Technical Report

End-to-end Goal Conditioned Social Navigation using Signal Temporal Logic

Jasmine Jerry Aloor[1],
Supervised by: Jay Patrikar[2] and Sebastian Scherer[2]

***Abstract*— With an anticipated increase in aerial traffic in the near future in the form of crewed air-taxis, aircraft, and autonomous aerial systems, achieving safe human-operated and independent vehicle coordination in shared airspace is crucial. Current studies have investigated pedestrian and ground vehicle trajectories in multiagent environments in great detail. However, these studies have not yet analyzed aerial trajectories. There is a need to have safely separated aircraft in the vicinity of airports that behave as required when entering and leaving. This work takes a step forward to achieve safe manned-unmanned vehicle operations while learning from each other. A motion planning algorithm is developed using methods of behavior cloning using the inputs of state, past trajectory, and intent to determine the subsequent actions with a discriminator reward based method. We use real-world aircraft flight data, augmented with the agent's goal to generate optimal trajectories as future actions. We use beam search to roll out our policy and select the best trajectory plan based on variable metrics. Our system incorporates continuous learning allowing autonomous aircraft to integrate into regular manned traffic flow safely.**

***Index Terms*— Motion and Path Planning, Imitation Learning, Social Navigation, Aerial Systems**

I. INTRODUCTION

The future of aviation will see a massive increase in the integration of low-altitude autonomous flying aircraft, air taxis, personal air vehicles, and commercial urban aerial mobility (UAM) systems. In an unstructured environment such as one near non-towered airspace, this entails a high degree of inter-aircraft interaction to remain safe and reason about the potential intentions of each 'agent.' General Aviation (GA) consists of all non-military, non-commercial civilian flight operations and takes up the bulk of air traffic in uncontrolled airspace. GA pilots primarily use visual observations of the path of other aircraft to detect, sense, and avoid. Augmenting this with knowledge of aviation regulations, current weather, and experience, pilots make predictions of the path of other aircraft. Terminal Airspace describes the airspace surrounding an airport, where aircraft converge and also perform maneuvers such as descent and turns. Since 2018, the Federal Aviation Administration (FAA) has seen an increase in the number of GA flight hours and a resulting rise in GA accident rates [1]. Conflicts between aircraft are common, and resolution involves direct pilot-to-pilot communication. Traditional detect and avoid systems (DAA such as TCAS or ACAS-X) are designed for the en-route phase of flight and try to prevent collisions rather than to reach goals and are unable to process multiple streams of information like vision and speech. Autonomous and crewed aircraft are strictly separated, which also limits flexibility and reduces the effectiveness of operations. In close vicinity of other vehicles or obstacles, they cannot react and avoid them to ensure safety.

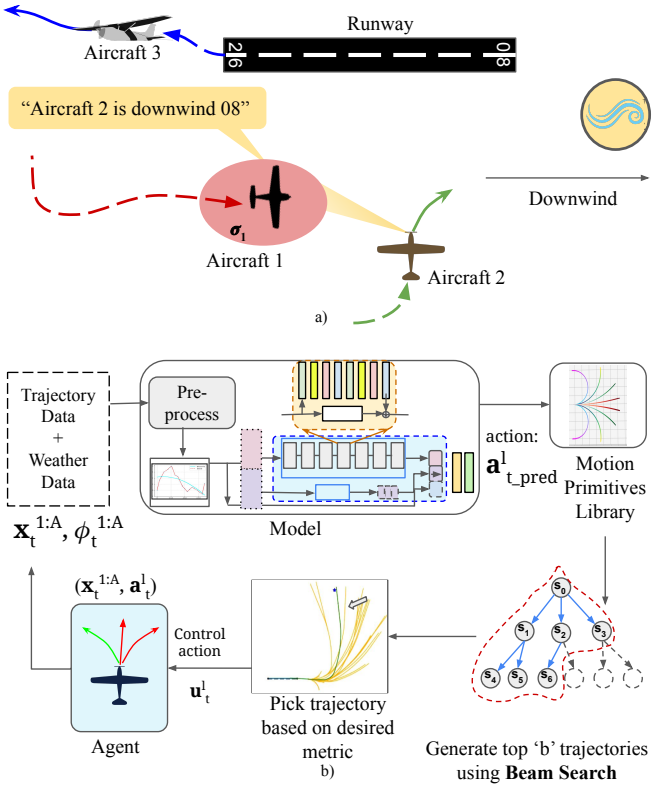


Fig. 1: Summary of the approach: a) The social navigation challenge for a pilot trying to land near a runway using a rectangular pattern in terminal airspace. b) Our solution involving imitation learning using trajectory and weather information to output the next action required to be taken.

Problem: Safe General Aviation Navigation: There is a need to have safely separated aircraft coordination, planning,

[1]Jasmine Jerry Aloor is with the Department of Aerospace Engineering, Indian Institute of Technology, Kharagpur, WB, India and Intern at The AirLab as a part of Robotics Institute Summer Scholars, Carnegie Mellon University, Pittsburgh, USA India jasminejerry@iitkgp.ac.in

[2]Jay Patrikar and Sebastian Scherer are with the Robotics Institute, School of Computer Science at Carnegie Mellon University, Pittsburgh, PA, USA {jpatrika, basti}@andrew.cmu.edu

and navigation in the absence of a centralized traffic control tower in the vicinity of airports. Pilots are expected to be socially compliant and follow the FAA guidelines to take actions that are acceptable within a social context. Following the basic rectangular traffic pattern reduces the possibility of conflicts at airports without an operating control tower. As the guidelines are not strictly enforced, there is scope for flexibility in each trajectory while maintaining separation. With each aircraft having a specific goal, requiring to maintain safety and behave as expected when entering and leaving the formation, this becomes a social navigation problem.

