Deconfliction of Motion Paths with Traffic Inspired Rules in Robot–Robot and Human–Robot Interactions

Federico Celi[1], Li Wang[2], Lucia Pallottino[1] and Magnus Egerstedt[2]

Abstract— **In this paper we investigate how to resolve conflicting motions for mixed robot–robot and human–robot multiagent systems. This work is motivated by atypical driving conditions, such as parking lots, where driving rules are not as strictly enforced as on standard roads. As a result, navigation algorithms should take into account the human drivers' behaviors, especially if they prove to be in conflict with the common rules of the road. In this work we make use of safety barrier certificates with a direction bias to deconflict agents' behaviour in a near-to-collision scenario, in compliance with local traffic rules. We also propose a tool to identify the driving direction bias—both for human and autonomous agents.**

I. Introduction

Autonomous and guidance navigation algorithms for autonomous vehicles, such as self-driving cars, are designed in compliance with the "traffic rules" that apply to the particular scenario they are developed for. In fact, traffic rules are not the same worldwide, with the most notable example in the existing difference in left- and right-hand driving. These dissimilarities in driving rules are reflected in human behavioral studies that suggest that it is unclear whether humans have a natural predisposition for one side of the road or the other [1], [2]. Despite differences in the standards, and possibly even personal preferences, certainly roads are governed by a strict set of rules. Nonetheless, there exist environments where rules are not as firm and they may even be broken in order to improve the overall driving experience. As is often seen in parking lots, it is largely acceptable for a driver to overtake on the *wrong* side, or to resolve a head-on scenario by steering to the left, or to the right, regardless of what the local rules would require.

Self-driving vehicles have been incrementally deployed in controlled environments, such as autonomous airport shuttles and trains. Recently, self-driving technology has been applied to more complex scenarios, with self-driving cars operating on regular roads, corresponding to a SAE (Society of Automotive Engineers) Level 3, or even 4, of Autonomy [3]. As increasingly complex environments are being populated by autonomous vehicles, the roads must be shared, at an increasing rate, by a mixture of autonomous vehicles (AV) and human–driven vehicles (HV).

In this paper, we consider this issue of mixed autonomous and human-driven vehicles explicitly in environments where the rules of the road are less strictly enforced. What this means is that the autonomous vehicle must be able to infer something about the human intent in order to operate effectively, safely, and in a manner that is transparent to the human. In the context of navigation of multiple agents, the problem of coordination and intention awareness ([4]) has been studied in the past, refer e.g., to the surveys [5], [6] and references there in. The question of multiagent systems with AV-HV interaction including safety and traffic rules notions has been considered in the context of hybrid systems theory [7], fuzzy logic-based decision making [8], and, more recently, with reinforcement learning [9]. The approach proposed in this paper is based on Control Barrier Functions (to ensure safety, i.e. collision avoidance) [10] and takes advantage of the particular geometry of the problem to break symmetries (path deconfliction with traffic-inspired rules).

One factor of success for a technology whose final goal is to be adopted among *non-expert users* is to build the users' trust [11]. Potential users will build a personal opinion based on their interaction experience with other AVs, therefore it is important to build tools that allow AVs to be provably safe for the user while *feeling* safe and *familiar* to an external HV's driver that interacts with it. It is in this context of unstructured environments that navigation algorithms must be more flexible when rules are less strict and the behaviour of outside agents may prove to be in contrast with the traffic rules.

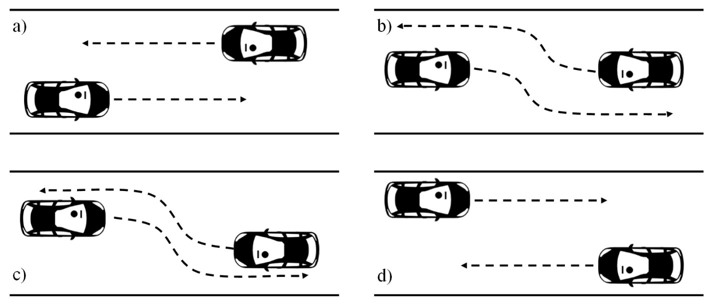
As a case scenario, picture two cars driving in opposite directions down a narrow corridor, as usually seen in parking lots and garages. When the road is clear, both drivers may tend to stay closer to the center of the roadway; consider the road to be wide enough to allow both cars to pass at the same time. When the two cars are facing *close enough*, they will start moving away from the center of the road in order to keep heading towards their respective goal; we will refer to this scenario as the *head-on scenario*.

The head-on scenario is illustrated in more detail in Figure 1, where two vehicles are operating in a right-hand driving environment. In the first example, Fig. 1a, the two agents are driving far enough from the center of the road, allowing both cars to pass; since no conflict is taking place, no further action needs to be taken. In Figs. 1b and 1c, the two cars are heading misaligned, therefore should avoid collision by

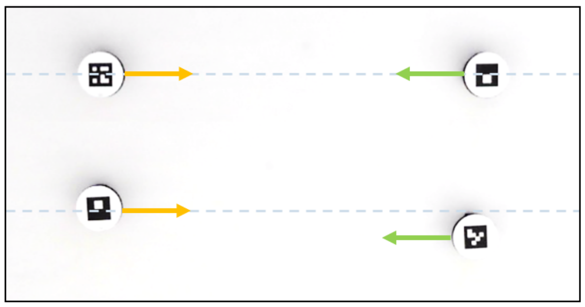
*This work was partially supported through a grant by Ford Motor Company, by the European Union's Horizon 2020 research and innovation programme as part of the projects ILIAD (grant no. 732737) and by the Italian Ministry of Education, and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence).

[1]F. Celi and L. Pallottino are with the Centro di Ricerca E. Piaggio, Dipartimento di Ingegneria dell'Informazione, University of Pisa, Italy `fed.celi@gmail.com, lucia.pallottino@unipi.it`

[2]L. Wang and M. Egerstedt are with the School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia, USA {`liwang, magnus`}`@gatech.edu`



(a) Four different head-on scenarios (right-hand driving). From a) to c) the cars overtake as the road rules impose; in d) the misalignment is strong enough for them to overtake regardless of the rules.



(b) Snapshot of two different head-on alignments for hardware experiments in the Robotarium.

Fig. 1: Head-on scenarios, defined as two agents driving in opposite directions down a narrow corridor.

steering to the right, in compliance with local traffic rules. Finally, in Fig. 1d, the two agents are, again, far enough from the center of the road, yet they are on the wrong side of the road, respectively. However, it is inefficient to force both cars to steer right, as road rules would require, since no conflicting motion is taking place.

Now, consider the scenario of Fig. 1b, showing a hardware implementation with two GRITSbots, the first utilizing an autonomous driving algorithm (AV) and the second controlled remotely via a joystick by a human (HV): a further design goal is to make the AV *aware* of the HV driving direction bias, e.g., if HV steers to the left, so should AV, even if in contrast to local traffic rules.

