Dom, cars don't fly!—Or do they? In-Air Vehicle Maneuver for High-Speed Off-Road Navigation

Anuj Pokhrel, Aniket Datar, and Xuesu Xiao

***Abstract*— When pushing the speed limit for aggressive off-road navigation on uneven terrain, it is inevitable that vehicles may become airborne from time to time. During time-sensitive tasks, being able to fly over challenging terrain can also save time, instead of cautiously circumventing or slowly negotiating through. However, most off-road autonomy systems operate under the assumption that the vehicles are always on the ground and therefore limit operational speed. In this paper, we present a novel approach for in-air vehicle maneuver during high-speed off-road navigation. Based on a hybrid forward kinodynamic model using both physics principles and machine learning, our fixed-horizon, sampling-based motion planner ensures accurate vehicle landing poses and their derivatives within a short airborne time window using vehicle throttle and steering commands. We test our approach in extensive in-air experiments both indoors and outdoors, compare it against an error-driven control method, and demonstrate that precise and timely in-air vehicle maneuver is possible through existing ground vehicle controls.**

I. Introduction

Off-road navigation presents various challenges that sharply contrast those encountered in on-road or indoor scenarios. In unstructured off-road environments, robots must detect and avoid obstacles, evaluate the traversability of varied terrain, and continuously adapt to complex vehicle-terrain interactions. Tackling all these challenges is essential to prevent terminal states that can jeopardize the mission and damage the robot, such as vehicle rollover and getting stuck.

One particular challenge of off-road navigation is addressing terrain unevenness. Current state-of-the-art approaches typically rely on perception based traversability estimation to avoid uneven terrain or enforce slow speed to prevent catastrophic failures. By circumventing or slowing down on uneven terrain, these approaches have yet to push the limits of off-road vehicles' capabilities to quickly traverse through challenging off-road environments.

During time-sensitive off-road missions where achieving the physically feasible maximum speed is necessary, interacting with uneven terrain will cause robots to become airborne from time to time. Additionally, leveraging appropriate terrain structure to take off the vehicle can also efficiently circumvent difficult terrain (in the z direction instead of on the x-y plane) without compromising path length or traversal time. However, after air time, deviations from an appropriate landing pose, depending on the receiving terrain geometry (e.g., nose- or tail-down pitch and sideways roll on a flat terrain, or vice versa), can significantly impact landing safety (see examples in Fig. 1). A critical yet often overlooked aspect of high-speed off-road navigation is the maneuver of a robot during these aerial phases. Despite limited work on planning before losing ground contact, to the best of our knowledge, no prior work has investigated ground vehicle in-air maneuver to facilitate safe landing.

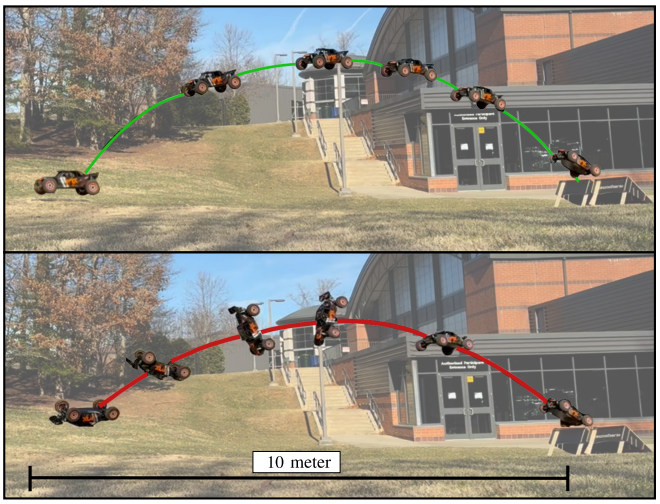

Fig. 1: In-air vehicle maneuvers are critical in ensuring safe vehicle landing during high-speed off-road navigation. Top: Precise and timely maneuvers prepare the robot to land with minimal impact. Bottom: Improper maneuvers cause the robot to land on its back, terminating mission execution and risking vehicle damage.

