1. 로봇 소프트웨어 플랫폼

메인보드,메모리,그래픽카드,인풋관련,아웃풋관련

모듈형으로 되어있다. 내용물 여러회사에서 제작하여 하나의 제품 구성

하드웨어결합이 가능한 하드웨어 모듈로 구성되어있다.

운영체제 – Window, Linux 같은거

핸드폰도 마찬가지로 안드로이드 등..

+ 그걸 기반으로 한 어플리케이션 탑재

하드웨어모듈 + 운영체제 + 앱 + 유저 = 생태계 4대요소

이거 4개가 있어야지 대중화된 제품이 될수있다.

펌웨어에서 OS로.

회사마다 전용하드웨어, 전용 운영체제 => 현재 컴퓨터와 폰은 분업체제로 진행중

그렇다면 로봇분야는?

혼자 다해 – 다른 산업군에서는 볼수없는짓

소프트웨어 플랫폼으로 인해 얻어진 변화

-하드웨어 인터페이스 통합

-하드웨어 추상화.규격화.모듈화

가격 떨어지고 성능 업

하드웨어 / 운영체제 /. 어플리케이션 분리

사용자 수요에 맞는 서비스에 집중 가능하다! – 앱 개발이 하드웨어에 종속된 회사가 아니고, 운영체제에 종속된 회사에서 만들어내는 것이 아니다. ( 유저증가 )

현재 로봇분야는 여러 운영체제의 춘추전국시대와같다. ( 약 40종류 )

OpenRTM opros naoqi… ROS도 마찬가지

ROS를 가장 많은 사람들이 쓰고 있다. 이유는?

운영체제 만드는데 소모전은 그만하자.

로봇스프트웨어 플랫폼이 가져올 미래

* 모듈형 하드웨어 플랫폼 확산
* 하드웨어와 소프트웨어 플랫폼 간의 인터페이스 확립
* 하드웨어에 대한 지식 없이 응용프로그램 작성 가능
* 다른 군( 게임소프트웨어개발자, 앱 개발자 등등..)의 로보틱스 진입 가능 ( 기초지식이 줄어들어서 )

1. 로봇 운영체제 ROs 소개

ROS란 ? 오픈소스, 메타오퍼레이팅 시스템

핵심은 오픈소스를 기반으로 하여 많은 개발자에게 주목받고, 쓰기쉬움

메타오퍼레이팅시스템(메타운영체제)

– 기초(리눅스 등) 운영체제를 토대로(이용해먹는 것)

엄밀히따지면 OS는 아니야.

로봇을 위한 운영체제임

로봇소프트웨어 개발을 위한 소프트웨어 프레임워크

* 노드간 메시지교환방법으로 복잡한 프로그램을 잘게 나눠 공동개발 가능
* 편리한 툴 갖추고있다
* 모델링 센싱 인식 내비게이션 매니퓰레이션 등 많이 사용되는 것을 기본기능으로

로보틱스 소프트웨어 개발을 전세계 레벨에서 공동작업 가능하도록

이기종 디바이스 간의 통신 지원한다

ROS를 사용 가능한 운영체제

* 우분투, 윈도우 등등.. 우분투 OS X에서 구동하는 것 추천
* OS를 탑재할 수 없는 마이크로컨트롤러 유닛(MCU)의 경우 시리얼 통신, 블루투스, LAN경유로 통신할 수 있는 라이브러리 제공

어떤 언어를 쓰던.. 다양한 언어를 사용할 수 있고 그를 위해 다양한 CLientLayer

90종류 이상의 로봇, 80종류 이상의 센서 지원

ROS의 특징

통신인프라

노드 간 데이터 동신을 제공 ( 프로세스 간의 통신 )

메세지 파싱 기능 – 캡슐화 및 코드 재사용을 촉진하는 노드들 간의 메시지 전달 인터페이스

메시지의 기록 및 재생 – 송수신 데이터 저장하고 필요 시 재사용 가능

( 이건 알고리즘이 나아졌는지 판단할 때 사용 가능 )

자바로 짜든 C++로 짜든 각 노드는 서로 다른 언어로 작성 가능

( 클라이언트 라이브러리 : roscpp, rospy, roslisp, rosjava 등등 )

로봇 특화 다양한 기능

다양한 데이터에 대해 표준을 만들어놔서 효율성 향상

(로봇기하학 라이브러리) TF 제공 – 로봇과 센서등의 상대적 구조를 tree화 시키는 TF구조를 제공

로봇의 물리적 특성을 설명하는 XML문서 기술

로봇의 상태를 한눈에 파악하는 진단시스템

센싱 인식관련 알고리즘 제공

네비게이션 관련하여 제대로 잘 되어있음

Manipulation 라이브러리 통째로 제공 ( 우리랑은 노상관 )

다양한 개발도구

ToolBox ( 로봇개발에 필요한 ) – CommandLine Tool, RViz, RQT, Gazebo

코맨트라인툴 – 커맨드 창에서 명령어 쳐서 할 때 툴 지원

Rviz - 3차원 시각화 툴. 어떤 센서든지 시각화 가능

RQT – GUI 기반의 툴박스

Gazebo – 물리엔진 탑재. 실제 시뮬레이션 해볼 수 있는.

