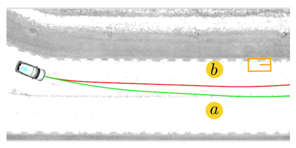
Liability, Ethics, and Culture-Aware Behavior Specification using Rulebooks

Andrea Censi, Konstantin Slutsky, Tichakorn Wongpiromsarn,
Dmitry Yershov, Scott Pendleton, James Fu, Emilio Frazzoli

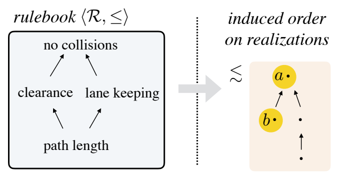
***Abstract*— The behavior of self-driving cars must be compatible with an enormous set of conflicting and ambiguous objectives, from law, from ethics, from the local culture, and so on. This paper describes a new way to conveniently define the desired behavior for autonomous agents, which we use on the self-driving cars developed at nuTonomy.**

We define a "rulebook" as a pre-ordered set of "rules", each akin to a violation metric on the possible outcomes ("realizations"). The rules are partially ordered by priority. The semantics of a rulebook imposes a pre-order on the set of realizations. We study the compositional properties of the rulebooks, and we derive which operations we can allow on the rulebooks to preserve previously-introduced constraints.

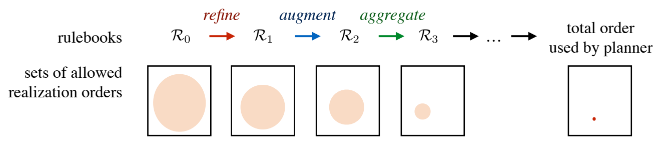
While we demonstrate the application of these techniques in the self-driving domain, the methods are domain-independent.



(a) Autonomy is about making the right choices.


rulebook $\langle \mathcal{R}, \leq \rangle$
no collisions
clearance
lane keeping
path length
induced order on realizations
$\lesssim$
a
b

(b) Rulebook and induced order on realizations (outcomes).



(c) Rulebook manipulation operations refine the specification

Fig. 1: The rulebooks formalism allows to specify the desired behavior for an autonomous agent by using a pre-ordered set of rules that induce a pre-order on the allowed outcomes. The rulebooks can be refined by a series of manipulation operations.

I. Introduction

One of the challenges in developing self-driving cars is simply *defining what the car is supposed to do*. The behavior specification for a self-driving car comes from numerous sources, including not only the (vaguely specified) "rules of the road", but also implementation limitations (e.g. the speed might be limited due to the available computation for perception), and numerous other soft constraints, such as the need of "appearing natural", or to be compatible with the local driving culture (e.g., the "Massachusetts left"). As self-driving cars are potentially life-endangering, also moral and ethical factors play a role [3], [12], [17]. For a self-driving car, the "trolley problems" [1], [6], [16] are not idle philosophical speculations, but something to solve in a split second. As of now, there does not exist a formalism that allows to incorporate all these factors in one specification, which can be precise enough to be taken as regulation for what self-driving cars designers must implement.

In this paper we describe a formalism called "rulebooks", which we use to specify the desired behavior of the self-driving cars developed at nuTonomy. While the formalism can be applied to any system, it is particularly well-suited to handle behavior specification for embodied agents in an environment where many, possibly conflicting rules need to be honored. We define a "rulebook" as a set of "rules" (Fig. 1b), each akin to a violation metric on the possible outcomes. The rules can be defined analytically, using formalisms such as LTL [8] or even deontic logic [20], or the violation functions can be learned from data, using inverse reinforcement learning [9], or any technique that allows to measure deviation from a model. In the driving domain, the rules can derive from traffic laws, from common sense, from ethical considerations, etc.

The rules in a rulebook are hierarchically ordered to describe their relative priority, but, following the maxim "*good specifications specify little*", the semantics of a rulebook imposes a *pre-order* on the set of outcomes, which means that the implementations are left with considerable freedom of action. The rulebooks formalism is "user-oriented": we define a set of intuitive operations that can be used to iteratively refine the behavior specification. For example, one might define an "international rulebook" for the rules that are valid everywhere, and then have derived region-specific rulebooks that describe rules such as which side the car should drive on.

While the rulebooks offer formidable generality in describing behavior, at the same time, when coupled with graph-based motion planning, the rulebooks allow a *systematic, simple, and scalable solution to planning* for a self-driving car:

1) Liability-, ethics-, culture-derived constraints are formulated as rules (preferences over trajectories), either manually or in a data-driven fashion, together with the rules of the road and the usual geometric constraints for motion planning.
2) Priorities between conflicting rules are established as a rulebook (ideally, by nation-wide regulations based on public discourse);
3) Developers customize the behavior by resolving ambiguities in the rulebook until a total order is obtained;

The authors are with nuTonomy, an Aptiv company (Boston, MA, Zurich, and Singapore). Please address correspondence to andrea@nutonomy.com.

