Behavioural contracts for linear dynamical systems

Shali, Brayan; van der Schaft, Arjan; Besselink, Bart

Published in:
2021 European Control Conference (ECC)

DOI:
10.23919/ECC54610.2021.9654875

Publication date:
2021

2021 European Control Conference (ECC)
June 29 - July 2, 2021. Rotterdam, Netherlands

Behavioural contracts for linear dynamical systems: input assumptions and output guarantees

B. M. Shali, A. J. van der Schaft, B. Besselink

Abstract— **We introduce contracts for linear dynamical systems with inputs and outputs. Contracts are used to express formal specifications on the dynamic behaviour of such systems through two aspects: assumptions and guarantees. The assumptions are a linear system that captures the available knowledge about the dynamic behaviour of the environment in which the system is supposed to operate. The guarantees are a linear system that captures the required dynamic behaviour of the system when interconnected with its environment. In addition to contracts, we also define and characterize notions of contract refinement and contract conjunction. Contract refinement allows one to determine if a contract expresses a stricter specifications than another contract. On the other hand, contract conjunction allows one to combine multiple contracts into a single contract that fuses the specifications they express.**

I. Introduction

Modern engineering systems, such as smart grids and intelligent transportation systems, often comprise a large number of interconnected physical components that are modelled as continuous dynamical systems. Specifications on such components typically belong to one of two categories: dissipativity or set-invariance. Dissipativity theory [1] provides an elegant unifying framework that captures requirements such as stability, passivity or performance, while set-invariance techniques [2] are natural candidates for expressing safety requirements. However, the growing complexity of modern engineering systems necessitates a theory of specifications that goes beyond dissipativity and set-invariance. Namely, these frameworks generally do not permit *dynamic* specifications: the supply rates that are central to dissipativity theory are static, as are typical invariant sets. Moreover, when employed as specifications on components of large-scale interconnected systems, these frameworks do not allow to explicitly take the (dynamics of the) environment of such a component into account, possibly making the specification unnecessarily conservative.

Motivated by these issues, we introduce *contracts* as specifications for linear dynamical systems. Contracts were initially developed in the field of software engineering [3] but were later adapted to a variety of system classes in computer science: rely-guarantee contracts were introduced in [4] to deal with programs that operate concurrently, while the popularisation of interface theories [5], [6] has led to a boom in the development of contract-based theories for cyber-physical systems [7], [8]. Consequently, there is a great diversity of styles and approaches to using contracts as specifications, which nevertheless follow a common philosophy, namely, to support the independent design of components within interconnected systems. This philosophy is captured in the meta-theory of contracts introduced in [9]. This meta-theory abstracts away the specific design choices made when developing a contract theory, while still formally defining all relevant concepts.

Inspired by this meta-theory, we define assume-guarantee contracts for linear dynamical systems with inputs and outputs. As the name suggests, these contracts consist of assumptions and guarantees, both of which are linear systems themselves, and which can be used to express specifications for a system in the following sense. First, the assumptions capture the available knowledge on the dynamic behaviour of the environment in which the system is supposed to operate, thus establishing a class of compatible environments. Second, the guarantees capture the required dynamic behaviour of the system when interconnected with a compatible environment, thus establishing a class of implementations. This is formalized very naturally using the behavioural approach to systems theory [10], [11].

In addition to defining contracts, we characterize contract implementation and provide a method for verifying that a given linear system implements a given contract. We also define and characterize the concepts of contract refinement and contract conjunction, again taking inspiration from the meta-theory in [9]. The notion of contract refinement allows us to reason when one contract represents a stricter specification than another contract. On the other hand, the notion of contract conjunction allows us to construct a contract that fuses the specifications expressed by several different contracts. As will be shown later in this paper, these two concepts are intimately related and can be characterized in a very intuitive and conceptually simple manner.

We regard this work as a first step towards a comprehensive contract theory for linear dynamical systems. Nevertheless, we note that ideas from contract theories have already been used to express specifications on dynamical systems. Assume-guarantee contracts that capture set-invariance properties were introduced in [12] and used in [13] for the design of symbolic controllers. Closely related are also the assume-guarantee contracts for safety introduced in [14]. However, in contrast to the contracts in this paper, the contracts presented in [13] and [14] do not allow specifications involving dynamics, neither in the assumptions nor in the guarantees. This is not the case for another class of contracts, called parametric assume-guarantee contracts [15],

The authors are with the Jan C. Willems Center for Systems and Control, and the Bernoulli Institute for Mathematics, Computer Science, and Artificial Intelligence, University of Groningen, Groningen, The Netherlands; Email: *b.m.shali@rug.nl*; *a.j.van.der.schaft@rug.nl*; *b.besselink@rug.nl*.

[16], which were introduced to express specifications on input-output gain properties. Nonetheless, the latter are only defined for discrete systems, whereas we consider continuous systems in this paper. In fact, the contracts in this paper are most closely related to the contracts introduced in [17], the main difference being that the external variable there is not assumed to be an input-output pair. Our work is also closely related to the work on compositional analysis and assume-guarantee reasoning for linear systems presented in [18], [19], [20], although these do not explicitly define contracts.

The remainder of this paper is structured as follows. In Section II, we discuss the class of systems that will be treated in this paper. There, we also review the concept of external behaviour together with some relevant results. In Section III, we define assume-guarantee contracts for the class of systems discussed in Section II. Moreover, we define and characterize the notions of contract implementation, contract refinement and contract conjunction. As such, Section III contains the main contributions of this paper. These are then demonstrated with an illustrative example in Section IV, followed by concluding remarks in Section V.

