# Adapt: Text-Promptable Perception and Diffusion-Predicted Avoidance with MPPI Control on the Polaris GEM e4

Polaris GEM e4 Autonomy Team
*CS 588 – Group 10*
*University of Illinois Urbana–Champaign*
Urbana, IL, USA

***Abstract*—We present Adapt, an autonomous driving stack on the Polaris GEM e4 that couples a learning-based pedestrian trajectory predictor with a sampling-based motion planner. The default Stanley lateral controller from the AutoShield baseline is replaced with a Model Predictive Path Integral (MPPI) planner that optimizes jointly over steering and acceleration. The planner consumes multi-step forecasts from a diffusion trajectory predictor in single-agent and joint multi-agent configurations. The system is implemented in ROS 2 Humble and runs end-to-end on the GEM e4, using its Ouster OS1-128 LiDAR, OAK-D LR RGBD camera, Septentrio GNSS/INS, and PACMod2 drive-by-wire interface. A complementary text-promptable perception module supplies open-vocabulary object goals for non-pedestrian targets. Together, these components enable a controlled comparison between a classical baseline and modern generative motion forecasts on the same vehicle, routes, and safety state machine.**

***Index Terms*—autonomous driving, MPPI, sampling-based MPC, pedestrian prediction, diffusion models, open-vocabulary perception**

## I. Introduction

Pedestrian-aware urban driving requires a tight coupling between predictive perception and reactive control. Classical lateral controllers such as Stanley and pure pursuit track a reference path well, but they cannot modulate longitudinal motion in response to predicted pedestrian behavior beyond a hard stop. Sampling-based model predictive control, and in particular Model Predictive Path Integral (MPPI) control, closes this gap by optimizing over the joint steering–acceleration space and incorporating arbitrary, non-differentiable cost terms, including trajectory-conditioned obstacle costs.

This work develops an end-to-end stack on the Polaris GEM e4 that exposes three prediction modes (constant-velocity, single-agent diffusion, and joint multi-agent diffusion) and two controllers (MPPI and Stanley) under a unified interface. A text-promptable perception module supplies goal poses for arbitrary prompted targets and supports simulation-based development. This work contributes

- A GPU-batched MPPI planner integrated with PACMod2, sharing a safety state machine with the Stanley baseline.
- Single-agent and joint diffusion pedestrian predictors deployed as ROS 2 nodes with sticky-mode trajectory selection.
- An open-vocabulary detection-and-LiDAR-fusion goal module that operates in both Gazebo and on the vehicle.
- An on-vehicle A/B comparison protocol over identical routes.

## II. Preliminaries

### A. Sampling-Based Model Predictive Control

Sampling-based model predictive control reformulates receding-horizon trajectory optimization as a Monte Carlo estimation problem. In place of a constrained nonlinear program solved at each control tick, the planner draws a batch of randomly perturbed control sequences, scores each by forward-simulating the vehicle dynamics, and returns a cost-weighted average. The approach is gradient-free, which permits arbitrary non-differentiable cost terms such as discrete safety overlays, hard clearance constraints, and outputs of learned trajectory predictors.

The planar planning state is $\boldsymbol{x}_t = [x_t, y_t, \psi_t, v_t]^\top \in \mathbb{R}^4$, representing the position, yaw, and longitudinal speed of the vehicle in an inertial frame, and the control input is $\boldsymbol{u}_t = [a_t, \delta_t]^\top$ with longitudinal acceleration $a_t$ and front-wheel steering angle $\delta_t$. Both inputs are box-constrained to actuator limits, $a_t \in [a_{\min}, a_{\max}]$ and $\delta_t \in [-\delta_{\max}, \delta_{\max}]$. The dynamics are an explicit Euler discretization of the rear-axle kinematic bicycle with wheelbase $L$ and step size $\Delta t$,

$$\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t) = \begin{bmatrix} x_t + v_t \cos\psi_t \, \Delta t \\ y_t + v_t \sin\psi_t \, \Delta t \\ \psi_t + (v_t/L) \tan\delta_t \, \Delta t \\ \max(0,\ v_t + a_t \, \Delta t) \end{bmatrix}. \tag{1}$$

Clipping the speed at zero enforces a forward-only motion model consistent with the operating regime of the platform.

