令和２年度

卒業研究論文

アクティブ乱流生成装置の回転翼の回転速度による風速応答性の評価に関する研究

1710950024　鵜久森　泰弘

近畿大学工学部機械工学科

流体エネルギー研究室

目次

第 1 章　緒言 3

1.1 自然風再現 4

1.2 研究目的 4

第 2 章　実験方法 6

2.1 乱流生成装置の設計 6

2.2 乱流生成装置の制御システムの構築 11

2.3　風洞実験概要 12

2.3.1　計測内容 12

2.3.2 熱線流速計の設定 13

2.3.3 熱線流速計の校正 15

第 3 章　結果と考察 18

3.1 回転翼の回転速度の変化による風速応答の変化について 18

第 4 章　結言 23

引用文献 24

謝辞 25

付録 26

第1章　　緒言

風は，私たちの生活の中で身近な運動のひとつである．大橋や高層ビルのような大型建造物であれば，ビル風や台風等の強風による横揺れや共振性といった影響を検証する必要がある．車や飛行機といった移動する物体風洞装置での実験は，横風への耐久性や車体形状による抵抗度合いの検証が必要とされる．風力に対する安定性等の空力特性を知ることは，製品や構造物の安全性の保障と品質向上のために重要である．[1,2] 風洞実験では模型を扱ったシミュレーション実験が行えるため，空力特性の検証ができ，コストを低減も併せて可能にする．また，理論値の計算との比較をすることで，データとしての信憑性を高めることができる．

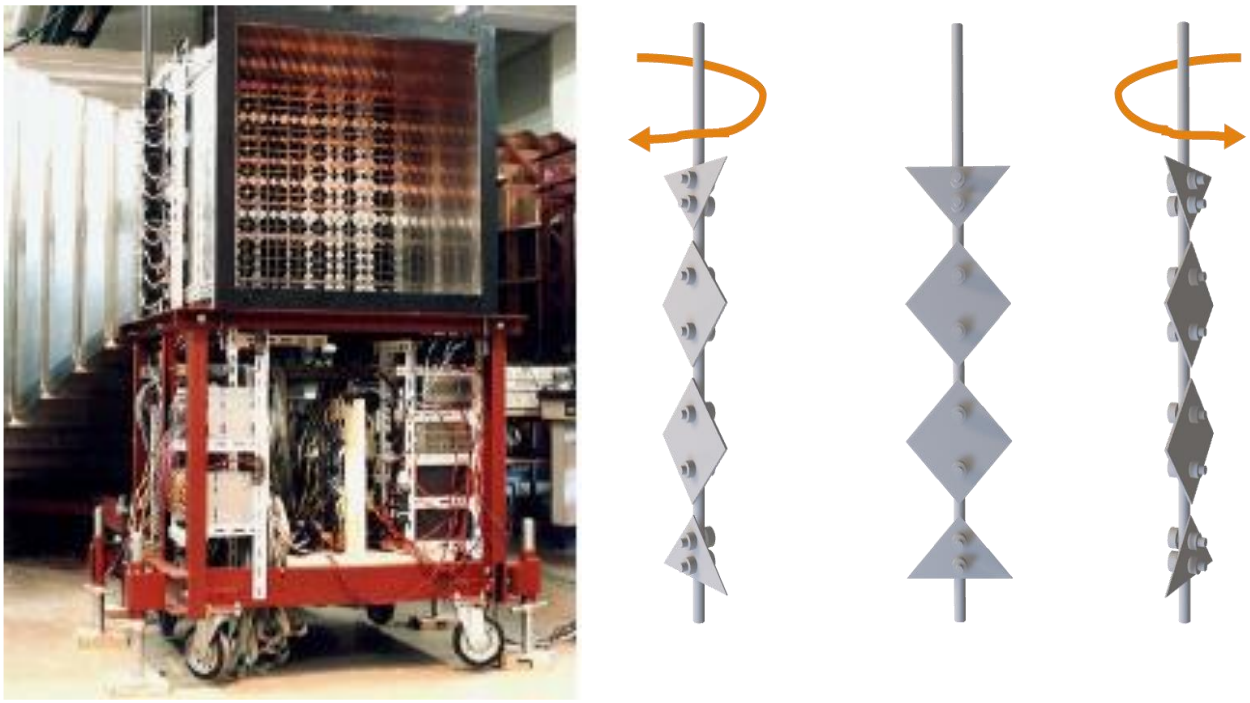
1940 年 11 月 7 日に，アメリカシアトル郊外で起きたタコマ・ナローズ橋の崩落事故について以下に示す．中央支間長 853m の当時世界第 3 位の大橋であったタコマ橋は，動的な風荷重の渦励振と本橋自身のねじれ振動の共振により崩落した．図 1.1 はタコマ橋が崩落した時の写真である．この崩落事故は，動的空力特性の安定性を考慮することの重要性を示した事故として有名である．

図 1.1　タコマ橋が崩落した様子[3.4]

1.1 自然風再現

高層ビルや大橋のような，風の影響を受けやすい構造物に対し，自然風と類似した風を実験的に生成し，耐久性や空力特性を評価することが工学的に重要である．同様に，風力発電機のような風のエネルギーを利用する流体機械の開発時に自然風と似た乱流風で風車の評価が可能になれば，より正確な性能評価につながる．

乱流場を形成する手法は乱流格子，噴流格子等があげられるが，乱流場の形成においては不十分とされている．そこで，現在自然風を実験環境下で再現できるとされる手法の一つとして，多数の回転翼から構成された乱流風洞装置があげられる．流れに動的な制御を加えることにより，乱れの度合いを大きく調整できることが期待されている．図 1.2(a)は全断面($0.7 \times 0.7m^2$)，回転軸が縦横各 20 本(各軸に撹拌翼が取り付けられている．)取り付けられた，乱流発生装置の図である．各回転軸はステッピングモータで個別に駆動されている．図 1.2(b)では，今回我々が(a)の乱流発生装置を参考に製作した回転軸の回転モデルを示す．回転軸をランダムに回転させることにより，任意の流速分布や乱流特性を持つ乱流場を形成することができる．



(a)大気乱流風洞及びせん断乱流発生装置　　(b)回転軸の回転モデル

図 1.2　乱流発生装置[5.6]

1.2 研究目的

1.1 で述べたような乱流生成装置が乱流の研究では広く使用されているが，その風速応答性の評価に関する報告は著者らが調べた限り見当たらない．本研究では，小型風洞用に乱流生成装置を設計・製作し，回転翼の回転速度による風洞測定部での風速の応答特性の評価を行う．この風速応答の評価は，自然風を再現するための乱流生成装置の制御アルゴ

リズムの開発に繋がると期待される.

1) 乱流生成装置の設計と製作
2) 回転翼制御用プログラムの構築
3) 回転翼の回転速度が及ぼす風速応答性の評価

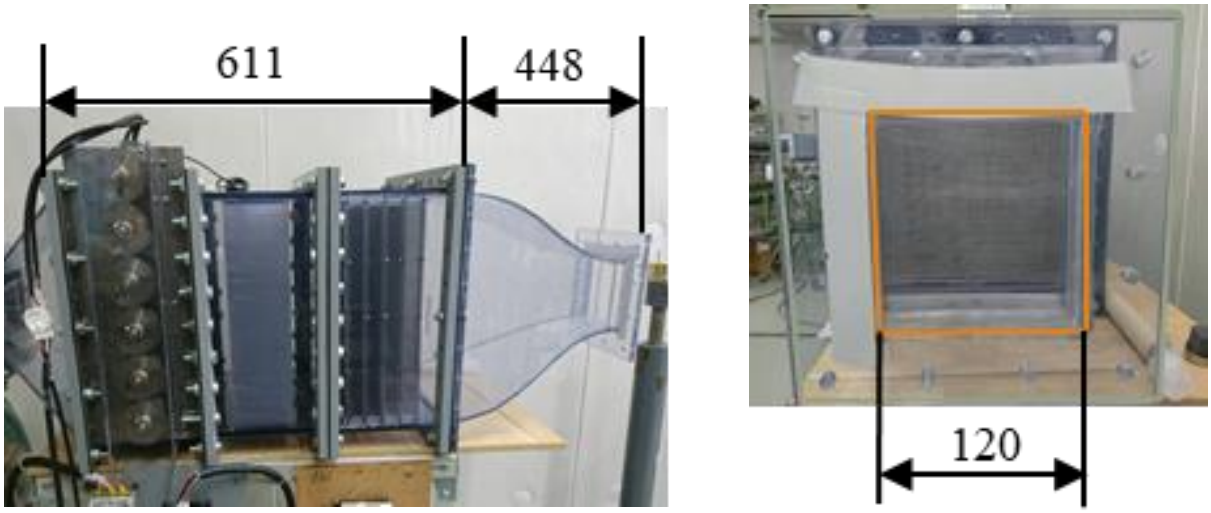
本論文は 5 つの構成から書かれている.