In this work we discuss a novel strategy of embedding traffic rules in navigation algorithms by i) making use of Barrier Functions to ensure collision avoidance, expanding the work in [12], and ii) by relaxing the rules, allowing agents to break them, under defined circumstances, in order to improve the system's performance and human user's experience.

II. Background

A. Notation

Throughout, $\mathbb{R}$, $\mathbb{R}_0^+$ denote the set of real and non–negative real numbers, respectively. Int$(\mathscr{C})$ and $\partial\mathscr{C}$ denote the interior and boundary of set $\mathscr{C}$, respectively. The open ball in $\mathbb{R}^n$ with radius $\epsilon \in \mathbb{R}^+$ and center at x_0 is denoted by $B_\epsilon(x_0) = \{x \in \mathbb{R}^n \mid \|x - x_0\| < \epsilon\}$. A continuous function $\alpha : [0, a) \to [0, \infty)$ for some $a > 0$ is said to be a class-$\mathscr{K}$ function if it is strictly increasing and $\alpha(0) = 0$. A continuous function $\beta : (-b, a) \to \mathbb{R}$ for some $a, b > 0$ is said to be an extended class-$\mathscr{K}$ function if it is strictly increasing and $\beta(0) = 0$. Given two vectors $\mathbf{x}_i$, $\mathbf{x}_j \in \mathbb{R}^n$, we define the difference $\Delta\mathbf{x}_{ij}$ as $\Delta\mathbf{x}_{ij} = \mathbf{x}_i - \mathbf{x}_j$. Given $f(x)$, $g(x)$ sufficiently smooth in a domain $D \subset \mathbb{R}^n$, we indicate the Lie Derivative as $L_f g(x) = \frac{\partial g}{\partial x} f(x)$.

B. Barrier Functions

A first step towards achieving the goal of securely deconflicting motion paths of vehicles, is to ensure that every action that the autonomous agents take is provably *safe* under appropriate assumptions, i.e., that agent-to-agent collisions will be avoided. To this end, we are going to use Safety Barrier Certificates (SBC) [12], based on Zeroing Control Barrier Functions (ZCBF) [10]. As the explicit purpose of this paper is to understand deconfliction in a formal manner, we are here focusing our attention on an idealized situation from an information-exchange vantage point. Throughout the paper, we thus make the assumption that relevant information about nearby agents are made available, such as each agent's control input and goal position. Note that these types of information can be obtained through other means, e.g., through sensory data. For the sake of clarity, we omit this aspect and instead focus directly on the motion deconfliction problem.

Here, the fundamentals of ZCBF are briefly recalled: consider a dynamical system in control affine form,

$$\dot{x} = f(x) + g(x)u, \tag{1}$$

where the state $x \in \mathbb{R}^n$ and control $u \in U \subset \mathbb{R}^n$, f and g are locally Lipschitz continuous, and the system is forward complete, i.e., $x(t)$ is defined for all $t \geq 0$. By defining a set of conditions on the states of a system, for example imposing a minimal distance between agents, SBC ensure that the time evolution of the system is such that the states always satisfy the original conditions. In this application, the system's states will be the position and velocity of each agent. Therefore, a safe state will be a combination of position and velocity that will not result in a collision, given an acceleration command as input u.

Now, let $\mathscr{C} \subset \mathbb{R}^n$ be the safe set, i.e., the states from which it is possible to avoid collisions. In order to guarantee that a controller u is safe, we need to prove that such a controller renders x forward invariant, i.e., if $x(0) \in \mathscr{C}$, then $x(t) \in \mathscr{C}$ for all $t \geq 0$. We can encode $\mathscr{C}$ through the super-level set of a ZCBF candidate function $h : \mathscr{D} \to \mathbb{R}$, with $\mathscr{C} \subseteq \mathscr{D} \subset \mathbb{R}^n$

$$\mathscr{C} = \{x \in \mathbb{R}^n : h(x) \geq 0\}, \tag{2}$$

which means that $h(x)$, a function of the states in (1), is non-negative if the state x is safe, and negative otherwise.

By differentiating $h(x)$ with respect to time t (note that the state x is time-dependent, however, to simplify notation we denote $x(t)$ simply as x) and substituting it in (1), we gather

$$\frac{dh(x)}{dt} = L_f h(x) + L_g h(x) u. \tag{3}$$

The function $h(x)$ is said to be a ZCBF if there exists an extended class-$\mathscr{K}$ function κ such that

$$\sup_{u \in U} \{L_f h(x) + L_g h(x) u + \kappa(h(x))\} \geq 0 \tag{4}$$

for all $x \in \mathscr{D}$. Given a ZCBF, we can define the admissible control space, as a function of the states, as

$$S(x) = \{u \in U \mid L_f h(x) + L_g h(x) u + \kappa(h(x)) \geq 0\}, \tag{5}$$

with $x \in \mathscr{D}$. We now have all the necessary ingredients to state the following key results, e.g. found in [13].

Theorem 1: Given a set $\mathscr{C} \subset \mathbb{R}^n$ defined by (2) and a ZCBF h defined on $\mathscr{D}$, with $\mathscr{C} \subseteq \mathscr{D} \subset \mathbb{R}^n$, any Lipschitz continuous controller $u : \mathscr{D} \to \mathbb{R}$ such that $u \in S(x)$ for the system in (1) renders the set $\mathscr{C}$ forward invariant. $\mathscr{C}$ is asymptotically stable in $\mathscr{D}$.

In this work, consistent with [12], the particular choice of $\kappa(h(x)) = \gamma h^3(x)$, with $\gamma > 0$, will be adopted, which means that the controller needs to satisfy

$$L_f h(x) + L_g h(x) u + \gamma h^3(x) \geq 0 \tag{6}$$

to render the set $\mathscr{C}$ forward invariant. Controlling the system in (1) with a controller $u \in S(x)$ and defining conditions on what renders a state x safe, encoded in $\mathscr{C}$, we can ensure that, indeed, if $x(0) \in \mathscr{C}$, then $x(t) \in \mathscr{C}$, for all $t > 0$.

C. System Model

Now that we have a general formulation of the Barrier Certificates, we can formulate them in the particular context of autonomous driving. For this purpose, consider N mobile agents moving on the plane, where each agent is indexed by $\mathscr{N} = \{i \mid i = 1, 2, ..., N\}$. We model the agents' dynamics as double integrators, as acceleration limitations play a significant role when avoiding collisions

$$\begin{bmatrix} \dot{\mathbf{p}}_i \\ \dot{\mathbf{v}}_i \end{bmatrix} = \begin{bmatrix} 0_{2\times2} & I_{2\times2} \\ 0_{2\times2} & 0_{2\times2} \end{bmatrix} \begin{bmatrix} \mathbf{p}_i \\ \mathbf{v}_i \end{bmatrix} + \begin{bmatrix} 0_{2\times2} \\ I_{2\times2} \end{bmatrix} \mathbf{u}_i, \tag{7}$$

where $\mathbf{p}_i = (x_i, y_i) \in \mathbb{R}^2$, $\mathbf{v}_i \in \mathbb{R}^2$, and $\mathbf{u}_i \in \mathbb{R}^2$ represent the positions, velocities, and inputs (acceleration commands) for agent i. Velocity and acceleration of the agent are limited by $\|\mathbf{v}_i\|_\infty \leq \beta_i$ and $\|\mathbf{u}_i\|_\infty \leq \alpha_i$, with α_i, $\beta_i \in \mathbb{R}^+$.

