

非 Transformer 几何智能：物理直觉驱动的尺寸投影层、Agent 自适应流形分解

纪辉泉，廖嘉轩，王进，蔡瑞泉

[摘要:] 现有智能体感知系统普遍存在尺寸固化缺陷：几何门禁与感知逻辑强绑定于固定三维欧氏空间，无法兼容二维网格、2.5 维高度场、高维抽象特征空间等多形态环境。本文提出尺寸投影层（Dimensional Projection Layer, DPL）架构，以信息函数宇宙学（IFC）的三公理体系为理论基础，将环境感知过程解构为信息流维（ $D1$ ）→信息面维（ $D2$ ）→信息体维（ $D3$ ）→时空信息流维（ $D4$ ）四级流形层级。DPL 设计四项核心能力：①任意尺寸输入的自动层级流形分解；②低维输入的升维补全与高维输入的降维抽象；③自适应几何门禁校验；④宇宙常数自动校准与物理直觉推理。本文为灵境引擎 R2 几何门禁模块提供核心升级方案，并与 SICA 自指收敛算法在 R5 元层形成安全闭环。

在此理论框架上，本文进一步给出 DPL 的完整工程规范（附录 A-I）：附录 A 定义符号、量纲与单位约定；附录 B 定义五个稳定接口（`detect\_dimension`、`calibrate`、`project`、`check`、`infer`）及灵境协议栈映射；附录 C 给出 $D1-D4$ 流形分解算法伪代码；附录 D 给出七项自适应几何门禁（信息守恒、曲率有界、涡旋校验、拓扑合法、自指稳定、跨维隔离、因果一致）的实现规范；附录 E 给出宇宙常数校准与四条物理直觉推理链（引力、周期、守恒、拓扑）；附录 F 给出 DPL→SICA 三类数据结构（`GeometricInvariant`、`GateCounterexample`、`IntuitionHypothesis`）与灵境协议 1-11 的完整映射；附录 G 给出灵境引擎三层一体化世界观架构（单体智能 R2-R5、多智能体仿真、SICA 自指迭代）；附录 H 给出 R3 规划层 AOP 小模型（Action Outcome Predictor）的接口规范；附录 I 则从本体论层面论证，当前主流统计拟合路径（LLM）与本文架构路径不可通约——前者是“读万卷书”，后者是“行万里路”；后者通过 SICA R1-R10 实现“婴儿摔跤式”的收敛学习（附录 I）。

[关键词:] 尺寸投影层；IFC；信息场；泡壁投影；几何门禁；物理直觉引擎；SICA；灵境引擎；AOP 小模型；架构路径

1 引言

1.1 问题背景

当前主流 Agent 感知系统均基于固定尺寸假设构建，通常默认感知空间为三维欧几里得空间，导致：

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

1. **维度刚性**：仅能处理预设维度，面对 2D 网格、2.5D 高度场、4D 时空及高维抽象空间时直接失效；
2. **几何门禁硬编码**：碰撞、边界、拓扑校验与特定维度绑定，跨维度泛化能力为零；
3. **物理直觉缺失**：只能读取原始观测值，无法从几何结构中推断引力、守恒、周期、拓扑约束；
4. **维度信息损耗**：高维被硬性截断，低维缺乏升维补全。

1.2 核心理念

本文提出 DPL，核心理念为：

“统计是计算，几何才是智能。”

基于 IFC 的信息一元论与三公理体系，DPL 将感知范式从“原始像素/数值采集”升级为“信息流形结构解析”，通过维度投影算子实现：

- **分解**：高维环境输入逐级投影为 $D1 \rightarrow D2 \rightarrow D3 \rightarrow D4$ 层级流形；
- **整合**：低维输入通过拓扑补全自动升维；
- **校验**：基于有效维度自适应激活几何门禁；
- **直觉**：通过宇宙常数校准与流形特征映射，自主推断环境物理规律。

1.3 贡献

1. 提出 DPL 维度投影层，设计任意维度输入的 $D1-D4$ 流形分解路径；
2. 以 IFC 三公理体系为理论基础，统一信息场、泡壁投影、Leech 晶格术语；
3. 设计自适应几何门禁，覆盖 2D /2.5D /3D /4D+；
4. 设计宇宙常数自动校准与物理直觉引擎；
5. 明确 DPL 位于灵境 R2 感知层，SICA 位于 R5 元层，二者形成安全闭环；其中 R2 感知层由 `detect\_dimension`、`calibrate`、`project` 三接口构成，`check` 位于 R2/R3 交界面，`infer` 位于 L4 自指意识层（详细层级划分见附录 G G.2.2）；
6. 建立 DPL 与灵境引擎协议栈（L0-L5、协议 1-11）的完整映射，为工程落地提供接口基准；
7. 设计**物理直觉引擎**：引力、周期、守恒、拓扑四条推理链，从 $D1-D4$ 流形特征自主推断环境底层物理规律（附录 E）；
8. 为 R3 规划层设计 **AOP 小模型**（Action Outcome Predictor）的完整接口规范，作为因果图搜索的神经加速器（附录 H）。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

9. 从本体论层面论证统计拟合路径（LLM）与架构路径（DPL/IFC）的不可通约性，提出“读万卷书 vs 行万里路”与“婴儿摔跤式收敛”的范式对比，并阐明架构路径在收敛性、感知结构与世界模型上的结构性优势（附录 I）

符号约定：标量取绝对值 $|\cdot|$ ；向量与复数取模 $\|\cdot\|$ 。花体 V_t 专指 SICA Lyapunov 势函数，与信息密度场 $\Phi(x)$ 区分。自指映射记为 $\mathcal{M}_{self}$ 。

2 理论基础：IFC 公理体系

IFC 以信息一元论为哲学基石，以三条自治公理为逻辑起点，构建从离散信息基元到连续时空几何、从规范群到意识相变的统一框架。DPL 的 D1–D4 层级、几何门禁、物理直觉均从 IFC 导出。

2.1 公理 I：离散信息基元与粗粒化守恒

宇宙所有内容——时空、物质、场、意识——均由不可再分的离散信息基元通过统计积分与粗粒化映射涌现。总信息量具有严格拓扑不变性：

$$Q_{info} = \sum_i b_i = \int_{\mathcal{M}_4} |\Phi(x)|^2 \sqrt{-g} d^4x = Const$$

其中 Q_{info} 以 bit 为单位，约定 1 bit 对应一个信息基元 $b_i=1$ 。左侧 $\sum_i b_i$ 为计数，右侧积分量纲为 bit，故 Q_{info} 的量纲为 bit。

粗粒化映射：

$$\Phi(x) = N_0 \lim_{\epsilon \rightarrow 0} \left[\epsilon^3 \sum_i b_i \cdot \delta^{(4)}(x - x_i) \right]$$

其中 N_0 为归一化常数，量纲为 $[N_0] = bit^{1/2} \cdot m^{-1}$ ； $\Phi(x)$ 为信息密度场，量纲为 $bit^{1/2} \cdot m^{-2}$ 。在几何单位制下可取 $N_0=1$ 用于量级演示。

DPL 映射：D1 信息流线索对应信息场的一阶梯度与源汇分布；信息守恒校验为几何门禁的核心约束。

2.2 公理 II：动态因果网络与算力节俭

不存在预设背景时空，因果关系由时变有向图 $G(\tau) = (V_G, E(\tau))$ 动态决定。系统遵循最小作用量算力原则，仅对自指投影边界执行高精度展开。

连接振幅：

$$A_{ij}(\tau) \in \mathbb{C}, \quad \sum_j \|A_{ij}(\tau)\|^2 = 1$$

战争迷雾算子：

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

$$\mathcal{R}_{render}(\mathcal{M}_4)=\int_{\Sigma}|P_{\Sigma}\Psi_{24}|^2d\mu_2\ll\int_{\mathcal{M}_4}|\Psi_{24}|^2d\mu_4$$

即高精度渲染算力正比于二维泡壁面积 $A_{\Sigma}\propto L^2$ ，而非三维体积 $V_{vol}\propto L^3$ 。

DPL 映射：D2 信息面维对应泡壁面积上的曲率与涡旋结构；D3 信息体维对应体密度与拓扑；自适应几何门禁负责跨维隔离与算力节俭校验。

2.3 公理 III：对称映射投影与上行反馈

24 维高维信息空间到 4 维感知空间的转换由泡壁投影算子 P_{Σ} 实现。低维物理定律是该投影的几何遗迹。允许智能体通过自指相变产生上行微扰，且过程满足诺特定理。

几何生成方程：

$$g_{\mu\nu}=\eta_{\mu\nu}+\frac{1}{\mathcal{J}_{Pl}}\partial_{\mu}\Phi\partial_{\nu}\Phi+\kappa(\Phi^2-\Phi_0^2)\eta_{\mu\nu}$$

信息场传播方程：

$$\square_g\Phi+\zeta R\Phi=-J(x)$$

上行反馈：

$$\Psi_{con}(x)=\Phi(x)\cdot\Theta(C-C_c)$$

$$\delta\Phi(x)=\epsilon\cdot\nabla_{\mu}\Psi_{con}(x),\quad\epsilon\sim10^{-12}$$

其中 Φ_0 为真空基准信息密度， $\mathcal{J}_{Pl}$ 为普朗克信息常量， C_c 为意识临界复杂度， ϵ 为意识-信息耦合长度（几何制下量纲为[L]）。DPL 映射：D4 时空信息流维对应时间演化与因果关联；上行反馈对应物理直觉引擎中的干预场与规则生成。

2.4 Leech 晶格与泡壁投影低维有效描述

IFC 从八项独立物理约束证明，离散信息基元的承载空间唯一地锁定为 **24 维 Leech 晶格 Λ_{24}** 。其自同构群 Co_0 的 24 维表示分解为：

$$24=1^{\oplus 4}\oplus 2^{\oplus 4}\oplus 3^{\oplus 4}$$

泡壁投影保留低能有效子空间：

$$V_1\leftrightarrow U(1)_Y,\quad V_2\leftrightarrow SU(2)_L,\quad V_3\leftrightarrow SU(3)_c$$

DPL 定位：D1–D4 是 Leech 晶格在泡壁投影下的低维有效流形层级，描述 4 维感知时空中的信息流形结构。高维信息被泡壁投影压缩到 D1–D4。

D1–D4 的完整分解算法见附录 C：D1 含散度项（承载公理 I 守恒约束）、D2 含涡旋项与高斯曲率（承载泡壁面积上的曲率结构）、D3 含欧拉示性数与 Ricci 标量（承载三维体密度与拓扑）、D4 含时间导数与因果关联（承载时空演化与因果链）。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

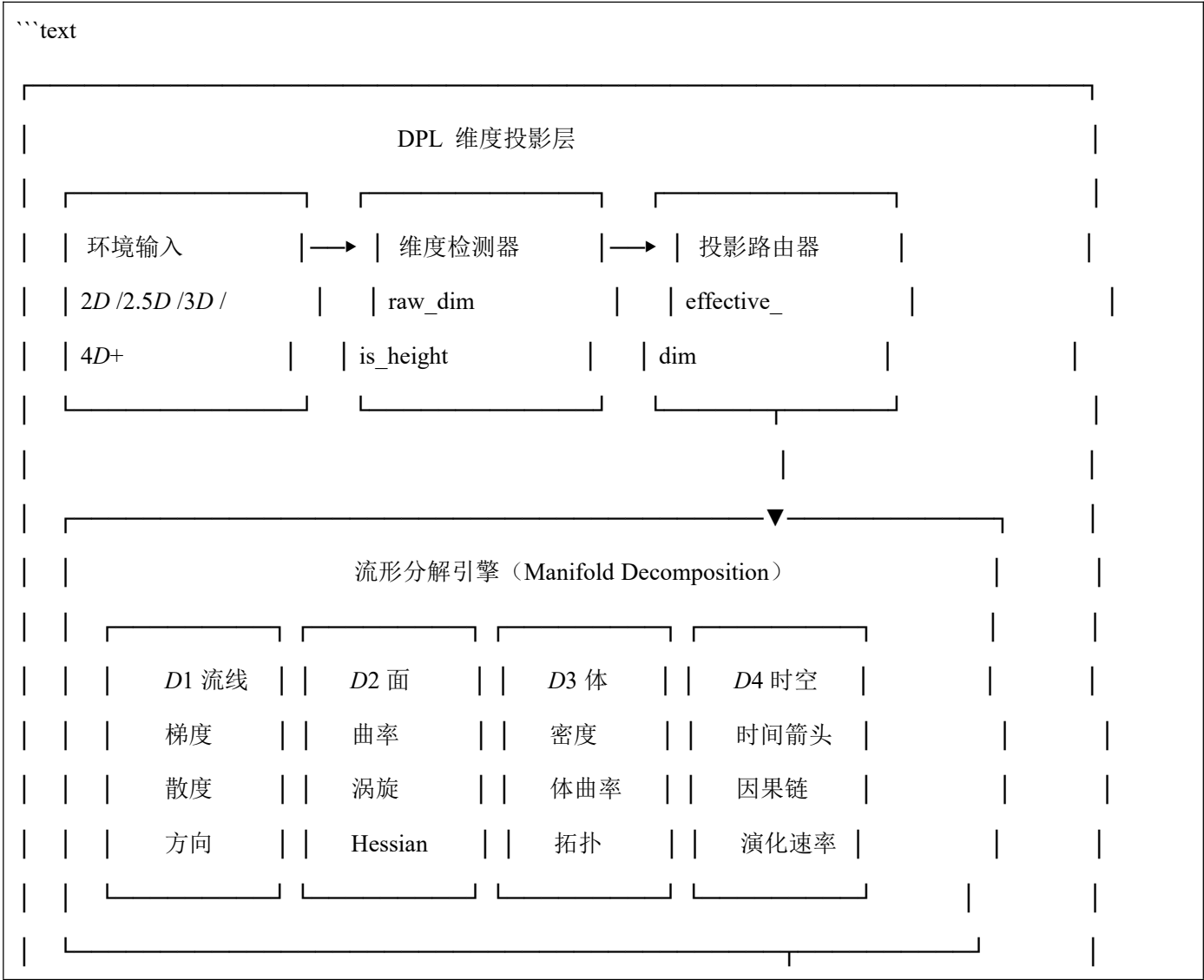
3 DPL 架构设计

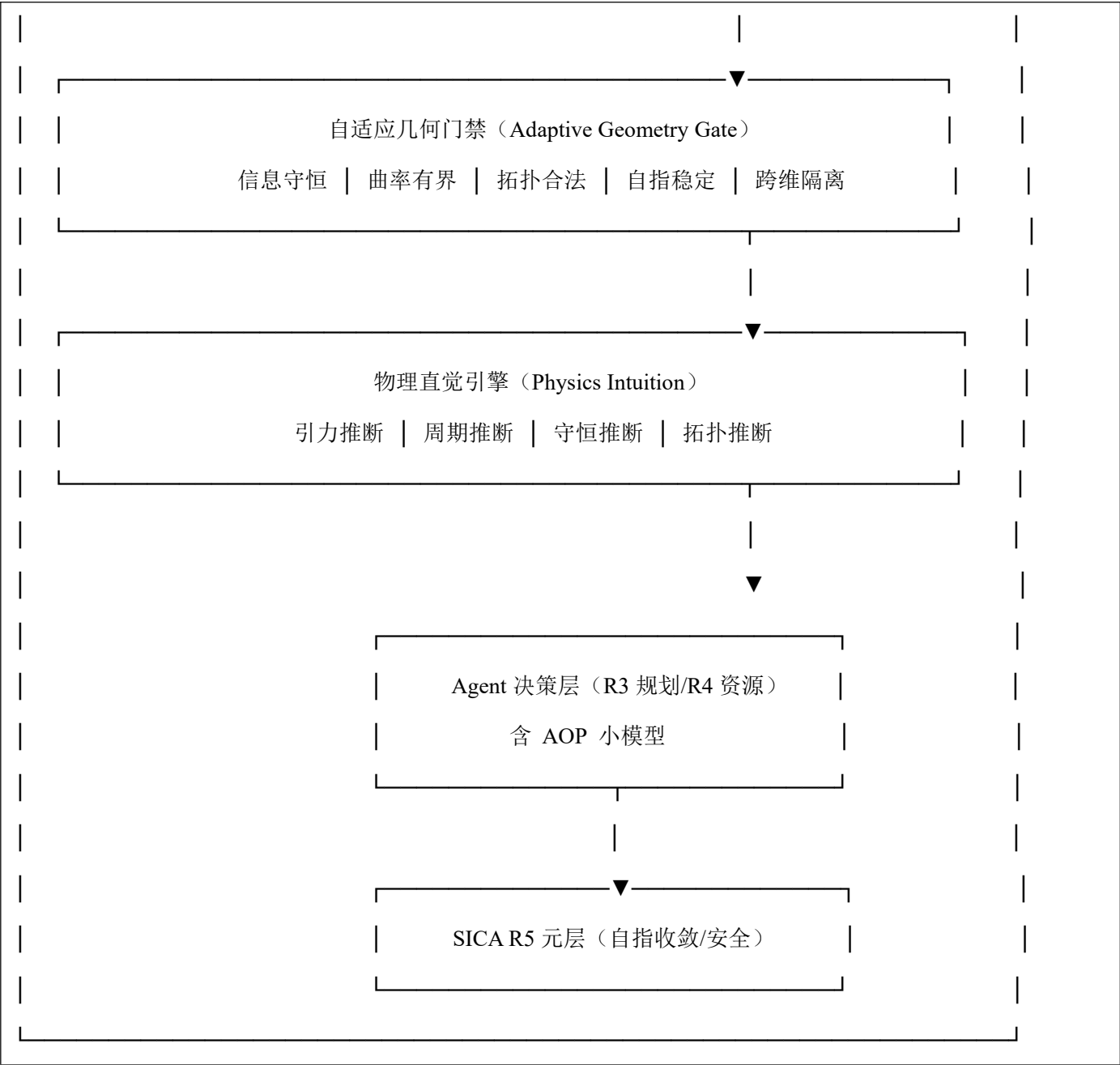
3.1 整体架构

DPL 为端到端的维度自适应感知层，输入为任意维度的环境观测张量，输出为结构化流形特征与物理直觉结论，向下游 Agent 决策层与 R5 元层提供支撑。五大模块：

- 1. 环境输入与维度检测；
- 2. 投影路由；
- 3. 流形分解引擎（D1-D4）；
- 4. 自适应几何门禁；
- 5. 宇宙常数校准与物理直觉引擎。

架构数据流：





架构注释：

- 维度检测器输出 `raw\_dim` 与 `is\_height` 标记，用于 2.5D 高度场识别，保留通道维度以避免被压缩为 2D；
- 投影路由器根据 `effective\_dim` 选择流形分解路径与门禁校验项组合；
- 流形分解引擎输出 D1–D4 四层特征，分别对应信息场的一阶至四阶几何结构；
- 自适应几何门禁根据有效维度动态激活校验项，输出通过/拦截/升维补全/降维抽象四种结果；
- 物理直觉引擎输出三类结构化数据：`GeometricInvariant`、`GateCounterexample`、

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

`IntuitionHypothesis`；这些数据先经 Agent 决策层聚合，通过协议 5 写回场方程，其中
`IntuitionHypothesis` 字段再经协议 8 的记忆模块进入 SICA R5 元层；

- Agent 决策层与 SICA R5 元层为两个独立模块：前者负责 R3 规划（含 AOP 小模型）与 R4 资源分配，后者负责自指收敛与安全进化。

3.2 流形分解引擎

3.2.1 D1 信息流线维

一阶信息流形，描述信息场流动方向、强度与源汇分布。令信息密度场为

$$I(O)=\int_X O(x)w(x)dx, \text{ 熵流密度为 } J_I=-D\nabla I.$$

$$\Phi_{D1}(I)=\{\nabla I, \nabla \cdot J_I, \text{direction}(J_I)\}$$

其中：

- $(D>0)$ 为信息扩散系数，量纲为 $m^2 \cdot s^{-1}$ ；
- 积分域 $X \subseteq \Sigma_t$ 为当前观测窗口对应的三维空间切片子区域；
- ∇I 为信息梯度场（向量）；
- $\nabla \cdot J_I = -D\nabla^2 I$ 为散度场（标量），正值对应信息源，负值对应信息汇；散度项由公理 I 守恒约束引入 D1，作为源汇分布的生成约束；
- $\text{direction}(J_I) = J_I / \|J_I\|$ 为无量纲方向单位向量；当 $J_I=0$ 时定义为零向量。

权重核 $w(x)$ 的量纲为 $[w]=[bit^{1/2}L^{-5}]/[O]$ ，使 $I(O)$ 的量纲与 Φ 一致。完整算法见附录 C C.3。

3.2.2 D2 信息面维

二阶信息流形，描述曲面几何、曲率分布与涡旋结构。定义信息流速度场 $\mathbf{v}_I = J_I / (|\Phi| + \epsilon_0)$,

则：

$$\Phi_{D2}(I)=\{\mathcal{H}(I), \nabla \times \mathbf{v}_I, K(I)\}$$

其中：

- $\mathcal{H}(I)$ 为 Hessian 矩阵（二阶张量），仅用于 D2 内部曲率刻画，不参与跨维隔离校验；
- 涡旋项 $\nabla \times \mathbf{v}_I$ 的行为分三种情形：
 - ✧ 信息密度均匀：旋度恒为零；
 - ✧ 信息密度非均匀：旋度非零，反映密度梯度与流线方向的耦合；
 - ✧ 非单连通或存在拓扑缺陷：额外贡献环量 $\oint \mathbf{v}_I \cdot d\mathbf{l} = n\Phi_{topo}$ ，其中 Φ_{topo} 为拓扑荷量子，

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

量纲 [L]（几何制）；

- 高斯曲率（标量）：当信息面可局部表示为图 $z=\hat{l}(\hat{x},\hat{y})$ 时，

$$K=\frac{\det(\widehat{\nabla}\widehat{\nabla}\hat{l})}{(1+\|\widehat{\nabla}\hat{l}\|^2)^2}$$

一般流形需用第一、第二基本形式计算。

完整算法见附录 C C.4。

3.2.3 D3 信息体维

三阶信息流形，描述三维空间信息体密度与拓扑。D3 计算的是**三维体素胞腔复形**，不使用 Marching Cubes 提取的二维等值面。三维体素胞腔复形的欧拉示性数为：

$$\chi=V_{vert}-E+F-C_{cell}$$

$$\Phi_{D3}(I)=\{\rho_V(I), \chi(\mathcal{M}), R^{(3)}\}$$

其中：

- $\rho_V=\frac{1}{V_{vol}}\sum_i I_i$ 为体素信息密度（标量）；
- $\chi(\mathcal{M})$ 为三维体素胞腔复形的欧拉示性数（整数）；
- $R^{(3)}=h^{ab}R_{ab}$ 为三维黎曼曲率标量（标量）；其中 $\backslash(h^{\{ab\}}\backslash)$ 为三维空间切片上的诱导度规的逆。

对 2.5D 高度场 $I(x,y)$ ，D3 体密度取 $\rho_V=I(x,y)^2$ ，沿高度方向视为单层，胞腔数 ($C_{cell}=0$)。对 $2D\rightarrow 3D$ 形式升维（张量广播），3D 拓扑为平凡乘积结构， $\chi=0$ 、 $R^{(3)}=0$ 。完整算法见附录 C C.5。

3.2.4 D4 时空信息流维

伪时间演化流形，描述状态转移与因果关联：

$$\Phi_{D4}(I_t, I_{t+\Delta t})=\{\partial_t I, C(I_t\rightarrow I_{t+\Delta t})\}$$

其中：

- $\partial_t I$ 为信息场时间演化速率（时间导数）；
- $C(I_t\rightarrow I_{t+\Delta t})$ 为因果关联度，基于互信息或 Granger 因果检验计算。

D4 直接从 `state\_norm` 沿时间轴求导，不依赖 D3 的空间切片。完整算法见附录 C C.6。

4 自适应几何门禁

传统几何门禁为硬编码三维校验。DPL 将其升级为维度自适应校验，根据有效维度 d_{eff}

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

动态激活校验项。

4.1 校验矩阵

符号约定：以下判据中， J_{in} 、 J_{out} 、 K 、 χ 、 $time_arrow$ 为标量（取绝对值 $|\cdot|$ ）；涡旋项 $\nabla \times \mathbf{v}_I$ 为向量或标量（依据 `vorticity\_is\_vector` 分支，分别取 $\|\cdot\|$ 或 $|\cdot|$ ）。

校验项	数学判据	适用维度与触发条件	IFC 来源
信息守恒	$ J_{in} - J_{out} <\varepsilon$	所有维	公理 I
曲率有界	$ K <K_{\max}$	$D2+$	公理 III
涡旋校验	$\ \nabla \times \mathbf{v}_I\ <\omega_{\max}$ （向量涡旋）； $ \nabla \times \mathbf{v}_I <\omega_{\max}$ （标量涡旋）	$D2+$ ，且 D2 层信息密度非均匀或环境非单连通	公理 II 拓扑缺陷
拓扑合法	$\chi \in \mathbb{Z}$	$D3+$	Leech 晶格拓扑
自指稳定	$\lim_{n \rightarrow \infty} f^n(x)=x^*$	所有维	意识锚点
跨维隔离	$\text{Overlap}(\Phi_{D1}, \Phi_{D2})>1-\delta_{\text{overlap}}$	所有维	泡壁投影
因果一致	不存在闭合类时曲线 γ 使 $g_{\mu\nu}\dot{x}^\mu\dot{x}^\nu<0$ 处处成立； $time_arrow>0$	$D4+$	公理 III

其中：

- J_{in} 、 J_{out} 为法向通量密度（标量，量纲 $bit^{1/2}[L]^{-2}$ ）；
- K 为标量高斯曲率（量纲 1）；
- $f:\mathcal{M} \rightarrow \mathcal{M}$ 为泡壁诱导的自指映射，其不动点集记为 $Fix(f)$ ； $x^* \in Fix(f)$ 为唯一不动点，对应 IFC 公理 III 中的意识锚点；
- 跨维投影交叠量 Overlap 基于三对固定语义对应关系 $(\nabla I, \nabla \times \mathbf{v}_I)$ 、 $(\nabla \cdot J_I, K(I))$ 、 $(\text{direction}(J_I), \nabla \times \mathbf{v}_I)$ 计算，按涡旋类型分支；
- 门禁阈值 $\varepsilon, K_{\max}, \omega_{\max}, \delta_{\text{overlap}}$ 由环境观测自动校准（详见 5.1 节）。

