TRACEBOT

Definition of the conceptual and reasoning framework and semantic models

Deliverable 5.1

Deliverable Title	D5.1 Definition of the conceptual and reasoning framework and semantic models.
Deliverable Lead:	University of Bremen (UOB)
Related Work Package:	WP5: Traceable Semantic Twin: Planning, reasoning, Audit Trail
Related Task(s):	T5.1: Definition of the domain process structure and taxonomy T5.2: Replication of medical lab environments into Traceable Semantic Twin Knowledge Bases for Reasoning T5.3: Traceability-aware process (re-)planning, reasoning and regulatory digital audit trail
Author(s):	Prof. Michael Beetz
Dissemination Level:	Public
Due Submission Date:	12/31/2021
Actual Submission:	12/21/2021
Project Number	101017089
Instrument:	Research and innovation action
Start Date of Project:	01.01.2021
Duration:	51 months
Abstract	This deliverable describes the concept of the reasoning apparatus with semantic models for the TraceBot project. The key approach in the reasoning framework is the usage of a hybrid knowledge-based approach which is powered by a Semantic Digital Twin for sterility testing use cases.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089), 2021-2025

Versioning and Contribution History

Version	Date	Modified by	Modification reason
v.01	29.11.2021	Prof. Michael Beetz(UOB)	Initial version
v.02	06.12.2021	Prof. Michael Beetz(UOB)	Ready for internal revision
v.03	14.12.2021	Prof. Michael Beetz(UOB)	Improvements based on internal feedback
v.04	20.12.2021	Prof. Michael Beetz(UOB)	Revised version ready for submission

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

Table of Contents

Horizon 2020
TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

1 Executive Summary

The goal of this deliverable is to introduce Traceable Semantic Twins (TSTs) and propose a conceptual and reasoning framework for the sterility testing domain. A TST is a methodology for checking assembly operations in task-critical applications such as medical and laboratory applications.

In sum, the TST will provide the robot with relevant information to perform the manipulation tasks involved in the TraceBot scenario with more robustness, efficiency and flexibility. In order to provide such information, the TST relies on versatile reasoning on a hybrid knowledge base with semantic models of the TraceBot environment namely a subsymbolic knowledge base made up of a photo-realistic and physics-faithful virtual TraceBot environment and a symbolic knowledge base principally made up of the environment ontology and grounded in the subsymbolic knowledge base for advanced reasoning. This advanced reasoning builds on three core mechanisms. The question-answering mechanism acts as interface between the robot and the TST, where the robot issues underdetermined queries (e.g., pick up the canister) to the TST and the latter reasons and returns a corresponding motion plan, parameterized in a context-specific way.

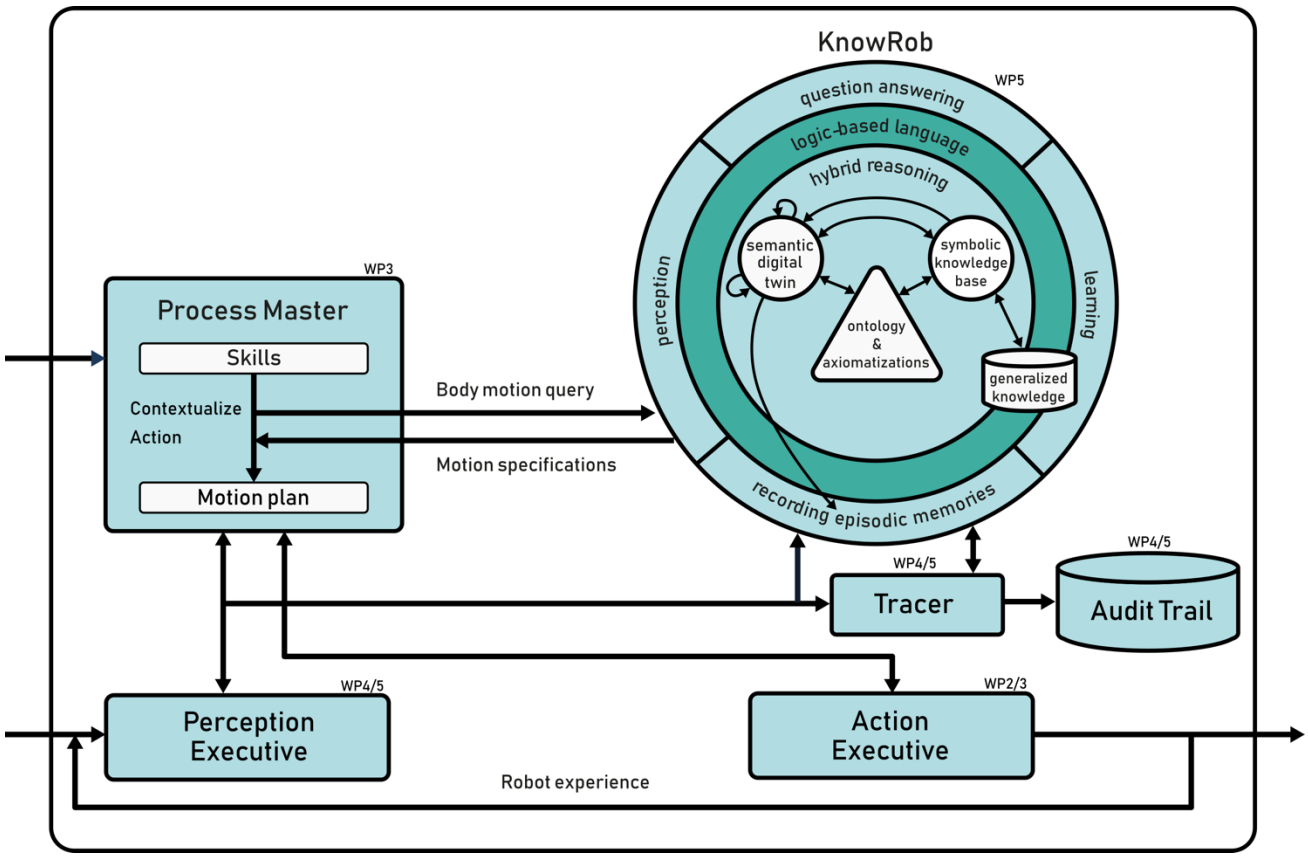


FIGURE 1: OVERVIEW OF THE CONCEPTUAL AND REASONING FRAMEWORK OF THE TST IN THE TRACEBOT ECOSYSTEM.

Though the ongoing reasoning is fundamentally symbolic and based on formal logics, embodied simulation is the second core mechanism used to enhance reasoning, where the robot mentally

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

performs the actions and evaluate the effects of the actions before actually engaging in the real environment. The third core mechanism is coined as imagistic reasoning, where the robot attempts to imagine how the scene would look like if certain actions were performed or to derive the causes that might lead to certain scene images. At the heart of benefits of these two mechanisms is anticipation, which is a cornerstone of cognition. Finally, the TST maintains episodic memories of the ongoing TraceBot processes at a symbolic level (i.e., actions, objects, chronology) as well as at a subsymbolic level (i.e., sensor & motor data) for the sake of generating audit trails, which themselves are chronological sets of records providing evidences that the processes were compliant with medical sterility test regulations: These episodic memories are coined in the research community as NEEMs (Narrative-Enabled Episodic Memories).

2 Introduction

Designing robotic agents to perform medical sterility tests raises at least two great challenges. On the one hand, the robots should cope with the complexity of the environment. Such complexity encompasses for instance the realtimeness of processes (e.g., pouring a precise quantity of liquid into the canister and reacting as fast as possible to slip) as well as the handling of involved quantities, objects and orifices such as needles, tubes, drops, fluids, which are usually articulated and tiny, therefore very difficult to reason in realtime about their physical states merely based on sensory information. In order to quickly recover such physical states with insufficient sensory information, anticipation is essential in human cognition, where a sufficiently rich photorealistic and physics-faithful semantic model of the world is maintained and simulated from time to time for prospection about robot actions and interactions among objects. Moreover, given the dynamicity of the environment where objects are permanently subject to unexpected physical changes, it becomes less and less relevant to preplan the actions and motions of robotic agents. In this situation, massive amount of knowledge about the world are formalized in a machine-interpretable manner so that action and motion plans can be automatically constructed in realtime based on reasoning about this knowledge and the actual context. On the other hand, medical sterility testing is a failure-critical process, in the sense that products or results must not only be error-free but shall be proved as such. This suggests that the robotic agents should not only execute actions but also ensure that the action was performed successfully or detect failures and react to these by replanning for instance the action accordingly. In this regard, we propose the TST as solution. The conceptual and reasoning framework with semantic models is presented in this document and the architecture is illustrated by Figure 1.

The TST is the combination of Semantic Digital Twin[1] technology with frameworks that enable us to trace performed actions created in WP4. Recently, Digital Twins (DT) have been proposed as digital replicates of environments for the digitization of manufacturing processes, ranging from industrial

[1] For more details about Semantic Digital Twins and the related reasoning capabilities, we kindly refer the reader to the chapter 3.2.4 Simulation-based Reasoning.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

production lines [11] to hardware in the loop testing [6]. The Semantic Digital Twin(semDT) facilitates geometric reasoning using a photorealistic virtual reality system and physics engine. It uses the scene graph data structures that are the implementation basis of virtual environments and integrates them with the symbolic knowledge representation system, providing every entity that is relevant for the robot agent, be it an object, object part, an articulation model, or a sub-scene, with a symbolic name.

