Towards Federated Diffusion for Robot Navigation

Distributed Training of Generative Control Policies

Thomas Cramer

Delft University of Technology

TU Delft

Towards Federated Diffusion for Robot Navigation

Distributed Training of Generative Control Policies

by

Thomas Cramer

MSc Thesis report

Supervisor: Dr. L. Ferranti
Place: Faculty of Mechanical Engineering, Delft

Cover: Created with Google Gemini

TU Delft

Towards Federated Diffusion for Robot Navigation: Distributed Training of Generative Control Policies

Thomas Cramer

***Abstract*—Diffusion Policies enable imitation learning by generating action sequences through iterative denoising. In robotics, policies are commonly trained centrally by pooling demonstrations into a single dataset, but demonstrations are often distributed across robots or sites and cannot be easily shared due to privacy, ownership, bandwidth, or operational constraints. This thesis studies whether diffusion based navigation policies can be trained with federated learning while maintaining closed loop performance under client heterogeneity and limited local data. We introduce FedDiff, a pipeline combining Diffusion Policy training with Federated Averaging (FedAvg), and evaluate it on a controlled 2D navigation benchmark with low dimensional beam observations.**

We compare three training regimes: (i) individual (no sharing), where each client trains only on its own data; (ii) centralized training on pooled data; and (iii) federated training via FedAvg. Experiments cover three scenario families: near-IID (clients have similar data distributions) with sufficient data, near-IID with scarce data, and non-IID compositional generalization across distinct navigation primitives. Across two sensing configurations (4-beam and 8-beam), we train 36 models and report success rates on held out start–goal configurations and, where applicable, on environments not used during training. Federated training is most beneficial under data scarcity and, in the 8-beam Near-IID setting, can match or slightly exceed centralized training on average, while single client training remains strongest on its own training room in the 4-beam data sufficient case. In the Non-IID setting, federated and centralized models perform strongly on primitive navigation skills but remain limited on composed environments that require combining those primitives. We also discuss training dynamics and practical sensitivities relevant to real world deployment.

I. Introduction

Robotic systems are increasingly deployed as fleets, such as delivery robots in buildings, mobile platforms in warehouses, and autonomous vehicles on public roads, where performance improves with scale and continual data collection. At the same time, recent advances in generative sequence models have enabled planners and policies that output long horizon action sequences rather than single step commands. In particular, diffusion based policies generate action trajectories through iterative denoising and have shown strong performance in centralized imitation learning. A practical limitation, however, is that these models typically benefit from large and diverse demonstration datasets to achieve robust closed loop behaviour and generalization.

In real deployments, such data are rarely available on a single robot or at a single site: demonstrations are naturally distributed across robots and locations, and centralizing raw trajectories can be undesirable due to privacy requirements, bandwidth limits, or operational policies. This motivates learning methods that leverage fleet scale experience without requiring raw data to leave the devices that generate it.

This thesis introduces FedDiff, a framework for training diffusion based control policies using federated learning (FL). In FL, clients keep data locally and periodically share model updates with a central server that aggregates them into a global model. FedDiff provides a controlled experimental pipeline to study how diffusion policies behave under federated conditions in a 2D navigation domain.

A key challenge in federated learning is that client datasets are often non-IID, meaning they are not independent and identically distributed across clients, which can induce client drift and unstable optimization. In addition, federated training requires repeated parameter exchange, and these effects can interact unfavourably with high capacity generative policies. FedDiff is designed to quantify these interactions and identify when federated training improves over local only learning and how close it approaches (or exceeds) centralized training.

A. Research objectives

We study whether diffusion policies can be trained federatively without sacrificing closed loop navigation performance under (i) environment heterogeneity and (ii) limited local data. Concretely, we compare individual (local only), centralized, and federated (FedAvg) training across multiple environment classes and observation/action representations.

B. Research questions

RQ1 Under what environment shifts (near-IID vs. non-IID) does federated training improve over local only diffusion policy training?

RQ2 How do observation/action representations (global vs. egocentric) affect stability and generalization of diffusion policies in federated learning?

RQ3 Can federated learning mitigate local data scarcity for diffusion policies, and what dataset interventions (e.g., recovery injection) are required to avoid degenerate behaviours?

C. Thesis outline

Section II reviews diffusion models, diffusion policies, and federated learning. Section III and IV introduce the formal preliminaries, notation, and problem formulation. Sections V and VI describe the FedDiff methodology and experimental design. Results and discussion are presented in Sections VII and VIII. Finally, a conclusion is given in Section IX.

II. Related work and contributions

Diffusion models generate complex structured outputs by iteratively denoising noise, with modern formulations such as Denoising Diffusion Probabilistic Models [1] and scalable latent diffusion models [2] building on earlier diffusion and score based approaches [3], [4], [5]. In robotics, diffusion has been successfully adapted to planning and control through conditional trajectory generation [6], policy level visuomotor control via action sequence diffusion [7], and navigation and path planning [8], with extensions targeting efficiency and real time execution [9] and goal conditioned imitation learning [10]. In parallel, Federated Learning (FL) enables collaborative model training without centralizing data, with FedAvg as the canonical algorithm [11], and a mature body of work addressing client heterogeneity, communication efficiency, and privacy risks [12], [13], [14], [15], [16], [17]. Despite strong progress in centralized diffusion based robotic control and extensive FL research, their intersection remains largely unexplored: existing diffusion policies are almost exclusively trained in centralized settings, and there is limited empirical understanding of how high capacity diffusion models behave under federated constraints. This leaves an open research gap in determining whether diffusion policies can be trained federatively while preserving good performance under client heterogeneity, motivating reproducible baselines such as federated Diffusion Policy with FedAvg.

A. Contributions

- A modular **FedDiff** pipeline for federated training and evaluation of diffusion policies in 2D navigation, enabling controlled comparisons between local only, centralized, and FL regimes.
- A structured experimental benchmark that evaluates federated diffusion policies under **(i) varying data similarity between clients** and **(ii) varying amounts of local data**, highlighting when FL helps and when it does not.
- An empirical comparison of two **state/action representations** for navigation (an absolute/global formulation and a relative/egocentric formulation) and how this choice affects generalization under FL.
- A practical dataset refinement to improve learning of **recovery and turning behaviour** in the egocentric setting, and an analysis of its effect on closed loop performance.

To facilitate reproducibility and reuse, the complete implementation of the proposed FedDiff pipeline, including environment generation, data processing, training, evaluation, and analysis scripts, is provided in a unified public repository on GitHub. [1]

III. Preliminaries

A. Diffusion

Diffusion models are generative models that learn a data distribution by reversing a gradual corruption (noising) process. A canonical instantiation is the Denoising Diffusion Probabilistic Model (DDPM) [1], which defines a forward Markov chain, denoted by q, that incrementally perturbs a clean sample $x_0 \sim p_{\text{data}}(x)$ with Gaussian noise:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\Big(x_t; \sqrt{1-\beta_t}\, x_{t-1},\, \beta_t \mathbf{I}\Big), \tag{1}$$

where $t = 1, \ldots, T$. T is the total number of diffusion steps, and $\{\beta_t\}_{t=1}^{T}$ is a variance schedule. This process admits a closed form:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \mathbf{I}), \tag{2}$$

with $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$.

DDPMs learn to reverse this process by parametrizing the reverse (denoising) distribution with a neural network $\varepsilon_\theta(x_t, t)$, where θ denotes the network parameters, and minimizing the standard noise prediction objective:

$$\mathcal{L}_{\text{DDPM}}(\theta) \;=\; \mathbb{E}_{x_0,t,\varepsilon}\Big[\big\|\varepsilon - \varepsilon_\theta(x_t, t)\big\|_2^2\Big], \tag{3}$$

This objective is not merely a denoising heuristic. In DDPMs, minimizing the noise prediction loss is equivalent to optimizing a variational lower bound on the log-likelihood of the data, meaning it provides a tractable way to train the model to assign high probability to the observed samples. [1].

B. Diffusion Policy

Diffusion Policy (DP) [7] adapts diffusion models to control by learning a conditional generative model over action sequences given observations:

$$p_\theta(\mathbf{A}_t \mid \mathbf{O}_t), \tag{4}$$

where p_θ denotes the learned conditional distribution, $\mathbf{O}_t$ is the observation at time t (e.g. images or proprioception), and $\mathbf{A}_t = (a_t, \ldots, a_{t+H-1})$ is a length H action sequence. Control is executed in a receding horizon manner by applying only the first few actions and replanning afterwards.

At inference time, DP performs denoising in the action sequence space, iteratively refining an action sequence initialized from noise and conditioned on $\mathbf{O}_t$. A stochastic refinement update is:

$$\mathbf{A}_t^{k-1} \;=\; \alpha\Big(\mathbf{A}_t^{k} \;-\; \gamma\, \varepsilon_\theta(\mathbf{O}_t, \mathbf{A}_t^{k}, k) \;+\; \mathcal{N}(0, \sigma^2 \mathbf{I})\Big), \tag{5}$$

where k indexes the diffusion refinement step, γ is a step size, σ controls injected sampling noise, and α denotes a scaling factor associated with the diffusion schedule. After K refinement steps, the resulting $\mathbf{A}_t^0$ is used for execution by applying only the first action (or a short sequence) in a receding horizon manner, and the procedure repeats at the next control cycle.

DP is trained using a conditional noise prediction objective. The per step loss is:

$$\mathcal{L} \;=\; \text{MSE}\Big(\varepsilon^k, \varepsilon_\theta\big(\mathbf{O}_t, \mathbf{A}_t^0 + \varepsilon^k, k\big)\Big), \tag{6}$$

where $\mathbf{A}_t^0$ is the clean action sequence and ε^k is injected noise at diffusion step k.

[1] https://github.com/thomascramer23/master_thesis

C. Federated Learning

Federated Learning (FL) trains a shared model across K clients without centralizing raw data. Each client k holds a private dataset $\mathcal{D}_k$ and performs local optimization; a server aggregates client updates into a global model. The canonical algorithm is Federated Averaging (FedAvg) [11]. At communication round t, the server broadcasts parameters θ_t; each client computes locally updated parameters θ_t^k (e.g., by running several steps of SGD on $\mathcal{D}_k$). The server aggregates using a weighted average:

$$\theta_{t+1} = \sum_{k=1}^{K} \frac{|\mathcal{D}_k|}{\sum_{j=1}^{K} |\mathcal{D}_j|} \theta_t^k. \tag{7}$$

FL is attractive for robotics because data is naturally distributed across robots or sites and may be sensitive or costly to transmit. However, non-IID client data, system heterogeneity, and communication constraints can complicate convergence and may require careful evaluation when federating high capacity models such as diffusion policies.

D. Notation and conventions

Symbol	Meaning
t	rollout timestep
i	index within horizon
k	denoising step
K	# denoising steps
T_o	observation horizon
T_p	prediction horizon
T_a	execution horizon ($\leq T_p$)
o_t	observation
a_t	action / target
A_t	action sequence $(a_t, \ldots, a_{t+T_p-1})$
π_θ	diffusion policy
θ	parameters
r	FL round
E	local epochs/round
K_{cli}	# clients
$\mathcal{S}_r$	clients in round r
n_k	samples on client k
$\theta^{(r)}$	global params
$\theta_k^{(r)}$	client params
SR	success rate (%)

TABLE I. Notation used throughout this thesis.

IV. Problem Formulation

We consider a planar navigation task defined over a two dimensional workspace $\mathcal{W} \subset \mathbb{R}^2$ populated with static, impenetrable obstacles. Obstacles form walls and blocked regions and it is assumed that they remain fixed for the duration of each episode. An episode corresponds to a single navigation trial, in which the robot is initialized at a randomized start pose and tasked with reaching a designated goal within the same environment layout. The environment layouts are manually designed to induce navigation scenarios of varying structural complexity, rather than being sampled from a stochastic generative process. This design choice enables controlled experimentation and systematic evaluation across distinct environment classes.

The robot is modelled as a unicycle with state $\mathbf{x} = [x,\ y,\ \theta]^\top$, where $(x, y) \in \mathcal{W}$ denotes the planar position and $\theta \in (-\pi, \pi]$ the heading. The robot obeys the standard nonholonomic unicycle motion model, assuming no lateral slipping, with control inputs given by linear velocity v and angular velocity ω [18]:

$$\begin{aligned} \dot{x} &= v\cos\theta, \\ \dot{y} &= v\sin\theta, \\ \dot{\theta} &= \omega, \\ v &\in [v_{\min}, v_{\max}], \quad \omega \in [\omega_{\min}, \omega_{\max}]. \end{aligned} \tag{8}$$

The objective of the robot is to navigate from a randomized start position to a goal location while respecting kinematic constraints and avoiding collisions. For each episode, a single environment layout is sampled and held fixed throughout execution. Figure 1 illustrates an example environment used in this work. At each time step t, the robot receives a partial observation $\mathbf{o}_t$ derived from local range measurements and goal information, and outputs an action sequence $\mathbf{a}_t$ that is used for rollout and evaluation.

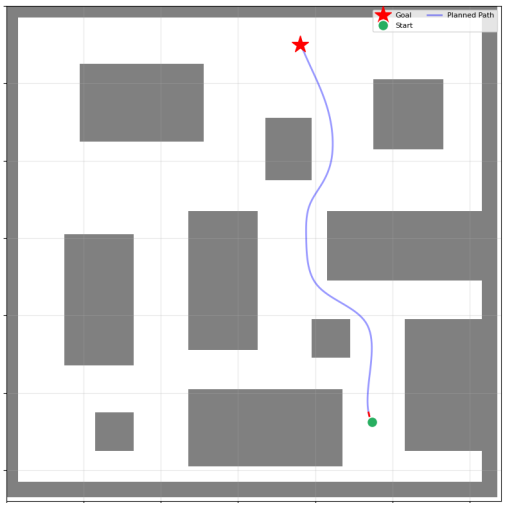

Fig. 1. Example room environment used for navigation experiments, showing the start position (green), the goal position (red), and the planned navigation path (blue) around obstacles.

