HOCHSCHULE
RAVENSBURG-WEINGARTEN
UNIVERSITY
OF APPLIED SCIENCES

AMR
FINAL PRESENTATION
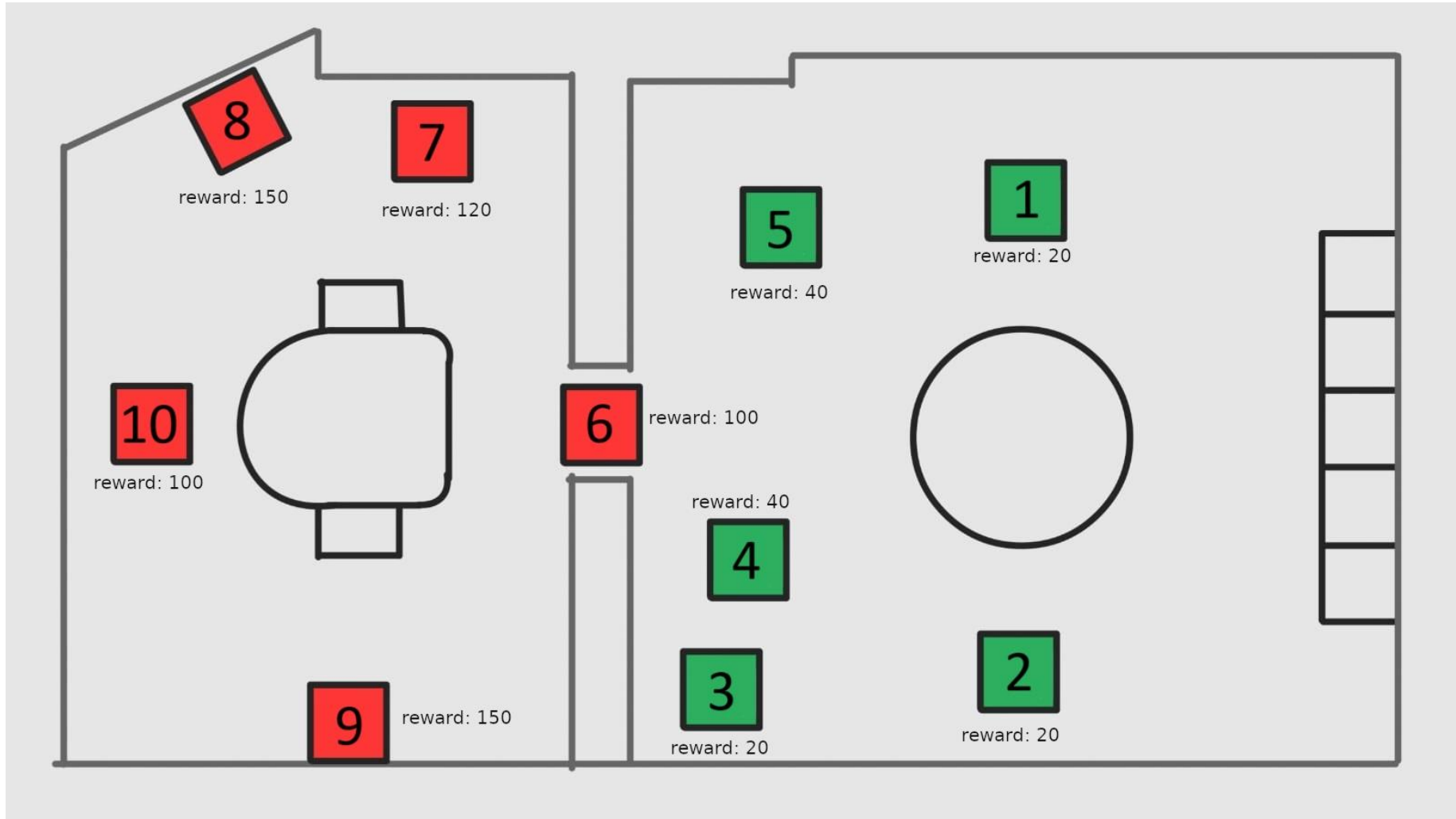
Fabian Schotte

Content

1. Task

2. My idea for the competition

3. Current situation

4. Problems

Task

reward: 150

reward: 120

reward: 20

reward: 40

reward: 100

reward: 40
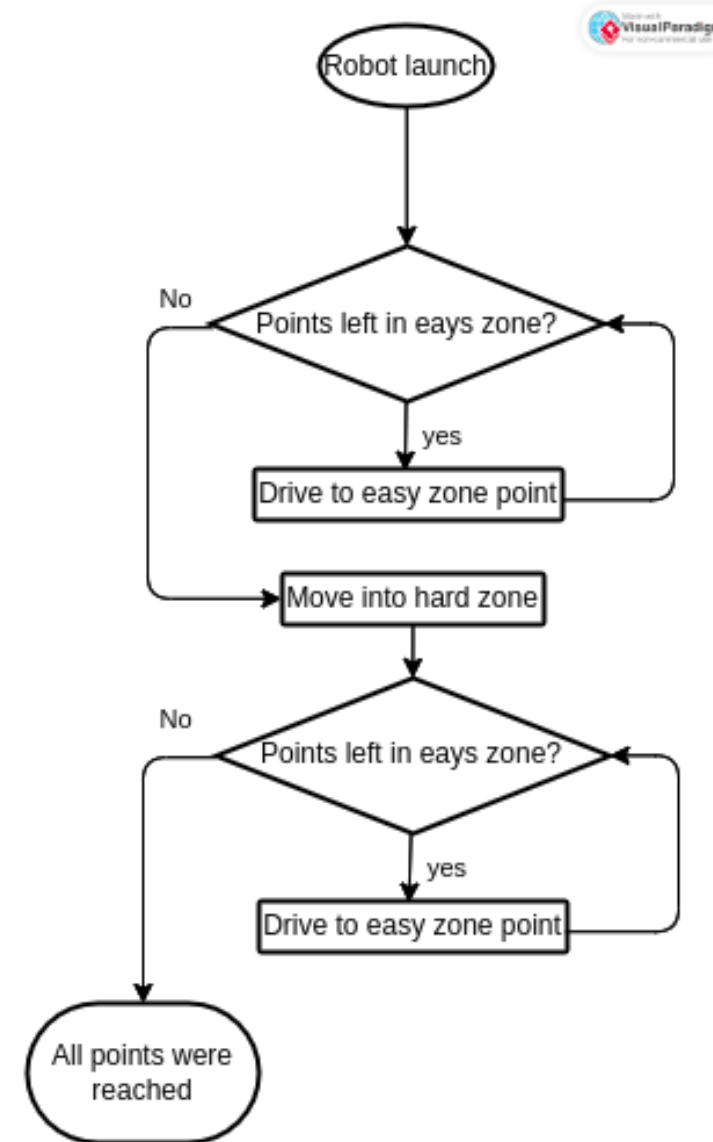
reward: 100

reward: 20

reward: 150

reward: 20

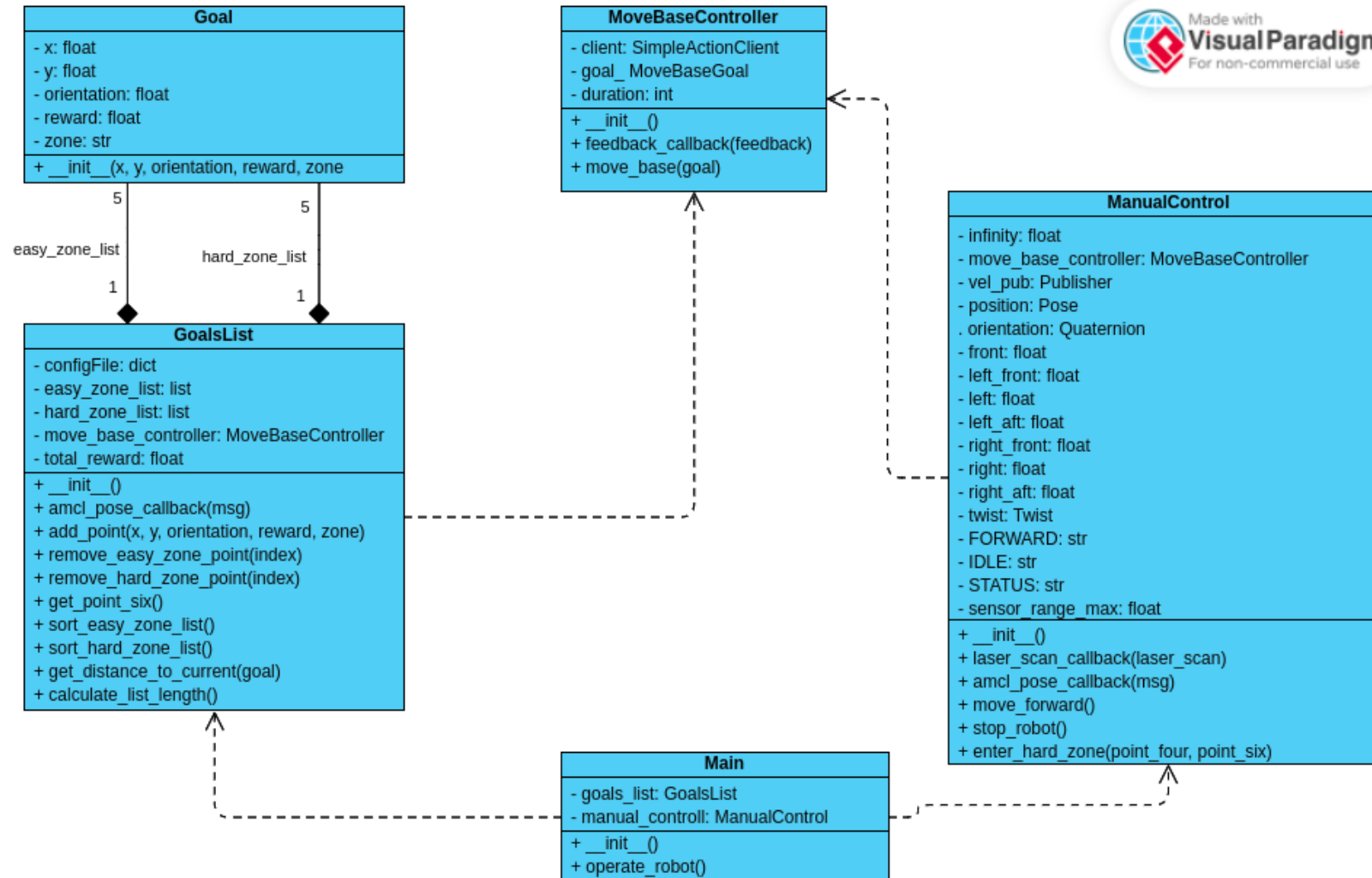# My ideo for the competition
Flow chart of the objectives

- Move around the easy zone and collect the points

- Move into the hard zone

- Move around the hard zone and collect the points

My idea for the competition
