1.function[North,South,Right,Left,autoch]=look(postion_obj,stop_position)

参数：

position_obj　　目标点位置坐标

stop_position　　机器人的位置坐标

返回值：

North　北

South　南

Right　右

Left　　左

autoch 当前方向

作用：

根据目标点与机器人当前位置进行对比，确定机器人当前前进的方向，（这里默认 y 轴正方向为北），返回 North 或 South。之后再确定目标点在机器人的左方还是右方还是当前前方，返回 Right Left autoch。

2. function transl = y_transl(start_position,end_position,Delta_y)

参数：

start_position　　起始位置

end_position　停止位置

Delta_y　　水平单位移动距离

返回值：

transl　　水平单位移动的次数

作用：

在阶梯状下潜的过程中，判断单位水平前进所需要的次数。

3.function[up,down]=target_image_pose(down_transl,start_position,end_position,Delta_y)

参数：

down_transl　　垂直下潜阈值

start_position　　同上

end_position　　同上

Delta_y　　同上

返回值：

up　前进

down　下降

作用：

对比前进所需次数与下降所需次数，决定前进还是下降。实现阶梯状下潜。

4.function position = approach(start_position,end_position,an,down_transl_spd, fps,Delta_y,Delta_z,down_transl)

参数：

start_position　　同上

end_position　　同上

an　动画

down_transl_spd　　逼近过程中的步长

fps　帧率

Delta_y　　同上

Delta_z　　同上

down_transl　　同上

返回值：

position　坐滩位置

作用：

找到目标点之后的阶梯状下潜。

5. function end_position = brake(start_pose, start_v, max_deceleration, al, fps)

参数：

start_pose　开始制动时的位姿信息（包括位置与朝向）

start_v　　要制动时的瞬时速度

max_deceleration　　设置的减速度

al　　动画

fps

返回值：

end_position　　停止时的位置

作用：

在发现目标点之后制动减速直到机器人停下。

6. function orientation = calc_orientation(start_position, end_position)

参数：

start_position

end_position

返回值：

orientation　　角度

作用：

计算从（end_position(1:2)-start_position(1:2)）与 x 轴之间的角度的函数，返回该角度。

7. function cur_position = corrent(cur_position, optimal_path, max_distance)

参数：

cur_position　　当前位置

optimal_path　　理想的巡航路线

max_distance　　允许的最大偏离距离

返回值：

cur_position　　是否需要纠偏（0　1）

作用：

纠正因洋流和浪偏离的巡航路线的函数，cur_position 为当前位置，optimal_path 为理想的巡航路线，max_distance 为允许的最大偏离距离，超过此距离即开始纠偏，若需要纠偏则返回值为 1，否则为 0。

8. function position = dive(start_position, end_position, down_transl_spd, an, fps)

参数：

start_position　　起始位置

end_position　　终止位置

down_transl_spd　　步长

an　　动画

fps　　帧率

返回值：

position　水平移动之后的位置

作用：

模拟视觉识别，当检测到目标之后，根据 look（）函数的信息，判断目标在当前机器人的左右还是当前方向，而后水平左（右）移动，直到目标与机器人共线（此过程模拟视觉图像中目标出现在屏幕中轴线上),返回水平移动之后的位置。

9. function position = dive_back(start_position, end_position, down_transl_spd, an, fps)

参数：

start_position　　初始位置

end_position　　终止位置

down_transl_spd　　步长

an　　动画

fps　　帧率

返回值：

position　　终止位置

作用：

从初始位置到终止位置，并动画展示。

10. function position = float(start_position, end_position, transl_spd, an, fps,down_transl_spd,z_cruising)

参数：

start_position　　初始位置

end_position　　终止位置

transl_spd　　步长

an　　动画

fps　　帧率

down_transl_spd　　步长

z_cruising　　巡航平面高度

返回值：

position　　发现目标时的巡航位置

作用：

在机器人完成下潜抓取之后，垂直上浮到巡航面，并返回到发现目标时的巡航位置。

11. function [position, orientation, objects] = grab(position, orientation, objects,index)

参数：

position 位置

orientation 角度

objects　　目标点

index　　下标

返回值：

position 位置

orientation 角度

objects　　目标点

作用：

贪心算法初始化

12. function [position, orientation, objects] = greedy_grab(pose_start, objects, work_radius, search_radius, transl_spd, rot_spd, al, fps)

