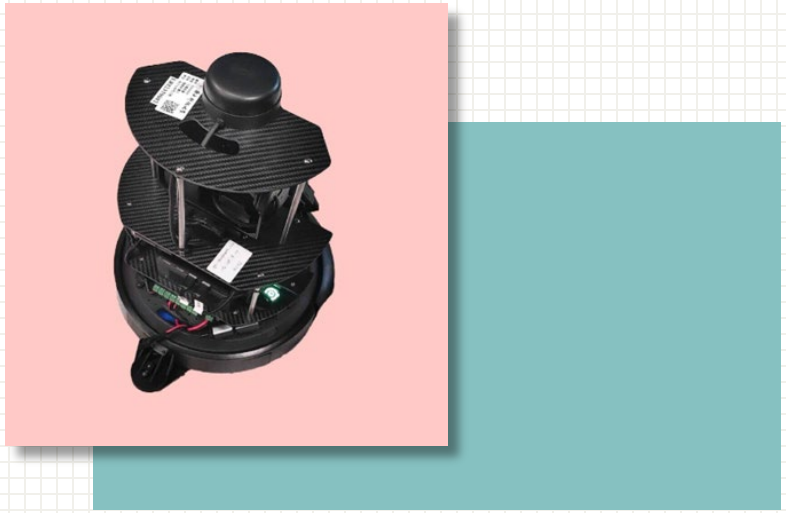
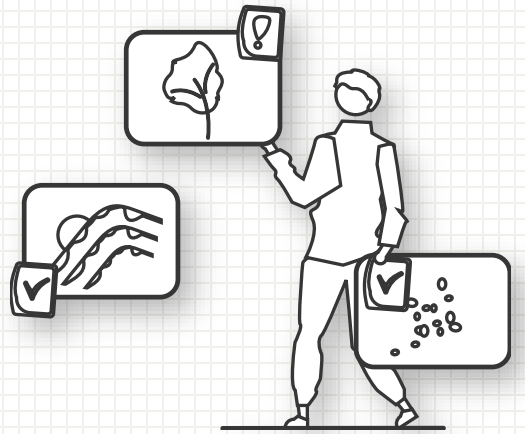
EE211 FINAL

