아두이노 시뮬레이션 프로그래밍

v1.0

김영준 저

공학박사, 목원대학교 겸임교수

前 Microsoft 수석연구원

헬로앱스

http://www.helloapps.co.kr

09 로봇 주행 기본 명령어

학습 목표

- 시뮬레이션 로봇의 동작을 제어할 수 있다.
- S자 미션을 완수할 수 있다.

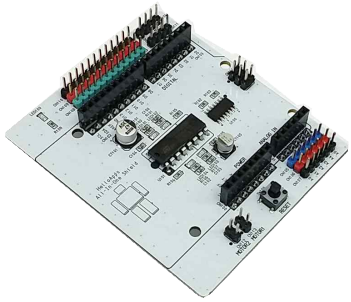
실습 개요

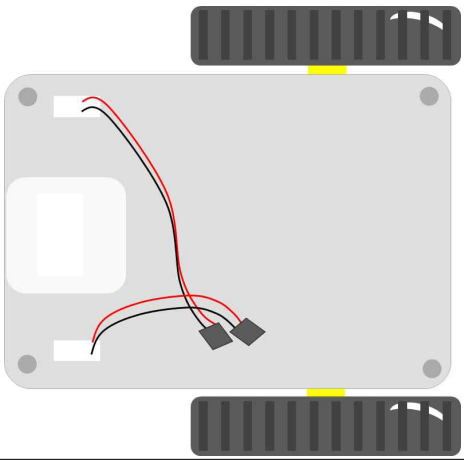
- 로봇 주행 제어 명령어를 구현해 본다.
- 춤추는 로봇 동작을 구현해 본다.
- 3D 장애물을 추가하는 명령어를 배워 본다.
- S자 미션을 수행해 본다.

9.1 준비하기

준비물

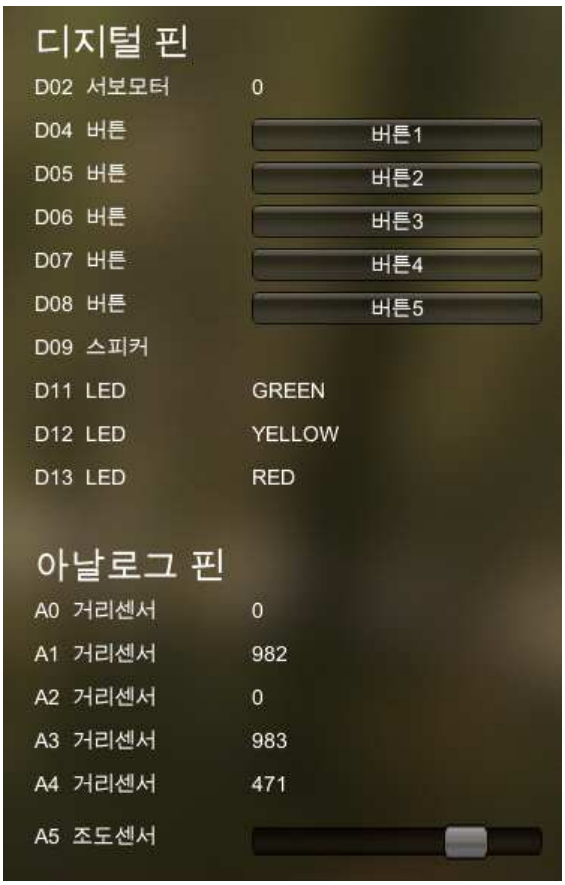
■ 아두이노 보드, 올인원 쉴드, 아날로그 PSD센서, LED 모듈, 스피커

	
아두이노 우노보드	올인원 쉴드

	
모바일 로봇 플랫폼	

시뮬레이션 상에서의 연결 정보

- 시뮬레이션 상에서는 디지털 LED 소자가 각각 디지털 11번, 12번, 13번에 연결되어 있으며, 차단기에 부착된 전방 PSD 거리센서는 아날로그 4번에 연결되어 있다.



- 디지털/아날로그 핀에 연결된 부품
  - 디지털 4번 ~ 8번: 디지털 버튼
  - 아날로그 0번: 로봇 우측 PSD 거리 센서
  - 아날로그 2번: 로봇 전방에 있는 PSD 거리 센서
  - 아날로그 4번: 차단기에 부착된 PSD 거리 센서

9.2 로봇 주행 제어 명령어

로봇 주행 명령어

■ 로봇을 제어하는 명령어는 DriveWrite 명령어이다. 내장 명령어의 [2-2-1] 명령어를 사용한다.

DriveWrite(왼쪽 모터값, 오른쪽 모터값)

- 모터 파워값: -255 ~ 255
- 모터 파워값은 -255에서 255 사이의 아무 값이나 사용할 수 있다.
- 값이 클수록 모터의 회전 속도가 크다.

