**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ- ĐH QUỐC GIA HN**

KHOA ĐIỆN TỬ VIỄN THÔNG

\*\*\*\*\*\*



**BÁO CÁO MÔN HỌC: TRẢI NGHIỆM VÀ KHÁM PHÁ ROBOT**

ĐỀ TÀI: ROBOT DÒ LINE

**Nhóm sinh viên thực hiện :**

Nguyễn Bá Chung 18020237

Hoàng Văn Học 18020577

***Hà Nội, tháng 12 năm 2019***

**Mục lục**

1. **MỞ ĐẦU**
2. Giới thiệu đề tài
3. Mục tiêu
4. Phạm vi nghiên cứu
5. **GIỚI THIỆU VI XỬ LÝ VÀ THÀNH PHẦN MẠCH**
6. Arduino Uno R3
7. Module điều khiển : Mạch cầu H dùng L298
8. Module thu phát hồng ngoại MH-IR01
9. Động cơ DC và bánh xe
10. **SƠ ĐỒ LẮP GHÉP VÀ GIẢI THUẬT ĐIỀU KHIỂN**
11. Sơ đồ nối các linh kiện
12. Thiết kế thuật toán
13. Phân tích chuyển động và điều khiển
14. **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**
15. Kết quả
16. Hướng phát triển
17. **TÀI LIỆU THAM KHẢO**
18. **MỞ ĐẦU**
19. **Giới thiệu đề tài**

Ngày nay, Robotics đã đạt được rất nhiều thành tựu to lớn trong sản xuất công nghiệp cũng như trong đời sống. Sản xuất robot ngày càng phát triển trong đó chúng ta không thể không nhắc đến Mobile Robot với những đặc thù rất riêng. Mobile Robot có thể di chuyển một cách linh hoạt do đó tạo nên một không gian hoạt động lớn. Có rất nhiều loại Mobile Robot nhưng trong đó Robot dò line được ứng dụng rất nhiều trong cuộc sống.

Robot dò line là một trong những phương tiện di chuyển tự động đầu tiên được con người chế tạo, vì môi trường nó yêu cầu khá đơn giản – một nền bằng phẳng và một vạch màu tương phản mạnh (hoặc có các đặc tính khác) khác biệt mạnh so với môi trường xung quanh. Ta sẽ dùng cảm biến để đo đạc, phát hiện sự khác biệt này, sau đó nhúng một thuật toán điều khiển nào đó vào để điều khiển xe bám theo vạch line đó.

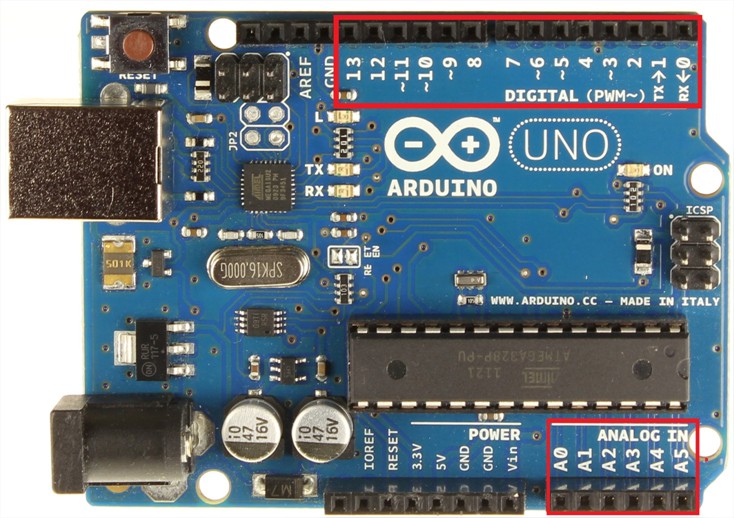
1. **Mục đích chọn đề tài**

Robot dò line tránh vật cản vừa có nhiều ứng dụng trong thực tế, vừa dễ dàng cho sinh viên vận dụng những kiến thức tiếp thu được thì khóa học trải nghiệm vào nó. Với những kết cấu đơn giản, kết hợp với những linh kiện điện tử( encoder, sensor dò line, sensor khoảng cách, động cơ,…) nên Robot này phù hợp cho sinh viên học tập và nghiên cứu.

