

# Team07 Simulation-Based Autonomous Driving In Crowded City Final Report

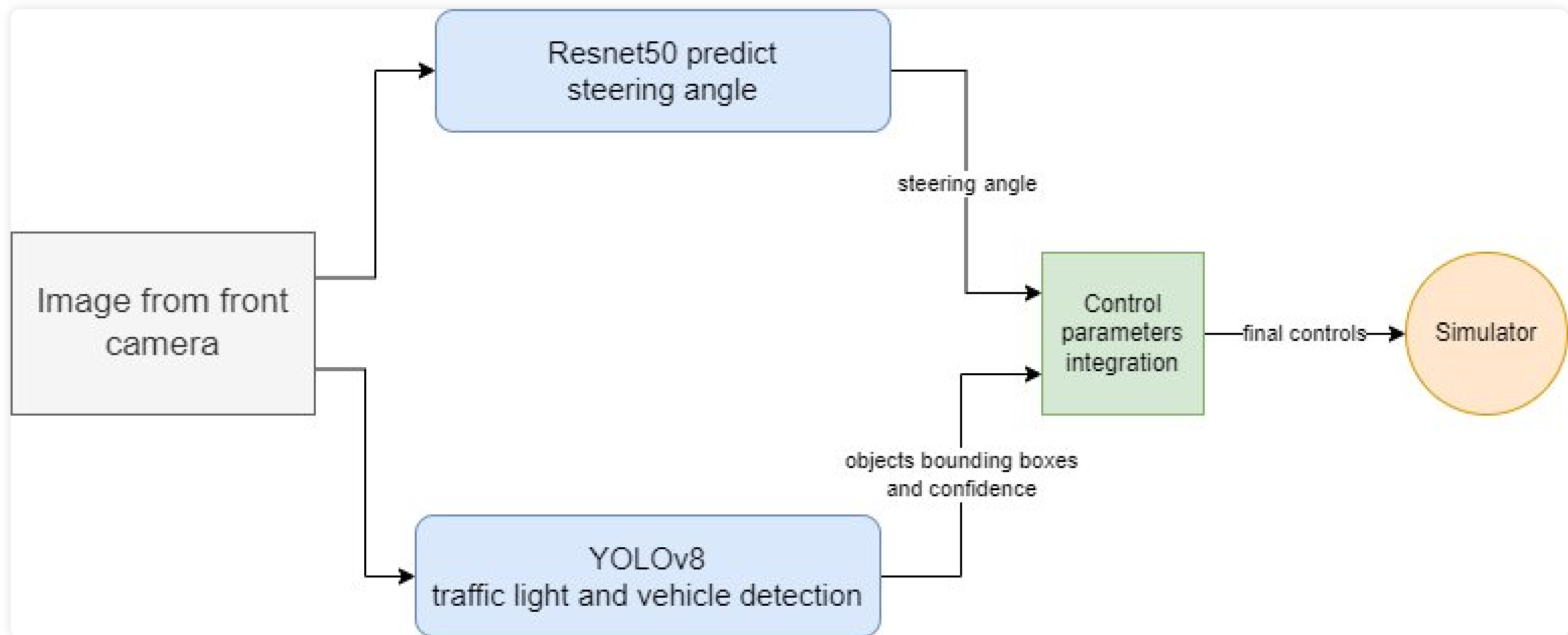
Shihong Zhang  
2024-03-08

# CONTENTS

- System Architecture
- Data Preparation
- Model Architecture
- Training Process
- YOLOv8 Model
- Synthetical Inference
- Conclusions and Future Direction

# System Architecture

- end to end autonomous driving project structure



# Data Preparation

- Data unbalancing: the amount of image data with steering\_angle=0 is significantly higher than image with steering\_angle!=0.
- write a script to solve data unbalancing

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split

# 步骤1: 读取数据
column_names = ['Image_path', 'throttle', 'brake', 'steering', 'velocity']
data = pd.read_csv(filepath_or_buffer='VehicleData.txt', sep=' ', names=column_names, header=None)
print(data.shape)

