MANCHESTER 1824
The University of Manchester

Assisted Counterexample-Guided Inductive Optimization for Robot Path Planning

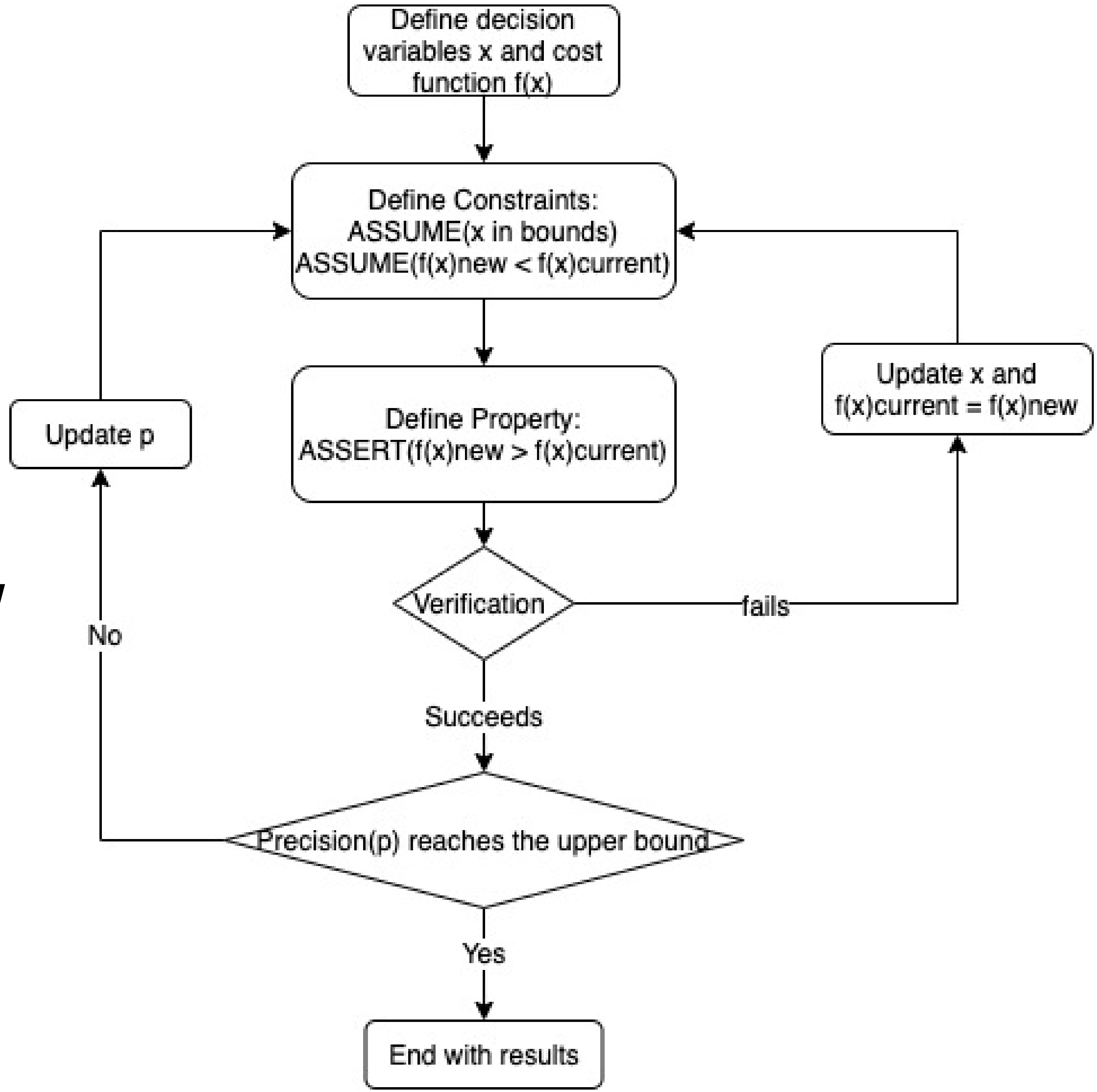
Mengze Li and Lucas Cordeiro

Synopsis

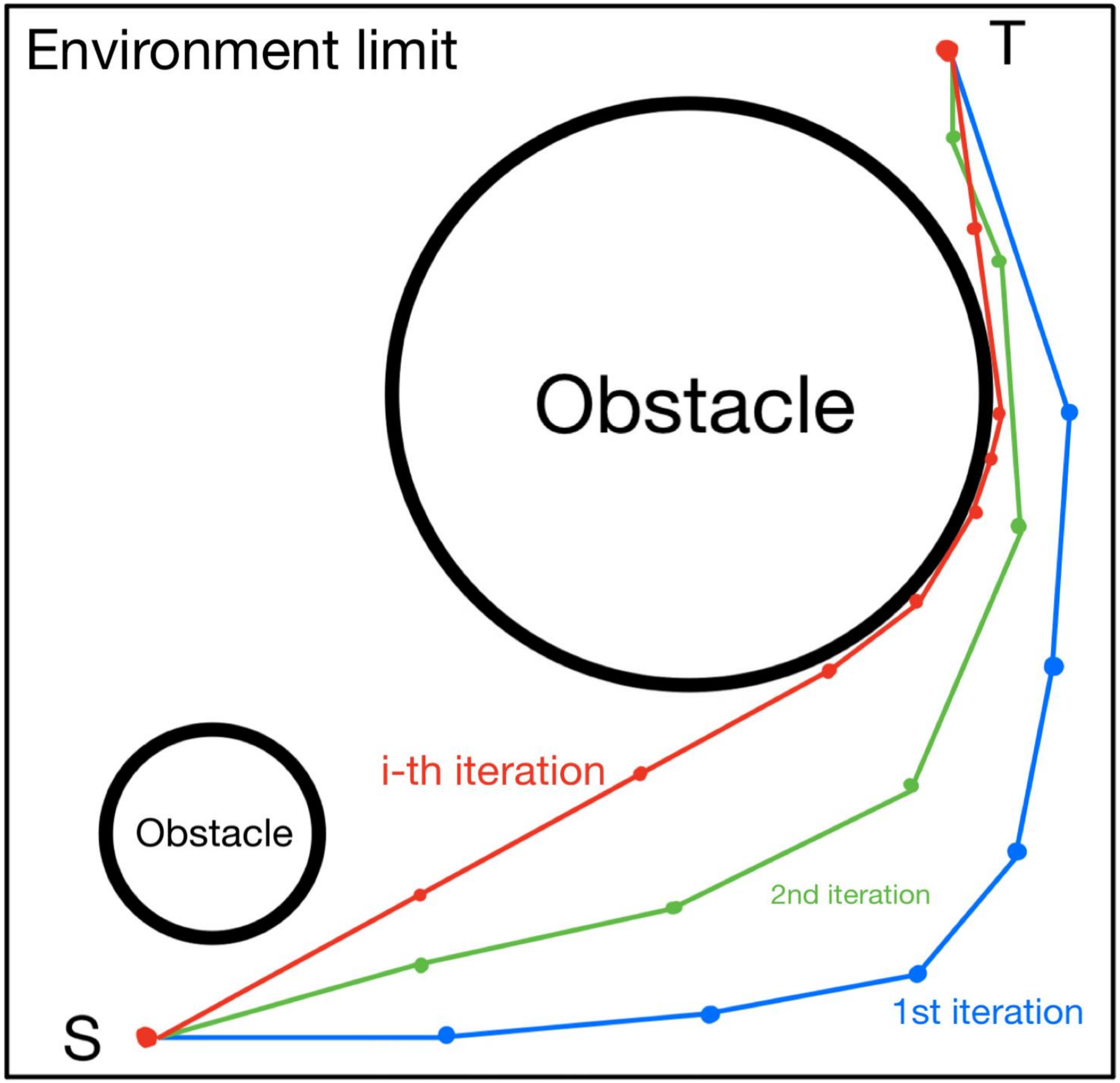
- Background knowledge
- Objectives
- ACEGIO-based path planning algorithm
- Experimental Evaluation
- Conclusion and Future work

The University of Manchester

CEGIO work flow



Background knowledge



- Path planning can be considered as an optimization problem
- CEGIO can be applied to achieve the optimal path by iteratively requesting counterexamples from SAT/SMT

MANCHESTER 1824