The notation in this paper is mostly standard. The space of smooth functions from $\mathbb{R}$ to $\mathbb{R}^n$ is denoted by $\mathcal{C}_n^\infty$. A matrix whose entries are polynomials is called a *polynomial matrix*, and a matrix whose entries are rational functions is called a *rational matrix*. All polynomials are univariate and have real coefficients. A rational function is *proper* if the degree of its denominator is greater than or equal to the degree of its numerator. A rational matrix is proper if all of its entries are proper rational functions. We say that a square polynomial matrix $P(s)$ is *invertible* if there exists a rational matrix $Q(s)$ such that $P(s)Q(s) = I$. We say that $P(s)$ is *unimodular* if there exists a *polynomial* matrix $Q(s)$ such that $P(s)Q(s) = I$. In both cases, we say that $Q(s)$ is the *inverse* of $P(s)$, which we denote by $P(s)^{-1}$.

II. Models of physical systems

Consider the linear system

$$\Sigma : \begin{cases} \dot{x} = Ax + Bu, \\ y = Cx + Du, \end{cases} \tag{1}$$

with state trajectory $x \in \mathcal{C}_n^\infty$, input trajectory $u \in \mathcal{C}_m^\infty$, output trajectory $y \in \mathcal{C}_p^\infty$. The system Σ is regarded as an open system in which the external variables u and y interact with the environment, whereas the state x is internal and does not interact with the environment. More precisely, Σ is seen

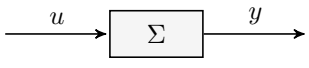

Fig. 1. The system Σ as a signal processor.

as a signal processor that takes an externally provided input signal $u \in \mathcal{C}_m^\infty$ and transforms it to an externally available output signal $y \in \mathcal{C}_p^\infty$, as shown by the diagram in Figure 1. Our goal in this paper is to develop a formal method for expressing specifications on the dynamic behaviour of such a system. The approach we take requires a method for system comparison; we need to be able to say when one system behaves "like" another system. This is formalized quite naturally using the behavioural approach to systems theory [21], [11]. In particular, we define the *external behaviour* $\mathfrak{B}(\Sigma)$ of Σ as the linear subspace

$$\mathfrak{B}(\Sigma) = \left\{ (u, y) \in \mathcal{C}_{m+p}^\infty \;\middle|\; \exists x \in \mathcal{C}_n^\infty \text{ s.t. (1) holds} \right\}.$$

Remark 1: The external behaviour can be used to compare systems in the following sense. Consider two systems Σ_1 and Σ_2 of the form (1). If $\mathfrak{B}(\Sigma_1) \subset \mathfrak{B}(\Sigma_2)$, then for a given input, the set of outputs generated by Σ_1 is contained in the set of outputs generated by Σ_2, thus Σ_2 can be considered to have "richer" dynamics than Σ_1. Taking this a step further, if $\mathfrak{B}(\Sigma_1) = \mathfrak{B}(\Sigma_2)$, then for a given input, the set of outputs generated by Σ_1 is precisely the same as the set of outputs generated by Σ_2, hence Σ_1 and Σ_2 cannot be distinguished on the basis of external behaviour alone. In such a case, we will view Σ_1 and Σ_2 as different representations of the same external behaviour rather than different systems, i.e., we identify the system with its external behaviour.

There are many different representations of a given external behaviour. As we are interested in providing specifications only on the external behaviour of a system, a representation that does not involve the internal state would be more appropriate for our purposes. With this in mind, consider the linear system

$$\Sigma : P\left(\tfrac{\mathrm{d}}{\mathrm{d}t}\right) y = Q\left(\tfrac{\mathrm{d}}{\mathrm{d}t}\right) u \tag{2}$$

with $u \in \mathcal{C}_m^\infty$, $y \in \mathcal{C}_p^\infty$ and real polynomial matrices $P(s)$ and $Q(s)$. If $P(s)$ is square and invertible and $P(s)^{-1}Q(s)$ is a proper rational matrix, then we say that Σ is in *input-output form* [11, Section 3.3]. The system Σ being in input-output form guarantees that u can be chosen freely, i.e., for all $u \in \mathcal{C}_m^\infty$, there exists $y \in \mathcal{C}_p^\infty$ such that $(u, y) \in \mathfrak{B}(\Sigma)$. In fact, it also guarantees that none of the components of y can be chosen freely, thus ensuring that u and y in (2) have the roles of input and output, respectively. Similarly to before, we define the external behaviour of Σ of the form (2) as

$$\mathfrak{B}(\Sigma) = \left\{ (u, y) \in \mathcal{C}_{m+p}^\infty \;\middle|\; (2) \text{ holds} \right\}.$$

The external behaviour of a system of the form (1) can be compared to that of a system of the form (2). In fact, we have the following result on representations.

Theorem 1: [22, Theorem 6.2] Let $\mathfrak{B} \subset \mathcal{C}_{m+p}^\infty$ be a linear subspace. There exists Σ_1 of the form (1) such that $\mathfrak{B}(\Sigma_1) = \mathfrak{B}$ if and only if there exists Σ_2 of the form (2) in input-output form such that $\mathfrak{B}(\Sigma_2) = \mathfrak{B}$.

In view of Theorem 1, it makes no difference whether we consider systems of the form (1) or systems of the form (2) in input-output form to represent external behaviours. However, the latter are better suited for the type of analysis that will be carried out in the following sections. For example, inclusion of external behaviours has a simple algebraic characterization for systems of the form (2), as shown next.

Theorem 2: Let $\mathfrak{B}_j$, $j \in \{1, 2\}$, be defined as

$$\mathfrak{B}_j = \left\{ w \in \mathcal{C}_k^\infty \;\middle|\; R_j\left(\tfrac{\mathrm{d}}{\mathrm{d}t}\right) w = 0 \right\},$$

where $R_j(s)$ is a polynomial matrix. Then $\mathfrak{B}_1 \subset \mathfrak{B}_2$ if and only if there exists a polynomial matrix $M(s)$ such that $R_2(s) = M(s)R_1(s)$.

