はじめに

手の上に棒を立てて遊ぶ、誰もが一度は経験している遊びだと思います。手に持てないような重い棒は論外として、それなりの長さの棒であれば少し練習することで簡単にバランスをとることができたと思います。もし、手元に適当な棒があれば、子供の頃を思い出してそれを立ててみてはいかがでしょうか。

この遊びは、棒を立てるという単純明快なものですが、結構子供にとっては楽しいものです(もしかしたら大人にとっても??)。その一つの理由が、棒が不安定であるということにあるのではないでしょうか。つまり手を動かさなければ倒れてしまうものを、棒を適切に動かすことによって立たせる。それによって人間本来が持つ征服欲が満たされる、というと少し大げさな言い方でしょうか。

ところで、棒を立てるという動作を**制御工学**(control engineering) 的な立場から少し考えてみましょう。棒を立たせるためにはいくつかの必要不可欠な要素があります。棒が必要であることは当然のことと思います(このように制御の対象となるものを**制御対象**(controlled object) といいます)。また、目的(**制御目的**(control objective)) も必要ですね。棒だけを与えられても何をしてよいのか困ってしまいます。この場合には、棒を立たせるということが目的となります。次に棒の現在の状態(たとえば傾きなど)を知る必要があります。このことは目(制御対象のある物理量を測定するものを**センサ**(sensor) といいます) を利用することで行えます。目を閉じても立たせることができる、という人もいるかもしれませんが、その人はおそらく手からの情報をうまく利用しているはずです。棒の状態をまったく知ることなくバランスをとることなど不可能であることは誰が考えても明らかです。このように目から取り込んだ情報に基づいて、次にどの様に手を動かすのかを判断することが必要になります。つまり頭です(この役目を果たすものを**制御器**(controller) といいます)。さらに指令通りに手を動かすためには力を発生す

るもの(**アクチュエータ**(actuator)) が必要です。人間の場合には筋肉です。以上が、棒を立たせるという動作に対して必要な要素です。
これらのことを**ブロック線図**(block diagram) と呼ばれる図で表現すると図 0.1 のようになります。

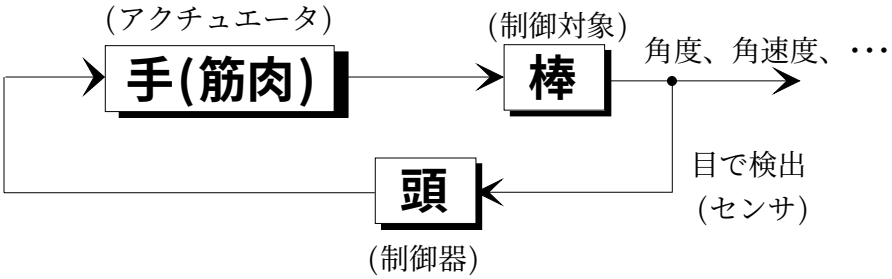

図 0.1 手の上に棒を立てる遊び

ここで注目してもらいたいことが一つあります。それは、ブロック線図中に一つのループが構成されているという点です。具体的にいうと、このループは、棒の現在の状況に従って筋肉を動かす指令が与えられることを意味しています。これを**フィードバックループ**(feedback loop) といいます。**制御** (control) の意味するところは後で説明しますが、棒を立たせる遊びのようにフィードバックを利用して制御を行う場合を**フィードバック制御** (feedback control) といいます。特に意識していないと思いますが、フィードバック制御は非常に身近なものです。たとえば、自動車や自転車の運転、ルームエアコン、こたつなどはフィードバック制御の例です(これらにおいてフィードバック制御がどの様に構成されているのかをぜひ考えてみてください。また、これら以外の例を考えてみてもよいでしょう)。

このように身の回りにフィードバック制御の例はたくさん見つけることができますが、決して "制御=フィードバック制御" ではありません。制御はもっと広い概念です。たとえば、日本工業規格(通称 JIS) には制御という用語が次のように定義されています。

> 『ある目的に適合するように対象となっているものに所用の操作を加えること』

これをブロック線図で表現すると、図 0.2 のようになります。フィードバック制御は、所用の操作を加えるために現在の対象の状態を利用する制御方法であるということができます。それでは例を通して、いくつかの制御の形態を説明しましょう。

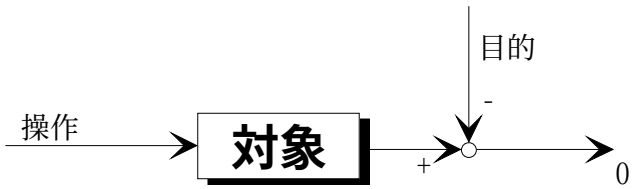

図 0.2「制御」のブロック線図

例題 0.1

洗濯を考えてみましょう。洗濯と制御を結び付けることに疑問を持つ方もいるかもしれませんが、『汚れた衣服』を対象として、『衣服を洗う』という所用の操作を行うことで『汚れを落とす』という目的を達成すると考えれば、それはまさしく制御と考えてもよさそうです。