1. 第 2 章前半：実験装置の紹介及び実験方法の説明
2. 第 2 章後半：校正の説明
3. 第 3 章前半：実験結果の説明
4. 第 3 章後半：考察
5. 第 4 章：結言

第 2 章　実験方法

2.1 乱流生成装置の設計

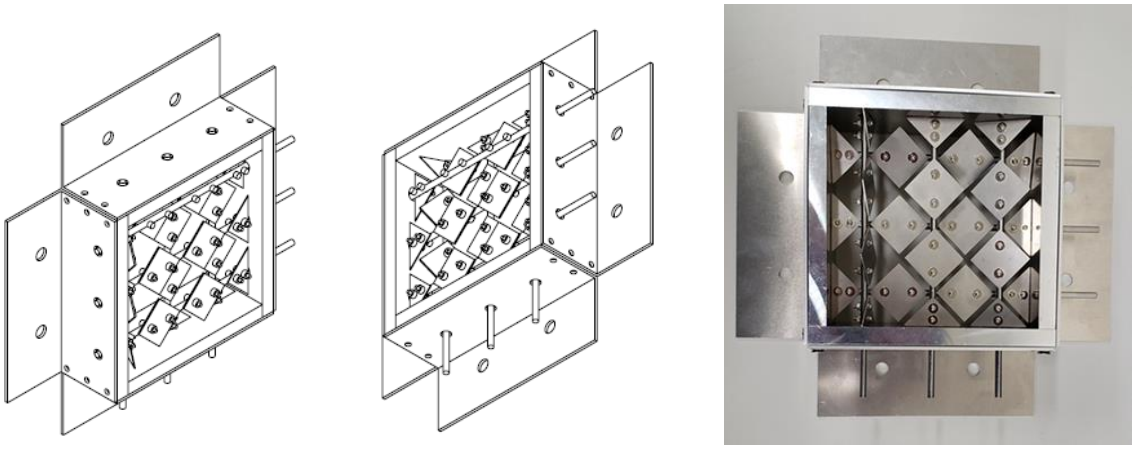
図 2.1 に示す風洞装置のノズル出口に合うよう，図 2.2 の乱流生成装置を設計した．設計・製作した乱流生成装置には縦横各 3 本ずつ，風速変動させるための軸と正方形の平板翼が備えられている(以下回転翼と呼ぶ)．この回転翼を回転させることにより，風速変動（乱流）を生じさせる．



(a) 側面図　　**(b)** ノズル出口

図 2.1　風洞装置　(単位 ： ㎜)．

今回設計した乱流生成装置（駆動装置を除く）を以下に図 2.2～2.9 として示す。
設計した部品の構成としては、回転翼（全翼・半翼　各 12 枚)、回転棒 6 本、枠（上下・左右別)、固定用アングル（上下・左右別）である。



