GENG5512 MPE Engineering Research Project Part 2

Final Report

Pedestrian-Aware Velocity Planning for Autonomous Campus Shuttle Navigation in Shared Spaces

Harry John Tolcher
23060513

School of Engineering, University of Western Australia

Supervisor: Thomas Bräunl
School of Engineering, University of Western Australia

Word count: 7209

School of Engineering
University of Western Australia

Submitted: 18 May 2026

Project Summary

Stock autonomous-driving software stacks treat every detected pedestrian like a roadside obstacle, stopping the vehicle whenever a pedestrian enters its safety envelope. On a shared campus pathway, where pedestrians and a low-speed shuttle routinely share the same space, this stop-start behaviour makes a useful service impossible. This thesis develops a pedestrian-aware velocity planner that addresses the problem, and evaluates it on the nUWAy EasyMile EZ10 shuttle operated by the University of Western Australia.

The planner is a drop-in plugin to the behaviour-velocity stage of the open-source Autoware planning stack, and has two components. The first is a crowd-matching velocity model that sets the shuttle's desired forward speed to the mean walking speed of pedestrians moving in the same direction within a 15 m radius, supported by the macroscopic flow literature in which the same speed–density relationship has been characterised at pedestrian densities. The second is an open-loop, quadratic ramp that smoothly reduces the commanded velocity as the nearest pedestrian in front closes from 10 m to 3 m, which accounts for stationary pedestrians or pedestrians walking more slowly than the crowd average. The stock Autoware obstacle-stop module is kept downstream as a hard safety backup at 0.8 m. The remaining default modules of the planning preset were removed because they are not needed for shared-pathway operation.

The proposed module was benchmarked against seven alternative configurations made up of different combinations of these components, including stock Autoware, a tuned version of Autoware, a Helbing–Molnár social-force controller, and a closed-loop PID variant. All configurations were evaluated in the AWSIM simulation environment with bidirectional pedestrian traffic on a 6 m shared pathway. On the straight scenario, the proposed configuration achieves an 8.4-fold reduction in velocity standard deviation and a 6.8-fold reduction in mean absolute jerk relative to stock Autoware, with zero stops compared to three for the stock baseline. On the perpendicular crossing scenario, it again records zero stops where the closed-loop PID variant records five. The 0.155 m/s^3 mean jerk and 1.4 m/s^2 maximum deceleration sit well within the comfort thresholds of ISO 22737:2021 and other literature baselines. A density sweep from 0 to 60 simultaneously-spawned pedestrians shows that all three smoothness metrics degrade gracefully across the campus operating envelope.

The principal contribution is framing pedestrian interaction as a flow-matching problem rather than an obstacle-avoidance problem. The solution is packaged as an Autoware plugin that requires no changes to the surrounding stack, and is portable to any vehicle running the same planning architecture. Recommended future work includes further simulation evaluation on curved geometries and more complex shared-space pedestrian distributions, validation of the model on a live shuttle in a campus environment, and investigation of model-predictive pedestrian tracking.

Acknowledgements

I would like to thank my supervisor, Professor Thomas Bräunl, for the opportunity to contribute to the nUWAy programme and for his guidance throughout my thesis. I would also like to thank PhD student Lee Le for his ongoing guidance and efforts to implement the Autoware software stack on nUWAy 4. Thanks are also due to Lecturer Kieran Quirke-Brown for recommending that I join the Renewable Energy Vehicle (REV) Project for my thesis and for his ongoing support. I would also like to extend my thanks to other members of the REV project not listed here for making my overall thesis experience enjoyable. I am grateful to the UWA REV Project for providing the unique opportunity to work on a state-of-the-art electric shuttle bus and the supporting infrastructure that made testing and data collection possible. Thanks also to the Engineering workshop for technical assistance with nUWAy, and to the Autoware Foundation maintainers whose open-source planning stack is the platform on which this work is implemented. Finally, I thank my family and friends for their support over the course of the project.

Nomenclature

Abbreviations

AWSIM	Autoware Simulator (Unity-based)
CAN	Controller Area Network
CPU	Central Processing Unit
GNSS	Global Navigation Satellite System
GPU	Graphics Processing Unit
HD map	High-Definition map
IMU	Inertial Measurement Unit
ISO	International Organization for Standardization
LiDAR	Light Detection And Ranging
LWR	Lighthill–Whitham–Richards (kinematic-wave model)
MFD	Macroscopic Fundamental Diagram
MPC	Model-Predictive Control
NDT	Normal Distributions Transform
PCD	Point Cloud Data
PID	Proportional–Integral–Derivative (controller)
REV	Renewable Energy Vehicle (UWA team)
ROS	Robot Operating System
SFM	Social Force Model
TTC	Time-To-Collision
UWA	The University of Western Australia

Symbols

v_{desired}	Desired velocity ($\mathrm{m/s}$)
v_{ego}	Ego vehicle velocity ($\mathrm{m/s}$)
v_{crowd}	Mean crowd velocity ($\mathrm{m/s}$)
τ	Relaxation time constant (s)
d	Distance to nearest pedestrian (m)
d_{brake}	Braking onset distance (m)
d_{min}	Proximity ramp floor distance (m)
r_{det}	Co-flow detection radius (m)
v_{thresh}	Co-flow speed threshold ($\mathrm{m/s}$)
n_{min}	Minimum co-flow count
ρ	Pedestrian-flow density ($\mathrm{ped/m^2}$)
q	Pedestrian-flow rate ($\mathrm{ped/m/s}$)

Chapter 1

Introduction, Literature Review and Project Objectives

1.1 Introduction

Shared-space navigation, where vehicles operate on paths shared with pedestrians without lane markings or right-of-way conventions, is among the harder open problems in low-speed autonomy. The UWA Renewable Energy Vehicle (REV) team's nUWAy is a direct example: an electric autonomous shuttle that operates almost entirely on shared pedestrian paths. The shuttle runs the open-source driving stack Autoware, which by default places a stop point on the path five metres short of every detected pedestrian. On a shared path this produces stop-and-go motion that makes the service nearly unusable and uncomfortable. The current workaround is to disable the perception system and rely on the safety operator to stop the vehicle via the dead-man switch or by switching back to manual driving.

This thesis proposes a Pedestrian-Aware Velocity Planner, an Autoware plugin that estimates the local pedestrian flow around nUWAy and adjusts forward speed to match it. A second component slows the shuttle smoothly as it approaches the nearest pedestrian in its path, and the stock obstacle-stop modules are kept as a fallback. The rest of this chapter reviews relevant literature (Section 1.2) and states the project objectives (Section 1.3).

1.2 Literature Review

1.2.1 Autonomous Shuttle Deployments

Low-speed automated shuttles are small electric buses carrying 8–15 passengers at up to about $25\,\mathrm{km/h}$ on fixed routes, driven autonomously with an on-board safety operator (EasyMile, Navya, 2getthere). The U.S. Department of Transportation State-of-the-Practice review [1] reports that deployed shuttles cope well on regular roads but struggle on shared pathways, where the usual fix is conservative stop-and-go with the operator taking over when needed. The survey is dominated by regular-road routes and so under-represents the shared-path failure mode this thesis addresses. The UNC Charlotte pilot of Golchin et al. [2] (825 trips on a mixed-traffic campus route similar to nUWAy's) shows the same pattern: pedestrian and cyclist interactions are the leading cause of the autonomous system being

switched off, with no software-side fix reported beyond operator hand-off. The closest pedestrian-aware shuttle work is the collision-avoidance system of Pascucci et al. [3], which also reduces to a binary stop-or-go decision and depends on roadside infrastructure not available on campus shared paths.

1.2.2 Autonomous Driving Software Stacks

The two main open-source autonomous-driving stacks, Autoware [4] and Apollo [5], split the planning pipeline into the same six stages (Figure 1.1): mission, behaviour-path, behaviour-velocity, motion-velocity, smoother, and control. Both let new modules plug into the behaviour-velocity stage without touching the rest of the stack.

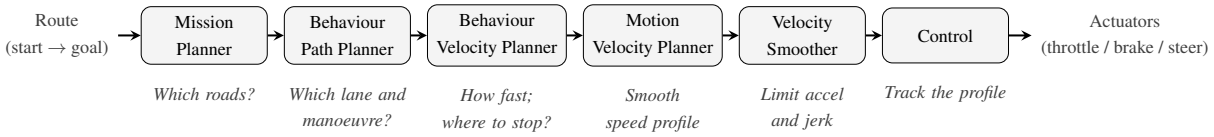

Figure 1.1: Planning subsystem of a typical autonomous-driving stack (Autoware, Apollo). The behaviour-velocity stage is the entry point for pedestrian-aware longitudinal logic.

This thesis sits in the behaviour-velocity stage. Stock Autoware ships three pedestrian-relevant modules: a crosswalk module that triggers on time-to-collision, a run-out module for obstacles hidden behind occlusions, and an obstacle-cruise planner that places a stop point on the path roughly $5\,\mathrm{m}$ short of any obstacle ahead. All three pick the single nearest relevant obstacle and ignore the rest of the scene (Figure 1.2). The Apollo modules [5] work the same way. The plugin interface is flexible, but every shipped pedestrian module reduces the scene to one pedestrian at a time. This template cannot handle many simultaneous pedestrians or reason about the density and flow of a co-moving crowd, both of which are the norm in a shared space.

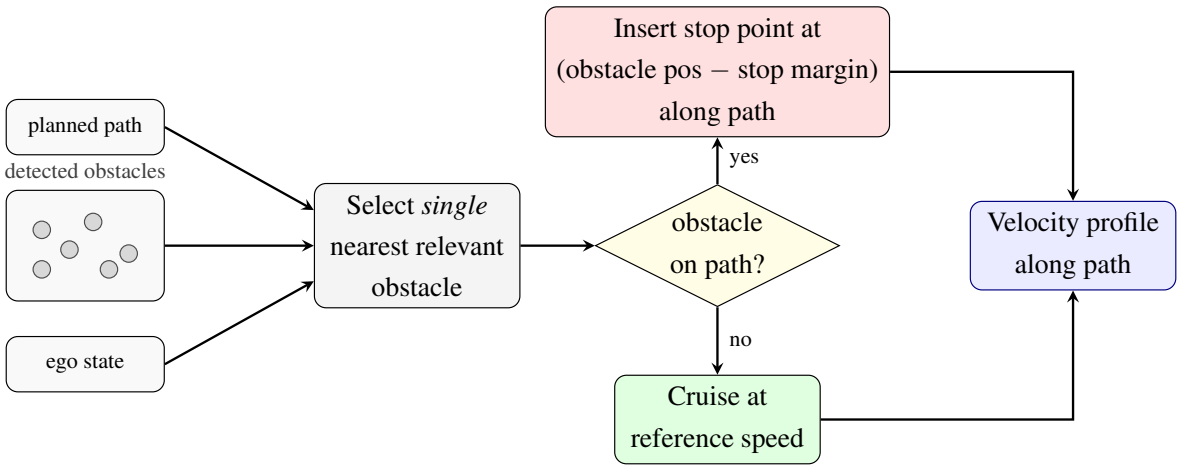

Figure 1.2: Internal structure of stock behaviour-velocity modules: a single nearest relevant obstacle is selected from the detected-obstacle set, and the module emits a stop-point at a fixed margin behind the obstacle along the path (the bus then decelerates toward it via downstream velocity smoothing), or a cruise velocity when the path is clear.

1.2.3 Passenger Comfort Engineering

A mechanically safe manoeuvre is not necessarily comfortable to ride. The relevant physical quantity is jerk, the rate of change of acceleration: passengers feel how abruptly acceleration is applied, not just its magnitude (Figure 1.3).

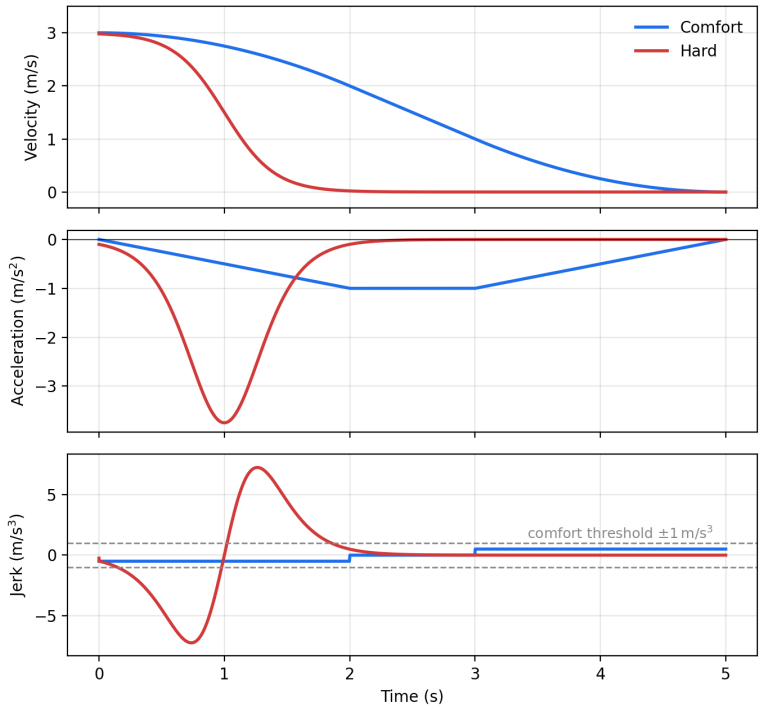

Figure 1.3: Two braking profiles to the same final speed: a bounded-jerk profile (blue) whose peak jerk sits exactly on the $\pm 1\,\mathrm{m/s^3}$ comfort threshold, and a near-step profile (red) whose jerk spike exceeds the threshold by several times.

Four sources propose how comfort parameters should be measured and bounded. ISO 2631-1:1997 [6] specifies frequency-weighted whole-body-vibration metrics for transit ride quality. De Winkel et al. [7] identify jerk as a dimension missing from ISO 2631-1:1997 for autonomous-vehicle ride comfort. ISO 22737:2021 [8] sets a 4.9 m/s^2 max for accelerations to be comfortable for low-speed automated driving systems on predefined roads. Shared spaces fall outside its scope, so this limit is moreso an extreme upper limit. Elbanhawi et al. [9] report longitudinal jerk as the strongest predictor of passenger discomfort during forward motion (separate from turning), with the comfort threshold being reported as $2.6\,\mathrm{m/s^3}$.

1.2.4 Pedestrian Behaviour in Shared Spaces

A "shared space" is public ground used simultaneously by pedestrians and vehicles, without kerbs, lane markings, or formal right-of-way conventions. Campus paths, plazas, and pedestrianised streets are typical examples. The defining feature is that vehicles must drive in among pedestrians rather than alongside them (Figure 1.4).

