Comparison of Control and Cooperation Frameworks for Blended Autonomy

Gregory Provan[1] and Yves Sohége[2]

Abstract— **Autonomous vehicles, e.g., cars, aircraft or ships, will need to accept some degree of human control for the coming years. Consequently, a method of controlling autonomous systems (ASs) that integrates control inputs from humans and machines is critical. We describe a framework for blended autonomy, in which humans and ASs interact with varying degrees of control to safely achieve a task. We empirically compare collaborative control tasks in which the human and AS have identical or conflicting objectives, under three main control frameworks:(1) leader-follower control (based on Stackelberg games); (2) blended control; and (3) switching control. We validate our results on a car steering control model, given communication delays, noise and different collaboration levels.**

I. Introduction

Fielded autonomous systems (ASs) need to interact with humans in a variety of ways, and hence need a framework to enable that. Here, we focus on ASs that can incorporate a variable degree of human control, which we call *blended autonomy*. For example, forthcoming autonomous cars will allow drivers to set a variety of autonomy levels; for example, lower levels include cruise control and/or collision avoidance, and full autonomy incorporates all driving functionality. Analogously, UAVs flying in commercial airspace will need to follow Air Traffic Control (ATC) instructions, entailing a mechanism to integrate human voice commands into autonomously-generated flight planning.

Our objective is to develop a control-theoretic framework for blended autonomy that will allow humans to interact with a collection of ASs. This is an extension of prior work on tele-operation [2], which assumes continuous human control of a vehicle. This work also extends frameworks for automotive autonomy, e.g., [10], in that we incorporate the impacts of noise and imperfect world models into the collaborative control paradigm.

We propose a framework that focuses on the following key properties. First, we describe a system that can flexibly operate in multiple autonomy levels as a multi-agent system with human and AS agents that must interact to accomplish a task. Second, the human/AS interaction may have varying degrees of cooperation and information sharing among the agents; incomplete information sharing may arise due to lack of transmission of state information or communications losses. We solve our blended autonomy problem using an optimal control framework, and demonstrate the resulting properties of Model-Predictive control (MPC) algorithms when applied to this task.

Our contributions are as follows:

- We formulate a control-theoretic framework for ASs incorporating varying degrees of human control inputs.
- We specify the AS framework in terms of a multi-agent hybrid system, and compare three control algorithms: (1) leader-follower (e.g., as done with Stackelberg game approaches [9]), (2) blended-control, based on weighted-sum optimization [8], and (3) switching-control, with control assigned to the agent whose output is closest to the optimal output.
- We illustrate our approach using MPC for a car steering system jointly controlled by human and AS inputs.

II. Related Work

Teleoperation is an area in which a human operates a vehicle at a distance. Most approaches to teleoperation assume full human control (e.g., [2]). However, some recent work has included blended human/robot control, e.g., [6].

A large body of work has been published on automotive blended human/vehicle control, e.g., [1], [10], [15]. The majority of this work assumes that human and AS controls are cooperative, and that no unsafe controls are possible. Recent work that incorporates safety includes [1]. In reference to this work on the automotive domain, we examine the impact of real-world conditions (delays/noise in applying control inputs) on steering control tasks, using a framework that can be extended to more than just 2 actors (driver and AS).

Control theory has been used for developing algorithms for blended autonomy. Common approaches include MPC [7] and optimal control [14]. Our approach adopts multi-objective optimization, for which a large body of work exists for general problems [8] and for control applications [5].

Game theory has been used for action generation in blended autonomy. For example, [3] describes experiments with different cooperation/non-cooperation strategies for human/AS. [9] proposes an algorithm based on a Stackelberg equilibrium for cooperative steering control.

Work also exists in attempting to model the driver's intent. For example, [4] proposes the use of motion primitives for cooperative blended-control driver assistance systems. [3] proposes several human models for robot interactions. [11] examines modes for adaptation in human-AS collaboration.

III. Technical Description

This section describes our technical formulation. We present a framework that can accommodate arbitrary numbers of human and AS agents, but for simplicity we restrict

The paper has been supported by SFI grants 12/RC/2289 and 13/RC/2094.

[1,2]Computer Science Department, UCC, Cork, Ireland
[1]g.provan@cs.ucc.ie, [2]yves.sohege@insight-centre.org

our discussion to a single instance of human and AS agent. We denote human with H and AS with Υ. The actions at time t are the joint human and AS actions, denoted $u(t) = (u_H(t), u_\Upsilon(t))$, and the total space of actions is $\mathcal{U}$. We assume that we have a set θ of parameters in our system model Φ. We can define a high-level version of our task as an optimal control problem:

Definition 1 (Blended-autonomy task): A system $\mathscr{S}$ solves a blended-autonomy task with objective function $\mathscr{J}$ using human and AS agents who collectively interact to optimize $\mathscr{J}$ within world state Ξ_w, where human and AS agents may take on differing levels of autonomy, by selecting actions that maximize $\mathscr{J}$:

$$u^* = \arg\max_{u \in \mathcal{U}} \mathscr{J}(u, \theta). \tag{1}$$

We will make this definition more precise in the following sections.

