Supplementary – DAGMapper: Learning to Map by Discovering Lane Topology

Namdar Homayounfar[1,2] Wei-Chiu Ma[1,3]
Justin Liang[1] Xinyu Wu [1] Jack Fan [1] Raquel Urtasun[1,2]
[1]Uber Advanced Technologies Group [2]University of Toronto [3] MIT
{namdar,weichiu,justin.liang,xinyuw,jackwindows,urtasun}@uber.com

In Sec. 1 we present the components of our model in detail. In Sec. 2, we discuss the training details. Finally, we showcase more qualitative examples in Sec. 3.

1. Network Architecture

Global Feature Network: In Fig. 1, we visualize a diagram of the global feature network. In the following:

- **Conv2D(kernel size, filters, stride)** is a 2D convolution layer.
- Each residual **Residual(kernel size, Number of filters, stride)** layer is comprised of two blocks of instance normalization, ReLU nonlineary and Conv2d(kernel size, filters, stride (or 1)). Note that some residual layers perform downsampling by setting the stride of the first convolution layer to two while keeping the stride of the second convolution as one.
- **IRC(kernel size, filters, stride)** corresponds to instance normalization, ReLU followed by a Conv(kernel size, filters, stride) while **IRUC(kernel size, filters, stride)** has an extra nearest neighbor upsampling layer after the ReLU.

Distance Transform Header: We input the features obtrained from the Global Feature Network of Fig. 1 to the distance transform (DT) header of Fig. 2. The GT places the value of 8 on the lane boundaries and decreases as we move away. As such we clip the output of the DT header to be between 0 and 10.

Direction Header: We crop a rotated RoI of size $h \times w$ along the lane boundary from the output of the global feature network and concatenate with the one-hot encoding of the previous inferred state to obtain a tensor with 11 channels. In the following:

- **ConvRNN(kernel size, filters, stride)** is a vanilla RNN that replaces matrix multiplication with convolutions and the $\tanh$ non-linearity with ReLU.
- **Linear(input size, output size)** is a simple linear linear

State Header: The state header has the exact same architecture as of the direction header. However, we replace the global max pooling with global average pooling.

Position Header: For the position header, we employ a convolutional RNN coupled with an encoder decoder network as outlined in Fig. 5.

