2014 European Control Conference (ECC)
June 24-27, 2014. Strasbourg, France

Application of Contract-based verification techniques for Hybrid Automata to Surgical Robotic Systems

Luzie Schreiter[1], Davide Bresolin[2], Marta Capiluppi[3], Joerg Raczkowsky[1], Paolo Fiorini[3] and Heinz Woern[1]

Abstract— **Surgical robotic systems have to deliver a high quality of safety, since they deal with human lives. Their safety specifications must ensure the absence of risks for the patient and the operating room staff. To respect the modular nature of a surgical system, we propose a contract based verification approach for safety. We introduce a case study based on a typical surgical robotic operation scenario and model its components by using hybrid automata. We exploit the theory of parallel composition of contracts to verify properties on each component and prove that the property of the overall system can be obtained by composition.**

Index Terms— **Safety Critical Systems, Hybrid Systems, Robotics**

I. INTRODUCTION

Surgical robotics is a technology where the robot helps the surgeons in achieving basic and iterative tasks during a surgical intervention. It was first introduced in the 1980's for neurosurgical intervention. In the last decades the application of surgical robots has been extended to other surgical areas, for instance prostatectomy, orthopaedic and even soft tissues surgery.

Surgical robots design aims at maximizing the cooperation between the medical staff (surgeons and nurses) and the surgical robotic system to perform the surgical intervention, while, of course, keeping in mind the interaction with the patients in a shared way of understanding and communication. For these reasons, one of the most important aspects of a surgical intervention, with the use of robots, is the safety aspect of man-machine interaction. Hence, safety verification is currently one of the urgent research topics for surgical robotic platforms in general. It includes the application of methods, procedures, tests and other evaluations, to determine whether a system is or has been operating as intended.

The aim of our work is to define a procedure for verifying safety in a surgical robotic system that is able to deal with the issues created by a medical context. Since safety is related to redundancy and component replacement, we start from an architectural design where every part of the system is as simple as possible. In this approach the design process is seen as an assembly of components, represented in terms of assumptions about their environment and guarantees about their behavior. The pair assumptions-guarantees is usually called the "contract" that the component must respect. In this way, the requirements of the whole system can be obtained by composition from simpler requirements that, if satisfied, guarantee that the overall requirement is satisfied. Every component is then verified separately and, if all components satisfy their "local" requirements, we can conclude that the whole system is correct [1]. The notion of contract has been first introduced to verify computer programs [2] and digital circuits [3]. Following the first seminal approaches, contract-based design and verification has been applied to a number of different formalisms for concurrent systems, ranging from discrete automata [4] to process algebra [5]. More recently, contract-based design has gained a strong interest from the embedded systems community, due to its capability of handling systems with heterogenous components, mixing discrete and continuous behaviour, in a natural way [6], [7].

We believe that surgical robotic systems are another example of heterogeneous systems where a contract-based verification approach can provide more accurate and more reliable answers compared to traditional monolithic approaches. From a more mathematical point of view, surgical robotic systems can be modeled as hybrid systems [8], i.e. systems where discrete events and continuous dynamics interact. Indeed, robotic systems have been represented as hybrid systems in different frameworks, even for verification of safety and reachability properties, e.g. in [9]. Therefore, we decided to model the components of the surgical robotic system under consideration as hybrid automata [8], a well-studied formalism for which a strong compositionality theory exists [10], and for which contract based verification methodologies have been recently developed by the scientific community [11], [12].

In this paper a simple case study of a surgical robotic system is presented, where a manipulator has to reach a desired target. The system under consideration operates with limited human intervention in a changing environment. We assume that medical and human-machine interaction aspects are not introduced in this work. The focus of it is to find a way to formalize surgical sub-procedures with the corresponding equipment and to verify this model by selected

*This research has been supported by the coordination action EuroSurge (grant no. 288233) funded by the European Commission in the 7th EC framework program.

