### 引言

本研究报告专注于开发和评估一种用于模拟火星探测车的控制器，采用行为基础机器人学（BBR）和进化机器人学（ER）方法。特别地，本文将深入探讨采用进化编程（EP）方法实现的ER控制器。火星探测车的主要任务是在指定区域收集土壤样本，并需在避免路径上障碍物的同时，迅速到达收集/奖励区域。本研究旨在通过模拟环境测试并验证控制器的有效性，进而为未来的火星探索任务提供理论和技术支持。

### 2. PID控制器（task 1）

**2.1 功能**

PID控制器在BBR方法中用于实现精确和平滑的导航。它调整探测车的速度和方向，以响应环境变化，如线路跟随和障碍物避免。

**2.2控制器代码中传感器的使用和功能**

**2.2.1使用的传感器及其功能**

**1.距离传感器 (ps)**

* **数量**：8个（ps0 到 ps7）
* **功能：**用于检测和避免障碍物。
* **实现方式**：通过 self.distance\_sensors[i].enable(self.time\_step) 启用每个传感器，并在 \_\_read\_distance\_sensors 方法中读取它们的值，用于判断机器人周围的障碍物并相应地调整行动。

**2.光线传感器 (ls)**

* **数量**：8个（ls0 到 ls7）
* **功能：**感应光线，用于辅助路径选择。
* **实现方式：**光线传感器通过 self.light\_sensors[i].enable(self.time\_step) 被启用，并在 \_\_read\_light\_sensors 方法中读取其值，用于辅助决定路径选择。

**3.地面传感器 (gs)**

* **数量**：3个 (gs0, gs1, gs2)
* **功能**：用于线路跟踪。
* **实现方式**：地面传感器分别为左、中、右地面红外传感器，通过 self.left\_ir.enable(self.time\_step) 等语句启用，并在 \_\_read\_ground\_sensors 方法中读取其值，用于实现线路跟踪。

**4.摄像头**

* **功能**：图像采集和分析。
* **实现方式**：通过 self.camera.enable(self.time\_step) 启用摄像头，并在 \_\_read\_camera 方法中处理摄像头捕获的图像数据，用于进一步的图像分析。

### ****2.3 控制器与PID的集成****

**2.3.1 数据读取**

控制器从地面、光线和距离传感器读取数据，以确定当前的环境状态和需求。

**2.3.2 行为决策**

根据传感器数据，控制器使用PID控制器的输出来决定如何调整探测车的行动，例如跟随线路或避免障碍物。

**2.3.3 行为执行**

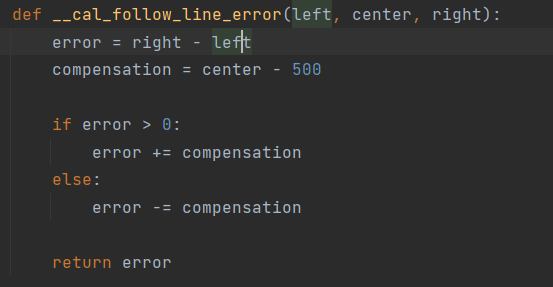
根据PID控制器的调整值，控制器调整电机的速度，以实现平滑的导航和有效的障碍物避免。

**2.4 实现**

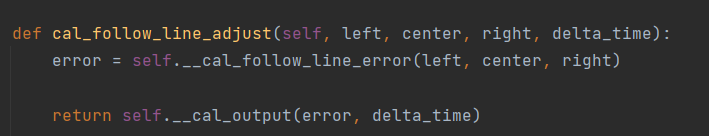
* **初始化:** 设定比例（P）、积分（I）和微分（D）的系数。



* **误差计算:** 根据左、中、右地面传感器的数据，计算探测车与预定路径的偏差。



* **调整输出:** 根据计算出的误差和时间差，使用PID公式计算输出调整量。



### 3. EP算法实施（task 2）

**3.1设计理由**

进化编程（EP）作为一种优化技术，通过模拟自然选择和遗传变异的过程来进化出更适应环境的解决方案。在我们的场景中，EP被用于开发一种能够有效导航并在火星表面收集土壤样本的火星车控制器。EP的核心优势在于其能够在复杂、多变的环境中，通过迭代过程自主学习和适应，进而找到高效的导航路径。

**3.2 实验设计**

**3.2.1 初始化阶段**

1. 设定时间步长、最大速度等参数。
2. 启用摄像头、左右轮电机、距离传感器、光线传感器和地面传感器。
3. 创建 Trainer 实例，用于后续的遗传算法优化和MLP网络训练。

**3.2.2 数据读取与运动控制阶段**

1. 从距离传感器、光线传感器、地面传感器和摄像头读取数据，用于决策和行动控制。
2. 根据MLP网络的输出调整机器人的左右轮速度，实现行进、转向和避 障。

**3.2.3 评估与优化阶段**

1. 在 run\_optimization 中运行遗传算法，优化MLP网络参数。
2. 在 run\_best 中执行最优化的MLP网络，控制机器人行动。
3. 在每次迭代中，通过 \_\_evaluate\_genotype 方法评估MLP网络控制下的机器人性能。
4. 在主循环中，连续调用数据读取、运动控制和性能评估方法，直到达成目标或条件终止。

