Reinforcement Learning for Structural Block Assembly

Rayan Gauderon, Tom Stanic, Andrej Kotevski, Alexandre Misrahi, Oskar Dabkowski,
Mouhamad Rawas, Leonardo Martella (Group: **BlockRL**)

Abstract

This work explores the application of reinforcement learning algorithms to solve complex block assembly tasks. We demonstrate how agents can learn to construct bridges, towers, and other structures by placing blocks strategically to reach target positions. Our work compares different RL approaches (DQN, PPO, Masked-REINFORCE and Masked-PPO) and shows that masking invalid actions significantly improves learning efficiency and task performance.

Introduction

Physical construction tasks present unique challenges for reinforcement learning:

- **Sparse rewards:** Success requires precise placement of multiple blocks
- **Physics constraints:** Structures must maintain stability while preventing collisions throughout assembly
- **Large action space:** Many possible block types, positions and orientations
- **Numerous invalid actions:** Most random actions lead to unstable and colliding structures

With the help of the Swiss Data Science Center (SDSC) and the Laboratory for Creative Computation (CRCL) at EPFL, we developed a block assembly environment based on rigid body physics that enables RL agents to learn strategic block placement for targeted construction tasks.

Environment Design

Our environment includes:

1. **Physics-based stability:** Using rigid body equilibrium (RBE) calculations
2. **Diverse block types:** Cubes, trapezoids, hexagons and wedges
3. **Target-based tasks:** Agents must place blocks to reach specific target points
4. **Collision detection:** Blocks cannot overlap with each other
5. **Reward structure:** Rewards based on proximity to target positions with penalties for invalid actions

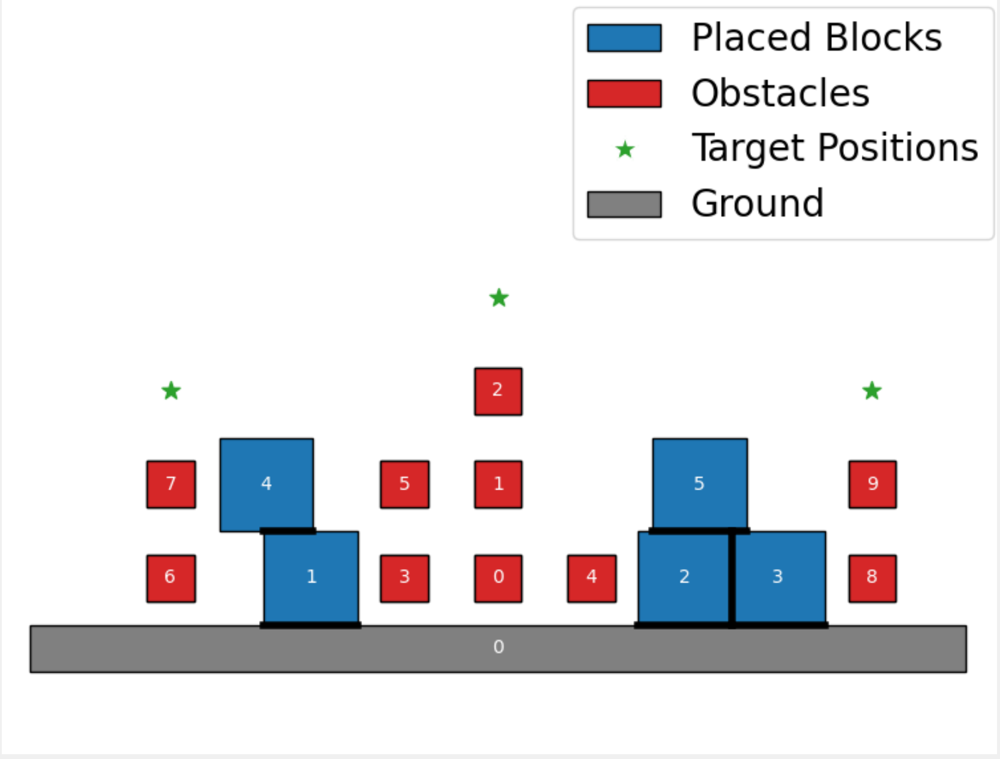


Figure 1. Experimental environment with target positions (green), placed blocks (blue) and obstacles (red).