To the best of our knowledge, digital twins do not treat the digital replicas as symbolic knowledge bases. In addition, we are not aware of digital twins being automatically created through robots.

SemDTs are also based on the idea of generating geometric structures using formal grammars. One prominent example is the notion of shape grammars [10], in which production rules are recursively applied on 2D or 3D geometric figures to generate more complex ones. They have been used in domains such as urban design [7] to generate different designs of cities.

For the TraceBot project, we provided a semDT that includes the environment of the intended use case and semantic representations of the scene. The environment, depicted in Figure 2 consists of models of the robot, the used objects like the pump, canisters and furniture. The semDT can be controlled using the middleware ROS. The distribution of the semDT is presented in 3.2.5.4.

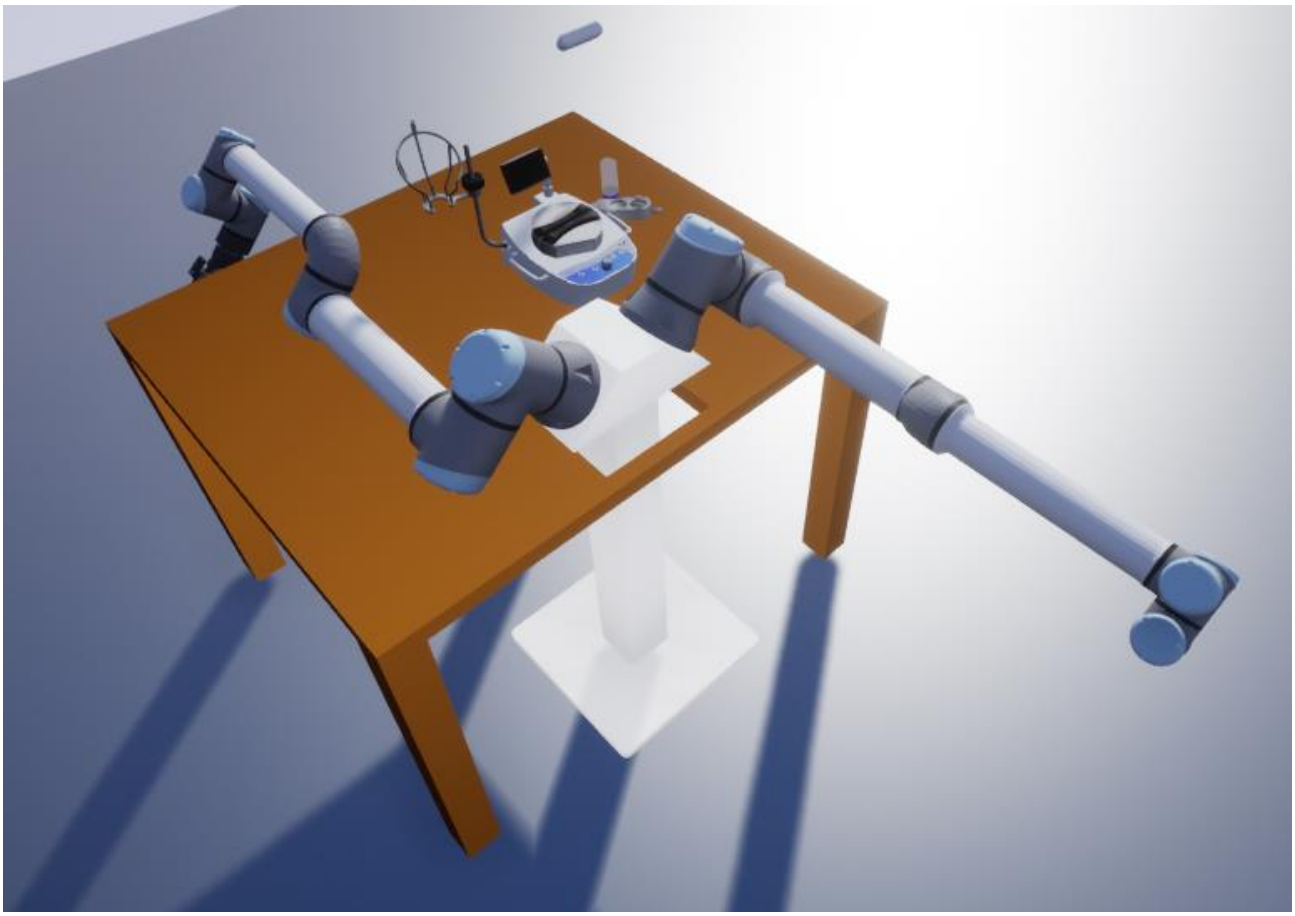


FIGURE 2: TRACEBOT ENVIRONMENT OF THE SIMPLIFIED USE CASE, CONSISTING OF ROBOT, PUMP, CANISTER AND TABLE.

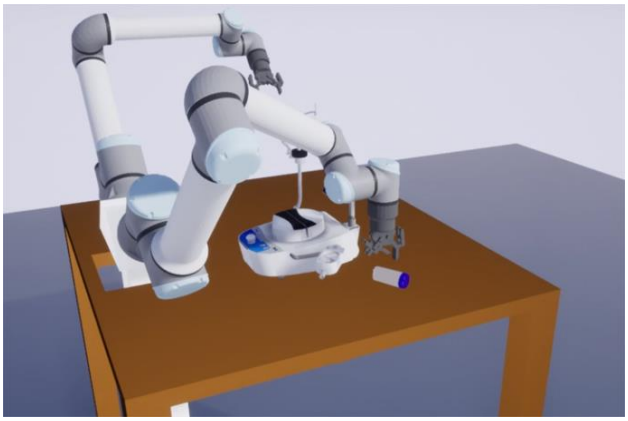
Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

3 Description of work & main achievements

In this chapter, we provide an overview of our key developments in the development of a reasoning apparatus and the creation of semantic models for the TraceBot project. Central components of the overall framework will be highlighted and motivated for the sterility testing use case.

3.1 Application in the TraceBot Scenario



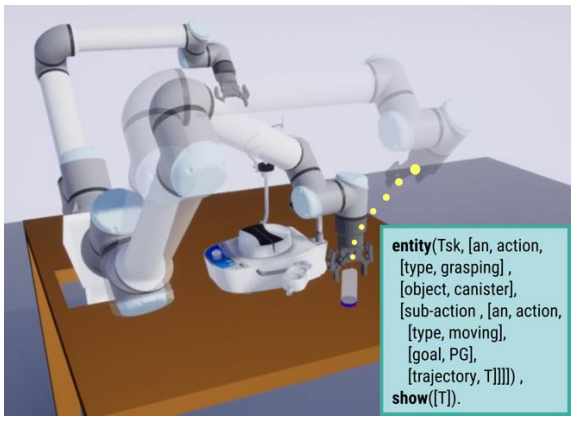


FIGURE 3 (LEFT): ACTION FAILURE FOR GRASPING ACTION; FIGURE 4(RIGHT): QUERY ON TRACEBOT NEEM

A key problem of TraceBot is "How to prove that actions are performed correctly". One part to solve this is the capability to ask the system what happened and make it possible to answer the question why it happened. To make this possible we are using the TST technology. Imagine we have a case where a grasp action failed as shown in Figure 3. How could it happen? Either the robot knocked the canister down, while going into the grasp position or the grasping is the reason. In Figure 4, we ask the system to show us the trajectory of the end effector during the approach to the grasp pose as well as the world state during begin and end of the action. As we can see, there is no problem here. This means we know that the problem was the grasping itself. By analyzing multiple instances of the action we can analyze whether this problem is systematically and has to be changed or if is occasionally and just needs optimizing. This reasoning capability enables the system to optimize its actions based on previous experiences.

3.2 Reasoning framework and semantic models

The TST has been defined as information provider in the introductory chapter where reasoning and knowledge were central. In this section, the reasoning framework will be presented, where we will explain how knowledge is represented and processed, how simulation enables reasoning, key components for the simulation-based reasoning, how the semDT should be queried and finally how the process traces are represented.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

3.2.1 Knowledge representation and processing

KnowRob [3] is a knowledge processing system designed for robots that combines knowledge representation and reasoning methods with techniques for acquiring knowledge and for grounding the knowledge in a physical system and can serve as a common semantic framework for integrating information from different sources. KnowRob combines static encyclopedic knowledge, common sense knowledge a.k.a ontology, task descriptions, environment models, semantic object information and information about observed actions that has been acquired from various sources (manually axiomatized, derived from observations, or imported from the web). It supports different deterministic and probabilistic reasoning mechanisms, clustering, classification and segmentation methods, and includes query interfaces as well as visualization tools. Beyond these classical reasoning mechanisms, it supports embodied simulations and imagistic reasoning mainly for the sake of anticipating. Its purpose is to equip robots with the capability to organize information in reusable knowledge chunks, and to perform reasoning in an expressive logic in order to turn undetermined queries (e.g., pick up the canister) into fully parameterized motion plans. Given that KnowRob has been designed for human-scale everyday manipulation activities, it also applies to TraceBot's medical sterility testing scenario. Figure 5 below highlights the core components of KnowRob.

