

1 移动机器人的建图、定位与导航简介

导航是移动机器人执行任务所需要必备的能力，移动机器人成功的导航需要四部分的支持：感知，机器人收集并理解传感器的信息，提取有意义的信息；定位，机器人确定它在环境中的位置；认知，机器人决定如何行动以达到目标；运动控制，机器人调节它的运动输出，以实现期望的轨迹、到达目标。

1.1 定位问题

在这四个部分中，定位一直是最受关注的部分，也是研究的热点问题，定位的实质是解决机器人“我在哪里？”的问题。定位不仅意味着要确定机器人在地球参考系中的绝对位置与姿态。考虑到机器人与环境或是人的交互，也许需要确定它在环境中或者与环境中的人相对的位置与姿态。同时，在机器人环境认知的部分，为了达到目标，机器人会选择一种可以到达特定目标位置的策略，这时仅仅知道机器人在环境中的位置与姿态还不够，机器人还需要获取或建立一个环境模型，即一张环境地图，它帮助机器人规划一条达到目标的路径。所以，定位仅仅是简单地确定机器人在空间中一个绝对位姿，它意味构建一张地图，而后确定机器人相对于地图的位置与姿态。定位研究中有三种类型问题，它们分别是：位置跟踪、全局定位和绑架机器人问题。

- 位置跟踪：在位置跟踪中，机器人的当前位置是根据对它的以前位置（跟踪）的知识而更新的。这说明机器人的初始位置假定为已知的。另外，机器人姿态的不确定性必须小。如果不确定性太大，位置跟踪会使定位机器人失败。
- 全局定位：相反，全局定位假定机器人的初始位置是不知道的。这意味着机器人可以被放在环境中的任何地方，不必有关于环境的知识——能够在环境内全局地定位。
- 绑架机器人问题：绑架机器人问题实际是机器人并被转移到环境中任意位置的情况。绑架机器人问题与全局定位问题相似，只要机器人知道已被绑架就需要确定它在环境中的位置与姿态。绑架问题的困难来源于，机器人不知道是否被转移至另一个地方，而从绑架中能够获得机器人的位姿信息，是机器人自主运行的一个必要条件。

1.2 地图

地图的描述实际上就是表示机器人移动所处的环境的问题，是表示机器人可能位置或位置的对偶问题。有关环境表示方法所做的决策，会影响到机器人位置表示可用的选择。通常，位置表示的准确性受到地图表示准确性的限制。在选择一个特殊的地图表示方法时，必须了解三个基本关系：

- 地图的精度必须恰当地匹配机器人需要达到目标的精度；
- 地图的精度和所表示的特征类型必须匹配机器人传感器所返回的数据类型和精确性；
- 地图表示的复杂性直接影响有关建图、定位和导航推理的计算复杂性。

1.3 概率栅格地图表示

连续值的地图是环境精确分解的一种方法。环境特征的位置可以在连续空间中精密地予以标记。但是仅仅在二维空间中进行连续地图的表示，都会因为更高的维数引起计算上的爆炸。而将地图进行某种形式的分解是对环境描述的一种抽象化表示，栅格地图就是一种固定分解的地图表达形式，在这里环境是棋盘格化的。为了获得地图表达，将连续的现实环境变换成离散近似，在一个占有栅格中，环境被表示成离散栅格。这里，各个单元或者被填满（障碍部分）；或者是空的（自由空间部分）。当机器人装备基于测距传感器时，这特别有价值，因为各传感器的测距值与机器人的绝对位置结合在一起，可以直接被用于更新各单元的填充值或空值。在占有栅格中，各单元可能有一个计数器。借此，0 值说明单元还未被任何测距所“击中”，所以单元可能是自由空间。当测距点测的数目增加时，单元的值递增超过一定的阈值，单元必定成为障碍物（1 值）。当测距点击越过单元，点击一个更远单元时，单元的值通常会被打折扣（0-1 值），这个折扣值就代表该栅格被占据的概率。概率栅格方法有两个主要缺点：

- 在机器人存储器中地图的尺寸随着环境规模的增长而增大。如果用小的单元尺寸，这个尺寸迅速变得难以办到。占有栅格方法与现实环境的假设不相容，后者能使连续表示在大的、稀疏环境中具有潜在的非常小的存储需求。相反，概率栅格必须为矩阵中的每一个单元保留存储器；
- 任何像这样的固定分解，不管环境细节如何，在环境上都要预先加上一个几何栅格，在几何特征并不明显的环境里，这显然是不合适的。

1.4 基于概率地图的定位

基于概率地图的定位技术，明确地辨识了可能的机器人位姿概率。通常，移动机器人定位的概率问题一直被认为是：处理从传感器获取的数据受测量误差影响的问题。所以，我们将计算机器人处在给定方位的概率称为“概率机器人学”。概率机器人学的关键性思想是用概率论来表示不确定性，换句话说，不是给出当前机器人方位的一个单独的最好估计，而是将机器人方位表示为对所有可能的机器人姿态的一个概率分布。当前，有两类较为流行的基于概率地图的定位方法：第一类是马尔可夫（Markov）定位，对所有可能的机器人位置，使用一个明确地指定的概率分布；第二个方法是卡尔曼滤波（Kalman Filter）定位，使用机器人位置的高斯概率密度的表示。与马尔可夫定位不同，卡尔曼滤波定位在机器人的方位空间中不独立考虑各个可能的姿态。有趣的是，如果机器人位置的不确定性被假定为具有高斯形式，则卡尔曼滤波定位过程由马尔可夫定位的公理产生。

2 状态空间模型与离散时间隐 Markov 模型

我们已在《现代控制理论》中学过系统的状态空间模型，这里我们简单地复习一下。以连续时不变系统为例，其状态空间由以下常微分方程表示：

$$\dot{x} = Ax + Bu, \quad y = Cx, x(0) = x_0 \quad (1)$$

其中 $x \in \mathbb{R}^n$ 为系统的状态， $u \in \mathbb{R}^m$ 为系统的外部输入， $y \in \mathbb{R}^p$ 是系统的输出， x_0 为系统的初始状态。我们不能直接对系统内部的状态 x_t 进行测量，而只能对其所呈现出来的输出 y_t 进行观测。

Example 1 考虑以下电路系统：

由基尔霍夫电压定律可得：

$$-u(t) + u_C(t) + u_L(t) + y(t) = 0,$$

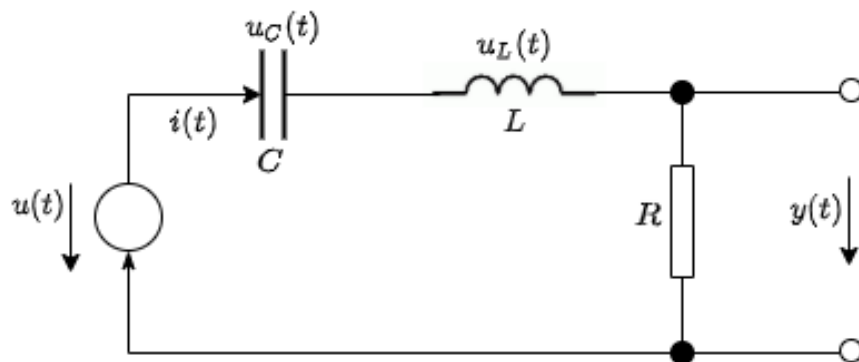


Figure 1: LCR 振荡电路

将电容、电感的模型 $C \frac{d}{dt} u_C(t) = i(t)$, $L \frac{d}{dt} i(t) = -u_L(t)$ 代入上式, 并令 $x(t) := [x_1(t)^T, x_2(t)^T]^T$, $x_1(t) := u_C(t)$, $x_2(t) := i(t)$ 可得该电路系统的状态空间模型:

$$\dot{x} := \frac{d}{dt} x = \begin{bmatrix} 0 & \frac{1}{C} \\ -\frac{1}{L} & -\frac{R}{L} \end{bmatrix} x + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} u$$

实际中系统的状态空间模型往往要比线性时不变系统(1)更复杂, 其状态空间模型由以下常微分方程表示:

$$\dot{x} = \hat{f}(x, u), \quad y = \hat{h}(x) \quad (2)$$

而由于机器人通常通过计算机控制, 其控制周期受到计算机指令周期的限制, 所以在实际应用中, 我们经常将系统的状态空间模型在时域上离散化, 获得如下的离散时间状态空间模型:

$$x_{t+1} = f(x_t, u_t), \quad y_t = h(x_t), \quad t = 1, 2, 3, \dots$$

在本课程中, 我们将主要与离散时间的状态空间模型打交道。直观上, 离散时间状态空间模型可以被理解为: 给定 t 时刻的系统状态 x_t 和外部输入 u_t , 系统处于 $t+1$ 时刻的系统状态 x_{t+1} 就能够由上式计算, 也即系统处于 t 时刻的状态衍化到了 $t+1$ 时刻的状态; 而在每个时刻 t , 我们都能够观测到系统的输出 y_t 。如果该系统在每个时刻的衍化与对该系统输出的观测均受到环境中随机噪声的影响, 则我们可以得到如下状态空间模型:

$$x_{t+1} = f(x_t, u_t) + w_t, \quad y_t = h(x_t) + v_t, \quad t = 1, 2, 3, \dots, \quad (3)$$

其中 $w_t \in \mathbb{R}^n, t = 1, 2, \dots$ 被称为“过程噪声”, $v_t \in \mathbb{R}^p, t = 1, 2, \dots$ 被称为“观测噪声”, 均为随机向量。系统(3)的观测和衍化过程可由 Fig. 2表示。从 Fig. 2中可以看出:

