1. ROS란?

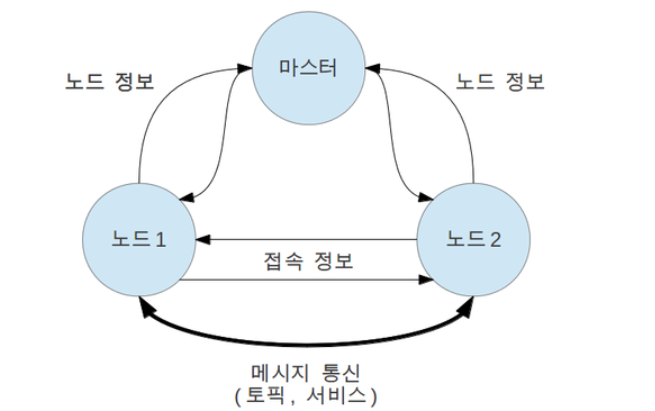
“Robot Operating System”, 로봇 응용 프로그램을 개발하기 위한 운영체제

1. ROS를 사용하는 이유

1) 서로 동시에 작동하는 프로그램 간 ROS가 들어있다면 간단하게 통신이 가능

2) 생태계 구축이 잘 되어있다: ROS를 사용하는 오픈소스 패키지의 종류가 굉장히 많고 접근이 쉽기 때문에 해당 패키지 안의 구동 방식을 자세히 이해할 필요 없이 자신이 원하는 분야에 적용할 수 있다.

1. ROS 개념, 용어 정리
2. 메시지 통신

로스의 핵심 개념은 노드간의 메시지 통신이다. 목적에 따라 세분화된 최소 단위의 실행 프로그램인 노드는 다른 노드와 처리 결과 값을 주고 받게 됨으로써 하나의 커다란 프로그램이 된다. 이를 위하여 메시지 통신에는 단 방향으로 연속적으로 메시지를 송수신하는 **토픽** 메시지 통신과 쌍 방향으로 요청과 응답 형태의 정해진 작업을 수행하게 하는 목적의 **서비스** 메시지 통신이 있다. 이러한 노드간의 메시지 통신에는 접속이 필요한데, 노드간의 접속을 돕는 것이 마스터이다. 마스터는 노드의 이름, 토픽 및 서비스의 이름, URI 주소와 포트, 매개변수 등의 네임 서버와 같은 역할을 하게된다. 즉, 노드는 생성과 동시에 마스터에 자신의 정보를 등록하게 되고, 다른 노드가 마스터를 통해 접속하려는 노드의 정보를 마스터로부터 취득한다. 그 후, 노드와 노드가 직접 접속하여 메시지 통신을 하는 것이다.

1. 노드

로스에서 최소 단위의 실행 프로세서를 가르키는 용어이다. 하나의 실행 가능한 프로그램으로 생각하면 된다.

로스에서는 하나의 목적에 하나의 노드를 작성하길 권하고 있으며 재사용이 쉽도록 구성하여 만들도록 권고하고 있다. 예를들어, 모바일 로봇의 경우, 로봇을 구동하기 위하여 각 프로그램을 세분화 시킨다. 예를들어, 센서 드라이브, 센서 데이타를 이용한 변환, 장애물 판단, 모터 구동, 엔코더 입력, 네이게이션 등 세부화된 작은 노드들을 이용한다.

노드는 생성과 함께 마스터에 노드이름, 발행자이름, 구독자이름, 토픽이름, 서비스이름, 메시지형태, URI 주소 및 포트를 등록한다. 이 정보들을 기반으로 각 노드는 노드끼리 토픽 및 서비스를 이용하여 메시지를 주고 받을 수 있다.

1. 메시지

노드는 메시지를 통해 노드간의 데이터를 주고받게 된다. 메시지는 integer, floating point, boolean 와 같은 변수형태이다. 또한, 메시지안에 메시지를 품고 있는 간단한 데이터 구조 및 메시지들의 배열과 같은 구조도 사용할 수 있다.

1. 토픽

토픽의 경우, 발행자 노드 및 구독자 노드 사이에서 전달되는 메시지의 주제라고 할 수 있다. 발행자, 구독자가 중지하지 않는 이상, 메시지가 연속적으로 발행되고 구독하게 된다.

1. 서비스

서비스는 토픽과 달리 1회에 한해 접속, 서비스 요청, 서비스 응답이 수행되고 서로간의 접속을 끊는다. 다시 필요한 경우 접속부터 다시 진행해야 한다.

1. ROS 설치

<http://wiki.ros.org/Installation/Ubuntu>

위의 사이트에 있는 melodic을 선택하여 가이드문서에 따라 설치를 진행한다.

1. Turtlesim 실습

<http://wiki.ros.org/turtlesim>

위의 사이트를 참고하여 진행한다.

1. Roscore을 실행한다.

**$ roscore**

새 마스터 노드를 생성하여 실행시킨다.

1. Turtlesim 시작하기

**$ rosrun turtlesim turtlesim\_node**

위 명령어를 입력하면 파란 배경의 거북이가 나타나고, 터미널에는 생성된 거북이의 좌표와 각도가 출력된다.

1. 방향키로 조종해보기

**$ rosrun turtlesim turtle\_teleop\_key**

명령어를 실행한 터미널에서 방향키를 누르면 거북이가 그에 맞춰 움직이는 것을 확인할 수 있다.

1. 명령어로 조종해보기

**$ rosnode list**

현재 실행중인 노드의 목록을 확인한다. /turtlesim 이 존재하는지 확인할 수 있다.

**$ rosnode info /turtlesim**

거북이노드와의 통신을 위한 정보를 확인할 수 있다.

**$ rostopic list**

거북이노드와의 통신할 메시지 주제들을 확인할 수 있다.

**$ rostopic echo /turtle1/pose**

현재 거북이의 좌표, 각도, 속도를 확인할 수 있다.

**$ rostopic type /turtle1/cmd\_vel**

통신할 메시지의 타입을 확인할 수 있다.

**$ rostopic pub -1 /turtle1/cmd\_vel geometry\_msgs/Twist**

위 명령어를 작성하고 Enter를 누르지 말고 tab을 누르면 거북이에게 이동에 대한 명령을 내릴 수 있는 인자 값들이 나타난다. 이를 수정한 뒤, Enter를 누르면 거북이가 해당 명령대로 움직이는 것을 확인할 수 있다.

**$ rosrun rqt\_graph rqt\_graph**

노드 간의 통신이 어떻게 이루어지는 지 그래픽으로 확인할 수 있다.