

Cross-Robot Evaluation of CNN Backbones for JetBot Lane Following in Low-Light Environments

Donggeon Lee

*Department of Electrical and
Computer Engineering
Sungkyunkwan University
Suwon, Republic of Korea
ldk0012@skku.edu*

Myeongjun Kim

*Department of Electrical and
Computer Engineering
Sungkyunkwan University
Suwon, Republic of Korea
gonihp1398@skku.edu*

Hyoseok Jang

*Department of Bio-Mechatronic
Engineering
Sungkyunkwan University
Suwon, Republic of Korea
sweater7719@skku.edu*

Jo Woon Chong[†]

*School of Electrical and
Electronic Engineering
Sungkyunkwan University
Suwon, Republic of Korea
jwchong@skku.edu*

Abstract—Small-scale lane-following studies often report of-line prediction error or results from a single robot, leaving transfer between physical robots and actual driving behavior less examined. This paper compares ResNet-18, MobileNetV2, ShuffleNetV2 1.0×, and GoogLeNet on three Jetson Nano-based JetBots operating on an indoor track with illuminance ranging from 1 to 30 lux. A three-fold leave-one-robot-out protocol yields 12 model-fold evaluations, with each model developed on two robots and evaluated on the third. Closed-loop tests measure lap time, steering behavior, onboard inference time, and control-loop latency, whereas open-loop video analysis measures coordinate error, heading error, and scenario-specific behavior. ShuffleNetV2 1.0× achieved the lowest mean control-loop latency at 42.44 ms, while GoogLeNet achieved the lowest mean open-loop heading error at 40.40°. Because the leading model changed with both the metric and the held-out robot, model selection should prioritize closed-loop testing on a held-out physical robot, with open-loop analysis used as a complementary diagnostic.

Index Terms—Convolutional neural networks, cross-robot generalization, embedded inference, Jetson, lane following, low-light environments

I. INTRODUCTION

Small camera-based robots allow lane-following models to be tested with perception, control, and embedded inference on the same physical platform. In standard JetBot lane following, a CNN maps each camera frame to a target coordinate, which a controller converts into steering. This compact pipeline suits the Jetson Nano, but coordinate error alone does not describe driving performance. A similar coordinate error can imply a different heading, while closed-loop behavior also depends on camera placement, wheel alignment, motor response, controller tuning, and inference delay. Evaluation on one robot can therefore hide changes that appear on another nominally identical unit.

The four backbones were selected to represent distinct CNN design strategies for embedded deployment. ResNet-18 provides a residual baseline. Each basic block learns a residual mapping and adds an identity or projection shortcut, which eases optimization [1]. Residual networks were demonstrated in image classification and object detection, and ResNet-18 has also been used for JetBot road following [2]–[4]. It therefore serves as a conventional reference model. MobileNetV2

targets mobile and resource-constrained inference. Its inverted residual block expands a narrow representation, applies a depthwise convolution, and linearly projects the result back to a bottleneck, with a shortcut when the input and output dimensions match [5]. The original work evaluated this design for classification, object detection, and semantic segmentation. MobileNetV2 is included to test whether mobile-specific efficiency yields a favorable prediction-quality–latency trade-off on the Jetson Nano.

ShuffleNetV2 1.0× targets platform-aware efficiency, based on the observation that FLOPs alone do not predict runtime because memory access and implementation overhead also affect speed [6]. Its stride-one unit splits features into identity and transformed branches, then concatenates and channel-shuffles the outputs. The original study evaluated the architecture on image classification and COCO object detection. In this study, its nominal computational complexity is considered alongside measured onboard latency. GoogLeNet represents multi-branch processing. Inception modules combine parallel convolutional and pooling paths, use 1×1 reductions before expensive spatial convolutions, and concatenate branch outputs [7]. Developed for classification and detection, it provides a contrasting multi-scale backbone for target-coordinate regression.

Prior JetBot, low-resource lane-following, and embedded-navigation studies examine model selection, driving metrics, processing time, and hardware effects [8]–[10], but they do not test whether backbone rankings remain stable under a robot-level holdout protocol across multiple physical JetBots.

