Génération de trajectoires avec contraintes temporelles

a.k.a. SCvx-STL

Danil BERRAH

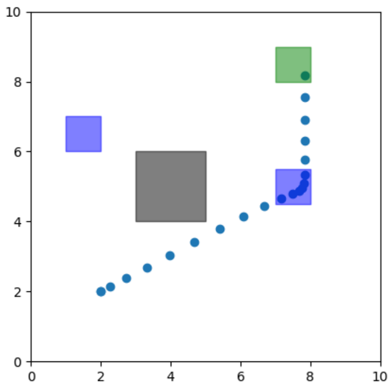
Advised by Goran FREHSE, Alexandre CHAPOUTOT, and Pierre-Loïc GAROCHE
ENSTA Paris

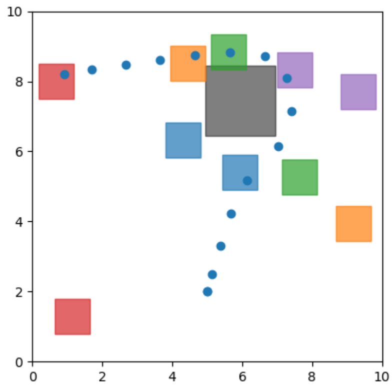
November 9, 2023

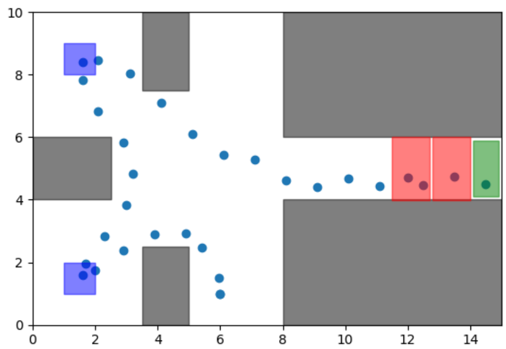
Motion planning with mission-based feature

Examples of behaviors of robots (w.r.t. dynamics) and mission planning

- two targets
- many targets
- hierarchy in targets

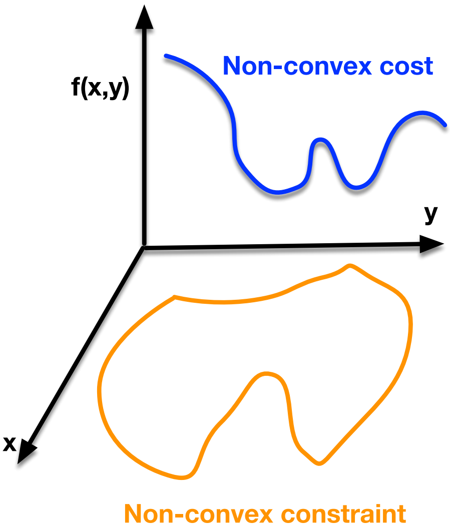




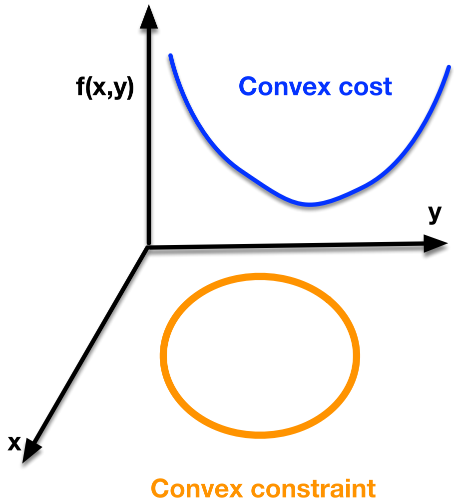


Summary

Why convexity matters?



Convexification
$\Longrightarrow$



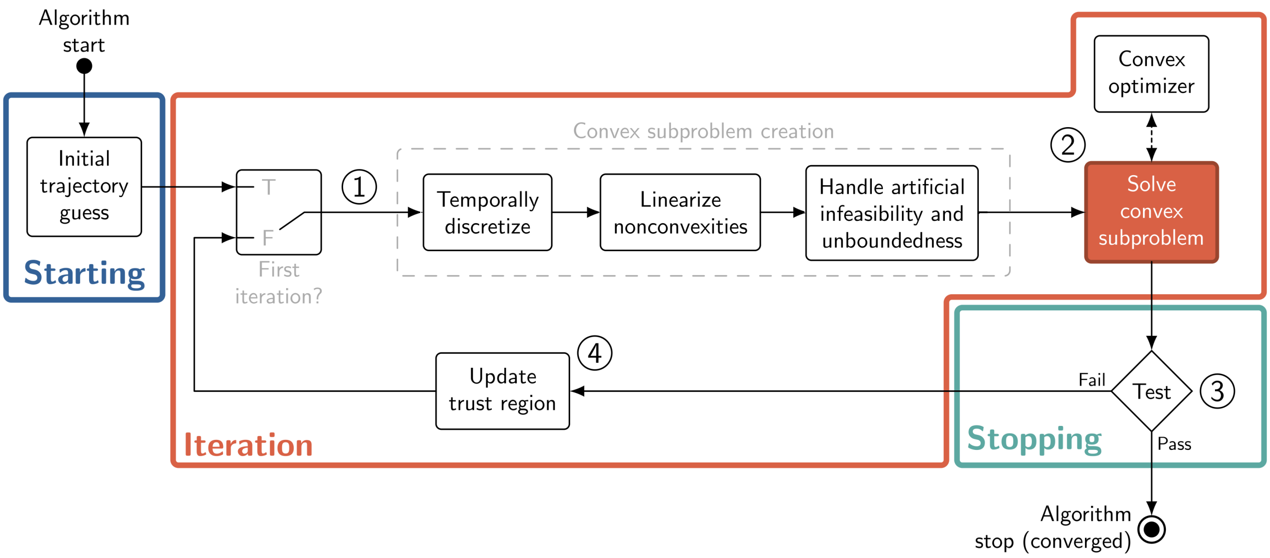
- No guarantees of convergence or complexity

Requires expert in the loop

- Guaranteed global optimum
- Polynomial-time complexity

No human in the loop needed

SCvx Algorithm: Overview[1]



Input: Non-convex problem and number of sample points

1. Choose an initial trajectory (*e.g.*, straight line)
2. Iteratively solve convex sub-problems until convergence

Output: a sequence of controls or a failure

[1] Convex Optimization for Trajectory Generation. Danylo Malyuta *et al.*, 2022

Convexification

Discrete-time Convex Subproblem

$$\min_{x,u,\hat{v}} \quad \mathcal{L}(x,u,\hat{v}) \tag{1a}$$

subject to

$$x_{k+1} = A_k x_k + B_k u_k + r_k + E_k v_k, \qquad \text{Linearized/discretized dynamics} \tag{1b}$$

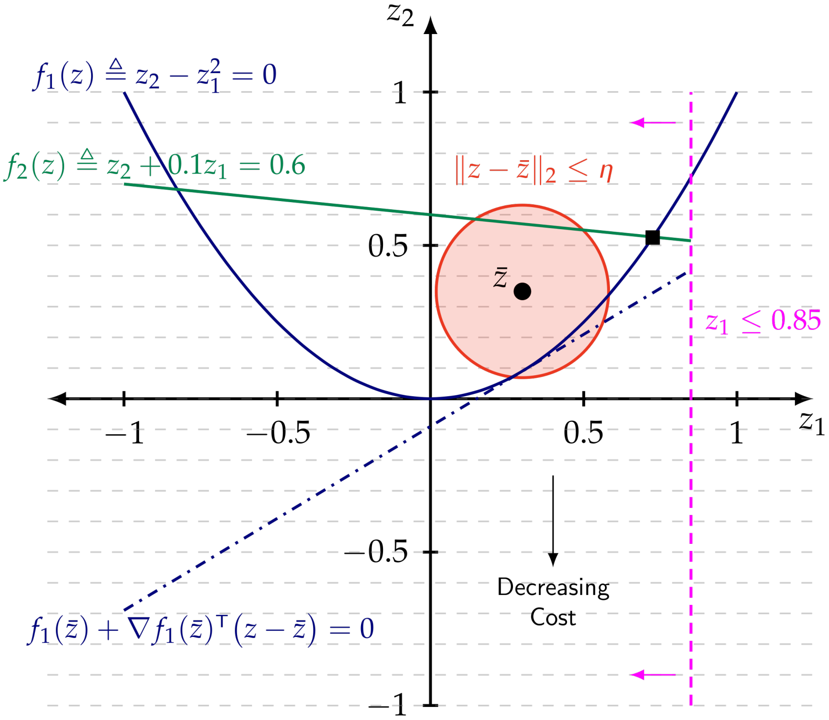
$$x_k \in \mathcal{X}_k, \qquad u_k \in \mathcal{U}_k, \qquad \text{State and control constraints} \tag{1c}$$

$$C_k x_k + D_k u_k + r'_k \le v_{s,k}, \qquad \text{Convexified/discretized path constraint} \tag{1d}$$

$$\|\delta x_k\|_q + \|\delta u_k\|_q \le \eta \qquad \text{Trust region} \tag{1e}$$

- x is a state vector, u a control vector
- $\mathcal{X}_k$ and $\mathcal{U}_k$ are convex sets
- A_k, B_k, C_k, D_k, E_k are matrices
- δ are the difference between the subproblem solution and the linearization point

Convex subproblem example: simple 2D cases



Remark: linearization can induce unbounded solution of the optimization problem. A *trust region* is inserted to keep the linearization "close to" the reference point.

