目次

第 1 章

はじめに

1.1　研究の背景

ドローンとは，無人で遠隔操作や自動制御によって飛行できる航空機の総称のことである．ドローンの始まりは図 1.1 に示す軍事用に偵察や攻撃用に作られたプレデターが始まりである．近年，図 1.2 に示す宅配サービスや図 1.3 に示す農薬を撒く農業用，図 1.4 図 1.5 に示す空撮，水中撮影，図 1.6 に示す掌に収まるくらいの小型用，図 1.7 に示す娯楽等のドローンが普及してきた．3 つ以上のメインローターを持つヘリコプターをマルチコプターと呼び，その中から 4 つのメインローターを持つクアッドコプター図 1.8 について研究する．室内で使うことのできるマルチコプターは多くない．

1.2　研究の目的

自動制御の課題がいくつかあり，その中に高度制御を利用した自動着陸があり，課題を達成するため高度制御について研究する．

1.3　本論文の構成

1 章では, 本研究の背景と簡略化した概要を示す．2 章では飛行ロボコンについて述べる．3 章ではマルチコプター運動方程式について述べる．4 章では高度制御について述べる．

図 1.1　プレデター

図 1.2　商業用

図 1.3　農業用

図 1.4　空撮

図 1.5　水中撮影用

5 章では高度制御のシミュレーション実験について述べる．6 章で考察を述べる．7 章で最後に本実験のまとめを述べる．

図 1.6　小型の娯楽用

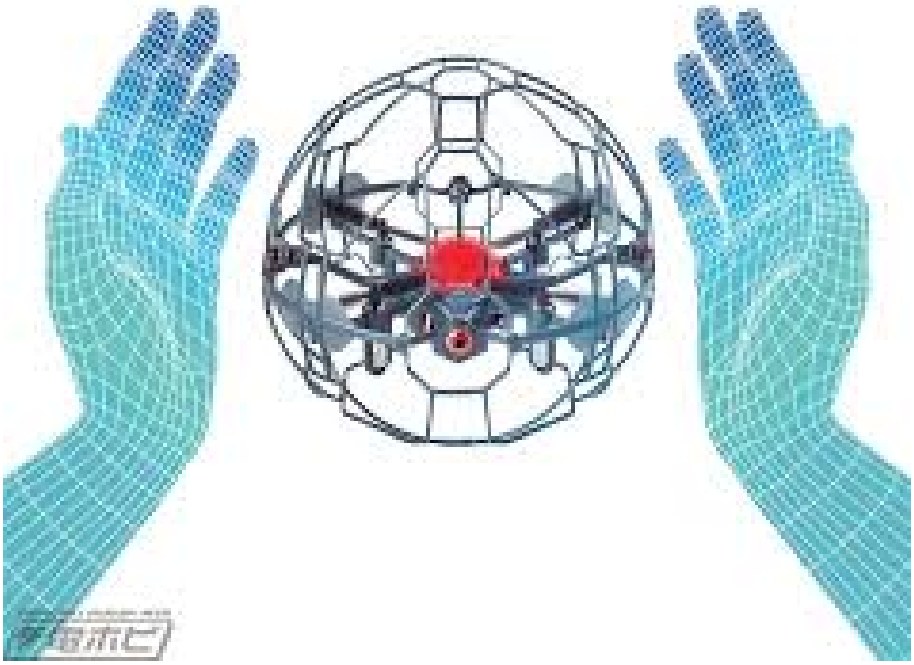

図 1.7　娯楽用

図 1.8　クアッドコプター

第 2 章

飛行ロボコン

2.1　大会ルール

ヘリポートから飛行を開始し，ミッションエリアにてミッションを完了したのち，ヘリポートに帰還する．大会のフィールドを図 2.1 に示す．

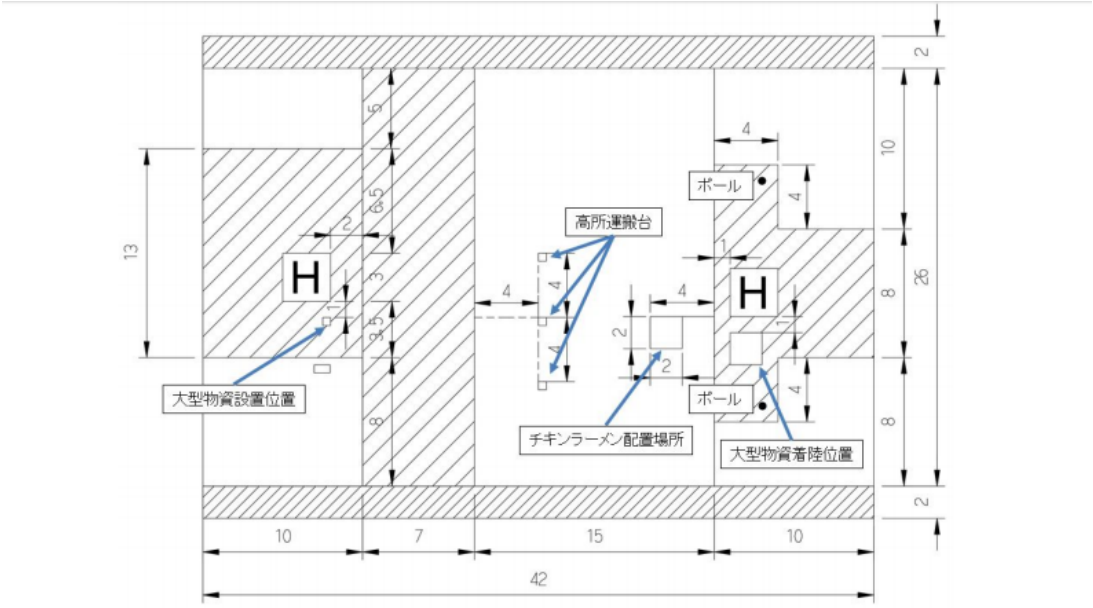

図 2.1　大会フィールド

2.2　機体条件

大会に出場するための機体には以下の条件を満たさないといけない．