ROS 버전 선택

ROS Kinetic Kame (LTS) 추천

Ubunto 16.04 Xenial Xerus LTS

Gazebo 7.0

1. ROS 개발환경 구축

ROS설치 – 리눅스 명령어 + 링크로 설치합니다. ( 한줄설치 )

홈페이지 적혀있습니다. 버전 LTS 키네틱 카메 우리는 우분투

ROS 환경설정? Alias? 치트키,단축키의 느낌

4:12 ROS 환경설정

5:45 동작테스트 ( roscore, rosrun )

동작테스트 – 터미널창에(컨트롤알트T) roscore를 치면

터미널에 cm쳐봐라

끌 때는 컨트롤c 하면 프로세스 종료다.

리눅스 붙여넣기방법 ( 오른쪽 눌러서 )

리눅스에서 컨트롤 쉬프트 V를 눌러야 붙여넣기가된다.

ROS에서 사용가능한 통합개발환경 (IDE)

QTcreator + Qt Creator Plugin for ROS ( GUI 방식에서 자주 )

이렇게 같이쓰면 상당히 활용도가 높다

VIsualStudiocode – 가벼운 에디터 + RosExtension ( 추가플러그인 ) --- 간단한 프로그램

Eclipse ( 많은사람들이 사용하는 익숙한 통합개발환경이나 무겁다. )

1. ROS 의 중요 컨셉 – 중요

로봇운영체제 ROS 용어 ( 엄청 많고 여기서는 중요한 용어만 )

가장중요

Node – ROS의 대표적인 특징이다. 최소단위의 실행가능한 프로세서

예를들어 회사 출입문에 RFID 태그카드라든지, 지문이라든지가 있어. 혹은 얼굴인식

첫관문 - 카메라에서 영상 따올거야

두번째 관문 – 영상처리필터( 배경삭제같은거 처리, 상체 없애기, 얼굴만남기기, 특징점 찾아내기 )

세번째 관문 – 얼굴과 매칭작업

노드란건 카메라부터 데이터 받은거 – 한 개의 노드

배경처리필터 – 한 개의 노드

상체 없애는 필터 – 한 개의 노드

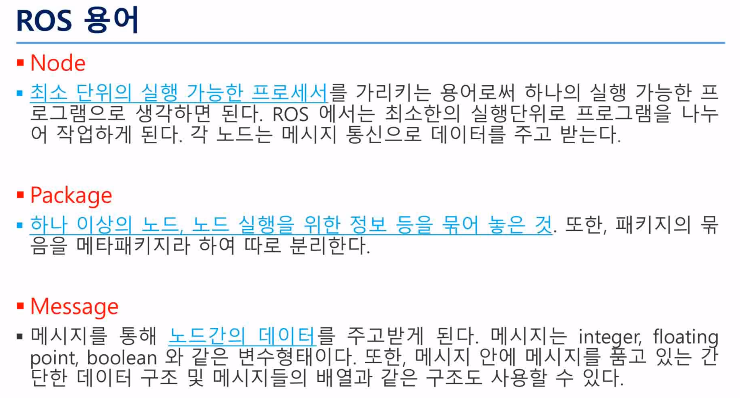
특징점 찾아내는 필터 – 한 개의 노드

이런 식으로 통으로짜는게 아니라 각각의 프로그램으로 작성하는 것.

각각의 실행가능한 프로그램으로 작성하여. 각각을 쪼개놓아 오픈해서 공개해놓으면 여러 사람이 가져다 쓸 수 있다.

단점 – 필터1에서 처리한걸 필터2에 어떻게 넘겨주지? 별개의 프로그램인데? 여기서 통신이 나와

ROS에서는 메시지통신이라고 해 – 로우데이터를 어떻게 보내는지, 그리고 받고나서 어떻게 전송할건지, IMU정보값, 뎁스카메라 값, 모터관절값 같은거 모두 메시지로 전송, 수신하기로되어있음.



Package-

20개의 노드를 하나의 목적으로 묶어놨을 때 하나의 실제적인 서비스를 제공하는것이잖아?

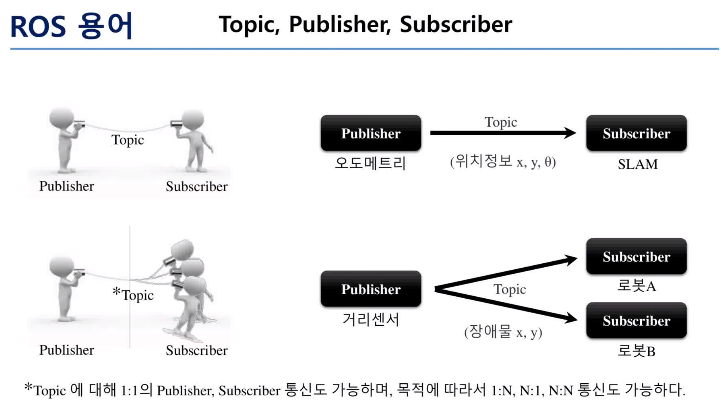
복수의 노드를 묶어놓은걸 패키지라고함. 또 패키지들을 다시 묶어서 메타패키지라고 함.