（a）モデル　　（ｂ）実物

図 2.1　　自然風再現用動的乱流誘起装置　（全体図）

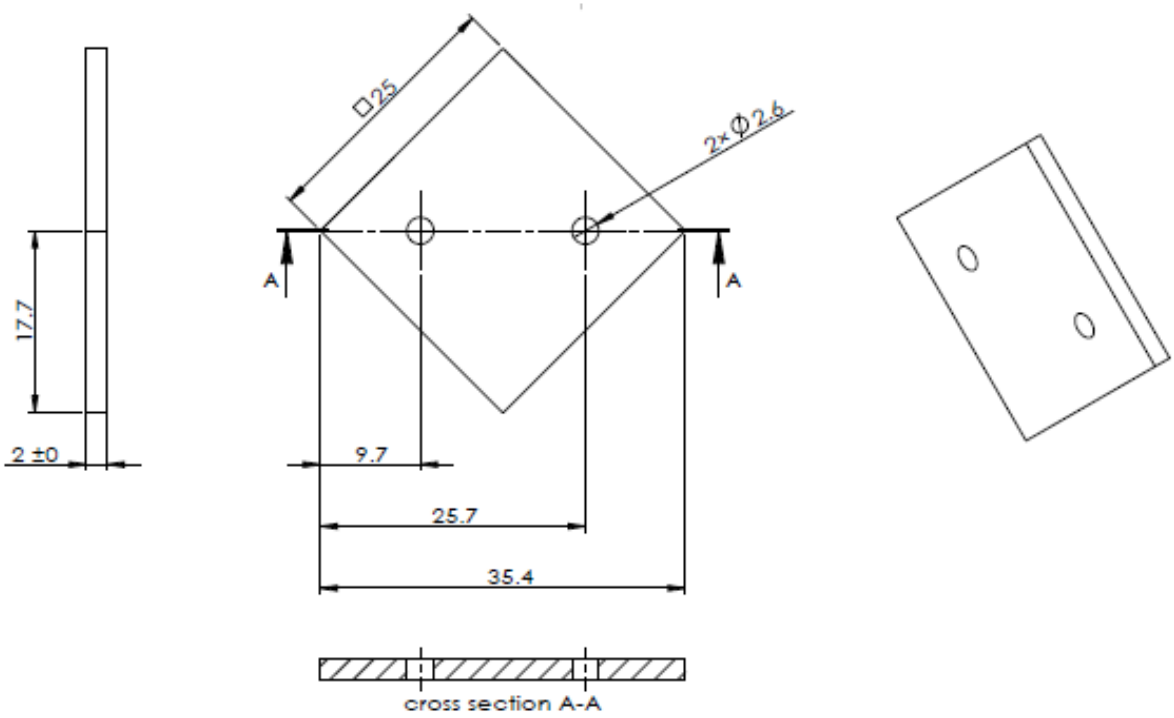


図 2.2　回転翼（全翼）

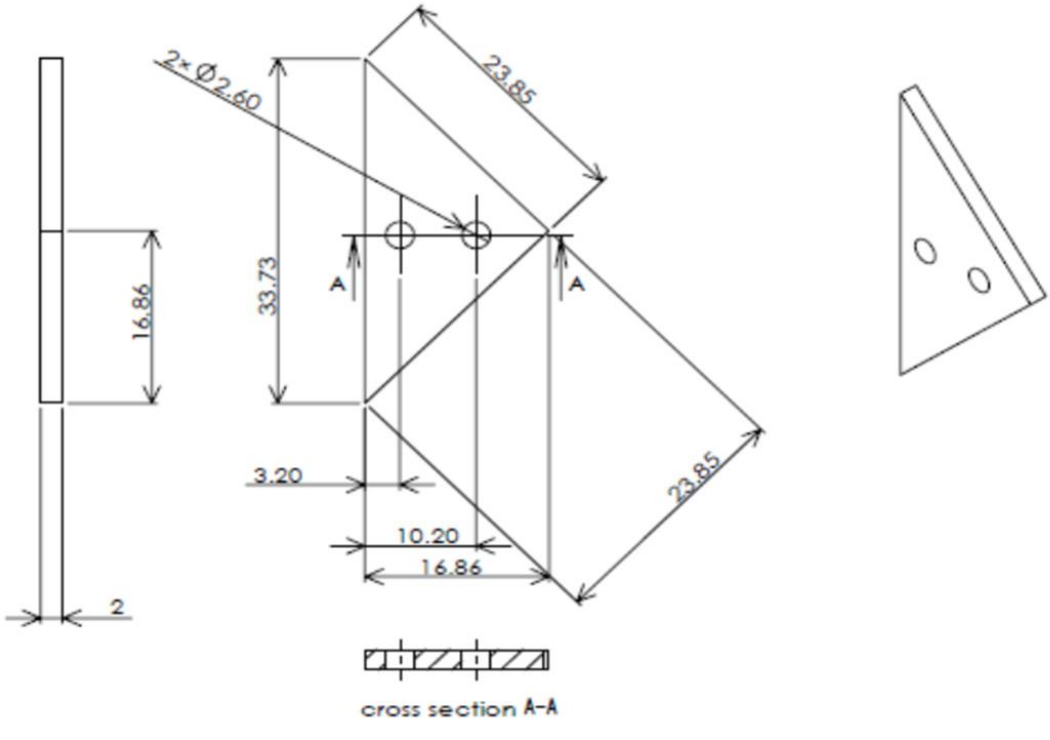


図 2.3　回転翼（半翼）

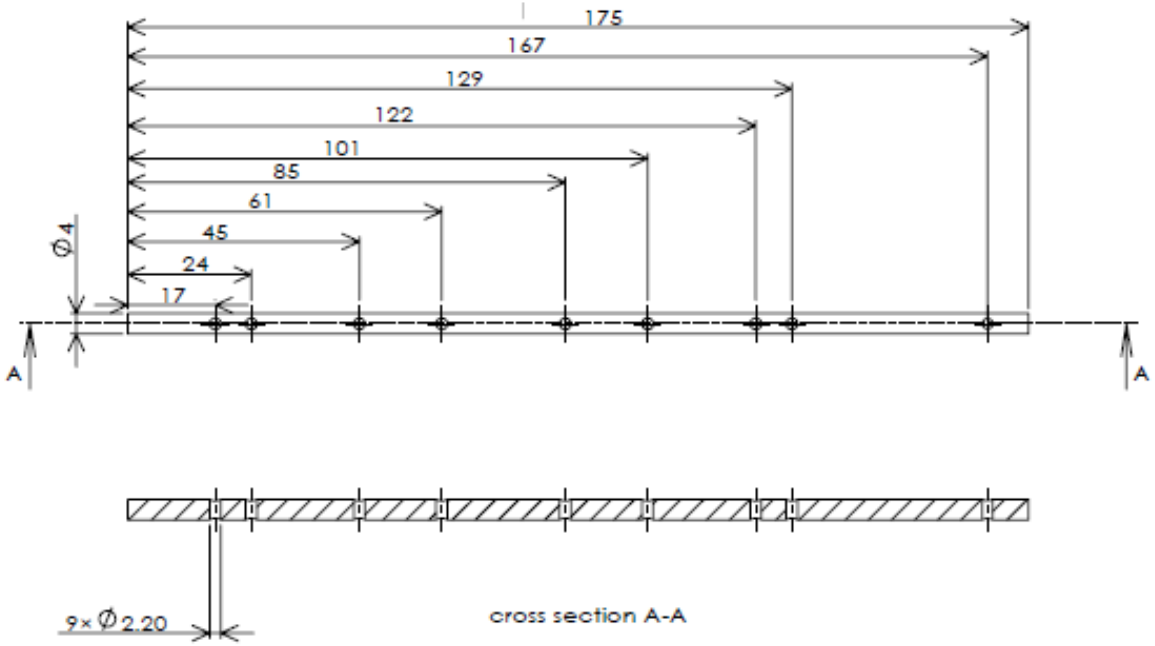


図 2.4　回転棒

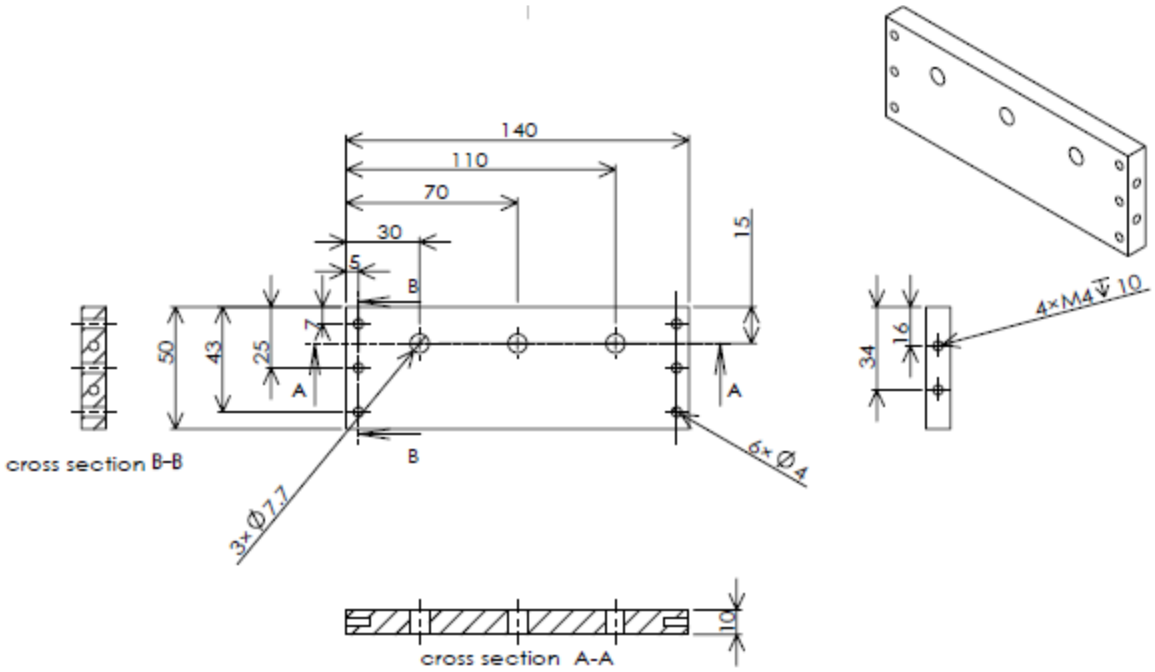


図 2.5　外枠　（上下）

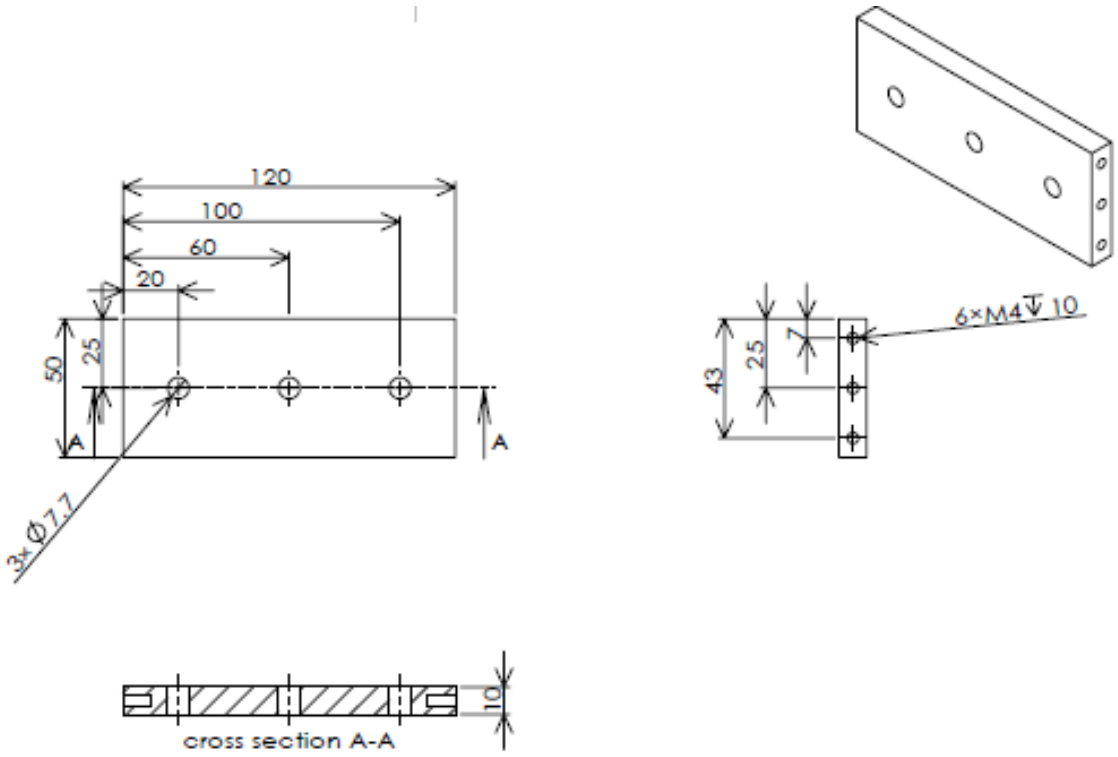


図 2.6　外枠　（左右）

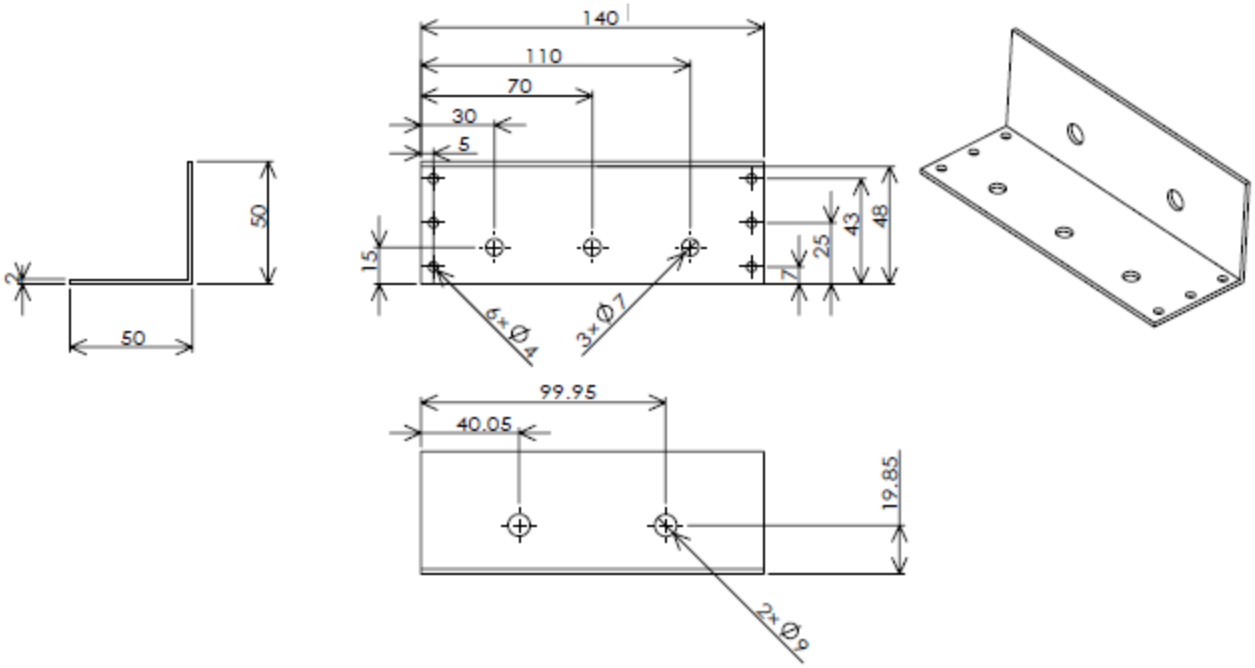


図 2.7　固定用アングル（上下）

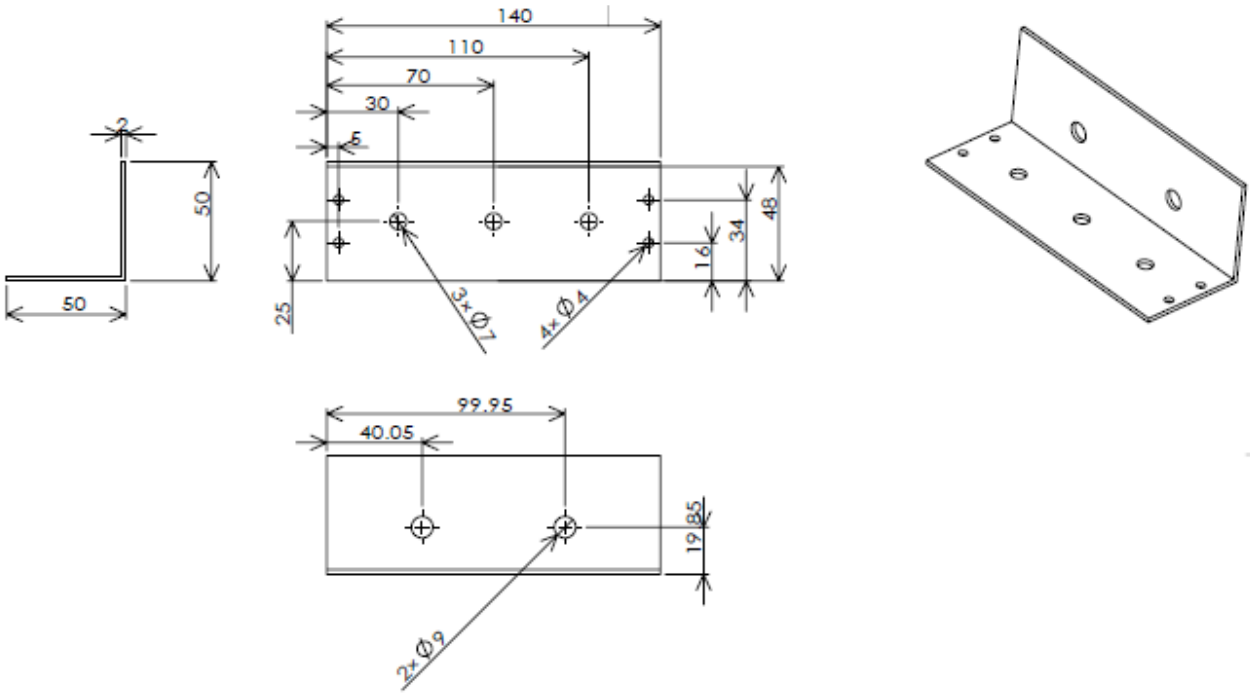


図 2.8　固定用アングル（上下）

また、駆動部分を合わせた乱流生成装置を以下に示す。駆動部分に関しては駆動部分に関して，スライダークランク機構を参考に製作を行った．[7]