1) 空虚重量が 350g 以下であり，地上補助装置との合計重量が 500g 以

下である．

2) 機体は自作である．

3) 推進力として2つ以上のプロペラを搭載する機体である．

4) 搭載する全てのプロペラ周りに安全のためにプロペラガードなどを取り付けること．

大会に出場した機体を図2.2に示す．

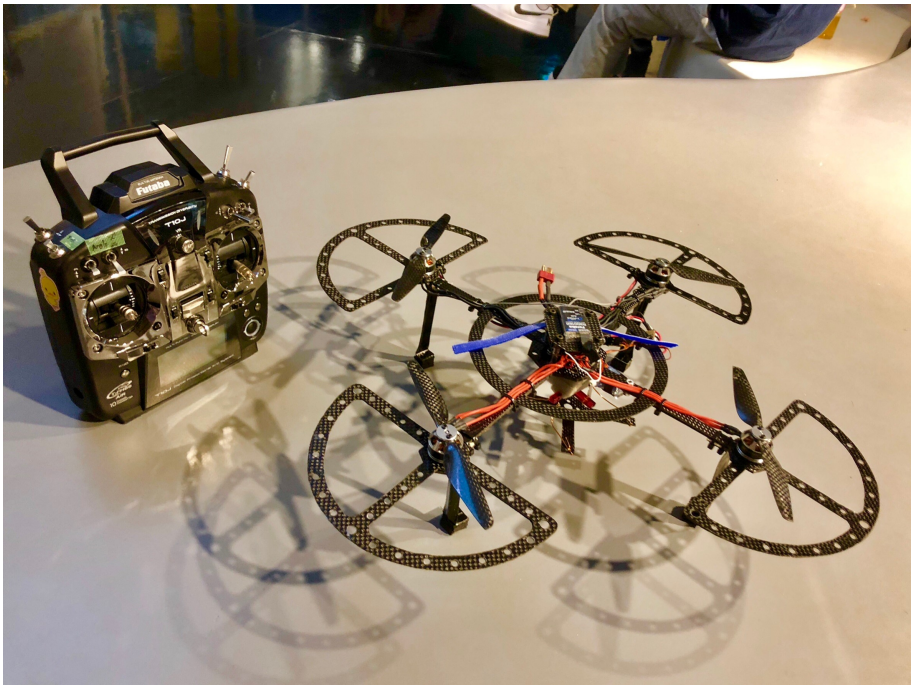

図 2.2　大会機

我々の機体の特徴を以下に示す．

1) カーボンシートを積層し軽量化かつ高い強度が得られるよう計った．カーボンを使用することで大会出場条件の350g以下になっている．

2) 前後の進行方向の区別を目視しやすくするためにピンク色のLEDランプを使用した．そのため操縦士から見やすくなり容易に操縦できるようになる．

3) 高所物資運搬のミッションで箱の中身を確認するのに超小型の動画撮影用のカメラを使用した．このカメラを使うことで，箱の中

身を確認した後に元の位置に戻る手間が無線じゃないため電波障害が起きることがない．

4) 操縦士がより機体を操縦できるように，機体の揺れを少なく制御を行った．

2.3 ミッション

マルチコプター部門のミッションは以下の5種類で，①のミッションは競技開始と同時に開始される．①のミッションが終了した後，②〜④のミッションの順番は問わない．⑤は最後に行うこと．

①「高所物資運搬」，②「大型物資運搬」，③「Rocking Wings」，④「8の字飛行」，⑤「自動離着陸」

予選と決勝におけるミッション予選では「高所物資運搬」「8の字飛行」の2つのみ挑戦をが認められる．決勝では全てのミッションへの挑戦が認められる．

2.4 大会結果

大会は11チーム出場し，我々の予選結果は840点を獲得し2位となり決勝へと駒を進めた．決勝は4チーム勝ち上がった．決勝で我々が挑戦したミッションは予選の「高所物資運搬」と「8の字飛行」に加えて，「大型物資運搬」に挑戦した．3つのミッションに挑戦した結果，1925点獲得し2位と500点近くの差をつけて優勝することができた．

