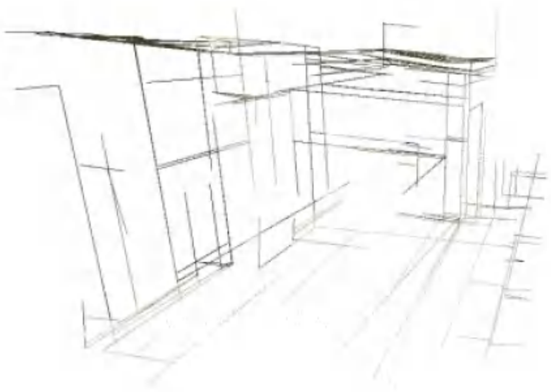
Role of VP in 3D Reconstruction

Line and vanishing point (VP) detection → Building constraint graph (junction detection) → VP-based camera pose estimation → Reconstruction via linear programming









Ramalingam and Brand. Lifting 3D Manhattan Lines from a Single Image. ICCV 2013.

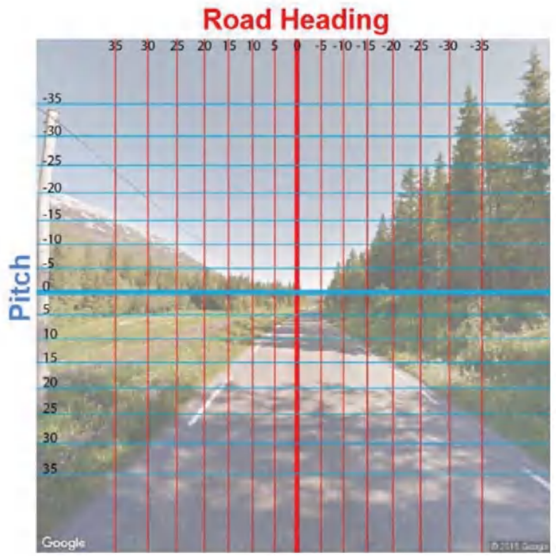
Geometric Methods

- Two-stage pipeline
  1. Line Segment Detection (e.g, LSD)
  2. Line Clustering (e.g., RANSAC)

Problem: Sensitive to missing lines and outliers

Deep Learning Methods

- Recent data-driven approaches
  - Do not rely on line detection
  - Divide image into patches and do classification



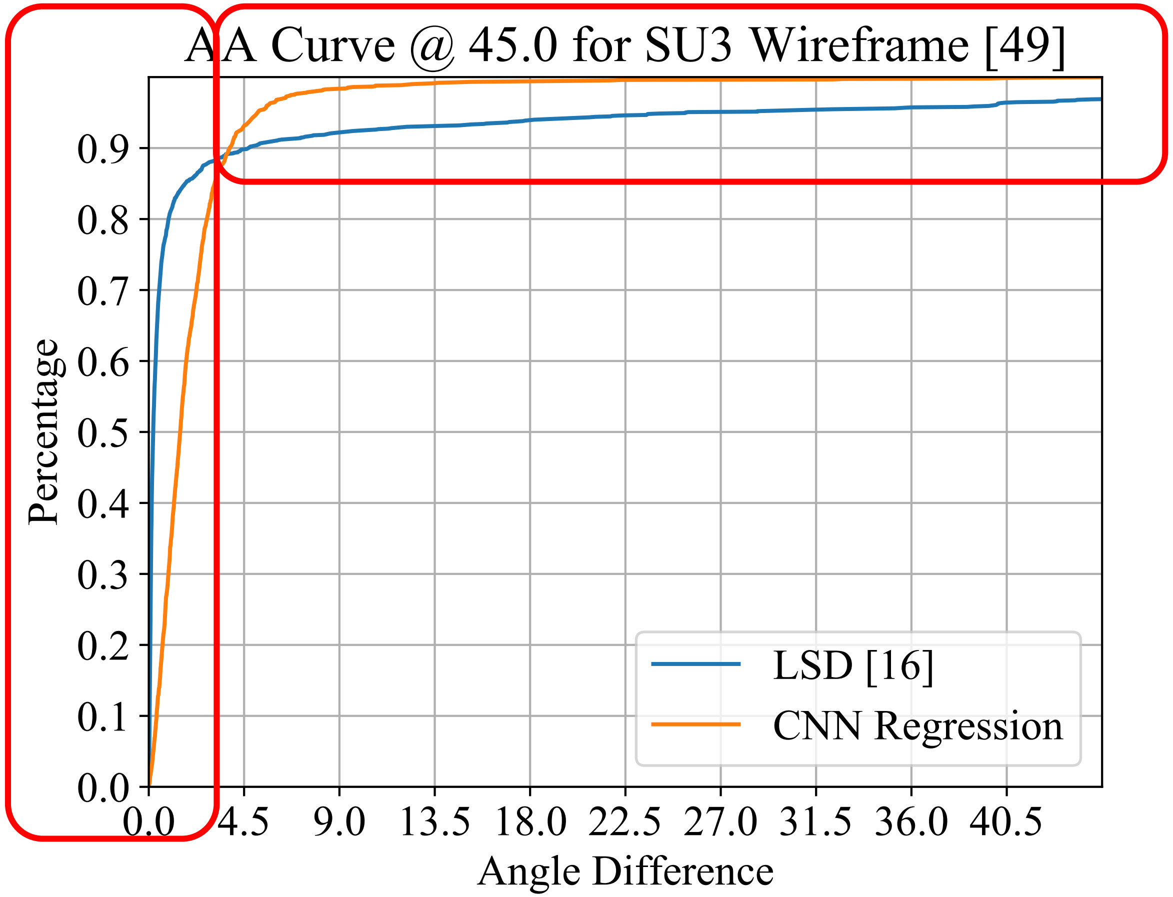
Problem: Hard to find vanishing point outside the image

[1] "Vanishing point detection with convolutional neural networks", Ali Borji, arXiv 2016
[2] "DeepVP: Deep learning for vanishing point detection on 1 million street view images", Chin-Kai Chang, Jiaping Zhao, and Laurent Itti. ICRA 2018
[3] "Dominant vanishing point detection in the wild with application in composition analysis", Xiaodan Zhang, Xinbo Gao, Wen Lu, Lihuo He, and Qi Liu. NeuralComputing 2018

Geometric vs. Learning-based

Deep learning method is more robust

Geometric method is more accurate



Design Philosophy of NeurVPS

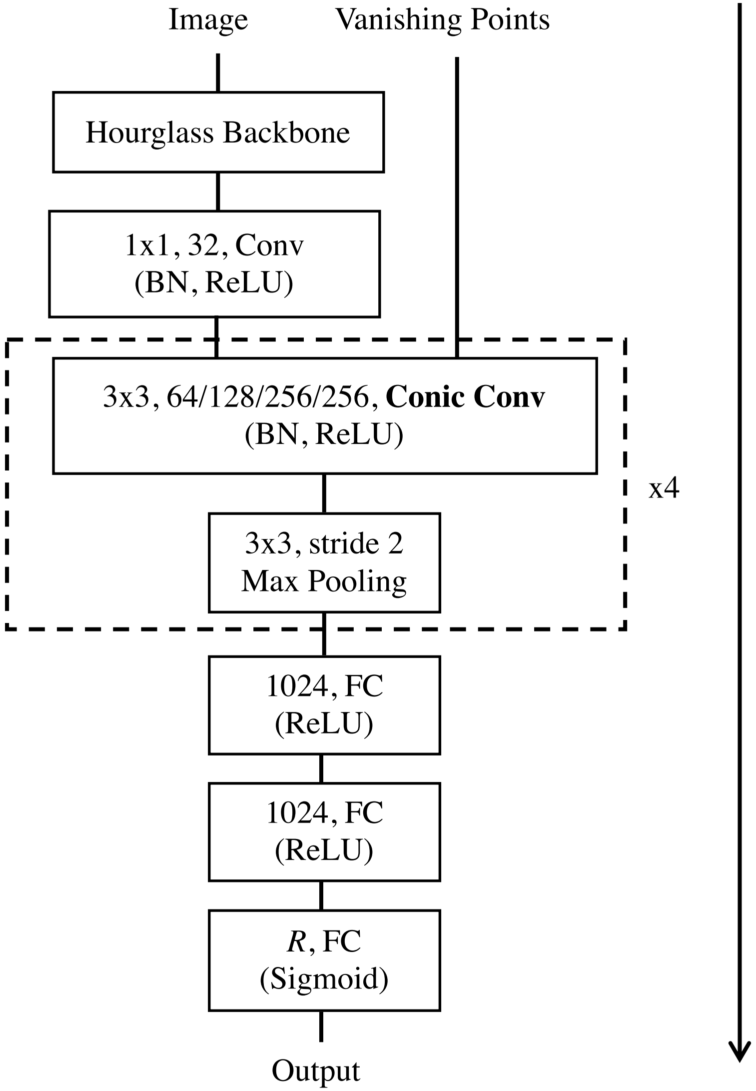
- The overall approach has the advantages of
  - ***accuracy*** of traditional line clustering algorithms
  - ***robustness*** of neural network-based algorithms

- New operators that captures geometric cues
  - vanishing points are the intersections of lines
  - operators should be *local* and *stackable*

Image Source: Wikipedia

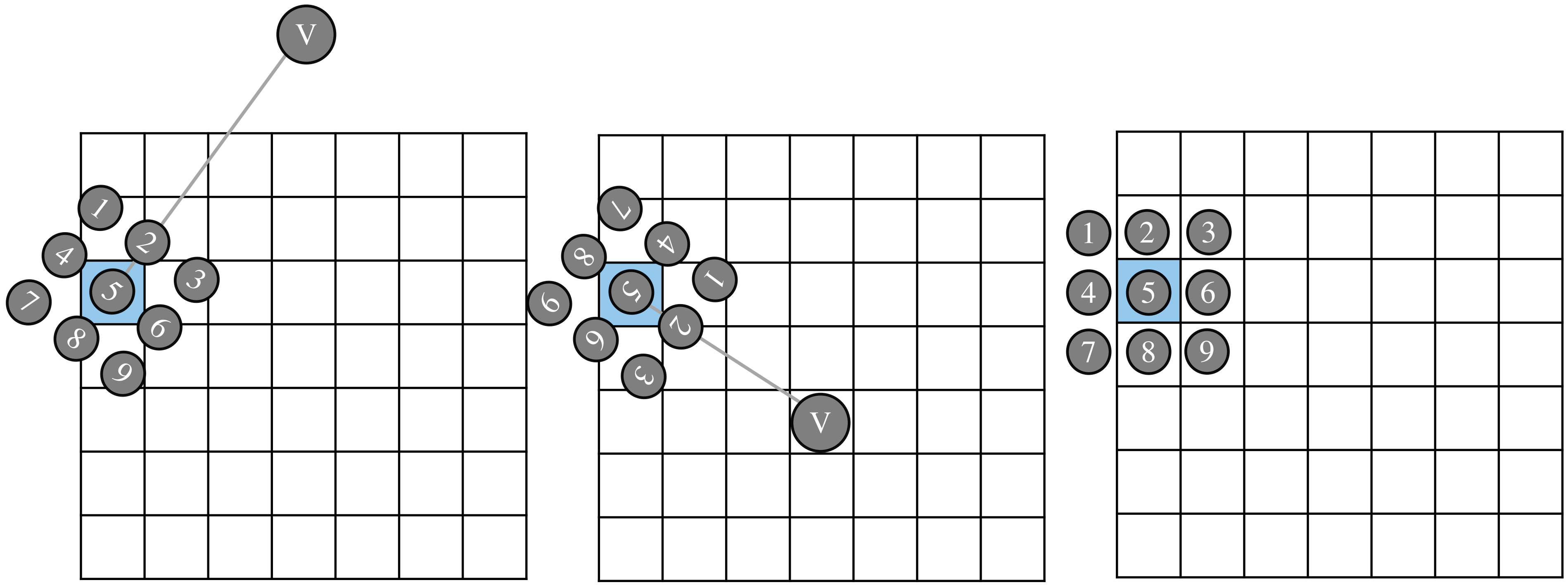
NeurVPS: Overall Pipeline

- Input
  - An image
  - A coordinate (vanishing point candidate)
- Output
  - likelihood of the existence of a vanishing point near that coordinate.
- Key Component
  - Conic Convolution