Proof: Suppose that there exists a polynomial matrix $M(s)$ satisfying $R_2(s) = M(s)R_1(s)$. Let $w \in \mathfrak{B}_1$. Then

$$R_2\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)w = M\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)R_1\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)w = 0,$$

hence $w \in \mathfrak{B}_2$ and thus $\mathfrak{B}_1 \subset \mathfrak{B}_2$. Conversely, suppose that $\mathfrak{B}_1 \subset \mathfrak{B}_2$, i.e., $R_1\big(\frac{\mathrm{d}}{\mathrm{d}t}\big)w = 0$ implies $R_2\big(\frac{\mathrm{d}}{\mathrm{d}t}\big)w = 0$. Using [23, Lemma 2.1], it follows that there exists a polynomial matrix $M(s)$ such that $R_2(s) = M(s)R_1(s)$. ∎

As a consequence of Theorem 2, given two systems Σ_1 and Σ_2 of the form (2), we have that $\mathfrak{B}(\Sigma_1) \subset \mathfrak{B}(\Sigma_2)$ if and only if there exists a polynomial matrix $M(s)$ such that

$$\begin{bmatrix} P_2(s) & -Q_2(s) \end{bmatrix} = M(s) \begin{bmatrix} P_1(s) & -Q_1(s) \end{bmatrix}.$$

The following lemma, which is proven in [24, Lemma 1], utilizes the Smith canonical form [25, Section 1.8] to produce a condition for the existence of such an $M(s)$.

Lemma 1: Let $R_1(s)$ and $R_2(s)$ be polynomial matrices and assume that $R_1(\lambda)$ has full row rank for some $\lambda \in \mathbb{C}$. Moreover, let $U_1(s)$ and $V_1(s)$ be unimodular matrices that bring $R_1(s)$ to its Smith canonical form, i.e., $R_1(s) = U_1(s) \begin{bmatrix} D_1(s) & 0 \end{bmatrix} V_1(s)$, where $D_1(s)$ is an invertible diagonal polynomial matrix. Then there exists a polynomial matrix $M(s)$ such that $R_2(s) = M(s)R_1(s)$ if and only if the following conditions hold:

1) $R_2(s)V_1(s)^{-1} \begin{bmatrix} 0 \\ I \end{bmatrix} = 0$;

2) $R_2(s)V_1(s)^{-1} \begin{bmatrix} D_1(s)^{-1} \\ 0 \end{bmatrix}$ is a polynomial matrix.

Note that if Σ_1 of the form (2) is in input-output form, then $[P_1(\lambda) \quad Q_1(\lambda)]$ has full row rank for some $\lambda \in \mathbb{C}$ because $P_1(s)$ is invertible, hence the assumption on $R_1(s)$ in Lemma 1 is satisfied. More generally, given a behaviour

$$\mathfrak{B} = \big\{ w \in \mathcal{C}_k^\infty \;\big|\; R\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)w = 0 \big\},$$

we can always find a polynomial matrix $R'(s)$ such that

$$\mathfrak{B} = \big\{ w \in \mathcal{C}_k^\infty \;\big|\; R'\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)w = 0 \big\}$$

and $R'(\lambda)$ has full row rank for some $\lambda \in \mathbb{C}$, hence the assumption on $R_1(s)$ in Lemma 1 is not restrictive at all.

We conclude this section with the following remark.

Remark 2: Although we have chosen to represent external behaviours without involving a state variable, it is often convenient to represent a given external behaviour with the help of so-called *latent variables*. These are variables that are included in the representation of the external behaviour but are not necessarily of interest to us. Therefore, most generally, we will consider systems of the form

$$\Sigma : P\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)y = Q\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)u + E\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)l, \tag{3}$$

where $l \in \mathcal{C}_r^\infty$ is a latent variable and $E(s)$ is a polynomial matrix. Note that Σ of the form (1) is actually a latent variable representation of its external behaviour $\mathfrak{B}(\Sigma)$ with the state x as the latent variable. We already know that for all Σ_1 of the form (1) there exists Σ_2 of the form (2) in input-output form such that $\mathfrak{B}(\Sigma_1) = \mathfrak{B}(\Sigma_2)$. More generally, the latent variable in Σ_3 of the form (3) can always be eliminated to obtain Σ_2 of the form (2) such that $\mathfrak{B}(\Sigma_3) = \mathfrak{B}(\Sigma_2)$, as follows from [11, Theorem 6.2.6].

III. Contracts

Consider a system Σ of the form (2) in input-output form. We want to express specifications on the external behaviour of such a system. To this end, as an open system, we can assume that Σ operates in interconnection with its environment. Having knowledge about this environment can ease the design burden of Σ, hence it should be taken into account when expressing specifications. With this in mind, we will assume that the environment of Σ is another system that generates inputs for it. More precisely, an *environment* E is a system of the form

$$\mathrm{E} : 0 = E\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)u, \tag{4}$$

where $u \in \mathcal{C}_m^\infty$ and $E(s)$ is a real polynomial matrix. The environment E defines the *input behaviour*

$$\mathfrak{B}_{\mathrm{i}}(\mathrm{E}) = \{ u \in \mathcal{C}_m^\infty \;|\; (4) \text{ holds} \}.$$

The interconnection of Σ with the environment E is obtained by setting the input generated by E as input of Σ. This results in the interconnection

$$\mathrm{E} \wedge \Sigma : \begin{bmatrix} P\big(\frac{\mathrm{d}}{\mathrm{d}t}\big) \\ 0 \end{bmatrix} y = \begin{bmatrix} Q\big(\frac{\mathrm{d}}{\mathrm{d}t}\big) \\ E\big(\frac{\mathrm{d}}{\mathrm{d}t}\big) \end{bmatrix} u, \tag{5}$$

which is represented graphically in Figure 2.

