Supplementary Materials: Testing the Safety of Self-driving Vehicles by Simulating Perception and Prediction

Kelvin Wong[1,2⋆], Qiang Zhang[1,3⋆], Ming Liang[1], Bin Yang[1,2], Renjie Liao[1,2], Abbas Sadat[1], and Raquel Urtasun[1,2]

[1] Uber Advanced Technologies Group, Toronto, Canada
[2] University of Toronto, Toronto, Canada
[3] Shanghai Jiao Tong University, Shanghai, China
{kelvin.wong, ming.liang, byang10, rjliao, asadat, urtasun}@uber.com
zhangqiang2016@sjtu.edu.cn

Abstract. In this document, we provide additional details to supplement the main text. We first describe additional experiment details (Sec. 1). Then, we provide additional quantitative results that study our approach's generalization performance (Sec. 2.1) and the effect of the training dataset's size on final performance (Sec. 2.2). Finally, in Sec. 3, we present a number of qualitative results that demonstrate the efficacy of using perception and prediction simulation for testing motion planning.

1 Additional Experiment Details

1.1 Model Architectures

MultimodalNoise. We implement MultimodalNoise as a Gaussian Mixture Model with $k = 8$ components, each with a full covariance matrix. We use the Scikit-learn implementation [5]. We also model misdetection noise by fitting a Bernoulli distribution to the rate of false negative detections in our training split.

ActorNoise. ActorNoise takes as input the actor's bounding box parameters (x, y, w, h, θ), where (x, y) is the box's center, (w, h) are the box's width and height, and θ is the box's heading angle, as well as the actor's past and future positions centered at (x, y). This input feature vector is then processed by a multi-layer perceptron. In particular, we use a model architecture consisting of: (i) an initial fully-connected layer with 128-dimensional hidden features, ReLU activations [1], and group normalization [7]; (ii) two fully-connected residual blocks [3] with 128-dimensional hidden features, ReLU activations, and group normalization; and (iii) a final fully-connected layer to predict perturbations to the actor's bounding box and future states as well as a misdetection score.

⋆ Indicates equal contribution. Work done during Qiang's internship at Uber ATG.

	Perception Metrics ↑				Prediction Metrics ↓			
	AP (%)		Max Recall (%)		ADE (cm)		FDE (cm)	
Vehicle	0.5 IoU	0.7 IoU	0.5 IoU	0.7 IoU	70% R	90% R	70% R	90% R
NoNoise	77.2	71.9	98.0	**94.5**	73	73	141	141
ContextNoise	**94.5**	**88.7**	**98.4**	93.4	**54**	**54**	**96**	**92**
Pedestrian	0.3 IoU	0.5 IoU	0.3 IoU	0.5 IoU	60% R	80% R	60% R	80% R
NoNoise	47.4	46.5	82.1	81.4	**36**	**36**	64	64
ContextNoise	**81.9**	**78.8**	**96.7**	**93.0**	37	38	**60**	**61**
Bicyclist	0.3 IoU	0.5 IoU	0.3 IoU	0.5 IoU	50% R	70% R	50% R	70% R
NoNoise	47.8	45.2	99.5	96.7	**77**	**77**	143	143
ContextNoise	**89.6**	**87.2**	**99.6**	**97.8**	79	83	**132**	**140**

	ℓ_2 Distance (cm) ↓			Collision Sim. (%) ↑		Driving Diff. (%) ↓		
	1.0s	2.0s	3.0s	IoU	Recall	Beh.	Jerk	Acc.
PLT								
NoNoise	1.2	3.5	5.6	70.3	72.8	0.32	0.69	1.71
ContextNoise	**0.7**	**1.7**	**2.6**	**85.2**	**90.4**	**0.08**	**0.06**	**0.01**
ACC								
NoNoise	1.7	8.0	19.8	62.2	62.6	-	0.26	0.46
ContextNoise	**1.4**	**6.3**	**15.1**	**76.8**	**77.8**	-	**0.17**	**0.17**

Table 1. Generalization to structured test scenarios. We evaluate NoNoise and ContextNoise (trained on ATG4D) on 500 logs of structured tests collected at a test track. **R** denotes the common recall point at which prediction metrics are computed.