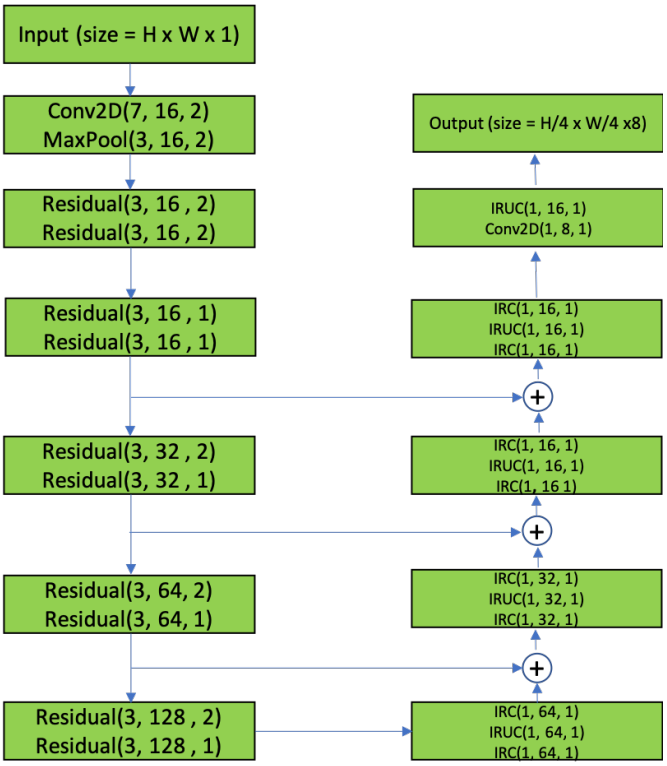

Figure 1: Global Feature Network

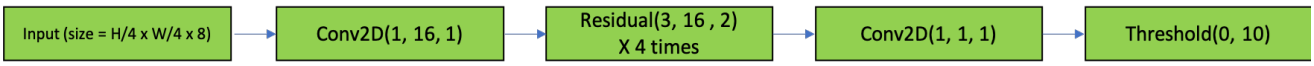

Figure 2: Distance Transform Header

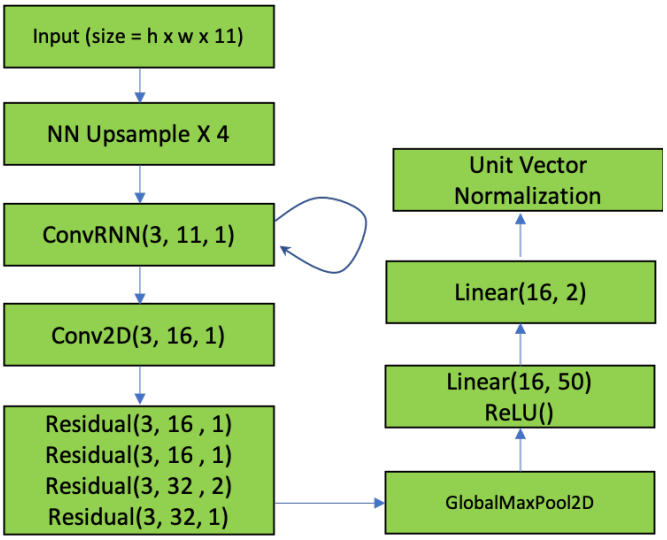

Figure 3: Direction Header

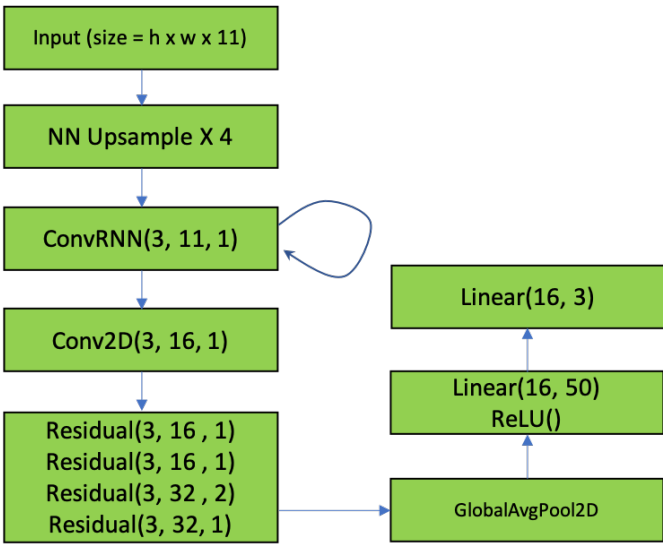

Figure 4: State Header

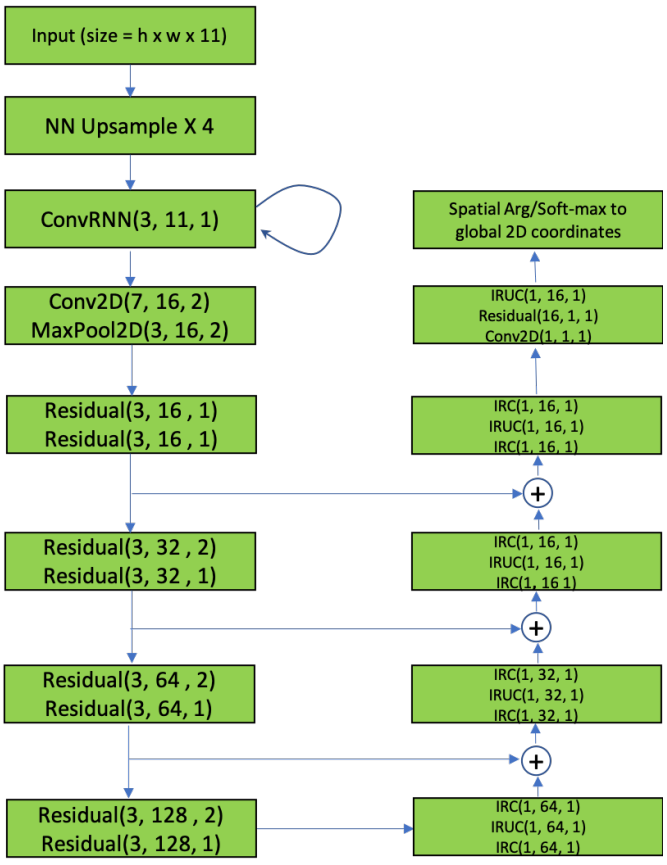

Figure 5: Position Header

2. Training Details

In this section, we describe the training implementation details. For learning, we optimize the following multi-task objective:

