Transport and Communications Science Journal

DETECTION AND LOCALIZATION OF HELIPAD IN AUTONOMOUS UAV LANDING: A COUPLED VISUAL-INERTIAL APPROACH WITH ARTIFICIAL INTELLIGENCE

Hoang Dinh Thinh*, Le Thi Hong Hieu

Department of Aerospace Engineering, Faculty of Transportation Engineering, Ho Chi Minh City University of Technology (HCMUT), 268 Ly Thuong Kiet Street, District 10, Ho Chi Minh City, Vietnam

ARTICLE INFO

TYPE: Research Article
Received: 24/7/2020
Revised: 25/9/2020
Accepted: 28/9/2020
Published online: 30/9/2020
https://doi.org/10.47869/tcsj.71.7.8

* *Corresponding author*
Email: hoangdinhthinh@hcmut.edu.vn; Tel: 0987365488

Abstract. Autonomous landing of rotary wing type unmanned aerial vehicles is a challenging problem and key to autonomous aerial fleet operation. We propose a method for localizing the UAV around the helipad, that is to estimate the relative position of the helipad with respect to the UAV. This data is highly desirable to design controllers that have robust and consistent control characteristics and can find applications in search – rescue operations. AI-based neural network is set up for helipad detection, followed by optimization by the localization algorithm. The performance of this approach is compared against fiducial marker approach, demonstrating good consensus between two estimations.

Keywords: artificial intelligence, machine learning, localization, UAV, landing.

1. INTRODUCTION

Unmanned Aerial Vehicles have become an essential force in development of smart cities and are playing a more prominent role in various economic and social activities, such as tele-sensing, agriculture, package delivery and aerial photography. Despite recent advances in sensor, control and mass deployment of artificial intelligence on-board, autonomous landing is a challenging problem that associates with significant risk of aircraft loss and is key to the fully autonomous UAV fleet operation, which is advantageous in the employment of UAV for continuous missions, such as food and package delivery, atmospheric information collection.

The landing of a rotary wing UAV on a helipad is challenging can be attributed to the difficulty in localizing the landing target to a precision level that often exceeds what can be delivered by satellite-based navigation systems. This is particularly difficult for small UAVs, operating in urban area where localization signal is interfered due to surrounding constructions. Solutions often rely on visible light cameras as they are available en-mass onboard UAVs today. Compared to other sensors such as RADAR, LIDAR, SONAR… the camera is compact, low-cost but it also requires a lot of computation in order to run computer vision algorithms on-board. These algorithms should be robust to fluctuating ambient lighting condition and adaptable to different helipad designs, while maintaining low computational complexity as a too complex algorithm may burn out the power and computational resources of a typical UAV computer – which is often very limited.

In this paper, we address a problem of detecting the helipad using RGB images from a visible light camera and infer the localization information, which include the relative distance from the UAV to the helipad. This information is in real-world metric scale and highly desirable for feedback control of UAVs, as opposed to projected pixel distance on the image plane of the camera. The latter suffers from the scale problem and gives different controller performance for different UAV altitude. We also make use of the aircraft attitude which is given by an Inertial Measurement Unit (IMU) filtered data – carried out by either Extended Kalman Filter or Complementary Filter instead of inferring this data from the helipad pattern (such as fiducial marker as helipad approaches). This makes the approach much more simple while retaining the effectiveness.

2. RELATED WORK

Several approaches are available regarding the detection of landing targets, including autonomous landing using specific and non-specific targets. For specific target methods, [1], [2] and [3] proposed specially designed helipad involving patterns of colors and a specialized object detector to detect the position of the helipad in the image. A PID controller then regulates the position of the UAV based on distance to the helipad in the image plane to zero. In [4], two colored discs are used as a landing target, which can be detected by a blob detector and 2 color filters. In [5], a non-specific landing target is proposed which is a box with an X letter in it. Instead of using color filters, the paper turned to detector that employed local features. This approach is much more robust to variance in ambient lighting and also to arbitrary scale and rotation. However, if the image contrast is not sufficient, the approach might suffer from degradation in performance as not enough features are captured to match with the predefined template.