항목	값 (왼쪽 모터값, 오른쪽 모터값)
전진	255, 255
정지	0, 0
후진	-255, -255
회전	-200, 200

전진하기

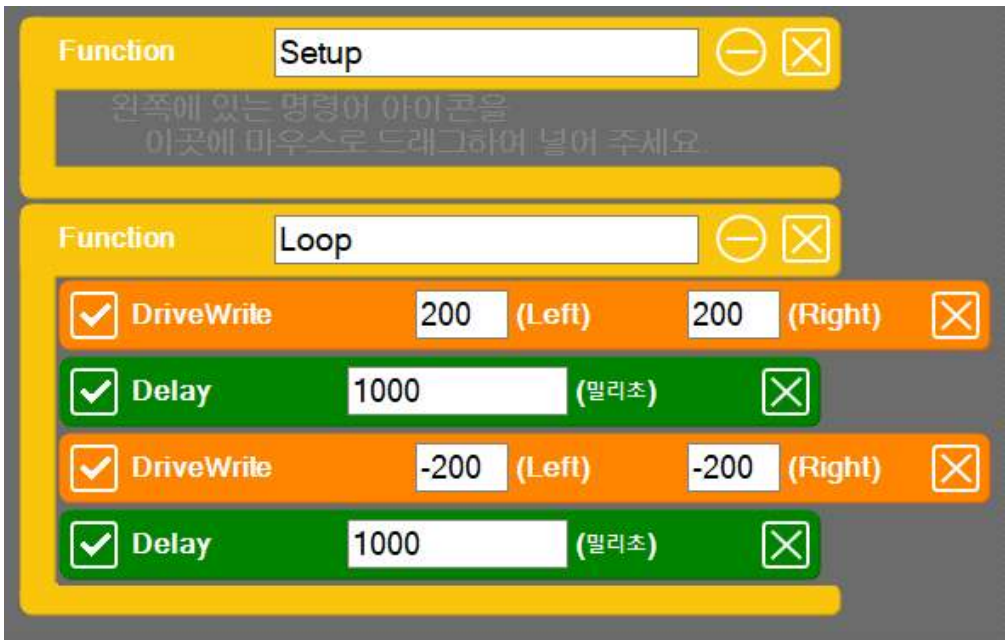
■ 다음과 같이 로봇이 전진하는 기능을 구현해 본다.

SPL 스크립트	스케치 코드

void setup()
{
 	
}
 	
void loop()
{
  DriveWrite(200, 200)
 	
  Delay(1000)
}
	void setup()
	{
	
	}
	
	void loop()
	{
	  DriveWrite(200, 200);
	
	  delay(1000);
	}

전진과 후진 반복하기

- 다음과 같이 전진과 후진을 반복하는 기능을 구현해 본다.
- Shift 키를 누른채 블록을 이동시키면 해당 블록 명령어가 복사된다.

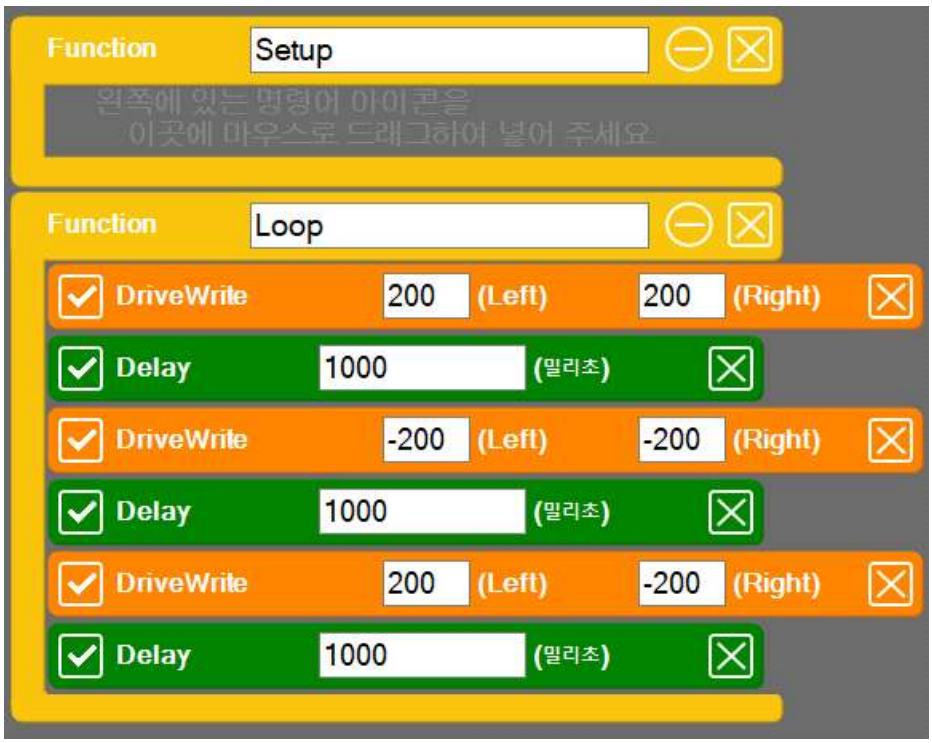


SPL 스크립트	스케치 코드

void setup()
{
 	
}
 	
void loop()
{
  DriveWrite(200, 200)
  Delay(1000)
  DriveWrite(-200, -200)
  Delay(1000)
}
	void setup()
	{
	
	}
	
	void loop()
	{
	  DriveWrite(200, 200);
	  delay(1000);
	  DriveWrite(-200, -200);
	  delay(1000);
	}

전진, 후진, 회전

■ 다음과 같이 전진과 후진, 회전을 반복하는 기능을 구현해 본다.

