Planning Parking Maneuvers for a Car-Trailer Vehicle

Autonomous and Mobile Robotics

Prof. Giuseppe Oriolo

Supervisors: Ph.D. Tommaso Belvedere

Ph.D. Michele Cipriano

Leandro Maglianella - 1792507

Lorenzo Nicoletti - 1797464

Olga Sorokoletova - 1937430

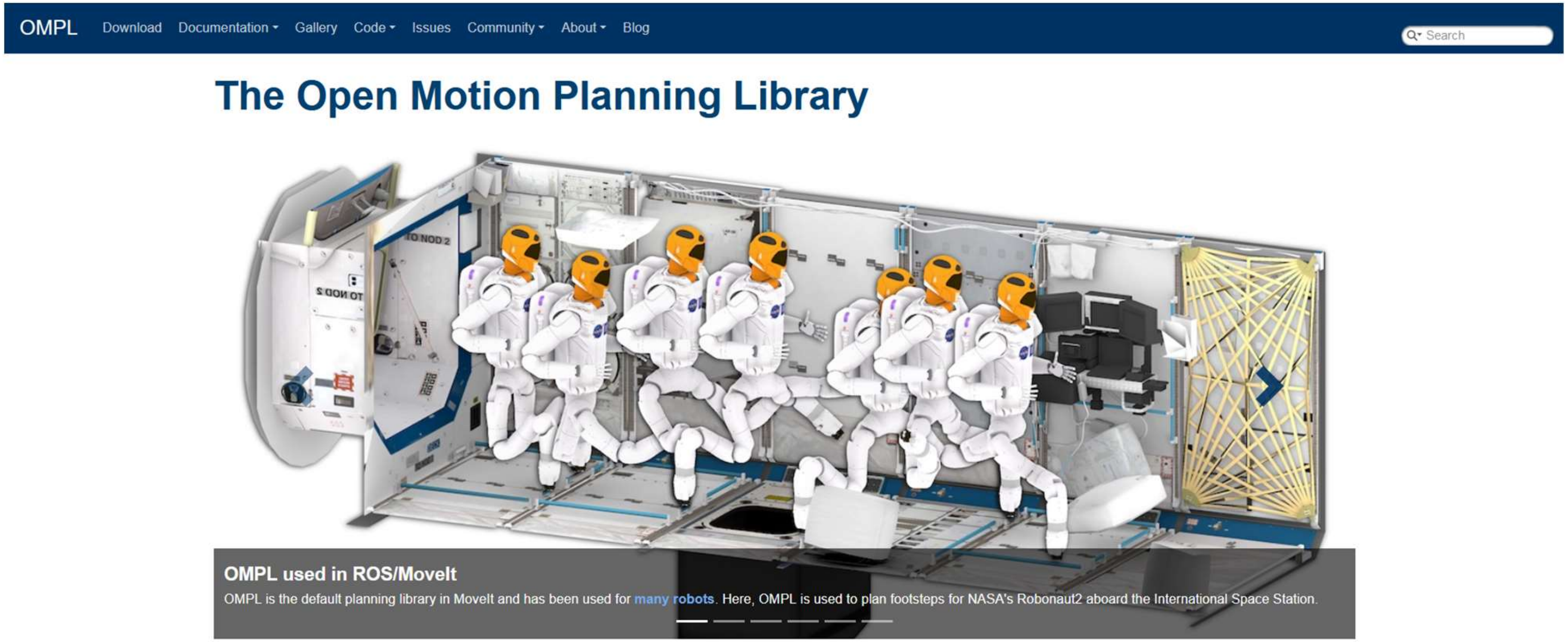
SAPIENZA
UNIVERSITÀ DI ROMA

Outline of the project

- **Project Description**
  - framework
  - car-trailer robot
  - jackknifing problem
  - experimental environments
  - baseline planner (RRT)
- **Optimal Planning**
  - optimal planner (SST)
  - optimization objectives
  - comparing results with baseline
- **Handling problem**
  - introducing obstacles
  - handling jackknifing (CL-RRT)
- **Conclusion**

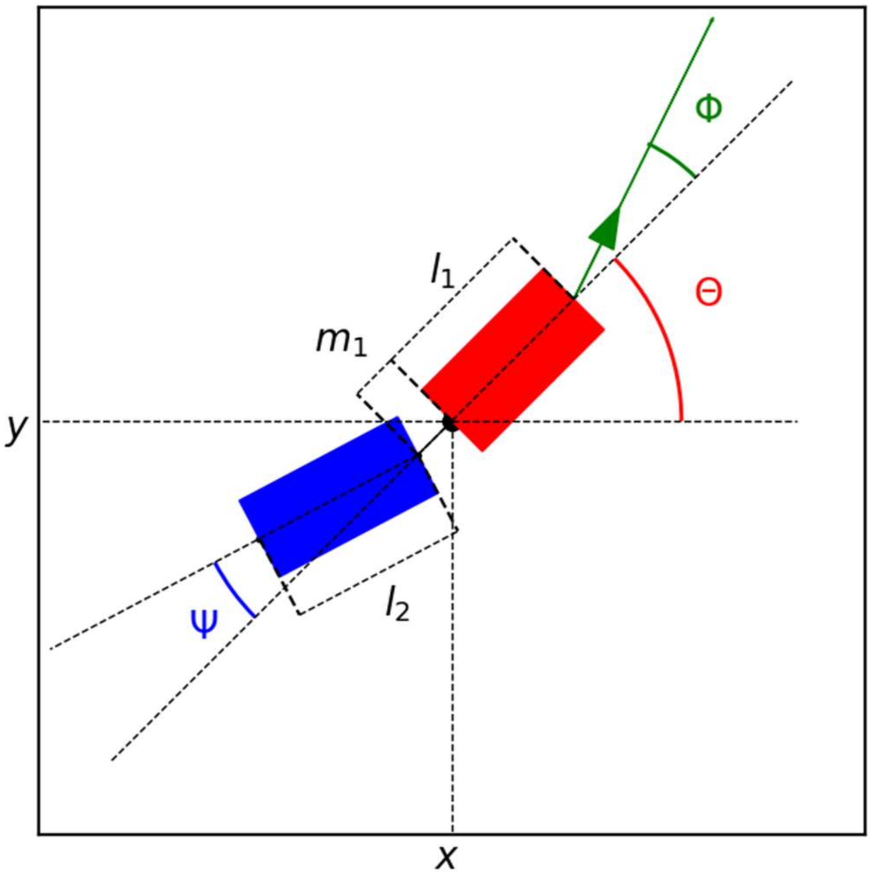
Project Description

Project Description: framework



- integrable into large variety of external systems
- environment specification, collision detection and visualization are left to the user

Project Description: car-trailer robot



Parameter	Value
l_1	0.25 m
l_2	0.26 m
m_1	0.07 m
$\lvert\psi_{max}\rvert$	45°
$\lvert\phi_{max}\rvert$	30°