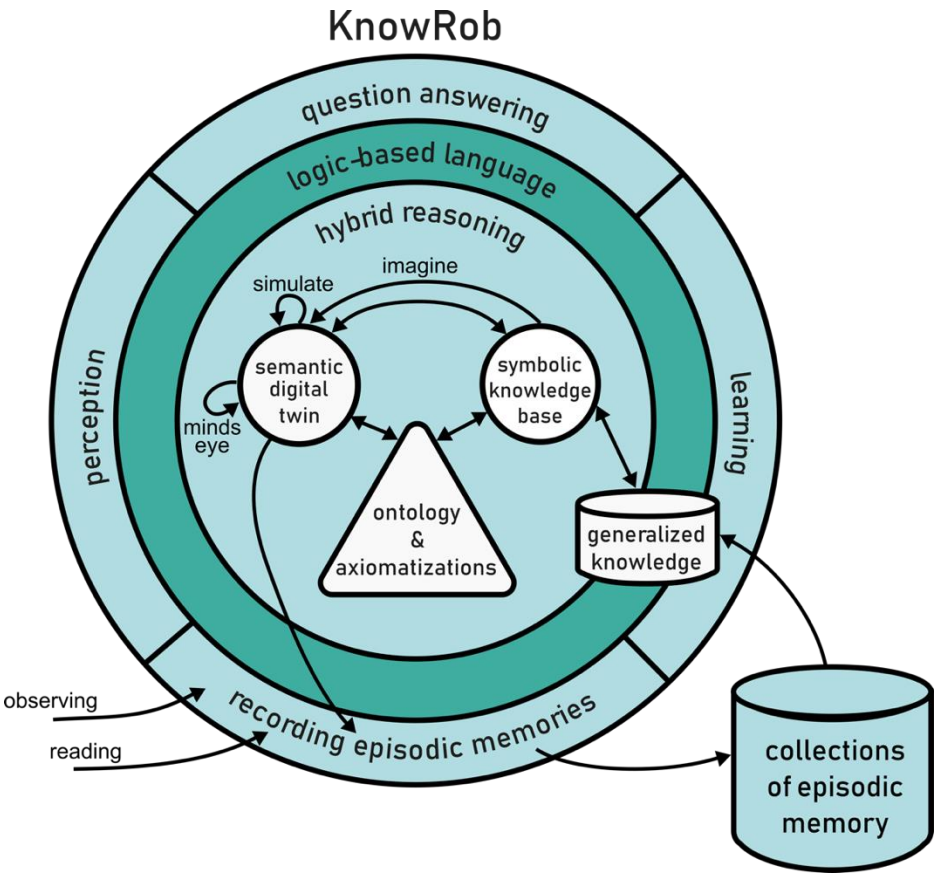


FIGURE 5: ARCHITECTURE OF KNOWROB

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

3.2.2 Ontology

The Socio-physical Model of Activities (SOMA)[4] is an ontological modeling approach for autonomous robotic agents performing everyday manipulation activities. It tries to catch the social as well as the physical context of activities. By physical model of activities, it is understood knowledge about the robot and the objects in its environments. The social context of activities refers to the common behavior of agents within a given context (i.e., the functions of actions). It allows to reduce the solution space of a query within a specific context and concretely allows the robot to act faster and softly. Notice that SOMA targets generic manipulation activities though recent works have been focusing on specializing it for kitchen-related activities. In this project, the SOMA ontology is extended with medical sterility testing-specific concepts.

Concretely, the ontology will encode common knowledge about TraceBot objects such as canisters, tubes, needles, pumps, the properties or attributes of such objects such as the color, material, shape, volume, location, 3D models but also the relations (e.g., spatial, compositional) among these objects. This information would then allow for instance to reason about plausible and feasible states of objects while interacting with them, which will then enable decision making and failure detection. Given the common location of the pump in the environment and the common spatial relations among the components of the pump, one might detect and locate with limited sensor data subtle parts of the pump such as specific buttons on the user interface. One might also infer that a glass bottle would get broken if it would fall or that the big bottle should not be placed on top of the canister because the canister shape does not allow it (i.e., thinner), while a work table might be a suitable supporting surface. Note that the ontology is also intended to encode knowledge about actions such as their temporal dependency (i.e., pouring precedes opening), but also their pre- and post-conditions. Reasoning on such knowledge would support contextual planning of processes (i.e., if bottle opened then pour liquid else open bottle), reducing on the one hand the space of possible plans and allowing online planning (i.e., in contrast to pre-planning) on the other hand, therefore better robustness to unexpected changes in the environment. Notice that while leveraging common sense knowledge about actions for planning, they are also regarded as process regulation guidelines that allow to detect failures (e.g. pouring action performed though bottle closed) but also establish audit trails as proof of the process execution success. The concrete programs responsible for the realization of these actions are referred in work package 3 to as skills, completing then the knowledge about actions. Finally, it should be noted that while leveraging the common sense knowledge for reasoning, the whole SOMA ontology rather than only TraceBot-specific concepts are used. For instance, though the concept canister was not in SOMA before TraceBot, extending SOMA for TraceBot connects the concept of canister as a sub-concept of the concept container to the latter. This being done, all the knowledge related to container in SOMA are reused for canister. For example, the following knowledge can then be inferred without explicit representations in the ontology: canisters can contain something; while grasping a canister one should care about its content. Figure 6 below provides a graphical overview of the SOMA ontology, where nodes represent concepts (e.g., object, action, situation) and arrows represent relations among concepts (e.g., a situation involves an action).

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

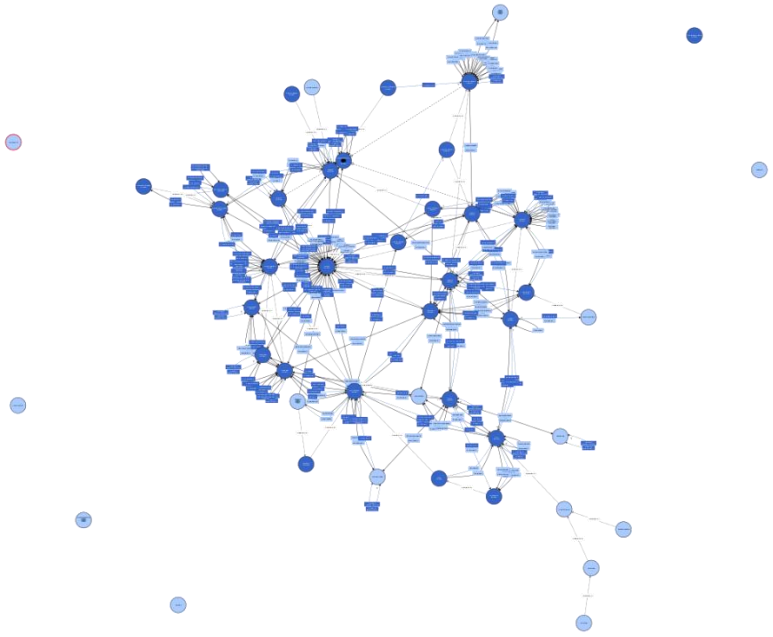


FIGURE 6: GRAPHICAL OVERVIEW OF THE BROAD SOMA ONTOLOGY

In contrast to the above figure, Figure 7 highlights in more detail the concept of canisters as SOMA is extended with the TraceBot domain. From this illustration, the concept of sterility test canister is a special medical test canister, which itself is a special canister. The canister is in turn a special container. Being a container, the sterility test canister contains fluids, is transparent and made up of glass. The container is an object which itself is an entity which is a top concept in the ontology. Top concept means that there is no concept on top of it or more abstract than it.