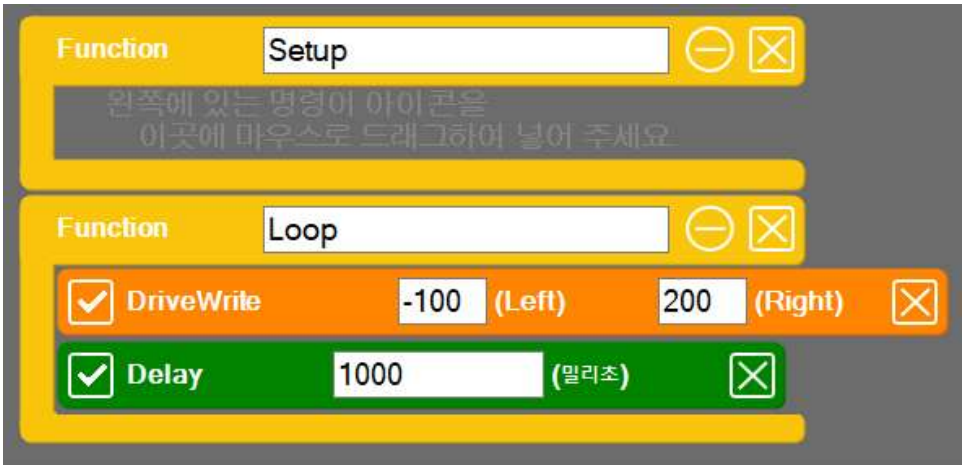


SPL 스크립트	스케치 코드

void setup()
{
 	
}
 	
void loop()
{
DriveWrite(200, 200)
Delay(1000)
DriveWrite(-200, -200)
Delay(1000)
DriveWrite(200, -200)
Delay(1000)
}
	void setup()
	{
	
	}
	
	void loop()
	{
	DriveWrite(200, 200);
	delay(1000);
	DriveWrite(-200, -200);
	delay(1000);
	DriveWrite(200, -200);
	delay(1000);
	}

축을 중심으로 회전

■ 한 쪽 바퀴 축을 중심으로 회전하는 기능은 다음과 같이 구현한다.



