Industry-track: Challenges in Rebooting Autonomy with Deep Learned Perception

Michael Abraham, Aaron Mayne, Tristan Perez,
Italo Romani De Oliveira, and Huafeng Yu
The Boeing Company
United States
Email: {Aaron.A.Mayne,michael.r.abraham}@boeing.com
Email: {tristan.perez,italo.romanideoliveira,huafeng.yu}@boeing.com

Chiao Hsieh,Yangge Li,
Dawei Sun, and Sayan Mitra
University of Illinois at Urbana-Champaign
United States
Email: {chsieh16,li213,daweis2,mitras}@illinois.edu

***Abstract*—Deep learning (DL) models are becoming effective in solving computer-vision tasks such as semantic segmentation, object tracking, and pose estimation on real-world captured images. Reliability analysis of autonomous systems that use these DL models as part of their perception systems have to account for the performance of these models. Autonomous systems with traditional sensors have tried-and-tested reliability assessment processes with modular design, unit tests, system integration, compositional verification, certification, etc. In contrast, DL perception modules relies on data-driven or learned models. These models do not capture uncertainty and often lack robustness. Also, these models are often updated throughout the lifecycle of the product when new data sets become available. However, the integration of an updated DL-based perception requires a reboot and start afresh of the reliability assessment and operation processes for autonomous systems. In this paper, we discuss three challenges related to specifying, verifying, and operating systems that incorporate DL-based perception. We illustrate these challenges through two concrete and open source examples.**

I. Introduction

The first aircraft autopilot was developed by Sperry Corporation in 1912 [1]. The autopilot connected a gyroscopic heading sensor and an altitude sensor to hydraulic elevators and rudder. This enabled the aircraft to fly straight and level on a compass course without a pilot's attention, thereby greatly reducing the pilot's workload. As newer sensor technologies became viable, they have been used to create new capabilities in manned and unmanned flight control systems. Deep learning (DL) has shown dramatic improvements in a number of vision-based detection and estimation tasks such as target and object detection [2], [3], lane detection [4], [5], distance estimation [6], pose estimation [7], [8], and free-space estimation. These DL models promise to help create the next level of autonomy for air vehicles in existing and new environments [9]. However, this does not come without associated challenges.

Embedded and autonomous systems with traditional sensors have tried and tested development processes and support tools [10]–[12]. Several steps in this process like requirements analysis, model-based development, unit testing, and system integration, are not compatible with the design and analysis processes for DL-based perception subsystems. For instance, DL models are evaluated against test data sets, and while it is accepted that these models should be aware of uncertainty when shown new data [13], they do not come with specifications that are supposed to be met in real world deployments. The models also exceed the capabilities of existing formal testing and verification approaches for assessing reliability. And, these models have to be maintained and improved throughout the product lifecycle via data collection and domain.

In this paper, we discuss three challenges related to specifying, verifying, and operating systems that incorporate DL-based perception. First, we introduce some terminology and two concrete autonomous systems that use deep learning-based models for perception. Executable code for experimenting with these systems is available at https://publish.illinois.edu/aproximated-abstract-perception/. Then, we discuss the three challenge problems in specification, analysis, and runtime assurance.

II. Vision-based Autonomous Systems

A. Drone Racing

The *autonomous drone racing* requires participants to develop the software for an autonomous drone that navigates through a race course using computer vision. A prominent version of this competition is the AlphaPilot Challenge [14] hosted by Lockheed Martin and the Game of Drones [15] by Microsoft. Both competitions define the vision-based control task as follows. The rough race course layout and gate positions are provided beforehand, however the autonomous system has to use a vision-based control strategy to go through the actual gates, as fast as possible. Here we discuss a drone racing pipeline as implemented in the AirSim simulator [15].

Fig. 1 shows a simplified version of the autonomous system's pipeline for tracking gates. The details of the subsystems for planning and control are omitted. The *perception subsystem* detects the gate using front camera images and estimates the relative position and orientation of the drone to the gate. The *tracking controller* computes control commands such as waypoints, velocity, or thrust to fly the drone through the gates. The perception subsystem includes both the gate detection and pose estimation. This subsystem could in general involve both deep learning models and classical functions (e.g., for projective transforms, scaling, etc.)