Low-light lane research has mainly focused on image enhancement, semantic segmentation, and lane-detection accuracy [11]. Adverse-visibility studies also show that poor contrast, fog, rain, and related distortions reduce camera-based perception performance [12]. These findings motivate evaluation under difficult lighting. Because the controller converts each predicted target coordinate into a steering command, prediction error alone does not fully characterize driving performance. We therefore evaluate coordinate and heading errors together with closed-loop driving behavior and onboard timing. All experiments were conducted on an indoor track with illuminance ranging from 1 to 30 lux.

[†]Corresponding author: Jo Woon Chong (jwchong@skku.edu).

To examine cross-robot consistency under these conditions, we compare the four backbones using a three-fold leave-one-robot-out protocol. In each fold, two JetBots are used for model development, while the third is reserved for both closed-loop driving and open-loop video analysis. The resulting 12 model-fold configurations assess whether backbone rankings change across robots, whether coordinate and heading measures agree with driving behavior, and whether nominal complexity corresponds to complete control-loop timing. This protocol links robot-level holdout, physical driving, fixed-video analysis, and scenario-specific diagnostics without introducing a new backbone or controller. The main contributions of this paper are as follows:

- 1) A three-fold leave-one-robot-out protocol evaluates four backbones on a physical JetBot excluded from model development.
- 2) Closed-loop driving and onboard timing are paired with open-loop coordinate and heading analysis on the same held-out robot.

II. METHODS

The evaluation comprised four stages: (1) partition the data at the robot level, (2) train each backbone on two development robots, (3) conduct closed-loop driving on the held-out robot, and (4) analyze fixed videos from that same robot in open loop.

Repeating this process for three held-out robots and four backbones yielded 12 model-fold configurations and paired driving and fixed-input measurements for each unseen robot.

A. Platform and Dataset

The experiments used three NVIDIA Jetson Nano-based JetBots, denoted as JetBot1, JetBot2, and JetBot3. Each JetBot was equipped with an IMX-219 CSI camera and an IR sensor. Although the robots used the same base platform, the camera mounting angle, camera position, wheel alignment, and motor response could differ slightly across robots. These differences motivated the cross-robot validation protocol described in Section II-B.

Images were collected on a low-light indoor lane-following track. The indoor lights were turned off, external light was blocked, and data collection was conducted in the evening to maintain a low-illumination condition. The illuminance range was 1–30 lux. Illuminance was measured at the center of the track map and at representative positions along the track centerline, with the measurement height set near the JetBot camera height. This setting was used to reflect the visual condition observed by the onboard camera during data collection and driving.

The dataset contains 1,561 images: 533 from JetBot1, 500 from JetBot2, and 528 from JetBot3. Filenames encode the normalized regression targets as $x_y_x_y_...$.jpg. Inputs were resized to 224×224 and normalized using ImageNet statistics. The 27 annotated videos, nine per JetBot, contain 5,313 frames labeled as Straight, Curve, or TrackOut.

TABLE I
LEAVE-ONE-ROBOT-OUT CROSS-ROBOT DATA SPLIT.

Fold	Training robots	Test robot	Train	Val.	Test
fold0	JetBot1 + JetBot2	JetBot3	833	200	528
fold1	JetBot1 + JetBot3	JetBot2	861	200	500
fold2	JetBot2 + JetBot3	JetBot1	828	200	533

TrackOut denotes frames recorded during outside-track recovery.

B. Cross-Robot Validation

Table I gives the folds. Each 200-image validation set is drawn only from the two development robots. No data from the held-out robot are used during model development. The split excludes all data from the held-out robot during model development, but it does not eliminate other sources of bias or establish superiority over random image splitting, which was not evaluated.

