The EU Framework Programme for Research and Innovation H2020
Research and Innovation Action

CENTAURO

Deliverable D4.4

Switching between Direct Control and Prediction Mode

Dissemination Level: Public

Project acronym:	CENTAURO
Project full title:	Robust Mobility and Dexterous Manipulation in Disaster Response by Fullbody Telepresence in a Centaur-like Robot
Grant agreement no.:	644839
Lead beneficiary:	RWTH – Rheinisch-Westfaelische Technische Hochschule Aachen MMI - Institute for Man-Machine Interaction
Authors:	Torben Cichon
Work package:	WP4 – Modeling and Simulation
Date of preparation:	2018-07-19
Type:	Report
Version number:	1.0

Document History

Version	Date	Author	Description
0.1	2018-06-05	Torben Cichon	initial version
0.2	2018-06-06	Torben Cichon	updated initial version
0.3	2018-06-07	Torben Cichon	draft completed
0.4	2018-06-27	Torben Cichon	KHG review included
0.5	2018-06-27	Torben Cichon	SSSA review included
0.6	2018-06-29	Torben Cichon	final updates
0.7	2018-07-16	Torben Cichon	updated based on integration meeting
0.8	2018-07-17	Tobias Klamt	final review by coordinator
0.9	2018-07-18	Torben Cichon	final review by RWTH
1.0	2018-07-19	Torben Cichon	missing citations added

Executive Summary

In D4.4 we present the developments towards the final CENTAURO system focusing on using the simulation in the loop for direct control and prediction and to switch from the control of the simulated robot to the control of the real robot.

The key element of this task is to integrate all relevant data sources into a complete virtual world with all important features of a directly teleoperated robot. With all these elements integrated, the operators will be able to control the simulated robot in a simulated disaster scenario in "offline mode" before switching control to the real robot.

Upon decision of the operators, control commands can be redirected from the physical CENTAURO robot towards the simulated robot. The physical robot remains in standby mode and keeps on sending updates to the operator station, after the Digital Twin (DT) of the robot and Digital Twins of the robots environment are instantiated. This is the basis for different look-ahead simulation runs, where the operators can evaluate multiple action alternatives.

The current task is heavily based on the results of previous developments from T4.1 - T4.3 and uses processing results from the operator interface (WP3), navigation (WP5), and manipulation (WP6) work packages. The complete robot and environment simulation will be connected to the operator interface developed in WP3, i.e. the exoskeleton or direct ROS control as well as visualization monoscopic and stereoscopic though head mounted display (HMD) technology.

During a live teleoperated mission the robot can be stopped and the operators have the chance to closely inspect the environment in virtual reality. With the DT, they can also try out different maneuvers and simulate the results of their actions. After such pure simulation phases, the best approach can be selected and the actions can be executed using the real robot.

Acronyms

DT	Digital Twin
RT	Real Twin
FPO	First Person Operator
SO	Support Operator
UI	User Interface
HMD	Head Mounted Display
ROS	Robot Operating System
URDF	Unified Robot Description Format
IF	Interface

Contents

1 Introduction

The 3D simulation system is the basis for the predictive robot model and the interaction of the operator with the look-ahead simulation (see Grant Agreement [9]):

> The key element of this task is to integrate all relevant data sources into a complete virtual world with all important features of a directly teleoperated robot. With all these elements integrated into the VTB, the operators will be able to control the robot in a simulated disaster scenario in "offline mode" (see bottom row in Figure for Task Description 4.1 in Sec. 3.1.2 in the Grant Agreement). As mentioned before, the predictive simulation is fed with the robot's control and perception modules (software-in-the-loop) to produce an estimate of the robot's actual behavior, which will be displayed to the operators. The Figure for Task Description 4.4 in Sec. 3.1.2 in the Grant Agreement illustrates the intended system architecture, which details the bottom left part of the mode overview from T4.1. Upon decision of the operators, control commands via exoskeleton can be redirected from the physical CENTAURO robot towards the simulated robot and its perceived environment. The physical robot remains in standby mode and keeps on sending updates to the original Central World Model (CWM), after a clone of the CWM (CWM' in the figure) is instantiated. This new CWM' is the basis for different look-ahead simulation runs, where the operators can evaluate multiple action alternatives. The current task is heavily based on the results of previous developments from T4.1 - T4.3 and uses processing results from WP3, WP5, and WP6. The complete robot and environment simulation will be connected to the operator interface developed in WP3, i.e., the exoskeleton with the motion capture system and the stereoscopic HMD display. Innovative methods for "just in time" operator training and benchmarking shall be developed as part of this task. During a live teleoperated mission, the robot can be stopped, and the operators have the chance to closely inspect the environment in virtual reality. With a predictive robot model, they can also try out different maneuvers and pre-simulate the consequences of their actions. After such pure simulation phases, the best approach can be selected and the actions can be executed using the real robot.

The core components of work package WP4 in general, also with respect to WP8 (Requirement Specification and Evaluation), are:

1. Integration and Interfaces,
2. Simulatable Robot Model, and
3. Simulatable Environment Model.

Thus, D4.4 is not only limited to the "Switch" from reality to simulation and back but moreover comprises the integration and connectivity of all modules with the simulator. The interface of the simulation model—from now on called **Digital Twin (DT)**—to real hardware components—called **Real Twin (RT)**—as well as external input or output modules and devices is essential. Additionally, D4.4 covers modeling and instantiating the DT of the robot (directly controlled by the support operator) and the creation of intuitive (stereoscopic) visualizations and an intuitive user interface for the operators.

2 Overview

The deliverable D4.4 (Switching between direct control and prediction mode) encompasses:

1. One holistic **Digital Twin of the CENTAURO robot**, including
   - Interfaces,
   - Rigid body simulation of the robot,
   - Sensor simulation of the sensor head,
2. **Visualization and Rendering**, including
   - Visualization of current robot state,
   - Visualization of raw sensor data,
   - Visualization of pre-processed sensor data (data output of WP5 and WP6),
   - Monoscopic and stereoscopic rendering of all this data for (i) the 1st person operator as well as (ii) the 3rd person support operator,
3. **Digital Twins of the Environment**: Instantiation of the simulated environment based on the robot's percepts, including
   - Online terrain instantiation,
   - Online object instantiation,
4. **Evaluation** of the holistic system ("the system at work"), including
   - Definition of simulation evaluation tasks, and
   - Definition of the "Switch" between direct control of the Real Twin and the (instantiated) Digital Twin.

Progress and results have also been presented in academic publications like [2, 3], as well as in the joint CENTAURO journal article [8] or [4].

2.1 Evaluation - Lessons learned

The developments are based on all prior workpackages as well as the "lessons learned" from the integration and evaluation of the first integrated CENTAURO system (MS3).

