计算机、互联网领域技术交底书模板

专利名称	基于进化-大模型协同的 AGV 动态调度自进化系统及方法		
专利类型		联系人	周游
联系电话		E-mail	

请发明人先按照自己的理解对下述问题进行解答，以作为双方沟通的基础，若对下述问题的解答存在问题，代理人会进一步与发明人沟通！

1、背景技术及现在技术存在的技术问题

现有工厂在……时，使用……方法进行……，该方法（或这些方法）存在以下问题：

一、技术痛点

1. 传统 AGV 静态路径规划，无法适应 15%货架变动率
2. 83%调度系统仅优化单一目标（时效/能耗不可兼得）
3. 机械参数调整需 4 小时人工重配置
4. 突发障碍响应延迟 > 500ms，急停事故率 12%

二、核心场景

场景	痛点	专利方案
智能仓储	货架移动 200 次/小时	六维向量生成器实时优化路径
汽车产线	10 分钟换 7 种车型	LLM 自动生成夹具策略（部署 < 90 秒）
危化品运输	震动须 < 0.3g	物理约束编译器确保 OSHA 合规

三、关键提升

- 响应速度：500ms→200ms（↑150%）
- 优化维度：单目标→6 维协同
- 策略更新：4 小时→实时

提供一种基于进化算法与大语言模型协同优化的 AGV 动态调度方法，解决现有调度算法在复杂工业场景下无法实时适应环境变化、难以平衡多目标优化需求的问题。

2、请解释导致该问题存在的原因是什么？

根本性技术障碍：

动态性缺陷：传统调度算法采用固定参数或离线训练模型，无法实时响应车间布局变更、订单优先级调整等动态事件

物理约束冲突：AGV 运动学约束（如最小转弯半径）与调度策略存在耦合，导致生成的路径方案实际不可行

多目标失衡：运输效率、能耗、安全等指标间的非线性关系难以用数学公式准确建模

3、现有技术是如何解决该问题的？存在哪些不足？

主要技术路线对比：

技术类型	实现方式	典型缺陷
传统遗传算法	路径编码+适应度函数	变异过程忽视物理约束，需二次验证
强化学习方法	基于奖励机制的策略训练	训练周期长，冷启动问题严重
规则引擎系统	专家经验编写 IF-THEN 规则	无法应对未预见的异常场景

