

Practical Course

Simulator Based Autonomous Driving in Crowded City

Team 8 - Aristotelis Tsoutsanis and Pascal Henschke

Motivation and Introduction

- The rise of autonomous driving in urban environments (Waymo)
- Navigational challenges in crowded city streets
- Goal: Develop an End-to-End simulator-based solution using vision
- Contribution: Integration of End-to-End CNN, YOLO model for object detection, and traffic light classification
- Potential Impact: Enhanced safety and efficiency in urban autonomous driving scenarios

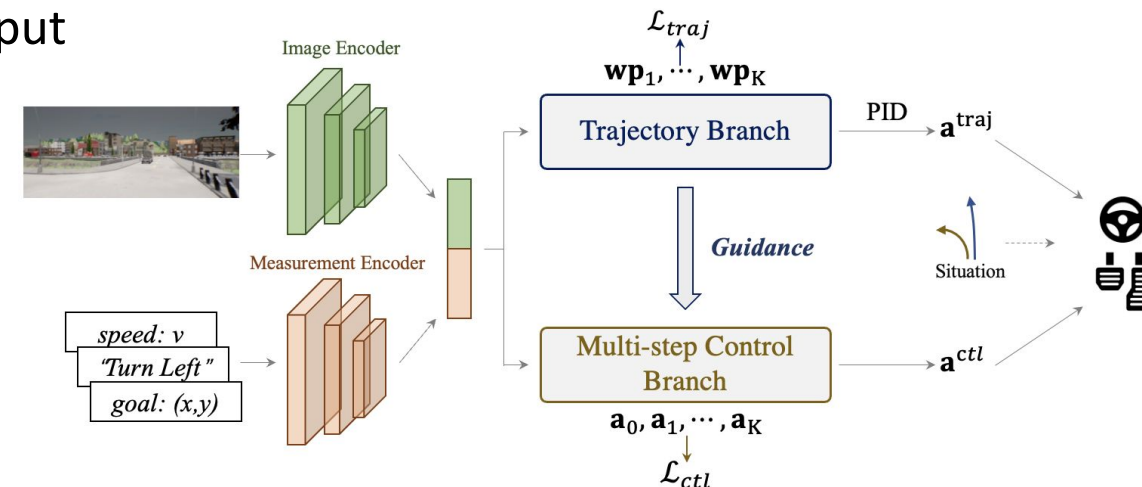
Related Research

ReasonNet, Shao et al. [CVPR 2023]

- 1st in CARLA benchmark
- LiDAR + Multi-View cameras as input + privileged information for training (e.g., BEV)
- Transformer-based architecture that includes temporal information and a global reasoning module

TCP, Wu et al. [NeurIPS 2022]

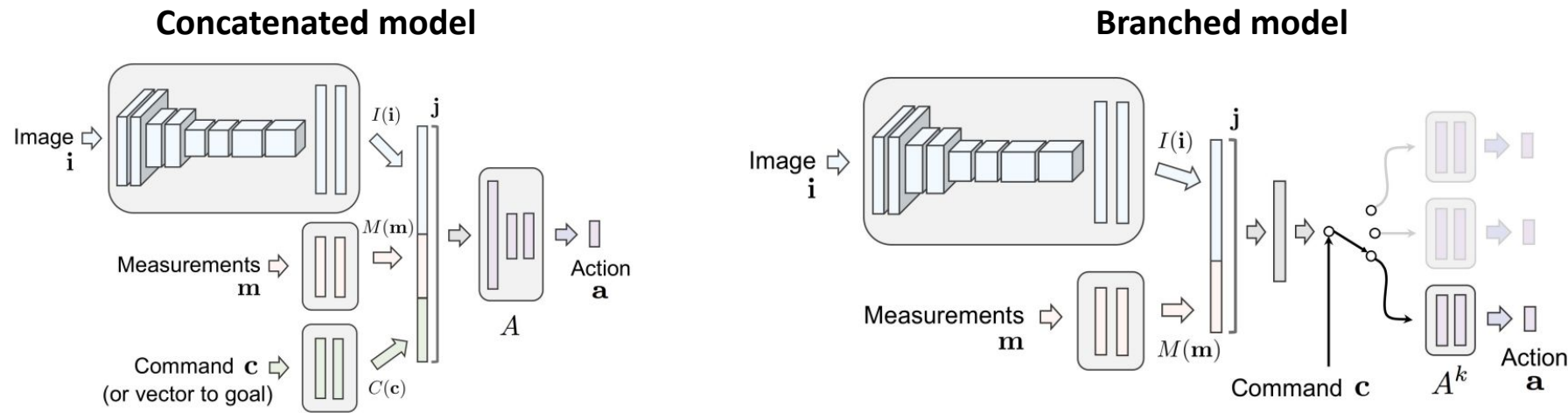
- 3rd in CARLA benchmark
- BUT: uses only front camera input (and no privileged information for training)
- combine trajectory planning and direct control output
- both branches use GRUs to capture temporal info



