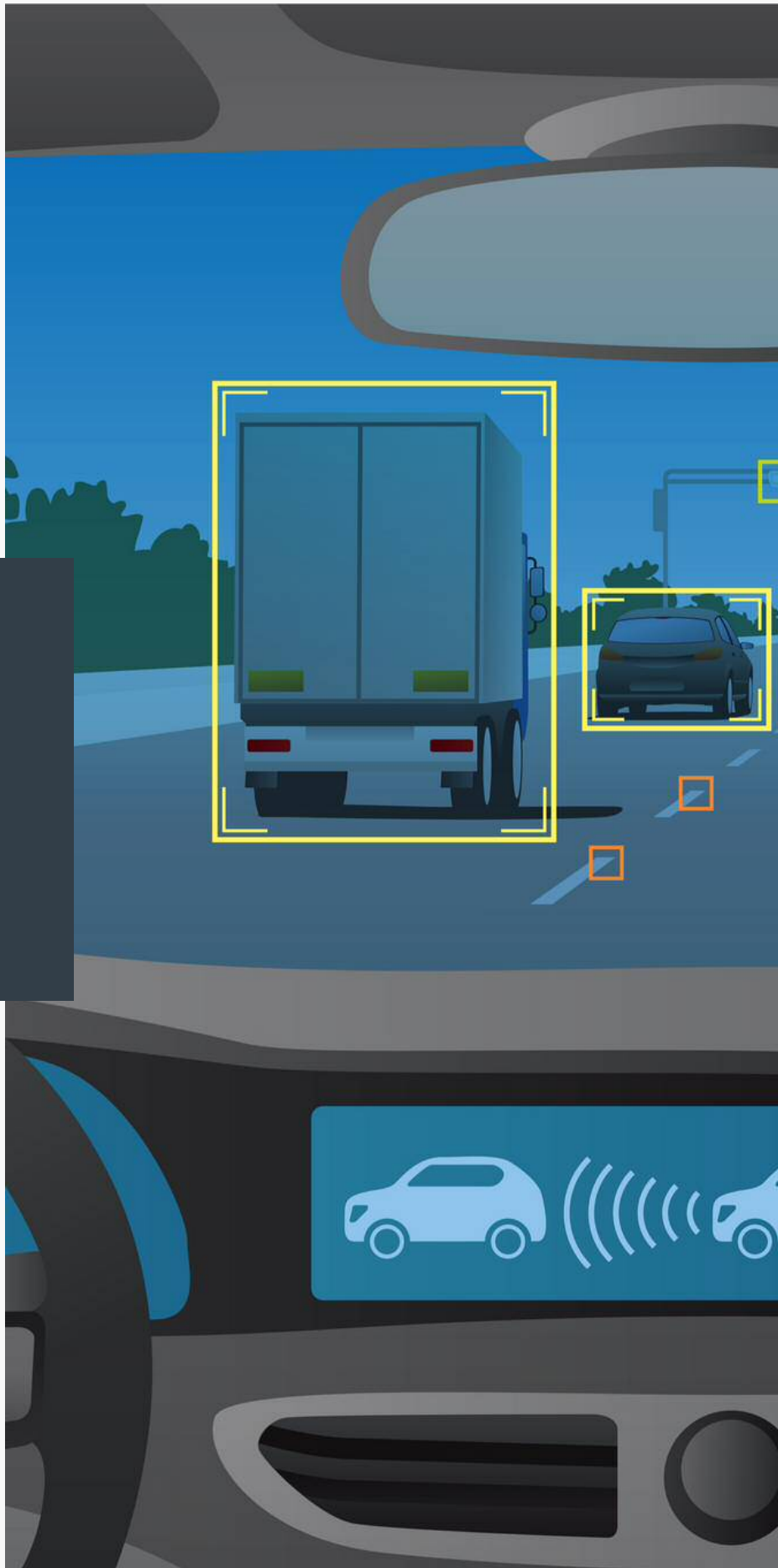
ASSURING AUTONOMY
INTERNATIONAL PROGRAMME

DEMONSTRATOR PROJECT

Final report

ATM Automatic Testing Mechanism



MARCH 2021

Towards an NCAP-like rating for Robotics and Autonomous Systems

Hanna Kurniawati (hanna.kurniawati@anu.edu.au)
School of Computing (fka. Research School of Computer Science)
Australian National University
5 March 2021

Project Description

As robotics and autonomous systems (RAS) start to become consumer products, questions arise on how to help consumers compare the safety of different RAS. This question motivates our project.

We aim to develop a mechanism that can automatically assess the safety of a RAS as a holistic system, with minimal reliance on information about the inner working of the RAS, and present this assessment in a user-friendly manner. For this project, we will focus on assessing catastrophic accidents.

Such a safety assessment is akin to the well-known NCAP safety rating for cars. However, the mechanism that generates the NCAP rating requires substantial modifications for RAS. NCAP rating has been designed to test the safety of the car's mechanics and hardware, whose behaviour generally changes gradually and very slowly –in the order of multiple years. In contrast, most of today's RAS is run by software that automatically adapts their behaviour –often in the order of days–, as more data are gathered and regular patches/updates are applied. Although these adaptations are supposed to improve performance, it is often unclear as to their effects on the fringe rare cases where catastrophic accidents often happen.

Euro-NCAP has started to assess the autonomy augmentation of new car models. However, these assessments still follow a rather ad-hoc mechanism, e.g., a mannequin is being "dragged around" following a certain fixed trajectory to test autonomous collision avoidance capabilities. The problem is, such an ad-hoc mechanism is easy to fool by a RAS once the sets of test scenarios are known.

To better test the fast adaptable nature of RAS, we propose an automatic testing mechanism that deliberately finds human behaviours that cause catastrophic accidents. The similarity between behaviours that cause catastrophic accidents and those of common human behaviour when interacting with the RAS indicates how likely catastrophic accidents happen, and can be used to compute the safety rating of the RAS.

Team member

- Jimy Cai Huang (Research Engineer, 0.5FTE)
- Hanna Kurniawati (Faculty member, PI, 0.15 FTE)

Summary of Results

1. Development of a novel mechanism to compute an NCAP-like rating of RAS. Details on this mechanism for (semi-)autonomous cars have been submitted as a paper to Robotics: Science and Systems (RSS) 2021. This work includes:
   (a) A novel mechanism that allow the rating to be computed within minutes, and is sufficiently general to be applied to a variety of robotics problems.
   (b) A novel safety indicator, based on a similarity measure between safe trajectories and unsafe trajectories closest to the safe trajectories. We also show that the probability a small deformation changes safe trajectories into unsafe ones is upper bounded by a value inversely proportional to the proposed safety indicator.
   (c) Validation of the mechanism on a high-fidelity simulator (Carla) for self-driving cars.

