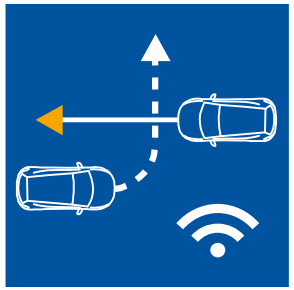


Lecture

Control and Perception in Networked and Autonomous Vehicles

Dr. Bassam Alrifaee | Patrick Scheffe, M. Sc. | Simon Schäfer, M. Sc.

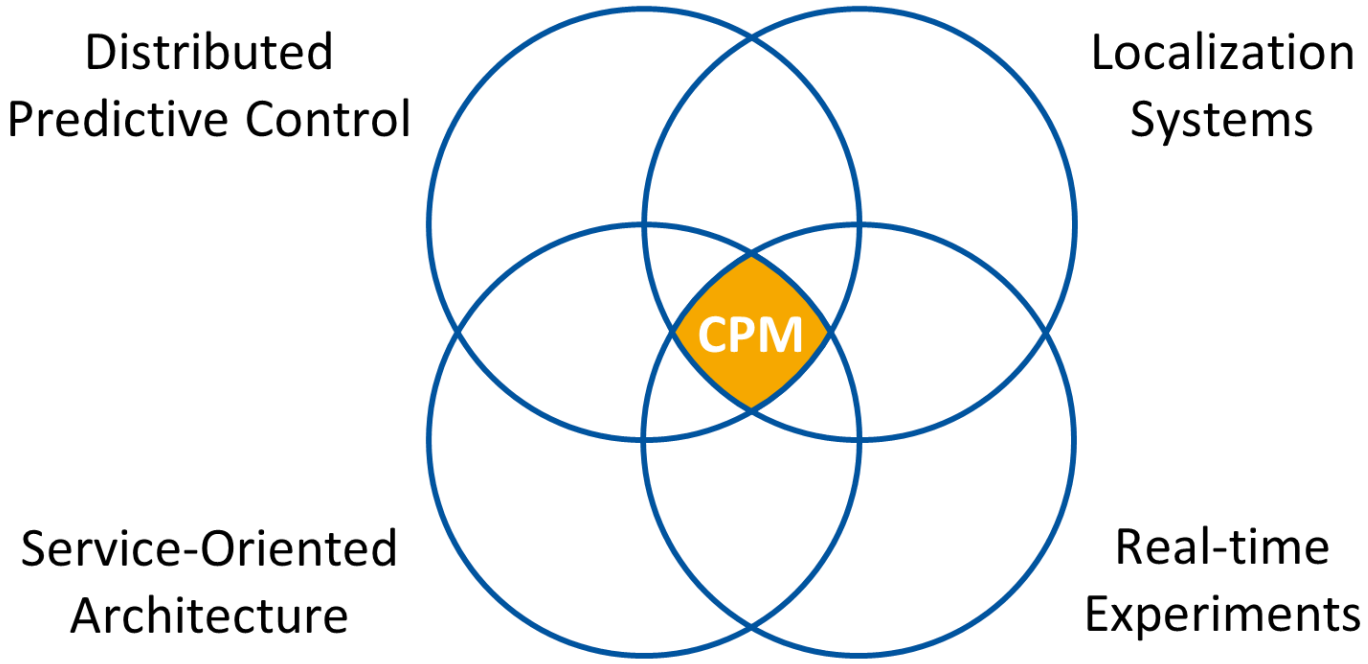
Winter Semester 2023/2024

Part 3

Control Engineering and Optimization

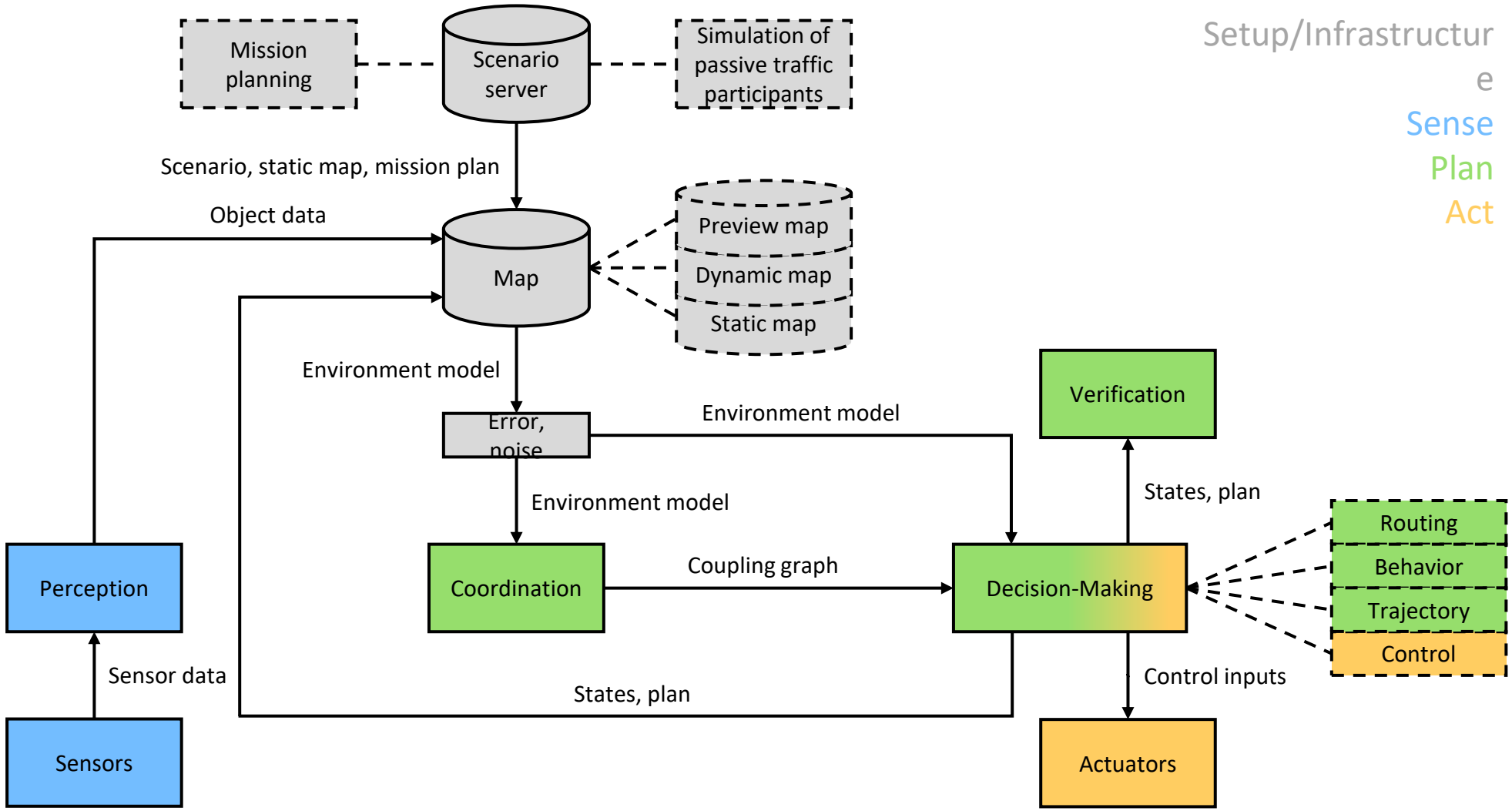
Course contents (CPM group course)

- Vehicle models
- Control and optimization
- Network and distribution
- Machine perception
- Software architectures and testing concepts
- Case study

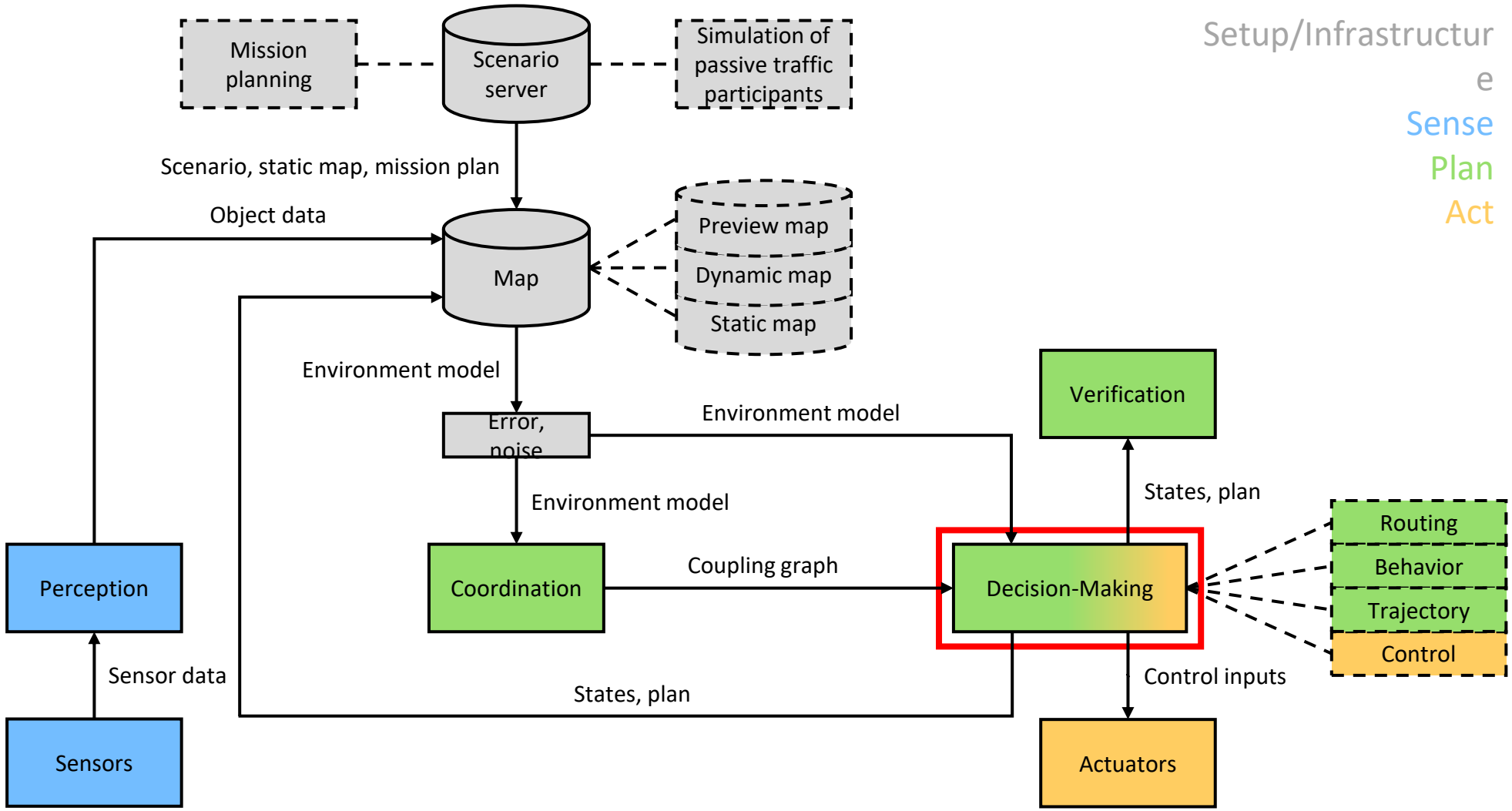


*CPM: Cyber-Physical Mobility

CPM Lab architecture



CPM Lab architecture



Literature

- J. Maciejowski. Predictive Control with Constraints. Prentice Hall, 2002.
- S. Boyd and L. Vandenberghe. Convex Optimization. Cambridge University Press, 2004.
- R.C. Dorf and R.H. Bishop. Modern Control Systems. Prentice-Hall, 2008.

Further literature (1)

- F. Borrelli, A. Bemporad, and M. Morari. Predictive Control for Linear and Hybrid Systems, Cambridge University Press, 2017.
- And many more…

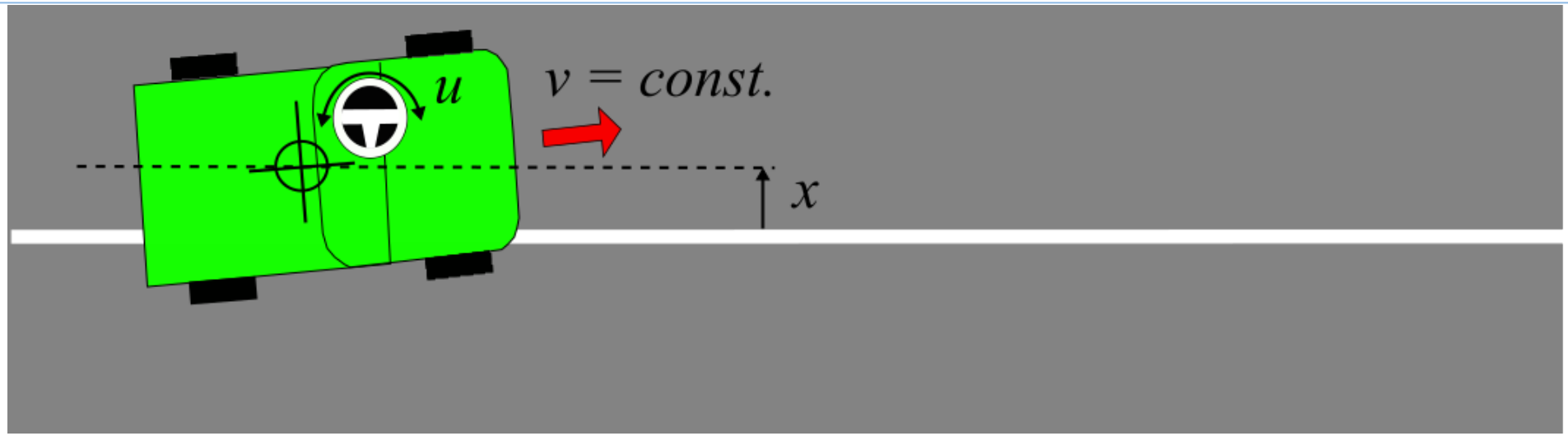
Further literature (2)

- B. Alrifaee. Networked Model Predictive Control for Vehicle Collision Avoidance. PhD thesis, RWTH Aachen University, 2017.
- B. Alrifaee. MATLAB Simulation of Networked Model Predictive Control for Vehicle Collision Avoidance, 2017. Available: https://doi.org/10.5281/zenodo.1252992
- Check out our website

