



# Pursuit-Evasion Games I

Tomáš Kroupa

Department of Computer Science  
Faculty of Electrical Engineering  
Czech Technical University in Prague

2022

## Pursuit-Evasion in Mobile Robotics

*One or more pursuers try to capture one or more evaders who try to avoid capture.*

- The study of motion planning problems in adversarial settings
  - Detecting intruders
  - Playing hide-and-seek
  - Catching burglars
- The **planner** seeks an optimal strategy against the worst-case **adversary**

## Classes of Pursuit-Evasion Games

### Differential

- Hamilton-Jacobi-Isaacs differential equations model the dynamics
- Their solutions are players' strategies as control inputs for achieving the objectives
- 👍 Velocity or acceleration are expressed explicitly as differential constraints
- 👎 The resulting equations are very complicated and difficult to solve

### Combinatorial

- A real environment is modeled as a polygon or graph
  - *The Cops and Robbers Game*
  - *Parson's game*
  - *The lion-and-man game*
- 👍 Complexity results and guarantees in terms of the size of game
- 👎 Abstraction from the continuous features of environment

## Lecture Goals and Outline

*To understand how*

- 👍 the robotic motion planning changes in the presence of an adversary pursuing their own goals and*
- 👍 the robot's navigation can be enhanced using the game-theoretic methods in this case.*

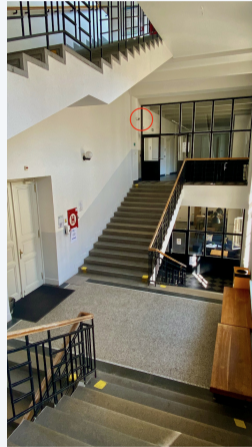
- 1 Motivation: An adversarial path planning problem
- 2 Two-player zero-sum games
- 3 Double Oracle algorithm for solving large games

# **Adversarial Path Planning Problem**

---

## What Path Should the Robot Follow to Avoid CCTV?

👍 The position of cameras is known



Adversarial Path Planning Problem

## What Path Should the Robot Follow to Avoid CCTV?

👉 The position of cameras is known

*The planner navigates a robot to a goal location in a previously mapped environment.*

### Planner

- Models the problem as a single-agent *Markov decision process*
- Must find a path minimizing the robot's visibility to cameras

### Adversary

- Not present in the model

## What Path Should the Robot Follow to Avoid CCTV?



👉 What are optimal camera locations?



Adversarial Path Planning Problem



## What Path Should the Robot Follow to Avoid CCTV?

👉 What are optimal camera locations?

*Both the planner and adversary can control the environment actively.*

### Planner

- Path  $\pi$  for the robot
- Finite set of paths  $\Pi$
- Probability distribution  $p \in \Delta_{\Pi}$

### Adversary

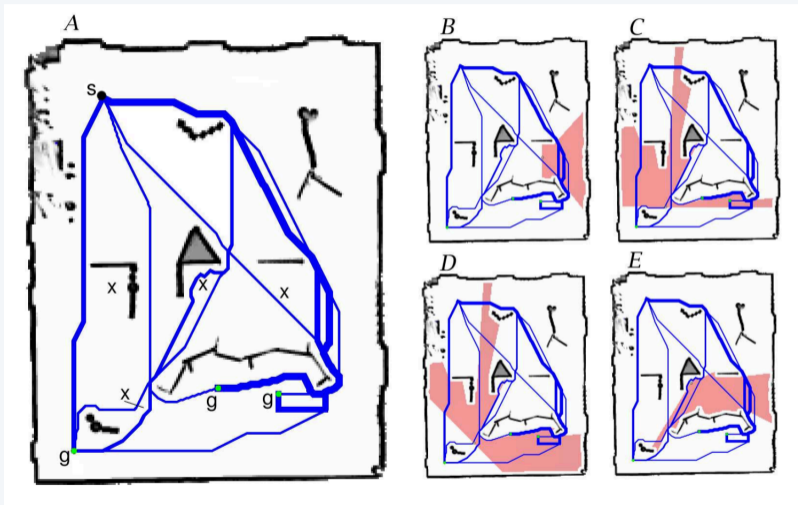
- Cost vector  $\mathbf{c}$
- Finite set of cost vectors  $\mathcal{C}$
- Probability distribution  $q \in \Delta_{\mathcal{C}}$