[1]L. Schreiter, J. Raczkowsky and H. Woern are with Institute of Process Control and Robotics, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany {luzie.schreiter, joerg.raczkowsky, woern}@kit.edu

[2]D. Bresolin is with Department of Computer Science and Engineering, University of Bologna, 40126 Bologna, Italy davide.bresolin@unibo.it

[3]M. Capiluppi and P. Fiorini are with Department of Computer Science, University of Verona, 37134 Verona, Italy {marta.capiluppi, paolo.fiorini}@univr.it

properties. To this end and as specified above, we use hybrid automata to model each component of the system, with the aim of preserving the modularity of the setup, which is required in a surgical robot. We, then, verify the correctness of some desired properties of the system. Composability is achieved by using the composition rules of contracts and their relation with mathematical logic [11]. We start from the assumption that safety is related to reachability, in the sense that the system has to perform as specified even under the occurrence of faults and component changes, without reaching any unsafe region. Indeed, we perform the analysis of contracts for each component of the presented case study and verify that their composition satisfies the contract of the overall system, hence satisfying safety specifications.

This work is an extension of [13], where the method used for verification of components was based on invariants. Indeed the contract-based procedures are more automatic and introduce the possibility of an implementation, which will be under study in future works. The main difference of the scenario presented in the case study, is given by the fact that in this paper we introduce a fault, making the switching between component subject to an unpredictable event, while in [13] the event occurrence was known and desired.

At the best of our knowledge, many approaches for contract-based verification exist, based on hybrid models, such as the ones cited above. Nevertheless, our method is based on hybrid input/output automata (HIOAs) [10], where the external behavior is explicit in the sense that some variables and actions are specialized to be used as input/output signals for communication between automata. We exploit the characteristics of this model to prove the fulfillment of the contracts for each component, while using a bottom-up approach. Indeed the contracts for each component are designed separately, since each component has to achieve its particular goal (related to the safety of the component itself). Although we use the same procedure for composition of contracts of [11], our approach starts from a different perspective. We aim at composing the contracts to obtain a general contract that is equivalent or a refinement of the specifications for the whole system. In [11] the focus is on the correct decomposition of contracts, while for us it is on the correct composition. Moreover, surgical robots have not been studied in this framework, while being an interesting application for it. The main reason why we decided to follow a bottom-up approach is that in surgical systems we usually know what are their components and what are the specifications that each component has to fulfill.

The paper is structured as follows: In section II we introduce the case study under consideration, in section III we will recall the theoretical aspects of contract based design and the hybrid automata framework, in section IV we present the modeling of the case study with hybrid automata, in section V the contracts for each component and the overall system are presented, in section VI we show how to verify each component contract, in section VII we prove that the composition of the contracts satisfies the overall desired contract, finally we conclude with some remarks.

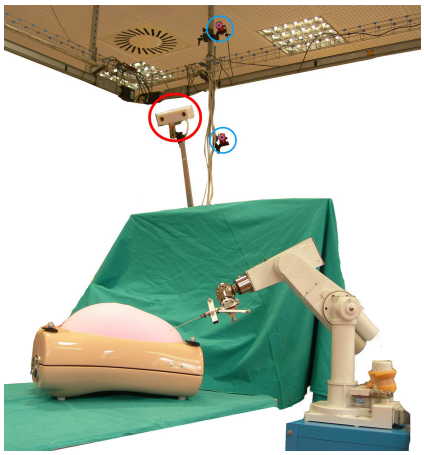

Fig. 1. Real setup of the case study at AltairLab, Verona, Italy.

II. CASE STUDY

To prove the effectiveness of the considered procedure, we apply it to a simplified version of the setup described in [9]. We consider a system composed by a manipulator holding a puncturing tool and moving freely in a space. We suppose that the end effector of the manipulator has to reach a target, that is the patient tissue. We only describe the movement from the stand-by position to the tissue. The case study aims at representing the fact that a robot has to be tracked to have a global control of the movements needed to assure a safe interaction with humans during a surgical operation. A possible real setup is presented in Fig. 1.

We describe the following situation. The robot moves freely in the surgical room during the setup procedure and its position is monitored by a tracker that controls the entire room. The robot approaches the patient to a target point at the patients body area called A.

We simulate a tracker failure. To deal with it, without losing the main functionality of the system, we use a redundant backup tracker, which is less powerful, having a bigger sampling time. When the fault occurs, the system has to switch between the two trackers and adapt to the new parameters. To this end, the control policy will slow down the speed of the robot, in order to cope with the new sampling time. We suppose that a new reference will be generated and the controller parameters will be adapted to it. We assume that only the first tracker can fail, while the backup tracker is absolutely reliable until the end of the task. We also assume that the fault can only occur before the manipulator reaches a certain position defined as ”too near” to the target.

