ISSN 2007-9737

Rocket Thrust Vectoring Attitude Control based on Convolutional Neural Networks

Rodolfo Garcia-Rodriguez*, Ivan Martinez-Perez, Luis E. Ramos-Velasco, Mario A. Vega-Navarrete[1]

Universidad Politécnica Metropolitana de Hidalgo,
Postgraduate Program in Aerospacial Engineering,
Mexico

{rogarcia, lramos, mvega}@upmh.edu.mx

Abstract. Launching and landing rockets on the Earth and in space have had intensive research and development in the last years. The idea of reducing costs is related regularly to the reusability of some mission stages. Though the launch has attracted attention due to including the main engines fundamental to boosting spacecraft onto an orbital or interplanetary trajectory, the landing takes relevance in space missions. While the rocket's landing has been carried out on Earth using an autonomous spaceport drone ship, it is challenging to design intelligent model-free systems that can continuously learn and compensate for slight deviations until they meet the target. This paper focuses on studying vertical rocket landing using convolutional neural networks. Assuming that the rocket is near the landing area, an attitude rocket control is proposed using a vision system to recognize it and drive -the nozzle TVC. Experimental results show the attitude control commanded by a nozzle TVC of an experimental rocket under different conditions.

Keywords. Rocket thrust vector control, convolutional neural networks, attitude rocket control.

1 Introduction

In the last years, there has been an increased need to reduce the costs in the space launches through spacecraft reusable. In this way, many companies, like the US company Space X, have focused their efforts on the vertical landing where the rocket's first stage should be reusable because it is higher cost.

In particular, the Vertical Takeoff Vertical Landing, VTVL, technology has been used successfully recently by companies as SpaceX (Falcon 9), Blue Origin (New Shepard) for reusable rockets. However, although VTLV has been demonstrated to be helpful, some open problems remain open in vertical takeoff technology and reusable landing rockets.

Since the beginning, the rocket's landing has attracted attention on how to get precision landing on the Earth and a more challenge in space where the parachute does not seem the best option [1].

In the literature, the rocket landing has been studied in two ways: a) as the reentry problem in the atmosphere where the high acceleration, dynamic pressure, and heating rate represents constraints of the optimization problem where the aerodynamic drag is fundamental to get deceleration effect and generates the reference rocket trajectory, and b) vertical landing problem where the altitude and speed of the rocket are controlling adjusting engine trust in a vacuum environment [15].

Due to the near of the landing area, the aerodynamic forces are in equilibrium with the gravity force; the remaining mass of the rocket defines the landing velocity. Thus, the rocket's altitude, speed, and attitude are adjusted to satisfy the landing time. This paper focuses on the vertical rocket landing on a floating landing platform known as the Autonomous Spaceport Droneship (ASDS), where precision is highly challenging.

Mainly to increase the rocket landing precision and divert or move sideways, the vectored thrust, TVC, is considered [3, 10, 6]. Finally, due to the

Table 1. Image augmentation parameters

Train_datagen / Evaluate_datagen	
Rescale	1./255
Rotation_range	40,
Width_shift_range	0.2
Height_shift_range	0.2
Shear_range	0.2
Zoom_range	0.2
Horizontal_flip	True

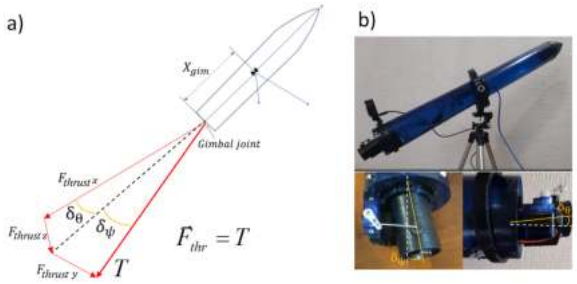

Fig. 1. (a) A TVC on the rocket, (b) Experimental prototype

Fig. 2. Images from the landing platform dataset

landing precision capabilities vary on the Earth and space, this problem implies the researches and developments of many fields as structural dynamics, engines, control systems taken place simultaneously. The rocket landing problem is studied and solved as an optimization problem under different constraints where the goal is to generate online flight trajectory [12, 8, 2].

The principal drawback of these approaches is the lack of knowledge of some parameters, aerodynamics or physics, in certain conditions, and robustness to compensate the parameters uncertainty, the wind perturbations, or noises that can affect the rocket's landing.

Recently, advances in machine learning tools, like deep learning, have been used to solve some problems in aircraft or rockets where the principal goal is a parameter estimation from the available data and improve the capacity of parameters estimation through a learning algorithm. The learning algorithms aim to estimate an unknown mapping from available data to predict future data, commonly known as generalization.

Deep Learning is part of Machine Learning, where information processing is carried out in hierarchical layers. That is, the deep learning algorithms are based on neural networks [14]. In addition to the traditional neural network approximating the unknown mapping from the input-output data, the deep learning algorithms can learn basic features from the input data in a deep learning algorithm.

They have the capacity to understand some characteristics of the new data that are useful to make predictions. The interesting feature of deep neural networks is that the learning capacity is related to hidden layers number [7]. Some approaches have been developed taking advantage of the deep learning algorithms as in [13] where deep neural networks are used to obtain the Hamilton-Jacobi-Bellman solution to guarantee aircraft landing, lunar landing in [5], and rocket launching in [17].

Furthermore, in [16] a deep convolutional neural network is used to process and classify a huge number of images data generated by optimal equipment. This paper uses a deep learning algorithm called Convolutional Neural Network, CNN, to learn and identify the landing platform to control attitude rocket.

