ISSN 2007-9737

Evaluation of Multiple Correction Methods for Depth Images for UAV Implementations

Arturo Javier Aceves Ramírez*, Leopoldo Altamirano Robles

National Institute of Astrophysics, Optics and Electronics,
Astrophysics Coordination, Puebla,
Mexico

arturo.aceves@inaoe.mx, robles@inaoep.mx

Abstract. Accurate perception of the local environment is essential for autonomous navigation, and depth information is crucial for path correction. Such information can be obtained using various sensing devices, like Ligth Detection and Ranging (LiDAR) sensors or, as in this work, stereo cameras. However, raw sensor measurements are often subject to error, which can lead to incorrect decision-making. To address this, it is necessary to develop error characterization techniques and correction algorithms that enhance the reliability of autonomous navigation systems. Building upon a previous implementation of an exponential decay method on a Jetson Nano, this paper presents an experimental comparison of multiple correction approaches based on polynomial regression (PR) and artificial neural network (ANN) models. A PR model utilizing QR decomposition and data analysis achieved the highest RMS error reduction at a 5-meter range, by 5.37%. A PR model based on Singular Value Decomposition (SVD) using Scikit-learn demonstrated improvements of 0.9% and 6.99% at the 3- and 7-meter ranges, respectively. All four evaluated models demonstrated real-time processing capability. Notably, the ANN model achieved the fastest performance, reaching 28.05 FPS, compared to 27.28 FPS obtained by the slower scikit-learn SVD model. The results of this study demonstrate that the proposed methods improve measurement accuracy within the 3- to 7-meter range while maintaining high-speed processing performance, contributing to enhanced data reliability for Unmanned Aerial Vehicles (UAV) applications.

Keywords. UAV, depth image, image correction, machine learning.

1 Introduction

Accurate environment perception is a cornerstone of autonomous navigation. Vehicles that operate without human intervention must be able to estimate distances, detect obstacles, and plan safe paths in real time. A wide variety of sensors can provide spatial information, including LiDAR, stereo vision, and time-of-flight devices. Among them, depth stereo cameras stand out for offering three-dimensional data at relatively low cost, which has led to their adoption in diverse fields such as geographic mapping [10], robotic welding systems [8], and anthropometric studies [7].

Despite their advantages, depth cameras often produce systematic errors that affect measurement accuracy. Unlike metrological instruments designed as standards, these sensors may report values that deviate from ground truth, which becomes problematic when high-level precision is required. In some cases, replacing the hardware is not always feasible, making it necessary to explore mathematical models that can compensate for these inaccuracies.

These limitations become particularly critical in the context of UAVs, where navigation depends on fast and reliable distance estimation. UAVs have gained importance in diverse applications, from commercial delivery [5] to defense operations [13], mainly due to their versatility and mobility.

Research on implementing depth sensing on UAVs is also being conducted, as various works [3, 6, 12] also utilize depth images as the basis for

obstacle avoidance problems; however, the success of these implementations strongly relies on accurate calibration and effective correction techniques to compensate for sensor inaccuracies.

In the previous work [1], we proposed an exponential decay-based correction algorithm and validated its use in real-time depth image analysis for UAV navigation. While this method proved effective, it raised the need to investigate alternative approaches that could yield higher accuracy or robustness.

The present study addresses this gap by evaluating various correction methods, including the already mentioned exponential decay correction, predictive models, and machine learning techniques. Furthermore, we explore their feasibility for embedded implementation, aiming again at real-time image processing and path correction in UAV systems.

2 Related Work

Depth image correction has been addressed using several strategies aimed at mitigating sensor limitations. Vasudevan et al. [17] and Park et al. [11], for example, propose techniques to minimize the presence of flying pixels and erroneous readings by leveraging RGB data and spatial relationships between neighboring pixels. Xin et al. [14] also implement pixel-level operations to suppress these artifacts, leading to improvements in segmentation quality and overall image consistency.

These studies highlight the potential of pixel refinement to produce cleaner depth maps with sharper object boundaries. Nonetheless, a standard limitation is that they primarily focus on reducing noise and invalid pixels, while offering little analysis regarding the accuracy of the measured distances when compared to ground truth values.

Research has also investigated the use of multi-sensor configurations, where stereo or multiple depth cameras are fused to obtain more reliable measurements. Zhang et al. [20] demonstrated how combining depth data from multiple devices and deep learning inference can improve 3D reconstruction of architectural models.

Similarly, Xie et al. [19] developed a three-camera fusion method that reduced the overall Root Mean Square (RMS) error of depth measurements. He et al. [4], in turn, proposed a weighted fusion strategy that increased accuracy on both flat and curved surfaces at a fixed distance of 1.5 meters. In general, these works emphasize the reconstruction quality and numerical precision achieved by integrating multiple depth sources.

