マイクロロボットを用いた熱駆動系の実験

東京電機大学 理工学研究科 情報システム工学専攻
小島慎護，大野寛和，吉田康平，川崎崇博

平成 20 年 12 月 19 日（金）

目 次

1　概要と目的

本実験では熱による形状記憶合金 (Shape Memory Alloy，SMA) を用いたロボット駆動の実験を行った．SMA は変態点以上の温度では，超弾性と呼ばれる元の形状に回復する性質を持ち，温度変化で駆動するアクチュエータとして用いられている．衣類やメガネなど変形しやすく，元の形状に戻したいが電気を利用できないものや，内視鏡など細くモータを用いることのできないものにも利用されている．また，人工筋などに応用する技術も研究されている．SMA はさまざまな場面で用いられ，また今後の利用も期待されている．

本実験で用いている SMA バイアスアクチュエータを図 1 に示す．

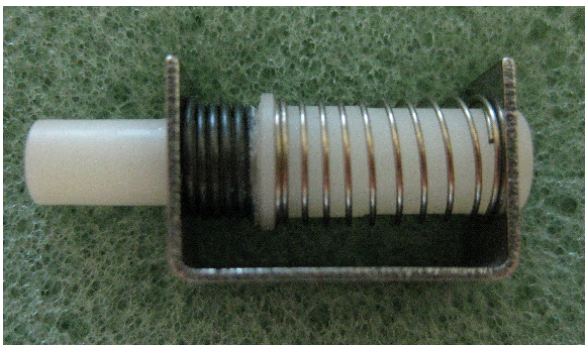

図 1: SMA バイアスアクチュエータ

図 1 左側が SMA になっており右側が通常のバネになっている．SMA を加熱すると，SMA の発生力が通常バネの発生力より大きくなり，右側にアクチュエータが動く．冷却すると，通常バネの発生力が SMA の発生力より大きくなり，左側にアクチュエータが動く双方向アクチュエータとなっている．本実験ではこの SMA の簡易的なモデルを導出し，シミュレーションを行う．実験にはジュール熱を利用し電気で SMA を駆動させるロボットを対象とし，電圧を入力，温度を状態，位置の変位を出力としたモデルを扱う．また，SMA を用いたマイクロロボットを製作し，実際に制御することで実機および制御系の有効性について検証を行う．

実際に動くものを実験対象とすることで，ロボットを作ることの楽しさを伝えることができ，楽しく科学に触れることのできる実験を目指す．

2　形状記憶合金の動作原理

2.1　マルテンサイト変態

鋼に焼き入れたときに非常に硬い相が現れる．焼入れ硬化した鋼材を研磨し，適当な化学腐食を行って顕微鏡で観察すると麻の葉状の組織が見える．この組織のことをマルテンサイトと呼ぶ．

高温で面心立方格子をもっている鋼が急冷されると，途中で拡散を必要とする相変化を起こさずに過冷却される．そのため，平衡状態図にはないが高温相より低い自由エネルギーをもつ体心正方格子の相に変換する．これがマルテンサイト変態であり，マルテンサイトは体心正方格子をもつこととなる．高温時の面心立方格子構造をもつ相をオーステナイト相と呼ぶ．ある金属組織から他の金属組織に変わることを変態と呼ぶ．

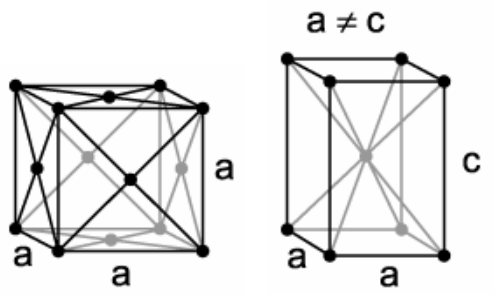

図 2: 面心立方格子 (左)，体心正方格子 (右)

この変態の最も本質的な特徴は，変態の際の変形に伴う歪エネルギーの増加が極めて大きいことであり，その大きさに応じて歪エネルギーの増大を避けるために種々の事象が同時に起こっていることである．

2.2 SMA の特徴

1. **熱履歴**　マルテンサイト変態には一般に過冷却，または過熱が必要である．また，変態を開始した温度では変態はほとんど進行しない．変態量と温度の間には一定の関係がある．

2. **冷却速度の影響**　変態温度は冷却速度にほとんど影響されない．したがって，多くの場合マルテンサイト変態はどのように冷却しても阻止することができない．ただし，変態温度が室温以下で合金組成が特別な場合には一部分だけ阻止することが可能である．

3. **格子関係**　変態相の結晶の方位はもとの結晶の方位と一定の関係を持っている．

4. **晶癖面**　マルテンサイトは多くの場合，特定の面に平行に板状が形成される．この結晶面の方向をもとの結晶の指数で表し，晶癖面と呼ぶ．

5. **巨視的変形**　板状に形成されたマルテンサイトの巨視的変形は，この板面に沿ってのせん断変形が大部分であって，その他の変形成分は無視することができる．マルテンサイトが結晶表面まで突き抜けると，双晶帯のように表面を傾ける．

6. **伝播速度**　マルテンサイトの成長速度は一般に大きいが，合金の種類および変態の条件によって異なる．特に速度が速い場合は鉄合金が爆発的に変態するときである．そのとき 1 枚 1 枚のマルテンサイトは音速で伝わると考えられ，10^{-6} 秒程度の短い時間内に数百枚のマルテンサイトが形成される．ある程度以上結晶粒度があれば，連鎖反応的に全体にわたって一様な密度のマルテンサイトが瞬間的に形成される．

7. **残留格子欠陥**　形成されたマルテンサイトは内部に双晶薄片，転位 (線状の格子欠陥)，積層欠陥などの格子欠陥を含むことが多い．どの種類の欠陥がどの程度形成されるかは合金の種類，変態の条件等によって異なる．

2.3 SMA の各特性

今回用いる SMA は NT 合金と呼ばれるニッケルとチタンの合金である．温度とともに変態を伴い，特性が変化する．図 3 に電気特性を示す．

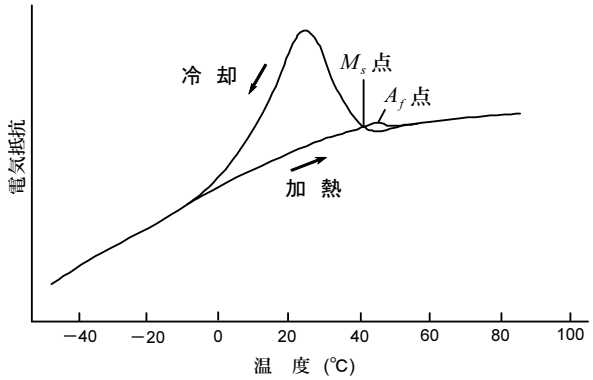

図 3: 電気特性 [12]

通常の金属では電気抵抗は，高温で大きく，温度変化に対してほぼ直線的に変化する．NT 合金の場合，変態点付近でマルテンサイト変態による特異な挙動を示す．加熱の際はほぼ直線的に電気抵抗値は変化するが，高温のオーステナイト相から温度を下げると特定の温度で急激に上昇し，ピークを経過した後，急傾斜で減少しもとの直線上に戻る．高温から温度を下げた際の曲線におけるピークの立ち上がり点が M_s 点に，低温度の変曲点が M_f 点に対応する．また，加熱時のピークが A_f 点である．

3　形状記憶合金のモデル化

3.1　温度と電圧の関係

SMA は温度によって発生する力が変化する．本実験では，ジュール熱を利用し電気で SMA を駆動させるロボットを開発する．ジュール熱を求めるため，SMA の抵抗値を導出する．SMA アクチュエータのモデル図を図 4 に示す．

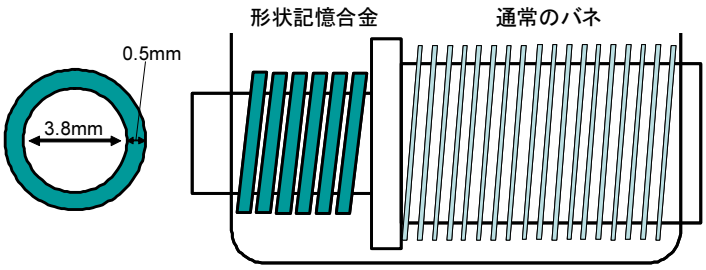

図 4: SMA モデル図

SMA の線径は 0.50[mm] であり，バネ状にしたときのバネの直径は 4.30[mm] である．バネのピッチ角を 0 とすれば，バネの巻き数は約 5 巻であるので線の長さ L は

$$\begin{aligned} L =& 4.30 \times \pi \times 5 \\ =& 21.5\pi[\mathrm{mm}] \end{aligned}$$