The University of Manchester

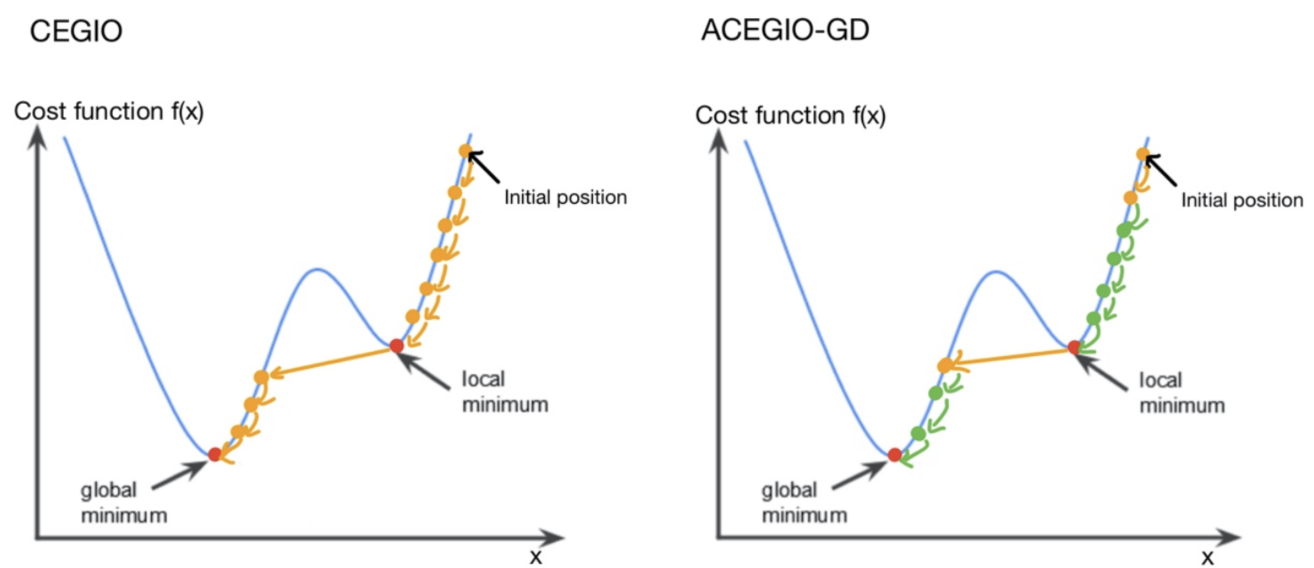
Background knowledge

- However, requesting counterexamples from SAT/SMT solvers is the most time-consuming process
- ACEGIO combines CEGIO with an auxiliary algorithm (Gradient Descent is selected in this work)
- ACEGIO relies on CEGIO to preserve the optimization ability and relies on the selected auxiliary algorithm to improve the efficiency

The University of Manchester

Illustrative Example

The optimization process of employing ACEGIO-GD contains fewer times of requesting counterexamples from SAT/SMT solvers, and Gradient Descent (much faster) is applied instead.



