



これは適当に話し合った結果をまとめたものなので、修正点は言ってください

1 ステート毎の挙動についての概略と注意すべき点

- state0
静止ステート
- state1
高速で良いと思われる。ただし、最初の周に関しては A 区間の途中から始まって、すぐにゲートがあるので、最初の周だけ例外的に低速で行くべきかもしれない。
- state2
state1 で高速になっていると考えられるので、最初のカーブを曲がる時に脱輪ないように減速するステート。state1 に入って一定時間経過後*1に自動的に state3 に入る。というか自動的にステート入るって無意味じゃない？自動的に入るならステートいらぬような気がする。減速→加速(登板)→減速→維持のステートとして一つだけ定義すれば state2,3,4 は一つにまとまる気がする。
- state3
登板用ステート。デューティ比を限界まで上げるべきと考えられる。やはり一定時間後に減速をする。ただし、もし限界までデューティ比を限界まで上げてそこまでスピードがでないのだとすれば登板区間終わりのカーブで脱輪することはないと考えられるので、state4 に遷移させずほっといてもピンポン球センサの検出が可能かもしれない。これは実験して決める。もし無理なら減速。
- state4
減速ステート。登板区間終わりのカーブを曲がる目的と、ピンポン球供給機の検出するための徐行が目的。検出し次第折り返す。登板でスピードが出ないのならまあほっといてもカーブは曲がれるだろうが、曲がった後の平らな部分で加速しないようにデューティを変更することは必要だろう。
- state5
X 地点から降板区間までは距離がないので、この区間は「降板部分用の低いデューティ比、低速走行」で走行してもあまり時間は変わらない気がする。だから、X から降板部分までのステートは一緒くたにして良い気がした。わざわざ短い区間のためにステートを設けることはない気もする。一定時間後に減速するための state6 に入る。
- state6
降板で加速しているだろうから、ゲート 2 を検出するために減速する・・・と思ったけど、もしかしたらここも state5 に統合してもいい気がする。すでに低いデューティ比で走行してるから、これ以上低くできるのか？ってことと、降板終わりからゲートまでの間は坂道ではないので勝手に減速してくれるのではないかと考えた
- state7
高速で動く。直線で稼ぐ？ただし、一定時間後に減速してゲート 1 の検出を出来るようにする必要がある。
- state8

*1 実験して最適な時間を測る

蛇行してるからゆっくり走れば良いと思われる。ゆっくり走ればきっとゲート 3 も検知できるはず。

- state9
すぐにゲートがあるのでゆっくり走れば良いだろう。state8 から state9 は時間がかからないはずなので、もし state8 になったあとにしばらくゲート検出がされなかった場合は state9 に自動で変更してしまうべきだろう。
- state10
折り返したあとにゲート 3 につくまでのステート。ここも距離が短いので低速で走れば良いだろう。
- state11
ゆっくりはする。ゲート 1 を感知したときに、現在何周したかを数えておいて停車するためのステートメントに移るかを定める。もしまだ回れそうなら回る。

2 注意すべきこと

- センサの誤作動として、同じゲートを複数回検出してしまう、いわゆるチャタリングのような状態に陥る可能性があること。すなわちソフトウェアの方でこれを防止する必要がある。具体的にはチャタリング防止のように、入力信号をパルス波に変換して、この値が立ち上がった瞬間から一定時間割り込みを禁止すれば良い？
- 周回数は、とりあえずプログラムには `#define n` 数とでもしておく。データを取って、もし一周するのにかかる時間の統計の分散があまりに大きかったら、しかたなく時間を見てまわるのをやめるかやめないか考えるプログラムをいれる。まあ実際そんなにきつくは無と思うけど。
- センサーの検出に関しては、フラグとして `flg` でも定義しておいて、閾値を超えた瞬間に `flg=1` となり、そのとき `state` を変更する。