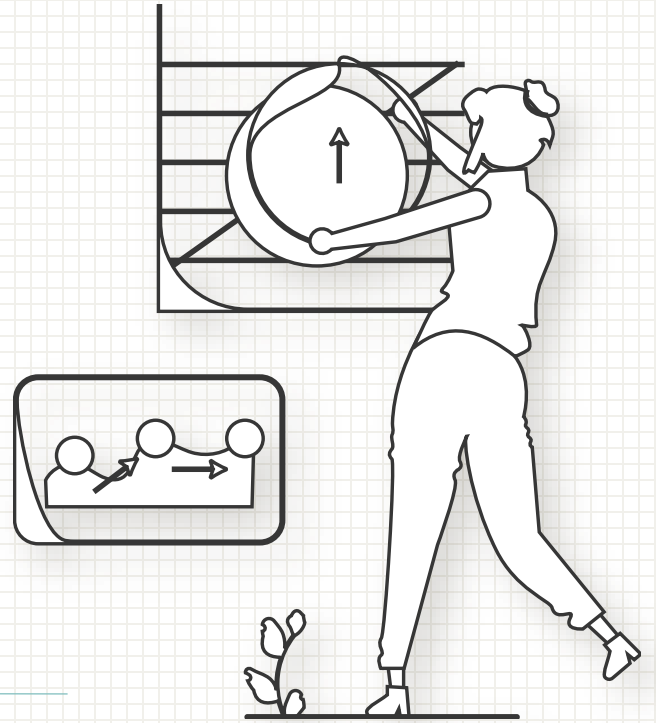
机器人感知与智能项目报告

汇报人：***，***，***

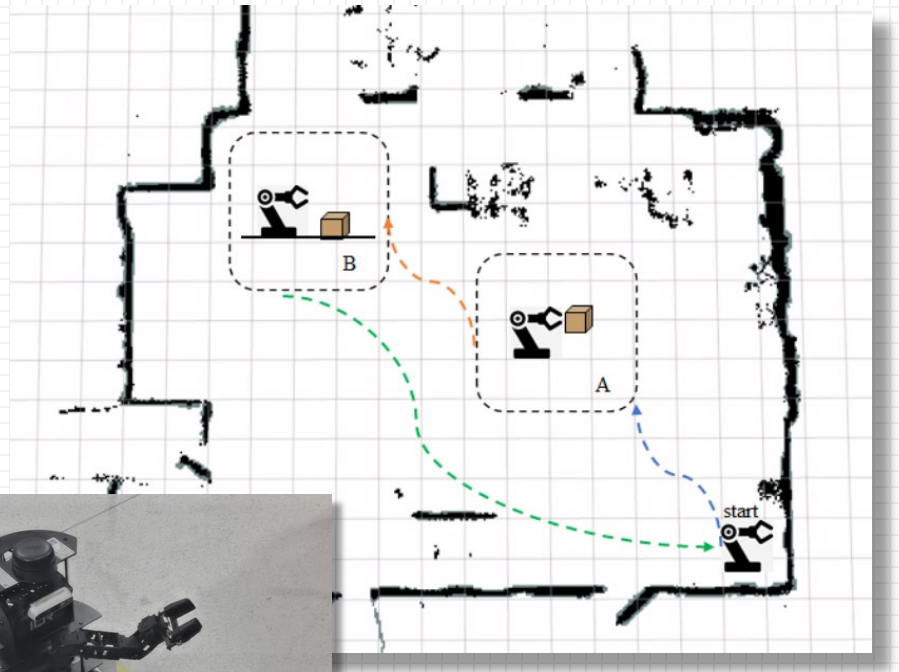




目录



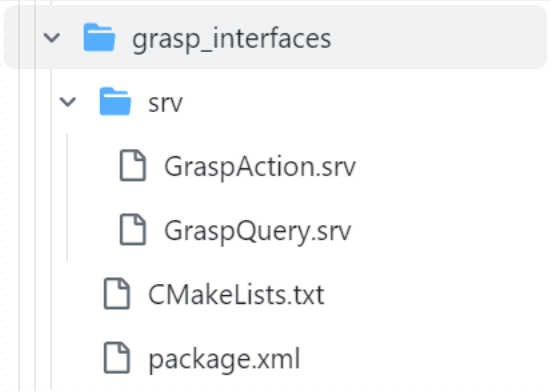
项目简介