Conic Convolution

- Guided by vanishing point candidates (convolution center)



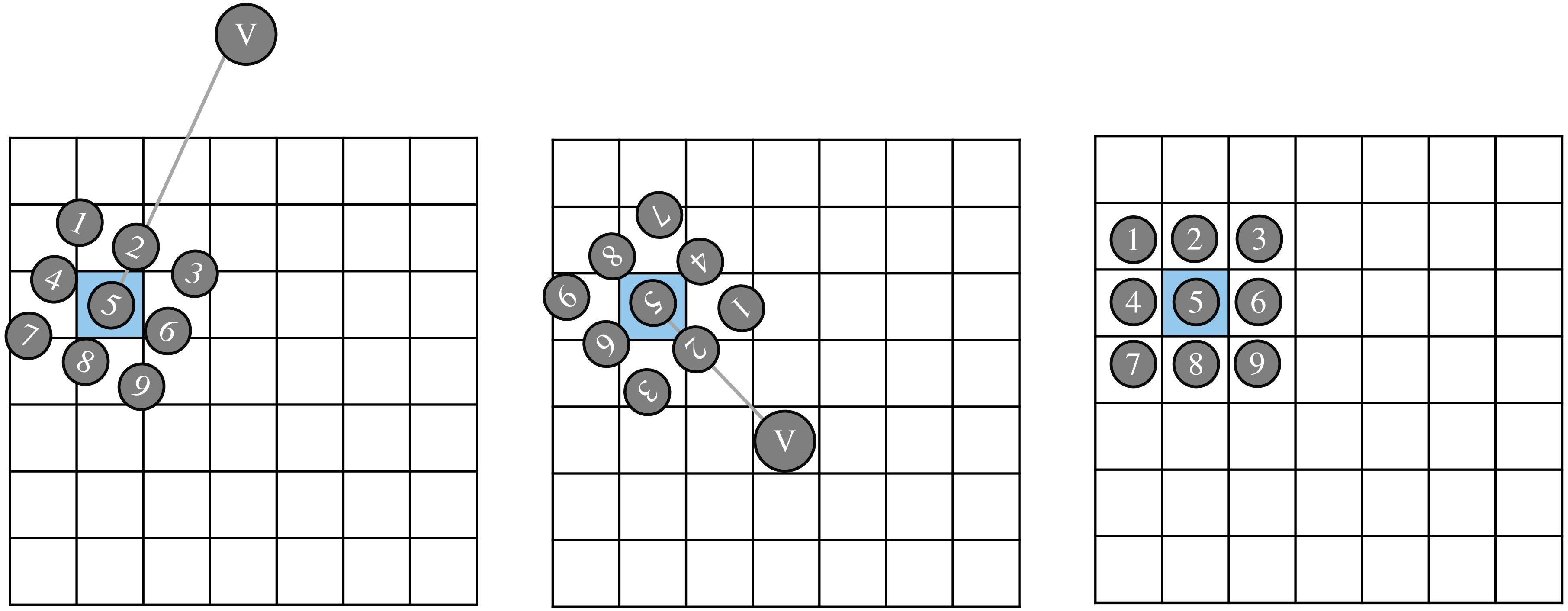
Conic Convolution
(vanishing point outside image plane)

Conic Convolution
(vanishing point inside image plane)

Plain Convolution

Conic Convolution

- Guided by vanishing point candidates (convolution center)



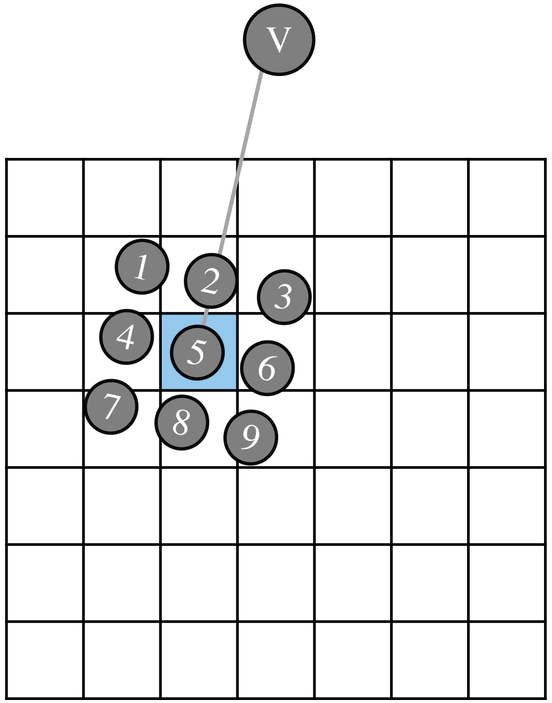
Conic Convolution
(vanishing point outside image plane)

Conic Convolution
(vanishing point inside image plane)

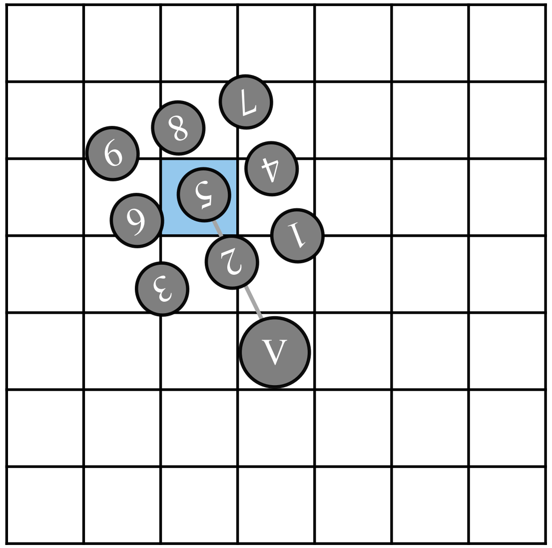
Plain Convolution

Conic Convolution

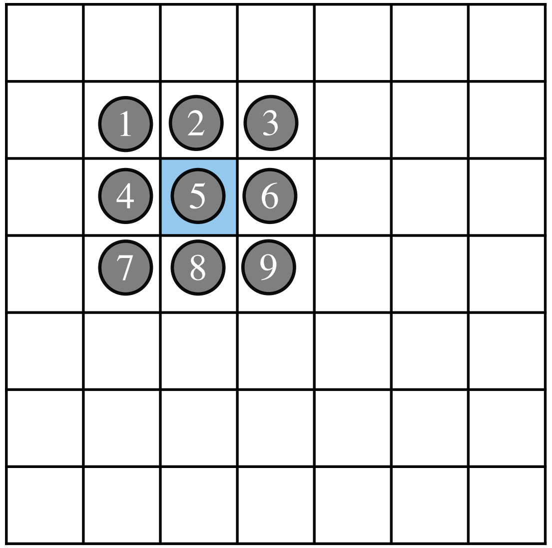
- Guided by vanishing point candidates (convolution center)



Conic Convolution
(vanishing point outside image plane)



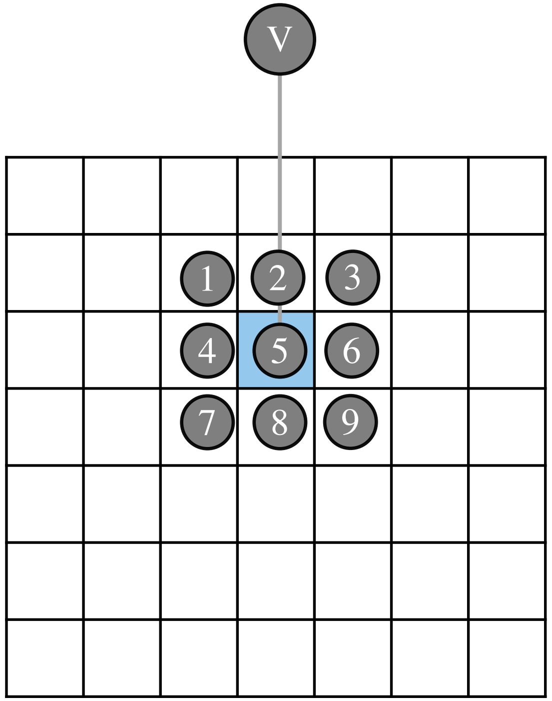
Conic Convolution
(vanishing point inside image plane)



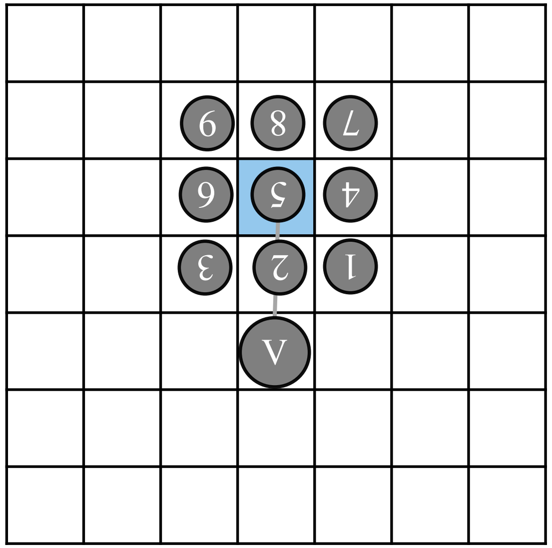
Plain Convolution

Conic Convolution

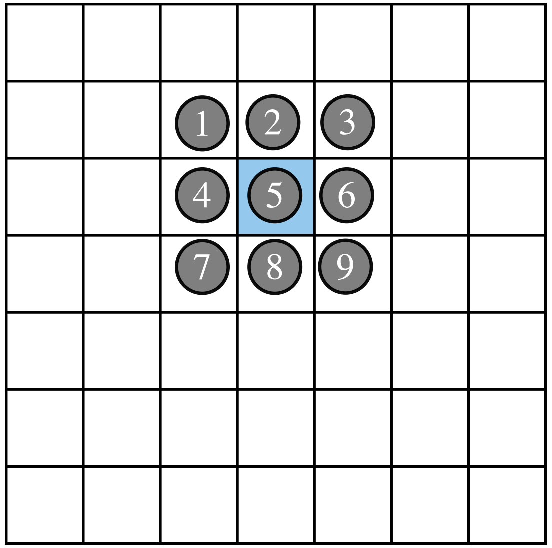
- Guided by vanishing point candidates (convolution center)



Conic Convolution
(vanishing point outside image plane)