Methodology and Implementation

Action Space Representation

- **Target block**: Which existing block to build upon
- **Target face**: Which face of the existing block to attach to
- **Shape selection**: Which block shape to place
- **Face selection**: Which face of the new block to attach
- **Offset**: Horizontal position adjustment

Reward Representation

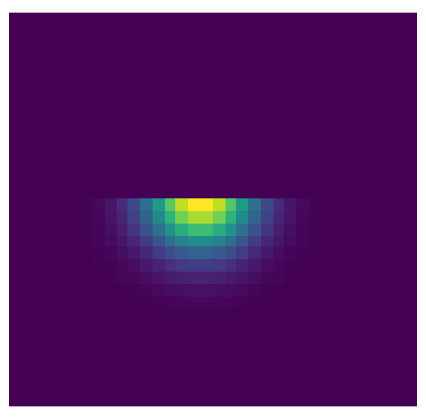
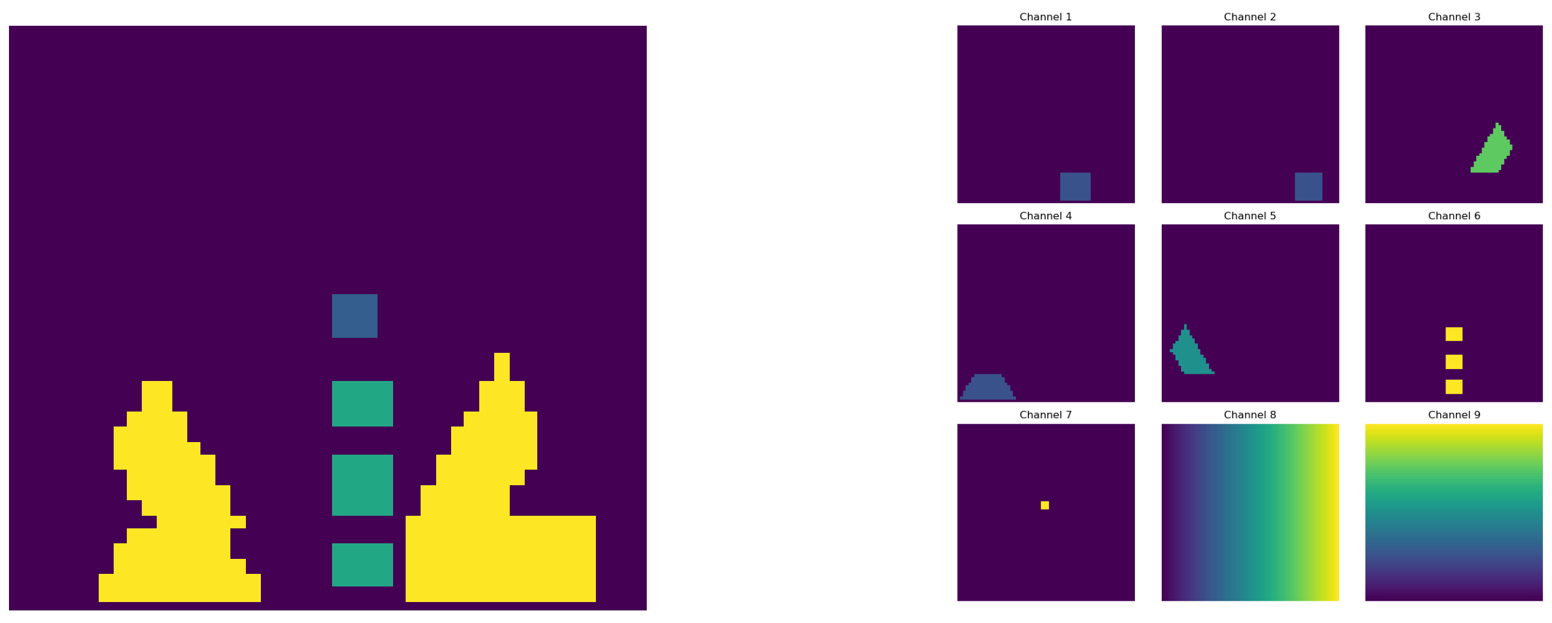


Figure 2. Gaussian reward representation centered at target position and cropped at target height.

