AIAA SciTech Forum
12-16 January 2026, Orlando, FL
AIAA SCITECH 2026 Forum

10.2514/6.2026-1708

LSTM-Based Trajectory Prediction for Human-Drone Interaction in Structured Environments

Henry Hellmann*, Sooyung Byeon†, and Inseok Hwang‡
Purdue University, West Lafayette, IN, 47907

Anticipating human movement is essential for robotic motion planning in structured environments such as corridors, hospitals, and factory floors. This paper presents a lightweight, real-time trajectory prediction pipeline for human-aware drone-navigation, using head-orientation motion as the primary cue for intent inference. The approach combines a branched Long Short-Term Memory (LSTM) network with a discrete intent classifier that identifies the most likely motion mode (e.g., left, right, or straight at a hallway junction), subsequently predicting the corresponding continuous trajectory. While prior work has addressed human motion prediction using heavy multimodal predictors, the proposed model is designed for on-board deployability and low-latency inference. In addition, few studies have explored structured environments where movement is constrained to discrete paths. We collect a synchronized dataset of head-motion trajectories using an inertial measurement unit (IMU) and motion-capture system, and train the LSTM to predict future trajectories of humans, with a normalized motion frame calibrated to the environment. To evaluate the model, we integrate the predictions in a structured testbed with finite path options and static obstacles to demonstrate how prediction accuracy affects human-drone coexistence. The results demonstrate that our lightweight model enables accurate trajectory forecasting and classification, enabling early avoidance planning. The work highlights the viability of simple, interpretable sequence models for real-time human-aware navigation in constrained spaces.

I. Introduction

Over the decades, mobile robots have become increasingly common in structured indoor environments such as hospitals, warehouses, and commercial buildings. These spaces are shared with human pedestrians and workers, creating scenarios where autonomous systems must move safely and efficiently alongside humans [1]. A core requirement for such shared autonomy is a path planning architecture that can anticipate human movement and adapt behavior accordingly. As deployments grow in scale and complexity, the need for reliable, real-time human path prediction in structured environments has become increasingly significant.

Human trajectory prediction has been extensively studied, with recent work dominated by data-driven deep learning approaches that infer future motion from observed human movement [1]. These models learn motion patterns directly from data rather than relying on explicitly defined human dynamics, and they form the basis of many modern predictors such as single-trajectory forecasting [2], maximum-likelihood path estimation [3], and probabilistic motion envelopes [4, 5]. In socially compliant navigation, related work incorporates human-motion prediction as a core component, for example, through confidence-aware Bayesian models that prioritize human movement during planning [6] and utilize the FaSTrack motion planning framework to guarantee tracking performance [7]. Collectively, these approaches demonstrate strong capability in learning and forecasting human motion across a wide range of scenarios and have become standard components in modern prediction pipelines.

Despite this progress, most existing prediction models are evaluated on offline human datasets or simulation environments and not integrated into closed-loop robotic systems that must continuously respond to human movement. As noted in [8], most prediction methods rely on passive observation data and lack closed-loop validation with a robot that navigates in a shared space. Similarly, there are few frameworks that allow robots to respond to predicted human motion in interactive real-world scenarios [9]. Moreover, prior work has focused primarily on unstructured or open environments where human motion unfolds along smooth, continuous trajectories. In contrast, structured indoor environments such as hallways, warehouse aisles, and T-junctions require discrete navigational choices (e.g., turning

*Graduate Student, School of Aeronautics and Astronautics, hhellman@purdue.edu.
†Postdoctoral Fellow, School of Aeronautics and Astronautics, sbyeon@purdue.edu.
‡Paul Stanley Professor, School of Aeronautics and Astronautics, ihwang@purdue.edu, AIAA Associate Fellow.

left, right, or moving forward), making the prediction task fundamentally different. Errors in anticipating such discrete decisions can lead to unsafe robot behavior or inefficient planning. These limitations underscore the need for prediction frameworks tailored to structured environments with clear geometric and semantic constraints.

A novel approach is proposed to improve robotic social navigation by enhancing human trajectory prediction in structured environments. Unlike traditional methods that model human motion as a single continuous trajectory, this work recognizes that in many indoor settings, such as hallways, warehouses, or factory floors, human movement often follows one of several discrete options dictated by the layout. The proposed model leverages this structure by predicting a finite set of plausible future trajectories, effectively reformulating the task as a multiple-choice classification problem. For example, at a hallway junction, the model selects the human's likely path from options such as turning left, continuing straight, or turning right. This choice-based formulation enables earlier and more accurate anticipation of human intent, allowing the robot to adapt its path planning strategy proactively while providing a more interpretable representation of human decision-making in structured spaces.

To autonomously predict both human trajectory and discrete path choice, we use a Long-Short Term Memory (LSTM)-based neural network that generates real-time predictions from past motion data. This allows the robot to anticipate human movement in a shared, structured environment and adjust its navigation accordingly. To evaluate the effectiveness of the proposed model, we conduct experiments in a human–drone shared space and measure performance using accuracy metrics including position error, heading angle error, and path choice error. Finally, as a proof-of-concept for prediction-informed planning, we integrate the predicted human motion into a Signal Temporal Logic (STL)-based framework [10], which enables the drone to compute safe, constraint-satisfying trajectories aligned with the human's anticipated movement.

This paper makes three key contributions to the field of human-aware robotic motion planning. First, we identify the unique challenges posed by structured indoor environments, where human movement is governed by discrete navigational decisions rather than smooth, continuous trajectories, and we frame human prediction in a way that reflects these environment-driven constraints. Second, we develop a deterministic prediction model that predicts both the human's intended direction and the corresponding future trajectory. Finally, we implement this approach using an LSTM-based real-time predictor and validate the system in a shared human–drone environment, demonstrating improved performance in forecasting both trajectory and discrete path choice.

II. Problem Definition

In structured environments such as hallways, building corridors, or warehouse floors, humans make discrete navigation choices at specific decision points. Unlike in open or unstructured environments such as the outdoors, structured environments impose constraints on motion, limiting feasible trajectories to a small set of discrete options. The environmental constraints lead to a multiple-choice prediction setting for the robot, where the goal is to identify the most probable future paths a human might follow. The problem we address in this work is predicting which of these discrete path options a human will take in real-time. Predicting which path a human is likely to take is critical for socially aware robotic navigation and safe human-robot interaction.