2.5 大会の勝因

我々の大会の勝因を以下に示す．

1) フレームすべてをカーボンにしたことで軽量で高い強度が得られた．またフレームの厚みを変えたことで軽くなった．

2) 高所物資運搬のミッションにチキンラーメンを箱の中に入れるた

めのサーボ機構が安定していた．そのため機体に与える影響が少なくなった．

3) 戦略がよく高得点が出た．自動を含むミッションに挑戦しなかったため電波障害が起きることがなかった．
4) 制御の安定性があったため機体の揺れが少なく，操縦しやすくなっていた．
5) 操縦士の技量があり，緊張もせずに挑むことができた．

2.6 大会の反省点

我々の大会の反省点を以下に示す．

1) 大会ルールの確認不足．大会で使用できるバッテリーのセル数は2セルなのだが確認不足だったため3セルで出場しようとしてしまい，会場の近くのお店に急いで買い出しに行くこととなった．
2) 大会前の準備期間のスケジュールがチーム内で全員が揃うことが少なかった．

第 3 章

マルチコプターの運動方程式

3.1　マルチコプターの上下運動に関する微分方程式（運動方程式）

$$m\ddot{y} = T - mg \tag{3.1}$$

$$\dot{v} = \ddot{y} = \frac{T - mg}{m} \tag{3.2}$$

(3.3)

y:マルチコプターの高度 [m]

T:4 つのプロペラの推力 [N]

m:マルチコプターの質量 [kg]

v:上昇・下降速度 [m/s]

3.2　モーターの回転運動に関する微分方程式（一次遅れモデル）

$$\tau\,\dot{\omega} + \omega = ku \tag{3.4}$$

$$\dot{\omega} = \frac{ku - \omega}{\tau} \tag{3.5}$$

(3.6)

ω:モーターの角速度 [rad/s]

τ:モーターの時定数 [s]

$$k:\text{モーターのゲイン } [\frac{rad/s}{V}] \tag{3.7}$$

3.3 角速度とプロペラの推力との関係

$$T = \frac{G\ \omega^2}{1 - \frac{64}{9D^2}(\frac{v}{\omega})^2} \tag{3.8}$$

T:推力

G:推力係数

ω:プロペラ角速度

D:プロペラ直径

v:上昇・下降速度

プロペラ直径が 6 インチ D=0.1524m とすると，上式は

$$T = \frac{G\ \omega^2}{1 - 306(\frac{v}{\omega})^2} \tag{3.9}$$

上昇・下降速度 v は ω に対して非常に小さいので

$$T \fallingdotseq G\ \omega^2 \tag{3.10}$$

として良い．

シミュレーションのためにマルチコプターの質量 m=0.65kg，モーターに与える電圧 e と回転数 N の関係は，

$$N = K_V e[rpm] \tag{3.11}$$

我々の使用しているモーターは

$$K_V = 2600[rpm/V] \tag{3.12}$$

単位変換すると

$$\omega = K_V' e[rad/s] \tag{3.13}$$

$$K_V' = \frac{K_V \cdot 2\ \pi}{60} = \frac{\pi\ K_V}{30} \fallingdotseq 272.3 \tag{3.14}$$

3.9 式と 3.16 式からホバリングに必要な電圧を電池の半分の電圧 5V を考えＧを求める．※モーターとプロペラ 4 個の推力がマルチコプターの重量に等しくなる．

