

# SeedCup 2025 智能桥梁生成系统技术报告

## 1. 系统概述

本系统旨在针对 SeedCup 2025 物理造桥比赛，提供一套自动化、高稳健性且具备成本效益的桥梁生成方案。系统核心基于几何分析与启发式规则，能够自适应地处理不同地形（深渊、悬崖、障碍物），并利用多线程并发技术实现高效的批量仿真与测试。

## 2. 核心算法架构 (BridgeSolver)

核心逻辑封装于 `frame/bridge_solver.py` 中，采用分层生成的策略，将复杂的造桥问题分解为路径规划、结构生成与优化剪枝三个阶段。

### 2.1 地形感知与环境建模

在生成任何结构前，算法首先对地图数据进行深度解析：

- **起终点识别**：自动识别地图中的 `pinned` 节点，并根据 Y 轴坐标区分主起点 (Main Start) 与副起点，支持多层桥面生成。
- **环境分类 (`_classify_environment`)**：基于起终点连线，将地图中的固定节点划分为：
  - **Ceiling (天花板节点)**：位于路径上方，可作为悬索的锚点。
  - **Support Clusters (地面支撑)**：位于路径下方，聚类后作为桥墩的着力点。

### 2.2 智能路径规划

- **避障逻辑 (Waypoints)**：递归检测起终点连线与地图障碍物的碰撞情况。一旦发现冲突，算法会自动生成“避险点” (Waypoints)，引导桥面绕过障碍。
- **平滑曲线生成**：采用分段线性插值与平滑算法 (`_smooth_path`)，将起点、避险点与终点串联，生成符合物理规律的拱形或平缓桥面路径。

### 2.3 混合式结构生成 (Hybrid Structural Generation)

这是本系统的核心创新点，结合了悬索结构与桁架结构的优势：

1. **悬索优先策略 (Suspension First)**：
  - 算法优先检测桥面节点上方是否存在可用的“天花板节点”。
  - 若存在且距离适中 (< 12.0 单位)，则生成钢索 (Steel) 进行悬挂支撑。
  - **优势**：有效利用地形，大幅减少下方支撑材料的使用，降低造价。
2. **自适应几何桁架 (Adaptive Geometric Truss)**：
  - 对于未被悬索覆盖的区域，算法生成经典的“倒三角”桁架结构。
  - **动态高度调整**：桁架的高度不再是固定的，而是根据下方支撑点的距离动态计算。
    - **深渊处**：桁架更高 (更厚)，提供更大的抗弯刚度。
    - **近地处**：桁架更扁平，节省材料。
  - **冗余剔除**：若某段桥面已被悬索拉住，下方的桁架会自动“压扁”至最小高度 (1.0)，仅保留基本的连接功能，避免结构冗余。
3. **自动锚定**：

- 生成的桁架节点会自动搜索附近的地图固定点 (Pinned Nodes)，若距离小于阈值 (8.0)，则自动建立连接，形成稳固的物理支撑。

## 2.4 结构优化与清理

- 迭代剪枝 (\_prune\_dangling\_sticks)**: 在结构生成完毕后，算法会进行多轮迭代，自动识别并删除那些只连接了一个非固定点的“悬空”木材杆件，进一步降低桥梁自重与成本。

## 3. 并行仿真流水线

为了应对多关卡的测试需求，系统设计了高效的并行仿真框架。

### 3.1 多进程并发 (run\_parallel.py)

系统使用 Python 的 `subprocess` 模块，能够同时启动多个仿真进程，并行运行不同的关卡。

- 流水线化**: 主控脚本遍历关卡列表，依次启动子进程，互不干扰。
- 参数透传**: 支持将 `--nosim` 等参数透传给子进程，灵活控制仿真行为。

### 3.2 自动化可视化与窗口管理

- Pygame Hook 技术**: 在 `main_auto.py` 中，通过 Hook `pygame.display.set_mode` 和 `pygame.display.flip`，实现了以下功能：
  - 强制缩放**: 将仿真窗口强制缩小为原尺寸的 50%，便于在屏幕上同时观察多个并行窗口。
  - 自动截图**: 在仿真首帧自动保存截图至 `img/` 目录，便于后续分析。

## 4. 关键代码展示

### 4.1 混合支撑生成逻辑 (\_grow\_supports)

```

def _grow_supports(self, deck_points, ceiling_nodes, ground_clusters,
is_main=False):
    # ... (前置逻辑)

    # [Suspension Logic] 悬索优先
    suspended_indices = set()
    if ceiling_nodes:
        for i in range(len(deck_points)):
            pt = deck_points[i]
            # 寻找最近的上方锚点
            # ... (搜索逻辑)
            if best_ceil and min_ceil_dist < 12.0:
                self._add_stick(pt, (best_ceil['x'], best_ceil['y']), 'steel')
            suspended_indices.add(i)

    # [Truss Generation] 桁架生成
    for i in range(len(deck_points) - 1):
        # ... (计算中点与法向量)

        # [Optimization] 高度自适应
        if i in suspended_indices or (i+1) in suspended_indices:
            truss_height = 1.0 # 悬挂区域使用最小高度
        else:

```

```
# 根据下方支撑距离动态计算高度  
base_height = max(2.0, 5.0 - (min_dist / 15.0))  
truss_height = max(1.0, base_height - 1.0)  
  
# ... (生成节点与连杆)
```

## 4.2 并行任务调度 (`run_parallel.py`)

```
def run_parallel_simulations():  
    processes = []  
    # ...  
    for i, level in enumerate(levels_to_run):  
        cmd = [sys.executable, 'main_auto.py', '--level', level] + extra_args  
        # 启动子进程  
        p = subprocess.Popen(cmd)  
        processes.append(p)  
        time.sleep(0.5) # 错峰启动  
  
    # 等待所有任务完成  
    for p in processes:  
        p.wait()
```

## 5. 总结

本系统通过结合几何算法、物理启发式规则与并行计算技术，实现了一个高效、稳健的自动化造桥解决方案。特别是“悬索优先”与“自适应桁架”的混合策略，在保证结构强度的同时，显著优化了材料成本与结构美观度。