ContextNoise. As we discussed in the main text, ContextNoise consists of three components: (i) a shared backbone feature extractor; (ii) a perception head to simulate bounding box outputs; and (iii) a prediction head to simulate future states outputs. We adapt the backbone network architecture described in [4] to process raster image inputs and output a 4x downsampled 256-dimensional feature map. Our perception head is a single 2D convolution layer with 1×1 kernels and our prediction head is a multi-layer perceptron adapted from the architecture used in ActorNoise. We use non-maximum suppression thresholds of 0.5 IoU for the cars and vehicles and 0.3 IoU for the pedestrians and bicyclists.

2 Additional Quantitative Results

2.1 Generalization to Structured Test Scenarios

In order to study our approach's generalization performance to novel interesting-to-test scenarios, we evaluate NoNoise and ContextNoise (trained on ATG4D) on 500 logs of structured tests collected at a test track. These logs contain rare

and safety-critical scenarios that are commonly used to evaluate self-driving vehicles. Note that this dataset is selectively labeled; that is, we only annotate actors that might interact with the SDV. As such, our noise models are given only these actors as input. To ensure a fair comparison, we compute metrics comparing our simulations against real perception and prediction outputs that are *near*[4] an annotated actor.

Our results are shown in Table 1. They indicate that ContextNoise generalizes to these novel scenarios and produces more realistic perception and prediction simulations than NoNoise. Importantly, they also show that ContextNoise enables more realistic testing of motion planning in simulation. This gives us confidence to use perception and prediction simulation to evaluate motion planning in many variations of these scenarios created by adding or removing actors, varying their speeds, changing the underlying map, *etc*. It is cost-prohibitive and unsafe to do the same with real-world testing.

2.2 Effect of Training Dataset Size

We also study the effects of training dataset sizes on the fidelity of our simulations. To this end, we train five ContextNoise models on progressively smaller subsets of the ATG4D training split, starting from 2500 scenarios (100%) to 125 scenarios (5%), and we evaluate their performance on the full ATG4D validation split. We also evaluate NoNoise, which serves as a baseline for a method that does not require training.

Table 2 shows our results. Unsurprisingly, ContextNoise's simulation fidelity is positively correlated with the size of the training dataset. It is worth noting, however, that even with much less training data, ContextNoise still retains good simulation fidelity. For example, IoU_{col} decreases by just 0.9% (*resp*., 1.2%) in absolute terms for PLT (*resp*., ACC) when the training dataset is halved. We also observe that the effect of the training dataset's size on validation performance varies by class—ContextNoise requires much fewer training scenarios in order to outperform NoNoise on common classes like vehicles than on rare classes like bicyclists.

3 Additional Qualitative Results

In Figs. 1 to 5, we exhibit a number of qualitative results on the ATG4D dataset. Each figure depicts the perception, prediction, and motion planning outputs for one frame of a scenario, which we unroll over three seconds. We depict perception and prediction outputs as purple boxes and the SDV as a red box. We also depict perfect perception and prediction as black boxes and the HD map as gray elements.

The top row of each figure shows the outputs from the PLT planner [6] given real perception and prediction outputs from PnPNet [4]. These outputs

[4] A detected actor is *near* an annotated actor if the intersection-over-union between their bounding boxes exceed 0.1%.

represent the oracle for our task since they are precisely what our simulations aim to emulate. Note that we obtain these outputs by passing real sensor data through PnPNet and PLT. This is possible since ATG4D provides both sensor data and our scenario representation (bounding boxes and trajectories) for every scenario. We emphasize, however, that our simulation approach does not require sensor data.

The middle and bottom row of each figure similarly depicts simulations obtained using NoNoise and ContextNoise respectively. NoNoise represents the prevalent approach of using perfect perception and prediction to test motion planning in simulation [2] whereas ContextNoise is our best simulation model.

Simulating mispredictions. In Fig. 1, we show an example of ContextNoise simulating a misprediction due to multi-modality. Here, both PnPNet and ContextNoise depict the highlighted vehicle as going straight, when it is in fact turning right. Since NoNoise assumes perfect perception and prediction, it falsely depicts the highlighted vehicle as turning right.