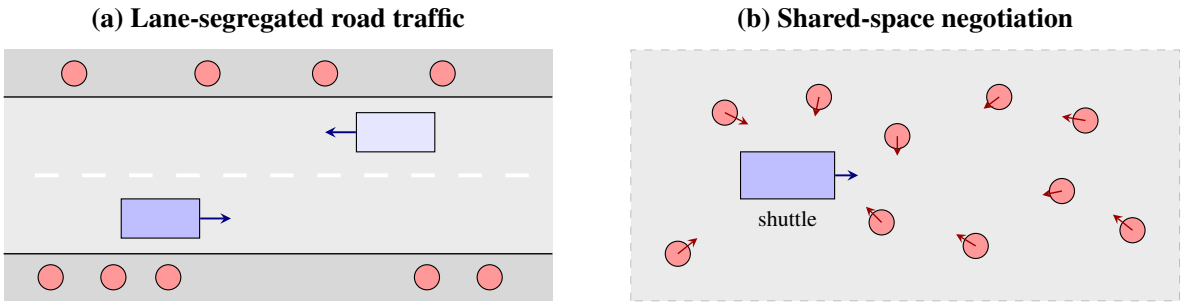

Figure 1.4: Lane-segregated road traffic (a) versus a shared space (b).

The CityMobil2 European programme [10] reports that pedestrians in shared spaces treat slow-moving vehicles as obstacles to step around rather than traffic with right of way, continuously adjusting their path and speed in response. The report's recommendations are about how to operate the service rather than how to program the controller, leaving the question of how the vehicle should respond open. Pedestrian density is the single best predictor of average vehicle speed, and shared-space behaviour is responsive, social, and density-dependent in ways that cannot be captured by treating each pedestrian as an independent obstacle.

1.2.5 Pedestrian Dynamics: Microscopic and Macroscopic Models

Pedestrian crowds are modelled at two scales: microscopically, as individual people, and macroscopically, as a continuous flow with an overall density and an average velocity.

Microscopic models. The Social Force Model (SFM) of Helbing and Molnár [11] treats each pedestrian as a particle under three "social" forces (Figure 1.5): an attractive driving force toward the destination at preferred walking speed, exponentially decaying repulsions from nearby pedestrians, and similar repulsions from walls.

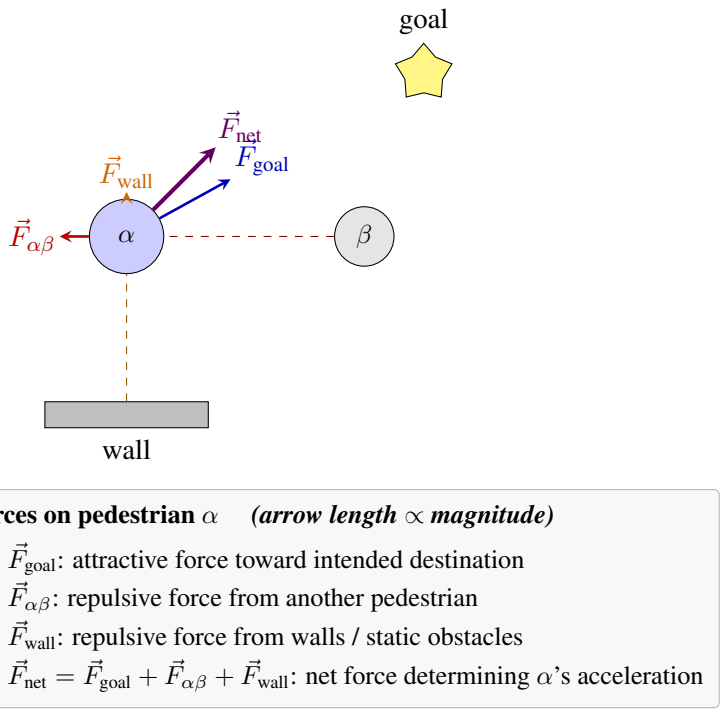

Figure 1.5: The three SFM force components acting on pedestrian α and their resultant $\vec{F}_{\text{net}}$ (purple). Arrow lengths are drawn proportional to magnitude.

Extensions cover group walking [12], [13] and vehicle–pedestrian interaction by treating the vehicle as another particle that exerts a repulsive force on pedestrians [14]. SFM was designed to describe how pedestrians move, not to tell a vehicle what to do. Section 3.4 shows that using it as a speed controller creates a feedback loop where higher vehicle speed produces stronger braking, leading to sustained oscillation. The deep-learning alternative reviewed by Korbmacher and Tordeux [15] predicts trajectories more accurately but needs more compute and training data than nUWAy currently has.

Macroscopic models. The macroscopic perspective treats a stream of agents like a fluid, with density ρ and average velocity v linked by the fundamental diagram: average speed falls roughly linearly with density, and the flow rate $q = v\rho$ is parabolic, peaking at half the jam density (Greenshields [16]). Lighthill, Whitham, and Richards [17], [18] extended this into a continuous wave model, and Daganzo [19] recast it as discrete cells for real-time control. Geroliminis and Daganzo [20] showed the same relation holds across an entire urban road network, providing the basis for city-wide perimeter control. The same relationship has been measured for pedestrian streams. This literature is referenced as theoretical support for treating a co-flowing crowd as a stream with its own typical speed rather than a set of independent obstacles, not as the source of the controller design. Existing applications operate at the network scale. No published work has applied the idea to a single vehicle moving with pedestrians.

1.2.6 Model-Predictive Control in Pedestrian-Dense Scenes

The dominant approach to vehicle control in pedestrian-dense scenes is model-predictive control (MPC). On every tick the controller scores candidate command sequences against a cost combining tracking error, near-miss penalties, and jerk, applies the first command of the lowest-cost sequence, then re-plans (Figure 1.6). A pedestrian-motion predictor (Section 1.2.5) usually runs inside the optimisation. Most published work controls steering and speed jointly. Aslam et al. [21] are representative: their MPC commands the wheel velocities of a service robot with a learned trajectory predictor inside the optimisation, at a per-tick cost that does not scale to the embedded compute on a low-speed shuttle. Yang and Özgüner [22] restrict the optimisation to forward speed only, with SFM as the in-loop predictor (vehicle as another particle exerting force). They beat reactive baselines on the safety-versus-throughput trade-off, but still need a quadratic-programming solve every tick. Because SFM is in the cost function, they inherit the same feedback loop where higher speed produces stronger braking, isolated in Section 3.4. Their forward-speed-only scope is the closest match to this thesis: a route-bound shuttle that hands steering to the stock controller.

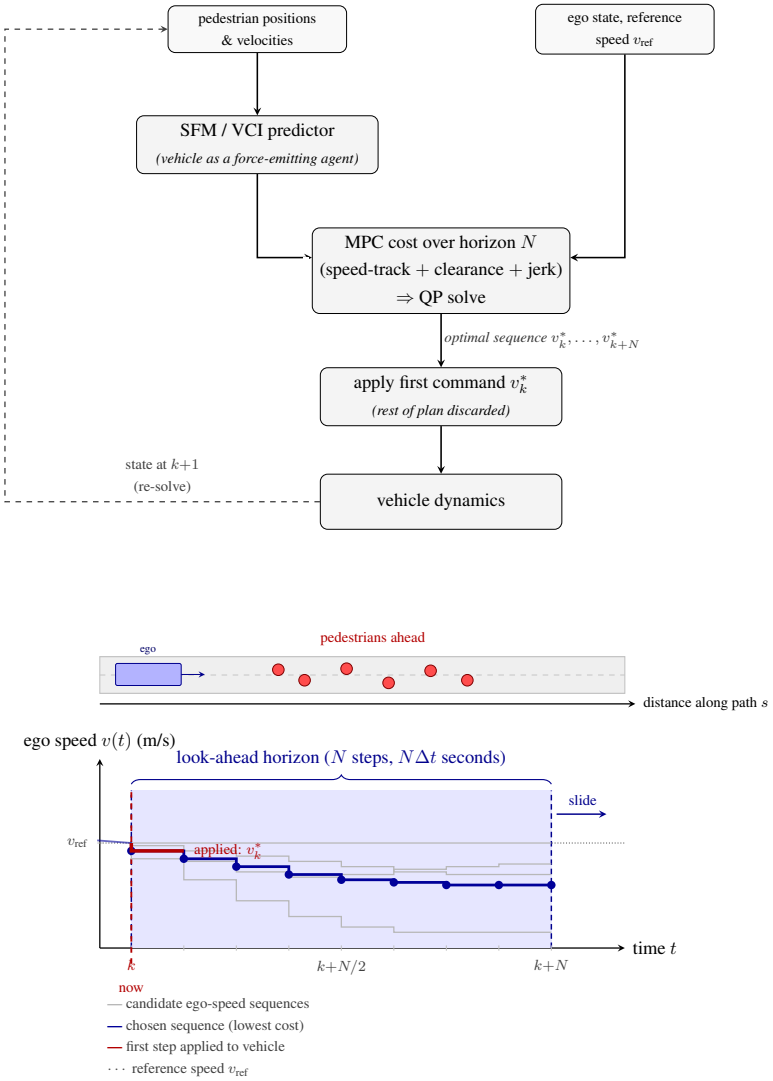

Figure 1.6: The SFM-with-MPC longitudinal-speed regulator of Yang and Özgüner [22]: architecture (top) and receding-horizon principle on ego speed (bottom).

1.2.7 Synthesis and Research Gap

Three observations follow. First, deployed shuttles and the stock planning stacks they run on choose forward speed by reacting to one obstacle at a time, with no continuous adjustment to the surrounding crowd. Second, pedestrian-dynamics modelling is well developed at both the individual and the crowd-flow scale, but neither approach has been put inside the controller of a deployed shuttle. Third, research on crowd-aware vehicle control is dominated by MPC schemes that control steering and speed together. Yang and Özgüner's speed-only formulation is the closest published precedent, and the macroscopic-flow approach has not yet been used to control a single vehicle moving with pedestrians.

The gap is the absence of a forward-speed controller for shared-space shuttles that adjusts the vehicle's speed based on the surrounding pedestrian flow, runs inside the stock behaviour-velocity tick, and works alongside the stock obstacle-stop modules as a fallback. The Pedestrian-Aware Velocity Planner targets that gap.

1.3 Project Objectives

The aim of this thesis is to implement a novel pedestrian based velocity controller on the nUWAy platform. The work spans the two consecutive units that make up the project: GENG5511 (Part 1, Integration and testing, Semester 2 2025) and GENG5512 (Part 2, Planner design and evaluation, Semester 1 2026). It is organised around six specific, measurable objectives.

1. **Integrate Autoware on nUWAy.** Build a Dockerised install for nUWAy's compute hardware (Nvidia Jetson Orin), integrate the on-board sensors (Velodyne LiDAR, SBG GNSS+IMU, vehicle CAN), create a pcd and lanelet2 HD map of the Crawley campus, bring up localization, and launch the stock planning pipeline.

2. **Establish a repeatable simulation environment.** Build AWSIM scenes (straight co-flow, perpendicular crossing, density sweep), a Social-Force-driven bidirectional pedestrian simulator with keep-left walking, a ground-truth object-relay pipeline, and a metric-extraction pipeline on recorded rosbags.

3. **Implement a classical Social Force Model controller.** Build a forward-speed controller that treats the vehicle as an SFM agent, in the same plugin framework. This provides a representative baseline from the prior microscopic-model literature.

4. **Design and implement the Pedestrian-Aware Velocity Planner.** Combine a crowd-matching speed target, drawn from the macroscopic-flow framing of Section 1.2.5, with an open-loop quadratic ramp that slows the shuttle as the nearest pedestrian in its path approaches. Implement it as a behaviour-velocity plugin that runs inside the stock Autoware planning pipeline, leaves the rest of the stack untouched, and keeps the stock obstacle-stop module as a fallback. Steering and lane following is left to the stock controller.

5. **Compare the controller to other methods.** Benchmark on the straight and perpendicular-crossing scenarios against stock Autoware, a parameter-tuned preset, the SFM baseline, a closed-loop PID variant, and configurations that isolate the crowd-matching and proximity components individually. Report velocity smoothness, safety (clearance, time-to-collision, stop count), and crowd-speed tracking accuracy against the thresholds of Section 1.2.3.

6. **Characterise the operating environment and fallback behaviour.** Bus sensor replay from a standard drive into the full Autoware stack to track pedestrian counts. Sweep pedestrian density from 0 to 60 simultaneously spawned at the proposed configuration to map performance degradation across the range. Stress-test the dual-layer safety stack with stationary obstacles at fixed onset distances and dynamically spawned pedestrians at set distances ahead of the vehicle.

Chapter 2

Model Formulation

This chapter presents the pedestrian-aware velocity planning module: a plugin for the Autoware behaviour-velocity planner that matches the speed of the surrounding pedestrian flow, slows smoothly as the nearest pedestrian in the path approaches, and falls back to a tuned obstacle-stop. It covers the system, simulation environments, planning preset, perception pipelines, module formulation, comparative-study design, validation, and limitations.

2.1 System Overview

The nUWAy autonomous shuttle is an EasyMile EZ10 platform operated by the Renewable Energy Vehicle (REV) Project at UWA. nUWAy operates on shared campus pathways 3–8 m wide, on which pedestrians and vehicles travel at walking speeds under a campus-mandated 5 km/h speed limit.

The Autoware planning pipeline processes a route through a sequence of stages: mission planning, behaviour path planning, behaviour velocity planning, motion velocity planning, velocity smoothing, and vehicle control. The module presented here is a plugin within the behaviour velocity stage (Figure 2.1). The configuration of the surrounding Autoware stack for shared-pathway operation is described in Section 2.3.

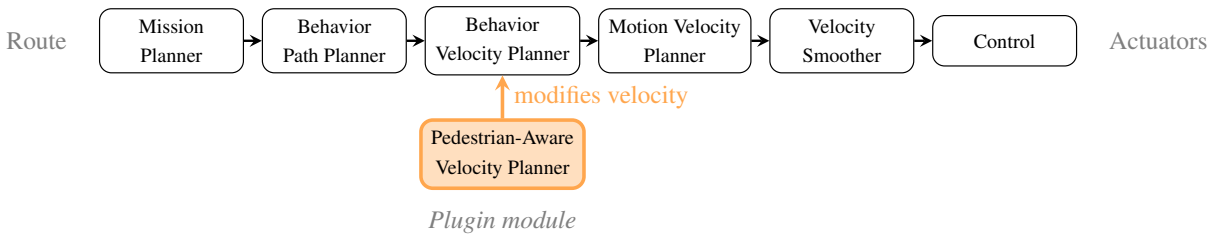

Figure 2.1: Autoware planning pipeline. The Pedestrian-Aware Velocity Planner operates as a plugin within the Behavior Velocity Planner, modifying the velocity profile along the existing planned path.

2.2 Simulation Environments

Testing was conducted in AWSIM, the Unity-based Autoware simulator, so the ISO 22737:2021 pedestrian-interaction scenarios [8] could be evaluated repeatedly without risk. A pedestrian simulator driven by the Helbing–Molnár social force model [11] generates bi-directional traffic with the

pedestrians keeping left. Two environments were used: a lightweight environment that bypasses Autoware localisation and perception in favour of AWSIM's ground-truth pose (Section 2.4.1), used for the comparative study, and a full-stack environment that runs the standard Autoware NDT + CenterPoint chain.