In contrast to these state-of-the-art techniques, our previous work explored a per-pixel correction function based on an exponential decay factor. This approach was designed to counteract the increasing error trend observed at longer ranges with the Intel RealSense D455 depth camera, resulting in more reliable distance estimates and improved real-time obstacle avoidance.

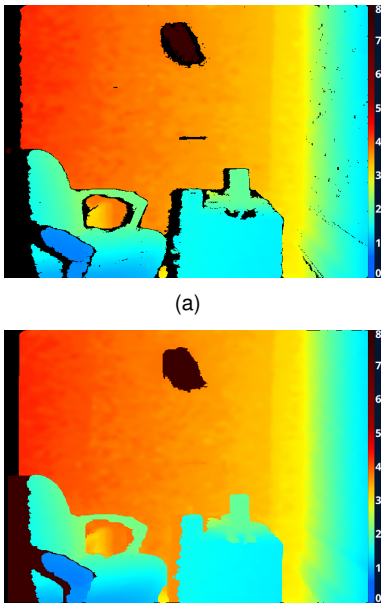
Other studies have investigated interpolation and regression methods for depth correction in specific domains. For example, Li et al. [9] evaluated several models for underwater mapping, combining classical interpolation techniques with machine learning to fill unmapped regions. Similarly, Alam et al. [2] applied multiple interpolation methods to maritime depth data, producing more detailed ocean maps. These works highlight the potential of model-based approaches to enhance depth information in scenarios where gaps or missing data are frequent.

3 Depth Image Sensing and Correction

To start, the Intel RealSense D455 depth camera was selected due to its balance between cost, range, and data reliability, making it suitable for UAV applications in outdoor environments. Stereo vision technology, employed in this model, offers a robust solution for dynamic scenarios, eliminating the need for active emitters and ensuring consistent operation under varying conditions.

The D455 offers a working range of 0.5 to 6 meters, with an absolute error of approximately 2% under 4 meters and 4% at greater distances, which is adequate for obstacle avoidance tasks. Its field of view of $87° \times 58°$ provides wide coverage, while the minimum detectable distance of 320 mm allows close-range sensing.

Another critical factor is the flexibility in frame rate and resolution. A resolution of 640×480 pixels was chosen, as it provides over 300,000 depth points



(a) (b)

Fig. 1. Jet-scaled depth images showed yielding different proportions of valid pixels. Color scale in meters, with black points representing invalid information. (a) No filter - 96.7%, (b) Hole filling filter - 100%. Source: Own work.

(a) (b)

Fig. 2. Camera setup: (a) support with embedded device and camera, (b) RGB image captured. Source: Own work.

per frame while maintaining a stable frame rate of 30 fps.

Overall, the Intel RealSense D455 provides a reliable depth perception capability with configurable parameters that strike a balance between accuracy, coverage, and computational efficiency, making it a suitable choice for the intended UAV system.

Additionally, the environment used for the experiments is a controlled indoor laboratory with an average value of 451 lux.

3.1 Image Acquisition and Filtering

Depth data from the Intel RealSense D455 was obtained through the Python SDK, which also provides several post-processing filters to improve image usability.

A recurring issue in stereo vision systems is image disparity error calculation, leading to invalid pixels or “shadows” near object edges or along image borders. These artifacts reduce the amount of usable information for obstacle detection, which is critical in UAV navigation. Due to the cameras’ relative positioning, disparity errors near the image borders are disregarded in pixel validity calculations, as they are inherent to the stereo configuration.

To address this, different filtering techniques were tested to maximize the number of valid pixels while maintaining real-time processing. Four filters from the SDK [15] were considered: Decimation, Spatial Edge-Preserving, Temporal, and Hole Filling. Each introduces specific trade-offs between accuracy, smoothness, and noise.

In this work, the priority is to ensure obstacle continuity in the image rather than absolute depth precision, since missing data points can compromise avoidance performance.

Having this in considerations, the experimental results showed that the hole-filling filter provided the most reliable output, effectively reducing disparity artifacts and achieving complete pixel coverage (Fig. 1). Although this method increases measurement noise by approximately 3%, it was

Table 1. Comparison of RMS errors between the ground truth, regression model, mean measurement, and subpixel estimation

G.T (mm)	RMS GT (mm)	RMS QR (mm)	Mean (mm)	Sub pixel
1000	2.91 (0.29%)	1.11 (0.11%)	997.00	0.09
3000	45.56 (1.52%)	16.76 (0.56%)	3042.00	0.16
5000	321.85 (6.44%)	64.71 (1.29%)	5315.00	0.41
7000	589.01 (8.41%)	58.46 (0.84%)	7585.00	0.38

selected as the most suitable option for the obstacle avoidance algorithm.

