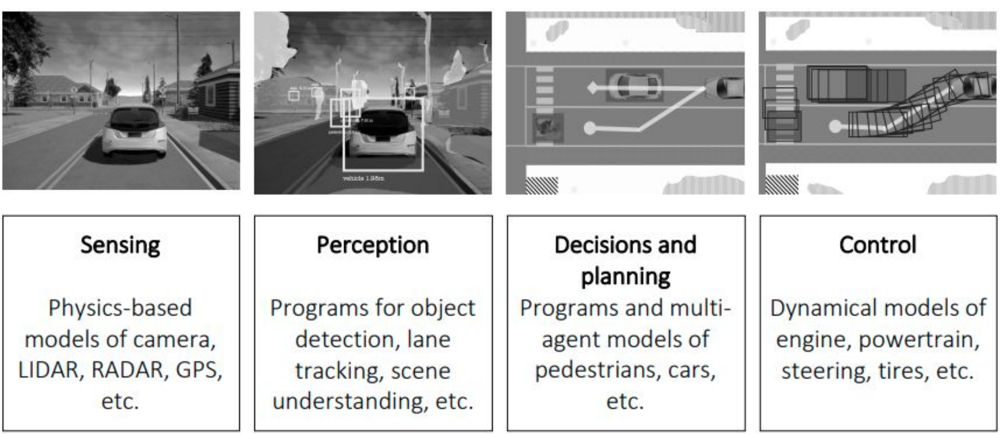
Using LiDAR for Enhanced Perception

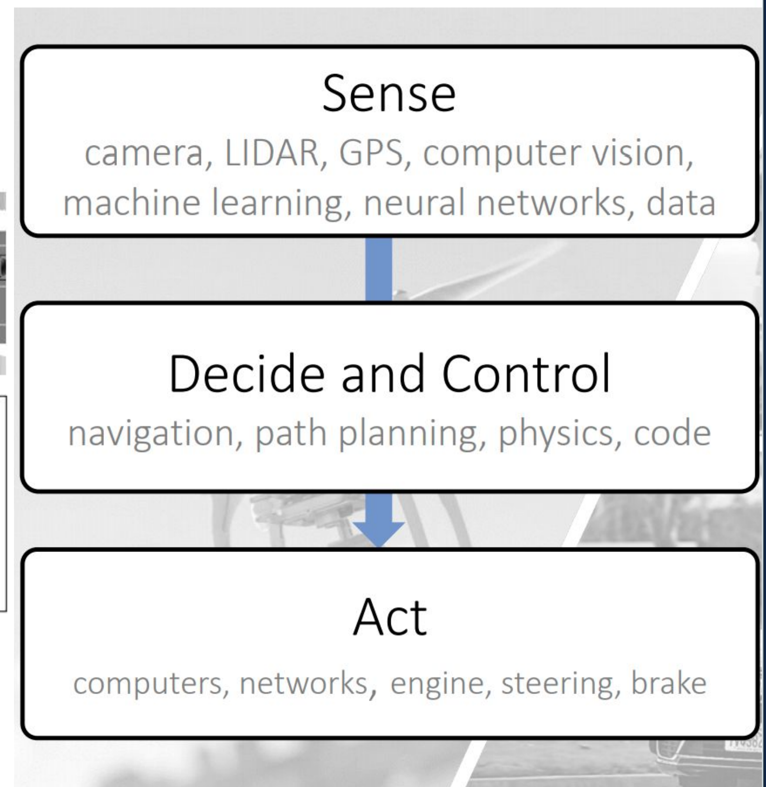
By Eric Zhang and Adit Umakanth (Group 14)

ILLINOIS

Perception

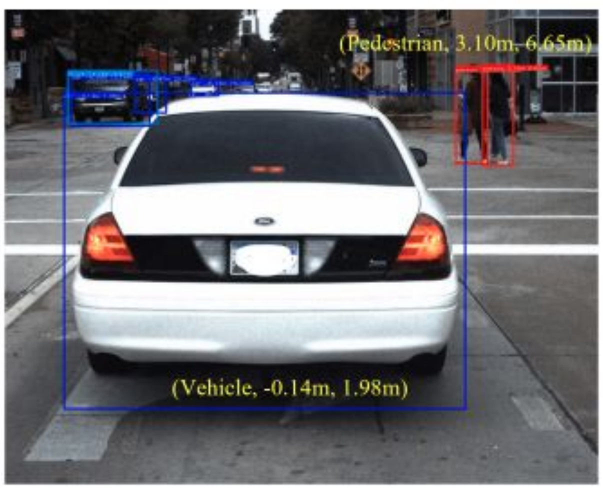
- One of the four cornerstones
- Heavily relied upon by decision and planning systems





