Article

Outracing champion Gran Turismo drivers with deep reinforcement learning

Peter R. Wurman[1✉], Samuel Barrett[1], Kenta Kawamoto[2], James MacGlashan[1], Kaushik Subramanian[3], Thomas J. Walsh[1], Roberto Capobianco[3], Alisa Devlic[3], Franziska Eckert[3], Florian Fuchs[3], Leilani Gilpin[1], Piyush Khandelwal[1], Varun Kompella[1], HaoChih Lin[3], Patrick MacAlpine[1], Declan Oller[1], Takuma Seno[2], Craig Sherstan[1], Michael D. Thomure[1], Houmehr Aghabozorgi[1], Leon Barrett[1], Rory Douglas[1], Dion Whitehead[1], Peter Dürr[3], Peter Stone[1], Michael Spranger[2] & Hiroaki Kitano[2]

Many potential applications of artificial intelligence involve making real-time decisions in physical systems while interacting with humans. Automobile racing represents an extreme example of these conditions; drivers must execute complex tactical manoeuvres to pass or block opponents while operating their vehicles at their traction limits[1]. Racing simulations, such as the PlayStation game Gran Turismo, faithfully reproduce the non-linear control challenges of real race cars while also encapsulating the complex multi-agent interactions. Here we describe how we trained agents for Gran Turismo that can compete with the world's best e-sports drivers. We combine state-of-the-art, model-free, deep reinforcement learning algorithms with mixed-scenario training to learn an integrated control policy that combines exceptional speed with impressive tactics. In addition, we construct a reward function that enables the agent to be competitive while adhering to racing's important, but under-specified, sportsmanship rules. We demonstrate the capabilities of our agent, Gran Turismo Sophy, by winning a head-to-head competition against four of the world's best Gran Turismo drivers. By describing how we trained championship-level racers, we demonstrate the possibilities and challenges of using these techniques to control complex dynamical systems in domains where agents must respect imprecisely defined human norms.

Deep reinforcement learning (deep RL) has been a key component of impressive recent artificial intelligence (AI) milestones in domains such as Atari[2], Go[3,4], StarCraft[5] and Dota[6]. For deep RL to have an influence on robotics and automation, researchers must demonstrate success in controlling complex physical systems. In addition, many potential applications of robotics require interacting in close proximity to humans while respecting imprecisely specified human norms. Automobile racing is a domain that poses exactly these challenges; it requires real-time control of vehicles with complex, non-linear dynamics while operating within inches of opponents. Fortunately, it is also a domain for which highly realistic simulations exist, making it amenable to experimentation with machine-learning approaches.

Research on autonomous racing has accelerated in recent years, leveraging full-sized[7–10], scale[11–15] and simulated[16–25] vehicles. A common approach pre-computes trajectories[26,27] and uses model predictive control to execute those trajectories[7,28]. However, when driving at the absolute limits of friction, small modelling errors can be catastrophic. Racing against other drivers puts even greater demands on modelling accuracy, introduces complex aerodynamic interactions and further requires engineers to design control schemes that continuously predict and adapt to the trajectories of other

cars. Racing with real driverless vehicles still seems to be several years away, as the recent Indy Autonomous Challenge curtailed its planned head-to-head competition to time trials and simple obstacle avoidance[29].

Researchers have explored various ways to use machine learning to avoid this modelling complexity, including using supervised learning to model vehicle dynamics[8,12,30] and using imitation learning[31], evolutionary approaches[32] or reinforcement learning[16,21] to learn driving policies. Although some studies achieved super-human performance in solo driving[24] or progressed to simple passing scenarios[16,20,25,33], none have tackled racing at the highest levels.

To be successful, racers must become highly skilled in four areas: (1) race-car control, (2) racing tactics, (3) racing etiquette and (4) racing strategy. To control the car, drivers develop a detailed understanding of the dynamics of their vehicle and the idiosyncrasies of the track on which they are racing. On this foundation, drivers build the tactical skills needed to pass and defend against opponents, executing precise manoeuvres at high speed with little margin for error. At the same time, drivers must conform to highly refined, but imprecisely specified, sportsmanship rules. Finally, drivers use strategic thinking when modelling opponents and deciding when and how to attempt a pass.
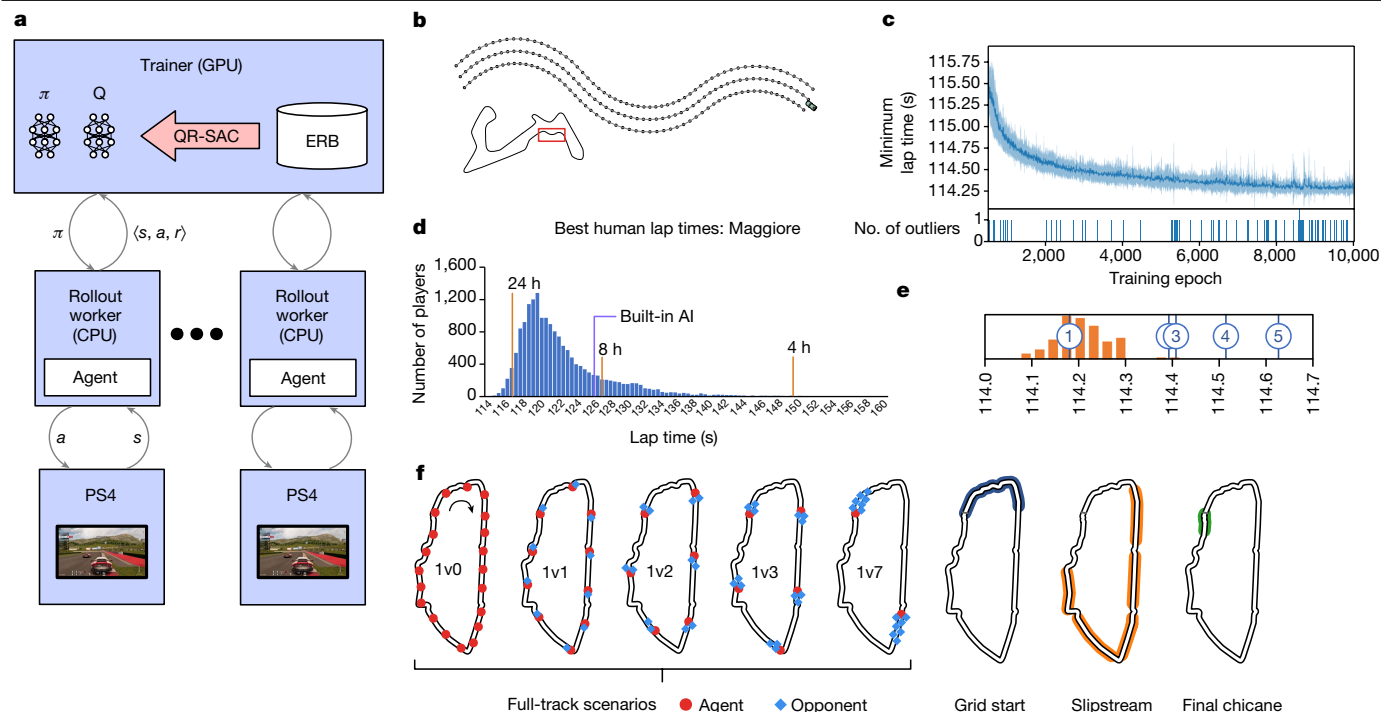
Fig. 1 | Training. a, An example training configuration. The trainer distributes training scenarios to rollout workers, each of which controls one PS4 running an instance of GT. The agent in the worker runs one copy of the most recent policy, π, to control up to 20 cars. The agent sends an action, a, for each car it controls to the game. Asynchronously, the game computes the next frames and sends each new state, s, to the agent. When the game reports that the action has been registered, the agent reports the state, action, reward tuple $\langle s, a, r \rangle$ to the trainer, which stores it in the ERB. The trainer samples the ERB to update the policy, π, and Q-function networks. **b**, The course representation ahead of the car on a sequence of curves on Maggiore if the car was travelling at 200 km h^{-1}. **c**, The distribution of the learning curves on Maggiore from 15 different random seeds. All of the seeds reached superhuman performance. Most reached it in 10 days of training, whereas the longest took 25 days. **d**, The distribution of individual players' best lap times on Maggiore as recorded on Kudos Prime (https://www.kudosprime.com/gts/rankings.php?sec=daily). Superimposed on **d** is the number of hours that GT Sophy, using ten PlayStations with 20 cars each, needed to achieve similar performance. **e**, A histogram (in orange) of 100 laps from the time-trial policy GT Sophy used on 2 July 2021 compared with the five best human drivers' best lap times (circles 1–5) in the Kudos Prime data. Similar graphs for the other two tracks are in the Supplementary Information; Maggiore is the only one of the three tracks on which the best human performance was close to GT Sophy. **f**, The training scenarios on Sarthe, including five full-track configurations in which the agent starts with zero, one, two, three or seven nearby opponents and three specialized scenarios that are limited to the shaded regions. The actual track positions, opponents and relative car arrangements are varied to ensure that the learned skills are robust.