1. 对每个时刻 t 的状态 x_t 来说, 若上一时刻的状态 x_{t-1} 和系统输入 u_{t-1} 已知, 则 x_t 的分布与之前历史的状态 x_1, \dots, x_{t-2} 无关;
2. 对每个时刻 t 的观测 y_t 来说, 若当前时刻的状态 x_t 已知, 则 y_t 的分布与与之前历史的状态 x_1, \dots, x_{t-2} 无关。

这事实上符合“离散时间隐 Markov 模型”的定义; (3)定义了一种离散时间隐 Markov 模型。下面我们给出离散时间隐 Markov 模型严谨的数学定义:

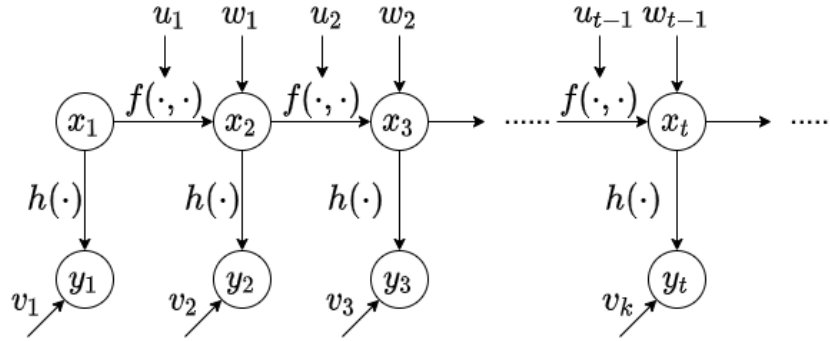


Figure 2: 离散时间隐 Markov 模型的观测和衍化过程图示

Definition 1 令 $\{X_t\}$ 和 $\{Y_t\}$ 为两组离散时间的随机过程，其中 $t \geq 1$ 。若

- 对于任意的 $t \geq 1$, X_t 不能被直接观测，且 $\mathbb{P}(x_{t+1} = x \mid X_1 = x_1, X_2 = x_2, \dots, X_t = x_t) = \mathbb{P}(x_{t+1} \in \mathcal{A} \mid x_t)$ (假设这两个条件概率分布是良定义的)；
- 对于任意的 $t \geq 1$, x_1, \dots, x_t 和 Borel 集 \mathcal{A} , 均有 $\mathbb{P}(Y_t \in \mathcal{A} \mid X_1 = x_1, \dots, X_t = x_t) = \mathbb{P}(Y_t \in \mathcal{A} \mid X_t = x_t)$ 成立，

则 $\{X_t\}$ 和 $\{Y_t\}$ 是一个离散时间隐 Markov 模型。

3 概率论基础知识与多元高斯分布

本节将回顾一些本课程中会用到的概率论基础知识。

1. 随机向量 $\xi = [\xi_1, \dots, \xi_n]^T$ 是一个从“结果空间” Ω 到 \mathbb{R}^n 的映射。我们用 $x = [x_1, \dots, x_n]^T \in \mathbb{R}^n$ 表示随机向量 X 的“结果”或“实现”。在本课程中，概率分布函数 $F_\xi(x)$ 和概率密度函数 $p_\xi(x)$ 可被大致地理解为：

$$F_\xi(x) = \mathbb{P}(\xi_1 \leq x_1, \xi_2 \leq x_2, \dots, \xi_n \leq x_n), \quad p_\xi(x) = \frac{dF_\xi(x)}{dx_1 dx_2 \dots dx_n};$$

概率分布函数 $F_\xi(x)$ 与概率密度函数 $p_\xi(x)$ 具有以下性质：

- $F_\xi(x) = \int_{-\infty}^x p_\xi(x) dx$;
- $\int_{-\infty}^{\infty} p_\xi(x) dx = 1$;
- 若 $X \in \mathbb{R}^n, Y \in \mathbb{R}^m$ 的联合概率密度函数为 $p_{X,Y}(x, y)$, 则 $p_X(x) = \int_{-\infty}^{\infty} p_{X,Y}(x, y) dy$, $p_Y(y) = \int_{-\infty}^{\infty} p_{X,Y}(x, y) dx$ 。

2. 在本课程中，随机向量 X 的数学期望可被大致地理解为：

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x p_X(x) dx;$$

而 X 的函数 $f(X)$ 的数学期望为

$$\mathbb{E}[f(x)] = \int_{-\infty}^{\infty} f(x)p_X(x)dx.$$

对于任意的随机向量 $X, Y \in \mathbb{R}^n$ 和常数 $\alpha \in \mathbb{R}$, 都有以下性质:

$$\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y] \quad \mathbb{E}[\alpha X] = \alpha\mathbb{E}[X];$$

X 的协方差矩阵的定义为

$$\begin{aligned} \text{cov}(X) &= \mathbb{E} \left[(X - \mathbb{E}[X]) (X - \mathbb{E}[X])^T \right] = \mathbb{E} [XX^T - \mathbb{E}[X]X^T - X\mathbb{E}[X]^T + \mathbb{E}[X]\mathbb{E}[X]^T] \\ &= \mathbb{E}[XX^T] - \mathbb{E}[X]\mathbb{E}[X]^T - \mathbb{E}[X]\mathbb{E}[X]^T + \mathbb{E}[X]\mathbb{E}[X]^T \\ &= \mathbb{E}[XX^T] - \mathbb{E}[X]\mathbb{E}[X]^T \end{aligned}$$

3. 对于 $x \in \mathbb{R}$, 高斯积分

$$\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}.$$

Proof: 我们首先需证明该积分值存在。令

$$I(a) = \int_{-a}^a e^{-x^2} dx,$$

我们需证明 $\lim_{a \rightarrow \infty} I(a) < \infty$ 。注意到由于 $-xe^{-x^2} > e^{-x^2}, \forall x \in (-\infty, -1]$ 以及 $xe^{-x^2} > e^{-x^2}, \forall x \in [1, \infty)$, 下式一定成立:

$$\begin{aligned} \int_{-\infty}^{\infty} e^{-x^2} dx &= \int_{-\infty}^{-1} e^{-x^2} dx + \int_{-1}^1 e^{-x^2} dx + \int_1^{\infty} e^{-x^2} dx \\ &< \int_{-\infty}^{-1} -xe^{-x^2} dx + \int_{-1}^1 e^{-x^2} dx + \int_1^{\infty} xe^{-x^2} dx \\ &= -\frac{1}{2} \int_{-\infty}^{-1} e^{-x^2} dx^2 + \int_{-1}^1 e^{-x^2} dx + \frac{1}{2} \int_1^{\infty} e^{-x^2} dx^2 \\ &= \frac{1}{2} \int_1^{\infty} e^{-y} dy + \int_{-1}^1 e^{-x^2} dx + \frac{1}{2} \int_1^{\infty} e^{-y} dy \\ &= e^{-1} + \int_{-1}^1 e^{-x^2} dx < \infty. \end{aligned}$$

所以 $\lim_{a \rightarrow \infty} I(a)$ 存在。

$$I^2(a) = \left(\int_{-a}^a e^{-x^2} dx \right) \left(\int_{-a}^a e^{-y^2} dy \right) = \int_{-a}^a \int_{-a}^a e^{-(x^2+y^2)} dx dy$$

把上述重积分转到极坐标系中, 可得

$$(1 - e^{-a^2}) \pi = \int_0^{2\pi} \int_0^a re^{-r^2} dr d\theta \leq I^2(a) \leq \int_0^{2\pi} \int_0^{\sqrt{2}a} re^{-r^2} dr d\theta = (1 - e^{-2a^2}) \pi$$

对上述不等式取极限, 可得 $\pi \leq \lim_{a \rightarrow \infty} I^2(a) \leq \pi$, 所以 $\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$. □

4. 如果一个随机向量 $X \in \mathbb{R}^n$ 的概率密度函数 $p_X(x)$ 为

$$p_X(x) = \frac{1}{(2\pi)^{n/2} \det(P)^{1/2}} e^{-\frac{1}{2}(x-m)^T P^{-1}(x-m)},$$

则 X 服从多元高斯分布, 记为 $X \sim \mathcal{N}(m, P)$ 。

若 $X \sim \mathcal{N}(m, P)$, $A \in \mathbb{R}^{m \times n}$, 且 A 行满秩, 则 $Y = AX + b \sim \mathcal{N}(Am + b, APA^T)$ 。

5. 连续随机向量的贝叶斯公式为

$$p_{X|Y}(x) = \frac{p_{X,Y}(x, y)}{p_Y(y)} = \frac{p_{Y|X}(y)p_X(x)}{p_Y(y)},$$

其中 $p_{X|Y}(x)$ 为给定 Y 条件下 X 的后验概率密度函数, $p_{X,Y}(x, y)$ 为 X 和 Y 的联合分布的概率密度函数, $p_Y(y)$ 为 Y 的边缘分布的概率密度函数, $p_{Y|X}(y)$ 是给定 X 条件下 Y 的似然函数, $p_X(x)$ 是 X 的先验概率密度函数。

后文为了简便起见, 将使用下列写法: $p(x|y) := p_{X|Y}(x)$, $p(x, y) := p_{X,Y}(x, y)$, $p(y) := p_Y(y)$, $p(x) := p_X(x)$, $p(y|x) := p_{Y|X}(y)$ 。

Example 2

$$p(x, y|z) = \frac{p(x, y, z)}{p(z)} = \frac{p(x|y, z)p(y, z)}{p(z)} = p(x|y, z)p(y|z).$$

Example 3 假设 $\{x_t\}$ 和 $\{y_t\}$ 是一个离散时间隐 Markov 模型。 $x_1, \dots, x_n, y_1, \dots, y_n$ 的联合分布的概率密度函数满足

$$p(x_{1:n}, y_{1:n}) = p(y_1|x_1)p(x_1) \prod_{t=2}^n p(y_t|x_t)p(x_t|x_{t-1}).$$