To this end, we ask the question *how can we control the in-air attitude of a ground robot in a short amount of airborne time only with existing vehicle controls?* To address this question, we present a novel in-air vehicle attitude planning and control approach by re-purposing existing throttle and steering actions. Our novel forward kinodynamic model uses the inertial and gyroscopic effects of the spinning wheels to derive the robot's angular accelerations. These accelerations are used in a Newtonian physics model to calculate the robot's future states. Leveraging our model, we also develop a fixed-horizon, sampling-based motion planner specifically designed for quick in-air goal convergence to prepare the vehicle orientation for proper landing. Our experiment results on a gimbal platform demonstrate the ability of our method to perform precise and timely in-air maneuvers, showing a significant improvement compared to a traditional error-

All authors are with the Department of Computer Science, George Mason University {apokhre, adatar, xiao}@gmu.edu

An extended version of this manuscript is currently under review for publication.

driven approach. Furthermore, our outdoor demonstration shows our method's ability to generalize under real-world disturbances like air resistance and uneven weight distribution. Our contributions can be summarized as:

- A hybrid dynamics model combining physics principles and data-driven learning for in-air vehicle maneuver driven only by the vehicle throttle and steering actions;
- A fixed-horizon, sampling-based motion planner that converges to an appropriate goal state for safe landing;
- A set of real-world robot experiments on a gimbal platform and outdoors to demonstrate the effectiveness of our kinodynamic model along with our motion planner.

II. Approach

In this section, we first define the in-air vehicle maneuver problem with airborne kinodynamic modeling. Then, we discuss the physics principles of why and how in-air vehicle maneuver is possible through existing vehicle controls, i.e., throttle and steering, based on the inertial and gyroscopic effects of the spinning wheels. Motivated by the difficulty in accurately and analytically modeling changing quantities purely based on physics, we present PHysics and Learning based model for In-air vehicle maneuver (PHLI, pronounced as "fly"), a precise and efficient hybrid modeling approach. Finally, we introduce our Dom Planner (*named after Dominic Toretto, who makes cars fly. So does our planner.*) for vehicle trajectory planning to reorient the robot from its current configuration to the goal precisely at when a limited airborne time window expires.

A. Problem Formulation

We first formulate a discrete-time forward kinodynamic modeling problem based on a bicycle model, where the subsequent state, $\mathbf{s}_{t+1} \in \mathcal{S}$, is derived from the current state, $\mathbf{s}_t \in \mathcal{S}$, and current action, $\mathbf{a}_t \in \mathcal{A}$, with $\mathcal{S}$ and $\mathcal{A}$ as the state and action spaces respectively. Considering that during aerial phases a conventional ground vehicle does not have control over the three translational components in $\mathbb{SE}(3)$ (which is determined solely by gravity), we include in the vehicle state $\mathbf{s}_t$ a tuple of the three angles in $\mathbb{SO}(3)$, i.e., roll, pitch, and yaw, together with their corresponding angular velocities. We also include rotation per minute of the wheels, rpm, and the front steering angle, ψ, in the state space, i.e., $\mathcal{S} \subset \mathbb{R}^8$. The state at time t is denoted as

$$\mathbf{s}_t = (\text{roll}_t, \dot{\text{roll}}_t, \text{pitch}_t, \dot{\text{pitch}}_t, \text{yaw}_t, \dot{\text{yaw}}_t, \text{rpm}_t, \psi_t) \in \mathcal{S}.$$

The vehicle action, $\mathbf{a}_t$, comprises the rate of change in wheel rpm (throttle) and in steering angle ψ, which are common controls available for ground robots. The action in the action space, $\mathcal{A} \subset \mathbb{R}^2$, is thus defined as a two-dimensional tuple:

$$\mathbf{a}_t = (\dot{\text{rpm}}_t, \dot{\psi}_t) \in \mathcal{A}.$$

A forward kinodynamics model is defined as a function, $f_\theta : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, parametrized by θ, such that:

$$\mathbf{s}_{t+1} = f_\theta(\mathbf{s}_t, \mathbf{a}_t). \tag{1}$$

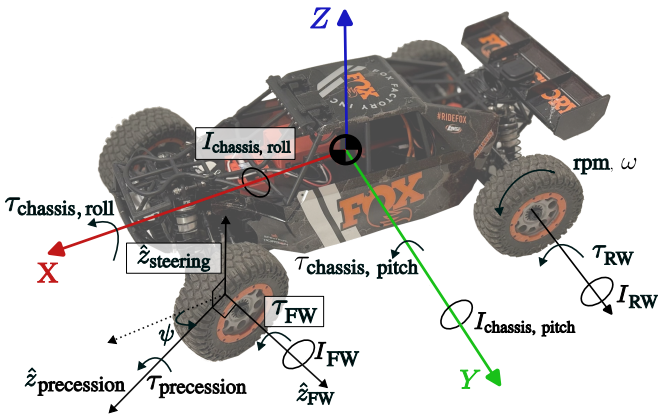

Fig. 2: Simplified Bicycle Model with Torques Acting on the Wheels and Chassis due to Wheel Acceleration and Steering.

