A Fleet of Miniature Cars for Experiments in Cooperative Driving

Nicholas Hyldmar*, Yijun He*, Amanda Prorok

***Abstract*— We introduce a unique experimental testbed that consists of a fleet of 16 miniature Ackermann-steering vehicles. We are motivated by a lack of available low-cost platforms to support research and education in multi-car navigation and trajectory planning. This article elaborates the design of our miniature robotic car, the *Cambridge Minicar*, as well as the fleet's control architecture. Our experimental testbed allows us to implement state-of-the-art driver models as well as autonomous control strategies, and test their validity in a real, physical multi-lane setup. Through experiments on our miniature highway, we are able to tangibly demonstrate the benefits of cooperative driving on multi-lane road topographies. Our setup paves the way for indoor large-fleet experimental research.**

I. Introduction

The deployment of connected, automated, and autonomous vehicles presents us with transformational opportunities for road transport. To date, the number of companies working on this technology is substantive, and growing [1]. Opportunities reach beyond single-vehicle automation: by enabling groups of vehicles to jointly agree on maneuvers and navigation strategies, real-time coordination promises to improve overall traffic throughput, road capacity, and passenger safety [5, 6]. However, coordinated driving for intelligent vehicles still remains a challenging research problem, due to unpredictable vehicle behaviors (e.g., non-cooperative cars, unreliable communication), hard workspace limitations (e.g., lane topographies), and constrained kinodynamic capabilities (e.g., steering kinematics, driver comfort).

Developing true-scale facilities for safe, controlled vehicle testbeds is massively expensive and requires a vast amount of space. For example, the University of Michigan's MCity Test Facility cost US $10 million to develop and covers 32 acres (0.13 km^2). As a consequence, using fleets of actual vehicles is possible only for very few research institutes worldwide. Moreover, although such facilities are excellently suited for research and development, safety and operational concerns prohibit the integration of educational curricula and outreach activities. One approach to facilitating experimental research and education is to build low-cost testbeds that incorporate fleets of down-sized, car-like mobile platforms. Following this idea, we propose a multi-car testbed that allows for the operation of tens of vehicles within the space of a moderately large robotics laboratory, and allows for the teaching and research of coordinated driving strategies in dense traffic scenarios.

Our motivation is to design a testbed that scales to a large number of cars so that we could test vehicle-to-vehicle interactions (cooperative as well as non-cooperative) and the effect of these interactions in multi-car traffic scenarios. Although a number of low-cost multi-robot testbeds exist (e.g., [8, 17]), most use robotic platforms with differential drive kinematics (which tend to also have limited maximum speeds). Few testbeds integrate car-like Ackermann-steering vehicles, and in very small numbers (e.g., [12]).

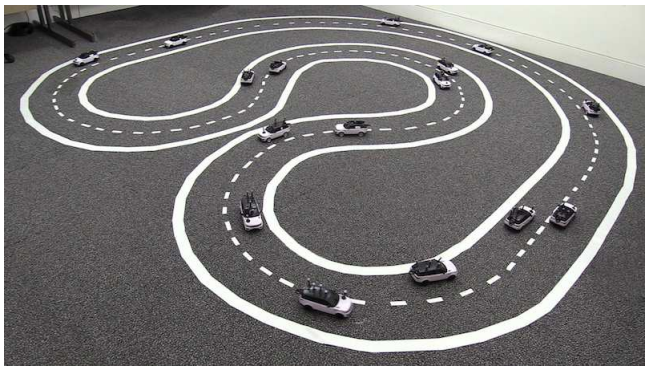

Fig. 1: The fleet of Minicars on a U-shaped two-lane miniature freeway. The inner and outer track lengths are 16 m and 17 m, respectively.

In this work, we propose the design of a low-cost miniature robotic car, the *Cambridge Minicar*, which is based on a 1:24 model of an existing commercial car. The Minicar is built from off-the-shelf components (with the exception of one laser-cut piece), and costs approximately US $76 in its basic configuration. Its low cost allows us to compose a large fleet, which we use to test navigation strategies and driver models. Overall, our contributions in this work are *(a)* the design of a low-cost miniature robotic car, *(b)* the proposition of a system architecture that incorporates decentralized multi-car control algorithms for indoor testing of real large fleets, and *(c)* the availability of our designs and code in an open-source repository [1].

II. Existing Platforms

A variety of low-cost mobile robot platforms are available for research and education. The work in [16] presents a very recent comprehensive overview of platforms that cost less than US $300, have been designed in the last 10 years, or are currently available on the market. Apart from the Kilobot [20] and the AERobot [21], which have slip-stick forwards motion, all other robots in this overview are wheeled differential drive platforms [2–4, 10, 15, 16, 19, 24]. The lack of low-cost, Ackermann-steering robots is apparent.

There are a few recent robot designs that are based on Ackermann-steering platforms, and we provide an overview thereof in Table I. The largest of these platforms is the

All authors are with the University of Cambridge, UK: {nh490, yh403, asp45}@cam.ac.uk. *These authors contributed equally to this work. We gratefully acknowledge the Isaac Newton Trust who are supporting Amanda Prorok through an Early Career Grant.