1. **Phạm vi nghiên cứu**

Trong phạm vi môn học này, ta sẽ chỉ nghiên cứu về các nguyên lý hoạt động cơ bản của các module cảm biến được sử dụng trong robot. Và sự chuyển động của động cơ.

1. **GIỚI THIỆU VI XỬ LÝ VÀ THÀNH PHẦN MẠCH**
2. **Board Arduino Uno R3**



Arduino Uno R3  có thể sử dụng 3 vi điều khiển họ 8bit AVR là ATmega8, ATmega168, ATmega328. Bộ não này có thể xử lí những tác vụ đơn giản như điều khiển đèn LED nhấp nháy, xử lí tín hiệu cho xe điều khiển từ xa, làm một trạm đo nhiệt độ - độ ẩm và hiển thị lên màn hình LCD,…

Arduino UNO có 14 chân digital dùng để đọc hoặc xuất tín hiệu. Chúng chỉ có 2 mức điện áp là 0V và 5V với dòng vào/ra tối đa trên mỗi chân là 40mA.

Arduino UNO có 6 chân analog (A0 → A5) cung cấp độ phân giải tín hiệu 10bit (0 → -1) để đọc giá trị điện áp trong khoảng 0V → 5V. Với chân **AREF** trên board, bạn có thể để đưa vào điện áp tham chiếu khi sử dụng các chân analog. Tức là nếu bạn cấp điện áp 2.5V vào chân này thì bạn có thể dùng các chân analog để đo điện áp trong khoảng từ 0V  → 2.5V với độ phân giải vẫn là 10bit.

**Một vài thông số của Arduino UNO R3**

|  |  |
| --- | --- |
| Vi điều khiển | ATmega328 họ 8bit |
| Điện áp hoạt động | 5V DC |
| Tần số hoạt động | 16MHz |
| Dòng tiêu thụ | ~30 mA |
| Điện áp khuyên dung | 7-12V DC |
| Điện áp giới hạn | 6-20V DC |
| Số chân digital I/O | 14 ( có 6 chân PWM) |
| Số chân Analog | 6 ( độ phân giải 10 bit) |
| Dòng tối đa trên mỗi chân I/O | 30mA |
| Dòng ra tối đa (5V) | 500mA |
| Dòng ra tối đa (3.3V) | 50mA |