关键不足：

响应滞后：算法更新周期＞车间动态变化频率（通常＞10 分钟）

知识固化：无法吸收现场运行数据持续优化规则库

硬件割裂：调度算法与 AGV 控制器缺乏双向数据通道

4、详细说明本专利是如何解决该问题的？**（该部分为交底书的核心部分，请详细描述）**

4.1 系统架构

我们提出了一种基于进化算法与大模型（LLM）协同的 AGV 动态调度自进化系统，包含以下核心模块。

AGV 集群子系统：搭载激光雷达、IMU 等传感器，实时采集位姿、能耗、载重等运行数据；定时上传压缩后的特征数据包至边缘网关。

边缘网关子系统：对数据进行降噪、特征提取，转发有效数据到云端。

多模态约束编译模块：动态将 AGV 物理约束转化为数学约束方程，同时生成自然语言描述的规则文档。

LLM 规则生成器：基于提示工程生成符合约束要求的策略代码，实现策略更新。

算法自我进化引擎：通过进化算法与 LLM 的协同演化机制，动态进行适应度评估和策略迭代。

热部署模块：通过热补丁机制无缝更新 AGV 控制程序，支持异常快速回滚。

如图 1 所示，系统通过实时数据采集、特征提取、约束建模、策略生成、进化优化和快速部署的全链路闭环流程实现动态调度自进化。

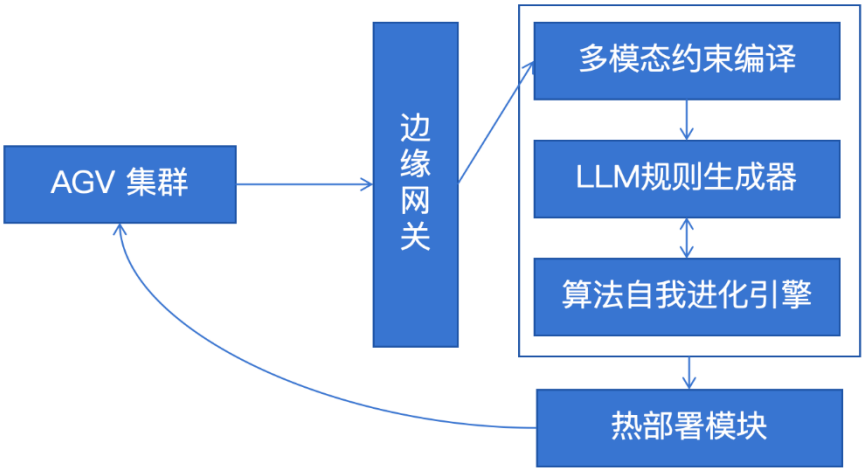

图 1 系统组成

4.2 具体实施方法

为实现以上系统架构，我们提出了一种基于进化-大模型协同的 AGV 动态调度自进化方法，围绕六大模块展开，该方法包括以下步骤。

步骤 1-2（数据采集与清洗）：每 30 秒触发，AGV 集群实时采集数据，并上传至边缘网关，边缘网关清洗、提取特征，压缩并上传云端。

步骤 3-4（约束编译与策略生成）：多模态约束编译模块生成动态物理约束，LLM 规则生成器依据约束生成调度策略代码，以事件驱动方式触发（如，当新障碍出现时）。

步骤 5（策略进化）：据预设时间周期（如每小时）或适应度变化阈值触发进化迭代 。

步骤 6（热部署）：通过消息队列实时监听策略更新事件。

4.2.1 **实时数据采集（AGV 集群）**

利用 AGV 本体传感器与环境感知传感器融合获取高精度数据，通过数据同步引擎实现传感器时间戳同步，定时向边缘网关上传数据，见图 2。

图 2 多源数据采集逻辑链

AGV 本体传感器搭载激光雷达、IMU 等传感器，环境感知传感器包括 UWB 定位基站、视觉传感器等，实时采集位姿、能耗、载重等多源数据，核心字段见表 1。

表 1 多源数据来源及示例

数据类型	数据字段	采集源
AGV 本体数据	位姿数据(x,y,θ) ...	激光雷达（±2cm 精度） IMU（1000Hz）
	电池 SOC ...	BMS 电池管理系统
	载重等效载荷 ...	压力传感器
环境数据	UWB 基站坐标(x_b,y_b) ...	UWB 定位系统（1 个/50㎡）
	动态障碍物边界坐标集(x_0,y_0) ...	视觉传感器（30Hz）
	车间温湿度 ...	环境监测终端
任务数据	当前工单 ID 目标货架坐标 任务优先级参数 ...	MES/WMS 系统接口

位姿数据(x,y,θ)通过激光雷达与 IMU 融合实现。电池 SOC（State of Charge）反映电池剩余电量百分比，采用库仑计数法与端电压法联合估计，计算方式为$SOC_{\text{final}} = 0.7 \times \text{SOC}_{\text{coulomb}} + 0.3 \times f(V_{\text{terminal}})$。载重状态通过压力传感器阵列检测，计算等效荷载公式为 $\text{Load}_{\text{valid}} = \sum_{i=1}^{8} F_i \cdot \cos(\theta_{\text{tilt}})$。

为消除多传感器数据的时间偏差，数据同步引擎采用 IEEE 1588 PTP 协议，确保传感器时间同步，最大时钟偏差≤1.5ms。所有数据每 30 秒打包压缩后上传至边缘网关数据池，将被进一步处理与特征化。

4.2.2 **数据特征提取（边缘网关）**

边缘网关对接收到的传感器数据进行数据清洗和特征提取，生成六维特征向量 $V=[\eta_{\text{trans}},\epsilon_{\text{energy}},\rho_{\text{collision}},\tau_{\text{delay}},\omega_{\text{load}},h_{\text{health}}]$，包含运输效能因子、能耗偏离度、冲突风险系数、时效衰减因子、负载动态因子、设备健康度。处理逻辑分为以下子步骤：

