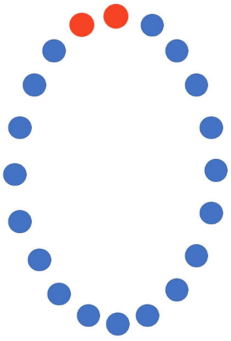
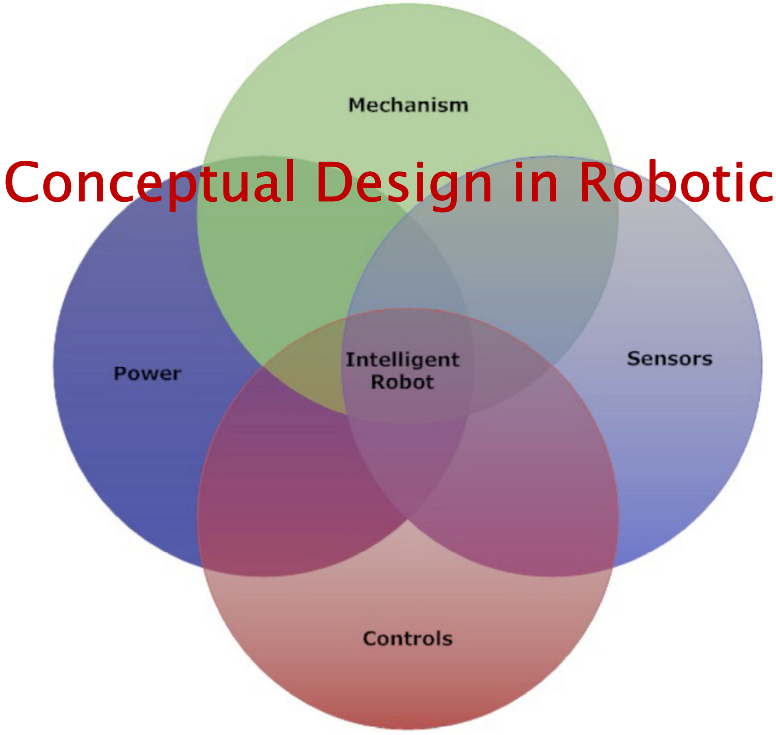
ZERO

Robotics Development Framework, Workspace & Workflow





Conceptual Design in Robotics

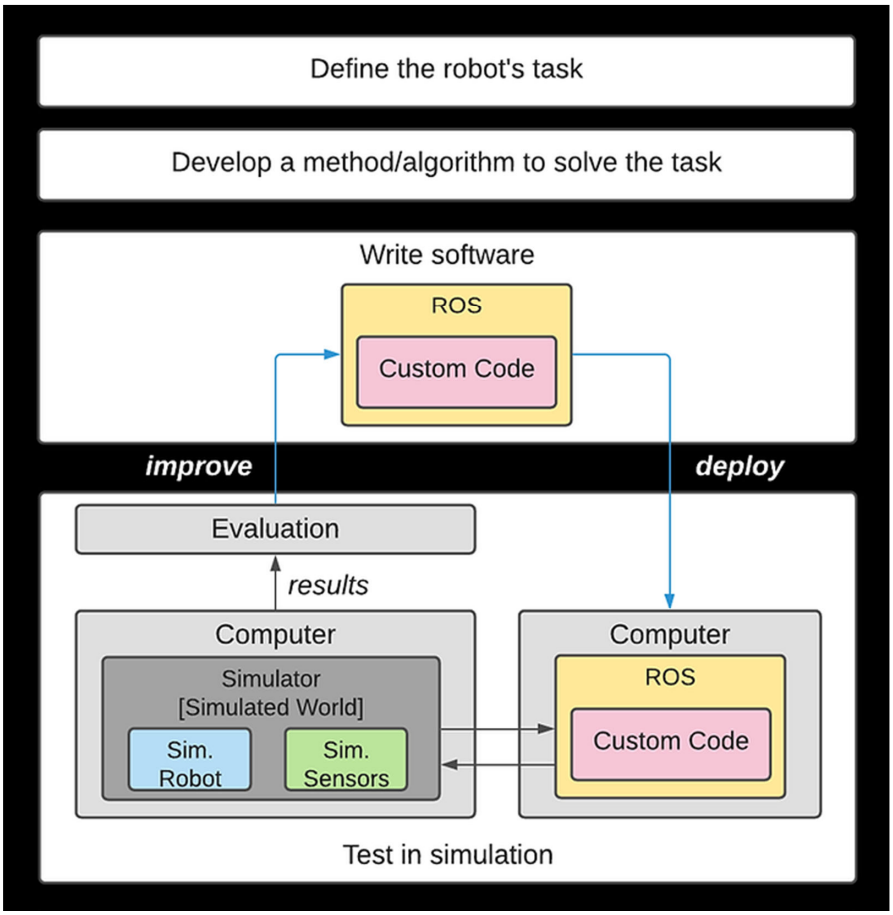
The requirements for a robot that collects table tennis balls:

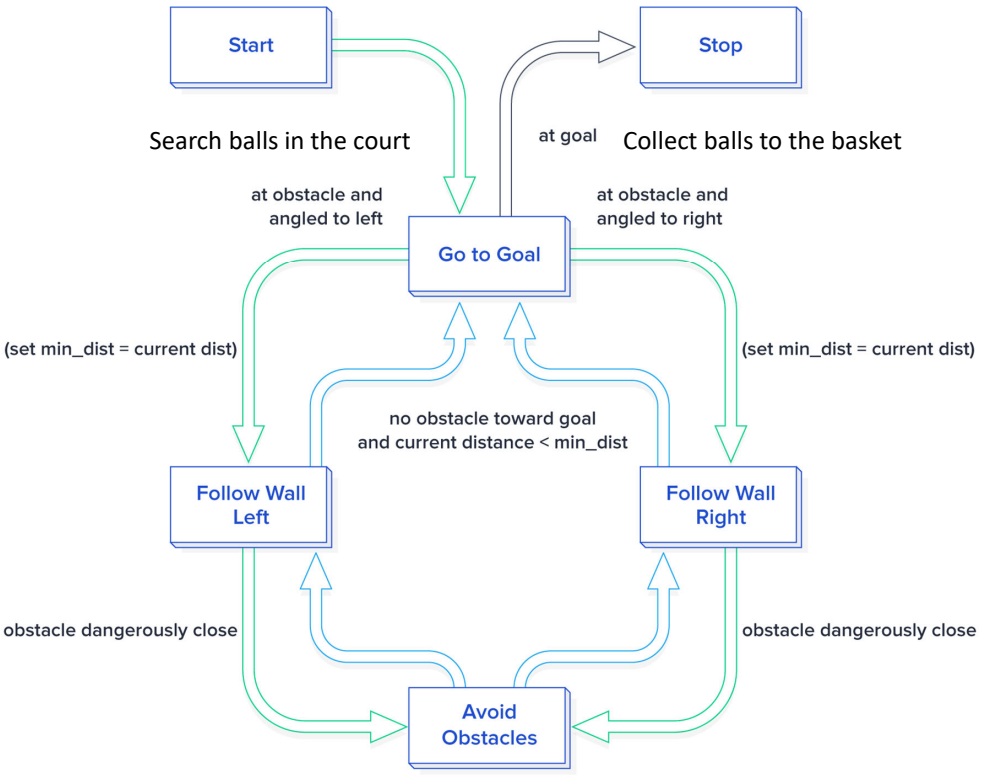
(a) It should be applicable to the setting of general table tennis courts, with a size of 8 m × 16 m;

(b) It can collect at least 30 table tennis balls within a reasonable time as well as show the number of table tennis balls collected;

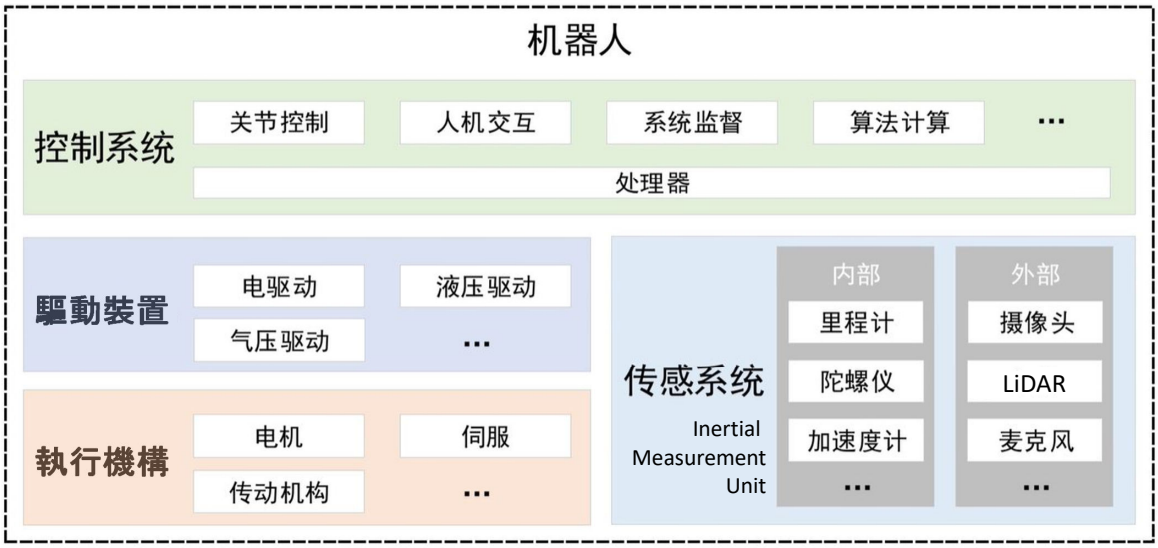
(c) When no one is inside the table tennis court, it can be activated to start the process for collecting table tennis balls automatically within the designated area;

(d) There should a basket to hold the collected table tennis balls and allow players to take out the balls easily to continue practising.