In [6], the authors used a number of AprilTag, which is a fiducial marker family that is designed for improved processing time and estimation accuracy for camera's pose. The measurement data extracted from camera images is augmented with IMU, fused together by an Extended Kalman Filter.

Recently, with the progress of Machine Learning, object detection has reached new standards thanks to the extreme robustness of convolutional neural networks (CNNs) to ambient lighting, scale, rotation, perspective transformation and even distortions. The network

can learn from simple features to more advanced, abstract features that present in the template. In [7], a single convolutional network was used for both object detection and UAV control tasks and achieved an impressive success rate of around 80%. A popular CNN network design called YOLOv3 was reconfigured and applied to object detection, coupled with a profile checker for validation against false positives and a Kalman filter to improve tracking performance [8].

3. PROBLEM FORMULATION

3.1. Frames

We denote the notation of frames we shall use in this article. The camera equipped on the drone is positioned as downward facing will be characterized by frame C whose origin stays at the center of the image plane with X axis pointing to the left hand side, Y axis pointing downward and Z axis pointing forward, away from the camera. The body frame is centered at the IMU, with X axis pointing forward, Y axis pointing to the right and Z axis pointing downward. The inertial frame will be denoted as I, which follows the North-East-Down (NED) convention and placed at the helipad. We denote another inertial frame It which is the frame aligned with the ARUCO tag [9] whose origin placed at the tag, X axis pointing to the right hand side and Z axis pointing upward, away from the tag. Finally, for convenience, we denote I' a 180o rotation of It around X axis. Henceforth, if we further assume the IMU and the camera lie on the planes parallel to each other, we yield the following relations:

$$R_B^C = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}, R_{I_t}^{I'} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}, R_{I'}^{I} = \begin{bmatrix} \cos\xi & -\sin\xi & 0 \\ \sin\xi & \cos\xi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The angle θ, ξ can be obtained via a calibration process, which will be detailed in another paper.

3.2. Problem

Given the video stream from the camera I(t), accelerometer $a^B(t)$ and gyroscope reading $\omega^B(t)$ from the IMU, find the relative position of the helipad, that is the vector $\overrightarrow{HC}(t)^I$ expressed in the I frame.

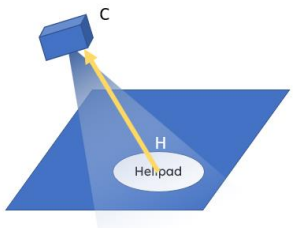

Figure 1. Problem formulation.

4. HELIPAD DETECTION

Due to the nature of the autonomous landing problem, observation of helipad is conducted from different perspective and distance, resulting in significant distortion of the helipad with perspective, scale and ambient lighting conditions. Among many template matching methods, object detection by Deep Learning has made great strides in recent years and achieved state-of-the-art result. In [8], the authors have demonstrated that the helipad can be detected even in low light conditions, which make deep learning a very appealing approach for this problem.

We base our approach on YOLOv3 [10] paper, but we also made some small changes. First, we use a Tiny YOLO configuration with 7 convolutional layers and 1 upsample. However, because the helipad needs to be recognized from different scales and many features might present in variety of sizes, we decided on having 3 YOLO layers to achieve more robust detection, the same approach found in the full-size YOLO configuration. We also reduce the number of anchor boxes to to two per YOLO layer, thus along with 3 YOLO layers yielding a total of 6 anchors to speed up training and detection time as we want the network capable of running real-time on Raspberry Pi hardware. The final network architecture is shown in Figure 2. In the figure, *axbsc* denotes the convolutional layer of a filters, size b and stride c. The "+" layer is the residual layer and X2 is upsampling. The output for prediction is the YOLO layer, which is evident that there are three of them, handling anchor boxes at different scales.