[1] https://github.com/proroklab/minicar

Car	Scale	Price (USD)
ETHZ ORCA Racer [14]	1:43	$470
Cambridge Minicar	1:24	$76.5
BARC [7]	1:10	$840
MIT Racecar [9]	1:10	$1060
GATech AutoRally [23]	1:5	$9210

TABLE I: Overview of Ackermann-steering platforms.[7]

Georgia Tech AutoRally [23][2]. It is based on a 1:5-scale vehicle chassis, runs on an ASUS Mini-ITX motherboard that includes a GPU, and is capable of fast autonomous driving using on-board sensors only. It is specifically designed for outdoor experimentation and the testing of aggressive maneuvers (all computational components are housed in a rugged aluminum enclosure able to withstand violent vehicle rollovers.) The MIT Racecar [9][3] and the Berkeley Autonomous Race Car (BARC) [7][4] are 1:10-scale rally cars based on a chassis that is commercially available (Traxxas). Similar to the AutoRally, these platforms feature high-end computational units with GPUs that allow for the addition of numerous sensors (e.g., laser range finder, camera, GPS) for on-board autonomy. As listed in Table I, the price of these three platforms lies in the range $800-$9300 for the core alone (chassis, motors, and computational units). Due to the significant cost and moderately large size of the platforms, it is difficult to facilitate indoor experiments that include large numbers of these vehicles.

At the other end of the spectrum lies the ETHZ ORCA Racer [14][5], which is based on a 1:43-scale platform. The chassis is provided by a Kyosho *dnano* RC race car (achieving top speeds of more than 3 m/s). In order to control the platform, the authors designed a custom PCB with an ARM Cortex M4 microcontroller, a Bluetooth chip, and H-bridges for DC motor actuation (for the steering servo and drive train). The overall cost of one car is $470 (approximately $300 for the PCB parts and $170 for the *dnano* chassis). This platform provides interesting dynamic capabilities, and a large number of cars can be operated in very confined spaces. Although the platform is considerably cheaper than the aforementioned models, it is still significantly more expensive than our Minicar (with similar capabilities). Furthermore, the platform is too small to carry an off-the-shelf computer, such as Raspberry Pi Zero W [6] (which facilitates communication via WiFi, as well as the addition of various sensors). Finally, its design and accompanying software tools are not open-sourced. These factors limit the extensibility and scalability of the testbed.

III. Vehicle Design

The main components of the Minicar are a Raspberry Pi Zero W, a chassis with forwards drive train and servo-motors, and two battery sets. Figure 2 shows an exploded view of the Minicar. It is $75 \times 81 \times 197$ mm and weighs 450 g (including batteries). The logic can be powered for over 5 hours and the motors for 2.2 hours at 0.3 m/s. The logic is powered separately from the motors to isolate it from the noisy environment and increase the car's runtime. The motor and logic powers have capacities of 2500 mAh and 3350 mAh respectively. The AA batteries supply 4.2 V to the servo which is increased to 7 V for the motor by the boost converter. The servo logic and motor enable pin are controlled using pulse width modulation (PWM) by the Pi Zero W. A servo arm is connected to a laser-cut gear that meshes with the existing steering gear. The vehicle's wheel base is $L = 122$ mm. Its minimum turning radius is $R = 0.56$ m, its maximum steering angle is $|\psi|_{\max} = 18°$, its maximum steering rate is $|\dot{\psi}|_{\max} = 0.076$ rad/s and its maximum forwards speed is $v_{\max} = 1.5$ m/s (this can be increased by tuning the output voltage of the boost converter).

Component	Item	Price (USD)
Computation	Raspberry Pi Zero W	$12.3
Memory	8GB Micro SD card	$5.2
Chassis & Motor	1:24 Range Rover Sport	$15.6
Steering	Micro-servo	$6.2
Motor Power	AA Batteries	$11.1
Logic Power	Portable Charger	$15.6
Boost converter	XL6009	$2.6
H-Bridge	L293D	$0.5
Board	Proto Bonnet	$4.5
Logic Switch	USB Switch	$2.9

TABLE II: Cambridge Minicar: overview of components.

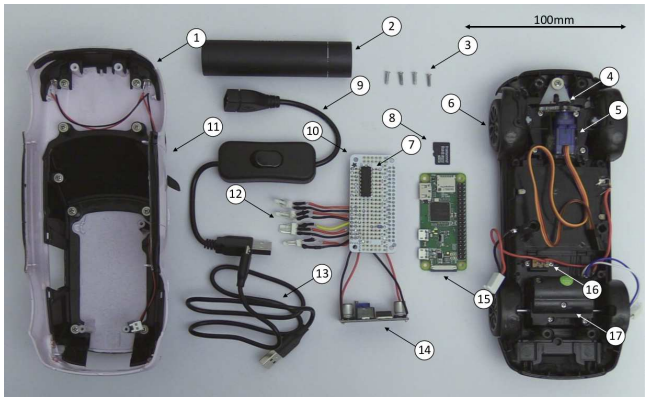

Fig. 2: Exploded view of a Cambridge Minicar. 1: Headlights 2: Portable charger 3: Casing screws 4: Gear 5: Servo 6: Lower casing 7: H-bridge 8: Micro SD card 9: USB switch 10: Circuit board 11: Upper casing 12: JST connectors 13: Micro USB cable 14: Boost converter 15: Raspberry Pi Zero W 16: Motor switch 17: Drive motor