Regarding the simulator, these lessons are summarized in the following including references where these topics will be addressed in this deliverable:

1. Most importantly a fully automated update on the robot model (**automated URDF work flow**) is essential to cope with the ongoing changes of the robotic setup (in terms of hands, sensors etc.).
   → addressed in Section 3.1.
2. **Flat visualizations** (like normal PC screens) are sometimes "more useful" or more intuitive than stereoscopically rendered three dimensional models, especially for the (multiple) support operators.
   → addressed in Section 5.3.

3. Stereoscopic rendering requires universal **acceptance** and excellent **performance**. Issues needs to be addressed otherwise the use and the prospects of this technology are limited.
   → addressed in Section 5.4.
4. Looking at a problem from different perspectives is a very promising idea to reduce complexity of a given task. Thus, the **point of view** is essential—simulation can be a benefit here in terms of position and orientation of the viewpoint, for the first and support operator.
   → addressed in Section 5.4.
5. **Bandwidth** can become an issue (especially regarding huge point cloud data updates).
   → addressed in Section 5.1.

2.2 Centauro Control Modes

As presented in deliverables D4.1, D4.2 and D4.3 [5, 6, 1] the core **operational control modes** are still (a) directly controlling the DT, (b) directly control the RT, or (c) the prediction mode where the directly controlled real system is stopped in a safe pose and we switch into directly controlling the DT to test out possible actions (cf. Fig. 1). The direct control can of course either be achieved via ROS or via the exoskeleton control.

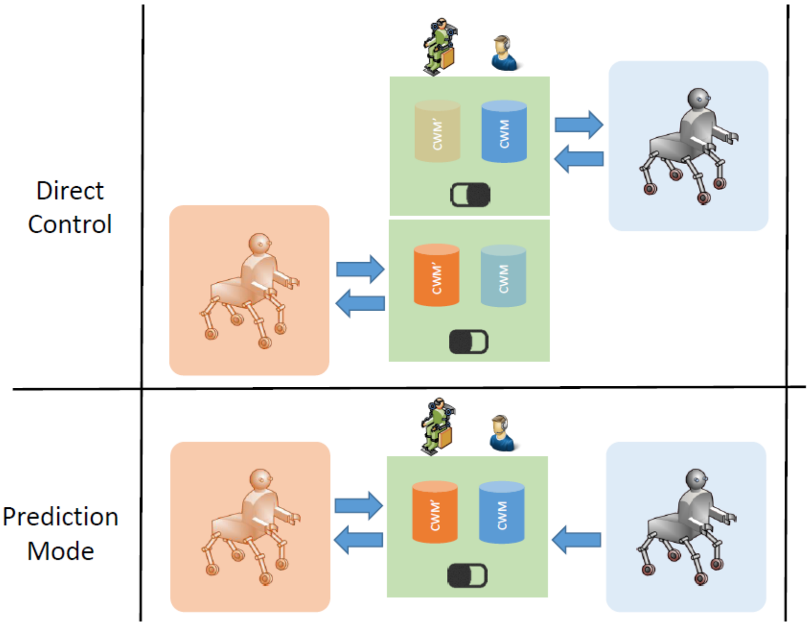


Figure 1: Centauro Control Modes (image based on [9]): (top): Direct Control of either Real or Digital Twin, bottom): Switching from directly operating the RT into directly operating the DT which is acting in a perceived environment based on real sensor data. (The background color orange indicates the DT, blue the RT).

3 The Digital Twin of the CENTAURO Robot

Based on the work of previous work packages (D4.1, D4.2 and D4.3) [5, 6, 1] we enlarged the scope towards the changed requirements of the robotic system, as the Digital Twin is a digital representation of the robot which includes

- interfaces,
- rigid body simulation of the robot, and
- sensor simulation of the sensor head.

This leads to the possibility for external modules—input modules for control, as well as output modules for sensor data processing—to be independent of whether interfacing the RT or the DT.

3.1 Creation of the DT

The work flow for automating the creation of the DT according to the RT (see also D4.1, D4.2 and D4.3 [5, 6, 1]) has been extended. Based on the commonly accepted URDF (Unified Robot Description Format) file the work flow involves

1. URDF import to create a first model M,
2. "flatten" this model M to create a visualization DT model M_v (the hierarchical tree of the URDF file is broken down into a compliant tree structure for the upcoming "dynamization" of the model to integrate it in the rigid body simulation framework),
3. "dynamize" this model M_v to create a dynamic DT model M_d (the model (links and joints) is equipped with actuators (motors) which allow a direct control of the joints),
4. automatically create and append ROS subscriber and publisher to the dynamic model M_d to create model $M_{d'}$, and
5. add pre-defined models of the sensor head (M^j_{camera}, M_{velodyne} and M_{kinect}) to the dynamic model $M_{d'}$ to create the final DT model M_{DT}.

Using this work flow we can quickly add modules or integrate modifications on the CENTAURO robot to its DT. The two models used in this approach are

- the visualization DT model M_v which is purely used for visualizing the current state of the robot mainly used for the 1st person operator and the rendering into the Head Mounted Display (HMD) and
- the final DT model M_{DT} which represents the RT and can be used equivalently.

3.2 Representations of the DT

The DT has different representations depicted in Fig. 2. Based on the URDF file we automatically generate the Extensible Markup Language (XML) file compliant with the VEROSIM file standard. Additionally, the 3D model shows directly the visual representation of the robot, whereas the "IOBoard" represents the functional system-theoretical data flow of the system in terms of input and output of all components.

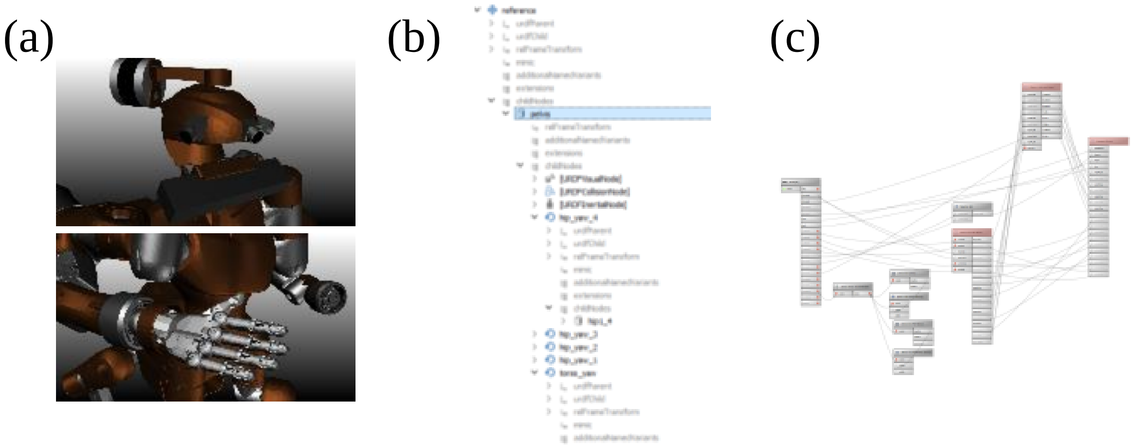


Figure 2: Representations of the DT: (a) 3D view, (b) XML structural view and (c) functional data flow view in the IOBoard.

