Generative Adversarial Network based heuristics for sampling based path planning

Project description

Azamat Kanametov   Alina Kolesnikova   Timofey Zinenko

April 15, 2021

1 Optimal path planning problem

Non-formal Determine a collision–free path from a start point to a goal point while optimizing a performance criterion such as distance, time or energy

Formal

Given:

- $\mathcal{X} \in \mathbb{R}^n$ – the state space, $n \in \mathbb{N}^n$, $n \geqslant 2$
- $\mathcal{X}_{obs}$ – obstacle space, $\mathcal{X}_{free} = \mathcal{X} \setminus \mathcal{X}_{obs}$ – free space
- $x_{init} \in \mathcal{X}_{free}$ – the initial state, $x_{goal} \in \mathcal{X}_{free}$ – the goal state
- $\mathcal{X}_{goal} = \left\{ x \in \mathcal{X}_{obs} \middle| \, \|x - x_{goal}\| < r \right\}$ – the goal region
- Σ – the set of all feasible paths
- $c(\sigma)$ – the cost function, $\sigma \in \Sigma$,

$$Cost(x_i, x_j) = \|x_i - x_j\|, \quad x_i, x_j \in \mathcal{X}_{free}$$

Find: feasible path $\sigma^* : [0, 1] \to \mathcal{X}_{free}$

$$\sigma^* = \arg\min_{\sigma \in \Sigma} c(\sigma), \quad s.t.\ \sigma(0) = x_{init}, \sigma(1) \in \mathcal{X}_{goal}$$

2 Approach

2.1 Background

- Sampling–based algorithms solve path planning problems through constructing space–filling trees to search a path σ.
- The tree is built incrementally with samples drawn randomly from the free space $\mathcal{X}_{free}$
- Drawbacks: the quality of initial solution, the convergence speed

2.2 Idea

- Use generative adversarial network (GAN) to learn promising regions and construct heuristic non–uniform sampling distribution $\mathcal{X}_H \subset \mathcal{X}_{free}$ to reduce sampling space
- Use this heuristic in sampling–base algorithm (e.g., RRT*)

2.3 Algorithm

Algorithm 1: Outline of GAN–based heuristic RRT*

Input : x_{init}, x_{goal}, Map – state space in the form of RGB image
Output: $G(V, E)$

$V = x_{init}$, $E = \varnothing$;
$\mathcal{S} \leftarrow$ ROIGenerator(x_{init}, x_{goal}, Map) ;
$\mathcal{X}_H \leftarrow$ Discretization($\mathcal{S}$);
$G(V, E) \leftarrow$ HeuristicSBP*(x_{init}, x_{goal}, Map, $\mathcal{H}$);
Return $G(V, E)$

Here $\mathcal{X}_H \subset \mathcal{X}_{free}$ is the state space where feasible paths exist with high probability.

The focus of work lies in establishing an efficient generator to predict promising region $\mathcal{S}$ under the given conditions x_{init}, x_{goal}, Map.

Algorithm 2: Comparison of RRT* and Heuristic RRT*

Input : x_{init}, x_{goal}, $\mathcal{H}$, Map, UseHeuristic
Output: $G(V, E)$

$V = x_{init}$, $E = \varnothing$;
for $i = 1 \cdots N$ **do**
    if $UseHeuristic = True$ **then**
        if $Rand() > \mu$ **then**
            $x_{rand} \leftarrow$ Non-UniformSample($\mathcal{X}_H$);
        else
            $x_{rand} \leftarrow$ UniformSample();
        end
    else
        $x_{rand} \leftarrow$ UniformSample();
    end
end
$x_{nearest} \leftarrow$ Nearest(G, x_{rand});
$x_{new} \leftarrow$ Steer($x_{nearest}$, x_{rand});
if $ObstacleFree(x_{nearest}, x_{rand})$ **then**
    Extend(G, x_{new});
    Rewire();
    if $x_{new} \in \mathcal{X}_{goal}$ **then**
        Return $G(V, E)$;
    end
end
Return Failure;

Line 3 to Line 8 is adopted by heuristic RRT* and Line 9 to Line 11 is in RRT*. Here $\mu\%$ samples are randomly chosen from the non–uniform sampling distribution, while $1 - \mu\%$ are sampled from the uniform sampling distribution to guarantee probabilistic completeness. We denote $\mathcal{X}_H$ as heuristic discrete points extracted from the sampling distribution.

3 GAN–based promising region generation

Trained with large amount of empirical promising region data, the GAN model is able to generate promising regions for non–uniform sampling under specified conditions. The input of the model is an RGB image representing the state space Map, start state x_{init}, and goal state x_{goal}. The output of the model is also an RGB image where the promising regions are highlighted.

3.1 Dataset generation

Algorithm 3: Dataset generation

output: Maps, points, promising regions

Generate randomly environment map, 64×64
    Color $\mathcal{X}_{obs}$ into black
    Color $\mathcal{X}_{free}$ into white
Randomly choose from 20 to 50 x_{init} and x_{goal} states from $\mathcal{X}_{free}$
    Color x_{init} into red
    Color x_{goal} into blue
Generate promising regions as ground truth
    Randomly select x_{init}, x_{goal} and run RRT 50 times

3.2 GAN training and evaluation

Algorithm 4: GAN training and evaluation

InputGen : $\mathcal{P}_{gt}$ – point images (with x_{init} and x_{goal}), $\mathcal{M}_{gt}$ – map images, $\mathcal{Z}$ – noise data
InputDiscr : $\mathcal{S}_{gt}$ – ground truths Region of Interest
OutputGen : $\mathcal{S}_p$ – generated Region of Interest
OutputDiscr: *True* or *Fake*

Training;
while *Generator does not converge* **do**
    $\mathcal{S}_p \leftarrow$ Generator($\mathcal{P}_{gt}$, $\mathcal{M}_{gt}$, $\mathcal{Z}$) ;
    $\mathcal{P}_p \leftarrow$ PointDiscriminator($\mathcal{S}_p$, $\mathcal{P}_{gt}$) ;
    $\mathcal{M}_p \leftarrow$ MapDiscriminator($\mathcal{S}_p$, $\mathcal{M}_{gt}$) ;
    obtain losses $\mathcal{L}_{D_{point}}(\mathcal{P}_p, \mathcal{P}_{gt})$, $\mathcal{L}_{D_{map}}(\mathcal{M}_p, \mathcal{M}_{gt})$, $\mathcal{L}_G(\mathcal{S}_p, \mathcal{S}_{gt})$;
    backprop
end
Evaluation;
$\mathcal{S}_p \leftarrow$ Generator($\mathcal{P}_{gt}$, $\mathcal{M}_{gt}$) ;
Return $\mathcal{S}_p$

4 Experiment results

TODO