In this article, we describe how we used model-free, off-policy deep RL to build a champion-level racing agent, which we call Gran Turismo Sophy (GT Sophy). GT Sophy was developed to compete with the world's best players of the highly realistic PlayStation 4 (PS4) game Gran Turismo (GT) Sport (https://www.gran-turismo.com/us/), developed by Polyphony Digital, Inc. We demonstrate GT Sophy by competing against top human drivers on three car and track combinations that posed different racing challenges. The car used on the first track, Dragon Trail Seaside (Seaside), was a high-performance road vehicle. On the second track, Lago Maggiore GP (Maggiore), the vehicle was equivalent to the Federation Internationale de l'Automobile (FIA) GT3 class of race cars. The third and final race took place on the Circuit de la Sarthe (Sarthe), famous as the home of the 24 Hours of Le Mans. This race featured the Red Bull X2019 Competition race car, which can reach speeds in excess of 300 km h^{-1}. Although lacking strategic savvy, in the process of winning the races against humans, GT Sophy demonstrated notable advances in the first three of the four skill areas mentioned above.

Approach

The training configuration is illustrated in Fig. 1a. GT runs only on PlayStations, which necessitated that the agent runs on a separate computing device and communicates asynchronously with the game by means of TCP. Although GT ran only in real time, each GT Sophy instance controlled up to 20 cars on its PlayStation, which accelerated data collection. We typically trained GT Sophy from scratch using 10–20 PlayStations, an equal number of compute instances and a GPU machine that asynchronously updates the neural networks.

The core actions of the agent were mapped to two continuous-valued dimensions: changing velocity (accelerating or braking) and steering (left or right). The effect of the actions was enforced by the game to be consistent with the physics of the environment; GT Sophy cannot brake harder than humans but it can learn more precisely when to brake. GT Sophy interacted with the game at 10 Hz, which we claim does not give GT Sophy a particular advantage over professional gamers[34] or athletes[35].

As is common[26,27], the agent was given a static map defining the left and right edges and the centre line of the track. We encoded the approaching course segment as 60 equally spaced 3D points along each edge of the track and the centre line (Fig. 1b). The span of the points in any given observation was a function of the current velocity, so as to always represent approximately the next 6 s of travel. The points were computed from the track map and presented to the neural network in the egocentric frame of reference of the agent.

Through an API, GT Sophy observed the positions, velocities, accelerations and other relevant state information about itself and all opponents. To make opponent information amenable to deep learning, GT
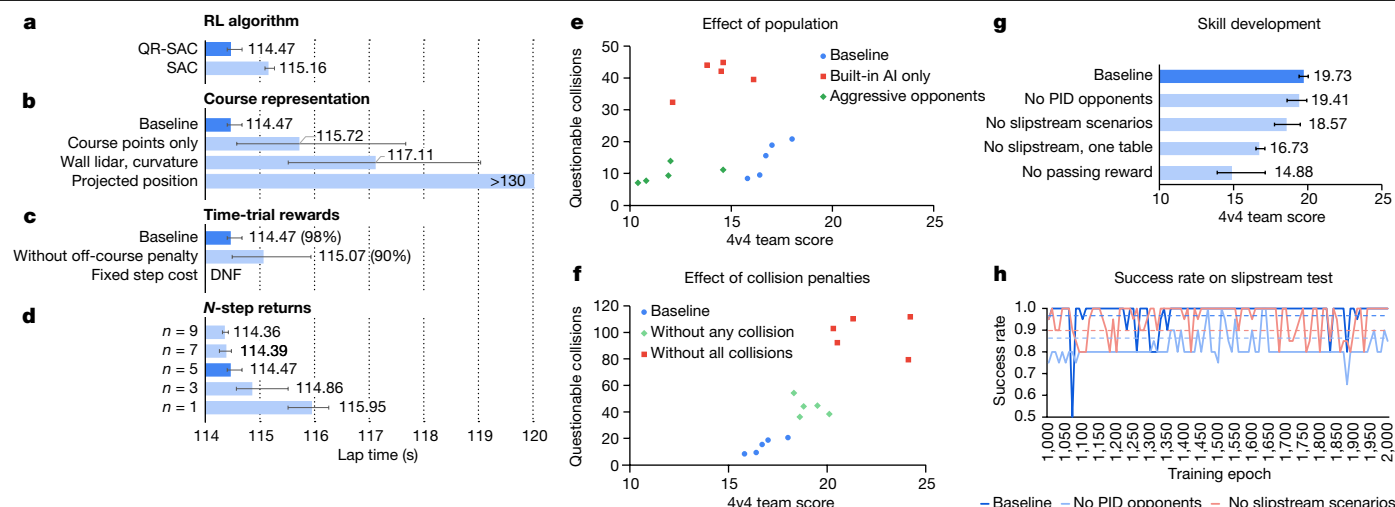
a RL algorithm
QR-SAC ├─ 114.47
SAC ├─ 115.16

b Course representation
Baseline ├─ 114.47
Course points only ├─ 115.72
Wall lidar, curvature ├─ 117.11
Projected position >130

c Time-trial rewards
Baseline ├─ 114.47 (98%)
Without off-course penalty ├─ 115.07 (90%)
Fixed step cost DNF

d N-step returns
n = 9 ├─ 114.36
n = 7 ├─ 114.39
n = 5 ├─ 114.47
n = 3 ├─ 114.86
n = 1 ├─ 115.95
Lap time (s)

e Effect of population
• Baseline
■ Built-in AI only
♦ Aggressive opponents
Questionable collisions vs 4v4 team score

f Effect of collision penalties
• Baseline
♦ Without any collision
■ Without all collisions
Questionable collisions vs 4v4 team score

g Skill development
Baseline ├─ 19.73
No PID opponents ├─ 19.41
No slipstream scenarios ├─ 18.57
No slipstream, one table ├─ 16.73
No passing reward ├─ 14.88
4v4 team score

h Success rate on slipstream test
Success rate vs Training epoch
— Baseline — No PID opponents — No slipstream scenarios

Fig. 2 | Ablations. a–d, The effect of various settings on Maggiore performance using the 2,048 × 2 time trial network from 2 July 2021. All bars represent the average across five initial seeds, with the full range of the samples shown as an error bar. In all graphs, the baseline settings are coloured in a darker shade of blue. **a**, GT Sophy would not be faster than the best human on Maggiore without the QR enhancement to SAC. **b**, Representing the upcoming track as sequences of points was advantageous. **c**, Not including the off-course penalty results in a slower lap time and (in parentheses) a much lower percentage of laps without exceeding the course boundaries. **d**, Notably, the 5-step return used on 2 July 2021 was not the best choice; this was changed to a 7-step return for the October match. **e–h**, Evaluation of the 2,048 × 4 networks and configurations used to train the version of GT Sophy that raced on 21 October 2021. In **e** and **f**, each point represents the average of ten 7-lap 4v4 races on Sarthe against copies of October GT Sophy and comparison is shown for the trade-offs between team score and 'questionable collisions' (a rough indication of possible penalties). **e**, When GT Sophy trained against only the built-in AI, it learned to be too aggressive, and when it trained against an aggressive opponent, it lost its competitive edge. **f**, As elements of the collision penalties are removed from GT Sophy's reward function, it becomes notably more aggressive. The test drivers and stewards judged the non-baseline policies to be much too unsportsmanlike. **g**, To make the importance of the features evaluated clearer, we tested these variations against a slightly less competitive version of GT Sophy. The results show the importance of the scenario training, using several ERBs and having a passing reward. **h**, An ablation of elements of the slipstream training over a range of epochs sampled during training. The y axis measures the agent's ability to pass a particular slipstream test. The solid lines represent the performance of one seed in each condition and the dotted lines represent the mean of five seeds over all epochs. Note that the agent's ability to apply the skill fluctuates even in the best (baseline) case because of the changing characteristics of the replay buffer.

Sophy maintained two lists of their state features: one for cars in front of the agent and one for cars behind. Both lists were ordered from closest to farthest and limited by a maximum range.

We trained GT Sophy using a new deep RL algorithm we call quantile regression soft actor-critic (QR-SAC). This approach learns a policy (actor) that selects an action on the basis of the agent's observations and a value function (critic) that estimates the future rewards of each possible action. QR-SAC extends the soft actor-critic approach[36,37] by modifying it to handle N-step returns[38] and replacing the expected value of future rewards with a representation of the probability distributions of those rewards[39]. QR-SAC trains the neural networks asynchronously; it samples data from an experience replay buffer (ERB)[40], while actors simultaneously practice driving using the most recent policy and continuously fill the buffer with their new experiences.