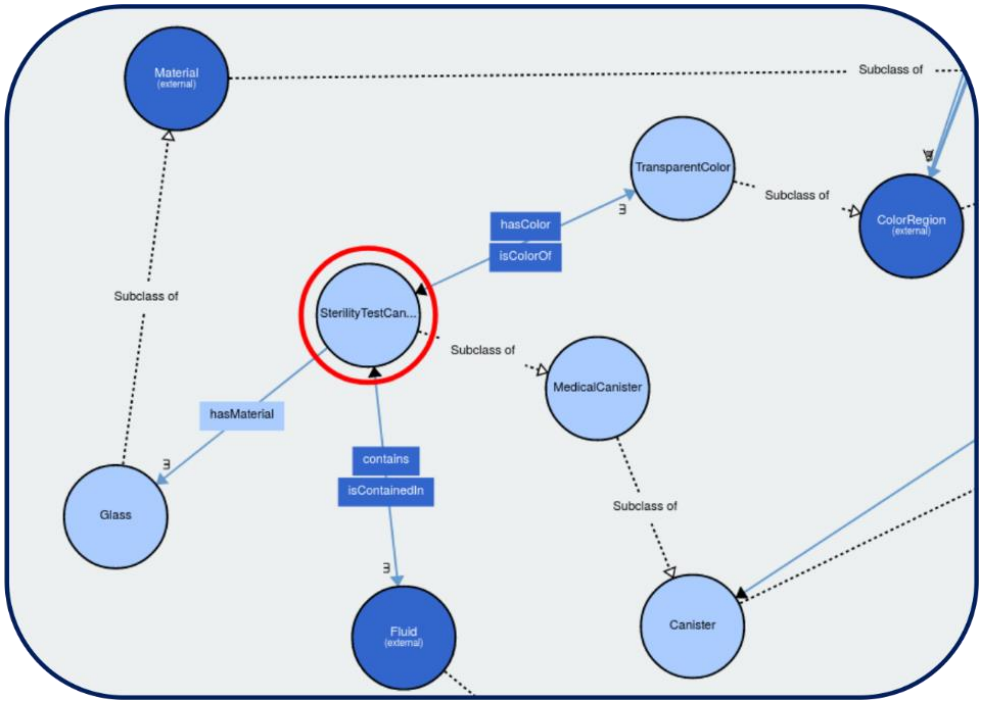


FIGURE 7: TOWARDS A TRACEBOT OBJECT DOMAIN IN THE BROAD SOMA ONTOLOGY

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

Notice that such an object description like illustrated above is essential for establishing the physical context of manipulation activities. In contrast to it, Figure 8 illustrates the concepts of action, situation, transition and the relations among them, which actually attempt to capture the social context of manipulation activities. The concept of situation can be commonly be regarded as social context which basically involves a state of the scene (e.g., objects + attributes such as the bottle is closed), expects an action a.k.a. event (e.g., open) whose the execution or handling will cause a state transition to yield a new situation, involving a new state of the scene (e.g., bottle is opened) and expecting a new action or event (e.g., pour).

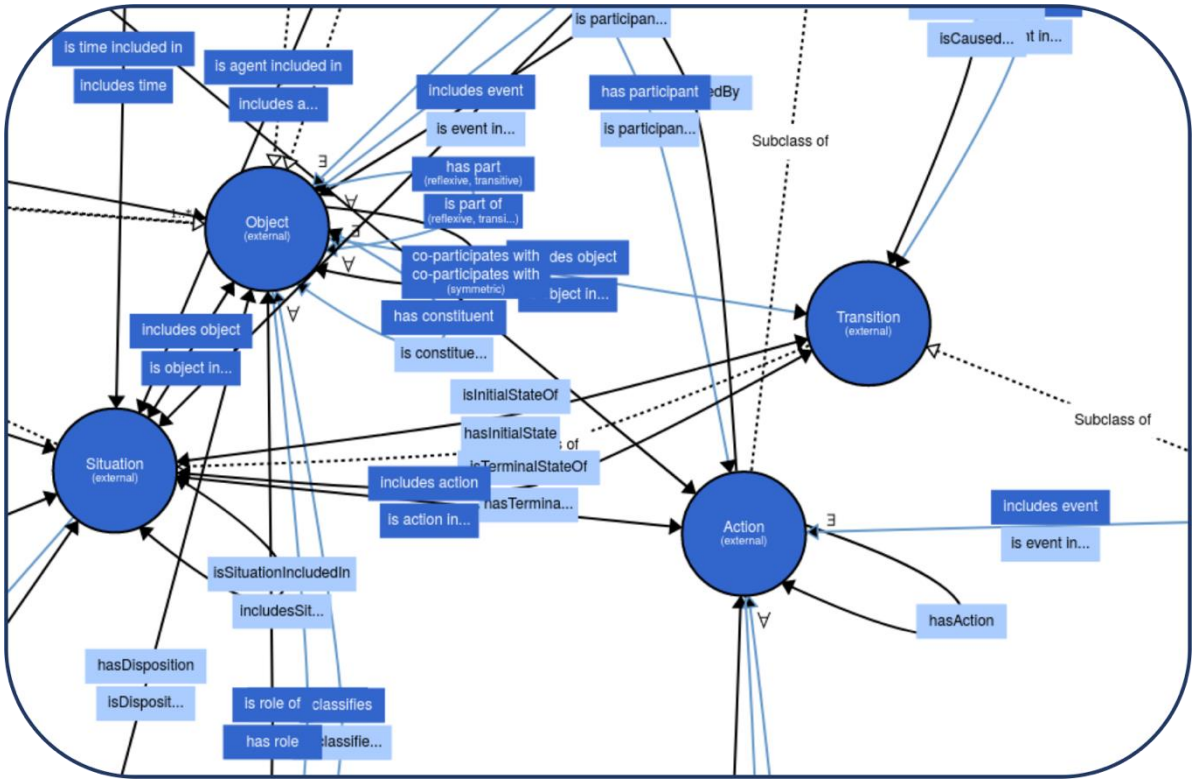


FIGURE 8: TOWARDS A TRACEBOT ACTION DOMAIN IN THE BROAD SOMA ONTOLOGY

3.2.3 Query Language

KnowRob provides a query language in order to retrieve information from knowledge bases. The expressive power — that is, the set of questions that can be asked — is provided by the ontology. One can retrieve all entities (e.g., instances) of a given entity category (i.e., concept such as canister) in the ontology and describe each entity using the attributes or properties (e.g., color, material) defined for the respective entity category (e.g., which containers are cylindrical and transparent?, How are they posed?, A small transparent and conical glass bottle has been detected). The queries are primarily asked to decide on how to manipulate objects or parameterize motions. For example, for medical sterility testing tasks typical queries are: where are the canisters, where to stand to pick up an object, how to reach for the object, how to position the grippers, how much force to apply, how

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

to move the arm to pick it up, how to hold it, etc. As information provider in the TraceBot system, a query interface was designed to allow the robot to query the TST. These higher-level queries are in turn translated into KnowRob query language and issued to KnowRob for answering. Notice that though most of these TraceBot-specific queries are related to objects and states, one can also state queries about actions such as illustrated by Figure 9 below.

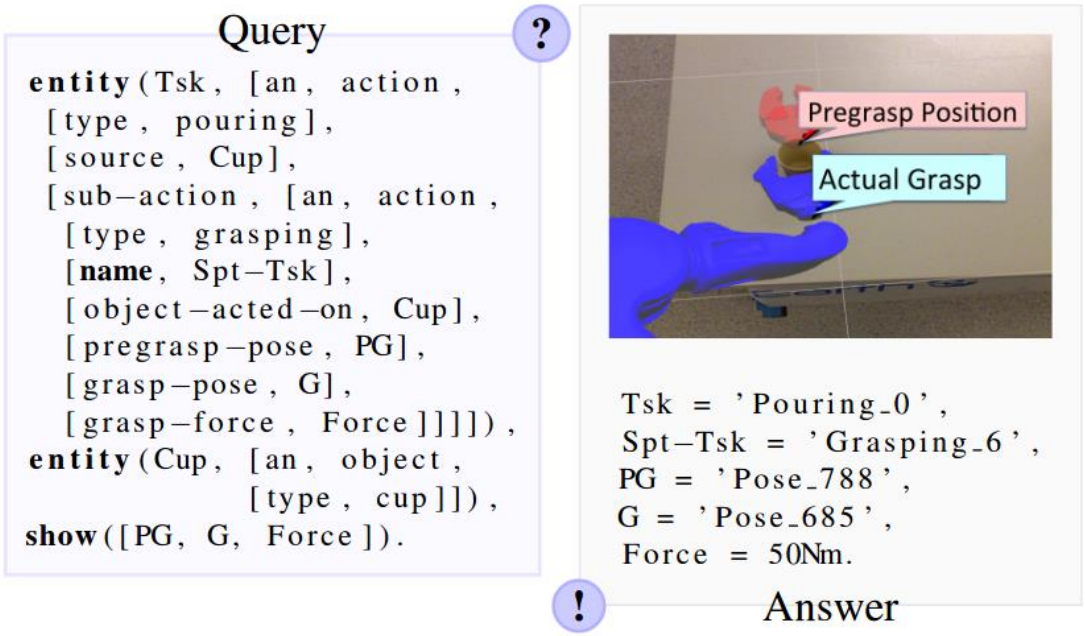


FIGURE 9: A TYPICAL QUERY IN KNOWROB ON THE LEFT, AND THE INFERRED ANSWER SHOWN VISUALLY ON THE RIGHT SIDE.

Based on a careful examination of the TraceBot use cases, a set of 11 higher-level queries, which will be translated into the KnowRob query language and answered by KnowRob, were derived:

- Assert new object: e.g., what is in the world?
- Update object: e.g., how is the world now?
- Request color image: e.g., mental visualization of scene
- Request Component Region: e.g., grasp region of needle's cap
- Request Object Component Pose: e.g., pose of the needle's cap
- Request Object Pose: e.g., pose of object?
- Request Region: e.g., canister's 3D bounding box?
- Request Region Color Image: e.g., mental visualization of partial scene
- Request Robot Parameter: e.g., grasp action success regarding the gripper state?
- Visual Expectation Match: e.g., how similar is the real state from the belief state?
- Query symbolic reasoning: e.g., actual audit trail state?