C. Backbone Models

The selected backbones represent four common architectural designs. The classification heads in the torchvision implementations were replaced with two-output regression heads. The models contain 11.18M parameters for ResNet-18, 2.23M for MobileNetV2, 1.26M for ShuffleNetV2 1.0 $\times$, and 5.60M for GoogLeNet. Each model was trained using mean squared error (MSE) loss and the Adam optimizer. The fixed training configuration used 100 epochs, a batch size of 32, a weight decay of 10^{-4} , and learning-rate candidates $\{10^{-3}, 10^{-4}, 5 \times 10^{-5}, 10^{-5}\}$. For each model and fold, the checkpoint with the minimum validation loss was selected. All four backbones were trained in each of the three leave-one-robot-out folds, yielding 12 completed model-fold configurations. Training was performed on a desktop computer equipped with an Intel Core Ultra 7 265k CPU, an NVIDIA GeForce RTX 5060 Ti GPU with 16 GB memory, and 32 GB system RAM.

D. Closed-Loop Evaluation

The closed-loop evaluation consisted of one predefined three-lap run for each of the 12 model-fold configurations. Lap times are calculated from successive completion timestamps and averaged within each run. Steering and timing values are taken from the run summaries and frame logs. Reported Mean $\pm$ SD values are computed across the robot fold values for each backbone rather than across individual frames. Because the exact timing boundaries are unavailable, offline timing is not compared directly with onboard inference or control-loop timing.

Controller settings were adjusted for each model and fold. The speed gain was 0.090–0.100 for GoogLeNet, 0.105–0.110 for MobileNetV2, and 0.110 for the other models. The closed-loop results therefore compare complete model-controller configurations rather than isolated backbones.

TABLE II
LOW-LIGHT CLOSED-LOOP DRIVING RESULTS. VALUES ARE MEAN $\pm$ SD
ACROSS THE THREE HELD-OUT-ROBOT FOLDS.

Model	Lap (s)	$ \theta $ ($^\circ$)	Std.	Steering MAD	Inf. (ms)	Loop (ms)
ResNet-18	44.29 ± 1.98	27.05 ± 0.52	0.021 ± 0.004	0.002 ± 0.001	25.56 ± 0.61	44.88 ± 0.57
MobileNetV2	48.73 ± 1.05	25.23 ± 4.40	0.022 ± 0.002	0.003 ± 0.002	25.83 ± 0.78	45.00 ± 0.81
ShuffleNetV2 1.0 $\times$	44.20 ± 0.83	29.12 ± 1.28	0.022 ± 0.003	0.003 ± 0.002	23.40 ± 0.61	42.44 ± 0.68
GoogLeNet	59.61 ± 2.70	20.84 ± 3.44	0.016 ± 0.001	0.002 ± 0.001	23.93 ± 0.67	42.98 ± 0.59

E. Open-Loop Analysis

The fold-level vertical offset is $\delta_y = \bar{y} - \hat{y}$, with $\hat{y}_i^{\text{cal}} = \hat{y}_i + \delta_y$. Define $h(x, y) = \text{atan2}(x, 1 - y)$. For frame i , $\theta_i = h(x_i, y_i)$ and $\hat{\theta}_i = h(\hat{x}_i, \hat{y}_i^{\text{cal}})$; their wrapped difference is

$$\Delta\theta_i = \text{mod}(\hat{\theta}_i - \theta_i + \pi, 2\pi) - \pi. \quad (1)$$

Here, $x = 0$ denotes zero heading and $e_i = (180^\circ/\pi)|\Delta\theta_i| \in [0^\circ, 180^\circ]$ is the absolute angular error.

With $p_i = (x_i, y_i)$ and $\hat{p}_i = (\hat{x}_i, \hat{y}_i)$, coordinate RMSE is

$$\text{RMSE}_{xy} = \left(\frac{1}{N} \sum_{i=1}^N \|\hat{p}_i - p_i\|_2^2 \right)^{1/2}. \quad (2)$$

Substituting $\hat{y}_i^{\text{cal}}$ for $\hat{y}_i$ in (2) gives calibrated RMSE.