A. Formal Problem Statement

As shown in Figure 1, the human in the structured environment scenario can follow one of several possible trajectories, e.g., turn left, right, or go straight. To formally describe this prediction problem, we first represent the human's past motion as follows. Given a sequence of observed human positions and orientations up to time t:

$$X_{0:t} = \{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_t\}, \tag{1}$$

we define the state vector at time i as:

$$\mathbf{x}_i = \begin{bmatrix} x_i & y_i & v_{x,i} & v_{y,i} & \psi_i & \dot{\psi}_i \end{bmatrix}^T \in \mathbb{R}^6, \tag{2}$$

where x_i, y_i denote position, $v_{x,i}, v_{y,i}$ represent velocity in the x and y directions, and ψ_i corresponds to the yaw or heading angle of the human head. $\dot{\psi}_i$ is the yaw rate. The goal of the prediction model is to generate deterministic future trajectories for each of the K feasible motion branches in the environment:

$$\hat{X}^1_{t+1:t+T},\ \hat{X}^2_{t+1:t+T},\ \ldots,\ \hat{X}^K_{t+1:t+T} \tag{3}$$

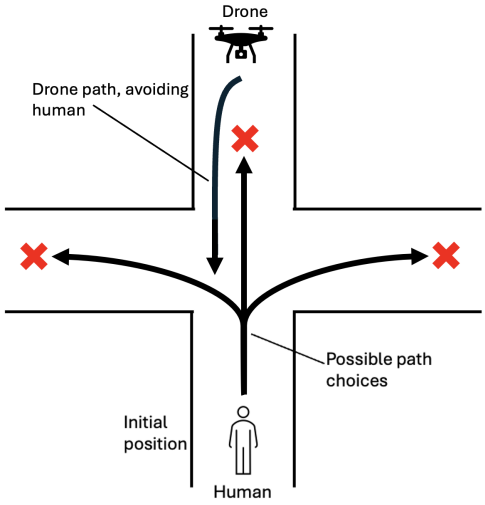


Fig. 1 Human motion branches in a structured indoor environment. The three arrows represent the distinct feasible paths—left, straight, and right—defined by the corridor geometry for human trajectory. The prediction model produces a deterministic trajectory prediction and a corresponding branch classification indicating which path the human is expected to follow, which informs the trajectory of the drone, avoiding collision with the human.

where each candidate trajectory $\hat{X}^k_{t+1:t+T}$ describes the predicted human motion over the next T time steps assuming the human continues along branch k. In the structured corridor environment considered in this work, $K = 3$ corresponding to *left*, *straight*, and *right* branches, see Figure 1.

The structured environment is encoded implicitly through the motion patterns present in the training data, rather than through explicit semantic maps or environment labels. The model therefore learns turning points, corridor alignments, and typical transition behaviors directly from observed trajectories. These predicted trajectories and the associated intent classification are then passed to the downstream STL-based planner, which generates a safe, constraint-satisfying drone trajectory that proactively adapts to anticipated human motion. The planning formulation is described in the next section.

B. Challenges

Human trajectory prediction in structured environments presents several unique challenges that must be addressed for reliable real-time planning and safe human–robot interaction. These challenges arise from the inherent uncertainty of human behavior, the limited availability of intent information, and the geometric constraints imposed by indoor spaces. We summarize the major challenges as follows.

Multimodality of human decisions at intersections. Human behavior in structured environments is inherently multimodal. A person approaching a junction may plausibly turn left, right, or continue forward given the same observed motion history. Capturing this uncertainty is challenging, as the model must represent multiple distinct future outcomes without collapsing them into an averaged, unrealistic prediction.

Lack of labeled intent data. Human intent is typically not explicitly annotated in trajectory datasets. As a result, the model must infer future decisions from low-level motion features such as position, velocity, and orientation. This weak form of supervision complicates the learning process and increases the likelihood of misclassification, particularly in ambiguous or transitional states.

Environmental and spatial constraints: Structured indoor environments impose strong geometric constraints due to walls, furniture, and fixed obstacles. These constraints restrict the feasible motion space and introduce sharp discontinuities in the set of allowable trajectories. Even small prediction errors can therefore lead to planning failures or unsafe robot behavior if not properly accounted for.

Real-time integration with planning. Prediction must operate at sufficiently high frequency to support real-time robot control. This requires models that are computationally efficient yet accurate, producing stable predictions under latency constraints while remaining robust to dynamic changes in the environment.

III. Approaches

Figure 2 illustrates the overall prediction and planning framework proposed in this work. The system operates as a sequential pipeline consisting of (i) human trajectory prediction, (ii) deterministic path selection, (iii) STL-based path planning, and (iv) real-time execution. At each cycle, the robot observes recent human motion, predicts a finite set of future trajectories, plans a safe path under STL constraints, and executes the resulting motion while continuously monitoring for deviations.

A. Human Trajectory Prediction

To realize the multi-hypothesis prediction problem in (3), we employ an LSTM-based recurrent neural network that maps a finite history of human states to future trajectory and path-choice predictions. Let L denote the observation window length and let T be the prediction horizon defined in (3). At time t, the model receives the past L states

$$X_{t-L+1:t} = \{\mathbf{x}_{t-L+1}, \ldots, \mathbf{x}_t\}, \tag{4}$$

where each state $\mathbf{x}_i$ is given in (2) as $\mathbf{x}_i = [x_i, y_i, v_{x,i}, v_{y,i}, \psi_i, \dot{\psi}_i]^\top$. The LSTM processes this sequence to capture temporal dependencies and longer-term motion intent.

The network has two output heads. The first head performs trajectory regression and predicts the future state sequence over the horizon T,

$$\hat{\mathbf{x}}_{t+1}, \ldots, \hat{\mathbf{x}}_{t+T}, \tag{5}$$

where each $\hat{\mathbf{x}}_{t+j}$ has the same structure as $\mathbf{x}_i$ and includes the predicted position, velocity, heading, and yaw rate. The second head performs path classification and outputs a categorical distribution over the K discrete path options (e.g., left, straight, right) that correspond to the candidate trajectories in (3).

In practice, position and velocity features are standardized using dataset statistics, and the heading angle ψ_i is encoded using $\sin\psi_i$ and $\cos\psi_i$ to avoid discontinuities at $\pm\pi$. The model uses a two-layer LSTM architecture with a hidden dimension of 128, implemented in `PyTorch`. Training is performed with a composite loss that combines Mean Squared Error (MSE) for the trajectory head and cross-entropy loss for the classification head, with a weighting factor chosen to balance the two objectives.

The model is trained end-to-end using the MSE for trajectory regression together with a cross-entropy loss for path classification, weighted as described in the Appendix. Optimization is performed using the Adam optimizer with a learning rate of 10^{-4}, and training is conducted for 100 epochs with gradient clipping to ensure stability. Further architectural details, the explicit LSTM update equations, and training hyper parameters are provided in the Appendix.

B. Human Motion Dataset

The human trajectory prediction model was trained using real-world motion data collected in a controlled laboratory environment using a Qualisys motion capture system, which has been shown to provide high-accuracy human motion measurements[11]. To ensure diversity in motion patterns, participants executed multiple path types including left-turn, right-turn, and straight trajectories. For each path type, three motion profiles were recorded: straight, curved, and zig-zag motion. Each motion profile was performed under two conditions, regular walking and stop-and-go behavior. This leads to a broad distribution of velocity and heading-rate patterns.