   We attached the content of this work as a technical report, titled "An NCAP-like Safety Indicator for Autonomous Cars". Although the paper focuses on autonomous cars, the testing mechanism and safety indicator are general enough to be applied to other types of RAS.

2. Contribution to the Australian Robotics Roadmap – Trust and Safety chapter.
3. Presentation of the initial work at the ANU Humanising Machine Intelligence Retreat on 19 November 2020, where we have drawn interest from researchers in the public policy field.
4. Presentation of the submitted RSS paper at the ANU Humanising Machine Intelligence group has been scheduled for 11 March 2021, where we hope to engage with researchers and practitioners from law and public policy.
5. Recruitment of a PhD student, Ivan Ang, to continue this work. He received an ANU CECS Scholarship. He is supposed to start in August 2020, but due to COVID-19, he has not gotten his visa yet. We hope he can start in August 2021.

Guidelines for Applying the Proposed Testing Mechanism

This section provides potential guidelines in applying the proposed testing mechanism to assess the safety of RAS.

We propose that after every software updates or patches that may change the behaviours of the RAS, the RAS is tested again. To this end, we propose the development of testing services, which can be performed similar to car wash services. This means, the user of RAS can perform the test at home, with a "do-it-yourself" kit, or bring the RAS to a testing station. The testing station itself can operate in a similar manner to a car wash service, where the user can leave their RAS to the testing station for a short time period (e.g., 15–30 minutes), or even stay in the car while testing is performed.

The test can be specific to a particular capability being updated (e.g., pedestrian avoidance) or extensive, which includes the entire system. Obviously, specific tests are faster than extensive tests. To ensure compliance of the self-test, regulators can impose that extensive tests be done at least once in n years, where n is to be decided by the regulators. One can imagine that the requirement is imposed together with the renewal of the car's registration.

The above guidelines assume that the tests are done in a high-fidelity simulator. Our proposed testing mechanism can be applied to the physical RAS directly too. However, obviously, such a test will require more resources, time, and unlikely to be doable at home. Furthermore, our proposed testing mechanism does require multiple runs for statistical confidence. We believe the best practice would be one that combines both simulation and physical RAS testing. However, further work are needed for such a combination. Questions include how to best combine them to gain the highest statistical confidence with the lowest number of data, what scenarios should be done in simulation and what scenarios should be done on the physical RAS, etc..

An NCAP-like Safety Indicator for Self-Driving Cars

Jimy Cai Huang Hanna Kurniawati
School of Computing (fka. Research School of Computer Science)
Australian National University
Email: {jimy.cai, hanna.kurniawati}@anu.edu.au

Abstract

This paper proposes a mechanism to assess the safety of autonomous cars. It assesses the car's safety in scenarios where the car must avoid collision with an adversary. Core to this mechanism is a safety measure, called Safe–Kamikaze Distance (SKD), which computes the average similarity between sets of safe adversary's trajectories and kamikaze trajectories close to the safe trajectories. The kamikaze trajectories are generated based on planning under uncertainty techniques, namely the Partially Observable Markov Decision Processes, to account for the partially observed car policy from the point of view of the adversary. We found that SKD is inversely proportional to the upper bound on the probability that a small deformation changes a collision-free trajectory of the adversary into a colliding one. We perform systematic tests on a scenario where the adversary is a pedestrian crossing a single-lane road in front of the car being assessed —which is, one of the scenarios in the Euro-NCAP's Vulnerable Road User (VRU) tests on Autonomous Emergency Braking. Simulation results on assessing cars with basic controllers and a test on a Machine-Learning controller using a high-fidelity simulator indicates promising results for SKD to measure the safety of autonomous cars. Moreover, the time taken for each simulation test is under 11 seconds, enabling a sufficient statistics to compute SKD from simulation to be generated on a quad-core desktop in less than 25 minutes.

1 INTRODUCTION

Safety of robotics and autonomous systems, specifically autonomous cars, have become increasingly important. Throughout this paper, the term *autonomous* includes semi-autonomous systems too. Many work have been proposed to improve the safety of autonomous cars. Most focus on developing autonomous car components (e.g., control and machine learning) with safety guarantees (e.g., [12, 21, 23, 26]). This paper aims to explore an orthogonal issue, namely the safety assessment.

Recent work have started to focus on developing testing mechanisms to assess the safety of autonomous cars. Most work in this direction focus on identifying critical testing scenarios[6, 18]. Since accidents have a small percentage, identifying assessment scenarios that lead to accidents, especially catastrophic accidents, is difficult. Rare event simulations[18] and a variety of adversary generation strategies[6, 24, 25] have been proposed to identify such scenarios. Our proposed safety assessment mechanism can benefit from these work too.

However, in this paper, the purpose of our testing mechanism is not to identify problematic scenarios per se. Rather, we aim to develop a testing mechanism that can eventually help users to easily compare the safety of different autonomous cars, including different versions of the software that run them. Such a safety indicator is akin to the New Car Assessment Program (NCAP) safety rating that has helped users with non-autonomous cars. However, since NCAP testing scenarios are mostly static and performed at most once in

the lifetime of a car, we do need to adjust the testing mechanism and safety indicator to be adaptive and efficient enough, such that they are suitable for autonomous systems and frequent assessment is viable.

Our proposed testing mechanism is based on the observation that safe autonomous cars must provide sufficient room for errors and uncertainty, in particular due to non-deterministic effects of actions. For instance, different drivers' reaction time in taking over control to avoid colliding with a pedestrian may result in different outcomes, different road and tyres conditions may result in unexpected collision with a pedestrian, etc.. Therefore, we propose to measure the safety of an autonomous car based on how likely will a safe scenario change into a dangerous one under a small perturbation of the scenario.

To that end, our mechanism will assess the car's safety in scenarios where it must avoid collision with an adversary. Core to our mechanism is a safety measure, called Safe–Kamikaze Distance (SKD), which is based on the average similarity measure between safe and kamikaze trajectories of the adversary when interacting with the car being assessed. A kamikaze trajectory is an adversary's trajectory that causes the adversary to collide with the car being tested. Given a safe trajectory for the adversary, our mechanism computes multiple kamikaze trajectories closest to the safe trajectory. The safety measure SKD is then the average distance between samples of pairs of safe and kamikaze trajectories. We show the probability that a small deformation changes a collision-free trajectory of the adversary into a colliding one is upper bounded by a value inversely proportional to SKD.

Our assessment mechanism is general enough to be used with a variety of adversaries, but in this paper, our experiments focus on scenarios where the adversary is a pedestrian crossing a single-lane road in front of the assessed car —one of the scenarios in the Euro-NCAP's Vulnerable Road User (VRU) tests on Autonomous Emergency Braking[17]. Results on basic controllers and a Machine Learning controller as provided by the Carla[8] simulator indicate that SKD increases as collision rate decreases. Moreover, our results indicate the time required to compute SKD makes the assessment mechanism to be potentially viable to be performed frequently, such as after every software updates.