$$mg = 4T \tag{3.15}$$

ということになる．

第 4 章

高度制御

高度センサーで高度を計測して，制御する．センサーの性能を測定した．いかにその結果を示す．

今回使用した超音波センサーを図 4.1 に示す．

図 4.1　超音波センサー

4.1　超音波センサーの性能確認

以下に実験手順を示す．

1) 距離と出力の確認.アクリル板を少しずつずらして，距離と出力の関係を確認した．

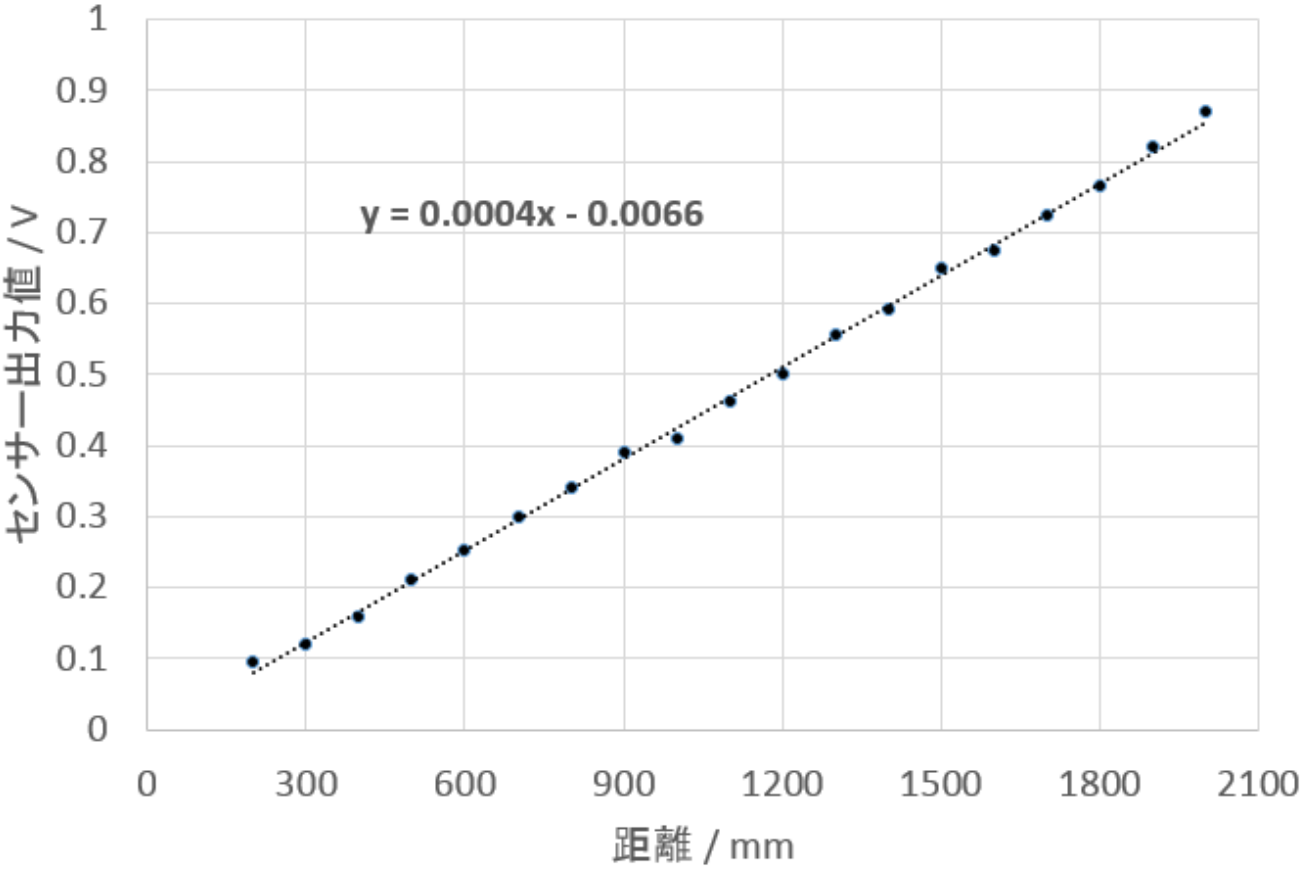

図 4.2　超音波センサーの距離と出力の関係

図 4.2 より，距離に応じてセンサーの出力値が比例していることがわかる．

2) 検知範囲の確認.