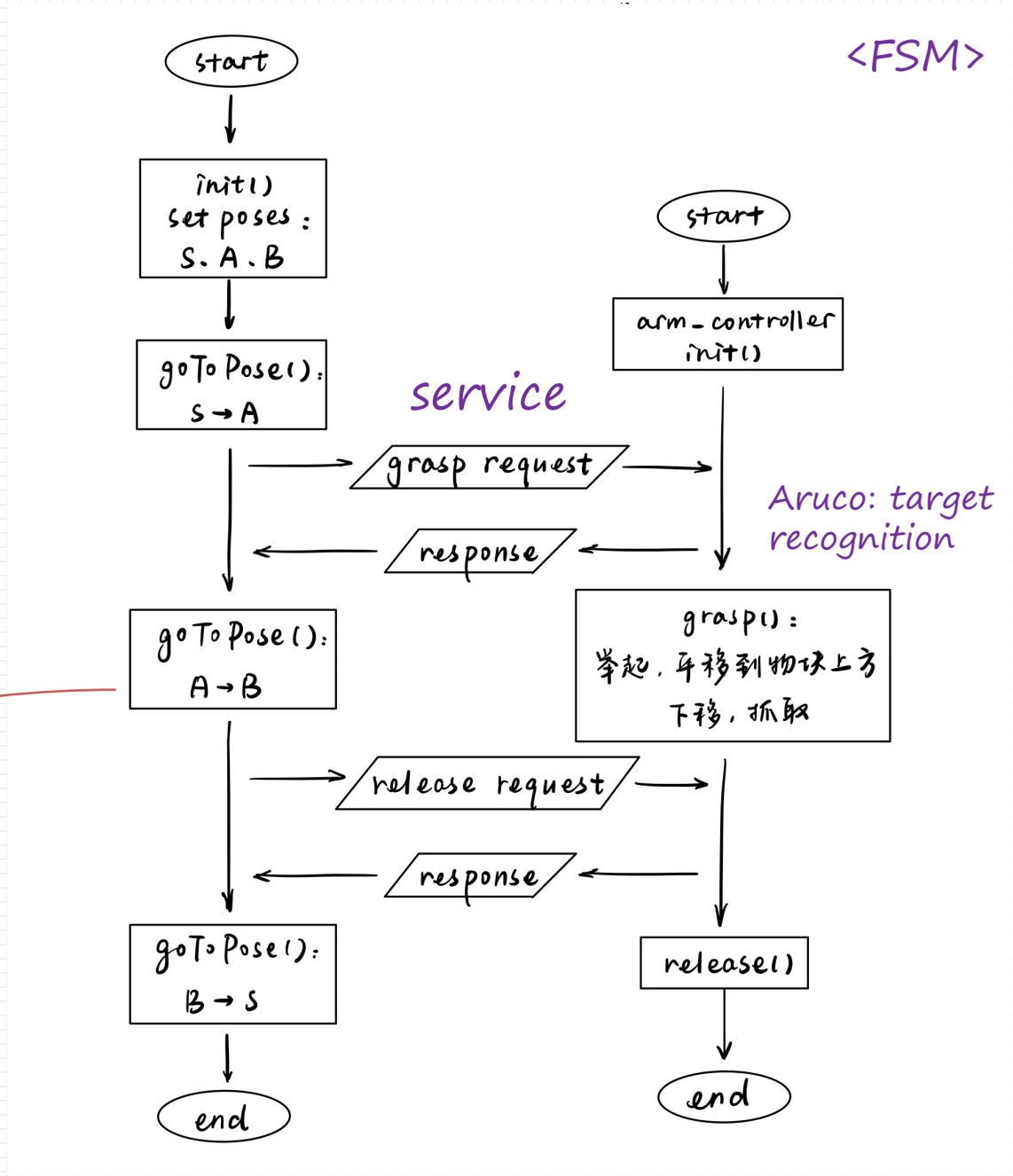


本项目基于ROS2平台，
通过机器人技术实现以下核心任务：

精确导航：机器人根据地图实现导航至目标地点。
机械控制：识别目标位姿并控制机械臂进行抓取。
路径规划：实现自主避障并完成所有模块可视化。
附加挑战：自设计规划器与随机放置物体的抓取。