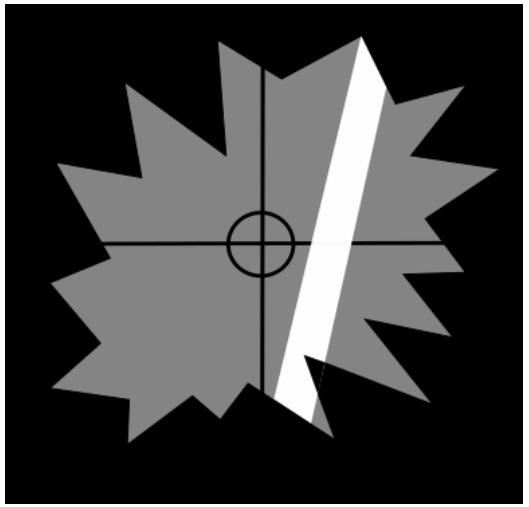
Model predictive control

- Flipped classroom
  - Group B should prepare a summary, ca. 15 minutes
- Watching
  - https://youtu.be/1A734g96Npk
- Reading
  - B. Alrifaee. Networked Model Predictive Control for Vehicle Collision Avoidance. PhD thesis, RWTH Aachen University, 2017.
    - Section 2.3, pages 8-14
    - Section 4.5.1.1, pages 100-101
  - J. Maciejowski. Predictive Control with Constraints. Prentice Hall, 2002. **[optional]**
    - Section 2.6.1, pages 53-56
    - Sections 2.6.2 and 2.6.3, pages 56-61 **[optional]**
    - Section 3.1.1, pages 74-77
    - Section 3.2.1, pages 81-84
    - Section 3.4, pages 97-99

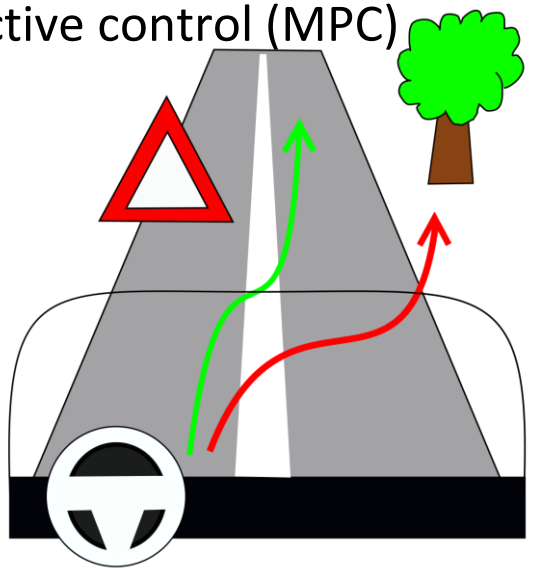
Model predictive control



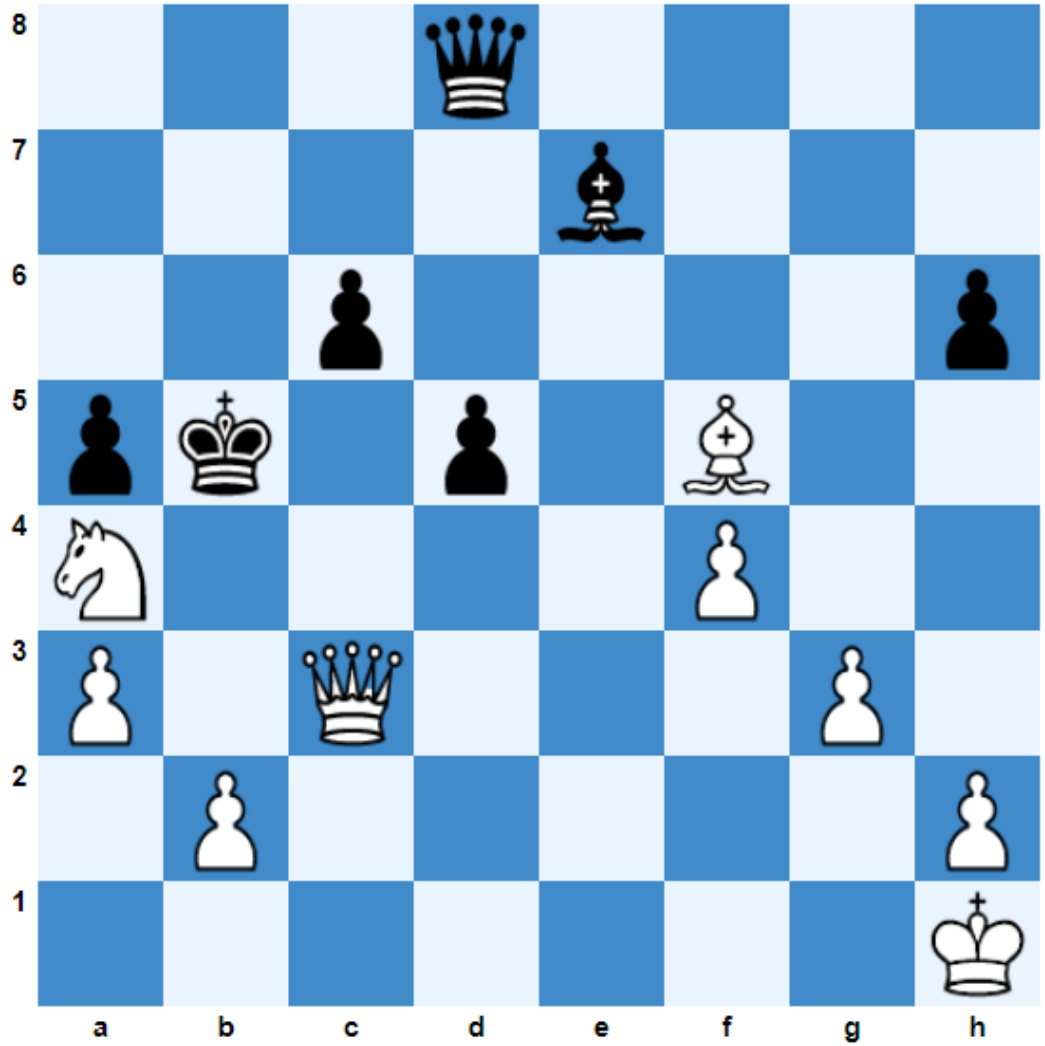
classical control (PID)



predictive control (MPC)