センサーの検知範囲を確認した．

図 4.3 より超音波センサーの音波は広範囲に広がって検知している．超音波センサーの性能を確認することができた．

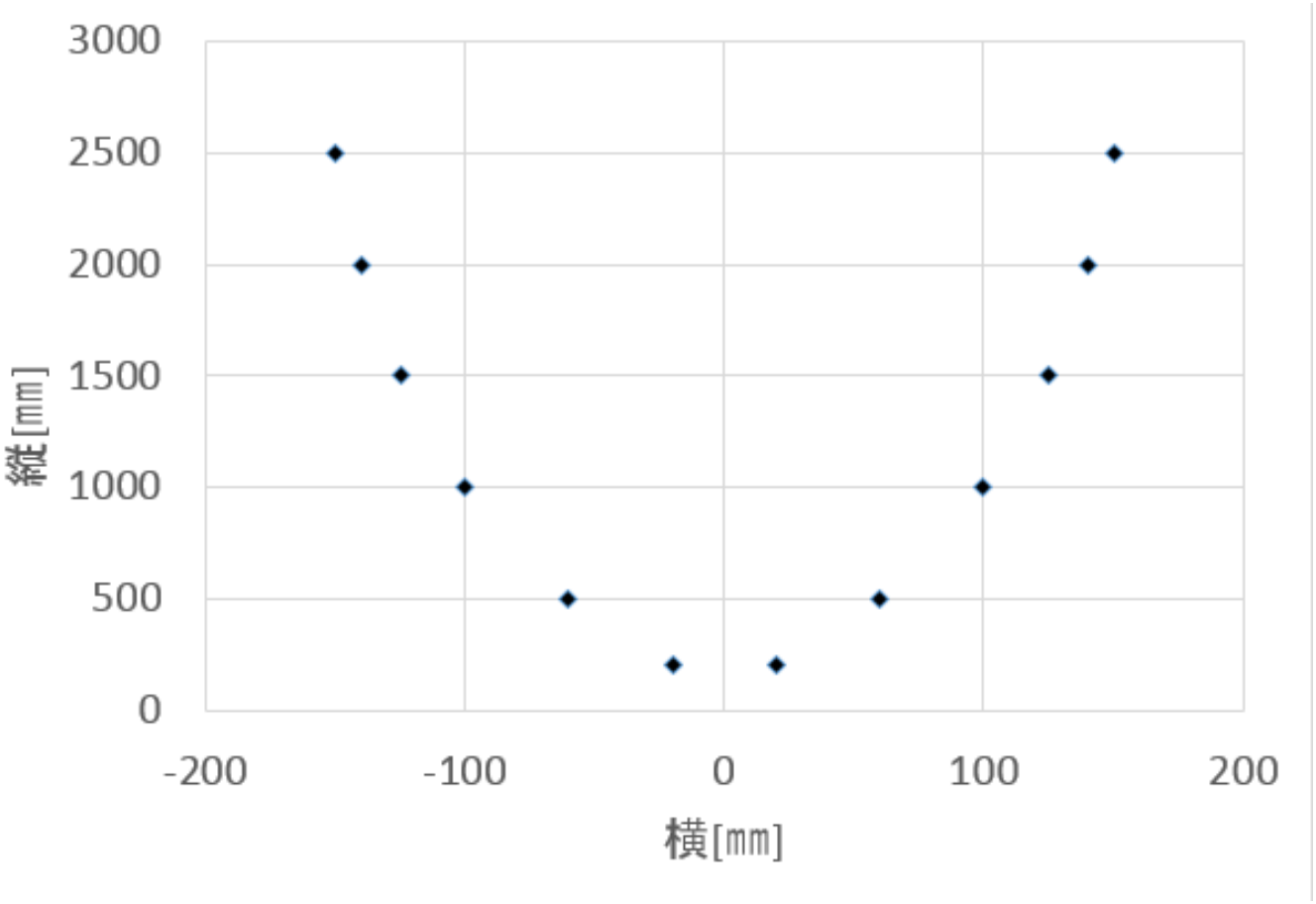

図 4.3　超音波センサーの検知範囲

第 5 章

高度制御のシミュレーション実験

制御性能をシュミレーションで確認する.

Python によって製作された制御シミュレーション CAD を用いて，PID 制御の比例ゲイン K_P, 積分ゲイン K_I, 微分ゲイン K_D に値を入れてシミュレーションを模擬実験する．

5.1 PID 制御

偏差（目標値と現在値の差）に比例して制御量が増える比例制御（P），偏差がある状態が長時間続くにつれ制御量が増えていく積分制御（I），偏差が急激な変化をするほど制御量が増える微分制御（D）を組み合わせた基本的な制御手法である.ブロック線図を図 5.1 に示す.

目標高度　h_{ref}

実際の高度　h

誤差　e

とすると，

$$e = h_{ref} - h \tag{5.1}$$

PID 制御とは，制御入力 u を次式で与える制御方式である．

$$u = K_P e + K_D e + K_I \int e dt + u_0 \tag{5.2}$$

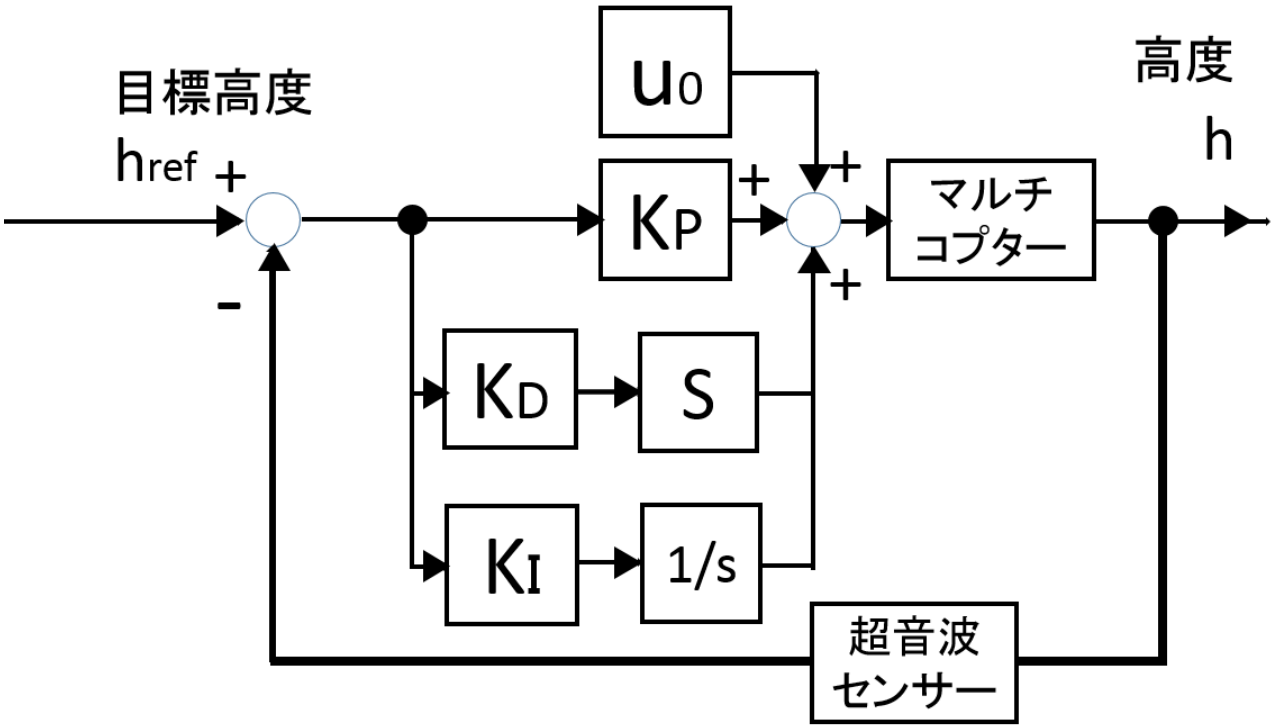

図 5.1　ブロック線図

e の微分$\dot{e}$と e の積分$\int edt$ については，実際のプログラムでは制御周期をΔt として　(5.3)

