Joint 48th IEEE Conference on Decision and Control and
28th Chinese Control Conference
Shanghai, P.R. China, December 16-18, 2009

Reachability analysis for continuous systems under shared control: Application to user-interface design

Nikolai Matni, Meeko Oishi

Abstract— **We extend techniques for a reachability-based abstraction to continuous systems under *shared control*, that is, systems which have both inputs controlled by the automation and inputs controlled by the human, to account for potential interactions between the human and the automation that affect safety. We broadly classify human input as assisting the automated input, neutral, or fighting against the automated input, resulting in three types of invariance. Using standard reachability tools to calculate invariant, user-invariant, and user-assisted-invariant sets, regions in the state-space are associated with three levels of safety: 1) safe, 2) marginally safe, and 3) recoverably safe. By partitioning the state-space according to intersections of the invariant sets, we create an abstraction to a discrete event system of minimal cardinality which can inform the information content of a discrete user-interface that preserves information about the safety levels of the system. We apply the reachable set calculation and abstraction method to an aircraft landing under shared control.**

Index Terms— **reachability analysis, nonlinear systems, mixed-initiative, user-interface, discrete event systems, shared control, flight management systems, invariance.**

I. Introduction

Computational techniques for verification can create a new level of confidence and reliability in safety-critical systems, such as aircraft autopilots, by predicting where failures might occur, and how human operators can predict them [1], [2], [3], [4]. However, when the human can affect the system's behavior (e.g. by selecting a mode of operation, providing a reference input), and consequently, its safety, verification techniques must be adapted to explicitly incorporate human-automation interaction. This paper focuses on applying Hamilton-Jacobi techniques [5], [6] for reachability analysis and controller synthesis to verification of continuous dynamical systems with continuous human input. The Level Set Toolbox [7] is used because of its sub grid accuracy and success in previous aircraft applications, although other techniques [8], [9], [10], [11], [12] could be implemented.

While verification techniques have been successfully applied to human-automation systems modeled as discrete event systems (DES) [13], [14], [15], [2], [16], [17], less work has been done on verification of continuous or hybrid human-automation systems [18], [1], [19]. In [20], an invariance-preserving abstraction was formulated for supervisory hybrid systems: that is, the human input was limited to discrete inputs. Human input was incorporated into the reachability analysis by simply isolating parts of the system which were autonomous, and then applying standard reachability tools to the autonomous subsystems. However, this strategy is not feasible for systems with continuous human input. In [21], we considered the effect of continuous human input on safety for a specific system. This paper aims to generalize that result to continuous time systems with continuous human input and continuous controlled input.

In [21], we considered an incident that occurred during an attempted manual landing of a passenger aircraft [22]. The flight crew accidentally switched the aircraft into a semi-automatic mode, that, combined with the pilot's continuous input, led to an unsafe aircraft configuration. Unbeknownst to the pilots, this configuration imposed restrictions on the pilots' input: when they attempted to manually abort the landing by applying a pitch-up command, the aircraft climbed too quickly and stalled. Based on previous work [20], we calculated multiple reachable sets, by 1) treating the pilot input as a disturbance input, 2) assuming the pilot was "hands off" the control, and 3) treating the pilot input as a controlled input. The reachable sets were used to inform the design of a discrete user-interface, through the creation of a DES of minimal cardinality.

In this paper, we formalize the approach in [21] for generic nonlinear continuous systems under both automation-controlled and human-controlled input: *shared control systems*. We create multiple levels of safety by imposing different bounds on the human's control authority, and relate those levels of safety to invariant sets computed using standard reachability tools. By choosing increasingly looser assumptions about the human's behavior, we create three broad classes of safety levels: *safe*, *marginally safe*, and *recoverably safe*. Each level of safety places different restrictions on human input to preserve system safety. Hence, user-interface content must be designed through a particular reachability-based abstraction to be "safety-informative", e.g., to succinctly provide information required by the user to ascertain the effect of their actions on system safety. Our main contributions are 1) formal definitions of invariance, user-invariance, and user-assisted invariance for shared control systems, and their relationship to computed reachable sets, and 2) a method to abstract the resulting reachable sets to a DES that contains relevant information regarding the effect of continuous human input on safety.

The paper is organized as follows: In Section II, we first

Research supported by NSERC Discovery Grant #327387, CFI Leaders Opportunity Fund #13113, and by an NSERC USRA grant.

N. Matni is a graduate student with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, Canada nmatni@ece.ubc.ca.

M. Oishi (corresponding author) is an Assistant Professor with the Department of Electrical and Computer Engineering, Vancouver, BC, Canada moishi@ece.ubc.ca.

model a generic continuous system under shared control, and introduce a simple example to motivate the effect of human input on safety. In Section III, standard reachability tools are applied to compute reachable sets of differing levels of safety. Section IV presents a general abstraction method for multiple safety levels. This method is applied to an aircraft landing under shared control [21] in Section V to create a *safety-informative* discrete user-interface. Section VI provides conclusions and directions for future work.

II. Safety in Human-Automation Systems under Shared Control

A. Modeling

Consider a continuous system under shared control

$$\dot{x} = f(x, u_c, u_h) \tag{1}$$

with states $x \in \mathcal{X} \subseteq \mathbb{R}^n$, automation-controlled continuous input $u_c \in \mathcal{U}_c = [\underline{u}_c, \overline{u}_c]$, human-controlled continuous input $u_h \in \mathcal{U}_h = [\underline{u}_h, \overline{u}_h]$, with $\underline{u}_h < 0, \overline{u}_h > 0$. We assume that the automation input $u_c = u_c(x)$ is strictly a function of the state, whereas the human input $u_h = u_h(r)$ is a function of a human-controlled reference input $r \in \mathcal{R} = [r_{\min}, r_{\max}]$.