である．また，断面積 A は

$$\begin{aligned} A =& 0.50^2 \times \pi \\ =& 0.25\pi[\mathrm{mm}^2] \end{aligned}$$

となる．今回用いる NT 合金の比抵抗 ρ は 500〜1100[$\mu\Omega$·mm] であるので，抵抗の公式 $R = \rho\frac{L}{A}$ より SMA の抵抗 R は

$$R = 43.0 \sim 94.6[\mathrm{m}\Omega]$$

と求めることができる．

この SMA に t 秒の間電流 I[A] を流したとすると，SMA の発熱量 Q は

$$Q = \frac{RI^2t}{4.2}[\mathrm{cal}]$$

となり，SMA の温度 T は

$$\begin{aligned} T =& \frac{RI^2t}{4.2m} \\ =& \frac{V^2t}{4.2Rm}[℃] \end{aligned} \tag{1}$$

と求めることができる．ただし，V は SMA にかかる電圧 [V]，m は SMA の質量で約 0.34[g] である．

3.2　温度と発生力の関係

SMA の温度と，発生力の関係は変位が大きい場合に対しては非線形の特性を示す．しかし，今回のアクチュエータとして用いた SMA は形状が小型であり，変位も微小である微小変形であると考えられる．この微小変形における温度と発生力との関係は，図 5 で示した特性のように線形で近似できる [12]．したがって，アクチュエータ全体の温度と発生力の関係も線形として扱うことができる．

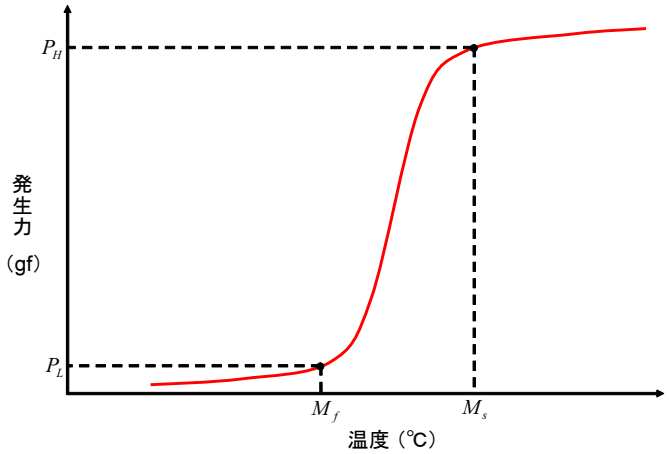

図 5: 形状記憶合金コイルバネの温度－発生力曲線 [12]

アクチュエータ全体を考えると形状記憶合金の温度が低い場合，形状記憶合金コイルバネは完全に縮んでいるため発生力は 0[gf] と考えられる．また，温度が高い場合にはアクチュエータの最大応力である 60[gf] の発生力が発生すると考えられる (今回使用したアクチュエータの仕様に基づく)．形状記憶合金が変形する温度範囲が 30(M_f)〜50(M_s)[℃] であるため [13]，アクチュエータ全体を線形モデルであると考えると図 6 の特性となる．

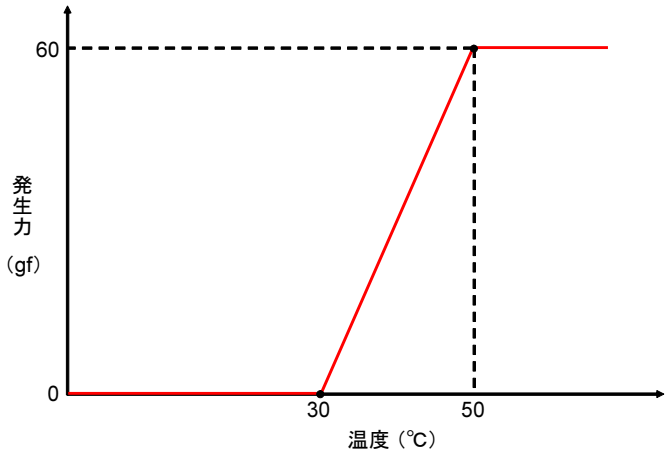

図 6: アクチュエータの温度－発生力近似曲線

図 6 から，発生力 F_s，温度 T として数式モデルを導出すると

$$F_s = \begin{cases} 0 & T < 30 \\ 3T - 90 & 30 \leq T \leq 50 \\ 60 & 50 < T \end{cases} \tag{2}$$

となる．

3.3　発生力と変位の関係

図 7 にロボット前進時の簡易モデル図を示す.

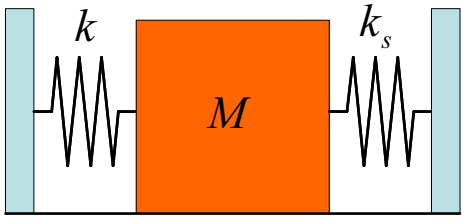

図 7: ロボット簡易モデル図

通常バネのバネ定数を k，ロボットの質量を M，SMA のバネ定数を k_s，SMA が発生する力を F_s とする，とこのモデルの運動方程式は

$$M\ddot{x} = kx - k_s x$$
$$M\ddot{x} = kx - F_s \tag{3}$$

とあらわすことができる．ただし，$k = 0.039$[N·mm] であり，最大変位は 5[mm] である．(3) は (1)，(2) より，

$$M\ddot{x} = kx - f(T)$$
$$M\ddot{x} - kx = \begin{cases} 0 & T < 30 \\ 90 - \frac{3V^2 t}{4.2Rm} & 30 \leq T \leq 50 \\ -60 & 50 < T \end{cases}$$

と表すことができ，電圧を入力とし，変位を出力とした運動方程式を導出することができた.

4　実験環境の構築

実験に用いた SMA，素子の詳細を表 1 に示す.

表 1: 実験機器・試料

名称	型番	シリアルナンバー	メーカ
バイアス式 2 方向性アクチュエータ	———	———	相互発條株式会社
温度センサ	LM35	———	National Semicondutor
距離センサ	GP2D12	———	SHARP
パルス幅変調制御 IC	TL1451ACN	———	TEXAS INSTRUMENTS
モータドライバ	TA8429HQ	———	TOSHIBA
直流電源	IPS-606D	504D160G1	ISO-TECH
AC アダプタ (9V)	NP12-1S0523	———	秋月電子通商
AC アダプタ (5V)	NP12-1S0912	———	秋月電子通商
高速 16 ビット AD/DA 変換 PC カード	CSI-360116	0178775229	Interface
68 ピン 0.8mm ピッチケーブル	CWB-5612	NR0821030	Interface
端子台	TNS-6811	0475924349	Interface
ノート PC	G50	L3-A4317	IBM
テスタ	DL-97	7050006	KENWOOD

- 抵抗：4.7k Ω × 1，62k Ω × 1
- 金属皮膜抵抗：75 Ω × 1
- 積層セラミックコンデンサ：15000pF × 1
- セメダイン エポキシパテ プラ用

4.1　マイクロロボットの製作

SMA バイアスアクチュエータを用いた移動型ロボットの機構について説明する．図 8 に CG イメージを示す．

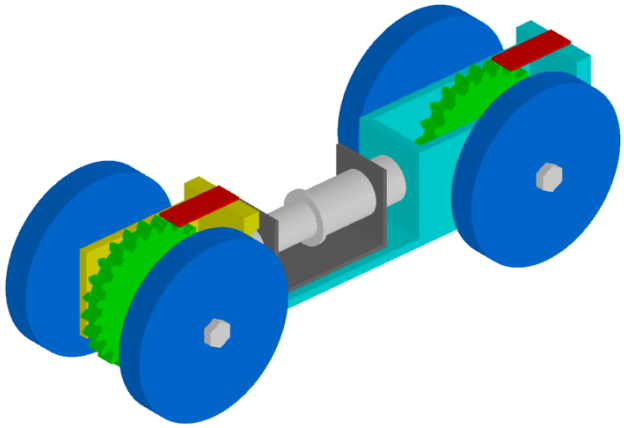

図 8: 全体図 CG イメージ

このロボットは，SMA バイアスアクチュエータの前後伸縮により，前進するロボットである．前後のユニットを SMA バイアスアクチュエータでつなぐ構造になっている．SMA バイアスアクチュエータの軸は前部のユニットに，金属フレームは後部のユニットに繋がっている．車輪は摩擦によって空転しないものとする．それぞれの車軸はラチェット機構となっており，一方向にしか回転しないようになっている．この機構を図 9 に示す．