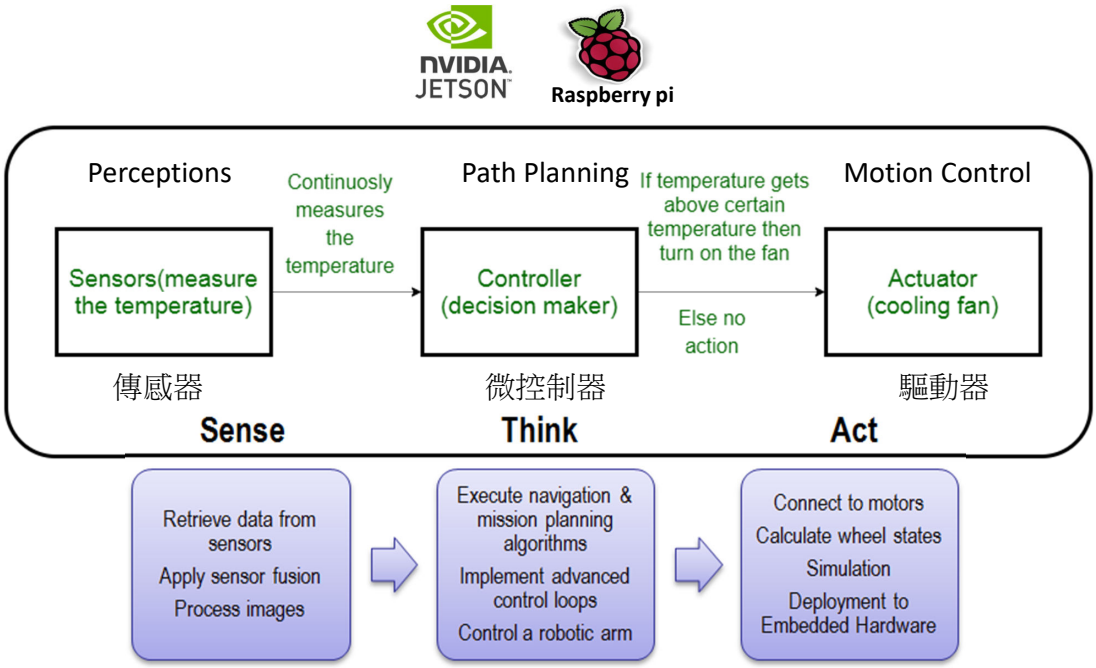




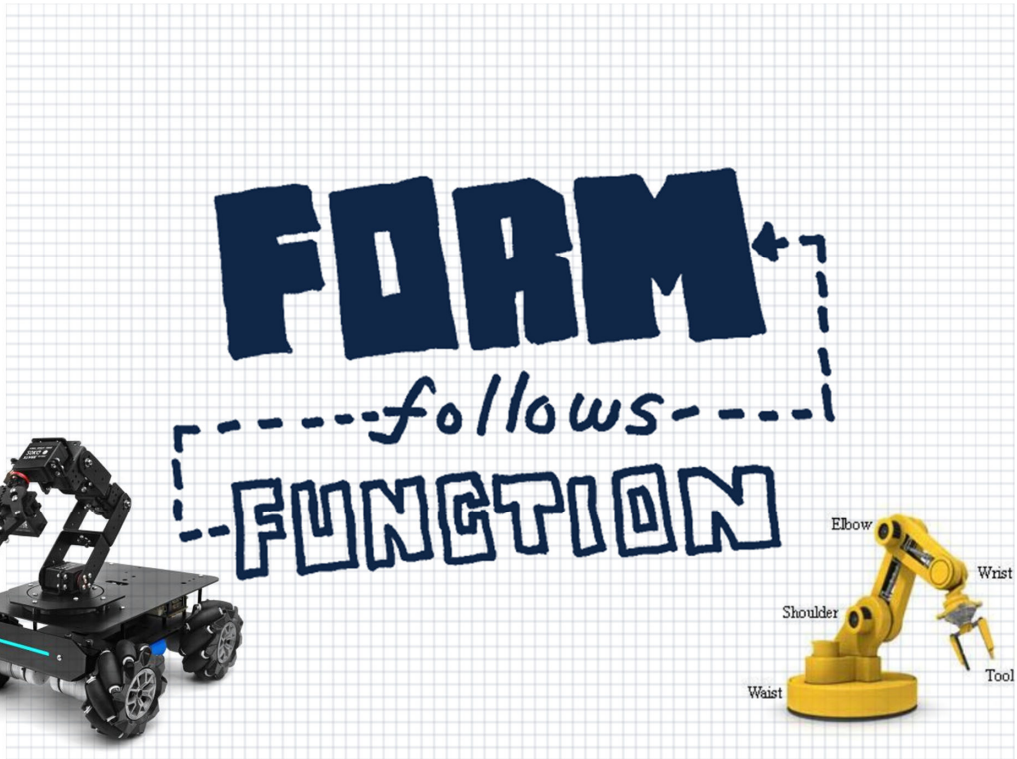
Define robot's tasks



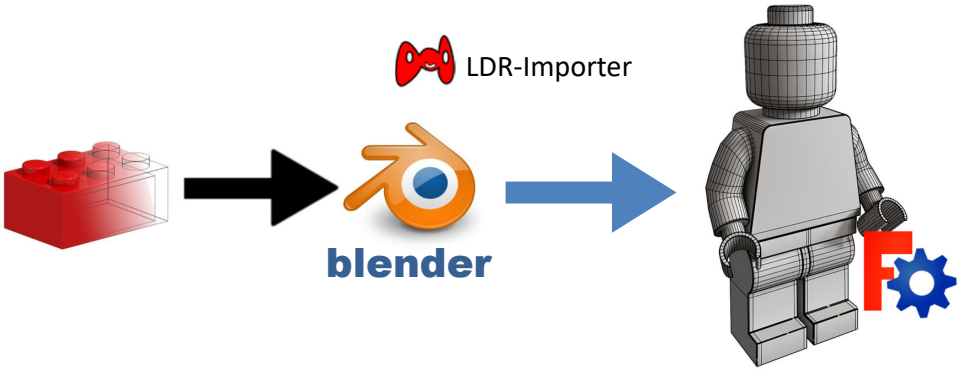
Define robot's functions & behaviour



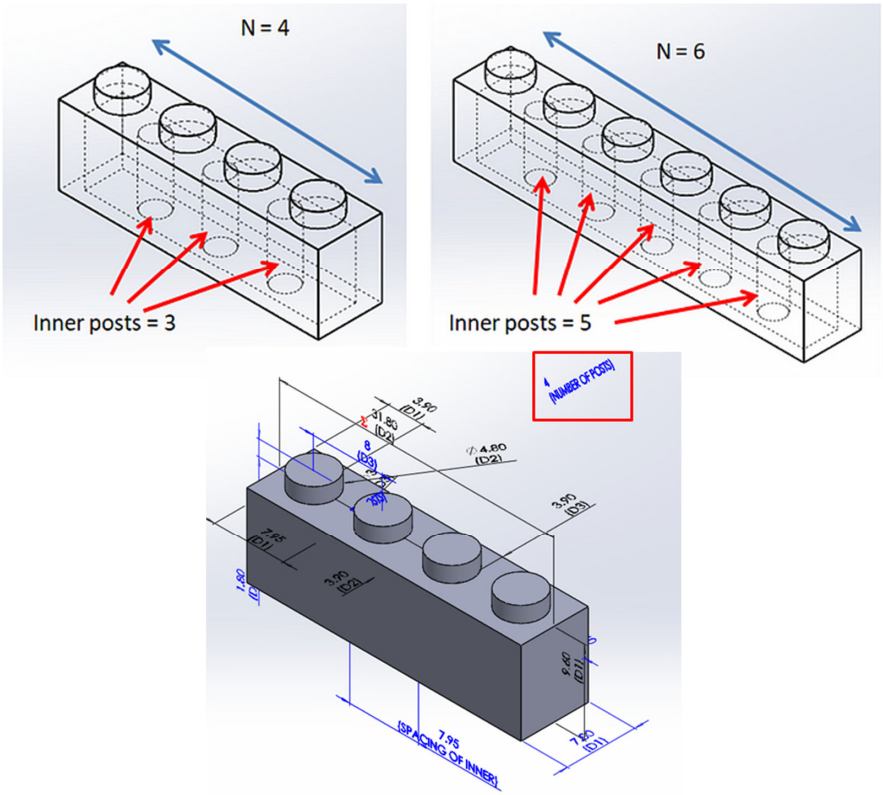
Robotics in physical computing



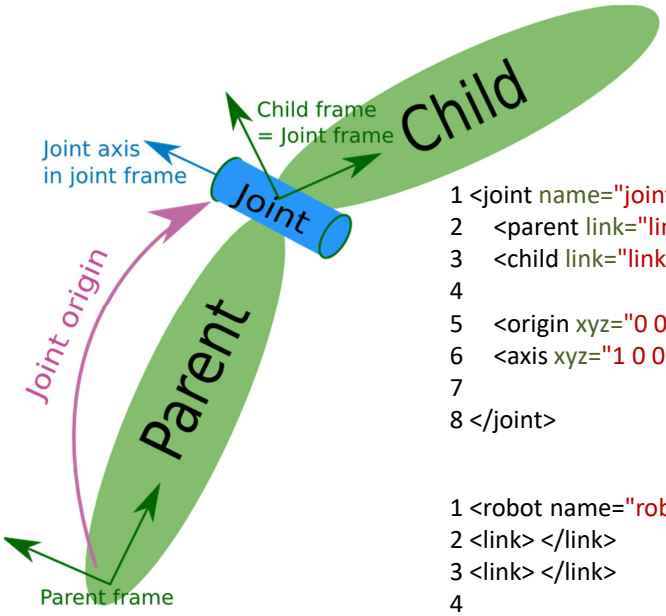
LDraw.org
Centralised LDraw Resources



Parametric modeling



Parametric modeling




```
1 <joint name="joint_name" type="continuous">
2    <parent link="link1"/>
3    <child link="link2"/>
4
5    <origin xyz="0 0 0" rpy="0 0 0"/>
6    <axis xyz="1 0 0"/>
7
8 </joint>
```

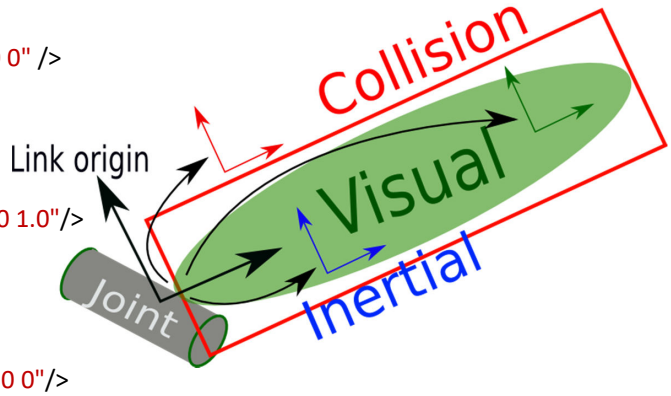
```
1 <robot name="robot_name">
2 <link> </link>
3 <link> </link>
4
5 <joint> </joint>
6 <joint> </joint>
7 </robot>
```

URDF modeling

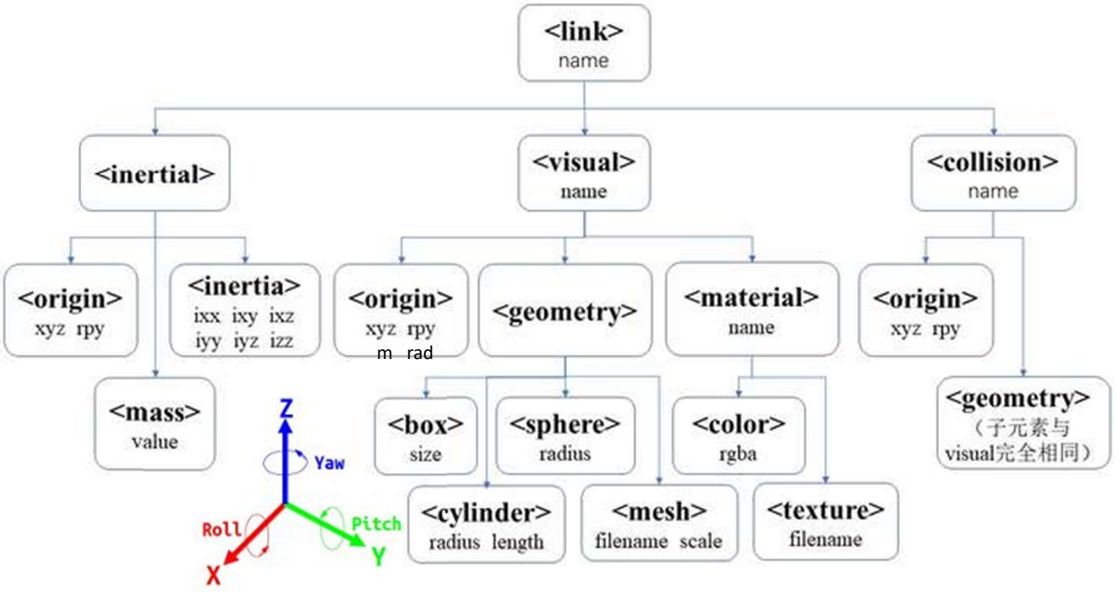
```
1 <link name="link_name">
2  <inertial>
3     <origin xyz="0 0 0.5" rpy="0 0 0“ />
4     <mass value="1“ />
5     <inertia ixx="100" ixy="0" ixz="0"
            iyy="100" iyz="0"
            izz="100" />
6  </inertial>
7
8  <visual>
9     <origin xyz="0 0 0" rpy="0 0 0" />
10    <geometry>
11          <box size="1 1 1" />
12    </geometry>
13    <material name="Cyan">
14          <color rgba="0 1.0 1.0 1.0"/>
15    </material>
16  </visual>
17
18  <collision>
19     <origin xyz="0 0 0" rpy="0 0 0"/>
20     <geometry>
21          <cylinder radius="1" length="0.5"/>
22     </geometry>
23  </collision>
24 </link>
```

$$\begin{pmatrix} \mathbf{ixx} & \mathbf{ixy} & \mathbf{ixz} \\ ixy & \mathbf{iyy} & \mathbf{iyz} \\ ixz & izy & \mathbf{izz} \end{pmatrix}$$

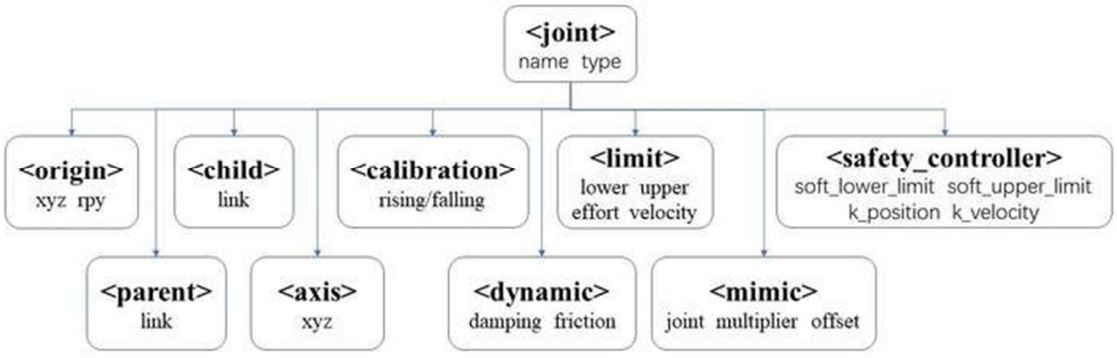
Inertia Tensor



URDF modeling



URDF modeling



URDF modeling

https://articulatedrobotics.xyz/category/concept-design

- Most robotics work are in teams include one who develops a CAD model of robot.
- Instead of crafting an URDF by hand it is possible to export an URDF model.
- Below is a list of available URDF exporters for a variety of CAD and 3D modeling software systems.

CAD Exporters

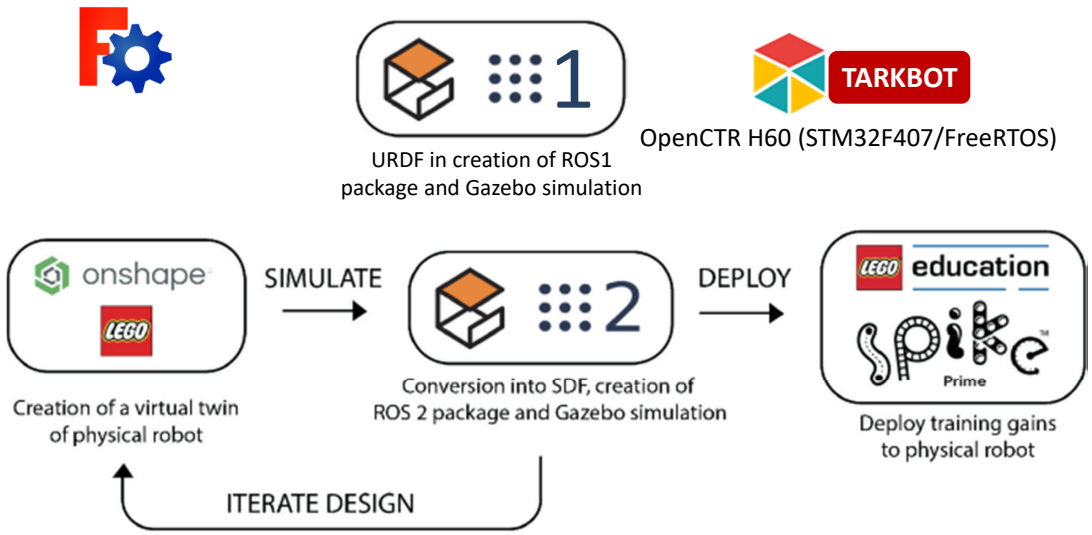
- Fusion 360 URDF Exporter
- SolidWorkds URDF Exporter
- ROS Workbench for FreeCAD
- OnShape URDF Exporter
- CREO Parametric URDF Exporter

Other URDF Export Tools

- Copellia Sim URDF Exporter
- Isaac Sim URDF Exporter
- Blender URDF Exporter
- Gazebo SDFormat to URDF Parser
- SDF to URDF Converter in Python
- The Blender Robotics Tools includes a tool to export URDF files from Blender

URDF from CAD models

https://download2.portableapps.com/portableapps/FreeCADPortable/FreeCADPortable_0.21.2.paf.exe



URDF from CAD models

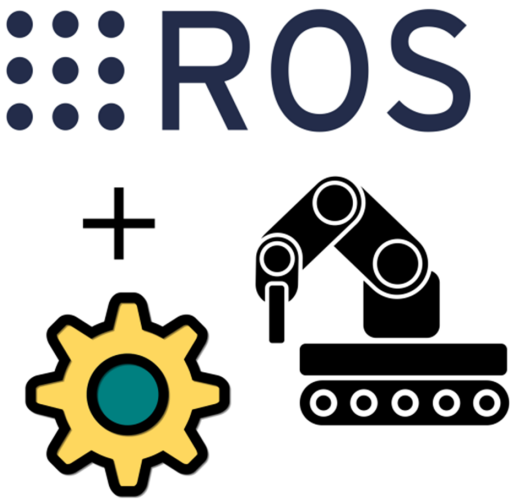
Mode	Pan	Rotate	Zoom	Select
CAD **(default)**	or CTRL	or SHIFT	or CTRL + SHIFT	
Touchpad	Hold Shift + drag	Alt +	PgUp / PgDn	
Blender	Hold Shift + or			