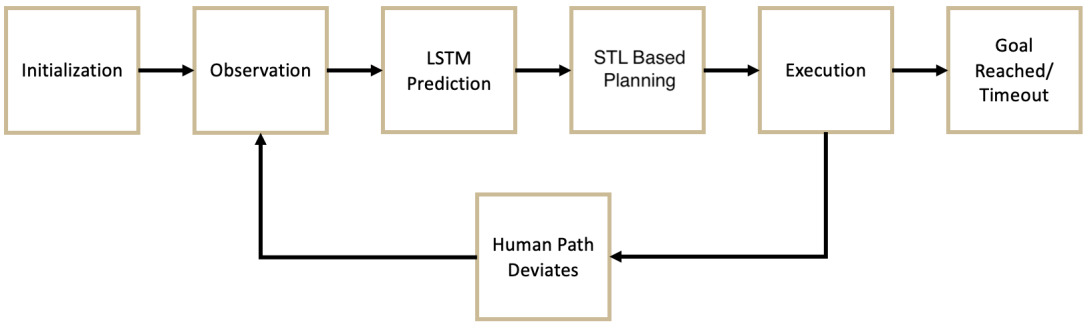


Fig. 2 Architecture of the overall prediction and planning framework.

Each trajectory was recorded for approximately 75 seconds at 50 Hz. Across all trials, the total dataset duration amounted to 76.25 minutes. In the experiment configuration, participants walked from a designated start point to a midpoint, then transitioned to one of three terminal branches (left, right, or straight). This structure enables the model to learn both continuous motion transitions and discrete navigational decisions. The dataset was split 80% for training and 20% for validation.

C. Interactive Drone Path Planning

To ensure safe navigation in the presence of a moving human whose future trajectory is uncertain, we employ an STL–based path planning framework (`stlpy` library is used [10]). STL provides a formal language for expressing time-dependent constraints over continuous signals [12]. An STL formula is constructed from *predicates*, which are Boolean conditions on the system state. A typical predicate takes the form $h(d_t) \leq 0$, where d_t is the decision variable and $h(\cdot)$ encodes a geometric or safety constraint such as remaining inside a corridor or outside a risk zone. These predicates are combined using logical operators such as conjunction ($\wedge$) and temporal operators including "eventually" ($\Diamond$) and "always" ($\Box$). For example, $\Box_{[a,b]}\varphi$ requires that the predicate φ hold for all $t \in [a, b]$, while $\Diamond_{[a,b]}\varphi$ requires that φ be satisfied at least once within that interval. Evaluating an STL formula over a candidate trajectory determines whether the trajectory satisfies the encoded navigation and safety requirements.

STL Specification for Human–Aware Navigation

Let $d_t = [x_{d,t}, y_{d,t}, \psi_{d,t}]^\top$ denote the discrete-time drone state (2D position and yaw angle) governed by the planar kinematic model,

$$x_{d,t+1} = x_{d,t} + \Delta t\, v_{d,t} \cos\psi_{d,t}, \tag{6}$$

$$y_{d,t+1} = y_{d,t} + \Delta t\, v_{d,t} \sin\psi_{d,t}, \tag{7}$$

$$\psi_{d,t+1} = \psi_{d,t} + \Delta t\, \omega_{d,t}, \tag{8}$$

where $v_{d,t}$ and $\omega_{d,t}$ denote linear and angular velocity commands. This discrete dynamics model is used within the STL planner to generate feasible drone motions.

Given a prediction horizon T, we divide the timeline into three phases that reflect the structure of the navigation task: (i) an *approaching* phase $[0, t_1]$ during which the drone moves toward an intermediate waypoint, (ii) a *waiting* phase $[t_1, t_2]$ where the drone must hold a safe corridor while the human's intent becomes clearer, and (iii) a *proceeding* phase $[t_2, T]$ during which the drone must avoid the predicted human-occupied region and reach the goal. t_1 and t_2 are estimated from the predicted human trajectory and can be updated during the operation. Using these intervals, the STL task specification is:

$$\Phi(d) = \Diamond_{[0,t_1]} W(d) \;\wedge\; \Box_{[t_1,t_2]} W(d) \;\wedge\; \Box_{[t_2,T]} \neg RZ(d) \;\wedge\; \Diamond_{[t_2,T]} RL_{\text{goal}}(d) \;\wedge\; \Phi_{\text{safe}}. \tag{9}$$

Here, $d \in \{\text{left, right, straight}\}$ is the discrete path choice. $W(d)$ is a predicate requiring the drone to remain within a designated waypoint corridor. The operator $\Diamond_{[0,t_1]}$ ensures the drone reaches this corridor during the approaching phase, and $\Box_{[t_1,t_2]}$ enforces that it remains within the corridor throughout the waiting phase. The predicate $RZ(d)$ denotes the predicted human-occupied region derived from the LSTM trajectory prediction; enforcing $\Box_{[t_2,T]}\neg RZ(d)$ ensures human-avoidance during the proceeding phase. The negation operator $\neg$ in STL reverses any predicate's truth value. The predicate $RL_{\text{goal}}(d)$ specifies the goal region to be reached before the horizon ends. Finally, Φ_{safe} captures general collision-avoidance requirements and is expressed compactly as $\Phi_{\text{safe}} = \neg\text{Collide}(d)$, where $\text{Collide}(d)$ is a predicate indicating whether the drone enters any static obstacle region.

IV. Experiment

This section describes the experimental setup used to evaluate the proposed human trajectory prediction model and its integration with STL-based drone navigation in a structured indoor environment.

A. Environment Setup

Experiments were conducted in a 5.5×3.0 meter motion-capture laboratory instrumented with a Qualisys camera system. A start point was marked on either side of the environment, along with three possible endpoints located opposite

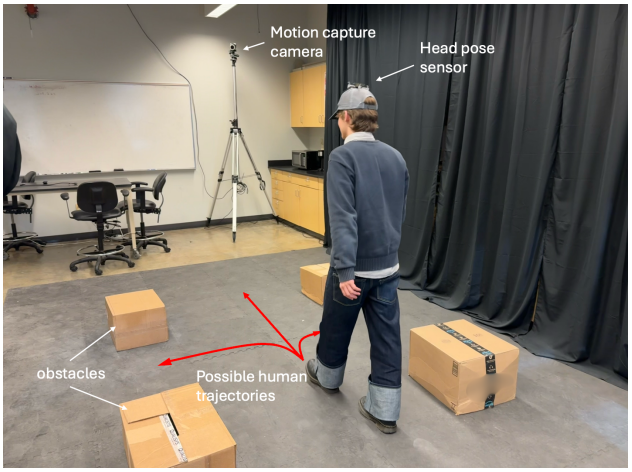


Fig. 3 Experiment environment

the start. A midpoint was placed between them, creating a multiple-choice decision scenario in which the human must choose one of three paths (left, straight, right) after reaching the midpoint. Cardboard boxes were placed as static obstacles to evaluate obstacle-aware drone path planning, shown in Figure 3.