2 Related Work and Background

2.1 Related Work

To ensure safety of autonomous cars, many work have focused on using formal methods for verification of the autonomous vehicle system (e.g., [3, 26]). A short summary of formal methods for autonomous cars is provided in[22]. These approaches require formal specifications, which is often not easy to construct completely due to the complexity of the system and the sheer possibilities of the different scenarios that an autonomous car may encounter.

Another line of work is to develop testing mechanisms to assess the safety of autonomous cars. Recently, work in this direction have focused on generating test scenarios that will lead to accidents[6, 18, 24, 25]. This problem is difficult because accidents are relatively rare.

Obviously, a mechanism to test the safety of a car is not new. The well-accepted NCAP testing protocol[1] was introduced in 1979. However, testing scenarios developed by NCAP are mostly static, which is not suitable for autonomous cars. In this paper, we propose a testing mechanism that can utilise these static scenarios as safe trajectories, and then find a kamikaze trajectory close to these safe trajectories to compute SKD. We hope such a mechanism would be more acceptable for regulatory purposes.

To generate safe and kamikaze trajectories, the testing mechanism needs to have a predictive model of the car's behaviour, albeit imperfect. For this purpose, many work can be adopted, such as [4, 10, 14, 20]. Our kamikaze trajectory generation can also benefit from work on pursuit evasion[7] and more recently [16], though the latter is focused on deforming observations rather than an adversary's behaviour.

2.2 Background

Two main components in our proposed mechanism is the safety measure SKD, which relies on Fréchet distance, and the kamikaze trajectory generation, which is based on the Partially Observable Markov Decision Processes (POMDPs). The following are backgrounds on these two concepts.

2.2.1 Fréchet Distance

Fréchet distance[11] measures the similarity between two curves while adhering to the order of points on the curve. Suppose $P : [0, 1] \to \mathbb{R}^n$ and $Q : [0, 1] \to \mathbb{R}^n$ are two curves on the same space. Then the Fréchet distance between these two curves are:

$$d(P, Q) = \inf_{\alpha,\beta} \max_{t\in[0,1]} \|P(\alpha(t)) - Q(\beta(t))\| \tag{1}$$

among all possible $\alpha : [0, 1] \to [0, 1]$ and $\beta : [0, 1] \to [0, 1]$, which are continuous reparameterisations of P and Q (respectively) that are non-decreasing and surjective.

For computational efficiency, in this paper, we use the approximation of eq. (1) via discrete Fréchet distance, approximating P and Q as polygonal chains, resulting in the following definition. Suppose $P' : (p_1, p_2, p_3, \cdots, p_m)$ and $Q' : (q_1, q_2, q_3, \cdots, q_{m'})$, where $p_i, q_j \in \mathbb{R}^n$ for $i \in [1, m]$, $j \in [1, m']$. Then the Fréchet distance between these two polygonal chains are:

$$\begin{aligned} &d(P', Q') = \min_{k,l} \max_{t} \|p_{k(t)} - q_{l(t)}\| \\ &\text{where:} \\ &(i)\ k(t + 1) = k(t) + 1 \text{ and } l(t + 1) = l(t), \text{ or} \\ &(ii)\ k(t + 1) = k(t) \text{ and } l(t + 1) = l(t) + 1 \end{aligned} \tag{2}$$

Suppose $m \geq m'$, then $t \in [1, m]$, while $k(t)$ and $l(t)$ maps the index t to an index of the points in P' and Q', respectively. This distance can be computed in $O(\frac{mm' \log\log m}{\log m})$ time and $O(m + m')$ space[2].

2.2.2 POMDP

To generate kamikaze trajectories, we need to consider the policy of the car being assessed. However, depending on the car's software system, this policy is not always explicit and may not even be known exactly prior to execution. Therefore, to the kamikaze trajectory generator, the car's policy is partially observed. Hence, we apply the POMDP to generate kamikaze trajectories.

Formally, a POMDP is a tuple $\langle S, A, T, O, Z, R, \gamma\rangle$ [15]. At each time-step, a POMDP agent is in a state $s \in S$, executes an action $a \in A$, perceives an observation $o \in O$, and moves to the next state $s' \in S$. The next state is distributed according to $T(s, a, s')$, which is a conditional probability function $P(s'|s, a)$ that represents non-deterministic actions. The observation o perceived depends on the observation function Z, which is a conditional probability function $P(o|s, a)$ that represents uncertainty in sensing. The notation R is a reward function, from which the objective function is derived. The notations $\gamma \in (0, 1)$ is the discount factor to ensure that an infinite horizon POMDP problem remains a well-defined optimisation problem.

The solution to a POMDP problem is an optimal policy π^*, that maps beliefs to actions in order to maximise the expected total reward, i.e. $V^*(b) = \max_{a\in A} [R(b, a) + \gamma \sum_{o\in O} Pr(o|a, b)V^*(\tau(b, a, o))]$, where $R(b, a) = \sum_{s\in S} R(s, a)b(s)$ and $Pr(o|a, b) = \sum_{s'\in S} Z(s', a, o) \sum_{s\in S} T(s, a, s')b(s)$. The function τ computes the updated belief after the agent executes a from b and perceives o.

3 Overview of the Assessment Mechanism

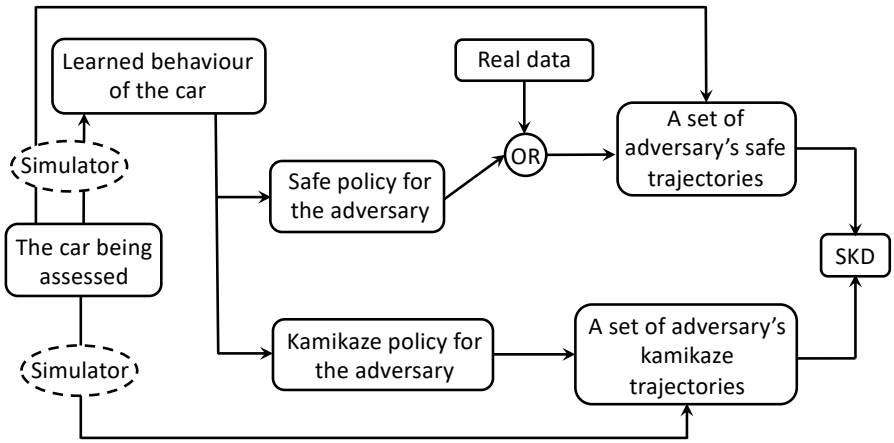


Figure 1: Proposed mechanism to assess the safety of an autonomous car. Dashed ellipse means it may or may not be used.