V. Methodology

This section presents the methodology underlying the proposed *FedDiff* framework. The objective is to provide a modular and extensible pipeline that supports:

1) The construction of navigation environments and expert demonstrations.
2) The definition of diffusion policy observation and action interfaces as controlled design variables
3) Training under individual, centralized, and federated regimes.
4) Systematic evaluation under consistent conditions

The framework is intentionally structured to enable direct comparisons between centralized and federated training while isolating the effects of representation choices (e.g., sensing fidelity and reference frame) on learning stability and generalization.

Figure 2 summarizes the FedDiff pipeline, combining the federated training loop (FedAvg) with the diffusion policy receding horizon control loop used at deployment. Subsection V-A describes environment and dataset construction, including map design, expert path generation, and the procedural construction of observations and actions. Subsection V-B then formalizes the policy interfaces used throughout this thesis, introducing the observation and action spaces considered in successive experimental phases. Finally, Subsection V-C details the training framework for diffusion policies under individual, centralized, and federated learning.

A. Data Generation

Data generation forms the foundation of the experimental pipeline, as both training and evaluation rely on high quality expert demonstrations collected in representative environments. This subsection describes how navigation environments and corresponding observations are constructed, and how expert paths are generated to serve as training data for diffusion policies. The design emphasizes reproducibility and control, enabling consistent comparisons between training regimes and representation choices. By decoupling environment construction from policy training, the framework enables consistent comparisons between individually trained, centrally trained, and federated models under identical conditions.

1) Map Generation: We design multiple 2D occupancy grid environments with varying navigation complexity:

- Corridor: straight line navigation scenario.
- L-turn: single turn navigation scenario.
- Cross: multi directional intersection.
- Room 1–5: complex indoor layouts with static obstacles.

All environments consist of axis aligned walls and rectangular obstacles, yielding piecewise orthogonal corridors and corners. Each map type represents a distinct navigation challenge and serves to simulate heterogeneous local data distributions in federated learning experiments.

2) Path Generation: Expert reference paths are generated to be collision free and kinematically feasible. For each environment, paths are computed using the $A^\star$ algorithm [19] with obstacle inflation, combining heuristic estimates with accumulated path costs to efficiently find shortest feasible routes between start and goal positions. The resulting discrete paths are subsequently smoothed using cubic spline interpolation to obtain continuous paths suitable for execution by the unicycle model. The robot's heading along the demonstration is defined by the local tangent direction of the curve, yielding a consistent orientation profile that can be used to derive actions (Section V-A3). Start and goal positions are randomized for each episode to generate diverse demonstrations per environment, ensuring broad coverage of the navigation space and reducing overfitting to specific path geometries.

3) Observation and Action Generation: To keep the study tractable and to avoid introducing a separate perception stack, we adopt the low dimensional observation variant of Diffusion Policy proposed in [7]. We use a CNN based denoising architecture (full configuration in Subsection V-C). High dimensional visual observations would require an additional vision encoder and substantially increase compute cost; in this thesis we therefore focus on low dimensional geometric observations.

Although our federated experiments are executed in a simulated setting (clients run sequentially on one machine; Section V-C2c) and we do not benchmark network communication, we intentionally keep the model and observation dimensionality modest to make eventual real world federated deployment more plausible (e.g., periodic parameter exchange over constrained links). Accordingly, we employ a sensor agnostic observation model based on sparse range measurements, effectively simulating a low dimensional LiDAR like representation. Throughout the remainder of this thesis, these range measurements are referred to as *beams*.

a) Range beams: At each time step, beams are cast from the robot's position in fixed angular directions relative to the robot's heading. Each beam returns the Euclidean distance from the robot to the nearest obstacle boundary along that ray. Distances are computed by ray marching through the occupancy grid and converted to metric units using the grid resolution. We consider both a four beam and an eight beam configuration; the exact observation vectors are defined formally in Section V-B. A small number of range measurements aligns with minimalist real world proximity sensing [20] and has been shown to be sufficient for mapless navigation in low dimensional settings [21]. An example of an eight beam setup is shown in Figure 3.

b) Action targets from expert paths: Actions used for supervised training are derived directly from the time discretized expert path. Depending on the experimental phase (elaborated in Section V-B), targets are computed either as (i) global target poses for the next waypoint(s), or (ii) relative displacements in the robot's local frame. For relative actions, the translational displacement between consecutive waypoints is rotated into the robot frame at time t, and the heading change is computed as the wrapped angle difference $\Delta\theta \in (-\pi, \pi]$. These targets are computed deterministically from the expert trajectory and paired with the corresponding observation at each time step to form training tuples.

c) Normalization: All observation and action dimensions are normalized using a fixed min–max scaling of range $[-1, 1]$ computed from the training data.

B. State and Control Space Parametrizations

While the underlying navigation task and robot dynamics remain fixed, the parametrization of the policy's observation and action spaces is treated as a design variable in this work. Different formulations are introduced and evaluated to study their impact on learning stability, generalization, and suitability for federated training.

At each discrete timestep t, the policy receives one or more observations $\mathbf{o}_t \in \mathcal{O}$ and outputs one or more actions $\mathbf{a}_t \in \mathcal{A}$.

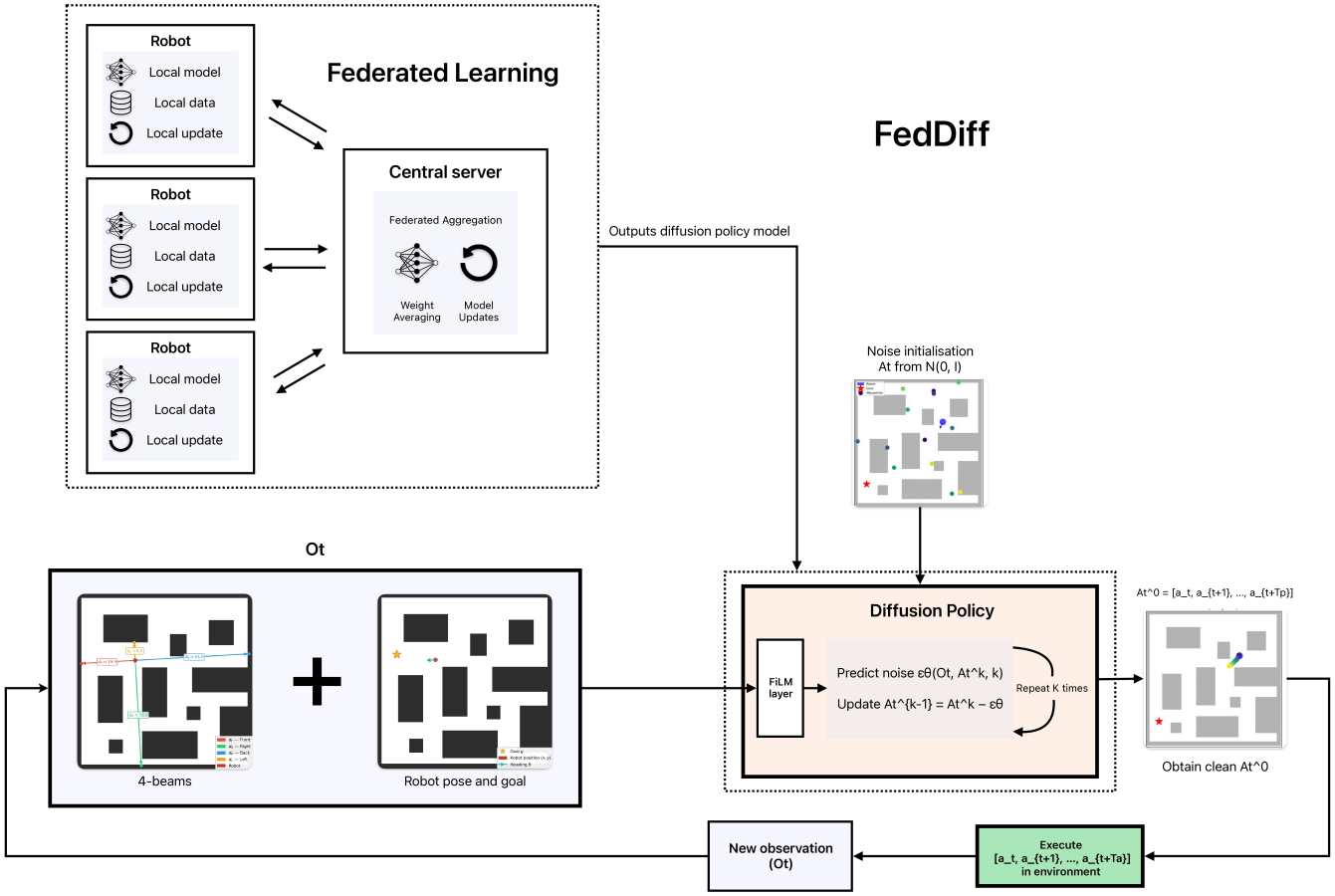

Fig. 2. Overview of the proposed FedDiff framework (4-beam interface). **Left: federated training.** Multiple robots (clients) each train a local diffusion policy on their own private navigation data and send model updates to a central server. The server performs federated aggregation by weight averaging to obtain a global diffusion policy model, which is then redistributed to the clients. **Bottom: closed loop deployment.** At time step t, the policy conditions on the current observation O_t, consisting of the local 4-beam observation together with the robot pose and goal information. Starting from a Gaussian noise sample $A_t^K \sim \mathcal{N}(0, I)$, the diffusion policy iteratively predicts and removes noise until a clean action sequence $A_t^0 = [a_t, a_{t+1}, \ldots, a_{t+T_p-1}]$ is obtained. Only the first T_a actions of this prediction horizon are executed in the environment, after which a new observation is collected and the planning process repeats in a receding horizon manner until the goal is reached.

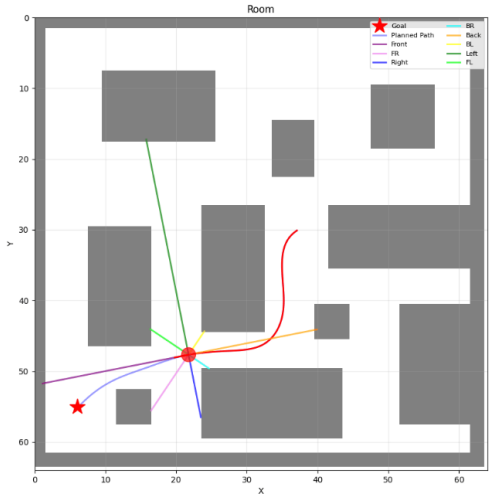

Fig. 3. Illustration of the 8-beam observation model in the Room environment. The robot is located at the center, and the eight colored rays show the fixed sensing directions used to measure distances to the nearest obstacle or boundary: front, front-right, right, back-right, back, back-left, left, and front-left. This representation provides a sparse but richer local description of the surrounding free space than the 4-beam formulation.

In all cases, the policy is trained to predict short action sequences in a receding horizon manner (Section V-C). The following subsections define the specific interfaces considered.

1) Global Observation Space with Four Beams: We begin with a minimal baseline that provides the policy with global pose and goal information together with sparse range sensing. The observation vector is defined as

$$\mathbf{o}_t^{(4,\text{global})} = \big[\, x_t,\ y_t,\ \sin\theta_t,\ \cos\theta_t,\ x_g,\ y_g,\ r_1,\ r_2,\ r_3,\ r_4 \,\big] \tag{9}$$

where (x_t, y_t, θ_t) is the robot pose in the global frame, (x_g, y_g) is the goal position in the same frame, and r_i are four range measurements taken in the forward, backward, left, and right directions relative to the robot heading, analogous to the 8-beam setup shown in Figure 3.

The corresponding action is parametrized in global coordinates as a target pose:

$$\mathbf{a}_t^{\text{global}} = \begin{bmatrix} x_t^{\text{target}} & y_t^{\text{target}} & \sin\theta_t^{\text{target}} & \cos\theta_t^{\text{target}} \end{bmatrix}. \tag{10}$$

This setup serves as a low dimensional baseline and establishes a reference performance level. While it achieves meaningful performance in the non-IID setting, especially in seen environments, its sparse sensing can introduce blind spots that reduce robustness and generalization in the near-IID setting (see Section VII).

2) Increased Sensing Fidelity with Global Observations: To improve environmental awareness while retaining a compact observation space, we extend the range sensing component to eight beams uniformly distributed at 45° intervals around the robot. This modification primarily targets diagonal blind spots in the four-beam setup, which are especially problematic near corners.

In pilot experiments, increasing sensing fidelity within the global coordinate formulation improved performance on the training episodes (high success when evaluated on held out start-goal pairs from the same maps) while performance on unseen start-goal pairs degraded sharply, despite lower training loss. This train-test discrepancy is consistent with map specific shortcut learning: because the policy is conditioned on absolute pose/goal in the global frame, additional range measurements can increase the capacity to exploit environment dependent correlations rather than learning transferable navigation behaviour. These observations motivate an egocentric reformulation.