Solution: To develop solutions to social navigation, two main directions have emerged in recent research: reinforcement learning (RL) [2] and inverse reinforcement learning (IRL) [3] based approaches. RL methods have a shortcoming when the reward functions structure is not present that can satisfy the requirements of navigation. Training in the real world is difficult and expensive with new policies, especially for safety-critical systems. Using a multi-agent simulator presents the issue of accurately modeling human-like piloting behavior for training a policy, which leads us to the initial problem. On the other hand, learning from expert human demonstrations using IRL and imitation learning (IL) can attempt to provide a natural, human-like motion.

Sequential decision-making problems using IL leverage the information of the next best action choice taken by an expert to learn a near-optimal policy more effectively than RL [4]. Imitation learning is a popular research direction that uses supervised learning for decision-making. Here, a policy is learned that imitates recorded behaviors, and the simplest form is known as behavior cloning. We present a novel trajectory planning method that uses inputs of state, past trajectory, and intent to determine the next actions with a safety system using behavior cloning as our baseline algorithm.

We use real-world GA aircraft flight data as trajectories that are encoded as states, the latent space is augmented with the agent's goal and generates the next actions to be taken. We use beam search as our branch and bound algorithm to expand the possible trajectories that the agent can take to reach the goal. We explore different metrics to prune the beam-tree during inference. We test the algorithm and learned policy to analyze the limitations of the beam-search-enabled imitation-learning approach. The results show that learning from demonstrations is able to reach the goals in more than half of the tests.

We introduce usage of signal temporal logic (STL) as a means to encode hard and soft constraints on the agent's motion. We also discuss utilization of interactive learning by applying expert human intervention when the agent leaves a safe set. We aim to enable safety by repairing bad actions and re-positioning waypoints. Our approach has wide-ranging applications, from being a smart co-pilot for the regular crewed operation to allowing autonomous aircraft to safely integrate into regular manned traffic flow safely.

Contributions: The main contributions of this work are summarized as follows:

1) A curated set of dynamically feasible motion primitives frequently observed in GA aircraft near terminal airspace, generated and utilized for aircraft motion prediction.
2) A trajectory planning method is developed that uses information from agent-agent, agent-environment, and agent-context interactions to generate next actions.
3) A beam search policy and scoring functions are introduced as a means to select the path required to travel by the agent to reach the goal.

The organization of the remainder of the paper is as follows: In Section II, we provide an overview of the methods used in aircraft motion prediction, social navigation, beam search and imitation learning. In Section III, we introduce the approach, motion primitives generated, and our algorithm. In Section IV, we test our baseline model with selected metrics. In Sections V and VI we outline future work using interventions and conclude respectively.

II. Related Work

Aircraft motion prediction: As the number of air taxis, autonomous aircraft, and aerial mobility systems increases, there exists a need to have an efficient operation in shared airspace. Knowing the accurate trajectory models in the vicinity of terminal airspace is essential to develop advanced control technologies during close-proximity interactions. Previous works involved predicting motion by estimation of the aircraft's state and using dynamic equations to propagate the estimate [5], combining different modes of operation to generate a trajectory [6] and learning probabilistic representations from historical data [7], [8]. Recent research has used Gaussian Mixture Models (GMMs) to learn aircraft trajectory positions and deviations from intended behavior in terminal airspace [9], [10].

Social navigation: Focusing on the prediction of a single aircraft's trajectory would leave out the information of interaction with multiple aircraft and the effects on all trajectories. Socially compliant navigation that is focused on modeling human pedestrian behavior and motion has been extensively studied [11] where an S-LSTM model uses a 'social pool' layer connecting to every agent's trajectory network to account for the inter-agent dynamics. Recent research [12], [13] applies this to predict multi-aircraft trajectories.

Beam Search: Beam search algorithm is a form of pruned breadth-first search, progressing level by level without backtracking [14]. It is a heuristic approach, where the algorithm expands the best b promising nodes (instead of all nodes) at each level, where b is the beam width. It has been widely used in structured prediction tasks such as machine translation and speech recognition [15], [16] to image captioning [17]. The vanilla beam search algorithm was developed as an attempt to reach a optimal (or sub-optimal) solution given memory constraints on large search spaces.

Imitation Learning: With an increase in the collection of expert demonstration data, many traditional imitation learning algorithms like behavior cloning have been developed policies that mimic human experts' directly [3], or via

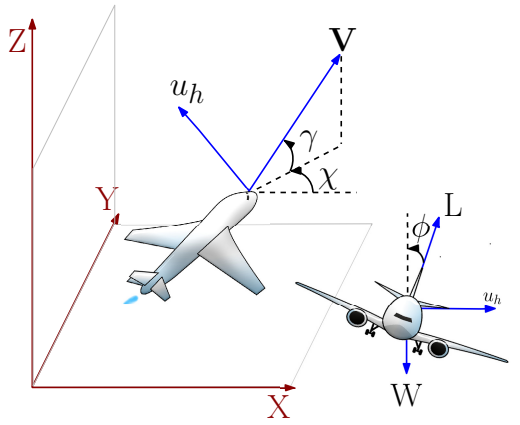


Fig. 2: Fixed wing 3D Coordinate frame

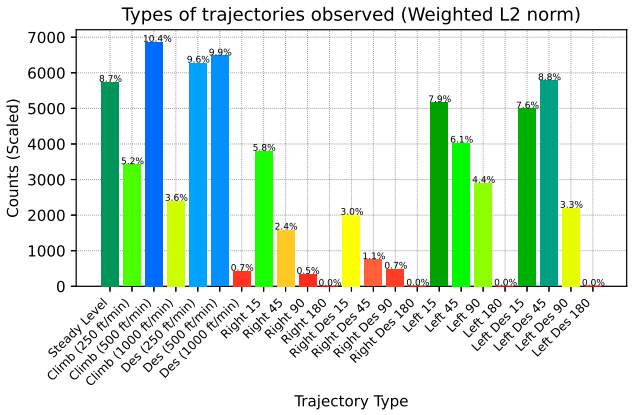


Fig. 3: Trajectory Classification: The broad categories of trajectories of the dataset along with climb rates (ft/min) and turning (heading angle °) are plotted

