Advancing Reproducibility, Benchmarks, and Education with Remote Sim2real

Spencer Teetaert*, Wenda Zhao*, Antonio Loquercio†, Siqi Zhou*,‡, Lukas Brunke*,‡, Martin Schuck*,‡, Wolfgang Hönig§, Jacopo Panerati*, and Angela P. Schoellig*,‡
*University of Toronto Institute for Aerospace Studies, ON, Canada †University of California, Berkeley, CA, United States ‡Technische Universität München, Germany §Technische Universität Berlin, Germany

Abstract

Shared and repeatable benchmark problems have historically been a fundamental driver of progress for scientific communities. In academic conferences, benchmarks, replication tracks, and competitions offer the opportunity to researchers, often with different origins, backgrounds, and levels of seniority, to quantitatively compare their ideas. In robotics, a hot and challenging topic is *sim2real*—porting approaches that work well in simulation to real robot hardware. Hence, this article motivates and describes an aerial sim2real hardware-software framework that we created and used in *(i)* a competition that ran during the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems and *(ii)* two graduate courses, "Development of Unmanned Aerial Systems (AER1217)" at the University of Toronto during Winter 2023, and "Autonomous Drone Racing" at Technical University of Munich. (In our case, creating a hybrid competition with both simulation and real robot components was also dictated by the uncertainties around travel and logistics in the post COVID-19 world.) This article describes the task specification, the details of the software infrastructure supporting both simulation and real-life experiments, and the sim2real performance of the solution of the top-placed teams in the competition (link to the video), as well as the lessons learned by participants and organizers.

Index Terms

Aerial Systems: Applications, Motion and Path Planning, Software Tools for Benchmarking and Reproducibility

I. Introduction

Advances in robotics promise improved functionality, efficiency, and quality with an impact on many aspects of our daily lives. Example applications include autonomous driving, drone delivery, and service robots. However, the decision-making of such systems often faces multiple sources of uncertainty (e.g., incomplete sensory information, uncertainties in the environment, interaction with other agents, etc.). Deploying an embodied autonomous learning system in real-world [1] and possibly commercial applications requires both *(i)* safety guarantees that the system acts reliably in the presence of the various sources of uncertainties and *(ii)* efficient deployment of the decision-making algorithm to the physical world (for performance and cost-effectiveness). As highlighted in the "Roadmap for US Robotics" [2], learning and adaptation are essential for next-generation robotics applications, and guaranteeing safety is an integral part of this.

Robotics competitions, between and within universities, are essential for two main reasons: they are catalysts for progress in robotics research and have great educational value. First, robotics competitions provide a standardized platform for researchers to test and showcase their latest innovations. To maximize research progress, competitions use three main tools: *(i)* standard metrics, *(ii)* sequestered test data, to avoid solutions overfitting to the test set, and *(iii)* public leaderboards and reports (such as this one), to understand what are the key ideas behind the most successful entries. These tools enable objective comparisons between approaches, speeding up progress.

The second key value of competitions is education. They provide an opportunity for students to learn about robotics and apply their knowledge hands-on while promoting collaboration and teamwork. On top of their educational value, competitions represent a medium for students and organizers to network beyond universities, creating a community.

Given these benefits, it comes as no surprise that the aerial robotics community has organized and continues to develop multiple competitions. These competitions can greatly differ in scale. Some are very large-scale, attracting both private and government investment. Examples include the DARPA Fast Lightweight Autonomy (FLA) [3] program or the MBZIRC aerial robotic league [4]. Such large-scale efforts are very important to foster robotics progress, but they have a smaller impact on education.

Conversely, smaller-scale competitions significantly impact education, since they provide opportunities for young students to design and build full-stack robotics systems. Examples are the DodgeDrone series [5] and the sim2real IROS competition presented herein (Figure 1).