3) Egocentric Observation Reformulation with Eight Beams: To encourage map agnostic behaviour, we reformulate the observation space in the robot's local reference frame. Concretely, the goal is expressed as a relative displacement in the robot frame, while the range beams remain defined relative to the robot's heading:

$$\mathbf{o}_t^{\text{ego}} = \begin{bmatrix} x_{g,t}^{\text{ego}} & y_{g,t}^{\text{ego}} & r_1 & \dots & r_8 \end{bmatrix}. \tag{11}$$

Here, $(x_{g,t}^{\text{ego}}, y_{g,t}^{\text{ego}})$ is obtained by first computing the displacement vector from the robot to the goal in the global frame and then rotating this vector into the robot's local frame using the inverse heading θ_t. This representation removes direct access to the robot's absolute position while preserving relative goal geometry.

When paired with the earlier global target pose action parametrization, this egocentric observation model fails systematically. The reason is geometric: the policy no longer observes its global position, yet is asked to predict actions in the global frame. Without an estimate of (x_t, y_t, θ_t), the mapping from local perceptual cues to an absolute target pose is ill posed, and the learned predictions become inconsistent across environments.

To restore observation–action coherence, we therefore replace the global target pose action space with a relative action parameterization in the robot frame:

$$\mathbf{a}_t^{\text{rel}} = \begin{bmatrix} \Delta x_t & \Delta y_t & \Delta\theta_t \end{bmatrix}, \tag{12}$$

where $(\Delta x_t, \Delta y_t)$ denotes a local frame displacement over the next step and $\Delta\theta_t$ the corresponding heading change. This change yields a well posed learning problem: both inputs and outputs are expressed in the same egocentric coordinate system, enabling the policy to learn navigation behaviours that are not tied to absolute map coordinates.

In summary, the observation and action formulations introduced above define two complementary configurations that are evaluated throughout this work. The first configuration combines the global coordinate four beam observation model $\mathbf{o}_t^{(4,\text{global})}$ with the global target pose action parametrization $\mathbf{a}_t^{\text{global}}$. The second configuration adopts an egocentric representation, using the eight beam observation model $\mathbf{o}_t^{\text{ego}}$ together with relative actions $\mathbf{a}_t^{\text{rel}}$. All experimental results reported in the following sections are obtained by evaluating both configurations, enabling a structured comparison of two evaluated configurations: a 4-beam/global setup and an 8-beam/egocentric setup.

C. Training Framework

Our approach builds upon the Diffusion Policy (DP) framework introduced by Chi et al. [7]. We adopt the convolutional (CNN based) variant of Diffusion Policy, which provides a stable and well understood baseline for low dimensional control and is less sensitive to hyperparameter tuning than the transformer based alternative in resource constrained settings. While increased model capacity can improve performance in centralized training [7], communication efficiency is a primary concern in federated learning, where model parameters must be exchanged repeatedly between clients and a central server. Accordingly, we employ a minimally sized denoising network that remains expressive enough to capture the dynamics of the navigation task while limiting communication overhead. This design balances performance and efficiency and better reflects deployment constraints on real robotic systems, where bandwidth and latency can cause problems when federated learning is used.

This subsection describes the configuration of the diffusion policy model and the training procedure under three regimes: individual (local only), centralized, and federated learning.

1) Diffusion Policy Configuration: We employ Diffusion Policy as a conditional generative model over short horizon action sequences. At each time step t, the policy conditions on a window of past observations and predicts a sequence of future actions, which are executed in a receding horizon fashion.

Following the formulation of Chi et al. [7], the policy is parametrized by three temporal horizons: (i) an observation horizon T_o, (ii) a prediction horizon T_p, and (iii) an execution horizon T_a. Specifically, the denoising model takes as input the most recent T_o observations

$$\mathbf{o}_{t-T_o+1:t},$$

and predicts a sequence of T_p future actions

$$\mathbf{a}_{t:t+T_p-1}.$$

During execution, only the first T_a actions of this sequence are applied to the environment before replanning at the next time step.

a) Diffusion process and noise schedule: The diffusion model is trained using the improved DDPM (iDDPM) formulation [22]. We adopt the squared cosine noise schedule, as recommended by Chi et al. [7], which concentrates diffusion steps in low noise regimes and has been shown to improve both training stability and sample quality for action sequence generation.

The model is trained with 100 diffusion steps, and inference is likewise performed using 100 denoising steps. This configuration provides a favourable trade off between computational cost and policy performance, and is used consistently across all experiments to ensure comparability. An example of such a denoising process can be seen in Figure 4.

b) Optimization, EMA, and learning rate scheduling: Training is performed using the AdamW optimizer [23], which decouples weight decay from gradient based updates and improves generalization stability in deep networks. To further stabilize training and evaluation, we maintain an exponential moving average (EMA) of the model parameters throughout training. Unless stated otherwise, policy rollouts are generated using the EMA smoothed parameters, following the standard practice established in Diffusion Policy [7].

The learning rate is scheduled using a linear warm up followed by cosine decay. During an initial warm up phase, the learning rate increases linearly to mitigate early optimization instability. After warm up, it is annealed using a cosine schedule, which provides smooth convergence and reduces sensitivity to local noise. This scheduling strategy is particularly beneficial in federated settings, where model updates originate from heterogeneous data distributions.

2) Training Regimes: We consider three complementary training regimes to assess the benefits and limitations of federated learning for diffusion based navigation policies.

a) Individual training: In the individual (local only) regime, each client trains a diffusion policy exclusively on its private dataset of expert demonstrations. This setting serves as a baseline for evaluating local performance and highlights the limitations imposed by restricted data diversity or quantity.

b) Centralized training: In the centralized regime, a single model is trained on the union of all client datasets. This provides an upper bound reference for performance when all data can be pooled and serves as a point of comparison for federated training outcomes.

c) Federated training (FedAvg) in a simulated FL setting: Federated training follows the standard Federated Averaging (FedAvg) procedure [11]. In each communication round, the server broadcasts the current global model parameters to participating clients. Each client performs E local epochs of training on its private dataset and returns updated parameters to the server. The server aggregates these updates using a dataset size weighted average:

$$\theta^{(r+1)} = \sum_{k=1}^{K} \frac{n_k}{\sum_{j=1}^{K} n_j} \theta_k^{(r)}, \tag{13}$$

where $\theta_k^{(r)}$ denotes the parameters after local training on client k in round r, and n_k is the number of training samples available at that client. All clients share the same model architecture and state/control representation within a given experiment, ensuring that parameter aggregation is well defined.

Federated experiments are run in a simulated setting (clients executed sequentially on one machine). This matches FedAvg algorithmically but does not evaluate system communication constraints (bandwidth/latency, stragglers, compression, secure aggregation). We therefore focus on statistical heterogeneity and generalization rather than communication benchmarking.

We nevertheless use a compact denoising network (≈ 65M parameters) and low dimensional observations to avoid a vision encoder and to keep compute manageable, thereby isolating federated optimization effects from perception stack complexity.

3) Evaluation Protocol: All models are evaluated using closed loop rollouts in held out environments and start–goal configurations not seen during training. An episode is considered successful if the robot reaches the goal without colliding with any obstacles within the episode horizon. The primary evaluation metric is the success rate.

In all experiments, we use the same horizon configuration as in [7], namely

$$T_o = 2, \qquad T_p = 16, \qquad T_a = 8.$$

This choice provides sufficient temporal context for action prediction while maintaining a responsive closed loop controller. Unlike highly dynamic manipulation tasks, the navigation environments considered in this work are static; consequently, shorter horizons are not required to react to rapidly changing scene dynamics, and longer prediction horizons primarily serve to stabilize motion and turning behaviour.

To assess generalization, policies are evaluated under the same protocols across environments of varying similarity and data availability, as described in the experimental scenarios of Section VI. Both state/control representations introduced in Section V-B are evaluated independently under identical training and evaluation conditions to enable a controlled comparison.

VI. Experimental Design

This section describes the experimental design used to evaluate federated Diffusion Policies under controlled variations in environment heterogeneity, task complexity, and data availability. The goal is to isolate the conditions under which federated training provides benefits over local and centralized baselines, and to analyse how representation choices interact with these conditions. To this end, we construct a set of evaluation scenarios that differ in their degree of distributional shift across clients and in the difficulty of the navigation task.

Across all experiments, the underlying robot dynamics, training procedure, and evaluation metrics are held fixed. All training runs, experiments, and evaluation tests were conducted on DelftBlue [24].

This evaluation is designed to operationalize the thesis research questions and contributions. In particular, varying the degree of similarity between client environments (Near-IID versus Non-IID) addresses RQ1, comparing the two

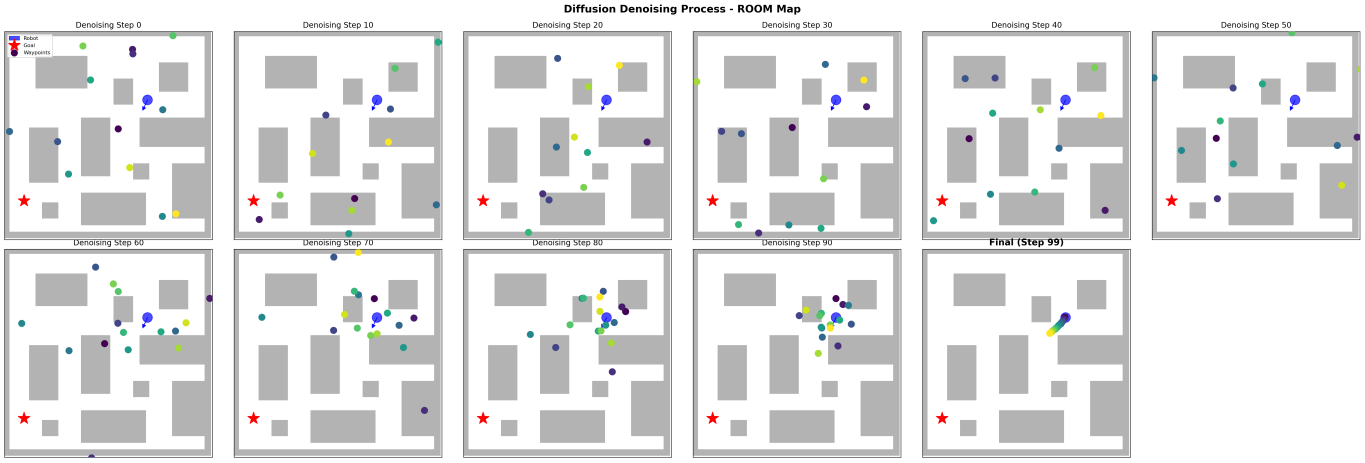

Fig. 4. Visualization of the diffusion denoising process on the Room map across 100 denoising steps. The intermediate panels show how the sampled trajectory is gradually refined from noisy initial predictions into a coherent path. The blue marker indicates the start position, and the red star indicates the goal position.

state/control formulations within each scenario addresses RQ2, and varying the amount of data available per client addresses RQ3. More broadly, the three scenarios instantiate the core contribution of this thesis: a controlled benchmark for comparing local only, centralized, and federated Diffusion Policy training under changes in client heterogeneity, data availability, and representation choice.

A. Environment Sets and Scenario Definitions

All experiments are conducted in 2D occupancy grid environments of size 64×64 cells with static, axis aligned obstacles. Each environment induces a distinct navigation topology, characterized by the number of turns, intersections, and open-space regions encountered along feasible paths.

We organize the environments into two sets: (i) a heterogeneous Non-IID set with varying task complexity, and (ii) a Near-IID set of structurally similar but distinct environments. These sets are used to instantiate the three evaluation scenarios considered in this thesis.

This split is intentional. The Near-IID set allows us to study aggregation when clients face similar navigation structure, thereby isolating the role of training regime and local data availability. By contrast, the Non-IID set induces stronger cross client heterogeneity and is used to test whether federated training can transfer or combine behaviours learned from qualitatively different navigation primitives. Using both sets therefore lets us distinguish the effects of statistical heterogeneity from those of task complexity, and assess when federated learning helps, when centralized pooling remains stronger, and when representation choice becomes the dominant factor.

B. Operational Definitions of Client Heterogeneity

We use the terms *near-IID* and *non-IID* as operational labels for the client distributions induced by the environment split:

- **Near-IID:** Clients are trained on environments drawn from the same environment family (e.g., room layouts), with similar geometric structure and navigation primitives.
- **Non-IID (compositional):** Clients are trained on distinct primitive environment classes (e.g., corridor, L-turn, cross), and evaluation is performed on a composed environment that requires combining these primitives.

These labels are not intended as strict statistical claims about the underlying data generating process, but as controlled experimental regimes that induce different degrees of client heterogeneity. In particular, a bootstrap based Jensen–Shannon divergence analysis of trajectory-level heading-change distributions showed that the primitive split is substantially more heterogeneous than the room split (mean within split JSD $\approx 9 \times 10^{-3}$ vs. $\approx 1 \times 10^{-3}$), supporting the *non-IID/near-IID* terminology in a relative, benchmark-specific sense; see Appendix A for details.

C. Non-IID Scenario: Compositional Generalization

The first experimental scenario is designed to evaluate compositional generalization in a heterogeneous, Non-IID setting. Here, each client is assigned a geometrically distinct primitive environment that emphasizes a specific navigation skill: straight line motion (`Corridor`), single 90 degree turning (`L-Turn`), and multi directional decision making (`Cross`).

These primitive environments are intentionally disjoint and do not individually contain the full set of behaviours required to solve more complex navigation tasks. As a result, a policy trained locally on any single environment should lack the experience needed to navigate a structurally richer layout.

Training is performed using individual, centralized, and federated regimes on the set of primitive environments. Evaluation is conducted zero shot on an unseen `Room` environment that combines multiple primitives (corridors, turns, and intersections) within a single layout. The different environments are presented in Figure 5. This scenario directly tests whether federated learning enables the policy to compose navigation skills learned across heterogeneous clients into a coherent strategy for solving a more complex task.