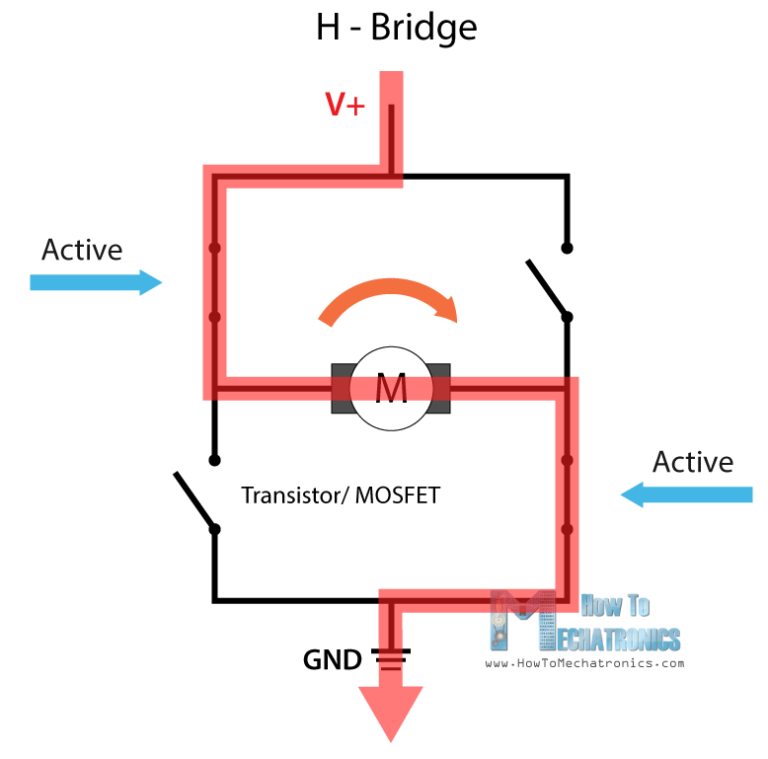
1. **Mạch cầu H dùng L298N**



* 1. **Nguyên lý điều khiển của mạch cầu H:**

Để điều khiển hướng quay, chúng ta chỉ cần đảo ngược hướng của dòng điện qua động cơ và phương pháp phổ biến nhất là chúng ta sử dụng mạch cầu H. Một mạch cầu H chứa 4 chân chuyển mạch, điện trở hoặc MOSFET. Bằng cách kích hoạt hai công tắc cùng một lúc, chúng ta có thể thay đổi đổi hướng của dòng điện, do đó thay đổi hướng của động cơ.

Do đó chúng ta có thể hoàn toàn kiểm soát được động cơ DC.



* 1. **Chip L298**

L298N là trình điều khiển động cơ H-Bridge kép cho phép ta điều khiển tốc độ và hướng của hai động cơ DC cùng một lúc. Module có thể điều khiển động cơ DC có điện áp từ 5-35V với dòng điện tối đa lên đến 2A

1. **Module thu phát hồng ngoại MH-IR01**

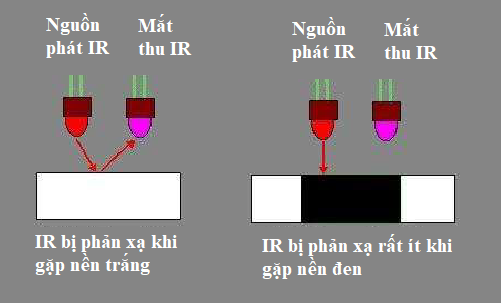


* 1. **Nguyên lý hoạt động của MH-IR01**

**MH-IR01** hoạt động theo nguyên lý thu nhận.

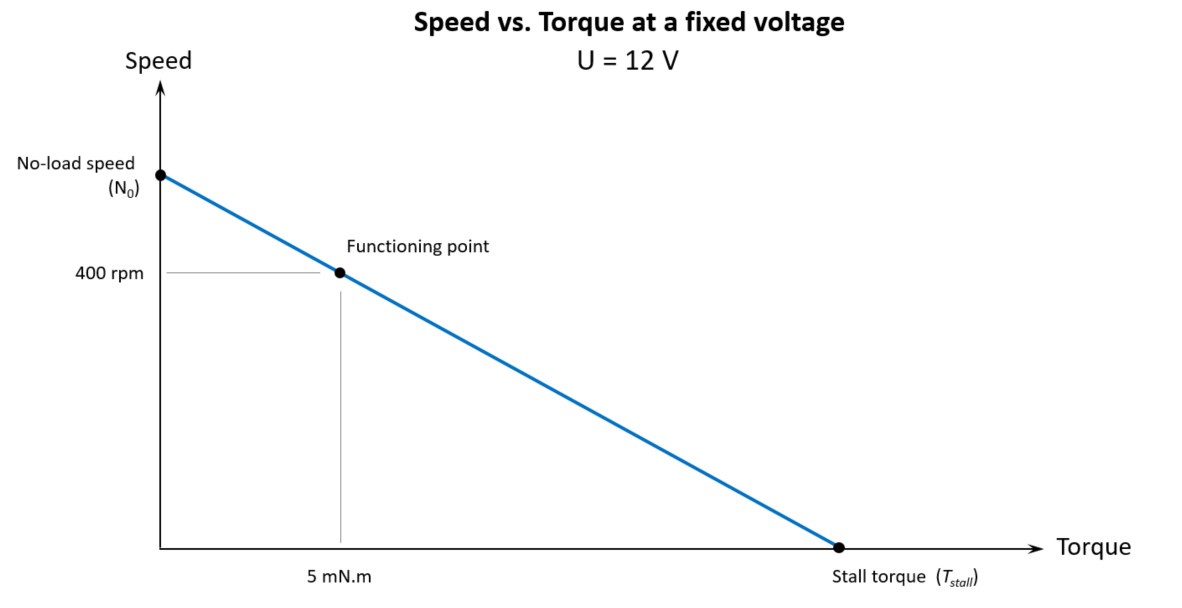
Khi led nhận được tia hồng ngoại từ những led IRE