The lightweight environment hosts two scenarios: a 400 m straight 6 m-wide shared pathway with a 5 km/h limit (Figure 2.2) and a crossing scenario that adds a perpendicular flow, exercising the ISO 22737:2021 vulnerable-road-user crossing geometry (Figure 2.3).

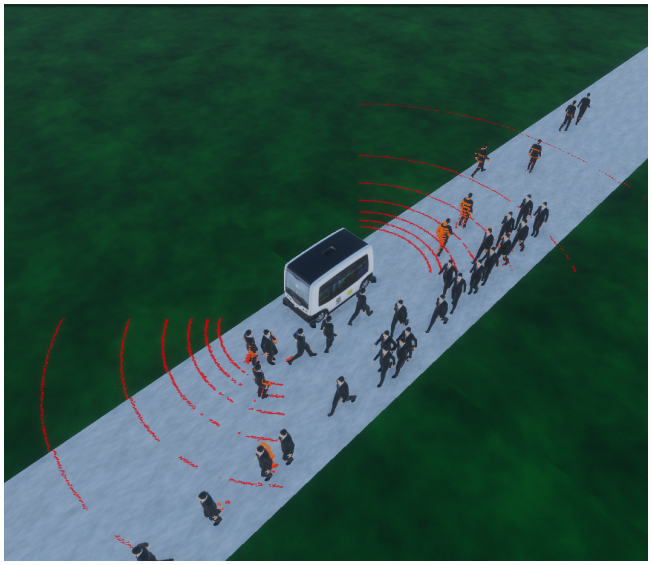

Figure 2.2: Isometric view of the straight scenario in the lightweight environment with bi-directional pedestrian traffic on the 6 m shared pathway.

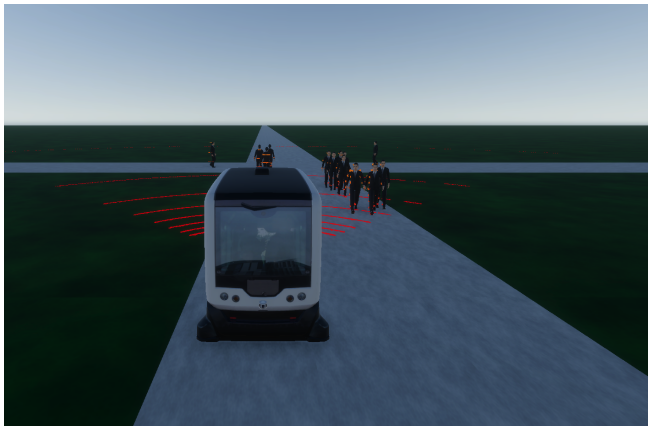

Figure 2.3: Lightweight environment, crossing scenario. A perpendicular pedestrian flow intersects the shuttle's path while bi-directional traffic continues along the shared pathway.

The full-stack environment runs the standard Autoware NDT + CenterPoint chain (Figure 2.4). Walls, pillars, trees, and buildings were added so NDT had enough geometric features to converge against the sparse VLP-16 returns, and the resulting environment was captured as a PCD map. Running the full chain alongside Unity saturated the GPU and CPU, producing uneven frame rates, so the

comparative study (Section 2.6) ran in the lightweight environment for consistent timing. The full-stack environment was kept for validating the Autoware multi-object tracker against ground truth in Section 3.3.

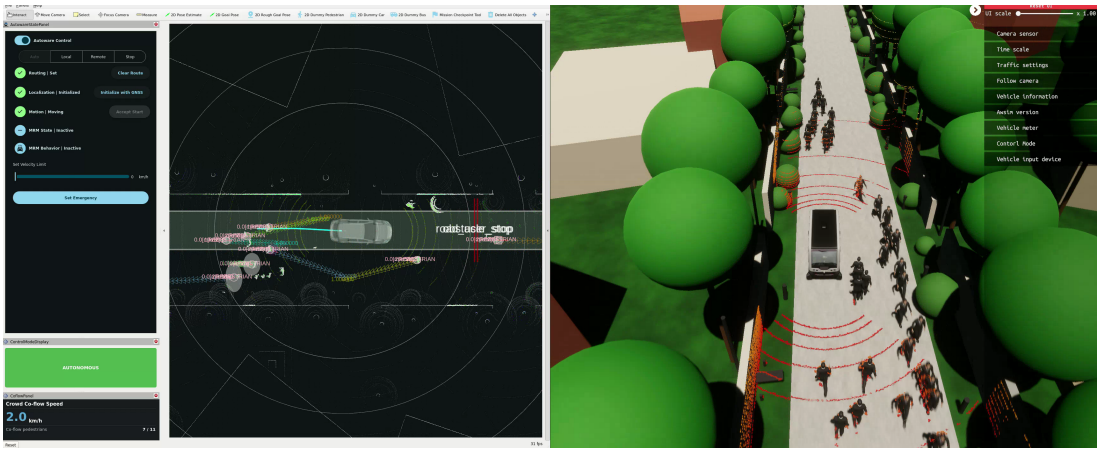

Figure 2.4: Full-stack simulation environment running the Autoware NDT localisation and CenterPoint perception pipeline, with RViz on the left and the AWSIM Unity scene on the right.

2.3 Planning Stack Configuration

The default Autoware preset is designed for structured public-road driving with conservative margins for higher-speed traffic [4]. Several of those modules are directly responsible for the stop-start behaviour this work aims to fix. A modified preset (Table 2.1) disables most behaviour and motion modules, keeping only the goal and start planners, the proposed plugin, and the path optimiser with elastic-band smoother. Autoware's obstacle stop module is kept downstream with its activation margin reduced to 0.8 m, as a last-resort fallback.

Table 2.1: Planning stack modules enabled in the default Autoware preset versus the modified preset used for shared-pathway operation.

Stage	**Default preset**	**Modified preset**
Behaviour path	Goal planner	Goal planner
	Start planner	Start planner
	Side shift	
	Static obstacle avoidance	
	Dynamic obstacle avoidance	
	Lane change (left / right)	
	Avoidance-by-lane-change	
	Sampling planner	
	Bidirectional traffic	
Behaviour velocity	Crosswalk	Velocity planner plugin
	Walkway	
	Traffic light	
	Intersection	
	Roundabout	
	Merge-from-private	
	Blind spot	
	Detection area	
	Virtual traffic light	
	No-stopping area	
	Stop line	
	Occlusion spot	
	Speed bump	
	No-drivable-lane	
Motion path	Path optimiser	Path optimiser
	Elastic-band smoother	Elastic-band smoother
Motion velocity	Obstacle stop	Obstacle stop (margin 0.8 m)
	Obstacle slow-down	
	Obstacle cruise	
	Dynamic obstacle stop	
	Out-of-lane	
	Obstacle velocity limiter	
	Run-out	
	Boundary departure prevention	
	Road user stop	
Velocity smoother	JerkFiltered	JerkFiltered

2.4 Perception Pipeline

The module consumes pedestrian detections with velocity estimates. Two pipelines were developed: a simulation pipeline isolating the planner from real-world perception error and a deployment pipeline detecting pedestrians from a sparse 16-beam LiDAR.

2.4.1 Simulation

In the lightweight environment, AWSIM exposes pedestrian and vehicle ground-truth state directly. Pedestrian objects with their ground-truth positions and velocities are republished onto the Autoware tracker output topic that the planner subscribes to, and the AWSIM ground-truth ego pose is fed in place of the NDT scan-matcher output. No detection, association, or velocity-estimation step sits between AWSIM and the planner (Figure 2.5). This isolates the comparative study from perception error and from the geometric-feature requirements of NDT, which the lightweight environment lacks. The full-stack environment restores the standard Autoware NDT + CenterPoint + multi-object tracker chain for the tracker-validation work of Section 3.3.

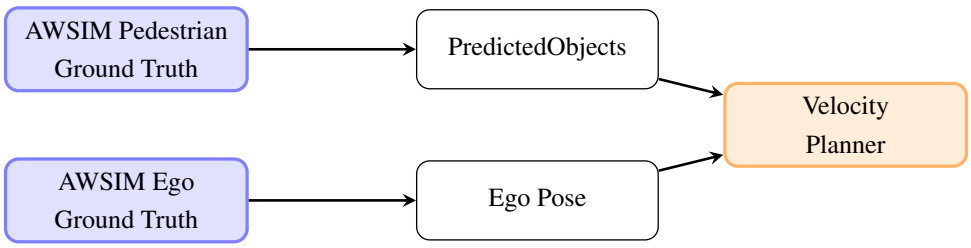

Figure 2.5: Simulation perception pipeline in the lightweight environment. AWSIM pedestrian and ego ground-truth state are republished directly onto the Autoware topics consumed by the velocity planner. No detection, association, or velocity-estimation step sits between AWSIM and the planner.

2.4.2 Live Deployment

nUWAy runs the standard Autoware tracker and map-based prediction. The default detector, CenterPoint [23], is trained on dense 64/128-beam LiDAR. nUWAy carries a 16-beam Velodyne VLP-16, which produces only a handful of returns per pedestrian at campus ranges, and the stock weights initially gave poor recall. Two detectors were evaluated on nUWAy rosbags: CenterPoint [23] and the Apollo instance-segmentation network [5], which uses clustering and was expected to degrade more gracefully at low point density. A post-filter dropping non-pedestrian classes and oversized boxes allowed the CenterPoint threshold to be relaxed without producing spurious tracks, recovering recall. The two detectors were then equivalent in performance (Figure 2.6), and CenterPoint was kept for consistency with mainline Autoware.

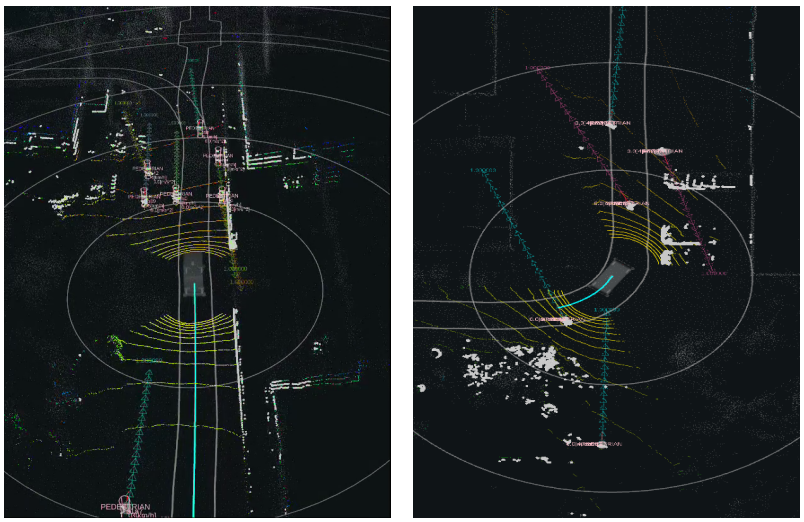

Figure 2.6: Live nUWAy perception output displayed in RViz during a campus rosbag replay showing pedestrian detections after the label-remap and bounding-box post-filter stage.

2.5 Velocity Planning Module

Two velocity-planning approaches were investigated. A classical social-force formulation [11], treating pedestrians as particles that push the vehicle away, proved unsuitable on shared paths (Section 2.5.1). This motivated the crowd-matching model with a proximity component described below.

2.5.1 Classical Social Force Approach

The classical SFM [11] computes a repulsive force between vehicle and pedestrian as an exponential function of separation (Figure 2.7):

$$F = A \exp\left(\frac{r_{\text{sum}} - d}{B}\right) \tag{2.1}$$

where A sets the strength of the force, B sets how quickly the force falls off with distance, r_{sum} is the sum of the vehicle half-width and the pedestrian radius, and d is the separation. Forces from all

nearby pedestrians are summed and mapped linearly to a velocity scale factor between 0 and 1.

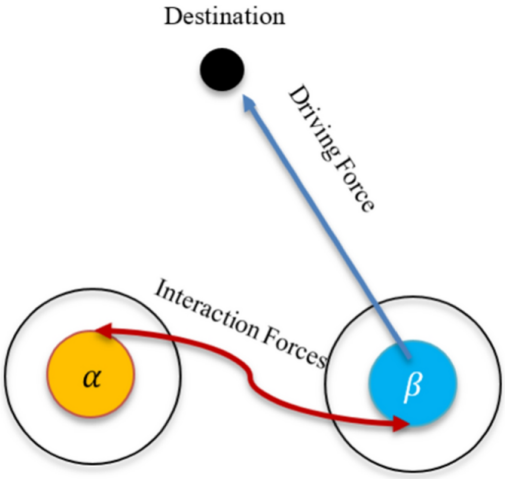

Figure 2.7: Classical social force interaction. The repulsive force decays exponentially with separation distance, producing late, abrupt braking when applied to a 2,050 kg shuttle.

The unmodified formulation produces late, abrupt braking on a 2,050 kg EZ10 [24]. A speed-proportional buffer $d_{\text{eff}} = d - kv_{\text{ego}}$ triggers the force earlier at higher speeds, but creates a feedback loop (Section 3.4): accelerating shrinks d_{eff}, inflates the force, decelerates the vehicle, then reverses. Removing the buffer ($k = 0$) kills the oscillation but restores the late braking, motivating the alternative below.

2.5.2 Crowd-Matching Velocity Model

Pedestrians on UWA campus pathways generally keep left, so counter-flow walkers tend to cross to the opposite side before nUWAy reaches them. nUWAy therefore also keeps left and matches the speed of co-flow traffic. The idea has theoretical support from the macroscopic fundamental diagram [16] and the kinematic-wave model [17], [18]: in a stream, an individual's stable speed is well approximated by the local stream mean.

Each pedestrian within a 15 m radius of nUWAy, in any direction, is classified as co-flow if its forward velocity $v_{\text{co},i} = \mathbf{v}_{ped,i} \cdot \hat{\mathbf{h}}$ exceeds the threshold v_{thresh}. Pedestrians anywhere within the 15 m radius are considered, but only those moving in the same direction as nUWAy count as co-flow. This differs from the proximity component in Section 2.5.3, which only looks ahead. When the co-flow count reaches n_{min}, the target velocity is

$$v_{\text{target}} = \min\left(v_{\text{desired}}, \frac{1}{n}\sum_{i=1}^{n} v_{\text{co},i}\right) \tag{2.2}$$

with $v_{\text{desired}} = 1.39$ m/s (5 km/h). When the co-flow group is lost, the last crowd speed is held for t_{hold} then ramped linearly back to v_{desired}. The target is applied through a first-order smoothing filter $v_{k+1} = v_k + \alpha(v_{\text{target}} - v_k)$ with $\alpha = 1 - e^{-\Delta t/\tau}$ and $\tau = 2.0$ s, so the response time does not depend on the controller's tick rate.