$$\mathcal{L}(\theta) = \lambda_1 \ell_{chamfer}(\theta) + \lambda_2 \ell_{cosine}(\theta) + \lambda_3 \ell_{focal}(\theta) + \lambda_4 \ell_{dt}(\theta) \tag{1}$$

where θ are the parameters of our neural network. We utilize the validation set to obtain a best set of hyperparameters as $\lambda_1 = 1, \lambda_2 = 100$ and $\lambda_3 = \lambda_4 = 10$. Since each image is of dimension 1200×8000 pixels and our model comprises of three convolutional RNNs, we are able to fit only one example in an NVIDIA 1080 Ti GPU during training. We trained our model on 8 GPUs using the distributed training environment Horovod [**?**] for 12 hours. We used the AMSgrad [**?**] optimizer with the learning rate of 0.001. We choose the final model based on the lowest precision and recall on the validation set.

During training, we provide all the GT initial vertices $\{v_{\text{init}} = (\theta_{\text{init}}, x_{\text{init}}, s_{\text{init}})\}$ with a small Gaussian perturbation on the initial positions x_{init}. Next, given a parent node $v_{\mathcal{P}(i)}$, we predict the components of the next vertex $\{v_i = (\theta_i, x_i, s_i)\}$. We replace all the $\arg\max$ operators with $softmax$. The ground truth for θ_i is generated on the fly by projecting $x_{\mathcal{P}(i)}$, the anchor for the current Region of interest (RoI), onto the closest lane boundary and computing the turning function at that point. The GT for the categorical random variable s_i is obtained also on the fly by determining whether the current RoI contains a fork or a terminating point and setting the GT accordingly. If neither topological change falls within the RoI, we set the GT to be the normal state.

While training, we use the predicted direction θ_i anchored at $x_{\mathcal{P}(i)}$ to obtain the next RoI unless $x_{\mathcal{P}(i)}$ is 30 pixels away from the lane boundary, in which case we use the GT direction. Moreover, we concatenate the GT states to each header during training. For training the convolutional RNN components of the direction, state and position header, we backpropagate 5 timesteps through time to avoid memory issues. We run the RNN until either the vertex position falls outside the image, the rotated RoI contains a GT terminate state or the RNN runs for 30 timesteps.

At test time, we use the predicted initial vertices, the directions and the states to infer the state of the DAG.

3. Qualitative Results

In Figs. 6, 7, 8 and 9 we demonstrate our predictions on the first highway. In Fig. 10 and 11 we showcase the results of training only on one highway and testing on another highway 1000 km away.

