Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi
Vol. 11, No. 3, Juli 2022

ISSN 0216 – 0544
e-ISSN 2301– 6914

# OBSTACLE AVOIDANCE IN QUADCOPTER NAVIGATION USING MODIFIED LOCAL MEAN K-NEAREST CENTROID NEIGHBOR METHOD

[a]Hendy Prasetyo, [b]Trihastuti Agustinah

[a, b, c]Department of Electrical Engineering, Institut Teknologi Sepuluh Nopember (ITS)
Surabaya 60111, Indonesia
E-mail: hendyprasetyo3010@gmail.com, trihastuti@ee.its.ac.id

***Abstract***

*Quadcopter is a type of Unmanned Aerial Vehicle (UAV) technology, characterized by simple mechanical structure, ease of flying and good maneuvering. In its usage, the quadcopter is required to evade obstacles in its path. Thus, an obstacle avoidance system in a 3D space with both static and dynamic obstacles is. Avoidance direction is determined by considering nearest distance based on the dimensions of the obstacle. Due to limited battery capacity, the quadcopter also needs to consider energy efficiency in obstacle avoidance. The obstacle's properties and movement direction are also needed in considering the correct avoidance direction. Using a modified Local Mean K-Nearest Centroid Neighbor (LMKNCN) algorithm results in a 97.5% accuracy for avoidance direction decision. The learning process between training data and testing data yielded a computation duration of 0.142341 seconds. The simulations showed that the quadcopter is able to avoid static and dynamic obstacles to reach its destination without collisions.*

*Key words: Energy Efficient, Obstacle Avoidance, Machine Learning, Modified LMKNCN, Movement Trends, Quadcopter Navigation.*

## INTRODUCTION

A quadcopter is a type of unmanned aerial vehicle (UAV) with 4 rotors serving as lift and propulsion. A quadcopter's advantages compared to other UAV configurations include simple mechanical structure, ease of flying and maneuvering. These advantages allow the quadcopter to be used in many fields, such as farming [1], surveillance [2], construction [3], search and rescue [4], delivery [5], and such others.

In doing its assigned tasks, a given quadcopter must avoid many obstacles in its path. Its avoidance system must be adaptable towards both static and dynamic obstacles to make sure the quadcopter is safe and undamaged.

According to [6], avoidance direction decision needs to consider energy usage and distance due to limitations in the quadcopter's battery capacity. This research used K-Nearest Neighbor (KNN) algorithm for avoidance, using travel distance and minimum energy usage. This method is simple and effective in avoiding static obstacles of various dimensions However, there are still a possibility that the quadcopter hits a dynamic obstacle with certain speed.

Dynamic obstacle avoidance mechanism by predicting movement trends (static, dynamically to the left or right) to determine the robot's linear speed is discussed in [7]. The robot may avoid the obstacle by increasing and decreasing speed, or outright stopping to wait for the obstacle to pass. This fuzzy logic-based method was able to predict movement trends and assist the robot in choosing the correct speed to avoid the obstacle. The movement trend prediction concept can be adapted towards an avoidance system for dynamic obstacles of varying speeds and directions.

Building upon [6], there is also a need for predicting the obstacle's movement trends to alter the quadcopter's movement direction in addition to its linear speed, thus minimizing evasion travel distance. However, running both movement trend prediction and speed adjustment burdened the KNN algorithm used in [6], resulting in more errors and eventual evasion failures. Local Mean K-Nearest Centroid Neighbor (LMKNCN) algorithm, a development from KNN, was proved by [8]-[9] to have less classification errors than its predecessor, thus its inclusion in the proposed system.

The LMKNCN method was tested in [8]-[9] to require a longer calculation method than KNN, which may influence calculation time, especially with high amount of training data. Thus, the algorithm needs to be modified to modify the algorithm to reduce the amount of executed training data. Grouping training data into a few clusters was proposed by [10]-[11], with each group being represented by a single data point, usually taken from the clusters' midpoints. The results gained by [10]-[11] showed that this method may reduce computation time despite the large number of training data involved.

From the previously discussed research, this paper proposes an avoidance system using a modified LMKNCN algorithm. The modification involved clustering the training data to reduce the amount of data processed by the algorithm in a given time, thus reducing computation time in the classification process. With this modification, it is hoped that the quadcopter will be able to avoid static and dynamic obstacles with a fast, accurate avoidance decision while also minimizing distance and energy. The avoidance classification system has five classes, that is evading to the left, right, up, down, or stopping, based on the obstacle's movement trends and dimensions between the quadcopter and the obstacle.

## MATERIAL AND METHODS

This chapter discusses the system's concepts. The quadcopter unit used in the system is a Quanser Qdrone as seen in Fig 1. The obstacles used will be static and dynamic, while the avoidance system used the LMKNCN method to determine evasion direction.