Proof:

$$\begin{aligned} p(x_{1:n}, y_{1:n}) &= p(y_n|x_n, \underbrace{x_{1:n-1}, y_{1:n-1}}_{\text{Markov 特性}})p(x_{1:n}, y_{1:n-1}) \\ &= p(y_n|x_n)p(x_n|x_{n-1}, \underbrace{x_{1:n-2}, y_{1:n-1}}_{\text{Markov 特性}})p(x_{1:n-1}, y_{1:n-1}) \\ &= p(y_1|x_1)p(x_1) \prod_{t=2}^n p(y_t|x_t)p(x_t|x_{t-1}) \end{aligned}$$

□

在移动机器人定位和建图中, 贝叶斯公式将扮演非常重要的角色。移动机器人上配备了激光雷达、毫米波雷达、声呐、IMU、GPS 等传感器。传感器返回的数据带有测量误差; 而机器人本身的移动会由于控制算法的精度、环境的变化、地面打滑等, 同样会引入误差, 所以机器人总体是可被理解为一个隐 Markov 模型, 包含机器人在 t 时刻的位姿 $\{x_t\}$, 传感器在 t 时刻的观测值 $\{y_t\}$ 以及地图 M 。我们可以进一步将移动机器人的定位、建图和 SLAM 大致地理解为:

- 安装在机器人上的传感器对当前环境（地图）的观测，总是基于机器人当前的位姿，其在每一时刻 t 观测值的概率密度函数为 $p(y_t|x_t, M)$;
- 离线建图：基于全部的传感器信息和机器人位姿信息，建立地图，即从 $p(M|x_{1:N}, y_{1:N})$ 中进行采样 $1 \leq t \leq N$;
- 定位：基于传感器历史信息与地图，推算机器人当前时刻 t 的位姿信息，即从 $p(x_t|y_{1:t}, M)$ 中进行采样;
- SLAM：基于传感器历史信息，推算机器人当前时刻 t 的位姿信息和地图信息 M_t ，即从 $p(x_t, M_t|y_{1:t})$ 中进行采样。

4 ROS 使用初探

4.1 机器人操作系统 ROS

1. ROS 是机器人操作系统（Robot Operating System）的英文缩写，是一个适用于机器人的开源的元操作系统。
2. 它提供了操作系统应有的服务，包括硬件抽象，底层设备控制，常用函数的实现，进程间消息传递，以及包管理。它也提供用于获取、编译、编写和跨计算机运行代码所需的工具和库函数。



Figure 3: ROS 系统发展进程

4.2 ROS 的架构

1. ROS 就是一套通信机制、一套开发工具、一系列应用工具加一个庞大的生态系统组成的集合，目标是提高机器人研发中的软件复用率。针对这样的目标，如图 Fig. 4，ROS 有五大特点。
2. ROS 的组成有以下四大部分，即通讯机制 + 开发工具 + 应用功能 + 生态系统。
3. ROS 提供了很多开发工具，目的是提高机器人开发效率。第一个是命令行工具，可以直接在 terminal 内进行操作；还有 TF 工具，可以帮助进行坐标变换；还有 QT 工具箱，提供很多可视化的工具；RVIZ 是一个三维可视化工具，可以显示机器人的全部数据；Gazebo 是一个三维仿真平台，可以进行机器人仿真。

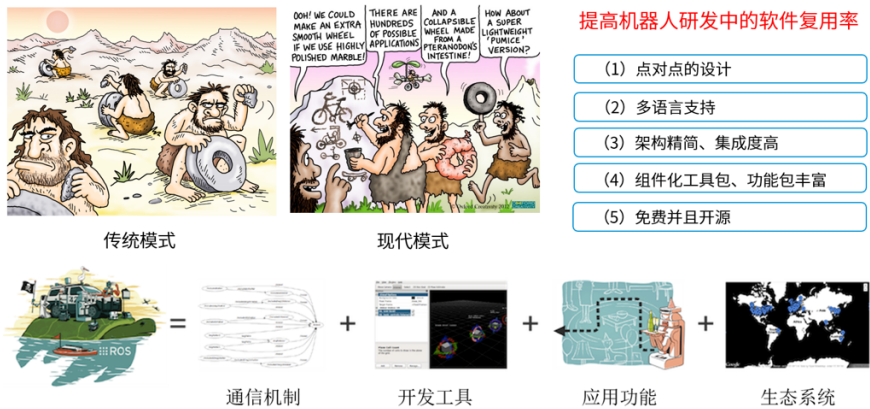


Figure 4: ROS 系统架构

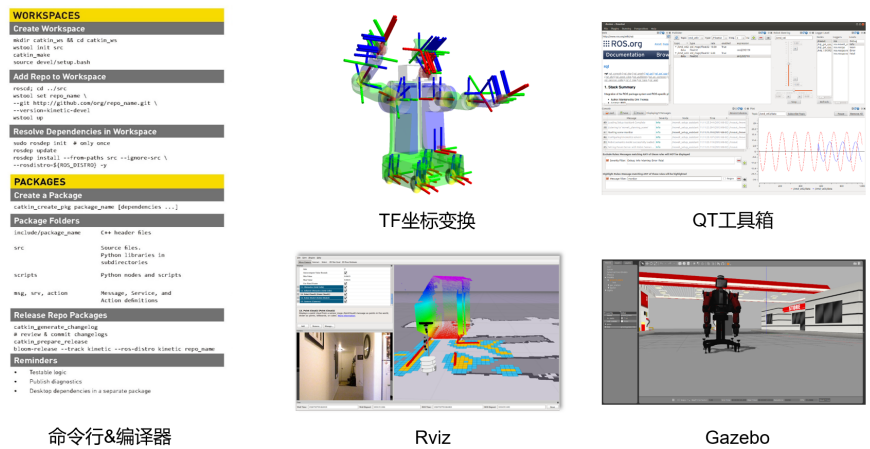


Figure 5: ROS 开发工具

4. 如图 Fig. 6, ROS 在生态层面讲就是这么几个重要的资源。发行版指 ROS 发布的版本。软件源中放置的是可直接安装的功能包。ROS Wiki 是 ROS 信息记录最完善的网站。邮件列表更多的用在 ROS 更新维护, 开发者使用较少。ROS Answers 是专门的 ROS 问答网站。博客会看到一些 ROS 的新闻、图片、视频等。

1. 发行版 (Distribution) : ROS发行版包括一系列带有版本号、可以直接安装的功能包。

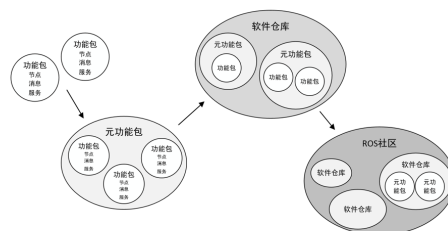
2. 软件源 (Repository) : ROS依赖于共享网络上的开源代码, 不同的组织机构可以开发或者共享自己的机器人软件。

3. ROS wiki: 记录ROS信息文档的主要论坛。

4. 邮件列表 (Mailing list) : 交流ROS更新的主要渠道, 同时也可以交流ROS开发的各种疑问。

5. ROS Answers: 咨询ROS相关问题的网站。

6. 博客 (Blog) : 发布ROS社区中的新闻、图片、视频 (<http://www.ros.org/news>)



ROS社区资源的组织形式

Figure 6: ROS 生态资源

4.3 ROS 的通讯机制

1. ROS 的通讯机制为松耦合分布式通讯。类似于图 Fig. 7在两个电脑中, 分别有一个节点在使用 topic 进行通讯, 为了更好的通讯, Central Node 负责去管理各个节点之间的通讯。

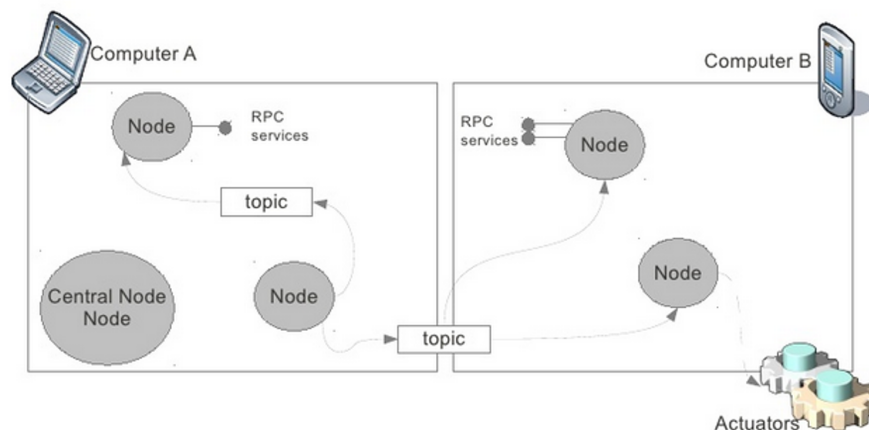


Figure 7: PC 通讯机制

2. ROS 通讯机制方面会看到这样一张图 (如图 Fig. 8), 机器人之间的通讯会形成一个网络, 称为计算图。在图当中会有很多节点, 椭圆部分, 方框是节点的命名空间, 箭头是数据传输的关系。这就是 ROS 在通讯层面的松耦合分布式通讯。松耦合就是说每一个节点不是和另一个节点强绑定的; 分布式是指各个节点可以运行在不同的计算机内
3. 在 Ros 中, 节点是最小可执行单元, 就是每一个进程。编译完成后的每一个可在执行文件就是一个节点, 实现一个功能。注意节点不能重名。节点管理器负责控制节点的注册及生成, 他

