

令和 8 年度大学院[春・秋]入学 3 月試験

未来ロボティクス専攻

論 述 試 験[Essay examination]

【60 分】[60 minutes]

[注意事項] [Important information]

1. 試験開始の合図があるまで、この問題冊子を開かないでください。
(Do not open this question booklet until the examination proctor signals you to begin.)
2. この問題冊子には論述試験の問題が記載されています。万一、落丁・乱丁があった場合は、手をあげて申し出てください。
(This question booklet contains questions for the essay examination. If you find any missing or misplaced questions, raise your hand to report them.)
3. 解答用紙は裏面も使用することができます。
(Both sides of the answer sheet can be used.)
4. 受験票、黒鉛筆、シャープペンシル、消しゴム、鉛筆けずり、時計、眼鏡、ハンカチ、目薬、袋から取り出したティッシュペーパー以外は机の上に置かないでください。
(Only put on your desk following items:
your examination voucher, pencils, mechanical pencils, erasers, pencil sharpeners, watches, glasses, handkerchiefs, eye drops, and tissue paper without the cover.)
5. 監督者の指示に従い、解答用紙に受験番号・氏名を記入してください。
(Fill in your examination number and name on the answer sheet according to the instructions of the examination proctor.)
6. 試験終了の合図で解答を終了し、監督者の指示があるまで席を立たないでください。また問題冊子は持ち帰ってください。
(Finish answering the questions at the signal of the end of the exam and do not leave your seat until instructed by the examination proctor. Take the question booklet home with you.)
7. 解答用紙は最大 2 枚まで提出できます（1 枚でも構いません）。2 枚提出する場合は解答用紙の右上に 1/2、2/2 の番号を記入してください。
(Up to two answer sheets can be submitted (submission of only one sheet is also permitted). If two sheets are submitted, “1/2” and “2/2” should be written in the upper right corner of each sheet, respectively.)
8. 解答は日本語もしくは英語のみとする。
(Answers must be in Japanese or English only.)

以下の A・B・C から 1 題を選択し，解答せよ．

(Choose one question from A, B, or C below and answer.)

- A. 図 A-1 のように水平に距離 L だけ離れた 2 つの台 A, B があり，その中央の鉛直な直線 C (図では一点鎖線) 上のどこかに重量 W の物体 D を保持する構造物を考案せよ．部材は 1 つでもよいし，複数でジョイント等の接続部があってもよい．台は上面のみを接続部として使用すること．複数の構造物を考案して解答してもよい．

ただし，台 A, B の上面の高さは同じで，いずれも動かないものとする．物体 D 以外の部材の質量はゼロとしてよい．台 A, B および物体 D の大きさは無視してよい．平面内の問題として考えてよいが，部材の奥行き方向の寸法も考慮すること．

以下の問に答えよ．

- (1) 考案した構造物を図示せよ．自在な部分がある場合には，はっきりわかるように示すこと．構造物の各部の寸法を文字として定義せよ．ただし，接続部等の微細な部分の寸法は無視してよい．また，使用する材料の物性を仮定し，定数の文字として定義せよ．
- (2) 部材の各部に発生する応力を考察せよ．特に最大の応力（あるいは設計上考慮すべき大きな応力）が発生する部分に注目して説明せよ．
- (3) 物体 D を保持しないときに対して，保持したときの各部の変位について考察せよ．