After an aerial phase, the vehicle should land on the ground with an appropriate configuration. Unlike traditional navigation planning problems, in which the goal is to achieve a desired state with certain notion of minimal cost (shortest path, lowest energy, etc.), for our in-air maneuver problem the goal is to achieve a goal configuration precisely when a small airborne time window expires, i.e., the landing time, T. Therefore, the goal is to achieve:

$$\mathbf{s}_T^g = (\text{roll}_T^g, \dot{\text{roll}}_T^g, \text{pitch}_T^g, \dot{\text{pitch}}_T^g, \text{yaw}_T^g, \dot{\text{yaw}}_T^g, \text{rpm}_T^g, \psi_T^g). \tag{2}$$

Notice that $\mathbf{s}_T^g$ must be achieved precisely at T, not later. If it is achieved at $t < T$, the vehicle needs to assure it maintains the same state at the landing time T. For example, when the landing region is horizontal, roll_T^g and pitch_T^g become zero, while $\dot{\text{roll}}_T^g$ and $\dot{\text{pitch}}_T^g$ should be as close to zero as possible to minimize impact. yaw_T^g and $\dot{\text{yaw}}_T^g$ are often less crucial since they do not cause significant impact during landing. rpm_T^g mostly needs be positive to avoid flipping over the vehicle upon ground contact with forward momentum, and ψ_T^g depends on what the immediate ground maneuver necessary for the vehicle to execute right after landing is, e.g., to quickly swerve to avoid an upcoming obstacle.

Therefore, the in-air vehicle maneuver problem for safe landing can be formulated as

$$\mathbf{a}_0^*, ..., \mathbf{a}_{T-1}^* = \underset{\mathbf{a}_0,...,\mathbf{a}_{T-1}}{\arg\min} \sum_{t=0}^{T} c(\mathbf{s}_t, \mathbf{s}_T^g),$$
$$\text{s. t.} \quad \mathbf{s}_{t+1} = f_\theta(\mathbf{s}_t, \mathbf{a}_t), \quad \mathbf{s}_0 \text{ is given}, \quad \text{and } \mathbf{s}_T = \mathbf{s}_T^g. \tag{3}$$

c is a state-wise cost function to be minimized, subject to the constraints by the forward kinodynamics, given initial state, and achieving the goal state at the end of the aerial phase T. Notice the difference of Problem (3) to traditional receding-horizon planning problems, where T is a receding horizon which gradually moves towards the goal. Here T is the end of the entire horizon of the problem, i.e., the landing point.

B. Physics Principles

In-air vehicle maneuvers are controlled using wheel acceleration ($\dot{\text{rpm}}$), steering angle (ψ), and steering rate ($\dot{\psi}$). Two key effects enable in-air maneuvers, the inertial and

gyroscopic effects of the wheels, which are leveraged to adjust the vehicle's roll and pitch orientations.

Inertial Effect: While a vehicle is in the air, accelerating the wheels applies reaction torques to the chassis due to the conservation of the angular momentum. For a simplified bicycle model, these torques cause roll and pitch rotations. Roll acceleration ($\ddot{\text{roll}}$) depends on $\dot{\text{rpm}}$ and $\sin(\psi)$, while pitch acceleration ($\ddot{\text{pitch}}$) depends on $\dot{\text{rpm}}$ and $\cos(\psi)$.

Gyroscopic Effect: Steering the spinning front wheels induces precession, generating torques which contributes to $\ddot{\text{roll}}$ (proportional to rpm, $\dot{\psi}$, and $\cos(\psi)$) and $\ddot{\text{pitch}}$ (proportional to rpm, $\dot{\psi}$, and $\sin(\psi)$). Together, these effects allow control over roll and pitch. However, yaw control is limited, though initial conditions and external factors (e.g., air resistance) can influence yaw behavior.