$$\dot{e} \fallingdotseq \frac{e_n - e_{(n-1)}}{\Delta t} \tag{5.4}$$

(5.5)

$$\int edt \fallingdotseq \sum e \tag{5.6}$$

ここで K_P :比例ゲイン

K_D :微分ゲイン

K_I :積分ゲイン

1)　比例制御のみの結果を図 5.2 に示す．比例制御だけでは発散して，高度制御ができないことがわかる．

2) 比例制御と微分制御の実験を行った結果を図 5.3 に示す．微分制御を加えたことで，安定するが目標値に一致しないことがわかる．

3) 積分制御を使った結果を図 5.4 に示す．積分制御を加えたことで，目標値に一致していることがわかる．

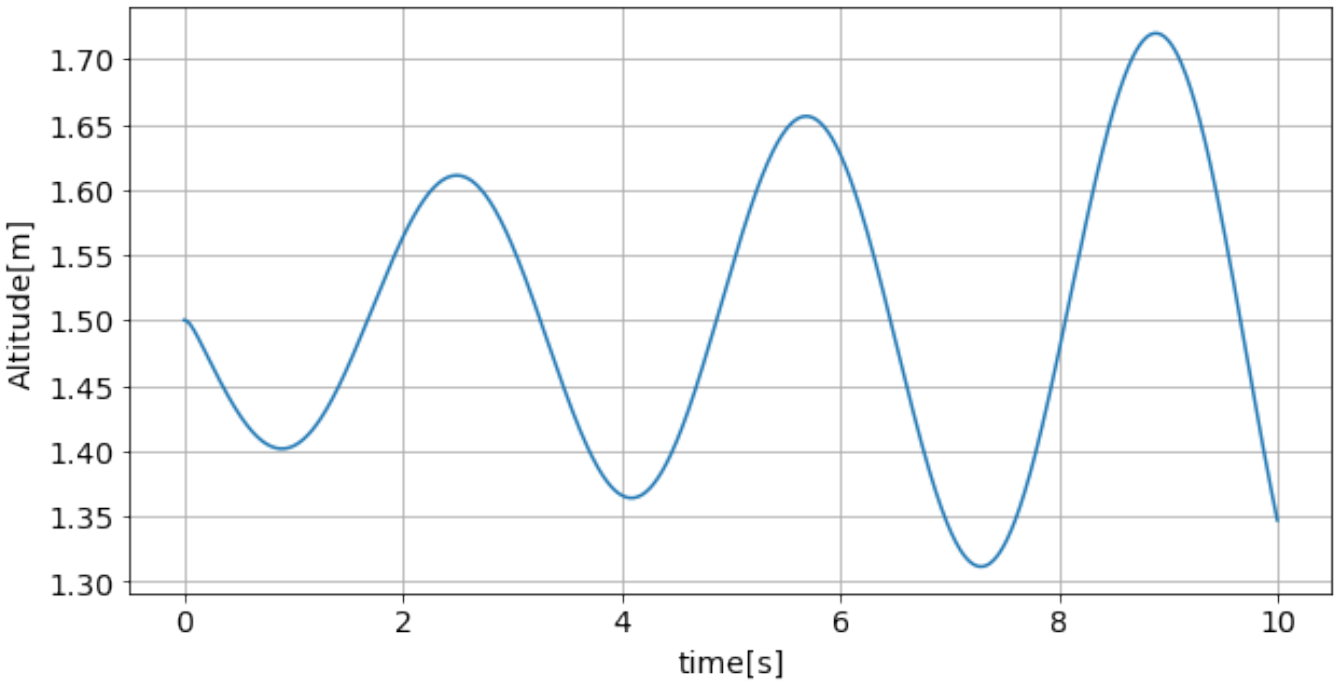

