2017 IEEE Intelligent Vehicles Symposium (IV)
June 11-14, 2017, Redondo Beach, CA, USA

Baidu Driving Dataset and End-to-End Reactive Control Model*

Hao Yu[1], Shu Yang[2], Weihao Gu[3] and Shaoyu Zhang[4]

***Abstract*— End-to-end autonomous driving system has obtained great progress recently. In this paper, we will introduce our open source dataset: Baidu Driving Dataset(BDD), and our end-to-end reactive control model trained on BDD.**

The BDD comes from Baidu street view project, which generates millions of kilometers driving data every year. Among them, we publish 10000 kilometers driving data for end-to-end autonomous driving research. The BDD consists of two parts: forward images and vehicle motion attitude. The vehicle motion attitude is derived from real time kinematic GPS location data with standard deviation of 3 centimeters.

Our reactive control model consists of lateral control and longitudinal control. We employ curvature instead of steering angle for lateral control, and leverage acceleration, not throttle or brake, for longitudinal control. CNN network is employed for lateral control model, mapping a single image from forward camera directly to corresponding curvature. For longitudinal control, stacked convolutional LSTM is used to extract spatial and temporal features from a sequence of frames, and to map the features with longitudinal control commands. The demo and data are in http://roadhackers.baidu.com. To the best of our knowledge, it is the first time that both lateral and longitudinal control are implemented in an end-to-end style.

I. INTRODUCTION

There are generally two kinds of solutions for autonomous driving. The first one is traditional style, which is mainly based on world model, namely High Definition Map(HD Map). The second one is end-to-end style, which directly extract control commands by mapping raw sensor data to control commands.

In traditional style, the autonomous driving system is divided into several sub-systems, such as sensor fusion, perception, self-localization, world model(HD Map) and planning. The details of each sub-system are beyond the scope of this paper. An example of traditional solution is BOSS[1]. The advantage of this style is that the whole system is decomposed into different less complicated modules which are easier to be understood and optimized by engineers. The disadvantages are mainly associated with two aspects: complexity and costs. Even though the system could be maturely decomposed into sub-systems, the complexity is still huge so that it often beyonds the capability of one company. The high cost not only comes from sub-systems' development, but also comes from the production and updating of HD Map. Therefore, most traditional solution are still limited in enclosed environments.

End-to-end solution has achieved significant improvement in recent years, benefiting from the explosion development of deep learning methods, data technology and high performance computing hardware, such as GPU and FPGA. The breakthrough came in 2012, when AlexNet[3] significantly improved the performance of image recognition in the annual ImageNet Large Scale Visual Recognition Challenge (ILSVCR). Afterwards, lots of network architectures were proposed in the fields of computer vision, voice recognition and natural language processing. In the field of robot control and planning, where autonomous driving belongs to in a broad sense, end-to-end network has also made great progress. Zhu[4] put forward an indoor navigation architecture through deep reinforcement learning. Finn[5] proposed a robot planning system considering physical interaction using video prediction. Tamar[6] proposed value iteration network, which adds path-planning model into reinforcement learning system using convolutional networks.

The beginning of end-to-end autonomous driving history can be traced back to 1989, when Dean[7] proposed the Autonomous Land Vehicle in a Neural Network(ALVINN) system. Employing shallow fully connected neural network on small size gray pictures, ALVINN showed the feasibility to map raw video image directly to steering commands. In 2004, DAVE [8], short for DARPA Autonomous Vehicle, demonstrated off-road driving using binocular cameras through end-to-end learning. In 2016, Nvidia published its autonomous driving system DAVE-2[9] using end-to-end method. With the great computing power of DRIVE-PX, DAVE-2 could dr the car on public road in open environments.

The drawbacks of end-to-end autonomous driving are mostly on two aspects: incomprehensible to human and lack of real driving dataset. In terms of the second point, even though several datasets were released for research past years, such as comma.ai[10], Udacity[11] and Oxford[12], the datasets are still not enough, especially for different road conditions and countries.

In this paper, we will introduce a new driving dataset: Baidu Driving Dataset(BDD), which contains 10000 kilometers' frontal camera images and vehicle motion attitude data of real read conditions. The data come from Baidu street view project and the detail of dataset will be represented in Section.II. Based on the dataset, we propose our end-to-end reactive control model, which inlcudes lateral control and longitudinal control. Lateral control is also usually called steering control. While longitudinal control means

*This work was supported by Baidu Inc.