The aggregate state of all N agent's positions and velocities will be denoted as $(\mathbf{p}^T, \mathbf{v}^T)^T \in \mathbb{R}^{4N}$. Since the ZCBF $h(x)$ in (2) is a function of the aggregate states, we want to express it as a function of each agents' position and velocity for the system in (7), i.e., $h_{ij}(\mathbf{p}, \mathbf{v})$. Considering the interaction between two agents, i and j, we can define the pairwise set $\mathscr{C}_{ij}$ as

$$\mathscr{C}_{ij} = \{(\mathbf{p}_i, \mathbf{v}_i) \in \mathbb{R}^4 \mid h_{ij}(\Delta\mathbf{p}_{ij}, \Delta\mathbf{v}_{ij}) \geq 0\}, \quad \forall j \in \mathscr{N}_i. \tag{8}$$

As shown in [12], it is possible to express the safety barrier constraint for system (7) as a linear constraint in $\mathbf{u}_i$, which can be represented as

$$A_{ij}\mathbf{u}_i \leq b_{ij}. \tag{9}$$

In particular, given the safety distance $D_S \in \mathbb{R}^+$, the safety barrier constraint satisfying (6) is, as proven in [12],

$$\begin{aligned} -\Delta\mathbf{p}_{ij}^T \Delta\mathbf{u}_{ij} \leq \gamma h_{ij}^3 \left\|\Delta\mathbf{p}_{ij}\right\| - \frac{(\Delta\mathbf{v}_{ij}^T \Delta\mathbf{p}_{ij})^2}{\left\|\Delta\mathbf{p}_{ij}^2\right\|} + \\ + \left\|\Delta\mathbf{v}_{ij}\right\|^2 + \frac{(\alpha_i + \alpha_j)\Delta\mathbf{v}_{ij}^T \Delta\mathbf{p}_{ij}}{\sqrt{2(\alpha_i + \alpha_j)(\left\|\Delta\mathbf{p}_{ij}\right\| - D_s)}}, \end{aligned} \tag{10}$$

for all $i \neq j$. We can write $A_{ij}\mathbf{u}_i \leq b_{ij}$ as

$$A_{ij} = [0, ..., -\Delta\mathbf{p}_{ij}^T, ..., \Delta\mathbf{p}_{ij}^T, ..., 0] \tag{11}$$

and

$$\begin{aligned} b_{ij} = \gamma h_{ij}^3 \left\|\Delta\mathbf{p}_{ij}\right\| - \frac{(\Delta\mathbf{v}_{ij}^T \Delta\mathbf{p}_{ij})^2}{\left\|\Delta\mathbf{p}_{ij}^2\right\|} + \left\|\Delta\mathbf{v}_{ij}\right\|^2 + \\ + \frac{(\alpha_i + \alpha_j)\Delta\mathbf{v}_{ij}^T \Delta\mathbf{p}_{ij}}{\sqrt{2(\alpha_i + \alpha_j)(\left\|\Delta\mathbf{p}_{ij}\right\| - D_s)}}, \end{aligned} \tag{12}$$

where $A_{ij} \in \mathbb{R}^{N-1\times2}$, and $b_{ij} \in \mathbb{R}$. Therefore, for any $\mathbf{u}_i$ satisfying the inequality in (9), we ensure that the control input is safe, that is, the acceleration inputs will keep the state of the system in (1) such that the relative velocity and position of any two agents will not result in a collision.

In the following, we will refer to $\hat{\mathbf{u}}_i$ as the nominal controller, i.e. how we would like to control our system (or agent i). With $\mathbf{u}_i^*$ we will indicate an input to the system that is safe, i.e. that will not result in a collision. Given a nominal controller $\hat{\mathbf{u}}_i$ for the system in (1), if the condition in (9) holds for $\hat{\mathbf{u}}_i$, the nominal control input is safe, therefore making $\mathbf{u}_i^* = \hat{\mathbf{u}}_i$ will pose no harm to the system. In the following section we are going to provide a tool to keep the control $\mathbf{u}_i^*$ safe when the linear inequality in (9) is not satisfied by the nominal controller $\hat{\mathbf{u}}_i$.

D. Decentralized Safety Barrier Certificates

The time evolution of the system in (1) will be regulated by the nominal controller $\hat{\mathbf{u}} = (\hat{\mathbf{u}}_1^T, ..., \hat{\mathbf{u}}_N^T)^T$; as collisions approach, we wish for the actual control input $\mathbf{u}_i^*$ to be safe by respecting the inequality $A_{ij}\mathbf{u}_i^* \leq b_{ij}$ for all $j \neq i$, while staying as close as possible to $\hat{\mathbf{u}}_i$. Since we are able to express the safety constraint as a linear formulation of $\mathbf{u}_i$, we can add a quadratic cost that penalizes deviations from the nominal controller, while ensuring safety, resulting in a Quadratic Programming problem (QP). The decentralized version of the QP-based controller, as suggested in [12], is, for all $i \in \mathscr{N}$,

$$\begin{aligned} \mathbf{u}_i^* = \underset{\mathbf{u}_i \in \mathbb{R}^2}{\operatorname{argmin}} \quad & \left\|\mathbf{u}_i - \hat{\mathbf{u}}_i\right\|^2 \\ \text{subject to} \quad & \bar{A}_{ij}\mathbf{u}_i \leq \bar{b}_{ij}, \quad \forall j \in \mathscr{N}_i \\ & \left\|\mathbf{u}_i\right\|_\infty \leq \alpha_i \end{aligned} \tag{13}$$

where $\bar{A}_{ij} = -\Delta\mathbf{p}_{ij}^T$ and $\bar{b}_{ij} = \frac{\alpha_i}{\alpha_i + \alpha_j} b_{ij}$.

$\mathscr{N}_i$ is the neighboring set of agent i, defined as

$$\mathscr{N}_i = \{j \in \mathscr{N} \mid \|\Delta \mathbf{p}_{ij}\| \leq D^i_{\mathscr{N}},\, j \neq i\}, \tag{14}$$

and where

$$D^i_{\mathscr{N}} = D_s + \frac{1}{2(\alpha_i+\alpha_{\min})}\left(\sqrt[3]{\frac{2(\alpha_i+\alpha_{\max})}{\gamma}} + \beta_i + \beta_{\max}\right)^2 \tag{15}$$

is the size of the radius of the neighbors associated with $\mathscr{N}_i$, where $\alpha_{\min} = \min_{j\in\mathscr{N}}\{\alpha_j\}$, $\alpha_{\max} = \max_{j\in\mathscr{N}}\{\alpha_j\}$ and $\beta_{\max} = \max_{j\in\mathscr{N}}\{\beta_j\}$ are the lower and upper bounds of all agents' acceleration limits and the upper bound of all agents' velocity limits, respectively. Using a controller $\mathbf{u}_i^*$ as the one defined in (13) and $\mathbf{u}^* = (\mathbf{u}_1^{*T}, ..., \mathbf{u}_N^{*T})^T$, ensures safety of the system since collisions are always avoided.