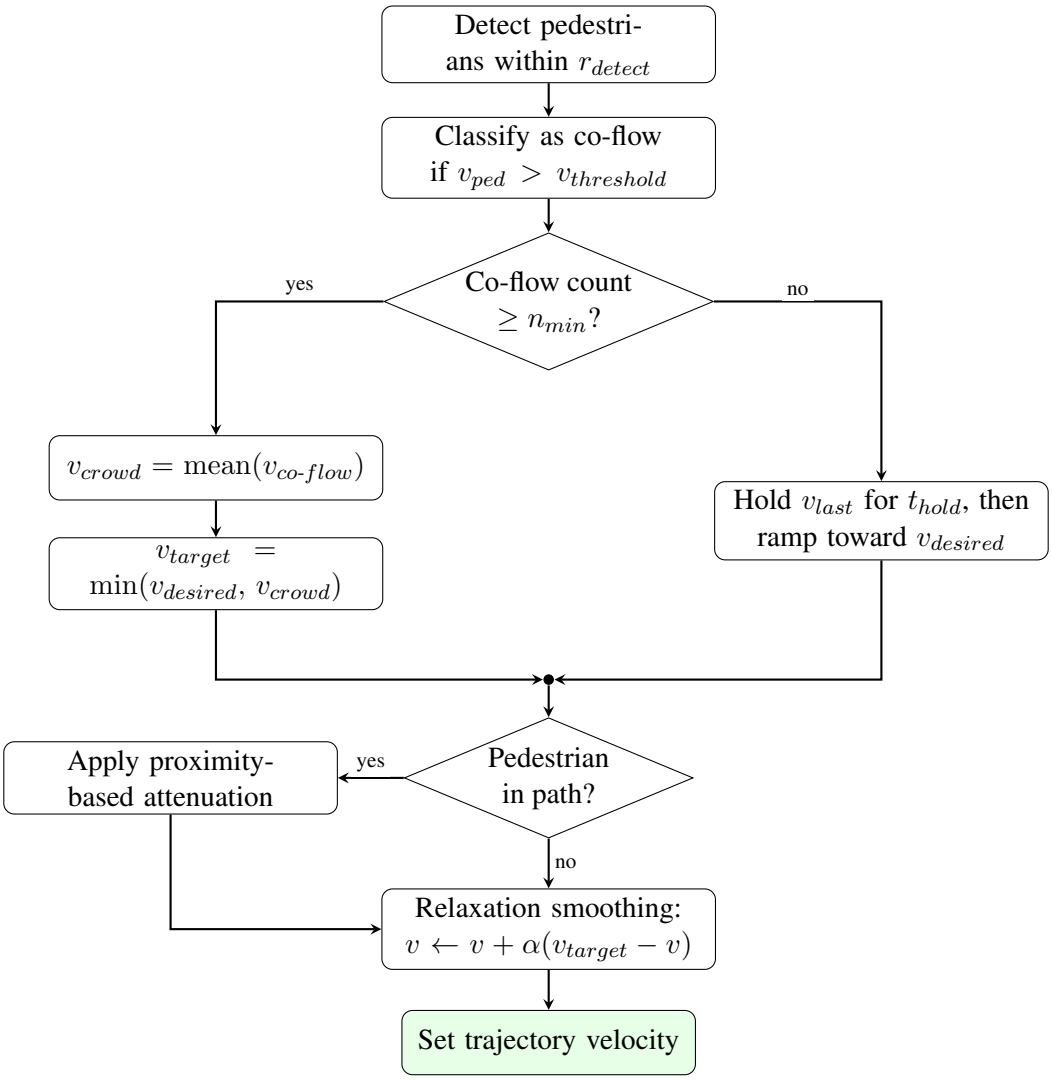

Figure 2.8: Crowd-matching velocity planning algorithm. Detected pedestrians are classified as co-flow or counter-flow. If sufficient co-flow pedestrians are present, the target velocity matches their mean speed; otherwise the previous crowd speed is held for t_{hold} before ramping back to the desired speed. A proximity-based following controller and exponential relaxation smoothing are applied before setting the trajectory velocity.

2.5.3 Proximity-Based Velocity Attenuation

Crowd-matching does not consider the distance to the nearest pedestrian directly ahead. When that pedestrian is slower than the crowd average, or is stationary, the speed must be reduced further. A distance-dependent velocity scale is applied after crowd-matching:

$$s = \text{clamp}\left(\frac{d - d_{\text{min}}}{d_{\text{brake}} - d_{\text{min}}},\ 0,\ 1\right)^2, \quad v_{\text{target}} = v_{\text{crowd}} \cdot s \tag{2.3}$$

with $d_{\text{brake}} = 10\,\text{m}$ onset and $d_{\text{min}} = 3\,\text{m}$ floor. The quadratic scaling produces gentle initial deceleration and stronger attenuation as d decreases.

This formulation is deliberately open-loop: it maps the distance directly to a target velocity, rather than trying to track a target following distance. A closed-loop PID variant was evaluated (Config H, Section 2.6). Delays in the downstream Autoware velocity smoother caused the PID controller to oscillate, with jerk five times that of the open-loop ramp. The Autoware obstacle stop module provides a hard 0.8 m fallback (Figure 2.9).

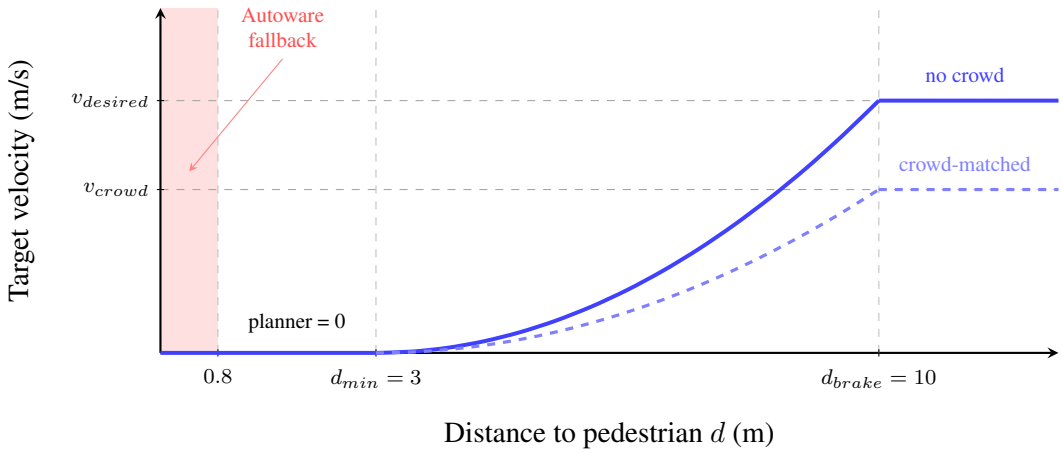

Figure 2.9: Velocity profile produced by the proximity attenuation, shown for the no-crowd regime (solid) and an example crowd-matched regime at $v_{crowd} = 0.9\,\text{m/s}$ (dashed).

2.5.4 Fallback Behaviour

The module sits inside an existing safety stack rather than providing its own (Figure 2.9). Under normal operation, the proximity component reduces commanded velocity from full speed at 10 m to zero at 3 m, applied after crowd-matching but before the smoothing filter. Two fallbacks sit underneath. Autoware's obstacle stop runs downstream at 0.8 m, bypassing the smoothing filter. The EZ10 platform also carries four corner safety LiDARs, hard-wired to trigger an emergency stop that must be manually rearmed [24]. Across the comparative-study scenarios of Sections 3.1–3.7 the maximum deceleration was 1.4 m/s^2, well below the 4.9 m/s^2 ISO 22737:2021 limit [8], and the minimum pedestrian clearance stayed above 2.0 m. The fallback layers under stationary and jumpout stress tests are characterised separately in Section 3.5.

2.6 Comparative Study Design

An eight-configuration comparative study isolates what each component contributes (Table 2.2). A and B are stock Autoware and a parameter-tuned version. C and G evaluate the classical SFM without and with the proximity component. D is the proposed solution. E and F isolate the crowd-matching and proximity branches individually. H replaces the open-loop proximity ramp with a PID distance controller.

Table 2.2: Comparative study design.

Config	Velocity Model	Distance Control	Purpose
A: Stock	None (5 m stops)	N/A	Baseline (stock Autoware)
B: Campus-Tuned	None (1.2 m stops)	N/A	Parameter tuning only
C: Classical SFM	Helbing repulsive	None	Traditional SFM approach
D: Crowd-Match+Prox	Crowd-matching	Proximity atten.	Proposed solution
E: Crowd Only	Crowd-matching	None	Isolate crowd-matching
F: Prox Only	None	Proximity atten.	Isolate proximity atten.
G: Classical+Prox	Helbing repulsive	Proximity atten.	SFM with distance control
H: Crowd+PID	Crowd-matching	PID controller	Closed-loop alternative

Configurations were evaluated on three scenarios: 60 s of straight bi-directional traffic, a 40 s perpendicular crossing, and a density sweep from 0 to 60 simultaneously spawned pedestrians. Metrics span smoothness (standard deviation of velocity, mean jerk), safety (minimum TTC, minimum clearance), and efficiency (distance travelled, stop count). Each configuration was tuned independently across up to five iterations, so observed differences reflect the algorithm rather than poorly chosen parameters. Configuration H was evaluated only on the crossing scenario, where its closed-loop behaviour was most clearly distinguished. Results are presented in Chapter 3.

2.7 Model Validation

On an empty pathway the shuttle held the 5 km/h limit throughout. Across the comparative-study scenarios (Chapter 3) the maximum observed deceleration was 1.4 m/s^2, well below the 4.9 m/s^2 ISO 22737:2021 comfort limit [8]. For nUWAy deployment, a CoflowPanel RViz plugin provides a live on-vehicle readout of the crowd-matched target during supervised driving, and end-to-end rosbag replay (Section 2.4.2) served as an integration check on recorded nUWAy data. The full-stack simulation (Section 2.2) provided a closed-loop check against the Autoware NDT + CenterPoint chain on synthetic traffic.

2.8 Model Limitations

Several limitations are acknowledged. Evaluation is simulation-only, with bus install blocked by a CUDA version mismatch. Test geometries are limited to the straight pathway and one perpendicular

crossing. Curves and intersections were not evaluated. The module controls forward speed only, so faced with an unmapped static obstacle it slows and stops but cannot route around it, and will wait indefinitely. The simulated pedestrians follow strict keep-left walking, a simple approximation rather than a realistic model of side-negotiation, so narrower shared spaces, plazas, and intersections where keep-left does not hold remain outside the validated envelope.

Chapter 3

Results and Discussion

This chapter reports the results of various testing done to evaluate the Crowd-Matching controller. The chapter starts with an assesment of the controller against alternative controllers outlined in Section 2.6, using the comfortability and safety metrics discovered in the literature. Following this, there will be further analysis into Config D's effectiveness specifically, as well as a review of why the SFM controller failed. Finally, a discussion of the results compares the module against the state of the art, and acknowledges limitations.

3.1 Velocity Smoothness

Ride smoothness is quantified by the standard deviation of forward velocity and the mean absolute forward jerk, compared against the comfort thresholds in ISO 22737 [8] and Elbanhawi et al. [9]. All data was collected from the minimal stright path simulation environment.

Table 3.1: Velocity smoothness and stop behaviour, straight pathway (60 s, mixed bi-directional pedestrian traffic).

Configuration	Avg vel. (m/s)	StdDev (m/s)	Avg $\lvert j \rvert$ (m/s^3)	Stops	Stopped (%)
A: Stock	0.425	0.404	1.056	3	36.4
B: Campus-Tuned	0.947	0.138	0.846	0	0.0
C: Classical SFM	0.810	0.163	0.685	0	0.0
D: Crowd-Match+Prox	**0.882**	**0.048**	**0.155**	**0**	**0.0**
E: Crowd-Match Only	0.962	0.159	0.425	0	0.0
F: Proximity Only	0.855	0.107	0.104	0	0.0
G: Classical+Prox	0.974	0.166	0.515	0	0.0

Configuration D (dark green in Figure 3.1) displays visually consistent velocity, achieving a velocity StdDev 8.4 times lower than stock (0.048 vs. 0.404 m/s) and a mean jerk 6.8 times lower (0.155 vs. 1.056 m/s^3, Table 3.1). Stock A displays erratic motion, halting three times and standing still for a large portion of the run. Configuration C exhibits characteristic oscillation - analysed further in Section 3.4. Configuration D's mean jerk of 0.155 m/s^3 sits an order of magnitude below the 2.6 m/s^3 threshold from Elbanhawi [9], however its 3.1 m/s^3 peak jerk exceeds it, this is consistent with all other controllers however, potentially revealing an issue in the way this was calculated or a deeper simulation issue.

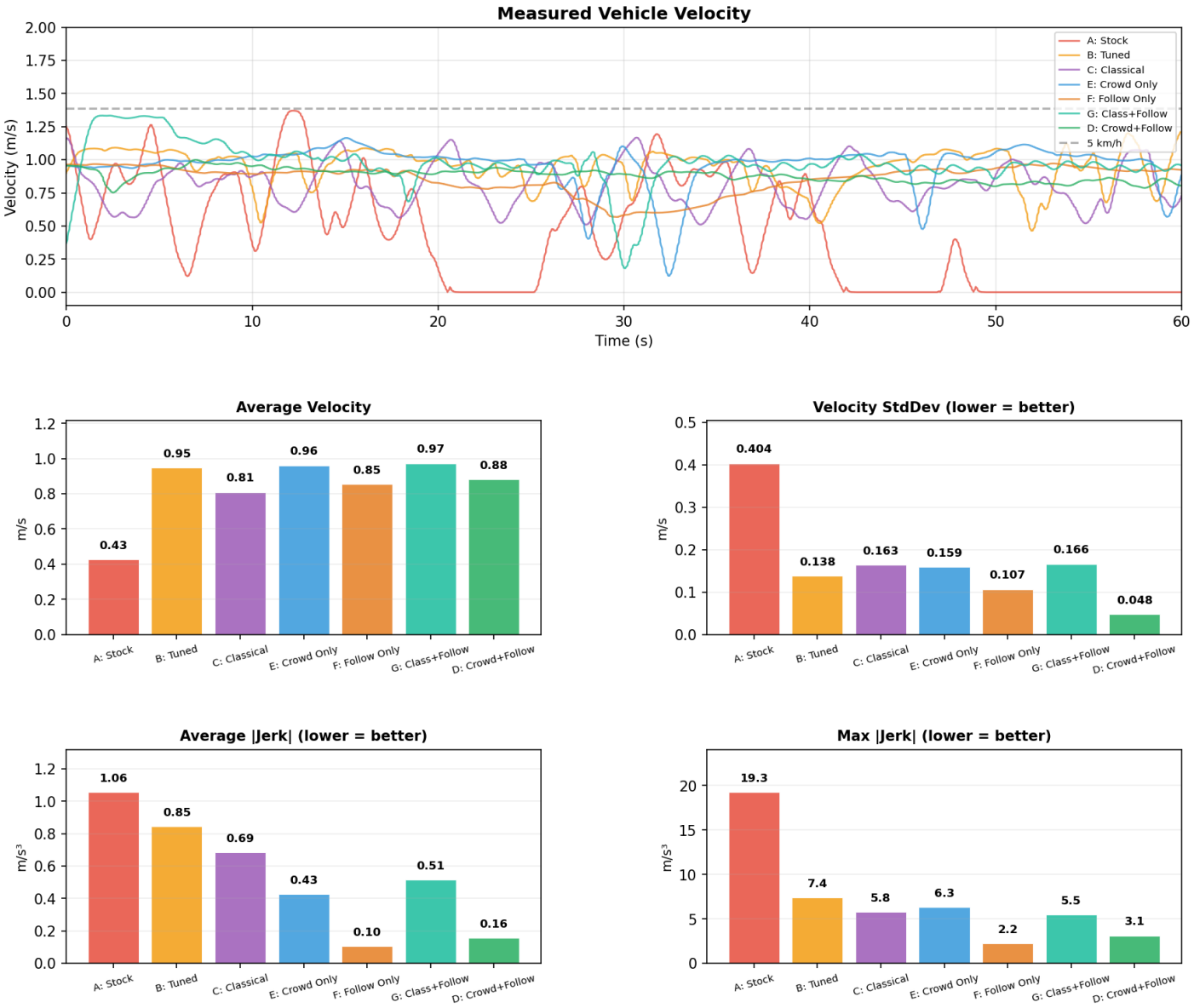

Figure 3.1: Smoothness summary across configurations: measured velocity profiles (top), and per-configuration averages of velocity, velocity StdDev, and mean and peak $|j|$.