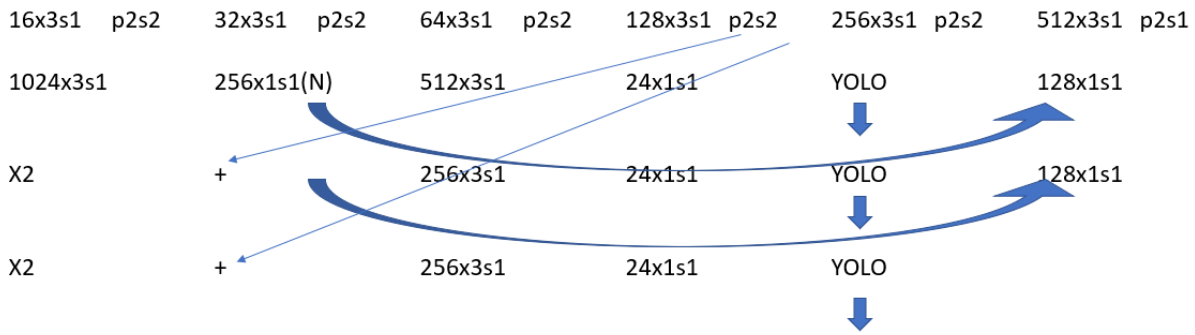

Figure 2. Customized YOLOv3 Configuration with 3 YOLO layers.

Note that this customized network has only 1 class of object: the helipad. The training data is obtained through an experimental device and labelled by hand, using the YOLOLabel tool from [11]. A sample image dataset was created using the prototyping device (described later) with 283 images, 183 of which were captured in sufficient lighting condition and the rest were captured in poor lighting condition. The images were captured from different perspectives and distance, and unsurprisingly with the images captured in poor lighting condition, images with a lot of motion blur. The dataset is then split into two sets: one for training and one for validation with the ratio of 7:3 respectively.

We use DarkNet with PyTorch from Ultralytics [12] with ADAM optimizer and train from scratch with Google Colab (Tesla T4 GPU) for 200 epochs with batch size of 64. The training took place in 13 minutes, the result is shown in Figure 3.

The network exhibits a very good precision and recall characteristics during validation, both exhibiting near 0.9. The final GioU is 0.361 with mean average precision at 0.5 around 0.995. The classificiation score is unnecessary since only one class of object is involved.

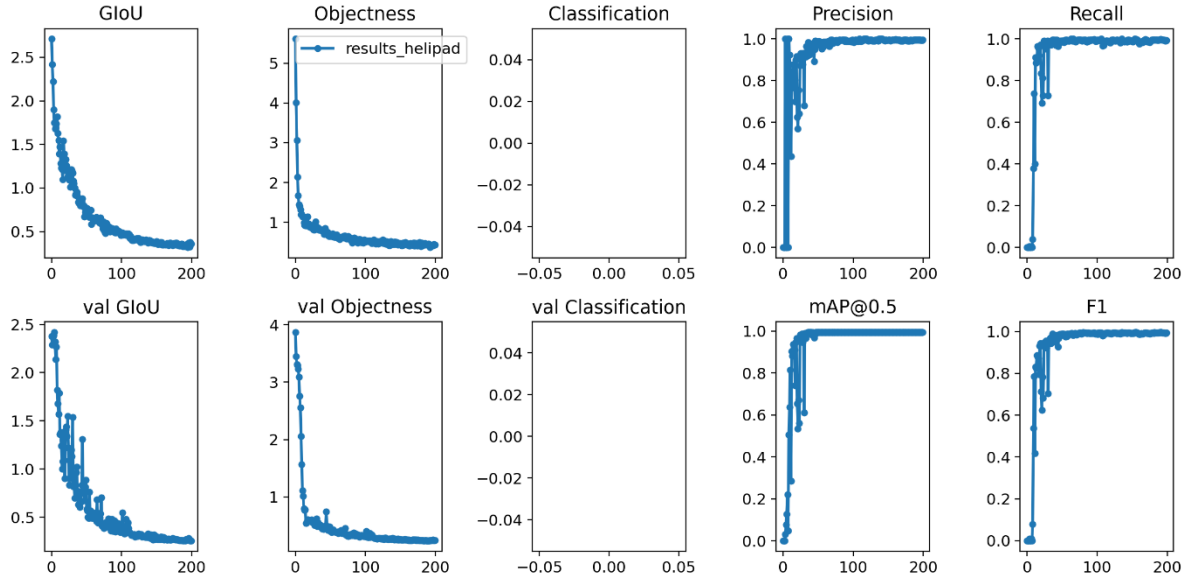

Figure 3. Training of the YOLOv3 Network.