3.2 Characterization and Correction

The accuracy of depth measurements can be improved using correction models. For this, the manufacturer provides an RMS error estimation (Eq. 1), which shows a quadratic growth with distance:

$$DeepthRMS = 2 \times tan(\frac{HFOV}{2}) \times \frac{D^2 \times Subpixel}{ResX \times Baseline}. \quad (1)$$

Experimental measurements (Fig. 2) at 1, 3, 5, and 7 meters confirmed this trend (Fig. 3 and Fig. 4), with RMS increasing significantly even more than the expected values at longer distances (Table 1).

A quadratic regression model was fitted, showing closer alignment with mean values but still leaving an offset at large distances.

To address this, we proposed an empirical approach, using an exponential decay correction (Eq. 2), which progressively reduces measured depth values, improving accuracy across the 3–7 m range:

$$D_m(d) = d \times e^{-d \times k}. \quad (2)$$

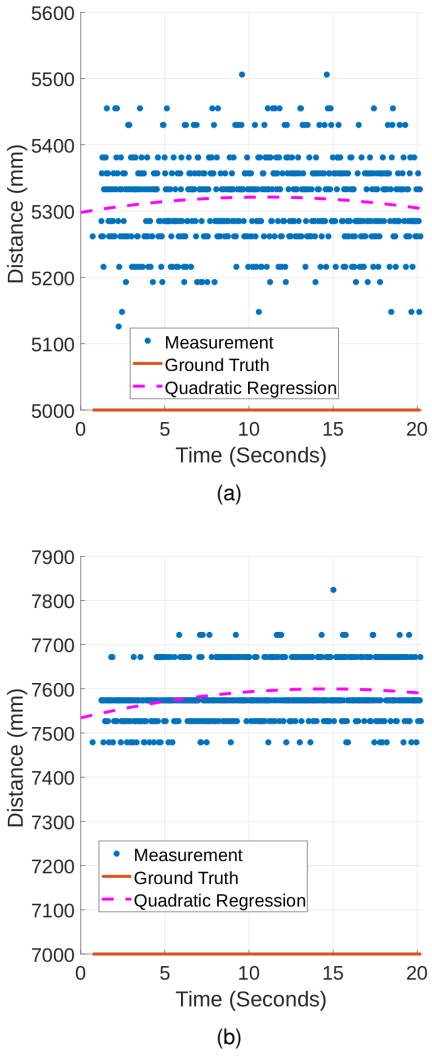


Fig. 3. Camera measurement results at varying distances, compared with ground truth and quadratic regression. (a) 5 meters and (b) 7 meters. Source: Own work

4 Predictive Modeling Approaches

However, the proposed correction strategy mainly relies on prior knowledge and empirical observations of the curve behavior. An alternative perspecttive is to leverage predictive models, which aim to estimate measurement errors based on data-driven approaches. These models can be constructed using well-established mathematical

Table 2. RMS values for the theoretical best performers values of k, and the real RMS of measurements with the pixel operation applied, where $k_1 = 5 \times 10^{-6}$ and $k_2 = 1 \times 10^{-5}$

Ground Truth (mm)	k_1 (mm)	k_2 (mm)	Operational RMS (mm)
1000	7.72 (0.77%)	13.60 (1.36%)	11.65 (1.16%).
3000	16.98 (0.57%)	60.05 (2%)	26.79 (0.89%)
5000	186.28 (3.73%)	59.18 (1.18%)	100.77 (2.02%)
7000	308.66 (4.41%)	55.33 (0.79%)	131.56 (1.88%)

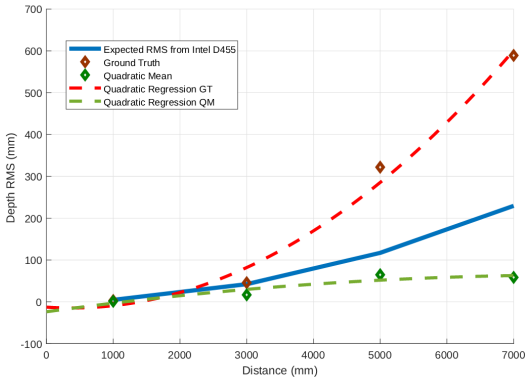


Fig. 4. Comparison of expected RMS error, ground-truth RMS error per measurement point, and quadratic regression RMS error (Quadratic Mean). The regression from the last two data sets illustrates the RMS error trend. Source: Own work

formulations or, more recently, with the aid of artificial intelligence techniques.

