A Vision-based Method for Autonomous Landing on a Target with a Quadcopter

Le-Anh Tran, Ngoc-Phu Le, Truong-Dong Do and My-Ha Le

***Abstract* – During autonomous flight procedure, autonomous landing on a specified object is one of the most dangerous and challenging processes, requiring an advanced study in both hardware and software approaches. In this paper, a new methodology is developed, including two major tasks, for vision-based autonomous landing systems executed by a quadcopter: (1) a vision-based algorithm is used to detect and predict an object's future motion using Kalman Filter; (2) PID algorithm is implemented in a quadcopter platform to autonomously balance and land this one on a stationary target. This autonomous task is obtained by two essential components: accelerometer and gyroscope. However, the sensors are susceptible to noise caused by driving forces in the system, such as the vibration of the motors. Therefore, we also investigate a use of complementary filter to make the outcome from two sensors as best as possible. Real quadcopter experiments have been implemented to validate the effectiveness of the proposed method.**

Keywords: autonomous landing, quadcopter, PID controller, object detection, Kalman filter;

I. INTRODUCTION

An Unmanned Arial Vehicle (UAV), which is commonly known as a Drone, is an aircraft without a human pilot aboard. There are many various degrees of autonomous flights: either under manual control by human or autonomous control by onboard computers. Due to the flexibility in control, UAV systems have a wide applications in all civilization and military fields, such as search and rescue operation, tracking object, delivery product. More recent researches relative to UAV application can be found in [5-6]. A Quadcopter is a type of UAV with 4 rotors, it has advantages compared to other types of UAV such as simply mechanic architecture, flexible moving ability in narrow spaces with the compact size, and easily manufactured [1-6].

Computer vision consists of theories and engineering aimed to create artificial systems in order to collect and process images or multi-dimensional database using cameras. The combination between computer vision and other engineering technologies provide many applications in science, military and many more.

Nowadays, experts are outlining positive methodologies to solve traffic problems; notably, tracking a crime or supporting victims of a traffic accident. A potential method proposed in this paper is to use the UAV as a supporting robot since its traveling path is less blocked from the air than that of the unmanned ground vehicles (UGVs). On the other hand, a difficult problem of UAVs is safety landing. Currently, researches on the autonomous landing for UAVs is still underway in the world. The purpose is to allow UAVs to be capable of landing on ships and trucks, or in the case of cyclones and other disasters [4]. This paper presents a solution for the autonomous landing of a quadcopter on a stationary target. A vision-based method for autonomous landing on a target with a quadcopter will be considered. This method includes a searching and a landing part.

The process is divided into 3 parts: 1 – detecting position, 2 – predicting position and 3 – autonomous landing. In the first part, images are collected by a camera and then processed by a Raspberry pi 3 Model B using HSV color space, thresholding method to detect the X-Y coordinates of the target on the 2D image plane. Then, a Kalman Filter is applied to predict the near future X'-Y' coordinates of the target in the second part. In the final part, the X'-Y' coordinates value are transformed to electronic signal in order to control 4 rotors for landing. In addition, the quadcopter stabilization is always guaranteed by using PID controllers. Besides the major components in the research, some different problems also need to be solved are reducing noises for better quadcopter stabilization, and designing the most suitable architecture for the quadcopter.

This paper is organized as follows. Section II firstly describes the quadcopter platform, then the PID controllers and a complementary filter for the quadcopter stabilization are presented as well. Next, a vision-based algorithm for object detection using HSV color space and an object position prediction method using a Kalman filter are discussed in section III. Section IV concerns the control algorithm for autonomously landing. Then, section V reports the experimental results. Finally, the conclusions of the paper are presented in section VI.

II. QUADCOPTER DYNAMICS AND CONTROL

A. Experimental Apparatus

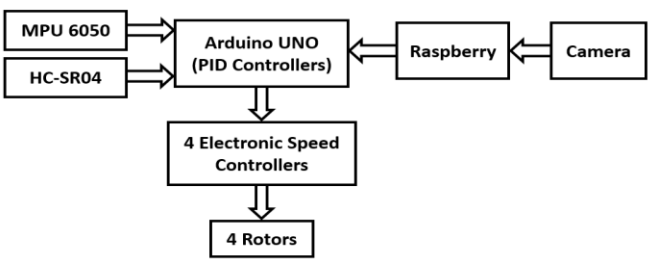

Figure 1. The block diagram of the system.

