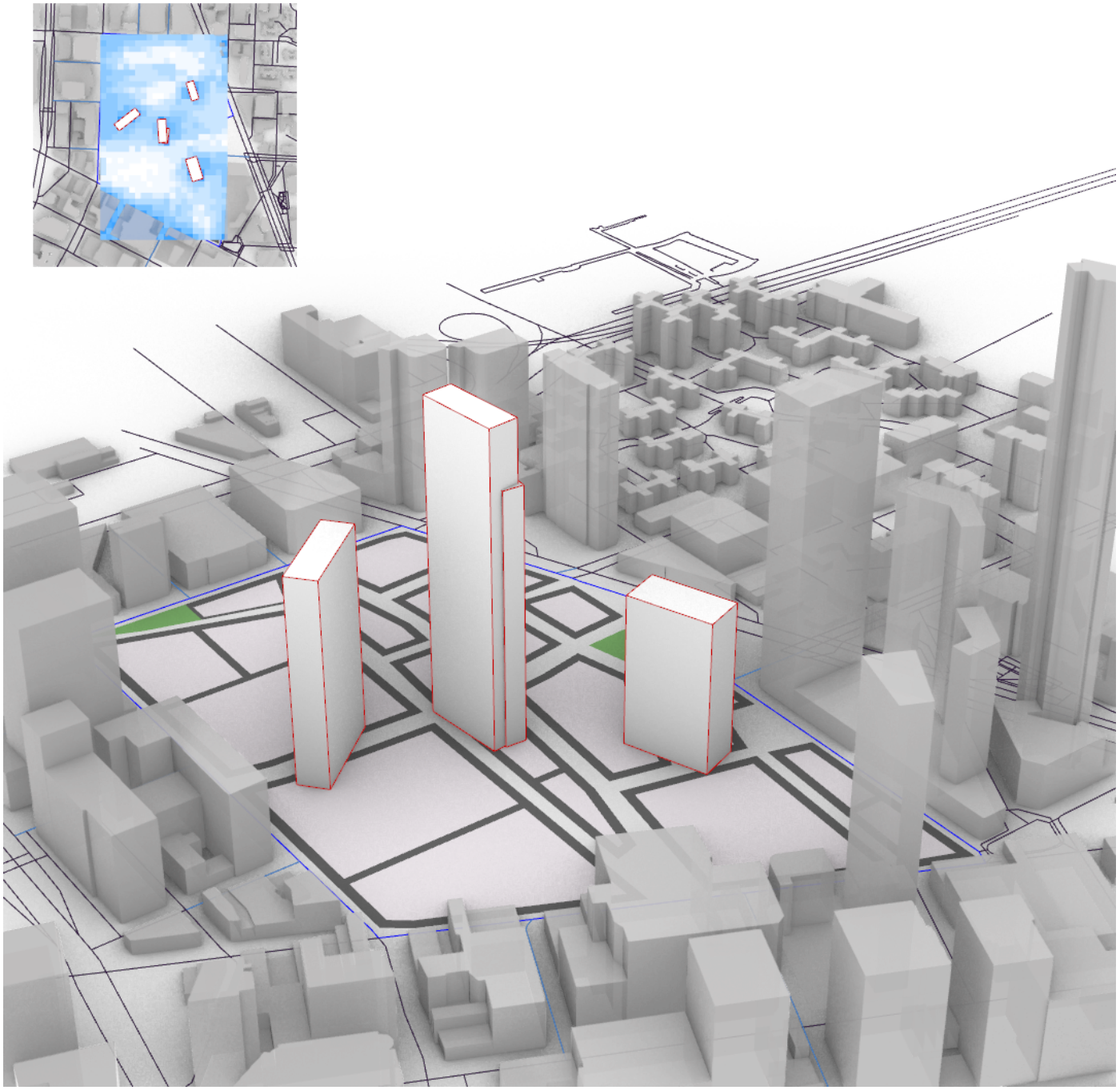
- Brooklyn_DeepRL
  - Action space
  - Reward function
    - Terminal state condition
  - Training
  - References

Brooklyn_DeepRL

Urban block renewal maximizing outdoor thermal comfort using deep reinforcement learning methods.



