マイクロマウス トラブル事例集2013

The collection of Micromouse’s trouble - 2013

自己紹介

加藤雄資　Yusuke Kato

マイクロマウス中部支部会員

- 2005年　中部地区初級者大会　初参加
- 2006年　中部地区大会　完走
- 2007年　全日本初参加
- 2008年　全日本　バンダイナムコ賞
- 2009年　台湾大会２位
  全日本　ハーフ優勝
- 2010年　全日本　クラシック優勝　ハーフ３位
- 2011年　APEC参加
  全日本　クラシック２位　自律賞
- 2012年　全日本　クラシック優勝　ハーフ２位

導入編
マイクロマウスで大切なこと

Introduction

マイクロマウスで最も大切なことは何でしょう？

- 最高速度？
- ターン速度？
- 加速度？
- 軽量化？

答えはどれでもありません

最も大切なのは、

確実に完走する能力です

ところがマイクロマウスには、完走を妨げる様々な

不具合の罠が潜んでいます。

そこで、この場をお借りして、過去に体験した、

あるいは目撃した様々なトラブルを紹介し、

その回避方法を提案していきます。

ただし、対策についてはヒントにとどめ、あまり深くは掘り下げません。

こうしたものはケースバイケースであることが多く、

手法だけを真似ることにあまり意味はないからです。

1章
ヒューマンエラー編

1. Human Error

CASE 1 - 1 ：

うっかり電源を切ったら迷路情報が消えてしまった

CASE 1 - 2 ：

ゴール位置の設定が間違っていた

CASE 1 - 3 ：

操作ミスで、迷路情報を消す探索モードで走らせてしまった

CASE 1 - 1 ～ 3

解説と対策 Explain & Measures

人は、**必ず間違いを犯す**ものです。

単に「注意する」という精神論では間違いを防ぐことはできません。特にマウスは・・・

- サイズや重量の制限からインターフェースが貧弱
- 大会本番では激しく緊張する

というミスをしやすい環境です。

反復練習と、以下のようなシステム面での対策によって失敗のダメージを最小化しましょう。

- 重要情報は必ず不揮発メモリに保存する
- マシン単体でゴール位置を変更できるようにする
- 走行開始前に、どのモードで走り出そうとしているのか操作者に伝わるようにする

CASE 1-4 :

マウスを落として壊してしまう

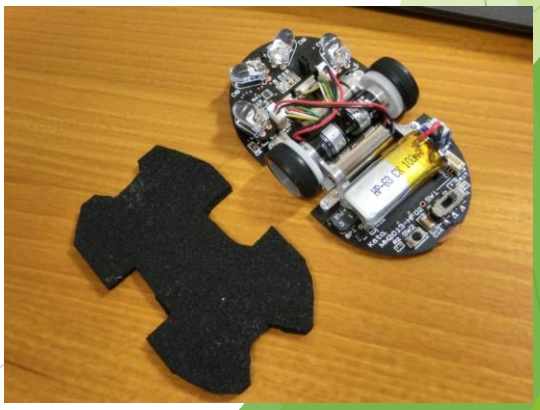
- 落とすわけがない、と思っていても、意外と落とします
- 落とすのは、だいたい以下の２パターンが多いです
  - 迷路から取り上げるときに落とす
  - 机の上から引っかけて落とす

CASE 1-4

解説と対策 Explain & Measures

- 迷路から取り上げるときに落とす　⇒
  - マウスが自力でスタート地点に帰還するようにする
  - 腰にもやさしい

- 机の上から引っかけて落とす　⇒
  - 敷き物を作ってタイヤが机に触れないようにする
  - 作業中は常にマウスが視界に収まるように置く
  - 机の端には置かない