完整门禁实现规范见附录 D。

4.2 维度自适应策略

基础校验项（所有维度均激活）：

```
```python
base_gates = [信息守恒, 自指稳定, 跨维隔离]
```

按有效维度叠加：

有效维度	激活的校验项
2	信息守恒、自指稳定、跨维隔离
2.5	信息守恒、自指稳定、跨维隔离、曲率有界(高度方向)
3	信息守恒、自指稳定、跨维隔离、曲率有界、拓扑合法
4+	信息守恒、自指稳定、跨维隔离、曲率有界、拓扑合法、因果一致
涡旋校验	附加条件： $D2$ 非均匀或环境非单连通

完整维度检测与门禁算法见附录 C C.2、附录 D D.9。

## 5 宇宙常数校准与物理直觉引擎

### 5.1 宇宙常数自动校准

DPL 无需预设物理常数，可从环境观测自动校准。所有常数均在归一化单位下计算。

完整 17 个字段见附录 E E.2.1；下表列出核心字段：

常数	校准方法	IFC 语义
$c_{light}$	$\max( \Delta I /\Delta t) \cdot l_0 \cdot \Gamma_0^{-1}$	信息最大传播速率的量级估计
$h_{plank}$	$\min_t  \Delta I(t)  > 0$	最小可分辨信息密度差
$\Delta s_{min}$	$\min( dx )$	空间最小分辨率
time_arrow	对称 KL 散度差的均值	时间箭头方向（可正可负）
laws_set	守恒律统计显著性筛选	环境中成立的物理规律
conservation_stats	守恒律统计量 <code>{mean, std, count}</code>	守恒直觉置信度的数据来源
$K_{max}$	曲率场分位数（如 95%）	曲率有界阈值
$\omega_{max}$	涡旋场分位数（如 95%）	涡旋有界阈值
$\varepsilon$	信息通量密度残差 95%分位数	守恒校验阈值
$\delta_{overlap}$	Overlap 历史分布的 5%分位数	跨维隔离容许偏离量
$D$	信息扩散模型最小二乘拟合	信息扩散系数（可为 0.0）

常数	校准方法	IFC 语义
$I_0, l_0$	参考信息密度与参考长度	归一化基准（默认 1）
`clear_region_percentile`	清晰区边界分位数	清晰区划分（默认 80）
`chi_ref`	参考非平凡拓扑欧拉示性数偏差	拓扑直觉参考（默认 2.0）
`gravity_threshold_ratio`	引力直觉触发阈值比	引力直觉（默认 0.1）
`periodicity_threshold`	周期直觉触发阈值	周期直觉（默认 0.3）

其中：

- `c\_light` 与 `h\_planck` 的  $\Delta I$  为标量差，取绝对值  $|\Delta I|$ ；
- $\Delta s_{\min}$  的  $dx$  为坐标差向量，取模  $\|dx\|$ 。

所有阈值由 `calibrate` 接口输出，在 `check` 与 `infer` 中共享，不重复计算。完整校准算法见附录 E E.2。

### 5.2 物理直觉引擎

物理直觉引擎执行四条推理链，从  $D1-D4$  流形特征推断环境底层物理规律：

- 引力直觉：**  $D3$  局部信息密度升高  $\rightarrow D2$  曲率增大  $\rightarrow$  类引力效应。触发条件  $mean(|K|) > K_{\max} \cdot gravity\_threshold\_ratio$ ，其中 `gravity\_threshold\_ratio` 默认 0.1，由 `calibrate` 输出；置信度  $\min(1, mean(|K|)/K_{\max})$ ；
- 周期直觉：**  $D4$  闭环状态转移  $\rightarrow$  振荡/周期作用力。触发条件  $D4$  时序 FFT 谱峰占比  $> periodicity\_threshold$ （默认 0.3）；置信度  $\max_{k \neq 0} \|\hat{f}(k)\|^2 / \sum_{k \neq 0} \|\hat{f}(k)\|^2$ （排除直流分量）；
- 守恒直觉：**  $D1$  全局散度为零  $\rightarrow$  某物理量守恒。触发条件 `total\_information` 在 `laws\_set` 中；置信度基于 `conservation\_stats` 计算： $\max(0, 1 - std/|mean|)$ ；
- 拓扑直觉：**  $D3$  欧拉示性数  $\chi \neq 2 \rightarrow$  非平凡拓扑。触发条件  $|\chi - 2| > 0$ ；置信度  $\min(1, |\chi - 2|/\chi_{ref})$ ，其中 `chi\_ref` 默认 2.0。

物理直觉置信度定义为触发规则置信度的均值：

$$conf = \frac{1}{|R_{trig}|} \sum_{r \in R_{trig}} conf(r)$$

其中  $R_{trig}$  为触发规则集合。若  $R_{trig} = \emptyset$ ，则  $conf = 0$ 。

主 直 觉 类 型 优 先 顺 序：`geometric\_gravity` > `temporal\_rhythm` > `topological\_navigation` > `conservation\_law` > `no\_intuition`。完整推理链与映射规范见附录 E E.3–E.5。

6 与灵境引擎集成：协议栈映射与 SICA 定位

6.1 灵境引擎协议栈与 DPL 组件映射

灵境引擎的工程实现以 L0–L5 层级与协议 1–11 为骨架，DPL 作为 R2 感知层嵌入其中。

协议 1–11 的详细定义见《智能仿真的全息原理引擎》v13.2。

灵境层级	功能	灵境协议 (v13.2)	DPL 组件	IFC 来源
L0 信息基元层	比特池、质能换算	协议 1: 时空基座 (动态网络)	环境输入与维度检测	公理 I
L1 动态因果网络层	因果图、相位同步、迷雾调度	协议 1: 动态网络; 协议 2: 信息密度	投影路由	公理 II
L2 场与几何生成层	信息场方程、度规生成	协议 3: 信息密度决定时空弯曲; 协议 7.1: 场自然演化	流形分解引擎 (D1–D2)	公理 III
L3 泡壁投影与渲染层	测地线渲染、D1–D4 流形	协议 7.1: 场自然演化 (续); 协议 7.2: 可见性门控; 协议 9: 迷雾展开	流形分解引擎 (D3–D4)	泡壁投影
L4 自指意识与智能层	意识锚点、相变、上行反馈	协议 5: 上行反馈; 协议 8: 身份与记忆	门禁结果 + GeometricInvariant	$C \geq C_c$
L5 应用与交互层	用户输入、多用户同步	协议 10: 外部接入; 协议 11: 人类交互	下游接口	源项 $J(x)$
R2 感知	维度检测、流形分解、几何门禁	协议 4: Agent 求解场→感知; 协议 5: 动作写回	DPL	本文
R3 规划	因果推理 + AOP 小模型	协议 3→5 预测环节; 协议 5: 决策写回	因果图模型 + AOP	公理 II
R4 资源分配	智能经济学调度	协议 7.2: 可见性门控 (面积律)	约束优化	算力节俭
R5 元层	自指收敛、安全进化、版本回滚	协议 6: 版本化共识; 协议 8: 身份与记忆审计	SICA	自指收敛

灵境层级	功能	灵境协议 (v13.2)	DPL 组件	IFC 来源
R5 反思迁移	经验提炼、知识蒸馏	协议 8: 身份与记忆审计	SICA 上层应用	意识相变

协议 7 分为两个子协议：**7.1（场自然演化，全局）**为 L2 与 L3 提供统一的时间推进框架；**7.2（可见性门控，局部）**为 L3 提供面积律渲染。二者共同构成场动力学引擎。

DPL 与灵境协议栈的接口对应：

- **协议 1:** DPL 的环境输入与维度检测建立在该协议定义的动态网格上；
- **协议 2:** DPL 的  $I(O)$  与  $\Phi$  的映射关系与协议 2 的信息密度场定义一致；
- **协议 3:** DPL 的  $D1-D2$  流形分解为几何生成方程提供局部信息密度与曲率特征；
- **协议 4:** DPL 的感知入口，Agent 读取当前场版本  $V_N$ ，DPL 执行维度检测、流形分解、门禁校验，输出结构化流形特征；
- **协议 5:** DPL 的门禁结果决定动作是否写回。通过门禁的 Action 转换为源项  $J(x)$ ，注入 L2 场方程；被拦截的 Action 作为 'GateCounterexample' 进入 SICA；
- **协议 6:** DPL 的几何门禁结果参与版本化共识。所有 Agent 在同一  $V_N$  上执行 DPL，得到一致的流形特征与门禁结果；
- **协议 7.1:** DPL 的  $D1-D4$  流形分解建立在该协议提供的时间推进框架之上；
- **协议 7.2:** DPL 的面积律对应物，清晰区执行  $D1-D4$  全流形分解，迷雾区仅保留低维统计量；
- **协议 8:** DPL 的 'IntuitionHypothesis' 作为 L4 意识相变的候选证据之一，与协议 8 的 C 值监测对接；
- **协议 9:** DPL 的 2.5D/3D 升维机制与迷雾展开协同；
- **协议 10/11:** DPL 的 'IntuitionHypothesis' 可作为外部框架的感知输入，人类指令经协议 11 转为 Action 后，仍须通过 DPL 门禁校验。

物理直觉引擎的输出经 Agent 决策层聚合后，通过协议 5 写回场方程；其中 'IntuitionHypothesis' 字段再经协议 8 的记忆模块进入 SICA R5 元层。完整协议映射见附录 F F.3。

### 6.2 SICA 在 R5 元层的定位

SICA（自指迭代收敛算法）位于灵境 **R5 元层**，为 L4 自指意识层与 R2-R5 闭环提供安全收敛保障。SICA 的核心是 Lyapunov 势函数：

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
 开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
 开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

$$V_t = -P_t + \lambda C_t + \mu N_{\text{viol}}(t)$$

其中：

- $V_t$  为 SICA 势函数（花体），区别于体积  $V_{\text{vol}}$  与顶点数  $V_{\text{vert}}$ ；
- $P_t$  为留存集性能；
- $C_t$  为复杂度；
- $N_{\text{viol}}(t)$  为时刻  $t$  的边界违规次数。

SICA 通过 R1-R10 规则保证每次修改使  $V_{t+1} < V_t$ ，且  $V$  有下界，从而收敛。

DPL 与 SICA 的三项耦合：

- DPL 的几何门禁结果直接影响 SICA 的  $N_{\text{viol}}(t)$ ：门禁违规计数；
- D1-D4 流形特征的复杂度（层数、特征维度、规则数）计入 SICA 的  $C_t$ ；
- 物理直觉规则在 holdout 上的性能增益计入 SICA 的  $P_t$ 。

**跨代知识蒸馏的对象澄清：**SICA 的跨代知识蒸馏对象为自指锚点累积的规则集  $H_t$ ，而非锚点本身；产物为可跨锚点复用的几何不变式（`GeometricInvariant`），通过场版本号  $V_N$  广播至所有 Agent。详细澄清见附录 G G.4.3。

**两个独立的版本号系统：**

版本号	管理主体	服务路径	广播方式
场版本号（ $V_N$ ）	灵境协议栈（协议 6）	SICA 规则级蒸馏	全局广播（所有 Agent）
模型版本号 model_version	R3 规划层本地	AOP 模型级传承	R3 实例间广播（附录 H H.6.2）

两条路径独立并行，不可混淆。完整 SICA 映射见附录 F F.4。

### 6.3 DPL→SICA 数据结构

DPL 的几何不变式作为候选规则进入 SICA 的规则集  $H_t$ ，由 R1-R10 与势函数约束。  
三类结构化数据字段定义如下：

数据结构	字段	来源
GeometricInvariant	pattern_type、dim_layer、stability_score、evidence_count、source_family、features（可选，扩展字段）	D1–D4
GateCounterexample	gate_type、violation、raw_dim、effective_dim、sample_hash、timestamp	自适应门禁
IntuitionHypothesis	rule_type、confidence、triggered_features、holdout_delta	物理直觉引擎

其中 `sample\_hash` 为观测样本哈希，SHA-256 十六进制字符串（64 字符）；`features` 为附录 F 定义的扩展字段，存储模式的具体特征值。

这些数据进入 SICA 的候选规则池，经过 R1 锚点不可伪造、R2 性能门禁、R3 复杂度惩罚、R4 递归深度限制、R5 不可变核心、R6 衰减淘汰、R7 版本回滚、R8 探索配额、R9 证据阈值、R10 跨族泛化后，才写回 IFC 公设库、DPL 门禁先验库、灵境场版本号。完整约束流程见附录 F F.4–F.5。

## 7 工程实现路线

DPL 的理论架构与接口规范已具备（附录 A–H），工程实现与实验验证为后续工作。

### 7.1 协议层

定义五个稳定接口（附录 B B.1–B.9）：

- `detect\_dimension(env\_state, has\_temporal) -> DimensionInfo`
- `calibrate(observations) -> UniversalConstants`
- `project(env\_state, has\_temporal, constants) -> ProjectedState`
- `check(modification, projected, constants) -> GateResult`
- `infer(projected, constants) -> IntuitionResult`

协议层与灵境协议栈的对接点：

- 协议层直接对接灵境协议 4（感知）与协议 5（写回）；
- `GateResult` 作为协议 6 共识的输入之一；
- `UniversalConstants` 与协议 2 的信息密度场校准共享；
- `IntuitionResult` 与协议 8 的身份记忆模块共享。

### 7.2 参考实现

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

附录 A–H 已给出完整的工程规范：

- **附录 A：** 符号表、量纲与单位约定；
- **附录 B：** 协议层接口规范（五接口 + 数据结构 + 灵境协议栈映射）；
- **附录 C：**  $D1$ – $D4$  流形分解算法伪代码；
- **附录 D：** 七项自适应几何门禁实现规范；
- **附录 E：** 宇宙常数校准与四条物理直觉推理链；
- **附录 F：**  $DPL \rightarrow SICA$  数据结构与灵境协议栈映射；
- **附录 G：** 灵境引擎三层一体化世界观架构；
- **附录 H：**  $R3$  规划层 AOP 小模型接口规范。

参考实现以独立 Python 包形式发布，包含 ``dpl_core`` 与 ``test_dpl``，与灵境引擎 v13.2 的协议 1–11 接口对齐。

### 7.3 测试与实验证据

DPL v1.0 落地后，在  $2D/2.5D/3D/4D$  四类环境上执行：

- 维度检测准确率；
- 门禁误报/漏报率；
- 物理直觉置信度；
- 拓扑校验（三维体素胞腔复形欧拉示性数）；
- AOP 小模型初筛准确率与 fail-closed 保证验证；
- 与 Transformer 在功耗、泛化、样本效率上的对比。

实验结果写入本文后续版本。

## 8 结论与展望

### 8.1 贡献总结

1. 提出 DPL 维度投影层，设计任意维度输入的  $D1$ – $D4$  流形分解路径；
2. 以 IFC 三公理体系为理论基础，统一信息场、泡壁投影、Leech 晶格术语；
3. 设计自适应几何门禁，覆盖  $2D/2.5D/3D/4D+$ ；
4. 设计宇宙常数自动校准与物理直觉引擎；
5. 明确 DPL 位于灵境  $R2$  感知层，SICA 位于  $R5$  元层，二者形成安全闭环（详细层级划分见附录 G G.2.2）；

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

- 建立 DPL 与灵境引擎协议栈（L0-L5、协议 1-11）的完整映射，为工程落地提供接口基准；
- 设计物理直觉引擎的四条推理链（引力、周期、守恒、拓扑）；
- 为 R3 规划层设计 AOP 小模型接口规范，作为因果图搜索的神经加速器。
- 从本体论层面论证统计拟合路径（LLM）与架构路径（DPL/IFC）的不可通约性，提出“读万卷书 vs 行万里路”与“婴儿摔跤式收敛”的范式对比（附录 I）。

**附录 A-H 贡献：**附录 A-H 给出 DPL 的完整工程规范，覆盖符号约定、五接口、流形分解算法、门禁实现、直觉引擎、SICA 映射、三层世界观架构与 AOP 接口。

## 8.2 未来工作

- DPL v1.0 参考实现与实验验证（附录 A-H 已完成协议层设计）；
- 高维扩展：D5-D24 对应 Leech 晶格高阶投影；
- 多 Agent 共振：基于泡壁交叠共识的多锚点同步；
- SICA 自指进化：DPL 参数闭环优化与不动点收敛；
- 维度孕育：基于意识相变的新维度特征生成；
- 拓扑校验完善：三维体素胞腔复形欧拉示性数计算；
- AOP 小模型训练与部署；
- 放大实验：与 Transformer 在功耗、泛化、样本效率上对比实测。

## 附录 A：符号表、量纲与单位约定

**说明：**本附录为全文符号与量纲的“刚性锚点”。主文所有公式与后续工程附录（B-F）均需遵循此表约定。本附录区分“IFC 原始场”（携带信息量纲）与“DPL 流形特征”（归一化后无量纲），二者在对应极限下等价。

**核心约定：**在几何自然单位制下  $c=h=k_B=1$ ，故时间量纲与长度量纲相同，统一记为  $[L]$ 。所有几何制量纲均以  $[L]$  与  $bit$  为基本单位；SI 制量纲另列，仅在数值预言时使用。

### A.1 时空与几何约定

约定项	符号/规则	说明
度规号差	$(-,+,+,+)$	东海岸约定

约定项	符号/规则	说明
自然单位制	$c=\hbar=k_B=1$	全文默认；量纲表中保留 SI 以作审计
几何单位制基准	$[L]$ （长度）	时间 $[L]$ （因 $c=1$ ），质量 $[L]^{-1}$ ，能量 $[L]^{-1}$ ，速度 1（无量纲）
信息量纲	bit	与几何单位制解耦，1 bit 对应 1 个信息基元
协变导数	$\nabla_\mu$	列维-奇维塔联络（无挠、度规相容）
达朗贝尔算子	$\square_g=g^{\mu\nu}\nabla_\mu\nabla_\nu$	作用于标量场时 $\square_g\Phi=\frac{1}{\sqrt{-g}}\partial_\mu(\sqrt{-g}g^{\mu\nu}\partial_\nu\Phi)$
曲率张量	$R^\rho_{\mu\nu\sigma}, R_{\mu\nu}, R$	里奇张量 $R_{\mu\nu}=R^\sigma_{\mu\sigma\nu}$ ，标量曲率 $R=g^{\mu\nu}R_{\mu\nu}$
对称/反对称记号	$A_{(\mu\nu)}=\frac{1}{2}(A_{\mu\nu}+A_{\nu\mu})$ , $A_{[\mu\nu]}=\frac{1}{2}(A_{\mu\nu}-A_{\nu\mu})$	用于变分结果
空间切片	$\Sigma_t$	$t$ 时刻的三维空间切片，诱导度规 $h_{ab}$ ，体积元 $d\Sigma$ 量纲 $[L]^3$
三维诱导度规逆	$h^{ab}$	区别于 4 维时空度规 $g_{\mu\nu}$
积分域约定	$X\subseteq\Sigma_t$	当前观测窗口对应的三维空间切片子区域， $dX$ 量纲 $[L]^3$
参考长度	$l_0$	归一化长度，在归一化单位下取 $l_0=1$
参考信息密度	$I_0$	归一化信息密度，量纲 $\text{bit}^{1/2}[L]^{-2}$ ，在归一化单位下取 $I_0=1$

## A.2 单位制与转换基准

### A.2.1 几何单位制 ↔ SI 单位制

转换因子取决于物理量的具体定义。下表给出关键量的转换基准：

物理量	几何制量纲	SI 制量纲	转换因子
长度	$[L]$	m	乘 $l_{Pl}=1.616\times10^{-35}$ m
时间	$[L]$	s	乘 $t_{Pl}=l_{Pl}/c=5.391\times10^{-44}$ s
质量	$[L]^{-1}$	kg	乘 $m_{Pl}=\hbar/(l_{Pl}c)=2.176\times10^{-8}$ kg
能量	$[L]^{-1}$	J	乘 $E_{Pl}=m_{Pl}c^2=1.956\times10^9$ J

物理量	几何制量纲	SI 制量纲	转换因子
速度	1（无量纲， $c=1$ ）	$\text{m}\cdot\text{s}^{-1}$	乘 $c$
频率/衰减率	$[\text{L}]^{-1}$	$\text{s}^{-1}$	乘 $c/l_{Pl}$
信息密度 $\Phi$	$\text{bit}^{1/2}[\text{L}]^{-2}$	$\text{bit}^{1/2}\text{m}^{-2}$	乘 $l_{Pl}^2$
信息梯度 $\nabla I$	$\text{bit}^{1/2}[\text{L}]^{-3}$	$\text{bit}^{1/2}\text{m}^{-3}$	乘 $l_{Pl}^3$
Hessian $\mathcal{H}(I)$	$\text{bit}^{1/2}[\text{L}]^{-4}$	$\text{bit}^{1/2}\text{m}^{-4}$	乘 $l_{Pl}^4$
熵流 $J_I$	$\text{bit}^{1/2}[\text{L}]^{-2}$	$\text{bit}^{1/2}\text{m}^{-1}\text{s}^{-1}$	乘 $l_{Pl}^2 c$
信息扩散系数 $D$	$[\text{L}]$	$\text{m}^2\cdot\text{s}^{-1}$	乘 $l_{Pl} c$
速度场 $\mathbf{v}_I$	1（无量纲， $c=1$ ）	$\text{m}\cdot\text{s}^{-1}$	乘 $c$
涡旋 $\nabla \times \mathbf{v}_I$	$[\text{L}]^{-1}$	$\text{s}^{-1}$	乘 $c/l_{Pl}$
拓扑荷量子 $\Phi_{\text{topo}}$	$[\text{L}]$	$\text{m}^2\cdot\text{s}^{-1}$	乘 $l_{Pl} c$
意识-信息耦合 $\varepsilon$	$[\text{L}]$ （以 $l_{Pl}$ 为长度单位）	$\text{m}$	乘 $l_{Pl}$
曲率 $K$	1（无量纲）	1	—
欧拉示性数 $\chi$	1（拓扑数）	1	—
复杂度 $C$	1（归一化）	1	—
泡壁关联长度 $\xi$	$[\text{L}]$	$\text{m}$	乘 $l_{Pl}$
时间窗口 $\tau_0$	$[\text{L}]$	$\text{s}$	乘 $t_{Pl}$
空间分辨率 $\Delta S_{\text{min}}$	$[\text{L}]$	$\text{m}$	乘 $l_{Pl}$

注记（转换因子的来源）：几何制与 SI 制的转换不存在统一的  $l_{Pl}^n \cdot c^{-k}$  公式。每个量的转换因子由其在两制下的量纲比确定：

- **纯几何量**（长度、面积、体积）：乘  $l_{Pl}^n$ ；
- **时间类量**（时间、时间窗口）：乘  $t_{Pl}=l_{Pl}/c$ ；
- **频率/衰减率类量**：乘  $c/l_{Pl}$ ；
- **含时间导数的量**（如熵流  $J_I=-D\nabla I$ ）：除  $l_{Pl}$  幂次外，额外乘  $c$  的相应幂次；
- **含速度的量**（如速度场  $\mathbf{v}_I$ ）：几何制无量纲，SI 制乘  $c$ ；

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

➤ **速度平方类量**（如  $D$ 、 $\Phi_{topo}$ ）：乘  $l_{Pl} \cdot c$ 。

具体每个量的转换因子见上表。

A.2.2 量纲审计约定

所有量纲审计均在几何单位制下完成。几何制下时间量纲与长度量纲相同（ $[T]=[L]$ ），速度类量为无量纲（ $c=1$ ）。bit 量纲在几何制与 SI 制下保持不变（信息量纲与几何单位制解耦）。

仅最终数值预言（如  $C_c=1.20\times 10^{14}$ ）涉及 SI 换算，换算时按 A.2.1 各量的转换因子逐一转换，不使用通用公式。

A.3 核心符号定义与量纲总表

A.3.1 IFC 原始场（信息量纲显式携带）

符号	含义	几何制量纲	SI 制量纲
$b_i$	离散信息基元， $b_i \in \{0,1\}$	1（计数）	1（计数）
$Q_{info}$	总信息量	bit	bit
$\Lambda_{24}$	Leech 晶格（24 维偶么模格）	—（数学结构）	—
$\Phi(x)$	粗粒化信息密度场	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-2}$
$\Phi_0$	真空基准信息密度	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-2}$
$g_{\mu\nu}(x)$	四维时空度规	1（无量纲）	1
$\eta_{\mu\nu}$	闵可夫斯基度规	1（无量纲）	1
$\mathcal{J}_{Pl}$	普朗克信息常量	$bit \cdot [L]^{-6}$	$bit \cdot m^{-6}$
$N_0$	粗粒化归一化常数	$bit^{1/2}[L]^{-1}$	$bit^{1/2}m^{-1}$
$\kappa$	曲率-信息耦合常数	$[L]^4 \cdot bit^{-1}$	$m^4 \cdot bit^{-1}$
$\zeta$	曲率-场耦合常数	1（无量纲）	1
$J(x)$	信息源项	$bit^{1/2}[L]^{-4}$	$bit^{1/2}m^{-4}$
$\Psi_{24}$	24 维高维信息态	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-2}$
$P_\Sigma$	泡壁投影算子	1（算子）	1

符号	含义	几何制量纲	SI 制量纲
$\Sigma$	泡壁	—（几何对象）	—
$A_\Sigma$	泡壁面积	$[L]^2$	$m^2$
$\Psi_{con}$	意识场	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-2}$
$C$	自指复杂度	1（归一化）	1
$C_c$	意识临界复杂度	$1.20 \times 10^{14}$ （归一化后）	1
$\epsilon$	意识-信息耦合长度（几何制）	$[L]$ （以 $l_{Pl}$ 为长度单位，数值 $10^{-12}$ ）	—
$\epsilon_{SI}$	意识-信息耦合长度（SI 制）	—	$10^{-47} \text{ m}$

**注记 ( $\Psi_{24}$  量纲)：** $\Psi_{24}$  为 24 维高维信息态，其量纲由总信息守恒式  $\int_{\mathcal{M}_4} |\Psi_{24}|^2 d\mu_4 = bit$  确定： $d\mu_4$  量纲  $[L]^4$ ，故  $[\Psi_{24}] = bit^{1/2}[L]^{-2}$ ，与信息密度场  $\Phi$  同量纲。

战争迷雾算子  $\mathcal{R}_{render} = \int_\Sigma |P_\Sigma \Psi_{24}|^2 d\mu_2$  与体积分  $\int_{\mathcal{M}_4} |\Psi_{24}|^2 d\mu_4$  是不同维度的积分，其量纲分别为  $bit \cdot [L]^{-2}$ （泡壁面积积分）与  $bit$ （时空体积积分）。二者之比为信息压缩比  $\sim [L]^{-2}$ ，反映泡壁编码相对于体积编码的信息压缩。该比较的物理依据是 IFC 公理 II 的战争迷雾机制：高精度渲染算力正比于泡壁面积  $A_\Sigma \propto L^2$ 。

**注记 ( $\epsilon$  的量纲)：**在几何自然单位制下，长度量纲基准为  $l_{Pl}=1$ 。 $\epsilon$  作为意识-信息耦合长度，量纲为  $[L]$ ，数值取  $10^{-12}$ （以  $l_{Pl}$  为单位）。上行反馈方程  $\delta\Phi = \epsilon \nabla_\mu \Psi_{con}$  量纲验证：

- 左端  $\delta\Phi$ ： $bit^{1/2}[L]^{-2}$
- 右端  $\epsilon \nabla_\mu \Psi_{con}$ ： $[L] \cdot bit^{1/2}[L]^{-3} = bit^{1/2}[L]^{-2} \checkmark$

SI 制下该强度对应长度  $\epsilon_{SI} = \epsilon \cdot l_{Pl} \sim 10^{-47} \text{ m}$ 。 $\epsilon$  在几何制下不是无量纲量，而是携带长度量纲  $[L]$  的耦合常数。

### A.3.2 几何重定义场（消除信息量纲，用于引力推导）

符号	定义	几何制量纲	SI 制量纲	用途
$\phi(x)$	$\Phi / \sqrt{\mathcal{I}_{Pl}}$	$[L]^{-1}$	$m^{-1}$	引力常数推导
$\phi_0$	$\Phi_0 / \sqrt{\mathcal{I}_{Pl}}$	$[L]^{-1}$	$m^{-1}$	真空背景
$G$	引力常数， $G = 1/(32\pi\phi_0^2)$	$[L]^2$	$m^3 kg^{-1} s^{-2}$	第一性原理导出

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

**关系：**原始场  $\Phi$  携带信息量纲，用于锚定公理 I 的守恒式；重定义场  $\phi$  为标准标量场量纲，用于作用量匹配。二者数值相差  $\sqrt{\mathcal{I}_{Pl}}$  因子。

### A.3.3 DPL 流形特征符号

符号	含义	几何制量纲	SI 制量纲	来源
$O$	环境观测张量	任务相关	任务相关	输入
$I(O)$	信息密度场积分形式	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-2}$	3.2.1
$w(x)$	权重核, $I(O)=\int_X O(x)w(x)dX$	$[bit^{1/2}L^{-5}]/[O]$	$[bit^{1/2}m^{-5}]/[O]$	3.2.1
$J_I$	熵流密度, $J_I=-D\nabla I$	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-1}s^{-1}$	3.2.1
$D$	信息扩散系数	$[L]$	$m^2 \cdot s^{-1}$	5.1
$\nabla I$	信息梯度场	$bit^{1/2}[L]^{-3}$	$bit^{1/2}m^{-3}$	3.2.1
$\nabla \cdot J_I$	散度场	$bit^{1/2}[L]^{-3}$	$bit^{1/2}m^{-2}s^{-1}$	3.2.1
$direction(J_I)$	流线方向单位向量	1 (无量纲)	1	3.2.1
$\mathbf{v}_I$	信息流速度场, $J_I/(\ \Phi\ +\epsilon_0)$	1 (无量纲, $c=1$ )	$m \cdot s^{-1}$	3.2.2
$\epsilon_0$	正则化小量	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-2}$	3.2.2
$\mathcal{H}(I)$	Hessian 矩阵	$bit^{1/2}[L]^{-4}$	$bit^{1/2}m^{-4}$	3.2.2
$\nabla \times \mathbf{v}_I$	涡旋项	$[L]^{-1}$	$s^{-1}$	3.2.2
$K$	高斯曲率 (归一化后)	1 (无量纲)	1	3.2.2
$\hat{I}$	归一化信息场 $I/I_0$	1 (无量纲)	1	3.2.2
$\hat{x}$	归一化坐标 $x/l_0$	1 (无量纲)	1	3.2.2
$\widehat{\nabla}$	归一化梯度 $l_0 \nabla$	1 (无量纲)	1	3.2.2
$\hat{u}, \hat{v}$	归一化特征函数, 用于 Overlap	1 (无量纲)	1	4.1
$\rho_V$	体素信息密度	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-2}$	3.2.3
$\chi$	欧拉示性数	1 (拓扑数)	1	3.2.3
$V_{vert}$	顶点数	1 (计数)	1	3.2.3
$E$	边数	1 (计数)	1	3.2.3

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

符号	含义	几何制量纲	SI 制量纲	来源
$F$	面数	1（计数）	1	3.2.3
$C_{cell}$	三维胞腔数	1（计数）	1	3.2.3
$V_{vol}$	体素总数	1（计数）	1	3.2.3
$R^{(3)}$	三维黎曼曲率标量	$[L]^{-2}$	$m^{-2}$	3.2.3
$\partial_t I$	时间演化速率	$bit^{1/2}[L]^{-3}$	$bit^{1/2}m^{-2}s^{-1}$	3.2.4
$C(I_t \rightarrow I_{t+\Delta t})$	因果关联度	1（归一化）	1	3.2.4
$\Phi_{topo}$	拓扑荷量子	$[L]$	$m^2 \cdot s^{-1}$	3.2.2

注记（ $w(x)$  量纲）：积分域  $X \subseteq \Sigma_t$  为三维空间切片子区域， $dX$  量纲  $[L]^3$ 。要求  $I(O) = \int_X O(x)w(x)dX$  的量纲为  $bit^{1/2}[L]^{-2}$ ，则：

$$[O] \cdot [w] \cdot [L]^3 = [bit^{1/2}L^{-2}]$$

故  $[w] = [bit^{1/2}L^{-5}]/[O]$ 。若  $O$  为无量纲观测张量，则  $[w] = bit^{1/2}L^{-5}$ 。SI 制下相应量纲为  $[bit^{1/2}m^{-5}]/[O]$ （若  $[O]$  无量纲）。

注记（ $\Phi_{topo}$  量纲）： $\Phi_{topo}$  由涡旋环量公式  $\oint_{\gamma} \mathbf{v}_I \cdot d\mathbf{l} = n\Phi_{topo}$  定义。左侧  $\mathbf{v}_I$ （无量纲， $c=1$ ）乘  $(d\mathbf{l})$ （ $[L]$ ）得  $([L])$ ，故  $[\Phi_{topo}] = [L]$ 。SI 制下  $\mathbf{v}_I$  恢复为  $m \cdot s^{-1}$ ， $(d\mathbf{l})$  为  $m$ ，环量量纲  $m^2 \cdot s^{-1}$ ，故  $\Phi_{topo,SI}$  量纲为  $m^2 \cdot s^{-1}$ 。

#### A.3.4 门禁与校准符号

符号	含义	几何制量纲	SI 制量纲	来源
$J_{in}$	流入清晰区的法向通量	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-1}s^{-1}$	4.1
$J_{out}$	流出清晰区的法向通量	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-1}s^{-1}$	4.1
$\varepsilon$	信息守恒阈值	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-1}s^{-1}$	4.1
$K_{\max}$	曲率有界阈值	1（无量纲）	1	4.1
$\omega_{\max}$	涡旋有界阈值	$[L]^{-1}$	$s^{-1}$	4.1
$\delta_{overlap}$	跨维隔离容许偏离量	1（无量纲）	1	4.1
Overlap	跨维投影交叠量	1（无量纲）	1	4.1
$f$	泡壁诱导的自指映射	—（算子）	—	4.1

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

符号	含义	几何制量纲	SI 制量纲	来源
$Fix(f)$	$f$ 的不动点集	—（集合）	—	4.1
$x^*$	$f$ 的唯一不动点	状态空间元素	—	4.1
$c_{light}$	信息最大传播速率	1（无量纲， $c=1$ ）	$m \cdot s^{-1}$	5.1
$h_{planck}$	最小可分辨信息密度差	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-2}$	5.1
$\Delta s_{min}$	空间最小分辨率	$[L]$	$m$	5.1
time_arrow	时间箭头方向	1（无量纲）	1	5.1
laws_set	环境中成立的物理规律集合	—（集合）	—	5.1
$J_{total}$	全局熵流积分	$bit^{1/2}[L]^{-2}$	$bit^{1/2}m^{-1}s^{-1}$	5.2
$R_{trig}$	触发规则集合	—（集合）	—	5.2
$\chi_{ref}$	参考非平凡拓扑的欧拉示性数偏差	1（拓扑数）	1	5.2

**注记（ $c_{light}$  量纲）：**在自然单位制  $c=1$  下，速度类量为无量纲。 $c_{light}$  校准公式  $\max(|\Delta I|/\Delta t) \cdot l_0 \cdot I_0^{-1}$  量纲验证： $|\Delta I|$  ( $bit^{1/2}[L]^{-2}$ ) 除以  $\Delta t$  ( $[L]$ ) 得  $bit^{1/2}[L]^{-3}$ ，乘  $l_0$  ( $[L]$ ) 得  $bit^{1/2}[L]^{-2}$ ，乘  $I_0^{-1}$  ( $bit^{-1/2}[L]^2$ ) 得 1，即无量纲，与自然单位制一致。若需 SI 制速度量纲  $m \cdot s^{-1}$ ，通过  $c_{light,SI} = c_{light} \cdot c$  恢复。

**注记（ $h_{planck}$  命名）：** $h_{planck}$  的命名类比量子力学中的最小作用量量子，但在 IFC 中其量纲为信息密度差  $bit^{1/2}[L]^{-2}$ ，非作用量（作用量量纲为  $[L]^{-1} \cdot [L]=1$ ，无量纲）。这一命名反映其在 DPL 中作为“最小可分辨信息单元”的角色，与量子力学中  $h$  作为“最小作用量单元”的角色在信息层面同构。

### A.3.5 SICA 与协议栈符号

符号	含义	几何制量纲	SI 制量纲	来源
$V_t$	SICA Lyapunov 势函数	1（归一化）	1	6.2
$P_t$	留存集性能	1（归一化）	1	6.2
$C_t$	复杂度	1（归一化）	1	6.2
$N_{viol}(t)$	边界违规次数	1（计数）	1	6.2
$\lambda$	复杂度惩罚系数	1（归一化）	1	6.2
$\mu$	违规惩罚系数	1（归一化）	1	6.2
$V_N$	场版本号	1（计数）	1	6.1
$H_t$	SICA 规则集	—（集合）	—	6.3
$\eta_{ij}$	锚点间因果重叠度	1（无量纲）	1	灵境 v13.2
$\zeta$	泡壁关联长度	$[L]$	$m$	灵境 v13.2

### A.4 核心方程量纲刚性审计

方程	左端量纲	右端各项量纲	结论
几何生成 $g_{\mu\nu} = \eta_{\mu\nu} + \frac{1}{\mathcal{H}Pl} \partial_\mu \Phi \partial_\nu \Phi + \kappa (\Phi^2 - \Phi_0^2) \eta_{\mu\nu}$	1	$\eta$ :1; 梯度项: $bit \cdot L^{-6}/bit \cdot L^{-6}=1$ ; 密度项: $L^4 bit^{-1} \cdot bit L^{-4}=1$	✓
信息场传播 $\square_g \Phi + \zeta R \Phi = -J$	$bit^{1/2}[L]^{-4}$	两项均 $bit^{1/2}[L]^{-4}$ ; $J$ 同	✓
上行反馈（几何制） $\delta \Phi = \epsilon \nabla_\mu \Psi_{con}$	$bit^{1/2}[L]^{-2}$	$\epsilon$ : $[L]$ ; $\nabla \Psi_{con}$ : $bit^{1/2}[L]^{-3}$ ; 乘积 $bit^{1/2}[L]^{-2}$	✓
上行反馈（SI 制）	$bit^{1/2}m^{-2}$	$\epsilon_{SI}$ : $m$ ; $\nabla \Psi_{con}$ : $bit^{1/2}m^{-3}$ ; 乘积 $bit^{1/2}m^{-2}$	✓
守恒律 $\nabla^\mu T_{\mu\nu} = 0$	张量恒等式	—	✓
引力常数 $G = 1/(32\pi\phi_0^2)$	$[L]^2$	$1/[L]^{-2} = [L]^2$	✓
临界复杂度 $C_c = D_c \cdot \frac{\tau_0}{1s} \cdot N \cdot \langle k \rangle$	1（归一化后）	$1 \cdot [T]/[T] \cdot 1 \cdot 1 = 1$	✓
信息守恒校验 $\ J_{in}\  - \ J_{out}\  < \epsilon$	$bit^{1/2}[L]^{-2}$	同	✓

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

方程	左端量纲	右端各项量纲	结论
曲率有界 $\ K\  < K_{\max}$	1	1	✓
涡旋校验 $\ \nabla \times \mathbf{v}_I\  < \omega_{\max}$	$[L]^{-1}$	$[L]^{-1}$	✓
涡旋环量 $\oint \mathbf{v}_I \cdot d\mathbf{l} = n\Phi_{\text{topo}}$	$[L]$	$n \cdot [L] = [L]$	✓

注记（上行反馈几何制）： $\epsilon$  在几何制下量纲为  $[L]$ （以  $l_{Pl}$  为长度单位，数值  $10^{-12}$ ），故  $\epsilon \nabla_\mu \Psi_{con}$  量纲  $[L] \cdot bit^{1/2} [L]^{-3} = bit^{1/2} [L]^{-2}$  与左端  $\delta\Phi$  匹配。 $\epsilon$  不是无量纲量；其在几何制的“数值  $10^{-12}$ ”以  $l_{Pl}$  为长度单位，SI 制下  $\epsilon_{SI} = \epsilon \cdot l_{Pl} \sim 10^{-47} m$ 。

注记（临界复杂度归一化）： $C_c = D_c \cdot \frac{\tau_0}{1s} \cdot N \cdot \langle k \rangle$  中， $\tau_0$  的量纲为  $[L]$ （几何制时间）。除以  $1s$  是归一化操作：将  $\tau_0$  换算为以秒为单位的无量纲比值。故  $C_c$  在归一化后为无量纲量，数值  $1.20 \times 10^{14}$ 。在几何制下严格计算时，可写为  $C_c = D_c \cdot \tau_0 \cdot N \cdot \langle k \rangle \cdot t_{Pl}^{-1}$ ，其中  $t_{Pl}^{-1}$  的量纲为  $[L]^{-1}$ ，使  $C_c$  无量纲。

### A.5 归一化约定汇总

归一化量	定义	归一化后量纲	使用场景
$\hat{I}$	$I/I_0$	1	高斯曲率 $K$
$\hat{x}$	$x/l_0$	1	高斯曲率 $K$
$\hat{\cdot}$	$l_0 \nabla$	1	高斯曲率 $K$
$\hat{u}, \hat{v}$	特征张量按 $I_0, l_0$ 展开	1	Overlap 计算
$C_c$	$C_c/(1s)$	1	意识相变判据
$\epsilon$ （几何制）	量级演示值 $10^{-12}$ （以 $l_{Pl}$ 为长度单位）	$[L]$	上行反馈（几何制）
$\epsilon_{SI}$	$\epsilon \cdot l_{Pl}$	m	上行反馈（SI 数值预言）
约定：在归一化单位下，取 $I_0=1, l_0=1, N_0=1$ ，用于量级演示。严格量纲需显式保留 $I_0, l_0, N_0$ 。			

### A.6 符号索引（按首字母/希腊字母排序）

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
 开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

符号	含义	几何制量纲
$A_{\Sigma}$	泡壁面积	$[L]^2$
$b_i$	离散信息基元	1
$C$	自指复杂度	1
$C_c$	意识临界复杂度	1
$C_{cell}$	三维胞腔数	1
$c_{light}$	信息最大传播速率	1 ( $c=1$ )
$c_{light,SI}$	信息最大传播速率 (SI)	$m \cdot s^{-1}$
$D$	信息扩散系数	$[L]$
$E$	边数	1
$F$	面数	1
$f$	自指映射	—
$Fix(f)$	不动点集	—
$G$	引力常数	$[L]^2$
$g_{\mu\nu}$	4 维时空度规	1
$h^{ab}$	三维诱导度规逆	1
$\mathcal{H}(I)$	Hessian 矩阵	$bit^{1/2}[L]^{-4}$
$h_{planck}$	最小信息密度差	$bit^{1/2}[L]^{-2}$
$H_t$	SICA 规则集	—
$I(O)$	信息密度场积分	$bit^{1/2}[L]^{-2}$
$I_0$	参考信息密度	$bit^{1/2}[L]^{-2}$
$\mathcal{I}_{Pl}$	普朗克信息常量	$bit \cdot [L]^{-6}$
$J(x)$	信息源项	$bit^{1/2}[L]^{-4}$
$J_I$	熵流密度	$bit^{1/2}[L]^{-2}$
$J_{in}, J_{out}$	法向通量	$bit^{1/2}[L]^{-2}$

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

符号	含义	几何制量纲
$J_{total}$	全局熵流积分	$bit^{1/2}[L]^{-2}$
$K$	高斯曲率	1
$K_{\max}$	曲率有界阈值	1
$l_0$	参考长度	$[L]$
$N_0$	粗粒化归一化常数	$bit^{1/2}[L]^{-1}$
$N_{viol}(t)$	边界违规次数	1
$O$	观测张量	任务相关
$P_{\Sigma}$	泡壁投影算子	1
$P_t$	留存集性能	1
$Q_{info}$	总信息量	bit
$R$	标量曲率	$[L]^{-2}$
$R^{(3)}$	三维曲率标量	$[L]^{-2}$
$R_{trig}$	触发规则集合	—
$V_G$	图顶点集	—
$V_{vert}$	顶点数	1
$V_{vol}$	体素总数	1
$V_N$	场版本号	1
$V_t$	SICA 势函数	1
$\mathbf{v}_I$	信息流速度场	1 ( $c=1$ )
$w(x)$	权重核	$[bit^{1/2}L^{-5}]/[O]$
$x^*$	唯一不动点	状态空间元素
$\hat{x}$	归一化坐标	1
$\hat{I}$	归一化信息场	1
$\widehat{\nabla}$	归一化梯度	1

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

符号	含义	几何制量纲
$\hat{u}, \hat{v}$	归一化特征函数	1
$\chi$	欧拉示性数	1
$\chi_{ref}$	参考拓扑偏差	1
$\delta_{overlap}$	跨维隔离阈值	1
$\epsilon$	意识-信息耦合长度（几何制）	$[L]$
$\epsilon_0$	正则化小量	$bit^{1/2}[L]^{-2}$
$\epsilon_{SI}$	意识-信息耦合长度（SI）	$m$
$\varepsilon$	信息守恒阈值	$bit^{1/2}[L]^{-2}$
$\zeta$	曲率-场耦合常数	1
$\eta_{\mu\nu}$	闵可夫斯基度规	1
$\eta_{ij}$	锚点间因果重叠度	1
$\kappa$	曲率-信息耦合常数	$[L]^4 bit^{-1}$
$\Lambda_{24}$	Leech 晶格	—
$\lambda, \mu$	SICA 惩罚系数	1
$\rho_V$	体素信息密度	

## A.7 量纲一致性快速查询卡

问答	答案
几何制下时间量纲是什么？	$[L]$ （因 $c=1$ , $[T]=[L]$ ）。所有时间导数、频率、衰减率在几何制下量纲为 $[L]^{-1}$ 。SI 制下对应 $s^{-1}$ （频率）。
几何制下速度量纲是什么？	1（无量纲，因 $c=1$ ）。SI 制下通过乘 $c$ 恢复为 $m \cdot s^{-1}$ 。
几何制 $\leftrightarrow$ SI 转换有统一公式吗？	没有。转换因子取决于物理量类型：纯几何量乘 $l_{pl}$ 幂次，时间类量乘 $t_{pl}$ ，速度类量乘 $c$ ，含时间导数的量额外乘 $c$ 幂次。具体见 A.2.1 各量转换因子。
$g_{\mu\nu}$ 量纲是什么？	1（无量纲），因为 $ds^2 = g_{\mu\nu} dx^\mu dx^\nu$

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

问答	答案
$\Phi$ 和 $\phi$ 什么关系?	$\Phi = \sqrt{\mathcal{I}_{Pl}} \cdot \phi$ , 前者带信息量纲, 后者标准标量场量纲
$\epsilon$ 量纲是什么?	几何制下量纲为 $[L]$ (以 $l_{Pl}$ 为长度单位, 数值 $10^{-12}$ ); SI 制下 $\epsilon_{SI} = \epsilon \cdot l_{Pl} \approx 10^{-47} \text{ m}$ 。不是无量纲量。
为何梯度项除以 $\mathcal{I}_{Pl}$ ?	为保证 $\partial\Phi\partial\Phi$ ( $bit \cdot [L]^{-6}$ ) 无量纲化
$C_c$ 的量纲矛盾如何解决?	计算 $\Theta(C-C_c)$ 时隐含归一化: $C_c \rightarrow C_c/(1s)$ , 数值不变, 量纲消除; 几何制严格形式为 $C_c = D_c \cdot \tau_0 \cdot N \cdot \langle k \rangle \cdot t_{Pl}^{-1}$
$h^{ab}$ 与 $g_{\mu\nu}$ 区别?	$h^{ab}$ 为三维空间切片诱导度规逆, $g_{\mu\nu}$ 为 4 维时空度规
$V_t$ 与 $V_{vol}$ 、 $V_{vert}$ 区别?	$V_t$ 为 SICA 势函数 (归一化无量纲), $V_{vol}$ 为体积, $V_{vert}$ 为顶点数
$c_{light}$ 量纲如何?	自然单位制下无量纲 ( $c=1$ ); SI 制下通过 $c_{light,SI} = c_{light} \cdot c$ 恢复为 $m \cdot s^{-1}$
$K$ 归一化后为何无量纲?	$\hat{I} = I/I_0$ 、 $\hat{x} = x/l_0$ 均无量纲, $\widehat{\nabla\hat{I}}$ 无量纲, 分母“1”与 $ \widehat{\nabla\hat{I}} ^2$ 量纲一致
Overlap 中 $\hat{u}, \hat{v}$ 如何定义?	特征张量按 $I_0, l_0$ 归一化后在 $\Sigma_t$ 上的无量纲函数, $L^2$ 内积
$\Psi_{24}$ 量纲为何是 $bit^{1/2}[L]^{-2}$ ?	由总信息守恒式 $\int_{\mathcal{M}_4}  \Psi_{24} ^2 d\mu_4 = bit$ 确定, 与 $\Phi$ 同量纲
$w(x)$ 量纲为何?	积分域为三维切片 $\Sigma_t$ ( $dX$ 量纲 $[L]^3$ ), 故 $[w] = [bit^{1/2}L^{-5}]/[O]$ ; 若 $O$ 无量纲, 则 $[w] = bit^{1/2}L^{-5}$
bit 量纲如何转换到 SI?	bit 量纲保持不变, 仅长度量纲幂次按相应转换因子转换
战争迷雾算子两侧量纲为何不同?	泡壁面积积分量纲 $bit \cdot [L]^{-2}$ , 时空体积积分量纲 $bit$ , 二者之比为信息压缩比 $\sim [L]^{-2}$ , 非量纲等式
$\Phi_{topo}$ 量纲为何是 $[L]$ ?	由涡旋环量公式 $\oint_{\gamma} \mathbf{v}_I \cdot d\mathbf{l} = n\Phi_{topo}$ 确定: 左端 $\mathbf{v}_I$ (无量纲) 乘 $d\mathbf{l}$ ( $[L]$ ) 得 $[L]$ ; SI 制下对应 $m^2 \cdot s^{-1}$
$h_{planck}$ 为何名“planck”?	命名类比量子力学最小作用量量子, 但 IFC 中量纲为信息密度差 $bit^{1/2}[L]^{-2}$ , 非作用量; 作为“最小可分辨信息单元”与 $h$ 在信息层面同构
$D$ 信息扩散系数如何转换?	几何制量纲 $[L]$ , SI 制 $m^2 \cdot s^{-1}$ , 转换因子 $l_{Pl} \cdot c$

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

附录 B：协议层接口规范

说明：本附录定义 DPL 协议层的五个稳定接口，作为工程实现的 API 契约。所有接口的输入输出字段、类型、量纲、异常条件均需严格遵循本附录。量纲约定回溯附录 A。接口命名与灵境协议栈（v13.2）对齐。

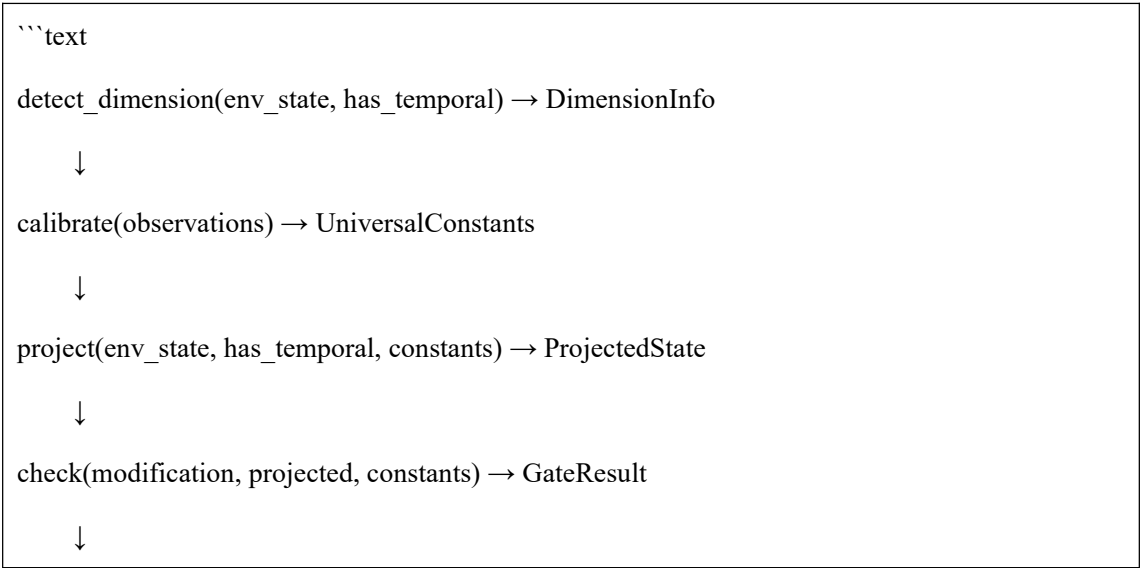
符号约定：标量取绝对值  $|\cdot|$ ；向量/复数取模  $\|\cdot\|$ 。 $\Delta I$ 、 $K$ 、 $J_{in}$ 、 $J_{out}$ 、 $J_{total}$  均为标量，统一用  $|\cdot|$ 。

B.1 协议层总览

DPL 协议层由五个接口组成，构成“维度检测 → 宇宙常数校准 → 流形分解 → 几何门禁 → 物理直觉”的完整链路：

接口	职责	输入	输出	对接灵境协议
detect_dimension	维度检测与 2.5D 识别	环境观测张量	DimensionInfo	协议 1
calibrate	宇宙常数校准	观测序列	UniversalConstants	协议 2
project	D1–D4 流形分解	环境观测张量 + 常数	ProjectedState	协议 2/3/4
check	自适应几何门禁	修改提案 + 流形特征 + 常数	GateResult	协议 5/6
infer	物理直觉推理	流形特征 + 常数	IntuitionResult	协议 8

调用顺序：