With our competition, which ran virtually during the 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), our goal was to bring together researchers from different communities to *(i)* solicit novel and data-efficient robot learning algorithms, *(ii)* establish a common forum to compare control and reinforcement learning approaches for safe robot decision-making [1], and *(iii)* identify the shortcomings or bottlenecks of the state-of-the-art algorithms with respect to real-world deployment.

Manuscript received XX; revised XX, 2024. Corresponding author: J. Panerati (email: jacopo.panerati@utoronto.ca).

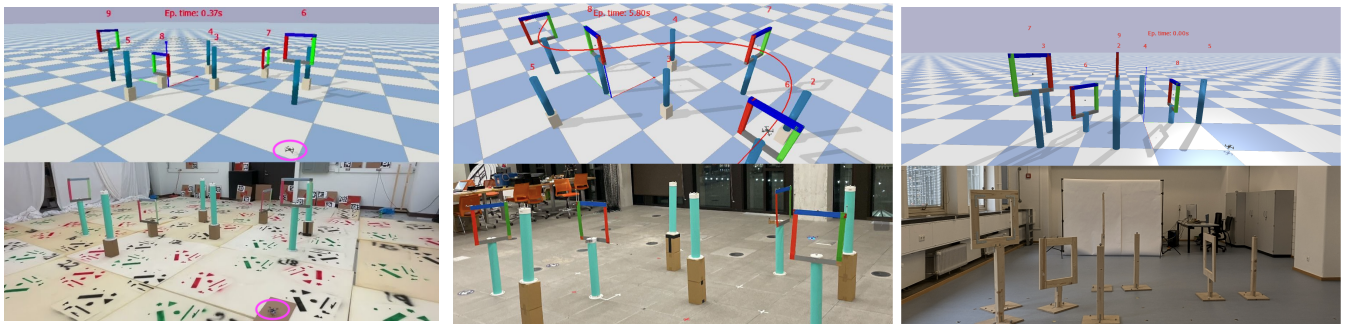

Fig. 1. Examples of the obstacle course in simulation (PyBullet, top) and in the real world (bottom *(left)* at the University of Toronto Institute for Aerospace Studies; *(center)* in AER1217 at the Myhal Centre for Engineering Innovation & Entrepreneurship; *(right)* and at Technical University of Munich).

TABLE I
SIMULATED AND REAL-WORLD EVALUATION SCENARIOS OF THE COMPETITION

Evaluation Scenario	Constraints	Rand. Inertial Properties	Randomized Obstacles, Gates	Rand. Between Episodes	Notes
level 0	**Yes**	*No*	*No*	*No*	Perfect knowledge
level 1	**Yes**	**Yes**	*No*	*No*	Adaptive
level 2	**Yes**	**Yes**	**Yes**	*No*	Learning, re-planning
level 3	**Yes**	**Yes**	**Yes**	**Yes**	Robustness
sim2real	**Yes**	Real-life hardware	**Yes**, injected	*No*	Sim2real transfer

Note: "Rand. Between Episodes" states whether randomized properties and positions vary or are kept constant across episodes.

In this article, we report on four major contributions:

- The benchmark/competition task design is intended to stress and evaluate the controllers proposed by the participant teams along multiple axes—safety, robustness, adaptiveness, and learning—using multiple levels of increasing complexity.
- The implementation details of the sim2real infrastructure used to let participants enter the competition remotely (before IROS 2022) while also having their solution evaluated in real flying robotic hardware (during IROS 2022). Importantly, this helped to foster diversity in the origin, background, and seniority of the competition's participants.
- The quantitative evaluation of the sim2real transfer performance of the solutions of the top 3 participant teams—University of Science and Technology of China, Ekumen, and H^2.
- The open sourcing of all the code associated with the sim2real framework (and the competition solutions). Crucial for reproducibility and democratization of robotics research [6].

The rest of the article is organized as follows. Section II outlines the competition task and the implementation of the simulation and sim2real frameworks. Sections III, IV and V report on the results obtained during the 2022 IROS competition, the 2023 University of Toronto AER1217 "Development of Autonomous UAS" (Figure 1) and the TUM "Autonomous Drone Racing" course. Finally, Section VI summarizes the lessons learned by participants and organizers.