In addition to the perception and the control subsystems, the closed-loop system has the dynamics of the drone and the camera in a simulated environment. In the actual drone, the camera produces the image, and in the simulation a model of

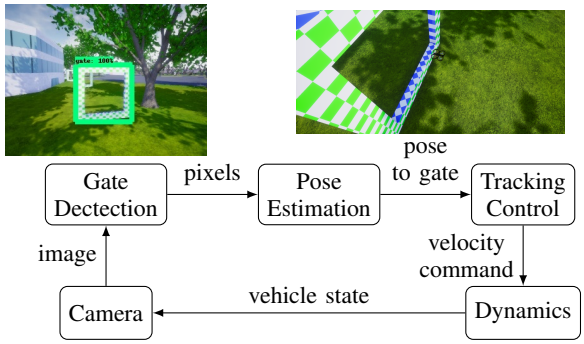

Fig. 1. Sub-systems in the drone racing autonomous system. Control decisions are made based on the gate pose estimates obtained from a DL-based perception pipeline.

the camera renders an image as seen by the camera. In either case, the image depends on many *environmental parameters*. These are factors such as lighting, fog, albedo, lens distortions, etc., that are not controlled by the autonomy software and are possibly not observable either. The range of environmental parameter variations that the system is designed to tolerate is called its *operational design domain (ODD)* [16]. There may also be unknown factors that impact the image, and therefore, the output of the perception subsystem.

B. Swarm Formation

The *Swarm formation* problem requires a collection of drones to create a sequence of formations using vision-based relative positioning. This is an abstracted version of various distributed coordination tasks such as formation flight, surveillance, and cooperative maintenance. Flocking and formation control algorithms have been extensively studied [17], [18]. There are many consensus-based algorithms for swarm formation control, and detailed characterization of their performance under the assumption that precise relative positions are available. Since vision-based control cannot provide exact relative position information, this problem surfaces the need for understanding the impact of *imprecise* state (position) estimates on distributed control.

Fig. 2 shows the system pipeline for each individual drone in the swarm. The perception subsystem here uses at least a pair of images—one from the ego drone and another from a neighboring drone. Common pixels (or features) in these images identified and *associated* using DL models. The associated features are then used to estimate the relative position of the drones by solving a set of linear equations [19]. The formation controller then computes velocity commands to fly the drone and achieve the desired formation shape. As in the racing problem, the closed-loop system involves vehicle dynamics and camera sensors. In addition to the environmental factors in the racing problem, the state estimates are also influenced by the camera mounting angles and focal length distortions.

Scope: Before proceeding, we note the challenge problems discussed here arise in architectures where DL models are integrated as sensors or state estimators in a modular pipeline (as shown in the above figures). Other autonomy architectures, such as end-to-end pixels-to-torque learning (e.g., as in [20]),

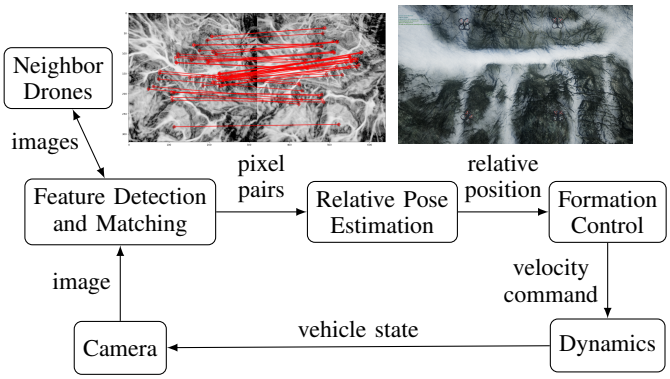

Fig. 2. Simplified drone formation flight systems with vision.

share some of these challenges, but we are not addressing them in this paper.

III. Challenge 1: Perception Datasheets

Specifications of the perception subsystem are useful for communication, testing, modular reasoning, and the design of downstream components. Abstractly, the perception subsystem is a state estimator, no different from GPS or a temperature sensor, and likewise it should be specified in terms of its accuracy, resolution, update frequency, operating tolerances, etc. An ML-powered perception subsystem should come with a data sheet just like ordinary sensors [21].