State Space:

$$SE(2) \times SO(2) \times SO(2)$$

State:

$$q = \begin{bmatrix} x & y & \theta & \psi & \phi \end{bmatrix}^T$$

Kinematic model:

$$\dot{x} = v\cos\theta$$

$$\dot{y} = v\sin\theta$$

$$\dot{\theta} = \frac{v\tan\phi}{l_1}$$

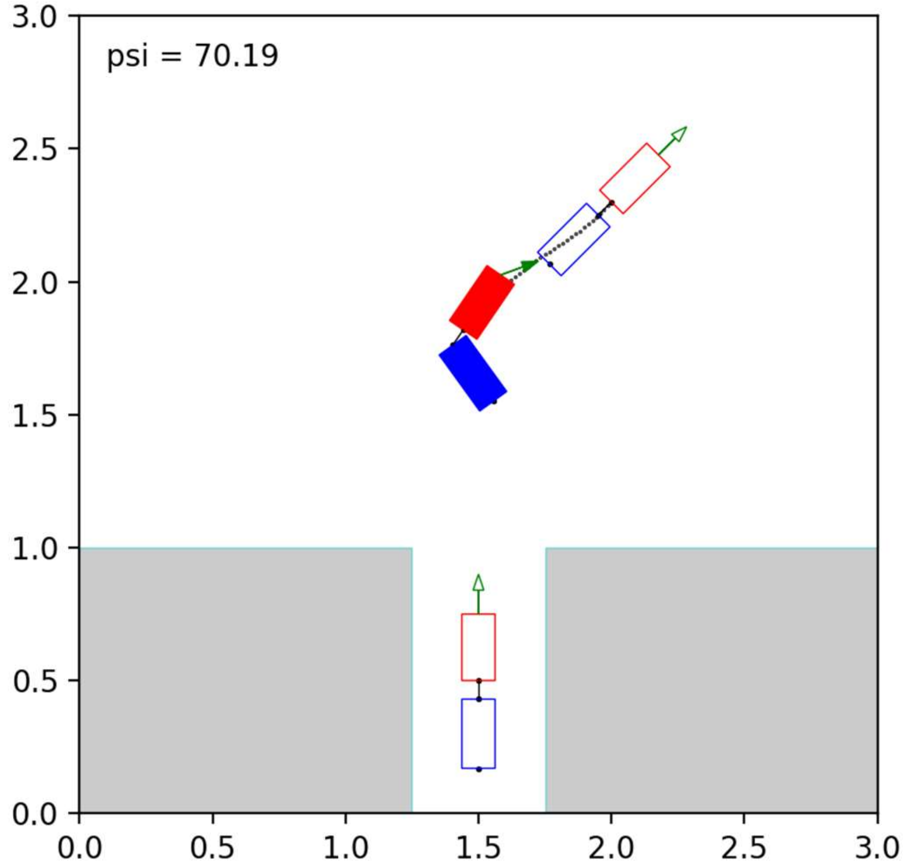
$$\dot{\psi} = -\frac{v\tan\phi}{l_1}\left(1 + \frac{m_1}{l_2}\cos\psi\right) - \frac{v\sin\psi}{l_2}$$

$$\dot{\phi} = \omega$$

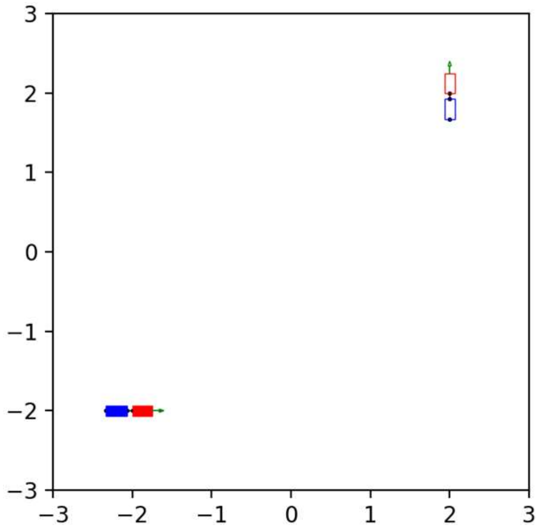
Project Description: jackknifing problem

Jackknifing phenomenon: while moving backward, the hitch angle can start to diverge (instability)

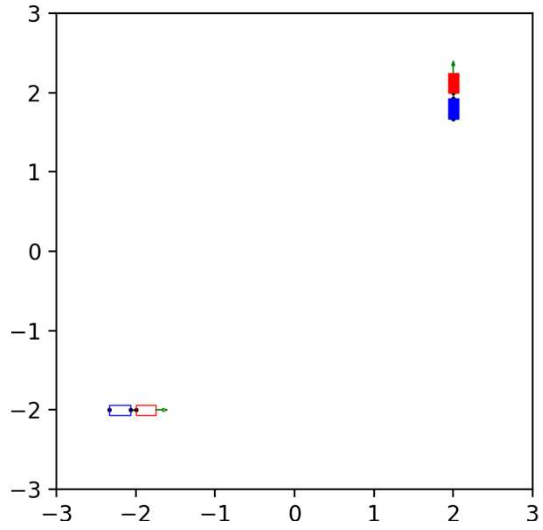
loss of maneuverability and **risk of self-collision**



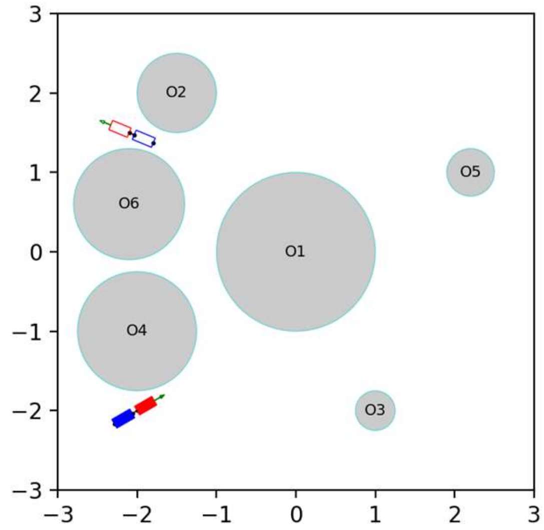
Project Description: experimental environments