II. REMOTE AERIAL SIM2REAL

A. Task and Evaluation Scenarios

The task is to design (in simulation) a controller/planner that enables a nano-quadrotor (specifically, the Bitcraze Crazyflie 2.x) to safely fly (in the real-world) through a set of gates and reach a predefined target despite uncertainties in the robot dynamics (i.e., mass and inertia), in the robot actuation (i.e., input disturbances), and the environment (i.e., wind and position of the gates).

Solutions are evaluated in terms of both their safety (i.e., producing no collisions with gates or obstacles) and performance (i.e., time to reach and stabilize at the target position, having navigated all the gates in the given order). The task was designed to encourage the exploration of both control and reinforcement learning approaches (e.g., robust, adaptive, predictive, learning-based and optimal control, and model-based/model-free reinforcement learning).

The controller/planner of each solution has access to the position and attitude measurements (provided by the simulation engine or a motion capture system in real-life experiments) and the pose of the closest next gate. To mimic a limited-range perception system, the exact gate pose is only revealed to the controller once the quadrotor is within half a meter of it. The controller sends position, velocity, acceleration, and heading references to the onboard Bitcraze controller.

For the IEEE/RJS IROS 2022 competition, all the proposed solutions were evaluated in 5 scenarios with different challenges, culminating in real-world evaluations (see Table I). Included were both state (do not leave the boundaries of the flight arena) and input (feasible action) constraints.

Each solution was required to re-implement 2 methods (the initialization and the action selection of the `Controller` class) and, optionally, could re-implement 2 more methods allowing learning between control steps (`interStepLearn()`) and between training episodes (`interEpisodeLearn()`).

For all levels in Table I (0-3, sim2real), solutions were evaluated—on the last episode of each level—by:

- *safety*, i.e., the ability to avoid *(i)* all collisions with gates and obstacles and *(ii)* all constraint violations (only episodes with 0 collisions and violations are considered as successful task completions);
- and *performance*, i.e., minimizing the task time (in seconds) required to complete the task (flying through all the gates, reaching, and stabilizing at the goal position).

For all levels in Table I (0-3, sim2real), solutions that accomplish the task (as described in the previous paragraph) were also evaluated—across all episodes in each level—by:

- *data & compute efficiency*, i.e., minimizing the simulation-clock/flight time (in seconds) plus the overall wall-clock learning time (in seconds) used by methods `interStepLearn()` and `interEpisodeLearn()`.

For all levels, the top 3 solutions ranked by *safety* and *performance* and the top 3 solutions ranked by *data & compute efficiency* were given 20, 10, and 5 points respectively. The sum of these points determined the final classification (see Table II and this **video of the top 3 solutions**.

For the AER1217 course, we wanted to create a course project focused on motion planning. Hence, we were able to leverage the same sim2real hardware-software framework. The students were tasked with developing a trajectory planning algorithm to navigate the quadrotor safely through all the gates in a pre-defined sequence and reach the target point. The simulation and real-world setup for this course project are showcased in Figure 1. There are four gates and four obstacles in the environment. We simplified the task by setting the gates to be the same height (one meter), leading to a 2-D motion planning problem, and eliminated the uncertainties of the gates' positions. We provide exact positions for the gates and nominal positions for the obstacles. Uniform noise $\mathbf{U}(-0.2, 0.2)$ is injected into the obstacles' nominal x and y positions to introduce uncertainty and create a more challenging environment.

We provided the simulation environment to the students two weeks in advance of the demo day. The information that was withheld from the students was the exact ordering of the gates for the final navigation task. Each team of students was required to use either (1) search-based [7] or (2) sampling-based [8] methods to develop their motion planning algorithm. During the day of real-world experiments, we announced the gate sequence and evaluated the algorithm in both simulation and real-world performance. The performance was ranked by the amount of time the quadrotor would take to complete the task.