For $\tau \in \{5^\circ, 10^\circ, 15^\circ, 30^\circ\}$, threshold agreement and mean direction cosine are

$$A_\tau = \frac{1}{N} \sum_{i=1}^N \mathbb{I}(e_i \leq \tau), \quad (3)$$

$$D_{\cos} = \frac{1}{N} \sum_{i=1}^N \cos(\Delta\theta_i).$$

HA@15 and HA@30 are A_{15° and A_{30° ; Fail@30 is $1 - A_{30^\circ}$, and agreement AUC is the normalized trapezoidal area over the four A_τ values. Turn, lateral, and trend accuracies compare classes defined, respectively, by headings split at $\pm 5^\circ$, x coordinates with a 0.03 deadband, and consecutive heading changes with a 3° deadband. Trend transitions with both changes inside the deadband are excluded; correlation is Pearson's r between the two headings.

For C-Score, let P contain the eight higher-is-better metrics and L contain heading error and Fail@30. With $z_m = (m - \mu_m)/s_m$,

$$C\text{-Score} = \frac{1}{8} \sum_{m \in P} z_m - \frac{1}{2} \sum_{m \in L} z_m. \quad (4)$$

P comprises HA@15, HA@30, the three accuracies, correlation, direction cosine, and agreement AUC. The mean μ_m and sample standard deviation s_m are computed across the four backbones, so C-Score is restricted to within-study comparisons.

III. RESULTS

A. Closed-Loop Driving Performance

Table II and Fig. 1 summarize the 12 model-fold runs, all of which completed the predefined three-lap protocol. Shuf-

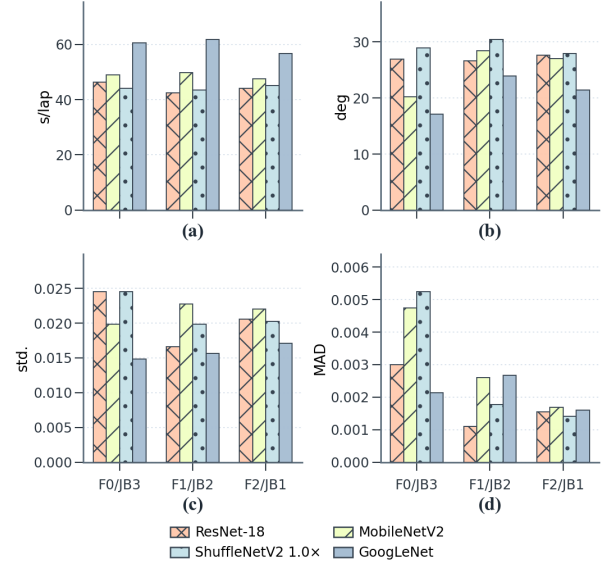


Fig. 1. Low-light closed-loop results by fold: (a) lap time, (b) mean absolute steering angle, (c) steering standard deviation, and (d) steering MAD.

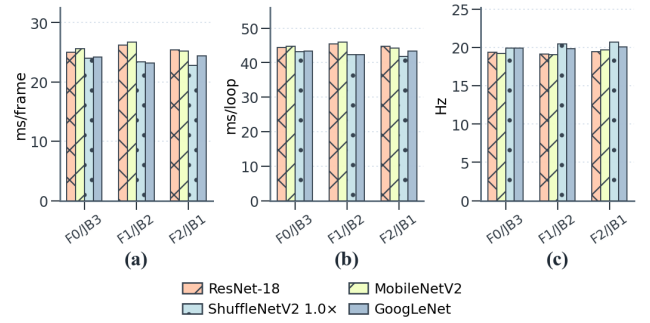


Fig. 2. Closed-loop timing by fold: (a) inference time, (b) control-loop latency, and (c) control FPS.

fleNetV2 1.0 $\times$ and ResNet-18 produced the shortest mean lap times. GoogLeNet had lower steering magnitude and variation, but it also used the lowest speed setting. MobileNetV2 was intermediate in lap time. Because controller gains differed by model and fold, these results describe tuned model-controller configurations rather than isolated backbone effects.

Fig. 2 shows the lowest mean loop latency for ShuffleNetV2 1.0 $\times$ at 42.44 ms, followed by GoogLeNet at 42.98 ms. MobileNetV2 had the highest value at 45.00 ms. The timing order did not follow parameter count: MobileNetV2 had fewer parameters than ResNet-18 but a higher mean loop latency.