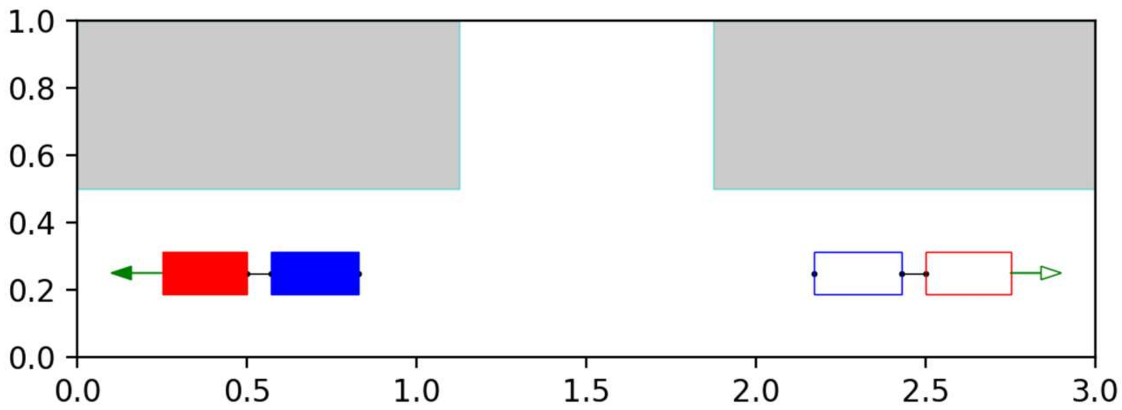
a) Simple forward motion



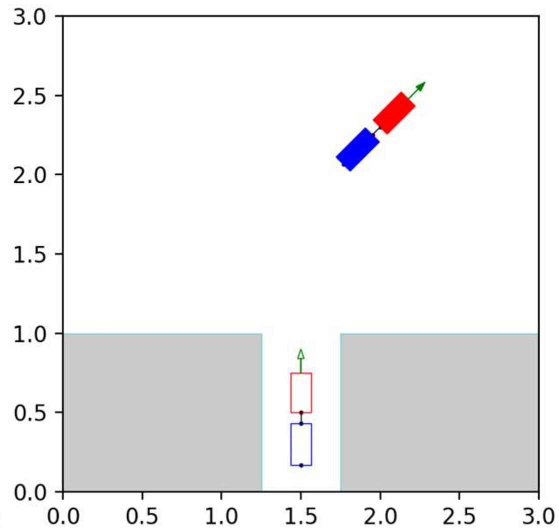
b) Simple backward motion



c) Circular obstacles avoidance



d) Three point turn



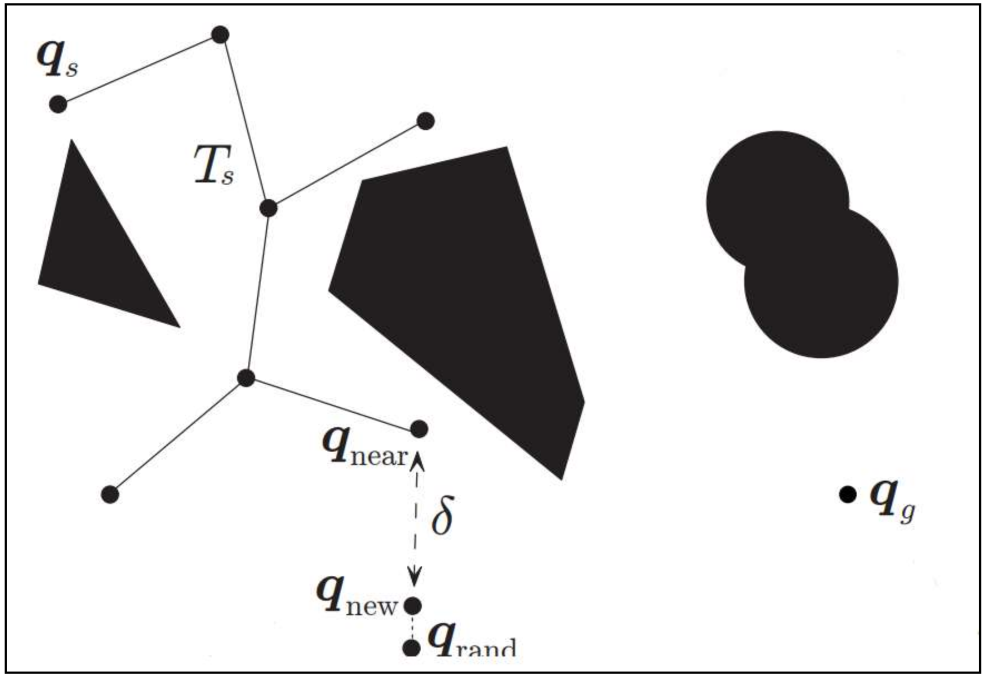
e) Real parking test

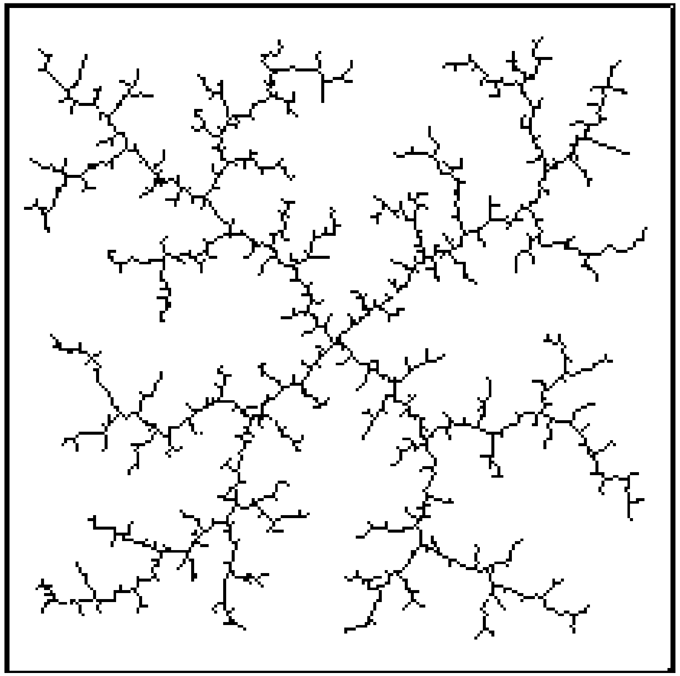
Project Description: baseline planner (RRT)

Rapidly-Exploring Random Tree (RRT) is a probabilistically complete sampling-based planning algorithm, which builds the exploration tree by performing the following steps:

- sampling of a random state in the configuration space
- searching for the closest amongst the visited states
- expansion until some new state is reached
- adding of a new state to the exploration tree, if no collisions

control inputs
driving velocity
steering velocity





Optimal Planning

SST planner

Optimal Planning: SST planner

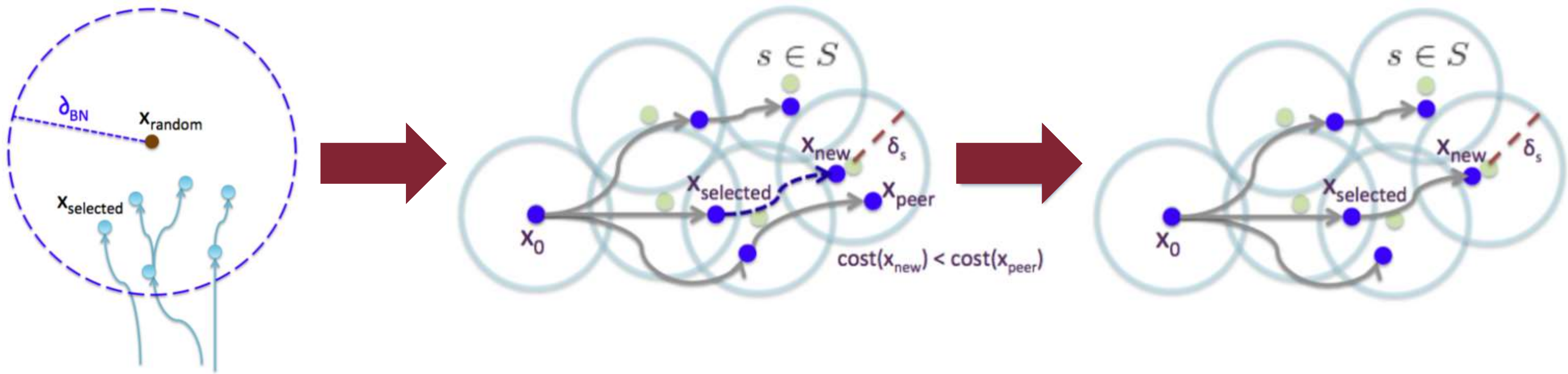
Stable Sparse RRT (SST) is an asymptotically near-optimal incremental modification of RRT, that enables optimal planning.

It can:

- keep the number of stored nodes small
- quickly converge to high-quality paths
- be combined with desired optimization objectives

control inputs
driving velocity
steering velocity

For N iterations, a **selection**/**propagation**/**pruning** procedure is followed.



