Carnegie Mellon University

DISCLOSURE OF INTELLECTUAL PROPERTY SOURCE CODE/ COPYRIGHTS/ APPS

All information requested in this document must be completed in order to expeditiously process this Disclosure. Any missing or incomplete information may delay processing your submission.

1. **Title**: AI Pilot: Close-Proximity Safe and Seamless Operation of Manned and Unmanned Aircraft in Shared Airspace

2. **Creator(s): By signing this form the undersigned Creators acknowledge and agree that they are bound by Carnegie Mellon University's Intellectual Prop erty Policy, on line at http://www.cmu.edu/policies/documents/IntellProp.html. Original signatures for all Creators are required. Therefore, by signing below: (i) if the Policy allows CMU to own this intellectual property and its associated intellectual p roperty rights, you hereby assign to Carnegie Mellon any and all ownership you have in such intellectual property and intellectual property rights; and (ii) if the Policy allows CMU to receive license rights to this intellectual property and its associated intellectual property rights, you hereby grant to CMU any and all such licenses. Original signatures for all Creators are required.**

a. **Lead Creator:**

Sebastian Scherer	Sebastian Scherer	10/24/2022
print or type name	signature	date

Robotics	412-736-6821	basti@andrew.cmu.edu	12.5
department	phone	current e-mail	% of contribution

Faculty _____

Carnegie Mellon Employment Status at the time the intellectual property was created (Faculty, Staff, Student, Visitor, Courtesy, etc.)

_____

Full institutional address (if not affiliated with Carnegie Mellon)

Pittsburgh, PA, USA _____
City/State/Country of Residence

USA _____
Country of Citizenship

b. **Next Creator:**

Jean Oh		10/24/2022
print or type name	signature	date

Robotics		hyaejino@andrew.cmu.edu	12.5
department	phone	current e-mail	% of contribution

Faculty _____

Carnegie Mellon Employment Status at the time the intellectual property was created (Faculty, Staff, Student, Visitor, Courtesy, etc.)

_____

Full institutional address (if not affiliated with Carnegie Mellon)

Pittsburgh, PA, USA _____
City/State/Country o f Residence

_____USA _____
Country of Citizenship

c. **Next Creator:**

Jay Patrikar _____ Jay Patrikar _____ 10/24/2022
print or type name signature date

Robotics _____ 412-692-1516 _____ _jaypat@cmu.edu 12.5
department phone current e-mail % of contribution

Student _____

Carnegie Mellon Employment Status at the time the intellectual property was created (Faculty, Staff, Student, Visitor, Courtesy, etc.)

_____

Full institutional address (if not affiliated with Carnegie Mellon)

_____
City/State/Country of Residence

_____
Country of Citizenship

d. **Next Creator:**

Ingrid Navarro _____ Ingrid Navarro Anaya _____ 10/24/2022
print or type name signature date

Robotics Institute _____ 412-999-5462 _____ ingridn@cmu.edu ___12.5_____
department phone current e-mail % of contribution

Studen t _____

Carnegie Mellon Employment Status at the time the intellectual property was created (Faculty, Staff, Student, Visitor, Courtesy, etc.)

_____

full institutional address (if not affiliated with Carnegie Mellon)

_____
City/State/Country of Residence

Mexico
Country of Citizenship

e. Next Creator:

Jasmine Jerry Aloor _____  10/24/2022
print or type name signature date

Robotics _____ 857-706-9454 jasminejerry@yahoo.in 12.5
department phone current e-mail % of contribution

Vistor Student

Carnegie Mellon Employment Status at the time the intellectual property was created (Faculty, Staff, Student, Visitor, Courtesy, etc.)

Indian Institute of Technology, Kharagpur, West Bengal, India Postal: 721302
full institutional address (if not affiliated with Carnegie Mellon)

Kharagpur, West Bengal, India
City/State/Country of Residence

India
Country of Citizenship

f. Next Creator:

Rohan Baijal _____  11/10/2022
print or type name signature date

Robotics _____ +91-9958206389 rohanbaijal2702@gmail.com 12.5
department phone current e-mail % of contribution

Visitor Student

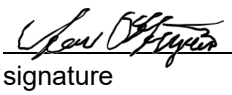
Carnegie Mellon Employment Status at the time the intellectual property was created (Faculty, Staff, Student, Visitor, Courtesy, etc.)

Indian Institute of Technology, Kanpur, Uttar Pradesh, India, Postal:208016
full institutional address (if not affiliated with Carnegie Mellon)

Kanpur, Uttar Pradesh, India
City/State/Country of Residence

India
Country of Citizenship

g. Next Creator:

Ian Higgins _____  11/21/2022
print or type name signature date

RI CMU	414-477-4598	ihiggins@andrew.cmu.edu	12.5
department	phone	current e-mail	% of contribution

Staff _____

Carnegie Mellon Employment Status at the time the intellectual property was created (Faculty, Staff, Student, Visitor, Courtesy, etc.)

_____

full institutional address (if not affiliated with Carnegie Mellon)

Pittsburgh/PA/USA _____
City/State/Country of Residence

USA _____
Country of Citizenship

h. **Next Creator:**

Joao Dantas	Joao Dantas	10/24/2022
print or type name	signature	date

Robotics	**412-214-3010 jdantas@andrew.cmu.edu**	**12.5**
department	phone	current e-mail	% of contribution

Visitor Student _____

Carnegie Mellon Employment Status at the time the intellectual property was created (Faculty, Staff, Student, Visitor, Courtesy, etc.)

_____

full institutional address (if not affiliated with Carnegie Mellon)

Pittsburgh/PA/USA _____
City/State/Country of Residence

Brazil_____
Country of Citizenship

3. Please provide a short description of the function and use of the intellectual property being disclosed in the space below and, if applicable, a copy of the code, link to the code or instructions where it can be accessed.