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

3.2.4 Simulation-enabled Reasoning

Knowledge-based systems provide a strong foundation for reasoning tasks and are well suited to formalize relations between entities and infer new facts from an existing knowledge base. What is still very challenging to model with these systems, is the representation of complex physical interactions and the associated effects. We propose to complement our knowledge processing infrastructure with a game engine based simulation environment to extend our reasoning capabilities in terms of dynamics and rich mental stimulation. We denote this simulation component as Semantic Digital Twin (semDT).

The key foundation to create such a reasoning apparatus, is the tight integration of the dynamic robot belief state with the game engine environment. All relevant entities from the agent and the world state will be modelled as a game engine environment. Instead of creating a rule-based system that describes state changes, we explicitly model the robot platform and the entities in the environment as a game in the game engine. The world model is then gained by starting the simulation process of the game engine and observing the changes in the environment. Each of these entities will be linked with the corresponding definitions in the symbolic knowledge base and can therefore be related to factual knowledge and semantics. What we gain with this combination is a knowledge model, that can exploit the simulation functionality the game engine provides.

Game engines provide advanced physics engines with a sufficient balance of speed, accuracy and realism which enables researchers and developers to create software solutions for robots in simulation without requiring real robot platforms [1]. Because game engines are specifically developed to be used in real time, it is now possible to reason about visual and physical effects in a fast and efficient way even if the environments are complex. These advancements allow us to integrate the game engine effectively into the robot control stack to provide reasoning methods which are hard to model in a symbolical way. Take for example the computation of spatial relations that are occur during manipulation actions. By replicating the actions in the world state of the robot into the game engine and basically replaying the actions in it, spatial relations can effectively be computed by requesting the contact information and spatial information from the API of the game engine. This information will also respect the influence of physics effects during the execution of the actions because of the physics engine running in the game engine.

Particle simulations for game engines can also be used to simulate manipulation actions which involve the handling of fluids. This would enable a system to reason about the fill levels of a container, after a certain action has been executed, like for example pouring a liquid into another container with a certain motion trajectory. If the complete pouring was successful, the full liquid volume of container A should be transferred to container B. Otherwise a potential spill can be detected and fed back to the high level task execution.

A key feature of game engines, is also the realtime rendering of photorealistic scenes. With the latest advancements of computer graphics software and hardware, the rendered material from the game engines has achieved unprecedented levels of visual fidelity. Because we are effectively constructing the belief state about the environment of the robot in the game engine, we can create a mental image of the estimated world state simply by using the virtual camera to render the word

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

in the game engine from the perspective of the robot. This enables the system to reason about the appearance of relevant objects before and after interactions with them.

Game engines provides a broad set of features, while also allowing fine detailed access to the interactions that record during manipulation actions. Force dynamic events like for example a grasp and lift action, can be divided into different motion phases like touching the object, keeping contact with the object and the object being lifted from the supporting surface. Each of these interaction properties can be fetched from the programming interface of the game engine and therefore be linked to our hybrid knowledge processing infrastructure to reason about substeps in manipulation actions.

Because the simulation capabilities of game engines are constantly extended, we propose to strongly integrate them as a simulation-based reasoning mechanism into our robot control system to benefit from the powerful opportunities these features offer for a comprehensive semantic world model. This will allow our hybrid knowledge representation and processing approach to scale with the technological advances that are integrated into the game engines and enable robots to perform manipulation actions more robustly by reasoning about the right action parametrizations in a general model.

3.2.5 Semantic Digital Twin

A central component of our reasoning framework is a simulation module based on game engine technology. This semDT enables the robot in TraceBot to reason about subsymbolic effects by combining the capabilities of the game engine with the semantic descriptions in our knowledge base. To enable this, we implemented several modules in our framework which are dedicated to support simulation-based reasoning about actions, objects, effects.

In the following chapter, we will highlight key developments in the semDT for the TraceBot project.

3.2.5.1 URoboSim

URoboSim [9] is a simulation framework that is the work horse of the semantic digital twin. Implemented as plugin for Unreal Engine 4, it allows us to import robots using URDF/SDF and simulate them. The scene of robot and environment is physics enabled and photorealistic, while the scene graph of the simulation is essentially a rendered knowledge base as depicted in Figure 10.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

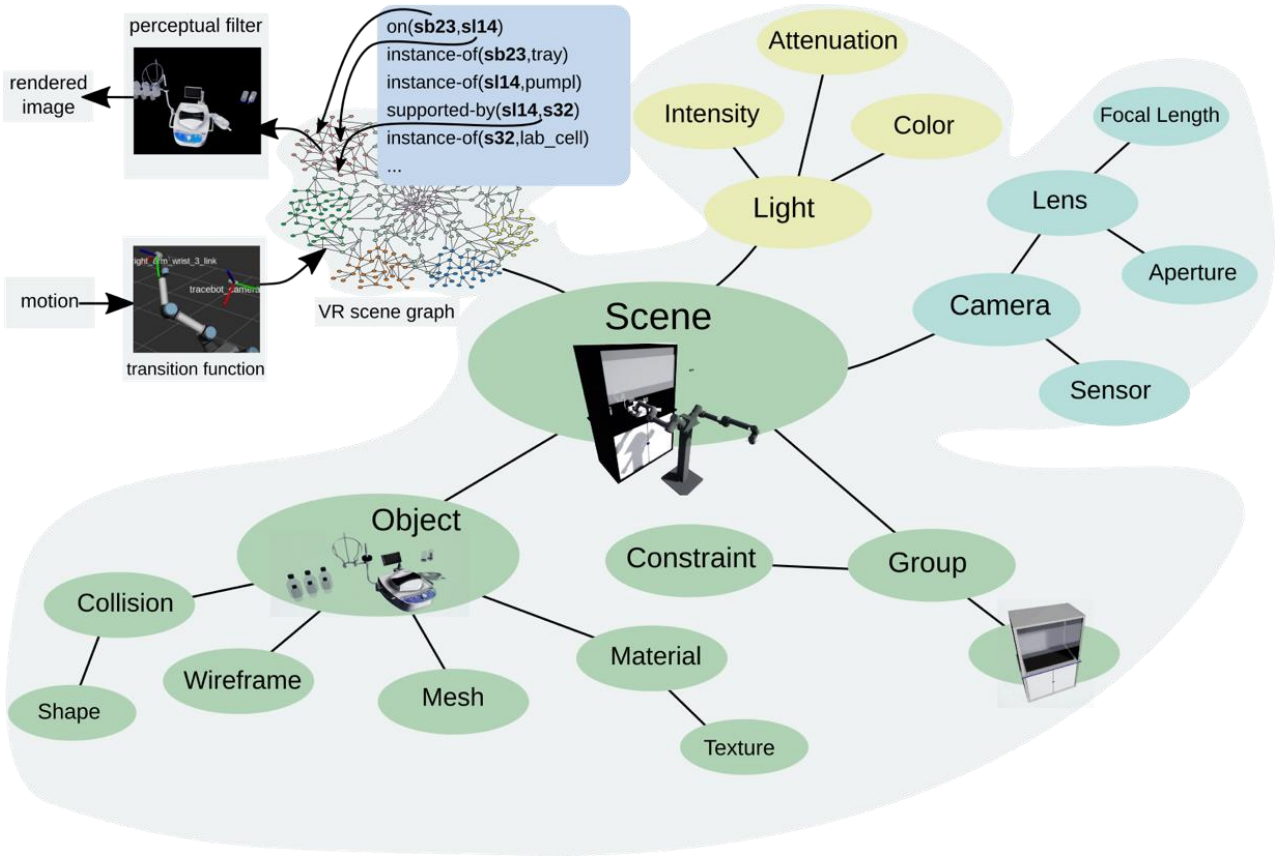


FIGURE 10: SCENE GRAPH OF THE TRACEBOT ENVIRONMENT.

URoboSim enables us to firstly, build a continues belief state, that provides the system with information about the environment. In each cycle of the perception-action loop the simulation environment is updated by physically simulating the body motions and their physical effects generated in the respective cycle and correcting the current scene in URoboSim to better match the captured camera image. Figure 11 shows how URoboSim is emulating the execution of a robot plan in order to compute the belief state of the robot.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

FIGURE 11: BELIEF-STATE (LEFT) DURING ACTION OF REAL ROBOT (RIGHT).

The second possible use case is to simulate actions using real action plans. URoboSim models robots using the data structures and service libraries of the open-source robot middleware ROS, which means that it is functionally equivalent to the robot simulation framework Gazebo. The first year milestone is focused on building and maintaining a belief state like in the previously mentioned first use case.

URoboSim includes models of different robots and a kitchen laboratory environment and a small drugstore. While the kitchen environment is handmade, the drugstore environment can be created by autonomous robot mapping given a grammar of how shelf systems are configured and realistic models of all products in the store and the laboratory environment can is generated by using URDF.