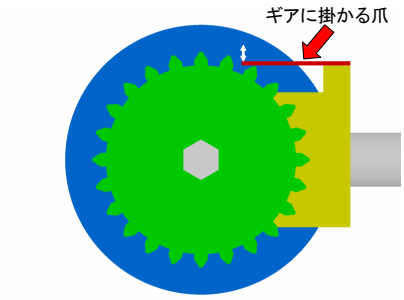

図 9: ラチェット機構

ラチェット機構は車軸についているギアと，フレームにはギアに掛かる爪が付いている．車輪が反時計回りする場合，ギアの歯が爪を上方向へ曲げるため，車輪は回転する．逆に，車輪が時計回りしようとすると，ギアの歯が爪に掛かるため，車輪は回転しない．この機構を利用した前後方向に進む原理について説明する．図 10 に SMA の伸縮ステップごとの図を示す．

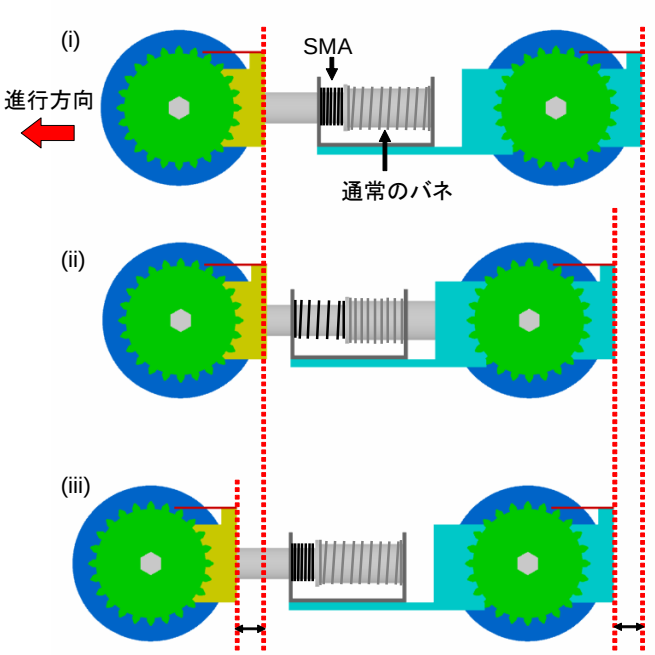

図 10: 動作ステップ

まず，(i) の状態を初期状態とし，SMA は通常のバネの力によって収縮状態にある．初期状態から，SMA は加熱されることにより伸びる．このとき，前輪の車輪は時計回りに回転しないため，車輪の摩擦によって位置は固定される．このため，後輪が回転し，ロボットは収縮した (ii) の状態になる．(ii) の状態から，SMA は冷やされることにより通常のバネの力によって収縮する．このとき，後輪の車輪は時計回りには回転しないため，車輪の摩擦によって，位置は固定される．このため，前輪が回転し，ロボットはアクチュエータの変位分だけ移動した (iii) の状態になる．このサイクルを繰り返すことにより，ロボットは前進運動を行う．

この原理を基にを持つロボットの製作を行った．製作したロボットを図 11 に示す．

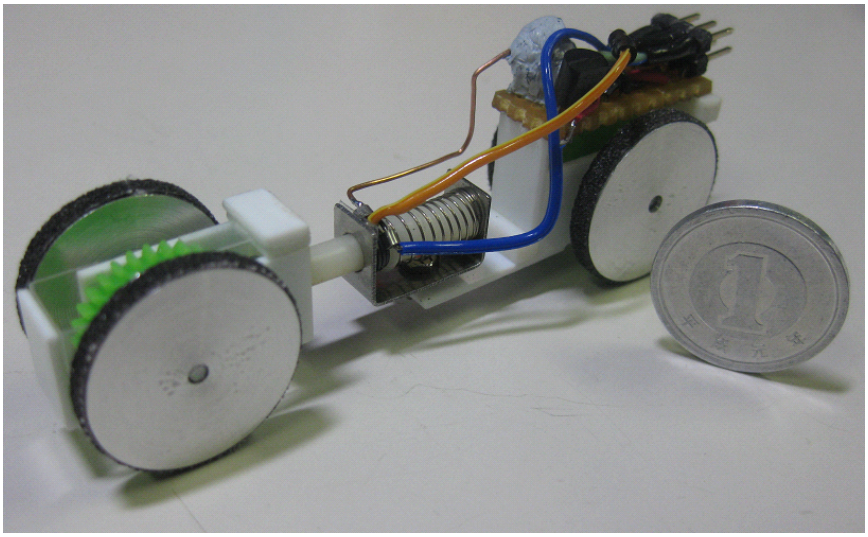

図 11: 製作した実機

このロボットの前後フレームは，プラ板・プラ角材を用いて製作した．車輪はアルミで製作し，直径

20[mm]・厚さ 3[mm] で，車輪表面には摩擦を増すための滑り止め材を貼り付けてある．ギアにはタミヤ製の歯数 30 のものを用いた．また，ロボット本体の回路を図 12 に示す．

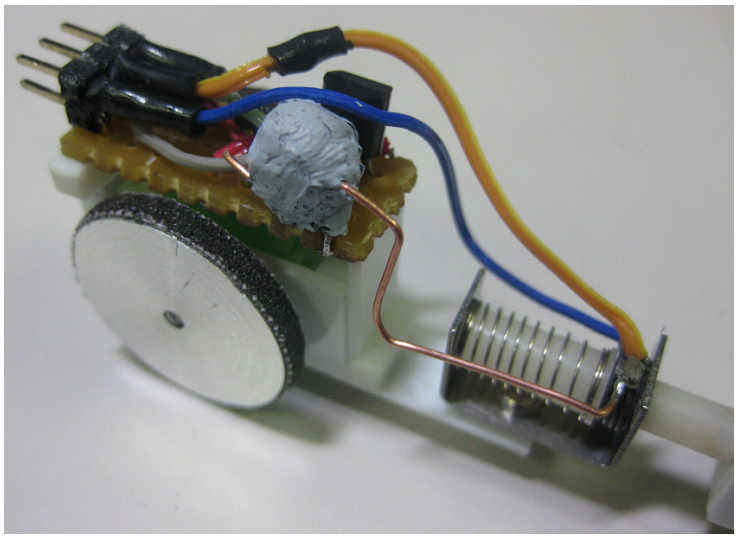

図 12: ロボットの回路

ロボット本体の回路には，温度センサがあり (詳細は 4.2.3 で示す)，SMA の片端に接続された銅線を温度センサに巻きつけエポキシパテで覆い外気温の影響を抑える．また，SMA の両端には青線と橙線によって電気が流れるようになっている．

4.2 回路装置の設定

4.2.1 PWM 生成回路

制御には PWM を用いて電圧制御を行う．TL1451 は内部で三角波を生成し，入力電圧とコンパレートすることでデューティ比を決定することのできる IC である．TL1451 データシート [14] の図 1 より図 13 のテスト回路を製作し，周波数を調べた．

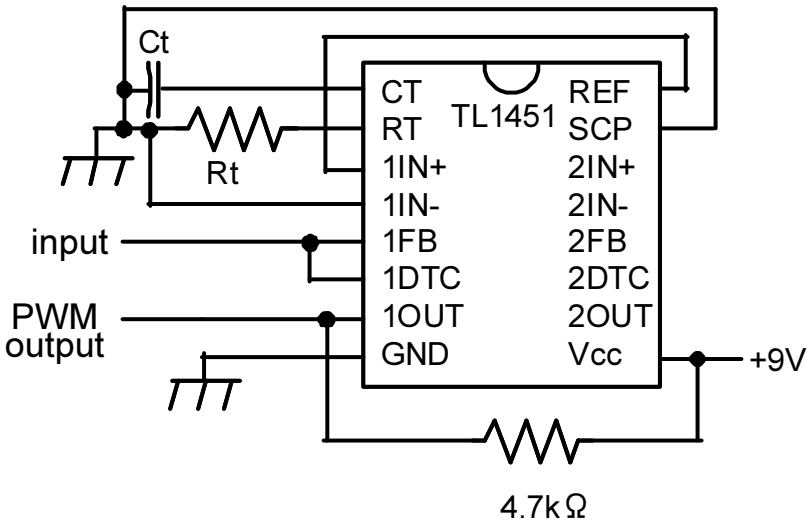

図 13: TL1451 テスト回路 [14]