Optimal Planning: explaining SST

- State space
- Control space
- Initial state
- Max propagation time
- Number of iterations
- Distance used for $x_{selected}$ choice
- Distance supervised by each witness s

Algorithm 1: STABLE_SPARSE_RRT ($\mathbb{X}$, $\mathbb{U}$, x_0, T_{prop}, N, δ_{BN}, δ_s)

$\mathbb{V}_{active} \leftarrow \{x_0\}, \mathbb{V}_{inactive} \leftarrow \emptyset;$
$G = \{V \leftarrow (\mathbb{V}_{active} \cup \mathbb{V}_{inactive}), \mathbb{E} \leftarrow \emptyset\};$
$s_0 \leftarrow x_0, s_0.rep = x_0, S \leftarrow \{s_0\};$
for N *iterations* **do**
  $x_{selected} \leftarrow Best\_First\_Selection\_SST(\mathbb{X}, \mathbb{V}_{active}, \delta_{BN});$
  $x_{new} \leftarrow MonteCarlo\_Prop(x_{selected}, \mathbb{U}, T_{prop});$
  if $CollisionFree(\overline{x_{selected} \rightarrow x_{new}})$ **then**
    if $Is\_Node\_Locally\_the\_Best\_SST(x_{new}, S, \delta_s)$ **then**
      $\mathbb{V}_{active} \leftarrow \mathbb{V}_{active} \cup \{x_{new}\};$
      $\mathbb{E} \leftarrow \mathbb{E} \cup \overline{x_{selected} \rightarrow x_{new}};$
      $Prune\_Dominated\_Nodes\_SST(x_{new}, \mathbb{V}_{active}, \mathbb{V}_{inactive}, \mathbb{E});$
return G;

V: tree nodes
E: tree edges
S: set of witnesses

Best node (already in the tree) to reach a randomly sampled x_{rand} in $\mathbb{X}$.

Node reached from $x_{selected}$ using the control sampled by MonteCarlo

Is the path to x_{new} safe?

Has x_{new} locally the best path cost?

- $\mathbb{V}_{active}$: set of nodes that in a neighborhood have the best path cost from the root.
- $\mathbb{V}_{inactive}$: set of dominated nodes that have children with good path cost in their neighborhoods (maintained on the tree for connectivity purposes).

Optimal Planning: optimization objectives

For **optimization problem**, formulated as: $\min_{v \in SearchTree} c(v)$

- **Path Length Minimization objective** optimizes the length of the path in order to avoid excessive wandering

$$c(v) = c(v.parent) + \|v - v.parent\|$$

- **Path Clearance Maximization objective** steers away from obstacles to efficiently increase safety