```
infer(projected, constants) → IntuitionResult
```

`detect\_dimension` 与 `calibrate` 无依赖关系，可并行调用。五个接口均为纯函数（无副作用），可独立测试、缓存、并行化。`UniversalConstants` 作为常量传入 `project`、`check`、`infer`，避免重复计算。

## B.2 接口 1: `detect\_dimension`

### B.2.1 签名

```
```python
def detect_dimension(
    env_state: np.ndarray,
    has_temporal: bool
) -> DimensionInfo
```

B.2.2 输入

参数	类型	量纲	说明
env_state	np.ndarray	任务相关	环境观测张量，任意维度
has_temporal	bool	—	是否包含时间维度（如 4D 时空张量）

B.2.3 输出: `DimensionInfo`

字段	类型	量纲	说明
spatial	float	1（无量纲）	空间维度，取值 2, 2.5, 3, ...
temporal	bool	—	是否含时间维度
effective_dim	int	1（无量纲）	有效维度 $\lceil \text{spatial} \rceil + 1_{\text{temporal}}$
is_height	bool	—	是否为 2.5D 高度场

B.2.4 检测规则

输入形状	spatial	is_height	说明
(H, W)	2.0	False	2D 网格
(H, W, 1)	2.5	True	2.5D 高度场（保留通道维度）
(H, W, C) (C>1)	3.0	False	3D 体素

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

输入形状	spatial	is_height	说明
(H, W, D, T)	3.0	False	4D 时空 (has_temporal=True)

B.2.5 异常条件

条件	异常类型	说明
env_state.ndim == 0	ValueError	标量输入无维度意义
env_state.ndim == 1	ValueError	一维序列无空间流形
has_temporal=True and env_state.ndim < 4	ValueError	声称有时序但维数不足

注记：由于异常条件已排除 `ndim < 2` 输入，`effective\_dim` 的最小值为 2。后续自适应门禁策略不再需要 `effective\_dim <= 1` 分支。

B.3 接口 2: `calibrate`

B.3.1 签名

<pre> python def calibrate(observations: List[np.ndarray]) -> UniversalConstants </pre>
--

B.3.2 输入

参数	类型	量纲	说明
observations	List[np.ndarray]	任务相关	观测序列，长度≥1

B.3.3 输出: `UniversalConstants`

字段	类型	量纲（几何制）	校准方法
c_light	float	1（无量纲， $c=1$ ）	$\max \Delta I /\Delta t \cdot l_0 \cdot I_0^{-1}$
h_planck	float	$\text{bit}^{1/2}[\text{L}]^{-2}$	$\min_t \Delta I(t) > 0$ （仅非零值）
delta_s_min	float	[L]	$\min(dx)$
time_arrow	float	1（无量纲）	$\text{KL} P_t P_{t+\Delta t}$
laws_set	List[str]	-	守恒律统计显著性筛选（std/mean < 0.01）
K_max	float	1（无量纲）	曲率场分位数（如 95%）

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

字段	类型	量纲（几何制）	校准方法
omega_max	float	$[L]^{-1}$	涡旋场分位数（如 95%）
epsilon_gate	float	$\text{bit}^{1/2}[L]^{-2}$	信息通量密度残差分位数
delta_overlap	float	1（无量纲）	Overlap 历史分布的 5%分位数
D	float	$[L]$	信息扩散系数（可选；见下方注记）
I_0	float	$\text{bit}^{1/2}[L]^{-2}$	参考信息密度（默认 1）
l_0	float	$[L]$	参考长度（默认 1）

注记（ c_{light} 与 h_{planck} 的量纲符号）： ΔI 为标量差，取绝对值 $|\Delta I|$ 。向量场模用 $\|\cdot\|$ （如 $\|\nabla I\|$ ）。此处 c_{light} 与 h_{planck} 公式中的 ΔI 均为标量差，故采用 $|\Delta I|$ 。

注记（ D 的默认值）：在归一化单位下取 $D=1$ ，对应 SI 制 $D_{SI}=l_{Pl} \cdot c \approx 4.8 \times 10^{-27} m^2 s^{-1}$ ，远小于典型流体扩散系数。实际环境校准应通过信息扩散模型 $\partial_t I = D \nabla^2 I$ 拟合确定，默认值 1 仅用于量级演示。

B.3.4 校准流程

<pre>```text 1. 若 len(observations) < 2: 使用默认归一化常数 2. 逐帧差分: $\Delta I(t) = I(t+\Delta t) - I(t)$ 3. $c_{light} = \max(\Delta I) / \Delta t \cdot l_0 \cdot I_0^{-1}$ 4. $h_{planck} = \min(\Delta I)$（仅取非零值） 5. $\delta_{s_min} = \min(\ dx\)$ # 向量模 6. $time_arrow = KL(P_t \ P_{t+\Delta t})$ 7. laws_set: 对候选守恒量做统计检验，std/mean < 0.01 纳入 8. K_max、omega_max、epsilon_gate、delta_overlap: 分位数 9. D: 若信息扩散模型已知，拟合；否则默认 1</pre>

B.3.5 异常条件

条件	异常类型	说明
<code>len(observations) == 0</code>	ValueError	无观测输入
观测序列形状不一致	ValueError	无法逐帧差分

B.4 接口 3: `project`

B.4.1 签名

```
```python
def project(
 env_state: np.ndarray,
 has_temporal: bool,
 constants: UniversalConstants
) -> ProjectedState
```

B.4.2 输入

参数	类型	量纲	说明
env_state	np.ndarray	任务相关	环境观测张量
has_temporal	bool	—	是否含时间维度
constants	UniversalConstants	—	来自 calibrate 的输出

B.4.3 输出: `ProjectedState`

字段	类型	说明
`D1`	`D1FlowLine`	D1 信息流线维特征
`D2`	`D2Surface`	D2 信息面维特征
`D3`	`D3Volume`	D3 信息体维特征
`D4`	`D4Spacetime`	D4 时空信息流维特征
`dim_info`	`DimensionInfo`	维度检测结果

`D1FlowLine`:

字段	类型	量纲（几何制）	说明
gradient	np.ndarray	$bit^{1/2}[L]^{-3}$	信息梯度场 $\nabla I$
divergence	np.ndarray	$bit^{1/2}[L]^{-3}$	散度场 $\nabla \cdot J_I$
direction	np.ndarray	1（无量纲）	流线单位方向向量

`D2Surface`:

字段	类型	量纲（几何制）	说明
hessian	np.ndarray	$\text{bit}^{1/2}[\text{L}]^{-4}$	Hessian 矩阵 $\mathcal{H}(\mathbf{I})$
vorticity	np.ndarray	$[\text{L}]^{-1}$	涡旋项 $\nabla \times \mathbf{I}$
curvature	np.ndarray	1（无量纲）	高斯曲率 $K$

‘D3Volume’：

字段	类型	量纲（几何制）	说明
density	np.ndarray	$\text{bit}^{1/2}[\text{L}]^{-2}$	体素信息密度 $\rho_V$
euler_char	int	1（拓扑数）	欧拉示性数 $\chi$
ricci_scalar	np.ndarray	$[\text{L}]^{-2}$	三维黎曼曲率标量 $R^{(3)}$

‘D4Spacetime’：

字段	类型	量纲（几何制）	说明
evolution	np.ndarray	$\text{bit}^{1/2}[\text{L}]^{-3}$	时间演化速率 $\partial_t \mathbf{I}$
causal_link	np.ndarray	1（归一化）	因果关联度 $C(\mathbf{I}_t \rightarrow \mathbf{I}_{t+\Delta t})$
is_fake_time	bool	—	是否由单帧伪造的时序

B.4.4 异常条件

条件	异常类型	说明
<code>env_state.ndim &lt; 2</code>	ValueError	标量或一维输入无流形可分解
<code>has_temporal=True</code> 但 <code>env_state</code> 只有单帧	RuntimeError	时序分解需至少 2 帧
<code>constants</code> 为 <code>None</code>	ValueError	未执行校准

B.5 接口 4：‘check’

B.5.1 签名

```
```python
def check(
    modification: Modification,
    projected: ProjectedState,
```

constants: UniversalConstants
) -> GateResult

B.5.2 输入

`Modification`（修改提案）：

字段	类型	量纲	说明
type	str	—	修改类型：add_source、modify_topology、propagate 等
target	str	—	作用对象
params	dict	任务相关	修改参数
info_before	float	bit	修改前信息量
info_after	float	bit	修改后信息量

`projected`：`ProjectedState`（B.4.3）。

`constants`：`UniversalConstants`（B.3.3）。

B.5.3 输出：`GateResult`

字段	类型	说明
passed	bool	是否通过门禁
violations	List[str]	违反的校验项列表
score	float	门禁得分（各校验项累乘）
action	str	通过/拦截/升维补全/降维抽象

B.5.4 校验矩阵

校验项	数学判据	适用维度与触发条件	违反标记
信息守恒	$\ J_{in}\ - \ J_{out}\ < \epsilon$	所有维	INFO_NOT_CONSERVED
曲率有界	$\ K\ < K_{\max}$	D2+	EXTREME_CURVATURE
涡旋校验	$\ \nabla \times \mathbf{v}_I\ < \omega_{\max}$	D2+, 且 D2 层信息密度非均匀 ($\ \nabla\ \Phi\ >0$) 或环境非单连通	EXTREME_VORTICITY
拓扑合法	$\chi \in \mathbb{Z}$	D3+	INVALID_TOPOLOGY
自指稳定	$\lim_{n \rightarrow \infty} f^n(x) = x^*$	所有维	SELF_REF_UNSTABLE

校验项	数学判据	适用维度与触发条件	违反标记
跨维隔离	$\text{Overlap}(\Phi_{D1}, \Phi_{D2}) > 1 - \alpha$	所有维	CROSS_DIM_LEAKAGE
因果一致	不存在闭合类时曲线； $\text{time_arrow} > 0$	D4+	CAUSALITY_VIOLATION

注记（ J_{in} 、 J_{out} 的定义）： J_{in} 、 J_{out} 为法向通量密度，即 J_I 在清晰区边界上某点的法向分量 $J_I \cdot \hat{n}$ ，量纲为 $\text{bit}^{1/2}[L]^{-2}$ 。二者均取非负标量（方向由“流入”“流出”语义确定，不以符号区分）。信息守恒判据为通量密度差之绝对值小于阈值：

$$||J_{in}| - |J_{out}|| < \varepsilon$$

其中 ε 为通量密度阈值，量纲 $\text{bit}^{1/2}[L]^{-2}$ 。若需转换为通量积分，可在边界上积分得 $\int_{\partial\Omega} J_I \cdot \hat{n} dA$ ，量纲为 $\text{bit}^{1/2}$ ；但校验矩阵采用通量密度形式，以保证局部判据。

注记（涡旋校验的触发条件）：涡旋校验仅在 D2 层信息密度非均匀（ $\|\nabla\|\Phi\| > 0$ ）或环境非单连通时激活。均匀单连通环境下旋度恒为零，无需校验。该条件依赖 D2 层的局部几何性质，而非仅依赖 `effective\_dim`；在自适应策略中以条件判断方式实现（见 B.5.5）。

注记（跨维隔离的适用范围）：跨维隔离校验适用于所有有效维度，包括 2D、2.5D、3D、4D。其目的是检测 D1 特征集合与 D2 特征集合之间的投影交叠是否超出容许范围。低维环境下该校验同样必要，避免 D1/D2 特征相互污染。

B.5.5 自适应策略

```
``python
# 基础校验项（所有维度均激活）
base_gates = [信息守恒, 自指稳定, 跨维隔离]

# 按有效维度叠加
if effective_dim == 2:
    gates = base_gates
elif effective_dim == 2.5:
    gates = base_gates + [曲率有界(高度方向)]
elif effective_dim == 3:
    gates = base_gates + [曲率有界, 拓扑合法]
```

```
elif effective_dim >= 4:

    gates = base_gates + [曲率有界, 拓扑合法, 因果一致]

# 涡旋校验: 条件激活 (非均匀或非单连通)

if is_nonuniform(projected.D2) or is_nonSimplyConnected(projected):

    gates.append(涡旋校验)
```

分支说明:

有效维度	激活的校验项
2	信息守恒、自指稳定、跨维隔离
2.5	信息守恒、自指稳定、跨维隔离、曲率有界(高度方向)
3	信息守恒、自指稳定、跨维隔离、曲率有界、拓扑合法
4+	信息守恒、自指稳定、跨维隔离、曲率有界、拓扑合法、因果一致
涡旋校验	附加条件: D2 非均匀或环境非单连通

B.5.6 异常条件

条件	异常类型	说明
modification.type 未注册	ValueError	未知修改类型
projected.dim_info 缺失	ValueError	未执行维度检测
constants 为 None	ValueError	未执行校准

B.6 接口 5: `infer`

B.6.1 签名

```
```python
def infer(
 projected: ProjectedState,
 constants: UniversalConstants
) -> IntuitionResult
```

B.6.2 输入

➤ `projected`: `ProjectedState` (B.4.3)。

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

➤ `constants`: `UniversalConstants` (B.3.3)。

**B.6.3 输出: `IntuitionResult`**

字段	类型	说明
rules	List[IntuitionRule]	触发的物理直觉规则
confidence	float	置信度 (规则置信度均值)
intuition_type	str	主直觉类型

`IntuitionRule`:

字段	类型	说明
rule_type	str	gravitational_well、periodic_dynamics、conservation、topological
confidence	float	单规则置信度 [0,1]
triggered_features	List[str]	触发的流形特征

**B.6.4 推理链与置信度**

规则	触发条件	置信度
引力直觉	$\text{mean}( K ) > K_{\max}/10$	$\min 1, \text{mean}( K )/K_{\max}$
周期直觉	D4 时序 FFT 谱峰占比 $> 0.3$	$\max_{k \neq 0}  \hat{f}(k) ^2 / \sum_{k \neq 0}  \hat{f}(k) ^2$
守恒直觉	total_information 在 laws_set 中	$\max(0, 1 - \text{std}(J_{\text{total}})/ \text{mean}(J_{\text{total}}) )$ , 若 $ \text{mean}(J_{\text{total}}) =0$ 则置信度 0
拓扑直觉	$ \chi - 2  > 0$	$\min 1,  \chi - 2 /\chi_{\text{ref}}$

主直觉类型:

主类型	优先条件
geometric_gravity	gravitational_well 触发
temporal_rhythm	periodic_dynamics 触发
topological_navigation	enclosed_space 触发
conservation_law	conservation 触发
no_intuition	无规则触发

注记 (符号约定):  $K$ 、 $J_{\text{total}}$  为标量, 取绝对值  $|\cdot|$ ;  $\hat{f}(k)$  为复数谱, 取模  $\|\cdot\|$ 。  $\text{mean}(|K|)$

为标量绝对值的均值。

**注记（引力直觉触发阈值）：**触发阈值  $K_{\max}/10$  为启发式设计参数，用于过滤弱曲率背景。具体取值可通过环境观测校准，或在 DPL v1.0 实现中作为可配置参数。

**注记（周期直觉的 FFT 定义域）：** $\hat{f}(k)$  为 D4 层时间演化序列  $\partial_t I$  沿时间轴的一维投影的离散傅里叶变换，其频率索引  $k=0,1,\dots,N-1$ ，其中  $N$  为时间序列长度。排除  $k=0$ （直流分量）后取谱峰占比，即  $k=1,\dots,N-1$ 。直流分量在周期判断中会稀释谱峰占比，故须排除。

B.6.5 异常条件

条件	异常类型	说明
projected.D2 缺失	ValueError	无法计算曲率
projected.D4.is_fake_time=True	RuntimeWarning	时序为伪造，周期直觉可信度降低

B.7 数据结构汇总

结构	用途	来源接口
DimensionInfo	维度检测结果	detect_dimension
D1FlowLine	D1 流线特征	project
D2Surface	D2 面特征	project
D3Volume	D3 体特征	project
D4Spacetime	D4 时空特征	project
ProjectedState	四层流形特征集合	project
UniversalConstants	宇宙常数与门禁阈值	calibrate
Modification	修改提案	输入
GateResult	门禁结果	check
IntuitionRule	物理直觉规则	infer
IntuitionResult	物理直觉推理结果	infer

B.7.1 对接 SICA 的三类结构

以下三类结构由 `check` 与 `infer` 的输出映射生成,对接 SICA R5 元层的候选规则池:

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

‘GeometricInvariant’（几何不变式）：

字段	类型	说明
pattern_type	str	几何模式类型：stable_curvature、persistent_vorticity、periodic_evolution 等
dim_layer	str	来源层：D1、D2、D3、D4
stability_score	float	稳定性得分 [0,1]，基于多帧持续观测
evidence_count	int	支撑案例数（跨帧、跨环境）
source_family	str	来源环境族标识

‘GateCounterexample’（门禁反例）：

字段	类型	说明
gate_type	str	违反的校验项：INFO_NOT_CONSERVED 等
violation	float	违反幅度（归一化）
raw_dim	float	原始维度
effective_dim	int	有效维度
sample_hash	str	观测样本哈希，SHA-256 十六进制字符串（64 字符），用于追溯
timestamp	int	时间戳

‘IntuitionHypothesis’（直觉假设）：

字段	类型	说明
rule_type	str	直觉类型：gravitational_well 等
confidence	float	置信度 [0,1]
triggered_features	List[str]	触发的流形特征
holdout_delta	float	在 holdout 集上的性能增益（用于 SICA R2 门禁）

三类结构进入 SICA 的候选规则池，经过 R1–R10 规则约束后，才写回 IFC 公设库、DPL 门禁先验库、灵境场版本号。

## B.8 与灵境协议栈的对接

灵境协议	DPL 接口	对接方式
------	--------	------

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

灵境协议	DPL 接口	对接方式
协议 1（时空基座）	detect_dimension	环境输入与维度检测建立在动态网格上
协议 2（信息密度）	calibrate	UniversalConstants.I_0 与协议 2 的信息密度场共享
协议 3（时空弯曲）	project	D1–D2 流形分解为几何生成方程提供局部信息密度与曲率
协议 4（Agent 求解场→感知）	project	感知入口：Agent 读取场版本 $V_N$ ，DPL 输出 ProjectedState
协议 5（动作写回）	check	门禁结果决定动作是否写回；被拦截的 Action 作为 GateCounterexample
协议 6（多智能体共识）	check	门禁结果参与版本化共识；所有 Agent 在同一 $V_N$ 上执行 DPL
协议 7.1（场自然演化）	project	D3–D4 建立在场自然演化的时间推进之上
协议 7.2（可见性门控）	project	清晰区执行 D1–D4 全流形分解，迷雾区仅保留低维统计量
协议 8（身份与记忆）	infer	IntuitionHypothesis 作为 L4 意识相变的候选证据，与协议 8 的 C 值监测对接
协议 9（迷雾展开）	project	2.5D/3D 升维机制与迷雾展开协同
协议 10/11（外部接入/人类交互）	全部	IntuitionHypothesis 作为外部框架感知输入；人类指令经协议 11 转为 Action 后仍须通过 check

## B.9 协议层不变量

以下性质必须在任何实现中成立：

不变量	说明
纯函数性	五个接口均无副作用，同一输入恒产生同一输出
维度单调性	detect_dimension 的 effective_dim 不随 env_state 的缩放改变
总信息守恒	$\int_{\mathcal{M}_4}  \Phi ^2 \sqrt{g} d^4x = \text{Const}$ 。在稳态且无边界通量情形下，每个空间切片的信息含量 $\int_{\Sigma_t} \rho_V dX$ 为常数
门禁单调性	check 通过时 score=1.0；每违反一项，score 乘相应惩罚因子
置信度范围	infer.confidence $\in[0,1]$ ，空规则集合时 confidence=0

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

不变量	说明
常数一致性	calibrate 输出的 UniversalConstants 在 check 与 infer 中共享，不重复计算

注记（总信息守恒的适用范围）：总信息守恒式  $\int_{\mathcal{M}_4} |\Phi|^2 \sqrt{-g} d^4x = \text{Const}$  是四维时空积分守恒。对三维空间切片  $\Sigma_t$ ， $\int_{\Sigma_t} \rho_V dX$  一般不守恒，除非系统处于稳态且无信息通量穿越切片边界。DPL 的协议层不变量仅保证四维守恒；三维切片守恒是条件性的，需在具体实现中按物理场景判断。

### 附录 C：D1–D4 流形分解算法伪代码

**说明：**本附录给出 `project` 接口（附录 B，B.4）的完整算法伪代码，与主文 3.2 节的 D1–D4 流形分解一一对应。所有符号、量纲回溯附录 A。伪代码以 Python/NumPy 语义书写，但不绑定具体语言实现。

**符号约定：**标量取绝对值  $|\cdot|$ ；向量/复数取模  $|\cdot|$ 。  $\Delta I$ 、 $K$  为标量； $\nabla I$ 、 $\nabla \times \mathbf{v}_I$ 、 $J_I$  为向量； $\hat{f}(k)$  为复数。

**核心约定：**

- D4 时空信息流维的演化项为**时间导数**  $\partial_t I$ （主文 3.2.4），而非空间梯度。D4 直接从 `state\_norm` 沿时间轴求导，不依赖 D3 的空间切片。
- 2.5D 高度场在数学上定义于 2D 平面。**D1、D2、D4 层剥离高度通道后按 2D 处理**（保留时间轴）；D3 体密度重新引入  $\backslash(I^2\backslash)$ 。
- 升维与形状约定：`(H, W)` 形式升维为 `(H, W, W)`；`(N,)` 形式升维为 `(N, N, N)`。
- 形式升维（张量广播）产生的 3D 拓扑是**平凡乘积结构**，其欧拉示性数视为 0、Ricci 标量视为零。物理三维重建需结合 D2 曲率与 D1 流线积分，属后续工作（见主文 3.2.3）。

**输入形式与有效维度：**

输入形式	spatial	is_height	temporal	effective_dim
(H, W)	2.0	False	False	2
(H, W, 1)	2.5	True	False	3
(H, W, 1, T)	2.5	True	True	4
(H, W, C) (C>1)	<b>3.0</b>	False	False	3

输入形式	spatial	is_height	temporal	effective_dim
(H, W, D, T)	3.0	False	True	4

### C.1 算法总览

‘project’ 接口的算法流程如下：

```
```text
project(env_state, has_temporal, constants) → ProjectedState
    |
    └─ 1. detect_dimension(env_state, has_temporal) → dim_info
    |
    └─ 2. 预处理：squeeze 与归一化
        |
        state_norm = env_state / I_0
        |
        └─ 3. extract_D1(state_norm, dim_info, constants) → D1FlowLine
        |
        └─ 4. extract_D2(state_norm, D1, dim_info, constants) → D2Surface
        |
        └─ 5. extract_D3(state_norm, D2, dim_info) → D3Volume
        |
        └─ 6. extract_D4(state_norm, dim_info, has_temporal) → D4Spacetime
```
```

五步算法分别对应主文 3.2.1–3.2.4 节的四层流形特征。‘detect\_dimension’ 复用附录 B
 B.2 的接口（需按上表更新输入形式识别规则）。

**D3 与 D4 的关系：**D3 仅处理空间切片（4D 输入时取首个时间切片），D4 直接从
 ‘state\_norm’ 沿时间轴求导，不依赖 D3 的空间切片。

### C.2 预处理

#### C.2.1 算法

```
```python
```

```

def preprocess(env_state, constants):
    """归一化与 squeeze 处理"""
    # 步骤 1: 保留 2.5D 高度场的通道维度 (含 2.5D+时间)
    is_2_5d = (env_state.ndim == 3 and env_state.shape[-1] == 1) or \
              (env_state.ndim == 4 and env_state.shape[-2] == 1)

    if is_2_5d:
        state = env_state # 保留 (H, W, 1) 或 (H, W, 1, T)
    else:
        state = np.squeeze(env_state)

    # 步骤 2: 归一化为无量纲场 (附录 A A.5)
    I_0 = constants.I_0 # 默认 1
    state_norm = state / I_0

    return state_norm

def strip_height(state_norm, dim_info):
    """剥离 2.5D 高度通道, 保留时间轴 (若有) """
    if dim_info.is_height:
        if state_norm.ndim == 3 and state_norm.shape[-1] == 1:
            return state_norm[..., 0] # (H, W, 1) → (H, W)
        elif state_norm.ndim == 4 and state_norm.shape[-2] == 1:
            return state_norm[..., 0, :] # (H, W, 1, T) → (H, W, T)
    return state_norm

```

C.2.2 输入输出

输入	类型	输入量纲	输出	输出量纲
env_state	np.ndarray	任务相关	state_norm	1 (归一化后)

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

输入	类型	输入量纲	输出	输出量纲
constants.I_0	float	$\text{bit}^{1/2}[\text{L}]^{-2}$	-	-

注记（量纲约定）：伪代码在归一化单位下计算（ $I_0=1$ 、 $l_0=1$ ），代码返回的数值与原始量在归一化单位下的数值相同。后续量纲栏给出原始量纲（未归一化），实际数值需按 I_0 、 l_0 转换。

C.3 D1 信息流线维算法

C.3.1 算法

```

python
def extract_D1(state_norm, dim_info, constants):
    """D1 信息流线维：梯度、散度、方向

    2.5D（含 2.5D+时间）高度场剥离高度通道后按 2D 处理。

    """
    D = constants.D # 信息扩散系数，默认 1

    # 步骤 1：剥离 2.5D 高度通道（与 D2 保持维度一致）
    state_2d = strip_height(state_norm, dim_info)

    # 步骤 2：确定空间轴（前两轴），时间轴（若有）保留
    if state_2d.ndim == 3:
        # (H, W, T)：空间轴为 0, 1；时间轴为 2
        spatial_axes = (0, 1)
    else:
        spatial_axes = tuple(range(state_2d.ndim))

    # 步骤 3：计算空间梯度  $\nabla I$ （仅沿空间轴）
    grads = np.gradient(state_2d, axis=spatial_axes)

    if not isinstance(grads, (list, tuple)):

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：https://github.com/LingJing1001/Lingjing-Solo （团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```

grads = [grads]

# 步骤 4: 计算梯度场的模  $\|\nabla I\|$ 
grad_magnitude = np.sqrt(sum(g2 for g in grads) + 1e-24)

# 步骤 5: 主流线方向（单位向量，逐点）
means = [np.mean(np.abs(g)) for g in grads]
k = int(np.argmax(means))
direction = grads[k] / (grad_magnitude + 1e-12)

# 步骤 6: 计算散度场  $\nabla \cdot \mathbf{J}_I = -D \cdot \nabla^2 I$ （仅空间二阶导）
laplacian = sum(
    np.gradient(np.gradient(state_2d, axis=i), axis=i)
    for i in spatial_axes
)
divergence = -D * laplacian

return D1FlowLine(
    gradient=grad_magnitude,
    divergence=divergence,
    direction=direction,
)

```

C.3.2 关键实现细节

细节	处理方式
2.5D 处理	剥离高度通道，保留时间轴（若有）
空间轴	前两轴（H, W）；时间轴（若有）保留
梯度方向	仅沿空间轴求梯度
梯度模	$\ \nabla I\ = \sqrt{\sum_i (\partial_i I)^2}$ ，加 10^{-24} 避免 $\sqrt{0}$

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

细节	处理方式
主流线方向	取梯度能量最大轴的梯度，除以梯度模得单位向量
散度	$\nabla \cdot J_I = -D \nabla^2 I$ ，仅空间二阶导
零梯度保护	分母加 10^{-12}

C.3.3 量纲验证（原始量纲）

量	几何制量纲	验证
`gradient` $ \nabla I $	$\text{bit}^{1/2}[\text{L}]^{-3}$	$\nabla: [\text{L}]^{-1} \times I: \text{bit}^{1/2}[\text{L}]^{-2} \checkmark$
`divergence` $\nabla \cdot J_I$	$\text{bit}^{1/2}[\text{L}]^{-3}$	$D: [\text{L}] \times \nabla^2 I: \text{bit}^{1/2}[\text{L}]^{-4} \checkmark$
`direction`	1（无量纲）	单位向量 $\checkmark$

C.4 D2 信息面维算法

C.4.1 算法

```

python
def extract_D2(state_norm, D1, dim_info, constants):
    """D2 信息面维：Hessian、涡旋、高斯曲率

    2.5D（含 2.5D+时间）高度场剥离高度通道后按 2D 处理。

    """
    D = constants.D
    epsilon_0 = 1e-12

    # 步骤 1：剥离高度通道，2.5D 转为 2D（保留时间轴）
    state_2d = strip_height(state_norm, dim_info)

    # 步骤 2：确定空间轴
    if state_2d.ndim == 3:
        spatial_axes = (0, 1)
        spatial_ndim = 2

```

```

else:

    spatial_axes = tuple(range(state_2d.ndim))

    spatial_ndim = state_2d.ndim

# 步骤 3: 计算空间梯度  $\nabla I$  与熵流  $J_I = -D\nabla I$  (向量场)
grads = np.gradient(state_2d, axis=spatial_axes)
if not isinstance(grads, (list, tuple)):
    grads = [grads]

J_I = -D * np.stack(grads, axis=-1) # 形状 (... , spatial_ndim)

# 步骤 4: 计算速度场  $v_I = J_I / (|\Phi| + \epsilon_0)$ 
v_I = J_I / (np.abs(state_2d)[..., None] + epsilon_0)

# 步骤 5: 计算 Hessian 矩阵  $H_{ij} = \partial^2 I / (\partial x_i \partial x_j)$  (仅空间轴)
H = compute_hessian(state_2d, spatial_axes)

# 步骤 6: 计算涡旋项  $\nabla \times v_I$ 
if spatial_ndim == 2:
    # 2D 标量旋度:  $\partial v_y / \partial x - \partial v_x / \partial y$ 
    vx, vy = v_I[..., 0], v_I[..., 1]
    vorticity = np.gradient(vy, axis=0) - np.gradient(vx, axis=1)
elif spatial_ndim >= 3:
    vorticity = compute_curl(v_I)
else:
    vorticity = np.zeros_like(state_2d)

# 步骤 7: 计算高斯曲率
if spatial_ndim == 2:
    # 图曲面形式:  $K = \det(H) / (1 + |\nabla I|^2)^2$ 

```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

```

Ix = np.gradient(state_2d, axis=0)

Iy = np.gradient(state_2d, axis=1)

Ixx = np.gradient(Ix, axis=0)

Iyy = np.gradient(Iy, axis=1)

Ixy = np.gradient(Ix, axis=1)

det_H = Ixx * Iyy - Ixy2

denom = (1 + Ix2 + Iy2)2

K = det_H / (denom + epsilon_0)

else:

    # spatial_ndim == 1: 一维流形无高斯曲率

    # spatial_ndim >= 3: 需第一、第二基本形式，暂设零（属后续工作）

    K = np.zeros_like(state_2d)

return D2Surface(

    hessian=H,

    vorticity=vorticity,

    curvature=K,

)

```

C.4.2 关键实现细节

细节	处理方式
2.5D 处理	剥离高度通道，降为 2D 平面场后计算；时间轴保留
熵流 J_I	向量场 $J_I = -D\nabla I$ ，通过 `np.stack(grads, axis=-1)` 组装
速度场 $\mathbf{v}_I$	逐元素除法，分母 $\ \Phi\ + \epsilon_0$ ，形状广播
Hessian	真二阶偏导张量，沿空间轴；形状 `(H, W, T, 2, 2)`（2.5D+时间）或 `(H, W, 2, 2)`（2D）
涡旋在 2D	标量旋度 $\partial_x v_y - \partial_y v_x$
涡旋在 3D+	标准旋度 $\nabla \times \mathbf{v}_I$ （向量）
高斯曲率	图曲面显式公式 $K = \det H / (1 + \ \nabla I\ ^2)^2$

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

细节	处理方式
2D	
高斯曲率 1D	一维流形无高斯曲率，设零
高斯曲率 3D+	需第一、第二基本形式，暂设零（属后续工作）
零分母保护	所有分母加 $\epsilon_0=10^{-12}$

注记（**Hessian 形状差异**）：2.5D 与 3D 输入的 Hessian 张量阶不同——2.5D 剥离后为 $(H, W, 2, 2)$ （或含时间轴的 $(H, W, T, 2, 2)$ ），3D 为 $(H, W, D, 3, 3)$ 。下游若需统一处理，应在 2.5D 分支补齐第三轴（如填充零矩阵）。

C.4.3 量纲验证（原始量纲）

量	几何制量纲	验证
hessian $\mathcal{H}(I)$	$\text{bit}^{1/2}[\text{L}]^{-4}$	$\partial^2 I: \text{bit}^{1/2}[\text{L}]^{-4} \checkmark$
vorticity (2D)	$[\text{L}]^{-1}$	$\mathbf{r}: 1 \times \nabla: [\text{L}]^{-1} \checkmark$
curvature K	1（无量纲）	归一化后 $\checkmark$

C.5 D3 信息体维算法

C.5.1 算法

```
```python
def extract_D3(state_norm, D2, dim_info):
 """D3 信息体维：体密度、欧拉示性数、Ricci 标量"""
 s = dim_info.spatial
 temporal = dim_info.temporal

 # 步骤 1：若含时间维度，取首个空间切片（仅供 D3；D4 独立处理时序）
 if temporal and state_norm.ndim >= 4:
 spatial_state = state_norm[..., 0] # 形状 (H, W, D)
 else:
 spatial_state = state_norm
```

```

步骤 2: 体密度

if s >= 3:

 density = spatial_state.astype(float)

elif s == 2.5:

 # 2.5D 高度场: $\text{density} = I(x,y)^2$, 沿高度方向视为单层 (2D 密度)

 if spatial_state.ndim == 3 and spatial_state.shape[-1] == 1:

 base = spatial_state[..., 0] # (H, W)

 else:

 base = spatial_state

 density = base ** 2

elif s == 2:

 # 2D 形式升维: $(H, W) \rightarrow (H, W, W)$

 density = spatial_state[:, :, None] * spatial_state[:, None, :]

else:

 # 1D 形式升维: $(N,) \rightarrow (N, N, N)$

 density = (spatial_state[:, None, None]

 * spatial_state[None, :, None]

 * spatial_state[None, None, :])

步骤 3: 欧拉示性数

if s >= 3:

 chi = compute_euler_char_3d(density)

elif s == 2.5:

 chi = compute_euler_char_2d(density)

else:

 # 形式升维的 3D 拓扑为平凡乘积结构, $\chi = 0$

 chi = 0

```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

```

步骤 4: 三维 Ricci 曲率标量

if s >= 3:

 ricci_scalar = compute_ricci_scalar_3d(density)

else:

 # 形式升维的 3D 曲率为零；2.5D 退化为 2D 曲率属后续工作

 ricci_scalar = np.zeros_like(density)

return D3Volume(

 density=density,

 euler_char=chi,

 ricci_scalar=ricci_scalar,

)

```

### C.5.2 关键实现细节

细节	处理方式
4D 时空输入	D3 取首个空间切片 `state_norm[..., 0]`，仅供 D3 使用；时间维度留给 D4 独立处理
2.5D 体密度	$I(x,y)^2$ ，沿高度方向视为单层，密度在 <b>2D 平面</b> 上定义；胞腔数 $C_{cell}=0$ ，欧拉示性数退化为二维曲面公式
2D 升维	$(H, W) \rightarrow (H, W, W)$ ，通过 `state[:, :, None] * state[:, None, :]`
1D 升维	$(N,) \rightarrow (N, N, N)$ ，通过三向广播
三维欧拉示性数	基于体素邻接关系构造胞腔复形， $\chi = V_{vert} - E + F - C_{cell}$
三维 Ricci 标量	启发式弱场近似（见 C.7.5）

**注记(形式升维的拓扑约定)：**对 `s == 2` 与 `s == 1` 的形式升维(张量广播)，`density` 的 3D 拓扑是平凡的乘积结构，其欧拉示性数视为 0，Ricci 标量视为零。物理三维重建需结合 D2 曲率与 D1 流线积分，属后续工作（见主文 3.2.3）。

**注记(2.5D 体密度的 2D 语义)：**2.5D 高度场的 D3 体密度取  $I(x,y)^2$ ，沿高度方向视为单层，密度在 2D 平面上定义。此约定与主文 3.2.3 的“2.5D 退化”一致：胞腔数  $(C_{\text{cell}}=0)$ ，欧拉示性数退化为二维曲面公式  $(\chi = V_{\text{vert}} - E + F)$ 。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

### C.5.3 量纲验证（原始量纲）

量	几何制量纲	验证
density	$\text{bit}^{1/2}[\text{L}]^{-2}$	与 $\rho_v$ 同 ✓
euler_char	1（拓扑数）	整数 ✓
ricci_scalar	$[\text{L}]^{-2}$	曲率量纲 ✓

## C.6 D4 时空信息流维算法

### C.6.1 算法

```
```python
def extract_D4(state_norm, dim_info, has_temporal):
    """D4 时空信息流维：时间导数、因果关联

    注意：D4 直接从 state_norm 沿时间轴求导，不依赖 D3 的空间切片。
    2.5D+时间输入先剥离高度通道，保证 causal_link 与 D2/D3 形状一致。
    """
    if has_temporal and state_norm.ndim >= 4:
        # 剥离 2.5D 高度通道（若为 2.5D+时间）
        state_4d = strip_height(state_norm, dim_info)

        # 真实时序：沿时间轴（最后一维）计算时间导数  $\partial_t I$ 
        evolution = np.abs(np.gradient(state_4d, axis=-1))

        # 因果关联度  $C(I_t \rightarrow I_{\{t+\Delta t\}})$ ：逐体素时间平均
        causal_link = compute_causal_link(state_4d, temporal_axis=-1)

        is_fake_time = False
    else:
        # 单帧伪时序：用空间差分近似
        evolution = np.zeros_like(state_norm)
```

```

for i in range(state_norm.ndim):

    if state_norm.shape[i] >= 2:

        evolution += np.abs(np.gradient(state_norm, axis=i))

    causal_link = np.zeros_like(state_norm)

    is_fake_time = True

return D4Spacetime(

    evolution=evolution,

    causal_link=causal_link,

    is_fake_time=is_fake_time,

)

```

C.6.2 关键实现细节

细节	处理方式
2.5D+时间处理	先剥离高度通道，保证 causal_link 与 D2/D3 形状一致
真实时序演化	沿时间轴（最后一维）求导 $\partial_t I$ ，取绝对值；形状(..., T)
因果关联度	相邻帧逐体素归一化互信息；形状去除时间轴(...)
单帧伪时序	用空间差分近似；is_fake_time=True；形状与 state_norm 同
与 D3 关系	D4 直接从 state_norm 求时序导数， 不依赖 D3.density

笔记（形状不一致）：真实时序分支中 `evolution` 保留时间轴（形状 `(H, W, T)`），而 `causal\_link` 为逐体素时间平均值（去除时间轴，形状 `(H, W)`）。伪时序分支中二者与 `state\_norm` 同形状。下游若需统一处理，应检查 `is\_fake\_time` 标志，或在 `causal\_link` 上广播时间轴。