Python implementation

Tools and Requirements

Starting point: CVXPY

CVXPy is an open source Python-embedded modeling language for convex optimization problems[a]. Main ingredients:

- Based on **Disciplined Convex Programming (every constraints must be convex)**
- Based on state of the art numerical algorithms (*e.g.*, NumPy) and convex solvers (*e.g.*, ECOS)

[a] `https://www.cvxpy.org`

First contribution: from CVXPY to SCVXPy

- Automatic linearization methods based on SymPy to generate symbolic gradients
- Handmade outer loop

Dubin's car example

Optimal problem defined by

- 2D Dubin's car dynamics

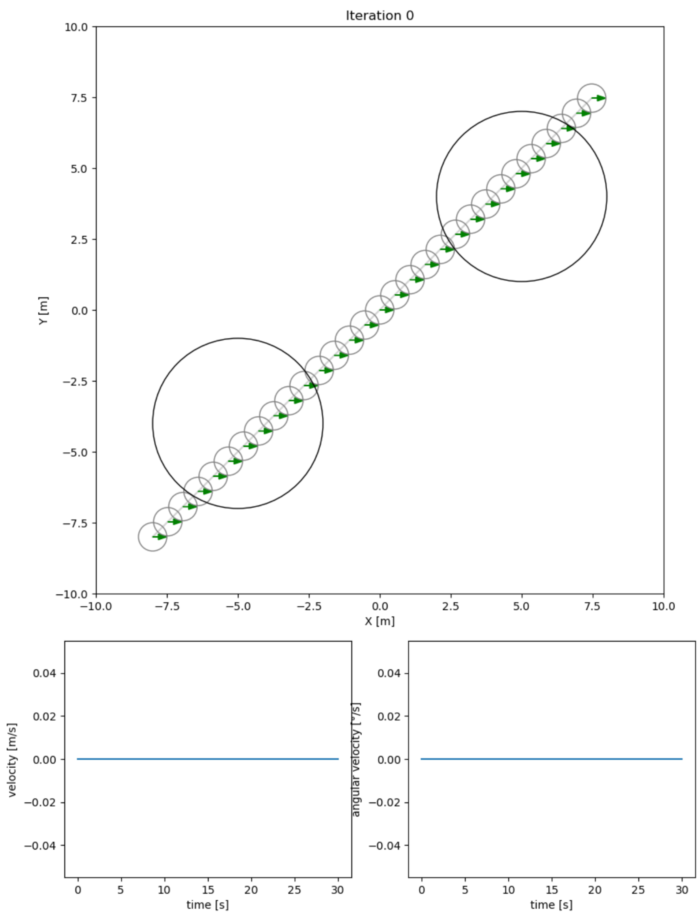
$$\dot{x} = u_1 \cos\theta, \quad \dot{y} = u_1 \sin\theta, \quad \dot{\theta} = u_2$$

- Mission: reach a target in 30 seconds and avoid obstacles

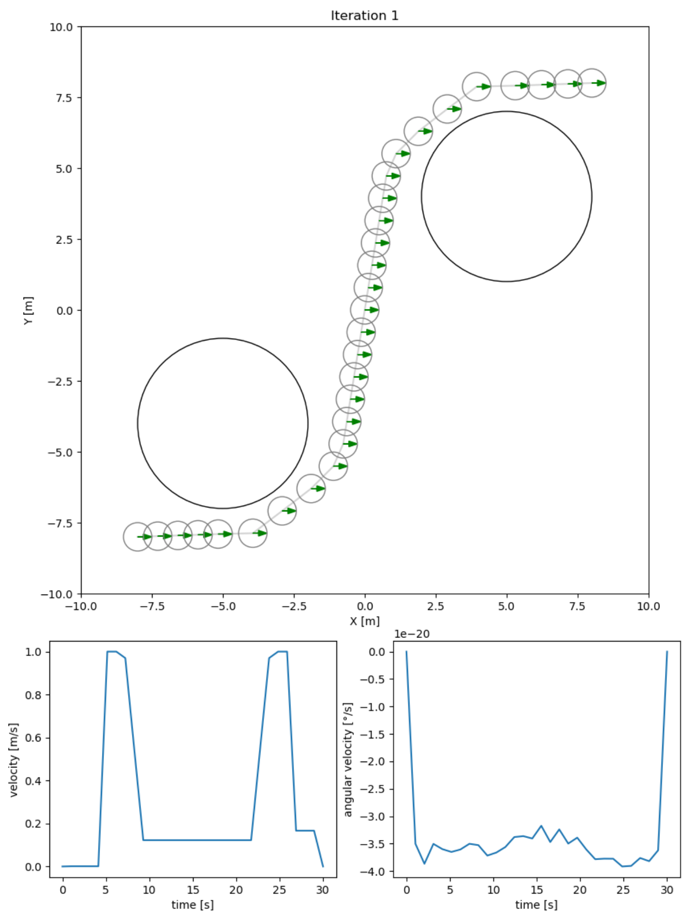
SCVX parameters

- 30 discretization points
- Initial trust region radius sets to 5
- 10000 intermediate points for linearization

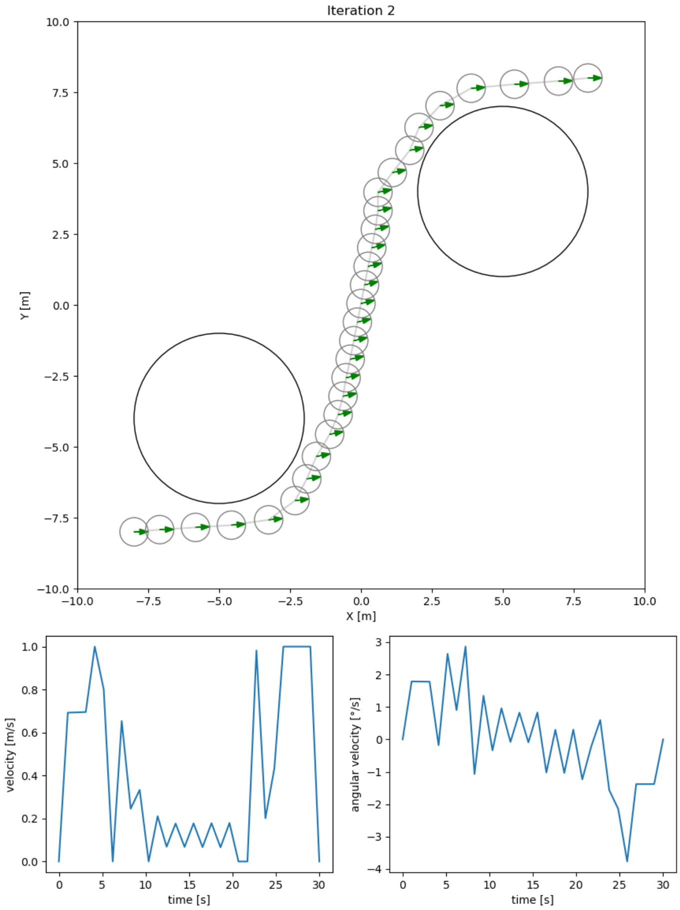
Solution on example (Python implem.) – iteration 0



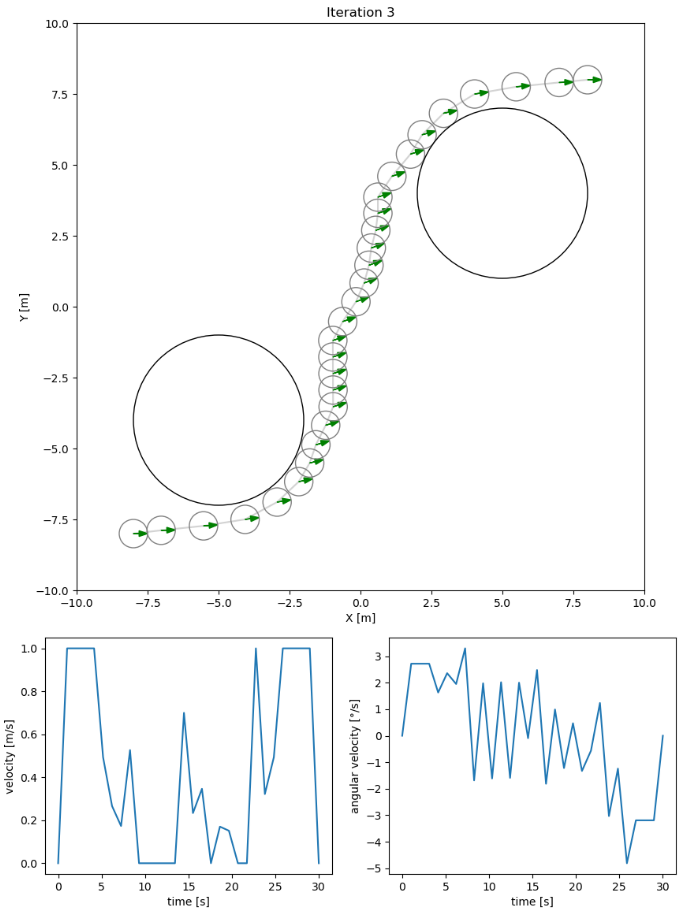
Solution on example (Python implem.) – iteration 1