E. Deadlock Detection

The QP problem in (13) ensures safety of the system in (1) regardless of the control input $\hat{\mathbf{u}}_i$ generated by the nominal high-level controller. Although safety is guaranteed, there are situations where the constraints imposed on the control input by (9) are such that the solution to (13) drives the acceleration and velocity of an agent to zero. This prevents the fulfillment of the original goal, if $\hat{\mathbf{u}}_i \neq \mathbf{0}$, and it depends on the geometry of the solution space and of the cost vector of (13).

Consider the admissible control space $\mathscr{P}_i$ for agent i

$$\mathscr{P}_i = \{\mathbf{u}_i \in \mathbb{R}^2 \mid \bar{A}_{ij}\mathbf{u}_i \leq \bar{b}_{ij}, \quad \forall j \in \mathscr{N}_i\}. \tag{16}$$

It is possible to evaluate the size of the feasible control space $\mathscr{P}_i$, termed *width of the feasible set* [14], with a Linear Program (LP)

$$\begin{aligned} \min_{\mathbf{u}_i\in\mathbb{R}^2,\delta_i\in\mathbb{R}} \quad & [\mathbf{0}_{1\times 2} \quad 1]\begin{bmatrix}\mathbf{u}_i^T & \delta_i\end{bmatrix}^T \\ \text{subject to} \quad & [\bar{A}_{ij} \quad -1]\begin{bmatrix}\mathbf{u}_i^T & \delta_i\end{bmatrix}^T \leq \bar{b}_{ij}, \quad \forall j \in \mathscr{N}_i \\ & \|\mathbf{u}_i\|_\infty \leq \alpha_i. \end{aligned} \tag{17}$$

The solution of the LP characterizes how much control margin is left for the strictest safety barrier constraint, i.e, if $\delta_i \leq 0$ then $\mathscr{P}_i$ is not empty. In other words, a negative δ_i indicates how much the b_{ij} of the strictest constraint can be translated before having $\mathscr{P}_i = \emptyset$.

Definition 2.1: An agent i is said to be in deadlock if it does not move, that is, if the solution to (13) is $\mathbf{u}_i^* = 0$ and the speed is zero ($\mathbf{v}_i = 0$), while the nominal control command is $\|\hat{\mathbf{u}}_i\| \neq 0$.

We identify three different deadlock scenarios, based on the geometry of the QP problem in (13) (see Fig. 2)

1) Type 1 deadlock: $\delta_i \leq 0, u_i \in \text{vertex}(\mathscr{P}_i)$;
2) Type 2 deadlock: $\delta_i \leq 0, u_i \in \text{edge}(\mathscr{P}_i)$;
3) Type 3 deadlock: $\delta_i > 0$.

In the following, we are going to build on the results in [12] by expanding the definition of deadlock, allowing agents to take deconflicting actions earlier in time, resulting in a faster responding disengagement. Moreover, given the particular case of interest seen in Fig. 1, we are going to provide formal and statistical results to show how this proposed method ensures collision avoidance in a pattern that matches the problem statement.

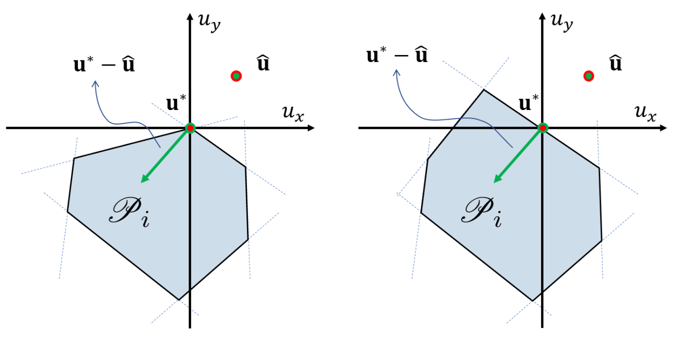


Fig. 2: Graphical representations of the QP problem in (13), with a Type 1 deadlock (left) and Type 2 deadlock (right).

III. Robot–Robot Interaction

As a first step towards understanding the general motion deconfliction problem, we consider a two-agent system, $N = 2$, where the robots are forced to interact in a pattern relatable to Fig. 1. With this assumption of $N = 2$, $\mathscr{N}_i = \{j\},\ i \neq j$ and $i, j = 1, 2$ if $\|\Delta \mathbf{p}_{ij}\| \leq D^i_{\mathscr{N}}$; $\mathscr{N}_i = \{\emptyset\}$, otherwise. This allows us to limit the deadlock types as seen in Definition 2.1: since we will never have more than one inequality constraint in the admissible control space in $\mathscr{N}_i$, for all $i \in \mathscr{N}$ the feasible set will never be empty. For this reason, for any solution, it holds that $\mathbf{u}_i^* \in \text{edge}(\mathscr{P}_i)$ and hence Type 1 and Type 3 deadlock cannot occur.

Definition 3.1: Given $\underline{\alpha}$, $\underline{\beta} \in \mathbb{R}^+$ as the acceleration and velocity thresholds, respectively, and $\underline{\epsilon} \in \mathbb{R}^+$ as the control threshold, an agent i is said to be in a quasi-deadlock if $\|\mathbf{u}_i\| \leq \underline{\alpha}$, $\|\mathbf{v}_i\| \leq \underline{\beta}$, and the nominal control $\|\hat{\mathbf{u}}_i\| > \underline{\epsilon}$.

By introducing a lower bound on the acceleration and velocity, we wish to identify those agents that are slowing down, while the difference between the nominal and the actual controller, $\underline{\epsilon}$, ensures that this slowing evolution is due to the safety constraints introduced in (9) and not by the actual control goal, and hence a risk of collision is approaching.

A. Quasi-deadlock Resolution

If an agent enters a deadlock it stops; [12] presents a deadlock resolution tool, but this requires the agent to have both speed and acceleration equal to zero. This results in a slow-reacting system as, before taking any deconfliciting motion, any agent must come to a complete stop. The quasi-deadlock aims at detecting when an agent is about to enter a deadlock, allowing the controller to take preventive actions without the need for either robot agent to come to a complete halt.