$$c(v) = c(v.parent) + \frac{1}{\gamma(q)}$$

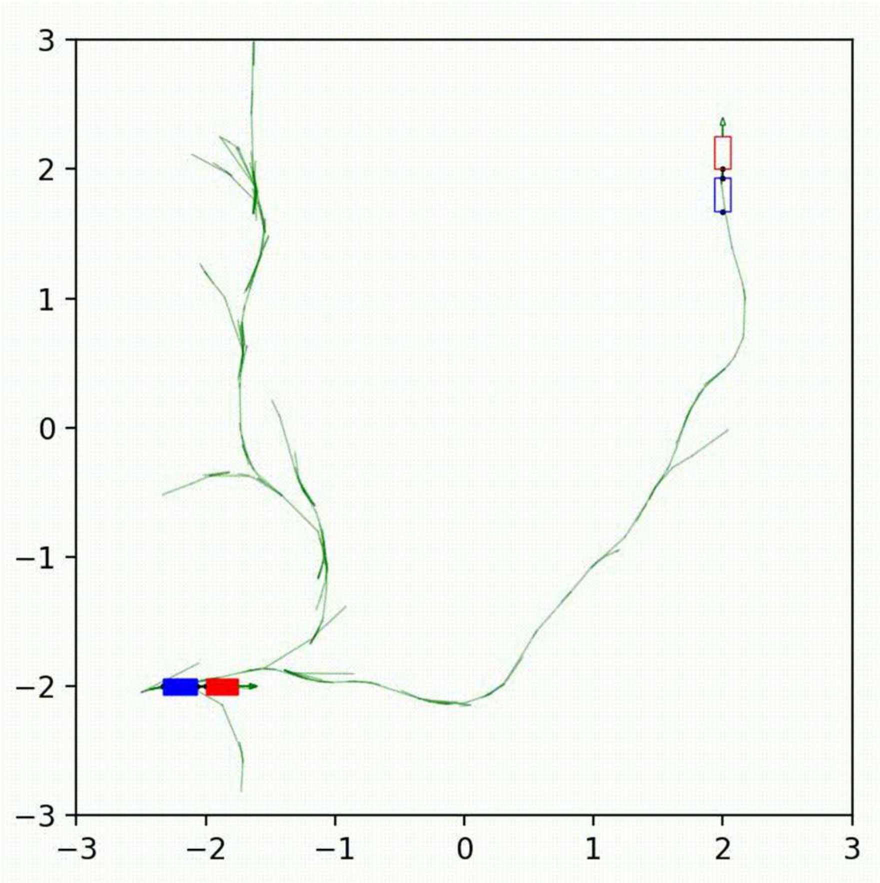
where

$$\gamma(q) = \min_{s \in \partial C_{free}} \|q - s\|$$

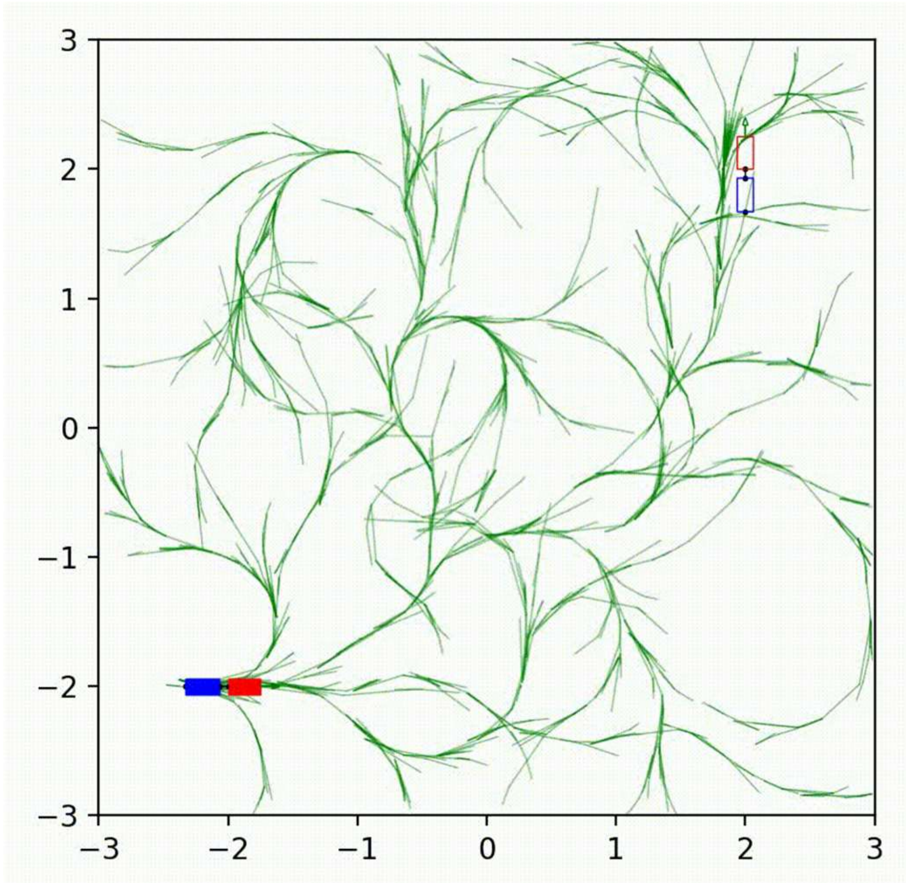
Optimal Planning

Compare results with baseline planner

Compare results: simple forward motion



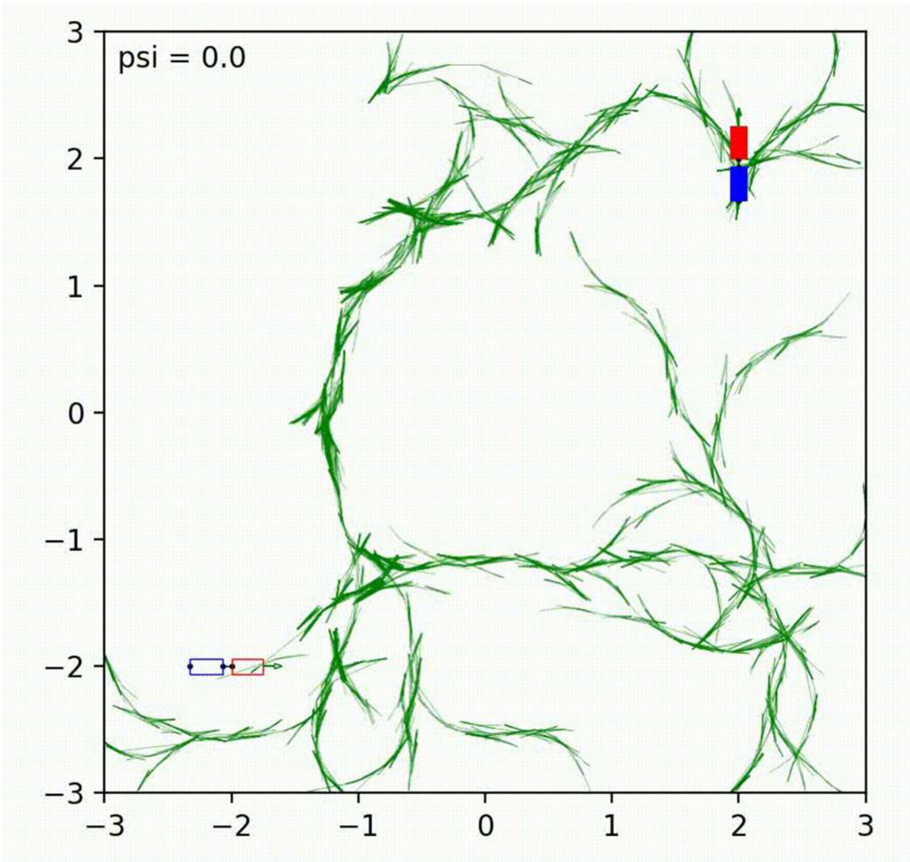
RRT



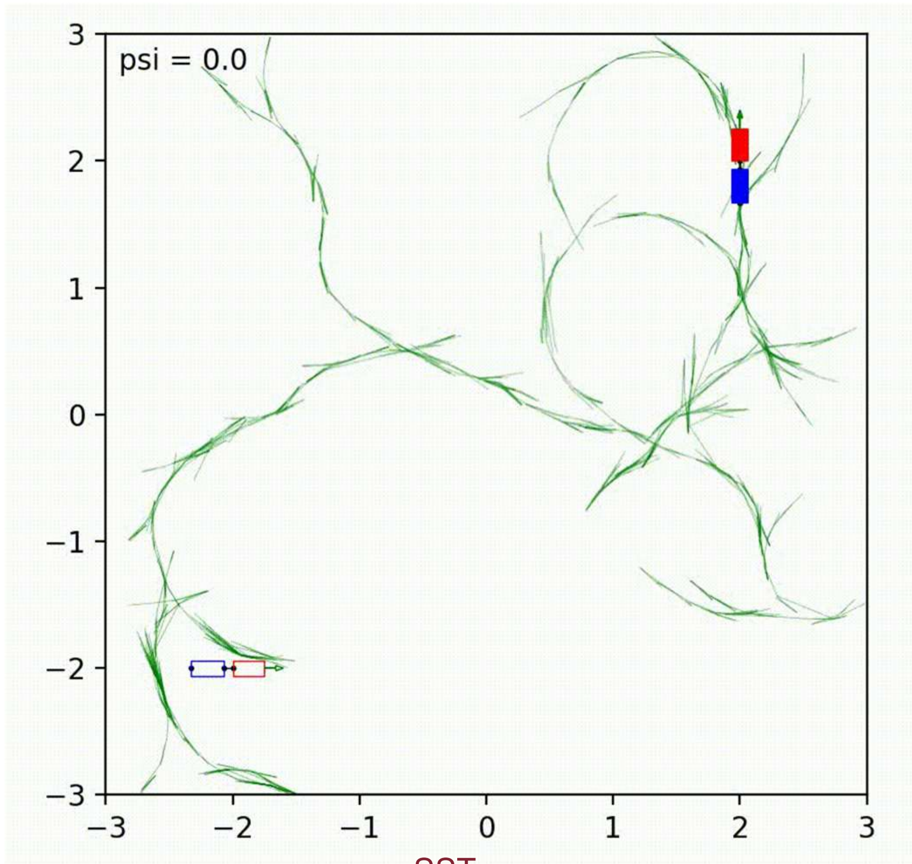
SST

Planner	%	Avg number of states			Avg length			Avg time
		exact	approx.	total	exact	approx.	total	
RRT	**58**	23948	84691	49460	24.9	26.7	25.7	**8.89**
SST	54	9456	25962	**17049**	23.6	24.8	**24.2**	14.51

