# 基于人工智能的 BIM 疏散设计自动化方法

梁裕卿[1,2]， 吉久茂[1,2]， 杨佳蕾[3]， 张东升[1,2]，
王 珂[4]， 王凌宇[1,2]

(1. 同济大学建筑设计研究院(集团)有限公司，上海 200092；
2. 上海建筑数字建造工程技术研究中心，上海 200092；
3. 普华永道信息技术(上海)有限公司数字方案部，上海 200021；
4. 同济大学建筑与城市规划学院，上海 200092)

**摘 要**：针对目前建筑信息模型(BIM)消防疏散路径人工绘制的耗时问题，从提高设计效率出发，提出了一种基于深度 Q 学习(DQN)与 A*结合的混合算法，并以此开发了一种基于该算法的 BIM 疏散自动设计工具。首先，房间疏散路径使用 A*算法进行绘制；然后使用改进的 DQN 算法确定楼层疏散中疏散门至安全出口的路径再以 A*算法绘制。在 DQN 算法的基础上重新设计了奖励矩阵赋值及增加了奖励矩阵验证机制提高了绘制正确性；最后，使用以该算法为基础的疏散设计自动化工具对实际项目进行了实验。结果表明，该算法不仅能正确绘制路线并且比手工绘制效率提高 2~3 倍。通过服务器部署及硬件的升级为该算法效率进一步提升提供了可能。目前基于该算法的自动设计工具已在同济设计院上海建筑数字中心的多项实际项目中使用。

**关 键 词**：人工智能；建筑信息模型；疏散；设计自动化；深度 Q 学习
**中图分类号**：TP 391 **DOI**：10.11996/JG.j.2095-302X.2021020299
**文献标识码**：A **文 章 编 号**：2095-302X(2021)02-0299-08

# BIM evacuation design automation based on artificial intelligence

LIANG Yu-qing[1,2], JI Jiu-mao[1,2], YANG Jia-lei[3], ZHANG Dong-sheng[1,2],
WANG Ke[4], WANG Ling-yu[1,2]

(1. Tongji Architectural Design (Group) Co.,Ltd, Shanghai 200092, China;
2. Shanghai Digital Architecture Fabrication Technology Center, Shanghai 200092, China;
3. Department Digital Solutions, PricewaterhouseCoopers Zhong Tian LLP (Shanghai), Shanghai 200021, China;
4. College of Architecture and Urban Planning, Tongji University, Shanghai 200092, China)

**Abstract:** To reduce the time cost of manual design of the Building Information Modeling (BIM) fire evacuation routes and to enhance design efficiency, an improved algorithm was proposed based on Deep Q Learning (DQN) and A* algorithm, through which a BIM evacuation design automation tool developed. First, the evacuation path of the room was drawn by A* algorithm. Then, the path from the evacuation door to safety exit in floor evacuation was determined using the improved DQN algorithm, and was drawn by A* algorithm. On the basis of the DQN algorithm, we redesigned reward matrix assignment and added reward matrix verification to improve the correctness of floor evacuation. Finally, we applied the improved algorithm-based evacuation design automation tool to practical projects. The results show that this improved algorithm can not only draw the route correctly, but also increase the efficiency by

**收稿日期**：2020-08-21；**定稿日期**：2020-10-23
**Received**：21 August，2020；**Finalized**：23 October，2020
**基金项目**：教育部重点实验室(同济大学)开放课题(2019010103)
**Foundation items**：Open Projects Fund of Key Laboratory of Ministry of Education (Tongji University) (2019010103)
**第一作者**：梁裕卿(1990–)，男，上海人，工程师，硕士。主要研究方向为 BIM 软件开发。E-mail：51lyq@tjad.cn
**First author**：LIANG Yu-qing (1990–), male, engineer, master. His main research interest covers BIM software development. E-mail：51lyq@tjad.cn

2 to 3 times compared with manual drawing. The efficiency of this algorithm can be further improved by server deployment and hardware upgrades. At present, the automatic design tool has been adopted in several actual projects of Shanghai Digital Architecture Fabrication Technology Center (SFAB), Tongji Architectural Design (Group) Co., Ltd.
**Keywords:** artificial intelligence; building information modeling; evacuation; design automation; deep Q-learning