Given that computational resources are limited in embedded platforms, it is essential to prioritize simpler and lightweight models as an initial evaluation strategy. Accordingly, this section aims to examine several predictive approaches and assess their performance in comparison with the baseline method.

4.1 Polynomial Regression Models

Regression models are widely used to infer relationships between input and output variables through mathematical formulations. One of the most common methods is polynomial regression, which allows fitting nonlinear $f(x) = y$ relationships using a linear statistical framework. This is accomplished by expanding the input variables, enabling the identification of a function that approximates the outputs.

Polynomial regression has been extensively used, as it provides an interpretable representation of data behavior when only one dependent variable is considered [21]. Its main limitation, however, is reduced accuracy when extrapolating beyond the minimum and maximum ranges of the training data.

In this work, polynomial expansion is achieved through the construction of a Vandermonde matrix. Two decomposition methods are compared: the classical QR decomposition and SVD (referred to as QR-based and Scikit-learn, respectively, later in the text). The latter is particularly appealing due to its straightforward integration into practical deployments using built-in functions from the Scikit-learn library.

For each decomposition method, we evaluated polynomial models of degrees 1, 2, 3, and 4, resulting in a total of eight models. These degrees were selected because they offer a favorable balance between computational efficiency and approximation accuracy

For all polynomial regression experiments, we include a bias term to allow the model to capture constant effects that are independent of the input values.

4.2 Artificial Neural Network Model

Neural networks are based on the concept of interconnected neurons arranged in multiple layers, enabling the adaptation of input information to generate predictions. The structure of such networks can enhance predictive performance, although they present certain drawbacks, such as reduced interpretability (and thus potentially less reliable predictions) and increased inference time.

For basic regression tasks, it is generally recommended to employ relatively simple network architectures [16]. In this study, we propose four single hidden-layer ReLU activation functions based on architectures with 8, 16, 32, and 64 neurons,

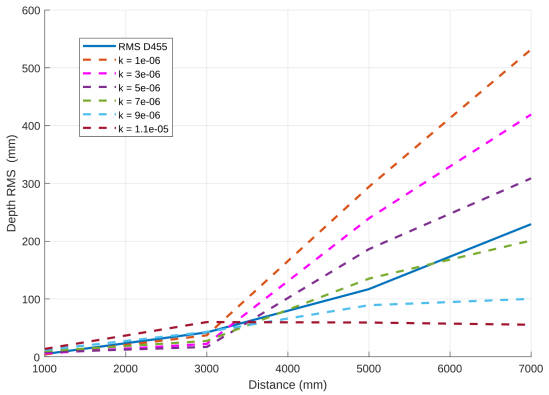


Fig. 5. RMS error comparison relative to the ground truth for different k values and the expected behavior. Source: Own work

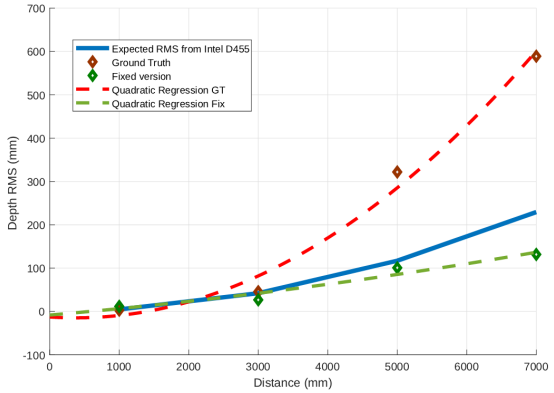


Fig. 6. RMS comparison relative to the ground truth for the original measurements, the processed measurements, and the expected behavior. Source: Own work

respectively. All models were trained using the Adam optimizer, a batch size of 10, and 100 training epochs.

4.3 Data Preparation

For the new experiments, we expanded the dataset to provide a more comprehensive basis for model training. Measurements were taken at intervals of 500 millimeters, ranging from 1000 to 7000 millimeters, resulting in a total of 14 intervals. From each interval, 50 samples were randomly selected to construct the training and testing datasets. Thus, each dataset consisted of 700 points.

The data were then prepared according to the requirements of each modeling approach.

In polynomial regression, the expansion of input values can lead to huge numerical values, potentially causing computational instability or convergence to local minima [18].

In the other hand, for regression neural networks the use of ReLU activation function is known to suffer from issues such as exploding or vanishing gradients when the data are not properly conditioned.

To mitigate these problems, a z-score normalization was applied to the input data, where the mean and standard deviation of the dataset were 3872 and 2133, respectively.