4) Graph-based motion planning, in particular variations of minimum-violation motion planning [5], [7], [18], [19], [21], allow to generate the trajectories that maximally respect the rules in the rulebooks.

In a nutshell, the above is how the nuTonomy cars work. (The topic of *efficiently* planning with rulebooks is beyond the goals of this paper; here, we focus on the use of the rulebooks as a specification, treating the planning process as a black box.)

II. Rulebooks definition

1) Realizations: Our goal is to define the desired agent *behavior*. Here, we use the word "behavior" in the sense of Willems [13] (and not in the sense of "behavior-based robotics" [2], [10]), to mean that what we want to prescribe is what we can measure objectively, that is, the externally observable actions and outcomes in the world, rather than the internal states of the agent or any implementation details.

Our goal is to describe the preference relations on a set of possible outcomes, which we call the set of *realizations* Ξ. For example, for a self-driving car, a realization $x \in \Xi$ is a world trajectory, which includes the trajectory of all agents in the environment.

We use *no* concept of infeasibility. Sometimes the possible outcomes are all catastrophically bad; yet, an embodied agent must keep calm and choose the least catastrophic option.

2) Rules: Our "atom" of behavioral specification is the "rule". In our approach, a rule is simply a scoring function, or "violation metric", on the realizations.

Definition 1 (Rule)**.** Given a set of realizations Ξ, a *rule* on Ξ is a function $r : \Xi \to \mathbb{R}_+$.

The function r measures the degree of violation of its argument. If $r(x) < r(y)$, then the realization y violates the rule r to a greater extent than does x. In particular, $r(x) = 0$ indicates that a realization x is fully compliant with the rule.

Any scalar function will do. The definition of the violation metric might be analytical, "from first principles", or be the result of a learning process.

In general, the rulebooks philosophy is to pay particular attention about specifying what we ought to do *when the rule has to be violated*, as described in the following examples.

Example 2 (Speed limit)**.** A naïve rule that is meant to capture a speed limit of 45 km/h could be defined as:

$$r(x) = \begin{cases} 0, & \text{if the car's speed is always below 45 km/h,} \\ 1, & \text{otherwise.} \end{cases}$$

However, this discrete penalty function is not very useful in practice. The rulebooks philosophy is to assume that rules might need to be violated for a greater cause. In this case, it is advisable to define a penalty such as:

$$r(x) = \text{interval for which the car was above 45 km/h.}$$

The effect of this will be that the car will try to stay below the speed limit, but if it cannot, it will minimize the time spent violating the limit. Alternatively, one can penalize also the magnitude of the speed violation:

$$r'(x) = r(x) \times (v_{\max} - 45 \text{ km/h}).$$

Example 3 (Minimizing harm)**.** It is easy enough to write a constraint describing the fact that we do not want any collision; but, assuming that a collision with a human is unavoidable given the circumstances, what should the car do? In this case, it would be advisable to define the violation function as:

$$r(x) = \text{kinetic energy transferred to human bodies,}$$

so that the car will try to avoid collisions, but, if a collision is inevitable, it will try to reduce the speed as much as possible.

3) Rulebooks: A rulebook $\mathcal{R}$ is a *pre-ordered* set of rules. We will use $\mathcal{R}$ both for the rulebook and for its underlying set of rules.

Definition 4 (Rulebook)**.** A *rulebook* is a tuple $\langle \mathcal{R}, \leq \rangle$, where $\mathcal{R}$ is a finite set of rules and $\leq$ is a preorder on $\mathcal{R}$.

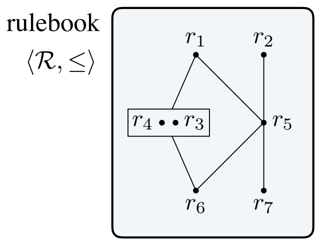


Fig. 2: Graphical representation of a rulebook. Rules are ordered vertically with the most important rules being at the top.

Being a preorder, any rulebook may be represented as a directed graph, in which each node is a rule, and an edge between two rules $r_1 \to r_2$ means that $r_1 \leq r_2$, i.e., the rule r_2 has higher rank. Figure 2 gives an example of a rulebook with 7 rules. In this example, rules r_1 and r_2 are incomparable, but both are greater than r_5. Rules r_3 and r_4 are of the same rank, meaning $r_3 \leq r_4$ *and* $r_4 \leq r_3$, and both are smaller than r_1, greater than r_6 and incomparable to r_5, r_2, or r_7.

Just like it might be convenient to learn some of the non-safety-critical rules from data, it is possible to learn some of the priorities from data as well. (See [11], [15] for a similar concept in a different context.)

4) Induced pre-order on realizations: We now formally define the semantics of a rulebook as specifying a pre-order on realizations. Because a rulebook is defined as a *pre-ordered* set of rules, not all the relative priorities among different rules are specified. We will see that this means that a rulebook can be used as a very flexible *partial* specification.