On the crossing scenario (Table 3.2), where a secondary bi-directional stream of pedestrians crosses the path, Configuration D again records the lowest StdDev (0.121 m/s) and mean jerk (0.261 m/s^3) of any configuration, at the highest average velocity (0.959 m/s). The SFM baseline (C) and the closed-loop PID variant (H) fall back to stock-like behaviour, registering six and five stops respectively. This is the experimental evidence for the open-loop choice in Section 2.5.3.

Table 3.2: Velocity smoothness and stop behaviour, crossing scenario (40 s, perpendicular pedestrian crossing during straight pathway traffic).

Configuration	Avg vel. (m/s)	StdDev (m/s)	Avg $\lvert j \rvert$ (m/s^3)	Stops	Stopped (%)
A: Stock	0.225	0.329	0.367	1	64.9
B: Campus-Tuned	0.444	0.513	0.571	5	50.7
C: Classical SFM	0.454	0.543	1.082	6	47.6
D: Crowd-Match+Prox	0.959	**0.121**	**0.261**	**0**	**0.0**
E: Crowd-Match Only	0.845	0.161	0.347	0	0.0
F: Proximity Only	0.905	0.159	0.385	0	0.0
G: Classical+Prox	0.961	0.196	0.583	0	0.0
H: Crowd-Match+PID	0.480	0.556	0.844	5	46.3

3.2 Safety Metrics

Safety is quantified by two key parameters, the distance to pedestrians, as well as the time to collision (TTC). The min and mean values of these parameters were obtained from 5 tests in the minimal stright path simulation environment, and are shown in Figure 3.2 The key result was that the stock baseline (A) is the most dangerous configuration on every reported metric.

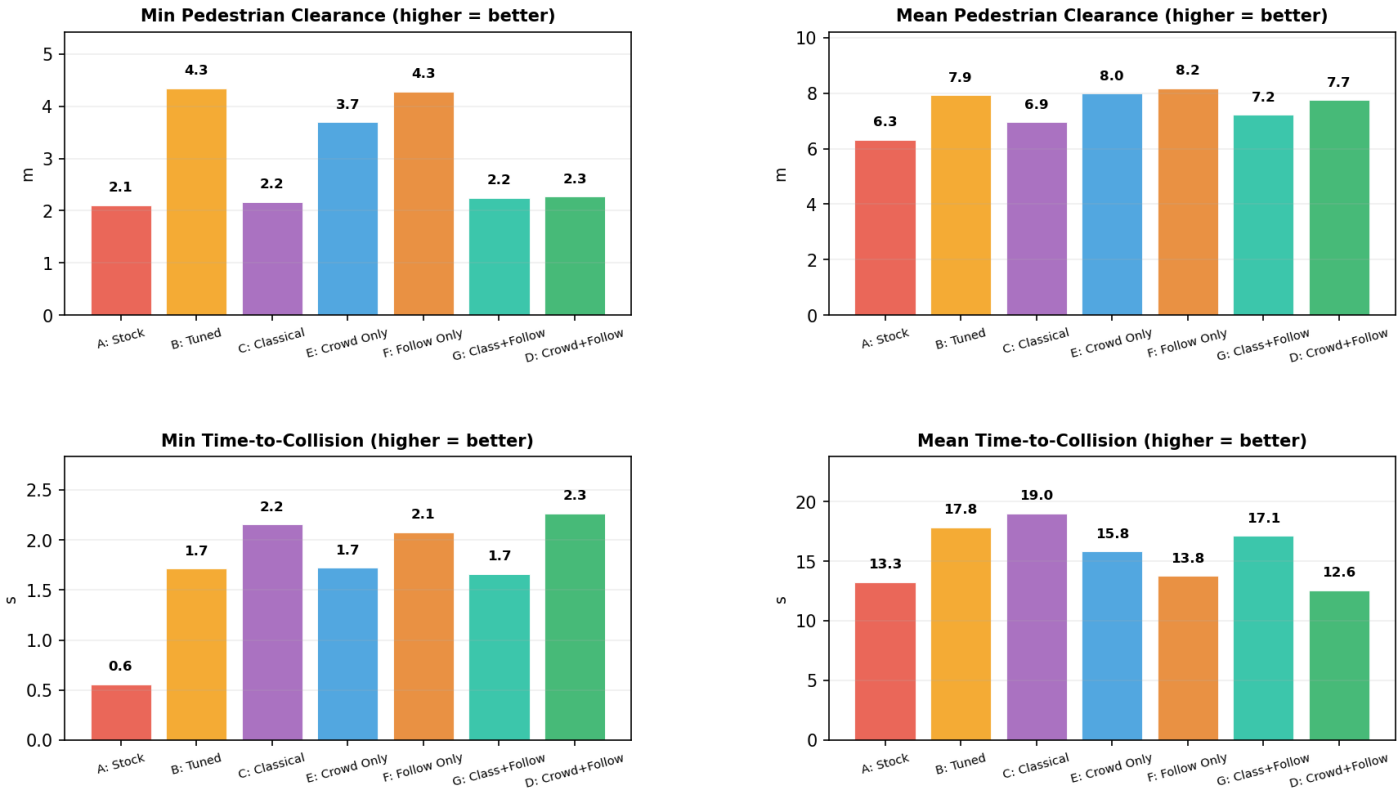

Figure 3.2: Safety summary across configurations on the straight pathway: minimum and mean pedestrian clearance, and minimum and mean time-to-collision.

The pedestrian clearance metric reveals interesting comparisons. Configuration D's minimum pedestrian clearance of 2.3 m was lower compared to the other controllers, potentially due to one - off pedestrian interaction events that could potentially be ruled out with more testing, however for now this can be seen as a limiatation of the model. Its mean pedestrian clearance of 7.7 m however is on the higher end, strengthening the previous one-off claim. It is clear to see however that Configs B,E, and F score best here, with consistently high values across both charts. Scince configs E and F are simply the components of config D, this raises questions as to whether config D could be tuned better, or if a dynamic weighting towards either component, depending on the situation, would be beneficial. In terms of the TTC values, config D scored the best on the minimum time to collision graph, displaying the highest value of 2.3 s. However, it had the lowest mean TTC value of 12.6 s. Initially this may seem to detract from the model's reliability, however since the value is above reasonable limits, this can instead be considered an efficiency metric, indicating continuous engagement with the pedestrian group in front of the bus, which is supported by the config's smooth velocity chart in figure 3.1. The ability of the controller to track pedestrian speed is further evaluated in Section 3.3

3.3 Crowd-Speed Tracking Accuracy

The crowd-matching velocity model heavily relies on per-pedestrian velocity estimates from the Autoware multi-object tracker, any errors will carry through as a velocity error on nUWAy that would have to be counteracted by the velocity attenuator, which is suboptimal and would undermine the whole idea of crowd matching as the velocity driver philosophy. As a result of this, testing was done to verify the Autoware tracker by validating its outputs against AWSIM ground truth pedestrian positions in the full-stack environment.

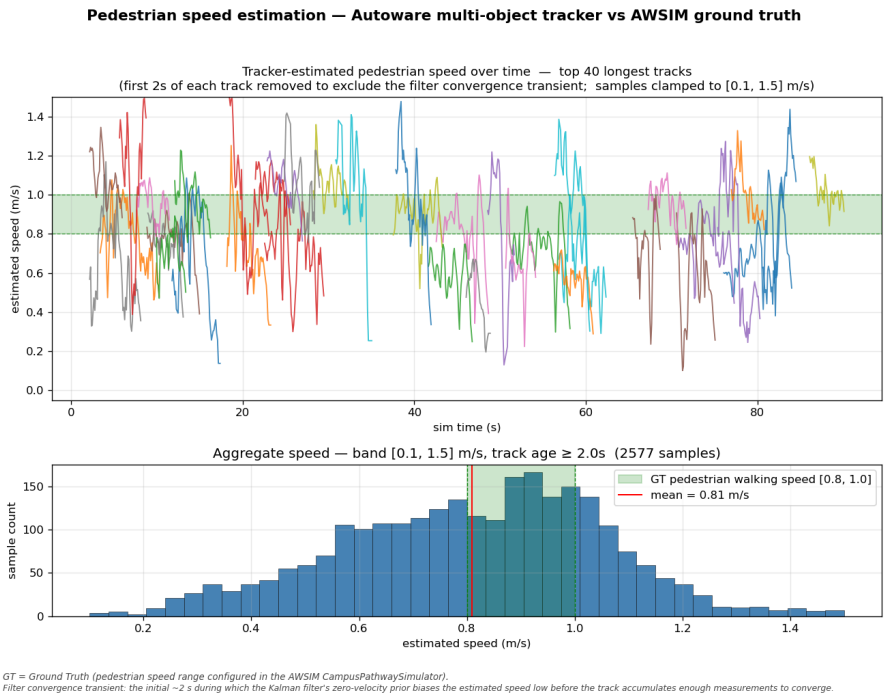

Figure 3.3: Autoware multi-object tracker pedestrian speed estimates against AWSIM ground truth, full-stack environment.

Across the 40 longest tracks of a 90 s straight run, the estimated walking-speed distribution has an average of 0.81 m/s, which is on the lower end of the 0.8-1.0 m/s evenly distributed ground truth (Figure 3.3). Individual tracks tend to approach the 0.8–1.0 m/s band within ≈2 s of first detection, this 2 second warm-up is a result of the object trackers' Kalman filter's initial convergence period, which has been excluded from the per-track plot (seen as a line quickly rising from zero to the settled value). The overall conclusion of this analysis is that the tracker tends to slightly under-estimate the actual crowd speed.

The pedestrian velocities in Figure 3.3 are visibly noisy. This is counteracted by both the minimum 2 simultaneously detected pedestrians requirement before crowd matching initiates, as well as the planner's $\tau = 2.0$ s relaxation that suppress the residual jitter, allowing for the smooth 0.048 m/s StdDev observed in Configuration D's straight profile in Section 3.1.

Following this analysis on the input side, a study of how well the velocity controller actually tracks the crowd-mean was conducted. Four trials were done in the minimal environment on the straight path with 10 pedestrians (indicative of real campus environment pedestrian count within 15m - see Section 3.6), with the bus velocity and ground truth average pedestrian velocity travelling in the same intended direction being recorded, as seen in Figure 3.4.

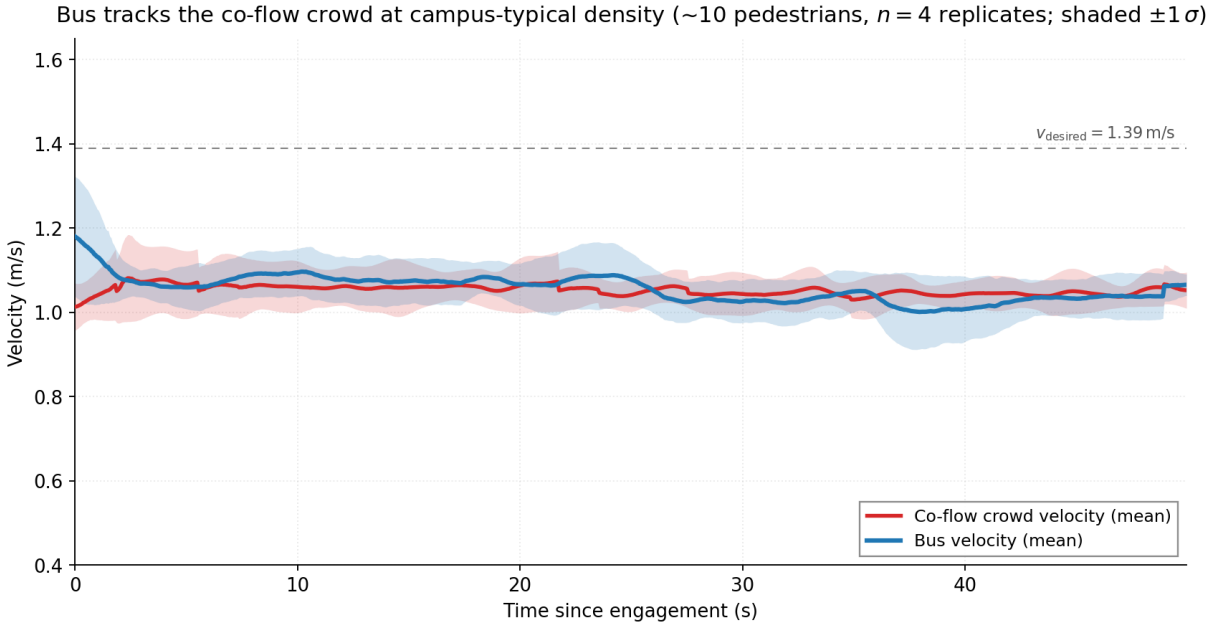

Figure 3.4: Configuration D crowd-tracking ability

The figure displays that the controller settles to the co-flow mean after roughly 2s of engagement (the rate set by τ), and after that the two traces overlap through the remainder of the run, with their standard-deviation bands fully overlapping. Interestingly the bias is slightly positive, which is contrast to the previous experiment which revealed that the Autoware object tracker underestimated the pedestrian velocity. An explanation for this is potentially that the proximity component introduces relative positive crowd tracking biases through transient events, but this would need to be tested.

3.4 SFM Failure Analysis

The regular oscillation visible in Configuration C's velocity profile (Figure 3.1, isolated in Figure 3.5) is a direct result of the speed-dependant safety-buffer feedback loop derived in Section 2.5.1. The buffered separation $d_{\text{eff}} = d - kv_{\text{ego}}$ modifies the social forces dependant on the bus speed (v_{ego}), so increasing v_{ego} shrinks d_{eff}, inflates the force, and decelerates nUWAy, which then reverses the cycle as the forces are now lower due to v_{ego} now decreasing. At $v_{\text{ego}} = 1.39\,\text{m/s}$ and $k = 2.0$, the speed-proportional term reduces the effective separation by 2.78 m, which is comparable to inter-pedestrian distances in the scene, so the loop sees a strong signal each cycle. This characteristic is the reason for its 0.685 m/s^3 mean jerk, more than four times Configuration D's at a comparable average velocity.