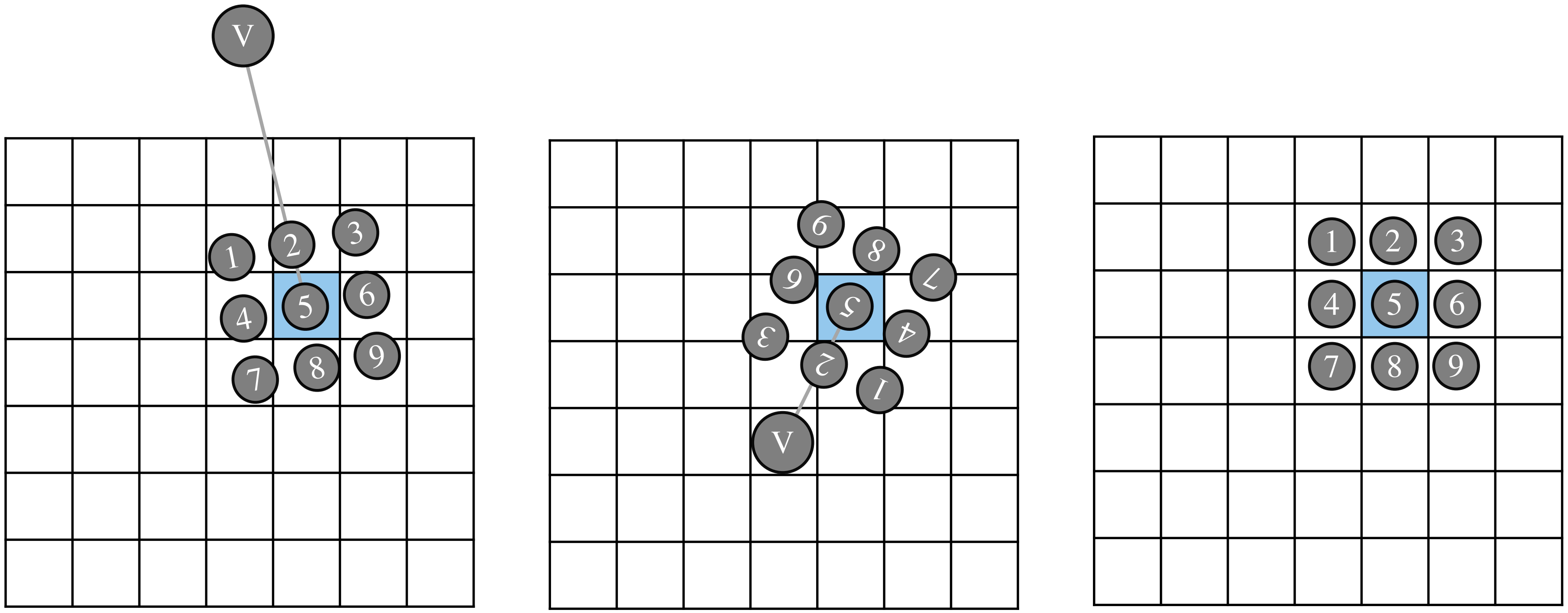
Conic Convolution
(vanishing point inside image plane)



Plain Convolution

Conic Convolution

- Guided by vanishing point candidates (convolution center)

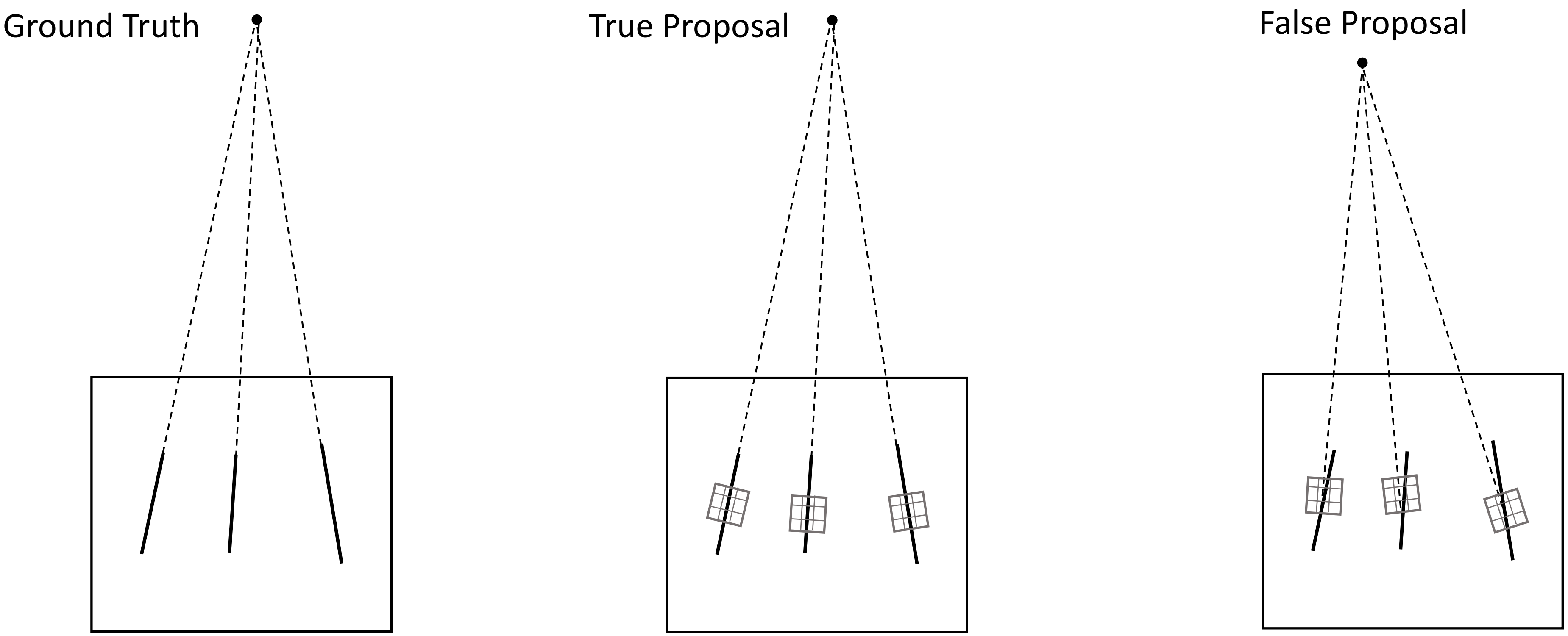


Conic Convolution
(vanishing point outside image plane)

Conic Convolution
(vanishing point inside image plane)

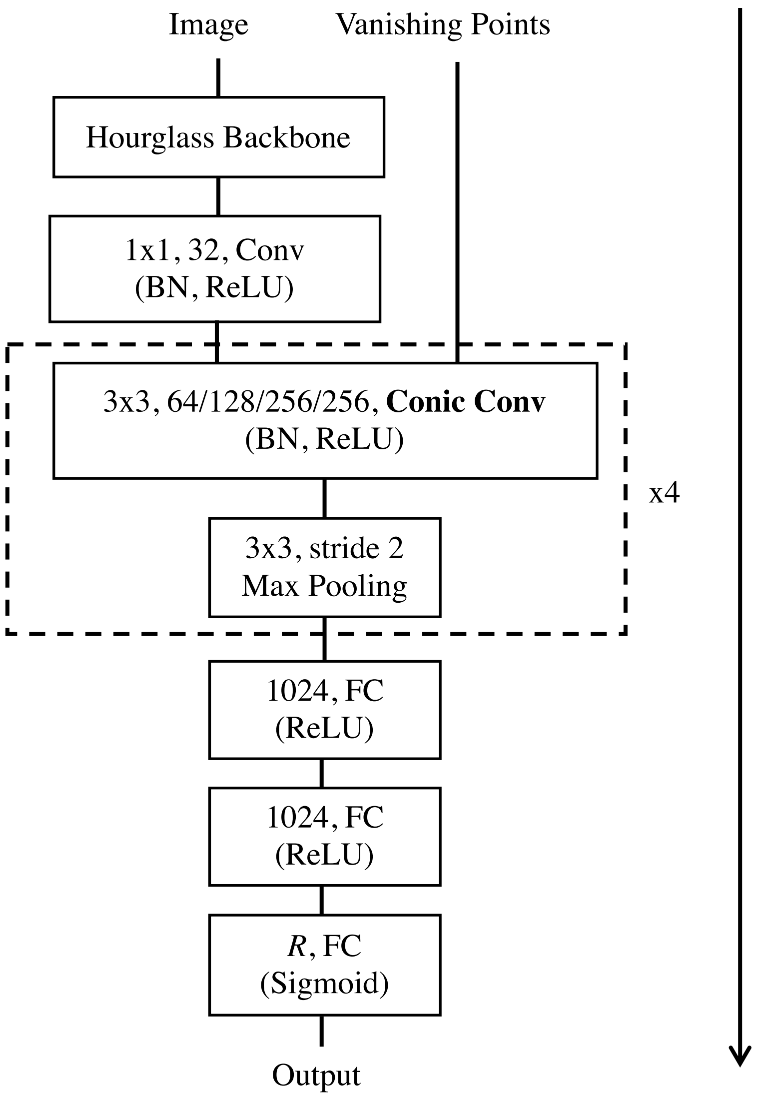
Plain Convolution

Intuition Behind Conic Convolution



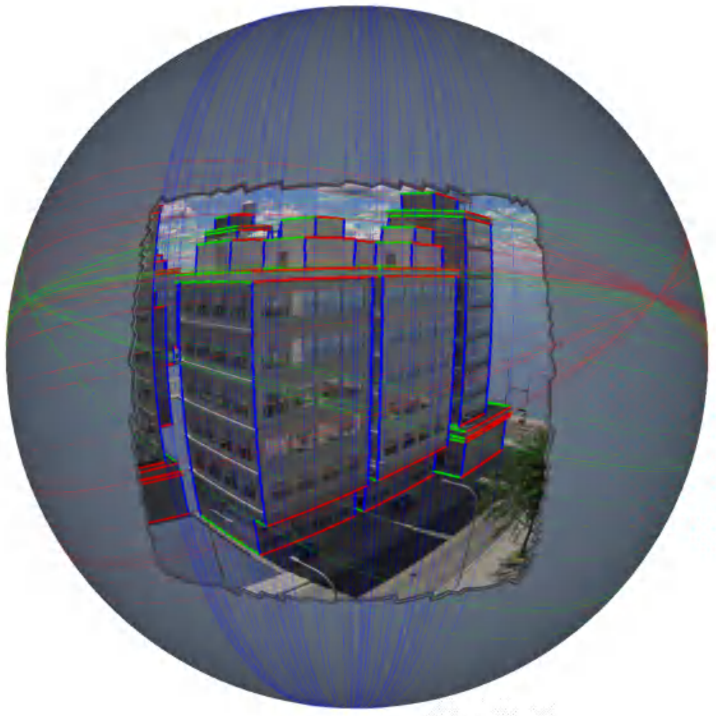
Coarse-to-Fine Inference

- The network is essentially a vanishing point classifier
- During evaluation
  1. Sample vanishing points
  2. Test it with our network classifier
- How to uniformly sample vanishing points?