近年来，建筑信息模型(building information modeling，BIM)在设计阶段的应用已从最初的二维翻模向正向设计逐步过渡。BIM 设计不仅仅只停留在原始的绘图出图，而是更多的为建筑设计提供数据化的建议与指导，帮助设计师及时查找纰漏，做出合理的优化。消防疏散路径绘制是验证设计是否符合建筑消防规范的重要环节[1]。BIM 建模人员在绘制消防疏散施工图时需要花费大量时间在 BIM 软件中勾勒出房间和楼层的疏散路径及疏散方向。所以，自动绘制路径的需求应运而生。经典的自动绘制路径算法有 Dijkstra 算法[2]，A*算法[3-5]等。Dijkstra 算法是典型的最短路径算法，其通过离出发点距离最近的点向外层扩展，至终点。该算法在每次经过下一个节点时需要遍历 Open 集合的所有点，效率较低。而且，距离最近的点是针对出发点而言的，不能保证最短路径的搜索方向始终指向终点，这样会导致搜索方向产生偏差不能得到最优解。针对 Dijkstra 算法的不足，A*算法引入了估价函数，通过约束 Dijkstra 算法的搜索方向，从而减少搜索空间，提升搜索效率。A*算法搜索性能依赖于估价函数的定义，由于其依托于 Dijkstra 算法，仍需要维护 Open/Close 集合，即 A*算法是一种内存受限的启发式搜索算法。所以，A*算法常常与其它自动绘制路径算法相结合，当节点数少时，使用 A*算法进行最后的处理。概率路线图(probabilistic roadmap，PRM)算法[6]是一种结合 A*算法的路径规划算法，其基本思想是从周围环境中布置随机采样点，以相对较少的采样点将连续空间转换成离散空间从而避免点的采样数量过高，然后利用 A*算法在路线图上寻找路径，提高搜索效率。快速扩展随机树(rapidly exploring random tree，RRT)算法[7-8]与 PRM 算法相类似，通过向空白区域扩展树枝得到节点，再通过 A*算法寻找起始点到终点的路径。上述 2 种与 A*结合的算法都是通过随机采样的方式获取节点位置，而在实际设计中楼层/房间的角点或凹点都是固定的，并不需要布置随机采样点。所以其不适用于 BIM 的自动化设计。

随着人工智能(artificial intelligence，AI)技术在各个领域的普及，一些使用 AI 算法绘制疏散路径的方案也纷纷提出。LEE 等[9]提出了基于图的通用流通网络(universal circulation network，UCN)算法，该算法是围绕建筑平面内的所有障碍物，如墙、门、楼梯等空余出一小段缓冲区域以模拟实际人群疏散的情形，同时将所有门点，邻近凹点进行连接形成网络，最后使用 Dijkstra 算法绘制出疏散路径。其不足之处在于依旧需要生成大量节点，并使用了无方向限制的 Dijkstra 算法作为最后的绘制方式。SHARMA 等[10]提出了基于 Q-Learning+DQN 的混合算法处理消防疏散问题，并将其与多种 DQN 变种算法进行横向比较。该算法速度更快，正确率更高。其是先将小空间范围内的房间使用 Q-Learning 算法得到 Q 矩阵，并将该矩阵作为 DQN 算法的初始矩阵，以期望在 DQN 运行之前就获取房间至出口的距离信息，提高运算效率。但经过使用相同的参数配置复现该先验方式发现，Q-Learning 算法[11]并不能适用于所有小空间分布情况，主要原因是 Q-Learning 易陷入局部最优而无法得出结果，同时还受制于矩阵维度约束，即需要另一套机制去分割空间以正常运行 Q-Learning 算法[12-14]或者说需要更强力的硬件支持(故原文作者采用了专用于 AI 研究的 NIVIDIA DGX)。上述 2 种方案都在 AI 方向做出了大胆的创新与尝试，为本文研究自动绘制消防疏散路径提供了丰富的借鉴。

同济设计院数字中心日常使用 Autodesk Revit 平台进行 BIM 设计，所以部门需要一款基于 Revit 的插件工具，方便设计师绘制消防疏散路线图并便于同平台模型检查，减少导入其他平台所产生的时间成本。虽然 Autodesk 已发布的 Revit 2019 版新增了行进路径功能[15]，但其有 4 项局限性：①无法批量对房间进行路径绘制；②只能在 Revit 2019 及以上版本使用；③绘制效果不符合设计院日常习惯；④无法应对较复杂房间构形。为了突破上述的局限性并提高设计效率，本课题组提出了一种基于改进的深度 Q 学习与 A*相结合的混合算法，以避免上述算法的一些先天缺陷并依据该算法开发了基于 Autodesk Revit 平台的疏散设计自动化工具插件，覆盖设计院常用的 Revit 2016 至 2020 版本。

本文提出的混合算法有以下若干特点：

(1) 针对房间的疏散路径使用 A*算法。

(2) 针对楼层疏散路径，首先使用改进的 DQN 算法获取防火分区内所有疏散门到最近安全出口的行走路径，再使用 A*算法进行绘制，避免了 A*算法本身的内存受限问题，提高了计算效率。