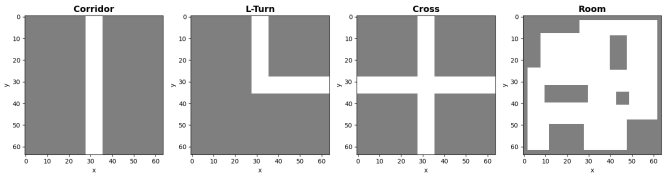

Fig. 5. The 2D environments corridor, L-turn, cross and room.

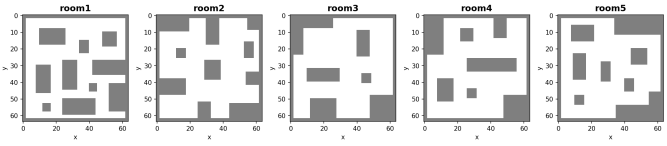

Fig. 6. The 2D environments room 1-5.

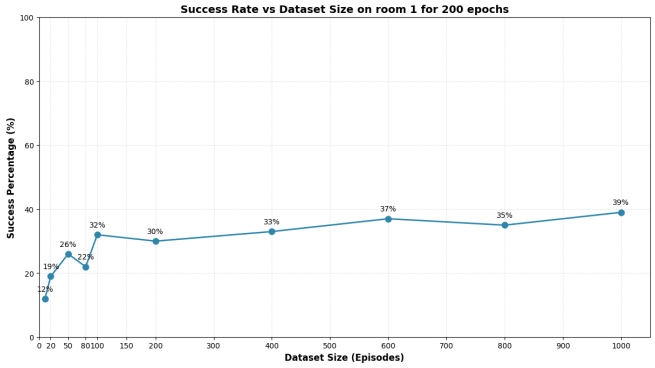

Fig. 7. Dataset ablation study showing navigation success rate as a function of the number of expert demonstrations for a Near-IID room environment. Performance improves rapidly at small dataset sizes but exhibits diminishing returns beyond approximately $N \approx 100$ episodes.

D. Near-IID Scenario with Sufficient Data

The second scenario evaluates generalization within a single environment class under Near-IID conditions, where data availability is not a limiting factor. The environments used in this scenario are presented in Figure 6. Clients are assigned four structurally similar room environments (`Room 1--4`), each with access to a sufficiently large local dataset of expert demonstrations.

Although the room layouts differ in their specific obstacle configurations, they share common structural properties such as grid resolution, obstacle geometry, and sensing characteristics. As a result, the client data distributions are closely aligned, and independent local training is expected to yield reasonable performance.

Models are trained under individual, centralized, and federated regimes on `Room 1--4` and evaluated on a held out environment (`Room 5`). This scenario assesses whether the learned policy generalizes to a new instance of the same environment class and serves as a reference case in which federated learning may approach centralized performance under relatively homogeneous client data distributions.

E. Near-IID Scenario under Data Scarcity

The third scenario considers a Near-IID setting in which local data availability is severely limited. Clients are assigned room like environments from the same class (`Room 1--5`), but each client has access to only a small number of expert demonstrations. The same environments are used as in the sufficient data scenario, see Figure 6. In this regime, policies trained independently on local data perform poorly due to insufficient coverage of the state space, even within their own environments.

Training is performed using individual, centralized, and federated regimes across all room environments. Evaluation is conducted on the same set of room layouts, with held out start–goal configurations. This setting tests whether federated learning can improve overall navigation performance by aggregating complementary experiences across clients, effectively increasing the diversity and coverage of the training data without centralized data sharing.

F. Dataset Sizes and Epoch Configurations

This subsection describes the selection of dataset sizes and training durations for all experimental scenarios. These choices were guided by a combination of empirical ablation studies, task complexity considerations, and practical computational constraints. Across all experiments, the number of training epochs was chosen to ensure sufficient convergence of the diffusion policy while remaining compatible with the available compute resources on the DelftBlue cluster.

a) Near-IID scenario with sufficient data: To determine an appropriate dataset size for the Near-IID, data abundant setting, we conducted a preliminary ablation study on `Room 1` using the 8 beam setup, evaluating navigation success as a function of the number of expert demonstrations. As shown in Figure 7, performance improves rapidly at small dataset sizes but exhibits diminishing returns beyond approximately $N \approx 100$ episodes. While success rates continue to increase marginally at larger scales, the curve largely flattens, indicating that additional data yields limited benefit.

Based on this observation, we select $N = 200$ episodes per client for the Near-IID sufficient data scenario. This choice provides a conservative margin beyond the saturation point to ensure stable learning, while remaining feasible within the computational constraints of training on the DelftBlue cluster. Models in this scenario are trained for 1000 epochs across all training regimes (individual, centralized, and federated), which was empirically sufficient for convergence without overfitting.

b) Near-IID scenario under data scarcity: In the data scarce Near-IID setting, the goal is to evaluate whether federated learning can compensate for insufficient local data. Here, each client is provided with only $N = 20$ expert demonstrations. At this scale, local policies consistently fail to generalize due to incomplete coverage of the environment state space, as confirmed by the ablation results.

To ensure that limited data, rather than insufficient optimization, is the dominant failure mode, models in this scenario are trained for an extended duration of 1500 epochs. This choice reflects a deliberate trade off: longer training is used to ex-

tract maximal information from the small datasets, while still respecting overall resource limitations. Centralized training aggregates the corresponding 100 episodes, while federated training combines 5×20 episodes across clients.

c) Non-IID scenario: compositional generalization: For the Non-IID compositional generalization scenario, dataset sizes are deliberately non uniform and reflect the differing topological complexity of the training environments. Rather than assigning equal data volume to each client, we scale dataset sizes according to the topological entropy of each environment.

Specifically, we use $N = 50$ episodes for the `Corridor`, $N = 100$ for the `L-Turn`, and $N = 200$ for the `Cross`. The `Corridor` environment primarily serves as a regularization prior for straight line stability and therefore requires comparatively little data, while the `L-Turn` introduces isolated turning behaviour. In contrast, the `Cross` environment contains multiple intersections, ambiguous navigation choices, and high conflict states, and thus serves as the primary source of complex geometric structure. Under Federated Averaging, this weighting ensures that the global model prioritizes resolving intersection level ambiguities over low conflict hallway behaviour.

For centralized training, the datasets from all primitive environments are pooled, resulting in a total of 350 episodes. All models in this scenario are trained for 1000 epochs.

In all the federated trainings, clients perform 2 local epochs per round, ensuring the same total number of epochs as in centralized training while reconciling conflicting gradients from heterogeneous environments early and reducing client drift toward environment specific behaviours.

d) Asymmetric data requirements of global and egocentric policies: A critical observation from preliminary experiments is the asymmetric data requirement between the global and egocentric policy formulations. Although both models are trained on expert demonstrations generated via the $A^\star$ planner, the distribution of actions in these demonstrations differs significantly from what is required for effective learning in the egocentric case.

Standard optimal paths are dominated by straight line segments, with turning actions occurring only at sparse waypoints. Quantitatively, we observe that more than 80% of time steps exhibit negligible angular velocity. For the global four beam policy, this imbalance does not pose a problem. Because the policy conditions on absolute position, it effectively learns a vector field over the map, and small deviations from the expert trajectory are immediately corrected through changes in the global coordinate input. As a result, standard $A^\star$ trajectories are sufficient for effective learning without additional data engineering.

In contrast, the egocentric eight beam policy lacks access to absolute position and must infer appropriate actions purely from local geometry. When trained on standard expert data, the model collapses to a degenerate straight line behaviour, failing to learn robust turning and recovery strategies. To correct this kinematic bias, we introduce two forms of recovery data injection exclusively for the egocentric setup: (i) short horizon recovery episodes initialized near walls with large heading error, and (ii) explicit in place rotation actions ($v = 0, \omega \neq 0$) that teach the policy to reorient before advancing in confined spaces.

These engineered episodes rebalance the action distribution, ensuring that the egocentric policy encounters sufficient turning and recovery states during training. Importantly, this intervention modifies only the composition of the dataset, not its total size, allowing direct comparison with the global formulation under identical data budgets.

e) Consistency across sensing configurations: For the four beam global and eight beam egocentric policies, identical dataset sizes and epoch counts are used within each experimental scenario. Although the egocentric formulation requires additional recovery focused demonstrations, the total number of episodes remains unchanged. A preliminary dataset size study for the four beam configuration indicates similar saturation behaviour, and the same dataset sizes are therefore used to keep episode counts and training budgets matched across representations, while noting that the egocentric setup includes additional recovery focused demonstrations.

VII. Results

A. Reporting conventions and evaluation protocol

We report closed loop navigation **success rate** (SR), defined as the fraction of rollouts in which the robot reaches the goal within a fixed horizon without collision (Section V-C3). For fair comparison across representations and training regimes, evaluation uses direct state updates consistent with the action parametrization rather than continuous unicycle control integration. This removes numerical integration effects, so performance differences reflect learned behaviour rather than simulator artifacts. Each SR entry is computed from 200 rollouts per evaluation environment using held-out start–goal pairs. For each trained model, we evaluate on:

- **Non-IID scenario:** Corridor, L-turn, Cross, and the composed Room environment (4 SR values per model).
- **Near-IID (sufficient data):** Rooms 1–4 (training class) and Room 5 (unseen within class) (5 SR values per model).
- **Near-IID (limited data):** Rooms 1–5 (5 SR values per model).

We present results for two sensing configurations: **4-beam** and **8-beam**. Each configuration yields 18 trained models, for a total of 36 models.

B. Success rates by scenario and sensing configuration

We organize results by scenario (rows correspond to trained models; columns to evaluation environments). For readability, each table also reports an **Avg** column (mean SR over evaluation environments).

1) Non-IID compositional generalization: This scenario tests whether federated aggregation over heterogeneous primitives (Corridor, L-turn, Cross) yields policies that generalize and compose on the Room environment. Table II and Table III report the Non-IID results for the 4-beam and 8-beam settings, respectively.

4-beam results on the Non-IID split:

Dataset	Corridor	L-turn	Cross	Room	Avg
Corridor	**100**	6	45	6	39
L-turn	19	96	49	11	44
Cross	**100**	82	94	48	74
Centralized	**100**	95	95	**51**	**85**
Federated	**100**	**96**	**97**	46	**85**

TABLE II. SR with 4-beam observations for the Non-IID split. Each local model is trained on demonstrations from a single primitive environment, while the centralized and federated models are trained on the union of these heterogeneous client datasets. Rows denote the training dataset or regime, and columns denote the evaluation environment. The results therefore test both cross primitive generalization and the extent to which aggregation supports compositional navigation.

8-beam results:

Dataset	Corridor	L-turn	Cross	Room	Avg
Corridor	**100**	30	57	38	56
L-turn	96	71	74	49	73
Cross	94	62	85	53	73
Centralized	90	64	77	**56**	72
Federated	91	**75**	**87**	52	**76**

TABLE III. SR with 8-beam observations for the Non-IID client split. The table evaluates both in domain performance on the primitive environments and transfer to the composed Room environment. Compared with the 4-beam results, performance is more evenly distributed across environments, suggesting weaker specialization to the training primitive and more transferable navigation behaviour, although the Room environment remains the most challenging test of compositional generalization.

a) Key takeaway: With **4-beam** observations, the locally trained models remain strongly environment specific. Corridor and L-turn perform well on their own primitives but transfer poorly to the other layouts, especially to the composed Room environment (6% and 11%, respectively). Cross is the only local model that generalizes broadly, reaching 100% on Corridor, 82% on L-turn, and 48% on Room. Aggregating the heterogeneous primitive datasets substantially improves robustness on the training primitives: both Centralized and Federated reach 95–100% on Corridor, L-turn, and Cross, with identical average success of 85%. However, transfer to the composed Room environment remains markedly lower (51% for Centralized and 46% for Federated), indicating that strong performance on the individual primitives does not automatically translate into reliable composition in a more complex layout.

For **8-beam** observations, performance becomes more uniform across environments. The local models no longer exhibit the same all or nothing specialization seen in the 4-beam setting: Corridor reaches an average success of 56%, while L-turn and Cross each achieve 73% on average and both transfer reasonably to Room (49% and 53%, respectively). This pattern suggests a qualitative difference between the two representations. In the 4-beam setting, the local policies appear to depend more strongly on environment specific regularities, achieving high success primarily on layouts that resemble the training environment. By contrast, the 8-beam policies show more balanced performance across different maps, which indicates that they are learning a more transferable navigation strategy rather than simply memorizing individual environments. In particular, the reduced gap between same environment and cross environment performance suggests that the 8-beam representation better captures the underlying task of driving through free space while avoiding obstacles. Among the aggregated regimes, Federated attains the best overall average success at 76% and performs best on L-turn and Cross (75% and 87%), whereas Centralized achieves the highest success on the composed Room environment at 56%. Overall, the 8-beam setting yields more stable cross environment performance than 4-beam, but Room remains the hardest test case, and aggregation still does not produce a decisive breakthrough in compositional generalization.

2) Near-IID (data sufficient): This scenario evaluates generalization within a consistent environment class, where all clients are trained on Room-type maps and local datasets are sufficiently large. The goal is to test whether federated aggregation remains beneficial once each client already has enough data to learn a competent policy on its own, and whether richer observations change how strongly policies specialize to individual rooms. Table IV and Table V report the Near-IID sufficient data results for the 4-beam and 8-beam settings, respectively.