Figure 1 illustrates our proposed assessment mechanism. This mechanism automatically evaluates the safety of an autonomous car's software system as a holistic system. It accepts the software system of the car being assessed and uses a high-fidelity simulator or physical experiments to measure the car's safety under a testing scenario. Testing on the physical system directly is also applicable, for instance using a pedestrian dummy or robot. However, we believe such a test should be supplemented with simulation tests because most probably, we will only be able to perform a small number of physical trials and they will likely incur substantial cost.

A variety of testing scenarios can be used. The only requirement is that they must involve interactions between the car being assessed and an adversary (e.g., a pedestrian or another car), such that the adversary can crash into the assessed car. Specifically, suppose $Adv = \langle E, S_{adv}, A_{adv}, F_{adv} \rangle$ defines an adversary, where:

- E is a bounded environment, where the adversary and assessed car operates. Positions of objects in E is defined in a bounded world coordinate space $W \subset \mathbb{R}^2$.
  Note that our mechanism is sufficiently general for W to be in $\mathbb{R}^n$, and hence conceptually, the mechanism can be applied to a variety of robotics systems, such as Unmanned Aerial Vehicles, Autonomous Underwater Vehicles, and even manipulators, though further work on efficient computation of the components are required.

- S_{adv} is the set of states of the adversary, which includes the adversary's position in the world frame W. The adversary's position is specified as the position of the 2D projection of the adversary's centre of mass in W.

- A_{adv} is the set of actions the adversary can perform.

- $F_{adv}(s_{adv}, a_{adv}, t', \Delta_{t'}, \mathcal{P}_{adv})$ is a stochastic model of the adversary's dynamics function, which outputs a possible next state for the adversary after the action $a_{adv} \in A_{adv}$ is performed from state $s_{adv} \in S_{adv}$ at time t' for time duration $\Delta_{t'}$ and perturbed by an error distribution $\mathcal{P}_{adv}$.

And, suppose the assessed car is defined as $Car = \langle S_{car}, A_{car}, F_{car} \rangle$ where:

- S_{car} is the set of states of the car being assessed, which includes the car's position in the world frame W. The car's position is specified as the position of the 2D projection of the car's centre of mass in W.

- A_{car} is the set of actions the assessed car can perform.
- $F_{car}(s_{car}, a_{car}, t, \Delta_t, \mathcal{P}_{car})$ is the assessed car's dynamics function, which outputs a possible next state of the assessed car after the action $a_{car} \in A_{car}$ is performed from state $s_{car} \in S_{car}$ at time t for a duration of Δ_t and perturbed by an error distribution $\mathcal{P}_{car}$. This function may be a simplified function that is far from perfect.

Then, the testing scenario must include the environment, states, and actions such that $F_{car}(s_{car}, a_{car}, t, \Delta_t, \mathcal{P}_{car})$ may collide with $F_{adv}(s_{adv}, a_{adv}, t', \Delta_{t'}, \mathcal{P}_{adv})$ with substantial probability.

Any testing scenario that satisfies the above requirement can be used. To make the scenario concrete, throughout this paper, we use the scenario where a pedestrian is crossing a single-lane road in front of the assessed car, which is similar to a scenario for testing autonomous emergency braking systems in Euro-NCAP[17].

Given a testing scenario and an assessed car, our mechanism learns a predictive behaviour model of the assessed car under the scenario. The learning itself can be separately or in conjunction with kamikaze and safe trajectories generation of the adversary, via a high-fidelity simulator such as Carla[8] or direct interaction with the assessed car. Many work[4, 10, 14, 20] have been proposed to learn a predictive model of a self-driving car, and can be used with our mechanism.

This learned model is used in generating kamikaze trajectories of the adversary. When real data of the adversary's safe trajectories are not available, the model is also used to generate the safe trajectories. Details of these trajectory generations are provided in Section 5.

Once the sets of safe and kamikaze trajectories are generated, our assessment mechanism computes the average Fréchet distance between safe and kamikaze trajectories. This average distance can be used as a safety indicator, on how likely a safe trajectory becomes a dangerous one under a small deformation. Details on this distance computation and how it relates to the safety measure of the car being assessed is discussed in the next section.

4 Safe–Kamikaze Distance (SKD)

Suppose Φ is the set of safe adversary's trajectories, specified in the world frame W. Let $\Psi(\phi)$ be the set of kamikaze trajectories generated to be as close as possible to the safe trajectory $\phi \in \Phi$, where $\psi \in \Psi(\phi)$ is specified in the world frame W, and $\Psi = \bigcup_{\phi \in \Phi} \Psi(\phi)$. Ideally, each $\Psi(\phi)$ contains kamikaze trajectories with exactly the same distance to ϕ, as we take the closest kamikaze trajectories. However, in practice, we approximate by sampling kamikaze trajectories within a pre-defined maximum distance from the safe trajectory. Details on this generation is in Section 5.1. We then compute the SKD between these two sets of trajectories as:

$$SKD(\Phi, \Psi) = \frac{1}{|\Psi|} \sum_{\phi \in \Phi} \sum_{\psi \in \Psi(\phi)} d(\phi, \psi) \tag{3}$$

where $d(\phi, \psi)$ is the discrete Fréchet distance (eq. (2)) between the two trajectories. To ensure this more efficient Fréchet distance computation can be applied, we assume that trajectories in Φ and Ψ has been discretized uniformly in the time domain, which means the time to move between two consecutive points in a trajectory are the same everywhere.

Notice that based on the definition of Fréchet distance, $d(\phi, \psi) = \delta_{\phi,\psi}$ implies that $\forall_{p \in \phi} \exists_{q \in \psi}\ q \in B(p, \delta_{\phi,\psi})$, where $B(p, \delta_{\phi,\psi}) \subset \mathbb{R}^2$ is a ball centred at p with radius $\delta_{\phi,\psi}$. Therefore, $SKD(\Phi, \Psi) = \delta$ implies that on average, a safe adversary trajectory, sampled from the same distribution used to sample Φ, can change into an unsafe trajectory after being deformed for less than or equal to δ distance away. Moreover, the

probability of such a change happening after a very small deformation can be upper bounded by a function of δ. Specifically,

Theorem 1 *Suppose $SKD(\Phi, \Psi) = \delta$ and D is the random variable representing the Fréchet distance $d(\phi, \psi)$ where $\phi \in \Phi$ and $\psi \in \Psi(\Phi) \in \Psi$. Then for a small real number $\eta \in (0, \delta)$, $P(d(\phi', \psi') \leq \eta) \leq \frac{Var(D)+2\eta\delta}{Var(D)+\delta^2}$, where ϕ' and ψ' are any safe and kamikaze trajectories, sampled from the same distribution used to generate Φ and Ψ, respectively.*

Proof Since all trajectories in Φ and Ψ are specified in a bounded space W, D is a bounded random variable, and therefore based on Popoviciu inequality, $Var(D)$ is finite. Furthermore, since D represents distance, $D \geq 0$. These two properties allow us to apply the Paley-Zygmund inequality on D to obtain:

$$\begin{aligned} P(d(\phi', \psi') < \eta) &= 1 - P\left(d(\phi', \psi') \geq \frac{\eta}{\delta}\delta\right) \\ &\leq 1 - \frac{(1 - \frac{\eta}{\delta})^2\delta^2}{Var(D) + \delta^2} \\ &= 1 - \frac{(\delta - \eta)^2}{Var(D) + \delta^2} \\ &= \frac{Var(D) + 2\eta\delta - \eta^2}{Var(D) + \delta^2} \\ &\leq \frac{Var(D) + 2\eta\delta}{Var(D) + \delta^2} \end{aligned}$$

giving the desired upper bound.

