Appendix to GAS: Generating Fast & Accurate Surrogate Models for Autonomous Vehicle Systems

Keyur Joshi*, Chiao Hsieh†,*, Sayan Mitra*, Sasa Misailovic*
* University of Illinois Urbana-Champaign † Kyoto University
{kpjoshi2, mitras, misailo}@illinois.edu hsieh.chiao.7k@kyoto-u.ac.jp

Appendix A
Crop-Monitor: real-world error distribution and perception model comparison

Figure 1 shows the error distribution of the ResNet-18 networks used by the Crop-Monitor vehicle [1] on the original validation dataset of *real-world* images collected by the vehicle's camera in a cornfield. The left and right plots are for the heading and distance perception error, respectively. The histogram shows the actual error frequency, while the line shows the fitted normal distribution. The distribution closely matches the histogram, indicating that the error of the neural network is normally distributed.

Figure 2 visually compares the neural network output distribution to the distribution predicted by M_{per} for images captured within Gazebo. The red dashed ellipse and the blue solid ellipse show the 3σ confidence boundaries for the neural network output distribution and the distribution predicted by the perception model, respectively. The two distributions closely match each other, especially when the vehicle is near the center and pointing straight ahead.

Appendix B
Crop-Monitor: sensitivity index convergence visualization

Figure 3 shows an example visual comparison of sensitivity indices calculated by GAS and MCS for the sensitivity of the change in state between time step 0 and 1 to the initial state in time step 0 for Crop-Monitor. The input variables include the heading and distance at time step 0, and the two components of the raw sample that is transformed into the perception neural network output distribution sample. The output variables are the heading and distance at time step 1. There is one sensitivity index corresponding to each input/output variable pair. The blue solid line shows the sensitivity calculated analytically by GAS, the blue dotted line shows the sensitivity calculated empirically using the GAS model, and the red dashed line shows the sensitivity calculated empirically using M'_V. In each subplot, the X-Axis shows the number of samples used for empirical estimation, while the Y-Axis shows the calculated sensitivity index. As the number of estimation samples is increased, the sensitivity indices calculated via estimation converge towards those calculated analytically. About 10^6 samples are needed for convergence.

Appendix C
Alternative approaches for constructing the surrogate model

A. Alternative approaches using GPC

The GPC model construction process evaluates the abstracted vehicle model at specific quadrature nodes. Instead of constructing a perception model, we attempted to directly calculate and use the actual perception neural network output distributions at these quadrature nodes. However, if the quadrature nodes change (e.g., by changing the state distribution or order of GPC), then new images must be captured and processed for the new quadrature nodes. In contrast, the perception model can be directly reused as it is agnostic to the quadrature nodes.

We also experimented with replacing only parts of M'_V with GPC as follows: first, we replaced the perception system with the perception model to create M'_V. Then, instead of replacing all of M'_V with a GPC model, we replaced only the vehicle control and dynamics systems (as the perception system had already been replaced by a polynomial model M_{per}). While the accuracy of this approach was the same as our main approach, the partially replaced model was about $3\times$ slower than the fully replaced model during evaluation for all benchmarks.

B. Alternative types of surrogate models

While we focus on GPC surrogate models, GAS also enables the use of other surrogate models by replacing the complex perception system with a perception model. That is, we can train a different type of surrogate model instead of a GPC model. We can then use the alternative surrogate model for state distribution estimation or for sensitivity analysis. We next briefly describe our results for two alternative surrogate models: polynomial regression and neural networks.

We experimented with using standard polynomial regression for creating the surrogate model (as opposed to generating polynomial surrogates via GPC). For training and testing data, we chose points in the safe state space using a Sobol sequence. Column 3 of Table I shows the t-test results for the regression surrogate model. For most benchmarks, the accuracy of the regression surrogate model is slightly lower than that of the GPC surrogate model (Column 2). This is in line with the fact that GPC produces the most optimal polynomial surrogate model for any order ([2][Equation 5.9]) in terms of