Simulating misdetections. In Fig. 2, we show an example where ContextNoise faithfully simulates a misdetection due to occlusion. In particular, both PnPNet and ContextNoise depict the highlighted pedestrians as a parked vehicle. By contrast, NoNoise fails to simulate this misdetection.

Detecting collisions in simulation. In Figs. 3 to 5, we show scenarios in which the PLT planner outputs a trajectory that results in a collision due to misprediction errors. Figs. 3 and 4 show examples of collisions with vehicles whereas Fig. 5 shows an example of a collision with a pedestrian. By faithfully simulating these misprediction errors with ContextNoise, we are able to identify these collision scenarios in simulation. In contrast, when given perfect perception and prediction, the motion planner safely (but unrealistically) avoids all collisions. These examples attest to our ability to realistically test motion planning using simulated outputs from ContextNoise.

References

1. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: Gordon, G.J., Dunson, D.B., Dudík, M. (eds.) Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011 (2011)
2. Gu, T., Dolan, J.M.: A lightweight simulator for autonomous driving motion planning development. In: ICIS 2015 (2015)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016 (2016)

4. Liang, M., Yang, B., Zeng, W., Chen, Y., Hu, R., Casas, S., Urtasun, R.: PnPNet: End-to-end perception and prediction with tracking in the loop. In: 2020 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 16-18, 2020 (2020)
5. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research (2011)
6. Sadat, A., Ren, M., Pokrovsky, A., Lin, Y., Yumer, E., Urtasun, R.: Jointly learnable behavior and trajectory planning for self-driving vehicles. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2019, Macau, SAR, China, November 3-8, 2019 (2019)
7. Wu, Y., He, K.: Group normalization. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIII (2018)

% of Training Split	Perception Metrics ↑ AP (%)		Max Recall (%)		Prediction Metrics ↓ ADE (cm)		FDE (cm)	
Vehicle	0.5 IoU	0.7 IoU	0.5 IoU	0.7 IoU	70% R	90% R	70% R	90% R
NoNoise (0%)	61.1	54.3	93.1	87.8	70	70	132	131
5%	88.0	80.3	93.8	87.4	74	77	123	127
10%	89.3	82.7	94.6	88.6	64	69	108	114
25%	90.6	84.6	95.1	89.5	53	57	86	91
50%	91.1	85.5	95.2	89.8	46	52	73	81
100%	**91.7**	**86.9**	**95.4**	**90.7**	**44**	**49**	**70**	**76**
Pedestrian	0.3 IoU	0.5 IoU	0.3 IoU	0.5 IoU	60% R	80% R	60% R	80% R
NoNoise (0%)	50.8	47.4	82.8	80.0	40	40	69	69
5%	69.6	59.0	84.7	75.8	50	52	78	82
10%	70.2	61.1	85.2	77.6	43	45	66	69
25%	71.4	63.1	85.2	77.2	36	38	55	57
50%	73.4	65.7	86.0	78.7	34	35	**51**	53
100%	**75.2**	**68.6**	**86.6**	**80.3**	**33**	**34**	**51**	**52**
Bicyclist	0.3 IoU	0.5 IoU	0.3 IoU	0.5 IoU	50% R	70% R	50% R	70% R
NoNoise (0%)	33.4	29.8	83.2	78.3	**49**	48	88	87
5%	62.4	47.8	89.4	74.3	126	118	204	190
10%	68.0	55.0	90.6	77.5	90	81	146	129
25%	70.5	60.6	91.0	81.4	64	62	101	96
50%	72.5	63.7	92.2	82.7	55	51	87	78
100%	**74.1**	**64.0**	**92.4**	**82.5**	**49**	**46**	**75**	**70**