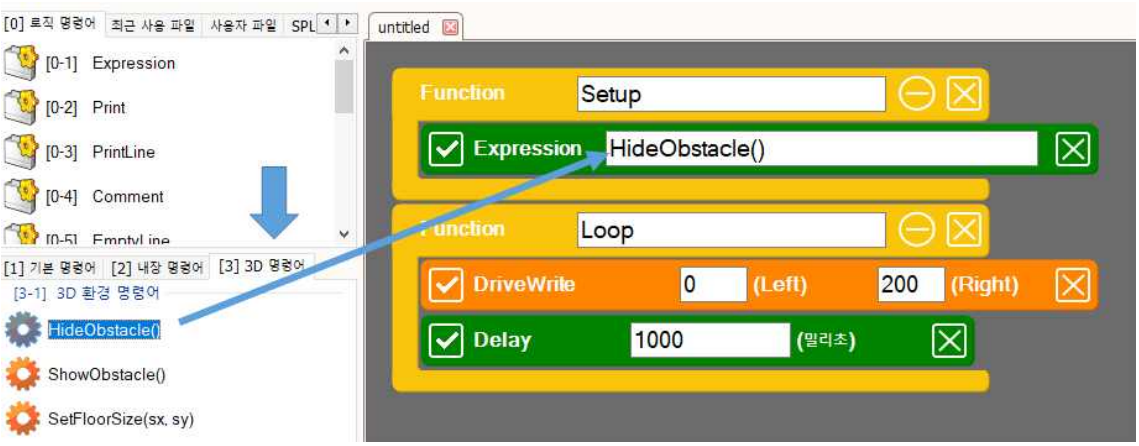
SPL 스크립트	스케치 코드

void setup()
{
 	
}
 	
void loop()
{
  DriveWrite(-100, 200)
  Delay(1000)
}
	void setup()
	{
	
	}
	
	void loop()
	{
	  DriveWrite(-100, 200);
	  delay(1000);
	}

9.3 춤추는 로봇 구현하기

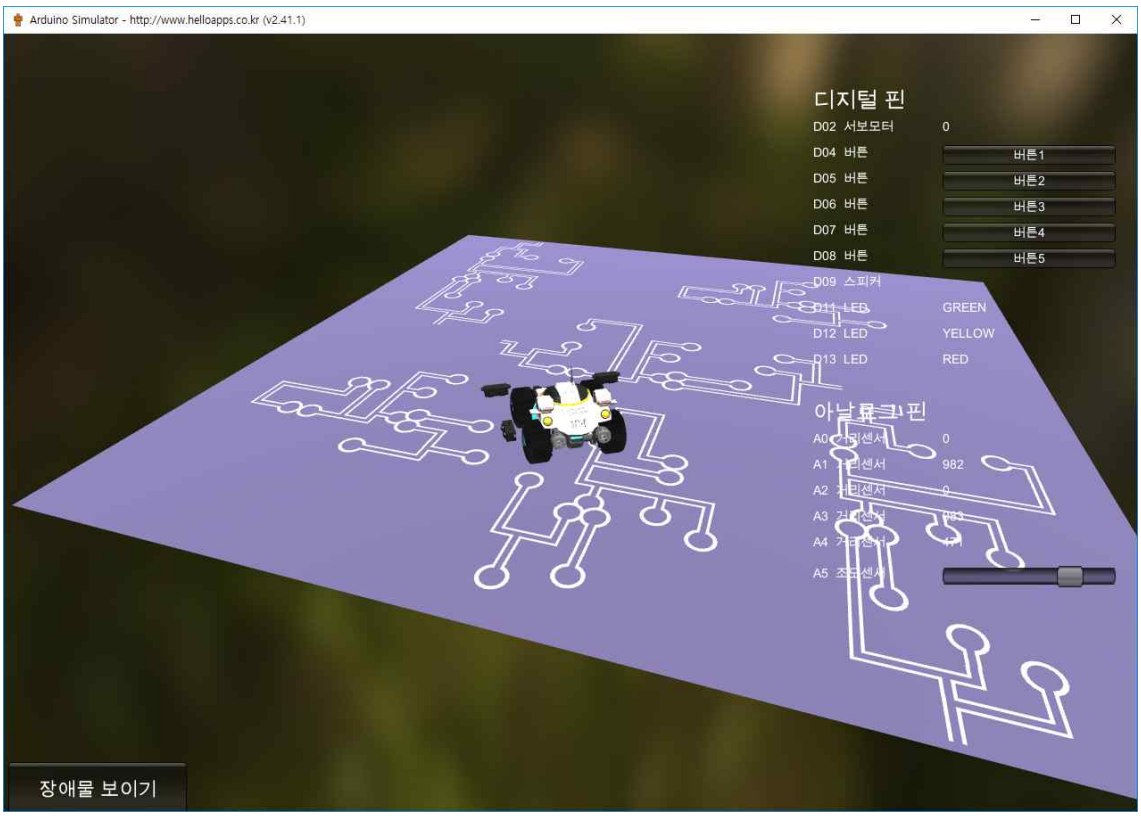
장애물 없애기

- 프로그램 실행시 자동으로 장애물이 없어지도록 하려면 다음과 같이 setup 함수에 장애물을 숨기는 명령어를 추가해 준다.
- 장애물을 숨기는 명령어는 HideObstable() 명령어로서 [3] 3D 명령어 탭에 있는 첫 번째 명령어이다.



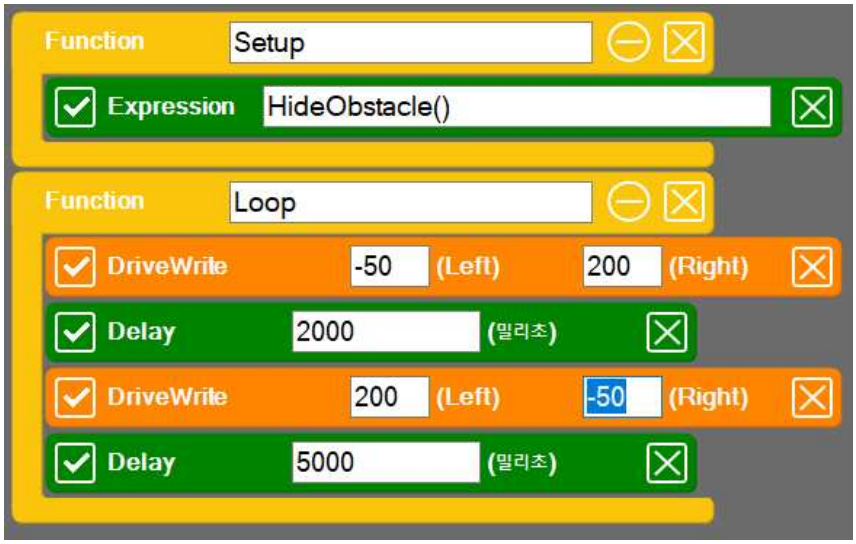
SPL 스크립트	스케치 코드

void setup()
{
HideObstacle()
}
 	
void loop()
{
DriveWrite(0, 200)
Delay(1000)
}
	void setup()
	{
	HideObstacle();
	}
	
	void loop()
	{
	DriveWrite(0, 200);
	delay(1000);
	}


Arduino Simulator - http://www.helloapps.co.kr (v2.41.1)
디지털 핀
D02 서보모터 0
D04 버튼 버튼1
D05 버튼 버튼2
D06 버튼 버튼3
D07 버튼 버튼4
D08 버튼 버튼5
D09 스피커
D11 LED GREEN
D12 LED YELLOW
D13 LED RED
아날로그 핀
A0 조도센서 0
A1 조도센서 982
장애물 보이기

원을 그리는 동작

■ 로봇이 원을 그리는 동작을 구현해 본다.



SPL 스크립트	스케치 코드

void setup()
{
HideObstacle()
}
 	
void loop()
{
DriveWrite(-50, 200)
Delay(1000)
DriveWrite(200, -50)
Delay(1000)
}
	void setup()
	{
	HideObstacle();
	}
	
	void loop()
	{
	DriveWrite(-50, 200);
	delay(1000);
	DriveWrite(200, -50);
	delay(1000);
	}

나만의 창의적인 동작 구현하기

- 모터 속도와 Delay 시간 값을 제어하여 나만의 창의적인 다양한 로봇 동작을 구현해 본다.

9.4 랜덤 동작 구현하기

랜덤 함수

- 랜덤 함수는 임의의 값을 생성해 주는 명령어이다.

```
a = random(5000)
```

- 위의 명령어를 실행하면 0 ~ 4999 사이의 임의의 값이 하나 생성된다.

- 아래의 명령어는 생성되는 값의 범위를 지정한 것이다.

```
a = random(2000, 5000)
```

- 위의 명령어를 실행하면 2000 ~ 4999 사이의 임의의 값이 하나 생성된다.