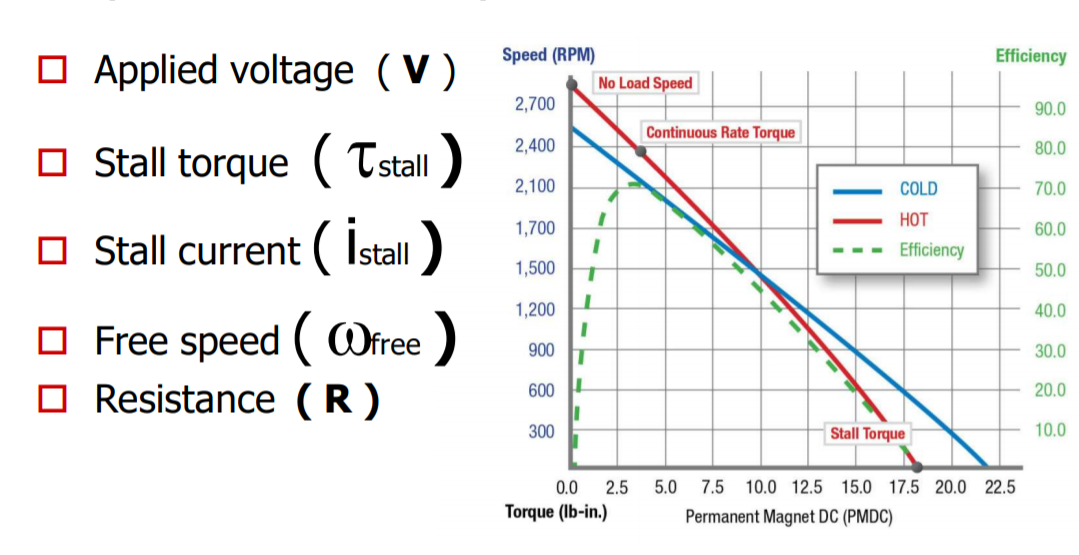
( nền trắng)nó sẽ làm cho các chân tín hiệu tại các chân OUT về mức 0, còn khi không nhận ( nền đen) thì nó sẽ ngắt chân tín hiệu ở các chân OUT về mức 1.