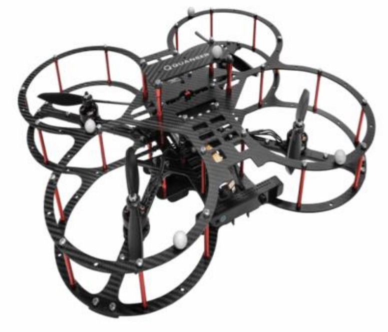


Fig 1. Quadcopter quanser qdrone

Table 1. Quanser qdrone parameters

| **Parameter** | **Symbol** | **Value** |
|---|---|---|
| Mass (kg) | $m$ | 1 |
| Gravity (kg/ m$^2$) | $g$ | 9.81 |
| Moment of inertia on the $X$ axis (kg.m$^2$) | $J_{xx}$ | 0.03 |
| Moment of inertia on the $Y$ axis (kg.m$^2$) | $J_{yy}$ | 0.03 |
| Moment of inertia on the $Z$ axis (kg.m$^2$) | $J_{zz}$ | 0.04 |
| Distance between rotor and center of mass (m) | $l$ | 0.2 |
| Drag force | $d$ | $3.13 \times 10^{-5}$ |
| Thrust force | $b$ | $7.5 \times 10^{-7}$ |
| *Bandwidth* actuator (*rad/*s) | $\omega$ | 15 |
| Thrust force constant (N) | $K$ | 120 |

### Quadcopter System

The Quanser Qdrone, a quadcopter designed for outdoor research, was chosen for this research. The quadcopter had a carbon fiber frame for a reduced weight, contributing to better maneuvering and less risk of catastrophic collisions. The drone has 40 cm × 40 cm ×15 cm dimensions and is equipped with propeller protectors.

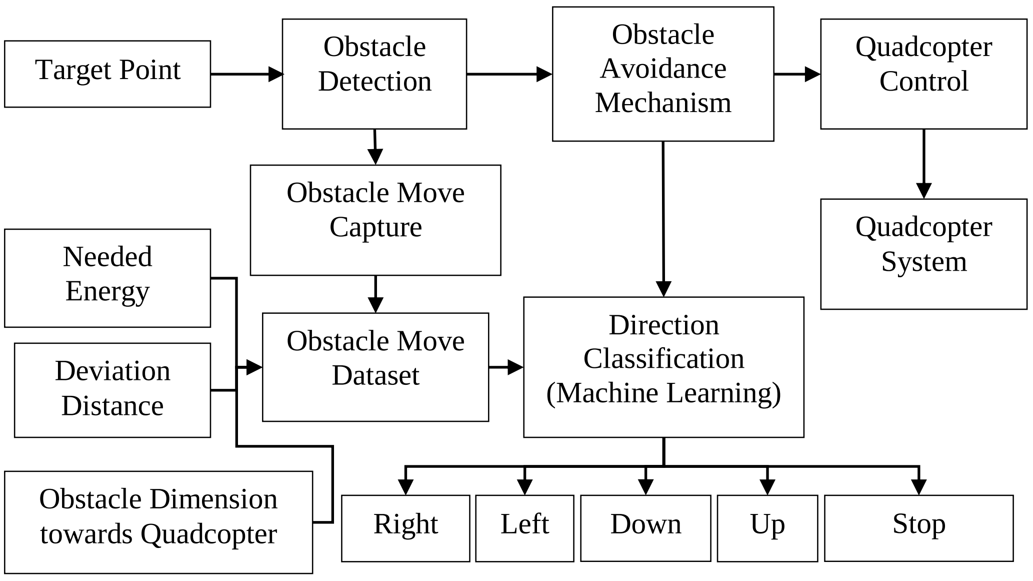


Fig 2. System scheme

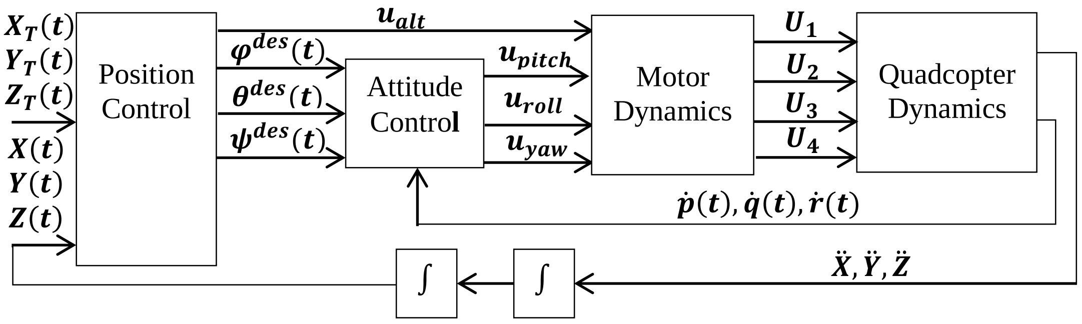


Fig 5. Quadcopter control system

$$
\begin{aligned}
\ddot{X} &= (\sin\psi \sin\varphi \\
&\qquad + \cos\psi \sin\theta \cos\varphi)\frac{U_1}{m} \\
\ddot{Y} &= (-\cos\psi \sin\varphi \\
&\qquad + \sin\psi \sin\theta \cos\varphi)\frac{U_1}{m} \\
\ddot{Z} &= -g + (\cos\theta \cos\varphi)\frac{U_1}{m} \\
\dot{p} &= \frac{J_{yy} - J_{zz}}{J_{xx}} qr + \frac{U_2 l}{J_{xx}} \\
\dot{q} &= \frac{J_{zz} - J_{xx}}{J_{yy}} pr + \frac{U_3 l}{J_{yy}} \\
\dot{r} &= \frac{J_{xx} - J_{yy}}{J_{zz}} pq + \frac{U_4 d}{J_{zz}}
\end{aligned} \tag{1}
$$

The quadcopter's system model [12] is represented in (1), where $X$, $Y$, $Z$ is the quadcopter's position while $p$, $q$, $r$ is the roll, pitch and yaw speed. The parameters used in the drone are shown in Table 1 depending on the type of drone. The system schema can be seen in Fig 2.