% of Training Split	ℓ_2 Distance (cm) ↓ 1.0s	2.0s	3.0s	Collision Sim. (%) ↑ IoU	Recall	Driving Diff. (%) ↓ Beh.	Jerk	Acc.
PLT								
NoNoise (0%)	1.2	3.9	7.5	58.5	65.7	0.16	0.29	0.41
5%	1.1	3.6	6.7	64.9	77.0	0.11	0.26	0.16
10%	1.2	4.1	7.9	67.0	83.2	0.14	0.91	**0.10**
25%	0.8	2.6	4.9	69.2	**86.0**	0.10	**0.07**	0.12
50%	**0.7**	2.4	4.6	74.0	83.2	0.09	0.34	0.23
100%	**0.7**	**2.2**	**4.2**	**74.9**	85.4	**0.07**	0.22	0.20
ACC								
NoNoise (0%)	1.4	7.3	18.5	52.3	53.3	-	0.40	0.27
5%	1.7	9.1	22.9	58.1	68.9	-	0.93	0.15
10%	1.5	7.9	20.0	59.5	73.6	-	0.36	**0.04**
25%	1.3	6.8	16.8	63.1	74.1	-	**0.10**	0.15
50%	**1.1**	5.8	14.7	63.1	**88.2**	-	0.84	0.36
100%	**1.1**	**5.6**	**14.2**	**64.3**	82.3	-	0.38	0.29

Table 2. Ablation of training dataset size on ATG4D validation.

Fig. 1. Simulating mispredictions. We demonstrate ContextNoise's ability to simulate mispredictions due to multi-modality. Here, both PnPNet and ContextNoise depict the highlighted vehicle as going straight when it is in fact turning right. NoNoise cannot simulate such mispredictions.