三角波は 1.4〜2.0[V] の幅で発生する．入力電圧を 1.4[V] とすればデューティ比は 0[%]，1.7V とすれば 50[%]，2.0[V] とすれば 100[%] となる．周波数は Ct と Rt の組み合わせによって変化する．TL1451 データシートの図 3 より周波数を 1[kHz] と設定するために，Ct=15000[pF]，Rt=62[k Ω] と設定した．入力電圧を 1.6[V] とした際の出力波形を図 14 に示す．

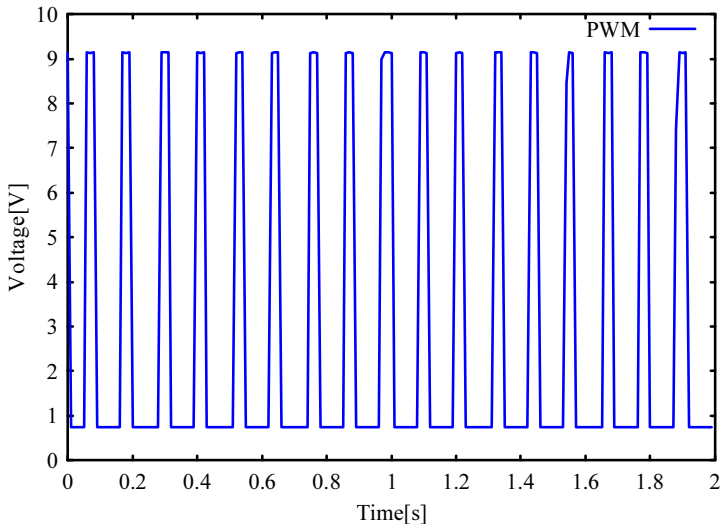

図 14: 1.6[V] を入力としたときの PWM 出力波形

4.2.2 距離測定センサ

マイクロロボットの移動距離を測定するために距離センサを用いる．距離センサにはSHARP の GP2D12 を用いる．このセンサはデータシート [17]8 ページのように距離に応じて電圧を出力する．実測した電圧とセンサ距離の関係を表 2，図 15 に示す．

表 2: GP2D12 の距離と電圧の関係

距離 [cm]	電圧 [V]	距離 [cm]	電圧 [V]
1.5	0.03	12.0	2.396
2.0	0.694	13.0	2.206
2.5	0.851	14.0	2.05
3.0	1.189	15.0	1.919
3.5	1.098	16.0	1.793
4.0	0.791	17.0	1.714
4.5	1.173	18.0	1.623
5.0	1.439	19.0	1.53
5.5	1.714	20.0	1.456
6.0	1.758	21.0	1.378
6.5	1.842	22.0	1.322
7.0	2.009	23.0	1.266
7.5	2.32	24.0	1.21
8.0	2.691	25.0	1.16
8.5	2.809	26.0	1.116
9.0	2.825	27.0	1.079
9.5	2.809	28.0	1.04
10.0	2.742	29.0	1.003
11.0	2.573	30.0	0.969

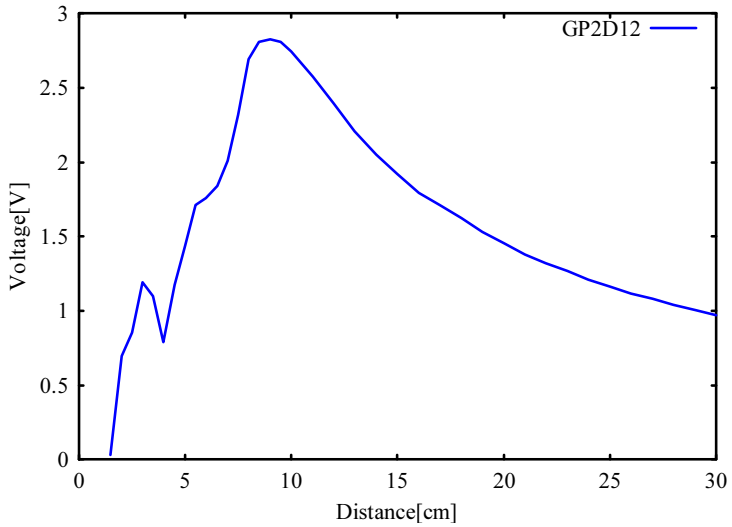

図 15: GP2D12 の距離と電圧の関係

図 15 のように 10[cm] 以下では電圧は線形ではないので，実験では 10[cm] 以上を用いる．10〜20[cm] の距離 L と電圧 V_d の近似式を得るために最小二乗法を用いる．距離 L と電圧 V_d は反比例の関係であるため，

$$V_d = \frac{a}{L} + b$$

と考えることができる．

しかし，このままでは最小二乗法を用いて近似することができないので，

$$Z = \frac{1}{L}$$

とおくと

$$V_d = aZ + b$$

となり，一次関数の式に書き換えることができる．下記の式より，係数 a，b を計算すると，

$$a = \frac{n\sum_{i=1}^{n} Z_i V_{di} - \sum_{i=1}^{n} Z_i \sum_{i=1}^{n} V_{di}}{n\sum_{i=1}^{n} Z_i^2 - \left(\sum_{i=1}^{n} Z_i\right)^2}$$
$$b = \frac{\sum_{i=1}^{n} Z_i^2 \sum_{i=1}^{n} V_{di} - \sum_{i=1}^{n} Z_i V_{di} \sum_{i=1}^{n} Z_i}{n\sum_{i=1}^{n} Z_i^2 - \left(\sum_{i=1}^{n} Z_i\right)^2}$$

$$a \approx 29.5, \quad b \approx 0.434$$

となり，距離 L と電圧 V_d の近似式を

$$\begin{aligned} V_d &= \frac{29.5}{L} + 0.434 \\ L &= \frac{29.5}{V_d - 0.434} \end{aligned} \tag{4}$$

と求めることができる．

4.2.3 温度測定センサ

SMA の温度を測定するために温度センサを用いる．温度センサには National Semiconductor の LM35 を用いる．このセンサの温度係数はリニアで+10.0[mV/℃] である [16]．よって温度 T と出力電圧 V_{temp} の関係は

$$T = 10.0V_{temp} \tag{5}$$

となる．センサ確認のため，温度センサによる室温の計測を行った．計測結果を図 16 に示す．

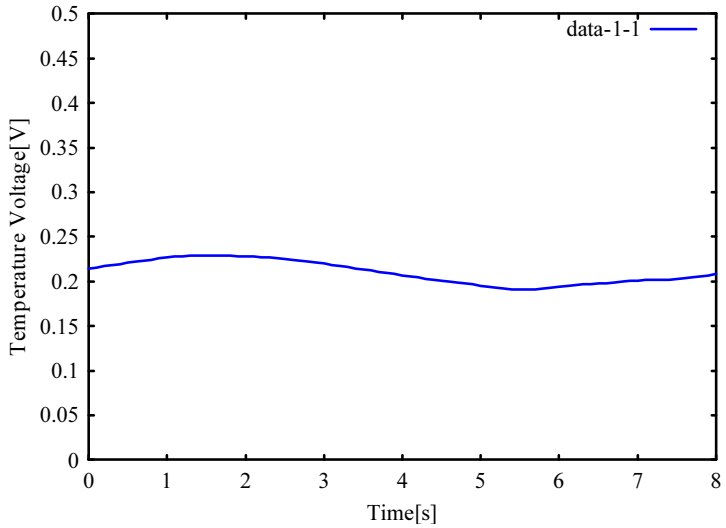

図 16: 温度センサによる室温計測結果

図 16 より，温度センサからの出力電圧は約 0.21[V] であり，(5) の温度センサ電圧－温度変換公式を用いることで，温度センサを用いての室温計測結果は約 21[℃] となる．また，他の室温計での測定結果も約 21 ℃を示していたことから，温度センサからの測定値は正確に取得できていることが確認できる．