B. Experimental Procedure

In each trial, the human begins at a designated start point, walks toward the midpoint, briefly orients their head toward one of the three endpoints to signal intent, and proceeds to the chosen endpoint. This turning event provides the LSTM model with early cues for discrete path prediction. Throughout the trial, position and orientation data from the human head are streamed to the prediction module, which outputs future trajectory estimates used by the STL planner to update the drone's motion.

C. Equipment

Human motion was captured using an active Qualisys motion-capture beacon combined with onboard inertial measurement unit (IMU) data from the drone-mounted sensor package. The Qualisys system tracks 3D position and orientation using infrared markers, while the Python `qualisys_python_sdk` [13] provides real-time state estimates through an extended Kalman filter. Drone motion is executed using a Crazyflie 2.1 micro-UAV, controlled via a custom Python interface. Static obstacles were implemented using lightweight cardboard boxes. The LSTM neural network and STL planning framework were implemented in `PyTorch`.

D. Data Collection and Processing

Motion data were recorded at 50 Hz and included planar position (x, y), planar velocity (v_x, v_y), and head orientation (roll, pitch, yaw), using the drone-mounted sensor. Because the motion-capture environment is globally calibrated, no coordinate transforms were required. A key preprocessing step involved removing backward walking segments, which occur when the human returned to the start position after each trial; these motions corrupted the training distribution. Outliers from marker swaps or sensor dropouts were also filtered. Position and velocity features were normalized, and head orientations were converted from degrees to radians. Each dataset was then segmented into fixed-length observation windows and corresponding prediction horizons for LSTM training.

E. Performance Metrics

To evaluate prediction quality, we report:

- **Trajectory accuracy:** position error (m), velocity error (m/s), and heading angle error (deg);
- **Discrete path accuracy:** classification error (%) and prediction confidence across the three choices.

These metrics quantify the model's ability to predict both continuous human motion and discrete navigation intent in a structured environment.

V. Results

This section presents the performance of the proposed LSTM-based human trajectory prediction model and its discrete path classification module, evaluated on held-out experimental data collected in the structured indoor environment described earlier.

A. Qualitative Prediction Examples

Before reporting quantitative results, we illustrate representative predictions produced by the model. Figure 4 shows examples of trajectory forecasting and discrete branch prediction for left and straight maneuvers. The left column depicts predicted future positions overlaid with ground truth trajectories, while the right column shows the evolution of branch predictions across all prediction sequences. Labels 0, 1, and 2 correspond to left, right, and straight paths, respectively.

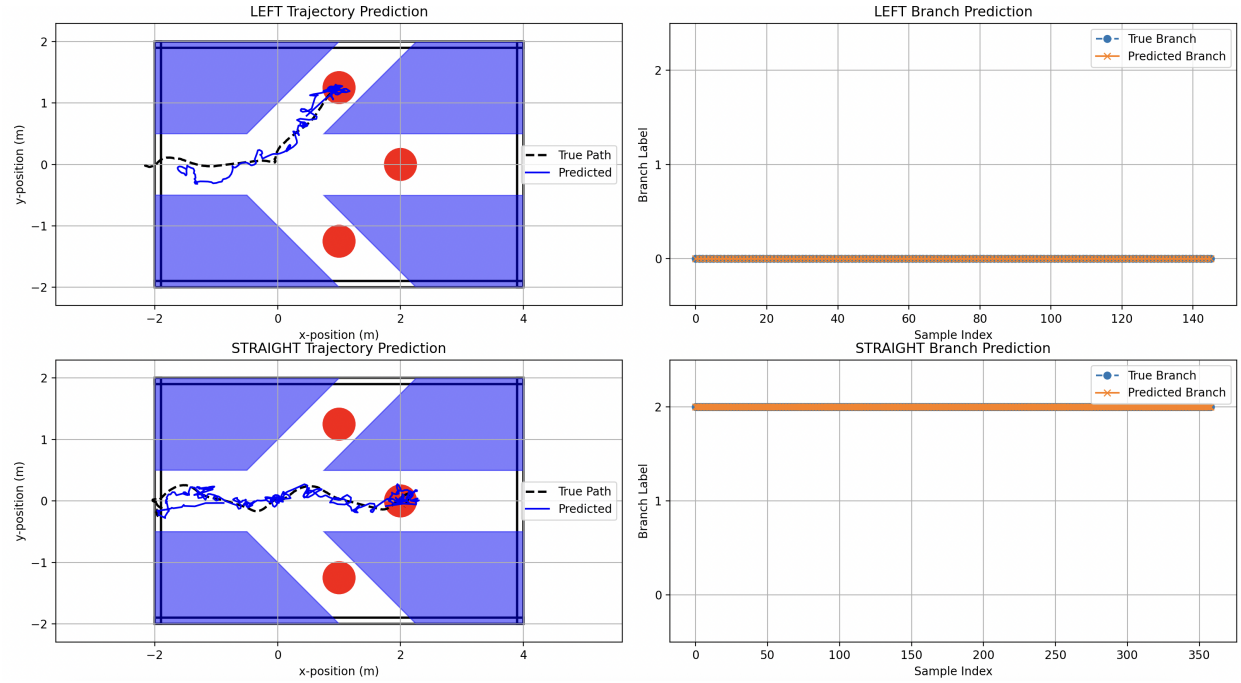


Fig. 4 Representative predictions of the LSTM model. Left: predicted trajectories human versus ground truth for left and straight maneuvers, where the human starts on the left side of the environment and walks toward one of three possible endpoints on the right (shown as the three red circles). Right: discrete human path classification (0: left, 1: right, 2: straight).

B. Trajectory Prediction Performance

To quantify the trajectory prediction performance, the validation dataset consists of 41 sequences of human motion collected in a single corridor environment, including 13 left-turn, 15 right-turn, and 13 straight-through trajectories. Each sequence contains 100 observed frames and 100 future frames, sampled at 50 Hz, with input features (x, y, v_x, v_y, ψ, $\dot{\psi}$). For evaluation, 20% of sequences were randomly selected as a validation set using a standard train-test split function with shuffling, ensuring no overlap between training and validation sequences. Tables 1–3 report aggregate quantitative metrics across all branches, including RMSE, mean Euclidean error, and error growth over increasing prediction horizons. See Tables 7, 8, and 9in Appendix for trajectory prediction error over each validation test for left, right, and straight segments, respectively. Across all trials, the average position error remains below 0.2 m, indicating stable performance despite the model's compact architecture and modest dataset.

Figure 5 provides a complementary visual analysis of prediction performance. The plot illustrates per-branch error distributions, horizon-dependent error growth, and horizon-dependent error distribution. The consistency across the three branches reflects the structured nature of the environment and the model's ability to generalize across different trajectory types. Error decreases at later time steps because branch-choice ambiguity collapses once the human initiates

a turn or proceeds straight, causing the remaining trajectory to align with a fixed corridor geometry, reducing the prediction uncertainty. Unlike standard unconstrained pedestrian settings where error typically increases monotonically, our environment causes ambiguity to resolve early, resulting in decreasing error after ~ 25 time steps.