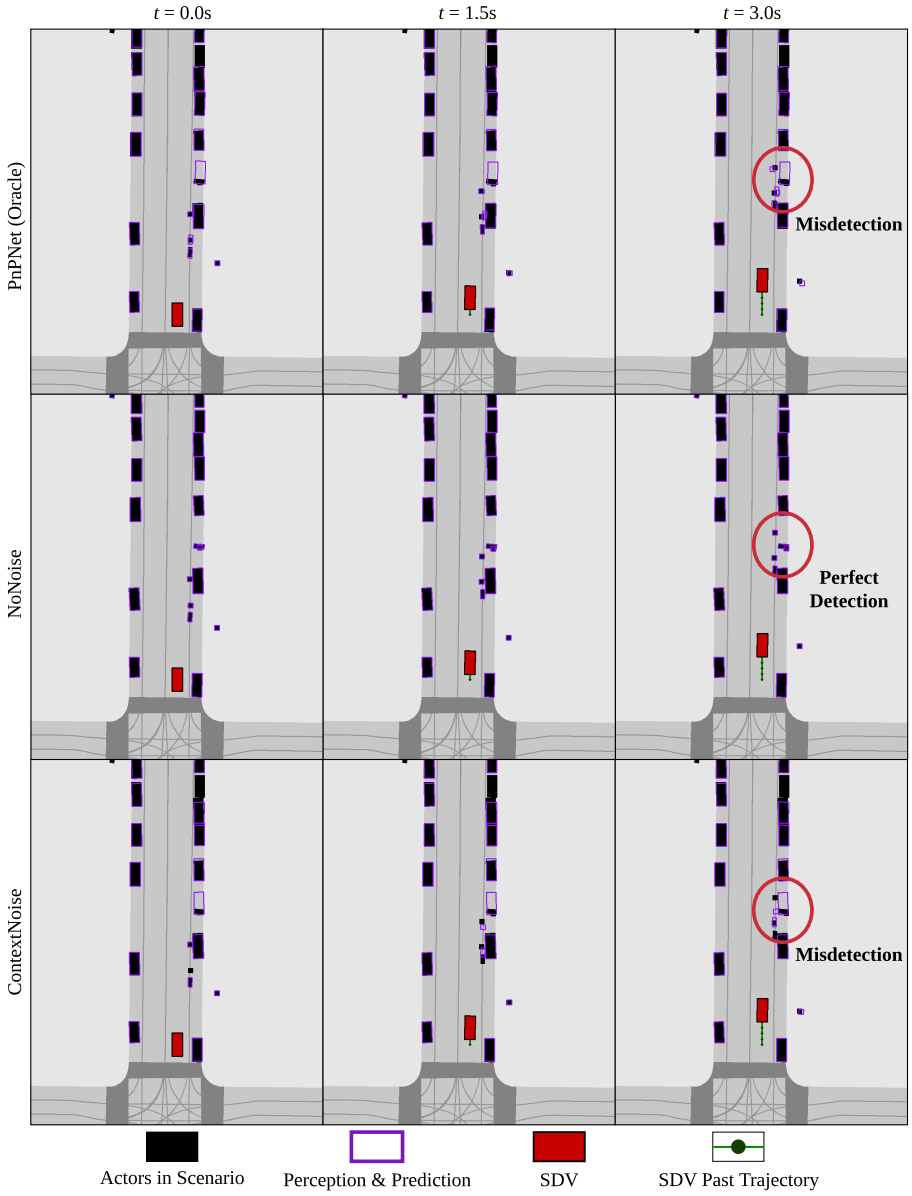

Fig. 2. Simulating misdetections. We demonstrate ContextNoise's ability to simulate misdetections due to occlusion. In particular, both PnPNet and ContextNoise depict the highlighted pedestrians as a parked vehicle. NoNoise fails to simulate this misdetection.

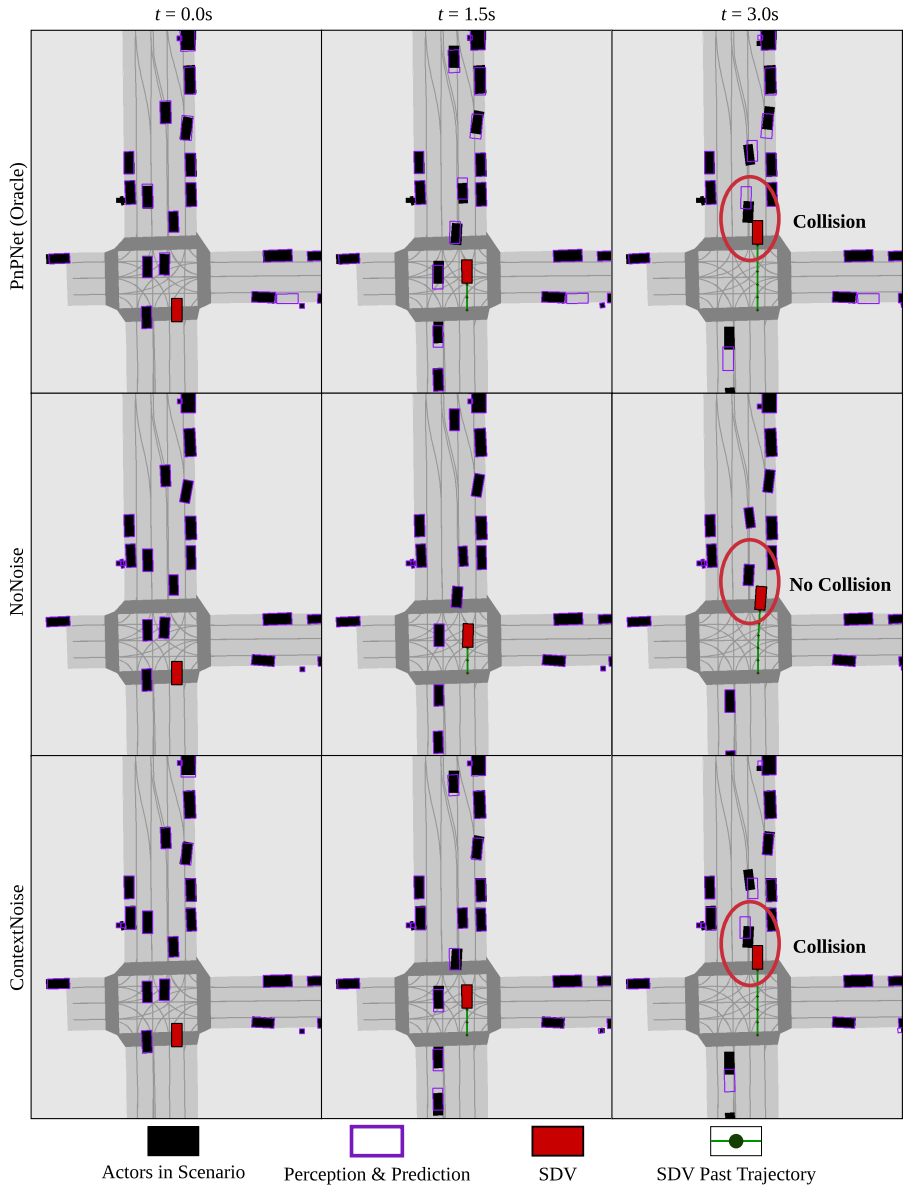

Fig. 3. Detection collisions in simulation. We show an example in which the PLT motion planner outputs a trajectory that results in a collision with a vehicle due to misprediction errors. Using simulated outputs from ContextNoise, we can identify this collision in simulation. In contrast, when given perfect perception and prediction, the motion planner safely (but unrealistically) avoids the collision.