(3) 重新设计了奖励矩阵的赋值方式，将门与门之间的长度关系反映到奖励矩阵的值上。

(4) 由于使用了神经网络算法不可避免地会产生小概率误差，所以在改进的 DQN 网络运算之后加入了奖励矩阵验证机制，避免了错误路径的绘制。

最后，在实际设计中对该算法进行了验证，取得了较好的效果，并将其应用于日常施工图消防疏散设计中。

## 1　消防疏散路径的绘制

消防疏散路径绘制包括 2 个部分：房间疏散路径绘制、楼层疏散路径绘制。

房间疏散路径表示的是房间内任一点至疏散门的路径(通常取最远点)。如果一个房间存在多个疏散门，则需要绘制对应数量的疏散路径。

楼层疏散路径表示的是房间内某一点至疏散门的路径以及疏散门至最近安全出口的路径。每一层楼有一个或多个防火分区，每个防火分区有一个或多个安全出口，绘制楼层疏散线就是将防火分区内所有房间及疏散门至最近安全出口的路径绘制出来。

### 1.1　房间疏散线绘制

房间疏散路径绘制需要将 Revit 模型转换成网络节点图。首先需要获取房间周围墙线，将各墙线的端点作为节点。房间周围墙面可能是直线，也可能是带有弧度的弧形墙。弧形墙可以对其进行微小切分，将弧线转换成无数个较短直线集合，将各短直线的端点作为节点。然后找到房间内疏散门的位置，将其加入网络中形成该房间的网络节点图，如图 1 所示。

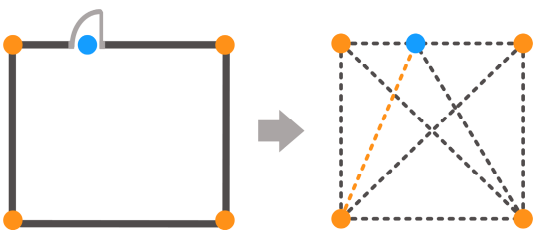

**图 1**　房间网络示意图

**Fig. 1**　Diagram of room network

通过疏散门位置与节点之间距离的大小比较，可以找到房间内离疏散门最远的节点。将该最远节点作为起始点，疏散门的位置作为终点，运用 A*算法得到最短路径并绘制，如图 2 所示。

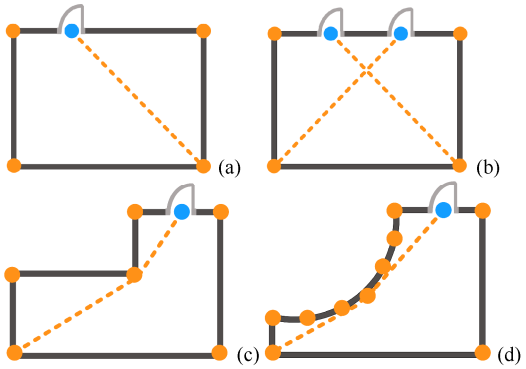


**图 2**　房间疏散示意图((a)单疏散门房间；(b)多疏散门房间；(c)带 L 拐角房间；(d)弧线墙房间)

**Fig. 2**　Diagram of room evacuation ((a) Room with single door; (b) Room with multi-door; (c) Room with L-corner; (d) Room with arc corner)

消防疏散线在图 2 中以虚线表示。A*算法可表示为

$$f(i)=g(i)+h(i) \tag{1}$$

其中，$i$ 为当前节点；$f(i)$为当前节点 $i$ 的估值；$g(i)$为起始点到当前节点 $i$ 的距离；$h(i)$为当前节点 $i$ 到终点的距离。如图 2(b)所示，该房间共有 2 个疏散门，所以分别得到房间内距其疏散门最远的节点作为起始点进行绘制。图 2(c)中，符合 A*算法的中间节点在 L 形角点处。图 2(d)是将弧形墙做了切分处理成若干离散节点形成网络，得到最短疏散路径。

### 1.2　楼层疏散线绘制

防火分区内包含房间及公共区域，如走廊、过道等等。根据上述房间疏散线的绘制思路，可将公共区域视作为“房间”并在 Revit 中为其添加“房间”标记，如图 3 所示。

对于一个公共区域而言，当其被标记为房间后(图 3(a))，只需要知道起始的疏散门与到达的疏散门或安全出口，使用上述的 A*算法即可绘制其消防疏散路径(图 3(b))。

一个防火分区可能存在多个安全出口，防火分区内任意一个疏散门如何找出其通往最近安全出口所经过的疏散门并保证该路径为最短路径？这是一个典型的智能体迷宫问题。由此，本文提出以改进的 DQN 算法来解决上述问题。

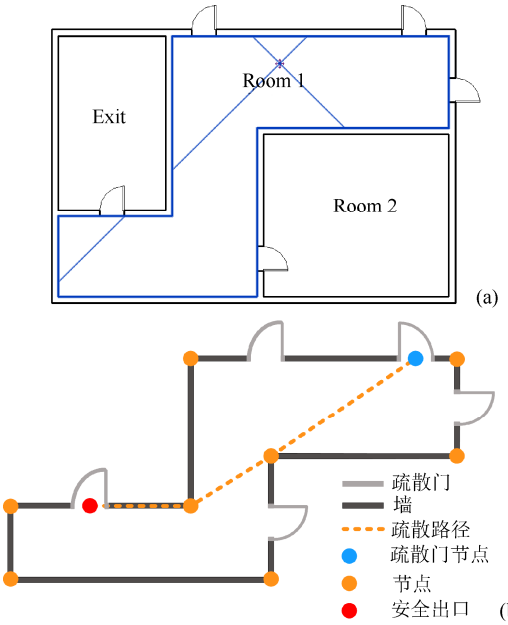


**图 3** 楼层疏散中的公共区域处理
((a) Revit 中的走廊；(b)示意图)
**Fig. 3** Process for public area in floor evacuation
((a) Corridor in Revit; (b) Diagram of corridor)

