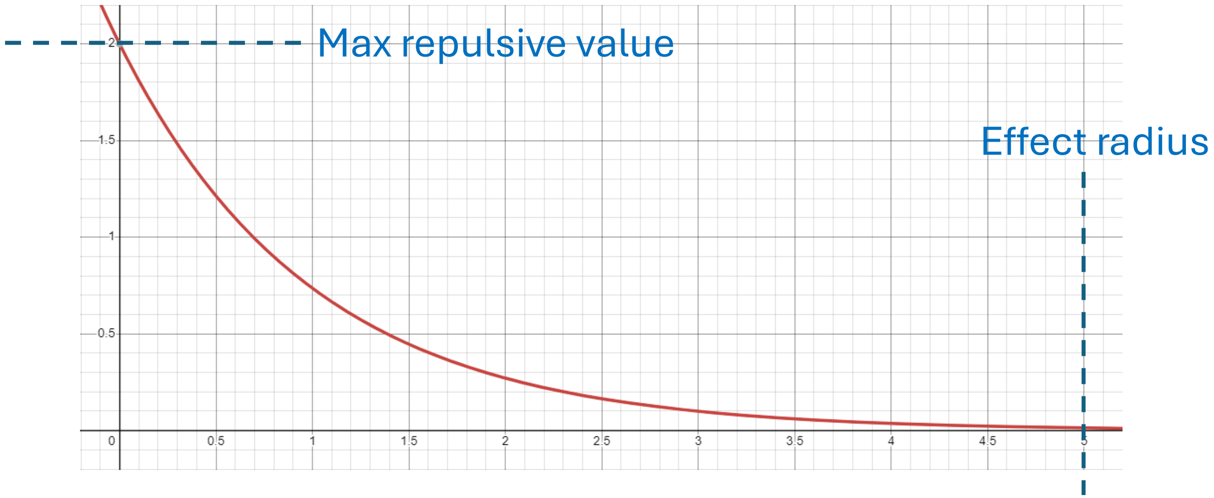
Vector field controller: Formulas implemented are the following

V(p) is a 2D potential created so that repulsive points (p_{ri}) are hills and the attractive point (p_a) is a hole.
The robot follows $-grad(V)$ to converge to the attractive point.

$$V(p) = \|p - p_a\|^{ka} + \sum_i \frac{kr_{heigth}}{kr_{slope}} \exp(kr_{slope}(kr_{dist} - \|p - p_{ri}\|))$$

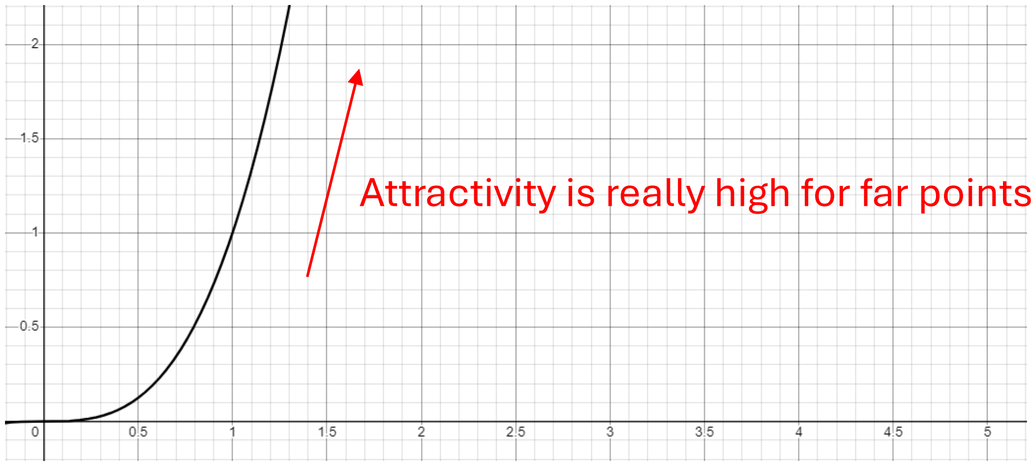
$$-grad(V) = -ka(p - p_a)\|p - p_a\|^{ka-1} + \sum_i \frac{p - p_{ri}}{\|p - p_{ri}\|} kr_{heigth} \exp(kr_{slope}(kr_{dist} - \|p - p_{ri}\|))$$

Repulsive potential shape, customizable with: $kr_{dist}, kr_{heigth}, kr_{slope}$