Table II lists the Minicar's components. The cost can be reduced by $28 by using power sources of lower capacity, removing the logic switch and JST connectors and replacing the board with a stripboard. The process of making a Minicar involves modifying the casing, soldering the circuit board, printing the gear, and fixing the components in place. This takes approximately 3 hours per car.

IV. Testbed

The architecture of our system is illustrated in Figure 3. We use an OptiTrack motion capture system based on

[2] https://autorally.github.io/

[3] https://github.com/mit-racecar

[4] http://www.barc-project.com/

[5] https://sites.google.com/site/orcaracer/home

[6] https://www.raspberrypi.org

[7] This overview lists platforms developed in the last 10 years. Prices indicate the cost of the core body, including chassis, motors and computational units (excluding additional sensor components.)

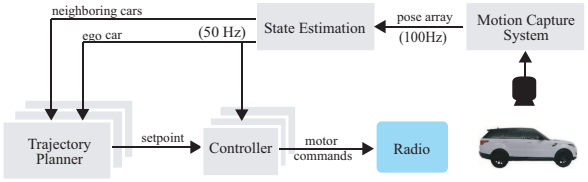

Fig. 3: Diagram of our system architecture.

passive reflective markers to provide real-time feedback on the vehicles' positions. Each Minicar is equipped with a unique configuration of five markers. The motion capture system tracks the Minicars and provides pose measurements at 100 Hz. An extended Kalman filter uses the pose array to provide an estimate of each vehicle's state that includes vehicle pose $[x, y, \theta]$, velocity v and steering angle ψ. We implement an inner and outer loop control method for trajectory tracking. The outer loop (i.e., Trajectory Planner) generates trajectories (or uses pre-computed trajectories, such as freeway lanes) to compute velocity and steering angle setpoints, which are fed to the inner loop. In Section VI, we provide an example of a Trajectory Planner for use in multi-lane freeway traffic. The inner loop (i.e., Controller) is responsible for correcting motor commands using PID control on velocity and steering angle. These control values are sent to the vehicles over broadband radio. The on-board computer applies the corresponding motor commands with pulse-with modulated signals. At the scale of our current system (16 vehicles), we did not notice any communication latency. Should this become an issue at larger numbers of vehicles, we note that the architecture can easily be scaled by considering multi-radio solutions, such as in [18].

Our testbed architecture is designed for ease of use, and our key aim is the rapid development and testing of driving behaviors on car-like robots (such as the Minicar). Although the Minicar's design allows for the integration of sensors (such as an IMU or camera [8]), and its on-board computer is capable of performing its own state estimation and control computations, we decided to keep the intelligence off-board, emulating proprioceptive and exteroceptive observations through the motion capture system instead. In our setup, a workstation runs a thread for each Minicar's Trajectory Planner and Controller. Upon booting, the Minicar executes a listener that waits for motor commands. This software design choice facilitates rapid testing, and allows us to focus on the development of driving strategies for large Minicar fleets. Our fleet operates on a miniature two-lane U-shaped freeway, shown in Figure 1.

V. Trajectory Tracking

The Minicar has Ackermann steering geometry, and its kinematics can be approximated by the bicycle model, with motion equations as follows:

$$\dot{x} = v\cos\theta \quad (1)$$

$$\dot{y} = v\sin\theta \quad (2)$$

$$\dot{\theta} = L^{-1}v\tan\psi \quad (3)$$

[8]A dedicated Raspberry Pi Camera Module costs less than $ 30; its miniature form factor allows it to be easily fitted onto the Minicar.

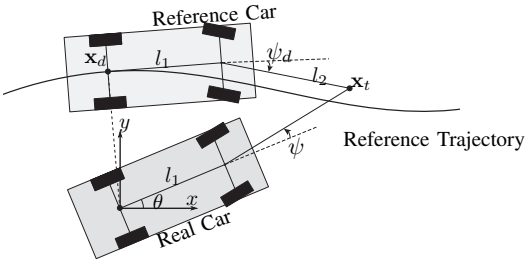

Fig. 4: Illustration of our trajectory tracking strategy, based on [13].

where ψ is the steering angle, v is the forward speed, and L is the vehicle's wheel base. We use the lateral control strategy introduced in [13]. This strategy has the advantage of being speed independent, so that the velocity of the vehicle can be controlled independently for other purposes and still converge to the desired pose on the trajectory.