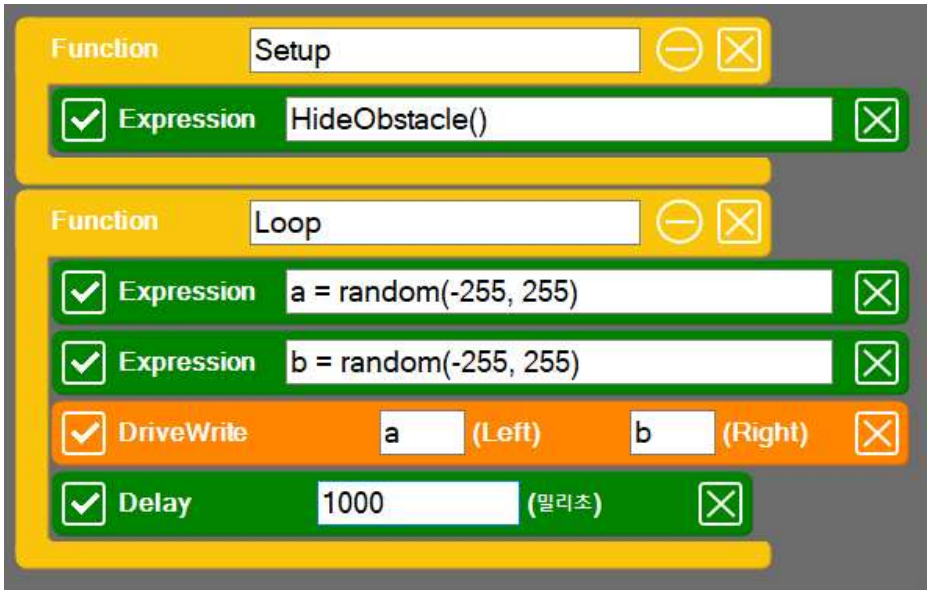
- 다음과 같이 명령어를 작성하여 0.1초 간격으로 임의의 값을 생성하여 출력해 보자.

SPL 스크립트	스케치 코드

void setup()
{
 	
}
 	
void loop()
{
a = random(100, 5000)
PrintLine(a);
Delay(100)
}
	void setup()
	{
	Serial.begin(115200);
	}
	
	void loop()
	{
	int a = random(100, 5000);
	Serial.println(a);
	delay(100);
	}

랜덤 함수로 파워값 생성하기

■ 다음과 같이 랜덤 함수로 모터의 파워값을 생성해 본다.

