Swarm-GPT: Combining Large Language Models with Safe Motion Planning for Robot Choreography Design

Aoran Jiao[1], Tanmay P. Patel[1], Sanjmi Khurana[1], Anna-Mariya Korol[1], Lukas Brunke[2,3], Vivek K. Adajania[3], Utku Culha[2], Siqi Zhou[2], and Angela P. Schoellig[2,3]

[1]Department of Engineering Science, University of Toronto
[2]Department of Computer Engineering, Technical University of Munich
[3]Institute for Aerospace Studies, University of Toronto

Abstract

This paper presents Swarm-GPT, a system that integrates large language models (LLMs) with safe swarm motion planning— offering an automated and novel approach to deployable drone swarm choreography. Swarm-GPT enables users to automatically generate synchronized drone performances through natural language instructions. With an emphasis on safety and creativity, Swarm-GPT addresses a critical gap in the field of drone choreography by integrating the creative power of generative models with the effectiveness and safety of model-based planning algorithms. This goal is achieved by prompting the LLM to generate a unique set of waypoints based on extracted audio data. A trajectory planner processes these waypoints to guarantee collision-free and feasible motion. Results can be viewed in simulation prior to execution and modified through dynamic re-prompting. Sim-to-real transfer experiments demonstrate Swarm-GPT's ability to accurately replicate simulated drone trajectories, with a mean sim-to-real root mean square error (RMSE) of 28.7 mm. To date, Swarm-GPT has been successfully showcased at three live events, exemplifying safe real-world deployment of pre-trained models.

1 Introduction

For millennia, dance has been a powerful medium for human expression and entertainment. However, our mastery of robot dynamics and control has only recently expanded to the realm of dancing and performance. One illustrative application of robot choreography in the entertainment industry is drone shows, where many unmanned aerial vehicles (UAVs) morph into intricate patterns in mid-air, synchronized with musical rhythms (1; 2; 3).

Imparting dance to robots was a natural progression but yet non-trivial (4). In current applications, the robots' spatiotemporal movements are often manually choreographed by experts to balance expressivity and safety. Generating robot choreographies can be laborious; as the number of robots increases, the complexity in choreographic design and safety analysis can quickly become intractable (5).

In this work, we propose Swarm-GPT — an automated drone choreography pipeline that translates language instructions into choreographed motions of a swarm of nano-drones. To facilitate intuitive interaction and safe deployment of language-based drone choreography, the system combines *(a)* a high-level motion generation layer that leverages the generative abilities of the large language model (LLM) to design unique drone swarm choreographies and *(b)* a low-level safety layer that uses model-based swarm planning frameworks to ensure feasible and safe deployment of the LLM-generated choreographies to physical robots.

NeurIPS 2023 Workshop on Robot Learning: Pretraining, Fine-Tuning, and Generalization with Large Scale Models, New Orleans, United States of America

Figure 1: Choreography of a 9-drone swarm by Swarm-GPT. Formations shown in the picture are synchronized to the music beat times. Users can interact with Swarm-GPT to generate different trajectories. Full experimental performance available from the video: http://tiny.cc/Swarm-GPT

Through simulation and physical experiments, we show how to effectively bridge the gap between high-level natural language instructions and low-level robot control and coordination. Swarm-GPT is a proof-of-concept showcasing the viability of LLM-directed drone swarm control, which can be accomplished safely. Our work demonstrates that LLMs can be used as an intuitive interface for non-expert users to generate complex drone behaviours while being augmented by underlying safe control and planning algorithms to enable safe coordination by design.

To the best of our knowledge, Swarm-GPT is the first system that enables the direct use of LLMs for drone choreography. Our contributions are as follows: *(i)* enabling non-experts to program and modify choreographies of drone swarms by using natural language, *(ii)* seamlessly integrating large language models with a model-based safety filter to guarantee safe execution, and *(iii)* demonstrating Swarm-GPT in real-time drone experiments.

2 Related Work

Robot Choreography. The study of robot choreography spans across different robot platforms, including both humanoid robots (6; 7; 8; 9) and non-anthropomorphic forms such as robot arms (10), drones (1; 2; 3), and quadruped robots (6; 7). While previous efforts have focused on automating the process by mapping music features(as in (7; 8; 2; 3)), the choreography process still often requires manual tuning and domain expertise, and provides minimal means for intuitive feedback, especially for non-expert users. This work focuses on drone swarm choreographies and introduces an automated process with a natural language feedback tool for synchronized spatiotemporal motion generation.