feedback querying [18], [19]. DAGGER algorithm [20] is an online sampling framework that trains their current policy by collecting the correct action labels from the expert. As the expert actions are aggregated, the algorithm trains a new policy with significant improvement. However, it becomes inefficient when querying at every state visited by the algorithm that the expert would normally not visit, and the learner keeps all control with the expert receiving no feedback. HG-DAGGER algorithm [21] is an extension that allows selective querying of the expert where the novice policy is rolled out until it enters an unsafe region. The expert has full control of the system and returns it to a safe and stable state-space during which the expert action labels are collected. The recent Expert Intervention Learning (EIL) algorithm [22] learns the expert actions through demonstrations as well as the timing of the correction. The policy minimizes the querying of the expert over time and quickly learns the good and bad states, thereby reducing the need for interventions. A formulation similar to Inverse Reinforcement Learning uses generative adversarial training to fit distributions of states and actions. The algorithm, Generative Adversarial Imitation Learning (GAIL) [23] learns policies directly from data. GAIL, is a popular formulation that explores policies closer to the expert actions using a discriminator reward. The ability to leverage it for learning policies without the need to recover the expert's reward function is a key capability we use. However, these works do not provide a guarantee of the safety of the system. We build on these to develop a learning algorithm that makes decisions using expert data and discuss future directions to the work.

III. Problem Formulation

A. Trajectory Data

This work uses the recently-released *TrajAir*[1] dataset, which is collected at the Pittsburgh-Butler Regional Airport (ICAO:KBTP) [24]. KBTP is a single runway GA airport located 10 miles North of the city of Pittsburgh, Pennsylvania. Non towered airports have an airport traffic pattern established for aircraft entering and leaving with specifications that include the direction and procedures. Airport traffic patterns are rectangular-shaped, developed to ensure that air traffic is flown into and out of an airport safely and in an orderly manner. KBTP has Left Traffic patterns for its runway, where all turns in the pattern are to the left. The dataset trajectories are smoothed using B-spline (basis-spline) approximate representation of order 2, an order k B-spline is formed by joining polynomials of degree $k-1$.

[1]http://theairlab.org/trajair/

B. Approach

We model the multi-agent system as a sequential decision making problem in continuous space, composed of A agents (vehicles) that interact over T time steps. At time t, let $\mathbf{x}_t^a = (x_t^a, y_t^a, z_t^a)$ and ϕ_t^a denote the position and weather context (taken as wind velocities) of the a^{th} agent. Let $\mathbf{x}_{t_1:t_2}^{1:A}$, $\phi_{t_1:t_2}^{1:A}$ and $\mathbf{a}_{t_1:t_2}^{1:A}$ denote the trajectories, context and actions over a $\{t_1, \ldots, t_2\}$ time-horizon for all $\{1, \ldots, A\}$ agents in that scene.

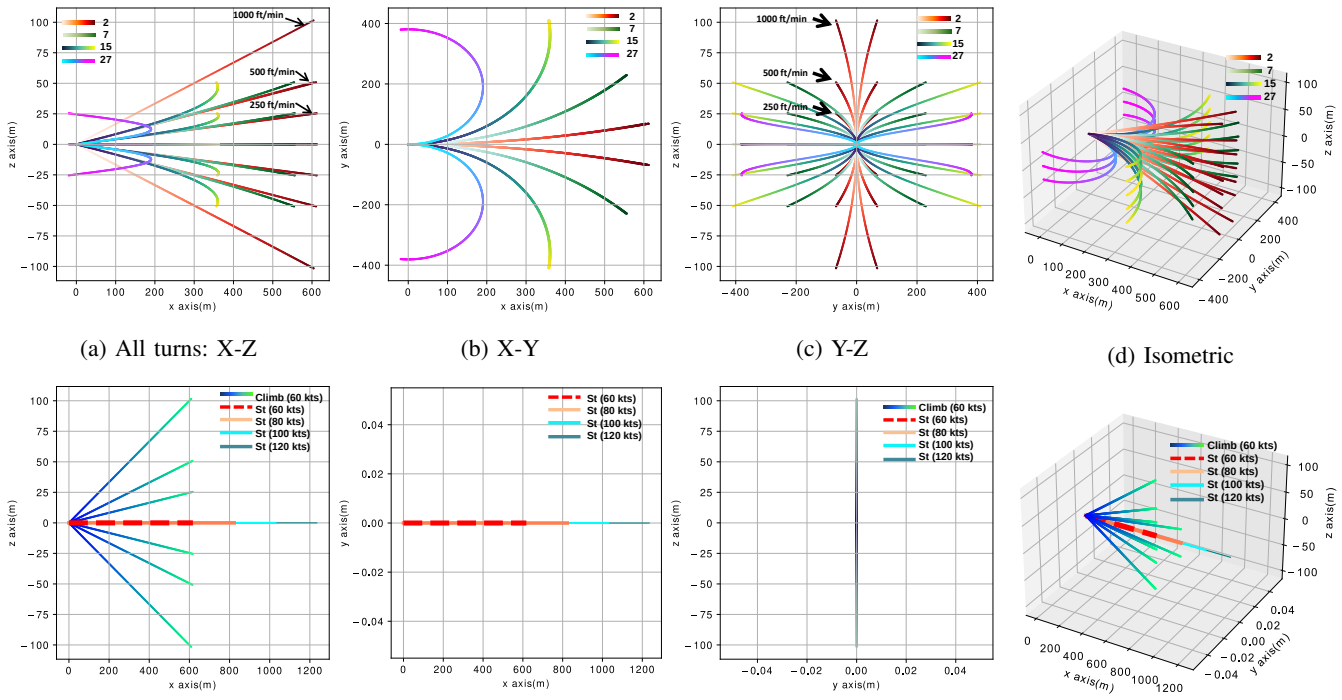
We formulate the problem as finding a distribution of future actions $\hat{\mathbf{a}}_{t_{obs}:t_{pred}+t_{obs}}^{1:A}$ conditioned on the past trajectories $\mathbf{x}_{1:t_{obs}}^{1:A}$ and context $\phi_{t:t_{obs}}^{1:A}$, where, t_{obs} is the observation time window and t_{pred} is the future time horizon.