## Obstacle Detection

An obstacle may be detected at a minimum distance of 1 meter to the quadcopter, which may be static or dynamic. When an obstacle is detected, direction detection is done to determine its movement, (moving to the left, right, up, down, or static).

After detecting an obstacle, the quadcopter reads its dimensions $h_u$, $h_l$, $h_r$, and $h_d$ to classify the obstacle using the modified LMKNCN method. The classification results will be turned into waypoints for the quadcopter's path, changing its direction from the initial target and avoiding the obstacle [13].

## Classification Method

This system used a modified Local Mean K-Nearest Centroid Neighbor (LMKNCN) algorithm to reduce the number of calculations needing to be done. Once a dataset has been chosen according to the detected obstacle, the next step is grouping the data into clusters, each with a cluster point determined using the following equation:

$$Cp_i = \frac{1}{k}\sum_{j=1}^{k} x_j \tag{2}$$

where $Cp_i$ is the *i*-th cluster point, $k$ is the number of data points, and $x_j$ is a data point.

In the testing data process, the first step is finding the nearest neighbor between new data and some cluster points. To calculate the nearest neighbor distance the following equation was used:

$$d(x, Cp) = \sqrt{\sum_{i=1}^{n}(x_i - Cp_i)^2} \tag{3}$$

where $x$ is new data and $Cp_i$ is the *i*-th cluster point

After finding the nearest cluster, the next step is finding the nearest neighbor between the new data and data inside the cluster using LMKNCN [8]-[9]. The steps are as follows:

a) Find the nearest neighbor $q_1$ between testing data $x$ and training data in cluster $j$ $a_j^{Cp}$ using an Euclidean distance equation shown in (4).

$$d(x, a_j^{Cp}) = \sqrt{\sum_{i=1}^{n}(x_i - a_{ji}^{Cp})^2} \tag{4}$$

b) Find the nearest centroid neighbor $C_k$ between testing data and each centroid data. The new centroid data is gleaned from the midpoint between a nearest neighbor $q_1$ to the *k*-th neighbor.

$$C_k = \frac{1}{k}\sum_{j=1}^{k} q_j \tag{5}$$

After finding the nearest centroid neighbor, the next step is finding *k* nearest neighbors.

c) Calculating local average centroid vector $u_{ik}^{NCN}$ from each class $c_i$.

$$u_{ik}^{NCN} = \frac{1}{k}\sum_{j=1}^{k} x_{ij}^{NCN} \tag{6}$$

where $x_{ij}^{NCN}$ is the training data for each class $c_i$.

d) Finding the distance $d(x, u_{ik}^{NCN})$ between $x$ and the local average centroid vector $u_{ik}^{NCN}$ using (7).

$$d(x, u_{ik}^{NCN}) = \sqrt{\sum_{k=1}^{n}(x - u_{ik}^{NCN})^2} \tag{7}$$

e) Finding testing data $x$ into class $c$, having the nearest distance between local average centroid vector $u_{ij}^{NCN}$ and testing data $x$ as (8).

$$c = \arg\min_{c_i} d(x, u_{ij}^{NCN}) \tag{8}$$

## Deviance Distance

The obstacle will be detected by the quadcopter at a distance of 1 meter. The quadcopter will record the obstacle's dimensions ($h_u$, $h_l$, $h_r$, $h_d$) against the intersection between the quadcopter and the target point.

Fig 3 illustrates how a quadcopter would read each deviance distance, with a side view in Fig 4 (a) and a top view in Fig 4 (b). Each deviance distance ($\delta_{l,}$ $\delta_{r,}$ $\delta_u$, $\delta_d$) will be calculated using (9).

$$\boldsymbol{\delta_i = \tan\theta_i \,.\, d_i}$$
$$\theta_i = tan^{-1}\left(\frac{h_i + t}{d_i}\right) \tag{9}$$

where $\delta_i$ is a deviance distance $i$, $\theta_i$ is the deviance angle $i$, $h_i$ is obstacle dimension $i$, $t$ is the safe distance and $d_o$ is the distance between the obstacle and the quadcopter.

## Energy

Alongside considering deviance distance, the quadcopter must also consider energy [14]. In this case, there are two energy types [15]. Kinetic energy happens where the quadcopter moves without any altitude changes, whereas potential energy happens where the quadcopter changed altitude. Calculating kinetic energy may be done using (10).

$$\Delta E_k = \frac{1}{2}mV^2 \tag{10}$$

where $m$ is the quadcopter's mass $(kg)$ and $V$ is its velocity $(m/s)$. Assume that there is no energy loss from the quadcopter changing altitudes. Calculating potential energy may be done using (11).

$$\Delta E_p = m.g.\Delta h \tag{11}$$

where $m$ is the mass of the quadcopter, $g$ is gravity and $\Delta h$ is the height difference between the quadcopter's and the avoidance point's coordinates.