The intellectual property consists of code that can be used as a system for providing an uncrewed vehicle or act as a co-pilot to coordinate actions with other crewed/uncrewed aircraft. Details are in the attached file.

4. Intellectual Property Protections

a. This Disclosure describes (please "X" all that apply):

___x__ Source Code
_____ Designs and/or other copyrightable materials
_____ Coretech or Coretech improvements (NREC only)

For the question #4b, please indicate the date in the format "Month/Day/ Year" (ex. 01/01/17).

b. State first date of:
a. Completion ___08/03/2022 _____
b. Publication/ Release (outside of CMU) _____

5. Sponsorship/ Use of 3rd Party Resources

a.

External Sponsor(s)	CMU Oracle #(s) AND	Contract or Grant #(s)
ARL IDIQ	52862.1A.1990753	W911NF-20-D-0002

(Your department administrator may be of assistance in identifying funding sources used.)

Have external sponsors been informed of or provided with the intellectual property? No

b. If funded under a grant or contract, please describe this intellectual property as one of the following:

_____ Background IP (developed prior to the grant but used in research funded by the grant)
_____ Background IP Improvement (developed both prior to and during the grant)
___x___ Foreground IP (developed only during the grant)

c. Internal Sponsor (Department Research Funds, etc.) _____

d. Was this intellectual property developed in collaboration with any other 3rd parties (companies, universities, etc.) or as a part of a research consortium? Please list below:
No

e. Have you used any software, libraries, etc. from other internal (e.g., CMU) sources (ex. projects or researchers) in the development of this technology or does the technology otherwise build upon earlier work at CMU? Please list below:

Yes, a background IP "2023 -100 Social -PatteRNN: An algorithm for socially -aware trajec tory prediction from motion patterns" (under submission) was used.

f. Was there any Open Source software, Creative Commons copyrights or other third party material used in the development of this intellectual property? Please list below:

The following externally developed BSD -licensed packages may be used: ROS, CUDA, Numpy, OpenCV, Pandas, Torch.

6. Have you / do you intend to release the Source Code pursuant to an Open Source license? If so, please indicate where the code is/ will be released (circle all that apply):

a. Your website
x b. SourceForge, github or other similar hosting provider
c. through a federal agency (e.g. NASA)
d. other (please identify) _____
e. Do NOT intend to release open source

7. How long would it take someone skilled in the art to recreate this copyrightable work? _12 months_

Please feel free to attach additional material or data that would provide us with helpful information.

Email the completed electronic copy of this Invention Disclosure form to:

innovation@cmu.edu

If unable to sign electronically, paper copies may be sent to:

Department Administrator
Center for Technology Transfer & Enterprise Creation
4615 Forbes Avenue, Suite 302

AI Pilot: Close-Proximity Safe and Seamless Operation of Manned and Unmanned Aircraft in Shared Airspace

1. Objectives: Develop an AI system to keep autonomous unmanned aircraft, in conjunction with manned traffic safely separated and behave as expected when entering and leaving the traffic/break/formation pattern.

2. Key Areas:

2.1. Intent Prediction

Understanding and predicting the intended motion of humans in an environment is an important skill that robots across various domains, such as aerial robotics, must be equipped with in order to enable safe interactions.

Motion and intent are influenced by several factors, making the task challenging. One factor is social behavior, which describes how agents interact, and it heavily depends on the context. Within AAM, goal points such as airports, pilots, ATC communications, and rules of air are often well-known. Using this definitive source of information and other implicit sources like weather can help decipher the intent of other aircraft and increase the length of reliable predictions. Respecting motion constraints is another relevant aspect in this setting, as it is related to restrictions that may arise from the agent's own physical constraints, or the constraints imposed by the environment, such as the topology of a scene, or the rules associated with it, *e.g.*, pilots should respect flying guidelines. Moreover, agent motion is driven by each agent's goals which vary depending on the situation and the type of agent: goals can be flexible or fixed, short or long-term, implicit or explicit. This varied nature makes them difficult to model. The majority of existing works on trajectory prediction are mainly focused on pedestrian behaviors where the context is relatively loosely defined, *i.e.*, except for the social norm that pedestrians try to maintain a comfortable distance from others, there are few rules that guide pedestrian behavior.

2.2. Planning

As the FAR rules only specify a rough guideline, autonomous vehicles must be equipped with the capability to make flexible decisions to comply with traffic norms and generalize to arbitrary situations. The idea of rule-based navigation is to learn to follow the observed traffic patterns. Generating actions that are not only safe but also follow rules is thus critical in generating behavior that is acceptable to human pilots co-habiting the same airspace.

Deep reinforcement learning methods have been successful in learning autonomous navigation. In sequential decision-making, policies are often represented by a Markov Decision Process

(MDP) with a well-defined reward function. The complexity of manually defining a reward function hampers the widespread applicability of reinforcement learning and optimal control algorithms. Learning a policy from expert demonstrations (LfD or imitation learning) has led to a variety of applications with state-of-the-art performance. Behavior cloning (BC) enables the implementation of supervised learning like methods but requires a large quantity of demonstration data and interactive learning to reduce compounding errors due to out-of-distribution data or covariate shift. These approaches, however, have the limitations of being sensitive to sub-optimal demonstrations and the inflexibility to generalize to other applications. Generative adversarial imitation learning (GAIL) and related works have approached imitation learning as a system consisting of a learner's policy and a discriminator network that attempts to distinguish between learner transitions from expert transitions. GAIL attempts to learn policies directly from simulation and has the limitations of sub-optimal demonstrations, and convergence becomes slow due to stability issues with low data.