*The research was supported by Intelligent Systems Laboratory (ISLab), Faculty of Electrical and Electronics Engineering, Ho Chi Minh City University of Technology and Education, Vietnam.

Le-Anh Tran, Ngoc-Phu Le and Truong-Dong Do are students in Ho Chi Minh City University of Technology and Education, Vietnam.
(e-mail: { tranleanh.nt, lephu0803, truongdong2406 }@gmail.com)

My-Ha Le is a Ph.D. in Faculty of Electrical and Electronics Engineering, Ho Chi Minh City University of Technology and Education, Vietnam. (corresponding author)
(e-mail: halm@hcmute.edu.vn)

Figure 2. The quadcopter platform.

Figure 1 depicts the block diagram of the system while figure 2 shows the hardware platform used in this research.

The platform used in this research is a DIY hardware platform. S500 Quadcopter Frame Kit is increasingly used in education and research because of its low cost, robustness and easily assembled. The system consists of a microcontroller Arduino Uno, 4 Electric Speed Controllers (ESCs), 4 Brushless DC motors, an Ultrasonic Sensor HC-HR04, a MPU 6050 sensor equipped with 6-degree-of-freedom inertial measurement unit: 3-axis gyroscope and 3-axis accelerometer, a Raspberry pi 3 model B, a 5.0 MP camera for the maximum resolution is 720p, and the resolution of 640x480 with 30fps at the bottom to look down vertically, and a 3800mAh battery for continuous flights of from 10 to 15 minutes. The quadcopter can achieve speed of about 3 m/s and operate very well for both indoor and outdoor.

At the begin, the quadcopter takes off under a remote controller. The remote controller consists of two components: a Transmitter Devo 7 and a Receiver RX-701 with 7 channels and the frequency of 2.4GHz. The flights operate under the controller by human until the camera is able to observe the target in its vision. Thus, the quadcopter can detect the target (shown in figure 3), predict its next position in 2D images and make the autonomous landing on it.

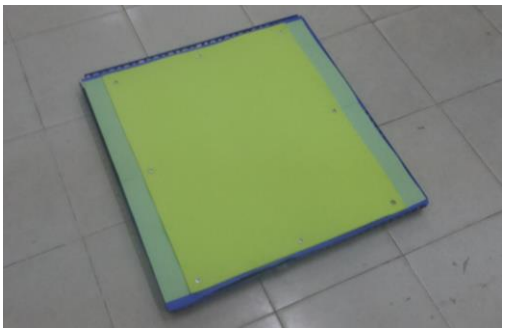

Figure 3. The target used in the research (60x60cm).

B. PID Controller

A controller plays the most important role in automatic control systems. The basic idea of a controller is to read a sensor, then calculate the desired actuator output. There are different controllers which are suitable for various systems such as PID, LQR, etc. In the research, the PID controller is chosen due to its versatility and facile implementation, while also providing a consistent response to the model dynamics [6-7].

PID (Proportional-Integral-Derivative) controller is the commonest control algorithm used in many applications to optimize the system automatically. A PID controller is capable of controlling the system to meet the quality criteria such as short transient time, fast response, and reducing the overshoot for the system [5-7]. Figure 4 shows the block diagram of a PID controller in a feedback loop.

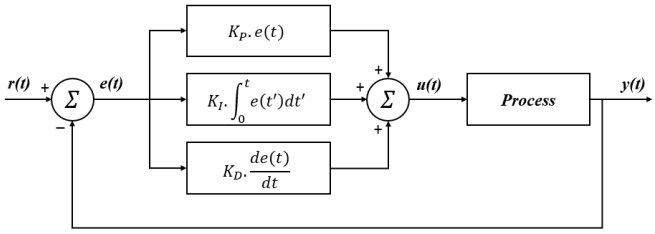

Figure 4. PID controller.

