2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Research and Educational Directions

Houssam Abbas
EECS, Oregon State University

and the F1/10 team

Oregon State University

978-1-7281-6211-9/20/$31.00 ©2020 IEEE

The role F1/10 can play in preparing students for research

Algorithmic Ethics

How *should* a car behave?

Cyber-Physical Security

Understanding the vulnerabilities of an autonomous vehicle to attacks

Fairness in Multi-Agent Control

Ensuring fair access to the airspace for UAS

F1/10: An Open-Source Autonomous Cyber-Physical Platform

Matthew O'Kelly*, **Houssam Abbas****, **Jack Harkins, Chris Kao, Yash Vardhan Pant & Rahul Mangharam**
Department of Electrical and Systems Engineering, University of Pennsylvania, USA
{mokelly, harkj, chriskao, yashpant, rahulm}@seas.upenn.edu
** School of Electrical Engineering and Computer Science, Oregon State University, USA
houssam.abbas@oregonstate.edu

Varundev Suresh Babu*, **Dipshil Agarwal & Madhur Behl**
Department of Computer Science, University of Virginia, USA
{varundev, dpa4va, madhur.behl}@virginia.edu

Paolo Burgio & Marko Bertogna
Dept. of Physics, Informatics & Mathematics, University of Modena & Reggio Emilia, Italy
{marko.bertogna, paolo.burgio}@unimore.it

* These authors contributed equally

ABSTRACT

In 2005 DARPA labeled the realization of viable autonomous vehicles (AVs) a *grand challenge*; a short time later the idea became a *moonshot* that could change the automotive industry. Today, the question of *safety* stands between *reality* and *solved*. Given the right platform the CPS community is poised to offer unique insights. However, testing the limits of safety and performance on real vehicles is costly and hazardous. The use of such vehicles is also outside the reach of most researchers and students. In this paper, we present F1/10: an open-source, affordable, and high-performance 1/10 scale autonomous vehicle testbed. The F1/10 testbed carries a full suite of sensors, perception, planning, control, and networking software stacks that are similar to full scale solutions. We demonstrate key examples of the research enabled by the F1/10 testbed, and how the platform can be used to augment research and education in autonomous systems, making autonomy more accessible.

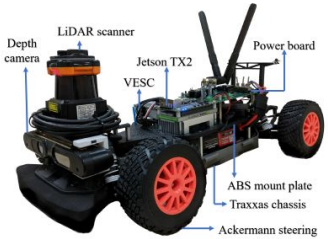


Figure 1: It takes only a couple of hours fully to assemble a F1/10 autonomous racecar, using detailed instructions available at http://f1tenth.org/

1 INTRODUCTION

F1/10: An Open-Source Autonomous Cyber-Physical Platform

Matthew O'Kelly*, Houssam Abbas**, Jack Harkins, Chris Kao, Yash Vardhan Pant & Rahul Mangharam
Department of Electrical and Systems Engineering, University of Pennsylvania, USA
{mokelly, harkj, chriskao, yashpant, rahulm}@seas.upenn.edu
** School of Electrical Engineering and Computer Science, Oregon State University, USA
houssam.abbas@oregonstate.edu

Dept. of

ABSTRACT

In 2005 DARPA labeled the real
hicles (AVs) a *grand challenge*; a
a *moonshot* that could change
the question of *safety* stands bet
right platform the CPS communit
However, testing the limits of sa
cles is costly and hazardous. The
the reach of most researchers and
F1/10: an open-source, affordable
autonomous vehicle testbed. Th
of sensors, perception, planning,
stacks that are similar to full sca
examples of the research enabl
the platform can be used to au
autonomous systems, making au

FormulaZero: Distributionally Robust Online Adaptation via Offline Population Synthesis

Aman Sin
Rahul M

amans@stanfor

Balancing performa
agent environments. I
conservative policies, hi
either make simplifying
adaptation. This work
a realistic, diverse set o

TUNERCAR: A Superoptimization Toolchain for Autonomous Racing

Matthew O'Kelly[1], Hongrui Zheng[1], Joseph Auckley[1], Achin Jain[1], Kim Luong[1,2], and Rahul Mangharam[1]

***Abstract*— The objective of this effort is to develop an optimal autonomous racer with safe and reusable core autonomy components that are agnostic to vehicle planning and control software. TUNERCAR is a toolchain that jointly optimizes racing strategy, planning methods, control algorithms and vehicle parameters for an autonomous racecar. In this paper, we detail the target hardware, software, simulators, and systems infrastructure for this toolchain. Our methodology employs a massively parallel implementation of CMA-ES which enables simulations to proceed 6 times faster than real-world rollouts. Besides a massive speed up, we show our approach can reduce the lap times in autonomous racing, given a fixed computational budget. We demonstrate improvements over naive random search of 2.0 seconds per lap, improvements over expert solutions of 0.81 seconds per lap, and beat a human driver by 6.52 seconds. For all tested tracks, our method provides the lowest lap-time, and relative lap-time improvements between 6 and 12 percent. We further compare the performance of our method against hand tuned solutions submitted by over 30 international**

This paper introduces the notion that component reuse and adaptation is analogous to creating a new kind of compiler that targets computational, physical, and external environmental details of a robot's operational domain. In general, the goal of a compiler is to validate and then transform a source program in one language to another (usually lower-level *e.g.* assembly) which is suitable for the target domain [2]. Modern optimizing compilers [3] also seek to improve the performance of the transformed program. To concretize the analogy, we define the source program as a parameterized description of the vehicle dynamics, tracking controllers, and local planning method which we wish to transform to perform safely and efficiently in the operational environment represented by a map, physical laws, and sensing capabilities.

We propose a solution to the cyber-physical compilation problem utilizing the concept of *superoptimization* (*c.f.* [4],

F1/10: An Open-Source Autonomous Cyber-Physical Platform

Matthew O'Kelly*, Houssam Abbas**, Jack Harkins, Chris Kao, Yash Vardhan Pant & Rahul Mangharam
Department of Electrical and Systems Engineering, University of Pennsylvania, USA
{mokelly, harkj, chriskao, yashpant, rahulm}@seas.upenn.edu

ABSTRAC
In 2005 DARP
hicles (AVs) a
a *moonshot* th
the question o
right platform
However, testi
cles is costly a
the reach of mo
F1/10: an open
autonomous v
of sensors, per
stacks that are
examples of th
the platform c
autonomous s

Teaching Autonomous Systems at $1/10^{th}$-scale: Design of the F1/10 Racecar, Simulators and Curriculum

Abhijeet Agnihotri*†
agnihota@oregonstate.edu
Oregon State University
Corvallis, Oregon

Matthew O'Kelly*
mokelly@seas.upenn.edu
University of Pennsylvania
Philadelphia, Pennsylvania

Rahul Mangharam
rahulm@seas.upenn.edu
University of Pennsylvania
Philadelphia, Pennsylvania

Houssam Abbas
houssam.abbas@oregonstate.edu
Oregon State University
Corvallis, Oregon

ABSTRACT

Teaching autonomous systems is challenging because it is a rapidly advancing cross-disciplinary field that requires theory to be continually validated on physical platforms. For an autonomous vehicle

adaptation. This work
a realistic, diverse set o

Autonomous Racing

uong[1,2], and Rahul Mangharam[1]

es the notion that component reuse and
s to creating a new kind of compiler that
, physical, and external environmental
rational domain. In general, the goal of
te and then transform a source program
ther (usually lower-level *e.g.* assembly)
e target domain [2]. Modern optimizing compilers [3] also seek to improve the performance of the transformed program. To concretize the analogy, we define the source program as a parameterized description of the vehicle dynamics, tracking controllers, and local planning method which we wish to transform to perform safely and efficiently in the operational environment represented by a map, physical laws, and sensing capabilities.