4-beam results:

Dataset	Room1	Room2	Room3	Room4	Room5	Avg
Room1	**53**	25	41	16	34	34
Room2	12	**60**	32	29	37	34
Room3	8	23	**68**	14	22	27
Room4	15	32	26	**61**	35	34
Centralized	40	50	55	39	**44**	**46**
Federated	41	50	54	40	40	45

TABLE IV. Near-IID data sufficient SR with 4-beam observations. Each local model is trained on demonstrations from a single room within the same room family, while the centralized and federated models are trained on the combined client data. Room5 is held out as an unseen room from the same environment class to evaluate within class generalization. The results show strong diagonal dominance for the local models, indicating substantial specialization to the training room, whereas the aggregated models achieve the strongest average performance across all rooms.

8-beam results:

Dataset	Room1	Room2	Room3	Room4	Room5	Avg
Room1	29	40	53	37	**55**	43
Room2	18	43	49	31	43	37
Room3	13	45	52	29	40	36
Room4	13	47	50	**39**	48	39
Centralized	26	50	57	**39**	46	44
Federated	**30**	**52**	**61**	**39**	53	**47**

TABLE V. Near-IID data sufficient SR with 8-beam observations. Relative to the 4-beam setting, the local models exhibit less diagonal dominance and more balanced cross room performance, suggesting that the 8-beam representation supports more transferable navigation behaviour. Among the aggregated regimes, the federated model achieves the highest average success across rooms.

a) Key takeaway: In the **4-beam** setting, the local models still show clear room specific specialization. Each local policy performs best on its own training room (diagonal entries of 53–68%), while transfer to the other rooms remains much weaker, including to the held out Room5 (22–37%). This indicates that even within a Near-IID room class, the 4-beam representation encourages policies that depend strongly on the particular geometry of the training environment. Both aggregated regimes improve the average performance substantially relative to the local models, with Centralized and Federated reaching 46% and 45% average success, respectively. However, this gain should be interpreted carefully: although aggregation clearly helps on average, it does not produce a dramatic improvement overall. In other words, training on more combined data improves robustness, but it does not fundamentally solve generalization within the room family.

The **8-beam** results exhibit a noticeably different pattern. First, the local models are much less dominated by the diagonal than in the 4-beam setting. In several cases, a model performs as well as or better on another room than on the room it was trained on. For example, the policy trained on Room1 performs best on the unseen Room5 (55%) rather than on Room1 itself (29%), and the policy trained on Room4 performs better on Room2, Room3, and Room5 (47%, 50%, and 48%) than on Room4 (39%). This suggests that, under the 8-beam representation, performance is influenced at least as much by the intrinsic difficulty of a map as by train–test identity. Put differently, the policy appears to be learning a more transferable navigation behaviour, driving through free space while avoiding obstacles, rather than primarily memorizing room specific action patterns.

This interpretation is also consistent with the improved Room5 generalization. The best local 8-beam model reaches 55% on the held out room, substantially above the best 4-beam local result of 37%. Federated training achieves the strongest overall average performance at 47% and remains competitive on Room5 at 53%, while Centralized reaches 44% on average and 46% on Room5. Thus, in the 8-beam setting, FedAvg is the best aggregate method overall and produces the strongest average generalization across rooms.

At the same time, the gains from aggregation remain relatively modest given that the aggregated models are trained on substantially more data than any individual local policy. This is somewhat surprising. One might expect that combining all room datasets would produce a clearly superior model, yet the improvement over the best local models is limited and not consistent on every room. In particular, the best local model still outperforms both aggregated approaches on Room5. This suggests that, once data are already sufficient and the environment class is relatively consistent, simply adding more data through aggregation does not automatically translate into much better navigation performance. Instead, the limiting factors may be representation or model capacity.

Overall, these results show a clear contrast between the two representations. With 4-beam observations, the policies appear to remain strongly tied to the geometry of the specific training room, whereas with 8-beam observations they generalize more evenly across rooms and sometimes perform better on different environments than on the training environment itself. This makes the 8-beam setting more consistent with learning a transferable navigation skill, even though aggregation still yields only moderate benefits in the data sufficient regime.

3) Near-IID (data scarce): This scenario evaluates whether federated aggregation can mitigate severe local data scarcity when the client environments remain Near-IID. All clients are trained on room maps drawn from the same environment class, but with substantially fewer demonstrations than in the data sufficient setting. The key question is whether aggregation can compensate for the lack of local coverage and whether this effect depends on the observation/action representation. Table VI and Table VII report the Near-IID limited data results for the 4-beam and 8-beam settings, respectively.

4-beam results:

Dataset	Room1	Room2	Room3	Room4	Room5	Avg
Room1	16	10	12	9	9	11
Room2	5	24	15	13	16	15
Room3	3	7	16	10	6	8
Room4	2	3	6	15	15	8
Room5	4	12	18	7	**32**	15
Centralized	**20**	**30**	**38**	**25**	30	**29**
Federated	15	27	31	24	25	24

TABLE VI. Near-IID data scarce success rates (%) with 4-beam observations. Each local model is trained on a reduced dataset from a single room, while the centralized and federated models are trained on the union of these scarce client datasets. The results show that aggregation substantially improves performance over purely local training, although centralized training remains stronger than FedAvg in this sparse 4-beam regime.

8-beam results:

Dataset	Room1	Room2	Room3	Room4	Room5	Avg
Room1	18	34	41	27	39	32
Room2	14	33	42	26	42	31
Room3	**19**	30	40	28	39	31
Room4	15	25	44	29	39	30
Room5	15	26	38	23	41	29
Centralized	18	36	44	**33**	41	34
Federated	17	**37**	**47**	31	**46**	**36**

TABLE VII. Near-IID data scarce success rates (%) with 8-beam observations. Compared with the 4-beam setting, both local and aggregated models are markedly more stable across rooms, and the federated model achieves the best overall average success. This suggests that the richer 8-beam representation provides a stronger learning signal under scarcity and makes federated aggregation more effective.

a) Key takeaway: In the **4-beam** data scarce regime, local training largely collapses. Individual models achieve only 8–15% average success, indicating that each client lacks sufficient data to learn robust obstacle avoidance and recovery behaviour from its own demonstrations alone. The strong diagonal structure seen in the data sufficient setting is also much weaker here, suggesting that the local policies are not even learning reliable room specific behaviours under such sparse supervision. Aggregation is therefore essential: Centralized training improves performance substantially on

every room and reaches the best overall average success of 29%, while Federated training also yields a clear gain over every local baseline with an average success of 24%. However, in this sparse 4-beam setting, FedAvg does not fully match centralized training and typically trails it by several percentage points on each room. This suggests that when both representation and local data are limited, the optimization noise introduced by federated averaging is harder to overcome.

The **8-beam** setting changes the picture considerably. First, performance improves for all regimes, including the local models, whose average success now lies between 29% and 32%. Second, the local 8-beam models are relatively consistent across rooms, with much smaller performance differences between train and test environments than in the 4-beam case. This again suggests that the richer representation encourages learning a more transferable navigation behaviour rather than relying primarily on room specific regularities. Most importantly, the aggregated methods remain the strongest overall, and the gap between federated and centralized training narrows substantially. In fact, Federated achieves the best average success at 36%, outperforming Centralized at 34%, and leads on Rooms 2, 3, and 5.

These results indicate that under Near-IID but data-scarce conditions, aggregation is highly beneficial, but its effectiveness depends strongly on the representation. With 4-beam observations, federated learning helps but remains clearly below centralized training. With 8-beam observations, the benefit of aggregation becomes stronger and FedAvg becomes fully competitive, even slightly outperforming centralized training on average. A plausible interpretation is that the 8-beam representation, together with the richer turning behaviour in the dataset, provides a more informative learning signal under scarcity, allowing the federated model to combine complementary client experience more effectively. Overall, this scenario provides the clearest evidence that federated training is most useful when local data are limited, and that its success is amplified by a representation that supports more transferable navigation behaviour. To further assess the robustness of these findings under a more conventional execution scheme, additional controller based tests are provided in Appendix C.

C. Training dynamics: centralized vs federated

To compare optimization behaviour between centralized and federated training, we plot the training action MSE (log scale) as a function of training progress for each scenario and sensing configuration (Figure 8). Centralized training is shown as a function of epochs on pooled data. For federated training, the horizontal axis is federated rounds, where one round corresponds to two local epochs of client side training followed by server aggregation (FedAvg).

a) Key takeaway: Across all settings, centralized training converges to a lower training action MSE than federated training, with the gap being small in 4-beam Non-IID (both curves approach similar values) but pronounced in the near-IID regimes where the federated curve plateaus at a higher error. In the 8-beam experiments, federated training also exhibits higher variance and intermittent spikes, consistent with noisier optimization under client wise updates and aggregation (e.g., client drift and round to round update inconsistency). Importantly, these convergence curves do not perfectly predict closed loop success: despite higher training MSE, federated models can achieve comparable (and in some cases slightly better) generalization success rates on held out environments, indicating that imitation loss alone is not a sufficient proxy for robustness in closed loop navigation.

For completeness, we include the corresponding training loss curves in Appendix B (Figure 12).

VIII. Discussion

This thesis investigates whether diffusion based control policies can be trained with federated learning while maintaining reliable closed loop navigation under two key challenges: (i) heterogeneous client data distributions and (ii) limited local demonstration data. We study these effects under two sensing configurations, a sparse 4-beam observation model and a higher fidelity 8-beam variant, to assess how observation richness interacts with federated optimization and generalization.

A. Interpreting the results across scenarios

Across the three scenarios in Section VI, the primary question is when federated training improves over local only training and how close it approaches centralized training. The results in Section VII suggest the following qualitative picture:

- **Near-IID, sufficient data:** When each client has sufficient demonstrations, the results show a clear interaction between aggregation and sensing fidelity. In the 4-beam setting, local training yields the best in domain performance: for each training room (Rooms 1–4) the locally trained model achieves the highest success rate on its own environment (diagonal entries 53–68%), exceeding both centralized and federated models. However, this specialization does not transfer reliably: the aggregated regimes provide better overall generalization (Avg 45–46%) and perform best on the held out Room5, where centralized slightly outperforms federated (44% vs. 40%), suggesting that FedAvg offers limited benefit beyond pooling in this configuration.
  In the 8-beam setting, the picture changes: local models become more uniform across rooms (reduced diagonal dominance) and the benefit of aggregation is more consistent. Federated training attains the best overall average (47%) and strong Room5 performance (53%), while the best local model (Room1) remains highly competitive on Room5 (55%). Overall, under near-IID and data sufficient conditions, 4-beam encourages environment specific specialization, whereas 8-beam promotes more transferable behaviour and allows FedAvg to match or slightly surpass centralized training on average.
- **Near-IID, data scarcity:** This is the regime where federated learning provides the clearest benefit over local only training. With 4-beam, local models largely collapse (Avg 8–15%), while Centralized (Avg 29%) and Federated (Avg 24%) substantially improve success rates across

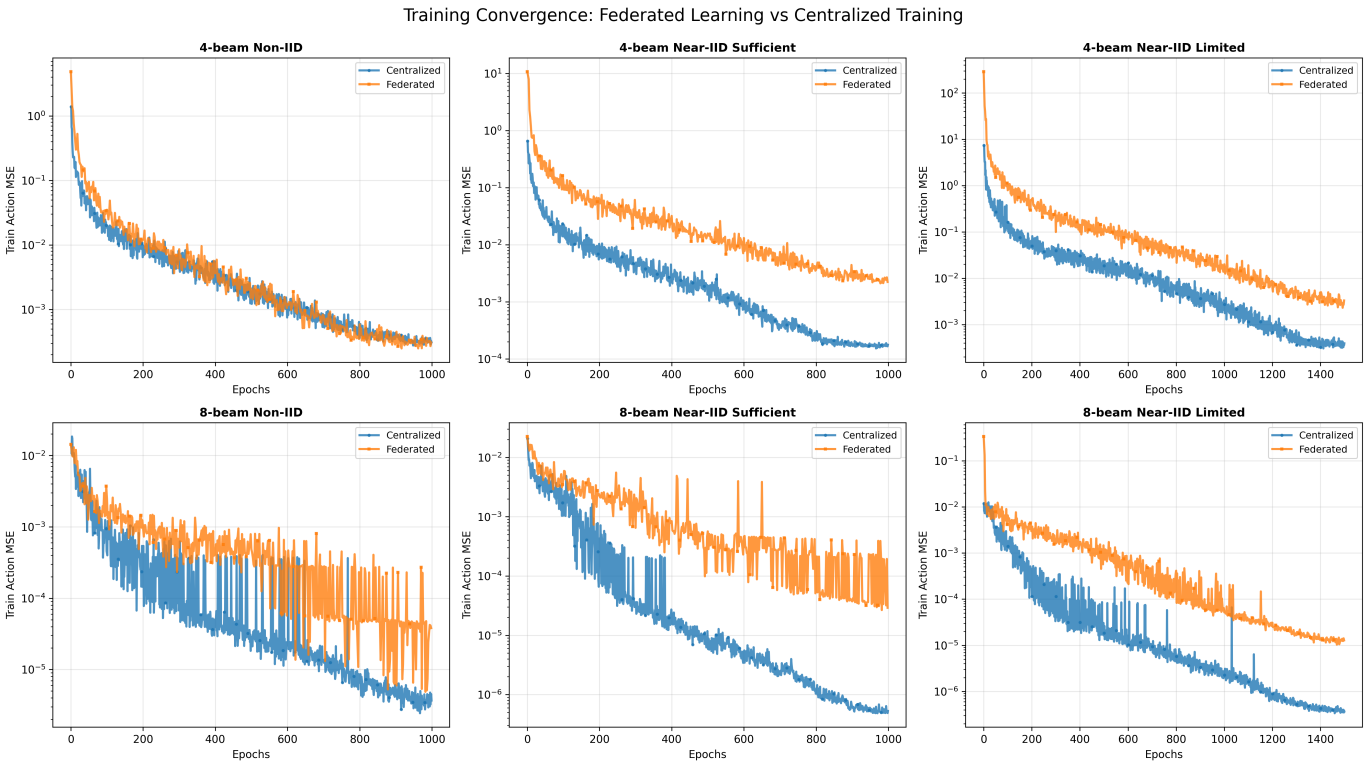

Fig. 8. Training convergence comparison between centralized and federated learning. Each subplot shows the training action MSE (log scale). Centralized curves are plotted versus epochs on pooled data. Federated curves are plotted versus federated rounds; one round corresponds to two local epochs followed by aggregation. Top row: 4-beam; bottom row: 8-beam.

all rooms, indicating that aggregation mitigates missing experience on each client. With 8-beam, all methods improve, consistent with the turn augmented representation providing a stronger learning signal under limited data; importantly, Federated becomes the best overall method (Avg 36%) and outperforms centralized on several rooms (notably Rooms 2, 3, and 5). Thus, when local datasets are small, federated aggregation improves robustness, and its advantage is amplified when the data/representation explicitly teaches turning and recovery behaviours.