In which, *r(t)* is the desired process value or set-point (SP), *y(t)* is the measured process value (PV), *e(t)* is the error value which is the difference between the SP and the PV. The PID controller attempts to minimize the error over time by adjustment of a control variable *u(t)*.

The control function can be expressed mathematically as follows:

$$u(t) = K_P.e(t) + K_{I.0}\int^t e(t')d(t') + K_D.de(t)/dt \qquad (1)$$

Next part presents PID controllers for the quadcopter balancing. The feedback are yaw, pitch and roll angle values calculated from the accelerometer sensor MPU 6050.

C. Quadcopter Dynamics

The yaw, pitch and roll angles of the quadcopter are initialized as figure 5:

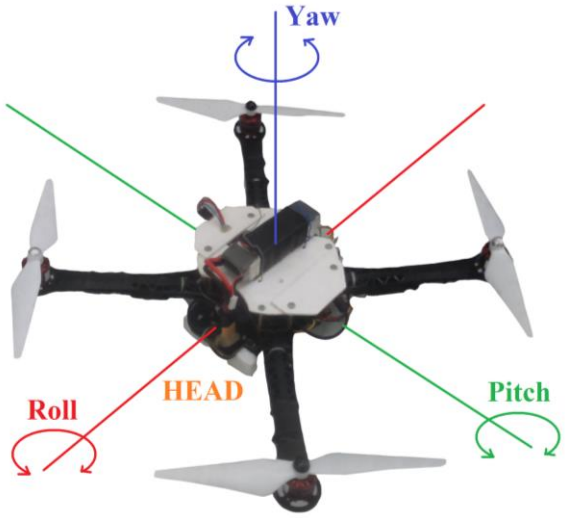

Figure 5. The yaw, pitch and roll angles.

Equation to calculate the angles from the gyroscope [2]:

$$\begin{bmatrix} \frac{\delta\psi}{\delta t} \\ \frac{\delta\theta}{\delta t} \\ \frac{\delta\phi}{\delta t} \end{bmatrix} = \frac{1}{\cos\phi} \times \begin{bmatrix} 0 & \sin\phi & \cos\phi \\ 0 & \cos\phi\cos\theta & -\sin\phi\cos\theta \\ \cos\theta & \sin\phi\sin\theta & \cos\phi\cos\theta \end{bmatrix} \times \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \quad (2)$$

Where Ψ, θ, Φ are the angles at the previous time (t-1). So, the real values of them at the current time are computed as below:

$$\begin{bmatrix} \psi(t) \\ \theta(t) \\ \phi(t) \end{bmatrix} = \begin{bmatrix} \psi(t-1) \\ \theta(t-1) \\ \phi(t-1) \end{bmatrix} + \begin{bmatrix} \frac{\delta\psi}{\delta t} \\ \frac{\delta\theta}{\delta t} \\ \frac{\delta\phi}{\delta t} \end{bmatrix} \times \Delta t \quad (3)$$

Based on x-axis, y-axis and z-axis accelerations measured from the accelerometer, the roll and pitch angles are computed [2]:

$$\begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\cos\psi & -\sin\theta \\ \cos\psi\sin\theta\sin\phi & \cos\phi\cos\psi + \sin\theta\sin\phi\sin\psi & \sin\theta\cos\phi \\ \cos\psi\sin\theta\cos\phi & -\sin\phi\cos\psi + \sin\theta\cos\phi\sin\psi & \cos\theta\cos\phi \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad (4)$$

Thus, the actual values of roll and pitch angles measured by the accelerometer are calculated as below:

$$Roll = \phi = \arcsin\left(\frac{A_y}{\cos\theta}\right) \quad (5)$$

$$Pitch = \theta = \arcsin(A_x) \quad (6)$$

In addition, in this platform, an ultrasonic sensor is used to maintain the altitude of the quadcopter in space. In this case, the echo time signal from the ultrasonic sensor is fed back to a PID controller.