1. 数据清洗

对 IMU 原始数据采用长度为$N=10$的移动平均滤波进行数据平滑，滤波公式为

$$x_{\text{filtered}}[k]=\frac{1}{N}\sum_{i=k-N+1}^{k}x_{\text{raw}}[i]$$

结合3σ准则剔除异常值，$|\mathrm{d}r\{\Delta x\}|>3\sigma$时视为抖动噪声。对缺失数据采用 Lagrange 插值法恢复，公式为

$$x(t)=\sum_{j=0}^{n}x(t_j)\prod_{m=0,m\neq j}^{n}\frac{t-t_m}{t_j-t_m}$$

2. 六维特征计算

六维特征向量是本专利定义的特征化数据结构，用于量化描述 AGV 运行状态与环境特征，是实现动态调度的数据基础，六维特征可以被嵌入调度评价函数中进行算法适应度评估，指导调度策略的持续演化，作为优化方向，见图 3。

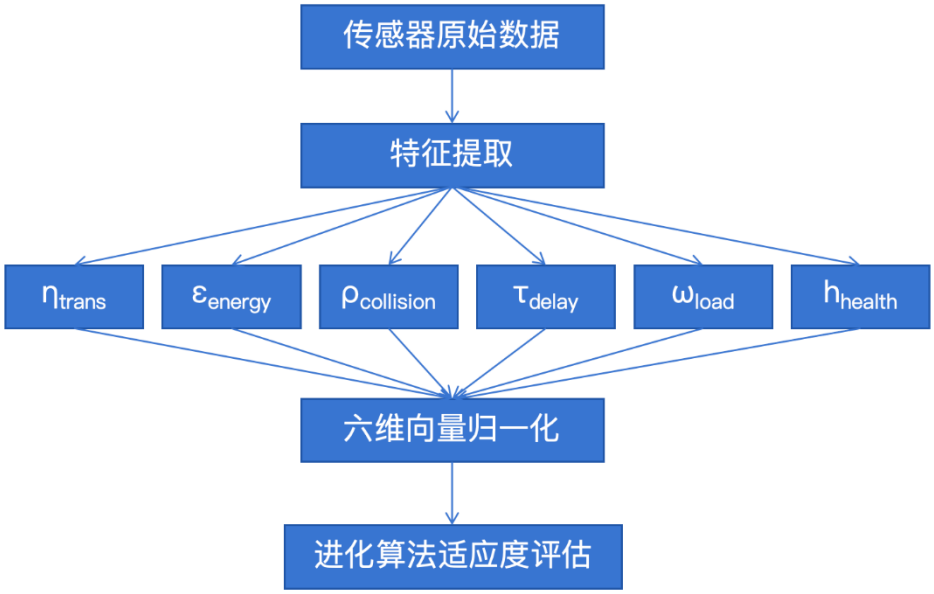

图 3 六维特征到算法适应度逻辑链

六维特征向量各分量的定义和计算方式见下表 2。

表 2 特征提取说明

特征维度	所需原始数据	数据来源	处理逻辑
η_{trans} （运输效能因子）	1. 实际完成订单量N_{actual} 2. 理论最大产出能力N_{theory}	MES/WMS 系统接口	$\eta_{\text{trans}} = \frac{N_{\text{actual}}}{N_{\text{theory}}} \times 100\%$
ϵ_{energy} （能耗偏离度）	1. 电池电压$U(t)$（0－48V） 2. 放电电流$I(t)$(0-100A) 3. 运行时长$[t_0, t_1]$ 4. 标准任务能耗（历史均值）E_{baseline}	BMS 电池管理系统	1. **当前能耗计算：** $E = \int_{t_0}^{t_1} U(t)I(t)dt$ 2. **偏离度计算：** $\epsilon_{\text{energy}} = \left\lvert 1 - \frac{E}{E_{\text{baseline}}} \right\rvert \times 100\%$
$\rho_{\text{collision}}$ （冲突风险系数）	1. UWB 基站坐标 2. 动态障碍物坐标集(x_i, y_i) 3. AGV 实时位姿(x, y) 4. 相对速度 v_{rel}	UWB 定位系统+视觉传感器	1. **计算最近障碍物距离：** $d_{min} = \min_i \sqrt{(x - x_i)^2 + (y - y_i)^2}$ 2. **结合速度计算 TTC：**$TTC = \frac{d_{min}}{v_{rel}}$ 3. **冲突系数：**$\rho_{\text{collision}} = \frac{1}{\text{TTC}+\varepsilon}$
τ_{delay} （时效衰减因子）	1. 计划到达时间 T_{plan} 2. 当前位姿 (x, y) 3. 路径节点 (x_k, y_k) 4. 当前速度 v	MES 系统 激光雷达	1. **剩余路径长度：** $L_{\text{remain}} = \sum_{k=1}^{n} \sqrt{(x_k - x_{k-1})^2 + (y_k - y_{k-1})^2}$ 2. **预测到达时间：**$T_{\text{est}} = \frac{L_{\text{remain}}}{v}$ 3. **延误时间：** $\Delta T = T_{\text{est}} + T_{\text{now}} - T_{\text{plan}}$ 4. **衰减因子：**$\tau_{\text{delay}} = e^{-\lambda \cdot \Delta T}$ 其中λ 为任务优先级相关衰减率
ω_{load} （负载动态因子）	1. 压力传感器 F1~F8 读数 2. 质心偏移角 θ	压力传感器	1. **前后轮组载重：** $F_{\text{front}} = \sum_{i=1}^{4} F_i, \quad F_{\text{rear}} = \sum_{j=5}^{8} F_j$ 2. **加权求和：** $\omega_{\text{load}} = 0.2 \cdot F_{\text{front}} + 0.3 \cdot F_{\text{rear}}$ 3. **动态修正：** 若 $\theta > 10°$，则： $\omega_{\text{load}} = \omega_{\text{load}} \cdot \sin(\theta)$