Model Predictive Path Integral (MPPI) control instantiates the sampling-based framework with Gaussian exploration noise and a soft-min update rule. Let $H$ denote the planning horizon and $K$ the number of rollouts. The planner maintains a nominal control sequence $\bar{\boldsymbol{U}} = (\bar{\boldsymbol{u}}_0, \ldots, \bar{\boldsymbol{u}}_{H-1})$ and, at each cycle, samples $K$ independent perturbations $\boldsymbol{\epsilon}_t^{(k)} \sim \mathcal{N}(\boldsymbol{0}, \Sigma_u)$ with $\Sigma_u = \mathrm{diag}(\sigma_a^2, \sigma_\delta^2)$. The candidate control sequences $\boldsymbol{u}_t^{(k)}$ are obtained by adding the noise to $\bar{\boldsymbol{u}}_t$ and projecting the result onto the actuator box, and the corresponding state trajectories are propagated by $\boldsymbol{x}_{t+1}^{(k)} = f(\boldsymbol{x}_t^{(k)}, \boldsymbol{u}_t^{(k)})$ from the current ego state. The $K$ rollouts are mutually independent,

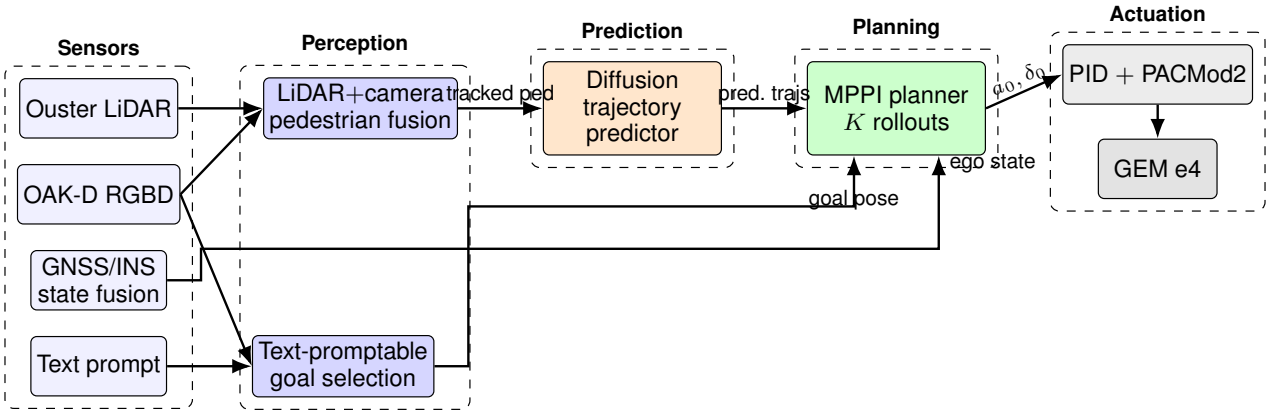


Fig. 1: High-level Adapt pipeline. Two perception paths share the LiDAR cloud and the camera image. The pedestrian-fusion path emits tracked detections to the diffusion trajectory predictor, while the text-promptable path emits a single navigation goal from a runtime prompt. The MPPI planner consumes the predicted trajectories, the goal pose, and the ego state estimate produced by the GNSS/INS sensor-fusion node, and forwards its first action to the downstream PID and PACMod2 stack that drives the Polaris GEM e4.

which admits an efficient GPU-batched implementation in the realized system (Section VI).

Each rollout is assigned a scalar cost combining a running cost $c(\cdot)$ accumulated along the horizon and a terminal cost $\phi(\cdot)$ evaluated at the final state,

$$S^{(k)} = \sum_{t=0}^{H-1} c\big(\boldsymbol{x}_t^{(k)}, \boldsymbol{u}_t^{(k)}, t\big) + \phi\big(\boldsymbol{x}_H^{(k)}\big). \tag{2}$$

The functional forms of $c$ and $\phi$, including the path-tracking, stability, and obstacle terms employed in this work, are specified in Section VI.

The MPPI update converts the sample costs into normalized importance weights via a softmax with temperature $\lambda$, where the offset $\beta = \min_k S^{(k)}$ is subtracted for numerical stability,

$$w^{(k)} = \frac{\exp\big(-\frac{1}{\lambda}(S^{(k)} - \beta)\big)}{\sum_{j=1}^{K} \exp\big(-\frac{1}{\lambda}(S^{(j)} - \beta)\big)}. \tag{3}$$

Smaller values of $\lambda$ concentrate the weights on the lowest-cost rollouts, while larger values produce a near-uniform average. The nominal sequence is updated by the weighted mean of the perturbations,

$$\bar{\boldsymbol{u}}_t \leftarrow \bar{\boldsymbol{u}}_t + \sum_{k=1}^{K} w^{(k)} \boldsymbol{\epsilon}_t^{(k)}, \qquad t = 0, \ldots, H-1. \tag{4}$$

Only the first element $\bar{\boldsymbol{u}}_0 = (a_0, \delta_0)$ is applied to the vehicle at each tick. The remainder of $\bar{\boldsymbol{U}}$ is shifted forward and reused as a warm start for the subsequent iteration. The applied acceleration $a_0$ is integrated into a velocity setpoint tracked by a downstream PID loop operating at rate $f_{\mathrm{ctrl}}$, $v_{\mathrm{cmd}} \leftarrow \mathrm{clip}\big(v_{\mathrm{cmd}} + a_0/f_{\mathrm{ctrl}},\, 0,\, v_{\mathrm{ref}}\big)$, which interfaces the planner output with the PACMod2 longitudinal channel.