$$\hat{\mathbf{a}}_{t_{obs}:t_{pred}+t_{obs}}^{1:A} \sim p(\hat{\mathbf{a}}_{t_{obs}:t_{pred}+t_{obs}}^{1:A} \mid \mathbf{x}_{1:t_{obs}}^{1:A}, \phi_{t:t_{obs}}^{1:A}) \quad (1)$$

The action space $\mathcal{A}$ consists of a fixed library of motion primitives represented by $\mathbf{x}_{1:t_{obs}}^{l}$, where, $\mathbf{x}_t^l = (x_t^l, y_t^l, z_t^l)$ denote the positions of each trajectory in the library at time t (see Fig. 4). Each action $\mathbf{a}_t^i$ is a pre-defined control (consisting of aircraft inertial speed v, rate of climb $\dot{z}$ and bank angle ϕ) and a sequence of resulting trajectories created by forward integration of the kinematic equations with the initial state set by the motion primitive library $\mathbf{a}_{t_{obs}}^i = (\hat{\mathbf{u}}_{1:t_{obs}}^{1:A}, \hat{\mathbf{x}}_{1:t_{obs}}^{1:A})$.

C. Motion primitives

We consider a fixed-wing aircraft flying in three-dimensional Euclidean space SO(3) (Fig. 2). The kinematic



(a) All turns: X-Z (b) X-Y (c) Y-Z (d) Isometric

(e) Straight, Climb, Descent: X-Z (f) X-Y (g) Y-Z (h) Isometric

Fig. 4: Motion Primitives: The three broad categories of motion chosen are steady level flight, steady climb/descent, and coordinated turns. Here we show the X-Y, Y-Z, X-Z, and isometric views of the motion primitives all at 60 knots. Fig. 4e-4h also has steady level flight plotted for velocities 80, 100, and 120 knots

equations (2) are used to calculate a set of trajectories in the inertial frame.

$$\dot{x} = v\cos\gamma\cos\chi \tag{2a}$$
$$\dot{y} = v\cos\gamma\sin\chi \tag{2b}$$
$$\dot{z} = v\sin\gamma \tag{2c}$$
$$\dot{\chi} = \frac{u_h}{v\cos\gamma} \tag{2d}$$

where, $(x, y, z) \in \mathbb{R}^3$, $v = \|\mathbf{V}\|$ is the inertial speed and the velocity vector $\mathbf{V}$ is a function of the inertial speed (v), elevation angle (γ) and azimuth angle (χ). The lateral acceleration is represented by u_h, which is a function of the aircraft's bank angle, ϕ. We choose three broad categories of motion

1) Steady Level, Climb and Descent flight
2) Coordinated Turns (Right, Left)

We vary the velocities from 60-120 knots in steps of 20 knots, we choose the rates of climb and descents ($\dot{z}$) to vary from $\{0, \pm 250, \pm 500, \pm 1000\}$ ft/min and set the bank angles as $\{2°, 7°, 15°, 27°\}$ to turn by $\{15°, 45°, 90°, 180°\}$ heading respectively over the chosen time-horizon.

D. Trajectory Matching

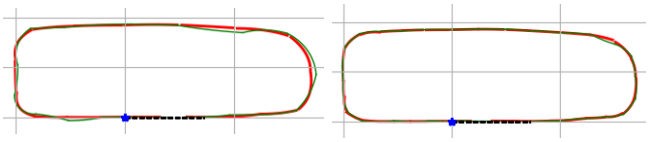
We classify the trajectories in the dataset into different climb rates based on the start and end heights ($\delta z/\delta t$). We then match the generated trajectories in the library with the trajectory dataset to understand the types of motions encountered using a weighted L2 Euclidean error distance over (x, y) points on the entire trajectory. We represent the minimum value and the matched library trajectories over all the scenes in $L(\mathbf{x}^a_{1:t_{obs}})$ and $T(\mathbf{x}^a_{1:t_{obs}})$ respectively. These are the expert trajectories.

$$\underset{d\in\mathcal{D}}{\forall} L(\mathbf{x}^a_{1:t_{obs}}) = \min_{\substack{l\in\{1\ldots\mathcal{N}\}\\ s.t.\ \dot{z}^l_t = \dot{z}^a_t}} \left(\sum_{i=t_1}^{t_2} \sum_{j\in(x,y)} (w_i[x_i^{l(j)} - x_i^{a(j)}])^2 \right)^{\frac{1}{2}} \tag{3a}$$

$$\underset{d\in\mathcal{D}}{\forall} T(\mathbf{x}^a_{1:t_{obs}}) = \arg\min_{\substack{l\in\{1\ldots\mathcal{N}\}\\ s.t.\ \dot{z}^l_t = \dot{z}^a_t}} \left(\sum_{i=t_1}^{t_2} \sum_{j\in(x,y)} (w_i[x_i^{l(j)} - x_i^{a(j)}])^2 \right)^{\frac{1}{2}} \tag{3b}$$

where, $(x_t^{l(x)}, x_t^{l(y)}) = (x_t^l, y_t^l)$, w_i is the weights applied to the segments, $\mathcal{N}$ is the total number of trajectories in the library and $\mathcal{D}$ is the number of scenes in the dataset. We choose the weights of the last quarter of the trajectory to have double the value of than the rest of the trajectory. The preliminary analysis showed the most frequent types of motions observed were straight steady-level flight, steady climbs, descents and left turns, which is expected from the traffic pattern to be followed near KBTP. We also observe a low number of steady steep climb and descent rates and no sharp turns with a steep climb and descent rates.