Given a rulebook $\langle \mathcal{R}, \leq \rangle$, our intention is to preorder all realizations such that $x \lesssim y$ can be interpreted as x being "at least as good as" y, i.e., the degree of violation of the rules by x is at most as much as that of y.

Definition 5 (Pre-order $\lesssim$ and strict version $<$)**.** Given a rulebook $\langle \mathcal{R}, \leq \rangle$ and two realizations $x, y \in \Xi$, we say that

$x \lesssim y$ if for any rule $r \in \mathcal{R}$ satisfying $r(y) < r(x)$ there exists a rule $r' > r$ such that $r'(x) < r'(y)$. We denote by $<$ the strict version of $\lesssim$.

Lemma 6. *Let $\langle \mathcal{R}, \leq \rangle$ be a rulebook, let $x, y, z \in \Xi$ be realizations such that $x \lesssim y$, $y \lesssim z$, and let $r \in \mathcal{R}$ be a rule. If either $r(x) \neq r(y)$ or $r(y) \neq r(z)$, then there exists a rule $r' \geq r$ such that $r'(x) < r'(z)$.*

Proof. We give a proof for $r(x) \neq r(y)$; the case $r(y) \neq r(z)$ is analogous. If $r(x) > r(y)$, then $x \lesssim y$ guarantees existence of $r_0 > r$ such that $r_0(x) < r_0(y)$. If $r(x) < r(y)$ to begin with, then we may set $r_0 = r$, and in either case we get $r_0 \geq r$ such that $r_0(x) < r_0(y)$. We are done if $r_0(y) \leq r_0(z)$, as one can take $r' = r_0$. Otherwise, $y \lesssim z$ implies existence of $r_1 > r_0$ such that $r_1(y) < r_1(z)$. Again, we are done if $r_1(x) \leq r_1(y)$, and if not, there has to be some rule $r_2 > r_1$ such that $r_2(x) < r_2(y)$. Continuing in the same fashion, one builds an increasing chain $r \leq r_0 < r_1 < r_2 < \cdots$. Since $\mathcal{R}$ is assumed to be finite, the chain has to stop, which is possible only if $r_n(x) < r_n(z)$ for some n.

Proposition 7. *Let $\langle \mathcal{R}, \leq \rangle$ be a rulebook, and let $x, y, z \in \Xi$ be realizations.*

1) *The relation $\lesssim$ on realizations is a preorder.*
2) *Two realizations x and y are equivalent if and only if $r(x) = r(y)$ for all rules $r \in \mathcal{R}$.*

Proof. 1) It is clear that $\lesssim$ is reflexive, so we only need to check transitivity. Suppose $x \lesssim y$, $y \lesssim z$, and let $r \in \mathcal{R}$ be such that $r(x) > r(z)$. Clearly either $r(y) \neq r(x)$ or $r(z) \neq r(y)$, so Lemma 6 applies, producing $r' > r$ such that $r'(x) < r'(z)$, hence $x \lesssim z$ as claimed.

2) Suppose towards a contradiction there are some realizations satisfying $x \lesssim y$ and $y \lesssim x$, yet $r(x) \neq r(y)$ for some $r \in \mathcal{R}$. Without loss of generality, let us assume that $r(x) < r(y)$. Since we have $x \lesssim y \lesssim x$, Lemma 6 produces some $r' \geq r$ such that $r'(x) < r'(x)$, which is absurd. The other direction $(\forall r, r(x) = r(y) \implies x \sim y)$ is obvious.

Remark 8. In the special case in which the rulebook is a linear order, the induced order on realizations is the *lexicographic order* used in the literature in minimum-violation planning.

III. Examples in the Driving Domain

In this section, we give a few examples of the types of rules that are useful in the driving domain. Rather than describing the full complexity of our production rules, which address subtle nuances of behavior and idiosyncrasies and corner cases of traffic laws, we prefer to give a few synthetic examples of rulebooks and rulebooks refinement.

Example 9 (Safety vs. infractions)**.** Consider the scenario in Figure 3. A vehicle is faced with an obstacle in front, and is given a choice between two trajectories a and b. Suppose the initial speed of the vehicle is sufficiently high, and there is no time to stop, so collision is unavoidable if a is chosen. Trajectory b, however, is collision free, but it violates a different rule, since it intersects a double solid line.

The rulebooks take on this situation is the following. A rule "not to collide with other objects" will have a higher priority than the rule of not crossing the double line (Fig. 3b). With this rulebook, the trajectory b will be chosen to avoid the collision.