[1]Hao Yu is with Baidu Intelligent Vehicle Department, Baidu Inc, Beijing. yuhao@baidu.com

[2]Shu Yang is with Beijing University of Posts and Telecommunications, Beijing. He was an intern in Baidu Inc. assureys@bupt.edu.cn

[3]Weihao Gu is with Baidu Intelligent Vehicle Department, Baidu Inc, Beijing. guweihao@baidu.com

[4]Shaoyu Zhang is with Baidu Intelligent Vehicle Department, Baidu Inc, Beijing. zhangshaoyu@baidu.com

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on August 01,2023 at 20:21:24 UTC from IEEE Xplore. Restrictions apply.

accelerating and braking control. The detail of this model will be introduced in Section.III. This paper ends, with results in Section.IV and conclusion in Section.V. To the best of our knowledge, It is the first time that both laeteral and longitudinal control are implemented in an end-to-end style.

II. BAIDU DRIVING DATASET

A. HD Data Collection Platform

The data was collected with the Baidu street view platform, as illustrated in Fig.1. The platform is equipped with following sensors:

- $3\times$Monocular Camera, $2448\times2048\times3$, 8Hz. One heads to front and two heads to sides.
- $1\times$Lidar, 32 beams to reconstruct 3D environment.
- $1\times$IMU, collects attitude of ego vehicle.

The platform achieves 3 centimeters locating precision with real time kinematic GPS.

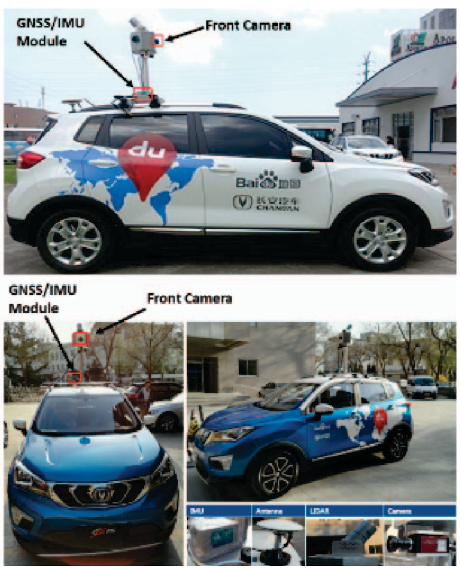

Fig. 1. Baidu HD Data Collection Platform. The platform is equipped with multiple sensors.

B. Raw Data from Device

In this paper, we will use two kinds of raw data from devices. Frontal Images were taken 8 frames per second to record road condition ahead of driver. Accurate location and attitude data were records 20 times per second. Fig.2 shows the format of raw data from GNSS/IMU module. a