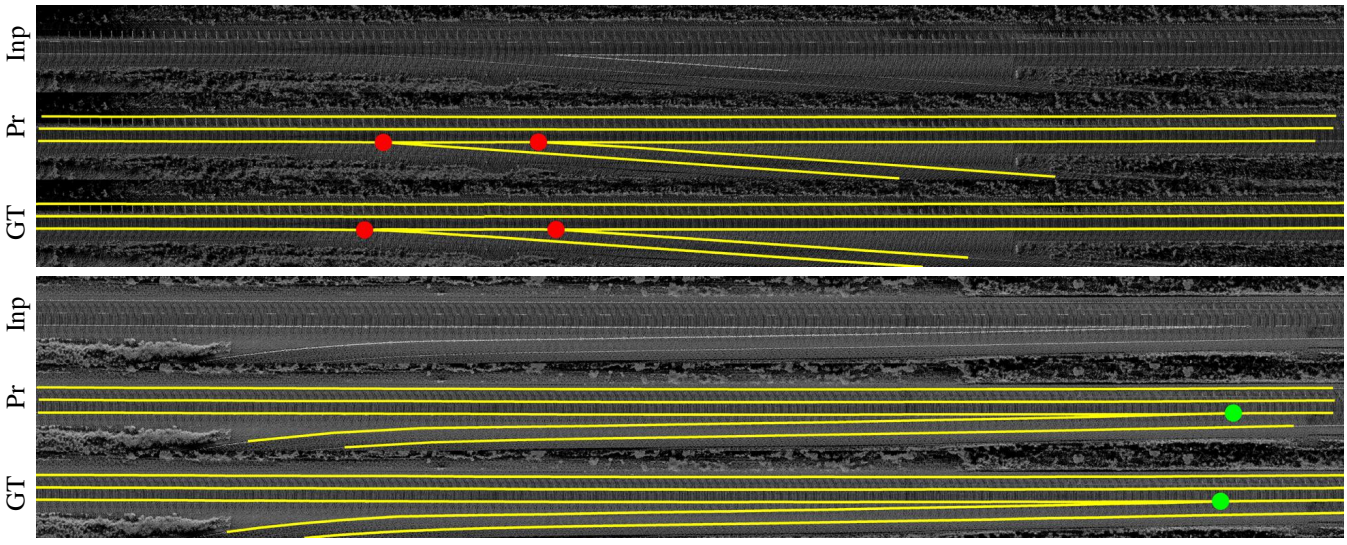

Figure 6: Qualitative results where we showcase the Input Lidar Image (Inp), the predictions (Pr) and the Ground Truth (GT).