Message-

메시지 종류는 상당히 많다. 예를들어 모터값이라든지, IMU값이라든지, - 나중에 자세히

--------------------------------------------------------------------------------------------------------------

메시지 방식 – 토픽, 서비스, 엑션 이 있다!



ROS에서의 Topic 의 정의? – 단방향, 연속성을 가진 통신방법 ( 메시지통신방법중의 하나 )

일방적으로 데이터를 계속 보내야 하는 상황에서 많이 사용 ( 예를들어 센서데이터 – 주구장창보내 )

메시지 보내는 사람을 퍼블리셔(Publisher)라고 하고, 받는놈을 Subscriber라고 함.

퍼블리셔 노드, 섭스크라이버 노드 라고 부르는게 있음.

1대 다수도 된다. 카메라의 로우데이터를 B라는애 말고도 C,D,E에게 복수로 전송할 수 있다.

반대로 퍼블리셔가 N개, 섭스크라이버가 1개여도 가능하다.

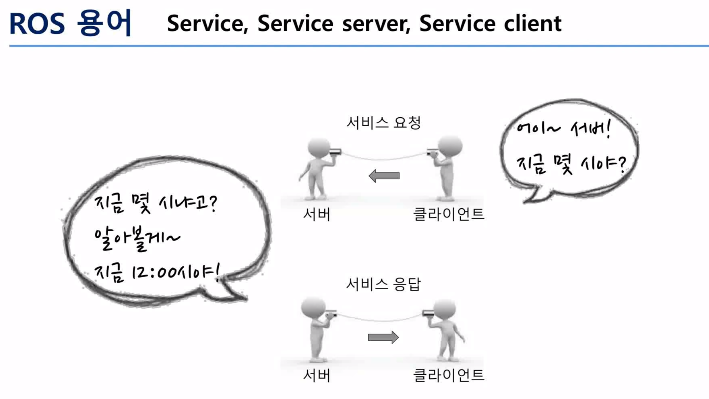
1:1 1:N N:1 N:N 의 경우 모두 가능하다.

ROS에서의 Service의 정의? – 양방향 ( 야 몇시야? – 대답까지 ) ( 서비스응답해주는 것 )

로봇으로 따지면. 너 어디로 가! 라고 명령 요청하면, 서비스서버는 동작관련 프로세스 처리

그다음에 처리됬으면 요청한거 끝났어! 라고 알려준다. 이럴 때 씀

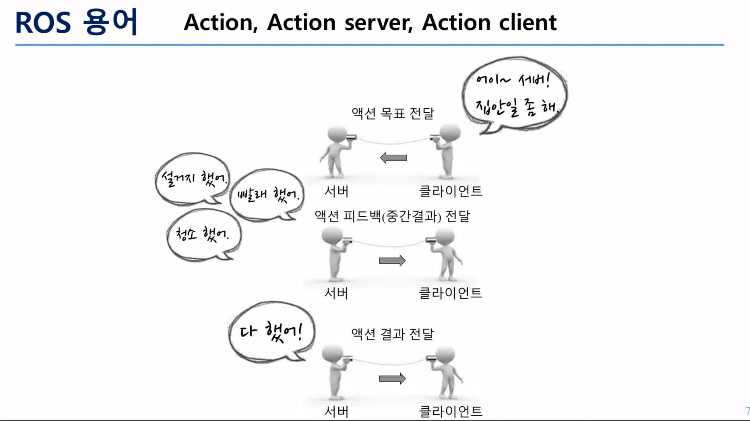
특징은 양방향!!( 주고받을 수 있다 ). 1회성이다.(위의 토픽처럼 주구장창보내는게아냐, 접속이 끊겨. 주고받거니 해서. 다시하려면 재접속해야돼)



ROS에서의 Action의 정의

클라이언트가 목표를 전달하면, 과정을 피드백으로 알려주고. 액션 결과를 다 했어! 라고 알려줌. 서비스랑 비슷하나 피드백을 계속 준다는게 장점. ( 복잡한 테스크, 혹은 장시간 걸리는거, 중간중간 결과를 알고싶은 것에서 좋음 )

그러나 90% Topic을 쓰고, 경험상 Action과 Service는 많이 안씀.