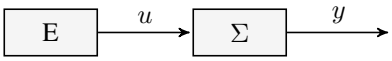

Fig. 2. The interconnection $\mathrm{E} \wedge \Sigma$.

As a means of expressing specifications on the external behaviour of Σ, we are interested in guaranteeing properties of Σ when interconnected with relevant environments E. In particular, if we define the *output behaviour*

$$\mathfrak{B}_{\mathrm{o}}(\mathrm{E} \wedge \Sigma) = \big\{ y \in \mathcal{C}_p^\infty \;\big|\; (5) \text{ holds} \big\},$$

then we want to guarantee properties of $\mathfrak{B}_{\mathrm{o}}(\mathrm{E} \wedge \Sigma)$ for all relevant environments E. We will make this explicit by introducing two systems: *assumptions* A and *guarantees* Γ. Assumptions A are a system of the form

$$\mathrm{A} : 0 = A\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)u, \tag{6}$$

and, like environments, they represent the input behaviour

$$\mathfrak{B}_{\mathrm{i}}(\mathrm{A}) = \{ u \in \mathcal{C}_m^\infty \;|\; (6) \text{ holds} \}.$$

On the other hand, guarantees Γ are a system of the form

$$\Gamma : G\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)y = 0, \tag{7}$$

and they represent the output behaviour

$$\mathfrak{B}_{\mathrm{o}}(\Gamma) = \big\{ y \in \mathcal{C}_p^\infty \;\big|\; (7) \text{ holds} \big\}.$$

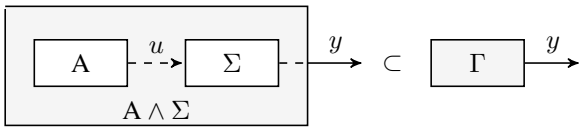

Fig. 3. Implementation of $\mathcal{C} = (\mathrm{A}, \Gamma)$.

Remark 3: Although we define environments, assumptions and guarantees as systems involving a single variable, we might represent the behaviours they define with the help of latent variables, as in the general form (3). Since latent variables can always be eliminated (recall Remark 2), the representations (4), (6) and (7) do not pose a restriction.

With assumptions and guarantees defined, we are ready to introduce the definition of a contract.

Definition 1: A *contract* $\mathcal{C}$ is a pair (A, Γ) of assumptions A and guarantees Γ.

Contracts can be used to express formal specifications for the external behaviour of systems Σ of the form (2) in input-output form, as captured in the following definition.

Definition 2: An environment E is *compatible* with the contract $\mathcal{C} = (\mathrm{A}, \Gamma)$ if $\mathfrak{B}_{\mathrm{i}}(\mathrm{E}) \subset \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$. A system Σ of the form (2) in input-output form *implements* $\mathcal{C}$ if $\mathfrak{B}_{\mathrm{o}}(\mathrm{E} \wedge \Sigma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma)$ for all environments E compatible with $\mathcal{C}$. In such a case, we say that Σ is an *implementation* of $\mathcal{C}$.

In other words, an environment is compatible with a contract if the inputs that it generates can be generated by the assumptions, and a system implements a contract if the outputs that it generates when interconnected with any compatible environment can be generated by the guarantees. A contract thus gives a formal specification for the external behaviour of a system through two aspects. First, it specifies (by the assumptions) the class of environments in which the system is supposed to operate. Second, it characterizes the required external behaviour of the system through the guarantees, which the system needs to satisfy for any compatible environment. We emphasize that both the assumptions and guarantees are dynamical systems, hence the specification that a contract expresses is also dynamic.

Although contract implementation is defined using the class of compatible environments, verifying whether a system Σ of the form (2) in input-output form is an implementation can be done directly via the assumptions and guarantees, i.e., without explicitly constructing the class of compatible environments. This is stated in the following theorem, which is represented graphically in Figure 3.

Theorem 3: A system Σ of the form (2) is an implementation of $\mathcal{C} = (\mathrm{A}, \Gamma)$ if and only if $\mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma)$.

Proof: Suppose that $\mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma)$. Note that $y \in \mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma)$ if and only if there exists $u \in \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$ such that $(u, y) \in \mathfrak{B}(\Sigma)$. With this in mind, let E be an environment compatible with $\mathcal{C}$, and let $y \in \mathfrak{B}_{\mathrm{o}}(\mathrm{E} \wedge \Sigma)$. Then there exists $u \in \mathfrak{B}_{\mathrm{i}}(\mathrm{E}) \subset \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$ such that $(u, y) \in \mathfrak{B}(\Sigma)$, hence $y \in \mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma)$. This shows that $\mathfrak{B}_{\mathrm{o}}(\mathrm{E} \wedge \Sigma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma)$ for all compatible environments E and thus Σ is an implementation of $\mathcal{C}$. Conversely, suppose that Σ is an implementation of $\mathcal{C}$. Since A is a compatible environment of $\mathcal{C}$, it follows that $\mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma)$. ■

Remark 4: Theorem 3 allows us to verify that Σ of the form (2) is an implementation of $\mathcal{C} = (\mathrm{A}, \Gamma)$ provided that we can verify that $\mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma)$. The latter can be done in two steps. First, in view of [11, Theorem 6.2.6], we can eliminate the variable u from $\mathrm{A} \wedge \Sigma$ to obtain

$$\mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma) = \left\{ y \in \mathcal{C}_p^\infty \;\middle|\; R\left(\tfrac{\mathrm{d}}{\mathrm{d}t}\right) y = 0 \right\} \tag{8}$$

for some polynomial matrix $R(s)$, where $R(\lambda)$ has full row rank for some $\lambda \in \mathbb{C}$. Second, from Theorem 2 we know that $\mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \mathrm{E}) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma)$ if and only if there exists a polynomial matrix $M(s)$ such that $G(s) = M(s)R(s)$, where the latter can be verified using Lemma 1. Note that due to Theorem 1, Theorem 3 is also valid for systems Σ of the form (1). For the latter, we need to eliminate x as well as u in order to obtain a polynomial matrix $R(s)$ such that (8) holds.