Let  $V(\pi, \mathbf{c})$  be the value of policy  $\pi$  and cost vector  $\mathbf{c}$ . Solve:

$$\min_{p \in \Delta_{\Pi}} \max_{q \in \Delta_{\mathcal{C}}} \sum_{\pi \in \Pi} \sum_{\mathbf{c} \in \mathcal{C}} p(\pi) q(\mathbf{c}) V(\pi, \mathbf{c})$$

## Example of Solution

Blum et al. (2003)



Adversarial Path Planning Problem

- The gridworld of size up to  $269 \times 226$
- The robot can move in any of 16 compass directions
- Each cell has cost 1 and a cost proportional to the distance of camera

### Computational limits

- 👍 Sets  $\Pi$  and  $C$  should be reasonably small
- 👍 Already  $\binom{100}{2} = 4950$  positions exist for 2 cameras in the gridworld  $10 \times 10$

# **Two-Player Zero-Sum Games**

---

## Two-Player Zero-Sum Game

👍 aka Matrix game

- 1 Players are the **planner** and the **adversary**
- 2 **Strategy sets** are  $M$  and  $N$
- 3 The **cost** matrix  $\mathbf{C} = [c_{ij}]_{i \in M, j \in N}$  for the planner
- 4 The loss  $c_{ij}$  for planner playing  $i \in M =$  the reward for adversary playing  $j \in N$

For example:

$$|M| = 2, \quad |N| = 4, \quad \mathbf{C} = \begin{bmatrix} 1 & 0 & 4 & -1 \\ -1 & 1 & -2 & 5 \end{bmatrix}$$

## Minmax/Maxmin Strategies

👉 We seek the optimal performance against the worst-case adversary

- Assume that the agents adopt **minmax/maxmin** strategies  $\bar{i} \in M$  and  $\bar{j} \in N$ :

$$\bar{i} = 1, \quad \bar{j} = 2, \quad \mathbf{C} = \begin{bmatrix} 1 & 0 & 4 & -1 \\ -1 & 1 & -2 & 5 \end{bmatrix}$$

The floor on the reward of adversary =  $0 \leq 4$  = The ceiling of the cost of planner

$$\max_{j \in N} \min_{i \in M} c_{ij} \leq c_{\bar{i}\bar{j}} \leq \min_{i \in M} \max_{j \in N} c_{ij}$$

- Now, the adversary can increase the profit by playing  $j = 3$
- In this case the planner would adopt  $i = 2$
- Then the adversary would play  $j = 4$  etc.

## Mixed Strategies

👉 Randomize!

A **mixed strategy** of a player is a probability distribution over the strategy set.

- Let  $\Delta_M$  and  $\Delta_N$  be the sets of mixed strategies of planner/adversary
- If the agents play  $\mathbf{x} \in \Delta_M$  and  $\mathbf{y} \in \Delta_N$ , the **expected loss** of planner is

$$\sum_{i \in M} \sum_{j \in N} x_i y_j c_{ij} = \mathbf{x}^T \mathbf{C} \mathbf{y}$$

In particular, if the adversary uses a **pure strategy**  $\mathbf{e}_j \in \Delta_N$  with  $j \in N$ ,

$$\sum_{i \in M} x_i c_{ij} = \mathbf{x}^T \mathbf{C} \mathbf{e}_j$$

## Minmax/Maxmin in Mixed Strategies

- ① A **minmax strategy** of the planner is a mixed strategy  $\bar{x} \in \Delta_M$  such that

$$\max_{y \in \Delta_N} \bar{x}^T C y = \min_{x \in \Delta_M} \max_{y \in \Delta_N} x^T C y$$

- ② A **maxmin strategy** of the adversary is a mixed strategy  $\bar{y} \in \Delta_N$  such that

$$\min_{x \in \Delta_M} x^T C \bar{y} = \max_{y \in \Delta_N} \min_{x \in \Delta_M} x^T C y$$

It is easy to show that

$$\underbrace{\max_{y \in \Delta_N} \min_{x \in \Delta_M} x^T C y}_{\text{The lower bound on the reward}} \leq \bar{x}^T C \bar{y} \leq \underbrace{\min_{x \in \Delta_M} \max_{y \in \Delta_N} x^T C y}_{\text{The upper bound on the cost}}$$