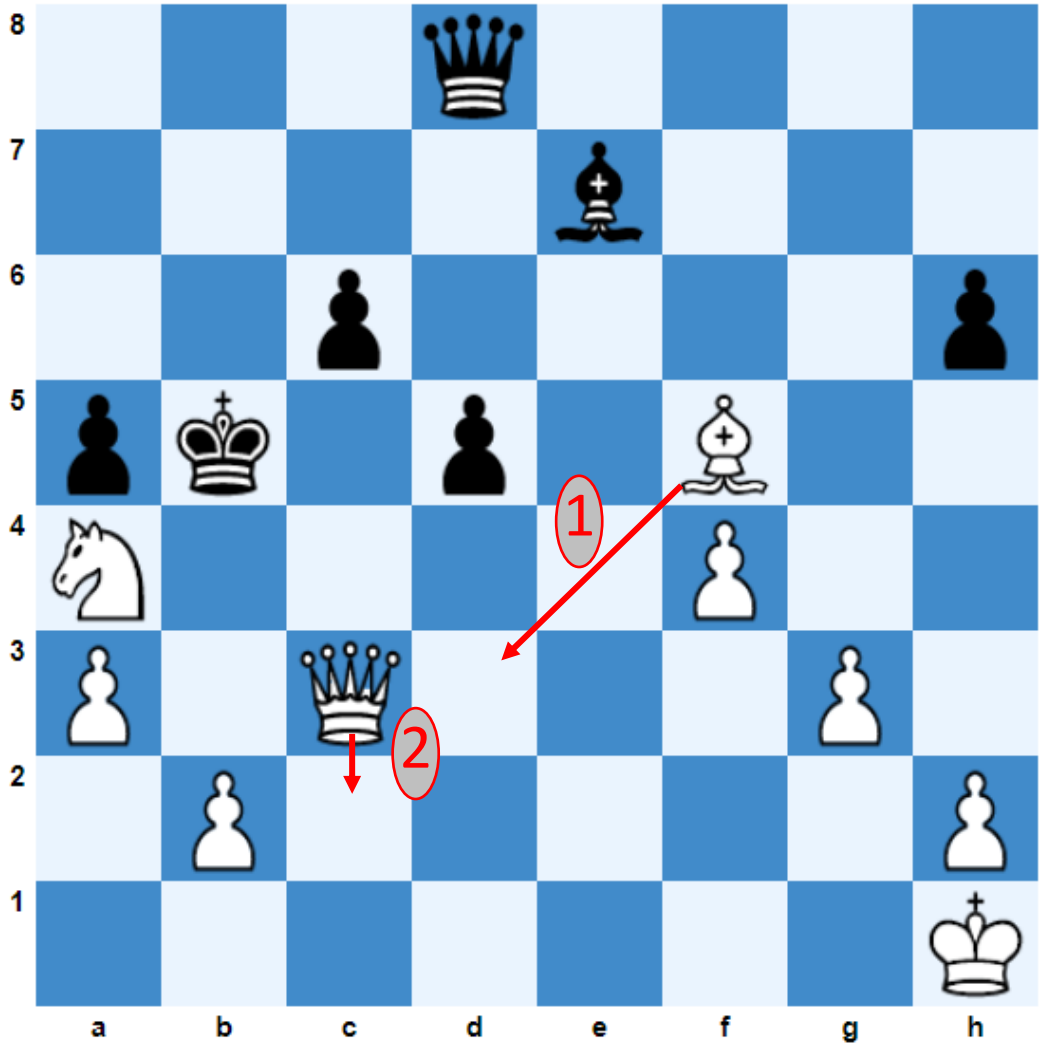
Model predictive control



schach-tipps.de, euroschach.de

Model predictive control

1. Bd3
2. Qc2

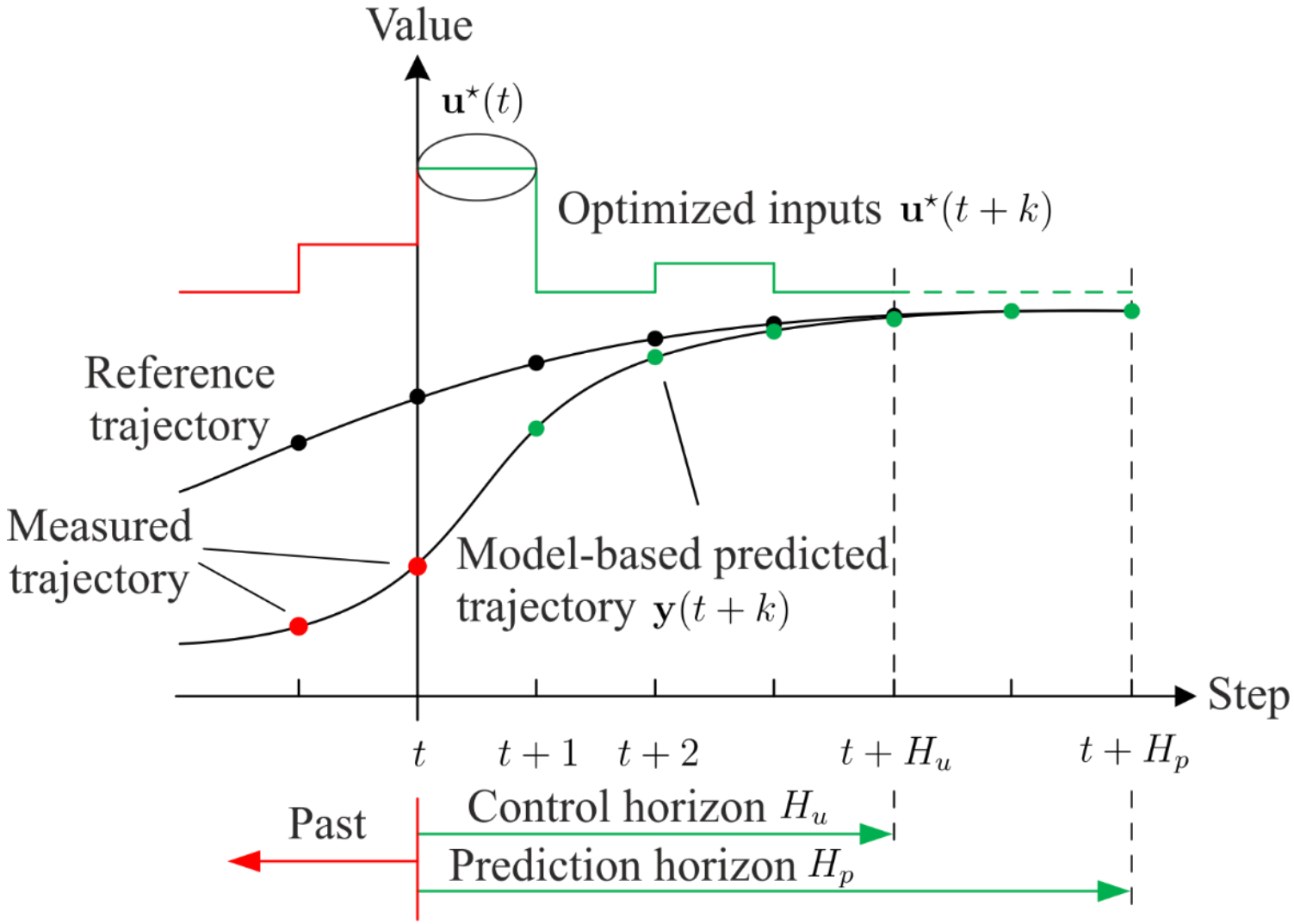


schach-tipps.de, euroschach.de

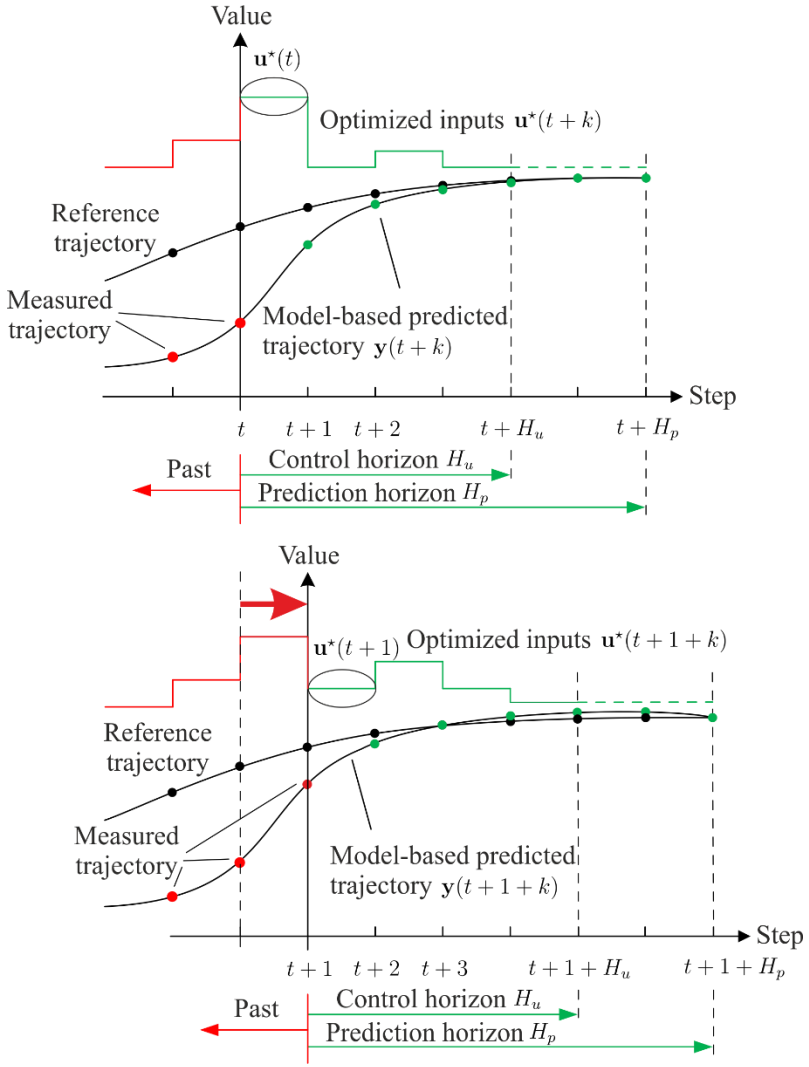
Model predictive control

- Further examples
  - Automotive systems
  - Aeronautic industry
    - See SpaceX landing, ClearSpace
  - Smart electricity grids
  - Drinking water networks
  - Financial engineering

Model predictive control



Model predictive control



Model predictive control

$$J^\star = \min_{\Delta \mathbf{u}(\cdot)} \sum_{k=1}^{H_p-1} l_x(\mathbf{x}(t+k), \mathbf{r}(t+k)) + l_{x_{H_p}}(\mathbf{x}(t+H_p), \mathbf{r}(t+H_p)) + \sum_{k=0}^{H_u-1} l_u(\Delta \mathbf{u}(t+k))$$

subject to:

$$\mathbf{x}(t+1+k) = f(\mathbf{x}(t+k), \mathbf{u}(t+k)),\ k = 0, \ldots, H_p - 1$$

$$\mathbf{x}(t+k) \in \mathcal{X},\ k = 1, \ldots, H_p - 1$$

$$\mathbf{x}(t+H_p) \in \mathcal{X}_{H_p}$$

$$\mathbf{u}(t+k) \in \mathcal{U},\ k = 0, \ldots, H_u - 1$$

$$\Delta \mathbf{u}(t+k) \in \Delta \mathcal{U},\ k = 0, \ldots, H_u - 1$$

where

$$\Delta \mathbf{u}(t+k) = \mathbf{u}(t+k) - \mathbf{u}(t+k-1),\ k = 0, \ldots, H_u - 1,\ \mathbf{u}(-1) = 0$$

Model predictive control

- Result

$$\Delta \mathbf{u}^\star(\cdot) = \begin{pmatrix} \Delta \mathbf{u}^\star(t) & \dots & \Delta \mathbf{u}^\star(t + H_p - 1) \end{pmatrix}^T$$

- Apply

$$\mathbf{u}^\star(t) = \mathbf{u}(t-1) + \Delta \mathbf{u}^\star(t), \quad \text{during} \quad [t, t+1)$$

$$J^\star = \min_{\Delta \mathbf{u}(\cdot)} \sum_{k=1}^{H_p - 1} l_x(\mathbf{x}(t+k), \mathbf{r}(t+k)) + l_{x_{H_p}}(\mathbf{x}(t+H_p), \mathbf{r}(t+H_p)) + \sum_{k=0}^{H_u - 1} l_u(\Delta \mathbf{u}(t+k))$$

subject to:

$$\mathbf{x}(t+1+k) = f(\mathbf{x}(t+k), \mathbf{u}(t+k)), \ k = 0, \dots, H_p - 1$$
$$\mathbf{x}(t+k) \in \mathcal{X}, \ k = 1, \dots, H_p - 1$$
$$\mathbf{x}(t+H_p) \in \mathcal{X}_{H_p}$$
$$\mathbf{u}(t+k) \in \mathcal{U}, \ k = 0, \dots, H_u - 1$$
$$\Delta \mathbf{u}(t+k) \in \Delta \mathcal{U}, \ k = 0, \dots, H_u - 1$$

where

$$\Delta \mathbf{u}(t+k) = \mathbf{u}(t+k) - \mathbf{u}(t+k-1), \ k = 0, \dots, H_u - 1, \ \mathbf{u}(-1) = 0$$

Model predictive control – pedestrian walking

- Objective J
  - Follow the line (minimize distance of your positions to markers on rope)

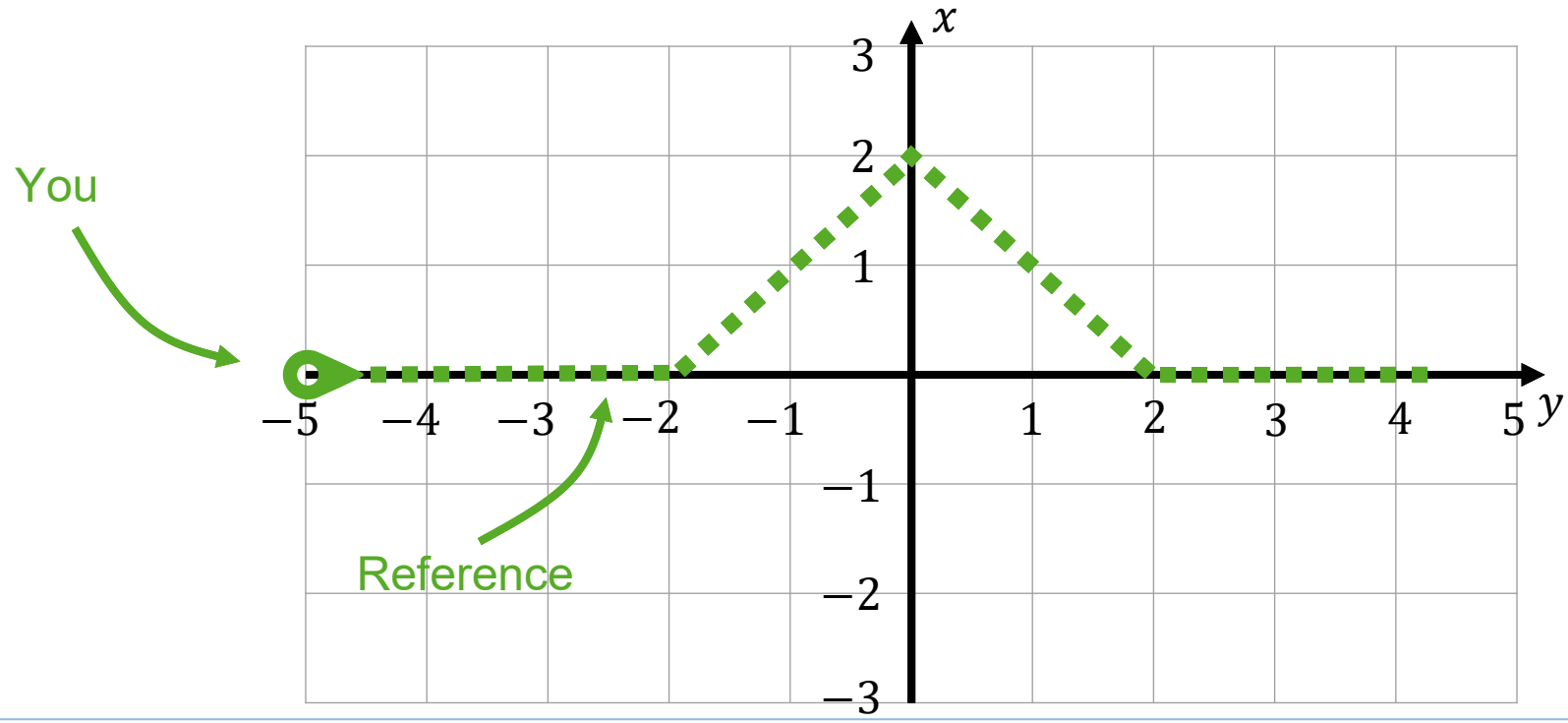
- Constraints
  - Model: $f(x, u)$ (position is integral of velocity)
    - Position: states $\mathbf{x}$
    - Velocity:
      - Forwards: constant, one step per time step
      - Sideways: given by input change $\Delta\mathbf{u}$
  - Input change: max. one step per time step to either side
  - States: must be collision-free (with obstacles)

Model predictive control – pedestrian walking

- Parameters
  - Prediction horizon $H_p = 3$
  - Control horizon $H_u = 1$
  - Time step duration T_s

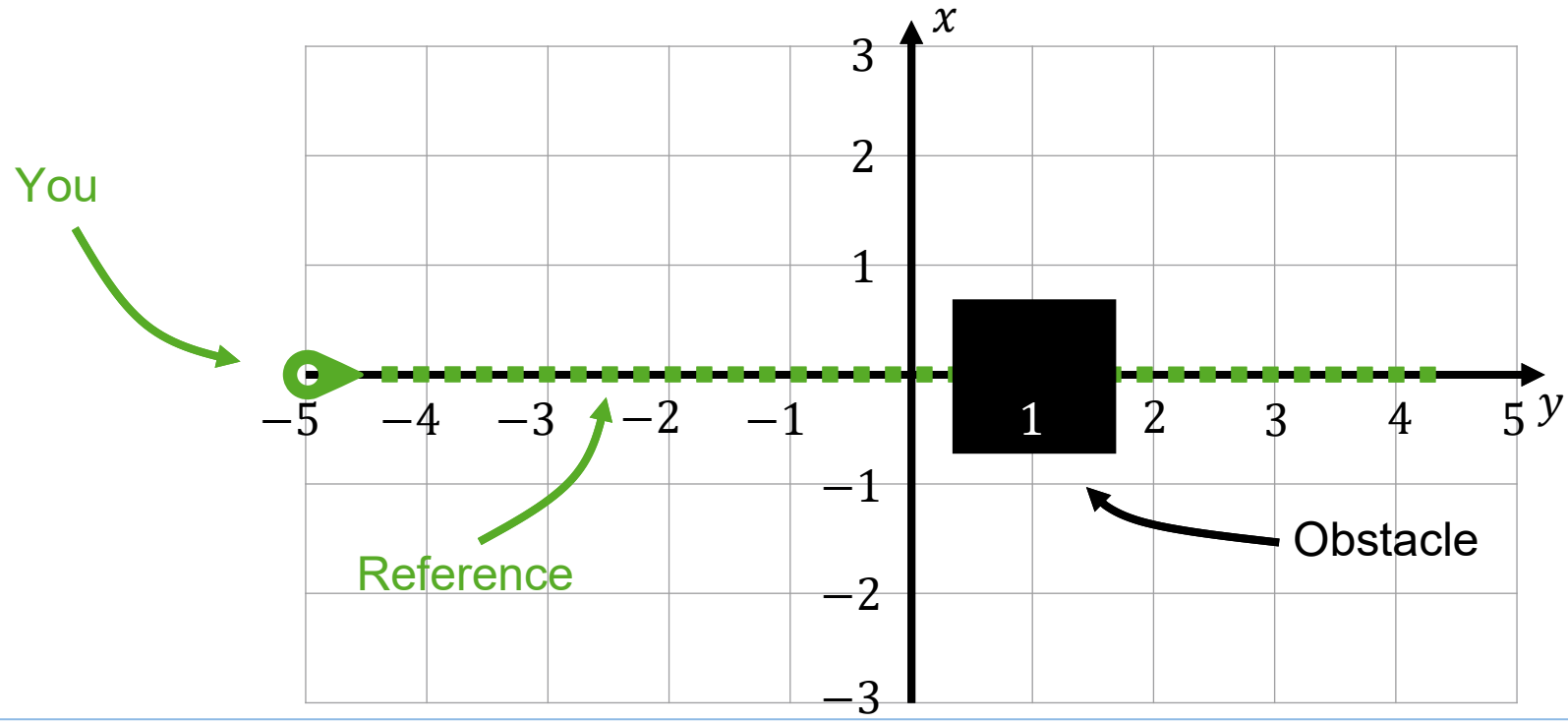
Model predictive control – pedestrian walking

- Process for agent i at time k:
  1. Form MPC optimization problem
  2. Optimize (generate plan): $\mathbf{x}_{\cdot|k}^{(i)}$
  3. Act (according to the first step of the plan): $\mathbf{u}_k^{(i)}$



Model predictive control – pedestrian walking

- Process for agent i at time k:
  1. Form MPC optimization problem
  2. Optimize (generate plan): $\mathbf{x}_{\cdot|k}^{(i)}$
  3. Act (according to the first step of the plan): $\mathbf{u}_k^{(i)}$



Model predictive control

- MATLAB exercise
  - Basic implementation
    - ModelPredictiveControl.m
    - ModelPredictiveControl_background.pdf
  - MPC toolbox **[optional]**
  - B. Alrifaee. MATLAB Simulation of Networked Model Predictive Control for Vehicle Collision Avoidance, 2017. Available: https://doi.org/10.5281/zenodo.1252992 **[optional]**