Ultimately, apply the yaw, pitch, roll angles and the echo time signal to the PID controllers. The system finally is structured as figure 6:

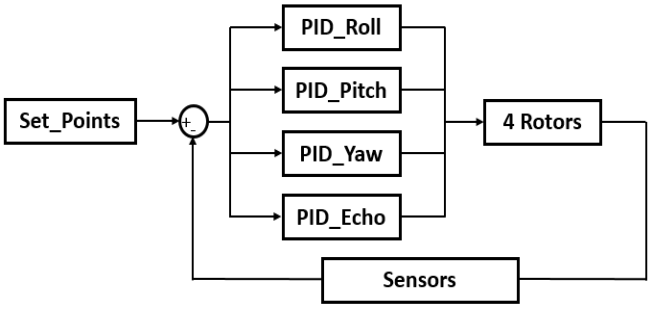

Figure 6. The overall control structure of the quadcopter.

D. Complementary Filter

As the way an accelerometer measures active forces on an object, it does not measure only gravitation but also other forces, every small force which is active on the object is completely measured and is considered to be a disturbance to the measurement. An accelerometer itself is a sensor susceptible to interferences. In quadcopters, the driving forces of the system, such as the vibration of the motors, will also affect the accelerometer. Therefore, using a filter is definitely necessary.

Besides, because of the integration over time, the measurement of the gyroscope tends to drift and do not return to zero when the system returns to its original position. The gyro data is only reliable in a short term and it starts to drift in a long term.

A complementary filter indicates an efficient solution. In the short term, the data from the gyroscope (*gyrData*) is used because of its high accuracy. In the long term, the data from the accelerometer (*accData*) is used since it does not drift. In the simplest form, the filter formula looks as below [1][7]:

angle = alpha(angle + gyrData*dt) + (1-alpha)*accData* (7)

In the equation (6), *alpha* is the filter coefficient (0<*alpha*<1) and *angle* is the output of the filter. Roll and pitch angle values are updated every iteration in an infinite loop. The filter will check whenever the values measured from the accelerometer is either reasonable or not. If any value is too large or too small, it is a complete disturbance, and the complementary filter attempts to decrease the influence of this disturbance for the better computation. For instance, if the *alpha* is 0.98, the filter will update the pitch angle and roll angle for calculation by taking 98% of the current values calculated by the gyroscope and an additional 2% of the angle values computed by the accelerometer. It is always guaranteed that the measured values will not drift and also are very precise in the short term. Figure 7 shows the original signal of roll angle measured from the accelerometer over time (blue) and the processed signal with a complementary filter (red). It is facile to realize that the blue one is easily affected by noise and the red one is a better input for the controller.

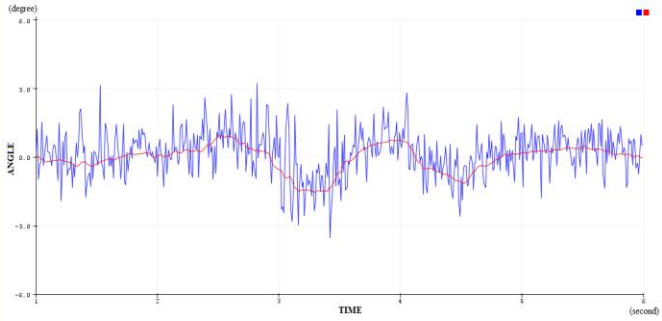

Figure 7. Complementary filter.

III. A VISION-BASED METHOD TO DETECT AND PREDICT AN OBJECT POSITION

This section presents the first main task in the paper, the vision-based method to detect and predict the position of a target and then land on it.

A. Color Detection

Generally, a photograph taken from a camera is formatted as a 3-color RGB color image. However, in the field of image processing, the color space is transferred to HSV because of its certain benefits. HSV is a color space commonly used in image editing, image analysis and computer vision. This color space is based on three parameters to describe colors: Hue (H), Saturation (S), and Value (V). Figure 8 depicts HSV color space:

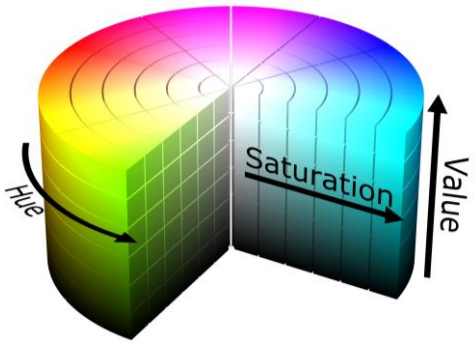

Figure 8. HSV color space.