Example 1: Consider the double integrator system

$$\begin{aligned} \dot{x} &= Ax + B(u_c(x) + u_h(r)) \\ A &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 1 \end{bmatrix}^T \\ u_c(x) &= -3 \cdot \mathrm{sign}(x_2), \quad u_h(r) = r \end{aligned} \tag{2}$$

with state $x \in \mathcal{X} \subseteq \mathbb{R}^2$, automatic control input $u_c \in [-3, 3]$, human control input $u_h \in [-3, 3]$ and constraint set $\mathcal{C} = [-5, 5] \times [-5, 5]$.

Consider trajectories starting from $x(0) = [4, 3]^T$ under different human inputs (Figure 1). 1) The human input $r = -\frac{2}{3}u_c(x)$ drives the state out of the constraint set ($\diamond$), effectively acting as a disturbance, leading to safety failure. 2) The human is "hands off" the controls ($r = 0$), but the resulting trajectory ($\circ$) also exits the constraint set $\mathcal{C}$. If system safety is to be preserved the human *must* assist the automation. 3) The human input $r = u_c(x)$ co-operates with the automation input to preserve system safety. The resulting trajectory ($\triangle$) remains within the constraint set $\mathcal{C}$. Hence for the specific value $x(0)$ chosen here, safety is maintained only when the human co-operates with the automation.

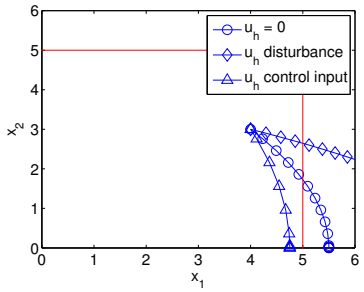

Fig. 1. Trajectories for Example 1 starting from $x(0) = [4, 3]^T$ for which: 1) ($\diamond$) the user acts as a disturbance 2) ($\circ$) the user is "hands-off" and 3) ($\triangle$) the user acts as a control input. The constraint set $\mathcal{C}$ is drawn with a solid red line.

B. Invariance under Shared Control

To perform reachability analysis on a system under shared control, we take into account how interactions between the human input and the automation input effect system safety. Our approach is to broadly classify the human's input as: 1) a disturbance, driving the system to unsafety, 2) neutral ("hands-off"), implying $u_h(r) = 0$, or 3) a controlled input, assisting the automation in preserving safety. Consider the following three types of invariant sets.

Definition 1: For a set $\mathcal{W}_I \subseteq \mathcal{X}$ to be *invariant* with respect to a constraint set $\mathcal{C}$, all trajectories $x(t)$ which start in $\mathcal{W}_I$ must remain within $\mathcal{C}$ for all $t \geq 0$ for all continuous human input $u_h \in \mathcal{U}_h$.

$$\begin{aligned} \mathcal{W}_I = \{x(0) \in \mathcal{C} \mid \forall u_h \in \mathcal{U}_h \; \exists u_c \in \mathcal{U}_c \\ \text{such that } x(t) \in \mathcal{C} \; \forall t \geq 0\} \end{aligned} \tag{3}$$

Definition 2: For a set $\mathcal{W}_{UI} \subseteq \mathcal{X}$ to be *user-invariant* with respect to a constraint set $\mathcal{C}$, all trajectories $x(t)$ which start in $\mathcal{W}_{UI}$ must remain within $\mathcal{C}$ for all $t \geq 0$ for all $u_h \in \mathcal{U}_{UI} \subseteq \mathcal{U}_h$.

$$\begin{aligned} \mathcal{W}_{UI} = \{x(0) \in \mathcal{C} \mid \forall u_h \in \mathcal{U}_{UI} \; \exists u_c \in \mathcal{U}_c \\ \text{such that } x(t) \in \mathcal{C} \; \forall t \geq 0\} \end{aligned} \tag{4}$$

Definition 3: For a set $\mathcal{W}_{UAI} \subseteq \mathcal{X}$ to be *user-assisted-invariant* with respect to a constraint set $\mathcal{C}$, there must exist a control input pair $(u_h, u_c) \in \mathcal{U}_{UAI} \times \mathcal{U}_c$ such that all trajectories $x(t)$ which start in $\mathcal{W}_{UI}$ will remain within $\mathcal{C}$ for all $t \geq 0$. Here, $\mathcal{U}_{UAI} \subseteq \mathcal{U}_h$.

$$\begin{aligned} \mathcal{W}_{UAI} = \{x(0) \in \mathcal{C} \mid \exists (u_h, u_c) \in \mathcal{U}_{UAI} \times \mathcal{U}_c \\ \text{such that } x(t) \in \mathcal{C} \quad \forall t \geq 0\} \end{aligned} \tag{5}$$

Invariant sets are computed by effectively treating the human input as a disturbance input. Often, this very conservative assumption leads to $\mathcal{W}_I = \{\emptyset\}$. In many systems, treating the operator as a disturbance is not realistic or necessary. By bounding the control authority given to the user when they are acting as a disturbance, a less conservative and possibly more useful result can be obtained.