松耦合分布式通信

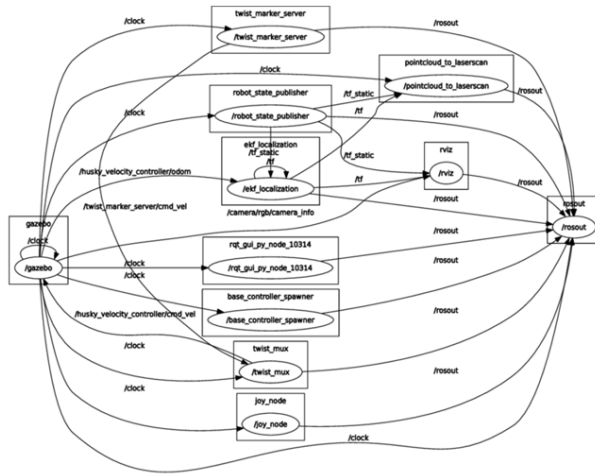


Figure 8: ROS 通讯机制

会跟踪和记录通讯，会辅助节点查找，建立连接，也会提供全局的参数服务器。

4. 话题是节点之间传输数据的一种单向的异步通讯的机制。下面这张图发布者通过话题发布数据给订阅者，单向的传输。其中通讯时的数据定义即为消息，具有一定的类型和数据结构，可以使用 ROS 提供的，也可以使用自定义的消息类型。

4.4 ROS 的安装

本课程的开发环境默认为 Ubuntu 18+ROS Melodic，为简化大家的安装流程，已经配置好 Docker。所以，环境安装可以选用 Windows 环境下安装 ROS Docker，Linux 环境下安装 ROS Docker 两种。（已装有虚拟机和双系统的同学，建议直接安装 ROS 或者参照 Ubuntu 环境下安装 ROS Docker 教程）

4.4.1 Windows 环境下，安装 ROS Docker:

一、安装 Docker

先决条件: 首先判断自己电脑是否满足以下 5 个条件，若不满足，建议安装虚拟机和双系统之后，参照 Ubuntu 环境下安装 ROS Docker 教程。

1. Windows 11 64bit 专业版、家庭版、企业版或教育版 21H2 或更高版本；
Windows 10 64bit 专业版或家庭版 2004(内部版本 19041) 或更高版本；企业版或教育版 1909(内部版本 18363) 或更高版本
查看方法：右键【此电脑】，点击【属性】。
2. BIOS 设置中启用硬件虚拟化支持
3. Windows 上启用 WSL2 功能
4. CPU 支持二级地址转换 (SLAT)
5. 内存 4GB 以上

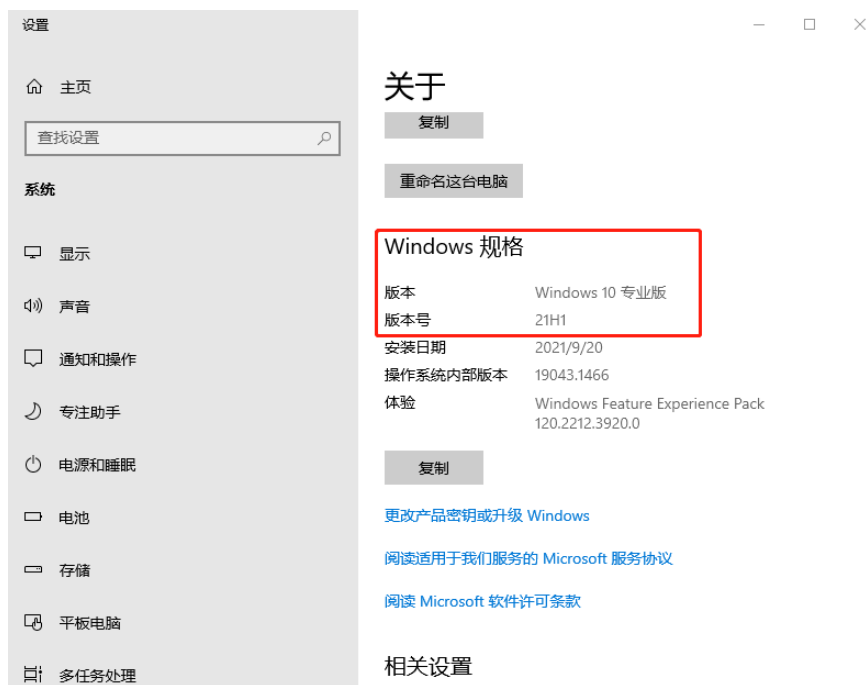


Figure 9: Windows Version

6. 系统盘（通常是 C 盘）有大于 20GB 的磁盘空间。

Windows 专业、企业或教育版安装步骤

1. 查看 BIOS 是否启用硬件虚拟化支持

查看方法：右键【Windows 开始图标】，打开【任务管理器】，【性能】一栏中，点击 CPU，看虚拟化是否是启用状态。若未启用，进入 BIOS（根据电脑型号不一样，有些是开机时多按几次 F2），然后选择在上方的【Configuration】选项，然后将最下方的【Intel Virtual Technology】，设置为【Enable】的状态，然后就可以按下 F10 保存退出，重启进入系统即可；如果是 AMD 平台同样是在 BIOS 中选择【Configuration】的选项，将【SVM Support】的选项设置成 Enable 的状态，同样是按下【F10】保存退出即可。

2. 开启 Hyper-V

查看方法：右键单击【Windows 开始图标】并选择【应用和功能】页面，单击【程序和功能】，单击【启用或关闭 Windows 功能】，确认 Hyper-V 已经被勾选，并单击确定按钮。按上述步骤操作完成后，会安装并开启 Hyper-V，如图 11 所示。这时需要重启操作系统。

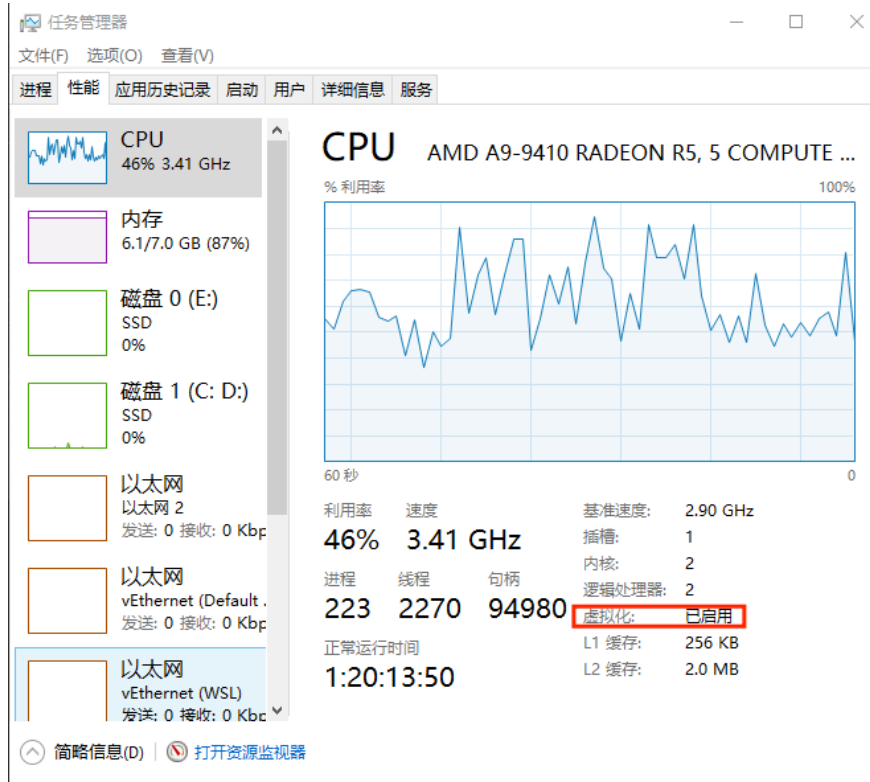


Figure 10: Virtual

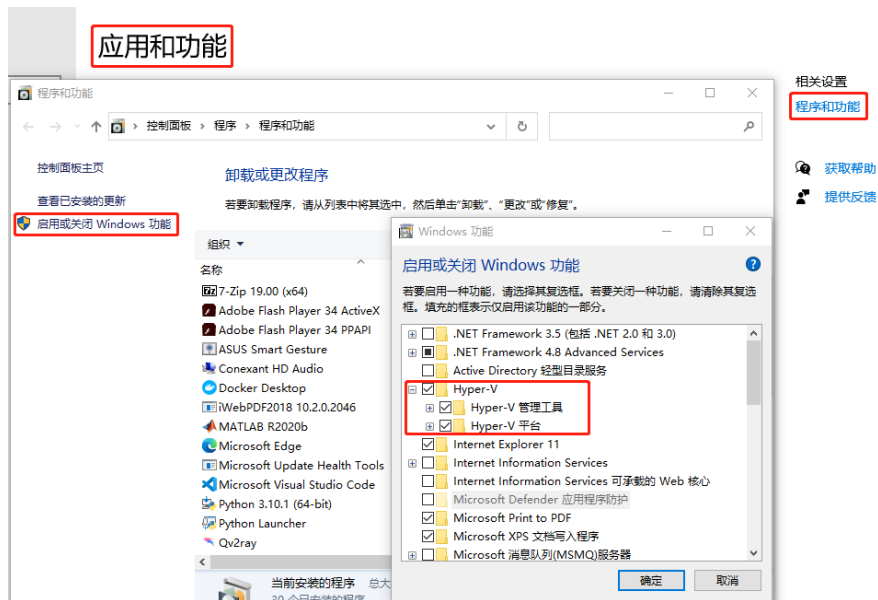


Figure 11: Hyper-V

3. 下载并安装 Docker for Windows

访问 [Docker 官网](#)，下载 Docker for Windows 安装包。等待下好后，安装时选项全部勾选后直接安装即可。安装完后，需要重启计算机，待重启结束，Docker 会自动运行。此时桌面上可以看到 Docker 小鲸鱼的图标，并且显示 Docker for Windows 的总界面如图 12 所示。