To evaluate the implementation of the models on the measurement process, samples were taken only at 1000, 3000, 5000, and 7000 millimeters.

5 Computing Device Characteristics

For implementation on a UAV, the onboard computer must process images efficiently to enable seamless navigation adjustments. Since the vehicle is intended to operate autonomously, all sensing and processing tasks must occur locally, thereby precluding reliance on external stations.

This requirement makes a lightweight, embedded computing platform essential.

In this study, an NVIDIA Jetson Nano (utilizing the 4 GB RAM board) was selected due to its compact size and direct compatibility with the Intel SDK.

This device features an integrated GPU, which is specifically designed for AI and image processing tasks, making it ideal for our work. The energy configuration used is the 5W input.

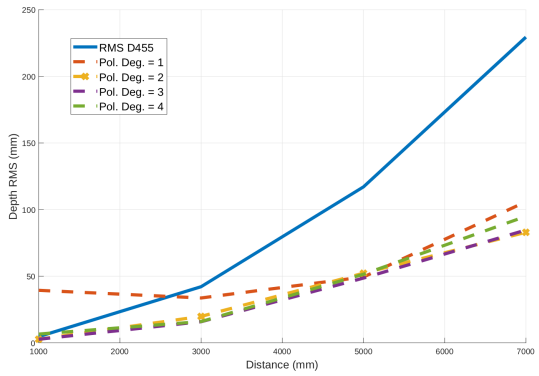


Fig. 7. RMS error comparison relative to the ground truth for different polynomial degrees obtained using QR decomposition. Source: Own work

6 Results and Discussion

For the evaluation of depth measurement error, multiple comparisons were conducted between the proposed empirical method and the predictive models.

Based on the results obtained with the test data, one representative model from each approach will be selected for implementation in the measurement process and subsequently evaluated against the theoretical behavior.

6.1 Exponential Decay Method

Testing different k values (Fig. 5), $k = 1 \times 10^{-5}$ minimized RMS at long distances (5 - 7 meters), while $k = 5 \times 10^{-6}$ was better at short range (1 - 3 meters).

The first value was selected for our implementation, as it reduced long-distance errors more effectively (Table 2). The implementation of this correction in Python validated the results, showing a clear reduction in RMS error compared to the raw data (Fig. 6).

Table 3. RMS values for the theoretical best-performing polynomial models and the RMS of real measurements after applying the pixel operation. P_2 denotes the quadratic model and P_3 the cubic model

Ground Truth (mm)	P_2 (mm)	P_3 (mm)	Operational RMS (mm)
1000	2.73 (0.27%)	2.73 (0.27%)	7.26 (0.72%).
3000	19.86 (0.66%)	15.85 (0.52%)	24.27 (0.81%)
5000	52.28 (1.04%)	48.63 (0.97%)	53.92 (1.07%)
7000	82.96 (1.18%)	84.66 (1.21%)	100.99 (1.44%)

6.2 Polynomial Regression models

Using the QR-based methodology, the resulting polynomials up to degree four are presented in Eq. 3–6:

$$2016.03x + 3750, \tag{3}$$

$$-47.59x^2 + 2018.62x + 3797.52, \tag{4}$$

$$-4.88x^3 - 47.18x^2 + 2027.36x + 3797.38, \tag{5}$$

$$9.6x^4 - 6.05x^3 - 71.33x^2 + 2028.68x + 3804.37. \tag{6}$$

The evaluation of these polynomials (Fig. 7) shows that the second- and third-degree models achieved the lowest error at long distances, with the quadratic model performing slightly better in the mid-range (3–5 meters), although somewhat worse at 7 meters.

Considering that the second-degree polynomial requires fewer computations and is less dependent on the input variable, it was selected as the most suitable option for deployment. A comparison between the two best polynomials and the deployed model is presented in Table 3.

For the Scikit-learn implementation, the results (Fig. 8) indicate that the fourth-degree model exhibited a noticeably larger error initially. The quadratic model provided better accuracy at longer distances (5–7 meters), while the cubic model was more accurate at short distances (1–3 meters).