User-invariant sets are effectively computed by ignoring the human input (assuming $u_h = 0$), hence some human inputs (outside the allowable range) may cause the state to exit the constraint set. The guarantee of safety is weaker than for invariant sets.

A user-assisted-invariant set represents the portion of the state space in which it is possible for the human to apply a prescribed input which maintains system safety.

The important distinction between user-invariant and user-assisted-invariant sets is that there are portions of user-assisted-invariant sets in which the human *must* apply an input to preserve system safety, as the automation is unable to prevent failure on its own. By contrast, in a user-invariant set, the human *may* apply an input to assist the automation to keep the system safe, but does not have to. In user-invariant sets, bounds on the human input can be interpreted as a recommendation – remaining within these bounds guarantees

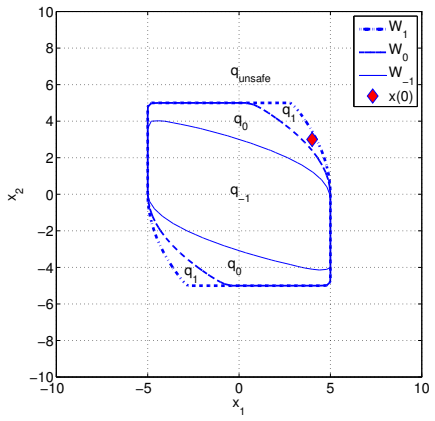

Fig. 2. The safe, marginally safe and recoverably safe sets $\mathcal{W}_{-1}$, $\mathcal{W}_0$ and $\mathcal{W}_1$ (Example 1), computed by treating the user as a disturbance, "hands off" and as a controlled input, respectively.

safety, but exceeding them will not cause failure. In user-assisted-invariant sets, the constraints are much stricter – an input must be applied to preserve system safety, and failing to do so will eventually lead to a violation of the safety constraints. The relationship between these sets will be described in Section III.

C. Using Invariant Sets to Create a User-Interface

The algorithm in [20] for user-interface design for supervisory hybrid systems to preserve system safety involves three steps: 1) separation of the hybrid system into subsystems which contain no human-initiated discrete inputs, 2) calculation of the reachable set for each subsystem, and 3) abstraction to a discrete event system based on the reachability result. The reachability result partitions the state-space into intersections of "safe" or "unsafe" regions in each subsystem. *Our aim is to abstract (1) to a discrete event system that conveys the safety information of multiple invariant, user-invariant and user-assisted-invariant sets, $\mathcal{W}_i,\ i \in \{1, ..., n\}$, to the user. Having this information allows the user to determine if the current state is in an invariant, user-invariant or user-assisted-invariant subset of the state space, and consequently, whether or not there are safety restrictions on the human input.* To accomplish this, 1) compute the invariant, user-invariant and user-assisted-invariant sets of (1) with respect to the constraint set $\mathcal{C}$, and 2) abstract the computed invariant, user-invariant and user-assisted-invariant sets to a DES. This DES conveys the safety information contained in these sets to the user.

III. Calculating Reachable Sets

Computing the reachable set involves representing all of the states which have a path to a target set. As in [6], for $\dot{x} = f(x, u, d)$, with control input $u \in \mathcal{U}$, disturbance input $d \in \mathcal{D}$, and constraint set $\mathcal{C}$, the "target" is encoded implicitly as a level set function $\overline{\mathcal{W}}_0 = \mathcal{C}^c = \{x \in \mathcal{X} \mid J_0(x) < 0\}$, $J_0 : \mathcal{X} \to \mathbb{R}$. The boundary of the target set is propagated backwards in time according to the system dynamics. Finding the backwards reachable set $\overline{\mathcal{W}}(t)$ requires solving the terminal value time-dependent modified Hamilton-Jacobi partial differential equation

$$\begin{aligned} 0 &= \frac{\partial J(x,t)}{\partial t} + \min\left[0, H\left(x, \frac{\partial J(x,t)}{\partial x}\right)\right] \\ H\left(x, \frac{\partial J(x,t)}{\partial x}\right) &= \max_{u\in\mathcal{U}} \min_{d\in\mathcal{D}} \frac{\partial J(x,t)}{\partial x}^T f(x,u,d) \end{aligned} \tag{6}$$

with $J(x, 0) = J_0(x)$ for $t = 0$ such that the invariant set is $\mathcal{W}(t) = \{x \in \mathcal{X} \mid J(x, t) \geq 0\}$.

Although the user typically acts to preserve system safety, it is extremely difficult and often non-generalizable to explicitly model a user's control actions. Instead, we compute an arbitrary number of reachable sets that encompass the full range of possible user behaviors. Define the set $\mathcal{U}_i \subseteq \mathcal{U}_h$ as a reduced set of inputs

$$\mathcal{U}_i = \alpha_i \mathcal{U}_h, \alpha_i \in [0, 1],\ i \in \{-N, ..., 0, ...M\} \tag{7}$$

where N and M are the arbitrarily chosen number of safe sets and recoverably safe sets, respectively.