A. Multi-Agent Framework

Definition 2 (Agent ω): We define an agent $\omega \in \Omega$ using the tuple $\langle c, \Phi, \mathcal{U}, r, \mathscr{J} \rangle$, where

- c is the agent type, $c \in \{H, \Upsilon\}$.
- Φ is the agent model, denoting the agent's hybrid dynamics.
- $\mathcal{U}$ is the system action space, such that $\mathcal{U} = \mathcal{U}_H \cup \mathcal{U}_\Upsilon$.
- r is the agent's reward function.
- $\mathscr{J}$ is the agent's objective function.

We assume that we can define three state types in our system, $\xi_w \in \Xi_w, \xi_\Upsilon \in \Xi_\Upsilon$, and $\xi_H \in \Xi_H$, denoting states for the world, AS and human, respectively.

We define a blended-autonomy system (BAS) as a multi-agent hybrid system as follows:

Definition 3 (BAS $\mathscr{S}$): A BAS is a multi-agent system consisting of a collection of agents $\omega_i \in \Omega, \quad i = 1, ..., n$, who attempt to achieve a collective task with system reward function R_w and objective function $\mathscr{J}_w = \varsigma(\mathscr{J}_1, ..., \mathscr{J}_n)$, where ς is an aggregation function for the agents. We characterize the agent interaction protocol using χ.

B. Autonomous Modes

In this article we generalize the notion of mode from an indicator for operational state to also include the autonomy levels of agents. Our notion of mode corresponds to the notion of autonomy level in many application domains. For example, in the automotive domain the SAE has defined six levels of autonomy for self-driving vehicles, with level 0 denoting full human control to level 5 denoting full AS control. The levels 1 through 4 consider a blended approach where the mode defines the driving mode (e.g., highway cruising) with some autonomous capabilities (e.g., steering, cruise control). We assume that an identical operational mode can be achieved through human or AS control. We formalize these notions in the following sections.

Mode Specification: We assume that a system $\mathscr{S}$ can operate in a discrete set of system modes $\Gamma = \{\gamma_1, \cdots, \gamma_k\}$. A mode is characterized by both the operational state and the health state of $\mathscr{S}$; here we focus only on the operational state. More specifically, we define an agent's operational mode in terms of its autonomy level and state. For example, an air vehicle's operational mode can be in *cruise*, and it's autonomy mode either autopilot or human control.

We formalize modes as follows. A system $\mathscr{S}$ can be in one of several operational modes at any time, denoted $\Gamma_O = \{\gamma_O^1, \cdots, \gamma_O^k\}$. Given an operational mode, the system can be in one of several autonomy modes, i.e., where $\mathscr{S}$ is controlled by both the AS or human to varying degrees.

Definition 4 (Mode γ_k): A system $\mathscr{S}$ can be in one of several modes, $\gamma \in \Gamma$, which is given by the pair of operational and autonomy modes: $\Gamma \subset \Gamma_O \times \Gamma_A$.

We use the notion of autonomy map to capture the different human/AS mode specifications that lead to task completion.

Definition 5 (Autonomy Map): We can compute the system's autonomy level using autonomy map $\varrho : \Gamma_O \times \Gamma_A \rightarrow \Gamma$. We can use this map to identify, for example, the AS autonomy level necessary to achieve a task given as input the human autonomy level and required mode.

We assume that Γ can take on a discrete set of values we can partition these values into safe and unsafe modes: $\varphi : \Gamma \rightarrow \{0, 1\}$, where 0 denotes unsafe and 1 denotes safe. For example, for a car with human and AS inputs, the overall autonomy level must be equivalent to full control due to inputs from human and AS for the vehicle to be safe.

C. Dynamical System Model

We assume that the system may operate using three different models: a world model Ψ_w, and models maintained by the AS and human, Ψ_Υ and Ψ_H, respectively.

Agent i possesses a unique dynamical model, which for a single mode $\gamma \in \Gamma$ we denote by a discrete-time equation

$$\begin{aligned} \xi_i(k+1) &= A_i \xi_i(k) + B_i u_i(k) \\ z_i(k) &= C_i \xi_i(k). \end{aligned} \tag{2}$$

The system evolves according to a state transition function $T : \Xi_w \times \mathcal{U}_\Upsilon \times \mathcal{U}_H \rightarrow \Xi_w$. The agents can also switch system modes in a discrete manner.

Since this system is hybrid, mode switches can occur due to discrete switch commands or to continuous state evolution.

IV. Multi-Agent Coordination

A. Assumptions

This section describes our assumptions. We assume that we have a collection of n agents, each characterized by tuple $\langle c, \Phi, \mathcal{U}, r, \mathscr{J} \rangle$. We denote the (possibly unique) i^{th} reward and objective function as r_i, $\mathscr{J}_i$, and Φ_i, respectively, for $i = 1, ..., n$. Φ_i is given by equations 2, for a given mode. This framework enables us to analyze the impact of agents who may have different objectives, or agents who may have different sensors, and hence obtain different sensor outputs $z(k)$ given world state $\xi_w(k)$.