Figure 4. Example of Object Detection by YOLO.

Figure 4 shows an example of helipad detection in an image captured of a helipad with radius 18.1cm.

5. LOCALIZATION

In the conventional machine vision based navigation with fiducial markers like AprilTag or ARUCOTag, the tag must provide enough information on the pose of the camera, which

includes relative position and the camera' attitude, denoted by a rotation matrix $R_I^C \in SO(3)$. However, typical machine learning approaches only give a bounding box of where the object is in the image, without telling anything about the camera's attitude, the depth of the object as well as relative position. To this aspect, we propose a method to estimate these parameters with help from an Inertial Measurement Unit, which is typically equipped on-board many modern UAVs.

From the Pin-hole camera model equations:

$$\pi\left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}\right) = \begin{bmatrix} \frac{fX}{Z} \\ \frac{fY}{Z} \end{bmatrix} = \begin{bmatrix} x_e \\ y_e \end{bmatrix}, \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = R_I^C \left(\begin{bmatrix} x_l - x_C \\ y_l - y_C \\ z_l - z_C \end{bmatrix}\right)$$

With f: the focal length of the camera lens. Let $x_l - x = \bar{x}, y_l - y = \bar{y}, -z = \bar{z}$, we have:

$$R_I^C \begin{bmatrix} \bar{x} \\ \bar{y} \\ \bar{z} \end{bmatrix} = \begin{bmatrix} Zx_e / f \\ Zy_e / f \\ Z \end{bmatrix} \tag{1}$$

Where $R_I^C = R_B^C R_I^B$. Because R_I^B is known from the Euler angles by fusing IMU accelerometer, gyroscope and magnetometer reading, for example by an Extended Kalman Filter [13]. If we denote:

$$R_I^C = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix}$$

From the last row of (1), the depth of the object is:

$$Z = a_3\bar{x} + b_3\bar{y} + c_3\bar{z}$$

Substituting this result into the remaining two rows of (1):

$$a_1\bar{x} + b_1\bar{y} + c_1\bar{z} = (a_3\bar{x} + b_3\bar{y} + c_3\bar{z})x_c / f$$
$$a_2\bar{x} + b_2\bar{y} + c_2\bar{z} = (a_3\bar{x} + b_3\bar{y} + c_3\bar{z})y_c / f$$

Or both can be rewritten as:

$$\begin{aligned} \left(a_1 - \frac{a_3 x_c}{f}\right)\bar{x} + \left(b_1 - \frac{b_3 x_c}{f}\right)\bar{y} + \left(c_1 - \frac{c_3 x_c}{f}\right)\bar{z} = 0 \\ \left(a_2 - \frac{a_3 y_c}{f}\right)\bar{x} + \left(b_2 - \frac{b_3 y_c}{f}\right)\bar{y} + \left(c_2 - \frac{c_3 x_c}{f}\right)\bar{z} = 0 \end{aligned} \tag{2}$$

Which we shall call as the projection constraints as they express the constrain of the projection map π. Now if we assume the two upper left (which we will call point 1) and bottom right (point 2) points of the bounding box (Figure 4) belong to the actual object, and that the helipad is lying flat on the ground $z = 0$, we yield the following equations:

$$\begin{aligned}
&\left(a_1-\frac{a_3x_{c1}}{f}\right)\overline{x_1}+\left(b_1-\frac{b_3x_{c1}}{f}\right)\overline{y_1}+\left(c_1-\frac{c_3x_{c1}}{f}\right)\overline{z}=0\\
&\left(a_2-\frac{a_3y_{c1}}{f}\right)\overline{x_1}+\left(b_2-\frac{b_3y_{c1}}{f}\right)\overline{y_1}+\left(c_2-\frac{c_3x_{c1}}{f}\right)\overline{z}=0\\
&\left(a_1-\frac{a_3x_{c2}}{f}\right)\overline{x_2}+\left(b_1-\frac{b_3x_{c2}}{f}\right)\overline{y_2}+\left(c_1-\frac{c_3x_{c2}}{f}\right)\overline{z}=0\\
&\left(a_2-\frac{a_3y_{c2}}{f}\right)\overline{x_2}+\left(b_2-\frac{b_3y_{c2}}{f}\right)\overline{y_2}+\left(c_2-\frac{c_3x_{c2}}{f}\right)\overline{z}=0
\end{aligned} \tag{3}$$