$x = \|p - p_r\|$ $\quad f(x) = 2 \cdot \exp(1 \cdot (0 - x))$

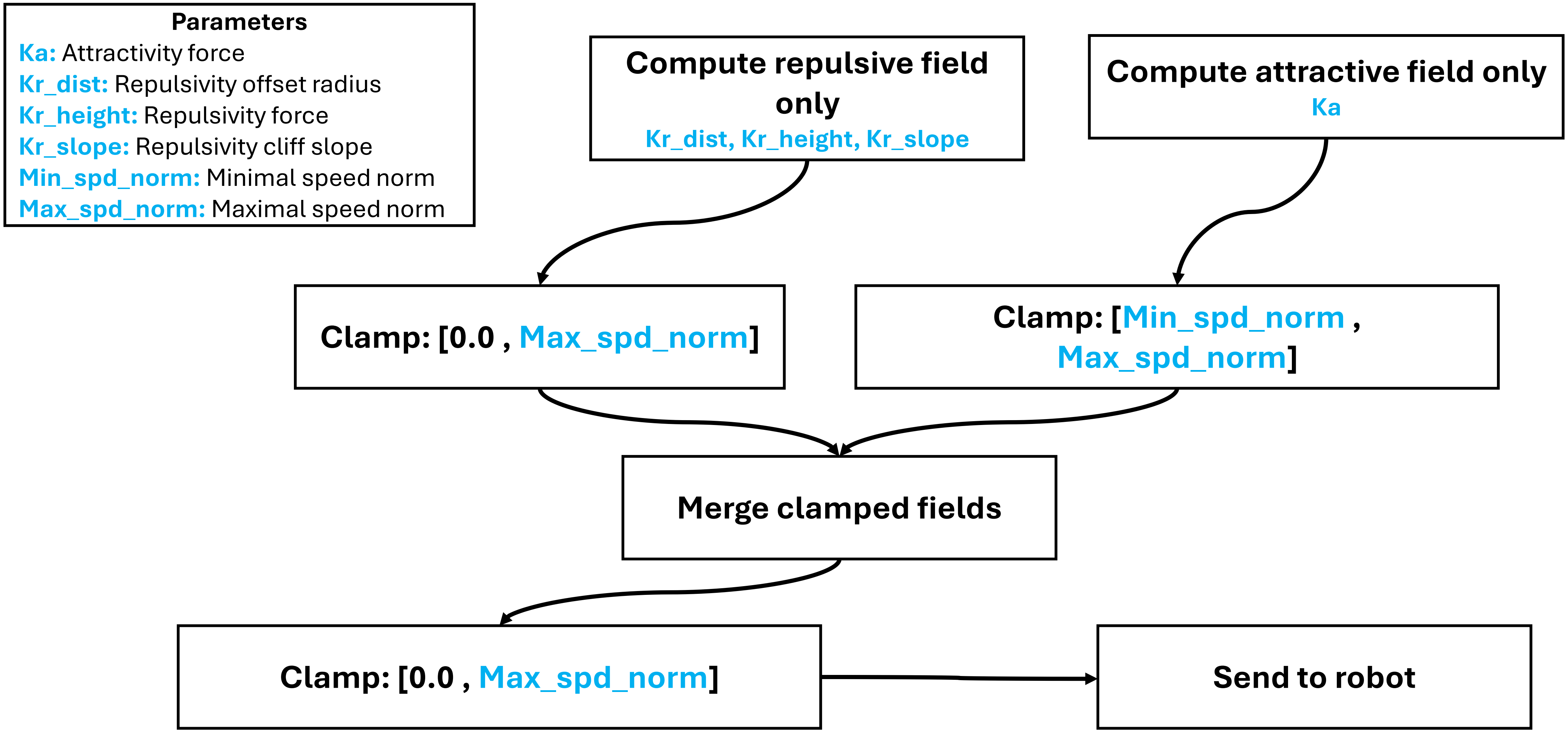
Attractive potential shape, customizable with: ka