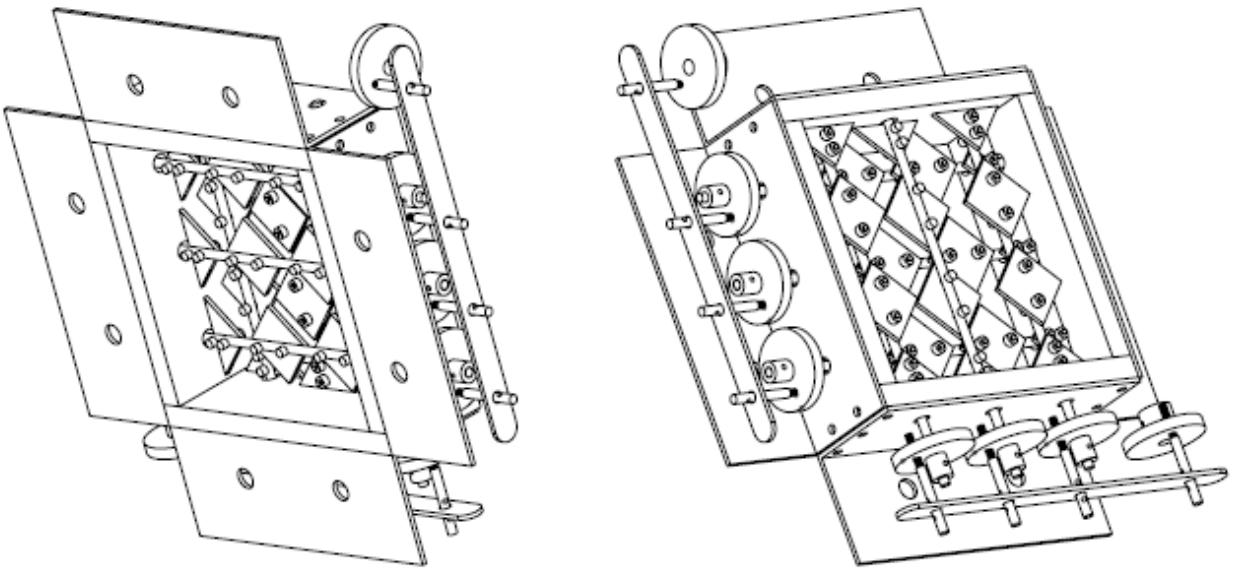
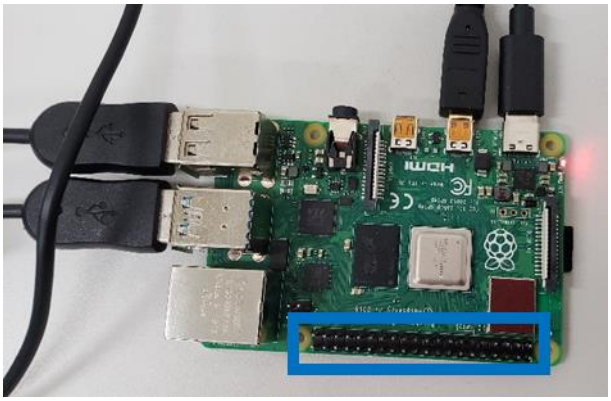


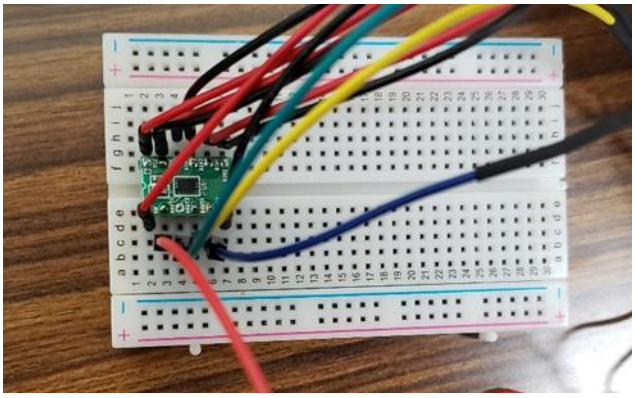
図 2.9　自然風再現用動的乱流誘起装置　（全体図・駆動部分あり）

2.2 乱流生成装置の制御システムの構築

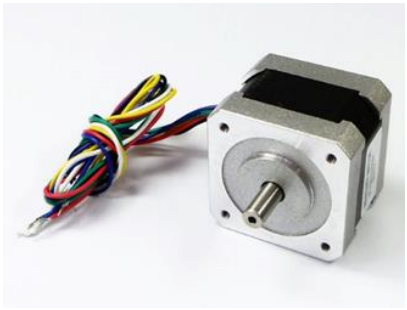
今回はプログラム制御に Raspberry Pi4 と python を用いた．使用した電子部品は図 2.10 のとおりである．



(a)Raspberry Pi 4（青枠：GPIOpin）



(b)ブレッドボード



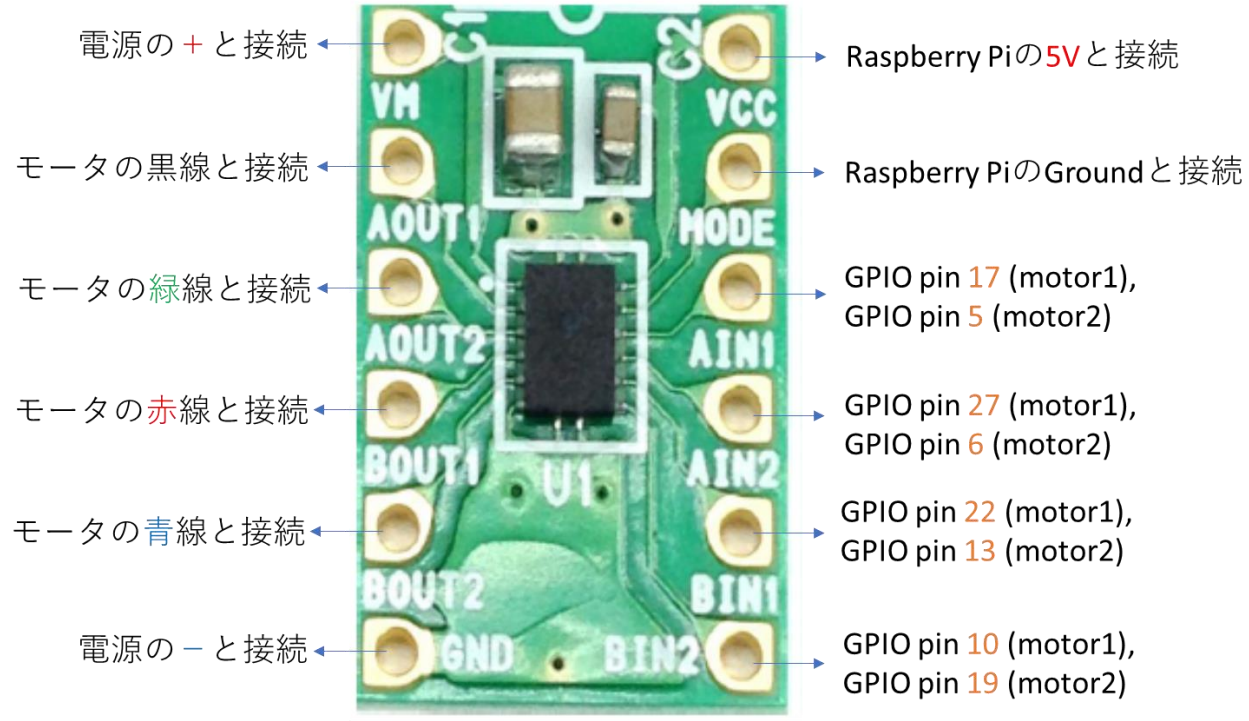
(c)バイポーラ
ステッピングモーター
SM-42BYG011

(d)バイポーラ
ステッピングモーター
ST-57BYG076

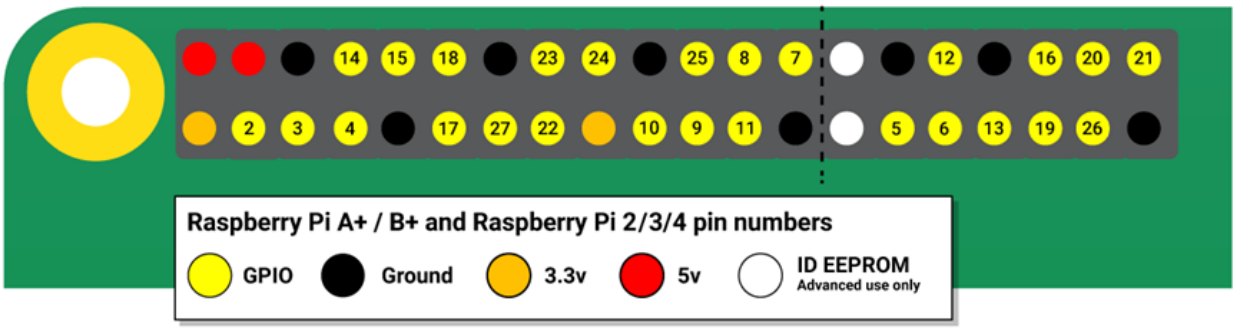
(e)使用ステッピング & DC モータドライバモジュール
DRV8835

図 2.10　制御機構構成用電子部品

図 2.11 のように GPIOpin とブレッドボードを接続し，ドライバーを介してステッピングモータと接続した．実験に用いたモータは大型のもので，小型のモータは試験運転用に用いた．モータへの接続は黒線と緑線で一組，赤線と青線で一組として対応しているので，ドライバーの A を黒線と緑線，B を赤線と青線と接続した．



(a) ドライバーの接続先



(b) GPIOpin の名称[8]

図 2.11　接続設定

プログラムに関しては，motor_1，motor_2，motor_1・motor_2 用の関数プログラムの 3 つのプログラムに分けた．プログラミングコードを，論文の最後に付録として示す．

2.3　風洞実験概要

2.3.1　計測内容

本研究では，回転翼を 4 種類の回転速度（1，5，20 および 30rpm）で運転し，回転翼の回転角度の範囲を複数変えて風速応答の計測を行った．回転角度を時間的に周期変化させたときの風速応答性を評価するための実験条件を表 1 に示す．なお，回転翼が水平な状態を，回転翼の角度 0° とする．

表 2.1 実験条件

Rotation angle(°)	Rotation speed (rpm)			
−10°to 10°	1	5	20	
−30°to 30°	1	5	20	
−45°to 45°	1	5	20	30
−60°to 60°	1	5	20	

風速計測は，熱線流速計（KANOMAX 社製 0248R-T5）を用いて風洞出口から120(mm) 下流位置で行われた．計測サンプリング周波数は500(Hz)で，取込時間は20(s)である．全ての実験は風速6.0m/sで行った．また，風速の計測開始時刻から約 5 秒後に乱流生成装置を運転させた．