B. Open-Loop Prediction Performance

1) *Aggregate Performance:* Table III summarizes the fixed-input evaluation across the three held-out robots. GoogLeNet had the strongest mean heading measures and the highest C-Score, while ShuffleNetV2 1.0 $\times$ ranked second by C-Score. ResNet-18 had lower coordinate RMSE than MobileNetV2 but

TABLE III
THREE-FOLD AVERAGE RESULTS FOR OPEN-LOOP HEADING-CENTRIC EVALUATION.

Model	RMSE _{xy} ^{cal}	Head. err. (°)	HA@15	Turn acc.	Fail.@30	Inf. (ms)	C-Score
ResNet-18	0.507	62.63	0.353	0.652	0.493	1.52	-2.18
MobileNetV2	0.545	54.56	0.421	0.734	0.427	2.41	-0.28
ShuffleNetV2 1.0×	0.454	51.32	0.387	0.756	0.423	3.01	-0.04
GoogLeNet	0.448	40.40	0.505	0.796	0.336	3.20	+2.50

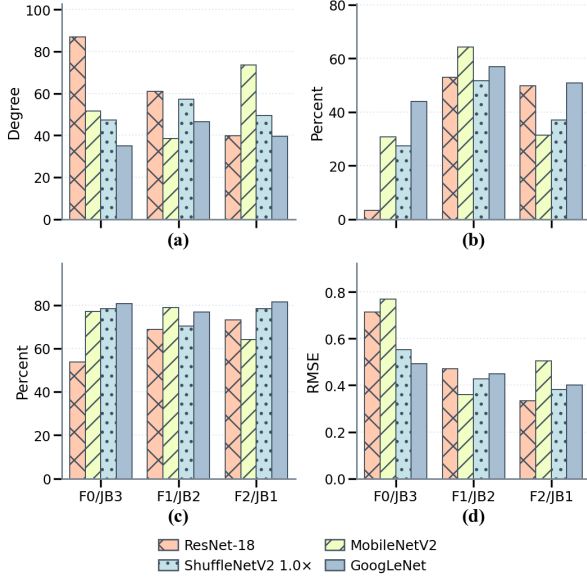


Fig. 3. Open-loop results by fold: (a) mean heading error, (b) heading agreement within 15°, (c) turn-direction accuracy, and (d) RMSE_{xy}.

higher heading error, lower turn accuracy, and higher Fail@30. Coordinate and directional metrics therefore produced different rankings.

2) *Cross-Robot Variation*: Fig. 3 shows that the leading backbone varied across held-out robots. GoogLeNet led all four measures on JetBot3 (fold 0), while MobileNetV2 did so on JetBot2 (fold 1). On JetBot1 (fold 2), GoogLeNet led the directional measures, whereas ResNet-18 achieved the lowest RMSE_{xy}^{cal}. Across folds, GoogLeNet remained first or second on the three heading measures. ShuffleNetV2 1.0× had the smallest heading-error range (10.13°), compared with 38.48°–73.69° for MobileNetV2 and 39.95°–87.06° for ResNet-18. ResNet-18’s low JetBot1 RMSE_{xy}^{cal} still did not yield the highest turn accuracy.

3) *Scenario-Specific Performance*: Table IV and Fig. 4 separate normal driving from recovery behavior. GoogLeNet had the lowest heading error in Curve, Straight, and TrackOut and the lowest TrackOut Fail@30. TrackOut heading error exceeded 51° for every model. MobileNetV2 had the highest TrackOut Fail@30, while ResNet-18 was weakest on Straight. The ordering of the remaining models changed with the scenario and metric, showing that aggregate results can hide scenario-specific failure modes.

TABLE IV
REPRESENTATIVE THREE-FOLD AVERAGE RESULTS BY SCENARIO (OPEN-LOOP SUPPLEMENTARY ANALYSIS).