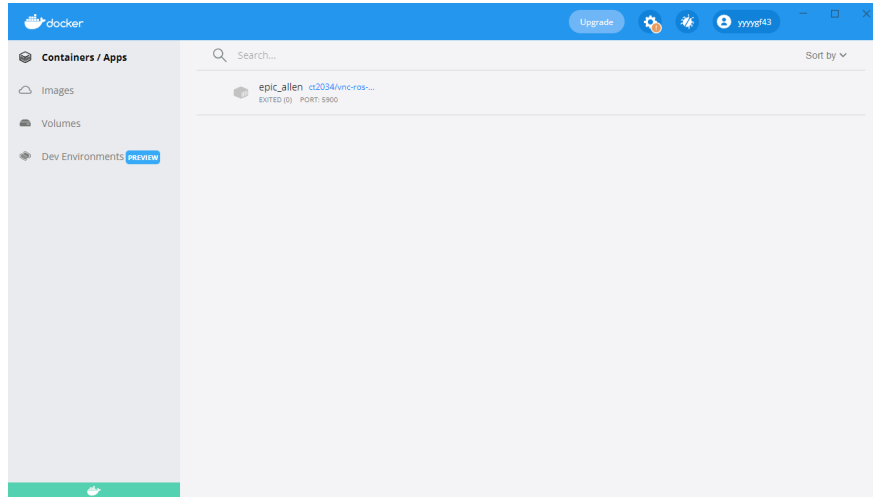


Figure 12: Docker Desktop

4. 下载并安装 Linux 内核更新包

有时候会收到“WSL 2 Installation is incomplete”的报错信息，此时去 [WSL 2 官网](#)，在步骤 4 下载 Linux 内核更新包一路安装即可。待安装好后，打开 Docker 一切正常。

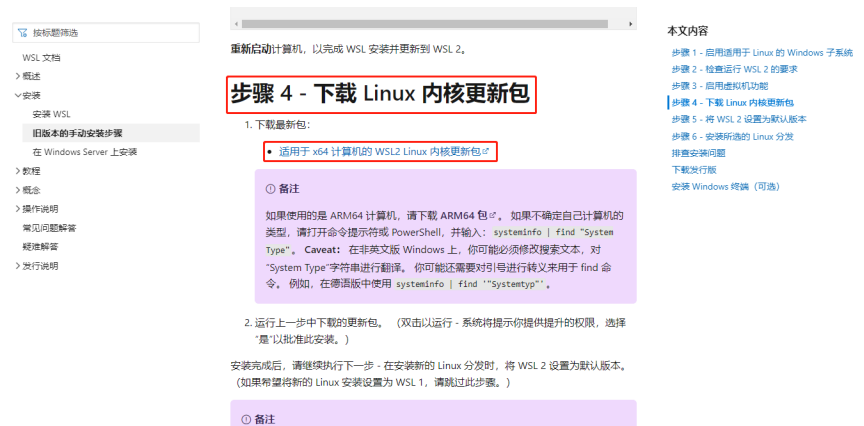


Figure 13: WSL 2 Upgrade

5. 简单检验 Docker 是否安装成功

同时按键盘上“Win+R”，打开 cmd，输入命令：

```
docker version
```

结果如图 14 所示，证明 Docker 安装成功。

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.19043.1466]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\75434>docker version
Client:
 Cloud integration: v1.0.22
 Version:          20.10.11
 API version:     1.41
 Go version:      gol.16.10
 Git commit:      dea9396
 Built:           Thu Nov 18 00:42:51 2021
 OS/Arch:        windows/amd64
 Context:         default
 Experimental:    true
```

Figure 14: Docker Version

二、配置 ROS Docker 以及可视化

1. 配置 ROS Docker

拉取 Docker 镜像：

`docker pull zehaowang/gazebo7.16`

```
C:\Windows\system32\cmd.exe
Microsoft Windows [版本 10.0.19043.1466]
(c) Microsoft Corporation. 保留所有权利。

C:\Users\75434>docker pull zehaowang/gazebo7.16
Using default tag: latest
latest: Pulling from zehaowang/gazebo7.16
0a01a72a68bc: Pull complete
cc899a5844da: Pull complete
19197c550758: Pull complete
716a454e56b6: Pull complete
12b8f8b5b3ed: Pull complete
33924d59a7d2: Pull complete
ef2e35eaf717: Pull complete
666435603b9f: Pull complete
7adb6c06d34: Pull complete
e65b84f6631c: Pull complete
90a5c5c75155: Pull complete
a41a1d669d05: Pull complete
3261bae9e229: Pull complete
1801b959b64b: Pull complete
f43e4f6dc7c3: Pull complete
b1c3ae3c74a9: Pull complete
a515b228c9a2: Pull complete
42257979557a: Pull complete
67b78f8fcff5: Pull complete
Digest: sha256:dab11f2ca3bb36279755421275031e1042d8c3c784d1d1ff042356e40070f1124
Status: Downloaded newer image for zehaowang/gazebo7.16:latest
docker.io/zehaowang/gazebo7.16:latest
```

Figure 15: pull

查看所有镜像及其相关信息：

`docker image list`

```
C:\Users\75434>docker image list
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
zhaowang/gazebo7.16 latest              28bf640c068f      32 hours ago       4.97GB
osrf/ros             noetic-desktop-full-focal 0a48f32eaf05      6 months ago       3.42GB
osrf/ros             noetic-desktop-focal    852def7e8eal      6 months ago       2.27GB
```

Figure 16: image list

利用该镜像创建容器：

`docker run -it -p 6080:80 -p 5900:5900 -e VNC_PASSWORD=mypassword zhaowang/gazebo7.16`

各参数具体含义：

-it: 交互式操作和终端

-p 6080:80: 开放网络端口映射

-p 5900:5900 -e VNC_PASSWORD=mypassword: 开放 VNC 端口映射，并设置访问密码

zhaowang/gazebo7.16: 镜像名

按“Ctrl+C”退出并关闭容器。

查看所有（包括正在运行和已经停止的）容器及其相关信息：

`docker ps -a`

查看正在运行容器及其相关信息

`docker ps`

```
C:\Users\75434>docker run -it -p 6080:80 -p 5900:5900 -e VNC_PASSWORD=0 zhaowang/gazebo7.16
stored passwd in /lib/systemd/7/dist-packages/supervisor/options.py:297: UserWarning: Supervisor is running as root and it is searching for its con
ms (including its current working directory), you probably want to specify a "-c" argument specifying an absolute path to a configurat
Supervisord is running as root and it is searching
2022-01-20 13:22:49,438 CRIT Supervisor running as root (no user in config file)
2022-01-20 13:22:49,439 WARN Included extra file "/etc/supervisor/conf.d/supervisord.conf" during parsing
2022-01-20 13:22:50,936 INFO RPC interface 'supervisor' initialized
2022-01-20 13:22:50,936 CRIT Server 'unix_http_server' running without any HTTP authentication checking
2022-01-20 13:22:50,936 INFO supervisord started with pid 22
2022-01-20 13:22:52,279 INFO spawned: 'xvfb' with pid 26
2022-01-20 13:22:59,022 INFO spawned: 'pcmanfm' with pid 27
2022-01-20 13:22:59,082 INFO spawned: 'lxpanel' with pid 28
2022-01-20 13:22:59,181 INFO spawned: 'lxsession' with pid 29
2022-01-20 13:22:59,229 INFO spawned: 'x11vnc' with pid 30
2022-01-20 13:23:00,082 INFO spawned: 'novnc' with pid 31
2022-01-20 13:23:03,677 INFO success: xvfb entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2022-01-20 13:23:03,679 INFO success: lxpanel entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2022-01-20 13:23:03,680 INFO success: lxsession entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2022-01-20 13:23:03,681 INFO success: x11vnc entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2022-01-20 13:23:03,682 INFO success: novnc entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2022-01-20 13:23:06,592 INFO exited: x11vnc (exit status 1; not expected)
2022-01-20 13:23:09,731 INFO spawned: 'x11vnc' with pid 45
2022-01-20 13:23:24,174 INFO success: x11vnc entered RUNNING state, process has stayed up for > than 1 seconds (startsecs)
2022-01-20 13:24:26,467 WARN received SIGINT indicating exit request
2022-01-20 13:24:26,468 INFO waiting for novnc, pcmanfm, lxpanel, xvfb, x11vnc, lxsession to die
2022-01-20 13:24:26,481 INFO stopped: novnc (terminated by SIGQUIT)
2022-01-20 13:24:26,496 INFO stopped: x11vnc (exit status 2)
2022-01-20 13:24:26,500 INFO stopped: lxsession (terminated by SIGABRT)
2022-01-20 13:24:26,529 INFO stopped: lxpanel (terminated by SIGQUIT)
2022-01-20 13:24:26,566 INFO stopped: pcmanfm (terminated by SIGQUIT)
2022-01-20 13:24:26,624 INFO stopped: xvfb (exit status 1)

C:\Users\75434>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED             STATUS             PORTS          NAMES
C:\Users\75434>docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED             STATUS             PORTS          NAMES
44897722d67   zhaowang/gazebo7.16  /startup.sh             4 minutes ago      Exited (0) 38 seconds ago
C:\Users\75434>
```

Figure 17: run

开启容器:

`docker start [NAMES]`

以交互终端形式进入容器:

`docker exec -it [NAMES] /bin/bash`

退出终端交互: (注意, 此时容器仍在运行中)

`exit`

```
C:\Users\75434>docker start upbeat_jackson
upbeat_jackson

C:\Users\75434>docker exec -it upbeat_jackson /bin/bash
root@448977722dd7:/root# exit
exit
```

Figure 18: start

一些较有用的 Docker 命令 (其他命令可参考[docker 菜鸟教程](#)):

关闭容器: `docker stop [CONTAINER ID]`

删除容器: `docker rm [CONTAINER ID]/[NAMES]`

删除镜像: `docker rmi [IMAGE ID]/[REPOSTORY:TAG]`

2. 可视化

可视化前先确保容器处于运行状态, 可以输入一下命令确认: `docker ps`