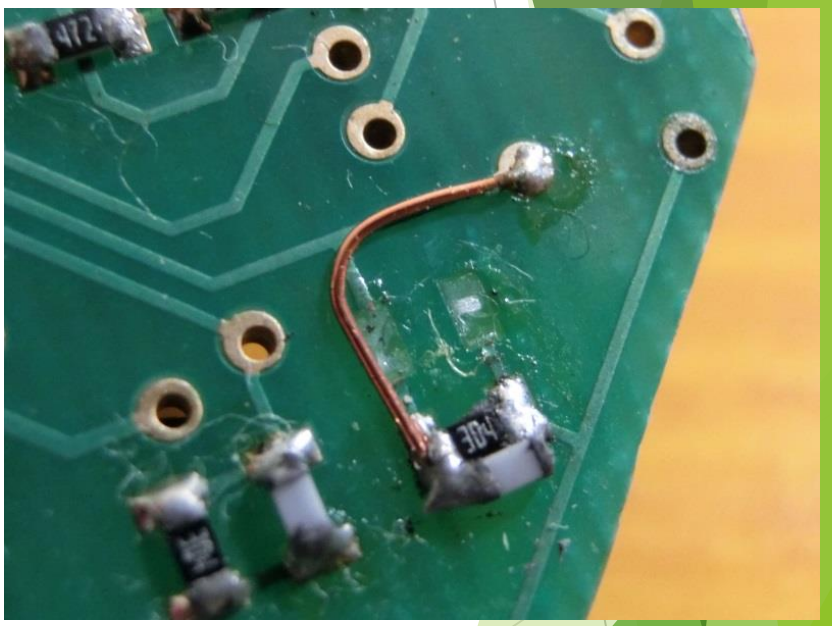
２章
メカ&ハードウェア編

2. Mechanism and Hardware

CASE 2-1：

底面実装コンデンサが段差に引っかかって吹っ飛ぶ

板マウスで基板の実装面積が苦しいとき、両面を活用したくなりますが、実は要注意。



部品が段差に引っかかると、容易に引っぺがされ、パターンごと持っていかれます。

次のどちらかの対策を講じましょう

- 背面には部品を実装しない
- 十分に車高を高くする

CASE 2-2 :

そもそも段差を乗り越えられない

クラシックマウスの段差の規定は最大1mm

ハーフマウスは0.5mm

忘れないようにしましょう。

迷路の床板の角が丸みを帯びているため見かけ上段差が小さくなっていても実際は思いのほか段差が大きいこともあります。

タイヤが浮いてしまえばマウスは走れません。

CASE 2-3：

ねじのゆるみに気づかずそのまま走らせてしまう

マウスは振動や衝撃が加わるので、気づかないうちにねじが緩んでいることがよくあります。

- サイズの小さいねじ、樹脂部品を固定しているねじ、あるいは樹脂ねじは特に要注意

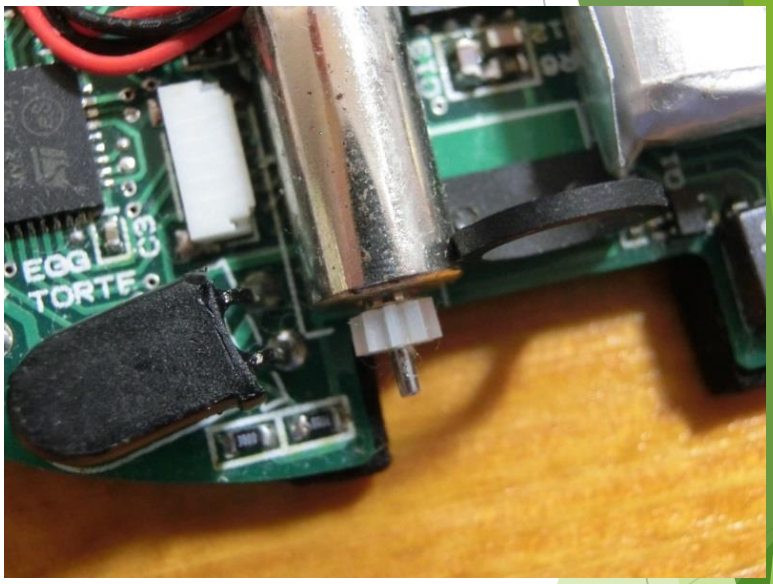
以下のように対策しましょう

- 定期的な点検
  - 特に大会直前は必ずチェック
- ねじロック剤の使用

CASE 2-4 :