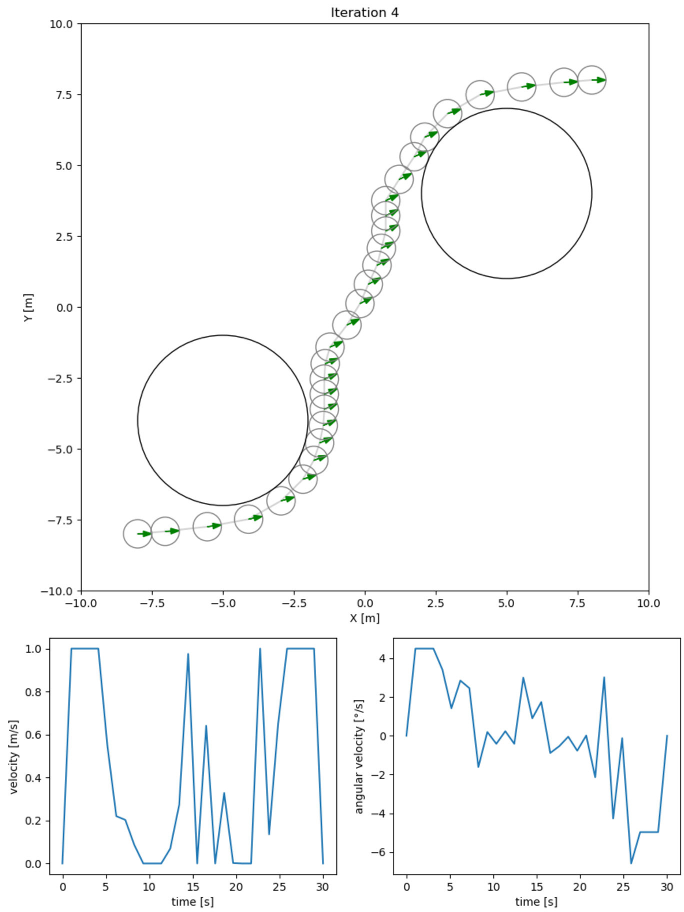
Solution on example (Python implem.) – iteration 2



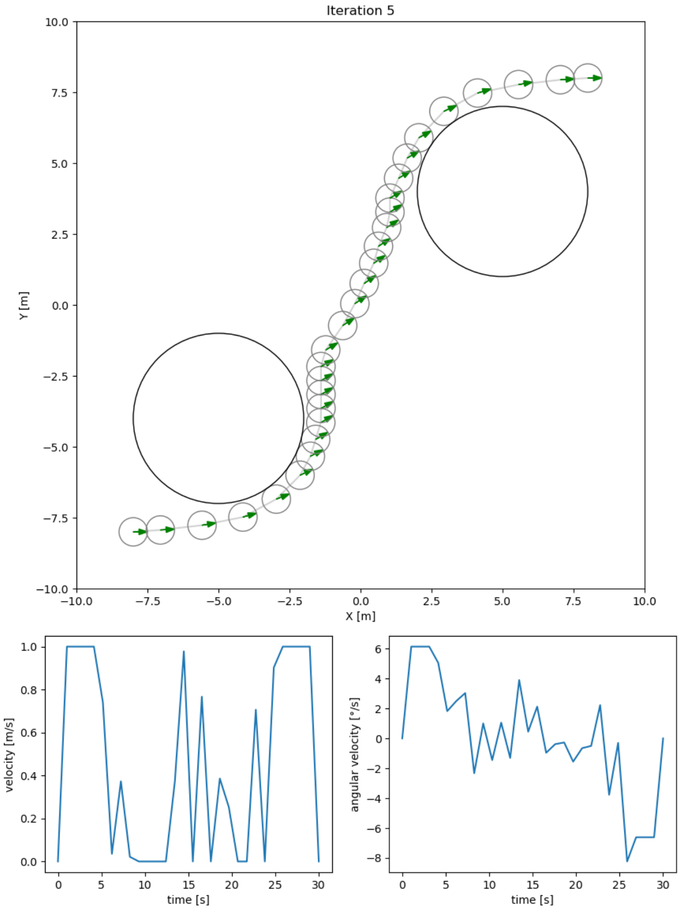
Solution on example (Python implem.) – iteration 3



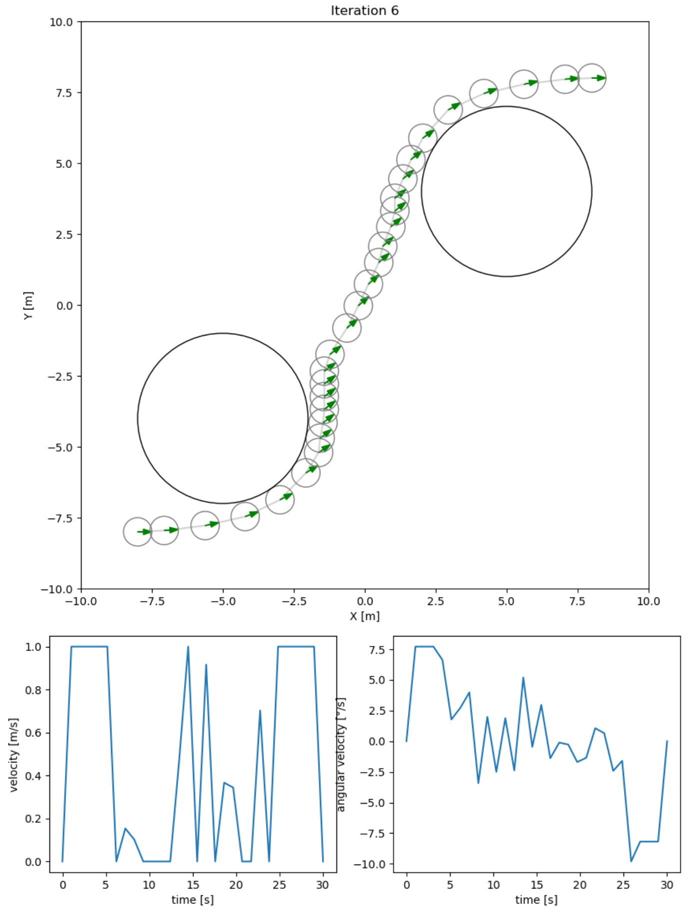
Solution on example (Python implem.) – iteration 4



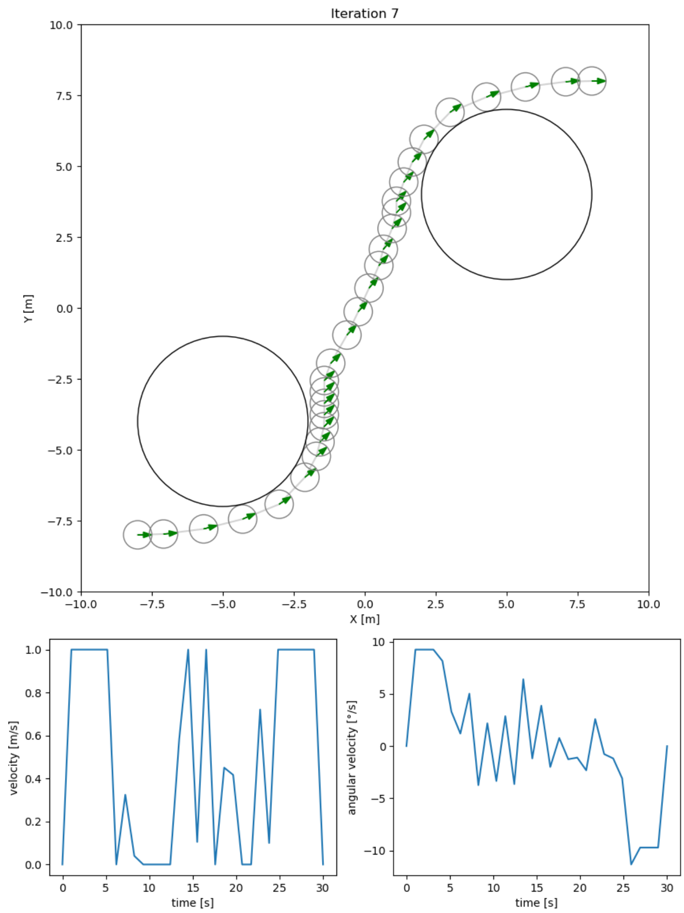
Solution on example (Python implem.) – iteration 5