The circle is the hue representation field (H), starting at the red primary at 0°, passing through the green primary at 120° and the blue primary at 240°, and then back to red at 360°. The brightness value (V) is represented by going from the bottom up to the top of the cylinder. At the bottom of the cylinder, the value V is 0, the smallest, and the top of the cylinder is the brightest (V=1). Moving from the center of the cylinder to the side surface of the cylinder is the saturation of the color (S), S=0 for the center where the color is the lightest and S=1 on the side surface of the cylinder, where the color value is most dense. Thus, each value (H, S, V) fully describes the hue, saturation, and brightness of a specific color [3][8].

The object detection task in the research was executed in Python and OpenCV, a library of programming functions mainly aimed at real-time computer vision. In the OpenCV library, the HSV color space has been slightly modified, the H value is not from 0° to 360° but from 0° to 180°, and the values of S and V are ranged from 0 to 255. Figure 9 presents a window with track bars to determine the H, S and V values for color detection with OpenCV.

Figure 9. Track bars to determine the H, S and V values for color detection with OpenCV.

Once the color is determined using HSV, the thresholding method is used to convert the image into binary format. After that, contours searching function is executed for drawing the bounding box to cover the current location of the target. The current position is defined as the center of the bounding box. Figure 10 describes all of the processes presented as stated above.

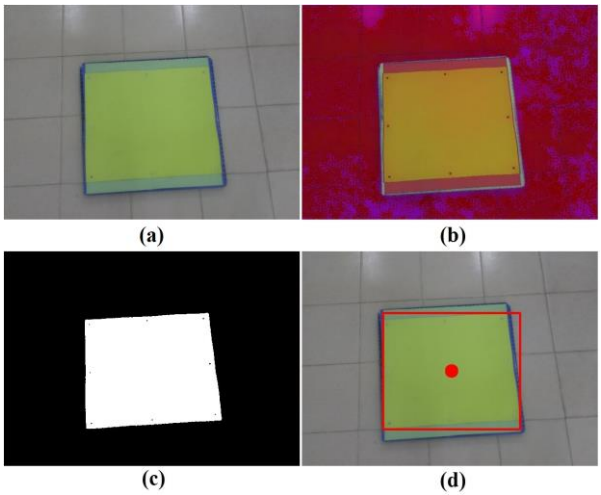

Figure 10. (a) original image, (b) converted HSV image, (c) thresholded image, (d) output image with the detection.

B. Kalman Filter for Predicting Position

A common hypothesis in many vision-based tracking algorithms presented so far is that the motion of an object will change very little between two subsequent frames. Thus an object's position in the actual frame is an approximation of its position in next frames [9].

This paper aims to exploit the concept of object trajectory. The assumption is that the change in the position of a object in each motion frame follows a mathematical model over time. And the motion vector is modeled as a discrete-time linear system described by the following equation [9]:

$$x[t+1] = F\,x[t] + v[t] \qquad (8)$$

In the equation above, $x[t]$ represents the system state and describes the exact value of each coefficient of the motion vector at time t, consists of the first derivative and the second derivative as well, matrix F represents the second order kinetic model, $v[t]$ stands for errors made by the modeling process. This error is defined as a sequence of zero-mean, white, Gaussian process noise [9].

In line with the equation (1), a measurement equation as follow is defined:

$$z[t+1] = H\,x[t+1] + w[t+1] \qquad (9)$$

This equation describes the relationship between state observation $z[t]$ and system state $x[t]$. This relationship is determined by the matrix H and by the error $w[t]$. The error modeled by $w[t]$ calculates all possible inaccuracies in the estimation of motion [9].