We propose a solution to the cyber-physical compilation problem utilizing the concept of *superoptimization* (*c.f.* [4],

implementation of CMA-ES which enables simulations to proceed 6 times faster than real-world rollouts. Besides a massive speed up, we show our approach can reduce the lap times in autonomous racing, given a fixed computational budget. We demonstrate improvements over naive random search of 2.0 seconds per lap, improvements over expert solutions of 0.81 seconds per lap, and beat a human driver by 6.52 seconds. For all tested tracks, our method provides the lowest lap-time, and relative lap-time improvements between 6 and 12 percent. We further compare the performance of our method against hand tuned solutions submitted by over 30 international

Sample research topics

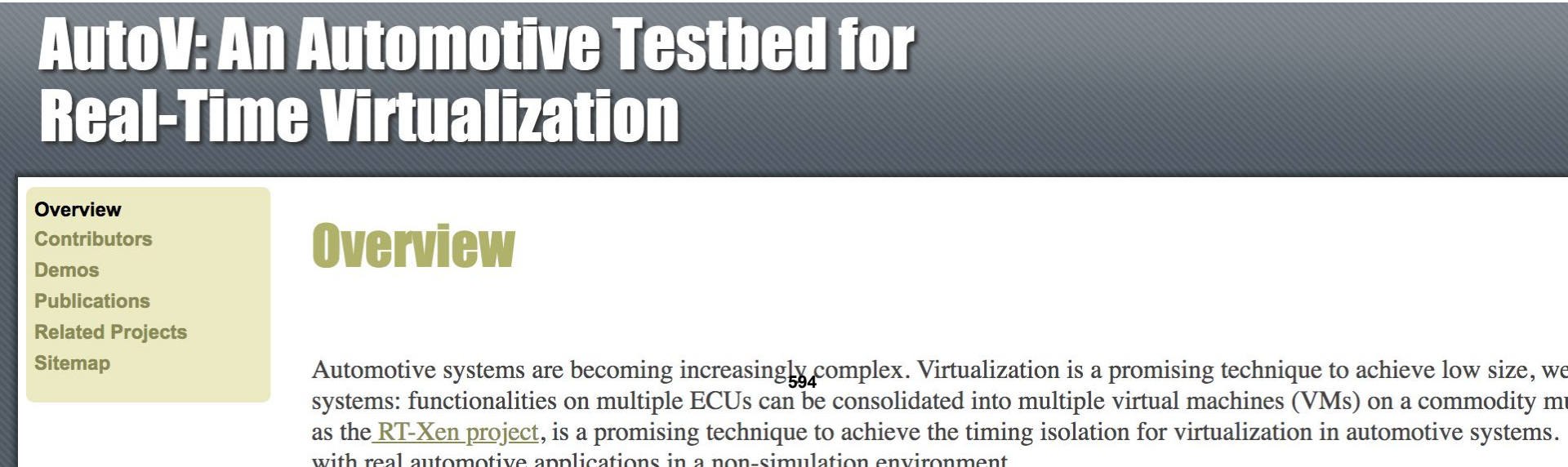
Real-time systems research

- Real workloads
- Representative hardware

Sample research topics

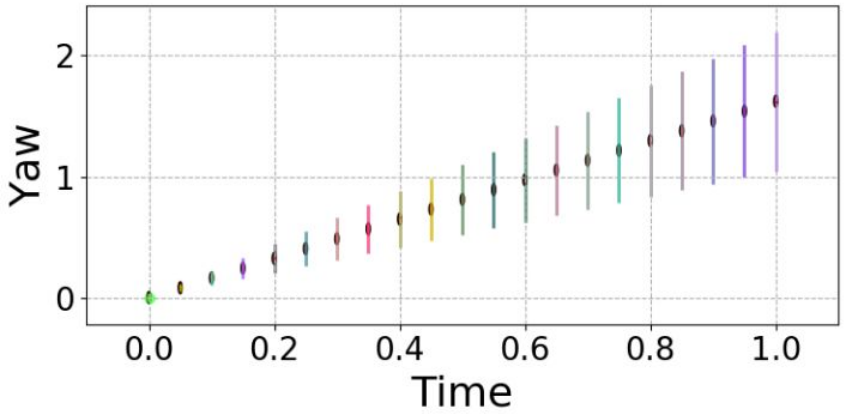
Real-time systems research

- Real workloads
- Representative hardware



Sample research topics

Real-time reachability



Sample research topics

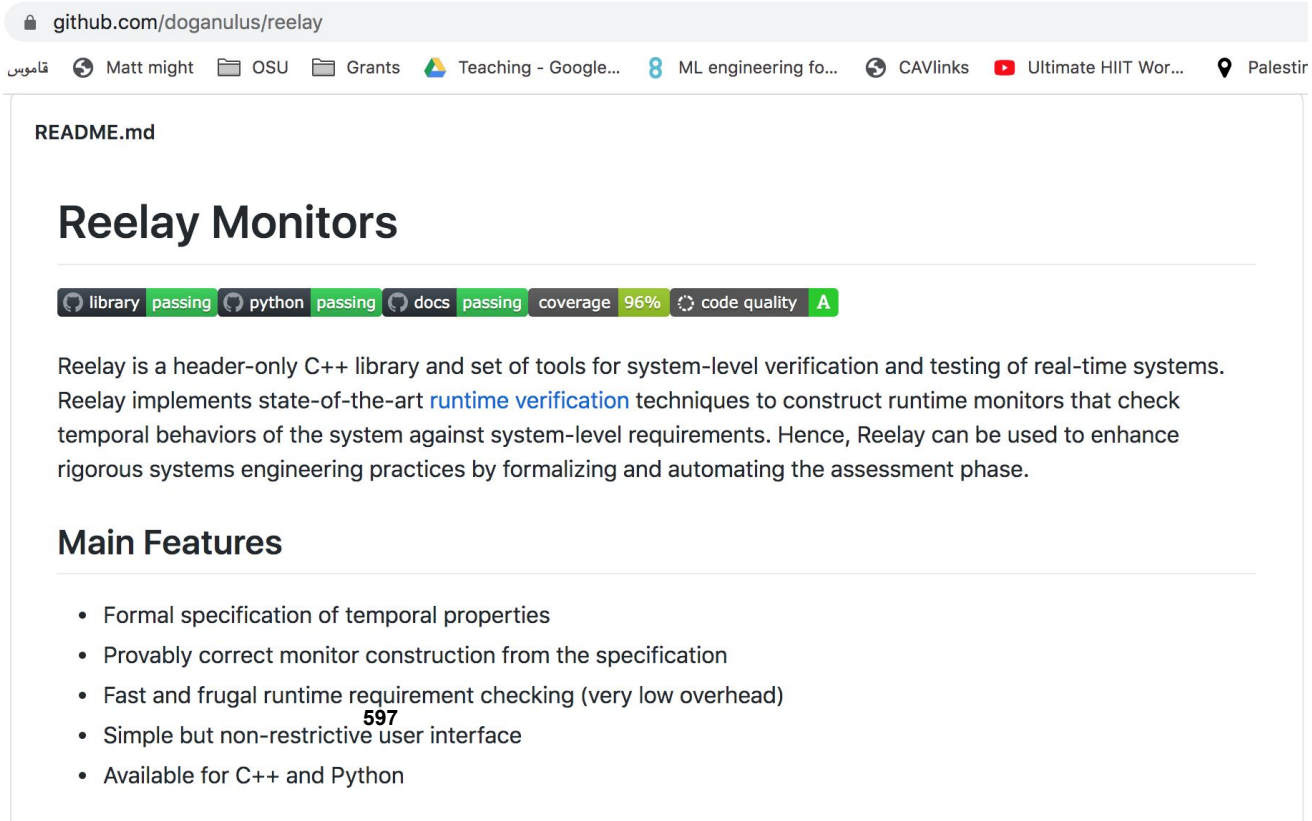
Real-time reachability

Runtime monitoring

Sample research topics

Real-time reachability

Runtime monitoring



Sample research topics

Real-time reachability

Runtime monitoring

Adversarial traffic

Scalable End-to-End Autonomous Vehicle Testing via Rare-event Simulation

Matthew O'Kelly*
University of Pennsylvania
mokelly@seas.upenn.edu

Aman Sinha*
Stanford University
amans@stanford.edu

Hongseok Namkoong*
Stanford University
hnamk@stanford.edu

John Duchi
Stanford University
jduchi@stanford.edu

Russ Tedrake
Massachusetts Institute of Technology
russt@mit.edu

F1/10 Onboard Reachability

Nathan Jewell-Carlton, CS undergrad
Autonomous Systems Lab OSU

Problem

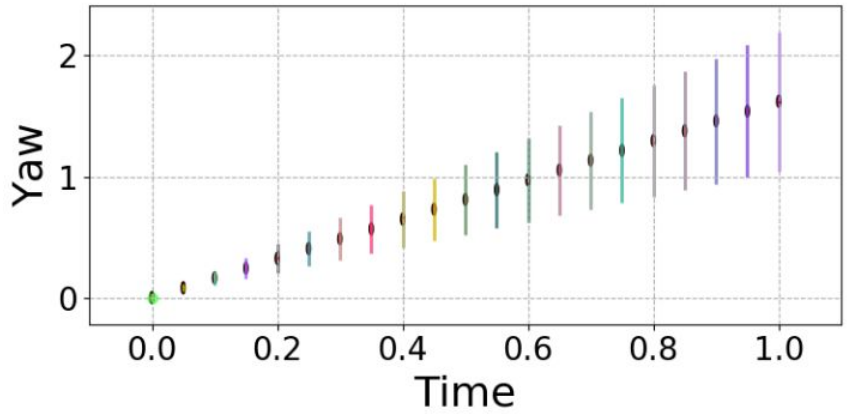
- How do we ensure safety?
- Where is the agent going to be?
- How is it going to get there?
- Will it hit an obstacle?
- Find out and react before it happens (speed!)