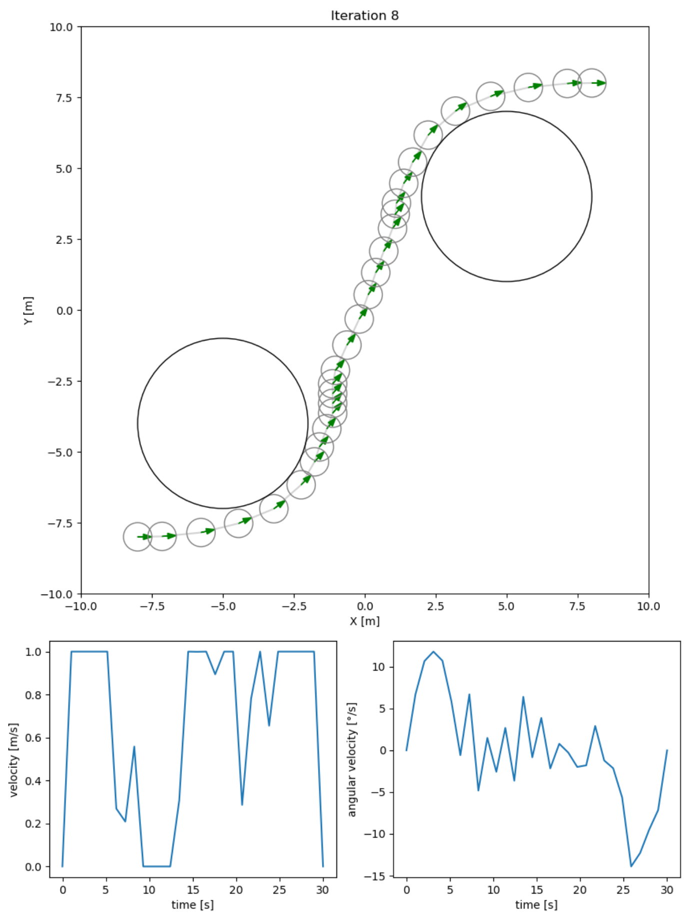
Solution on example (Python implem.) – iteration 6



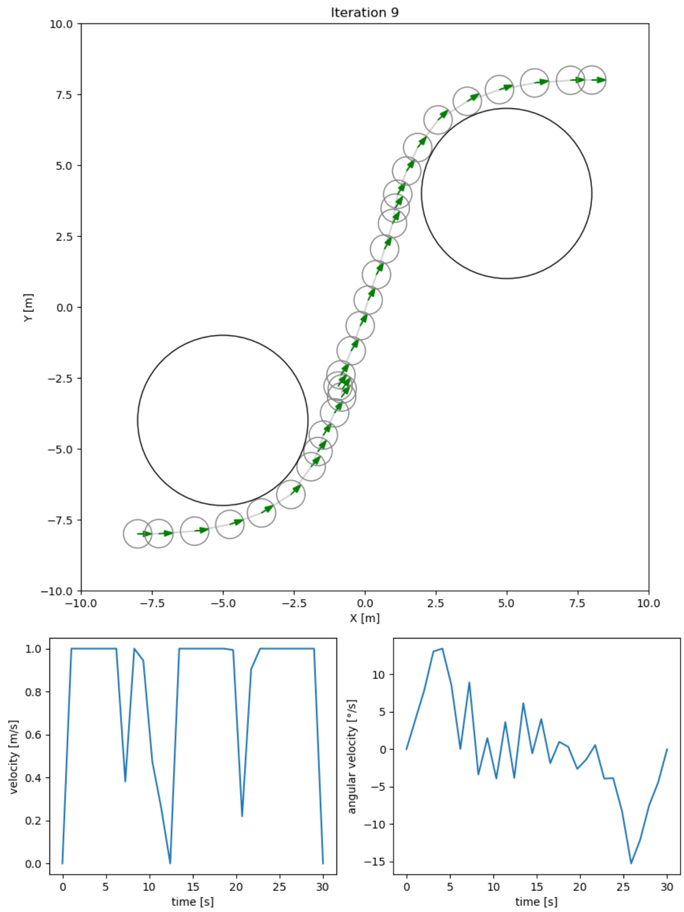
Solution on example (Python implem.) – iteration 7



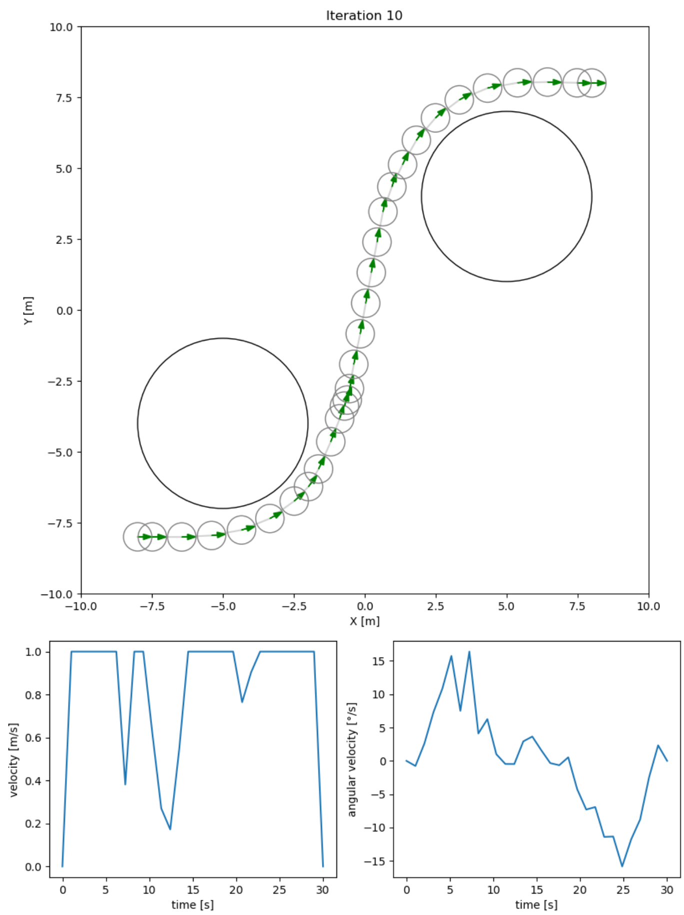
Solution on example (Python implem.) – iteration 8



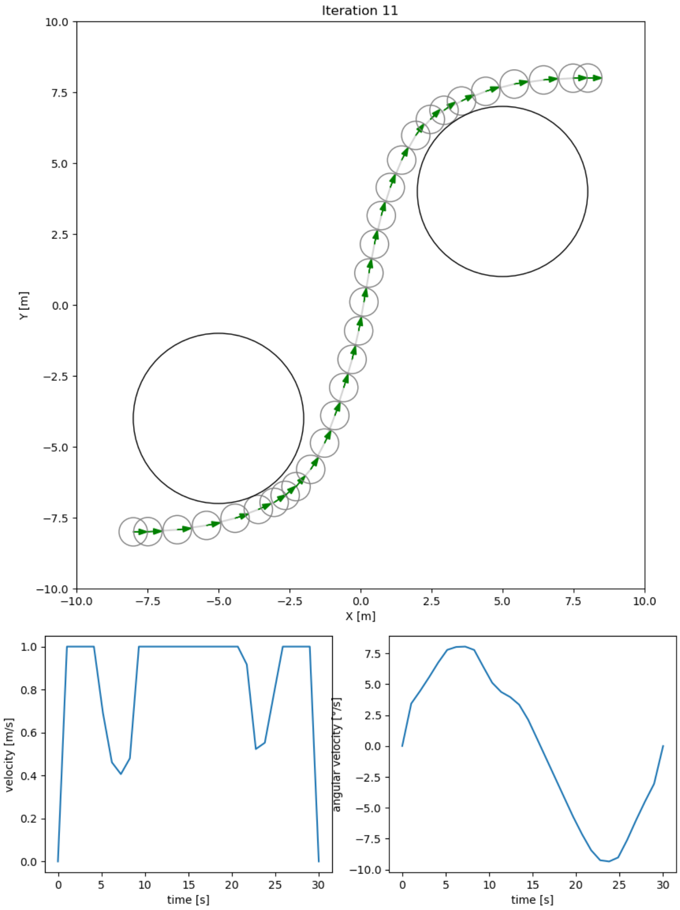
Solution on example (Python implem.) – iteration 9