The effective sample size $\mathrm{ESS} = 1/\sum_k (w^{(k)})^2$ provides a runtime measure of weight concentration and is monitored as a diagnostic for noise-scale tuning.

## III. System Architecture

The runtime stack is organized in four layers covering perception, fusion, prediction, and control. Figure 1 summarizes the dataflow at a high level.

### A. Perception

A LiDAR pipeline applies density-based clustering to the Ouster point cloud with human-geometry filters on height and width, producing pedestrian detections in polar form. In parallel, an RGBD pipeline applies YOLOv11 to the OAK-D RGB stream and back-projects the bounding-box centroid using the stereo depth image. A second perception path, described in Section V, performs text-conditioned 2D detection fused with the same LiDAR cloud to generate goal poses for arbitrary prompted targets.

### B. Fusion

An approximate time-synchronized association between LiDAR and camera detections yields a single fused pedestrian estimate. Range is weighted toward the LiDAR for its superior depth accuracy, while bearing is weighted toward the camera for its higher angular resolution.

### C. Prediction

A single inference node consumes the fused pedestrian track and publishes future trajectories together with motion and time-to-collision estimates. The active prediction model is selected at launch, allowing downstream nodes to remain mode-agnostic.

### D. Control

The MPPI node subscribes to either the fused pedestrian estimate (treated as a static obstacle) or the predicted trajectories with per-step covariance, and issues PACMod2 commands. A high-level state machine emits CRUISE, SLOW_CAUTION,

STOP_YIELD, and CREEP_PASS states based on time-to-collision and pedestrian motion, providing a coarse safety overlay that is independent of the chosen predictor.

## IV. Diffusion Trajectory Predictor

The trajectory predictor is a joint multi-agent denoiser (JointTrajectoryDenoiser) that adapts the Motion Indeterminacy Diffusion (MID) formulation of Gu et al. [1] to a ROS-deployable, scene-level conditional model. All pedestrians in a scene are denoised jointly so that each trajectory is conditioned on the presence and history of the others through cross-agent attention. Training follows DDPM [2] on the noise-prediction objective, and inference uses 10-step DDIM [3] sampling under a cosine $\beta$ schedule [4]. Two design choices follow MID directly. The network predicts the noise added during the forward diffusion rather than the future trajectory itself, and the decoder queries are constructed from the noised future trajectory rather than from learned query tokens, so the conditioning between the encoded history and the candidate future is exercised at every denoising step.

### A. Architecture

Each agent history $\boldsymbol{h}^{(m)} \in \mathbb{R}^{20\times 4}$, consisting of five seconds of $(x, y, v_x, v_y)$ at $\Delta t_{\text{ped}} = 0.25\,\text{s}$, is encoded by a six-layer Transformer with shared weights, positional embeddings, and joint conditioning on the diffusion timestep and the ego velocity. The per-agent embeddings are mean-pooled, refined by a three-layer cross-agent encoder, and added back to the per-agent representations to produce interaction-aware memory tokens. The decoder embeds the noisy future trajectory through a $\mathbb{R}^2 \to \mathbb{R}^{256}$ projection, applies positional embeddings, and runs four Transformer-decoder layers that cross-attend to the enriched history. A small MLP head predicts the per-step noise $\hat{\boldsymbol{\epsilon}} \in \mathbb{R}^{20\times 2}$. The model uses $d = 256$, nhead $= 8$, FFN dimension $512$, and dropout $0.1$, supports up to $M = 16$ agents per scene, and totals $8.18\,\text{M}$ parameters. The deployed configuration is summarized in Table I.

### B. Training

The model is trained on the pedestrian split of Argoverse 2 Motion Forecasting [5], with all tracks of object type PEDESTRIAN resampled to $4\,\text{Hz}$ and re-expressed in pedestrian-centric coordinates anchored at the last observed position. Each sample consists of an agent history tensor of shape $(M, 20, 4)$, a history mask, a ground-truth future of shape $(M, 20, 2)$, a per-agent validity mask, and the ego velocity. The training split contains $49{,}202$ scenes and the validation split contains $6{,}100$ scenes, each padded to $M = 16$. Three augmentations are applied during training, namely a per-scene random rotation in $\pm 15^\circ$, per-agent random dropout of 0–$20\%$ of valid history steps, and a per-scene translation jitter of $\pm 0.1\,\text{m}$.