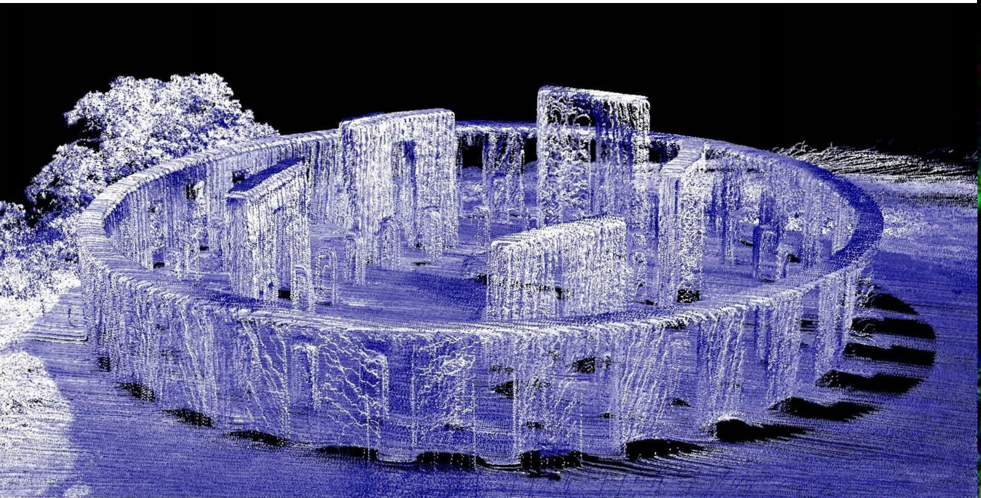
Perception on GEM

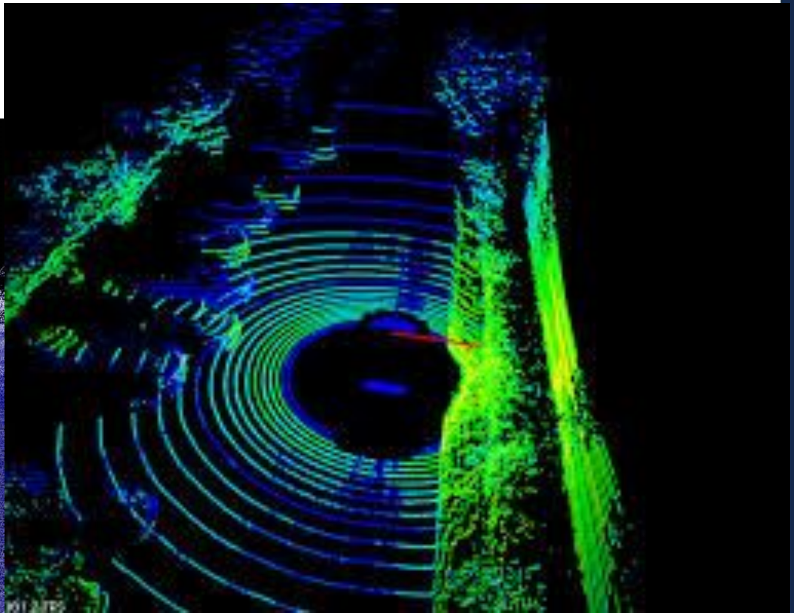
- Only uses camera
- Assumes width of pedestrian (0.47m) when calculating distance
- Processes objects from a few classes
  - Pedestrians, vehicles, traffic signs, etc.



Why use LiDAR?