The system is composed by a manipulator with a tool on the tip, i.e. a needle. The manipulator is controlled by a control board, based on the sampled position given by a tracker following a marker on the tip. The tracker is composed by a set of cameras monitoring the entire room (blue circles in Fig. 1). When this tracker fails, a backup tracker substitutes it (red circles in Fig. 1), less powerful and focussed on the tip. The setup is monitored by a software supervisor that gives the performance and safety

specifications for the overall system, even in case of fault.

The aim of the verification procedure is to check whether the system recognizes that a fault has occurred, i.e. that the tracker has changed. Indeed, after the fault, it is necessary to verify that it is still able to reach the desired target and that this is done in a safe way, adapting the parameters of each component to the new situation. Moreover, since we aim at keeping the modular nature of the system and its reliability, we want the verification procedure to be able to verify the properties of each component separately, and to prove that the composition still respects the requirements.

III. CONTRACT-BASED MODELS

In this section we recall the theory of contract-based models and hybrid automata.

A hybrid automaton is a finite state machine enriched with continuous dynamics labelling each discrete state (or *location*). Our definition of hybrid automaton extends the one given in [8] to support the compositional description and analysis of systems, when they are too complex to be understood all at once and must be decomposed. To this end, we introduce a distinction between input, output and internal variables and actions as in [10]. Input, output variables and events describe the continuous and discrete interactions of a component with the environment and the other components of the system, thus defining how components can be put together to make complex systems out of simpler ones. Evolution of a hybrid automaton alternates *continuous* and *discrete* steps. In a continuous step, the location does not change, while the continuous variables change following the continuous dynamics of the location. A discrete evolution step consists of the activation of a *discrete transition* that can change both the current location and the value of the state variables, in accordance with the reset function associated to the transition. The interleaving of continuous and discrete evolutions is decided by the *invariant* of the location, which must be true for the continuous evolution to keep on going, and by the *guard* predicates, which must be true for a discrete transition to be activated.

A contract is an assertion on the behaviors of a component subject to certain assumptions [7]. An assertion is a property that may or may not be satisfied by a behavior, where a behavior is a run of a hybrid automaton, i.e. the alternation of continuous and discrete steps. A contract is then defined as the pair $C = (A, G)$, where A is the assumption and G the guarantee. A behavior satisfies the contract when it satisfies the guarantee subject to the assumption.

Parallel composition of contracts formalizes how contracts related to different components have to be combined to represent the contract for the whole system, and is defined as in [7].

Parallel composition *Let* $C_1 = (A_1, G_1)$ *and* $C_2 = (A_2, G_2)$ *be contracts. The parallel composition* $C = (A, G) = C_1 \parallel C_2$ *is given by:*

$$A = (A_1 \cap A_2) \cup \neg(G_1 \cap G_2)$$

$$G = (G_1 \cap G_2)$$

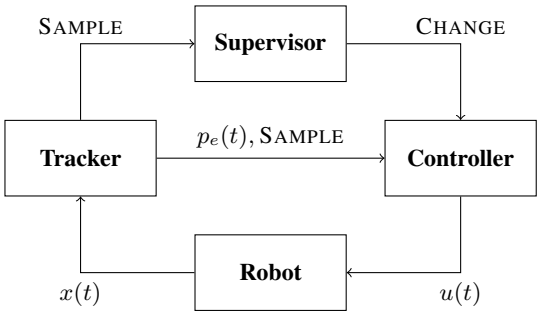

Fig. 2. Overview of the model for the Case Study

Contract semantics can be defined as implementations and environments [11], i.e. set of traces. In more detail, let I and E be sets of traces, we say that I is an implementation satisfying C iff $I \cap A \subseteq G$, and E is an environment satisfying C iff $E \subseteq A$. A hybrid automaton H naturally defines an implementation $\mathcal{I}(H)$, corresponding to all possible behaviors of the automaton. Hence, we say that a hybrid automaton H satisfies a contract C if and only if $\mathcal{I}(H) \cap A \subseteq G$.