After Argoverse-2 pre-training, two auxiliary corpora are used for domain-adaptation fine-tuning. The first is the standard ETH/UCY pedestrian benchmark, whose four canonical scenes are read from the native $2.5\,\text{Hz}$ annotations, linearly interpolated to the model's $4\,\text{Hz}$ rate, and re-windowed into the same 20-step history and 20-step future format with the same pedestrian-centric normalization; because ETH/UCY contains no ego vehicle, the ego-velocity conditioning channel is set to zero on those samples. The second is a procedurally generated synthetic corpus of pedestrian arc trajectories that densifies the long-tail of tight-turn motions, which are under-represented in vehicle-centric urban data. Each synthetic sample is drawn from one of five parametric primitives, namely constant-radius arcs, S-bends, spirals, U-turns, and constant-curvature accelerating arcs, with the radius, the linear speed, and the turn direction sampled independently, followed by a random rotation, a random translation, and small additive position noise. Both corpora are emitted in the same tensor format as the Argoverse-2 pre-processing pipeline, so the encoder, the cross-agent interaction layers, and the loss are all reused unchanged. To improve robustness on the synthetic corpus, an additional observation-noise augmentation is enabled during training, separate from the three always-on augmentations defined above. It injects zero-mean Gaussian perturbations into the history, with standard deviation $\sigma_{\text{pos}} = 0.05\,\text{m}$ on each valid position and a matching velocity perturbation $\sigma_{\text{vel}} = \sigma_{\text{pos}}/\Delta t_{\text{ped}} = 0.2\,\text{m/s}$ on the corresponding velocity channel. The noise is sampled independently per timestep and per real agent, and padded agent slots are left unchanged. Fine-tuning starts from the Argoverse-2 checkpoint and uses a reduced learning rate, leaving the architecture and the diffusion schedule of Table I untouched.

The forward diffusion follows the standard DDPM objective, with a masked mean-squared error between predicted and

TABLE I: JointTrajectoryDenoiser architecture, diffusion process, and training settings.

| Component | Value |
|---|---|
| $d_{\text{model}}$, heads, FFN | 256, 8, 512 |
| Per-agent encoder layers | 6 |
| Cross-agent interaction layers | 3 |
| Decoder layers | 4 |
| Dropout | 0.1 |
| Max agents per scene | 16 |
| Total parameters | 8.18 M |
| Diffusion timesteps (DDPM) | 100 |
| $\beta$ schedule | cosine ($s = 0.008$) |
| DDIM steps (inference) | 10 |
| DDIM $\tau$ schedule | $[99, 88, 77, \ldots, 11, 0]$ |
| Samples per scene $K$ | 20 |
| Optimizer | AdamW, weight decay $10^{-4}$ |
| Learning rate | $2\times10^{-4} \to 10^{-5}$ (cosine) |
| Epochs | 200 |
| Batch size | 16 |
| Gradient clipping | 1.0 |
| EMA decay | 0.999 |
| Mixed precision | AMP fp16 |
| Random seed | 1729 |
| Validation frequency | every 5 epochs |
| Checkpoint selection | best val. minFDE-20 |

true noise computed over real agents only,

$$\mathcal{L}_{\mathrm{DDPM}} \;=\; \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}}\big[\,\big\|\boldsymbol{\epsilon}-\hat{\boldsymbol{\epsilon}}_\theta(\boldsymbol{x}_t,t,\boldsymbol{c})\big\|_2^2\,\big], \tag{5}$$

where $\boldsymbol{c}$ aggregates the encoded history and ego conditioning. Optimization uses AdamW with weight decay $10^{-4}$, a cosine learning rate schedule from $2{\times}10^{-4}$ to $10^{-5}$, batch size 16, gradient clipping at 1.0, an exponential moving average with decay 0.999, and mixed-precision (AMP fp16) updates with a gradient scaler. Validation is performed every five epochs over a 200-epoch budget, and the checkpoint with the lowest validation minFDE-20 is retained.

### C. Evaluation and Deployment

Performance is reported on the AV2 validation split using three standard trajectory-prediction metrics computed over $K = 20$ DDIM samples per scene.

*a) Minimum average displacement error (minADE-K):* For each real agent the mean $\ell_2$ distance between the predicted and ground-truth position is computed over all twenty future time steps for each of the $K$ samples, the minimum across samples is retained, and the mean of these per-agent values is reported over the validation set.

*b) Minimum final displacement error (minFDE-K):* For each real agent the $\ell_2$ distance between the predicted and ground-truth position at the final time step of the five-second horizon is computed for each of the $K$ samples, the minimum across samples is retained, and the mean over all real agents is reported.

*c) Miss rate at 2 m:* The fraction of real agents for which the best-of-$K$ final displacement error exceeds two meters.

Table II collects the deployed checkpoint scores. The single forward pass at $M{=}8$ pedestrians and $K{=}20$ samples runs in approximately 11 ms on the on-vehicle RTX 3060, well within the 30 ms budget allocated to prediction.

TABLE II: Joint diffusion predictor on AV2 validation.

| Metric | Value |
|---|---|
| minADE-20 | 0.302 m |
| minFDE-20 | 0.532 m |
| miss rate at 2 m | 5.66% |
| Inference latency ($M{=}8$, $K{=}20$) | $\sim$ 11 ms |
| Latency budget | $<$ 30 ms |

At deployment the model runs as a ROS 2 node at 10 Hz, consuming fused pedestrian tracks and publishing predicted trajectories as a $M{\times}20{\times}2$ tensor that the MPPI planner indexes through $\tau(t)$ in (9). To suppress jitter under multi-modal predictions, a sticky closest-to-mean selector retains the current trajectory per pedestrian and switches only when a competing candidate consistently outperforms the incumbent across several cycles.