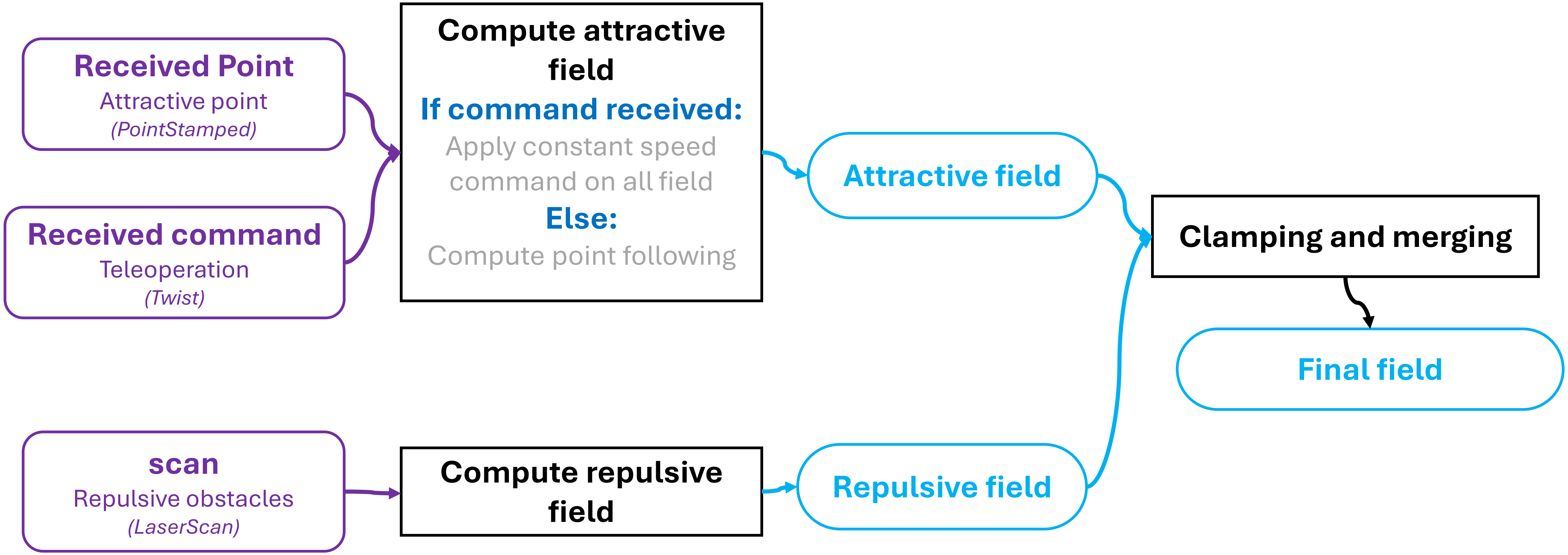
$x = \|p - p_r\|$ $\quad f(x) = x^3$

We need to clamp those functions to avoid unwanted very high gradient values

Speeds Clamping strategy

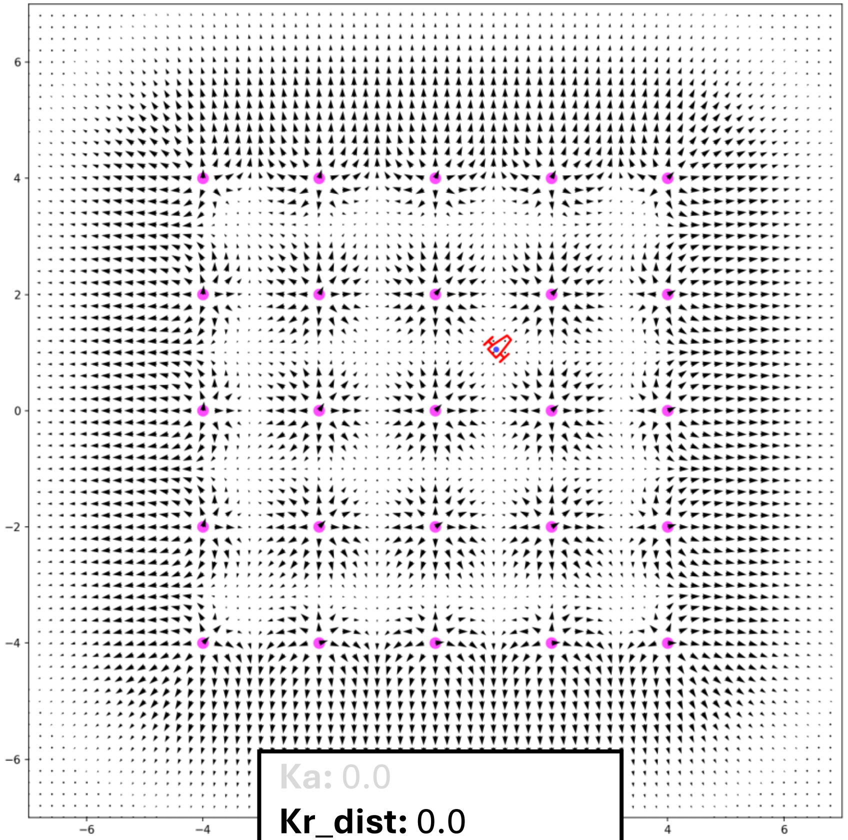


Vector field controller Implementation



The package directly manage point following and speed commands (teleoperation) features

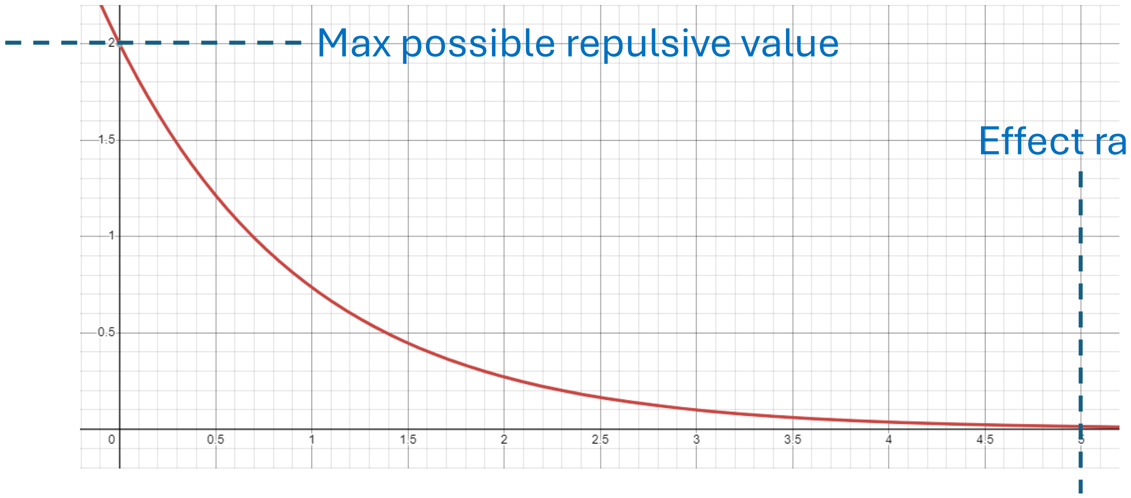
Visualization (Python program): Repulsive only



Ka: 0.0
Kr_dist: 0.0
Kr_height: 2.0
Kr_slope: 1.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