参数：

pose_start　初始位置

objects　目标点列表

work_radius　机械臂的工作半径

search_radius 能够探测的范围半径

transl_spd　步长

rot_spd　角速度

al　动画

fps　帧率

返回值：

position　执行完贪心之后的位置

orientation　执行完贪心之后的角度位姿

objects　更新过后的目标列表

作用：

在海底搜寻目标时的贪心算法，pose_start 为刚从巡航平面下降至海底平面时的位姿，objects 用于存放搜寻的目标，work_radius 为机械臂的工作半径，search_radius 为能够探测的范围半径，若在 pose_start 时检测到在机械臂工作半径之内的目标则直接抓取，能检测到但在工作半径之外的目标则需要移动至该位置进行抓取，重复上述过程直至附近无检测目标。

13. function [index, dist] = nearest_obj(position, objects)

参数：

position　当前位置

objects　目标列表

返回值：

index　最近目标的下标

dist　到最近目标的距离

作用：

在海底时用于检测最近目标点供贪心算法使用。

14. function current = noise()

返回值：

current

作用：

模拟巡航时洋流与浪的噪声函数（最简单的高斯噪声），大概率产生分三个方向（x,y,z）的小洋流，小概率产生较大的浪。

15. function orientation = rotate(start_orient, end_orient, rot_spd)

参数：

start_orient　　初始角度

end_orient　　终止角度

rot_spd　　角速度

作用：

计算从 start_orient 到 end_orient 的角度

16. function [has_object, position] = search_object(objects, pose, detect_radius, work_radius)

参数：

objects　　目标列表

pose　　位姿信息

detect_radius　　检测半径

work_radius　　工作半径

返回值：

has_object 是否找到

position　当前巡航点检测到的有效目标

作用：

巡航平面搜寻海底目标的过程，其搜索半径不会超过每条航路的间距。

17. function show_trajectory(al, traj, fps)

作用：

用于展示运动轨迹。

18. function [optimal_path_new, vels, deceleration_x, deceleration_y, deceleration_z, max_vels_x, max_vels_y, max_vels_z, t_x, t_y, t_z,v_] = traj_new(optimal_path, transl_spd, approximate_max_vels_x, approximate_max_vels_y, approximate_max_vels_z, acceleration_x, acceleration_y, acceleration_z, approximate_deceleration_x, approximate_deceleration_y, approximate_deceleration_z)

参数：

optimal_path　　路径关键点

transl_spd　　步长

approximate_max_vels_x　　x 轴方向最大速度

approximate_max_vels_y　　y 轴方向最大速度

approximate_max_vels_z　　z 轴方向最大速度

acceleration_x　　x 轴方向加速度

acceleration_y　　y 轴方向加速度

acceleration_z　　z 轴方向加速度

approximate_deceleration_x　　x 轴方向预估减速度

approximate_deceleration_y　　y 轴方向预估减速度

approximate_deceleration_z　　z 轴方向预估减速度

返回值；

optimal_path_new　　新的路径关键点

vels　　各点三方向的速度

deceleration_x　　实际的 x 轴方向减速度

deceleration_y　　实际的 y 轴方向减速度

deceleration_z　　实际的 z 轴方向减速度

max_vels_x　　实际的 x 轴方向最大速度

max_vels_y　　实际的 y 轴方向最大速度

max_vels_z　　实际的 z 轴方向最大速度

t_x　　x 上时间

t_y　　y 上时间

t_z　　z 上时间

v_　每点的所有速度信息

作用：

traj_new 是替换原有 MATLAB 机器人工具箱中规划两点运动轨迹的函数(初速度与到达时速度均为 0)，使用动力学三公式（初中物理），涉及参数较多，首先需要提供要规划的两点及步长等信息,并提供三个方向的加速度(acceleration_x, acceleration_y, acceleration_z)，预估的三个方向的减速度及最大速度（原本想设计成可人工控制加减速度且可以给出匀速时的最大速度，但设计上有失误，无法规划中间的匀速运动，目前只能实现给出加速度做匀加速运动，再根据给出的预估减速度计算出真正的减速度，再进行匀减速运动，即实际上代码中有

用的只有匀加速——匀减速部分，但代码中还存在失败的匀速部分的代码，故在输入最大速度参数时会输入一个无法达到的速度，在匀速过程开始前进入减速过程），根据步长给出两点之间的规划的各点的位置与三项速度，返回的是规划的各点、各点三项速度、实际的三项减速度、实际的最大速度及每点之间相隔的三项时间。

19.function result = within_boundary()

作用：

返回 true。