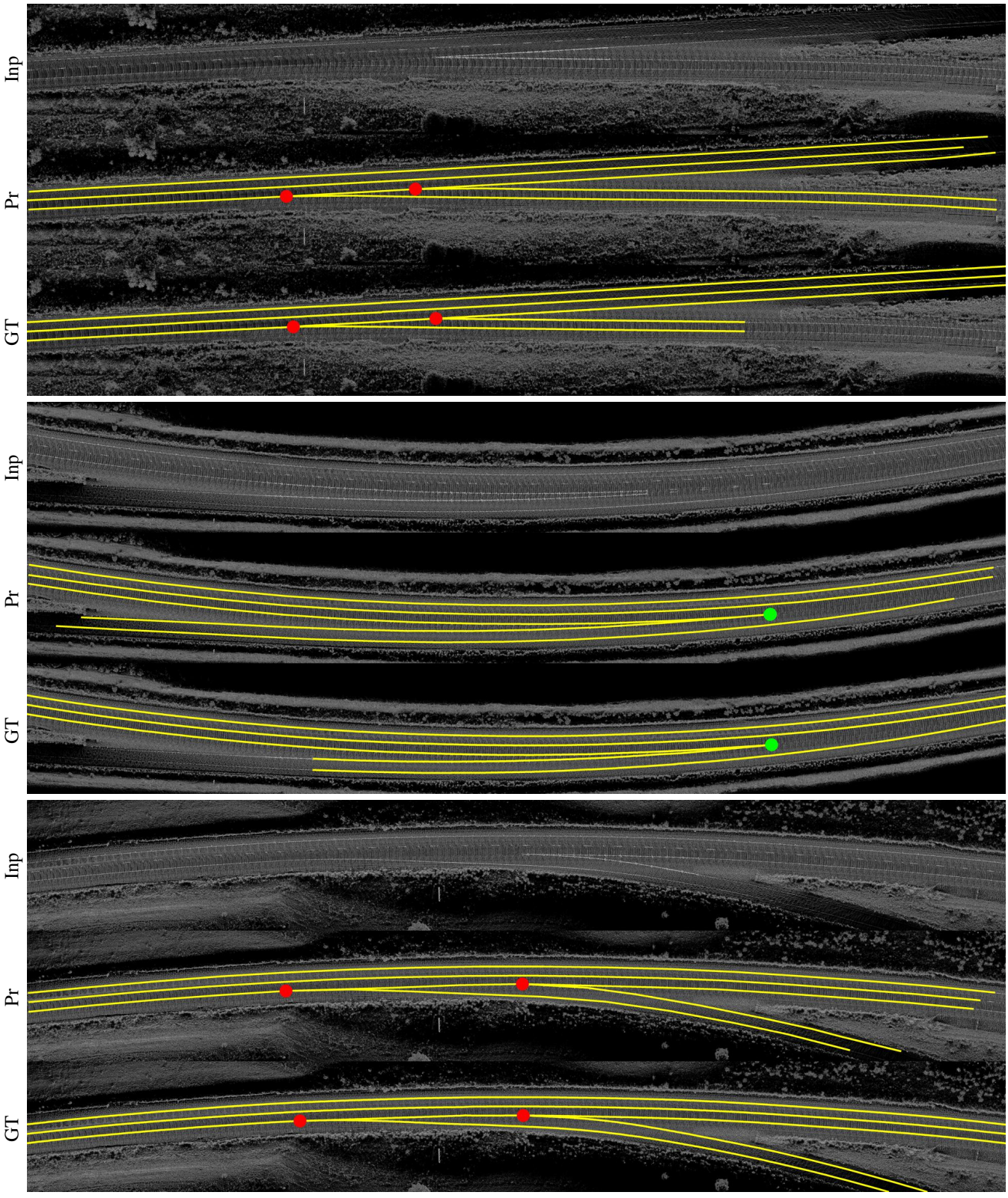

Figure 7: Qualitative results where we showcase the Input Lidar Image (Inp), the predictions (Pr) and the Ground Truth (GT).