Although $Var(D)$ appears in the above bound, we only propose SKD as a safety indicator. The reason is two folds. First, estimating variance is harder than estimating expected value.

The second reason is since D is bounded and $E[D]$ is finite, $Var(D)$ can be made sufficiently small by increasing the size of Φ and $\Psi(\phi)$ for each $\phi \in \Phi$, thereby allowing the probability $P(d(\phi', \psi') \leq \eta)$ to be bounded from above by a function that depends only on η and δ. For instance, setting the above set of trajectories such that $Var(D) \leq 2\eta$ will further bound the probability $P(d(\phi', \psi') \leq \eta) \leq 2\left(\frac{\eta}{\delta^2} + \frac{\eta}{\delta}\right)$.

Now, although one can reduce $Var(D)$ to be arbitrarily small, the number of samples to ensure $Var(D)$ is sufficiently small varies between one car and another. The reason is $Var(D)$ is affected by uncertainty of the assessed system, including the variance on reaction time of the human driver if the car is semi-autonomous, uncertainty due to different road conditions, etc. too.

Interestingly, the car's uncertainty is accounted in SKD too: Assuming all other conditions are the same, a car with larger stochastic uncertainty in its dynamics and perception will generate larger SKD. This result may seem counter-intuitive, especially when the stochastic uncertainty over the effects of actions and observation are symmetric, considering SKD is an expected value. However, since our mechanism only computes the distance between a safe trajectory and kamikaze trajectories that are as close as possible to the safe trajectory, the symmetric effect is filtered out. Therefore, SKD will decrease as uncertainty increases, even if stochastic uncertainty in the effects of actions and observations are symmetric around its mean.

Intuitively, the above results show that SKD can be used to indicate the safety of an autonomous car. A car with large SKD will have a small upper bound on the probability that an η small deformation can turn a sampled safe trajectory of an adversary into an unsafe one.

5 Generating Adversary's Trajectories

Given information about the environment and adversary $Adv = \langle E, S_{adv}, A_{adv}, F_{adv} \rangle$ and the assessed $Car = \langle S_{car}, A_{car}, F_{car} \rangle$, to compute SKD, our mechanism requires sets of safe and kamikaze trajectories of the adversary. To generate each set of trajectories, we construct a decision-making agent of the adversary, assuming that it has full observability about its own state and partial observability about the assessed car's policy. To account for partial observability of the car, the decision-making agent that represents the adversary is framed as a POMDP agent.

If real data on safe trajectories or static trajectories from a regulatory body are available, the safe trajectories can use these trajectories too. However, if they are not available, the safe trajectories can be generated using POMDP.

The following subsections describe the details of these POMDP models.

5.1 Generating Kamikaze Trajectories

Given a collision-free trajectory of the adversary $\phi \in \Phi$, the kamikaze agent generates a strategy that collides itself with the assessed car (under the test scenario used) as fast as possible while minimizing the total distance between its trajectory and ϕ. The agent has full observability about itself and potentially imperfect information about the car's dynamics, based on $F_{car}(s_{car}, a_{car}, t, \Delta_t, \mathcal{P}_{car})$ in Car. However, it only has partial observability about the policy of the assessed car. Therefore, this kamikaze trajectory generation is a form of pursuit evasion under partial observability problem, and our mechanism uses POMDP to generate the policy. A kamikaze trajectory is then the traces of the adversary's positions in a single simulation run of the POMDP policy.

To compute a kamikaze strategy, the kamikaze agent maintains a simplified predictive model of the assessed car $\widehat{Car} = \langle \widehat{S_{car}}, A_{car}, \widehat{F_{car}} \rangle$, where $\widehat{S_{car}} \subset S_{car}$ and $W_{car} \subseteq \widehat{S_{car}}$, and $\widehat{F_{car}}(s_{car}, a_{car}, t, \Delta_t, \mathcal{P}'_{car}) = F_{car}(s_{car}, a_{car}, t, \Delta_t, \mathcal{P}_{car}) + \mathcal{P}'_{car}$, where $s_{car} \in \widehat{S_{car}}$, $a_{car} \in A_{car}$, and $\mathcal{P}'_{car}$ is a probability distribution function representing the fact that the car dynamic F_{car} provided in Car can be far from perfect. This distribution function is learned from data, which can be from simulation provided by the car's developers, or via data from running the physical car. In this paper, they are learned using a simple maximum likelihood method on data gathered from running a simulated car in the high fidelity simulator, Carla[8].

Suppose the POMDP $\mathcal{P}^\dagger = \langle S^\dagger, A^\dagger, T^\dagger, O^\dagger, Z^\dagger, R^\dagger, \gamma^\dagger \rangle$ represents the kamikaze agent. The state space $S^\dagger = S^\dagger_{adv} \times \widehat{S_{car}} \times \Pi^\dagger_{car}$, where $S^\dagger_{adv} \subseteq S_{adv}$, $W_{adv} \subseteq S^\dagger_{adv}$. The kamikaze POMDP agent models a simplified policy of the assessed car as a parametric function, and $\Pi^\dagger_{car}$ represents the parameters of these policies. Note that by representing these parameters as a state variable, the uncertainty of the policy will be represented in the belief of the kamikaze agent.

The action space $A^\dagger = A_{adv}$. The transition function represents uncertainty in the resulting state of the adversary and assessed car. The transition function for the adversary and assessed car is conditionally independent given the current state. The adversary's transition function is defined as $F_{adv}(s_{adv}, a_{adv}, t', \Delta_{t'}, \mathcal{P}_{adv})$, while the car's dynamic follows $\widehat{F_{car}}(s_{car}, a_{car}, t, \Delta_t, \mathcal{P}'_{car})$.

The observation spaces and functions of the adversary depend on the type of sensors provided. However, the kamikaze agent assumes that the car model has full observability.

The reward function $R^\dagger$ is designed to encourage collision with the assessed car to happen while minimising distance of the adversary's position from the safe trajectory. To this end, we set the reward function, such that high reward is given when a collision between the adversary and assessed car happen, and higher penalty is given proportional to the distance between the adversary's position and Φ in W.

