WashU

Real-Time Obstacle Detection and Avoidance using Low-Level CBF/CLF and CBF/PID Controllers

Nicolas Hernandez

Outline

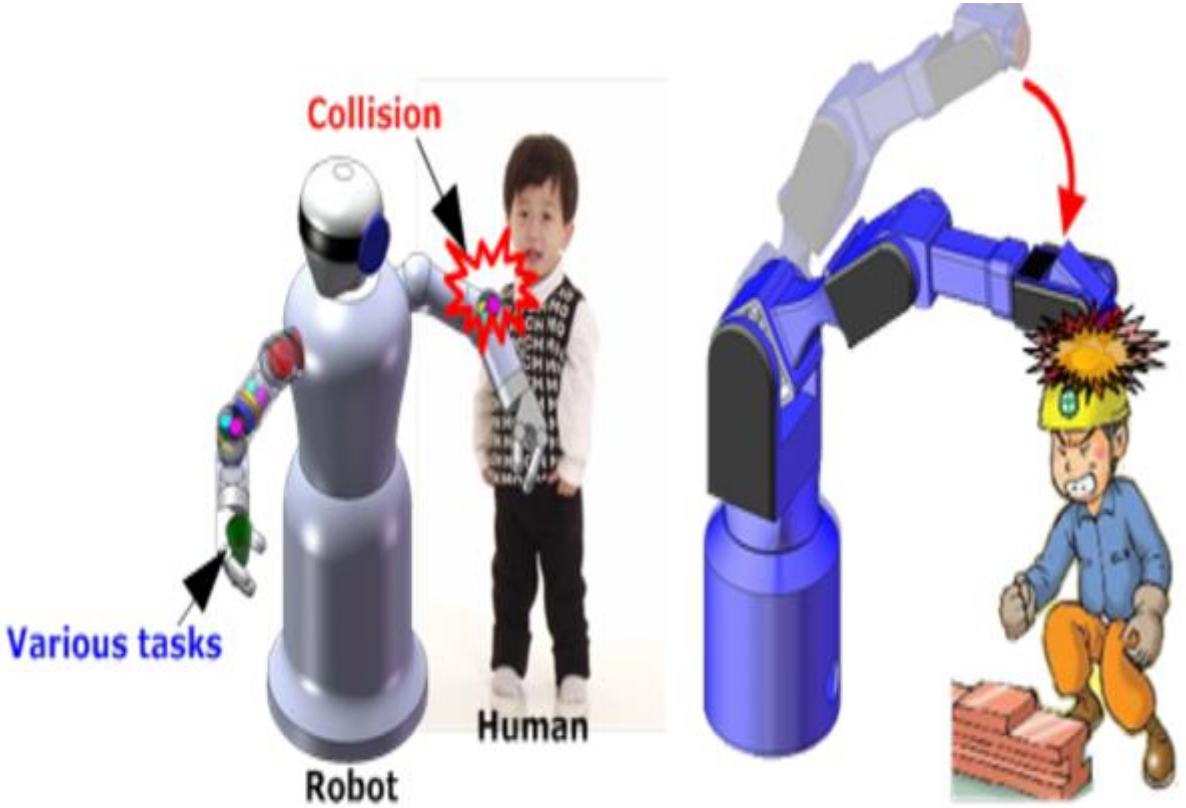
- Safety and Robotics
- Problem Statement
- Perception Pipeline
- CBF-Based Control
- Results
- Next Steps

Safety and Robotics

Motivation

Increasing demand for autonomous systems in dynamic environments highlights the need for real-time safe collision avoidance techniques

- The U.S. autonomous vehicle market has an expected annual growth of 27.5% [1]
- Self-driving vehicles are more than twice as likely as traditional vehicles to get into auto accidents [2]”
- US Global humanoid robot has an expected annual growth of 15.57% [3]
- A robot arm at Tesla’s factory reportedly attacked and injured an engineer[4]

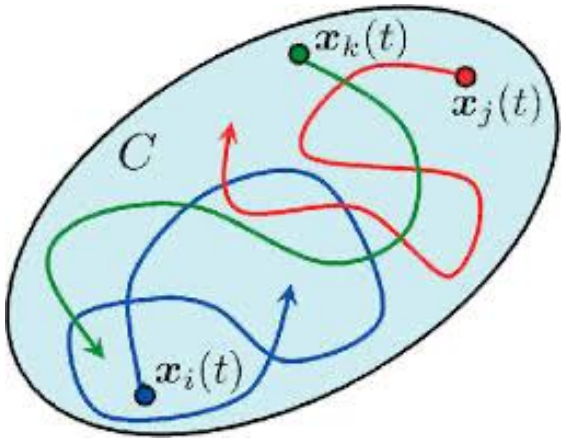


WashU

How do we define safety?

$$\dot{x} = f(x) + g(x)u$$

$$x = \begin{bmatrix} p_x \\ p_y \end{bmatrix}$$



- The safe set C is defined everywhere in the state space except for the obstacle.

$$\mathcal{C} = \{x \in R^n : b(x) \leq 0\}$$

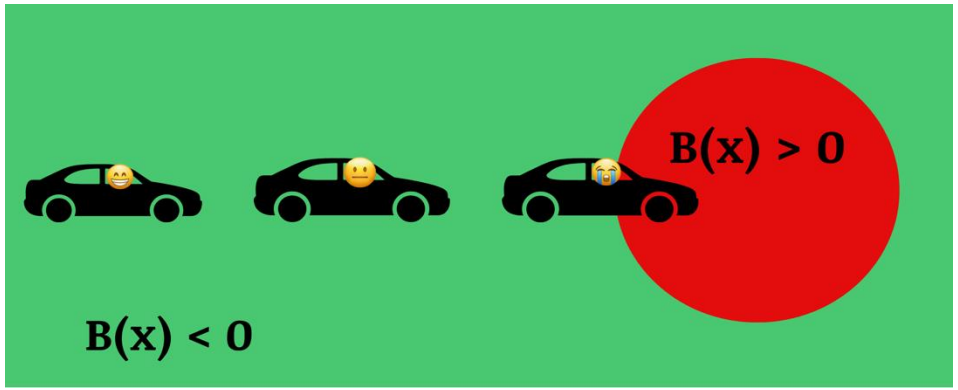
- Safety means the set is forward invariant i.e. if the state starts in C it never leaves C

$$x(0) \in \mathcal{C} \Rightarrow x(t) \in \mathcal{C}, \ \forall t \geq 0$$

WashU

Control Barrier Functions (CBFs)

$$b(x) = R^2 - \left[(p_x - x_{\text{obs}})^2 + (p_y - y_{\text{obs}})^2\right]$$

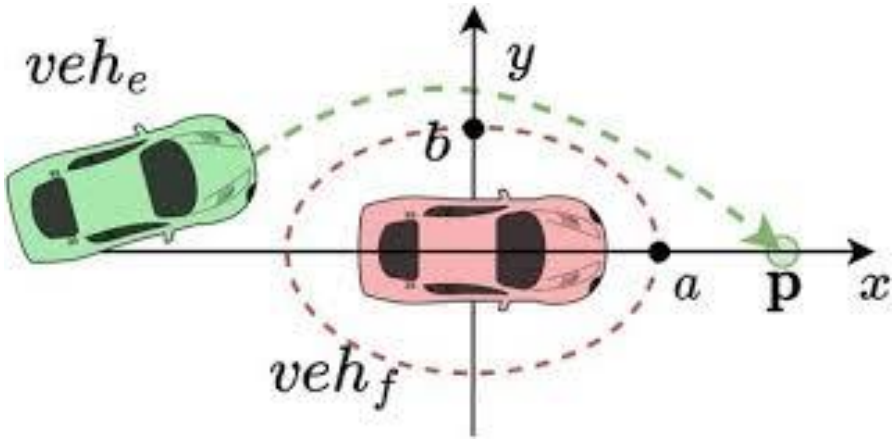


$$\text{s.t.} \quad \frac{db}{dt} \geq -\alpha b(\mathbf{x}) \qquad \frac{db}{dt} = \frac{\partial b}{\partial \mathbf{x}}\left(f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}\right)$$

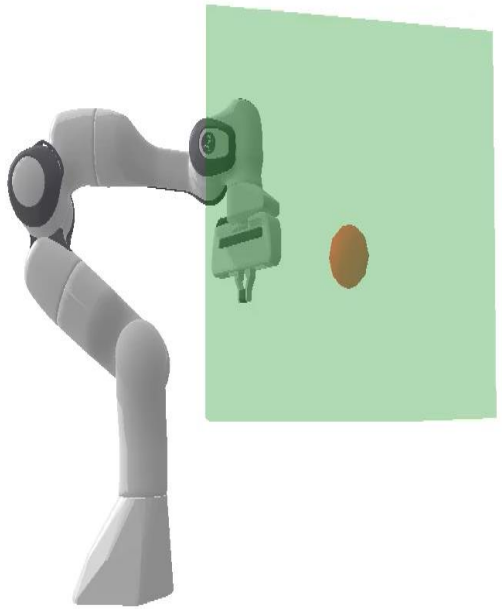
- A Control Barrier Function (CBF) defines the safe and unsafe regions
- We define the unsafe region or obstacle as a circle
- We ensure safety by imposing a constraint on the CBF

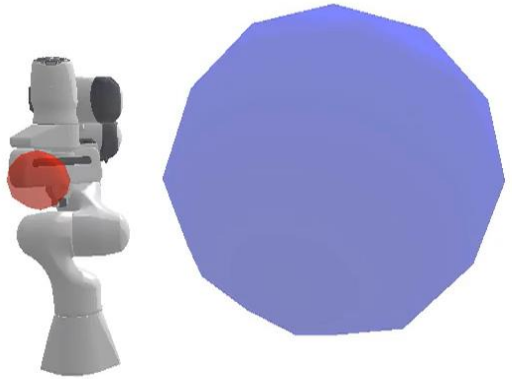
Applications of CBF's

Autonomous Driving and Locomotion



Robot Manipulators





Problem Statement

In a partially unknown environment, autonomously navigate from a starting location to a goal while preventing collisions with unforeseen and Dynamic obstacles.

