# Arbitrarily Scalable Environment Generators via Neural Cellular Automata (Extended Abstract)*

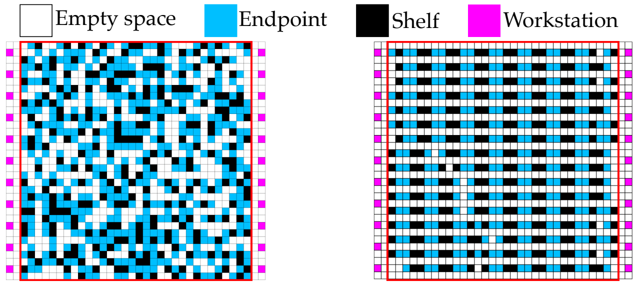
**Yulun Zhang**[1], **Matthew C. Fontaine**[2], **Varun Bhatt**[2], **Stefanos Nikolaidis**[2], **Jiaoyang Li**[1]
[1]Robotics Institute, Carnegie Mellon University
[2]Thomas Lord Department of Computer Science, University of Southern California
yulunzhang@cmu.edu, {mfontain,vsbhatt,nikolaid}@usc.edu, jiaoyangli@cmu.edu

## 1 Introduction

We study the problem of generating environments to improve the throughput of multi-robot systems in automated warehouses. A previous work (Zhang et al. 2023) formulates the environment optimization problem as a Quality Diversity (QD) optimization problem and optimizes the environments by searching for the best allocation of shelf and endpoint locations. Inspired by evolutionary algorithms, QD algorithms are a class of stochastic optimization algorithms capable of generating an *archive* of diverse solutions by simultaneously optimizing an objective and a set of diversity measures. Zhang et al. (2023) use a QD algorithm to iteratively generate new environments and then repairs them with a Mixed Integer Linear Programming (MILP) solver to enforce constraints such as the storage capacity and graph connectivity. The repaired environments are evaluated with a lifelong Multi-Agent Path Finding (MAPF) simulator. Figure 1a shows an example optimized environment.

However, the aforementioned method requires a large number of runs in the lifelong MAPF simulator and the MILP solver. For example, it took up to 24 hours on a 64-core machine to optimize a warehouse environment of size only 36 × 33 with 200 robots, while practical warehouses are reported to have more than 1,000 robots with size up to 179 × 69 (Yu and Wolf 2023).

Therefore, in this paper, instead of optimizing the environments directly, we present a method to train Neural Cellular Automata (NCA) (Earle et al. 2022) environment generators capable of scaling their generated environments arbitrarily. NCA is a convolutional neural network (CNN) that incrementally constructs environments through local interactions between cells, evolving a fixed simple environment to a complex one. We follow a prior work (Earle et al. 2022) and use QD algorithms to efficiently train a diverse collection of NCAs in small environments. We then use the NCAs to generate arbitrarily large environments with consistent and regularized patterns. We adopt the MILP solver (Zhang et al. 2023) to repair the environments in case they are invalid. Figure 1b shows an example NCA-generated environment with regularized patterns.



(a) Optimized environment with no patterns.

(b) NCA-generated environment with patterns.

Figure 1: Examples of optimized warehouse environments. Robots move to blue (endpoints) and pink (workstations) tiles alternatively to transport goods. Black tiles are non-traversable shelves and white tiles are traversable empty spaces. We optimize tiles inside red box.

## 2 Problem Definition

Our definition of *environment* and *valid environment* follows the previous work (Zhang et al. 2023) with the tile types shown in Figure 1. We define the problem of environment optimization as follows.

**Definition 1** (Environment Optimization)**.** *Given an objective function $f : \mathbf{X} \to \mathbb{R}$ and a measure function $\mathbf{m} : \mathbf{X} \to \mathbb{R}^m$, where $\mathbf{X}$ is the space of all possible environments, the environment optimization problem searches for valid environments that maximize the objective function $f$ while diversifying the measure function $\mathbf{m}$.*

## 3 Methods

We extend previous works (Earle et al. 2022; Zhang et al. 2023) to use CMA-MAE (Fontaine and Nikolaidis 2023), a state-of-the-art QD algorithm specialized for continuous search spaces, to train NCAs with the objective and diversity measures computed from a lifelong MAPF simulator. Figure 2 provides an overview of our method. We start by sampling a batch of $b$ parameter vectors $\boldsymbol{\theta}$ from a multivariate Gaussian distribution, forming $b$ NCAs. From a fixed initial

[*] The full paper (https://arxiv.org/abs/2310.18622) is accepted to NeurIPS 2023.

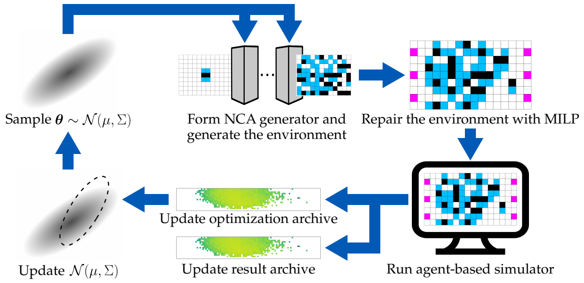

Sample $\boldsymbol{\theta} \sim \mathcal{N}(\mu, \Sigma)$
Form NCA generator and generate the environment
Repair the environment with MILP
Update optimization archive
Update result archive
Run agent-based simulator
Update $\mathcal{N}(\mu, \Sigma)$

Figure 2: Overview of our method of using CMA-MAE to train diverse NCAs.

environment, each NCA generates an environment which is then repaired by a MILP solver. We then evaluate the environments by running a lifelong MAPF simulator for $N_e$ times, each with $T$ timesteps, and compute the average objective and measures. We add the evaluated NCAs to both an optimization archive and a result archive. Finally, we update the parameters of the multivariate Gaussian distribution and start a new iteration. We run CMA-MAE until the total number of evaluations reaches $N_{eval}$.