3.3 Centauro Hard- and Software Setup

Besides the control modes of the CENTAURO robot, the hardware and software setup regarding the simulator is shown in Fig. 3.

A CENTAURO mission consists of the CENTAURO robot (RT), modules for incoming and outgoing data and the Digital Twin (DT). On the operator side the DT is used for visualization purposes and has a reduced complexity (cf. Section 3.1). Additionally, on the robot side (which is not meant spatially) there needs to be a DT instantiated from the RT percepts when needed. Although this instance could also run on the robot hardware, we will instantiate the DT on a computer at the operator control station where we already have all necessary data of the DT used for visualization. A detailed description of this "switch" can be found in Section 7.2.

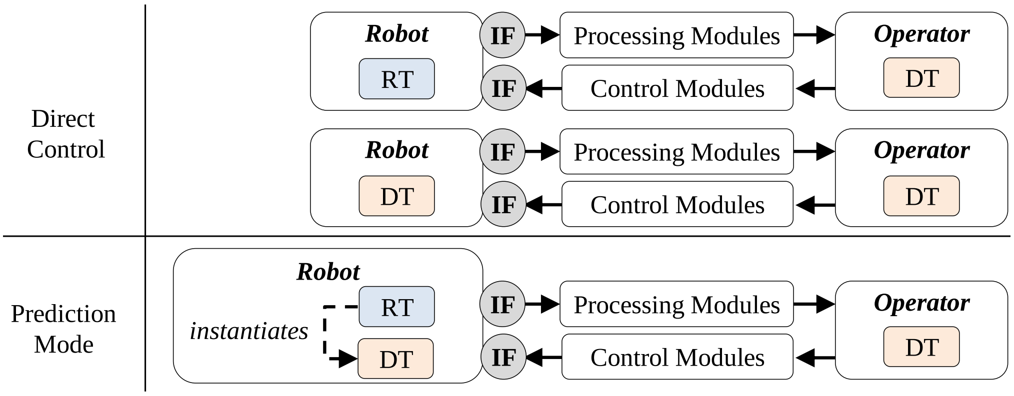


Figure 3: Centauro Hard- and Software Setup in Terms of Data Flow (w.r.t. Fig. 1): (i) Using the DT for visualization on the operator side, (ii) directly controlling either RT or DT by means of the same in- and output data streams and (iii) the switch of instantiating a DT based on the RT data for prediction mode.

3.4 ROS Interface

The overall structure of the ROS interface of the simulator is still the same as described in previous deliverables. We used the modular implementation scheme motivated in D4.1, D4.2 and D4.3 [5, 6, 1] and extended the functionalities towards the needs of all project partners. This led to the following functionalities:

- std_msgs/*
  Float64, Bool, Int, etc.
- sensor_msgs/*
  JointState, Image, CompressedImage, PointCloud2, etc.
- tf2_msgs/*
  tfMessage
- rosgraph_msgs/*
  Clock
- nav_msgs/*
  Path
- centauro_msgs/*
  HeightMap, ModelPose, DrivingMovement, LegMovement, TexturedPolygonMesh

Depending on the use case, publisher and/or subscriber of these message types have been implemented, with respect to the CENTAURO setup (see also Fig. 3). For the DT used for visualization the simulator has to subscribe to everything the robot publishes, whereas the final DT model has to produce and publish all sensor data in the simulator for the processing nodes in the system. Thus, each DT has other requirements for the type and number of ROS nodes used.

3.5 XBotCore Interface

Based on the developments in D4.3 [1] a simulator abstraction layer has been initialized to use either Gazebo or VEROSIM or whatever simulator is preferred. Due to heavy developments on the XBotCore interface these development could not be merged into the trunk yet but are still one option to interface the DT, if merged.

Besides this integration of XBotCore into the simulator it is now also possible to use the commonly accepted ROS standard directly. This means a direct mapping of subscribed ROS commands onto the actuator motors as well as publishing the current robot's state accordingly. This led to the implementation of the XBotCore specific ROS msg types:

- xcm/ADVRJointState (publisher)
- xcm/ADVRJointCommand (subscriber)

By using these, it is directly possible to access all actuators of the DT which is capable of publishing its current state as well.

4 User Interface

This section summarizes the user interface which has been developed throughout the whole project.

The user interface (UI) of the man-in-the-loop system of the CENTAURO project consists of in- and output between the user and the robot (and its DT). There are two types of operators: the *first person operator (FPO)* and the *support operator (SO)*. In case of the FPO we will use an exoskeleton for control and the HTC Vive for visualization in a “3D immersive view”, rendered via VEROSIM.[1]

The SO can either support the FPO by “toggling” different visualizations or by rearranging the operators view point etc. On the other hand, the SO can also directly control the robot via ROS control. The input devices of the SO can vary, e.g., keyboard, 6D space mouse etc. Visualization is realized either through external tools (like command line output) or through VEROSIM in a “2D mode” on standard PC screens, whereas it is also possible for the SO to use the 3D immersive view.

The SO can then also use the simulator in the loop to switch and predict actions (see also Section 7.2). Besides control, exoskeleton feedback and visual feedback it is also possible to use an independent ROS node for audio feedback recorded by the Kinect v2 sensor.

To summarize the UI,

- we will have two operators with different control devices and middlewares (exoskeleton (UDP), direct control with standard PC input devices (ROS)),
- visualization can be done either in 3D immersive mode or in 2D mode rendered in VEROSIM aided by external ROS tools, and
- the support operators either support the 1st person operator or directly control the RT or DT via ROS.

---

[1] Although the usage of an HMD is favored for the FPO, it is theoretically also possible to render everything in the “2D mode” on (multiple) LCD displays as well. This option should only be used if a “simple” solution is necessary and the operator is not able to cope with the immersive experience and a successful execution of the given task is jeopardized.

5 Visualization and Rendering

In general, we already presented the rendering pipeline for example in D4.2 [6] using ROS as the commonly accepted middleware to transmit generated and processed data. As described in Section 3.4, we primarily use standard ROS sensor message types, accompanied by custom message types for processed vision data. The **Visualization and Rendering** includes

- visualization of the current robot state,
- visualization of raw sensor data,
- visualization of pre-processed sensor data (output of WP5, WP6), and
- monoscopic and stereoscopic rendering of all this data.

Based on the developments presented in D4.2 [6] and D4.3 [1] we want to focus on continuous expansion of this concept here.

5.1 Rendering Performance

Due to the necessity of pausing the incoming point cloud to update the render process, we established a flag on every ROS subscriber that toggles its activity. Thus, it is possible to pause the updates on point cloud data if necessary. During evaluation one could clearly see that the point cloud was almost static and changes were negligible small.

Additional performance evaluation has been conducted regarding the Kinect v2 textured mesh presented in the following sections.