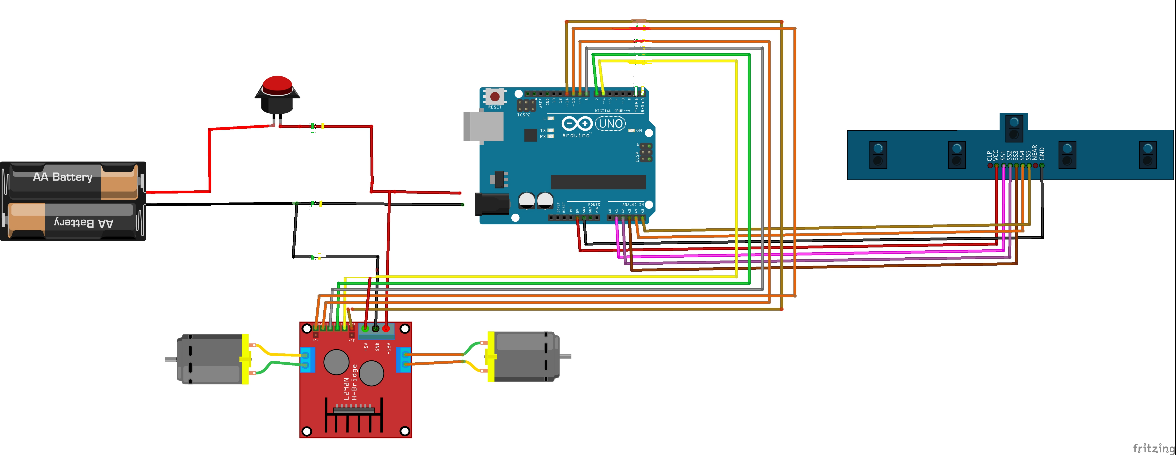
1. **Động cơ và bánh xe**





****

1. **SƠ ĐỒ LẮP GHÉP VÀ GIẢI THUẬT ĐIỀU KHIỂN**
2. **Sơ đồ nối các linh kiện**

****Cách thức nối các linh kiện như sau:

-Nối chân dương của nguồn vào chân +12V DC, nối chân âm của nguồn vào chân GND của L298

-Nối nguồn 2 động cơ vào L298

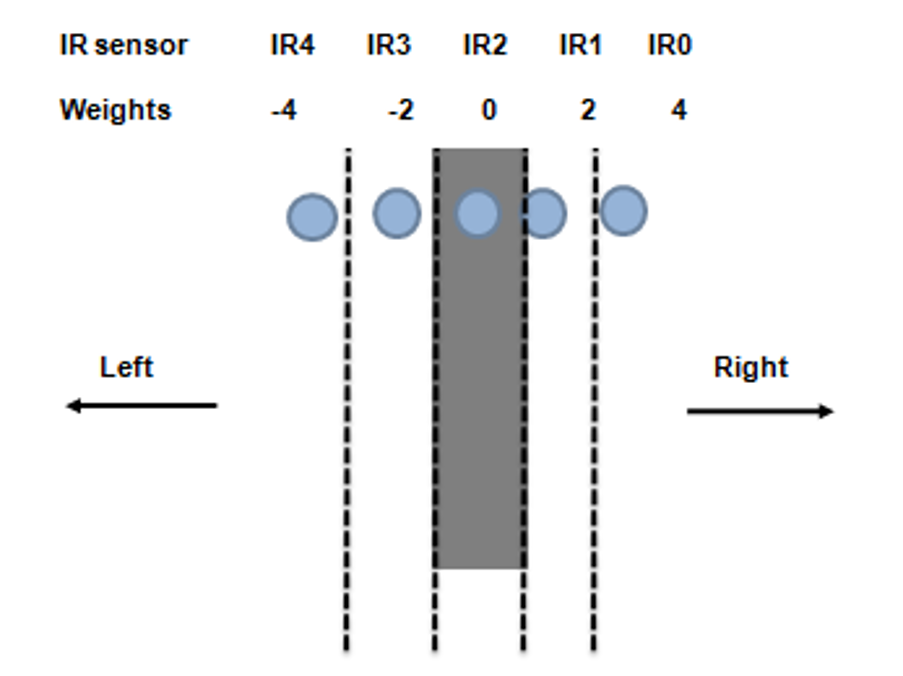
-

-

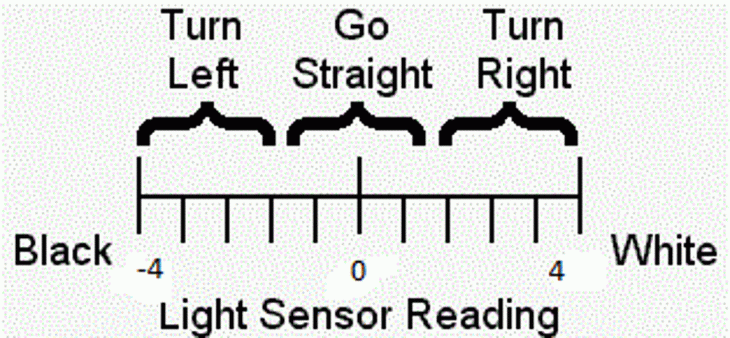
- Ở đây mình sử dụng 2 động cơ DC 5V và số lượng cảm biến HC-IR01 là 5.