Objectives: We assume that agent i has objective function $\mathscr{J}_i$, and that there exists a system-level objective function $\mathscr{J}_w$ that we want to optimize subject to Ψ_w, u_H, u_Υ, collaboration χ, plus safety and other constraints.

B. Collaboration Types

[3] defines, in a qualitative manner, four key types of collaboration for interaction χ between human and AS: collaboration that is (1) perfect; (2) collaborative but approximately optimal; (3) subject to incomplete or corrupted communication; (4) non-collaborative.

The key question is whether every agent knows the autonomy mode and reward model for the other agents. These data can be communicated among agents, but if we have disturbances in collaboration (e.g., incomplete communication) then complete information will not be globally held. In this article we empirically compare the impact of agents having different models, objective functions, and incomplete/corrupted communications.

V. Control Approaches

In the general case, we assume that the human and AS generate controls through solving two different optimization problems. This may arise in several situations. For example, a human air traffic controller (ATC) may specify a landing plan for a UAV that optimizes over all planes within the ATC's sector, where an individual UAV has computed a plan that optimizes its individual fuel consumption. In a different case of a car, the human driver may may be tired and issue dangerous commands while the AS may compute safe commands for the car.

In the following we explore situations in which the human model Φ_H is either incomplete or receives incomplete sensor inputs, so that the human actions u_H are sub-optimal. This contrasts with work that either assumes a perfect human model Φ_H (e.g., [1]) or learns/estimates Φ_H (e.g., [11]).

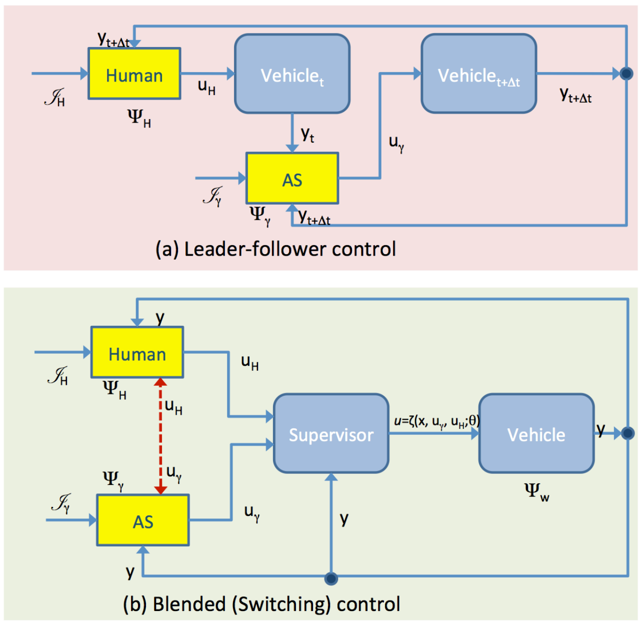

Fig. 1. Architecture for blended autonomy system: (a) shows the leader-follower approach, and (b) shows the blended approach

We now summarize three control approaches: Leader-Follower, Blended and Switching. Figure 1 shows the control approaches (applied at each step t) that we compare.

A. Leader-Follower Control

We use a Leader-Follower (LF) model to study blended autonomy in which we establish a control regime with the human as the leader and the AS as a follower. Figure 1(a) shows the Leader-Follower (LF) approach, where the human agent applies a control $u_H(t)$ to the vehicle, then the AS agent observes $y(t)$ and applies $u_\Upsilon(t+\delta t)$ to the vehicle (for some small δt), and then the process repeats. A Stackelberg equilibrium is an equilibrium solution composed of the optimal strategies of the leader and the followers [9]. Given observability of control settings for both agents, we can compute a strategy profile that optimizes the agents' objective functions, given the strategies of the other agent. Given the computed strategies, we implement them using a control switching framework, in which our controller switches from u_H to u_Υ at each step.

This approach does not attempt to incorporate stochastic adaptation to or learning of the other agent's objective function or dynamical model. It contrasts with the Bounded-Memory Adaptation Model of [12], which uses a parameter α to capture an agent's inclination to adapt. For example, if human and AS disagree, the human may switch from their control u_H to the AS's control u_Υ at the next time step with probability α.

We assume that both human and AS agents minimize their respective objective function. Then, a solution of the differential game requires solving a multi-objective optimization problem. We adopt a receding horizon optimization approach, i.e., the optimal controls are calculated by solving the open-loop optimal control problems for the horizon $[t, t+k]$.

B. Blended Control

Blended control enables the AS and human to simultaneously compute controls (u_Υ and u_H) and then "integrates" them in the supervisory control module in an optimal manner. Figure 1(b) depicts the architecture for blended autonomy.