Reachability Process

- Find physical properties of the agent
- Develop dynamical model

Reachability Process

- Find physical properties of the agent
- Develop dynamical model
- **Minimally over-approximate possible future system states**



602

Reachability Process

- Find physical properties of the agent
- Develop dynamical model
- **Minimally over-approximate possible future system states**
- Ensure there are no failure modes
- Adjust control systems

Reachability Process

- Find physical properties of the agent
- Develop dynamical model
- **Minimally over-approximate possible future system states**
- Ensure there are no failure modes
- Adjust control systems

this is the essence of reachability

Successive Linearization

Nonlinear reachability - slow, accurate

Linearize then run Linear reachability - fast, cheap, less accurate

Successive Linearization

- Multiple linear eq. Combined
- Fast computation algorithms
- Controllable accuracy
- Linearization overhead

Non-Linear Dynamical Model

Variables of state are decided by the systems' dynamics.
x : the vehicle's x coordinate in meters
y : the vehicle's y coordinate in meters
ψ : the vehicle's yaw in Radian

Input variables are decided by the control system.
v : the vehicles velocity
d : the vehicles change if ψ

$$\dot{x} = v\cos\psi + d \tag{1}$$

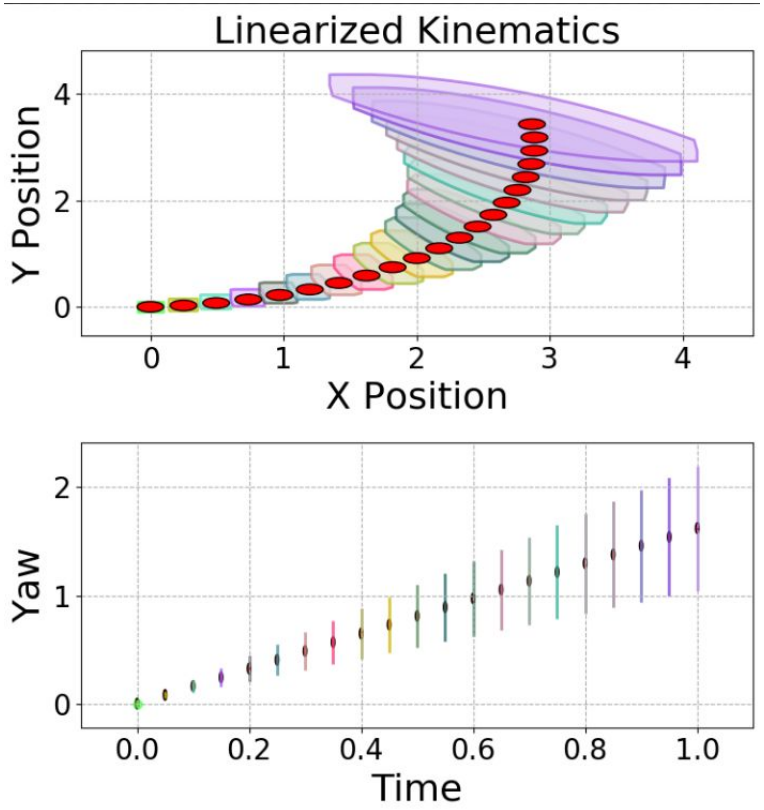
$$\dot{y} = v\sin\psi + d \tag{2}$$

$$\dot{\psi} = v\frac{1}{L}\sin(d) \tag{3}$$

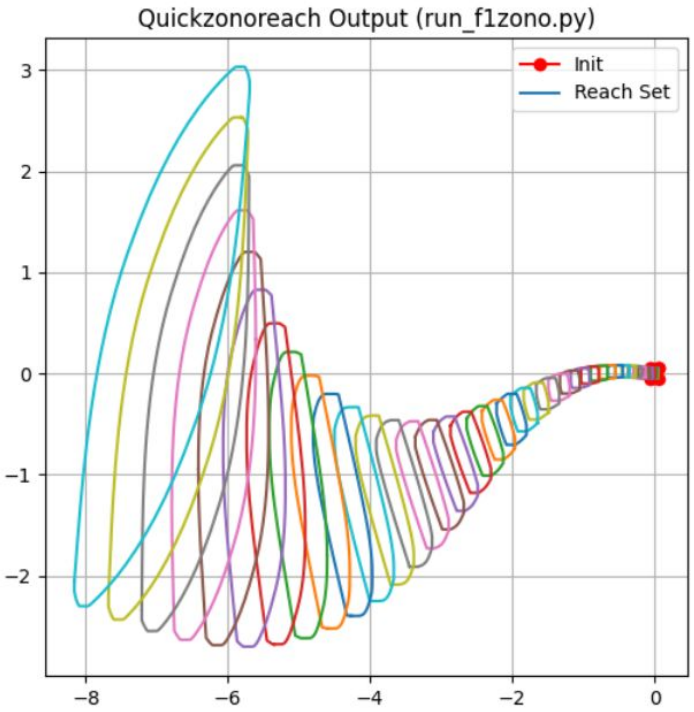
Output

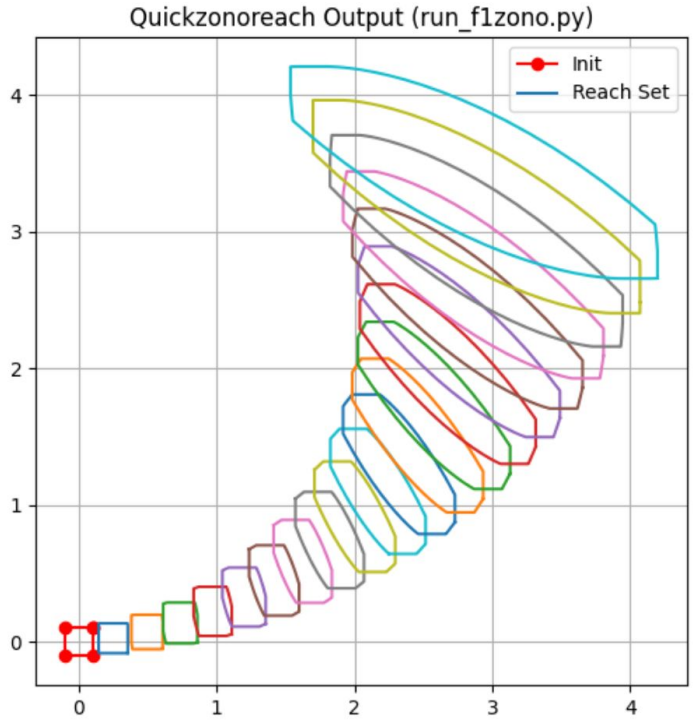
Uses HYLAA,

a reachability tool by Stanley Bak



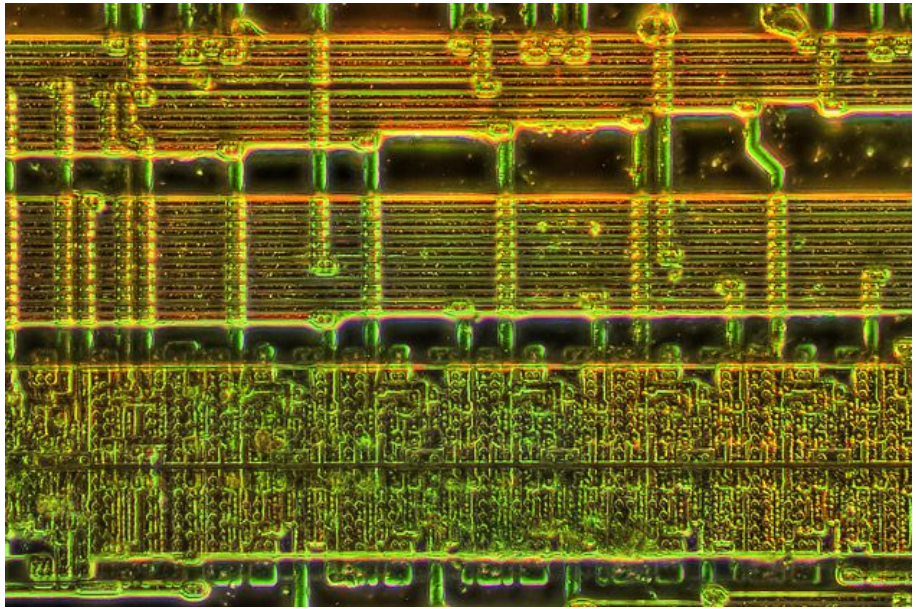
Output


Quickzonoreach Output (run_f1zono.py)
Init
Reach Set
3
2
1
0
−1
−2
−8
−6
−4
−2
0

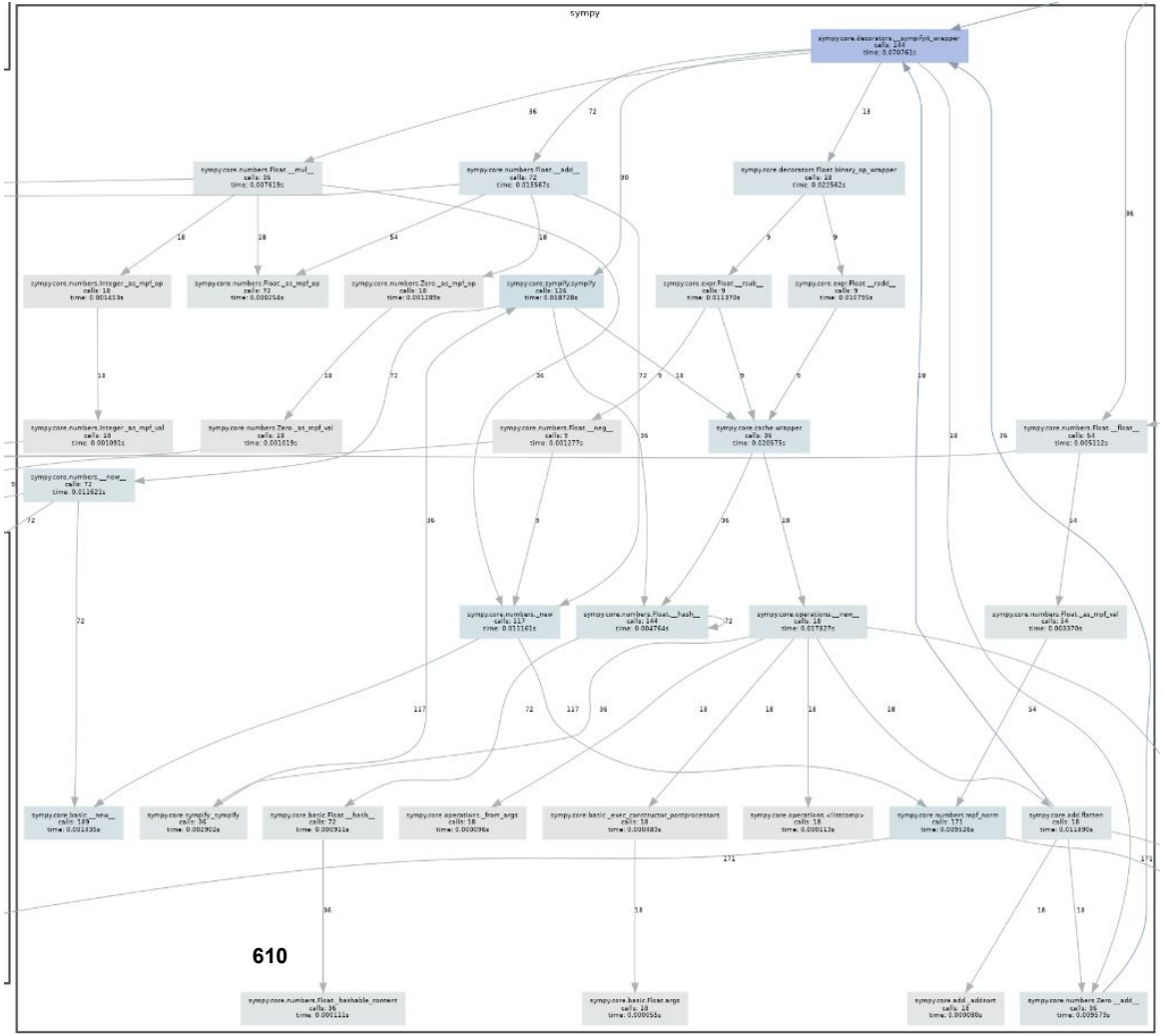

Quickzonoreach Output (run_f1zono.py)
Init
Reach Set
4
3
2
1
0
0
1
2
3
4

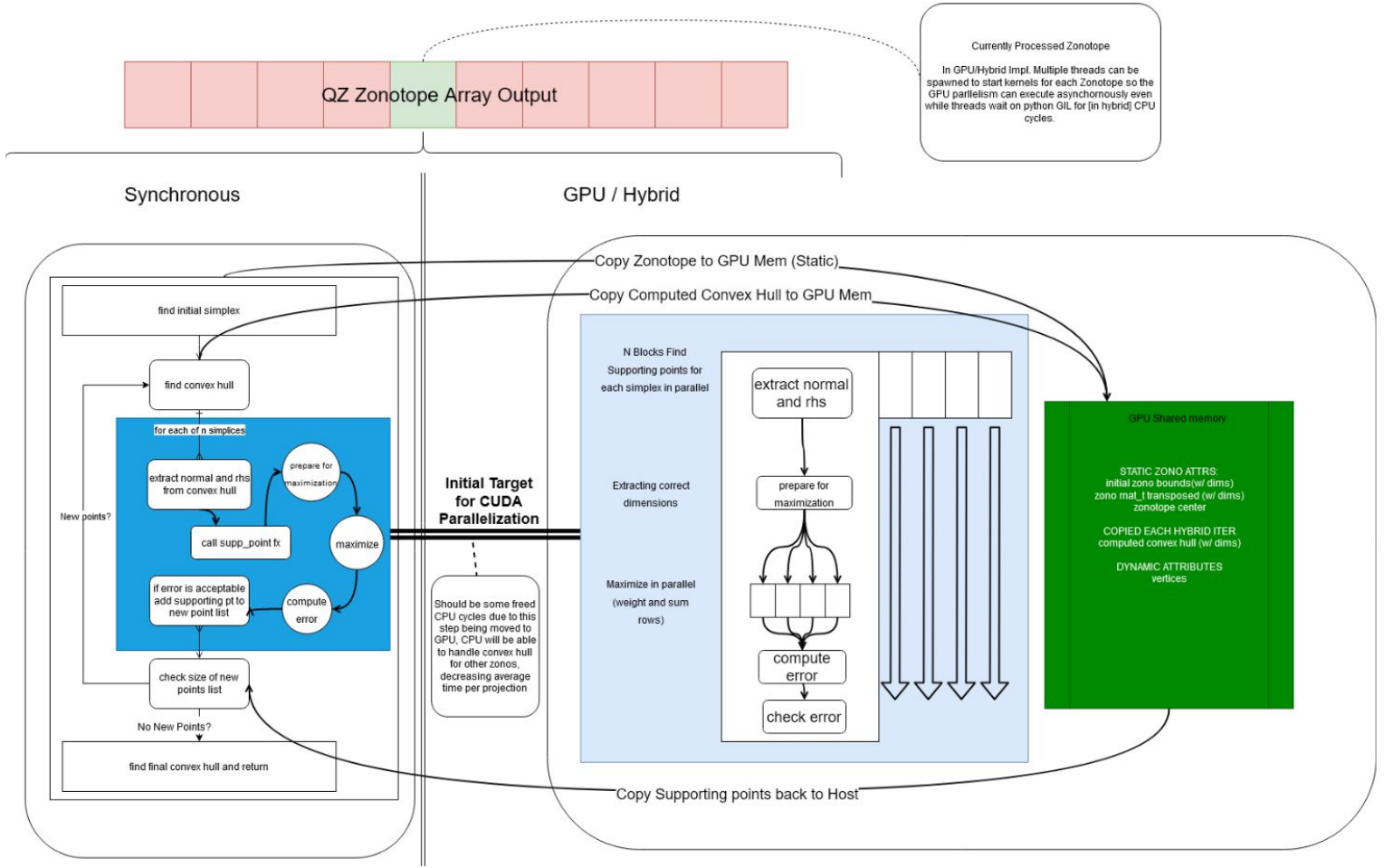
GPU Parallelism

- How can performance be increased?
- Use more of our hardware!
- Free cpu cycles
- Improve latency
- Significant refactoring required