A. Safe Sets

Let $i = -N, ..., -1$, with N the number of *safe sets* $\mathcal{W}_i$ to be calculated by solving (6) with the Hamiltonian

$$H_i\left(x, \frac{\partial J(x,t)}{\partial x}\right) = \max_{u_c\in\mathcal{U}_c} \min_{u_h\in\mathcal{U}_i} \frac{\partial J(x,t)}{\partial x}^T f(x, u_c, u_h) \tag{8}$$

and $\mathcal{U}_i = \alpha_i \mathcal{U}_h$, $\alpha_i \in (0, 1]$ $\alpha_{i+1} < \alpha_i$ such that $\mathcal{U}_{i+1} \subset \mathcal{U}_i$. Note that the following property holds [23]:

$$\mathcal{W}_{-N} \subset \mathcal{W}_{-N+1} \subset \ldots \subset \mathcal{W}_{-1} \tag{9}$$

The sets $\mathcal{W}_i,\ i \in \{-N, ..., -1\}$ are "safe" because they represent portions of the state-space in which the user can apply any input $u_h \in \mathcal{U}_i$ without violating the constraints for safety. The invariance preserving control law is *not* enforced along the boundaries of the sets, allowing the user to transition between sets by choosing inputs $u_h \notin \mathcal{U}_i$.

Example 1: The safe set $\mathcal{W}_{-1}$, calculated with

$$\begin{aligned} H_{-1}\left(x, \frac{\partial J(x,t)}{\partial x}\right) &= \max_{u_c\in\mathcal{U}_c} \min_{u_h\in\mathcal{U}_{-1}} \frac{\partial J(x,t)}{\partial x}^T f(x, u_c, u_h) \\ &= \frac{\partial J(x,t)}{\partial x_1} x_2 + \left|\frac{\partial J(x,t)}{\partial x_2}\right| \end{aligned} \tag{10}$$

and $\mathcal{U}_{-1} = \frac{2}{3}\mathcal{U}_h$ is shown in Figure 2. As expected, the initial condition $x(0) = [4, 3]^T$, lies outside of $\mathcal{W}_{-1}$.

B. Marginally Safe Sets

Let $i = 0$, and $\mathcal{U}_0 = 0$ to calculate the *marginally safe set* $\mathcal{W}_0$ by solving (6) with Hamiltonian

$$H_0\left(x, \frac{\partial J(x,t)}{\partial x}\right) = \max_{u_c\in\mathcal{U}_c} \frac{\partial J(x,t)}{\partial x}^T f(x, u_c, 0) \tag{11}$$

The set $\mathcal{W}_0$ is "marginally safe" because it represents the portion of the state-space in which the automation is capable of maintaining system safety without user interference or assistance. As long as the user remains neutral, or "hands-off" the controls, safety is guaranteed.

Example 1: $\mathcal{W}_0$ (shown in Figure 2) is calculated with

$$\begin{aligned} H_0\left(x, \tfrac{\partial J(x,t)}{\partial x}\right) &= \max_{u_c \in \mathcal{U}_c} \tfrac{\partial J(x,t)}{\partial x}^T f(x, u_c, 0) \\ &= \tfrac{\partial J(x,t)}{\partial x_1} x_2 + 3|\tfrac{\partial J(x,t)}{\partial x_2}| \end{aligned} \quad (12)$$

As expected, $x(0) = [4,3]^T \notin \mathcal{W}_0$.

Lemma 1: Safe sets and marginally safe sets are user-invariant.

Proof: By construction: For $\mathcal{W}_i,\ i \in \{-N, ..., -1\}$ computed with $u_h \in \mathcal{U}_i \subseteq \mathcal{U}_h$, for all $x(0) \in \mathcal{W}_i$, $x(t) \in C$ for all $t \geq 0$ as long as $u_h \in \mathcal{U}_i$. Thus $\mathcal{W}_i,\ i \in \{-N, ..., -1\}$ are user-invariant by definition. Similarly, for $\mathcal{W}_0$ computed with $u_h \in \mathcal{U}_0 = 0 \subset \mathcal{U}_h$, for all $x(0) \in \mathcal{W}_i$, $x(t) \in C$ for all $t \geq 0$ as long as $u_h \in \mathcal{U}_0$. Thus $\mathcal{W}_0$ is user-invariant. ■

C. Recoverably Safe Sets

Let $i = 1, ..., M$, with M the number of *recoverably safe sets* $\mathcal{W}_i$ to be calculated by solving (6) with the Hamiltonian

$$H_i\left(x, \tfrac{\partial J(x,t)}{\partial x}\right) = \max_{u_c \in \mathcal{U}_c} \max_{u_h \in \mathcal{U}_i} \tfrac{\partial J(x,t)}{\partial x}^T f(x, u_c, u_h) \quad (13)$$

with $\mathcal{U}_i = \alpha_i \mathcal{U}_h$, $\alpha_i \in (0, 1]$ and $\alpha_i < \alpha_{i+1}$ such that $\mathcal{U}_i \subset \mathcal{U}_{i+1}$. Note that the following property holds [23]:

$$\mathcal{W}_1 \subset \mathcal{W}_2 \subset ... \subset \mathcal{W}_M \quad (14)$$

The sets $\mathcal{W}_i, i \in \{1, ..., M\}$ are "recoverably safe" because they contain portions of the state space in which there always exists a control pair $(u_h, u_c) \in \mathcal{U}_i \times \mathcal{U}_c$ which maintains system safety. As with safe sets, the invariance preserving control law is *not* enforced along the boundaries of the sets. The recoverably safe sets provide information about what the user *must* do in order to preserve system safety, in case a disturbance input (external or user-applied) pushes the system into a configuration that the automation is unable to recover from on its own (i.e. states outside of $\mathcal{W}_0$).

