Faster Motion Planning Using Learned Local Viability Models

Maciej Kalisiak[1] Michiel van de Panne[2]

[1]Department of Computer Science
University of Toronto

[2]Department of Computer Science
University of British Columbia

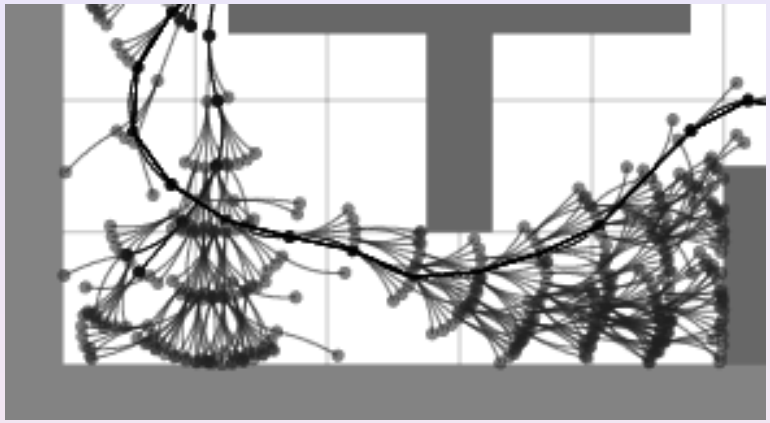
International Conference on Robotics & Automation, 2007

Outline

Outline

General observations

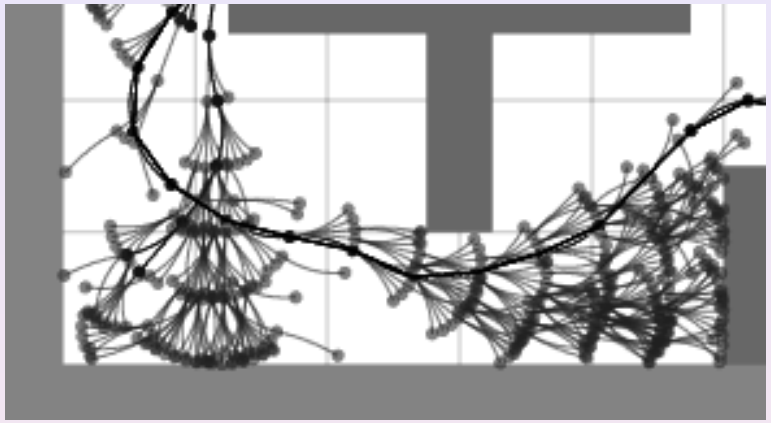
- differential constraint planners: room for improvement
- current planners:
  - do questionable explorations (e.g., try to drive into walls)
  - they keep doing this, repeatedly (i.e., "experience" not accumulated)
- underlying problems:
  - planners cannot "see"
  - planners do not learn (transferrable skills)



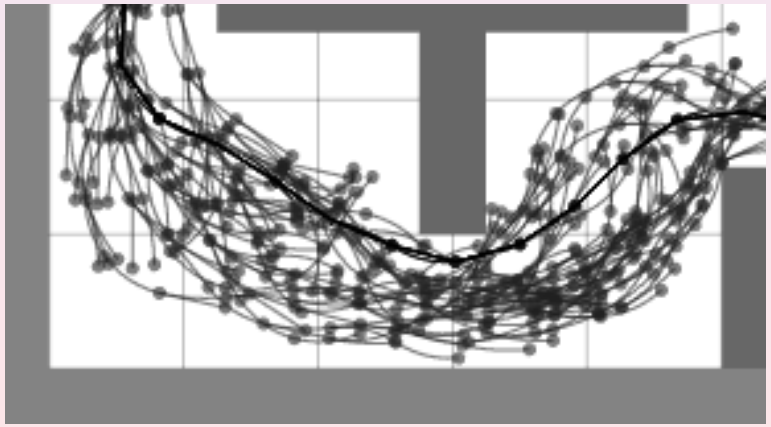
inefficient

Specific problems addressed

- we address these shortcomings
  - provide "sight"
  - avoid "questionable" exploration
- the point: greater efficiency, speed up of up to 10x



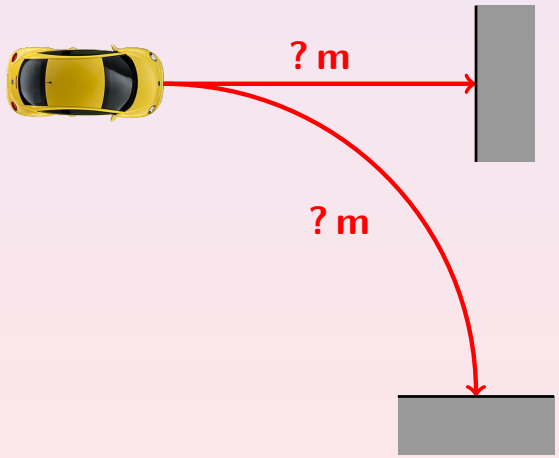
inefficient



better

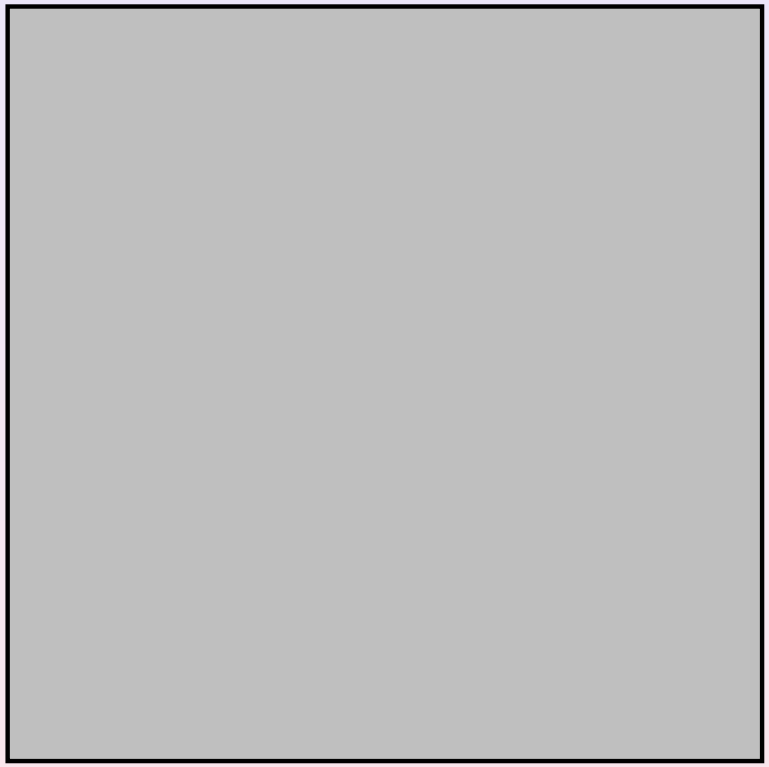
Adding "sight"

- "sight"
  - needed to anticipate and avoid traps, behave smarter
  - collision-check: only a tactile sense
  - need longer range, "perception at a distance"
- $\Rightarrow$ augment agent with virtual sensors
  - measure *agent* $\leftrightarrow$ *environment* distance along line or curve