- The red dot is global minimum or local minimum
- The orange dot is position updated by requesting counterexamples from SMT solvers
- The green dot is position updated by Gradient Descent
- The orange arrow is the optimizing path generated by requesting counterexamples
- The green arrow is the optimizing path generated by Gradient Descent

MANCHESTER 1824
The University of Manchester

Objectives

The main objective of this work is to propose and evaluate a novel offline mobile robot path planning algorithm based on Assisted Counterexample-Guided Inductive Optimization

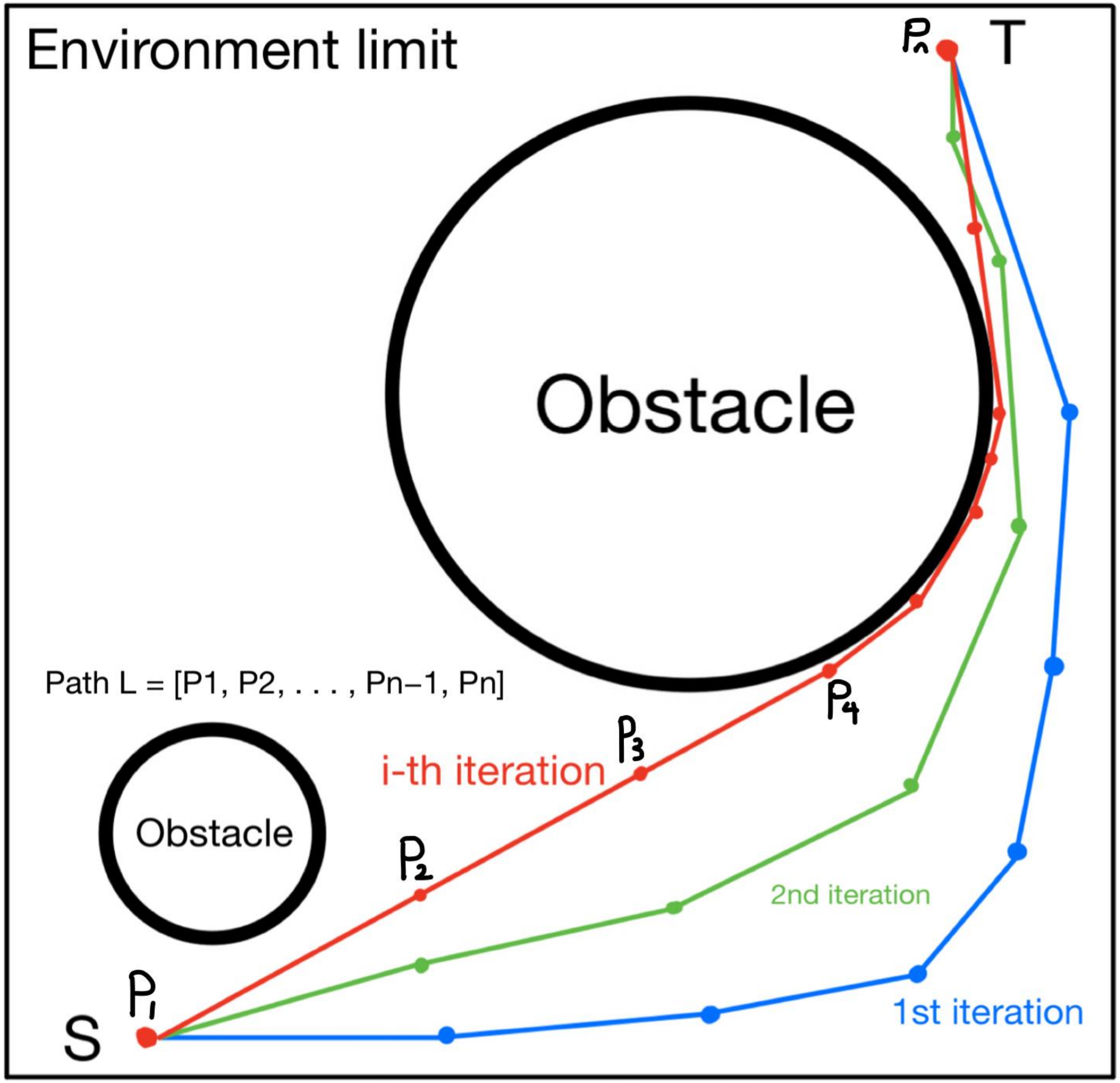
- Develop ACEGIO algorithm with Gradient Descent as the auxiliary algorithm to generate optimal paths
- Evaluate the proposed ACEGIO-based path planning algorithm
- Compare the results with other traditional optimization techniques based path planning algorithms

MANCHESTER 1824
The University of Manchester

Two steps of applying ACEGIO to path planning problems

1. Formulate the path planning problem as an optimization problem (define decision variables, cost function and set constraints)
2. Apply ACEGIO-GD to find the optimal path that minimizes the cost function and satisfies the constraints

Path planning problem formulation



- Starting position S(P1) and target position T(Pn)
- Points consisted the path L = [P1, P2, . . . , Pn−1, Pn]

Path planning problem formulation

- The cost function: $J(\mathrm{L}) = \sum_{\mathrm{i}=1}^{\mathrm{n}-1} \|\mathrm{P}_{\mathrm{i}+1} - \mathrm{P}_{\mathrm{i}}\|_2$, n is the number of points on the desired path
- Obstacles $\mathbb{O}$ and environments limits $\mathbb{E}$
- Optimization problem for path planning can be written as

$$\begin{aligned} \min_L \quad & \mathrm{J}(\mathrm{L}) \\ \text{s.t.} \quad & \mathrm{p}_{\mathrm{i}\lambda}(\mathrm{L}) \notin \mathbb{O} \\ & \mathrm{p}_{\mathrm{i}\lambda}(\mathrm{L}) \in \mathbb{E} \\ & \mathrm{i} = 1, \ldots, \mathrm{n}-1 \end{aligned}$$