# 步骤2: 分离出steering angle不为0的数据和为0的数据
non_zero_steering = data[data['steering'] != 0]
zero_steering = data[data['steering'] == 0]

# 步骤3: 从steering angle为0的数据中随机选择同等数量的项
zero_steering_sample = zero_steering.sample(n=len(non_zero_steering))

# 步骤4: 合并数据
final_data = pd.concat([non_zero_steering, zero_steering_sample])

# 步骤5: 分为训练集和验证集
train_set, test_set = train_test_split(*arrays: final_data, test_size=0.2) # 以80%训练集, 20%验证集的比例分割

# 可选: 保存结果到文件
train_set.to_csv('train_set.csv', index=False)
test_set.to_csv('test_set.csv', index=False)

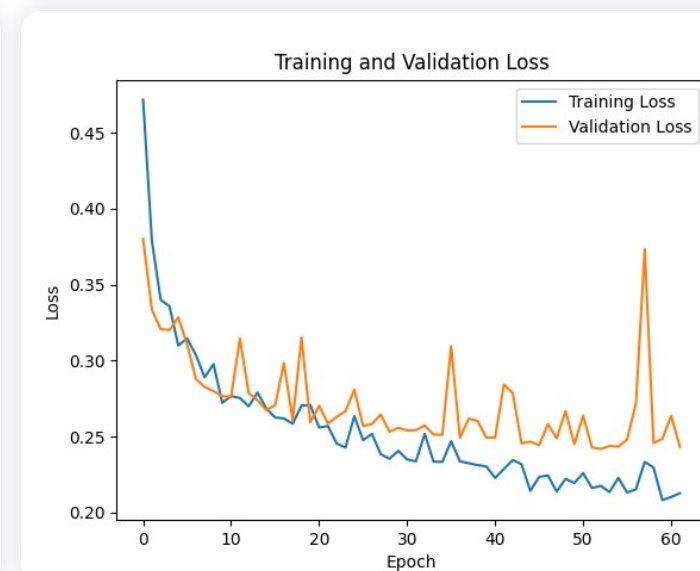
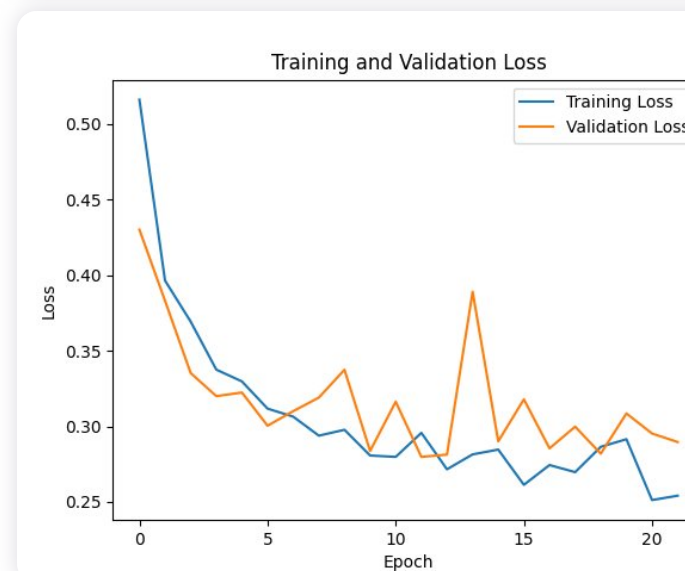
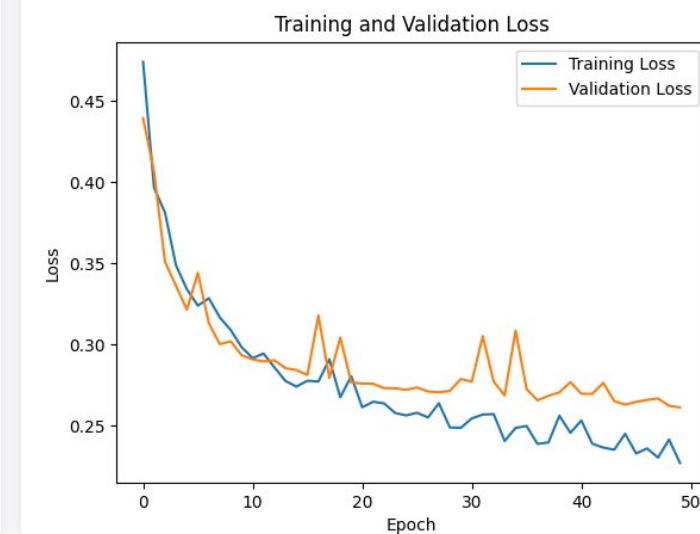
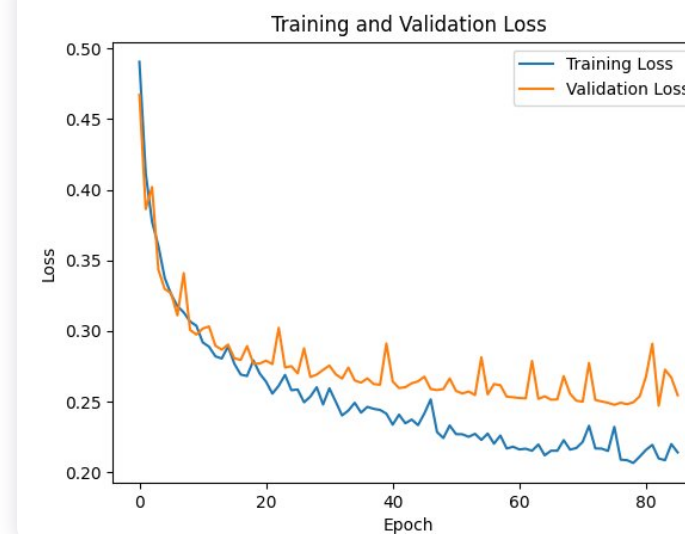
print("训练集和验证集已生成并保存。")
```

# Model Architecture

- Resnet50
- Freeze feature extraction layers
- change fc layer's output channel to 1

# Training Process

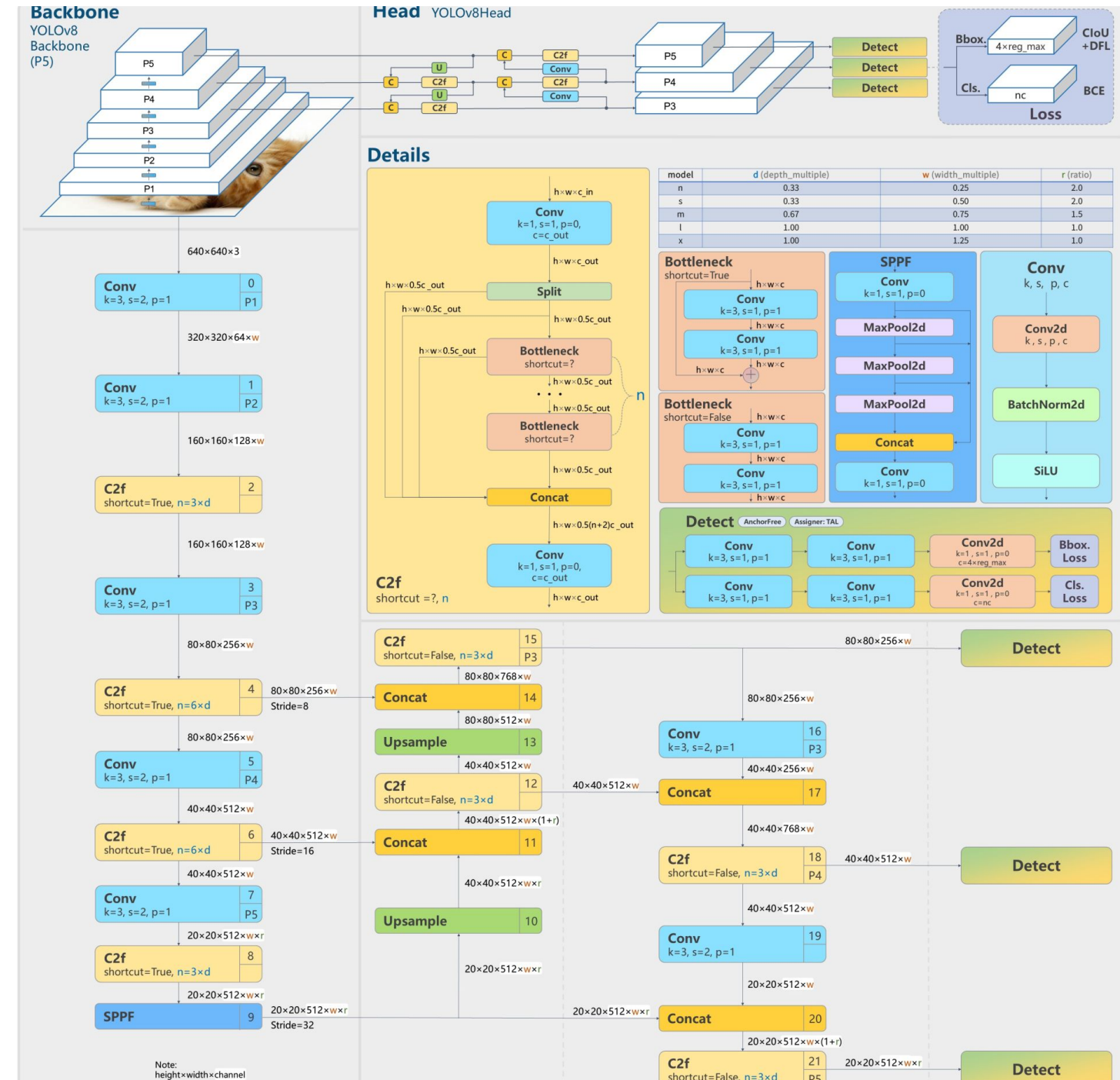
- MSE loss
- Adam Optimizer
- batch size :128
- lr: 0.0005
- early stopping at epoch 85