If we define $\mathcal{I}(C)$ the implementations and $\mathcal{E}(C)$ the environments of a contract C, we say that a contract C' refines (is stronger than) a contract C ($C' \leq C$) iff $\mathcal{I}(C') \subseteq \mathcal{I}(C)$ and $\mathcal{E}(C) \subseteq \mathcal{E}(C')$. The following theorem is valid.

Theorem 1 ([11]): $(A', G') \leq (A, G)$ iff $A \subseteq A'$ and $G' \cap A \subseteq G$.

IV. MODELING OF THE CASE STUDY

In this section we describe the formal model of the case study, which will be used in the following sections to verify the correctness of the robotic manipulator system. Figure 2 gives an overview of the whole model and describes the data and signal flow between the four components. We decided to model each component separately as a hybrid automaton and to verify each component separately, using the compositional theory of contracts introduced in section III. This will help the designer to illustrate and to respect the modularity of the set-up, which is very important in surgical systems, for safety reasons. Moreover, the decomposition will allow the user to change the components without effects on the reliability of the overall system.

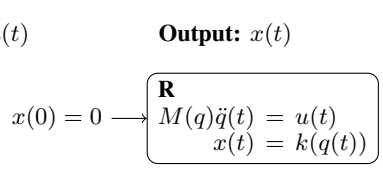

Fig. 3. Model of component Robot

Fig. 3 represents the component Robot. It is modeled by a continuous-only system with dynamics $M(q)\ddot{q}(t) = u(t)$, where q is the vector of generalized coordinates, related to the position and orientation of the end effector by the direct kinematics function $x = k(q)$, and $u(t)$ is the input torque given to the joints following the control law computed by

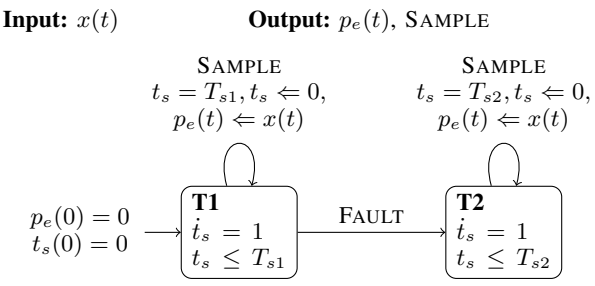

Fig. 4. Model of component Tracker

the component Controller. The robot manipulator sends the end-effector position $x(t)$ to the Tracker component, which monitors it.

The component Tracker is represented in Fig. 4. We choose to model the two tracker as a hybrid system with two locations: T1 represents the first tracker that is used in nominal conditions, while T2 represents the second tracker that is used in case of fault of the first one. The switching from one tracker to the other is activated by the internal (and thus hidden to the other components) action FAULT. The input of the Tracker is the current position $x(t)$ sent by the robot. T1 and T2 initialize an internal clock variable t_s and update the output variable $p_e(t)$ with the current marker position with sampling time T_{s1} and T_{s2}, respectively (we assume there is no tracking error). At each sampling time, the Tracker generates the output event SAMPLE to synchronize with the components Controller and Supervisor.

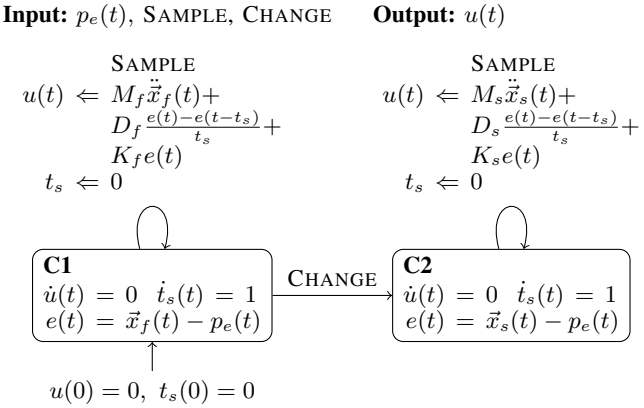

Fig. 5. Model of component Controller

The Controller is a discrete-time system, and it is represented in Fig. 5. The controller stores in the internal variable $e(t)$ the value of the error between the reference trajectory and the position of the marker sent by the Tracker. At each SAMPLE, the new value for the output torque $u(t)$ is computed, and then kept constant until the next sampling time. In analogy with the Tracker, the Controller has two different modes of operation: in location C1 the control objective has to follow a fast reference trajectory $\vec{x}_f$, under the assumption that the Tracker is in nominal conditions. The output torque $u(t)$ is computed using a PID controller with parameters