- **Non-IID compositional generalization:** In the 4-beam setting, individual models show strong same environment performance but poor cross primitive transfer (e.g., the Corridor-only model fails on L-turn and Room), while Centralized and Federated achieve near ceiling performance on the primitive environments (95–97%) yet remain limited on the composed Room (46–51%). This indicates that the main difficulty is not learning each primitive skill in isolation, but composing them reliably in a novel layout. In the 8-beam setting, cross environment performance becomes more uniform and the local models are less specialized to their training primitive. The L-turn and Cross local models each achieve an average success rate of 73%, and both transfer reasonably to Room (49% and 53%, respectively), suggesting that the 8-beam representation captures a more general driving and obstacle avoidance behaviour rather than memorizing a single primitive. Among the aggregated regimes, Federated attains the best overall average success (76%) and performs best on L-turn and Cross (75% and 87%), while Centralized achieves the highest success on the composed Room environment (56%). Relative to the 4-beam setting, both aggregated methods improve on Room (Centralized: 51% $\rightarrow$ 56%, Federated: 46% $\rightarrow$ 52%), but they lose some reliability on the simplest primitive, Corridor. Taken together, the Non-IID results suggest a trade off: richer turning behaviour improves transfer and compositional performance, but may reduce peak robustness on the easiest cases.

Finally, the training dynamics in Figure 8 are consistent with these trends: centralized training typically reaches lower training MSE than federated training, especially in the near-IID settings, yet lower imitation error does not fully predict closed loop success, most notably in the 8-beam data scarce regime where federated models achieve the highest success rates despite higher training error.

B. *Implementation sensitivities and practical limitations*

Beyond the scenario level trends, three implementation level effects were observed that materially influence the absolute success rates and the comparability of the 4-beam and 8-beam formulations.

a) Factors Limiting Absolute Success Rates: Across scenarios, absolute success rates remain moderate even for centralized training. This is consistent with several limitations of the current setup: (i) the observation is intentionally low dimensional (4 or 8 beams), which leaves the policy partially

observable near corners and in clutter; (ii) behaviour cloning with a diffusion objective can suffer from compounding errors in closed loop execution, especially when the expert demonstrations under represent recovery states; (iii) the evaluation metric is unforgiving (all failures are collisions), so small control deviations can dominate SR. Several straightforward directions could raise performance: increase observation richness (more beams or a higher resolution LiDAR downsample), expand the dataset with recovery focused trajectories (near wall starts, larger heading perturbations), reduce the execution horizon T_a to re-plan more frequently, and tune diffusion inference (e.g., fewer/more denoising steps or modified sampling noise). More substantial improvements would likely require closed loop data aggregation (DAgger style relabeling) or adding a lightweight safety layer (e.g., collision checking or a conservative local controller) that filters unsafe actions while preserving the learned policy's nominal behaviour.

b) Action execution mismatch: direct state update vs. controller-based unicycle dynamics: For comparability across representations, the main results in this thesis apply the predicted actions through direct state updates that match the action parametrization used during training (Section V-C3), rather than executing them through a low level controller and continuous unicycle integration. Additional controller based experiments, reported in Appendix C, show that this choice can materially affect the observed performance. In the 4-beam setting, replacing direct updates with a standard proportional tracking controller and unicycle dynamics does not degrade performance and can even improve it slightly; for example, in the Near-IID data sufficient setting, the Room1-trained model improves from 53% to 69% on Room1, while the centralized and federated models increase from 46% and 45% to 53% and 52%, respectively. By contrast, the same controller leads to a marked degradation in the 8-beam setting, where success rates drop sharply and most failures become collisions. A plausible explanation is that the two representations interact differently with the controller interface: the 4-beam policy predicts larger target poses that can be smoothed into feasible motion by the controller, whereas the 8-beam policy predicts smaller egocentric increments that work well under direct rollout but become brittle when converted into bounded (v, ω) commands and integrated over time. This suggests that deployment fidelity depends not only on the learned policy itself, but also on how its outputs are mapped to robot dynamics. Improving sim-to-real consistency will therefore require tighter alignment between training targets and execution, for example by training the egocentric policy directly on velocity commands or by rescaling predicted deltas to better match the control time step and actuator limits.

C. Real world analogues of the three scenarios

Unfortunately, testing FedDiff on physical robots was out of scope for this thesis. Nevertheless, the scenario families in Section VI map naturally to several real world fleet learning settings in which data are distributed across agents and cannot be centrally collected. Below, some examples are given.

a) Scenario A: Near-IID, sufficient data (fleet at scale): A large ride hailing fleet (e.g., an Uber like operator) collecting abundant driving trajectories across many vehicles operating in broadly similar urban conditions. Over time, each vehicle accumulates substantial interaction data (effectively "infinite" at fleet scale), and the overall distribution is near-IID in the sense that vehicles share common road structure, traffic rules, and driving behaviours, even though they experience different routes and neighbourhoods. In this setting, centralizing raw logs may be undesirable due to privacy sensitivity (passenger pickup/drop off locations, driving patterns) and regulatory constraints. FedDiff offers a way to train a shared generative policy by aggregating model updates instead of raw trajectories, enabling continual fleet improvement while keeping data local to vehicles.

b) Scenario B: Near-IID, limited data (rare deployment, limited connectivity): Robots deployed in remote facilities (mines, offshore platforms, polar stations) where collecting expert demonstrations is expensive and connectivity is intermittent. Each site has similar navigation structure (near-IID), but only a small number of trajectories can be logged early on. FL helps pool experience across sites without uploading raw logs, improving robustness from sparse data.

c) Scenario C: Non-IID compositional generalization (heterogeneous environments requiring skill composition): A mixed fleet spanning qualitatively different navigation contexts: narrow service corridors, open lobbies, warehouse aisles, and crowded intersections. Individual sites may specialize (a warehouse robot sees aisles; a concierge robot sees lobbies). A new deployment (e.g., a combined hospital wing) requires composing these skills. FL can help if the shared representation supports transfer and if aggregation does not collapse into shortcuts tied to absolute map cues.

D. Trusted server assumption and security implications

FedDiff, as evaluated here, assumes a trusted central server that honestly aggregates client updates (FedAvg) and is not adversarial. This is the standard "cross-silo" FL assumption in many robotics deployments, where a single organization (e.g., hospital or warehouse operator) controls both robots and the coordinating server.

This assumption matters because model updates can leak information about client data, particularly for high capacity models. In a real deployment where the server is not fully trusted (or where clients do not trust each other), additional mechanisms may be required:

- Secure aggregation to prevent the server from inspecting individual client updates.
- Differential privacy (client level or example level) to bound information leakage, at the cost of utility.
- Robust aggregation to tolerate malicious or faulty clients (Byzantine robustness).

These defences are out of scope in this thesis, but are important when translating federated diffusion policies to safety and privacy critical settings.

E. Why federated learning instead of edge only training?

"Edge computing" typically means that inference (and sometimes training) occurs locally on each robot without relying on continuous cloud connectivity. Pure edge only training corresponds to the individual baseline in this thesis: each robot learns from its own data, which is privacy preserving but data limited and prone to brittle behaviour under scarce or biased experience.

Federated learning occupies the middle ground: it preserves the edge constraint on raw data (demonstrations remain local) while enabling collaborative learning through parameter aggregation. Compared to simply "training on the edge" without coordination, FL is preferable when:

- multiple robots experience complementary corner cases (rare failures, recovery manoeuvrers),
- regulations or operational constraints prevent uploading raw sensor logs,
- data transfer costs are high (e.g., large logs, intermittent connectivity),
- a shared policy is desired for maintainability and consistent behaviour across a fleet.

In practice, many deployments combine both: inference and local data collection occur on the edge, while periodic federated rounds synchronize policy improvements.

F. Transfer to the real world: what will break first?

The experiments use simplified 2D occupancy grids, deterministic geometry, and a beam based observation model (Section V). This abstraction captures an important subset of real navigation (local obstacle distances and goal relative geometry), but real world transfer could introduce several failure modes:

a) Sensor realism and the "beam" model: The beams in this thesis are computed by ray marching in a known grid map. In the real world, analogous measurements come from 2D LiDAR, depth sensors, ultrasonic sensors, or radar. Key gaps are:

- Noise and dropouts: real range returns have noise, reflections, and missing values (glass, specular surfaces).
- Dynamic obstacles: humans and carts create transient occlusions that a static map does not capture.
- Extrinsic calibration: beam directions depend on sensor mounting; miscalibration shifts perceived geometry.

A practical sim-to-real step is to train with observation corruption (range noise, random dropouts, spurious returns) and domain randomization over obstacle geometry and sensor pose.

Importantly, the thesis deliberately uses a small number of beams (4 or 8) as a low dimensional proxy rather than attempting to match the full resolution of a physical LiDAR. This can be advantageous for sim-to-real transfer: a real 2D LiDAR provides many more rays per scan, so a deployed system can downsample to the same beam geometry while still retaining redundancy to tolerate dropouts and spurious returns [21]. In other words, the real sensor can supply "extra" measurements that compensate for missing or corrupted rays, whereas the policy remains conditioned on a stable, low dimensional representation.

Dynamic obstacles pose a harder gap because the training environments are static. However, the receding horizon structure of diffusion policies provides a practical mitigation: by reducing the action execution horizon T_a (and replanning more frequently), the controller can react to newly observed obstacles at a higher rate, similar in spirit to classical local planners. This does not remove the need for explicit dynamic obstacle modeling or safety layers, but it can reduce the severity of failures caused by transient occlusions by limiting how long the robot commits to an open loop action sequence.

b) System constraints in real federated deployments: All federated experiments in this thesis are executed in a simulated setting (clients run sequentially on a single machine), and therefore we do not measure end to end networking overhead. In a real deployment, however, communication cost can become a practical bottleneck, especially for parameter heavy diffusion models. Although we do not measure network communication, the parameter payload per client update can be estimated from the number of parameters P and parameter precision. With float32 parameters, a single model upload or download is approximately $4P$ bytes. A full round with $|\mathcal{S}_r|$ participating clients transfers approximately $2 \cdot 4P \cdot |\mathcal{S}_r|$ bytes (server$\rightarrow$clients + clients$\rightarrow$server), ignoring protocol overhead. In our experiments the denoising network has $\approx$ 65M parameters, yielding ≈ 260 MB per upload/download in float32 (or roughly half in float16). This estimate is provided for context only and does not reflect measured wall clock communication.

This consideration partly motivates the architectural choices in FedDiff: a relatively compact denoiser and low dimensional observations avoid the additional parameter and activation costs of a vision encoder and make future real world FL deployments more plausible. Nonetheless, scaling to large fleets or bandwidth constrained links would likely require further systems oriented techniques (e.g., lower precision, update compression, fewer participating clients per round, or less frequent synchronization), which are outside the scope of this thesis.

c) State estimation and the cost of "global" representations: The global coordinate formulation assumes accurate global pose and a global goal. In real systems, this requires localization (e.g., AMCL or SLAM) and map alignment. Localization error can directly induce unsafe actions because the policy conditions on absolute coordinates. The egocentric formulation reduces dependence on global position but still relies on heading and a goal vector; producing that goal vector robustly often requires either (i) a map and localization to compute the relative goal, or (ii) a local goal/waypoint generator (e.g., local planner) that provides a short horizon target.

G. Future work

Several extensions would strengthen real world relevance:

- System-level FL evaluation: measure communication cost and training time per federated round, and test whether

compression or quantization can reduce the cost of transmitting diffusion policy updates.
- Asynchrony and stragglers: evaluate federated averaging methods that remain effective when some clients are slow, delayed, or only participate intermittently.
- Privacy and robustness: incorporate secure aggregation or differential privacy into the federated setup, and quantify the resulting performance trade-off for diffusion based control.
- Richer sensing: replace sparse beam observations with raw 2D LiDAR scans or depth images, and evaluate how this affects navigation performance
- Dynamic obstacles: introduce moving obstacles into the environment and evaluate whether FedDiff can learn collision avoidance behaviour in dynamic settings.

IX. Conclusion

This thesis introduced *FedDiff*, a framework for training diffusion based navigation policies under federated learning, and evaluated whether federated optimization can preserve closed loop performance under (i) heterogeneous client data and (ii) limited local demonstrations. Using a controlled 2D navigation environment, we compared local only, centralized, and FedAvg training across three scenario families and two low dimensional sensing configurations (4-beam and 8-beam), resulting in 36 trained models and a systematic success rate evaluation across multiple environments.

