

REPORT 21/04/2018

FRA501 Principles of Model-based Design in Robotics

อาจารย์รัชชา ชูพจน์เจริญ

# Quadrotor Dynamic Modelling

## การโมเดลระบบของอากาศยานสี่ใบพัด

### Member

นายจุฬภัทร จิรชัย	57340500013
นายเจตน์นัท หอมจันทนากุล	57340500015
นายวุฒิภัทร ไชคอนันตทรัพย์	57340500067

Institute of Field Robotics, KMUTT



ชื่อหัวข้อ	Quadrotor Dynamic Modelling การโมเดลระบบของอากาศยานสี่ใบพัด
รายวิชา	FRA501 Principles of Model-based Design in Robotics
สาขาวิชา	วิศวกรรมหุ่นยนต์และระบบอัตโนมัติ
คณะ	สถาบันวิทยาการหุ่นยนต์ภาคสนาม
สถาบัน	มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี
ปีการศึกษา	2560

---

## บทคัดย่อ

## สารบัญ

เรื่อง	หน้า
บทคัดย่อ .....	ก
สารบัญ .....	ข
บทที่ 1 Introduction .....	1
บทที่ 2 System Description .....	2
2.1 Quadroter mathematical model.....	2
2.1.1 Preliminar notions .....	2
2.1.2 Euler angles and Quaternions .....	4
2.1.3 Qaudroter mathematical model.....	6
2.1.4 Motor model .....	7
2.2 Linear Quadratic Regulator.....	8
2.3 Hector Quadrotor ROS Stack.....	9
2.3.1 hector_quadrotor_description.....	9
2.3.2 hector_quadrotor_gazebo.....	9
2.3.3 hector_quadrotor_teleop .....	10
2.3.4 hector_quadrotor_gazebo_plugin.....	10
บทที่ 3 Simulation .....	11
3.1 สั่งให้ quadrotor บินขึ้นเหนือพื้น .....	12
3.2 สั่งให้ quadrotor บินวนเป็นเกลียว .....	12
3.3 สั่งให้ quadrotor บินเป็นรูปดาว .....	13
บทที่ 4 Analysis/Discussion.....	14
บทที่ 5 Conclusion.....	15
เอกสารอ้างอิง.....	15

## บทที่ 1

### Introduction

อากาศยานสี่ใบพัด (quadrotor) คือเฮลิคอปเตอร์หลายใบพัด (multi-helicopter) ที่มีลักษณะใบพัดเรียงตัวกันเป็นรูปสี่เหลี่ยมมุมฉาก หรือเป็นรูปทรงที่มีความสมมาตร เพื่อใช้ในการพยุงพาหนะและควบคุมทิศทางได้อย่างเป็นอิสระมากขึ้นเมื่อเทียบกับเฮลิคอปเตอร์ทั่วไป โดยในปัจจุบัน อากาศยานสี่ใบพัดได้ถูกนำมาประยุกต์ใช้งานในหลากหลายรูปแบบ ยกตัวอย่างเช่น ด้านเกษตรกรรม(ใช้ในการฉีดยาป้องกันศัตรูพืชหรือตรวจสอบผลผลิตผลเพื่อช่วยเกษตรกร) ด้านอุตสาหกรรม(ใช้สำรวจรอยรั่วที่หลังคาของโรงงาน บ้านเรือน) ด้านการทหาร(ใช้ในการสำรวจพื้นที่ ตรวจสอบที่ตั้งของศัตรู) ด้านกู้ภัยฉุกเฉิน(ใช้ในการหาพื้นที่ประสบภัย ไฟไหม้) ด้านการบันเทิง(ใช้แปรรหัสเป็นรูปภาพและเคลื่อนไหวในรูปแบบต่างๆ) เป็นต้น ซึ่งจะเห็นได้ว่าอากาศยานสี่ใบพัดนั้นสามารถนำไปใช้ได้หลากหลายทาง แต่ด้วยจำนวนของใบพัดที่มีมาก ทำให้ระบบมีกลไกและกระบวนการในการควบคุมที่มีความซับซ้อนมากขึ้น ทางคณะผู้จัดทำจึงสนใจที่จะทำโครงการเกี่ยวกับอากาศยานสี่ใบพัด เพื่อศึกษาพลศาสตร์และ ออกแบบระบบควบคุม รวมไปถึงการจำลองการเคลื่อนที่ของอากาศยาน โดยทำผ่านโปรแกรมจำลอง และนำความรู้ทั้งหมดไปใช้ในการพัฒนาการทำงานของอากาศยานสี่ใบพัดต่อไปในอนาคต