$\Delta E_{R,i}$ is the sum of energy needed by the quadcopter to move between its initial coordinates $(X_R, Y_R, Z_R)$ to the avoidance point's coordinates $(X_K, Y_K, Z_K)$. The needed energy sum $\Delta E_{R,i}$ is defined as:

$$\Delta E_{i,j} = \Delta E_p + \Delta E_k \tag{12}$$

where $\Delta E_p$ is the potential energy and $\Delta E_k$ is the kinetic energy of the quadcopter.

Evading in different directions require different amounts of energy needed. Evading in lateral directions (left or right) results in potential energy $\Delta E_p$ being zero due to no changes in altitude. However, when the quadcopter evades in vertical directions (up or down), the altitude changes mean the presence of potential energy $\Delta E_p$.

## Quadcopter Control

This research used a proportional-derivative (PD) controller to control the quadcopter's movements. The controller's inner loop is devoted to attitude control, while its outer loop is devoted to position control [16]. The quadcopter control system is illustrated in Fig. 5.

From the tuning experiments, the proportional-derivative control parameters used in this research are as follows:

$$k_{p,X} = 5, k_{p,Y} = 5\,, k_{p,Z} = 20$$
$$k_{d,X} = 5, k_{d,Y} = 5\,, k_{d,Z} = 10$$
$$k_{p,\phi} = 3000\,, k_{p,\theta} = 3000, k_{p,\psi} = 3000$$
$$k_{d,\phi} = 300\,, k_{d,\theta} = 300, k_{d,\psi} = 300$$

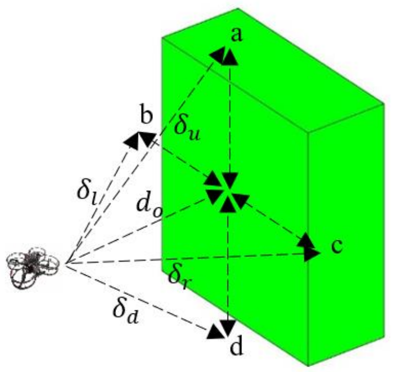
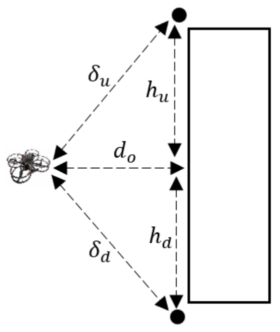
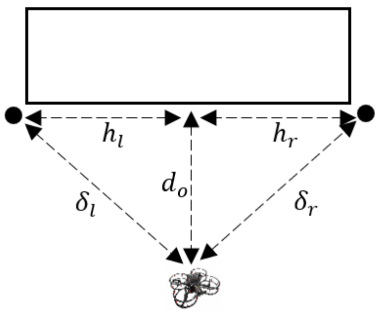


Fig 3. Deviance distance illustration

Explanation:
a : Deviance Up
b : Deviance Left
c : Deviance Right
d : Deviance Down



(a)



(b)

Fig 4. Deviance distance illustration (a) Side view (b) Top view

Table 2. Cluster training data

| **No.** | $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | **Cluster** |
|---|---|---|---|---|---|
| 1 | 0.5 | 0.75 | 2.25 | 2.5 | 1 |
| 2 | 0.5 | 1.5 | 1.5 | 2.5 | 2 |
| 3 | 0.5 | 2.25 | 0.75 | 2.5 | 3 |
| 4 | 1.5 | 0.75 | 2.25 | 1.5 | 4 |
| 5 | 1.5 | 1.5 | 1.5 | 1.5 | 5 |
| 6 | 1.5 | 2.25 | 0.75 | 1.5 | 6 |
| 7 | 2.5 | 0.75 | 2.25 | 0.5 | 7 |
| 8 | 2.5 | 1.5 | 1.5 | 0.5 | 8 |
| 9 | 2.5 | 2.25 | 0.75 | 0.5 | 9 |

Table 3. Obstacle training data

| No. | $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Class |
|---|---|---|---|---|---|
| 1 | 0.01 | 0.2 | 2.8 | 2.99 | Up |
| 2 | 0.04 | 0.4 | 2.6 | 2.96 | Down |
| 3 | 0.21 | 0.2 | 2.8 | 2.79 | Left |
| 4 | 0.24 | 0.4 | 2.6 | 2.76 | Left |
| 5 | 0.4 | 2 | 1 | 2.6 | Right |
| 6 | 0.5 | 1.8 | 1.2 | 2.5 | Right |
| 7 | 2.99 | 0.2 | 2.8 | 0.01 | Down |
| 8 | 2.96 | 0.4 | 2.6 | 0.04 | Down |
| : | : | : | : | : | : |
| 316 | 2.79 | 2.8 | 0.2 | 0.21 | Right |

Table 4. Testing data

| No. | $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Class | Status |
|---|---|---|---|---|---|---|
| 1 | 0.3001 | 0.6 | 0.6 | 0.2999 | Right | Correct |
| 2 | 0.4001 | 0.71 | 0.49 | 0.1999 | Right | Correct |
| 3 | 0.2401 | 0.4 | 2.6 | 2.76 | Left | Correct |
| 4 | 0.1801 | 0.59 | 0.61 | 0.4199 | Left | Correct |
| 5 | 0.0701 | 0.31 | 0.89 | 0.5299 | Left | Correct |
| 6 | 0.0101 | 0.51 | 0.69 | 0.5899 | Up | Correct |
| 7 | 0.0401 | 0.8 | 0.4 | 0.5599 | Up | Correct |
| 8 | 0.5901 | 0.86 | 0.34 | 0.0099 | Down | Correct |
| 9 | 0.5881 | 0.49 | 0.71 | 0.0119 | Down | Correct |
| 10 | 0.4301 | 0.87 | 0.33 | 0.1699 | Right | Correct |
| : | : | : | : | : | : | : |
| 40 | 0.0801 | 0.87 | 0.33 | 0.5199 | Up | Incorrect |