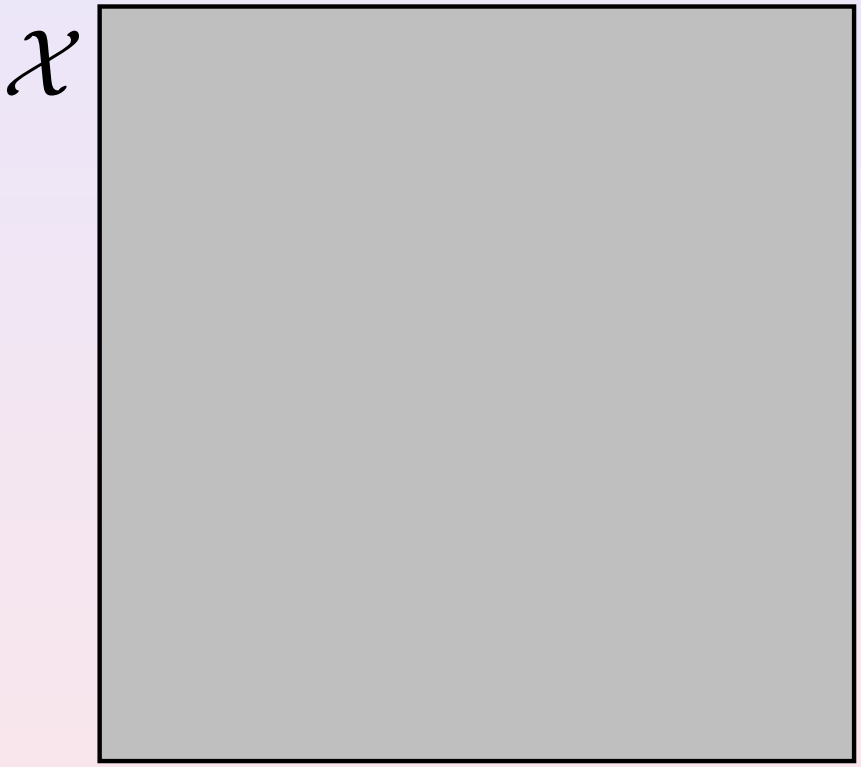
Learning smarter behaviour

- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



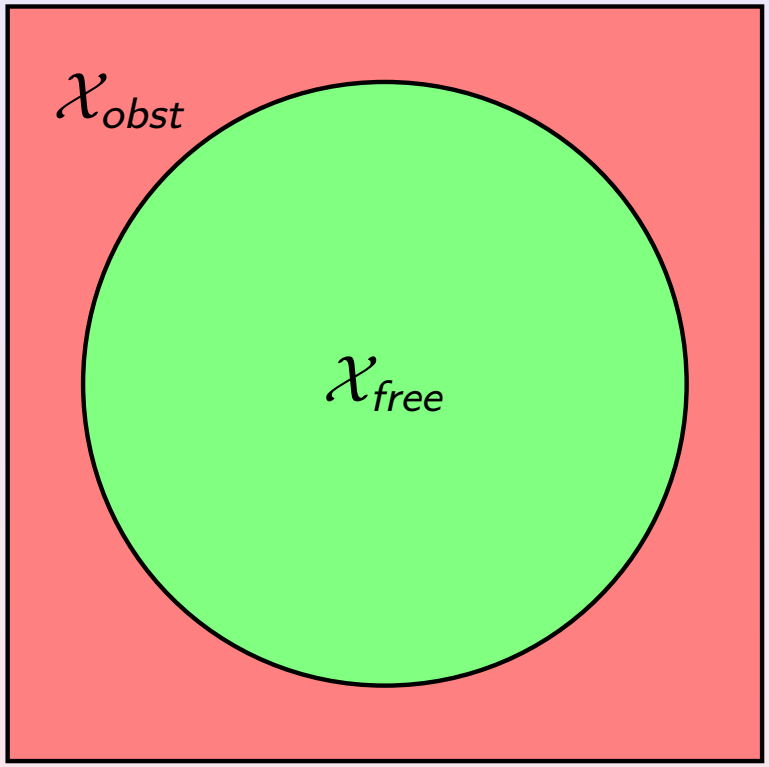
Learning smarter behaviour

- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



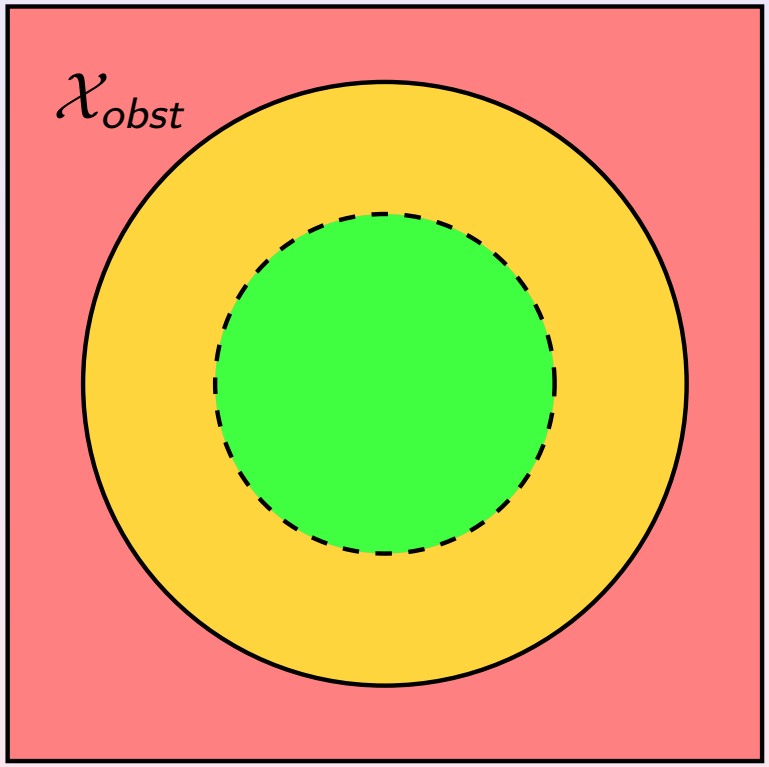
Learning smarter behaviour

- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



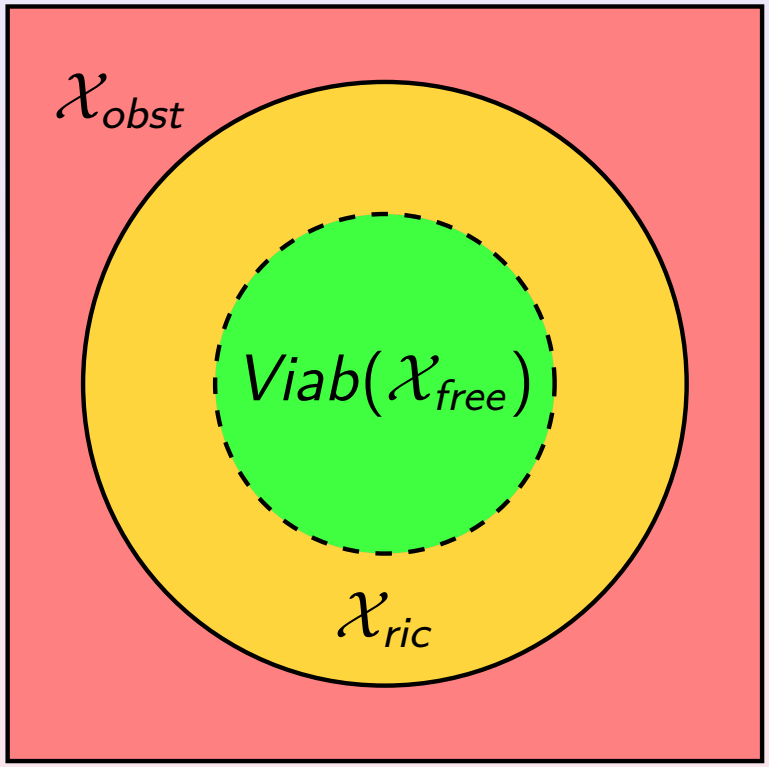
Learning smarter behaviour

- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



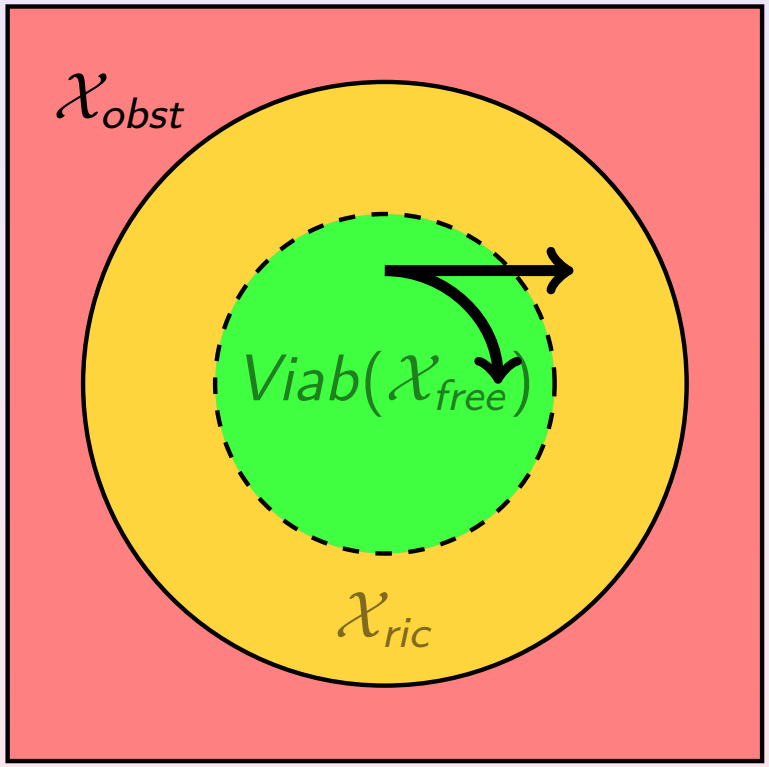
Learning smarter behaviour

- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



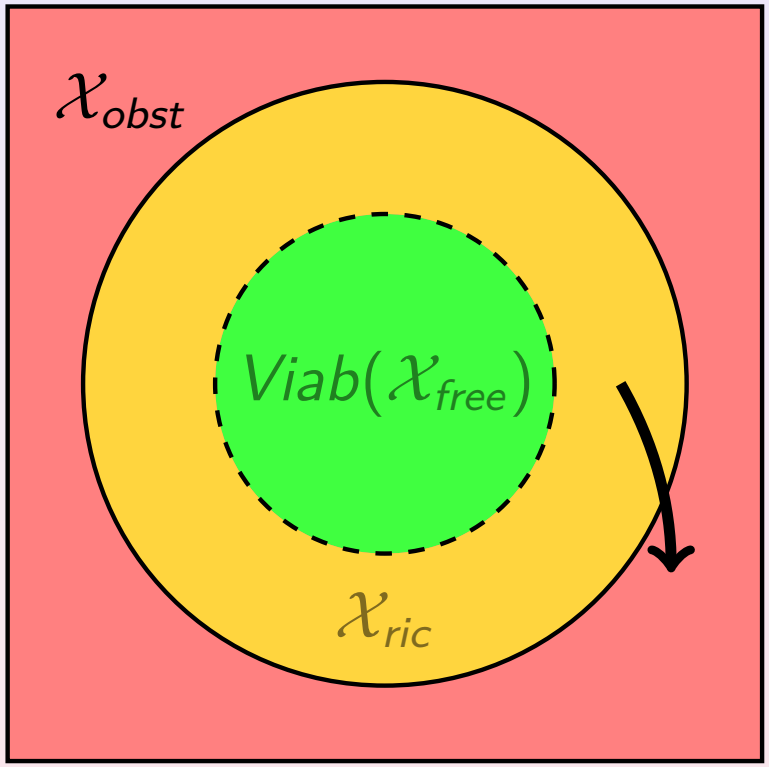
Learning smarter behaviour

- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



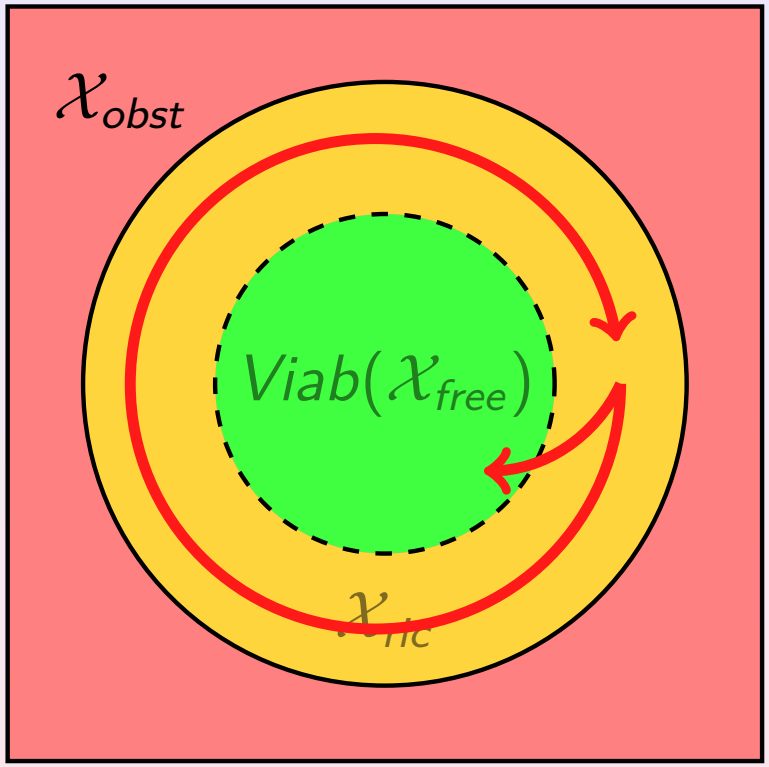
Learning smarter behaviour

- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



Learning smarter behaviour

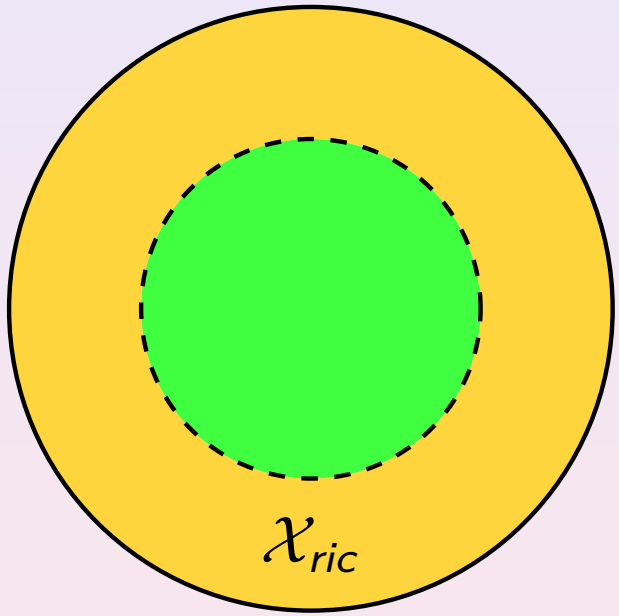
- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



impossible!