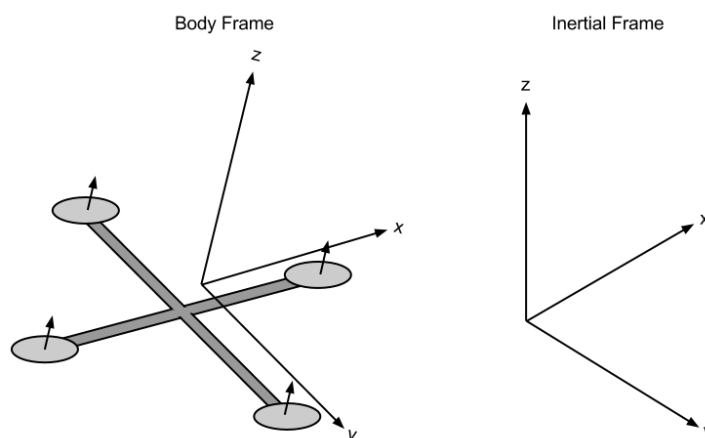
## บทที่ 2

### System Description

#### 2.1 Quadrotor mathematical model

##### 2.1.1 Preliminary notions

โดรนเป็นอากาศยานชนิดหนึ่งที่มีมอเตอร์เป็นตัวขับเคลื่อนอยู่ 4 ตัว มีบอร์ดควบคุมและสั่งการอยู่ตรงกลางระหว่างมอเตอร์ทั้งสี่ ก่อนจะหาสมการคณิตศาสตร์ของโดรนนั้น จะต้องเริ่มจากรู้จักเฟรมอ้างอิงก่อน เฟรมอ้างอิงนี้จะช่วยบอกได้ว่าตัวโดรนนั้นอยู่ที่ตำแหน่งไหน เอียงอย่างไรอยู่ เฟรมอ้างอิงของโดรนนั้นใช้เพียง 2 เฟรมด้วยกัน คือเฟรมที่อยู่กับที่ไม่ขยับไปไหน และเฟรมที่เคลื่อนที่ไปมา เฟรมแรกคือเฟรมที่อยู่กับที่นั่นมีชื่อว่า “Inertial frame” เฟรมนี้เอาไว้สำหรับใช้ในการอ้างอิงพื้นโลก โดยที่ทิศทางของแรงโน้มถ่วง (gravity) จะมีทิศทางไปทางแกน  $-Z$  ของ “Inertial frame” ส่วนเฟรมอีกอันคือ “Body frame” เฟรมนี้เอาไว้ใช้สำหรับในการบอกตำแหน่งและการเอียงของตัวโดรน โดยการบอกนั้นจะบอกเทียบกับ “Inertial frame” การตั้งเฟรมนี้นั้นจะให้แกน  $+Z$  ชี้ไปตามทิศของมอเตอร์ทั้งสี่ ดังรูปที่ 2.1

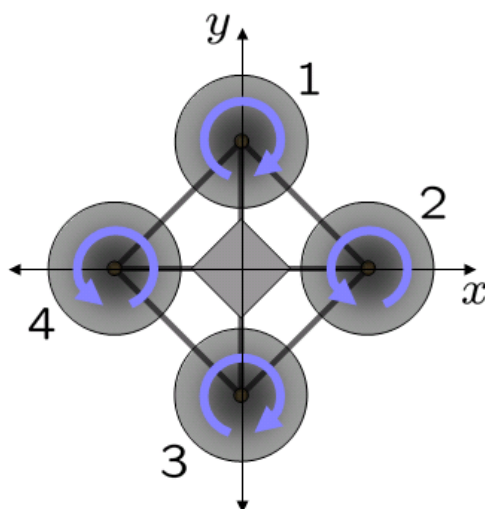


รูปที่ 2.1: เฟรมอ้างอิงของโดรน

การควบคุมตำแหน่งและท่าทางของโดรนนั้น เราสามารถควบคุมได้โดยการสั่งให้มอเตอร์ทั้งสี่ตัวมีความเร็วที่แตกต่างกัน เมื่อมอเตอร์หมุนด้วยความเร็วไม่เท่ากันจะทำให้เกิดแรงและโมเมนต์ขึ้นที่ตัวโดรน แรงยกนั้นจะเกิดขึ้นจากการหมุนมอเตอร์ทั้งหมด ส่วนการเอียง Pitch (หมุนรอบแกน Y ของ Body frame), Roll (หมุนรอบแกน X ของ Body frame) จะเกิดจากความแตกต่างระหว่างแรงยกของมอเตอร์ทั้งสี่ตัว แรงโน้มถ่วงของโลก และ Yaw (หมุนรอบแกน Z ของ Body frame) เกิดจากการที่มอเตอร์หมุนด้วยความเร็วที่ไม่สมดุลกัน โดรนจะไม่หมุน Yaw หากมีมอเตอร์หมุนไปในทิศทางตรงกันข้ามกัน ดังนั้นทำให้เราสามารถแบ่งใบพัดของโดรนออกเป็น 2 กลุ่ม แต่ละกลุ่มจะมีทิศทางการหมุนตรงข้ามกันและอยู่ฝั่งตรงกันข้ามกัน

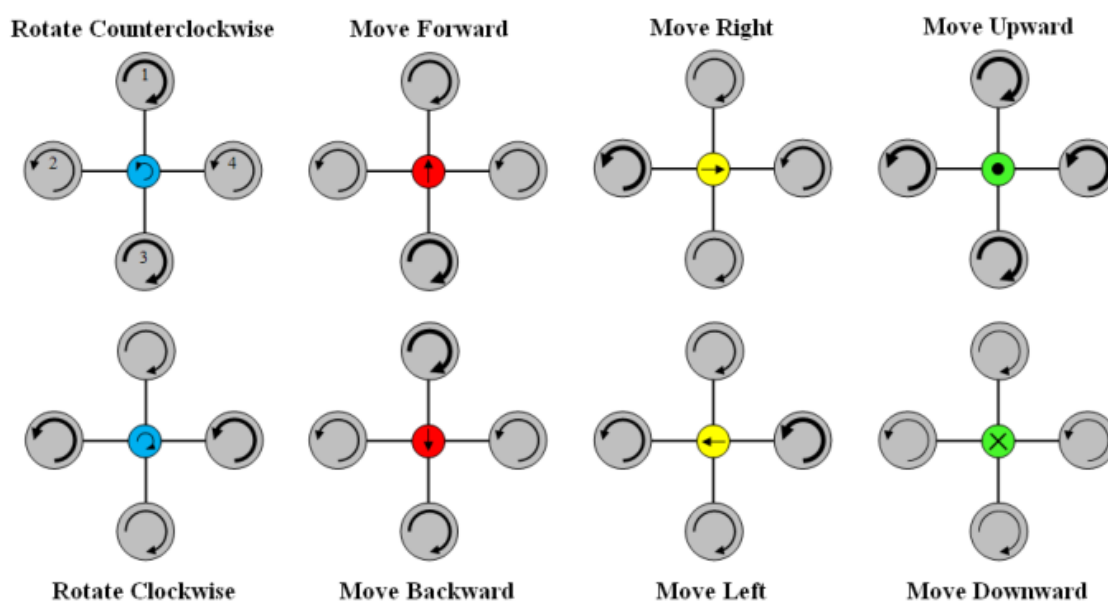
- ใบพัด หน้า และ หลัง (เลข 2 และเลข 4 ในรูปที่ 2.2) จะหมุนทวนเข็มนาฬิกา CCW
- ใบพัด ซ้าย และ ขวา (เลข 1 และเลข 3 ในรูปที่ 2.2) จะหมุนตามเข็มนาฬิกา CW

การเคลื่อนที่ในปริภูมิของโดรนนั้นเราสามารถแบ่งได้ออกเป็น 2 ส่วนคือ การเคลื่อนที่ตามแนวแกน และการเคลื่อนที่หมุนรอบแกน ในการอธิบายการเคลื่อนที่ของโดรนนั้น หากนับองศาอิสระจะได้ทั้งหมดเป็น 6 องศาอิสระ โดย 6 องศาอิสระนี้คือ การเคลื่อนที่ตามแนวแกน 3 แกน (X Y Z) และการเคลื่อนที่หมุนรอบแกน (Roll Pitch Yaw) การควบคุมการเคลื่อนที่ใน 6 องศาอิสระนั้นสามารถทำได้โดยปรับความเร็วการหมุนของมอเตอร์ให้มีความแตกต่างกัน การเคลื่อนที่ไปข้างหน้า ถอยหลัง ไปด้านข้าง ขึ้นลง หมุนรอบ Roll Pitch Yaw, ที่โดรนสามารถหมุนรอบแกน Yaw ได้นั้นเกิดจากทอร์ค