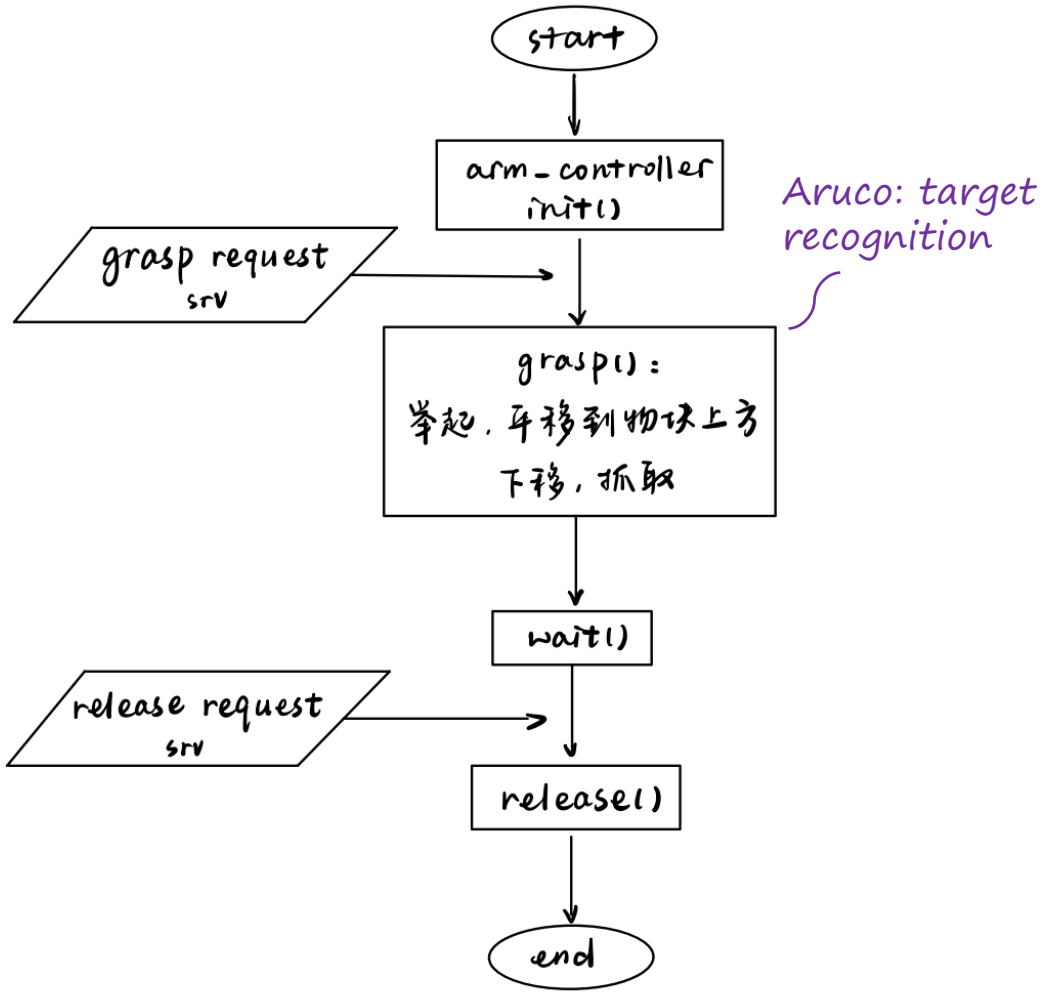
技术阐述
Overview



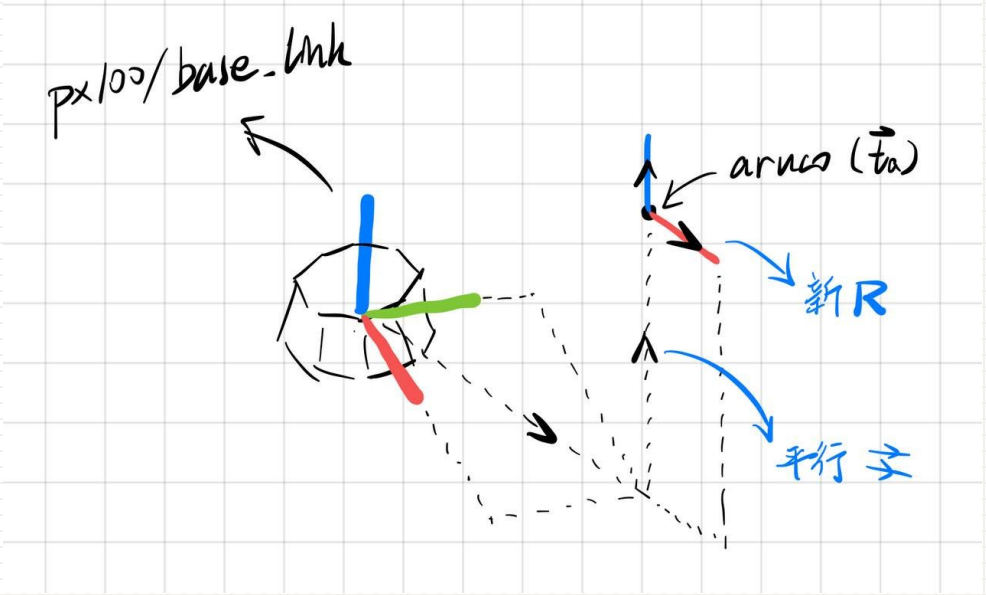


Could be overrided
by nav2 plugins

技术阐述
Arm Controller

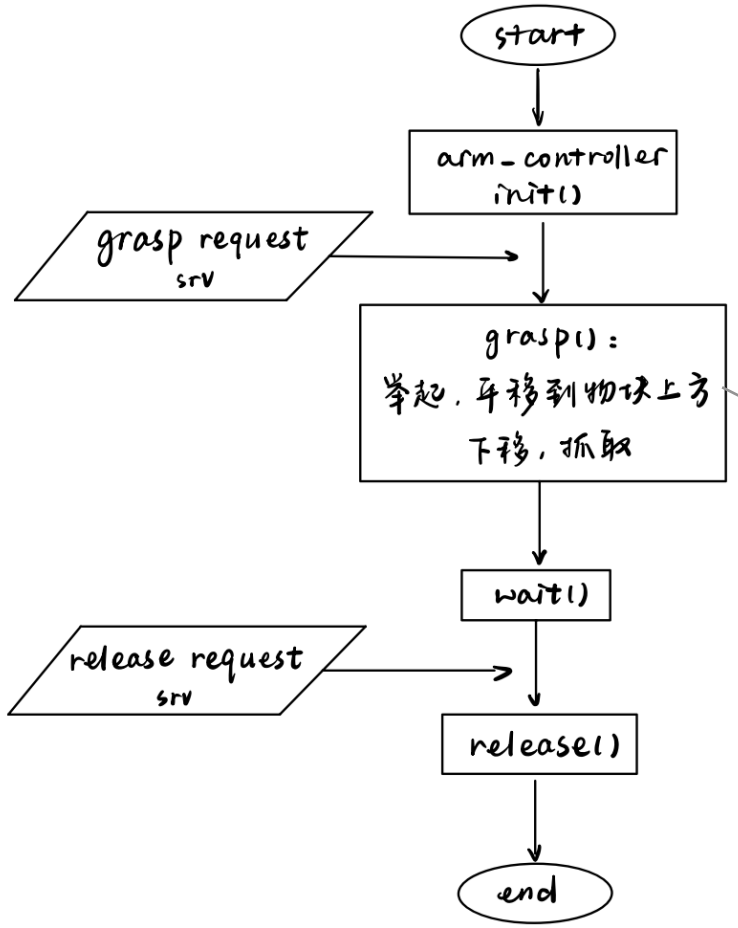


方向信息




```python
def substitute_R(matrix):
    dx, dy, dz = matrix[0, 3], matrix[1, 3], matrix[2, 3]
    d = math.sqrt(dx * dx + dy * dy)
    nx, ny = dx / d, dy / d
    return np.array([
        [nx, -ny, 0, dx],
        [ny,  nx, 0, dy],
        [ 0,   0, 1, dz],
        [ 0,   0, 0,  1]
    ])
```

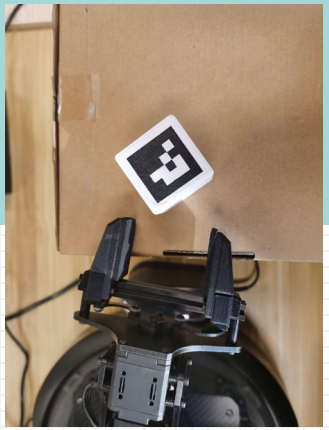
技术阐述
Arm Controller



Aruco: target recognition

```python
def compute_yaw_inclination(T_block_center):
    dx, dy = T_block_center[0, 3], T_block_center[1, 3]
    theta = math.atan2(dy, dx)

    x = T_block_center[:3, 0]
    y = T_block_center[:3, 1]
    z = T_block_center[:3, 2]
    zmin_axis = x
    if abs(y[2]) < abs(zmin_axis[2]):
        zmin_axis = y
    if abs(z[2]) < abs(zmin_axis[2]):
        zmin_axis = z
    beta = math.atan2(zmin_axis[1], zmin_axis[0])

    yaw_relative = beta - theta
    yaw_regular = (yaw_relative - np.pi / 4) % (np.pi / 2) - np.pi / 4
    return yaw_regular
```

抓取点设计



技术阐述
Navigator from Nav2

```python
def go_to_goal(self, goal: PoseStamped, blocking = True):
    self.goToPose(goal)
    if blocking:
        while not self.isTaskComplete():
            feedback = self.getFeedback()
        return self.getResult()
    else:
        return None
```

* 以及 nav2 参数调整：
    - global/local_costmap
    - planner_server
    - controller_server
    - …

技术阐述
Customized Navigator

Global Planner (as a plugin)