洗濯は、最初は人間の手を使って行われていました(もちろん、この方法は現在でも使用されています)。これを**手動制御**(manual control) といいます。経験に基づいた優れた判断能力を利用できるという点では、良い成果を得ることができます(誰しもが洗濯名人ではありませんが)。しかし、人手と時間が必要です。これを回避する目的で登場したのが洗濯機です。人手を介す手動制御に対して、人手を介さない制御を**自動制御**(automatic control) といいます。現在の洗濯機はかなり自動化が進んでいることは事実ですが、まだまだ人間の経験に頼る部分も多くあります。その意味では、半自動制御という言い方が適切かもしれません。

ところで、洗濯機によっては、洗う時間、すすぎの回数、脱水の時間などをあらかじめ洗濯する人が希望(汚れ) に応じて選定することができます。この場合、洗濯機は汚れの状態がどのようであろうが、指定された順に自動的に動作を切り替えて洗濯を行います。このような制御のことを**シーケンス制御**(sequence control) といいます。最近では、汚れセンサを装備して、汚れの具合を判断する洗濯機も登場しているようです。これは、フィードバック制御に向けての発展といえるのかもしれません。ただし、フィードバック

制御はあくまでも一つの制御の形態であり、それがあらゆる場合にベストな制御方法であるというわけではないことに十分注意してください。たとえば、各種自動販売機や工場のオートメーションなどはシーケンス制御が活躍できる場です。

例題 0.2

陸上競技の一つの種目として 400m 競争があるのはご存知ですね。各競技者が与えられたコース上を走り抜けてタイムを競うものです。コースに従って走るという意味では、これは次の例で説明する追従制御として考えられます。たとえば、足元だけの情報から適切にフィードバック制御を施して走ることも可能ですが、そのように走っている競技者などはどこにもいません。目というセンサを利用してコースの状況(たとえばコーナの位置)をあらかじめ判断し、その情報をうまく活用しているはずです。このように先を見越して制御を行う形態を**フィードフォワード制御**(feedforward control)といいます。工場における無人搬送車やトレースロボット† の多くは足元の情報に基づくフィードバック制御ですが、CCD カメラを搭載して目を持たせることによりフィードフォワード制御が可能となります。

例題 0.3

家庭用空調機の目的は、様々な**外乱**(disturbance:制御対象の状態を乱そうとするもの)に対して部屋の温度を決められた値に一定に保つことです。このような制御を目標が一定であるという意味で**定値制御**(constant-value control)といいます。一方、産業用のマニピュレータを考えた場合、手先の位置をあらかじめ決まった軌道に追従させることが目的となります。これを**追従制御**(tracking control)といいます。定値制御も一定値に追従させると考えれば、追従制御の一つと考えられるかもしれませんが、通常は分けて考えます。

† 毎年行われている全日本マイクロマウスコンテストの競技種目の一つとして決められたラインをトレースしてその時間を競うものがあります。

例題 0.4

最後に大学における講義を考えてみましょう。この場合、制御目的は『学生の知識を高めること』であり、所用の操作は『講義する』ことになります。もし、学生の知識レベルや学習能力が完全に把握できる(つまり、このように講義すれば必ず学生は理解できるという名(?)講義ができる) ならば、シーケンス制御でも十分に目的は達成できるかもしれません。しかし、学生の現状をより的確に知るためには、時折試験やレポートを課すことが必要となります。ただそれだけではフィードバック制御であるとはいえません。提出されたレポートをチェックして、十分理解されていないと思われる箇所については、再度指導する、これによってフィードバック制御が実現できます。このようにきめ細やかな対応ができれば学生の知識も確実に向上するでしょう。しかし、学生の意欲、レベルのばらつき、授業時間数等の問題で理想は理解できてもその実現が困難であるのが現状ではないでしょうか(教官側の独り言??)。

一口に制御といってもいろいろな形態があることは理解できましたか。制御目的に従って、どの形態を利用すべきなのかを十分に検討する必要があります。

ここで、棒を立てる遊びに話を戻します。この遊びは、これまでの説明からもわかるように、フィードバックを利用した手動制御[†] で定値制御に分類できると思います。時計の振り子を逆さまに立てる、という意味でこれを**倒立振子**(inverted pendulum) といいます。この倒立振子は、それを模擬した実験装置を構成するのが容易であり、本質的に不安定なものを制御によって安定化するという意味で制御目的がはっきりしているなどの点から、制御工学の分野ではよく知られた実験装置の一つです。制御理論を勉強し始めた初心者から、自ら構築した理論を検証する目的の研究者まで広く利用されています。ただ、いかに装置の構成が簡単だからといって、実際に製作するとなると、物品の選定から始まり、それらを発注し、加工組立が必要です。また、希望する性能をもつ装置となっているのかを検証しなければなりません。本倒立振子キット(以下、振子キット) の役割がここにあります。本振子キッ