Related Research

CIL and CILRS, Codevilla et al. [ICRA 2018 and ICCV 2019]

- High-level navigation input is needed (i.e, “go left/straight/right” and/or target coords)
- Build two simple models that incorporate this information
- CILRS also predicts speed from latent image, which has a regularizing effect

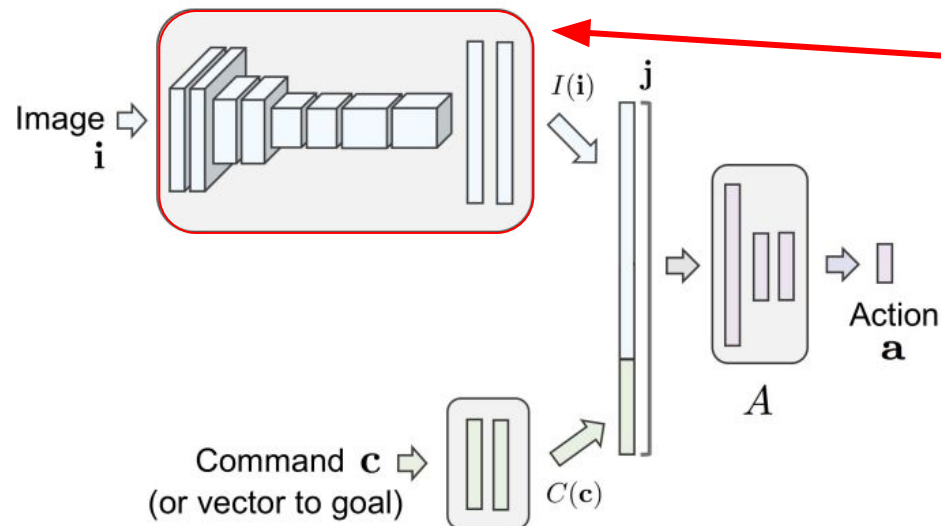


Our Methodology - Steering Model

We have:

- No access to privileged information
- No access to automated expert driver (i.e. need to drive manually)
- Limited resources

=> Use simple conditional architecture based on the concatenated CIL model.



- Image encoder: simple 6-layer convolutional network with ReLUs, BatchNorms, Dropout, and L1 Loss (have also tried ResNets)
- We only predict steering angle so far
 - Braking is hard-coded based on object detection

Our Methodology - Data

- Manually drive the car using the mouse
- Need navigation command labels
 - Do separate runs for different commands (left, right, straight)
- Do separate 'recovery runs' (to teach the model recovery)
- Try to balance data (i.e., reduce near-zero angle observations)
- Apply simple augmentations (cropping sky, blur, brightness, color jitter)