4.3　制御，計測回路の製作

回路構成図を図 17 に，製作した回路を図 18 に示す．

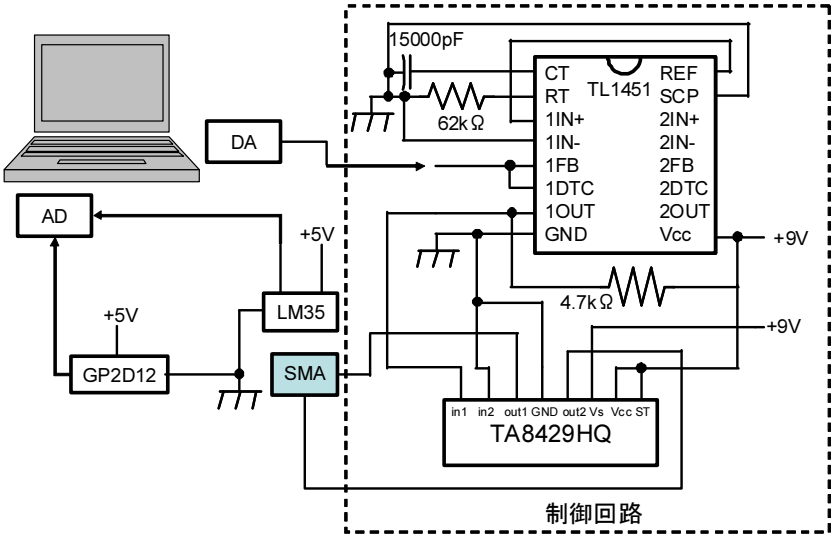

図 17: 回路構成図

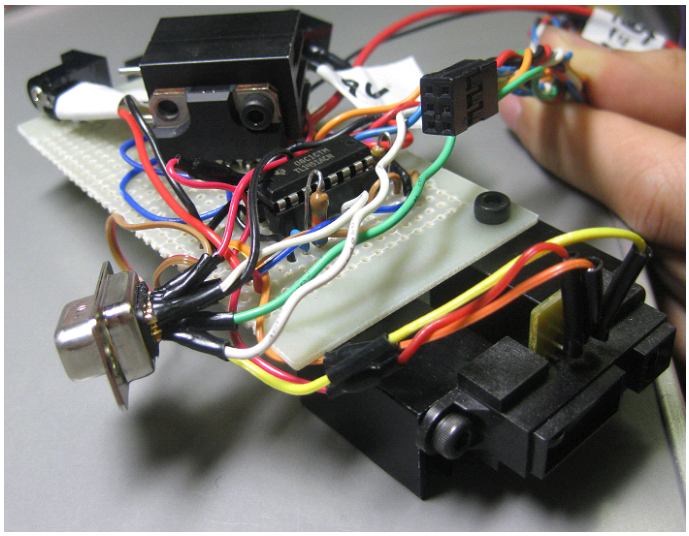

図 18: 製作した回路

回路は PC からの指令電圧に応じて PWM を発生させるものである．生成した PWM をモータドライバを用いて増幅し，SMA に加える．モータドライバには TOSHIBA の TA8429HQ を用いた．本来は IN1, IN2 の正負でモータの正転逆転を行う IC であるが，今回は増幅にのみ用いるため IN2 は接地させた．また，トランジスタではなくモータドライバを用いた理由は，大電流が流れることが予想され，それに耐えうるトランジスタを入手することが困難であったため，出力電流が 3.0[A](AVE.)4.5[A](PEAK)[15] と大容量な TA8429HQ を用いた．

4.4 実験環境

4.1，4.2，4.3 より構築した実験環境の全体図を図 19 に，実際の実験環境を図 20 示す．

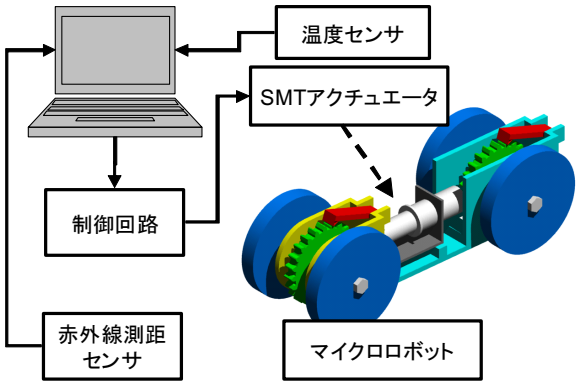

図 19: 実験環境

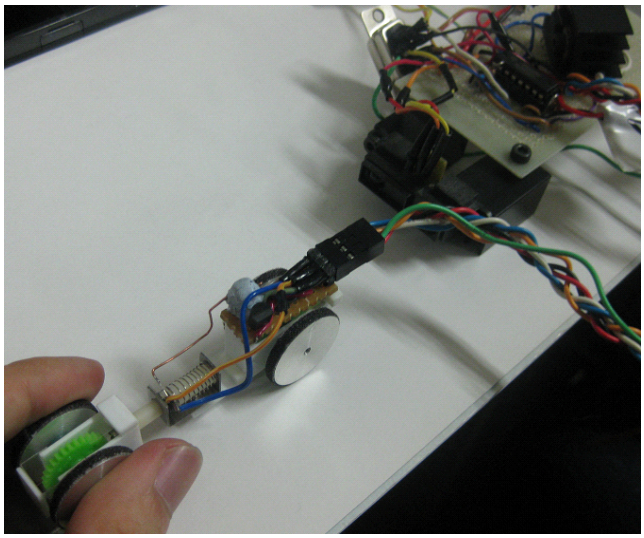

図 20: 実際の実験環境

温度センサの出力を AD で取得し，それに応じて PWM のデューティ比を決定する．デューティ比に対応する電圧を DA 変換を用いて電圧出力し，モータドライバを通して SMA に電圧をかける．マイクロロボットが動いた距離は距離センサを用いて取得する．

5　実験内容と結果

5.1　温度センサを用いた計測

図 19，図 20 の実験環境を用いてマイクロロボットを駆動させた．各 IC への入力電圧は図 17 の通りである．入力には，テスト用入力として図 21 のようなステップ入力を与えた．DA からの出力電圧を図 21 に，モータドライバ入出力電圧を図 22 に，距離センサの出力を図 23 に，温度センサの出力を図 24 に示す．