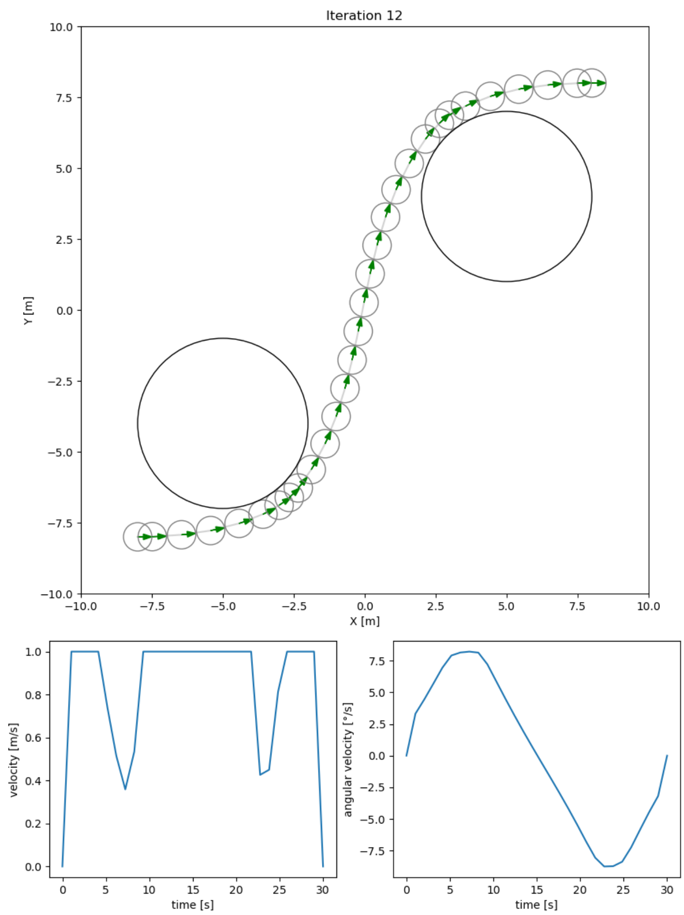
Solution on example (Python implem.) – iteration 10



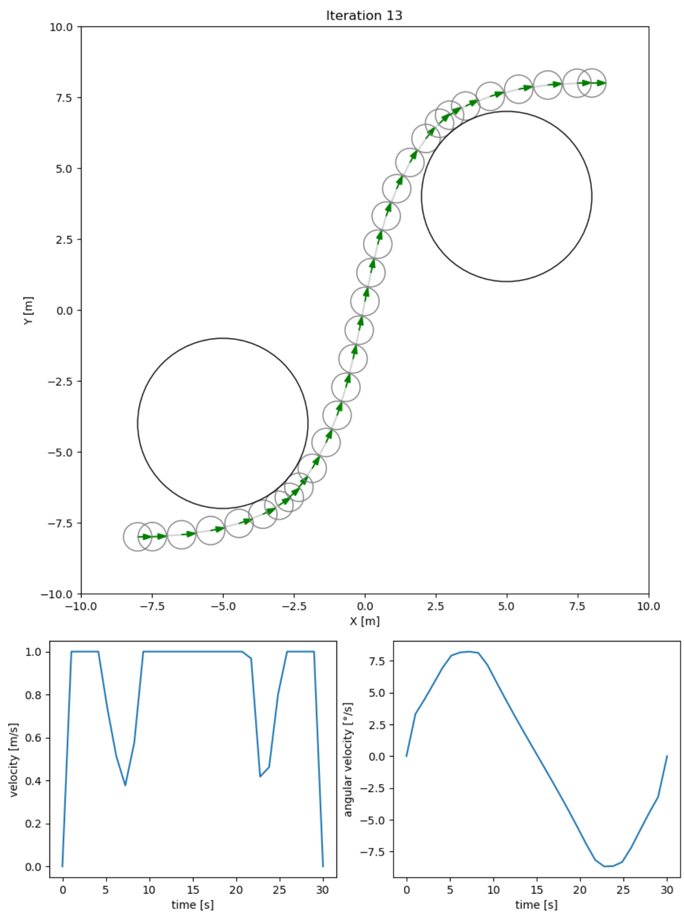
Solution on example (Python implem.) – iteration 11



Solution on example (Python implem.) – iteration 12



Solution on example (Python implem.) – iteration 13



Python code performance analysis

Command being timed: "python SCvx.py"

- User time (seconds): 3.08
- System time (seconds): 0.03
- Percent of CPU this job got: 100
- Elapsed (wall clock) time (h:mm:ss or m:ss): 0:03.12
- Maximum resident set size (kbytes): 161068

C code generation

Tools and Requirements

Starting point: CVXPYGen

CVXPy is an open source Python-embedded modeling language for convex optimization problems[a]. Main ingredients:

- Based on **Disciplined Convex Programming** and **Disciplined Parameterized Programming**
- Based on state of the art C code generator for convex problems (*e.g.*, CVXPYgen)

[a] `https://www.cvxpy.org`

Second contribution: from CVXPY to SCVXPyGen

- Automatic linearization methods based on SymPy to generate symbolic gradients and C code
- Handmade outer loop in C

Autocoding in C

Disciplined Parameterized Programming

DPP ensures that each constraints is composed of at most 1 parameter.
Michael's constraint that creates a repulsive vector:

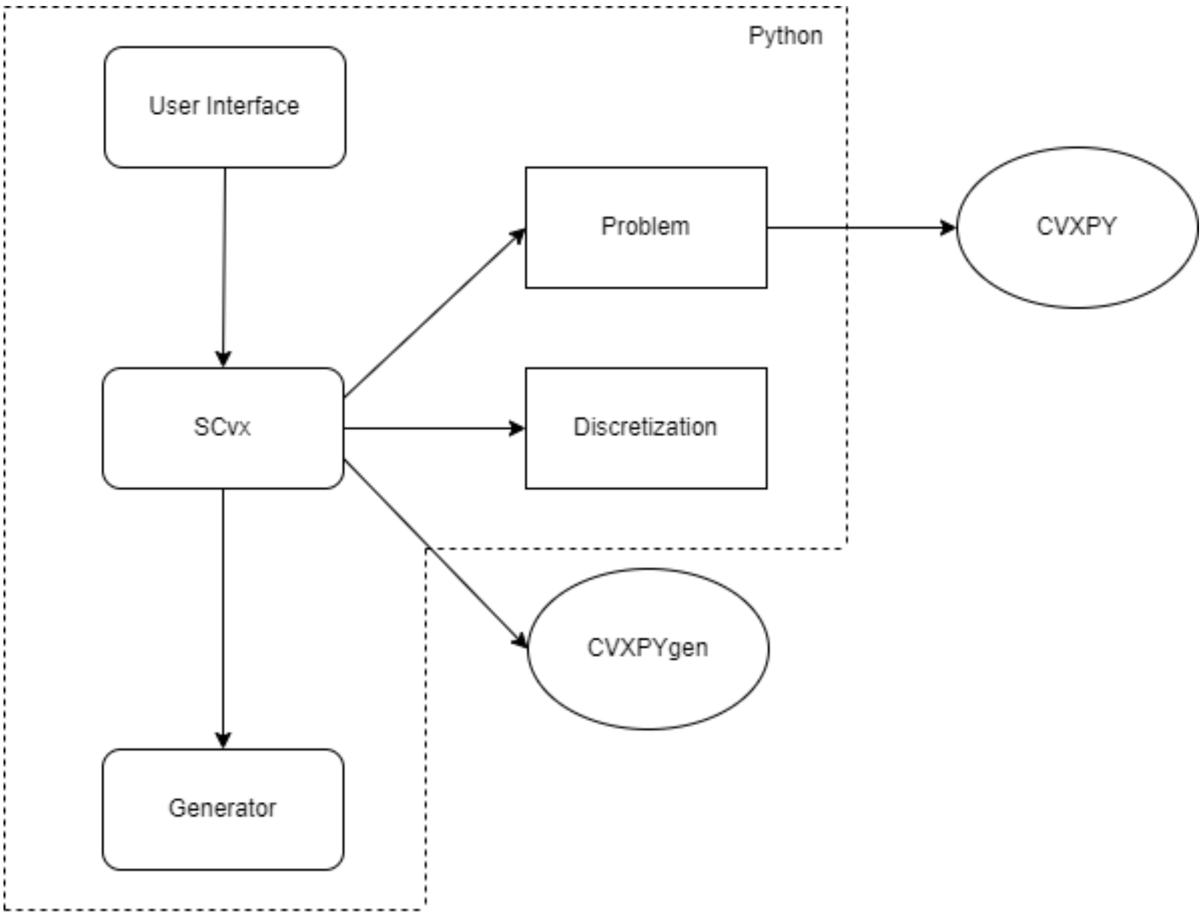
$$r - \frac{\left(x_k^{(i-1)} - p\right)\left(x_k^{(i)} - p\right)}{||x_k^{(i-1)} - p||_2 + 10^{-6}} < 0$$