Temporal logics such as Signal Temporal Logic (STL) provide a mathematically robust representation to encode such spatio-temporal constraints. They can be used to logically specify desired behavior translated from requirements expressed in natural language. STL is used to specify properties over real-valued dense time signals often generated by continuous dynamical systems. Quantitative semantics associated with STL provides a real value called robustness which quantifies the degree of satisfaction or violation. This property enables the designer to encode domain-specific constraints and quantitatively measure their satisfaction with motion planning and control. In order to infuse the STL specifications to improve the base LfD, we propose using a variant of Monte Carlo Tree Search (MCTS) that uses an offline pre-trained network to generate simulated roll-outs. MCTS is a powerful heuristic search algorithm often deployed for long-horizon decision making tasks.. MCTS, when used with a UCT heuristic, has properties like anytime convergence to the best action. This makes it well suited to be used as a long-horizon goal directed planner within large state spaces with a time budget.

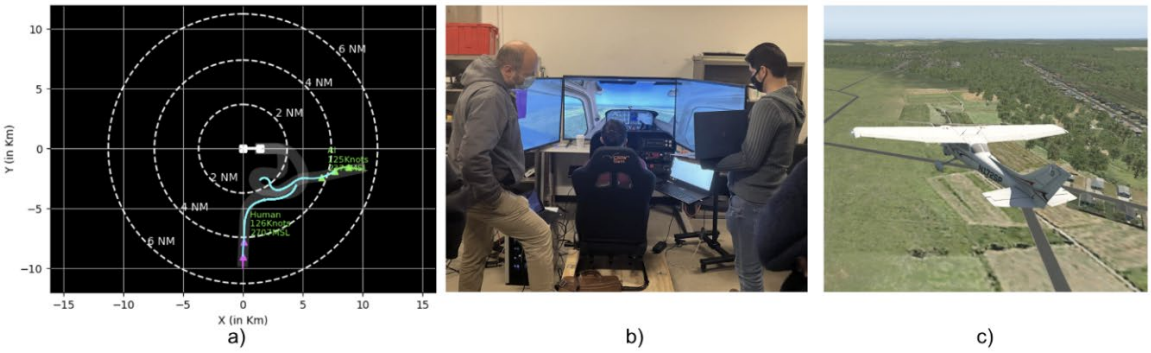
2.3. Automated speech recognition and production

Establishing clear communication between a human operator/pilot and an AI system in our target problem domain is critical. Understanding and decoding aviation-specific terminology, which differs from everyday speech constructions, is a big challenge. Other challenges such as radio background noise, incomplete instructions, and radio phraseology must also be addressed.

To this end, we propose a bi-directional communication mechanism for human-AI collaboration, focusing on clarity instead of naturalness to accomplish an acceptable performance level to produce a language covering the controlled vocabulary used in airspace operations. We employ learning approaches for the AI system to understand complex concepts, e.g., learning from demonstrations, and, to language understanding, develop a language generation system, which can summarize visually perceived information. This language generation capability can also clarify potential ambiguity when receiving commands from a human operator/pilot.

2.4. High-fidelity Simulator and Integration

A simulator that demonstrates a specific performance level and satisfies some standards may be applied to training, testing, and checking instead of doing the same during flight. The aviation-training community believes that a high level of fidelity is mandated to deliver the highest level of transfer of learning to the actual equipment. The fidelity of a flight simulator is essential to the pilots' performance in a real world. We are constantly working to deliver high-fidelity simulation-based assessment devices to enhance the expert pilot performance in real-world situations. Therefore, simulations have been implemented to help aviators refine their skills to become exceptional performers in the aviation industry.



The figure shows the high-fidelity simulator setup that enables Human-AI interaction. Figure a) shows a top-down view with one human agent (magenta) interacting and one AI agent (lime) while trying to land on the same runway. The most likely branch of the MCTS forward propagation tree for both the agents is shown in cyan. White lines show the reference trajectories. Figure b) shows the physical simulator setup with an immersive environment for the human pilot. Figure c) shows a screengrab of the visual rendering of the simulator using the X-Plane 11 flight simulator backend.

3. Technical Results

3.1. Intent Prediction

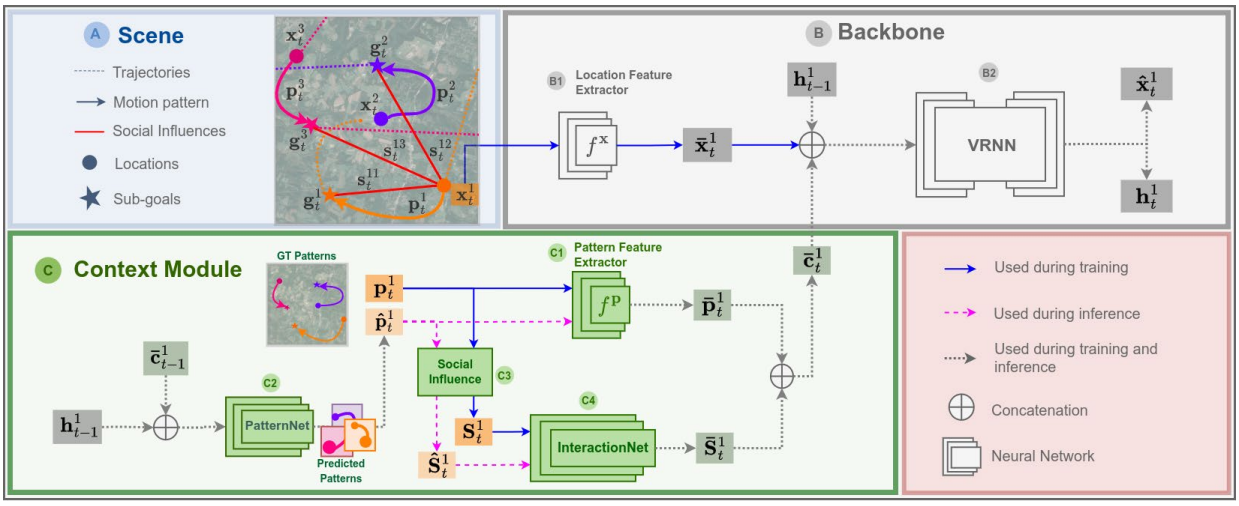


Figure shows the proposed intent prediction model.