Applying a Kalman filter for the system determined by the equation (8) and (9), it is possible to perform a recursive motion tracking. At each new observation of the object motion, $z[t]$ is obtained through the estimation of motion. According to the measured observations and the kinetic models, the Kalman filter updates the state vector $x[t]$. This filter integrates over time the available temporal information for each object [9].

However, Kalman filter does not always provide a correct prediction. This commonly happens when objects are not predictable and that mislead the Kalman filter. In this case, for better results, a rough prediction is used, that is the

last estimated motion of the object, *x[t]*. This will represent a first guess of the future object position and motion [9].

A test is always necessary to decide when it is better to use, as a prediction, the last estimated motion, *x[t]*, instead of using the Kalman motion prediction *x[t+1=t]*. To solve this, a technique was proposed which is based on the motion compensation error estimation. The predicted object is compensated with both *x[t]* and *x[t+1=t]*. The motion compensation between these two that produces the smaller mean square error is chosen as the better prediction [9].

Figure 11 indicates the Kalman filter equations which are divided into two parts, "prediction" and "correction". The "prediction" part estimates the motion or the next state and the "correction" part is responsible to calculate error then update the filter coefficient for the next prediction. Figure 12 shows the predictions of the target position in 2D image, the red circles are the actual position detections and the green circles are the predictions of the next target position.

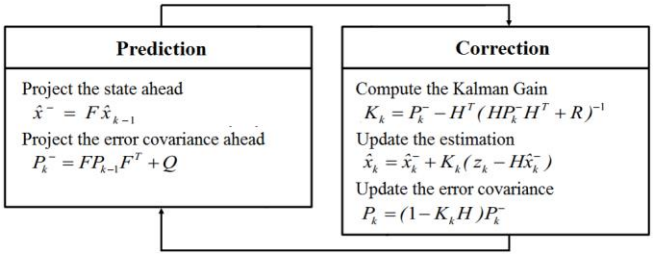

Figure 11. Kalman Filter equations.

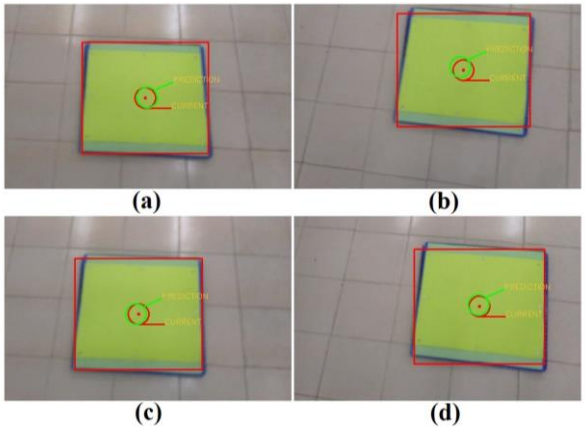

Figure 12 (a-d). Target predictions.

IV. CONTROL ALGORITHM FOR LANDING

As stated above, the predicted position of the target has been determined. This section presents the control algorithm for landing using the predicted coordinates of the target to generate an autonomous landing quadcopter control signal.

The idea behind the algorithm is that the quadcopter will autonomously turn left or right for adjusting its position in space by changing the set-points of roll and pitch angles so that the distance between the center point of the camera frame and the predicted coordinates as small as possible. In this case, the camera frame has a size of 320x240, the distance between the center point of the camera frame and the predicted coordinates is compared to 50 pixels in 2D image with x-y coordinates, that distance is computed as below:

$$Distance = sqrt[\ (160-x)\hat{}2 + (120-y)\hat{}2\ ] \qquad (10)$$

Where (*160, 120*) is the x-y coordinate of the frame center. When the distance is smaller than 50 pixels, the throttle is adjusted to be decreased regularly then the quadcopter can land on the target as accurate as possible. The flow chart for the landing algorithm is presented in figure 13.