## 2 算法原理

首先，DQN 算法来源于 Q-Learning 算法，Q-Learning 算法是一种传统的强化学习算法。强化学习是机器学习中的一个子集。强化学习的智能体可以从与环境的直接交互中学习，而无需明确的监督或完整的环境模型。

Q-Learning 中的 Q 表示的是在某一时刻 $s$ 状态($s\in S$)下，采取 $a$ 动作($a\in A$)能够获得收益的期望。所有的状态与动作会形成一张 $\boldsymbol{Q}(s,a)$矩阵。通过 Q-Learning 公式不断更新 $\boldsymbol{Q}$ 矩阵以达到收敛状态

$$\boldsymbol{Q}(s,a)=\boldsymbol{Q}(s,a)+\alpha[r+\gamma\max_{a'}\boldsymbol{Q}(s',a')-\boldsymbol{Q}(s,a)] \quad (2)$$

其中，$\alpha$为学习率；$\gamma$为折扣率；$s'$为下一个状态；$a'$为 $s'$的所有可执行动作中选择奖励值最大的行为。

Q-Learning 对于大维度运算具有局限性并且容易陷入局部最优而处于一直运算的状态无法得到运算结果。DQN 算法[16-17]是将神经网络与 Q-Learning 算法相结合。主要改进在于：

(1) DQN 利用神经网络来逼近值函数，突破原有算法的维数限制。

(2) DQN 利用经验回放训练强化学习的学习过程，让有用的样本得到更多的训练次数，提高样本使用效率，减少训练时的方差，并减轻陷入局部最优的情况。

(3) DQN 独立设置了目标网络来单独处理时序差分中的偏差，增强网络训练的稳定性。

### 2.1 DQN 算法示例

通过图 4 来阐述 DQN 算法的处理过程。

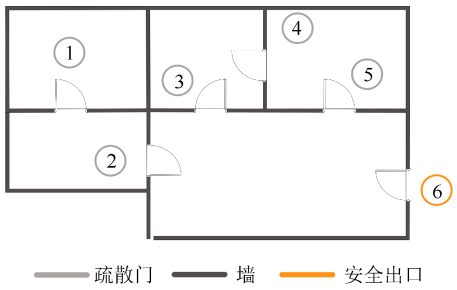


**图 4** 防火分区示意图
**Fig. 4** Diagram of anti-fire section

假设存在一个如图 4 所示的防火分区。首先对每个房间的疏散门进行编号，6 号门作为安全出口，其他房门均为疏散门。根据每个房间与其他房间是否互通的关系，可以得到图 5 的网络示意图。

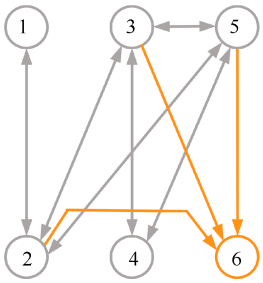


**图 5** 网络示意图
**Fig. 5** Diagram of network

每个房间疏散门都是一种状态(state)，在本例中一共有 6 种状态。每种状态对应的动作(action)可以使用奖励矩阵来表示，如图 6 所示。疏散门如果可直接到达安全出口，其奖励值记为 100；互不相通的疏散门或安全出口，奖励值记为–1；其他情况皆记为 0。

| State \ Action | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | –1 | 0 | –1 | –1 | –1 | –1 |
| 2 | 0 | –1 | 0 | –1 | 0 | 100 |
| 3 | –1 | 0 | –1 | 0 | 0 | 100 |
| 4 | –1 | –1 | 0 | –1 | 0 | –1 |
| 5 | –1 | 0 | 0 | 0 | –1 | 100 |
| 6 | –1 | 0 | 0 | –1 | 0 | 100 |

**图 6** 奖励矩阵
**Fig. 6** Reward matrix

获得该防火分区的奖励矩阵后，即可代入 DQN 算法进行下一步的处理，具体步骤如下：

**步骤 1.** 设置参数：奖励矩阵 $\boldsymbol{R}$，学习率$\alpha$，折扣率$\gamma$，迭代总数 *episodes*，初始化容量为 $N$ 的经验池 $D$。初始化随机权重$\theta$的网络 $Q$。初始化权重为$\theta^-$的网络 $\boldsymbol{Q}_{\text{target}}$，令$\theta^-=\theta$。