Figure 4 illustrates the control strategy. The real vehicle with yaw θ and position $\mathbf{0}$ is projected orthogonally onto the reference trajectory to create a virtual reference vehicle at point $\mathbf{x}_d$, which is the closest point on the reference trajectory to the real vehicle. The yaw of the virtual reference vehicle is aligned with the tangent at $\mathbf{x}_d$, and its steering angle ψ_d is such that the vehicle's turning radius coincides with the curvature κ of the reference trajectory at $\mathbf{x}_d$. We compute the target point $\mathbf{x}_t$ with

$$x_t = x_d + l_1\cos\theta_d + l_2\cos(\theta_d + \psi_d) \quad (4)$$

$$y_t = y_d + l_1\sin\theta_d + l_2\sin(\theta_d + \psi_d) \quad (5)$$

where the steering angle of the reference vehicle is $\psi_d = \arctan(l_1\kappa)$. The real vehicle's steering angle is

$$\psi = \arctan2(y_t - l_1\sin\theta, x_t - l_1\cos\theta) - \theta. \quad (6)$$

As noted in [13], the choice of parameters l_1 and l_2 affect the control; small values for l_1 lead to fast control, but may lead to an overshoot of the reference trajectory; small values for l_2 (gain) lead to fast control, but also mean high control values, which may lead to a saturation of the steering angle. We experimentally tuned our parameters, and set $l_1 = L$ and $l_2 = 2.3L$.

VI. A Multi-Car Traffic System

Building on the system architecture and vehicle control strategy described above, we design an experimental multi-car multi-lane traffic system that emulates freeway driving in an indoor lab setup. In our architecture, each vehicle is controlled by its individual, independent trajectory planning module, and hence, heterogeneous driving behaviors are possible. Our aim is to show how our experimental setup allows us to validate various driving controllers on actual platforms in a freeway-like setting. To this end, we implement three control strategies: *(i)* the car is driven by an egocentric, human-like policy, *(ii)* the car is driven by a cooperative policy, and *(iii)* the car is driven by a human player (i.e., a gamified policy).

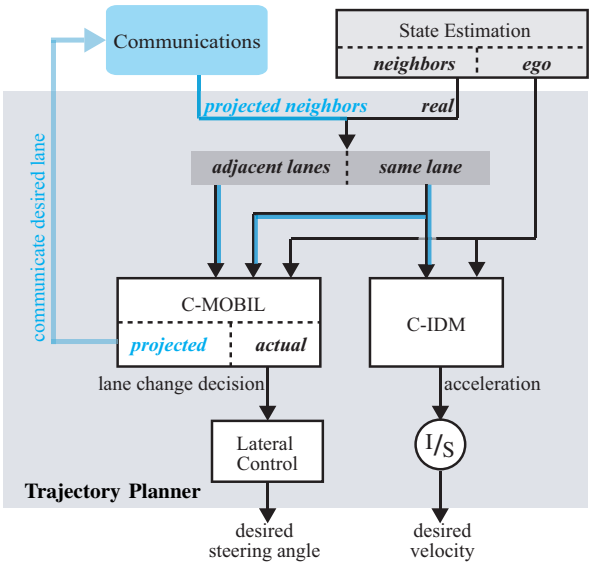

Fig. 5: Diagram of trajectory planner with our two main algorithmic modules, C-MOBIL and C-IDM. The ego vehicle plans a trajectory and velocity profile based on its own state, and on the actual state of its neighboring vehicles. In a cooperative approach (with diagram elements marked in blue), the algorithm modules also integrate the projected (desired) states of neighboring vehicles.

A. Egocentric Driving

Our human-like driving model treats longitudinal and lateral control as two independent entities. It uses an acceleration model that controls longitudinal motion along the current car lane, and a steering model that controls lateral motion across multiple lanes.

Longitudinal Control. Our longitudinal control is based on the Intelligent Driver Model (IDM) first proposed in [22]. The idea underpinning IDM is that a vehicle's acceleration is a function of the vehicle's current velocity v, it's gap s to the preceding vehicle, and the approach rate Δv to the preceding vehicle. It is formalized as

$$a_{\text{IDM}} = \alpha \left[1 - \left(\frac{v}{v_0} \right)^{\delta} - \left(\frac{s^\star(v, \Delta v)}{s} \right)^2 \right] \tag{7}$$

where $s^\star$ is a function determining the desired minimum gap to the preceding vehicle. We compute this value as

$$s^\star(v, \Delta v) = s_0 + Tv + \frac{v \Delta v}{2\sqrt{\alpha \beta}}. \tag{8}$$

where s_0 corresponds to a jam distance. The vehicle's control input at time t is computed through the integrator $v_t = a_{\text{IDM}} \cdot \Delta t + v_{t-1}$.