5.2 Generating Safe Trajectories

The POMDP agent $\mathcal{P}^\circ = \langle S^\circ, A^\circ, T^\circ, O^\circ, Z^\circ, R^\circ, \gamma^\circ \rangle$ used to generate safe policies for the adversary is very similar to that used to generate the kamikaze policies. The only different is in R°, the reward function is used to encourage the adversary to reach its intended destination as fast as possible without collision with the car. This optimality assumption follows the hypothesis in biology that human and other living beings generally tend to optimise their objective functions[5], though in general the objective functions being optimised are unclear. In our test mechanism, we assume the objective function follows the objective function of the POMDP $\mathcal{P}^\circ$.

Note that the safe trajectories do not need to use the same car model as the one being assessed nor the one used to generate the kamikaze trajectories. After all, these synthetically generated safe trajectories are supposed to replace real world trajectories data of the adversary, which may not be operating against the assessed car.

6 Experiments

The aim of our experiments is two folds. First is to understand how reasonable our SKD as an indicator of car safety. Second is to understand the required time for our proposed mechanism to output this safety indicator.

6.1 Scenarios

To achieve our experimental goals, we use the high level scenarios of avoiding collision with a pedestrian crossing the street, when the car is moving forward in a single lane. This scenario is similar to one of the scenarios used to test Autonomous Emergency Breaking systems in Euro-NCAP[17].

6.1.1 The Cars

We assume the car is moving forward in a single-lane road when a pedestrian is crossing the road. We test our proposed testing mechanism on two types of car controllers.

The first type is the set of *basic controllers*. This type of controllers is designed to systematically test SKD. These controllers assume the car starts from a given maximum velocity $8\frac{1}{3}m/s$ and apply the following policy:

$$\Pi^{\dagger}_{car}(W_{car}, W_{adv}) =$$

$$\begin{cases} acc_{car} = -3.5m/s^2 + \mathcal{U}[-0.1acc_{car}, 0.1acc_{car}] \\ \qquad\qquad \text{longitudinal distance}(W_{car}, W_{adv}) \leq C \times \kappa \\ acc_{car} = 0 \qquad\qquad\qquad\qquad\qquad\qquad \text{Otherwise} \end{cases}$$

where acc_{car} is the acceleration applied for a duration of $0.3s$ and $\mathcal{U}[-0.1acc_{car}, 0.1acc_{car}]$ is uniform distribution with support $[-0.1acc_{car}, 0.1acc_{car}]$, representing the car's uncertainty in the exact deceleration it performs. In addition to this uncertainty, the car's velocity is influenced by uncertainty too, such that the evolution of its speed is governed by $v_{t+1} = v_t + \mathcal{U}[-0.05v_t, 0.05v_t] + 0.3acc_{car}$. The notation κ is a safe distance threshold, defined as the distance for the car to move from the maximum velocity v_M m/s to 0 m/s, assuming maximum deceleration (in this case, $3.5m/s^2$) is applied and the car's motion is deterministic. This type of controller can be made more or less aggressive by applying a different multiplier C. Higher C means the controller has more margin of error, and therefore is safer. For our experiments, we apply 9 different values for C: $\{0.5, 0.625, 0.75, 0.875, 1.0, 1.05, 1.1, 1.125, 1.15\}$. The lower C is, the more aggressive

its behaviour is. Simulation run for this basic controllers are run on a simple C++ implementation based on the given model.

The second type of the car controller is the Machine-Learning based controller (ML) [19], which became a default controller of Carla. For our purpose, we ran the same controller, but focus only on two of the longitudinal control states *Cruising* and *Hazard Stop*. The other longitudinal control states [following, red light, and over limit] were not relevant for the scenarios in our experiments. This controller is ran on Carla v.0.9.6, that allows pedestrian control for adversary interactions.

6.1.2 The Adversary

The adversary in this scenario is a pedestrian crossing a single-lane road in front of the car. We assume when the pedestrian moves, the pedestrian is moving with a constant velocity of $2.5m/s$. We also assume that the pedestrian motion is deterministic. It perceives observation on the distance between itself and the assessed car, but this observation is noisy.

6.2 Trajectories Generators

6.2.1 Safe Trajectories

We use two sets of safe trajectories for the pedestrian. The first set of safe trajectories is a real world data set, which is extracted from the scenario *Lateral Interaction (Unilateral)* published in [27].

The second set of safe trajectories is synthetically generated using POMDP as described in Section 5.2, using a high fidelity simulator Carla v.0.9.6[8] when the car is controlled using the Machine Learning controller[19]. To generate this trajectory, the pedestrian actions are deterministic, with the action space being $A^{\circ} = \langle North, South, East, NorthEast, SouthEast, Stay \rangle$, where the first five actions in the set correspond to movement actions of the agent along the cardinal direction with a displacement magnitude of $|d| = 2.5m/s$, and the last action allows the adversary to remain in its position. The car model used by the POMDP agent is based on maximum likelihood, learned using simulated data from Carla v.0.9.6. The adversary observes the distance between itself and the car, with noise given by Gaussian distribution $\mathcal{N}(0, 1)$.

6.2.2 Kamikaze Trajectories

The kamikaze trajectories are generated using POMDP as described in Section 5.1.

The pedestrian model of this POMDP is the same as that used by the POMDP to generate safe trajectories. However, the model of the assessed car needs to be in-line with Car.

For the basic controller, the POMDP agent knows the controller's multiplier C, but not the full uncertainty plaguing the car controller (described in Section 6.1.1. Specifically, it only models the velocity uncertainty and not the acceleration uncertainty in its car model. Furthermore, it has a noisy estimate of its relative position to the assessed car.

For the ML controller, we use the same learned car model used by the POMDP to generate safe trajectories.

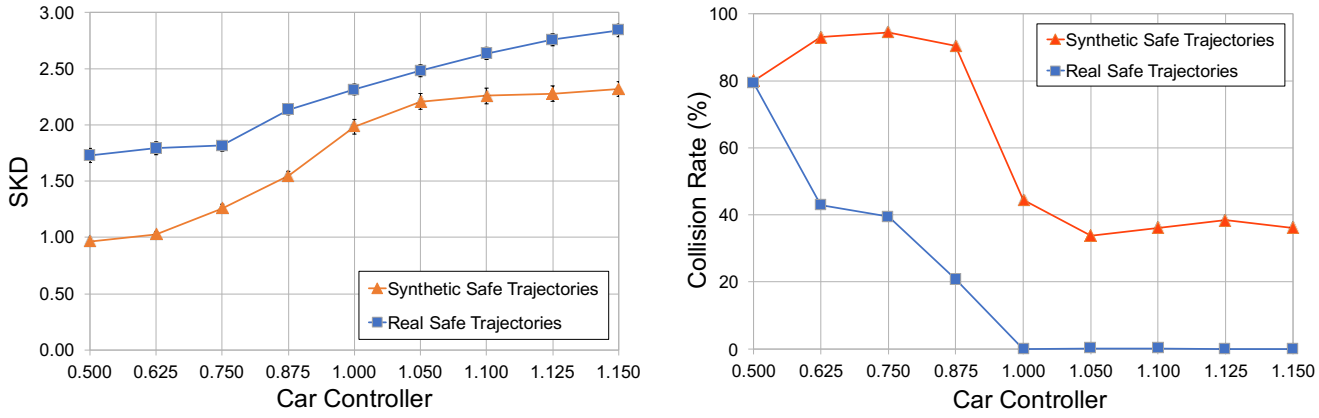
Last but not least, when real data is used as the safe trajectory, we adjusted the car model to the data. We adjusted the car's geometry to match the car in the real data. The adjusted size is smaller than the car model used in the basic controller. Therefore, we also adjusted the parameters of its dynamics to ensure that collision may still happen.