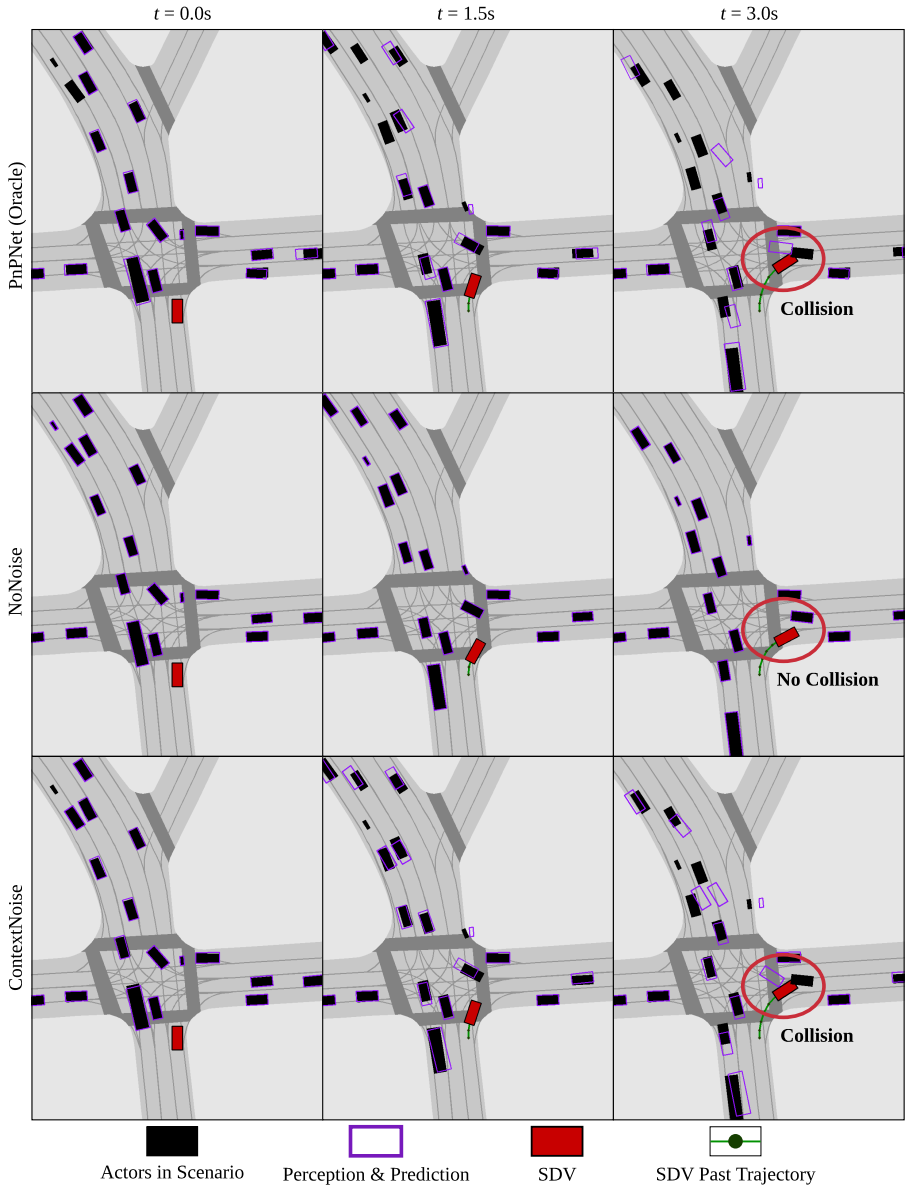

Fig. 4. Detection collisions in simulation. We show an example in which the PLT motion planner outputs a trajectory that results in a collision with a vehicle due to misprediction errors. Using simulated outputs from ContextNoise, we can identify this collision in simulation. In contrast, when given perfect perception and prediction, the motion planner safely (but unrealistically) avoids the collision.