LLMs for Robotics. Robot decision-making through natural language recently started gaining traction (11; 12), especially in the areas of visual-language navigation (13), language-grounded robot affordances (14), language-based trajectory modification (15), and high-level task planning through LLM-based code-generation (16; 17). These examples highlight the potential of leveraging LLMs to facilitate natural interaction with robotic systems. In this work, we pursue a similar goal but with a focus on language-based drone swarm choreography and its safe real-world deployment.

Safe Robot Swarm Decision-Making. Safe robot decision-making approaches generally exploit our prior knowledge about the robot to provide desired safety guarantees (18). In the context of swarm coordination, model-based motion planning offers a natural means to explicitly account for safety and feasibility constraints. Recent advances in model-based swarm motion planning include both centralized methods (19; 20) and distributed methods (21; 22; 23). Given the scalability advantages of distributed methods (21; 22; 23), in this work, we adopt a state-of-the-art distributed drone swarm motion planning framework (23) as a safety filter for LLM-based swarm choreography design. This integration addresses the challenge of safe deployment by effectively leveraging our prior knowledge about the robots while maximally preserving the motion generated by the LLM.

3 Methodology

In this section, we provide an overview of the proposed Swarm-GPT framework. A block diagram of the Swarm-GPT system is shown in Figure 2 and further technical details are included in Appendix A.

LLM Interface. The LLM interface offers a means for users to provide high-level task specifications such as song selection and swarm behaviour through natural language. In parallel, the features of the selected song are extracted using an audio analysis tool, librosa, (24). The language input and extracted music features together formulate an LLM prompt template with preliminary instructions

about the physical constraints of the operating environment. We include details of the prompting process in Appendix A.1. Given the language input, song audio file, and high-level prior knowledge of the environment, the LLM generates choreography for the drone swarm as a series of timed position waypoints for each drone, synchronized with the beats of the selected song.

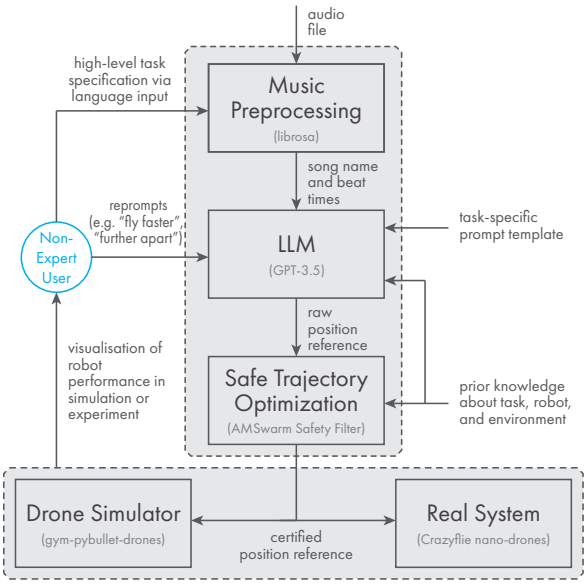

Figure 2: Swarm-GPT system block diagram

Safe Trajectory Optimization. The safe trajectory optimization module is a safety filter formulated based on a state-of-the-art distributed drone swarm motion planning framework, the AMSwarm (23). The AMSwarm safety filter allows us to incorporate our prior knowledge about the robot system (e.g., maximum allowable speed and actuation limits of the drones as well as ellipsoidal collision envelopes of the drones) and subsequently computes feasible and collision-free swarm trajectories that best match the choreography waypoints generated by the LLM. The formulation of the AMSwarm-based safety filter design is further discussed in Appendix A.2.

Trajectory Modification (Re-Prompting). These waypoints are treated as position commands and are visualized using a drone swarm simulator, the gym-pybullet-drones (25). The user may deploy the generated choreography onto the real drone system or modify it by re-prompting. To re-prompt, the user can enter a custom prompt (e.g., "go faster, higher"), which is appended to the original prompt. The LLM then modifies the swarm by updating the waypoints based on the reprompt. Collision avoidance and processing are applied to the updated LLM outputs.