รูปที่ 2.2: ทิศทางการหมุนของแต่ละใบพัด

ของมอเตอร์ทั้งสี่ ผลรวมของทอร์คจะส่งผลต่อความเร็วในการหมุนรอบแกน Yaw หากมอเตอร์ทุกตัวหมุนด้วยความเร็วเท่ากัน ผลรวมทอร์คจะมีค่าเท่ากับศูนย์ทำให้โดรนไม่หมุน แต่ถ้ามอเตอร์หมุนด้วยความเร็วไม่เท่ากัน จะทำให้ผลรวมทอร์คมีค่าไม่เท่ากับศูนย์จะส่งผลให้โดรนเกิดการหมุนรอบแกน Yaw ได้ ถ้ามอเตอร์ทุกตัวหมุนด้วยความเร็วเพิ่มขึ้นหรือลดลงพร้อมกันจะทำให้โดรนเคลื่อนที่ขึ้นหรือลงตามแนวดิ่ง และด้วยการที่เรามี 4 Inputs แต่มี 6 Outputs นั้นทำให้การควบคุมโดรนเป็นแบบ Underactuated ซึ่งมีความซับซ้อนพอสมควร เพื่อที่จะคอนโทรลโดรนได้นั้น ให้สมมุติว่าตัวโดรนเป็นวัตถุชิ้นเดียว (Rigid body) โครงสร้างมีความสมมาตร (Symmetric) ความเร็วของแต่ละมอเตอร์ จะทำให้ใบพัดมีแรงยก เราสามารถบอกลักษณะการเคลื่อนที่ของโดรนได้ ดังรูปที่ 2.3



รูปที่ 2.3: ทิศทางการเคลื่อนที่ของโดรนเมื่อมอเตอร์หมุนด้วยความเร็วต่างๆ

### 2.1.2 Euler angles and Quaternions

#### Euler angles

มุมออยเลอร์เป็นมุม 3 มุมที่คิดโดย Leonhard Euler เพื่อที่จะเอาไว้ใช้อธิบายการเอียงของวัตถุในปริภูมิ ใช้ตัวแปรเพียงแค่ 3 ตัวเท่านั้น การบอกมุมออยเลอร์สามารถบอกได้หลายวิธี ในที่นี้เราใช้ ZYX Euler angles ในการอธิบายมุมเอียงของเฟรมอ้างอิงที่เราสนใจเทียบกับเฟรมอ้างอิงเฟรมอื่น มุมออยเลอร์เกิดจากการหมุนเฟรมรอบแกนสามแกน มาหมุนเรียงต่อกัน ต่อไปจะเป็นการรวมแมทริกการหมุนสามแกนเข้าด้วยกัน

$$R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c(\phi) & -s(\phi) \\ 0 & s(\phi) & c(\phi) \end{bmatrix} \quad (2.1)$$

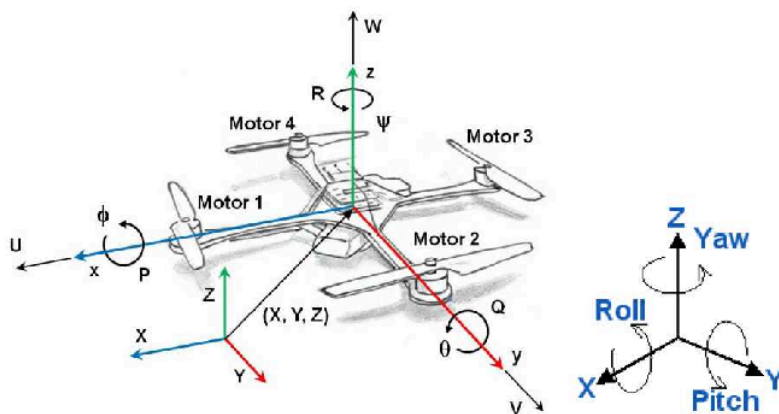
$$R_y(\theta) = \begin{bmatrix} c(\theta) & 0 & s(\theta) \\ 0 & 1 & 0 \\ -s(\theta) & 0 & c(\theta) \end{bmatrix} \quad (2.2)$$

$$R_z(\psi) = \begin{bmatrix} c(\psi) & -s(\psi) & 0 \\ s(\psi) & c(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$

โดยที่  $c(\phi) = \cos(\phi)$ ,  $s(\phi) = \sin(\phi)$ ,  $c(\theta) = \cos(\theta)$ ,  $s(\theta) = \sin(\theta)$ ,  $c(\psi) = \cos(\psi)$ ,  $s(\psi) = \sin(\psi)$  จากแมทริกการหมุนแสดงให้เห็นว่า “Inertial frame” และ “Body frame” มีความสัมพันธ์กันเป็นแมทริกการหมุนคือ  $R_{zyx}(\phi, \theta, \psi)$

$$\begin{aligned} R_{zyx}(\phi, \theta, \psi) &= R_z(\psi)R_y(\theta)R_x(\phi) \\ &= \begin{bmatrix} c(\theta)c(\psi) & s(\phi)s(\theta)c(\psi) - c(\phi)s(\psi) & c(\phi)s(\theta)c(\psi) + s(\phi)s(\psi) \\ c(\theta)s(\psi) & s(\phi)s(\theta)s(\psi) + c(\phi)c(\psi) & c(\phi)s(\theta)s(\psi) - s(\phi)c(\psi) \\ -s(\theta) & s(\phi)c(\theta) & c(\phi)c(\theta) \end{bmatrix} \end{aligned} \quad (2.4)$$

แมทริกจากสมการที่ 2.4 เป็นแมทริกที่อธิบายถึงการหมุนจาก “Body frame” ไปยัง “Inertial frame” ดังรูปที่ 2.4