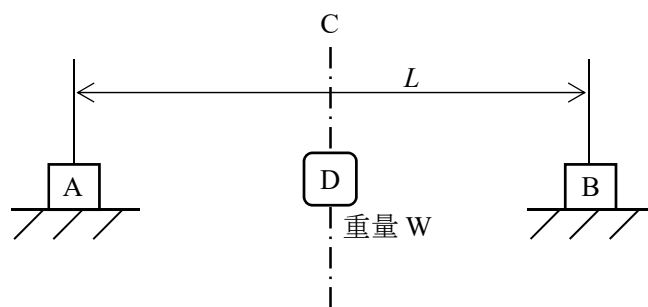
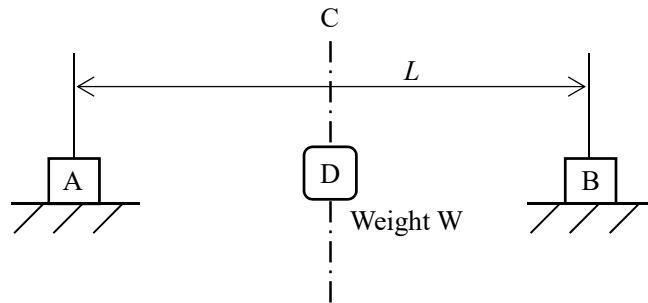


図 A-1

A. Figure A-1 shows the two identical bases, A and B, fixed on the horizontal surface. The horizontal distance between A and B is L . Design a structure on both A and B which holds a particle D weighing W in the perpendicular bisector C of A and B. The structure may be constructed by one member or some members with joints. You may treat the problem as two-dimensional structure but include the depth of every member. You may show more than one structure. Answer the following questions.

- 1) Draw the proposed structure with the parameters of its lengths and physical properties. You may ignore the length of small member.
- 2) Consider the stress of each member and discuss where the maximum or large (if needed) stress acts.
- 3) Estimate the displacement of each member. State the effect of D.



☒ A-1

B. 前進および後退が可能な差動二輪移動ロボットの制御回路を設計せよ。各車輪は DC モータで駆動し、マイクロコントローラによって各車輪の速度制御系を構築する。

車輪の半径を (r) とし、モータ軸と車輪の結合には、減速比 $(1/n)$ のギアボックスを介するものとする。

以下の問に答えよ。

- (1) 車輪速度制御を実現するために使用するセンサの種類と、その基本機能を説明せよ。

また、マイクロコントローラへセンサ信号を入力するために必要な I/O ポートの種類および個数を設計し、その理由を示せ。

- (2) センサの基本性能仕様を定義し、センサを装着する位置（どの部位に取り付けるか）を明記せよ。

さらに、定義したセンサ出力から車輪の移動速度（線速度）を計算する式を示せ。

- (3) モータの速度制御には PWM（パルス幅変調）方式を採用する。モータ駆動にはトランジスタまたは MOSFET を用いるものとする。このとき、マイクロコントローラから出力する I/O ポートの個数と、それぞれに必要な基本特性を説明せよ。

さらに、マイクロコントローラからモータ制御までの回路構成に必要な素子を列挙し、回路図を描け。

- (4) センサ情報を用いて車輪の速度制御を実現する際に、速度フィードバックゲインがどのように用いられるかを説明せよ。

その際、式を併記してよい。また、使用する記号を定義すること。

B. Design the control circuitry for a differential-drive, two-wheeled mobile robot capable of moving forward and backward. Each wheel is driven by a DC motor, and a microcontroller is used to implement a speed control system for each wheel. Let the wheel radius be (r) , and assume that the motor shaft is connected to the wheel via a gearbox with a reduction ratio of $(1/n)$.

Answer the following questions.

- (1) Describe the type(s) of sensor(s) used to realize wheel speed control and explain their basic functions. In addition, design the type and number of I/O ports required on the microcontroller to accept the sensor signals, and justify your choices.
- (2) Define the basic performance specifications of the sensor(s) and specify the mounting location(s) (i.e., which part(s) of the robot) for the sensor(s). Furthermore, provide the formula for calculating the wheel linear speed from the defined sensor outputs.
- (3) Adopt PWM (pulse-width modulation) for motor speed control. Assume that the motor driver uses transistors or MOSFETs. In this case, specify the number of output I/O ports required on the microcontroller and describe the basic required characteristics of each. Additionally, list all components needed to build the circuit from the microcontroller to the motor drive, and draw the circuit diagram.
- (4) Explain how the speed feedback gain is used when implementing wheel speed control using sensor information. You may include equations; if so, define all symbols used.