4 Simulation and Experimental Evaluation

The proposed Swarm-GPT system was tested with four songs of different genres {*Perfect*, *Ode to Joy*, *Mission Impossible Theme*, *Here Comes the Sun*} and swarms up to nine drones in both simulation and real-time experiments. In this section, we present key evaluation results; additional discussions are included in Appendix B. A video of the robot experiments is available at this link: http://tiny.cc/Swarm-GPT.

Assessment of Sim-to-Real Transfer. We use gym-pybullet-drones (25) as the simulator of Swarm-GPT. The simulator is equipped with the same controllers as the real drone platform. To examine the sim-to-real transfer gap, the control signal input into the simulator and the real system are identical. The root mean square error (RMSE) of the simulated x, y, z positions and the ground truth x, y, z positions are plotted in the bar graph for six different choreography configurations. As Fig. 3 illustrates, the average sim-to-real RMSE among the swarm of drones does not exceed 60 mm in the worst case. The average RMSE across the six choreographies is 28.7 mm. With the high sim-to-real transfer quality, we use simulation to collect larger quantities of data to illustrate the safety features and generative abilities of Swarm-GPT.

Response to Safety Constraints. LLMs are excellent creative tools, but they have limited awareness of dynamics and robot constraints. Our pipeline complements the creative power of LLMs with well-tested and robust safety filters. Collectively, the pipeline achieves safe drone motion that humans can design through natural language. Fig. 4 shows the number of collision events before and after applying our safety filter. To count collision events, we consider ellipsoidal collision-avoidance envelopes that account for both true dimensions and aerodynamic effects (e.g., downwash) of the drones (23). Across 97 simulation trials, only 21.65% of the raw LLM-generated trajectories were collision-free. This figure reached 100% after the safety algorithm, meaning all collisions were eliminated. We note

that, during real-world deployment, there could be additional dynamic uncertainties or imperfections; these uncertainties could be systematically accounted for by using robustly safe formulations (18).

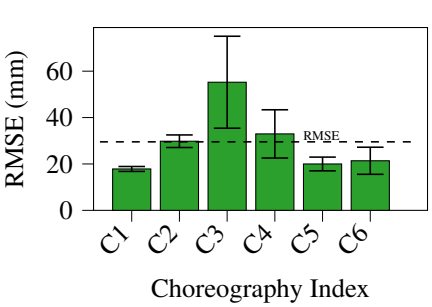

Figure 3: Summary of sim-to-real RMSE for six different choreographies.

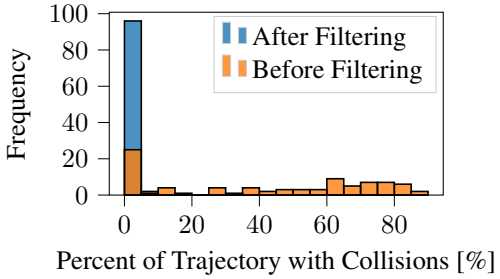

Figure 4: Histogram of the percentages of collisions along trajectories before and after applying our AMSwarm safety filter. All data points were collected with nine-drone swarms.

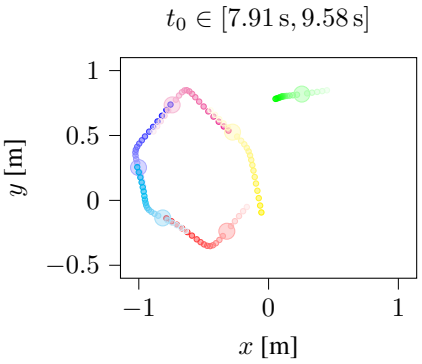

Figure 5: Visualization of Swarm-GPT's choreography. Different colours correspond to different drones. The large dots represent beat times, and the colour gradient moves from lighter to more saturated as time progresses.

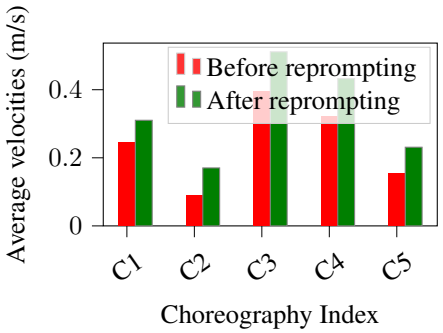

Figure 6: The LLM can be reprompted to modify trajectories. This figure shows the mean drone velocity before and after reprompting averaged across 10 choreographies per song. The reprompt used instructed the drones to "fly faster."