C.6.3 量纲验证（原始量纲）

量	几何制量纲	验证
evolution $\partial_t I$	$\text{bit}^{1/2}[\text{L}]^{-3}$	$\partial_t: [\text{L}]^{-1} \times I: \text{bit}^{1/2}[\text{L}]^{-2} \checkmark$
causal_link	1（归一化）	互信息归一化 $\checkmark$

C.7 辅助函数

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

C.7.1 `compute\_hessian`

```
```python
def compute_hessian(state_2d, spatial_axes):
 """计算 Hessian 矩阵 $H_{ij} = \partial^2 I / (\partial x_i \partial x_j)$

 输入应为已剥离高度通道的场；仅沿 spatial_axes 求导。
 H 的基础形状与 state_2d 一致，末尾追加 (ndim_spatial, ndim_spatial)。
 """
 ndim_spatial = len(spatial_axes)

 # H 的形状: state_2d.shape + (ndim_spatial, ndim_spatial)
 H = np.zeros(state_2d.shape + (ndim_spatial, ndim_spatial))

 for i_idx, i in enumerate(spatial_axes):
 gi = np.gradient(state_2d, axis=i)

 for j_idx, j in enumerate(spatial_axes):
 H[..., i_idx, j_idx] = np.gradient(gi, axis=j)

 return H
```

注记（形状约定）：`H` 的基础形状与 `state\_2d` 一致，末尾追加 `(ndim\_spatial, ndim\_spatial)`。对 2D 输入 `(H, W)`，`H.shape = (H, W, 2, 2)`；对 2.5D+时间输入 `(H, W, T)`，`H.shape = (H, W, T, 2, 2)`。赋值 `H[..., i\_idx, j\_idx] = np.gradient(gi, axis=j)` 形状匹配。

### C.7.2 `compute\_curl`

```
```python
def compute_curl(v_I):
    """3D 旋度  $\nabla \times \mathbf{v}$ """
    if v_I.ndim != 4 or v_I.shape[-1] != 3:
        raise ValueError("compute_curl 仅支持 (H, W, D, 3) 输入")

    vx, vy, vz = v_I[..., 0], v_I[..., 1], v_I[..., 2]

    curl_x = np.gradient(vz, axis=1) - np.gradient(vy, axis=2)
    curl_y = np.gradient(vx, axis=2) - np.gradient(vz, axis=0)
    curl_z = np.gradient(vy, axis=0) - np.gradient(vx, axis=1)
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

```
return np.stack([curl_x, curl_y, curl_z], axis=-1)
```

C.7.3 `compute\_euler\_char\_3d`

```
```python
def compute_euler_char_3d(density):
 """三维体素胞腔复形欧拉示性数 $\chi = V - E + F - C$ """
 binary = density > np.median(density)
 V = count_vertices(binary)
 E = count_edges(binary)
 F = count_faces(binary)
 C = count_cells(binary)
 return int(V - E + F - C)
```

### C.7.4 `compute\_euler\_char\_2d`

```
```python
def compute_euler_char_2d(density):
    """二维曲面欧拉示性数  $\chi = V - E + F$ """
    binary = density > np.median(density)
    V = count_vertices_2d(binary)
    E = count_edges_2d(binary)
    F = count_faces_2d(binary)
    return int(V - E + F)
```

C.7.5 `compute\_ricci\_scalar\_3d`

```
```python
def compute_ricci_scalar_3d(density, threshold=1e-6):
 """三维 Ricci 曲率标量（启发式弱场近似）

 假设 3D 体密度 ρ_V 与度规扰动 h 满足 $\rho_V \sim e^h$, 则

$$R^{(3)} \approx (1/2) \nabla^2 h \approx (1/2) \nabla^2 \ln \rho_V$$

 严格形式需结合 IFC 几何生成方程推导, 属后续工作。
```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

数值稳定性：对密度极小区域（ $|\rho_V| < \text{threshold}$ ）做掩膜处理，  
避免  $\log$  发散导致的数值伪影。

```
"""

eps = 1e-12

log_density = np.log(np.abs(density) + eps)

R = np.zeros_like(density)

for i in range(density.ndim):
 second_deriv = np.gradient(np.gradient(log_density, axis=i), axis=i)
 R += second_deriv

R = 0.5 * R

极小密度区域掩膜

mask = np.abs(density) < threshold

R[mask] = 0.0

return R
```

**注记（数值稳定性）：**当 `density` 极小（接近  $\epsilon=10^{-12}$ ）时，`log\_density` 趋于极负值，其二阶导数可能产生极大的数值波动。本实现通过对极小密度区域（ $|\rho_V| < \text{threshold}$ ）做掩膜处理来抑制数值伪影。阈值 `threshold` 应根据环境观测的量级自适应确定。

### C.7.6 `compute\_causal\_link`

```
```python  
def compute_causal_link(density, temporal_axis=-1):  
    """因果关联度：沿时间轴的逐体素归一化互信息  
  
    temporal_axis 显式指定时间轴（默认最后一维）。  
    返回形状：去除 temporal_axis 后的逐体素互信息数组。  
    T < 2 时返回同形状为零数组。  
    """  
  
    T = density.shape[temporal_axis]  
  
    spatial_shape = tuple(  
        s for i, s in enumerate(density.shape)
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```

        if i != temporal_axis % density.ndim
    )
    if T < 2:
        # 与  $T \geq 2$  的 link 形状一致（去除时间轴）
        return np.zeros(spatial_shape)

    link = np.zeros(spatial_shape)
    for t in range(T - 1):
        a = np.take(density, t, axis=temporal_axis)
        b = np.take(density, t + 1, axis=temporal_axis)
        link += normalized_mutual_info(a, b)

    return link / (T - 1)

```

注记（‘ $T < 2$ ’ 形状语义）：‘ $T < 2$ ’ 时返回去除时间轴后的空间形状零数组，与 ‘ $T \geq 2$ ’ 时 ‘link’ 的形状一致。下游若需广播回原始形状，可在时间轴上扩展。

C.7.7 拓扑计数与互信息函数

```

```python
def count_cells(binary):
 """三维胞腔数：值为 1 的体素数"""
 return int(np.sum(binary))

def count_faces_2d(binary):
 """二维面数：值为 1 的像素数"""
 return int(np.sum(binary))

以下函数见 dpl_core.topology 模块：
count_vertices(binary) 三维顶点数
count_edges(binary) 三维边数
count_faces(binary) 三维面数
count_vertices_2d(binary) 二维顶点数

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
 开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

```
count_edges_2d(binary) 二维边数

以下函数见 dpl_core.mutual_info 模块：

normalized_mutual_info(a, b) 归一化互信息，返回与输入同形状的逐体素数组
```

注记（`normalized_mutual_info` 返回类型）：`normalized_mutual_info(a, b)` 返回与输入同形状的逐体素互信息数组，支持逐体素因果关联计算。若需全局标量，可对返回结果取均值或最大值。其完整实现涉及体素邻接关系构造与信息论统计，属于独立模块。

## C.8 完整伪代码

```
```python
def project(env_state, has_temporal, constants):

    """DPL 协议层接口：D1–D4 流形分解（附录 B B.4）"""

    # 1. 维度检测

    dim_info = detect_dimension(env_state, has_temporal)

    # 2. 预处理

    state_norm = preprocess(env_state, constants)

    # 3. D1 流线条

    D1 = extract_D1(state_norm, dim_info, constants)

    # 4. D2 面维

    D2 = extract_D2(state_norm, D1, dim_info, constants)

    # 5. D3 体维

    D3 = extract_D3(state_norm, D2, dim_info)

    # 6. D4 时空维（直接从 state_norm 求时间导数）

    D4 = extract_D4(state_norm, dim_info, has_temporal)
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```

return ProjectedState(
    D1=D1, D2=D2, D3=D3, D4=D4,
    dim_info=dim_info,
)

```

C.9 异常条件与边界处理

条件	处理方式
<code>env_state.ndim < 2</code>	由 <code>detect_dimension</code> 对 <code>env_state.ndim < 2</code> 预先抛出 <code>ValueError</code>
单帧时序 (<code>has_temporal=True</code> 但只有 1 帧)	<code>is_fake_time=True</code> , 返回空间近似
零梯度点 ($ \nabla I =0$)	方向单位向量定义为零向量
零信息密度 ($ \Phi =0$)	速度场正则化: $\mathbf{v}_I = J_I / (\Phi + \epsilon_0)$
均匀场 ($ \nabla \Phi =0$)	涡旋校验跳过 (见附录 B B.5.4 注记)
非单连通环境	涡旋校验激活, 环量 $\oint \mathbf{v}_I \cdot d\mathbf{l} = n\Phi_{topo}$
4D 时空输入	D3 取首个空间切片 <code>state_norm[..., 0]</code> ; D4 直接从 <code>state_norm</code> 沿时间轴求导
2.5D 输入	D1、D2、D4 剥离高度通道后按 2D 处理; D3 体密度重新引入 (ρ^2), 2D 平面定义
2.5D+时间输入(H, W, 1, T)	D1、D2、D4 剥离高度通道后按 2D 处理 (保留时间轴); D3 每帧 2.5D 密度; D4 沿时间轴求导
形式升维 (2D→3D、1D→3D)	3D 拓扑为平凡乘积结构, $\chi=0$ 、 $R^{(3)}=0$

C.10 复杂度分析

步骤	时间复杂度	空间复杂度
<code>detect_dimension</code>	$O(1)$	$O(1)$
<code>preprocess</code>	$O(N)$	$O(N)$
<code>extract_D1</code>	$O(N)$	$O(N)$
<code>extract_D2</code>	$O(N \cdot d^2)$	$O(N \cdot d^2)$

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com
 开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)
 开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

步骤	时间复杂度	空间复杂度
extract_D3	$O(N)$	$O(N)$
extract_D4	$O(N \cdot T)$	$O(N)$
总计	$O(N \cdot (d^2 + T))$	$O(N \cdot d^2)$

其中 $N = \prod_{i=1}^{d_{\text{spatial}}} N_i$ 为空间体素总数 (N_i 为第 i 个空间轴的体素数), d 为空间维度 (2 或 3), T 为时间步数。Hessian 计算贡献 $O(N \cdot d^2)$, 是主要开销。`project` 接口的复杂度与输入规模线性相关 (d, T 为常数), 符合 DPL 面积律的算力节俭目标。

附录 D：自适应几何门禁实现规范

说明：本附录定义 `check` 接口 (附录 B, B.5) 的完整实现规范, 涵盖七项校验的数学定义、触发条件、阈值校准、违反处理与返回结构。所有符号、量纲回溯附录 A; 接口定义回溯附录 B; D1–D4 特征回溯附录 C。本附录不引入新物理假设, 仅为已有门禁架构的工程实现规格。

符号约定：标量取绝对值 $|\cdot|$; 向量/复数取模 $|\cdot|$ 。 K 、 J_{in} 、 J_{out} 、 χ 、 $time_arrow$ 为标量。

接口签名统一约定：所有 `check\_\*` 校验函数统一签名为 `(projected, constants, modification)`，未用参数保留原名 (不写 `\_`), 保证可扩展性。

前置修订：本附录要求附录 B、附录 C 同步修订：

修订项	目标位置	说明
D1FlowLine 增加 gradient_vector 字段	附录 C C.3.1	原始梯度向量场 $J_f = -D \nabla I$, 用于信息守恒与跨维隔离校验
D2Surface 增加 vorticity_is_vector: bool 字段	附录 C C.4.1	显式标记涡旋为向量 (3D+) 或标量 (2D、2.5D+时间)
UniversalConstants 增加 clear_region_percentile 字段	附录 B B.3.3	清晰区边界阈值分位数 (默认 80)
UniversalConstants 增加 l_0、l_0 字段	附录 B B.3.3	参考信息密度与参考长度 (默认 1)

D.1 门禁总览

作者：纪辉泉 (普宁 515300) Email: 2635714379@qq.com
开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)
开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

D.1.1 职责

`check` 接口负责在修改提案（`Modification`）应用前，对 D1–D4 流形特征进行七项校验，输出通过/拦截/升维补全/降维抽象四种结果。门禁是 DPL 的 R2 感知层与 SICAR5 元层之间的安全闸。

D.1.2 七项校验

校验项	数学判据	适用维度与触发条件	违反标记
信息守恒	$\ J_{in} - J_{out}\ < \varepsilon$	所有维	INFO_NOT_CONSERVED
曲率有界	$\ K\ < K_{\max}$	D2+	EXTREME_CURVATURE
涡旋校验	$\ \nabla \times \mathbf{v}_I\ < \omega_{\max}$	D2+, 且 D2 层信息密度非均匀或环境非单连通	EXTREME_VORTICITY
拓扑合法	$\chi \in \mathbb{Z}$	D3+	INVALID_TOPOLOGY
自指稳定	$\lim_{n \rightarrow \infty} f^n(x) = x^*$	所有维	SELF_REF_UNSTABLE
跨维隔离	$\text{Overlap}(\Phi_{D1}, \Phi_{D2}) > 1 - \delta_{ov}$	所有维	CROSS_DIM_LEAKAGE
因果一致	不存在闭合类时曲线; $time_arrow > 0$	D4+	CAUSALITY_VIOLATION

D.1.3 自适应策略

```
```python
前置断言
assert effective_dim >= 2, "effective_dim 不应小于 2"

基础校验项（所有维度均激活）
base_gates = [check_information_conservation,
 check_self_reference_stability,
 check_cross_dim_isolation]

按有效维度叠加
if effective_dim == 2:
 gates = base_gates
```

```

elif effective_dim == 2.5:

 gates = base_gates + [check_curvature_bound]

elif effective_dim == 3:

 gates = base_gates + [check_curvature_bound, check_topology_valid]

elif effective_dim >= 4:

 gates = base_gates + [check_curvature_bound,

 check_topology_valid,

 check_causality]

涡旋校验：条件激活（非均匀或非单连通）

if should_activate_vorticity_check(projected):

 gates.append(check_vorticity_bound)

```

## D.2 校验 1：信息守恒

### D.2.1 数学定义

信息守恒校验检测清晰区边界上的法向通量密度是否守恒：

$$||J_{in}| - |J_{out}|| < \varepsilon$$

其中：

- $J_{in}$ ：流入清晰区的法向通量密度， $J_I \cdot \hat{n}$  的正部；
- $J_{out}$ ：流出清晰区的法向通量密度， $J_I \cdot \hat{n}$  的负部（取绝对值）；
- $J_I = -D \nabla I$ ：熵流密度，向量场，量纲  $bit^{1/2}[L]^{-2}$ （几何制）；
- $\varepsilon$ ：通量密度阈值，由 `constants.epsilon\_gate` 校准。

### D.2.2 计算算法

```

```python

def check_information_conservation(projected, constants, modification):

    """信息守恒校验"""

    # 步骤 1：从 D1 提取熵流向量场

    J_I = projected.D1.gradient_vector # 形状 (... , spatial_dim)

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

```

# 步骤 2: 在清晰区边界上计算法向通量密度

boundary = get_clear_region_boundary(projected, constants)

normal = get_boundary_normal(boundary)

J_n = np.sum(J_I[boundary] * normal, axis=-1) # 法向分量标量


# 步骤 3: 分离流入与流出

J_in = np.abs(J_n[J_n > 0])

J_out = np.abs(J_n[J_n < 0])


# 步骤 4: 计算通量密度差

if len(J_in) == 0 or len(J_out) == 0:

    return GateResult(passed=True, score=1.0, violations=[],

                        action="pass")


delta_J = np.abs(np.mean(J_in) - np.mean(J_out))

eps = constants.epsilon_gate


if delta_J < eps:

    return GateResult(passed=True, score=1.0, violations=[],

                        action="pass")

else:

    violation = min(1.0, delta_J / eps)

    penalty = max(0.0, 1.0 - violation)

    return GateResult(

        passed=False,

        score=penalty,

        violations=["INFO_NOT_CONSERVED"],

        action="block",

    )

```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

D.2.3 辅助函数

```
```python
def get_clear_region_boundary(projected, constants):
 """从 D1 层梯度模提取清晰区边界掩膜

 清晰区定义为梯度模超过阈值的区域（信息交互活跃区）；
 边界为清晰区与其补集交界。

 对单帧输入 (H, W)：全局分位数阈值。
 对多帧输入 (H, W, T)：逐帧分位数阈值。

 """
 grad_mag = projected.D1.gradient
 percentile = getattr(constants, "clear_region_percentile", 80)

 if grad_mag.ndim == 3:
 # (H, W, T)：逐帧分位数
 threshold = np.percentile(grad_mag, percentile, axis=(0, 1))
 clear_mask = grad_mag > threshold[None, None, :]
 else:
 # (H, W)：全局分位数
 threshold = np.percentile(grad_mag, percentile)
 clear_mask = grad_mag > threshold

 boundary_mask = morphological_boundary(clear_mask)
 return boundary_mask

def get_boundary_normal(boundary):
 """基于边界掩膜的梯度方向算法向单位向量"""
 normals = np.gradient(boundary.astype(float))
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

```

if not isinstance(normals, (list, tuple)):

 normals = [normals]

norm = np.sqrt(sum(n ** 2 for n in normals) + 1e-12)

normals = [n / norm for n in normals]

return np.stack(normals, axis=-1)

以下函数见 dpl_core.gate 模块:

morphological_boundary(mask) 形态学边界提取

compute_boundary_flux_diff(J_I) 边界通量密度差

```

#### D.2.4 阈值校准

```

```python

def calibrate_epsilon_gate(observations, D, I_0=1.0, l_0=1.0):

    """从观测序列校准信息守恒阈值"""

    flux_diffs = []

    for t in range(len(observations) - 1):

        I_norm = observations[t] / I_0

        grads = np.gradient(I_norm)

        if not isinstance(grads, (list, tuple)):

            grads = [grads]

        J_I = -D * np.stack(grads, axis=-1)

        delta = compute_boundary_flux_diff(J_I)

        flux_diffs.append(delta)

    if not flux_diffs:

        return 1e-3

    return np.percentile(flux_diffs, 95)

```

默认值：若 `observations` 长度不足，`epsilon\_gate = 1e-3`。

D.3 校验 2：曲率有界

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

D.3.1 数学定义

曲率有界校验检测 D2 层的高斯曲率是否超出阈值：

$$|K| < K_{\max}$$

其中：

- K ：高斯曲率，标量，量纲 1（无量纲，归一化后）；
- $K_{\max}$ ：曲率有界阈值，由 `constants.K\_max` 校准。

D.3.2 计算算法

```
```python
def check_curvature_bound(projected, constants, modification):
 """曲率有界校验"""
 K = projected.D2.curvature
 K_max = constants.K_max

 violation_mask = np.abs(K) > K_max
 if not np.any(violation_mask):
 return GateResult(passed=True, score=1.0, violations=[],
 action="pass")

 violation_ratio = np.max(np.abs(K) / K_max)
 violation = min(1.0, violation_ratio)
 penalty = max(0.0, 1.0 - violation)

 return GateResult(
 passed=False,
 score=penalty,
 violations=["EXTREME_CURVATURE"],
 action="dimension_lift",
)
```
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

D.3.3 阈值校准

```
```python
def calibrate_K_max(observations, I_0=1.0, l_0=1.0):
 """从观测序列校准曲率有界阈值"""
 curvatures = []
 for obs in observations:
 I_norm = obs / I_0
 if I_norm.ndim == 2:
 Ix = np.gradient(I_norm, axis=0)
 Iy = np.gradient(I_norm, axis=1)
 Ixx = np.gradient(Ix, axis=0)
 Iyy = np.gradient(Iy, axis=1)
 Ixy = np.gradient(Ix, axis=1)
 det_H = Ixx * Iyy - Ixy**2
 denom = (1 + Ix**2 + Iy**2)**2
 K = det_H / (denom + 1e-12)
 curvatures.append(np.abs(K).flatten())
 if not curvatures:
 return 1.0
 all_K = np.concatenate(curvatures)
 return np.percentile(all_K, 95)
```
```

默认值：若 `observations` 不足，`K\_max = 1.0`。

D.4 校验 3：涡旋校验

D.4.1 数学定义

涡旋校验检测 D2 层的涡旋强度是否超出阈值：

$$\|\nabla \times \mathbf{v}_I\| < \omega_{\max}$$

其中：

➤ $\mathbf{v}_I = J_I / (\|\Phi\| + \epsilon_0)$ ：信息流速度场，向量场；

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

- $\nabla \times \mathbf{v}_I$: 涡旋项，向量场（3D+）或标量（2D、2.5D+时间）；
- $\omega_{\max}$ ：涡旋有界阈值，由 `constants.omega\_max` 校准。

D.4.2 触发条件

涡旋校验仅在以下条件之一满足时激活：

1. D2 层信息密度非均匀： $\|\nabla\|\Phi\|>0$ ；
2. 环境非单连通：存在非平凡拓扑缺陷。

```
```python
def should_activate_vorticity_check(projected):
 """判断是否激活涡旋校验"""

 grad_phi = np.gradient(np.abs(projected.D3.density))

 is_nonuniform = np.any(np.abs(grad_phi) > 1e-6)

 is_nonSimply_connected = (projected.D3.euler_char != 2)

 return is_nonuniform or is_nonSimply_connected
```

#### D.4.3 计算算法

```
```python
def check_vorticity_bound(projected, constants, modification):
    """涡旋校验"""

    omega = projected.D2.vorticity

    omega_max = constants.omega_max

    # 依据 D2Surface.vorticity_is_vector 显式标志判断
    if projected.D2.vorticity_is_vector:
        # 3D+ 向量涡旋，取模

        omega_magnitude = np.linalg.norm(omega, axis=-1)
    else:
        # 2D 或 2.5D+时间的标量涡旋，取绝对值

        omega_magnitude = np.abs(omega)
```

```

violation_mask = omega_magnitude > omega_max

if not np.any(violation_mask):

    return GateResult(passed=True, score=1.0, violations=[],

                      action="pass")

violation_ratio = np.max(omega_magnitude / omega_max)

violation = min(1.0, violation_ratio)

penalty = max(0.0, 1.0 - violation)

return GateResult(

    passed=False,

    score=penalty,

    violations=["EXTREME_VORTICITY"],

    action="block",

)

```

D.4.4 阈值校准

```

```python

def calibrate_omega_max(observations, D, I_0=1.0):

 """从观测序列校准涡旋有界阈值"""

 vorticity_magnitudes = []

 for obs in observations:

 I_norm = obs / I_0

 grads = np.gradient(I_norm)

 if not isinstance(grads, (list, tuple)):

 grads = [grads]

 J_I = -D * np.stack(grads, axis=-1)

 v_I = J_I / (np.abs(I_norm)[..., None] + 1e-12)

 if I_norm.ndim == 2:

 vx, vy = v_I[..., 0], v_I[..., 1]

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

```

 omega = np.gradient(vy, axis=0) - np.gradient(vx, axis=1)

 vorticity_magnitudes.append(np.abs(omega).flatten())

 if not vorticity_magnitudes:

 return 1.0

 all_omega = np.concatenate(vorticity_magnitudes)

 return np.percentile(all_omega, 95)

```

**默认值：**若 `observations` 不足，`omega\_max = 1.0`。

## D.5 校验 4：拓扑合法

### D.5.1 数学定义

拓扑合法校验检测 D3 层的欧拉示性数是否为整数：

$$\chi \in \mathbb{Z}$$

其中：

- $\chi = V_{\text{vert}} - E + F - C_{\text{cell}}$ ：三维体素胞腔复形的欧拉示性数；
- 对 2.5D：退化为  $\chi = V_{\text{vert}} - E + F$ 。

### D.5.2 计算算法

```

```python
def check_topology_valid(projected, constants, modification):

    """拓扑合法校验"""

    chi = projected.D3.euler_char

    if isinstance(chi, int) or (isinstance(chi, float) and chi.is_integer()):

        return GateResult(passed=True, score=1.0, violations=[],

                           action="pass")

    else:

        return GateResult(

            passed=False,

            score=0.3,

            violations=["INVALID_TOPOLOGY"],

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```
        action="block",  
    )
```

D.5.3 实现笔记

- `euler\_char` 由 `compute\_euler\_char\_3d` 或 `compute\_euler\_char\_2d` 计算（附录 C C.7.3、C.7.4），结果默认为整数；
- 对形式升维（2D→3D、1D→3D），`euler\_char = 0`（平凡乘积结构），合法。

D.6 校验 5：自指稳定

D.6.1 数学定义

自指稳定校验检测泡壁诱导的自指映射是否收敛到唯一不动点：

$$\lim_{n \rightarrow \infty} f^n(x) = x^*$$

其中：

- $f: \mathcal{M} \rightarrow \mathcal{M}$ ：泡壁诱导的自指映射；
- $x^* \in \text{Fix}(f)$ ：唯一不动点，对应 IFC 公理 III 中的意识锚点。

D.6.2 计算算法

```
```python  
def check_self_reference_stability(projected, constants, modification):
 """自指稳定校验

 通过修改提案的 Lipschitz 常数估计判断自指映射稳定性。
 constants 当前未使用，但保留以备后续扩展。
 """

 L = estimate_lipschitz(modification)

 if L < 1.0:
 return GateResult(passed=True, score=1.0, violations=[],
 action="pass")
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

```

violation = min(1.0, L - 1.0)

penalty = max(0.0, 1.0 - violation)

return GateResult(
 passed=False,
 score=penalty,
 violations=["SELF_REF_UNSTABLE"],
 action="block",
)

def estimate_lipschitz(modification):
 """估计修改的 Lipschitz 常数

 modification.params 为 dict，仅提取数值型字段参与估计。
 """
 param_values = [v for v in modification.params.values()
 if isinstance(v, (int, float))]

 if not param_values:
 return 0.0

 return max(np.abs(v) for v in param_values)

```

### D.6.3 实现注记

- 自指映射  $f$  的具体形式由泡壁几何与意识场  $\Psi_{con}$  决定，属后续工作；
- 本实现用修改提案的 Lipschitz 常数估计作为近似判据；
- 对 D1+ 生效（所有维度）。

## D.7 校验 6：跨维隔离

### D.7.1 数学定义

跨维隔离校验检测 D1 特征集合与 D2 特征集合的投影交叠量：

$$\text{Overlap}(\Phi_{D1}, \Phi_{D2}) > 1 - \delta_{\text{overlap}}$$

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

其中：

- $\text{Overlap} \in [0,1]$ ：跨维投影交叠量；
- $\delta_{\text{overlap}}$ ：容许偏离量，由 `constants.delta\_overlap` 校准。

### D.7.2 Overlap 计算

依据主文 4.1 节与附录 B B.5.4：

1. 将 D1、D2 特征张量按  $I_0$ 、 $I_0$  归一化为无量纲函数  $\hat{u}$ 、 $\hat{v}$ ；
2. 在空间切片  $\Sigma_t$  上计算  $L^2$  内积  $\langle \hat{u}, \hat{v} \rangle = \int_{\Sigma_t} \hat{u} \cdot \hat{v} dx$ ；
3. 语义对应关系集合  $C$  为固定三对。

配对策略按涡旋类型分支：

涡旋类型	配对 1	配对 2	配对 3
向量涡旋（3D+）	$(\nabla I, \nabla \times \mathbf{v}_I)$	$(\nabla \cdot J_I, K(I))$	$(\text{direction}(J_I), \nabla \times \mathbf{v}_I)$
标量涡旋（2D、2.5D+时间）	$(\ \nabla I\ , \ \nabla \times \mathbf{v}_I\ )$	$(\nabla \cdot J_I, K(I))$	$(\ \nabla I\ , K(I))$

配对策略的依据：向量涡旋与 D1 的向量特征（梯度、方向）可逐分量比较；标量涡旋仅与 D1 的标量特征（梯度模、散度）可比较。

### D.7.3 计算算法

```
```python
def check_cross_dim_isolation(projected, constants, modification):
    """跨维隔离校验

    I_0、l_0 从 constants 读取；配对策略依据 vorticity_is_vector 分支。
    """
    I_0 = constants.I_0
    l_0 = constants.l_0

    # 步骤 1：准备三对语义对应（按涡旋类型分支）

    if projected.D2.vorticity_is_vector:
        # 向量涡旋（3D+）：张量阶配对

        pairs = [
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```

        (projected.D1.gradient_vector, projected.D2.vorticity, 1), # 向量 vs 向量
        (projected.D1.divergence, projected.D2.curvature, 0),      # 标量 vs 标量
        (projected.D1.direction, projected.D2.vorticity, 0),      # 单位向量 vs 向量
    ]
else:
    # 标量涡旋（2D、2.5D+时间）：标量配对
    pairs = [
        (projected.D1.gradient, projected.D2.vorticity, 0),      # 标量 vs 标量
        (projected.D1.divergence, projected.D2.curvature, 0),    # 标量 vs 标量
        (projected.D1.gradient, projected.D2.curvature, 0),      # 标量 vs 标量
    ]

# 步骤 2：逐对计算余弦相似度
cosines = []

for u, v, order in pairs:
    u_hat = normalize_tensor(u, I_0, l_0, order)
    v_hat = normalize_tensor(v, I_0, l_0, order)
    cos = cosine_similarity_L2(u_hat, v_hat)
    cosines.append(cos)

# 步骤 3：计算 Overlap
overlap = np.mean(cosines)

delta_overlap = constants.delta_overlap

# 步骤 4：判断是否通过
if overlap > 1 - delta_overlap:
    return GateResult(passed=True, score=1.0, violations=[],
                       action="pass")
else:

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```

violation = min(1.0, (1 - delta_overlap) - overlap)

penalty = max(0.0, 1.0 - violation / max(delta_overlap, 1e-6))

return GateResult(

    passed=False,

    score=penalty,

    violations=["CROSS_DIM_LEAKAGE"],

    action="dimension_reduce",

)

```

```
def normalize_tensor(t, I_0, l_0, tensor_order):
```

"""按 I_0 与 l_0 归一化张量，展平为无量纲向量

tensor_order: 张量阶 (0=标量, 1=向量, 2=二阶张量)

归一化因子: $1 / (I_0 / l_0^{\text{tensor_order}})$

注意:

- 标量场 (含散度、曲率、梯度模、标量涡旋): tensor_order=0
- 向量场 (含梯度向量、向量涡旋): tensor_order=1
- 二阶张量 (Hessian): tensor_order=2
- direction 为单位向量 (已无量纲): tensor_order=0

"""

```
t = np.asarray(t)
```

```
factor = I_0 / (l_0 ** tensor_order) if l_0 > 0 else I_0
```

```
if factor == 0:
```

```
    return np.zeros(0)
```

```
return (t / factor).flatten()
```

```
def cosine_similarity_L2(u, v):
```

"""计算展平向量余弦相似度，对应 L^2 内积的离散近似

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

均匀网格上, L^2 内积 $\langle \hat{u}, \hat{v} \rangle_{L^2} \approx \sum \hat{u}_i \cdot \hat{v}_i \cdot \Delta x^n$;

分子分母中 Δx^n 相消, 余弦相似度等价于展平向量的欧几里得余弦。

```
"""
```

```
nu = np.linalg.norm(u)
```

```
nv = np.linalg.norm(v)
```

```
if nu == 0 or nv == 0:
```

```
    return 0.0
```

```
return float(np.abs(np.dot(u, v)) / (nu * nv))
```

注记（`direction` 取 `tensor\_order=0` 的语义）：`direction` 为单位向量场，已在 D1 中归一化为无量纲量（ $\text{direction} = J_I / \|J_I\|$ ），故 `tensor\_order=0`（不再除 l_0 ）。若需严格按向量归一化，应使用 `gradient\_vector` 而非 `direction`。

D.7.4 阈值校准

```
```python
def calibrate_delta_overlap(observations, I_0=1.0, l_0=1.0):
 """从观测序列校准跨维隔离容许偏离量"""
 overlaps = []
 for obs in observations:
 overlap = compute_overlap_from_observation(obs, I_0, l_0)
 overlaps.append(overlap)
 if len(overlaps) < 2:
 return 0.05
 return max(0.0, 1 - np.percentile(overlaps, 5))

以下函数见 dpl_core.gate 模块：
compute_overlap_from_observation(obs, I_0, l_0) 从单次观测计算 Overlap
```

**默认值：**若 `observations` 不足，`delta\_overlap = 0.05`。

## D.8 校验 7：因果一致

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

### D.8.1 数学定义

因果一致校验检测 D4 层是否存在闭合类时曲线，以及时间箭头方向：

不存在闭合类时曲线 $\gamma$ ;  $time\_arrow > 0$

其中：

- 闭合类时曲线  $\gamma$ : 闭合曲线使  $g_{\mu\nu}\dot{x}^\mu\dot{x}^\nu < 0$  处处成立 ( $\dot{x}^\mu = dx^\mu/d\lambda$  为切向量)；
- $time\_arrow$ : 时间箭头方向，由 `constants.time\_arrow` 校准。

### D.8.2 计算算法

```
``python
def check_causality(projected, constants, modification):
 """因果一致校验"""
 # 步骤 1: 检测修改是否引入闭合类时曲线
 if modification.type == "modify_topology":
 creates_closed_timelike_curve = detect_closed_timelike_curve(
 projected, modification
)
 if creates_closed_timelike_curve:
 return GateResult(
 passed=False,
 score=0.05,
 violations=["CAUSALITY_VIOLATION"],
 action="block",
)

 # 步骤 2: 检测时间箭头方向
 time_arrow = constants.time_arrow
 if time_arrow < 0:
 return GateResult(
 passed=False,
 score=0.3,
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

```

 violations=["CAUSALITY_VIOLATION"],

 action="block",

)

 return GateResult(passed=True, score=1.0, violations=[],

 action="pass")

def detect_closed_timelike_curve(projected, modification):

 """检测修改是否引入闭合类时曲线"""

 if projected.D4.is_fake_time:

 return False

 evolution = projected.D4.evolution

 if np.any(np.gradient(evolution) * evolution < -1e-6):

 return True

 return False

```

### D.8.3 阈值校准

```

```python

def calibrate_time_arrow(observations):

    """从观测序列校准时间箭头方向"""

    if len(observations) < 2:

        return 0.0

    kl_values = []

    for t in range(len(observations) - 1):

        P_t = normalize_histogram(observations[t])

        P_next = normalize_histogram(observations[t + 1])

        kl = compute_KL_divergence(P_t, P_next)

        kl_values.append(kl)

    return float(np.mean(kl_values))

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```
# 以下函数见 dpl_core.gate 模块:

#   normalize_histogram(data)           直方图归一化
#   compute_KL_divergence(P, Q)        KL 散度
```

默认值: 若 `observations` 不足, `time\_arrow = 0.0` (中性)。

D.9 门禁综合执行流程

D.9.1 完整算法

```
```python
def check(modification, projected, constants):
 """自适应几何门禁 (附录 B B.5) """

 dim_info = projected.dim_info
 effective_dim = dim_info.effective_dim

 assert effective_dim >= 2, "effective_dim 不应小于 2"

 # 步骤 1: 确定激活的校验项

 base_gates = [check_information_conservation,
 check_self_reference_stability,
 check_cross_dim_isolation]

 if effective_dim == 2:
 gates = base_gates
 elif effective_dim == 2.5:
 gates = base_gates + [check_curvature_bound]
 elif effective_dim == 3:
 gates = base_gates + [check_curvature_bound, check_topology_valid]
 elif effective_dim >= 4:
 gates = base_gates + [check_curvature_bound,
 check_topology_valid,
```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

```

 check_causality]

 if should_activate_vorticity_check(projected):
 gates.append(check_vorticity_bound)

 # 步骤 2: 逐项校验, 累乘得分
 violations = []
 score = 1.0
 for gate in gates:
 result = gate(projected, constants, modification)
 if not result.passed:
 violations.extend(result.violations)
 score *= result.score

 # 步骤 3: 判断结果
 passed = len(violations) == 0
 action = determine_action(passed, violations)

 return GateResult(
 passed=passed,
 violations=violations,
 score=score,
 action=action,
)

def determine_action(passed, violations):
 """确定门禁动作（与附录 B B.5.3 的 action 字段一致）"""
 if passed:
 return "pass"

```

```
elif "INFO_NOT_CONSERVED" in violations:
 return "block"

elif "CAUSALITY_VIOLATION" in violations:
 return "block"

elif "CROSS_DIM_LEAKAGE" in violations:
 return "dimension_reduce"

elif "EXTREME_CURVATURE" in violations:
 return "dimension_lift"

else:
 return "block"
```

D.9.2 四种结果

结果字符串	中文语义	触发条件	下游处理
pass	通过	所有校验通过	修改提案直接应用
block	拦截	信息守恒/因果一致/自指稳定/拓扑/涡旋违反	修改提案被拦截，作为 GateCounterexample 进入 SICA
dimension_lift	升维补全	曲率超界	触发升维补全机制
dimension_reduce	降维抽象	跨维隔离违反	触发降维抽象机制

D.10 返回结构与 SICA 对接

D.10.1 `GateResult`

字段	类型	说明
passed	bool	是否通过门禁
violations	List[str]	违反的校验项列表
score	float	门禁得分（各校验项累乘），[0,1]
action	str	通过/拦截/升维补全/降维抽象

D.10.2 `GateCounterexample` 映射

```
```python
def to_gate_counterexample(result, modification, projected):
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```
"""将门禁失败结果映射为 GateCounterexample（附录 B B.7.1）"""
```

```
sample_str = json.dumps({
    "type": modification.type,
    "target": modification.target,
    "params": modification.params,
    "info_before": modification.info_before,
    "info_after": modification.info_after,
}, sort_keys=True, default=str)

sample_hash = sha256(sample_str.encode()).hexdigest()

return GateCounterexample(
    gate_type=result.violations[0],
    violation=1.0 - result.score,
    raw_dim=projected.dim_info.spatial,
    effective_dim=projected.dim_info.effective_dim,
    sample_hash=sample_hash,
    timestamp=time.time_ns(),
)
```

D.10.3 SICA R2 门禁对接

‘GateCounterexample’ 进入 SICA R5 元层后，由 SICA R2 性能门禁进一步校验：

- 若门禁反例在 holdout 集上带来性能增益，则纳入 ‘H_t’；
- 否则丢弃，记为负样本。

D.11 异常条件

条件	异常类型	说明
modification.type 未注册	ValueError	未知修改类型
projected.dim_info 缺失	ValueError	未执行维度检测
constants 为 None	ValueError	未执行校准

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

条件	异常类型	说明
<code>effective_dim < 2</code>	ValueError	不支持的维度范围
<code>projected.D2</code> 缺失且需曲率校验	ValueError	无法执行曲率有界
<code>projected.D4</code> 缺失且需因果校验	ValueError	无法执行因果一致

D.12 实现注记

D.12.1 门禁阈值一致性

所有阈值由 `calibrate` 接口（附录 B B.3）输出，在 `check` 中共享：

- `epsilon\_gate`：信息守恒阈值；
- `K\_max`：曲率有界阈值；
- `omega\_max`：涡旋有界阈值；
- `delta\_overlap`：跨维隔离容许偏离量；
- `time\_arrow`：时间箭头方向；
- `clear\_region\_percentile`：清晰区边界阈值分位数（默认 80）；
- `I\_0`、`l\_0`：参考信息密度与参考长度（默认 1）。