B. Implementation Details

One of the main contributions of our work is to break away from both purely simulated and in-person-only competitions by creating a framework for the remote and asynchronous development of novel robot controllers.

1) Sim2real: We developed a novel software framework to minimize the differences between simulation and real-world application, enabling controllers designed in simulation to be tested on flight hardware without the need for fine-tuning. Our *sim2real* framework consists of three parts:

- `pycffirmware`[1]: an extension of Bitcraze's Crazyflie firmware[2] Python bindings;
- a firmware wrapper module to interface between `pycffirmware` and `safe-control-gym`[3]
- and a module to execute controllers on the flight hardware using *Crazyswarm*[4].

The firmware wrapper was designed to use the Crazyswarm API so that controllers could be copy-and-pasted *as is* between simulation and flight hardware. The module wraps the Crazyflie firmware to ensure onboard changes to command signals are captured in `safe-control-gym` [9].

Our goal was to let the competition's submissions be developed using the sim2real pipeline in simulation and then, to be able to test the different solutions in both simulation and on flight hardware.

To validate the sim2real pipeline, we developed 7 test controllers and trajectories in simulation and flew each one of them 30 times on real Crazyflies. The error between the simulated flight path and real-world flight path varied by less than 5cm on average, with total flight path lengths varying from less than 2m to greater than 10m.

2) `pycffirmware`: We developed an extension to the Crazyflie Python bindings to facilitate full wrapping of the functional elements of the Crazyflie's firmware, that is, modules in the firmware that limit execution rates or change a reference signal's value. For this competition, we required teams to use the firmware's Mellinger controller. We note that `pycffirmware` was developed in parallel to extensions of Bitcraze's official Python binding that are used in the Crazyflie Webot simulation wrapper and Crazyswarm2.

3) `safe-control-gym`: We developed a firmware wrapper module for `safe-control-gym` [9] that interfaces with the `pycffirmware` bindings. This provides an interface between the firmware variables and the simulation physics engine. A quadratic mapping is used to convert motor PWM commands from the firmware to thrust magnitudes used by `safe-control-gym`. The competition included timing hints on the execution of each new controller but did not enforce a hard cutoff, allowing the development of computationally expensive solutions. These would perform above average in simulation but also incur a poorer sim2real transfer.

[1] `https://github.com/utiasDSL/pycffirmware`
[2] `https://github.com/bitcraze/crazyflie-firmware`
[3] `https://github.com/utiasDSL/safe-control-gym`
[4] `https://github.com/USC-ACTLab/crazyswarm`

TABLE II
SCORES (SAFETY & PERFORMANCE; DATA EFFICIENCY) AND FINAL RANKINGS

	lvl0	lvl1	lvl2	lvl3	sim2real	Tot. (Rank)
USTC	20; 5	20; 5	5; 5	5; 5	10; 5	85 (3)
Ekuflie	10; 10	10; 10	20; 10	20; 10	5; 10	115 (2)
H^2	5: 20	5; 20	10; 20	10; 20	20; 20	150 (1)

4) Crazyswarm: To ensure compatibility between the sim2real implementation in `safe-control-gym` and the real-world implementation we limited the use of the Crazyswarm [10] Python API to functions that we validated in `safe-control-gym`. This includes `goTo()`, `cmdFullState()`, `takeoff()`, `land()`, and `stop()` from the `Crazyflie` class. Commands were sent from a ground station to the Crazyflie robot at a rate of 30 Hz. We used a Vicon motion tracking system also to measure the locations of each obstacle at the start of each trial and provide accurate information to the controller of each solution.