Four Steps of Autonomous Navigation

Localization

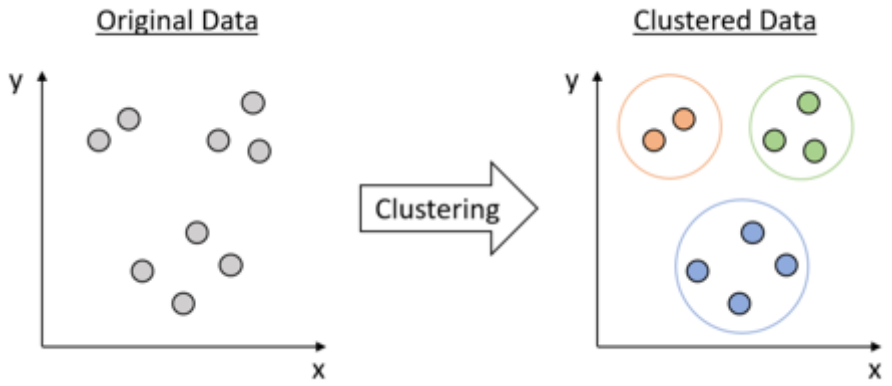
Perception

Path Planning

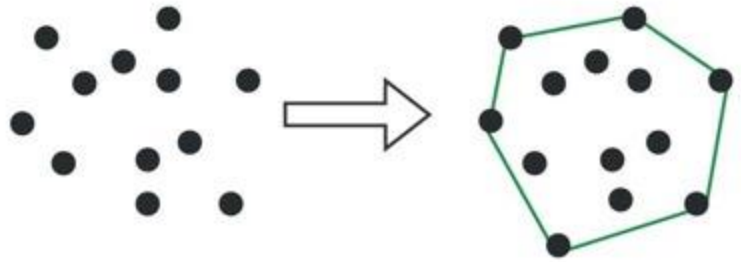
Control

Perception Pipeline

Perception Pipeline



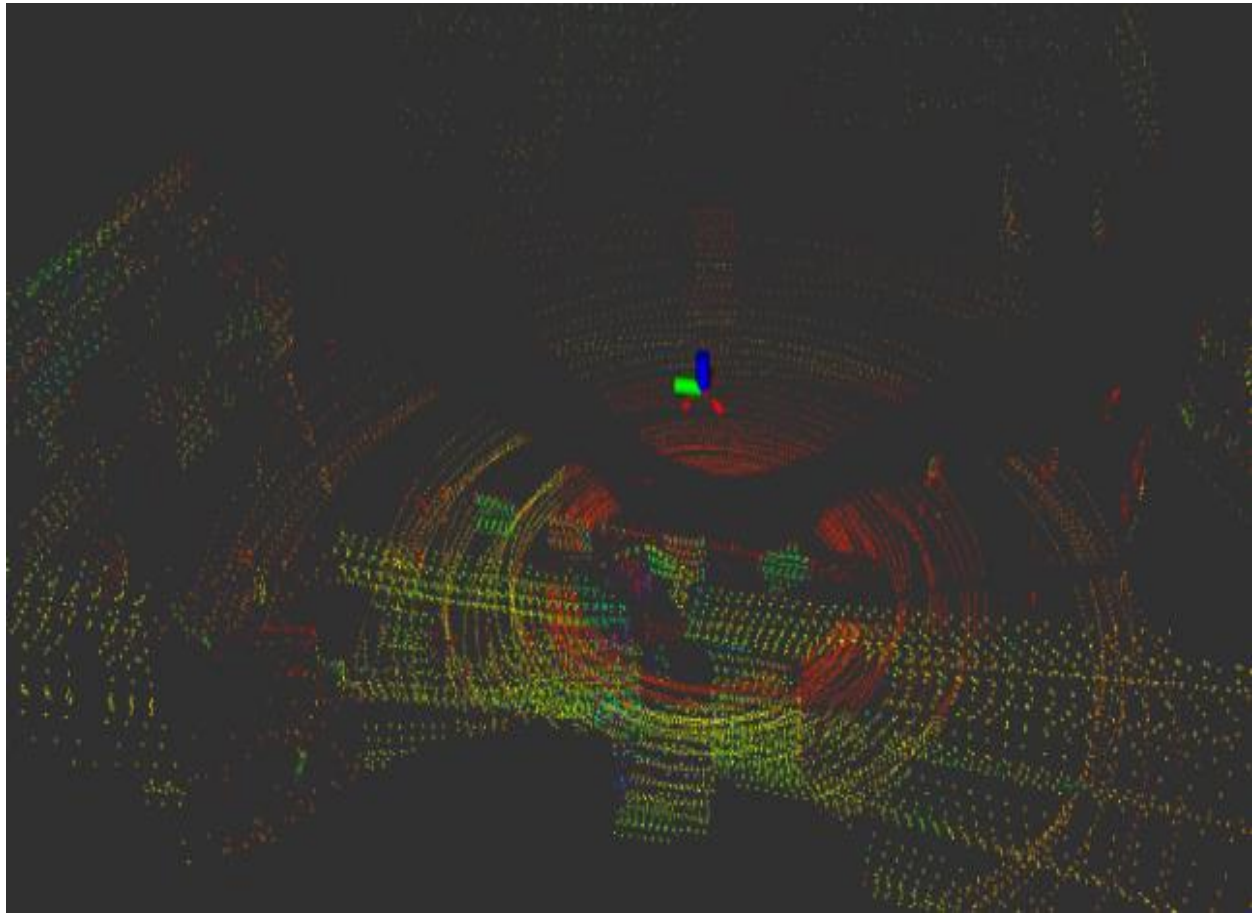
Filtering and Clustering



Shape Wrapping

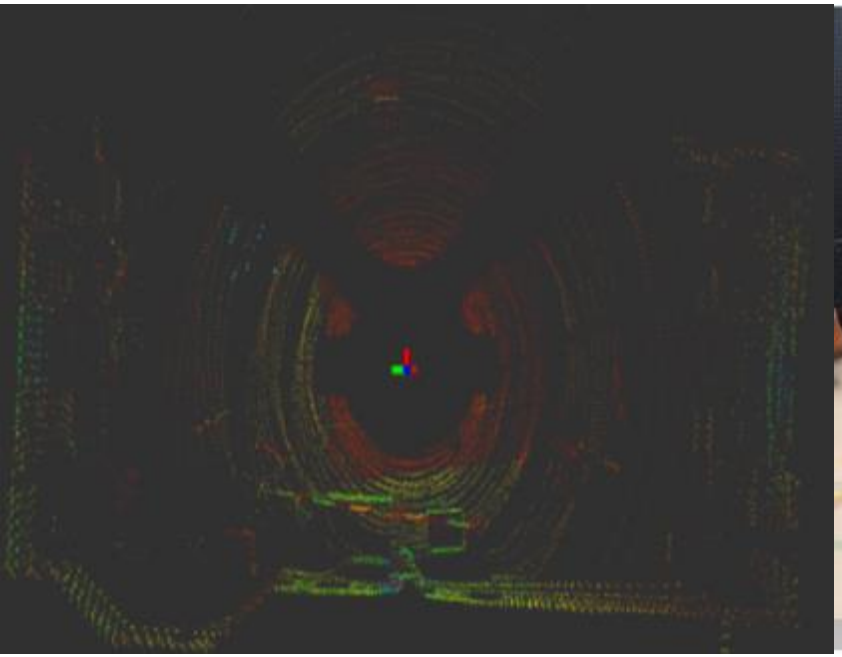
Filtering Lidar Points

- Original Point Cloud
  - ➢ *~ 1500 data points*
- Down-sampled using Voxel Grid Filter
  - ➢ *~ 800 data points*
- Filter points outside a 2x1 meter box from the Plant
  - ➢ *~ 150 data points*
- Filter Out Floor(1.2 Meters from Z Plane)
  - ➢ *~ 100 data points*



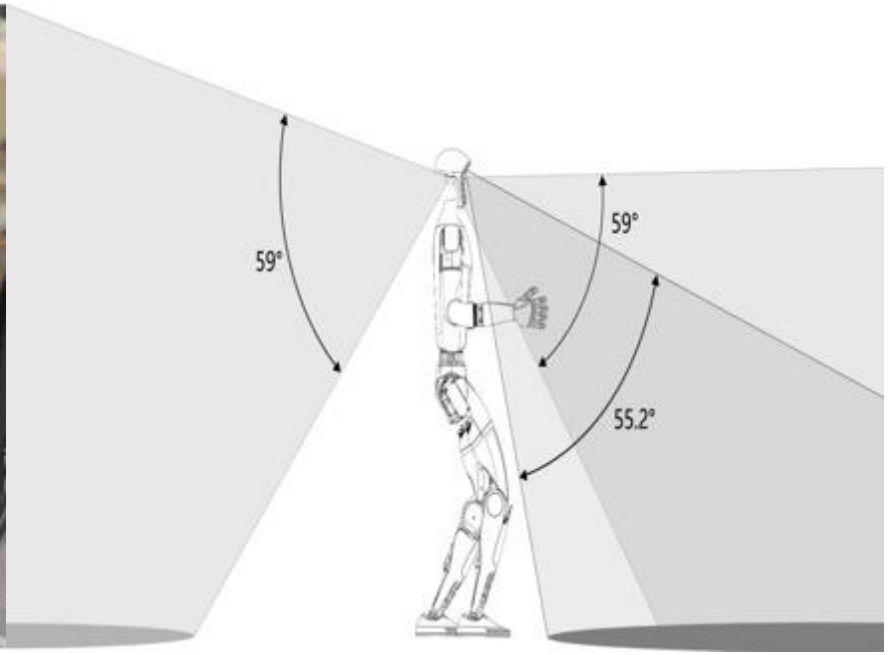
Unfiltered Lidar Data From Humanoid

Humanoid Blind Spots



Top Down View of Lidar Data

Back View of Lidar



FOV of Lidar and Camera

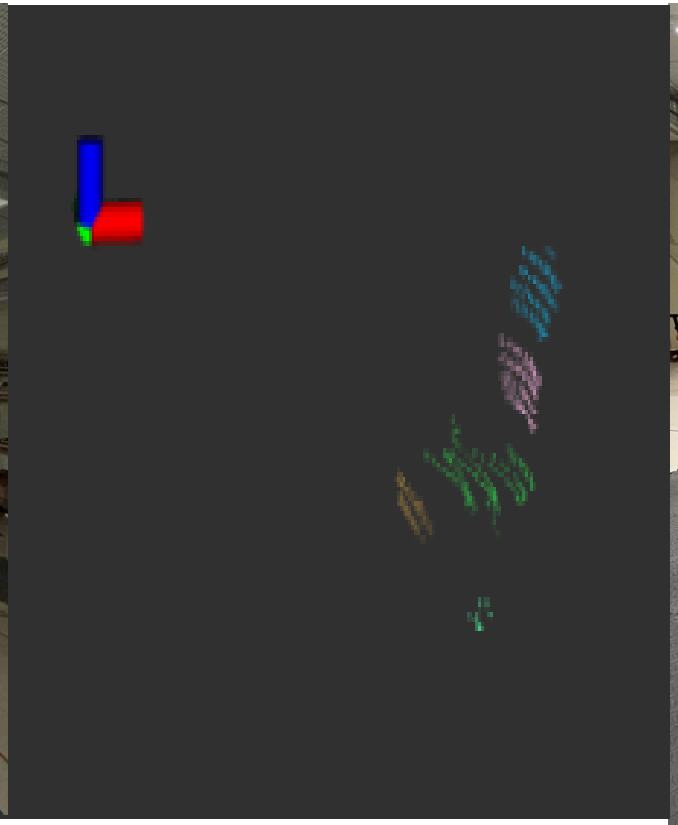
$$d = h \cdot \tan(\theta) = 1.2 \cdot \tan(59°) \approx 1.997 \text{ m} \qquad d = (h - h_o) \cdot \tan(\theta)$$

WashU

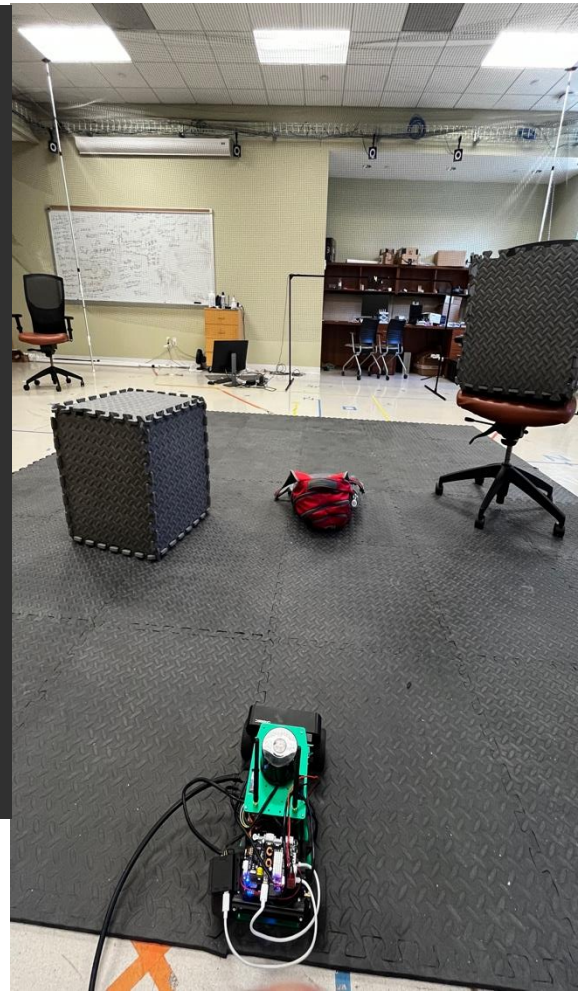
Clustering Lidar Points

What we see

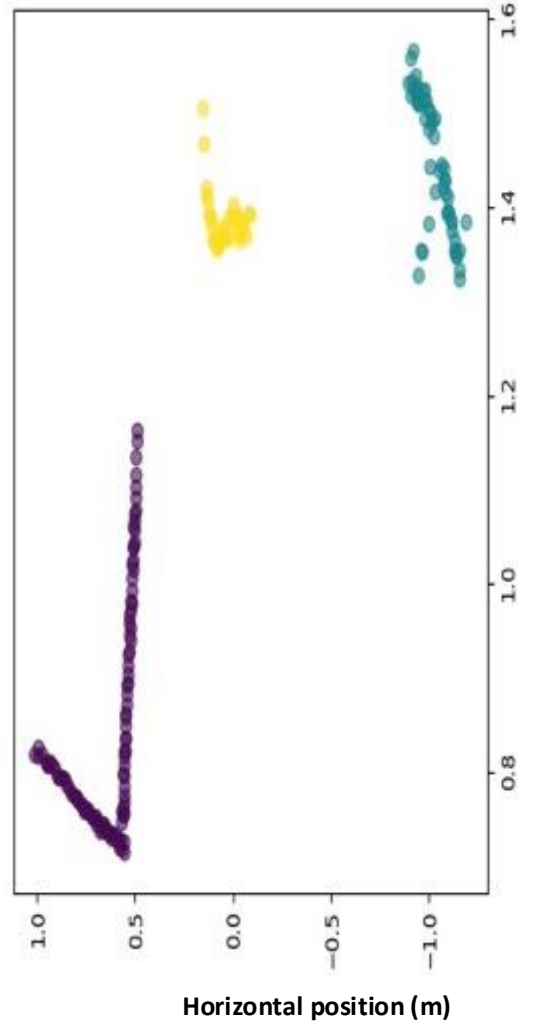
What the Humanoid Detects



What we see



What the Car detects



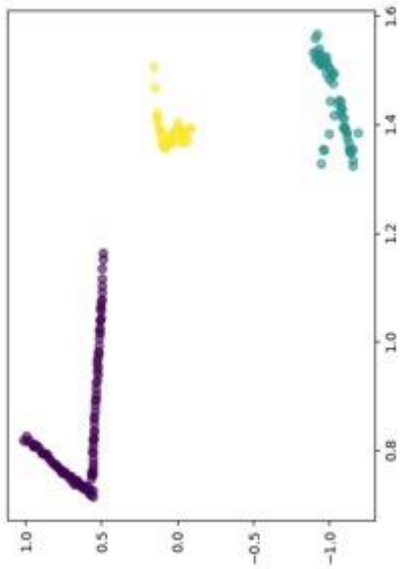
- Latency Humanoid: (.3s)
- Latency 1/10 Car: (.8s)