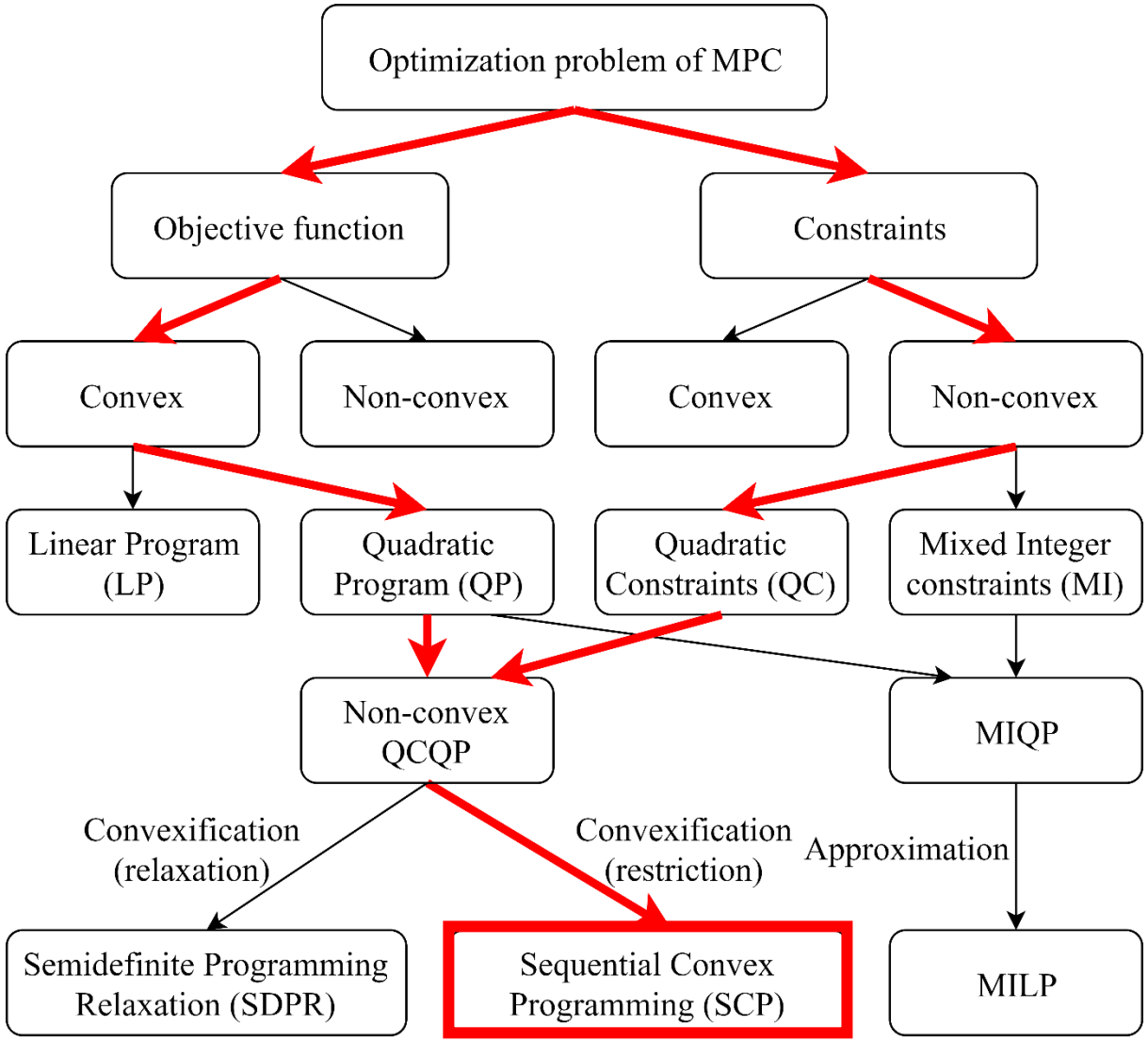
Sequential convex programming

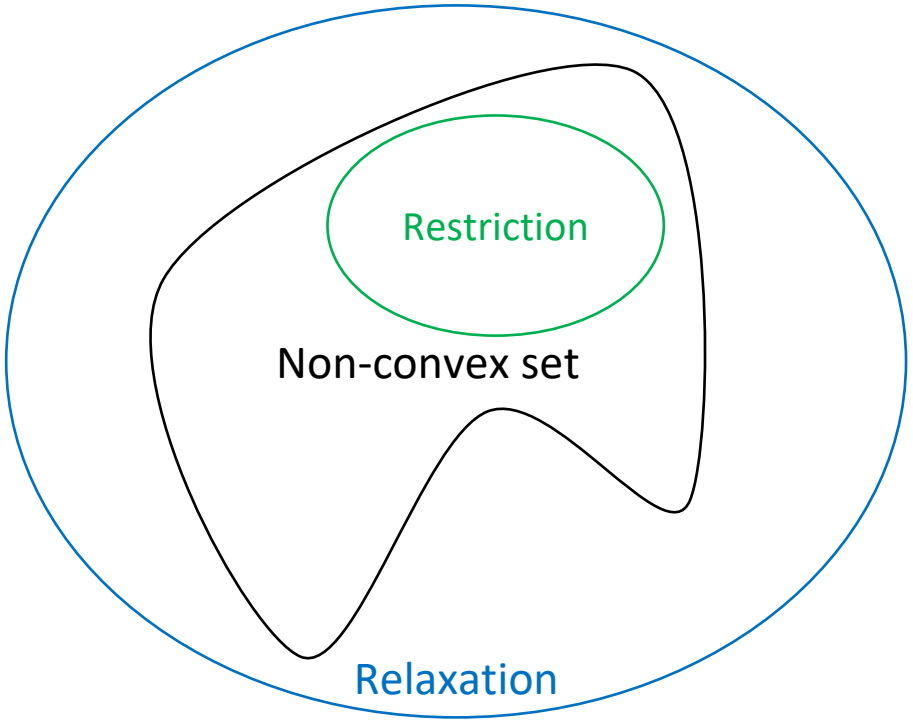
- Flipped classroom
  - Group C should prepare a summary, ca. 15 minutes
- Watching
  - https://youtu.be/upMWYV7S1Y0
- Reading
  - B. Alrifaee. Networked Model Predictive Control for Vehicle Collision Avoidance. PhD thesis, RWTH Aachen University, 2017.
    - Section 4.5, the introduction part, pages 99-100
    - Section 4.5.3, pages 110-119

Sequential convex programming – concept

- “Solves” non-convex optimization problems
- Idea
  - Convexify non-convex parts of the objective function and constraints using convex approximations and preserve their convex parts
  - Solve convex problem
    - Global solution
    - Computationally efficient
  - New approximation around optimal solution
  - Repeat until convergence

Sequential convex programming





Sequential convex programming

- Application to “solve” non-convex QCQPs
- QCQP (Quadratic Constrained Quadratic Program)

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{x}^T \mathbf{P}_0 \mathbf{x} + \mathbf{q}_0^T \mathbf{x} + r_0 \\ \text{subject to:} \quad & \mathbf{x}^T \mathbf{P}_i \mathbf{x} + \mathbf{q}_i^T \mathbf{x} + r_i \leq 0, \; i = 1, \ldots, m \end{aligned}$$

- $\mathbf{P}_i$ is symmetric but not positive semi-definite
  - In the vehicle example, the collision avoidance constraints are concave functions

Sequential convex programming – concept

- “Solves” non-convex optimization problems
- Idea
  - Convexify non-convex parts of the objective function and constraints using convex approximations and preserve their convex parts
  - Solve convex problem
    - Global solution
    - Computationally efficient
  - New approximation around optimal solution
  - Repeat until convergence

Sequential convex programming – algorithm

Algorithm 1 SCP algorithm to "solve" non-convex QCQPs

Input: A non-convex QCQP
Output: Solution vector $\mathbf{x} \in \mathbb{R}^n$

1: $c := 1$ {Iteration counter}
2: Determine starting point $\mathbf{x}_c$
3: Compute the objective value J_c of the non-convex program using $\mathbf{x}_c$
4: Form a convex approximation of the non-convex parts of the inequality constraints using $\mathbf{x}_c$
5: Compute the optimal solution $\mathbf{x}_{c+1}$ of the resulting convex program
6: Compute the objective value J_{c+1} of the non-convex program using $\mathbf{x}_{c+1}$
7: **if** $J_c - J_{c+1} \leq \epsilon$ **then**
8: **return** $\mathbf{x}_{c+1}$
9: **end if**
10: $c := c + 1$
11: **go to** 4

Sequential convex programming – algorithm

Starting point

- May use time-shifted solution from previous time step
- No feasible solution
  - May be bad local minimum
  - Try other starting points

Algorithm 1 SCP algorithm to "solve" non-convex QCQPs

Input: A non-convex QCQP
Output: Solution vector $\mathbf{x} \in \mathbb{R}^n$

1: $c := 1$ {Iteration counter}
2: Determine starting point $\mathbf{x}_c$
3: Compute the objective value J_c of the non-convex program using $\mathbf{x}_c$
4: Form a convex approximation of the non-convex parts of the inequality constraints using $\mathbf{x}_c$
5: Compute the optimal solution $\mathbf{x}_{c+1}$ of the resulting convex program
6: Compute the objective value J_{c+1} of the non-convex program using $\mathbf{x}_{c+1}$
7: **if** $J_c - J_{c+1} \leq \epsilon$ **then**
8: **return** $\mathbf{x}_{c+1}$
9: **end if**
10: $c := c + 1$
11: **go to** 4

Sequential convex programming – algorithm

Convex approximation (local)

- Linearization of quadratic constraints around $\mathbf{x}_c$
  - Best convex approximation of a concave function is its affine approximation*
  - Affine function lies above the concave function at all points, i.e., the affine function is globally upper bound on the concave function
  - Resulting program is a restriction of the original one

Algorithm 1 SCP algorithm to "solve" non-convex QCQPs

Input: A non-convex QCQP
Output: Solution vector $\mathbf{x} \in \mathbb{R}^n$

1: $c := 1$ {Iteration counter}
2: Determine starting point $\mathbf{x}_c$
3: Compute the objective value J_c of the non-convex program using $\mathbf{x}_c$
4: Form a convex approximation of the non-convex parts of the inequality constraints using $\mathbf{x}_c$
5: Compute the optimal solution $\mathbf{x}_{c+1}$ of the resulting convex program
6: Compute the objective value J_{c+1} of the non-convex program using $\mathbf{x}_{c+1}$
7: **if** $J_c - J_{c+1} \leq \epsilon$ **then**
8: **return** $\mathbf{x}_{c+1}$
9: **end if**
10: $c := c + 1$
11: **go to** 4

* D. Zwick. Best Approximation by Convex Functions. The American Mathematical Monthly, 94(6):528–534, 1987. Link

Sequential convex programming – algorithm

Solve convex sub-problem

- Fast
- Iterating around the new solution produces solutions with lower objective values, because convex; and therefore, of original program

Algorithm 1 SCP algorithm to "solve" non-convex QCQPs

Input: A non-convex QCQP
Output: Solution vector $\mathbf{x} \in \mathbb{R}^n$

1: $c := 1$ {Iteration counter}
2: Determine starting point $\mathbf{x}_c$
3: Compute the objective value J_c of the non-convex program using $\mathbf{x}_c$
4: Form a convex approximation of the non-convex parts of the inequality constraints using $\mathbf{x}_c$
5: Compute the optimal solution $\mathbf{x}_{c+1}$ of the resulting convex program
6: Compute the objective value J_{c+1} of the non-convex program using $\mathbf{x}_{c+1}$
7: **if** $J_c - J_{c+1} \leq \epsilon$ **then**
8: **return** $\mathbf{x}_{c+1}$
9: **end if**
10: $c := c + 1$
11: **go to** 4

Sequential convex programming – algorithm

Progress evaluation

- Decreases in objective value of original problem indicate progress

Algorithm 1 SCP algorithm to "solve" non-convex QCQPs

Input: A non-convex QCQP
Output: Solution vector $\mathbf{x} \in \mathbb{R}^n$

1: $c := 1$ {Iteration counter}
2: Determine starting point $\mathbf{x}_c$
3: Compute the objective value J_c of the non-convex program using $\mathbf{x}_c$
4: Form a convex approximation of the non-convex parts of the inequality constraints using $\mathbf{x}_c$
5: Compute the optimal solution $\mathbf{x}_{c+1}$ of the resulting convex program
6: Compute the objective value J_{c+1} of the non-convex program using $\mathbf{x}_{c+1}$
7: **if** $J_c - J_{c+1} \leq \epsilon$ **then**
8: **return** $\mathbf{x}_{c+1}$
9: **end if**
10: $c := c + 1$
11: **go to** 4

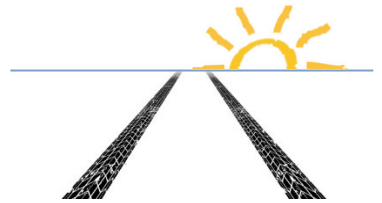
Sequential convex programming – discussion

- Fast (in our applications <100ms)
- Locally optimal solution of QCQP problem
- SCP sub-problem is restriction of QCQP
  - SCP feasible implies QCQP feasible
  - Feasibility issues, SCP too restrictive
  - Solution: slack variable

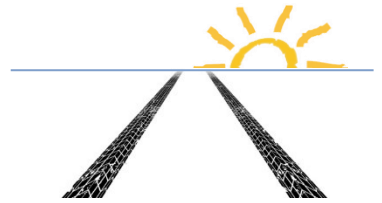
Model predictive control – vehicle example

Objective Function

- Follow a predefined reference trajectory
  - Minimize the distances between the vehicle position and the reference trajectory



- Only accept small input changes
  - Keep the steering angle changes as small as possible