# YOLOv8 Model

get pre-trained yolov8 directly  
from ultralytics



# Synthetical Inference

- combine the results from resnet and yolov8
- implement yolov8 for vehicle detection and output bbs if exist
- if brake=0, implement yolov8 for red light detection and output bbs if exist
- define a function to recognise if there is any vehicle in front of the car, input bbs from last step(2 thresholds)
- define a function to recognise if there is any red light in the img, input bbs and original img
- implement resnet to img and get steering\_angle as output
- define a logic to predict throttle, brake and steering\_angle.

The screenshot displays a PyCharm IDE with a project named 'autonomous\_driving\_shihong\_zhang'. The left sidebar shows the project structure, including folders for data, data\_preparation, figures, networks, test, train, and weights, along with files like .gitignore, CapturedImage199.jpg, drive.py, drive\_demo.py, drive\_origin.py, load\_weights.py, outputs\_to\_labels.py, README.md, requirements.txt, and Resnet50\_cifar10.py. The main editor shows the code in 'drive\_demo.py', which includes logic for vehicle detection, red light detection, and steering angle prediction. The code uses YOLOv8 for vehicle and traffic light detection and a ResNet50 model for steering angle prediction. The execution output at the bottom shows the following results:

```
0: 288x640 1 truck, 82.0ms
Speed: 4.0ms preprocess, 82.0ms inference, 110.5ms postprocess per image at shape (1, 3, 288, 640)
0 1 tensor([[-0.1190]], device='cuda:0', grad_fn=<TanhBackward0>)
```

The process finished with exit code 0.



# Conclusions and Future Direction

- this project has achieved a base function of autonomous driving
- real world is much more sophisticated
- only vehicle and traffic light were taken into consideration
- In next step, more complicated circumstances like pedestrian, weather condition should be added



**THE END**  
**THANKS**