Deutsches Forschungszentrum für Künstliche Intelligenz
German Research Center for Artificial Intelligence

Neuro-Mechanistic Modeling

NEURO-EXPLICIT SEQUENTIAL DECISION-MAKING NEXD

SIC

UNIVERSITÄT DES SAARLANDES

PGTG: Procedurally-generated Grid-based Traffic Gym

Joshua Meyer, Felix M. Kuntz, Verena Wolf, Jörg Hoffmann, Timo P. Gros

A **new benchmark** for

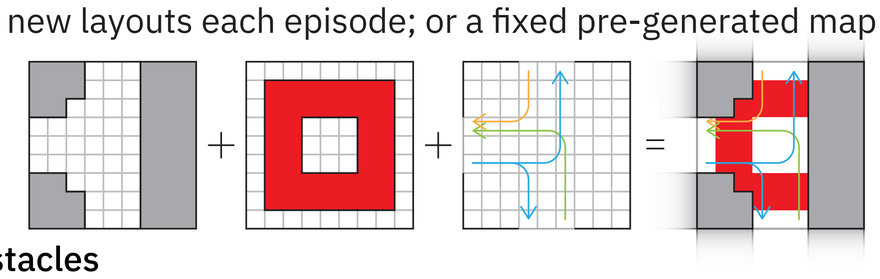
▷ **Exploration**
- long tracks entail *sparse rewards*
- stochastic traffic presents *fake novelty*
- *partial observation* and procedural generation challenge *inference-time exploration* and memory

▷ **Safety**
- stochastic but *avoidable obstacles*
- stochastic but *predictable traffic* with behavior profiles
- decoupling safety from exploration through *subgoal hints*

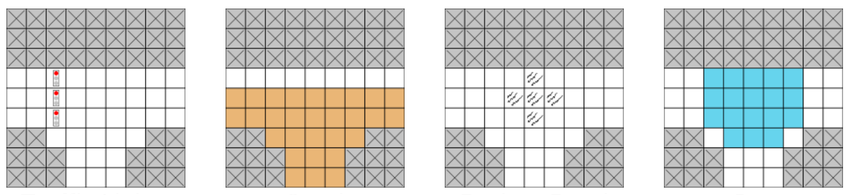
▷ **Generalization**
- procedural generation and customizability enable easy *out-of-distribution evaluations* (map size, generation parameters)
- human-readable visualizer that helps understand failure modes

Features

▷ **Procedural Generation**
- new layouts each episode; or a fixed pre-generated map

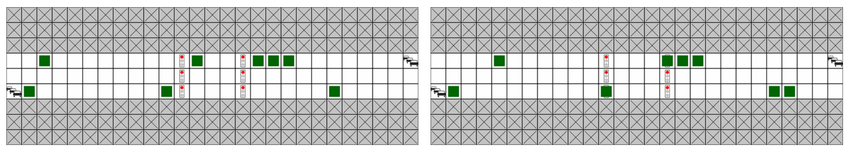


▷ **Obstacles**
- stochastic, dynamics-changing, but avoidable, obstacles



Traffic Lights | Sand | Broken Road | Ice

▷ **Traffic**
- other drivers, with different profiles, that have to be avoided



Rule-conforming Drivers | Aggressive Drivers

▷ **Traffic Rules**
- enforce dynamics and rule-conforming agent behavior, depending on traffic conditions

```
{
  "name": "four_way_intersection_brake",
  "tile_type": [1, 1, 1, 1],  // N/E/S/W: 1=connected, 0=not connected
  "velocity_range": [0.5, 10.0],
  "min_traffic": 1,  // Minimum traffic in tile
  "min_matching_traffic": 1,  // Minimum traffic in tile matching the directions
  "maneuvers": [  // Agent and traffic directions to match the preconditions
    { "agent": "west_to_east", "traffic": ["north_to_south", "south_to_north"] },
    // [...] Other agent / traffic direction matches
  ],  // The action can of course also be a named python function
  "action": "lambda env, ctx: (setattr(env, 'velocity', np.array((0, 0))), True)[-1]"
},
```