Prompting and Synchronization to Music. To ensure an accurate synchronization between the choreography and the music and facilitate intuitive interactions, we instruct ChatGPT to generate waypoints at the extracted timestamps of the beats for the music provided while achieving desired behaviours (e.g., "symmetric formation"). An example is shown in Fig. 5, where the (x, y) positions of 6 drones in one choreography are plotted against the beat times at each large solid dot. The fleet of drones maintains a desired formation and distinctly changes their trajectory around the beat times. Fig. 6 shows further examples of reprompting, where the drones are instructed to "fly faster."

5 Conclusion

We present a novel generative choreography system, Swarm-GPT, that integrates high-level LLMs and low-level safe drone swarm motion planning for safe and interactive drone swarm choreography. The design methods are demonstrated in experiments, showing the efficacy of combining LLM with model-based safety filter designs for safe real-world deployment.

References

[1] Cirque du Soleil, "SPARKED: Behind the technology," September 2014, [Online] Available: https://youtu.be/7YqUocVcyrE?si=GRWwC_bbIisFsuMu.

[2] A. Schoellig, F. Augugliaro, S. Lupashin, and R. D'Andrea, "Synchronizing the motion of a quadrocopter to music," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 3355–3360.

[3] X. Du, C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Fast and in sync: Periodic swarm patterns for quadrotors," in *Proc. of the International Conference on Robotics and Automation (ICRA)*, 2019, pp. 9143–9149.

[4] M. Apostolos, M. Littman, S. Lane, D. Handelman, and J. Gelfand, "Robot choreography: An artistic-scientific connection," *Computers & Mathematics with Applications*, vol. 32, no. 1, pp. 1–4, 1996.

[5] M. Waibel, B. Keays, and F. Augugliaro, "Drone shows: Creative potential and best practices," ETH Zurich, Tech. Rep., 2017, [Online] Available: http://hdl.handle.net/20.500.11850/125498.

[6] Boston Dynamics, "Do You Love Me?" December 2020, [Online] Available: https://www.youtube.com/watch?v=fn3KWM1kuAw.

[7] E. Guizzo, "By leaps and bounds: An exclusive look at how Boston Dynamics is redefining robot agility," *IEEE Spectrum*, vol. 56, no. 12, pp. 34–39, 2019.

[8] G. Xia, J. Tay, R. Dannenberg, and M. Veloso, "Autonomous robot dancing driven by beats and emotions of music," in *Proc. of the International Conference on Autonomous Agents and Multiagent Systems*, 2012, pp. 205–212.

[9] M. Boukheddimi, D. Harnack, S. Kumar, R. Kumar, S. Vyas, O. Arriaga, and F. Kirchner, "Robot dance generation with music based trajectory optimization," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 3069–3076.

[10] A. Rogel, R. Savery, N. Yang, and G. Weinberg, "RoboGroove: Creating fluid motion for dancing robotic arms," in *Proc. of the International Conference on Movement and Computing*, 2022, pp. 1–9.

[11] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill *et al.*, "On the opportunities and risks of foundation models," *arXiv:2108.07258*, 2021, [Online] Available: https://crfm.stanford.edu/assets/report.pdf.

[12] S. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, "ChatGPT for Robotics: Design Principles and Model Abilities," 2023, [Online] Available: http://arxiv.org/abs/2306.17582.

[13] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual language maps for robot navigation," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 10 608–10 615.

[14] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian *et al.*, "Do as I can, not as I say: Grounding language in robotic affordances," in *Proc. of the Conference on Robot Learning*, 2023, pp. 287–318.

[15] A. Bucker, L. Figueredo, S. Haddadin, A. Kapoor, S. Ma, S. Vemprala, and R. Bonatti, "Latte: Language trajectory transformer," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 7287–7294.

[16] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: Generating situated robot task plans using large language models," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 11 523–11 530.

[17] Y. Tang, W. Yu, J. Tan, H. Zen, A. Faust, and T. Harada, "Saytap: Language to quadrupedal locomotion," *arXiv:2306.07580*, 2023, [Online] Available: https://arxiv.org/abs/2306.07580.