Given that the quadratic model performed better at long ranges, it was chosen for deployment. A comparison of the two best-performing models

ISSN 2007-9737

Table 4. RMS values for the best-performing polynomial models implemented with scikit-learn, and the RMS of real measurements after the pixel operation. D_2 denotes the quadratic model and D_3 the cubic model

Ground Truth (mm)	D_2 (mm)	D_3 (mm)	Operational RMS (mm)
1000	2.72 (0.27%)	2.24 (0.22%)	6.18 (0.62%).
3000	17.34 (0.58%)	15.63 (0.52%)	18.62 (0.62%)
5000	46.75 (0.93%)	48.56 (0.97%)	54.75 (1.09%)
7000	84.91 (1.21%)	85.76 (1.22%)	99.70 (1.42%)

Table 5. RMS values for the best-performing neural configurations and the RMS of real measurements with the pixel operation applied, where N_8 denotes 8 neurons and N_{64} denotes 64 neurons

Ground Truth (mm)	N_8 (mm)	N_{64} (mm)	Operational RMS (mm)
1000	33.24 (3.32%)	34.16 (3.41%)	36.84 (3.68%).
3000	69.45 (2.32%)	71.10 (2.37%)	72.79 (2.43%)
5000	50.49 (1.01%)	51.43 (1.03%)	64.05 (1.28%)
7000	130.67 (1.87%)	128.29 (1.83%)	139.06 (1.99%)

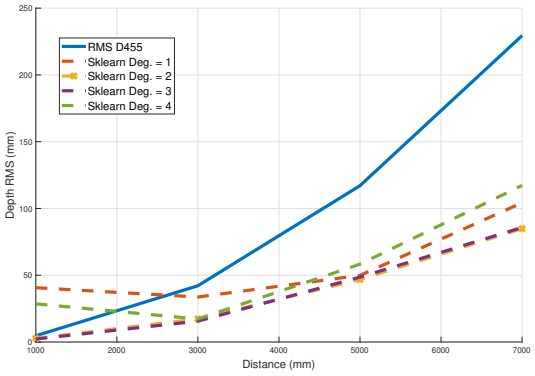


Fig. 8. RMS error comparison relative to the ground truth for polynomial models of different degrees implemented in Scikit-learn. Source: Own work

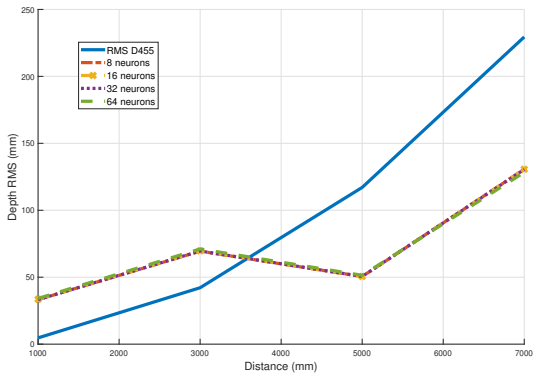


Fig. 9. RMS error relative to the ground truth for different neuron configurations and their expected behavior. Source: Own work

and the deployed configuration is summarized in Table 4.

6.3 Artificial Neural Network Model

As shown in Fig. 9, all configurations reduced the RMS error at longer ranges (5–7 meters) but increased it at shorter ranges (1–3 meters). All the models exhibit similar trends; however, the network with eight neurons achieved the lowest error at 1, 3, and 5 meters, while the 64–neuron model yielded the lowest error at 7 meters.

Considering both accuracy across most distances and computational efficiency for deployment on embedded hardware, the 8-neuron configuration was selected. A comparison between the two best configurations and the deployed pixel operation is reported in Table 5.

6.4 Models Measurement Behavior

Based on the outputs of each deployed model, a global comparison was carried out with the general results (Fig. 10), while the corresponding numerical values are presented in Table 6.

At the 1 m range, all models increase the measurement error; however, the Sklearn model achieves the best performance at this distance, with an improvement of 0.54%.

Table 6. Comparison of RMS error differences between ground truth and each evaluated model. The highlighted values indicate the gain or loss relative to the exponential decay model

G.T (mm)	Exp. Dec. (% mm)	Pol (% mm)	Sklearn (% mm)	ANN (% mm)
1000	-0.87	-0.43	-0.33	-3.39
		0.44	**0.54**	**-2.52**
3000	0.63	0.71	0.9	-0.91
		0.08	**0.27**	**-1.54**
5000	4.42	5.37	5.35	5.16
		0.95	**0.93**	**0.74**
7000	6.53	6.97	6.99	6.42
		0.44	**0.46**	**-0.11**

This increase in error in all the models may be attributed to several factors: the scale of the values used during model construction (given that the dataset spans a relatively wide range), the limited complexity of the models, which may be insufficient to fully characterize the system, or the fact that the raw measurements at this distance are already close to the ground truth.

At the 3 m range, the neural network underperforms, a behavior also noted in Section 6.3. The Sklearn model provides the best outcome at this distance, improving the measurement by 0.27%.

At a range of 5 m, all models achieve improvements, with the QR-based regression delivering the highest gain of 0.95%.

Finally, at the 7 m range, the neural network again shows lower performance, while the Sklearn model exhibits the best result, with an improvement of 0.44%.