Learning smarter behaviour

- "questionable explorations" =
  - nonviable states
    (Viability Theory: J.P.Aubin)
  - $\mathcal{X}_{ric}$
    (J.Kuffner & S.LaValle)
  - Inevitable Collision States (ICS)
    (T.Fraichard *et al.*)
- goal: learn these states, avoid them
  i.e., viability filtering
- same solutions, less time & effort



Why viability filtering makes sense

basic observation from viability theory

- a nonviable state (e.g., $x_{nv} \in \mathcal{X}_{ric}$) cannot lead to a viable state
- if it did, x_{nv} would be viable, by definition

Thus

- if x_{goal} viable:
  - x_{nv} cannot lead to x_{goal}
  - $\Rightarrow$ x_{nv} cannot be part of a solution
  - $\Rightarrow$ exploring x_{nv} = pointless, waste of effort
- if x_{goal} nonviable:
  - still partially helpful
  - automatically resolved when using two trees (see paper)

Outline

Adding "viability filtering" to a planner

Retrofitting a planner

Simple:

1. build or obtain a local viability model for agent
2. replace calls to

   `collision_check(`x`)`

   with

   `nonviable_check(`x`)`

Modeling viability

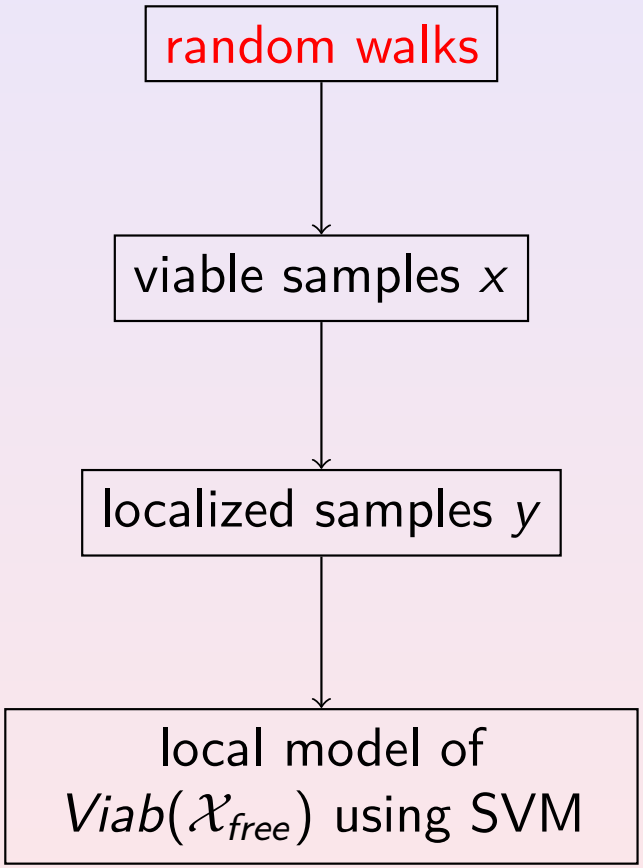
problem

- $Viab(K)$ usually not known ahead of time;
  where does $Viab(K)$ end and $\mathcal{X}_{ric}$ start?

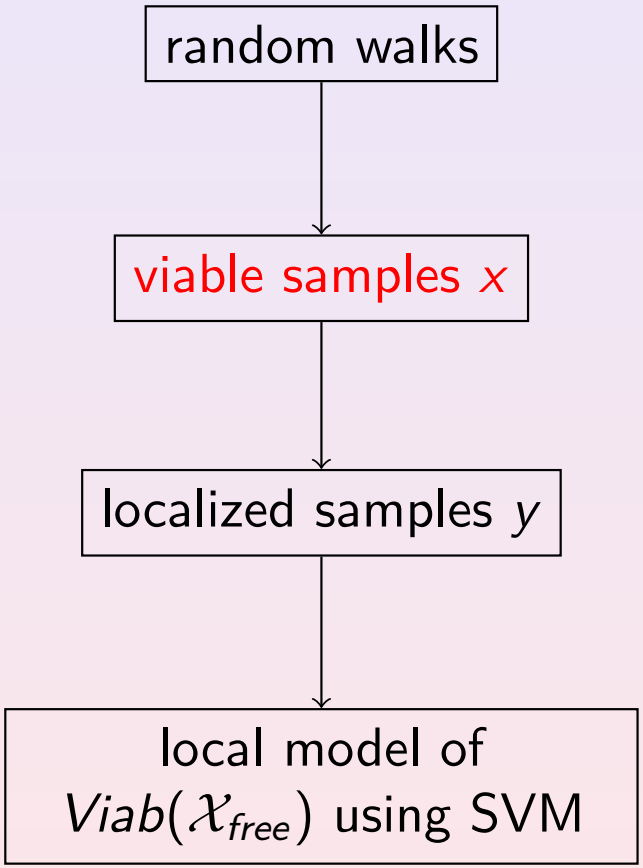
solution

- empirical data + simple heuristic $\longrightarrow$ approximate model
- prior solution trajectories: potential empirical data source
- model is local: parametrized by virtual sensors' output