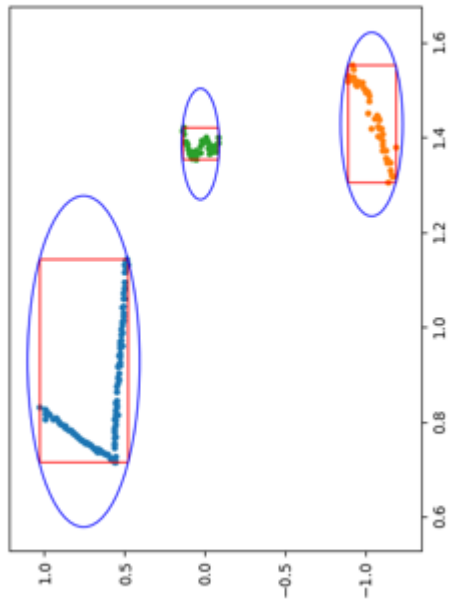
WashU

Obstacle Shape Wrapping

Box Method

- Compute the minimum and maximum x and y coordinates of the cluster.
- Define box vertices and calculate the centroid.





$$X_1 = min(x_1, x_2, \dots, x_n)$$
$$X_2 = max(x_1, x_2, \dots, x_n)$$
$$Y_1 = min(y_1, y_2, \dots, y_n)$$
$$Y_2 = max(y_1, y_2, \dots, y_n)$$

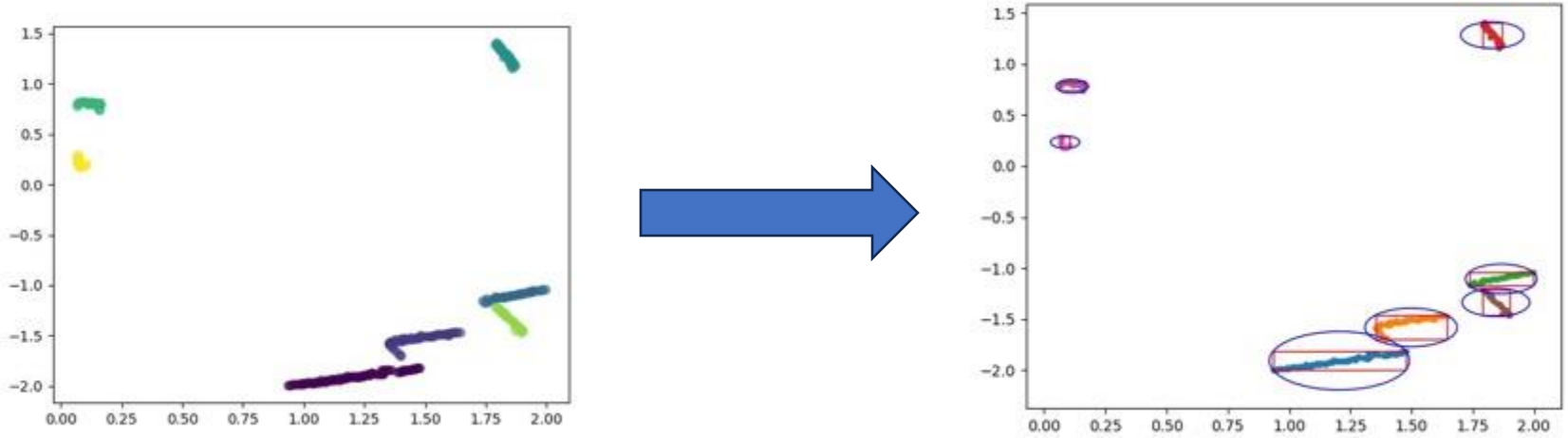
$$x_C = \frac{X_{min} + X_{max}}{2}$$
$$y_C = \frac{Y_{min} + Y_{max}}{2}$$

WashU

Obstacle Shape Wrapping

Integration into CBF filter

- $X_C = X_{obs}$
- $Y_C = Y_{obs}$

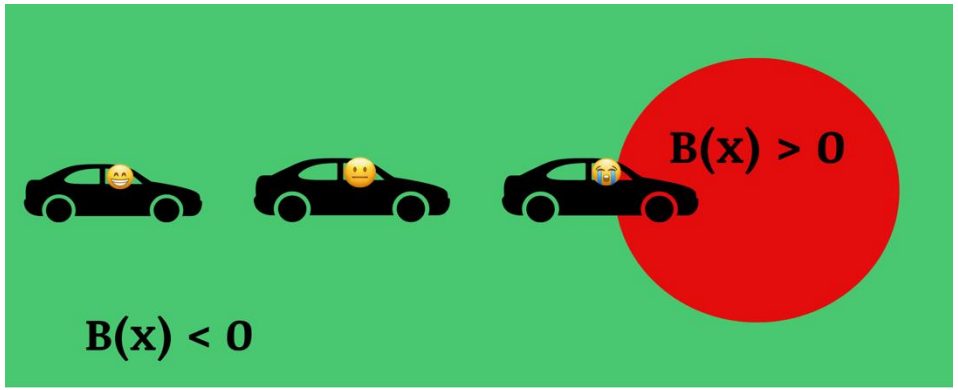


$$\text{radius} = \sqrt{(x_{\text{obs}} - x_{\text{max}})^2 + (y_{\text{obs}} - y_{\text{max}})^2}$$

CBF Based Control

Control Barrier Functions (CBFs)

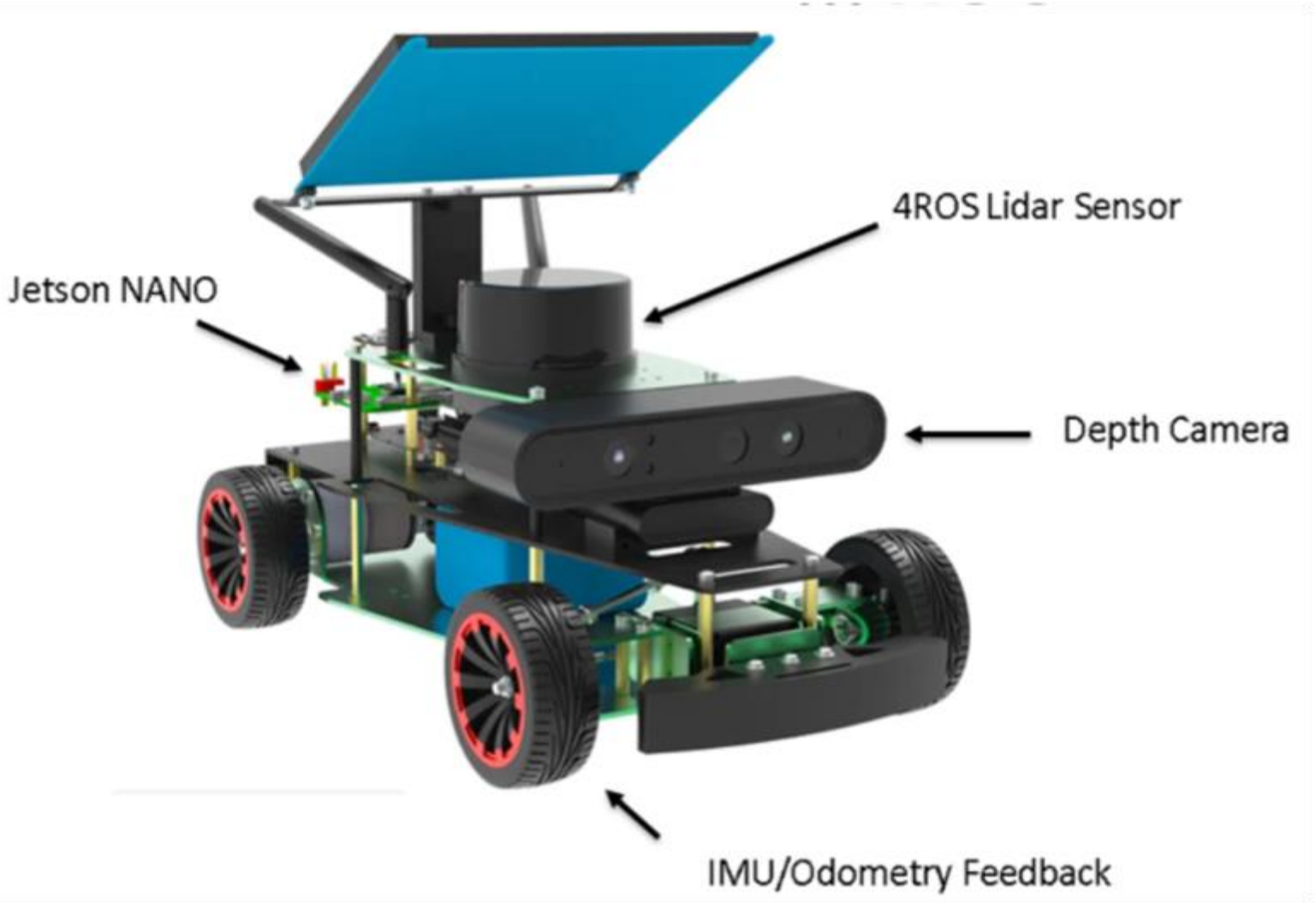
$$b(x) = R^2 - \left[(p_x - x_{\text{obs}})^2 + (p_y - y_{\text{obs}})^2\right]$$



$$\text{s.t.} \quad \frac{db}{dt} \geq -\alpha b(\mathbf{x}) \qquad \frac{db}{dt} = \frac{\partial b}{\partial \mathbf{x}}\left(f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}\right)$$

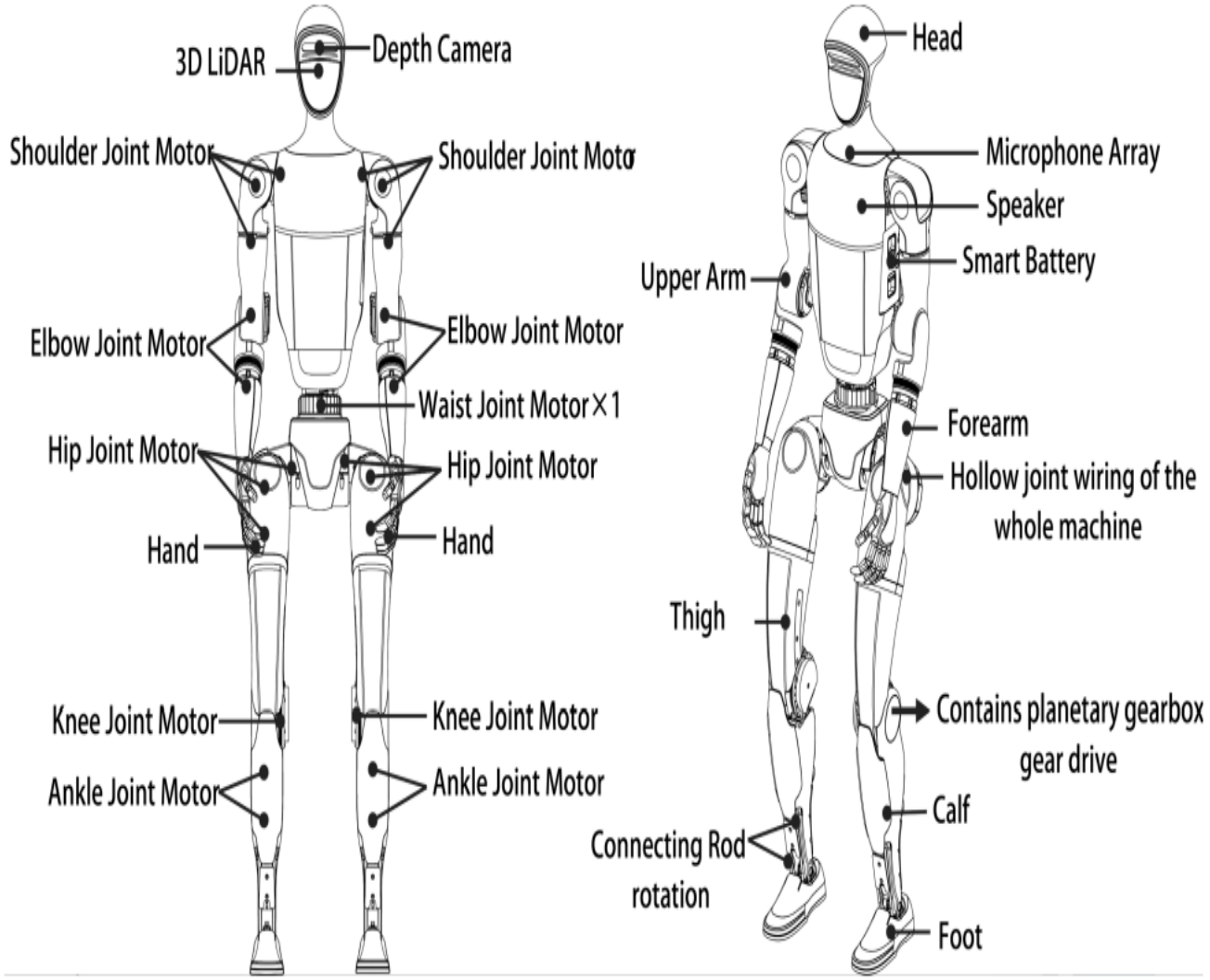
- A Control Barrier Function (CBF) defines the safe and unsafe regions
- We define the unsafe region or obstacle as a circle
- We ensure safety by imposing a constraint on the CBF