Example 1: The recoverably safe set $\mathcal{W}_1$ is calculated with

$$\begin{aligned} H_1\left(x, \tfrac{\partial J(x,t)}{\partial x}\right) &= \max_{u_c \in \mathcal{U}_c} \max_{u_h \in \mathcal{U}_1} \tfrac{\partial J(x,t)}{\partial x}^T f(x, u_c, u_h) \\ &= \tfrac{\partial J(x,t)}{\partial x_1} x_2 + 6|\tfrac{\partial J(x,t)}{\partial x_2}| \end{aligned} \quad (15)$$

and $\mathcal{U}_1 = \mathcal{U}_h$. Since $x(0) = [4,3]^T \in \mathcal{W}_1$, with appropriate user assistance, a trajectory starting at $x(0)$ will remain safe.

Lemma 2: Recoverably safe sets are user-assisted-invariant.

Proof: By construction: For $\mathcal{W}_i,\ i \in \{1, ..., M\}$ computed with $u_h \in \mathcal{U}_i \subseteq \mathcal{U}_h$, for all $x(0) \in \mathcal{W}_i$, there exists a control pair $(u_h, u_c) \in \mathcal{U}_i \times \mathcal{U}_c$ such that $x(t) \in \mathcal{C}$ for all $t \geq 0$. Thus $\mathcal{W}_i,\ i \in \{1, ..., M\}$ are user-assisted-invariant by definition. ■

To summarize, we constructed $N + M + 1$ sets to encompass all possible human input. Combining (9) and (14),

$$\mathcal{W}_{-N} \subset \mathcal{W}_{-N+1} \subset ... \subset \mathcal{W}_0 \subset \mathcal{W}_1 \subset ... \subset \mathcal{W}_M \quad (16)$$

The set $\mathcal{W}_0$ acts as a reference – if the system is in a state outside of $\mathcal{W}_0$, a human input *must* be applied to prevent failure, as the automation is unable to preserve safety unassisted. The set $\mathcal{W}_M$ corresponds to the standard "safe" invariant set [20]; its complement $\overline{\mathcal{W}}_M$ corresponds to the unsafe subset of the state-space. A controller could be synthesized to ensure that the set $\mathcal{W}_M$ is never exited and safety is preserved.

IV. Abstraction to a DES

Definition 4: Let the index i denote the safety level of the invariant set $\mathcal{W}_i$, where safety level decreases as i increases. Let the safety level of a point $x \in \mathcal{X} \subseteq \mathbb{R}^n$ be given by the smallest i such that $x \in \mathcal{W}_i$. A region of the state space $\mathcal{M} \subseteq \mathcal{C}$ has a *homogeneous safety level* i if all $x \in \mathcal{M}$ have the same safety level i.

Definition 5: A DES abstraction of a continuous system under shared control (1) is considered *safety informative* if 1) each mode corresponds to a region of the state space which has homogeneous safety level and 2) the DES conveys whether the system is in a user-invariant or user-assisted-invariant subset of the state space.

A. Generation of modes

Let $i = -N, \ldots, 0, \ldots, M$, where N is the number of safe sets, and M is the number of recoverably safe sets. Define a map from the continuous state-space to the discrete state-space, based on a partition that divides $\mathcal{X}$ into $N + M + 2$ disjoint regions q_i as follows:

1) $\mathcal{W}_{-N} \rightarrow q_{-N}$
2) $\mathcal{W}_i \cap \overline{\mathcal{W}}_{i-1} \rightarrow q_i$, for $i = -N + 1, ..., 0, ..., M$,
3) $\overline{\mathcal{W}}_M \rightarrow q_{unsafe}$

Lemma 3: Modes defined by the above mapping represent cells of the state-space with homogeneous safety level.

Proof: By construction: For $\mathcal{W}_{-N} \rightarrow q_{-N}$, the cell defined by $\mathcal{W}_{-N}$ is of homogeneous safety level $-N$. For modes $\mathcal{W}_i \cap \overline{\mathcal{W}}_{i-1} \rightarrow q_i,\ i \in \{-N + 1, ..., 0, ..., M\}$, recall that by (16), $\mathcal{W}_{i-1} \subset \mathcal{W}_i$. Therefore the cells $\mathcal{W}_i \cap \overline{\mathcal{W}}_{i-1}$ are by definition of homogeneous safety level i. Thus the modes $q_i,\ i \in \{-N, \ldots, 0, \ldots, M\}$ correspond to cells of the state-space that have homogeneous safety level. ■

Example 1: The cells in Figure 2 map to modes q_{-1}, q_0, q_1 and q_{unsafe}, as shown in Figure 4.

B. Transition function

Define the set of events $\Sigma = \{\sigma_{up}, \sigma_{down}\}$, corresponding to an increase or decrease in safety level, respectively. These events are state-based transitions that occur when the state crosses into a neighboring cell:

$$\begin{aligned} \sigma_{up}: &\quad x(t^-) \in \mathcal{W}_i \cap \overline{\mathcal{W}}_{i-1} \rightarrow x(t^+) \in \mathcal{W}_{i-1} \\ \sigma_{down}: &\quad x(t^-) \in \mathcal{W}_{i-1} \cap \overline{\mathcal{W}}_i \rightarrow x(t^+) \in \overline{\mathcal{W}}_{i-1} \cap \mathcal{W}_i \end{aligned} \quad (17)$$

An important consequence of (16) for this mapping is that transitions can only occur between neighboring modes. Hence the transition function R is defined as

$$\begin{aligned} R(q_i, \sigma_{up}) &= q_{i-1}, \qquad i \in \{-N + 1, ..., 0, ..., M\} \\ R(q_i, \sigma_{down}) &= q_{i+1}, \qquad i \in \{-N, ..., 0, ..., M - 1\} \\ R(q_M, \sigma_{down}) &= q_{unsafe} \end{aligned} \quad (18)$$

Note that in general, σ_{up} may not exist.