Compare results: simple backward motion



RRT



SST

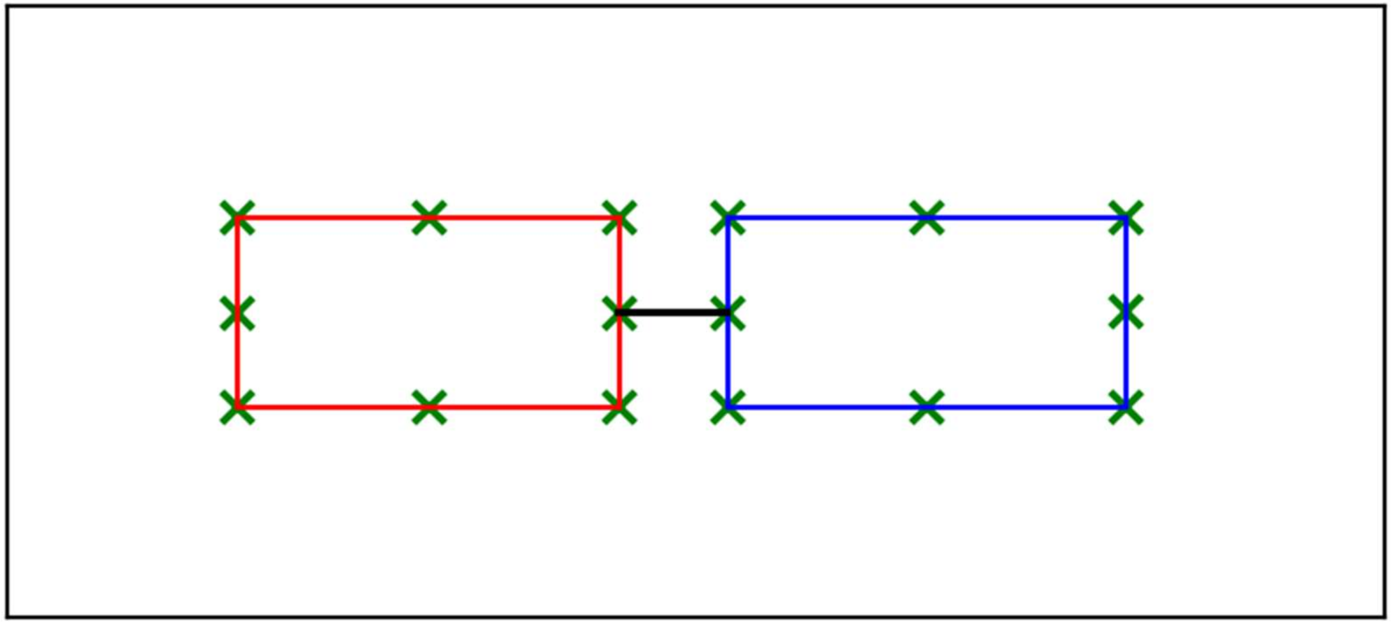
Planner	%	Avg number of states			Avg length			Avg time
		exact	approx.	total	exact	approx.	total	
RRT	**46**	31096	60313	46873	57.1	57.7	57.4	**16.93**
SST	44	8062	14662	**11758**	54.0	57.0	**55.7**	20.46

Handling problem

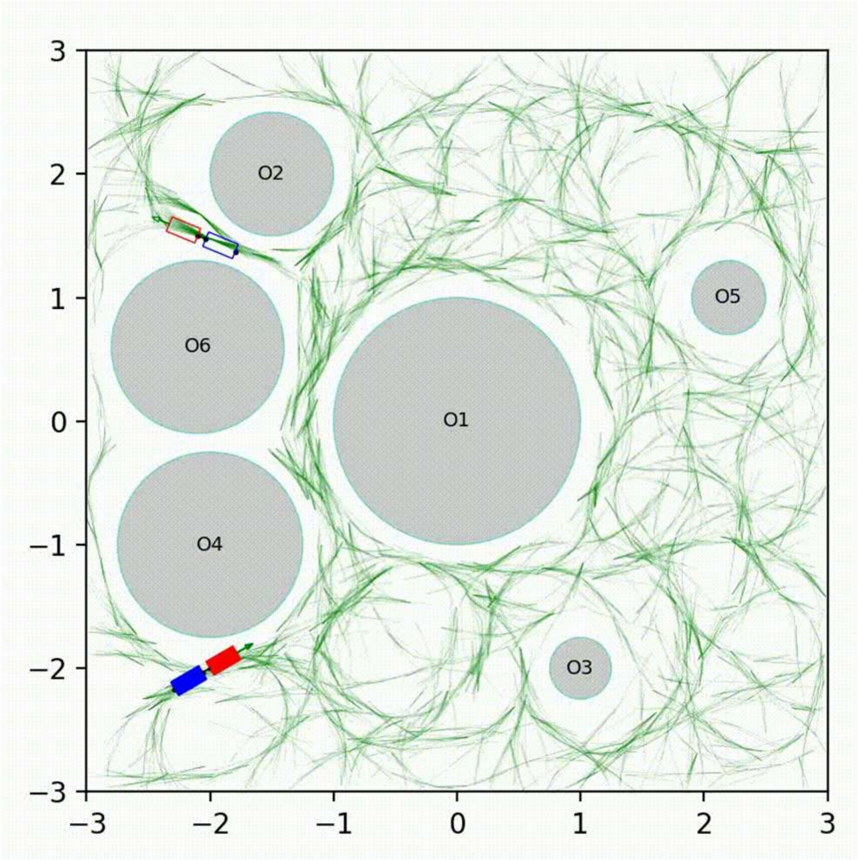
Introducing obstacles

Introducing obstacles: collision checker

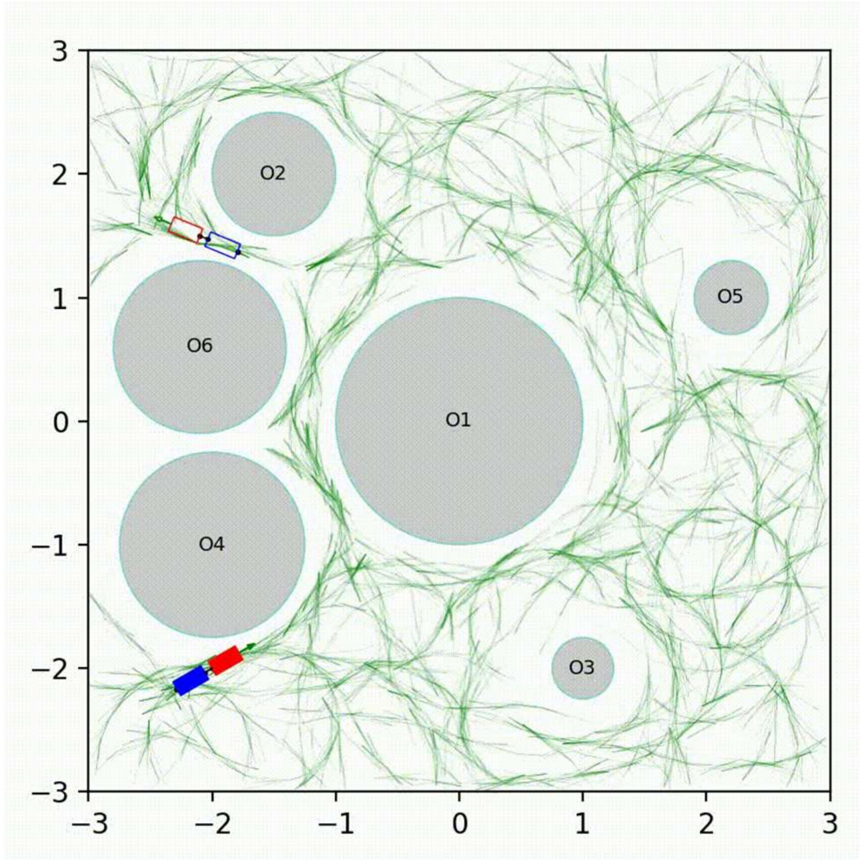
Collision-checking is performed considering a finite set of control points, defined along the robot's body:



Introducing obstacles: circular obstacles avoidance



SST Path Length



SST Path Clearance

Planner	%	Avg number of states			Avg length			Avg time
		exact	approx.	total	exact	approx.	total	
SST Length	92	4301	15663	5210	34.3	47.5	35.4	9.30

Handling problem

Handling jackknifing

Handling jackknifing: CL-RRT

Closed-Loop RRT (CL-RRT) is a modification of RRT which aims to overcome the problem of the hitch angle divergence by countersteering before reaching a jackknife state.

CL-RRT's underlined idea is to perform the cascaded control by means of introduction of a closed-loop dynamics.