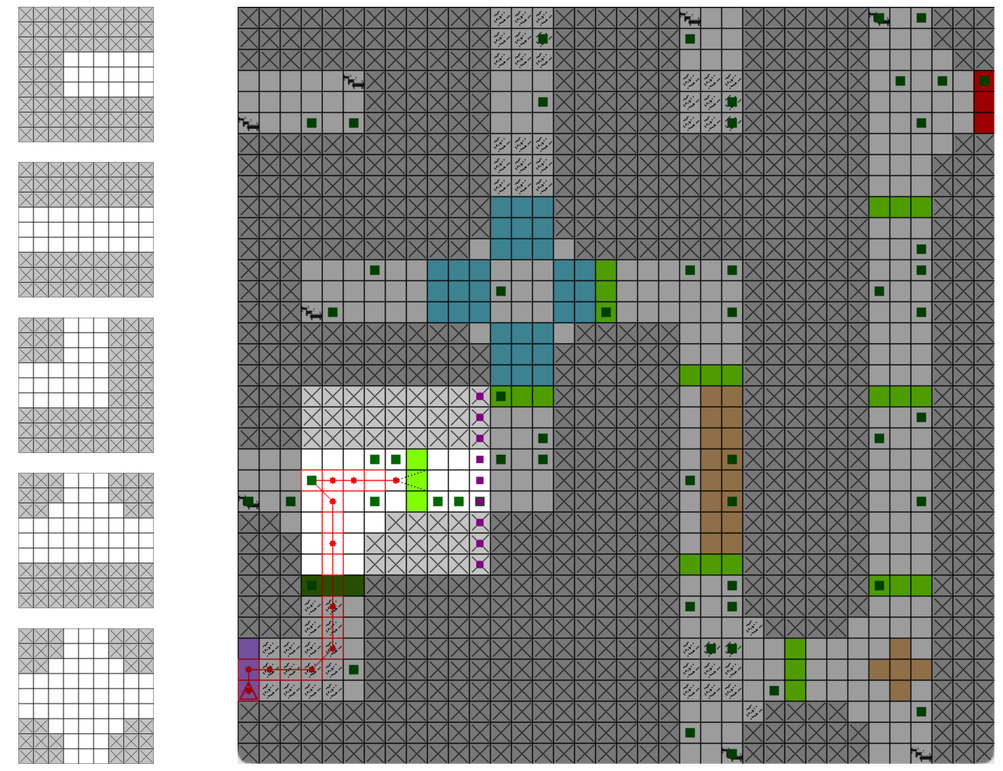
Design Philosophy

▷ **Customizability**
- *customize* track generation, obstacle, and driver probabilities, observation and reward functions *without touching internals*

▷ **Extensibility**
- add *your own obstacles*, behaviors, rules, and tile sets/masks!

```
@pgtg_obstacle("ice", act_probability=0.1, obstacle_weight=1.0, observe=True)
def ice_behavior(env: PGTGEnv, local_ctx: StepContext)-> bool:
    ice_action = ice_behavior.rng.choice(len(ACTIONS_TO_ACCELERATION))
    local_ctx.velocity_substep = ACTIONS_TO_ACCELERATION[ice_action]
    return False
```