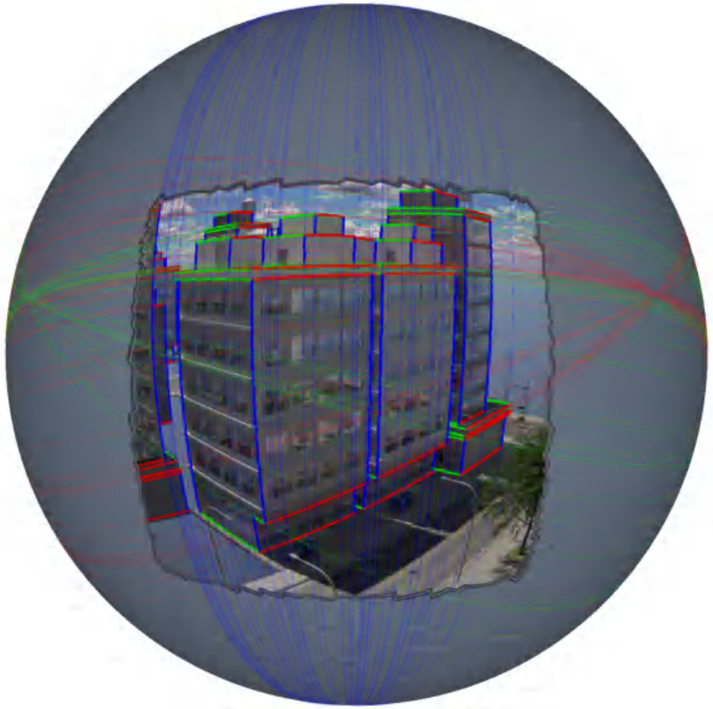


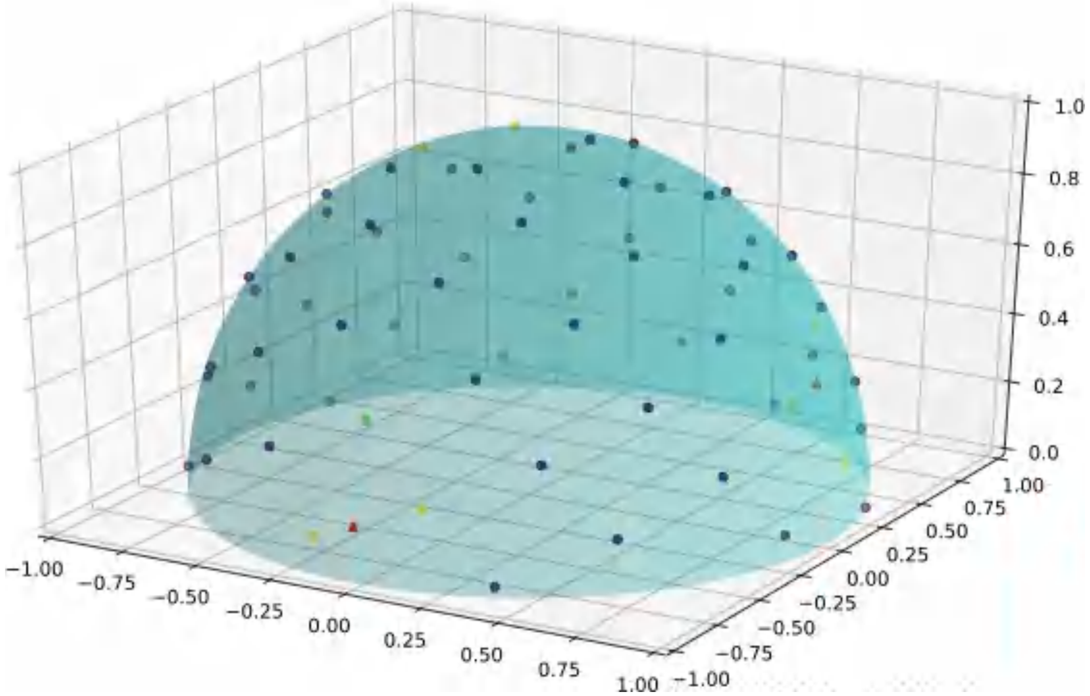
Gaussian Sphere

- Each VP corresponds to a point on the Gaussian Sphere (a 3D unit vector)