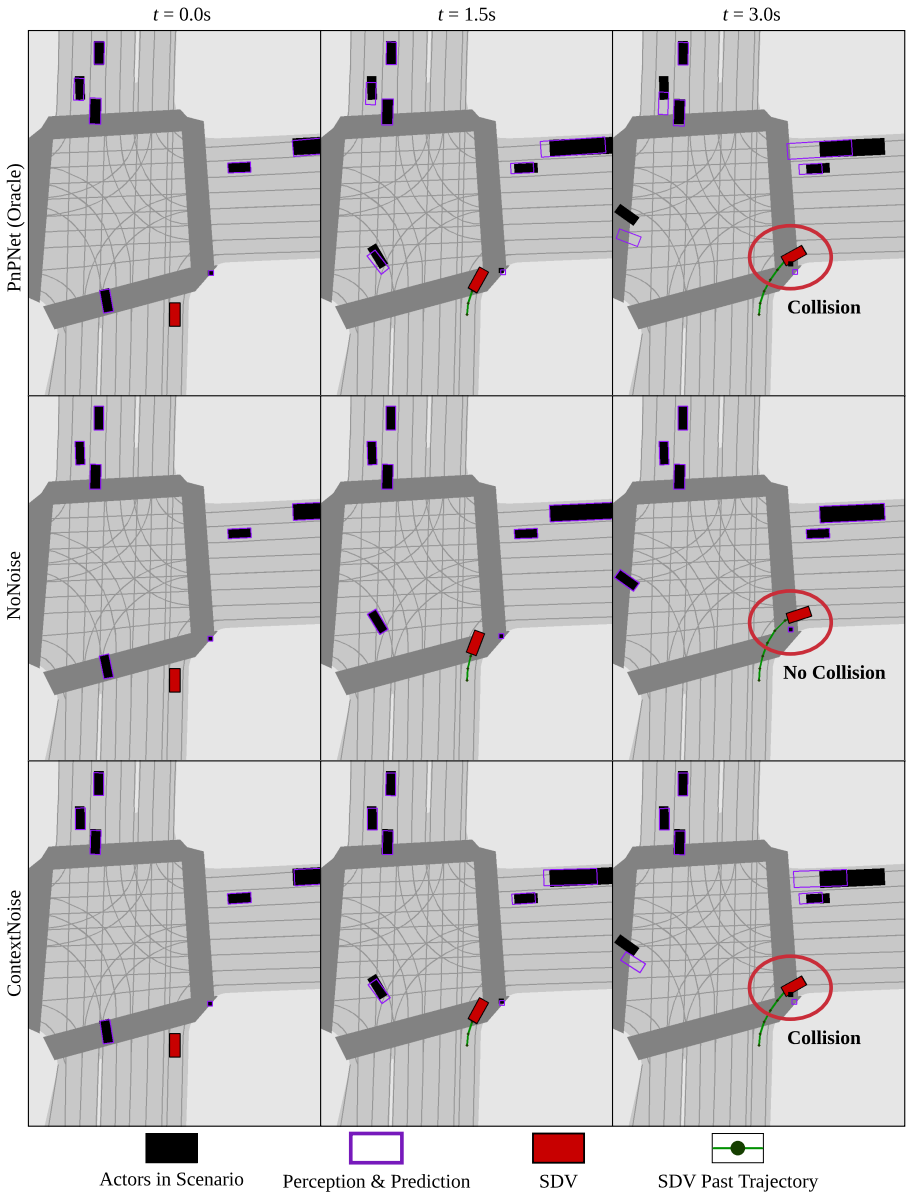

Fig. 5. Detection collisions in simulation. We show an example in which the PLT motion planner outputs a trajectory that results in a collision with a pedestrian due to misprediction errors. Using simulated outputs from ContextNoise, we can identify this collision in simulation. In contrast, when given perfect perception and prediction, the motion planner safely (but unrealistically) avoids the collision.