C. Path Classification Performance

Discrete intent classification complements continuous trajectory estimation by explicitly identifying the navigational branch, enabling downstream planners to reason over high-level intent as well as precise motion. Discrete path classification accuracy was evaluated over the same validation sequences. Table 4 summarizes average accuracy for each branch. Overall classification accuracy exceeds 98%, with perfect accuracy for left-turn segments. See Table 10 in Appendix for classification accuracy over each validation trial.

Notably, the classification accuracy for trial "straight_013" (Table 10) is 76.90%, which is an outlier compared to the near-perfect performance in other trials. This reduction is attributed to significant heading variability in this specific sequence; the subject momentarily oriented their body toward the left branch near the decision point before correcting to a straight path. This transient heading deviation created ambiguity in the LSTM inputs, resulting in temporary misclassification.

D. Prediction Performance Comparison

For context, Social-LSTM [14] reports displacement errors in the range of 0.3–0.5 m on the ETH/UCY pedestrian datasets, while state-of-the-art multimodal models such as Trajectron++ [15] achieve ADE (Average Displacement Error) values around 0.2–0.3 m. Although these datasets differ fundamentally from our structured indoor setting and sensing modality, the comparison provides a useful reference point. Our model achieves sub-0.2 m average position error with a lightweight architecture and minimal sensing, demonstrating competitive performance despite the scope differences. Lower absolute errors are expected in our environment due to its constrained corridor geometry and

Table 1 Aggregate Prediction Error Statistics Across All Validation Datasets

Metric	Value (m)
RMSE Mean	0.1711
RMSE Median	0.1627
RMSE Std	0.0309
Mean Eucl. Mean	0.1858
Mean Eucl. Median	0.1782
Mean Eucl. Std	0.0268

Table 2 Error Growth Over Prediction Horizon

Timestep	RMSE (m)	Mean Eucl. (m)	95th Percentile (m)
10	0.2794	0.3374	0.7275
25	0.1758	0.2056	0.4811
50	0.1167	0.1393	0.3110
75	0.1075	0.1300	0.2824
100	0.1104	0.1308	0.2850

Table 3 Per-Branch Prediction Error Summary

Branch	Files	RMSE (m)	Mean Eucl. (m)
Left	13	0.1871 ± 0.0377	0.1948 ± 0.0334
Right	15	0.1641 ± 0.0234	0.1816 ± 0.0211
Straight	13	0.1631 ± 0.0241	0.1817 ± 0.0228

Table 4 Summary of Classification Accuracy Across Validation Branches

Branch	Num. Files	Total Sequences	Avg. Accuracy (%)
Left	13	4,005	100.00
Right	15	7,414	99.65
Straight	13	5,697	95.40

shorter prediction horizons compared to open-space pedestrian datasets. Because no existing baseline directly addresses discrete left/right/straight intent classification using head-motion cues in indoor corridors, path classification is evaluated internally via accuracy and confusion metrics, which adequately characterize performance for this application.

E. Interactive Path Planning Performance

To evaluate how the STL-based path planner responds to evolving human motion, we conducted closed-loop trials in which the drone continuously receives LSTM trajectory predictions and replans its motion accordingly. In all trials, the drone's task is to travel from the right side of the environment to a designated goal region on the left, while maintaining safe separation from the human. Figure 6 illustrates two representative cases showing how the planner adapts its trajectory when the human selects different branches in the structured environment.

In Case 1, the human commits to the left branch, which corresponds to the upward direction in the layout. As the LSTM model updates its future-motion prediction, the predicted human-occupied region expands into the upper corridor. The STL safety constraints prevent the drone from entering this region, so the planner keeps the drone inside its designated wait zone until the human clears the intersection. Once the intersection is cleared, the drone proceeds leftward toward its goal.

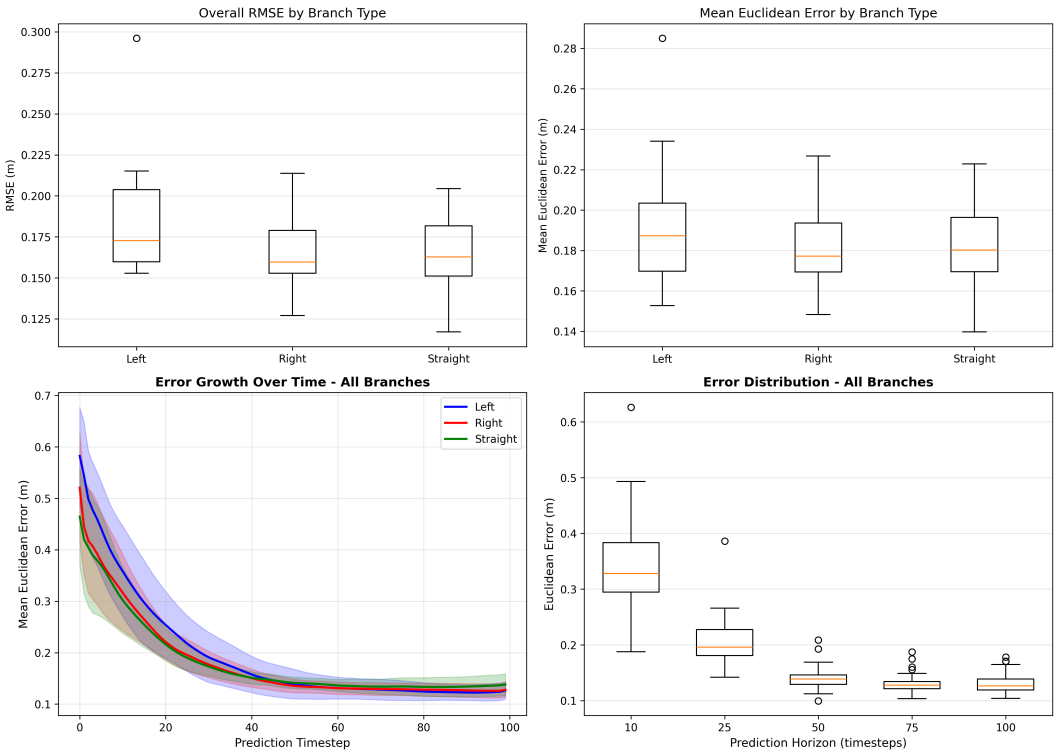
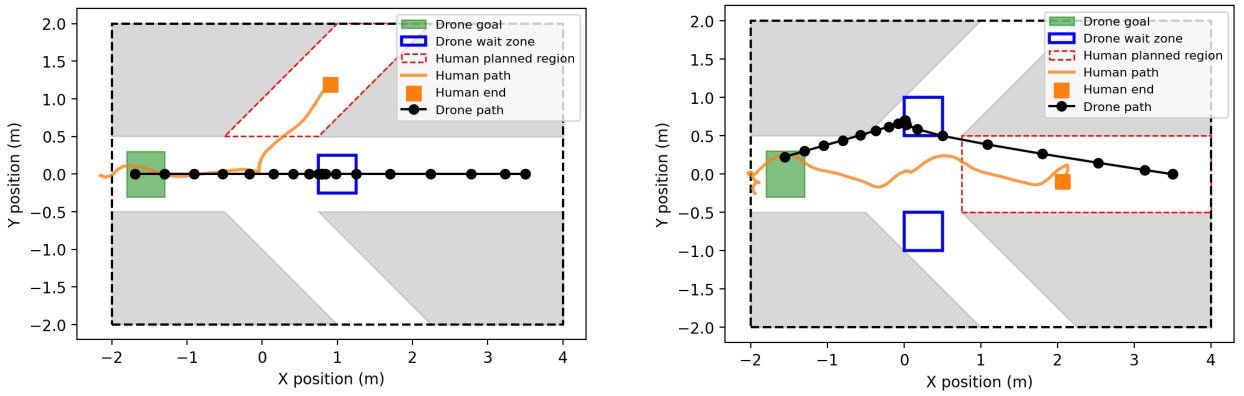