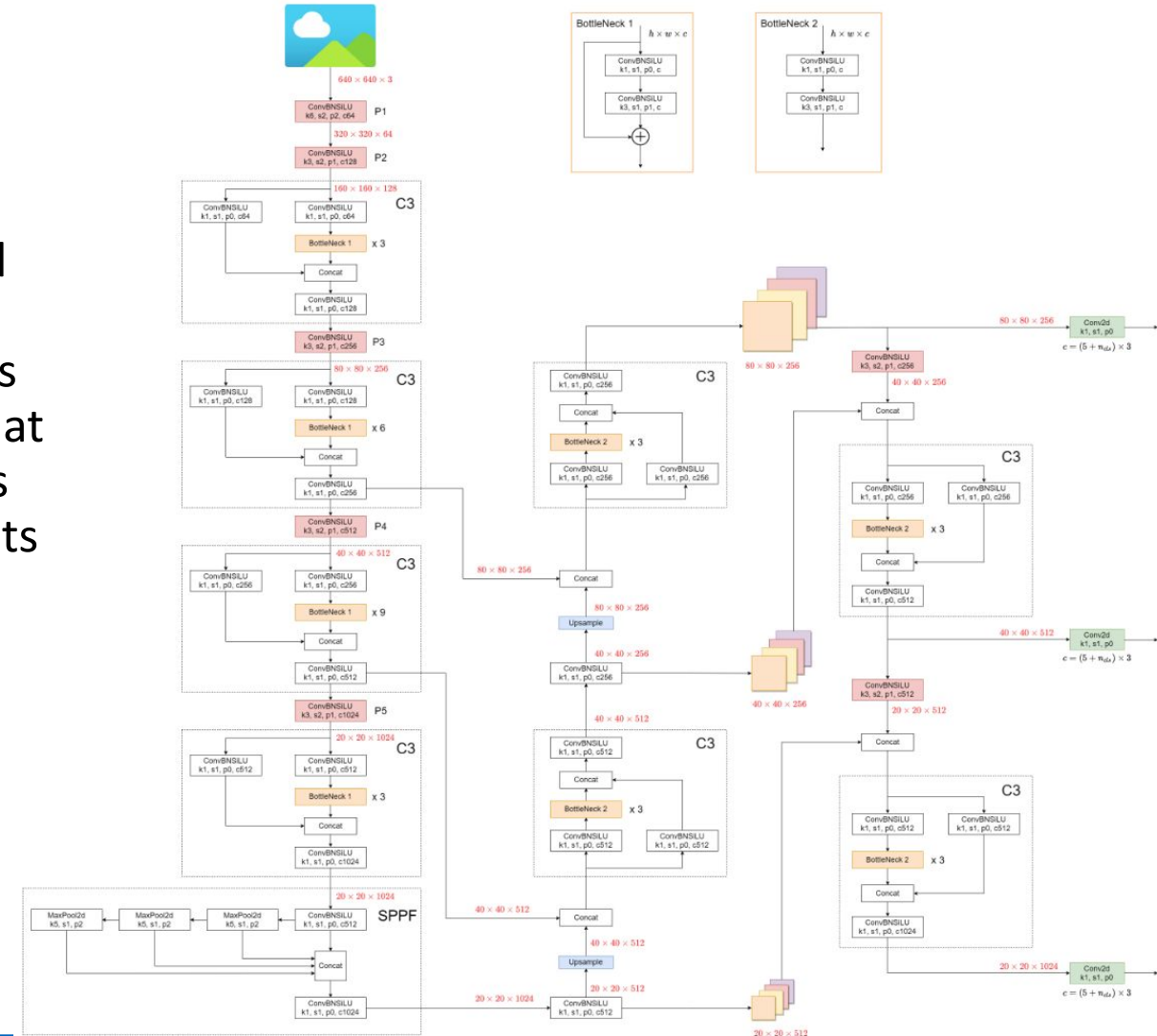
Next steps:

- Accurate data balancing across whole steering angle spectrum
- Use side cameras with a corrective factor (to teach recovery)

Our Methodology - Object Detection

YOLOv5

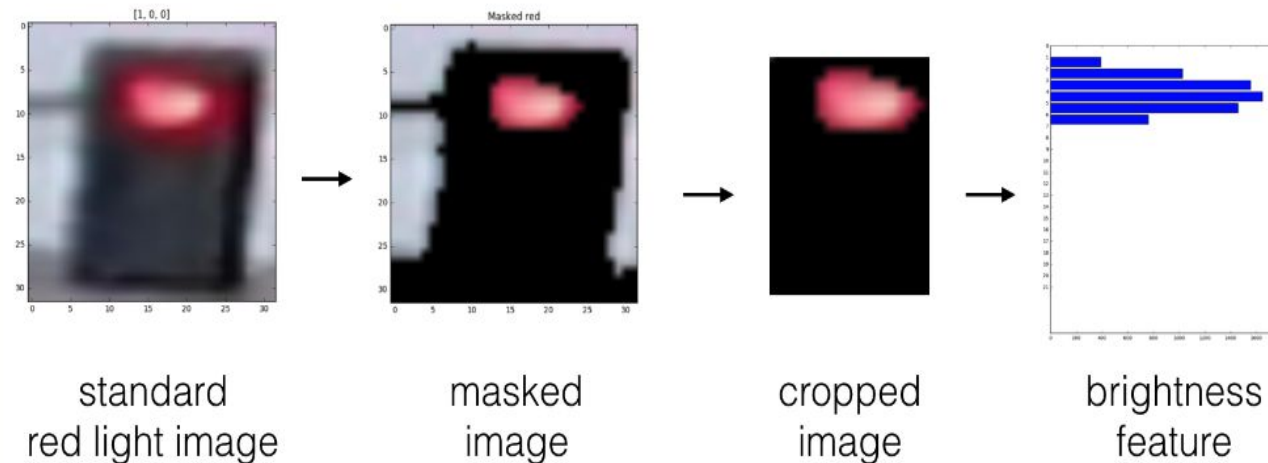
- Uses CSP-Darknet53 as its backbone
- SPPF (another variant of the SPP block) and PANet in the model neck
- YOLOv5 uses the same head as YOLOv3. It is composed from three convolution layers that predicts the location of the bounding boxes (x,y,height,width), the scores and the objects classes.