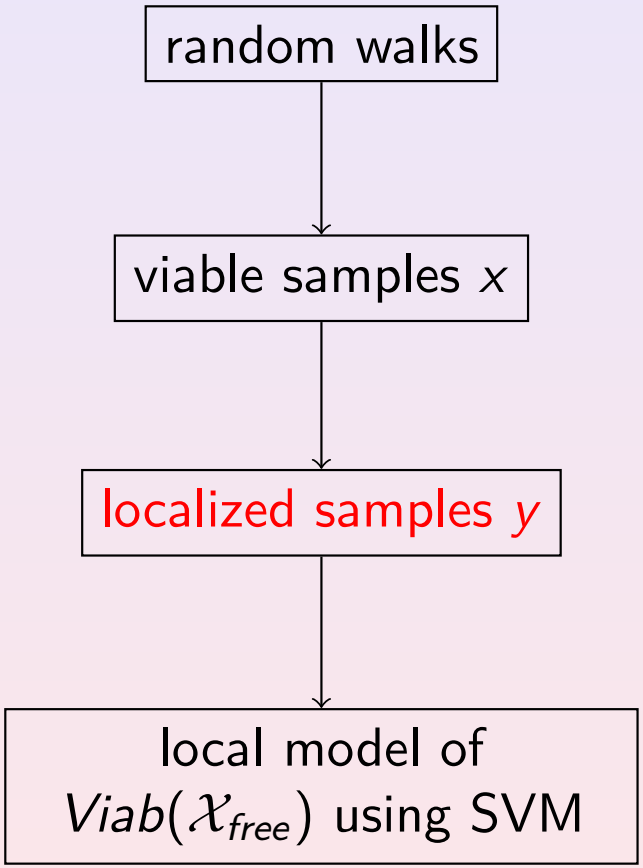
Our model building process

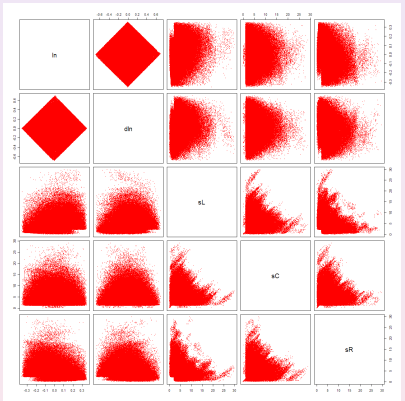


Our model building process

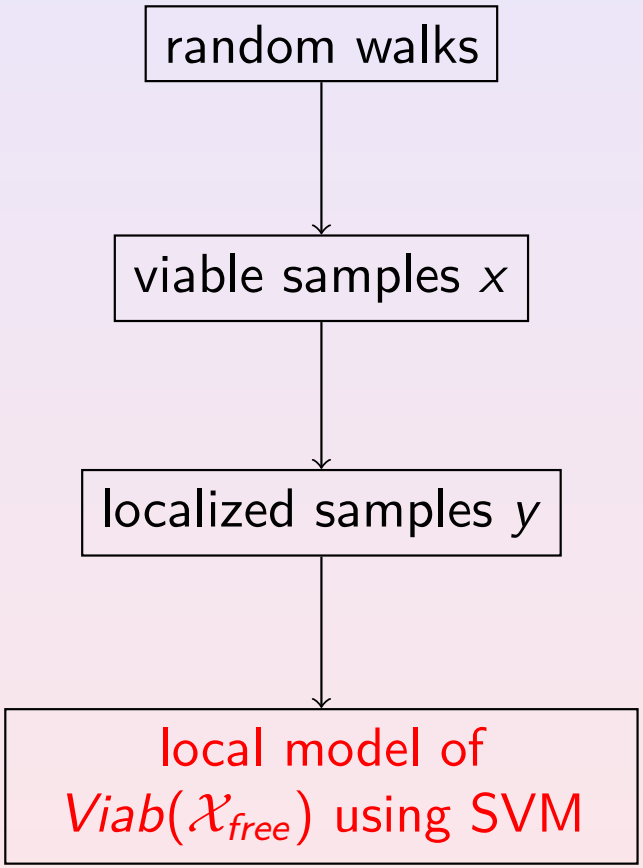


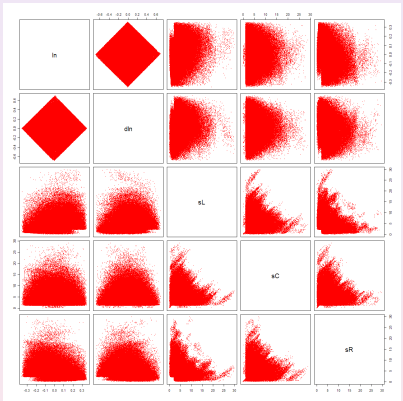
Our model building process

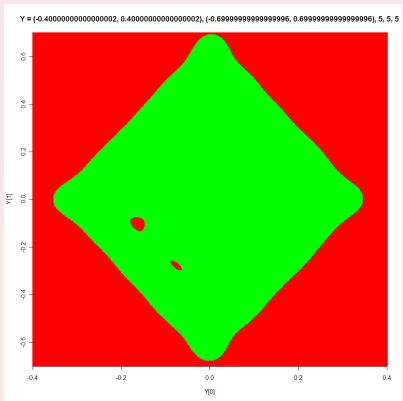


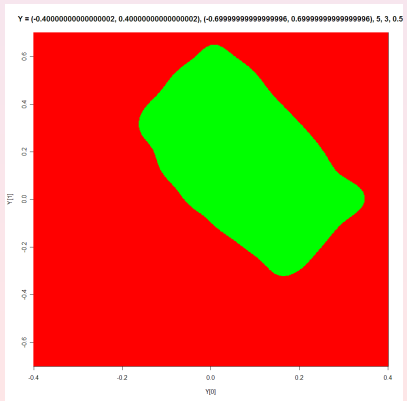


Our model building process



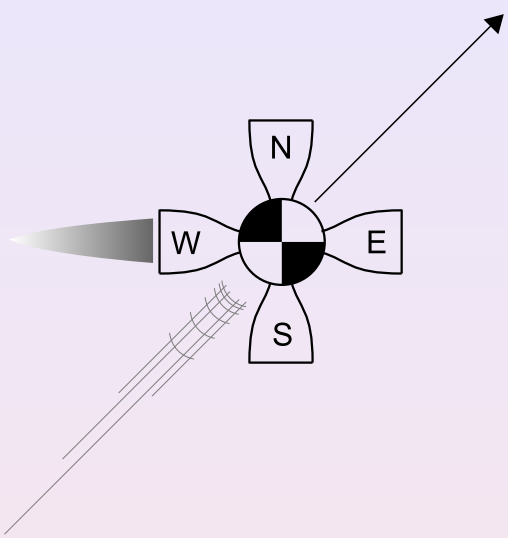


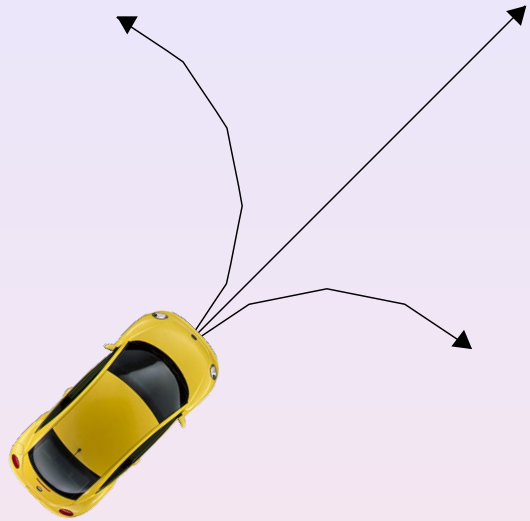


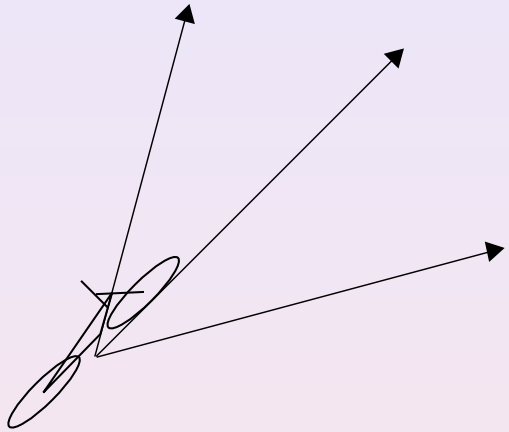


Outline

Agents & sensors







inertial point

- one thruster always "on"
- sensor along velocity vector

car

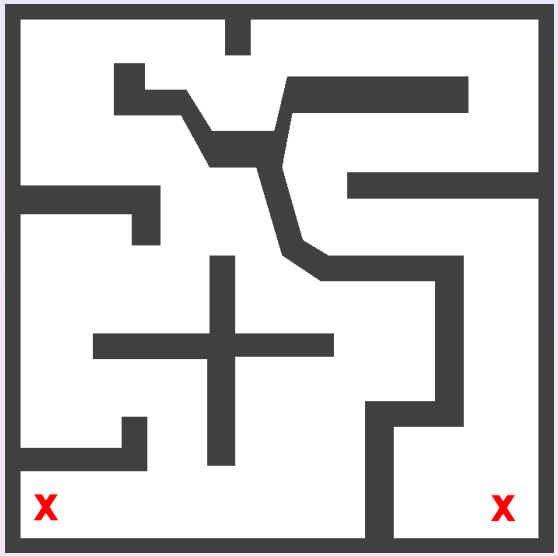
- minimum turning radius: large
- fixed forward velocity
- curved path sensors: max 180°

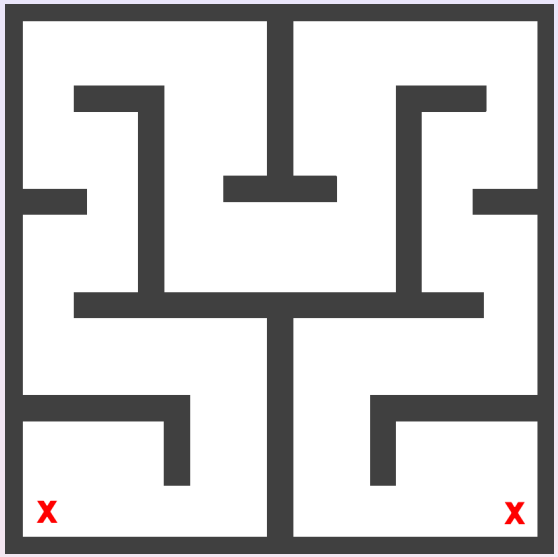
bike

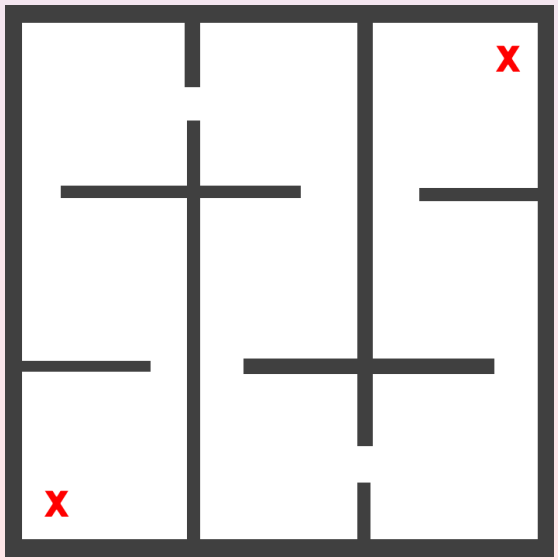
- fixed forward velocity
- steering for balance and navigation
- failure if lean exceeds 60°