Fig. 5 Model performance by branch. Top left: Average RMSE for each branch type across all validation trials. Top right: Average Euclidean error for each branch type across all validation trials. Bottom left: Error growth over increasing prediction horizons for each branch, with error bounds. Bottom right: Error distribution across prediction horizons for all branches.



Case 1: Human takes the left branch. **Case 2: Human continues straight.**

Fig. 6 Interactive STL-based path planning in response to predicted human motion. The red dashed region denotes the predicted human-occupied zone generated from the LSTM model. The drone trajectory (black) adapts online to avoid the occupied region while respecting STL specification in (9)**.**

In Case 2, the human continues straight through the intersection. Because the predicted human-occupied region lies directly along the corridor that the drone would nominally traverse, the STL constraints force the drone to yield by shifting laterally out of the way rather than proceeding into the shared path. The planner therefore selects a side path that preserves safe separation while still making progress toward the left-side goal. This case demonstrates the system's ability to modify the nominal route when human motion overlaps the primary corridor, enabling smooth and conflict-free interaction without manual heuristics.

Across all interactive trials, the STL-based controller generated collision-free trajectories and consistently maintained separation from the predicted human motion region. These results highlight the advantage of combining structured trajectory prediction with formal temporal-logic planning for safe, adaptive navigation in shared indoor environments.

F. Limitations

The current model has several notable limitations. First, it was trained using data from a single human subject, which restricts its ability to generalize to individuals with different gait patterns or motion characteristics. Expanding the dataset to include a diverse set of participants would likely improve robustness. Second, the model was trained and tested in a single, fixed corridor layout, limiting its applicability to new environments. Training across multiple indoor geometries would help the model learn environment-invariant motion patterns. Third, although motion capture+IMU sensing provides high-precision data, it does not reflect the sensing constraints of real-world deployment. Incorporating onboard perception, such as vision-based pose estimation or wearable sensing, would improve practicality. Finally, the model outputs deterministic trajectories without an associated measure of uncertainty. This lack of probabilistic reasoning limits its ability to represent prediction confidence and can reduce planning performance in ambiguous scenarios.

VI. Conclusion and Future Work

This work presented a human trajectory prediction framework for structured indoor environments using an LSTM-based model that captures both continuous motion patterns and discrete navigational intent. When integrated with an STL-based path planning algorithm, the approach enables a robot to navigate safely and proactively around humans by anticipating likely future motions. The combined system provides a computationally efficient method for reducing unsafe or uncomfortable human–robot interactions in structured shared spaces.

Future work will focus on relaxing the sensing assumptions and improving the robustness of prediction and planning. Vision-based perception could be incorporated to replace the motion-capture infrastructure and enable deployment in more realistic environments. In addition, future models may interpret richer human behaviors such as gestures or head cues to improve early intent recognition. Finally, integrating uncertainty quantification or risk-aware decision making,

for example through Bayesian or conformal prediction techniques, may provide calibrated confidence estimates that support safer and more reliable robot behavior.

Appendix

A. Feature Normalization

To stabilize training, position and velocity features are standardized:

$$\tilde{x}_t = \frac{x_t - \mu_x}{\sigma_x}, \quad \tilde{v}_{x,t} = \frac{v_{x,t} - \mu_{v_x}}{\sigma_{v_x}}$$

where μ and σ denote mean and standard deviation computed over the training set. Angular features $\sin(\psi_t)$ and $\cos(\psi_t)$ are used directly without normalization as they are naturally bounded in $[-1, 1]$.

B. LSTM Encoder Architecture

The model employs a multi-layer Long Short-Term Memory (LSTM) network to encode temporal dependencies in the observed trajectory. For each input sequence, the LSTM computes hidden states:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \\
\mathbf{f}_t &= \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{b}_f) \\
\mathbf{o}_t &= \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \\
\tilde{\mathbf{c}}_t &= \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned}$$

where $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ are the input, forget, and output gates, respectively; $\mathbf{c}_t$ is the cell state; $\mathbf{h}_t \in \mathbb{R}^{d_h}$ is the hidden state; $\sigma(\cdot)$ is the sigmoid function; $\odot$ denotes element-wise multiplication; and $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{b}$ are learnable weight matrices and bias vectors, respectively. The model uses $L = 2$ stacked LSTM layers with hidden dimension $d_h = 128$. For notational clarity, let $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_T]$ denote the sequence of hidden states from the final LSTM layer. A summary of architectural parameters is provided in Table 5.

Table 5 Model Architecture Summary

Component	Configuration
Input Dimensions	6: (x, y, vx, vy, ψ, $\dot{\psi}$)
LSTM Layers	2
Hidden Dimensions	128
Output Dimension	6: (predicted state)
Classification outputs	3: (left, straight, right)
Sequence length (T)	100 timesteps
Prediction horizon (H)	100 timesteps
Total parameters	230,000