becomes,

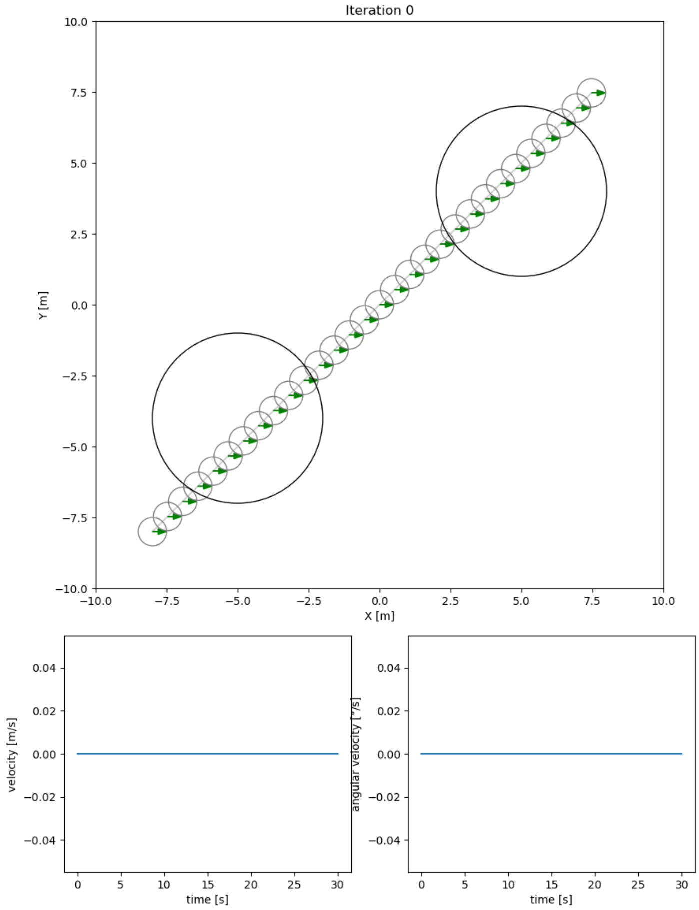
$$r - \left(\text{ObstacleParam}_k^{(i)}\left(x_k^{(i)} - p\right)\right) < 0$$

Autocoding in C

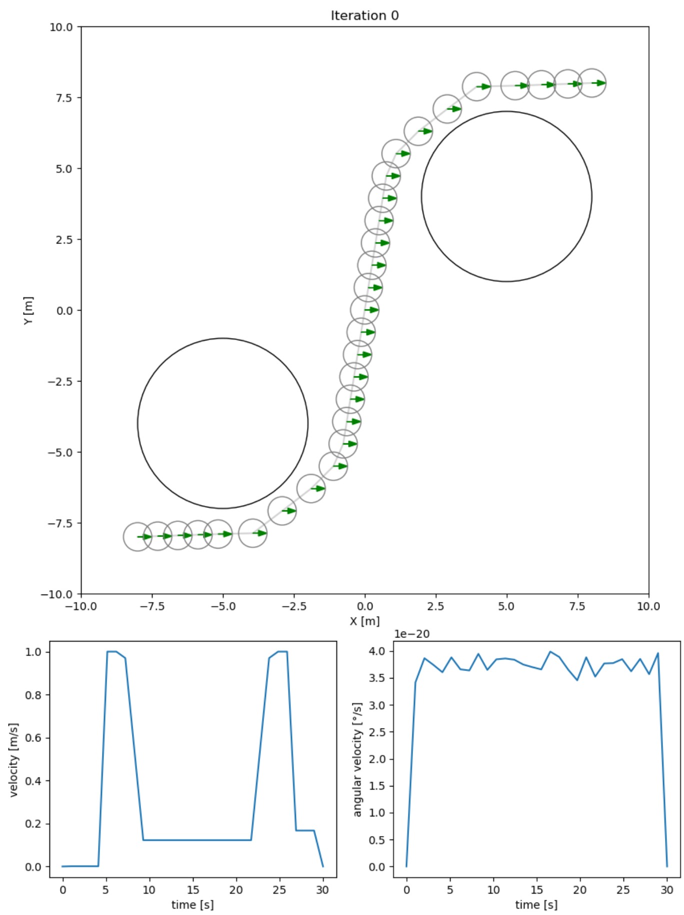
Architecture



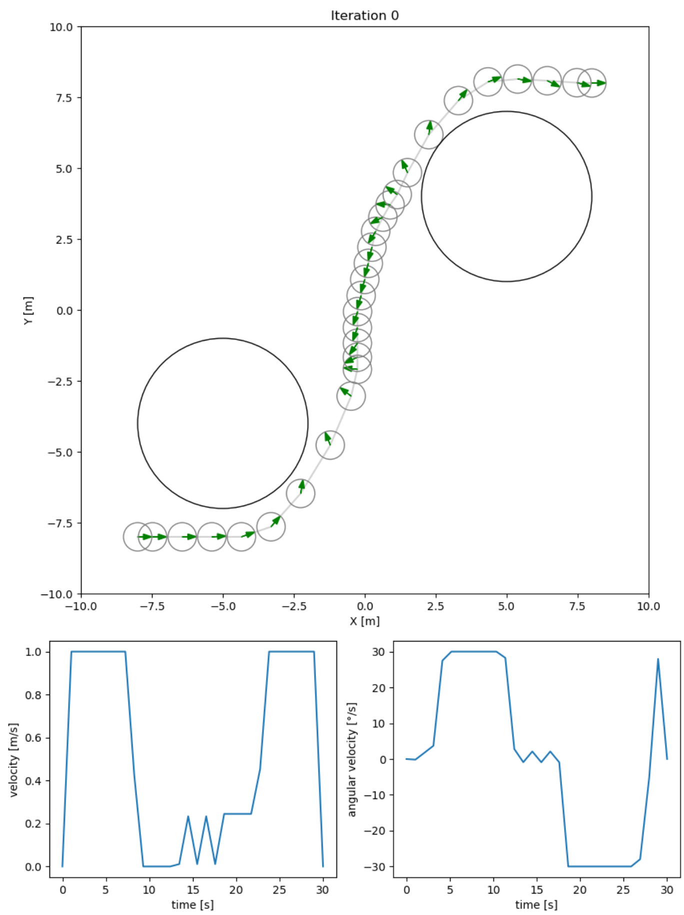
Solution on example (C implem.) – iteration 0



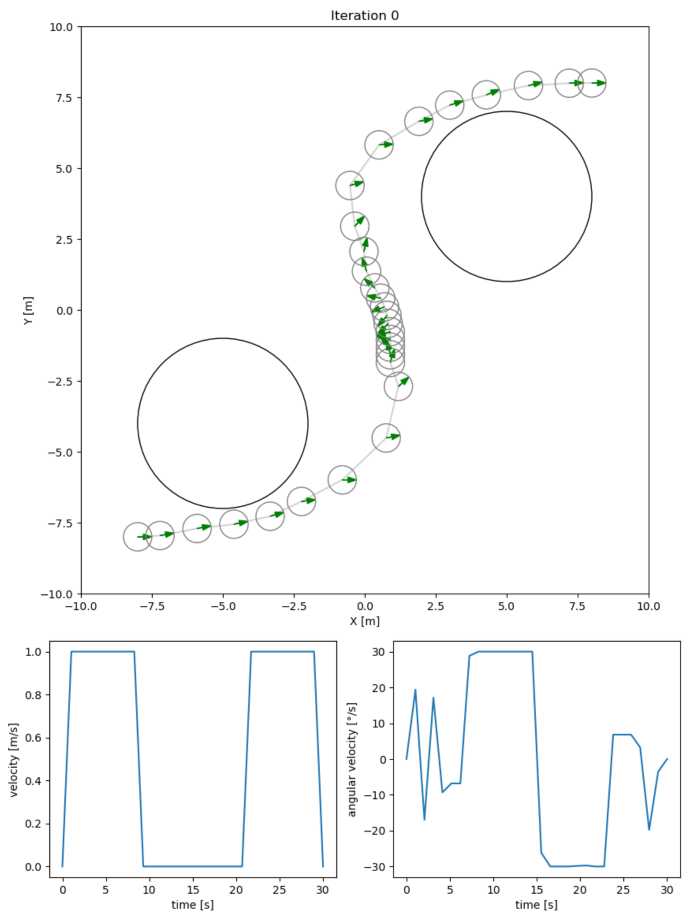
Solution on example (C implem.) – iteration 1



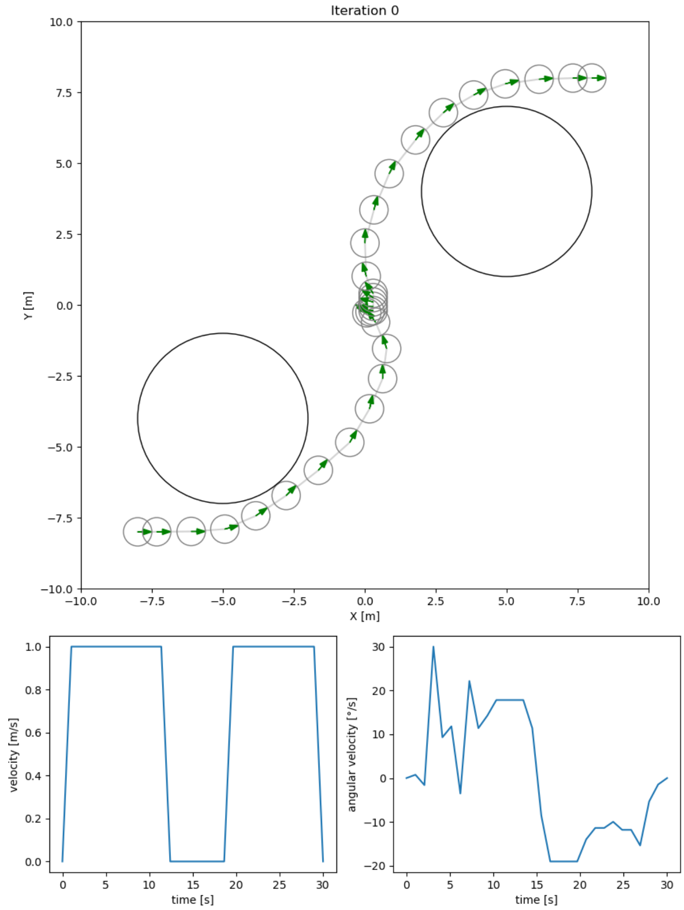
Solution on example (C implem.) – iteration 2