h_{health} （设备健康度）	1. 电机温度 $temp$ 2.振动频谱评分 vib_score 3. 电池 SOH	电机编码器 IMU BMS 电池管理系统	$h_{\text{health}} = 0.6 \cdot \text{SOH} + 0.3 \cdot \left(1 - \frac{\text{temp}}{70}\right) + 0.1 \cdot \text{vib\_score}$

其中，TTC 表示 AGV 与最近障碍物的预计碰撞时间，单位一般为秒，TTC 值越小，碰撞越紧迫。

4.2.3 **物理约束动态注入（多模态约束编译）**

基于边缘网关上传的实时数据，结合 AGV 机械参数、环境配置参数等固有属性，动态生成数学约束。约束类型包括运动学约束、动力学约束、能量约束和环境约束等，确保调度策略在物理可行性和安全性范围内运行。系统固有参数部分见表 3。

为有效管理各类约束条件并实现调度策略的优先级控制，系统引入分层二次规划（Hierarchical Quadratic Programming, HQP）方法。该方法通过分层建模和求解的方式，对各类约束条件进行统一处理。最终输出结果将同步生成机器可执行代码和人类可读规则文档，实现双模输出验证机制。

表 3 约束固有参数

数据类型	数据字段示例
AGV 机械参数	电机空载功率(P_{base})、电池容量($C_{battery}$)、最小转弯半径(r_{min})、 最大加速度(a_{max})、最大载重(m_{max})、机械臂伸展范围(l_{arm})等
环境配置参数	基础安全距离(d_{safe_base})、通道有效宽度($w_{channel}$)、 摩擦系数(μ)、最大允许坡度(θ_{max})等

1. 约束转换

首先基于机械参数建立物理可行性约束模型，将 AGV 的几何特性、动力学参数和能量管理要求转化为可执行的数学约束方程，流程见图 4。

在几何约束部分，基于阿克曼转向模型计算最小转弯半径对应的最大转向角，表达式为 $\phi_{\max} = \arctan\left(\frac{L}{r_{\min}}\right)$，其中 L 为轴距。当 $r_{\min} = 0.8\,\text{m}$ 时，可得最大转向角为 $\phi_{\max} = 56.4°$。另一个关键几何约束为通道通过性验证，其不等式为 $w_{\text{channel}} - 2d_{\text{safe}} \geq w_{\text{AGV}} + 0.2\,\text{m}$，该条件用于防止 AGV 与通道侧边碰撞，若被违反则触发路径重规划。