C. Dual-Task Output Heads

Path Classification Head. The path choice is predicted using the final hidden state $\mathbf{h}_T$, which encodes information from the entire input sequence. A fully connected layer maps this to logits for each class:

$$\mathbf{z}_c = \mathbf{W}_c \mathbf{h}_T + \mathbf{b}_c$$

where $\mathbf{z}_c \in \mathbb{R}^3$ contains logits for the three path choices. The predicted class probabilities are obtained via softmax:

$$p(c = k|\mathbf{X}) = \frac{\exp(z_{c,k})}{\sum_{j=1}^{3} \exp(z_{c,j})}.$$

The predicted class is:

$$\hat{c} = \arg\max_k p(c = k|\mathbf{X}).$$

Trajectory Prediction Head. Future trajectory is predicted using the last H hidden states from the LSTM output sequence. This approach leverages the temporal structure learned by the LSTM. For each future timestep $t' \in [T+1, T+H]$, we compute:

$$\hat{\mathbf{y}}_{t'} = \mathbf{W}_y \mathbf{h}_{T+t'-T} + \mathbf{b}_y$$

where $\mathbf{W}_y \in \mathbb{R}^{6\times d_h}$ and $\mathbf{b}_y \in \mathbb{R}^6$ are learnable parameters. Note that we reuse the LSTM hidden states from the observation window for prediction, effectively treating the last H time steps as representative of future motion patterns. The complete predicted trajectory is:

$$\hat{\mathbf{Y}} = [\hat{\mathbf{y}}_{T+1}, \hat{\mathbf{y}}_{T+2}, ..., \hat{\mathbf{y}}_{T+H}].$$

D. Loss Function

The model is trained using a composite loss function that balances trajectory prediction accuracy and path classification:

$$\mathcal{L} = \mathcal{L}_{\text{traj}} + \lambda \mathcal{L}_{\text{class}}.$$

This multi-task learning approach allows the model to simultaneously learn spatial-temporal dynamics through trajectory prediction while capturing high-level behavioral patterns through path classification. The composite structure enables the shared LSTM encoder to learn representations that are beneficial for both tasks, improving generalization.

Trajectory Loss ($\mathcal{L}_{\text{traj}}$) quantifies the accuracy of future position predictions and is computed as the mean squared error (MSE) between predicted and true future states:

$$\mathcal{L}_{\text{traj}} = \frac{1}{H} \sum_{t'=T+1}^{T+H} \|\hat{\mathbf{y}}_{t'} - \mathbf{y}_{t'}\|_2^2$$

where H is the prediction horizon, $\hat{\mathbf{y}}_{t'}$ represents the predicted state at time t', and $\mathbf{y}_{t'}$ is the ground truth state. The MSE loss penalizes deviations in predicted coordinates quadratically, making the model sensitive to large prediction errors. The L_2 norm captures Euclidean distance in the state space, providing an intuitive measure of positional accuracy. Averaging over the prediction horizon ensures that the loss magnitude remains consistent regardless of H.

Classification Loss ($\mathcal{L}_{\text{class}}$) measures the model's ability to correctly identify the overall trajectory category and uses cross-entropy:

$$\mathcal{L}_{\text{class}} = -\sum_{k=1}^{3} \mathbb{1}[c = k] \log p(c = k|\mathbf{X})$$

where $\mathbb{1}[\cdot]$ is the indicator function that equals 1 when the condition is true and 0 otherwise, c is the true class label, and $p(c = k|\mathbf{X})$ is the predicted probability for class k given input sequence $\mathbf{X}$. Cross-entropy loss is particularly effective for classification tasks as it strongly penalizes confident but incorrect predictions while providing gentle gradients for correct predictions, facilitating stable training. This loss encourages the model to learn discriminative features that distinguish between the three path categories.

The weighting parameter $\lambda = 0.5$ balances the two objectives, determined empirically to prevent either task from dominating during training. This value ensures that both trajectory accuracy and classification performance contribute equally to the optimization process, allowing the model to develop a balanced understanding of both fine-grained positional dynamics and coarse-grained behavioral patterns. A summary of architectural parameters is provided in Table 6.

Table 6 Loss Function Notation Summary

Symbol	Definition	Units
$\mathbf{X}$	Observed trajectory sequence	
$\mathbf{Y}$	True Future Trajectory	
$\hat{\mathbf{Y}}$	Predicted future trajectory	
c	True path choice	0, 1, 2
$\hat{c}$	Predicted path choice	0, 1, 2
T	Observation window length	timesteps
H	Prediction Horizon	timesteps
$\mathcal{L}_{\text{traj}}$	Trajectory MSE loss	m^2
$\mathcal{L}_{\text{class}}$	Classification cross-entropy	nats
λ	Loss balancing weight	

E. Training Procedure

Label Assignment Strategy: To address the challenge of trial-level labels (e.g., "this trial turns right") not matching instantaneous motion, we employ smart labeling:

$$c_i = \begin{cases} c_{\text{trial}} & \text{if } \Delta\psi_i > \theta_{\text{turn}} \text{ and } c_{\text{trial}} \in \{\text{left, right}\} \\ \text{straight} & \text{otherwise} \end{cases}$$

where $\Delta\psi_i = \psi_{i+H} - \psi_i$ is the heading change of the human head over the prediction window, and $\theta_{\text{turn}} = 0.26$ rad ($\approx 15°$) is the turn detection threshold. This ensures labels reflect actual motion in the prediction window.

The training loop is as follows. For each epoch, iterate through training batches, then forward pass to compute $\hat{\mathbf{Y}}$ and $p(c|\mathbf{X})$. Compute composite loss $\mathcal{L}$ backward pass with gradient clipping and update parameters using Adam optimizer. Validate on a held-out set (80/20 train/validation split), then save the model checkpoint if validation loss improves. Training terminates after 100 epochs or when validation loss plateaus. The model with lowest validation loss is selected for evaluation.

F. Implementation Details

The model is implemented in PyTorch and trained on CUDA backend. Key implementation choices include numerical stability with gradient clipping that prevents exploding gradients common in RNN training. Batch normalization is not used as LSTM internal gating provides sufficient regularization. Dropout between LSTM layers is avoided as it degraded performance in preliminary experiments. Efficient inference is achieved for real-time deployment through predictions that are computed at each timestep using a sliding window. At time t, we predict trajectories for $[t+1, t+T]$ using observations $[t-L+1, t]$, enabling continuous trajectory updates as new measurements arrive.

Table 7 Prediction Error Metrics for Left-Turn Segments

File	Sequences	RMSE (m)	Mean Eucl. (m)
left_001	382	0.1583	0.1694
left_002	297	0.2112	0.2340
left_004	193	0.1588	0.1770
left_005	255	0.1728	0.1697
left_006	204	0.1914	0.1998
left_007	253	0.1529	0.1528
left_008	146	0.2961	0.2850
left_009	197	0.2152	0.2118
left_011	166	0.1598	0.1678
left_012	230	0.2037	0.2034
left_013	711	0.1859	0.1960
left_014	642	0.1613	0.1782
left_015	429	0.1647	0.1872

Table 8 Prediction Error Metrics for Right-Turn Segments

File	Sequences	RMSE (m)	Mean Eucl. (m)
right_000	424	0.1854	0.2004
right_001	439	0.1617	0.1719
right_002	483	0.1519	0.1763
right_003	416	0.1538	0.1751
right_004	600	0.1597	0.1776
right_005	569	0.1652	0.1772
right_006	240	0.2137	0.2268
right_009	535	0.1939	0.2117
right_012	484	0.1271	0.1484
right_013	139	0.1929	0.2078
right_014	311	0.1296	0.1528
right_015	165	0.1723	0.1868
right_016	548	0.1560	0.1794
right_018	516	0.1573	0.1669
right_019	469	0.1403	0.1648