[18] 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, pp. 411–444, 2022.

[19] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 1917–1922.

[20] T. Schouwenaars, B. De Moor, E. Feron, and J. How, "Mixed integer programming for multi-vehicle path planning," in *Proc. of the IEEE European Control Conference (ECC)*, 2001, pp. 2603–2608.

[21] C. E. Luis, M. Vukosavljev, and A. P. Schoellig, "Online trajectory generation with distributed model predictive control for multi-robot motion planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 604–611, 2020.

[22] E. Soria, F. Schiano, and D. Floreano, "Distributed predictive drone swarms in cluttered environments," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 73–80, 2021.

[23] V. K. Adajania, S. Zhou, A. K. Singh, and A. P. Schoellig, "AMSwarm: An alternating minimization approach for safe motion planning of quadrotor swarms in cluttered environments," in *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1421–1427.

[24] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proc. of the Python in Science Conference*, vol. 8, 2015, pp. 18–25.

[25] J. Panerati, H. Zheng, S. Zhou, J. Xu, A. Prorok, and A. P. Schoellig, "Learning to fly—a gym environment with pybullet physics for reinforcement learning of multi-agent quadcopter control," in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 7512–7519.

[26] J. Zeng, B. Zhang, and K. Sreenath, "Safety-critical model predictive control with discrete-time control barrier function," in *Proc. of the IEEE American Control Conference (ACC)*, 2021, pp. 3882–3889.

A Further Details on Methodology

A.1 LLM Interface

We use GPT-3.5 as the LLM to generate a time series of waypoints to choreograph the swarm of drones. To ensure an effective and generalizable prompt, we iteratively tune the initial prompt inspired by techniques from previous works like (12) and based on our empirical experience from the performance of different prompts. The prompt is constructed with three core ideas: creativity, safety, and generalization. Fig. 7 shows the GUI for Swarm-GPT for prompting and re-prompting. It consists of a repository of songs for users to select, a Chatbot to show the pre-generated prompts and user reprompts, and an accurate simulation for the swarm trajectories.

To make the choreography generated by GPT artistic, synchronized to music, and creative, we first assign GPT the role of an expert choreographer for a swarm of small drones. We use beat times to guide the music synchronization to mimic the human choreography process. The extracted timestamps of the beats of the song (gathered using Librosa) are fed into the prompt, where GPT is instructed to change the formation of the drone swarm at every beat by generating unique waypoints for each drone at each beat timestamp. We also prompt GPT to interpret the mood and tone of the well-recognized songs so that it can generate different types of choreography for different music genres. Keywords such as "harmonic", "symmetric", "artistic", and "synchronized" are also included in the prompt to drive GPT to generate aesthetic choreographs.

The second aspect of the prompt emphasizes safety. Although we are using AMSwarm as an additional safety layer, we still prompt GPT to handle some rudimentary collision avoidance and constraint violations. We input the following constraints into the prompt: physical limitations of the flight volume, maximum drone velocity and acceleration, and explicitly instruct GPT to avoid having two waypoints that are too close to each other at the same timestamp.

To make the interface more robust and easy to use for non-experts, we make the prompt generalizable by feeding in the initial positions of the drones detected from the configuration file and specifying the output format so that it could be seamlessly processed by the next module in the pipeline.

Finally, we instruct GPT to format its output as a series of waypoints for each drone. This waypoint format provides GPT full autonomy to generate a waypoint-based choreography creatively since it is not restricted to a specific library of motion primitives or maneuvers. This format is also easy for GPT to interact with and does not require domain-specific robotics knowledge that GPT may not be trained on or augmented with.

GPT is asked to represent its response as a string that can be easily parsed into a JSON object. This was found to be the simple and reliable way to extract numerical data from GPT. The outputted string is verified for correct formatting, with superfluous text discarded and parsed automatically into JSON format. The JSON object is then converted to an array of waypoints sorted by timestamp.

A.2 AMSwarm-Based Safety Filter Design

Waypoints generated by the LLM are not always feasible and require preliminary processing and checks before being processed by our formal collision-avoidance algorithm or deployed onto the quadrotors. To ensure that the waypoints generated by the LLM are within the limits of the permissible flying region, the raw waypoints are normalized to fit the volume whenever they violate positional constraints.