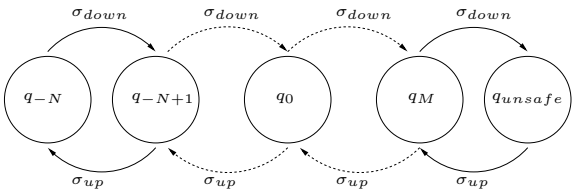

Fig. 3. DES $G = (Q, \Sigma, R)$, an abstraction of (1), constructed using (16) and reachability calculated with Hamiltonians (8), (11) and (13). The dashed transitions indicate a repeated pattern of transitions for a generic system with $N + M + 1$ modes, eventually passing through q_0.

C. Construction of the DES

The discrete event system $G = (Q, \Sigma, R)$ is constructed as illustrated in Figure 3. Since the designer decides how many modes to generate, G is of minimal mode cardinality. Details of the abstraction (and proof of its determinism) are presented in [20].

Lemma 4: The discrete event system $G = (Q, \Sigma, R)$ as defined in Figure 3 is safety informative.

Proof: The first condition is satisfied by Lemma 3. The second condition is satisfied by construction: modes $q_i,\ i \in \{-N, ..., 0\}$ correspond to user-invariant subsets of the state-space by Lemma 1, and modes $q_i,\ i \in \{1, ..., M\}$ correspond to user-assisted-invariant subsets of the state-space by Lemma 2. The transition $R(q_0, \sigma_{down}) = q_1$ from (18) corresponds to a transition from a user-invariant to a user-assisted-invariant subset of the state-space. ∎

The main advantage is that this abstraction provides the user with a warning that their actions may lead to unsafety. When in a user-invariant mode (i.e. $q_i,\ i \in \{-N, ..., 0\}$), the user is informed of safety restrictions on their input, but also free to violate these restrictions if they choose to. Is the user input violates safety restrictions, the system simply transitions to a user-assisted-invariant mode (i.e. $q_i,\ i \in \{1, ..., M\}$), indicating what input the user *must* apply in order to maintain system safety. Essentially, the user-assisted-invariant modes act as a buffer, allowing the user to "recover" to a higher safety level before the system enters the unsafe region of the state-space. Having multiple user-assistant-invariant modes provides more opportunities for correction. As the mode index i increases, so does the necessity for control action - a designer may choose to have increasing levels of alerts corresponding to increasing level of unsafety.

For Example 1, this algorithm results in the DES in Figure 4. In q_{-1}, the user is free to apply any input $|r| \leq 2$ without risking transitioning to a lower safety level. If the user violates these constraints, the system may transition into q_0. In this case, if the user is "hands-off" the controls ($r = 0$), the automation will still be able to maintain system safety. Once again, the user is free to apply inputs that drive the system to a lower safety level. However, once in q_1, the user *must* apply an input $r = 3 \cdot \text{sign}(x_2)$ to maintain that safety.

V. Example: Aircraft in Manual Mode

Consider manual control mode of the aircraft longitudinal dynamics introduced in [21], in which the flight crew sets the reference flight path angle, while the automation performs low level control tasks. Using the short period approximation, the state $x = [\alpha, \dot{\theta}, \gamma]$ consists of angle of attack α, pitch rate $\dot{\theta}$, and flight path angle γ [24]. The reference input $r \in \mathcal{R}$ consists of the reference flight path angle for γ. Elevator deflection δ_e is used to implement a static full-state feedback controller, yielding the closed loop dynamics [21]:

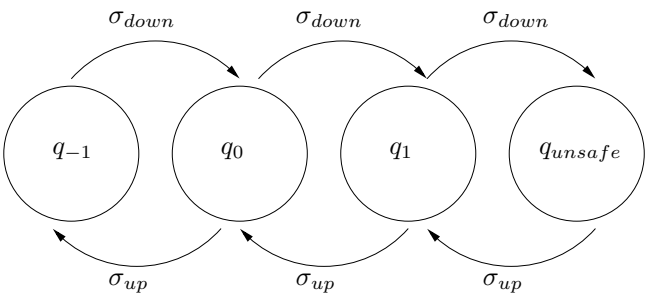

Fig. 4. DES $G = (Q, \Sigma, R)$ for Example 1. Note that q_{-1} represents a region in the continuous state-space that is safe, q_0 represents a region that is marginally safe, q_1 represents a region that is recoverably safe, and q_{unsafe} represents a region that is unsafe.

$$\begin{aligned} f_{\text{MAN}}(x, r) &= Ax + B(u_c(x) + u_h(r)) \\ &= A_{cl}x + B_{cl}r \end{aligned} \tag{19}$$

with $u_c(x) = -Kx$, $u_h(r) = N_r r$ and

$$\begin{aligned} A_{cl} &= \begin{bmatrix} -0.6486 & 0.9376 & -0.0963 \\ -2.6226 & -3.0477 & -3.0803 \\ 0.6486 & 0.0624 & 0.0963 \end{bmatrix} \\ B_{cl} &= -2.3 \begin{bmatrix} -0.0418 & -1.3391 & 0.0418 \end{bmatrix}^T \end{aligned} \tag{20}$$

where K is a state feedback matrix such that A_{cl} has eigenvalues at $-1.2, -1.2 \pm 0.12j$, and $N_r = -2.3$.