The CNN is a feedforward neural network which by convolutional layers applied filters to the input data to obtain features from it, called feature learning. The application of CNN has

Table 2. Fit generator parameters

Training configuration using ImageDataGenerator	
train datagen.flow from directory	target size=(128,128) / batch size=64 /class mode='binary'
validation datagen. flow from directory	target size=(128,128) / batch size=64 / class mode='binary'
model.fit generator	train generator / epochs=30 / steps per epoch=63 / validation data=validation generator / validation steps=7 / workers=4

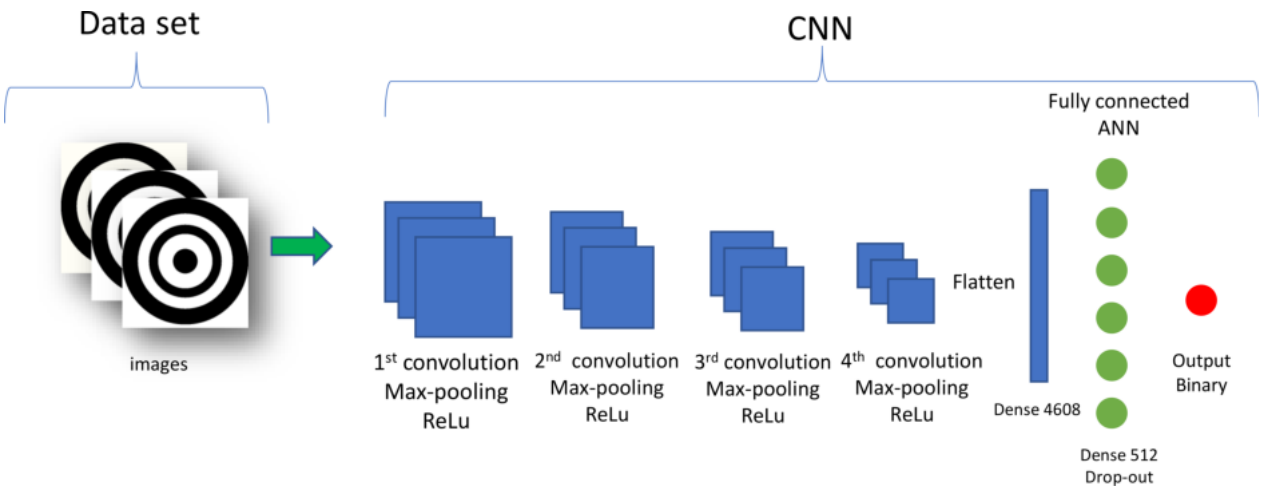

Fig. 3. Architecture of the CNN

been extended recently in many areas as computer vision, speech recognition, machine translation, to name a few. Motivating from concerns mentioned above, the vertical rocket landing, and the learning capabilities of the CNN, the problem to solve is defined as follows.

1.1 Problem Statement

Although the rocket has many phases of the return-to-launch-site mission, we focus on the vertical landing problem, specifically on the rocket's orientation near the landing area. Due to the aircraft does not have a straight path to its destination.

On the contrary, regularly, there is a slight deviation concerning the angle route or trajectory. The problem stands for the attitude rocket control avoiding any knowledge system.

Thus, to get autonomous and intelligent space landings, a vision system will drive a nozzle TVC to guarantee the attitude rocket control while the CNN is trained to recognize the landing platform.

Assuming that the landing platform is moving and the rocket is fixed in its center of gravity, in this paper, a landing platform dataset is built, and the CNN is trained to identify it.

Once a vision system identifies a target in its field of view, FOV, a bounding box is created around it while simultaneously, the coordinates of the center of the landing platform are obtained.

ISSN 2007-9737

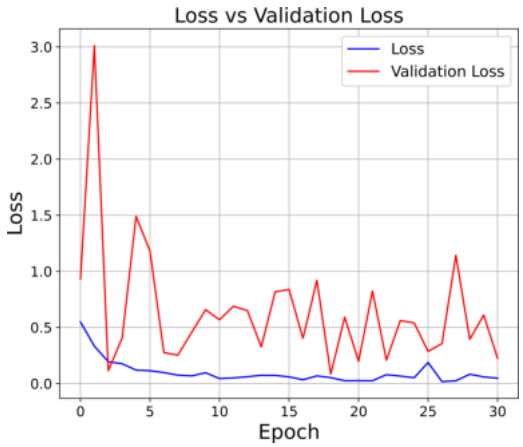

Fig. 4. Loss curve (1484 images)

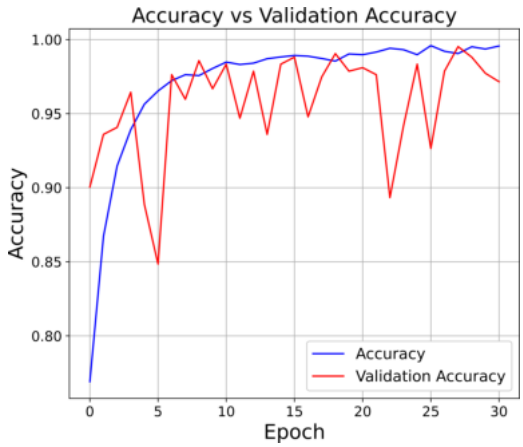

Fig. 5. Accuracy curve (1484 images)

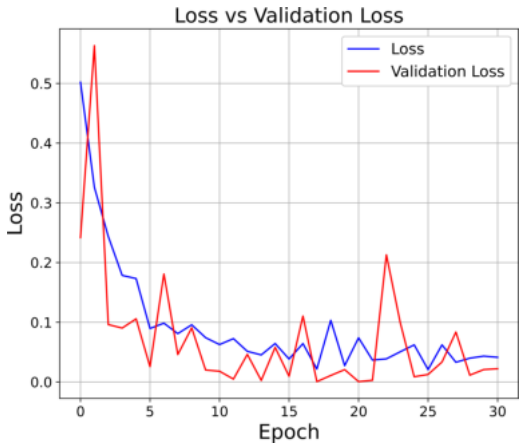

Fig. 6. Loss curve (8 feature extraction layers)