Sampled desired steering angle is used to compute the reference steering angle:

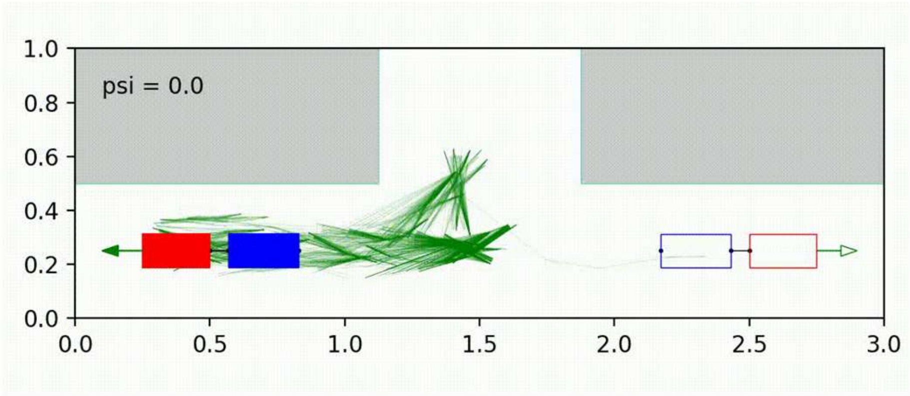
$$\phi_r = \phi_d - K_{stab}\psi$$

control inputs
driving velocity
desired steering angle

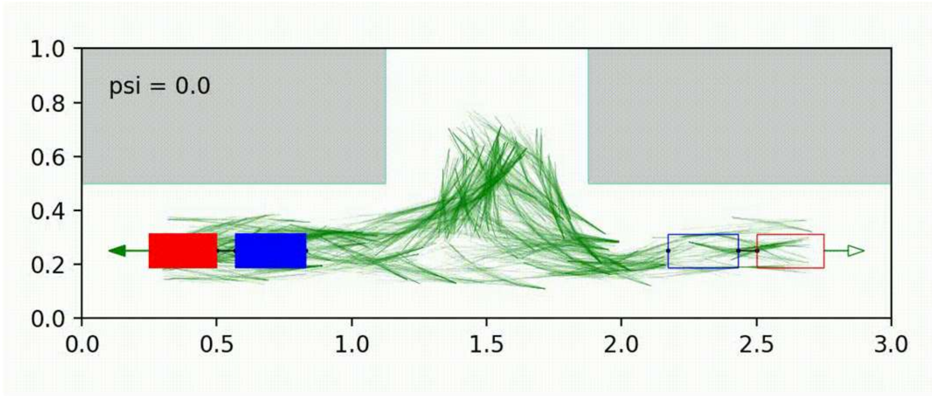
Then the steering velocity is used to track such reference quantity using the standard proportional feedback control:

$$\dot{\phi} = \omega = K_{reg}(\phi_r - \phi)$$

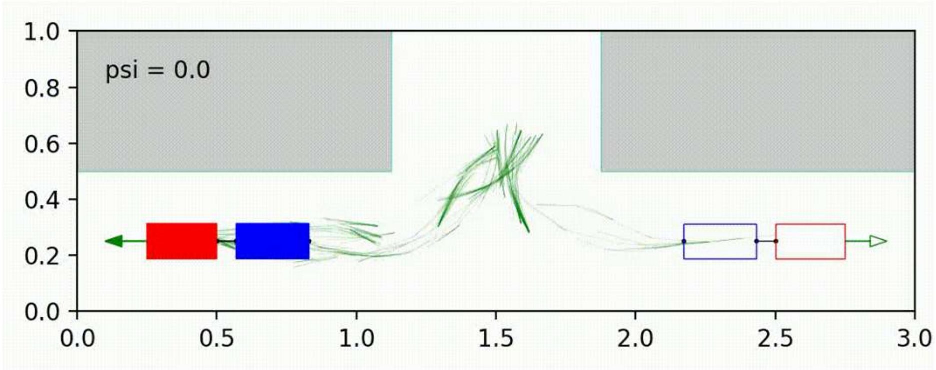
Handling jackknifing: three-point turn



RRT



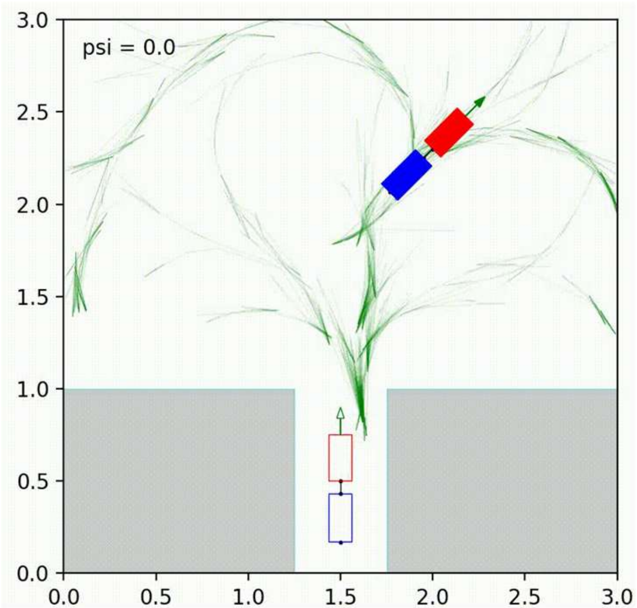
SST



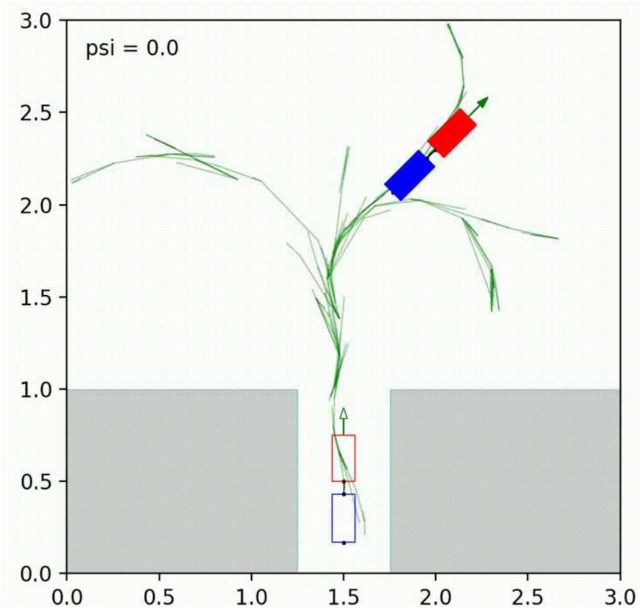
CL-RRT

Planner	%	Avg number of states			Avg length			Avg time
		exact	approx.	total	exact	approx.	total	
RRT	76	19865	53277	27884	22.8	23.1	22.9	14.70
SST	26	777	1380	**1223**	18.1	14.4	**15.4**	20.56
CL-RRT	**92**	13084	56093	16524	24.4	21.5	24.2	**8.87**