2.3.2 熱線流速計の設定

熱線流速計を図 2.12 のようにセッティングし，計測を行った．熱線のセッティング手順を，図 2.13 を参考に以下に示す。

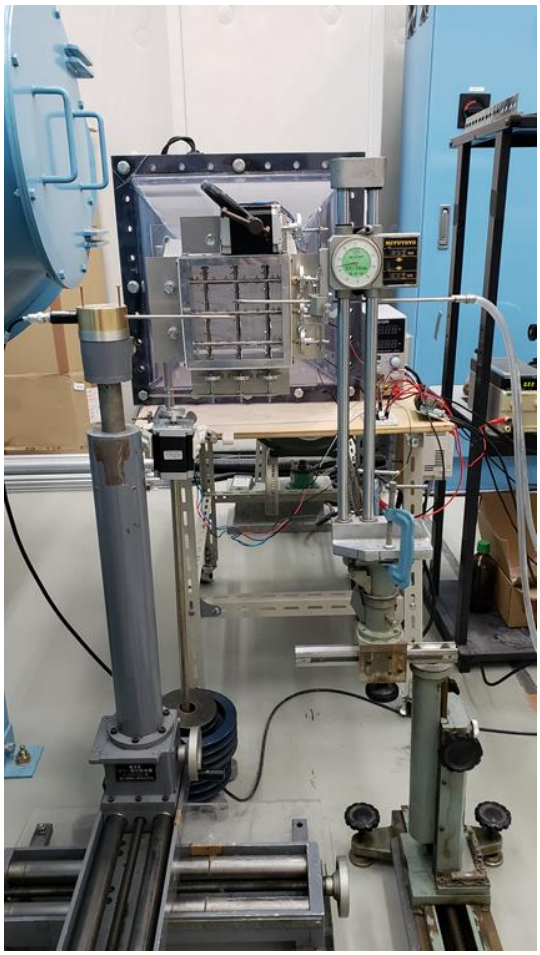


図 2.12 実験装置

図 2.13 電源（電源ケースモニター　Model1008）　※　赤：A　黒：B　緑：C

・熱線流速計の接続設定

1. 熱線流速計サポート部分にショートプローブを取り付ける.
2. A の power を ON にし，INPUT を 1 にセットする.
3. C にある黒いつまみを回して R MEAS に合わせる.
4. C の REF SET を，A の液晶が 2.00V になるまで回す. この時，表示が安定するまで待つ.
5. 表示が安定したら,C の TEMP-R MEAS レバーを R MEAS に合わせたまま,PROBE RESISTANCE にある ZERO を，液晶の表示が 2.00V になるまで回す.
6. ショートプローブを熱線プローブに付け替える.（計測機で接続を確認する。）
7. TEMP-R MEAS レバーを R MEAS に合わせた時，液晶に表示される値が，R MEAS に合わせる前の値と変化しないように PROBE RESISTANCE の値を変える。中心の黒い 2 つのつまみを使う。
8. PROBE RESISTANCE の値を，合わせた値の 1.5 倍の値に変える.
9. REF SET を回し，液晶の値を 15V に合わせる。合わせた後，表示が安定するまで待つ.
10. INPUT を 2 に合わせる.

2.3.3 熱線流速計の校正

校正に関して，ピトー管による U 字示差圧力計を用いた．液面(エタノール)の計測には図 2.13(a)の顕微鏡(カートン光学 NRM-D-2XZ)を用いた．風速の調整はインバータ(富士電機製 FVR-C11 FVRO 4C11S-2 3PH 0.95KVA 200-230V)と送風機(西村電機製　汎用送風機 NK-5)を用いて行った．室温と大気圧は(b)の LAD WEATHER SENSOR MASTER IV(クラージュ社製)，エタノールの温度は赤液棒状温度計をエタノールにつけて計測した．

(a)計測用顕微鏡　　(b)気温・大気圧計　　(c)エタノール

図 2.13　校正用使用機器

測定した項目は表 2.2 のとおりである．ピトー管はノズル吹き出し口中心(端から 60 ㎜)，距離 120 ㎜の位置で計測した．計測数値をもとに表 2.3 を作成し，表 2.4 のように流速を算出した．

表 2.3 で求めた値の算出式は以下のとおりである．

気圧に関して，単位を水銀柱ミリメートルに直した．1013[hPa] = 1[atm] = 760[Hgmm]より，気圧[Hgmm]をP_{Hg}とすると，式は 2.1 のようになる．

$$P_{Hg} = 1003.6 * \frac{760}{1013} = 752.9 \tag{2.1}$$

よって、大気圧は752.9[Hgmm]として換算できた。

空気密度は，乾燥状態，0[°]，1[atm] = 760[Hgmm]の時，1.293[kg/m^3]である．気体の状態方程式を考えれば，圧力P[Pa]，密度ρ[kg/m^3]，温度t[K]，気体乗数Rとしたとき，

$$\mathrm{P} = \rho \mathrm{RT} \tag{2.2}$$

式 2.1 から，密度は温度と圧力に依存するので，0℃の時の温度をt_0，密度をρ_0 ，圧力をP_0 ，X℃の時の温度をt，密度をρ，圧力をPとすれば，求める密度ρは式 2.3 のようになる．

$$\frac{P_0}{t_0} \ : \ \frac{P}{t} = \rho_0 : \quad \rho \tag{2.3}$$

$$\rho = \rho_0 \times \frac{P t_0}{t P_0} \tag{2.4}$$

よって、X=19.4° の時，式 2.3 に表 2.2，2.3 の数値を当てはめれば，

$$\rho = 1.293 \times \frac{752.9 \times (273.15 + 0)}{(273.15 + 19.4) \times 760} = 1.195 \quad (2.5)$$

空気密度は$1.2[\mathrm{kg}/m^3]$となる.

エタノールの場合も同様に計算を行った.国際アルコール表から,エタノールの温度15°,体積濃度98%の時の密度は$793.51[\mathrm{kg}/m^3]$であった.

次に，室温$T = 273.13 + 19.4[\mathrm{K}]$として，空気粘度$\mu[\mathrm{Pa \cdot s}]$の計算式を示す.

$$\mu = \frac{1.4592 \times 10^{-6} \times T^{\frac{3}{2}}}{109.10 + T} = 1.818 \quad (2.6)$$

空気粘土は$1.818[\mathrm{Pa \cdot s}]$となる.

表 2.2　実験環境　(2021/1/11)

空気温度(°)	19.4
エタノール温度(°)	20.0
気圧(hPa)	1003.6
マノメータ初期水位(mm)	39

表 2.3　実験環境　(2021/1/11)

空気密度(kg/m^3)	1.20
空気粘度（Pa・s)	1.81894E-05
気圧(mmHg)	752.9
エタノール密度(kg/m^3)	789.32

次に、表 4 の計算式を示す.

圧力差の算出に関して，圧力差を$P_{12}[\mathrm{Pa}]$，マノメータ初期水位との差を$h_{12}(\mathrm{mm})$，エタノール密度と空気密度の差をρ_{12}，重力加速度$g = 9.81[\mathrm{m}/s^2]$とすると，式 2.7 のように表せる.

$$P_{12} = \rho_{12} g h_{12} \quad (2.7)$$

表の値より，$10[\mathrm{Hz}]$の時の圧力差は，$8.891[\mathrm{Pa}]$

$$P_{10} = \frac{g(789.32 - 1.2) \times 1.15}{1000} = 8.891 \quad (2.8)$$

流速$\mathrm{V}[\mathrm{m/s}]$に関して，圧力差を$\mathrm{P12}[\mathrm{Pa}]$，ピトー管係数1，空気密度$\rho[\mathrm{kg}/m^3]$として式 2.9 のように表せる.

$$V = \mathrm{C}\sqrt{\frac{2 \times P_{12}}{\rho}} \quad (2.9)$$

表の値より，$10[\mathrm{Hz}]$の時の流速は，$3.86[\mathrm{m/s}]$

$$V_{27.7} = \sqrt{2 \times \frac{8.891}{1.195}} = 3.857 \tag{2.10}$$

27.7[Hz]の時も同様に計算する.

27.7[Hz]の時の圧力差は，式 2.11 より48.708[Pa]

$$P_{12} = \frac{g(789.32 - 1.2) \times 6.3}{1000} = 48.708 \tag{2.11}$$

27.7[Hz]の時の流速は，式 2.12 より 9.03[m/s]

$$V_{27.7} = \sqrt{2 \times \frac{48.708}{1.195}} = 9.028 \tag{2.12}$$

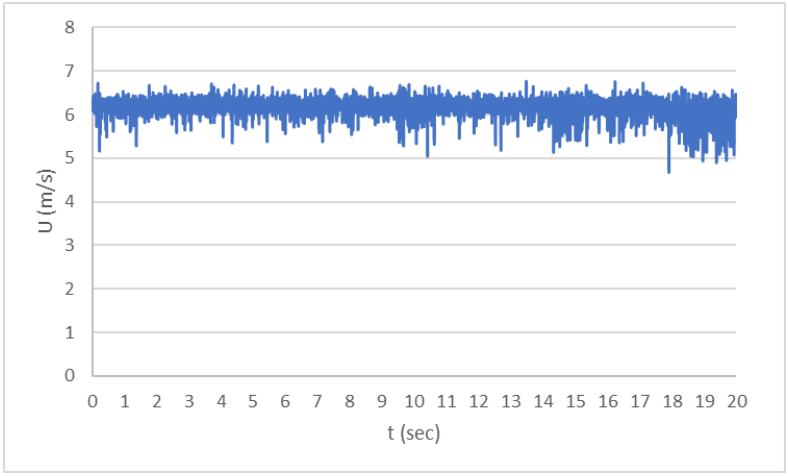
表 2.4　流速の算出

インバータ出力(Hz)	マノメータ水位(mm)	マノメータ初期水位との差(mm)	マノメータ初期水位との圧力差(Pa)	流速(m/s)
10	40.15	1.15	8.891	3.86
27.7	45.3	6.3	48.7087	9.02