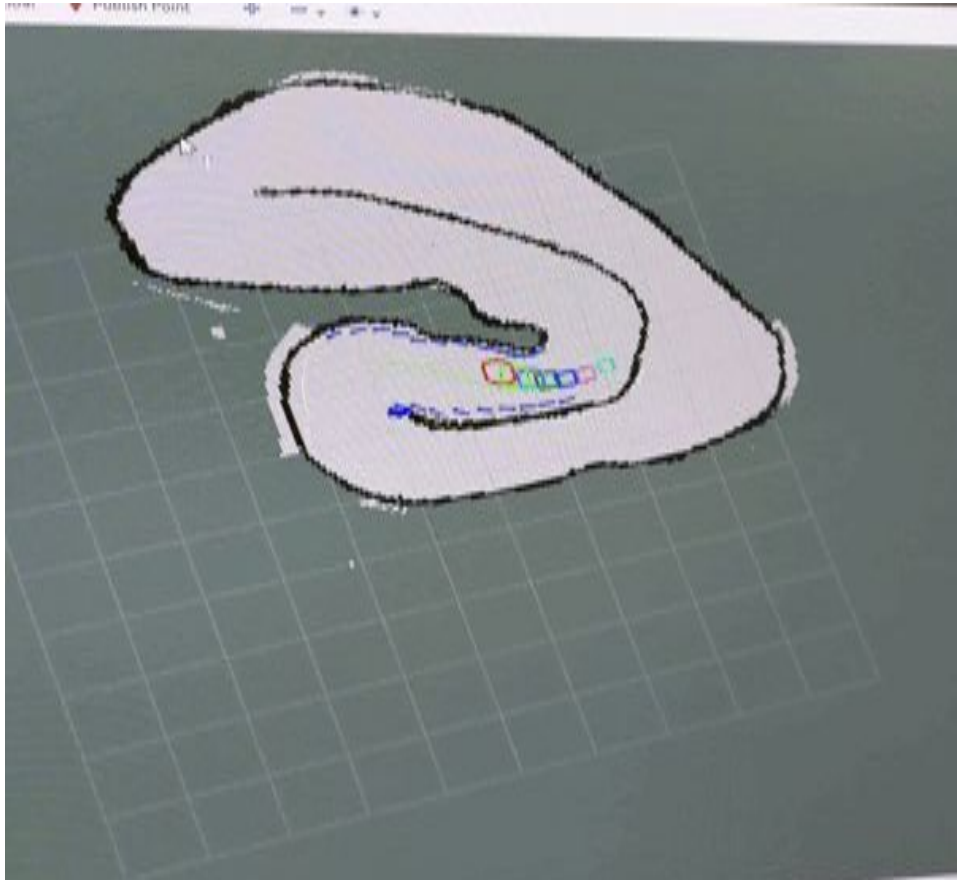
Digging into function call timings (pycallgraph output)





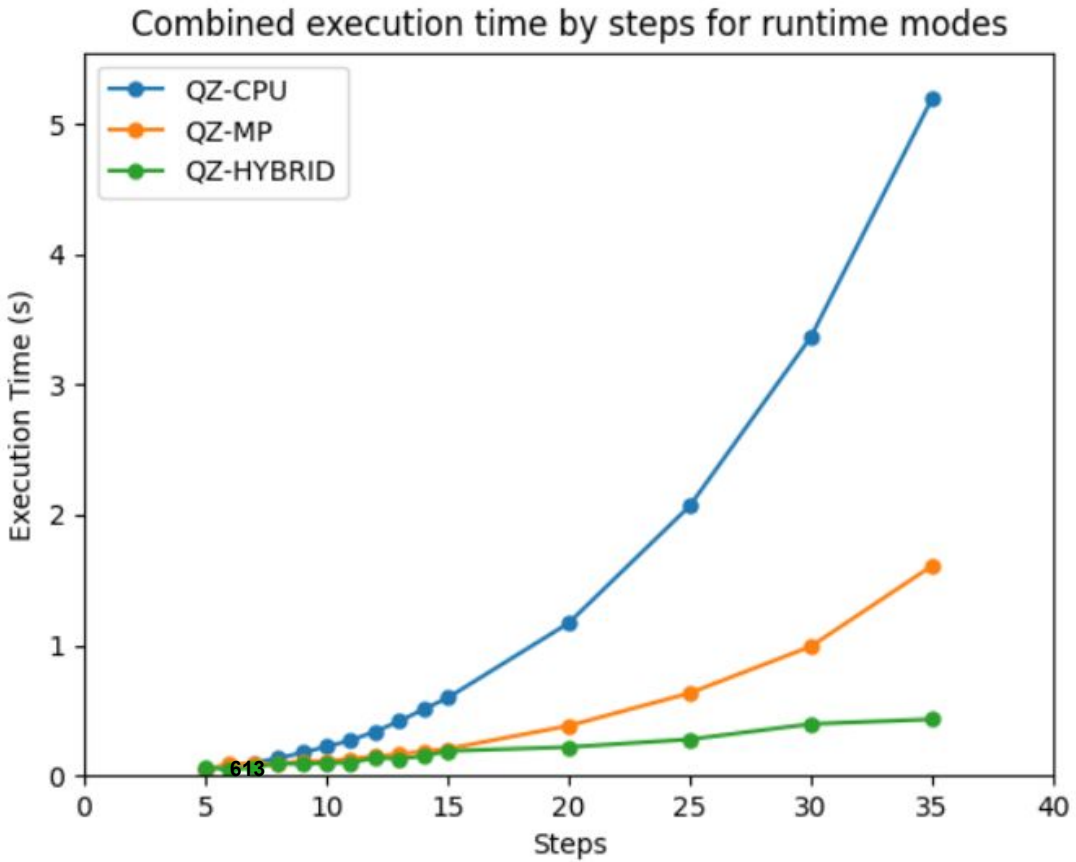
Flow diagram for refactoring CPU based reachability to run on NVIDIA GPU