**3.3 MLP的使用**

**1. 决策制定：**

MLP作用：MLP网络作为决策制定的核心，根据输入的传感器数据输出控制信号，这些信号决定了小车的运动行为。

实现方式：通过 self.trainer（一个 Trainer 类实例）来管理MLP网络。MLP网络可能根据接收到的传感器数据（如距离传感器、光线传感器和地面传感器的读数）来生成输出。

**2. 运动控制：**

MLP输出：MLP网络的输出被用于确定小车的左右轮速度（self.velocity\_left 和 self.velocity\_right）。

动态调整：根据环境条件的变化，MLP网络能够实时调整输出，以适应新的导航和避障需求。

**3. 遗传算法优化：**

GA与MLP结合：遗传算法（GA）用于优化MLP网络的权重。通过 self.trainer.population 和 self.trainer.update\_mlp() 方法，MLP网络在遗传算法的帮助下进行优化，以提高决策的准确性和效率。

适应度评估：在 \_\_evaluate\_genotype 方法中，通过运行小车模拟并评估其性能（适应度），来决定哪些MLP网络的权重配置是最优的。

**3.3实现过程：**

* **初始种群的生成（ga.py）：**初始时，创建一个包含多个随机生成控制器参数的火星车种群。

1. num\_generations: 进化的代数。
2. num\_population: 种群中的个体数量。
3. num\_elite: 每代中优秀个体的数量。
4. cp (crossover probability): 交叉的概率。85
5. mp (mutation probability): 变异的概率。35

* **适应度评估(train.py)：**

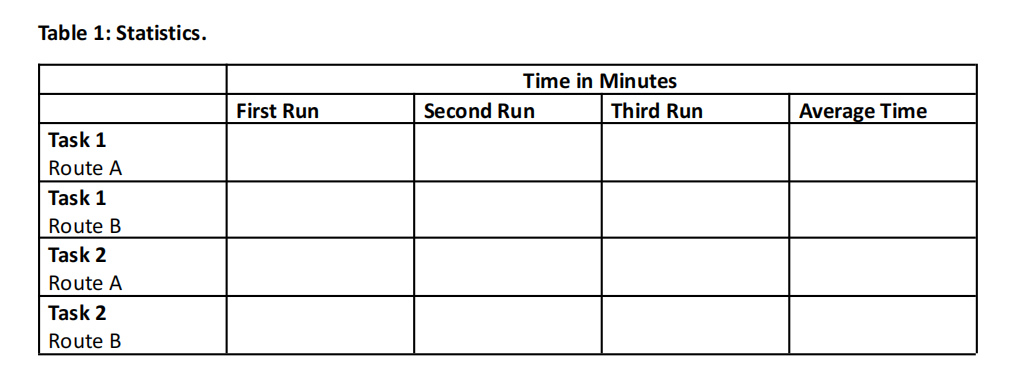
1. 控制火星探测车的运动（get\_output\_and\_cal\_fitness），基于神经网络 的输出来调整车轮的速度。
2. 计算适应度（cal\_fitness\_with\_reward），以评估火星探测车的性能。 适应度函数考虑了前进速度、避免碰撞和转向的能力。
3. 使用光传感器、地面传感器和距离传感器的数据作为神经网络的输入， 帮助探测车做出更好的决策。

* **选择和遗传算法：**

1. 使用ga.py创建随机种群。
2. 通过遗传操作（如选择、交叉和变异）来进化种群。种群繁殖 （population\_reproduce）:
3. 选择操作基于个体的适应度，优秀的个体有更高的机会被选中繁殖。
4. 交叉操作混合两个个体的基因（代表神经网络权重的数组）来产生 新的个体。
5. 变异操作随机改变个体的某些基因，引入新的遗传变异。
6. 迭代优化过程:
7. 通过多次重复这个繁殖过程，遗传算法逐渐优化个体（在这个场景 中是控制器的参数）的性能。
8. 每一代都会选择适应度最高的个体，以此确保解决方案朝着更优的 方向发展。
9. 评估每个个体的适应度，并选择最佳个体。

### 3. 结果与分析

* 数据展示: 使用表格和图表展示实验结果。



* 性能评估: 分析EP控制器的性能，包括成功率、速度和效率。
* 比较分析: 如果可能的话，将你的结果与基于BBR的方法或其他ER方法的结果进行比较。

### 4. 讨论

* 实验观察: 讨论实验中的任何有趣发现或意外情况。
* 决策影响: 探讨影响性能的关键决策，并解释其原因。
* 方法限制: 讨论EP方法的局限性及其对项目结果的可能影响。

### 5. 结论

* 总结: 简要总结EP方法的有效性和研究结果。