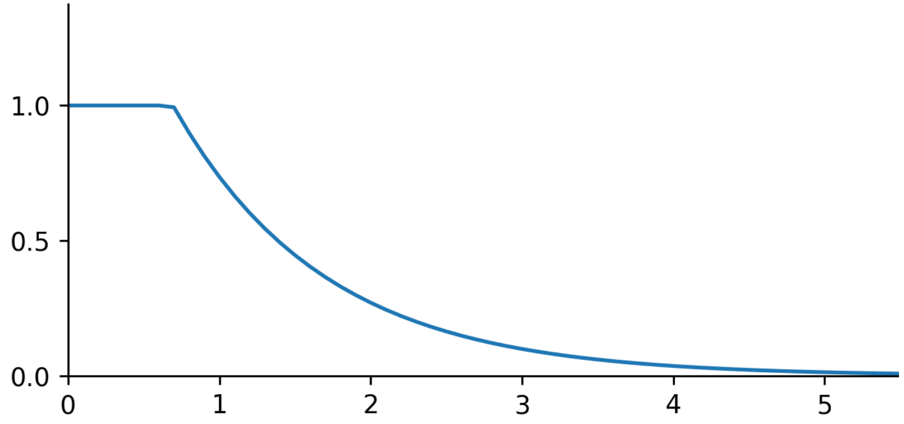
Unclamped Repulsive potential

$$f(x) = 2 \cdot \exp(1 \cdot (0 - x))$$

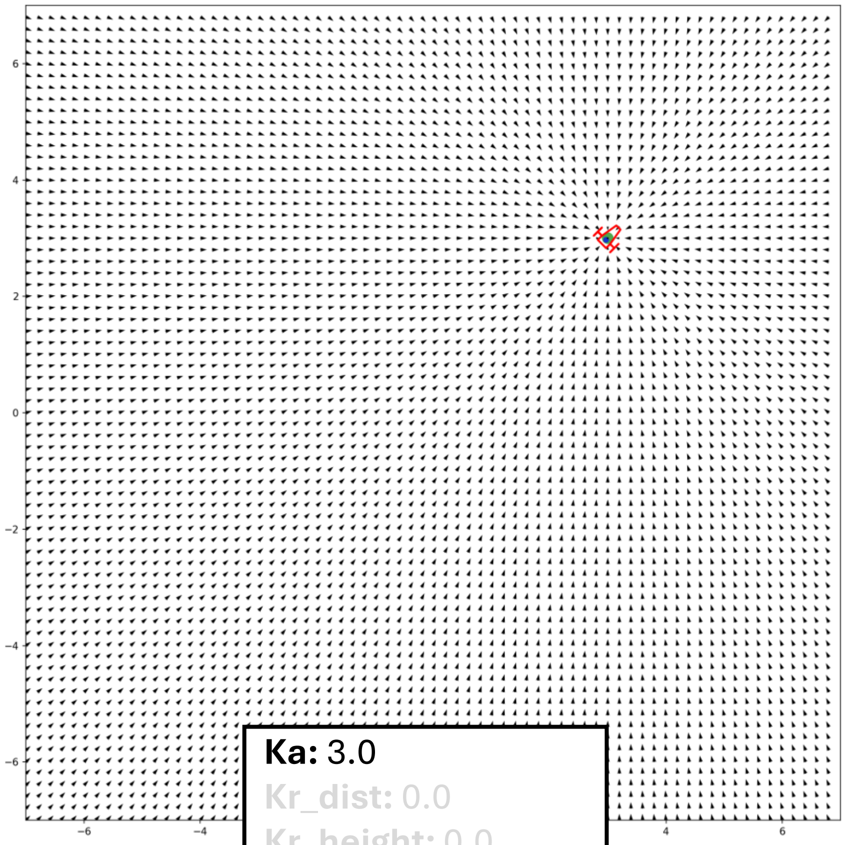


$x = \|p - p_r\|$

Clamped Repulsive potential



Visualization (Python program): Attractive only



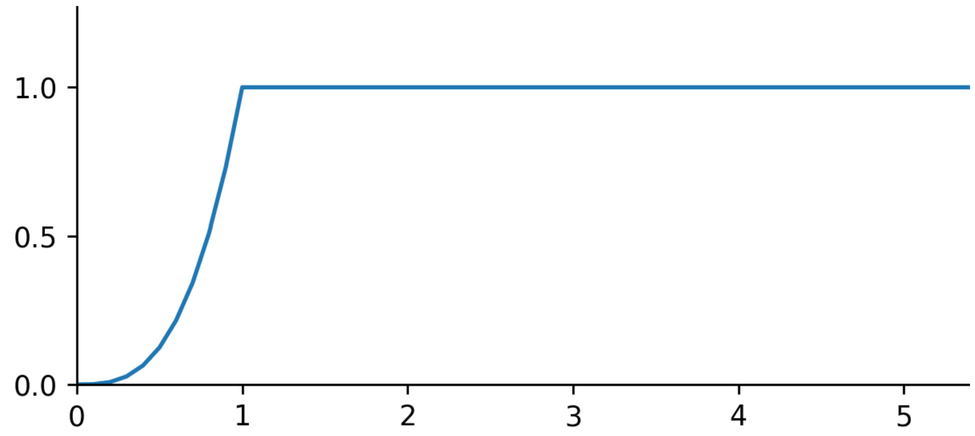
Ka: 3.0
Kr_dist: 0.0
Kr_height: 0.0
Kr_slope: 1.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