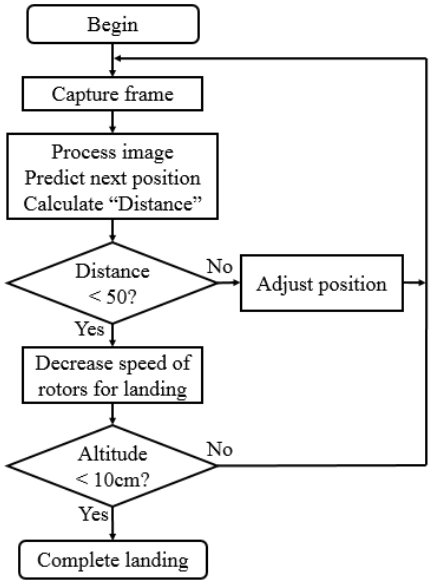

Figure 13. The flow chart of the landing algorithm.

V. EXPERIMENTAL RESULTS

Figure 14 (a-d) and 14 (e-h) depict the experiments which were executed indoor at the altitudes of 100cm and 150cm above the ground.

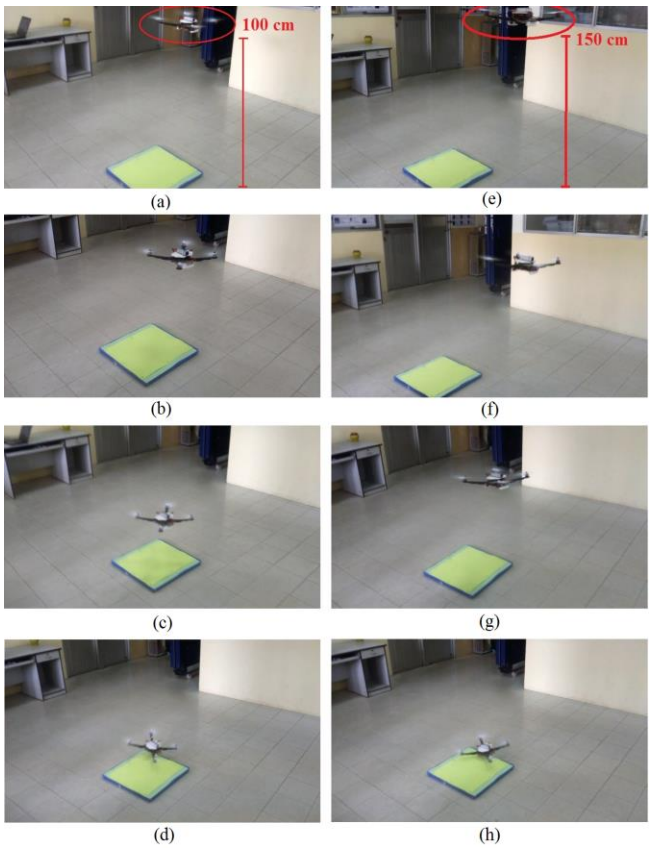

Figure 14 (a-h). The indoor experiments.

The results showed that the quadcopter was able to land on the limited area of the target, yet with low accuracy.

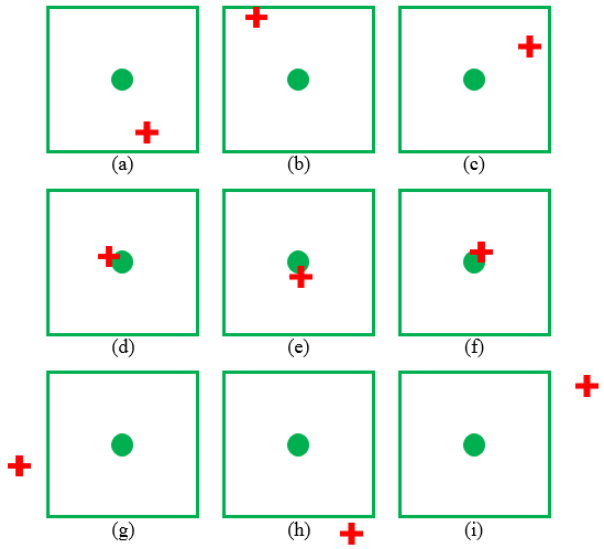

Figure 15 (a-i). Positions of landing.

Figure 15 (a-i) indicate some more experiments of landing. The landing position changes in different experiments. There are some analysis:

Figure 15 (a-c): The quadcopter is able to complete landing with final positions are in the limited area of the target, yet not high accuracy. The reason might be that the rotor speed is reduced too fast when landing, therefore the quadcopter gets out of the balanced state.