We adopt the widely-used *weighted-sum method* for optimization [8], since solving a multi-objective function $\boldsymbol{\mathcal{J}}_w = \varsigma(\mathcal{J}_1, ..., \mathcal{J}_n)$ is computationally intractable. We need to represent vectors for the collection of n agents, so we denote $\boldsymbol{\Phi} = [\Phi_1, \cdots, \Phi_n]^T$, and $\boldsymbol{\mathcal{J}}(u, \alpha) = [\mathcal{J}_1(u_1, \alpha_1), \cdots, \mathcal{J}_n(u_n, \alpha_n)]^T$. We map the multiple objective functions into a single weighted function, i.e., assigning a non-negative weight λ_i to each of the i objective functions, given by $\mathcal{J} = \boldsymbol{\lambda}^T \boldsymbol{\mathcal{J}}(u, \Phi)$, where $\boldsymbol{\lambda}^T = [\lambda_1, \cdots, \lambda_n]^T$ is the weight vector. The objective is to optimize $\mathcal{J}$ by selecting weights such that $\sum_i \lambda_i = 1$ and $1 \geq \lambda_i \geq 0, \;\; i = 1, \cdots, n$. The optimal solution is given by $u^* = \arg\min_u \boldsymbol{\lambda}^T \boldsymbol{\mathcal{J}}(u, \Phi)$.

In general, we need to normalize the objective functions since not all objectives have the same range of values; after normalization we use standard optimization algorithms to compute optimal solutions.

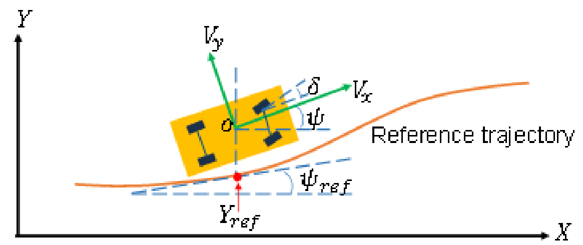

Fig. 2. Bicycle Model of a car

x, y	Position of the center of gravity (CoG) of the vehicle in the ground framework in (x, y)-plane
ψ, δ	Yaw and steering angle of the car body
v_x, v_y	Longitudinal and lateral speed of the vehicle in its inertial frame
M	Total mass of the vehicle
l_f, l_r	Distance from front and rear axle to CoG
$\beta(\delta)$	Slip angle at the CoG
C_f, C_r	Front and rear cornering stiffness
I_z	Polar moment of intertia
i_s	Steering Ratio

Fig. 3. Notation used in the article

C. Switching Control

Switching control is a framework for integrating control inputs from multiple agents by switching among the agents' inputs using a switching protocol. In this article we use the world model ξ_w as the basis for switching. We assume that an oracle Θ has perfect knowledge of ξ_w, and the protocol switches control to the agent whose output is closest to that of the oracle Θ.

VI. Bicycle Model

We conducted experiments on car steering control using the bicycle model [13], whose lateral vehicle dynamics have two degrees of freedom, lateral position and yaw angle. This section describes the vehicle model (as depicted in Figure 2), using notation defined in Figure 3.

The lateral velocity of the vehicle v_x is constant and hence the *control input* corresponds to the front wheel steering angle δ of the vehicle, under the assumption that only the front wheels can be steered.

Definition 6 (Kinematic Bicycle Model): We define the kinematic bicycle model as [13]:

$$\begin{aligned}\dot{x} &= vcos(\psi+\beta(\delta))\\ \dot{y} &= vsin(\psi+\beta(\delta))\\ \dot{\psi} &= \frac{v}{l_r}sin(\beta(\delta))\\ \beta(\delta) &= tan^{-1}\left[tan(\delta)\frac{l_r}{l_f+l_r}\right]\end{aligned}$$

Since this vehicle model has more degrees of freedom than control inputs its classified as under-actuated.

Definition 7 (Vehicle Model): The dynamics of the longitudinal, lateral and yaw motions of the whole vehicle are given in the form of state space equation 2, where $\xi(k) =$

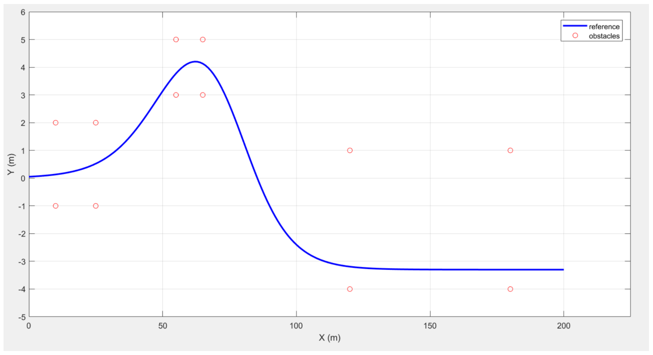

Fig. 4. Double Lane change reference trajectory

$[v(k) \quad \omega(k) \quad y(k) \quad \psi(k)]^T$, output $z(k) = [y(k) \quad \psi(k)]^T$, and A, B and C are given as follows:

$$A = \begin{bmatrix} \frac{-(C_f+C_r)}{Mv_x} & \frac{-(l_fC_f-l_rC_r)}{Mv_x}-v_x & 0 & 0\\ \frac{-(l_fC_f-l_rC_r)}{I_zv_x} & \frac{-(l_f^2C_f+l_r^2C_r)}{I_zv_x} & 0 & 0\\ 1 & 0 & 0 & v_x\\ 0 & 1 & 0 & 0\end{bmatrix}$$

$$B = \begin{bmatrix}\frac{C_f}{i_sM} & \frac{l_fC_f}{i_sI_z} & 0 & 0\end{bmatrix}^T, \quad C = \begin{bmatrix}0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$