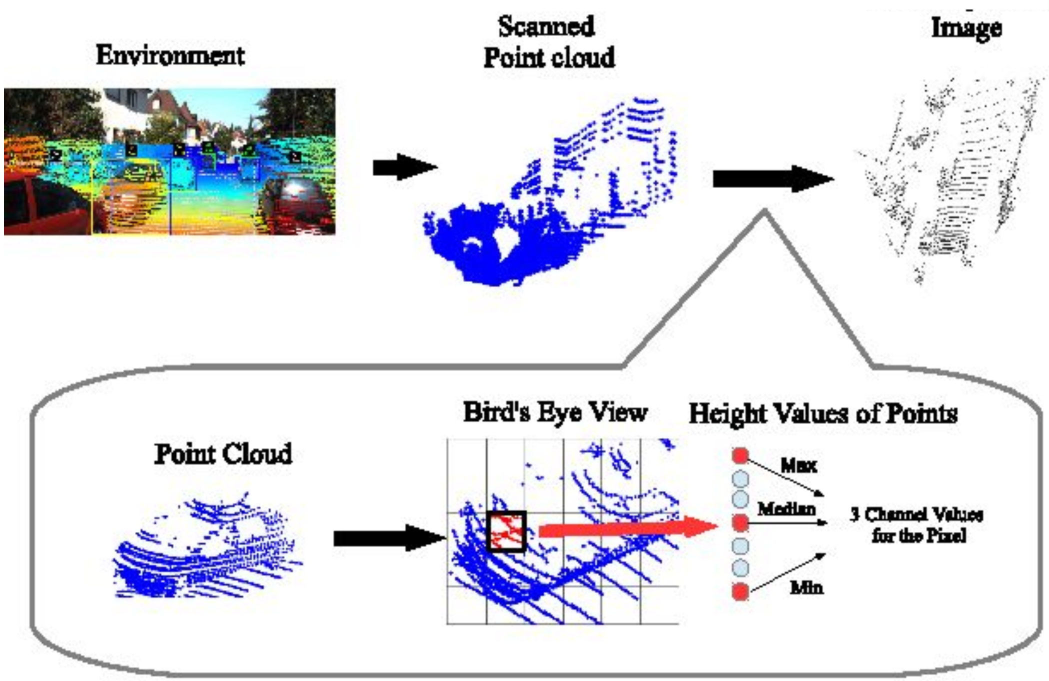
- Cameras are heavily constrained by lighting
- Distance measurement built into the sensing
- Can detect neighboring objects that the camera isn’t trained to detect
  - Strangely-shaped vehicles, barriers, etc.





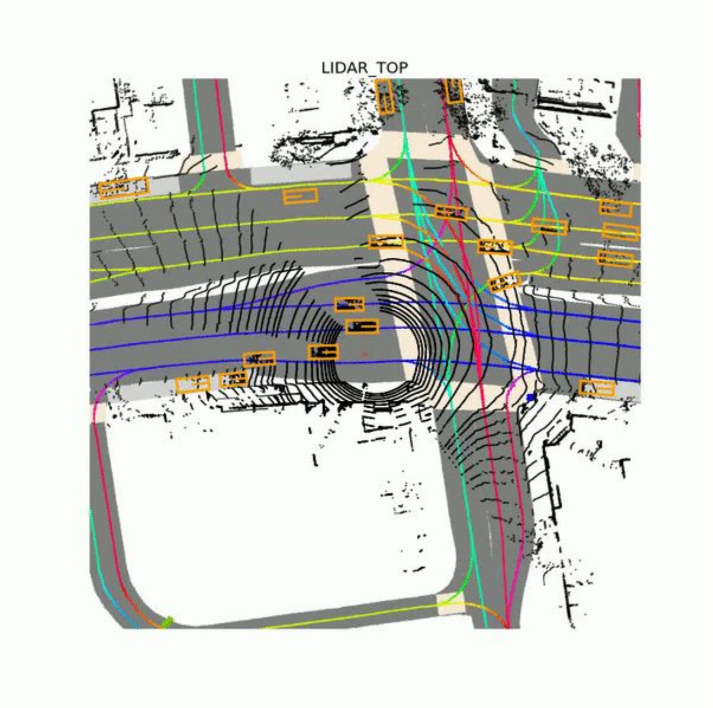
Our Goal

- Detect neighboring objects using LiDAR
  - Speed estimation
- Don’t be limited to specific objects
- Reasonable computation time
  - Laptop GPU
  - 30 FPS