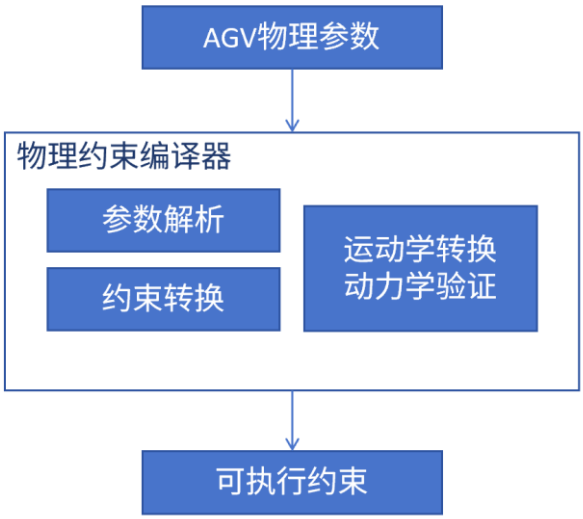

图 4 物理约束注入流程

动力学约束部分涵盖了最大加速度与载重稳定性两个方面。最大加速度由牛顿第二定律近似表示为 $a_{\max} = \frac{\mu \cdot (m_{\text{AGV}} + m_{\text{load}}) \cdot g}{m_{\text{AGV}} + m_{\text{load}}}$，其中摩擦系数 μ来自环境配置参数，m_{AGV}为车体质量，m_{load}来自实时压力传感器数据。载重稳定性约束用于防止车辆在坡道中倾覆，其表达为 $m_{\text{load}} \leq m_{\max} \cdot \left(1 - \frac{\theta_{\text{current}}}{\theta_{\max}}\right)$，其中$\theta_{\text{current}}$ 为当前坡度（IMU 测量值）。

在能量约束方面，调度系统需验证剩余任务可完成性。续航约束公式为 $t_{\text{remain}} = \frac{\text{SOC} \cdot C_{\text{battery}}}{P_{\text{base}} + 0.1 \cdot \omega} \geq t_{\text{required}}$，其中 ω 表示负载动态因子。另一个能量相关限制是电机功率限制，用于防止过载，其表示为$P_{\text{current}} = P_{\text{base}} + 0.05 \cdot v^2 \leq P_{\text{rated}}$，其中 v 为当前车速。

其次考虑环境约束，将障碍物、动态环境信息映射为局部避障约束，主要包括避障约束和坡度约束。

在避障约束方面，基于实时冲突风险系数，构建以 AGV 为中心的自适应安全区域 $\mathcal{S} = \left\{(x, y) \mid \sqrt{(x - x_0)^2 + (y - y_0)^2} \geq d_{\text{safe}}\right\}$，其中安全距离为 $d_{\text{safe}} = d_{\text{safe\_base}} + 0.3 \cdot \rho \cdot v$，$\rho$为冲突风险系数。

在坡度约束方面，考虑坡度约束以防止 AGV 打滑，具体为 $\theta_{\text{current}} \leq \theta_{\max} \cdot \left(1 - \frac{\rho}{2}\right)$。

2. HQP（层次二次规划）设计

在 AGV 动态调度场景中，系统需同时满足多类约束条件（如安全避障、机械性能、能源限制等），但这些约束的优先级和性质存在显著差异，传统单层优化难以区分优先级，可能导致关键约束被次要目标牺牲。例如，为降低能耗而逼近障碍物可能引发安全隐患。

利用 HQP 分层求解处理约束冲突，在满足高优先级约束的前提下，逐级优化性能指标。HQP 按照硬性与软性优先级将上一步中的约束分级处理。按层级求解约束，输出最优控制指令。首先约束分层如下表所示

表 4 约束层级划分

层级	约束类型	约束内容
L1	避障安全约束（硬性）	避障约束、最大加速度约束
L2	几何/动力学约束（硬性）	转向角约束、通道通过性约束、载重稳定性约束
L3	能源效率约束（软性）	续航约束、电机功率限制

HQP 求解后的输出结果是一个分层优化的控制指令集合，以下为一个输出示例，通过 HQP 处理后的约束集，将作为输入引导下一步的调度策略生成过程:

a. 安全限速：$v_{safe} = min(v_{max}, (d_{obs} - d_{min})/\tau)$

b. 转向角约束：$|\varphi| \leq \varphi_{\max}$

c. 加速度约束：$|a| \leq a_m ax \cdot (1 - \rho_{collision})$

d. 路径规划约束：避开所有不满足$(x, y) \in S$ 的区域

为确保系统可靠性，本模块同步生成符合 ROS2 框架的机器可执行代码（Python 类）和人类可读的规则文档（Markdown/PDF 格式）。通过抽象语法树（AST）比对技术，严格保证调度逻辑的一致性、可追溯性和可维护性。