- Lý do mình sử dụng 5 cảm biến hồng ngoại để tăng độ chính xác cho xe

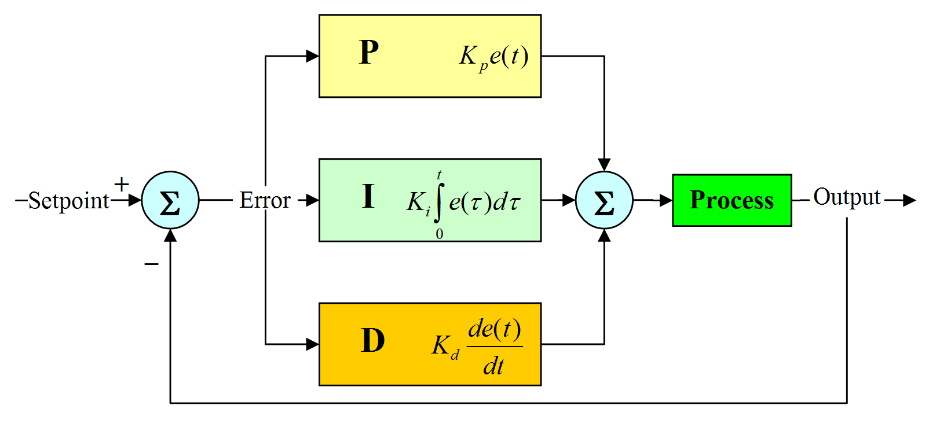
1. **Thiết kế thuật toán**
   1. **Giải thuật dò line**



Ở đây mình mã hóa 5 cảm biến của mình thành các giá trị từ -4 đến 4. Khi hoạt động ánh sáng từ led chiếu xuống đường nếu gặp đường nếu gặp đường nền màu trắng thì ánh sáng phản chiếu trở lại quang trở tương ứng, lúc đó tín hiệu được truyền về chân tương ứng của vi điều khiển là mức 0. Nếu ánh sáng từ led xuống gặp line đen thì ánh sáng hầu như bị hấp thụ gần hết, lúc đó tín hiệu từ sensor báo về vi điều khiển về mức 1.



* 1. **Giải thuật PID điều khiển tốc độ 2 động cơ xe:**



PID là một bộ điều khiển sử dụng cơ chế phản hồi feedback trong hệ thống công nghiệp. PID sẽ tính toán các giá trị “ sai số” là hiệu số đo được của thiết bị trên thực tế và giá trị mà mình mong muốn. Giải thuật tính toán của PID bao gồm 3 thông số riêng biệt, gọi là điều khiển 3 khâu: các giá trị **tỉ lệ ( Proportional), tích phân ( Integral), đạo hàm ( Derivative),** giá trị **tỉ lệ** xác định của sai số hiện tại, giá trị **tích phân** xác định tác động của các sai số quá khứ và **vi phân** xác tác động của tốc độ biến đổi sai số. Nhờ vậy mà ta có mối quan hệ: **P** phụ thuộc vào sai số hiện tại, **I** phụ thuộc vào tích lũy sai số trong quá khứ và **D** dự đoán các sai số trong tương lai, dựa vào tốc độ thay đổi của hiện tại.

**MV**(t)=**P**out + **I**out + **D**out

1. Khâu tỉ lệ ( Proportional)

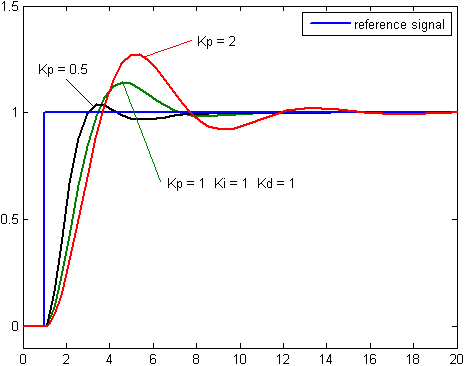
Trong đó

: tỉ lệ đầu ra

: là độ lợi tỉ lệ, thông số điều chỉnh

: sai số .

: thời gian tức thời.