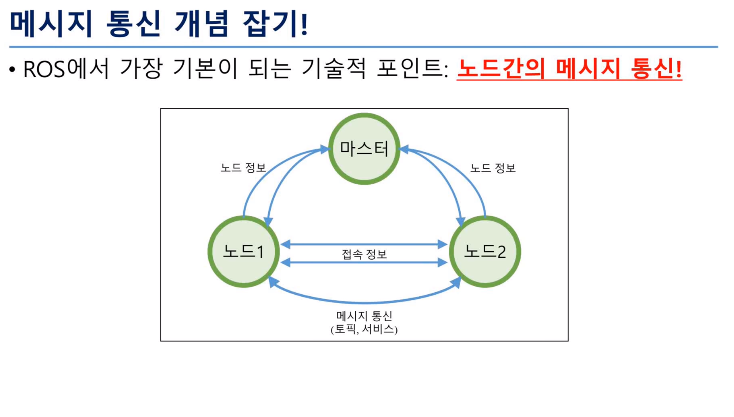
--------------------------------------------------------------------------------------------------------------

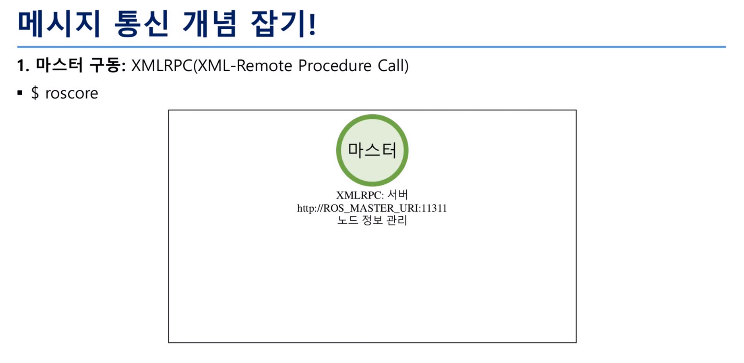
메시지 통신 개념 잡기

ROS의 기본이 되는 기술적 포인트 – 노드간의 메시지 통신

구현되는 방식

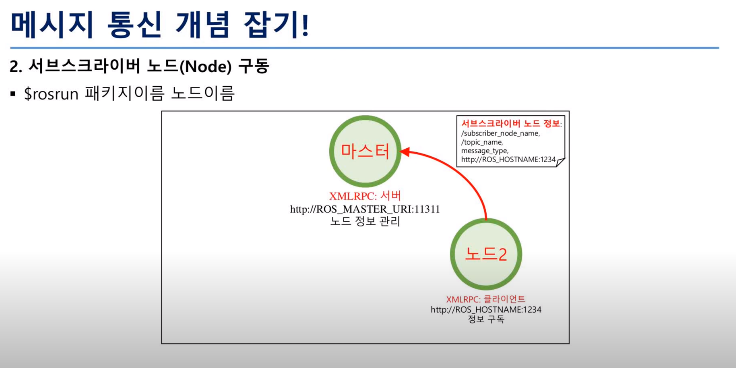
기본적으로 이런 형태의 다이아그램인데





마스터는 노드 정보를 관리 – 실제로 수행하는건 각 실행되는 노드들의 정보를 관리해서 노드들의 통신을 연결시켜주는 매개체의 역할

Roscore 를 통해서 마스터를 구성하고 XMLRPC라는 간단한 서버클라이언트 시스템을 구동시킨다봐라

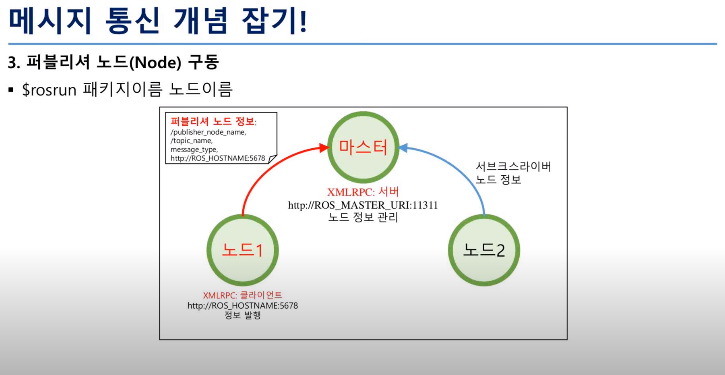


Subscriber 노드를 하나 실행시켰다 치면 ( 토픽을 받는 애 )

실행되자마자 subscriber노드정보를 마스터에게 보내준다.

노드가 실행되면 노드 정보를 마스터가 전송받도록 되어있다.

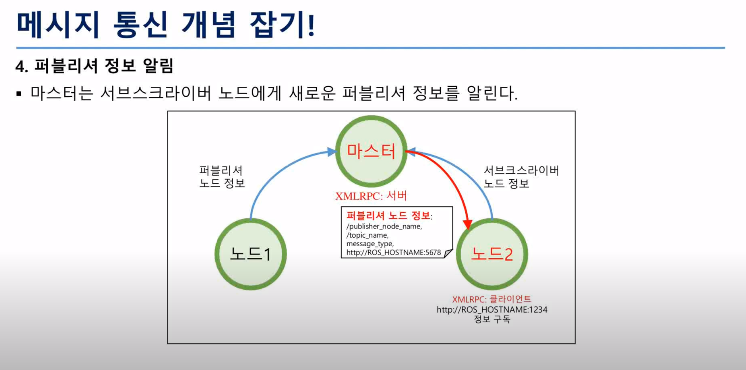
(노드의 이름, 토픽의 이름(어떤 토픽으로 보낼 지 정하게되어있음), 어떤 형태의 메시지를 보낼건지(카메라인지 IMU정보인지 등등), IP, 포트번호<다른 노드랑 접속시키기 위해 네트워크정보 보내는것> )