[†] 手動制御は、基本的にフィードバックが前提となります

トを利用することで、これらのことから解放され、理論の勉強や研究に専念することが可能となります。

図 0.3 は本振子キットの概略を描いたものですが、ここで簡単に説明しておくことにしましょう。まず、手に相当するのが**台車**(cart) で、それを駆動する筋肉に相当するものとして **DC サーボモータ**(DC servo motor) を使用しています。台車とモータはベルトで接続されており、モータが回転することにより、台車がレールの上を左右に移動します。また、現在の状態を知るために 2 個の角度センサが取り付けられています。一つは振子の角度計測用でもう一つは台車の位置を計測するためのものです。人間の頭に相当するのが、ディジタル計算機(なお、これは振子キット外です) で、その上で制御理論に基づいて設計した制御器のプログラミングを行います。計算機はセンサから取り込んだ信号に対して、プログラムに従って台車の動かし方を計算し、それをモータに与えることで台車が動くといったフィードバック構成です。ところで、人間が手の上の棒を立てる動作を忠実に実現しようとすれば、少なくとも台車をある平面内で(人間は手を 3 次元的に動かしています) 自由に動かすことが必要になりますが、構成が非常に複雑になります。そこで、本質を損なわない程度に簡単化する目的で、振子は台車上に置かれた軸に対してのみ自由に回転できる構成とします。この場合、台車はその回転軸に対して一方向に動くことでバランスをとることができるのは直感的に理解できますね。

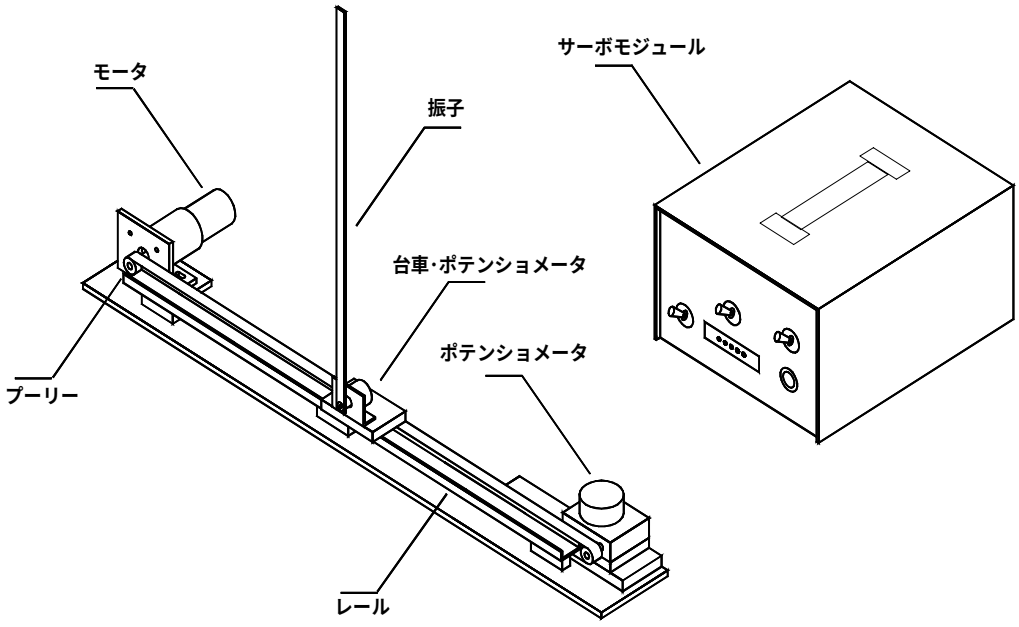

図 0.3 倒立振子キット概略図

このように本振子キットの構成はもっとも基本的です。これまでにいろいろな目的に合わせて、多くの振子系が報告されています。その一部を図 0.4～図 0.6 に示しますが、本振子キットに対して改良を加えることで、これらの振子系を構成することも可能です。ぜひ挑戦してみてください。