- C. 次のコードの `search` 関数は、長さ `len` の `int` 型、値が昇順でソート済みの配列 `data` を受け取り、値が `value` の要素のインデックスを返す関数である。例えば、`value=-5` なら `0` を返し、`value=6` なら `5` を返す。また、`value` が `data` 内に存在しなければ `-1` を返し、`value` が `data` 内に 2 つ以上あれば、どちらか一方を返す。例えば `value=-3` なら `-1` を返し、`value=5` なら `3` あるいは `4` を返す（実装次第でどちらを返してもよい）。

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int search(int value, int *data, int len) {
5     /* ??? */
6 }
7
8 int main(int argc, char const *argv[]) { //omitting error handling
9     int value = atoi(argv[1]);
10    int data[10] = {-5, -2, 4, 5, 5, 6, 7, 7, 8, 9}; //sorted
11    int ans = search(value, data, 10);
12    printf("%d\n", ans);
13    exit(0);
14 }
```

- (1) まず簡単な実装として、`data` の各要素が `value` と一致するかをインデックス `0` から順番に調べ、一致したらインデックスを返し、最後まで一致しなければ `-1` を返す `search` 関数を実装せよ。答案用紙には関数全体を記述すること。同等な関数を `Python` あるいは `C++` で記述して解答してもよい。
- (2) (1)の実装は、上記コードのように `data` の要素数が `10` 個程度であると何も問題はないが、`data` の要素数が数万、数億になるとより効率のよい実装に置き換えた方がよい。そこで、より効率のよいアルゴリズムをひとつ示し、言葉で説明せよ。また、そのアルゴリズムがなぜ効率がよいかを説明せよ。なお、ここでいう「効率のよさ」とは、`value` の値をランダムに選んだ場合に、時間計算量が小さくなることを意味する。
- (3) (2)のアルゴリズムで、(1)同様、`search` 関数を実装せよ。答案用紙には関数全体を記述すること。同等な関数を `Python` あるいは `C++` で記述して解答してもよいが、アルゴリズムの多くの部分を含む関数やメソッドなどをライブラリから選んで使用してはならない。

- C. The search function in the following code takes an array `data` (the number of elements is the argument `len`) sorted in ascending order. Then it returns the index of the element whose value matches the argument `value`. For example, if `value=-5`, it returns `0`, and if `value=6`, it returns `5`. Also, if `value` does not exist in `data`, it returns `-1`. If `value` exists more than once in `data`, it returns one of the indexes. For example, if `value=-3`, it returns `-1`, and if `value=5`, it returns either `3` or `4`. Which index is returned can be determined by the implementation.

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int search(int value, int *data, int len) {
5     /* ??? */
6 }
7
8 int main(int argc, char const *argv[]) { //omitting error handling
9     int value = atoi(argv[1]);
10    int data[10] = {-5, -2, 4, 5, 5, 6, 7, 7, 8, 9}; //sorted
11    int ans = search(value, data, 10);
12    printf("%d\n", ans);
13    exit(0);
14 }
```

- (1) As a simple implementation, write the `search` function that checks whether each element of `data` matches `value` from index `0` to `len-1`. When it finds a matched index, it returns the index immediately, otherwise it returns `-1`. Please show the entire function on the answer sheet. You may also answer by writing an equivalent function in Python or C++.
- (2) The implementation of (1) works without any problem when the number of elements in `data` is around `10`. However, when the number of elements in `data` is tens of thousands or hundreds of millions, it is better to replace it with a more efficient implementation. Think of a more efficient algorithm and explain it in words. Also, explain why that algorithm is efficient. Note that "efficiency" here means that the expected calculation time is shorter when the value of `value` is chosen randomly.
- (3) Implement the `search` function using the algorithm in (2). Write the entire function on the answer sheet. You may also write an equivalent function in Python or C++, but you should not use library functions or methods that contain large parts of the algorithm.

令和 8 年度大学院[春・秋]入学 3 月試験 未来ロボティクス専攻 論述試験 解答例

A. 解答例

(1) 高さ h , 奥行き b の矩形断面で, 長さ L の梁とする.