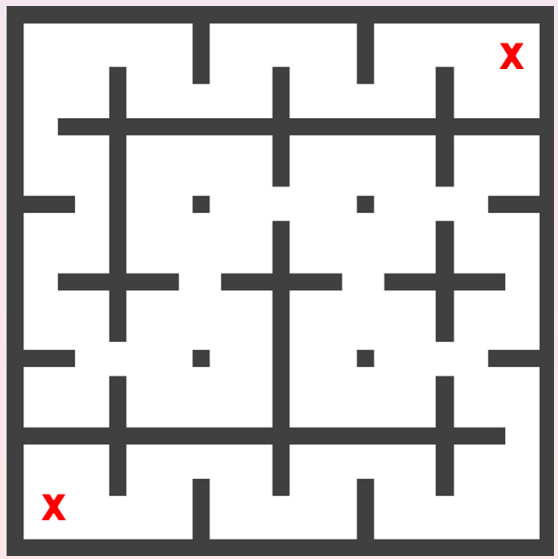
Environments

some environments tested

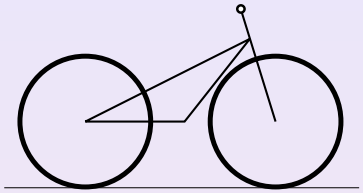




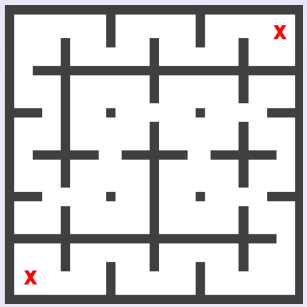




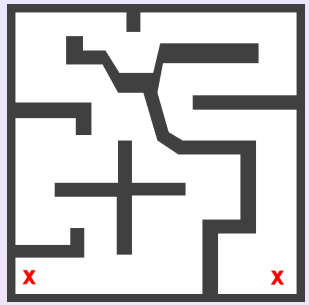
Sample results



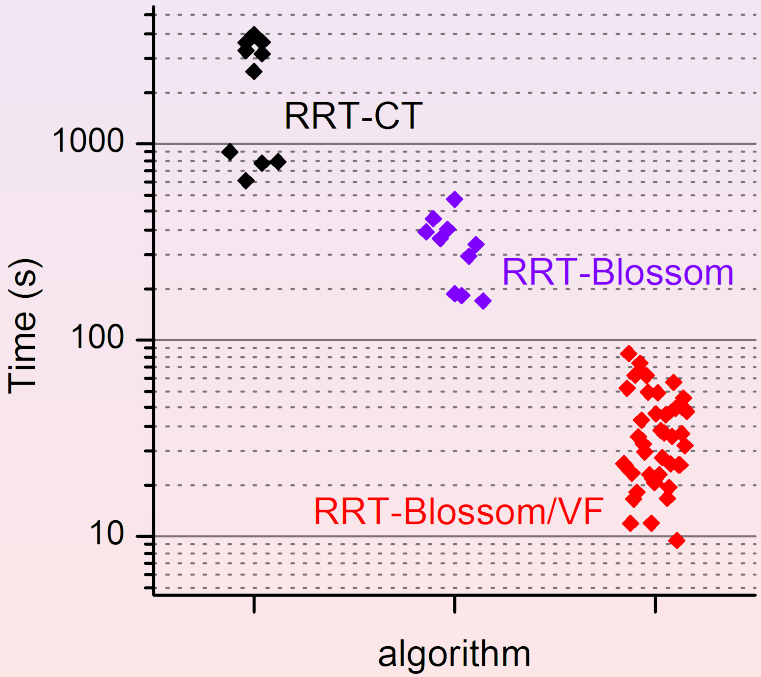
agent



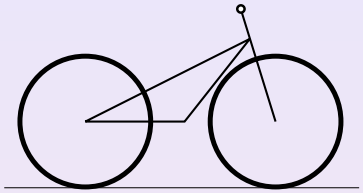
problem posed



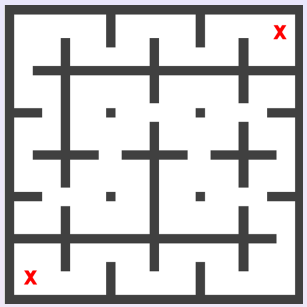
model trained on



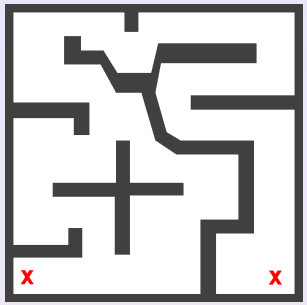
Sample results



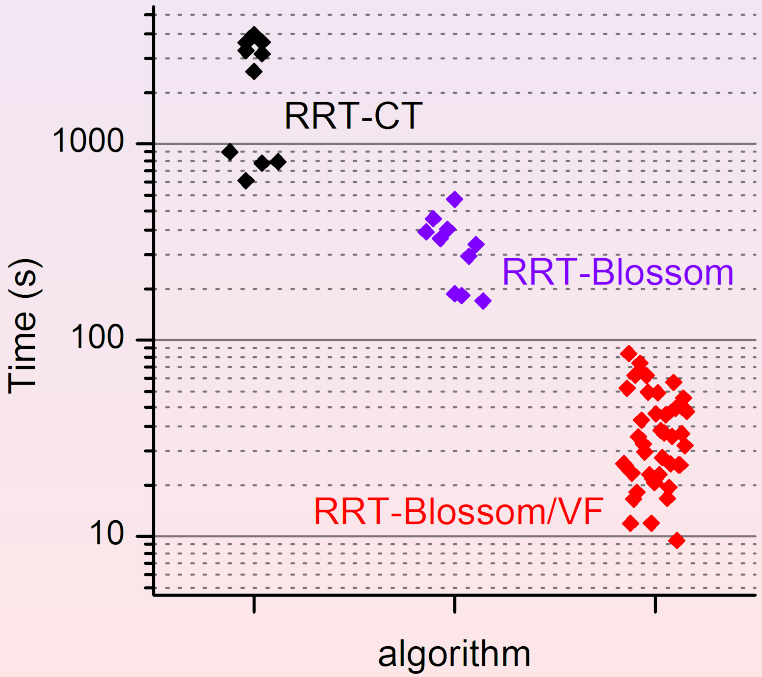
agent

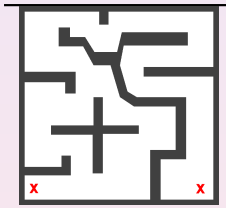
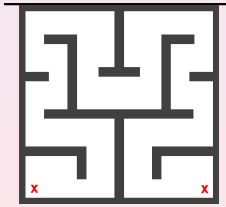
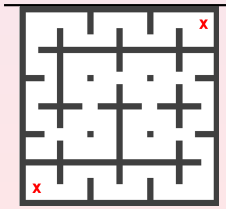


problem posed



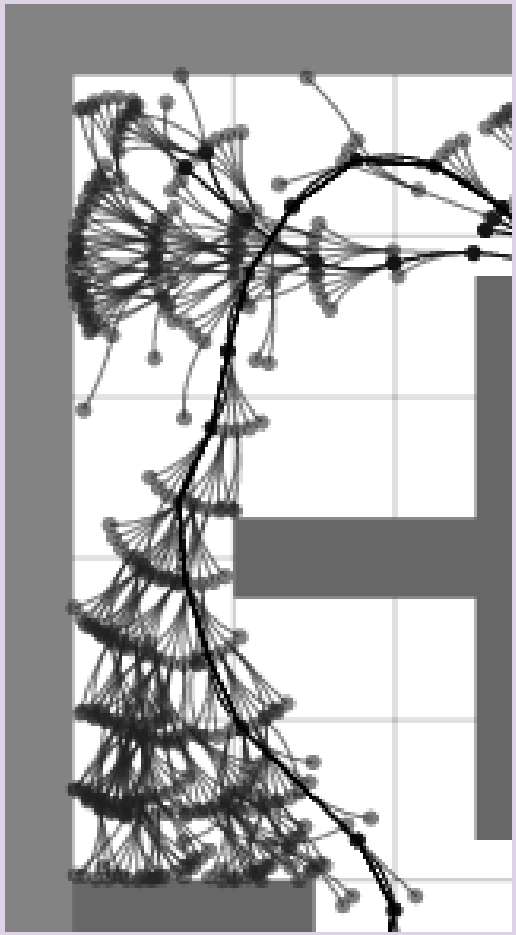
model trained on



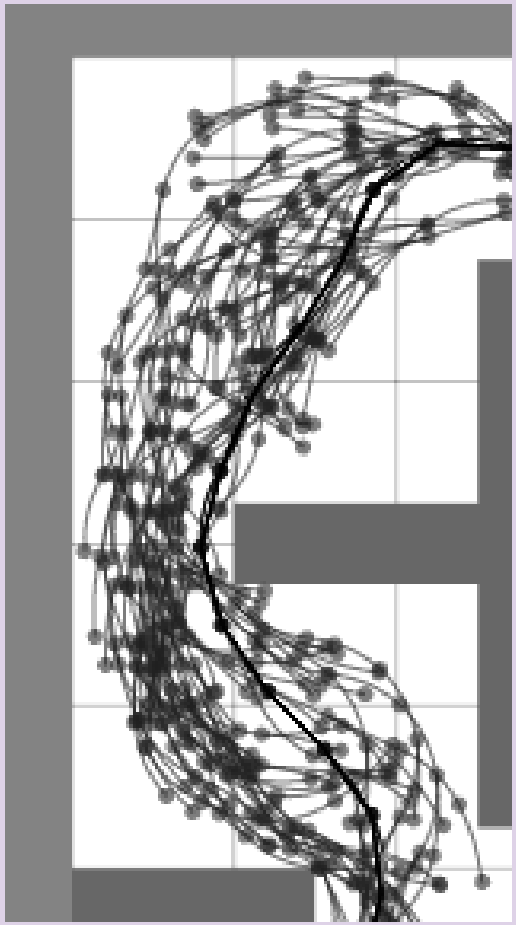
problem	algo.	runtimes
	RRT-CT	371.5s
	RRT-Blossom	21.0s
	RRT-Blossom-VF	5.6s
	RRT-CT	209.9s
	RRT-Blossom	13.5s
	RRT-Blossom-VF	3.6s
	RRT-CT	2148.6s
	RRT-Blossom	305.7s
	RRT-Blossom-VF	34.3s

Effect of viability filtering on tree branches

without filtering

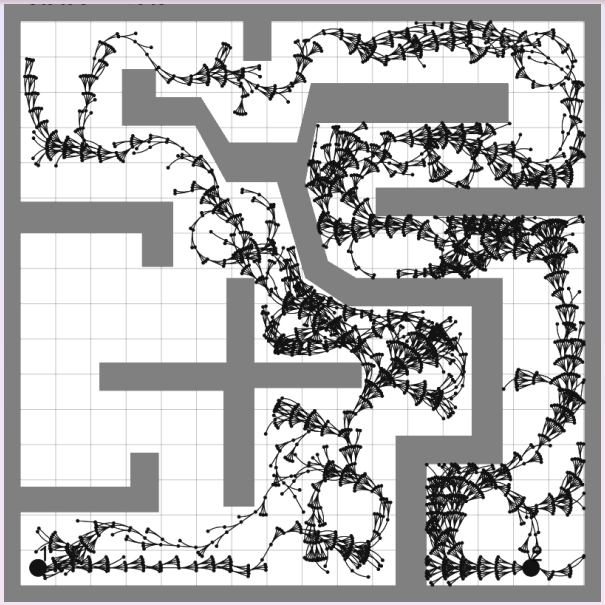


with filtering

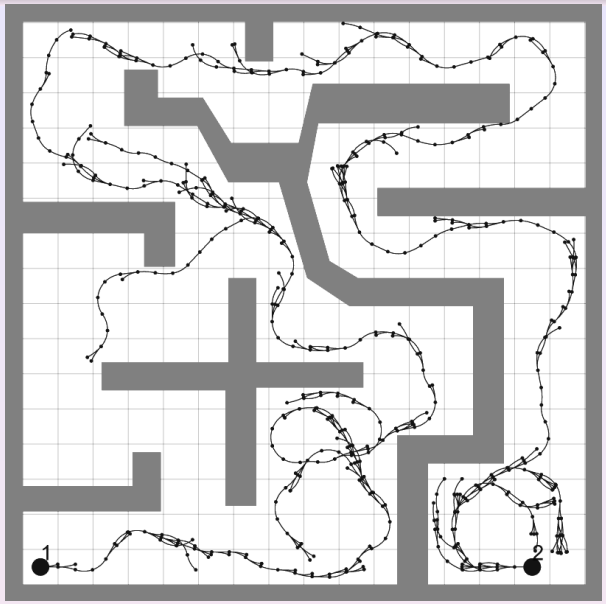


Tree structure comparison

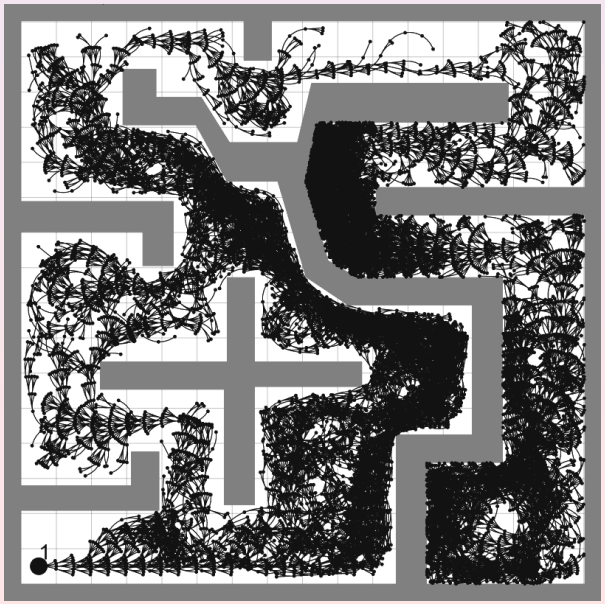
RRT-Blossom (plain)