State Representation



(a) (b)

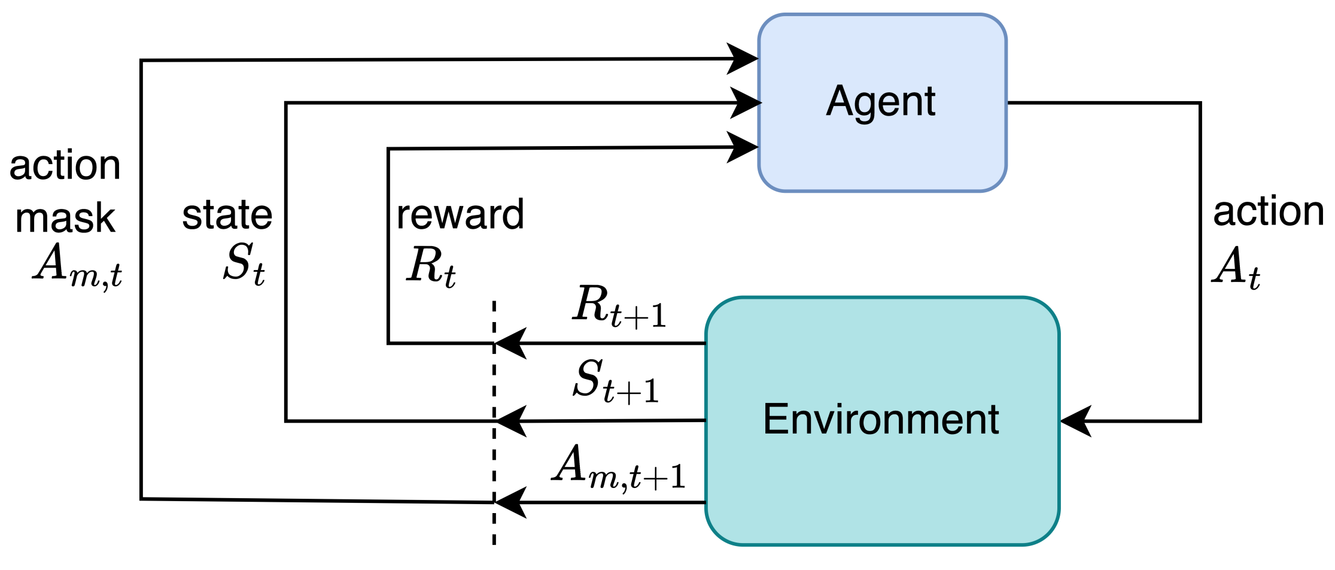
Figure 3. State representation visualization: (a) single-channel encoding of environment elements; (b) multi-channel encoding with separate channels for blocks, obstacles, targets, and positions.

Definitions:

- **Unavailable actions:** those that our action-enumeration procedure never lists (e.g. positions or attachments that aren't geometrically possible in the current state).
- **Invalid actions:** actions that are enumerated but, if executed, would violate stability or collision constraints (e.g. overlapping blocks or tipping the structure).

How Action Masking Works

1. **Generate available actions:** Enumerate all possible block placements from the **current state**
2. **Check validity:** For each action, verify if it would result in collisions or instability
Note: Due to computational constraints, we omit the physics-based validity checks in favor of a more efficient approach. Instead, we use the available actions and allow the agent to learn through experience which of these actions result in stable structures.
3. **Create binary mask:** Generate a mask where valid actions=1, invalid actions=0
4. **Apply to policy:** Use the mask to zero out probabilities of invalid actions before sampling



Agent

Environment

action mask $A_{m,t}$

state S_t

reward R_t

action A_t

R_{t+1}

S_{t+1}

$A_{m,t+1}$

Figure 4. Reinforcement learning framework with action masking. The agent selects actions based on current policy, which are executed in the environment. The environment returns the next state, reward, and an action mask identifying valid actions, improving exploration efficiency and learning performance.