The conflict resolution tool proposed in the present paper as an improvement over [12] can be summarized as follows: given the nominal control $\hat{\mathbf{u}}_i$, if agent i finds itself to be in quasi-deadlock, perturb $\hat{\mathbf{u}}_i$ such that the new nominal control

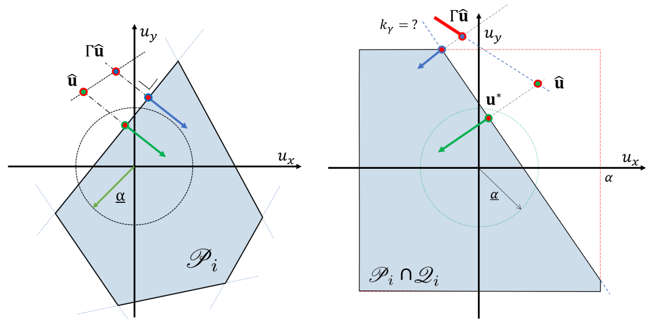


Fig. 3: Graphical representation of $\mathscr{P}_i$, $(N_i = 5)$ with quasi-deadlock resolution. On the left, $\hat{\mathbf{u}}$ is projected, resulting in $\Gamma\hat{\mathbf{u}}$; consequently the optimal solution $\mathbf{u}^*$ changes. On the right, k_γ can be computed precisely as long as the optimal solution to $\Gamma\hat{\mathbf{u}}$ is $\in$ edge$(\mathscr{P}_i \cap \mathscr{Q}_i)$.

is $\Gamma_i\hat{\mathbf{u}}_i$ where $\Gamma_i = I + k_{\gamma i}\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}$ and $k_{\gamma i} \in \mathbb{R}$ for all $i \in \mathscr{N}$. This resolution tool is graphically explained in Fig. 3. $\Gamma\hat{\mathbf{u}}_i$ is a nominal perturbation to the left, or to the right, of $\hat{\mathbf{u}}_i$, depending on the sign of $k_{\gamma i}$.

This translates into $k_{\gamma i}$ encoding the concept of left-hand and right-hand driving, based on its sign. In fact, if $k_{\gamma i} > 0$ ($k_{\gamma i} < 0$) the control input is perturbed to the left (right): when an agent i slows down due to the constraint imposed by the presence of another agent j, agent i will reshape its control input, steering slightly to the left (or right), based on sign$\{k_{\gamma i}\}$. Moreover, according to the module of $k_{\gamma i}$, the perturbation will be more or less aggressive, that is,

$$k_{\gamma i}^{(1)} > k_{\gamma i}^{(2)} \rightarrow \left\|\hat{\mathbf{u}}_i - \Gamma_i^{(1)}\hat{\mathbf{u}}_i\right\| \geq \left\|\hat{\mathbf{u}}_i - \Gamma_i^{(2)}\hat{\mathbf{u}}_i\right\|. \tag{18}$$

For this reason, we will also refer to $k_{\gamma i}$ as agent i's *direction bias*. The quasi-deadlock resolution is summarized in Algorithm 1, where the subroutine *Decentralized_LP* computes the width of the feasible set as in (17) and *Decentralized_QP* computes the optimal safe controller as in (13).

Algorithm 1 Quasi-deadlock resolution

```
input $\mathbf{u}_i, \hat{\mathbf{u}}_i, \mathbf{v}_i$
$\delta \leftarrow$ Decentralized_LP
if $\|\mathbf{u}_i\| \leq \underline{\alpha}$ & $\|\mathbf{v}_i\| \leq \underline{\beta}$ & $\|\hat{\mathbf{u}}_i\| > \underline{\epsilon}$ & $\delta \leq 0$ then
    $\hat{\mathbf{u}}_i \leftarrow \Gamma_i\hat{\mathbf{u}}_i$
$\mathbf{u}_i^* \leftarrow$ Decentralized_QP$(\hat{\mathbf{u}}_i)$
return $\mathbf{u}_i^*$
```

B. Direction Bias Estimation

Assume that every agent's goal is shared on the network, therefore $\hat{\mathbf{u}}_j$ is known by every agent i, while $k_{\gamma j}$ is not. We also assume that the final optimal control $\mathbf{u}_j^*$ is known by every agent. As a first approach, we are going to ignore limits on the acceleration input, i.e. $\alpha_i = +\infty$. The problem to be faced is, can agent i learn $k_{\gamma j}$ from observing j's behavior?

Proposition 1: Consider the QP problem in (13), where $\alpha_i = +\infty, \forall j \in \mathscr{N}$, with the quasi-deadlock resolution defined in Algorithm 1. If the nominal controller $\hat{\mathbf{u}}_j$, and the solution to the QP problem $\mathbf{u}_j^*$ are known for all $j \in \mathscr{N}_i$, then agent i can compute $k_{\gamma j}$.

Proof: Let us again consider the simple scenario of two agents driving toward each other (position swapping) as described in the previous sections. Since we are solving the problem in (13), a QP problem in $\mathbb{R}^2$, we know that the solution $\mathbf{u}_j^*$ will solve all inequality constraints and at least one of the inequalities will actually be solved as an equality [15]. Using the assumption that only two agents are present in the network, we can conclude that the matrix A_{ji} will always have one row, and therefore the solution will always be along the line $\bar{A}_{ji}\mathbf{u}_j^* = \bar{b}_j$, where $\bar{A}_{ji} \in \mathbb{R}^{1\times 2}$ and $\bar{b}_j \in \mathbb{R}$. We can express the cost function in (13) as

$$\|\mathbf{u} - \Gamma\hat{\mathbf{u}}\|^2 = \mathbf{u}^T\mathbf{u} + \hat{\mathbf{u}}^T\Gamma^T\Gamma\hat{\mathbf{u}} - 2\mathbf{u}^T\Gamma\hat{\mathbf{u}}, \tag{19}$$

where, to ease up notation, we define $\hat{\mathbf{u}}_j = \hat{\mathbf{u}} = [\hat{u}_x, \hat{u}_y]^T$ and $\mathbf{u}_j = \mathbf{u} = [u_x, u_y]^T$. Using the Lagrangian multiplier $\lambda \in \mathbb{R}$, we can express the equality

$$H = \mathbf{u}^T\mathbf{u} + \hat{\mathbf{u}}^T\Gamma^T\Gamma\hat{\mathbf{u}} - 2\mathbf{u}^T\Gamma\hat{\mathbf{u}} + \lambda(A\mathbf{u} - b) \tag{20}$$

and, differentiating H along $\mathbf{u}$ we get

$$\frac{\partial H}{\partial \mathbf{u}} = 2\mathbf{u} - 2\Gamma\hat{\mathbf{u}} + \lambda A^T = 0. \tag{21}$$