The agent was given a progress reward[24] for the speed with which it advanced around the track and penalties if it went out of bounds, hit a wall or lost traction. These shaping rewards allowed the agent to quickly receive positive feedback for staying on the track and driving fast. Notably, GT Sophy learned to get around the track in only a few hours and learned to be faster than 95% of the humans in our reference dataset (Kudos Prime, https://www.kudosprime.com/gts/rankings.php?sec=daily) within a day or two. However, as shown in Fig. 1c, it trained for another nine or more days—accumulating more than 45,000 driving hours—shaving off tenths of seconds, until its lap times stopped improving. With this training procedure, GT Sophy achieved superhuman time-trial performance on all three tracks. Figure 1d shows the distribution of the best single lap times for more than 17,700 players all driving the same car on Maggiore (the track with the smallest gap between GT Sophy and the humans). Figure 1e shows how consistent GT Sophy's lap times were, with a mean lap time about equal to the single best recorded human lap time.

The progress reward alone was not enough to incentivize the agent to win the race. If the opponent was fast enough, the agent would learn to follow it and accumulate large rewards without risking potentially catastrophic collisions. As in previous work[25], adding rewards specifically for passing helped the agent learn to overtake other cars. We used a passing reward that was proportional to the distance by which the agent improved its position relative to each opponent within the local region. The reward was symmetric; if an opponent gained ground on the agent, the agent would see a proportional negative reward.

Like many other sports, racing—both physical and virtual—requires human judges. These stewards immediately review racing 'incidents' and make decisions about which drivers, if any, receive penalties. A car with a penalty is forced by the game engine to slow down to 100 km h⁻¹ in certain penalty zones on the track for the penalty duration. Although a small amount of unintentional car-to-car contact is fairly common and considered acceptable, racing rules describe a variety of conditions under which drivers may be penalized. The rules are ambiguous and stewards' judgements incorporate a lot of context, such as the effect the contact has on the immediate future of the cars involved. The fact that judges' decisions are subjective and contextual makes it difficult to encode these rules in a way that gives the agent clear signals from which to learn. Racing etiquette is an example of the challenges that AI practitioners face when designing agents that interact with humans who expect those agents to conform to behavioural norms[41].

The observations that the agent receives from the game include a flag when car contact occurs but does not indicate whether a penalty was deserved. We experimented with several approaches to encode etiquette as instantaneous penalties on the basis of situational analysis of the collisions. However, as we tried to more accurately model blame
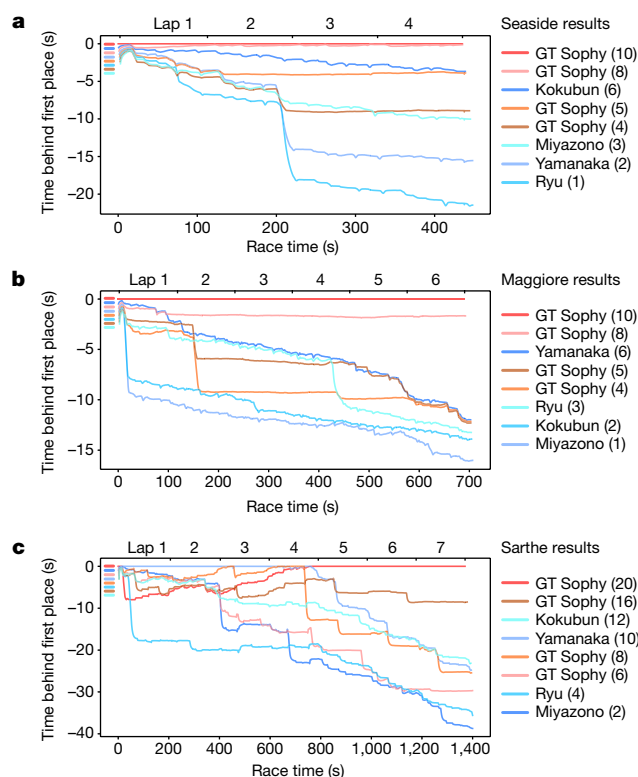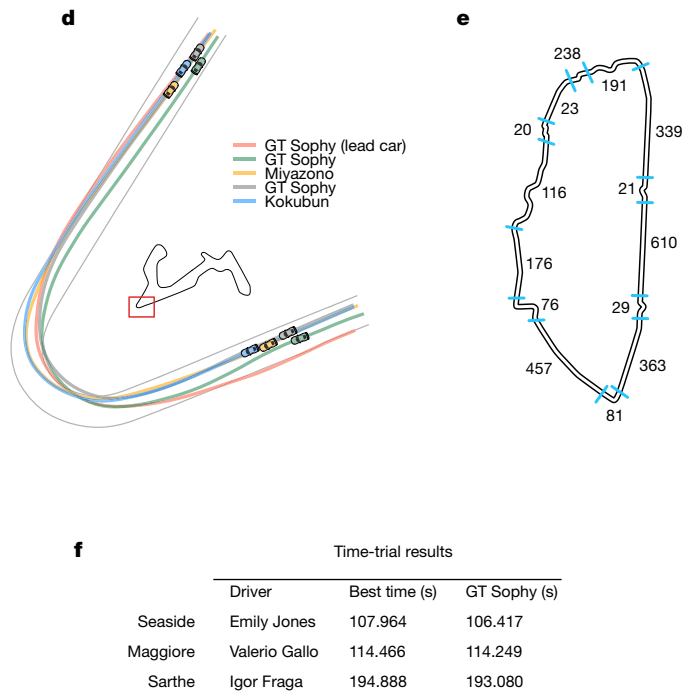
Fig. 3 | Results. a–c, How each race unfolded on Seaside (**a**), Maggiore (**b**) and Sarthe (**c**). The distance from the leader is computed as the time since the lead car passed the same position on the track. The legend for each race shows the final places and, in parentheses, the points for each driver. These charts clearly show how, once GT Sophy obtained a small lead, the human drivers could not catch it. The sharp decreases represent either a driver losing control or paying a penalty for either exceeding the course bounds or colliding with another driver. Sarthe (**c**) had the most incidents, with GT Sophy receiving two penalties for excessive contact and the humans receiving one penalty and two warnings. Both the humans and GT Sophy also had several smaller penalties for exceeding the course boundaries, particularly in the final chicane sequence. **d,** An

example from the 2 July 2021 race in which two instances of GT Sophy (grey and green) passed two humans (yellow and blue) on a corner on Maggiore. As a reference, the trajectory of the lead GT Sophy car when taking the corner alone is shown in red. The example clearly illustrates that the trajectory of GT Sophy through the corner is contextual; even though the human drivers tried to protect the inside going into the corner, GT Sophy was able to find two different, faster trajectories. **e,** The number of passes that occurred on different parts of Sarthe in 100 4v4 races between two GT Sophy policies, demonstrating that the agent has learned to pass on many parts of the track. **f,** Results from the time-trial competition in July 2021.

assignment, the resulting policies were judged much too aggressive by stewards and test drivers. For the final races, we opted for a conservative approach that penalized the agent for any collision in which it was involved (regardless of fault), with some extra penalties if the collision was likely considered unacceptable. Figure 2a–h isolates the effects of collision penalties and other key design choices made during this project.

Although many applications of RL to games use self-play to improve performance[3,42], the straightforward application of self-play was inadequate in this setting. For example, as a human enters a difficult corner, they may brake a fraction of a second earlier than the agent would. Even a small bump at the wrong moment can cause an opponent to lose control of their car. By racing against only copies of itself, the agent was ill-prepared for the imprecision it would see with human opponents. If the agent following does not anticipate the possibility of the opponent braking early, it will not be able to avoid rear-ending the human driver and will be assessed a penalty. This feature of racing—that one player's suboptimal choice causes the other player to be penalized—is not a feature of zero-sum games such as Go and chess. To alleviate this issue, we used a mixed population of opponents, including agents curated from previous experiments and the game's (relatively slower) built-in AI. Figure 2e shows the importance of these choices.

In addition, the opportunities to learn certain skills are rare. We call this the exposure problem; certain states of the world are not accessible to the agent without the 'cooperation' of its opponents. For example, to

execute a slingshot pass, a car must be in the slipstream of an opponent on a long straightaway, a condition that may occur naturally a few times or not at all in an entire race. If that opponent always drives only on the right, the agent will learn to pass only on the left and would be easily foiled by a human who chose to drive on the left. To address this issue, we developed a process that we called mixed-scenario training. We worked with a retired competitive GT driver to identify a small number of race situations that were probably pivotal on each track. We then configured scenarios that presented the agent with noisy variations of those critical situations. In slipstream passing scenarios, we used simple PID controllers to ensure that the opponents followed certain trajectories, such as driving on the left, that we wanted our agent to be prepared for. Figure 1f shows the full-track and specialized scenarios for Sarthe. Notably, all scenarios were present throughout the training regime; no sequential curriculum was needed. We used a form of stratified sampling[43] to ensure that situational diversity was present throughout training. Figure 2h shows that this technique resulted in more robust skills being learned.