```
C:\Users\75434>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS
448977722dd7   zehacwang/gazebo7.16  "/startup.sh"          43 minutes ago Up About a minute  0.0.0.0:5900->5900/tcp, 0.0.0.0:6080->80/tcp   upbeat_jackson
```

Figure 19: ps

可以有以下两种访问方式：

(1) 网页访问

打开浏览器，输入网址`localhost:6080`，输入创建容器时设置的密码，可以看见 ubuntu 桌面。

打开终端，输入`roscore`；再开一个终端，输入`roslaunch rviz rviz`，可以看到 rviz 界面。

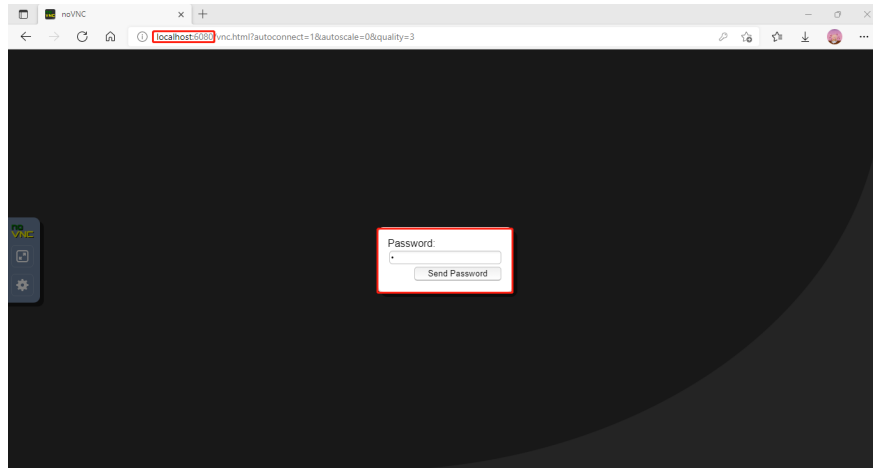


Figure 20: no vnc

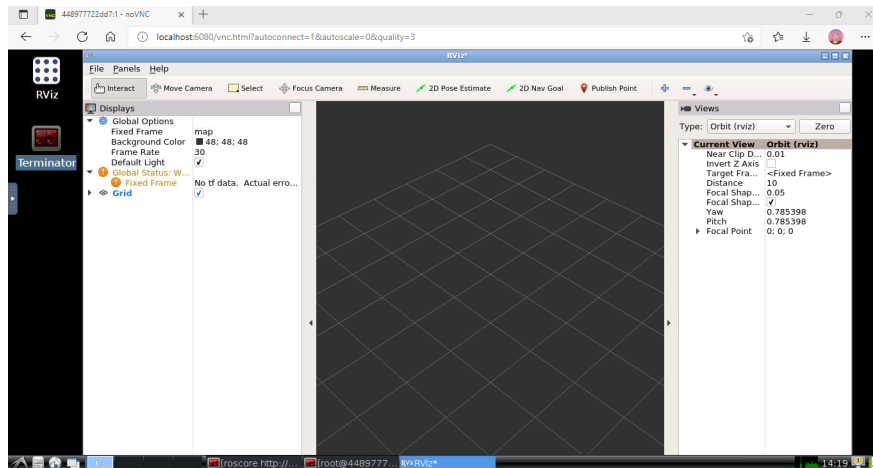


Figure 21: no vnc: rviz

(2) VNC Viewer 访问

官网下载 Windows VNC Viewer 安装包，一路安装完成，桌面出现“VNC Viewer”图标。点击“VNC Viewer 图标”，在【Connect】一栏输入：127.0.0.1:5900，会弹出输入密码界面，输入后可以看见 ubuntu 桌面。

打开终端，输入 `roscore`；再开一个终端，输入 `roslaunch gazebo_run gazebo`，可以看到 gazebo 界面，并进行后续操作。

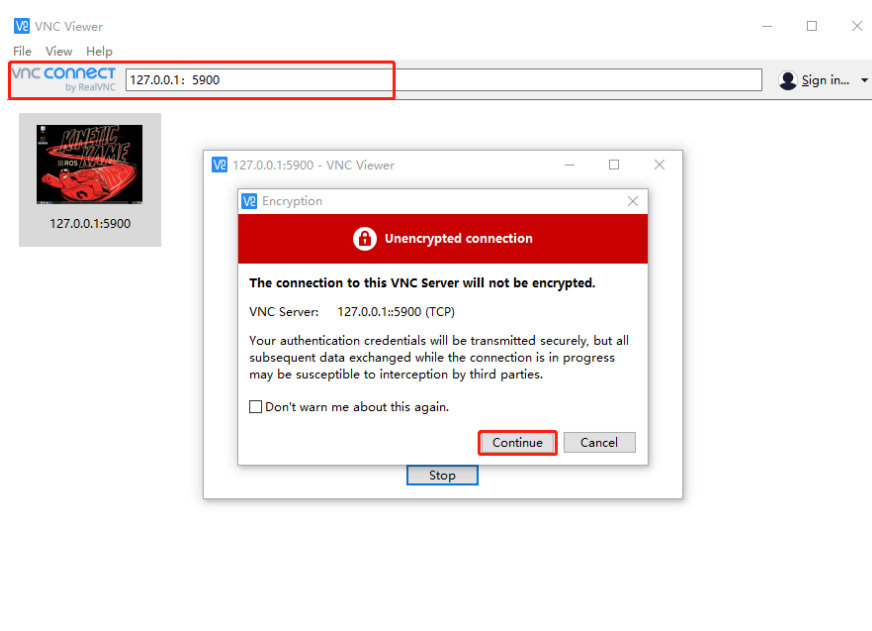


Figure 22: vnc

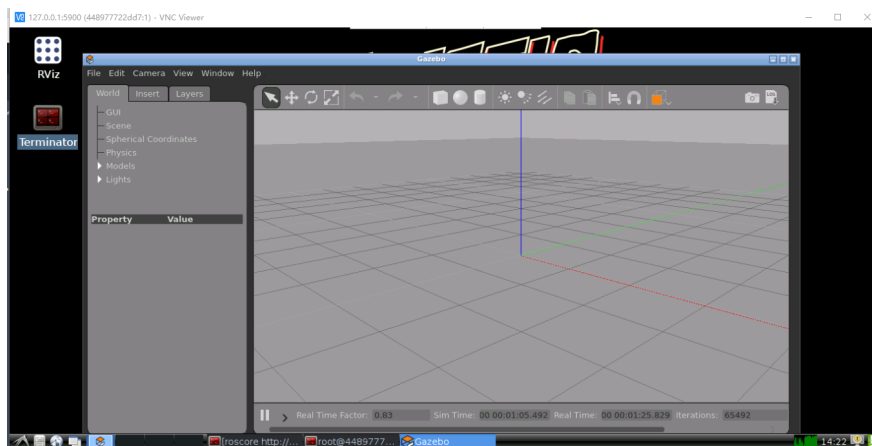


Figure 23: vnc: gazebo

Windows 家庭版安装步骤

由于 Windows 家庭中文版不自带 Hyper-V，所以在第 2 步中需要自己添加 Hyper-V。其余步骤与专业版一样。

【添加 Hyper-V 方法】:

1. 桌面新建一个 Hyper-V.bat 文件, 如图:

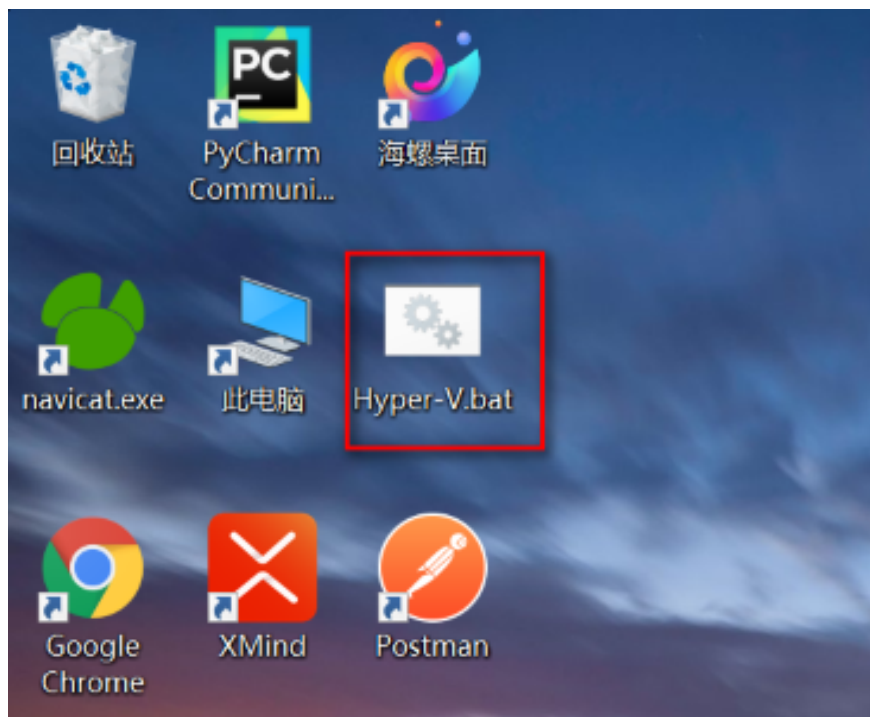


Figure 24: Hyper-V.bat

2. 将以下内容拷贝到文件中:

```
pushd "%~dp0"
dir /b %SystemRoot%\servicing\Packages\*Hyper-V*.mum >hyper-v.txt
for /f %i in ('findstr /i . hyper-v.txt 2^>nul') do dism /online /norestart /add-package:
del hyper-v.txt
Dism /online /enable-feature /featurename:Microsoft-Hyper-V-All /LimitAccess /ALL
```

3. 鼠标右键选中“以管理员身份运行”，窗口运行执行代码，直到运行结束，显示提示是否重启，输入 Y，重启电脑，如图:

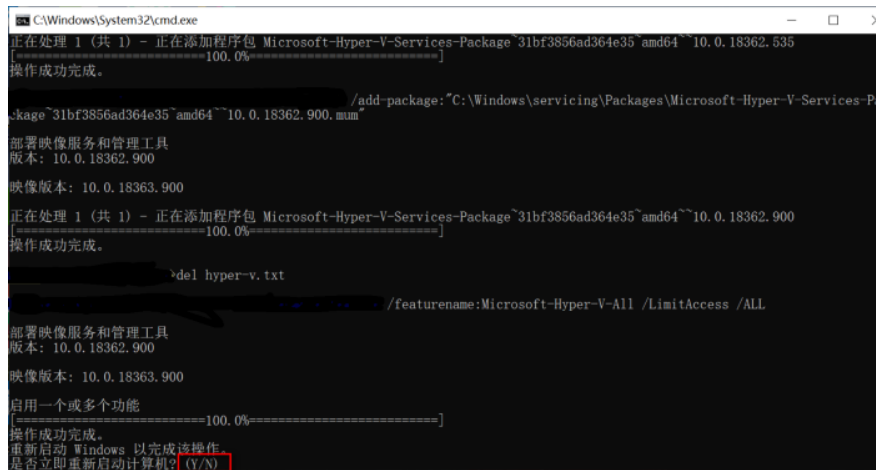


Figure 25: Hyper-V cmd

4. 重启完毕，打开控制面板-> 程序-> 程序和功能，点击“启用和关闭 Windows 功能”，弹出窗口，可看到 Hyper-V 已添加，如图：

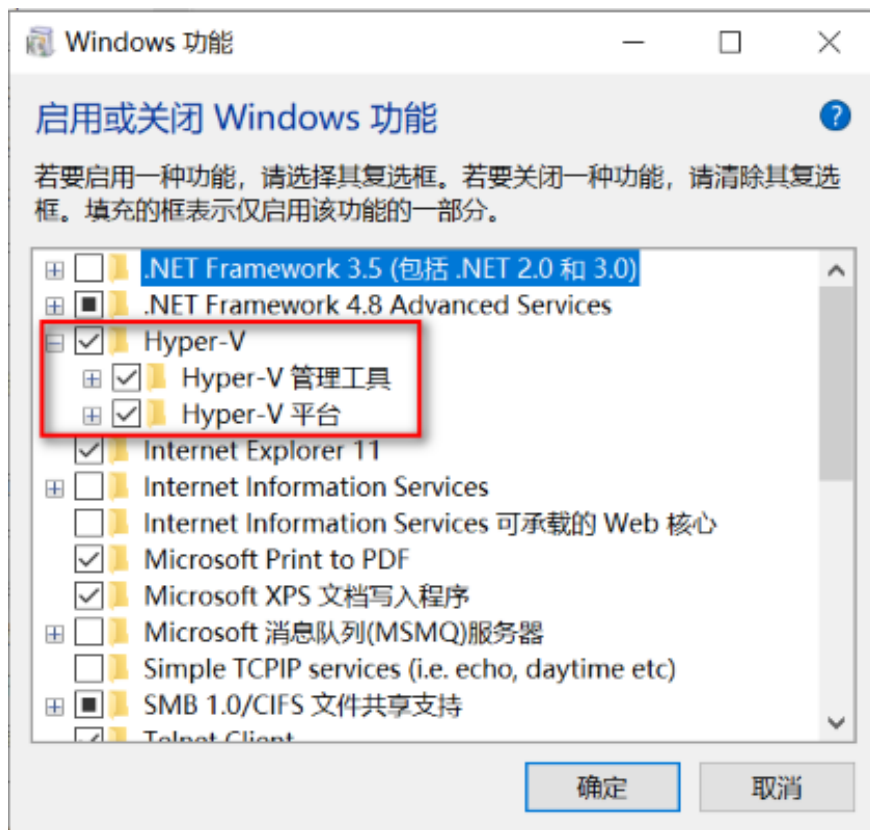
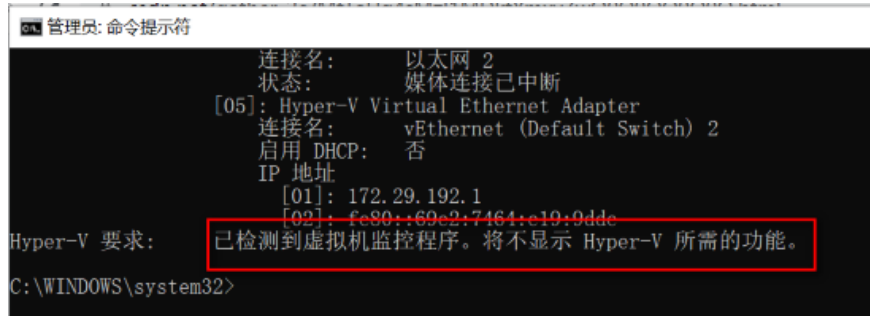


Figure 26: Hyper-V windows

5. 另外也可以以管理员身份运行 cmd，输入 systeminfo，若显示截图标红字段，表示 Hyper-V 已启用，如图：



```
管理员: 命令提示符
连接名: 以太网 2
状态: 媒体连接已中断
[05]: Hyper-V Virtual Ethernet Adapter
连接名: vEthernet (Default Switch) 2
启用 DHCP: 否
IP 地址
[01]: 172.29.192.1
[02]: fe80::69e2:7464:e19:9dde
Hyper-V 要求: 已检测到虚拟机监控程序。将不显示 Hyper-V 所需的功能。
C:\WINDOWS\system32>
```

Figure 27: Hyper-V systeminfo

4.4.2 Ubuntu 环境下，安装 ROS Docker:

一、安装 Docker

先决条件: 64 位版本 Ubuntu + Ubuntu 内核的最小版本不低于 3.10。

查看内核版本方法:

`uname -r`

安装方法有三种: 详见 [Docker 官方 Ubuntu 安装文档](#)

在这里可以采用第三种，也是最简便方式，使用官方脚本自动安装，安装命令如下:

`curl -fsSL https://get.docker.com | bash -s docker --mirror Aliyun`

二、配置 ROS Docker 以及可视化可参考上面教程。

课程后，如果需要进行项目开发，需要硬盘安装时，需要先安装 ROS 软件源，添加密钥，然后即可安装 ROS，安装完成后需要进行初始化，之后设置环境变量，最后再安装一些其他依赖即可。其他操作系统或者安装方法可以参考 ROSwiki 中的安装步骤。

4.5 小海龟仿真器的运行与实践

4.5.1 小海龟仿真器

安装完成后可以使用 ROS 自带的小海龟例程进行验证。首先打开一个终端运行 roscore，启动 rosmaster；再打开一个终端启动小海龟仿真器，最后再打开一个终端启动海龟控制节点。此时可以使用方向键控制小海龟运动即可说明 ROS 安装成功。注意控制小海龟时要注意输入焦点在控制节点对应的终端中。

1. roscore 指令启动:

`roscore`

roscore 是基于 ROS 的系统的先决条件的节点和程序的集合，必须运行 roscore 才能使 ROS 节点进行通信。

2. 启动小海龟节点:

`roslaunch turtlesim turtlesim_node`

启动成功后会打开一个可视化终端，效果如图 Fig. 28。注意：其中小海龟的图样是随机的，会有所不同。

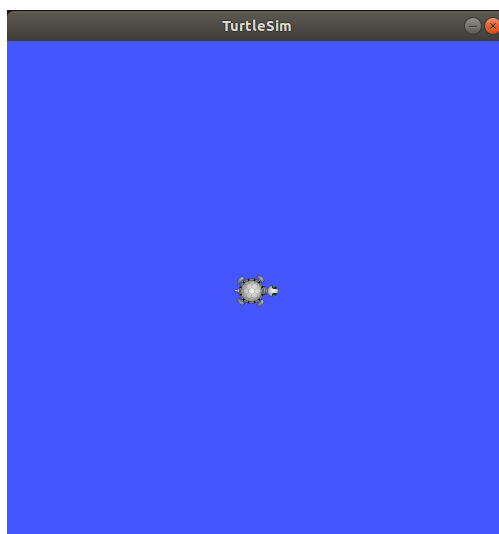


Figure 28: turtlesim_node

3. 启动小海龟键盘控制节点：