The accuracy metrics used for DL models vary with tasks. For example, mean average precision (mAP) is used for multi-class object detection [2], [3]; mean intersection over union (mIoU) is used for multi-class semantic segmentation [22]; root mean squared error (RMSE) is used for depth estimation [6]; average distance (ADD) is used for object pose estimation [7], [8] which is the relevant metric for the perception subsystem racing, which estimates the pose of the next gate relative to the drone. These metrics are usually defined in terms of training and test set data, and do not readily translate to new images seen in the deployed system. Relatedly, the DL models do not come with guarantees or recommendations about tolerable environmental parameter variations. Thus, the challenge in specifying deep-learning based perception subsystems can be summarized as follows:

> *Create datasheets for the perception subsystems for pose estimation including accuracy and expected domain of operation.*

Nature of the challenge and approaches: The perception subsystem estimates a quantity or a state—say, relative pose—and all of the above metrics can be computed provided we have the ground truth values of the quantity to be estimated. However, *ground truth pose value* is not always available nor computable from the input image. Only in very special cases, we can derive this relationship using geometry of image formation, but this relationship will not be robust to vicissitudes of real world images with shadows, occlusions, flares, etc. For real images, getting the ground truth involves getting correctly labeled data. For synthetic or simulator generated data, the ground truth is readily available. For pose, depth,

and speed estimation ground truth can be estimated from alternative sensors (GPS, lidar, and radar). Various datasets for this purpose are available, such as KITTI [23] and the Waymo open dataset [24].

Defining valid domain of operation: An ordinary sensor's specification not only spells out its performance parameters like accuracy (with respect to ground truth), resolution, and the range over which it is supposed to work, but it also specifies the environmental variations it is supposed to tolerate. This GPS sensor's datasheet [21] states that it works below an altitude of $18km$ and in the temperature range of $-40°$C to $85°$C. Motivated by such engineering datasheets, Gebru et al. [25] have proposed that similar types of datasheets be published for datasets used for training general ML models. They argue that documenting the motivation, composition, collection process, and recommended uses of datasets will improve communication between creators and consumers of datasets, and encourage transparency and accountability in machine learning.

For ML-based perception models, we need the datasheet to specify the variations on environmental conditions under which the model's accuracy is guaranteed, or at least guaranteed with high probability. Images depend on many environmental conditions such as lighting, albedo, occlusions, and distortions that may not be represented in the datasets. For autonomous systems, the notion of tolerances have to be extended to cover behaviors of other agents in the environment, and all this is covered under the umbrella term *operational design domains (ODDs)* [16]. Other factors adding to the challenge are the so called *unknown unknowns*, i.e., unidentified factors that impact the image and the perception subsystem but only emerge in real world deployments. Identifying and spelling out the tolerable ranges for the known parameters is a starting point for creating a datasheet or a contract for the perception subsystem. Impacts of unidentified factors could potentially be managed through out-of-distribution detection and continuously updating the perception models and we discuss this more in Section V.

IV. Challenge 2: Testing and Verification with Perception

Replacing the DL-based perception subsystems in Fig. 1 and Fig. 2 with corresponding perfect state estimators, the closed-loop systems become standard cyber-physical systems. Many testing and verification techniques are available for such systems [11], [12]. However, DL-based perception breaks the continuity and differentiability assumptions baked into many of these techniques. This leads to the second challenge:

> *Test and formally verify the racing and swarm formation systems, assuming that the perception subsystems satisfy the specifications in the datasheet.*

Current state of the art on simulation-based testing and falsification relies on simulating the system behavior using synthetic images. The difficulty is to first formally describe the numerous scenarios, i.e., combinations of environmental parameter values, within the operational design domain, and second to systematically and efficiently simulate and search through all combinations and construct a reasonable scenario that falsify the system. For the drone racing as an example, it is unrealistic to expect the system to function in a scenario that the gate is completely obstructed by a tree (out of ODD), but the system should still work when the gate is not obstructed but cast with the shadow of the leaves (inside ODD). How to implement the scenario generation and search within ODD for simulation remains a challenge.

Dually, the formal verification aims to prove the correctness of the system given the contract of perception subsystems. Using the contract of perception subsystem greatly reduces the complexity and make the system verification more tractable. In addition, if only a small part of the perception model changes and the contract is only slightly changed, then it is unnecessary to verify the whole pipeline over the entire space. There is an opportunity for exploiting incremental verification to further avoid scalability issues. However, the components of the perception subsystem, such as ReLU or sigmoid activation functions or max pooling layers, suggests that the contract of the perception is likely non-differentiable and nonlinear. Solving this problem would require the verification of hybrid system with nonlinear dynamics.