We are interested in controlling the and velocity v of the vehicle, which is given by equation 3:

$$\begin{aligned}\dot{x} &= v_xcos\psi - v_ysin\psi\\ \dot{y} &= v_xsin\psi + v_ycos\psi \qquad (3)\end{aligned}$$

Definition 8 (Control Problem): Given a list of waypoints $\mathcal{W} = (w_i)_{i\in I} = (x_i, y_i, v_i)_{i\in I}$, where (x_i, y_i) are the successive reference positions of the vehicle in the ground frame and v_i the successive speed references at position (x_i, y_i), the control problem consists in finding a sequence of feasible control inputs δ that optimizes an objective function $\mathcal{J}$, subject to constraints imposed by $\mathcal{W}$.

Figure 4 shows a double lane change manoeuvre that we use as the reference trajectory for the experiments. We define $\mathcal{J}$ based on errors in lateral displacement and yaw tracking.

VII. Experimental Analysis

For our experiments we use the bicycle steering model described previously. We implemented the Leader-Follower, Oracle-Based Switching and Blended control framework for this vehicle on top of a Matlab Autonomous Steering demo. To simulate the interaction of control for the described framework in previous sections between an AS and a Human we define two Model Predictive Controllers (MPC) for u_H and u_Υ, respectively. with weight assignment for the measured variables ψ and Y being set to [1,1] for u_H and [1,0.1] for u_Υ. Intuitively the Human controller focuses more on reference tracking between ψ_{ref} and ψ instead of mostly on the y position tracking, which should give smoother tracking overall. We minimize the total tracking error $\epsilon = \sum_{i\in\mathcal{W}}|y_{ref}(k)-y(k)|$ over displacement waypoints $\mathcal{W}$. We empirically study the impact of the following across the control approaches:

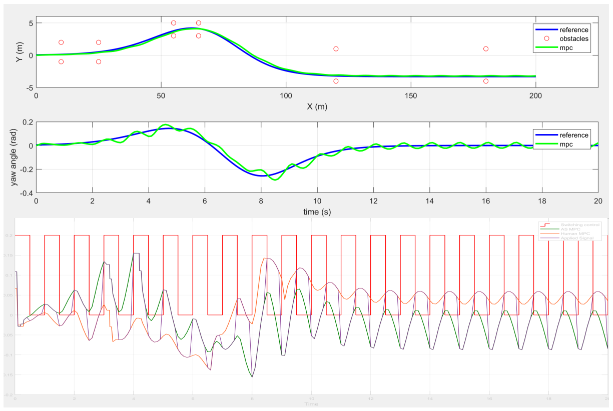

Fig. 5. Leader-Follower framework in nominal operating conditions

- Bad u_H signals, such as a collision course or no input;
- Signal delays;
- Sporadic noise on the dominant control signals.

A. *Implemented Control Models*

We now describe the implemented control models.

1) Leader-Follower: The Leader-follower (LF) framework alternates the control signals applied to the car model at intervals of $\delta = 1s$. Figure 5 shows the LF framework executing the double lane change manoeuvre in nominal operating conditions. The figure is broken into 3 sub-charts each showing different metrics during the simulation. The top consists of the (x, y) position of the reference path (blue), actual path taken (green) and obstacles to avoid (red circles). In the middle chart the yaw angle of the car can be seen, again reference position (blue) and actual position (green). The last chart consists of the MPC output signal u_H (orange) and u_Υ (green), the applied control signal u (purple) and η (red) which signifies the current control proportions between Human controller and AS used to calculate u. The X-axis represents time and the Y-axis the magnitude of the signal.

2) Blending: Blending combines the u_H and u_Υ signals using weight parameter η such that: $u(k) = u_H(k)(1-\lambda) + u_\Upsilon(k)(\lambda)$, with $0 \leq \lambda \leq 1$. We predict the tracking error ϵ between y and y_{ref} for the current time point k. Since by nature the current tracking error ϵ will gradually increase, we set $\lambda = \epsilon$ so that if control starts to diverge, u_Υ will gradually become more influential and should re-stabilize control. Since u_H plays a more dominant part in the control signal u we call this **Human Dominant Blending**. Alternatively we also experimented with **AS Dominant Blending** which is defined by $u(k) = u_H(k)(\lambda) + u_\Upsilon(k)(1-\lambda)$.