Table 9 Prediction Error Metrics for Straight-Line Segments

File	Sequences	RMSE (m)	Mean Eucl. (m)
straight_001	660	0.1628	0.1803
straight_002	537	0.1584	0.1729
straight_003	399	0.1893	0.2022
straight_004	359	0.1627	0.1830
straight_005	317	0.1606	0.1777
straight_006	276	0.1511	0.1695
straight_007	322	0.1318	0.1491
straight_008	507	0.1170	0.1397
straight_009	398	0.1358	0.1633
straight_011	250	0.2044	0.2228
straight_012	305	0.1817	0.2109
straight_013	381	0.1890	0.1963
straight_014	286	0.1758	0.1938

Table 10 Detailed Branch Classification Accuracy Across All Validation Trials

Left Turn			Right Turn			Straight		
File	Seq.	Acc.(%)	File	Seq.	Acc.(%)	File	Seq.	Acc.(%)
left_001	382	100.00	right_000	424	100.00	straight_001	660	100.00
left_002	297	100.00	right_001	439	100.00	straight_002	537	99.07
left_004	193	100.00	right_002	483	100.00	straight_003	399	100.00
left_005	255	100.00	right_003	416	100.00	straight_004	359	100.00
left_006	204	100.00	right_004	600	100.00	straight_005	317	100.00
left_007	253	100.00	right_005	569	94.73	straight_006	276	100.00
left_008	146	100.00	right_006	240	100.00	straight_007	322	100.00
left_009	197	100.00	right_009	535	100.00	straight_008	507	100.00
left_011	166	100.00	right_012	484	100.00	straight_009	398	100.00
left_012	230	100.00	right_013	139	100.00	straight_011	250	100.00
left_013	711	100.00	right_014	311	100.00	straight_012	305	100.00
left_014	642	100.00	right_015	165	100.00	straight_013	381	76.90
left_015	429	100.00	right_016	548	100.00	straight_014	286	100.00
			right_018	516	100.00			
			right_019	469	100.00			

References

[1] Golchoubian, M., Ghafurian, M., Dautenhahn, K., and Azad, N. L., "Pedestrian Trajectory Prediction in Pedestrian-Vehicle Mixed Environments: A Systematic Review," *IEEE Transactions on Intelligent Transportation Systems*, Vol. 24, No. 11, 2023, p. 11544–11567. https://doi.org/10.1109/tits.2023.3291196, URL http://dx.doi.org/10.1109/TITS.2023.3291196.

[2] Wang, C., Wang, Y., Huang, Z., and Chen, Z., "Simple Baseline for Single Human Motion Forecasting," *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2021, pp. 2260–2265. https://doi.org/10.1109/ICCVW54120.2021.00255.

[3] Fu, Y., Yan, Q., Wang, L., Li, K., and Liao, R., "MoFlow: One-Step Flow Matching for Human Trajectory Forecasting via Implicit Maximum Likelihood Estimation based Distillation," *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025, pp. 17282–17293. https://doi.org/10.1109/CVPR52734.2025.01611.

[4] de Groot, O., Sridharan, A., Alonso-Mora, J., and Ferranti, L., "Probabilistic Motion Planning and Prediction via Partitioned Scenario Replay," *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 7546–7552. https://doi.org/10.1109/ICRA57147.2024.10611014.

[5] Salzmann, T., Pavone, M., and Ryll, M., "Motron: Multimodal Probabilistic Human Motion Forecasting," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 6447–6456. URL https://api.semanticscholar.org/CorpusID:247315463.

[6] Fridovich-Keil, D., Bajcsy, A., Fisac, J. F., Herbert, S. L., Wang, S., Dragan, A. D., and Tomlin, C. J., "Confidence-aware motion prediction for real-time collision avoidance1," *The International Journal of Robotics Research*, Vol. 39, No. 2-3, 2020, pp. 250–265. https://doi.org/10.1177/0278364919859436.

[7] Herbert, S. L., Chen, M., Han, S., Bansal, S., Fisac, J. F., and Tomlin, C. J., "FaSTrack: A modular framework for fast and guaranteed safe motion planning," *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, IEEE, 2017. https://doi.org/10.1109/cdc.2017.8263867, URL http://dx.doi.org/10.1109/CDC.2017.8263867.

[8] Rudenko, A., Palmieri, L., Herman, M., Kitani, K. M., Gavrila, D. M., and Arras, K. O., "Human motion trajectory prediction: a survey," *The International Journal of Robotics Research*, Vol. 39, No. 8, 2020, p. 895–935. https://doi.org/10.1177/0278364920917446, URL http://dx.doi.org/10.1177/0278364920917446.

[9] Finean, M. N., Petrović, L., Merkt, W., Marković, I., and Havoutis, I., "Motion planning in dynamic environments using context-aware human trajectory prediction," *Robotics and Autonomous Systems*, Vol. 166, 2023, p. 104450. https://doi.org/https://doi.org/10.1016/j.robot.2023.104450, URL https://www.sciencedirect.com/science/article/pii/S0921889023000891.

[10] Kurtz, V., and Lin, H., "Mixed-Integer Programming for Signal Temporal Logic with Fewer Binary Variables," *IEEE Control Systems Letters*, 2022.

[11] Patel, G., Mullerpatan, R., Agarwal, B., Shetty, T., Ojha, R., Shaikh-Mohammed, J., and Sujatha, S., "Validation of wearable inertial sensor-based gait analysis system for measurement of spatiotemporal parameters and lower extremity joint kinematics in sagittal plane," *Proceedings of the Institution of Mechanical Engineers, Part H: Journal of Engineering in Medicine*, Vol. 236, No. 5, 2022, pp. 686–696. https://doi.org/10.1177/09544119211072971, URL https://doi.org/10.1177/09544119211072971, pMID: 35001713.

[12] Rodionova, A., Lindemann, L., Morari, M., and Pappas, G. J., "Time-Robust Control for STL Specifications," *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 572–579. https://doi.org/10.1109/CDC45484.2021.9683477.

[13] Qualisys, "qualisys_python_sdk: Python SDK for Qualisys Motion Capture Systems," https://github.com/qualisys/qualisys_python_sdk, 2025. Accessed: 2025-05-21.

[14] Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and Savarese, S., "Social LSTM: Human Trajectory Prediction in Crowded Spaces," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[15] Salzmann, T., Ivanovic, B., Chakravarty, P., and Pavone, M., "Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data," *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII*, Springer-Verlag, Berlin, Heidelberg, 2020, p. 683–700. https://doi.org/10.1007/978-3-030-58523-5_40, URL https://doi.org/10.1007/978-3-030-58523-5_40.