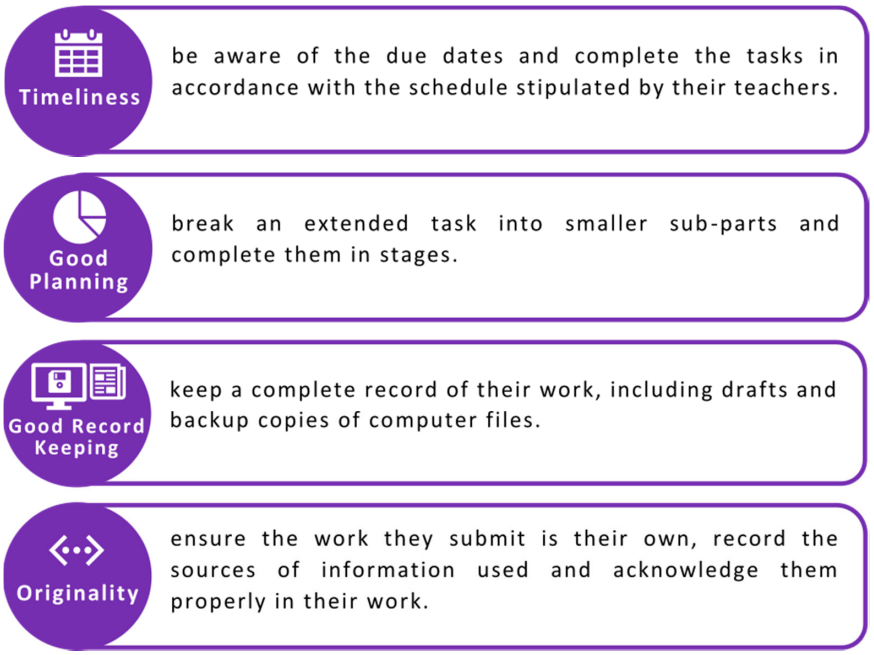
FreeCAD mouse navigation

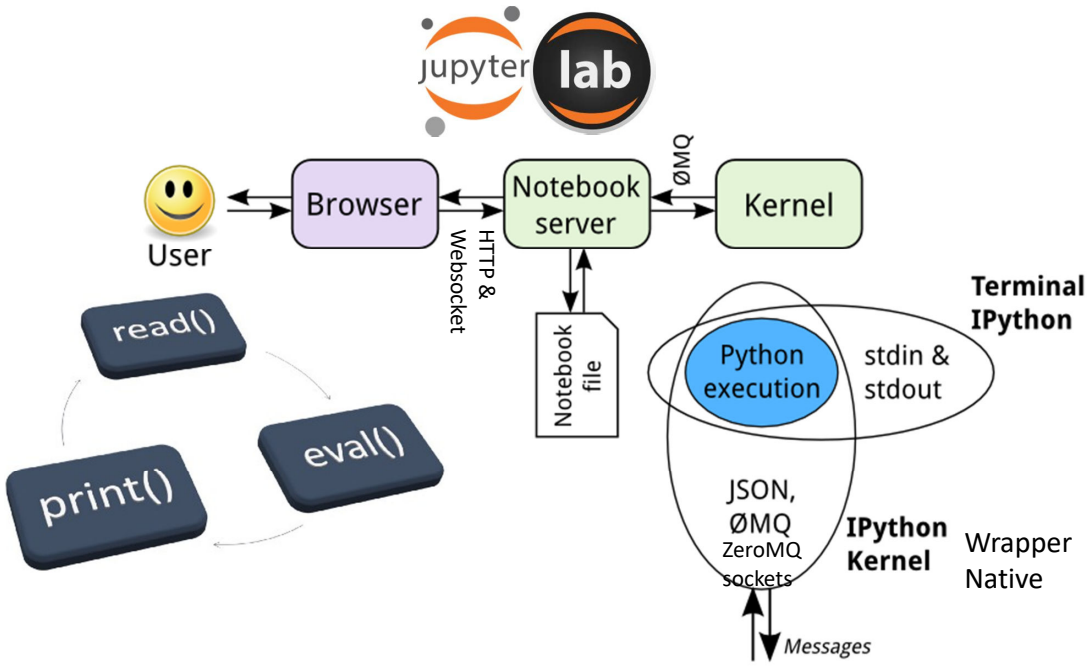
- Ctrl + + and Ctrl + − to zoom in and out, respectively.
- The arrow keys, ◄ ► ▲ ▼, to shift the view left/right and up/down
- Shift + ◄ and Shift + ► to rotate the view by 90 degrees
- The numeric keys, 0 1 2 3 4 5 6, for the seven standard views: Isometric, Front, Top, Right, Rear, Bottom, and Left
- V O will set the camera in Orthographic view.
- While V P sets it in Perspective view.
- Ctrl will allow you to select more than one object or element

FreeCAD keyboard navigation

Design Workspace in Robotics

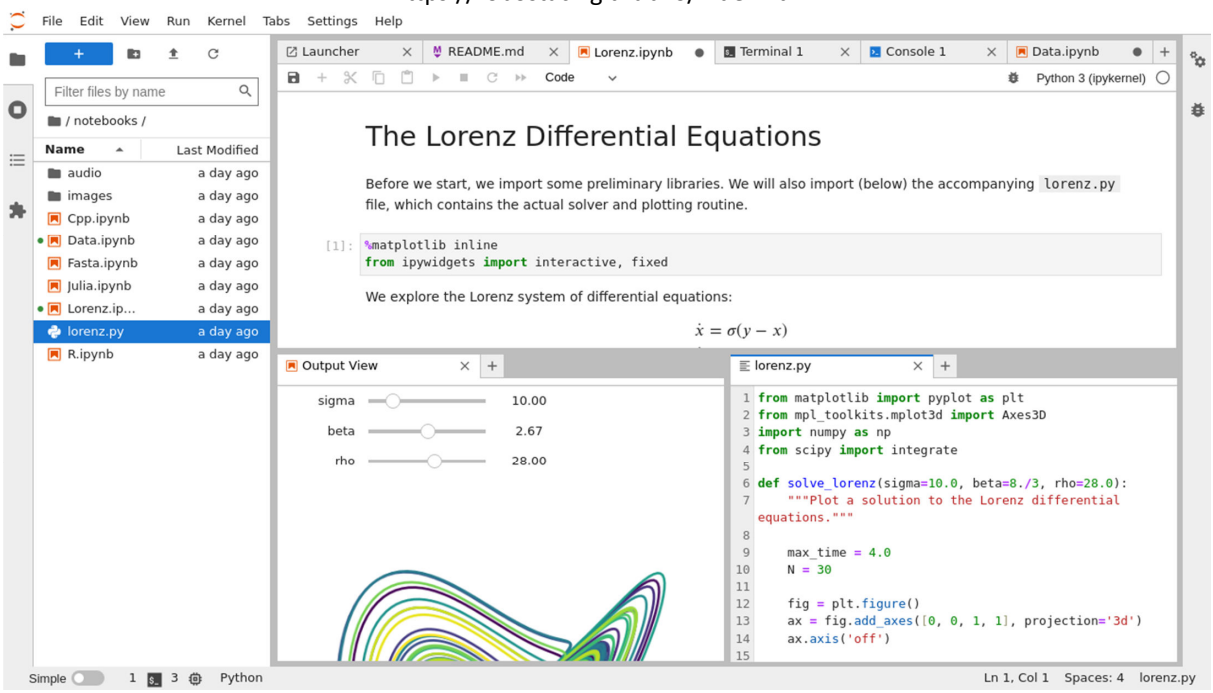




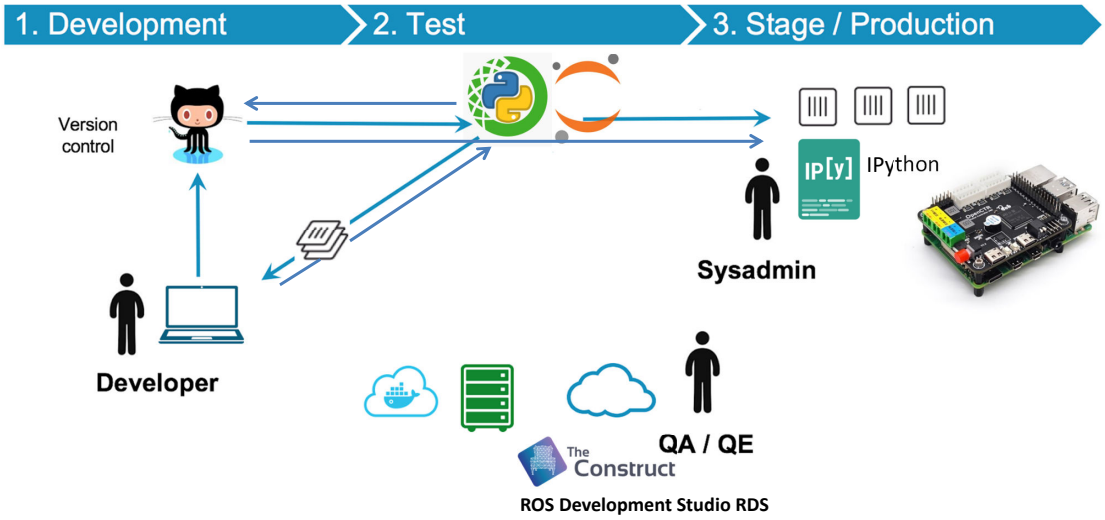


Proposed Python workspace

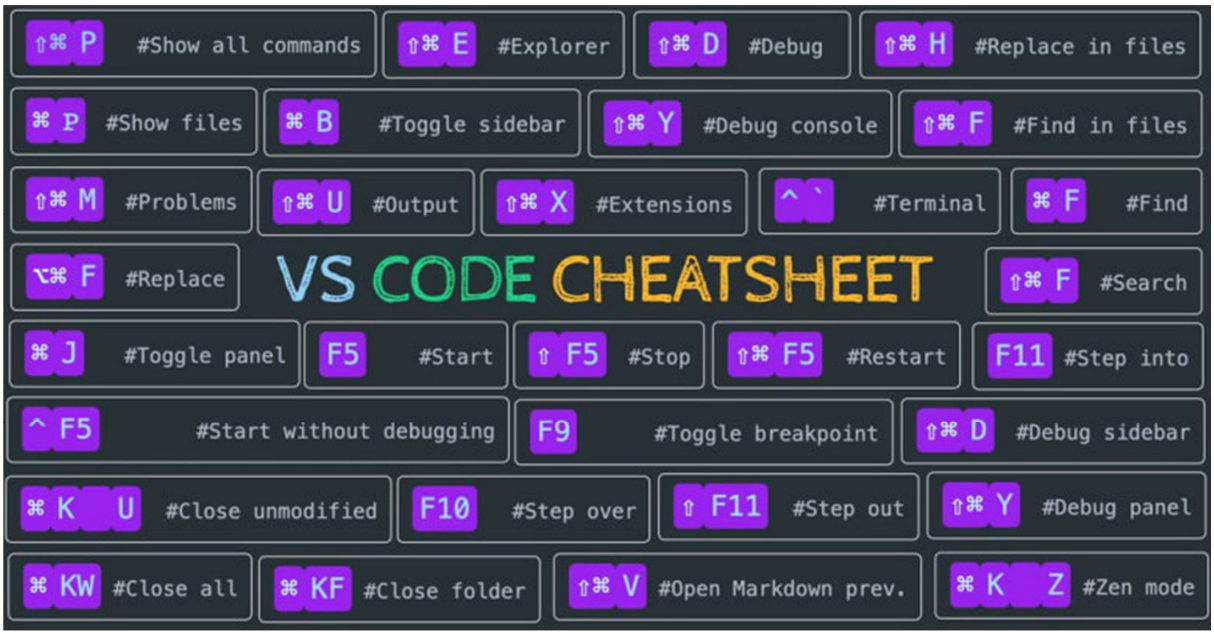
https://robostack.github.io/index.html

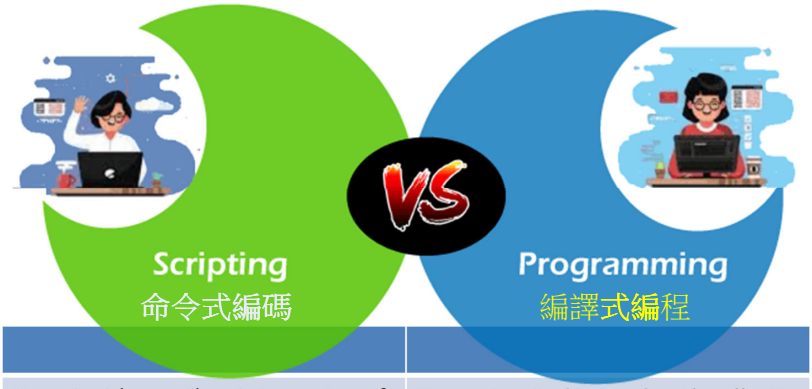


Proposed Python workspace



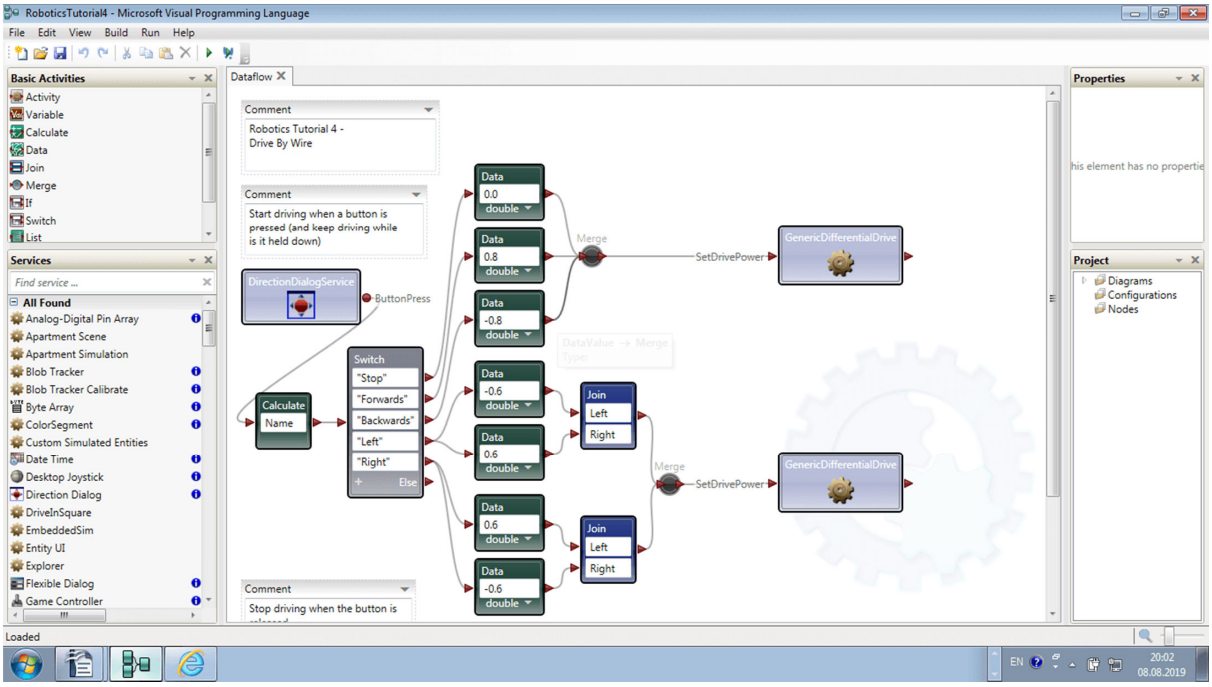
Proposed CI/CD workflow





Runs inside another interpreter of a particular runtime environment	Executes independently albeit required runtime libraries
Do not create a ***.exe*** file	These generate ***.exe*** files
Glue languages for existing programs and libraries	High-speed languages to work with API
Useful in rapid prototyping	
Command statements (declarative language)	