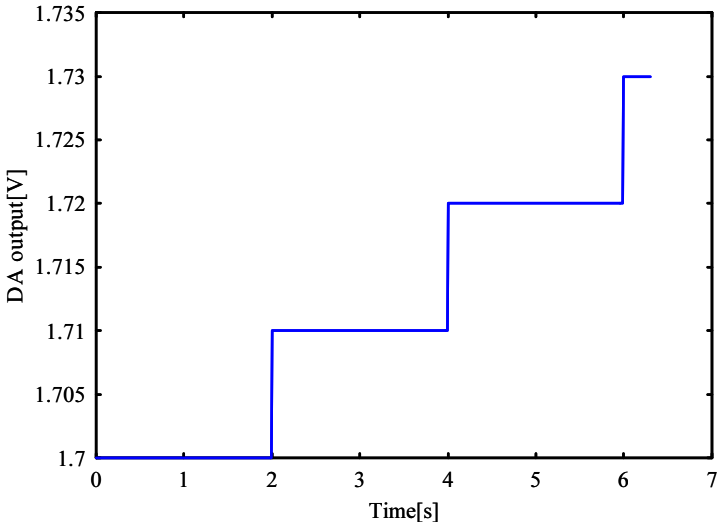

図 21: DA 出力電圧

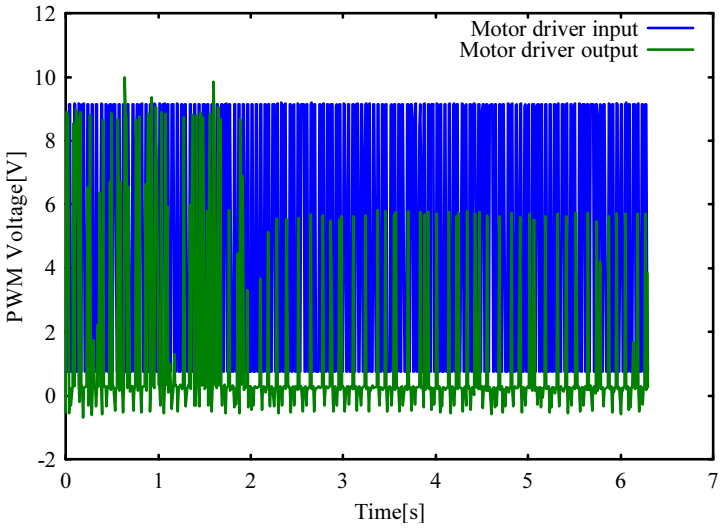

図 22: モータドライバ入出力

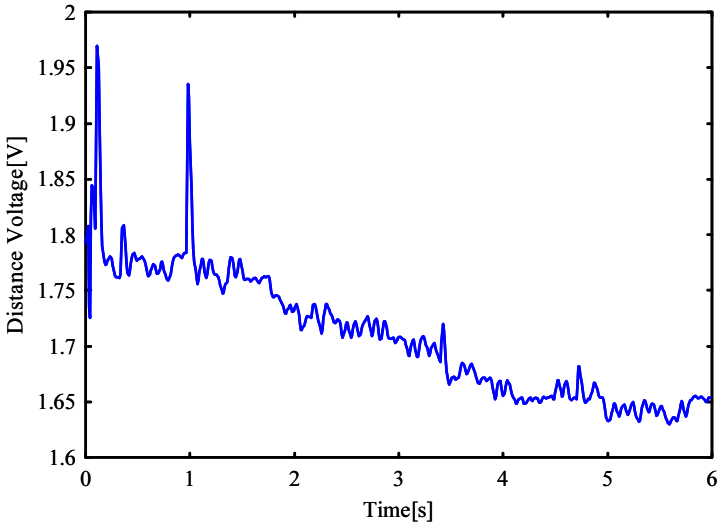

図 23: 距離センサ出力

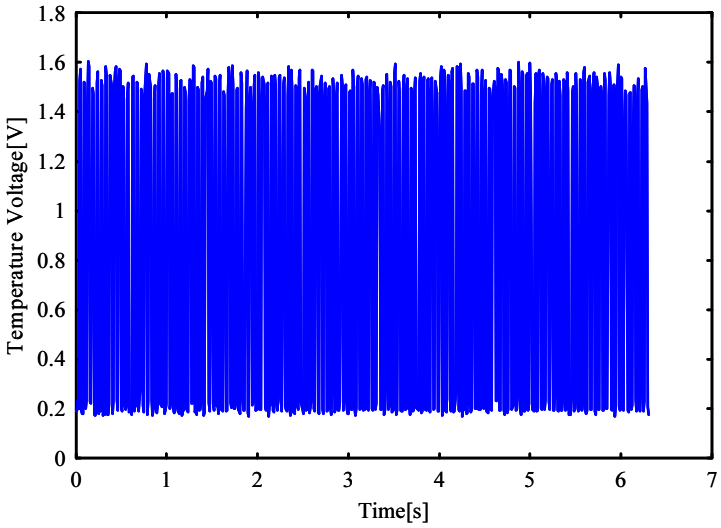

図 24: 温度センサ出力

図 22 緑線のような電圧を SMA に加えたとき，図 23 より距離センサの出力電圧は約 1.78[V] から 1.65[V] まで変化しており，(4) より約 1[cm] 移動していることがわかる．しかし，図 24 では 0.2[V] から 1.8[V] まで値が乱れており，正確な温度が計測できていない．4.2.3 より，温度センサによる室温の測定は正確に行えていたことから，センサ自体の問題ではないことがわかる．考えられる問題としては,「銅線を温度センサに巻きつけたために電流により磁場が生じ，センサに影響をあたえている」,「温度センサが取り付けられている基板，ケーブルに大電流が流れるためノイズが発生しセンサに影響を与えている」という事柄が挙げられる．銅線のコイルをセンサから外した状態で計測した結果より，図 24 と同様の結果となった．よって後者が問題であると判断したが，別基板を載せるスペースが確保できない，直径 0.5[mm] 程度のワイヤの温度の計測が難しいなどの問題から，次の方法を用いてマイクロロボットをコントロールすることとした．

距離センサをマイクロロボットの前後に取り付け，その変化によって SMA の状態を推定し，入力電圧を決定する．SMA の駆動には先の実験と同様にジュール熱を利用した駆動方法を用いる．

5.2 距離センサを用いた計測

距離センサを 2 つ用いたときの実験環境を図 25 に示す.

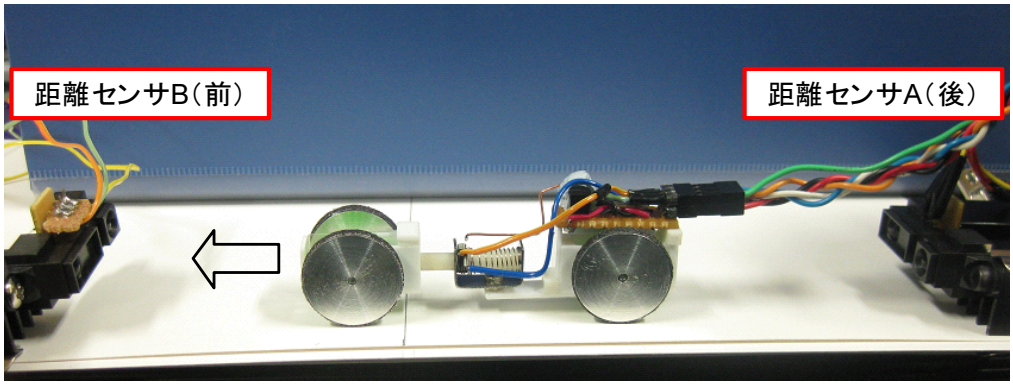

図 25: 距離センサを用いた制御の実験環境

マイクロロボットは図 10 のように動くため，1 回の加熱で移動できる距離が決まっている．1 回の加熱でギア $\frac{3}{30}$ 枚分タイヤが回転する．タイヤの直径は 20.0[mm] であるため，1 回の加熱で最大約 6.28[mm] 移動する．そこで，距離センサ A の値が加熱開始時状態よりも $6.28 \times \frac{2}{3}$[mm] 分増加したら加熱を停止，そのときの値を保持する．距離センサ B の値が冷却開始状態よりも $6.28 \times \frac{2}{3}$[mm] 分減少したら加熱を開始し，そのときの値を保持するといった ON/OFF 制御を行う．DA からの出力電圧を図 26 に，モータドライバ入出力を図 27 に，距離センサ A の出力を図 28 に，距離センサ B の出力を図 29 に示す.