Additional constraint is required for unique solution within bounds, which we will call as the scale constraint as it resolves the arbitrary scale problem by relating the dimensions of the helipad on the image plane with the real-world metric dimensions:

$$(\overline{x_2}-\overline{x_1})^2+(\overline{y_2}-\overline{y_1})^2=R^2 \tag{4}$$

In which, R is the diameter of the helipad. The main source of error for estimation depends on whether the backprojected point of the the top-left and bottom-right points of the bounding box, stay close to the helipad.

From (3) and (4), it is now possible to solve for $\overline{x_1}, \overline{y_1}, \overline{x_2}, \overline{y_2}, \overline{z}$ through any nonlinear optimization algorithm. In our case, we prefer the Trust Region Reflective method due to its robustness and fast convergence.

6. EXPERIMENTAL RESULT AND DISCUSSION

Figure 5. Prototyping device.

A prototyping device (Figure 5) which comprises of a Raspberry Pi 3B (1GB model) and a TDK InvenSense MPU9250 was made. The IMU consists of two dies, each houses a 3-axis gyroscope and a 3-axis accelerometer respectively. We use the RTIMULib2 library for communication with the MPU and utilize the I2C communication. The gyroscope was configured to yield output at approximately 100Hz in the range of 500deg/s while the accelerometer's range was set to 4g. Further specifications of the IMU can be found in [14].

PiCamera library obtained burst shots from a Pi Camera V2. For the experiment setup, we place the helipad and an ARUCO tag side by side to compare results obtained from our algorithm and ARUCO tag's pose estimator included in the OpenCV library [15]. A sample taken from the dataset can be found on Figure 6.

Figure 6. Tag and Helipad Setup.

From the extrinsic camera matrix formulation:

$$K = [R^C_{I_t} \mid t], R^C_{I_t} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = R^C_{I_t} TC^{I_t} = t$$

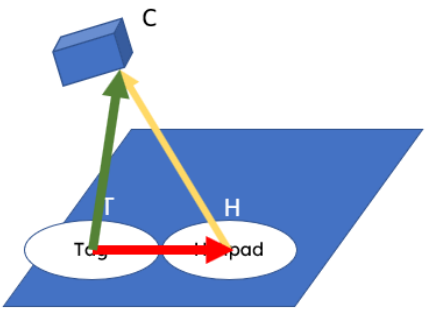

Figure 7. Helipad and Tag.

From Figure 7:

$$\begin{aligned} \overrightarrow{CH}^I &= \overrightarrow{HT}^I + \overrightarrow{TC}^I \\ CH^I &= R^C_{I_t} HT^{I_t} + R^I_{I_t} TC^{I_t} \\ CH^I &= R^C_{I_t} HT^{I_t} + R^I_{I_t} R^{I_t}_C t \end{aligned} \tag{5}$$

(5) relates the estimation of the pose from ARUCO tag $[R^C_{I_t} \mid t]$ and the position in frame I, which should be the same as estimation from Section 5. After collection of 45 seconds of trajectory in adequate lighting condition (brightness approximately 600lux), the helipad is detected with a customized YOLOv3 in Section 4 and processed for localization information inference in Section 5. The comparison between the trajectories is depicted in Figure 8.