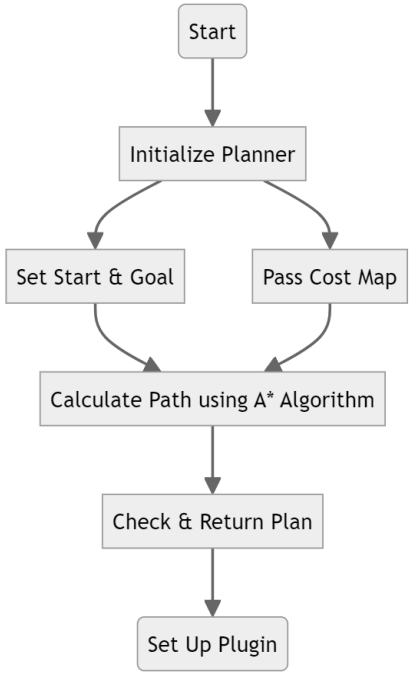
↳ A* Planner

Local Controller

↳ Trajectory Feedback Control

Astar (MyPlanner)

```
class AStar: public nav2_core::GlobalPlanner {
public:
        AStar();
        ~AStar();

        void configure(
                const rclcpp_lifecycle::LifecycleNode::WeakPtr & parent,
                std::string name,
                std::shared_ptr<tf2_ros::Buffer> tf,
                std::shared_ptr<nav2_costmap_2d::Costmap2DROS> costmap_ros
        ) override;

        void cleanup() override;
        void activate() override;
        void deactivate() override;

        nav_msgs::msg::Path createPlan(
                const geometry_msgs::msg::PoseStamped & start,
                const geometry_msgs::msg::PoseStamped & goal
        ) override;
```



Trajectory Feedback (MyController)

```
8     alpha = -theta + atan2 (y, x);
9     beta = -theta - atan2(y, x);
10    v = k_rho * rho;
11    w = k_alpha * alpha + k_beta * beta;
12    x += cos(theta) * v * t;
13    y += sin(theta) * v * t;
14    theta += w * t;
```