## RESULT AND DISCUSSION

This chapter discusses the results of the modified LMKNCN algorithm in doing certain tests. In the evasion direction avoidance tests, the features used in training data [17] is first discussed, as well as accuracy testing. Then, the algorithm is tested in quadcopter flight plans that must reach its target point with static and dynamic obstacles in the way.

The LMKNCN classification features used in this research is the dimensions of the obstacle against the quadcopter's position. The dimensions feature data is processed into deviance distance data. This feature data consisted of 4 parameters, that of upper span $h_u$, left span $h_l$, right span $h_r$, and lower span $h_d$. The deviance distance data $\delta$ consisted of 4 parameters, that of left, right, up and down deviances. Table 2 shows the feature data used in cluster training data, Table 3 shows the obstacle training data, and Table 4 shows the testing data, all of which resolves as correct.

The simulation tests used a computer with Intel Core i3 CPU of 1.70 GHz and 4 Gb RAM. The tests result in an accuracy of $97.5\%$ (Table 4). The learning process between training and testing data required a computation time of $0.142341$ seconds.

### Case 1

In Case 1, the start point is in coordinate $(0.5,4,2)$ and the target point is in coordinate $(7.5,4,2)$. This case has 1 static obstacle in coordinate $(4,4.1,2)$, shown in top view in Fig 6 and side view in Fig 7.

### Case 2

In Case 2, there is 1 dynamic obstacle moving up and down the positive z-axis. This obstacle has an innate velocity of $0.005 m/s$. The quadcopter is positioned at the start point $(0.5,4,2)$ and has the target point $(7.5,4,2)$. The dynamic obstacle has an initial coordinate $(4, 4, 1.6)$ shown in Fig 8 and Fig 9.

### Case 3

In Case 3, the quadcopter is placed in the start point $(0.5,4,2)$ and its target point is $(7.5,4,2)$. There are 2 dynamic obstacles with starting coordinates of $(3.4, 2.2, 2.26)$ and $(5.2,6.41,1.88)$. The first obstacle has an innate velocity of $0.02\ m/s$, moving the the left (positive y-axis). The second obstacle has an innate velocity of $0.01\ m/s$, moving to the right (negative y-axis) as shown in Fig 10 and Fig 11.

### Result Case 1

The obstacle is detected when the quadcopter is in coordinate $(2.8, 4, 2)$, in $4s$. The detected obstacle's dimensions from the quadcopter are $\mathrm{h_u} = 0.4001$, $h_l = 0.7$, $h_r = 0.5$ and $h_d = 0.1999$.

The closest training cluster data (Table 5) in this case is in Cluster 8. The first closest data in the cluster is the 12th data point. The centroids are located in the 15th and the 16th, shown in Table 6. The most efficient evasion direction is to the right due to the 3 nearest neighbors to the static obstacle training data in Cluster 8 (Table 7) showed the right class.

## Result Case 2

The quadcopter detected the obstacle in $4s$ in coordinate $(4, 4.1, 2)$. The detected dimensions are $h_u = 0.3101$, $h_l = 0.6$, $h_r = 0.6$ and $h_d = 0.0399$.

Table 8 showed that the closest cluster to the obstacle training data is Cluster 8. The first closest training data is the 11th data. The centroids are located in the 12th and 6th data (Table 9). The 3 nearest neighbors showed the *down* class (Table 10), therefore the evasion direction is to the bottom.

## Result Case 3

The quadcopter detected the first obstacle in $4s$ in coordinate $(2.3, 4, 2)$. In $13s$, the quadcopter detected the second obstacle in coordinate $(4, 4, 2)$. The first obstacle's dimensions are $h_u = 0.5601$, $h_l = 0.2$, $h_r = 0$ and $h_d = 0.0399$. The second obstacle's dimensions are $h_u = 0.1808$, $h_l = 0$, $h_r = 0.75$ and $h_d = 0.4192$.

The closest cluster for the first obstacle is Cluster 9, as shown in Table 11. The closest cluster for the second obstacle is Cluster 1, as shown in Table 14. The closest training data in each cluster is the 8th for the first obstacle and the 5th for the second obstacle. The centroids for the first obstacle are the 7th and the 4th (Table 12). The centroids for the second obstacle are the 6th and the 7th (Table 15). For the first obstacle, the most efficient evasion direction is to the right, as shown in Table 13. For the second obstacle, the chosen evasion direction is to the left as shown in Table 16.