Defining $\bar{A}_{ji} = A = [a_1, a_2]$ and $\bar{b}_j = b$, and recalling that $A\mathbf{u} = b$, we obtain $\mathbf{u} = \Gamma\hat{\mathbf{u}} - \frac{1}{2}\lambda A^T$ from the previous equation and we get

$$\lambda = 2(AA^T)^{-1}(A\Gamma\hat{\mathbf{u}} - b). \tag{22}$$

Finally, given that $\mathbf{u} = \Gamma\hat{\mathbf{u}} - \frac{1}{2}\lambda A^T$ we conclude that

$$\mathbf{u} = \Gamma\hat{\mathbf{u}} - (AA^T)^{-1}(A\Gamma\hat{\mathbf{u}} - b)A^T \tag{23}$$

which is linear with respect to the free parameter Γ. We define $(AA^T)^{-1} = \gamma$ and note that $\gamma \in \mathbb{R}, \quad \forall A \in \mathbb{R}^{1\times N}$ and $AA^T \neq 0$, since $A \neq \mathbf{0}$ being it a distance (see (11)); moreover $\|\hat{\mathbf{u}}_i\| > 0$ from the definition of quasi-deadlock (Def. 3.1). Equation (23) is a two-equation system with one unknown parameter, k_γ; solving for $\mathbf{u}_x$, we obtain:

$$k_\gamma^{\text{est}} = \frac{\hat{u}_x - u_x - \gamma a_1(a_1\hat{u}_x + a_2\hat{u}_y - b)}{\hat{u}_y + \gamma a_1(a_2\hat{u}_x - a_1\hat{u}_y)} \tag{24}$$

What this means is that, for a two-agent system, if we know the objective controller $\hat{\mathbf{u}}$ and the optimal safe controller $\mathbf{u}^*$ for every agent, it is possible to obtain k_γ of the agents once they enter a quasi-deadlock scenario, i.e., $\Gamma \neq 1$, since if $\Gamma = 1 \rightarrow k_\gamma = 0 \rightarrow \|\mathbf{u} - \Gamma\hat{\mathbf{u}}\|^2 = \|\mathbf{u} - \hat{\mathbf{u}}\|^2$. ☐

In Proposition 1 we did not take into consideration an important constraint of the QP problem in (13): the acceleration limits $\|\mathbf{u}_j\|_\infty \leq \alpha_j$. These limits introduce a saturation in the system in the form of inequality constraints, limiting the admissible control space $\mathscr{P}_j \cap \mathscr{Q}_i$, where $\mathscr{Q}_i = \{\mathbf{u}_i \in \mathbb{R}^2 \mid \|\mathbf{u}_i\|_\infty \leq \alpha_i\}$. This limits the $k_{\gamma j}$ estimation tool we developed.

Proposition 2: Consider the QP problem in (13), with

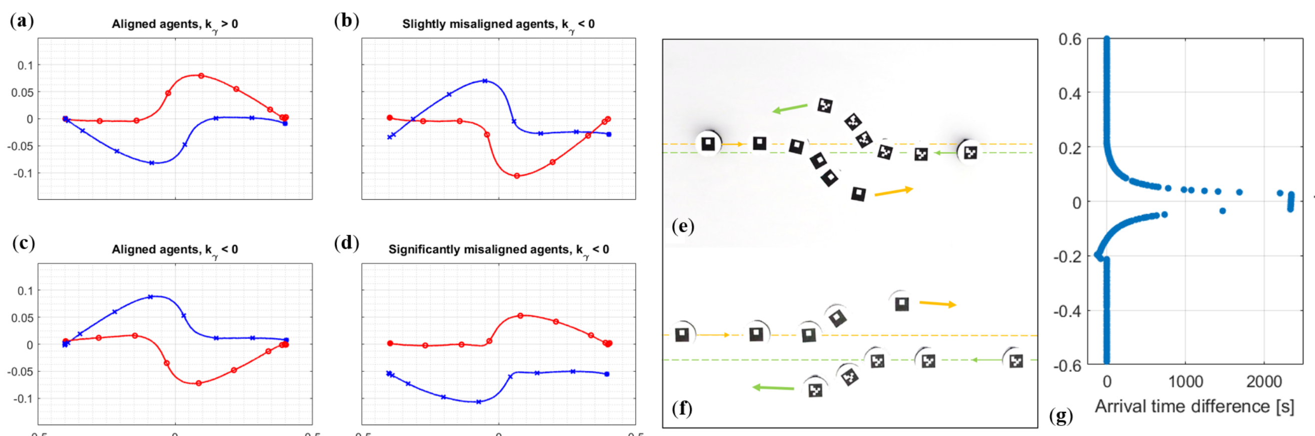


Fig. 4: Experimental results for different head on resolutions. Figs. (c)–(e), and (d)–(f) show the same experiments, respectively, proposed as a series of consecutive snapshots from the Robotarium. In (g) the arrival time comparison (original formulation - quasi-deadlock resolution) of two agents heading towards each other, with a simulation time limit of 60 seconds is shown. The initial displacement between the two agents is randomly selected, for a total of 500 simulations.

the quasi-deadlock resolution defined in Algorithm 1. If the nominal controller $\hat{\mathbf{u}}_j$, and the solution to the QP problem $\mathbf{u}_j^*$ are known for all $j \in \mathscr{N}_i$, then agent i can compute a lower bound on $k_{\gamma j}$, where the solution to (24) is such that $\left\|k_{\gamma j}^{\text{est}}\right\| \leq \|k_{\gamma j}\|$.

Proof: As described in Fig. 3b, if the projection $\Gamma\hat{\mathbf{u}}_j$ falls along the half-line on the right of $\Gamma\hat{\mathbf{u}}_j$, the new optimal solution $\mathbf{u}_j^*$ will fall on the vertex of $\mathscr{P}_j \cap \mathscr{Q}_i$, regardless of the actual $k_{\gamma j}$. However, the sign of $k_{\gamma j}$ will still be computed correctly and the resulting $|k_{\gamma j}|$, although wrong, will provide a lower bound on the actual value of $k_{\gamma j}$. The existence of $\mathbf{u}_j^*$ is guaranteed by the definition of Type 2 quasi-deadlock. $\square$

In this Section we propose a simple mathematical model to encode rule–based quasi-deadlock resolution, thanks to the direction bias k_γ: left vs right handed, $sign\{k_\gamma\}$, smoother vs rougher, $abs\{k_\gamma\}$. We then provide a way, for each agent i to compute the driving direction bias k_j for $j \in \mathscr{N}_i$: this allows each agent to gain knowledge of the driving behavior of the other nearby.

C. Experimental Results

The Decentralized SBC with quasi-deadlock detection and resolution is here implemented and tested on the Robotarium at the Georgia Institute of Technology [16], a remotely accessible swarm-robotics testbed. In particular, two AVs are controlled with the goal of swapping positions on the plane; four series of experiments are proposed, in close relation to the problem statement of Fig. 1. The results are shown in Fig. 4.

In Figs. 4a and 4b the two agents are perfectly aligned, both implementing, respectively, right ($k_\gamma < 0$) and left ($k_\gamma > 0$) handed driving bias. In Fig. 4c, the two agents are slightly misaligned to their left on the vertical axis (both agents have $k_\gamma < 0$); however, since the misalignment is not significant, both agents steer to the right, as the rules of the road require. In the experiment of Fig. 4d, instead, the misalignment is significant, as in Fig. 1d, resulting in both agents holding their side of the road. Figs. 4e and 4f present these two experiments as a series of snapshots, to better visualize the agents' behaviour.