The 10000 kilometers data collection spans the period of March 2016 to November 2016, recording express way driving conditions in dozens of major city in China. The vehicle was driven manually throughout the period of data collection. Drivers, also data collectors, were asked to drive moderately and smoothly during collecting. The total size of the raw data is approximately 10TB before preprocessing. Fig.4 presents a montage of images taken from different clips, illustrating the range of complicated situations of our dataset.

```
Column/Variable Contents, Units and Description:
  01: Week        GPS Weeks                Week number starting at Jan 4, 1980
  02: GPSTime     Seconds of the Week      Time of epoch or feature - Receiver time frame
  03: Longitude   Decimal Degrees (signed) East/West Geographic coordinate
  04: Latitude    Decimal Degrees (signed) North/South Geographic coordinate
  05: H-Ell       Metres                   Height above the current ellipsoid
  06: VNorth      Meters per Second        North local level velocity
  07: VEast       Meters per Second        East local level velocity
  08: VUp         Meters per Second        Up local level velocity
  09: Heading     Decimal Degrees (signed) GNSS/INS computed heading value - rotation about body z-axis
  10: Pitch       Decimal Degrees (signed) GNSS/INS computed pitch value - rotation about body x-axis
  11: Roll        Decimal Degrees (signed) GNSS/INS computed roll value - rotation about body y-axis
  12: Q                                    Quality factor where 1 is best and 6 is worse
  13: NS                                   Number of total satellites (GPS+GLONASS+BeiDou) used in solution
  14: PDOP                                 Position Dilution of Precision, which is a measure of X, Y, Z position geometry
  15: iFlag                                Principle (strongest) type of updated applied in IMU processing
  16: StdDev      Metres                   Estimated local level position error (east, north and up axes)
  17: HdngSD      Decimal Degrees (signed) GNSS/INS estimated heading accuracy
  18: PitchSD     Decimal Degrees (signed) GNSS/INS estimated pitch accuracy
  19: RollSD      Decimal Degrees (signed) GNSS/INS estimated roll accuracy
  20: Text                                 $C8$

   Week   GPSTime    Longitude     Latitude   H-Ell  VNorth  VEast    VUp Heading  Pitch   Roll Q NS  PDOP iFlag  StdDev HdngSD PitchSD RollSD Text
(weeks)     (sec)        (deg)        (deg)     (m)   (m/s)  (m/s)  (m/s)   (deg)  (deg)  (deg)      (dop)           (m)  (deg)   (deg)  (deg)
   1887 442531.00 116.654117071 39.95188788  14.061   0.766 11.544 -0.129  86.656  0.371  0.637 1 14  1.29 GPS      0.024  0.035   0.010  0.009 $C8$
   1887 442531.05 116.654123822 39.95188823  14.054   0.757 11.531 -0.146  86.668  0.275  0.610 1 14  1.29 GPS      0.024  0.038   0.012  0.011 $C8$
   1887 442531.10 116.654130565 39.95188857  14.047   0.763 11.526 -0.139  86.677  0.294  0.615 1 14  1.29 GPS      0.024  0.041   0.014  0.013 $C8$
   1887 442531.15 116.654137305 39.95188891  14.040   0.771 11.512 -0.158  86.686  0.253  0.593 1 14  1.29 GPS      0.024  0.044   0.016  0.014 $C8$
   1887 442531.20 116.654144041 39.95188926  14.032   0.752 11.514 -0.164  86.695  0.317  0.577 1 14  1.29 GPS      0.024  0.047   0.017  0.015 $C8$
   1887 442531.25 116.654150780 39.95188959  14.024   0.754 11.525 -0.161  86.702  0.414  0.598 1 14  1.29 GPS      0.024  0.049   0.019  0.016 $C8$
```

Fig. 2. A glimpse of raw data. It mainly includes GPS and attitude information.

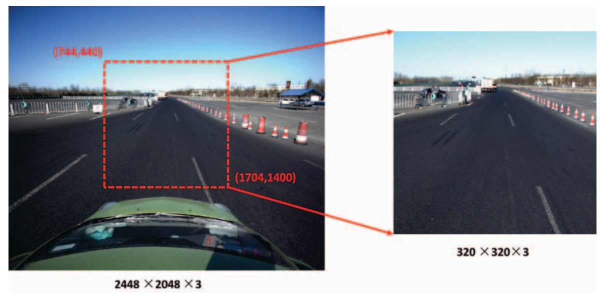

Fig. 3. Raw image is tailored and undersampled.

C. Data Preprocessing

Each raw image contains a large amount of environmental information while the central part is of great importance. To reduce complexity for CNN-training, a sub image is extracted as well as under-sampled. As Fig.3 shows, a ready-to-feed image is at size of $320\times320\times3$.

Vehicle curvature is a critical parameter for driving. According to curvature's formula:

$$k(t) = \frac{f^{''}(t)}{(1+f^{'}(t))^{3/2}} \tag{1}$$

$f^{'}(t)$ is the speed of ego vehicle, and $f^{''}(t)$ is acceleration respectively. $f^{'}(t)$ is directly accessible from raw data with 20 times/s. $f^{''}(t)$ could be calculated from $f^{'}(t)$ easily. Thus, curvature of time t could be calculated from (1). We called this method.1. Despite its simplicity, method.1 has a disadvantage. The disadvantage comes from its lacking of resistance to speed noise. The noise in speed $f^{'}(t)$ significantly disturbs acceleration $f^{''}(t)$, thus having a negative impact on curvature calculation. The curvature falls into oscillation when the speed is smaller than 5m/s. With this curvature calculation method, nearly 40%-50% attitude data could not be used.