1) Suitability of the motion primitives chosen: We check for the suitability of our motion primitives by checking how



(a) Actions are updated after execution for 20 seconds: Maximum average displacement error is 125.15 m

(b) Actions are updated after execution for 10 seconds: Maximum average displacement error is 82.12 m

Fig. 5: The motion primitives are stitched end to end to demonstrate their suitability for a full takeoff and landing loop trajectory performed at the airport from the general aviation dataset.

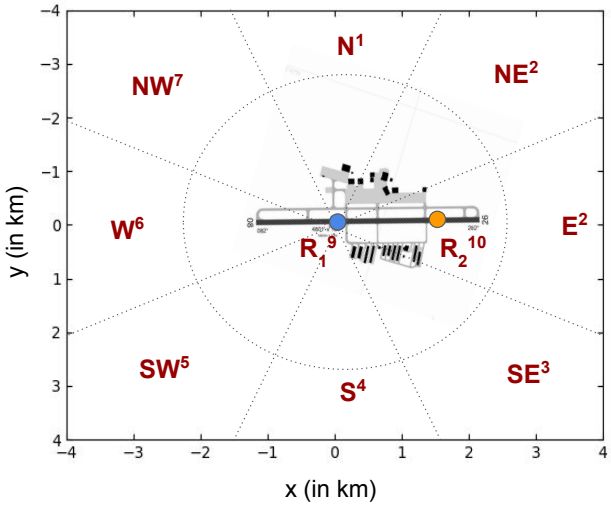


Fig. 6: Goal representation is in the form of a one-hot vector where each region the aircraft travels is the respective goal element in the vector

well it can be used to stitch together a full pattern trajectory. We use 10 second and 20 seconds action updates and find that the given set of motion primitives have small average displacement error as seen in Fig. 5. The 10 second update provides an even smaller error.

E. Model Details

We propose a behavior cloning algorithm that takes as input the past trajectories of all the agents, along with the weather context, to predict their possible action distribution. The network uses a Temporal Convolutional Network (TCN) to process the sequential trajectory data. TCN layers encode a trajectory's spatio-temporal information into a latent vector without losing the underlying data's temporal (causal) relations in the underlying data [25]. We use TCNs as an alternative to using LSTMs [12] for encoding the trajectories.

We break the trajectories in a scene into sequences of length $t_{obs}+t_{pred}$ with a certain minimum number of agents constant across each sequence. The number of agents can change from sequence to sequence. For each agent in a given scene, the raw trajectory in absolute coordinates is encoded using the same TCN layers.

$$h^a_{obs} = TCN_{obs}(\mathbf{x}^a_{1:t_{obs}}) \;\forall\; a \in \{1,\ldots,A\} \tag{4}$$

where, h^a_{obs} is the encoded vector of agent a. The encoded information of the other agents is appended to the ego agent's encoded state.

$$\overline{h}^a_{obs} = \forall_{b\in\{1,\ldots,A\}\neq a}\ \{h^b_{obs}\} \tag{5}$$

$$h^a_{enc} = h^a_{obs} \oplus \overline{h}^a_{obs} \;\forall\; a \in \{1,\ldots,A\} \tag{6}$$

In the goal-conditioned network, we append the agent's goal as a one-hot vector representation of the final goal of a particular agent as the eight cardinal directions along with two runway ends as shown in Fig. 6. The vector is represented as [N, NE, E, SE, S, SW, W, NW, R_1, R_2], with each element representing the final region the aircraft is desired to reach.

Finally, the concatenated latent vector is passed through two Fully Connected layers to predict the next action from the motion primitive library.

$$a^l_{t_{obs}:t_{obs}+t_{pred}} = MLP(h^a_{enc}) \tag{7}$$

We use a goal conditioned generative adversarial imitation learning (GAIL) method based on [26]. A discriminator D_ψ is trained to distinguish expert transitions $(s,a) \sim \tau_{\text{expert}}, T(\mathbf{x}^a_{1:t_{obs}})$, from agent transitions $(s,a) \sim \tau_{\text{agent}}, a^l_{t_{obs}:t_{obs}+t_{pred}}$, while the agent is trained to 'confuse' the discriminator into thinking itself is the expert. The discriminator is trained to minimize $\mathcal{L}_{GAIL} = \mathbb{E}_{(s,a)\sim\tau_{\text{agem}}}[\log D_\psi(s,a)] + \mathbb{E}_{(s,a)\sim\tau_{\text{exper}}}[\log(1-D_\psi(s,a))]$; while the agent is formally trained to maximize $\mathbb{E}_{(s,a)\sim\tau_{\text{agent}}}[\log D_\psi(s,a)]$ by using the output of the discriminator $\log D_\psi(s,a)$ as reward [23]. With the goal conditioned network, the discriminator is also conditioned on the goal becoming $D_\psi(a,s,g)$, and we minimize

$$\begin{aligned}\mathcal{L}_{GAIL}(D_\psi,) =& \mathbb{E}_{(s,a,g)\sim\text{policy}}[\log D_\psi(a,s,g)] + \\ & \mathbb{E}_{(s,a,g)\sim\text{expert}}[\log(1-D_\psi(a,s,g))]\end{aligned} \tag{8}$$

The $\mathcal{L}_{act}$ measures how close the predicted action is to the expert action using a Cross-Entropy Loss (CEL).

$$\mathcal{L}_{act} = CEL(\hat{\mathbf{a}}^a_{t_{obs}:t_{pred}+t_{obs}}, T(\mathbf{x}^a_{1:t_{obs}})) \tag{9}$$

The combination of these two loss functions is used to train the model.

$$\mathcal{L}_{total} = \mathcal{L}_{act} + \mathcal{L}_{GAIL} \tag{10}$$

For training, we use the Adam optimizer with a learning rate of $1e-4$.