Unclamped merged potentials

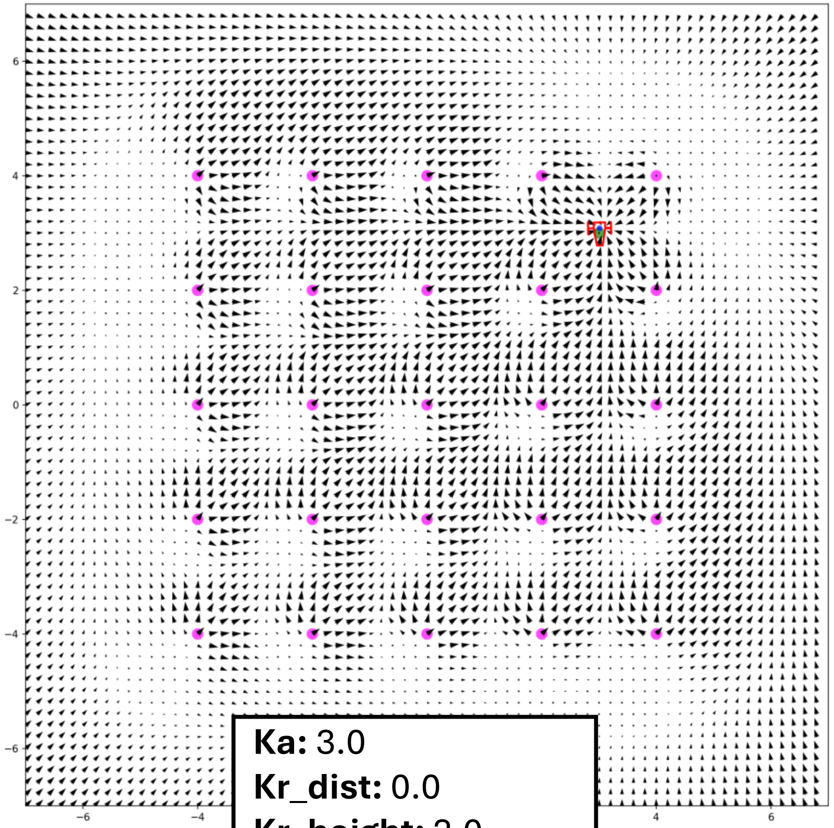
$$f(x) = x^3$$

Clamped merged potential

$$x = \|p - p_r\|$$



Visualization (Python program): Attractive + repulsive

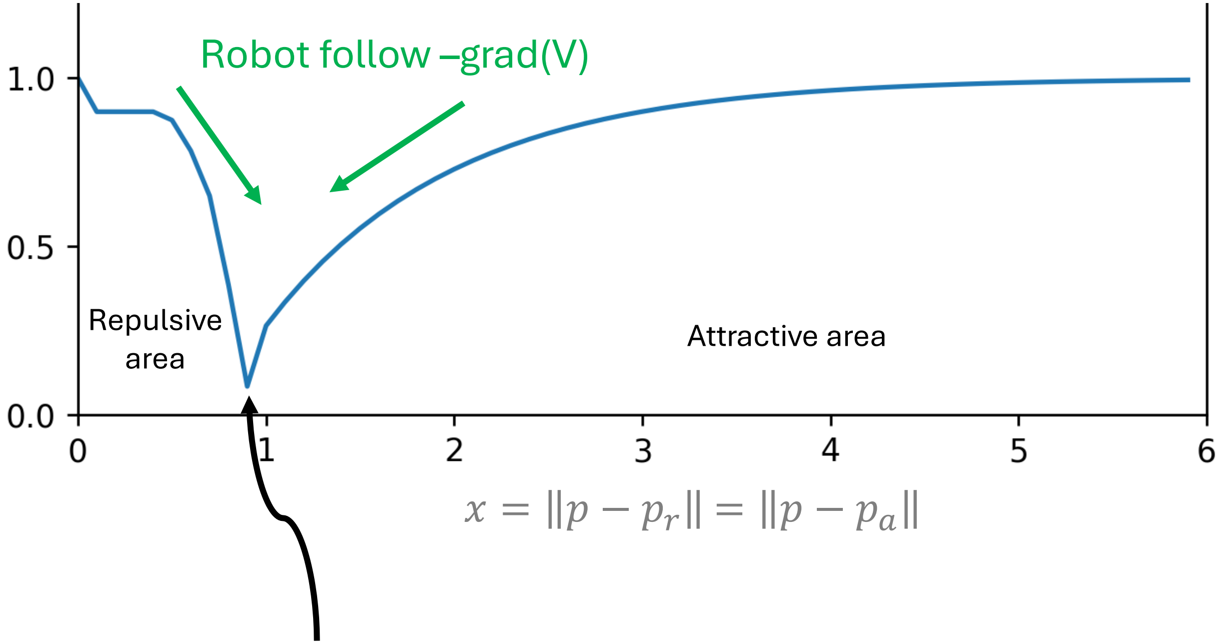


Ka: 3.0
Kr_dist: 0.0
Kr_height: 2.0
Kr_slope: 1.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

Example of potential for coincident attractive and repulsive point

Visuals of how would be the potential V around a point $p_r = p_a$

$$f(x) = abs(clamped(2 \cdot \exp(1 \cdot (0 - x))) - clamped(x^3))$$

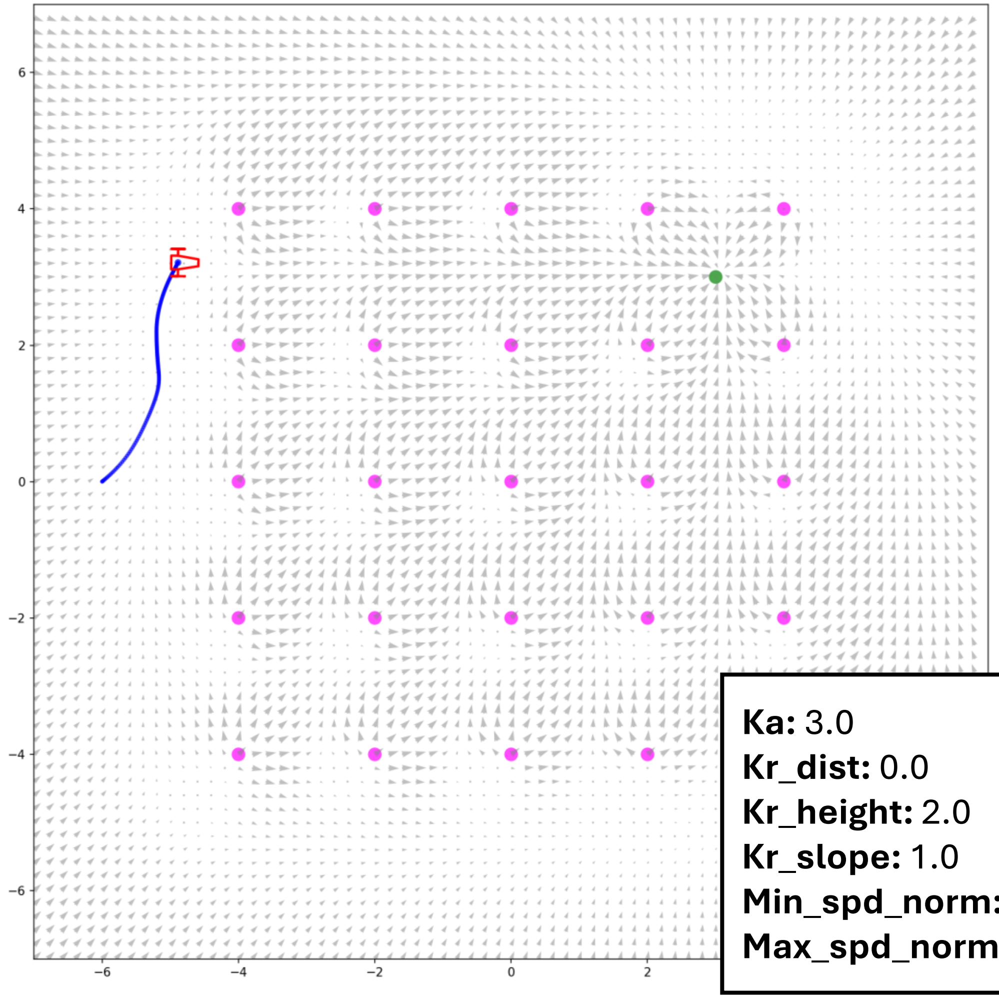


In this example the robot would converge to a position almost 1m away from the obstacle p_r