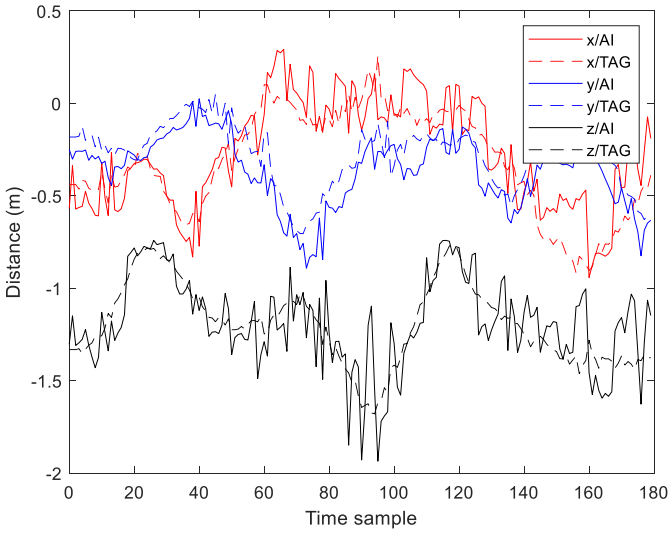

Figure 8. Comparison between AI localization and localization by ARUCO tag in adequate lighting condition.

Overall, the trend of the AI inferred position and from ARUCO tag closely match with each other. It is noteworthy that the detection from AI tends to be much noisier, since the bounding box size is not consistently accurate. An Euclidean norm of the error between the two estimations revealed that the peak error is around 0.58m, while the low is less than 10cm. The mean is 0.22m and the distribution of error shows non-specific distribution, with 90th percentile of error is 0.3568m 95th percentile is 0.4855m.

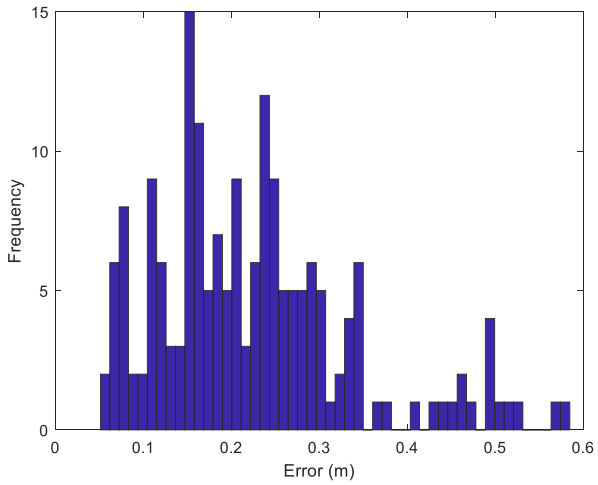

Figure 9. Distribution of error between 2 localization methods.

Another experiment was conducted in poor lighting condition, with the average brightness of approximately 50lux. The camera compensated by setting longer exposure time, resulting in a blurrier image induced by motion. Nevertheless, the YOLOv3 detector still exhibited very strong performance with no missed frames. However, the accuracy degrades a little bit, with estimation error less than 0.612m 90% of time. Figure 11 shows the good

agreement between the two estimators (AI and ARUCOTag), with AI estimation tends to be noisier.