Overall, when compared directly with the exponential decay model, the Sklearn regression achieved improvements of 0.54% at 1 m, 0.27% at 3 m, and 0.44% at 7 m, while the QR-based regression reached the highest improvement of 0.95% at 5 m. The neural network, in contrast, presented lower performance at 3 m and 7 m relative to this reference.

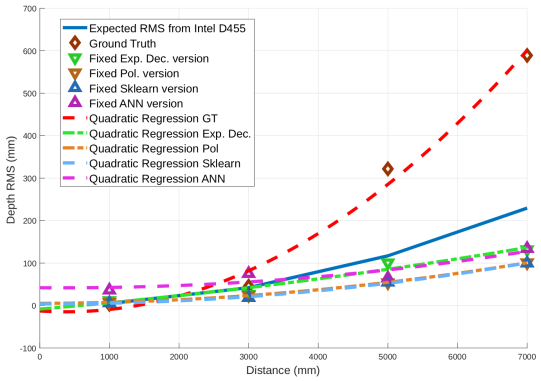


Fig. 10. RMS comparison relative to the ground truth for the original measurements, the corrected data using the top-performing model from each section, and the expected trend. Source: Own work

6.5 Embedded Device Time Analysis

For a time analysis, the models were deployed on a Jetson Nano computer, where the image acquisition and processing algorithms were executed 30 times to compare them with the 30 FPS configured initially. Table 7 summarizes the obtained results.

Overall, the execution times do not differ significantly across the models. The neural network achieved the fastest fitting performance, whereas the Sklearn model required more time to process a complete image. Considering the results mentioned in Section 6.4, these models are capable of correcting depth information without fully exhausting the computational resources of devices with characteristics similar to those used in this work, while still maintaining the frame rate configured in the SDK initially.

Compared to the exponential decay approach, the proposed predictive methods offer greater flexibility and adaptability for deployment on embedded platforms. However, all of these proposed methods require careful data preparation, which may lead to potential issues (such a overflows caused by the restricted numerical ranges of embedded hardware) if the underlying assumptions are not well understood. Additionally, the implementation of artificial neural networks demands more specialized knowledge, and

Table 7. Comparison of the time performance of each model on the Jetson Nano, based on the elapsed time for each measurement

Frame	Exp. Dec. FPS	Pol FPS	Sklearn FPS	ANN FPS
1	25.15	26.03	19.98	28.49
5	27.30	29.80	29.32	28.28
10	25.37	26.42	25.85	25.55
15	27.44	29.62	29.23	27.46
20	27.99	26.30	25.94	26.82
25	30.26	29.76	29.15	30.01
30	28.02	26.45	26.04	29.85
Mean	27.48	27.92	27.28	28.05

their training process can significantly increase development time.

Although the exponential decay provides a straightforward (and simpler implementation) solution, the implementation of predictive models offers greater potential for integration into more complex systems, as they allow for tailored preprocessing and data adaptation strategies.

7 Conclusion and Future Work

Image processing and correction are key elements for achieving fully autonomous navigation, as these techniques enhance the mapping of diverse environments. Such approaches are particularly valuable in systems where replacing a defective sensor is not feasible, such as in space missions or IoT implementations, thereby extending the lifecycle of these projects. Evaluating different correction methods also provided insights into their relative performance, revealing that the main distinction among simple correction techniques lies in their implementation. All of them are capable of improving the accuracy of the information at different levels without demanding significant computational resources.

Future research may focus on optimizing each model, for instance, by reducing the polynomial degree through an evaluation of confidence levels for each order, or by refining the architecture of the artificial neural network.

Additionally, future work will include the analysis of images captured during real flight tests, which is expected to provide deeper insight into the errors introduced by outdoor conditions and platform instabilities, such as vibrations, variable lighting, and continuous obstacle perception in volatile environments.

Acknowledgments

We sincerely appreciate the support of Secihti in providing financial assistance through a master's scholarship for the first author.