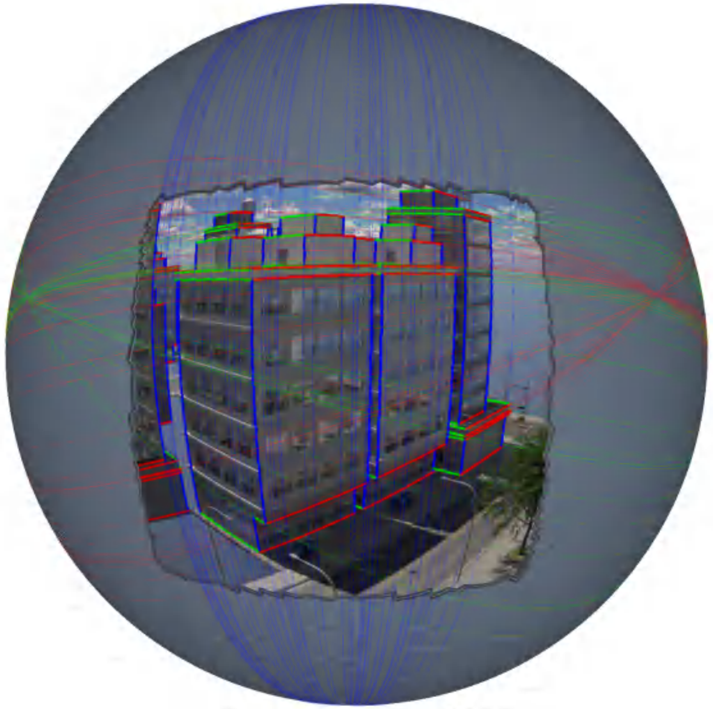


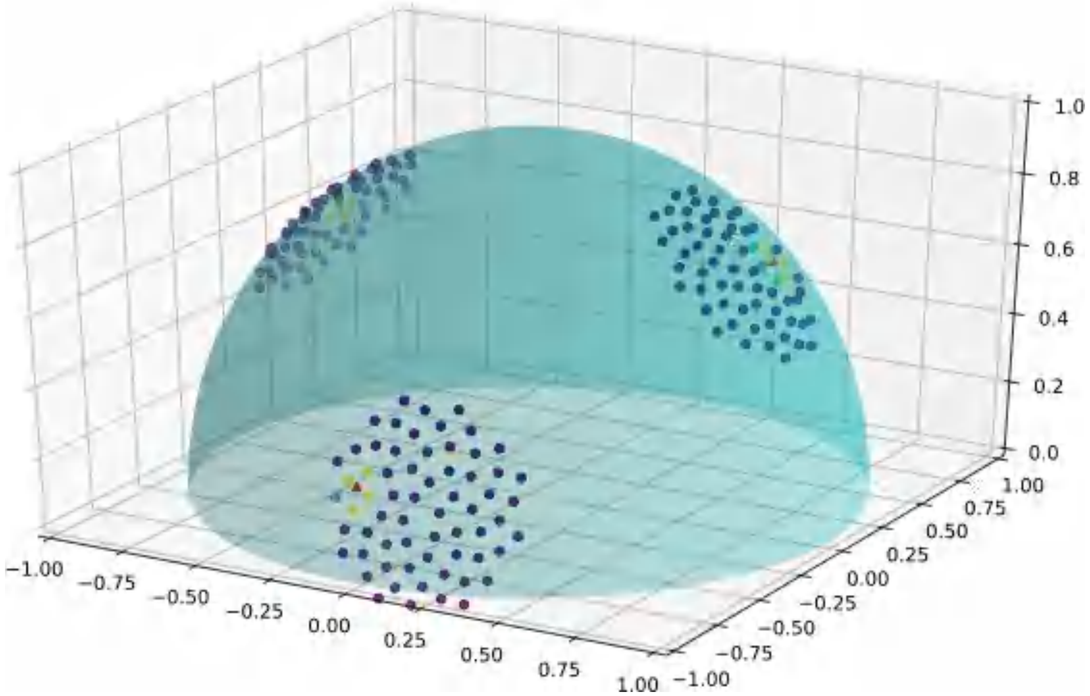
Hierarchical Inference



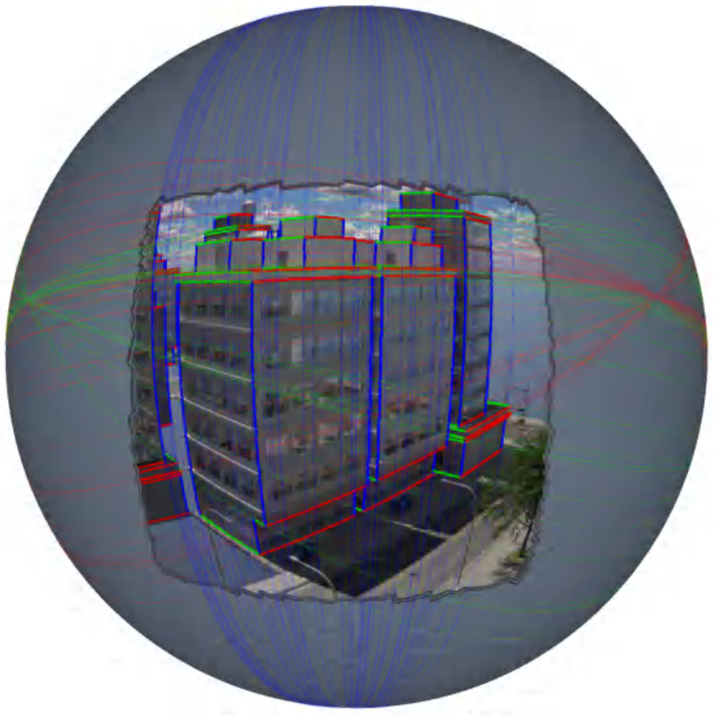


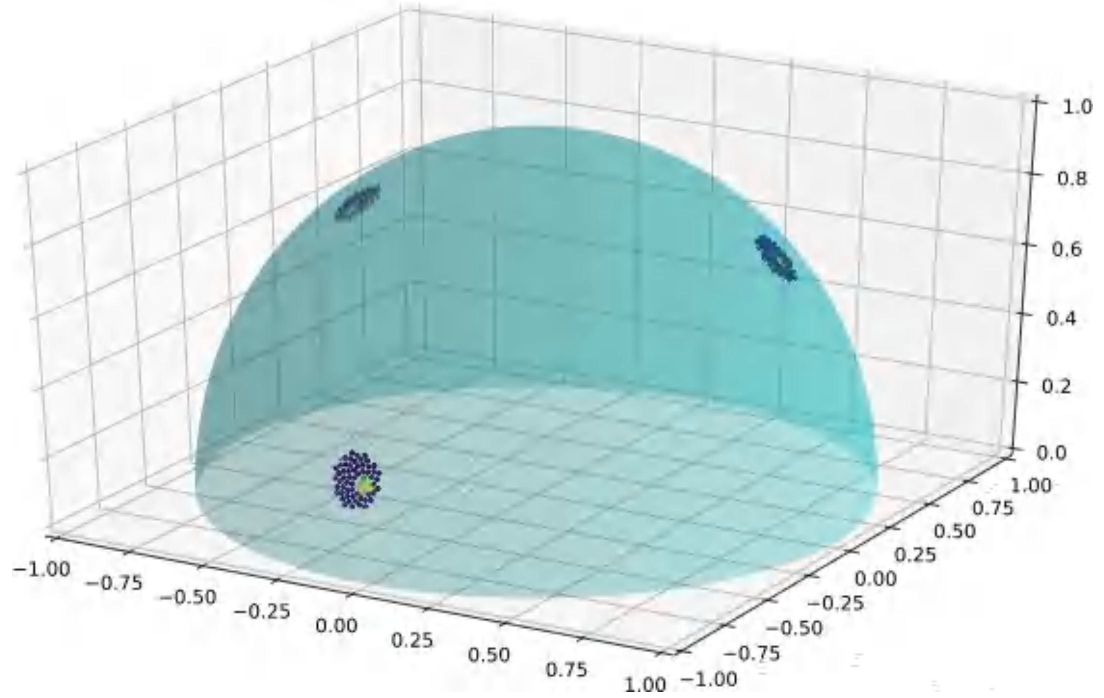
Hierarchical Inference





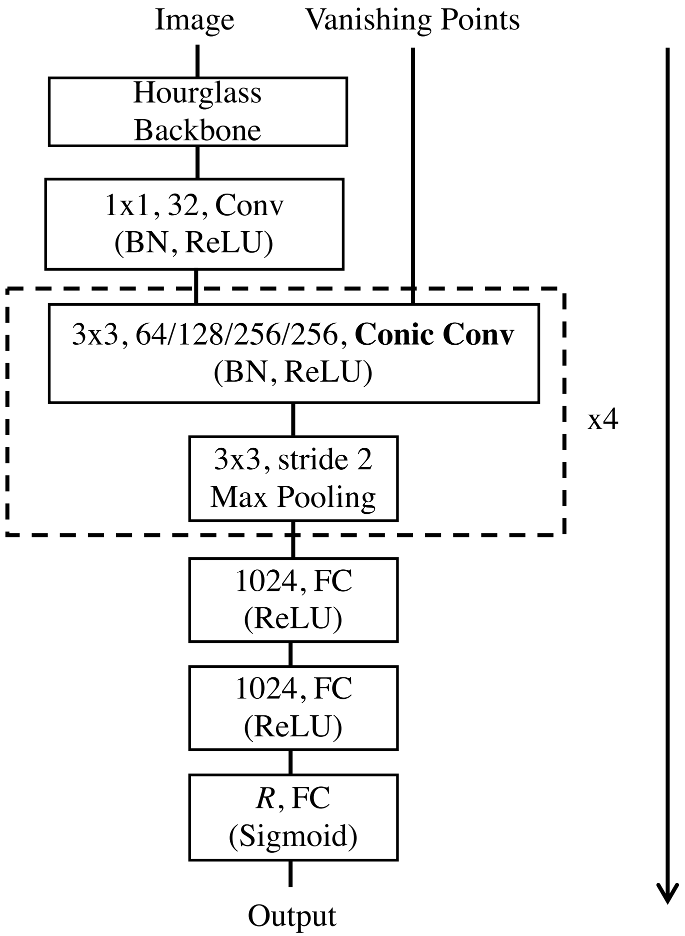
Hierarchical Inference





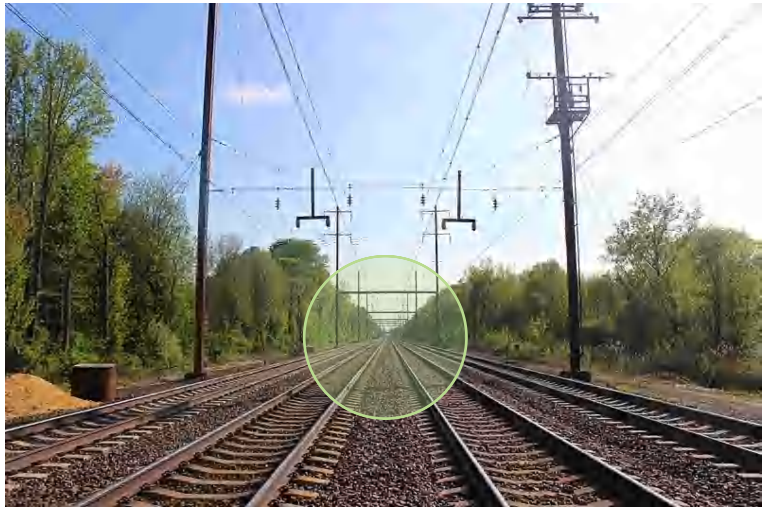
Training

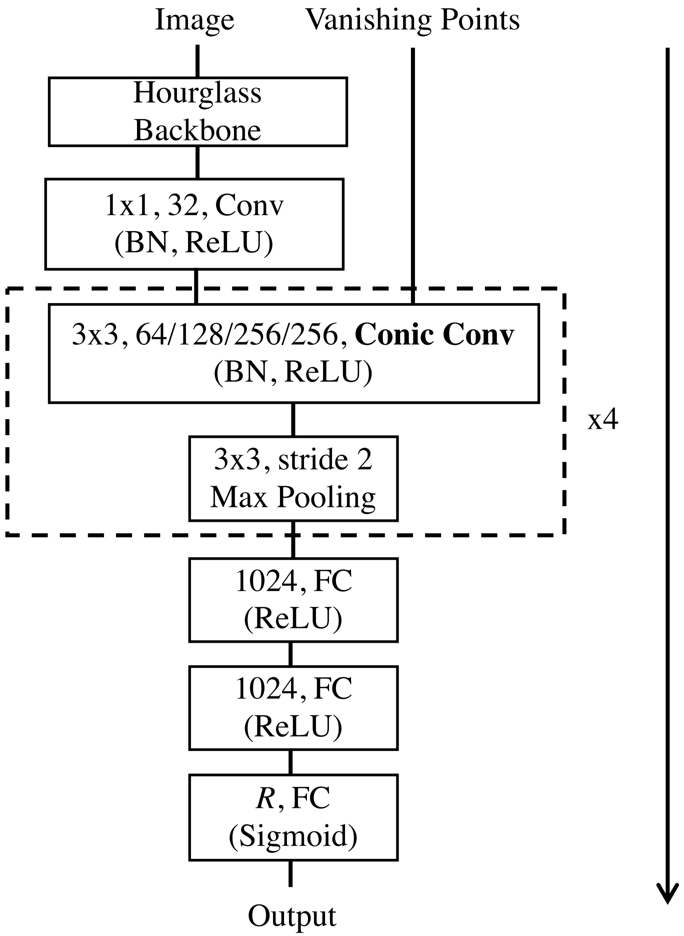
- Train multiple classifiers, each of which corresponds to a different threshold;
- Sample one positive & one negative vanishing points for each threshold;
- Randomly sample three vanishing points to reduce bias.



Training

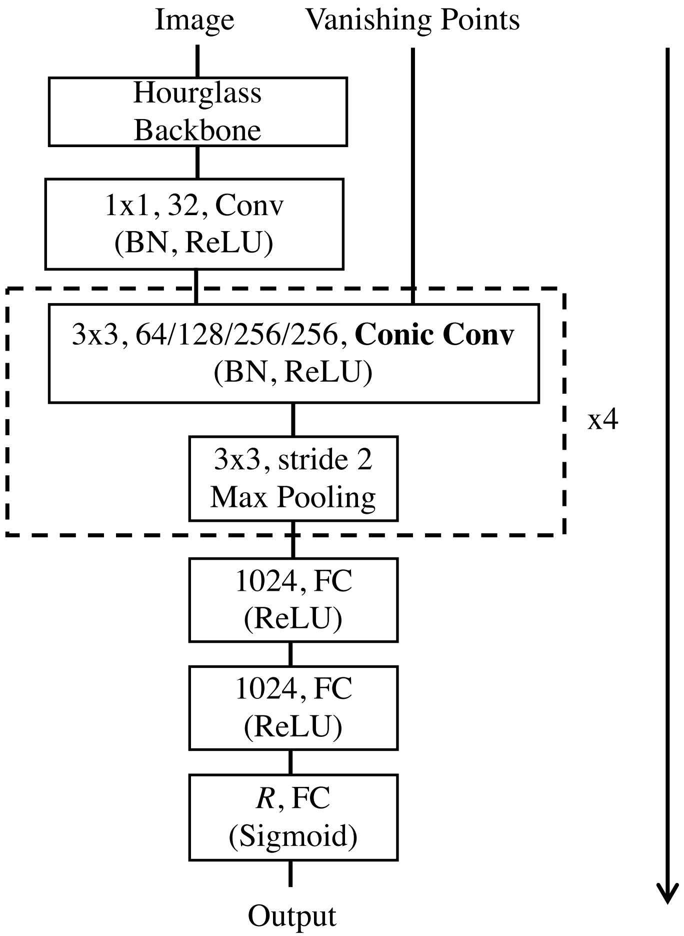
- Train multiple classifiers, each of which corresponds to a different threshold;
- Sample one positive & one negative vanishing points for each threshold;
- Randomly sample three vanishing points to reduce bias.





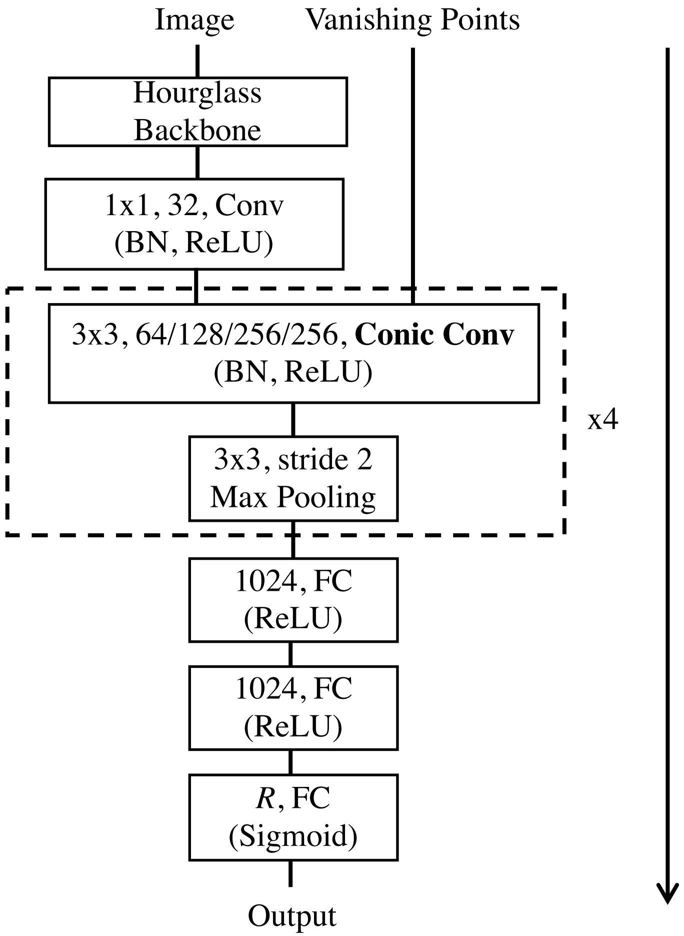
Training

- Train multiple classifiers, each of which corresponds to a different threshold;
- Sample one positive & one negative vanishing points for each threshold;
- Randomly sample three vanishing points to reduce bias.



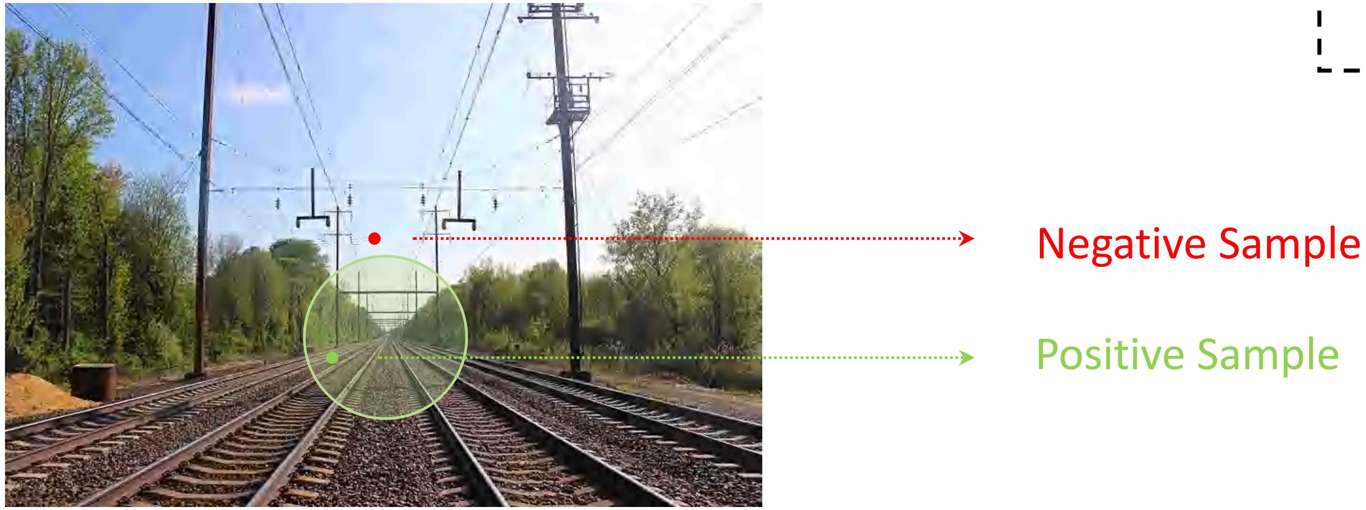
Training

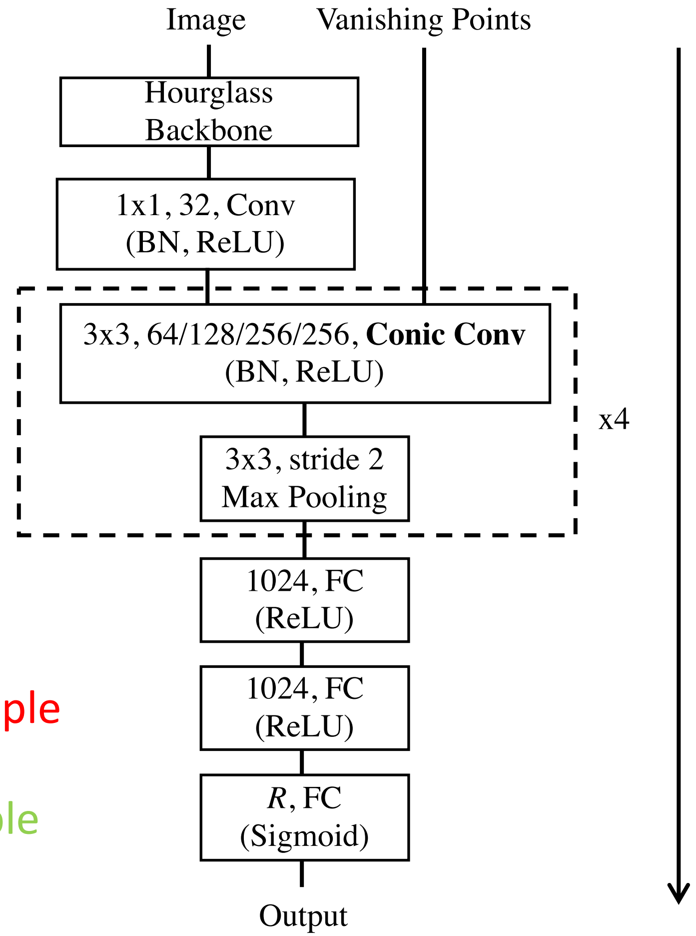
- Train multiple classifiers, each of which corresponds to a different threshold;
- Sample one positive & one negative vanishing points for each threshold;
- Randomly sample three vanishing points to reduce bias.