For this project, we specifically focus on the settings where there exist navigation guidelines relatively strictly defined yet not clearly marked in their physical environments, e.g., in aerial navigation, although there are strict rules analogous to ground navigation, the air space does not display visible lanes. To address this challenge, we propose an approach, known here as Social-PatteRNN [1], where we aim to learn such contexts in the form of motion patterns and social influences extracted from the patterns. **This work builds onbackground IP "Social-PatteRNN: An algorithm for socially-aware trajectory prediction from motion patterns" (under submission).** We hypothesize that, within these domains, in the short term, agents tend to exhibit some motion patterns that reveal both their general directions of motion and their intermediate goals. We also believe that such patterns further explain general rules of motion, which may be explicit, e.g., sports rules or flying guidelines, or implicit, e.g., social etiquette, as well as, capture admissible motions. From these intuitions, we propose a data-driven approach to learn these patterns of motion and use them as a conditioning signal for predicting multimodal trajectories. Then, we use the learned patterns to extract sub-goal information which we aggregate to our model as the social context.

We also assessed the generalizability of the proposed approach across different domains: humans in crowds, humans in sports, and manned aircraft. Experimental results show that the proposed approach has consistent performance across these domains.

Table shows the results for various intent prediction algorithms compared to ours. Results are in terms of Average Displacement Error and Final Displacement Error ADE / FDE.

		Trajair (km)
1	A-VRNN [5]	0.64 / 1.31
2	DAG-Net [5]	0.78 / 1.53
3	TrajAirNet [3]	0.77 / 1.50
4	S-PEC [10]	0.96 / 2.05
5	Social-PatteRNN (Ours)	0.56 / 1.20
6	Social-PatteRNN-ATT (Ours)	**0.55** / **1.19**

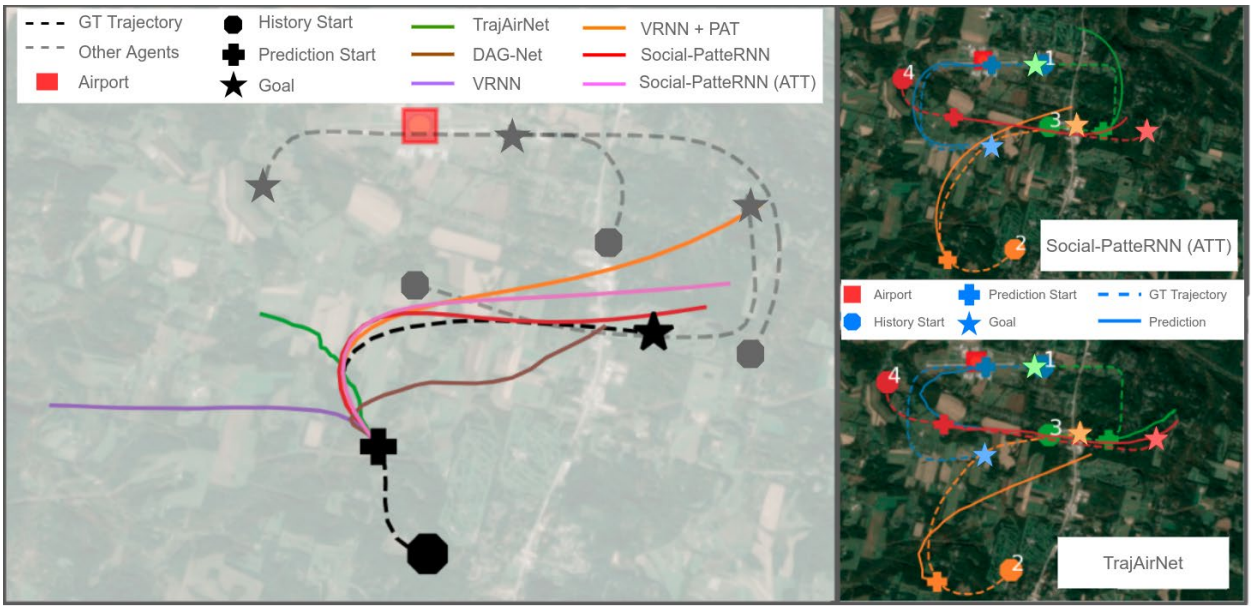


Figure shows various models' intent prediction results compared against our model, *Social-PatteRNN.* Dashed lines represent the ground truth trajectory, while solid lines represent the predicted intent. The proposed approach exhibits less jerkiness, producing more stable trajectories and closer to the ground truth, compared to other models.