5.2 Textured Mesh

The generalized renderer represents the already developed **Kinect renderer** (Fig. 4) as well as the unit sphere rendering approach of (LIU) which we call **LIU renderer** (Fig. 5).

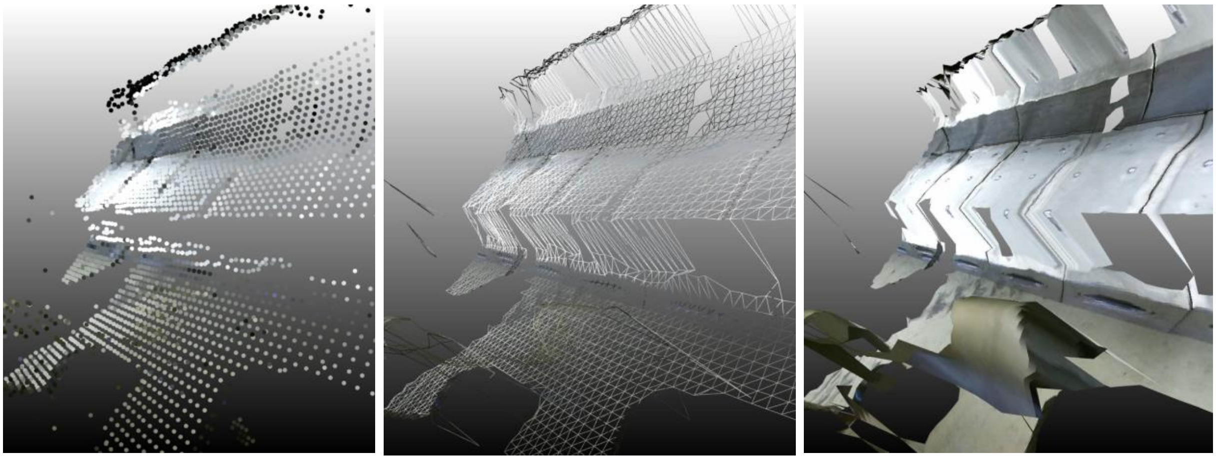


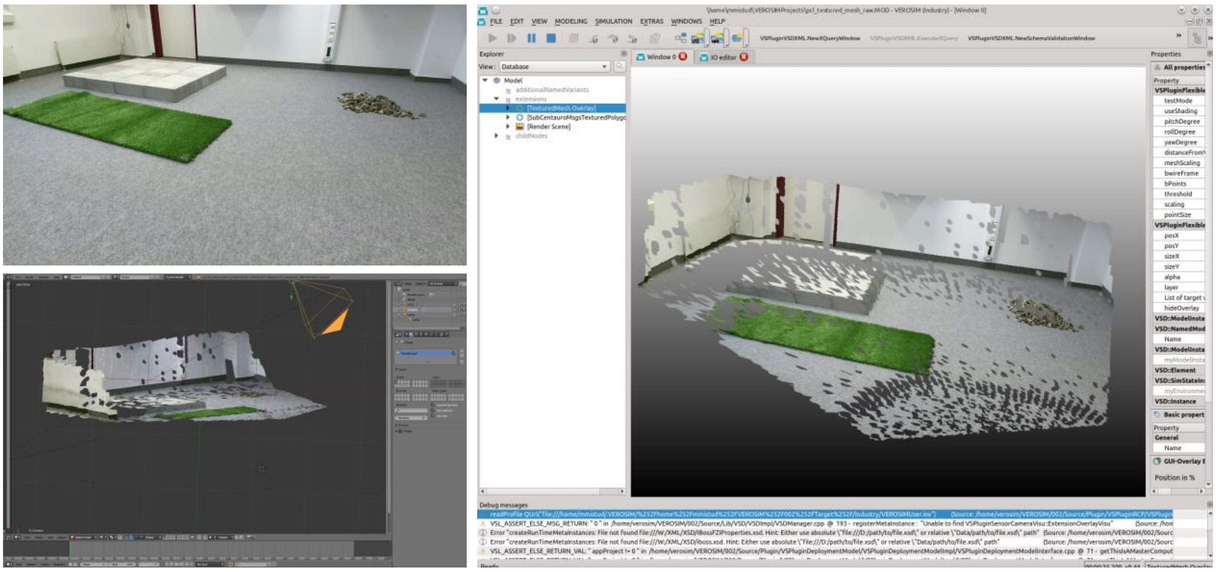
Figure 4: Textured Mesh Kinect Renderer: Adding point cloud (l.), mesh (c.), and texture data (r.).

The Kinect renderer is based on developments made by (UBO) and integrated, optimized and extended by us. This renderer is now capable of using either SD, QHD or HD Kinect data (point clouds as well as image data) as well as mixtures of these.

The generalized rendering approach by (LIU), based on [7], will be a standalone ROS node that handles (and fuses) the pre-processing of sensor data from the Velodyne, Kinect, and the three RGB cameras. The consortium decided on the following customized ROS msgs types:

- *centauro_msgs/Polygons*,
- *centauro_msgs/SubMeshTexCoords*,
- *centauro_msgs/SubMeshVerts*,
- *centauro_msgs/TexturedPolygonMesh*, and
- *centauro_msgs/TextureMesherInput*

where the **input** of the system is represented by the *TextureMesherInput* data type and consists of the point cloud, image, and camera info data. The **output** of the system (and thus the input to the simulator) is represented by the *TexturedPolygonMesh* which especially comprises the point cloud, vertices, texture coordinates (u,v) and images. These can then directly be rendered in the simulator (or any other renderer used), as shown in Fig. 5.

