

基於深度強化學習的自動化佈局與路徑規劃系統技術規格書

1. 系統概述 (System Overview)

本文件描述一套基於 **Neural Combinatorial Optimization (神經組合優化)** 的自動化佈局系統。該系統旨在解決二維空間內的 **動態物件佈局 (Dynamic Object Placement)** 與 **資源路徑最小化 (Resource Routing Minimization)** 問題。

系統核心採用 **Transformer** 與 **CNN** 的混合架構，並結合 **Pointer Network** 機制，使其能夠處理變動數量的輸入物件，並在滿足無重疊、邊界限制及特定資源連接需求的前提下，輸出最佳化的空間座標與旋轉角度。

2. 問題定義 (Problem Definition)

本問題被定義為一個 **序列決策過程 (Sequential Decision Making Process)**，並建模為 **馬可夫決策過程 (MDP)**。

- 目標：**將集合 $S = \{O_1, O_2, \dots, O_N\}$ 中的 N 個異質物件，放置於固定大小的網格環境 G 中。
- 約束：**
 - 幾何約束：**物件之間不可重疊，且不可超出環境邊界。
 - 資源約束：**每個物件需連接至特定的資源節點 (Resource Nodes)，需最小化路徑成本。
 - 維護約束：**物件周圍需保留特定的操作空間 (Clearance)。

3. 系統架構設計 (System Architecture)

系統採用 **Encoder-Decoder** 架構，分為特徵提取 (Encoding)、上下文融合 (Context Embedding) 與 策略輸出 (Policy Head) 三個階段。

3.1 狀態空間設計 (State Representation)

由於輸入物件的數量與尺寸是變動的，系統設計了雙流 (Dual-stream) 輸入處理：

A. 空間輸入 (Spatial Input) - 環境感知

- 格式：** $H \times W$ 的多通道網格圖 (Grid Map) 或圖像。
- 編碼器：**使用 **CNN (卷積神經網路)** 提取空間特徵。
- 通道設計 (Channels)：**
 - 障礙物層：**標示環境中既有的不可用區域 (0/1)。
 - 佔用層：**標示已放置物件的區域。
 - 資源接入點層：**標示水、電、氣等資源源頭的位置。
- 輸出：**扁平化後的環境特徵向量 h_{map} 。

B. 序列輸入 (Sequence Input) - 物件屬性

- **格式**：長度為 N 的特徵向量列表 (List of Vectors)。
- **編碼器**：使用 **Transformer Encoder (Multi-Head Attention)**。
- **特徵內容**：包含物件的幾何尺寸 (長、寬)、資源需求類型、優先級權重等。
- **優勢**：利用 Self-Attention 機制建立全局觀 (Global Context)，模型能學習物件之間的隱性關係 (如：高振動物件需遠離精密物件)，且具有 **置換不變性 (Permutation Invariant)**。
- **輸出**： N 個包含上下文資訊的物件嵌入向量 (Embeddings) $h_{objects} = \{e_1, e_2, \dots, e_N\}$ 。

注意：不使用 Positional Encoding，因為待佈局物件的輸入順序不應影響結果，視其為集合 (Set) 而非序列 (Sequence)。

4. 決策模型與動作空間 (Policy Network & Action Space)

模型在每個時間步 t (Time Step) 處理一個物件，執行複合動作 a_t ：

$$a_t = \{a_{index}, a_{pos}, a_{rot}\}$$

4.1 子動作 1：物件選擇 (Selection - a_{index})

決定「下一個要放置哪一個物件」。這是一個基於 **Pointer Network** 的機制。

- **輸入**：所有物件的嵌入向量 $h_{objects}$ 。
- **機制**：
 1. **Query**: 當前環境特徵 h_{map} 。
 2. **Key**: 候選物件特徵 e_i 。
 3. **Score Calculation**: 計算相視度 (Dot Product) 或經過 Attention 層。

$$Score_i = \text{Network}(e_i, h_{map})$$

4. **Action Masking (關鍵技術)**：維護一個 `Available_Mask`，將已放置物件的 Score 強制設為 $-\infty$ 。
5. **Output**: 經過 Softmax 得到選擇概率，採樣出物件索引 k 。

4.2 特徵融合 (Feature Fusion)