MANCHESTER 1824

The University of Manchester

CEGIO-based path planning algorithm

input : Cost function $J(\mathbf{L})$, is a set of obstacles constraints $\mathbb{O}$ and a set of environment constraints $\mathbb{E}$, which define Ω and a desired precision η

output: The optimal path $\mathbf{L}^*$ and the optimal cost function value $J(\mathbf{L}^*)$

Initialize $J(\boldsymbol{L}^{(0)})$ *randomly*;
Initialize precision variable with $p = 1$, $k = 0$ *e* $i = 1$;
Initialize number of points, $n = 1$;
Declare decision variables vector $\boldsymbol{L}^i$ *as non-deterministic integer variables*;
while $k \leq \eta$ **do**
    Define upper and lower limits of $\mathbf{L}$ *with directive* ASSUME, *such as* $L \in \Omega^k$;
    Describe the objective function model $J(\boldsymbol{L})$;
    do
        do
            Define the constraint $J(\boldsymbol{L}^{(i)}) < J(\boldsymbol{L}^{(i-1)})$ *with directive* ASSUME;
            Verify the satisfiability of $J_{optimal}$ *given by* $\mathrm{J}_{\mathrm{optimal}} \Leftrightarrow \mathrm{J}\left(\mathbf{L}^{(\mathrm{i})}\right) \geq \mathrm{J}\left(\mathbf{L}^{(\mathrm{i-1})}\right)$
            Update $\boldsymbol{L}^* = \boldsymbol{L}^{(i)}$ *e* $J(\boldsymbol{L}^*) = J(\boldsymbol{L}^{(i)})$ *based on the counterexample*;
            Do $i = i + 1$;
        while $\neg J_{optimal}$ *is satisfiable*;
        if $\neg J_{optimal}$ *is not consecutively satisfiable* **then**
            break
        end
        else
            Update the number of points, n;
        end
    while *TRUE*;
    Do $k = k + 1$;
    Update the set Ω^k;
    Update the precision variable, p;
end
$\boldsymbol{L}^* = \boldsymbol{L}^{(i)}$ *e* $J(\boldsymbol{L}^*) = J(\boldsymbol{L}^{(i)})$;
return $\boldsymbol{L}^*$ *e* $J(\boldsymbol{L}^*)$;

- Directive ASSUME is used for modeling the constraints set $\mathrm{J}_{\mathrm{optimal}} \Leftrightarrow \mathrm{J}\left(\mathbf{L}^{(\mathrm{i})}\right) \geq \mathrm{J}\left(\mathbf{L}^{(\mathrm{i-1})}\right)$
- Directive ASSERT is used for holding the global optimization condition
- Variable P is used to control precision and discretizes the state-space
- Optimal candidate $J\left(\boldsymbol{L}^*\right)$ is updated if a smaller cost function value is generated from the counterexample

The University of Manchester

ACEGIO-based path planning algorithm

Algorithm 2: ACEGIO-based Path Planning Algorithm

Input : Cost function $J(\mathbf{L})$, a set of obstacles constraints $\mathbb{O}$, a set of environment constraints $\mathbb{E}$, which define Ω and a desired precision η, and a Gradient Descent function $G(\mathbf{L})$

Output: The optimal path $\mathbf{L}^*$ and the optimal cost function value $J(\mathbf{L}^*)$

Initialise $J\left(\boldsymbol{L}^{(0)}\right)$ randomly;

Initialise precision variable with $p = 1, k = 0\ i = 1$;

Initialise precision variable with $n = 1$;

Declare decision variables vector $\boldsymbol{L}^i$ as non-deterministic integer variables;

while $k \leq \eta$ **do**

  Define upper and lower limits of $\boldsymbol{L}$ with directive ASSUME, such as $L \in \Omega^k$;

  Describe the objective function model $J(\boldsymbol{L})$;

  do

    do

      Define the constraint $J\left(\boldsymbol{L}^{(i)}\right) < J\left(\boldsymbol{L}^{(i-1)}\right)$ with directive ASSUME;

      Verify the satisfiability of J_{optimal} given by $\mathrm{J}_{\text{optimal}} \Leftrightarrow \mathrm{J}\left(\mathbf{L}^{(i)}\right) \geq \mathrm{J}\left(\mathbf{L}^{(i-1)}\right)$ directive ASSERT;

      if $\neg l_{optimal}$ *is satisfiable* **then**

        Update $\boldsymbol{L}^* = \boldsymbol{L}^{(i)}$, $J(\boldsymbol{L}^*) = J\left(\boldsymbol{L}^{(i)}\right)$ based on the counterexample;

        Do $i = i + 1$;

        Update $\boldsymbol{L}^{(i)} = G\left(\boldsymbol{L}^{(i-1)}\right)$, and $J\left(\boldsymbol{L}^{(i)}\right) = J\left(\mathrm{G}\left(\boldsymbol{L}^{(i-1)}\right)\right)$;

      end

    while $\neg J_{optimal}$ *is satisfiable*;

    if $\neg J_{optimal}$ *is not consecutively satisfiable* **then**

      break ;

    else

      Update the number of points, n;

    end

  while *TRUE*;

  Do $k = k + 1$;

  Update the set Ω^k;

  Update the precision variable, p;

  Set $\boldsymbol{L}^i$ as non-deterministic integer variables;

end

$\boldsymbol{L}^* = \boldsymbol{L}^{(i-1)}$, $J(\boldsymbol{L}^*) = J\left(\boldsymbol{L}^{(i-1)}\right)$;

return $\boldsymbol{L}^* J(\boldsymbol{L}^*)$;