3) Oracle-based Switching: We implemented Oracle-based switching (OBS) such that the oracle Θ has knowledge of the reference path Θ_{ref}, obstacles Θ_{obs} as well as the car's current trajectory Θ_σ, which is defined by fitting a polynomial function of degree 4 to the last 5 (x, y) coordinates. Evaluating $\Theta_\sigma(t+\tau)$ allows us to predict the path the vehicle will take during the next τ seconds.

We define our control $u(k)$ such that the human control is applied ($u = u_H$) if (a) the predicted displacement error is within 0.2m or (b) no collisions are predicted; otherwise $u = u_\Upsilon$. Formally, $u(k)$ is generated such that

$$u(k) = \begin{cases} u_H, & \text{if } (div \geq \beta) \;\vee\; (col \leq \beta) \\ u_\Upsilon & \text{otherwise} \end{cases}$$

where we define $div = |\Theta_\sigma(t+\alpha) - Y_{ref}(t+\alpha)|$ as the expected future divergence to Y_{ref}, and $col = |\Theta_{obs_i}(y) - \Theta_\sigma(t+\alpha)(y)| \quad \forall \Theta_{obs_i} \in \Theta_{obs}$ as the distance to an obstacle. $\beta = 0.2$ and $\alpha = [0.1, 0.2, 0.3]$.

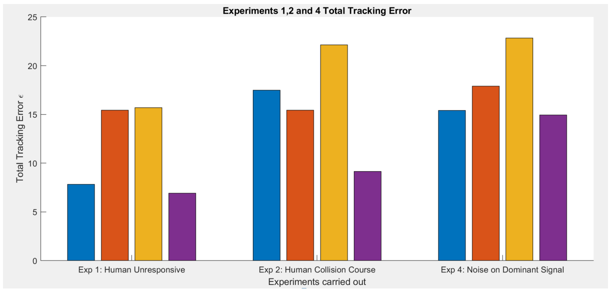

Fig. 6. Total tracking error for double lane change for Experiments 1,2 and 4. Blue: Leader Follower, Orange: Oracle-based Switching , Yellow: Human Dominant Blending, Purple: AS Dominant Blending

B. *Experiments*

1) Experiment 1: No Human input: We set $u_H = 0$ to model a Human that has fallen asleep or is otherwise unable to issue control commands. Figure 6 (left) shows the difference in total tracking error between the frameworks for the described double lane change manoeuvre. Surprisingly the OBS and Human Dominant Blending perform very poorly. We assume this is due to the system being allowed to get into a bad state, from which it tries to recover; in contrast, the Leader-Follower and AS-Dominant Blending approaches always assign a degree of control to the AS, such that even if $u_H = 0$ the AS-Dominant Blending and Leader-Follower are only affected slightly. However, all frameworks were able to re-stabilize control and execute the manoeuvre without crashing into any obstacles.

2) Experiment 2: Human on Collision Course: We next investigated the effect of bad u_H. The Human MPC controller was modified to have a slower sampling rate, which gives it worse tracking and causes a collision during the manoeuvre. This experiment is designed to test how the frameworks respond to bad human input. Again, Figure 6 (middle) shows ϵ for the four frameworks. Again all frameworks were able to avoid collision and complete the manoeuvre. The LF framework experienced a thrashing yaw angle signal, but the y position tracking is good. Human Dominant Blending and OBS trashed less but overall tracking was worse. AS Dominant Blending performed very smoothly for both signals and achieved best control. This is because the Human agent has little or no control of the system, so being on a collision course does not affect the overall system. Given results from Experiments 1 and 2, we can conclude that AS Dominant blending performs the best if we cannot guarantee a good u_H input.

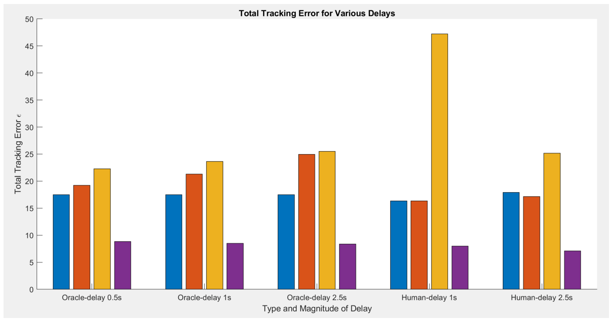

Fig. 7. Total tracking error with oracle and human delays.

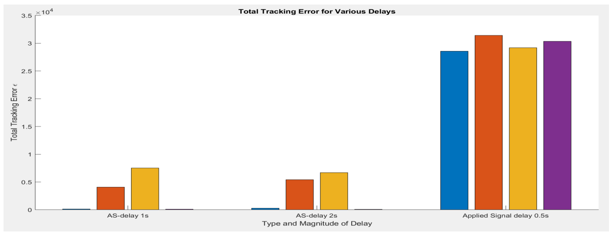

Fig. 8. Total tracking error with AS and applied signal delays.

3) Experiment 3:Delays on various signals: We studied the effect of delays in the following signals:

1) Oracle computation of Θ for $\delta = [0.5, 1, 2]$.
2) AS control signal u_Υ for $\delta = [1, 2]$.
3) Human control signal u_H for $\delta = [1, 2]$.
4) Overall applied system u for $\delta = [0.5]$.