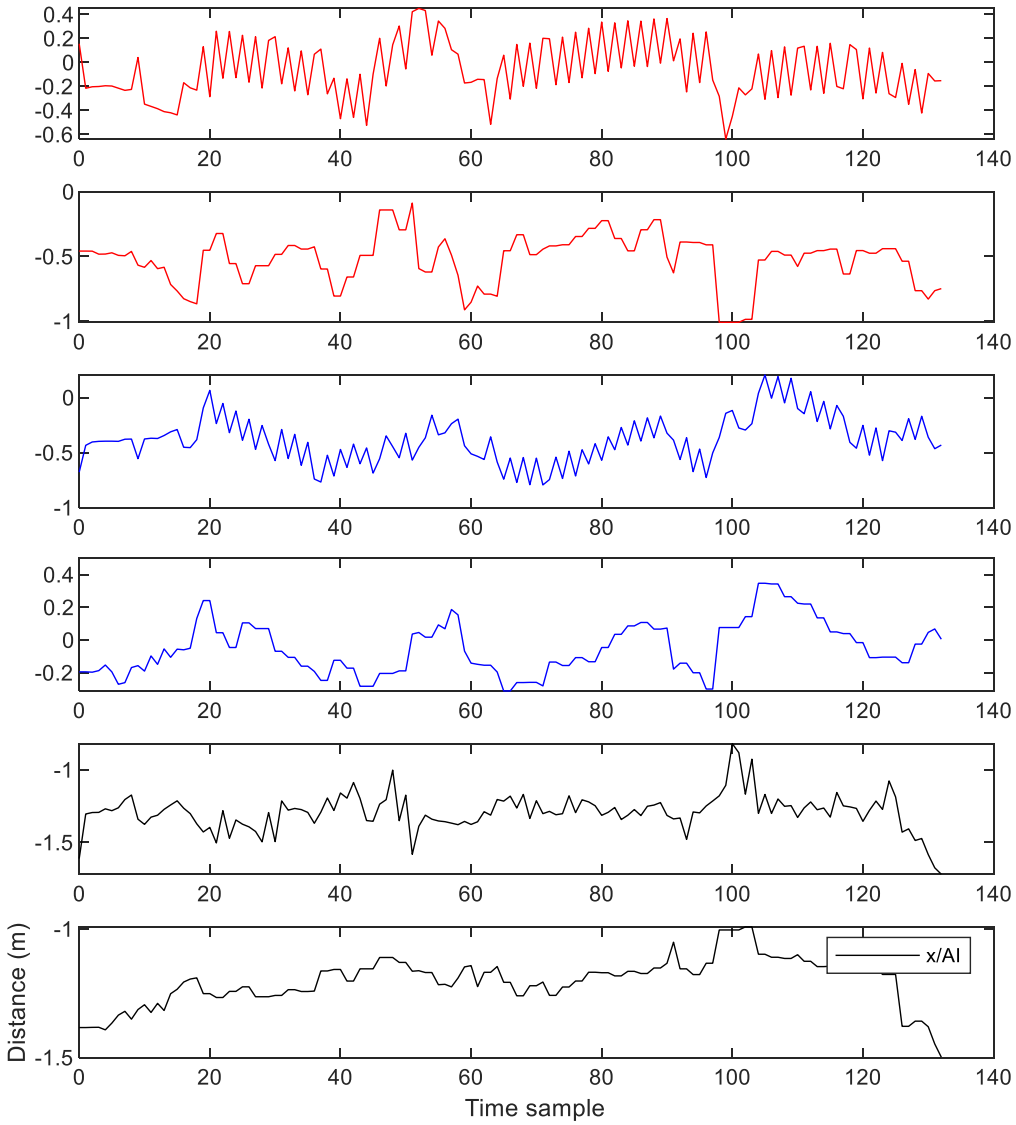

Figure 10. Comparison between AI localization and localization by ARUCO tag in poor lighting condition. From top to bottom: X coordinate (AI), X coordinate (ARUCOTag), Y coordinate (AI), Y coordinate (ARUCOTag), Z coordinate (AI), Z coordinate (ARUCOTag).

It is thought that the sources of error can be traced to two reasons: inaccurate bounding box size and the backprojected top-left and bottom-right corners are not close to the helipad. The first relates to the IoU of the detection algorithm, while the second can be ameliorated by obtaining the convex hull of the helipad with the region of interested prescribed by the bounding box. It is from these two factors that lead to inaccuracy in the estimation of depth, which in turn propagates to the rest variables. Nevertheless, the algorithm, albeit simple, demonstrate good localization capability, as 90% of time, the error should be less than around

30cm. It is also worth to mention that the ARUCO's estimation is assumed to be ground-truth values here, but in reality it should come with some instability and inaccuracy too.

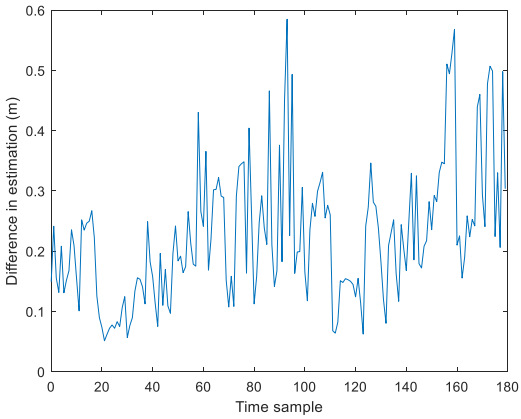

Figure 11. Euclidean norm of Error between 2 localization methods.