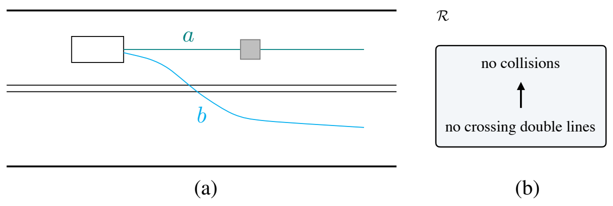


Fig. 3: The rulebook allows the agent to cross the double white line to avoid a collision. (This assumes that there are no other agents outside the frame that might trigger the "no-collision" rule.)

Example 10 (Liability-aware specification)**.** Let's change the situation slightly by assuming that trajectory b is also in collision, but with a different agent — an oncoming vehicle on the opposite lane. Under these assumptions, we may be interested in choosing the outcome where the ego vehicle is *not* at fault for the collision.

This behavior specification can be achieved by the rulebook of Fig. 4b, having two collision rules, where one evaluates the degree of collision, where the ego vehicle is at fault, and the other evaluates collisions caused by third-party, which is below the former in the rulebooks hierarchy. This will force the ego vehicle to prefer trajectory a over b.

This example fully captures the concept of the "responsibility-sensitive safety" model described in [14].

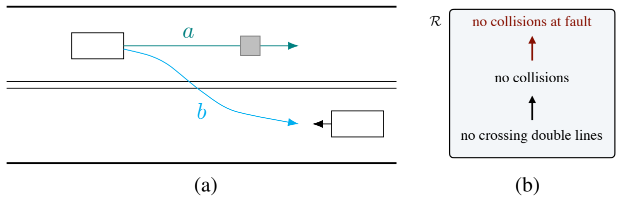


Fig. 4: The rulebook instructs the agent to collide with the object on its lane, rather than provoking an accident, for which it would be at fault.

Example 11 (Partial priorities specification)**.** Consider the scenario depicted in Fig. 5a, where the vehicle encounters an obstacle along its route. For simplicity, we focus on four discrete representative trajectories, called a, b, c, d. A minimal rulebook that allows to deal with this situation would contain at least four rules, detailed below. For simplicity, we write the violation metrics as binary variables having value 0 or 1 on the test trajectories, while in practice these would be continuous functions.

1) Rule β - **Blockage**, attaining value 1 if the trajectory is blocked by an obstacle, and 0 otherwise:

$$\beta(x) = \begin{cases} 0, & \text{for } x = b, c, d; \\ 1, & \text{for } x = a. \end{cases}$$

2) Rule λ - **Lane Keeping**, 1 iff the trajectory intersects the lane boundary:

$$\lambda(x) = \begin{cases} 0, & \text{for } x = a, b; \\ 1, & \text{for } x = c, d. \end{cases}$$

3) Rule κ - **Obstacle clearance**, 1 iff the trajectory comes closer to an obstacle than some threshold C_0:

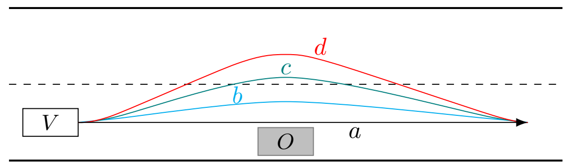
$$\kappa(x) = \begin{cases} 0, & \text{for } x = c, d; \\ 1, & \text{for } x = a, b. \end{cases}$$

Remark 12 (Learning while preserving safety). While parameters such as the minimum clearance from an obstacle C_0 can be specified manually, in practice, given an adequate data analytics infrastructure, they are great candidates to be *learned* from the data. This allows the car to adapt the behavior to the local driving culture. By still having the safety-preserving rules at the top of the hierarchy, the rulebooks allow the system to be adaptive without ever compromising safety, not even with adversarial data. (See [4] for a similar principle in a different context.)

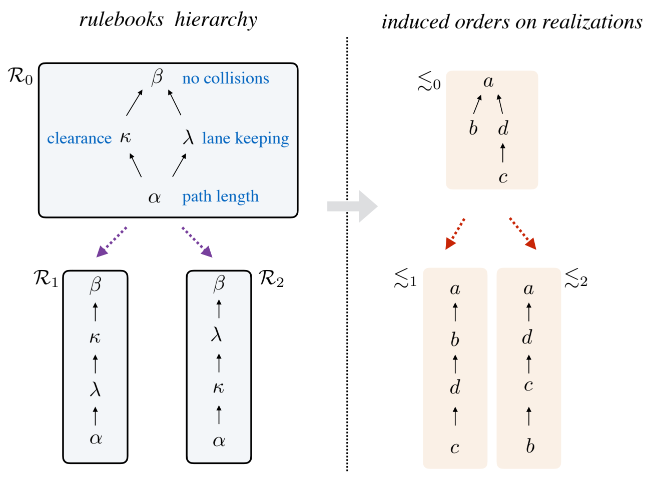
4) Rule α - **Path length**, whose value is the length of the trajectory:

$$\alpha(a) < \alpha(b) < \alpha(c) < \alpha(d).$$

Out of these rules, we can make different rulebooks by choosing different priorities. For example, defining the rulebook $\mathcal{R}$ with ordering $\alpha < \kappa < \beta$ and $\alpha < \lambda < \beta$, depicted in Fig. 5b, the following order on trajectories is imposed: $b < a$ and $c < d < a$, but note that b is not comparable with either d or c. This is an important feature of a *partial specification:* we leave freedom to the implementation to choose the details of the behavior that we do not care about.