As a further performance test, simulations are performed in order to compare the behaviour of the Decentralized SBC with quasi-deadlock resolution (QD), as presented in this paper, with their original formulation, as presented in [17] (NQD). In Fig. 4g the results are shown for different misalignment conditions (ordinate); the simulations are performed with the same starting conditions with both formulations and the difference $(NQD_{time} - QD_{time})$ in the time arrival of the agents is shown (abscissa). Positive values of the time difference indicate a faster goal achievement for the QD resolution; a limit on the simulation time is fixed at 60 seconds. As the results suggest, the QD resolution outperforms the NQD around the point of interest (perfect alignment). We observe that there is an area where the NQD is faster; however, we argue that overall this new resolution is faster and it never encounters a new deadlock (in the case of QD, all goals are met before the time limit).

A video of the experiments is available online at `https://goo.gl/ctzmM3`.

IV. Human–Robot Interaction

Consider now the case of a mixed two-agent network, where the first agent is autonomous and implements Decentralized SBC with quasi-deadlock resolution as described in Section III, and the second agent is remotely controlled by a human driver. The human driver controls the robot agent via a joystick or keyboard and is able to set the acceleration input at any instant; the human is asked to drive the robot *as straight as possible* to a specific goal point in the xy–plane, resulting in a head-on scenario like the one described in Fig. 1b. We make the assumption that the human driver is not ill-intentioned, i.e., he or she will not try to deliberately hurt the system, for example by avoiding to take any deconflicting action when a collision is imminent. Instead, the focus is on establishing driving biases as a way of deconlicting motions under relatively benign driving conditions.

We note that a human operator will not necessarily drive enforcing the notion of SBC as discussed in the previous sections of this paper. Safety of the systems in (13) can be guaranteed only when all agents respect the conditions of the problem. In the experiments presented in this section, we will assume that the human drives without the intent of harming the system, avoiding collision by steering (changing direction) when she or he considers it to be necessary. Even though we have no reason to assume that human drivers employ SBC, the autonomous vehicle will act as if the human was actually using SBC. As will be seen in the subsequent section, this assumption does indeed provide insight into a human driver's behavior when related to her or his driving direction bias.

Assume that the AV has perfect knowledge of the HV input controller and of the HV control goal, e.g., a position in the xy-plane. Let $\mathbf{u}_i$ be the control input for HV, i.e. the control signal imposed from the joystick. HV implements a modified version of SBC, i.e., it computes the inequality constraints for the problem in (13) as if in the autonomous case, without however applying the computed optimal control, since it is directly imposed by the outside user. The AV can observe the behavior of HV and has all the necessary tools to mimic its QP problem, as seen in Section III for the fully autonomous case.

Let $\mathbf{u}_{0i}(t)$ be the control input expected from agent i at any instant t and, dropping the notion of time for ease of notation, let $\mathbf{u}_{0i}^{\perp}$ and $\mathbf{u}_i^{\perp}$ be the projection of $\mathbf{u}_{0i}$ and $\mathbf{u}_i$, respectively, on the equality constraint $A_{ij}\mathbf{u}_i = b_i$, where $A_{ij} \in \mathbb{R}^{1\times 2}$ and $b \in \mathbb{R}$. Let $B_\epsilon(\mathbf{u}_{0i}^{\perp})$ be the ball of radius ϵ centered at $\mathbf{u}_{0i}^{\perp}$ and let $\mathscr{L}_i = \{\mathbf{u}_i \mid \mathbf{u}_i^{\perp} \in B_\epsilon(\mathbf{u}_{0i}^{\perp})\}$. This is exlained graphically in Fig. 5.

Definition 4.1: A user's control input $\mathbf{u}_i$ is said to be *goal compatible* with the control goal $\mathbf{u}_{0i}$ if $\mathbf{u}_i \in \mathscr{L}_i$.

If the autonomous agent finds that HV is driving with goal compatible inputs, no further actions will be taken, as AV considers that the human driver is trying to achieve her or his original goal, i.e., it is accelerating or decelerating in the direction of the goal. On the other hand, if $\mathbf{u}_i \notin \mathscr{L}_i$, AV considers HV to be perturbing its control away from the original goal, e.g., to avoid a collision, as seen in Fig. 5.

With this new tool we can construct the human–robot interaction problem as the robot–robot problem of Section III, where $\mathbf{u}_0^{\perp} = \mathbf{u}^*$ and $\mathbf{u}^{\perp} = \mathbf{u}_Q^*$, where $\mathbf{u}_Q^*$ is the optimal solution associated to $\Gamma\hat{\mathbf{u}}$ of Algorithm 1.

A. Experimental Results

A limited number of experiments is performed (more than 10) to validate the illustrated results. In particular, the same human subject is presented with various head on scenarios, each time with a different initial value of the autonomous agent k_γ; the human subject was asked to deconflict the agent according to personal preference and the reaction of the autonomous agent and the $k_{\gamma H}$ estimation was analyzed.

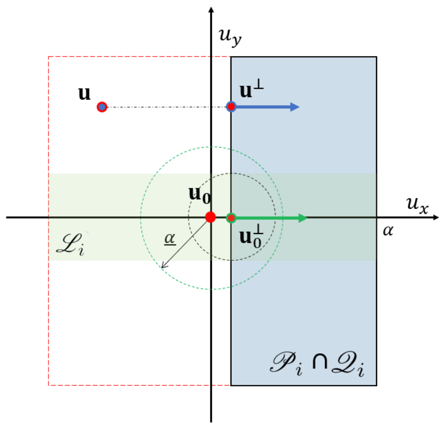


Fig. 5: Graphical representation of $\mathscr{P}_i$ of HV as seen from AV in a simple two agents network. In this example, $\mathbf{u} \notin \mathscr{L}_i$, therefore the control $\mathbf{u}$ is considered to be not goal compatible, revealing a right-hand biased for HV.

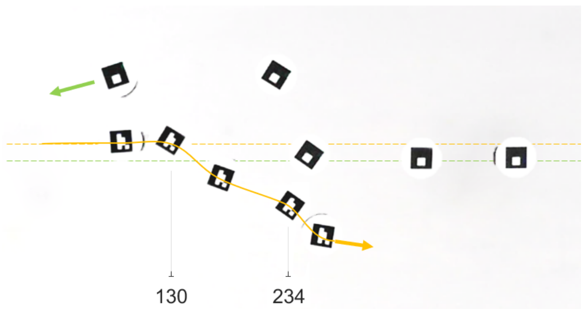
No significant difference in the behaviour of the autonomous agent was observed.

Experimental results for one of such experiments are shown here in Fig. 6. In particular, the AV agent is programmed with $k_{\gamma A} > 0$ (left-hand driving) and the HV is allowed to steer to the left or to the right, as controlled by a human driver via a joystick. The autonomous agent ignores whether the upcoming agent is autonomous (and using SCB) or is controlled by a human. Despite the lack of this information, given the generality of the approach, the AV is able to estimate the driving direction bias and can update its internal value of $k_{\gamma H}$ accordingly.