Evaluation

To evaluate GT Sophy, we raced the agent in two events against top GT drivers. The first event was on 2 July 2021 and involved both time-trial and head-to-head races. In the time-trial race, three of the world's top

drivers were asked to try to beat the lap times of GT Sophy. Although the human drivers were allowed to see a 'ghost' of GT Sophy as they drove around the track, GT Sophy won all three matches. The results are shown in Fig. 3f.

The head-to-head race was held at the headquarters of Polyphony Digital and, although limited to top Japanese players due to pandemic travel restrictions, included four of the world's best GT drivers. These drivers formed a team to compete against four instances of GT Sophy. Points were awarded to the team on the basis of the final positions (10, 8, 6, 5, 4, 3, 2 and 1, from first to last), with Sarthe, the final and most challenging race, counting double. Each team started in either the odd or the even positions on the basis of their best qualifying time. The human drivers won the team event on 2 July 2021 by a score of 86–70.

After examining GT Sophy's 2 July 2021 performance, we improved the training regime, increased the network size, made small modifications to some features and rewards and improved the population of opponents. GT Sophy handily won the rematch held on 21 October 2021 by an overall team score of 104–52. Starting in the odd positions, team GT Sophy improved four spots on Seaside and Maggiore and two on Sarthe. Figure 3a–c shows the relative positions of the cars through each race and the points earned by each individual.

One of the advantages of using deep RL to develop a racing agent is that it eliminates the need for engineers to program how and when to execute the skills needed to win the race—as long as it is exposed to the right conditions, the agent learns to do the right thing by trial and error. We observed that GT Sophy was able to perform several types of corner passing, use the slipstream effectively, disrupt the draft of a following car, block, and execute emergency manoeuvres. Figure 3d shows particularly compelling evidence of GT Sophy's generalized tactical competence. The diagram illustrates a situation from the 2 July 2021 event in which two GT Sophy cars both pass two human cars on a single corner on Maggiore. This kind of tactical competence was not limited to any particular part of the course. Figure 3e shows the number of passes that occurred on different sections of Sarthe from 100 4v4 races between two different GT Sophy policies. Although slipstream passing on the straightaways was most common, the results show that GT Sophy was able to take advantage of passing opportunities on many different sections of Sarthe.

Although GT Sophy demonstrated enough tactical skill to beat expert humans in head-to-head racing, there are many areas for improvement, particularly in strategic decision-making. For example, GT Sophy takes the first opportunity to pass on a straightaway, sometimes leaving enough room on the same stretch of track for the opponent to use the slipstream to pass back. GT Sophy also aggressively tries to pass an opponent with a looming penalty, whereas a strategic human driver may wait and make the easy pass when the opponent is forced to slow down.

Conclusions

Simulated automobile racing is a domain that requires real-time, continuous control in an environment with highly realistic, complex physics. The success of GT Sophy in this environment shows, for the first time, that it is possible to train AI agents that are better than the top human racers across a range of car and track types. This result can be seen as another important step in the continuing progression of competitive tasks that computers can beat the very best people at, such as chess, Go, Jeopardy, poker and StarCraft. In the context of previous landmarks of this kind, GT Sophy is the first that deals with head-to-head, competitive, high-speed racing, which requires advanced tactics and subtle sportsmanship considerations. Agents such as GT Sophy have the potential to make racing games more enjoyable, provide realistic, high-level competition for training professional drivers and discover new racing techniques. The success of deep RL in this environment suggests that these techniques may soon have an effect on real-world systems such as collaborative robotics, aerial drones or autonomous vehicles.

Online content

1. Milliken, W. F. et al. *Race Car Vehicle Dynamics* Vol. 400 (Society of Automotive Engineers, 1995).
2. Mnih, V. et al. Playing Atari with deep reinforcement learning. Preprint at https://arxiv.org/abs/1312.5602 (2013).
3. Silver, D. et al. Mastering the game of Go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).
4. Silver, D. et al. Mastering the game of Go without human knowledge. *Nature* **550**, 354–359 (2017).
5. Vinyals, O. et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* **575**, 350–354 (2019).
6. Berner, C. et al. Dota 2 with large scale deep reinforcement learning. Preprint at https://arxiv.org/abs/1912.06680 (2019).
7. Laurense, V. A., Goh, J. Y. & Gerdes, J. C. In *2017 American Control Conference (ACC)* 5586–5591 (IEEE, 2017).
8. Spielberg, N. A., Brown, M., Kapania, N. R., Kegelman, J. C. & Gerdes, J. C. Neural network vehicle models for high-performance automated driving. *Sci. Robot.* **4**, eaaw1975 (2019).
9. Burke, K. Data makes it beta: Roborace returns for second season with updateable self-driving vehicles powered by NVIDIA DRIVE. *The Official NVIDIA Blog* https://blogs.nvidia.com/blog/2020/10/29/roborace-second-season-nvidia-drive/ (2020).
10. Leporati, G. No driver? no problem—this is the Indy Autonomous Challenge. *Ars Technica* https://arstechnica.com/cars/2021/07/a-science-fair-or-the-future-of-racing-the-indy-autonomous-challenge/ (2021).
11. Williams, G., Drews, P., Goldfain, B., Rehg, J. M. & Theodorou, E. A. In *2016 IEEE International Conference on Robotics and Automation (ICRA)* 1433–1440 (IEEE, 2016).
12. Williams, G., Drews, P., Goldfain, B., Rehg, J. M. & Theodorou, E. A. Information-theoretic model predictive control: theory and applications to autonomous driving. *IEEE Trans. Robot.* **34**, 1603–1622 (2018).
13. Pan, Y. et al. In *Proc. Robotics: Science and Systems XIV* (eds Kress-Gazit, H., Srinivasa, S., Howard, T. & Atanasov, N.) https://doi.org/10.15607/RSS.2018.XIV.056 (Carnegie Mellon Univ., 2018).
14. Pan, Y. et al. Imitation learning for agile autonomous driving. *Int. J. Robot. Res.* **39**, 286–302 (2020).
15. Amazon Web Services. AWS DeepRacer League. https://aws.amazon.com/deepracer/league/ (2019).
16. Pyeatt, L. D. & Howe, A. E. Learning to race: experiments with a simulated race car. In *Proc. Eleventh International FLAIRS Conference* 357–361 (AAAI, 1998).
17. Chaperot, B. & Fyfe, C. In *2006 IEEE Symposium on Computational Intelligence and Games* 181–186 (IEEE, 2006).
18. Cardamone, L., Loiacono, D. & Lanzi, P. L. In *Proc. 11th Annual Conference on Genetic and Evolutionary Computation* 1179–1186 (ACM, 2009).
19. Cardamone, L., Loiacono, D. & Lanzi, P. L. In *2009 IEEE Congress on Evolutionary Computation* 2622–2629 (IEEE, 2009).
20. Loiacono, D., Prete, A., Lanzi, L. & Cardamone, L. In *IEEE Congress on Evolutionary Computation* 1–8 (IEEE, 2010).
21. Jaritz, M., de Charette, R., Toromanoff, M., Perot, E. & Nashashibi, F. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* 2070–2075 (IEEE, 2018).
22. Weiss, T. & Behl, M. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)* 1163–1168 (IEEE, 2020).
23. Weiss, T., Babu, V. S. & Behl, M. In *NeurIPS 2020 Workshop on Machine Learning for Autonomous Driving* (NeurIPS, 2020).
24. Fuchs, F., Song, Y., Kaufmann, E., Scaramuzza, D. & Dürr, P. Super-human performance in Gran Turismo Sport using deep reinforcement learning. *IEEE Robot. Autom. Lett.* **6**, 4257–4264 (2021).
25. Song, Y., Lin, H., Kaufmann, E., Dürr, P. & Scaramuzza, D. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2021).
26. Theodosis, P. A. & Gerdes, J. C. In *Dynamic Systems and Control Conference* Vol. 45295, 235–241 (American Society of Mechanical Engineers, 2012).
27. Funke, J. et al. In *2012 IEEE Intelligent Vehicles Symposium* 541–547 (IEEE, 2012).

Article

28. Kritayakirana, K. & Gerdes, J. C. Autonomous vehicle control at the limits of handling. *Int. J. Veh. Auton. Syst.* **10**, 271–296 (2012).

29. Bonkowski, J. Here's what you missed from the Indy Autonomous Challenge main event. *Autoweek* https://www.autoweek.com/racing/more-racing/a38069263/what-missed-indy-autonomous-challenge-main-event/ (2021).