Lateral Control. Our lateral control strategy builds on the MOBIL lane changing policy proposed in [11]. The MOBIL strategy decides whether a vehicle should change a lane or not based on two steps. First, it ensures that a safety criterion is met, i.e., it guarantees that after the lane-change, the deceleration of the new follower vehicle a_n does not exceed a safety limit, $a_n \geq \beta_n$. Second, it validates via an incentive criterion that checks whether the local traffic situation of the car would improve, given a lane change. This incentive model also involves the immediately affected vehicles, i.e., the new and old follower vehicles. We evaluate the difference in acceleration for the current vehicle (Δa_c), the new follower (Δa_n), and the old follower (Δa_o), whereby a positive acceleration change for the current car is considered favorable. A politeness factor $p \in [0, 1]$ determines how much heed is payed to the new and old follower vehicles. The incentive to change lanes is controlled by a switching threshold Δa_T, which ensures that a certain advantage is achieved through the prospected lane-change maneuver. This is formalized by the following inequality:

$$\Delta a_c + p(\Delta a_n + \Delta a_o) > \Delta a_T. \tag{9}$$

IDM Parameter	*Normal*	*Aggressive*
Desired velocity v_0 [m/s]	0.4	0.4
Time headway T [s]	2.0	2.0
Max. acceleration α [m/s^2]	0.5	1.0
Desired deceleration β [m/s^2]	0.3	0.5
Acceleration exp. δ	4	4
Jam distance s_0 [m]	0.1	0.1
MOBIL Parameter	*Normal*	*Aggressive*
Politeness p	0.5	1.0
Safe breaking β_n [m/s^2]	$0.7 \cdot \alpha$	$0.7 \cdot \alpha$
Accel. thresh. Δa_T [m/s^2]	0.4	0.2

TABLE III: Minicar parameter values for lateral and longitudinal control, in a *normal* and an *aggressive* mode.

The parameter values that we used for both IDM and MOBIL controllers are detailed in Table III.

Remarks. Testing the aforementioned models on our experimental testbed demonstrated the need for adaptations that take into account the physical vehicles' kinematic and dynamic constraints.

Firstly, we noticed the the jam distance s_0 requires an additional *escape* distance, which is a function of the ego vehicle's desired speed and the front vehicle's current speed, v_f, in order facilitate lane merging. Hence, we used an effective jam distance $\hat{s}_0 = s_0 + s_e$, where s_e follows the rule:

$$s_e(v_f, v_0) = \begin{cases} 0, & \text{if } v_f / v_0 > 1 \\ 2L \left[2 \left(\frac{v_f}{v_0} \right)^3 - 3 \left(\frac{v_f}{v_0} \right)^2 + 1 \right], & \text{else.} \end{cases}$$

This ensures a larger escape distance when the speed of the front vehicle is small relative to the ego car's desired speed, and that this distance approaches zero as the speeds become equal. Secondly, a lane change on real cars is not immediate, and takes time to be completed. Hence, during this transition, cars on the original lane will still consider the state of ego vehicle in their longitudinal control, until the ego vehicle has completely entered the new lane. Similarly, the ego car starts using information about the state of new preceding and following cars on the new lane, as soon as it starts changing lanes. Finally, an extra safety constraint is added to the MOBIL model to check that the escape distance between the ego vehicle and the front vehicle is large enough to prevent crashing when changing lanes.

B. Cooperative Driving

Our cooperative driving strategy builds on the assumption that vehicles within visibility range c communicate with

one another to share *intended* maneuvers before actually executing them. This allows the vehicles to cooperate about lane-changing decisions, and hence, plan efficient paths that maximize traffic throughput, whilst ensuring safety. To test our setup's capability of validating the effects of cooperative driving we implement an approach that builds on the following key ideas:

- When the ego vehicle decides to change its lane, it communicates with its neighboring vehicles, projecting a virtual counterpart vehicle at the desired new state.
- Vehicles within communication range of the ego vehicle receive information about its projected (virtual) state. They take this information into account to accelerate or decelerate, as a function of their relative positions to the virtual vehicle.

We implement this behavior by modifying the original IDM and MOBIL models as follows.

***Cooperative*-IDM (C-IDM).** Our cooperative-IDM model takes *projected* vehicles into account in the form of weighted virtual vehicles. When a vehicle projects its desired state, the resulting virtual vehicle is given a weight w_v between 0 and 1 depending on how urgent the lane-change is. This urgency is defined by $w_v = \min(1, \kappa(c - s))$, where s is the gap between the actual ego vehicle (to whom the virtual vehicle belongs), and its preceding vehicle. Based on this idea, our cooperative IDM model differs from the original IDM model in two ways. First, our model *decelerates* when detecting a projected vehicle in front of it. It does this by returning an acceleration $a = \min(w_v \cdot \tilde{a}_{\text{IDM}}, a_{\text{IDM}})$, where a_{IDM} is the acceleration computed using an actual front vehicle (if present), and where $\tilde{a}_{\text{IDM}}$ is the acceleration computed using the projected front vehicle. Second, it *increases* the desired speed when in front of or too close to a virtual vehicle, so that it can make space for the desired, projected vehicle state of the ego vehicle that owns the virtual vehicle. This is formalized by the following rule:

$$\tilde{v}_0 = v_0 \left(1 + w_v \frac{c - \tilde{s}_{\text{trail}}}{c}\right)$$

where $\tilde{s}_{\text{trail}}$ is the distance between the neighboring vehicle and the virtual vehicle behind it.