To make full use of our dataset, we calculate curvature from trajectories by method.2. We first find the best spline fitting of short-term(within several seconds) trajectory, then calculate curvature by (1). More specifically, for time t, a series of trajectories from $(t-0.875,\ t+0.875)$, totally 16 location points, are used to generate a most suitable spline curve. As Fig.5(a) demonstrates, 16 discrete points are fitted into a smooth curve, which well represents the vehicles trajectory. As Fig.5(b) shows, method.2 is still effective when

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on August 01,2023 at 20:21:24 UTC from IEEE Xplore. Restrictions apply.

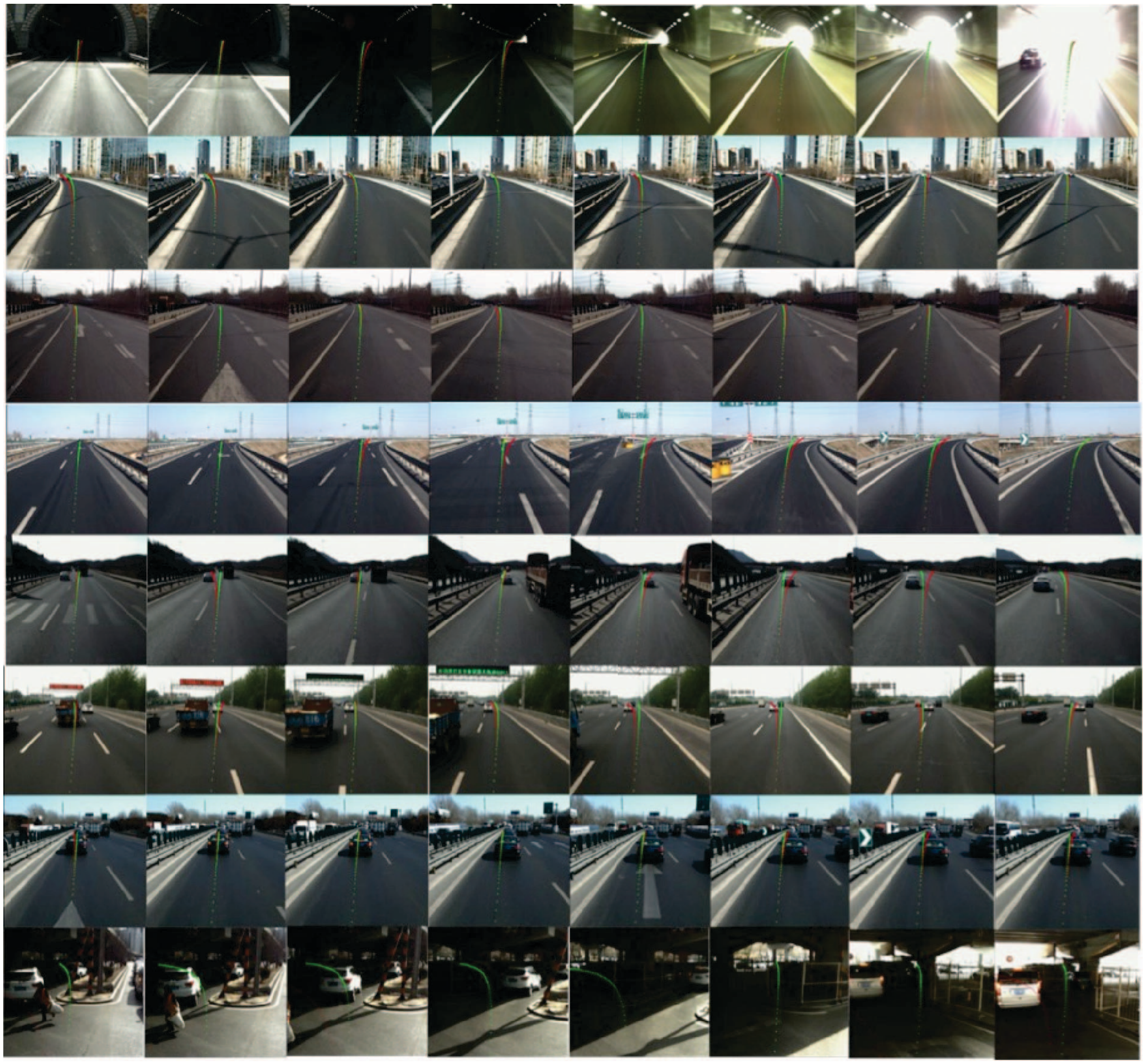

Fig. 4. A montage of images taken from 8 different clips. Each row presents a series of continoues images.

the speed is extremely low and where noise has a relative large influence.