III. IROS 2022 COMPETITION RESULTS

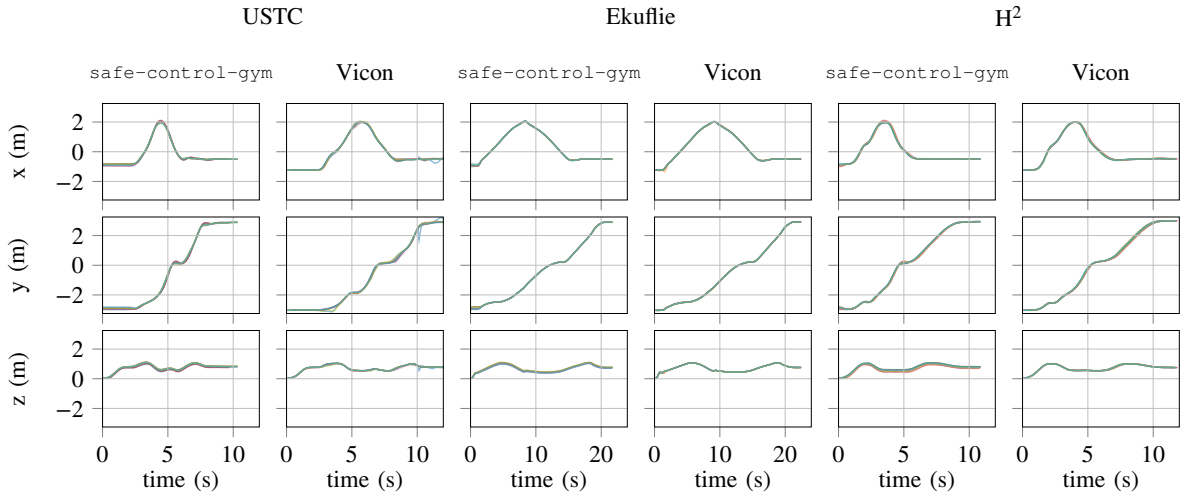

Fig. 2. Comparison over 10 repeated simulations (`safe-control-gym`) and 10 real-life experiments (Vicon) of the trajectories followed by the Crazyflie commanded by the solutions of each of the 3 finalist teams (USTC, 1st and 2nd columns, Ekuflie, 3rd and 4th, H^2, 5th and 6th).

The top 3 solutions were submitted by teams H^2 from Singapore[5], Ekuflie from Argentina[6], and University of Science and Technology of China[7], respectively[8].

We performed simulations on a Lenovo P52 workstation laptop, equipped with 32GB of RAM, an Intel i7-8850H CPU, and an NVIDIA Quadro P2000 GPU, running Ubuntu 20.04.4 LTS. The same laptop was used, together with the University of Toronto's Vicon motion capture system, to run Crazyswarm [10] for real-life experiments.

Figure 2 reports the positions, over time, in x, y, and z of the Crazyflie drone in simulation (columns titled `safe-control-gym` and in the real world experiments (as tracked by Vicon) for the top 3 solutions. We observe the pair-wise similarity, along the columns, on each row. This shows that there is a close match between the trajectories followed by the simulated and the real-life quadrotor. This includes both the directions traveled the most (i.e. x and y, for $\sim$4m) as well as z (it is important to observe that the gates were placed at different heights). The results show an effective sim2real transfer for all solutions, especially w.r.t. their safety that, in our competition setup was expressed through multiple position constraints.

We quantify the sim2real performance of each of the 3 solutions by plotting the average position error (again, in x, y, and z) between 10 repeated simulations and 10 repeated real-life experiments[9]. The lowest sim2real position RMSE (0.21, 0.19, 0.07) is registered by team Ekuflie, however, theirs is the slowest solution ($\sim$21s). Team H^2 achieved similar RMSE with a much faster ($\sim$11s) completion time. Team USTC's solution was similarly quick but also resulted in a much higher sim2real mismatch: once transferred to the real system, the controller could not run in real-time, lagging a few seconds behind its simulated counterpart.