A distinguishing feature of using contracts for specifications is that contracts themselves can be compared through a notion of refinement.

Definition 3: The contract $\mathcal{C}_1$ *refines* the contract $\mathcal{C}_2$, denoted by $\mathcal{C}_1 \preccurlyeq \mathcal{C}_2$, if all compatible environments of $\mathcal{C}_2$ are compatible environments of $\mathcal{C}_1$ and all implementations of $\mathcal{C}_1$ are implementations of $\mathcal{C}_2$.

We have that $\mathcal{C}_1$ refines $\mathcal{C}_2$ if it specifies stricter guarantees that have to be satisfied for a larger class of environments. In such a case, it is clear that $\mathcal{C}_1$ expresses a more restrictive specification than $\mathcal{C}_2$. Just like contract implementation, contract refinement can be verified on the basis of assumptions and guarantees alone, i.e., without explicitly constructing the classes of compatible environments and implementations of the two contracts. This is the content of the following theorem, whose proof can be found in [24, Appendix].

Theorem 4: The contract $\mathcal{C}_1 = (\mathrm{A}_1, \Gamma_1)$ refines the contract $\mathcal{C}_2 = (\mathrm{A}_2, \Gamma_2)$ if and only if $\mathfrak{B}_{\mathrm{i}}(\mathrm{A}_2) \subset \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_1)$ and $\mathfrak{B}_{\mathrm{o}}(\Gamma_1) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma_2)$.

If a contract $\mathcal{C}$ refines both contracts $\mathcal{C}_1$ and $\mathcal{C}_2$, then $\mathcal{C}$ captures both the specifications that $\mathcal{C}_1$ expresses and the specifications that $\mathcal{C}_2$ expresses. This suggests that multiple contracts can be combined into a single contract that represents a fusion of their specifications. This motivates the following definition.

Definition 4: The *conjunction* of contracts $\mathcal{C}_1$ and $\mathcal{C}_2$, denoted by $\mathcal{C}_1 \wedge \mathcal{C}_2$, is the largest (with respect to contract refinement) contract that refines both $\mathcal{C}_1$ and $\mathcal{C}_2$.

Note that the largest contract that refines both $\mathcal{C}_1$ and $\mathcal{C}_2$ corresponds to the least restrictive contract that fuses the specifications of $\mathcal{C}_1$ and $\mathcal{C}_2$. This suggests that the guarantees of $\mathcal{C}_1 \wedge \mathcal{C}_2$ should be the guarantees common to $\mathcal{C}_1$ and $\mathcal{C}_2$ and nothing more, and the assumptions of $\mathcal{C}_1 \wedge \mathcal{C}_2$ should include the assumptions of both $\mathcal{C}_1$ and $\mathcal{C}_2$ and nothing less. The following lemma captures some of this intuition.

Lemma 2: A contract $\mathcal{C} = (\mathrm{A}, \Gamma)$ refines both contracts $\mathcal{C}_1 = (\mathrm{A}_1, \Gamma_1)$ and $\mathcal{C}_2 = (\mathrm{A}_2, \Gamma_2)$ if and only if $\mathfrak{B}_{\mathrm{i}}(\mathrm{A}_1) + \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_2) \subset \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$ and $\mathfrak{B}_{\mathrm{o}}(\Gamma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma_1) \cap \mathfrak{B}_{\mathrm{o}}(\Gamma_2)$.

Proof: Due to Theorem 4, the contract $\mathcal{C}$ refines both contracts $\mathcal{C}_1$ and $\mathcal{C}_2$ if and only if $\mathfrak{B}_{\mathrm{i}}(\mathrm{A}_i) \subset \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$ and $\mathfrak{B}_{\mathrm{o}}(\Gamma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma_i)$ for all $i \in \{1, 2\}$. Since behaviours are

linear, this is the case if and only if $\mathfrak{B}_{\mathrm{i}}(\mathrm{A}_1) + \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_2) \subset \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$ and $\mathfrak{B}_{\mathrm{o}}(\Gamma) \subset \mathfrak{B}_{\mathrm{o}}(\Gamma_1) \cap \mathfrak{B}_{\mathrm{o}}(\Gamma_2)$. ■

In view of Theorem 4, Lemma 2 tells us that if there exist assumptions A with $\mathfrak{B}_{\mathrm{i}}(\mathrm{A}) = \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_1) + \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_2)$ and guarantees Γ with $\mathfrak{B}_{\mathrm{o}}(\Gamma) = \mathfrak{B}_{\mathrm{o}}(\Gamma_1) \cap \mathfrak{B}_{\mathrm{o}}(\Gamma_2)$, then (A, Γ) is the largest contract that refines both $\mathcal{C}_1 = (\mathrm{A}_1, \Gamma_1)$ and $\mathcal{C}_2 = (\mathrm{A}_2, \Gamma_2)$, hence $\mathcal{C}_1 \wedge \mathcal{C}_2 = (\mathrm{A}, \Gamma)$. Fortunately, such assumptions and guarantees exist and are defined below.