Training

- Train multiple classifiers, each of which corresponds to a different threshold;
- Sample one positive & one negative vanishing points for each threshold;
- Randomly sample three vanishing points to reduce bias.

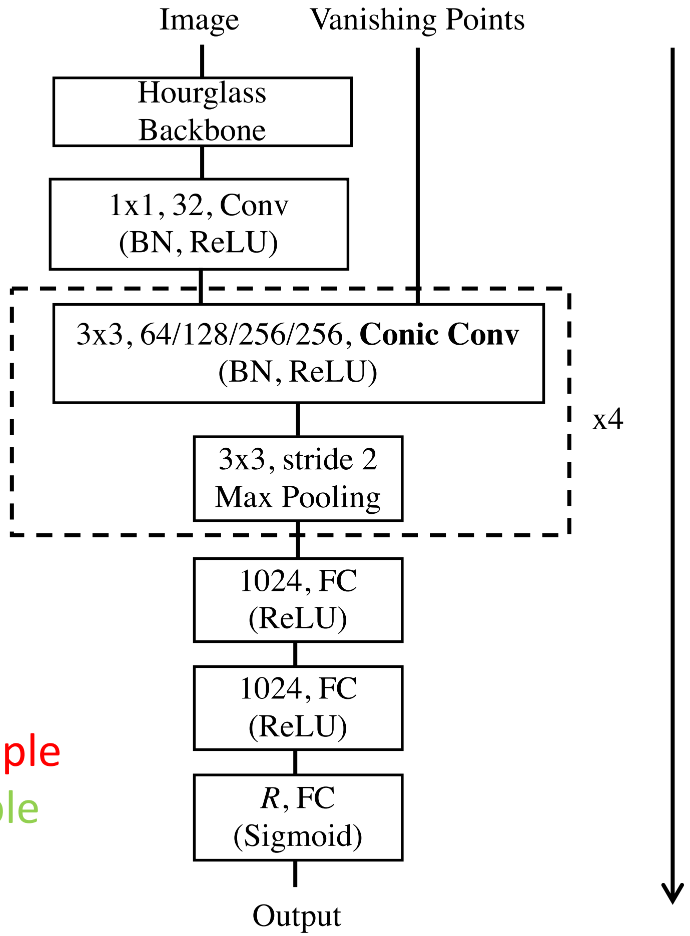




Training

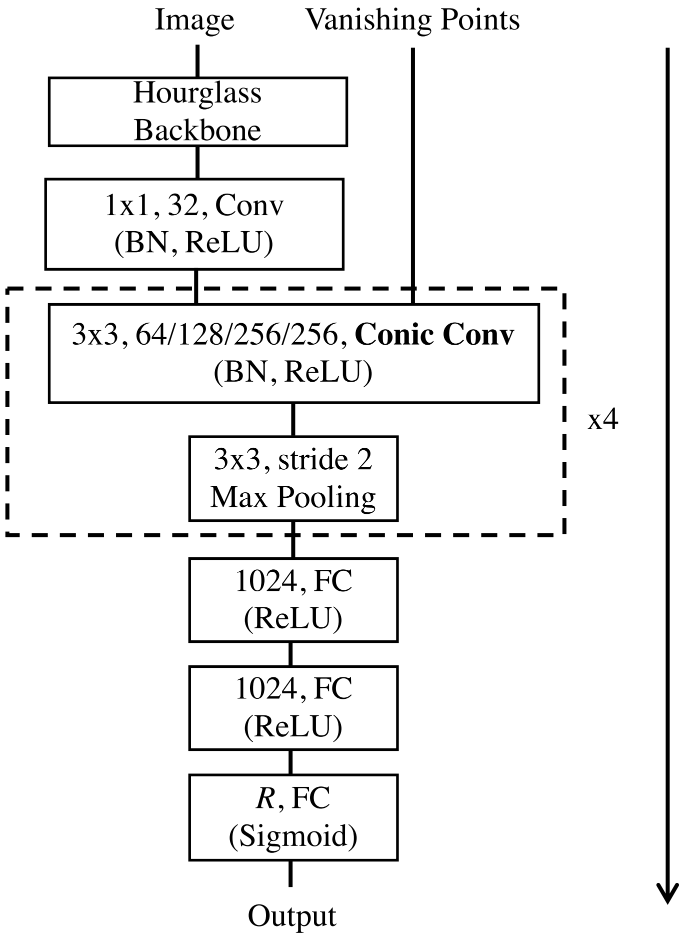
- Train multiple classifiers, each of which corresponds to a different threshold;
- Sample one positive & one negative vanishing points for each threshold;
- Randomly sample three vanishing points to reduce bias.

Negative Sample
Positive Sample



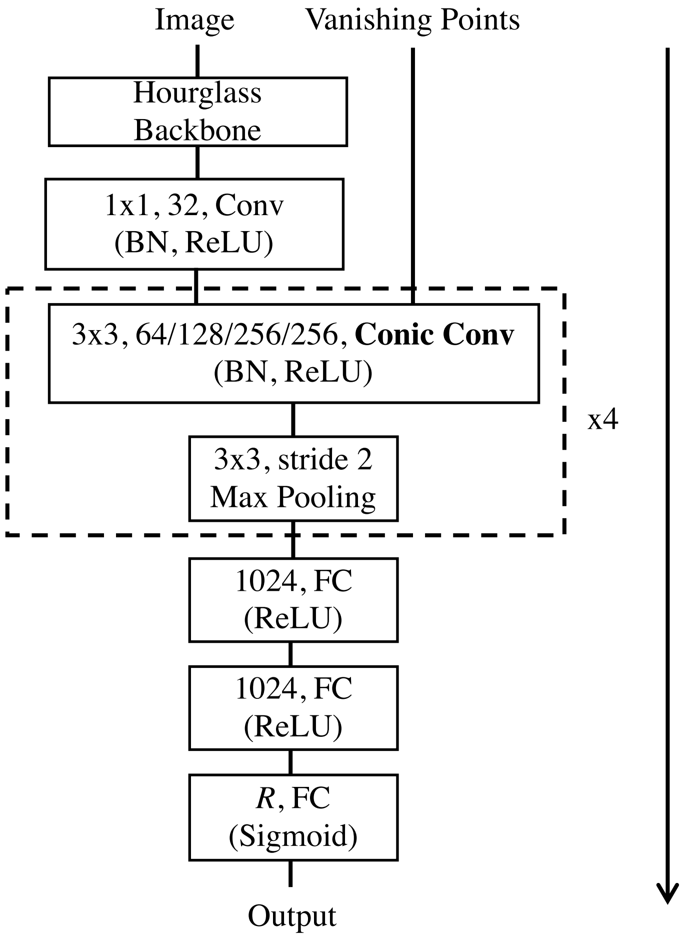
Training

- Train multiple classifiers, each of which corresponds to a different threshold;
- Sample one positive & one negative vanishing points for each threshold;
- Randomly sample three vanishing points to reduce bias.



Training

- Train multiple classifiers, each of which corresponds to a different threshold;
- Sample one positive & one negative vanishing points for each threshold;
- Randomly sample three vanishing points to reduce bias.



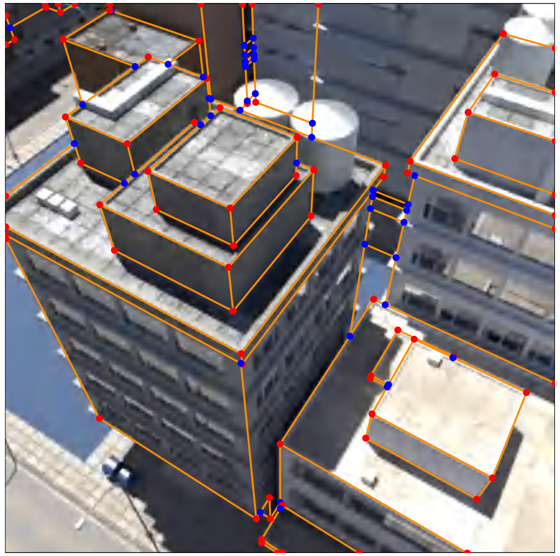
Experiment Settings

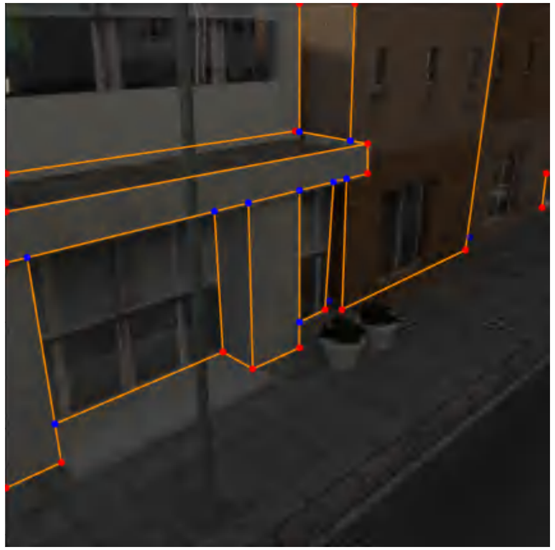
- ConicConv
  - Testing with different number of layers
    - **2 conic** convolution layers
    - **4 conic** convolution layers
    - **6 conic** convolution layers
- Geometric baselines
  - **LSD + J-Linkage [1]**
  - **Contour + J-Linkage [2]** (Only for dominating vanishing point detection)
- Deep learning baselines:
  - Use the same number of parameters as 4x ConicConv
  - **REG**: directly regress the vanishing point coordinates
  - **CLS**: use vanishing point coordinates as features and do classification

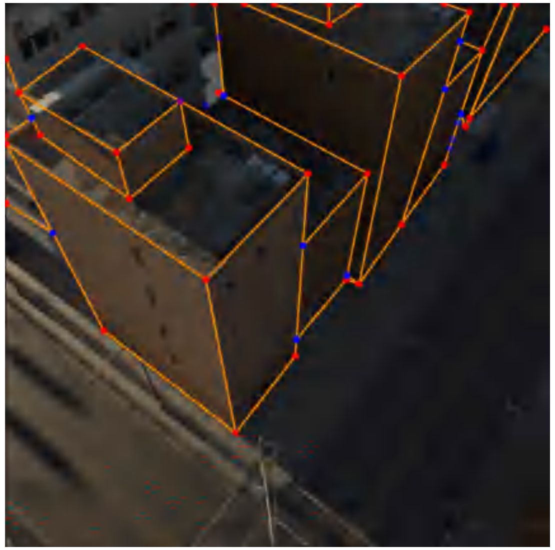
[1] “Semi-automatic 3D Reconstruction of Piecewise Planar Building Models From Single Image ” Chen Feng, Fei Deng, Vineet R. Kamat.