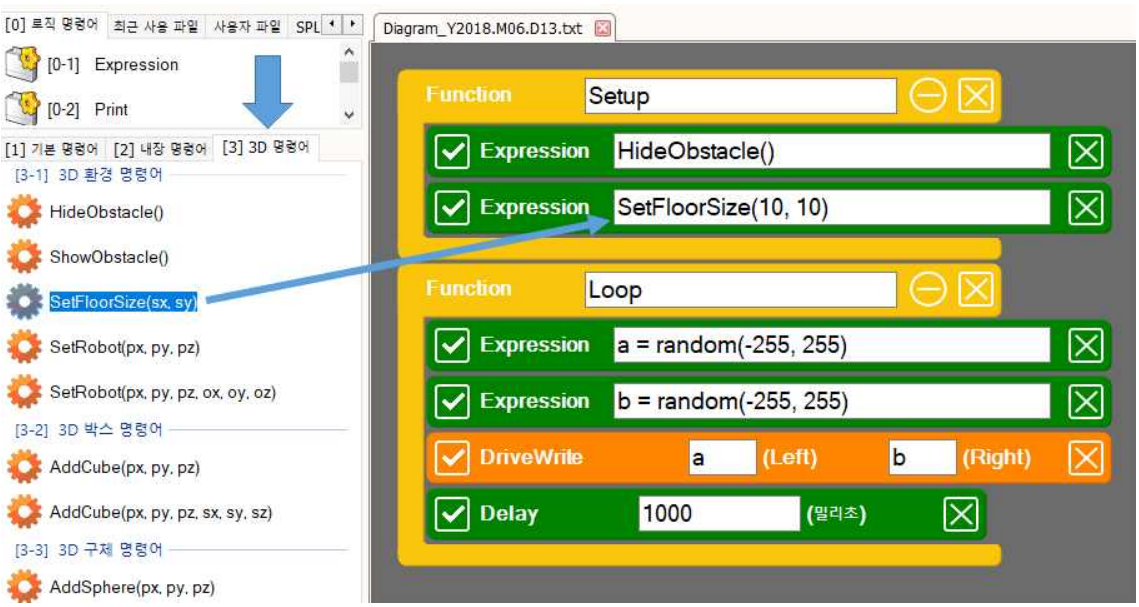


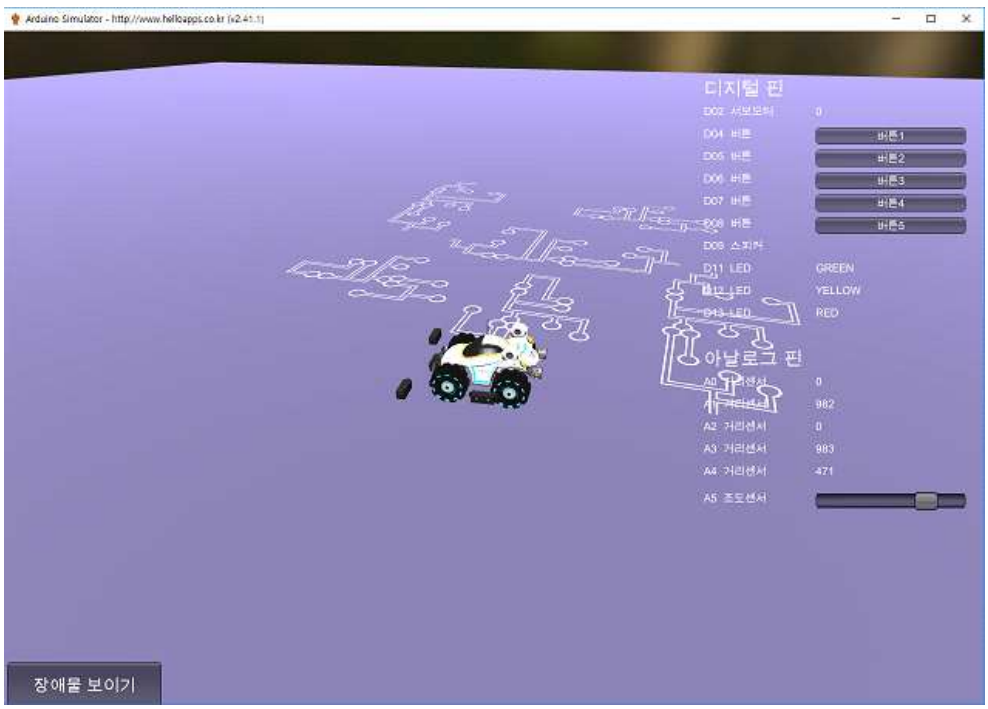
SPL 스크립트	스케치 코드

void setup()
{
  HideObstacle()
}
 	
void loop()
{
  a = random(-255, 255)
  b = random(-255, 255)
  DriveWrite(a, b);
  Delay(100)
}
	void setup()
	{
	  HideObstacle();
	}
	
	void loop()
	{
	  int a = random(-255, 255);
	  int b = random(-255, 255);
	  DriveWrite(a, b);
	  delay(1000);
	}

바닥판 크게 하기

- 다음과 같이 3D 명령어 탭에 있는 SetFloorSize 명령어를 setup 함수에 추가한 후, 값을 각각 10, 10으로 수정하여 바닥판이 10x10 배로 커지도록 한다.





SPL 스크립트	스케치 코드

void setup()
{
HideObstacle()
SetFloorSize(10, 10)
}
 	
void loop()
{
a = random(-255, 255)
b = random(-255, 255)
DriveWrite(a, b);
Delay(100)
}
	void setup()
	{
	HideObstacle();
	SetFloorSize(10, 10);
	}
	
	void loop()
	{
	int a = random(-255, 255);
	int b = random(-255, 255);
	DriveWrite(a, b);
	delay(1000);
	}

랜덤 Delay 시간

■ 이번에는 마지막으로 delay 시간 까지도 랜덤 값으로 설정해 본다.

SPL 스크립트	스케치 코드

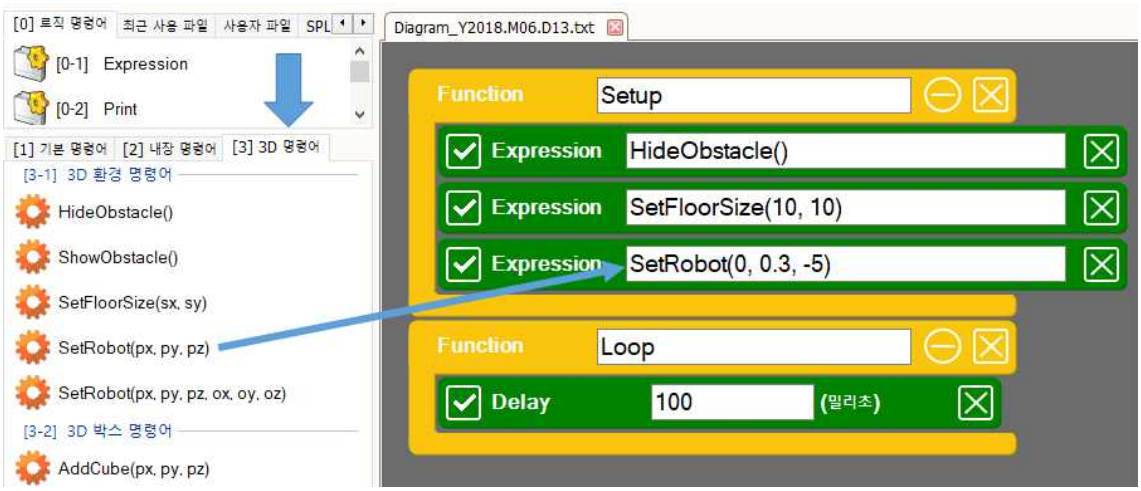
void setup()
{
HideObstacle()
SetFloorSize(10, 10)
}
 	
void loop()
{
a = random(-255, 255)
b = random(-255, 255)
d = random(100, 2000)
DriveWrite(a, b);
Delay(d)
}
	void setup()
	{
	HideObstacle();
	SetFloorSize(10, 10);
	}
	
	void loop()
	{
	int a = random(-255, 255);
	int b = random(-255, 255);
	int d = random(100, 2000);
	DriveWrite(a, b);
	delay(d);
	}

9.5 로봇의 위치 제어하기

로봇의 위치 설정

- 시뮬레이션 로봇은 필요한 시점에 로봇의 위치를 이동시킬 수 있다.
- 먼저 시작 시점에 다음과 같이 SetRobot 명령어를 이용하여 로봇의 위치를 변경시켜 보자.

SetRobot(x, y, z)