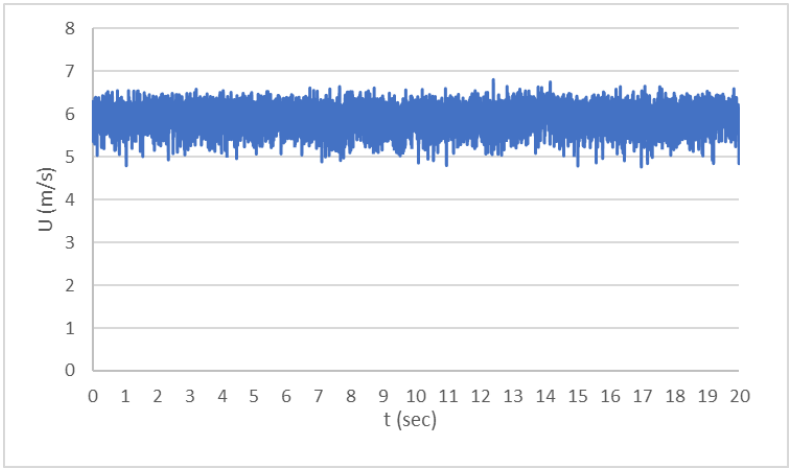
第 3 章　結果と考察

3.1 回転翼の回転速度の変化による風速応答の変化について

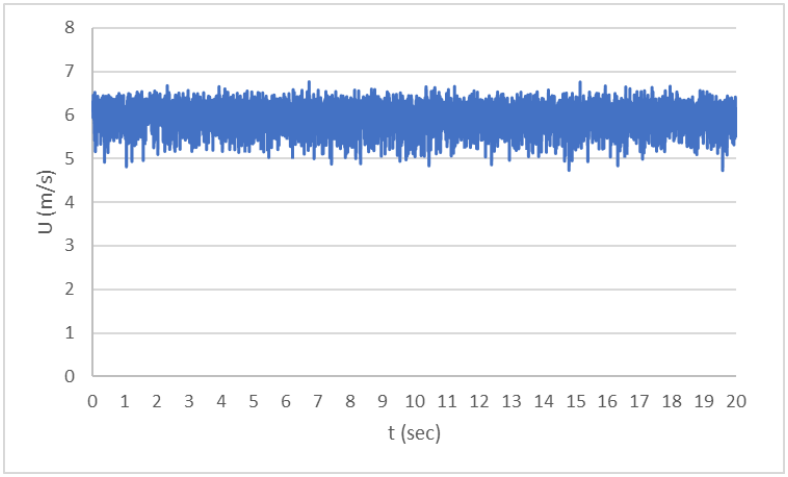
表 2.1 の条件で行った実験結果を以下に示す.