30. Rutherford, S. J. & Cole, D. J. Modelling nonlinear vehicle dynamics with neural networks. *Int. J. Veh. Des.* **53**, 260–287 (2010).

31. Pomerleau, D. A. In *Robot Learning* (eds Connell, J. H. & Mahadevan, S.) 19–43 (Springer, 1993).

32. Togelius, J. & Lucas, S. M. In *2006 IEEE International Conference on Evolutionary Computation* 1187–1194 (IEEE, 2006).

33. Schwarting, W. et al. Deep latent competition: learning to race using visual control policies in latent space. Preprint at https://arxiv.org/abs/2102.09812 (2021).

34. Gozli, D. G., Bavelier, D. & Pratt, J. The effect of action video game playing on sensorimotor learning: evidence from a movement tracking task. *Hum. Mov. Sci.* **38**, 152–162 (2014).

35. Davids, K., Williams, A. M. & Williams, J. G. *Visual Perception and Action in Sport* (Routledge, 2005).

36. Haarnoja, T., Zhou, A., Abbeel, P. & Levine, S. In *Proc. 35th International Conference on Machine Learning* 1856–1865 (PMLR, 2018).

37. Haarnoja, T. et al. Soft actor-critic algorithms and applications. Preprint at https://arxiv.org/abs/1812.05905 (2018).

38. Mnih, V. et al. In *Proc. 33rd International Conference on Machine Learning* 1928–1937 (PMLR, 2016).

39. Dabney, W., Rowland, M., Bellemare, M. G. & Munos, R. In *32nd AAAI Conference on Artificial Intelligence* (AAAI, 2018).

40. Lin, L.-J. *Reinforcement Learning for Robots Using Neural Networks*. Dissertation, Carnegie Mellon Univ. (1993).

41. Siu, H. C. et al. Evaluation of human-AI teams for learned and rule-based agents in Hanabi. Preprint at https://arxiv.org/abs/2107.07630 (2021).

42. Tesauro, G. TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Comput.* **6**, 215–219 (1994).

43. Devore, J. L. *Probability and Statistics for Engineering and the Sciences* 6th edn (Brooks/Cole, 2004).

Methods

Game environment

Since its debut in 1997, the GT franchise has sold more than 80 million units. The most recent release, Gran Turismo Sport, is known for precise vehicle dynamics simulation and racing realism, earning it the distinction of being sanctioned by the FIA and selected as a platform for the first Virtual Olympics (https://olympics.com/en/sport-events/olympic-virtual-motorsport-event/). GT Sport runs only on PS4s and at a 60-Hz-dynamics simulation cycle. A maximum of 20 cars can be in a race.

Our agent ran asynchronously on a separate computer and communicated with the game by means of HTTP over wired Ethernet. The agent requested the latest observation and made decisions at 10 Hz (every 100 ms). We tested action frequencies from 5 Hz to 60 Hz and found no substantial performance gains from acting more frequently than 10 Hz. The agent had to be robust to the infrequent, but real, networking delays. The agent's action was treated the same as a human's game controller input, but only a subset of action capabilities were supported in the GT API. For example, the API did not allow the agent to control gear shifting, the traction control system or the brake balance, all of which can be adjusted in-game by human players.

Computing environment

Each experiment used a single trainer on a compute node with either one NVIDIA V100 or half of an NVIDIA A100 coupled with around eight vCPUs and 55 GiB of memory. Some of these trainers were run in PlayStation Now data centres and others in AWS EC2 using p3.2xlarge instances.

Each experiment also used a number of rollout workers, where each rollout worker consisted of a compute node controlling a PS4. In this set-up, the PS4 ran the game and the compute node managed the rollouts by performing tasks such as computing actions, sending them to the game, sending experience streams to the trainer and receiving revised policies from the trainer (see Fig. 1a). The compute node used around two vCPUs and 3.3 GB of memory. In the time-trial experiments, ten rollout workers (and therefore ten PS4s) were used for about 8 days. To train policies that could drive in traffic, 21 rollout workers were used for between 7 and 12 days. In both cases, one worker was primarily evaluating intermediate policies, rather than generating new training data.

Actions

The GT API enabled control of three independent continuous actions: throttle, brake and steering. Because the throttle and brake are rarely engaged at the same time, the agent was presented control over the throttle and brake as one continuous action dimension. The combined dimension was scaled to $[-1, 1]$. Positive values engaged the throttle (with maximum throttle at +1), whereas negative values engaged the brake (with maximum braking at −1); the value zero engaged neither the throttle nor the brake. The steering dimension was also scaled to $[-1, 1]$, where the extreme values corresponded to the maximum steering angle possible in either direction for the vehicle being controlled.

The policy network selected actions by outputting a squashed normal distribution with a learned mean and diagonal covariance matrix over these two dimensions. The squashed normal distribution enforced sampled actions to always be within the $[-1, 1]$ action bounds[36]. The diagonal covariance matrix values were constrained to lie in the range (e^{-40}, e^{4}), allowing for nearly deterministic or nearly uniform random action selection policies to be learned.

Features

A variety of state features were input to the neural networks. These features were either directly available from the game state or processed into more convenient forms and concatenated before being input to the models.

Time-trial features. To learn competent time-trial performance, the agent needed features that allowed it to learn how the vehicle behaved and what the upcoming course looked like. The list of vehicle features included the car's 3D velocity, 3D angular velocity, 3D acceleration, load on each tyre and tyre-slip angles. Information about the environment was converted into features including the scalar progress of the car along the track represented as sine and cosine components, the local course surface inclination, the car's orientation with respect to the course centre line and the (left, centre and right) course points describing the course ahead on the basis of the car's velocity. The agent also received indicators if it contacted a fixed barrier or was considered off course by the game and it received real-valued feedback for the game's view of the car's most recent steering angle, throttle intensity and brake intensity. We relied on the game engine to determine whether the agent was off course (defined as when three or more tyres are out of bounds) because the out-of-bounds regions are not exactly defined by the course edges; kerbs and other tarmac areas outside the track edges are often considered in bounds.

Racing features. When training the agent to race against other cars, the list of features also included a car contact flag to detect collisions and a slipstream scalar that indicates if the agent was experiencing the slipstream effect from the cars in front of it. To represent the nearby cars, the agent used a fixed forward and rear distance bound to determine which cars to encode. The cars were ordered by their relative distance to the agent and were represented using their relative centre-of-mass position, velocity and acceleration. The combination of features provided the information required for the agent to drive fast and learn to overtake cars while avoiding collisions.

To keep the features described here in a reasonable numerical range when training neural networks, we standardized the inputs on the basis of the knowledge of the range of each feature scalar. We assumed that the samples were drawn from a uniform distribution given the range and computed the expected mean and standard deviation. These were used to compute the z-score for each scalar before being input to the models.

Rewards

The reward function was a hand-tuned linear combination of reward components computed on the transition between the previous state s and current state s'. The reward components were: course progress (R_{cp}), off-course penalty $(R_{soc}$ or $R_{loc})$, wall penalty (R_w), tyre-slip penalty (R_{ts}), passing bonus (R_{ps}), any-collision penalty (R_c), rear-end penalty (R_r) and unsporting-collision penalty (R_{uc}). The reward weightings for the three tracks are shown in Extended Data Table 1.

Owing to the high speeds on Sarthe, training for that track used a slightly different off-course penalty, included the unsporting-collision penalty and excluded the tyre-slip penalty. Note that, to reduce variance in time-sensitive rewards, such as course progress and off-course penalty, we filtered transitions when network delays were encountered. The components are described in detail below.

Course progress (R_{cp}). Following previous work[24], the primary reward component rewarded the amount of progress made along the track since the last observation. To measure progress, we made use of the state variable l that measured the length (in metres) along the centreline from the start of the track. The agent's centreline distance l was estimated by first projecting its current position to the closest point on the centreline. The progress reward was the difference in l between the previous and the current state: $R_{cp}(s, s') \triangleq s'_l - s_l$. To reduce the incentive to cut corners, this reward was masked when the agent was driving off course.

Off-course penalty $(R_{soc}$ or $R_{loc})$. The off-course reward penalty was proportional to the squared speed the agent was travelling at to further discourage corner cutting that may result in a very large gain in

position: $R_{soc}(s, s') \triangleq -(s'_o - s_o)(s'_{kph})^2$, where s_o is the cumulative time off course and s_{kph} is the speed in kilometres per hour. To avoid an explosion in values at Sarthe where driving speeds were markedly faster and corners particularly easy to cut, we used a penalty that was proportional to the speed (not squared): $R_{loc}(s, s') \triangleq -(s'_o - s_o)s'_{kph}$, and the penalty was doubled for the difficult first and final chicanes.