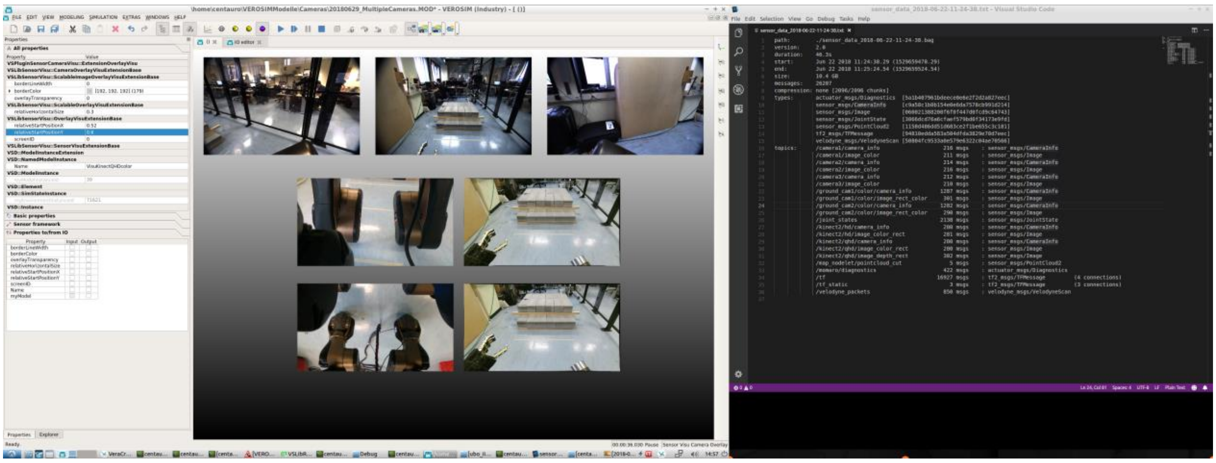


```
readProfile QUrl("file:///home/mmistud/VEROSIM/%252Fhome%252Fmmistud%252FVEROSIM%252F002%252FTarget%252F/Industry/VEROSIMUser.inx")     (Source: /home/verosim/VEROSIM/002/Source/Plugin/VSPluginRCP/VSPlugin
VSL_ASSERT_ELSE_MSG_RETURN: " 0 " in /home/verosim/VEROSIM/002/Source/Lib/VSD/VSDImpl/VSDManager.cpp @ 193 - registerMetaInstance : "Unable to find VSPluginSensorCameraVisu::ExtensionOverlayVisu"     (Source: /hom
Error "createRunTimeMetaInstances: File not found file:///W:/XML/XSD/IBossFZIProperties.xsd. Hint: Either use absolute \"file:///D:/path/to/file.xsd\" or relative \"Data/path/to/file.xsd\" path"  (Source: /home/verosim/VEROSIM/002/Sourc
Error "createRunTimeMetaInstances: File not found file:///W:/XML/XSD/iboss.xsd. Hint: Either use absolute \"file:///D:/path/to/file.xsd\" or relative \"Data/path/to/file.xsd\" path"     (Source: /home/verosim/VEROSIM/002/Sourc
VSL_ASSERT_ELSE_RETURN_VAL: " appProject != 0 " in /home/verosim/VEROSIM/002/Source/Plugin/VSPluginDeploymentModel/VSPluginDeploymentModelImpl/VSPluginDeploymentModelInterface.cpp @ 71 - getThisIsAMasterComput
```

Figure 5: Textured Polygon Mesh LIU Renderer: Generally meshed input data visualized in the simulator. (top left) RGB Image, (bottom left) Meshed object file in Blender and (right) TexturedPolygonMesh in VEROSIM.

5.3 Support Operator View

Based on the evaluation at (KHG), the need for classical visualization options was substantial. Thus, we added the feature of placing camera image streams arbitrarily on the screen shown in Fig. 6.



```
path:        ./sensor_data_2018-06-22-11-24-38.bag
version:     2.0
duration:    46.3s
start:       Jun 22 2018 11:24:38.29 (1529659478.29)
end:         Jun 22 2018 11:25:24.54 (1529659524.54)
size:        10.4 GB
messages:    28287
compression: none [2096/2096 chunks]
types:       actuator_msgs/Diagnostics  [5a1b407961bdeece0e6e2f2d2a827eec]
             sensor_msgs/CameraInfo     [c9a58c1b0b154e0e6da7578cb991d214]
             sensor_msgs/Image          [060021388200f6f0f447d0fcd9c64743]
             sensor_msgs/JointState     [3066dcd76a6cfaef579bd0f34173e9fd]
             sensor_msgs/PointCloud2    [1158d486dd51d683ce2f1be655c3c181]
             tf2_msgs/TFMessage         [94810edda583a504dfda3829e70d7eec]
             velodyne_msgs/VelodyneScan [50804fc9533a0e579e6322c04ae70566]
topics:      /camera1/camera_info                      216 msgs    : sensor_msgs/CameraInfo
             /camera1/image_color                      211 msgs    : sensor_msgs/Image
             /camera2/camera_info                      214 msgs    : sensor_msgs/CameraInfo
             /camera2/image_color                      216 msgs    : sensor_msgs/Image
             /camera3/camera_info                      212 msgs    : sensor_msgs/CameraInfo
             /camera3/image_color                      210 msgs    : sensor_msgs/Image
             /ground_cam1/color/camera_info           1287 msgs    : sensor_msgs/CameraInfo
             /ground_cam1/color/image_rect_color       301 msgs    : sensor_msgs/Image
             /ground_cam2/color/camera_info           1282 msgs    : sensor_msgs/CameraInfo
             /ground_cam2/color/image_rect_color       298 msgs    : sensor_msgs/Image
             /joint_states                            2138 msgs    : sensor_msgs/JointState
             /kinect2/hd/camera_info                   700 msgs    : sensor_msgs/CameraInfo
             /kinect2/hd/image_color_rect              701 msgs    : sensor_msgs/Image
             /kinect2/qhd/camera_info                  700 msgs    : sensor_msgs/CameraInfo
             /kinect2/qhd/image_color_rect             700 msgs    : sensor_msgs/Image
             /kinect2/qhd/image_depth_rect             702 msgs    : sensor_msgs/Image
             /map_nodelet/pointcloud_cut                 5 msgs    : sensor_msgs/PointCloud2
             /momaro/diagnostics                       422 msgs    : actuator_msgs/Diagnostics
             /tf                                     16927 msgs    : tf2_msgs/TFMessage         (4 connections)
             /tf_static                                  3 msgs    : tf2_msgs/TFMessage         (3 connections)
             /velodyne_packets                         850 msgs    : velodyne_msgs/VelodyneScan
```

Figure 6: Flat Visualization of camera input using VEROSIM (also possible to arrange views in a 3D scene). Presented here are the three RGB cameras, two additional cameras below the robot pelvis and the Kinect v2 images in HD and in QHD resolution.

Additionally, the visualization of different options (for locomotion or manipulation) is a great advantage. These have already been presented in D4.3 Section 5.1.

5.4 First Person Operator View

Based on D4.3 Section 6.1 and the evaluation of the first person operator, we implemented:

- flat visualizations in VR, and
- free and pre-defined positioning of the point of view (using a “node magnet extension”).

Flat visualizations can either be shown on a normal PC screen or rendered into VR as virtual screens. These screens can be positioned freely in the VR space but for camera images it sometimes makes sense to project these images into their view plane (see Fig. 7). It is also possible to position camera images which are acquired below the robot in line of sight of the operator wherever he demands.

Thus, it is possible to have the **full immersive user experience of an HMD**—especially useful for 3D models, point clouds and in general three-dimensional data—combined with the **familiar 2D screens**—like RGB camera images or status monitors—which can even be projected at a 3D pose where they are most valuable.