Acknowledgements: This research was supported by the Swiss Data Science Center (SDSC) and the Laboratory for Creative Computation (CRCL) at EPFL. We thank our advisor Paul Rolland for his expertise throughout this project.

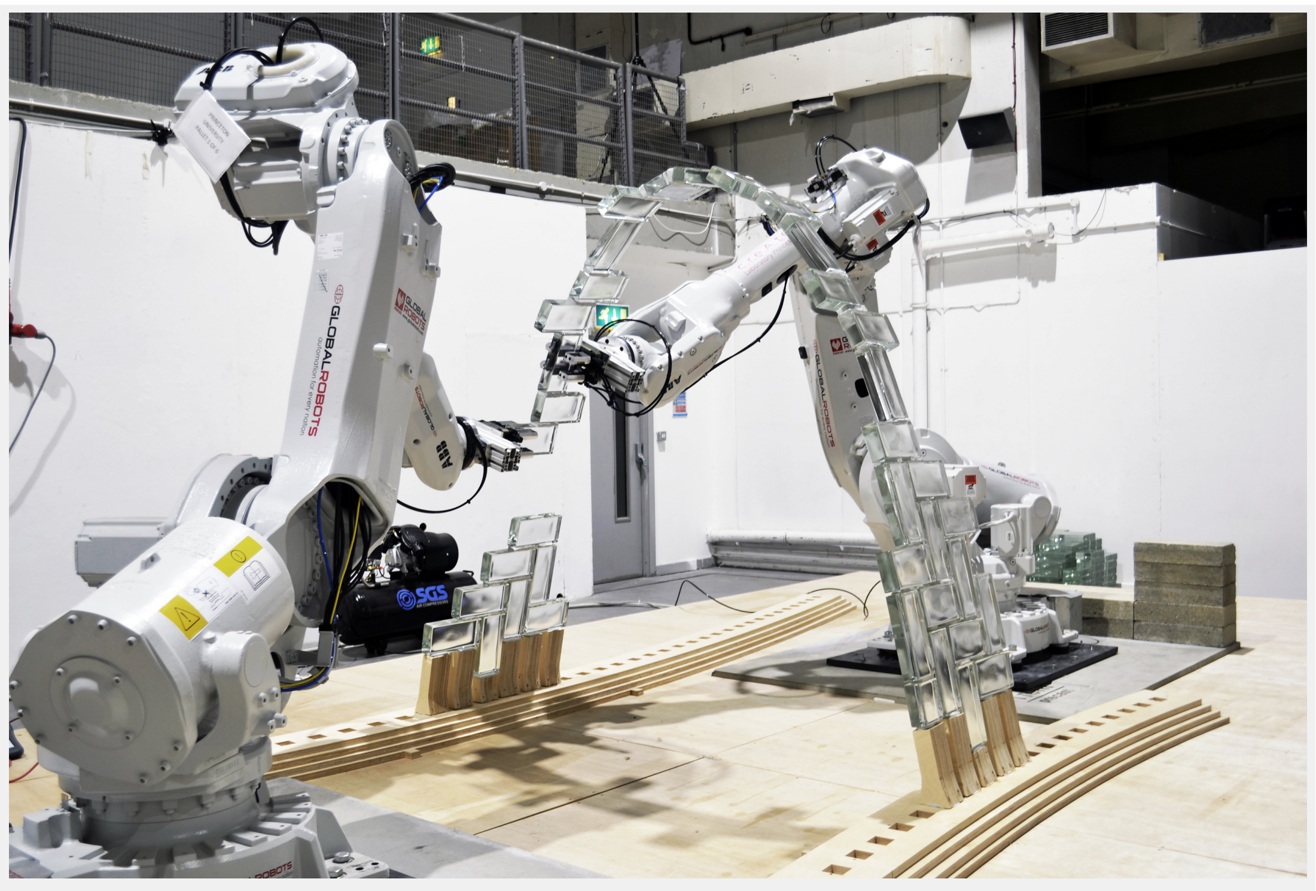


Figure 5. Physical implementation objective: A robotic system constructing a tower structure. While our current work focuses on 2D simulation environments, this represents the ultimate goal of transferring learned policies to real-world robotic assembly tasks. *Image credit: Stefana Parascho, CRCL lab, EPFL.*