State constraints (due to the flight envelope) and control constraints (due to feedback under saturation) define

$$\begin{aligned} J_0(x) &= \min_x\{J_0^{\text{state}}(x), J_0^{\text{sat}}(x)\},\ \text{with} \\ J_0^{\text{state}}(x) &= \min_x \{x - x_{\min}, x_{\max} - x\} \\ J_0^{\text{sat}}(x) &= \min_x \{u_{\max} - \max_{r\in\mathcal{R}} \delta_e(x, r), \\ &\qquad \min_{r\in\mathcal{R}} \delta_e(x, r) - u_{\max}\} \end{aligned} \tag{21}$$

with state bounds $x_{\min} \leq x \leq x_{\max}$, $x_{\min} = [-11.5°, -15°, -13.3°]$, $x_{\max} = -x_{\min}$, $u_{\max} = 50°$, and $r \in \mathcal{R} = [-13.3°, 13.3°]$.

Choosing $N = M = 1$, invariant sets $\mathcal{W}_{-1}$,$\mathcal{W}_0$ and $\mathcal{W}_1$ are calculated as shown in Figure 5 (dark green solid, light yellow transparent, and red mesh, respectively). Safe set $\mathcal{W}_{-1}$ is computed by bounding the pilot's input to 25% of $\mathcal{R}$, a reasonable estimate of pilot behavior under normal operating conditions, with $\alpha_{-1} = \frac{.25|N|r_{max}}{u_{max}}$, $\mathcal{U}_{-1} = \alpha_{-1}\mathcal{U}_h$, and Hamiltonian as defined in (8). Marginally safe set $\mathcal{W}_0$ is calculated as in (11). Recoverably safe set $\mathcal{W}_1$ is calculated with $\alpha_1 = \frac{|N|(r_{max})}{u_{max}} = \frac{(2.3)(13.3°)}{50°}$, and $\mathcal{U}_1 = \alpha_1\mathcal{U}_h$ – we assume the pilot has full control authority, as per (13).

The state space is partitioned into four disjoint regions: $\mathcal{W}_{-1} \to q_{-1}$, $\mathcal{W}_0 \cap \overline{\mathcal{W}}_{-1} \to q_0$, $\mathcal{W}_1 \cap \overline{\mathcal{W}}_0 \to q_1$, and $\overline{\mathcal{W}}_1 \to q_{unsafe}$. The transition function R and DES G are shown in Figure 4 (the same DES as in Example 1, although the

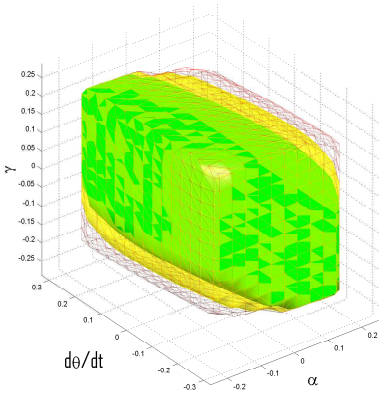

Fig. 5. The solid green (dark), transparent yellow (light) and red mesh sets represent, respectively, safe set $\mathcal{W}_{-1}$, marginally safe set $\mathcal{W}_0$, and recoverably safe set $\mathcal{W}_1$. $\mathcal{W}_{-1}$ is user-invariant (the user can apply any input $u_h \in \mathcal{U}_{-1}$ without affecting system safety). $\mathcal{W}_0$, although also user-invariant, is computed assuming $u_h(r) = 0$ (the automation can preserve safety without interference or assistance from the user). $\mathcal{W}_1$ is user-assisted-invariant – for states within this set but not contained in $\mathcal{W}_0$, the user *must* apply an input to preserve system safety.

events σ_{up} and σ_{down} correspond to state-based transitions defined in Figure 5).

The DES can be used as a user-interface, whose main benefit is that the flight crew knows at all times 1) what inputs can be applied without affecting system safety, 2) what inputs can be applied that reduce system system safety without causing failure and 3) what inputs must be applied to preserve system safety.

VI. Conclusion

We extended techniques for a reachability-based abstraction to continuous systems under shared control. To accommodate interactions between the human input and the automation input, we defined three invariance properties: invariance, user-invariance and user-assisted-invariance. These invariance properties were related to three levels of safety – 1) safe, 2) marginally safe and 3) recoverably safe – each determined by different assumptions on user behavior. We computed reachable sets for each of these safety levels by relating assumptions on user intent to the effect of user input on the Hamiltonian. We implemented an algorithm to create a reachability-based abstraction to a DES, by partitioning the state-space according to the computed invariant sets. This DES can be used to inform the design of a user-interface which conveys safety restrictions on the user's input. Finally, the utility of our method is demonstrated on an aircraft scenario under manual control.

We aim to develop a general theory to identify problems in human-automation interaction early in the design process. Future work includes extensions of the method presented here to generic hybrid systems under shared control, with both continuous and discrete human inputs.