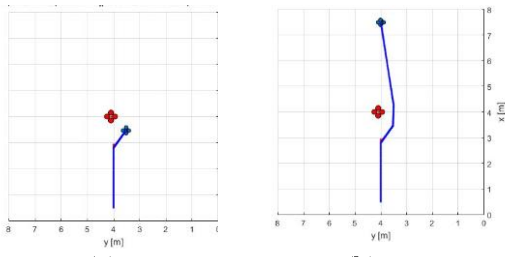

(a) (b)

Fig 5. Case 1 top view (a) 9s (b) 17s

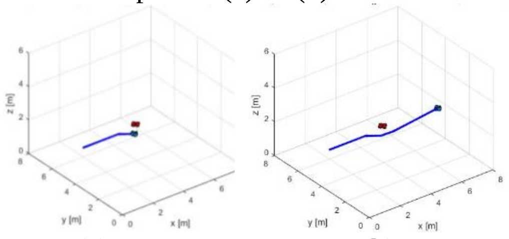

(a) (b)

Fig 6. Case 1 side view (a) 9s (b) 17s

Table 5. Nearest cluster training data for case 1

| $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Cluster | Euclid Distance |
|---|---|---|---|---|---|
| 0.5 | 0.6 | 0.6 | 0.1 | 8 | 0.1999 |
| 0.3 | 0.6 | 0.6 | 0.3 | 5 | 0.2001 |
| 0.5 | 0.9 | 0.3 | 0.1 | 9 | 0.316165 |

Table 6. Centroid data

| Data Pairs | $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Euclid Distance |
|---|---|---|---|---|---|
| 12&15 | 0.45 | 0.68 | 0.52 | 0.15 | 0.076 |
| 12&16 | 0.46 | 0.72 | 0.48 | 0.14 | 0.0893 |
| 12&13 | 0.42 | 0.64 | 0.56 | 0.18 | 0.0894 |
| 12&14 | 0.43 | 0.64 | 0.56 | 0.16 | 0.0982 |

Table 7. Nearest static obstacle training data in cluster 8 for case 1

| $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Class |
|---|---|---|---|---|
| 0.4 | 0.64 | 0.56 | 0.2 | Right |
| 0.5 | 0.72 | 0.48 | 0.1 | Right |
| 0.52 | 0.8 | 0.4 | 0.08 | Right |
| 0.44 | 0.64 | 0.56 | 0.16 | Right |
| 0.47 | 0.64 | 0.56 | 0.13 | Right |

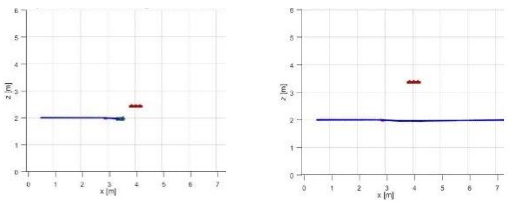

(a) (b)

Fig 7. Case 2 top view (a) 9s (b) 17s

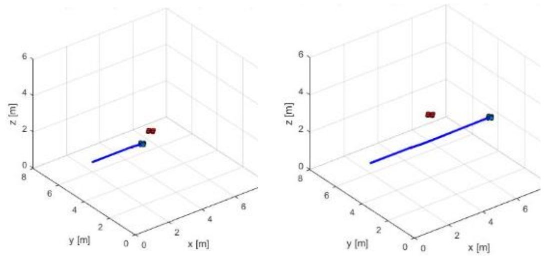

(a) (b)

Fig 8. Case 2 side view (a) 9s (b) 17s

Table 8. Nearest cluster training data for Case 2

| $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Cluster | Euclid Distance |
|---|---|---|---|---|---|
| 0.29 | 0.6 | 0.6 | 0.06 | 8 | 0.0261 |
| 0.17 | 0.6 | 0.6 | 0.17 | 5 | 0.191 |
| 0.06 | 0.6 | 0.6 | 0.29 | 2 | 0.356 |

Table 9. Centroid data

| **Data Pairs** | $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | **Euclid Distance** |
|---|---|---|---|---|---|
| 11 & 12 | 0.3 | 0.6 | 0.6 | 0.05 | 0.0178 |
| 11 & 6 | 0.29 | 0.6 | 0.6 | 0.06 | 0.0343 |
| 11 & 5 | 0.28 | 0.6 | 0.6 | 0.07 | 0.0467 |
| 11 & 13 | 0.31 | 0.64 | 0.56 | 0.04 | 0.0568 |

Table 10. Nearest static obstacle training data in cluster 8 for case 2

| $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | **Class** |
|---|---|---|---|---|
| 0.3 | 0.56 | 0.64 | 0.05 | Down |
| 0.3 | 0.64 | 0.56 | 0.05 | Down |
| 0.27 | 0.64 | 0.56 | 0.08 | Down |
| 0.26 | 0.64 | 0.56 | 0.09 | Right |
| 0.31 | 0.72 | 0.48 | 0.03 | Down |