1. Khâu tích phân ( Integral)

Trong đó:

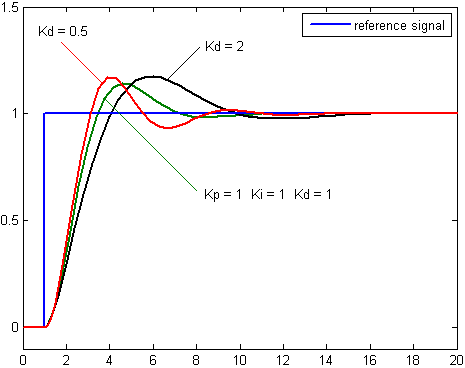
: thừa số tích phân của đầu ra

: độ lợi tích phân, 1 thông số điều chỉnh

: sai số

: thời gian tức thời

: một biến trung gian



1. Khâu vi phân ( Derivational)

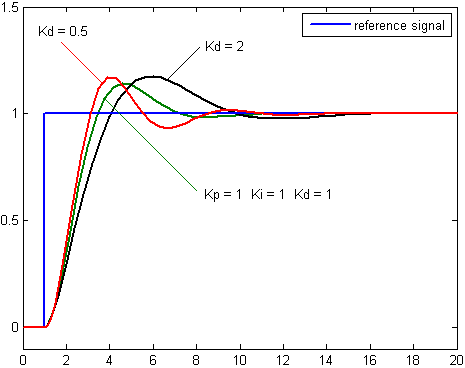
Trong đó:

: thừa số vi phân đầu ra

: Độ lợi vi phân, một thông số điều chỉnh

: sai số

: thời gian tức thời



1. Tóm lại

**Độ lợi tỉ lệ, {\displaystyle K\_{p}}**

Giá trị càng lớn thì đáp ứng càng nhanh do đó sai số càng lớn, bù khâu tỉ lệ càng lớn. Một giá trị độ lợi tỉ lệ quá lớn sẽ dẫn đến quá trình mất ổn định và dao động.

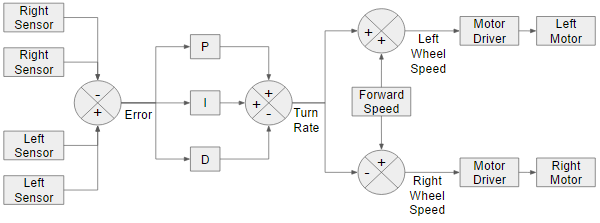
**Độ lợi tích phân, {\displaystyle K\_{i}}**

Giá trị càng lớn kéo theo sai số ổn định bị khử càng nhanh. Đổi lại là độ vọt lố càng lớn: bất kỳ sai số âm nào được tích phân trong suốt đáp ứng quá độ phải được triệt tiêu tích phân bằng sai số dương trước khi tiến tới trạng thái ổn định.

**Độ lợi vi phân, {\displaystyle K\_{d}}**

Giá trị càng lớn càng giảm độ vọt lố, nhưng lại làm chậm đáp ứng quá độ và có thể dẫn đến mất ổn định do khuếch đại nhiễu tín hiệu trong phép vi phân sai số.

1. **Phân tích chuyển động và điều khiển**
   1. **Giải thuật điều khiển**



Chương trình điều khiển sử dụng giải thuật PID. Ở đây mình mã hóa 5 cảm biến thành các giá trị từ -4 đến 4. Và thuật toán sẽ luôn đặt setpoint =0 để xe luôn bám theo vạch. Bằng việc code mình đã chia nhỏ độ phân giải của cảm biến thành các giá trị -4 -3 -2 -1 0 1 2 3 4 thay vì -4 -2 0 2 4.