**步骤 2.** 对于每一次迭代，执行：

**步骤 2.1.** 随机选择一个状态 $s$。

**步骤 2.2.** 以概率$\varepsilon$在当前状态 $s$ 的所有可执行动作中(即奖励矩阵所在 $s$ 行不等于–1 的列)任意选取一种动作 $a$ 或选取奖励值最大作为动作 $a$。

**步骤 2.3.** 将状态 $s$ 与动作 $a$ 代入奖励矩阵 $R$，得到奖励值 $r$，将选定的动作 $a$ 作为下一个状态 $s'$。

**步骤 2.4.** 将当前状态 $s$，当前执行的动作 $a$，奖励值 $r$，下一个状态 $s'$放入经验池 $D$ 中。如果超过记忆的容量 $N$，则将最久远的记忆移除。

**步骤 3.** 从经验池 $D$ 中随机选取一小批记忆样本，对于每个样本，如果其奖励值为–1，则网络 $\boldsymbol{Q}_{\text{target}}$ 记录该奖励值；否则网络 $\boldsymbol{Q}_{\text{target}}$ 根据以下公式记录结果

$$y=r+\gamma\max_{a'}\boldsymbol{Q}(s',a';\theta) \tag{3}$$

**步骤 4.** 根据损失函数 $L(\theta)$按梯度下降法更新网络权重$\theta$，损失函数为

$$L(\theta)=E[(y-\boldsymbol{Q}(s,a;\theta))^2] \tag{4}$$

将损失函数为最小时的模型参数保存到 data 文件中。

**步骤 5.** 每隔 $C$ 步更新网络 $\boldsymbol{Q}_{\text{target}}$=$\boldsymbol{Q}$。

**步骤 6.** 从 data 文件中取出损失函数值为最小时的矩阵 $\boldsymbol{Q}_{\text{target}}$。

当迭代结束后，可得到损失函数最小时的矩阵 $\boldsymbol{Q}_{\text{target}}$，如图 7 所示。以疏散门 1 为例，在状态为 1 时，所在行最大值为 168.1，即到达了疏散门 2 的位置。状态为 2 时，所在行最大值为 186.78，即直接到达了安全出口，由此形成了疏散门 1 的完整疏散路径：1-2-6，最后使用 A*算法在 Revit 中绘制出具体路径。

## 2.2　改进的 DQN 算法

从图 7 可以观察到，疏散门 4 在 $\boldsymbol{Q}_{\text{target}}$ 矩阵中有 2 个相同的值 168.1，对应疏散门 3 和 5。而疏散门 5 距离安全出口 6 的距离更近，但未在矩阵的值中体现。同时，神经网络算法不可避免地会产生小概率误差，运算结果需要一种简单、有效的方式进行验证。

| State \ Action | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| 1 | −1.64 | 168.1 | −1 | −1.29 | −0.98 | −0.91 |
| 2 | 151.23 | −1 | 168.11 | −1.05 | 168.1 | 186.78 |
| 3 | 151.23 | −1 | 168.11 | −1.05 | 168.1 | 186.78 |
| 4 | −0.94 | −1 | 168.1 | −0.892 | 168.1 | −1.1 |
| 5 | −0.92 | 168 | −1.021 | 151.62 | 168.1 | 186.77 |
| 6 | −1 | 168.1 | 168.11 | 151.28 | −1 | 186.78 |

**图 7**　最终矩阵

**Fig. 7**　Final matrix

因此本文提出了改进的 DQN 算法，主要的改进点在：重新设计了奖励矩阵的赋值方式，将门与门之间的长度关系反映到奖励矩阵的值上；在 DQN 网络运算之后加入了奖励矩阵验证机制，避免了错误路径的绘制。

奖励矩阵的赋值关系可重新定义为：

当状态 $s$ 无法执行动作 $a$ 时，即

$$q(s,a)=-1 \tag{5}$$

当防火分区内所有门与门之间的长度都相同，状态 $s$ 可选择动作 $a$ 时，即

$$q(s,a)=\begin{cases}0, & s\ \&\ a\ \text{not exit}\\ \beta, & s\ \text{or}\ a\ \text{is exit}\end{cases} \tag{6}$$

其中，系数 $\beta\gg 0$ 。

当防火分区内所有门与门之间的长度不同，状态 $s$ 可选择动作 $a$ 时，即

$$q(s,a)=\begin{cases}1-\left(\dfrac{l_{\overline{sa}}-l_{\min}}{l_{\max}-l_{\min}}\right), & s\ \&\ a\ \text{not exit}\\ \beta+\eta\times\left[1-\left(\dfrac{l_{\overline{sa}}-l_{\min}}{l_{\max}-l_{\min}}\right)\right], & s\ \text{or}\ a\ \text{is exit}\end{cases} \tag{7}$$