3.2.5.2 Object-specific Controllers

The required simulation abilities for a semDT of a competent robotic agent tasked with the challenge of sterility testing are manifold. In the previous chapter, we have introduced URoboSim which provides a comprehensive foundation to simulate different robots and handle environmental descriptions defined in typical ROS data formats. Besides that it is also important to consider the simulation of the objects the robot has to manipulate. In our TST architecture, our goal is to provide rich semantic models of typical manipulation tasks that are to be conducted in the studied scenario.

An important category of models in the semDT are semantic models of objects, which allow for realistic handling as well as appearance. Because game engines, upon which the semDT is built on, do not contain realistic, premade interaction models of objects in a sterility testing lab scenario, our

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

first goal was to analyze the use case description based on the process descriptions from WP1 to derive the classes as well as characteristics of the relevant objects.

An important distinction between the objects is their degree of dynamics and interaction. Many parts of the environment of the report can be considered as static and be jointly modeled with the robot description which is used in the TraceBot system to calculate motion trajectories. However, many objects that are relevant for the sterility testing use case can be considered dynamic. This involves for example the sterility testing kit, as clamps, tubings, canisters and needles. The scenario also includes different bottles for specific purposes in the process as well as a special pump which is tailored towards the sterile pumping of liquids based on the sterility testing kit. Bottles as well as the pump both show a certain degree of interactivity. Bottles are largely rigid but can also be interacted with, by either screwing their lid of or pushing a needle into the septum of it.

The pump on the other hand is an electric device which can change its behavior based on the control commands one has given to it. After the tubing has been inserted into the pump, the pump head has to be locked, which is causing a rotation of it. When the locking action is complete, it is possible to activate the pump which is then starting the liquid transfer process from one side to the other.

Simulating all of the aforementioned ways of interacting with the objects in the required amount of expressiveness and machine understandability is a challenging task with standard representation techniques. We propose to exploit the powerful and fast simulation capabilities of modern game engines, which are tailored towards the realistic simulation of complex worlds with manifold objects, as a method to represent advanced, semantic object models.

FIGURE 12: ILLUSTRATION OF THE CREATED PUMP MODEL FOR THE STERILITY TESTING USE CASE. ON THE LEFT IS THE COMPOSED OBJECT WHERE ON THE RIGHT YOU CAN SEE THE DIVISION INTO THE INDIVIDUAL, FUNCTIONAL SUBPARTS. THIS DECOMPOSITION ALLOWS US TO MODEL THE FUNCTIONS OF THE DEVICES IN A DETAILED WAY BY CREATING OBJECT-SPECIFIC CONTROLLERS FOR THE RELEVANT FUNCTIONALITY.

The creation of the game engine enabled object models is typically divided into three phases. The first one is the acquisition of the overall geometric and visual features to create a 3D model of the desired objects. In the TraceBot project we have access to the relevant objects of the use case and

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

can either create object models manually, scan them with specialized 3D-scanning hardware or also import CAD models which might be directly available. However, these models still need to be adapted to be usable in a flexible way in the game engine in the second phase. An important requirement for the proper simulation of the objects, is the precise recreation of the individual subparts that are required for the function of the device while still maintaining an optimal amount of model complexity, to still be able to simulate the full scene in real time. This means for example that the scanned 3D models need to be separated into individual meshes like for example the head of the pump which is to be rotated during the process. On the other side, the relevant features like for example the threading for screws in the device can be left out to reduce the complexity and ease the simulation load. Finally, the specific controllers for the semantic object models are conceptualized and implemented. To be able to generalize and fully employed the functionality of the game engine, we are implementing the controllers as Unreal Engine Plug-ins which provide the required functionality as so-called Components, which can be assigned to the objects that are to be controlled. Inside of these Components, we have full access to the simulation capabilities that the game engine provides. This includes physics simulation features like constraints-based definition of subpart relations, object to object attachments, world and object manipulations and more. By employing these methods and simulating the functionality of the objects, we can enable our system to effectively reason about action effects by replicating the behavior of the object.

In the following, we would like to highlight two exemplary controllers for simulation tasks. The first is dedicated to the simulation of the electrical devices, such as the pump. These devices typically have a set of defined inputs (buttons to start the process or modify the operation of the device), outputs (LCD displays, LEDs, etc.) and effects. In the case of the pump, the operation is controlled by five different buttons which can start or stop the process, (de)activate the locking mechanism or change the parameters. These interactions can be modelled by implementing the detection of button presses in the game engine with either three-dimensional trigger volumes when pressing short action buttons or implementing bigger, spring-like knobs with physics constraints and connect them with the corresponding logic that simulates the operation and configuration of the device. Outputs can also be simulated in the framework with different methods, where one functionality that we have investigated is of particular interest.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

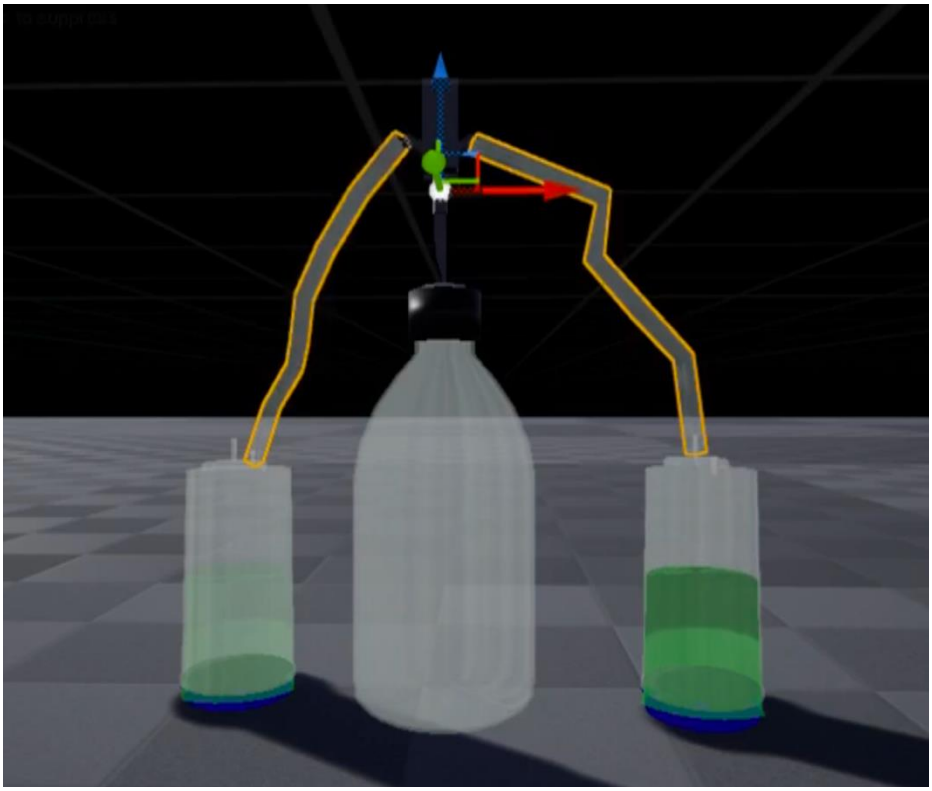


FIGURE 13: NAIVE LIQUID SIMULATION RESULT FOR TRANSPARENT CONTAINERS IN THE STERILITY TESTING USE CASE

The employed game engine supports the recreation of displays by providing an API for the definition of UI widgets. We have created a first UI widget that can recreate the LCD display on the pump, which is providing feedback to the user on the ongoing process, for example if the locking action has been completed or if the pump is currently pumping. This will effectively enable our reasoning system to estimate the visual feedback of the pump, given a certain operation that should currently be running on the pump. The effects of electrical devices can be manifold. It could be as simple, as a rotation of a subcomponent or be as complex as the pumping of liquids from one container to another by compressing a tube that is mounted in the pumping head. Depending on the use case, one has to choose the right approximation to balance between the required simulation accuracy and the available computational resources. In the TraceBot scenario the robot should be able to reason about the fact that liquids will be transferred in the process between different containers and how they will look like after the transfer has been finished. To implement this, we propose our second type of controls for simulation tasks which is dedicated to the simulation of objects states. We have created a naive liquid simulation (see Figure 13) for the sterility testing use case, which can set fill levels on containers with arbitrary liquids. This functionality is compatible and controllable by the controller for the pump, which can automatically influence the fill levels of the corresponding containers when the pumping operation is running. Based on these types of specific controllers for objects, we have laid out the foundation for the subsymbolic reasoning of action effects in the sterility testing use case.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

3.2.5.3 Game Engine extension modules

The exploitation of game engines as an additional apparatus for robotic reasoning frameworks is a novel topic. While game engines do provide powerful mechanisms to simulate physical or visual phenomena, they still lack a tight integration of these capabilities into state-of-the-art reasoning frameworks, but also in robotics. In order to make use of the game engine capabilities for reasoning about verification actions, we have identified three key elements that are required to combine a semDT model of the sterility testing use case with the ROS-based control framework that is developed in the TraceBot project. The first key element is the implementation of object as well as robot behavior. This element is mainly covered by the development and integration of URoboSim (see Chapter 3.2.5.1) for (virtual) robot control as well as the object controllers (see Chapter 3.2.5.2) to replicate their real-world behavior in the game engine.