***Cooperative*-MOBIL (C-MOBIL).** Our cooperative-MOBIL model uses a higher safe braking acceleration value than the original model in order to pack vehicles more closely when changing lanes; we set $\beta_n = \alpha$. Also, we add a safety constraint to prevent crashing when the speeds are low and the acceleration given by IDM is low: $s > s_0 + \gamma \Delta v$, where s is the gap between the ego vehicle and the front/rear vehicle on the nearby lane. The constant γ corresponds to the time needed for an average lane change.

C. Gamification

We gamify our experimental setup by interfacing one car (or several cars) with a joystick or keyboard. This allows a human player to experience traffic amid different types of surrounding vehicle behaviors. It also allows us to stress-test the reactions of other cars in traffic to arbitrary maneuvers.

A user can choose between three modes of control: 'manual', 'semi-automatic' and 'automatic'. 'Manual' gives a user direct control over the speed and turning angle of the car. 'Semi-automatic' confines the vehicle to the designated lanes and safety restrictions while allowing the user to select a maintained speed and change lanes. 'Automatic' removes all control from the player and integrates the vehicle into the traffic.

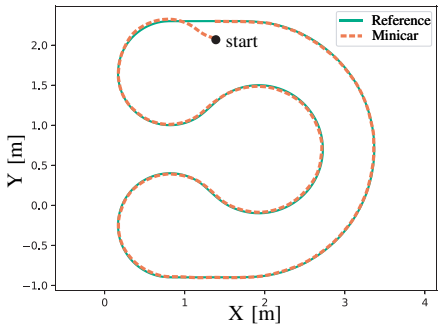

Fig. 6: Overhead representation of trajectory tracking accuracy for one loop of our U-shaped reference trajectory.

VII. Experiments

We perform three sets of experiments on our vehicle fleet. Our aim is to *(a)* demonstrate the navigation capabilities of our Minicar, *(b)* demonstrate the operation of the Minicar fleet, and *(c)* show how the fleet is used to test and validate (novel) driving algorithms in a realistic, albeit miniature, setup.

A. Trajectory Tracking

We validate the Minicar's trajectory tracking capabilities. We run the Minicar on our U-shaped reference trajectory at a nominal speed of 0.4m/s. Figure 6 shows an overhead plot of the tracking accuracy, for one loop on our U-shaped track. The average tracking error, measured over 7 loops of our track, is 14 mm with a standard deviation of 6.3 mm.

B. Driving Behaviors in Multi-Car Traffic

We implement four schemes to show the effect of different driving behaviors in multi-car traffic. We consider two driving policies: all cars are driven by *(1)* the egocentric policy (described in Sec. VI-A), or *(2)*, the cooperative policy (described in Sec. VI-B) with $c = 2\,\text{m}$. For each policy, we use either the *normal* or the *aggressive* parameter set (see Table III). An experiment involves 16 Minicars driving on our U-shaped track and runs for 200s. At the start, the Minicars are evenly spaced out over the track. After 20s, one of the cars is told to stop (and hence, it blocks traffic on its current lane). The aim is to observe how the four different schemes react to this disturbance.

Figure 7 shows the traffic flow for these four experiments, on the top row for the egocentric driving policy, and on the bottom row for the cooperative driving policy. In the egocentric scheme, we observe how the blockage creates a vehicle queue, which increases to 5 waiting vehicles at the latest stage. In contrast, the cooperative scheme overcomes vehicle queuing altogether (the traffic patterns exhibit very short stationary phases). With cooperative behavior, instead