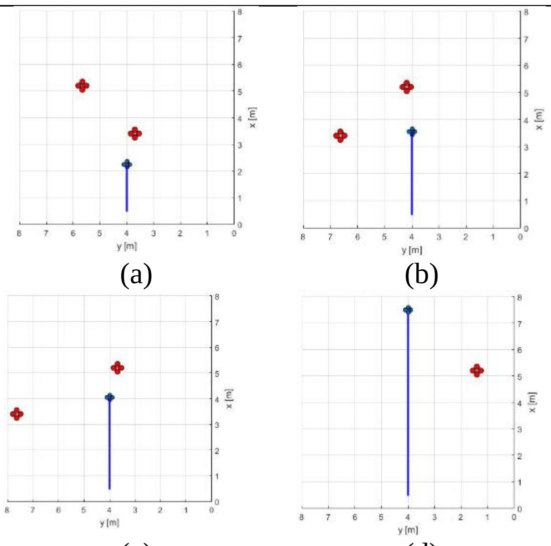


Fig 9. Case 3 top view (a) 9s (b) 13s (c) 23s (d) 31s

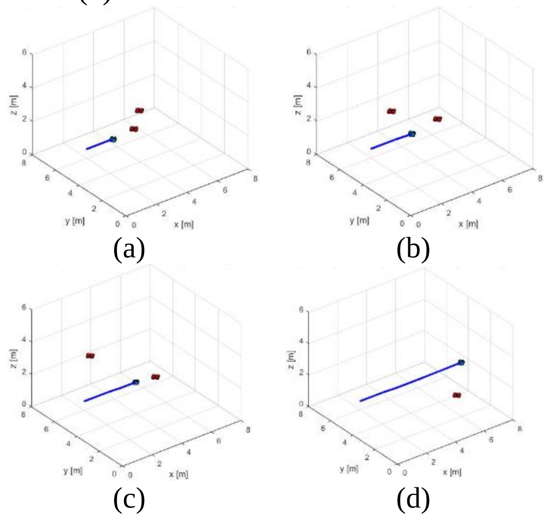


Fig 10. Case 3 Side View (a) 9s (b) 13s (c) 23s (d) 31s

Table 11. Nearest cluster training data for case 3 first obstacle

| $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Cluster | Euclid Distance |
|---|---|---|---|---|---|
| 0.5 | 0.15 | 0.05 | 0.1 | 9 | 0.1106 |
| 0.5 | 0.1 | 0.1 | 0.1 | 8 | 0.165 |
| 0.5 | 0.05 | 0.15 | 0.1 | 7 | 0.2285 |

Table 12. Centroid data

| **Data Pairs** | $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | **Euclid Distance** |
|---|---|---|---|---|---|
| 8 & 7 | 0.55 | 0.18 | 0.02 | 0.045 | 0.0292 |
| 8 & 4 | 0.58 | 0.187 | 0.01 | 0.022 | 0.0316 |
| 8 & 3 | 0.57 | 0.18 | 0.02 | 0.025 | 0.0353 |
| 8 & 2 | 0.57 | 0.17 | 0.03 | 0.031 | 0.0398 |

Table 13. Nearest right-moving dynamic obstacle training data in cluster 9 for case 3 first obstacle

| $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Class |
|---|---|---|---|---|
| 0.56 | 0.19 | 0.01 | 0.04 | Right |
| 0.55 | 0.17 | 0.03 | 0.05 | Right |
| 0.6 | 0.19 | 0.01 | 0.002 | Down |
| 0.59 | 0.17 | 0.03 | 0.01 | Down |
| 0.58 | 0.16 | 0.04 | 0.02 | Down |

Table 14. Nearest cluster training data for Case 3 second obstacle

| $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Cluster | Euclid Distance |
|---|---|---|---|---|---|
| 0.1 | 0.19 | 0.56 | 0.5 | 1 | 0.2887 |
| 0.3 | 0.19 | 0.56 | 0.3 | 4 | 0.3142 |
| 0.5 | 0.19 | 0.56 | 0.1 | 7 | 0.5235 |

Table 15. Centroid data

| Data Pairs | $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Euclid Distance |
|---|---|---|---|---|---|
| 5 & 6 | 0.04 | 0.07 | 0.67 | 0.55 | 0.2194 |
| 5 & 7 | 0.05 | 0.1 | 0.65 | 0.55 | 0.2317 |
| 5 & 1 | 0.02 | 0.05 | 0.7 | 0.58 | 0.2354 |
| 5 & 2 | 0.02 | 0.07 | 0.67 | 0.57 | 0.2445 |

Table 16. Nearest static obstacle training data in Cluster 1 for Case 3 second obstacle

| $h_u$ $(m)$ | $h_l$ $(m)$ | $h_r$ $(m)$ | $h_d$ $(m)$ | Class |
|---|---|---|---|---|
| 0.042 | 0.05 | 0.7 | 0.558 | Left |
| 0.048 | 0.1 | 0.65 | 0.552 | Left |
| 0.06 | 0.15 | 0.6 | 0.54 | Left |
| 0.002 | 0.05 | 0.7 | 0.598 | Up |
| 0.008 | 0.1 | 0.65 | 0.592 | Up |

## CONCLUSION

This research has been This research discussed obstacle avoidance for quadcopter in a 3D environment. The system is designed to resolve efficient avoidance direction by minimizing energy and distance needed to evade static and dynamic obstacles. A modified Local Mean K-Nearest Centroid Neighbor (LKMNCN) algorithm is used in the system by splitting training data into multiple clusters. The quadcopter in this research used a proportional-derivative controller to reach the desired waypoints.

The training data in this research is divided into cluster training data and obstacle training data. The obstacle training data is divided into 5 parts corresponding to the obstacle's characteristics. These include static obstacle data, up-moving, down-moving, right-moving, and left moving data. The nearest neighbor number $k$ is set at 3.

The simulation results show that the designed system resolves an avoidance decision accuracy of 97.5%. Learning time between training and testing data required a measured computation time of 0.142341 seconds. With this system, the quadcopter was able to avoid static and dynamic obstacles of varying velocities.