## Minimax Theorem

👍 von Neumann, 1928

$$\underbrace{\min_{x \in \Delta_M} \max_{y \in \Delta_N} x^T C y}_{\text{The value of the game}} = \max_{y \in \Delta_N} \min_{x \in \Delta_M} x^T C y$$

- 👍 An **equilibrium** is a pair of minmax/maxmin strategies  $(\bar{x}, \bar{y})$
- 👍 For any equilibrium  $(\bar{x}, \bar{y})$ , we obtain

$$\bar{x}^T C \bar{y} = \text{the value of the game}$$

## Computing Minmax Strategy

👉 Linear programming

The inner max can be evaluated using pure strategies only:

$$\min_{\mathbf{x} \in \Delta_M} \max_{\mathbf{y} \in \Delta_N} \mathbf{x}^\top \mathbf{C} \mathbf{y} = \min_{\mathbf{x} \in \Delta_M} \max_{j \in N} \mathbf{x}^\top \mathbf{C} \mathbf{e}_j.$$

We obtain a **linear program** with variables  $x_i$  ( $i \in M$ ) and  $v$  for the planner:

$$\begin{array}{ll} \text{Minimize} & \max_{j \in N} \mathbf{x}^\top \mathbf{C} \mathbf{e}_j \\ \text{subject to} & \mathbf{x} \in \Delta_M \end{array}$$

$$\begin{array}{ll} \text{Minimize} & v \\ \text{subject to} & \mathbf{x}^\top \mathbf{C} \mathbf{e}_j \leq v, \quad \forall j \in N \\ & x_i \geq 0, \quad \forall i \in M \\ & \sum_{i \in M} x_i = 1 \end{array}$$

## Computing Minmax Strategy

👉 Example

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 4 & -1 \\ -1 & 1 & -2 & 5 \end{bmatrix}$$

$$\begin{array}{ll} \text{Minimize} & v \\ \text{subject to} & x_1 - x_2 \leq v \\ & x_2 \leq v \\ & 4x_1 - 2x_2 \leq v \\ & -x_1 + 5x_2 \leq v \\ & x_1, x_2 \geq 0 \\ & x_1 + x_2 = 1 \end{array}$$

The equilibrium strategies are  $\bar{\mathbf{x}} = (\frac{7}{12}, \frac{5}{12})$ ,  $\bar{\mathbf{y}} = (0, 0, \frac{1}{2}, \frac{1}{2})$ , and the value is  $\bar{v} = \frac{3}{2}$ .

## Computing Equilibrium

☹ Problems

- 👍 The strategy sets  $M$  and  $N$  are too large in the path planning problems
- 👍 The set of paths and the cost matrix may not be given a priori

We show an iterative method relying on 2 principles:

- 1 Small subgames can be solved efficiently
- 2 Subgames are expanded with best responses

The **best response** of player to a mixed strategy  $\mathbf{y} \in \Delta_N$  is a strategy  $i' \in M$  such that

$$\min_{i \in M} \mathbf{e}_i^\top \mathbf{C} \mathbf{y} = \mathbf{e}_{i'}^\top \mathbf{C} \mathbf{y}.$$

# Double Oracle Algorithm

---

## Double Oracle Algorithm

Blum et al. (2003)

- 1 Pick initial subsets of strategies for each player
- 2 Compute an equilibrium of the subgame
- 3 Expand the current strategy sets with the best responses
- 4 Repeat 2. and 3. until the current equilibrium is good enough

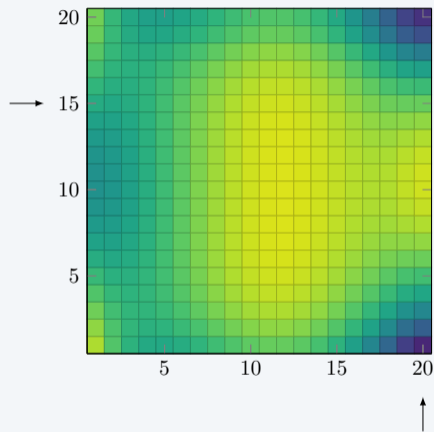
MASTER PROBLEM

SUB-PROBLEM

## Double Oracle Algorithm

Initialize with random pure strategies.