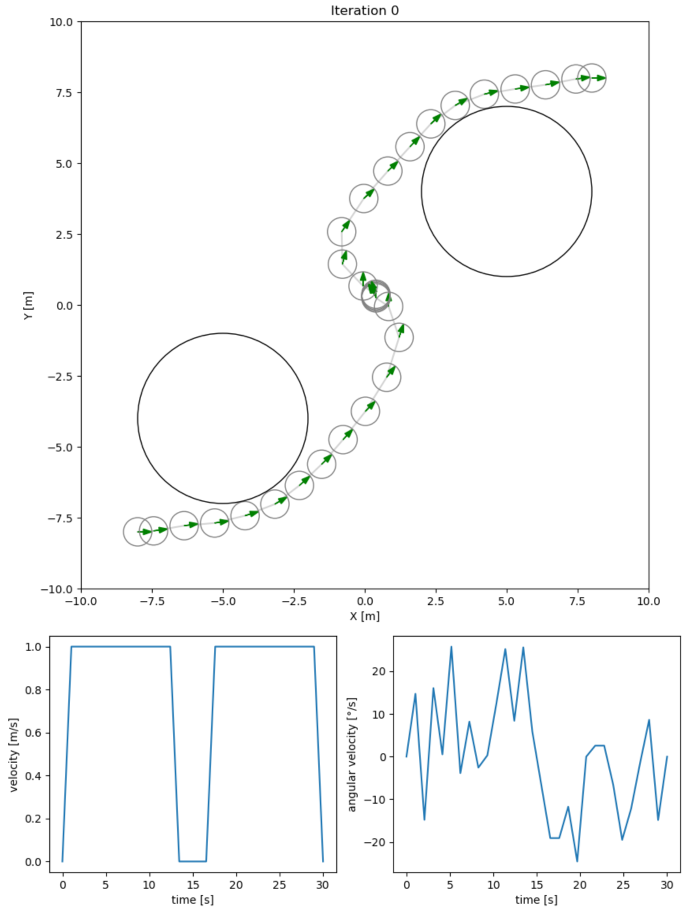
Solution on example (C implem.) – iteration 3



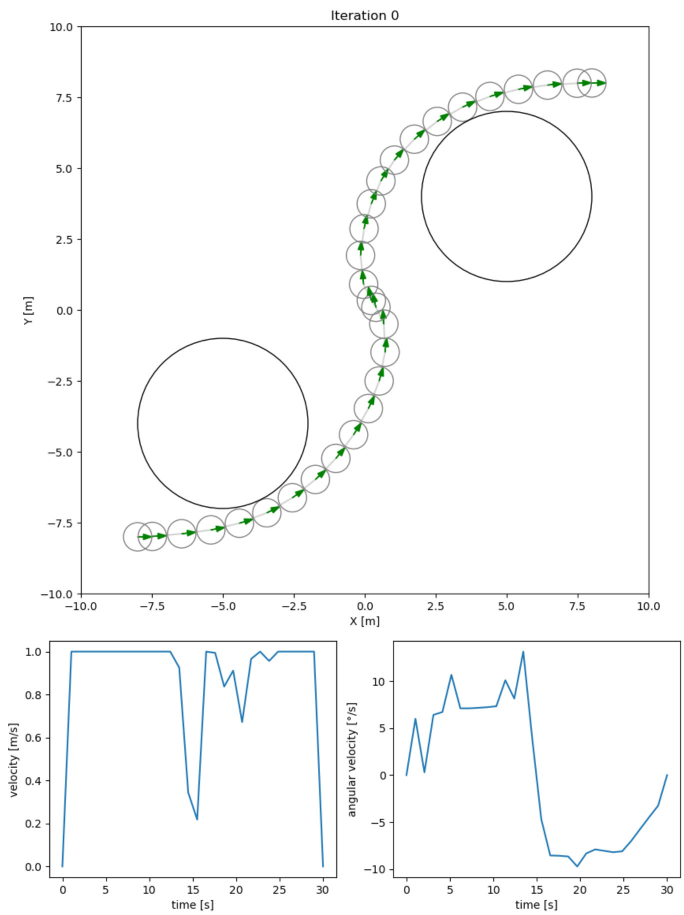
Solution on example (C implem.) – iteration 4



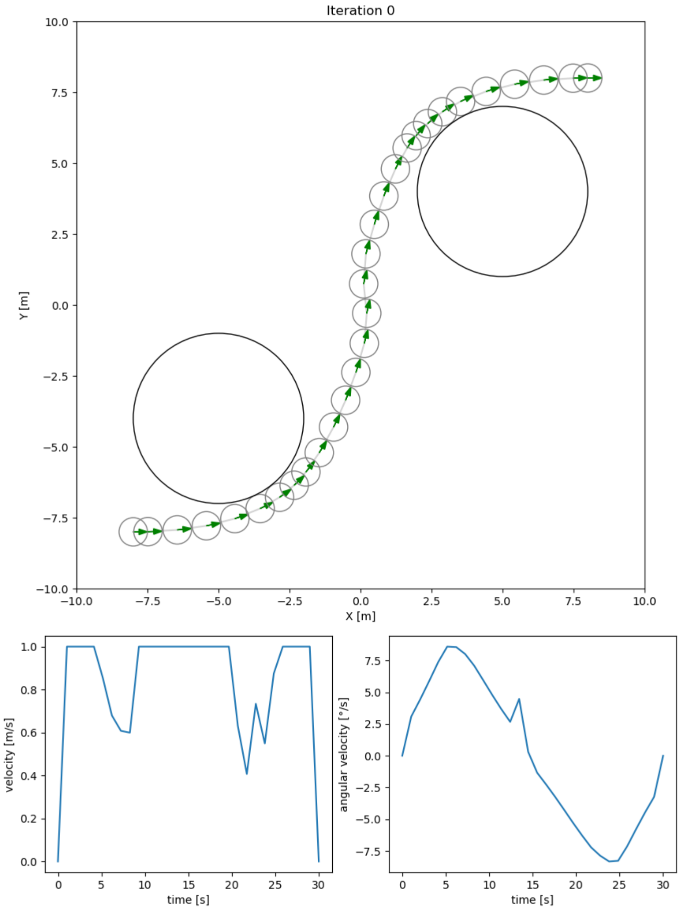
Solution on example (C implem.) – iteration 5



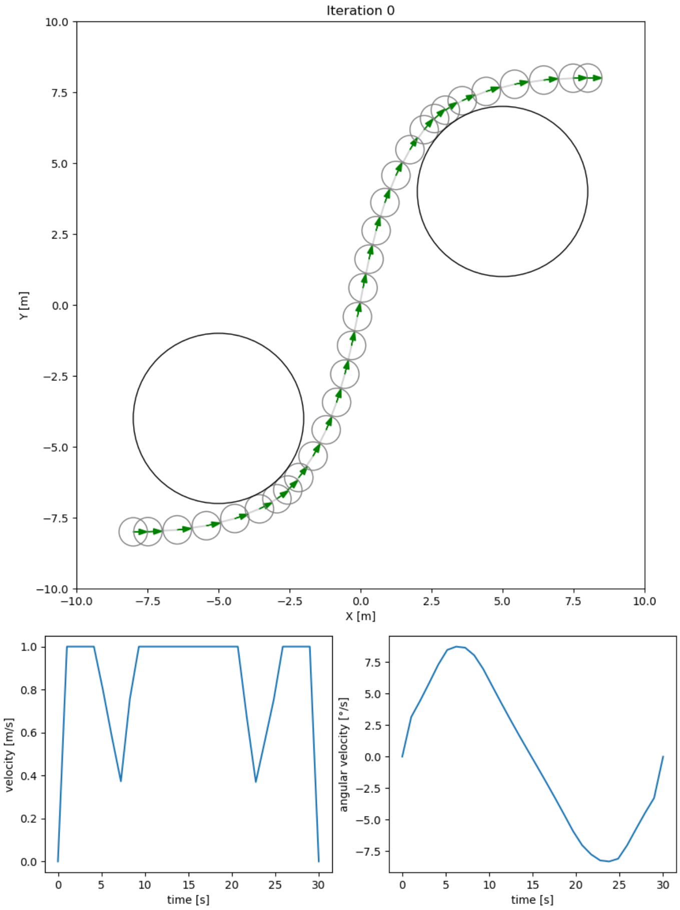
Solution on example (C implem.) – iteration 6