1/10 Scale Ackerman Car



- NVIDIA Jetson TX2-NX module
  - Quadcore ARM Cortex CPU
  - 2.0 GHz(typical: ~1.5 GHz)
- Sensors: IMU, LiDAR, Depth Camera, Odometer
- Localizes using EKF Sensor Fusion (IMU + LiDAR + Odometry)

G1- Humanoid



- 43 Degrees of Freedom (DoF)
- NVIDIA Jetson Orion module
  - Octa-core Arm Cortex-CPU
  - 2.0 GHz (typical: ~1.6 GHz)
- Sensors: Imu,Lidar,Depth Camera, Odometer
- Localizes using Odometer in World Frame

Dynamics

$$\dot{x} = f(x) + g(x)u$$

Single Integrator Dynamics

$$x = \begin{bmatrix} p_x \\ p_y \end{bmatrix}, \quad u = \begin{bmatrix} v_x \\ v_y \end{bmatrix}, \quad \dot{x} = \begin{bmatrix} v_x \\ v_y \end{bmatrix}$$

$$f(x) = 0 \text{ and } g(x) = I$$

Unicycle Dynamics

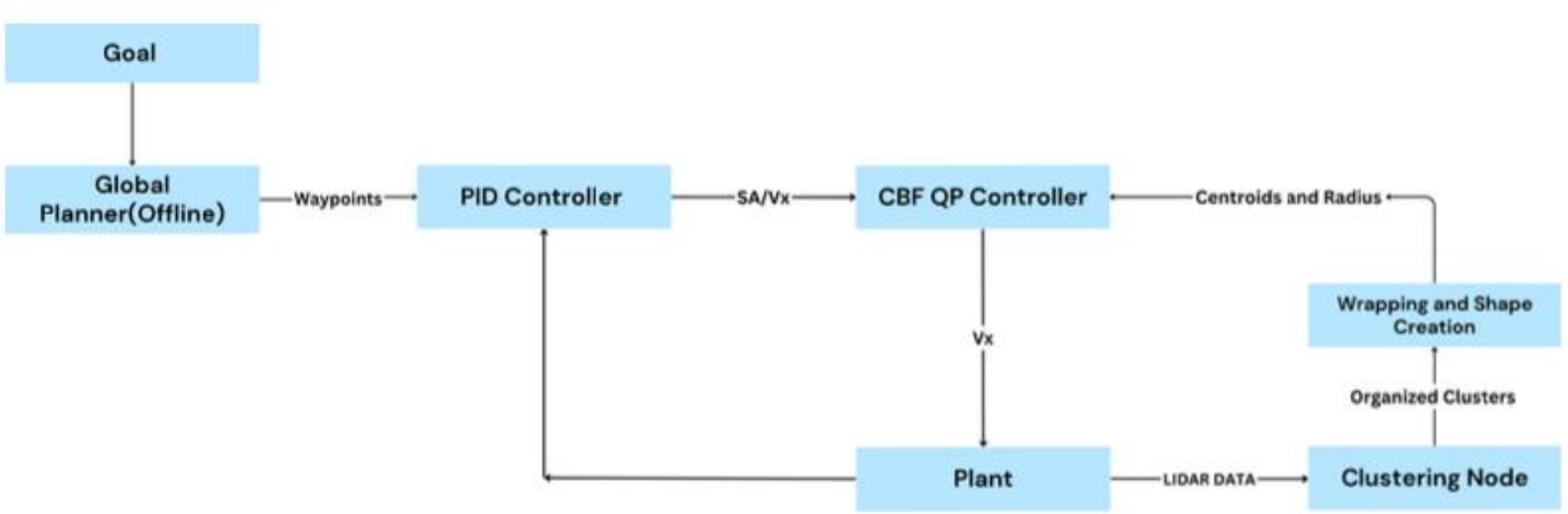
$$x = \begin{bmatrix} \theta \\ p_x \\ p_y \end{bmatrix}, \quad u \in R^{2\times 1}, \quad u = \begin{bmatrix} \omega \\ v \end{bmatrix}, \quad \dot{x} = \begin{bmatrix} \omega \\ v\cos(\theta) \\ v\sin(\theta) \end{bmatrix}$$

$$\dot{x} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & \cos(\theta) \\ 0 & \sin(\theta) \end{bmatrix} \begin{bmatrix} \omega \\ v \end{bmatrix}$$

	Unicycle	Single Integrator
Humanoid	✓	✓
1/10 Scale Car	✓	✗

PID/CBF Quadratic Program (QP) Controller

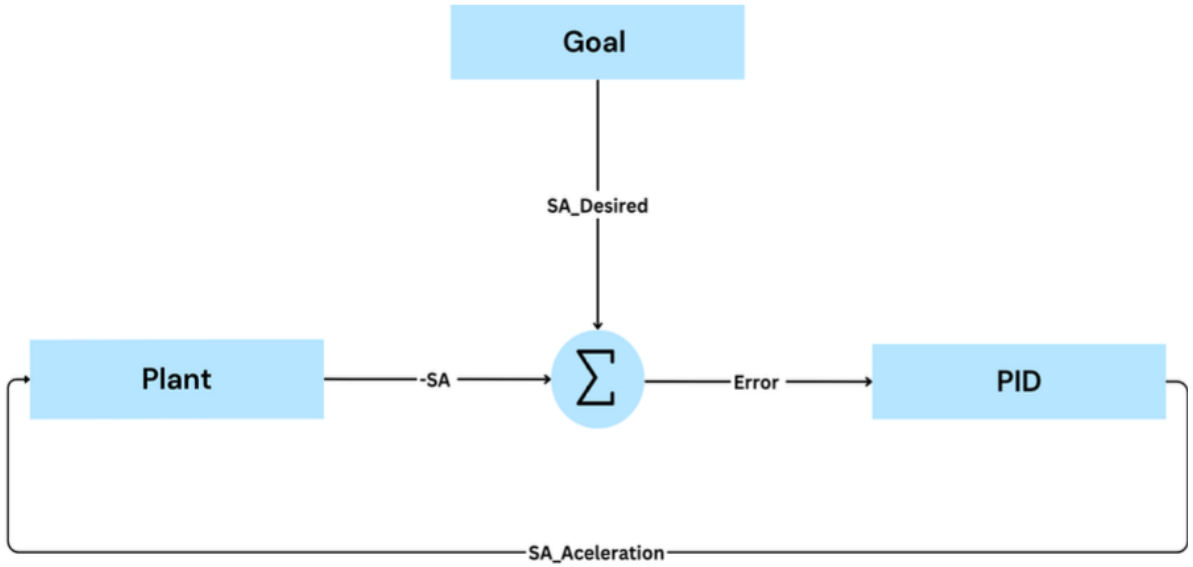
Reach & Avoid Pipeline



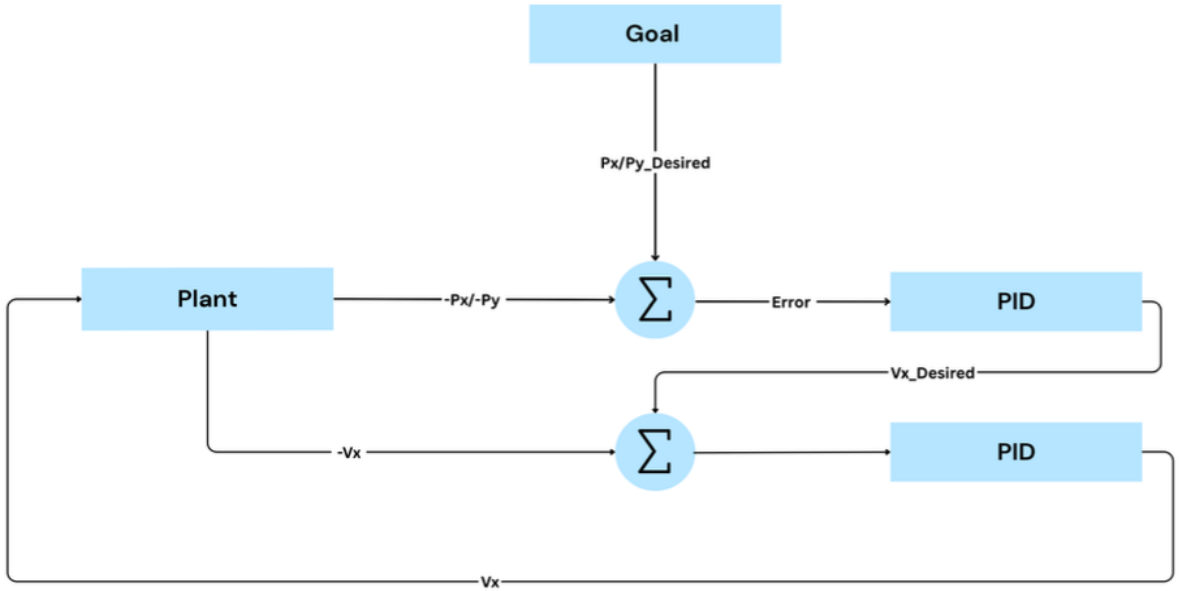
WashU

PID Formulation

Our PID Controllers controls States $x = \begin{bmatrix} \theta \\ p_x \\ p_y \end{bmatrix}$ through control Inputs $u = \begin{bmatrix} \omega \\ v \end{bmatrix}$



Steering Angle Controller



Position Controller

WashU

QP Formulation

Goal: Find U that is as close as possible to the PID signal, while ensuring safety by satisfying the barrier function constraint.

$$\mathbf{u} = \min \|\mathbf{u} - \mathbf{u}_{\text{PID}}\|^2 \quad \text{s.t.} \quad \frac{db}{dt} \geq -\alpha b(\mathbf{x})$$

$$\textbf{where} \quad \frac{db}{dt} = \frac{\partial b}{\partial \mathbf{x}} \left(f(\mathbf{x}) + g(\mathbf{x})\mathbf{u}\right)$$

PID/CBF Sticking Point 1/10 Car

Sticking Point: a non-ideal equilibrium point where the dynamics or constraints prevent the plant from naturally moving towards the goal.

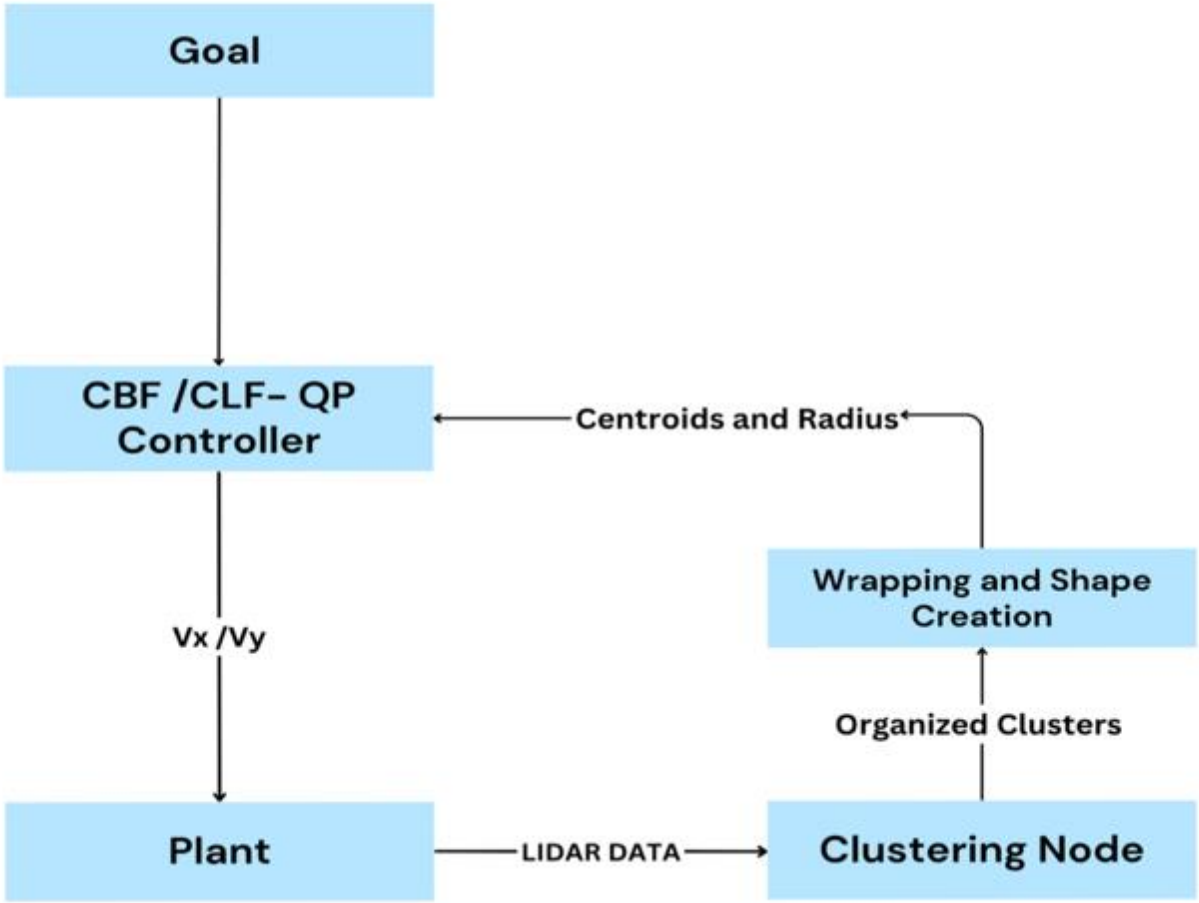
CBF: $b(x) = R^2 - \left[(p_x - x_{\text{obs}})^2 + (p_y - y_{\text{obs}})^2\right]$

CBF Constraint Unicycle Dynamics

$$\left[2(p_x - x_{\text{obs}})\cos\theta + 2(p_y - y_{\text{obs}})\sin\theta\right] v + 0 \cdot \omega \geq 0$$

CLF/CBF QP Controller

Reach & Avoid Pipeline



WashU

Control Lyapunov Function (CLF)

$$V(x) = \alpha\left((p_x - x_{\text{goal}})^2 + (p_y - y_{\text{goal}})^2\right)$$

- $V(x) > 0$ for all $x \neq x_{goal}$, and $V(x_{goal}) = 0$,
- $V(x) \to \infty$ as $\|x - x_{goal}\| \to \infty$,
- $\dot{V}(x) = \nabla V(x)^T(f(x) + g(x)u) < 0$ for some control input u.

