NATIONAL CHENG KUNG UNIVERSITY
Department of Aeronautics and Astronautics
Tainan, Taiwan, R.O.C.

Homework #

Control

INSTRUCTOR: Prof. (教授)
STUDENT: Chih, Chao-Hcien (池昭賢)
STUDENT ID: P46104269

November 26, 2024

這是一個模板供使用。

I'm writing to demonstrate use of automatically-generated footnote markers[1] and footnotes which use a marker value provided to the command [42].

Now, I will use another automatically-generated footnote marker [2].

1 Equation Example

In this paper, the 3–2–1 intrinsic convention is considered. The direction cosine matrix (DCM), ${}^{\mathcal{G}}\mathbf{C}^{\mathcal{B}}(\phi, \theta, \psi)$, is constructed in terms of the Euler angles as follows:

$$ {}^{\mathcal{G}}\mathbf{C}^{\mathcal{B}}(\phi, \theta, \psi) = \mathbf{R}_z(\psi)\mathbf{R}_y(\theta)\mathbf{R}_x(\phi) \tag{1} $$

where

$$ \begin{aligned} \mathbf{R}_x(\phi) &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi & c_\phi \end{bmatrix}, \quad \mathbf{R}_y(\theta) = \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix}, \\ \mathbf{R}_z(\psi) &= \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{2} $$

are the rotation matrices and the notations $s_{(\cdot)}$ and $c_{(\cdot)}$ represent $\sin(\cdot)$ and $\cos(\cdot)$, respectively.

$$ \begin{aligned} \dot{\mathbf{e}}_{1,\Theta} &= \mathbf{e}_{2,\Theta} \\ \dot{\mathbf{e}}_{2,\Theta} &= -\mathbf{T}^{-1}\mathbf{J}^{-1}\mathbf{J}_0\mathbf{T}\Big(\mathbf{K}_{p,\Theta}\mathbf{e}_{1,\Theta} + \mathbf{K}_{i,\Theta}\int_0^t \mathbf{e}_{1,\Theta}\, d\tau \\ &\quad + \mathbf{K}_{d,\Theta}\mathbf{e}_{2,\Theta}\Big) + [\mathbf{JT}]^{-1}\mathbf{d}_\Theta \\ &\quad + \left([\mathbf{JT}]^{-1}\mathbf{J}_0\mathbf{T} - \mathbf{I}_3\right)\ddot{\mathbf{x}}_{1d,\Theta} \\ &\quad - [\mathbf{JT}]^{-1}\left([\mathbf{T}\mathbf{x}_{2,\Theta}]^{\times}\Delta\mathbf{J}\mathbf{T}\mathbf{x}_{2,\Theta} - \Delta\mathbf{J}\dot{\mathbf{T}}\mathbf{x}_{2,\Theta}\right) \end{aligned} \tag{3} $$

2 Figure Example

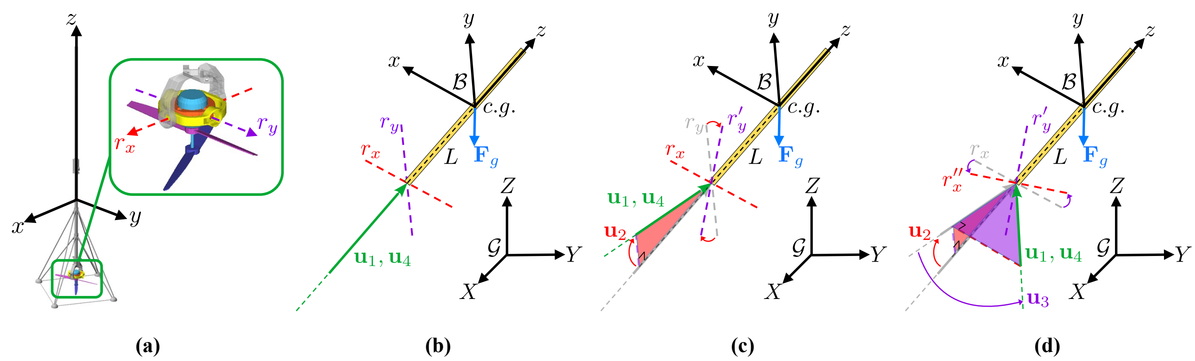

Figure. 1. Illustration of the rocket-type UAV force analysis.