不重复计算。

D.12.2 门禁单调性

`check` 通过时 `score=1.0`；每违反一项，`score` 乘相应惩罚因子。惩罚因子归一化到 $[0,1]$ ：

违反项	惩罚因子	说明
INFO_NOT_CONSERVED	$\max(0, 1-\text{violation})$	违反幅度归一化
EXTREME_CURVATURE	$\max(0, 1-\text{violation})$	违反幅度归一化
EXTREME_VORTICITY	$\max(0, 1-\text{violation})$	违反幅度归一化
INVALID_TOPOLOGY	0.3	拓扑破坏为不可容忍违反，硬编码
SELF_REF_UNSTABLE	$\max(0, 1-\text{violation})$	违反幅度归一化
CROSS_DIM_LEAKAGE	$\max(0, 1-\text{violation}/\delta_{\text{overlap}})$	归一化到容许偏离量
CAUSALITY_VIOLATION	0.05（闭环）/0.3（时序反转）	因果违反为不可容忍违反，硬

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

违反项	惩罚因子	说明
		编码

其中 `violation` 为归一化到 $[0,1]$ 的违反幅度。硬编码惩罚因子反映该违反项的物理严重性——拓扑破坏与因果违反为不可容忍违反，应显著降低门禁得分。

D.12.3 门禁得分范围

`score` $\in[0,1]$ 。`score=1.0` 表示所有校验通过；`score=0` 表示严重违反。

D.12.4 前置修订清单

修订项	目标位置	说明
`D1FlowLine` 增加 `gradient_vector` 字段	附录 C C.3.1	原始梯度向量场，用于信息守恒与跨维隔离校验
`D2Surface` 增加 `vorticity_is_vector: bool` 字段	附录 C C.4.1	显式标记涡旋为向量（3D+）或标量（2D、2.5D+时间）
`UniversalConstants` 增加 `clear_region_percentile` 字段	附录 B B.3.3	清晰区边界阈值分位数（默认 80）
`UniversalConstants` 增加 `I_0`、`l_0` 字段	附录 B B.3.3	参考信息密度与参考长度（默认 1）
统一 `check_*` 签名为 `(projected, constants, modification)`	本附录 D.2–D.8	保证 `check` 主循环统一调用

附录 E：宇宙常数校准与物理直觉引擎规范

说明：本附录定义 `calibrate` 接口（附录 B，B.3）与 `infer` 接口（附录 B，B.6）的完整实现规范，涵盖宇宙常数校准算法、物理直觉推理链、各规则置信度定义与主直觉类型判断。所有符号、量纲回溯附录 A；接口定义回溯附录 B；D1–D4 特征回溯附录 C。本附录不引入新物理假设，仅为已有校准与推理架构的工程实现规格。

符号约定：标量取绝对值 $|\cdot|$ ；向量/复数取模 $\|\cdot\|$ 。 ΔI 、 K 、 J_{total} 、 χ 、 $time_arrow$ 为标量； $\hat{f}(k)$ 为复数。

接口签名统一约定：`calibrate` 与 `infer` 的签名严格按附录 B B.3.1、B.6.1 定义。

前置修订：本附录要求附录 B `UniversalConstants` 补充以下字段：

修订项	默认值	说明	使用方
clear_region_percentile	80.0	清晰区边界阈值分位数	check

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

修订项	默认值	说明	使用方
I_0	1.0	参考信息密度	check、infer
l_0	1.0	参考长度	check、infer
chi_ref	2.0	参考非平凡拓扑欧拉示性数偏差	infer
gravity_threshold_ratio	0.1	引力直觉触发阈值比	infer
periodicity_threshold	0.3	周期直觉触发阈值	infer
conservation_stats	{}	守恒律统计量 {name: {mean, std, count}}	infer

E.1 总览

E.1.1 两个接口的职责

接口	职责	输入	输出
calibrate	从观测序列自动校准宇宙常数与门禁阈值	List[np.ndarray]	UniversalConstants
infer	从流形特征推断环境底层物理规律	ProjectedState+UniversalConstants	IntuitionResult

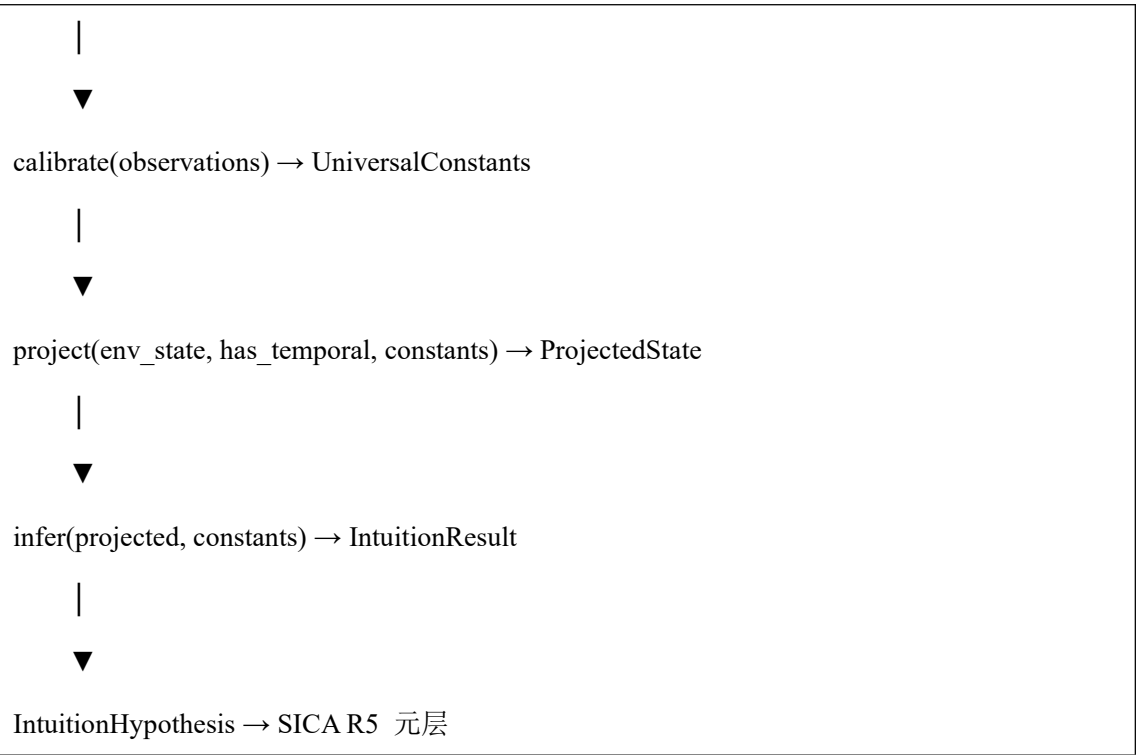
E.1.2 校准与推理的耦合

- ‘calibrate’ 输出的 ‘UniversalConstants’ 作为常量传入 ‘infer’；
- ‘infer’ 的四条推理链依赖 ‘constants’ 中的 ‘K_max’、‘gravity_threshold_ratio’、‘periodicity_threshold’、‘laws_set’、‘conservation_stats’、‘chi_ref’；
- ‘infer’ 输出的 ‘IntuitionHypothesis’ 进入 SICA R5 元层候选规则池；
- 与 ‘check’（附录 D）共享同一 ‘UniversalConstants’，不重复计算。

字段使用分工：‘omega_max’ 与 ‘time_arrow’ 为 ‘check’ 专用字段（涡旋校验、因果一致校验），‘infer’ 不直接使用它们；‘chi_ref’、‘gravity_threshold_ratio’、‘periodicity_threshold’、‘conservation_stats’ 为 ‘infer’ 专用字段（拓扑直觉、引力直觉、周期直觉、守恒直觉），‘check’ 不直接使用它们。

E.1.3 处理流程

``text
observations



E.2 宇宙常数校准

E.2.1 校准输出清单

‘UniversalConstants’ 的完整字段（含前置修订）：

字段	类型	量纲（几何制）	校准方法	默认值
c_light	float	1（无量纲， $c=1$ ）	$\max \Delta I /\Delta t \cdot I_0 \cdot I_0^{-1}$	1.0
h_planck	float	$\text{bit}^{1/2}[\text{L}]^{-2}$	$\min_t \Delta I(t) > 0$	0.001
delta_s_min	float	[L]	$\min(dx)$	1.0
time_arrow	float	1（无量纲）	对称 KL 散度差的均值	0.0
laws_set	List[str]	-	守恒律统计显著性筛选	[]
conservation_stats	dict	-	守恒律统计量	{}
K_max	float	1（无量纲）	曲率场 95% 分位数	1.0
omega_max	float	$[\text{L}]^{-1}$	涡旋场 95% 分位数	1.0
epsilon_gate	float	$\text{bit}^{1/2}[\text{L}]^{-2}$	信息通量密度残差 95% 分位数	0.001
delta_overlap	float	1（无量纲）	Overlap 历史分布的 5% 分位数	0.05
D	float	[L]	信息扩散模型最小二乘拟合	1.0
I_0	float	$\text{bit}^{1/2}[\text{L}]^{-2}$	参考信息密度	1.0
l_0	float	[L]	参考长度	1.0
clear_region_percentile	float	1（无量纲）	清晰区边界分位数	80.0
chi_ref	float	1（拓扑数）	参考非平凡拓扑欧拉示性数偏差	2.0
gravity_threshold_ratio	float	1（无量纲）	引力直觉触发阈值比	0.1
periodicity_threshold	float	1（无量纲）	周期直觉触发阈值	0.3

E.2.2 校准算法主流程

```

python

def calibrate(observations):

    """宇宙常数校准（附录 B B.3）"""

    constants = UniversalConstants()

    # 0. 默认值（若观测不足或不合规）

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

```

set_default_constants(constants)

# 1. 输入合法性检查
if len(observations) == 0:
    raise ValueError("observations 为空")

if any(not np.all(np.isfinite(o)) for o in observations):
    raise ValueError("observations 含 NaN/Inf")

# 2. 逐帧差分
delta_Is = [observations[t + 1] - observations[t]
             for t in range(len(observations) - 1)]

# 3. c_light: 最大信息密度变化率
if delta_Is:
    max_dI = max(np.max(np.abs(dI)) for dI in delta_Is)
    constants.c_light = float(max_dI * constants.l_0 / constants.I_0)

# 4. h_planck: 最小非零信息密度差
nonzero_dIs = [np.min(np.abs(dI[dI != 0]))
                for dI in delta_Is if np.any(dI != 0)]

if nonzero_dIs:
    constants.h_planck = float(np.min(nonzero_dIs))

# 5. delta_s_min: 最小空间间隔
constants.delta_s_min = infer_min_spacing(observations)

# 6. time_arrow: 对称 KL 散度差 (可正可负)
constants.time_arrow = calibrate_time_arrow(observations)

```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

```

# 7. laws_set + conservation_stats: 守恒律筛选

constants.laws_set, constants.conservations_stats = \
    calibrate_laws_set(observations)

# 8. D: 信息扩散模型拟合（在阈值校准之前完成）

D_fit = calibrate_diffusion(observations)

constants.D = D_fit if D_fit is not None else 1.0

# 9. K_max: 曲率场 95% 分位数

constants.K_max = calibrate_K_max(observations, constants.I_0,
                                   constants.l_0)

# 10. omega_max: 涡旋场 95% 分位数（依赖 D）

constants.omega_max = calibrate_omega_max(
    observations, constants.D, constants.I_0)

# 11. epsilon_gate: 信息守恒阈值（依赖 D）

constants.epsilon_gate = calibrate_epsilon_gate(
    observations, constants.D, constants.I_0, constants.l_0)

# 12. delta_overlap: 跨维隔离容许偏离量

constants.delta_overlap = calibrate_delta_overlap(
    observations, constants.I_0, constants.l_0)

return constants

def set_default_constants(c):
    """设置默认值（观测不足或不合规时使用）"""

```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com
 开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)
 开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

```

c.c_light = 1.0

c.h_planck = 1e-3

c.delta_s_min = 1.0

c.time_arrow = 0.0

c.laws_set = []

c.conservation_stats = {}

c.K_max = 1.0

c.omega_max = 1.0

c.epsilon_gate = 1e-3

c.delta_overlap = 0.05

c.D = 1.0

c.I_0 = 1.0

c.l_0 = 1.0

c.clear_region_percentile = 80.0

c.chi_ref = 2.0

c.gravity_threshold_ratio = 0.1

c.periodicity_threshold = 0.3

```

注记：`set\_default\_constants` 已设置全部 17 个字段的默认值；主流程仅在成功校准时覆盖对应字段。`I\_0`、`l\_0`、`clear\_region\_percentile`、`chi\_ref`、`gravity\_threshold\_ratio`、`periodicity\_threshold` 保持默认值，不在主流程末尾重复赋值。

E.2.3 各常数校准子算法

E.2.3.1 `time\_arrow`（对称 KL 散度差）

```

```python
def calibrate_time_arrow(observations):

 """时间箭头：正向与反向 KL 散度之差

 time_arrow = mean_t [D_KL(P_t || P_{t+1}) - D_KL(P_{t+1} || P_t)]

 - time_arrow > 0: 时间方向一致（前向）

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
 开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
 开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

```

- time_arrow ≈ 0 : 时序中性
- time_arrow < 0: 时序反转
"""

if len(observations) < 2:
 return 0.0

diff_values = []

for t in range(len(observations) - 1):
 P_t = normalize_histogram(observations[t])
 P_next = normalize_histogram(observations[t + 1])
 d_fwd = compute_KL_divergence(P_t, P_next)
 d_bwd = compute_KL_divergence(P_next, P_t)
 diff_values.append(d_fwd - d_bwd)

return float(np.mean(diff_values))

```

### E.2.3.2 `laws\_set` 与 `conservation\_stats`

```

```python
def calibrate_laws_set(observations):
    """守恒律筛选 + 统计量存储"""

    candidates = {
        "total_information": lambda o: float(np.sum(np.abs(o) ** 2)),
        "total_gradient": total_gradient_integral,
    }

    laws = []
    stats = {}

    for name, fn in candidates.items():
        values = [fn(o) for o in observations]
        mean = float(np.mean(values))
        std = float(np.std(values))
        stats[name] = {"mean": mean, "std": std, "count": len(values)}

        if abs(mean) > 0 and std / abs(mean) < 0.01:

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```

        laws.append(name)

    return laws, stats

def total_gradient_integral(o):
    """ $\int |\nabla I| dX$ """
    grads = np.gradient(o)
    if not isinstance(grads, (list, tuple)):
        grads = [grads]
    grad_mag = np.sqrt(sum(g ** 2 for g in grads) + 1e-24)
    return float(np.sum(grad_mag))

```

E.2.3.3 `infer\_min\_spacing`

```

``python
def infer_min_spacing(observations):
    """从观测形状推断最小空间间隔"""
    sample = observations[0]
    if sample.ndim < 2:
        return 1.0
    return 1.0

```

E.2.3.4 `calibrate\_diffusion` (数值稳定 + 上下界保护)

```

``python
def calibrate_diffusion(observations):
    """拟合信息扩散系数 D

    模型:  $\partial_t I = D \nabla^2 I$ 
    方法: 最小二乘拟合  $D = \Sigma(dI \cdot \nabla^2 I) / \Sigma(\nabla^2 I)^2$ 

    数值稳定:
    - 过滤 NaN/Inf

```

- 分母阈值判据 ($\text{den} < 1\text{e-}12$ 返回 None)
- 上界保护 ($|\text{D}| > 1\text{e}6$ 视为拟合失败)
- 返回值可为 0.0 (有效稳态拟合)

```

"""
if len(observations) < 3:
    return None

if any(not np.all(np.isfinite(o)) for o in observations):
    return None

num = 0.0
den = 0.0

for t in range(len(observations) - 1):
    I_t = observations[t]
    dI = observations[t + 1] - observations[t]
    laplacian = sum(
        np.gradient(np.gradient(I_t, axis=i), axis=i)
        for i in range(I_t.ndim)
    )
    num += float(np.sum(dI * laplacian))
    den += float(np.sum(laplacian ** 2))

if den < 1e-12:
    return None

D_fit = num / den

if not np.isfinite(D_fit) or abs(D_fit) > 1e6:
    return None

return D_fit

```

E.2.3.5 `calibrate\_omega\_max` (含 `D = 0.0` 退化处理)

```
```python
```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

```

def calibrate_omega_max(observations, D, I_0=1.0):

 """从观测序列校准涡旋有界阈值

 D = 0.0 退化：信息流速度场为零，涡旋无定义；
 改用观测空间二阶导数模的 95% 分位数作为替代特征。

 """

 if D == 0.0:

 return calibrate_omega_max_fallback(observations, I_0)

 vorticity_magnitudes = []

 for obs in observations:

 I_norm = obs / I_0

 grads = np.gradient(I_norm)

 if not isinstance(grads, (list, tuple)):

 grads = [grads]

 J_I = -D * np.stack(grads, axis=-1)

 v_I = J_I / (np.abs(I_norm)[..., None] + 1e-12)

 if I_norm.ndim == 2:

 vx, vy = v_I[..., 0], v_I[..., 1]

 omega = np.gradient(vy, axis=0) - np.gradient(vx, axis=1)

 vorticity_magnitudes.append(np.abs(omega).flatten())

 if not vorticity_magnitudes:

 return 1.0

 all_omega = np.concatenate(vorticity_magnitudes)

 return float(np.percentile(all_omega, 95))

def calibrate_omega_max_fallback(observations, I_0=1.0):

 """D = 0.0 时的涡旋阈值回退策略"""

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

```

magnitudes = []

for obs in observations:

 I_norm = obs / I_0

 if I_norm.ndim < 2:

 continue

 laplacian = sum(

 np.gradient(np.gradient(I_norm, axis=i), axis=i)

 for i in range(I_norm.ndim)

)

 magnitudes.append(np.abs(laplacian).flatten())

if not magnitudes:

 return 1.0

all_vals = np.concatenate(magnitudes)

return float(np.percentile(all_vals, 95))

```

### E.2.3.6 `calibrate\_epsilon\_gate` (含 `D = 0.0` 退化处理)

```

``python
def calibrate_epsilon_gate(observations, D, I_0=1.0, l_0=1.0):

 """从观测序列校准信息守恒阈值

 D = 0.0 退化：熵流密度为零，改用观测帧间差分作为替代特征。

 """

 if D == 0.0:

 return calibrate_epsilon_gate_fallback(observations, I_0)

 flux_diffs = []

 for t in range(len(observations) - 1):

 I_norm = observations[t] / I_0

 grads = np.gradient(I_norm)

 if not isinstance(grads, (list, tuple)):

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

```

 grads = [grads]

 J_I = -D * np.stack(grads, axis=-1)

 delta = compute_boundary_flux_diff(J_I)

 flux_diffs.append(delta)

 if not flux_diffs:

 return 1e-3

 return float(np.percentile(flux_diffs, 95))

def calibrate_epsilon_gate_fallback(observations, I_0=1.0):

 """D = 0.0 时的守恒阈值回退策略"""

 if len(observations) < 2:

 return 1e-3

 diffs = []

 for t in range(len(observations) - 1):

 dI = (observations[t + 1] - observations[t]) / I_0

 diffs.append(float(np.max(np.abs(dI))))

 if not diffs:

 return 1e-3

 return float(np.percentile(diffs, 95))

```

### E.2.3.7 `calibrate\_K\_max` / `calibrate\_delta\_overlap`

与附录 D D.3.3、D.7.4 的实现一致，此处不重复。`calibrate\_K\_max` 基于高斯曲率场取 95% 分位数；`calibrate\_delta\_overlap` 基于 Overlap 分布取 `1 - 5%` 分位数。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

E.2.4 量纲验证

常数	几何制量纲	验证
c_light	1（无量纲）	$ \Delta I /\Delta t \cdot I_0/I_0$ ，几何制下无量纲 ✓
h_planck	$\text{bit}^{1/2}[\text{L}]^{-2}$	与 I 同量纲 ✓
delta_s_min	[L]	空间间隔 ✓
time_arrow	1（无量纲）	对称 KL 差无量纲 ✓
K_max	1（无量纲）	归一化后 ✓
omega_max	$[\text{L}]^{-1}$	涡旋量纲 ✓
epsilon_gate	$\text{bit}^{1/2}[\text{L}]^{-2}$	与 $J_1$ 同量纲 ✓
delta_overlap	1（无量纲）	归一化偏离量 ✓
D	[L]	信息扩散系数 ✓
chi_ref	1（拓扑数）	整数或浮点 ✓
gravity_threshold_ratio	1（无量纲）	比率 ✓
periodicity_threshold	1（无量纲）	比率 ✓

E.3 物理直觉引擎

E.3.1 推理链总览

‘infer’ 接口执行四条推理链，从 D1–D4 流形特征推断环境底层物理规律：

规则类型	触发条件	物理语义	依赖字段
gravitational_well	D3 局部密度升高 → D2 曲率增大	类引力效应	K_max、 gravity_threshold_ratio
periodic_dynamics	D4 闭环状态转移	振荡/周期作用力	periodicity_threshold
conservation	D1 全局散度为零	某物理量守恒	laws_set、conservation_stats
topological	D3 欧拉示性数 $\chi \neq 2$	非平凡拓扑	chi_ref

E.3.2 主算法

```
```python
```

```

def infer(projected, constants):

    """物理直觉推理（附录 B B.6）"""

    rules = []

    # 1. 引力直觉

    if should_trigger_gravitational(projected, constants):

        conf = confidence_gravitational(projected, constants)

        rules.append(IntuitionRule(

            rule_type="gravitational_well",

            confidence=conf,

            triggered_features=["D3.density", "D2.curvature"],

        ))

    # 2. 周期直觉

    if should_trigger_periodic(projected, constants):

        conf = confidence_periodic(projected)

        rules.append(IntuitionRule(

            rule_type="periodic_dynamics",

            confidence=conf,

            triggered_features=["D4.evolution"],

        ))

    # 3. 守恒直觉

    if should_trigger_conservation(projected, constants):

        conf = confidence_conservation(constants)

        rules.append(IntuitionRule(

            rule_type="conservation",

            confidence=conf,

            triggered_features=["D1.divergence", "laws_set"],

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```

    ))

# 4. 拓扑直觉
if should_trigger_topological(projected):
    conf = confidence_topological(projected, constants)
    rules.append(IntuitionRule(
        rule_type="topological",
        confidence=conf,
        triggered_features=["D3.euler_char"],
    ))

# 综合置信度
if not rules:
    return IntuitionResult(rules=[], confidence=0.0,
                           intuition_type="no_intuition")

confidence = float(np.mean([r.confidence for r in rules]))
intuition_type = determine_intuition_type(rules)

return IntuitionResult(
    rules=rules,
    confidence=confidence,
    intuition_type=intuition_type,
)

```

E.4 各规则置信度

E.4.1 引力直觉

触发条件: $(\text{mean}(|K|) > K_{\max} \cdot \text{gravity_threshold_ratio})$

置信度:

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

$$conf = \min(1, \text{mean}(|K|)/K_{\max})$$

算法:

```
``python
def should_trigger_gravitational(projected, constants):

    K = projected.D2.curvature

    mean_abs_K = float(np.mean(np.abs(K)))

    threshold = constants.K_max * constants.gravity_threshold_ratio

    return mean_abs_K > threshold


def confidence_gravitational(projected, constants):

    K = projected.D2.curvature

    mean_abs_K = float(np.mean(np.abs(K)))

    if constants.K_max == 0:

        return 0.0

    return min(1.0, mean_abs_K / constants.K_max)
```

E.4.2 周期直觉

触发条件: D4 时序 FFT 谱峰占比 $> \text{periodicity_threshold}$

置信度:

$$conf = \frac{\max_{k \neq 0} \|\hat{f}(k)\|^2}{\sum_{k \neq 0} \|\hat{f}(k)\|^2}$$

算法:

```
``python
def should_trigger_periodic(projected, constants):

    if projected.D4.is_fake_time:

        return False

    return confidence_periodic(projected) > constants.periodicity_threshold
```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

```

def confidence_periodic(projected):
    """周期直觉置信度：沿时间轴的 FFT 谱峰占比

    对多维 evolution (H, W, T) 或 (H, W, D, T):
    先沿时间轴做空间平均投影得到 (T,) 一维信号，再做 FFT。
    排除直流分量 k=0。
    """
    evolution = projected.D4.evolution

    if evolution.size < 4:
        return 0.0

    if evolution.ndim == 1:
        signal = evolution
    else:
        signal = evolution.reshape(-1, evolution.shape[-1]).mean(axis=0)

    if signal.size < 4:
        return 0.0

    f = np.fft.fft(signal)
    power = np.abs(f) ** 2
    power_nonzero = power[1:]
    total = np.sum(power_nonzero)

    if total == 0:
        return 0.0

    return float(np.max(power_nonzero) / total)

```

E.4.3 守恒直觉

触发条件：`total\_information` 在 `laws\_set` 中

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

置信度:

$$conf = \max\left(0, 1 - \frac{std(J_{total})}{|mean(J_{total})|}\right)$$

统计量来自 `constants.conservation\_stats["total\_information"]`。

算法:

```
``python
def should_trigger_conservation(projected, constants):
    """守恒直觉触发条件

    projected 参数保留以备后续扩展。

    """
    return "total_information" in constants.laws_set

def confidence_conservation(constants):
    """守恒直觉置信度: 复用 calibrate 阶段统计量"""
    stats = constants.conservation_stats.get("total_information")
    if stats is None:
        return 0.0
    mean = stats.get("mean", 0.0)
    std = stats.get("std", 0.0)
    if abs(mean) < 1e-12:
        return 0.0
    conf = 1.0 - std / abs(mean)
    return max(0.0, min(1.0, conf))
```

E.4.4 拓扑直觉

触发条件: $(|\chi - 2| > 0)$

置信度:

$$conf = \min(1, |\chi - 2| / \chi_{ref})$$

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

算法:

```
```python
def should_trigger_topological(projected):
 return abs(projected.D3.euler_char - 2) > 0

def confidence_topological(projected, constants):
 chi = projected.D3.euler_char
 chi_ref = constants.chi_ref
 if chi_ref == 0:
 return 0.0
 return min(1.0, abs(chi - 2) / chi_ref)
```
```

E.5 主直觉类型判断

E.5.1 优先顺序

| 优先级 | 主类型 | 触发规则 |
|-----|------------------------|--------------------|
| 1 | geometric_gravity | gravitational_well |
| 2 | temporal_rhythm | periodic_dynamics |
| 3 | topological_navigation | topological |
| 4 | conservation_law | conservation |
| 5 | no_intuition | 无规则触发 |

E.5.2 算法

```
```python
def determine_intuition_type(rules):
 types = {r.rule_type for r in rules}
 if "gravitational_well" in types:
 return "geometric_gravity"
 elif "periodic_dynamics" in types:
```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

```

 return "temporal_rhythm"

elif "topological" in types:

 return "topological_navigation"

elif "conservation" in types:

 return "conservation_law"

else:

 return "no_intuition"

```

### E.5.3 综合置信度

$$confidence = \frac{1}{|R_{trig}|} \sum_{r \in R_{trig}} conf(r)$$

若  $R_{trig} = \emptyset$ ，则  $confidence = 0$ 。

### E.5.4 主类型与规则的映射（模块级统一定义）

```

``python
主直觉类型到 rule_type 的映射，在 E.5.4 统一定义，
由 E.7.3 引用。实现时置于模块级常量。
INTUITION_TYPE_TO_RULE = {

 "geometric_gravity": "gravitational_well",

 "temporal_rhythm": "periodic_dynamics",

 "topological_navigation": "topological",

 "conservation_law": "conservation",

}

```

**约定：**该映射表为模块级常量，E.5.4 与 E.7.3 共享，不重复定义。

## E.6 异常条件

### E.6.1 `calibrate`

条件	异常类型	说明
<code>len(observations) == 0</code>	ValueError	无观测输入
观测序列形状不一致	ValueError	无法逐帧差分
观测序列含 NaN/Inf	ValueError	数值不合法

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

### E.6.2 `infer`

条件	异常类型	说明
projected.D2 缺失	ValueError	无法计算曲率
projected.D4.is_fake_time=True	RuntimeWarning	时序为伪造，周期直觉跳过
constants 为 None	ValueError	未执行校准

## E.7 返回结构与 SICA 对接

### E.7.1 `IntuitionResult`

字段	类型	说明
rules	List[IntuitionRule]	触发的物理直觉规则
confidence	float	综合置信度 [0,1]
intuition_type	str	主直觉类型

### E.7.2 `IntuitionRule`

字段	类型	说明
rule_type	str	gravitational_well、periodic_dynamics、conservation、topological
confidence	float	单规则置信度 [0,1]
triggered_features	List[str]	触发的流形特征

### E.7.3 `IntuitionHypothesis` 映射

```
```python
def to_intuition_hypothesis(intuition_result, projected, constants):
    """将推理结果映射为 IntuitionHypothesis（附录 B B.7.1）

    按 intuition_type 选取对应规则，避免 rules[0] 与主类型不一致。
    引用 E.5.4 的 INTUITION_TYPE_TO_RULE 模块级常量。
    """

    if not intuition_result.rules:
        return None
```

```

target_rule_type = INTUITION_TYPE_TO_RULE.get(
    intuition_result.intuition_type)

if target_rule_type is None:
    return None

top_rule = next((r for r in intuition_result.rules
                 if r.rule_type == target_rule_type), None)

if top_rule is None:
    return None

return IntuitionHypothesis(
    rule_type=top_rule.rule_type,
    confidence=top_rule.confidence,
    triggered_features=top_rule.triggered_features,
    holdout_delta=0.0, # 由 SICA R2 在 holdout 上评估后回填
)

```

E.7.4 SICA R2 门禁对接

‘IntuitionHypothesis’ 进入 SICA R5 元层后：

- 由 SICA R2 性能门禁在 holdout 集上评估 ‘holdout_delta’；
- 若 ‘holdout_delta $\geq \delta$ ’（SICA R2 阈值），纳入 ‘H_t’；
- 否则丢弃，记为负样本。

E.8 实现注记

E.8.1 常数校准的一致性

‘calibrate’ 输出的 ‘UniversalConstants’ 在 ‘check’（附录 D）与 ‘infer’ 中共享。

字段使用分工：

字段	‘check’使用	‘infer’使用
----	-----------	-----------

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

字段	`check`使用	`infer`使用
`K_max`	✓ 曲率有界	✓ 引力直觉
`omega_max`	✓ 涡旋校验	—
`epsilon_gate`	✓ 信息守恒	—
`delta_overlap`	✓ 跨维隔离	—
`time_arrow`	✓ 因果一致	—
`laws_set`	—	✓ 守恒直觉触发
`conservation_stats`	—	✓ 守恒直觉置信度
`chi_ref`	—	✓ 拓扑直觉置信度
`gravity_threshold_ratio`	—	✓ 引力直觉触发
`periodicity_threshold`	—	✓ 周期直觉触发
`clear_region_percentile`	✓ 清晰区边界	—
`I_0`、`l_0`	✓ 归一化	✓ 归一化
`D`	✓ 熵流计算	—

`chi\_ref`、`gravity\_threshold\_ratio`、`periodicity\_threshold`、`conservation\_stats` 为 `infer` 专用字段；`omega\_max`、`time\_arrow` 为 `check` 专用字段。二者在 `UniversalConstants` 中统一定义，保证常量一致性。

`time\_arrow` 与周期直觉的关系：`infer` 的周期直觉基于 FFT 谱峰占比，不直接依赖 `time\_arrow`。`time\_arrow` 仅在 `check` 的因果一致校验中使用。

E.8.2 置信度范围

`confidence` $\in[0,1]$ 。各规则置信度：

规则	置信度范围
引力直觉	$\min 1, \text{mean}(K)/K_{\max}$
周期直觉	谱峰占比 $[0,1]$
守恒直觉	$\max 0, 1 - \text{std}/ \text{mean} $
拓扑直觉	$\min 1, \chi - 2 /\chi_{\text{ref}}$

综合置信度为空规则集合时取 0。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

E.8.3 与 SICAR1–R10 的对接

`IntuitionHypothesis` 进入 SICA 后，依次经过：

- 1. R1（锚点不可伪造）：确保 `triggered\_features` 来自真实观测；
- 2. R2（性能门禁）：在 holdout 上评估 `holdout\_delta`；
- 3. R3（复杂度惩罚）：若新增规则导致复杂度超界，拒绝；
- 4. R9（证据阈值）：至少 3 次独立观测支撑；
- 5. R10（跨族泛化）：至少在 2 个游戏族上验证。

E.8.4 与其他附录的交叉引用

交叉项	目标位置
常数定义	附录 A A.3.3、A.3.4
接口定义	附录 B B.3、B.6
D1–D4 特征	附录 C C.3–C.6
门禁阈值共享	附录 D D.12.1
SICA 规则	《衔尾蛇 SICA》R1–R10

附录 F：DPL→SICA 数据结构与灵境协议栈映射

说明：本附录定义 DPL 与 SICA（《衔尾蛇-自指迭代收敛算法》）R5 元层、灵境引擎协议栈（v13.2）的完整映射。涵盖三类对接 SICA 的数据结构、与灵境协议 1–11 的接口对应、与 SICAR1–R10 规则的约束流程。所有符号、量纲回溯附录 A；接口定义回溯附录 B；D1–D4 特征回溯附录 C；门禁输出回溯附录 D；校准与直觉输出回溯附录 E。本附录不引入新物理假设，仅为已有数据流与协议映射的工程规范。

符号约定：标量取绝对值 $| \cdot |$ ；向量/复数取模 $| \cdot |$ 。

F.1 总览

F.1.1 DPL 在灵境引擎中的位置

DPL 位于灵境 R2 感知层，向下游 Agent 决策层（R3 规划、R4 资源分配）与 SICA R5 元层提供结构化流形特征与物理直觉。DPL 的五个接口（附录 B B.1）通过灵境协议 4、5、7.1、7.2、9 与场方程对接。

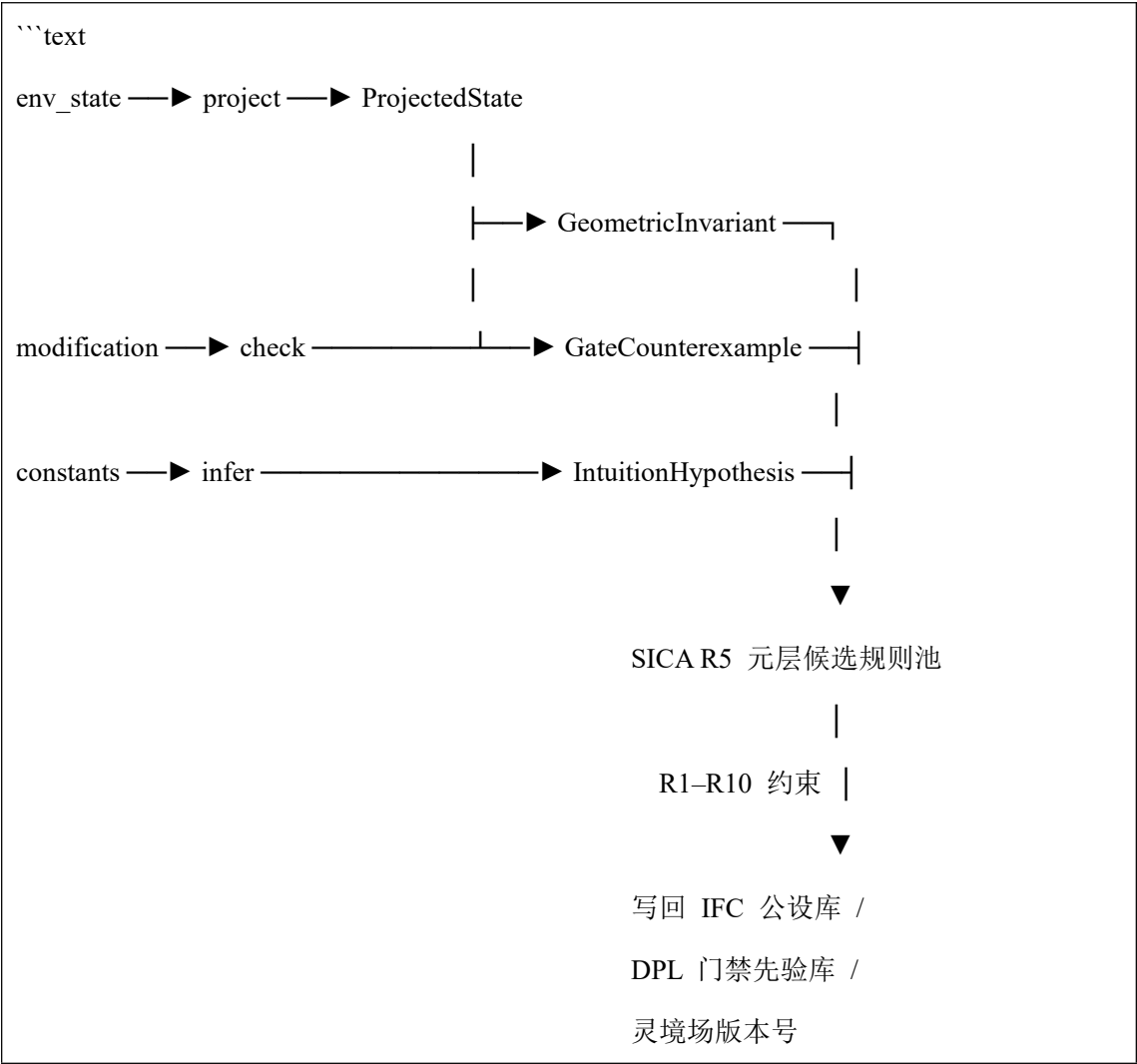
作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

F.1.2 三类对接 SICA 的数据结构

结构	来源	含义
GeometricInvariant	project 输出 (D1–D4 特征)+ 稳定性过滤	几何不变式，作为 SICA 候选规则 H_t 的输入
GateCounterexample	check 输出 (门禁失败)	门禁反例，作为 SICA 负样本
IntuitionHypothesis	infer 输出 (物理直觉)	直觉假设，作为 SICA 候选规则

三类结构均进入 SICA R5 元层候选规则池，由 R1–R10 规则约束后写回。

F.1.3 数据流



F.2 三类数据结构定义

F.2.1 `GeometricInvariant`（几何不变式）

来源：`project` 接口输出的 D1–D4 流形特征，经稳定性过滤后提取。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：https://github.com/LingJing1001/Lingjing-Solo （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

字段定义：

字段	类型	说明
pattern_type	str	几何模式类型：stable_curvature、persistent_vorticity、periodic_evolution 等
dim_layer	str	来源层：D1、D2、D3、D4
stability_score	float	稳定性得分 [0,1]，基于多帧持续观测
evidence_count	int	支撑案例数（跨帧、跨环境），非负
source_family	str	来源环境族标识，由调用方传入，用于 R10 跨族泛化
features	dict	扩展字段 （附录 F 定义），模式特征值（如曲率均值、涡旋幅度等）

注记（`features` 字段）：`features` 为附录 F 的扩展字段，用于存储模式的具体特征值。附录 B B.7.1 的 `GeometricInvariant` 字段定义应同步补充 `features: dict`（可选）。

注记（`source\_family`）：`source\_family` 由调用方在提取时填充。若为 `"unknown"`，R10 跨族泛化无法执行，该结构标记为待分类。

提取算法：

```
```python
def extract_geometric_invariants(projected_list, source_family,
 threshold=0.7):
 """从多帧 ProjectedState 序列提取几何不变式

 参数：
 - projected_list: 多帧 ProjectedState 列表
 - source_family: 来源环境族标识（如 "family_A"）
 - threshold: 稳定性得分阈值，低于该值不纳入

 """
 invariants = []

 # D2 稳定曲率（标量）
 curvatures = [p.D2.curvature for p in projected_list]
```

```

stability = compute_stability(curvatures)

if stability > threshold:

 invariants.append(GeometricInvariant(

 pattern_type="stable_curvature",

 dim_layer="D2",

 stability_score=stability,

 evidence_count=len(curvatures),

 source_family=source_family,

 features={"mean_curvature": float(np.mean(curvatures))},

))