其中，系数 $\eta\approx 0.1\beta$ 。

对于 DQN 网络运算之后得到的疏散路线集合 $P=\{p_1,\ p_2,\ p_3,\ p_4,\cdots,p_N\}$，其中每条疏散路径 $p=\{Node_1,\ Node_2,\ Node_3,\ Node_4,\cdots,Node_N\}$需要通过奖励矩阵的验证，即

$$q(Node_i,Node_{i+1})\neq -1 \tag{8}$$

综上，改进的 DQN 算法流程图如图 8 所示。

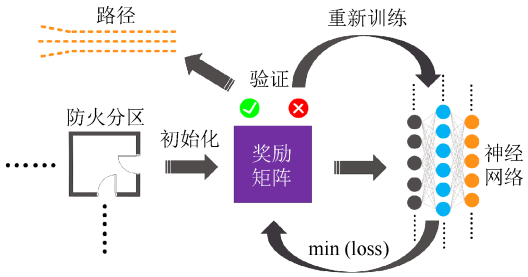


**图 8** 改进的 DQN 算法流程图

**Fig. 8** Flow chart for improved DQN algorithm

## 3 实验与分析

根据本文上述房间疏散与楼层疏散绘制的实现方式，在 Autodesk Revit 平台上使用 C#语言编写了房间疏散与楼层疏散 2 个功能的插件。房间疏散针对单个房间进行绘制，而楼层疏散针对楼层防火分区内所有房间及公共区域进行绘制。改进的 DQN 算法以 Google TensorFlow 的 Python 脚本实现。C#与 TensorFlow 脚本以 csv 格式文件来传递矩阵信息。本文以模型线来表示疏散路径，并以箭头族表示防火分区内室内或疏散门至安全出口的疏散方向。红色虚线表示疏散路径不符合消防疏散规范；而绿色虚线表示符合规范。

参数设置如下：学习率$\alpha$=0.001，折扣率$\gamma$=0.9，探索率从 0.1 降低至 0.000 1。迭代总数为 3 000 000 次，初始化容量为$N$=5000 的经验池，每隔观察 1 000 次更新 $\boldsymbol{Q}$ 矩阵。

### 3.1 房间疏散路径实验

本文以图 2 为例在 Revit 中绘制 4 种不同构形的房间，并执行插件中的“房间疏散”功能，结果如图 9 所示。

通过图 9 可知，使用 A*算法可以正确地绘制出房间疏散路径。设计师只需要点击功能命令，再选中需要绘制的房间即可得到结果，节省了画线、调整线型、放置箭头族并转动正确方向的时间。

### 3.2 单安全出口疏散路径实验

单安全出口疏散路径实验是以某幼儿园项目为例，其防火分区图如图 10 所示，其中安全出口已使用圆圈标出。

执行改进的 DQN 算法得到如图 11 所示结果，整个疏散路径以绿色虚线表示，同时绘有疏散箭头族指向安全出口。疏散路径的结果是正确的。

表 1 对比了单安全出口实验在人工手绘、A*算法、Q-Learning 算法及改进的 DQN 算法中的执行时间。

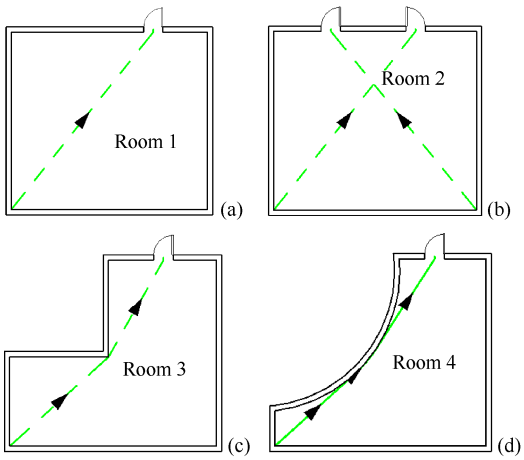


**图 9** Revit 实现房间疏散绘制((a)单疏散门房间疏散路径；(b)多疏散门房间疏散路径；(c)带 L 拐角房间疏散路径；(d)弧线墙房间疏散路径)

**Fig. 9** Room evacuation path in Revit ((a) Path of single door room; (b) Path of multi-door room; (c) Path of L-corner room; (d) Path of arc corner room)