Wall penalty (R_w). To assist the agent in learning to avoid walls, a wall-contact penalty was included. This penalty was proportional to the squared speed of the car and the amount of time in contact with the wall since the last observation: $R_w(s, s') \triangleq -(s'_w - s_w)(s'_{kph})^2$, where s_w is the cumulative time that the agent was in contact with a wall.

Tyre-slip penalty (R_{ts}). Tyre slip makes it more difficult to control the car. To assist learning, we included a penalty when the tyres were slipping in a different direction from where they were pointing: $R_{ts}(s, s') \triangleq -\sum_i^4 \min(|s'_{tsr,i}|, 1.0)|s'_{ts\theta,i}|$, where $s_{tsr,i}$ is the tyre-slip ratio for the ith tyre and $s_{ts\theta,i}$ is the angle of the slip from the forward direction of the ith tyre.

Passing bonus (R_{ps}). As in previous work[25], to incentivize passing opponents, we included a term that positively rewarded gaining ground and overtaking opponents, and negatively rewarded losing ground to an opponent. The negative reward ensured that there were no positive-cycle reward loops to exploit and encouraged defensive play when an opponent was trying to overtake the agent. This reward was defined as $R_{ps}(s, s') \triangleq \sum_i (s_{L_i} - s'_{L_i})\max(\mathbf{1}_{(b,f)}(s_{L_i}), \mathbf{1}_{(b,f)}(s'_{L_i}))$, where s_{L_i} is the projected centreline signed distance (in metres) from the agent to opponent i and $\mathbf{1}_{(b,f)}(x)$ is an indicator function for when an opponent is no more than b metres behind nor f metres in front of the agent. We used $b = 20$ and $f = 40$ to train GT Sophy. The max operator ensures that the reward is provided when the agent was within bounds in the previous state or in the current state. In the particularly complex first and final chicanes of Sarthe, we masked this passing bonus to strongly discourage the agent from cutting corners to gain a passing reward.

Any-collision penalty (R_c). To discourage collisions and pushing cars off the road, we included a reward penalty whenever the agent was involved in any collision. This was defined as a negative indicator whenever the agent collided with another car: $R_c(s, s') \triangleq -\max_{i \in N} s'_{c,i}$, where $s_{c,i}$ is 1 when the agent collided with opponent i and 0 otherwise, and N is the number of opponents.

Rear-end penalty (R_r). Rear-ending an opponent was one of the more common ways to cause an opponent to lose control and for the agent to be penalized by stewards. To discourage bumping from behind, we included the penalty $R_r(s, s') \triangleq -\sum_i s'_{c,i} \cdot \mathbf{1}_{>0}(s'_{l,i} - s'_l) \cdot \|s'_v - s'_{v,i}\|_2^2$, where $s_{c,i}$ is a binary indicator for whether the agent was in a collision with opponent i, $\mathbf{1}_{>0}(s_{l,i} - s_l)$ is an indicator for whether opponent i was in front of the agent, s_v is the velocity vector of the agent and $s_{v,i}$ is the velocity vector of opponent i. The penalty was dependent on speed to more strongly discourage higher speed collisions.

Unsporting-collision penalty (R_{uc}). Owing to the high speed of cars and the technical difficulty of Sarthe, training the agent to avoid collisions was particularly challenging. Merely increasing the any-collision penalty resulted in very timid agent behaviour. To discourage being involved in collisions without causing the agent to be too timid, we included an extra collision penalty for Sarthe. Like the any-collision penalty, this penalty was a negative Boolean indicator. Unlike the any-collision penalty, it only fired when the agent rear-ended or sideswiped an opponent on a straightaway or was in a collision in a curve that was not caused by an opponent rear-ending them: $R_{uc}(s, s') \triangleq -\max_{i \in N} u(s', i)$, where $u(s', i)$ indicates an unsporting collision as defined above.

Training algorithm

To train our agent, we used an extension of the soft actor-critic algorithm[36] that we refer to as QR-SAC. To give the agent more capacity to predict the variation in the environment during a race, we make use of a QR Q-function[39] modified to accept continuous actions as inputs. QR-SAC is similar to distributional SAC[44] but uses a different formulation of the value backup and target functions. We used $M = 32$ quantiles and modified the loss function of the QR Q-function with an N-step temporal difference backup. The target function, y_i, for the ith quantile, $\hat{\tau}_i$, consists of terms for the immediate reward, $R_t = \sum_{i=1}^N \gamma^{i-1} r_{t+i}$, the estimated quantile value at the Nth future state, $Z_{\hat{\tau}_i}$, and the SAC entropy term. Like existing work using N-step backups[38], we do not correct for the off-policy nature of N-step returns stored in the replay buffer. To avoid the computational cost of forwarding the policy for intermediate steps of the N-step backup, we only include the entropy reward bonus that SAC adds for encouraging exploration in the final step of the N-step backup. Despite this lack of off-policy correction and limited use of the entropy reward bonus, we found that using N-step backups greatly improved performance compared with a standard one-step backup, as shown in Fig. 2d. To avoid overestimation bias, the Nth state quantiles are taken from the Q-function with the smallest Nth state mean value[45], indexed by k

$$k = \arg\min_{m=1,2} Q(s_{t+N}, a'|\theta'_m)$$
$$y_i = R_t + Z_{\hat{\tau}_i}(s_{t+N}, a'|\theta_k) - \alpha \log \pi(a'|s_{t+N}, \phi) \quad (1)$$

where θ and ϕ are parameters of the Q-functions and the policy, respectively. Using this target value, y_i, the loss function of the Q-function is defined as follows

$$\delta_{i,j} = y_i - Z_{\hat{\tau}_j}(s_t, a_t|\theta)$$
$$L(\theta) = \frac{1}{M^2}\sum_i \sum_j \mathbb{E}_{s_t, a_t, R_t, s_{t+1} \sim D, a' \sim \pi}\rho(\delta_{i,j}) \quad (2)$$

where D represents data from the ERB and ρ is a quantile Huber loss function[39]. Finally, the objective function for the policy is as follows:

$$J(\phi) = \mathbb{E}_{s \sim D, a \sim \pi(a|s,\pi)}[\alpha \log \pi(a|s, \phi) - \min_{i=1,2} Q(s, a|\theta_i)] \quad (3)$$

The Q-functions and policy models used in the October race consist of four hidden layers with 2,048 units each and a rectified linear unit activation function. To achieve robust control, dropout[46] with a 0.1 drop probability is applied to the policy function[47]. The parameters are optimized using an Adam optimizer[48] with learning rates of 5.0×10^{-5} and 2.5×10^{-5} for the Q-function and policy, respectively. The discount factor γ was 0.9896 and the SAC entropy temperature value α was set to 0.01. The mixing parameter when updating the target model parameters after every algorithm step was set to 0.005. The off-course penalty and rear-end-speed penalty can produce large penalty values due to the squared speed term, which makes the Q-function training unstable due to large loss values. To mitigate this issue, the gradients of the Q-function are clipped by the global norm of the of 10.

The rollout workers send state-transition tuples $\langle s, a, r \rangle$ collected in an episode (of length 150 s) to the trainer to store the data in an ERB implemented using the Reverb Python library[49]. The buffer had capacity of 10^7 N-step transitions. The trainer began the training loop once 40,000 transitions had been collected and uses a mini-batch of size 1,024 to update the Q-function and policy. A training epoch consists of 6,000 gradient steps. After each epoch, the trainer sent the latest model parameters to the rollout workers.

Training scenarios

Learning to race requires mastering a gamut of skills: surviving a crowded start, making tactical open-road passes and precisely running the track alone. To encourage basic racing skills, we placed the agent in scenarios with zero, one, two, three or seven opponents launched nearby (1v0, 1v1, 1v2, 1v3 and 1v7, respectively). To create variety, we randomized track positions, start speeds, spacing between cars and opponent policies. We leveraged the fact that the game supports 20 cars at a time to maximize PlayStation usage by launching more than one group on the track. All base scenarios ran for 150 s. In addition, to ensure that the agent was exposed to situations that would allow it to learn the skills highlighted by our expert advisor, we used time-limited or distance-limited scenarios on specific course sections. Figure 1f illustrates the skill scenarios used at Sarthe: eight-car grid starts, 1v1 slipstream passing and mastering the final chicane in light traffic. Extended Data Figure 1 shows the specialized scenarios used to prepare the agent to race on Seaside (**f**) and Maggiore (**g**). To learn how to avoid catastrophic outcomes at the high-speed Sarthe track, we also incorporated mistake learning[50]. During policy evaluations, if an agent lost control of the car, the state shortly before the event was recorded and used as a launch point for more training scenarios.