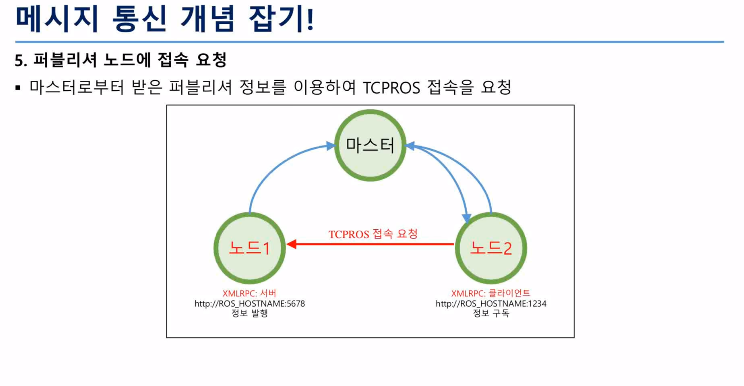


이번에는 퍼블리셔 노드가 나타났다!

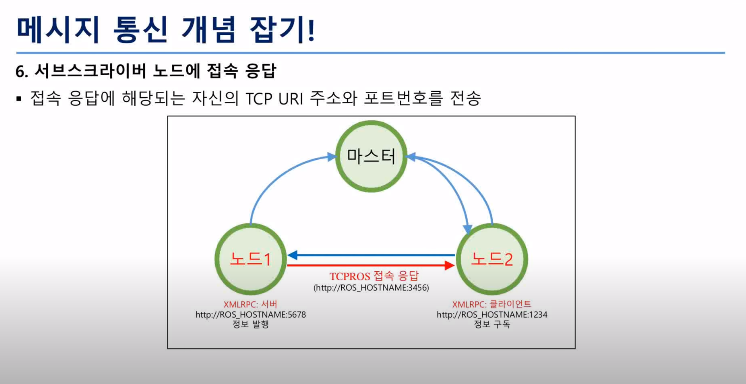
마스터는 중매쟁이 같은 느낌이다. 마찬가지로 노드1은 노드정보 마스터에게 보낸다.

마스터는 매칭해준다. 토픽이름이랑 메시지 형태 따져보고, 맞으면 정보 전달해준다.



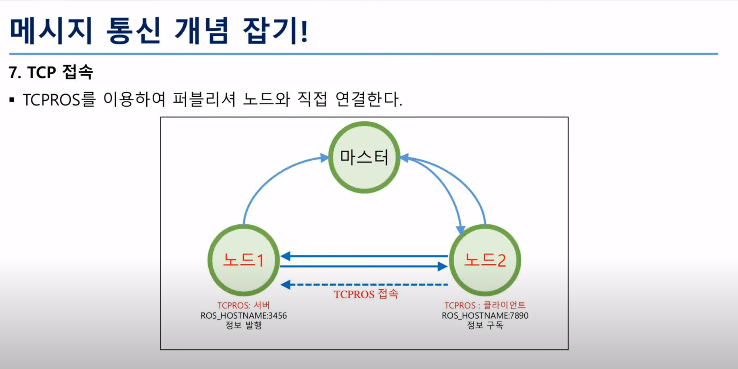


이제 IP번호 포트번호 아니까 직접적으로 TCPROS 접속 요청한다.

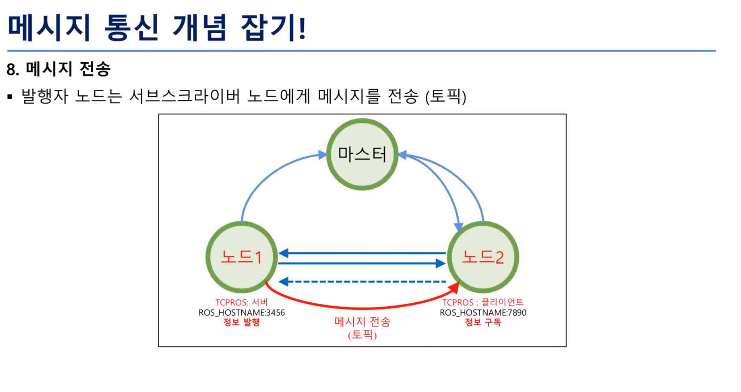


TCPROS 접속 응답을 하고 ( OK )

그 다음



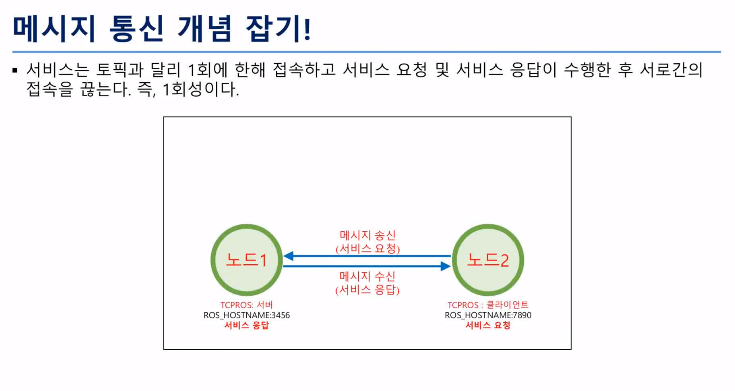
TCPROS라고 해서 TCP 통신이라 생각하면 됨.