7. CONCLUSION

In this paper, we have presented a simple method for localization of the helipad for autonomous landing of a rotary wing type UAV using Artificial Intelligence for Object Detection. Experiments demonstrated that this is a plausible approach for localization of the helipad, and further work that involves designing controller and localization when helipad went missing can be pursuit.

ACKNOWLEDGEMENT

This research is funded by Ho Chi Minh City University of Technology (HCMUT), VNU-HCM under grant number T-KTGT-2019-73. We thank Google Colab for providing free GPU for the network training process, and the warm-hearted ophthalmologist, Mrs. Huynh Vo Mai Quyen M.D for her endless kindness and care for me during my difficult days of treatment.

REFERENCES

[1]. T. Venugopalan, T. Taher, G. Barbastathis, Autonomous landing of an Unmanned Aerial Vehicle on an autonomous marine vehicle, in 2012 Oceans, 2012, pp. 1-9. https://doi.org/10.1109/OCEANS.2012.6404893

[2]. A. B. Junaid, A. Konoiko, Y. Zweiri, M. N. Sahinkaya, L. Seneviratne, Autonomous wireless self-charging for multi-rotor unmanned aerial vehicles, Energies, 10 (2017) 803. https://doi.org/10.3390/en10060803

[3]. J. Kim et al., Autonomous flight system using marker recognition on drone, in 2015 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV), IEEE, 2015, pp. 1-4. https://doi.org/10.1109/10.1109/FCV.2015.7103712

[4]. R. Bartak, A. Hraško, D. Obdržálek, A controller for autonomous landing of AR. Drone, in The 26th Chinese Control and Decision Conference (2014 CCDC), IEEE, 2014, pp. 329-334.

https://doi.org/10.1109/CCDC.2014.6852167

[5]. M. Skoczylas, Vision analysis system for autonomous landing of micro drone, acta mechanica et automatica, 8 (2014) 199-203. https://doi.org/10.2478/ama-2014-0036

[6]. O. Araar, N. Aouf, I. Vitanov, Vision based autonomous landing of multirotor UAV on moving platform, Journal of Intelligent & Robotic Systems, 85 (2017) 369-384. https://doi.org/10.1007/s10846-016-0399-z

[7]. D. K. Kim, T. Chen, Deep neural network for real-time autonomous indoor navigation, arXiv preprint arXiv:1511.04668, 2015. https://arxiv.org/abs/1511.04668

[8]. P. H. Nguyen, M. Arsalan, J. H. Koo, R. A. Naqvi, N. Q. Truong, K. R. Park, LightDenseYOLO: A fast and accurate marker tracker for autonomous UAV landing by visible light camera sensor on drone, Sensors, 18 (2018) 1703. https://doi.org/10.3390/s18061703

[9]. F. J. Romero-Ramirez, R. Muñoz-Salinas, R. Medina-Carnicer, Speeded up detection of squared fiducial markers, Image and vision Computing, 76 (2018) 38-47. https://doi.org/10.1016/j.imavis.2018.05.004

[10].J. Redmon, A. Farhadi, Yolov3: An incremental improvement, arXiv preprint arXiv:1804.02767, 2018.

[11].Y. Kwon, (2018), Yolo_Label, Available: https://github.com/developer0hye/Yolo_Label

[12].Ultralytics, (2018), YOLOv3, Available: https://github.com/ultralytics/yolov3

[13].F. L. Markley, Attitude error representations for Kalman filtering, Journal of guidance control and dynamics, 26 (2003) 311-317. https://doi.org/10.2514/2.5048

[14].T. InvenSense. (2020). MPU-9250 Nine-Axis (Gyro + Accelerometer + Compass) MEMS MotionTracking™ Device. Available: https://invensense.tdk.com/products/motion-tracking/9-axis/mpu-9250/

[15].G. Bradski, The opencv library, Dr Dobb's J. Software Tools, 25 (2000) 120-125. https://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/reference/ReferencesPapers.aspx?ReferenceID=1692176