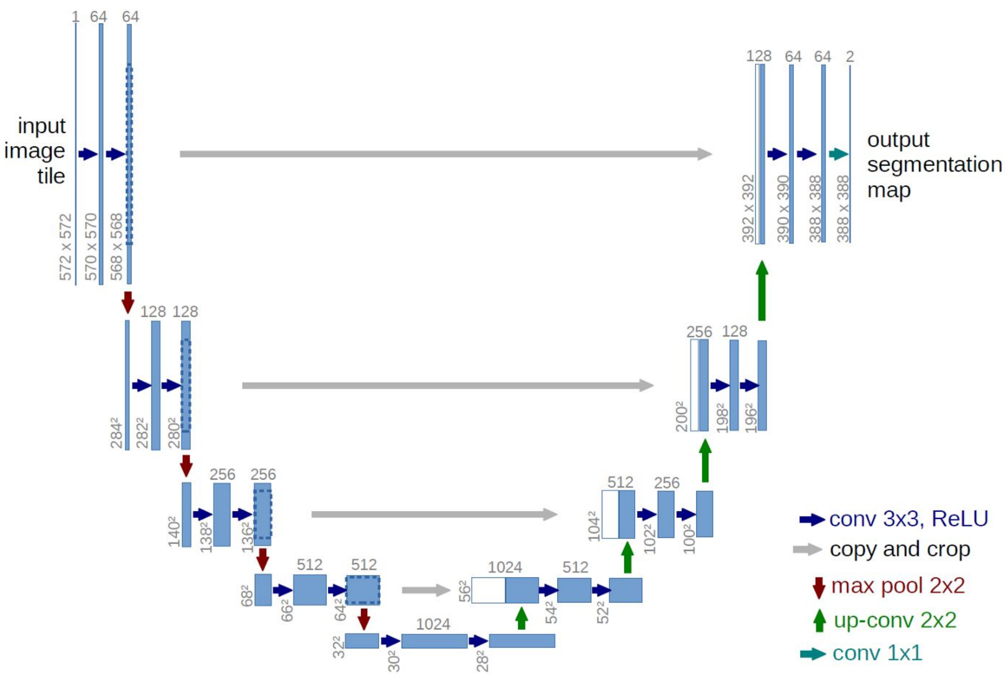
What We Used

- U-Net
  - Fully Convolutional Network for biomedical image segmentation
- Lyft Level 5 Dataset
  - 50,000+ annotated frames
  - LiDAR, Camera



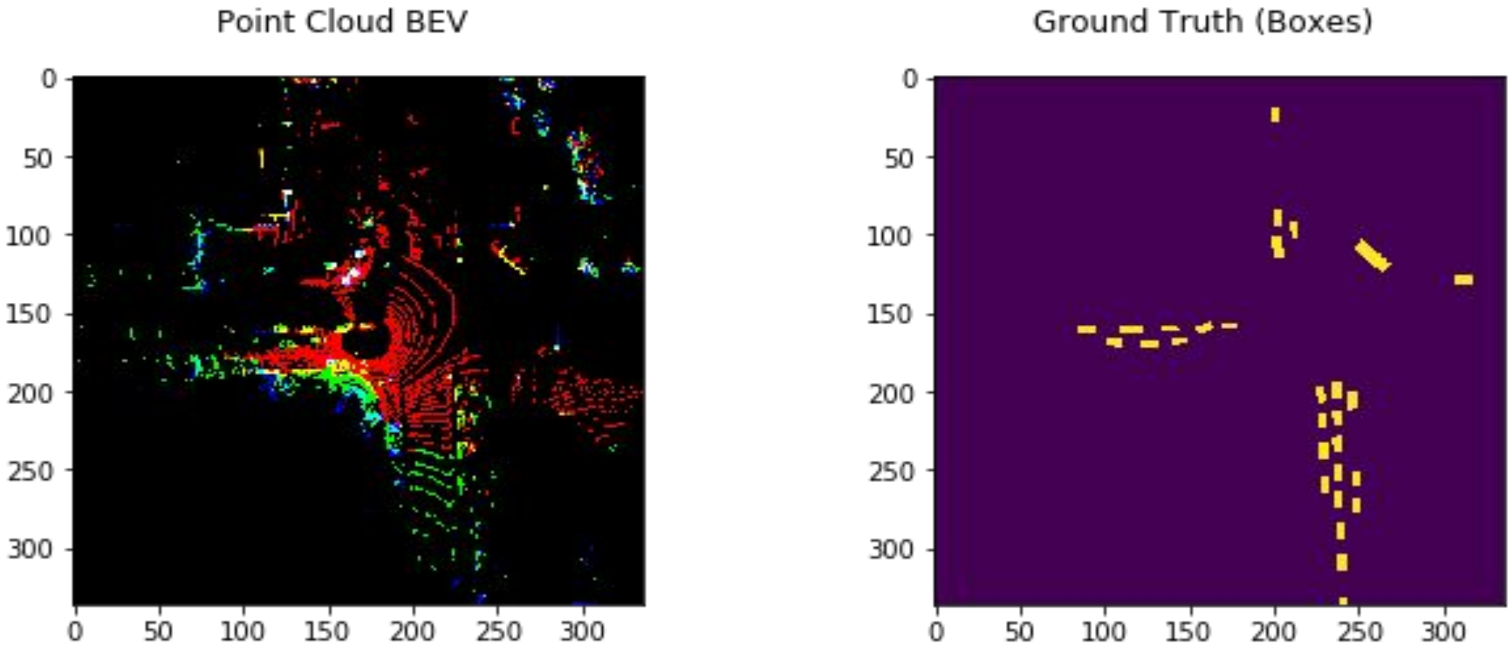
U-Net

- Semantic segmentation
- Image -> Segmentation Map
- “What” and “Where”