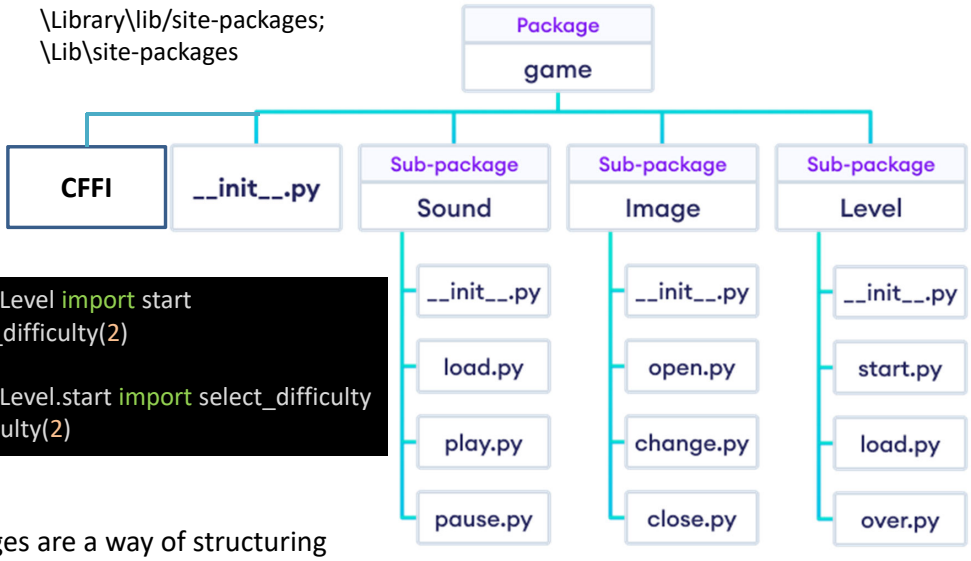
Python scripting



Visual scripting

\Library\lib/site-packages;
\Lib\site-packages

C Foreign Function Interface Module



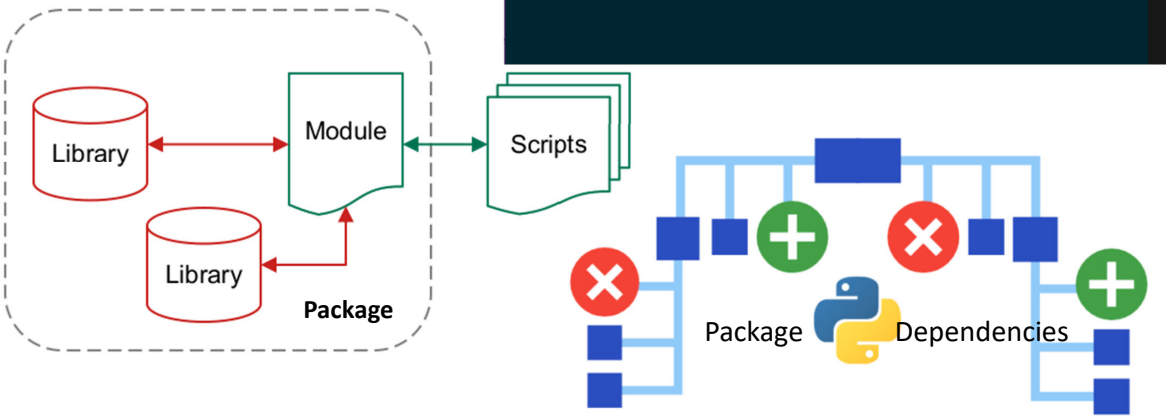

```
from Game.Level import start
start.select_difficulty(2)

from Game.Level.start import select_difficulty
select_difficulty(2)
```

Packages are a way of structuring Python's module namespace by using "dotted module names".

Python packages

A module is a file containing Python definitions & statements and use them in a script or in an interactive instance of the interpreter. The file name is the module name with the suffix .py appended.

```
$ cat hello_world.py
#!/bin/env python

print("Hello world!")
$ chmod +x  hello_world.py
$ ./hello_world.py
Hello world!
$
$ python hello_world.py
Hello world!
$
```