$$\rho = \sqrt{\Delta x^2 + \Delta y^2}$$

$$\alpha = -\theta + \arctan\left(\frac{\Delta y}{\Delta x}\right)$$

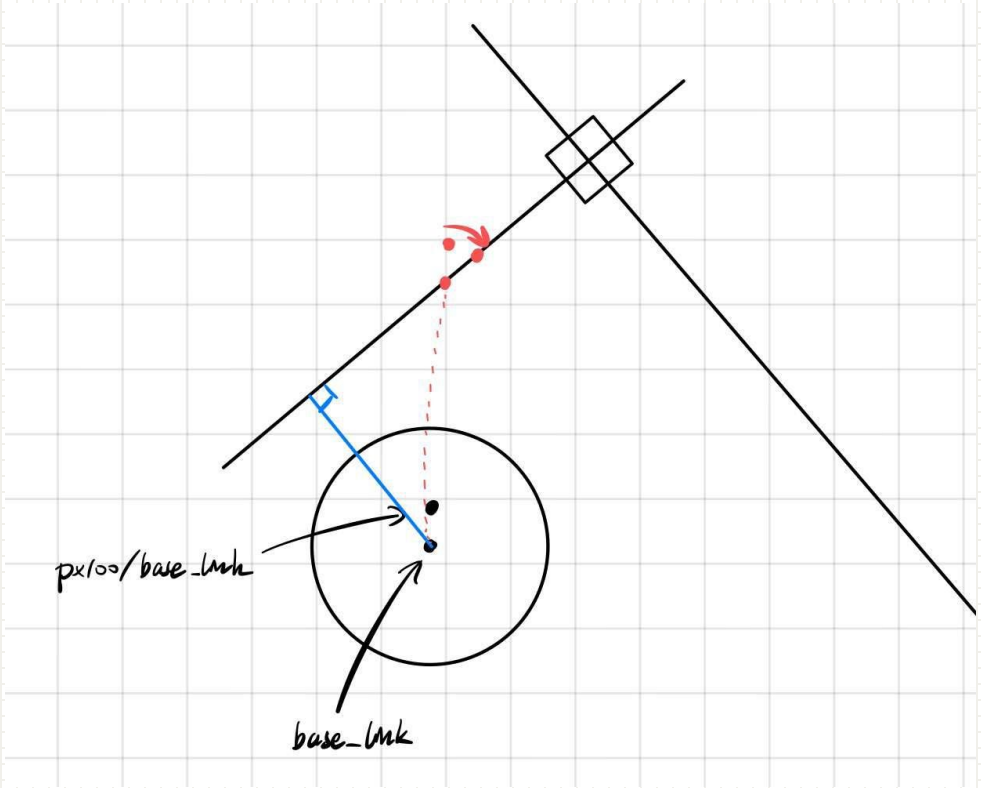
$$\beta = -\theta - \alpha$$

$$v = k_\rho \rho$$

$$w = k_\alpha \alpha + k_\beta \beta$$

技术阐述
Navigator

接近物块时的行为 (Approach I)



Q
里程计不准确，应以视野中的 Aruco 为准；
行进过程中物块可能从视野中消失；

A
可以通过行进同时驱动云台跟踪物块解决，
坐标变换通过 TF Tree 自动完成；

Q
相机重新标定过的前提下，
云台旋转影响了 Aruco 的估计坐标，
机器人的结构配置文件疑似不准；

A
修改繁琐，
采取更简单的方法（x

技术阐述
Navigator

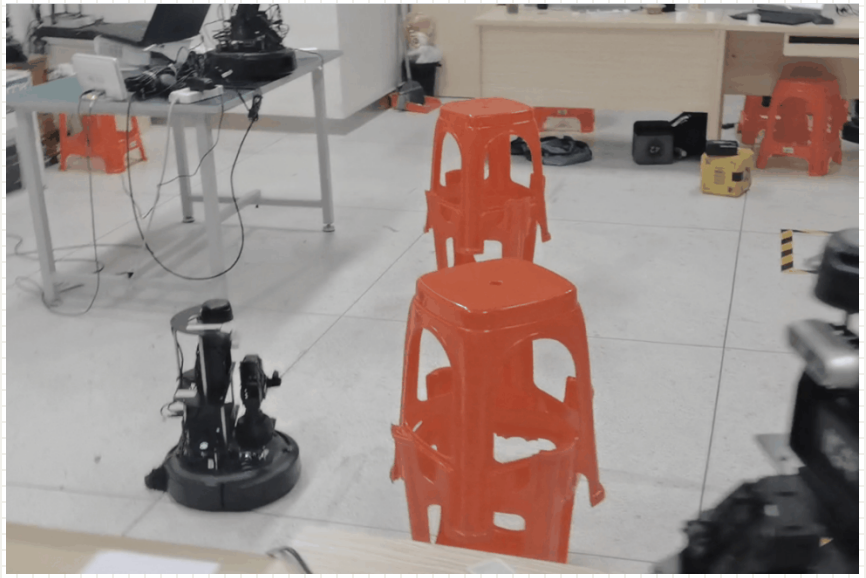
接近物块时的行为 (Approach II)

```
# Turn to block
T_0a = pose_to_matrix(block_pose.pose)
T_ba = np.dot(T_b0, T_0a)
dx, dy = T_ba[0, 3], T_ba[1, 3]

while temporary_commander.amcl_pose is None:
    rclpy.spin_once(temporary_commander)
yaw = temporary_commander.warp_angle(temporary_commander.get_current_yaw() + np.arctan2(dy, dx))
temporary_commander.turn_to_yaw(yaw)

# Move slowly
while True:
    is_solved = temporary_client.grasp_query_solved()
    if is_solved:
        temporary_commander.stop()
        break
    temporary_commander.cmd_vel(linear_x = 0.03)
time.sleep(1.5)
```

前面实现的抓取容错大，
对有解位置要求高，
但对物块所在方向要求低，
故正对着走上去基本就没问题。

grasp_query_solved() 可获取
当前机械臂解存在状态，
在慢速接近过程中不断判断，
直到进入可抓取区域，
停车开抓。

成果展示



总结

Navigation

能够实现精确导航到目标点

Controller

完成目标识别与机械臂抓取

Planner

自主实现路径规划器

VeryCute

好评如潮！