RUTGERS
WINLAB | Wireless Information Network Laboratory

Virtual ROS Based Self Driving Car Model

Victor Abril, Nathan Yu, Shounak Rangwala
Advisors: Ivan Seskar, Jenny Shane

Project Overview

The goal of this project is to work on development of the WINLAB self driving car simulator. The project includes development of ~1/14 scale vehicles for use as a remote self-driving car testing platform, as well as a virtual simulation environment which will model both the physical vehicles and the testbed environment. Robot Operating System (ROS) will be used for both halves of the project, with the simulation running in Gazebo.

Objectives include:

- Incorporation of ROS control into existing car software
- Use of AI/machine learning algorithms for self driving behavior

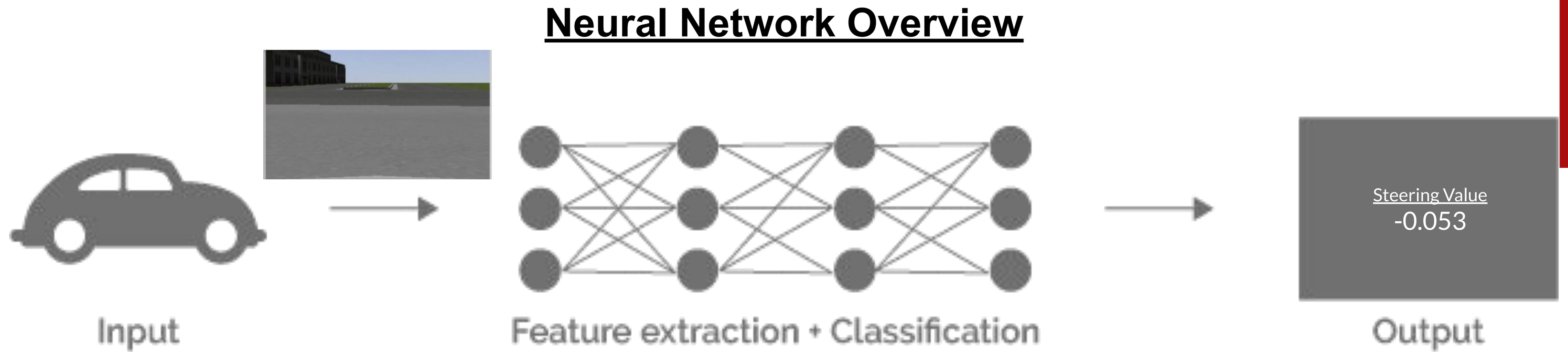
Prius Model

Gazebo Environment

Neural Network

- Image dimensions: 800x800x3
- Model input dimensions: 424x240x3 (cropped using OpenCV)
- Optimizer : Stochastic Gradient Descent (SGD)
- Loss function : MSELoss (because we had continuous labels)
- Batch size for training data : 32 images
- Output : Float 32 byte value

Neural Network Overview



Loss Output

```
Train Epoch: 1 [1696/1827 (91%)]    Loss: 0.000160
Train Epoch: 1 [1728/1827 (93%)]    Loss: 0.000640
Train Epoch: 1 [1760/1827 (95%)]    Loss: 0.019821
Train Epoch: 1 [1792/1827 (97%)]    Loss: 0.001764
```

Selected References

- https://github.com/osrf/car_demo/tree/master/car_demo

Data Collection / Pipeline

Collection:

- Scripted WASD control for the car in Gazebo simulated environment
- Recorded several ROS bagfiles storing steering and image data
- Converted bagfiles into NumPy .npz arrays. Mapped images to corresponding steer commands using timestamps

Training:

- Scripted a DataLoader class to load training data into our model as tensors
- Scripted a training module that used the data from the DataLoader to train the model, and saved the model parameters as a weights file for testing.