รูปที่ 2.4: ภาพแสดงการบอกแมทริกการหมุนของ Body frame

## Quaternions

ในบางครั้งหรือบางกรณี Euler angles อาจจะทำให้เกิด Gimbal lock ได้ ซึ่งจะทำให้การควบคุมเสียไป 1 องศาอิสระ เพื่อที่จะหลีกเลี่ยงเหตุการณ์นี้จึงได้นำ quaternions มาใช้ quaternions ใช้เพื่อที่จะบอกมุมเอียงของวัตถุใดๆ ได้เหมือนกับ Euler angles โดยที่เราสามารถแปลง Euler angles ให้เป็น quaternions ได้จากสมการที่ 2.5

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_4 \end{bmatrix} = \begin{bmatrix} c(\frac{\phi}{2})c(\frac{\theta}{2})c(\frac{\psi}{2}) + s(\frac{\phi}{2})s(\frac{\theta}{2})s(\frac{\psi}{2}) \\ s(\frac{\phi}{2})c(\frac{\theta}{2})c(\frac{\psi}{2}) + c(\frac{\phi}{2})s(\frac{\theta}{2})s(\frac{\psi}{2}) \\ c(\frac{\phi}{2})s(\frac{\theta}{2})c(\frac{\psi}{2}) + s(\frac{\phi}{2})c(\frac{\theta}{2})s(\frac{\psi}{2}) \\ c(\frac{\phi}{2})c(\frac{\theta}{2})s(\frac{\psi}{2}) + s(\frac{\phi}{2})s(\frac{\theta}{2})c(\frac{\psi}{2}) \end{bmatrix} \quad (2.5)$$

และเราสามารถที่จะเขียนแมทริกการหมุนให้อยู่ในรูปของ quaternions ได้ดังสมการที่ 2.6

$$R_{zyx}(\phi, \theta, \psi) = \begin{bmatrix} q_1^2 + q_2^2 - q_3^2 - q_4^2 & 2(q_2q_3 - q_1q_4) & 2(q_1q_3 + q_2q_4) \\ 2(q_2q_3 + q_1q_4) & q_1^2 - q_2^2 + q_3^2 - q_4^2 & 2(q_3q_4 - q_1q_2) \\ 2(q_2q_4 - q_1q_3) & 2(q_1q_2 - q_3q_4) & q_1^2 - q_2^2 - q_3^2 + q_4^2 \end{bmatrix} \quad (2.6)$$



### 2.1.3 Qaudroter mathematical model

ในส่วนนี้จะเป็นการอธิบายสมการการเคลื่อนที่ของโดรน โดยใช้สมการของ Newton และ Euler มาช่วยในการอธิบายพลวัต (Dynamics) ของโดรน เพื่อใช้ทำแบบจำลอง (Simulating) และควบคุม (Controlling) ทำทางของโดรนด้วย เริ่มจากให้  $[x \ y \ z \ \phi \ \theta \ \psi]^T$  เป็นเวกเตอร์ที่บอกตำแหน่งและมุม (linear/angular position) ของโดรนโดยเทียบจากเฟรมโลก (Inertial frame) และ  $[u \ v \ w \ p \ q \ r]^T$  เป็นเวกเตอร์ที่บอกความเร็วเชิงเส้นและความเร็วเชิงมุม (linear/angular velocity) ของโดรนโดยเทียบจากเฟรมโดรน (Body frame) พลวัตของโดรนจะเปิดจากเฟรมอ้างอิงสองเฟรมนี้มีความสัมพันธ์กัน

$$\begin{aligned}\nu &= R\nu_B \\ \omega &= T\omega_B\end{aligned}\tag{2.7}$$

โดยที่  $\nu = [\dot{x} \ \dot{y} \ \dot{z}]^T$ ,  $\omega = [\dot{\phi} \ \dot{\theta} \ \dot{\psi}]^T$ ,  $\nu_B = [u \ v \ w]^T$ ,  $\omega_B = [p \ q \ r]^T$  และ  $T$  เป็นเมทริกซ์การแปลงมุมการหมุน (angular transformation)

$$T = \begin{bmatrix} 1 & s(\phi)t(\theta) & c(\phi)t(\theta) \\ 0 & c(\phi) & -s(\phi) \\ 0 & \frac{s(\phi)}{c(\theta)} & \frac{c(\phi)}{c(\theta)} \end{bmatrix}\tag{2.8}$$

โดยที่  $t(\theta) = \tan(\theta)$  ดังนั้นเราจะได้สมการจลศาสตร์ (Kinematic model) ของโดรนเป็น

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} w[s(\phi)s(\psi) + c(\phi)c(\psi)s(\theta)] - v[c(\phi)s(\psi) - c(\psi)s(\phi)s(\theta)] + u[c(\psi)c(\theta)] \\ v[c(\phi)c(\psi) + s(\phi)s(\psi)s(\theta)] - w[c(\psi)s(\phi) - c(\phi)s(\psi)s(\theta)] + u[c(\theta)s(\psi)] \\ w[c(\phi)c(\theta)] - u[s(\theta)] + v[c(\theta)s(\phi)] \\ p + r[c(\phi)t(\theta)] + q[s(\phi)t(\theta)] \\ q[c(\phi)] - r[s(\phi)] \\ r\frac{c(\phi)}{c(\theta)} + q\frac{s(\phi)}{c(\theta)} \end{bmatrix}\tag{2.9}$$

จากกฎของนิวตันระบุความสัมพันธ์ของแรงรวมที่กระทำต่อโดรนดังเมทริกซ์ต่อไปนี้

$$m(\omega_B \wedge \nu_B + \dot{\nu}_B) = \mathbf{f}_B\tag{2.10}$$

โดย  $m$  คือน้ำหนักของโดรน,  $\wedge$  คือ cross product และ  $\mathbf{f}_B = [f_x \ f_y \ f_z]^T \in \mathbf{R}^3$  คือแรงรวมจากสมการออยเลอร์ ให้แรงบิดรวมที่ใช้กับโดรน เป็นไปดังนี้

$$I.\dot{\omega}_B + \omega_B \wedge (I.\omega_B) = \mathbf{m}_B\tag{2.11}$$

โดย  $\mathbf{m}_B = [m_x \ m_y \ m_z]^T \in \mathbf{R}^3$  เป็นแรงบิดรวม และ  $I$  เป็นเมทริกซ์ของความเฉื่อย :

$$I = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix} \wedge \mathbf{R}^{3 \times 3}\tag{2.12}$$