Pre Processing

- Bird's eye view
  - Centered around vehicle
- Ground truth
  - Draw boxes

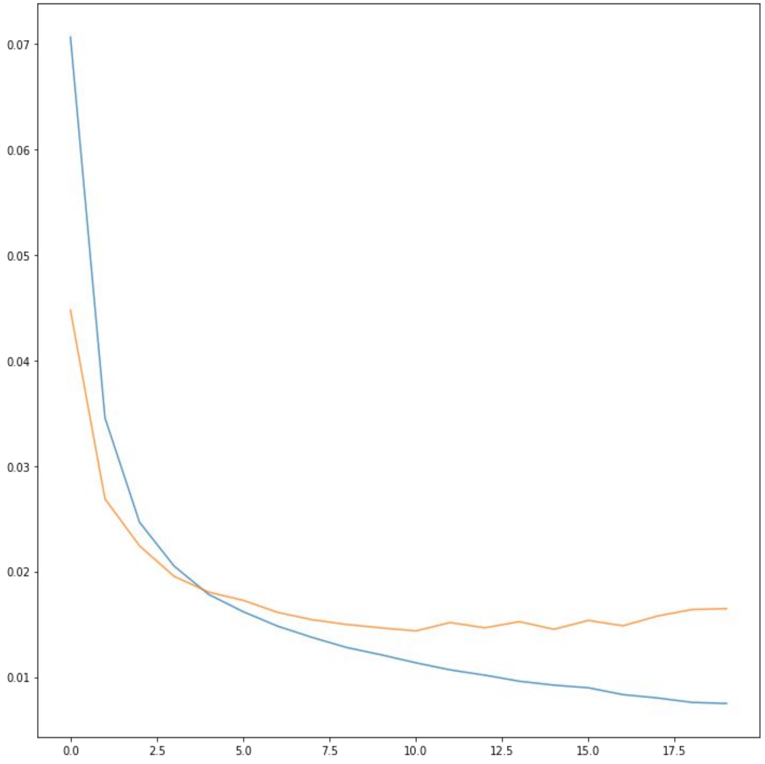


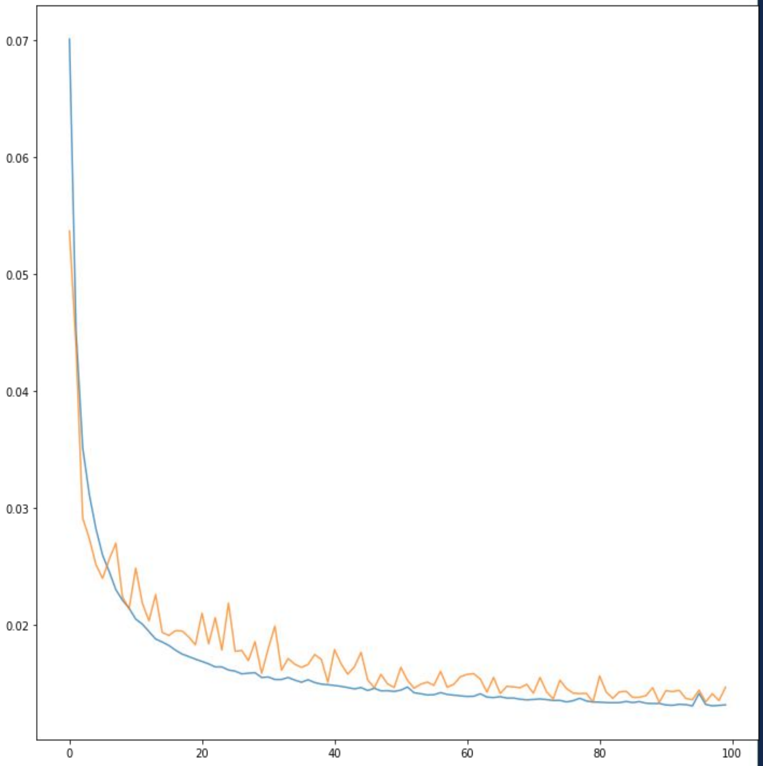
Implementation

- 80/20 Train Validation Split
- U-Net (modified)
  - Slightly less deep
- Training
  - Adam for optimization
  - Cross Entropy Loss
  - Overfitting