(a) 1 rpm

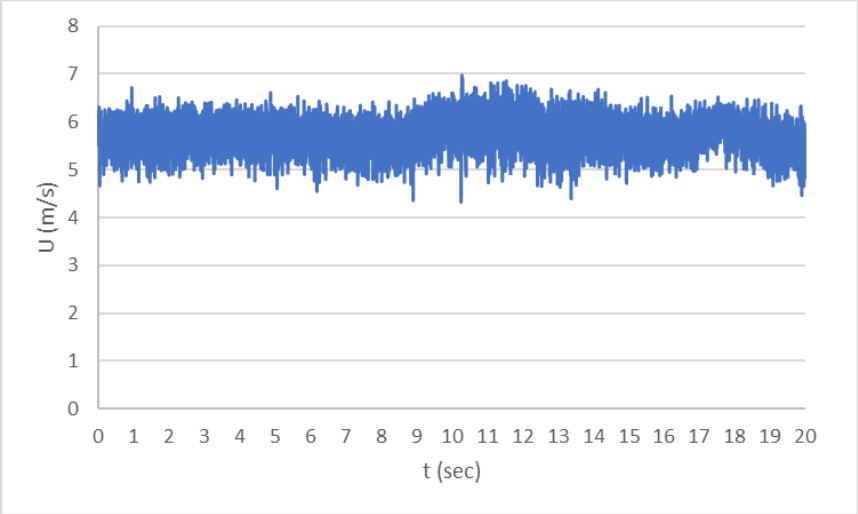


(b) 5 rpm

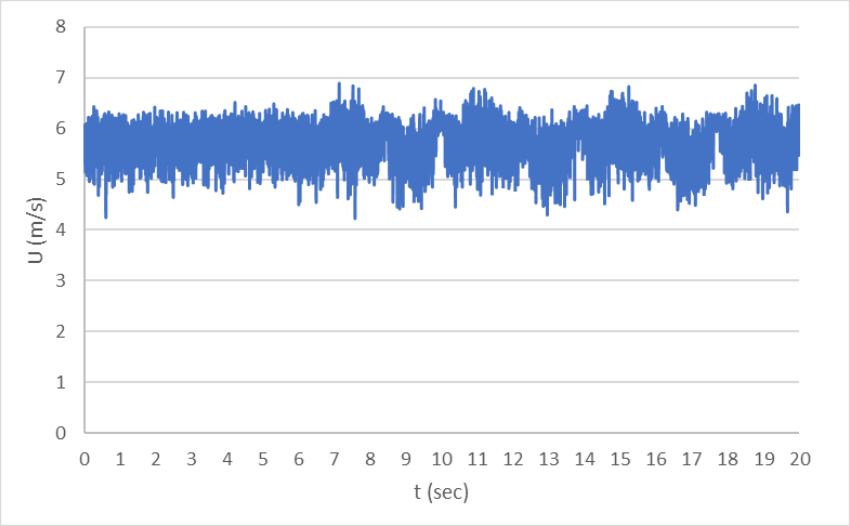


(c) 20 rpm

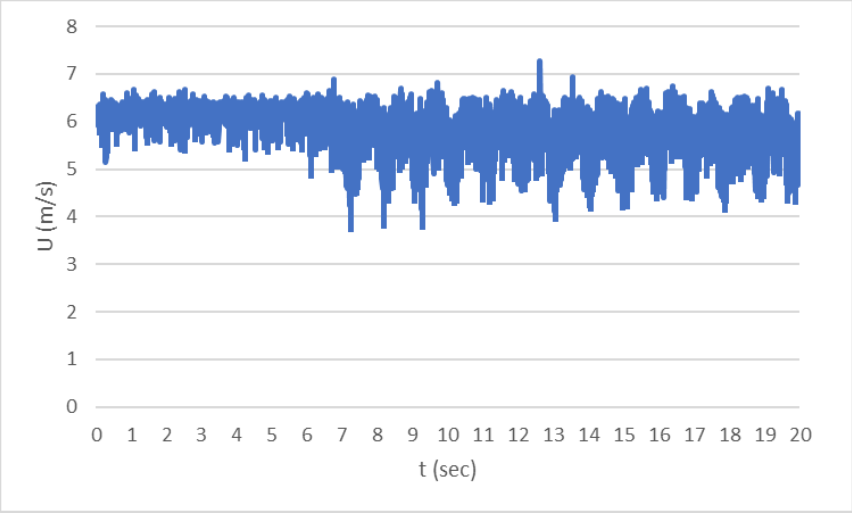
図 3.1　翼角度　-10° to 10° の実験結果



(a) 1 rpm

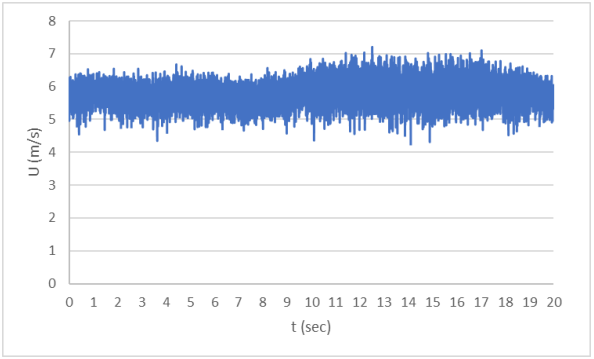


(b) 5 rpm

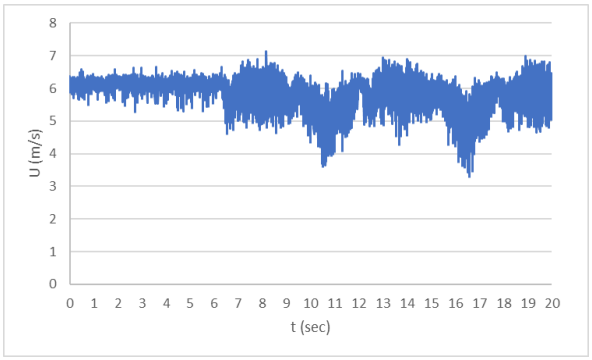


(c) 20 rpm

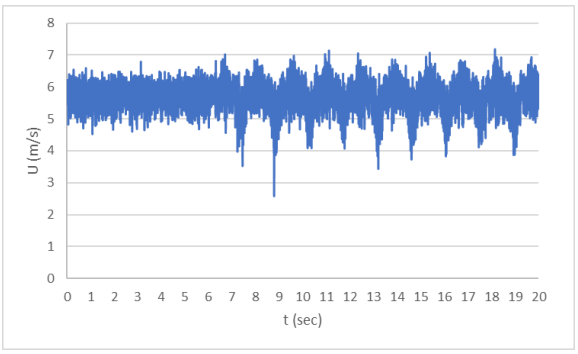
図 3.2　翼角度　-30° to 30° の実験結果



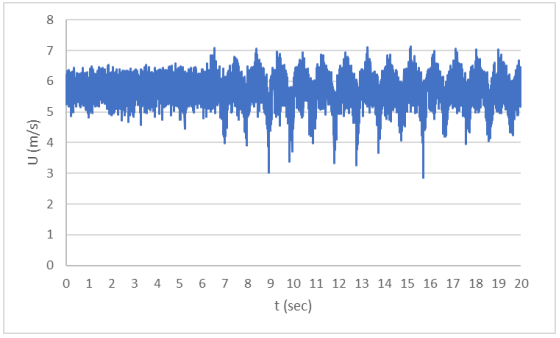
(a) 1 rpm



(b) 5 rpm

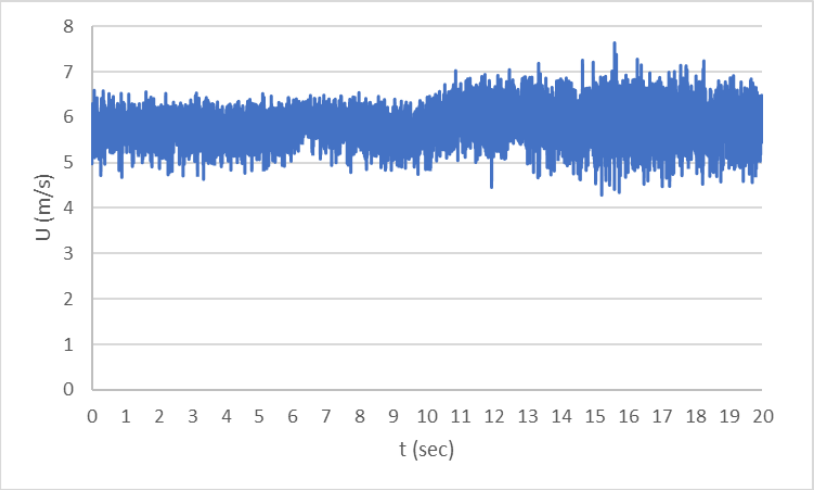


(c) 20 rpm

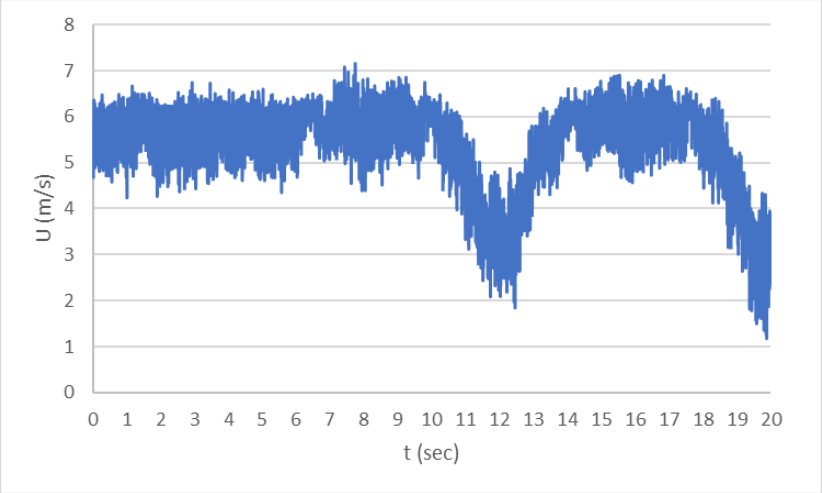


(d) 30 rpm

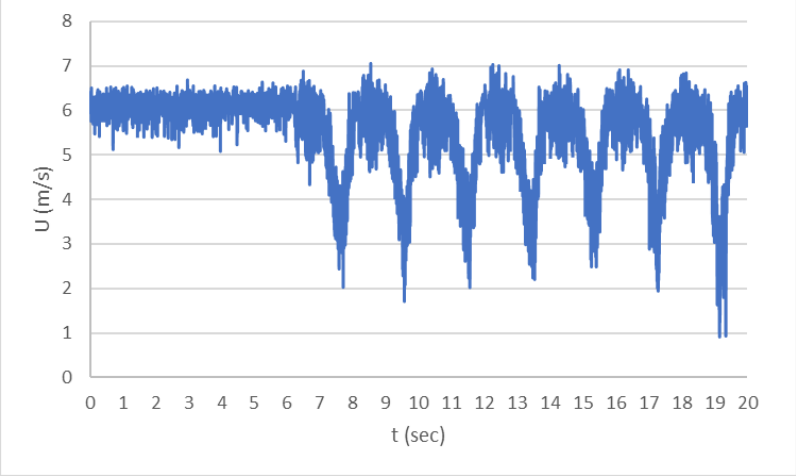
図 3.3　翼角度　-45° to 45° の実験結果



(a) 1 rpm



(b) 5 rpm

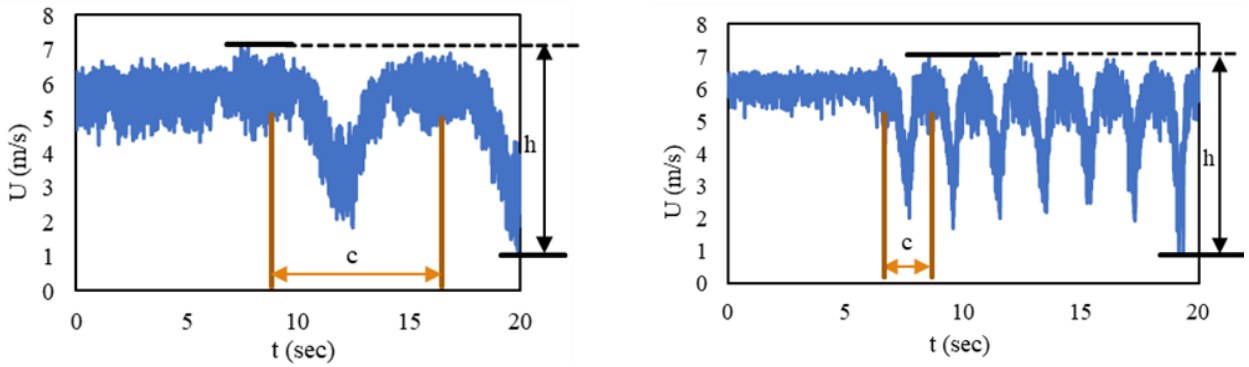


(c) 20 rpm

図 3.4　翼角度　-60° to 60° の実験結果

回転角度が低い(-10° ～10°)とき，回転速度によらず風速に大きな変化は見られなかった．回転角度-30° ～30° の実験から少しずつ変化が起き始めた．本乱流生成装置では，回転軸(直径 4mm の棒)と 0° で静止している回転翼により生成される乱流が比較的大きいため，回転角度が低いとき，風速に変化が観察されなかった原因であると考えられる．

図 3.5 に回転角度範囲-60° ～60° ，回転速度5 (rpm)と20(rpm)で周期変化させた結果を示す．5(rpm)のときは風速変動の 1 周期(c)は約 8 秒， 20rpm のときは約 2 秒である．これらの値は，回転翼の周期と一致している．最大風速変動幅 (h) は，5(rpm)で約6.8 (m/s)，20 (rpm)で約7.1 (m/s)で両者ほぼ同じ変動幅である．図示していないが，回転角度範囲が(-10° ～10°)を除いて，回転角度範囲が狭くなると，回転速度が速いほど風速変動幅が大きくなる傾向がみられた．回転角度範囲が(-10° ～10°)では，回転速度によらず風速変動に大きな変化は見られなかった．本乱流生成装置では回転軸(直径4(mm)の棒)と 0° で静止している回転翼により生成される乱流が比較的大きいことが，回転角度範囲が狭いとき風速変動に変化が観察されなかった原因であると考えられる．



(a) 翼角度 -60° to 60° (at 5 rpm)　　(b) 翼角度 -60° to 60° (20 rpm)

図 3.5　風速応答性 (C：1 周期，h：最大風速変動幅)

5(rpm)の時の風速変動の 1 周期(C)にかかる時間は約 8 秒で 20(rpm)の時は約 2 秒であり，回転翼の周期と一致している．最大風速変動幅(h)は，5(rpm)で約1.2～6.8(m/s)，20(rpm)で約0.9～7.1(m/s)で両者ほぼ同じ変動幅を持っている．しかし，低い回転角度では，回転速度が速いほど変動幅が大きくなる傾向がみられた．回転数が低い場合でも，回転数が高い場合の最大風速変動幅に近い値になるが，回転速度が高い方が瞬間的な乱流度合いを大きくしやすいと考えられる．

第 4 章　結言

小型風洞装置用の乱流生成装置を設計・製作した.
RaspberryPi4 を用いて行い，python で行った．ステッピングモータでの回転制御では，を行った．回転角度の周期変化の実験から，回転角度範囲が(-60°～60°)のとき，1 周期あたりの最大風速変動幅は回転速度によらずほぼ同じであった．回転数が低い場合(5rpm)でも，回転数が高い場合の最大風速変動幅に近い値になり，瞬間的な乱流度合いが一番大きくなったのは，10°, 30°, 60°の場合では20(rpm)，45°の場合では30(rpm)であった.

引用文献

1. 進藤　章二朗　著, “低速風洞実験法”, 1992/08/20
2. “風洞試験/試験・研究/”, 一般財団法人日本建築総合試験所 https://www.gbrc.or.jp/test_research/wind_tunnel/, 閲覧日:2021/01/29
3. 松本　勝, “タコマ橋落橋の謎と教訓”, 日本風工学会誌, 96 号, 2003/07
4. 松田一俊, “長大橋の風による振動とその制振対策”, 総合講座流れより, ながれ 21 (2002), pp.269-273
5. 蒔田秀治, “乱流風洞”, 日本流体力学会誌 「ながれ」, 21 巻, 5 号, (2002/10/25), pp. 389-390.
6. Hideharu Makita, “Realization of a large-scale turbulence field in a small wind tunnel”, Fluid Dynamics Research, Vol. 8, Issues 1-4 (1991), pp. 53-64.
7. 日本機械学會論文集. B 編, 50 巻(452), p1105-1113,1984 https://www.jstage.jst.go.jp/article/kikaib1979/50/452/50_452_1105/_pdf/-char/ja, 閲覧日:29/01/2021
8. “GPIO - Raspberry Pi Documentation”, RASPBERRY PI FOUNDATION UK REGISTERED CHARITY 1129409 https://www.raspberrypi.org/documentation/usage/gpio/, 閲覧日:2021/02/01