## V. Text-Promptable Perception

For non-pedestrian targets such as cones and signs, the system includes a text-conditioned camera and LiDAR fusion module that emits a single goal pose from a runtime-supplied prompt (e.g., *"red cone"* or *"pedestrian"*). Two interchangeable detection backends are provided. YOLO-World produces a bounding box that is used as a proxy mask, whereas LangSAM produces a dense binary mask via Grounding-DINO followed by SAM. On systems with Python $\geq$ 3.10, LangSAM invokes the SAM 2.1 Hiera-Small checkpoint through the unified `lang_sam` API, with detection thresholds box $= 0.25$ and text $= 0.20$. On earlier interpreters, the module falls back to the two-stage Grounding-DINO (SwinT-OGC) + SAM 1 (ViT-B) pipeline. The same Python core runs in both the Gazebo simulator and on the vehicle.

### A. Pipeline

The top detection by confidence is selected at every cycle. Let $\mathcal{M} \subset \Omega$ denote the binary mask region in image coordinates $\Omega$ and $\boldsymbol{u}^\star \in \Omega$ the pixel centroid of the bounding box (or of $\mathcal{M}$ when available). The mask is fused with the LiDAR cloud through the following stages, all of which operate on points expressed in the vehicle frame base_link.

*a) Projective frustum clip:* Each LiDAR point $\boldsymbol{p}_i^{\mathrm{lidar}}$ is transformed to the camera frame and projected to the image plane through the calibrated intrinsics $K$ (with the camera distortion model applied), $\boldsymbol{u}_i = \pi_K(T_{\mathrm{cam,lidar}}\boldsymbol{p}_i^{\mathrm{lidar}})$. Only points whose pixel projection rounds to a foreground pixel of the mask, $\mathcal{M}[\boldsymbol{u}_i] = 1$, are retained.

*b) Height filter and outlier rejection:* The surviving points are filtered to a vehicle-relative height band $z_{\min} \leq z \leq z_{\max}$ ($z_{\min} = 0.15\,\mathrm{m}$, $z_{\max} = 5.0\,\mathrm{m}$) that removes ground returns and overhead clutter. A statistical outlier filter then computes, for each point, the mean distance $\bar{d}_i$ to its $k = 8$ nearest neighbors and rejects points satisfying $\bar{d}_i > \bar{d} + \kappa\,\sigma_d$, with $\kappa = 2.0$ and $\bar{d}, \sigma_d$ the global mean and standard deviation of $\bar{d}_i$.

*c) Density clustering:* DBSCAN is applied to the remaining points with neighborhood radius $\varepsilon = 0.4\,\mathrm{m}$ and core-point threshold $n_{\min} = 3$. Clusters of fewer than three points are discarded.

*d) Cluster selection by reprojection:* For each surviving cluster $\mathcal{C}_j$ with centroid $\boldsymbol{\mu}_j$, the centroid is reprojected to the image, $\hat{\boldsymbol{u}}_j = \pi_K(T_{\mathrm{cam,base}}\boldsymbol{\mu}_j)$, and the cluster minimizing the 2D pixel distance to the detection target is selected,

$$\hat{\jmath} = \arg\min_j \big\|\hat{\boldsymbol{u}}_j - \boldsymbol{u}^\star\big\|_2. \tag{6}$$

This pixel-space coupling rather than the more naive 3D-closest rule prevents background returns inside the same frustum, such as a wall behind a cone, from displacing the intended goal.

*e) Ray-cast fallback:* When no cluster survives filtering, the system unprojects $\boldsymbol{u}^\star$ to a unit ray in the camera frame, scales it by a fixed distance $d_{\mathrm{est}} = 15\,\mathrm{m}$, and transforms the result to base_link. The resulting pose is marked as estimated and is replaced by a measured cluster once the vehicle closes the distance.