Our Methodology - Traffic Light Classifier

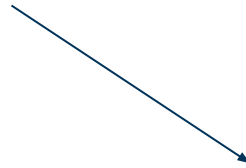
Classical Approach Algorithm based on hue space

- Standardize the image
- Create a mask image for every color
- Slice the masked images
- Classify the color based on the brightness value



Our Methodology


We currently use a **brake area**

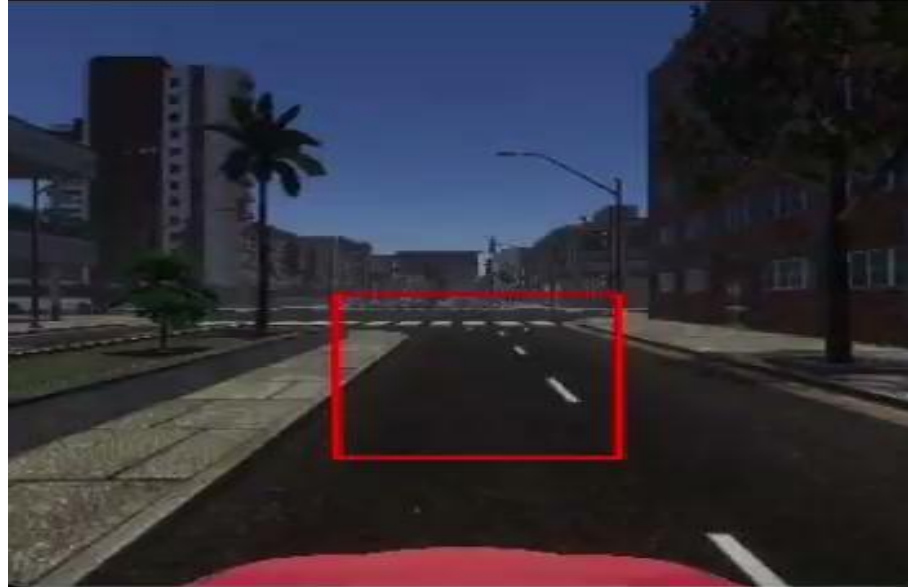


Planning Rules:

- Don't hit cars - If there is a car in the brake area with $\text{IoU} \geq 0.1$ then the car brakes
- Stop at red traffic lights - If there is a red traffic light in front then the car brakes
- Force the car to drive with a low speed- if the car exceeds 12 km/h then we set throttle to zero

First Results

- Steering model L1 validation loss: 0.028
- What the car  - Object Detection Pipeline in an intersection and current issues



Discussion

Compared to SoTA:

- No capturing of temporal information (e.g. Transformers, 3D-Networks or RNNs)
 - Path and trajectory planning not available
 - Bird's-eye view is not offered
 - No model-based longitudinal output, just steering angle and detection-based stopping
-
- Additional multi-task learning likely to be beneficial

Outlook

Next steps:

- Improve the data and the pre-processing steps (use side images, better balancing)
- Use branched conditional model instead of concatenated
- Improve object-detection-based braking rules (incorporate size of bounding box)
- Apply instance segmentation instead of object detection (produces more meaningful inference of images and gives richer output)
- Add the car's speed as an output of our model
- Navigation

Thank you!

Reference List

1. Shao, Hao, et al. "ReasonNet: End-to-End Driving with Temporal and Global Reasoning." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023.
2. Wu, Penghao, et al. "Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline." Advances in Neural Information Processing Systems 35 (2022): 6119-6132.
3. J. L. Binangkit and D. H. Widyantoro, "Increasing accuracy of traffic light color detection and recognition using machine learning," 2016 10th International Conference on Telecommunication Systems Services and Applications (TSSA), Denpasar, Indonesia, 2016, pp. 1-5, doi: 10.1109/TSSA.2016.7871074.
4. Codevilla, Felipe, et al. "End-to-end driving via conditional imitation learning." 2018 IEEE international conference on robotics and automation (ICRA). IEEE, 2018.
5. Codevilla, Felipe, et al. "Exploring the limitations of behavior cloning for autonomous driving." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2019.
6. Codevilla, Felipe, et al. "On offline evaluation of vision-based driving models." Proceedings of the European Conference on Computer Vision (ECCV). 2018.
7. Jocher, G. (2020). YOLOv5 by Ultralytics (Version 7.0) [Computer software]. <https://doi.org/10.5281/zenodo.3908559>
8. Liu, Fei, Zihao Lu, and Xianke Lin. "Vision-Based Environmental Perception for Autonomous Driving." arXiv preprint arXiv:2212.11453 (2022).