モーター軸にホコリが巻き付いて過大なトルク損失

何の変更も加えていないのに、急にマウスの動作が不安定になり、どんなにゲインを調整したり回路を調べても原因がわからない・・・



そんな時、意外な場所に原因が見つかることがあります。

モーターとピニオンの間に、ホコリが詰まっていたりはしませんか？

CASE 2-4：

モーター軸にホコリが巻き付いて過大なトルク損失

この不具合は、知らないとなかなか気づきません。

特にハーフマウスでは小さいモーターを使うため、わずかな摩擦トルクの増大が致命的な影響を及ぼすことがあります。

対策は単純で、

- モーターとピニオンの間に隙間を設ける

ということです。

しかし、設計段階で考慮されていないと難しいです。

最低限、いざというときに掃除できる構造にしておきましょう。

CASE 2-5：

衝突の衝撃で
はんだクラックが発生

マウスではなかなか避けて通れない問題です。

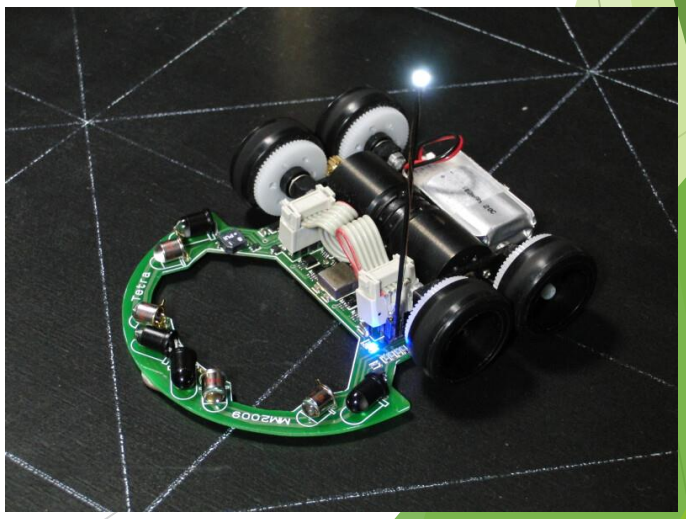
本来であれば基板には一切衝撃が加わらないことが理想ですが、少しでも重量を減らしたいため、そうもいきません。

CASE 2-5

解説と対策 Explain & Measures

完璧な対策は難しいですが、クラックの確率を下げるための方案をいくつか例示してみます。

- むやみに基板を薄くしない
- 意図的に弱い個所を作り衝撃を吸収する
- 機体のねじれ剛性を高める
- クラックしやすそうな部品は機体後部に集める
- クッションの取り付け



CASE 2-6：

コンデンサ選定ミスで昇圧回路が発振

- 昇圧ICの推奨回路の通りに回路を設計
- ・・・のつもりが、コンデンサの容量しか見ていなかった

- セラミックコンデンサは、**DC電圧がかかると容量が低下**する
- 特製の悪い小型大容量セラコンだと、10%程度まで下がってしまうことも
- 結果、容量不足で発振・・・

容量だけでなく、他の諸特性もしっかり確認しましょう。

CASE 2-7 :