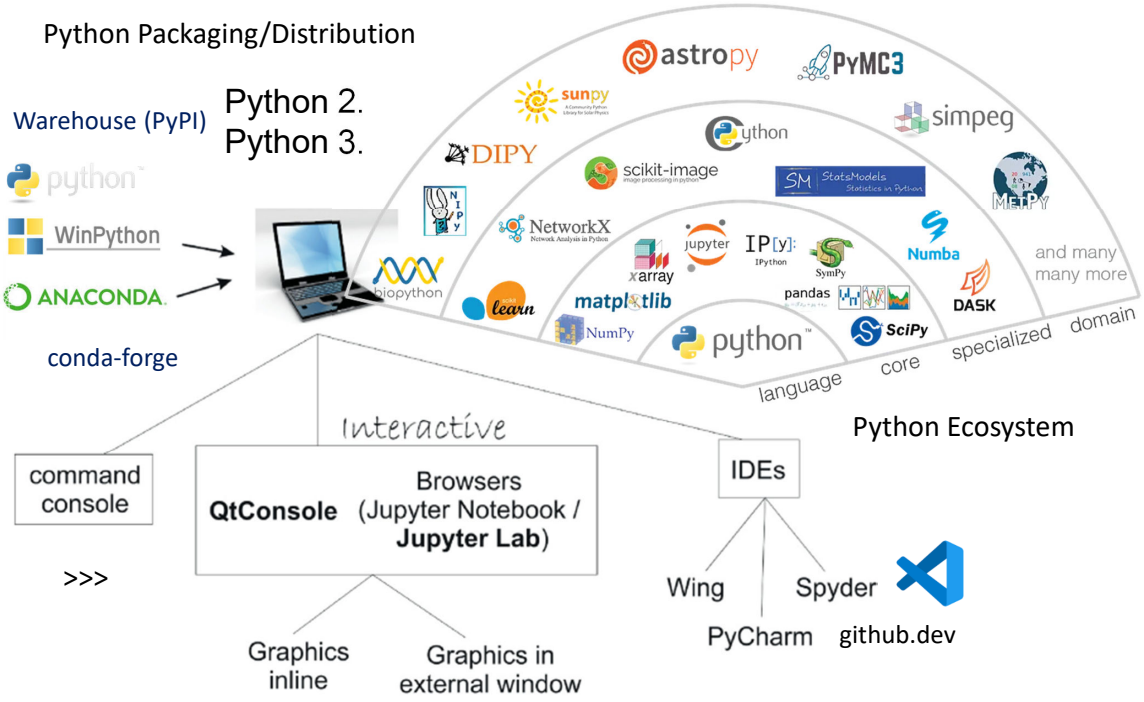


Python packages

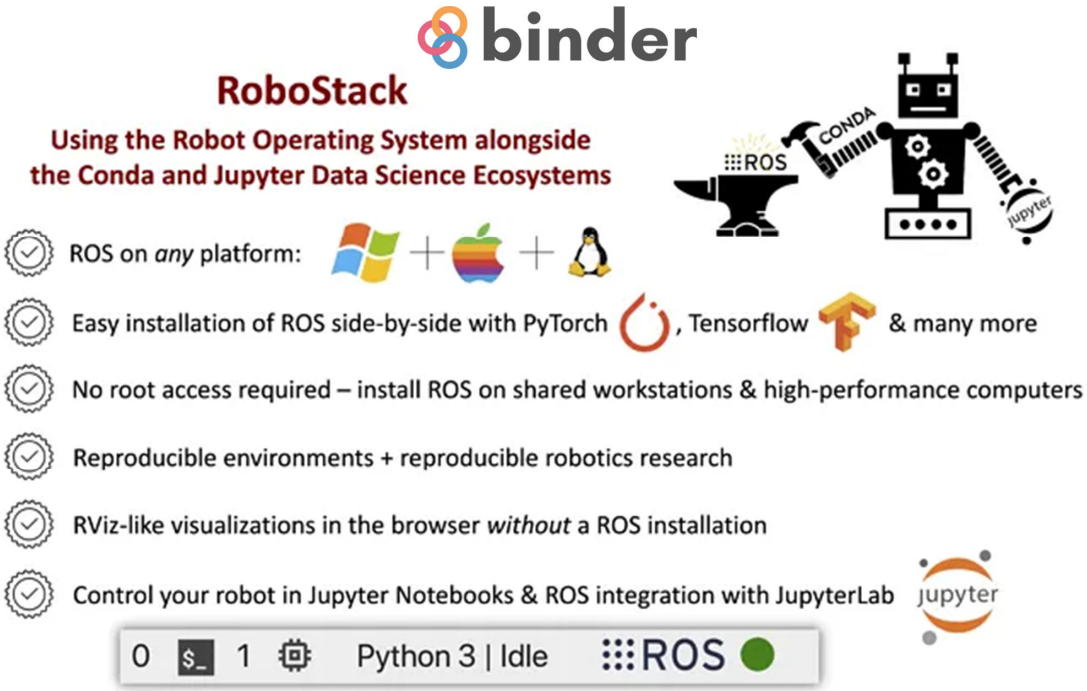
conda env export > environment.yml

conda env create -f environment.yml

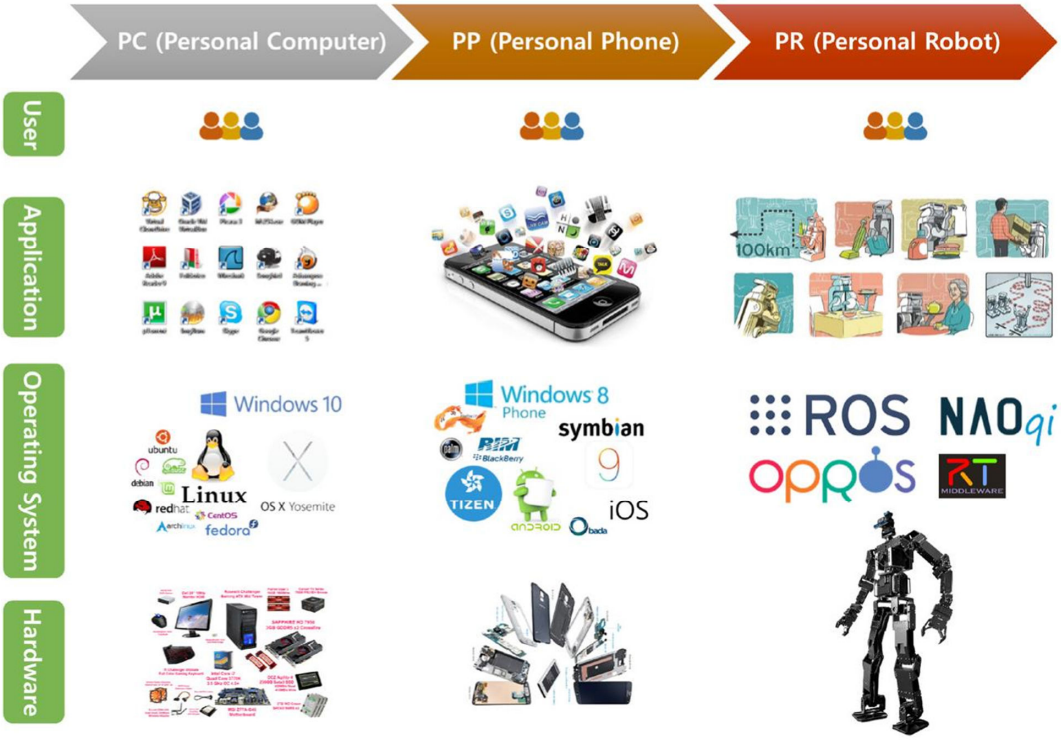
Python virtual environment



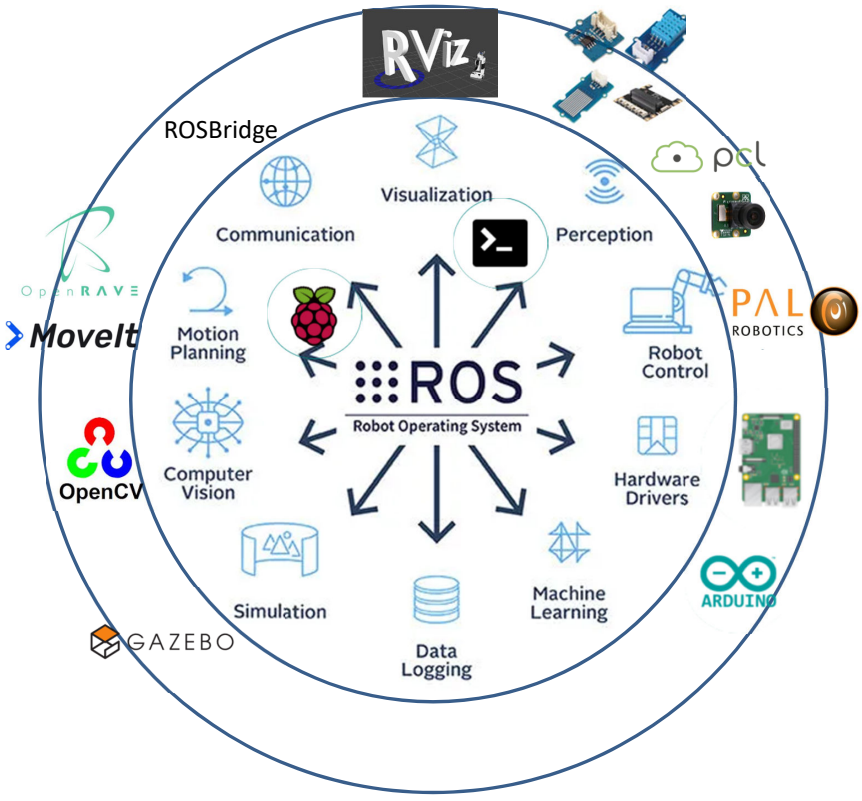
Python robotic development

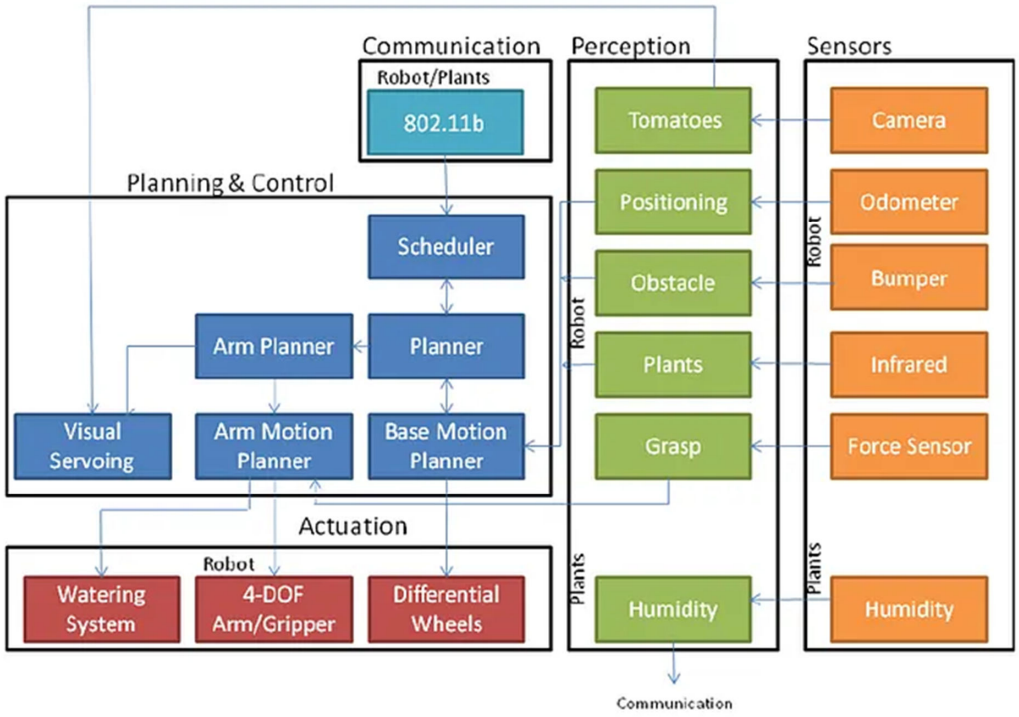


Python robotic development

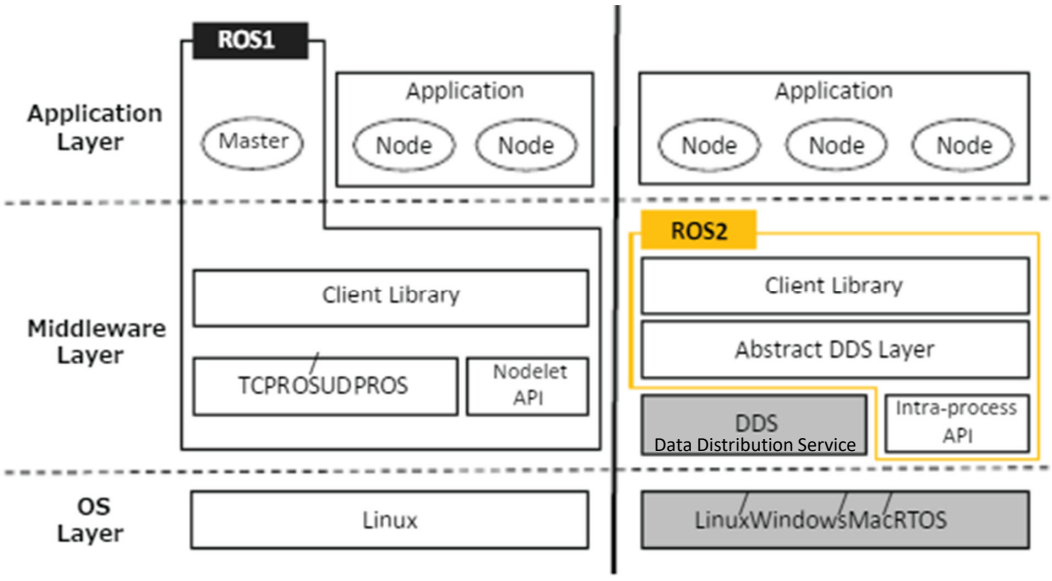


Operating System

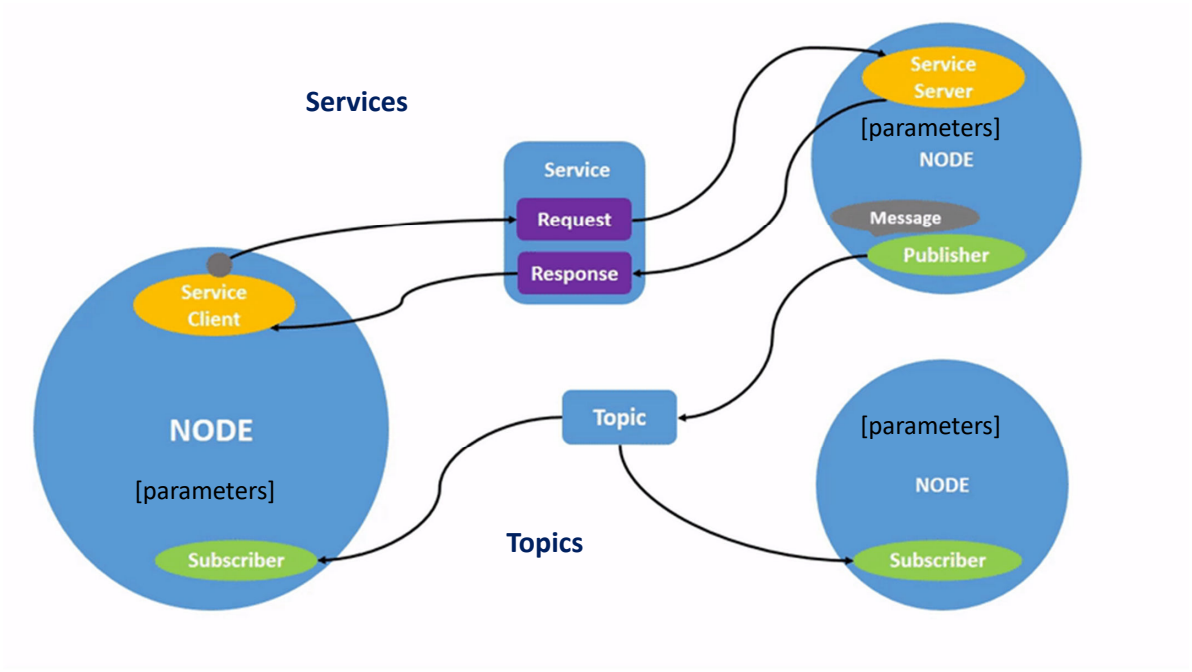




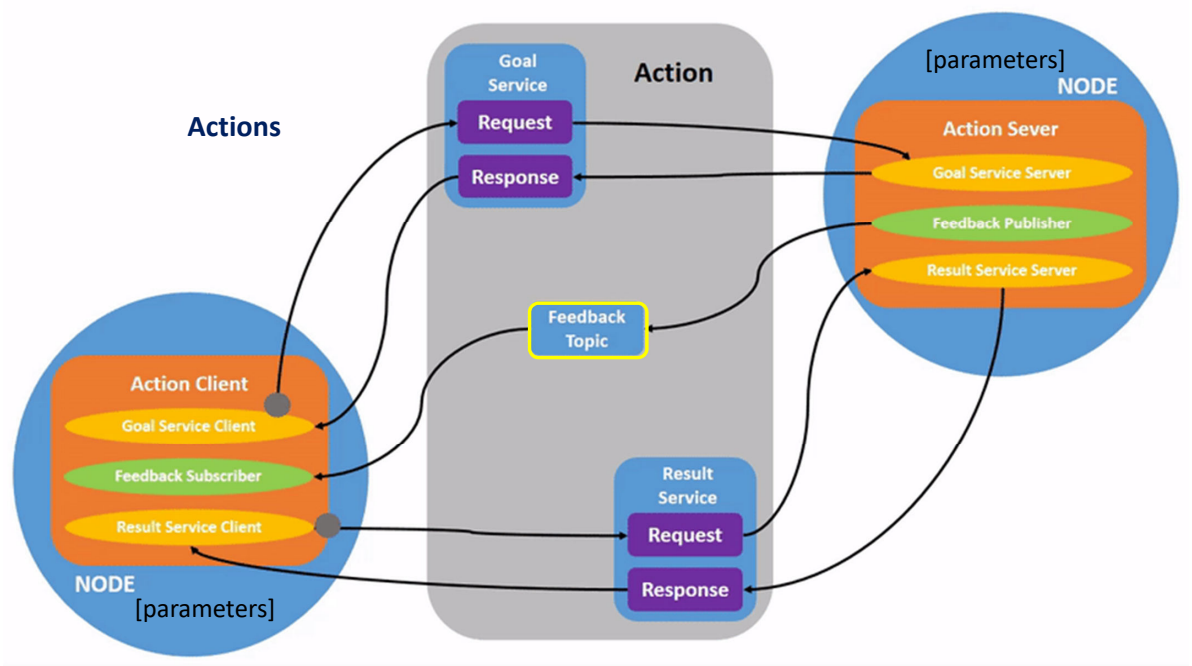
Robot Operating System



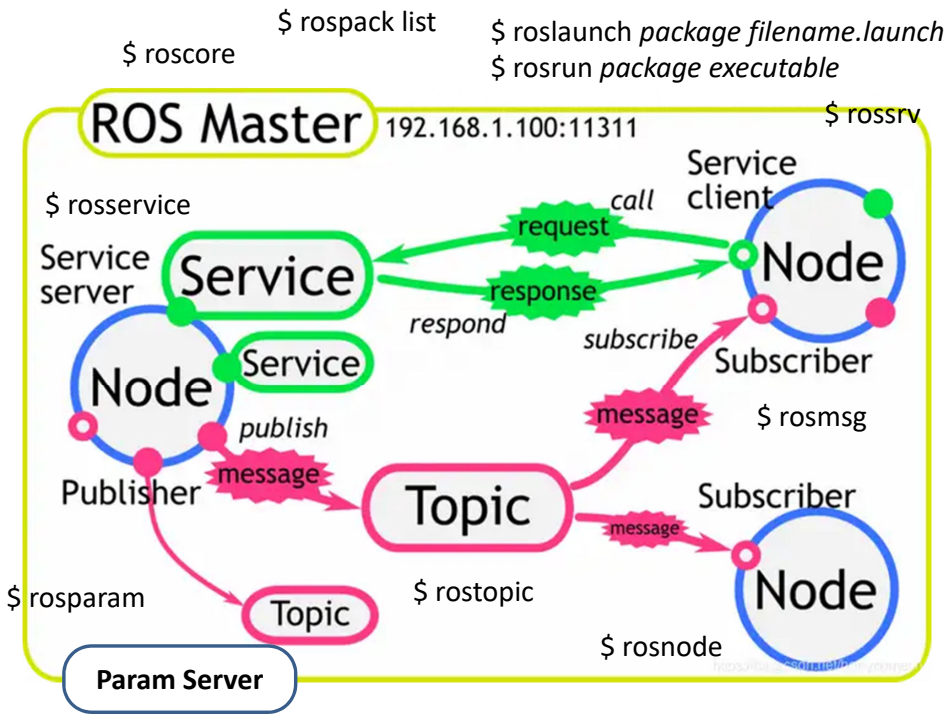
Robot Operating System



Robot Operating System

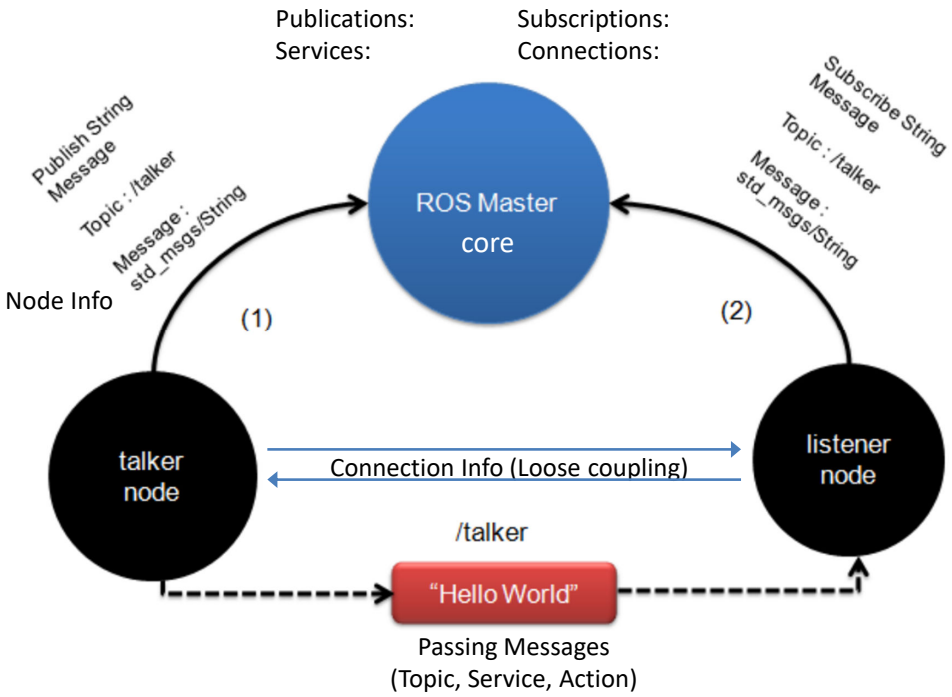


Robot Operating System

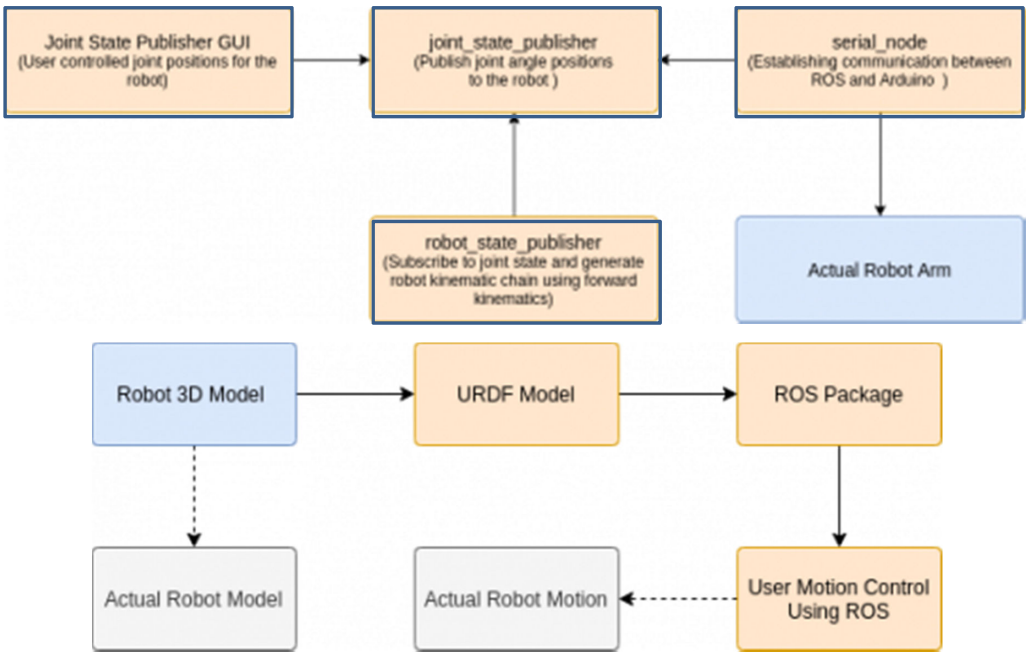


Robot Operating System

```
### activate ros_noetic env
$ conda activate K:\Miniconda3\envs\ros_noetic

### launch ros1 cli
$ roscore
$ rostopic list

### launch rviz GUI
$ rosrun rviz rviz