IV. Evaluations and Discussion

We evaluate the trajectories on 111 days of data in the TrajAir dataset. To accurately capture the dynamics of a particular motion in x-y-z coordinates, we use $t_{obs} = 20$ sec and $t_{pred} = 20$ sec. We select the action predicted from the trajectory library at $t_{obs}+1$ and match it with the trajectory followed by the agent in that scene.

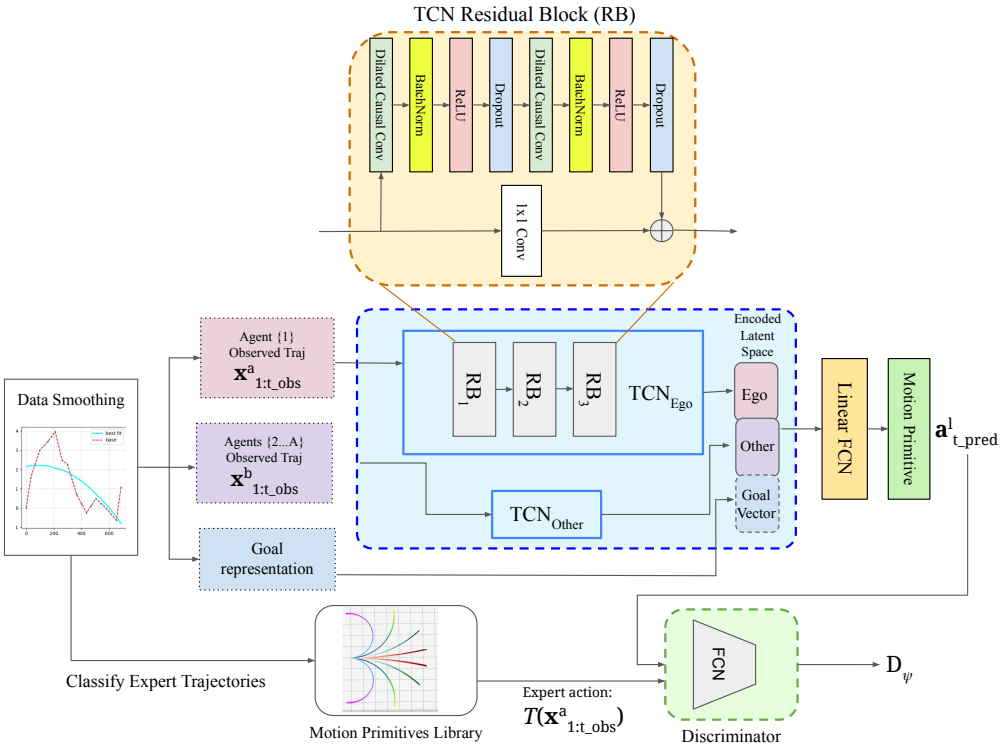


Fig. 7: Overview of the policy model: The smoothed trajectories of all agents are passed into the TCN separately. For each agent, the encoded space of all other agents is concatenated after a max pool. If the network inputs the agent's goal (g^a), then it is appended to the encoded variable along with the previous iteration's prediction. The expert actions and the network policy's actions are sent to the discriminator to get the reward used while training.

The evaluation metrics used are

1) Average Displacement Error (ADE): L2 Euclidean distance over the entire trajectory
2) Weighted Displacement Error (wADE): Weighted end-point L2 Euclidean distance over the whole trajectory.
3) Cross entropy loss (CEL): For the selection of right actions from the library.

We use results of the test data to record all the metrics (ADE/wADE/CEL scores).

Metrics	ADE (m)	wADE (m)	CEL
Entire dataset	126.52	212.33	1.60

TABLE I: Evaluation metrics for the best case observed in the test data.

A. *Beam Search Inference*

Beam search is an improved version of greedy search given a search space G, we create a beam search space $G_k = (V_k, E_k)$, where k is a hyper-parameter named beam width. V_k is the set of beams formed, E_k is the set of trajectories taken in a beam path, and each node is represented by b. The initial beam state $b_{(0)} \in V_k$ is the singleton set with the initial state $v_{(0)} \in V$. At each time step, we select k tokens with the highest conditional probabilities as output from our policy network. Each of them will be the first token of k candidate output sequences, respectively. At each subsequent time step, based on the k candidate output sequences at the previous time step, we continue to select k candidate output sequences with the highest conditional probabilities from $\mathcal{N}$ possible choices. We expand the search space until we reach the beam depth d, which is set as our planning horizon.

In our experiments, we choose two sets of beam widths and depths

1) k = 15, d = 280 s
2) k = 50, d = 280 s

With an input vector, $b_{(0)}$, selected from the dataset to be a trajectory fed to the network we get the a set of trajectories chosen based on least Euclidean distance from original goal location (seen in Fig. 8a and 8b) or the generated trajectory goal to match original goal-vector (seen in Fig. 8c and 8d. We use two metrics to decide our choice of trajectory from the beam tree:

1) ability to reach desired goal location.
2) ability to avoid collisions.

The results are listed in Table II.

Metrics	Euclidean distance		One-hot goal check	
	k = 15	k = 50	k = 15	k = 50
Reach Goal	$\geq$ 50%	$\geq$ 60%	$\geq$ 60%	$\geq$ 75%

TABLE II: Results for beam search metric: reach goal, from test data.