Demo

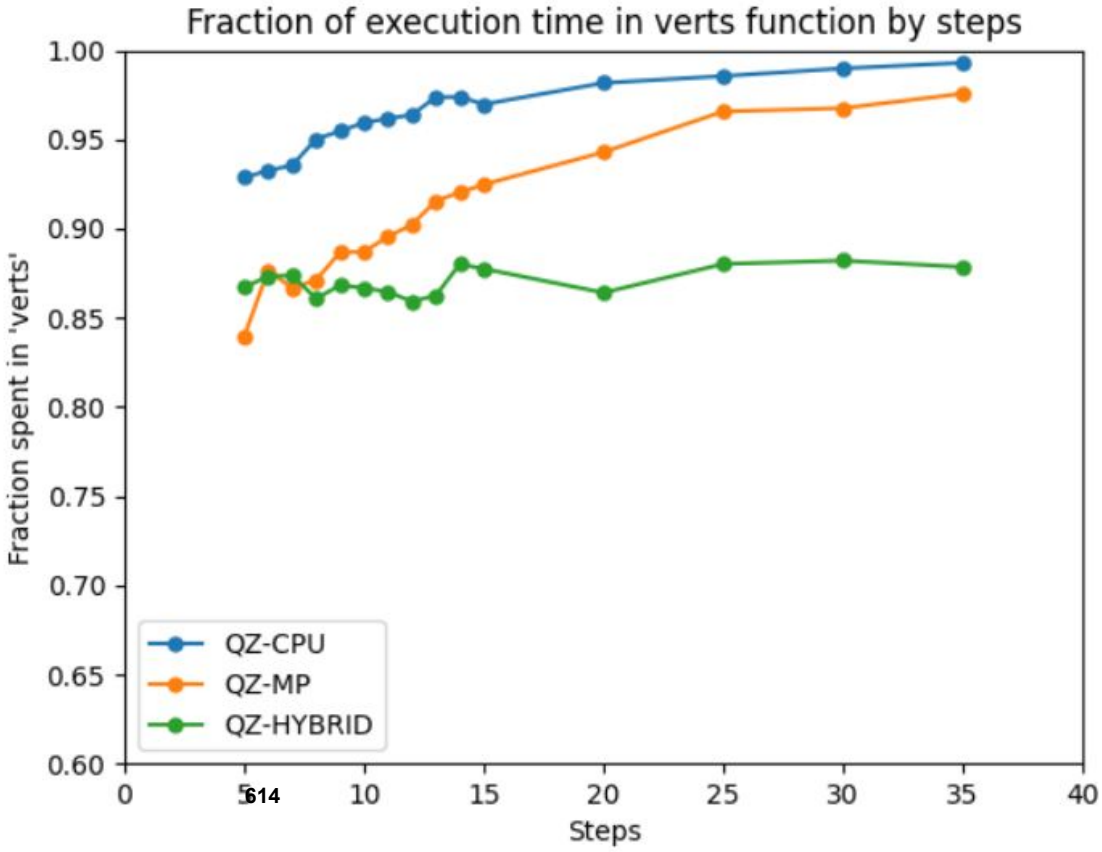


RVIZ of car running around demo track - live onboard reachability

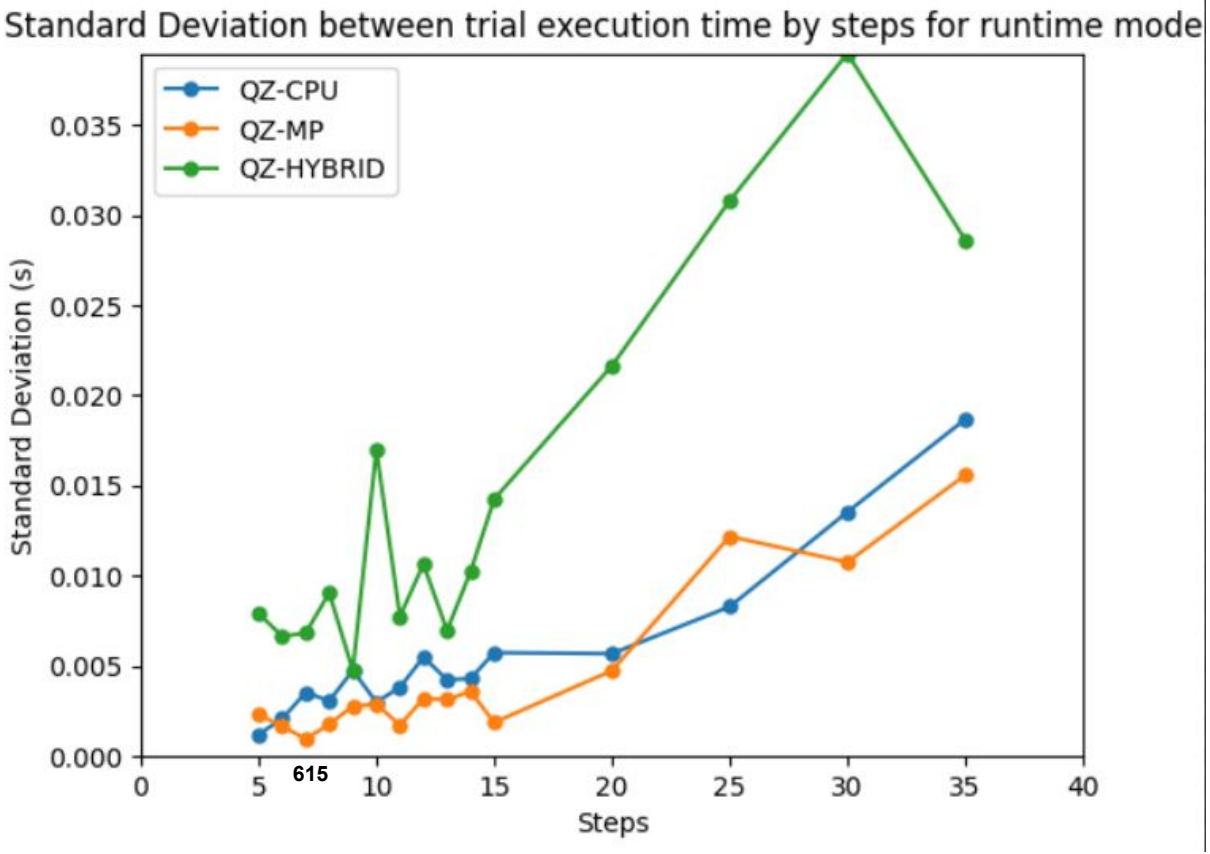
Performance



Performance



Performance



Nonlinear MPC and MPCC for F1/10 cars

Niraj Basnet, M.Sc. student
Autonomous Systems Lab, OSU

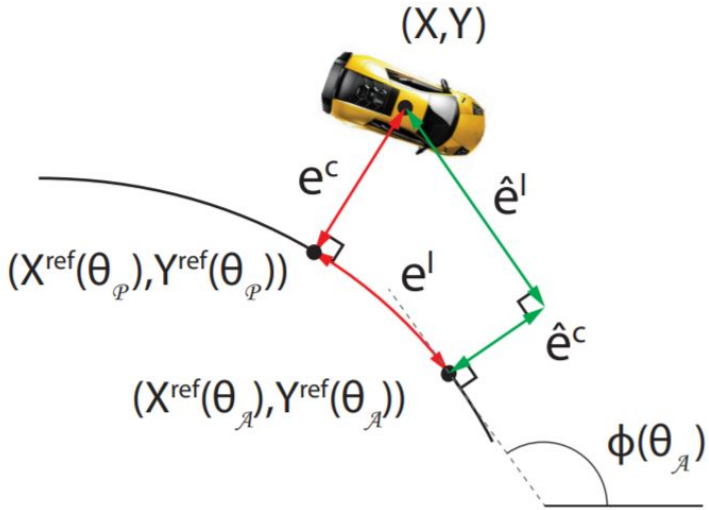
Nonlinear MPC

- Uses nonlinear system model
- Better disturbance rejection capability
- Better step response
- Better tracking performance
- Slower than Linear MPC
- Modern solvers are improving runtimes

$$
\begin{aligned}
\min_{\boldsymbol{U},\boldsymbol{X}} \int_{t=0}^{T} & \left\|\boldsymbol{x}(t)-\boldsymbol{x}_{ref}(t)\right\|_{\boldsymbol{Q}_x}^2+\left\|\boldsymbol{u}(t)-\boldsymbol{u}_{ref}(t)\right\|_{\boldsymbol{R}_u}^2 dt \\
& +\left\|\boldsymbol{x}(T)-\boldsymbol{x}_{ref}(T)\right) \|_{\boldsymbol{P}}^2 \\
\text{subject to} \quad & \dot{\boldsymbol{x}}=\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}); \\
& \boldsymbol{u}(t) \in \mathbb{U} \\
& \boldsymbol{x}(0)=\boldsymbol{x}\left(t_0\right).
\end{aligned}
$$

Model Predictive Contouring Control (MPCC)

- Spatial reference path sufficient for tracking
- No reference control inputs needed
- Centerline of the racetrack acts as reference contour, parameterized by its arc length θ
- Uses lag and contouring error to generate trajectory with trade-off between tracking precision and speed
- Path planning and tracking combined into single nonlinear optimization problem
- Able to generate locally optimal racing lines



Contouring error(e^c) and Lag error(e^l) with linear approximations $\hat{e}^c$ and $\hat{e}^l$

Nonlinear MPCC

Problem Formulation

$$\min \quad \sum_{k=1}^{N} \begin{bmatrix} \hat{e}_k^c \\ \hat{e}_k^l \end{bmatrix}^T \begin{bmatrix} q_c & 0 \\ 0 & q_l \end{bmatrix} \begin{bmatrix} \hat{e}_k^c \\ \hat{e}_k^l \end{bmatrix} - q_v v_{\theta,k} + \Delta u_k^T R_\Delta \Delta u_k$$

$$\text{s.t.} \quad x_0 = x(0)$$

$$x_{k+1} = f(x_k, u_k)$$ → System model

$$\hat{e}^c(x_k) = \sin\left(\Phi^{\text{ref}}(\theta_k)\right)\left(X_k - X^{\text{ref}}(\theta_k)\right) - \cos\left(\Phi^{\text{ref}}(\theta_k)\right)\left(Y_k - Y^{\text{ref}}(\theta_k)\right)$$ → Contouring error

$$\hat{e}^l(x_k) = -\cos\left(\Phi^{\text{ref}}(\theta_k)\right)\left(X_k - X^{\text{ref}}(\theta_k)\right) - \sin\left(\Phi^{\text{ref}}(\theta_k)\right)\left(Y_k - Y^{\text{ref}}(\theta_k)\right)$$ → Lag error

$$\Delta u_k = u_k - u_{k-1}$$ → Rate of input change

$$x_k \in \mathcal{X}_{\text{Track}}$$ → Path constraints

$$\underline{x} \le x_k \le \bar{x}$$ → State constraints

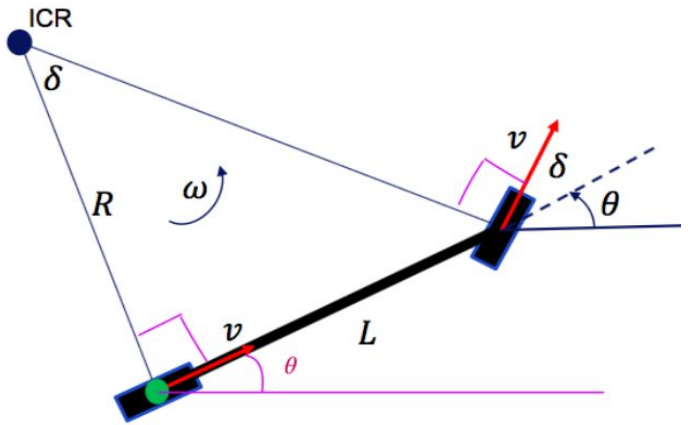
$$\underline{u} \le u_k \le \bar{u}$$

$$\underline{\Delta u} \le \Delta u_k \le \overline{\Delta u}$$ → Actuator constraints