The origin of the HMD has a null frame $F^{\text{HMD}}_{\text{null}}(p, O)$ including position p and orientation O. This frame is normally positioned within the robot head to enhance the immersion of the operator in the exoskeleton. As this frame is not fixed to this position (and orientation) it is also possible to modify this frame during runtime to F^{HMD}_{i} for $i = 1 \cdots N$. For a smooth transition from F^{HMD}_{i} to F^{HMD}_{i+1} we implemented a so-called “node magnet extension”. If enabled, this executes the transformation from frame i to $i + 1$ only based on a timing property t [s]. Based

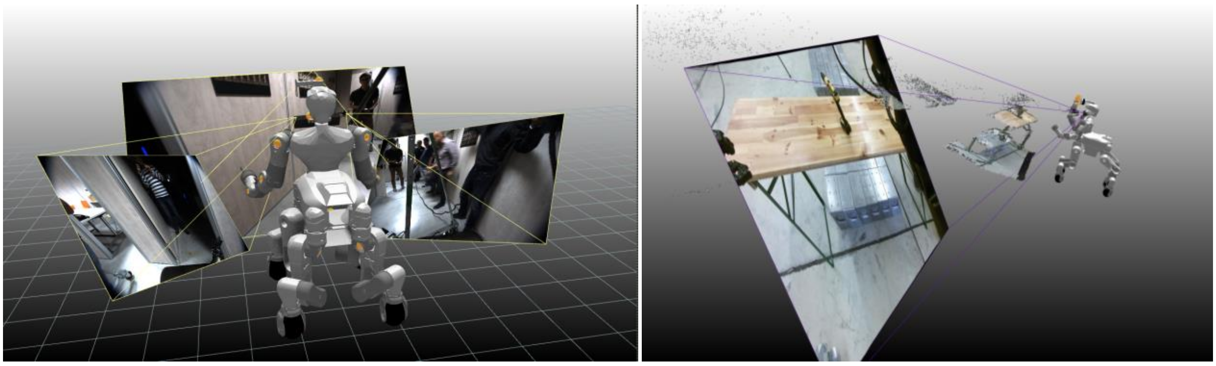


Figure 7: Visualizations for the first operator: (left) flat camera images visualized at the side of occurrence rendered stereoscopically and (right) modifications on the null frame of the HMD for a better understanding of the situation.

on experience or expert knowledge, it is possible to define relative frames (relative to the robot head and the null pose) prior to the mission, like (a) below the base of the robot, (b) in the left or (c) right hand, (d) facing the rear of the robot etc. Additionally, we can add more frames during runtime based on the current situation, e.g., in manipulation tasks to see the object of interest from all sides. Of course, this should be done by the support operator. For this reason we also implemented a “keyboard key listener extension” where we can connect each keyboard key to a digital input in the simulator. Thus, we can change the frames by pressing the associated key.

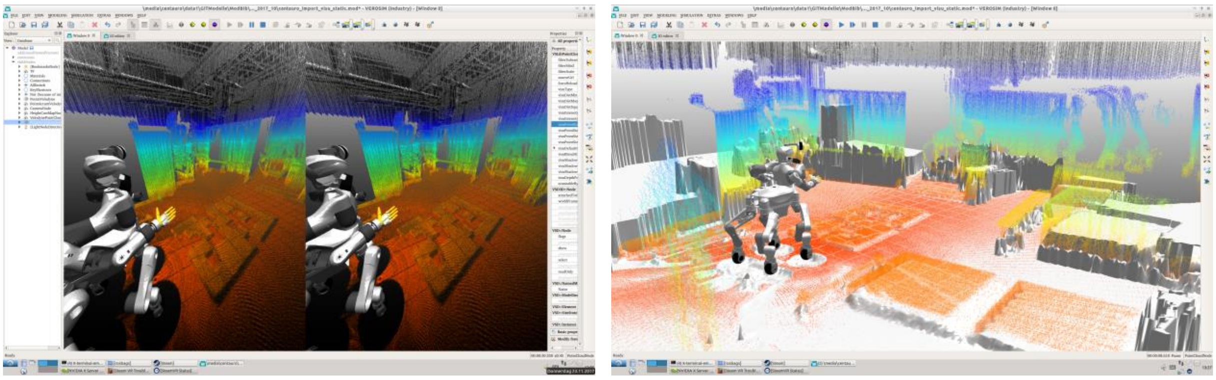


Figure 8: Sterescopic and monoscopic rendering of the robot state, the registered point cloud and the height field.

In general, the use of the presented integration scheme of HMDs in the simulator presented in D4.3 Section 5.2.3 has already been used in the evaluation of the first integrated CENTAURO system (cf. Fig. 8). Still, *OpenVR* and *SteamVR* are not supported completely under Linux, yet. Resulting performance issues could already been reduced by means of decoupling the render refresh rate and the rate of incoming data streams but still require continuous optimization.

6 Digital Twins of the Environment

Besides static pre-defined scenarios, dynamic environments comprise two different aspects of the instantiation of the simulated environment based on the robots percepts:

1. Online terrain instantiation and
2. online object instantiation based on pre-defined object models.

The generation of this environment is of course limited to the sensors used. For environments, we use height maps and for the object instantiation we use template-based model insertion.

In short, the terrain height map work flow is:

$$\text{Robot} \xrightarrow[ROS]{data} \text{WP5} \xrightarrow[ROS]{centauro\_msgs/HeightMap} \text{WP4} \rightarrow$$

$$\rightarrow \text{WP4} \xrightarrow[internal]{} \text{Rigid Body HeightField}$$

For the template-based model insertion the work flow is:

$$\text{Robot} \xrightarrow[ROS]{data} \text{WP6} \xrightarrow[internal]{object\ detection} \text{WP6} \xrightarrow[ROS]{centauro\_msgs/ModelPose(frame,name)} \text{WP4} \rightarrow$$

$$\rightarrow \text{WP4} \xrightarrow[internal]{name} \text{model(name)} \xrightarrow[internal]{model,frame} \text{Rigid Body Model (frame, name)}$$

Specific details about these can be found in D4.3 Section 6.2.

The terrain height map work flow was also evaluated during for visualization during the evaluation camp (see also Fig. 8) and for dynamic environment generation during an integration meeting between (RWTH) and (UBO). Here we could use the Momaro sensor head to generate raw data, process this data into a height map and generate a rigid body height field within VEROSIM. The result can be seen in Fig. 9.

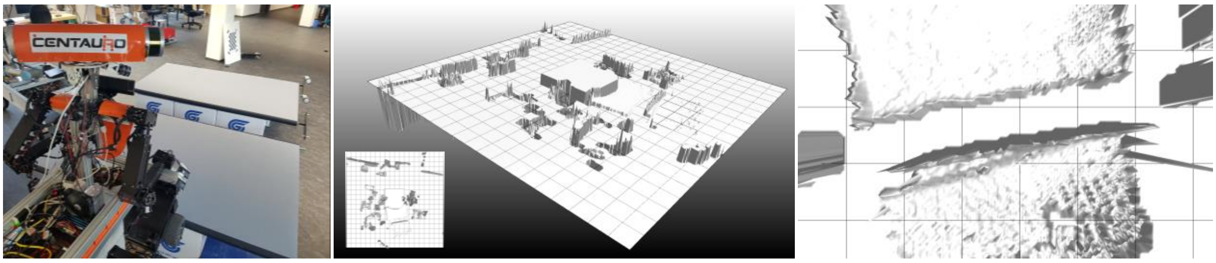


Figure 9: Heightfield generation in the simulation evaluation task S1 “stepping over a gap”: Using the Momaro sensor head to generate a heightfield which is converted into a rigid body in simulation.