### launch gazebo GUI
$ gazebo --verbose

### launch a demo node talker
$ roscd rospy_tutorials
$ 001_talker_listener\talker

### from another terminal activate the ros_noetic env and launch
a demo listener
$ roscd rospy_tutorials
$ 001_talker_listener\listener
```

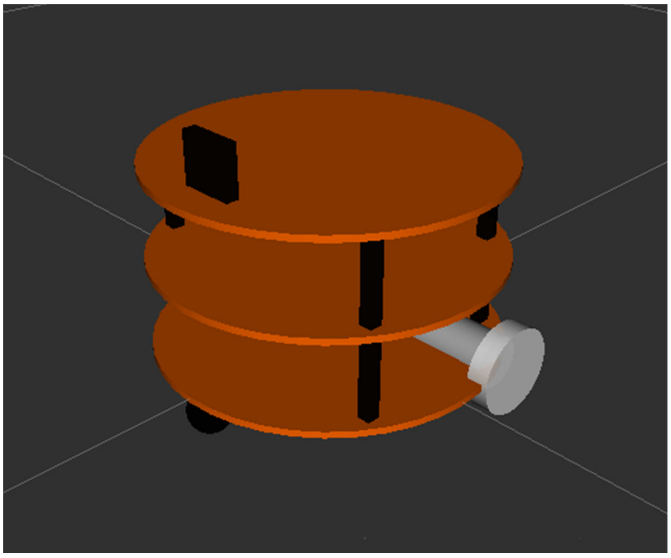


ROS nodes and messages



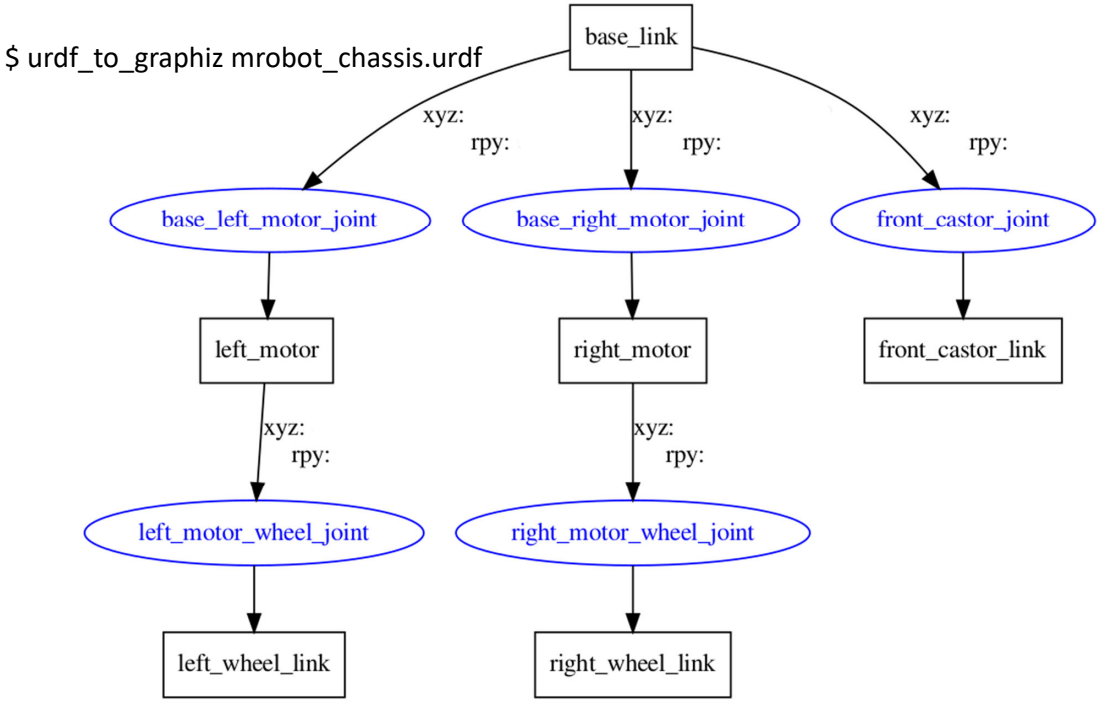
ROS URDF models

ROS机器人开发实践 -- 胡春旭
https://github.com/1027243334/robot_mrobot



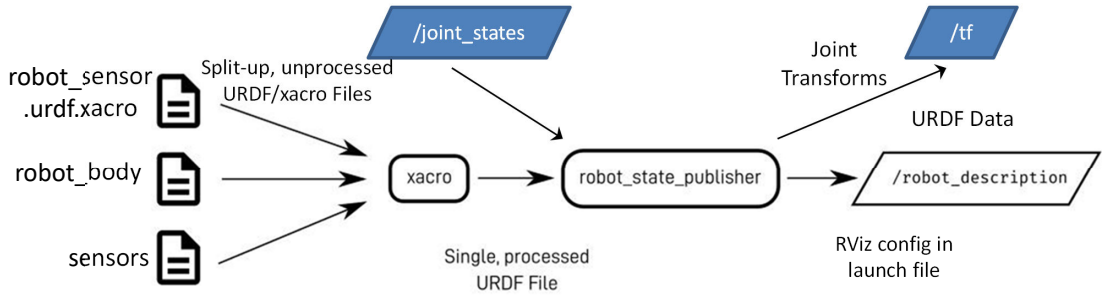
ROS URDF models

$ urdf_to_graphiz mrobot_chassis.urdf



ROS URDF models

```
<xacro:include filename=".xacro" />
```



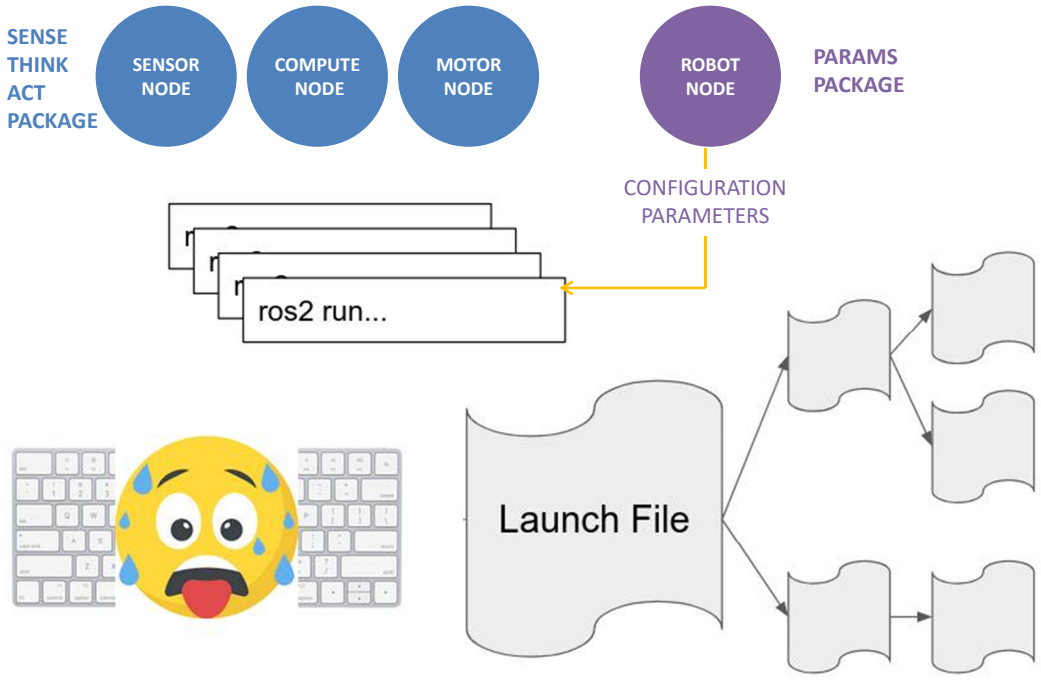

```
<?xml version="1.0"?>
<robot name="robot_name" xmlns:xacro="http://www.ros.org/wiki/xacro>"
```

```
<xacro:property name="M_PI" value="3.14159"/>
```

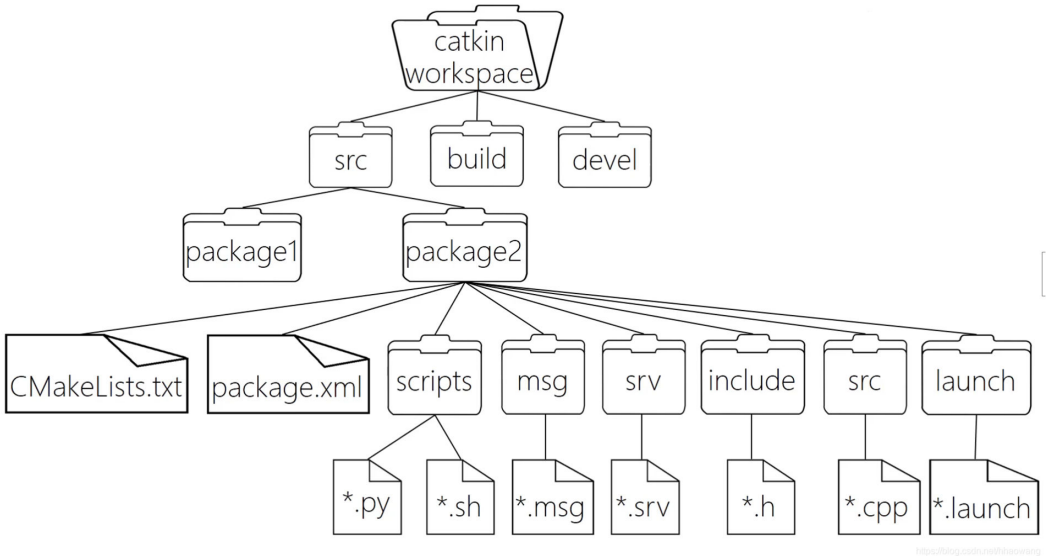
```
<xacro:property name="M_PI" value="3.14159"/>
```

```
<xacro:macro name="urdf_object">
```