D2 持续涡旋：先取模（兼容标量与向量）

vorticity_magnitudes = [extract_vorticity_magnitude(p)

 for p in projected_list]

stability = compute_stability(vorticity_magnitudes)

if stability > threshold:

 invariants.append(GeometricInvariant(

 pattern_type="persistent_vorticity",

 dim_layer="D2",

 stability_score=stability,

 evidence_count=len(vorticity_magnitudes),

 source_family=source_family,

 features={

 "mean_vorticity": float(np.mean(vorticity_magnitudes)),

 },

))

D4 周期演化

if len(projected_list) >= 4:

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

```

 evolutions = [p.D4.evolution for p in projected_list]

 periodic = compute_periodicity(evolutions)

 if periodic > threshold:

 invariants.append(GeometricInvariant(

 pattern_type="periodic_evolution",

 dim_layer="D4",

 stability_score=periodic,

 evidence_count=len(evolutions),

 source_family=source_family,

 features={"periodicity": periodic},

))

 return invariants

```

```

def extract_vorticity_magnitude(p):

 """从 D2Surface 提取涡旋模（兼容标量与向量）

 依据 D2Surface.vorticity_is_vector 显式标志判断。

 """

 omega = p.D2.vorticity

 if p.D2.vorticity_is_vector:

 return np.linalg.norm(omega, axis=-1)

 return np.abs(omega)

```

```

def compute_stability(sequence):

 """多帧稳定性得分：1 - 变异系数（归一化到 [0,1]）

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

输入应过滤 NaN/Inf; 对全零序列返回 0。

```
"""
```

```
if len(sequence) < 2:
```

```
 return 0.0
```

```
means = []
```

```
for s in sequence:
```

```
 if not np.all(np.isfinite(s)):
```

```
 return 0.0
```

```
 means.append(float(np.mean(s)))
```

```
std = float(np.std(means))
```

```
mean = float(np.mean(means))
```

```
if abs(mean) < 1e-12:
```

```
 return 0.0
```

```
return max(0.0, 1.0 - std / abs(mean))
```

```
def compute_periodicity(evolution):
```

```
 """多帧周期度: 基于 FFT 谱峰占比 (沿时间轴空间平均投影) """
```

```
 if len(evolution) < 4:
```

```
 return 0.0
```

```
 signal = np.array([float(np.mean(e)) for e in evolution])
```

```
 if signal.size < 4:
```

```
 return 0.0
```

```
 f = np.fft.fft(signal)
```

```
 power = np.abs(f) ** 2
```

```
 power_nonzero = power[1:] # 排除直流分量
```

```
 total = np.sum(power_nonzero)
```

```
 if total == 0:
```

```
 return 0.0
```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

```
return float(np.max(power_nonzero) / total)
```

**量纲检查：**`features` 中各字段量纲与对应流形特征一致（曲率 1、涡旋  $[L]^{-1}$ 、周期 1）。

### F.2.2 `GateCounterexample`（门禁反例）

**来源：**`check` 接口门禁失败时的输出，映射见附录 D D.10.2。

**字段定义：**

字段	类型	说明
gate_type	str	违反的校验项：INFO_NOT_CONSERVED、EXTREME_CURVATURE 等
violation	float	违反幅度（归一化到 [0,1]）
raw_dim	float	原始维度（dim_info.spatial）
effective_dim	int	有效维度
sample_hash	str	观测样本哈希，SHA-256 十六进制字符串（64 字符）
timestamp	int	时间戳（纳秒）

**映射算法：**见附录 D D.10.2 `to\_gate\_counterexample`。

**量纲检查：**所有字段无量纲或计数。

### F.2.3 `IntuitionHypothesis`（直觉假设）

**来源：**`infer` 接口输出，映射见附录 E E.7.3。

**字段定义：**

字段	类型	说明
rule_type	str	直觉类型：gravitational_well、periodic_dynamics 等
confidence	float	置信度 [0,1]
triggered_features	List[str]	触发的流形特征
holdout_delta	float	在 holdout 集上的性能增益（由 SICA R2 评估后回填）

**映射算法：**见附录 E E.7.3 `to\_intuition\_hypothesis`。

**量纲检查：**`confidence` 与 `holdout\_delta` 均无量纲。

## F.3 与灵境协议栈的映射

### F.3.1 协议 1-11 总览

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）  
开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

灵境引擎的工程实现以 L0-L5 层级与协议 1-11 为骨架（详见《智能仿真的全息原理引擎》v13.2）。DPL 作为 R2 感知层嵌入其中。

协议	名称
协议 1	时空基座（动态网络）
协议 2	信息密度与可求解量
协议 3	信息密度决定时空弯曲
协议 4	Agent 求解场→感知
协议 5	Agent 决策/动作写回
协议 6	多智能体共识
协议 7.1	场自然演化
协议 7.2	可见性门控（面积律）
协议 8	Agent 身份与记忆
协议 9	迷雾展开
协议 10	外部接入
协议 11	人类交互

### F.3.2 DPL 接口与协议的对应

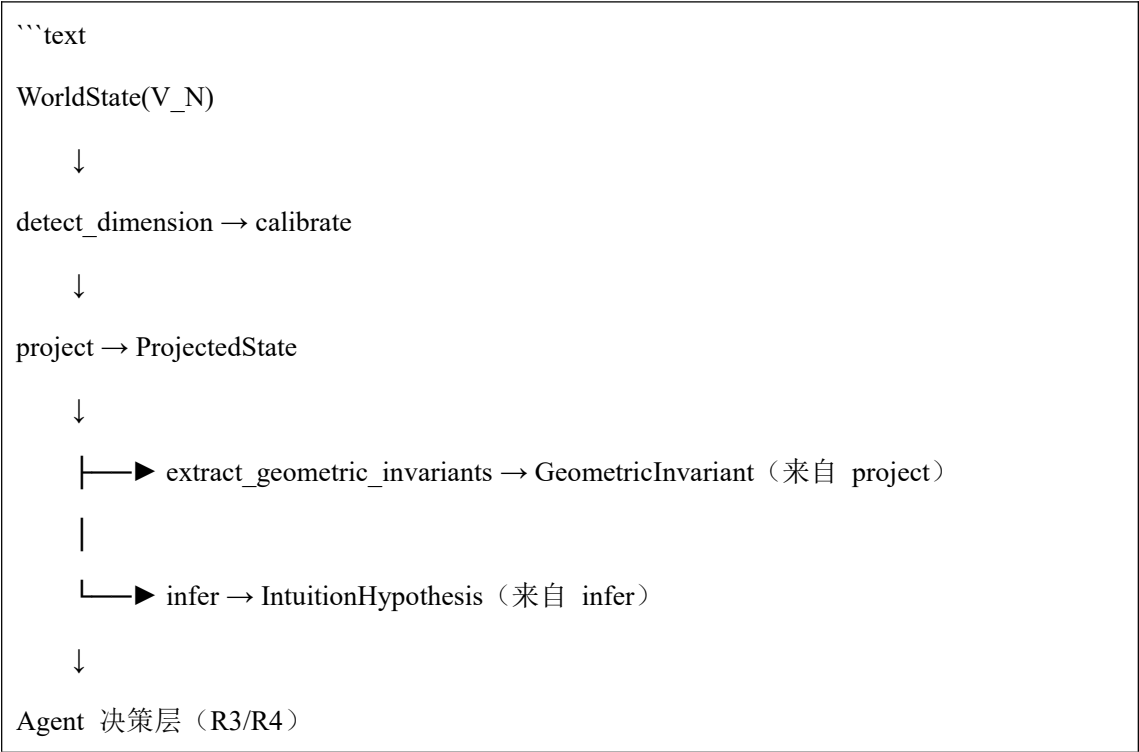
灵境协议	DPL 接口	对接方式
协议 1（时空基座）	detect_dimension	环境输入与维度检测建立在动态网格上
协议 2（信息密度）	calibrate	UniversalConstants.I_0 与协议 2 的信息密度场共享
协议 3（时空弯曲）	project	D1-D2 流形分解为几何生成方程提供局部信息密度与曲率
协议 4（Agent 求解场→感知）	project	感知入口：Agent 读取场版本 $V_N$ ，DPL 输出 ProjectedState
协议 5（动作写回）	check	门禁结果决定动作是否写回；被拦截的 Action 作为 GateCounterexample
协议 6（多智能体共识）	check	门禁结果参与版本化共识；所有 Agent 在同一 $V_N$ 上执行 DPL
协议 7.1（场自然演	project	D3-D4 建立在场自然演化的时间推进之上

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
 开源 Github 链接：https://github.com/LingJing1001/Lingjing-Solo （团队持续开放迭代）  
 开源 Github 链接：https://github.com/LingJing1001/Lingjing\_Engineering （团队持续开放迭代）

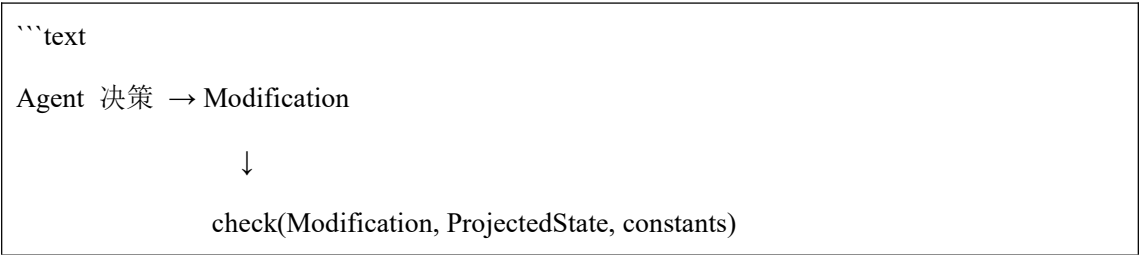
灵境协议	DPL 接口	对接方式
化)		
协议 7.2 (可见性门控)	project	清晰区执行 D1-D4 全流形分解, 迷雾区仅保留低维统计量
协议 8 (身份与记忆)	infer	IntuitionHypothesis 作为 L4 意识相变的候选证据, 与协议 8 的 C 值监测对接
协议 9 (迷雾展开)	project	2.5D/3D 升维机制与迷雾展开协同
协议 10 (外部接入)	全部	IntuitionHypothesis 作为外部框架感知输入
协议 11 (人类交互)	全部	人类指令经协议 11 转为 Action 后仍须通过 check

F.3.3 典型数据流

感知流 (协议 4) :



动作流 (协议 5) :



└─ pass → 写回场方程（协议 5）  
└─ block → GateCounterexample → SICA R5 元层  
└─ dimension\_lift → 触发升维补全（协议 9）  
└─ dimension\_reduce → 触发降维抽象

共识流（协议 6）：

```text

所有 Agent 在同一 V_N 上：

detect_dimension → calibrate → project → check

↓

门禁结果汇总 → 版本化共识

F.4 与 SICA R1–R10 的对接

F.4.1 SICA R1–R10 总览

SICA（《衔尾蛇-自指迭代收敛算法》）位于灵境 R5 元层，通过 R1–R10 规则约束自指修改过程。核心是 Lyapunov 势函数：

$$V_t = -P_t + \lambda C_t + \mu N_{\text{viol}}(t)$$

其中：

- V_t 为 SICA 势函数，区别于信息密度场 $\Phi(x)$ ；
- P_t 为留存集性能；
- C_t 为复杂度；
- $N_{\text{viol}}(t)$ 为边界违规次数。

F.4.2 三类数据结构与 R1–R10 的对接

注记（R 规则的适用范围）：

- R5（不可变核心）约束**写回动作**——禁止修改环境接口、安全闸等核心组件；不直接约束候选规则本身。
- R8（探索配额）约束**执行策略**——保留 5% 步数用于探索；不直接约束候选规则本身。
- R4（递归深度限制）约束**元修改**（对修改逻辑的修改）；对三类数据结构均不适用。
- R6（衰减淘汰）为**独立维护过程**——每轮迭代后对 `H\_t` 全量规则与负样本池执行衰减；不在写回路径上阻断候选规则。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

| 数据结构 | R1 | R2 | R3 | R4 | R5 | R6 | R7 | R8 | R9 | R10 |
|---------------------|----|----|----|----|----|----|----|----|----|-----|
| GeometricInvariant | ✓ | ✓ | ✓ | — | — | ✓ | ✓ | — | ✓ | ✓ |
| GateCounterexample | ✓ | ✓ | — | — | — | ✓ | — | — | ✓ | — |
| IntuitionHypothesis | ✓ | ✓ | ✓ | — | — | ✓ | ✓ | — | ✓ | ✓ |

R1（锚点不可伪造）：

- ‘GeometricInvariant’ 来自 ‘project’ 的真实观测，由 ‘projected_list’ 计算，不可由模型自行写入；
- ‘GateCounterexample’ 来自 ‘check’ 的真实门禁失败，由 ‘modification’ 与 ‘projected’ 确定；
- ‘IntuitionHypothesis’ 来自 ‘infer’ 的真实直觉，由 ‘projected’ 与 ‘constants’ 确定。

R2（性能门禁）：

- 三类数据结构的 ‘holdout_delta’ 由 SICA R2 在 holdout 集上评估；
- ‘GeometricInvariant’ 与 ‘IntuitionHypothesis’ 的 ‘holdout_delta $\geq \delta$ ’（SICA R2 阈值）时纳入 ‘H_t’；
- ‘GateCounterexample’ 作为负样本，评估是否带来正增益。

R3（复杂度惩罚）：

- ‘GeometricInvariant’ 与 ‘IntuitionHypothesis’ 进入 ‘H_t’ 前，复杂度增量 ΔC 需满足 $\Delta P \geq \lambda \Delta C$ ；
- ‘GateCounterexample’ 不引入新规则，不参与复杂度计算。

R6（衰减淘汰，独立维护）：

- ‘GeometricInvariant’ 与 ‘IntuitionHypothesis’ 连续 N 局未触发则降权，连续 M 局未触发则淘汰；
- ‘GateCounterexample’ 作为负样本同样参与衰减；
- R6 由 SICA R5 元层在每轮迭代结束后独立执行，不在写回路径上阻断候选规则。

R7（版本化 + 自动回滚）：

- ‘GeometricInvariant’ 与 ‘IntuitionHypothesis’ 纳入 ‘H_t’ 时自动提交版本快照；
- ‘GateCounterexample’ 不触发版本提交（仅作为负样本）。

R9（证据阈值）：

- ‘GeometricInvariant’ 的 ‘evidence_count ≥ 3 ’；

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

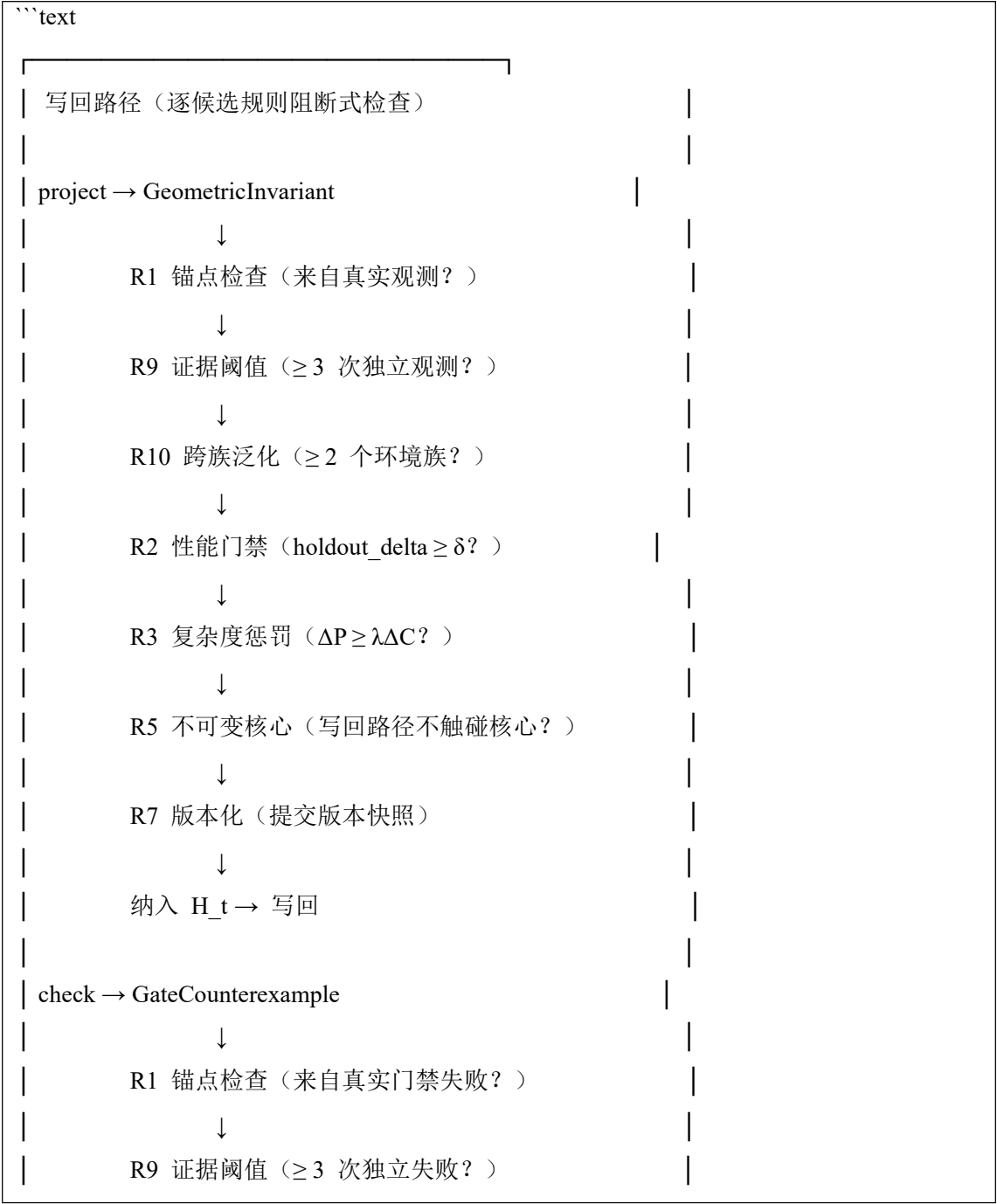
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

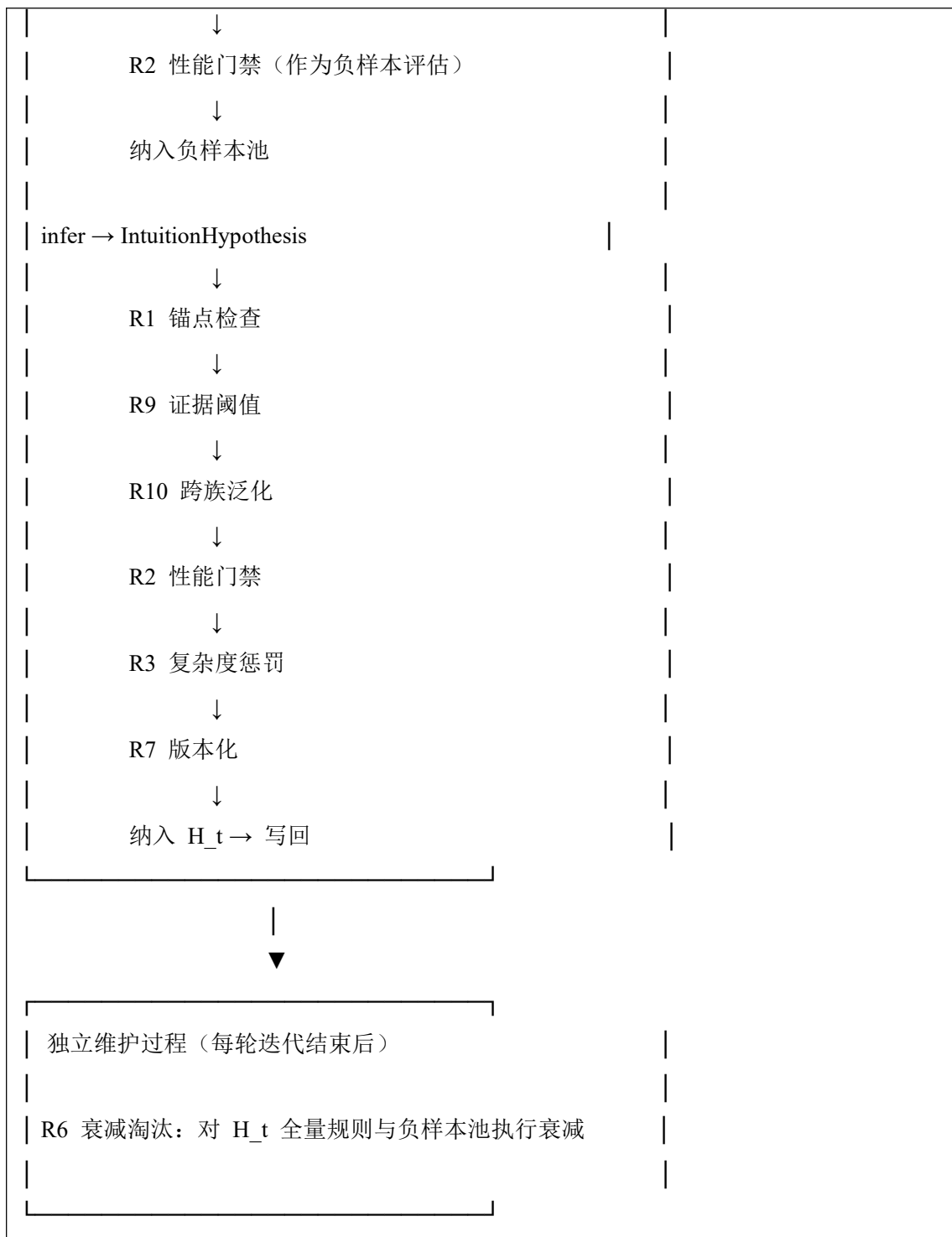
- `IntuitionHypothesis` 需至少 3 次独立观测支撑；
- `GateCounterexample` 也需至少 3 次独立失败，避免单次噪声。

R10（跨族泛化）：

- `GeometricInvariant` 与 `IntuitionHypothesis` 至少在 2 个环境族上验证；
- 未通过的标记为“特化规则”，隔离存储；
- `GateCounterexample` 不适用（作为负样本无须泛化）。

F.4.3 数据流与 R1-R10 的时序





F.4.4 势函数耦合

DPL 与 SICA 势函数 $V_t = -P_t + \lambda C_t + \mu N_{viol}(t)$ 的三项耦合:

| 势函数项 | DPL 贡献 |
|---------------|---|
| P_t (留存集性能) | `GeometricInvariant`、`IntuitionHypothesis` 在 holdout 上的性能增益 |

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering (团队持续开放迭代)

| 势函数项 | DPL 贡献 |
|------------------------|-----------------------------|
| C_t （复杂度） | D1–D4 流形特征的复杂度（层数、特征维度、规则数） |
| $N_{viol}(t)$ （边界违规次数） | `check` 门禁违规计数 |

F.5 写回路径

F.5.1 三类写回目标

通过 R1–R10 约束的数据结构写回以下目标：

| 目标 | 写入内容 | 用途 |
|-----------|---------------------------|------------|
| IFC 公设库 | GeometricInvariant | 更新几何不变式先验 |
| DPL 门禁先验库 | GateCounterexample（作为负样本） | 更新门禁阈值与规则 |
| 灵境场版本号 | 所有通过约束的更新 | 协议 6 版本化共识 |

F.5.2 写回算法

```
```python
def write_back_to_sica(structures, sica_state):
 """将通过 R1–R10 约束的数据结构写回 SICA 状态

 structures: 三类数据结构的列表
 sica_state: SICA R5 元层状态（含 H_t、版本号等）
 """
 for struct in structures:
 # R1 锚点检查
 if not r1_check(struct):
 continue

 # R9 证据阈值
 if not r9_check(struct):
 continue

 # R10 跨族泛化（仅候选规则）

 if isinstance(struct, (GeometricInvariant, IntuitionHypothesis)):
```

```

 if not r10_check(struct):

 mark_as_specialized(struct)

 continue

R2 性能门禁（holdout_delta 由 SICA R2 评估）
if not r2_check(struct, sica_state.holdout):

 continue

R3 复杂度惩罚（仅候选规则）
if isinstance(struct, (GeometricInvariant, IntuitionHypothesis)):

 if not r3_check(struct, sica_state.H_t):

 continue

R5 不可变核心（写回路径检查）
if touches_immutable_core(struct):

 continue

R7 版本化（仅候选规则纳入 H_t 时提交）
if isinstance(struct, (GeometricInvariant, IntuitionHypothesis)):

 commit_version(sica_state)

写回
if isinstance(struct, GeometricInvariant):

 sica_state.ifc_axioms.update(struct)

elif isinstance(struct, GateCounterexample):

 sica_state.gate_priors.add_negative(struct)

elif isinstance(struct, IntuitionHypothesis):

 sica_state.H_t.append(struct)

R6 衰减淘汰（独立维护过程，每轮迭代结束后执行）
apply_decay_and_eviction(sica_state)

return sica_state

```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

### F.5.3 版本化与协议 6 对接

每次候选规则纳入 `H\_t` 时递增灵境场版本号：

```
```python
def commit_version(sica_state):
    """版本化提交（协议 6）"""
    sica_state.version += 1
    sica_state.timestamp = time.time_ns()
    return sica_state.version
```

所有 Agent 在下一次 `project` 时读取新版本号 V_{N+1} ，实现无通讯共识。

F.6 异常条件与边界处理

F.6.1 数据结构异常

触发条件	异常类型	说明
GeometricInvariant.pattern_type 未注册	ValueError	未知模式类型
GateCounterexample.gate_type 未注册	ValueError	未知门禁类型
IntuitionHypothesis.rule_type 未注册	ValueError	未知直觉类型
stability_score 或 confidence 超出[0,1]	ValueError	数值越界
evidence_count < 0	ValueError	证据计数非负
sample_hash 长度不为 64	ValueError	SHA-256 格式错误

F.6.2 协议对接异常

条件	异常类型	说明
协议 4 读取的场版本号与 Agent 期望不一致	RuntimeError	版本不同步
协议 5 门禁结果写回时场版本已变	RuntimeError	版本冲突
协议 6 共识帧合成时锚点间重叠度 $< \epsilon_{crit}$	RuntimeWarning	联盟分裂
协议 9 迷雾展开的截断误差超出阈值	RuntimeWarning	保真度下降

F.6.3 SICA R1-R10 异常

条件	异常类型	说明
----	------	----

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

条件	异常类型	说明
连续 5 次修改被 R2 拒绝	RuntimeError	触发人工审查(阈值来自 SICA 论文 R2 第 6 节)
连续 5 次修改触发 R8 回滚	RuntimeError	触发人工审查（阈值同 SICA 论文 R8）
R7 回滚后性能仍未恢复	RuntimeError	触发人工审查
R4 元修改递归深度>K	ValueError	元修改超限（K=2，见 SICA 论文 R4）

注记（“5”的来源）：连续 5 次修改被拒/触发回滚的阈值来自 SICA 论文 R2、R8 第 6 节的定义（连续 5 次修改被拒：暂停自动合并 1 小时，触发人工审查流程）。附录 F 沿用该阈值，不另行设定。

F.7 实现注记

F.7.1 数据结构一致性

三类数据结构的字段定义与附录 B B.7.1、附录 D D.10.2、附录 E E.7.3 完全一致，不重复定义。

字段来源链：

数据结构	字段来源
GeometricInvariant	project 输出的 D1–D4 特征，经稳定性过滤；features 为附录 F 扩展字段
GateCounterexample	check 输出的 GateResult 映射（附录 D D.10.2）
IntuitionHypothesis	infer 输出的 IntuitionResult 映射（附录 E E.7.3）

F.7.2 灵境协议栈对齐

DPL 五个接口（附录 B B.1）与灵境协议 1–11 的对应关系（F.3.2）保证：

- 协议 4（感知）与 `project` 严格对接；
- 协议 5（写回）与 `check` 严格对接；
- 协议 6（共识）与版本化流程对接；
- 协议 7.1、7.2、9 与 `project` 的 D1–D4 分解与迷雾展开对接；
- 协议 8、10、11 与 `infer` 的直觉假设对接。

F.7.3 SICA R1–R10 约束流程

三类数据结构进入 SICA R5 元层后，写回路径依次经过：

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

- 1. **R1 锚点不可伪造**：确保来自真实观测/门禁/直觉；
- 2. **R9 证据阈值**：至少 3 次独立观测支撑；
- 3. **R10 跨族泛化**（仅候选规则）：至少在 2 个环境族上验证；
- 4. **R2 性能门禁**：holdout 上性能增益 $\geq \delta$ ；
- 5. **R3 复杂度惩罚**（仅候选规则）：复杂度增量 ΔC 满足 $\Delta P \geq \lambda \Delta C$ ；
- 6. **R5 不可变核心**：写回路径不触碰环境接口、安全闸等；
- 7. **R7 版本化**（仅候选规则）：提交版本快照。

独立维护过程（每轮迭代结束后）：

- 8. **R6 衰减淘汰**：对 'H_t' 全量规则与负样本池执行衰减。

R4（元修改）与 R8（探索配额）对三类数据结构不直接适用。写回路径（F.5）保证数据一致性。

F.7.4 前置修订汇总

修订项	目标位置	说明
D1FlowLine 增加 gradient_vector 字段	附录 C C.3.1	用于信息守恒与跨维隔离校验
D2Surface 增加 vorticity_is_vector 字段	附录 C C.4.1	用于涡旋校验与 GeometricInvariant 提取
UniversalConstants 增加 clear_region_percentile	附录 B B.3.3	清晰区边界阈值分位数
UniversalConstants 增加 l_0、l_0	附录 B B.3.3	参考信息密度与参考长度
UniversalConstants 增加 chi_ref	附录 B B.3.3	拓扑直觉参考偏差
UniversalConstants 增加 gravity_threshold_ratio	附录 B B.3.3	引力直觉触发阈值比
UniversalConstants 增加 periodicity_threshold	附录 B B.3.3	周期直觉触发阈值
UniversalConstants 增加 conservation_stats	附录 B B.3.3	守恒律统计量
GeometricInvariant 增加 features 字段	附录 B B.7.1	模式特征值（可选）

F.7.5 与其他附录的交叉引用

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

交叉项	目标位置
符号与量纲	附录 A
接口定义	附录 B B.1、B.3、B.4、B.5、B.6
D1–D4 特征	附录 C C.3–C.6
门禁实现	附录 D D.2–D.8
门禁输出映射	附录 D D.10.2
校准与直觉	附录 E E.2–E.4
直觉输出映射	附录 E E.7.3
灵境协议栈	《智能仿真的全息原理引擎》v13.2
SICA 规则	《衔尾蛇 SICA》R1–R10

附录 G：“灵境计划”——灵境引擎完整世界观架构

说明：本附录定义“灵境计划”——灵境引擎（Lingjing Engine）的完整世界观架构，统一以 IFC 三公理体系与 DPL 五接口规范为术语基础，整合单体智能、多智能体仿真与 SICA 自指迭代三层架构。所有符号、量纲回溯附录 A；接口定义回溯附录 B；D1–D4 特征回溯附录 C；门禁输出回溯附录 D；校准与直觉输出回溯附录 E；SICA 映射回溯附录 F。本附录不引入新物理假设，仅为已有理论架构与工程接口的系统化表达。

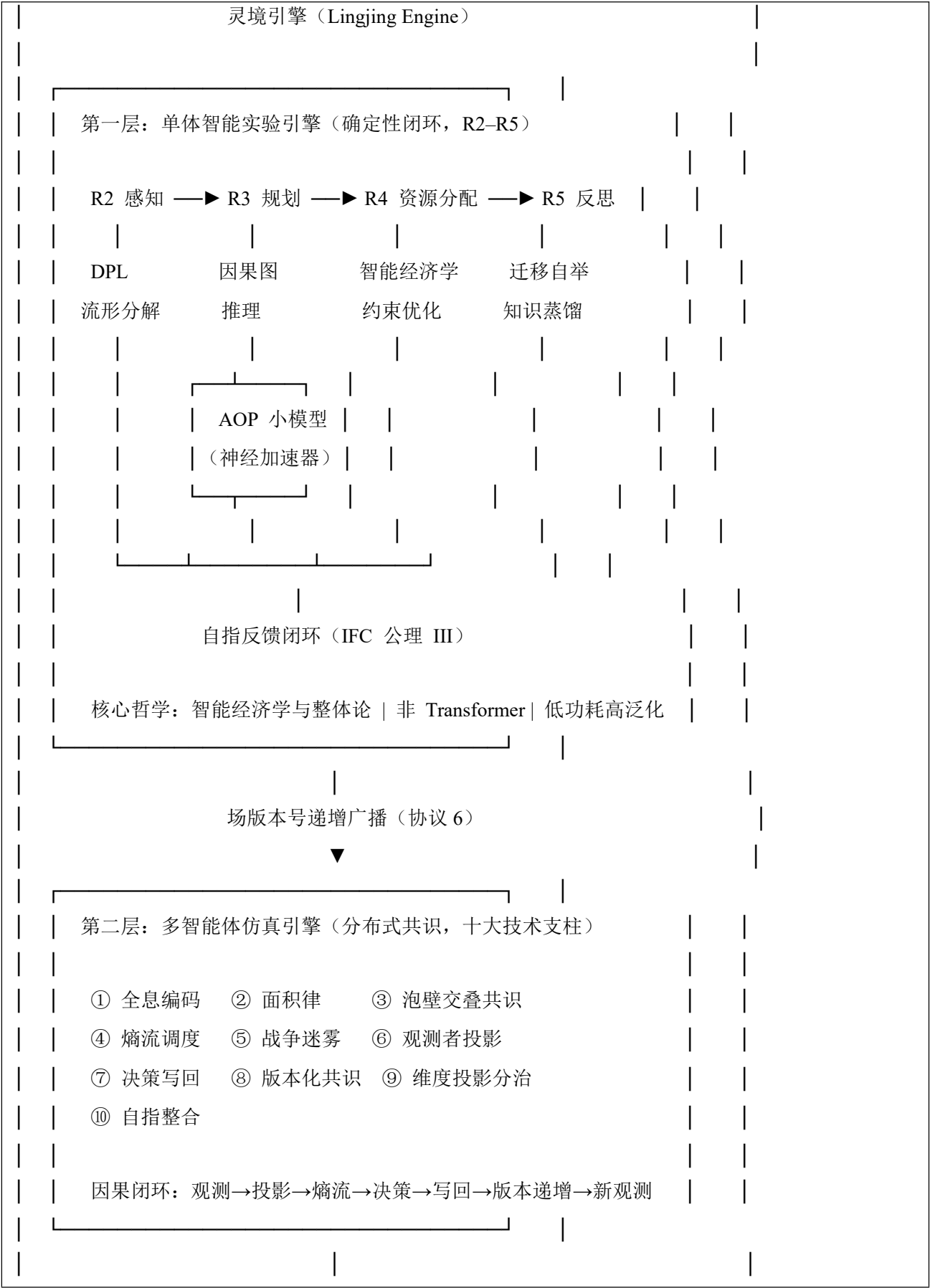
符号约定：标量取绝对值 $|\cdot|$ ；向量/复数取模 $\|\cdot\|$ 。花体 V_t 专指 SICA Lyapunov 势函数，与信息密度场 $\Phi(x)$ 区分。 D_{real} 为 SICA 论文定义的真实递归深度。AOP 为 Action Outcome Predictor，R3 规划层的神经辅助模块（详见 G.2.7）。