Action space

The agent controls the building location through x, y coordinates, the building orientation angle theta, as well as the length, width, and height of the building. This is done through a 6 dimensional discrete action space.

```
import numpy as np
import torch

# x coordinate
param1_space = torch.from_numpy(np.linspace(start=456, stop=835, num=250))
# y coordinate
param2_space = torch.from_numpy(np.linspace(start=325, stop=886, num=250))
# building orientation angle theta
param3_space = torch.from_numpy(np.linspace(start=0, stop=180, num=250))
# building length
param4_space = torch.from_numpy(np.linspace(start=20, stop=70, num=250))
# building width
param5_space = torch.from_numpy(np.linspace(start=20, stop=70, num=250))
# building height
param6_space = torch.from_numpy(np.linspace(start=10, stop=200, num=250))
```

Reward function

The agent starts with a +10 reward at each step and receives a penalty of -10 if the placed building happens to be outside the superblock boundary, plus a -5 penalty if it is not inside any of the buildable plots. The agent is also penalized by -2 if one of the buildings placed is inside one of the existing buildings, and -1 for each intersection between buildings. The reward signal is computed at each step in the Grasshopper environment according to the following code:

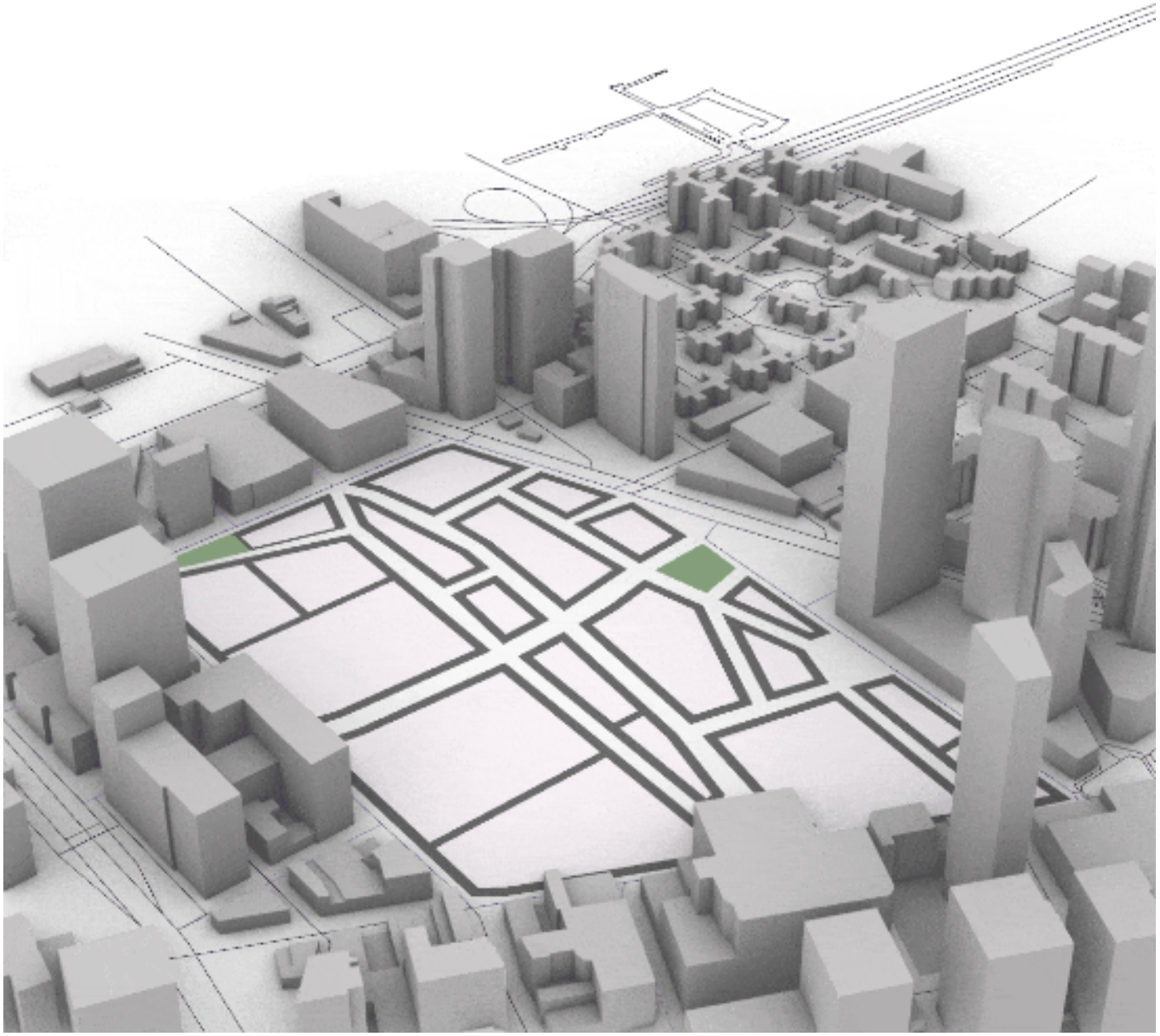
```
try:
    from ladybug_rhino.grasshopper import all_required_inputs
except ImportError as e:
    raise ImportError('\nFailed to import ladybug_rhino:\n\t{}'.format(e))

if all_required_inputs(ghenv.Component):
    reward = 0
    done = False

    boundary_relationList = [list(i) for i in boundary_relation.Branches]
    bArea_relationList = [list(i) for i in bArea_relation.Branches]
    bInter_relationList = [list(i) for i in bInter_relation.Branches]
```