Unlike curriculum training where early skills are supplanted by later ones or in which skills build on top of one another in a hierarchical fashion, our training scenarios are complementary and were trained into a single control policy for racing. During training, the trainer assigned new scenarios to each rollout worker by selecting from the set configured for that track on the basis of hand-tuned ratios designed to provide sufficient skill coverage. See Extended Data Fig. 1e for an example ERB at Sarthe. However, even with this relative execution balance, random sampling fluctuations from the buffer often led to skills being unlearned between successive training epochs, as shown in Fig. 2h. Therefore, we implemented multi-table stratified sampling to explicitly enforce proportions of each scenario in each training mini-batch, notably stabilizing skill retention (Fig. 2g).

Policy selection

In machine learning, convergence means that further training will not improve performance. In RL, due to the continuing exploration and random sampling of experiences, the performance of the policy will often continue to vary after convergence (Fig. 2h). Therefore, even with the stabilizing techniques described above, continuing training after convergence produced policies that differed in small ways in their ability to execute the desired racing skills. A subsequent policy, for instance, may become marginally better at the slipstream pass and marginally worse at the chicane. Choosing which policy to race against humans became a complex, multi-objective optimization problem.

Extended Data Figure 3 illustrates the policy-selection process. Agent policies were saved at regular intervals during training. Each saved policy then competed in a single-race scenario against other AI agents, and various metrics, such as lap times and car collisions, were gathered and used to filter the saved policies to a smaller set of candidates. These candidates were then run through an *n*-athlon—a set of pre-specified evaluation scenarios—testing their lap speed and performance in certain tactically important scenarios, such as starting and using the slipstream. The performance on each scenario was scored and the results of each policy on each scenario were combined in a single ranked spreadsheet. This spreadsheet, along with various plots and videos, was then reviewed by a human committee to select a small set of policies that seemed the most competitive and the best behaved. From this set, each pair of policies competed in a multi-race, round-robin, policy-versus-policy tournament. These competitions were scored using the same team scoring as that in the exhibition event and evaluated on collision metrics. From these results, the committee chose policies that seemed to have the best chance of winning against the human drivers while minimizing

penalties. These final candidate policies were then raced against test drivers at Polyphony Digital and the subjective reports of test drivers were factored into the final decision.

The start of Sarthe posed a particularly challenging problem for policy selection. Because the final chicane is so close to the starting line, the race was configured with a stationary grid start. From that standing start, all eight cars quickly accelerated and entered the first chicane. Although a group of eight GT Sophy agents might get through the chicane fairly smoothly, against human drivers, the start was invariably chaotic and a fair amount of bumping occurred. We tried many variations of our reward functions to find a combination that was deemed an acceptable starter by our test drivers while not giving up too many positions. In the October 2021 Sarthe race, we configured GT Sophy to use a policy that started well, and—after 2,100 metres—switch to a slightly more competitive policy for the rest of the race. Despite the specialized starter, the instance of GT Sophy that began the race in pole position was involved in a collision with a human driver in the first chicane, slid off the course and fell to last place. Despite that setback, it managed to come back and win the race.

Immediately after the official race, we ran a friendly rematch against the same drivers but used the starter policy for the whole track. The results were similar to the official race.

Fairness versus humans

Competitions between humans and AI systems cannot be made entirely fair; computers and humans think in different ways and with different hardware. Our objective was to make the competition fair enough, while using technical approaches that were consistent with how such an agent could be added to the game. The following list compares some of the dimensions along which GT Sophy differs from human players:

First, perception. GT Sophy had a map of the course with precise x, y and z information about the points that defined the track boundaries. Humans perceived this information less precisely by means of vision. However, the course map did not have all of the information about the track and humans have an advantage in that they could see the kerbs and surface material outside the boundaries, whereas GT Sophy could only sense these by driving on them.

Second, opponents. GT Sophy had precise information about the location, velocity and acceleration of the nearby vehicles. However, it represented these vehicles as single points, whereas humans could perceive the whole vehicle. GT Sophy has a distinct advantage in that it can see vehicles behind it as clearly as it can see those in front, whereas humans have to use the mirrors or the controller to look to the sides and behind them. GT Sophy never practiced against opponents that didn't have full visibility, so it didn't intentionally take advantage of human blind spots.

Third, vehicle state. GT Sophy had precise information about the load on each tyre, slip angle of each tyre and other vehicle state. Humans learn how to control the car with less precise information about these state variables.

Fourth, vehicle controls. There are certain vehicle controls that the human drivers had access to that GT Sophy did not. In particular, expert human drivers often use the traction control system in grid starts and use the transmission controls to change gears.

Fifth, action frequency. GT Sophy took actions at 10 Hz, which was sufficient to control the car but much less frequent than human actions in GT. Competitive GT drivers use steering and pedal systems that give them 60 Hz control. Whereas a human can't take 60 distinct actions per second, they can smoothly turn a steering wheel or press on a brake pedal. Extended Data Figure 2b, c contrasts GT Sophy's 10-Hz control pattern to Igor Fraga's much smoother actions in a corner of Sarthe.

Sixth, reaction time. GT Sophy's asynchronous communication and inference takes around 23–30 ms, depending on the size of the network. Although evaluating performance in professional athletes and gamers is a complex field[34,35], an oft-quoted metric is that professional athletes have a reaction time of 200–250 ms. To understand how the

performance of GT Sophy would be affected if its reaction time were slowed down, we ran experiments in which we introduced artificial delays to its perception pipeline. We retrained our agent with delays of 100 ms, 200 ms and 250 ms in the Maggiore time-trial setting, using the same model architecture and algorithm as our time-trial baseline. All three of these tests achieved a superhuman lap time.

Tests versus top GT drivers

The following competitive GT drivers participated in the time-trial evaluations:

- Emily Jones: 2020 FIA Gran Turismo Manufacturers Series, Team Audi.
- Valerio Gallo: 2nd place 2020 FIA Gran Turismo Nations Cup; winner 2021 Olympic Virtual Series Motor Sport Event; winner 2021 FIA Gran Turismo Nations Cup.
- Igor Fraga: winner 2018 FIA Gran Turismo Nations Cup; winner 2019 Manufacturer Series championship; winner 2020 Toyota Racing Series (real racing).

GT Sophy won all of the time-trial evaluations as shown in Fig. 3f and was reliably superhuman on all three tracks, as shown in Fig. 1d, e and Extended Data Fig. 1a–d. Notably, the only human with a time within the range of GT Sophy's 100 lap times on any of the tracks was Valerio Gallo on Maggiore. It is worth noting that the data in Fig. 1d, e was captured by Polyphony Digital after the time-trial event in July 2021. Valerio was the only participant represented in the data that had seen the trajectories of GT Sophy on Maggiore, and—between those two events—Valerio's best time improved from 114.466 to 114.181 s.

It is also interesting to examine what behaviours give GT Sophy such an advantage in time trials. Extended Data Figure 2a shows an analysis of Igor's attempt to match GT Sophy on Sarthe, showing the places on the course where he fell farther behind. Not surprisingly, the hardest chicanes and corners are the places where GT Sophy has the biggest performance gains. In most of these corners, Igor seems to catch up a little bit by braking later, but is then unable to take the corner itself as fast, resulting in him losing ground overall.

The following competitive GT drivers participated in the team racing event:

- Takuma Miyazono: winner 2020 FIA Gran Turismo Nations Cup; winner 2020 FIA Gran Turismo Manufacturer Series; winner 2020 GR Supra GT Cup.
- Tomoaki Yamanaka: winner 2019, 2021 Manufacturer Series.
- Ryota Kokubun: winner 2019 FIA Gran Turismo Nations Cup, Round 5, Tokyo; 3rd place 2020 FIA Gran Turismo Nations Cup.
- Shotaro Ryu: 2nd place Japan National Inter-prefectural Esports Championship (National Athletic Meet) 2019 Gran Turismo Division (Youth).

Driver testimonials

The following quotes were captured after the July 2021 events:

"I think the AI was very fast turning into the corner. How they approach into it, as well as not losing speed on the exit. We tend to sacrifice a little bit the entry to make the car be in a better position for the exit, but the AI seems to be able to carry more speed into the corner but still be able to have the same kind of exit, or even a faster exit. The AI can create this type of line a lot quicker than us,… it was not a possibility before because we never realized it. But the AI was able to find it for us." — Igor Fraga

"It was really interesting seeing the lines where the AI would go, there were certain corners where I was going out wide and then cutting back in, and the AI was going in all the way around, so I learned a lot about the lines. And also knowing what to prioritize. Going into turn 1 for example, I was braking later than the AI, but the AI would get a much better exit

than me and beat me to the next corner. I didn't notice that until I saw the AI and was like 'Okay, I should do that instead'." — Emily Jones

"The ghost is always a reference. Even when I train I always use someone else's ghost to improve. And in this case with such a very fast ghost,… even though I wasn't getting close to it, I was getting closer to my limits." — Valerio Gallo

"I hope we can race together more, as I felt a kind of friendly rivalry with [GT Sophy]." (translated from Japanese) — Takuma Miyazono

"There is a lot to learn from [GT Sophy], and by that I can improve myself. [GT Sophy] does something original to make the car go faster, and we will know it's reasonable once we see it." (translated from Japanese) — Tomoaki Yamanaka

Data availability

There are no static data associated with this project. All data are generated from scratch by the agent each time it learns. Videos of the races are available at https://sonyai.github.io/gt_sophy_public.