ピニオンが空転

ピニオンの固定は、マウス制作者を悩ませる問題のひとつです。

その小ささからキーなどは使えないので、主に以下のような固定方法が使われます。

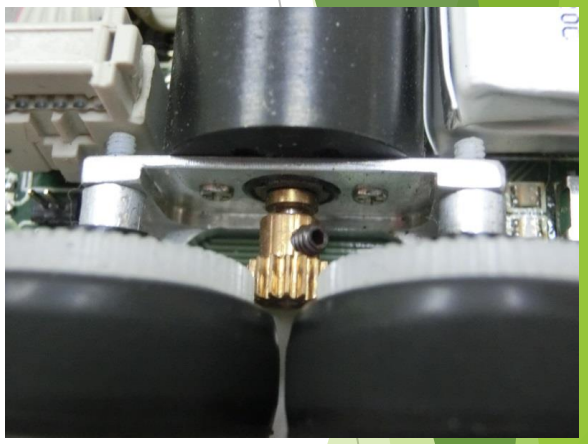
- イモねじ止め
- 圧入
- 接着

それぞれ推奨される使い方や問題への対策が異なります。

CASE 2-7

解説と対策 Explain & Measures

- イモねじ止め　⇒
  - ピニオンにねじ穴を切り、イモねじを締めてシャフトに固定する方法
    - 確実性は高め
    - 幅が広くなってしまうのが欠点
    - 樹脂ピニオンに使うのは危険
  - イモねじが緩むことで、ピニオンがすべるようになる
  - ほかのねじのゆるみと同じく、定期的な点検と、ねじロック剤の使用が効果的



CASE 2-7

解説と対策 Explain & Measures

- 圧入　⇒
  - ピニオンの穴をシャフト径よりわずかに小さくして押し込み、摩擦力で固定する
  - 金属ピニオンだと圧入力が大きく、シャフトが後ろに出ているモータでなければ軸受破損の恐れあり
  - 樹脂ピニオンの推奨しめしろは0.01～0.02mmぐらい
  - しめしろが適切に管理されていれば、ほぼ緩むことはない

CASE 2-7

解説と対策 Explain & Measures

- 接着　⇒
  - ピニオンの穴をシャフト径よりわずかに大きくして、接着剤で固定する
  - 接着剤の選定がキモ
    - 接着剤自体の強度があり、粘性の低いものが必要
    - 有機溶剤系は強度が低く、粘性も高い
    - エポキシ系は強度はあるが、粘性が高く、うまく流れ込まないおそれ
    - おすすめは、はめあい用嫌気性接着剤
      - ロックタイト600番台が該当

3章
ソフトウェア編

3. Software

CASE 3-1：

演算子の優先順位を勘違いして計算結果が正しくない

CASE 3-2 ：

比較演算子と代入演算子を間違えて使う（ = と == ）

CASE 3-3 ：

変数の初期値に変な値が入っていてバグる

CASE 3-1～3

解説と対策 Explain & Measures

マウスに限らず、C言語では定番のバグです。

定番すぎてあまり語ることもないですが、

一度何かしらのコーディング規約を学んでみるとよいでしょう。

CASE 3-4：

空ループが最適化によって消されてしまう

マイコンでは、タイマを使うほどでもないちょっとしたウェイトを挿入するときに空ループが使用されることがしばしばあります

ところがコンパイラから見ると無意味なコードのため、最適化によってループそのものが削除されてしまうことがあります

CASE 3-4

解説と対策 Explain & Measures

- ループ変数にvolatile修飾子を付けることで最適化を抑止する

```
volatile unsigned int i;
for(i=0;i<10000;i++);
```

- その他volatile修飾子を付けるべきもの
  - マイコンのペリフェラルレジスタへのアクセス
  - mainループと割り込みの双方から参照するフラグ変数

CASE 3-5：

ペリフェラルレジスタへのアクセスの仕方で挙動が変わる

- whileループ内で特定のペリフェラルレジスタのフラグを監視していたところ、立つはずのフラグがいつまでたっても立たない
- whileループ内に少しの空ループを入れてみたところ正常に立つように・・・

- マイコンの周辺回路はあくまで電子回路なので、あまりロジック的でない動きをすることもありえる
- 回路自体に不具合があったり、マニュアルに間違いがあったりすることもあるので、すべてを疑うべし

