# Race 3 The Best One You Can Do

Part 1. Race 3 Tracks

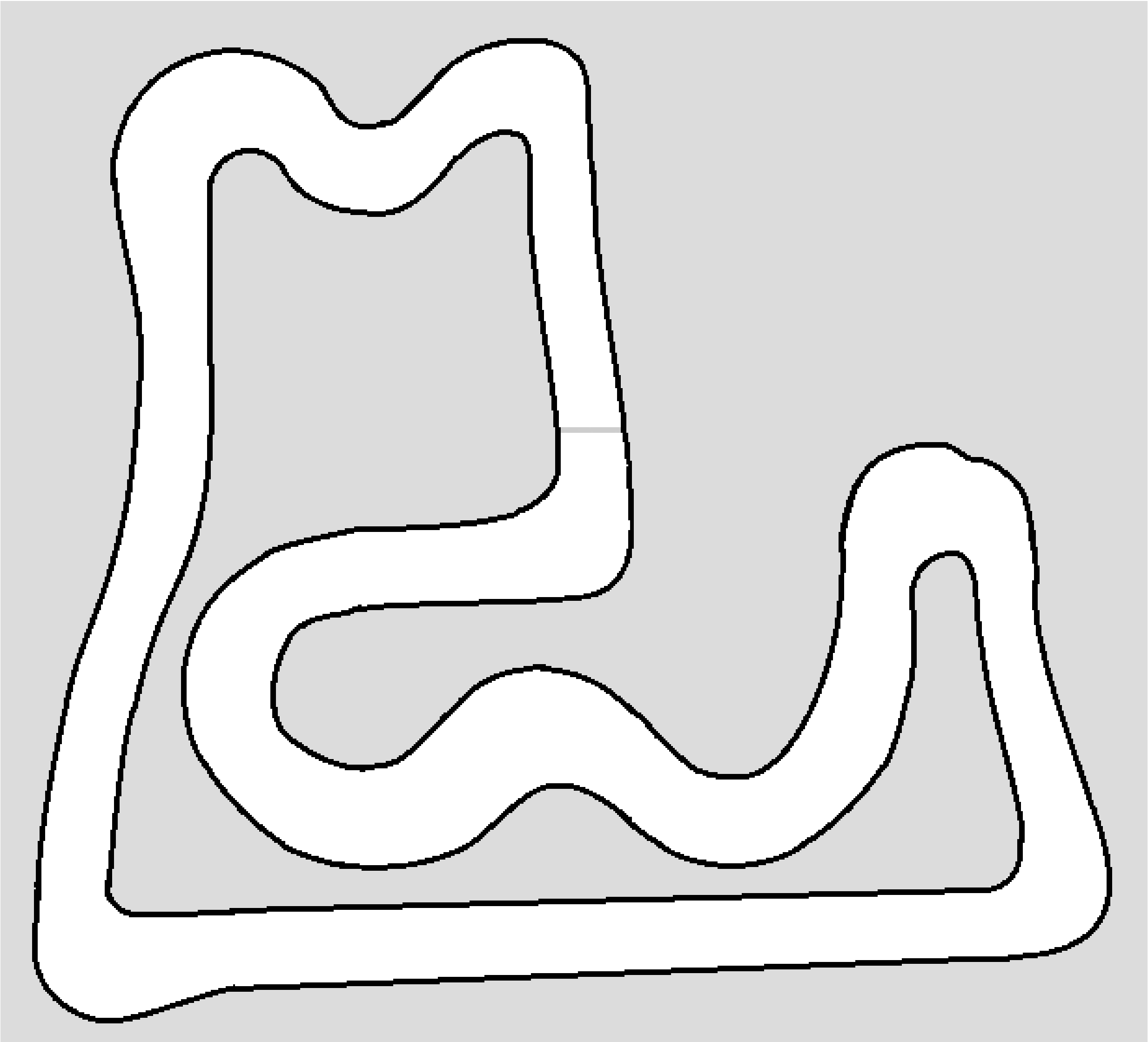
Part 2: Race Line Optimization

References:

https://f1tenth.org and UPenn ESE 680 Slides

# Track 1 – Nice and Smooth

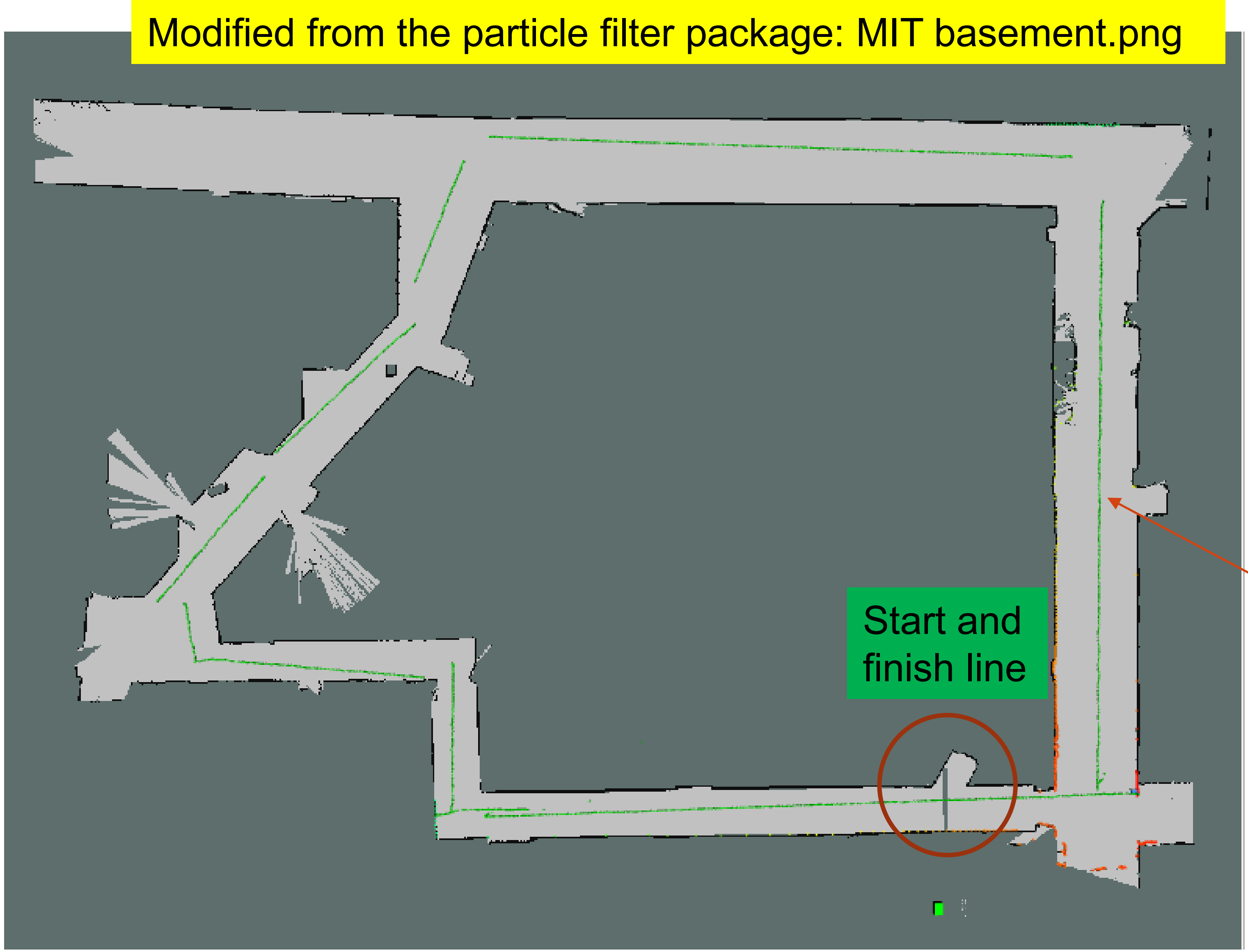
LEHIGH UNIVERSITY

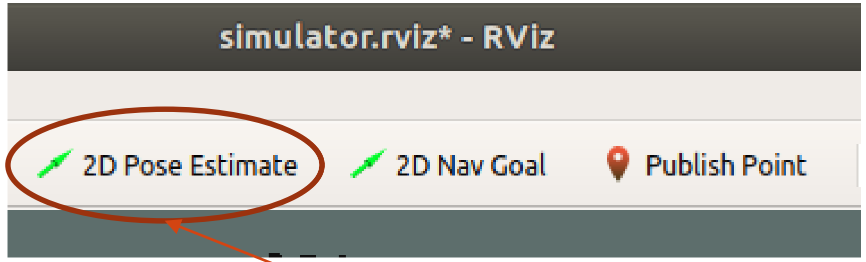


- Race 3 Rules and Grading:
  - Complete 3 laps in 120 sec on each of the two tracks: race3track1.pgm and race3track2.pgm
  - Use your choice of algorithms: may use different algorithms for different tracks. More advanced algorithms are encouraged.
  - Grading (similar to Race 2):
    - 90 points for race and programs,
    - 60 points for report
    - Weights adjusted for individual effort
  - Help sessions: TuTh 3-5 pm zoom ID 824-078-908, M-T-W-T 11:30 am-1 pm, zoom ID 661-616-855.