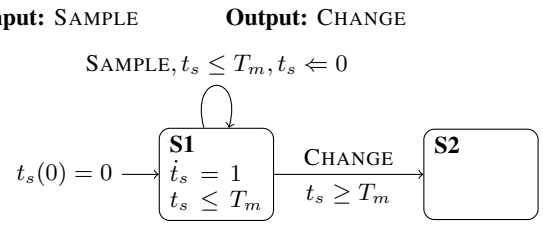

Fig. 6. Model of component Supervisor

K_f, M_f, and D_f. In location C2 the reference trajectory is replaced by a slower one $\vec{x}_s$, and the PID controller parameters K_s, M_s, and D_s are changed accordingly. The assumption is that in location C2 the Controller is able to follow the slower reference trajectory even when the Tracker is faulty. Since the occurrence of the fault is hidden to the Controller, the switching between the two control laws is activated by the special input action CHANGE, generated by the Supervisor.

The component Supervisor is shown in Fig. 6. The role of the supervisor is to determine whether the Tracker is faulty or not. To this end, it monitors the time elapsing between two successive occurrences of the action SAMPLE: if this time is less or equal to a given threshold T_m, the Tracker is assumed to behave normally (location S1). As soon as the sampling time becomes greater than T_m, the Tracker is assumed to be faulty, and the action CHANGE is sent to the Controller (location S2). Notice that the value of the threshold can be, in principle, independent from the Tracker's sampling times T_{s1} and T_{s2}. Clearly, not all values of T_m guarantee the correct operation of the system. The safe values for T_m are specified by the contracts described in the next section.

V. CONTRACTS FOR THE CASE STUDY

In this section we define the global contract for the robotic manipulator system we want to verify, and we will show how the main contract is related to the components contracts.

Before entering into the details, we need to introduce some notation. For any discrete action ACT and natural number $n \geq 1$, we will denote with $\text{ACT}(n)$ the time instant t of the n-th occurrence of ACT. If no n-th occurrence of ACT exists, then $\text{ACT}(n) = +\infty$.

The overall requirement for the complete system is that the end-effector of the manipulator follows the reference trajectory with a given bound on the error, even after the occurrence of a fault in the tracker. This can be formally expressed as follows.

Contract 1 (Main Contract):

A: $x(0) = \vec{x}_f(0)$, $T_{s1} < T_m < T_{s2}$

G: $\forall t\big(t \leq \text{FAULT}(1) \rightarrow \|x(t) - \vec{x}_f(t)\| < \varepsilon\big)$
$\forall t\big(t \geq \text{FAULT}(1) + T_{s2} \rightarrow \|x(t) - \vec{x}_s(t)\| < \varepsilon\big)$

In Contract 1, the constraints on T_{s1}, T_m and T_{s2} are necessary to guarantee that the supervisor can identify the fault. The guarantee states that the end-effector should follow the reference trajectory $\vec{x}_f$ before the fault, and the reference trajectory $\vec{x}_s$ after one sampling period from the fault.

The contracts for the components specify the correct behavior of every subsystem. In section VII we will show that their composition implies Contract 1.

We first consider the contract for the component Tracker: under no assumptions, it guarantees that the Tracker returns the correct positions $p_e(t)$, and that the sampling time is T_{s1} in nominal condition and T_{s2} in faulty condition, respectively. This is reported in Contract 2.

Contract 2 (Tracker):

A: $T_{s1} \leq T_{s2}$

G: $\forall n, t\big(t = \textsc{Sample}(n) \rightarrow p_e(t) = x(t)\big)$
$\forall n\big(\textsc{Sample}(n) \leq \textsc{Fault}(1) \rightarrow$
$\textsc{Sample}(n) - \textsc{Sample}(n-1) = T_{s1}\big)$
$\forall n\big(\textsc{Sample}(n) > \textsc{Fault}(1) \rightarrow$
$\textsc{Sample}(n) - \textsc{Sample}(n-1) = T_{s2}\big)$

The contract 3 for the Supervisor assumes that the sampling time is either T_{s1} or T_{s2}, and that $T_{s1} < T_m < T_{s2}$, and it guarantees that CHANGE occurs as soon as the sampling time switches from T_{s1} to T_{s2}.