The output is then checked to ensure no simultaneously occurring waypoints are within a certain threshold distance of each other. If they are, these waypoints are pulled apart by a pre-selected offset, thus eliminating the collision. Without this step, the formal collision avoidance algorithm outlined in the next paragraph would begin in a state of collision and would not converge on a solution.

The formal collision avoidance features an alternating minimization approach known as AMSwarm (23) to process the LLM-generated waypoints. It receives the pre-processed waypoints and interpolates between them in a way that ensures the satisfaction of physical constraints while minimizing inter-agent collisions. Unlike other collision-avoidance methods that struggle with approximations and linearizations, this approach reformulates constraints in a polar form and applies an alternating minimization (AM) algorithm to solve the resultant optimization problem, all while maintaining the quadratic program (QP) structure (23).

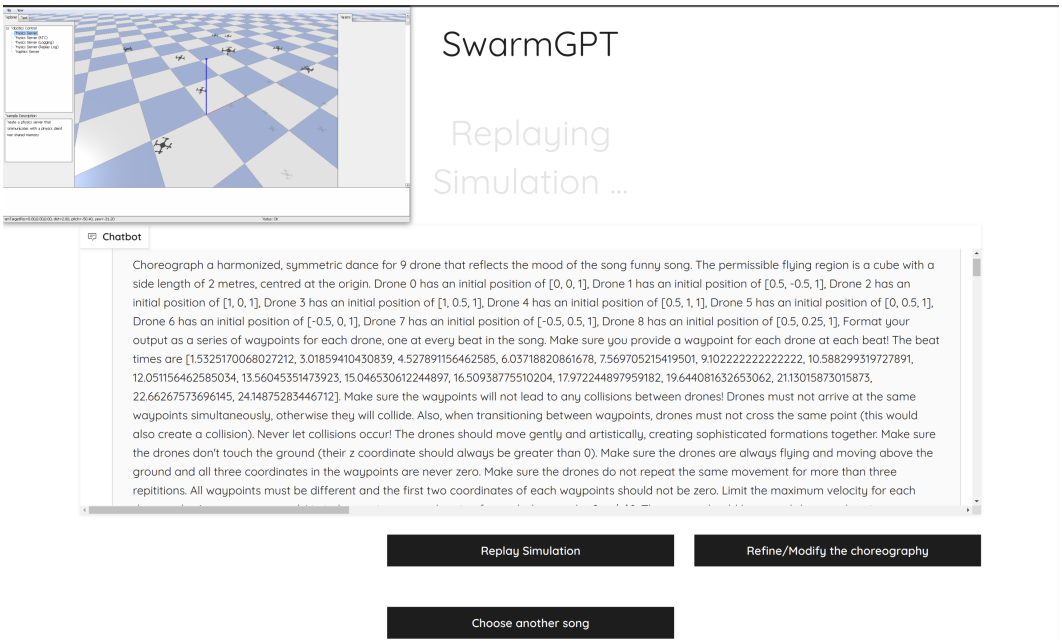

Figure 7: Illustration of the Swarm-GPT Graphical User Interface. The user starts by selecting a song from a given list. Then, the Chatbot shows the pre-generated prompt incorporating music features, including extracted beat times, mood, and genre. After the Chatbot generates the raw trajectory, the user is able to pass it through our safety filter (AMSwarm), and simulate the filtered trajectory. Re-prompting is available at this step for the user to modify the trajectory using high-level language inputs. Finally, the deployment button will establish communication with the drones and execute the planned trajectories on the swarm.

For a single drone i of the swarm size N, the initial and goal positions ($p_{i,o}$, $p_{i,g}$) are described with the three-dimensional position vector $p = [x, y, z]^T$ and are given by the LLM choreography. At each time step k, each drone i solves the following constrained optimization problem:

$$\min_{p_i} \quad \omega_g \sum_{k=K-\kappa}^{K-1} \left\| p_i[k] - p_{i,g} \right\|^2 + \omega_s \sum_{k=0}^{K-1} \left\| {p_i}^{(q)}[k] \right\|^2 \tag{1}$$

$$\text{subject to} \quad {p_i}^{(q)}[0] = p_{i,a}^{(q)},\ \forall q = \{0, 1, 2\}, \tag{2}$$