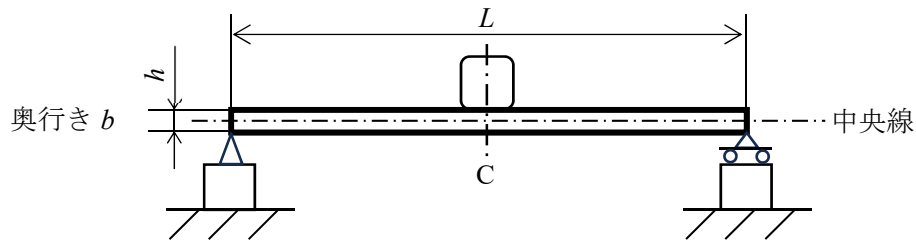


図 A-2

部材の縦弾性率 E , 横弾性率 G

(2) 部材の中央線より上側に圧縮応力, 下側に引張応力が発生し, その大きさは中央線からの距離に比例する. また, 曲げモーメントは左右の台上ではゼロで直線 C 上が最大となる. そのため, 圧縮応力が最大になるのは C 上の最上面, 引張応力は C 上の最下面が最大で, その絶対値は同じである.

断面係数 Z は

$$Z = \frac{bh^2}{6}$$

C 上の曲げモーメント M は

$$M = \frac{WL}{4}$$

であるので, 最大応力 $\sigma_{\max}$ の大きさは

$$\sigma_{\max} = \frac{M}{Z} = \frac{3WL}{2bh^2}$$

となる.

一方, せん断応力 τ の大きさは部材全体で等しく,

$$\tau = \frac{W}{2bh}$$

である.

(3) 変位は C 上が最大である. (理由が示されていなくても可とする)

断面 2 次モーメント I は

$$I = \frac{bh^3}{12}$$

であるので、変位 δ は

$$\delta = \frac{WL^3}{48EI} = \frac{WL^3}{4Ebh^3}$$

である。（導出の過程が示されていない可とする）

一方、せん断応力による長さあたりのずれ量は τ/G であり、台から C 上までの積算ずれ量 δ_s は

$$\delta_s = \frac{\tau}{G} \cdot \frac{L}{2} = \frac{WL}{4Gbh}$$

となる．曲げによる変位とせん断による変位の比は

$$\frac{\delta_s}{\delta} = \frac{h^2}{L^2} \cdot \frac{E}{G}$$

であり、 $G \doteq 0.7E$, $h \ll L$ とすると、 δ_s は δ にくらべて無視できるほど小さい．

別解

（１）断面積 S_1 の梁 2 本を角度 30 度で 3 角形に組む．接続部は回転自在とする．梁の長さは $L/\sqrt{3}$ である．この中央部に断面積 S_2 、長さ L_2 のワイヤで物体を吊るす．

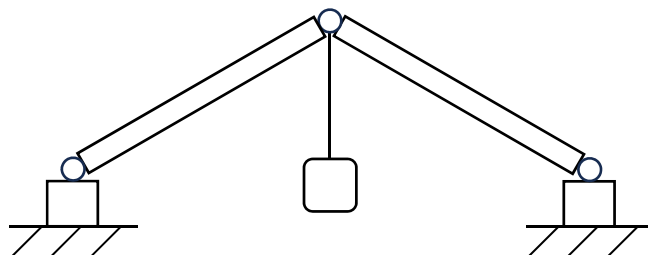
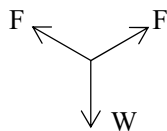