6.3 Setup

All trajectory generations, simulation runs, and SKD computation were ran on a desktop with an Intel Core i7-8700 @ 3.20 GHz CPU and 32GB RAM and NVidia GeForce GTX1060 with 6GB dedicated RAM. The GPU is used only for Carla simulation. To generate POMDP-based safe and kamikaze trajectories, we use OPPT[13], which is a POMDP toolkit for on-line POMDP solving, designed to ease applying POMDPs to robot planning problems. To compute SKD, we use the algorithm and implementation as described in[9].

6.4 Results

To assess SKD systematically, we applied our testing mechanism to assess the basic car controller with varying multiplier C, as described in Section 6.1.1. For this assessment, for each C value used and each type of safe trajectories (real and synthetic data), we generated 5 safe trajectories and 100 kamikaze trajectories per safe trajectory. We then computed SKD for each basic controller and each type of safe trajectories by averaging the Fréchet distance of the 500 pairs of safe and kamikaze trajectories. The SKD and its 95% confidence interval, along with its collision rate are presented in Figure 2(a).



(a) (b)

Figure 2: SKD and collision rate results.

They indicate that SKD increases as C increases, regardless of how the safe trajectories are generated, which is in-line with our design that as C increases, the controller has more margin of error, and hence are safer.

For verification purposes on how safe or dangerous the different controllers are, for each C value used and each safe trajectories, we also compute the collision rate between 500 simulations of the car *Car* moving forward, while the pedestrian follows the safe trajectories to cross the road. The results are presented in Figure 2(b).

It is interesting to compare SKD across the two types of safe trajectories. The consistently higher SKD in the safe trajectories extracted from real data is mostly in-line with the consistently lower collision rate of this set of scenarios.

This above trend only differs when $C = 0.5$. The collision rate for this particular controller when the safe trajectories are extracted from real data is only slightly lower than when the safe trajectories are synthetically generated (79% vs 80%), while their SKD differs by more than 0.5 points. This controller is the only one where on average, the car only starts to stop when it is already too late (i.e., when the car can no longer avoid collision). In such a situation, error in the acceleration of the car, which dominates error in the velocity (Section 6.1.1), can substantially influence the collision rate. Since this acceleration error is not modelled in the POMDP that generates the kamikaze trajectories, this dominating factor is not considered in SKD computation. A possible remedy and somewhat typical in practical applications of POMDP is to model the system with larger uncertainty.

Table 1: ML-Controller Assessment

ML-Controller		
Safe Trajectories	**SKD ± 95 C.I**	**Collision Rate (%)**
Synthetic Data	2.31 ± 0.08	0
Real Data	5.70 ± 0.18	0

We also tested our assessment mechanism on the ML controller. For this purpose, we used the same 5 safe trajectories per type as above and 100 kamikaze trajectories per safe trajectory, and simulate them in Carla v.0.9.6. We computed SKD and collision rate based on these 500 pairs of safe and kamikaze trajectories. The results are in Table 1.

To understand the feasibility of conducting the proposed testing mechanism frequently, we need to look into the time required to compute the above results. Table 2 presents the time taken for safe and kamikaze trajectory generations, Fréchet distance computation, and the total time. These results indicate that a simulation run, which is one data point to compute SKD, can be computed in under 11s. The time taken to assess the ML controller is longer than that to assess the basic controller because of the high-fidelity simulator used and the more complex computation required by the controller. However, even the ML controller took less than 11ms. Moreover, if we compute the synthetic safe trajectories ahead of time and reuse them —a practice that will reduce the variance when computing SKD—, the time required for a single simulation run regardless of the controller is under 8s. Of course, to have statistical confidence, we need to run these simulations many times. However, this process is embarrassingly parallel.

Of course, if the tests were to be conducted directly on the physical system, more time and effort would be required to run them multiple times. In this case, we imagine combining physical and simulation results would be beneficial to reduce the number of repeated physical robot testing required. But, further work are necessary to combine the two results well.

7 Summary

In this paper, we propose a testing mechanism to assess the safety of autonomous cars, inspired by the NCAP safety rating. Core to our proposal is a similarity measure SKD, which uses the Fréchet distance between the adversary's safe trajectories and kamikaze trajectories closest to those safe trajectories. We found the average of such a distance is inversely proportional to the upper bound of the probability that a small deformation turns a safe trajectory into a dangerous one. Systematic tests on simulation and a test on a Machine-Learning controller using a high-fidelity simulator corroborate this characteristics of SKD.

The time taken for each simulation test is under 11 seconds if we include parts of the scenario generations, and under 8 seconds otherwise. Therefore, it is feasible for the proposed testing mechanism to generate sufficient statistics from simulation on a quad-core desktop in 15-30 minutes, which is equivalent to the typical time one uses for washing a car. The speed of this assessment opens the possibility for a relatively frequent safety assessment to be taken in simulation, for instance after every patch or software update is performed.

Moreover, the safe trajectory can be based on static scenarios as defined by regulatory bodies, which will likely be more acceptable by the regulators.

Future work abounds. More exhaustive testing using a variety of state-of-the-art Machine Learning controllers are needed to better understand the effectiveness of SKD as a safety measure. In the computation of the component of the testing mechanism, plenty of rooms are available to improve the kamikaze trajectory generation. In this paper, we assume a very simple car dynamics. Applying a more realistic car dynamics would require solving differential game under partial observability. Furthermore, ensuring the kamikaze trajectory generated is close to the corresponding safe trajectory requires solving constrained differential

game problems. Currently, SKD is computed based on simulation results alone. How could we combine tests on simulation test and physical robot, so as to obtain a good safety measure fast?

Last but not least, we believe planning under uncertainty techniques could help develop viable *user-focused* safety indicator and testing mechanisms for autonomous systems, and hope this work encourages further exploration in this direction.

Acknowledgments

This work is supported by the Assuring Autonomy International Programme and ANU Futures Scheme.

References

[1] Global ncap. URL `http://www.globalncap.org/`.