Figure 15 (d-f): The quadcopter is able to complete landing with higher accuracy than previous because of the improvement in the reduction of rotor speed.

Figure 15 (g-i): The experiments that the quadcopter cannot complete landing and gets out of the limited area of the target. The reason is that the quadcopter cannot observe and detect the target in its vision.

VI. CONCLUSIONS

After many efforts of research and development, we have designed and assembled a complete quadcopter model. Besides, an autonomous landing method based on vision with a quadcopter platform is presented in this paper. The experiments were executed indoor many times at different altitudes of landing. During the experimental time, there are many problems that we had to face, especially in the landing part. After improving the landing algorithm and fine-tuning the code, the quadcopter now operates with good results. The outcome proved that the quadcopter is capable of autonomously landing on a stationary target very well with 80% accuracy by using a single camera attached to the bottom, it means that there are 8 out of 10 times the quadcopter land successfully on the target.

An improvement for the system to be capable of autonomously landing on a moving target with a higher accuracy will be proposed in the future.

ACKNOWLEDGMENT

The authors would like to thank Tuan-Thong Le for his help, especially in doing experiments. This research is supported by Intelligent Systems Laboratory (ISLab), Ho Chi Minh City University of Technology and Education.

REFERENCES

[1] V.P. Kodgirwar, Vivek Kumar, Manish Shegokar, and Sushant Sawant, "Design of Control System for Quadcopter using Complementary Filter and PID Controller," *IJERT*, vol. 3 Issue 4, April - 2014.

[2] Minh Quan Huynh, Weihua Zhao, and Lihua Xie, "Adaptive Control for Quadcopter: Design and Implementation," in *2014 13th International Conference on Control, Automation, Robotics and Vision (ICARCV 2014)*, Marina Bay Sands, Singapore, Dec. 10-12th, 2014.

[3] Chi-Tinh Dang, Hoang-The Pham, Thanh-Binh Pham, and Nguyen-Vu Truong, "Vision Based Ground Object Tracking Using AR.Drone Quadrotor," in *International Conference on Computer Applications & Information Security (ICCAIS)*, Nha Trang City, Vietnam, Nov. 25-28, 2013, pp. 146-151.

[4] Daewon Lee, Tyler Ryan, and H. Jin. Kim, "Autonomous Landing of a VTOL UAV on a Moving Platform Using Image-based Visual Servoing," in *2012 IEEE International Conference on Robotics and Automation (ICRA 2012)*, RiverCentre, Saint Paul, Minnesota, USA, May. 14-18, 2012, pp. 971-976.

[5] David Howard, and Torsten Mer, "A Platform for the Direct Hardware Evolution of Quadcopter Controllers," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, Sept. 28 – Oct. 2, 2015, pp. 4614-4619.

[6] Lucas M. Argentim, Willian C. Rezende, Paulo E. Santos, and Renato A. Aguiar, "PID, LQR and LQR-PID on a Quadcopter Platform," in *2013 International Conference on Informatics, Electronics and Vision (ICIEV)*, Dhaka, Bangladesh, May 17-18, 2013.

[7] Kartik Madhira, Ammar Gandhi, and Aneesha Gujral, "Self-balancing Robot using Complementary filter," in *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, India, 2016, pp. 2050-2054.

[8] Khamar Basha Shaik, Ganesan P, V.Kalist, B.S.Sathish, and J.Merlin Mary Jenitha, "Comparative Study of Skin Color Detection and Segmentation in HSV and YCbCr Color Space," in *3rd International Conference on Recent Trends in Computing 2015 (ICRTC 2015)*, Ghaziabad, India, Mar. 12-13, 2015, pp. 41-48.

[9] Francesco Ziliani, and Fabrice Moscheni, *Kalman Filtering Motion Prediction for Recursive Spatio-Temporal Segmentation and Object Tracking*. Signal Processing Laboratory, Swiss Federal Institute of Technology, CH-1015 Lausanne, Switzerland.