References

1. **Aceves Ramírez, A. J., Altamirano-Robles, L. (2025).** Real-Time image analysis using a depth camera for UAV applications. DOI: 10.1007/978-3-031-96255-4{19.
2. **Alam, M. M. M., Nuruzzaman, M. (2025).** Comparative analysis of interpolation methods commonly used in bathymetric data generation. World Journal of Advanced Research and Reviews, Vol. 25, No. 3, pp. 2018–2029. DOI: 10.30574/wjarr.2025.25.3.0835.
3. **Haque, A. U., Nejadpak, A. (2017).** Obstacle avoidance using stereo camera.
4. **He, Y., Chen, S. (2020).** Error correction of depth images for multiview time-of-flight vision sensors. International Journal of Advanced Robotic Systems, Vol. 17, No. 4. DOI: 10.1177/1729881420942379.
5. **Holt, K. (2023).** Amazon plans to start drone deliveries in the UK and Italy next year.
6. **Kalidas, A. P., Joshua, C. J., Md, A. Q., Basheer, S., Mohan, S., Sakri, S. (2023).** Deep reinforcement learning for vision-based navigation of uavs in avoiding stationary and mobile obstacles. Drones, Vol. 7, No. 4. DOI: 10.3390/drones7040245.
7. **Krzeszowski, T., Dziadek, B., FranÃ§a, C., Martins, F., Gouveia, R., Przednowek, K. (2023).** System for estimation of human anthropometric parameters based on data from kinect v2 depth camera. Sensors, Vol. 23, No. 7. DOI: 10.3390/s23073459.

8. **Li, M., Huang, J., Xue, L., Zhang, R. (2023).** A guidance system for robotic welding based on an improved YOLOv5 algorithm with a RealSense depth camera. Scientific Reports, Vol. 13, No. 1. DOI: 10.1038/s41598-023-48318-8.
9. **Li, Z., Peng, Z., Zhang, Z., Chu, Y., Xu, C., Yao, S., García-Fernández, F., Zhu, X., Yue, Y., Levers, A., Zhang, J., Ma, J. (2023).** Exploring modern bathymetry: A comprehensive review of data acquisition devices, model accuracy, and interpolation techniques for enhanced underwater mapping. Frontiers in Marine Science, Vol. Volume 10 - 2023. DOI: 10.3389/fmars.2023.1178845.
10. **Masiero, A., Sofia, G., Tarolli, P. (2020).** Quick 3d with uav and tof camera for geomorphometric assessment. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. XLIII-B1-2020, pp. 259–264. DOI: 10.5194/isprs-archives-XLIII-B1-2020-259-2020.
11. **Park, H., Kim, K. (2020).** Depth image correction for intel realsense depth camera. Indonesian Journal of Electrical Engineering and Computer Science, Vol. 19, pp. 1021. DOI: 10.11591/ijeecs.v19.i2.pp1021-1027.
12. **Park, J., Choi, A. J. (2023).** Vision-based in-flight collision avoidance control based on background subtraction using embedded system. Sensors, Vol. 23, No. 14. DOI: 10.3390/s23146297.
13. **Parrot (2022).** Parrot Selected by the U.S. Defense Innovation Unit as a major drone supplier. https://www.parrot.com/en/newsroom/parrot-diu-blue-suas. Online, accesed 25 August 2025.
14. **Qiao, X., Ge, C., Yao, H., Deng, P., Zhou, Y. (2020).** Valid depth data extraction and correction for time-of-flight camera. **Osten, W., Nikolaev, D. P.**, editors, Twelfth International Conference on Machine Vision (ICMV 2019), International Society for Optics and Photonics, SPIE, Vol. 11433, pp. 114332K. DOI: 10.1117/12.2557533.
15. **Realsense (2024).** Post-processing filters. https://dev.realsenseai.com/docs/post-processing-filters. Online, accesed 18 November 2025.
16. **Tensorflow (2024).** Basic regression: Predict fuel efficiency. www.tensorflow.org/tutorials/keras/regression. Online, accesed 25 August 2025.
17. **Vasudevan, E., N. Sridhara, S., Pavez, E., Ortega, A., Singh, R., Kalluri, S. (2024).** Color-guided flying pixel correction in depth images. DOI: 10.48550/arXiv.2410.08084.
18. **Watt, J., Borhani, R., Katsaggelos, A. K. (2020).** Feature engineering and selection. Cambridge University Press, 2 edition.
19. **Xie, Z., Zhao, M., He, Y. (2023).** Research on depth information acquisition of array camera based on depth map fusion. 2023 5th International Conference on Frontiers Technology of Information and Computer (ICFTIC), pp. 759–762. DOI: 10.1109/ICFTIC59930.2023.10456113.
20. **Zhang, J., Li, S., Luo, Z., Fang, T., Yao, Y. (2022).** VIS-MVSNET: Visibility-Aware Multi-view Stereo Network. International Journal of Computer Vision, Vol. 131, No. 1, pp. 199–214. DOI: 10.1007/s11263-022-01697-3.
21. **Åke Björck (2024).** Basic Numerical Methods, chapter 2. pp. 39–114. DOI: 10.1137/1.9781611977950.ch2.

Article received on 03/09/2025; accepted on 24/11/2025.
**Corresponding author is Arturo Javier Aceves Ramírez.*