CASE 3-6：

ぶつかって位置情報がずれた後も探索し続けて迷路情報が壊れる

CASE 3-7：

ぶつかった後に全力走行を続けようとして暴れまわる

これらは基本的に、フェールセーフ機能がないために起こります。

フェールセーフには誤作動のリスクもありますが、フェールセーフがないことによる上記のリスクのほうが圧倒的に有害です。

CASE 3-6～7

解説と対策 Explain & Measures

- 自分のマウスのフェールセーフ機能の概要
  - 角度偏差が一定以上、または速度偏差が一定以上の状態が80ms連続したらエラーとする
    - 急激な値の変化がある値（加速度や角速度）は使わない
    - エンコーダと慣性系センサの両方を組み合わせるとタイヤのロックと空転の両方を検知できる
  - エラーが出たら直前に見た壁の情報をいくつか消去して迷路情報を保存
- ステッパマウスであっても、フェールセーフは不可能ではない
  - ジャイロの使用
  - 指令速度と前壁センサ値の変化の比較

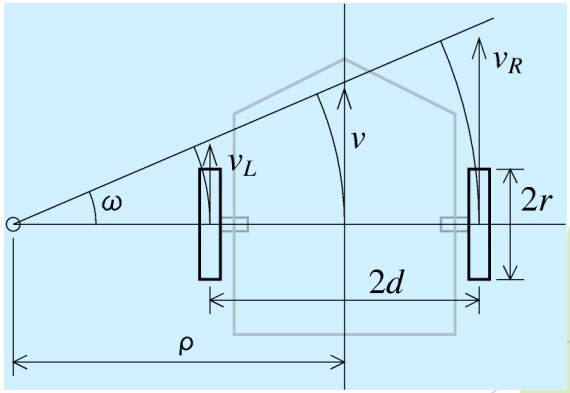
４章
制御＆複合トラブル編

4. Control & Complex

CASE 4-1：

オドメトリで計算した走行距離や角度が、実際の値と合わない

- オドメトリとは？
  - 左右の車輪の回転速度から移動速度と角速度を求め，それを積分して位置と姿勢を求める自己位置推定法の一種
- タイヤの移動が軸と垂直の方向に拘束されることを利用
- しかし、きっちりと正確な直径やトレッドを入力しているのに、実際の軌道が合わない…



図は東北学院大学HP
http://www.mech.tohoku-gakuin.ac.jp/rde/contents/course/robotics/wheelrobot.html
より引用