이 때부터는 TCP/IP 통신이라 보면 된다.

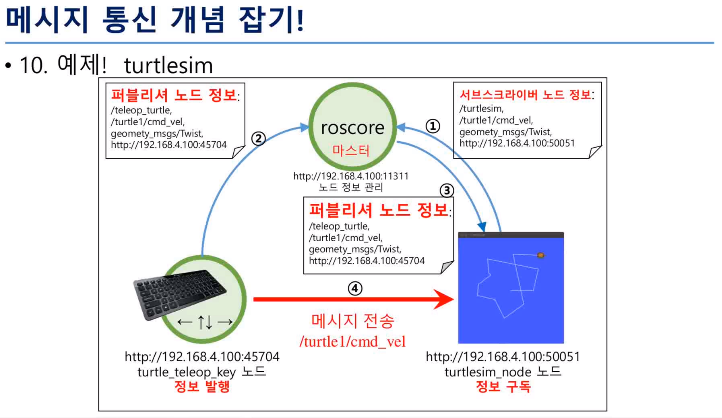
이제부터 마스터 필요없어. 중매쟁이는 빠지면 된다.

토픽얘기로 했지만 서비스도 마찬가지로 마스터가 매칭 시킨 후 똑같이 진행된다.



Rosrun – CommandLine tools중의 하나이다. 노드 하나!!를 실행 시킬 때 사용되는 명령어이다. ROS에서.

Rosrun turtlesim turtlesim\_node 의 뜻. 프로그램 하나 실행시켜라! turtlesim패키지 안에 turtlesim\_node라는 노드를!



Roscore 한번만 시키면 된다. 노드가 수백개여도 roscore는 한번만.

첫번째. Roscore를 무조건 실행시킨다. 로스코어는 복수로 실행시키는게 아니야. 노드가 수백개여도 노드정보를 관리한다.

Subscriber 노드 실행하면 마스터에 노드정보 넘길거야.

/turtlesim 이라는건 노드이름

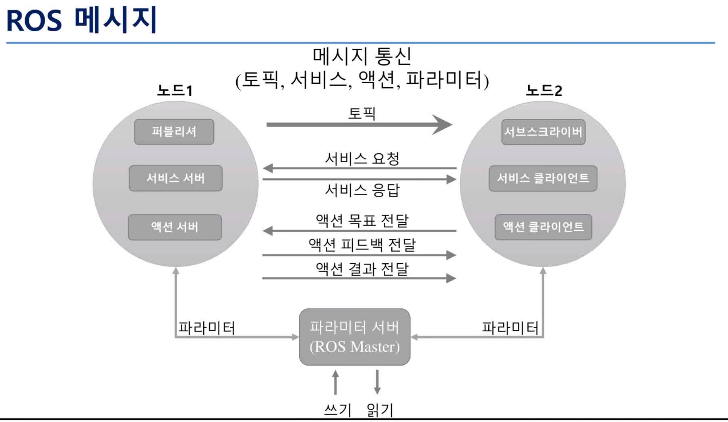
/turtle1/cmd\_vel 은 토픽의 이름.

Geometry\_msgs/Twist 전진속도 회전속도 메시지형태로 보내는걸 말함

퍼블리셔쪽도 노드정보 다 넘기고. 마스터는 서브스크라이버쪽에 그 노드정보 보내준다.

전진속도 회전속도가 메시지로 전송되고, 거기에 응당하는 프로세싱을 한다. ( 로봇이 움직인다 )

ROS 메시지



토픽 서비스 액션 + 비슷한 파라미터라고 있긴 함 ( 비슷하면서도 아닌것같긴한데 )

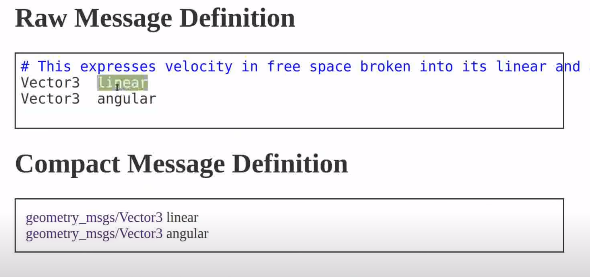
파라미터라는 개념은 어떤 변수를 (프로그램으로 치면 글로벌 변수) 같은걸 네트워크에 지정해놓을 수 있고, 그 변수를 다른 노드에서 실시간으로 (혹은 임의의 시간에) 받아서 자신의 프로세스를 바꿀 수 있다. 예를들어 위의예시에서 얼굴인식 필터중에서 shape 기능을 쓸까 안쓸까를 파라미터로 설정해놓은다음에 외부에서 변수를 바꾸면 프로세스가 컴파일이나 할 필요 없이 자동적으로 바뀌도록 할 때. 파라미터를 외부에서 가져다 씀.



1번링크 – 메시지에 관련된 정보.