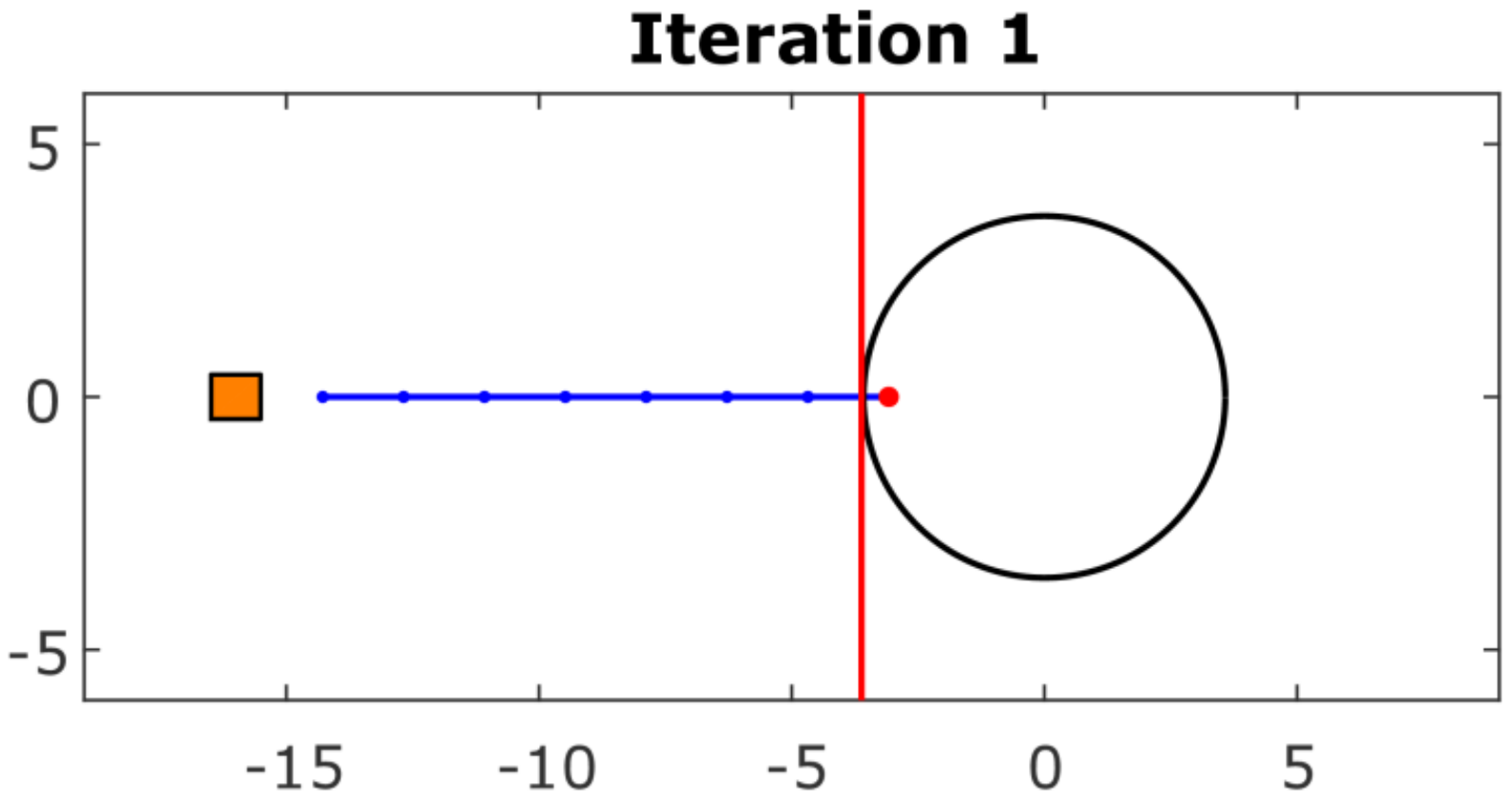
Model predictive control – vehicle example

Constraints

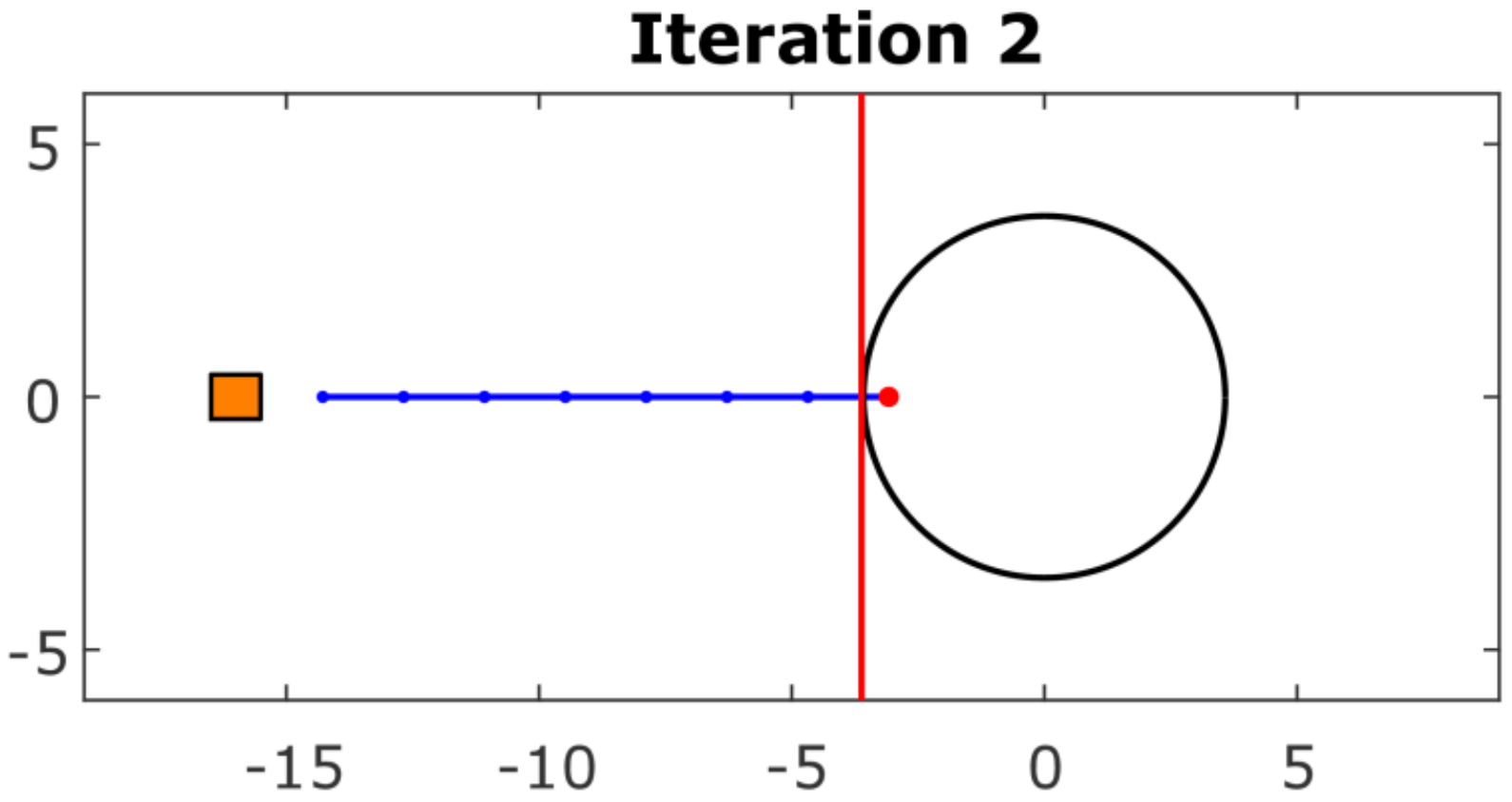
- Input constraints, do not exceed
  - A maximum steering angle
  - A maximum change of the steering angle per time step
  - A maximum lateral acceleration

- Collision avoidance constraints
  - Do not collide with other vehicles or obstacles

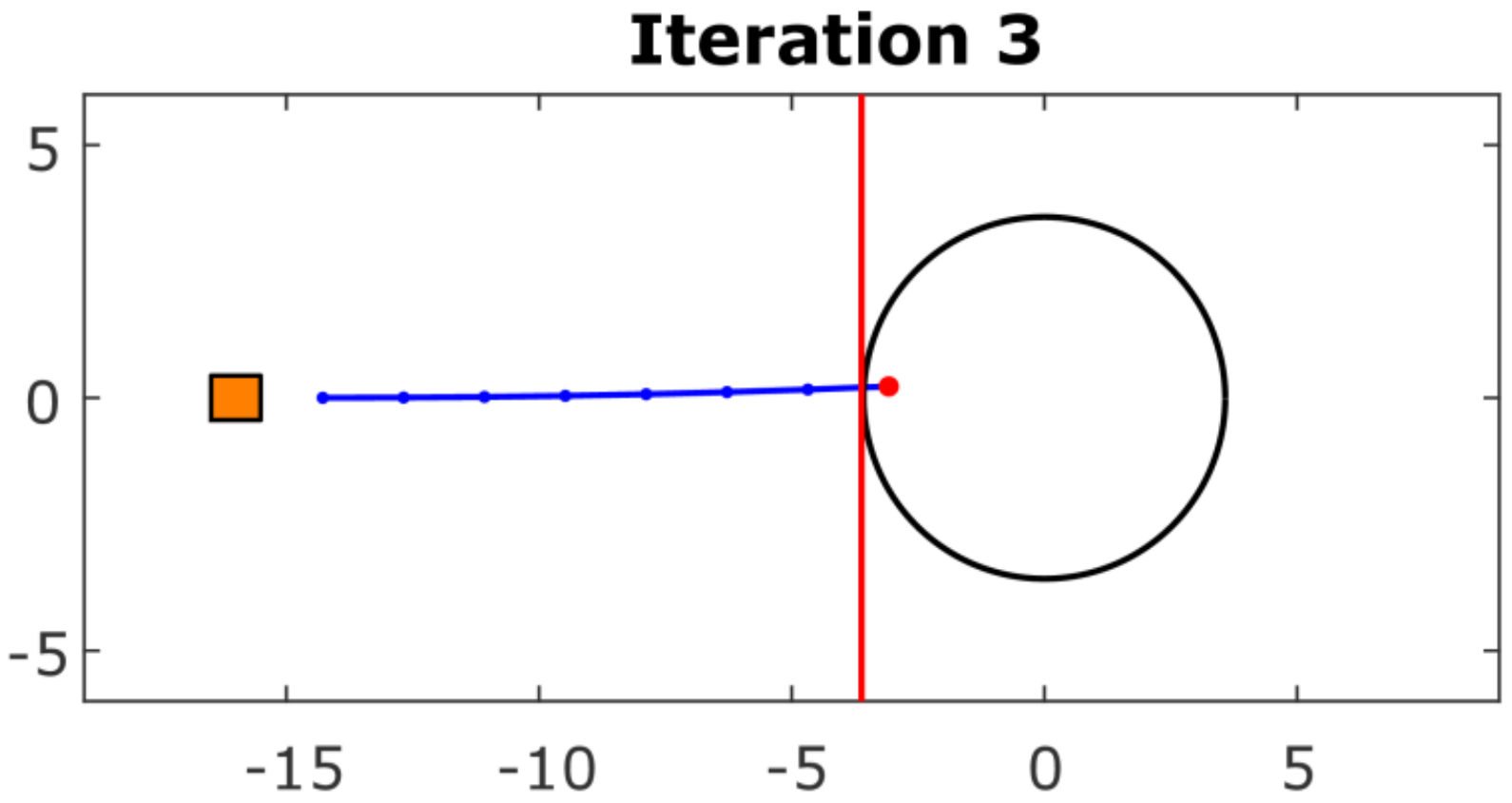
Sequential convex programming – vehicle example



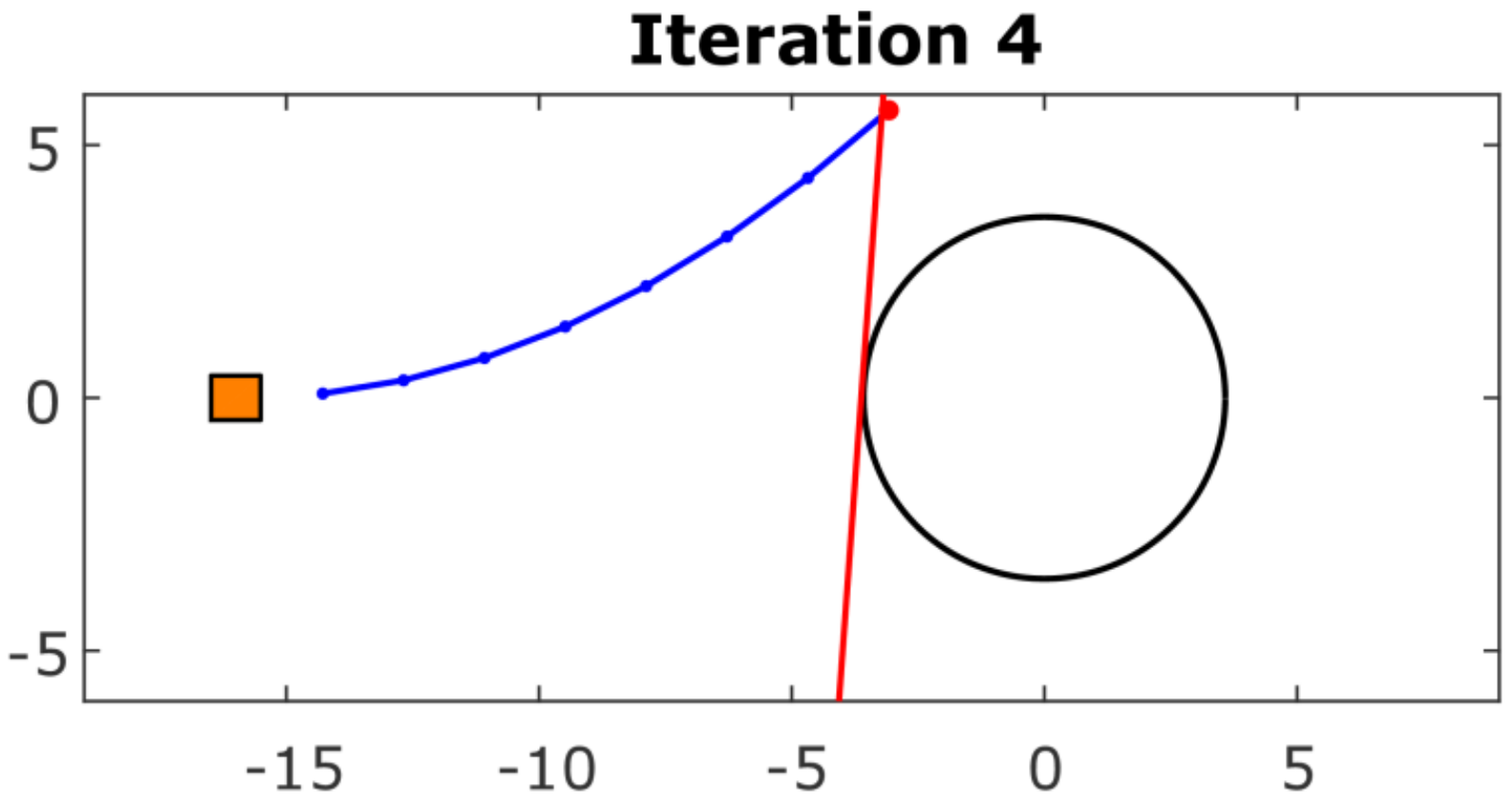
Sequential convex programming – vehicle example