Definition 5: The *join* of assumptions A_1 and A_2, denoted by $\mathrm{A}_1 \vee \mathrm{A}_2$, is defined as the assumptions

$$\mathrm{A}_1 \vee \mathrm{A}_2 : \begin{bmatrix} I & I \\ A_1\big(\frac{\mathrm{d}}{\mathrm{d}t}\big) & 0 \\ 0 & A_2\big(\frac{\mathrm{d}}{\mathrm{d}t}\big) \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \end{bmatrix} = \begin{bmatrix} I \\ 0 \\ 0 \end{bmatrix} u.$$

The *meet* of guarantees Γ_1 and Γ_2, denoted by $\Gamma_1 \wedge \Gamma_2$, is defined as the guarantees

$$\Gamma_1 \wedge \Gamma_2 : \begin{bmatrix} G_1\big(\frac{\mathrm{d}}{\mathrm{d}t}\big) \\ G_2\big(\frac{\mathrm{d}}{\mathrm{d}t}\big) \end{bmatrix} y = 0.$$

We remark that although the join $\mathrm{A}_1 \vee \mathrm{A}_2$ is represented with latent variables, these can be eliminated to obtain assumptions as in (6). However, the latent variable representation clearly indicates that every $u \in \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_1 \vee \mathrm{A}_2)$ is obtained by summing $l_1 \in \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_1)$ and $l_2 \in \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_2)$. Having defined the join and meet, we can now write the conjunction of contracts $\mathcal{C}_1$ and $\mathcal{C}_2$ explicitly.

Theorem 5: The conjunction of contracts $\mathcal{C}_1 = (\mathrm{A}_1, \Gamma_1)$ and $\mathcal{C}_2 = (\mathrm{A}_2, \Gamma_2)$ is given by

$$\mathcal{C}_1 \wedge \mathcal{C}_2 = (\mathrm{A}_1 \vee \mathrm{A}_2, \Gamma_1 \wedge \Gamma_2).$$

Proof: It is easy to see that $\mathfrak{B}_{\mathrm{i}}(\mathrm{A}_1 \vee \mathrm{A}_2) = \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_1) + \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_2)$ and $\mathfrak{B}_{\mathrm{o}}(\Gamma_1 \wedge \Gamma_2) = \mathfrak{B}_{\mathrm{o}}(\Gamma_1) \cap \mathfrak{B}_{\mathrm{o}}(\Gamma_2)$. Therefore, due to Lemma 2 and Theorem 4, $(\mathrm{A}_1 \vee \mathrm{A}_2, \Gamma_1 \wedge \Gamma_2)$ is the largest contract that refines both $\mathcal{C}_1$ and $\mathcal{C}_2$. ■

Theorem 5 allows us to check when Σ of the form (2) in input-output form is simultaneously an implementation of $\mathcal{C}_1$ and $\mathcal{C}_2$. Indeed, this is the case if and only if Σ implements the conjunction $\mathcal{C}_1 \wedge \mathcal{C}_2$, whose assumptions and guarantees we can compute explicitly. Given the assumptions and guarantees of $\mathcal{C}_1 \wedge \mathcal{C}_2$, we have already explained how to verify that Σ implements $\mathcal{C}_1 \wedge \mathcal{C}_2$ in Remark 4.

IV. Illustrative example

In this section, we will illustrate the concepts and results from last section with a simple example. To this end, suppose that we need to design a car component that does not vibrate while the car is moving, e.g., the driver's seat. For simplicity, we only consider the vertical motion of the car, which we describe by a quarter car model, as shown in Figure 4. The model consists of two masses: the mass m_1 of the body and the mass m_2 of the wheel. The wheel is attached to the body of the car through a spring with constant k_1 and a damper with coefficient b. The wheel is connected to the ground by a tire represented as a spring with constant k_2. Let u_1 and u_2 be the vertical positions of the body and the wheel, respectively, and let l be the reference signal from the ground. Then the dynamics of u_1 and u_2 are given by

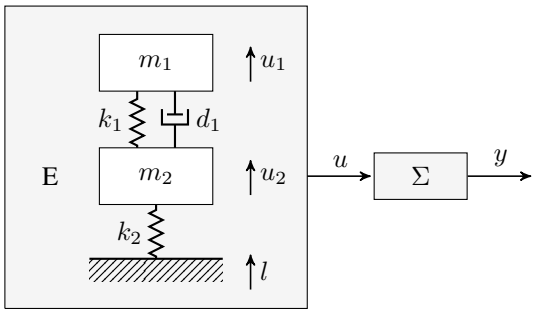

Fig. 4. Quarter car model of the environment.

$$m_1\ddot{u}_1 = -b(\dot{u}_1 - \dot{u}_2) - k_1(u_1 - u_2), \tag{9}$$

$$m_2\ddot{u}_2 = -b(\dot{u}_2 - \dot{u}_1) - k_1(u_2 - u_1) - k_2(u_2 - l). \tag{10}$$

Now, suppose that the vertical position of our component is controlled electronically on the basis of the positions of the body and the wheel. In other words, the input to our system is given by $u = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^\top$. The component, with vertical position y, will be devoid of vibrations if it has zero acceleration, hence the desired guarantees Γ are given by (7) with $G(s) = s^2$. If there is no information about the reference signal l, i.e., there is no information about the surface on which the car moves, then the available information about u is given by (9). Therefore, the assumptions A are given by (6) with $A(s) = \begin{bmatrix} m_1 s^2 + bs + k_1 & -bs - k_1 \end{bmatrix}$ and our specification is captured by the contract $\mathcal{C} = (\mathrm{A}, \Gamma)$.