ดังนั้น จะได้ dynamic model ของโดรนโดยอ้างอิง body frame ดังนี้

$$\begin{bmatrix} f_x \\ f_y \\ f_z \\ m_x \\ m_y \\ m_z \end{bmatrix} = \begin{bmatrix} m(\dot{u} + qw - rv) \\ m(\dot{v} - pw + ru) \\ m(\dot{w} + pv - qu) \\ \dot{p}I_x - qrI_y + qrI_z \\ \dot{q}I_y + prI_x - prI_z \\ \dot{r}I_z - pqI_x + pqI_y \end{bmatrix}\tag{2.13}$$

#### 2.1.4 Motor model

ในแต่ละใบพัดของ quadrotor จะให้แรงที่มีทิศทางตรงข้ามกับแกน  $Z$  ของ body frame แรงที่เกิดจากการหมุนจะทำให้เกิดแรงบิดรอบแนวแกน roll pitch yaw โดยแรงและแรงบิดจะขึ้นอยู่กับความเร็วของใบพัดและค่าคงที่ค่าหนึ่ง ซึ่งสามารถแสดงความสัมพันธ์ของแรงได้ดังสมการที่ 2.14 และความสัมพันธ์ของแรงบิดได้ดังสมการที่ 2.15

$$F_i = k_f \Omega_i^2 \quad (2.14)$$

$$T_i = -k_t \text{sgn}(i) \Omega_i^2 \quad (2.15)$$

โดยที่ตัวห้อย  $i$  จะแสดงถึงลำดับของใบพัดแต่ละตัว  $\Omega_i$  แสดงถึงความเร็วการหมุนของใบพัด  $k_f$  และ  $k_t$  เป็นค่าคงที่การคูณของแรงและแรงบิด  $\text{sgn}$  เป็นฟังก์ชันที่ใช้บอกทิศทางการหมุนของใบพัดที่มีความแตกต่างกันตามการติดตั้งแรงทั้งหมดที่เกิดบน quadrotor สามารถที่จะเขียนสมการให้อยู่ในรูปของความเร็วการหมุนของใบพัดคูณกับแมทริกการแปลง  $M$  ดังสมการที่ 2.16

$$\begin{bmatrix} F \\ \tau_\phi \\ \tau_\theta \\ \tau_\psi \end{bmatrix} = M \begin{bmatrix} \Omega_1^2 \\ \Omega_2^2 \\ \Omega_3^2 \\ \Omega_4^2 \end{bmatrix} \quad (2.16)$$

โดยที่ แมทริกการแปลง  $M$  มีการกำหนดไว้ดังสมการที่ 2.17

$$M = \begin{bmatrix} k_f & k_f & k_f & k_f \\ \frac{k_f d}{\sqrt{2}} & -\frac{k_f d}{\sqrt{2}} & -\frac{k_f d}{\sqrt{2}} & \frac{k_f d}{\sqrt{2}} \\ \frac{k_f d}{\sqrt{2}} & \frac{k_f d}{\sqrt{2}} & -\frac{k_f d}{\sqrt{2}} & -\frac{k_f d}{\sqrt{2}} \\ k_t & -k_t & k_t & -k_t \end{bmatrix} \quad (2.17)$$

โดยที่  $\Omega_i^2$  เป็นความเร็วการหมุนของใบพัดกำลังสอง และ  $d$  คือระยะทางจากจุดศูนย์กลางมวลของ quadrotor ไปถึงจุดกึ่งกลางของใบพัด

## 2.2 Linear Quadratic Regulator

## 2.3 Hector Quadrotor ROS Stack

Hector Quadrotor เป็น ROS stacks (stack เป็นการรวบรวม packages ที่มีหลากหลายฟังก์ชันเอาไว้ด้วยกัน) โดยภายในประกอบไปด้วย ROS packages ดังต่อไปนี้

- 1) hector\_quadrotor\_description
- 2) hector\_quadrotor\_gazebo
- 3) hector\_quadrotor\_teleop
- 4) hector\_quadrotor\_gazebo\_plugin

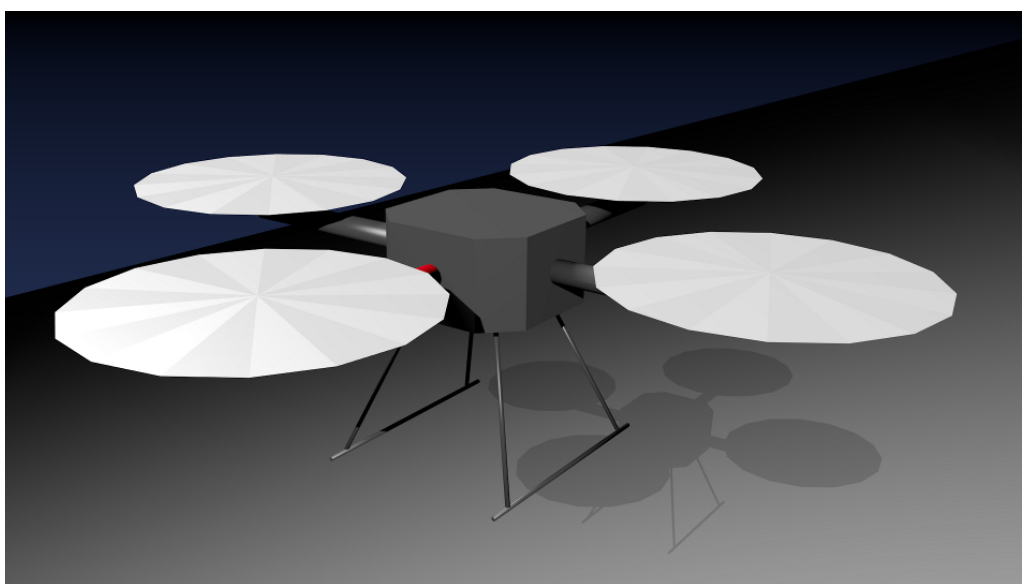
### 2.3.1 hector\_quadrotor\_description

package นี้ให้ URDF model ของ quadrotor มี visual geometry เป็นไฟล์ COLLADA และ collision geometry เป็นไฟล์ STL โดยจะเป็นส่วนเสริมที่ทำให้เราสามารถที่จะใช้ model ในโปรแกรม Gazebo ได้<sup>1</sup>

