# 方向制御システムの構築に対する提案

山形大理:　郡司修一
ISAS/JAXA:　斎藤芳隆
阪大理:　林田清、穴吹直久、常深博
理化学研究所:　三原建弘
山形大理：佐々木大悟、星野利典、藤田直樹、門叶冬樹、櫻井敬久
KEK：岸本祐二

## 1　はじめに

我々は硬X線領域での偏光観測を世界に先駆けて実現するために、硬X線偏光度検出器PHENEXを開発し、岩手県三陸町、北海道大樹町で数度に渡る気球実験を行ってきた。それらの実験で大きな問題となったのが、検出器のアジマス方向の制御である。特に2006年の実験では、6時間のレベルフライト中2時間程度しか正常に制御を行う事ができなかった。後の解析から、リアクションホイールの不感帯が原因である事が分かり、それに対しての対処を行い、2009年に再度大樹町で気球実験を行った。しかし、その実験でも方向制御が正しく行われなかった。その時の原因は予想したよりも吊り紐が上空で固くなり、リアクションホイールのパワーが足りなくなってしまったためである。そこで、我々はリアクションホイールの慣性モーメントを2倍に増やし、さらに制御のレートを1Hzから16Hzに上げたシステムを開発した。また大きな電流が流れた時によりもどしモーターのドライバーが停止してしまい、復帰しないというハードウエア的なバグもつぶした。そして、2011年に神戸大学の青木先生がPIとして行われた気球実験に、我々の制御システムを相乗りさせてもらい、制御の実験を行った。この実験では天体を追尾するという事は行わず、検出器を指定した方向に向け続けるという事を行った。実際のゴンドラのアジムス方向の角度が図1に示されている。図1の横軸は時間(sec)であり、縦軸が目標方向とゴンドラの方向の差である。制御が正常にかかっていれば、グラフが0度付近で一定になる。我々はこの図で2000~3000secの時間帯で方向制御の実験を行った。図から分かるとおり、多少のオフセットが履いてはいるものの約1度以下の精度で制御が成功している事が分かる。この実験により、方

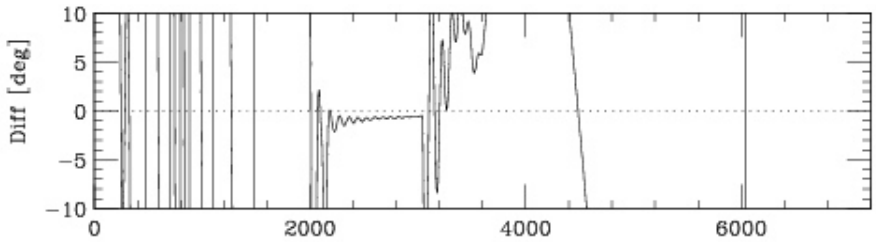


図 1: 横軸が時間、縦軸が目標方向とゴンドラの方向の差を取ったグラフ。制御を行っていた2000~ 3000secの間で、ゴンドラが1度以内の精度で目標方向を向いていた事が分かる。

向制御の問題がほぼクリアできたという事は分かったが、まだ幾つか細かい問題が残っている。一つは以上の実験結果でも分かるように、検出器が目標方向から0.5度程度離れた方向を向こうとしていることである。この原因として、ゴンドラの慣性主軸上に吊り紐がなかったために縦揺れが誘起されたという説があるが、その理解で良いのかは現在検討中である。二つ目に、我々は現在方向制御のためのシミュレーションプログラムを構築して使用しているが、そのシミュレー

ション結果と地上での制御試験の結果が、多少違っていた。以上の事から考えれば、次の実験で方向制御が成功する確率はかなり高くなったものの、まだ方向制御に関して完全に理解している段階には達していないと思われる。