QP Formulation

Goal: Find U that is as close as possible to the CLF signal, while ensuring safety by satisfying the barrier function constraint.

$$\min_{u} \quad (u - u_{\text{CLF}})^T (u - u_{\text{CLF}})$$

$$\text{s.t.} \quad \nabla_x b_i(x)(f(x) + g(x)u) \geq -\alpha b_i(x), \quad \forall i = 1, \ldots, N$$

$$u_{\text{CLF}} = \alpha \begin{bmatrix} (p_x - x_{\text{goal}}) \\ (p_y - y_{\text{goal}}) \end{bmatrix}$$

CBF for Multiple Obstacles

$$b(x) = R^2 - \left[(p_x - x_{\text{obs}})^2 + \left((\,|p_y - y_{\text{obs}}| + 1\,)^3 - 1\right)\right]$$

- Cube term emphasizes obstacles directly in front of the vehicle, making the system more sensitive to obstacles aligned with its forward trajectory.
- If the y term is below 1, when cubed, it becomes smaller. In order to keep the steepness of the y-dependent term, 1 was added in and then subtracted after taking the cube
- Obstacles farther to the side contribute less, because the squared term grows more slowly, allowing the system to place stronger constraints on objects within the forward motion of the plant

QP Constraints for Multiple Obstacles

$$\min_{u} \quad (u - u_{\text{CLF}})^T (u - u_{\text{CLF}})$$

$$\text{s.t.} \quad \nabla_x b_i(x)(f(x) + g(x)u) \geq -\alpha b_i(x), \quad \forall i = 1, \ldots, N$$

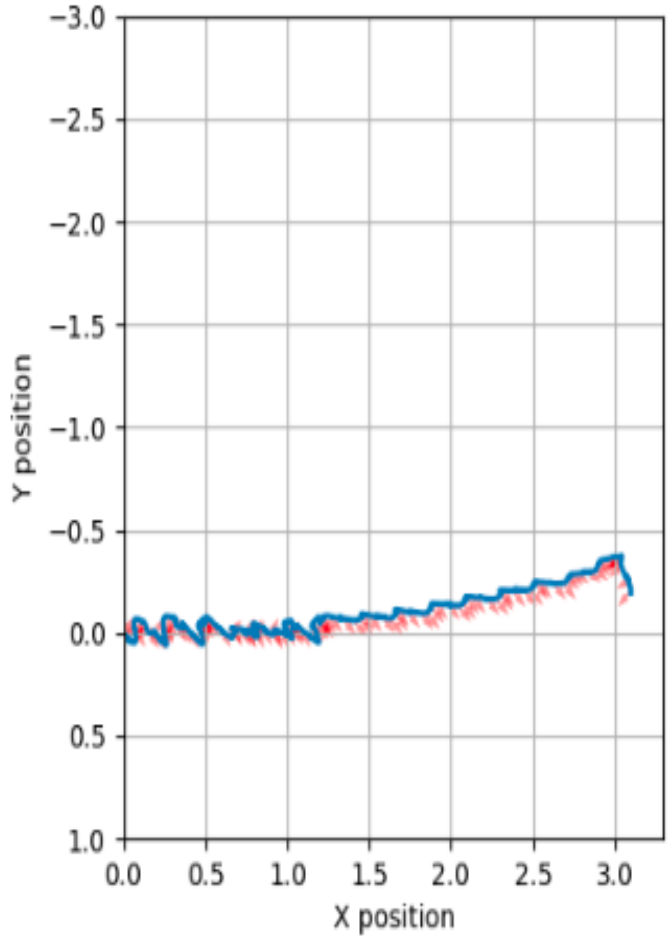
For every detected obstacle, a new constraint is added to the QP to ensure safety

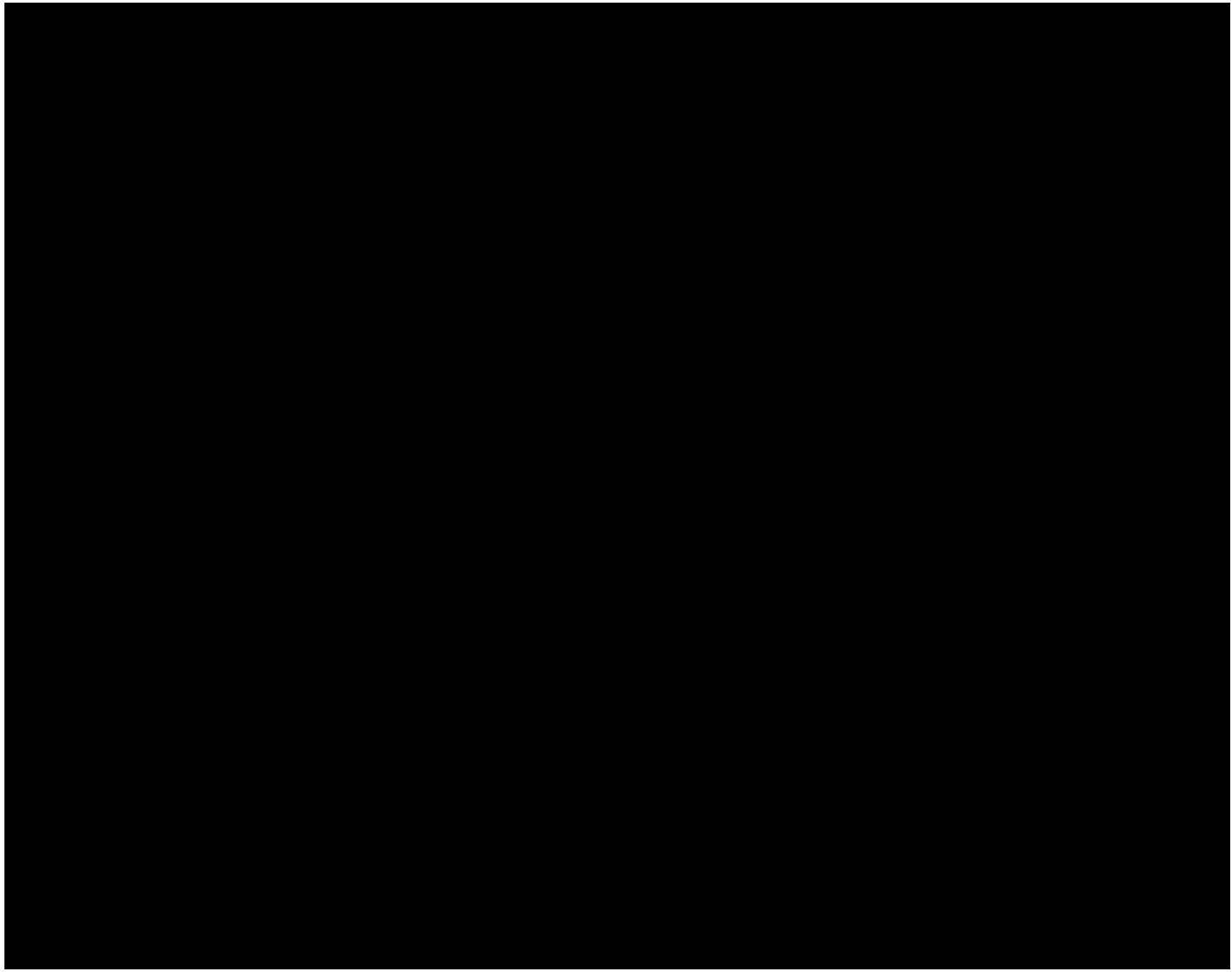
$$\alpha_i = \frac{\text{safety gain}}{(1 + y_{\text{centroid}i})^3} \quad i = 1, 2, \ldots, N$$

Alpha weights the constraints and prioritizes obstacles aligned with the trajectory of the car.