[2] “Detecting Dominant Vanishing Points in Natural Scenes with Application to Composition-Sensitive Image Retrieval” Zihan Zhou, Farshid Farhat, and James Z. Wang

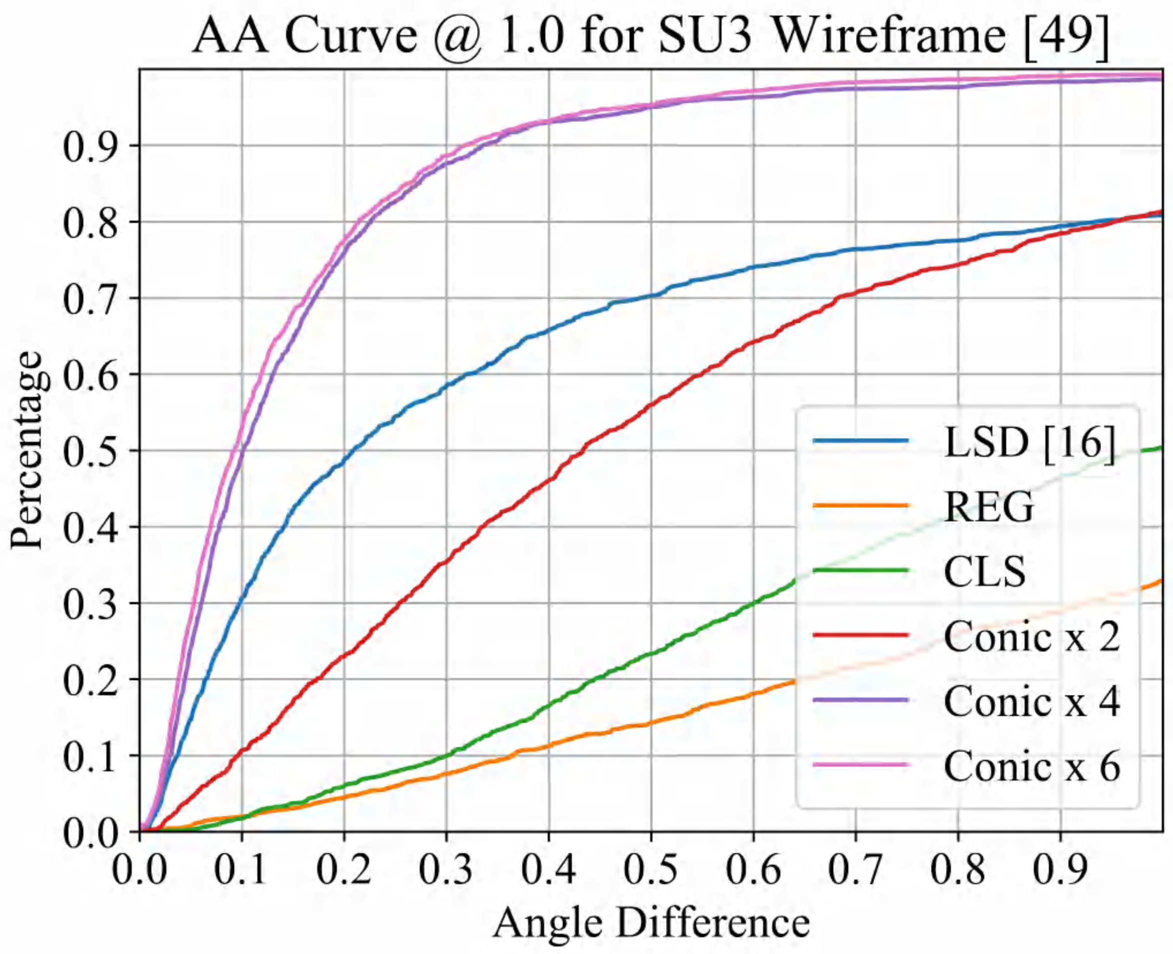
Synthetic Urban 3D Dataset



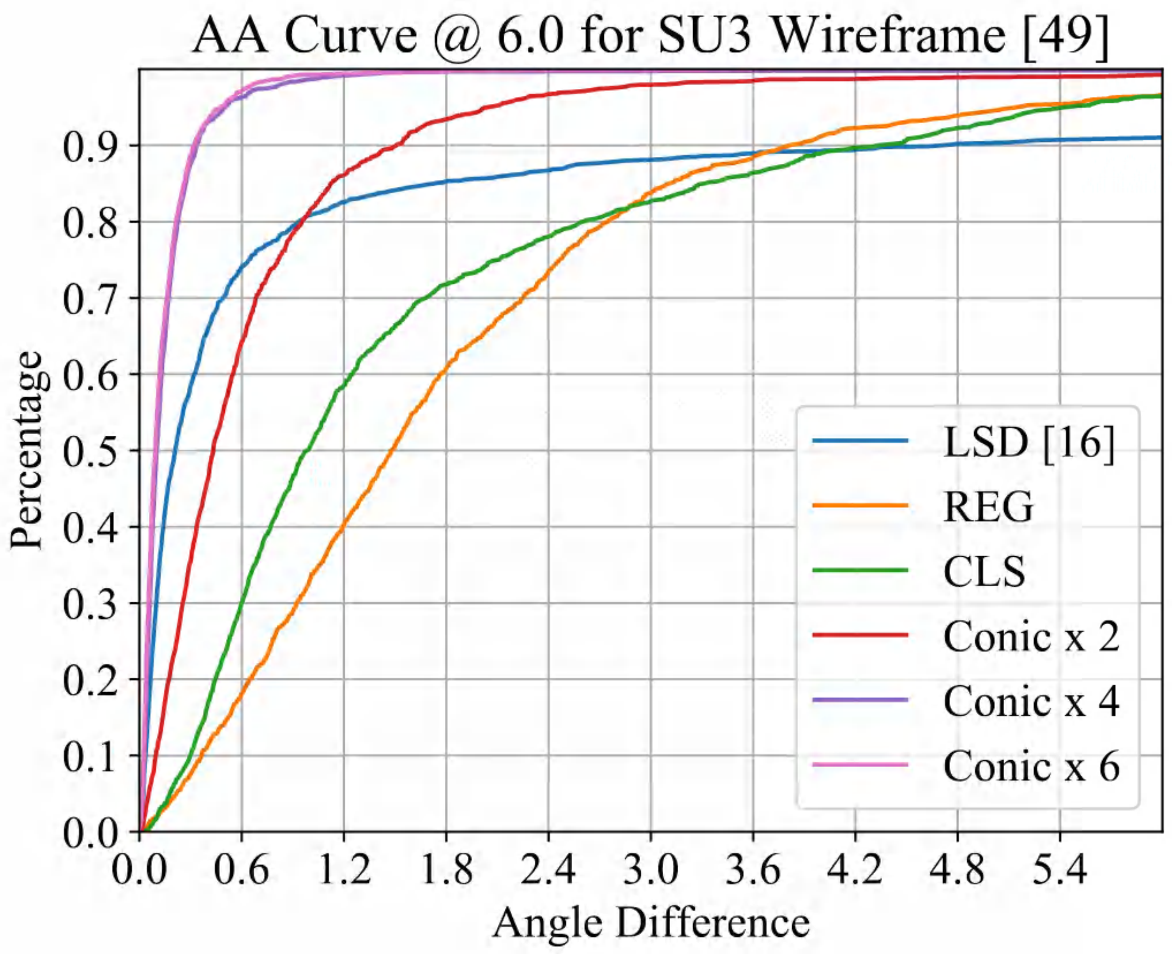




Zhou, Yichao, et al. "Learning to Reconstruct 3D Manhattan Wireframes from a Single Image." arXiv preprint arXiv:1905.07482 (2019).

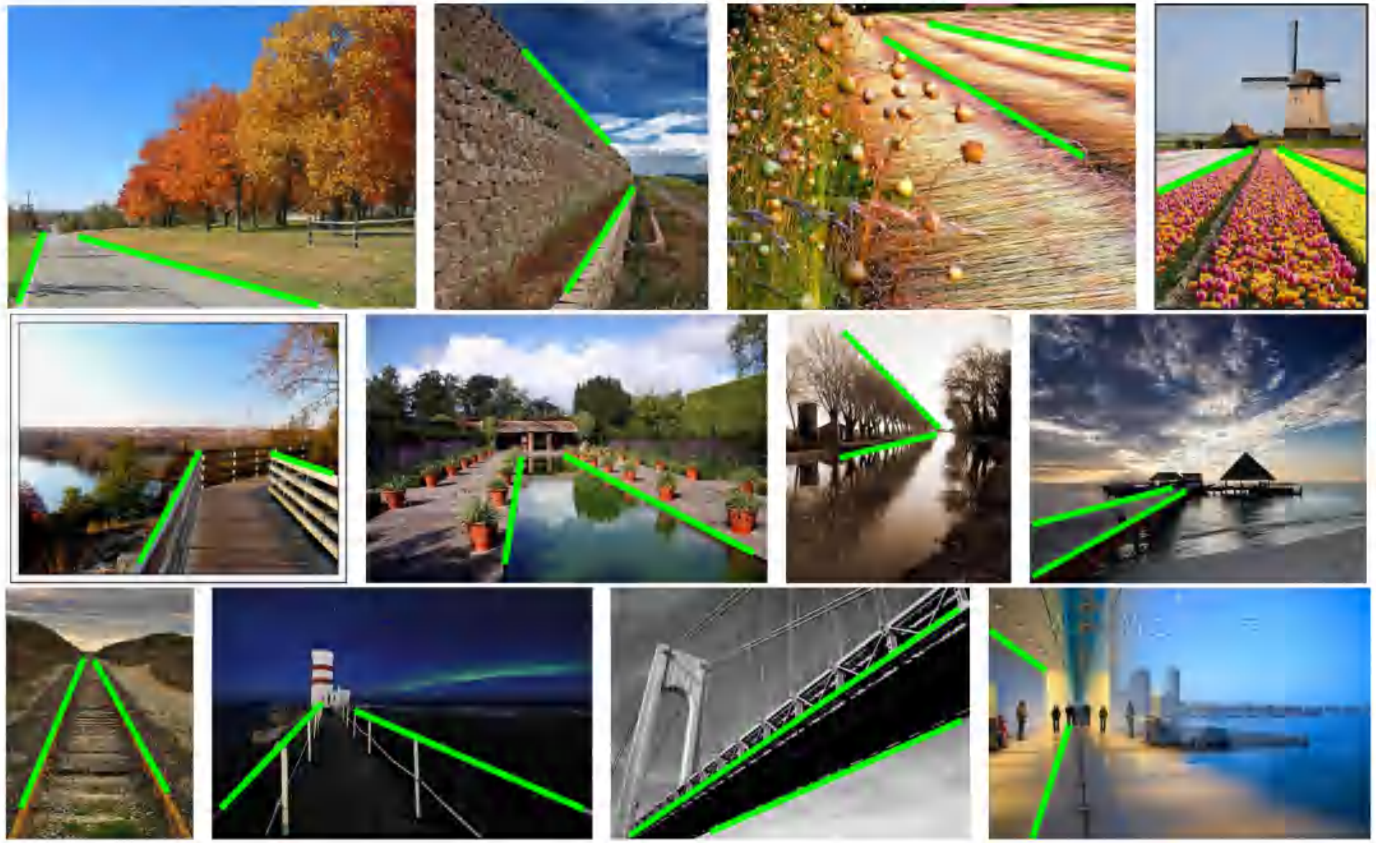


(a) Angle difference ranges from 0° to 1°.