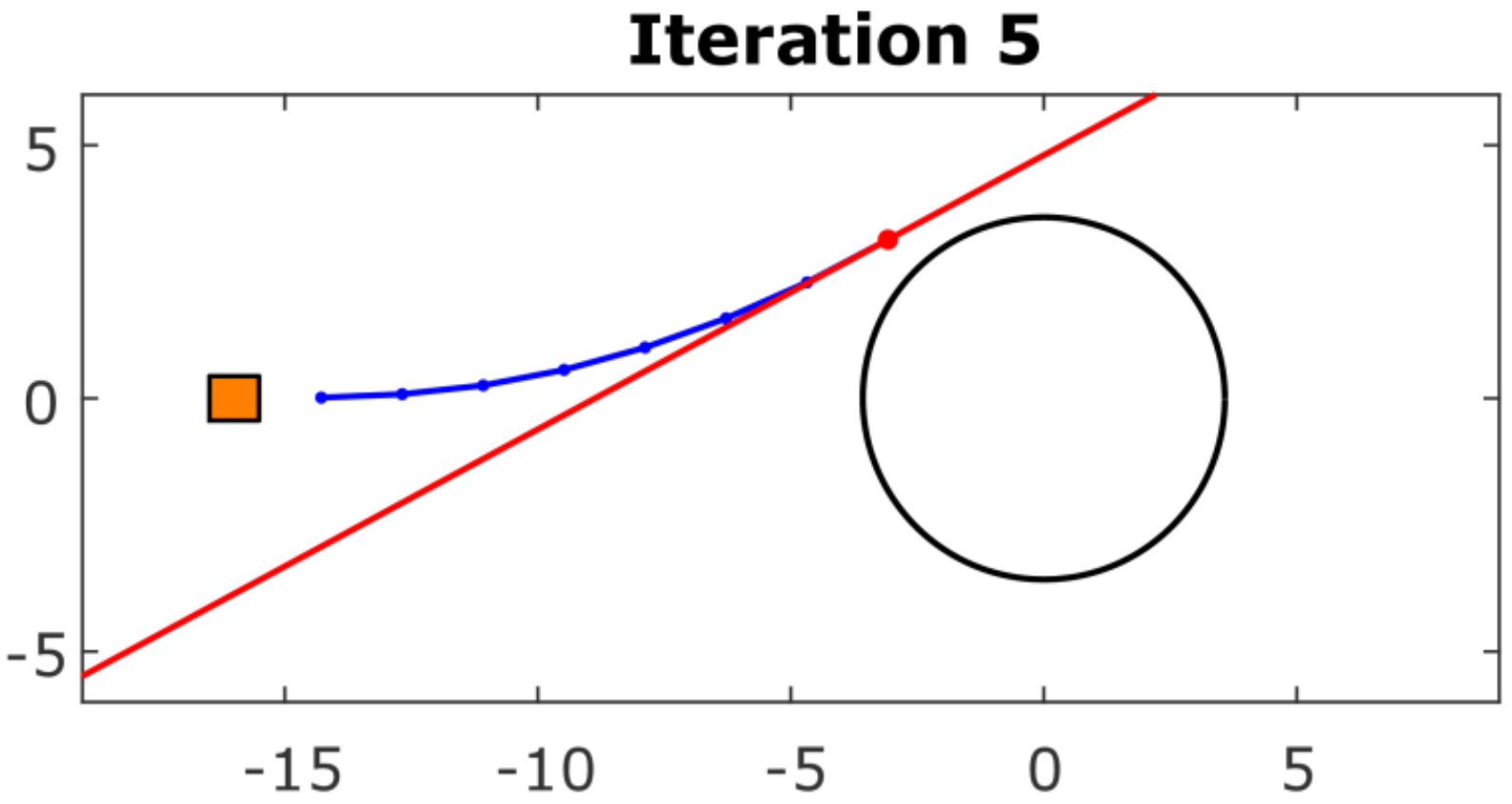
Sequential convex programming – vehicle example



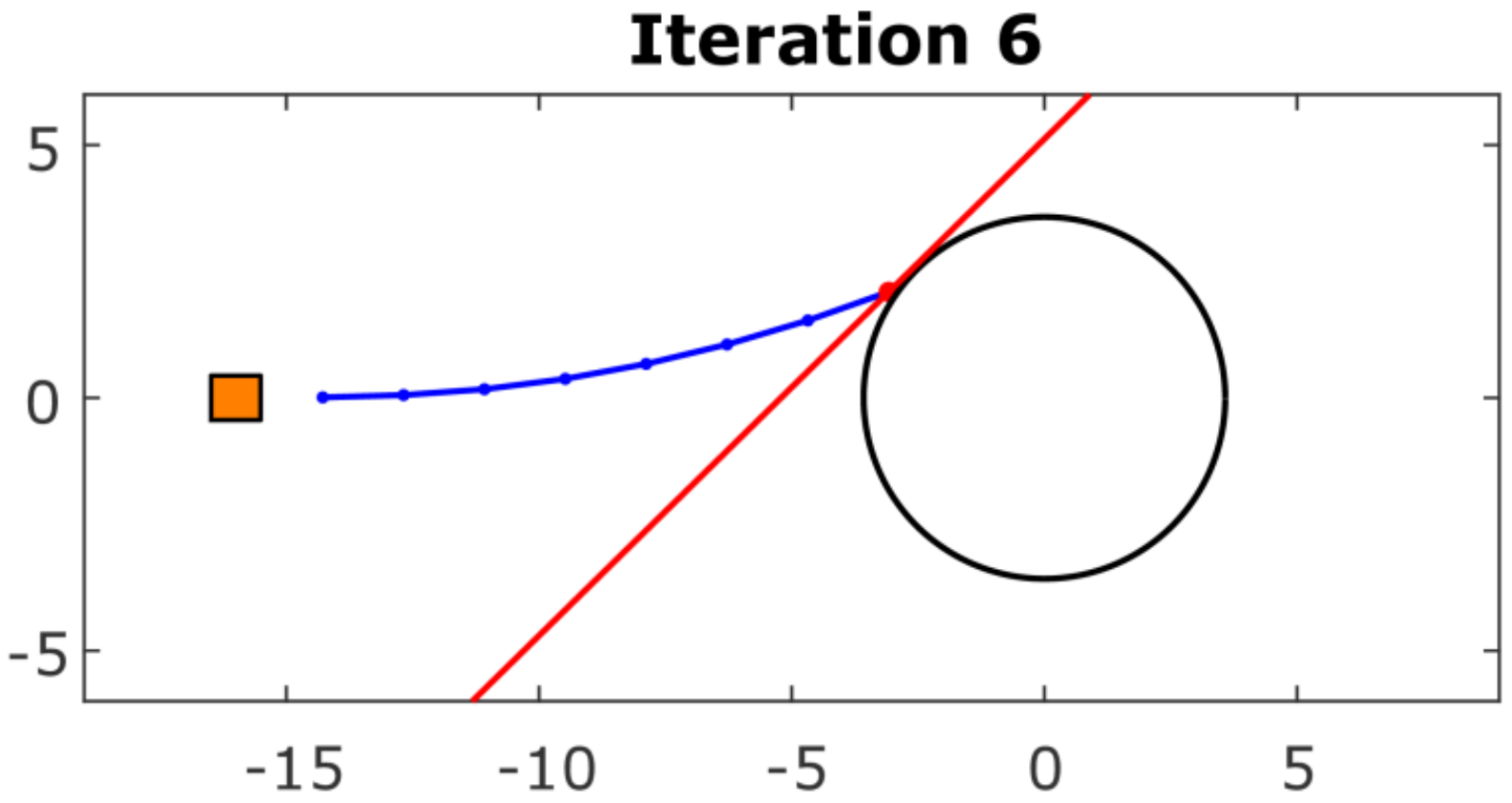
Sequential convex programming – vehicle example



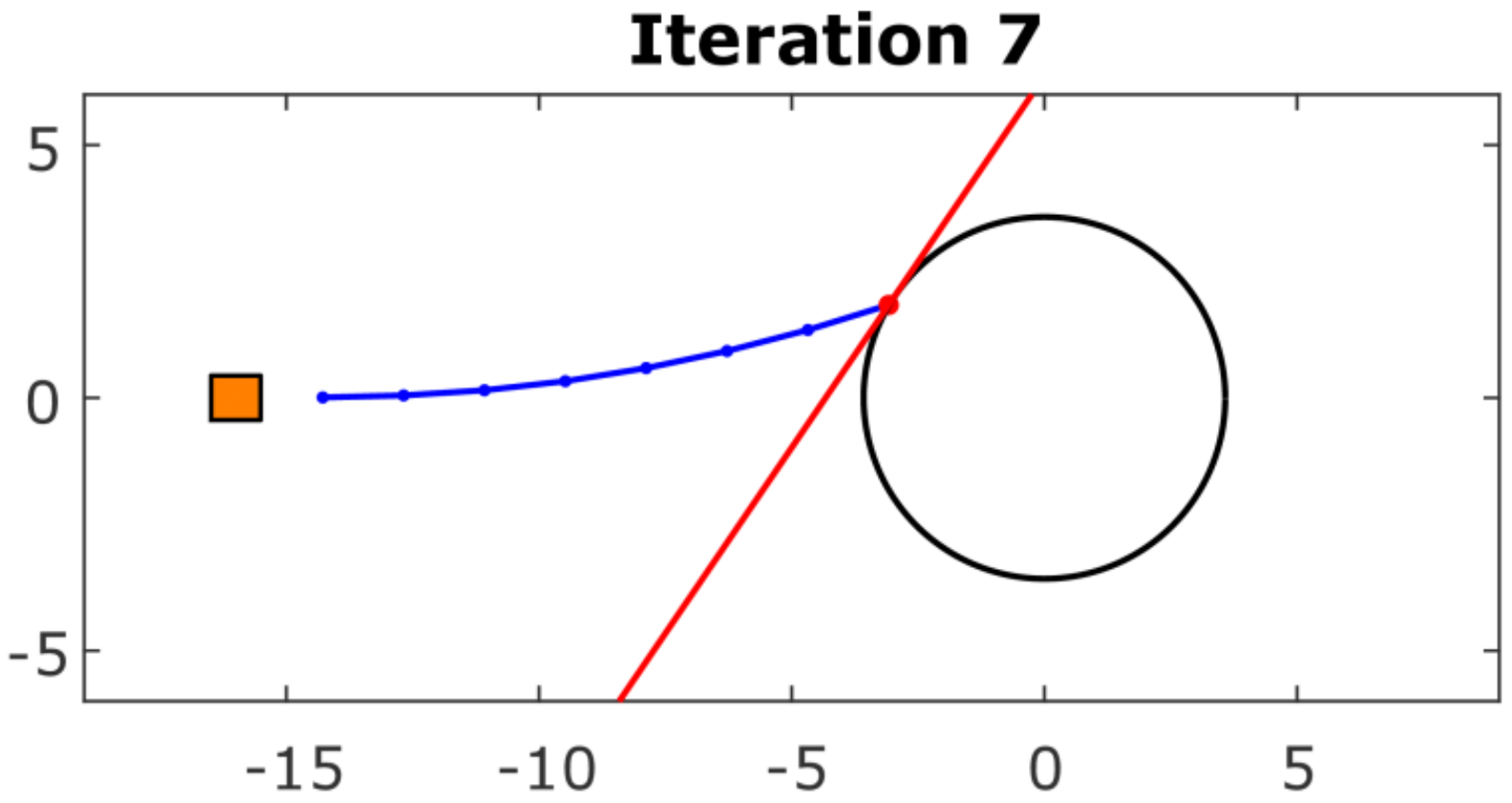
Sequential convex programming – vehicle example