Dealing with overfitting

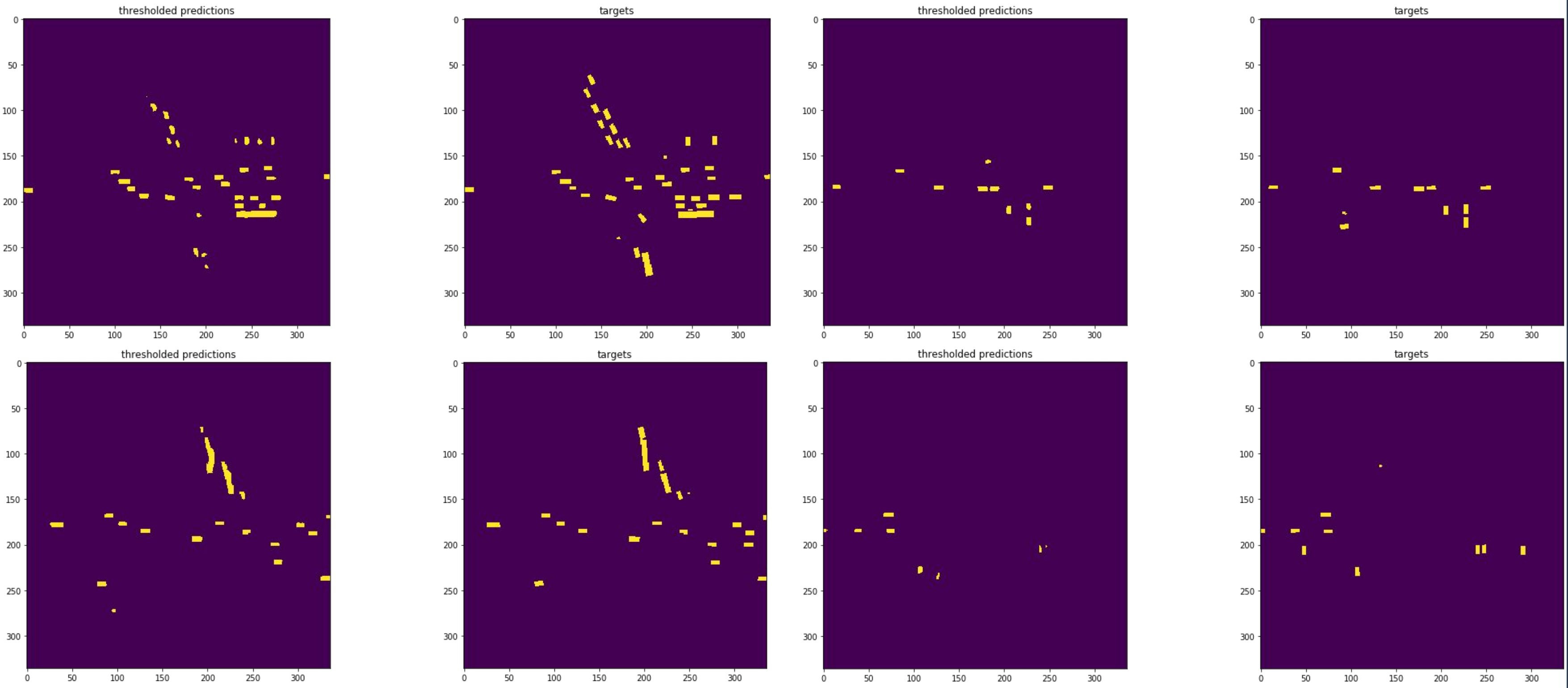
- Training loss: Blue
- Validation loss: Orange
- Added Dropout Layers