Scenario	Model	RMSE _{xy} ^{cal}	Head. err. (°)	HA@15	Fail.@30
Straight	ResNet-18	0.460	81.67	0.157	0.704
Straight	MobileNetV2	0.438	58.61	0.392	0.434
Straight	ShuffleNetV2 1.0×	0.379	57.63	0.207	0.519
Straight	GoogLeNet	0.378	52.03	0.357	0.424
Curve	ResNet-18	0.392	42.73	0.478	0.319
Curve	MobileNetV2	0.418	33.92	0.466	0.275
Curve	ShuffleNetV2 1.0×	0.331	31.55	0.465	0.256
Curve	GoogLeNet	0.319	18.93	0.611	0.198
TrackOut	ResNet-18	0.693	72.57	0.355	0.477
TrackOut	MobileNetV2	0.807	68.61	0.376	0.520
TrackOut	ShuffleNetV2 1.0×	0.688	68.50	0.440	0.471
TrackOut	GoogLeNet	0.671	51.32	0.535	0.382

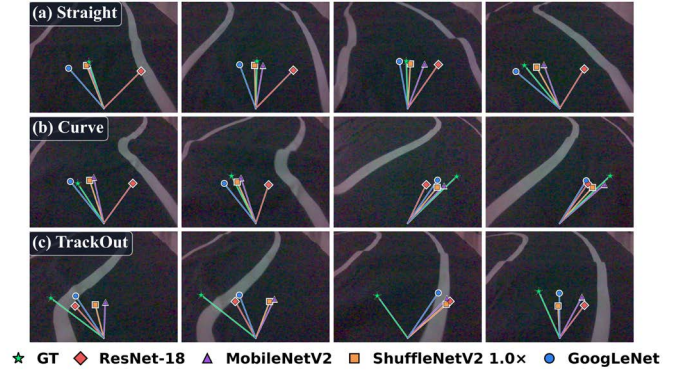


Fig. 4. Representative model predictions and ground truth by scenario.

IV. DISCUSSION

A. Relating Closed- and Open-Loop Evaluation

Closed-loop driving should be the primary criterion for deployment because each prediction changes the robot trajectory and, through feedback, the camera frames observed next. It therefore evaluates the complete model–controller system, including controller gains, timing, actuation, camera geometry, and robot dynamics. The same realism limits attribution: differences in lap time or steering cannot be assigned to the backbone alone when controller settings and physical responses differ.

Open-loop analysis removes this feedback by replaying fixed videos from the same held-out robot. Every model receives the same frames, making coordinate, heading, and scenario-level errors repeatable and directly comparable within each fold. Open-loop results cannot establish driving success or replace closed-loop testing, and offline inference time is not a substitute for onboard control-loop latency. Their value is diagnostic. They can reveal directional or recovery weaknesses that route completion and lap time may not expose, particularly when all configurations complete the predefined run. In this study, the strongest open-loop heading and recovery results did not coincide with the shortest closed-loop lap times. The appropriate hierarchy is therefore to use closed-loop behavior as the primary deployment evidence and open-loop analysis

as a complementary diagnostic for interpreting disagreement across metrics, robots, and scenarios.

B. Backbone Trade-Offs

The model comparison did not identify a universal winner. Closed-loop results favored ShuffleNetV2 1.0 $\times$ for control-loop latency and, together with ResNet-18, for lap time, whereas open-loop heading and recovery measures favored GoogLeNet. The mismatch between ResNet-18's coordinate and heading results also shows that image-plane proximity does not guarantee directional agreement. Thus, latency, coordinate accuracy, heading quality, and tuned driving behavior should be treated as distinct selection criteria rather than collapsed into parameter count or a single aggregate rank. Because controller gains differed and no architectural ablation was performed, these findings characterize the tested model-controller configurations rather than causal effects of individual network components.

C. Limitations and Implications

The study is limited to three physical JetBots, one indoor track, and one indoor low-light setting with illuminance ranging from 1 to 30 lux. It has no matched well-lit baseline and does not control exposure and illumination independently. Model-specific controller tuning prevents a backbone-only closed-loop comparison, while one three-lap run per model-fold configuration limits repeated-run analysis. No formal significance test, architecture ablation, or operation-level runtime profiling was performed. Differences in camera mounting, motors, wheel alignment, battery state, and surface friction may contribute to fold results. Each fold also changes both the development-robot composition and the held-out robot, so the study demonstrates robot-dependent transfer variation without isolating its cause or the causal effect of an architecture.