図 A-3

梁とワイヤの縦弾性係数 E

（２）梁には圧縮力がかかり，その大きさ F は，図のような力ベクトルのつり合いから W に等しい．この力は梁を座屈させるほどのものではないとする．



圧縮応力は梁の内部で一様な値 σ_1 で

$$\sigma_1 = \frac{W}{S_1}$$

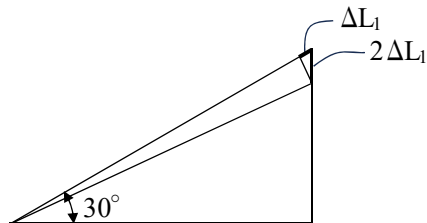
ワイヤの張力は W で，引張応力 σ_2 は

$$\sigma_2 = \frac{W}{S_2}$$

(3) 梁の長さの縮み量 ΔL_1 は

$$\Delta L_1 = \frac{\sigma_1}{E} \cdot \frac{L}{\sqrt{3}} = \frac{WL}{\sqrt{3}S_1E}$$

中央接続部の下がり変位は $2\Delta L_1$ と近似的に計算できる.



一方、ワイヤの伸び量 ΔL_2 は

$$\Delta L_2 = \frac{\sigma_2}{E} L_2 = \frac{WL_2}{S_2E}$$

であり、物体 D の下がり変位量 δ_2 は

$$\delta_2 = 2\Delta L_1 + \Delta L_2 = \left(\frac{2L}{\sqrt{3}S_1} + \frac{L_2}{S_2} \right) \frac{W}{E}$$

である.

B. 解答例

- (1) 車輪速度制御を実現するためのセンサとして、ロータリエンコーダ（インクリメンタル型）を用いる。

ロータリエンコーダは、回転軸の角変位または回転数に比例したパルス信号を出力するセンサであり、一定時間内のパルス数を計測することで回転速度を求めることができる。

各車輪の駆動部に 1 個のエンコーダを取り付け、A 相・B 相の 2 相信号を用いることで、回転速度および回転方向の判別が可能であり、前進と後退の運動速度を計測できる。

マイクロコントローラに入力する I/O ポートとしては、

- ・デジタル入力ポート：2 ポート × 2 輪 = 4 ポート

が必要となる。これらのポートは割り込み入力に対応していることが望ましく、高速なパルス計数を行うためにタイマ／カウンタ機能と組み合わせて使用する。

- (2) エンコーダの基本性能仕様として以下を定義する。

- ・分解能：N [pulse/rev]
- ・最大応答回転数：モータ最大回転数以上
- ・出力形式：デジタルパルス出力、A 相・B 相の 2 相信号

エンコーダはモータ軸に取り付けるものとする。

減速比が $1/n$ のとき、モータ角速度 ω_m と車輪角速度 ω_w の関係は

$$\omega_w = \omega_m / n$$

で与えられる。

一定時間 Δt に観測されたパルス数を ΔP とすると、

$$\omega_m = (2\pi / N) \cdot (\Delta P / \Delta t)$$

となる。よって車輪の線速度 v は

$$v = r \cdot \omega_w = r \cdot (1/n) \cdot (2\pi / N) \cdot (\Delta P / \Delta t)$$

で求められる。

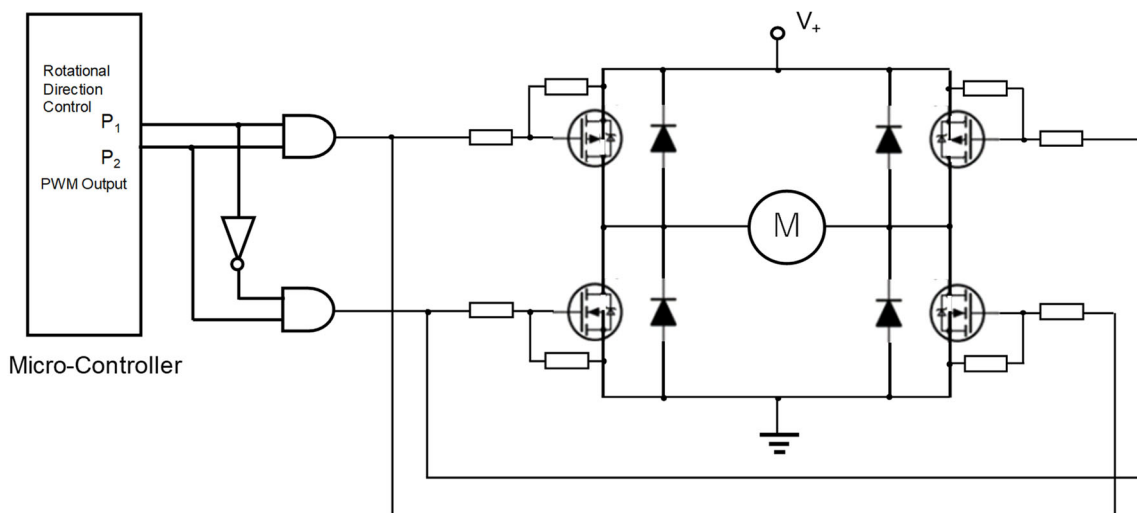
- (3) モータの速度制御には PWM 方式を用いる。PWM のデューティ比を制御することで、モータに印加される平均電圧を変化させ、回転速度を調整する。