3.3 Planning

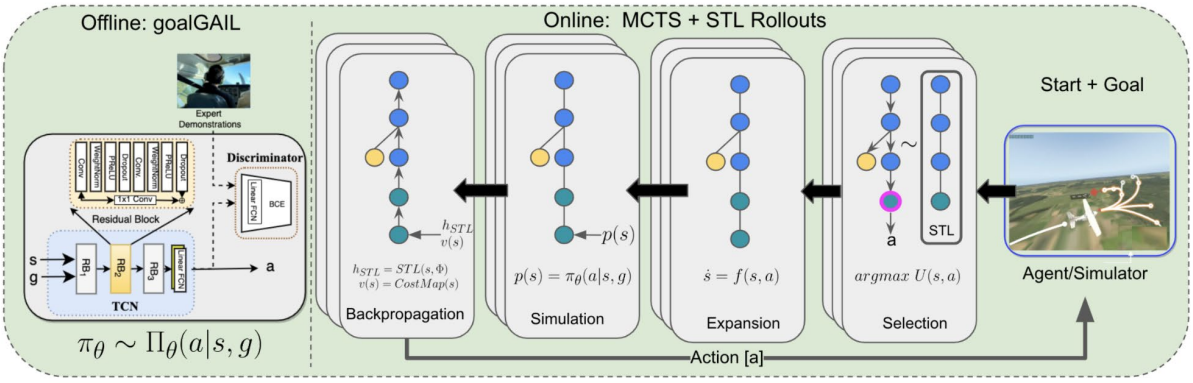


Figure shows the proposed planning algorithm

In this work, we present a novel decision-making method that uses MCTS to build a tree structure that guides the offline pre-trained LfD policies online with STL specifications. We achieve this by biasing the MCTS heuristic sampling towards branches with higher STL satisfactions. The primary insight is that while the LfD policy decides on the low level executions in line with the expert demonstrations, STL encourages the satisfaction of high-level objectives. This hierarchical approach provides a method to encode rules while allowing the agent to choose how to satisfy the constraints in a learning-enabled framework. The STL specifications thus provide a guide rail against the low-bandwidth noise in the expert demonstrations.

Given a continuous-space dynamic system of the form s˙ = f(s, a), we define a discrete-time Markov Decision Process without rewards, (MDP \R). Let M = (S, A, T, ρ0, G) where S is the set of states s ∈ S, a ∈A is the discrete set of actions or motion primitives that follow f(·), T : S ×A ⇒ S is the transition function, ρ0 ∈S is an initial state distribution, and G is the goal distribution. The task is to produce a policy π(θ) from a start location s0 ∈ ρ0 to goal location g ∈G that leads to a trajectory τ = (s0, a0, s1, a1, . . . , g). We also assume access to expert trajectories D = {(sj0, aj0, sj1, aj1, . . . , gj)} and high-level STL specification Φ that encodes any rules we expect the system to follow. An STL formula Φ can be built recursively from predicates using the following grammar

Φ := ⊤ | μc | ¬Φ | Φ ∧ Ψ | ♢[a,b]Φ | □[a,b]Φ | Φ1U [a,b]Φ2

where Φ1, Φ2 are STL formulas, ⊤ is the Boolean True, μc is a predicate of the form μ(s) > c, ¬ and ⋀ the Boolean negation and AND operators, respectively, and 0 ≤ a ≤ b < ∞denote time intervals. The temporal operators ♢, □ and U are called “eventually”, “always”, and “until” respectively. The quantitative semantics of a formula with respect to a signal □xt can be used to compute robustness values for the specifications used in our application.

Framework

Algorithm 1: Plan (θ,Φ)

$s \leftarrow Sample(\rho_0)$
$g \leftarrow Sample(G)$
while $s \neq g$ **and** *not maxSteps* **do**
  while $timeElapsed \leq planHorizon$ **do**
    $N(\cdot) \leftarrow MCTS(s_{0:t}, g, \theta, \Phi, 0)$
  end
  $a \leftarrow choice_{a'}(N(s, a'))$
  $s \leftarrow T(s, a)$
end

We first train a LfD policy by formulating the problem as finding a distribution of future actions conditioned on the past trajectories and the goal where tobs is the observation time horizon. The MCTS (see Algorithm 2) uses the policy to generate simulations. Each simulation starts from the root state and iteratively selects moves that maximize the STL modified UCT heuristic. For each state transition, we maintain a directed edge in the tree with an action value Q(s, a), prior probability P(s, a), STL heuristic H(s, a) and a visit count N(s, a). The total heuristic value is calculated as a weighted sum by controlling the degree of exploration and STL heuristic's weight. Starting with the initial state s(0), at each time step, we calculate the action to take, which maximizes U(s, a) (Line 12). If the next state already exists in the tree, we continue our simulation, else a new node is created in our tree, and we initialize its P(s, ·) = □pθ(s) from our policy πˆθ (Line 8). The expected reward v = vθ(s) can be provided by the user as a learned value function or as a cost-map (Line 9). The heuristic hST L is calculated using the STL specification (Line 14). We initialize Q(s, a), H(s, a) and N(s, a) to 0 for all a. We then propagate the cost v and the STL heuristic hST L back up the MCTS tree, updating all the Q(s, a) and H(s, a) values seen during the simulation, and start again from the root.

After running forward simulations of the MCTS, the N(s, a) values provide a good approximation for the optimal stochastic process from each state. Hence, the action we take is randomly sampled from a distribution of actions with probability proportional to N(s, a). We expand the search space until we exhaust the planning budget time plan horizon. After each action is selected, the MCTS tree is reinitialized from the actual trajectory followed by the agent. The planner is terminated when the goal is reached, or the maximum number of steps is reached, whichever comes first.

Algorithm 2: MCTS ($s_{0:t}$,g,θ,Φ,h_{STL})

if $s \in G$ **then**
  return s == g, h_{STL}
end
if $s \notin Tree$ **then**
  $Tree \leftarrow Tree \cup s$
  $Q(s,a) \leftarrow 0$
  $N(s,a) \leftarrow 0$
  $P(s,a) \leftarrow \hat{\pi}_\theta(s)$
  $v(s) \leftarrow CostMap(s)$
  return $v(s), h_{STL}$
else
  $a \leftarrow argmax_{a'} \Big[Q(s,a') + \frac{c_1 P(s,a')\sqrt{N(s)}}{1+N(s,a')} + c_2 H(s,a') \Big]$
  $s' \leftarrow T(s,a)$
  $h_{STL} \leftarrow STL(s' + s_{0:t}, \Phi)$
  $v, h_{STL} \leftarrow MCTS(s' + s_{0:t}, g, \theta, \Phi, h_{STL})$
  $N(s,a) \leftarrow N(s,a) + 1$
  $Q(s,a) \leftarrow \frac{N(s,a)Q(s,a)+v}{1+N(s,a)}$
  $H(s,a) \leftarrow \frac{N(s,a)*H(s,a)+h_{STL}}{1+N(s,a)}$
  return v, h_{STL}
end