  - Check out the new website at: f1tenth.org, all course slides available in "Learn" tab.

# Track 2 – Real Map Obtained by SLAM

LEHIGH UNIVERSITY




Step 1: Use 2D Pose Estimate with keyboard operation: w for forward, k to toggle keyboard on/off – car would go and stop

Step 2: while running the car, run waypoint_logger.py to log points

Step 3: Clean out the ones with zero speed as there would be lots of data points logged when the car stopped;

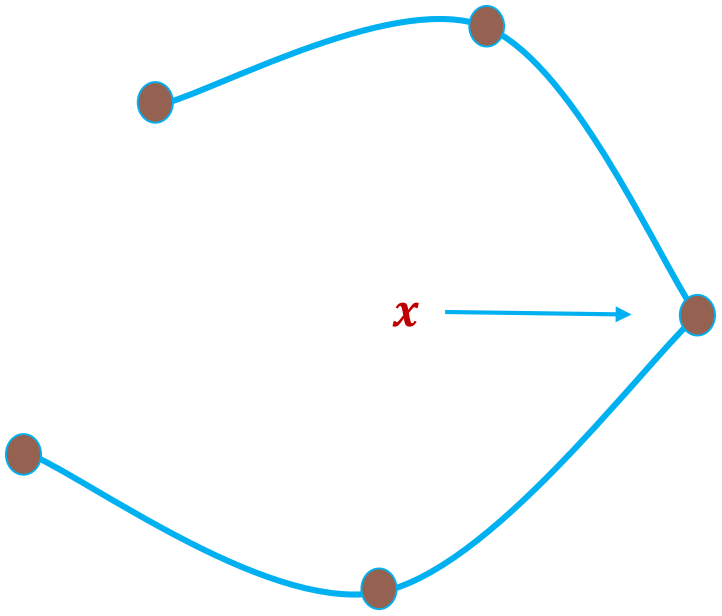
Step 4: Race line interpolation

# How to design a smooth path?

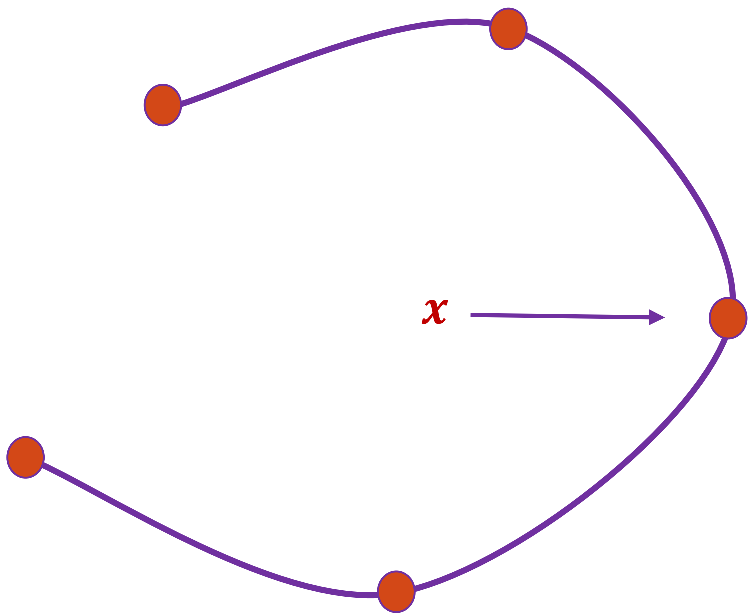


- Continuities defined:

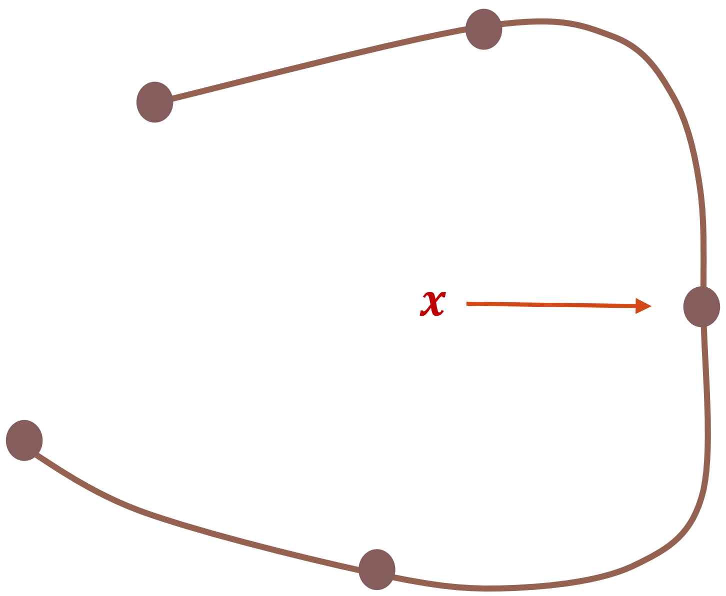
$C_0$ Continuity:
continuous in position
$x^- = x^+$




$C_0$ and $C_1$ Continuity:
continuous in position
and tangent line
$x^- = x^+, \dot{x}^- = \dot{x}^+$



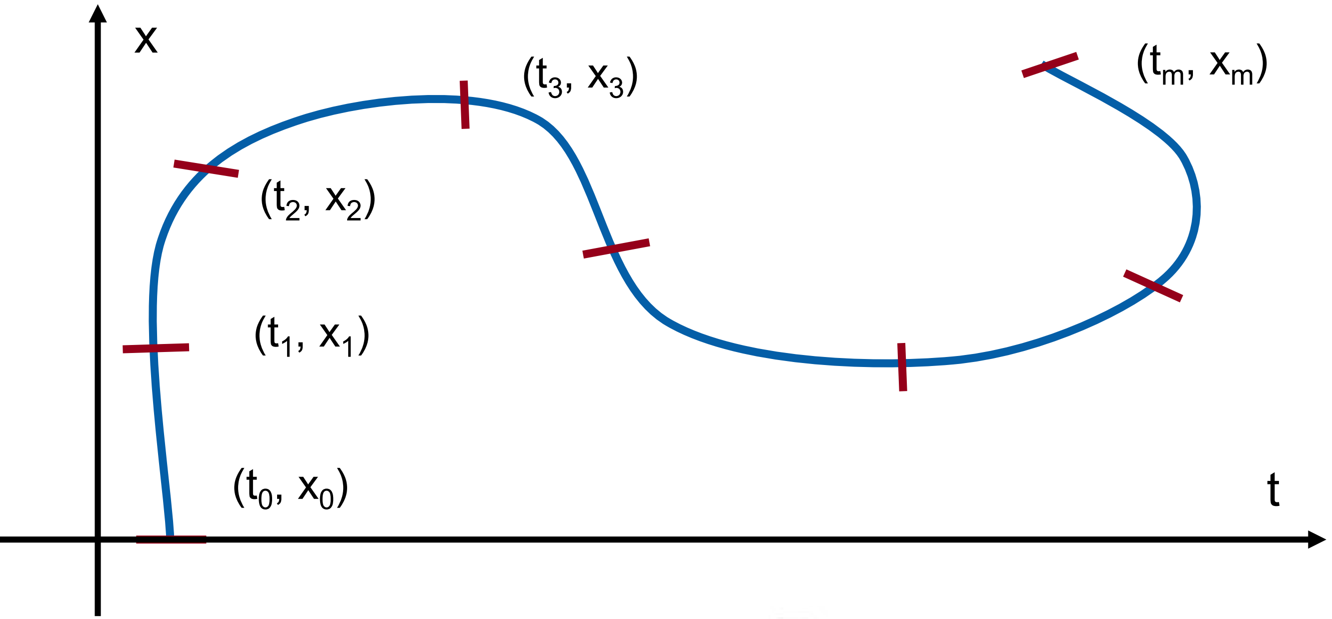

$C_0$, $C_1$ and $C_2$ Continuity:
continuous in position,
tangent line, and curvature
$x^- = x^+, \dot{x}^- = \dot{x}^+, \ddot{x}^- = \ddot{x}^+$