An implementation of $\mathcal{C}$ is given by the system

$$\Sigma : \begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \frac{1}{m_1} \begin{bmatrix} b & -b \\ k_1 & -k_1 \end{bmatrix} u, \\ y = \begin{bmatrix} 1 & 0 \end{bmatrix} x + \begin{bmatrix} 1 & 0 \end{bmatrix} u. \end{cases}$$

Indeed, we can eliminate x from Σ to obtain

$$\tfrac{\mathrm{d}^2}{\mathrm{d}t^2} y = \tfrac{\mathrm{d}^2}{\mathrm{d}t^2} u_1 + \tfrac{b}{m_1} \tfrac{\mathrm{d}}{\mathrm{d}t}(u_1 - u_2) + \tfrac{k_1}{m_1}(u_1 - u_2)$$

where the right-hand side is zero for all $u \in \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$. Then

$$\mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma) = \left\{ y \in \mathcal{C}_p^\infty \;\middle|\; \tfrac{\mathrm{d}^2}{\mathrm{d}t^2} y = 0 \right\},$$

and thus $\mathfrak{B}_{\mathrm{o}}(\mathrm{A} \wedge \Sigma) = \mathfrak{B}_{\mathrm{o}}(\Gamma)$, which shows that Σ is an implementation of $\mathcal{C}$ because of Theorem 3.

Alternatively, if there is information about the reference signal l, then we can use that information to construct a contract which is easier to implement. For example, if we assume that $l = 0$, then (9) and (10) yield the assumptions

$$\mathrm{A}_0 : A_0\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big) u = 0,$$

where the polynomial matrix $A_0(s)$ is given by

$$A_0(s) = \begin{bmatrix} m_1 s^2 + bs + k_1 & -bs - k_1 \\ -bs - k_1 & m_2 s^2 + bs + k_1 + k_2 \end{bmatrix}.$$

Consequently, the contract $\mathcal{C}_0 = (\mathrm{A}_0, \Gamma)$ is implemented by

$$\Sigma_0 : \begin{cases} \dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x + \frac{1}{m_2} \begin{bmatrix} -b & b \\ -k_1 & k_1 + k_2 \end{bmatrix} u, \\ y = \begin{bmatrix} 1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 & 1 \end{bmatrix} u. \end{cases}$$

Indeed, eliminating x from Σ_0 yields

$$\tfrac{\mathrm{d}^2}{\mathrm{d}t^2}y = \tfrac{\mathrm{d}^2}{\mathrm{d}t^2}u_2 + \tfrac{b}{m_2}\tfrac{\mathrm{d}}{\mathrm{d}t}(u_2 - u_1) + \tfrac{k_1}{m_2}(u_2 - u_1) + \tfrac{k_2}{m_2}u_2,$$

where the right-hand side is zero for all $u \in \mathfrak{B}_{\mathrm{i}}(\mathrm{A}_0)$, hence $\mathfrak{B}_{\mathrm{o}}(\mathrm{A}_0 \wedge \Sigma_0) = \mathfrak{B}_{\mathrm{o}}(\Gamma)$. Note that Σ_0 does not implement $\mathcal{C}$ because the right-hand side of the latter is not necessarily zero for all $u \in \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$. However, we clearly have that $\mathfrak{B}_{\mathrm{i}}(\mathrm{A}_0) \subset \mathfrak{B}_{\mathrm{i}}(\mathrm{A})$, hence $\mathcal{C}$ refines $\mathcal{C}_0$ due to Theorem 4. This implies that Σ implements $\mathcal{C}_0$, and more generally, that any implementation of $\mathcal{C}$ is an implementation of $\mathcal{C}_0$. Of course, we already expected a component that works for arbitrary reference signal l to work for $l = 0$ in particular.

Finally, we might want to take into account different possibilities for the reference signal l. For example, we might want to consider reference signals that satisfy $\dot{l} = 0$ ("flat" road) and $\ddot{l} + l = 0$ ("wavy" road). Substituting these in equations (9) and (10) results in the assumptions

$$\mathrm{A}_1 : A_1\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)u = 0, \quad \mathrm{A}_2 : A_2\big(\tfrac{\mathrm{d}}{\mathrm{d}t}\big)u = 0,$$

with polynomial matrices $A_1(s)$ and $A_2(s)$ given by

$$A_1(s) = \begin{bmatrix} 1 & 0 \\ 0 & s \end{bmatrix} A_0(s), \quad A_2(s) = \begin{bmatrix} 1 & 0 \\ 0 & s^2 + 1 \end{bmatrix} A_0(s).$$

Then a component that works for a "flat" road needs to satisfy the contract $\mathcal{C}_1 = (\mathrm{A}_1, \Gamma)$, and a component that works for a "wavy" road needs to satisfy the contract $\mathcal{C}_2 = (\mathrm{A}_2, \Gamma)$. Therefore, a component that works for both roads needs to satisfy the conjunction $\mathcal{C}_1 \wedge \mathcal{C}_2 = (\mathrm{A}_1 \vee \mathrm{A}_2, \Gamma)$, which we have obtained using Theorem 5. Note that $\mathcal{C}$ refines both $\mathcal{C}_1$ and $\mathcal{C}_2$, while $\mathcal{C}_1$ and $\mathcal{C}_2$ both refine $\mathcal{C}_0$. Therefore, $\mathcal{C}$ refines the conjunction $\mathcal{C}_1 \wedge \mathcal{C}_2$, which in turn refines $\mathcal{C}_0$.