Federated learning generally improved average performance over local only baselines, with the clearest gains under data scarcity. In the near-IID, data scarce regime, local models frequently collapsed to low success rates, whereas federated aggregation substantially increased success and, in the 8-beam setting, achieved the strongest overall performance. In contrast, under near-IID, sufficient data, the benefit of federated learning depended on the sensing representation: with 4-beam observations, local models achieved the best in domain performance on their own training environments, while aggregation primarily improved average cross room generalization and performance on the held out Room5; with 8-beam observations, behaviour became more uniform across rooms and FedAvg matched or slightly exceeded centralized training on average. In the non-IID compositional setting, aggregation enabled near ceiling performance on the primitive environments, but success on the Room environment remained substantially lower, indicating that composition rather than primitive skill acquisition is the dominant challenge for diffusion policies under heterogeneous training.

A central qualitative finding is that representation and dataset composition strongly shape generalization under FL. The 8-beam formulation, which introduces richer turning behaviour, produced more stable cross environment performance for individual models and modestly improved room like environment success, but it also reduced reliability on the simplest primitive environments. Moreover, training dynamics showed that federated optimization typically converged to a higher training error than centralized training, yet this gap did not reliably predict closed loop success, highlighting that imitation loss alone is an imperfect proxy for robust navigation. Finally, several practical sensitivities were identified, including the limited expressiveness of sparse beam observations, and mismatches between action parametrizations and unicycle dynamics execution, which help explain the moderate absolute success rates and point to concrete avenues for improving sim-to-real fidelity.

Overall, the results support the conclusion that federated learning is a viable training paradigm for diffusion based navigation policies when the goal is to leverage distributed demonstrations without centralizing data, particularly under local data scarcity. However, achieving consistently high success rates and robust real world deployment will require tighter alignment between training targets and execution dynamics, richer or more robust observation models, and additional mechanisms for safety and recovery in closed loop operation. Future work should extend FedDiff to dynamics realistic simulators and physical robots, incorporate communication and privacy preserving FL mechanisms (e.g., compression and secure aggregation), and investigate methods that explicitly encourage combining skills and recovery behaviour under heterogeneous data distributions.

References

[1] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html

[2] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-Resolution Image Synthesis with Latent Diffusion Models," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, Jun. 2022, pp. 10 674–10 685. [Online]. Available: https://ieeexplore.ieee.org/document/9878449/

[3] J. Sohl-Dickstein, E. A. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep Unsupervised Learning using Nonequilibrium Thermodynamics," 2015. [Online]. Available: https://proceedings.mlr.press/v37/sohl-dickstein15.pdf

[4] Y. Song and S. Ermon, "Generative Modeling by Estimating Gradients of the Data Distribution," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/hash/3001ef257407d5a371a96dcd947c7d93-Abstract.html

[5] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," Oct. 2022, arXiv:2010.02502 [cs]. [Online]. Available: http://arxiv.org/abs/2010.02502

[6] M. Janner, Y. Du, J. B. Tenenbaum, and S. Levine, "Planning with Diffusion for Flexible Behavior Synthesis," Dec. 2022. [Online]. Available: http://arxiv.org/abs/2205.09991

[7] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion," Jun. 2023, arXiv:2303.04137 [cs] version: 4. [Online]. Available: http://arxiv.org/abs/2303.04137

[8] J. Liu, M. Stamatopoulou, and D. Kanoulas, "DiPPeR: Diffusion-based 2D Path Planner applied on Legged Robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 9264–9270. [Online]. Available: https://ieeexplore.ieee.org/document/10610013/

[9] Z. Dong, J. Hao, Y. Yuan, F. Ni, Y. Wang, P. Li, and Y. Zheng, "DiffuserLite: Towards Real-time Diffusion Planning." [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2024/file/dd6a47bc0aad6f34aa5e77706d90cdc4-Paper-Conference.pdf

[10] M. Reuss, M. Li, X. Jia, and R. Lioutikov, "Goal-Conditioned Imitation Learning using Score-based Diffusion Policies," Jun. 2023, arXiv:2304.02532 [cs]. [Online]. Available: http://arxiv.org/abs/2304.02532

[11] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. PMLR, Apr.

2017, pp. 1273–1282, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html

[12] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated Learning: Challenges, Methods, and Future Directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, May 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9084352/

[13] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic Controlled Averaging for Federated Learning," in *Proceedings of the 37th International Conference on Machine Learning*. PMLR, Nov. 2020, pp. 5132–5143. [Online]. Available: https://proceedings.mlr.press/v119/karimireddy20a.html

[14] J. Wang, Q. Liu, H. Liang, and G. Joshi, "Tackling the Objective Inconsistency Problem in Heterogeneous Federated Optimization." [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/564127c03caab942e503ee6f810f54fd-Paper.pdf

[15] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical Secure Aggregation for Privacy-Preserving Machine Learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. Dallas Texas USA: ACM, Oct. 2017, pp. 1175–1191. [Online]. Available: https://dl.acm.org/doi/10.1145/3133956.3133982

[16] R. C. Geyer, T. Klein, and M. Nabi, "Differentially Private Federated Learning: A Client Level Perspective," Mar. 2018, arXiv:1712.07557 [cs]. [Online]. Available: http://arxiv.org/abs/1712.07557

[17] N. Truong, K. Sun, S. Wang, F. Guitton, and Y. Guo, "Privacy preservation in federated learning: An insightful survey from the GDPR perspective," *Computers & Security*, vol. 110, p. 102402, Nov. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404821002261

[18] Y. Naveh, P. Z. Bar-Yoseph, and Y. Halevi, "Nonlinear Modeling and Control of a Unicycle," *Dynamics and Control*, vol. 9, no. 4, pp. 279–296, Oct. 1999. [Online]. Available: https://doi.org/10.1023/A:1026481216262

[19] P. E. Hart, N. J. Nilsson, and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, Jul. 1968. [Online]. Available: https://ieeexplore.ieee.org/document/4082128/

[20] P. Gonçalves, P. Torres, C. Alves, F. Mondada, M. Bonani, X. Raemy, J. Pugh, C. Cianci, A. Klaptocz, S. Magnenat, J.-C. Zufferey, D. Floreano, and A. Martinoli, "The e-puck, a Robot Designed for Education in Engineering," *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, vol. 1, Jan. 2009.

[21] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 31–36, iSSN: 2153-0866. [Online]. Available: https://ieeexplore.ieee.org/document/8202134/

[22] A. Q. Nichol and P. Dhariwal, "Improved Denoising Diffusion Probabilistic Models," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, Jul. 2021, pp. 8162–8171, iSSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v139/nichol21a.html

[23] I. Loshchilov and F. Hutter, "Decoupled Weight Decay Regularization," Jan. 2019, arXiv:1711.05101 [cs]. [Online]. Available: http://arxiv.org/abs/1711.05101

[24] Delft High Performance Computing Centre (DHPC), "DelftBlue Supercomputer (Phase 2)," 2024. [Online]. Available: https://www.tudelft.nl/dhpc

[25] J. Lin, "Divergence measures based on the Shannon entropy," *IEEE Transactions on Information Theory*, vol. 37, no. 1, pp. 145–151, Jan. 1991. [Online]. Available: https://ieeexplore.ieee.org/document/61115/

APPENDIX A
QUANTIFYING CLIENT HETEROGENEITY VIA $\Delta\theta$ DISTRIBUTIONS

This appendix provides the quantitative justification for the heterogeneity regimes introduced in Section VI-B. In the main text, the terms *near-IID* and *non-IID* are used as operational labels for the relative degree of client heterogeneity induced by the benchmark split. They are not intended as strict statistical claims about the underlying data generating process. Instead, they describe two controlled experimental regimes with comparatively low and high client heterogeneity, respectively.

To support this distinction empirically, we quantify heterogeneity directly from the demonstration datasets used for training.

A. Behavioural Representation

We compare datasets along a task relevant, frame invariant behavioural axis: the per step heading change $\Delta\theta$ between consecutive trajectory segments. Intuitively, $\Delta\theta$ captures how turn dominated a dataset is, which is a key distinction between corridor like navigation and more intersection or corner heavy navigation.

For each dataset i, we construct a normalized histogram P_i over $\Delta\theta \in [-\pi, \pi]$.

B. Dissimilarity Measure

Pairwise dissimilarity between datasets i and j is measured using the Jensen–Shannon divergence (JSD) [25]:

$$\mathrm{JSD}(P_i, P_j) = \frac{1}{2} D_{\mathrm{KL}}(P_i \,\|\, M) + \frac{1}{2} D_{\mathrm{KL}}(P_j \,\|\, M),$$

where $M = \frac{1}{2}(P_i + P_j)$ is the average histogram and $D_{\mathrm{KL}}(\cdot \,\|\, \cdot)$ denotes the Kullback–Leibler divergence.

To reduce sensitivity to unequal dataset sizes, each pairwise divergence is estimated using bootstrap subsampling. For every dataset pair (i, j), we repeatedly draw the same number of time steps from both datasets, compute the JSD for each resample, and report the bootstrap mean.

C. Results

Figures 9 and 10 summarize the resulting divergences. Figure 9 shows the full pairwise divergence matrix for the primitive datasets (Corridor, L-turn, Cross) and the room datasets (Room1–Room4). The room family forms a tight low divergence cluster, with typical values on the order of 10^{-3}, indicating relatively high behavioural homogeneity within that split. By contrast, the primitive family is substantially more heterogeneous, with Corridor differing most strongly from the turn dominated datasets.

Figure 10 summarizes the unique pairwise divergences by comparison type. The mean within primitive divergence ($\mu \approx 9 \times 10^{-3}$) is roughly an order of magnitude larger than the mean within room divergence ($\mu \approx 1 \times 10^{-3}$). This supports the use of the term *near-IID* for the room split and *non-IID* for the primitive split, in a relative and benchmark-specific sense along this behavioural axis.

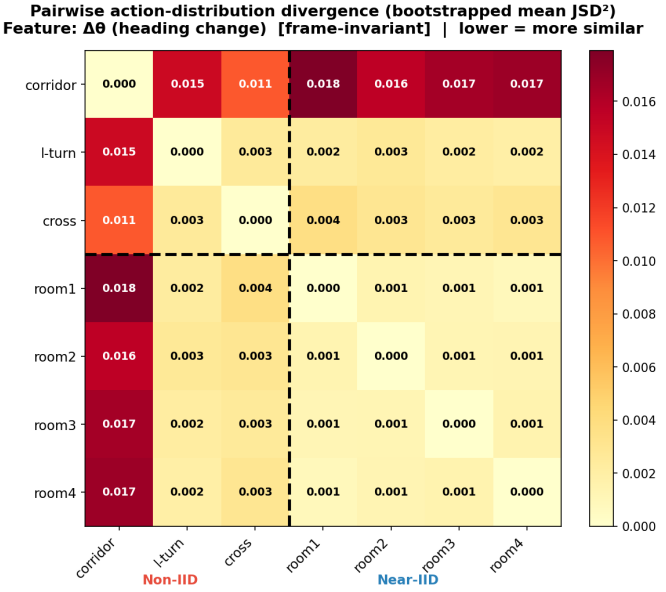

Fig. 9. Bootstrapped mean pairwise Jensen–Shannon divergence between heading-change histograms for all demonstration datasets. Lower values indicate more similar behavioural distributions. Dashed lines separate the primitive split (Corridor, L-turn, Cross) from the room split (Room1–Room4).

The primitive split is also visibly multimodal in $\Delta\theta$-space. Figure 11 shows the corresponding per dataset $\Delta\theta$ histograms. Corridor trajectories are concentrated near $\Delta\theta \approx 0$, reflecting predominantly straight motion, whereas the L-turn and Cross datasets contain substantially more turning and are therefore much closer to one another. Consequently, the within primitive dispersion is driven primarily by the separation between Corridor and the turn dominated primitive datasets. This also explains why some cross group comparisons can be comparable to, or slightly smaller than, the within primitive mean: along the single behavioural axis captured by $\Delta\theta$, the room datasets lie between these two extremes.

Overall, this analysis does not claim that the room clients are strictly IID. Rather, it provides a quantitative justification that the room split exhibits substantially lower client heterogeneity than the Corridor/L-turn/Cross split, consistent with the intended experimental design.

a) Remark.: The near-IID and non-IID labels should therefore be interpreted as relative descriptors of heterogeneity within this benchmark, rather than formal assumptions of independence and identical distribution across clients.

APPENDIX B
ADDITIONAL TRAINING DYNAMICS

Figure 12 reports the evolution of the training loss for centralized and federated training across all scenarios and sensing configurations. The qualitative trend matches the action MSE convergence shown in the main text: centralized training typically reaches a lower final loss, while federated training converges more slowly and often plateaus at a higher value, reflecting the additional optimization noise introduced by client wise updates and aggregation. Importantly, lower training loss does not necessarily translate to higher closed

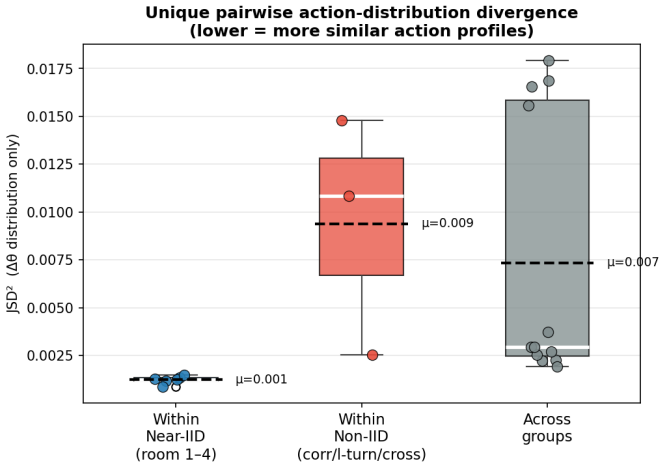

Fig. 10. Distribution of unique pairwise heading-change divergences, grouped into within-room, within-primitive, and cross-group comparisons. Dashed horizontal lines indicate group means.

loop success, so these curves are included primarily to document optimization behavior and to provide a complete view of training convergence under FedAvg.