Initialize

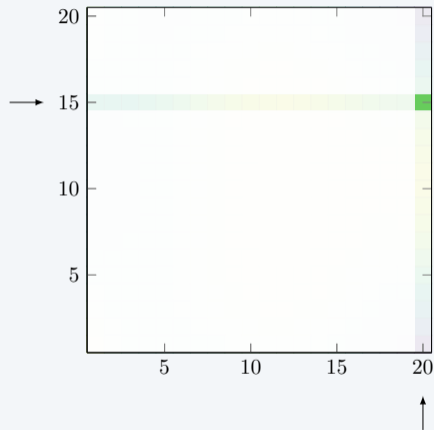


Double Oracle Algorithm

## Double Oracle Algorithm

Find an equilibrium of the  $1 \times 1$  subgame.

Master Problem



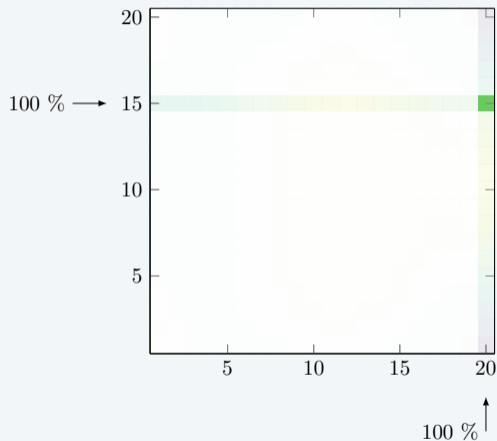
Double Oracle Algorithm



## Double Oracle Algorithm

Find an equilibrium of the  $1 \times 1$  subgame.

Master Problem



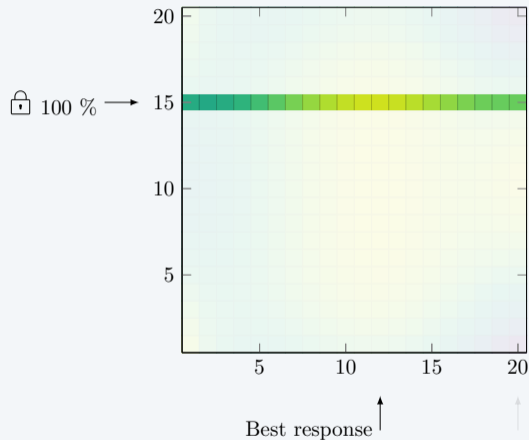
Double Oracle Algorithm

25

## Double Oracle Algorithm

Find adversary's best response against a fixed strategy of the planner.

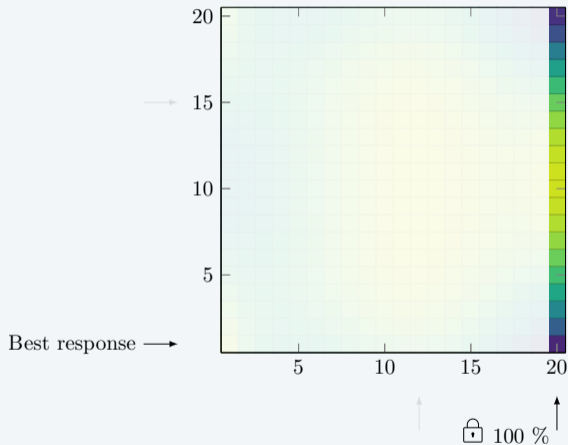
Best Response (adversary)



## Double Oracle Algorithm

Find planner's best response against a fixed strategy of the adversary.

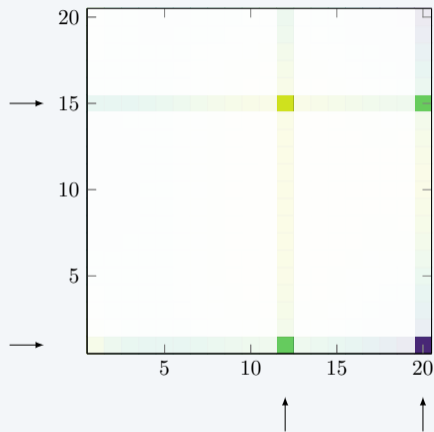
Best Response (planner)



## Double Oracle Algorithm

Find an equilibrium of the  $2 \times 2$  subgame.