**quadroter\_base.urdf.xacro** เป็นไฟล์ xacro ของโมเดลขั้นพื้นฐานของ quadroter

**quadrotor.urdf.xacro** เป็นไฟล์ xacro สำหรับการแสดงโมเดลขั้นพื้นฐานโดยไม่มีเซนเซอร์เพิ่มเติม

**quadrotor\_hokuyo\_utm30lx.urdf.xacro** เป็นไฟล์ xacro สำหรับการแสดงโมเดลขั้นพื้นฐานโดยเพิ่มเติมเซนเซอร์ Hokuyo เข้าไปสำหรับสแกนสิ่งแวดล้อม



รูปที่ 2.5: ภาพ quadrotor ที่ render ด้วย Blender

### 2.3.2 hector\_quadrotor\_gazebo

package นี้ให้ quadroter model ที่มาจากไฟล์ urdf และมาแสดงผลในโปรแกรม Gazebo โดย features<sup>2</sup> ที่มีคือ

- 1) Colored COLLADA quadrotor mesh model (.dae)
- 2) URDF description
- 3) Publishing of ground truth pose and simulated imu data
- 4) Spawnable in Gazebo
- 5) Controller สำหรับใช้ใน Gazebo โดยสามารถควบคุมผ่าน geometry\_msgs/Twist message ใน topic 'cmd\_vel'

<sup>1</sup>[http://wiki.ros.org/hector\\_quadrotor\\_description](http://wiki.ros.org/hector_quadrotor_description)

<sup>2</sup>[http://wiki.ros.org/hector\\_quadrotor\\_gazebo](http://wiki.ros.org/hector_quadrotor_gazebo)

### 2.3.3 hector\_quadrotor\_teleop

package นี้จะทำให้เราสามารถที่จะควบคุม quadrotor ผ่าน gamepad ได้ โดยจะมี node ที่ publish geometry\_msgs/Twist message ไปที่ topic 'cmd\_vel' ปัจจุบันมี gamepad 2 ชนิดที่นำมาใช้ได้ก็คือ

- 1) logitech\_gamepad.launch สำหรับ Logitech gamepads
- 2) xbox\_controller.launch สำหรับ Xbox controller gamepads

แต่หากต้องการใช้งานนอกเหนือจากนี้ สามารถที่จะสร้าง node ขึ้นมาใหม่ได้

### 2.3.4 hector\_quadrotor\_gazebo\_plugin

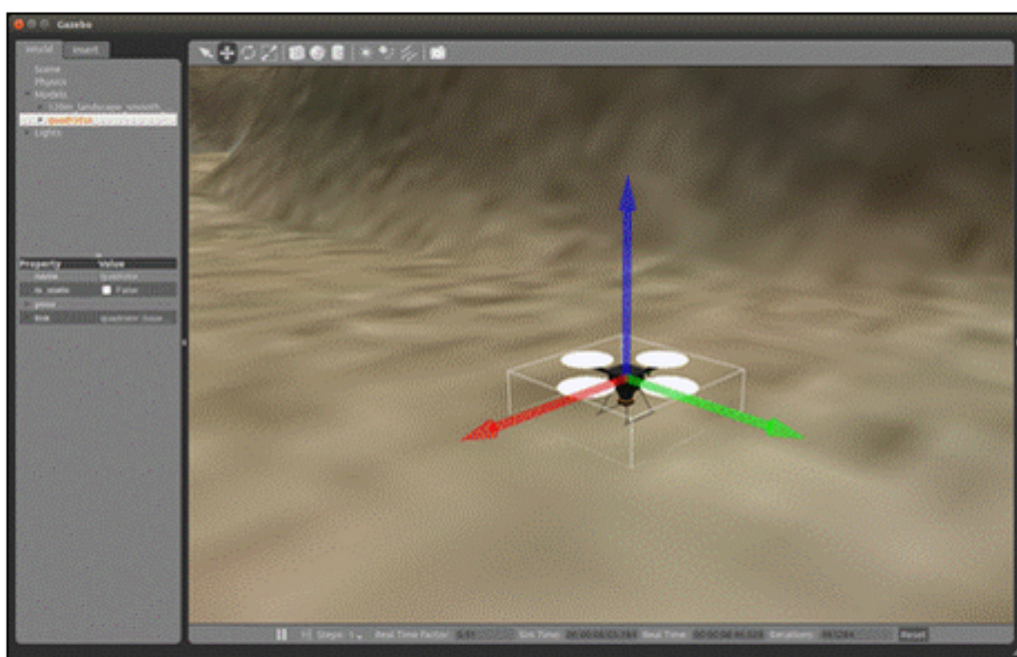
package นี้เป็นส่วนเสริมของ Gazebo ที่จะทำให้มีเซนเซอร์ต่างๆเข้ามาในระบบได้

**Barometer Plugin** เป็นส่วนเสริมที่จำลองเซนเซอร์ที่ใช้วัดความสูงของ quadrotor โดยมีพื้นฐานมาจากการวัด barometric pressure

**Simple Controller Plugin** package นี้เป็นส่วนเสริมของ Gazebo ที่จะทำให้เราสามารถควบคุม Velocities และ Yaw rate ได้โดยการคำนวณ แรงและแรงบิด โดยในปัจจุบันยังไม่สามารถที่จะควบคุมตำแหน่ง ทิศทางการหันหัว และความสูงได้ แต่สามารถที่จะเขียนทับเข้าไปได้

**Quadrotor Propulsion Plugin / Quadrotor Aerodynamics Plugin** package นี้เป็นส่วนเสริมของ Gazebo ที่จะทำให้เราสามารถควบคุมการหมุนของใบพัด quadrotor ได้ โดยการควบคุม voltages ของมอเตอร์ และยังสามารถจำลอง wind vector ได้อีกด้วย

**Matlab Compilation** ในการเขียนสมการต่างๆของระบบ quadrotor มักจะนิยมใช้ Matlab script และคอมไพล์เป็น C libraries โดยใช้ Matlab Coder ส่วนเสริมตัวนี้จะช่วยให้สามารถเชื่อมต่อกับ Matlab ได้