謝辞

乱流生成装置の部品の製作の際，近畿大学工学部の梅西浩二氏に多大なご協力を頂いた．ここに記して謝意を表す．

付録

moter_1 のプログラム

```
import RPi.GPIO as GPIO  #GPIO library
import time
time.sleep
import sys

# Developed functions
from function_stepper_motor import turn_clockwise
from function_stepper_motor import turn_counterclockwise

#--------------moter1's input starts here----------
motor1_cycle=11
motor1_angle_per_cycle=90
motor1_sleep=0

motor1_clockwise=False # Clockwise
motor1_counterclockwise=False # Flag for counter clockwise
motor1_flap_motion=True  # Flapping motion

# stepper motor properties
motor1_step=1.8 # deg
motor1_spr = 200 # step per revolution...because step=360 deg/spr



# Input values
motor1_rpm =60 # revolution per minute...limit value is 100 rpm
motor1_dt_step = 1.0/(motor1_spr*motor1_rpm/60.0)

# dt_step can be set manually as well depending upon the purpose
#dt_step = 0.05 # time per step
                # limiting value is 0.003

# Gpio Pins
motor1_pin_A1 =17 # red
motor1_pin_A2 =27 # green
```

```
motor1_pin_B1 =22 # yellow
motor1_pin_B2 =10 # blue

print('motor1_dt_step= ',motor1_dt_step)
angle_out_total=0.0

GPIO.setmode(GPIO.BCM) # to use gpio numbering system for pins
GPIO.cleanup() # Return all the channels back to input
GPIO.setwarnings(False) # To turn on off warnings

GPIO.setup(motor1_pin_A1, GPIO.OUT) #Set GPIO pin to output
GPIO.setup(motor1_pin_A2, GPIO.OUT)
GPIO.setup(motor1_pin_B1, GPIO.OUT)
GPIO.setup(motor1_pin_B2, GPIO.OUT)

#--------------moter1's input ends here----------
#------------motor1's order starts here------------
if motor1_clockwise:
    print("motor1_Turning clockwise")
    for i in range(motor1_cycle):
        print('motor1= ',i)
        time.sleep(motor1_sleep)

motor1_angle_out=turn_clockwise(motor1_pin_A1,motor1_pin_A2,motor1_pin_B1,moto
r1_pin_B2,motor1_dt_step,motor1_step,motor1_spr,angle=motor1_angle_per_cycle)
        motor1_angle_out_total=angle_out_total+motor1_angle_out

if motor1_counterclockwise:
    print("motor1_Turning counter clockwise")
    for i in range(motor1_cycle):
        print('moter1= ',i)
        time.sleep(motor1_sleep)

motor1_angle_out=turn_counterclockwise(motor1_pin_A1,motor1_pin_A2,motor1_pin_
B1,motor1_pin_B2,motor1_dt_step,motor1_step,motor1_spr,angle=motor1_angle_per_
cycle)
        motor1_angle_out_total=angle_out_total+motor1_angle_out
```

```
if motor1_flap_motion:
    print("motor1_flapping motion")
    for i in range(motor1_cycle):
        print('motor1= ',i)
        time.sleep(motor1_sleep)

motor1_angle_out=turn_clockwise(motor1_pin_A1,motor1_pin_A2,motor1_pin_B1,moto
r1_pin_B2,motor1_dt_step,motor1_step,motor1_spr,angle=motor1_angle_per_cycle)
        motor1_angle_out_total=angle_out_total+motor1_angle_out

        time.sleep(motor1_sleep)

motor1_angle_out=turn_counterclockwise(motor1_pin_A1,motor1_pin_A2,motor1_pin_
B1,motor1_pin_B2,motor1_dt_step,motor1_step,motor1_spr,angle=2*motor1_angle_pe
r_cycle)
        motor1_angle_out_total=angle_out_total+motor1_angle_out



        time.sleep(motor1_sleep)

motor1_angle_out=turn_clockwise(motor1_pin_A1,motor1_pin_A2,motor1_pin_B1,moto
r1_pin_B2,motor1_dt_step,motor1_step,motor1_spr,angle=motor1_angle_per_cycle)
        motor1_angle_out_total=angle_out_total+motor1_angle_out
#------------motor1's order ends here--------------

GPIO.cleanup() # Return all the channels back to input
```

motor_2 のプログラム

```
import RPi.GPIO as GPIO  #GPIO library
import time
time.sleep
import sys

# Developed functions
from function_stepper_motor import turn_clockwise
from function_stepper_motor import turn_counterclockwise
```

```
#--------------moter2's input starts here----------
motor2_cycle=11
motor2_angle_per_cycle=90
motor2_sleep=0

motor2_clockwise=False # Clockwise
motor2_counterclockwise=False # Flag for counter clockwise
motor2_flap_motion=True  # Flapping motion



# stepper motor properties
motor2_step=1.8 # deg
motor2_spr = 200 # step per revolution...because step=360 deg/spr

# Input values
motor2_rpm =60 # revolution per minute...limit value is 100 rpm
motor2_dt_step = 1.0/(motor2_spr*motor2_rpm/60.0)

# dt_step can be set manually as well depending upon the purpose
#dt_step = 0.05 # time per step
                # limiting value is 0.003

# Gpio Pins

motor2_pin_A1 =5  # red
motor2_pin_A2 =6  # green

motor2_pin_B1 =13 # yellow
motor2_pin_B2 =19 # blue

print('motor2_dt_step= ',motor2_dt_step)
angle_out_total=0.0

GPIO.setmode(GPIO.BCM) # to use gpio numbering system for pins
GPIO.cleanup() # Return all the channels back to input
GPIO.setwarnings(False) # To turn on off warnings
```

```
GPIO.setup(motor2_pin_A1, GPIO.OUT) #Set GPIO pin to output
GPIO.setup(motor2_pin_A2, GPIO.OUT)
GPIO.setup(motor2_pin_B1, GPIO.OUT)
GPIO.setup(motor2_pin_B2, GPIO.OUT)

#--------------motor2's input ends here----------



#------------motor2's order starts here------------


if motor2_clockwise:
    print("motor2_Turning clockwise")
    for i in range(motor2_cycle):
        print('motor2= ',i)
        time.sleep(motor2_sleep)

motor2_angle_out=turn_clockwise(motor2_pin_A1,motor2_pin_A2,motor2_pin_B1,moto
r2_pin_B2,motor2_dt_step,motor2_step,motor2_spr,angle=motor2_angle_per_cycle)
        motor2_angle_out_total=angle_out_total+motor2_angle_out

if motor2_counterclockwise:
    print("motor2_Turning counter clockwise")
    for i in range(motor2_cycle):
        print('motor2= ',i)
        time.sleep(motor2_sleep)

motor2_angle_out=turn_counterclockwise(motor2_pin_A1,motor2_pin_A2,motor2_pin_
B1,motor2_pin_B2,motor2_dt_step,motor2_step,motor2_spr,angle=motor2_angle_per_
cycle)
        motor2_angle_out_total=angle_out_total+motor2_angle_out

if motor2_flap_motion:
    print("motor2_flapping motion")
    for i in range(motor2_cycle):
        print('motor2= ',i)
        time.sleep(motor2_sleep)
```

```
motor2_angle_out=turn_clockwise(motor2_pin_A1,motor2_pin_A2,motor2_pin_B1,moto
r2_pin_B2,motor2_dt_step,motor2_step,motor2_spr,angle=motor2_angle_per_cycle)
        motor2_angle_out_total=angle_out_total+motor2_angle_out

        time.sleep(motor2_sleep)

motor2_angle_out=turn_counterclockwise(motor2_pin_A1,motor2_pin_A2,motor2_pin_
B1,motor2_pin_B2,motor2_dt_step,motor2_step,motor2_spr,angle=2*motor2_angle_pe
r_cycle)
        motor2_angle_out_total=angle_out_total+motor2_angle_out



        time.sleep(motor2_sleep)

motor2_angle_out=turn_clockwise(motor2_pin_A1,motor2_pin_A2,motor2_pin_B1,moto
r2_pin_B2,motor2_dt_step,motor2_step,motor2_spr,angle=motor2_angle_per_cycle)
        motor2_angle_out_total=angle_out_total+motor2_angle_out



print('before',motor2_angle_out_total)

# return the plate to zero angle
return_to_zero= input("Want to retunr to zero? If true type T, if false type F
¥n ")
if return_to_zero=='T':
    return_to_zero=True
elif return_to_zero=='F':
    return_to_zero=False
else:
    print('You can only use T or F to define this flag')

if return_to_zero:
    motor2_angle_out_total=motor2_angle_out_total%360
    print('motor2_after mod',motor2_angle_out_total)

motor2_angle_out=turn_counterclockwise(motor2_pin_A1,motor2_pin_A2,motor2_pin_
B1,motor2_pin_B2,motor2_dt_step,motor2_step,motor2_spr,angle=motor2_angle_out_
```

```
total)
    motor2_angle_out_total=motor2_angle_out_total+motor2_angle_out
    print('motor2_finally',motor2_angle_out_total)
   #------------motor2's order ends here--------------



GPIO.cleanup() # Return all the channels back to input
```

motor_1, motor_2 の関数用プログラム

```
# Functions for stepper motor control

import RPi.GPIO as GPIO  #GPIO library
import time
import sys



#---------------
# turn clockwise
#---------------
def turn_clockwise(pin_A1,pin_A2,pin_B1,pin_B2,dt_step,step,spr,angle):

    iter_max = int(angle/(step*4))

    for i in range(iter_max):
        #---each loop will turn 1.8 *4 = 7.2 deg

        GPIO.output(pin_A1,1) #high
        GPIO.output(pin_B2,0) #low
        GPIO.output(pin_A2,0) #low
        GPIO.output(pin_B1,1) #high

        time.sleep(dt_step)

        GPIO.output(pin_A1,0) #low
        GPIO.output(pin_B2,0) #low
        GPIO.output(pin_A2,1) #hgh
```

```
        GPIO.output(pin_B1,1) #high
        time.sleep(dt_step)

        GPIO.output(pin_A1,0) #low
        GPIO.output(pin_B2,1) #high
        GPIO.output(pin_A2,1) #hih
        GPIO.output(pin_B1,0) #low
        time.sleep(dt_step)

        GPIO.output(pin_A1,1) # high
        GPIO.output(pin_B2,1) # high
        GPIO.output(pin_A2,0) # low
        GPIO.output(pin_B1,0) # low
        time.sleep(dt_step)
    angle_out=angle
    return angle_out
#---------------




#----------------
# turn counterclockwise
#----------------
def turn_counterclockwise(pin_A1,pin_A2,pin_B1,pin_B2,dt_step,step,spr,angle):

    iter_max = int(angle/(step*4))

    for i in range(iter_max):
        #---each loop will turn 1.8 *4 = 7.2 deg

        GPIO.output(pin_A1,1) #high
        GPIO.output(pin_B2,1) #low
        GPIO.output(pin_A2,0) #low
        GPIO.output(pin_B1,0) #high
        time.sleep(dt_step)

        GPIO.output(pin_A1,0) #low
        GPIO.output(pin_B2,1) #low
```

```
        GPIO.output(pin_A2,1) #hgh
        GPIO.output(pin_B1,0) #high
        time.sleep(dt_step)

        GPIO.output(pin_A1,0) #low
        GPIO.output(pin_B2,0) #high
        GPIO.output(pin_A2,1) #hih
        GPIO.output(pin_B1,1) #low
        time.sleep(dt_step)

        GPIO.output(pin_A1,1) # high
        GPIO.output(pin_B2,0) # high
        GPIO.output(pin_A2,0) # low
        GPIO.output(pin_B1,1) # low
        time.sleep(dt_step)
    angle_out=-angle
    return angle_out
#----------------
```