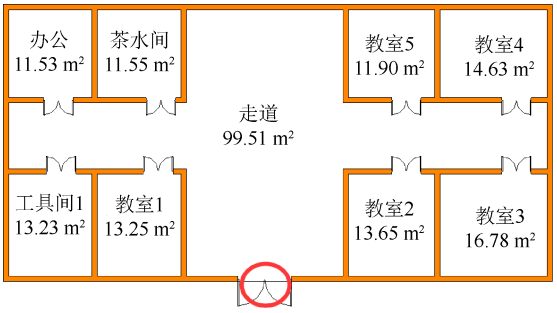


**图 10** 单安全出口的某项目

**Fig. 10** Project of single exit

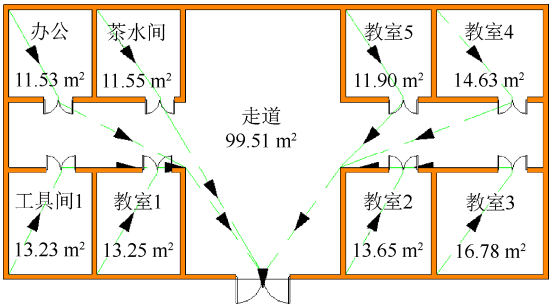


**图 11** 改进的 DQN 运行结果

**Fig. 11** Result of improved DQN for single exit

**表 1 单安全出口实验各绘制方式所用时间对比**

**Table 1 Time cost of different drawing methods for single exit experiment**

| 绘制方式 | 时间(s) |
|---|---|
| 人工手绘 | ~220 |
| A*算法 | <1 |
| Q-Learning 算法 | <1 |
| 改进的 DQN 算法 | 50 |

通过表 1 可以得出，A*算法与 Q-Learning 算法在该项目中的执行速度是非常快的，点击按钮就能

立即得到疏散路径图。其次是改进的 DQN 算法，在防火分区较小的情况下，使用神经网络在本机运行的情况下不具有效率优势，改进的 DQN 算法需要自我训练才能得出结果，而训练时间占用了一定的开销。时间消耗最多的是人工手绘方式，画线、改线样式、放置疏散箭头每个步骤都需要正确操作。

### 3.3　多安全出口疏散路径实验

多安全出口疏散实验以金鼎中心项目为例。某层共有 2 个防火分区，左侧防火分区有 3 个安全出口，右侧防火分区有 2 个安全出口，以绿色指示箭头，如图 12 中红色画圈位置。

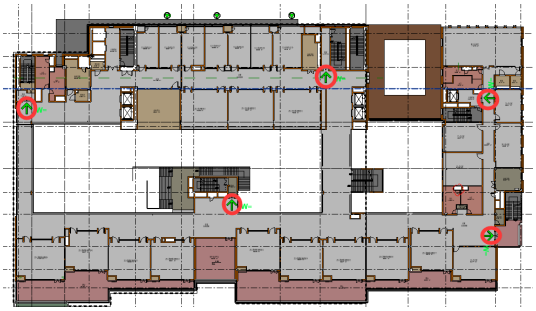

**图 12**　金鼎中心项目某防火分区

**Fig. 12**　Anti-fire section of JD Center

执行改进的 DQN 算法得到如图 13 所示结果。

**图 13**　改进的 DQN 运行结果

**Fig. 13**　Result of improved DQN algorithm for multi-exit

项目确定的消防疏散图如图 14 所示。

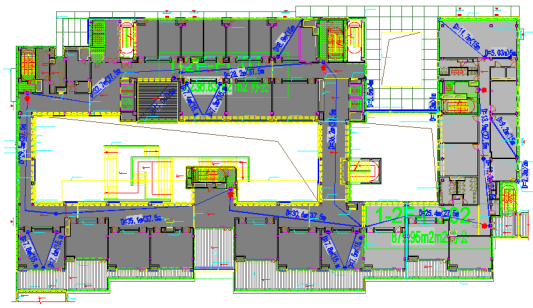

**图 14**　金鼎中心项目消防疏散图

**Fig. 14**　CAD Evacuation path of JD Center

对比图 13 与图 14，发现区别在于：

(1) 图 14 以 CAD 进行绘制，没有标明所有房间的疏散路径，而是以离各安全出口最远处房间为例进行路径绘制。即如果最远处房间能通过消防规范路径长度的话，其他房间亦能通过。

(2) 图 13 是在 Revit 平台上进行自动绘制，对每个房间及公共区域都绘制了疏散路径，比传统 CAD 疏散图更加详尽，在绘图表达上完全体现了建筑设计防火规范 5.5.17 中 2 个距离原则[1]，即“房间疏散门至最近安全出口的直线距离”与“房间内任一点至房间疏散门的直线距离”，并为之后使用该 BIM 数据进行运维提供了便利。

同样，表 2 对比了多安全出口实验在人工手绘、A*算法、Q-Learning 算法及改进的 DQN 算法中的执行时间。

**表 2　多安全出口项目各绘制方式所用时间对比**

**Table 2　Time cost of different drawing methods for multi-exit experiment**

| 绘制方式 | 时间(min) |
|---|---|
| 人工手绘 | ~26 |
| A*算法 | / |
| Q-Learning 算法 | / |
| 改进的 DQN 算法 | ~11 |

从表 2 可知，只有手绘方式与改进的 DQN 算法得出了结果。A*与 Q-Learning 算法由于用时过长一直处于执行状态未记录时间。改进的 DQN 算法用时约 11 min，手绘用时约 26 min。需要说明的是这里的手绘是对每个房间及走道都有连线并放置箭头族，而并非采用图 14 所示的绘图方式。改进的 DQN 算法的效率是手工绘制的 2~3 倍。