V. Conclusion

We have introduced assume-guarantee contracts for linear dynamical systems with inputs and outputs. We defined these as a pair of linear systems, called assumptions and guarantees, which were used to characterize the class of compatible environments and the class of implementations through the notion of system behaviour, and in particular, by inclusion of behaviours. In addition to defining contracts, we also characterized contract implementation and proposed a method for verifying it. Moreover, we characterized contract refinement by mirrored inclusions of behaviours of assumptions and guarantees. Using this, we provided an explicit characterization of contract conjunction in terms of the join of assumptions and meet of guarantees. We also demonstrated our setup and results with an illustrative example. Finally, we still require a suitable notion of contract composition. This would enable the component-based analysis and design of interconnected systems, hence its definition and characterization will be the focus of future work.

References

[1] J. C. Willems, "Dissipative dynamical systems part I: General theory," *Archive for Rational Mechanics and Analysis*, vol. 45, no. 5, pp. 321–351, 1972.

[2] F. Blanchini, "Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[3] B. Meyer, "Applying 'design by contract'," *Computer*, vol. 25, no. 10, pp. 40–51, 1992.

[4] C. Jones, "Specification and design of (parallel) programs.," in *Proceedings Of IFIP Congress '*, vol. 83, pp. 321–332, 1983.

[5] A. Chakrabarti, L. de Alfaro, T. A. Henzinger, and F. Y. C. Mang, "Synchronous and bidirectional component interfaces," in *Computer Aided Verification* (E. Brinksma and K. G. Larsen, eds.), pp. 414–427, Springer Berlin Heidelberg, 2002.

[6] L. de Alfaro and T. A. Henzinger, "Interface-based design," in *Engineering Theories of Software Intensive Systems* (M. Broy, J. Grünbauer, D. Harel, and T. Hoare, eds.), pp. 83–104, Springer Netherlands, 2005.

[7] A. Davare, D. Densmore, L. Guo, R. Passerone, A. L. Sangiovanni-Vincentelli, A. Simalatsar, and Q. Zhu, "metroII: A design environment for cyber-physical systems," *ACM Transactions on Embedded Computing Systems*, vol. 12, no. 1s, 2013.

[8] A. Sangiovanni-Vincentelli, W. Damm, and R. Passerone, "Taming dr. Frankenstein: Contract-based design for cyber-physical systems," *European Journal of Control*, vol. 18, no. 3, pp. 217–238, 2012.

[9] A. Benveniste, B. Caillaud, D. Nickovic, R. Passerone, J.-B. Raclet, P. Reinkemeier, A. Sangiovanni-Vincentelli, W. Damm, T. A. Henzinger, and K. G. Larsen, *Contracts for System Design*. Foundations and Trends in Electronic Design Automation, Now Publishers, 2018.

[10] J. C. Willems, "Models for dynamics," in *Dynamics Reported: A Series in Dynamical Systems and Their Applications* (U. Kirchgraber and H. O. Walther, eds.), pp. 171–269, Vieweg+Teubner Verlag, 1989.

[11] J. W. Polderman and J. C. Willems, *Introduction to Mathematical Systems Theory*. Springer-Verlag New York, 1998.

[12] A. Saoud, A. Girard, and L. Fribourg, "On the composition of discrete and continuous-time assume-guarantee contracts for invariance," in *Proceedings of the European Control Conference*, pp. 435–440, 2018.

[13] A. Saoud, A. Girard, and L. Fribourg, "Contract based design of symbolic controllers for interconnected multiperiodic sampled-data systems," in *Proceedings of the IEEE Conference on Decision and Control*, pp. 773–779, 2018.

[14] A. Eqtami and A. Girard, "A quantitative approach on assume-guarantee contracts for safety of interconnected systems," in *Proceedings of the European Control Conference*, pp. 536–541, 2019.

[15] E. S. Kim, M. Arcak, and S. A. Seshia, "A small gain theorem for parametric assume-guarantee contracts," in *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, HSCC '17, pp. 207–216, 2017.

[16] M. Al Khatib and M. Zamani, "Controller synthesis for interconnected systems using parametric assume-guarantee contracts," in *Proceedings of the American Control Conference*, pp. 5419–5424, 2020.

[17] B. Besselink, K. H. Johansson, and A. van der Schaft, "Contracts as specifications for dynamical systems in driving variable form," in *Proceedings of the European Control Conference*, pp. 263–268, 2019.

[18] F. Kerber and A. van der Schaft, "Assume-guarantee reasoning for linear dynamical systems," in *Proceedings of the European Control Conference*, pp. 5015–5020, 2009.

[19] F. Kerber and A. van der Schaft, "Compositional analysis for linear systems," *Systems & Control Letters*, vol. 59, no. 10, pp. 645–653, 2010.

[20] F. Kerber and A. J. van der Schaft, "Decentralized control using compositional analysis techniques," in *Proceedings of the IEEE Conference on Decision and Control and European Control Conference*, pp. 2699–2704, 2011.

[21] J. C. Willems, "The behavioral approach to open and interconnected systems," *IEEE Control Systems Magazine*, vol. 27, no. 6, pp. 46–99, 2007.

[22] J. C. Willems, "Input-output and state-space representations of finite-dimensional linear time-invariant systems," *Linear Algebra and its Applications*, vol. 50, pp. 581–608, 1983.

[23] J. W. Polderman, "A new and simple proof of the equivalence theorem for behaviors," *Systems & Control Letters*, vol. 41, no. 3, pp. 223–224, 2000.

[24] B. M. Shali, A. J. van der Schaft, and B. Besselink, "Behavioural contracts for linear dynamical systems: input assumptions and output guaranteess," *arXiv e-prints*, p. arXiv:2103.12646, 2021.

[25] T. Kaczorek, *Polynomial and Rational Matrices*. Springer-Verlag London, 2007.