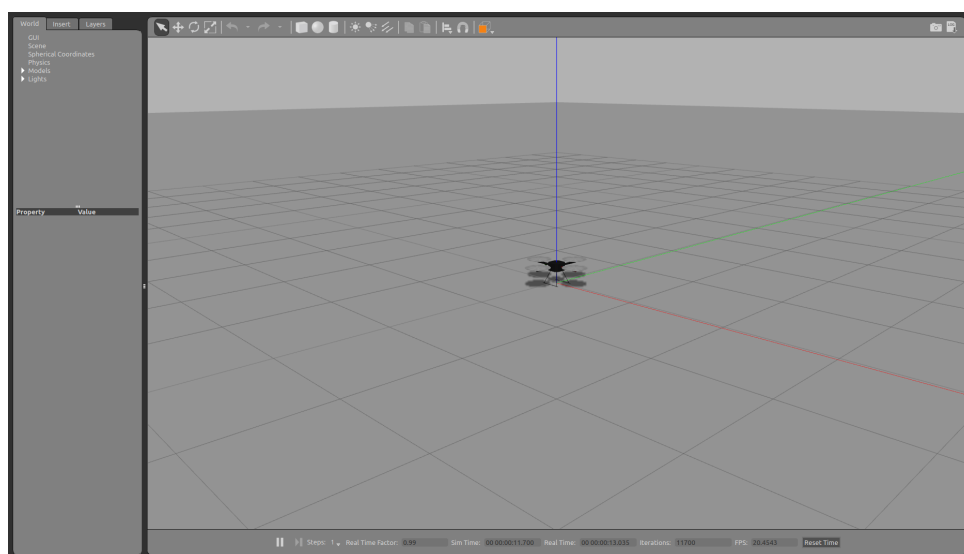
รูปที่ 2.6: ภาพ quadrotor ในโปรแกรม Gazebo

## บทที่ 3

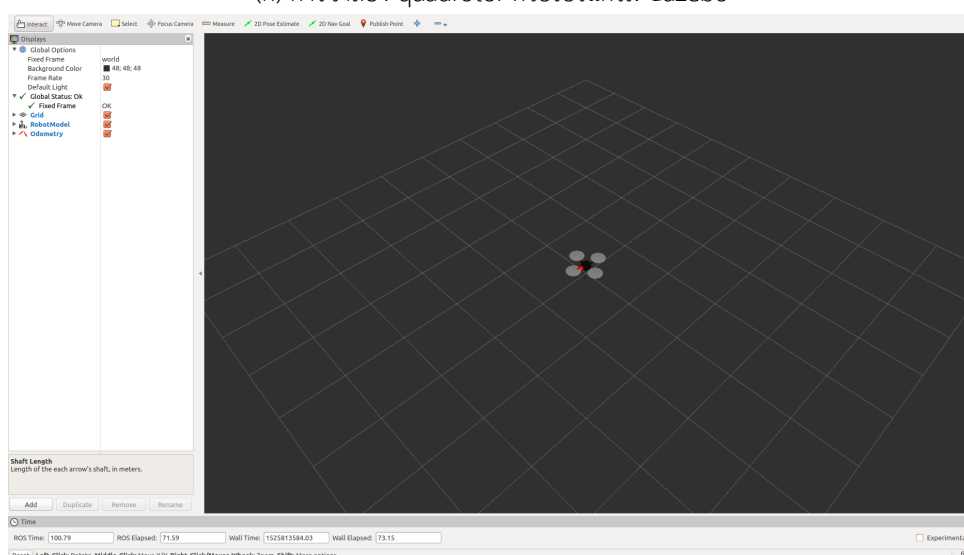
### Simulation

ทดสอบการทำงานของโปรแกรม Matlab โดยจำลองในโปรแกรม Gazebo และแสดงผลภาพใน Rviz การทดสอบแบ่งเป็น 3 ตัวอย่าง

- 1) สั่งให้ quadrotor บินขึ้นเหนือพื้น
- 2) สั่งให้ quadrotor บินวนเป็นเกลียว
- 3) สั่งให้ quadrotor บินเป็นรูปดาว



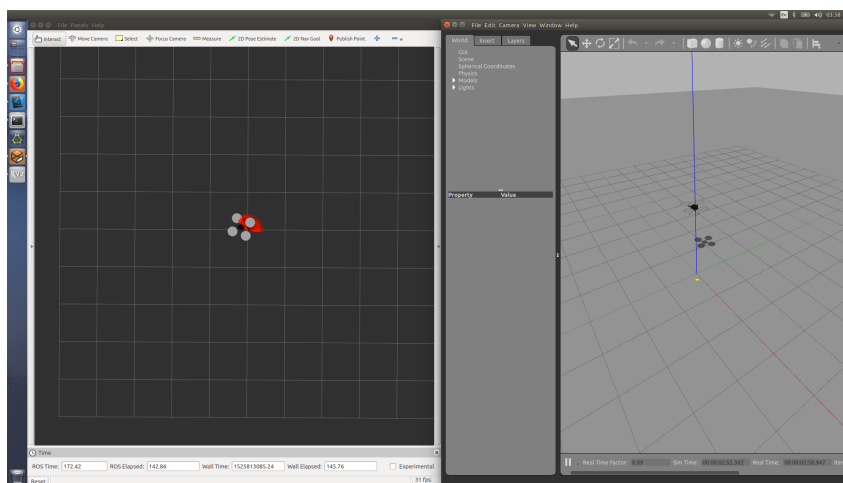
(ก) การจำลอง quadrotor ด้วยโปรแกรม Gazebo



(ข) การแสดงผลภาพ quadrotor ด้วยโปรแกรม Rviz

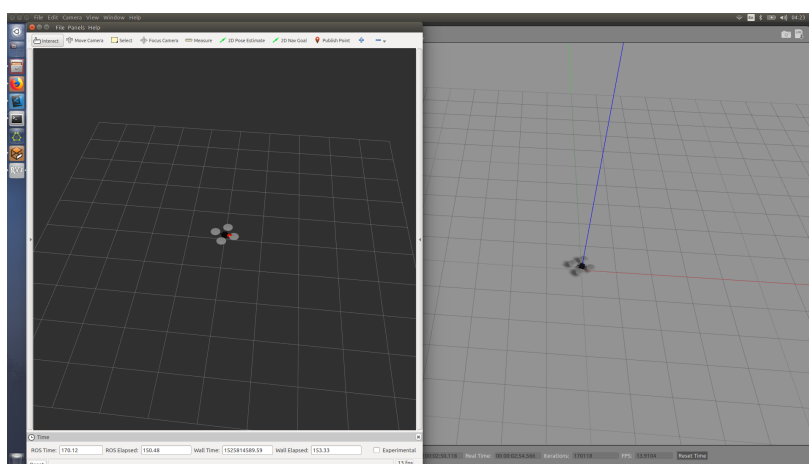
รูปที่ 3.1: การจำลองและการแสดงผลภาพของ quadrotor