(a) Trajectories available to a vehicle before an avoidance maneuver.



(b) Rulebook hierarchy and induced hierarchy on realizations order.

Fig. 5

3) **Rule augmentation** (Def. 17): this operation consists in adding another rule at the lowest level of priority.

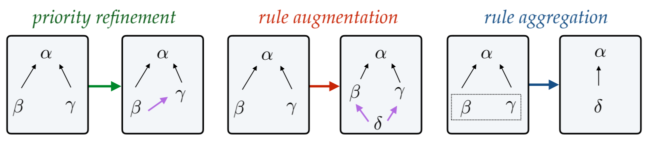


Fig. 6: Three operations for manipulation of rulebooks.

IV. Iterative Specification Refinement with Rulebooks Manipulation

We formalize this process of *iterative specification refinement* (Fig. 1c), by which a user can add rules and priority relations until the behavior of the system is fully specified to one's desire.

Example 13. Regulations in different states and countries often share a great deal of similarity. It would be ineffective to start the construction of rulebooks from scratch in each case; rather, we wish to be able to define a "base" rulebook that can then be particularized for a specific legislation by adding rules or priority relations.

1) Operations that refine rulebooooks: We will consider three operations (Fig. 6):

1) **Priority refinement** (Def. 14): this operation corresponds to adding another edge to the graph, thus clarifying the priority relations between two rules.
2) **Rule aggregation** (Def. 16): this operation allows to “collapse” two or more equi-ranked rules into one.

2) Priority refinement: The operation of refinement adds priority constraints to the rulebook.

Definition 14. An allowed *priority refinement* operation of a rulebook $\langle \mathcal{R}_1, \leq_1 \rangle$ is a rulebook $\langle \mathcal{R}_1, \leq_2 \rangle$, where the order $\leq_2$ is a refinement of $\leq_1$.

Example 15. Continuing the example in Fig. 5, we can create two refinements of the rulebook by adding priority constraints that resolve the incomparability of rules κ and λ one way or the other. For example, choosing the totally ordered rulebook $\alpha \to \kappa \to \lambda \to \beta$, the order on trajectories is $b < c < d < a$, while for the rulebook $\alpha \to \lambda \to \kappa \to \beta$, the order is $c < d < b < a$.

3) Rule aggregation: Suppose that a rulebook includes two rules that are in the same equivalence class. The minimal example is a rulebook $\langle \mathcal{R}, \leq \rangle$ that has two rules r_1, r_2 such that $r_1 \leq r_2$ and $r_2 \leq r_1$. The induced order $\lesssim$ on the realizations is that of the product order:

$$x \lesssim y \quad \text{iff} \quad r_1(x) \leq r_1(y) \ \wedge \ r_2(x) \leq r_2(y).$$

We might ask whether we can "aggregate" the two rules into one. The answer is positive, given the conditions in the following definition.

Definition 16 (Rule aggregation operation)**.** Consider a rulebook $\langle \mathcal{R}, \leq \rangle$ in which there are two rules $r_1, r_2 \in \mathcal{R}$ that are in the same equivalence class defined by $\leq$. Then it is allowed to "aggregate" the two rules into a new rule r', defined by

$$r'(x) = \alpha(r_1(x), r_2(x)),$$

where α is an embedding of the product pre-order into $\mathbb{R}_+$.

In particular, allowed choices for α include linear combinations with positive coefficients ($\alpha(r_1, r_2) = a\, r_1 + b\, r_2$) and other functions that are strictly monotone in both arguments.

4) Rule augmentation: Adding a rule to a rulebook is a potentially destructive operation. In general, we can preserve the existing order only if the added rule is below every other.

Definition 17 (Rule augmentation)**.** The operation of rule augmentation consists in adding to the rulebook $\mathcal{R}$ a rule r' such that $r' < r$ for all $r \in \mathcal{R}$.

5) Properties preserved by the three operations: We will show that the three operations create a rulebook that is a refinement of the original rulebook, in the sense of Def. 18.

Definition 18. A rulebook $\langle \mathcal{R}_1, \leq_1 \rangle$ a *strict refinement* of $\langle \mathcal{R}_2, \leq_2 \rangle$ if its induced strict pre-order $<_2$ refines $<_1$.

One can prove this theorem:

Theorem 19. *Applying one of the three operations (augmentation, refinement, aggregation) to a rulebook $\mathcal{R}_1$ creates a rulebook $\mathcal{R}_2$ that is a strict refinement of $\mathcal{R}_1$ in the sense of Def. 18.*

Detailed proofs of this and ancillary results are available in the extended version of this report, available at website `http://rulebooks.tech`, together with high-resolution videos of the experiments.