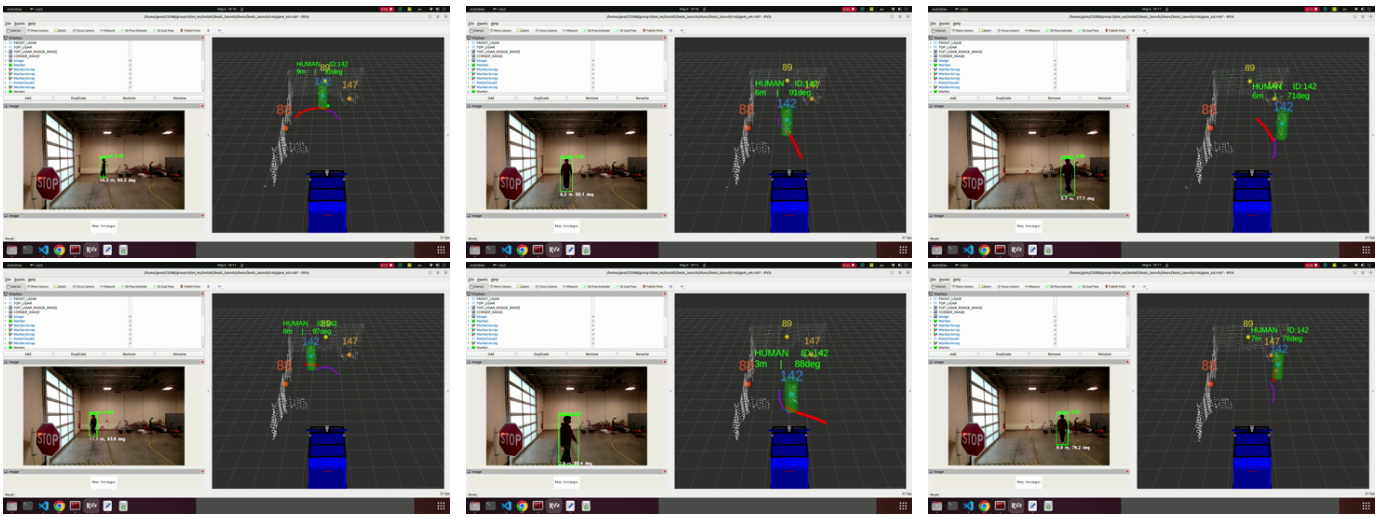

Fig. 2: Keyframes from a diffusion-predicted pedestrian-avoidance rollout on the GEM e4. Frames advance left-to-right, top-to-bottom. The joint multi-agent predictor proposes future pedestrian trajectories that the MPPI planner integrates as time-varying obstacle costs while tracking the reference path.

*f) Goal latching:* To suppress per-frame jitter and short detection dropouts, a goal-hold buffer accumulates candidate poses over a window of $T_h = 3.0\,\mathrm{s}$ and emits the elementwise median once $n_h \geq 5$ samples are collected. A flag accept_estimated controls whether ray-cast fallbacks contribute to the buffer. Once latched, the goal is held until externally reset, which preserves continuity across transient occlusions.

The module outputs the goal pose in both the map and vehicle frames, an estimated-goal flag, the filtered cluster cloud, and a 3D bounding-box marker colored by the estimated flag for downstream visualization. Table III collects the deployed parameters.

TABLE III: Text-promptable perception parameters.

| Symbol | Meaning | Default |
|---|---|---|
| $z_{\min}$, $z_{\max}$ | height band (m) | 0.15, 5.0 |
| $k$, $\kappa$ | SOR neighbors, $\sigma$-multiplier | 8, 2.0 |
| $\varepsilon$, $n_{\min}$ | DBSCAN radius (m), core threshold | 0.4, 3 |
| $d_{\text{est}}$ | ray-cast fallback distance (m) | 15.0 |
| $T_h$, $n_h$ | hold window (s), min samples | 3.0, 5 |
| box, text thr. | Grounding-DINO thresholds | 0.25, 0.20 |

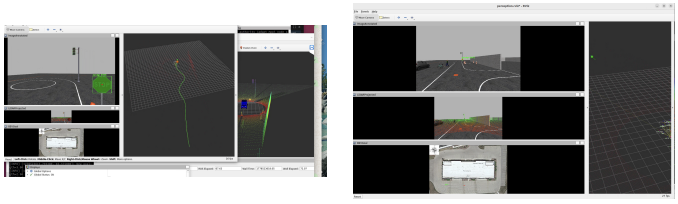

Fig. 3: Open-vocabulary goal selection in the Gazebo simulator. The LangSAM backend (Grounding-DINO + SAM 2.1) returns a mask for the prompted target (left), and the corresponding LiDAR frustum-clip cluster is reprojected to confirm the goal pose (right).

## VI. MPPI Motion Planner

This section specializes the sampling-based MPC framework of Section II-A to pedestrian-aware urban driving on the GEM e4 by instantiating the running cost $c(\cdot)$ and terminal cost $\phi(\cdot)$ of (2). The planner accepts two mutually exclusive pedestrian inputs. When full predicted trajectories from the diffusion node are available, the obstacle centroid advances along the rollout horizon by indexing into the predicted trajectory at each rollout step, while the Gaussian half-width $\sigma_{\text{ped}}$ remains fixed. In the legacy mode the predictor is replaced by a constant-velocity extrapolation whose covariance grows with the effective lead time, so that uncertainty widens at longer prediction horizons.

### A. Running Cost

The running cost decomposes into path-tracking, stability, and obstacle terms,

$$c(\boldsymbol{x}_t, \boldsymbol{u}_t, t) = c_{\text{pos}} + c_{\text{vel}} + c_{\text{stab}} + c_{\text{ped}} + c_{\text{cone}}. \quad (7)$$

Let $\mathcal{P} = \{\boldsymbol{p}_n\}$ be the reference waypoints. The tracking and stability terms are

$$c_{\text{pos}} = w_{\text{pos}} \min_n \left\| [x_t, y_t]^\top - \boldsymbol{p}_n \right\|_2, \; c_{\text{vel}} = w_{\text{vel}} |v_t - v_{\text{ref}}|, \quad (8)$$

with the stability term $c_{\text{stab}} = w_{\text{curv}}|\delta_t|v_t$ discouraging large steering at high speed. The cross-track form pulls rollouts toward the nearest waypoint rather than a fixed look-ahead.