Table II summarizes the point assignments that each of the top three finalists received, according to the scoring system in Section II. Winning team H^2 consistently outperformed its adversaries in terms of data efficiency and in the sim2real

[5] `github.com/huiyulhy/safe-control-gym/tree/lspb_s_curve`
[6] `github.com/ekumenlabs/safe-control-gym`
[7] `github.com/ustc-arg/Safe-Robot-Learning-Competition`
[8] "A Remote Sim2real Aerial Competition" (detailed solutions on arXiv)
[9] `github.com/JacopoPan/iros-competition-analysis`

stage of the competition. Ekuflie's performance was especially good in the more complex (2 and 3) simulation levels but could not effectively transfer to the sim2real stage of the competition because of the lack of real-time constraints on the interstep optimization stage (that led to the controller falling back on a more conservative approach). Team USTC had the best performance in the easier simulation levels but fell behind in the more complex ones. The three teams were awarded Bitcraze AI, STEM Ranging, and STEM bundles for the 1st, 2nd, and 3rd position placements, respectively[10].

IV. University of Toronto AER1217 Course

We evaluated the algorithms developed and submitted by the student teams in the graduate course AER1217 in the flight arena of the University of Toronto's Myhal Centre for Engineering Innovation & Entrepreneurship. In the two-story indoor flight arena, a Vicon motion capture system is installed to offer millimeter-level accurate optical tracking, serving as an external localization solution for indoor flight experiments.

Students were able to develop their algorithms in simulation in advance of the course evaluation day. On the day of the course flight experiments, we provided the final sequence of the gates for the navigation task and assessed the submitted algorithms in a real-world setup. Each team had 3 opportunities to execute their algorithms on a real-world quadrotor, and their solutions were assessed based on the quadrotor's fastest time while ensuring the safe execution of the navigation task.

The students were able to effectively transfer their algorithms developed in simulation to the real-world setup with minimal changes, ensuring a smooth day of flight operations to conclude the AER1217 course. All eight teams completed the course project successfully, and the winning margin was under a second, making for some very exciting racing[11].

To further evaluate the proposed `sim2real` software and hardware architecture, we executed the algorithms submitted by 5 student teams from the course across 5 repeated trials. The zero-shot sim2real trajectories are shown in Figure 3. The experimental results from the repeated sim2real flights consistently demonstrate the efficacy of the proposed software and hardware architecture, achieving a reliable and representative sim2real transfer with an average error of ∼9cm and an average maximum error of ∼31cm. This finding substantiates the suitability of the proposed software and hardware architecture as a reliable and effective sim2real platform for both researchers and educators within the aerial robot community.