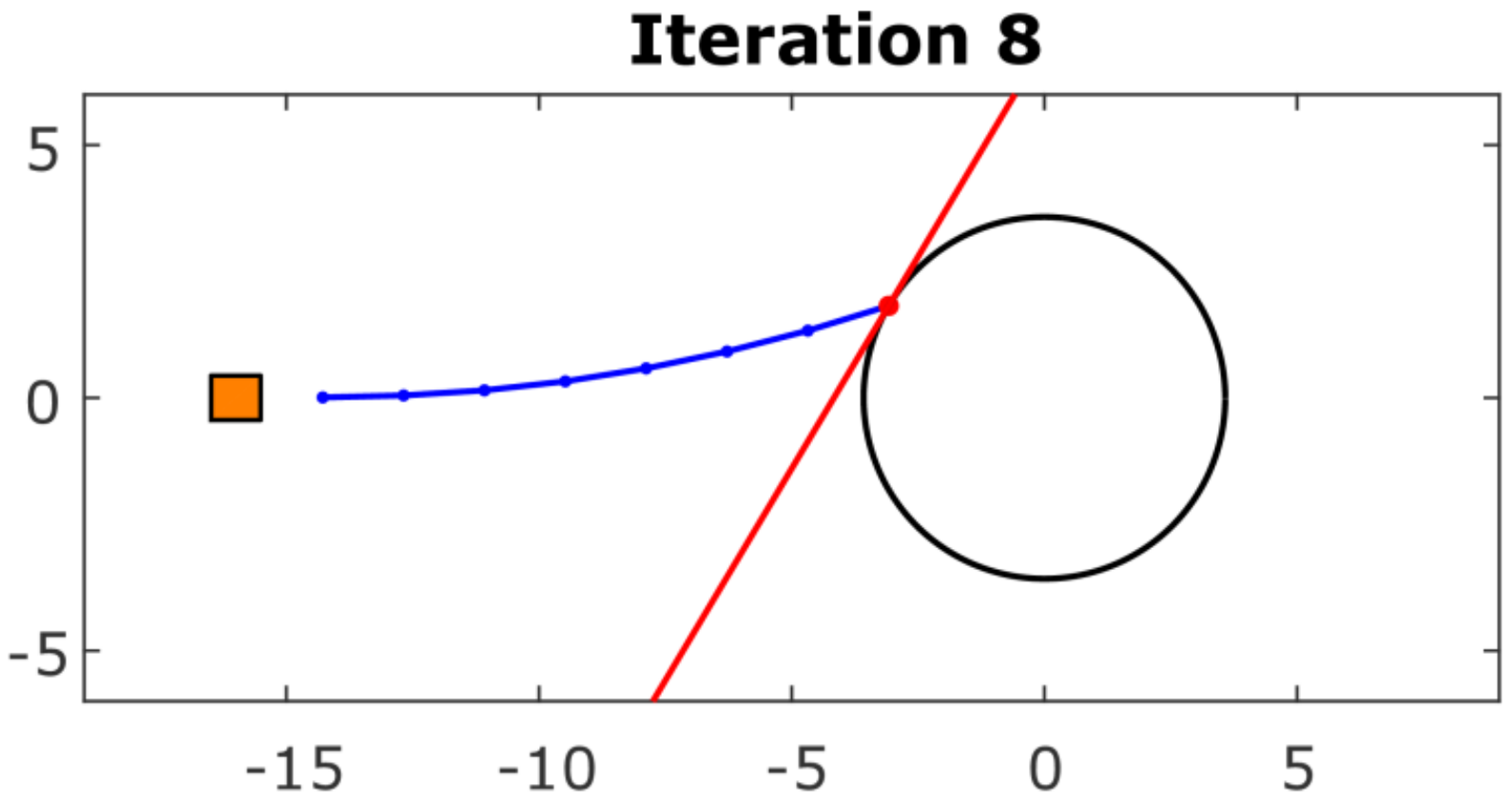
Sequential convex programming – vehicle example



Sequential convex programming – vehicle example

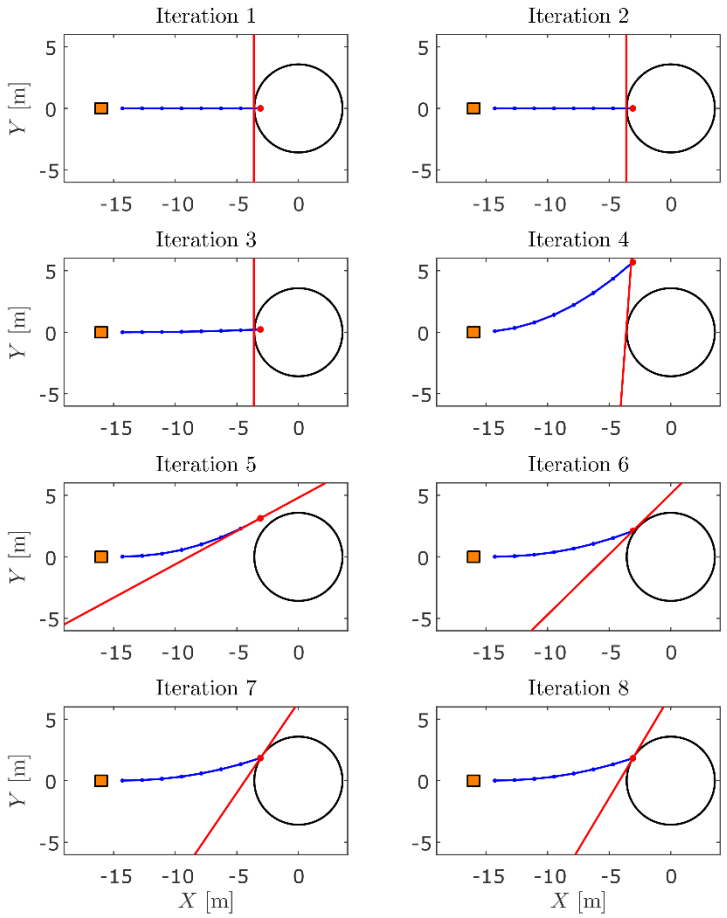


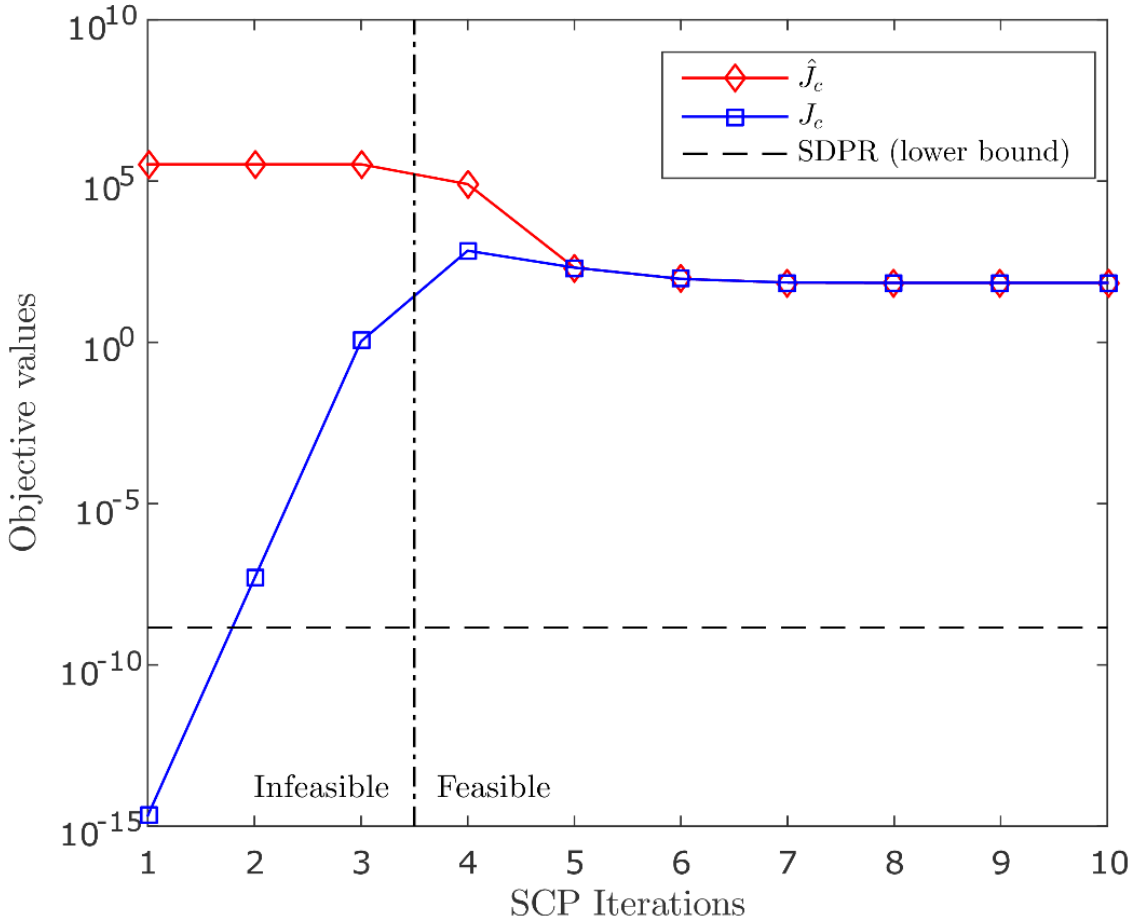
Sequential convex programming – vehicle example



Sequential convex programming – vehicle example

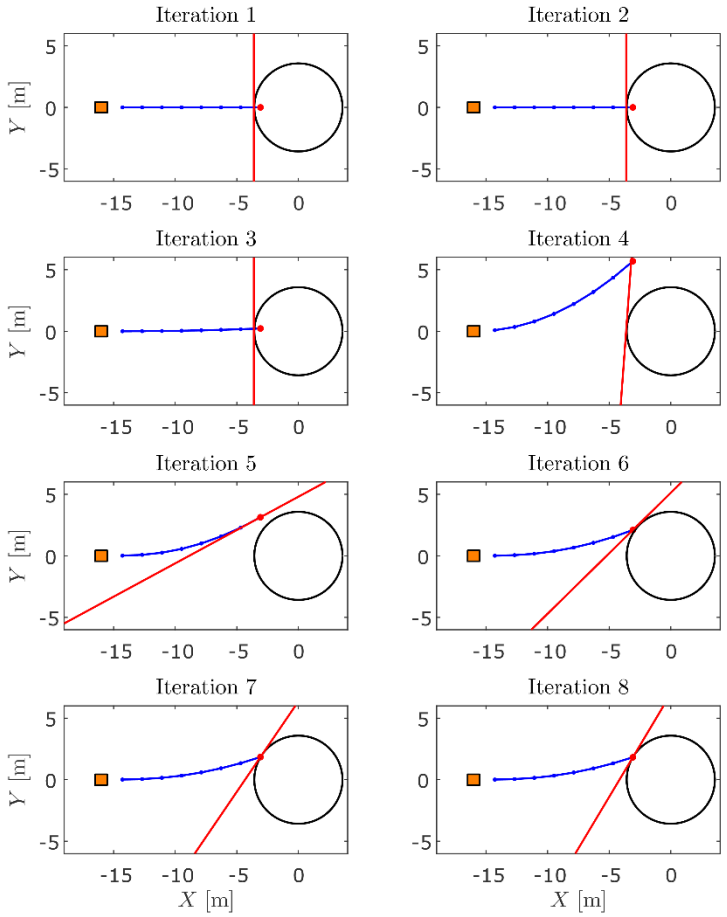
- Convergence of SCP: $\widehat{J_c}$, Progress of SCP: J_c

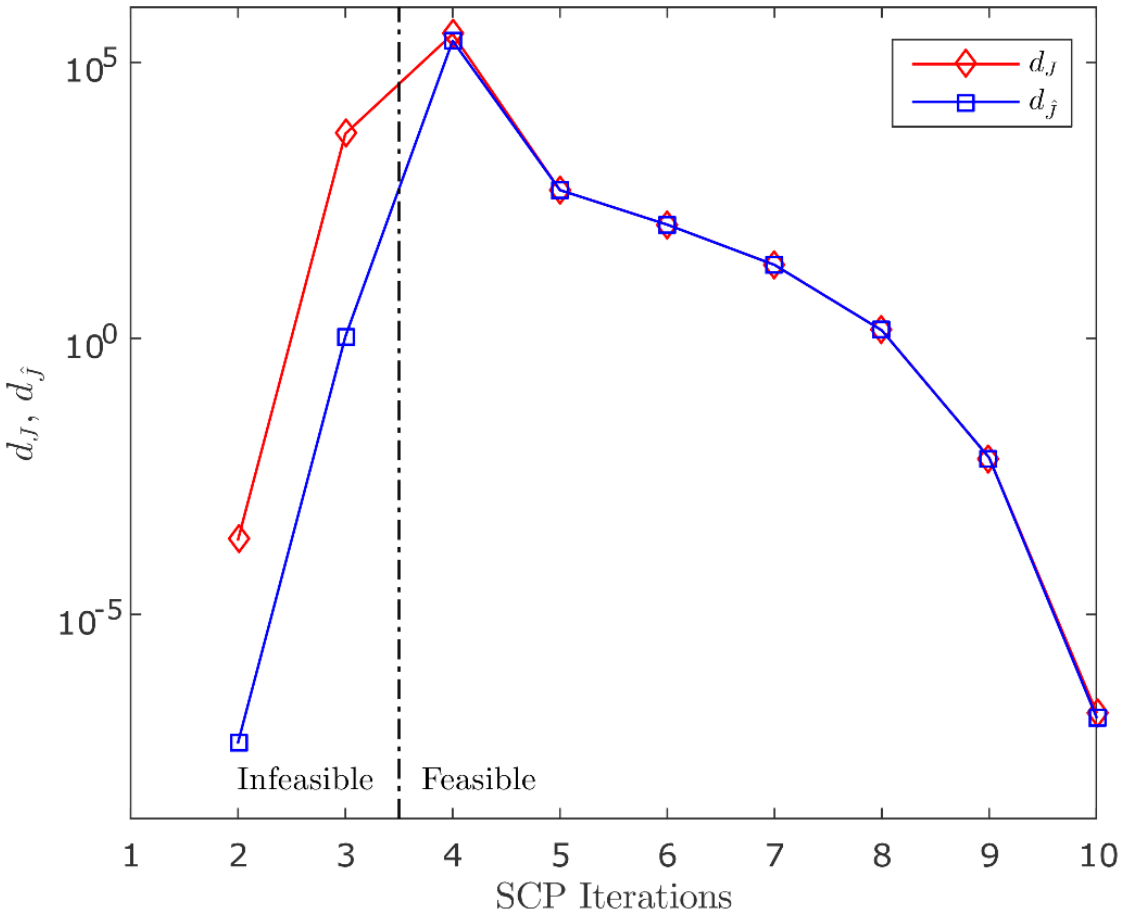




Sequential convex programming – vehicle example

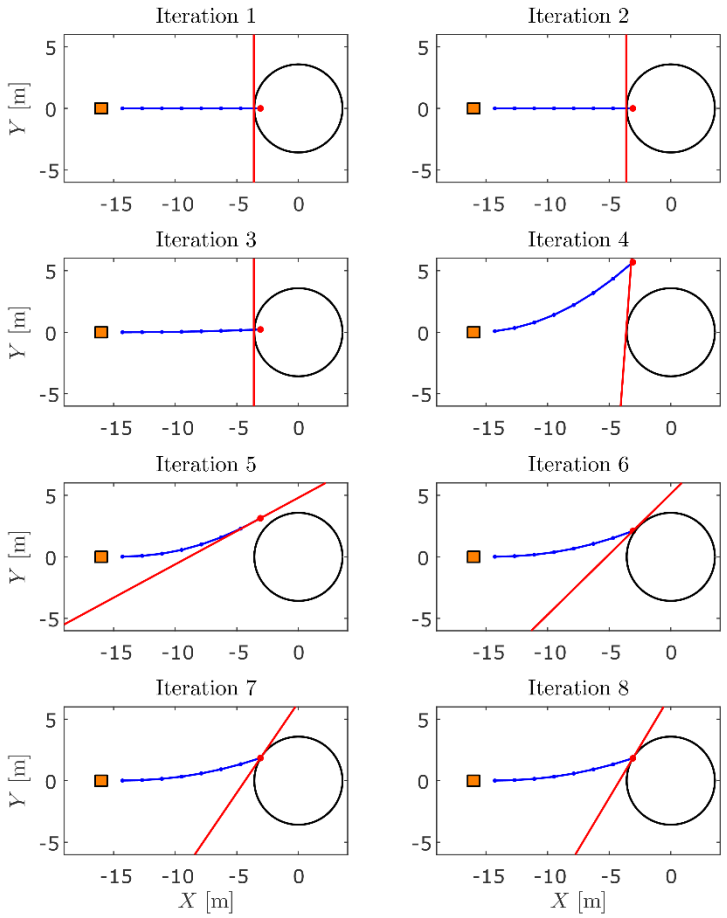
- Predicted and real decrease of the objective value
  - $d_{\hat{J}} \leq d_J$ because approximate program is a restriction