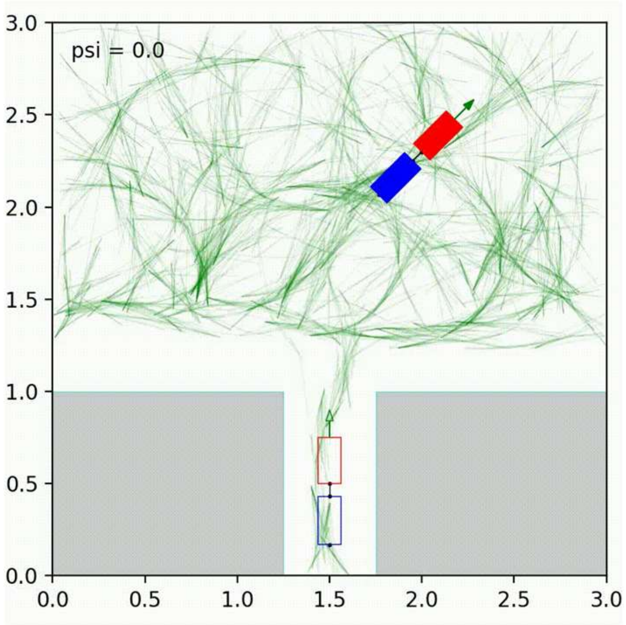
Handling jackknifing: real parking test



RRT



CL-RRT



SST

Planner	%	Avg number of states			Avg length			Avg time
		exact	approx.	total	exact	approx.	total	
RRT	30	25261	61613	50707	23.2	22.3	22.6	11.43
SST	36	4173	11036	**8565**	20.8	18.7	**19.5**	14.12
CL-RRT	**82**	23266	65609	30888	26.5	31.9	27.5	**10.97**

Conclusion

Conclusion

- we developed a **programming pipeline that provides OMPL's omitted components** to perform motion planning with a car-trailer vehicle
- all the three planners (RRT, SST, CL-RRT) have been tested and evaluated in a wide set of experiments and scenarios in order to **fully explore their potentialities**
- the **optimal planning** introduced with SST has allowed to achieve **higher-quality paths** and **better performances** with respect to the baseline RRT
- the **jackknifing phenomenon has been efficiently handled** in increasing-complexity environments
- the most satisfactory results were obtained with the introduction of the **Closed-Loop** in the dynamic system
- the project can be further extended to include **new objectives**, **N-trailer structures** and additional **CL-based planners**.

References

1. **Evestedt N., Ljungqvist O., Axehill D.**
   "Motion planning for a reversing general 2-trailer configuration using Closed-Loop RRT", IROS 2016

2. **Beglini M., Lanari L., Oriolo G.**
   "Anti-Jackknifing Control of Tractor-Trailer Vehicles", 2020 IEEE International Conference on Robotics and Automation (ICRA)

3. **Open Motion Planning Library**
   http://ompl.kavrakilab.org/

4. **Li Y., Littlefield Z., Bekris K. E.**
   "Asymptotically Optimal Sampling-based Kinodynamic Planning", 2016

Appendix

Environment	Boundaries		q_s					q_g				
	$[x_l, x_h]$	$[y_l, y_h]$	x	y	θ	ψ	ϕ	x	y	θ	ψ	ϕ
Simple Forward	$[-3,3]$	$[-3,3]$	-2	-2	0	0	0	2	2	$\frac{\pi}{2}$	0	0
Simple Backward	$[-3,3]$	$[-3,3]$	2	2	$\frac{\pi}{2}$	0	0	-2	-2	0	0	0
Circular Obstacles	$[-3,3]$	$[-3,3]$	-2	-2	$\frac{\pi}{6}$	0	0	-2.1	1.5	$\frac{7\pi}{8}$	0	0
Three-point Turn	$[0,3]$	$[0,1]$	0.5	0.25	$-\pi$	0	0	2.5	0.25	0	0	0
Real Parking	$[0,3]$	$[0,3]$	2	2.3	$\frac{\pi}{4}$	0	0	1.5	0.5	$\frac{\pi}{2}$	0	0

Environmental parameters: state space bounds for the Cartesian components, initial and goal configuration.

Environment	Planner	v		ω / ϕ_d		ϵ	Bias	Step	Dur.	Time
		min	max	min	max					
S. Forward	**RRT**	-0.25	0.5	-1	1	0.2	0.3	0.1	$(1,10)$	45
	SST									
S. Backward	**RRT**	-0.5	0.25	-1	1	0.3				
	SST									
Circular O-s	**SST Min**	-0.5	0.5	-1	1	0.25				
	SST Max									
3-point Turn	**RRT**	-0.5	0.5	-1	1	0.25				
	SST			-1	1					
	CL-RRT			-25	25					
Real Parking	**RRT**	-0.5	0.5	-1	1	0.25				
	SST			-1	1					
	CL-RRT			-25	25					

Hyper-parameters: driving velocity, steering velocity, goal threshold ϵ, goal bias, propagation step size, control duration and maximum time limit to return an exact solution. SST Min and SST Max stand for SST with Minimum Path Length objective and SST with Maximum Path Clearance objective, respectively.

Appendix

Algorithm 3: `MonteCarlo-Prop`$(x_{prop}, \mathbb{U}, T_{prop})$

$t \leftarrow$ Sample$(0, T_{prop})$; $\Upsilon \leftarrow$ Sample$(\mathbb{U}, t)$;

return $x_{new} \leftarrow \int_0^t f(x(t), \Upsilon(t))\, dt + x_{prop}$;

Algorithm 6: `Best_First_Selection_SST`$(\mathbb{X}, \mathbb{V}, \delta_{BN})$

$x_{rand} \leftarrow$ `Sample_State`$(\mathbb{X})$;

$X_{near} \leftarrow$`Near`$(\mathbb{V}, x_{rand}, \delta_{BN})$;

If $X_{near} = \emptyset$ **return** `Nearest`$(\mathbb{V}, x_{rand})$;

Else return $\arg\min_{x \in X_{near}} cost(x)$;

Algorithm 7: `Is_Node_Locally_the_Best_SST`(x_{new}, S, δ_s)

$s_{new} \leftarrow$ `Nearest`(S, x_{new});

if $||x_{new} - s_{new}|| > \delta_s$ **then**

    $S \leftarrow S \cup \{x_{new}\}$;

    $s_{new} \leftarrow x_{new}$;

    $s_{new}.rep \leftarrow NULL$;

$x_{peer} \leftarrow s_{new}.rep$;

if $x_{peer} == NULL$ *or* `cost`$(x_{new}) <$ `cost`(x_{peer}) **then**

    return true;

return false;

Appendix

Algorithm 8: `Prune_Dominated_Nodes_SST`(x_{new}, $\mathbb{V}_{active}$, $\mathbb{V}_{inactive}$, $\mathbb{E}$)

$s_{new} \leftarrow$ `Nearest`(S,x_{new});
$x_{peer} \leftarrow s_{new}.rep$;
if $x_{peer}! = NULL$ **then**
  $\mathbb{V}_{active} \leftarrow \mathbb{V}_{active} \setminus \{x_{peer}\}$;
  $\mathbb{V}_{inactive} \leftarrow \mathbb{V}_{inactive} \cup \{x_{peer}\}$;
$s_{new}.rep \leftarrow x_{new}$;
while $x_{peer}! = NULL$ **and** `IsLeaf` (x_{peer}) **and** $x_{peer} \in \mathbb{V}_{inactive}$ **do**
  $x_{parent} \leftarrow$ `Parent`(x_{peer});
  $\mathbb{E} \leftarrow \mathbb{E} \setminus \{\overline{x_{parent} \rightarrow x_{peer}}\}$;
  $\mathbb{V}_{inactive} \leftarrow \mathbb{V}_{inactive} \setminus \{x_{peer}\}$;
  $x_{peer} \leftarrow x_{parent}$;