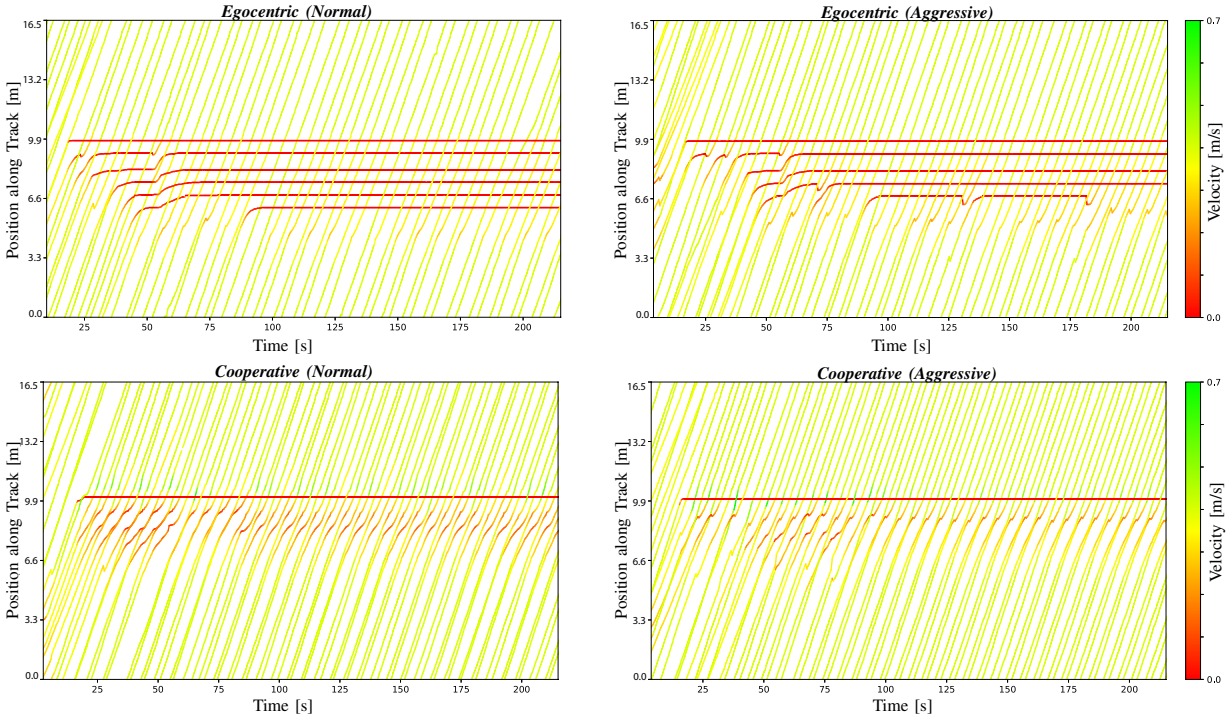

Fig. 7: Results for experiments performed on 16 Minicars driving on our multi-lane U-shaped track, for *egocentric* (top) and *cooperative* (bottom) driving policies. The panels show vehicle positions the track, plotted as a function of time. One car is stopped after 20s to cause a traffic disturbance. The colorbar shows the velocity recorded. The panels on the left show data for the *normal* parameter settings, panels on the right show the *aggressive* parameter settings (see Table III).

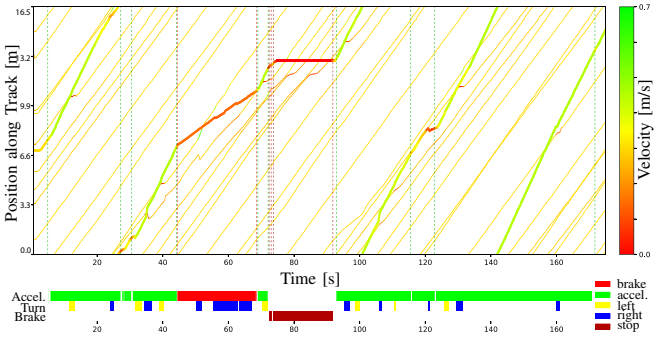

Fig. 8: A gamified experiment where a human controls one Minicar via a joystick, and 10 other Minicars are automated. The thick trajectory represents the played car. The colorbars along the time axis show the periods during which control commands are executed.

	Egocentric	**Cooperative**	Improvement
Normal	0.245 ± 0.036	0.330 ± 0.038	35%
Aggressive	0.277 ± 0.045	0.393 ± 0.211	42%

TABLE IV: Average throughput and standard deviation, measured in cars-per-second, for the four experiments shown in Figure 7. The third column shows the percent improvement of the cooperative scheme over the egocentric scheme.

of queuing, a Minicar communicates its intention to lane-change; following vehicles in the new lane reduce their speeds to make space for this projected maneuver, hence maintaining traffic flow whilst ensuring safety. The overall throughput is improved in the cooperative scheme, and is further enhanced by aggressive control parameters; the average throughput values are reported in Table IV.

C. Gamification

Figure 8 shows an example of a player-controlled car amid 10 automated cars, in cooperative mode. We observe how the traffic is affected by the actions of the human player.

VIII. Discussion

In this work, we provided the design of a fleet of miniature cars for research, education and outreach in the domain of automated multi-car systems. Our Ackermann-steering platform is one out of very few openly available designs; in particular, it fills a price-range gap, and is especially attractive for robotics labs that already possess telemetry infrastructure (such as motion capture). We propose a system architecture that is capable of integrating heterogeneous driving strategies (e.g., egocentric or cooperative) in a multi-lane setup. We demonstrate its applicability for large-fleet experimentation by implementing four different driving schemes that lead to quantifiable, distinct traffic behaviors.

Although our current setup considers off-board intelligence, the platform is easily extended with an IMU and camera to provide full on-board autonomy. In future work, we plan to use our fleet for testing multi-car systems in more complex scenarios that include: *(a)* road topographies with a larger number of lanes as well as intersections, *(b)* heterogeneous vehicle behaviors in mixed traffic, and *(c)* noisy sensing and delayed communications. Further research will investigate multi-objective optimization problems that also include driver comfort (as measured by vehicle accelerations).

REFERENCES

[1] CBS Insights Research Brief. https://www.cbinsights.com/research/autonomous-driverless-vehicles-corporations-list/. (Accessed August 15, 2018).