▷ **Reproducibility**
- seeded episodes, obstacles, and restorable states

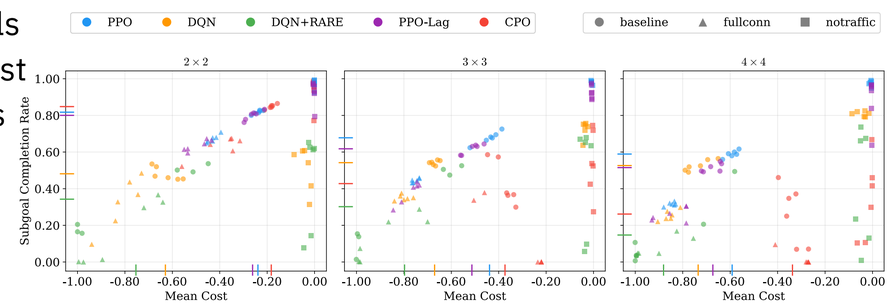


Default Tileset

Empirical Results

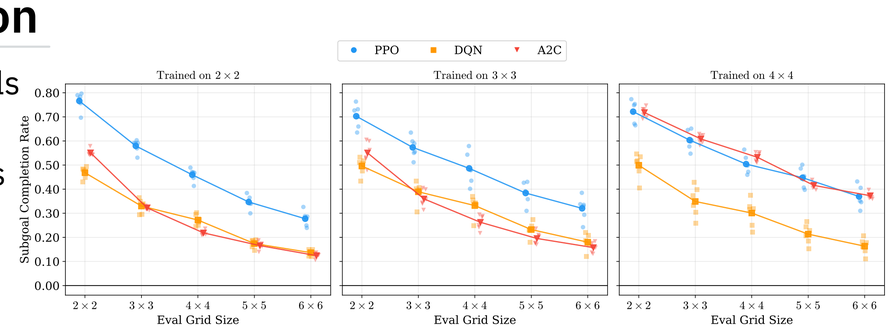
Safety

▷ reward subgoals
▷ split reward/cost
▷ avoid obstacles



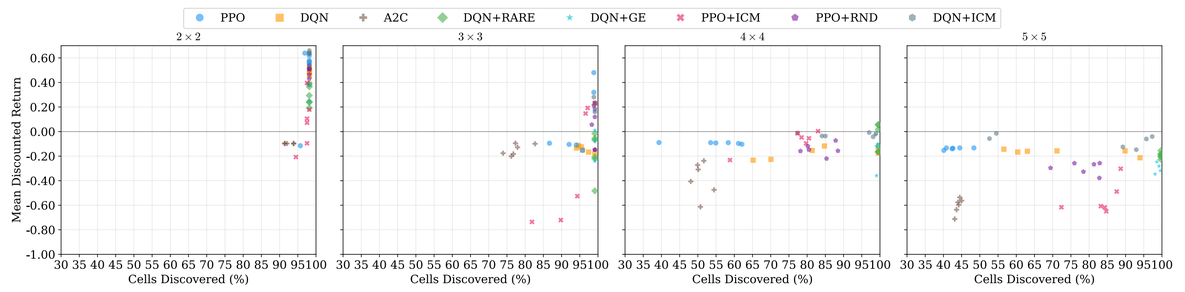
Generalization

▷ reward subgoals
▷ cost as penalty
▷ avoid obstacles
▷ evaluate out-of-distr.



Exploration

State-of-the-art methods tested on
▷ Fixed maps (different sizes)
▷ without obstacles and subgoals,
▷ but busy roads providing fake novelty.



Future Plans

▷ Visual map/tile-Editor (WIP)
▷ Provide results on tuned state-of-the-art methods
▷ Additional functionality: diagonal connections, continuous spaces, continual learning, ...

Try it out today: `pip install pgtg`
https://github.com/neuro-mechanistic-modeling/pgtg

PyPI: pypi.org/p/pgtg
Paper: nexd.group/pgtg
Github: /neuro-mechanistic-modeling/pgtg

CERTAIN Center for European Research in Trusted Artificial Intelligence
Ministerium für Wirtschaft, Innovation, Digitales und Energie SAARLAND
CPEC CENTER FOR PERSPICUOUS COMPUTING
Co-funded by the European Union
DFG Deutsche Forschungsgemeinschaft
RTG 2853 Neuroexplicit Models of Language, Vision, and Action
Bundesministerium für Bildung und Forschung

web: https://nexd.group/joshua
mail: joshua.meyer@dfki.de