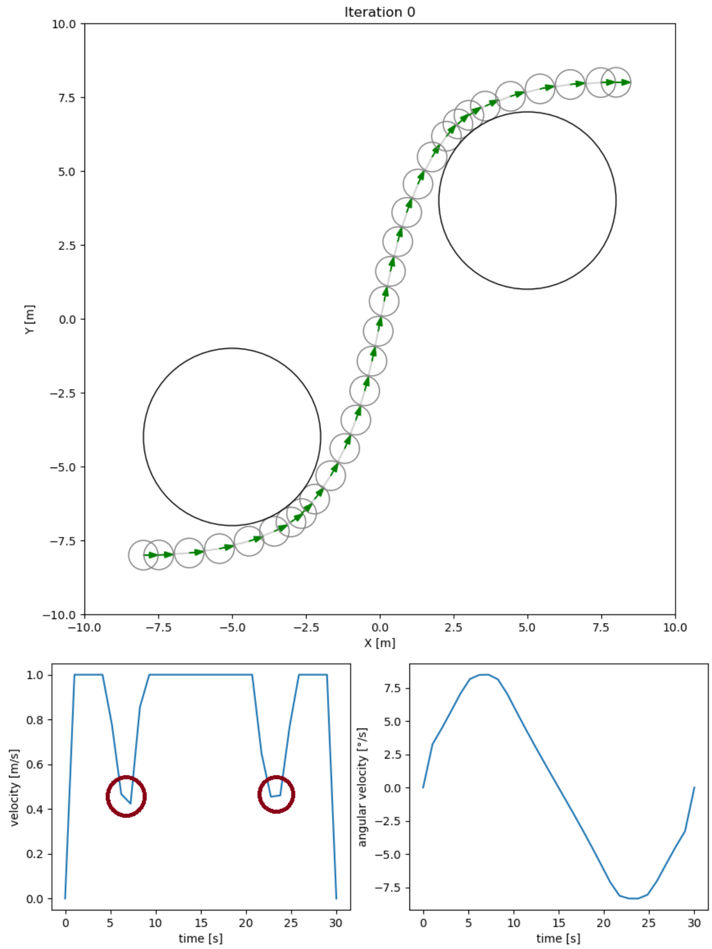
Solution on example (C implem.) – iteration 7



Solution on example (C implem.) – iteration 8



Solution on example (C implem.) – iteration 9



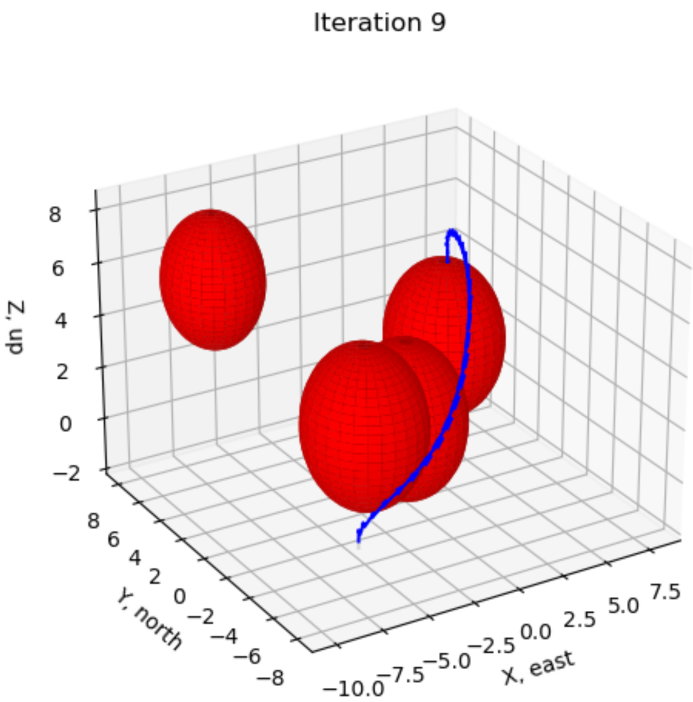
C code performance analysis

Command being timed: "./SCvx"

- User time (seconds): 0.88
- System time (seconds): 0.00
- Percent of CPU this job got: 100
- Elapsed (wall clock) time (h:mm:ss or m:ss): 0:00.88
- Maximum resident set size (kbytes): 5248

Remarks: embedded constraints may be met by this C code generator. A lot of improvements can be made to reduce memory consumption.

Ongoing example



- 30 discretization points
- 8 secondes trajectory
- Avoids obstacles
- Ongoing work to automatically generate C code for 3D dynamics

1st Year summary

What have been done on technical side?

- Implementation of SCvxPy
- Implementation of SCvxPyGen

Next steps

- Continue testing on more complex examples
- Reduce memory consumption
- Implement this algorithm on real robots
- Plan missions using Temporal Logic

Promoting results: paper submission at ECC'24 (deadline October 25).

Main uses of STL

Temporal logics specify patterns that timed behaviors of systems may or may not satisfy. Many flavors but we consider STL.

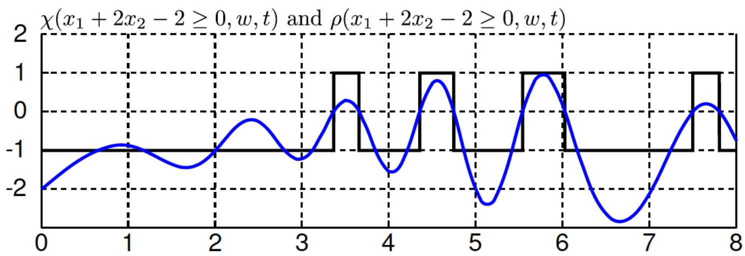
STL formula example

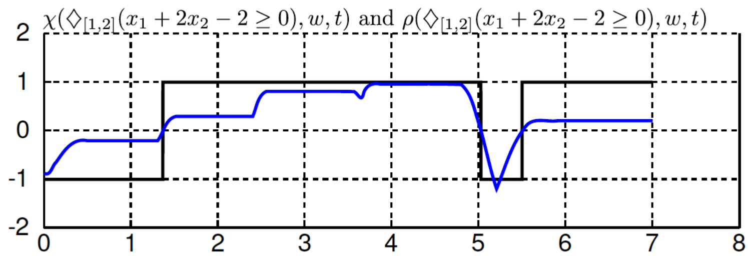
Between 2s and 6s the signal is between -2 and 2

$\phi := \text{Globaly}_{[2,6]}(|\ x(t)\ | < 2)$ Note: temporal modalities are bounded

STL is mainly used:

- to perform (offline/online) **monitoring**
- to perform **controller synthesis**





Long term goal: integrate STL constraints into SCvx

Non-convex Problem with STL constraints:

$$\underset{x_i, u_i}{\text{minimize}} \quad \sum_{i=1}^{t_f} \phi(x_i, u_i),$$

subject to

$$\begin{aligned}
&x_{i+1} = f(x_i, u_i) && i = 1, 2, \ldots, t_f - 1, \\
&s(x_i) \leq 0 && i = 1, 2, \ldots, t_f, \\
&u_i \in U_i, x_i \in X_i && i = 1, 2, \ldots, t_f - 1, \\
&\boxed{(x, u) \models \varphi} &&
\end{aligned}$$

What should be done

- A PoC has been done[2] **but need to automatize the linearization**
- **Manage nested temporal logic operator of STL**

[2] SCVx for Optimal Control with STL Specifications, Mao *et al.*, HSCC'22

References

- Y. Mao, M. Szmuk, and B. Acikmese, “Successive convexification of non-convex optimal control problems and its convergence properties,” in 2016 IEEE 55th Conference on Decision and Control (CDC). IEEE, dec 2016.
- A. Domahidi, E. Chu, and S. Boyd, “ECOS: An SOCP solver for embedded systems,” in 2013 European Control Conference (ECC). IEEE, Jul. 2013.
- A. Agrawal, R. Verschueren, S. Diamond, and S. Boyd, “A rewriting system for convex optimization problems,” Journal of Control and Decision, vol. 5, no. 1, pp. 42–60, 2018.
- M. Schaller, G. Banjac, S. Diamond, A. Agrawal, B. Stellato, and S. P. Boyd, “Embedded code generation with CVXPY,” IEEE Control. Syst. Lett., vol. 6, pp. 2653–2658, 2022.
- Yuanqi Mao, Behcet Acikmese, Pierre-Loïc Garoche, Alexandre Chapoutot. Successive Convexification for Optimal Control with Signal Temporal Logic Specifications. 25th ACM International Conference on Hybrid Systems: Computation and Control (HSCC ’22), May 2022, Milan, Italy. 10.1145/3501710.3519518. hal-03663984f