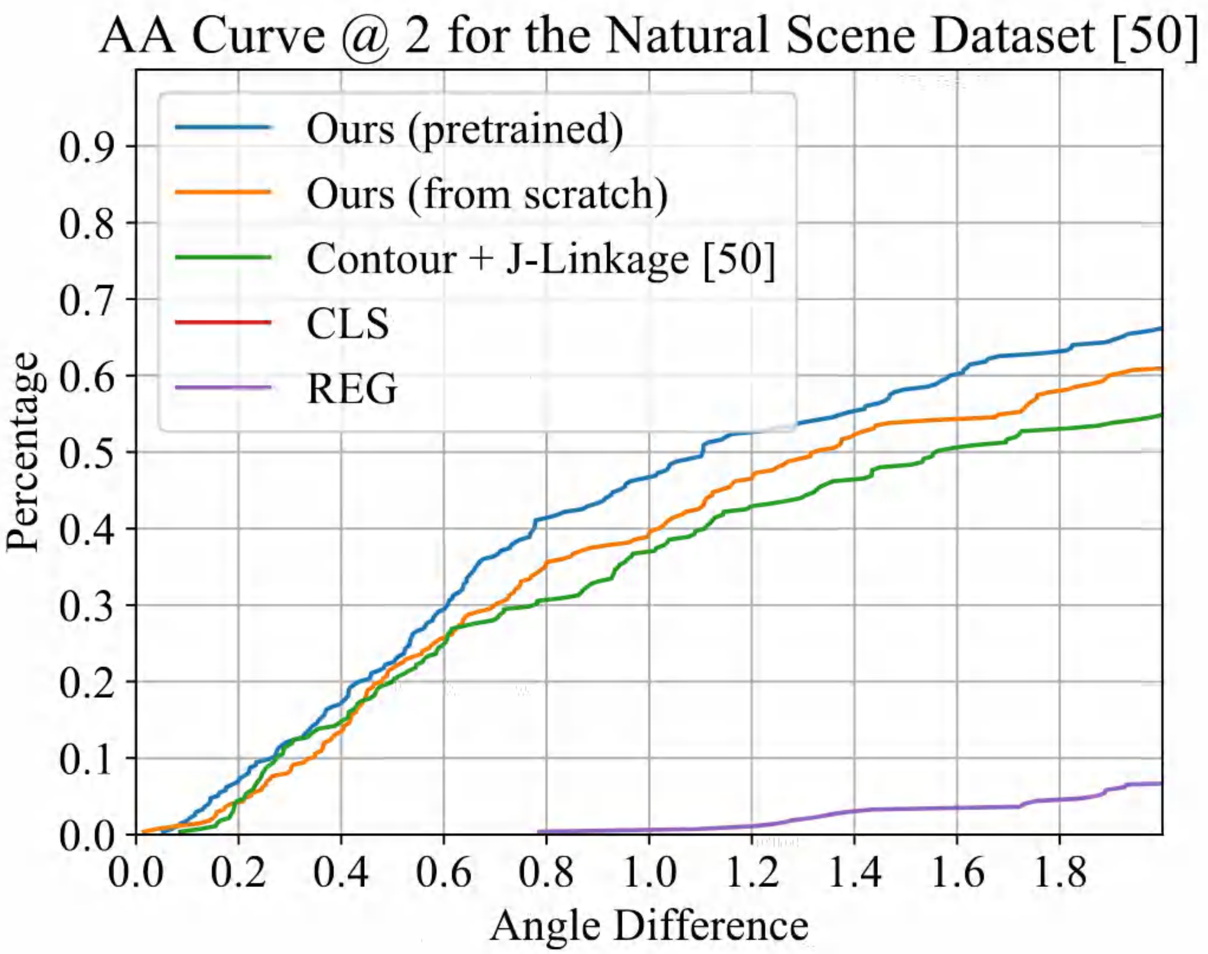


(b) Angle difference ranges from 0° to 6°.

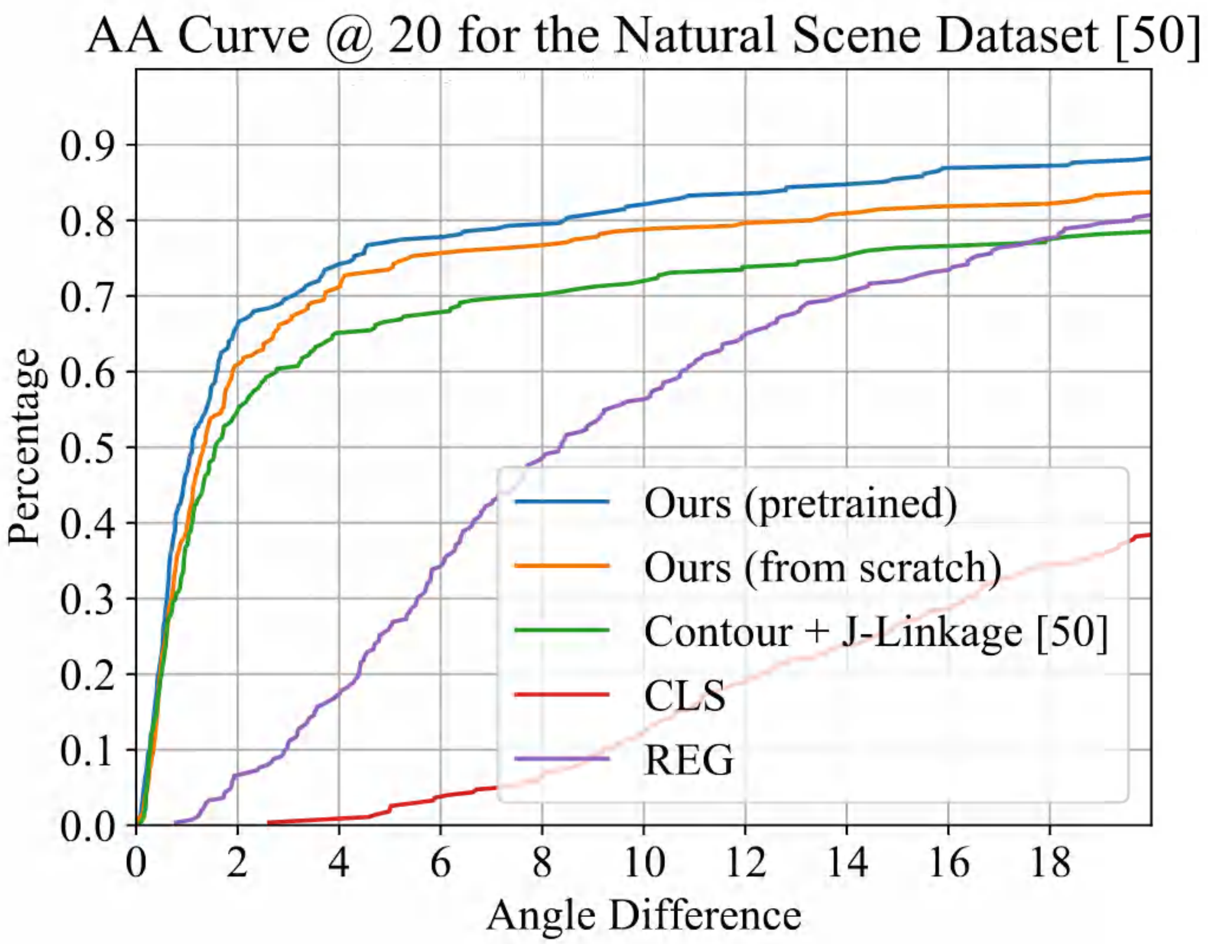
Natural Scene Dataset



Zhou, Zihan, Farshid Farhat, and James Z. Wang. "Detecting dominant vanishing points in natural scenes with application to composition-sensitive image retrieval." IEEE Transactions on Multimedia 19.12 (2017): 2651-2665.



(a) Angle difference ranges from 0° to 2°.



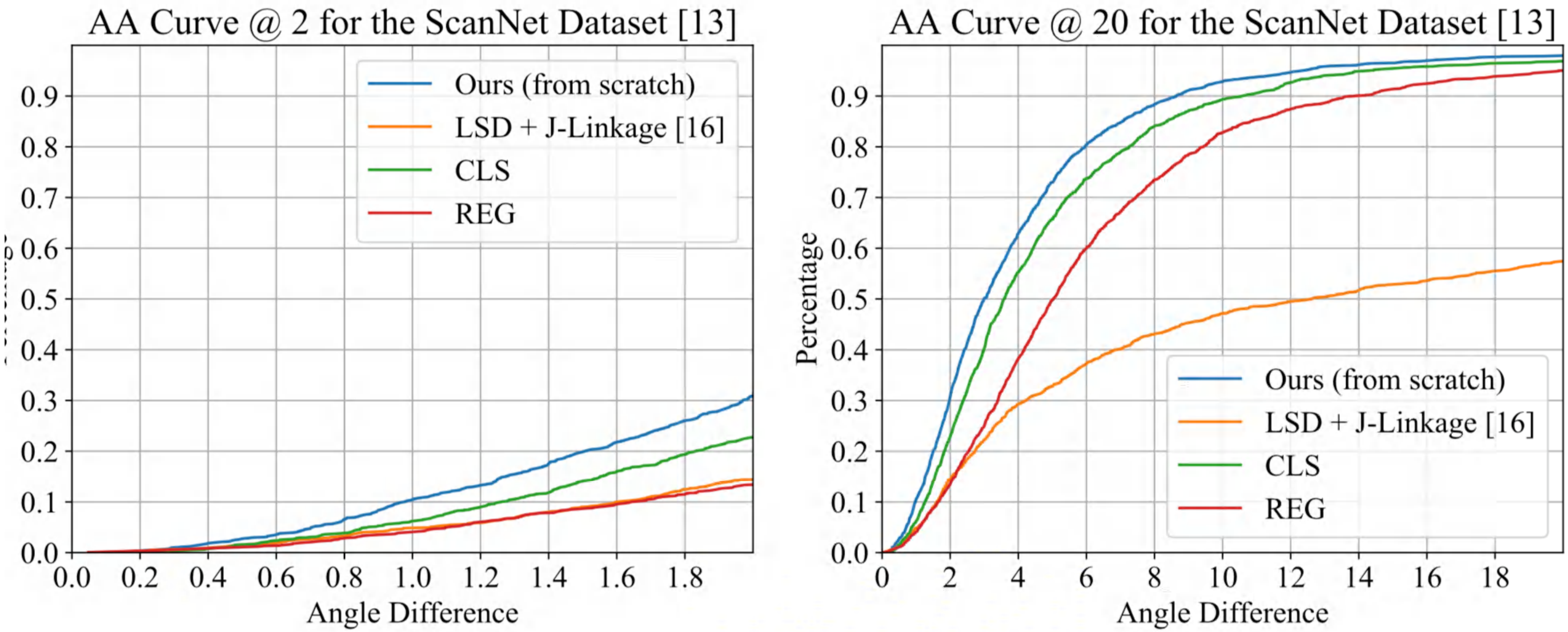
(b) Angle difference ranges from 0° to 20°.

ScanNet Dataset

ScanNet Images

Ground Truth Vanishing Points

Dai, Angela, et al. "Scannet: Richly-annotated 3d reconstructions of indoor scenes." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.



AA Curve @ 2 for the ScanNet Dataset [13]
Ours (from scratch)
LSD + J-Linkage [16]
CLS
REG
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
Angle Difference
AA Curve @ 20 for the ScanNet Dataset [13]
Percentage
0
2
4
6
8
10
12
14
16
18
Angle Difference

Summary

Geometric Methods	Learning-based Methods
• Detect structures by grouping local image cues in a bottom-up fashion	• Detect structures by learning from information provided by humans
• Sequential processing • Errors accumulate over stages	• End-to-end training • Backpropagate errors to all units in the network
• Method design driven by geometric principles	• Method design (sometimes) driven by trial-and-error

Thank you!

Main References:

- CVPR’18: **Learning to Parse Wireframes in Images of Man-Made Environments**, Kun Huang, Yifan Wang, Zihan Zhou, Tianjiao Ding, Shenghua Gao, Yi Ma.

- CVPR’19: **PPGNet: Learning Point-Pair Graph for Line Segment Detection**, Ziheng Zhang*, Zhengxin Li*, Ning Bi, Jia Zheng, Jinlei Wang, Kun Huang, Weixin Luo, Yanyu Xu, Shenghua Gao

- ICCV’19a: **Learning to Reconstruct 3D Manhattan Wireframes from A Single Image**, Yichao Zhou, Haozhi Qi, Simon Zhai, Qi Sun, Zhili Chen, Li-Yi Wei, Yi Ma.

- ICCV’19b: **End-to-End Wireframe Parsing**, Yichao Zhou, Haozhi Qi, and Yi Ma.

- NeurIPS’19: **NeurVPS: Neural Vanishing Point Scanner via Conic Convolution**, Yichao Zhou, Haozhi Qi, Jingwei Huang, Yi Ma.