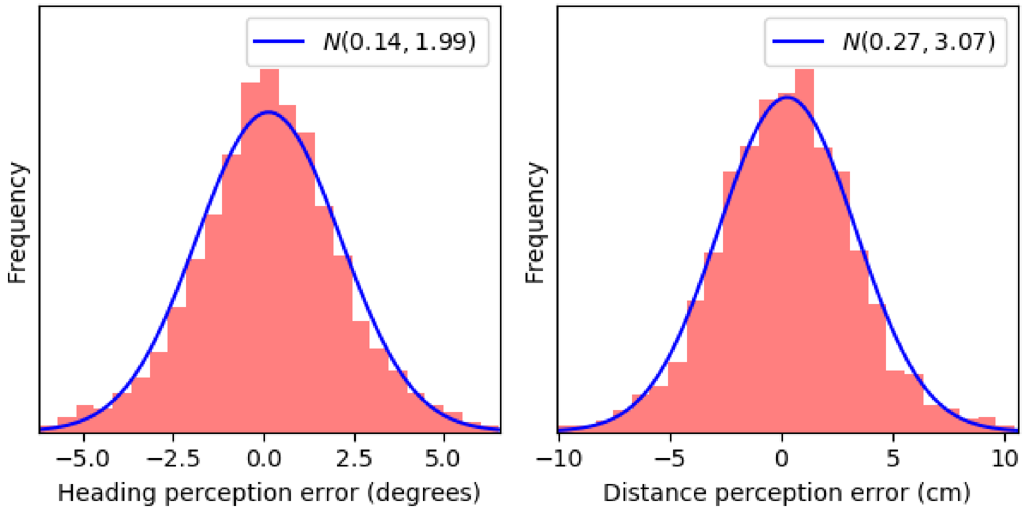

Fig. 1: Error distribution of the neural network for the validation set of *real-world* images.

TABLE I: t-test results among surrogate model candidates

Benchmark	**GPC**	**Poly Reg**	**DNN**
Crop-Monitor	99/100	100/100	$(52 \pm 40)/100$
Cart-Straight	97/100	97/100	$(49 \pm 39)/100$
Cart-Curved	98/100	97/100	$(46 \pm 35)/100$
ACAS-Table	100/100	99/100	$(46 \pm 01)/100$
ACAS-NN	100/100	98/100	$(46 \pm 04)/100$

ℓ_2 error. Further increasing the order of the polynomial does not increase accuracy, but rather causes overfitting. The other safe state probability similarity metrics show similar trends.

We also experimented with using a neural network surrogate model. Once again, we chose points in the safe state space using a Sobol sequence to generate training/testing data. We experimented with various neural network topologies and activations, with the number of parameters being at least the number used by the GPC model. We found that the accuracy of this surrogate model varied widely with the initial seed. Column 4 of Table I shows the t-test results for the NN surrogate model. Due to the dependence on the initial seed, the results are presented in the form (mean $\pm$ std). The standard deviation is high for the first three benchmarks, and the mean is low for all benchmarks. Without the original vehicle model for comparison, it would not be possible to know which random restart produced the most accurate model. In contrast, GPC deterministically produces an optimal polynomial model.

References

[1] A. Sivakumar, S. Modi, M. Gasparino, C. Ellis, A. Velasquez, G. Chowdhary, and S. Gupta, "Learned visual navigation for under-canopy agricultural robots," 2021.

[2] D. Xiu, *Numerical Methods for Stochastic Computations: A Spectral Method Approach*. Princeton Press, 2010.