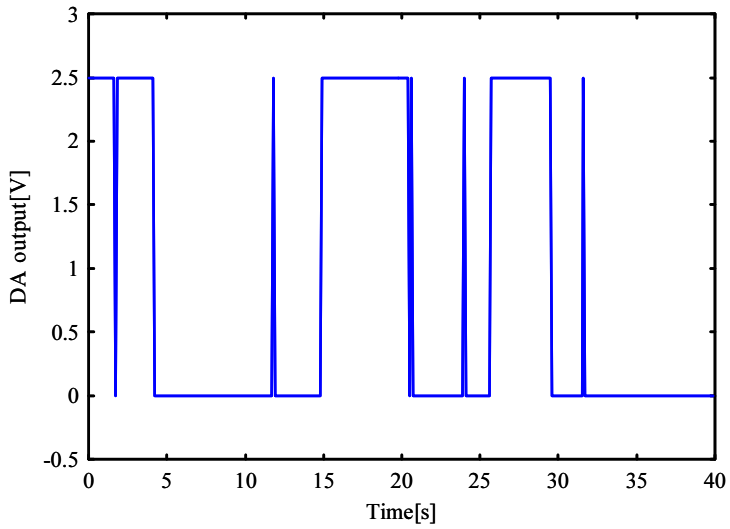

図 26: DA からの出力電圧

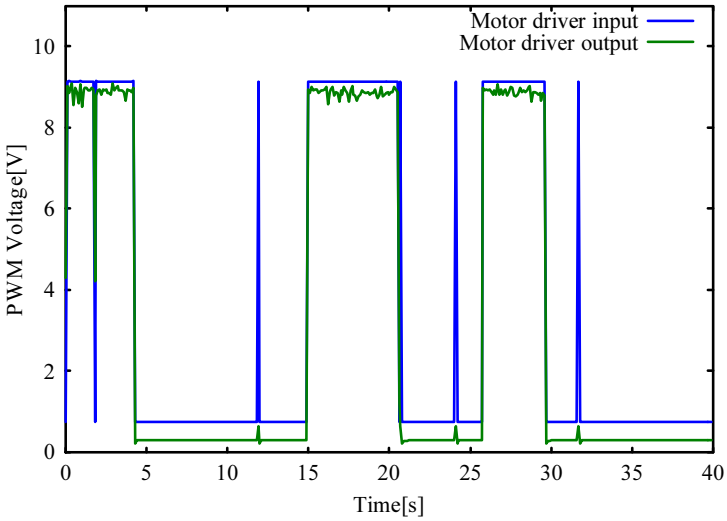

図 27: モータドライバ入出力

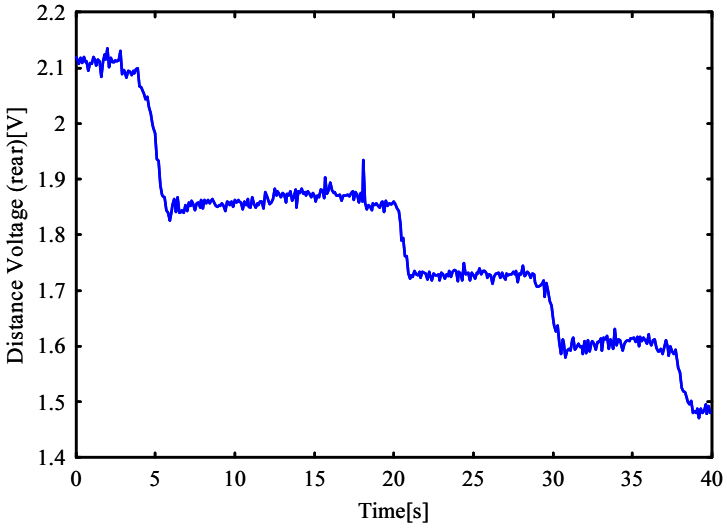

図 28: 距離センサ (後) 出力

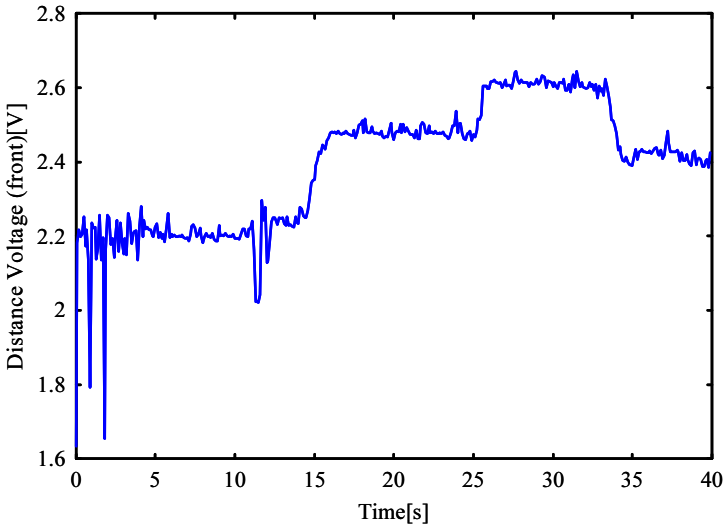

図 29: 距離センサ (前) 出力

ジュール熱を用いて SMA を駆動させることにより，マイクロロボットが前進することを確認した．また，図 28 より距離センサ (後) の電圧は下がり，図 29 より距離センサ (前) の電圧はあがっていることからも少しずつロボットが前進していることがわかる．図 29 の 33〜40[s] では電圧が下がっているが，これは図 15 の特性より，約 10[cm] 以下にマイクロロボットが入ってしまったからである．また，波形に多少ノイズが発生しており，図 26 の入力に影響を与えている．これにより，多少入力が触れているが微小であるため，コントロールにはほとんど影響は出ていない．

6　まとめ

形状記憶合金の性質・原理について調査し，調査結果を基に形状記憶合金およびアクチュエータのモデルを導出した．また，導出したモデルを参考に，形状記憶合金の性質を利用した制御系設計を行なうとともに，形状記憶合金をアクチュエータとしたマイクロロボットおよび実機制御用回路の設計・製作を行った．これらを用いて実際に制御を行ったところ，温度センサ周辺に発生したノイズの影響により，温度センサを用いた制御は困難であるとの結論を得た．この問題に対し，温度センサを排除し距離センサのみを用いた制御系を考案した．この制御系を用いて実機による検証を行うことにより，実際にマイクロロボットの動作を確認するとともに，今回考案したマイクロロボットおよび形状記憶合金アクチュエータの制御系の有効性を示した．

今後の課題としては，温度による抵抗値の変化や温度と発生力のモデルについて再度検証を行い，これらを用いた制御系設計およびロボット設計を行なう．また，今回の実験において行うことのできなかった温度センサによる温度制御の問題を解決し，形状記憶合金の温度－応力特性を有効に活用する形での制御を行う．

謝辞

本実験は株式会社八光電機製作所熱の実験コンテストにおいて助成を賜りました．また，このような貴重な場を設けていただき，ここに深甚なる謝意を表します．

本実験を行うにあたり，多大なご助力を与えて下さった本学の先生方，研究室員に対し深く感謝を申し上げます．

参考文献

[1] 高橋 昌樹，林 巖，岩附 信行，鈴森 康一，黄木 昇：『みみずの運動を応用した細管内移動マイクロロボットの研究』，精密工学学会，Vol.61，No.1，1995

[2] 大野 学，加藤 充宏，青木 裕一，新井 徹，加藤 重雄：『柔軟な細い管を走行できるマイクロロボットの試作』，日本機械学会関東支部ブロック合同講演会，No.231，1995

[3] 大野 学，笹崎 智哉，秋葉 崇志，加藤 重雄：『豚の腸を走行できる管内マイクロロボットの研究』，日本機械学会関東支部ブロック合同講演会，No.236，1995

[4] 大野 学，泉 俊之，濱野 聡明，加藤 重雄：『気液相変化アクチュエータを動力源とする管内走行マイクロロボット』，日本機械学会，No.817，2005

[5] 石川 敏也，中田 毅：『人口筋肉を目指した形状記憶合金アクチュエータ (巻フィルムチューブ方式の提案)』，日本機械学会論文集 (C 編)，71 巻，703 号，2005

[6] 寺内 美奈，前場 恒太，島田 明：『形状記憶合金とモータを併用したロボット指の実現』IEEJ Trans.IA，Vol.128，No.5，2008

[7] 山下 次郎，間瀬 正一，石黒 浩三，守谷 亨，遠藤 裕久，松原 武生，藤田 英一，平林 真，鈴木 平，鈴木 秀次，高村 仁一，二宮 敏行：『金属の物理的性質』 日本物理学会編，東京 裳華房，昭和 47 年 6 月 3 日 第 2 版発行

[8] 田中 喜久昭，戸伏 壽昭，宮崎 修一：『形状記憶合金の機械的性質』，養賢堂，1993 年 7 月 30 日 第 1 版発行

[9] 宮崎 修一，佐久間 俊雄，渋谷 壽一：『形状記憶合金の応用展開』，シーエムシー出版，2006 年 10 月 24 日 普及版 第 1 版発行

[10] C.M.WAYMAN 著，清水 謙一 訳：『マルテンサイト変態の結晶学』，丸善株式会社，昭和 44 年 3 月 30 日 第 1 版発行

[11] 川村 貞夫，野方 誠，田所 諭，早川 恭弘，松浦 貞裕：『制御用アクチュエータの基礎』，コロナ社，2006 年 4 月 28 日 第 1 版発行

[12] 鈴木 雄一：『実用形状記憶合金』，工業調査会，1987 年 10 月 1 日 第 1 版発行

[13] 戸伏 壽昭，田中 喜久昭，堀川 宏，松本 實：『形状記憶材料とその応用』，コロナ社，2004 年 6 月 28 日 第 1 版発行

[14] 『DUAL PULSE-WIDTH-MODULATION CONTROL CIRCUITS TL1451 データシート』，"http://www.datasheetcatalog.com/datasheets_pdf/T/L/1/4/TL1451.shtml"，Texas Instruments

[15] 『DC モータ用フルブリッジドライバ TA8429HQ データシート』，"http://www.technobase.jp/eclib/OTHER/DATASHEET/ta8429hq.pdf"，TOSHIBA

[16] 『高精度・摂氏直読温度センサ IC LM35 データシート』，"http://www.national.com/JPN/ds/LM/LM35.pdf"，Natoinal Semiconductor

[17] 『測距センサユニット GP2D12 データシート』，"http://www.mech.tohoku-gakuin.ac.jp/rde/contents/tech/h8train/gp2d12_.pdf"，SHARP