図 5.2　比例制御のみの高度結果

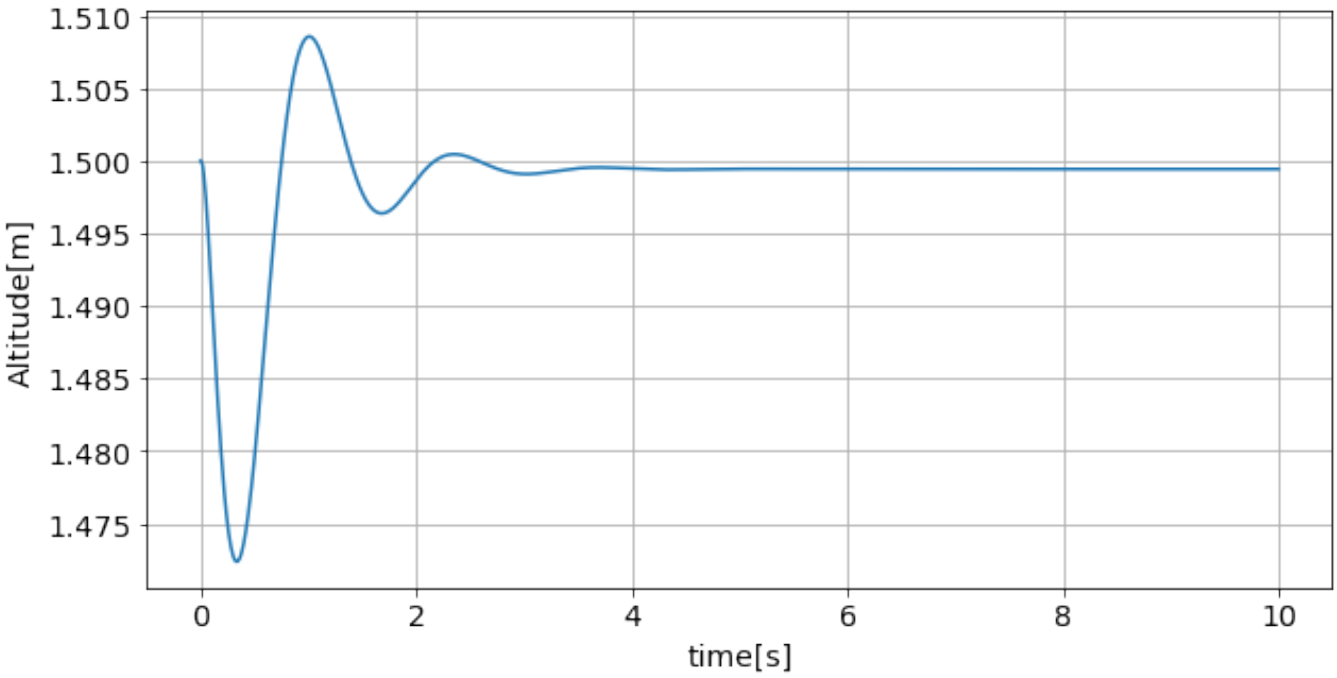

図 5.3　比例制御，微分制御の高度結果

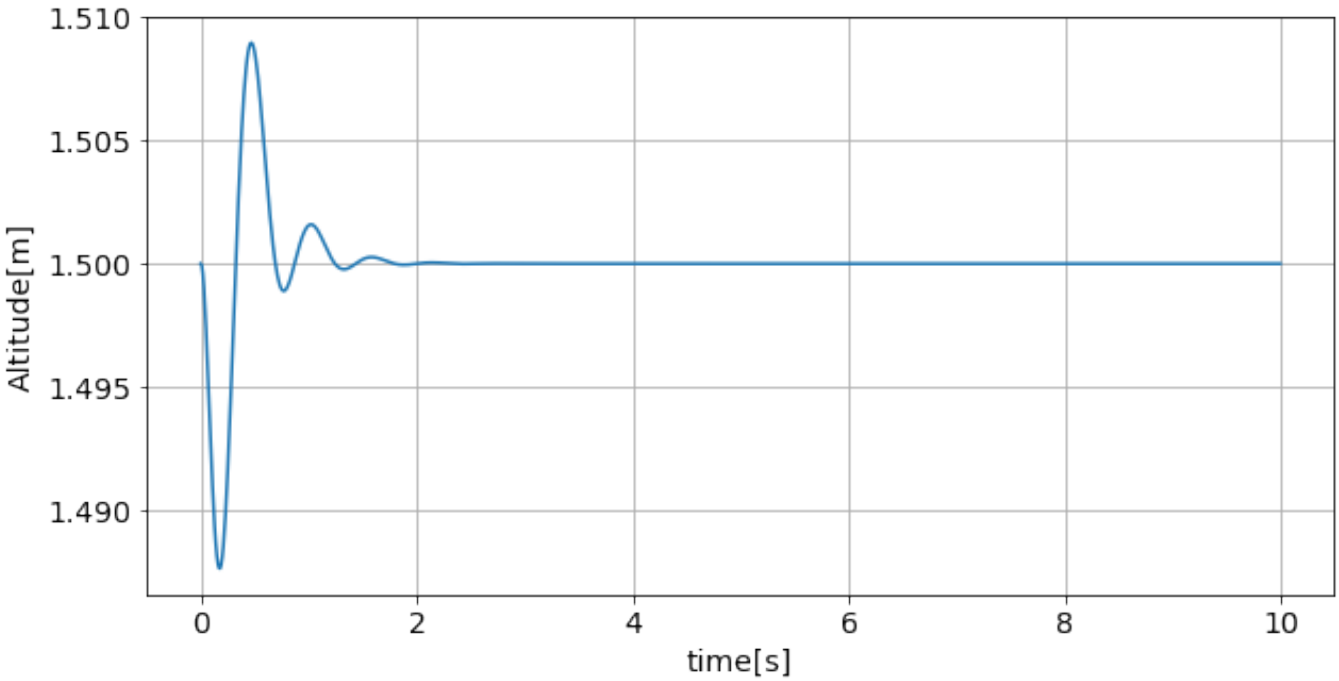

図 5.4　比例制御，微分制御，積分制御の高度結果

第 6 章

考察

図 5.2 より比例制御だけではマルチコプターが上下に発散して，高度制御ができないことがわかる．次に図 5.3 より比例制御に微分制御を加えたことで，安定するが目標値に一致しないことがわかる．最後に図 5.4 より比例制御と微分制御に積分制御を加えたことで，目標値に一致していることがわかる．これらの図より，初めに落ちてしまっている．これは 1 メモリ 0.01m なので実際には約 10mm しか落ちないので，あまり影響がないと考える．