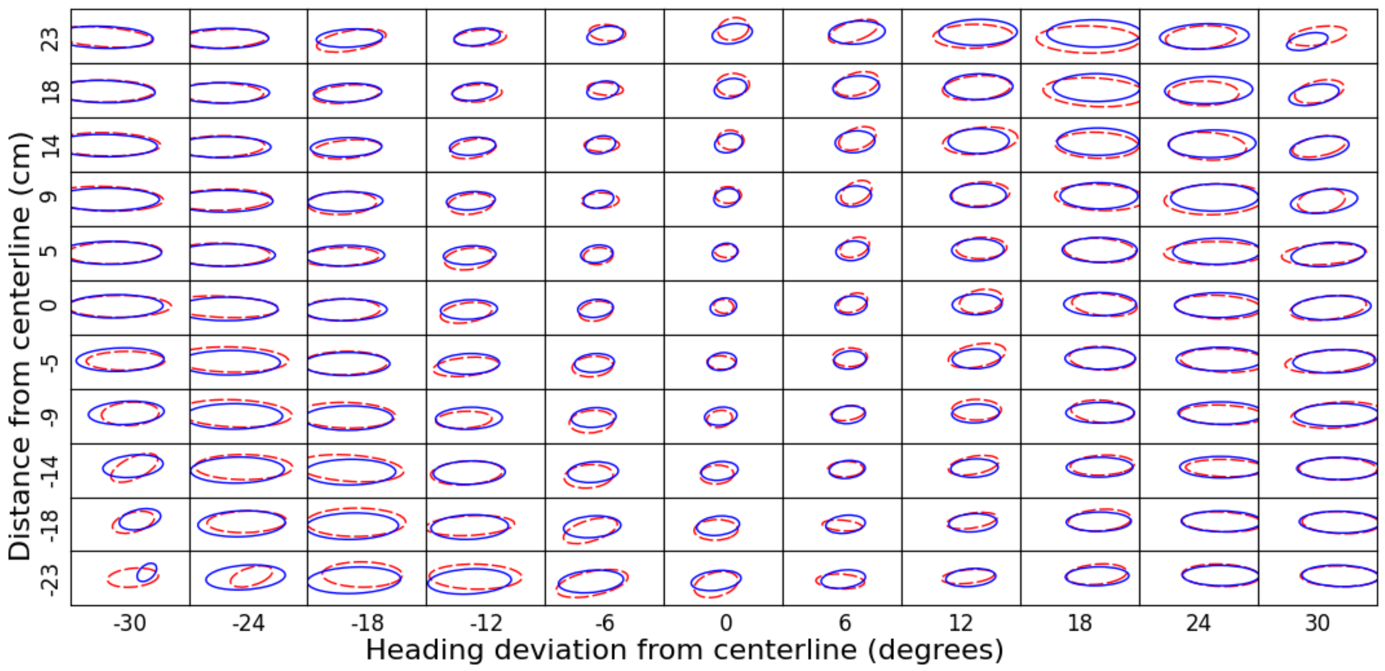

Fig. 2: Comparison of neural network output distribution (dashed red ellipse) to distribution predicted by perception model (solid blue ellipse). Each box represents a distinct ground truth state used to construct the perception model. The X and Y-Axes vary the ground truth heading and distance, respectively.

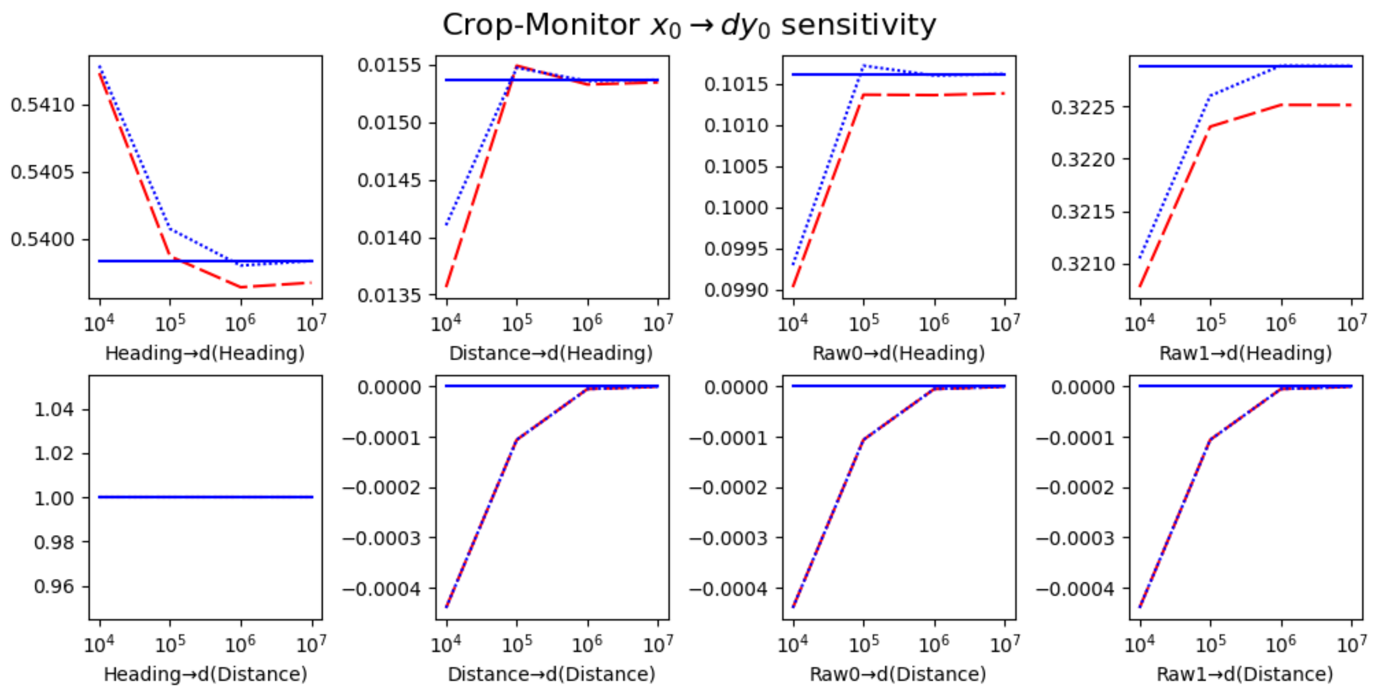

Fig. 3: Calculated sensitivity index comparison. Blue solid: GAS (analytical method), blue dotted: GAS (empirical method), red dashed: MCS.