C. PHLI

Calculating $\ddot{\text{roll}}$ and $\ddot{\text{pitch}}$ requires accurate moment of inertia measurements for the vehicle chassis and for the wheels. These measurements are difficult to analytically derive or precisely measure due to changes in the center of gravity and weight distribution by steering and as a result of suspension movement. Furthermore, expansion of the tires at high speeds due to centrifugal forces changes the moment of inertia of the wheels as well. Hence, we decompose the forward kinodynamics model $\mathbf{s}_{t+1} = f_\theta(\mathbf{s}_t, \mathbf{a}_t)$ into two parts, g_ϕ and h_ξ, i.e., $\mathbf{s}_{t+1} = h_\xi(g_\phi(\mathbf{s}_t, \mathbf{a}_t), \mathbf{s}_t, \mathbf{a}_t)$, and then employ a data-driven and a physics-based approach to derive g_ϕ and h_ξ respectively.

g_ϕ takes the state and action as input and derives the resultant acceleration of roll, pitch, and yaw

$$\ddot{\text{roll}}_t, \ddot{\text{pitch}}_t, \ddot{\text{yaw}}_t = g_\phi(\mathbf{s}_t, \mathbf{a}_t), \tag{4}$$

where model parameters ϕ are learned in a data-driven manner by minimizing a supervised loss. The ground truth acceleration values are derived from the recorded Inertial Measurement Unit (IMU) data via differentiation over time.

Once g_ϕ is trained, its output angular accelerations are fed into h_ξ, along with the state $\mathbf{s}_t$ and action $\mathbf{a}_t$. For $\mathbf{s}_{t+1} = h_\xi((\ddot{\text{roll}}_t, \ddot{\text{pitch}}_t, \ddot{\text{yaw}}_t), \mathbf{s}_t, \mathbf{a}_t)$, we have

$$\begin{aligned}
\dot{\text{roll}}_{t+1} &= \dot{\text{roll}}_t + \ddot{\text{roll}}_t \cdot dt, \\
\text{roll}_{t+1} &= \text{roll}_t + \dot{\text{roll}}_t \cdot dt + \frac{1}{2}\ddot{\text{roll}}_t \cdot dt^2, \\
\text{rpm}_{t+1} &= \text{rpm}_t + \dot{\text{rpm}}_t \cdot dt, \\
\psi_{t+1} &= \psi_t + \dot{\psi}_t \cdot dt,
\end{aligned}$$

with analogous update equations for pitch and yaw. The only parameter for h_ξ is the integration interval dt. This completes PHLI's transition to next state $\mathbf{s}_{t+1}$ given current state $\mathbf{s}_t$ and action $\mathbf{a}_t$ through g_ϕ and h_ξ.

D. Dom Planner

Given the short airtime, we adopt a fixed-horizon, instead of receding-horizon, planning approach, where the horizon is determined by the discretized time remaining until landing. The planner samples action sequences over this fixed horizon and uses the learned PHLI to rollout the corresponding state trajectories. Each trajectory is then evaluated using a cost function. The planner selects the trajectory with the minimal cost, executes its first action, and replans with the updated (and reduced) remaining time. The next planning cycle samples around the best action from the last cycle.

One critical consideration of Dom Planner is that, in addition to general physical actuation limits (e.g., limited motor current and torque causing that the actions can never exceed certain thresholds), the under-actuated vehicle controls of $\dot{\text{rpm}}$ and $\dot{\psi}$ are further constrained due to state-dependent actuation limits:

$$\begin{aligned}
\text{rpm}_{\text{min}} - \text{rpm}_t \le \dot{\text{rpm}}_t^{\text{feasible}} \le \text{rpm}_{\text{max}} - \text{rpm}_t, \\
\psi_{\text{min}} - \psi_t \le \dot{\psi}_t^{\text{feasible}} \le \psi_{\text{max}} - \psi_t,
\end{aligned}$$

which means if rpm_t (ψ_t) is at the minimal value, $\dot{\text{rpm}}_t^{\text{feasible}}$ ($\dot{\psi}_t^{\text{feasible}}$) cannot be negative, and if $\dot{\text{rpm}}_t$ (ψ_t) is at the maximal value, $\dot{\text{rpm}}_t^{\text{feasible}}$ ($\dot{\psi}_t^{\text{feasible}}$) cannot be positive. This limited range of feasible $\dot{\text{rpm}}$ and $\dot{\psi}$ reduces possible in-air maneuver options and makes simple error-driven controllers, like PID controllers, inappropriate to enable complex in-air vehicle maneuvers.