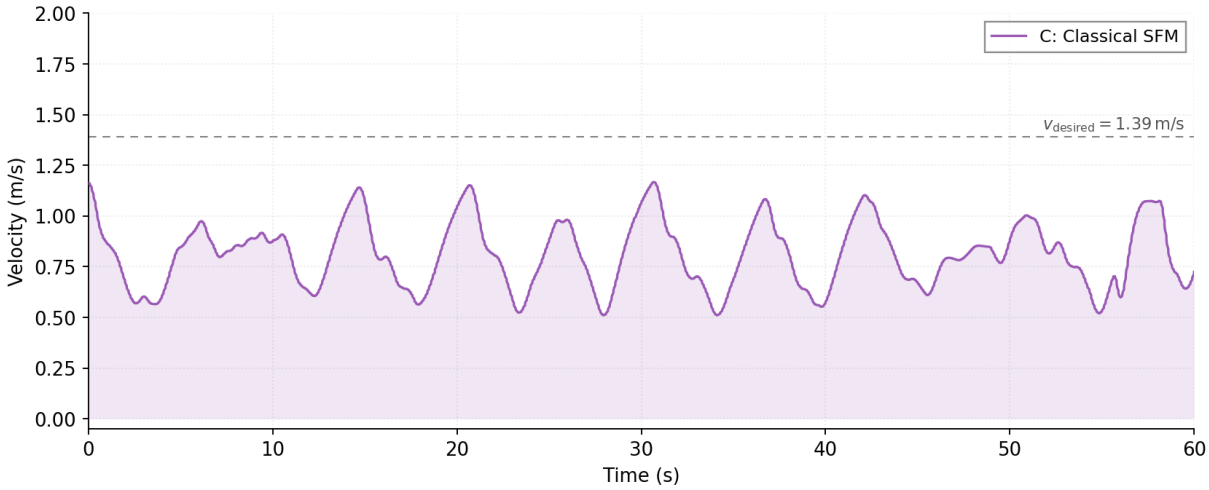

Figure 3.5: Configuration C (Classical SFM) measured velocity on the 60 s straight pathway.

It is worth noting that this failure is built into the model rather than being a tuning artefact. Five sweep iterations across A, B, k, force saturation, and τ produced no parameter set that suppressed the oscillation while preserving forward progress. Adding the proximity component does not cure it either: Configuration G's StdDev (0.166 m/s) is of simlar magnitude to config C's, which places the failure firmly in the velocity model rather than the absence of distance control.

Crowd-matching avoids the loop entirely, Equation 2.2 computes the target from the co-flow group's velocity alone, so v_{ego} never enters the feedback path. The contrast between the C/G and D/E pairs in Table 3.1 confirms this experimentally.

3.5 Fallback Behaviour

Two scenarios were run that stress-test the proximity component and the downstream obstacle stop fallback in isolation to ensure adequate safety of the module. The first scenario placed a single stationary pedestrian on the path 30 m ahead of nUWAy. The second scenario spawned a pedestrian in front of the bus after it had reached cruising speed ($v_{\text{spawn}} = 1.39$ m/s) at bumper-to-ped distances of 5, 8, or 15 m ahead of nUWAy. All clearances are reported as bumper-to-pedestrian distance.

The stationary test (Figure 3.6) verifies that the proximity ramp brings nUWAy to rest at the $d_{\text{min}} = 3$ m floor. With no co-flow pedestrians in the scene the bus accelerates to the $v_{\text{desired}} = 1.39$ m/s speed limit, cruises at that speed until the stationary pedestrian enters the $d_{\text{brake}} = 10$ m bumper-to-ped onset, and the quadratic ramp of Equation 2.3 produces smooth deceleration over the 7 m envelope. The bus settles at a bumper-to-pedestrian clearance of 3.11 m, 0.11 m above the ramp floor and well clear of the 0.8 m obstacle stop margin. A small velocity wobble of $\approx$0.2 m/s appears between 3.5 m and 3.1 m as the bus comes to rest, this wobble persists at zero co-flow density and is likely a result of the velocity smoother and low-speed controller dynamics, rather than to the proximity ramp itself, since the debug log shows the commanded velocity staying near zero throughout this region.

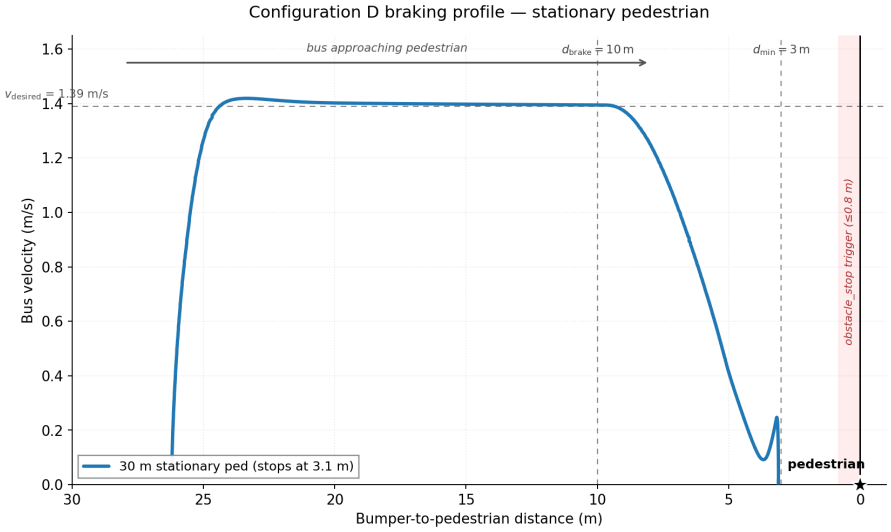

Figure 3.6: Stationary obstacle braking profile for Configuration D.

The jumpout tests (Figure 3.7) characterise the proximity ramp's response to a pedestrian appearing inside its detection envelope while nUWAy is at cruise. At a 15 m spawn the ramp has the full $d_{\text{brake}} - d_{\text{min}} = 7$ m available to decelerate from cruise, and the bus settles at a minimum bumper-to-ped clearance of 3.13 ± 0.01 m, matching the $d_{\text{min}} = 3$ m floor of Equation 2.3 to within 1 cm. This is the design intent of the quadratic ramp: smooth deceleration converging to zero velocity at the floor, leaving the 0.8 m obstacle stop layer unused. The 8 m performs similarily, just with a slightly steeper velocity ramps down to zero to accomodate for the slightly later braking onset. The 5 m spwan distance causes the bus to overshoot below the 3m quadratic attenuator floor. Peak jerk grows in the emergency-stop regime, reaching 23.3 m/s^3 at 5 m, an order of magnitude above the 2.6 m/s^3

Elbanhawi threshold [9], expected as this case can be considered an emergency stop.

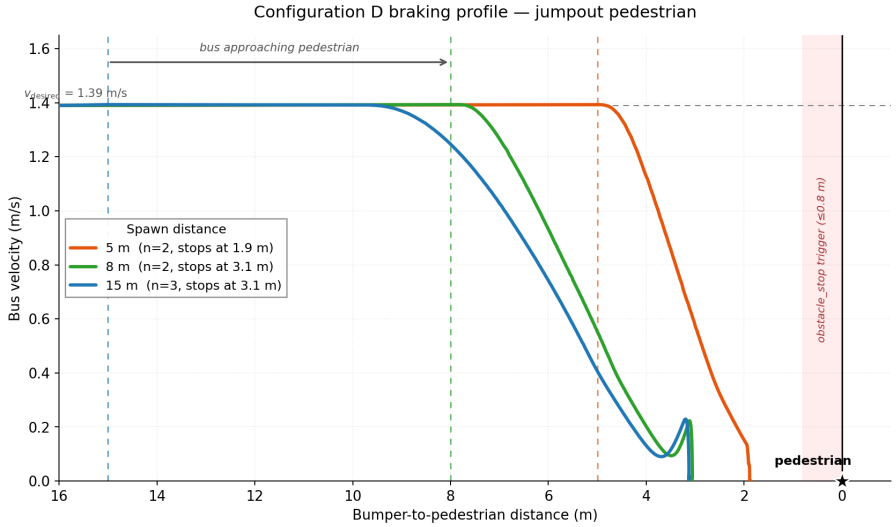

Figure 3.7: Jumpout braking profile for Configuration D.

3.6 Campus Pedestrian Density Distribution Analysis

To obtain standard pedestrian densities that the model would have to navigate on the UWA Crawley campus. A recording of sensor data during a lunchtime bus run was replayed via a rosbag, essentially emulating the sensor data for direct input to the full Autoware stack on a different off-site computer.

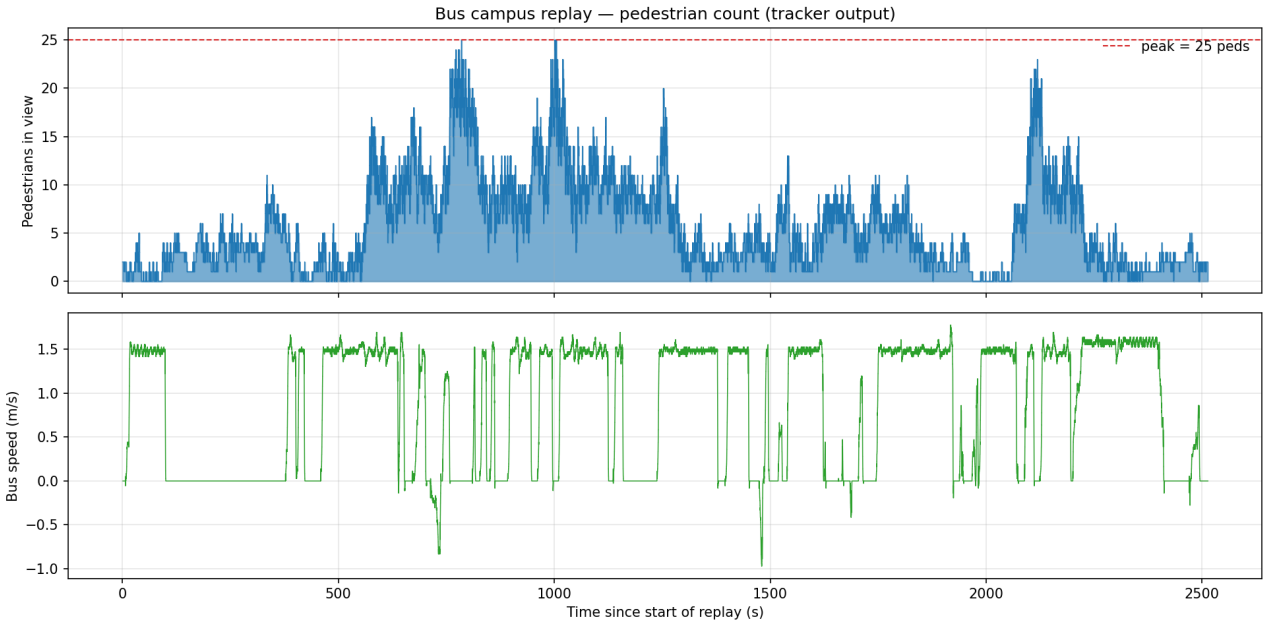

Figure 3.8: Tracked pedestrian count and nUWAy longitudinal velocity

Figure 3.8 reports tracked pedestrians against time alongside nUWAy's velocity profile. There is evidently significant change in pedestrian density depending on where on the route the bus was. With some sections spiking to up to 25 pedestrians simultaneously within 15m of the vehicle. It is also

clear to see that nUWAy's velocity drops to zero during the busier periods, where the operator needed to yield to the dense foot traffic.

This same data is also displayed on an OpenStreetMap of the campus and bus route. The high pedestrian count locations are around Reid library, near the student guild, and at the Barry J. Marshall library, as expected.

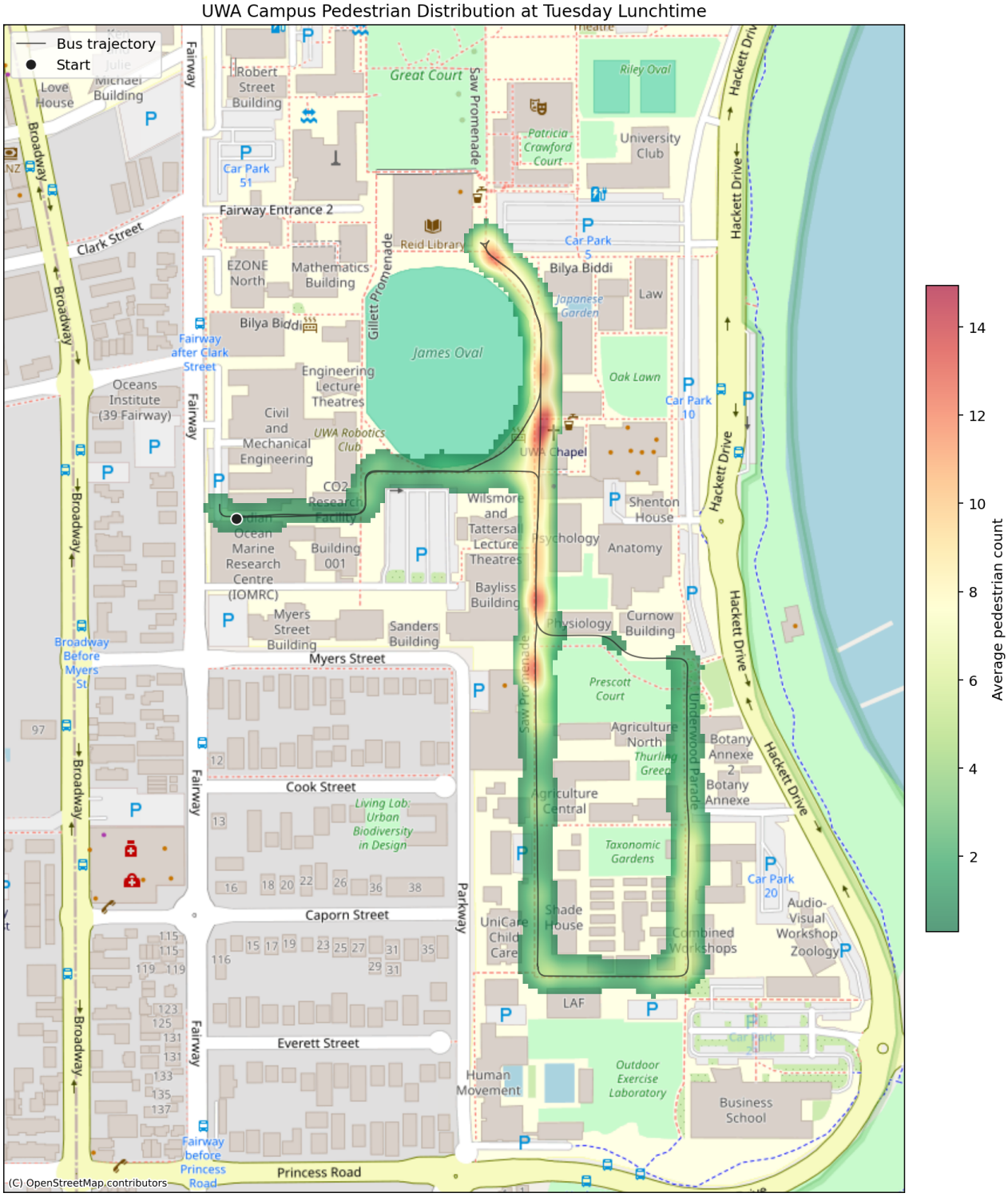

Figure 3.9: Pedestrian density on an OpenStreetMap basemap of UWA Crawley.