Simulation (Python program): Attractive + repulsive

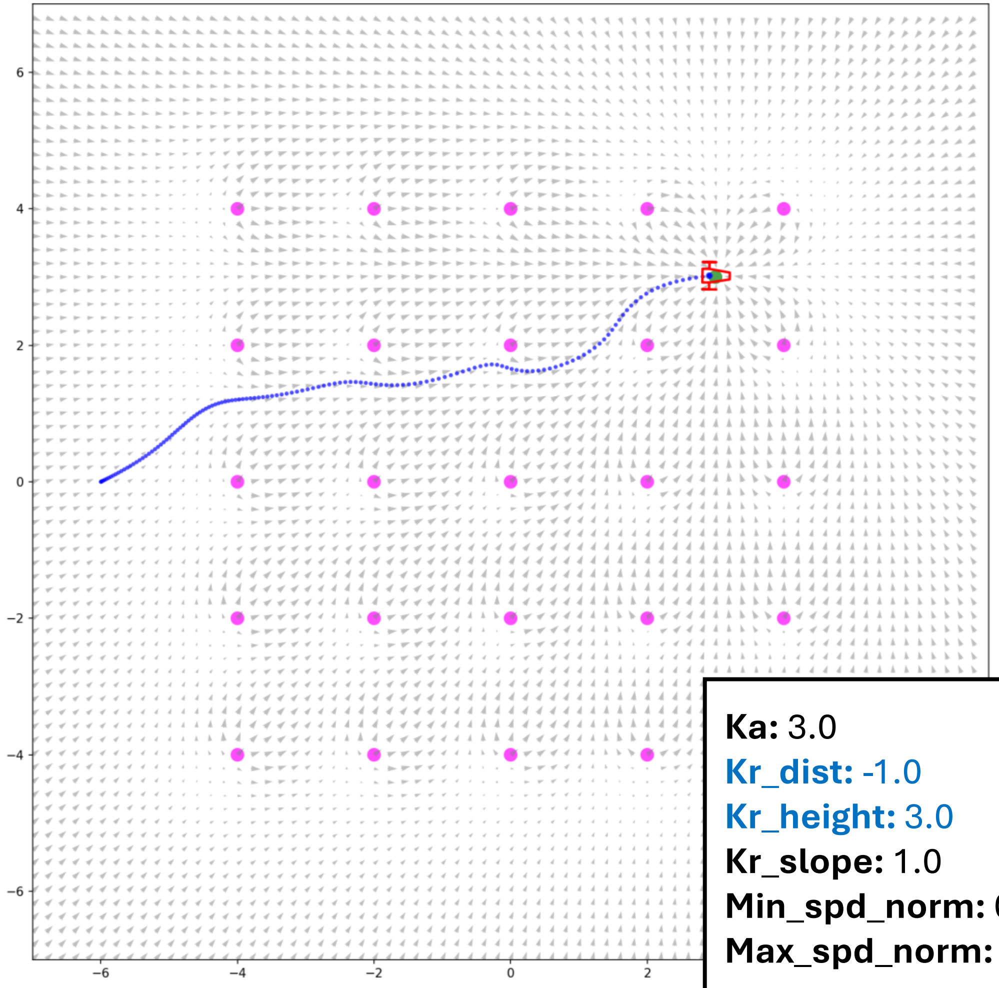
We can customize the navigation behavior by changing the parameters

Can not go through



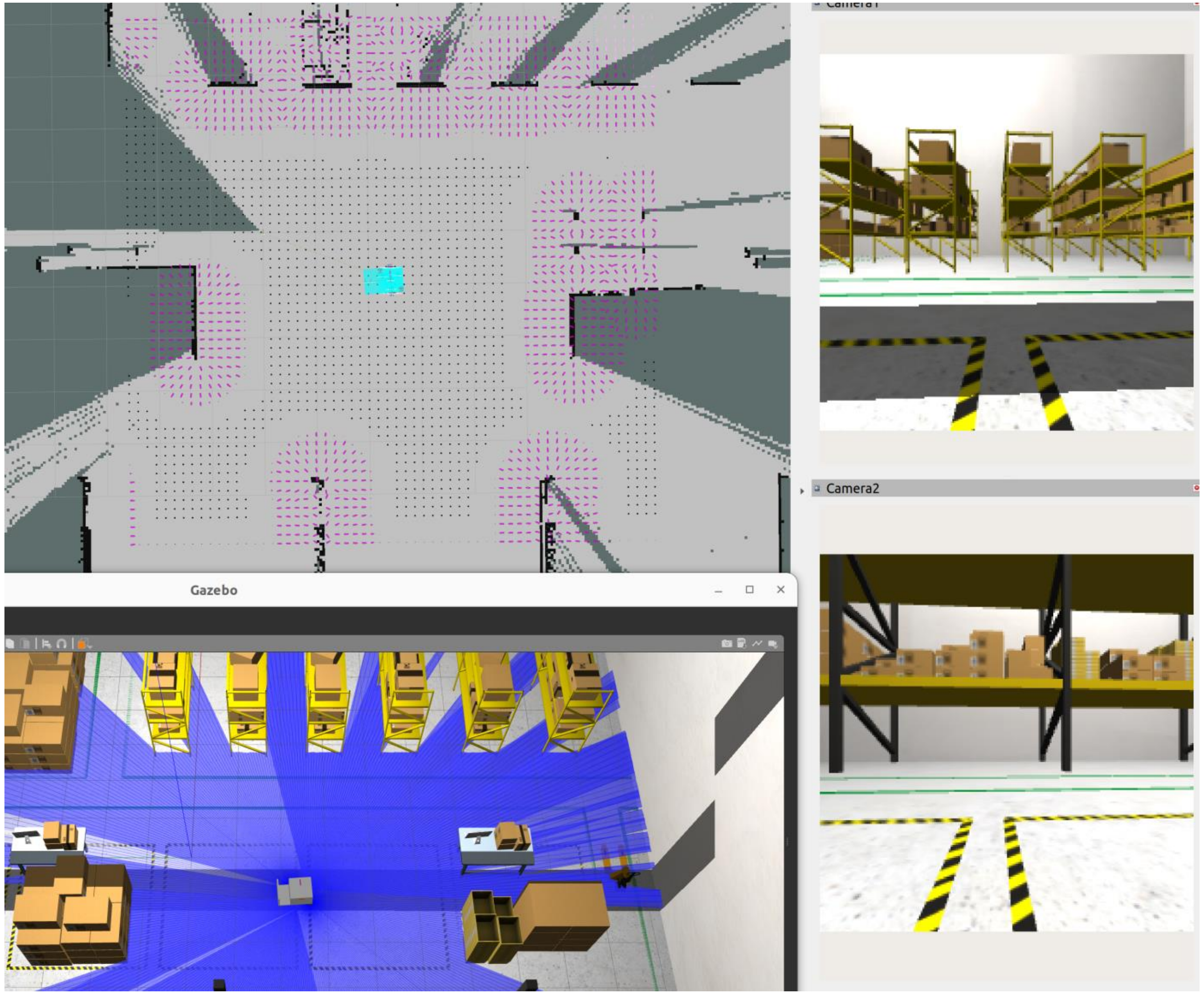
Ka: 3.0
Kr_dist: 0.0
Kr_height: 2.0
Kr_slope: 1.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

