**라인트레이서 요약**

우리의 라인트레이서는 Choco pie 라인트레이서를 이용하고 scratch를 이용하여 코딩을 실시한다. scratch와 라인트레이서에 대해 간단하게 언급하자면 **라인트레이서**는 규격에 맞게 조립을 실시한 뒤, Choco pie 홈페이지에서 scratch를 다운을 받는다. 이어 블루투스와 페어링을 실시하여 원격제어를 성공 시킨 후 scratch로 코딩을 실시한다. **scratch**는 블록화 되어있는 함수들을 마우스로 끌어와 값을 입력하여 제어를 한다. 다음은 scratch 사용방법과 간략한 알고리즘을 설명하겠다.  
  
 앞에서 언급하였듯이 scratch는 블록화 되어있는 함수를 마우스로 이동하여 코딩을 한다. 우리는 라인트레이서를 사용할 것이기에 라인트레이서의 제어와 각 센서의 활용을 다룰 것이다. 크게 이동, 정지, 라인을 따라가는 것이 주가 될 것이다.  
  
1. 라인트레이서 하단 3개의 센서를 통하여 라인이 있는 곳을 구분한다.   
2. DC모터에 운동방향과 값을 입력하고 라인에서는 조건문으로 이동과 정지를 한다.  
3. 라인이 조금 벗어나기 시작하면 센서를 통하여 라인을 추적하여 이동한다.  
4. 라인을 벗어나면 다시 정지한다.  
  
 대략적인 라인트레이서의 방법이다. 현재 우리의 구현은 3, 4번을 구현 중에 있다. 이어서 구현 중인 단계별 설명이다. **1번**은 하단에 IR이라는 센서가 세가지 있다. 이 세가지 센서의 거리 값을 통하여 제어를 실행한다. 우리는 중간의 IR2의 값이 IR1, IR3와 동일해지거나 라인의 특정 값이 들어오는 것을 통하여 구분을 한다. **2번**은 1번에서 라인 구분한 것을 이용하여, 산술연산자와 조건문으로 이동과 정지를 실시한다. **3번**은 1번에서 라인 구분한 것을 이용하여, 센서 값에 따라 좌∙우 모터의 운동값을 달리하여 방향 조절을 가능하게 한다. 현재 구현은 하였으나 자연스러운 이동과 커브구간 이동이 되지 않아 수정되어야 할 사안이다. **4번**은 세가지가 모두 일반 바닥의 값을 가지게 되면 정지하는 것으로 설정하였다.  
 현재 우리는 일반 바닥의 거리 값, 즉 IR값을 200으로 설정하여 제어를 실행하고 있다. 그리고 제어의 대부분이 if문을 통하여 실행이 되고 있으며, 좌∙우 모터의 운동 값의 변화는 한가지 경우로 규정을 하고 있으나, 커브의 정도에 따라서 추가 설정이 되어야 할 것이다.

Ex) if( 200 > IR2 && 200 > IR1 && 200 > IR3 )  
 STOP  
 else  
 GO