Related approaches: In falsification, the Scenic [26] language for autonomous driving system can describe abstract traffic scenarios in 2D geometry with probability distributions, and VerifAI [27] provides various sampling strategies to search for scenarios. Our drone racing and formation examples introduce much higher complexity in describing 3D scenarios and require better search strategies.

Plenty of recent works focus on system verification directly with neural networks instead of contracts [28]. These neural networks however are much smaller compared to DL-based perception. Katz et al. [29] trains a simple generative adversarial network (GAN) to represent the perception subsystem. [30] infers and verifies the system with an abstract approximate perception (AAP). These two works shows promising results how an abstraction or a contract can help address the scalability issue due to DL-based perception.

V. Challenge 3: Runtime Assurance with Perception

The third challenge is about perception error detection and handling. An *error* is an event where the accuracy of the computed output from the perception subsystem violates the specification. There are two related problems.

Error detection is the problem of determining that an error event has occurred. This problem of error is a special case of the broader problem of anomaly detection [31] and the problem detecting of out of distribution events and distribution shifts [32]. Without ground truth direct error detection is impossible, but probabilistic detection based on alternative sensors and correlations is possible. For example, if a vision-based pose estimator is supposed to work with certain accuracy in the illumination range of 3-1000 Lux, then potential error in this estimator can be flagged by another sensor for gross

illumination levels. The overall reliability of the system will then also depend on the specifications of the illumination sensor, and so on.

An autonomous system's decision logic has to decide when a perception error is serious enough to switch the overall system to a lower-capability or a safe recovery mode. A balance has to be struck in this decision between unnecessarily low-performance owing to spurious or unimportant perception errors, and the risk of missing important perception errors that cause system-level failures. The basic version of decision problem, without the complications of ML-based sensing, is called the *runtime assurance problem (RTA)* [33]–[35]. ML-based perception, brings a complication to RTA as state estimation errors now have larger or unknown variances.

> *Design the detector and optimal runtime decision logic for protecting the racing and swarm autonomous systems against out of distribution perception errors.*

Related approaches: There is a rich and growing body of work on runtime assurance [33]. A notable clustering-based approach for detecting anomalous perception is presented in [36]. To the best of our knowledge, design of runtime assurance with perception has not been addressed so far.

ACKNOWLEDGMENT

The authors would like to acknowledge the constructive feedback from the anonymous reviewers. The Illinois researchers were supported by research grants from the National Science Foundation (Award number NSF-SHF-2008883) and from the Boeing Company.

REFERENCES

[1] "Now–the automatic pilot," *Popular Science Monthly*, p. 22, Feb. 1930. [Online]. Available: https://books.google.com/books?id=4ykDAAAAMBAJ&pg=PA22#v=onepage&q&f=false

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 580–587.

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 779–788.

[4] D. Neven, B. D. Brabandere, S. Georgoulis, M. Proesmans, and L. V. Gool, "Towards end-to-end lane detection: an instance segmentation approach," in *2018 IEEE Intell. Veh. Symp.*, 2018, pp. 286–291.

[5] Z. Wang, W. Ren, and Q. Qiu, "Lanenet: Real-time lane detection networks for autonomous driving," 2018. [Online]. Available: https://arxiv.org/abs/1807.01726

[6] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 270–279.

[7] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes," in *Proc. Robotics: Science and Systems*, Pittsburgh, Pennsylvania, June 2018.

[8] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *Asian Conf. Comput. Vis.* Springer, 2012, pp. 548–562.

[9] D. J. Fremont, J. Chiu, D. D. Margineantu, D. Osipychev, and S. A. Seshia, "Formal Analysis and Redesign of a Neural Network-Based Aircraft Taxiing System with VerifAI," in *Proc. 32nd Int. Conf. Computer Aided Verification*, 2020, pp. 122–134.

[10] E. A. Lee and S. A. Seshia, *Introduction to Embedded Systems*, 2nd ed. Cambridge, Massachuetts: MIT Press, 2017.

[11] R. Alur, *Principles of Cyber-Physical Systems*. MIT Press, 2015.

[12] S. Mitra, *Verifying Cyber-Physical Systems: A Path to Safe Autonomy*. Cambridge, MA, USA: MIT Press, 2021.