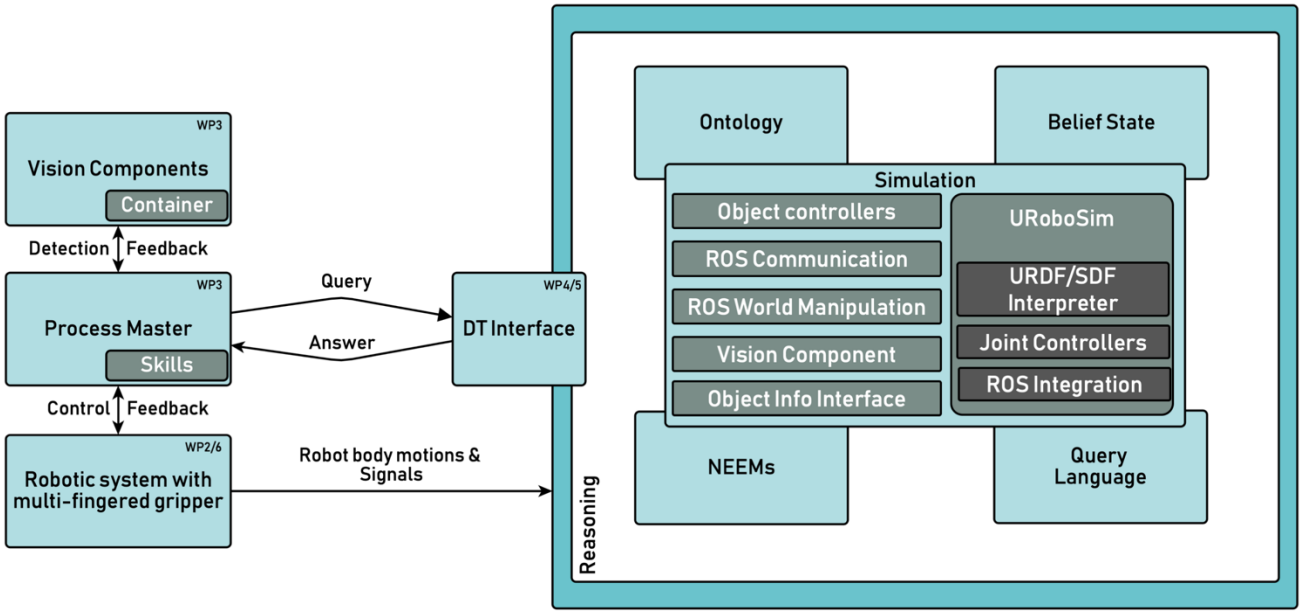


FIGURE 14: HIGH-LEVEL ARCHITECTURE OVERVIEW OF THE SEMDT SYSTEM WITH A FOCUS ON THE GAME ENGINE MODULES THAT ARE PROVIDING THE REQUIRED SIMULATION CAPABILITIES IN THE TRACEBOT USE CASE.

These main components will allow us to build a forward model of the robot actions to reason about their expected effects. This could for example include the estimated state of the gripper after grasping the canister from the sterility testing kit or how the pump head changes when the pump is being locked. The second key element of the game engine extension modules is dedicated to the communication between the rest of the framework and the TST via ROS interfaces. By providing a direct ROS integration in the game engine, we can integrate components by other partners which are running in the full robot software stack. This includes for example the use of standard pose definitions from ROS easing the integration or also the replication of robot joint states from the live system into the semDT model effectively updating the robot state in the simulation efficiently. Our third main element in the game engine eco system is devoted to the manipulation of the simulation environment. Over the course of the robot execution, it is necessary to update the estimated world state also in the semDT. An example for this, is the detection of objects necessary for the task

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

executions. When a canister has been detected at a certain pose p, then it should automatically be put at the corresponding place in the simulation environment of the TST. Such functionality will be provided by a ROS world manipulation layer directly in the game engine, allowing changes to be made from all nodes in the ROS system, allowing the control to be done very flexibly. The last key element is dedicated to general information and data retrieval. To verify actions that the robot has done, we want to be able to request information about the object according to the semDT model. Imagine that the robot is about to grasp the canister, lift it up and move it towards the drain tray. In this scenario, the semDT model should replicate the same behavior and also simulate the grasp of the canister and the motion trajectory towards the drain tray. At the end of this movement, we can ask the semDT model where the canister is expected to be. If it estimated that the canister could not be grasped and moved with the given motion parameterization, a deviation between the estimated pose of the object and the inferred pose from the semDT could be detected and communicated to the process execution.

We also want to be able to retrieve semantic information like the spatial relation of objects after certain actions have been conducted to reason about the desired effects of object properties relevant for the executed task. If we recall the example of putting the canister in the drain tray, we could use our semDT to infer that *in(canister,drain tray)* should hold after the manipulation action of the real robot has been replicated by our forward model and check if this relation holds by computing that information from the data structures in the game engine.

Another source of information that we have included in our reasoning apparatus is a component to generate estimated visual sensory inputs from the semDT. This module basically allows the system to imagine the percepts that should be perceived through the available sensors in a given task and scene context. Please note that the semDT model always emulates what is happening in the real world execution and provides therefore a virtual replica of the target environment. By depicting what the scene should look like after a certain action and comparing the real world scene, the system can reason about the deviations that might have occurred on the task-critical entities in the environment and therefore has an important, visual clue for the success and the verification of a task-step. The virtual sensor in the game engine can render percepts of typical RGBD(RGB + Depth) sensors and automatically generate perfectly segmented object masks which identify which region in the image belongs to which objects. Object-centric queries about the estimated appearance are therefore possible and can be rendered by the semDT. The semDT vision component can also be flexibly parametrized to emulate the characteristics of typical RGBD sensors like resolution or field of view. We have successfully created a set of core functionalities that enable semDT reasoning for sterility testing use cases, that provides an effective extension to our knowledge base explained in the previous chapters.

3.2.5.4 SemDT Distribution model

Unreal Engine 4 provides two different ways to distribute a project, either as an Unreal project or as a compiled program. Both require the installation of the Vulkan graphics runtime. In order to use the Unreal project, it is required to install Unreal Engine 4, which is cross-platform compatible. On the other hand, the compiled program is self contained without additional requirements, but is only

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

compatible with the system it is compiled for. For TraceBot, we decided to provide our partners the compiled program. This has the advantage (a) that our partners do not have to install Unreal Engine and debug possible problems and (b) can start the semDT simply using the command line. This makes it possible to restart the simulation automatically in case of problems. The initial version of the semDT has already been deployed as part of the first integration milestone.

3.2.6 Perception Executive

The goal of the Perception Executive in the TST is the observation of dynamic task-relevant information while also providing an integration with the reasoning capabilities of the framework. Vision capabilities are provided by TUW which are developing state of the art methods for recognizing properties in challenging sterility testing scenes such as transparent flexibles or bottles and delicate objects. These capabilities are encapsulated in RoboSherlock [2], which is a general perception framework developed by UOB. It provides a symbolic query language, that allows robotic systems to formulate perception tasks in a generic form. Each perception task will be decomposed in a task-specific set of vision experts that are executed to generate object hypotheses based on the perceived sensory data and the task description. Imagine that the robot is supposed to detect a transparent bottle, which is hard to observe in the data provided by typical depth sensing cameras. RoboSherlock can automatically infer, based on its knowledge, that this perception task can be solved by focusing on the 2D RGB sensor data and involving a detection module that is defined to detect the given class and affordances.

RoboSherlock also provides state of the art methods for mental imagery [8]. This is accomplished by replicating the hypothesized belief into a game engine environment and generating virtual percepts how the world should be perceived by the robot, given that the belief is true. These “minds eye” percepts are analyzed by RoboSherlock and compared with the real-world sensory inputs and the associated expert annotations. In this step, the system can also infer alternative hypotheses, replicate these into the game engine environment and compare the resulting virtual percepts to optimize the correspondence between the real and virtual sensory inputs. This system provides an important foundation for cognitive robot architectures, by enabling the creation of mental images and using them to reason about the validity of the inferred world state. In the TraceBot project, this functionality will extend the set of verification actions into the visual domain, allowing the robot to check the visual correspondence between the desired action effects and the current perceived scene.