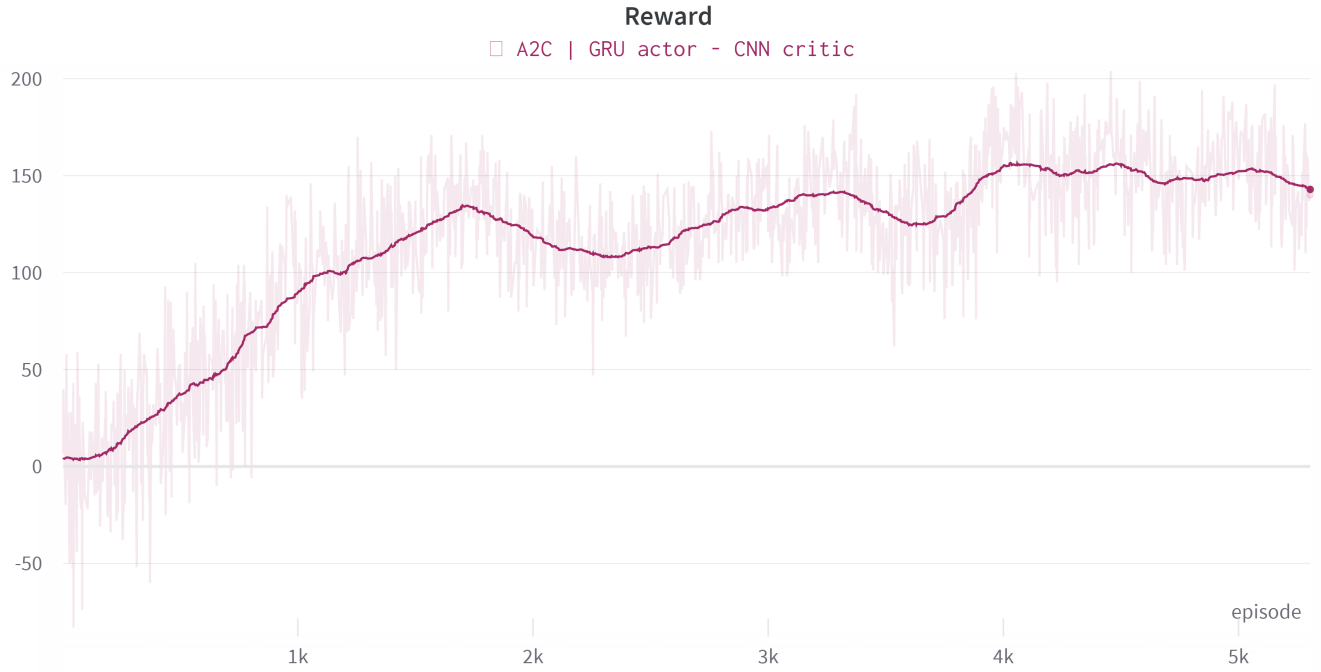
```
        reward += 10
        for r in boundary_relationList:
            if r[0] != 0:
                reward -= 10
        for b in bArea_relationList:
            if 0 in b:
                reward += 2
            else:
                reward -= 5
        for i in bInter_relationList:
            for r in i:
                if r == 0:
                    reward -= 2
                elif r == 1:
                    reward -= 1
            reward += 1
        for gen_area in gen_env_areas:
            if env_area_start < gen_area < env_area_end:
                reward += 1
            else:
                reward -= 1

        if builtarea_ratio >= 0.9:
            done = True

        if len(bldg_bool) == 0:
            done = False
```

Terminal state condition

The terminal state is reached after 25 buildings are placed or if the built area ratio reaches 90%.

Training




Reward
A2C | GRU actor - CNN critic
200
150
100
50
0
-50
episode
1k
2k
3k
4k
5k

References

- Asynchronous Methods for Deep Reinforcement Learning - Mnih et al. (2016).
- Efficient Entropy for Policy Gradient with Multidimensional Action Space - Zhang et al. (2018).