Appendix A 駆動プログラム

Appendix A.1 ライブラリ

Appendix A.1.1 AD ライブラリ

```
#include <windows.h>
#include <stdio.h>
#include <tchar.h>
#include "Gpcad.h"
#include "Fbiad.h"

#pragma comment(lib, "FbiAd.lib")
#pragma comment(lib, "FbiAdDC.lib")

extern "C" int __cdecl AD_Open(void);
extern "C" int __cdecl AD_Close(void);
extern "C" int __cdecl AD_Get(int);

HANDLE AdDeviceHandle;
ADSMPLCHREQ AdSmplChReq[4];
WORD AD_Data[4];

////////////AD デバイスオープン/////////
int AD_Open(void){
AdDeviceHandle = AdOpen(reinterpret_cast<LPCTSTR>("FBIAD1"));
//失敗した場合には、INVALID_HANDLE_VALUE(FFFFFFFFh) が返される

AdSmplChReq[0].ulChNo = 1; //入力チャンネル
AdSmplChReq[0].ulRange = AD_10V; //電圧レンジ (-10〜10V)
AdSmplChReq[1].ulChNo = 3; //入力チャンネル
AdSmplChReq[1].ulRange = AD_10V; //電圧レンジ (-10〜10V)
AdSmplChReq[2].ulChNo = 5; //入力チャンネル
AdSmplChReq[2].ulRange = AD_10V; //電圧レンジ (-10〜10V)
AdSmplChReq[3].ulChNo = 7; //入力チャンネル
AdSmplChReq[3].ulRange = AD_10V; //電圧レンジ (-10〜10V)

if(AdDeviceHandle == INVALID_HANDLE_VALUE){
printf("AD Device Open:\t\t Miss\n");
return -1; //失敗
}
else{
printf("AD Device Open:\t\t Success\n");
return 0; //成功
}
}

////////////AD デバイスクローズ/////////
int AD_Close(void){
int msg;
msg = AdClose(AdDeviceHandle); // デバイスクローズ
//正常終了すると AD_ERROR_SUCCESS を返す

if(msg == AD_ERROR_SUCCESS){
printf("AD Device Close:\t Success\n");
return 0; //成功
}
else{
printf("AD Device Close:\t Miss\n");
return -1; //失敗
}
```

```
}

//////////////AD デバイスデータ取得////////
int AD_Get(int ch){
int msg;

msg = AdInputAD(AdDeviceHandle, 1, AD_INPUT_SINGLE, &AdSmplChReq[ch-1], &AD_Data[ch -1]);
//(デバイス名, チャンネル数, 入力方式 [シングルエンド], アナログ入力を行うチャンネル番号, データ格納位置へのポイ
ンタ)
//正常に終了すると AD_ERROR_SUCCESS を返します。

if(msg != AD_ERROR_SUCCESS){
printf("AD デバイスデータ取得:\t 失敗%x\n",msg);
return -1;
}
else{
return AD_Data[ch - 1];
}
}
```

Appendix A.1.2 DA ライブラリ

```
#include <windows.h>
#include <stdio.h>
#include "Gpcda.h"
#include "Fbida.h"
#include <tchar.h>

extern "C" int __cdecl DA_com_Open(void);
extern "C" int __cdecl DA_com_Close(void);
extern "C" int __cdecl DA_com_Out(WORD);

#pragma comment(lib, "FbiDa.lib")
#pragma comment(lib, "FbiDaDC.lib")

/////////グローバル変数///////
HANDLE DacomDeviceHandle; // DA デバイスハンドル
DASMPLCHREQ DacomSmplChReq[1];

//////DA デバイスオープン/////
int DA_com_Open(void){
DacomDeviceHandle = DaOpen(reinterpret_cast<LPCTSTR>("FBIDA1"));
//オープンに失敗した場合には、INVALID_HANDLE_VALUE(FFFFFFFFh) が返されます。

DacomSmplChReq[0].ulChNo = 1; // 出力チャンネル
DacomSmplChReq[0].ulRange = DA_10V; // 電圧レンジ (+-10V)
// DacomSmplChReq[1].ulChNo = 2;
// DacomSmplChReq[1].ulRange = DA_10V;

if(DacomDeviceHandle == INVALID_HANDLE_VALUE){
printf("DA_com Device Open:\t Miss\n");
return -1;
}else{
printf("DA_com Device Open:\t Success\n");
return 0;
}
}

//////////DA デバイスクローズ///////////////
int DA_com_Close(void)
```

```
{
int msg;
msg = DaClose( DacomDeviceHandle ); // デバイスクローズ
//が正常終了すると DA_ERROR_SUCCESS を返します。

if(msg == DA_ERROR_SUCCESS){
printf("DA_com Device Close:\t Success\n");
return 0;
}else{
printf("DA_com Device Close:\t Miss\n");
return -1;
}
}

//////////////DAOUT////////////////////
int DA_com_Out(WORD Output_1)
{
int msg;
WORD DA_Data[1]; // 出力データ格納用配列
DA_Data[0] = Output_1; // CH1
// DA_Data[1] = Output_2; // CH2

msg = DaOutputDA( DacomDeviceHandle, 1, &DacomSmplChReq[0], DA_Data);
//(デバイス名，チャンネル数，アナログ出力するチャンネルの番号，アナログ出力するデータが格納してある位置へのポイ
ンタ)
//正常終了すると DA_ERROR_SUCCESS を返します。

if(msg != DA_ERROR_SUCCESS){
printf("DA_com Output Error:\t%x\n",msg);
}
return msg;
}
```

Appendix A.2 駆動用プログラム

```
//---------------AD_lib 関数---------------
Integer AD_Open();
Integer AD_Close();
Integer AD_Get();

//---------------DA_lib 関数---------------
Integer DA_com_Open();
Integer DA_com_Close();
Integer DA_com_Out();

//---------------グローバル変数---------------
Real fro,bac;
Real dt, u, L, x;
Integer cnt, i, de;
Array T, D;
Matrix AD;

//*****************      main      *************************
Func void main()
{
void on_task(), break_task();

Integer flag, tmp;

dt   = 0.1; dt   = rtSetClock(dt);     //サンプリング周期
```

```
flag = 1; cnt = 0;
i = 0; de = 0;                          //初期化

L = 2.0*PI/10.0;                        //ロボットが 1 回の加熱で移動できる距離 [cm]

x = 1.0;                                //重み
read x;

T = []; D = [];                         //配列初期化
AD = [0.0, 0.0, 0.0, 0.0]';

rtSetTask(on_task);
rtSetBreak(break_task);

//---------------デバイスオープン---------------
tmp = AD_Open();
tmp = tmp + DA_com_Open();
if(tmp != 0)
{
print "プログラムを終了します. \n";
pause;
exit;
}

pause  "実時間処理を開始します. \n 何かキーを押してください. \n";

bac = AD_Get(1)*(20.0/65535.0)-10.0; //距離センサ後初期値
bac = 29.5/(bac - 0.434);

fro = AD_Get(2)*(20.0/65535.0)-10.0; //距離センサ前初期値
fro = 29.5/(fro - 0.434);

rtStart();                                  //実時間処理開始

printf("\n\n Esc : 停止\n");

while(flag){
if(kbhit()){
   switch(getch()){
     case 27: flag = 0;
break; //Esc を押したらブレイク
     default: break;
   }
}
else{
printf("Time: %f [s]\r", cnt*dt); //時間の表示
}
}

print "\n 実時間処理を終了しました. \n";
rtStop();                                   //実時間処理終了

//---------------デバイスクローズ---------------
AD_Close();

  DA_com_Out(32767); //0V 出力
DA_com_Close();

pause;
//---------------グラフ表示---------------
print [[T][D]] >> "log_T_D.txt";            //ログの保存
```

```
//---------------プログラム終了---------------
pause;
exit;
}

//*****************       実時間処理       *************************
Func void on_task()
{
AD = [  AD_Get(1)*(20.0/65535.0)-10.0,//距離センサ後ろ
AD_Get(2)*(20.0/65535.0)-10.0,//距離センサ前
AD_Get(3)*(20.0/65535.0)-10.0,
AD_Get(4)*(20.0/65535.0)-10.0]';

if(de == 0){
DA_com_Out(Integer( 2.5/10.0*32768 + 32767 ));
u = 2.5;
if( (29.5/(AD(1)-0.434)-bac) > L*x/3   ){//現在の距離-動き出すときの距離 > 2.0*PI*x/30(x=1to3)
de = 1;
bac = 29.5/(AD(1) - 0.434);
}
}
else{
DA_com_Out(Integer( 0.0/10.0*32768 + 32767 ));
u = 0.0;
if( (fro-(29.5/(AD(2)-0.434))) > L*x/3   ){// 動き出すときの距離-現在の距離 > 2.0*PI*x/30(x=1to3)
de = 0;
fro = 29.5/(AD(2) - 0.434);
}
}

T = [T,[dt*cnt]];  D = [D,[[AD][u]]];

cnt = cnt + 1;
}

//*****************       緊急停止       *************************
Func void break_task()
{
printf("\n\n 割り込み (Ctrl+c) により実時間処理を停止します. \n\n");
rtStop();
AD_Close();

DA_com_Out(32767); //0V 出力
DA_com_Close();
}
```