第 7 章

おわりに

7.1　結論

高度制御について超音波センサーの性能確認の実験とPID制御を利用したシミュレーション実験を行った．PID制御をすることにより高度制御を成功した．

7.2　今後の課題

制御し始めに少しだけ落ちてしまっているため更に研究が必要であると考えられる．

また，PID制御を実機に加えて実験をする必要があると考える．

参考文献

[1] 和 栗 雄 大 郎 , 模 型 飛 行 機 の 科 学 , 養 賢 堂 , 2005/07/01

謝辞

本論文作成にあたり研究の考え方, 方法のまとめ方など長期にわたって熱意のあるご指導, ご鞭撻していただいた, 伊藤恒平教授に厚く御礼申し上げます.

特に制御においても論文の書き方においても論文を何度も読んでいただき, 指導していただいた伊藤恒平教授に大変ご苦労をかけてしまいましたことにも心よりお詫び申し上げます.

その他, 助けていただいた多くの皆様に心から感謝しております. ありがとうございました.

付録 A

プログラム

saucecode A.1　PID 制御プログラム

```
matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
from scipy.integrate import odeint

olderr=0.0
oldt=0.0
oldtt=0.0
oldu=0.0
esum = 0.0
T=[]
U=[]


def control(output, ref, t):制御
    #

    global olderr,oldt,esum

    Kp = 5 比例ゲイン#
    Kd = 1. 微分ゲイン#
    Ki = 0. 積分ゲイン#

    u0 = 5.0 ホバリングに必要な電圧#[V]

    err = ref − output
    #diff error
    if t−oldt==0.0:
        derr =0.0
    else:
        derr= (err − olderr)/(t−oldt)
    #print(err,olderr,t,oldt, derr)
    olderr=err
    oldt=t

    #integrate error
    esum+=err
```

```
38
39      up = Kp ∗ err
40      ud = Kd ∗ derr
41      ui = Ki ∗ esum
42
43      u = up + ud + ui + u0
44
45      if u<0.0:
46          u=0.0
47      elif u>11.1:
48          u=11.1
49      #print(u)
50
51      T.append(t)
52      U.append(u)
53      return u
54
55  def sensor(height):
56      return height;
57
58  def motor(omega,u):
59
60      if(u<0.0):
61          u=0.0モータゲイン
62
63      #
64      Kv=272.3モータ時定数
65
66      #
67      tau =0.05
68
69      omegadot = (−omega + Kv∗u)/tau
70
71      if (omega<0.0 or omega==0.0):
72          if omegadot<0.0:
73              omegadot = 0.0
74      #print(u,omega,omegadot)
75      return omegadot
76
77  def propeller(omega):
78      Ct = 8.59e−7 推力係数#
79      Thrust = 4.0 ∗ Ct ∗ omega∗∗2
80      #print(Thrust)
81      return Thrust
82
83  def drone(x, t, ref, tmp):
84      global oldtt,oldu
85
86      mass = 0.65 マルチコプタの重量#kg
87      g = 9.81
88      #print(t)
89      v=x[0]
90      height=x[1]
91      omega=x[2]
```

```
92
93
94      vdot=(propeller(omega) − mass ∗ g)/mass
95      if(height<0.0 or height==0.0):
96          if vdot<0.0:
97              vdot=0.0
98
99      heightdot = v
100
101     if t−oldtt>0.01:
102         u=control(sensor(height),ref,t)
103         oldtt=t
104         oldu=u
105         #print(t,u)
106     else:
107         u=oldu
108
109     omegadot = motor(omega, u)
110
111     return [vdot, heightdot, omegadot]
112
113 if __name__=='__main__':
114     ref=1.5シミュレーション開始時のドローンの初期値
115
116     #
117     v0=0.0 初期速度#
118     height0=ref 初期高度#
119     omega0=1362 初期モータ角速度#
120
121     x0=[v0 , height0, omega0]
122
123     t=np.arange(0.0, 10.0, 0.01)
124     x = odeint(drone, x0, t, args=(ref,0),hmax=0.001)
125
126     plt.figure(figsize=(10,5))
127     plt.rcParams['font.size']=14
128     plt.plot(t,x[:,1],label='Altutude')
129     #plt.title('Altitude')
130     plt.xlabel('time[s]')
131     plt.ylabel('Altitude[m]')
132     plt.grid()
133     plt.show()
134     plt.plot(t,x[:,2],label='Omega')
135     plt.title('Omega')
136     plt.grid()
137     plt.show()
138     plt.plot(t,x[:,0],label='Velocity')
139     plt.title('Velocity')
140     plt.grid()
141     plt.show()
142     plt.plot(T,U,label='Control')
143     plt.title('Control')
144     plt.grid()
145     plt.show()
```

```
146     plt.plot(t,x[:,0]/x[:,2],label='V/omega')
147     plt.title('V/omega')
148     plt.grid()
149     plt.show()
```