B. Signal Temporal Logic Constraints

We encode the metrics of reaching desired goal and selecting trajectories away from a threshold distance from other agents for collision avoidance (hard constraint) using signal temporal logic (STL).

$$\psi_1 = \Diamond_{[t_{obs}:t_{pred}]}((\mathbf{x} > \mathbf{x}_{\text{goal}} - \delta\mathbf{x}) \ \wedge \ (\mathbf{x} < \mathbf{x}_{\text{goal}} + \delta\mathbf{x})) \tag{11a}$$

$$\psi_2 = \Box_{[t_{obs}:t_{pred}]}(\mathbf{x}^a \ \neg \ (\text{inside } \mathbf{x}^b)) \tag{11b}$$

The robustness values of the STL constraints are weighted and summed to the ranking by the beam search algorithm. We regularize the beam search ranking processing by augmenting the function with a term that penalizes negative robustness values. Specifically, the ranking function becomes,

$$\mathcal{V} = \mathcal{V}_0 + \gamma\mathcal{V}_\psi \tag{12}$$

where $\mathcal{V}$ is the original rank function (e.g., beam search rank), and, in this example, $\mathcal{V}_\psi$ is the robustness loss. The model with robustness regularization is able to obey ψ_1 and ψ_2 more robustly.

C. Visualization

Figure 9 shows the qualitative results for a right turn scenario along with the X-Plane visualization. The x-y tracked trajectory is shown on the top right, which is a segment from the rectangular pattern followed by the agent in the dataset. Table I shows the quantitative results for the goalGAIL based behavior cloning network.

V. Future Works

Behavior cloning is known to require a large number of expert demonstrations and, at the same time, deviates significantly when encountered with out-of-distribution states [3], [27]. An aircraft pilot has multiple directives needed to follow to navigate safely in an airspace. We seek to determine all constraints and encode them appropriately to be included while training our model. Approaches that can collect feedback interactively by querying the expert have been explored [18], but can be impractical due to the algorithm querying in states that are not natural for the human expert. Additionally, the lack of full control to the expert causes a delayed response. Another method is to provide the human expert full control and the option to intervene when required, which is more natural and provides the algorithm the additional information of the 'bad' states to avoid and technique to recover from it [21], [22]. We plan to build on this method to use interventions from a supervisor to handle achieve safe manned-unmanned vehicle teaming to improve the system performance and have each teammate learn from each other in different aircraft operations. When our agent goes out of the safe zone, we will use the expert intervention to bring it back to safety and update the learner algorithm.

VI. Conclusion

This work presents a model using imitation learning for socially aware navigation of dynamical systems in the general aviation domain. It also offers a curated set of motion primitives for aircraft that are frequently observed in general aviation navigation. The current results for behavior cloning are good given the inputs of trajectory and weather information. It is, however, the first step in the aviation domain for social robotics and automation. There is a great scope for improvement using interactive learning and intervention-based methods that will be compared to our proposed method.

Acknowledgment

This work was sponsored by the AirLab at the Robotics Institute, Carnegie Mellon University, as a part of the Robotics Institute Summer Scholars (RISS) program.

References

[1] "Preliminary aviation statistics," USA National Transportation Safety Board, 2000-2019. [Online]. Available: http://www.ntsb.gov/investigations/data/pages/aviation_stats.aspx

[2] M. Everett, Y. F. Chen, and J. P. How, "Collision avoidance in pedestrian-rich environments with deep reinforcement learning," *IEEE Access*, vol. 9, pp. 10 357–10 377, 2021.

[3] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.

[4] W. Sun, A. Venkatraman, G. J. Gordon, B. Boots, and J. A. Bagnell, "Deeply aggrevated: Differentiable imitation learning for sequential prediction," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3309–3318.

[5] G. Chatterji, "Short-term trajectory prediction methods," in *Guidance, Navigation, and Control Conference and Exhibit*, 1999, p. 4233.

[6] R. Slattery and Y. Zhao, "Trajectory synthesis for air traffic automation," *Journal of guidance, control, and dynamics*, vol. 20, no. 2, pp. 232–238, 1997.

[7] M. J. Kochenderfer, L. P. Espindle, J. K. Kuchar, and J. D. Griffith, "A comprehensive aircraft encounter model of the national airspace system," *Lincoln Laboratory Journal*, vol. 17, no. 2, pp. 41–53, 2008.

[8] C. Lowe and J. P. How, "Learning and predicting pilot behavior in uncontrolled airspace," in *AIAA Infotech@ Aerospace*, 2015, p. 1199.

[9] S. T. Barratt, M. J. Kochenderfer, and S. P. Boyd, "Learning probabilistic trajectory models of aircraft in terminal airspace from position data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3536–3545, 2018.

[10] S. Jung and M. J. Kochenderfer, "Learning terminal airspace traffic models from flight tracks and procedures," in *2019 IEEE/AIAA 38th Digital Avionics Systems Conference (DASC)*. IEEE, 2019, pp. 1–8.