Code structure as an UML

Goal

- x: float
- y: float
- orientation: float
- reward: float
- zone: str

+ __init__(x, y, orientation, reward, zone

MoveBaseController

- client: SimpleActionClient
- goal_ MoveBaseGoal
- duration: int

+ __init__()
+ feedback_callback(feedback)
+ move_base(goal)

ManualControl

- infinity: float
- move_base_controller: MoveBaseController
- vel_pub: Publisher
- position: Pose
. orientation: Quaternion
- front: float
- left_front: float
- left: float
- left_aft: float
- right_front: float
- right: float
- right_aft: float
- twist: Twist
- FORWARD: str
- IDLE: str
- STATUS: str
- sensor_range_max: float

+ __init__()
+ laser_scan_callback(laser_scan)
+ amcl_pose_callback(msg)
+ move_forward()
+ stop_robot()
+ enter_hard_zone(point_four, point_six)

5   5

easy_zone_list   hard_zone_list

1   1

GoalsList

- configFile: dict
- easy_zone_list: list
- hard_zone_list: list
- move_base_controller: MoveBaseController
- total_reward: float

+ __init__()
+ amcl_pose_callback(msg)
+ add_point(x, y, orientation, reward, zone)
+ remove_easy_zone_point(index)
+ remove_hard_zone_point(index)
+ get_point_six()
+ sort_easy_zone_list()
+ sort_hard_zone_list()
+ get_distance_to_current(goal)
+ calculate_list_length()

Main

- goals_list: GoalsList
- manual_controll: ManualControl

+ __init__()
+ operate_robot()