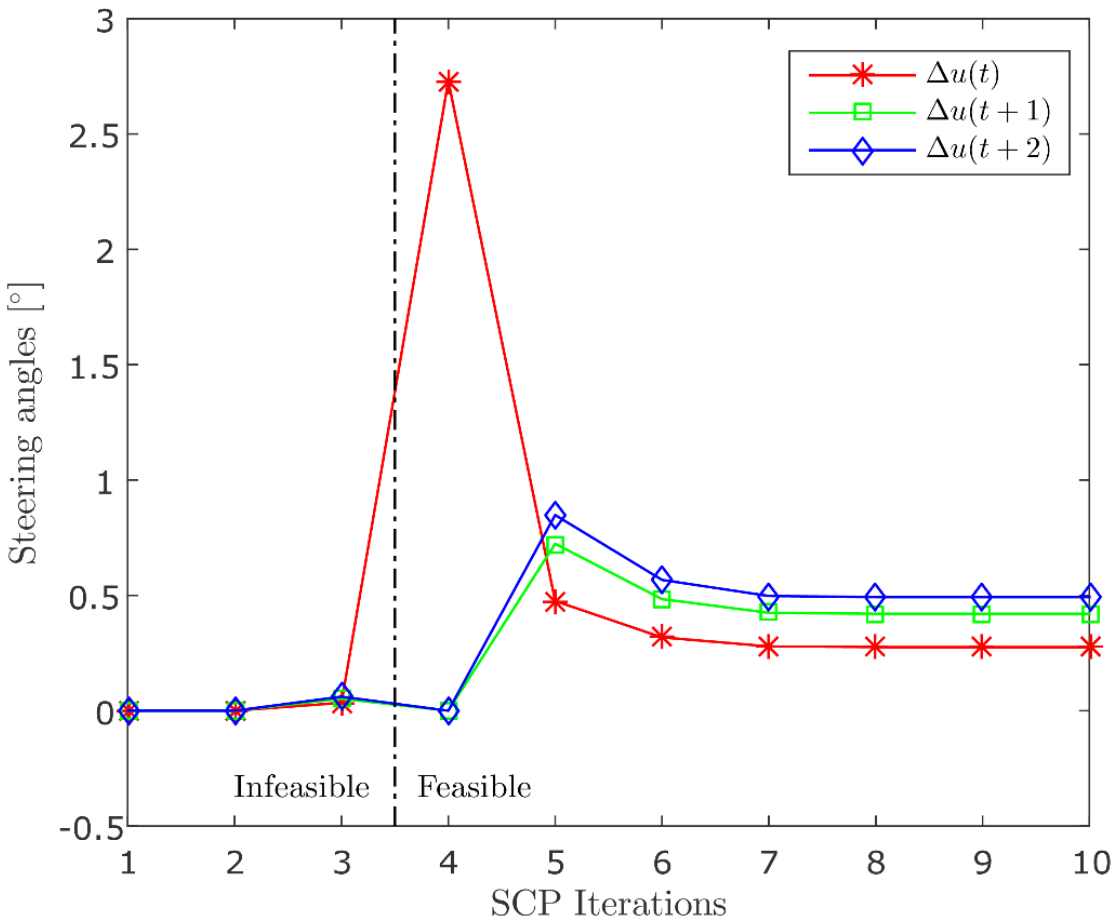




Sequential convex programming – vehicle example

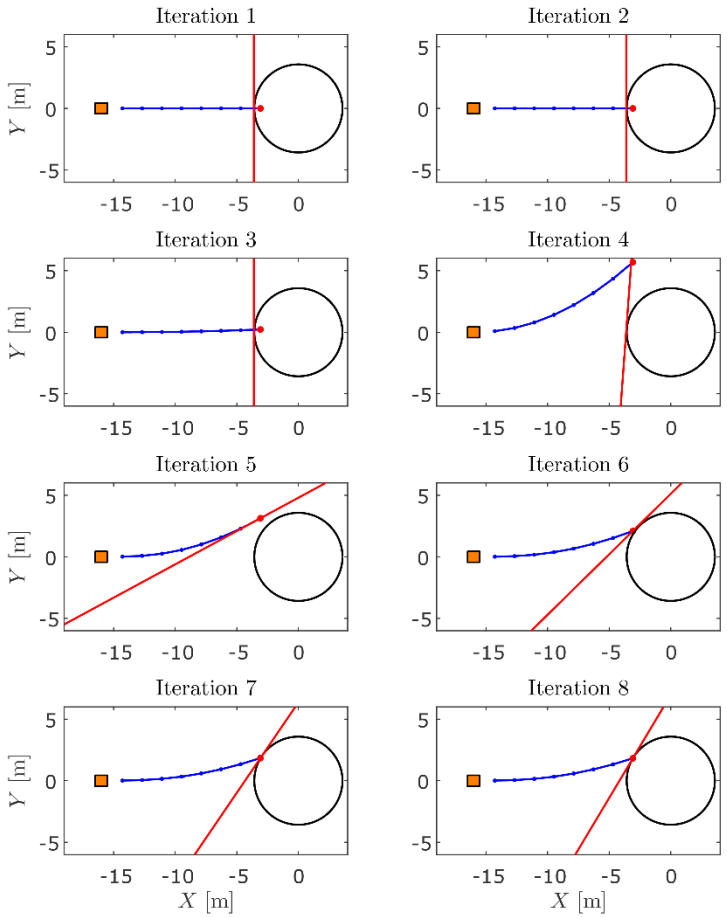
- Convergence of the control input (control horizon = 3)

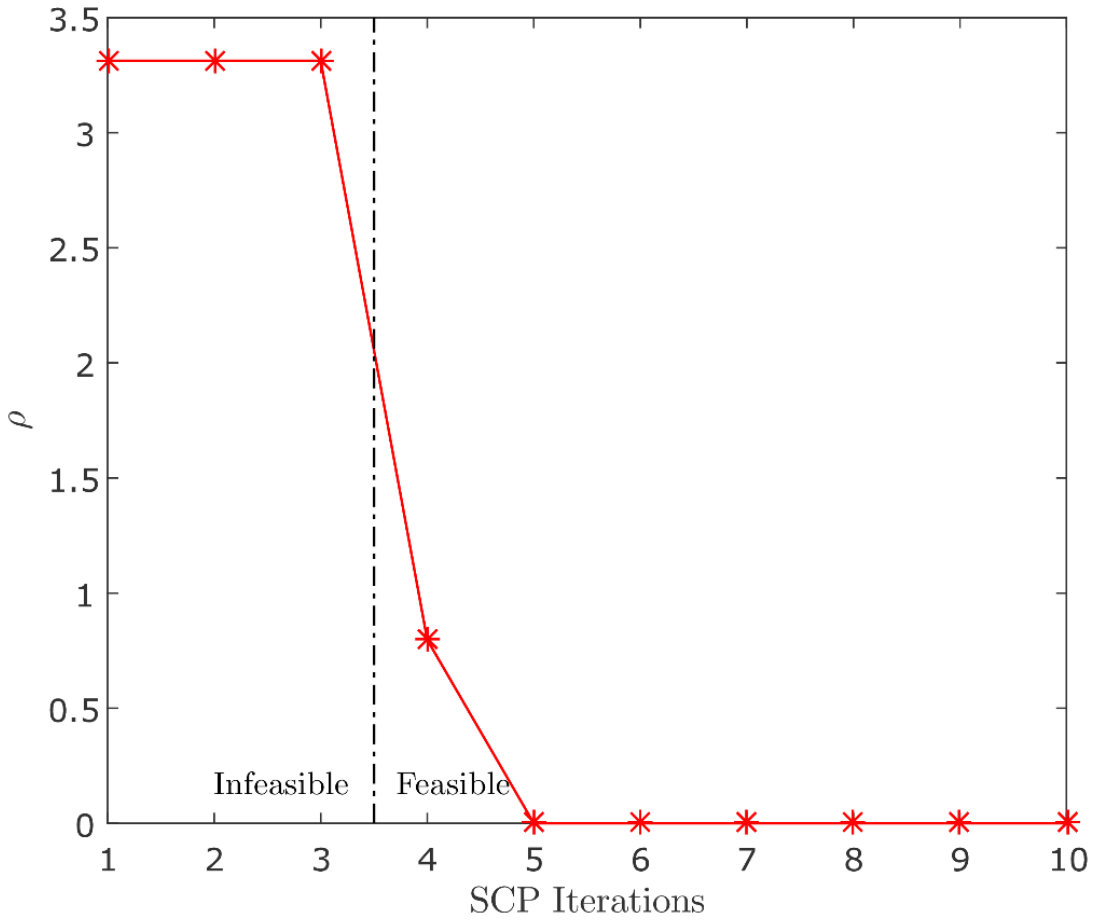




Sequential convex programming – vehicle example

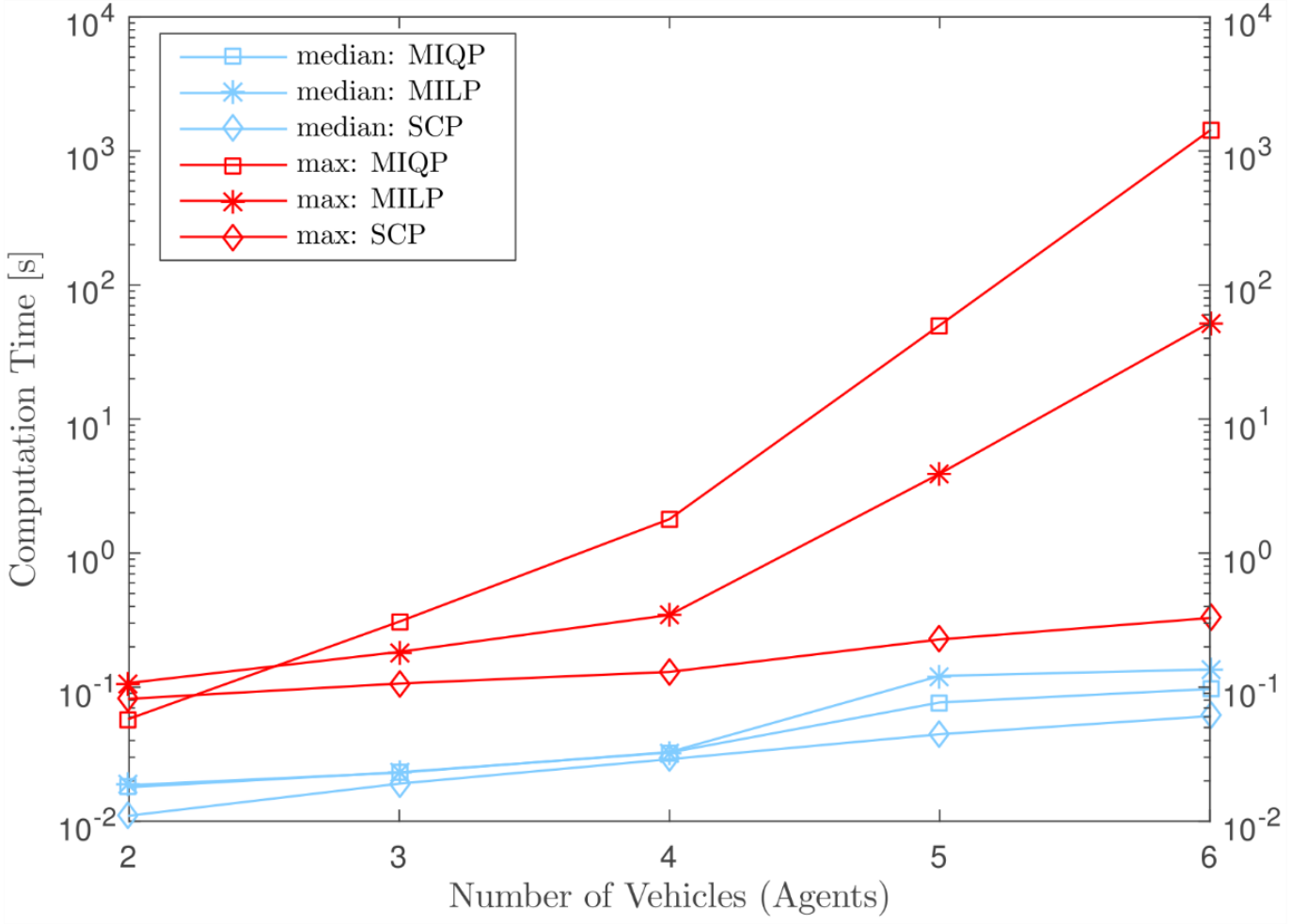
- Slack variable, progress of SCP





Sequential convex programming – vehicle example

- Computation time



Sequential convex programming – vehicle example

- Video of “Frogger” simulation
- Video of experimental results
- Video of trust
- Vehicle racing, videos of simulation results
  - https://youtu.be/t4tkZA8yZkg
  - https://youtu.be/_BxhYhIbORk

Sequential convex programming

- MATLAB exercise
  - B. Alrifaee. MATLAB Simulation of Networked Model Predictive Control for Vehicle Collision Avoidance, 2017. Available: https://doi.org/10.5281/zenodo.1252992
    - Function \controller\SCP_optimizer.m

Sequential convex programming – conclusion

- SCP finds a (good) upper bound on the non-convex optimization problem starting from the time-shifted last solution of MPC

Next Part

Network and Distribution

- Networked model predictive control