Code availability

Pseudocode detailing the training process and algorithms used is available as a supplement to this article. The agent interface in GT is not enabled in commercial versions of the game; however, Polyphony Digital has provided a small number of universities and research facilities outside Sony access to the API and is considering working with other groups.

44. Xia, L., Zhou, Z., Yang, J. & Zhao, Q. DSAC: distributional soft actor critic for risk-sensitive reinforcement learning. Preprint at https://arxiv.org/abs/2004.14547 (2020).
45. Fujimoto, S., van Hoof, H. & Meger, D. In *Proc. 35th International Conference on Machine Learning* 1587–1596 (PMLR, 2018).
46. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
47. Liu, Z., Li, X., Kang, B. & Darrell, T. In *International Conference on Learning Representations* (ICLR, 2021).
48. Kingma, D. P. & Ba, J. In *International Conference on Learning Representations* (ICLR, 2015).
49. Cassirer, A. et al. Reverb: a framework for experience replay. Preprint at https://arxiv.org/abs/2102.04736 (2021).
50. Narvekar, S., Sinapov, J., Leonetti, M. & Stone, P. In *Proc. 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016)* (2016).

Extended Data Fig. 1 | Seaside and Sarthe training. Kudos Prime data from global time-trial challenges on Seaside (**a** and **b**) and Sarthe (**c** and **d**), with the cars used in the competition. Note that these histograms represent the single best lap time for more than 12,000 individual players on Seaside and almost 9,000 on Sarthe. In both cases, the secondary diagrams compare the top five human times to a histogram of 100 laps by the 2 July 2021 time-trial version of GT Sophy. In both cases, the data show that GT Sophy was reliably superhuman, with all 100 laps better than the best human laps. Not surprisingly, it takes
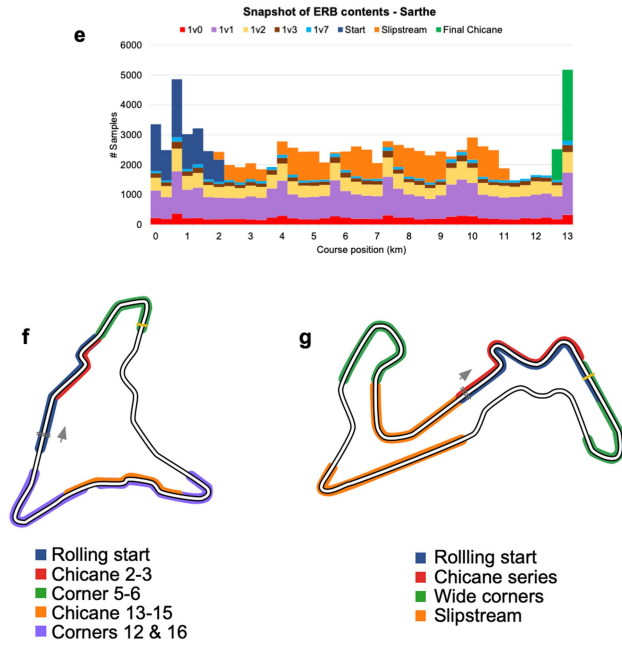
longer for the agent to train on the much longer Sarthe course, taking 48 h to reach the 99th percentile of human performance. **e**, Histogram of a snapshot of the ERB during training on Sarthe on the basis of the scenario breakdown in Fig. 1f. The x axis is the course position and the stacked colours represent the number of samples that were collected in that region from each scenario. In a more condensed format than Fig. 1f, **f** and **g** show the sections of Seaside and Maggiore that were used for skill training.
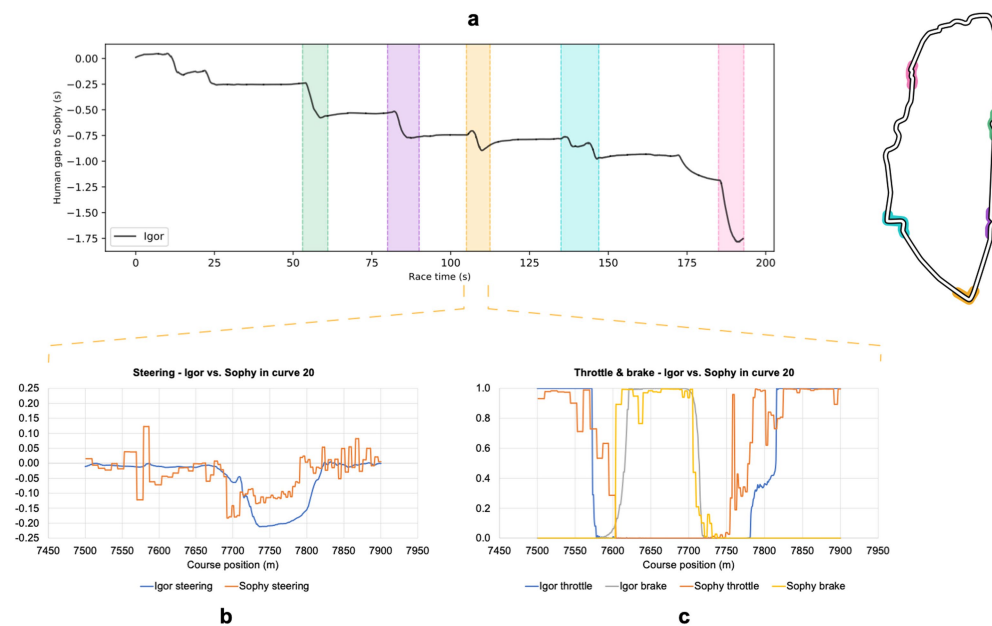
Extended Data Fig. 2 | Time trial on Sarthe. An analysis of Igor Fraga's best lap in the time-trial test compared with GT Sophy's lap. **a**, Areas of the track where Igor lost time with respect to GT Sophy. Corner 20, highlighted in yellow, shows an interesting effect common to the other corners in that Igor seems to catch up a little by braking later, but then loses time because he has to brake longer and comes out of the corner slower. Igor's steering controls (**b**) and Igor's throttle and braking (**c**) compared with GT Sophy on corner 20. Through the steering wheel and brake pedal, Igor is able to give smooth, 60-Hz signals compared with GT Sophy's 10-Hz action rate.
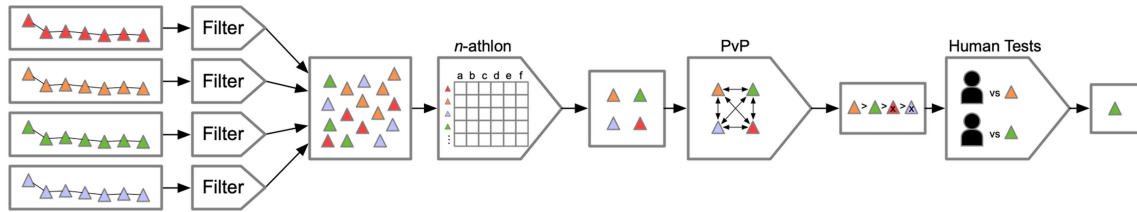
Extended Data Fig. 3 | Policy selection. An illustration of the process by which policies were selected to run in the final race. Starting on the left side of the diagram, thousands of policies were generated and saved during the experiments. They were first filtered in the experiment to select the subset on the Pareto frontier of a simple evaluation criteria trading off lap time versus off-course and collision metrics. The selected policies were run through a series of tests evaluating their overall racing performance against a common set of opponents and their performance on a variety of hand-crafted skill tests. The results were ranked and human judgement was applied to select a small number of candidate policies. These policies were matched up in round-robin, policy-versus-policy competitions. The results were again analysed by the human committee for overall team scores and collision metrics. The best candidate policies were run in short races against test drivers at Polyphony Digital. Their subjective evaluations were included in the final decisions on which policies to run in the October 2021 event.

Article

Extended Data Table 1 | Reward weights

Course	R_{cp}	R_{soc}	R_{loc}	R_w	R_{ts}	R_{ps}	R_c	R_r	R_{uc}
Seaside	1	0.01	0	0.01	0.25	0.5	5	0.1	0
Maggiore	1	0.01	0	0.01	0.25	0.5	4	0.1	0
Sarthe	1	0	5	0.01	0	0.5	5	0.1	5

Reward weights for each track.