APPENDIX C
ADDITIONAL EVALUATION UNDER CONTROLLER BASED EXECUTION

The main experiments in this thesis evaluate policy rollouts by directly applying the predicted action representation used by each model. To assess whether the conclusions are robust to a more conventional execution scheme, an additional evaluation was performed in which predicted target states were tracked by a standard proportional controller and the robot state was updated through unicycle dynamics.

Given a predicted target pose $(x^\star, y^\star, \theta^\star)$, the controller first computes the Euclidean distance to the target position and a desired heading toward the target point. Let

$$d = \sqrt{(x^\star - x)^2 + (y^\star - y)^2}, \tag{14}$$

and let the heading-to-target error and orientation error be

$$\begin{aligned} e_h &= \mathrm{wrap}\big(\mathrm{atan2}(y^\star - y, x^\star - x) - \theta\big), \\ e_\theta &= \mathrm{wrap}(\theta^\star - \theta), \end{aligned} \tag{15}$$

where $\mathrm{wrap}(\cdot)$ maps angles to $[-\pi, \pi]$. The controller blends these two terms as

$$e = 0.8\, e_h + 0.2\, e_\theta. \tag{16}$$

This blended angular error was introduced to account for the kinematic constraints of a unicycle robot. In particular, the model predicted orientation can be locally correct even when the robot is not yet facing the next target position. This is especially relevant when the robot is far from the goal: the terminal orientation error may be small, i.e. $e_\theta \approx 0$, even though the robot must first rotate toward the next point before it can move forward. A controller based only on e_θ would therefore underestimate the turning required for position tracking. By weighting e_h more strongly than e_θ, the controller prioritizes turning toward the next target point while still encouraging alignment with the predicted final orientation near the end of the motion segment. Intuitively, this makes the behaviour closer to what is required for a unicycle system: turn toward the waypoint, move toward it, and then align with the desired heading.

If the angular error is large, the robot rotates in place first; otherwise it moves forward while correcting its heading:

$$(v, \omega) = \begin{cases} (0,\ k_\theta e), & |e| > \pi/20, \\ (k_p d,\ k_\theta e), & \text{otherwise}, \end{cases} \tag{17}$$

with $k_p = 5.0$ and $k_\theta = 3.0$. The commanded velocities are clipped to $v \in [0, 3.0]$ and $\omega \in [-\pi, \pi]$, after which the state is updated using standard unicycle integration:

$$\begin{aligned} x_{t+1} &= x_t + v\cos(\theta_t)\,\Delta t, \\ y_{t+1} &= y_t + v\sin(\theta_t)\,\Delta t, \\ \theta_{t+1} &= \mathrm{wrap}(\theta_t + \omega\Delta t). \end{aligned} \tag{18}$$

This experiment is not intended as a new benchmark, but as a robustness check on the execution model. In particular, it tests whether policies that perform well under the direct rollout procedure used in the main experiments remain effective when their predictions are interpreted through a conventional low level controller.

a) 4-beam results: Table VIII reports the results for the Near-IID 4-beam setting under controller based execution. In contrast to the 8-beam case, the 4-beam models remain stable under this execution scheme and do not exhibit a performance drop relative to the main evaluation. If anything, the aggregated models improve slightly: the centralized model increases from 46% to 53% average success, and the federated model increases from 45% to 52%. This suggests that the coarser 4-beam policy outputs are reasonably compatible with the proportional controller and unicycle integration used here. Although overall success remains moderate, almost all failures are due to collisions rather than timeout, indicating that the controller generally preserves forward progress but still struggles to maintain sufficient clearance in cluttered regions.

Dataset	Room1	Room2	Room3	Room4	Room5	Avg	v clipped	ω clipped
Room1	69	27	32	20	38	37	65	6
Room2	12	68	28	31	34	35	68	5
Room3	10	30	62	14	22	27	67	5
Room4	19	28	23	71	38	37	69	4
Centralized	54	62	52	47	51	53	69	4
Federated	51	63	52	44	49	52	69	4

TABLE VIII. Near-IID data sufficient success rates with 4-beam observations under controller based execution.

A notable detail is that the linear velocity command is clipped in a large fraction of controller steps for all 4-beam models, whereas angular clipping remains rare. This suggests that the 4-beam predictions frequently imply relatively large translational corrections when interpreted as controller targets. In practice, this produces aggressive forward motion that is often sufficient to reach the general vicinity of the goal, but also increases the chance of collision in cluttered parts of the map.

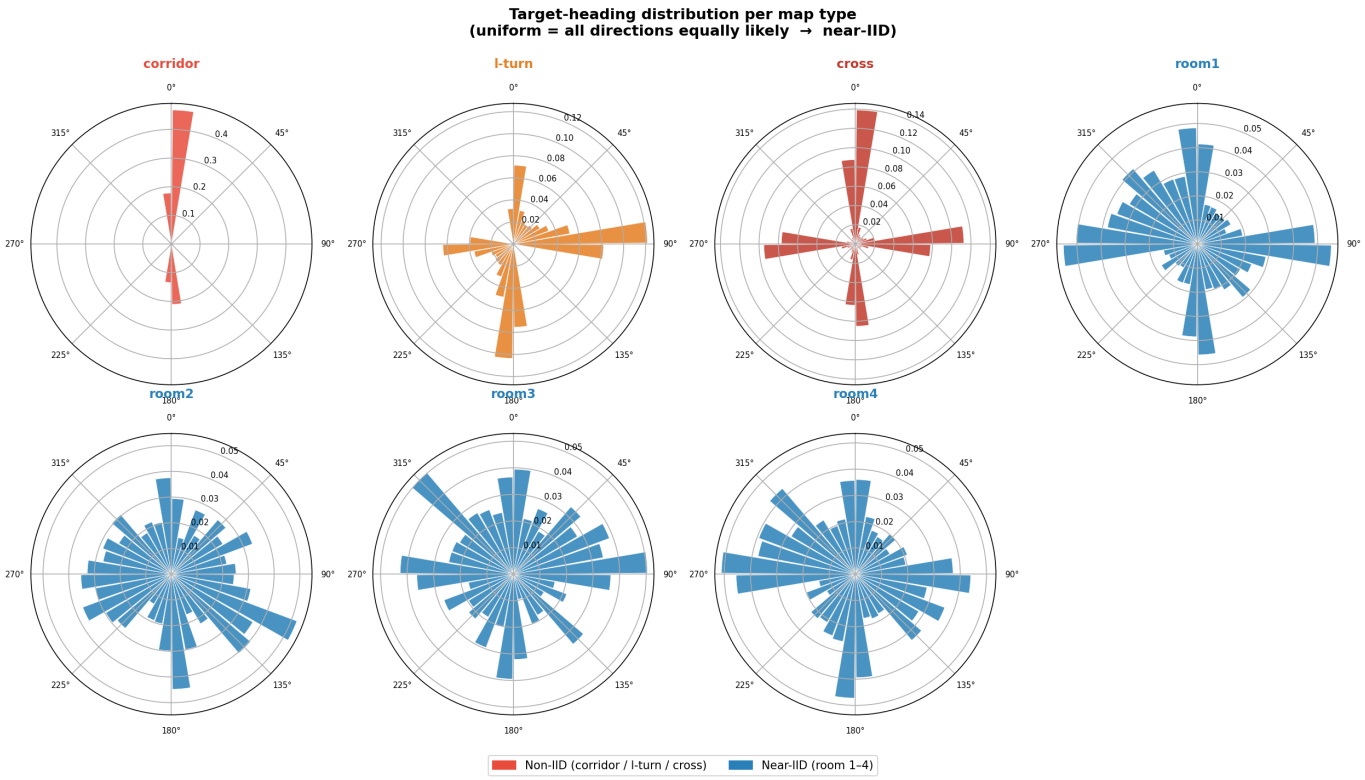

Fig. 11. Per-dataset heading-change ($\Delta\theta$) histograms for the primitive and room datasets used in the heterogeneity analysis. Corridor trajectories are concentrated near $\Delta\theta \approx 0$, whereas L-turn and Cross exhibit broader turning distributions. The room datasets are more similar to one another, consistent with their lower pairwise divergence.

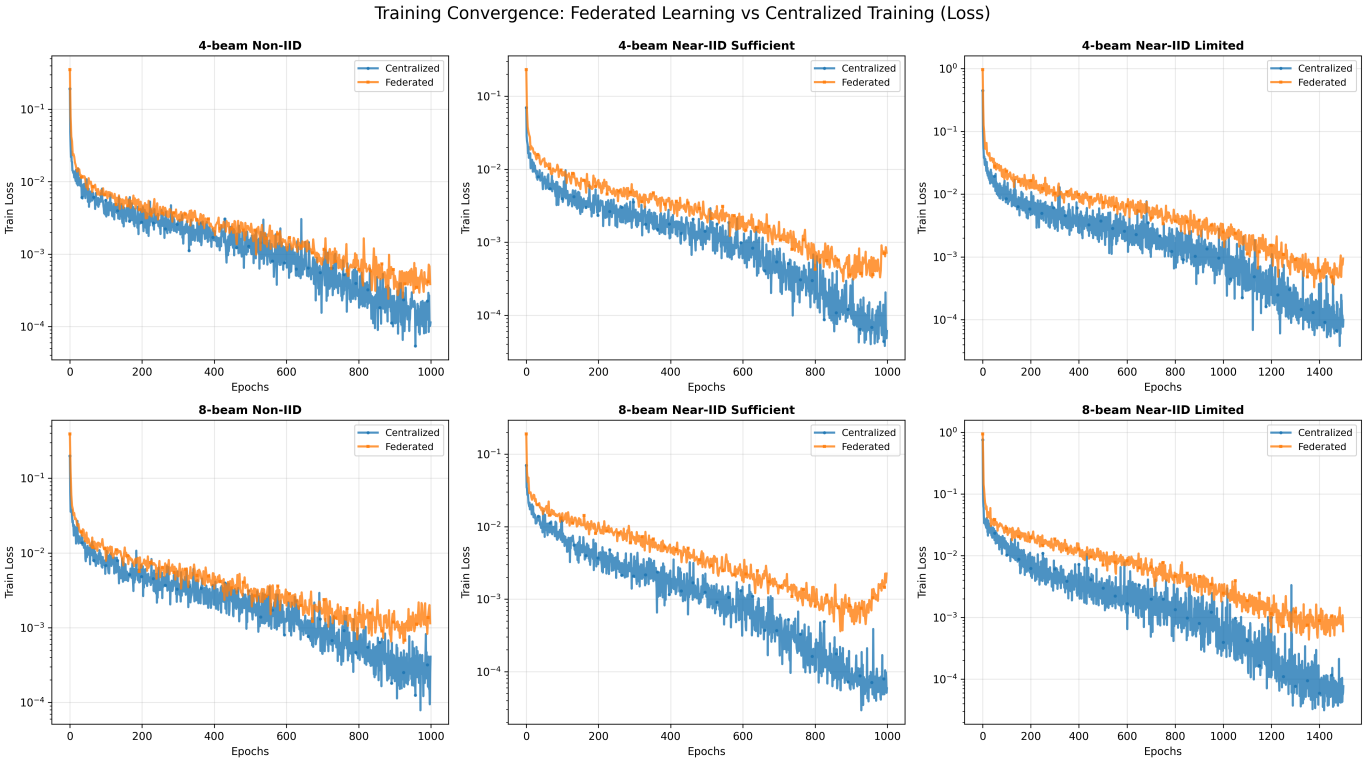

Fig. 12. Training loss comparison between centralized and federated learning. Centralized curves are plotted versus epochs on pooled data. Federated curves are plotted versus federated rounds; one round corresponds to two local epochs followed by aggregation. Top row: 4-beam; bottom row: 8-beam. This figure is provided for completeness; main text comparisons use the training action MSE metric (Figure 8).

b) 8-beam results: The same controller performed poorly for the 8-beam setting. Success rates dropped to 6.9%–18.8% across the tested room maps, with collision rates between 65.0% and 90.5%, and only a small fraction of runs ending in timeout. In contrast to the 4-beam case, the linear velocity command was never clipped, while angular clipping remained limited to approximately 3–4% of controller steps. This strongly suggests that the predicted target displacements in the 8-beam formulation are much smaller and are therefore interpreted by the controller as very conservative motion commands.

A plausible explanation is that the 8-beam policy produces shorter, more finely resolved action increments that are compatible with the direct rollout procedure used in the main experiments, but are not well matched to this simple proportional tracking controller. Under controller based execution, these small increments appear insufficient to produce stable progress through cluttered scenes, causing the robot to accumulate local tracking errors and collide before recovery is possible. Increasing the number of controller substeps did not resolve this issue, and increasing k_p further worsened performance, suggesting that the failure is not simply due to insufficient controller aggressiveness.

Overall, these controller tests show that the evaluation protocol matters. The conclusions of the main experiments remain valid for the direct rollout setting studied in this thesis, but the appendix results indicate that policy quality does not translate trivially across execution models. In particular, the 8-beam formulation appears more sensitive to how predicted actions are converted into physical motion, which should be taken into account in future work on deployment oriented evaluation.