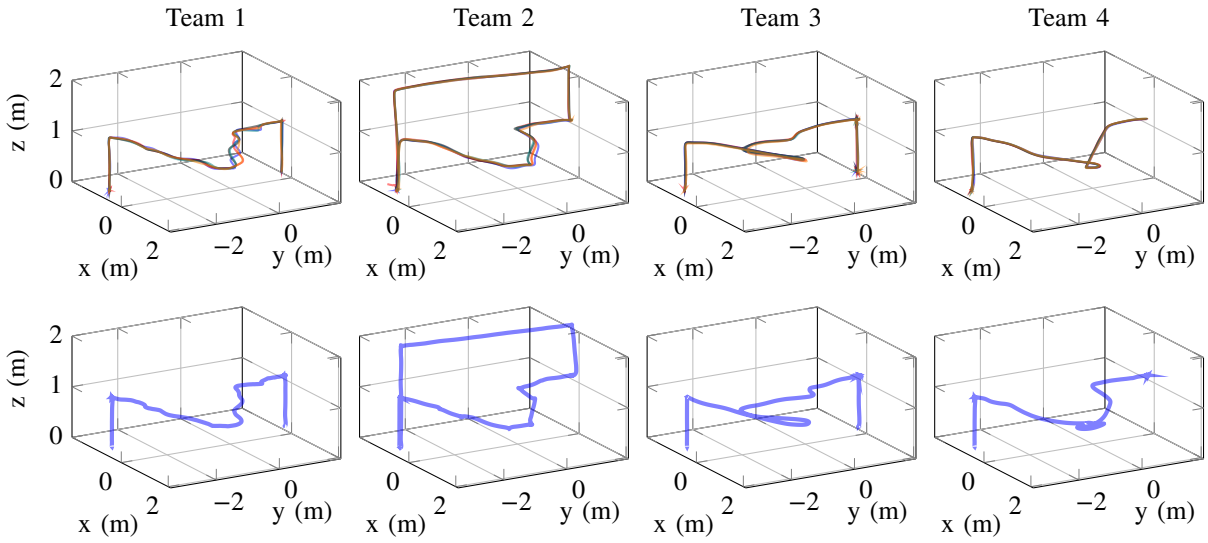

Fig. 3. Repeated (5 times each), zero-shot sim2real trajectories (top row) corresponding to the different solutions—developed in simulation (bottom row)—by 4 student teams enrolled in the University of Toronto's AER1217 course. Data was collected in the Myhal Centre for Engineering Innovation & Entrepreneurship.

V. TUM Autonomous Drone Racing Activity

Since 2023, our sim2real challenge has also been re-proposed twice for the semester-long graduate course "Autonomous Drone Racing" at the Technical University of Munich (TUM). Ten student teams competed throughout the semester in simulated races and deploy their solutions at the Learning Systems and Robotics drone lab (Figure 1, right) at the end of the term. Similarly to AER1217, we use a Vicon Valkyrie system with six cameras for external localization of the drones. In the first half of the course, students developed their algorithms purely in simulation leveraging the same `sim2real` implementation and disturbances described in II-B. To encourage competition among teams, new solutions are continuously tested and posted on an online leaderboard[12]. In the second half of the term, we allowed teams to test their implementation on the real drones

[10] https://www.bitcraze.io/2022/12/iros-2022-crazyflies-adventures-in-robotic-research/

[11] www.utias.utoronto.ca/news/flying-through-obstacles-the-gravity-defying-course-on-unmanned-aerial-systems

[12] www.kaggle.com/competitions/lsy-drone- racing-ss24

to get familiar with the hardware. The course concluded with a race day, where teams deployed their finalized approaches on the real track.

VI. Lessons Learned and Future Outlook

Concerning demographics, a survey run after the completion of the IEEE/RSJ IROS competition showed that 55.6% of the participants came from Asia and 44.4% from the Americas (we should note that the synchronous part of the competition—final evaluations and awards—was held in a timezone that was especially inconvenient to European and African participants). The survey also showed that team sizes ranged from 1 to 7 people, they comprised 33.3% of industry professionals, 44.4% of master's, and 22.2% of undergraduate students. One-third of the respondents to the survey stated that they worked on their solution for over 4 months (i.e., from the earliest public announcement of the competition codebase). Bringing together participants from such different geographies, in real-life robotic experiments, demonstrates the value of our remote sim2real pipeline as a research and collaboration tool. The fairness of the remote competition was also only possible thanks to the minimization of the sim2real gap.

On a scale from 1 to 5, the participants scored the competition relevance as 4.5 for *robotics*, 3.4 for *safety*, and 3.1 for *learning*. The lower scores given to the learning aspect were supported by the following comments: "the scenarios were not diverse enough to make it a requirement and the scoring system would penalize solutions that spent [...] time learning to improve performance."; "I think that the plausibility for reinforcement learning-based methods [...] for the solution was relatively less as compared to the traditional control-based approaches [...]. Using adaptive control might suffice for tackling the external disturbances/rejections."; "Having a Mellinger controller in the loop restricts the [...] ability to shape the system response and thus no attempt was made to learn and correct for uncertainty or bias in model dynamics." Indeed, these are all important lessons for follow-up and future competitions aiming at benchmarking learning-based control.