Here, δ represents the delay on the specified signal. Due to limited space we show only the overall tracking error ϵ for the specified delays. Figure 7 correspond to oracle and human delays, Figure 8 to the AS and Applied signal delays. Delays can represent a variety of factors such as computation time for AS and Oracle and delayed human response and are common in real systems. Hence evaluating the resilience to delays for the frameworks is of utmost importance.

4) Experiment 4: Noise Rejection: Our final experiment concerns the effects of sporadic noise on the dominant controller. For this we injected a single offset of 0.5 at $t = 3s$ for a duration of 1 second into the dominant control signal. We evaluate each approach in detail and show the paths taken. Figure 9 shows the tracking results for the LF framework.

The deviation due to noise injected is clearly visible in the Y position tracking (top). The time taken to re-stabilize on the path is about 2 seconds. This is due to the fact the LF framework periodically switches the actual signals, seen in the bottom plot, which can be analysed to find that the human had just started its control period when the noise was injected. This means the full effect of the noise were experienced for 1 seconds before the AS could intervene and we see a large deviation. We also notice slight trashing for the remainder of the simulation Y position but more significantly in the steering angle.

Figure 10 shows the Oracle-based Switching simulation results. The deviation is a lot smaller since this framework

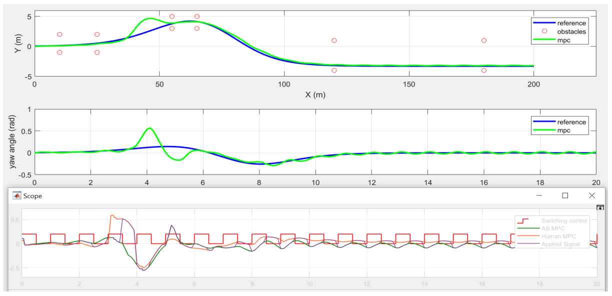

Fig. 9. Leader Follower Total tracking error for double lane change with noise injected to u_H at t=3s.

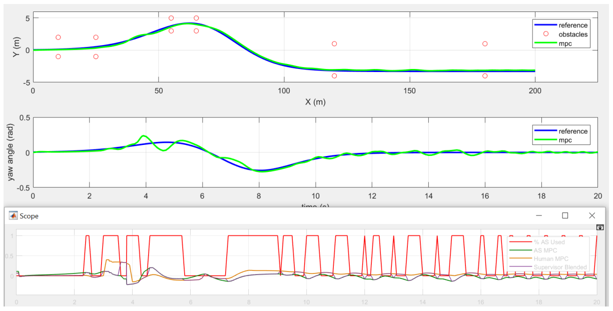

Fig. 10. Oracle-based Switching Total tracking error for double lane change with noise injected to u_H at t=3s.

identifies the bad trajectory and switches controllers. This is again obvious from the bottom chart in this Figure. We see the red line representing the currently controlling agent switch to the AS slightly before the u_H noise was injected. We assume this was before the path deviated before the noise and hence the AS agent took control and the effects of the noise were limited. However, we still see slight thrashing in the steering angle for the remainder of the simulation.

Figure 11 describes the Human Dominant Blending under noisy conditions. The deviation is significant but the framework is able to stabilize and execute the manoeuvre. We notice the framework started using the AS before the injection of noise which is why the effects were not as severe as for LF. Note here that there is no thrashing on the yaw angle and both tracking objectives are smooth.

This is the final framework we test again this scenario. One difference to the other Noise injection experiments

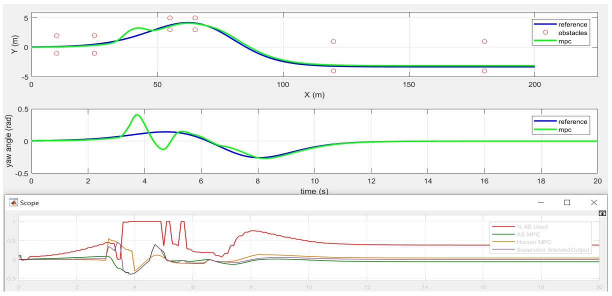

Fig. 11. Human Dominant Blending Total tracking error for double lane change with noise injected to u_H at t=3s.

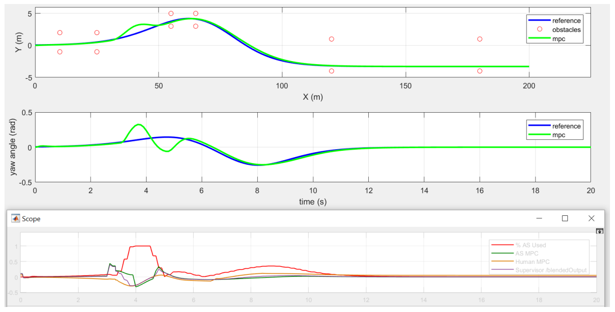

Fig. 12. AS Dominant Blending Total tracking error for double lane change with noise injected to u_{AS} at t=3s.

is that we applied the noise to the AS Signal instead of Human signal, since having it on the Human will have no effect. Even with that change this framework performs the best overall. It is very smooth in the lead up to the noise and then recovers quickly. The deviation is still significant, however. The bottom chart shows that the blend of signals, in contrast to all other frameworks, is exceptionally smooth and it resumes close tracking within 3 seconds.