### 3.1 สั่งให้ quadrotor บินขึ้นเหนือพื้น

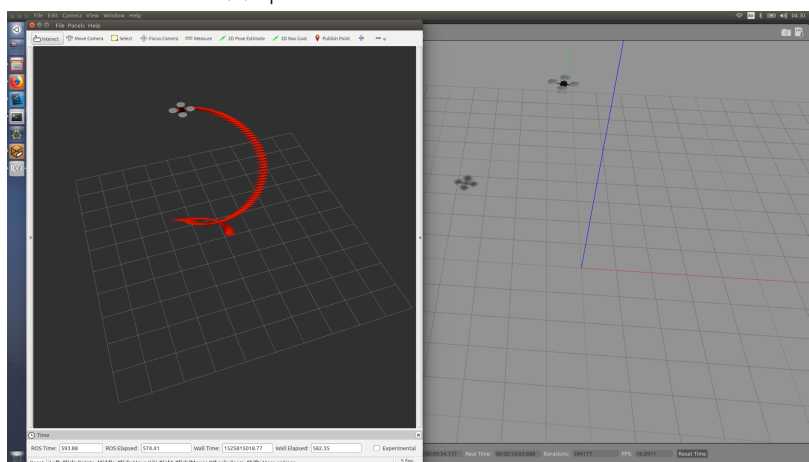


รูปที่ 3.2: ผลการสั่งให้ quadrotor บินขึ้นเหนือพื้น 2 เมตร

### 3.2 สั่งให้ quadrotor บินวนเป็นเกลียว



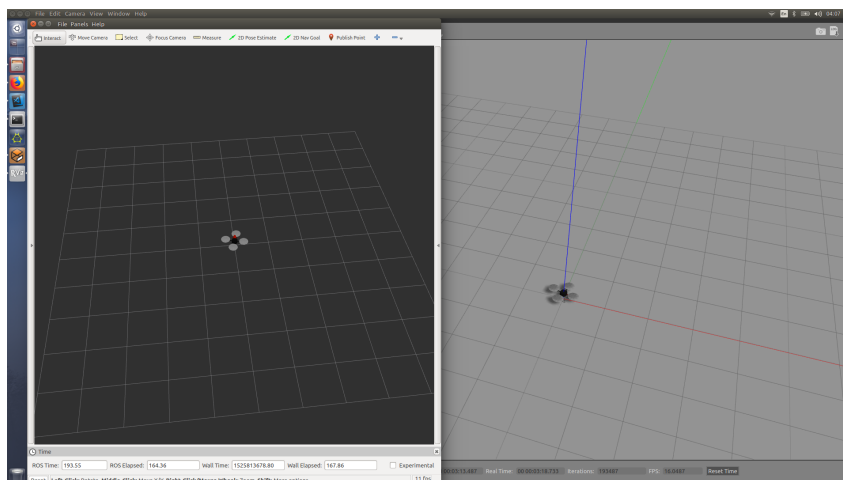
(ก) quadrotor บินวนเป็นเกลียว ชั้นที่ 1



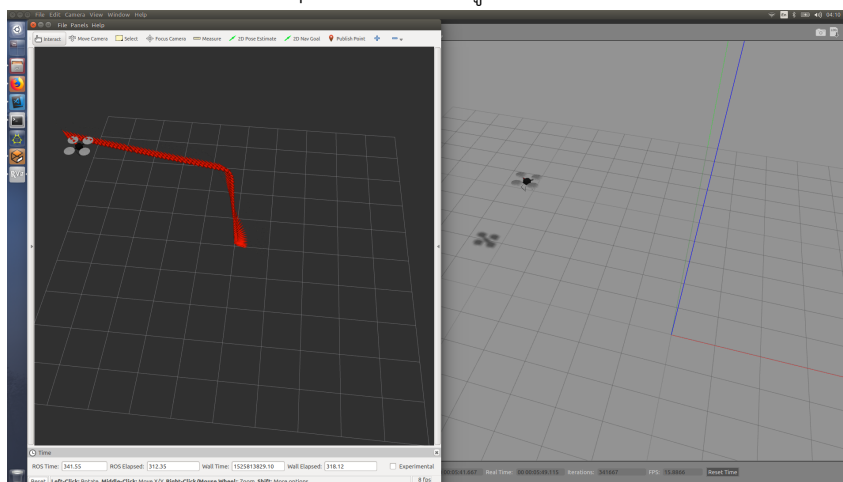
(ข) quadrotor บินวนเป็นเกลียว ชั้นที่ 2

รูปที่ 3.3: ผลการสั่งให้ quadrotor บินวนเป็นเกลียว

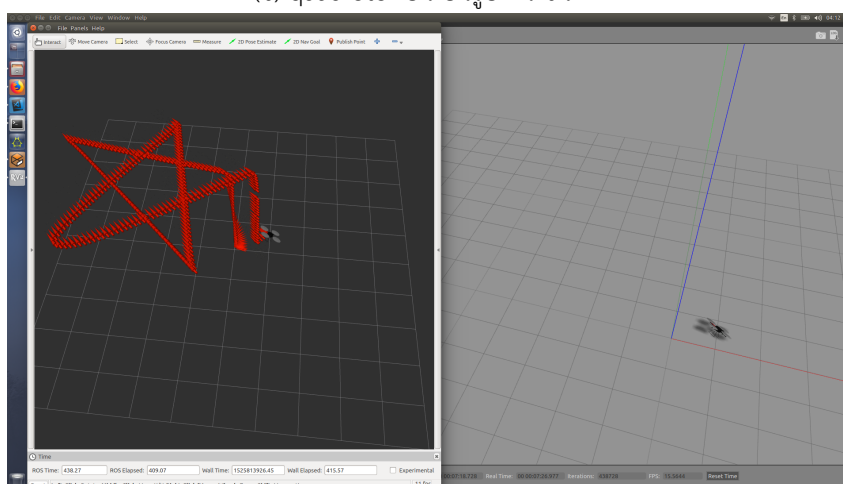
### 3.3 สั่งให้ quadrotor บินเป็นรูปดาว



(ก) quadrotor บินเป็นรูปดาว ขั้นที่ 1



(ข) quadrotor บินเป็นรูปดาว ขั้นที่ 2



(ค) quadrotor บินเป็นรูปดาว ขั้นที่ 3

รูปที่ 3.4: ผลการสั่งให้ quadrotor บินเป็นรูปดาว



## บทที่ 4

### Analysis/Discussion

## บทที่ 5

### Conclusion