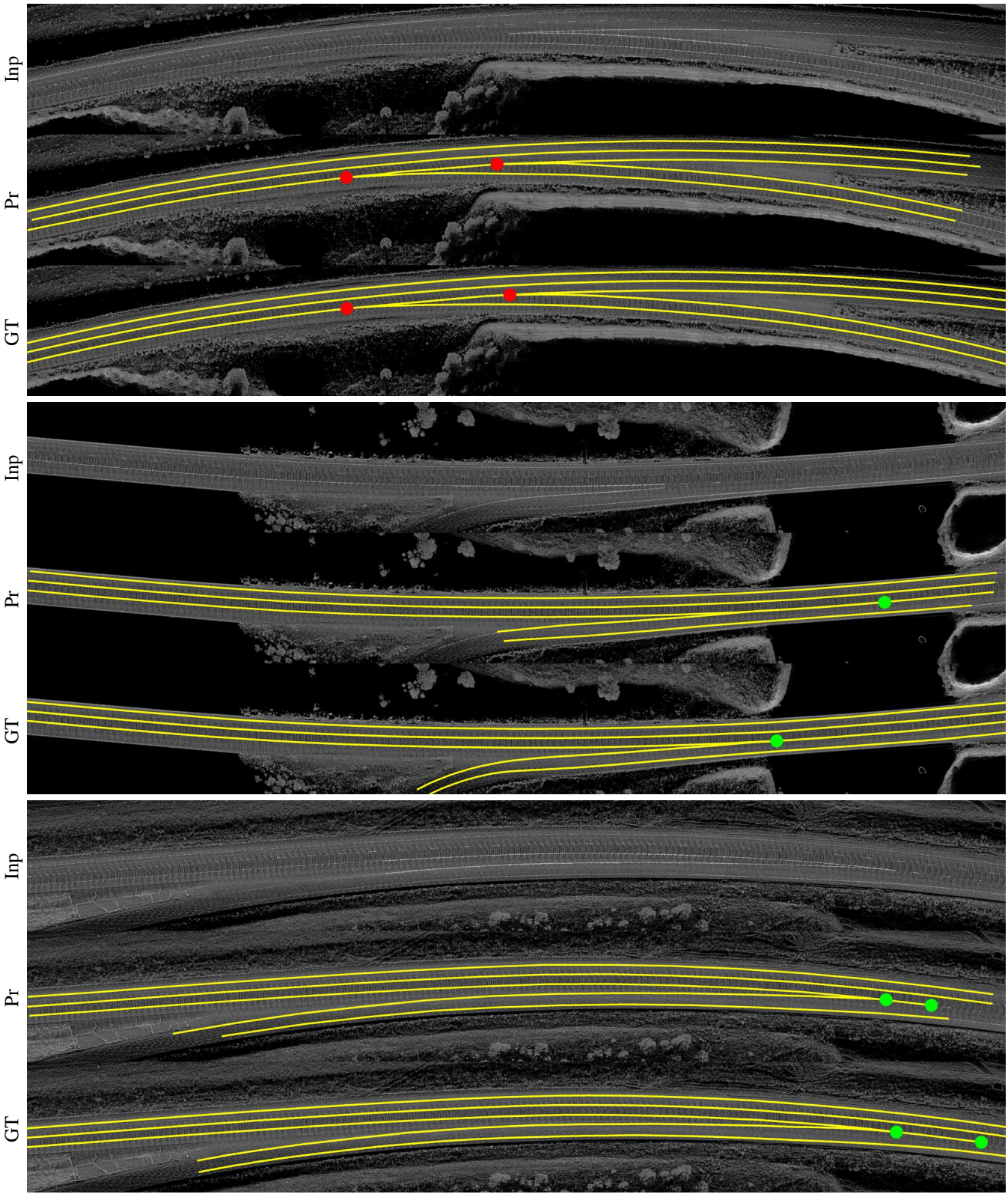

Figure 8: Qualitative results where we showcase the Input Lidar Image (Inp), the predictions (Pr) and the Ground Truth (GT).

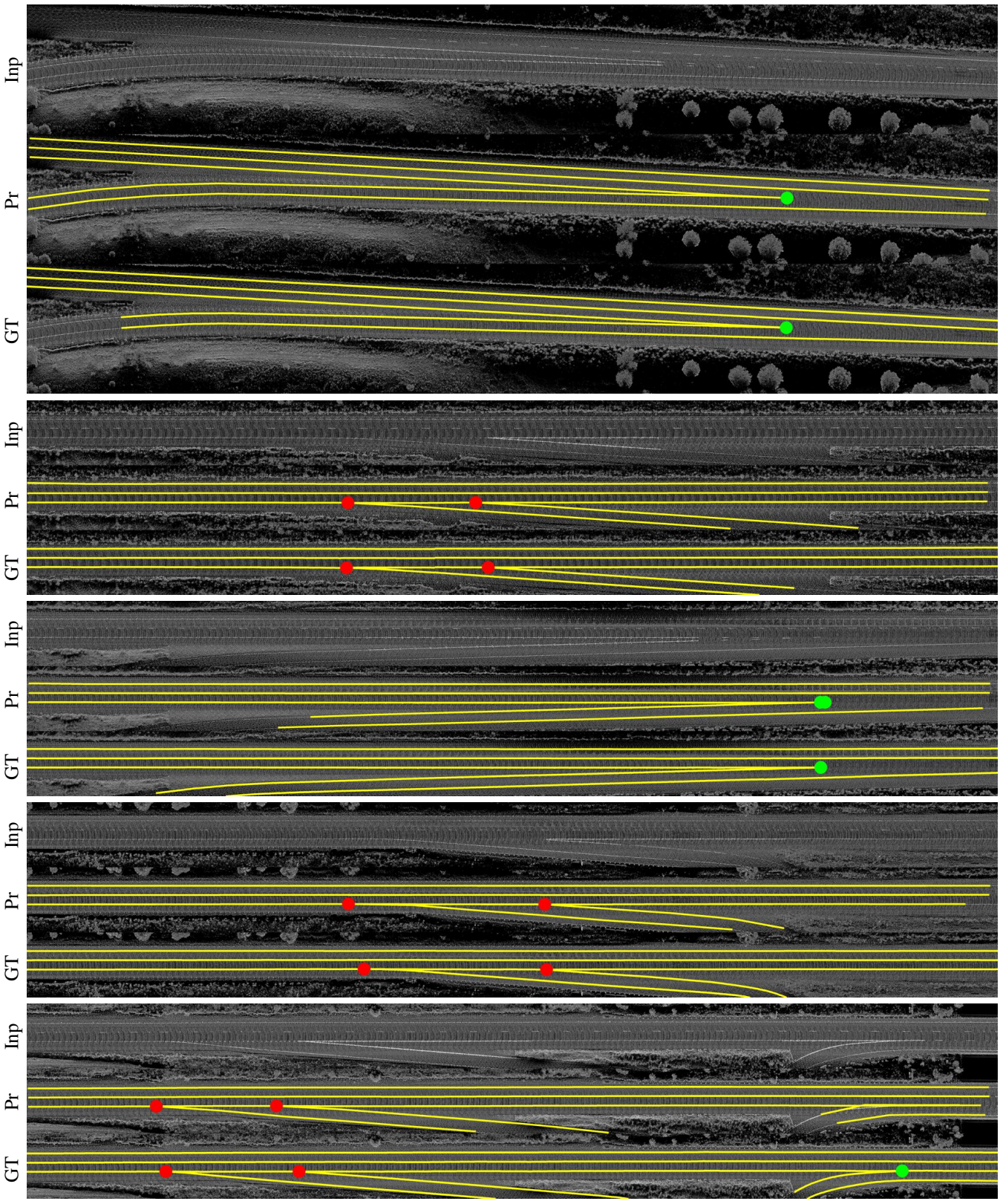

Figure 9: Qualitative results where we showcase the Input Lidar Image (Inp), the predictions (Pr) and the Ground Truth (GT).