Contract 3 (Supervisor):

A: $T_{s1} < T_m < T_{s2}$
$\forall n\big(\textsc{Sample}(n) - \textsc{Sample}(n-1) \in \{T_{s1}, T_{s2}\}\big)$

G: $\forall n\big(\textsc{Sample}(n+1) - \textsc{Sample}(n) = T_{s2} \rightarrow$
$\textsc{Sample}(n+1) > \textsc{Change}(1)\big)$

To conclude this section we give a contract defining the correct behavior of the composition of the Controller with the Robot. We chose to give a single contract for the two components, instead of two separate contracts, because the design of a controller strictly depends on the actual plant to be controlled, and thus the achievement of control objectives can be guaranteed only when the controller is paired with the correct plant.

In our case, the contract for the Controller and the Robot assumes that the initial position of the robot is equal to $\vec{x}_f(0)$, that the input positions $p_e(t)$ (sent from the Tracker) correspond to the actual positions of the end-effector at the sampling times, that the sampling time is respectively T_{s1} before CHANGE and T_{s2} afterwards, and it guarantees that the manipulator follows the reference trajectories $\vec{x}_f$ and $\vec{x}_s$ with an error that is bounded by some constant ε.

Contract 4 (Controller‖Robot):

A: $x(0) = \vec{x}_f(0)$, $\forall n, t\big(t = \textsc{Sample}(n) \rightarrow p_e(t) = x(t)\big)$
$\forall t, n\big(\textsc{Sample}(n) < \textsc{Change}(1) \rightarrow$
$\textsc{Sample}(n) - \textsc{Sample}(n-1) = T_{s1}\big)$
$\forall t, n\big(\textsc{Sample}(n) > \textsc{Change}(1) \rightarrow$
$\textsc{Sample}(n) - \textsc{Sample}(n-1) = T_{s2}\big)$

G: $\forall t\big(t < \textsc{Change}(1) \rightarrow \|\vec{x}_f(t) - x(t)\| < \varepsilon\big)$
$\forall t\big(t > \textsc{Change}(1) \rightarrow \|\vec{x}_s(t) - x(t)\| < \varepsilon\big)$

VI. VERIFICATION OF THE CONTRACTS

Once the contracts for the different components of the system have been defined, it is necessary to check that the actual implementation of the components respects them. A number of different formal and semi-formal techniques can be used at this stage, like logical reasoning, model checking, testing or simulation. One of the advantages of a compositional approach is that every component can be verified using the most appropriate method. Then, the different results are combined by composing the contracts to obtain an answer to the global verification problem, as we will show in the next section.

The considered case-study is simple enough to allow the contracts to be verified directly by some reasoning on the model of the components, with the exception of Contract 4. We start the verification from Contract 2 for the Tracker. It is easy to see that the model depicted in Fig. 4 is such that:

1) at each SAMPLE, the value of $p_e(t)$ became equal to $x(t)$;
2) when the tracker is in location **T1**, a new SAMPLE is generated exactly every T_{s1} time units;
3) when the tracker is in location **T2**, a new SAMPLE is generated exactly every T_{s2} time units;
4) the tracker stays in **T1** until the FAULT occurs, and then switches to **T2**.

Hence, the model of the Tracker satisfies Contract 2.

Consider now the model of the Supervisor depicted in Fig. 6. The behavior of the component is the following one:

1) it starts from location **S1**;
2) if the time between two SAMPLE is less than T_m, it remains in location **S1**;
3) as soon as the time between two SAMPLE is greater than T_m, it switches to location **S2** and generates a CHANGE;
4) it remains in **S2** forever.

Hence, under the assumption that $T_{s1} < T_m < T_{s2}$, and that the time elapsing between two SAMPLE is either T_{s1} or T_{s2}, we have that the Supervisor satisfies the guarantee of Contract 3.

To conclude this section we have to show that the composition of the Controller with the Robot satisfies Contract 4. In this case, it is not possible to use a general reasoning argument. Indeed, to prove that the Controller can force the Robot to follow the reference trajectories it is necessary to know at least:

- the actual dynamics $M(q)\ddot{q}(t)$ and kinematics function $k(q)$ of the Robot;
- the exact values of the sampling times T_{s1} and T_{s2};
- the parameters K_f, D_f, M_f and K_s, D_s, M_s of the two PID controllers.