Implementation Details

Consider a fixed-wing aircraft at time t, let st = (xt, yt, zt, χt) ∈ R3 ×SO(2) denote the position and where v is the aircraft's inertial speed, v2D is the speed in the 2-D plane, ϕ is the roll-angle, and vh is vertical speed. Finally, g is the acceleration due to gravity. We assume a zero wind condition.
The action space A is a fixed library of 30 motion primitives that discretizes each of the control inputs. We use the inertial velocities (v) 70 and 90 knots, the vertical velocities (vh) +500 ft/min and -500 ft/min and the bank angle (ϕ) discretization such that χ changes by 45◦ and 90◦ heading over the chosen time-horizon of 20 sec. The goal distribution G is 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. 3. The set G is represented as G = {N, NE, E, SE, S, SW, W, R08, R26} with each element representing the final region the aircraft is desired to reach as shown in Fig. 3. For simplicity we also set the start states equal to G.

The implementation details are split between online and offline components.

1) Dataset: We use the TrajAir dataset (5.2.1) , which is collected at the Pittsburgh-Butler Regional Airport (ICAO:KBTP). The dataset contains 111 days of transponder data that can be

used to extract expert demonstrations from pilots as they navigate the un-towered airspace. The dataset trajectories are smoothed using a B spline (basis-spline) approximate representation of order 2.

2) Offline LfD Policy Details: The LfD policy takes as input the past trajectories of the agent to predict its possible action distribution. While the method can use any LfD policy, we use a goal-conditioned generative adversarial imitation learning (GoalGAIL) method. The GoalGAIL is modified to use Temporal Convolutional Layers (TCNs) 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. We use TCNs as an alternative to using LSTMs for encoding the trajectories.

We break the trajectories in a scene into sequences of length tobs + tpred where tobs = 11sec and tpred = 20sec. In a given scene, the raw trajectory in absolute coordinates of the agent is encoded using the TCN layers as hobs. The agent's goal g □ G is encoded through an MLP layer, φ1, and is concatenated with the encoded trajectory vector. We measure how close the predicted trajectory is to the expert trajectory using a mean squared error (MSE) loss.
A discriminator Dψ is trained to distinguish expert transitions from policy transitions .The combination of these two loss functions is used to train the model.

In order to convert sˆ to aˆ, we match the generated trajectories from the control inputs in the library A using a weighted L2 Euclidean error distance over (x, y, z) points on the trajectory. For training, we use the AdamW optimizer with a learning rate of 3e − 3.

3) Signal Temporal Logic Specifications: We evaluate the performance of our agent based on reaching the goal while adhering to the airport traffic pattern. The goal objective, as well as traffic pattern compliance, are both encoded using STL specifications. We use the three stages for landing pattern. Φ1, Φ2, and Φ3 represent the STL formulas encoding occupancy of regions corresponding to the downward, base, and final stages, respectively. The landing STL specification becomes:

$$\Phi L = \Diamond(\Phi 1 \wedge (\Diamond (\Phi 2 \wedge (\Diamond \Box \Phi 3))))$$

$\Diamond(\Phi)$ can be interpreted as "Eventually" being in a region represented by Φ. The nested $\Diamond$ operators encode a sequential visit of regions represented by Φ1, Φ2, and Φ3 . Similarly, the takeoff STL specification is defined based on the goal regions reached by the aircraft. By defining reaching a goal region g □ G by an STL formula Φ4, we get the takeoff specification:

$$\Phi T = \Diamond(\Phi 4)$$

The robustness values of the STL specifications are eval uated on the state trajectory traces generated by the search tree. The first element of the traces is the tree root node, and the last element of the trace is the node whose value is being computed.

4) Online Monte Carlo Tree Search : The MCTS is implemented as a recursive function where each iteration ends with a new leaf that corresponds to an action in the trajectory library. The implementation uses a normalized costmap v(s) that is built by counting the frequencies of the aircraft in the TrajAir dataset at particular states s.

Evaluations

Evaluation of the proposed approach is performed using a custom simulator that follows the dynamics defined in Eq. 4. The network implementations are in PyTorch. We use the recently released rtamt package, an online monitoring library for calculating STL robustness values. To showcase real-time online evaluations, simulations are performed on an Intel NUC computer with Intel® Core™ i7- 8559U CPU @ 2.70GHz × 8. The complete implementation details and parameter details are in the open-sourced code base3and the associated Readme.

A. Qualitative results

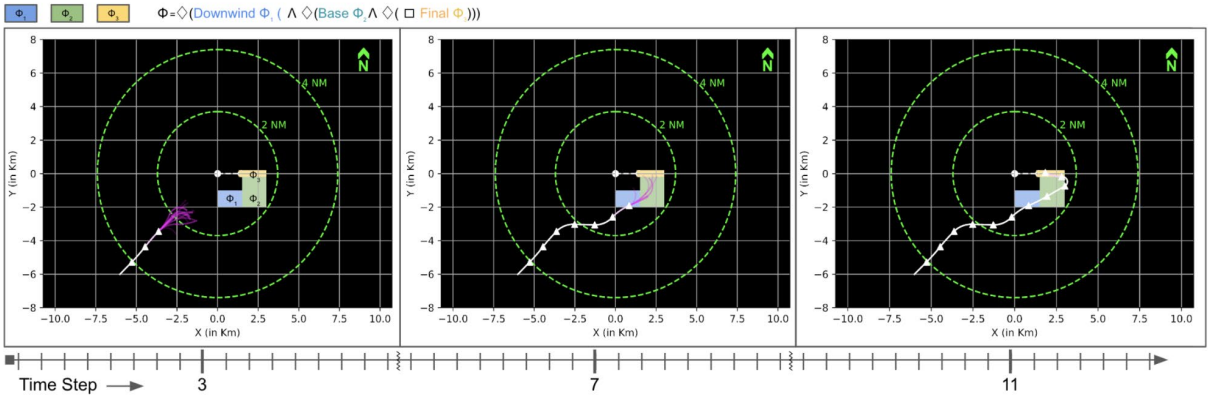


Figure shows an example scenario where the aircraft starts from the South-West and is tasked with landing at R26 while following the standard FAA traffic pattern. The rules of the traffic pattern are encoded as STL specifications. The white marked line shows the aircraft's position at each step. At every step, the MCTS replans, and the resulting tree is shown in magenta. The STL sub-specifications are shown as rectangles. As can be seen, the aircraft manages to reach the runway while satisfying the specifications. The size of the search tree is a function of available planHorizon.