[1] Automatically generated footnote markers work fine!

[42] ...is that the answer to everything?

[2] Now, footnote markers are 1, 42, but then back to 2? That will be confusing if the automatically-generated number also reaches 42!

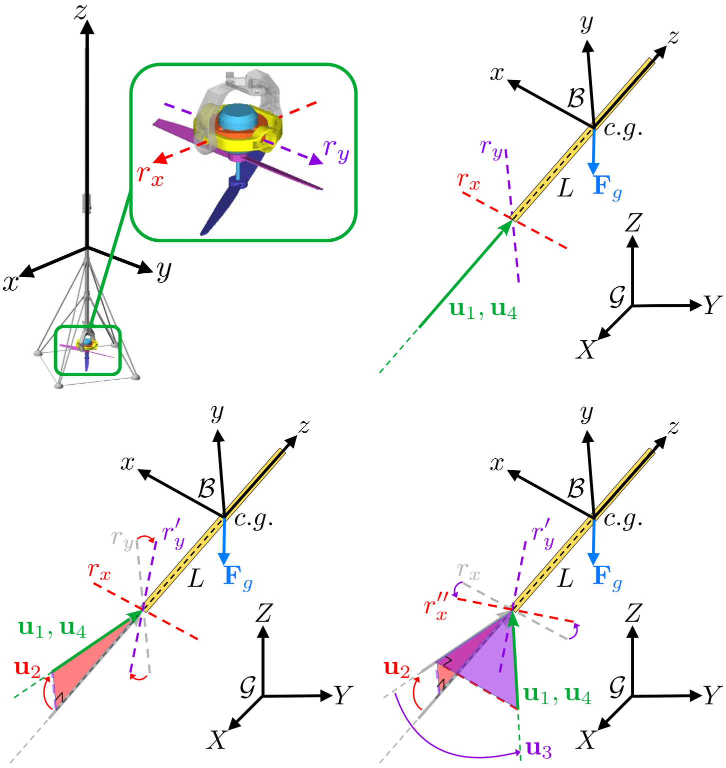

Figure. 2. Illustration of the rocket-type UAV force analysis.

3 Table Example

Table. 1. Simulated system parameters

m (kg)	g (m/s^2)	$\mathbf{J}$ (kg-m^2)	L (m)	C_d
0.65	9.8	$\begin{bmatrix} 0.1287 & 0 & 0 \\ 0 & 0.1287 & 0 \\ 0 & 0 & 0.052 \end{bmatrix}$	0.45	0.05

Table. 2. Levenberg-Marquardt Method pseudo code.

Algorithm 1 Levenberg-Marquardt Method

given an initial value $\mathbf{u}^{(0)}$, $\lambda^{(0)} = 1000$, $\epsilon = 10^{-5}$.

repeat

1. Determine a Jacobian matrix $\mathbf{J}_{\mathbf{r}}^{(k)}$.
2. Update the damping parameter $\lambda^{(k)}$.
3. Update the LM step.
   $\mathbf{d}^{(k)} = -\left(\mathbf{J}_{\mathbf{r}}^{(k)T}\mathbf{J}_{\mathbf{r}}^{(k)} + \lambda^{(k)}\mathbf{I}_4\right)^{-1}\mathbf{J}_{\mathbf{r}}^{(k)T}\mathbf{r}(\mathbf{u}^{(k)})$.
4. Update the control variables.
   $\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \mathbf{d}^{(k)}$.
   $k \leftarrow k+1$.
5. Compute the residual error vector to evaluate the stopping condition.
   $\mathbf{r}(\mathbf{u}^{(k+1)}) = [r_1, r_2, r_3, r_4]^T$

until $\|\mathbf{r}\| < \epsilon$ is satisfied, $\mathbf{u}^* = \mathbf{u}^{(k+1)}$.