*a) Trajectory-conditioned pedestrian cost:* The diffusion node supplies $M$ predicted pedestrian trajectories of length $H_{\text{ped}}$ at step $\Delta t_{\text{ped}}$. The rollout step $t$ is mapped to a predictor index $\tau(t) = \min(\lfloor t\,\Delta t/\Delta t_{\text{ped}} \rfloor,\, H_{\text{ped}} - 1)$, and the cost is an isotropic Gaussian repulsion around each predicted position $\boldsymbol{q}_{\tau(t)}^{(m)}$,

$$c_{\text{ped}}(\boldsymbol{x}_t, t) = w_{\text{obs}} \sum_{m=1}^{M} \exp\!\Big(-\tfrac{1}{2}\,\big\|[x_t, y_t]^\top - \boldsymbol{q}_{\tau(t)}^{(m)}\big\|_2^2 / \sigma_{\text{ped}}^2\Big). \tag{9}$$

In the legacy constant-velocity mode the predicted position propagates linearly with an effective lead time $t_{\text{eff}} \propto \|\boldsymbol{x}_{xy} - \boldsymbol{x}_{\text{ego},xy}\|/v_{\text{ego}}$, the covariance is anisotropic in the pedestrian's heading frame, and its scale grows as the per-detection confidence decays with $t_{\text{eff}}$. The same cost is combined with a hard clearance step $\mathbf{1}(d < r_{\text{clear}})$ at weight $w_{\text{obs,hard}}$ and a static exponential falloff at weight $w_{\text{obs,soft}}$. Static obstacles (cones) are treated identically with their own clearance radius $r_{\text{cone}}$ and weights $w_{\text{cone,hard}}$, $w_{\text{cone,soft}}$.

### B. Terminal Cost

The terminal cost anchors forward progress and is gated by $w_{\text{term}}$,

$$\phi(\boldsymbol{x}_H) = w_{\text{term}} \big\|[x_H, y_H]^\top - \boldsymbol{g}_{xy}\big\|_2, \tag{10}$$

where $\boldsymbol{g}_{xy}$ is the last reference waypoint. The default $w_{\text{term}} = 0$ disables the term and is re-enabled when strong obstacle repulsion pushes rollouts away from the path.

### C. Hyperparameter Defaults

The deployed hyperparameters are summarized in Table IV.

TABLE IV: MPPI hyperparameter defaults on the GEM e4.

| Symbol | Meaning | Default |
|---|---|---|
| $K$ | rollout samples | 100 |
| $H$ | horizon (steps) | 100 |
| $\Delta t$ | rollout step (s) | 0.1 |
| $f_{\text{ctrl}}$ | control rate (Hz) | 20 |
| $\sigma_a,\ \sigma_\delta$ | noise std. $(a, \delta)$ | 0.5, 0.15 |
| $\lambda$ | MPPI temperature | 0.1 |
| $v_{\text{ref}}$ | reference speed (m/s) | 4.0 |
| $L$ | wheelbase, Gazebo model (m) | 1.75 |
| $a_{\min},\ a_{\max}$ | accel bounds (m/s$^2$) | $-1.0$, 2.0 |
| $\delta_{\max}$ | steering bound (rad) | 0.61 |
| $w_{\text{pos}}$ | cross-track weight | 15.0 |
| $w_{\text{vel}}$ | velocity tracking | 5.0 |
| $w_{\text{curv}}$ | stability | 2.0 |
| $w_{\text{obs}}$ | Gaussian repulsion | 150.0 |
| $w_{\text{obs,hard}}$ | hard clearance | 250.0 |
| $w_{\text{obs,soft}}$ | exp. falloff | 40.0 |
| $w_{\text{cone,hard}}$ | hard cone exclusion | 250.0 |
| $w_{\text{cone,soft}}$ | exp. cone falloff | 40.0 |
| $w_{\text{term}}$ | terminal weight | 0.0 |
| $\sigma_{\text{ped}}$ | Gaussian width (m) | 1.5 |
| $r_{\text{clear}}$ | clearance radius (m) | 1.5 |
| $r_{\text{cone}}$ | cone radius (m) | 0.8 |
| $\Delta t_{\text{ped}}$ | predictor step (s) | 0.25 |
| $H_{\text{ped}}$ | predictor horizon (steps) | 20 |