```
roslaunch turtlesim turtle_teleop_key
```

如图 Fig.29，打开键盘节点后，使用键盘的方向键即可控制小海龟向对应方向运动。注意，要保证输入焦点在 turtle_teleop_key 的终端才能正常使用。

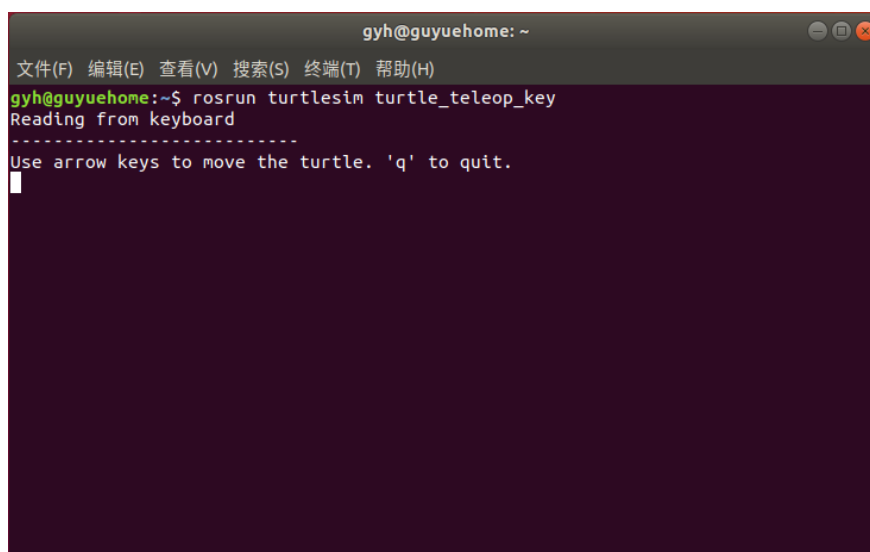


Figure 29: turtle_teleop_key

4.5.2 查看 ROS 计算图

1. rqt_graph 提供了一个 GUI 插件，用于可视化 ROS 计算图。可以使用 rqt_graph 命令打开可视化的计算图：

rqt_graph

2. 打开后即可看到如图 Fig.30所示的可视化弹窗。其中显示了当前所开启的全部节点，包括

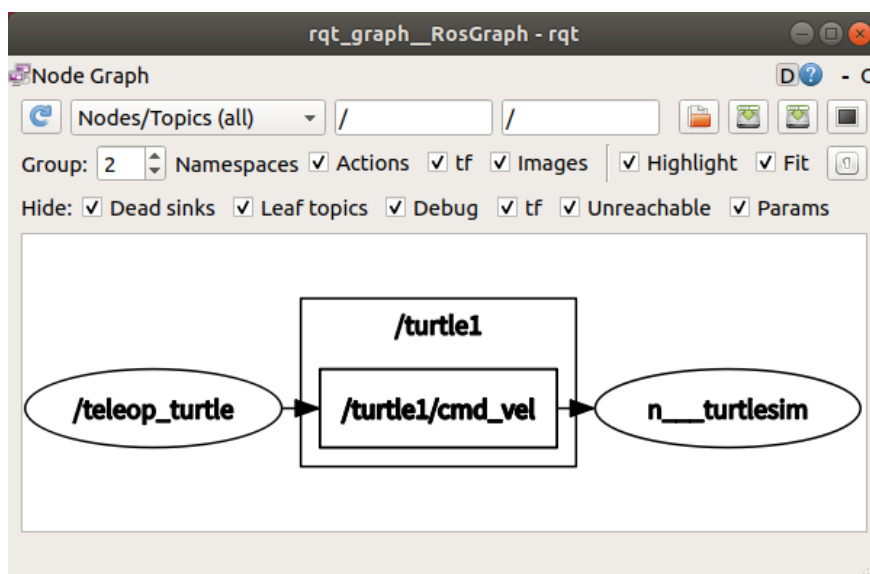


Figure 30: rqt_graph

turtlesim 和 turtle_teleop_key 及其对应的话题。

4.5.3 ROS 的常用命令

rostopic 是一个命令行工具，用于显示有关 ROS 节点的调试信息，包括发布，订阅和连接。

1. 其中 rostopic info 可以显示节点的相关信息，包括发布和订阅：

```
$ rostopic info /turtlesim
```

```
Node [/turtlesim]
Publications:
 * /rosout [roscpp_msgs/Log]
 * /turtle1/color_sensor [turtlesim/Color]
 * /turtle1/pose [turtlesim/Pose]
Subscriptions:
 * /turtle1/cmd_vel [unknown type]
Services:
 * /clear
 * /kill
 * /reset
```

```
* /spawn
* /turtle1/set_pen
* /turtle1/teleport_absolute
* /turtle1/teleport_relative
* /turtlesim/get_loggers
* /turtlesim/set_logger_level
contacting node http://192.168.3.154:45825/ ...
Pid: 2406
Connections:
* topic: /rosout
  * to: /rosout
  * direction: outbound (36671 - 192.168.3.154:35814) [24]
  * transport: TCPROS
```

2. `rostopic list` 可以显示当前节点的列表:

```
$ rostopic list
/rosout
/teleop_turtle
/turtlesim
```

3. `rostopic` 包含 `rostopic` 命令行工具, 用于显示有关 ROS 话题的调试信息, 包括发布者, 订阅者, 发布率和 ROS 消息。`rostopic list` 可以显示当前话题列表:

```
$ rostopic list
/rosout
/rosout_agg
/turtle1/cmd_vel
/turtle1/color_sensor
/turtle1/pose
```

4. `rostopic info` 可以打印话题有关的信息:

```
$ rostopic info /turtle1/cmd_vel
Type: geometry_msgs/Twist
```

Publishers:

```
* /teleop_turtle (http://192.168.3.224:42663/)
```

Subscribers:

```
* /turtlesim (http://192.168.3.224:40187/)
```

5. `rostopic echo` 可以显示话题发布的消息。例如查看小海龟的位姿数据:

```
$ rostopic echo /turtle1/pose
x: 5.544444561
y: 5.544444561
```



```
theta: 0.0
linear_velocity: 0.0
angular_velocity: 0.0
```

6. `rostopic pub` 可以发布话题数据，可以通过此命令发布小海龟的速度消息，使小海龟进行圆周运动：

```
$ rostopic pub /turtle1/cmd_vel geometry_msgs/Twist
"linear:
x:10.0
y:0.0
z:0.0
angular:
x:0.0
y:0.0
z:10.0"
```

其中可以增加 `-r` 参数，表示速率，默认为 10hz。

`rosservice` 包含用于列出和查询 ROS 服务的命令行工具，它包含一个 Python 库，用于检索有关服务的信息并动态调用它们。

7. `rosservice list` 显示活动中的服务的信息：

```
$ rosservice list
/clear
/kill
/reset
/rosout/get_loggers
/rosout/set_logger_level
/spawn
/teleop_turtle/get_loggers
/teleop_turtle/set_logger_level
/turtle1/set_pen
/turtle1/teleport_absolute
/turtle1/teleport_relative
/turtlesim/get_loggers
/turtlesim/set_logger_level
```

8. `rosservice info` 可以打印服务有关的信息：

```
$ rosservice info /turtle1/set_pen
Node: /turtlesim
URI: rosrpc://nvidia:35261
Type: turtlesim/SetPen
Args: r g b width off
```

9. `rosservice call` 可以用输入的参数调用服务：

```
$ rosservice call /turtle1/set_pen "r:255,g:0,b:0,width:5,'off':0"
```

课堂示例是请求/turtle1/set_pen服务的命令。所使用的“255 0 0 5 0”是对应于用于/turtle1/set_pen服务的参数 (r, g, b, width, off) 的值。红色的 r 的最大值是 255，因为 g 和 b 都是 0，所以笔的颜色是红色的。width 设置为 5，off 为 0 (假)。

5 习题

1. 如 Fig. 31所示，假设二维平面上有一个质量为 m 的质量块。该质量块的位置为 $[x_1^T, x_2^T]^T$ ，且受到了 $F = [u_1^T, u_2^T]^T$ 的外加作用力。假设我们能观测到其在 x_1 和 x_2 方向上的速度。

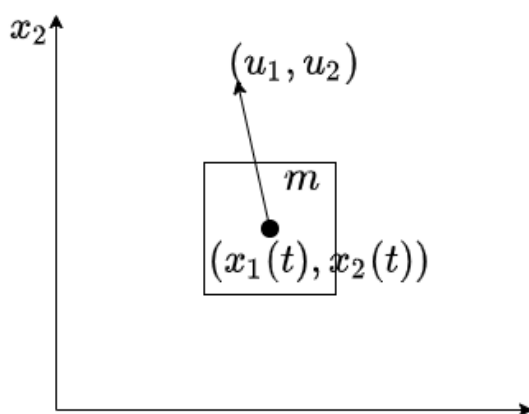


Figure 31: 二维平面质量块

- (a) 列出该系统的连续时间状态空间模型
 - (b) 若采样周期为 Δt ，并假设外加作用力在任意采样周期内 $[k\Delta t, (k+1)\Delta t), \forall k \in \mathbb{N}$ 保持不变，列出该系统离散时间的状态空间模型。
2. 若 $X \in \mathbb{R}^n, X \sim \mathcal{N}(m, P)$ ，证明 $\mathbb{E}[X] = m, \text{cov}[X] = P$ 。若 $AX + b = Y \in \mathbb{R}^m, Y \sim \mathcal{N}(0, I)$ ，试求 $A \in \mathbb{R}^{n \times m}, b \in \mathbb{R}^n$ 。
 3. 若 x 服从一元高斯分布， $x \sim \mathcal{N}(m, \sigma^2)$ 。试求 $\mathbb{E}[x], \mathbb{E}[x^2], \mathbb{E}[x^3], \mathbb{E}[x^4]$ 。
 4. 若 X, Y 的联合分布的概率密度函数为

$$p_{X,Y}(x, y) \begin{cases} \frac{6}{7}(x+y)^2, & \text{若 } 0 \leq x \leq 1, 0 \leq y \leq 1 \\ 0, & \text{其他} \end{cases}$$

试求 $[X^T, Y^T]^T$ 的期望和协方差。

5. 假设 v, e 为随机向量，且服从期望为 0 的联合高斯分布。假设 e 的协方差矩阵严格正定。证明存在一个唯一的矩阵 B ，使得 $v = Be + w$ ，其中 e 和 w 相互独立。
6. 安装预先配置好的 docker image。

7. 在 ROS 上实现小海龟编程

- (a) 使用 `topic echo` 指令打印小海龟的位姿消息，并使用键盘控制改变其位姿信息；
- (b) 使用 `rostopic pub` 指令控制小海龟绘画半径为 3 单位长度的圆；
- (c) 使用 `rosservice call` 指令将小海龟的轨迹改为绿色，宽度为 3。