To validate two methods of curvature calculation, 50 pieces of straight trajectories are selected and set as benchmarks. We calculate std(standard deviation) on 50 pieces of trajectories by two methods. In Fig.5(c), a comparison illustrated two things: First of all, slower speed could render bigger std. Secondly, method.2 with a spline fitting process, outperforms method.1 by decreasing 50% to 80% std.

Finally, image and curvature are associated by time. considering the decision-making delay of drivers, we associate the of $image_t$ with $curvature_{t+0.125}$.

D. Open Source in Road Hackers Project

We have released the BDD in Road Hackers, an open source project for autonomous driving. In accordance with the laws of China, locations with centimeter-precision are prohibited to release for public. Therefore, we only opens images and curvatures with a total size of 1.6TB, which covers major China cities. Both image and curvature data are stored in *HDF5* format. Road Hackers also provides a benchmark for industrial and academic research. The full dataset is available at http://roadhackers.baidu.com.

III. REACTIVE MODEL

Before surveying into reactive model, it is necessary to introduce the architecture of traditional autonomous driving system and make a comparison between the traditional and end-to-end systems in functional perspective.

A. Comparison

Typical pipeline of traditional system consists of sensor fusion, HD Map, perception, planning and actuation. After the perception step, the world model is built including both dynamic objects and static features. On the basis of world model, planning can be divided into reactive planning and

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on August 01,2023 at 20:21:24 UTC from IEEE Xplore. Restrictions apply.

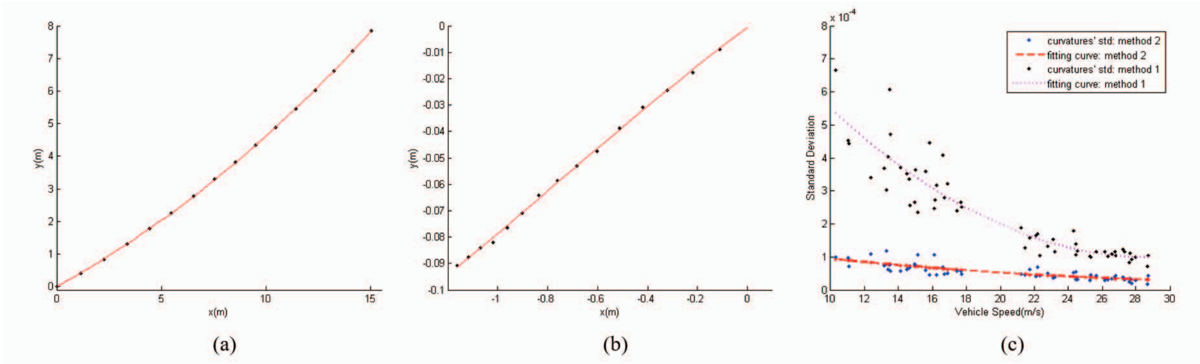

Fig. 5. curvature fitting and the noise. (a)curvature fitting under high speed. (b)curvature fitting under low speed. (c)a comparison of two fitting methods.

proactive planning[13]. Reactive planning operates on a few simple rules, such as follow a line or move toward open space. In contrast, proactive planning involves an entire sequence of movements or a high level path from current to goal position. Generally, proactive planning can be implemented with pipeline of navigation, decision making, trajectory planning, and reactive planning acts in low level, executing the most basic instructions including lateral and longitudinal control. The reactive functions are like Adaptive Cruise Control (ACC), Autonomous Emergency Braking (AEB), Lane Keeping System (LKS).

Traditional style atonomous driving system has already involved proactive control, while current end-to-end solution still stays at reactive control. No proactive path or trajectory planning function is implemented with end-to-end style. In fact, only lateral control, namely steering control, has been implemented on the publication [8][9]. Table.I shows a comparison of state-of-the-art.