[11] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

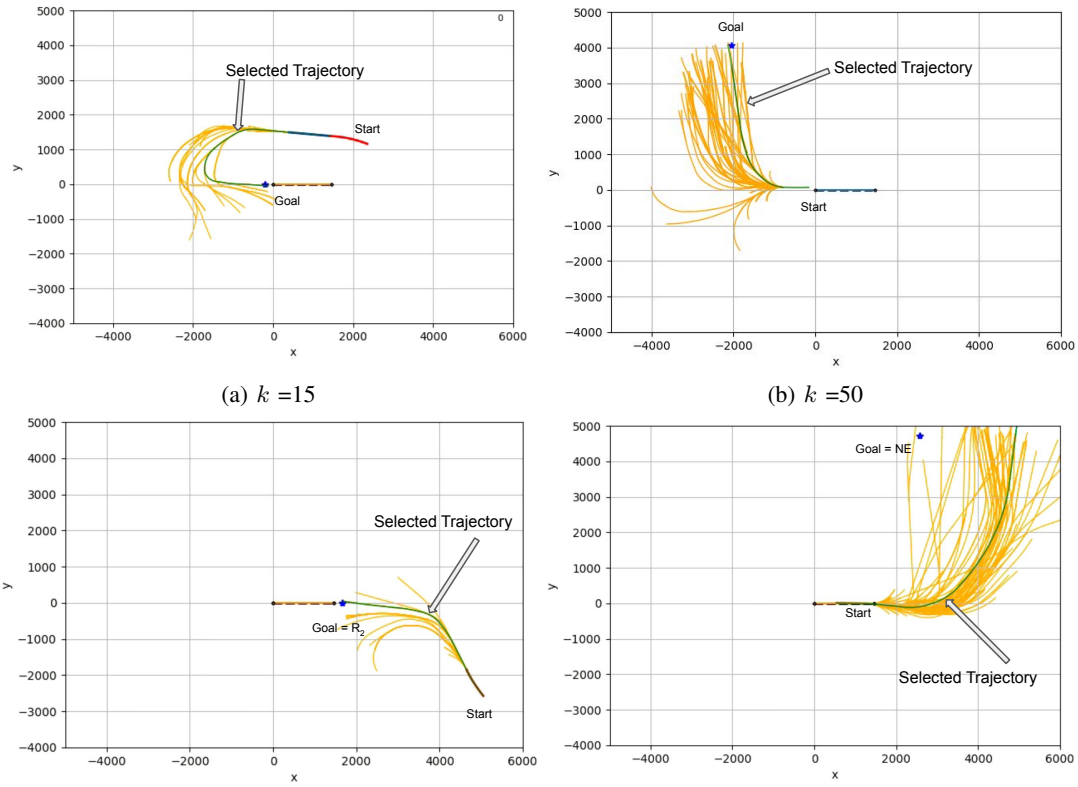
[12] Z. Zhao, W. Zeng, Z. Quan, M. Chen, and Z. Yang, "Aircraft trajectory prediction using deep long short-term memory networks," in *CICTP 2019*, 2019, pp. 124–135.

[13] Z. Xu, W. Zeng, X. Chu, and P. Cao, "Multi-aircraft trajectory collaborative prediction based on social long short-term memory network," *Aerospace*, vol. 8, no. 4, p. 115, 2021.

[14] I. Sabuncuoglu and M. Bayiz, "Job shop scheduling with beam search," *European Journal of Operational Research*, vol. 118, no. 2, pp. 390–412, 1999.

[15] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[16] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. Ieee, 2013, pp. 6645–6649.



(a) k =15 (b) k =50

(c) k =15 (d) k =50

Fig. 8: Beam Tree generated based on two metrics: (a), (b): trajectory goal has least Euclidean distance from original goal location; (c), (d): trajectory goal to match original goal-vector

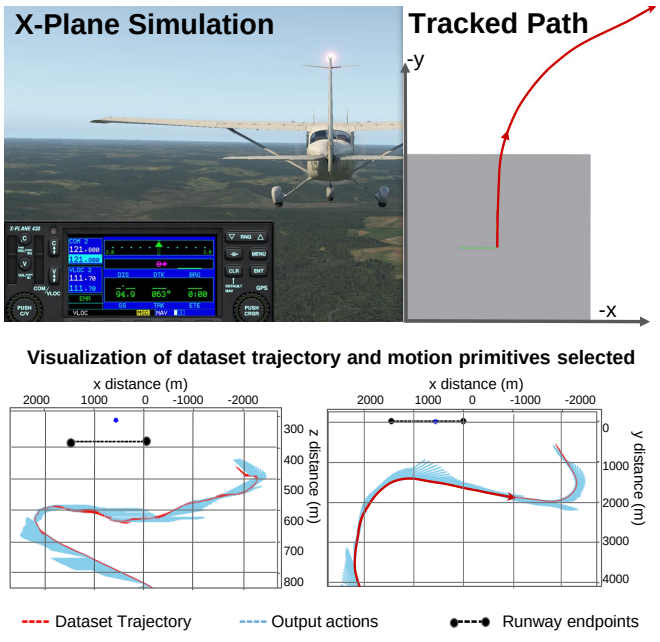


Fig. 9: Actions input to the X-Plane visualizer. Top right: The tracked trajectory. Bottom: The dataset trajectory and predicted actions

[17] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.

[18] S. Ross and D. Bagnell, "Efficient reductions for imitation learning," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 661–668.

[19] S. Ross, G. J. Gordon, and J. A. Bagnell, "No-regret reductions for imitation learning and structured prediction," in *In AISTATS*. Citeseer, 2011.

[20] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[21] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, "Hg-dagger: Interactive imitation learning with human experts," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8077–8083.

[22] J. Spencer, S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa, "Learning from interventions," in *Robotics: Science and Systems (RSS)*, 2020.

[23] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, pp. 4565–4573, 2016.

[24] J. Patrikar, B. Moon, S. Ghosh, J. Oh, and S. Scherer, "Trajair: A general aviation trajectory dataset," Jun 2021. [Online]. Available: https://doi.org/10.1184/R1/14866251.v1

[25] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[26] Y. Ding, C. Florensa, M. Phielipp, and P. Abbeel, "Goal-conditioned imitation learning," *arXiv preprint arXiv:1906.05838*, 2019.

[27] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and

J. Peters, "An algorithmic perspective on imitation learning," *arXiv preprint arXiv:1811.06711*, 2018.