To enable goal convergence within a short aerial phase, the state-wise cost function defined in the fixed-horizon, instead of receding-horizon, problem in Eqn. (3) is formulated as:

$$c(\mathbf{s}_t, \mathbf{s}_T^g) = \sum_{i=1}^{K} w_i(t) \cdot c_i(\mathbf{s}_t, \mathbf{s}_T^g), \tag{5}$$

which is a combination of the costs of K state dimensions (in our case, $K = 8$). A key difference from conventional state-wise cost function is that each state dimension is dynamically weighed by a weight term $w_i(t)$ as a function of time t. For example, for initial time steps, it is more important to quickly align the vehicle angles to prepare for a safe landing pose, while the focus will gradually shift to the angular velocities, wheel rpm, and steering components later on to ensure landing with minimal impact on the vehicle.

Additionally, based on the sampled trajectory with the lowest cost, Dom Planner checks its feasibility of reaching the goal state within the time remaining till landing by computing the difference between the final state $\mathbf{s}_T$ and goal state $\mathbf{s}_T^g$. This feasibility check allows the system to either choose a viable trajectory or, if necessary, select an alternate goal to mitigate landing impact.

III. Implementations

We present the implementation details of our PHLI and Dom Planner onboard a 1/5th-scale vehicle.

A. PHLI *Implementations*

The learnable function of PHLI, g_ϕ, is a 4-layer multi-layer perceptron that produces a three-dimensional output of predicted angular accelerations, $(\ddot{\text{roll}}_{t+1}, \ddot{\text{pitch}}_{t+1}, \ddot{\text{yaw}}_{t+1})$. g_ϕ is trained to minimize the difference between the predicted angular accelerations and the derived ground truth from IMU. The outputs of g_ϕ is fed into h_ξ to calculate the next state of the vehicle $\mathbf{s}_{t+1}$.

Algorithm 1 Dom Planner

1: **Parameters:** integration interval dt (0.2), sample count N (4000), actuation limits λ, sampling range $\sigma_{\dot{\text{rpm}}}$ (2000) and $\sigma_{\dot{\psi}}(0.2)$, and PHLI f_θ
2: **Input:** Time till landing T, initial state $\mathbf{s}_0$, goal state $\mathbf{s}_T^g$, and last best action $(\dot{\text{rpm}}_{\text{best}}, \dot{\psi}_{\text{best}})$
3: Compute fixed landing horizon: $H = \frac{T}{dt}$
4: $\dot{\text{rpm}}_l, \dot{\text{rpm}}_h = \dot{\text{rpm}}_{\text{best}} \pm \sigma_{\dot{\text{rpm}}}$ ▷ *sampling range for* $\dot{\text{rpm}}$
5: $\dot{\psi}_l, \dot{\psi}_h = \dot{\psi}_{\text{best}} \pm \sigma_{\dot{\psi}}$ ▷ *sampling range for* $\dot{\psi}$
6: **for** $(\dot{\text{rpm}}_i,\ \dot{\psi}_i)$, $i \in [1, N]$, sampled from range $[\dot{\text{rpm}}_l, \dot{\text{rpm}}_h]$ and $[\dot{\psi}_l, \dot{\psi}_h]$ **do**
7: $T_i = \{\mathbf{s}_0\}$
8: $U_i = (\dot{\text{rpm}}_i, \dot{\psi}_i)$
9: **for** $t \in [0, H-1]$ **do**
10: $\mathbf{s}_t$ = CLAMP_STATE($\mathbf{s}_t$, λ)
11: $\mathbf{a}_t$ = CLAMP_ACTION($\dot{\text{rpm}}_i, \dot{\psi}_i, \mathbf{s}_t$, λ)
12: $\mathbf{s}_{t+1} = f_\theta(\mathbf{s}_t, \mathbf{a}_t)$
13: T_i.add($\mathbf{s}_{t+1}$)
14: **end for**
15: C_i = CALCULATE_COST($T_i, \mathbf{s}_T^g$)
16: **end for**
17: $T_{\text{best}} = T_{\arg\min_i(C_i)}$ ▷ *minimal-cost trajectory*
18: $U_{\text{best}} = U_{\arg\min_i(C_i)}$ ▷ *best action of this cycle*
19: Return T_{best}, U_{best}