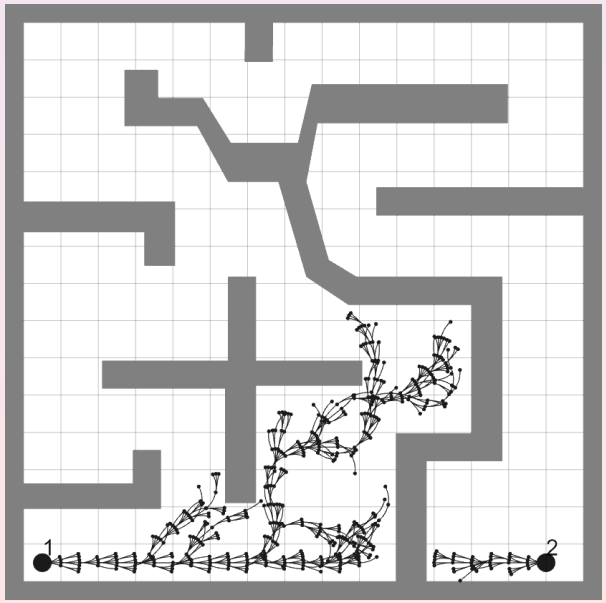
RRT-Blossom (filtered)



RRTExtExt-CT

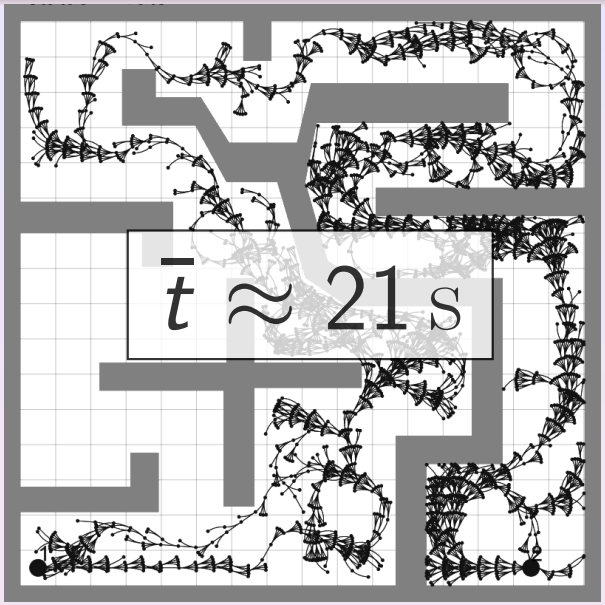


RRTExtExt

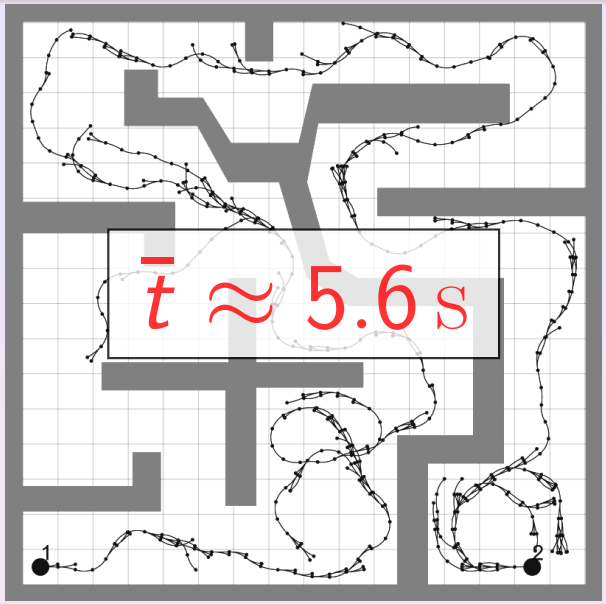


Tree structure comparison

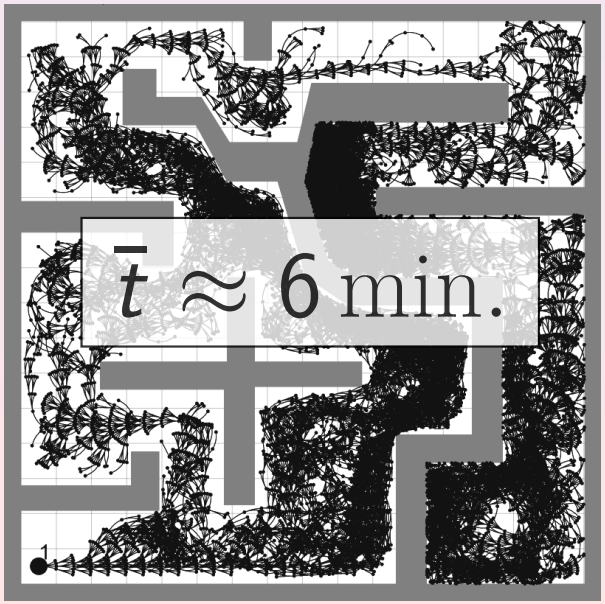
RRT-Blossom (plain)



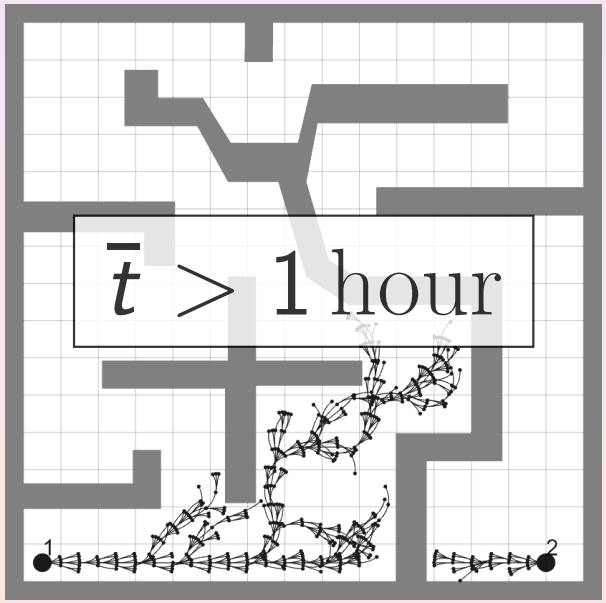
RRT-Blossom (filtered)



RRTExtExt-CT



RRTExtExt



Summary

Key points:

- current planners do not “see”, nor “learn” transferrable lessons
- limit planner to $Viab(\mathcal{X}_{free})$: same solutions, significant speed-up (e.g., 4x–10x)
- good results despite heavily imperfect models