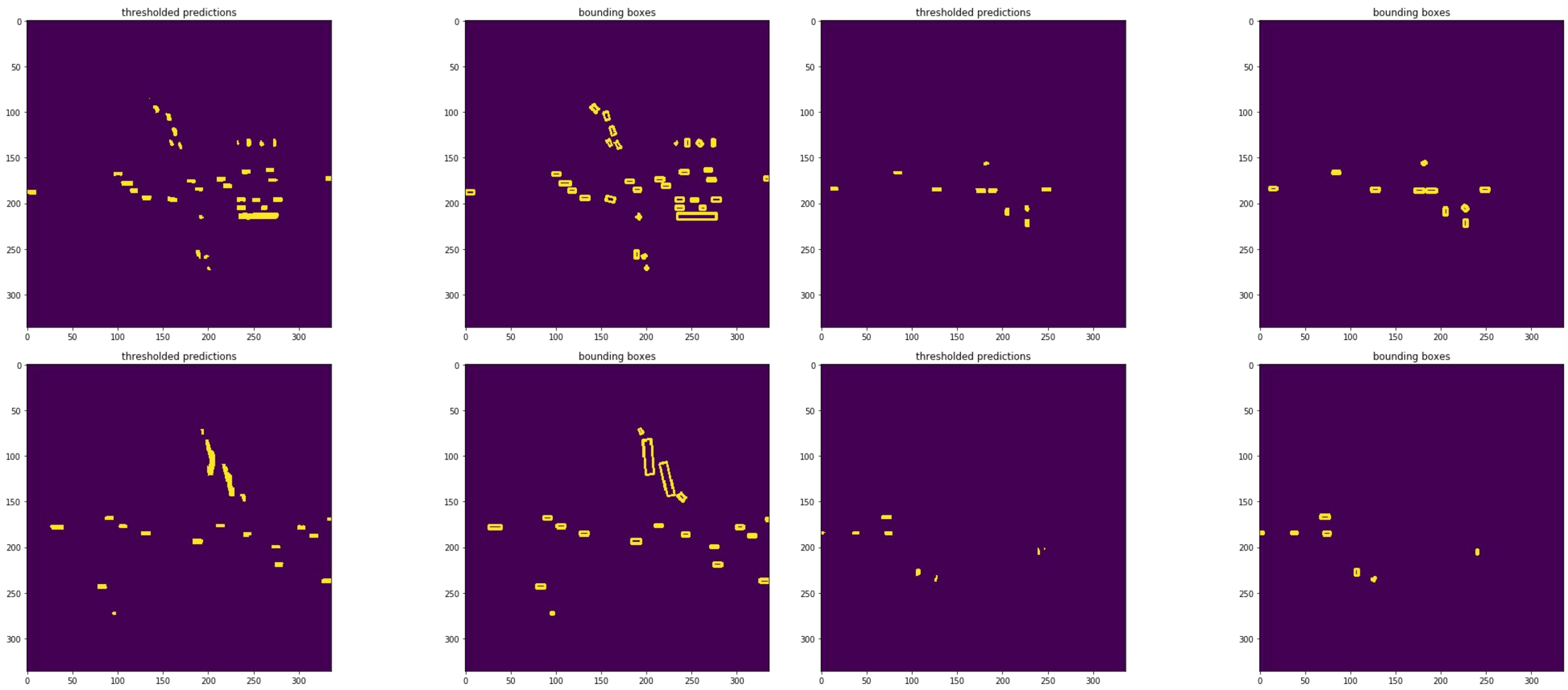
Results

- Threshold probability -> binary image



Post Processing

- Binary image -> List of rectangles
  - Find contours
  - Contours -> (rotated) bounding rectangle

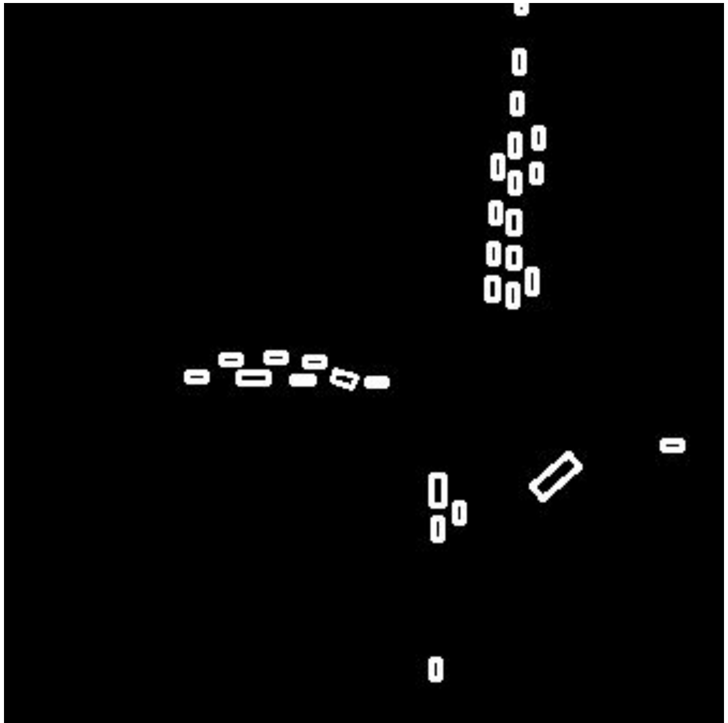


Speed Estimation

- Track relative speeds of nearby objects
- Accurately predict collisions
- Safe lane-switching

Speed Estimation Algorithm

- Identify objects over consecutive frames
- Store changes in position
- Summarize average speed