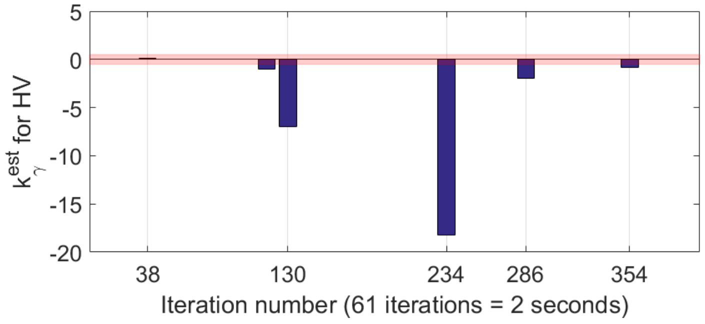
In the experiment of Fig. 6, the autonomous agent knows that the HV needs to reach a point in the plane along its current trajectory of motion; any acceleration input along that direction will not influence the computation of $k_{\gamma H}$. In Fig. 6a the evolution of one particular experiment performed on a pair of GRITSbots is shown and in Fig. 6b the $k_{\gamma H}$ estimation performed by the AV is plotted against time. At about 1 second from the start of this experiment (iteration 38) a *goal compatible* acceleration input is given: the user commands the agent to accelerate forward, towards the goal. Later in the experiment, just after the 4-second mark (iteration 125) the user starts steering the robot agent significantly, resulting in a well-defined sign of $k_{\gamma H}$, suggesting that the HV is, indeed, steering to the negative values of k_γ, i.e. to the right of the goal.

V. Conclusions

A tool to solve conflicting motion paths for networks of multiagent robots was presented. Focusing on the interaction of two robot vehicles heading towards a collision, we provided a tool to deconflict the agents' motion, while ensuring safety with the notion of traffic rules. Based on the agents' relative position and velocity, the agents will act according to road rules, or decide to break them if



(a) Snapshot evolution of head-on interaction between HV (left) and AV (right).



(b) k_γ of HV, estimated by AV. The shaded area highlights the actions that are considered *goal compatible* and, therefore, ignored.

Fig. 6: Head-on scenario with mixed human-robot interaction. The AV starts with a left driving bias, but updates it based on HV's behavior.

particular conditions are present. We introduced the notion of driving direction bias, as the direction (left or right) that will preferably be used by the autonomous agent to deconflict its motion, together with a tool to estimate other agents' direction bias. Finally, we investigated a strategy to adapt the model, presented for the autonomous vehicles, to a mixed human and robot interaction, where the robot vehicle morphs its driving direction bias based on the observed human behavior.

Even though there is no reason to believe that human drivers employ barrier certificates, the construction in this paper suggests that this assumption still allows for an effective deconfliction strategy between human and autonomous drivers. Moreover, we believe that these results can be further generalized, removing the need to share the agents' state on the network and exploiting tools already present in the literature to estimate the local agents' states from sensor data [18]. Indeed, the performed experiments in the Human-Robot interaction scenario are limited to just one human subject. Further work is required to present a statistically significant set of experiments that should aim at benchmarking the behaviour of this resolution tool with different human subjects and, therefore, driving behaviours.

References

[1] Nicole A. Thomas, Owen Churches, Ian White, Christine Mohr, Yann Schrag, Sabrina Obucina, and Michael E.R. Nicholls. An investigation of left/right driving rules on deviations while walking. *PLoS ONE*, 12(10):1–11, 2017.

[2] Sujeong Kim, Stephen J. Guy, Wenxi Liu, David Wilkie, Rynson W.H. Lau, Ming C. Lin, and Dinesh Manocha. BRVO: Predicting pedestrian trajectories using velocity-space reasoning. *International Journal of Robotics Research*, 34(2):201–217, 2015.

[3] S A E International. U.S. Department of Transportation's New Policy on Automated Vehicles Adopts SAE International's Levels of Automation for Defining Driving Automation in On-Road Motor Vehicles. *SAE international*, page 1, 2016.

[4] Tirthankar Bandyopadhyay, Sung Kok Won, Emilio Frazzoli, David Hsu, Wee Sun Lee, and Daniela Rus. Intention-aware motion planning. In *Algorithmic Foundations of Robotics X*, pages 475–491. Springer-Velag, 2013.

[5] Craig Boutilier. Planning, learning and coordination in multiagent decision processes. *Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*, pages 195–210, 1996.

[6] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9(1):427–438, 2013.

[7] Lucia Pallottino, Vincenzo G. Scordio, Antonio Bicchi, and Emilio Frazzoli. Decentralized cooperative policy for conflict resolution in multivehicle systems. *IEEE Transactions on Robotics*, 23(6):1170–1183, 2007.

[8] L. P. Perera, J. P. Carvalho, and C. Guedes Soares. Fuzzy logic based decision making system for collision avoidance of ocean navigation under critical collision conditions. *Journal of Marine Science and Technology*, 16(1):84–99, 2011.

[9] Yu Fan Chen, Miao Liu, Michael Everett, and Jonathan P. How. Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 285–292, 2017.

[10] Aaron D. Ames, Xiangru Xu, Jessy W. Grizzle, and Paulo Tabuada. Control Barrier Function Based Quadratic Programs for Safety Critical Systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2017.

[11] Phillip Gough, Caitlin de Berigny Wall, and Tomasz Bednarz. Affective and Effective Visualisation: Communicating Science to Non-Expert Users. *IEEE Pacific Visualization Symposium Affective*, pages 335–339, 2014.

[12] Li Wang, Aaron D. Ames, and Magnus Egerstedt. Safety barrier certificates for collisions-free multirobot systems. *IEEE Transactions on Robotics*, 33(3):661–674, 2017.

[13] Xiangru Xu, Paulo Tabuada, Jessy W. Grizzle, and Aaron D. Ames. Robustness of Control Barrier Functions for Safety Critical Control. *IFAC-PapersOnLine*, 48(27):54–61, 2015.

[14] Benjamin Morris, Matthew J. Powell, and Aaron D. Ames. Sufficient conditions for the lipschitz continuity of QP-based multi-objective control of humanoid robots. *Proceedings of the IEEE Conference on Decision and Control*, pages 2920–2926, 2013.

[15] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[16] Daniel Pickem, Paul Glotfelter, Li Wang, Mark Mote, Aaron Ames, Eric Feron, and Magnus Egerstedt. The Robotarium: A remotely accessible swarm robotics research testbed. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1699–1706, 2017.

[17] Li Wang, Aaron Ames, and Magnus Egerstedt. Safety barrier certificates for heterogeneous multi-robot systems. In *Proceedings of the American Control Conference*, volume 2016-July, pages 5213–5218, 2016.

[18] Stephen Prajna, Ali Jadbabaie, and George J. Pappas. A framework for worst-case and stochastic safety verification using barrier certificates. *IEEE Transactions on Automatic Control*, 52(8):1415–1428, 2007.