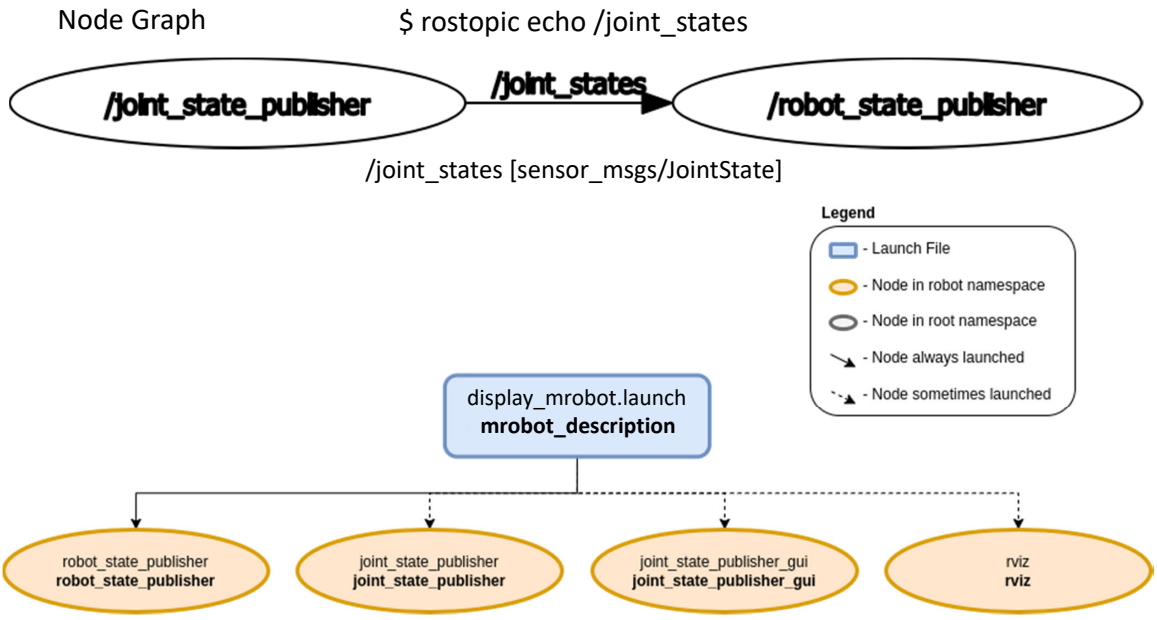
ROS URDF models



ROS URDF models



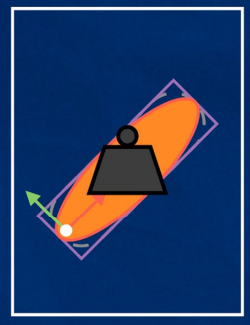
ROS applications



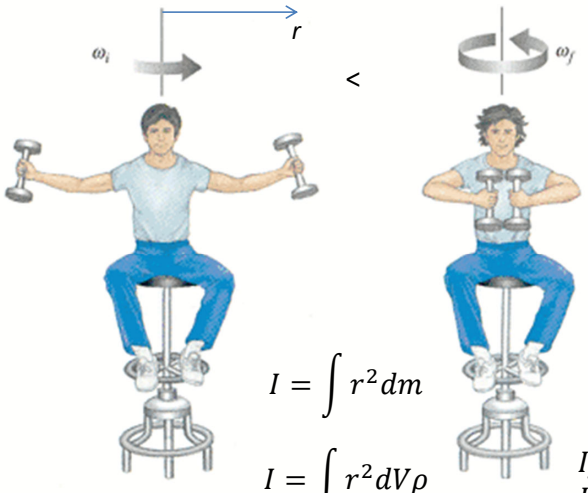
ROS visualizations

```
<link name="base_link">
    <visual>
        <origin xyz=" 0 0 0" rpy="0 0 0" />
        <geometry>
            <cylinder length="0.005" radius="0.13"/>
        </geometry>
        <material name="yellow">
            <color rgba="1 0.4 0 1"/>
        </material>
    </visual>
    <collision>
        <origin xyz=" 0 0 0" rpy="0 0 0" />
        <geometry>
            <cylinder length="0.005" radius="0.13"/>
        </geometry>
    </collision>
    <inertial>
        <mass value="2" />
        <origin xyz="0 0 0" />
        <inertia ixx="0.01" ixy="0.0" ixz="0.0"
                 iyy="0.01" iyz="0.0" izz="0.5" />
    </inertial>

</link>
```

```
<link name="arm_link">
    <visual>
        <geometry>
        <origin>
        <material>
    </visual>
    <collision>
        <geometry>
        <origin>
    </collision>
    <inertial>
        <mass>
        <origin>
        <inertia>
    </inertial>
</link>
```



```
<launch>
        <param name="robot_description" textfile="$(find
mrobot_description)/urdf/mrobot_chassis_withPhy.urdf" />
```



<

$$I = \int r^2 dm$$

$$I = \int r^2 dV\rho$$

$$I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix}$$

$$I_{xx} = \int \rho(\hat{y}^2 + \hat{z}^2) dV$$

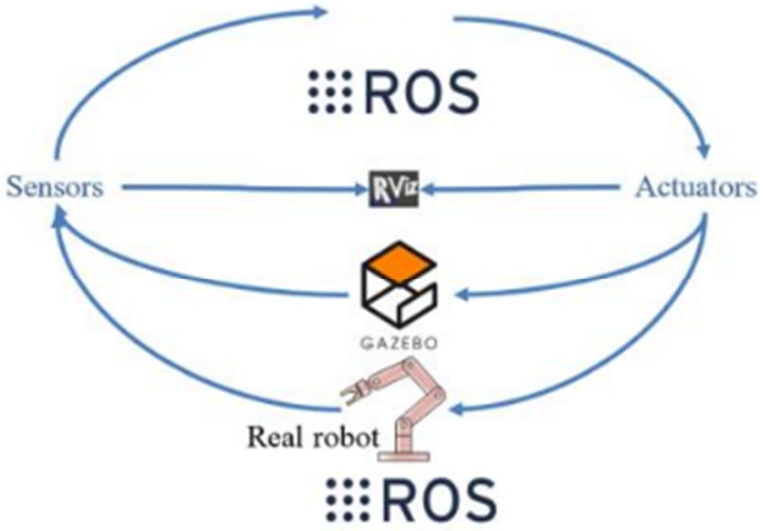
$$I_{xy} = \int \rho(\hat{x}\hat{y}) dV$$

I_{xx} ***moment*** *of inertia* about the *x*-axis
I_{xy} ***product*** *of inertia* about a pair of given perpendicular axes

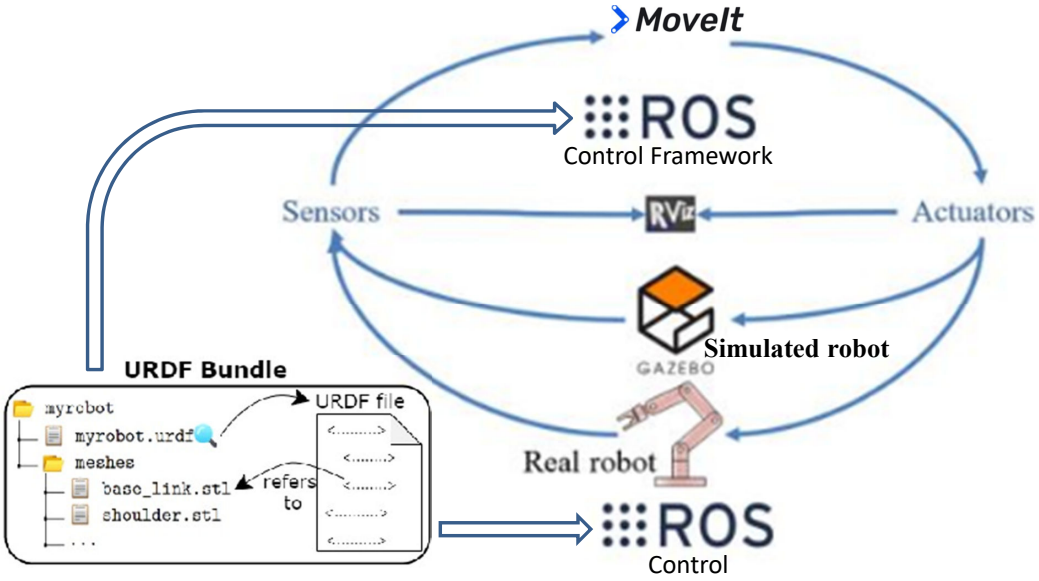
center of gravity $I \triangleq \iiint_V r^2 \rho(r) dV$

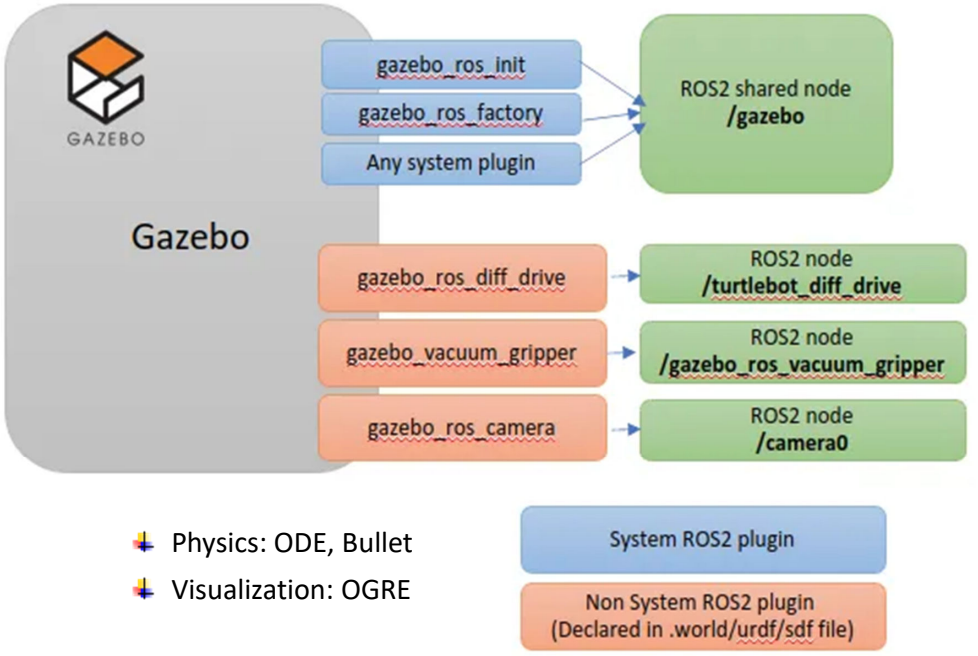
Physical properties in ROS URDF

Design Simulations in Robotics

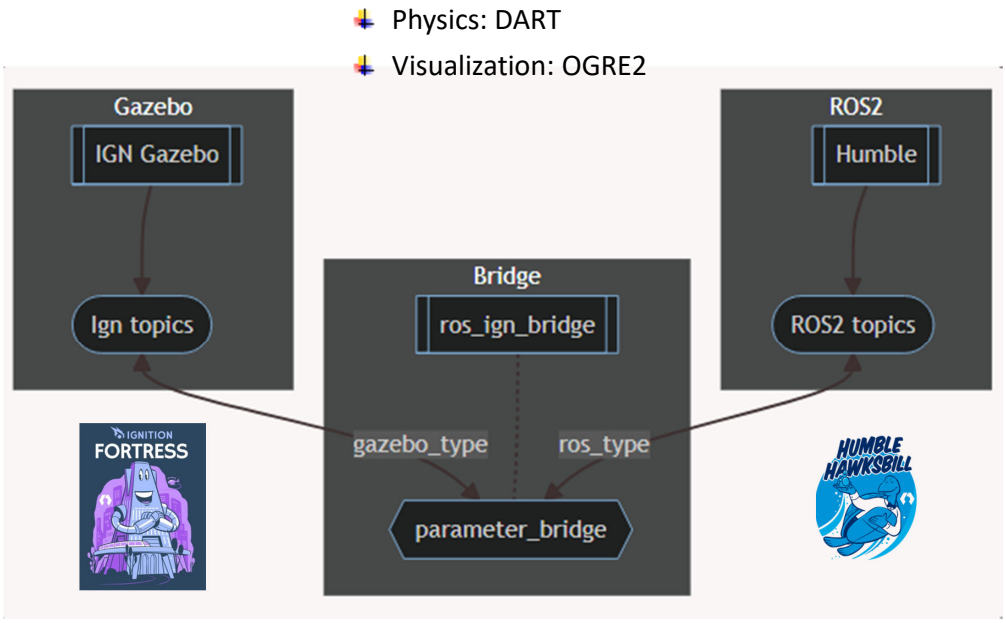


". . . RViz shows you what the robot thinks is happening, while Gazebo shows you what is really happening."