Results

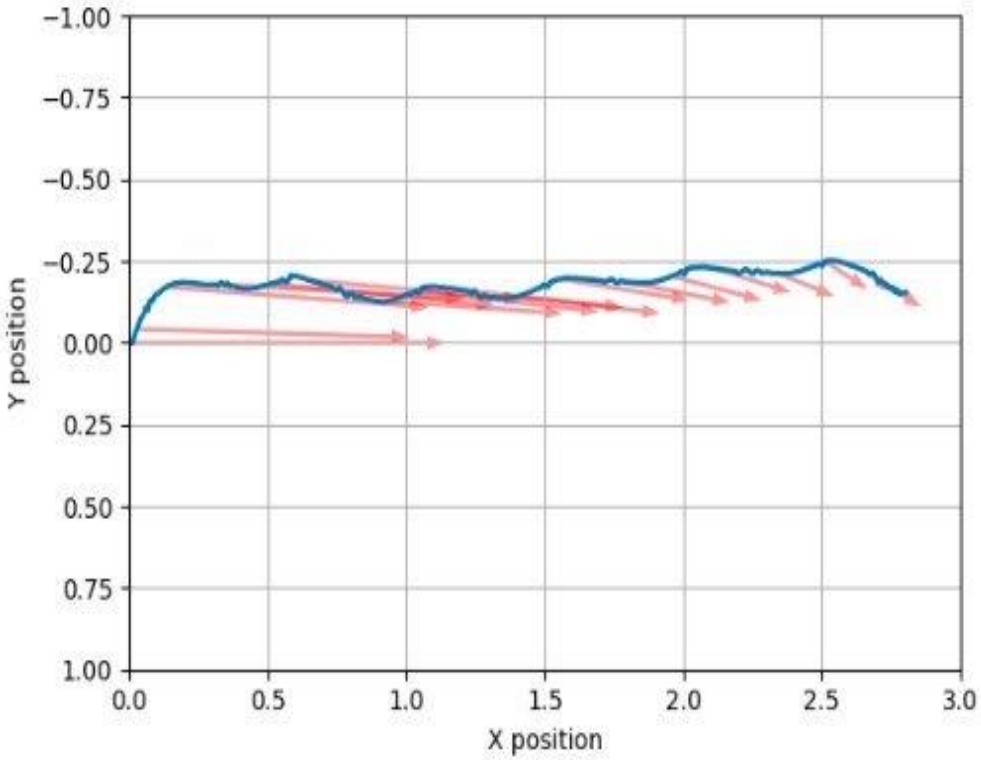
No Obstacle CBF/CLF



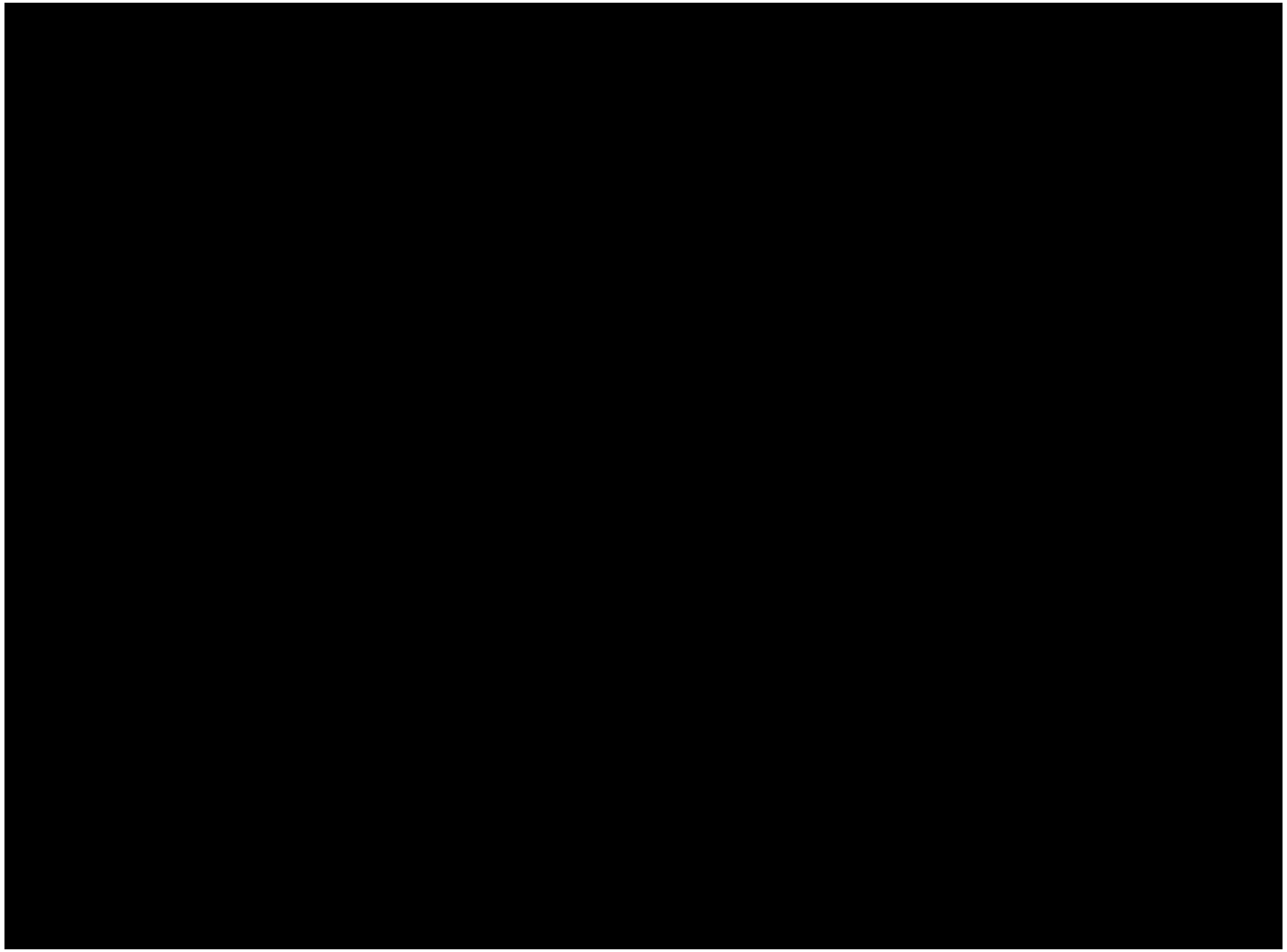


WashU

No Obstacle PID



(b) VxVy PID Controller



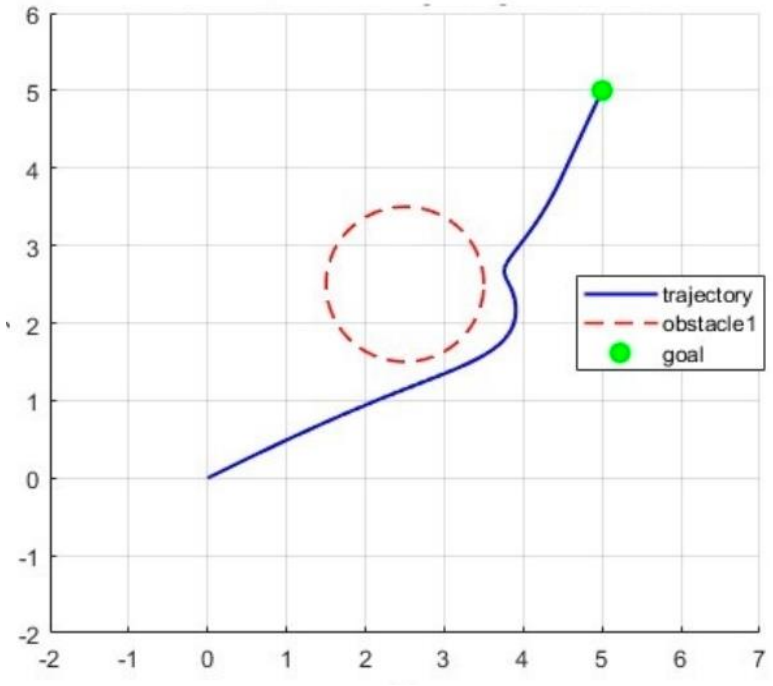
WashU

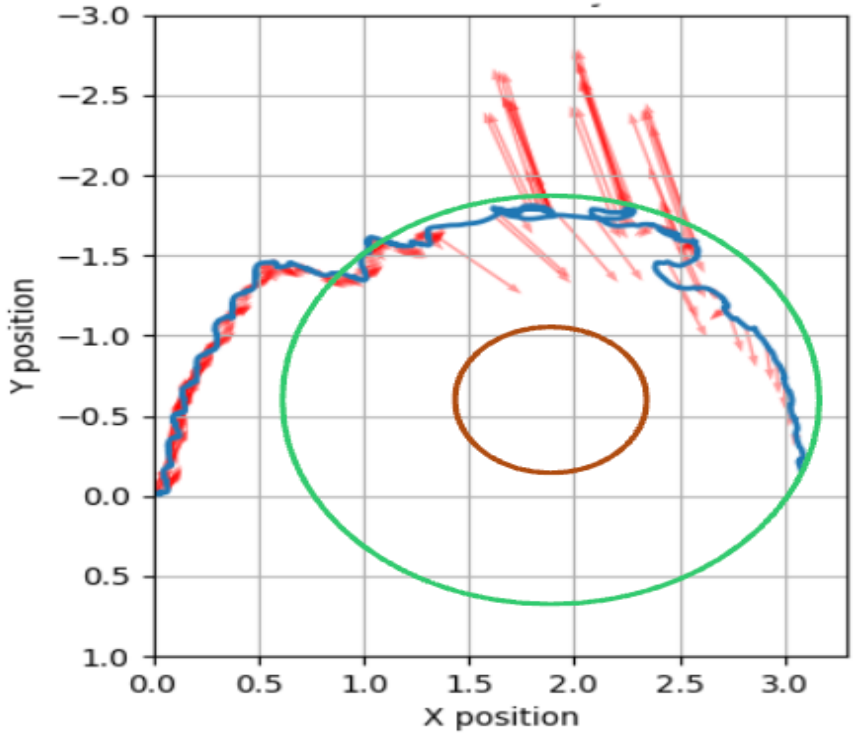
Preliminary Remarks

- **Odometry drift** affected the smoothness and accuracy of trajectory plots
- **LiDAR range limited**: Obstacles >2 m ahead or ±1 m sideways were ignored
- **Safety gain** set to 1.2 for all runs
- **CBF response triggered** only when safety constraint was violated
- **Velocity plots down sampled**: Only 1 in 16 commands shown for clarity
- **Forward motion prioritized**: Vx_ reduced more than Vy due to frontal obstacle bias