Can go through



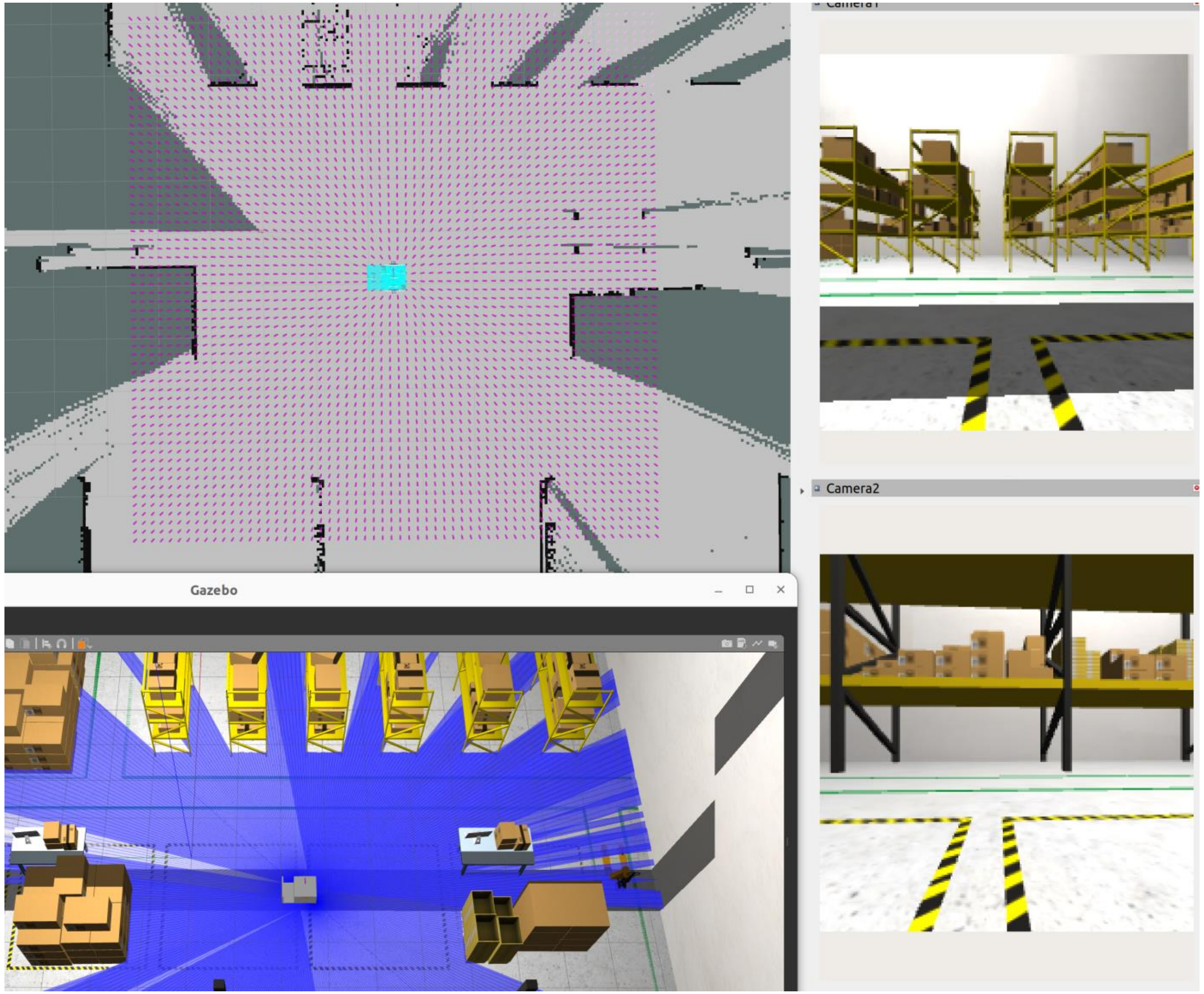
Ka: 3.0
Kr_dist: -1.0
Kr_height: 3.0
Kr_slope: 1.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

Visualization with ROS2 and Gazebo (simulated robot): Repulsive only



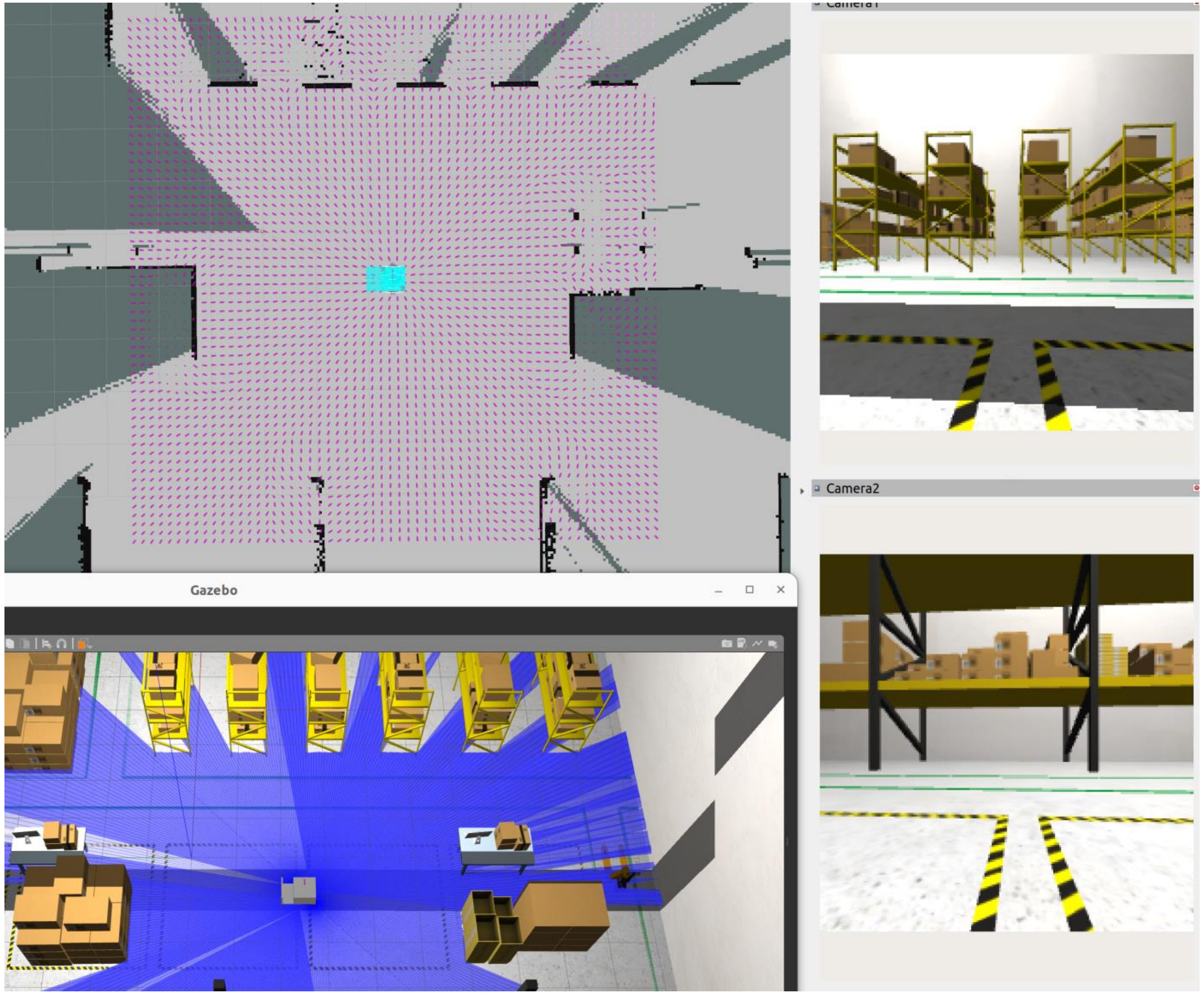
Ka: 0.0
Kr_dist: 0.8
Kr_height: 1.0
Kr_slope: 30.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