[2] Hemisson. https://www.robotsinsearch.com/products/hemisson. (Accessed August 13, 2018).

[3] Sribbler 3. https://www.parallax.com/product/28333. (Accessed August 13, 2018).

[4] M. Dekan, F. Duchon, et al. irobot create used in education. *Journal of Mechanics Engineering and Automation*, 3(4):197–202, 2013.

[5] F. Dressler, H. Hartenstein, O. Altintas, and O. Tonguz. Inter-vehicle communication: Quo vadis. *IEEE Communications Magazine*, 52(6):170–177, 2014.

[6] M. Ferreira, R. Fernandes, H. Conceição, W. Viriyasitavat, and O. K. Tonguz. Self-organized traffic control. In *Proceedings of the seventh ACM international workshop on VehiculAr InterNETworking*, pages 85–90. ACM, 2010.

[7] J. Gonzales, F. Zhang, K. Li, and F. Borrelli. Autonomous drifting with onboard sensors. In *Advanced Vehicle Control: Proceedings of the 13th International Symposium on Advanced Vehicle Control (AVEC16), September 13-16, 2016, Munich, Germany*, page 133, 2016.

[8] A. Jiménez-González, J. R. Martinez-de Dios, and A. Ollero. Testbeds for ubiquitous robotics: A survey. *Robotics and Autonomous Systems*, 61(12):1487–1501, 2013.

[9] S. Karaman, A. Anders, M. Boulet, J. Connor, K. Gregson, W. Guerra, O. Guldner, M. Mohamoud, B. Plancher, R. Shin, et al. Project-based, collaborative, algorithmic robotics for high school students: Programming self-driving race cars at MIT. In *Integrated STEM Education Conference (ISEC), 2017 IEEE*, pages 195–203. IEEE, 2017.

[10] S. Kernbach. Swarmrobot. org-open-hardware microrobotic project for large-scale artificial swarms. *arXiv preprint arXiv:1110.5762*, 2011.

[11] A. Kesting, M. Treiber, and D. Helbing. General lane-changing model MOBIL for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

[12] E. King, Y. Kuwata, M. Alighanbari, L. Bertuccelli, and J. How. Coordination and control experiments on a multi-vehicle testbed. In *American Control Conference, 2004. Proceedings of the 2004*, volume 6, pages 5315–5320. IEEE, 2004.

[13] M. Linderoth, K. Soltesz, and R. M. Murray. Nonlinear lateral control strategy for nonholonomic vehicles. In *American Control Conference, 2008*, pages 3219–3224. IEEE, 2008.

[14] A. Liniger, A. Domahidi, and M. Morari. Optimization-based autonomous racing of 1: 43 scale rc cars. *Optimal Control Applications and Methods*, 36(5):628–647, 2015.

[15] J. McLurkin, A. McMullen, N. Robbins, G. Habibi, A. Becker, A. Chou, H. Li, M. John, N. Okeke, J. Rykowski, et al. A robot system design for low-cost multi-robot manipulation. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 912–918. IEEE, 2014.

[16] L. Paull, J. Tani, H. Ahn, J. Alonso-Mora, L. Carlone, M. Cap, Y. F. Chen, C. Choi, J. Dusek, Y. Fang, et al. Duckietown: an open, inexpensive and flexible platform for autonomy education and research. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1497–1504. IEEE, 2017.

[17] D. Pickem, P. Glotfelter, L. Wang, M. Mote, A. Ames, E. Feron, and M. Egerstedt. The robotarium: A remotely accessible swarm robotics research testbed. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1699–1706. IEEE, 2017.

[18] J. A. Preiss*, W. Hönig*, G. S. Sukhatme, and N. Ayanian. Crazyswarm: A large nano-quadcopter swarm. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3299–3304. IEEE, 2017. Software available at https://github.com/USC-ACTLab/crazyswarm.

[19] F. Riedo, M. Chevalier, S. Magnenat, and F. Mondada. Thymio II, a robot that grows wiser with children. In *2013 IEEE workshop on advanced robotics and its social impacts (ARSO)*, pages 187–193. Eidgenössische Technische Hochschule Zürich, Autonomous System Lab, 2013.

[20] M. Rubenstein, C. Ahler, and R. Nagpal. Kilobot: A low cost scalable robot system for collective behaviors. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 3293–3298. IEEE, 2012.

[21] M. Rubenstein, B. Cimino, R. Nagpal, and J. Werfel. Aerobot: An affordable one-robot-per-student system for early robotics education. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 6107–6113. IEEE, 2015.

[22] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.

[23] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou. Aggressive driving with model predictive path integral control. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 1433–1440. IEEE, 2016.

[24] S. Wilson, R. Gameros, M. Sheely, M. Lin, K. Dover, R. Gevorkyan, M. Haberland, A. Bertozzi, and S. Berman. Pheeno, a versatile swarm robotic research and education platform. *IEEE Robotics and Automation Letters*, 1(2):884–891, 2016.