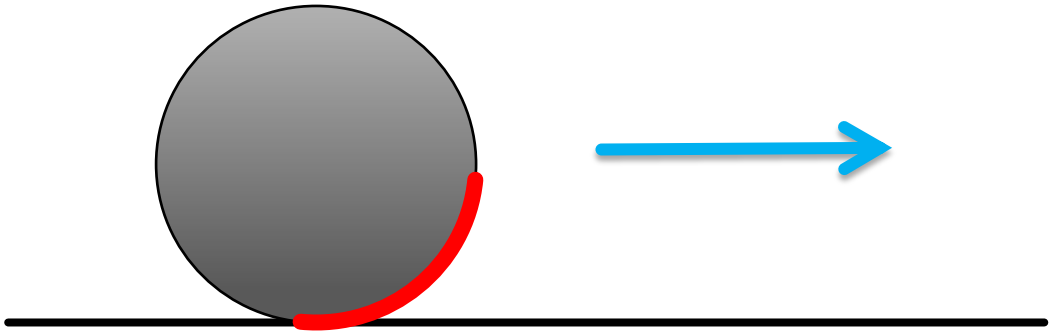
CASE 4-1

解説と対策 Explain & Measures

一言でいうなら、**タイヤを信じるな**ということです。

タイヤが力を発生させるとき、何が起こるでしょうか？

タイヤは弾性体ですから、力を生み出す際には必ず変形します。この変形によって、**タイヤが完全にグリップした状態であっても**ある程度のスリップが発生します

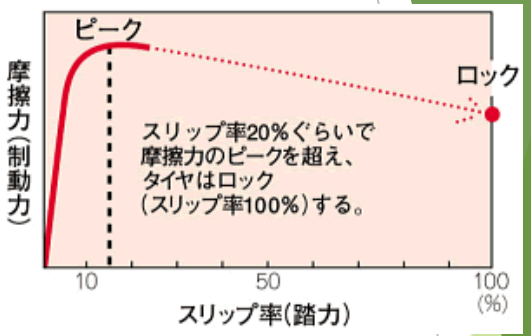


CASE 4-1

解説と対策 Explain & Measures

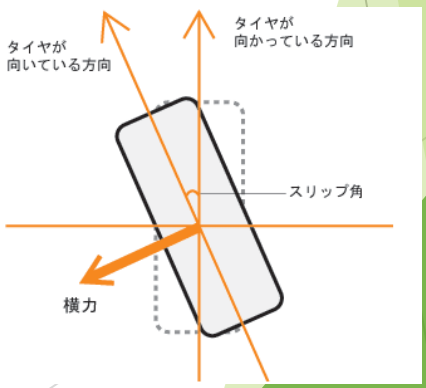
スリップの量を表す指標として以下の２つの値が用いられます。

- スリップ率
  - 進行方向の回転量と実際の進行距離の差
  - 縦方向の力が大きくなるほど大きくなるが、限界あり



ニッポンレンタカーHP
http://www.nipponrentacar.co.jp/info34_4.htm
より引用

- スリップ角
  - タイヤの向きと実際の進行方向の差
  - 横方向の力が大きくなるほど大きくなるが、限界あり



日経ものづくりHP
http://techon.nikkeibp.co.jp/rcolumn/DM/COLUMN/20041124/
より引用

CASE 4-1

解説と対策 Explain & Measures

- Maneuverの最大スリップ率とスリップ角はどのぐらい？

最大スリップ率：約２５％

最大スリップ角：約２０度

このレベルになると、オドメトリは完全に破綻します。

影響を抑えるために、以下のようなアプローチが必要です。

- スリップがあることを前提に、ほかのセンサと組み合わせた制御を行う
- 加速度や横Gに対するスリップの量を小さくする

CASE 4-1

解説と対策 Explain & Measures

- スリップがあることを前提に、ほかのセンサと組み合わせた制御を行う
  - スリップ量を推定して補正する
  - 壁切れによる走行距離の補正
  - ジャイロや加速度センサといった慣性系センサの利用
- 加速度や横Gに対するスリップの量を小さくする
  - 軽量化
  - タイヤ硬度の最適化
  - ワイドタイヤの利用
  - 変則４輪
  - 吸引・・・

CASE 4-2 :

ジャイロセンサで正確な角度が求められない

- 近年のマイクロマウスでは広く普及が進んでいるジャイロセンサ
- 小型、高精度、安価なジャイロが容易に手に入るようになったのは、ここ数年のこと
  - カメラやゲーム機で採用されたことで一般化が進んだ
  - 最近ではスマートフォンへ向けが大きい

- ジャイロセンサの出力は基本的に「角速度」
- 角度を求めるためには積分する必要があり、わずかな狂いが蓄積されて大きな誤差になる

CASE 4-2

解説と対策 Explain & Measures

ジャイロを狂わせる要因には様々なものがあり、一筋縄ではいきません。

思いつく限りを挙げてみます。

- ジャイロそのものの誤差
- 電源のノイズ
- A/D変換の誤差
- すりこぎ運動による誤差
- ジャイロの応力歪みによる誤差

CASE 4-2

解説と対策 Explain & Measures

- ジャイロそのものの誤差　⇒
  - 温度ドリフト
    - 角速度０の時の出力電圧には、ある程度の温度依存性がある
    - 気温や、通電による温度上昇でゼロ点が狂うことは十分に考えられる
      - 走行の直前にゼロ点補正を行うことで対応
  - 非線形性
    - 実際の角速度と出力電圧は完全には比例しない
    - ゆっくり９０度回った場合と、すばやく９０度回った場合で結果が異なる
      - 線形性の良いジャイロを選ぶ
      - キャリブレーションを行う（難しい）