3.7 Density Sweep

Following the Campus pedestrian density evaluation, a density sweep was then done across four pedestrian densities (0, 10, 30, and 60) on the straight pathway in the lightweight simulation environment of Section 2.2. Five trials were run, with the purpose of observing how gracefully the crontroller slowed vehicle movement as the pathway became more crowded.

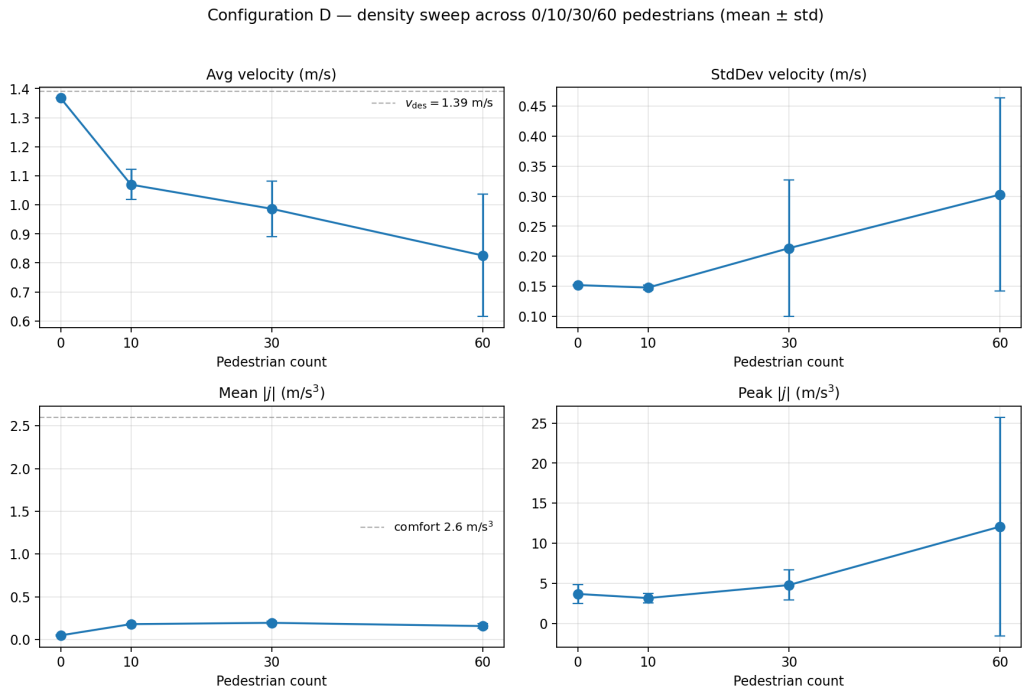

Figure 3.10: Density sweep for Configuration D across 0, 10, 30, and 60 pedestrians.

Three observations follow from this data. First, the zero-pedestrian trial tracks the 1.39 m/s desired velocity within 2 % (mean 1.37 m/s), confirming that the crowd-matching gate cleanly defaults to free-flow speed when no co-flow group is present. Second, average velocity is observed to gracefully decline with increasing density (1.37, 1.07, 0.99, 0.83 m/s) as the crowd-mean target aligns to the slower pedestrian flow (explaned by Greenshields [16]), meaning the controller correctly tracks the stream's average speed. Third, smoothness metrics remain inside the comfort envelope across the full range, with mean jerk peaking at 0.20 m/s^3 at 30 pedestrians, an order of magnitude below the 2.6 m/s^3 Elbanhawi threshold [9]. It is clear to see however that the velocity standard deviation grows with density, where the StdDev-velocity error bar at 60 pedestrians (0.30 ± 0.16 m/s) is wider than the cell mean itself, indicating either the controller operating at the upper edge of its usable region, or a simulation artifact where the crowd velocity itself widely started to ossscilate, which the bus would have simply tracked. However since this 60 pedestrian density sits well above the average campus pedestrian density established in Section 3.6, this is not a concern for normal operation, and is instead useful in showing where the model begins to lose effectiveness.

3.8 Discussion

This section interprets the results, positions the module against prior art (Chapter 1), and identifies the limitations the results expose.

3.8.1 Result Interpretation

The headline contrast is between Configurations A and D on the straight pathway. Stock Autoware halts three times in 60 s and stands still for 36.4% of the run, while the proposed module flows continuously, achieving an 8.4-fold StdDev reduction and 6.8-fold jerk reduction (Table 3.1). This order-of-magnitude difference is significant, rather than incremental, separating a vehicle that cannot operate on a shared pathway from one that flows with pedestrian traffic, in the same way pedestrian-to-pedestrian flow self-organises into co-moving streams [11], [12]. The closed-loop PID variant (H) fails on the crossing geometry, registering five stops where D registers zero, which is the experimental verdict on the open-loop choice in Section 2.5.3 and locates the failure in the closed-loop branch rather than in crowd-matching itself.

3.8.2 Comparison with State of the Art

When compared against the literature, the social force model [11] produces the oscillation predicted in Section 2.5.1. The same diagnosis explains why Yang and Özgüner's MPC regulator [22], which keeps pedestrians as repulsive agents inside its cost function, must lengthen its prediction horizon and smoothness weights to suppress the same loop. Crowd-matching takes a different framing by treating the surrounding co-flow as a slow-moving stream rather than a swarm of point-obstacles, which is supported by the macroscopic flow literature [16]–[18] where the same speed–density relationship has been characterised for vehicular and pedestrian traffic. The closest precedent is the network-level MFD control of Geroliminis and Daganzo [20], however that operates at the city scale rather than at a single vehicle. Configuration D's 0.155 m/s^3 mean jerk sits an order of magnitude below the 2.6 m/s^3 Elbanhawi threshold [9], and its 1.4 m/s^2 maximum deceleration sits well within the 4.9 m/s^2 ISO 22737 comfort ceiling [8].

3.8.3 Practical Implications

The module is built as a behaviour-velocity plugin (Section 2.5), which keeps the surrounding Autoware stack untouched beyond the preset and leaves obstacle stop as a hard backup. The algorithm assumes only that the vehicle is travelling along a routed path with co-directional pedestrians and that those pedestrians have velocity estimates from the perception stack. Use cases therefore extend beyond campus pathways to airside ground-support vehicles, hospital corridors, pedestrianised retail streets, and factory AGVs. The set of tunable parameters is small (v_{desired}, v_{thresh}, $n_{\min}$, t_{hold}, τ, d_{brake}, $d_{\min}$), meaning the module can be retuned for a new deployment without restructuring the algorithm itself, and the CoflowPanel RViz plugin gives the safety operator a live readout of the crowd-matched target and co-flow count to ensure the module is functioning properly.

3.8.4 Limitations

Several limitations are acknowledged. The headline numbers were obtained in the lightweight simulation environment, which exercises a much more minimal perception path than the full pipeline. Live-vehicle validation is currently blocked by the CUDA mismatch noted in Section 2.8, however nUWAy's CenterPoint + label-remap chain catches nearly every pedestrian geometrically visible to the LiDAR on campus rosbag replays (Section 3.6). The test geometry is also restricted to a straight pathway and one perpendicular crossing, meaning on a sharp bend the co-flow classifier would need to be redefined relative to the path tangent rather than the vehicle heading. Counter-flow pedestrians are handled only by the proximity ramp and the keep-left lanelet, which is adequate on a 6 m campus pathway but becomes a stronger limitation in narrower shared spaces where lane discipline cannot be assumed. Finally, the module slows to a stop in front of an unmapped static obstacle and waits indefinitely without re-routing, which is a limitation inherited from the simplification of the behaviour-velocity stage for campus usage, rather than from the module itself.

3.8.5 Arbitrary Choices and Alternatives

Several design parameters were settled by iterative tuning rather than first-principles derivation. $v_{\text{thresh}} = 0.3$ m/s rejects near-zero estimates from short-lifetime tracks while retaining slow walkers. $n_{\text{min}} = 2$ treats a single match as noise, since the proximity ramp already handles lone walkers. $\tau = 2.0$ s balances responsiveness against comfort. $d_{\text{brake}} = 10$ m and $d_{\text{min}} = 3$ m place perceptible deceleration $\approx$7 s before a hypothetical collision at 1.39 m/s. The quadratic exponent was preferred over linear after the linear version produced an uncomfortable jerk step partway through the ramp. A replicated 3×3 sweep over $\tau \in \{1.0, 2.0, 3.0\}$ s and $d_{\text{brake}} \in \{6, 10, 14\}$ m (Appendix A.3, Table A.3) shows that StdDev velocity is comparable across the central plateau and that within-cell run-to-run variance dominates between-cell differences. Initially this may seem to suggest the baseline is suboptimal, however it instead indicates that the baseline settings sit inside a robustness plateau rather than around a sharp optimum, meaning the module is forgiving of small parameter changes rather than brittle to them. The open-loop ramp was chosen over closed-loop PID after Configuration H produced five times the jerk on the crossing scenario, which is the experimental basis for the design decision in Section 2.5.3. The main alternative not pursued here is a short-horizon model-predictive velocity planner using a crowd-speed forecast [21], [22], which is recommended as future work.

Chapter 4

Conclusions and Future Work

4.1 Conclusions

This thesis presents a pedestrian-aware velocity planner for autonomous shuttle navigation in shared spaces, developed and evaluated for the nUWAy EZ10 on the UWA Crawley campus. Built as an Autoware behaviour-velocity plugin, it combines a crowd-matching velocity model that sets the desired speed to the mean forward speed of co-directional pedestrians inside a 15 m radius (Section 2.5.2) with an open-loop quadratic ramp that reduces the commanded velocity as the bumper-to-pedestrian distance to the nearest in-path pedestrian closes from 10 m to 3 m (Section 2.5.3).

All six project objectives of Section 1.3 were addressed. Autoware was integrated on nUWAy during GENG5511. The AWSIM simulation environment, pedestrian simulator, object-relay pipeline, and rosbag analysis pipeline were built. The proposed planner and a classical-SFM baseline were implemented as plugins. The eight-configuration comparative study was conducted, and the operating envelope characterised across a 0–60 pedestrian density sweep informed by a campus-rosbag replay that recorded peaks of 25 simultaneous pedestrians within 15 m of the vehicle during the lunchtime between-class period (Section 3.6). Stationary and jumpout fallback stress tests then showed that the proximity ramp brings nUWAy to rest at the $d_{\text{min}} = 3$ m floor to within 11 cm for a stationary obstacle at 30 m and to within 1 cm for a 15 m jumpout when given its full braking envelope, with the boundary at 5 m and 8 m jumpouts characterised separately as the region where the downstream obstacle stop layer becomes the active safety margin. The headline finding is that the proposed module reduces velocity standard deviation 8.4-fold and mean absolute jerk 6.8-fold relative to stock Autoware on the straight pathway, while simultaneously holding the highest minimum time-to-collision (2.3 s) of any configuration tested and a minimum pedestrian clearance above 2.0 m, with zero stops on both straight and crossing scenarios. The controller settles to the co-flow mean within roughly 2 s of engagement and tracks it for the remainder of the run (Section 3.3), confirming the crowd-matching premise in closed-loop operation. A replicated 3×3 sweep over (τ, d_{brake}) in Appendix A.3 places the baseline inside a robustness plateau rather than at a sharp optimum, indicating the module is forgiving of small parameter changes rather than brittle to them.

Many significant findings surface. First, the crowd-matching model treats the surrounding co-flow as a slow-moving stream rather than a swarm of point-obstacles, a framing supported by the macroscopic flow literature of Greenshields and Lighthill–Whitham–Richards, and an alternative to the prevailing social-force-as-obstacle approach. Second, the classical SFM-as-controller formulation was shown

to be intrinsically unstable for forward-speed control: the speed-proportional safety buffer creates a feedback loop that sustained oscillation across five parameter sweeps (Section 3.4), placing the failure in the model rather than in tuning, and explaining why the MPC regulator of Yang and Özgüner [22] must lengthen its horizon and smoothness weights to suppress the same loop. Third, the proximity ramp is open-loop by design: the closed-loop PID variant (Configuration H) registered roughly five times the jerk of the open-loop ramp and five stops on the crossing scenario where the open-loop ramp registered zero (Section 3.8.5), locating the failure in the closed-loop branch rather than in crowd-matching itself. Fourth, the controller is a drop-in plugin to Autoware's behaviour-velocity stage and sits inside an explicit dual-layer safety stack (proximity ramp plus downstream obstacle stop fallback) whose boundary of operation was mapped under stress rather than inferred, making the controller portable to any vehicle running the same architecture without reopening the safety case. Together these constitute the principal contribution: a deployable forward-speed planning module, evaluated against ISO 22737 and the everyday-driving comfort literature, that lets a low-speed shuttle flow with, rather than disrupt, pedestrian traffic.

4.2 Future Work

The most immediate follow-on is sim-to-real transfer. Live deployment is blocked by the CUDA-version mismatch noted in Section 2.8. Once resolved, the next step is a supervised-driving campaign on the Crawley pathway with the CoflowPanel tool displayed on the safety operator's screen. The simulation-tuned parameters of Appendix A should be ported unchanged, and the same smoothness, safety, and stop-count metrics of Chapter 3 should be recorded under three increasing levels of difficulty: an empty-pathway baseline, sparse co-flow traffic, and the worst-case 25-pedestrian envelope observed during the between-class transition in the nUWAy replay of Section 3.6. The main variable likely to limit transfer is not algorithm recall (the production CenterPoint + label-remap chain catches nearly every pedestrian geometrically visible to the LiDAR on the replay) but LiDAR range and occlusion at the sparse VLP-16 beam pattern beyond 20 m, against which the $n_{\min} = 2$ gate should be re-examined.

The planner should then be evaluated on the geometries omitted here: curved sections, T-intersections, and the marked pedestrian crosswalks on the central campus loop. The co-flow classifier currently uses vehicle heading as its reference direction. On a sharp bend this misclassifies pedestrians walking along the corridor as counter-flow during the turn. Replacing the reference direction with the local path tangent (already used by the elastic-band smoother) closes this gap without introducing new state. A formal parameter-sensitivity study should extend the 3×3 sweep over (τ, d_{brake}) reported in Appendix A.3: that grid identifies a robustness plateau but is limited in replicate count and parameter coverage.