- The input contains Gradient descent function G(L)
- ACEGIO-GD additionally employs Gradient Descent to calculate optimal candidates iteratively
- New optimal candidates are generated either by extracting from counterexamples or by applying Gradient Descent

The University of Manchester

Experimental Evaluation

Experimental goals:

1. **Effectiveness**: evaluate the effectiveness of the ACEGIO-based path planning algorithm
2. **Efficiency**: compare with CEGIO-based path planning algorithm
3. **State-of-the-art**: compare with other state-of-the-art approaches

MANCHESTER
1824
The University of Manchester

Experimental Setup

Tools for executing CEGIO-based path planning algorithm and ACEGIO-based path planning algorithm:

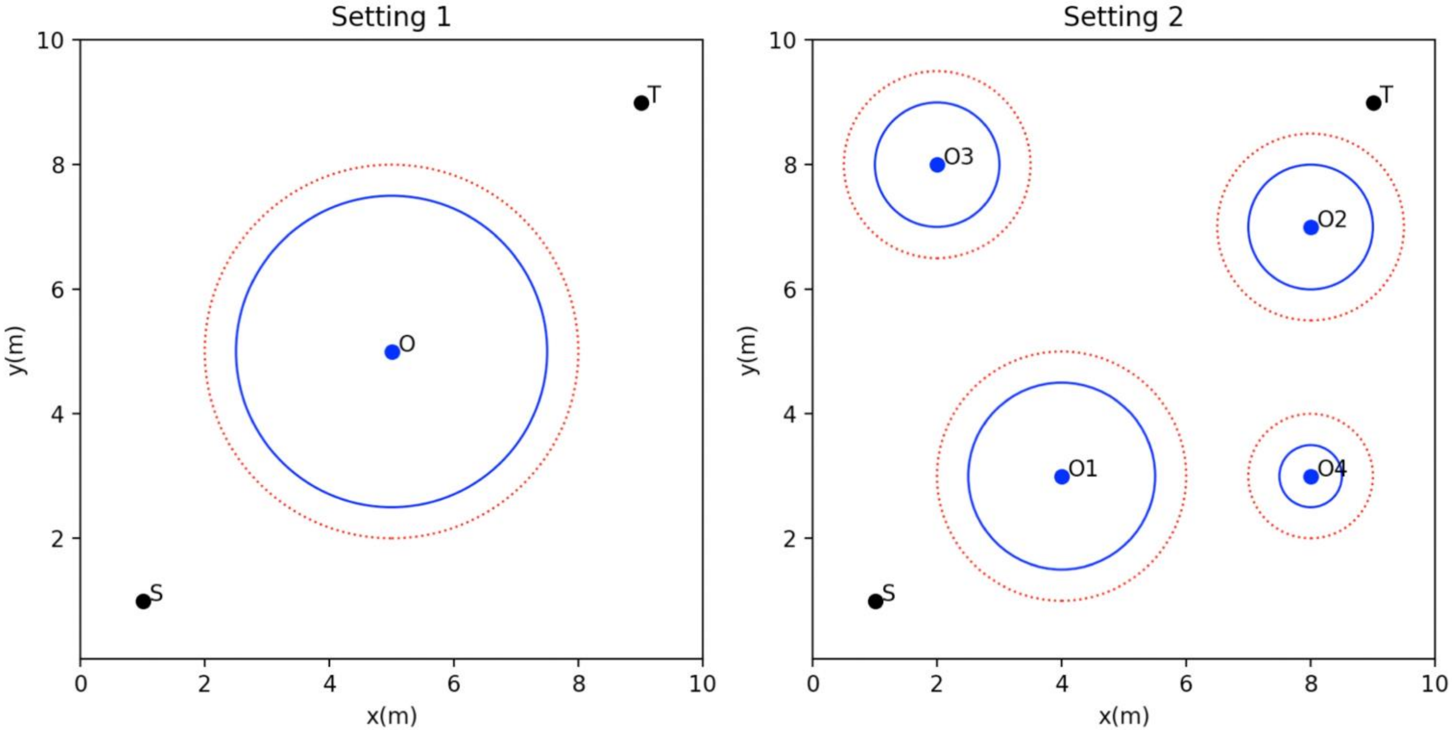
- Model checker: ESBMC 6.4.0 64-bits
- SMT solver: Boolector 3.0
- 2.3 GHz OCTA Intel Core i9 processor with 16GB of RAM, running macOS Catalina 10.15.6 64-bits

Tools for executing GA-based path planning algorithm and PSO-based path planning algorithm:

- Matlab R2021a

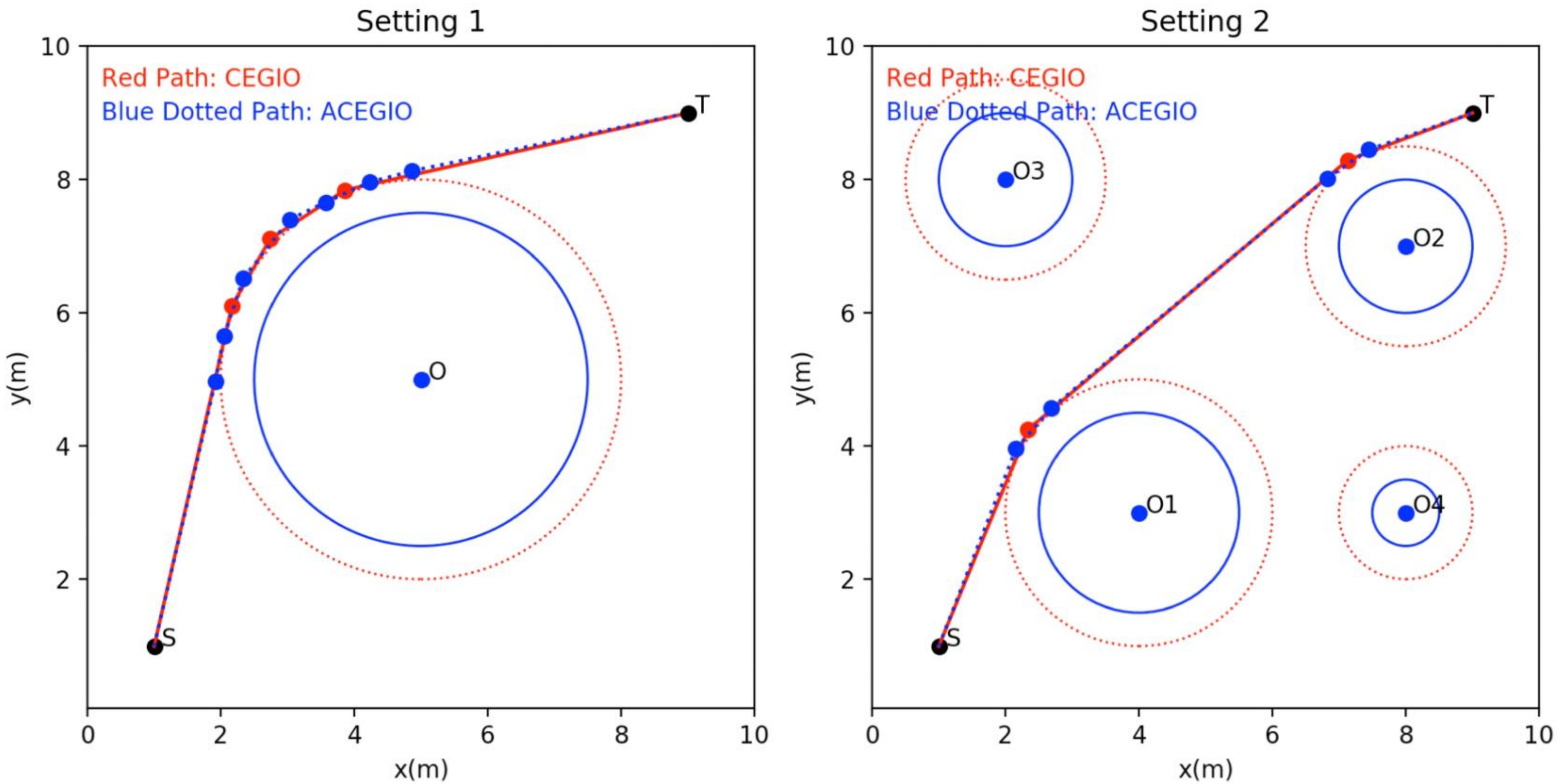
Experimental environment settings

- Motion space is modeled as square
- Obstacles are modeled as circles (blue circles), safety margin is represented by red circles
- Point S (1,1) is the starting point and Point T (9,9) is the target point



Experimental Results

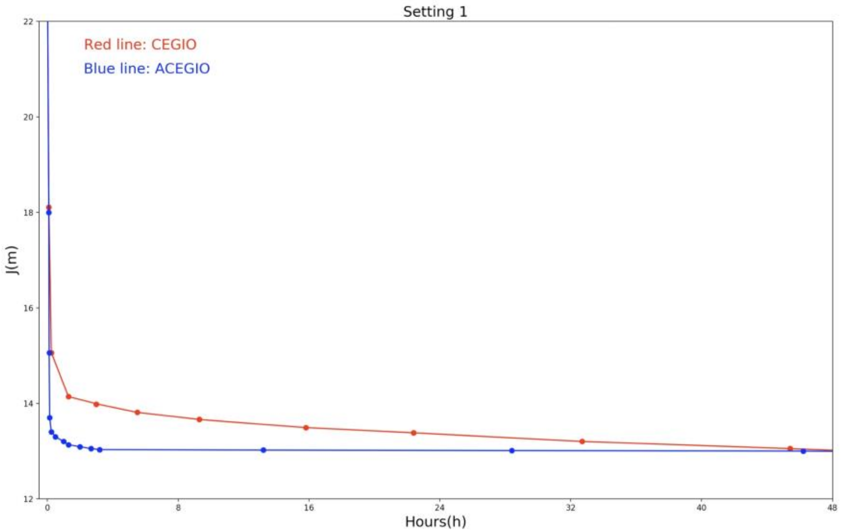
Evaluate CEGIO-based path planning algorithm and ACEGIO-based path planning algorithm

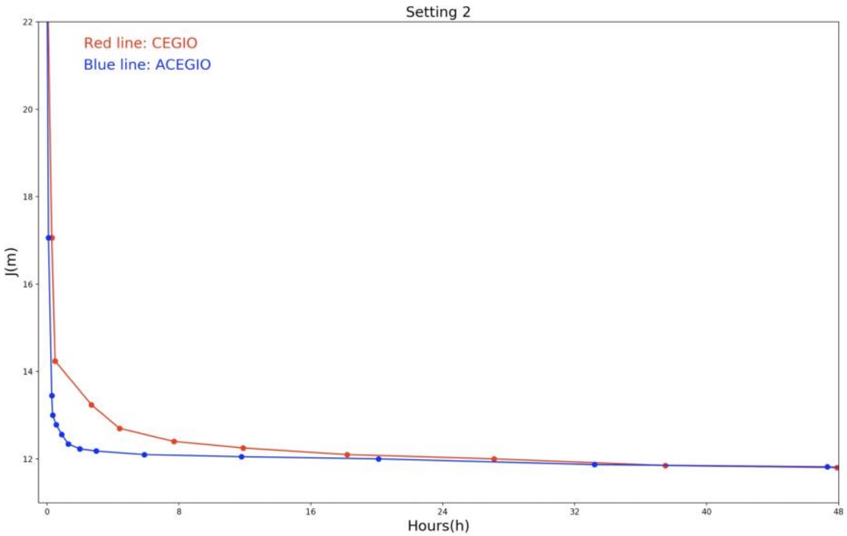


EG1: ACEGIO-based path planning algorithm can generate the global optimal path or a path close to the global optimal path