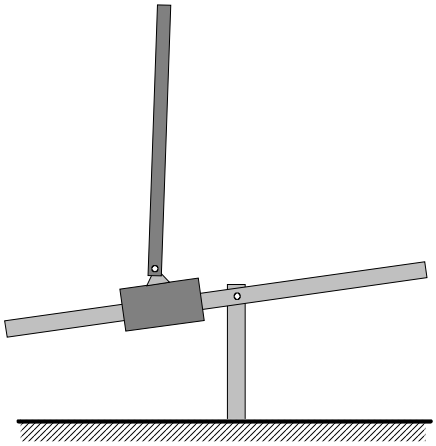

図 0.4 シーソー型倒立振子

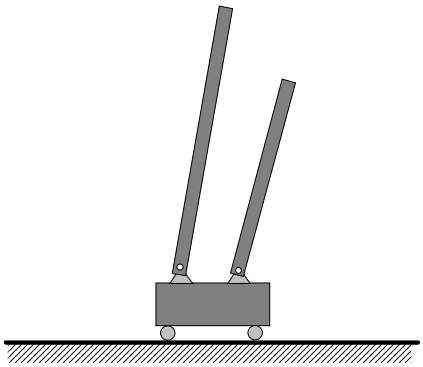

図 0.5 並列型倒立振子

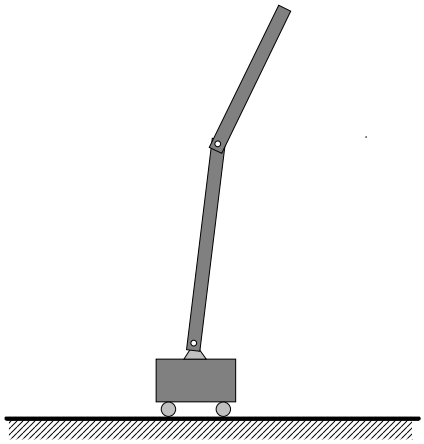

図 0.6 直列二重型倒立振子

本マニュアルは、振子キットを活用する上で必要と思われる事項をまとめたもので、以下で述べるように 3 部構成となっています。

第 1 部は、振子キットがお手元に届いてから、それを計算機に接続し安定化実験を行うまでの標準的な手順を説明しています。ぜひ一通りは試してみてください。特にトラブルがなければ、説明された手順に従うことで、数時間で自動制御による安定化を体験することができるはずです。

第 2 部は、振子を立てるための制御理論をまとめたものです。2-1 章では

ラグランジュ法を用いて本振子キットに対する運動方程式を導出し、それを線形化して状態空間モデルと呼ばれる線形制御理論の基礎となるモデルの導出を行います。2-2 章では、状態空間モデル中の物理パラメータの同定法について述べています。キットに付属の振子を対象とするのであれば、本章末にそれらの標準値が載せてありそれを利用すればよいので本章を読み飛ばしても構いません。しかし、振子を変更したりキットを改良した場合、それに対応して物理パラメータ値を求める必要がでてきます。その場合、本章で述べる方法を参考にしてください。2-3 章では、このようにして得られたモデルに対して、線形制御理論を用いた設計の概略について説明します。ここでは、その大筋を理解することを目的としていますので、工学的な意味を重視し、定理の証明等は省略しました。それらについて興味のある方は、多くの制御理論の本が出版されていますので、そちらを参考にしてください。2-4 章では 2-3 章で設計した制御器を計算機上に実装する方法について説明します。

第 3 部では、本キットで使用しているハードウエアの詳細な説明を行います。ひととおり振子の安定化を試された方が、キットに独自の改良をする際に役に立つ情報です。DC サーボモータとサーボアンプ、ポテンショメータを使うメカトロニクスの工作の際には参考になると思いますので、ぜひ御一読されることをお勧めします。

ところで、制御対象のパラメータ同定や線形制御理論を用いた制御器の設計、それを評価するための数値シミュレーションを行う場合、計算機の援助が必要不可欠です。ここでは、下記に示す計算機環境を前提としています。

ハードウエア　: DOS/V
　　CPU　DX2 66MHz 以上推奨
　　メモリ　16MB 以上推奨
　　ハードディスクの空き容量
　　1M 程度の空き
　DTx1-EZ ボード[†] (A/D,D/A 変換ボード)
ソフトウエア　: Microsoft MS-DOS ver.5.0/V 以上
　Microsoft Windows 3.1

[†] DT01-EZ ボード、DT21-EZ ボード、DT31-EZ ボードをまとめて DTx1-EZ ボードと呼ぶことにします。使用するボード名に置き換えて読み進めて下さい

DT VEE(デモ版でも可)
MATLAB ver.4.2c.1
SIMULINK ver.1.3c
Control toolbox ver.3.0b
Robust toolbox ver.2.0b
Microsoft Visual C++ ver.1.0 以上‡

なお、添付のディスクにサンプルソフトが提供されています。本文の指示に従ってお持ちの計算機にインストールしてください。

本キットを用いた制御実験を通して、一人でも多くの方が制御理論に興味を持つことができれば望外の喜びです。それでは、前書きはこのくらいにしておいて、さっそく実験を開始することにしましょう。

‡ 第 2 部 4-3-2 では Watcom C/C++ ver.10.0a 以上が必要です