TABLE I

A FUNTION COMPARISON OF STATE-OF-THE-ART

Function		Traditional	End-to-End
Proactive	Navigation	√	×
	Decision Making	√	×
	Trajectory Planning	√	×
Reactive	longitudinal Contrl	√	×
	Lateral Control	√	√

B. Lateral Control

In the way of end-to-end, most lateral control methods map the input image directly to steering wheel angle[8][9]. However, we use the curvature of vehicle instead of steering wheel angle. There are two main reasons here:

- There is a relatively mature functional model mapping from steering angle/speed to curvature: Ackermann model[14], which can be fulfilled in either hand-draft function or in a simple and shallow network. When the speed is slow, the relationship can be obtained by Ackermann steering geometry. And when the speed is high, the relationship is more complicated, including slip factors, whether condition. However, the highly accurate mapping function can be approximated with simple and robust functions.
- Curvature is more universal than steering angle. For example, at a certain point of a high-speed road curve, different vehicles should run along the road with the same curvature. Nonetheless, given the same curvature, different vehicles may take different steering angles, resulting from different steering ratio, wheelbase, tire friction and other vehicle dynamics parameters. Therefore, a steering-oriented control model trained from one vehicle type is hard to transfer to another type directly. Threfore, we employ the curvature as the output of the lateral model, leaving the mapping from curvature to steering angle to be fulfilled in an adaptive module, such as a hand-draft function or a simple neural network.

The architecture of our lateral control model is similar to DAVE-2's end-to-end steering model. It is shown in Fig.6 and consists of a preprocessing layer, 5 convolutional layers and 2 fully connected layers. The input image is RGB format with size of 320×320 pixels. In the preprocess layer, the image data are normalized from $[0, 255]$ into $[-1, 1]$ with mean value of 0. Like DAVE-2, the convolution layers perform the feature extraction. After a series of experiments, some parameters are configured and tested. The first convolution layer is set a 4×4 stride and a 5×5 kernel. The following two layers have 2×2 stride and 5×5 kernel. The last two layers are with 2×2 stride and 3×3 kernel. Between convolution layers, LeakyReLU with alpha value of 0.2 is employed. Our fully connected layers only contains tow layers which are much smaller than DAVE. The first fully connected layer has 512 neurons, and the second one has 1 neuron acting as the final linear output unit. Before each fully connected layer, dropout layer is used for better generalization. Finally, the loss function is MSE(mean squared error) between the truth value and the output value of the network.

To get better performance, some tricks are leveraged in the training phase. The first trick is that training data are filtered by two conditions: one condition is that the speed must be greater than 5m/s, another condition is the curvature must be smaller than 0.5. The speed filter comes for noise reducing, when speed is higher the observation error of curvature is smaller. Mostly, the turning radius is larger than 2 meters,

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on August 01,2023 at 20:21:24 UTC from IEEE Xplore. Restrictions apply.

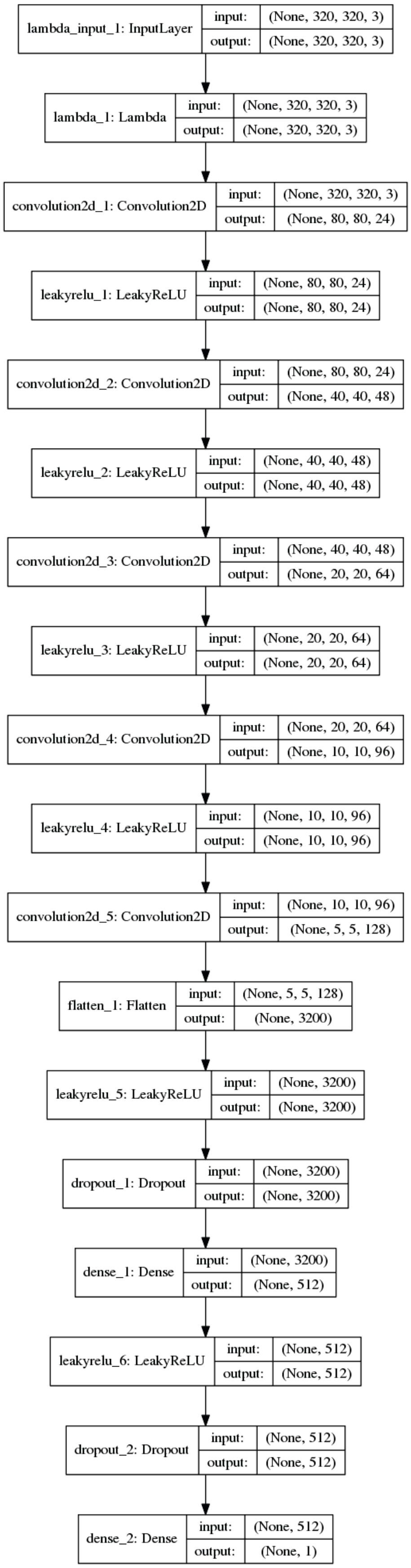

Fig. 6. The architecture of our lateral control model. It consists of preprocessing layer, convolutional layers and fully connected layers.