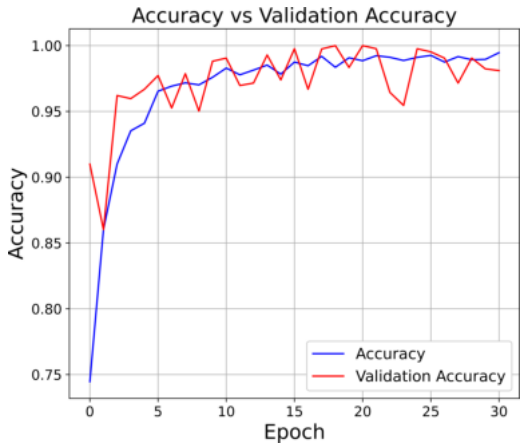

Fig. 7. Accuracy curve (8 feature extraction layers)

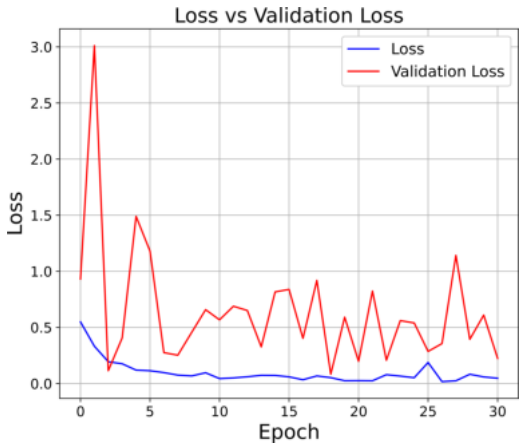

Fig. 8. Loss curve (742 images)

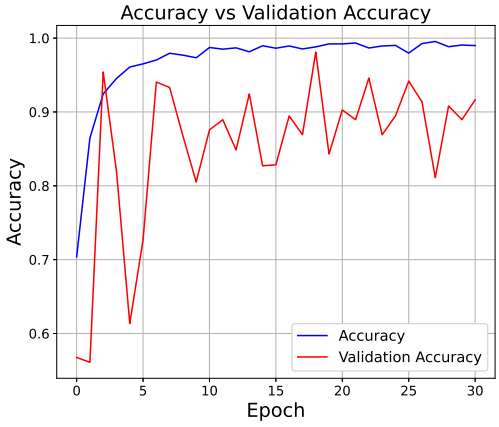

Fig. 9. Accuracy curve (742 images)

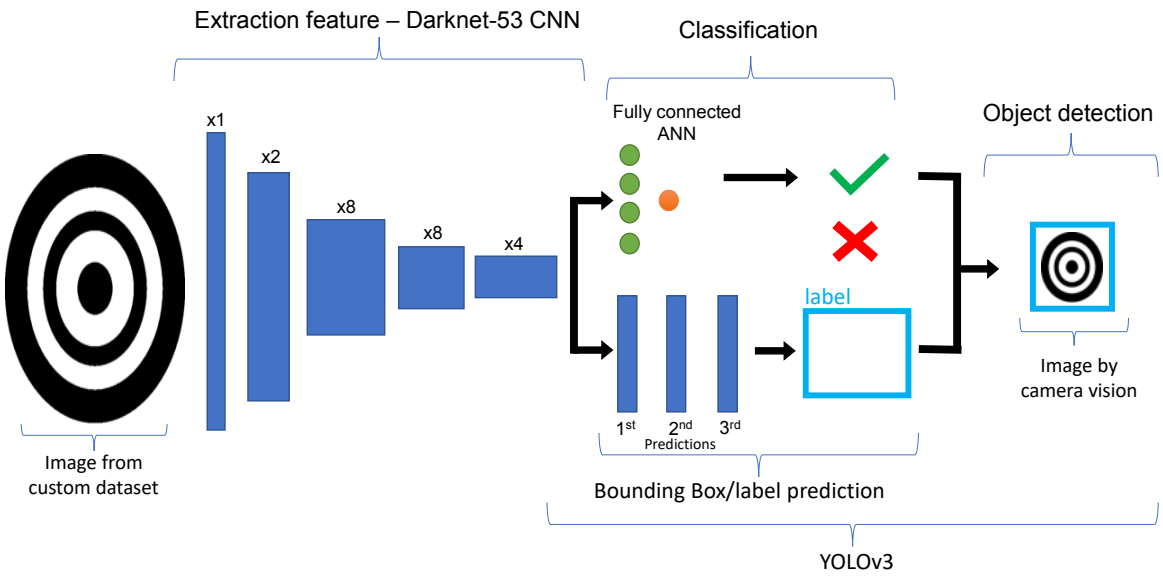

Fig. 10. Landing platform detection algorithm using Yolov3

```
--- [Epoch 29/30, Batch 135/136] ----
+------------+--------------+--------------+--------------+
| Metrics    | YOLO Layer 0 | YOLO Layer 1 | YOLO Layer 2 |
+------------+--------------+--------------+--------------+
| grid_size  | 11           | 22           | 44           |
| loss       | 2.444641     | 1.354754     | 1.698880     |
| x          | 0.007233     | 0.023857     | 0.095619     |
| y          | 0.004091     | 0.065025     | 0.088745     |
| w          | 0.110043     | 0.011270     | 0.122568     |
| h          | 0.010075     | 0.004914     | 0.011742     |
| conf       | 2.312531     | 1.248507     | 1.379636     |
| cls        | 0.000668     | 0.001181     | 0.000569     |
| cls_acc    | 100.00%      | 100.00%      | 100.00%      |
| recall50   | 0.500000     | 0.500000     | 0.500000     |
| recall75   | 0.500000     | 0.500000     | 0.500000     |
| precision  | 0.500000     | 0.333333     | 0.111111     |
| conf_obj   | 0.472997     | 0.545002     | 0.530408     |
| conf_noobj | 0.002855     | 0.001043     | 0.001118     |
+------------+--------------+--------------+--------------+
Total loss 5.498274803161621
--- ETA 0:00:00
```