Master Problem (Iteration 2)



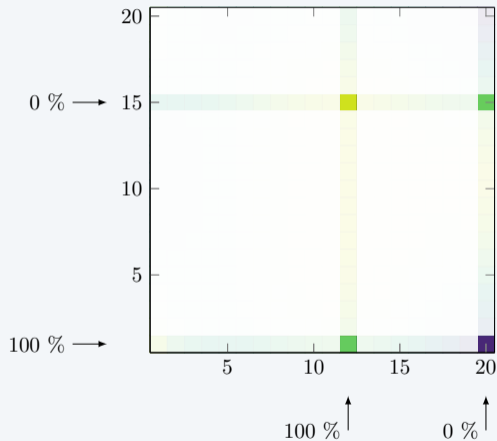
Double Oracle Algorithm

28

## Double Oracle Algorithm

Find an equilibrium of the  $2 \times 2$  subgame.

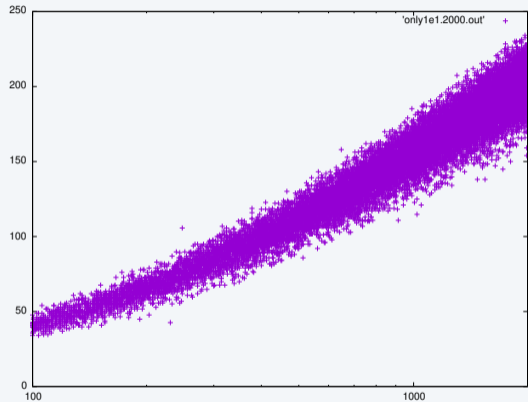
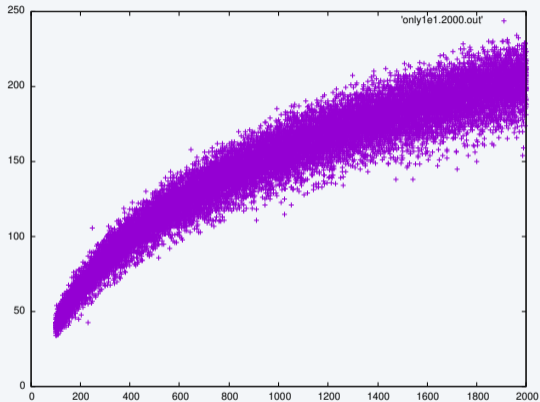
Master Problem (Iteration 2)



- 👍 The algorithm recovers an exact equilibrium in finitely many steps
- 👍 The approximation of equilibrium/value of the game
- 👍 Easy to implement using efficient LP solvers
- 👎 It may need  $O(|M| + |N|)$  iterations

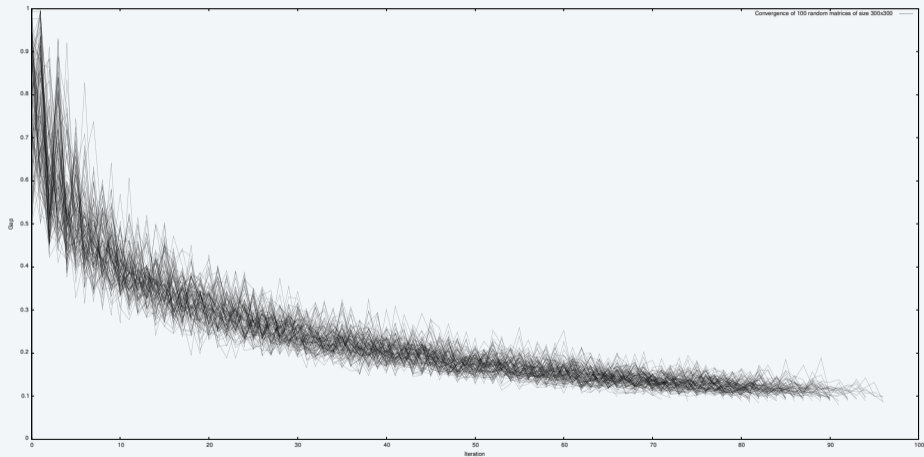
## Convergence to 0.1-equilibrium

👍 The size of matrix vs #iterations



## Convergence to 0.1-equilibrium



👍 #iterations vs convergence criterion  
for  $300 \times 300$  games



Double Oracle Algorithm



## References

-  Chung, Timothy H., Geoffrey A. Hollinger, and Volkan Isler. Search and Pursuit-Evasion in Mobile Robotics. *Autonomous Robots* 31 (4): 299–316, 2011.
-  McMahan, H. Brendan, Geoffrey J. Gordon, and Avrim Blum. Planning in the Presence of Cost Functions Controlled by an Adversary. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 2003.