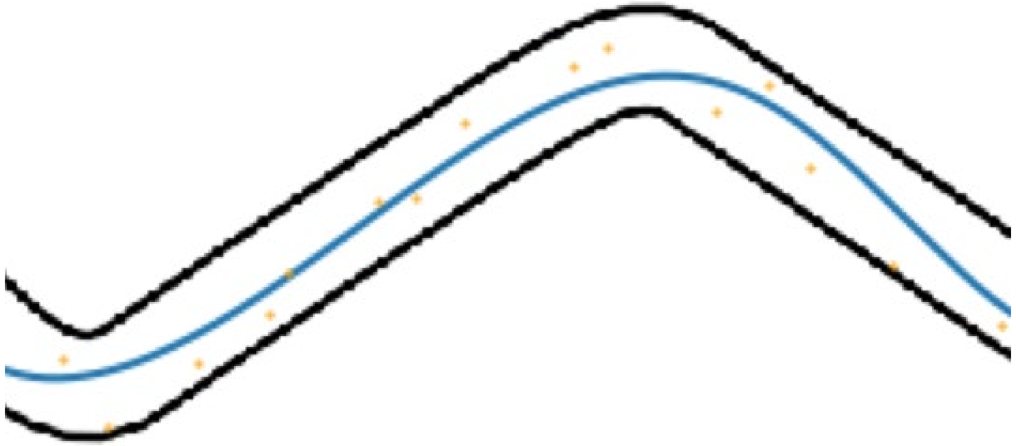



Modified from Upenn ESE680 Lecture 17 Slides

# Cubic Spline Interpolation





- Spline interpolation for m points

$$t = [t_0 \quad t_1 \quad t_2 \quad \dots \quad t_m]^T$$
$$x = [x_0 \quad x_1 \quad x_2 \quad \dots \quad x_m]^T$$

Where t is path, x is the location of goal point, subscript is order of visit

- Design algorithm is to find coefficient $c_{m,k}$

$$x(t) = \begin{cases} x_1(t) = c_{1,3}t^3 + c_{1,2}t^2 + c_{1,1}t + c_{1,0}, & t_0 \le t < t_1 \\ x_2(t) = c_{2,3}t^3 + c_{2,2}t^2 + c_{2,1}t + c_{2,0}, & t_1 \le t < t_2 \\ \dots & \\ x_m(t) = c_{m,3}t^3 + c_{m,2}t^2 + c_{m,1}t + c_{m,0}, & t_{m-1} \le t < t_m \end{cases}$$

Subject to continuity constraints:

$\boldsymbol{x}^- = \boldsymbol{x}^+, \dot{\boldsymbol{x}}^- = \dot{\boldsymbol{x}}^+, \ddot{\boldsymbol{x}}^- = \ddot{\boldsymbol{x}}^+$ or

$x_i(t_i) = x_{i+1}(t_i) = x_i,\ \dot{x}_i(t_i) = \dot{x}_{i+1}(t_i)$,
$\ddot{x}_i(t_i) = \ddot{x}_{i+1}(t_i)$, for $i = 0, 1, \dots, m$.

- There are *4m* unknowns and *4m* constraints



Upenn ESE680 RRT Lab project description

# What are the constraints?



- Dynamics of the car:

Centrifugal Matrix

$$R(q)u = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + d(q)$$

Control Matrix

Control Inputs

Mass Matrix

Configuration Vector

Force as a function of configuration

- Control Inputs:

$$(\dot{q}^2, \ddot{q}, u) \in \mathcal{C}(q)$$

Constraints on velocity

Constraints on acceleration

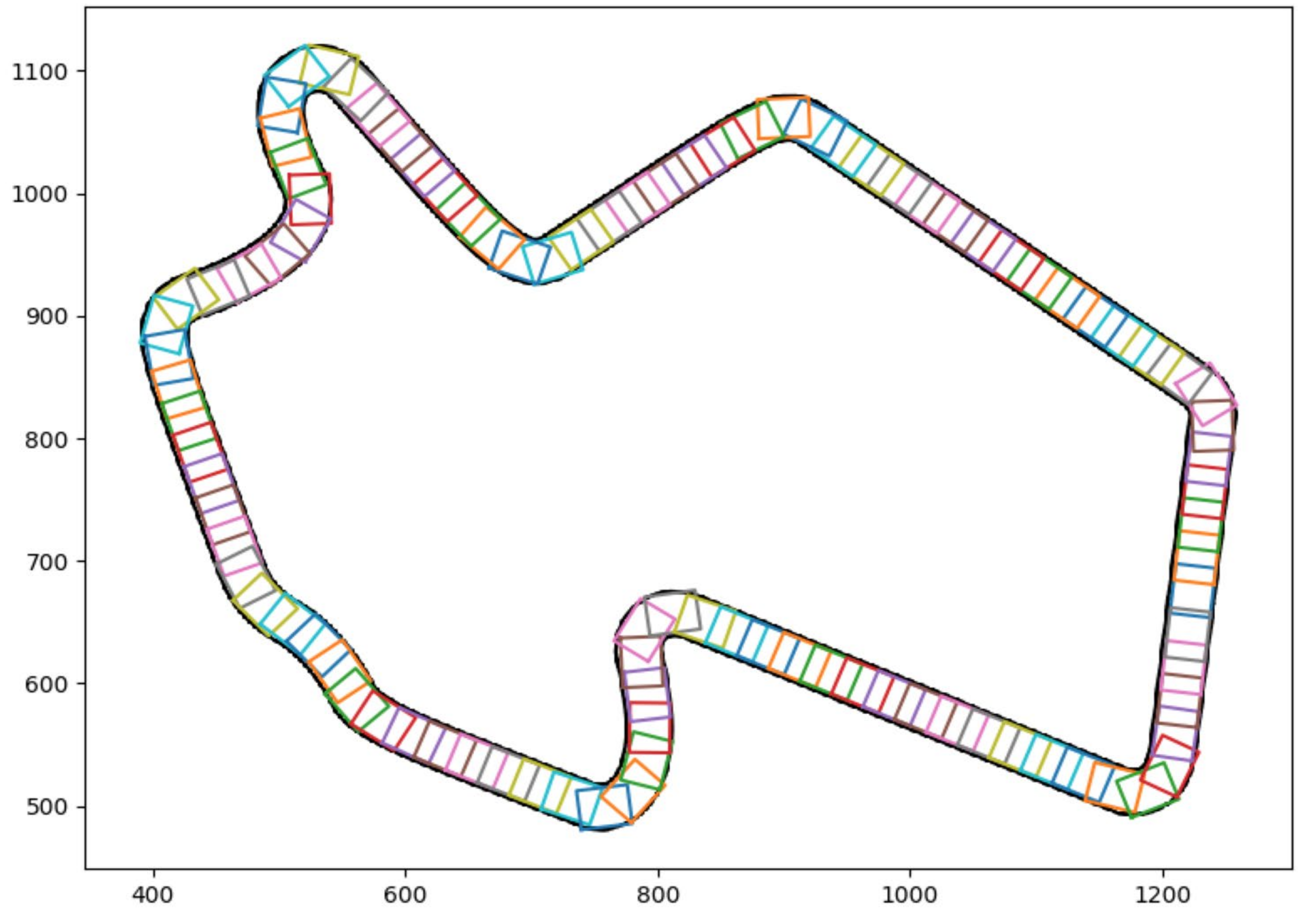
Constraints on actuation

- Mapping of the path:

$$s(\theta(t)) = q(t),\ \ t \in [0,T],\ \ \theta : [0,T] \to [0,1]$$

# Search for solution

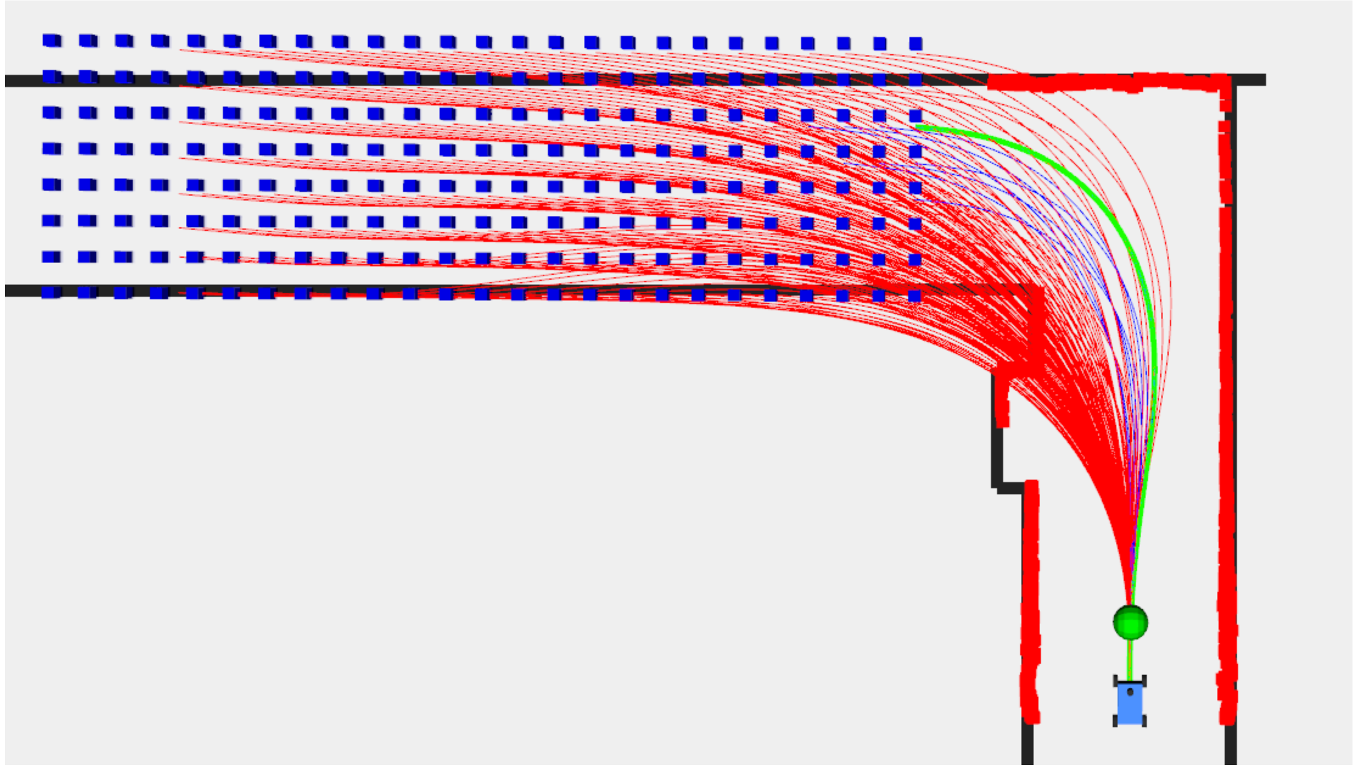
LEHIGH UNIVERSITY




Upenn ESE680 Lecture 17 Slides

- First divide the drivable on a given track into equal sized rectangular boxes.
- Every single box provides a bound on x and y coordinate. A point (x, y) could be sampled inside a box;
- With n boxes, we can form a (2n) vector that defines the control points, then fit a spline
- Different types of splines, some may not go through the control points:
  - Hermite spline, Bezier spline,
  - Catmull-Rom spline,
  - Natural Cubic spline, B-Spline,
  - NURBS, Pythagorean Hodograph

# Genetic Algorithm



Matthew O'Kelly, Hongrui Zheng, Achin Jain, Joseph Auckley, Kim Luong, and Rahul Mangharam, "Tech Report: TUNERCAR: A Superoptimization Toolchain for Autonomous Racing", . September 2019. https://repository.upenn.edu/mlab_papers/122

- GA is a heuristic algorithm: parents, children fitness function, natural selection;
- Successful in solving Traveling salesperson problem and many other search problems
- When applied to path planning, use covariance matrix adaptation.
- Reference:
  - N. Hansen, S. D. Müller and P. Koumoutsakos, "Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES)," in Evolutionary Computation, vol. 11, no. 1, pp. 1-18, March 2003.
  - David Ho, et. al, A visual guide to evolution strategies, http://blog.otoro.net/2017/10/29/visual-evolution-strategies/