[13] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané, "Concrete problems in AI safety," 2016. [Online]. Available: https://arxiv.org/abs/1606.06565

[14] (2019) Alphapilot - lockheed martin ai drone racing innovation challenge. Lockheed Martin. [Online]. Available: https://www.herox.com/alphapilot

[15] R. Madaan, N. Gyde, S. Vemprala, M. Brown, K. Nagami, T. Taubner, E. Cristofalo, D. Scaramuzza, M. Schwager, and A. Kapoor, "AirSim Drone Racing Lab," in *Proc. NeurIPS 2019 Competition and Demonstration Track*, ser. Proc. Machine Learning Research, vol. 123. PMLR, Dec. 2020, pp. 177–191.

[16] P. Koopman and F. Fratrik, "How many operational design domains, objects, and events?" in *SafeAI@ AAAI*, 2019.

[17] R. Olfati-Saber, "Flocking for multi-agent dynamic systems: Algorithms and theory," *IEEE Trans. Autom. Control*, vol. 51, no. 3, pp. 401–420, 2006.

[18] M. Mesbahi and M. Egerstedt, *Graph theoretic methods in multiagent networks*. Princeton University Press, 2010.

[19] D. Nistér, "An efficient solution to the five-point relative pose problem," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 26, no. 6, pp. 756–770, 2004.

[20] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik, "Masked Visual Pre-training for Motor Control," 2022. [Online]. Available: https://arxiv.org/abs/2203.06173

[21] R. F. Solutions, "Low-power high-performance and low-cost 65 channel SMD GPS module," in *Data Sheet Version 1.4*, 2009. [Online]. Available: https://docs.rs-online.com/c0e9/0900766b80df94d1.pdf

[22] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3431–3440.

[23] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Robot. Res. (IJRR)*, 2013.

[24] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recog.*, 2020, pp. 2446–2454.

[25] T. Gebru, J. Morgenstern, B. Vecchione, J. W. Vaughan, H. Wallach, H. D. III, and K. Crawford, "Datasheets for Datasets," *Commun. ACM*, vol. 64, no. 12, p. 86–92, Nov. 2021.

[26] D. J. Fremont, E. Kim, T. Dreossi, S. Ghosh, X. Yue, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, "Scenic: a language for scenario specification and data generation," *Machine Learning*, Feb. 2022.

[27] T. Dreossi, D. J. Fremont, S. Ghosh, E. Kim, H. Ravanbakhsh, M. Vazquez-Chanlatte, and S. A. Seshia, "VerifAI: A Toolkit for the Formal Design and Analysis of Artificial Intelligence-Based Systems," in *Proc. 31st Int. Conf. Computer Aided Verification*, 2019, pp. 432–442.

[28] M. Everett, "Neural network verification in control," in *2021 60th IEEE Conf. Decision Control (CDC)*, 2021, pp. 6326–6340.

[29] S. M. Katz, A. L. Corso, C. A. Strong, and M. J. Kochenderfer, "Verification of image-based neural network controllers using generative models," in *Proc. 40th IEEE/AIAA DASC*, 2021, pp. 1–10.

[30] A. Anonymous, "Verifying controllers with vision-based perception using safe approximate abstractions," Accepted by EMSoft 2022, 2022.

[31] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, July 2009.

[32] S. Liang, Y. Li, and R. Srikant, "Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks," in *Proc. 6th Int. Conf. Learning Representations (ICLR)*. OpenReview.net, 2018.

[33] J. D. Schierman, M. D. DeVore, N. D. Richards, and M. A. Clark, "Runtime assurance for autonomous aerospace systems," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 12, pp. 2205–2217, 2020.

[34] A. Aiello, J. Berryman, J. Grohs, and J. Schierman, "Run-time assurance for advanced flight-critical control systems*," in *AIAA Guidance, Navigation, and Control Conference*. American Institute of Aeronautics and Astronautics, 2010.

[35] P. Nagarajan, S. K. Kannan, C. Torens, M. E. Vukas, and G. F. Wilber, "Astm f3269-an industry standard on run time assurance for aircraft systems," in *AIAA Scitech 2021 Forum*, 2021, p. 0525.

[36] Y. Yang, R. Kaur, S. Dutta, and I. Lee, "Interpretable detection of distribution shifts in learning enabled cyber-physical systems," in *Intl. Conf. on Cyber-Physical Systems*. IEEE, May 2022.