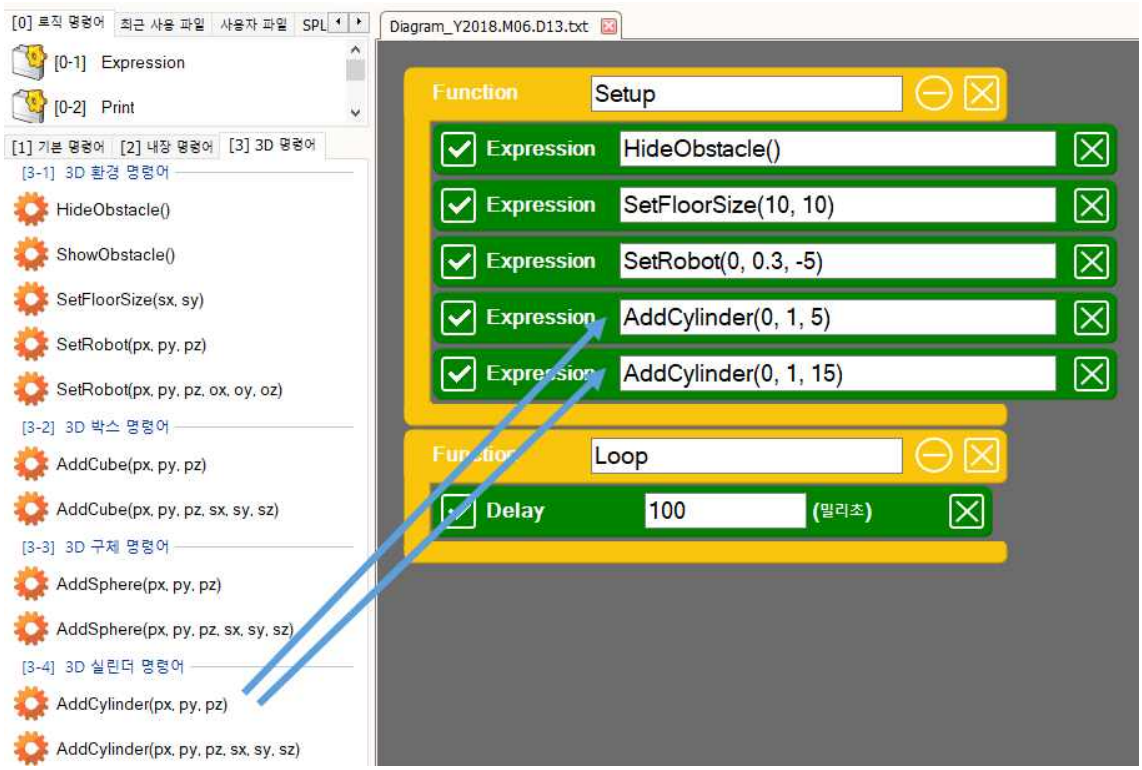


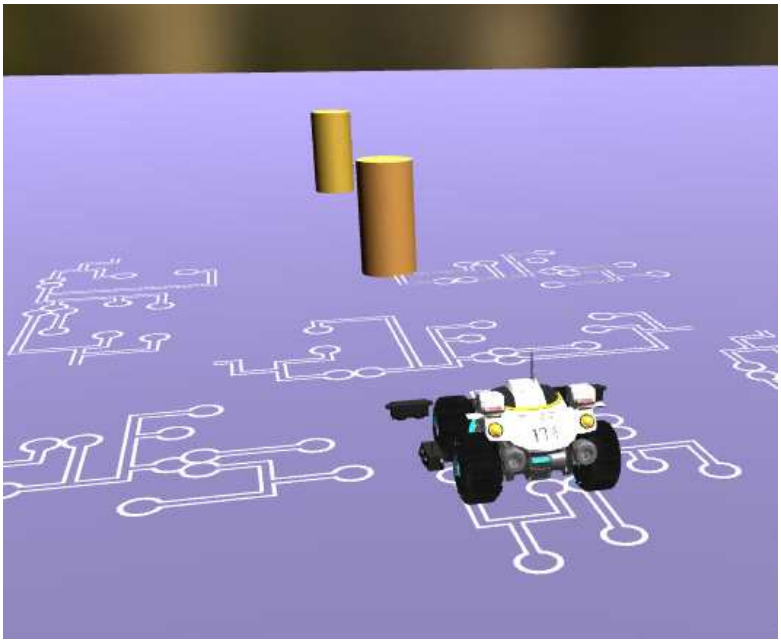
- 위의 명령어는 로봇을 Z축으로 -5m 이동시킨 것이다. (화면 앞쪽으로 이동)

9.6 장애물 추가하기

로봇 장애물 추가하기

- AddCylinder 명령어를 이용하여 다음과 같이 2개의 원통 기둥을 바닥면에 추가해 보자.