B. Comparative results

In order to perform quantitative studies, we compare the performance of the proposed algorithm with vanilla LfD policies. We uniformly sample 100 start-goal pairs randomly from ρ0, G. G is truncated to N, S, E, W, R where R rep resents both R08 and R26 to condense the results. We then provide these to Algorithm 1, which plans for the agent.

Based on the selected start and goal pair, a relevant STL specification is chosen. Each episode ends when the agent reaches the goal or if the maximum steps are exceeded. In addition to the goalGAIL policy, we also train a pure Behavior Cloning (BC) policy. The BC policy uses a TCN to encode the history and outputs an action without a goal vector or a discriminator.

Comparisons were carried out with both these policies integrated into the MCTS framework with ablation on the STL heuristic to show the impact of the STL specification.
We define our evaluation metrics as follows:
• Success Rate: Fraction of episodes that were successful in reaching their goal locations.
• STL Score: Average of the normalized fraction of the STL robustness value satisfied over all the episodes. A higher value indicates better satisfaction.
Table I shows the quantitative results. Our proposed algorithm performs significantly better than the baselines for all start-goal pairs. We get an almost perfect success rate in the aircraft takeoff scenarios. The aircraft landing cases are more challenging due to following the specific landing patterns when incoming from different sides of the runway, which is reflected in the success rates shown. Additionally, we observe the baseline BC performs similarly to GAIL on the success metric but not on the STL robustness. This shows that while BC policies are comparable in reaching the goals, the transient performance of GoalGAIL is better. Incorporating STL improves the robustness values for both LfD policies.

Algorithm	**Takeoff Specification Φ_T**				**Landing Specification Φ_L**				**Total**
	N	S	E	W	N	S	E	W	
BC + MCTS	0.2 / 0.2	0.0 / 0.1	0.3 / 0.1	0.3 / 0.1	0.0 / 0.0	0.1 / 0.4	0.1 / 0.4	0.3 / 0.4	0.1 / 0.2
GoalGAIL + MCTS	0.0 / 0.3	0.1 / 0.4	0.0 / 0.3	0.0 / 0.4	0.3 / 0.6	0.1 / 0.2	0.3 / 0.7	0.1 / 0.5	0.1 / 0.4
BC + MCTS + STL	1.0 / 0.9	1.0 / 0.9	0.6 / 0.3	0.7 / 0.4	0.2/ 0.5	0.2 / 0.5	0.2 / 0.5	0.2 / 0.5	0.5 / 0.6
GoalGAIL + MCTS+STL	**1.0 / 0.9**	**1.0 / 0.9**	**0.8 / 0.7**	**1.0 / 0.9**	**0.7 / 0.9**	**0.8 / 0.9**	**0.3 / 0.8**	**0.5 / 0.9**	**0.7 / 0.8**

TABLE I: Table shows the quantitative results with randomly sampled start and goal states for two LfD policies with ablation studies with the STL heuristic. Results show the Success Rate ↑ / STL Score ↑ for two vanilla LfD policies and their ablations with the STL heuristic. Results show both Landing $X \Rightarrow R$ and Takeoff $R \Rightarrow X$ scenarios for each of the cardinal directions X.

3.4 Speech Recognition

We introduce two off-the-shelf modules to realize the automatic speech recognition and speech generation functionality. For speech-to-text, we use a Speech to Text Transformer model trained on the Librispeech corpus. This model achieves a Word Error Rate of 41.3% and a Character Error Rate of 12.9% on our in-house dataset. To further improve the model's prediction accuracy, we use string matching to replace the predicted words or phrases to those more probable in the air traffic control scenario (e.g., "one way two six" → "runway two six").

Additionally, to investigate the capability of the speech-to-text model even further, we evaluate the model without further fine-tuning on a subset of the LDC Air Traffic Control Corpus, which is a dataset collected from actual pilot utterance recordings. Speech collected in the Air Traffic Control domain typically contains excessive background noise coming from aircraft, and speakers tend to talk much quicker than the Librispeech and Commonvoice corpora general speech-to-text models are trained on. As expected, in this dataset, the model achieves a Word Error Rate of 91% and a Character Error Rate of 60.6%. The relatively high error rate can be explained by the distinct properties of speech recorded in Air Traffic Control domains, and expect that in-domain fine-tuning can boost the recognition accuracy in relevant datasets. For text-to-speech, we employ the recently proposed FastSpeech 2 model and apply slot-filling to a pre-defined set of template utterances.