V. Experiments

We show planning results for different rulebooks for the nuTonomy R&D platform (Renault Zoe). The experiments assume left-hand traffic (Singapore/UK regulations).

A. Unavoidable collision

This experiment illustrates unavoidable collision as described in Example 10. We set up the scenario (Fig. 7) such that the planner is led to believe that 2 vehicles instantaneously appear at approximately 12 m from the ego vehicle and slowly move towards the ego vehicle at 1.0 m/s, while the speed of the ego vehicle is 9.5 m/s. Fig. 7 shows the belief state when the vehicles first appear. We also limit the allowed deceleration to 3.5 m/s^2. It can be verified that collision is unavoidable under these conditions.

For any given trajectory x, we define the collision cost as

$$\mu(x) = v_{x,\text{col}}, \tag{1}$$

where $v_{x,\text{col}}$ is the expected longitudinal speed of the ego vehicle at collision, assuming that the ego vehicle applies the maximum deceleration from the current state.

First, consider the case where the collision cost (1) is applied to any collision. In this case, it is more preferable to swerve and hit the traffic vehicle in the opposite lane since the swerving trajectory gives the ego vehicle more distance to decelerate; hence, reducing the expected speed at collision.

Next, collision at fault μ_1 is differentiated from collision caused by third-party μ_2 with priority $\mu_2 < \mu_1$. The optimal trajectory in this case is to stay within lane and collide with the traffic vehicle that is moving against the direction of traffic.

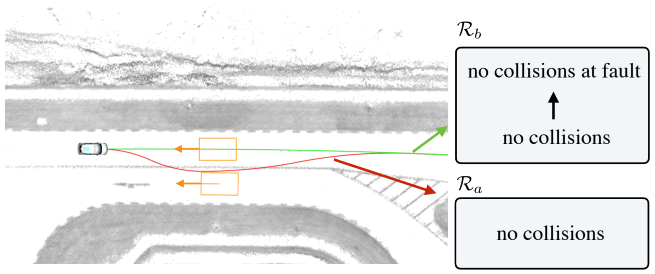


Fig. 7: Trajectories planned in the unavoidable collision scenario with different versions of the rulebooks. (See attached videos for experiment.) The orange rectangles are the traffic vehicles, moving towards the ego vehicle at the speed of 1.0 m/s. The red trajectory is chosen when collision at fault and collision caused by third-party are treated equally whereas the green trajectory is chosen when collision at fault is higher in the rulebooks hierarchy than the collision caused by third-party.

1) Clearance and lane keeping: In this experiment, we demonstrate how different rulebooks in Example 11 lead to different behaviors when overtaking a stationary vehicle. The blockage cost β, lane keeping cost λ and length α are defined as in Example 11, but we re-define the clearance cost as

$$\kappa(x) = \max(0, C_0 - l_x), \tag{2}$$

where l_x is the minimum lateral distance between the stationary vehicle and trajectory x.

In particular, we consider two different rulebooks (Fig. 5):

$$\mathcal{R}_1 = \{\alpha < \lambda < \kappa < \beta\}, \quad \text{(clearance first)} \tag{3}$$
$$\mathcal{R}_2 = \{\alpha < \kappa < \lambda < \beta\}. \quad \text{(lane keeping first)} \tag{4}$$

The rulebook described in (4) corresponds to the case where satisfying the lane keeping rule is preferred over satisfying the clearance rule whereas the rulebook described in (3) corresponds to the case where satisfying the clearance rule is preferred over satisfying the lane keeping rule.

Fig. 8 shows the optimal paths found by the system in the two cases. With rulebook $\mathcal{R}_2$, the optimal trajectory is such that the vehicle footprint remains within lane, leading to the violation of the clearance rule. In contrast, when rulebook $\mathcal{R}_1$ is applied, the trajectory crosses the lane boundary to give sufficient clearance from the stationary vehicle.

2) Lane change near intersection: Consider the scenario where the autonomous vehicle needs to perform a lane change in the vicinity of an intersection as shown in Fig. 9. Here,

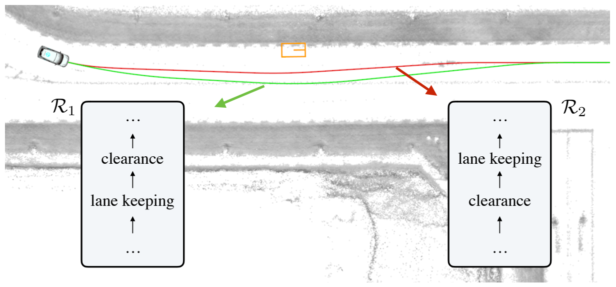


Fig. 8: Trajectories planned in the vehicle overtaking scenario with different rulebooks. (See attached videos for experiment.) The orange rectangle is the stationary vehicle. The red trajectory is when the rulebook (4) is used, whereas the green trajectory is when the rulebook (3) is used.

the autonomous vehicle needs to turn left at the intersection. It is therefore required to be on the left lane before entering the intersection. However, there is a stationary vehicle that prevents it from completing the maneuver at an appropriate distance from the intersection.