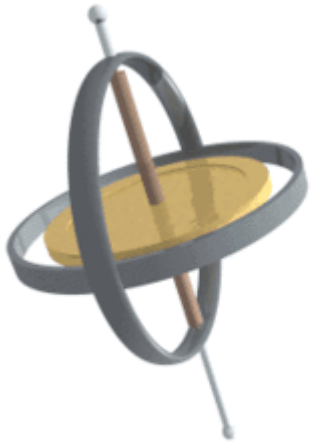
CASE 4-2

解説と対策 Explain & Measures

- 電源のノイズ　⇒
- A/D変換の誤差　⇒
  - ジャイロとA/D変換器の電源電圧は一定で安定しているか
    - フィルタの挿入
    - デジアナ分離
  - ジャイロとA/D変換器のGNDに電位差はないか
    - １点接地を心がける
    - 大電流経路を離す
  - A/D変換器自体にも非線形性あり
    - データシートで確認を
    - A/D変換器内蔵のデジタルジャイロを使う

CASE 4-2

解説と対策 Explain & Measures



- すりこぎ運動による誤差　⇒
  - すりこぎ運動とは
    - 自転している物体の回転軸が、円をえがくように振れる現象
  - 原理は異なるが、マウスでも同じような動きが発生する
    - ２輪マウスは前後に傾くため、加減速時に旋回軸も前後に振れる
  - すると、軸上の回転角度と姿勢変化の角度が一致しなくなる
  - 変則４輪は前後の傾きが少ないため有利

CASE 4-2

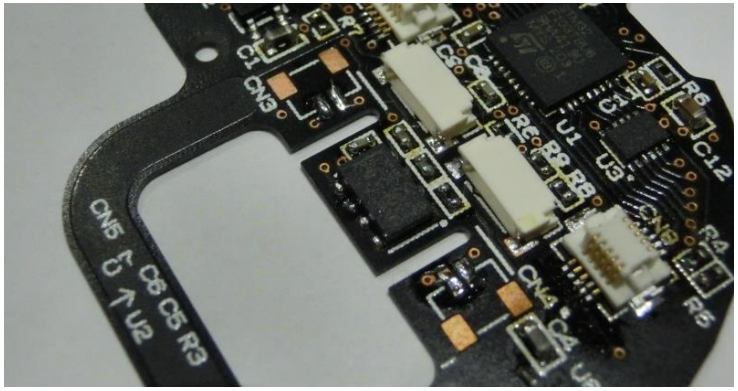
解説と対策 Explain & Measures

- ジャイロの応力歪みによる誤差　⇒
  - MEMSジャイロの原理
    - MEMS技術によって微小なおもりとばねを半導体上に作りこみ、慣性力によるおもりの移動を圧電効果や静電容量によって検出する
  - そのため、ジャイロ素子自体に応力がかかって歪むと、誤差となって表れてしまう
  - 加減速によって前後補助輪に力がかかって基板が曲げられたり、遠心力で基板がねじられるとジャイロが歪む危険性あり

CASE 4-2

解説と対策 Explain & Measures

- ジャイロの応力歪みによる誤差　⇒
  - 完全に別基盤にして柔軟物で貼り付ける
    - 重量増加がネック
  - 基板の形状で応力を逃がす工夫を盛り込む



  - 変則４輪は、基板への応力がかかりにくいので有利

おわりに

Conclusion

いろいろなトラブルを見てきましたが、ここに紹介したものはロボット製作上でぶち当たる様々なトラブルの、ほんの氷山の一角にすぎません。

とはいえ、もし皆さんが今後マウスを制作していく中でトラブルに遭遇した時、この公演の内容を思い出して、原因の特定、そして解決に至ることができたならば、これに勝る喜びはありません。

みなさんも、不具合に気を付けて、

よいロボットライフを！

ご清聴ありがとうございました

質疑応答

Q&A