Các bước thực hiện chương trình PID như sau:

P=error;

I=I + previous\_I;

D= error - previous\_error;

PID\_value= (Kp \* P) + (Ki \* I) + (Kd \* D);

previous\_I=error;

previous\_error = error;

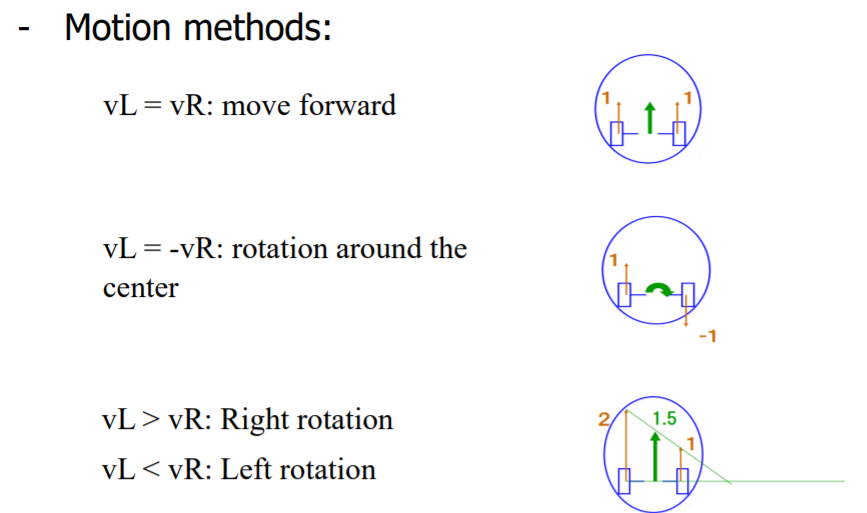
Trong đó:

P là sai số hiện tại

I là sai số trong quá khứ

D là sai số tương lai

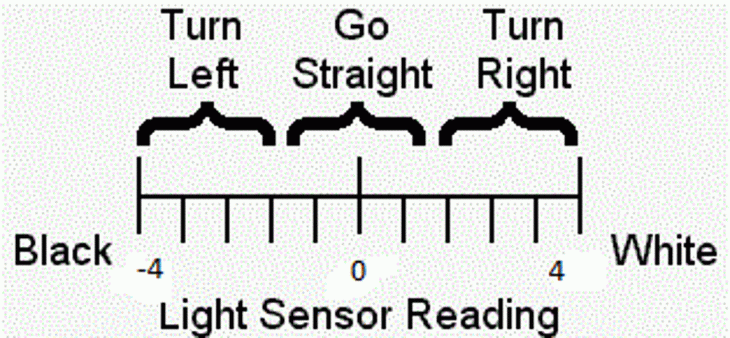
* 1. **Phân tích chuyển động**

****

- Khi vận tốc của hai bánh bằng nhau sẽ tiến thẳng

- Khi hai bánh có cùng vận tốc nhưng ngược chiều robot sẽ quay tròn

- Khi vận tốc của bánh trái lớn hơn bánh phải thì robot rẽ phải, bánh phải lớn hơn bánh trái thì quay trái



* Khi error =0 robot tiến thẳng.
* Khi error = -3 -4 robot rẽ trái.
* Khi error = 3 4 robot rẽ phải.

1. **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**
2. **Kết quả**

* Độ chính xác: Chưa cao
* Vọt lố: Lớn
* Những điểm rẽ mạnh: Chưa ổn định
* Tốc độ của xe: Thấp

Nguyên nhân:

* Do về mặt cơ khí.
* Do tốc độ 2 bánh chưa đều.
* Do quán tính.

Khắc phục:

* Thiết kế chính xác về cơ khí.
* Khắc phục động cơ.
* Cải thiện về phần code

1. **Hướng phát triển**

* Tiếp tục nghiên cứu về xe dò line và hướng đến dò line tránh vật cản.
* Cải tiến xe nhanh hơn, chính xác hơn bằng việc có thể sử dụng động cơ có encoder.
* Xây dựng thuật toán hoàn chỉnh và chính xác.

1. **TÀI LIỆU THAM KHẢO**

<https://vietmachine.com.vn/dieu-khien-arduino-dc-l298n-pwm-mach-cau-h.html>

<http://arduino.vn/bai-viet/42-arduino-uno-r3-la-gi>

<https://courses.uet.vnu.edu.vn/pluginfile.php/22650/mod_resource/content/1/6.Motion%20Mechanism.pdf>

<https://obitvn.wordpress.com/2019/05/24/arduino-pid-robot-do-line/>