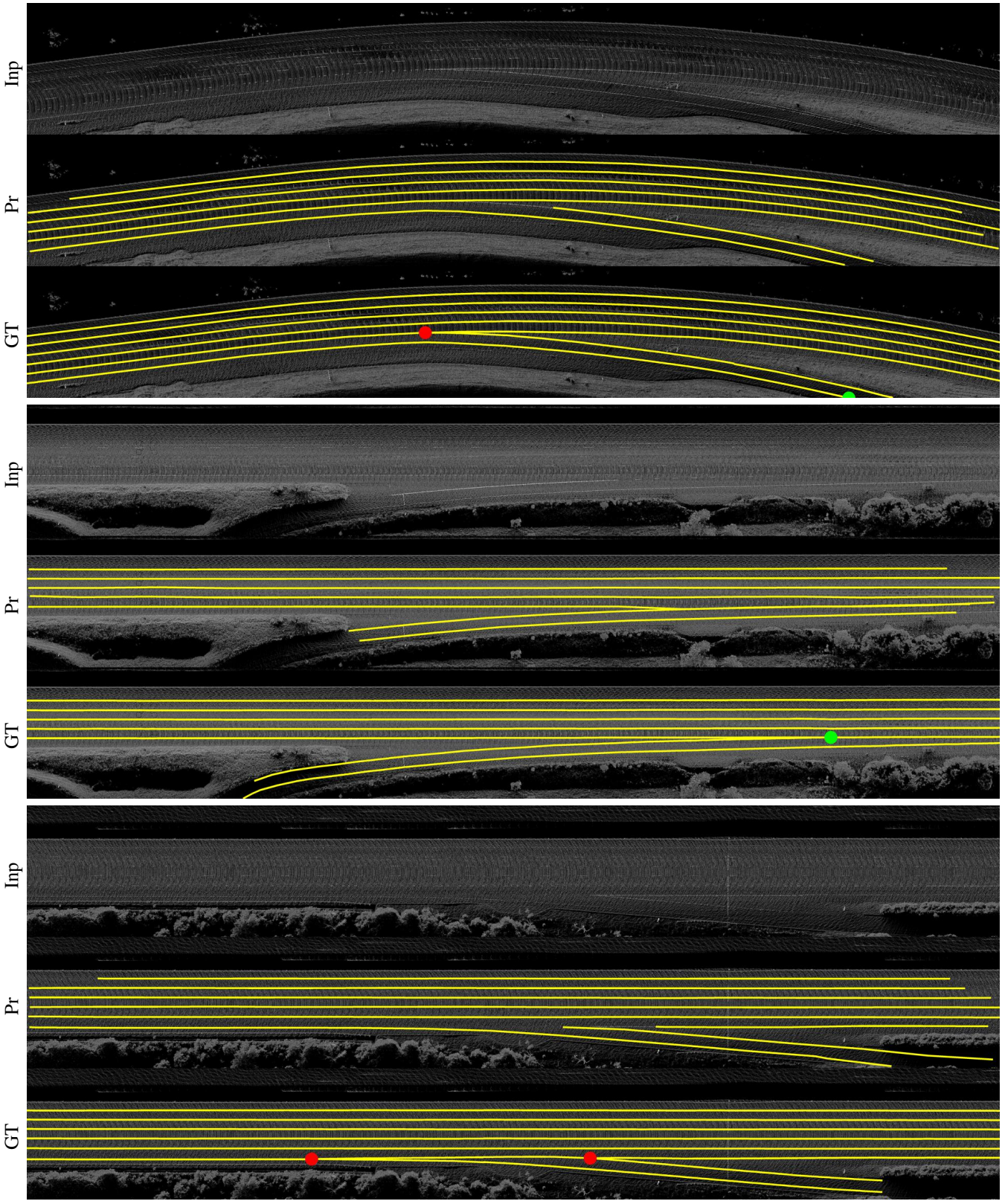

Figure 10: Generalization from training on one highway and testing on another 1000 km away. we showcase the Input Lidar Image (Inp), the predictions (Pr) and the Ground Truth (GT).