2번링크 – 일반적으로 많이 사용되는 메시지를 모아놓은 것. ( 액션메세지, 지오메트리메세지, 센서메세지 등등. )

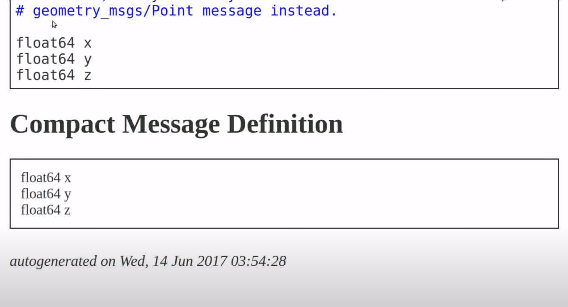
예를 들어 twist메시지의 경우



아까 거북이 트위스트 서브스크라이버노드 얘기할 때 twist뭐시기 있었잖아?

Linear, angular값이 전진속도(병진속도?)랑 회전속도 의미하는거고

Vector3도 들어가보면 아래그림처럼 나와



결국 twist메시지를 쓴다는 것은 무엇이냐? 총 6개의 속도값을 날릴거다. 3개는 linear, 3개는 angular값이다!.

단순자료형

Float65, int32 이런거. 예) 정수, 부동소수, Boolean 등

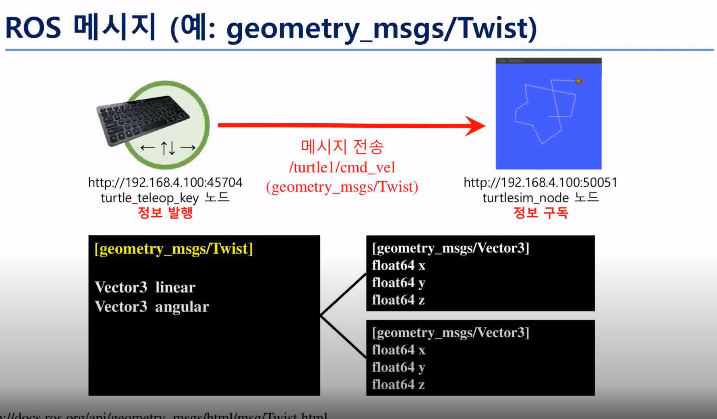
메시지 안에 메시지를 품고 있는 간단한 구조

예를들어 twist에는 vector3라는 메시지가 또 사용되어있다.

메시지들이 나열된 배열과 같은 구조

많이쓰인다. 레이저스캔같은거 하게되면 2차원 거리값 보내게되서 데이터가 배열로 날아간다.

카메라 정보도 픽셀별 RGB값 있고, 배열로 나간다.



예를들어 twist를 보면 Linear anglar가 xyz xyz로 날아간다.

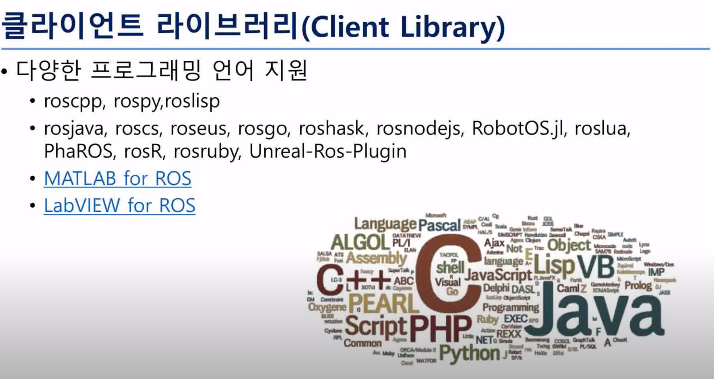
Name, TF, Client Library 이기종 디바이스 간의 통신  
Name? – 노드의 이름, 메시지(토픽 같은거) 이름같은게 있을거아니야?

노드이름을 변경시키거나 메시지 이름을 변경시키거나 할 때

내용이 많아서 다음 강의시간에 자세히..

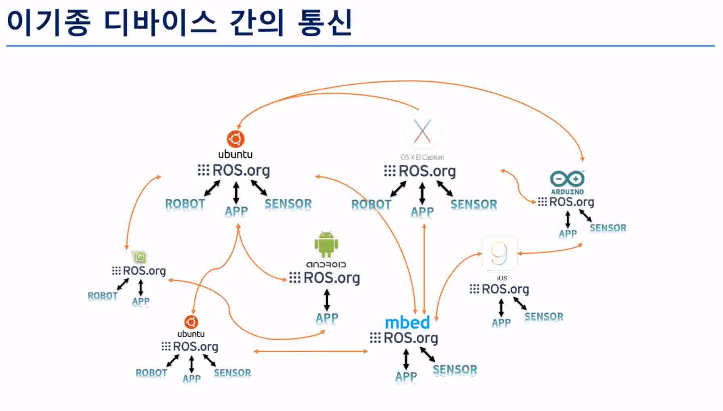
좌표변환은 나중 강의에서. – TF(좌표변환)