Kinematic vehicle model

- Simplified motion model with easiest parameter identification
- Imposes non-holonomic constraints
- No regards for tyre forces
- Suitable for slow speeds only
- Prone to understeering and oversteering
- Computationally cheap to compute

$$x = [X, Y, \psi, v, \theta]$$
$$u = [a, \delta, v_\theta]$$



Rear axle bicycle kinematic model

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

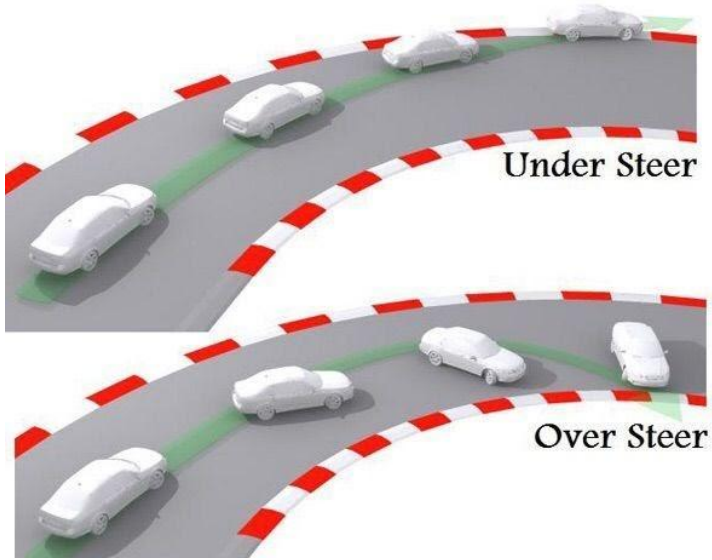
$$\dot{y} = v\sin(\theta)$$

$$\dot{\theta} = \frac{v}{L}\tan(\delta)$$

$$\dot{v} = a$$

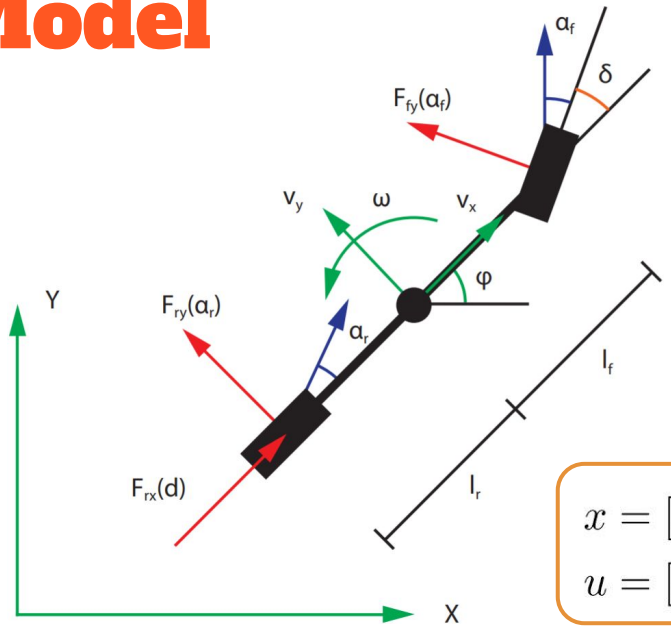
Dynamic Vehicle Model

- Considers both lateral and longitudinal dynamics
- Uses linear or nonlinear tyre forces to account for slip at high speeds
- Involves complicated system modeling by an expert
- Handles high speed behaviors
- Defines understeer and oversteer
- Computationally demanding



Dynamic Vehicle Model

$$\dot{X} = v_x \cos(\varphi) - v_y \sin(\varphi)$$
$$\dot{Y} = v_x \sin(\varphi) + v_y \cos(\varphi)$$
$$\dot{\varphi} = \omega$$
$$\dot{v}_x = \frac{1}{m}(F_{r,x} - F_{f,y} \sin\delta + m v_y \omega)$$
$$\dot{v}_y = \frac{1}{m}(F_{r,y} - F_{f,y} \cos\delta - m v_x \omega)$$
$$\dot{\omega} = \frac{1}{I_z}(F_{f,y} l_f \cos\delta - F_{r,y} l_r)$$
$$\dot{\theta} = v_\theta$$



$$x = [X, Y, \varphi, v_x, v_y, \omega, \theta]$$
$$u = [d, \delta, v_\theta]$$

Tyre Model (Pacejka)

$$F_{f,y} = D_f \sin(C_f \arctan(B_f \alpha_f)) \quad \text{where} \quad \alpha_f = -\arctan\left(\frac{\dot{\varphi} l_f + v_y}{v_x}\right) + \delta$$
$$F_{r,y} = D_r \sin(C_r \arctan(B_r \alpha_r)) \quad \text{where} \quad \alpha_r = \arctan\left(\frac{\dot{\varphi} l_r - v_y}{v_x}\right)$$
$$F_{r,x} = (C_{m1} - C_{m2} v_x) d - C_r - C_d v_x^2$$

Hybrid Vehicle Model

- Dynamic model(DM) is ill-defined for slow speeds due to slip angles
- Kinematic model(KM) fails at high speeds
- Combine Dynamic and Kinematic model to get the best of both
- Uses a velocity threshold to switch between two models
- Velocity threshold used : v_{th} = 0.8m/s
- Actuator space is kept same while state space solution is remapped during the transition

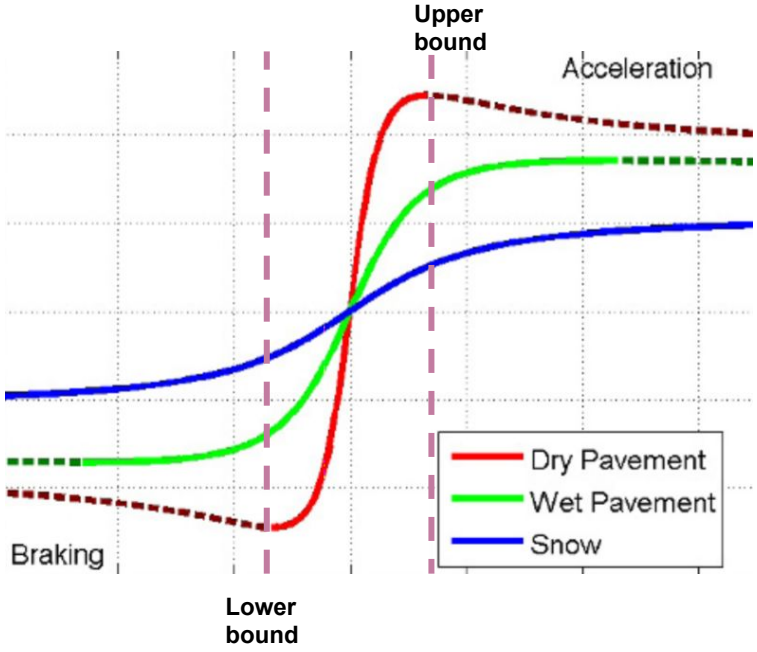
$$f_{hyb}(x,u) = \lambda f_{kin}(x,u) + (1-\lambda) f_{dyn}(x,u)$$

$$\lambda = \begin{cases} 1 & \text{if } v_x \leq v_{th} \\ 0 & \text{otherwise} \end{cases}$$