Visualization with ROS2 and Gazebo (simulated robot): Attractive only



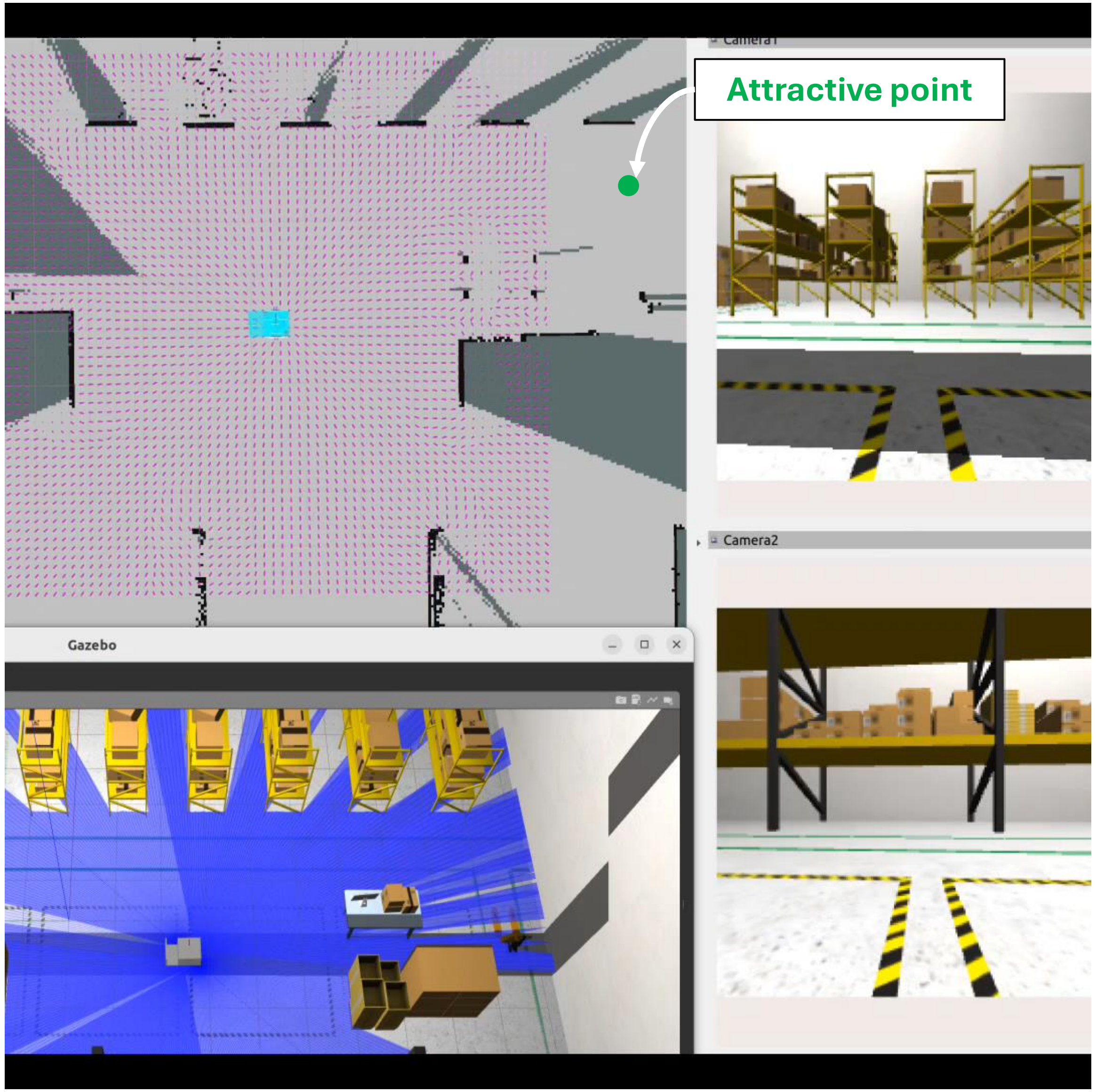
Ka: 3.0
Kr_dist: 0.8
Kr_height: 0.0
Kr_slope: 30.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

Visualization with ROS2 and Gazebo (simulated robot): Attractive + Repulsive



Ka: 3.0
Kr_dist: 0.8
Kr_height: 1.0
Kr_slope: 30.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

Simulation with ROS2 and Gazebo (simulated robot): Attractive + Repulsive



Ka: 3.0
Kr_dist: 0.8
Kr_height: 1.0
Kr_slope: 30.0
Min_spd_norm: 0.1
Max_spd_norm: 1.0

(Open doc.pptx to play the video)