References

[1] S. Umeno and N. Lynch, "Safety verification of an aircraft landing protocol: A refinement approach," in *Hybrid Systems: Comp & Ctr* (A. Bemporad, A. Bicci, and G. Buttazzo, eds.), LNCS 4416, pp. 557–572, Springer Verlag, April 2007.

[2] J. Crow, D. Javaux, and J. Rushby, "Models and mechanized methods that integrate human factors into automation design," in *Int'l Conf HCI Aeronautics*, (Toulouse, France), pp. 163–168, September 2000.

[3] A. Joshi, S. P. Miller, and M. P. Heimdahl, "Mode confusion analysis of a flight guidance system using formal methods," in *IEEE Digital Avionics Sys Conf (DASC 2003)*, pp. 2D.1–21–12 vol.1, Oct 2003.

[4] R. Boyatt and J. Sinclair, "A lightweight formal methods perspective on investigating aspects of interactive systems," in *Proc Int'l Workshop Formal Methods Interactive Sys (FMIS)*, (UK), Elsevier, Sept 2007.

[5] C. Tomlin, J. Lygeros, and S. Sastry, "A game theoretic approach to controller design for hybrid systems," *Proc IEEE*, vol. 88, no. 7, pp. 949–970, 2000.

[6] C. Tomlin, I. Mitchell, A. Bayen, and M. Oishi, "Computational techniques for the verification of hybrid systems," *Proc IEEE*, vol. 91, no. 7, pp. 986–1001, 2003.

[7] I. Mitchell, *A Toolbox of Level Set Methods*. Department of Computer Science, University of British Columbia, June 2004. www.cs.ubc.ca/∼mitchell/ToolboxLS.

[8] S. Prajna, A. Papachristodoulou, P. Seiler, and P. A. Parrilo, *SOSTOOLS: Sum of squares optimization toolbox for MATLAB*, 2004.

[9] G. Frehse, "PHAVer: Algorithmic verification of hybrid systems past HyTech.," in *Hybrid Systems: Comp & Ctr* (M. Morari and L. Thiele, eds.), LNCS 34143, pp. 258–273, Springer Verlag, 2005.

[10] M. Kvasnica, P. Grieder, and M. Baotic, "Multi-Parametric Toolbox (MPT)," 2004.

[11] A. Chutinan and B. Krogh, "Computational techniques for hybrid system verification," *IEEE Trans Auto Ctr*, vol. 48, pp. 64–75, 2003.

[12] E. Asarin, T. Dang, and A. Girard, "Reachability analysis of nonlinear systems using conservative approximation," in *Hybrid Systems: Comp & Ctr* (O. Maler, A. Pneuli, eds.), LNCS 2623, pp. 20–35, Springer Verlag, 2003.

[13] A. Degani and M. Heymann, "Formal verification of human automation interaction," *Human Factors*, vol. 44, no. 1, pp. 28–43, 2002.

[14] A. Suzuki, T. Ushio, and M. Adachi, "Detection of automation surprisees in discrete event systems operated by multiple users," in *SICE-ICASE Int'l Joint Conf*, (Korea), pp. 1115–1119, October 2006.

[15] L. Sherry and R. Feary, "Task design and verification testing for certification of avionics equipment," in *Proc AIAA/IEEE Digital Avionics Sys Conf*, pp. 10.A.3–10.A.10, September 2004.

[16] A. Cerone, P. Lindsay, and S. Connelly, "Formal analysis of human-computer interaction using model-checking," in *IEEE Int'l Conf Software Eng & Formal Methods*, pp. 352–361, IEEE, Sept 2005.

[17] J. Bowen and S. Reeves, "Formal models of informal GUI designs," in *Proc 1st Int'l Conf on Software Formal Methods for Interactive Sys*, Elec Notes Theor Comp Sci, pp. 57–72, Elsevier Science, July 2007.

[18] J. Sequeira and I. Ribeiro, "Hybrid control of semi-autonomous robots," in *Int'l Conf on Intelligent Robots & Sys (IROS)*, (Sendai, Japan), pp. 1838–1844, October 2004.

[19] C. Lesire and C. Tessier, "Estimation and conflict detection in human controlled systems," in *Hybrid Systems: Comp & Ctr* (J. Hespanha and A. Tiwari, eds.), LNCS 3927, pp. 407–420, Springer Verlag, 2006.

[20] M. Oishi, I. Mitchell, A. M. Bayen, and C. J. Tomlin, "Invariance-preserving abstractions of hybrid systems: Application to user interface design," *IEEE Trans Ctr Sys Tech*, vol. 16, pp. 229–244, March 2008.

[21] N. Matni and M. Oishi, "Reachability-based abstraction for an aircraft landing under shared control," in *Proc Amer Ctr Conf*, (Seattle, WA), pp. 2278–2284, June 2008.

[22] P. Ladkin and H. Sogame, "Aircraft accident investigation report 96-5." http://sunnyday.mit.edu/accidents/nag-contents.html, July 1996.

[23] E. Cruck and P. Saint-Pierre, "Nonlinear impulse target problems under state constraint: A numerical analysis based on viability theory," *Set-Valued Analysis*, vol. 12, pp. 383–416, December 2004.

[24] D. McRuer, I. Ashkenas, and D. Graham, *Aircraft dynamics and automatic control*. Princeton University Press, 1973.