Since we left those values unspecified, in this paper we do not enter into the details of this proof of satisfaction, and we simply assume that Contract 4 is satisfied by the parallel composition of the Robot with the Controller. In a real scenario, where all the details on the actual robot and controller are given, Contract 4 can be verified in many different ways. A formal approach would be to use a controller design methodology that guarantees the required properties by construction, or to use automatic verification tools to prove that the contract is respected [12], [9]. When the component is too complex to be formally verified, extensive simulation campaigns can be used to tune the PID parameters and to achieve at least a good degree of confidence that the contract is respected.

VII. COMPOSITION OF CONTRACTS

In this section we will show how to compose the contracts for the four components of the model to obtain the Main Contract (Contract 1). Together with the proofs of satisfaction of the contracts given in section VI, this result will allow us to conclude that the model of the robotic manipulator respects Contract 1, and thus that it is able to follow the reference trajectory within the given error bounds (see the rules in section III).

The contract satisfied by the composition of the Tracker with the Supervisor is given by Contract 5. When composing Contract 2 with Contract 3 we can drop the assumption $\forall n\big(\textsc{Sample}(n) - \textsc{Sample}(n-1) \in \{T_{s1}, T_{s2}\}\big)$, since it is a consequence of the guarantee of Contract 2. Moreover, since the sampling time is T_{s1} before $\textsc{Fault}(1)$ and T_{s2} afterwards, and since CHANGE is activated as soon as the sampling time switches to T_{s2}, we can conclude that $\textsc{Fault}(1) < \textsc{Change}(1) < \textsc{Fault}(1) + T_{s2}$.

Contract 5 (Tracker||Supervisor):
A: $T_{s1} < T_m < T_{s2}$
G: $\forall n, t\big(t = \textsc{Sample}(n) \rightarrow p_e(t) = x(t)\big)$
$\forall n\big(\textsc{Sample}(n) \leq \textsc{Fault}(1) \rightarrow \textsc{Sample}(n) - \textsc{Sample}(n-1) = T_{s1}\big)$
$\forall n\big(\textsc{Sample}(n) > \textsc{Fault}(1) \rightarrow \textsc{Sample}(n) - \textsc{Sample}(n-1) = T_{s2}\big)$
$\textsc{Fault}(1) < \textsc{Change}(1) < \textsc{Fault}(1) + T_{s2}$

When composing Contract 5 with Contract 4 of Controller || Robot, we have that the assumption $\forall n, t\big(t = \textsc{Sample}(n) \rightarrow p_e(t) = x(t)\big)$ can be dropped: it is included in the guarantees of Contract 5. Moreover, since $\textsc{Fault}(1) < \textsc{Change}(1) < \textsc{Fault}(1) + T_{s2}$ we have that the sampling time is T_{s1} before $\textsc{Change}(1)$ and T_{s2} afterwards, and thus also the assumptions on the sampling times of Contract 4 can be removed.

Contract 6 (Tracker||Supervisor||Controller||Robot):
A: $x(0) = \vec{x}_f(0)$, $T_{s1} < T_m < T_{s2}$
G: $\textsc{Fault}(1) < \textsc{Change}(1) < \textsc{Fault}(1) + T_{s2}$
$\forall t\big(t < \textsc{Change}(1) \rightarrow \|\vec{x}_f(t) - x(t)\| < \varepsilon\big)$
$\forall t\big(t > \textsc{Change}(1) \rightarrow \|\vec{x}_s(t) - x(t)\| < \varepsilon\big)$

Since $\textsc{Fault}(1) < \textsc{Change}(1) < \textsc{Fault}(1) + T_{s2}$ we have that the guarantees of Contract 6 imply the guarantees of the Main Contract 1. Hence, Contract 6 is a stricter contract with respect to Contract 1 (see Theorem 1).

From the results of section VI we know that every component satisfies the corresponding contract. Hence, we can conclude that the model of the case study satisfies Contract 6, which in turns implies that the Main Contract 1 is satisfied, and thus that the end-effector of the manipulator can follow the reference trajectory with the required bound on the error, even after the occurrence of a fault in the tracker.