4.2.4 智能规则生成（LLM 规则生成器）

在获得六维状态特征与物理约束表达式后，系统进入策略生成阶段。该阶段通过结构化提示工程驱动 LLM 生成符合物理可行性与调度目标的策略代码。整个生成过程可分为以下三个子步骤，分别为提示工程构造、代码生成与安全验证机制。策略生成依赖于前两阶段提取的六维特征向量与第三阶段构建的 HQP 约束表达式，作为结构化提示的一部分引导 LLM 推理。

1. 提示工程构造

构造含领域知识的结构化指令，结构化指令包含以下几个方面。

首先是约束条件提示，来源于前述多模态约束编译模块中生成的 HQP 约束表达式，确保生成代码始终满足当前物理环境与设备能力的限制。其次是优化目标提示，依据六维特征向量中的任务参数、资源状态和环境因素，构建优化目标函数，如运输效率最大化或能耗最小化等。此外，指令中还包含代码规范提示，要求生成代码符合 ROS2 框架的开发标准，明确类型注解、加入必要的异常处理逻辑，并完成日志记录机制，以满足工业系统的可部署要求。为提高生成质量，系统还嵌入历史策略作为示例提示，包括 Top-3 策略片段的摘要与注释，帮助模型从过往成功经验中学习调度逻辑。

例如，在约束条件中，系统可能明确指出“转向角不得超过 56.4°”、“载重应随坡度自动调整”，而优化目标则侧重“最大化运输效能”、“最小化单位能耗”等。此外，指令可能引用“2023-11 策略 A”等历史策略，以提供结构模板和实现思路。

2. LLM 代码生成

在获得结构化指令后，基于特征向量与历史运行数据，LLM 将进入实际的代码生成阶段，生成新的调度策略 Python 代码片段。

首先，模型从输入的约束与优化目标中抽取核心调度规则，明确任务的边界条件与优化重点。随后，依据任务类型和特征向量所反映的问题结构，自动选择合适的求解算法，如 A* 搜索、启发式规则、图优化方法等。在算法框架确定后，模型合成对应的 Python 代码片段，构建调度策略类，完成任务分配、路径规划、资源检查等函数的定义，并在必要处嵌入异常处理逻辑，保证系统在边界情况的稳定性。生成代码还将包含结构化注释，用于明确各段代码与其约束或目标函数之间的对应关系，便于后期维护与可视化展示。

3. 安全验证机制

在 Docker 沙箱中对生成代码进行冒烟测试，确保代码逻辑与约束条件的一致性。

验证流程包括静态语法检查，确保无语法错误或不符合规范的实现；接着进行单元测试，系统自动生成一组测试用例，在典型与极端条件下验证策略行为；随后进行约束验证，检查代码逻辑是否严格满足先前注入的物理与任务约束；最后还会评估代码在资源受限条件下的执行效率，确保其满足边缘计算场景对响应时效和资源占用的要求。

若代码通过所有验证环节，将被系统标记为“已验证”，并交由下一阶段的演化优化模块继续迭代。若验证未通过，系统将记录失败原因，自动调整提示工程的相关字段，并重新触发生成过程，形成闭环的生成—验证—优化工作流。

4.2.5 策略优化筛选（进化引擎）

为实现调度策略的持续自适应与性能提升，进化引擎通过进化算法对生成的调度策略代码结合 LLM 进行迭代优化，持续提升调度性能。融合个体进化、自我反思、语义融合与集体经验提取四大机制，如图 5 所示，其核心流程如下。