A more extensive future work scope is a model-predictive successor architecture [21], [22]: the crowd-matched velocity becomes a soft tracking target inside the MPC cost, and the proximity component becomes a hard constraint on minimum separation. This generalises naturally to curved paths and

intersections without reintroducing the closed-loop instability that disqualified Configuration H. Such a formulation could also use a short-horizon crowd-velocity forecast rather than the instantaneous mean used here. Two further extensions arise from the limitations of Section 3.8.4. Multi-vehicle shared-space coordination becomes relevant once a second shuttle enters the pathway: each vehicle's co-flow set would then include the other, and a tiebreak protocol would be needed. A complementary lateral-control study, on how the vehicle picks a side during counter-flow encounters, would relax the keep-left assumption that the forward-speed-only treatment depends on, expanding the operating envelope to plazas and narrower walkways where keep-left does not hold.

Bibliography

[1] U.S. Department of Transportation, "Low-speed automated shuttles: State of the practice final report," Federal Transit Administration, Tech. Rep. FTA-Report-0166, 2020.

[2] M. Golchin, A. Grandhi, N. Gore, S. S. Pulugurtha, and A. Ghasemi, "UNC Charlotte autonomous shuttle pilot study: An assessment of operational performance, reliability, and challenges," *Machines*, vol. 12, no. 11, p. 796, 2024.

[3] F. Pascucci, N. Rinke, C. Schiermeyer, B. Friedrich, and V. Berkhahn, "Collision avoidance of low speed autonomous shuttles with pedestrians," *International Journal of Automotive Technology*, vol. 21, no. 4, pp. 881–892, 2020.

[4] S. Kato, S. Tokunaga, Y. Maruyama, *et al.*, "Autoware on board: Enabling autonomous vehicles with embedded systems," in *Proc. IEEE Int. Conf. on Autonomic Computing (ICAC)*, 2018.

[5] Baidu Apollo Team, *Apollo: An open autonomous driving platform*, `https://github.com/ApolloAuto/apollo`, Accessed 2026-04, 2023.

[6] International Organization for Standardization, *ISO 2631-1:1997 – mechanical vibration and shock – evaluation of human exposure to whole-body vibration*, Standard, 1997.

[7] K. N. de Winkel, T. Irmak, R. Happee, and B. Shyrokau, "Standards for passenger comfort in automated vehicles: Acceleration and jerk," *Applied Ergonomics*, vol. 106, p. 103 881, 2023.

[8] International Organization for Standardization, *ISO 22737:2021 – intelligent transport systems – low-speed automated driving (LSAD) systems for predefined routes – performance requirements, system requirements and performance test procedures*, Standard, 2021.

[9] M. Elbanhawi, M. Simic, and R. Jazar, "In the passenger seat: Investigating ride comfort measures in autonomous cars," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, no. 3, pp. 4–17, 2015.

[10] A. Alessandrini, A. Campagna, P. Delle Site, F. Filippi, and L. Persia, "Automated vehicles and the rethinking of mobility and cities," *Transportation Research Procedia*, vol. 5, pp. 145–160, 2015.

[11] D. Helbing and P. Molnár, "Social force model for pedestrian dynamics," *Physical Review E*, vol. 51, no. 5, pp. 4282–4286, 1995.

[12] M. Moussaïd, D. Helbing, and G. Theraulaz, "How simple rules determine pedestrian behavior and crowd disasters," *Proceedings of the National Academy of Sciences*, vol. 108, no. 17, pp. 6884–6888, 2011.

[13] M. Moussaïd, N. Perozo, S. Garnier, D. Helbing, and G. Theraulaz, "The walking behaviour of pedestrian social groups and its impact on crowd dynamics," *PLoS ONE*, vol. 5, no. 4, e10047, 2010.

[14] F. Pascucci, N. Rinke, and B. Friedrich, "Simulation of pedestrian interaction with autonomous vehicles via social force model," *Simulation Modelling Practice and Theory*, vol. 130, p. 102 871, 2024.

[15] R. Korbmacher and A. Tordeux, "Review of pedestrian trajectory prediction methods: Comparing deep learning and knowledge-based approaches," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 24 126–24 144, 2022.

[16] B. D. Greenshields, "A study of traffic capacity," *Highway Research Board Proceedings*, vol. 14, pp. 448–477, 1935.

[17] M. J. Lighthill and G. B. Whitham, "On kinematic waves II. A theory of traffic flow on long crowded roads," *Proc. R. Soc. Lond. A*, vol. 229, no. 1178, pp. 317–345, 1955.

[18] P. I. Richards, "Shock waves on the highway," *Operations Research*, vol. 4, no. 1, pp. 42–51, 1956.

[19] C. F. Daganzo, "The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory," *Transportation Research Part B: Methodological*, vol. 28, no. 4, pp. 269–287, 1994.

[20] N. Geroliminis and C. F. Daganzo, "Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings," *Transportation Research Part B: Methodological*, vol. 42, no. 9, pp. 759–770, 2008.

[21] M. P. Aslam, B. Derajic, M.-K. Bouzidi, S. Bernhard, and J. O. Ringert, *Model predictive control for crowd navigation via learning-based trajectory prediction*, arXiv preprint arXiv:2508.07079, 2025.

[22] D. Yang and Ü. Özgüner, "Combining social force model with model predictive control for vehicle's longitudinal speed regulation in pedestrian-dense scenarios," in *Proc. IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019.

[23] T. Yin, X. Zhou, and P. Krähenbühl, "Center-based 3D object detection and tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 11 784–11 793.

[24] Land Transport Guru, *EasyMile EZ10*, https://landtransportguru.net/easymile-ez10/, Accessed 2026-04, 2024.

Appendix A

Module Configuration Parameters

This appendix lists the parameter values of the velocity-planning module used in the comparative study of Chapter 3. Symbols are introduced in Chapter 2 and reproduced here together with justifications and the ranges explored during tuning. Values are those of Configuration D unless stated otherwise.

A.1 Crowd-Matching and Proximity Parameters

Table A.1: Velocity-planning module parameters as used in Configuration D. Where a range is shown the value listed was selected after iterative tuning across straight-pathway and crossing scenarios.

Symbol	Description	Value	Range tested	Justification
v_{desired}	Free-flow target velocity	1.39 m/s	fixed	5 km/h campus speed limit, encoded as the lanelet `speed_limit` tag (Section 2.1).
r_{det}	Co-flow detection radius	15 m	10–20 m	Spans the visible co-flow group at walking pace without admitting unrelated traffic at the pathway shoulder.
v_{thresh}	Co-flow speed threshold	0.3 m/s	0.1–0.5	Rejects near-zero velocity estimates from short-lifetime tracks while retaining slow walkers (Section 3.8.5).
	Co-flow hysteresis	0.15 m/s	0.05–0.25	Entry threshold $v_{\text{thresh}} + 0.15$ suppresses chatter around the boundary.
n_{min}	Min. co-flow count	2	1–4	Treats a single match as noise. The proximity ramp handles lone walkers (Section 3.8.5).
t_{hold}	Crowd-speed hold time	1.5 s	0.5–3.0	Prevents return-to-free-flow during transient occlusions of a stable co-flow group.
τ	Relaxation time constant	2.0 s	1.0–4.0	Balances responsiveness against ride comfort. The 3×3 sweep of Section A.3 found smoothness comparable for $\tau \in \{1.0, 3.0\}$ at $d_{\text{brake}} \in \{10, 14\}$ m. The 2.0 s value sits in the middle of this plateau.
d_{brake}	Proximity ramp onset	10 m	6–12 m	Places perceptible deceleration approximately 7 s before a hypothetical collision at v_{desired}.
d_{min}	Proximity ramp floor	3 m	2–4 m	Targets zero commanded velocity at 3 m clearance. The 0.8 m obstacle stop sits beneath this as a backup (Section 2.5.4).
r_{ped}	Pedestrian radius	0.3 m	fixed	Conservative effective collision radius used by both the proposed module and the classical baseline.
$w_{\text{veh}}/2$	Vehicle half-width	1.2 m	fixed	EZ10 platform geometry [24].

A.2 Auxiliary Parameters

The classical social-force baseline (Configuration C) and the closed-loop variant (Configuration H) use the additional parameters listed below. They are reported for reproducibility of the failure modes diagnosed in Sections 3.4 and 3.8.5. They are not active in Configuration D.

Table A.2: Additional parameters for the classical SFM and PID-based comparison configurations.

Symbol	Description	Value	Active in
A	Helbing force magnitude coefficient	2.1	Config C, G
B	Helbing exponential decay scale	0.3 m	Config C, G
k	Speed-proportional safety buffer	2.0	Config C, G
F_{sat}	Force saturation (velocity to zero)	10.0	Config C, G
d^*_{pid}	PID target following gap	5 m	Config H
K_p, K_i, K_d	PID gains	0.15, 0.02, 0.05	Config H

A.3 τ versus d_{brake} Sensitivity Sweep

A 3×3 sweep over $\tau \in \{1.0, 2.0, 3.0\}$ s and $d_{\text{brake}} \in \{6, 10, 14\}$ m was run in the lightweight environment of Section 2.2 with all other Configuration D parameters held fixed. Each cell was repeated up to four times to estimate run-to-run variance, after the engagement-timing fixes described in the deployment notes. Cells with $n < 4$ are those where the diagnostic graph did not enter `is_autonomous_mode_available` within 60 s on at least one replicate and the run was excluded. Table A.3 reports the cell means and within-cell standard deviations. Figure A.1 shows the same grid as a heatmap.

Table A.3: Velocity smoothness and mean-jerk across the $\tau \times d_{\text{brake}}$ grid for Configuration D on the 60 s straight scenario. Values are cell mean ± within-cell standard deviation across n replicates.

τ (s)	d_{brake} (m)	n	Avg velocity (m/s)	StdDev velocity (m/s)	Mean $\lvert j \rvert$ (m/s^3)
1.0	6	3	0.82 ± 0.12	0.31 ± 0.13	0.23 ± 0.02
1.0	10	4	0.83 ± 0.25	0.26 ± 0.16	0.24 ± 0.09
1.0	14	3	0.94 ± 0.07	**0.21 ± 0.06**	0.21 ± 0.04
2.0	6	4	0.79 ± 0.16	0.35 ± 0.13	0.20 ± 0.03
2.0	10	4	0.87 ± 0.17	0.25 ± 0.16	0.23 ± 0.07
2.0	14	2	0.81 ± 0.15	0.29 ± 0.17	0.26 ± 0.07
3.0	6	4	0.88 ± 0.09	0.29 ± 0.15	0.22 ± 0.05
3.0	10	4	0.95 ± 0.04	**0.22 ± 0.05**	0.28 ± 0.08
3.0	14	4	0.85 ± 0.17	0.23 ± 0.14	0.28 ± 0.12

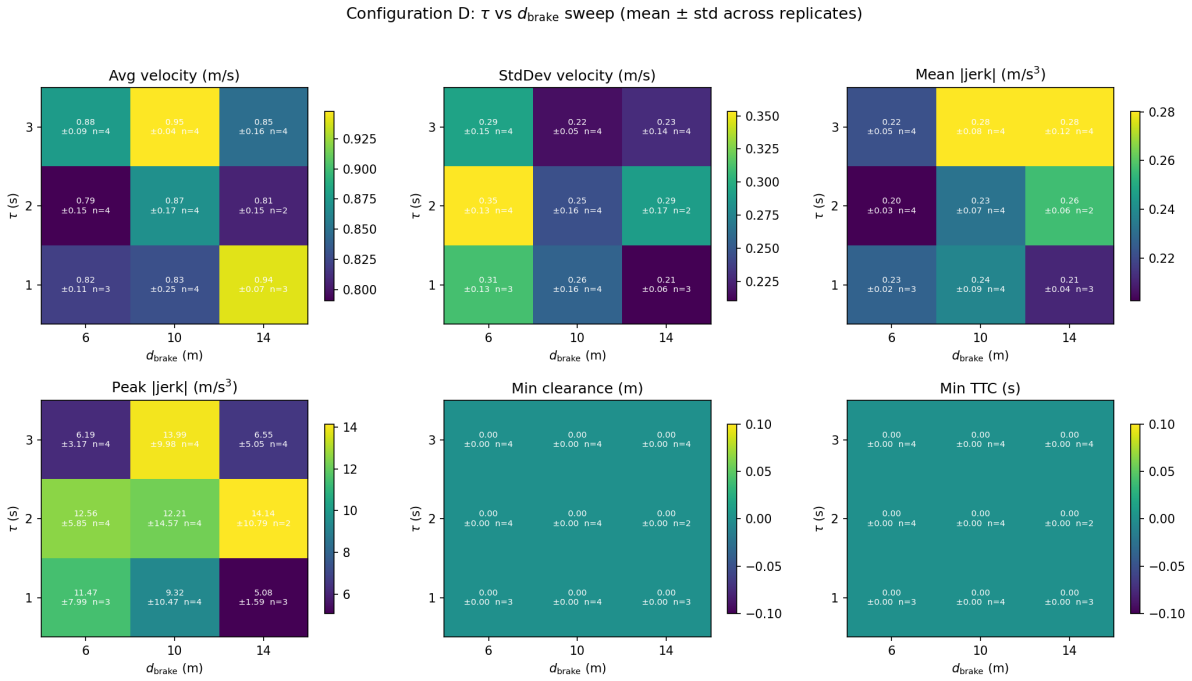

Figure A.1: τ versus d_{brake} sensitivity sweep for Configuration D on the 60 s straight scenario. Each cell shows mean $\pm$ standard deviation across n replicates (n printed in-cell). Within-cell variance is comparable to between-cell variance across most of the grid, indicating parameter robustness in the explored range.

Three observations follow from the data. First, within-cell standard deviations of the StdDev-velocity metric are 0.05–0.17 m/s, comparable to the between-cell spread of cell means (0.21–0.35 m/s). The sweep therefore does not isolate a statistically significant "best" setting. The grid sits on a robustness plateau rather than around a sharp optimum. Second, the tightest cells are $\tau = 1.0$ s with $d_{\text{brake}} = 14$ m (0.21 ± 0.06) and $\tau = 3.0$ s with $d_{\text{brake}} = 10$ m (0.22 ± 0.05), bracketing the chosen Configuration D baseline ($\tau = 2.0$, $d_{\text{brake}} = 10$) at 0.25 ± 0.16. On this evidence the baseline is not optimal but is well inside the plateau. Third, the $d_{\text{brake}} = 6$ m column trends slightly worse on average (0.29–0.35) but overlaps the other columns within one standard deviation, weaker evidence of an effect than the single-run dataset of Section 3.7 suggested.

The remaining parameter groups were not swept. The classifier gates v_{thresh} and n_{min} shape co-flow membership and therefore the long-run mean velocity rather than its variance: relaxing them admits noisy short-lived tracks, while tightening them collapses Configuration D toward Configuration F (proximity-only) at low pedestrian counts. A larger-replicate Latin-hypercube design over the full $(\tau, d_{\text{brake}}, d_{\text{min}}, n_{\text{min}})$ space is left for future work (Section 4.2).