Fig. 11. Object detection: Training results

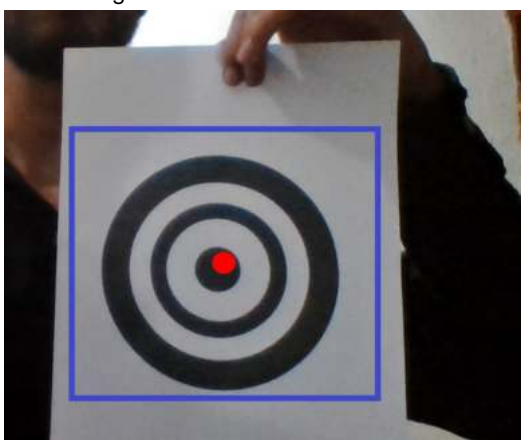

Fig. 12. Landing platform detection on the FOV

Finally, the orientation desired angles are defined, taking as reference the FOV of the vision system, which nozzle TVC uses to guide the appropriate rocket.

Specifically, the nozzle TVC system used on the rocket prototype has two orientations commanded by electromechanical actuators each one, see Fig. 1.

The propulsion used is constant, where a portable air compressor pump supplies it.

The remainder of this paper is organized as follows.

Section 2 presents the description of training and testing of the CNN to get an attitude rocket control by vision system.

Experimental results are presented under different conditions in Section 3.

Finally, some conclusions are presented in Section 4.

ISSN 2007-9737

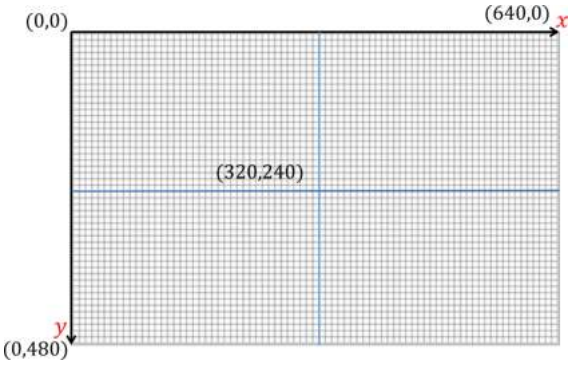

Fig. 13. Reference systems of FOV

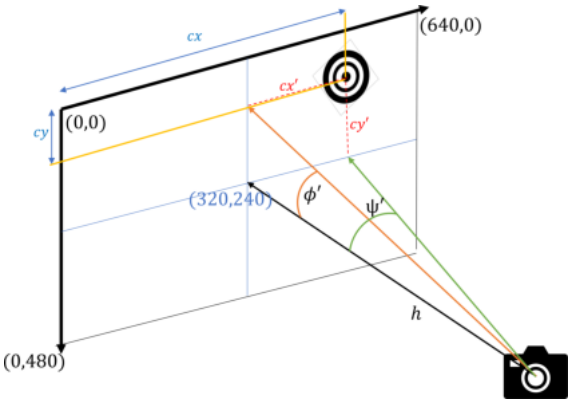

Fig. 14. Obtaining desired angles from FOV

Fig. 15. Experimental prototype

2 Approach

This section presents the different stages to control the rocket attitude using CNN as part of its landing back mission.

Stages as training of CNN, landing platform recognition by the vision system, and the generation of the orientation desired angles are developed in Anaconda Python distribution under Windows©10 operating system. The principal packages installed are Tensorflow 1.13.1, Keras 2.3.1, and OpenCV-python 4.4.0.

2.1 Training of CNN

The first step before CNN training is to build a landing platform dataset.

2.1.1 Landing Platform Database

The landing platform dataset was built using a mobile phone with a resolution of 25 MP. The dataset images have the same size, are labeled, and within them, the landing platform is not centered. The landing platform database has 742 images, where 372 are positive images with different degrees of illumination and photo angles to make the learning and detection more robust, while 370 negative images of anything; other than the target. In Fig. 2 we can see a sample of the database created. Once the landing platform has been built, the next step is to train the CNN to recognize the landing platform and use it to rocket control orientation.

2.1.2 Convolutional Neural Network

The training goal of the CNN is to get a valuable set of weights that allow recognizing the landing platform and the generalization ability of the model. Finally, the set of weights is saved for testing the learning. Before training the CNN, the image augmentation technique is applied to the dataset images to get multiple transformed copies of an image.

The image augmentation technique will be helpful to generalize the model due to adding variations levels of unseen data and avoiding model overfitting. The Image data generator is a class of Keras that configure and convert the database into useful data to be entered in CNN. Finally, each new batch of our data is adjusted randomly.