$$\underline{p} \preceq p_i[k] \preceq \overline{p},\ \forall k, \tag{3}$$

$$\left\| \dot{p}_i[k] \right\|^2 \leq \overline{v}^2,\ \forall k, \tag{4}$$

$$\underline{f}^2 \leq \left\| \ddot{p}_i[k] + g \right\|^2 \leq \overline{f}^2,\ \forall k, \tag{5}$$

$$h_{ij}[k] = \left\| \Theta_{ij}^{-1}(p_i[k] - \xi_j[k]) \right\|^2 - 1 \geq 0,\ \forall k, j, \tag{6}$$

where K denotes the planning horizon length, $\|\cdot\|$ the Euclidean norm, and the superscript (q) the q-th time derivative of a variable. There are three components in the trajectory optimization problem, which are respectively described below:

1) Cost Function: The cost function in Eq. 1 consists of two terms weighted by the constants ω_g and ω_s. The former term, i.e., the goal cost, penalizes the deviation of the position of the drone from the specified goal position generated by the LLM over the last $\kappa < K$ steps in the prediction horizon. The latter, i.e., the smoothness cost, penalizes the q-th derivatives of the position trajectory.

2) Initialization and Feasibility Constraints: The equality constraints in Eq. 2 initialize the trajectory position and its higher derivatives for each drone. The conditions in Eq. 3-5 enforce lower and upper boundaries on the position $(\underline{p}, \overline{p})$, velocity $(-\underline{v}, \overline{v})$, and acceleration $(\underline{f}, \overline{f})$.

3) Collision Avoidance Constraints: Eq. 6 guarantees the avoidance of collision with either the j-th neighbouring drone (or an obstacle) at the position $\xi_j[k]$. The diagonal matrix Θ_{ij} consists of elements whose scalars characterize the lengths of the ellipsoidal envelopes for each axis around the neighbouring drone (or the obstacles). The vector g denotes the gravitational acceleration. We can further extend the collision avoidance by using barrier function (BF) constraints to introduce different levels of safe behaviours (26):

$$h_{ij}[k] - h_{ij}[k-1] \geq -\gamma\, h_{ij}[k-1],\ \forall k, j, \tag{7}$$

where $\gamma \in [0, 1]$ controls how fast each drone is allowed to approach the constraint boundary given by $h_{ij} = 0$. We can set $\gamma = 1$ to restore the condition in Eq. 6; smaller values generally promote more gradual and conservative collision avoidance behaviour.

This approach involves iteratively optimizing trajectory parameters while ensuring that various constraints, such as collision avoidance, position constraints, and kinematic feasibility, are satisfied. The AM algorithm switches between optimizing different aspects of the trajectory, ensuring that each step adheres to the imposed constraints. This process produces trajectories that strike a balance between safety and efficiency.

This algorithm was selected due to its consistent and reliable performance in complex and dynamic environments as well as its scalability properties (23). Overall, the AMSwarm-based safety filter permits the choreography of larger drone swarms and future-proofs of our system to behave robustly within dense, interactive environments. The algorithm also interfaces well with our LLM outputs (i.e. waypoints). Other approaches require more information about the desired drone motions (for example, sinusoidal motion primitives), which can be difficult for language models to generate.

Our application of this algorithm features some minor changes. A new optimization problem is formulated for each inter-waypoint time window, and the alternating-minimization algorithm is re-applied for each one. The position control commands generated by optimization are then resampled at both the desired simulation and control frequencies prior to deployment.

The re-sampled position commands are simulated via gym-pybullet-drones (25) as an on-demand, visual confirmation of collision-free motions.

B Additional Evaluation Results

In this appendix, we present additional evaluation results for the proposed Swarm-GPT system.

Assessment of Sim-to-Real Transfer. Figure 8 illustrates the close match between the simulated and real trajectory for one drone during a choreography. This corresponds to the low RMSE values we obtained in Fig. 3; this behaviour is generally observed across different drones and choreographies.

Formation Synchronization to Music. Figure 9 shows a synchronized motion among a swarm of 6 drones during two more beat intervals (in addition to that presented in Fig. 5). These results demonstrate the capability of Swarm-GPT to generate creative and expressive drone formations in synchronization with the music features.