so the curvature is limited to be smaller than 0.5. The second trick is value amplification. Because the MSE of curvature is usually quite small, the truth-value is multiplied by 1000 in order to get bigger gradient and faster convergence rate. The third trick, rmsprop method is chosen to be the optimizer. There are totaly 500 training epochs with epoch size of 10000.

C. Longitudinal Control

We believe that one single image is insufficient for longitudinal control. For example, in a traffic jam scene, it almost impossible to judge whether the front vehicle is stationary or starting. The decision should be made though analysis and prediction of a series of timed data, which are usually implemented by Kalman filtering or particle filtering in traditional autonomous driving system. In this paper, we classify the longitudinal control problem as a spatiotemporal sequence prediction problem, and make use of latest video frames to be trained in end-to-end style.

Lots of researches have been conducted on video prediction. In 2010, Ji[15] proposed 3D convolution, adding temporal information on the base of 2D convolution to recognize human action. Vondrick[16] proposed a generative adversarial network combined with spatial-temporal convolution, which separats foreground and background streams to predict plausible futures of static images. Shi[17] employed convolutional LSTM network to predict the future rainfall intensity. All these models can extract features of videos. Convolutional LSTM method is employed in this paper, because it not only extracts the spatiotemporal features but also preserves the advantages of LSTM. We hold that lots of research progresses of LSTM can be applied into autonomous driving system.

The architecture of longitudinal control model is described in Fig.7. The input is last 5 frames, which is taken from front camera with frequency of 8FPS. Each frame is scaled into 80×80 in RGB format. The frames are fed into the stacked convolutional LSTM model. The first layer has 64 channels with kernel size of 5×5. The following layers have more channels and smaller kernels. The channel size we choose here is relatively big, because it is more convenient to be extended to deeper networks in the future. Following the last convolutional layer is the two fully connected layers. The activation neuron is LeakyReLU. The output unit is linear unit. The loss function is MSE and the optimizer is rmsprop.

There are totally 600 epochs are employed in the longitudinal control training phase. In the first 400 epochs, the learning rate is 0.0001. And in the last 200 epochs, the learning rate is set 0.000001.

IV. Results

We divide the dataset into three subsets: training set, validate set and test set. The total size of the 10000 kilometers dataset is 1.6 TB, among which 58 GB data are randomly chosen to be the test data and 57 GB data as validate data. The rest data are used for training.

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on August 01,2023 at 20:21:24 UTC from IEEE Xplore. Restrictions apply.

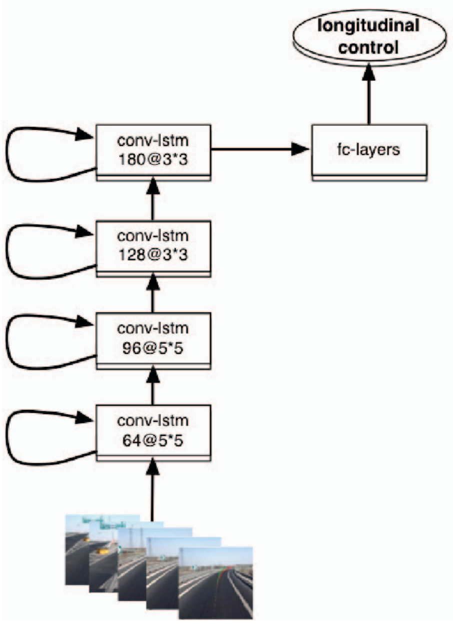

Fig. 7. The architecture of our longitudinal control model.