The paired design supports joint evaluation but does not establish which open-loop measures predict closed-loop performance. Future work should preserve closed-loop testing as the primary endpoint and use repeated runs, common or systematically controlled controller settings, and matched illumination and exposure conditions to examine this relationship. Statistical analysis is needed to determine whether coordinate, heading, and scenario-specific errors reliably predict or explain lap time, steering behavior, recovery, and completion. Camera-pose calibration, balanced sampling, repeated training, and detailed runtime profiling could help reduce confounding.

V. CONCLUSION

Four backbones were evaluated in three leave-one-robot-out folds, with closed-loop driving and open-loop video analysis performed on the same held-out JetBot. The leading model changed across robots and evaluation measures, and lap time, coordinate error, heading quality, steering behavior, and latency did not point to the same backbone. Closed-loop testing should remain the primary basis for deployment decisions because it evaluates the complete physical system under feedback. Open-loop analysis should be retained as a

complementary, repeatable diagnostic of coordinate, heading, and scenario-specific errors. Model selection should therefore prioritize closed-loop evaluation on a physical robot excluded from development and use open-loop results to interpret disagreement across metrics, robots, and scenarios. No single metric or parameter count identified the most suitable model for every objective.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. RS-2023-00221186). This work was supported by the BK21 FOUR Project.

REFERENCES

- [1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
- [2] C.-C. Chang, Y.-H. Juan, C.-L. Huang, and H.-J. Chen, "Scenario analysis for road following using JetBot," in *Proc. Eurasia Conf. on IOT, Communication and Engineering (ECICE)*, Yunlin, Taiwan, Oct. 2020, pp. 403–406.
- [3] P. Leigh and Y. Xing, "Evaluation of multiple convolutional neural networks in training the NVIDIA JetBot," in *Proc. Annual Ubiquitous Computing, Electronics & Mobile Communication Conf. (UEMCON)*, New York, NY, USA, Oct. 2022, pp. 524–530.
- [4] G. R. Vidangos Ponce *et al.*, "Real-time autonomous navigation with ResNet-18 and PyTorch based on AI computer Jetson Nano," in *Proc. CHILEAN Conf. on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON)*, Valparaíso, Chile, Oct. 2025.
- [5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. Computer Vision and Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, Jun. 2018, pp. 4510–4520.
- [6] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. European Conf. on Computer Vision (ECCV)*, Munich, Germany, Sep. 2018, pp. 116–131.
- [7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1–9.
- [8] C. D. d. S. Bezerra, I. H. L. Cardoso, and F. H. T. Vieira, "Deep reinforcement learning with convolutional networks applied to autonomous navigation of real robots using virtual scenario training," in *Proc. Latin American Robotics Symposium (LARS), Brazilian Symposium on Robotics (SBR), and Workshop on Robotics in Education (WRE)*, Salvador, Brazil, Oct. 2023, pp. 302–307.
- [9] B. Froemming-Aldanondo *et al.*, "Evaluating low-resource lane following algorithms for compute-constrained automated vehicles," in *Proc. Artificial Intelligence, Robotics and Control (AIRC)*, Savannah, GA, USA, May 2025, pp. 121–127.
- [10] S. Ma, P. S. Karkal, J. Ma, and F. Alam, "A comparison of the performance of ResNet and DenseNet for obstacle avoidance and path following on a mobile robot using embedded GPUs," in *Proc. International Conf. on Agents (ICA)*, Wollongong, Australia, Dec. 2024, pp. 7–12.
- [11] S. Song, W. Chen, Q. Liu, H. Hu, T. Huang, and Q. Zhu, "A novel deep learning network for accurate lane detection in low-light environments," *Proc. Inst. Mech. Eng., Part D: J. Automob. Eng.*, vol. 236, no. 2–3, pp. 424–438, May 2022.
- [12] T. Rothmeier, D. Wachtel, T. von dem Bussche-Hunnefeld, and W. Huber, "I had a bad day: Challenges of object detection in bad visibility conditions," in *Proc. Intelligent Vehicles Symposium (IV)*, Anchorage, AK, USA, Jun. 2023.