Experimental Results

Tested Algorithms

Algorithm	Approach	Key Features	Limitations/Improvements
DQN	Value-based approach with experience replay	Used discrete action space with all possible combinations	Many invalid actions led to inefficient exploration
PPO	Policy gradient method with clipped objective	More stable learning than traditional policy gradients	Still struggled with large invalid action space
Masked-REINFORCE	Policy gradient method with action masking	Direct policy updates and invalid action elimination	Despite masking, marginal efficiency improvements
Masked-PPO	Extended PPO algorithm with action masking	Dynamically eliminated invalid actions	Significantly improved learning efficiency

Table 1. Comparison of Reinforcement Learning Algorithms used for the Block Assembly Environment

Learning Efficiency

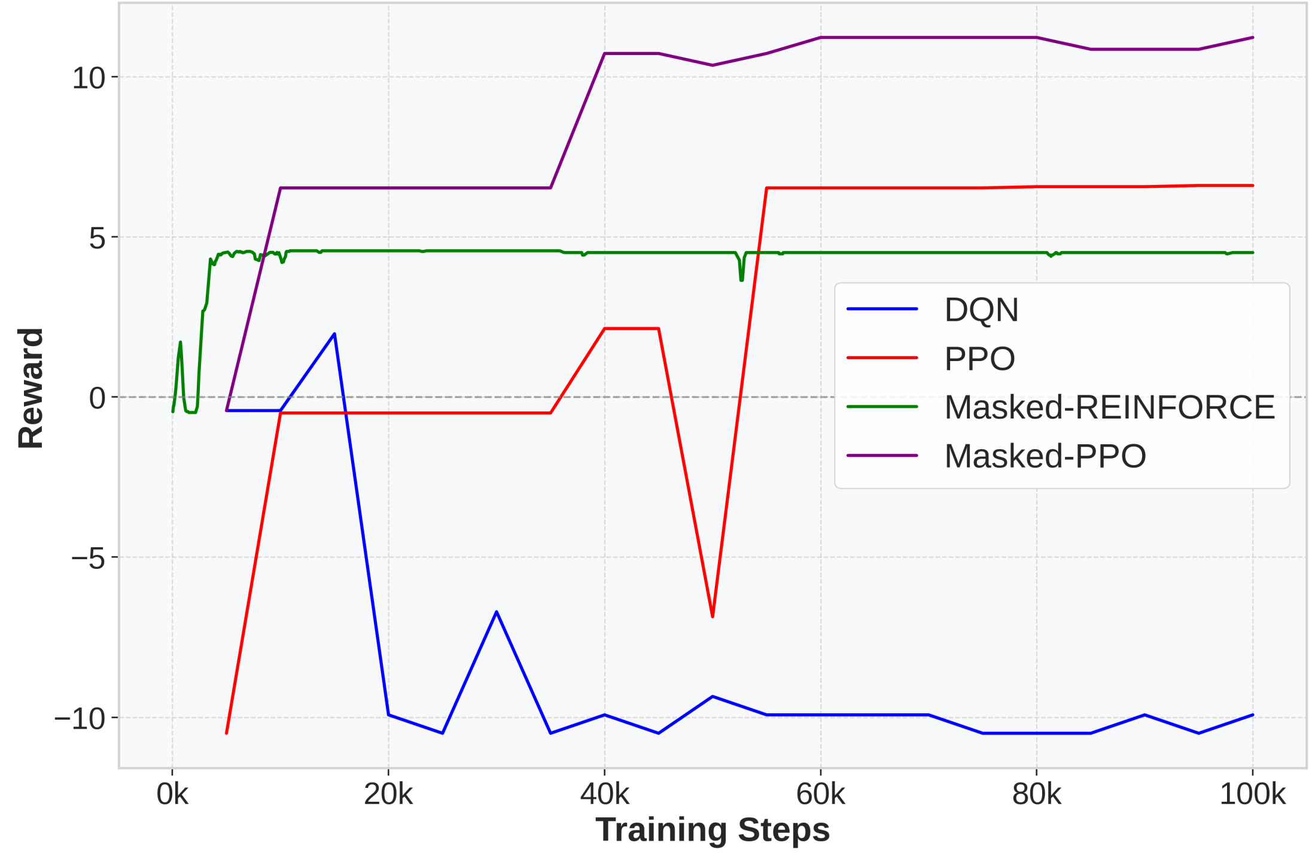


Figure 6. Comparison of learning efficiency measured by reward values across training steps

Algorithm Performance Comparison

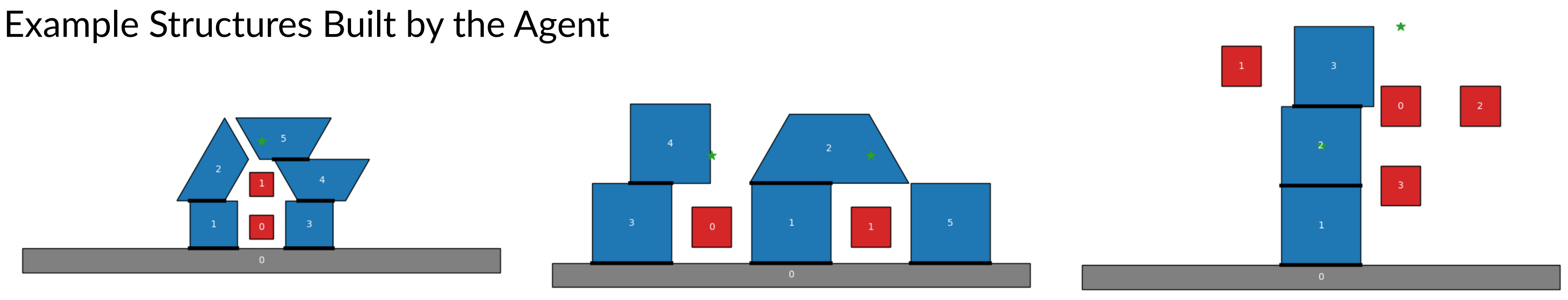
Algorithm	Unavailable action rate	Invalid action rate	Reward collected
DQN	88.9%	63.5%	2.0
PPO	27.3%	43.0%	6.6
Masked-REINFORCE	$\mathbf{0}\%$	41.3%	4.6
Masked-PPO	$\mathbf{0}\%$	$\mathbf{24.7}\%$	$\mathbf{11.2}$

Table 2. Performance metrics: lower unavailable and invalid action rates indicate better constraint satisfaction, while higher reward values represent superior task completion.