B. Dom Planner Implementation

Dom Planner's implementation is shown in Algorithm 1. Line 1 first defines Dom Planner's parameters, including integration interval, sample count, actuation limits, sampling range, and PHLI. Actuation limits include rpm rate limit $\dot{\text{rpm}}_{\text{min}}$ and $\dot{\text{rpm}}_{\text{max}}$ ($\pm$5000), rpm limit rpm_{min} (0) and rpm_{max} (1980), steering rate limit $\dot{\psi}_{\text{min}}$ and $\dot{\psi}_{\text{max}}$ ($\pm$6.5 rad/s), and steering limit ψ_{min} and ψ_{max} ($\pm$0.65 rad). Line 2 specifies the algorithm input. Given gravity, time till landing is determined by the initial take-off velocity and terrain geometry. Initial state is the robot state at take-off. Goal state can be defined by the geometry of the receiving terrain. Last best action is the action executed in the last planning cycle. The planning cycle is initiated as soon as the vehicle becomes airborne. The planner calculates the fixed landing horizon $H = \frac{T}{dt}$ by discretizing the time till landing with the interval dt in line 3. For each planning cycle, the algorithm uniformly samples 4000 candidate input pairs $(\dot{\text{rpm}}_i,\ \dot{\psi}_i)$ within the ranges determined by $\sigma_{\dot{\text{rpm}}}$ and $\sigma_{\dot{\psi}}$ centered around the last best action $(\dot{\text{rpm}}_{\text{best}}, \dot{\psi}_{\text{best}})$ in lines 4 and 5. For each sample (line 6), the planner rolls out a trajectory over the horizon H (line 9) using our PHLI f_θ to compute the state transition dynamics (line 12). At every time step, the functions CLAMP_STATE and CLAMP_ACTION enforce the state and action limits (lines 10-11). After rolling out the full trajectory, CALCULATE_COST evaluates each trajectory's cost according to Eqn. (5) (line 15). For $w_i(t)$ in Eqn. (5), we simply use higher and lower weights for the angular position and velocity components respectively in the first half of the rollouts, and vice versa in the second half. We select the best trajectory corresponding to the minimum cost (line 17) and execute the corresponding action (line 18), The best trajectory and action is returned to initialize the last best action $(\dot{\text{rpm}}_{\text{best}}, \dot{\psi}_{\text{best}})$ for the next planning cycle. We re-plan at 50 Hz and update the time horizon at every cycle based on the remaining time.

C. Robot, Gimbal, and Dataset

We implement PHLI and Dom Planner on a 1/5-scale Losi DBXL E2, 4WD Desert Buggy platform, with a top speed of 80+ km/h. The vehicle is equipped with a 9-DoF IMU, Intel RealSense D435i, NVIDIA Jetson Orin NX for perception and planning, two wheel encoders for front and rear wheels respectively, and an Arduino Mega microcontroller for all low-level actuators and the wheel encoders. For simplicity, we only allow the wheels to rotate forward, i.e., $\text{rpm} \in [0, \text{rpm}_{\text{max}}]$.

To collect a dataset while ensuring the safety of the robot, we construct a 2-axis 1.3 m$\times$1 m$\times$0.65 m aluminum gimbal platform capable of rotating in the roll and pitch axis. The purpose of the gimbal is to simulate weightlessness and in-air dynamics. When mounting the robot on the gimbal with robot's center of gravity aligned to the roll and pitch axis of the gimbal, the robot can freely rotate around the roll and pitch axis. Because existing vehicle controls do not have a direct effect on vehicle yaw acceleration and the gimbal introduces significant extra moment of inertial along the yaw axis, we do not include the yaw angle on the gimbal.

While the robot is mounted on the gimbal, the robot is commanded with a diverse range of $\dot{\text{rpm}}$ and $\dot{\psi}$ to capture all possible ways to change the roll and pitch of the vehicle, including behaviors at the extremes of rpm and steering ψ. The robot configurations during the execution of these commands are recorded and processed to create our tuple of current state, current action, and next state ground truth of the angular accelerations to train g_ϕ. Our dataset includes one hour of gimbal data, split 85/15 for training and validation.

IV. Experiments

For practical reasons, the majority of our experiments are performed indoors on the custom-built gimbal, with additional real-world validation provided by outdoor experiments. Please see our video `https://www.youtube.com/watch?v=jW-c0OP8pQQ` for the experimental results and extended version of our paper `https://arxiv.org/abs/2503.19140` for more information.

V. Conclusions

In this paper, we present the Dom Planner and PHLI to enable in-air vehicle maneuver for high-speed off-road navigation. Based on the precise in-air forward kinodynamics enabled by the hybrid PHLI model using physics-based and data-driven approaches, Dom Planner is able to accurately and timely maneuver the vehicle roll, pitch, yaw, and their velocities to desired states. Extensive experiments showcase that our method allows existing ground vehicle controls, i.e., throttle and steering, to prepare the vehicle in-air for safe landing during a short airborne period. *So, cars do fly.*