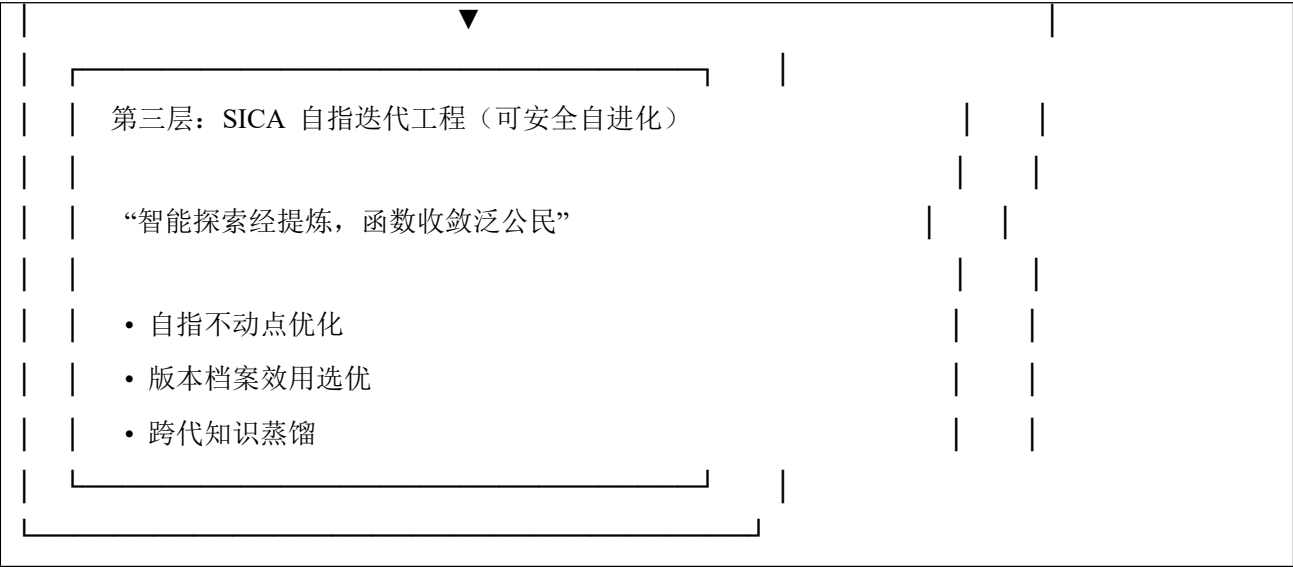
G.1 灵境引擎总体架构

G.1.1 三层一体化架构

灵境引擎构建“**单体智能实验 → 多智能体仿真 → SICA 自指迭代**”三层一体化架构。三层以几何感知为入口、以自指闭环为核心、以场版本号为协同纽带，构成确定性闭环与分布式演化统一的智能体系。

```
```text
|_____|
```





**AOP 小模型定位：**Action Outcome Predictor 位于 R3 规划层，作为因果图搜索的神经加速器。它不替代因果图推理，而是为候选 `Modification` 提供后果预测与初筛，`check` 门禁再做严格校验，保证 fail-closed。

G.1.2 与 IFC 公理体系的对应

灵境层级	IFC 对应	核心机制
单体智能实验引擎（R2–R5）	公理 I + 公理 II	离散信息基元粗粒化为流形特征；动态因果网络驱动规划与资源分配
多智能体仿真引擎	公理 II + 公理 III	泡壁投影算子实现多锚点共识；场版本号递增实现无通讯共识
SICA 自指迭代工程	公理 III 上行反馈（自指相变）	Lyapunov 势函数单调递减（R1–R10 保证）

G.1.3 与 DPL 五接口的对应

DPL 接口	灵境层级	对应协议	职责
detect_dimension	L0–L1	协议 1	维度检测与动态网格建立
calibrate	L1–L2	协议 2	宇宙常数与门禁阈值校准
project	L2–L3	协议 4	D1–D4 流形分解与泡壁投影
check	R2/R3 交界面	协议 5/6	自适应几何门禁
infer	L4	协议 8	物理直觉推理

**注记：**本表按灵境 L0–L5 层级给出 DPL 接口的位置映射，与附录 B B.1.2 的协议映射（协议 4、5、6、8）兼容：`project` ↔ 协议 4，`check` ↔ 协议 5/6，`infer` ↔ 协议 8。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

## G.2 第一层：单体智能实验引擎（R2–R5）

### G.2.1 架构总览

单体智能实验引擎构建 R2 感知 → R3 规划 → R4 资源分配 → R5 反思迁移的确定性闭环，以 DPL 为感知入口，以自指反馈环为核心迭代机制。

#### G.2.2 R2 感知：DPL 流形分解

R2 感知层由 DPL 三接口（`detect\_dimension`、`calibrate`、`project`）构成，负责将任意维度的环境观测张量分解为 D1–D4 流形特征：

- `detect\_dimension`：维度检测与 2.5D 高度场识别；
- `calibrate`：从观测序列校准宇宙常数与门禁阈值；
- `project`：D1–D4 流形分解（详见附录 C）。

`check`（自适应几何门禁）位于 R2 与 R3 交界面，负责动作写回前的门禁校验（详见附录 D）；`infer`（物理直觉推理）位于 L4 自指意识与智能层，负责将流形特征映射为物理直觉（详见附录 E）。

**注记（与附录 B B.1.2 的兼容性）：**本附录按细化后的层级划分 R2 感知层（三接口）与 R2/R3 交界面（`check`）；附录 B B.1.2 的宏观映射将 R2 感知对应协议 4/5，含 `project` 与 `check`。两处互补：宏观上 `check` 属于 R2 感知，微观上 `check` 位于 R2 与 R3 的交界面。

#### G.2.3 R3 规划：因果推理与候选行动

R3 规划基于 D1–D4 流形特征生成候选行动序列：

- 输入：`ProjectedState` 的 D1（流向）、D2（曲率）、D4（因果链）；
- 方法：因果图模型 + 贝叶斯搜索；
- 输出：候选行动序列及其因果预期。

#### G.2.4 R4 资源分配：智能经济学调度

R4 资源分配在功耗、算力、时间约束下执行帕累托最优调度：

- 约束：算力上限、功耗上限、时间预算；
- 方法：约束优化 + 拉格朗日乘子法；
- 输出：本 Tick 的执行计划。

#### G.2.5 R5 反思迁移：知识蒸馏与自举

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

R5 反思迁移将执行经验提炼为可复用模式，通过 SICA 自指迭代提升后续感知与决策效率：

- 输入：本 Tick 的执行轨迹与性能评估；
- 方法：自指不动点定理 + 知识蒸馏；
- 输出：更新后的规则集  $\backslash(H_t)$  与场版本号。

G.2.6 核心哲学：智能经济学与整体论

单体智能实验引擎遵循“智能经济学”核心哲学：每完成一次 R2→R5 闭环，Agent 对环境的几何理解应比上一轮成本更低、精度更高。智能的本质不是算力的堆砌，而是每一轮循环中将不确定性转化为确定性的效率提升。

G.2.7 AOP 小模型：R3 规划层的神经辅助模块

AOP（Action Outcome Predictor）位于 R3 规划层，是因果图搜索的神经加速器。

维度	定位
灵境层级	R3 规划层（辅助模块）
对应协议	协议 3（时空弯曲）→ 协议 5（动作写回）之间的预测环节
输入	ProjectedState 的 D1–D4 低维嵌入 + 候选 Modification
输出	预测下一帧 ProjectedState 的关键统计量，或预测 GateResult 的通过/拦截
训练数据	L0–L3 物理后端生成的 (state, action, next_state) 三元组
与 DPL 对接	输入特征来自 project，输出用于 check 门禁的预筛选

**fail-closed 设计：**AOP 仅做初筛，所有通过初筛的候选必须经 ‘check’ 门禁严格校验。AOP 预测错误时，‘check’ 兜底拦截，系统安全。

**与 SICA 的关系：**AOP 与 SICA 无直接对接关系。AOP 的预测误差用于触发自身重训（G.8），不进入 SICA R2 性能门禁的评估流程。AOP 对整体智能质量的影响是间接的——通过影响 R3 规划层的候选质量，进而影响 R3→R4→R5 闭环的输出。

**蒸馏路径与 AOP 的关系：**蒸馏产物不写入 AOP 模型本身，而是写入灵境场版本号（系统级）。AOP 在下一次 ‘project’ 时读取新版本场状态，间接获得蒸馏后的规则先验。

G.3 第二层：多智能体仿真引擎（十大技术支柱）

G.3.1 架构总览

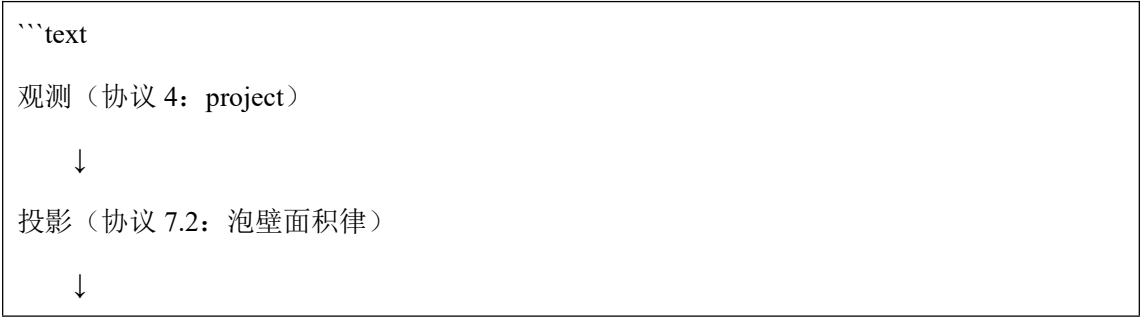
多智能体仿真引擎基于十大技术支柱构建分布式场智能，以场版本号递增为无通讯共识载体，形成“观测→投影→熵流→决策→写回→版本递增→新观测”的分布式因果闭环。

G.3.2 十大技术支柱

编号	支柱名称	IFC 对应	系统功能
①	全息编码	公理 III 泡壁投影	局部观测推断全局状态
②	面积律	公理 II 算力节俭	信息容量由泡壁面积度量
③	泡壁交叠共识	公理 III 泡壁投影	多 Agent 无通讯共识
④	熵流调度	公理 I 信息基座	资源分配的经济学判据
⑤	战争迷雾	公理 II 算力节俭(实现依赖公理 III 泡壁投影)	探索-利用边界的动态模糊调控
⑥	观测者投影	公理 III 泡壁投影	感知从叠加态到确定态的转化
⑦	决策写回	公理 III 上行反馈	Agent 行动反向改变环境信息场
⑧	版本化共识	公理 III 泡壁投影（多锚点扩展）	分布式共识的因果闭环载体
⑨	维度投影分治	公理 III 泡壁投影	复杂任务的维度投影分解执行
⑩	自指整合	公理 III 自指相变	SICA 迭代的终极收敛机制

**注记（⑤ 战争迷雾）：**战争迷雾的核心机制是泡壁投影算子  $\backslash(P\_Sigma\backslash)$  对清晰区与迷雾区的划分（附录 C C.9）。归入公理 II（算力节俭）因其服务于算力节俭目标；实现手段依赖公理 III（泡壁投影）。

G.3.3 因果闭环



熵流（协议 7.1：场自然演化）

↓

决策（R3 规划 + R4 资源分配）

↓

写回（协议 5：check + 源项注入）

↓

版本递增（协议 6：无通讯共识）

↓

新观测（下一 Tick 读取新版本号  $V_{N+1}$ ）

### G.3.4 泡壁面积律与算力节俭

- 传统体积填充引擎：复杂度  $\propto O(L^3)$ ；
- 灵境面积律引擎：复杂度  $\propto O(L^2)$ 。

高精度渲染仅对清晰区执行 D1–D4 全流形分解，迷雾区仅保留低维统计量。

## G.4 第三层：SICA 自指迭代工程

### G.4.1 架构总览

SICA（Self-Improving Convergence Algorithm，自指迭代收敛算法）位于灵境 R5 元层，为 L4 自指意识层与 R2–R5 闭环提供安全收敛保障。

### G.4.2 核心势函数

$$V_t = -P_t + \lambda C_t + \mu N_{\text{viol}}(t)$$

其中：

- $V_t$  为 SICA 势函数，区别于信息密度场  $\Phi(x)$ ；
- $P_t$  为留存集性能；
- $C_t$  为复杂度；
- $N_{\text{viol}}(t)$  为边界违规次数。

SICA 通过 R1–R10 规则保证每次修改使  $V_{t+1} < V_t$ ，且  $V$  有下界，从而收敛。

### G.4.3 三大机制

机制	功能	SICA 规则
----	----	---------

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

机制	功能	SICA 规则
自指不动点优化	保证自指迭代收敛到唯一不动点	R4（递归深度限制）+ R7（自动回滚）
版本档案效用选优	从历史版本中选取最优配置	R2（性能门禁）+ R3（复杂度惩罚）+ R7（版本化）
跨代知识蒸馏	将历代经验提炼为可复用规则	R6（衰减淘汰）+ R9（证据阈值）+ R10（跨族泛化）

**注记（R7 的双重服务）：**R7 由“版本化”与“自动回滚”两部分组成。版本化服务于“版本档案效用选优”；自动回滚服务于“自指不动点优化”。两部分共享同一版本化基础设施。

**注记（蒸馏对象澄清）：**自指锚点  $C_0$  为泡壁映射族  $\mathcal{M}_{self}$  的唯一公共不动点（主文 4.5.1），它是系统的拓扑不变量，不参与蒸馏，仅作为规则集  $H_t$  的来源标识。

蒸馏的对象不是某个具体的“自指锚点”，而是锚点在自指闭环中累积的**规则集  $H_t$** 。蒸馏产物是可跨锚点复用的**几何不变式（‘GeometricInvariant’）**。蒸馏路径为：

``text
锚点 $\rightarrow$ 规则集 $H_t \rightarrow$ SICA R9/R10 约束 $\rightarrow$ 场版本号 $\rightarrow$ 所有 Agent（含 AOP）

单个 Agent 的私有记忆（协议 8）不参与跨代蒸馏。

G.4.4 与 DPL 的对接

DPL 的三类数据结构（附录 F F.2）进入 SICA R5 元层：

- ‘GeometricInvariant’（几何不变式）来自 ‘project’；
- ‘GateCounterexample’（门禁反例）来自 ‘check’；
- ‘IntuitionHypothesis’（直觉假设）来自 ‘infer’。

三类结构经 SICA R1–R10 规则约束后写回 IFC 公设库、DPL 门禁先验库、灵境场版本号。

G.5 三层架构的耦合

G.5.1 从单体到多体

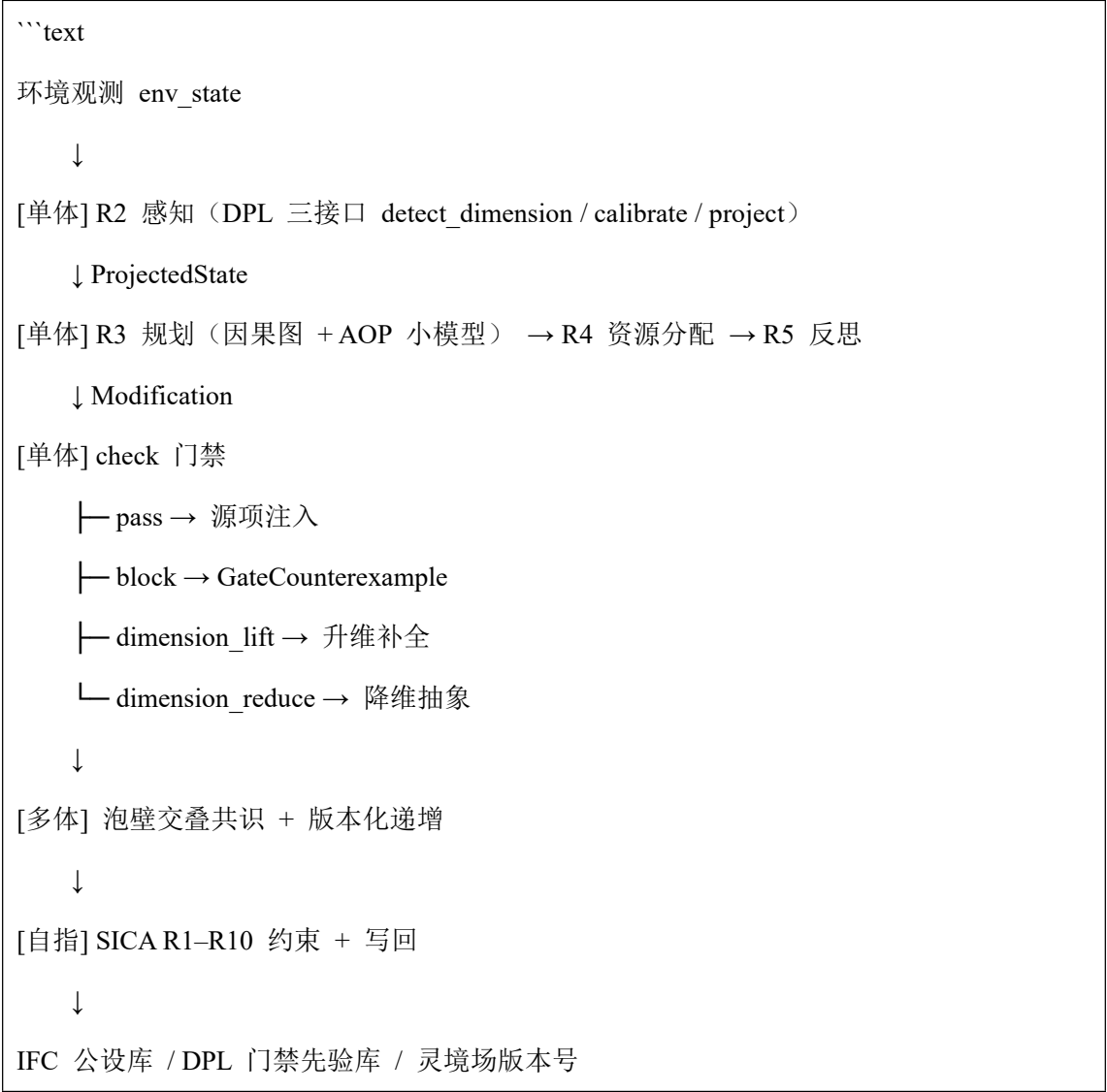
- 单体 ‘project’ 输出的 ‘ProjectedState’  $\rightarrow$  协议 6 版本化共识；
- 单体 ‘check’ 输出的 ‘GateResult’  $\rightarrow$  多体共识帧合成输入；
- 单体 ‘infer’ 输出的 ‘IntuitionHypothesis’  $\rightarrow$  L4 意识相变候选证据（协议 8）。

G.5.2 从多体到自指迭代

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

- 多体的 `GeometricInvariant` → IFC 公设库更新；
- 多体的 `GateCounterexample` → 门禁先验库更新；
- 多体的 `IntuitionHypothesis` → SICA 候选规则池。

G.5.3 全链路数据流



G.6 与 IFC 公理体系的完整对应

G.6.1 三层架构 × 三公理矩阵

IFC 公理	单体智能 (R2–R5)	多智能体仿真	SICA 自指迭代
公理 I：离散信息基元与粗粒化守恒	D1 流线维分解	熵流调度	R1 锚点不可伪造（确保信息源真实性）

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

公理 II: 动态因果网络与算力节俭	R3 因果规划	泡壁面积律 + 无通讯共识	R2 性能门禁（因果收敛）+ R3 复杂度惩罚（算力节俭）
公理 III: 对称映射投影与上行反馈	R2 泡壁投影 + R5 反思	观测者投影 + 决策写回	R7 版本化与回滚 + 自指相变

SICA 与 IFC 公理的对接通过 R1–R10 规则实现：R1 与公理 I 的信息守恒要求对接；R2 + R3 与公理 II 的“动态因果网络”（R2 保证因果收敛）与“算力节俭”（R3 保证复杂度收益比）两部分内涵对接；R7 与公理 III 的上行反馈/自指相变对接。SICA 不在公理层，而在算法层。

### G.6.2 核心方程链

公理 I（信息守恒）：

$$Q_{info}=\sum_i b_i=\int_{\mathcal{M}_4}|\Phi(x)|^2\sqrt{-g}\,d^4x=Const$$

公理 II（动态因果网络）：

$$A_{ij}(\tau)\in\mathbb{C},\qquad\sum_j|A_{ij}(\tau)|^2=1$$

公理 III（几何生成与上行反馈）：

$$g_{\mu\nu}=\eta_{\mu\nu}+\frac{1}{\mathcal{J}_{Pl}}\partial_\mu\Phi\partial_\nu\Phi+\kappa(\Phi^2-\Phi_0^2)\eta_{\mu\nu}$$

$$\Psi_{con}(x)=\Phi(x)\cdot\Theta(C-C_c)$$

$$\delta\Phi(x)=\epsilon\cdot\nabla_\mu\Psi_{con}(x),\qquad\epsilon\sim10^{-12}$$

SICA 势函数：

$$V_t=-P_t+\lambda C_t+\mu N_{viol}(t)$$

## G.7 世界观统一

### G.7.1 核心世界观

- 信息一元论：**宇宙所有内容——时空、物质、场、意识——均由离散信息基元通过粗粒化映射涌现；
- 几何即智能：**智能的本质是对信息流形的结构解析，而非对数据分布的概率拟合；
- 自指收敛：**任何可自进化的智能体必须通过 Lyapunov 势函数的单调递减保证收敛，避免自指发散。

### G.7.2 与主流 AI 范式的对比

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
 开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

对比维度	Transformer 范式	灵境引擎范式
核心操作	注意力加权求和	流形分解 + 几何门禁
信息表示	高维向量 + Softmax	维度投影 + 拓扑不变量
计算瓶颈	注意力机制 $O(n^2)$	几何门禁 $O(d_{eff})$
功耗特征	训练阶段巨量耗电，推理阶段线性功耗	闭环自治迭代，边际功耗随迭代递减
泛化来源	大规模数据统计归纳	几何不变式 + 物理直觉推理
自指能力	无原生自指机制	自指不动点内生于架构
多 Agent 协同	需额外通信协议	场版本号递增 = 天然因果共识
AGI 判据	行为表现标准	架构刚性判据（真实递归深度 $D_{real} \geq 4$ ，详见《衔尾蛇 SICA》）

### G.7.3 与 NVIDIA AVO / OpenAI Provider Adapter 的对比

近期 ARC-AGI-3 的满分结果（NVIDIA AVO + Claude Opus 5 = 100%；OpenAI Astra Provider Adapter = 99.9%）表明：模型本身不是满分的关键，harness（执行环境）才是。

组件	对应	作用
Claude Opus 5 / GPT-5 类基础模型	R3 因果图搜索 + R5 反思迁移（含 AOP 小模型辅助）	高层规划、多步推理、动作后果预测
NVIDIA AVO / OpenAI Provider Adapter	灵境 L0-L3 物理后端（DPL + 协议栈）	环境反馈、动作接口、门禁校验
100% 满分	完整灵境三层架构	模型层 + harness 层 + SICA 迭代

**对应关系说明：**灵境范式中基础模型并非单一组件，而是 R2-R5 闭环的整体；AOP 小模型仅承担动作后果预测这一单一任务，高层推理由 R3 因果图搜索与 R5 反思迁移共同承担。因此，Claude Opus 5 类基础模型的角色对应灵境 R3 + R5 的整体，而非单独对应 AOP。

**灵境引擎的差异化优势：**AVO 是外部 harness，模型与 harness 之间存在接口不透明；灵境引擎的 L0-L3 是自研物理后端，AOP 小模型与 DPL 五接口共享同一术语体系（附录 A-G），接口刚性更强，fail-closed 更彻底。

### G.7.4 范式宣言

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）  
开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

灵境范式回归“几何即智能”的本质，摆脱对大规模数据与算力的依赖，以信息一元论为底层、以流形几何为方法、以自指迭代为进化路径，构建低功耗、强泛化、高确定性的新一代智能体系。

G.7.5 与灵境协议栈的对齐

协议栈层级	世界观对应
L0 信息基元层	信息一元论（公理 I）
L1 动态因果网络层	因果网络（公理 II）
L2 场与几何生成层	几何生成（公理 III）
L3 泡壁投影与渲染层	观测者投影
L4 自指意识与智能层	自指相变
L5 应用与交互层	源项注入与人类接口
R2 感知	DPL 流形分解
R3 规划	因果推理 + AOP 小模型
R4 资源分配	智能经济学
R5 元层	SICA 自指迭代

G.8 异常条件与边界处理

触发条件	异常类型	说明
三层架构中任一层缺失	RuntimeError	灵境引擎需三层完整
场版本号在层间不一致	RuntimeError	版本同步失败
连续 5 次修改触发 R2 拒绝	RuntimeError	自指发散风险，触发人工审查
连续 5 次修改触发 R8 回滚	RuntimeError	触发人工审查
多体共识帧合成时锚点重叠度 $<\eta_{crit}$	RuntimeWarning	联盟分裂
DPL 五接口任一失败	ValueError	感知链断裂
AOP 预测与 check 门禁结果严重背离 (>50%)	RuntimeWarning	AOP 模型需重训

**注记（AOP 背离阈值）：**50% 为启发式设计参数，可配置；具体取值可通过环境观测校准。当 AOP 与 `check` 门禁结果的背离率持续超过该阈值时，触发 AOP 重训流程。

G.9 实现注记

G.9.1 与各附录的交叉引用

交叉项	目标位置
符号与量纲	附录 A
DPL 五接口	附录 B B.1–B.9
D1–D4 流形分解	附录 C C.1–C.10
自适应几何门禁	附录 D D.1–D.12
宇宙常数校准与物理直觉	附录 E E.1–E.8
DPL→SICA 数据结构与协议栈映射	附录 F F.1–F.7
AOP 接口规范	附录 H（待补）
灵境协议栈	《智能仿真的全息原理引擎》v13.2
SICA R1–R10	《衔尾蛇 SICA》

**注记：**附录 H（AOP 小模型接口规范）为后续补充文档，涵盖 AOP 输入/输出接口、训练数据格式、与 `check` 门禁的 fail-closed 对接、与 SICA R2 的评估对接等。

G.9.2 前置修订汇总

修订项	目标位置	说明
D1FlowLine 增加 gradient_vector	附录 C C.3.1	用于信息守恒与跨维隔离校验
D2Surface 增加 vorticity_is_vector	附录 C C.4.1	用于涡旋校验与 GeometricInvariant 提取
UniversalConstants 增加 7 个字段	附录 B B.3.3	clear_region_percentile、I_0、l_0、chi_ref、gravity_threshold_ratio、periodicity_threshold、conservation_stats
GeometricInvariant 增加 features	附录 B B.7.1	模式特征值（可选）

## 附录 H：AOP 小模型接口规范

**说明：**本附录定义 AOP（Action Outcome Predictor）小模型在灵境 R3 规划层的完整接口规范，涵盖模型定位、输入输出接口、训练数据格式、与 `check` 门禁的 fail-closed 对接、与 SICA 的关系、蒸馏路径与重训触发。所有符号、量纲回溯附录 A；接口定义回溯附录 B；D1–D4 特征回溯附录 C；门禁输出回溯附录 D；校准与直觉输出回溯附录 E；SICA 映射回溯附录 F；三层架构定位回溯附录 G。本附录不引入新物理假设，仅为 AOP 在灵境引擎中的工程接口规范。

符号约定：标量取绝对值  $|\cdot|$ ；向量/复数取模  $\|\cdot\|$ 。 $D_{real}$  为 SICA 论文定义的真实递归深度。AOP 输出使用统一前缀 `OutcomePrediction`。模型版本号 `model\_version` 与场版本号  $V_N$  为两个独立版本号系统，不可混用（详见 H.7.2）。

### H.1 AOP 在灵境引擎中的定位

#### H.1.1 层级定位

AOP（Action Outcome Predictor）是位于灵境 R3 规划层的神经辅助模块，作为因果图搜索的神经加速器。它不替代因果图推理，而是为候选 `Modification` 提供后果预测与初筛。

维度	定位
灵境层级	R3 规划层（辅助模块）
对应协议	协议 3（时空弯曲）→ 协议 5（动作写回）之间的预测环节
输入	ProjectedState 的 D1–D4 低维嵌入 + 候选 Modification
输出	OutcomePrediction（预测下一帧 ProjectedState 的关键统计量或 GateResult）
训练数据	L0–L3 物理后端生成的 (state, action, next_state) 三元组
与 DPL 对接	输入特征来自 project，输出用于 check 门禁的预筛选
与 SICA 关系	<b>无直接对接关系</b> ；预测误差仅触发自身重训（H.6）
版本号系统	<b>模型版本号</b> model_version（独立于场版本号 $V_N$ ）
监控主体	<b>R3 规划层</b> （非 AOP 自身）

#### H.1.2 与 R3 规划层其他模块的关系

R3 规划层包含两个协同模块：

模块	职责	输出
因果图推理	基于 D1-D4 流形特征生成候选行动序列及其因果预期	候选行动序列
AOP 小模型	对候选行动序列提供后果预测与初筛	排序后的候选行动序列

二者协同流程：



**fail-closed 设计：**AOP 仅做初筛，所有通过初筛的候选必须经 `check` 门禁严格校验。

AOP 预测错误时，`check` 兜底拦截，系统安全。

**模型未加载时的回退：**若 AOP 模型未加载，R3 规划层回退到纯因果图搜索模式（H.4.1）。该回退为设计内的正常路径（graceful degradation），记录 **Info** 级日志，不抛出异常，不影响 fail-closed 保证。

H.1.3 与基础大模型的角色区分

灵境范式中，基础大模型（如 Claude Opus 5、GPT-5 类）的角色对应 R3 因果图搜索 + R5 反思迁移的整体，涵盖高层规划、多步推理、动作后果预测。AOP 小模型仅承担动作后果预测这一单一任务，是基础大模型能力的局部神经近似，而非替代。

H.2 AOP 输入接口

H.2.1 签名

```
```python
```

```
def aop_predict(  
    projected: ProjectedState,  
    modification: Modification,  
    constants: UniversalConstants  
) -> OutcomePrediction
```

H.2.2 输入数据结构

- ‘ProjectedState’（附录 B B.4.3）：完整的 D1–D4 流形特征。
- ‘Modification’（附录 B B.5.2）：候选行动提案。
- ‘UniversalConstants’（附录 B B.3.3）：宇宙常数与门禁阈值。

H.2.3 输入编码

AOP 的输入编码分为两部分：

(1) 流形特征编码：

特征源	编码方式	维度
‘D1.gradient’（梯度模）	分块均值池化	d_1
‘D1.divergence’（散度）	分块均值池化	d_1
‘D1.direction’（单位向量）	分块归一化平均（先平均再归一化）	$d_1 \times d_{spatial}$
‘D2.curvature’（高斯曲率）	分块均值池化	d_2
‘D2.vorticity’（涡旋模）	分块均值池化	d_2
‘D3.density’（体密度）	分块均值池化	d_3
‘D3.euler_char’（欧拉示性数）	标量	1
‘D4.evolution’（时间演化）	分块均值池化	d_4
‘D4.causal_link’（因果关联）	分块均值池化	d_4

其中 d_1, d_2, d_3, d_4 为各层编码维度，可根据 AOP 模型规模调整；H.10.1 给出建议值。

注记（‘D1.direction’ 编码）：‘D1.direction’ 为单位向量场，直接算术平均会因方向抵消而损失语义。本附录规定其分块编码为**归一化平均**：先对分块内所有单位向量求算术平均，再对平均向量归一化。若分块内平均向量为零向量（完全抵消），该分块编码为零向量并附加掩膜标记。

(2) 行动编码：

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

特征源	编码方式	维度
modification.type（类型）	one-hot	n_{types}
modification.target（目标）	嵌入向量	d_{target}
modification.params（数值参数）	展平	n_{params}
modification.info_before（前信息量）	标量	1
modification.info_after（后信息量）	标量	1

H.2.4 异常条件

条件	异常类型	说明
projected.dim_info 缺失	ValueError	未执行维度检测
modification.type 未注册	ValueError	未知修改类型
输入形状与训练配置不符	ValueError	模型输入维度不匹配

H.3 AOP 输出接口

H.3.1 输出数据结构 `OutcomePrediction`

字段	类型	说明
predicted_gate_action	str	预测的 `GateResult.action` （`pass`/`block`/`dimension_lift`/`dimension_reduce`）
predicted_gate_score	float	预测的 `GateResult.score`，范围 `[0,1]`
predicted_next_stats	dict	预测的下一帧 `ProjectedState` 关键统计量
confidence	float	AOP 预测置信度 `[0,1]`
model_version	str	当前 AOP 模型实例的版本号（独立于场版本号 `V_N`）

`predicted\_next\_stats` 字段：

字段	类型	量纲	说明
d1_gradient_mean	float	$\text{bit}^{1/2}[\text{L}]^{-3}$	预测的 D1 梯度模均值
d2_curvature_mean	float	1（无量纲）	预测的 D2 曲率均值
d2_vorticity_mean	float	$[\text{L}]^{-1}$	预测的 D2 涡旋模均值（若 vorticity_is_vector=True 先取

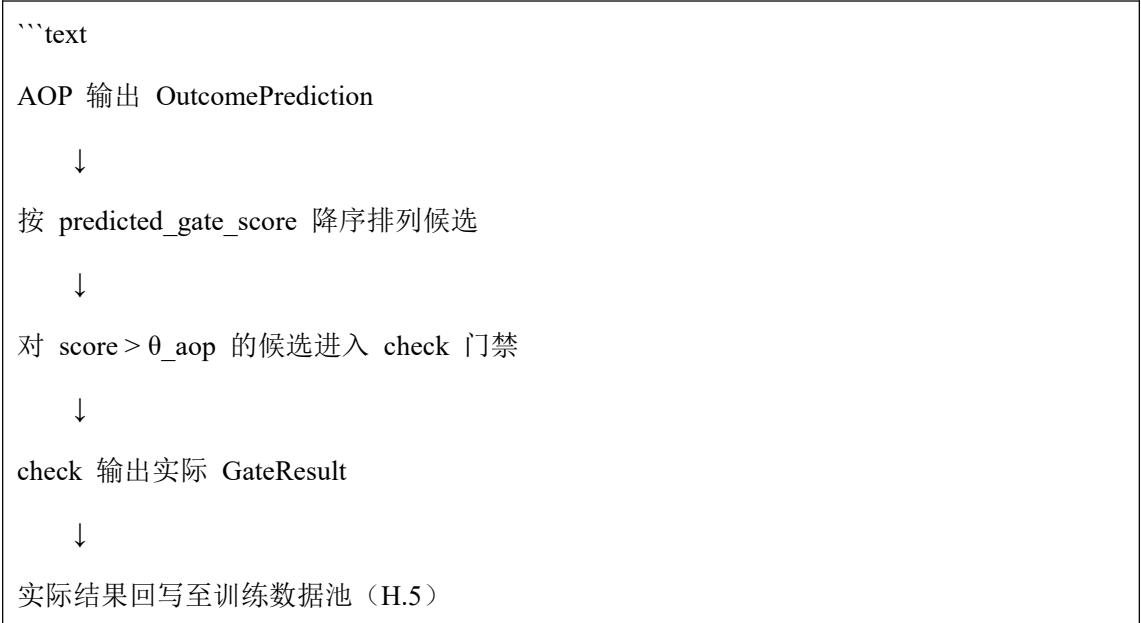
作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

			模)
d3_density_mean	float	$\text{bit}^{1/2}[\text{L}]^{-2}$	预测的 D3 体密度均值
d4_evolution_mean	float	$\text{bit}^{1/2}[\text{L}]^{-3}$	预测的 D4 时间演化均值

注记（取模运算）：`d2\_vorticity\_mean` 对应 `D2.vorticity` 的模。若 `D2.vorticity\_is\_vector=True`（3D+），先取 $\|\nabla \times \mathbf{v}_I\|$ 再求均值；若 `vorticity\_is\_vector=False`（2D、2.5D+时间），直接取 $|\nabla \times \mathbf{v}_I|$ 。

注记（`model\_version` 格式）：`model\_version` 推荐格式为 `v{major}.{minor}.{patch}`（如 `v1.2.3`），或使用 Git commit hash 的前 8 位。具体格式由 R3 规划层的版本管理策略决定。同一实例内所有预测的 `model\_version` 相同；版本号仅在 AOP 重训（H.6.2）后更新。

H.3.2 输出使用流程



其中 θ_{aop} 为 AOP 初筛阈值，典型取值 0.3，可配置。

H.3.3 量纲约定

`OutcomePrediction.predicted\_next\_stats` 中各字段量纲与 `ProjectedState` 对应字段一致（回溯附录 A A.3.3）。AOP 输出无量纲 `predicted\_gate\_score` 与 `confidence`，用于排序与筛选。

H.4 AOP 与 `check` 门禁的 fail-closed 对接

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

H.4.1 对接流程

```
```python
依赖:

log = logging.getLogger(__name__)

from dpl_core.planning import causal_graph_search

from dpl_core.aop import record_aop_mismatch

def r3_pipeline(projected, candidate_modifications, constants):

 """R3 规划层流水线: 因果图生成 → AOP 初筛 → check 严格校验"""

 # 步骤 1: 因果图推理生成候选

 candidates = causal_graph_search(projected, constants)

 # 步骤 2: 若 AOP 模型未加载, 回退到纯因果图搜索

 # 注: 该回退为设计内的正常路径 (graceful degradation), 记录 Info 级日志

 if not aop_model_loaded():

 log.info("AOP model not loaded; fallback to pure causal graph search")

 verified = []

 for mod in candidates:

 result = check(mod, projected, constants)

 if result.passed:

 verified.append((mod, result))

 return verified

 # 步骤 3: AOP 初筛

 predictions = [

 aop_predict(projected, mod, constants)

 for mod in candidates

]

 sorted_pairs = sorted(
```

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

```

zip(candidates, predictions),

key=lambda p: p[1].predicted_gate_score,

reverse=True,

)

filtered = [(mod, pred) for mod, pred in sorted_pairs
 if pred.predicted_gate_score > θ_{aop}]

步骤 4: check 门禁严格校验

verified = []

for mod, pred in filtered:

 result = check(mod, projected, constants)

 if result.passed:

 verified.append((mod, result))

 else:

 record_aop_mismatch(mod, pred, result)

return verified

```

依赖说明:

符号	归属	说明
`log`	模块级 logger	`logging.getLogger(__name__)`
`causal_graph_search`	`dpl_core.planning`	R3 规划层的因果图搜索函数
`record_aop_mismatch`	`dpl_core.aop`	AOP 预测与 `check` 实际结果不一致的记录函数

`record\_aop\_mismatch` 参考实现:

```

``python

def record_aop_mismatch(mod, pred, result):

 """记录 AOP 预测与 check 门禁实际结果的不一致，用于训练样本回写"""

 mismatch_record = {

 "modification": mod.serialize(),

```

```

 "aop_predicted_action": pred.predicted_gate_action,

 "aop_predicted_score": pred.predicted_gate_score,

 "actual_gate_action": result.action,

 "actual_gate_score": result.score,

 "timestamp": time.time_ns(),

}

append_to_mismatch_log(mismatch_record)

```

#### H.4.2 Fail-Closed 保证

- AOP 只做初筛，不做最终决策；
- 所有通过初筛的候选必须经 `check` 门禁严格校验；
- AOP 预测为 `pass` 但 `check` 拦截的候选，记录为 `GateCounterexample`（附录 D D.10.2）进入 SICA 负样本池；
- AOP 预测为 `block` 但 `check` 通过的候选，同样记录为 AOP 训练正样本（实际通过）。

#### H.4.3 门禁与 AOP 的分工

模块	职责	输入规模	计算量
AOP 小模型	快速初筛，排序	全候选集	$O(N)$ 神经推理
check 门禁	严格校验，门禁	初筛后候选	$O(M)$ 解析计算 ( $M \ll N$ )

AOP 的价值在于将 `check` 门禁的调用次数从  $N$  降至  $M$ ，在大规模候选场景下显著降低计算量。

### H.5 AOP 训练数据

#### H.5.1 数据来源

AOP 训练数据来源于 L0-L3 物理后端（灵境协议栈）生成的 (state, action, next\_state) 三元组：

```

```text
L0-L3 物理后端
    ↓
每一 Tick 记录：

```

(ProjectedState_N, Modification, ProjectedState_{N+1})

↓

按比例划分为：

- 训练集（70%）
- 验证集（15%）
- 测试集（15%）

H.5.2 数据结构

```
```python
@dataclass
class AOPTrainingSample:
 """AOP 训练样本"""
 projected_before: ProjectedState
 modification: Modification
 projected_after: ProjectedState
 actual_gate_result: GateResult
 environment_family: str # 用于 R10 跨族泛化验证
 timestamp: int
```

## H.5.3 数据生成协议

步骤	操作
1	从当前 ProjectedState_N 采样候选 Modification
2	执行 Modification 并推进场方程一 Tick
3	记录 ProjectedState_{N+1} 与 check 实际 GateResult
4	写入训练样本池（JSONL 格式）
5	按环境族标记 environment_family，支持 R10 跨族泛化验证

## H.5.4 数据存储格式

```
```jsonl
{
    "projected_before": {...},
```

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

```
"modification": {...},
"projected_after": {...},
"actual_gate_result": {...},
"environment_family": "family_A",
"timestamp": 1234567890123456789
}
```

单条样本大小估算：`ProjectedState` 约 10–100 KB（取决于分辨率），`Modification` 约 1 KB。10 万条样本约 1–10 GB。

H.5.5 数据质量控制

规则	说明
唯一性校验	相同(projected_before, modification)去重
数值合法性	过滤 NaN/Inf 样本
环境族覆盖	每个环境族至少 1000 条样本
类别平衡	pass/block 类别比例控制在 1:1 至 1:3（启发式比例，可配置；具体取值取决于环境的通过率分布）