Constraints

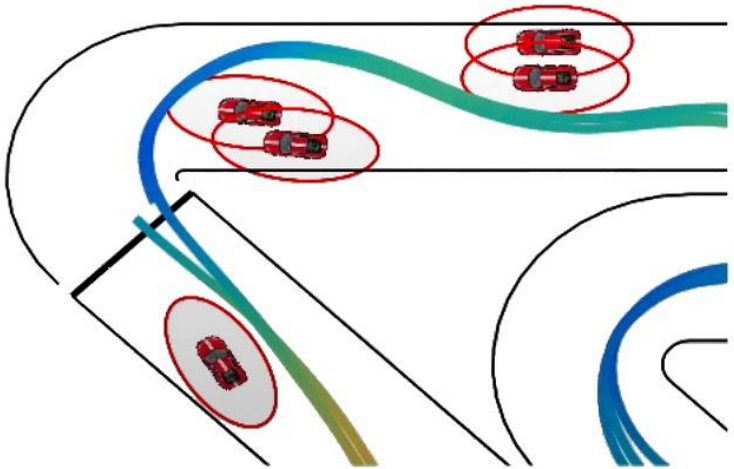
Tyre constraints

Slip angle constraints on front wheel and rear wheel



Obstacle constraints

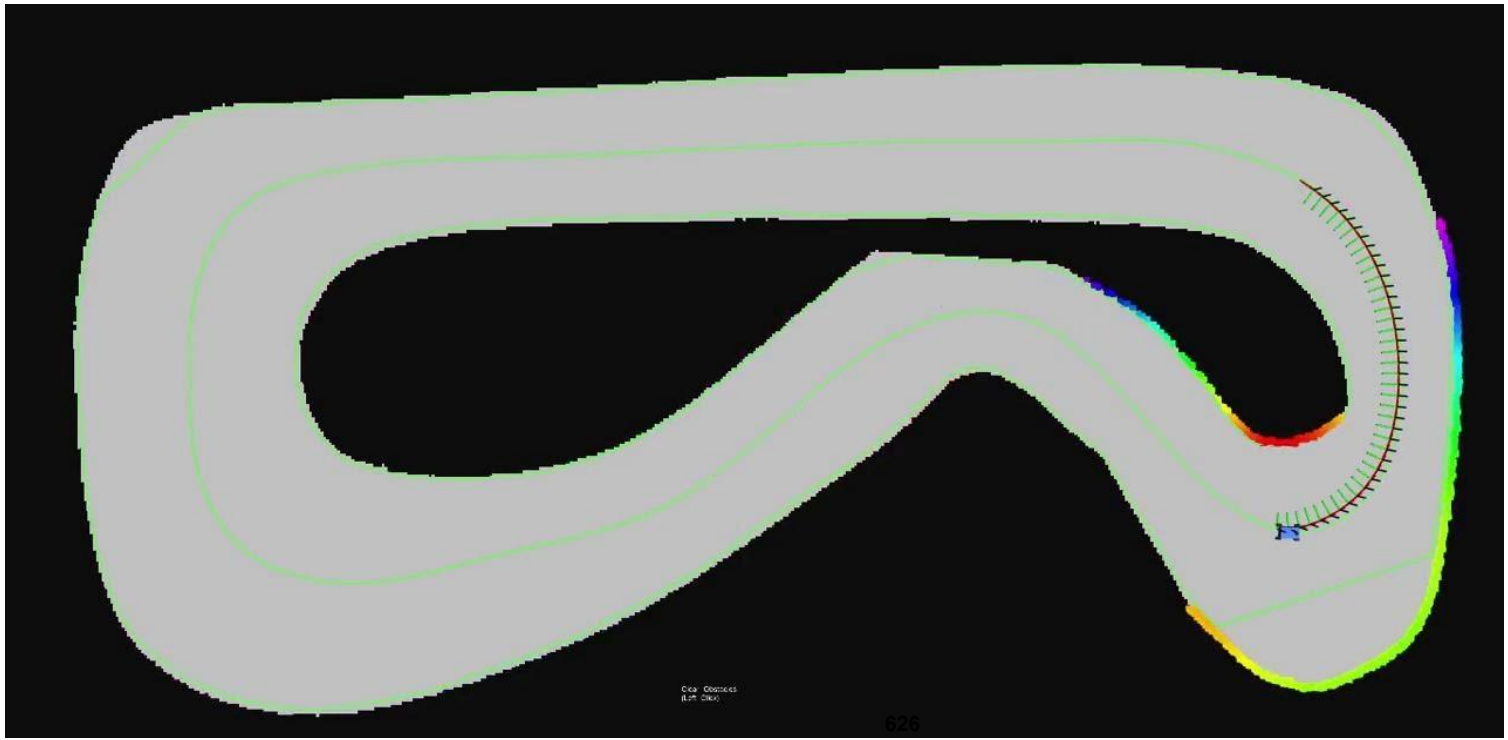
Ellipsoidal approximation of other obstacles(cars)



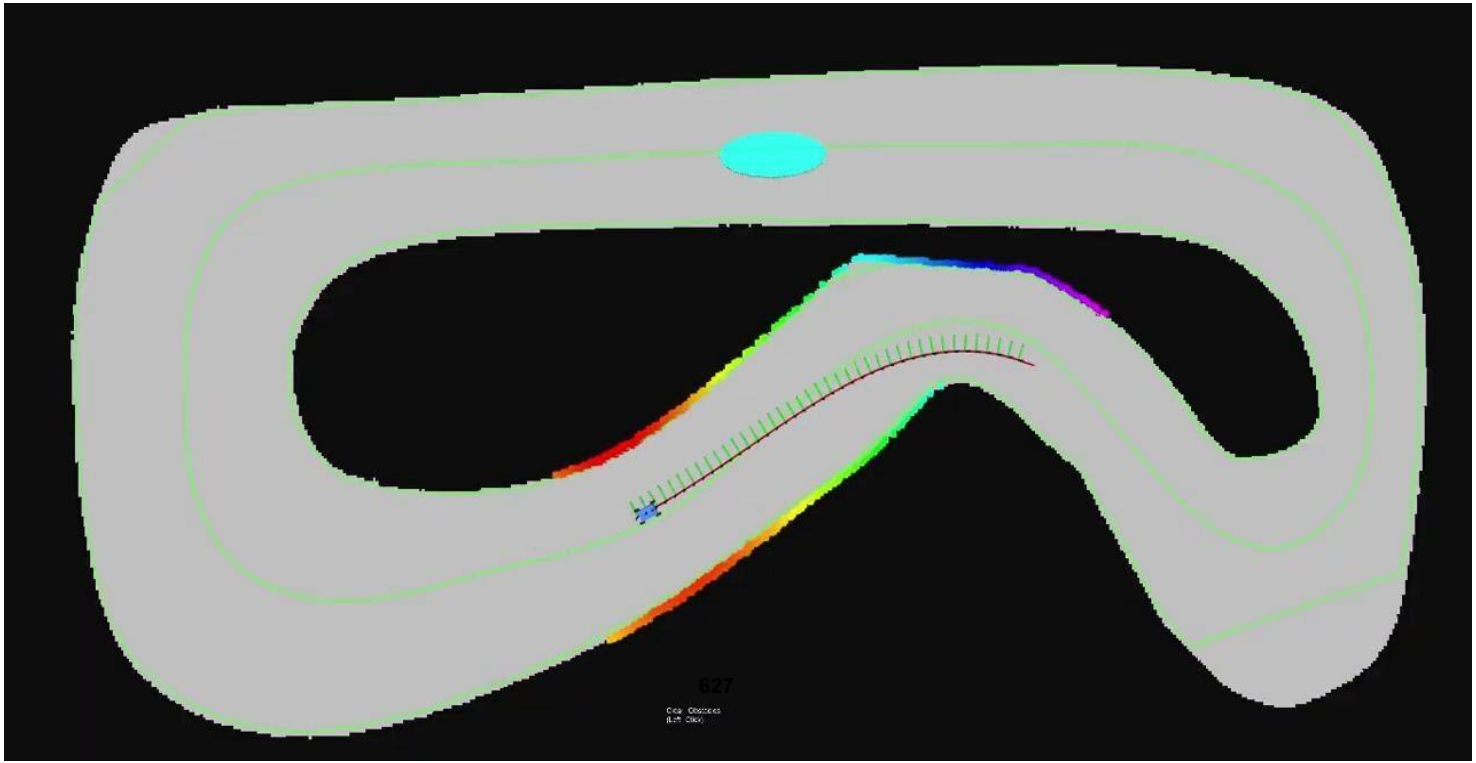
Solvers for Nonlinear MPCC

- Interior point based solvers like Ipopt , Hpipm, Acados, ForcesPro(Commercial)
- Can be structured into sequence of locally convex QPs to use SQP based solvers
- Casadi framework for Algorithmic differentiation
- Implementation details of Nonlinear MPCC(dynamical model)
    - **Sampling time of 50ms**
    - **Horizon N=40**
    - **4th order Runge Kutta discretization**
    - **Solver: ForcesPro - Embotech**
    - **Controller PC with i7@3.2Ghz**
    - **Mean Solver time of 10ms (Real time)**

MPCC Kinematic (Precision Tracking mode)



MPCC Kinematic (Contouring mode with an obstacle)



MPCC Dynamic(High speed driving)

References

- D. Lam, C. Manzie, and M. Good, “Model predictive contouring control,” in Conference on Decision and Control (CDC), pp. 6137– 6142, 2010
- Liniger, A., Domahidi, A. and Morari, M.Optimization-based autonomous racing of 1: 43 scale RC cars.Optimal Control Applications and Methods, 2015, 36(5), pp.628-647.
- J. Kong, M. Pfeiffer, G. Schildbach and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," 2015 IEEE Intelligent Vehicles Symposium (IV), Seoul, 2015, pp. 1094-1099, doi: 10.1109/IVS.2015.7225830.
- Kirchner, William & Southward, Steve. (2011). An Anthropomimetic Approach to High Performance Traction Control. Paladyn, Journal of Behavioral Robotics. 2. 25-35. 10.2478/s13230-011-0013-9.

The role F1/10 can play in preparing students for research