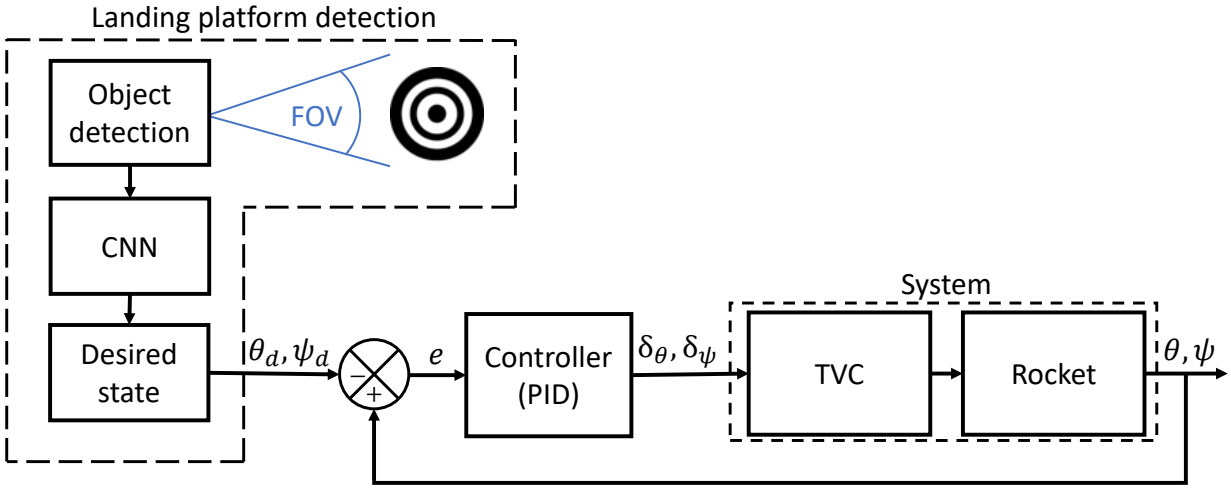

Fig. 16. Rocket thrust vectoring control system using convolutional neural network, CNN

In Table 1 are shown the augmentation parameters apply to the dataset images while parameters of the Flow_from_dataframe and Keras Fit_generator method are shown in Table 2. The former directly augment images by reading its name and target value from a dataset, and the latter accepts a batch of the dataset updates the model's weights.

Now, we are ready to train our CNN. The CNN uses an image 224 $\times$ 224 pixels, four feature extraction layers, and two fully connected neural networks in the output layer, see Fig. 3. The training model was carried out using 30 epochs with a batch size set to 64.

The optimization algorithm used was the RMSprop with a learning rate set to 0.0003. The optimization algorithm used was the RMSprop with a learning rate set to 0.0003. Finally, the results of the training using augmenting images database are given in Fig. 8 and Fig. 9 with the loss and accuracy curves, respectively.

It is observed from Fig. 8 that the training loss curve moves down until it reaches a value of approximately 0.1, while validation loss has an expected downward trend but in some epochs display peaks. These peaks represent the images that the model never saw and cannot recognize.

On the other hand, the training and validation accuracy curves, Fig. 9, tend to increase which means the training is adequate even to the small number of epochs used for the training. Thus, the neural network has trained to generalize the images from loss and accuracy curves.

Remark. Fig. 4 and 5 show the training and validation results using 1,848 images. The performance is better than in Fig. 8-9 but appears some peaks that represent that there are images that cannot be learned for the CNN. By another hand, if the feature extraction layers are increased to eight, the performance of the training and validation significantly improves, see Fig. 6-7. Thus, the features extraction layers are fundamental in the learning process to predict future data accuracy. Once the CNN is trained, the next step is to landing platform recognition using the vision system.

2.2 Rocket Thrust Vectoring Control Using CNN

2.2.1 Landing Platform Recognition

Learning an object or image using CNN is generally used for classification or categorization tasks where the object recognized within an image is fundamental.