一方我々以外に、数度程度の精度で方向制御を行おうとしている実験グループが少なくとも 3 つは存在する。本来それらのグループと協力し、共同でシステムを作り上げれば、より効率よく安定なシステムを構築できるはずである。自分としてはこの様な共同開発体制の整備を気球グループにお願いできればと考えており、最終的には個々のユーザーが制御システムを開発するのではなく、気球グループがスタンダードなものを提供して頂けないだろうかと考えている。しかし、以上の事を実現するには、まずは我々自身がスタンダードなシステムを開発する努力をしなくてはならない。特に他のグループと共同で開発を進めていく枠組みを構築するには、我々の方から歩み寄る必要がある。その中でも一番大きなものが制御に対するシミュレーション手法である。我々は今まで C 言語によるシミュレーションプログラムを開発し、使用してきた。しかし、制御関連の多くの専門家は MATLAB/Simulink というソフトウエアにより、制御のシミュレーションを行っているため、我々も急遽その手法を取り入れて、我々の制御システムを MATLAB/Simulink を使って構築する事を始めた。本プロシーディングスでは、その構築状況を説明する。

## 2 MATLAB/Simulink に関して

方向制御を行うには、3 つの要素が必ず必要になる。一つはリアクションホイールやよりもどしモーター等の実際にゴンドラを回転させるためのアクチュエーターと呼ばれる装置、もう一つはアクチュエーターの状態や現在のゴンドラの向きを調べるためのセンサー、そして最後にセンサーからの信号を読み込みアクチュエーターに指令を出すコントローラーである。従って、制御のシミュレーションを行うためには、これら 3 つがシミュレーターに入っていなくてはいけない。その中でも一番重要なのがアクチュエーターのシミュレーションである。アクチュエーターは電気信号によって力学的な動作をするハードウエアであり、その動作は電気的な微分方程式と力学的な微分方程式で表現される。従って簡単に言ってしまえば、制御のシミュレーションをするとは、その微分方程式を解くということである。MATLAB/Simulink とは、微分方程式を視覚的に分かりやすく表現して解くことができる道具である。ここではまずリアクションホイール単体の非常にシンプルな例を示し、そのやり方を説明する。まずリアクションホイールとは図 2 の左図に示されている様な装置で、真ん中の図に示されている様に中心にモーターがあり、そのモーターが回ると 4 本の重りが連動して回る装置である。モーターとは、回転できるコイルが磁場中に存在しているものであると捉える事ができるが、その様子が右図に示されている。リアク