$$v_x = x[0] \cdot 0.05, \quad v_y = x[1] \cdot 0.4$$

Single Obstacle Simulation vs Real





Safety Buffer

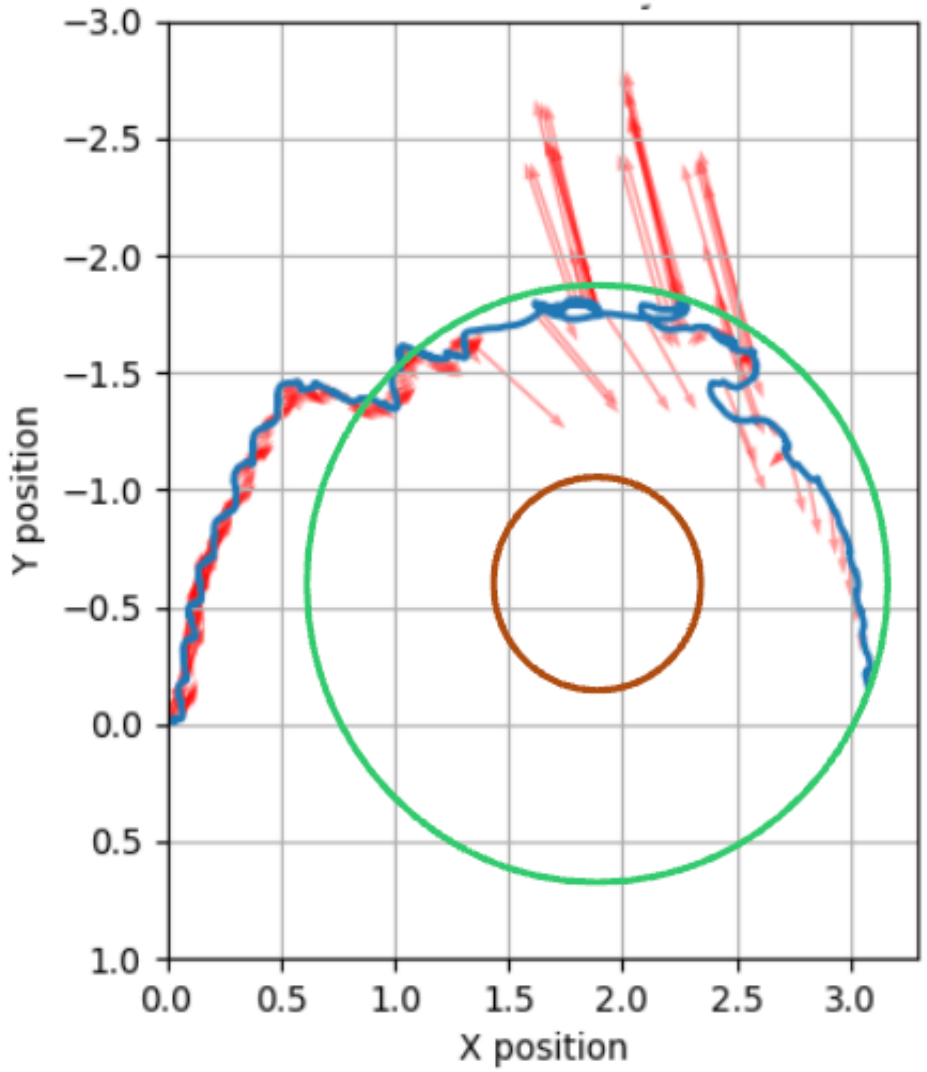
Obstacle Position

Humanoid Position

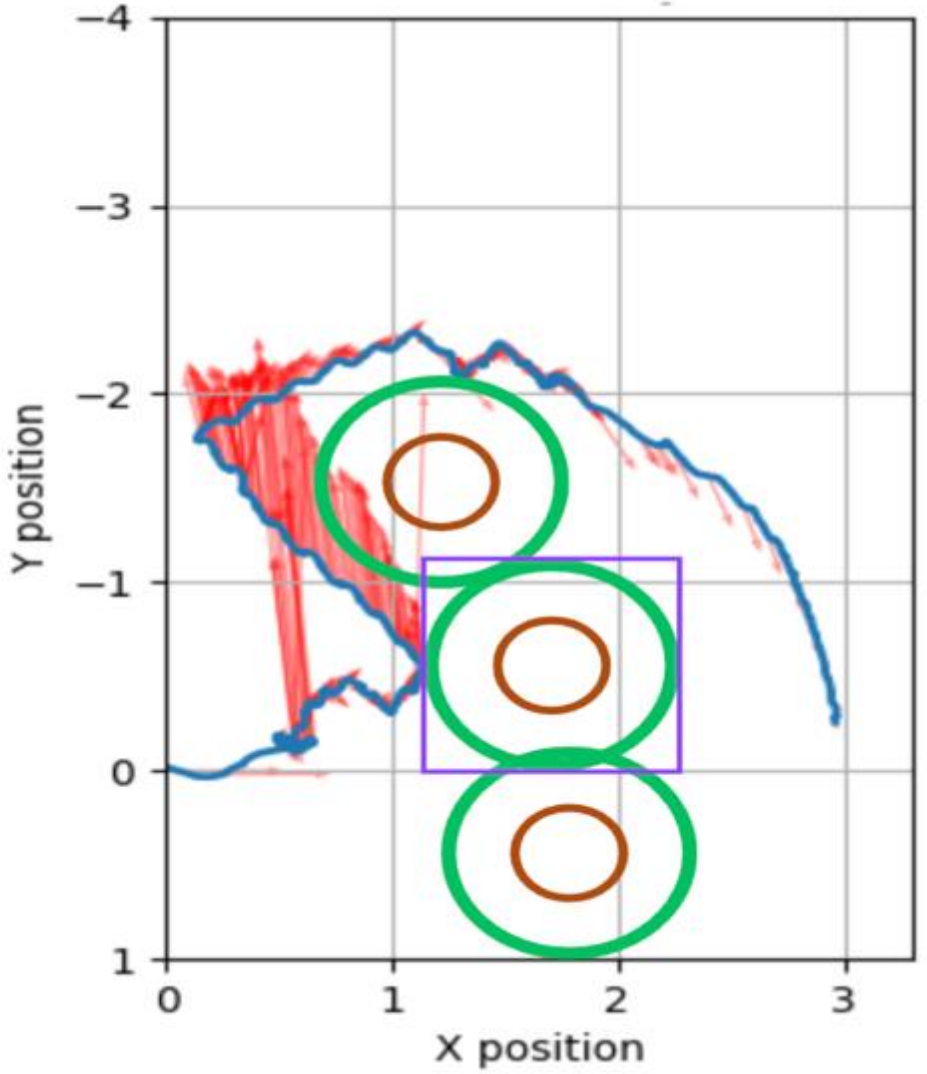
Humanoid Velocity

WashU

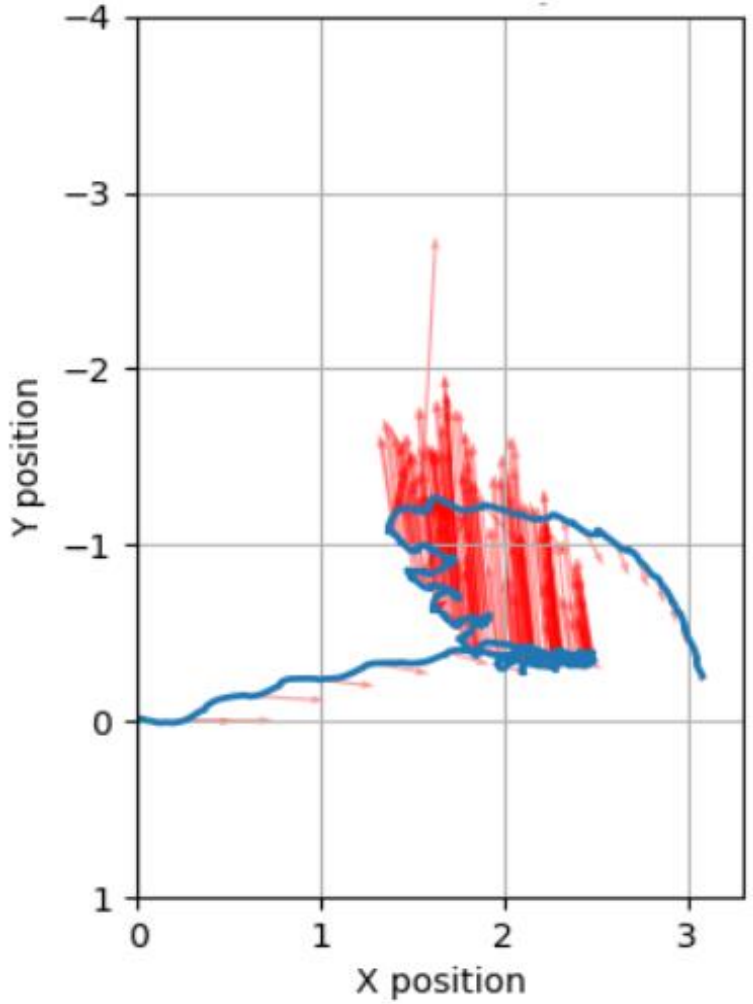
Single Obstacle

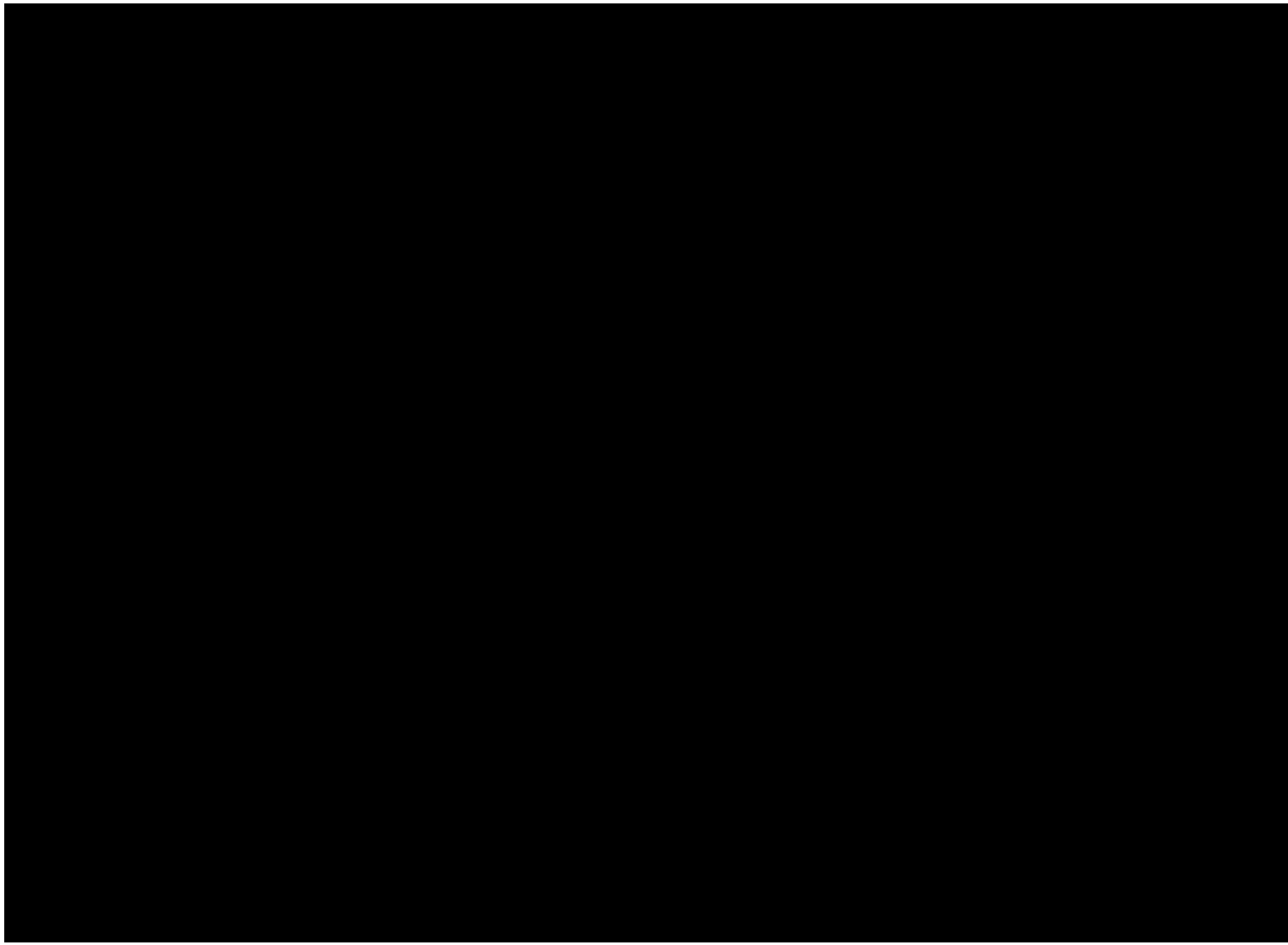


Wall of Obstacles



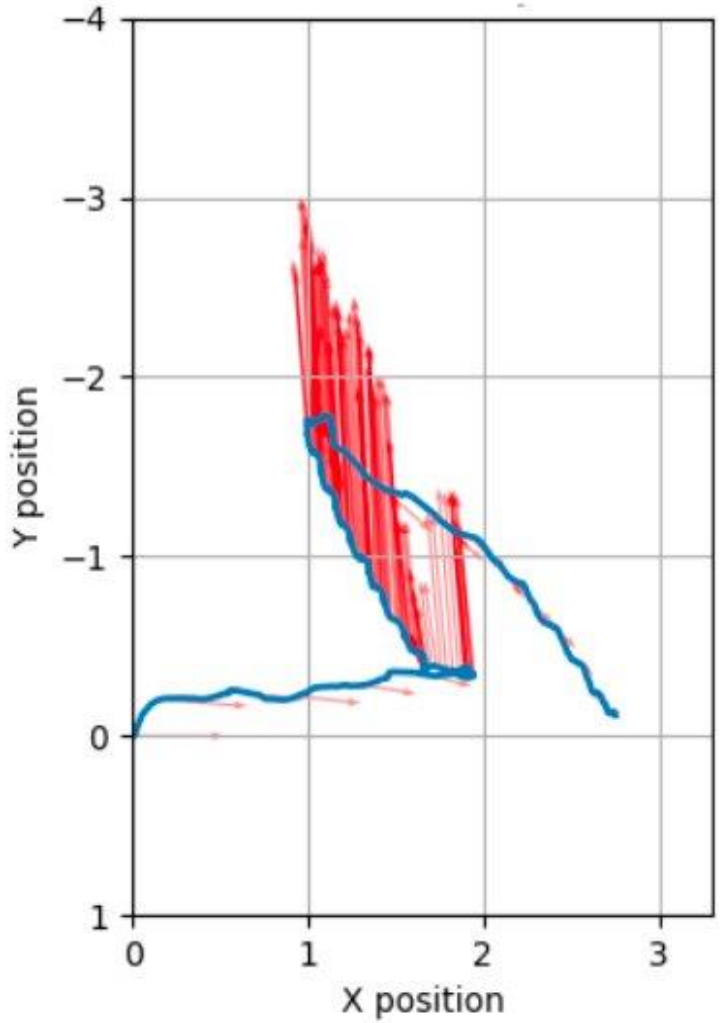
Moving Obstacle on Goal Point





WashU

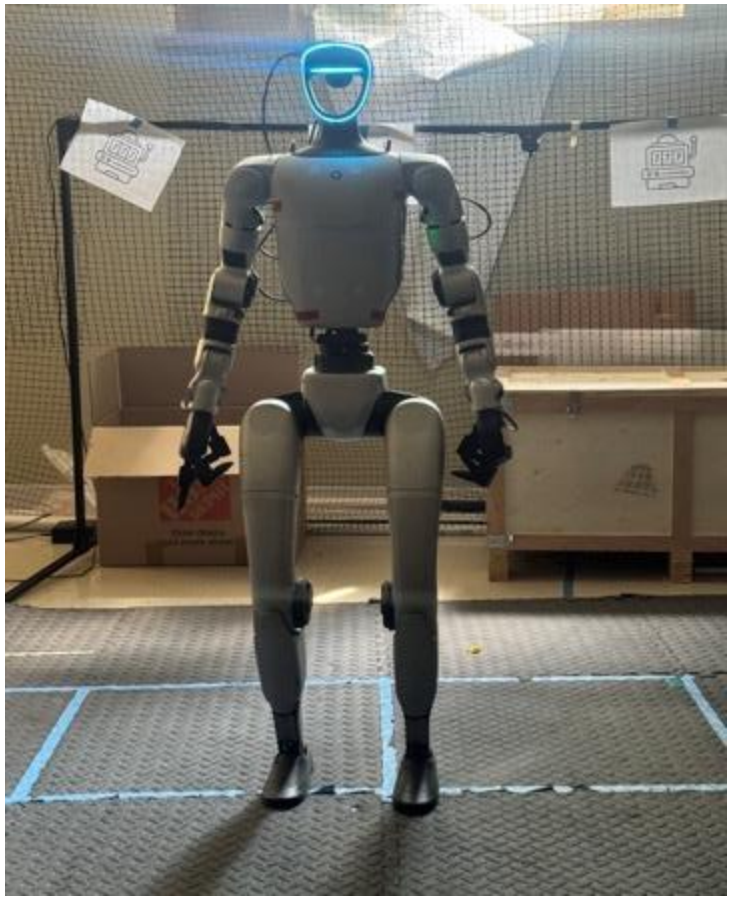
Approaching Moving Obstacle



WashU

Final Remarks

- Controller enables safe, convergent behavior on the G1 Humanoid
- Successfully avoids both static and dynamic obstacles without path planning
- Failures were not due to control formulation—but sensor and hardware limitations
- Main issues: LiDAR blind spots and odometry drift
- QP controller reacts correctly when obstacles re-enter field of view



Contributions

1/10 Scale Car

- Open Loop Control
- Create Perception Pipeline
- Position and SA PID Controller
- Static & Dynamic CBF/PID QP Controller
- Multiple Obstacle CBF/PID QP Controller