Current situation

- The robot is capable of moving from point to point

- The robot detects objects in its way and moves around them

- If an goal is taken the robot moves to the next goal according to the list

- The robot has problems at the wall opening
  - It does drive towards the opening but does drive through it
  - Possible Problem: The position reading is inaccurate
  - Solution: Replacing the reading by a new accurate one

Problems
Backup strategy for navigating to the goals

- All point list are sorted after the distance to the current position

- The list will be resorted after each successfully reached point

- When an point cannot be reached, the code will continue with the next point

- After the second point is reached, the code will reattempt to reach the first point
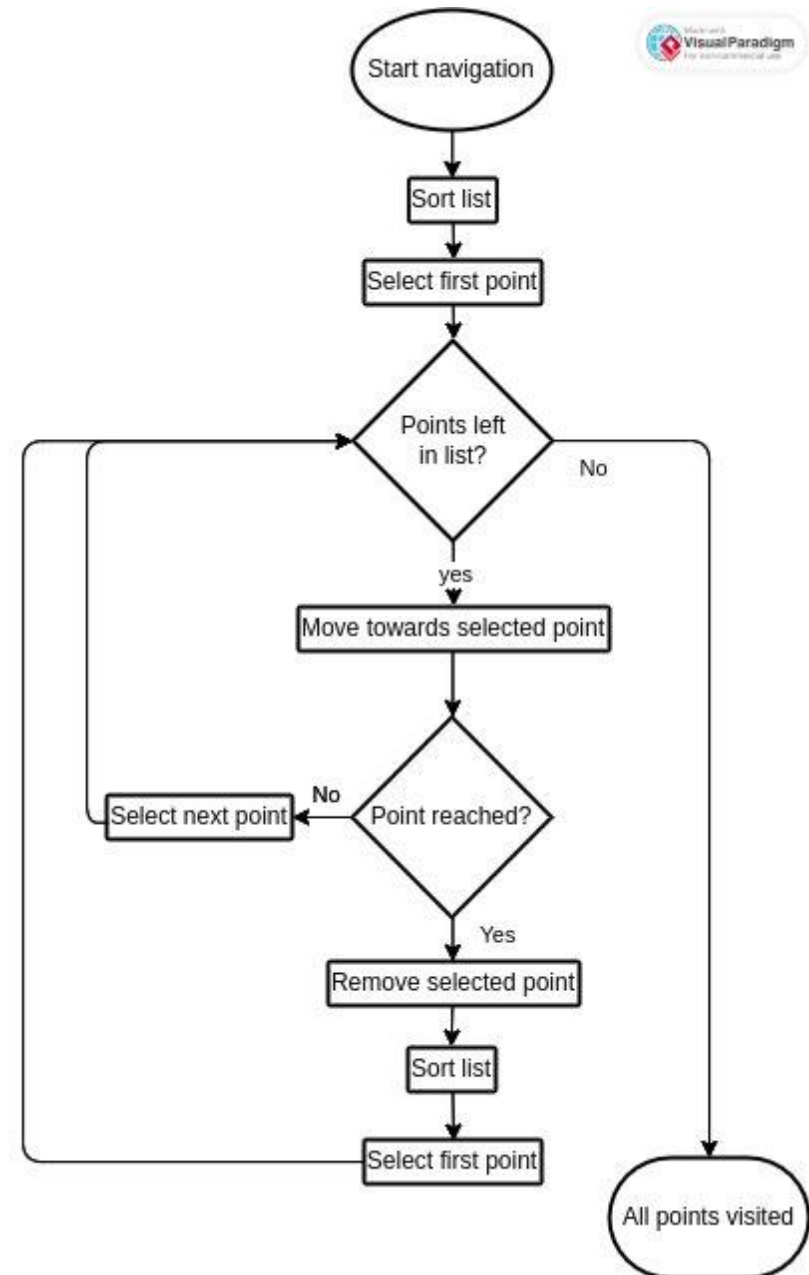
Problems
Collision Detection

- Changed the delta value in the config for slam from 0.5 to 0.01 to improve the map accuracy

- If recorded goal position was inaccurate, it was replaced by an new and accurate reading of the goal position

- Possible to increase the allowed distance to other objects because the robots are often too close to each other

Problems
Moving throught the opening in the wall