4. Significant Hardware or Software Developed:

4.1 High Fidelity Simulator

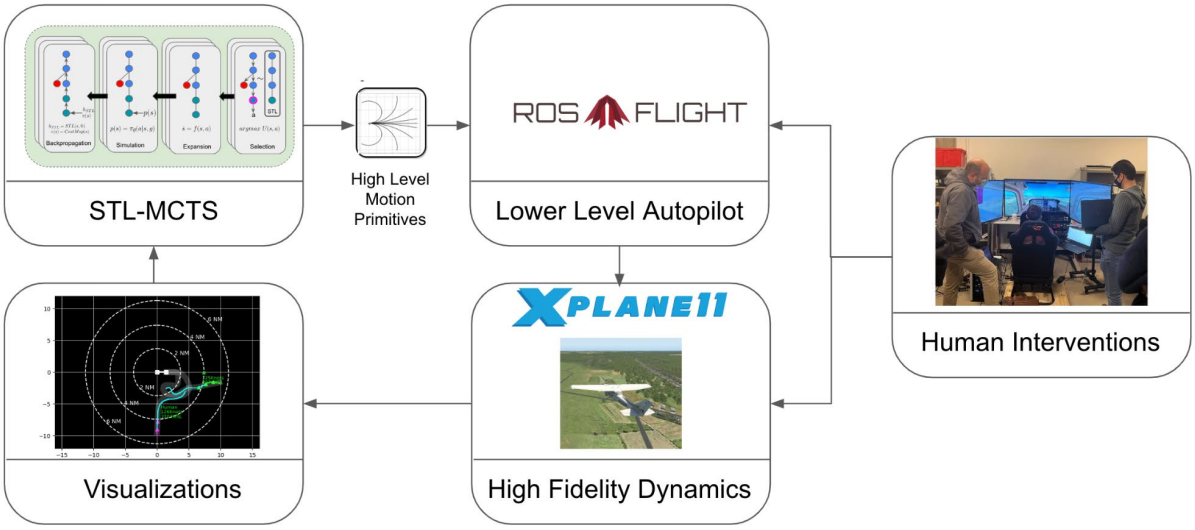


Figure. XPlaneROS Architecture

XPlaneROS (5.2.2) integrates a high-fidelity simulator with a state-of-the-art autopilot. The complete system enables the use of high-level or lower-level commands to control a general aviation aircraft in realistic world scenarios anywhere in the world. We chose X-Plane 11 as our simulator because of its open API and realistic aircraft models and visuals. For the lower-level control, we've integrated ROSplane as the autopilot. ROSplane is a control stack for fixed-wing aircraft developed by the BYU MAGICC Lab.

XPlaneROS interfaces with XPlane 11 using NASA's XPlaneConnect. With XPlaneROS, the information from XPlane is published over ROS topics. The ROSplane integration then uses this information to generate actuator commands for ailerons, rudder, elevator, and throttle based on higher-level input to the system. These actuator commands are then sent to XPlane through XPlaneConnect.

ROSplane uses a cascaded control structure and has the ability to follow waypoints with Dubin's Paths. XPlaneROS provides additional capabilities to follow a select set of motion primitives.

There have also been some extensions to ROSplane like employing a proper takeoff, additional control loops for vertical velocity rates and a rudimentary autonomous landing sequence.

5. Publications this Period:

5.1 Papers:

[1] Ingrid Navarro & Jean Oh. *Social-PatteRNN: Socially-Aware Trajectory Prediction Guided by Motion Patterns.* In proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022 (to appear) [ArXiv] .

[2] Aloor, J. J., Patrikar, J., Kapoor, P., Oh, J., & Scherer, S. (2022). Follow The Rules: Online Signal Temporal Logic Tree Search for Guided Imitation Learning in Stochastic Domains. arXiv preprint arXiv:2209.13737.[ArXiv]

[3] Ghosh, S., Patrikar, J., Moon, B., & Hamidi, M. M. (2022). AirTrack: Onboard Deep Learning Framework for Long-Range Aircraft Detection and Tracking. arXiv preprint arXiv:2209.12849.[ArXiv]

[4] Patrikar, J., Moon, B., Oh, J., & Scherer, S. (2022, May). Predicting like a pilot: Dataset and method to predict socially-aware aircraft trajectories in non-towered terminal airspace. In 2022 International Conference on Robotics and Automation (ICRA) (pp. 2525-2531). IEEE.

[5] Patrikar, J., Dantas, J. P., Ghosh, S., Kapoor, P., Higgins, I., Aloor, J. J., ... & Scherer, S. Challenges in Close-Proximity Safe and Seamless Operation of Manned and Unmanned Aircraft in Shared Airspace. Workshop, In 2022 International Conference on Robotics and Automation (ICRA). IEEE.

5.2 Dataset and Software:

[1] Patrikar, Jay; Moon, Brady; Ghosh, Sourish; Oh, Jean; Scherer, Sebastian (2021): TrajAir: A General Aviation Trajectory Dataset. Carnegie Mellon University. Dataset. https://doi.org/10.1184/R1/14866251.v1

[2] Baijal, Rohan; Patrikar, Jay; Moon, Brady; Scherer, Sebastian; Oh, Jean (2021): XPlaneROS : ROS Wrapper for Autonomous Fixed Wing Applications. Carnegie Mellon University. Software. https://doi.org/10.1184/R1/16589924.v1

5.3 Videos:

[1] STL-MCTS https://www.youtube.com/watch?v=fiFCwc57MQs

[2] AirTrack https://www.youtube.com/watch?v=H3lL_Wjxjpw&t=1s

[3] TrajAirNet https://www.youtube.com/watch?v=elAQXrxB2gw&t=15s

[4] Demo https://www.youtube.com/watch?v=iU_MyMwuE8E

5.4 Blogs:

[1] How do you train AI Pilots? [Link]

[2] XPlaneROS : ROS Wrapper for Autonomous Fixed Wing Applications [Link]

[3] Long-range Aircraft Detection and Tracking [Link]

[4] TrajAir: A General Aviation Trajectory Dataset [Link]

5.5 Media Coverage:

[1] Move over, autopilot: This AI can avoid other planes, Popular Science, Sept 2022

[2] CMU's AI pilot lands in the news, Practical AI – Episode #189, Sept 2022

[3] Researchers Develop AI Pilot for Navigating Crowded Airspace, Aviation Today, Aug 2022

[4] AI Pilot Can Navigate Crowded Airspace, CMU School Of Computer Science, Aug 2022