Experimental Results

Compare CEGIO-based path planning algorithm and ACEGIO-based path planning algorithm





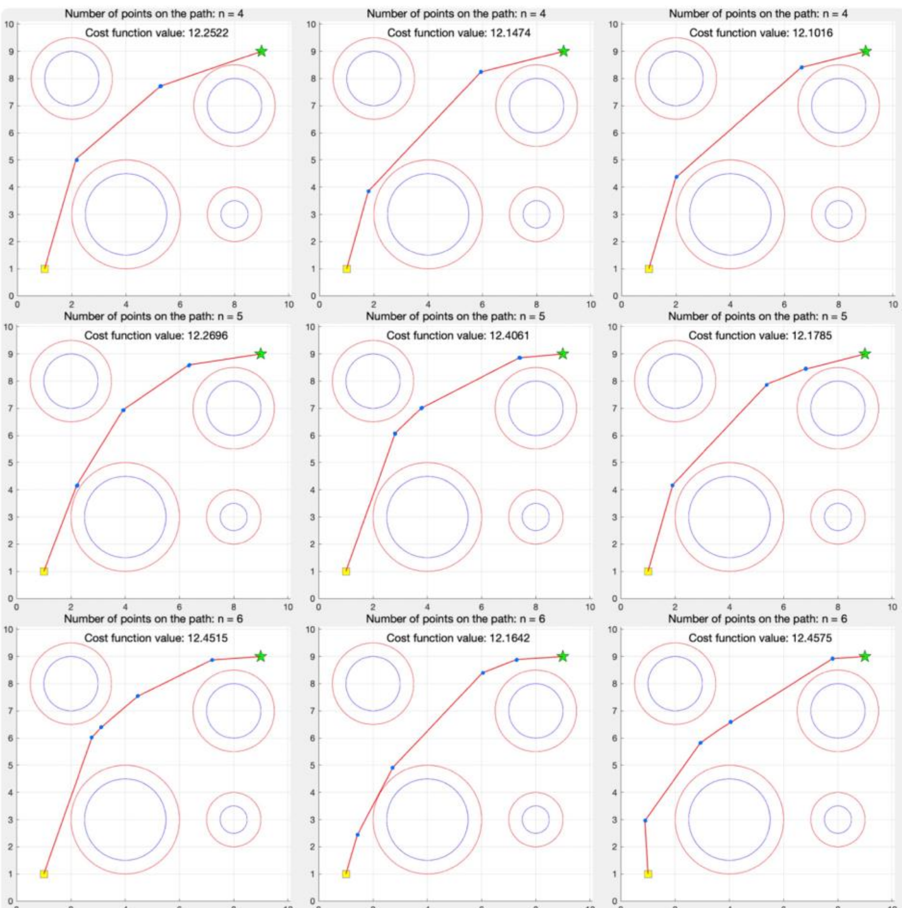
- Reduction trend of cost function values which were obtained by the evaluated algorithms
- Horizontal axis represents the time of optimizing the path planning problem
- Cost function values are shown in the vertical axis

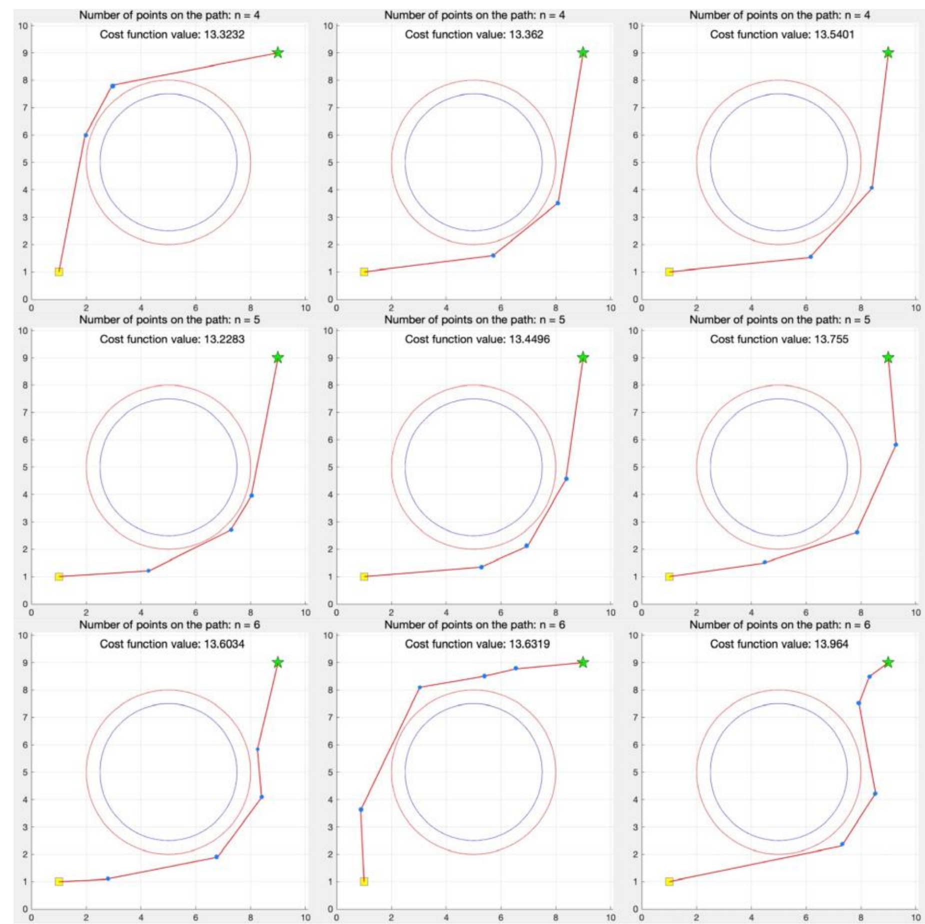
EG2: ACEGIO-based path planning algorithm generate paths closer to the global optimality with less execution time comparing to CEGIO-based path planning algorithm