## REFERENCES

[1] B. H. Y. Alsalam, K. Morton, D. Campbell, and F. Gonzalez, "Autonomous UAV with Vision Based On-Board Decision Making for Remote Sensing and Precision Agriculture," in *2017 IEEE Aerospace Conference*, Big Sky, MT, USA, Mar. 2017, pp. 1–12. doi: 10.1109/AERO.2017.7943593.

[2] D. Popescu, L. Ichim, and T. Caramihale, "Flood Areas Detection based on UAV Surveillance System," in *2015 19th International Conference on System Theory, Control and Computing (ICSTCC)*, Cheile Gradistei, Romania, Oct. 2015, pp. 753–758. doi: 10.1109/ICSTCC.2015.7321384.

[3] W. Wang, Y. Chen, and X. Zhang, "A Rural Constuction Land Extraction Algorithm for UAV Images based on Improved Gaussian Mixture Model and Markov Random Field," p. 4.

[4] M. B. Bejiga, A. Zeggada, and F. Melgani, "Convolutional Neural Networks for Near Real-Time Object Detection from UAV Imagery in Avalanche Search and Rescue Operations," in *2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, Beijing, China, Jul. 2016, pp. 693–696. doi: 10.1109/IGARSS.2016.7729174.

[5] E. Camci and E. Kayacan, "Waitress Quadcopter Explores how to Serve Drinks by Reinforcement Learning," in *2016 IEEE Region 10 Conference (TENCON)*, Singapore, Nov. 2016, pp. 28–32. doi: 10.1109/TENCON.2016.7847952.

[6] I. S. Asti, T. Agustinah, and A. Santoso, "Obstacle Avoidance with Energy Efficiency and Distance Deviation Using KNN Algorithm for Quadcopter," in *2020 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, Surabaya, Indonesia, Jul. 2020, pp. 285–291. doi: 10.1109/ISITIA49792.2020.9163788.\

[7] Y. Liu, D. Chen, and S. Zhang, "Obstacle Avoidance Method based on the Movement Trend of Dynamic Obstacles," in *2018 3rd International Conference on Control and Robotics Engineering (ICCRE)*, Nagoya, Apr. 2018, pp. 45–50. doi: 10.1109/ICCRE.2018.8376431.

[8] J. Gou, Z. Yi, L. Du, and T. Xiong, "A Local Mean-Based K-Nearest Centroid Neighbor Classifier," *Comput. J.*, vol. 55, no. 9, pp. 1058–1071, Sep. 2012, doi: 10.1093/comjnl/bxr131.

[9] S. Damavandinejadmonfared, "Kernel Entropy Component Analysis using Local Mean-based K-Nearest Centroid Neighbour (LMKNCN) as a Classifier for Face Recognition in Video Surveillance Camera Systems," in *2012 IEEE 8th International Conference on Intelligent Computer Communication and Processing*, Cluj-Napoca, Cluj, Romania,

Aug. 2012, pp. 253–256. doi: 10.1109/ICCP.2012.6356195.

[10] D. Pan, Z. Zhao, L. Zhang, and C. Tang, “Recursive Clustering K-Nearest Neighbors Algorithm and the Application in the Classification of Power Quality Disturbances,” in *2017 IEEE Conference on Energy Internet and Energy System Integration (EI2)*, Beijing, Nov. 2017, pp. 1–5. doi: 10.1109/EI2.2017.8245652.

[11] D. Zhang, B. Xu, and J. Wood, “Predict Failures in Production Lines: A Two-Stage Approach with Clustering and Supervised Learning,” in *2016 IEEE International Conference on Big Data (Big Data)*, Washington DC,USA, Dec. 2016, pp. 2070–2074. doi: 10.1109/BigData.2016.7840832.

[12] T. Agustinah, F. Isdaryani, and M. Nuh, “Tracking Control of Quadrotor Using Static Output Feedback with Modified Command-Generator Tracker,” *Int. Rev. Autom. Control IREACO*, vol. 9, no. 4, p. 242, Jul. 2016, doi: 10.15866/ireaco.v9i4.9431.

[13] K. Y. Chee and Z. W. Zhong, “Control, Navigation and Collision Avoidance for an Unmanned Aerial Vehicle,” *Sens. Actuators Phys.*, vol. 190, pp. 66–76, Feb. 2013, doi: 10.1016/j.sna.2012.11.017.

[14] H. V. Abeywickrama, B. A. Jayawickrama, Y. He, and E. Dutkiewicz, “Algorithm for Energy Efficient Inter-UAV Collision Avoidance,” in *2017 17th International Symposium on Communications and Information Technologies (ISCIT)*, Cairns, Australia, Sep. 2017, pp. 1–5. doi: 10.1109/ISCIT.2017.8261200.

[15] J. Viegas, *Kinetic and Potential Energy: Understanding Changes within Physical Systems*. New York: Rosen Pub. Group, 2005.

[16] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, “The GRASP Multiple Micro-UAV Testbed,” *IEEE Robot. Autom. Mag.*, vol. 17, no. 3, pp. 56–65, Sep. 2010, doi: 10.1109/MRA.2010.937855.

[17] M. Stamp, *Introduction to Machine Learning with Applications in Information Security*. Boca Raton London New York: CRC Press, Taylor & Francis Group, 2018.

.