Key Findings

- Masked-PPO achieved reward values at least **2x higher** than competing algorithms
- Action masking reduced the effective action space by $\approx$ **85%**
- Masking was crucial for the agent to learn complex structures

Example Structures Built by the Agent



(a) Bridge (b) Double Bridge (c) Tower

Figure 7. Structures built using masked-PPO agent for different environments.

Conclusions and Future Work

Conclusions

- **Action masking** is crucial for efficient learning in construction tasks with many invalid actions.
- **Masked-PPO** outperforms standard algorithms in both learning speed and final performance
- The approach shows promise for broader applications in **robotic assembly** and **architectural design**

Future Work

- Extend to 3D assembly tasks with additional constraints
- Adding uncertainty in block placement for more realistic robot applications
- Introducing regularization through moving targets.

References

Gene Ting-Chun Kao, Antonino Iannuzzo, Bernhard Thomaszewski, Stelian Coros, Tom Van Mele, and Philippe Block. Coupled rigid-block analysis: Stability-aware design of complex discrete-element assemblies. *Computer-Aided Design*, 2022.

Cheng-Yen Tang, Chien-Hung Liu, Woei-Kae Chen, and Shingchern D. You. Implementing action mask in proximal policy optimization (ppo) algorithm. *ICT Express*, pages 200–203, 2020.