MANCHESTER 1824
The University of Manchester

Experimental Results

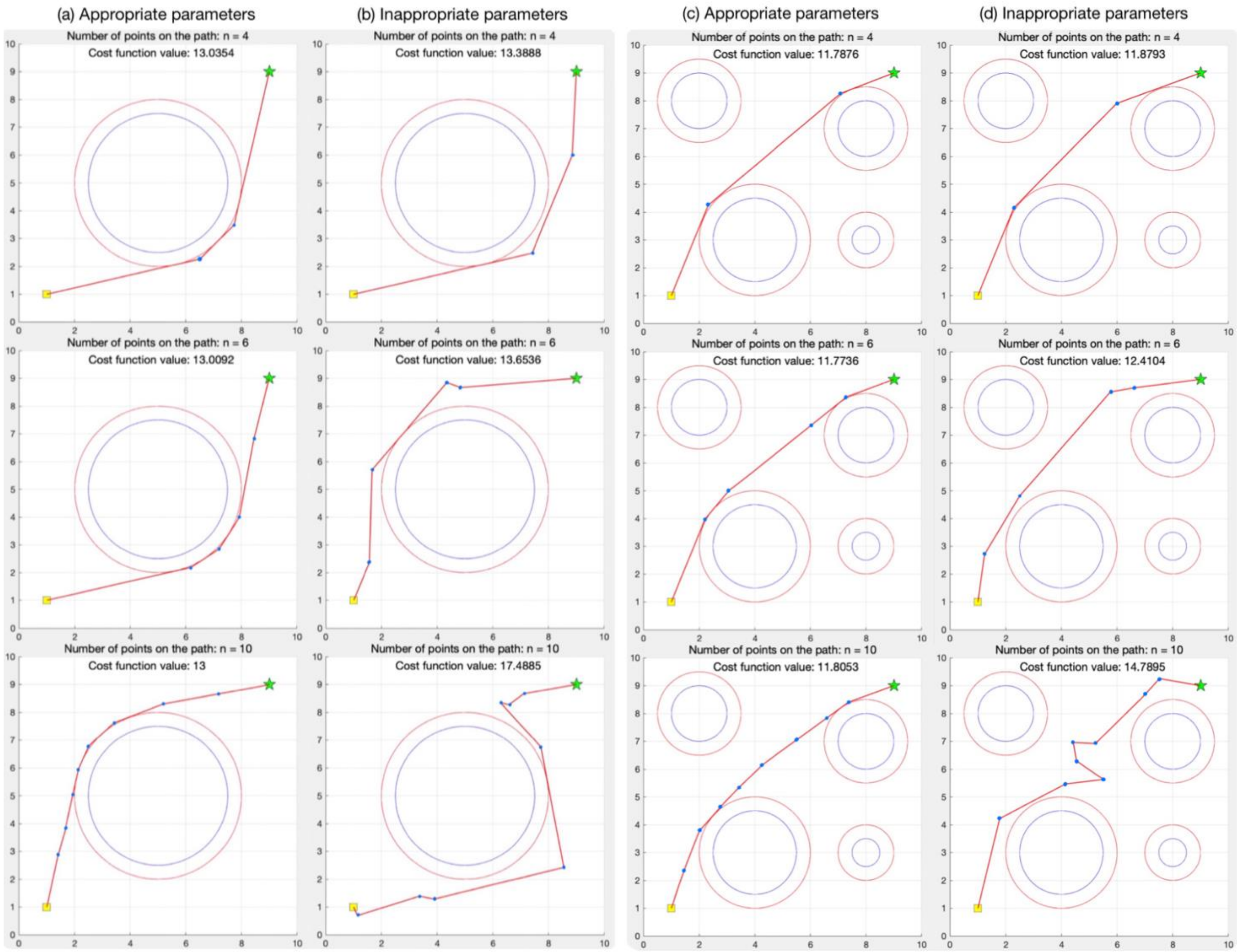
Evaluate GA-based path planning algorithm





Experimental Results

Evaluate PSO-based path planning algorithm



Experimental Results

Compare among path planning algorithms based on ACEGIO, GA and PSO

1. **GA-based path planning algorithm**: much faster, while **ACEGIO-based path planning algorithm**: more effective and more stable
2. **PSO-based path planning algorithm**: stable, fast and reliable, but not easy to tune parameters. **ACEGIO-based path planning algorithm:** much easier to tune.

EG3: If compared to path planning algorithms based on GA and PSO, the execution time of the proposed algorithm is relatively high, whereas its performance is stable, reliable and robust

MANCHESTER 1824
The University of Manchester

Conclusions

- We presented a novel mobile robot path planning algorithm, which relies on the ACEGIO-GD algorithm to solve the optimal path planning problem
- The proposed algorithm can generate optimal paths with significantly shorter execution time than the original CEGIO-based path planning algorithm
- If compared to GA-based path planning algorithm, ACEGIO-based path planing is slow but more stable and reliable
- If compared to PSO-based path planning algorithm, ACEGIO-based path planning is slow but more robust

Future work

- develop other auxiliary algorithms to assist CEGIO
- explore the best auxiliary algorithm for CEGIO to solve optimal path planning problems