For the simplicity of the presentation, we assume that any trajectory x only crosses the lane boundary once at η_x. The lane change near intersection cost is then defined as $\zeta(x) = \max(0, D_{lc} - d_{\text{int}}(\eta_x))$, where D_{lc} is a predefined threshold of the distance from intersection, beyond which changing lane is not penalized and for any pose p, $d_{\text{int}}(p)$ is the distance from p to the closest intersection.

Additionally, we define the turning cost $\tau(x)$ as the L_1-norm of the heading difference between x and the nominal trajectory associated with each lane. Consider the case where ζ and τ are in the same equivalence class and these rules are aggregated (16) as

$$r_{\zeta,\tau}(x) = \zeta(x) + c_\tau \tau(x), \tag{5}$$

where $c_\tau > 0$ is a predefined constant. In this experiment, we consider the aggregated cost $r_{\zeta,\tau}$ and the blockage cost β defined in Example 11 with priority $r_{\zeta,\tau} < \beta$. Fig. 9 shows how the choice of c_τ affects the optimal trajectory.

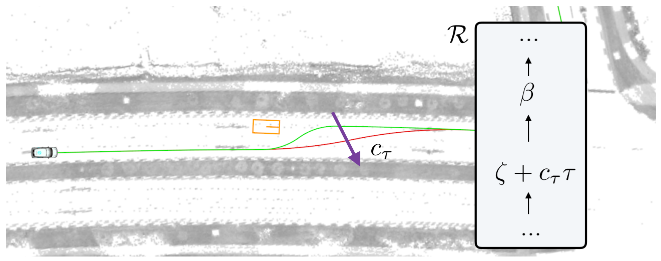


Fig. 9: Trajectories planned in the lane changing near intersection scenario. (See attached videos for experiment.) The orange rectangle is the stationary vehicle at pose p_v with $d_{int}(p_v) < D_{lc}$. The green trajectory is the optimal trajectory for $c_\tau = 0$ whereas the red trajectory is the optimal trajectory for some $c_\tau > 0$.

VI. Discussion and Future Work

We have shown by way of a few particular examples how the rulebooks formalism allows easy tuning of self-driving behavior in an intuitive way.

What is difficult to convey in a short paper is the ability of the formalism to scale up. In our production code at nuTonomy, corresponding to level 4 autonomy in a relatively limited operating domain, our rulebooks have about 15 rules. For complete behavior coverage for US-Massachusetts or Singapore, including rare corner cases (e.g. "do not scare farm animals"), we estimate about 200 rules, to be organized in about a dozen ordered priority groups (Fig. 10).

At the top of the hierarchy, we have the rules that guarantee safety of humans; at the bottom, we have comfort constraints and progress goals. At the top, the rules tend to be written analytically (e.g. a precise expression for the kinetic energy to minimize harm to people); at the bottom, rules are learned based on observed behavior, and also tend to be platform- and implementation- specific.

Except the extrema of safety at the top, and comfort and service level last, all the other priorities among rule groups are somehow open for discussion. What we realized is that some of the rules and rules priorities, especially those that concern safety and liability, must *be part of nation-wide regulations* to be developed after an *informed public discourse*; it should not be up to engineers to choose these important aspects. The rulebooks formalism allows to have one such shared, high-level specification that gives minimal constraints to the behavior; then, the rest of the rules and priority choices can be considered "implementation details" that might change from manufacturer to manufacturer.

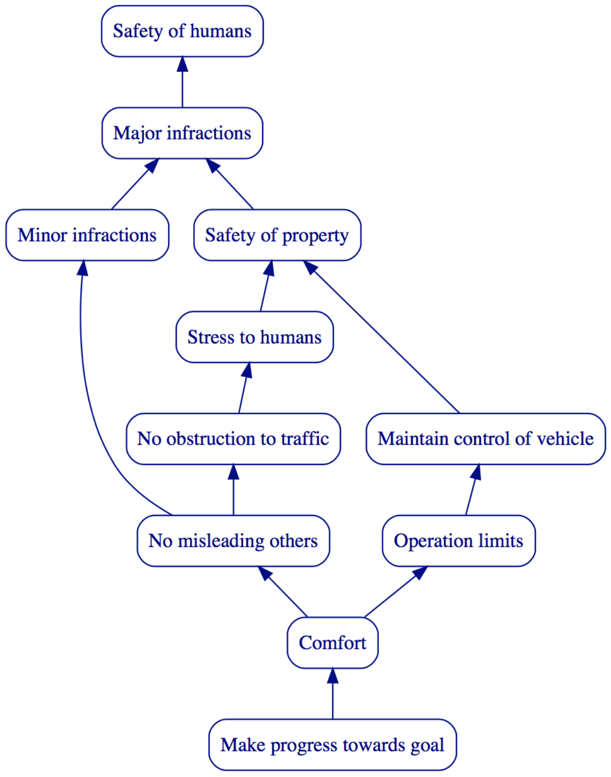


Fig. 10: Rule groups for a level-4 autonomous car, each containing other rules hierarchically. Except the extrema of safety at the top, and comfort and service level last, all the other priorities among rule groups are somehow open for discussion.