1. 工厂模型初始化与适应度函数构建

在优化前，需构建基本工厂模型，涵盖环境布局、设备参数等，同时准备测试数据集和仿真环境。
适应度函数用于评估调度策略的优劣性，定义为：

$$Fitness(S) = \sum_{i=1}^{6} w_i \cdot f_i(S)$$

其中S是调度策略个体，F_i为六维特征向量的第 i 项归一化后数值，w_i为特征权重，满足 $\sum w_i = 1$，可以反映业务偏好。

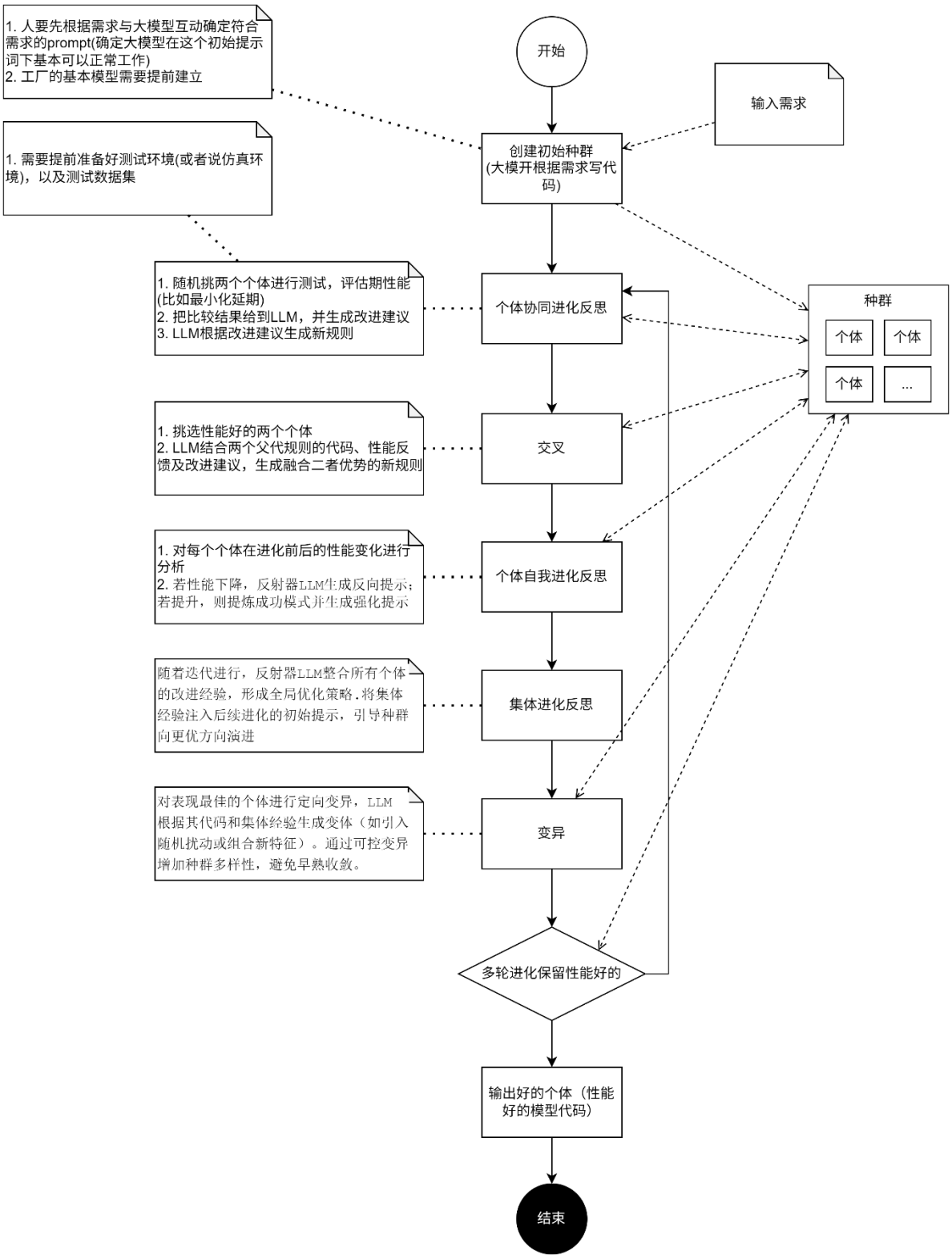

图 5 算法自我进化核心流程

2. 进化流程

a. 初始种群构建与基因编码

原始种群：LLM 生成的 N=50 个策略代码（通过 AST 解析为语法树）

策略基因编码：基因片段定义为可替换子树，如调度优先级表达式、路径算法（如 A*、Dijkstra）、条件逻辑（如能耗阈值、避障策略）等

b. 个体协同进化机制

LLM 评估两个个体的适应度差异，生成改进指令，创建新个体。

具体步骤为为对于策略个体S_i 与 S_j，若满足$\left|Fitness(S_i) - Fitness\left(S_j\right)\right| > \delta$，则触发 LLM 协助生成新个体 S_{new}，其改进建议包括可调基因段$G_{replace}$和替代建议G_{new}，得到$S_{new} = S_i \setminus G_{replace} \cup G_{new}$。