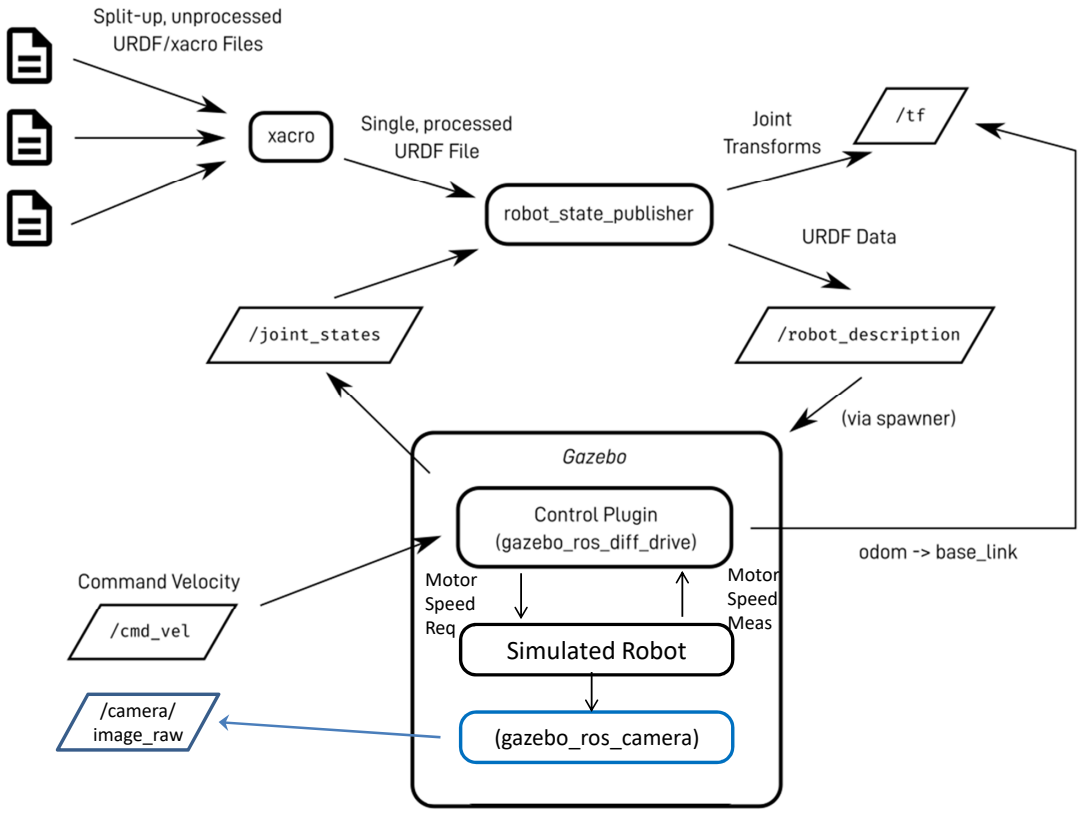




Gazebo Classic

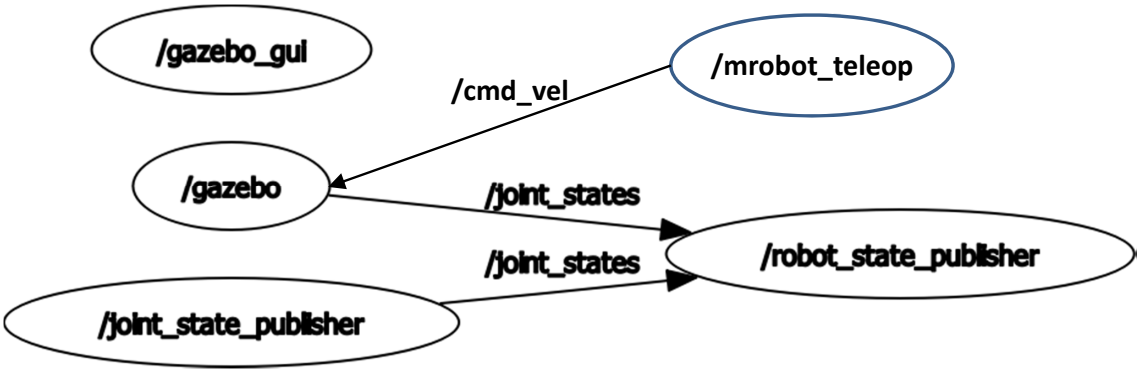


Gazebo Ignition

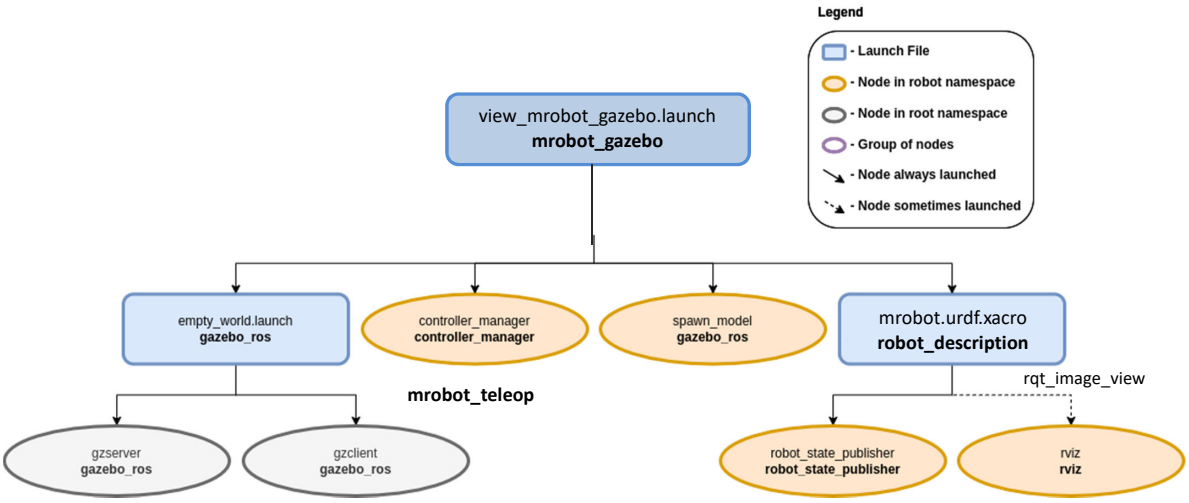


Node Graph

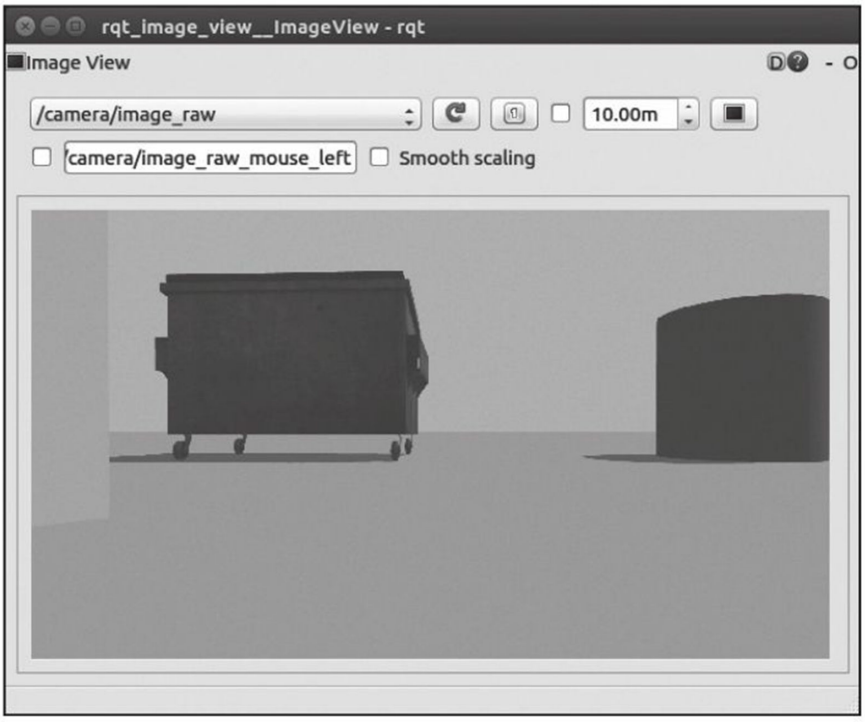
$ rostopic echo /cmd_vel



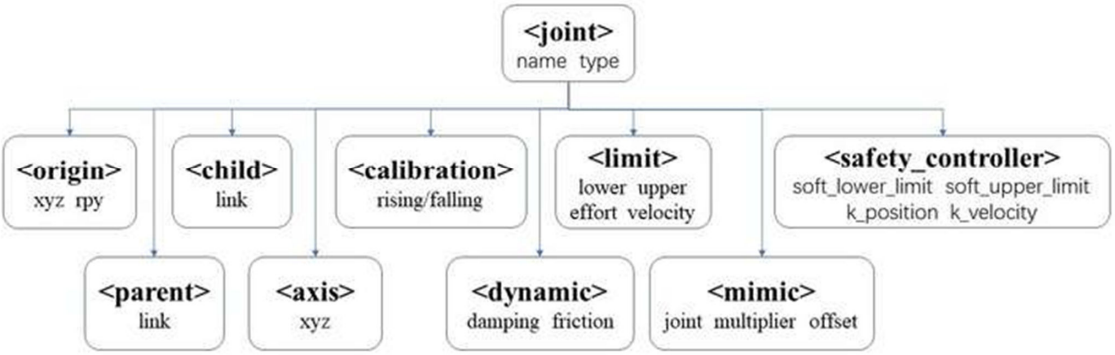
ROS simulation

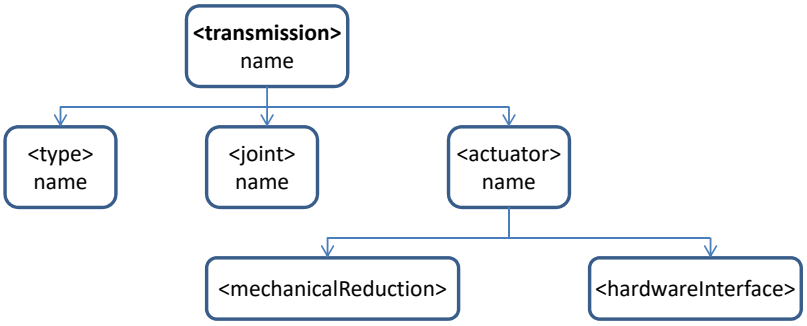


ROS simulation

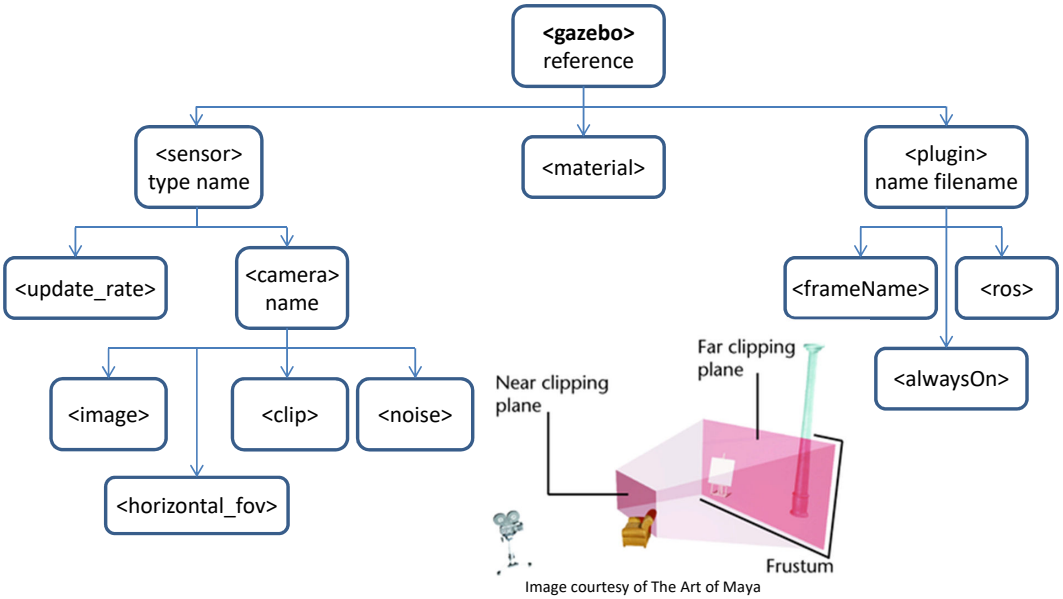


ROS simulation

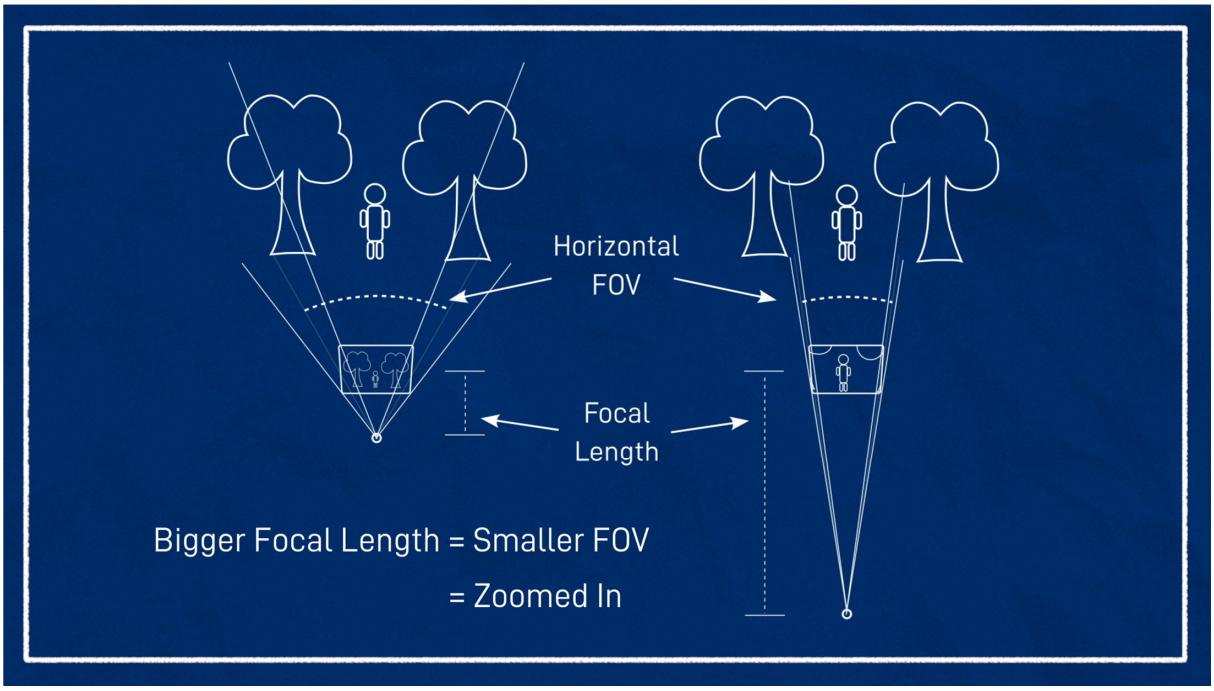




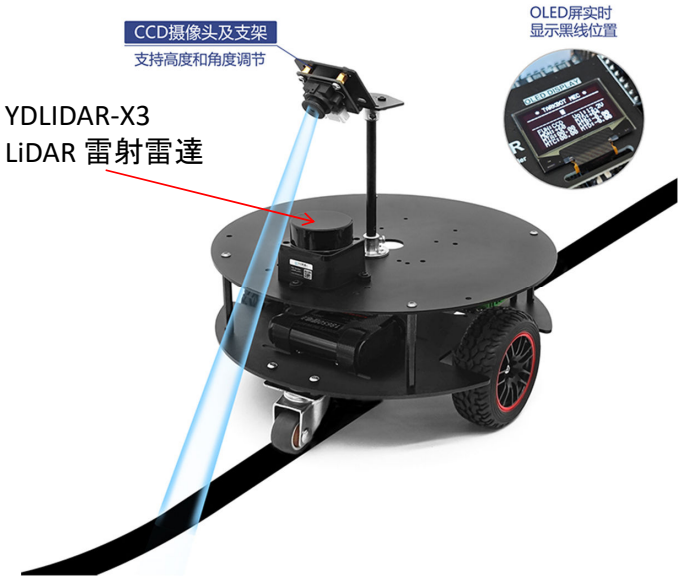
Gazebo simulation



Gazebo simulation



Gazebo simulation



Homework

https://repo.anaconda.com/miniconda/Miniconda3-py311_24.5.0-0-Windows-x86_64.exe

>Downloads\Miniconda3-py311_24.5.0-0-Windows-x86_64.exe
/InstallationType=JustMe /AddToPath=0 /RegisterPython=0 /NoRegistry=1 /S
/D=K:\Miniconda3\py311

(base) ➢ conda install mamba -c conda-forge
(base) ➢ conda install git
(base) ➢ mamba create -p K:\Miniconda3\envs\ros_noetic python=3.11
(base) ➢ mamba activate K:\Miniconda3\envs\ros_noetic
(ros_noetic)
(ros_noetic) ➢ conda config --env --add channels conda-forge
(ros_noetic) ➢ conda config --env --add channels robostack-staging
(ros_noetic) ➢ conda config --env --remove channels defaults
(ros_noetic) ➢ mamba install ros-noetic-desktop ros-noetic-gazebo-ros
(ros_noetic) ➢ mamba install ros-noetic-gazebo-plugins ros-noetic-gazebo-ros-pkgs ros-noetic-gazebo-ros-control ros-noetic-xacro
(ros_noetic) ➢ mamba deactivate

(base) ➢ mamba activate K:\Miniconda3\envs\ros_noetic