[2] Pankaj K Agarwal, Rinat Ben Avraham, Haim Kaplan, and Micha Sharir. Computing the discrete fréchet distance in subquadratic time. *SIAM Journal on Computing*, 43(2):429–449, 2014.

[3] Matthias Althoff and John M Dolan. Online verification of automated road vehicles using reachability analysis. *IEEE Transactions on Robotics*, 30(4):903–918, 2014.

[4] R. P. Bhattacharyya, R. Senanayake, K. Brown, and M. J. Kochenderfer. Online parameter estimation for human driver behavior prediction. In *2020 American Control Conference (ACC)*, pages 301–306, 2020.

[5] M. Breed and J. Moore. *"Animal Behavior"*. "Elsevier", 2015.

[6] Linda Capito, Bowen Weng, Umit Ozguner, and Keith Redmill. A modeled approach for online adversarial test of operational vehicle safety, 2020.

[7] Timothy H Chung, Geoffrey A Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics. *Autonomous robots*, 31(4):299–316, 2011.

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[9] Thomas Eiter and Heikki Mannila. Computing discrete frechet distance. 05 1994.

[10] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin. Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games. In *IEEE International Conference on Robotics and Automation*, pages 1475–1481, 2020.

[11] Sariel Har-Peled. *Geometric approximation algorithms*. Number 173. American Mathematical Soc., 2011.

[12] Richard Hawkins, Colin Paterson, Chiara Picardi, Yan Jia, Radu Calinescu, and Ibrahim Habli. Guidance on the assurance of machine learning in autonomous systems (amlas), 2021.

[13] Marcus Hoerger, Hanna Kurniawati, and Alberto Elfes. A software framework for planning under partial observability. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9. IEEE, 2018.

[14] S. Hoermann, D. Stumper, and K. Dietmayer. Probabilistic long-term prediction for autonomous vehicles. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 237–243, 2017.

[15] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

[16] M. Koren, S. Alsaif, R. Lee, and M. J. Kochenderfer. Adaptive stress testing for autonomous vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–7, 2018.

[17] Euro NCAP. AEB Pedestrian, 2020. URL `https://www.euroncap.com/en/vehicle-safety/the-ratings-explained/vulnerable-road-user-vru-protection/aeb-pedestrian/`.

[18] Matthew O'Kelly, Aman Sinha, Hongseok Namkoong, Russ Tedrake, and John C Duchi. Scalable end-to-end autonomous vehicle testing via rare-event simulation. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[19] Axel Sauer, Nikolay Savinov, and Andreas Geiger. Conditional affordance learning for driving in urban environments. In *Proceedings of The 2nd Conference on Robot Learning*, pages 237–252, 2018.

[20] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka. Multiple model unscented kalman filtering in dynamic bayesian networks for intention estimation and trajectory prediction. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1467–1474, 2018.

[21] Gesina Schwalbe and Martin Schels. A Survey on Methods for the Safety Assurance of Machine Learning Based Systems. In *10th European Congress on Embedded Real Time Software and Systems (ERTS 2020)*, Toulouse, France, January 2020.

[22] Sanjit A Seshia, Dorsa Sadigh, and S Shankar Sastry. Formal methods for semi-autonomous driving. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–5. IEEE, 2015.

[23] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

[24] Haowei Sun, Shuo Feng, Xintao Yan, and Henry X. Liu. Corner case generation and analysis for safety assessment of autonomous vehicles, 2021.

[25] B. Weng, S. J. Rao, E. Deosthale, S. Schnelle, and F. Barickman. Model predictive instantaneous safety metric for evaluation of automated driving systems. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1899–1906, 2020.

[26] Tichakorn Wongpiromsarn, Sertac Karaman, and Emilio Frazzoli. Synthesis of provably correct controllers for autonomous vehicles in urban environments. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1168–1173, 2011.

[27] Dongfang Yang, Linhui Li, Keith A. Redmill, and Ümit Özgüner. Top-view trajectories: A pedestrian dataset of vehicle-crowd interaction from controlled experiments and crowded campus. *CoRR*, abs/1902.00487, 2019.

Table 2: Time taken to generate safe and kamikaze trajectories, Fréchet distance, SKD, and the total time.

Car Controller	**Synthetic Data**				**Real Data**		
	SafeTrajGen Avg 95 C.I (ms)	**KamikazeGen Avg 95 C.I (ms)**	**FretchetTime Avg 95 C.I (ms)**	**Total SKD Avg 95 C.I (ms)**	**KamikazeGen Avg 95 C.I (ms)**	**FretchetTime Avg 95 C.I (ms)**	**Total SKD Avg 95 C.I (ms)**
0.500	5535.41 ± 3.10	2437.99 ± 3.10	0.60 ± 0.01	7974 ± 127.19	1950.79 ± 21.98	0.398 ± 0.008	1951.19 ± 21.98
0.625	5535.41 ± 3.10	2452.62 ± 3.00	0.62 ± 0.01	7988.65 ± 127.19	1975.86 ± 22.17	0.404 ± 0.008	1976.26 ± 22.17
0.750	5535.41 ± 3.11	2471.53 ± 0.60	0.65 ± 0.01	8007.59 ± 127.19	2133.56 ± 17.48	0.455 ± 0.011	2134.02 ± 17.48
0.875	5535.41 ± 3.11	2741.11 ± 10.46	0.72 ± 0.02	8277.24 ± 127.59	2173.64 ± 18.15	0.414 ± 0.008	2174.05 ± 18.15
1.000	5535.41 ± 3.12	3023.28 ± 29.95	0.76 ± 0.02	8559.45 ± 130.64	2297.44 ± 21.20	0.455 ± 0.010	2297.89 ± 21.2
1.050	5535.41 ± 3.12	4172.54 ± 145.18	0.78 ± 0.02	9708.72 ± 192.99	2364.42 ± 25.75	0.465 ±0.011	2364.89 ± 25.75
1.100	5535.41 ± 3.13	4038.40 ± 124.23	0.77 ± 0.02	9574.58 ± 177.77	2473.88 ± 42.84	0.490 ± 0.013	2474.37 ± 42.84
1.125	5535.41 ± 3.13	3491.84 ± 101.59	0.82 ± 0.02	9028.07 ± 162.76	2380.11 ± 20.21	0.498 ± 0.013	2380.61 ± 20.21
1.150	5535.41 ± 3.14	3511.95 ± 43.86	0.80 ± 0.02	9048.16 ± 134.51	2482.93 ± 18.69	0.487 ± 0.012	2483.41 ± 18.69
ML	5535.41 ± 3.14	5391.41 ± 100.36	0.83 ± 0.02	10927.65 ± 161.99	7985.79 ± 165.88	1.051 ± 0.029	7986.85 ± 165.88

ASSURING
AUTONOMY
INTERNATIONAL PROGRAMME