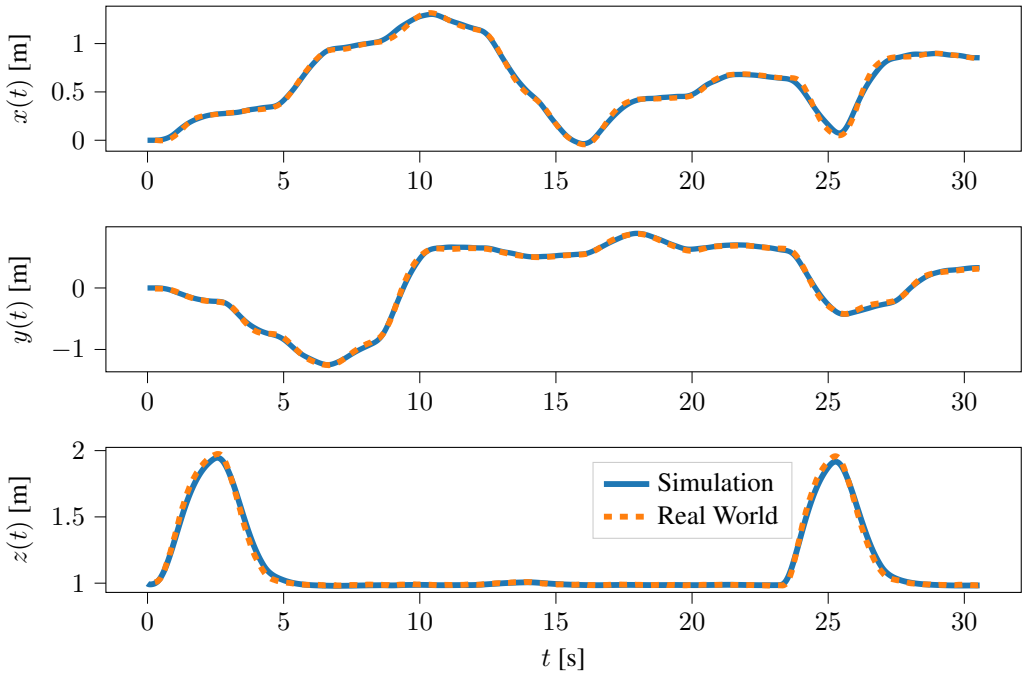

Figure 8: The simulated and actual position trajectories of a single drone. The simulated agent and the physical drone were given the same reference inputs. As can be seen from the plots, the actual positions of the drone closely match that of the simulated response.

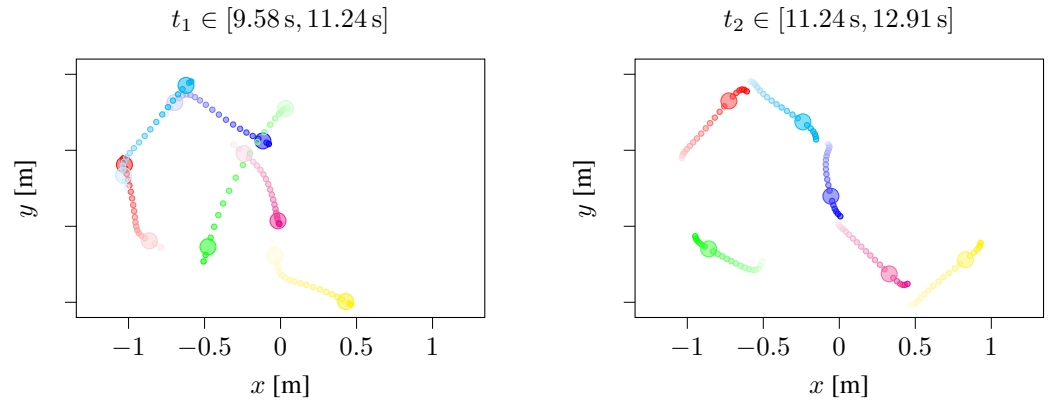

Figure 9: A six-drone swarm 2D trajectory visualizations during one musical beat between 9.58 and 11.24 s *(left)* and between 11.24 and 12.91 s *(right)*. Similar to Fig. 5, different colours correspond to different drones, the large dots represent beat times, and the colour gradient moves from lighter to more saturated as time progresses. These plots showcase the generative capability of the Swarm-GPT for creating interesting formations synchronized with music beats while complying with desired safety constraints.