一旦選定物件 k ，提取其專屬特徵 e_k ，與環境特徵結合：

$$Z = \text{Concat}(h_{map}, e_k)$$

此向量代表：「在當前地圖狀態下，要放置這個特定物件」。

4.3 子動作 2：位置選擇 (Positioning - a_{pos})

- **輸入**：融合特徵 Z 。

- **解碼器**：使用 **Deconvolution (轉置卷積)** 或 MLP 將特徵重塑回網格狀分佈。
- **Output**： $H \times W$ 的 Logits Map。
- **Masking**：根據物件 k 的尺寸，計算網格中所有無法放置（會重疊或超界）的中心點，將其 Logits 設為 $-\infty$ 。
- **Action**：採樣得到座標 (x, y) 。

4.4 子動作 3：旋轉選擇 (Rotation - a_{rot})

- **輸入**：融合特徵 Z 與位置特徵。
- **Action Space**：離散空間 $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$ 。
- **Output**：4 個方向的機率分佈。

5. 獎勵函數設計 (Reward Shaping)

引導 Agent 行為的核心在於多目標獎勵函數：

$$R_{total} = w_1 \cdot R_{validity} + w_2 \cdot R_{routing} + w_3 \cdot R_{maintenance} + w_4 \cdot R_{adjacency}$$

1. $R_{validity}$ (**合法性獎勵**)：成功放置且無重疊給予大獎勵。若使用了 Action Masking，此項可簡化為「放置完成率」。
2. $R_{routing}$ (**路徑成本**)：計算從物件到資源點的距離。距離越短、轉折越少，獎勵越高（或懲罰越低）。
3. $R_{maintenance}$ (**空間保留**)：物件周圍是否保留了足夠的通道寬度（基於 Manhattan Distance 計算）。
4. $R_{adjacency}$ (**關聯性**)：若物件 A 的輸出需傳遞給物件 B，則兩者距離應盡可能接近。

分層處理策略 (Hierarchical Approach)

為了避免 RL 動作空間過大，採用「分工」策略：

- **大腦 (RL Agent)**：只負責決定高層次的佈局 (x, y, θ) 。
- **手腳 (Solver)**：當 RL 決定位置後，立即調用傳統演算法（如 A Algorithm* 或 Dijkstra）計算具體的連線路徑。
- **回饋**：將 Solver 算出的「路徑總長度」轉化為負的 Reward 回傳給 RL。

6. 訓練流程 (Training Process)

6.1 演算法選擇

採用 **PPO (Proximal Policy Optimization)**。PPO 適合處理連續或高維度的離散動作空間，且訓練穩定性高。

6.2 課程學習 (Curriculum Learning)

因環境與物件組合極其複雜，訓練必須採用漸進式難度：

1. **Level 1**：正方形規則房間，無障礙物，少量物件 ($N=3$)。
2. **Level 2**：長方形房間，含內部柱子，中量物件 ($N=5$)。
3. **Level 3**：複雜多邊形房間，固定資源點，大量物件 ($N=10+$)。
4. **Level 4**：真實歷史案例模擬。

6.3 推論流程 (Inference Loop)

1. **Input**: 獲取初始環境地圖 G_0 與待放置列表 S_{remain} 。
2. **Select**: 模型輸出 a_{index} 選擇物件 k 。
3. **Place**: 模型輸出位置 (x, y) 與角度 θ 。
4. **Update**:
 - 在地圖 G 上繪製物件 k (更新佔用層)。
 - 從列表 S_{remain} 中移除 k (更新 Mask)。
 - 調用 Solver 計算並記錄路徑成本。
5. **Loop**: 重複上述步驟直到 S_{remain} 為空。

7. 總結 (Conclusion)

本架構整合了 CNN 的空間感知能力與 Transformer 的序列推理能力，並透過 Pointer Network 解決了變動輸入數量的難題。相比於傳統的啟發式演算法 (Heuristic) 或遺傳演算法 (GA)，本 RL 系統具備以下優勢：

1. **泛化能力**：能夠處理未曾見過的房間形狀與物件組合。
2. **端到端優化**：直接針對最終的佈局指標（路徑總長、空間利用率）進行優化。
3. **隱性知識學習**：模型能夠自動學會「將大物件靠邊放」、「將高資源需求物件靠近源頭」等策略，而無需人工撰寫規則。