Finally, Figure 6 (right) shows the comparison of total tracking errors for this experiment. Even though the LF framework experienced a large deviation and trashing, it still tracks accurately, since it assigns 100% of the control to a single controller, allowing that controller to get very good tracking for that time. It is difficult for Blending to achieve this level of tracking once the agents signals start to diverge, since the u will always be somewhere *between* u_H and u_{AS}, but it's rare that one agent has full control to generate an unmodified (optimal) control signal. This is an advantage the Switching frameworks have over the Blending. In contrast, this same phenomenon causes trashing on the output as both controllers compete for control of the system.

C. Discussion

We summarize our results as follows:

- *Bad u_H signals:* Control performance improves given less reliance of the framework on u_H. This is obvious from Experiments 1 and 2, which show AS blending performing the best.
- *Signal delays:* Signal delays can affect performance dramatically. AS Dominant Blending is the most resilient to delays on all signals expect when u gets delayed. But performance from all frameworks is catastrophic in this scenario.
- *Sporadic noise:*. Noise causes control thrashing, and consequently affects steering performance. AS Dominant Blending once again performed the best, even though marginally, and is far smoother than the switching frameworks in regards to yaw control.

VIII. Conclusions

This article has studied the impact on collaborative control of collaboration types (χ), agent/world models (Φ), and environmental disturbance (noise and control delays).

- *collaboration types* were studied using three control techniques: leader-follower, blended and oracle-based switching.
- *agent models* included complete models, models with delayed response, and empty models (no human response).
- *environmental disturbance* included noise on control signals and communication delays.

We empirically studied collaborative control using the domain of car steering control, with a double lane-change task. Our objective function measured the lateral displacement errors and yaw errors relative to a reference trajectory. Our results indicate that, given an accurate AS controller, steering control is optimized when the AS has primary control: this minimizes the effect of human errors and delays in control actions being implemented.

References

[1] F. Altché, X. Qian, and A. de La Fortelle. An algorithm for supervised driving of cooperative semi-autonomous vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[2] L. Chan, F. Naghdy, and D. Stirling. Application of adaptive controllers in teleoperation systems: A survey. *IEEE Transactions on Human-Machine Systems*, 44(3):337–352, 2014.

[3] A. D. Dragan. Robot planning with mathematical models of human state and action. *arXiv preprint arXiv:1705.04226*, 2017.

[4] M. Flad, L. Fröhlich, and S. Hohmann. Cooperative shared control driver assistance systems based on motion primitives and differential games. *IEEE Transactions on Human-Machine Systems*, 2017.

[5] A. Gambier. Mpc and pid control based on multi-objective optimization. In *American Control Conference, 2008*, pages 4727–4732. IEEE, 2008.

[6] S. Gray, R. Chevalier, B. Caimano, and J. Scatena. Graduated automation for humanoid manipulation. In *Automation Science and Engineering (CASE), 2016 IEEE International Conference on*, pages 1366–1373. IEEE, 2016.

[7] H. Kim, J. Cho, D. Kim, and K. Huh. Intervention minimized semi-autonomous control using decoupled model predictive control. In *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pages 618–623. IEEE, 2017.

[8] R. T. Marler and J. S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395, 2004.

[9] X. Na and D. J. Cole. Application of open-loop stackelberg equilibrium to modeling a driver's interaction with vehicle active steering control in obstacle avoidance. *IEEE Transactions on Human-Machine Systems*, 2017.

[10] A.-T. Nguyen, C. Sentouh, and J.-C. Popieul. Driver-automation cooperative approach for shared steering control under multiple system constraints: Design and experiments. *IEEE Transactions on Industrial Electronics*, 64(5):3819–3830, 2017.

[11] S. Nikolaidis, J. Forlizzi, D. Hsu, J. Shah, and S. Srinivasa. Mathematical models of adaptation in human-robot collaboration. *arXiv preprint arXiv:1707.02586*, 2017.

[12] S. Nikolaidis, D. Hsu, and S. Srinivasa. Human-robot mutual adaptation in collaborative tasks: Models and experiments. *The International Journal of Robotics Research*, page 0278364917690593.

[13] R. Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[14] D. Richards and A. Stedmon. To delegate or not to delegate: A review of control frameworks for autonomous cars. *Applied ergonomics*, 53:383–388, 2016.

[15] L. Song, H. Guo, F. Wang, J. Liu, and H. Chen. Model predictive control oriented shared steering control for intelligent vehicles. In *Control And Decision Conference (CCDC), 2017 29th Chinese*, pages 7568–7573. IEEE, 2017.