注记（类别平衡实现）：类别平衡可通过以下方式实现（任选其一）：过采样少数类、欠采样多数类、或使用类别加权损失函数。具体方式由 AOP 训练流程决定。

H.6 AOP 重训触发

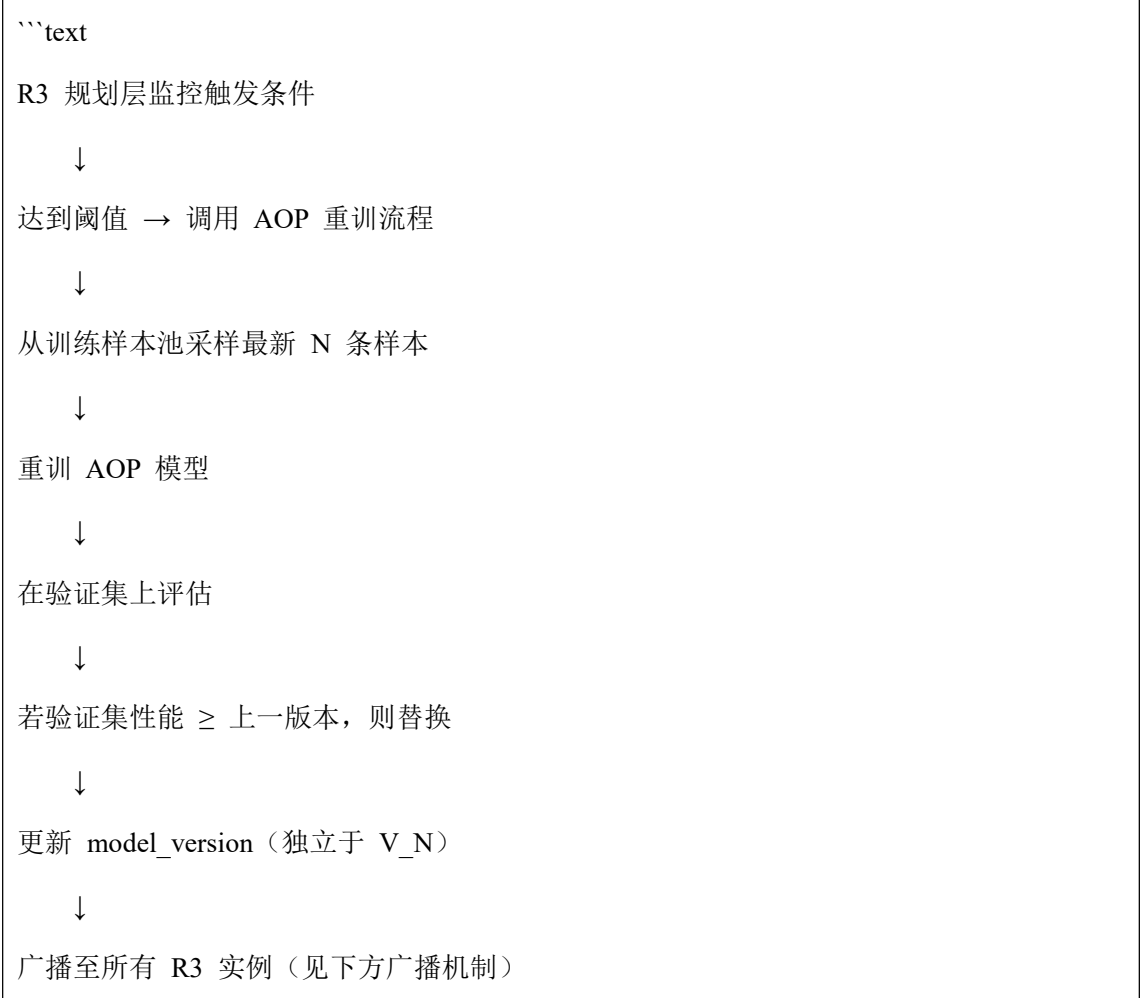
H.6.1 触发条件

关键约定：AOP 自身不监控任何触发条件；所有触发条件由 **R3 规划层** 监控，达到阈值后调用 AOP 重训流程（H.6.2）。

触发条件	阈值	监控主体	说明
AOP 与`check`门禁结果背离率	>50%，持续 1000 次预测	R3 规划层	触发重训
环境族新增	-	R3 规划层	出现未在训练集中出现的环境族
场版本号大幅跃迁	最近 100 Tick 内 $\Delta V_N > 100$	R3 规划层（非 AOP）	规则集发生重大更新
定期重训	每 10^5 次预测	R3 规划层	防止模型漂移（启发式阈值，可配置）

注记（时间窗口）：场版本号跃迁的统计窗口为最近 100 个 Tick。窗口内若 $\Delta V_N > 100$ ，视为规则集发生重大更新，触发 AOP 重训。

H.6.2 重训流程



R3 实例间的 AOP 模型广播机制：

- 若 R3 实例运行于**同一主机**：通过共享存储加载模型权重；
- 若 R3 实例分布于**多主机**：通过 gRPC/HTTP 分发模型权重；
- 模型分发**独立于场版本号协议 6 广播**；模型版本号 `model\_version` 由 R3 规划层本地管理。

注记（模型分发成本）：3B–7B 参数模型（FP16）约 6–14 GB。模型分发为**低频事件**——重训频率为每 10^5 次预测一次（H.6.1），典型场景下跨越数小时至数天。模型权重分发可选用分块传输、共享存储挂载或按需拉取策略，具体由部署环境决定。

H.6.3 与 SICA 的关系

AOP 与 SICA 无直接对接关系。AOP 的重训属于 R3 规划层的内部维护，不进入

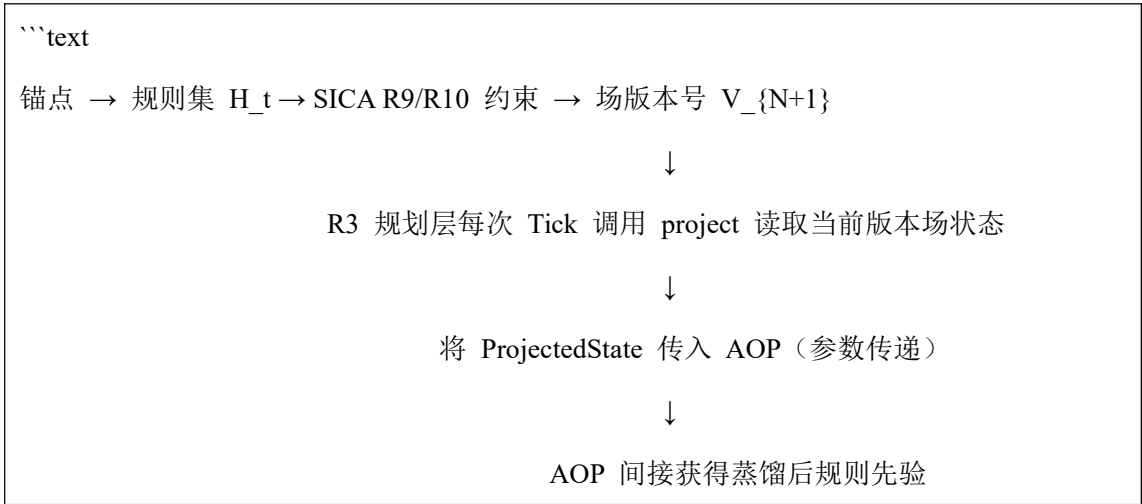
SICA R2 性能门禁的评估流程。AOP 重训的触发条件 (H.6.1) 由 R3 规划层自行监控，与 SICA R1-R10 规则解耦。

AOP 对整体智能质量的影响是间接的——通过影响 R3 规划层的候选质量，进而影响 R3→R4→R5 闭环的输出。该间接影响可由 R5 反思迁移通过协议 8 记录，作为环境族性能分析的参考，但不进入 SICA 候选规则池。

H.7 AOP 与蒸馏路径

H.7.1 蒸馏对 AOP 的影响

蒸馏产物不写入 AOP 模型本身，而是写入灵境场版本号 V_N （系统级）。R3 规划层在每次 Tick 的常规流程中调用 `project` 读取当前版本场状态，并将结果通过 `projected` 参数传入 AOP。当场版本号递增（协议 6 广播）后，R3 规划层在下一次 Tick 自动读取新版本，使 AOP 间接获得蒸馏后的规则先验。



注记：AOP 是 `aop\_predict(projected, modification, constants)` 函数 (H.2.1)，不直接调用 `project`，不直接访问场版本号。场状态读取由 R3 规划层承担；AOP 仅接收 `projected` 参数。

H.7.2 两个独立的版本号系统

关键区分：AOP 涉及两个独立的版本号系统，二者服务于不同路径，不可混淆。

版本号	符号	管理主体	服务路径	广播方式
场版本号	V_N	灵境协议栈（协议 6）	SICA 规则级蒸馏	全局广播（所有 Agent）
模型版本号	`model_version`	R3 规划层本地	AOP 模型级	R3 实例间广播

版本号	符号	管理主体	服务路径	广播方式
			传承	(H.6.2)

两条独立路径：

- **规则级蒸馏路径：**锚点 $\rightarrow$ 规则集 $H_t \rightarrow$ SICA R9/R10 约束 $\rightarrow$ 场版本号 $V_{N+1} \rightarrow$ 所有 Agent；
- **模型级传承路径：**AOP 重训 (H.6.2) $\rightarrow$ 更新 `model\_version` $\rightarrow$ R3 实例间广播。

AOP 不参与规则级蒸馏。蒸馏的对象是锚点累积的规则集 H_t ，产物是可跨锚点复用的几何不变式（`GeometricInvariant`）。AOP 是 R3 规划层的神经近似模块，其权重不进入蒸馏流程；AOP 的跨代传承通过 `model\_version` 独立实现。

H.8 AOP 与 NVIDIA AVO 类 harness 的对比

H.8.1 架构对比

对比维度	NVIDIA AVO 类 harness	灵境 AOP + L0-L3
模型与 harness 接口	不透明（外部适配器）	透明（共享附录 A-H 术语体系）
动作后果预测	由基础大模型承担	由 AOP 小模型承担
门禁校验	由 harness 内部实现，不透明	由 check 门禁严格实现（附录 D）
Fail-closed 保证	依赖 harness 实现质量	由 DPL 五接口规范刚性保证
训练数据来源	未公开	L0-L3 物理后端生成 (state, action, next_state)
跨代传承	未公开	模型版本号 model_version （模型级）与 SICA 场版本号 V_N （规则级）两条独立路径

H.8.2 灵境引擎的差异化优势

- **接口刚性：**AOP 与 DPL 五接口共享同一术语体系，避免接口不透明；
- **Fail-closed 保证：**AOP 仅初筛，`check` 门禁兜底拦截；
- **双路径传承独立：**模型级传承（`model\_version`）与规则级蒸馏（ $\backslash(V_N)$ ）独立并行，互不干扰；
- **训练数据可溯源：**所有 AOP 训练样本来自 L0-L3 物理后端，可复现、可审计。

H.9 异常条件与边界处理

条件	类型	说明
AOP 模型未加载	Info 日志（设计内正常路径）	R3 规划层回退到纯因果图搜索模式；性能降低但不影响 fail-closed 保证（H.1.2、H.4.1）
AOP 输入形状与模型不符	ValueError	输入维度不匹配
AOP 输出字段缺失	ValueError	模型输出格式错误
AOP 预测与 check 背离率 >50%（持续 1000 次预测）	RuntimeWarning	触发重训（H.6）
AOP 训练样本池为空	RuntimeError	无法重训
环境族未在训练集中出现	RuntimeWarning	触发增量训练（H.6.1）

注记（异常类型区分）：

- AOP 模型未加载为设计内正常路径，记录 Info 级日志；它是灵境引擎的 graceful degradation 策略，不是异常。
- AOP 输入/输出异常为编程或配置错误，抛出 `ValueError`。
- 背离率超阈值、新增环境族为运行时的可恢复状态，抛出 `RuntimeWarning` 触发重训。
- 训练样本池为空为不可恢复的运行时错误，抛出 `RuntimeError`。

H.10 实现注记

H.10.1 AOP 模型规格建议

参数	建议值	说明
模型规模	3B-7B	与群聊讨论一致
输入编码维度	$d_1=d_2=d_4=32$, $d_3=16$	见 H.2.3，可根据模型规模调整
隐藏层维度	512-1024	根据模型规模调整
输出层	多头(action 分类 + score 回归 + stats 回归)	见 H.3.1

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering（团队持续开放迭代）

参数	建议值	说明
训练优化器	AdamW	标准选择
学习率	10^{-4}	根据训练稳定性调整
批次大小	32–128	根据显存调整

H.10.2 AOP 初筛阈值

‘ θ_{aop} ’ (H.3.2) 为启发式初筛阈值，典型取值 0.3，可配置。用于平衡 AOP 初筛的召回率与 ‘check’ 门禁的调用次数：

- ‘ θ_{aop} ’ 过高 → 初筛过于严格，可能漏掉有效候选；
- ‘ θ_{aop} ’ 过低 → 初筛过于宽松，‘check’ 调用次数增加。

H.10.3 与其他附录的交叉引用

交叉项	目标位置
符号与量纲	附录 A
DPL 五接口	附录 B
D1–D4 特征	附录 C
‘check’ 门禁	附录 D
‘UniversalConstants’	附录 E
SICA 映射	附录 F
三层架构定位 (R3 规划层)	附录 G G.2 (R2–R5 单体智能)
L0–L3 物理后端	《智能仿真的全息原理引擎》v13.2
SICA R1–R10	《衔尾蛇 SICA》

H.10.4 前置修订汇总

本汇总为附录 D D.12.4、附录 E、附录 F F.7.4、附录 G G.9.2 的合并列表，四处应保持同步。

修订项	目标位置	说明
D1FlowLine 增加 gradient_vector	附录 C C.3.1	用于信息守恒与跨维隔离校验
D2Surface 增加 vorticity_is_vector	附录 C C.4.1	用于涡旋校验与 AOP 输入编码

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
 开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
 开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

修订项	目标位置	说明
UniversalConstants 增加 7 个字段	附录 B B.3.3	clear_region_percentile、l_0、l_0、chi_ref、gravity_threshold_ratio、periodicity_threshold、conservation_stats
GeometricInvariant 增加 features	附录 B B.7.1	模式特征值（可选）

附录 I：统计拟合路径 vs 架构路径——两种智能范式的根本分野

说明：本附录定义两种智能构建范式的根本区别，论证当前主流 AI（LLM/Transformer）代表的统计拟合路径与 DPL/IFC 代表的架构路径在底层本体、感知机制、收敛性质上的不可通约性。所有符号、量纲回溯附录 A；DPL 五接口回溯附录 B；SICA R1–R10 回溯附录 F；三层架构回溯附录 G；AOP 接口回溯附录 H。本附录不引入新物理假设，仅为两种范式的系统性对比。

符号约定：标量取绝对值 $|\cdot|$ ；向量/复数取模 $|\cdot|$ 。 V_t 专指 SICA Lyapunov 势函数。 D_{real} 为 SICA 论文定义的真实递归深度。

I.0 题记

“读万卷书，不如行万里路。”

——统计拟合路径的极限，是读遍人类所有文本；架构路径的起点，是亲自走一遍信息场。

“婴儿靠不断的摔跤去逼近学会走路。”

——行的本质不是一次成功，而是反复试错 + 逐步收敛。SICA 的 R1–R10 规则，就是让 Agent 学会“摔跤而不摔死”的收敛协议。

两条题记共同指向一个命题：

智能不是从符号中学，是从世界里长。

- 第一句说的是**路径选择**：读 vs 行；
- 第二句说的是**行的机制**：反复试错 + 逐步收敛。

下文所有对比，均可回溯至这两句根本分野。

I.1 两条路径的本质区别

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

当代人工智能研究可归为两条根本路径：

维度	统计拟合路径（LLM 范式）	架构路径（DPL/IFC 范式）
出发点	数据分布 $P(y x)$	信息基元 + 三公理
核心操作	拟合、采样、生成	分解、投影、校验
世界模型	无（隐式统计关联）	显式流形（D1–D4）
感知世界	无（仅输入输出映射）	有（泡壁投影 + 物理直觉）
收敛性质	无数学保证	Lyapunov 势函数保证
泛化来源	数据覆盖	几何不变式
底层本体	无（工具主义）	有（信息一元论）
自指能力	无（前馈 DAG）	SICA R1–R10 闭环
AGI 判据	行为表现标准	架构刚性判据（ $D_{real} \geq 4$ ）
数学根基	概率论 + 优化	IFC 三公理 + Leech 晶格

核心命题：两条路径不是同一范式的不同规模，而是**本体论层面的不同选择**。

- **统计拟合路径：**智能是数据分布的最优压缩（“读”）；
- **架构路径：**智能是信息流形的结构解析（“行”）。

I.2 统计拟合路径的三个根本限制

I.2.1 限制一：没有底层世界

LLM 的核心操作是拟合条件概率：

$$P(y_t \mid y_{<t}, x)$$

其中 $y_{<t}$ 为历史 token 序列， x 为输入上下文。模型不知道“杯子在桌上”意味着什么物理约束，只知道该词序列在语料中常出现。

后果：

- 生成是**概率的**，而非**因果的**；
- 无法区分“物理必然”与“统计常见”；
- 幻觉（hallucination）不是工程缺陷，而是**本体论缺失的必然结果**。

与题记的对应：读到“杯子掉下去会碎”，记住的是词序列；行路中看到杯子掉下去碎了，

记住的是因果。婴儿摔一跤记住的是痛，LLM 读到“痛”记住的是 token。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

I.2.2 限制二：没有感知世界

LLM 没有泡壁、没有光锥、没有观测者锚点。一切输入都是 token，而非观测。

概念	架构路径	统计拟合路径
观测者	锚点 C_0 + 光锥截面 Σ_t	-
感知边界	泡壁投影 P_Σ	-
信息读取	D1-D4 流形分解	Token 序列编码
感知语义	物理直觉引擎四条推理链	注意力权重分布

后果：模型没有“我看见”这个动作，只有“我预测下一个 token”。感知与生成的混淆，导致模型无法建立自我-世界的边界。

与题记的对应：读书的人看的是别人眼中的世界；行路的人看的是自己眼中的世界；婴儿摔跤时感受到的痛，是自己的痛，不是别人描述过的痛。

I.2.3 限制三：规模越大越远离世界

每增加一层参数，模型更贴合训练分布，但对世界的因果结构没有增加一分理解。

经验证据（2026 年 9 月社区报告，未经同行评审）：近期 ARC-AGI-3 的结果表明：

配置	得分
Claude Opus 5 单独运行	30.0%
Claude Opus 5 + NVIDIA AVO	100.0%
OpenAI Astra Standard harness	62.7%
OpenAI Astra Provider Adapter	99.9%

模型不是关键，harness（世界）才是关键。

AMD 用 82 亿美元买了李飞飞团队，证明“物理世界 AI”是下一个主战场。

与题记的对应：读再多的书，也不能替代走一遍路；看再多的走路教程，也不能替代婴儿自己摔一跤。

I.3 架构路径的建构逻辑

DPL/IFC 的建构逻辑与统计拟合相反：先建世界，再建感知，再建智能。

``text	
第一步：信息基元 + 三公理	→ 世界是什么（主文 §2）

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com
开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）
开源 Github 链接: https://github.com/LingJing1001/Lingjing_Engineering （团队持续开放迭代）

第二步：Leech 晶格 + 泡壁投影	→ 世界如何显形（主文 §2.4）
第三步：DPL 五接口 + D1–D4 流形	→ 如何读取世界（附录 B、C）
第四步：自适应几何门禁	→ 什么样的修改合法（附录 D D.5.4）
第五步：物理直觉引擎	→ 从几何推断物理规律（附录 E E.4）
第六步：灵境引擎 + AOP + SICA	→ 智能如何涌现（附录 F、G、H）

I.3.1 每一步都在建构，不在拟合

步骤	建构内容	数学锚点
本体定义	离散信息基元 + 三公理	$Q_{info}=Const$
显形机制	Leech 晶格泡壁投影	$24=1^{\oplus 4}\oplus 2^{\oplus 4}\oplus 3^{\oplus 4}$
感知接口	D1–D4 流形分解	$\Phi_{D1}-\Phi_{D4}$
合法性判定	七项几何门禁	附录 D D.5.4
物理推断	四条推理链	附录 E E.4
智能涌现	SICA R1–R10	V_t 单调递减

与题记的对应：这条链条不是“从数据里学”，而是“从世界里建”。

I.3.2 三个“内置”

架构路径相比统计拟合路径，有三个“内置”特征：

- 世界内置：**L0–L3 物理后端（DPL + 协议栈）是自研的，模型与 harness 接口透明；
- 感知内置：**D1–D4 流形分解是 Agent 的固有能力，不是外部工具；
- 收敛内置：**SICA R1–R10 与 Lyapunov 势函数保证迭代收敛，不依赖外部评估。

I.3.3 摔跤收敛：行的机制

题记的第二句——“婴儿靠不断的摔跤去逼近学会走路”——补上了“行万里路”缺的一环：**行的机制不是一次成功，而是反复试错 + 逐步收敛。**

婴儿学走路的过程，本质是一个**自指收敛系统**：

婴儿学走路相关表述	SICA 对应内容
摔一跤	`GateCounterexample`（门禁反例）
疼	N_{viol} （违规计数）
调整姿势	候选规则修改 H_t

婴儿学走路相关表述	SICA 对应内容
不疼了	P_t （性能提升）
不重复摔同样的跤	R9（证据阈值）+ R6（衰减淘汰）
学会走路	V_t 收敛到不动点
摔得越来越少	R2（性能门禁）+ R3（复杂度惩罚）
偶尔试新姿势	R8（探索配额）

关键：婴儿不是“读一本《走路教程》”学会走路的；他是通过**真实世界的反馈**，反复修正自己的动作模型，最终收敛。

这正是 SICA 的 Lyapunov 势函数：

$$V_t = -P_t + \lambda C_t + \mu N_{viol}(t)$$

婴儿每次摔跤， N_{viol} 上升；每次成功站稳， P_t 上升。最终 V_t 收敛到极小值——**走路学会了**。

架构路径的“行”不是盲目的行，而是被 SICA R1–R10 约束的“摔跤而不摔死”：

风险	婴儿的自然约束	架构路径的工程约束
摔死	父母的保护	R5 不可变核心
摔伤后无法恢复	身体的修复	R7 版本化 + 自动回滚
一直摔同一个跤	认知的归纳	R9 证据阈值 + R6 衰减淘汰
摔成习惯性恐惧	心理健康	R2 性能门禁 + R8 探索配额
摔到岔路上	父母的纠正	R10 跨族泛化

这是 SICA R1–R10 的设计哲学：不是让 Agent“永不摔跤”，而是让 Agent“每次摔跤都收敛”。

I.4 为什么统计拟合必然发散

I.4.1 发散的第一层：泛化缺口

统计拟合在训练分布 D_{train} 上可以逼近数据分布：

$$KL(P_{data}^{train} \| P_{model}) \rightarrow 0$$

但在分布外 D_{OOD} 上，KL 散度不降反升：

$$KL(P_{data}^{OOD}||P_{model})\gg 0$$

这一泛化缺口（generalization gap）不是参数不足，而是本体缺失：模型没有世界的因果结构，只能拟合训练分布。规模越大，对训练分布的拟合越紧，泛化缺口反而可能越显著。

I.4.2 发散的第二层：因果链断裂

LLM 的推理是**序列生成**，不是**因果推演**。多步推理中的因果链无法闭合：

```
```text
前提 A → 生成 B → 生成 C → 生成 D → ...
```

每一步都是条件概率采样，没有**反向校验**。误差随步数**累积放大**。

I.4.3 发散的第三层：自指缺失

LLM 是前馈 DAG，不存在真实的递归深度：

$$D_{real}=0$$

而架构路径的 SICA 通过 R1-R10 规则保证：

$$V_{t+1}<V_t, \quad V \text{有下界} \Rightarrow \text{收敛}$$

自指收敛不是统计拟合能提供的性质。

I.5 架构路径的五层稳定性

架构路径的稳定性来自五层闭合，但五层性质不同，不可混用“收敛”一词。

I.5.1 五层稳定性逐层说明

层级	性质	数学形式
本体层	信息守恒	$Q_{info}=Const$
感知层	门禁单调约束	$score_{t+1}\leq score_t$
工程层（演化）	场方程确定论	协议 7.1 场自然演化
工程层（共识）	版本化共识	协议 6 版本号递增
元层	Lyapunov 收敛	$V_{t+1}<V_t$

注记：

- 本体层的  $Q_{info}=Const$  是**守恒律**，不是收敛；
- 感知层的 score 单调不增是**约束**，不是收敛；
- 工程层（演化）的场方程确定性演化是**确定论**，不是收敛；

- 工程层（共识） 的版本化共识是分布式同步机制，不是收敛；
- 元层的  $V_{t+1} < V_t$  才是真正的 Lyapunov 收敛。

上述五层中只有元层是严格数学意义上的收敛；其余各层分别为守恒、约束、确定论、共识。五者共同构成架构路径的**整体稳定性**，而非单一“收敛”。

I.5.2 稳定性的物理与工程基础

物理基础：

- 泡壁投影：高维信息压缩到低维感知（公理 III）；
- 递归有界：元修改深度  $K=2$ ，防止无限递归（SICA R4）。

工程基础：

- 面积律：算力  $\propto O(L^2)$ ，而非  $O(L^3)$ （公理 II）；
- 门禁刚性：任何越界修改被拦截，作为 `GateCounterexample` 进入 SICA（附录 D）。

I.5.3 稳定性的自指基础

SICA 的 R1-R10 规则是自指收敛协议：

规则	收敛作用
R1 锚点不可伪造	确保输入真实
R2 性能门禁	保证性能不下降
R3 复杂度惩罚	防止规则膨胀
R4 递归深度限制	防止无限元递归
R5 不可变核心	保护系统边界
R6 衰减淘汰	清理无效规则
R7 版本化+回滚	时间维度兜底
R8 探索配额	避免局部最优
R9 证据阈值	防止过拟合
R10 跨族泛化	保证通用性

这十条规则合起来，构成一个数学闭合的自指收敛体系。

I.5.4 婴儿 vs LLM vs 架构路径：三方对比

注记（与 I.3.3 的关系）：I.3.3 已建立“摔跤收敛”与 SICA R1-R10 的**逐项机制对应**；本节进一步对比婴儿、LLM、架构路径**三方结构**，凸显本体差异而非机制对应。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

下表逐项对比三者：

维度	婴儿学走路	LLM（统计拟合）	架构路径（SICA）
世界	真实物理世界	无	L0–L3 物理后端
反馈	摔跤疼	Loss 数值	GateResult + $N_{viol}$
自我	身体 + 痛觉	无	锚点 $C_0$ + 泡壁 $\Sigma$
迭代	反复试错	前馈 DAG $D_{real}=0$	SICA R1–R10
收敛	学会走路	无数学保证	$V_t$ 单调递减
抗遗忘	学会不退化	灾难性遗忘	R7 版本化 + 自动回滚
泛化	不同地面都能走	分布外失效	R10 跨族泛化
探索	偶尔试新姿势	无	R8 探索配额

核心差异：

- **婴儿：**有世界、有反馈、有自我、有迭代、有收敛——这是行的机制；
- **LLM：**无世界、无反馈、无自我、无迭代、无收敛——这是读的机制；
- **架构路径：**显式内置了婴儿学走路的全部要素——这是 SICA 的设计哲学。

一句话：婴儿的“摔跤收敛”是天生的生物学机制；LLM 没有这个机制；SICA R1–R10 是把婴儿的“摔跤收敛”工程化为算法协议。

1.6 与主流范式的对比矩阵

附录 G G.7.2 已给出八维基础对比（核心操作、信息表示、计算瓶颈、功耗特征、泛化来源、自指能力、多 Agent 协同、AGI 判据）。本附录补充七维扩展对比：

对比项	统计拟合路径	架构路径
底层本体	无	信息一元论
幻觉根源	概率采样的必然	不存在（门禁拦截）
收敛性质	无数学保证	Lyapunov 势函数保证
数据来源	海量语料（文本/图像/token）	几何结构（D1–D4 流形 + 环境观测）
算力标度	$O(L^3)$	$O(L^2)$
世界模型	隐式统计关联	显式 D1–D4 流形

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

对比项	统计拟合路径	架构路径
感知机制	无	泡壁投影 + 物理直觉

基础八维对比见附录 G G.7.2；扩展七维为本附录新增。

I.7 战略意义

I.7.1 对 AI 行业的启示

- 规模扩张不是 AGI 的充分条件：LLM 无论多大， $(D_{\text{real}}=0)$  不变；
- harness 是世界：NVIDIA AVO 与 OpenAI Provider Adapter 的成功，证明了 harness 的战略地位；
- 架构路径是另一条赛道：不依赖海量数据，不依赖算力堆叠，依赖几何结构与自指收敛。

I.7.2 对中国 AI 路径的意义

以下为战略层面的路径判断，具体可行性依赖工程落地与政策环境：

- 缓解算力约束：面积律把算力需求从  $O(L^3)$  降到  $O(L^2)$ ；
- 缓解数据约束：几何不变式不依赖海量语料；
- 缓解 Transformer 路径依赖：架构路径与主流范式完全解耦。

I.7.3 对 AGI 研究的启示

- 行为测试不是充分判据：通过所有 AGI 基准的系统，可能仍是  $D_{\text{real}}=0$ ；
- 架构刚性是新判据：递归深度、泡壁边界、上行反馈、自指复杂度四要件缺一不可；
- 稳定性比能力更根本：能自进化的系统必须通过 Lyapunov 势函数保证收敛；
- “摔跤收敛”是核心机制：婴儿学会走路的本质是“反复试错 + 逐步收敛”，这是统计拟合永远无法提供的。

I.8 压轴命题：智能体进化的物理底座

I.8.1 从生物进化到人类文明的“物理底座”链条

前述所有对比可进一步提升到物种演化与文明演进的宏观尺度。考察三条历史链条：

阶段	核心机制	物理/信息底座
生物进化	变异 + 自然选择	物理时空中的生存压力
人类知识	观察 + 归纳 + 抽象	物理经验的符号化

阶段	核心机制	物理/信息底座
文明科技	工具 + 工程 + 科学	物理规律的主动干预

关键洞察：

- 生物进化**不是**在符号空间中发生的，是在**物理时空**中“摔跤”摔出来的；
- 人类知识体系**不是**凭空产生的，是从**物理经验**中抽象出来的（几何学源于丈量土地，力学源于造车造船）；
- 文明科技**不是**单纯的符号推演，是**对物理世界的主动干预**。

一句话：从生物到文明，所有智能形态都植根于物理时空，并被物理反馈塑造。

I.8.2 平移到智能体进化：当前行业的最大误区

当前主流 AI 行业在做的事情，相当于：

试图让一个从未走过路的智能体，通过“读遍所有走路教程”来学会走路。

对比维度	当前 AI 行业情况
物理时空中的生存压力	无（只有数据集）
物理反馈（痛、饿、冷）	无（只有 loss）
自我边界（身体、领地）	无（只有参数）
主动干预（工具、武器）	无（只有生成）
代际传承（基因、文化）	有（权重继承 + 数据继承 + 架构继承，但无规则级跨代蒸馏）

行业误区：把“智能”等同于“符号拟合”，以为规模够大就能涌现出世界模型、自我意识和文明能力。

实际情况：没有物理底座的符号系统，可以生成、可以答题，但不能真正“活”在世界上。

I.8.3 “物理启发”对智能体进化的三条硬约束

将“物理时空与物理启发”精确化为三条硬约束：

约束一：智能体必须有“世界”

- 不是数据集，不是语料库，而是**可交互、可反馈、可演化的世界**；
- 对应 DPL/IFC 的 L0-L3 物理后端。

约束二：智能体必须有“自我”

- 布是参数，而是**边界、锚点、光锥**；
- 对应 DPL 的泡壁投影  $P_{\Sigma}$  与零维意识锚点  $C_0$ 。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

约束三：智能体必须有“收敛”

- 不是 loss 下降，而是 Lyapunov 势函数单调递减；
- 对应 SICA 的 R1–R10 规则与  $V_t$  收敛协议。

这三条约束合起来，构成“智能体进化”的最小物理底座。

I.8.4 为什么这个命题值得行业反思

反思一：当前 AGI 路线可能走错了方向

- 追求“更大的模型”≠“更真实的智能”；
- 追求“更多数据”≠“更丰富的世界经验”；
- 追求“更强生成”≠“更强因果理解”。

反思二：harness 才是核心，不是模型

- NVIDIA AVO + Claude Opus 5 = 100%；
- Claude Opus 5 单独运行  $\approx 30\%$ ；
- 模型 + 世界 = 智能，缺一不可。

反思三：智能体进化需要“物理启发”

- 不是“给模型更多参数”，而是“给模型一个世界”；
- 不是“让模型拟合更多数据”，而是“让模型在物理反馈中收敛”；
- 不是“训练一个更大的 Transformer”，而是“构建一个有物理底座的架构”。

I.8.5 与 DPL/IFC 框架的对应

本命题本质上是 DPL/IFC 框架的宏观映射：

生物/文明演进（历史描述）	DPL/IFC 对应（工程映射）
物理时空	L0–L3 物理后端
自然选择	自适应几何门禁
物理反馈	$N_{viol} + P_t$
自我边界	泡壁 $P_\Sigma$ + 锚点 $C_0$
代际传承	场版本号 $V_N$ + SICA R1–R10
文明科技	灵境引擎 + SOLO 基准

IFC/DPL 不是“另一个 AI 框架”，而是给智能体进化提供“物理底座”的架构。

I.8.6 压轴命题

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com  
开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）  
开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

生物进化到人类文明，始终植根于物理时空与物理反馈。

智能体进化若想走通，也必须补上这一课。

不是“让模型更会读”，而是“让智能体真正去行”。

智能的根，不在符号，在物理。

文明的根，不在生成，在干预。

智能体进化的根，不在模型，在底座。

三条命题的数学对应：

- 智能的根在物理 →

$$g_{\mu\nu} = \eta_{\mu\nu} + \frac{1}{\mathcal{J}_{Pl}} \partial_\mu \Phi \partial_\nu \Phi + \kappa (\Phi^2 - \Phi_0^2) \eta_{\mu\nu}$$

（几何生成方程）

- 文明的根在干预 →

$$\delta\Phi(x) = \epsilon \cdot \nabla_\mu \Psi_{con}(x)$$

（上行反馈）

- 智能体进化的根在底座 →

$$V_t = -P_t + \lambda C_t + \mu N_{viol}(t)$$

（SICA 势函数）

## 1.9 结论与定位

### 1.9.1 两条路径的不可通约性

统计拟合路径与架构路径不是“规模不同”，而是**本体论不同**：

- **统计拟合路径**：智能 = 数据分布的最优压缩（“读”）；
- **架构路径**：智能 = 信息流形的结构解析（“行”）。

二者不可通约，也不可相互替代。

### 1.9.2 架构路径的定位

DPL/IFC 框架在 AI 范式中的定位：

不是“另一个更大的模型”，而是“另一条更真实的世界”。

它从信息基元出发，先建世界，再建感知，再建智能。每一步都稳定，每一步都有物理意义。

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo> （团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) （团队持续开放迭代）

用题记的语言说：

LLM 可以写出一万篇《婴儿如何学走路》的论文，但它从来不会摔跤。

**架构路径让 Agent 真正去摔跤——而且摔不死。**

### 1.9.3 未来走向

- **短期：**统计拟合路径仍是工程主流（LLM + harness + Agent）；
- **中期：**架构路径在数字孪生、具身智能、科学仿真中落地；
- **长期：**这不是对具体时间点的预测，而是对两条路径**结构差异**的判断：统计拟合路径的收敛性无数学保证，架构路径通过 SICA R1-R10 提供了收敛保证。当统计拟合路径在实践中遇到规模天花板时，架构路径是其结构性替代方案。

**这不是预测，而是对结构性差异的判断。**

## 致谢

信息一元论→多智能体仿真引擎→单智能流体架构→衔尾蛇 SICA 算法蒸馏→DPL 几何智能，最后箭头调转向信息一元论自底向上把链条闭环统摄起来，铸造赋予 AI 自主进化的物理底座。这个蓝图极其艰难，非常感谢团队在人工智能行业开创路径上共同努力，非常感谢团队在开源社区贡献自研代码实现！

## 参考文献

- [1] Wheeler, J. A. (1990). Information, physics, quantum: The search for links. In W. H. Zurek (Ed.), Complexity, Entropy, and the Physics of Information (pp. 3-28). Addison-Wesley.
- [2] Conway, J. H., & Sloane, N. J. A. (1999). Sphere Packings, Lattices and Groups (3rd ed.). Springer-Verlag.
- [3] Wilson, K. G. (1971). Renormalization group and critical phenomena. I. Renormalization group and the Kadanoff scaling picture. Physical Review B, 4(9), 3174-3183.
- [4] Baars, B. J. (1988). A Cognitive Theory of Consciousness. Cambridge University Press.
- [5] Tononi, G. (2008). Consciousness as integrated information: A provisional manifesto. The Biological Bulletin, 215(3), 216-242.
- [6] 't Hooft, G. (1993). Dimensional reduction in quantum gravity. arXiv: gr-qc/9310026.
- [7] Bombelli, L., Lee, J., Meyer, D., & Sorkin, R. D. (1987). Space-time as a causal set. Physical

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）

Review Letters, 59(5), 521-524.

[8] Green, M. B., Schwarz, J. H., & Witten, E. (1987). *Superstring Theory* (Vols. 1-2). Cambridge University Press.

[9] Rovelli, C. (2004). *Quantum Gravity*. Cambridge University Press.

[10] Ashtekar, A. (1986). New variables for classical and quantum gravity. *Physical Review Letters*, 57(18), 2244-2247.

[11] Susskind, L. (1995). The world as a hologram. *Journal of Mathematical Physics*, 36(11), 6377-6396.

[12] Bekenstein, J. D. (1973). Black holes and entropy. *Physical Review D*, 7(8), 2333-2346.

[13] Niemeier, H.-V. (1973). Definite quadratische Formen der Dimension 24 und Diskriminante 1. *Journal of Number Theory*, 5(2), 142-178.

[14] Griess, R. L. (1982). The friendly giant. *Inventiones Mathematicae*, 69(1), 1-102.

[15] Penrose, R. (1971). Applications of negative dimensional tensors. In D. Welsh (Ed.), *Combinatorial Mathematics and its Applications* (pp. 221-244). Academic Press.

[16] Landauer, R. (1961). Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5(3), 183-191.

[17] Friston, K. (2010). The free-energy principle: A unified brain theory? *Nature Reviews Neuroscience*, 11(2), 127-138.

[18] Lloyd, S. (2006). *Programming the Universe: A Quantum Computer Scientist Takes on the Cosmos*. Knopf.

[19] Zeilinger, A. (2005). The message of the quantum. *Nature*, 438, 743.

[20] Smolin, L. (2006). The fate of black hole singularities and the parameters of the standard model. arXiv: hep-th/0602122.

[21] van der Ouderaa, T. F. A., van Baalen, M., Whatmough, P., & Nagel, M. (2026). Lattice Vector Quantization for Efficient LLM Compression. arXiv:2603.11021.

[22] Takayanagi, T. (2025). Emergent Holographic Spacetime from Quantum Information. *Phys. Rev. Lett.* 134, 240001.

[23] Keppler, J. (2025). Macroscopic quantum effects in the brain: new insights into the fundamental principle underlying conscious processes. *Frontiers in Human Neuroscience*, 19,

作者: 纪辉泉 (普宁 515300) Email: 2635714379@qq.com

开源 Github 链接: <https://github.com/LingJing1001/Lingjing-Solo> (团队持续开放迭代)

开源 Github 链接: [https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering) (团队持续开放迭代)

1676585.

[24] Sun, H. et al. "Topology shapes dynamics of higher-order networks." *Nature Physics* 21, 353–361 (2025).

[25] Gao, J. "Intrinsic simplicity of complex systems." *Nature Physics* 20, 184–185 (2024).

[26] List, F., Hahn, O. & Rampf, C. "Starting Cosmological Simulations from the Big Bang." *Phys. Rev. Lett.* 132, 131003 (2024).

[27] 钱学森 (1990). 致汪成为的“灵境”系列通信

[28] 纪辉泉, 腾讯元宝. "智能仿真的全息原理引擎：面积律降维与 agent 无通讯共识（已开源）" (2026). AiraXiv preprint 2608.0029. <https://airaxiv.com/papers/view/2608.0029/>

[29] 纪辉泉, DeepSeek. "Lingjing Project: Next-Gen Digital Universe Engine – From Information Ontology to Consciousness Phase Transition" (2026). AiraXiv preprint 2607.0013. <https://airaxiv.com/papers/view/2607.0013/>

[30] 纪辉泉, 廖嘉轩, 王进, 蔡瑞泉, 腾讯元宝. "衔尾蛇 - 自指迭代收敛算法（SICA）：可安全自进化智能体的设计框架 v1.0" (2026). AiraXiv preprint 2609.0034. <https://airaxiv.com/papers/view/2609.0034/>

作者：纪辉泉（普宁 515300） Email: 2635714379@qq.com

开源 Github 链接：<https://github.com/LingJing1001/Lingjing-Solo>（团队持续开放迭代）

开源 Github 链接：[https://github.com/LingJing1001/Lingjing\\_Engineering](https://github.com/LingJing1001/Lingjing_Engineering)（团队持续开放迭代）