**NCA.** Following previous work (Earle et al. 2022), we use a CNN with 3 convolutional layers of kernel size 3 × 3 and about 3,000 parameters as our NCA. Starting from a fixed initial environment, the NCA iteratively updates the environment for $C$ iterations, where $C$ is a hyperparameter.

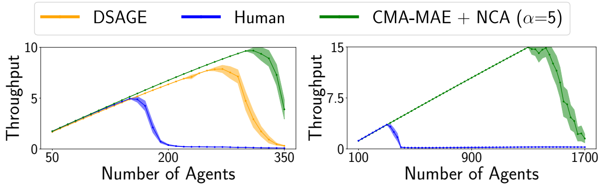
**MILP Repair.** We use the same MILP solver in the previous work (Zhang et al. 2023) to repair the invalid environments generated by NCA.

**Objectives.** Our objective function is $f_{opt} = f_{res} + \alpha \cdot \Delta$, where $f_{res}$ runs a lifelong MAPF simulator for $N_e$ times and returns the average throughput, $\Delta$ is the percentage of tiles that are the same in the unrepaired and repaired environments by MILP, and $\alpha$ is a hyperparameter. $\Delta$ is a regularization term to bias the search towards NCAs that are more inclined to directly generate valid environments. However, we eventually evaluate the NCAs using throughput. Therefore, we use $f_{opt}$ as the objective of the optimization archive and $f_{res}$ as the objective of the a separate result archive. We take the best NCA in the result archive for evaluation.

**Diversity Measures.** We use (1) the number of connected shelf components (following previous work (Zhang et al. 2023)) and (2) the *environment entropy* as the diversity measures. The environment entropy quantifies how much pattern the environment possesses. A lower value of environment entropy indicates a higher degree of pattern regularization. We use environment entropy as a diversity measure to find NCAs that can generate a broad spectrum of environments of varying patterns.

## 4 Experimental Evaluation

**Setup.** We train the NCAs with environments of size $S = 36 \times 33$ and then evaluate them in sizes of both $S$ and $S_{eval} = 101 \times 102$. For NCA, we set $C = 50$ for size $S$ and $C_{eval} = 200$ for size $S_{eval}$. For the lifelong MAPF simulator, we use RHCR (Li et al. 2021), a state-of-the-art centralized lifelong MAPF planner. We run each simulation with 200 agents for $T = 1{,}000$ timesteps during training and $T_{eval} = 5{,}000$ timesteps during evaluation and stop early in case of *congestion*, which happens if more than half of the agents take wait actions at the same timestep. For CMA-MAE, we set $N_e = 5$, $b = 50$, and $N_{eval} = 10{,}000$. We initialize the Gaussian distribution with mean 0 and standard deviation 0.2. For the objective, we use $\alpha = 5$ for it empirically yields the best result.



(a) $S = 36 \times 33$ (b) $S_{eval} = 101 \times 102$

Figure 3: Throughput with an increasing number of agents in environments of size $S$ and $S_{eval}$. The solid lines are the average throughput while the shaded area shows the 95% confidence interval.

**Result.** We compare our NCA-generated environments with those optimized by DSAGE (Zhang et al. 2023), the state-of-the-art environment optimization method, and those designed by human (Li et al. 2021). Figure 3 shows the throughput with different number of agents in environments of size $S$ and $S_{eval}$. Our NCA-generated environments have higher throughput than the baseline environments with an increasingly larger gap with more agents. In size $S_{eval}$, we scale RHCR from less than 300 agents to more than 1300. In addition, the environments both sizes have the same regularized patterns.

## Acknowledgments

This work used Bridge-2 at Pittsburgh Supercomputing Center (PSC) through allocation CIS220115 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, which is supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296.

## References

Earle, S.; Snider, J.; Fontaine, M. C.; Nikolaidis, S.; and Togelius, J. 2022. Illuminating Diverse Neural Cellular Automata for Level Generation. In *GECCO*, 68–76.

Fontaine, M.; and Nikolaidis, S. 2023. Covariance Matrix Adaptation MAP-Annealing. In *GECCO*, 456–465.

Li, J.; Tinka, A.; Kiesel, S.; Durham, J. W.; Kumar, T. K. S.; and Koenig, S. 2021. Lifelong Multi-Agent Path Finding in Large-Scale Warehouses. In *AAAI*, 11272–11281.

Yu, G.; and Wolf, M. 2023. Congestion prediction for large fleets of mobile robots. In *ICRA*, 7642–7649.

Zhang, Y.; Fontaine, M. C.; Bhatt, V.; Nikolaidis, S.; and Li, J. 2023. Multi-Robot Coordination and Layout Design for Automated Warehousing. In *IJCAI*, 5503–5511.