## 4　总结与展望

本文提出了一种基于改进 DQN 算法与 A*算法相结合的混合算法用以在 BIM 施工图消防疏散路径中进行绘制与检查。针对房间疏散路径，使用 A*算法进行绘制；针对楼层疏散路径，使用改进的 DQN 算法确定疏散门至安全出口的路径并以 A*算法绘制。本文在 DQN 算法的基础上重新设计了奖励矩阵的赋值方式，将门与门之间的长度关系反映到奖励矩阵的值上；在 DQN 网络运算之后加入了奖励矩阵验证机制，避免了错误路径的绘制。本文提出的算法优势在于适用于大面积的批

量执行，避免了传统算法的内存受限及易局部最优的缺陷。在本文的实验中，该混合算法比手工绘制的效率提高了 2~3 倍。未来可通过将本算法部署于服务器开放 WebAPI 及使用高性能 AI 显卡脱离客户机的硬件限制，进一步提升算法的执行效率。目前，基于该算法的疏散设计自动化工具已应用于同济设计院上海建筑数字中心的多个实际项目中。

## 参考文献 (References)


[1] 中华人民共和国住房和城乡建设部. 建筑设计防火规范: GB 50016—2014[S]. 北京: 中国计划出版社, 2018: 79-80.
Ministry of Housing and Urban-Rural Development of the People's Republic of China. Code for fire protection in building design: GB 50016—2014[S]. Beijing: China Planning Press, 2018: 79-80 (in Chinese).

[2] DIJKSTRA E W. A note on two problems in connexion with graphs[J]. Numerische Mathematik, 1959, 1(1): 269-271.

[3] NILSSON N J. The quest for artificial intelligence[M]. Stanford University: Cambridge University Press, 2009: 1-578.

[4] 张超超, 房建东. 基于定向加权 A*算法的自主移动机器人路径规划[J]. 计算机应用, 2017, 37(S2): 77-81.
ZHANG C C, FANG J D. Path planning of autonomous mobile robot based on directional weighted A* algorithm[J]. Journal of Computer Applications, 2017, 37(S2): 77-81 (in Chinese).

[5] 张启飞, 郭太良. 基于多阶段决策的机器人全局路径规划算法[J]. 计算机工程, 2016, 42(10): 296-302.
ZHANG Q F, GUO T L. Global path planning algorithm of robot based on multistage decision[J]. Computer Engineering, 2016, 42(10): 296-302 (in Chinese).

[6] KAVRAKI L E, SVESTKA P, LATOMBE J C, et al. Probabilistic roadmaps for path planning in high-dimensional configuration spaces[J]. IEEE Transactions on Robotics and Automation, 1996, 12(4): 566-580.

[7] LAVALLE S M. Rapidly-exploring random trees: a new tool for path planning[R]. Technical Report. Computer Science Department, lowa State University, 1998.

[8] LAVALLE S M, KUFFNER J J. Randomized kinodynamic planning[J]. The International Journal of Robotics Research, 2001, 20(5): 378-400.

[9] LEE J K, EASTMAN C M, LEE J, et al. Computing walking distances within buildings using the universal circulation network[J]. Environment and Planning B: Planning and Design, 2010, 37(4): 628-645.

[10] SHARMA J, ANDERSEN P A, GRANMO O C, et al. Deep q-learning with q-matrix transfer learning for novel fire evacuation environment[EB/OL]. [2019-05-23]. https://xueshu.baidu.com/usercenter/paper/show?paperid=1f2d0jm0ms7r0ew0hx3m0th0j9655136&site=xueshu_se.

[11] WATKINS C J. Learning from Delayed Rewards[D]. Cambridge: Cambridge University, 1989.

[12] CHOSHEN L, FOX L, AIZENBUD Z, et al. On the weaknesses of reinforcement learning for neural machine translation[EB/OL]. [2019-05-23]. https://xueshu.baidu.com/usercenter/paper/show?paperid=10370e60073n0pb0pq7b0870y4229607&site=xueshu_se&hitarticle=1.

[13] ANDREY K. Reinforcement learning's foundational flaw[EB/OL]. [2018-07-08]. https://thegradient.pub/why-rl-is-flawed/.

[14] DULAC-ARNOLD G, MANKOWITZ D, TODD H. Challenges of real-world reinforcement learning[EB/OL]. [2019-04-29]. https://arxiv.org/pdf/1904.12901.pdf.

[15] Autodesk Inc. Revit: What's new in the latest Revit release[EB/OL]. [2018-04-18]. https://www.autodesk.com/products/revit/new-features.

[16] MNIH V, KAVUKCUOGLU K, DAVID SILVER, et al. Play atari with deep reinforcement learning[EB/OL]. [2013-12-19]. https://arxiv.org/abs/1312.5602.

[17] MNIH V, KAVUKCUOGLU K, D SILVER, et al. Human-level control through deep reinforcement learning[J]. Nature, 2005, (518): 529-533.