3.2.7 NEEMS: A foundation for audit trails

NEEMs [5] stands for Narrative-enabled Episodic Memories. They actually encode chronological traces of activities at a symbolic (i.e., semantic) as well as at a subsymbolic level. At the symbolic level also known as NEEM narrative, the story of the ongoing activity is collected including for instance annotations of scene objects (i.e., spatial relations, classes, colors, shapes) and robot actions (action types, transitions, events), whereas the subsymbolic description also known as

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

NEEM experience encompasses sensor data such as entities' trajectories, poses, etc. In this project, NEEMs constitute the foundation of audit trails which themselves are chronological sets of records providing evidences that the processes were compliant with medical sterility test regulations. Through NEEMs, it will be for instance possible to check that all the actions involved in a process were performed but also that the precedence order among those actions was respected. On the other hand, NEEM experience containing sensor data could be examined to actually check that the bottle was actually empty at the beginning of the process (e.g., visual image of the bottle, mass of the bottle), which are crucial in case the human examiner is skeptical on the semantic results provided by the computer. Moreover, as shown by the first row of Figure 15 below, the afore presented ontology is grounded in the NEEMs, meaning that more information than what is contained in the NEEMs can be derived. For instance, if the NEEMs relate that the big bottle was place on top of the canister, then it could be inferred also based on the ontology of spatial relationship between the big bottle and the canister that something went wrong since it is not possible to place the canister on top of the canister. Another potential of NEEMs as foundation of audit trails, as illustrated by the second row of Figure 15 below, is that they can be replayed within specific timeframes which is of great importance since raw data are usually cumbersome to read in a way to understand what was going on. By replaying a portion of NEEM experience, the human visualizer can even extract more semantics than what is contained in the NEEM narrative. OpenEASE is a platform with web-interface designed to allow visual, online access to NEEMs. Finally, notice that NEEMs can be exploited as training data where NEEM experiences are regarded as raw data and NEEM narratives as annotations of these raw data.

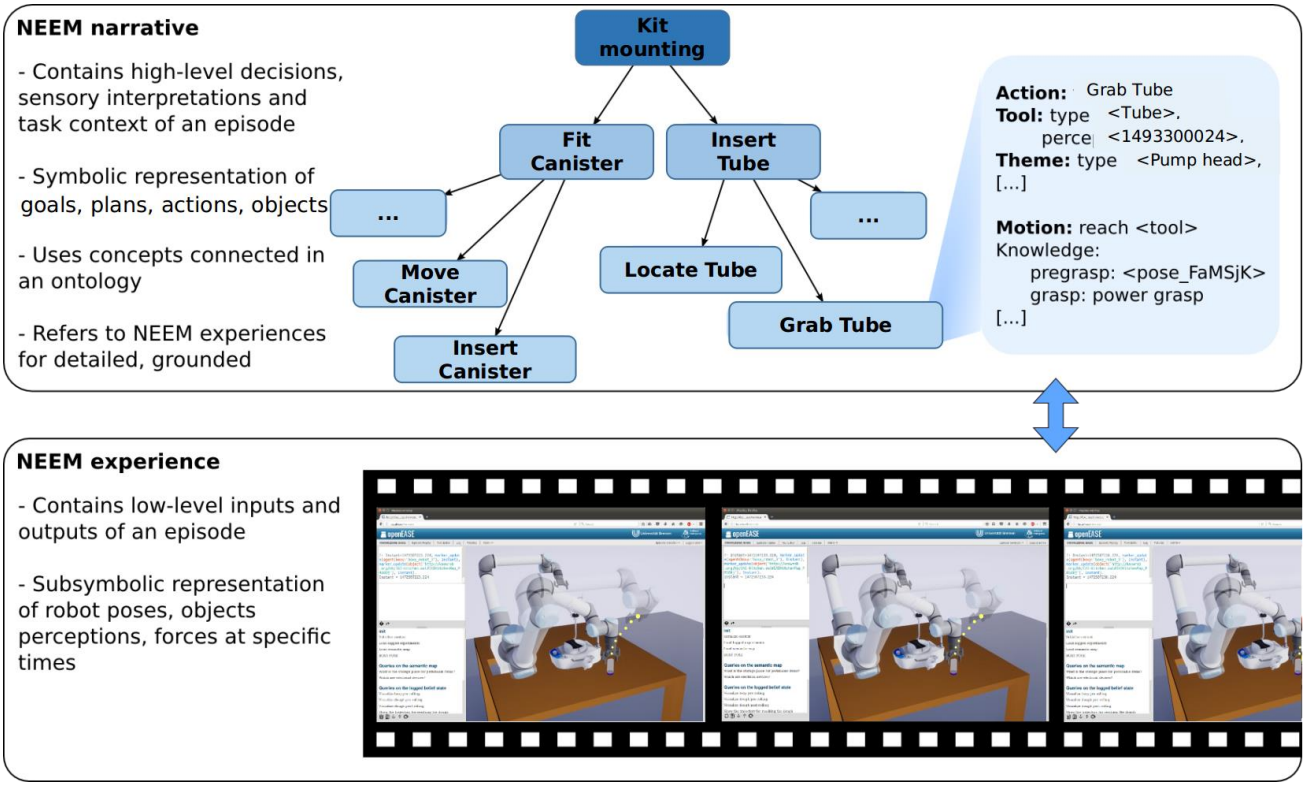


FIGURE 15: NEEM — A FOUNDATION OF AUDIT TRAIL IN TRACEBOT PROCESSES

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

4 Deviations from the workplan

No major deviation has been detected, and the document has been delivered on time.

5 Conclusion

In this deliverable, we have outlined the conceptual and reasoning framework that we have developed for the TraceBot use case. A key challenge of the TraceBot project is the implementation of novel methods to provide robots with means of awareness to be able to validate their actions. We want to tackle this challenge by developing a Traceable Semantic Twin that features a hybrid knowledge-based system that is built on Semantic Digital Twin technology and combines a VR scene graph with a symbolic knowledge base. The process master of the robotic system can query this system in order to parametrize the manipulation actions to be conducted in TraceBot. The Semantic Digital Twin simulation enables the system to conduct mental simulations in order to increase the quality of the estimated world state and ultimately infer if executed actions have been successful. This provides robots with valuable information to perform manipulation tasks with more robustness and efficiency. Our reasoning apparatus also provides the potential to trace the task executions and combine them with rich episodic memories. These data structures allow us to build audit trails for regulated robotic processes, which can be thoroughly analyzed with a provided query language, allowing for detailed introspection.

In this year, we have already developed the first version of the Semantic Digital Twin in TraceBot and provided a deployed version in the context of the integration milestone that is connected to the task execution in the mockup simulation. Our next actions are focused on the goal-oriented extension and adaptation of the Semantic Digital Twin and reasoning capabilities during the evaluation of the robotic execution of manipulation actions on the sterility testing use case. We will also further work on the enablement of NEEMS as a source of rich semantic descriptions for regulatory audit trails by combining the context of the task execution with our hybrid knowledge-based approach.

6 References

[1] Nvidia isaac sim. https://developer.nvidia.com/isaac-sim. Accessed: 2021-12-01.

[2] Michael Beetz, Ferenc Balint-Benczedi, Nico Blodow, Christian Kerl, Zoltan-Csaba Marton, Daniel Nyga, Florian Seidel, Thiemo Wiedemeyer, and Jan-Hendrik Worch. Robosherlock: Unstructured infor- mation processing framework for robotic perception. Handling Uncertainty and Networked Structure in Robot Control, pages 181–208, 2015.

[3] Michael Beetz, Daniel Beßler, Andrei Haidu, Mihai Pomarlan, Asil Kaan Bozcuoglu, and Georg Bartels. Knowrob 2.0 – a 2nd generation knowledge processing framework for cognition-enabled robotic agents. In International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 2018.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)

[4] Daniel Beßler, Robert Porzel, Mihai Pomarlan, Abhijit Vyas, Sebastian Höffner, Michael Beetz, Rainer Malaka, and John Bateman. Foundations of the socio-physical model of activities (soma) for autonomous robotic agents. In Boyan Brodaric and Fabian Neuhaus, editors, Formal Ontology in Information Systems - Proceedings of the 12th International Conference, FOIS 2021, Bozen-Bolzano, Italy, September 13-16, 2021, Frontiers in Artificial Intelligence and Applications. IOS Press, 2021. Accepted for publication.

[5] A. Bozcuoglu. Fast robot learning using prospection and experimental knowledge : A cognitive approach with narrative-enabled episodic memories and symbolic knowledge. 2019.

[6] Christian Dufour, Zareh Soghomonian, and Wei Li. Hardware-in-the-loop testing of modern on-board power systems using digital twins. pages 118–123, 06 2018.

[7] Milos Mandic and Bojan Tepavcevic. Analysis of shape grammar application as a tool for urban design. Environment and Planning B: Planning and Design, 42(4):675–687, 2015.

[8] Patrick Mania, Franklin Kenghagho Kenfack, Michael Neumann, and Michael Beetz. Imagination- enabled robot perception. In International Conference on Intelligent Robots and Systems (IROS), 2021. Best Paper Award on Cognitive Robotics.

[9] Michael Neumann, Andrei Haidu, and Michael Beetz. Urobosim—a simulation-based predictive modelling engine for cognition-enabled robot manipulation.

[10] George Stiny, James Gips, George Stiny, and James Gips. Shape grammars and the generative specification of painting and sculpture. In Segmentation of Buildings for 3DGeneralisation. In: Proceedings of the Workshop on generalisation and multiple representation , Leicester, 1971.

[11] Jan Vachalek, Lukas Bartalsky, Oliver Rovny, Dana Sismisova, Martin Morhac, and Milan Loksik. The digital twin of an industrial production line within the industry 4.0 concept. In 2017 21st international conference on process control (PC), pages 258–262. IEEE, 2017.

Horizon 2020

TraceBOT receives funding from the European Union's H2020-EU.2.1.1. INDUSTRIAL LEADERSHIP programme (grant agreement No 101017089)