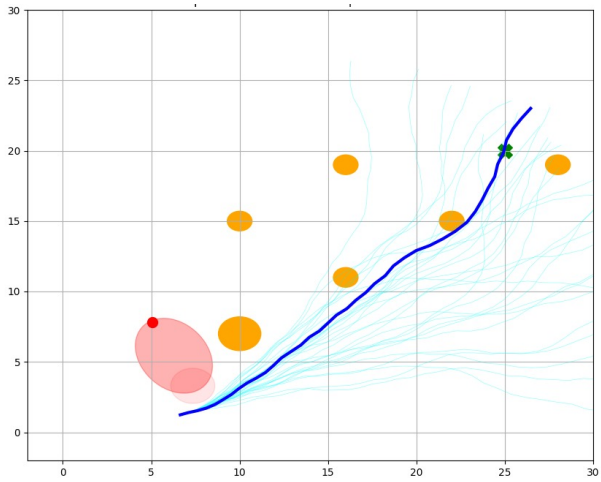


Fig. 4: MPPI samples the control space and generates candidate trajectories, from which one is selected by cost-weighted softmin.

## VII. Implementation Progress

**Hardware bring-up.** The Ouster OS1-128, OAK-D LR, Septentrio GNSS/INS, and PACMod2 have been integrated under a unified sensor-initialization launch with a lifecycle handoff that brings up cameras, GNSS, and visualization only after the LiDAR is active.

**Baseline controllers.** Pure pursuit and Stanley run end-to-end on the GEM e4 with PID longitudinal control, gated on the joystick enable signal.

**MPPI controller (on-vehicle tested).** A GPU-based Torch implementation has been validated on the vehicle, achieving

low lateral tracking error on the reference loop and reacting to injected static obstacles within a single control cycle.

**Diffusion predictor.** Both the single-agent and joint models have been trained on Argoverse 2 and integrated as a ROS 2 node, with trajectory smoothing added to reduce per-frame mode switches. Fine-tuning on GEM e4 rosbags is ongoing.

**Sensor fusion and high-level safety.** The fused pedestrian estimate drives both the raw-mode MPPI and the prediction node, and the high-level node produces the four safety states consistently across all prediction modes.

**Text-promptable perception.** The YOLO-World and LangSAM backends operate in both the Gazebo simulator and the on-vehicle workspace. Prompt switching, frustum clipping, the ray-estimated fallback, and the goal-hold layer have been verified against highbay-track recordings.

## VIII. Remaining Work

**GEM rosbag fine-tuning.** Training windows are being extracted from on-vehicle rosbags, and both checkpoints will be fine-tuned prior to the final demonstrations.

**On-vehicle A/B comparison.** Identical routes will be evaluated under three configurations, namely (i) Stanley with constant-velocity prediction, (ii) MPPI with constant-velocity prediction, and (iii) MPPI with joint diffusion. The primary metrics are minimum clearance, intervention rate, and lap time.

**Latency budget reduction.** The end-to-end perception-to-control latency remains within the current budget, but the diffusion-mode worst case must be reduced further by batching DDIM steps on the GPU and removing a CPU round-trip in the tracker.

**Failure-mode analysis.** A dedicated set of edge cases, including occluded pedestrians behind parked vehicles, diagonal group crossings, and false positives from foliage, will be used to characterize the relative strengths of each predictor.

**Cross-frame tracking in text-promptable perception.** An IoU-based 2D tracker would stabilize the goal when multiple candidates compete across frames, and a depth-camera fallback would improve performance at mid-range distances.

## IX. Risks and Mitigations

The most significant open risk is prediction overconfidence, in which a narrow diffusion sample may underestimate variance when a pedestrian is about to change behavior, leading MPPI to plan overly aggressive maneuvers. The confidence-growth obstacle cost partially mitigates this effect, and the high-level safety node provides a hard STOP_YIELD fallback whenever time-to-collision falls below threshold, regardless of the active predictor. A second risk is sensor degradation under low-light or rainy conditions. The LiDAR-and-camera fusion weights keep LiDAR dominant in range, so the system degrades gracefully if the camera becomes unreliable. The same principle protects the text-promptable perception path, in which the LiDAR frustum clip continues to bound the goal even when the visual mask is noisy.

## X. Conclusion

We have developed an end-to-end pedestrian-aware autonomy stack on the Polaris GEM e4 that couples a sampling-based MPPI planner with a diffusion trajectory predictor and an open-vocabulary perception module. The MPPI formulation supports both constant-velocity and trajectory-conditioned obstacle costs under the same kinematic-bicycle dynamics, enabling controlled comparison against the Stanley baseline. Remaining work focuses on GEM-domain fine-tuning, on-vehicle A/B evaluation, and latency reduction.

## References


[1] T. Gu, G. Chen, J. Li, C. Lin, Y. Rao, J. Zhou, and J. Lu, "Stochastic Trajectory Prediction via Motion Indeterminacy Diffusion," in *Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[2] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[3] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," in *Proc. Int. Conf. on Learning Representations (ICLR)*, 2021.

[4] A. Nichol and P. Dhariwal, "Improved Denoising Diffusion Probabilistic Models," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2021.

[5] B. Wilson *et al.*, "Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting," in *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2021.