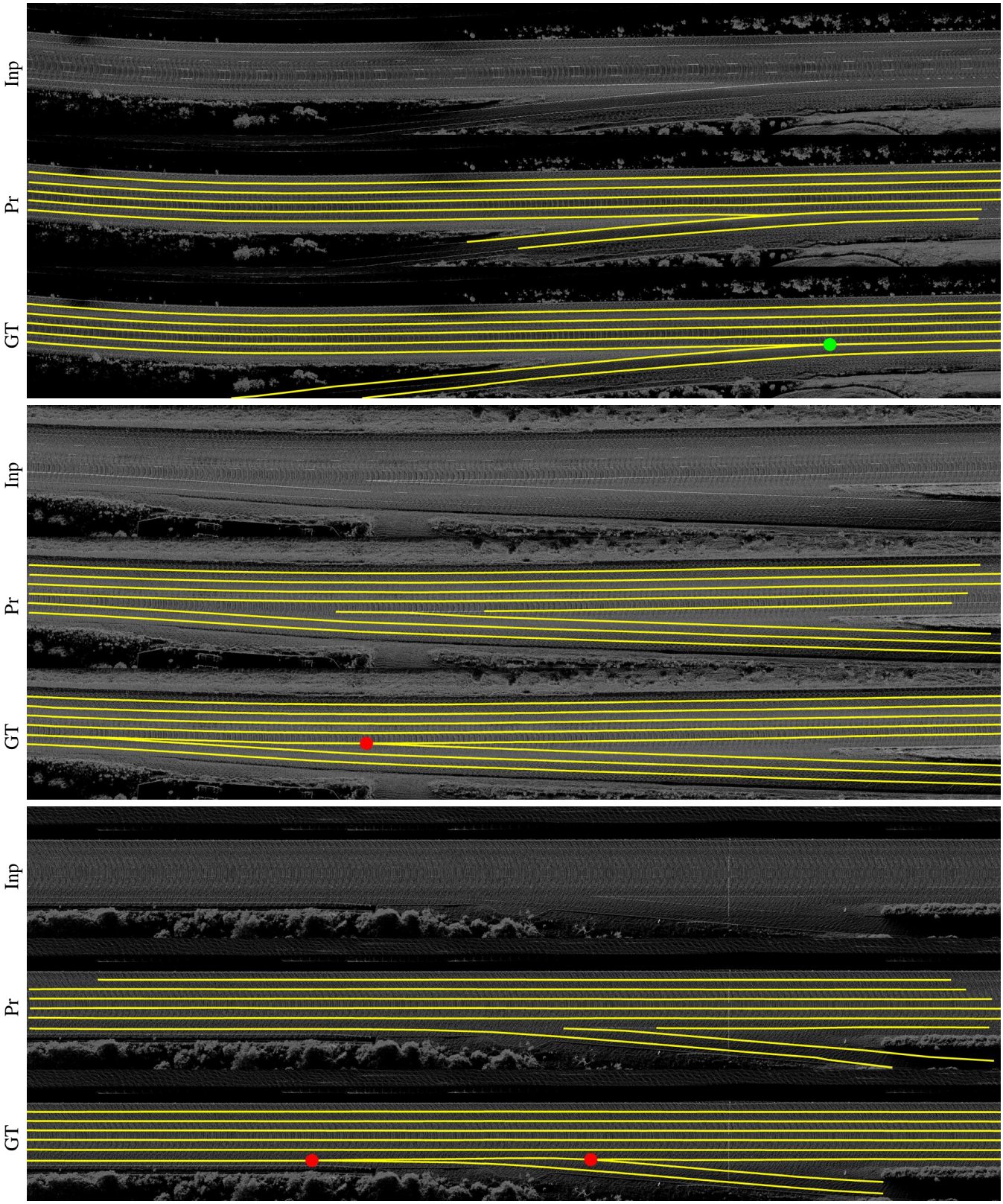

Figure 11: Generalization from training on one highway and testing on another 1000 km away. we showcase the Input Lidar Image (Inp), the predictions (Pr) and the Ground Truth (GT).

References