ISSN 2007-9737

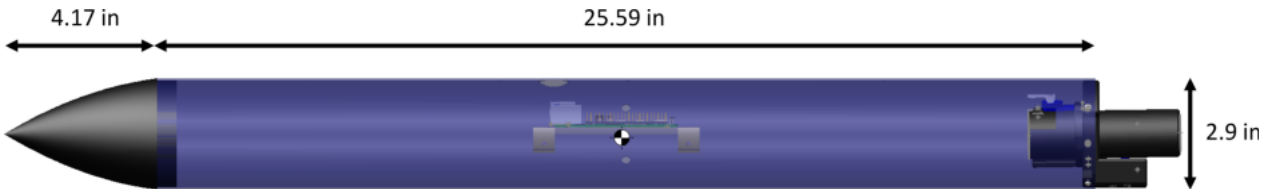

Fig. 17. Experimental rocket dimensions

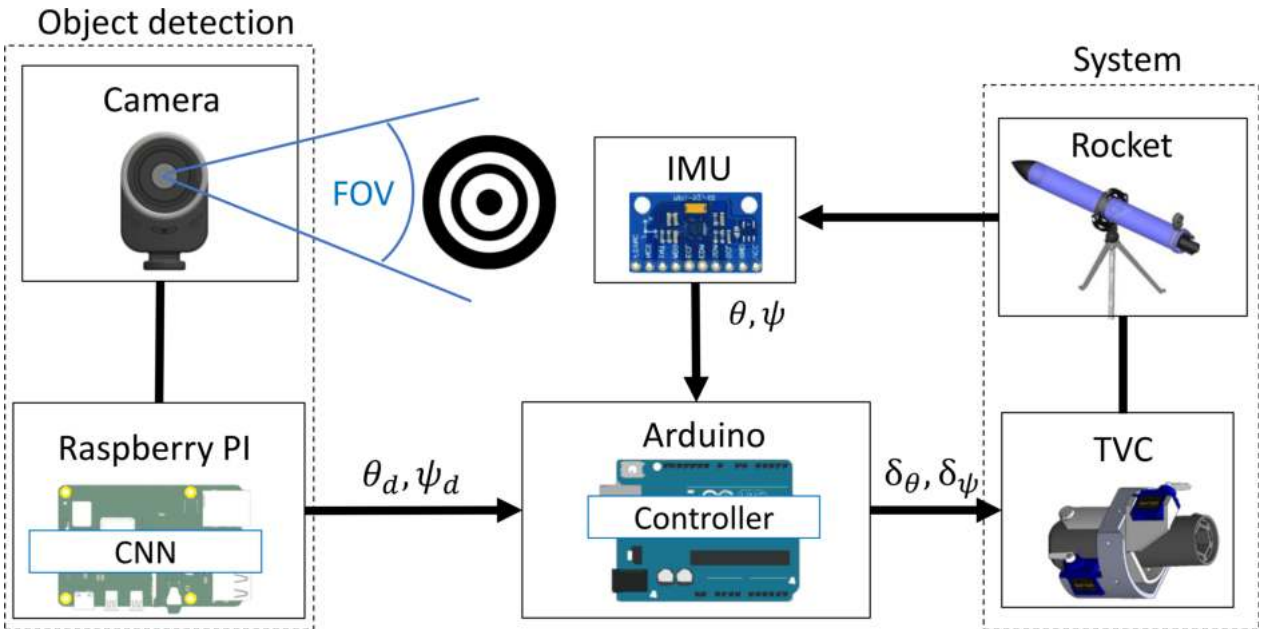

Fig. 18. General scheme used to generate the TVC signals

In this paper, we are interested, in addition, to knowing the exact location of the landing platform, see Fig. 12. Thus, the images from the dataset were labeled with coordinates of the object's location embedded into the images. To creates the labels is used the labelImg application. It is a graphical image labeling application tool written in Python where the coordinates are manually saved in a TXT file for each image.

Finally, in conjunction with labels, this dataset is trained to obtain its location in the camera's field of view. To get the object detection[1] real-time object detection algorithm named Yolov3 is used where a Darknet-53 CNN is in charge of learning and classifying the desired image, Fig. 10.

Thus, simultaneously the bounding box regressor[2] and the label's prediction are made in the case of having several classes (different objects/images) [4, 9, 11].

The bounding box uses residual blocks algorithms for better performance in tiny images, also Intersection Over Union, IOU, techniques to get a perfect box over the object. In addition, it uses three prediction scales to get better performance, while the labeling uses binary cross-entropy loss because there are many overlapping labels. Given that CNN complete training used in YoloV3 takes much computational time and requires significant computational power, a transfer learning technique is used to reduce neural network training time.

[1]The process of recognizing and getting its location is called object detection.

[2]Box that marks the target's location (landing platform) on the image.

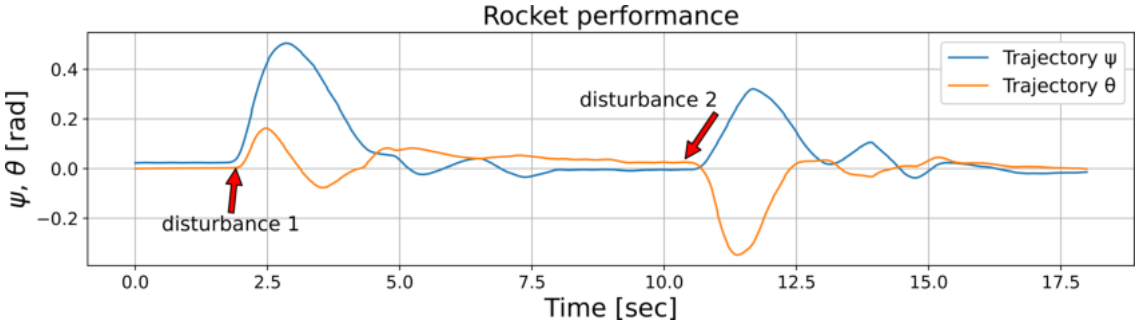

Fig. 19. Experiment 1: Performance of θ and ψ angles under different perturbations

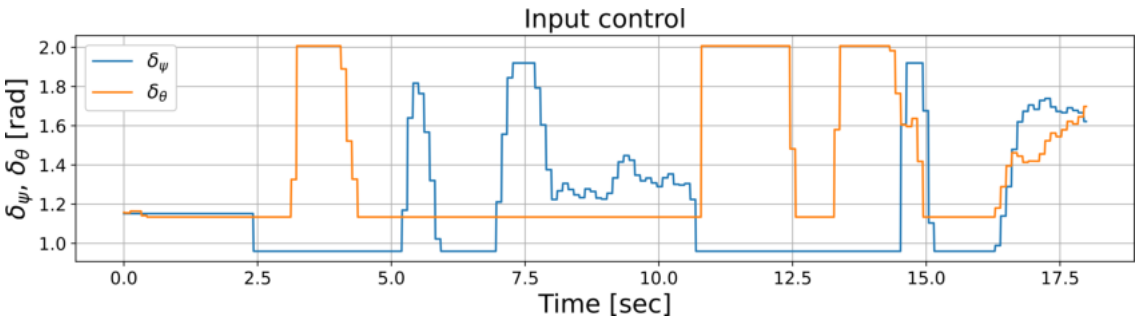

Fig. 20. Experiment 1: Control signals:δ_θ and δ_ψ

Thus, using a pre-trained CNN that contains the information to recognize faster generic features from any image, the last three layers of the CNN are trained to recognize a particular object, in this case, the landing platform.

That is, the pre-trained weights remain fixed while the training takes place. The training model parameters used for the complete training of the object detection were 30 epochs with a batch size set to 8. The optimization algorithm used was the MSEloss. Fig. 11 show the training results.

2.2.2 Generating the Orientation Desired Angles

Once the artificial vision system recognizes the landing platform and its location, it is possible to calculate the desired orientation of the rocket angles. Let the coordinates of the field of view be defined according to Fig. 13, where the origin is placed in the upper left corner, the horizontal axis has 640 pixels, and the vertical axis has 480 pixels.

Thus, using the landing platform coordinates is possible to calculate the angles concerning each of the axes of the field of view, see Fig. 14. Such that θ_d and ψ_d are given as:

$$\theta_d = \tan^{-1}\left(\frac{cy'}{h}\right) = \tan^{-1}\left(\frac{cy - 480/2}{h}\right), \quad (1)$$

$$\psi_d = \tan^{-1}\left(\frac{cx'}{h}\right) = \tan^{-1}\left(\frac{cx - 640/2}{h}\right), \quad (2)$$

where $(cx,\ cy)$ are the coordinates from the origin (0,0) to the landing platform detected, $(cx',\ cy')$ are the coordinates from the median axes, with origin in (320, 240), to the landing platform, and h is the distance from the rocket's camera to the target where for this application is assumed constant.