4 Subfiles Example

4.1 Subfiles

$$\left[\begin{array}{c|cc} x_1 & x_2 & x_3 \\ \hline x_1 & x_2 & x_3 \\ x_1 & x_2 & x_3 \end{array}\right] \tag{4}$$

A Matlab Code

```
clc; clear; close all;

%% Simulink

dT = 0.001;
t = 0.01:dT:3;

A = [0.9 -2.1  0.3  0.5  0.9;
      0.7  0.2  0.3  0.0  0.7;
     -1.3  0.5 -1.1 -2.3 -0.2;
     -0.3  0.8  1.7 -1.2 -0.3;
     -3.5 -4.3 -0.7  0.0 -8.3];
B = [1.0 -0.4;-1.7 -1.7;-0.2 1.2;0.7 0.1;0.9 1.4];
C = [1 -2 -1 0 3;0 2 1 0 4];

Css = eye(5); Dss = zeros(5,2);
Dss2 = zeros(5,5);
```

B C++ Code

```
#include <Arduino.h>
#include <MPU9250.h>
#include <BMP280_DEV.h>

#define PIN_BUTTON 6
#define PIN_LED 5

#define PIN_SERVO_RX 4
#define PIN_SERVO_RY 3

#define PWM_FREQUENCY 50.0
#define PWM_RESOLUTION 15
#define PWM_MAX_VALUE pow(2,PWM_RESOLUTION) - 1
#define PWM_1MS round(1000.0 / (1000000.0 / PWM_FREQUENCY) * PWM_MAX_VALUE)
#define PWM_MIN round(0.7 * PWM_1MS)
#define PWM_MID round(1.5 * PWM_1MS)
#define PWM_MAX round(2.3 * PWM_1MS)



int jump_flag = 0;
double pre_mag[3];

// Debounce variables
unsigned long lastDebounceTime = 0;
unsigned long debounceDelay = 100;

void button_triger();


// Project Begin
void setup()
{
    // Serial.begin(38400);
    Serial.begin(115200);
    pinMode(PIN_BUTTON,INPUT);
    attachInterrupt(PIN_BUTTON,button_triger,RISING);
    pinMode(PIN_LED,OUTPUT);
    digitalWrite(PIN_LED,0);

    pinMode(PIN_SERVO_RX, OUTPUT);
    pinMode(PIN_SERVO_RY, OUTPUT);
    analogWriteResolution(PWM_RESOLUTION);
    analogWriteFrequency(PIN_SERVO_RX, PWM_FREQUENCY);
    analogWriteFrequency(PIN_SERVO_RY, PWM_FREQUENCY);
```

```
    // analogWrite(PIN_SERVO_RX, PWM_MAX);
    // analogWrite(PIN_SERVO_RY, PWM_MAX);
}

void loop()
{
    // put your main code here, to run repeatedly:
    float sensorVoltage;
    float sensorValue;

    sensorValue = analogRead(A1);
    sensorVoltage = sensorValue/1024*3.3;
    Serial.print(sensorValue);
    Serial.print(" ");
    Serial.print(sensorVoltage,6);
    Serial.println(" ");
    delay(10);
}

// put function definitions here:
void button_triger() {
    // Read the state of the button
    int buttonState = digitalRead(PIN_BUTTON);

    // Check if the button state has changed
    if (buttonState == HIGH) {
        // Check if it's been long enough since the last button press to consider it a valid press
        if ((millis() - lastDebounceTime) > debounceDelay) {
            // Toggle the LED state
            digitalWrite(PIN_LED, !digitalRead(PIN_LED));
        }
        // Update the last debounced time
        lastDebounceTime = millis();
    }
}
```