After 500 epochs, the lateral control model converges to a error about 10^{-6}, as shown in Table.II. The results can also been seen in a video in the following URL: http://roadhackers.baidu.com. We can see that this model can predict the right curvature not only according to the road curve but also according to the relative position and angle between the vehicles. It works just like a human driver. We can also observe that this model cannot handle lane changing or overtaking cases on the road. From a statistical point of view, this model chooses the most frequently used curvature from training dataset given a certain driving condition without contexts, so no further planning is made.

TABLE II
LATERAL RESULTS

Data Category	Error
Train	0.8904×10^{-6}
Validate	0.7728×10^{-6}
Test	1.2445×10^{-6}

As shown in Table.III, the error of longitudinal control is reduced to 0.1071 and test error is 0.1108. From the demo video, it is clearly that this model can predict the longitudinal commands when a braking is in need. It proves our hypothesis that longitudinal commands should be inferred from driving context, which can be represented by a sequence of frames of the environments.

TABLE III
LONGITUDINAL RESULTS

Data Category	Error
Train	0.1071
Validate	0.1329
Test	0.1108

V. CONCLUSION

In this paper, we introduced a new driving dataset: BDD, derived from hundreds of street view cars running over China. Based on BDD, we introduced and demonsatrated our end-to-end reactive control model, including lateral and longitudinal control. To the best of our knowledge, it is the first time that both lateral and longitudinal control are implemented in an end-to-end style.

Practically, before road test, reactive control should consider not only safety but also comfort degree. To ensure safety, more complete verification methods and measurments are needed. For comfort, the network should be improved to generate more smooth commands. How to let the network interact with navigation instruction to drive on real road remains challenging and promising.

REFERENCES

[1] Urmson, Chris, et al. "Autonomous Driving in Urban Environments: Boss and the Urban Challenge." Journal of Field Robotics 25.8(2008):425466.
[2] Ian Goodfellow, Yoshua Bengio and Aaron Courville. "Deep Learning." (2016): 2016. Web.
[3] Krizhevsky, Alex, I. Sutskever, and G. E. Hinton. "ImageNet classification with deep convolutional neural networks." International Conference on Neural Information Processing Systems Curran Associates Inc. 2012:1097-1105.
[4] Zhu, Yuke, et al. "Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning." (2016).
[5] Finn, Chelsea, Ian Goodfellow, and Sergey Levine. "Unsupervised learning for physical interaction through video prediction." Advances In Neural Information Processing Systems. 2016.
[6] Tamar, Aviv, et al. "Value iteration networks." Advances in Neural Information Processing Systems. 2016.
[7] Pomerleau, Dean A. "ALVINN, an autonomous land vehicle in a neural network." No. AIP-77. Carnegie Mellon University, Computer Science Department, 1989.
[8] LeCun, Y., Muller, U., Cosatto, E., & Flepp, B. (2006). " Off-road obstacle avoidance through end-to-end learning." Advances in neural information processing systems. 2005.
[9] Bojarski, Mariusz, et al. "End to end learning for self-driving cars." arXiv preprint arXiv:1604.07316 (2016).
[10] Santana, Eder, and George Hotz. "Learning a driving simulator." arXiv preprint arXiv:1608.01230 (2016).
[11] Udacity Self Driving Car, https://www.udacity.com/self-driving-car.
[12] Maddern W, Pascoe G, Linegar C, et al. "1 year, 1000 km: The Oxford RobotCar dataset." The International Journal of Robotics Research. 2016.
[13] Lee, D. C. "The Map-Building and Exploration Strategies of a Simple Sonar-Equipped Mobile Robot." Cambridge University Press, New York. 1996.
[14] Mitchell, Wm C., Allan Staniforth, and Ian Scott. "Analysis of ackermann steering geometry." No. 2006-01-3638. SAE Technical Paper, 2006.
[15] Ji, Shuiwang, et al. "3D convolutional neural networks for human action recognition." IEEE transactions on pattern analysis and machine intelligence 35.1 (2013): 221-231.
[16] Vondrick, Carl, Hamed Pirsiavash, and Antonio Torralba. "Generating videos with scene dynamics." Advances In Neural Information Processing Systems. 2016.
[17] Xingjian, S. H. I., et al. "Convolutional LSTM network: A machine learning approach for precipitation nowcasting." Advances in Neural Information Processing Systems. 2015.

Authorized licensed use limited to: New Jersey Institute of Technology. Downloaded on August 01,2023 at 20:21:24 UTC from IEEE Xplore. Restrictions apply.