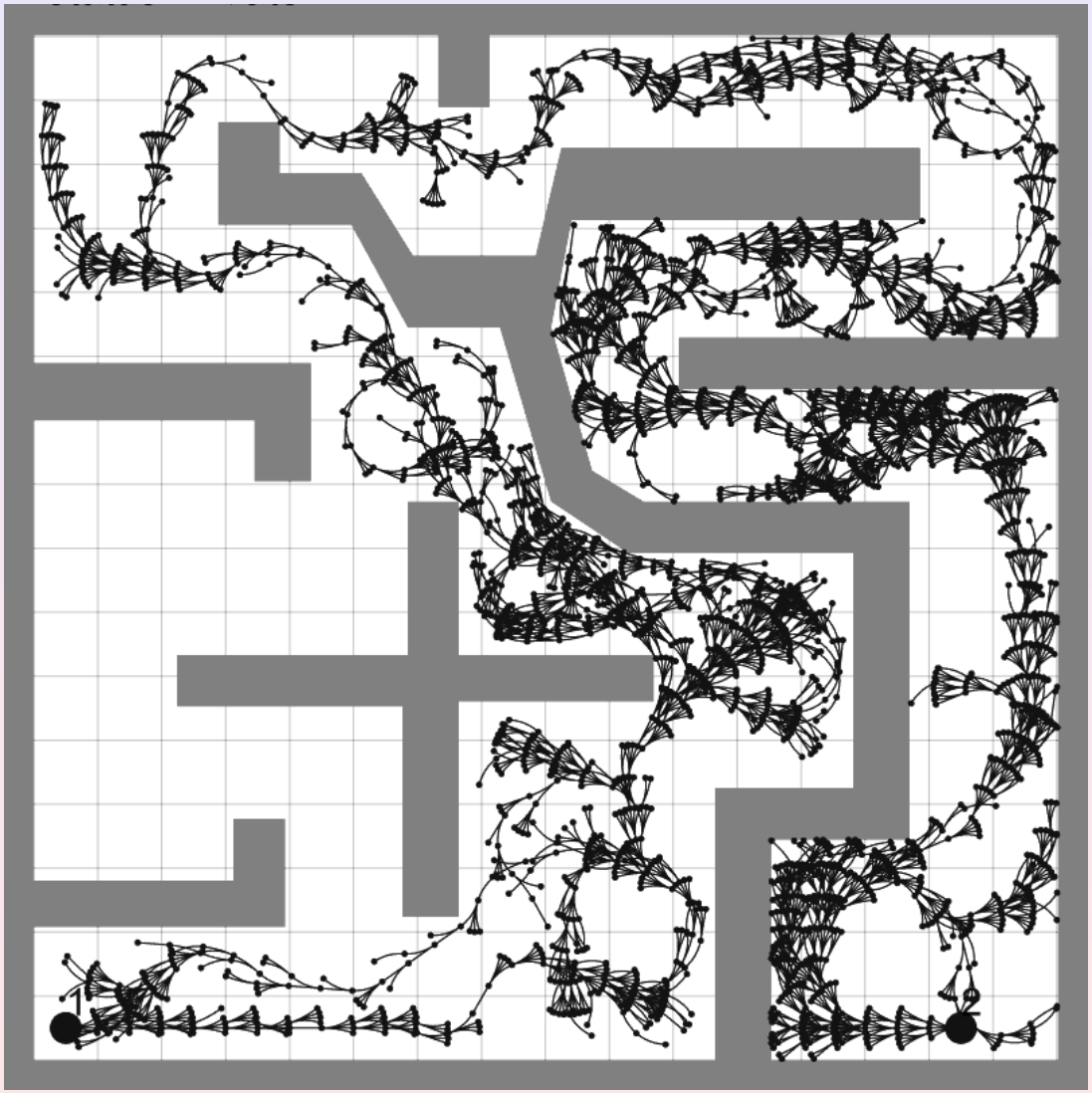
Additional information

- `http://www.dgp.toronto.edu/~mac/research/viability-filtering/`

Appendix

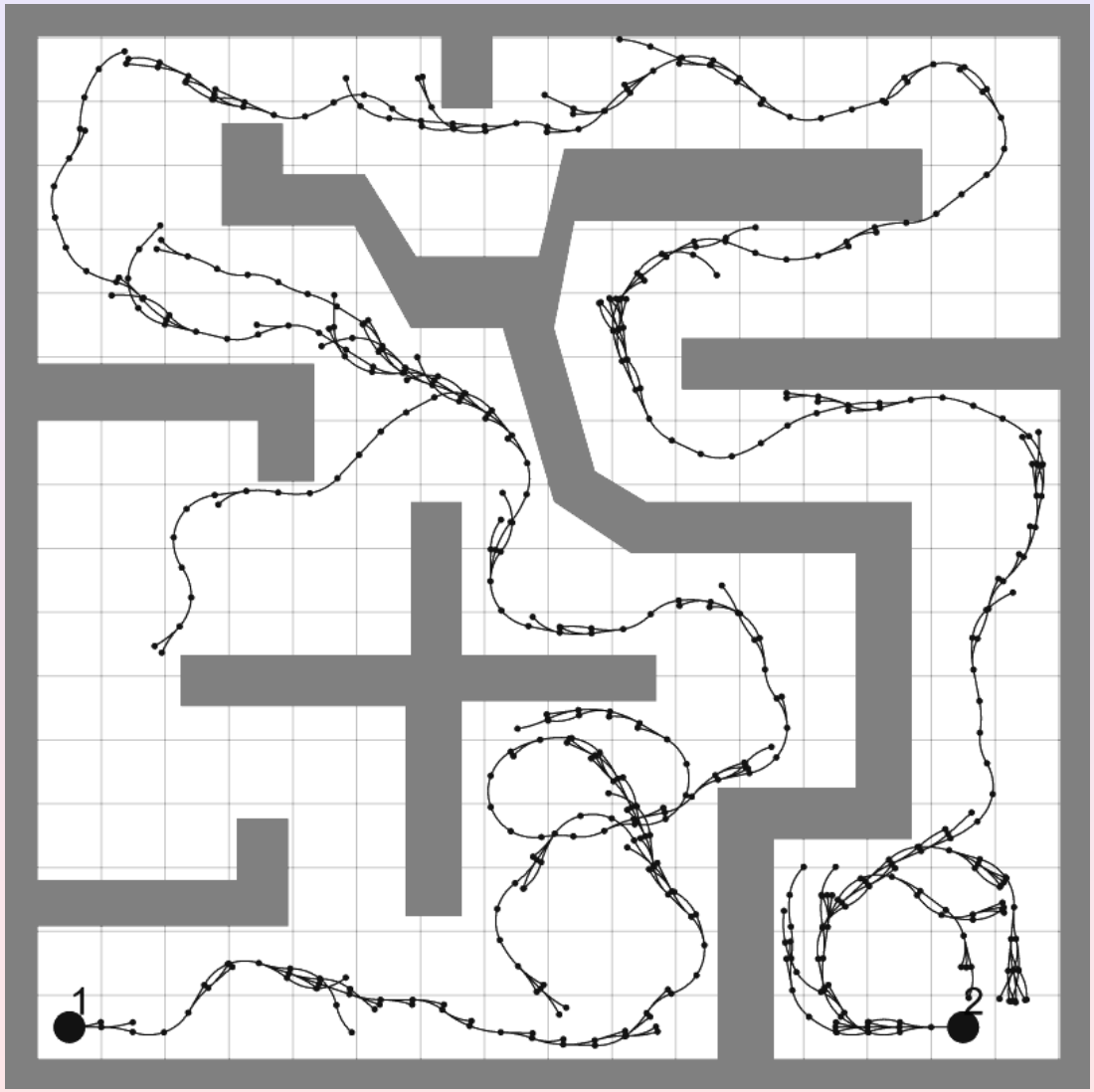
Tree structure comparison

RRT-Blossom (plain)

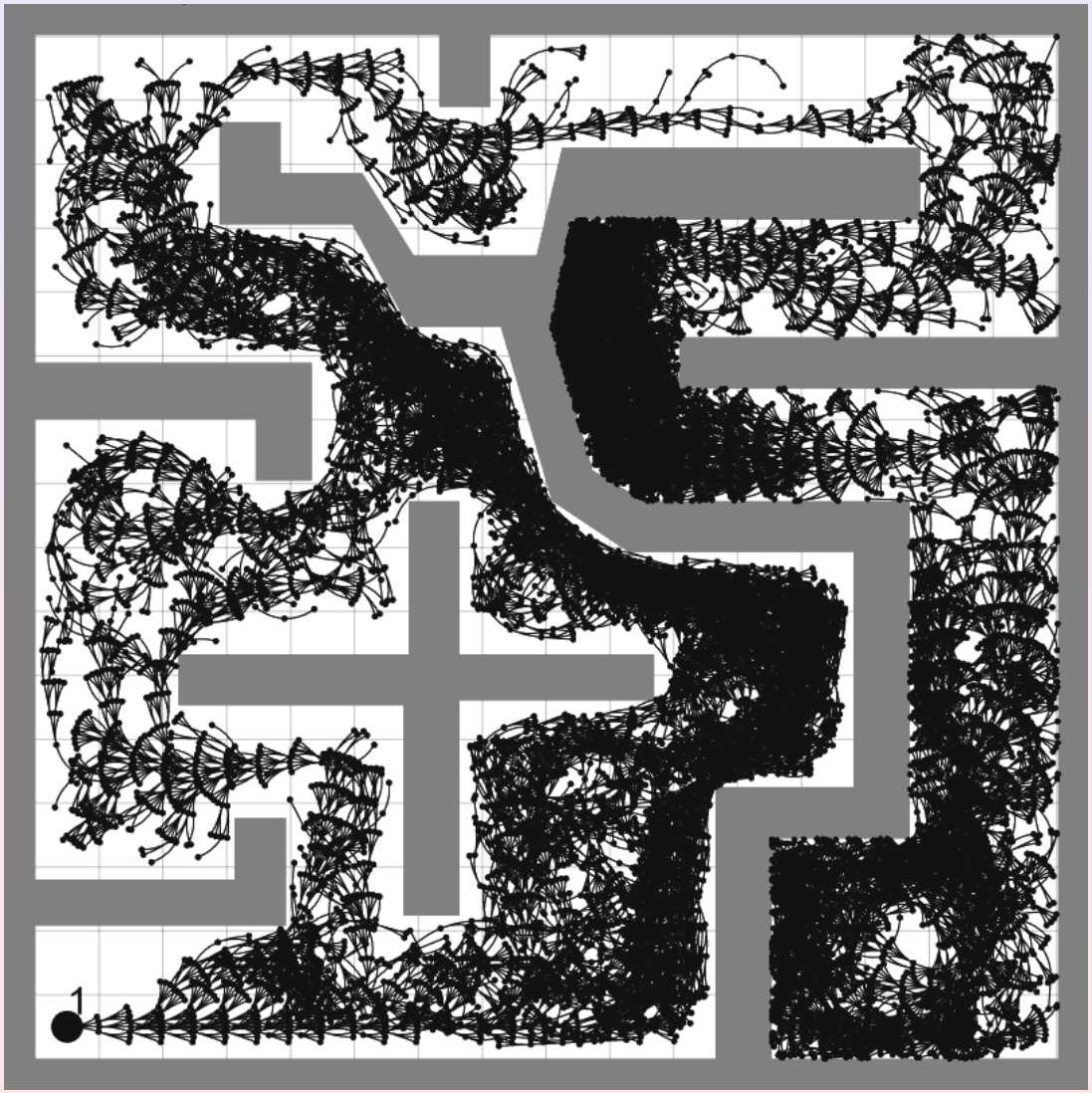


Tree structure comparison

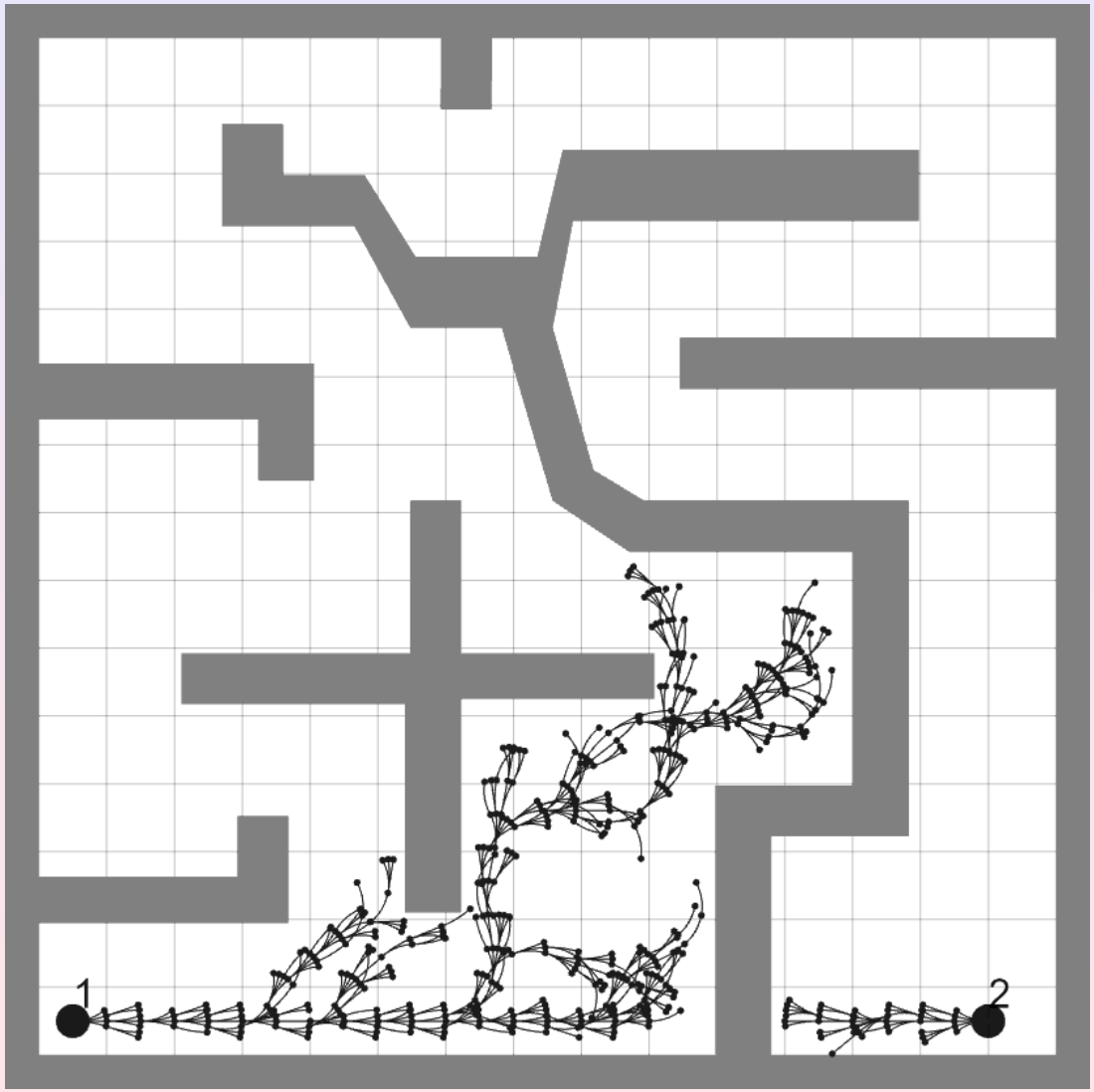
RRT-Blossom (viability-filtered)