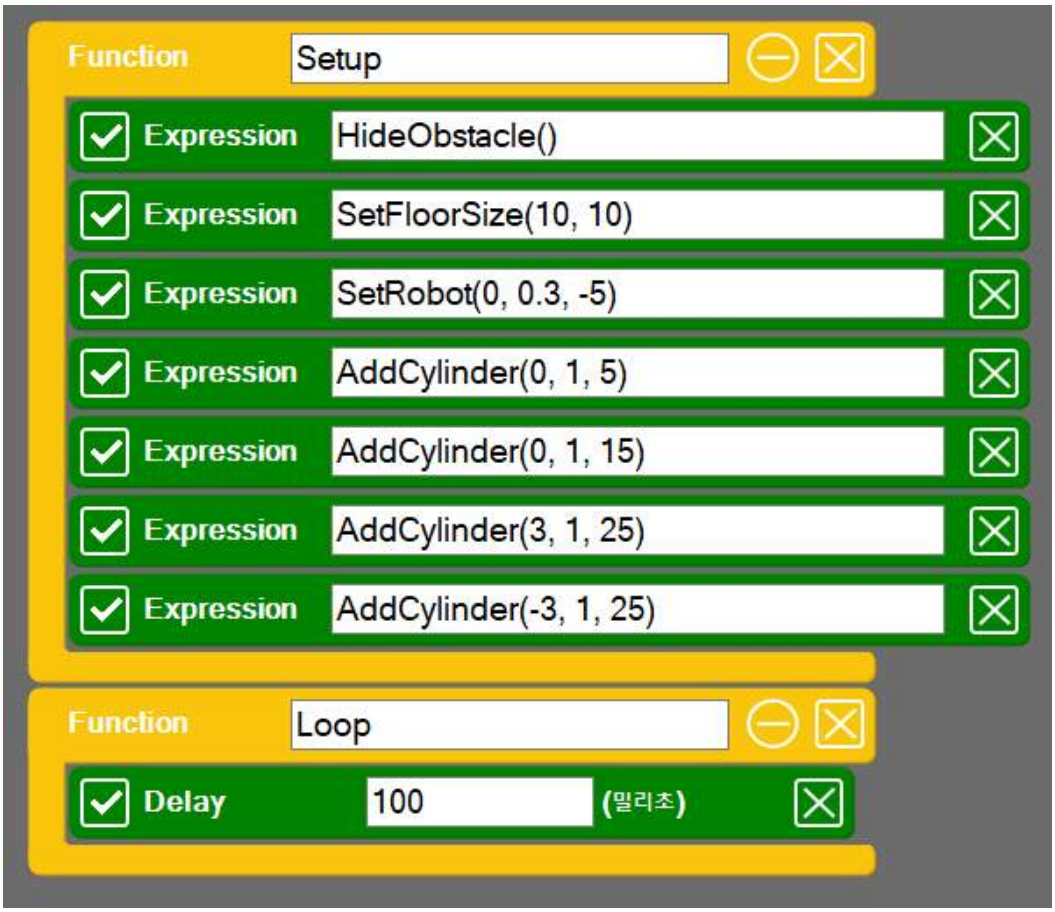
9.7 S자 미션 수행하기

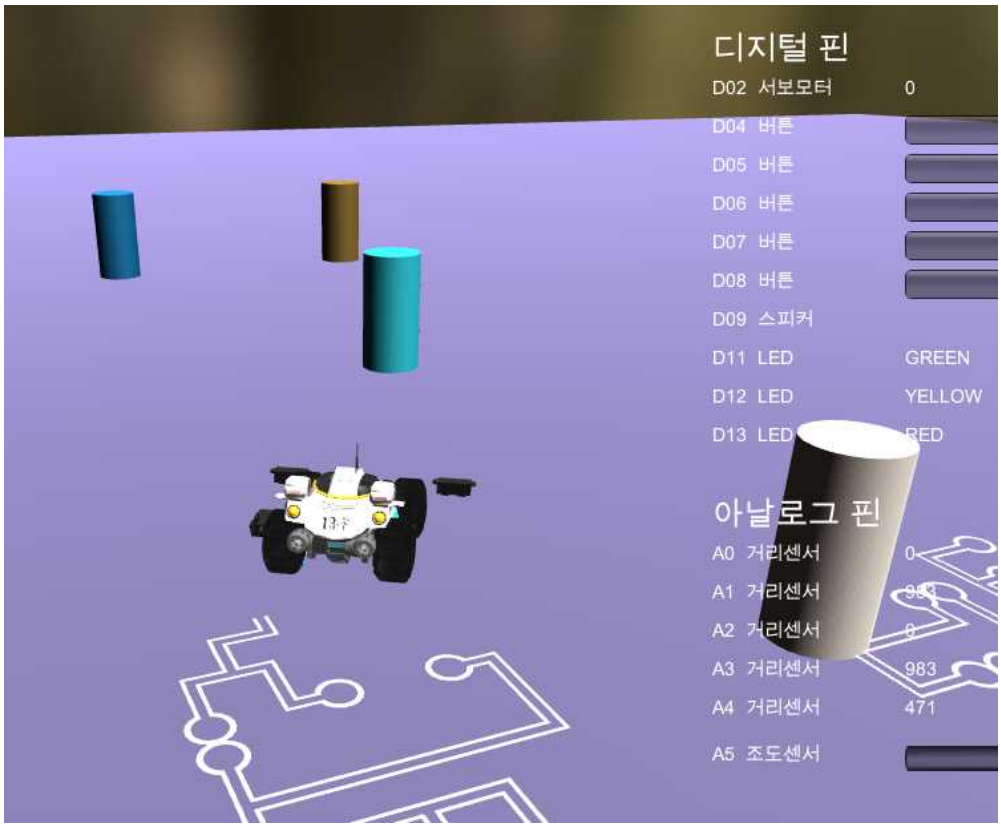
원기둥 빠져나가기 미션

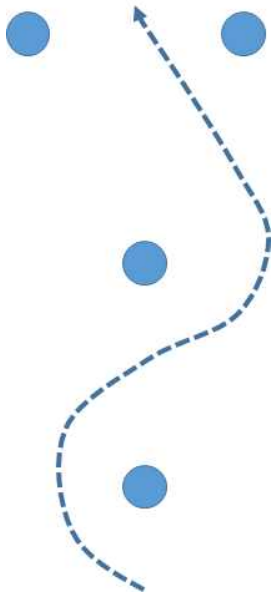
■ 로봇이 2개의 원기둥 사이를 빠져 나가는 미션을 구현해 본다.

S자 미션 수행하기

■ 원기둥 2개를 더 추가하여 다음과 같이 로봇이 S자로 원기둥을 빠져 나가도록 미션 환경을 구현해 본다.




디지털 핀
D02 서보모터 0
D04 버튼
D05 버튼
D06 버튼
D07 버튼
D08 버튼
D09 스피커
D11 LED GREEN
D12 LED YELLOW
D13 LED RED
아날로그 핀
A0 거리센서 0
A1 거리센서
A2 거리센서
A3 거리센서 983
A4 거리센서 471
A5 조도센서



■ 미션을 완성해 본다.