Overall, the difficulty level and the scoring systems were considered either fair or only slightly too difficult by 88.8% and 77.7% of the survey respondents, respectively. The entirety of the respondents stated that they would like to see the same competition re-proposed in the future in its current format (66.7%) or with only minor modifications (33.3%).

For the AER1217 course, a total of 21 master's students, organized into 8 groups, participated in the course project. With a mere two weeks of development in simulation, all eight teams managed to effectively transfer their algorithms to the real world and completed the demo day navigation task. However, we also observed the sim2real gap increasing with the students attempting faster and faster solutions. The quadrotor also experienced crashes on fast and aggressive trajectories despite these being successfully validated in simulation. In the upcoming framework releases we will incorporate stress evaluations, issuing warnings when the system approaches the boundaries of the simulation representation.

We believe that the software infrastructure we built for the competition and education is also useful for the effective and efficient validation of many other aerial research topics. With regard to the sim2real component, we observed that the competition infrastructure measured (and raised warnings associated with) the run-time of the learning and optimization methods of each solution controller. However, it did not contain provisions for their clipping. This allowed for solutions' complexity (e.g., Ekuflie's) to remain potentially unbounded, resulting, on occasions, in different simulation and real-world performances. In future iterations of the framework, we will implement strict run-time constraints in the simulation code to avoid this phenomenon. We are also actively exploring the use of learned dynamics in the simulation based on data collected on the real drones to further narrow the sim2real gap. Another ongoing plan is to integrate some of our developed components for wider use. Specifically, we plan to integrate our additions in `pycffirmware` to the official Python bindings and the physics simulation of `safe-control-gym` to Crazyswarm2.

References

[1] L. Brunke, M. Greeff, A. W. Hall, Z. Yuan, S. Zhou, J. Panerati, and A. P. Schoellig, "Safe learning in robotics: From learning-based control to safe reinforcement learning," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, no. 1, 2022. [Online]. Available: https://doi.org/10.1146/annurev-control-042920-020211

[2] H. Christensen, N. Amato, H. Yanco, M. Mataric, H. Choset, A. Drobnis, K. Goldberg, J. Grizzle, G. Hager, J. Hollerbach, S. Hutchinson, V. Krovi, D. Lee, B. Smart, J. Trinkle, and G. Sukhatme, 2021.

[3] S. Paschall and J. Rose, "Fast, lightweight autonomy through an unknown cluttered environment: Distribution statement: A — approved for public release; distribution unlimited," in *2017 IEEE Aerospace Conference*, 2017, pp. 1–8.

[4] "Mohamed Bin Zayed International Robotics Challenge (MBZIRC)," https://cps.iisc.ac.in/research/mbzirc/, 2017.

[5] A. Loquercio, E. Kaufmann, L. Bauersfeld, Y. Song, and D. Scaramuzza, "ICRA 2022 DodgeDrone Challenge: Vision-based agile drone flight." [Online]. Available: https://uzh-rpg.github.io/icra2022-dodgedrone/

[6] S. Zhou, L. Brunke, A. Tao, A. W. Hall, F. P. Bejarano, J. Panerati, and A. P. Schoellig, "What is the impact of releasing code with publications? statistics from the machine learning, robotics, and control communities," 2023.

[7] A. Le, T. Le, and R. Roka, "Search-based planning and replanning in robotics and autonomous systems," *Advanced Path Planning for Mobile Entities*, pp. 63–89, 2018.

[8] A. Orthey, C. Chamzas, and L. E. Kavraki, "Sampling-based motion planning: A comparative review," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 7, 2023.

[9] Z. Yuan, A. W. Hall, S. Zhou, L. Brunke, M. Greeff, J. Panerati, and A. P. Schoellig, "Safe-control-gym: A unified benchmark suite for safe learning-based control and reinforcement learning in robotics," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 11 142–11 149, 2022.

[10] J. A. Preiss, W. Honig, G. S. Sukhatme, and N. Ayanian, "Crazyswarm: A large nano-quadcopter swarm," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3299–3304.