References

[1] Moral machine (online). 2016. http://moralmachine.mit.edu.

[2] Ronald C. Arkin. *Behavior-based Robotics*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[3] Ronald C Arkin. Governing Lethal Behavior: Embedding Ethics in a Hybrid Deliberative/Reactive Robot Architecture Part I: Motivation And Philosophy. *Proceedings of the 3rd international conference on Human robot interaction - HRI '08*, page 121, jan 2008.

[4] Anil Aswani, Humberto Gonzalez, S. Shankar Sastry, and Claire Tomlin. Provably safe and robust learning-based model predictive control. *Automatica*, 49(5):1216–1226, jan 2013.

[5] L. I. Reyes Castro, P. Chaudhari, J. TÅ¯movÃ¡, S. Karaman, E. Frazzoli, and D. Rus. Incremental sampling-based algorithm for minimum-violation motion planning. In *52nd IEEE Conference on Decision and Control*, pages 3217–3224, Dec 2013.

[6] Philippa Foot. The problem of abortion and the doctrine of double effect. *Oxford Review*, 5:5–15, 1967.

[7] Emilio Frazzoli and Karl Iagnemma. US patent US9645577B1: Facilitating vehicle driving and self-driving.

[8] Hadas Kress-Gazit, Morteza Lahijanian, and Vasumathi Raman. Synthesis for robots: Guarantees and feedback for robot behavior. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):211–236, 2018.

[9] Sergey Levine and Vladlen Koltun. Continuous inverse optimal control with locally optimal examples. In *ICML '12: Proceedings of the 29th International Conference on Machine Learning*, 2012.

[10] Maja J. Mataric and François Michaud. Behavior-based systems. In *Springer Handbook of Robotics*, pages 891–909. 2008.

[11] Valerio Modugno, Gerard Neumann, Elmar Rueckert, Giuseppe Oriolo, Jan Peters, and Serena Ivaldi. Learning soft task priorities for control of redundant robots. In *IEEE International Conference on Robotics and Automation (ICRA 2016)*, Stockholm, Sweden, May 2016.

[12] E. Pires Bjørgen, S. Øvervatn Madsen, T. Skaar Bjørknes, F. Vonheim Heimsæter, R. Håvik, M. Linderud, P.N. Longberg, L.A. Dennis, and M. Slavkovik. Cake, death, and trolleys: dilemmas as benchmarks of ethical decision-making. In *AAAI/ACM Conference on Artificial Intelligence, Ethics and Society*, New Orleans, USA, 2018. forthcoming.

[13] Jan Willem Polderman and Jan C. Willems. *Introduction to Mathematical Systems Theory: A Behavioral Approach*. Springer-Verlag, Berlin, Heidelberg, 1998.

[14] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a formal model of safe and scalable self-driving cars. 08 2017.

[15] João Silvério, Sylvain Calinon, Leonel Dario Rozo, and Darwin G. Caldwell. Learning competing constraints and task priorities from demonstrations of bimanual skills. *CoRR*, abs/1707.06791, 2017.

[16] Judith Jarvis Thomson. The trolley problem. 94(6):1395–1415, 1985.

[17] S. M. Thornton, S. Pan, S. M. Erlien, and J. C. Gerdes. Incorporating ethical considerations into automated vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 18(6):1429–1439, June 2017.

[18] Jana Tumova, Luis I. Reyes Castro, Sertac Karaman, Emilio Frazzoli, and Daniela Rus. Minimum-violation LTL Planning with Conflicting Specifications. *American Control Conference*, jan 2013.

[19] Jana Tumova, Gavin C. Hall, Sertac Karaman, Emilio Frazzoli, and Daniela Rus. Least-violating control strategy synthesis with safety rules. In *Proceedings of the 16th International Conference on Hybrid Systems: Computation and Control*, HSCC '13, pages 1–10, New York, NY, USA, 2013. ACM.

[20] Jeroen Van Den Hoven and Gert-Jan Lokhorst. Deontic logic and computer-supported computer ethics. *Metaphilosophy*, 33(3):376–386.

[21] Cristian Ioan Vasile, Jana Tumova, Sertac Karaman, Calin Belta, and Daniela Rus. Minimum-violation scLTL motion planning for mobility-on-demand. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1481–1488, 2017.