各モータにつき、PWM 出力ポート 1 個、回転方向制御用デジタル出力 1 個が必要であり、2 輪分で合計 4 出力ポートが必要となる。

回路構成に必要な主な素子は以下の通りである。

- MOSFET またはトランジスタ
- フライバックダイオード
- ゲート抵抗
- モータ電源
- マイクロコントローラ
- And ゲート素子と Not ゲート素子

設計回路の一例（マイコン・論理回路電源および V_+ が 5V の場合）



(4) 目標車輪速度を v_{ref} 、実測速度を v とすると、速度誤差 e は

$$e = v_{\text{ref}} - v$$

で定義される。比例制御では制御入力 u は

$$u = K_v \cdot e$$

で与えられる。ここで K_v は速度フィードバックゲインである。

K_v を大きくすると応答は速くなり、定常誤差は減少する。高すぎる K_v の設定は不安定になりやすく、小さくすると安定性は高いが応答が遅くなる。

上記のフィードバック制御は、比例制御の例である。一般的により良い目標速度への追従性能を実現したり、定常偏差を減らしたりするためには、速度誤差に対する PID 制御系を構築するのが一般的である。

C. 解答例

(1) 例

```
int search(int key, int *data, int len) {  
    for(int i=0; i<len; i++) {  
        if(key == data[i]){  
            return i;  
        }  
    }  
  
    return -1;  
}
```

- (2) ※細部の説明は求めず，原理を理解しており，幹となるアルゴリズム，効率の良さの説明がなされていれば満点とする．

アルゴリズムの説明：配列の 0 番目の要素側を「左」、 $\text{len}-1$ 番目の要素側を「右」として説明する．また、「探索対象の範囲」の「左端」，「右端」を考える．最初、左端，右端をそれぞれ配列の最も左，最も右にとる．以後、左端、右端の midpoint の要素の値と value を比較し，要素の値が value と一致したらその要素のインデックスを返す．一致せず、 value が大きいなら左端，小さいなら右端をその要素の位置に移す．これを繰り返し，範囲がなくなったら -1 を返す．

なぜ効率が良いか：上記のアルゴリズムは，探索の範囲が一度の比較で半分ずつ狭くなっていく．範囲がなくなるまでの比較の回数は，要素数を N とすると最悪 $\log_2 N$ 回となる．一方，(1) のアルゴリズムでの探索回数の期待値は $N/2$ となる．したがって， N が数万，数億と大きくなると，(2) のアルゴリズムの最悪の探索回数が (1) のアルゴリズムの探索回数の期待値を大きく下回ることになる．たとえば $N=8192$ の場合， $\log_2 N=13$ 、 $N/2=4096$ となり，(1) から (2) の `search` 関数に変更すると，300 倍以上の高速化が期待できる．

(3)例

```
int search(int value, int *data, int len) {
    int left = 0;
    int right = len;
    int pos = len/2;
    int prev = -1;
    while(prev != pos) {
        prev = pos;
        if(data[pos] == value)      return pos;
        else if (data[pos] < value) left = pos;
        else                        right = pos;
        pos = (right + left)/2;
    }

    return -1;
}
```