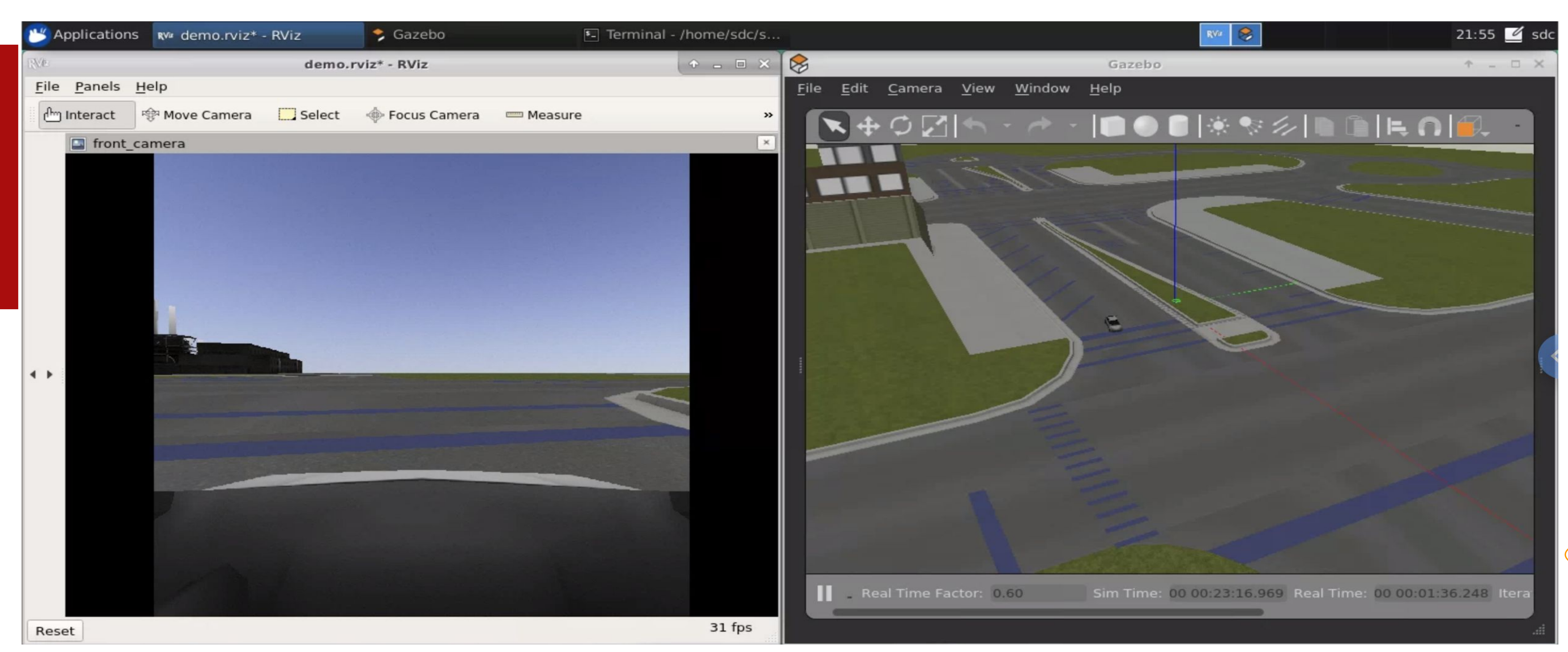
Image Message Info

```
ing/catkin_ws/src:/opt/ros/melodic/share
sdc@node1-1:~/self_driving/catkin_ws$ rosmsg show sensor_msgs/Im
age
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
uint32 height
uint32 width
string encoding
uint8 is_bigendian
uint32 step
uint8[] data
```

Control Message Info

```
sdc@node1-1:~/self_driving/catkin_ws$ rosmsg show prius_msgs/Co
ntrol
uint8 NO_COMMAND=0
uint8 NEUTRAL=1
uint8 FORWARD=2
uint8 REVERSE=3
std_msgs/Header header
  uint32 seq
  time stamp
  string frame_id
float64 throttle
float64 brake
float64 steer
uint8 shift_gears
```

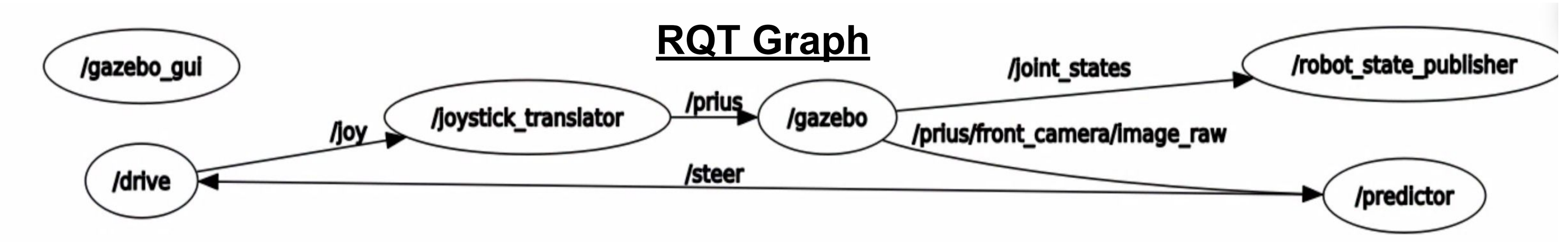
Data Collection Setup



Testing:

- Created ROS node Predictor
  - Subscribes to topic: /prius/front_camera/image_raw, msg type: Image
  - Initializes model from weights file, evaluates a steering value for a given image
  - Publishes to topic: /steer, msg type: Float32
- Created ROS node Drive
  - Subscribes to topic: /steer, msg type: Float32
  - Creates Joy message using steering value and a throttle value of 0.3
  - Publishes to topic: /joy, msg type: Joy

RQT Graph



Prediction Output to Control Message

Results

We trained the model 4 times, each iteration increasing the number of epochs and number of bagfiles.

- 5 epochs / 10 bagfiles: car drove in a constant radius and crashed into the curb
- 10 epochs / 15 bagfiles: car exhibited rudimentary path following, but wobbled left and right at a slow velocity
- 20 epochs / 15 bagfiles: car retained wobbly behavior but less pronounced and higher velocity
- 20 epochs / 37 bagfiles: car drove in a very smooth fashion with high velocity, and was able to merge onto another lane with ease.

Future Plans

For the future directions of this project, we are planning to combine the virtual self driving car model that was developed, and the miniature car model to merge with the Smart Intersection / Intersection Simulation groups.

Other objectives include:

- Implement turning behavior given an input
- Implement Ackermann steering to generalize steering values using turn radius
- Averaging multiple samples to create one steer value for smoothing movement
- Create controller node between the output of CNN and Control message to translate car-specific command signals to real world signals of velocity/turn radius

WINLAB