Notice that the angles concerning each of the axes of the field of view, (1)-(2), are no more than the desired angles to be used rocket attitude control.

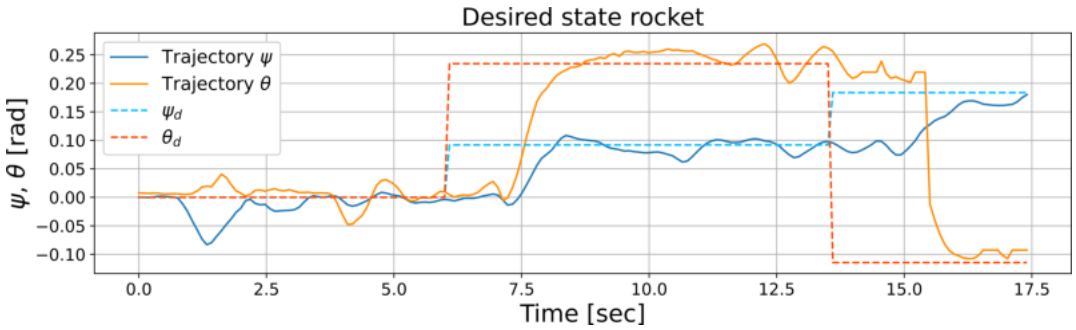

Fig. 21. Experiment 2: Desired and actual rocket angles

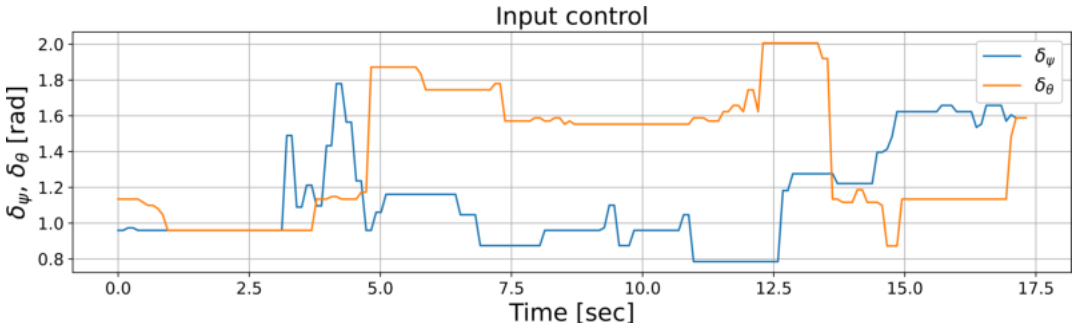

Fig. 22. Experiment 2: Control signals: δ_θ and δ_ψ

2.2.3 Rocket Thrust Vectoring Control

Finally, once the desired angles θ_d and ψ_d are defined, we can rocket attitude control using nozzle TVC. The general control scheme proposed is shown in Fig. 16. For simplicity in this paper, we consider a traditional PID controller on a process variable (PV) and a setpoint in discrete form defined as:

$$\delta_*(t_k) = P + I + D, \tag{3}$$

where:

$$P = K_P\, e(t_k),$$

$$I = K_I \sum_{i=1}^{k} e_i(t_k)\Delta t_k,$$

$$D = K_D \frac{\Delta e(t_k)}{\Delta t_k},$$

where K_P, K_I, K_D are the feedback gains, $e(t_k) = *(t_k) - *(t_K)_d$ is the tracking error at time step t_k with $*(t_k)$ the actual state and, $*(t_k)_d$ the desired state, and Δt_k is the step size. In this case the PID controller is utilized to control the states θ, and ψ. Finally, Fig. 15 shows the prototype model rocket used for experiments. More technical details about the prototype is given in the next section.

3 Experimental Results

3.1 Experimental Setup

To demonstrate the performance of the proposed scheme an experimental results were carried out on a fixed at the center of gravity rocket, see Fig. 15. The rocket comprises a nose cone, body tube, and TVC nozzle with two degrees of freedom.

The experimental rocket principal dimensions are shown in Fig. 17 where the fineness of the nose is 1.39, and its approximated total weight is 615 g. As the rocket is fixed at the center of gravity, as propulsion is used a portable air compressor pump with pressure around 300 KPa, obtaining a thrust of 6.2×10^{-3} kN.

To identify the landing platform, on the rocket is placed the vision system through a web camera, Genius Qcam 6000 of 2MP. In addition, the OpenCV library is installed on a Raspberry Pi 4, 8Gb RAM, located inside the body tube to recognize the landing platform. Finally, CNN is programming on Raspberry Pi 4, too. On the other hand, an IMU MPU-6050 is used to carry out the rocket orientation measurement: θ, and ψ angles. Thus, once the actual angles and the coordinates landing platform are available, the nozzle TVC control signals are calculated on Arduino UNO, also inside the body tube. See Fig. 18.

Remark. The desired angles information in sent from the Raspberry Pi to Arduino through SERIAL communication (SSH). Although this communication kind produce a delay, this communication has a type of switch which causes the sent data to pile up, the attitude control is carried out adequately. Due to data Arduino arriving in a non-existent format, a delay of 3 sec. is programmed on the Raspberry.

3.2 Experimental Conditions

The objective of the experiments is to rocket thrust vectoring control orientation; that is, the rocket should be oriented according to the landing platform that is assumed is moving.