7 Evaluation

Besides the graphical rendering of all data (raw or processed) the main goal of using the simulator in the loop is to 'try out' different options with the DT in simulation first before executing them using the RT. Thus, switching online into using the DT instead of the RT is the main application here.

7.1 Simulation Evaluation Tasks

Evaluating the simulation the consortium decided to have one main (and one additional optional) evaluation task. For a holistic evaluation of the capabilities the main evaluation tasks are:

S1	Locomotion:	*Stepping over a gap*	(w.r.t. D8.3 Locomotion task L5)
S2	Manipulation:	*Grasping a drilling tool* (optional)	(w.r.t. D8.3 Manipulation task A1)

Based on **S1 Locomotion** the approach is:

- Try to step over the gap in simulation with arms in front of the robot upper body
  → Center of mass is in a bad position,
  → Robot falls over
- Try to step over the gap in simulation with arms behind the robot upper body
  → See if it works
- Execute the second operation with the real robot
  → Success

Based on **S2 Manipulation (optional)** the approach is:

- Calculate different approaches to grasp a drilling tool
  → Visualization of these multiple approaches in the simulator
- Choose the best suited approach and execute it in simulation first
  → See if it works
- Execute the approach with the real robot
  → Success

We can then evaluate, if the expected success and fail assumptions can be confirmed or negated. Additionally, we can evaluate if the successful operation in simulation can also successfully be conducted in reality.

7.2 Switch into Prediction Mode

The "Switch" of directly controlling the real robot to directly controlling its Digital Twin requires different steps (see also Fig. 1). These steps were conducted in cooperation with UBO as a first conceptual draft how this could be accomplished:

1. Pause the real system (in a safe state).
    (a) Stop the real system listening to commands.

    (b) Stop the real system publishing joint states/ sensor data.

2. Instantiate a DT in the simulation.
3. Construct a virtual world (rigid body simulation).
4. Use the DT (final DT model).
    (a) Start the DT listening to commands.
    (b) Start the DT publishing joint states/ sensor data.

These steps result in the *Prediction Mode* where we use the DT identically to the RT, using the same input devices, the same commands, resulting in the same kind of feedback from the system.

To physically execute this ”Switch” we need to take all interfaces into account. Simply said, we just have to redirect all messages from and to the robot to the simulator. In Fig. 10 this scenario is described in more detail. Besides the general ROS commands which can be redirected, additional switches are required for (i) the robot (IIT), (ii) the ROS control toolbox (UBO), (iii) the exoskeleton (SSSA) and (iv) the simulation itself (RWTH). Subscribing processing modules should continue to work as before (LIU, KTH). For the exoskeleton SSSA would have to redirect the UDP packets to the operator station instead of the mobile robot. For the simulator we instantiate the DT model and use the modular I/O board where we can easily toggle an enabled/disabled flag for all relevant components. The presented switch in Fig. 10 has been tested in a bilateral integration meeting of (UBO) and (RWTH). The direct control of the DT, the instantiation of static and dynamic environments has been evaluated in the gap task (see Fig. 11). The switch itself has been integrated in the UBO ROS control software module which redirects control messages onto new topics using the namespace *“/verosim/”*.

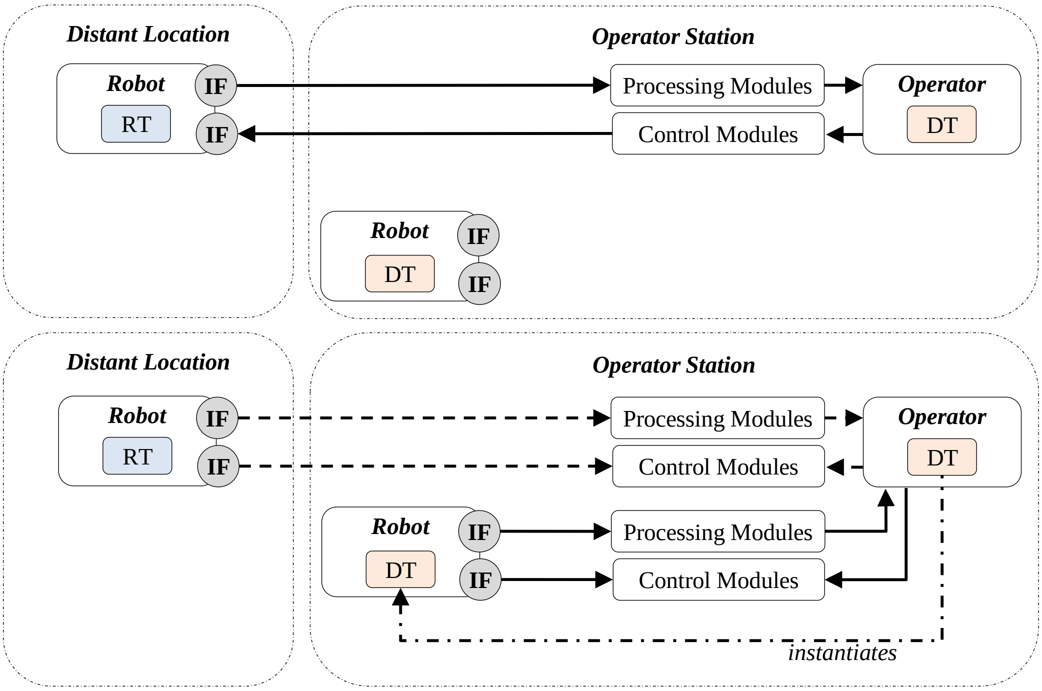
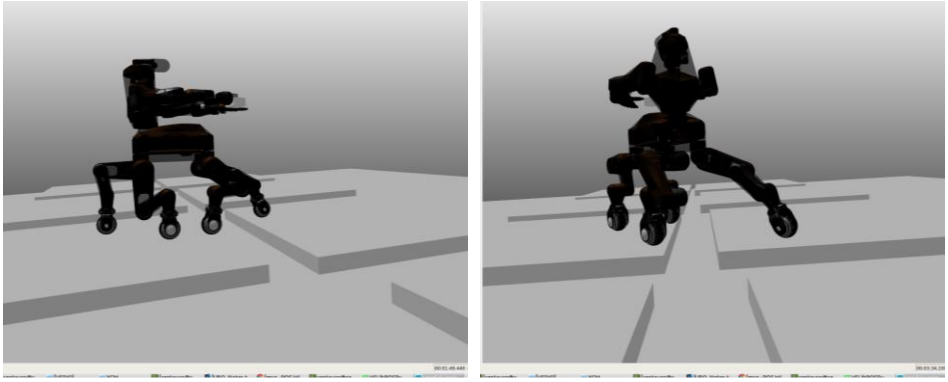
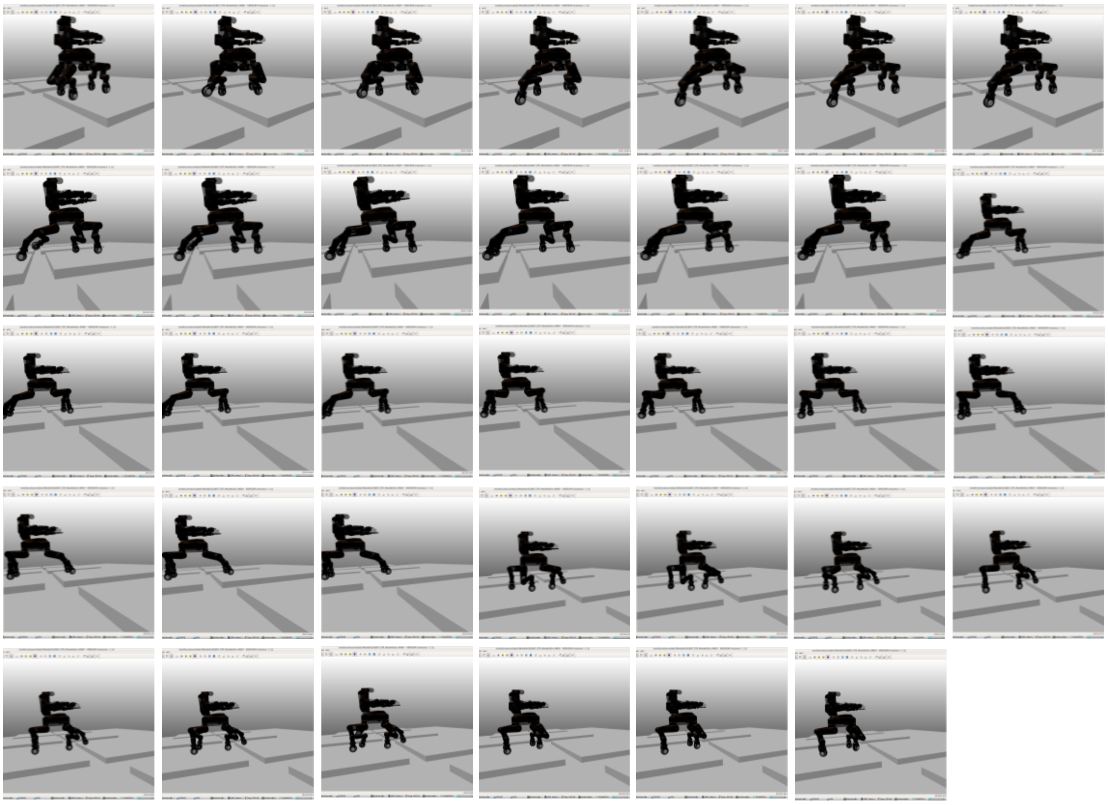