VIII. CONCLUSIONS

In this paper a simple case study based on a surgical robotic system was presented, where a manipulator has to reach a desired target. We used hybrid automata to model each component of the system, with the aim of preserving the modularity of the setup, which is required in a surgical system. Moreover we exploited a contract-based verification approach to verify the correctness of some desired safety properties of the system. To this end, we performed the analysis of contracts for each component of the presented case study and verified that their composition satisfies the contract of the overall system, hence satisfying the specifications.

Future work involves mainly the automation of the described flow into a suitable toolchain. To this end we are considering the use of temporal logic to specify the contracts [14], [15] and of reachability analysis tools to verify the contracts [12], [16]. From the application point of view, we aim to extend the case study involving the medical knowledge and human-interaction aspects.

REFERENCES

[1] G. Gössler, S. Graf, M. Majster-Cederbaum, M. Martens, and J. Sifakis, "An approach to modelling and verification of component based systems," *SOFSEM 2007: Theory and Practice of Computer Science*, pp. 295–308, 2007.

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

[3] D. L. Dill, *Trace theory for automatic hierarchical verification of speed-independent circuits*, ser. ACM distinguished dissertations. MIT Press, 1989.

[4] L. de Alfaro and T. A. Henzinger, "Interface automata," *SIGSOFT Softw. Eng. Notes*, vol. 26, no. 5, pp. 109–120, Sept. 2001.

[5] R. Negulescu, "Process spaces," in *Proceedings of the 11th International Conference on Concurrency Theory*, ser. CONCUR '00. London, UK, UK: Springer-Verlag, 2000, pp. 199–213.

[6] L. Benvenuti, A. Ferrari, E. Mazzi, and A. Sangiovanni Vincentelli, "Contract-based design for computation and verification of a closed-loop hybrid system," in *Hybrid Systems: Computation and Control*, ser. LNCS, vol. 4981, 2008, pp. 58–71.

[7] A. Benveniste, B. Caillaud, A. Ferrari, L. Mangeruca, R. Passerone, and C. Sofronis, "Multiple viewpoint contract-based specification and design," in *Formal Methods for Components and Objects*, ser. LNCS, vol. 5382. Springer, 2008, pp. 200–225.

[8] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," *THEORETICAL COMPUTER SCIENCE*, vol. 138, pp. 3–34, 1995.

[9] R. Muradore, D. Bresolin, L. Geretti, P. Fiorini, and T. Villa, "Robotic surgery: Formal verification of plans," *IEEE Robotics and Automation Magazine*, vol. 18, no. 3, pp. 24–32, 2011.

[10] N. Lynch, R. Segala, and F. Vaandrager, "Hybrid I/O automata," *Information and Computation*, vol. 185, no. 1, pp. 105 – 157, 2003.

[11] A. Cimatti and S. Tonetta, "A property-based proof system for contract-based design," in *Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on*, sept. 2012, pp. 21 –28.

[12] L. Benvenuti, D. Bresolin, P. Collins, A. Ferrari, L. Geretti, and T. Villa, "Assume-guarantee verification of nonlinear hybrid systems with ARIADNE," *International Journal of Robust and Nonlinear Control*, vol. 24, no. 4, pp. 699–724, 2014.

[13] M. Capiluppi, L. Schreiter, P. Fiorini, J. Raczkowsky, and H. Woern, "Modeling and verification of a robotic surgical system using hybrid input/output automata," in *Proceedings of the 12th biannual European Control Conference*, 2013.

[14] D. Bresolin, "Improving hyltl model checking of hybrid systems," in *Proc. of GandALF 2013*, ser. EPTCS, vol. 119. Open Publishing Association, 2013, pp. 79–92.

[15] A. Cimatti, M. Roveri, and S. Tonetta, "Requirements validation for hybrid systems," in *Proc. of CAV 2009*, ser. LNCS, vol. 5643. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 188–203.

[16] G. Frehse, C. Le Guernic, A. Donzé, S. Cotton, R. Ray, O. Lebeltel, R. Ripado, A. Girard, T. Dang, and O. Maler, "SpaceEx: Scalable verification of hybrid systems," in *Proc. of CAV 2011*, ser. LNCS, vol. 6806. Springer Berlin / Heidelberg, 2011, pp. 379–395.