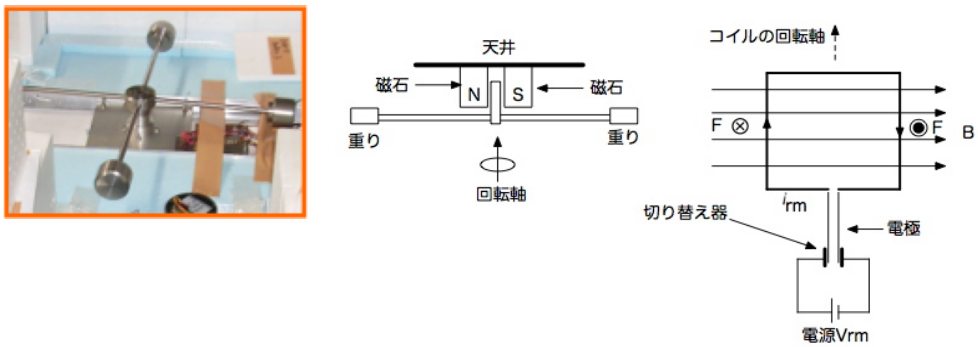


図 2: 左図はリアクションホイールの図。真ん中と右図はモーターを描いた図。

ションホイールに流れる電流や回転の様子は、以下の 4 つの微分方程式で表現される。

$$V_{rm} = L_{rm}\frac{di_{rm}}{dt} + R_{rm}i_{rm} + V_{rmr} \tag{1}$$

$$N_{rm} = K_t i_{rm} \tag{2}$$

$$V_{rmr} = K_r\frac{d\phi}{dt} \tag{3}$$

$$N_{rm} = I_{rm}\frac{d^2\phi}{dt^2} \tag{4}$$

一番目の式はモーターという回路に対するキルヒホッフの法則である。$V_{rm}$ はモーターに与える電圧、$L_{rm}$ はコイルの自己インダクタンス、$R_{rm}$ はコイルの抵抗、$i_{rm}$ はコイルを流れる電流、$V_{rmr}$ はコイルが回転する事で生じる逆起電力である。二番目の式は、コイルに流れる電流とモーターが発生するトルク ($N_{rm}$) が比例するという式である。3 番目はモーターに発生する逆起電力は、リアクションホイールの回転速度 ($\dot{\phi}$) に比例するという式である。最後の式はリアクションホイールに働くトルクと慣性モーメント ($I_{rm}$) の関係式である。これらの一つ一つの微分方程式を MATLAB/Simulink ではブロック線図というもので表す。このブロック線図が図 3 に示されている。これらのブロック線図の信号同士をさらに接続し、ユーザーが入力信号として ($V_{rm}$) を

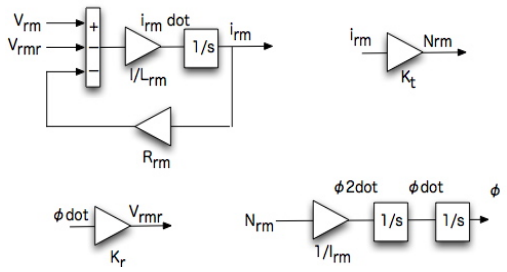


図 3: それぞれの微分方程式対応したブロック線図。

与えると様々なパラメーターがモニターできる仕組みになっている。以上の様に微分方程式が立てられれば、後はソフトウエアが自動的にシミュレーションを行ってくれるため、バグが入りにくいシステムを構築できる。またそれ以外にも以下の様な利点が挙げられる。1) 最適な制御パラメーターを探す仕組みが用意されている。2) この様に製作されたブロック線図を C 言語のプログラムとして出力する事ができる。そのため、気球制御の CPU にその C 言語で書かれたプログラムをインストールすれば、シミュレーション通りの制御が行えるようになる。

## 3　気球制御のシミュレーションと今後の実験

実験室での制御の挙動とシミュレーションが合っている事をまず確かめる必要がある。そのためには少なくとも吊り紐、よりもどしモーター、ゴンドラ、リアクションホイールの系をシミュレーションするためのブロック線図を描く必要がある。これら 4 つの要素に対しての微分方程式をまず立てる必要がある。その作業はかなり複雑であるが、我々は回転運動を並進運動のアナロジーとして捉える方法により、実際のブロック線図を書くことができた。その様子が図 4 に描かれている。左図がシミュレーションすべき系であり、真ん中の図はそれを並進運動のアナロジーとして考える方法を示してある。そして、右図はそのアナロジーを基に構築したブロック線図である。またブロック線図にはサブルーチンの様にサブブロックというものが定義できる。その様子が真ん中下の方に描かれている。

このブロック線図の中には制御を表したサブブロックが存在するが、今回は以下の様な制御を行う事とした。以下の式で $\theta_0$ がゴンドラを向けたい方向。$\theta$ がゴンドラの現在向いている方向。$\gamma$ がよりもどしモーターの回転角度、$V_y$ と $V_{rm}$ がよりもどしモーターとリアクションホイール