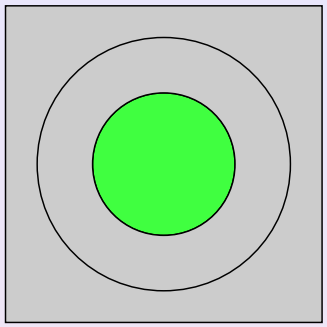
Tree structure comparison
RRT w/Collision Tendency (RRT-CT)



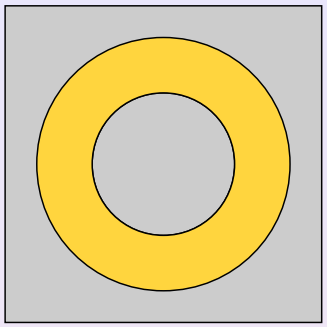
Tree structure comparison
plain RRT (RRTExtExt)



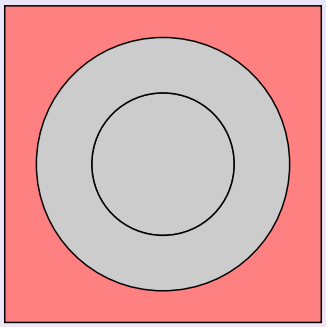
Viability vs. collision-checking



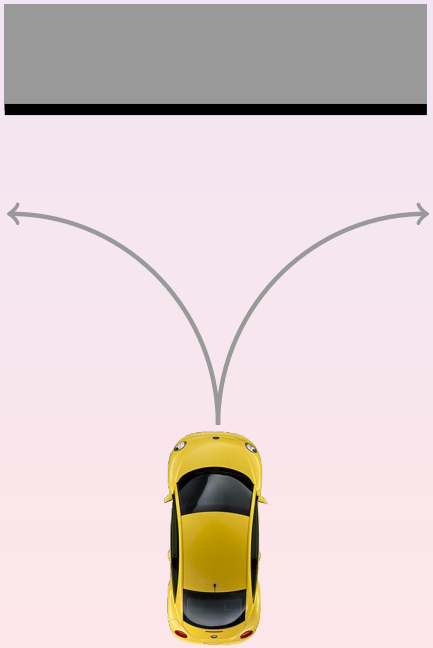
$x \in Viab(\mathcal{X}_{free})$

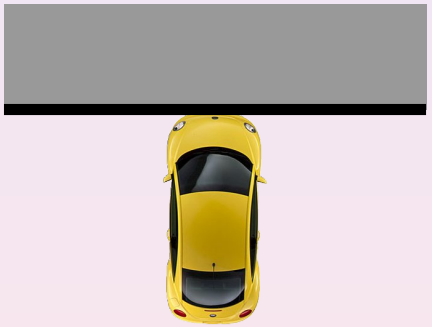


$x \in \mathcal{X}_{ric}$

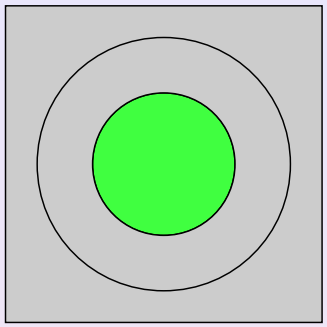


$x \in \mathcal{X}_{obst}$

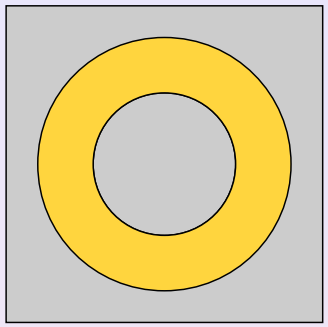




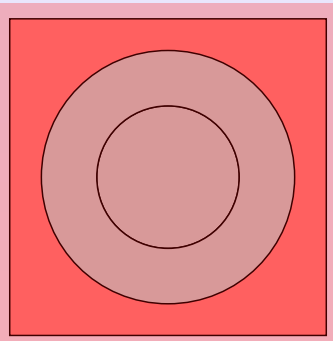
Viability vs. collision-checking



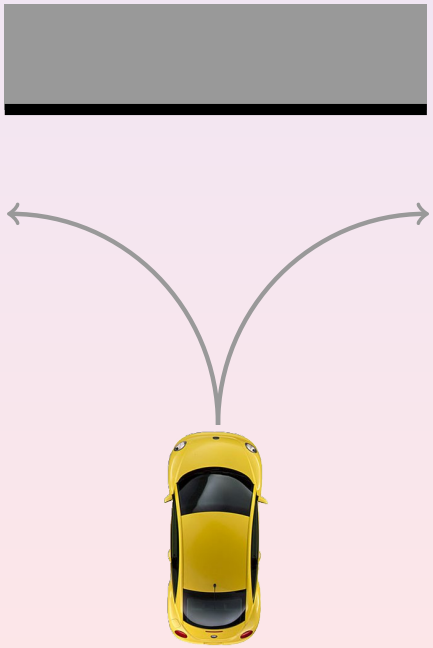
$x \in Viab(\mathcal{X}_{free})$

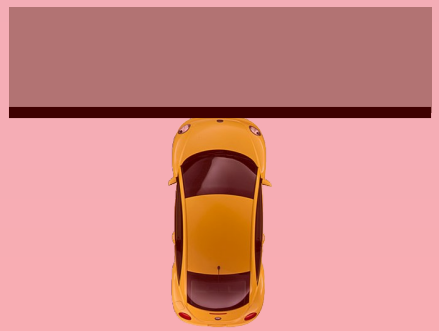


$x \in \mathcal{X}_{ric}$



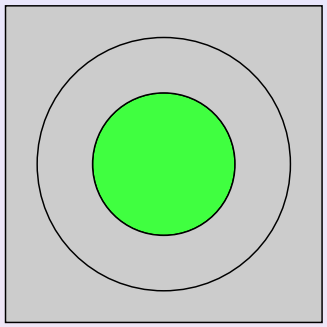
$x \in \mathcal{X}_{obst}$



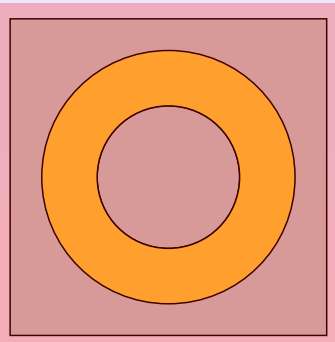


"collision filtering"

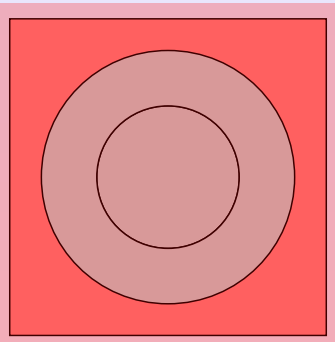
Viability vs. collision-checking



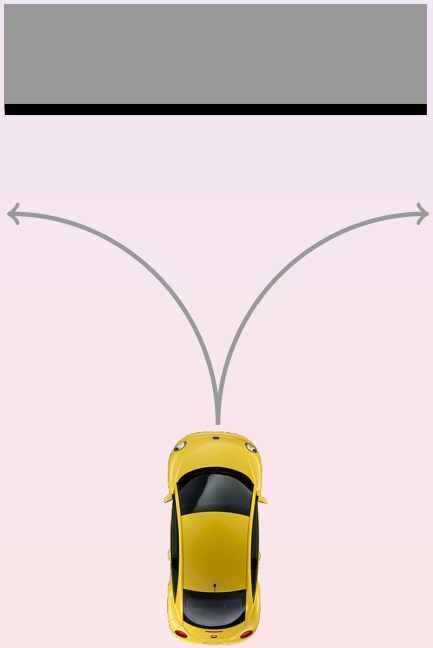
$x \in Viab(\mathcal{X}_{free})$

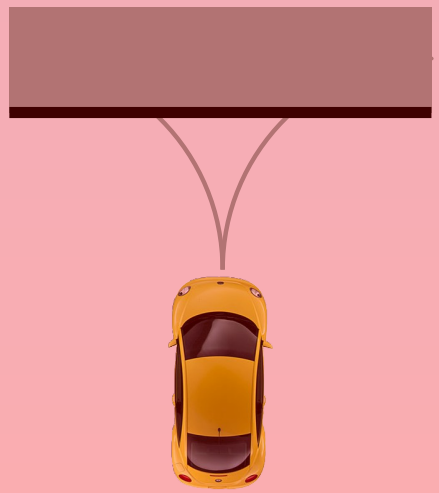


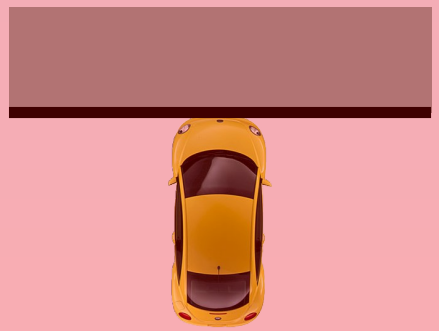
$x \in \mathcal{X}_{ric}$



$x \in \mathcal{X}_{obst}$







viability filtering