In the first experiment, the goal is to test that the TVC nozzle can maintain the rocket in equilibrium positions under different perturbations, that is, $\theta = \psi \approx 0$. Fig. 19 shows the performance of the θ and ψ angles when the rocket is subject to two manual perturbations. Notice that, after the induced perturbations, the TVC nozzle can stabilize the rocket, that is, $\theta = \psi \to 0$.

In Fig. 20 is shows the control signals applied to the TVC nozzle. In the second experiment, the vision system on the rocket is used to recognize and follow the landing platform using a TVC nozzle. The CNN is used to acknowledge and calculate the landing platform coordinates representing the desired angles, θ_d, and ψ_d.

Thus, the controller's goal of the TVC nozzle is to generate the control signals to get the desired rocket orientation. Fig. 21 shows how once the desired angles are generated, the controller produces the adequate control signals applied to the TVC that guarantee to reach the desired angles. Notice that exists a delay of around 7 s between each sending of coordinates. This is due to delay communication between Arduino and Raspberry. Therefore, the rocket reaches the desired angles despite this delay and noise on orientations measurements. Finally, in Fig. 22 the control signals performance of θ_d and ψ_d are show.

4 Conclusion and Future Work

The vertical rocket landing using CNN, especially the attitude control by nozzle TVC without knowing the system or landing trajectory, is presented. Assuming that the landing platform is moving, the rocket's orientation is driven by a vision system that recognizes the target, controlling the rocket orientation by the nozzle TVC. In addition, the building dataset, training, and testing of the CNN is presented. Although experimental results are done under a controlled environment, the rocket is fixed on the center of gravity. The results show that the proposed scheme is robust enough to latency, noise in sensors, and motors, to control rocket orientation using CNN. In this way, the CNN can be considered part of an intelligent system for launching and landing tasks where the learning characteristics from new data to make predictions without any knowledge of the system or desired trajectory can be helpful from space missions.

Acknowledgments

This work was partially supported by CONACYT Scholarships with reference number 0055803.

References

1. **Blackmore, L. (2016).** Autonomous precision landing of space rockets. Frontiers of Engineering, Vol. 46, No. 4, pp. 15–20.

2. **Blackmore, L., Açikmeşe, B., Scharf, D. P. (2010).** Minimum-landing-error powered-descent guidance for mars landing using convex optimization. Journal of Guidance, Control, and Dynamics, Vol. 33, No. 4, pp. 1161–1171. DOI: 10.2514/1.47202.
3. **Devlin, T., Dickerhoff, R., Durney, K., Forrest, A., Pansodtee, P., Adabi, A., Teodorescu, M. (2018).** ElbowQuad: Thrust vectoring quadcopter. AIAA Information Systems-AIAA Infotech @ Aerospace, pp. 8–12. DOI: 10.2514/6.2018-0893.
4. **Farhadi, A., Redmon, J. (2018).** Yolov3: An incremental improvement. Computer Vision and Pattern Recognition, pp. 1804–2767. DOI: 10.48550/ARXIV.1804.02767.
5. **Furfaro, R., Bloise, I., Orlandelli, M., Di-Lizia, P., Topputo, F., Linares, R. (2018).** Deep learning for autonomous lunar landing. Master's thesis, Politecnico di Milano and University of Arizona.
6. **Knuth, D. E. (1992).** Literate programming. , No. 27, pp. 97–111.
7. **Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012).** Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, Curran Associates, Inc., Vol. 25.
8. **Liu, X., Shen, Z., Lu, P. (2016).** Entry trajectory optimization by second-order cone programming. Journal of Guidance, Control, and Dynamics, Vol. 39, No. 2, pp. 227–241. DOI: 10.2514/1.g001210.
9. **Mao, Q. C., Sun, H. M., Liu, Y. B., Jia, R. S. (2019).** Mini-YOLOv3: Real-time object detector for embedded applications. IEEE Access, Vol. 7, pp. 133529–133538. DOI: 10.1109/access.2019.2941547.
10. **Oates, G. C. (1984).** Aerothermodynamics of gas turbine and rocket propulsion. American Institute of Aeronautics and Astronautics. DOI: 10.2514/4.861345.
11. **Redmon, J., Divvala, S., Girshick, R., Farhadi, A. (2016).** You only look once: Unified, real-time object detection. Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788. DOI: 10.48550/arXiv.1506.02640.
12. **Sagliano, M., Mooij, E., Theil, S. (2017).** Onboard trajectory generation for entry vehicles via adaptive multivariate pseudospectral interpolation. Journal of Guidance, Control, and Dynamics, Vol. 40, No. 2, pp. 466–476. DOI: 10.2514/1.G001817.
13. **Sánchez-Sánchez, C., Izzo, D. (2018).** Real-time optimal control via deep neural networks: Study on landing problems. Journal of Guidance, Control, and Dynamics, Vol. 41, No. 5, pp. 1122–1135. DOI: 10.2514/1.G002357.
14. **Vasilev, I., Slater, D., Spacagna, G., Roelants, P., Zocca, V. (2019).** Python deep learning: Exploring deep learning techniques and neural network architectures with PyTorch, Keras, and TensorFlow. Packt Publishing.
15. **Wang, C., Song, Z. (2019).** Trajectory optimization for reusable rocket landing. Chinese Automation Congress, pp. 3052–3057. DOI: 10.1109/CAC48633.2019.8997476.
16. **Zhang, L., Chen, Z., Wang, J., Huang, Z. (2018).** Rocket image classification based on deep convolutional neural network. 10th International Conference on Communications, Circuits and Systems, pp. 383–386. DOI: 10.1109/ICCCAS.2018.8769176.
17. **Zhou, G., Fan, Y., Cui, R., Bian, W., Zhu, X., Gai, K. (2018).** Rocket launching: A universal and efficient framework for training well-performing light net. DOI: 10.48550/arXiv.1708.04106.

Article received on 30/08/2022; accepted on 15/04/2024.
** Corresponding author is Rodolfo Garcia-Rodriguez.*