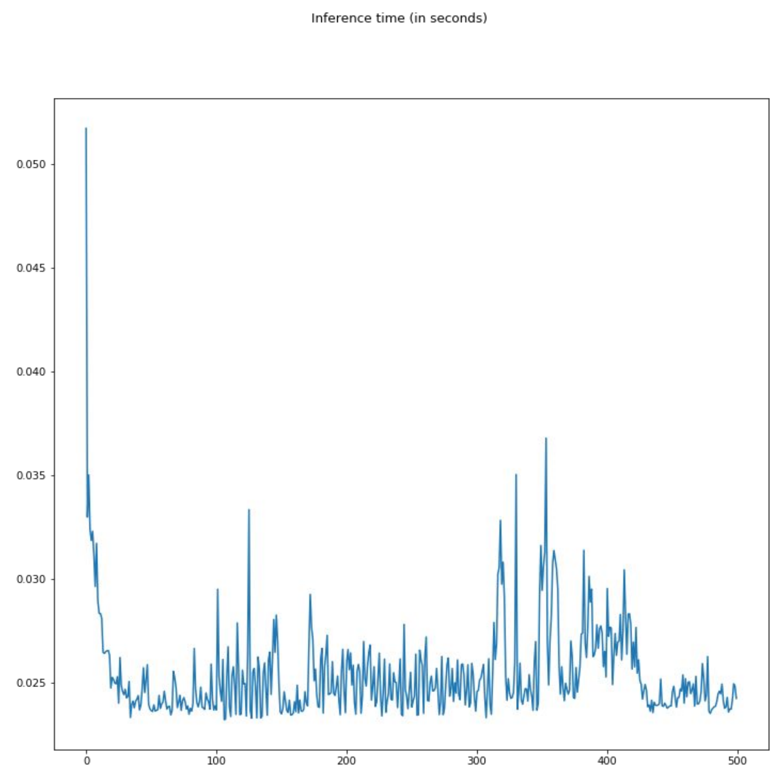


Lack of Ground Truth

- Currently impossible to test algorithm accuracy
- Can run a test with cars whose exact speeds are known

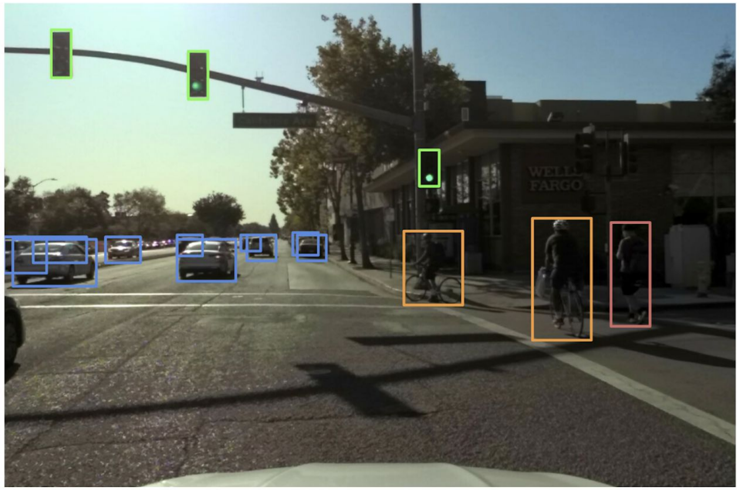
Cost

- 0.026 seconds per frame for inference
  - 38 FPS
- GTX 1050 Ti
  - 4 GB Memory
- No additional hardware



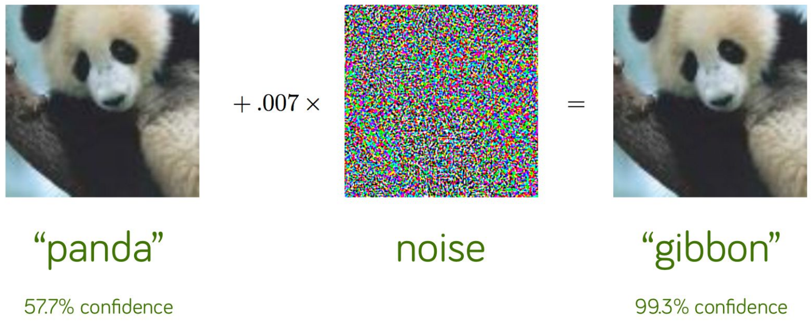
Future Work

- Fuse with other sensors
  - Camera, Radar
- Collision Prediction



Potential Downsides

- Neural Nets are a “black box”
  - Verification
  - Adversarial attacks
- LiDARs are expensive
- Distinguishing between objects



Sources

1. U-Net: Convolutional Networks for Biomedical Image Segmentation
    a. https://arxiv.org/abs/1505.04597
2. Online monitoring for safe pedestrian-vehicle interactions
    a. https://arxiv.org/abs/1910.05599
3. Lyft Dataset: https://www.lyft.com/level5/data
4. PyTorch: https://pytorch.org/
5. OpenCV: https://opencv.org/

Questions?

- Thanks to Prof Mitra and Prof Kim along with Ted and Yangge for this awesome class!

Project Contributions

- Adit
  - Training + Tuning Model
  - Speed detection
- Eric
  - Training + Tuning Model
  - Data Preprocessing/Postprocessing