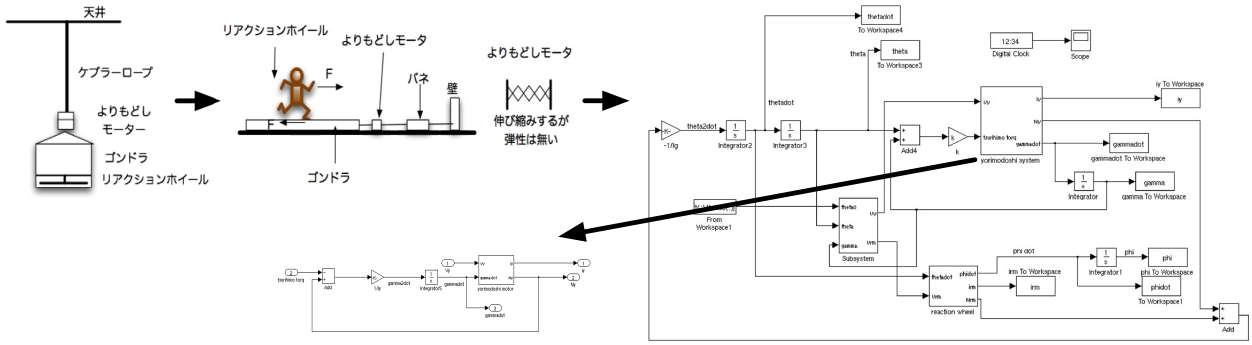


図 4: 左図が考えるべき系。真ん中の図がそれを並進運動に焼き直したアナロジー。右図が全系のブロック線図。真ん中下の図は、サブブロックである。

に与える電圧である。また $A, B$ は比例定数である。一応今回のシミュレーションは、吊り紐は制御が始まる段階では捩れてはいないという仮定で行った。

$$V_y = A \times (\gamma - \theta_0) \tag{5}$$

$$V_{rm} = B \times (\theta - \theta_0) \tag{6}$$

上の式は、「現在の方向から目標方向の差だけ、吊り紐を捩りなさい」という事を意味しており、下の式は、「ゴンドラが目標方向を向くまでリアクションホイールに電圧を与えなさい」という事を意味している。つまり、目標方向を向いたときに吊り紐の捩れが無くなる様にし、さらにゴンドラが目標方向を向くようにリアクションホイールを回転せよという命令を与えていることになる。実際にこの制御方法によって、ゴンドラがどの様な回転を行うかが、図 5 に示されている。上図と下図の横軸は時間 (sec) であり、上の図の縦軸はゴンドラの角度、下の図の縦軸は目標方向とゴンドラの角度の差である。図から分かるように 300 秒後にはほぼ 0.1 度の精度でゴンドラが目標方向を向くことが分かる。

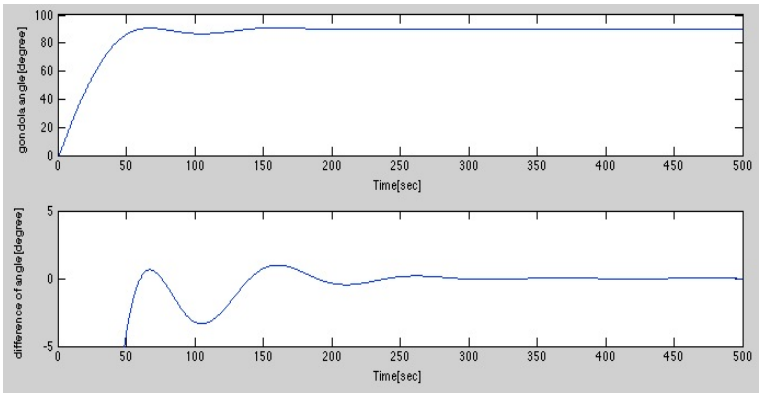


図 5: 制御シミュレーションの結果。ゴンドラを 90 度回転させて止めるという制御を行った結果。

この様な制御法はあまりにもシンプルであるが、直感的に何をやっているのか非常に良く分かる制御であり、ハードウエアのバグを調べるには格好の方法となる。そのため、この様な制御でシミュレーションとハードウエアの挙動の比較を今後行っていく事を計画している。また今回は摩擦の影響や繋いでいる電池の性能に関してはまだ十分考慮されていない。今後はこの様な事も全部考慮したフルシミュレーターを完成させる予定である。そして、他のグループと共同の開発体制を構築し、それに参加できればと考えている。