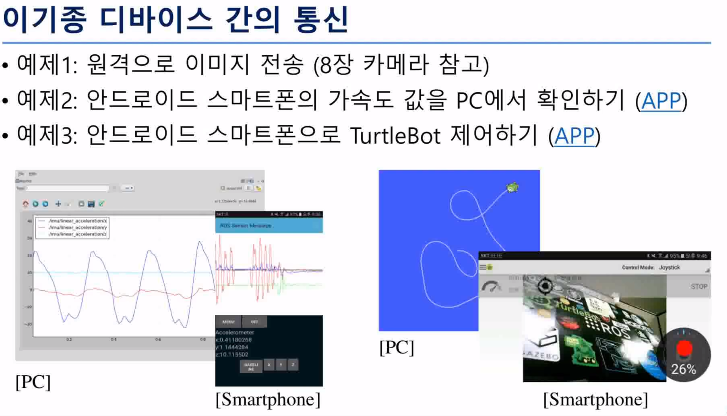
각 조인트들의 상대좌표 변환



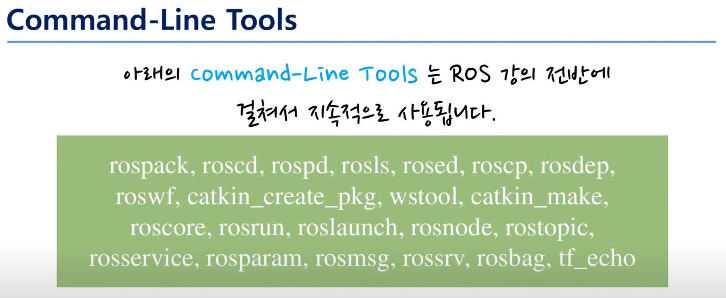
우리는 roscpp 사용하면 되겠다. C++

매트랩에서도 할 수 있도록

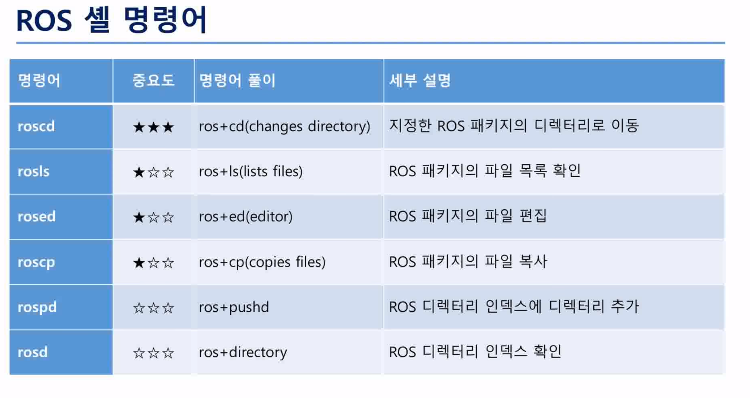




1. 로스 명령어

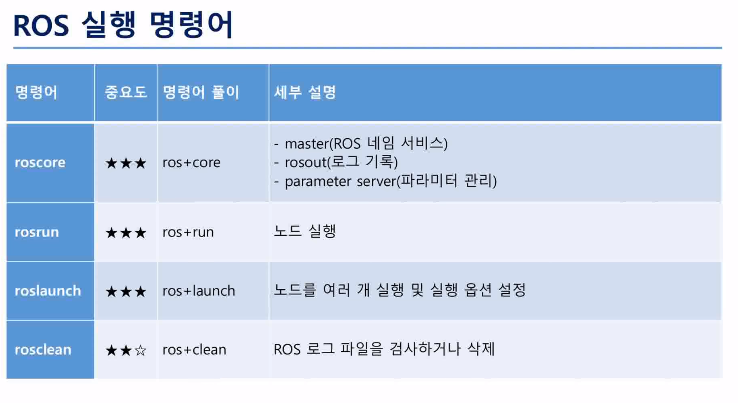


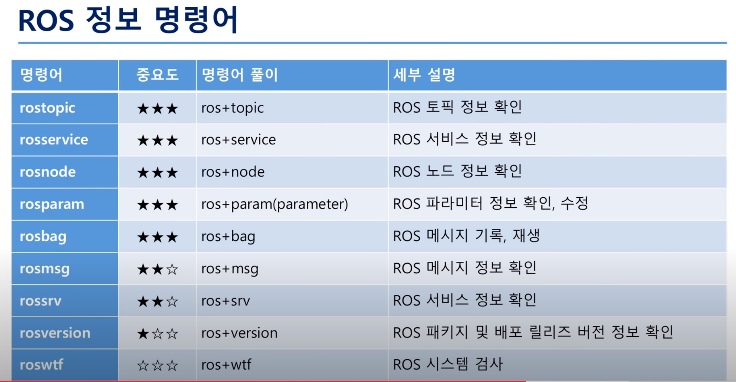
1:50 링크 하나



Roscd – cd는 원래 리눅스에서 쓰던 명령어인데

지정한 로스패키지 디렉토리로 이동하는 것







catkin명령어? – catkin은 ros의 컴파일(빌드)관련 시스템