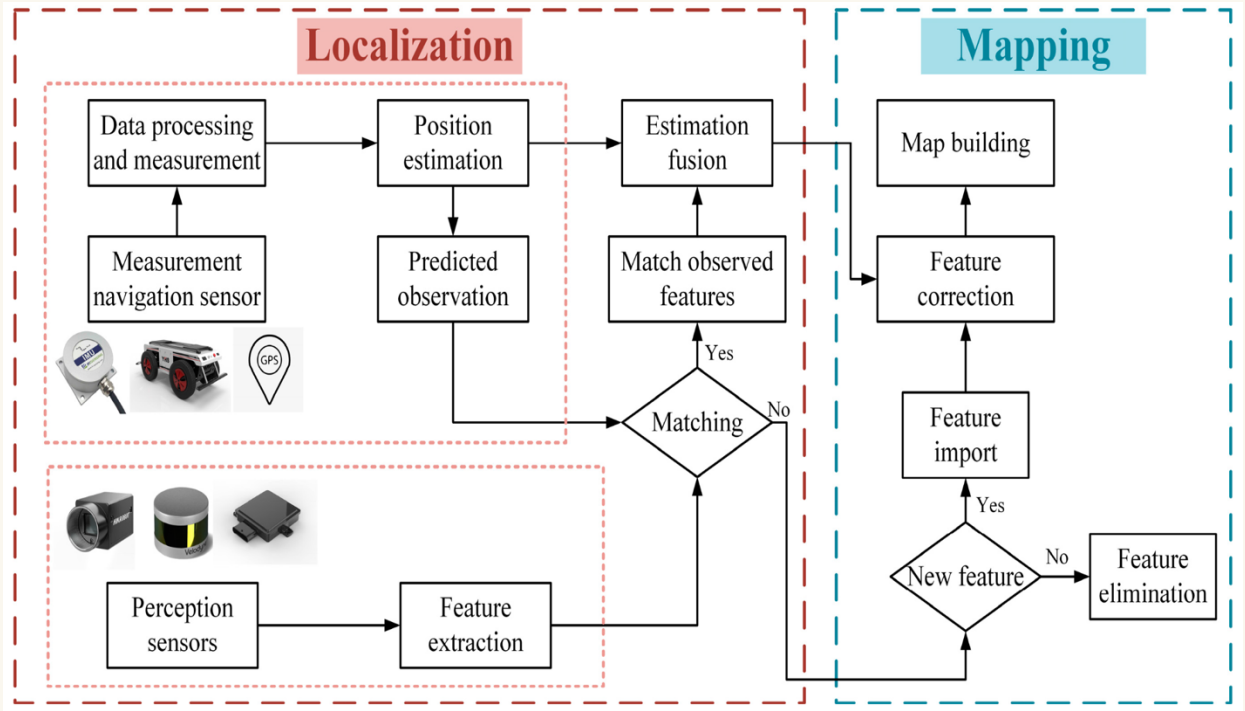
G1 Humanoid

- Open Loop Control
- Create Perception Pipeline
- Position and SA PID Controller (Unicycle Dynamics)
- Position PID Controller
- Static & Dynamic CBF/CLF QP Controller
- Multiple Obstacle CBF/CLF QP Controller

Future Work

Future Work

- Increased performance testing
- Implement SLAM Sensor Fusion
- Improve shape wrapping to reduce obstacle overestimation
- Adapt CBF/CLF to unicycle dynamics
- Extend CBF with alignment-based term:



SLAM Based Sensor Fusion Pipeline

$$b(x, y, \theta) = (x - x_o)^2 + (y - y_o)^2 + \theta^2 - R^2$$

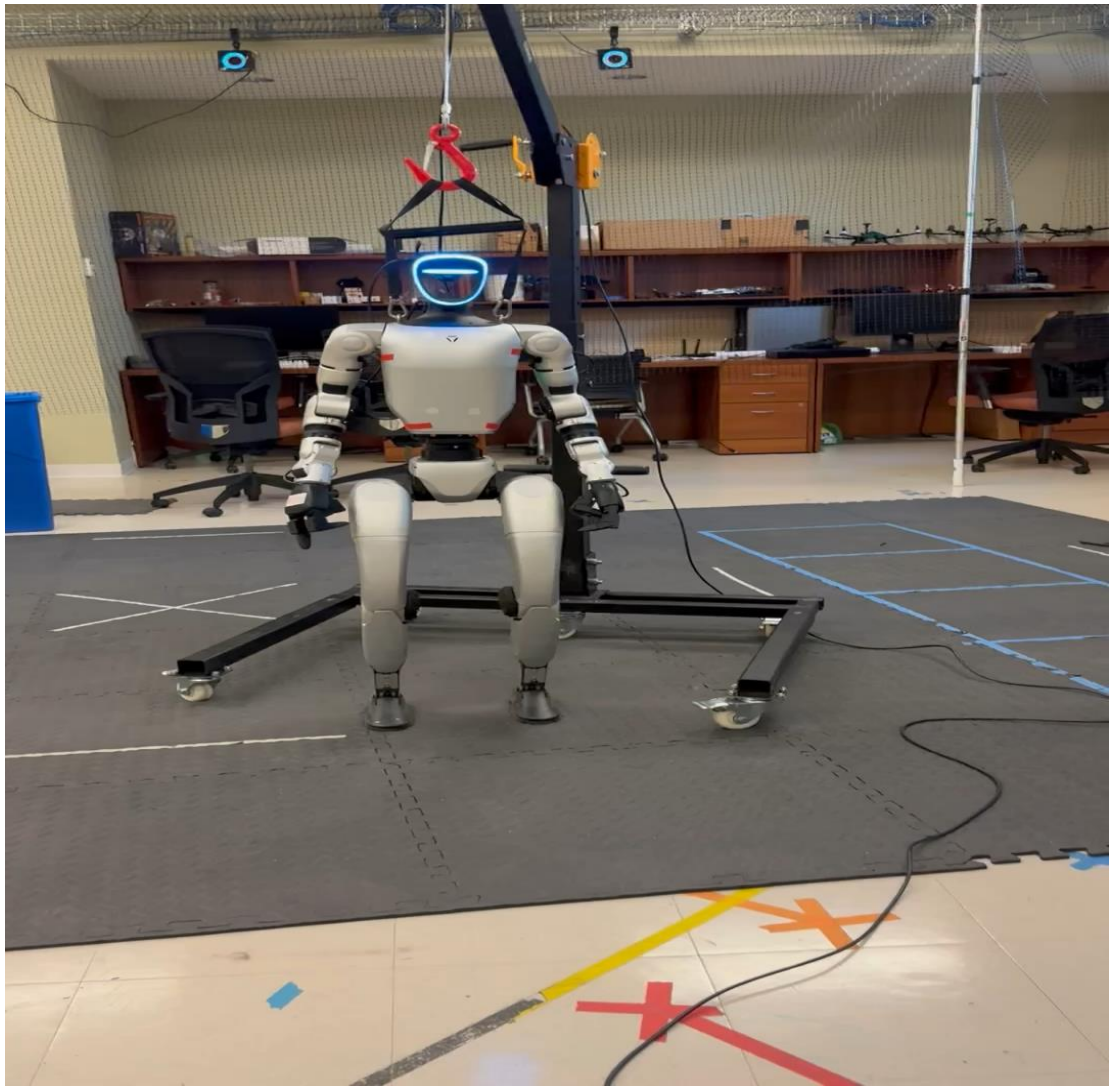
WashU

Acknowledgments

Andrew Clark	**Yiannis Kantaros**	**ShiNung Ching**
Jash Narwani	**Michal Zemanek**	**HongChao Zhang**
Dzenan Zecevic	**Mario Rodriguez**	**Ziyi Peng**

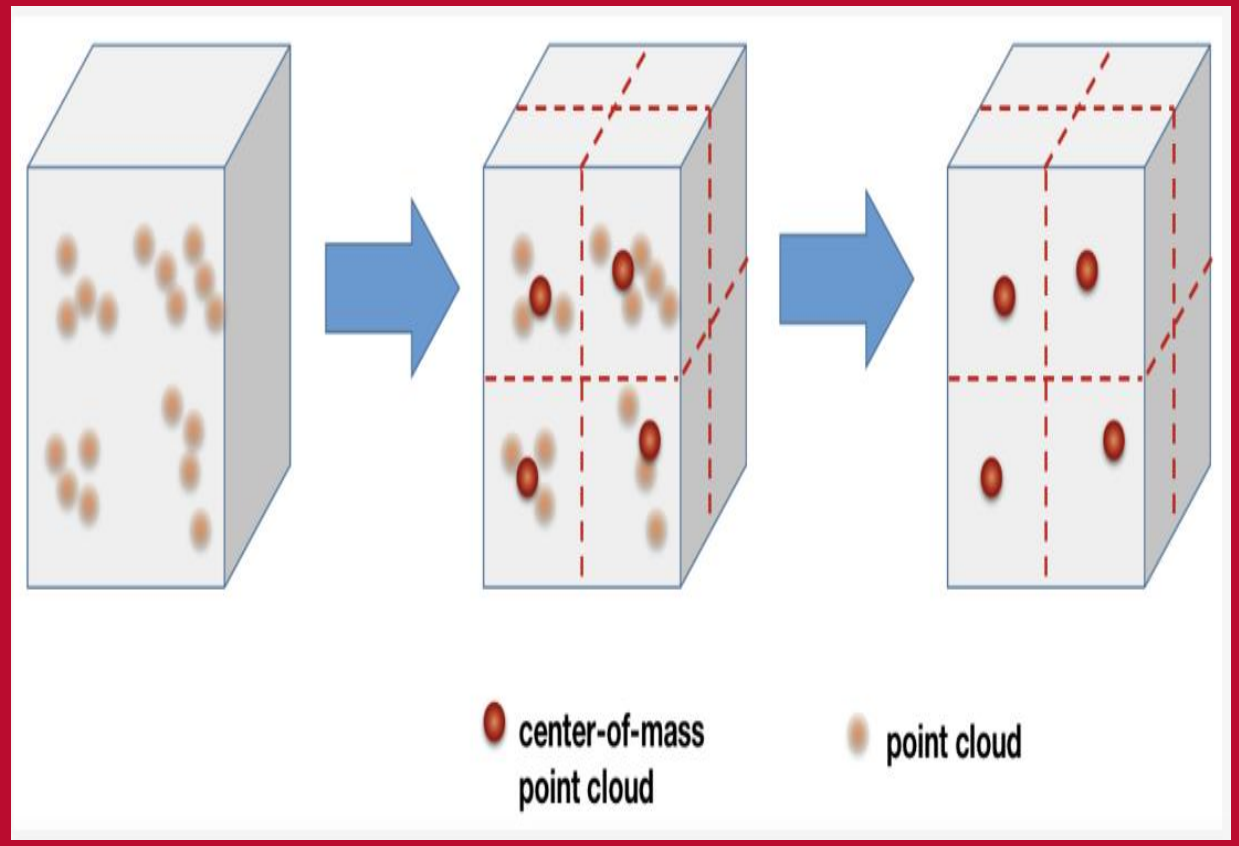
WashU

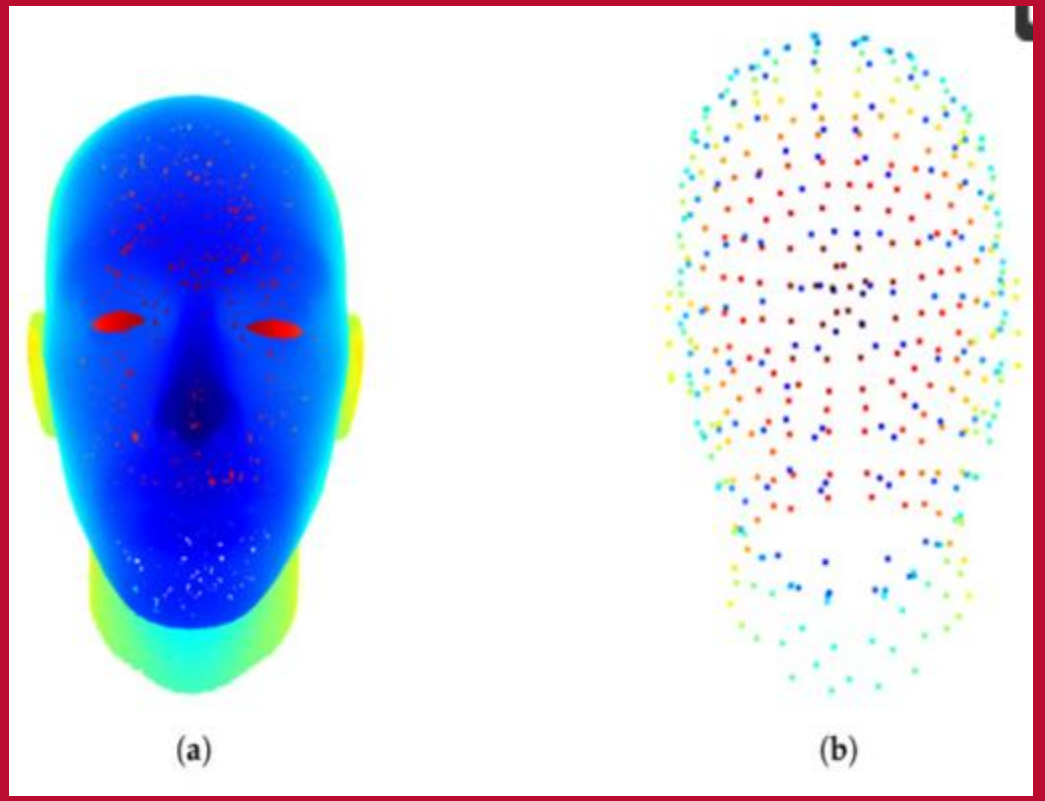
Thank You!



WashU

Voxel Filter





CBF Constraint Single Integrator Dynamics

$$2(\mathrm{p}_x - x_{\mathrm{obs}})u_x + 2(p_y - y_{\mathrm{obs}})u_y \geq 0$$