c. 语义交叉融合机制

选择性能优异的策略，交换语义等价代码块，进行 AST 子树级别融合。

具体为对两个高适应度个体 S_a, S_b，识别等价语义段$\{G_a^i\}, \{G_b^i\}$，交换子树生成新个体S_{cross} ，即$S_{cross} = S_a \setminus G_a^i \cup G_b^i$，确保语义一致性由 LLM 语义检查器辅助完成。

d. 个体自我反思机制

构建性能变化追踪序列，若性能下降，LLM 生成改进建议。具体为每个个体在连续演化中记录其性能变化序列$\{Fitness^t\}$，计算一阶差分$\Delta Fitness^t = Fitness^t - Fitness^{t-1}$，若 $\Delta Fitness^t < 0$，LLM 触发“性能下降诊断”，生成自我修复指令。

e. 集体演化反思机制

在每一代后汇总全体个体的变异过程与效果，构建经验池 $\mathcal{E} = \{\left(G_{replace}, \Delta Fitness\right)\}$，使用注意力机制计算每类变异的平均提升权重：$\alpha_i = \frac{\exp(\Delta Fitness_i)}{\sum_j \exp(\Delta Fitness_j)}$，用于指导下一代提示工程中策略变异优先级。

f. 可控变异机制

对表现最优的个体进行定向变异，比如进行参数变异和逻辑变异。具体为在物理约束范围内微调参数，如

$\phi_{max}^{new} = \phi_{max} \pm \Delta\phi, \quad \Delta\phi \in [-5°, 5°]$，或修改函数中的权重组合，$\overrightarrow{w^{new}} = \vec{w} + \Delta\vec{w}$， 约束：$\sum w_i = 1$。

g. 进化停止准则

持续进行多轮进化，保留性能优异的个体，通过交叉、变异、自我反思和集体反思不断优化策略。进化过程停止条件设置为以下任意条件满足：达到最大迭代次数（默认 100 代）、连续 20 代最优适应度提升小于 0.5%、适应度达到预设目标值（例如 ≥ 0.95）。

h. 可解释性输出

最终输出性能最优的模型代码，作为调度策略的最新版本，包括策略代码以及进化过程可视化图谱和可解释性报告，可视化展示策略变异路径与性能变化曲线，且包含每一代性能指标与演化路径。

经过进化引擎评估与迭代优化后，系统选出最优策略代码，并将其作为部署输入，交由热部署模块实现在线无缝更新。

4.2.6 无损策略热部署（热部署模块）

热部署模块实现调度策略代码的无缝更新，确保 AGV 控制系统不间断运行。具体实施方法包括以下两步。

1. 差分补丁生成与部署

通过抽象语法树差分（AST Diff）算法与 BSDiff 压缩生成≤2KB 的增量补丁，注入补丁并重启调度线程，实现 10μs 内无中断切换。操作步骤为首先识别策略版本S_{old} 与S_{new} 的语法差异$\Delta_{AST} = AST(S_{old}) \oplus AST(S_{new})$，再通过 BSDiff 算法压缩为补丁文件 δS，满足$|\delta S| \leq 2\,\mathrm{KB},$ 平均部署时间 $\leq 10\,\mu s$。

2. 版本管理与回滚

实现版本控制机制，支持策略版本记录、对比和回滚。

每个策略版本添加唯一标识符和时间戳，如$\mathrm{strategy_ID} = \mathrm{SHA256}(S) + \mathrm{timestamp}$，以版本日志形式记录参数变化和性能指标差异，支持可视化对比。

当新策略性能下降超过阈值时触发回滚，如$\Delta\epsilon_{\mathrm{energy}} > +10\%$将自动触发回滚流程，停止当前策略调度线程，注入上一个已验证策略补丁，重启调度模块，恢复控制逻辑。