Figure 10: “Switch” from (top) directly controlling the RT to (bottom) the prediction mode (w.r.t. Fig. 3 and Fig. 1) considering the spatial relation of the robotic system and the control station.



(a) Evaluating different stepping procedures



(b) Process of stepping over a 30 cm gap over time

Figure 11: Using the switch in the simulation evaluation task S1 “stepping over a gap”.

8 Conclusions

Taking everything into consideration, we are able to generate a holistic Digital Twin of the CENTAURO robot. The automated URDF-import can be used to update and keep the link between the real robot and its Digital Twin. The generation of environments and associated DTs can be done online. Useful visualizations through textured meshes and RGB images were generated for the first operator as well as for the support operator (based on user requirements). The switch between reality and virtuality is established but requires additional testing throughout the next months. Additionally, evaluation scenarios for simulation have been defined. These can be tested in a pure virtual environment already before the final evaluation with the help of the DT and processing modules of other project partners. Thus, the next steps mainly involve fine tuning of the required ROS interface, performance optimization on the message passing of ROS and the stereoscopic rendering. Additionally, onsite training of the first operator (w.r.t. T3.5) are planned to underline the necessity, benefits and to optimize the usability of the HMD as well as training of the support operator mainly in terms of the evaluation task S1 is necessary.

References

[1] Torben Cichon. Deliverable D4.3 constructing simulated world from robot percepts.

[2] Torben Cichon and Jürgen Roßmann. Robotic teleoperation: Mediated and supported by virtual testbeds. In *Safety, Security, and Rescue Robotics (SSRR), 2017 IEEE International Symposium on*. IEEE, 2017.

[3] Torben Cichon and Jürgen Roßmann. Simulation-based user interfaces for digital twins: Pre-, in-, or post-operational analysis and exploration of virtual testbeds. In *Proceedings of the 31st European Simulation and Modelling Conference (ESM 2017), October 25-27, 2017, IST, Lisbon, Portugal*. IEEE, 2017.

[4] Torben Cichon and Jürgen Roßmann. Digital twins: Assist and support cooperation in human-robot teams. In *International Conference on Control, Automation, Robotics and Vision*. ICARCV, 2018. (not published yet).

[5] Torben Cichon and Christian Schlette. Deliverable D4.1 virtual testbed concept.

[6] Torben Cichon and Christian Schlette. Deliverable D4.2 simulation of CENTAURO robot and environment.

[7] Dirk Holz and Sven Behnke. Fast range image segmentation and smoothing using approximate surface reconstruction and region growing. In *Intelligent autonomous systems 12*, pages 61–73. Springer, 2013.

[8] Tobias Klamt, Max Schwarz, Christian Lenz, Lorenzo Baccelliere, Domenico Buongiorno, Torben Cichon, Xi Chen, Antonio Di Guardo, David Droeschel, Massimuliano Gabardi, Karl Holmquist, Felix Järemo-Lawin, Malgorzata Kamedula, Navvab Kashiri, Arturo Laurenzi, Daniele Leonardis, Luxa Muratore, Dmytro Pavlichenko, Arul Selvam Periyasamy, Andreas Robinso, Diego Rodriguez, Fabisn Schilling, Massimuliano Solazzi, Michael Felsberg, John Folkesson, Antonio Frisoli, Michael Gustmann, Patrix Jensfeld, Klas Nordberg, Jürgen Roßmann, Uwe Süss, Nikos G. Tsagarakis, and Sven Behnke. Solving disaster-response tasks through the centaur-like robot centauro: Full-body telepresence and autonomous operator assistance. 2018. (not yet published).

[9] European Commission Directorate General for Communications Networks, Content and Technology Components and Systems: Robotics. Grant Agreement 644839: CENTAURO—Robust Mobility and Dexterous Manipulation in Disaster Response by Full-body Telepresence in a Centaur-like Robot, 2014.

Digital Twins:
Assisting and Supporting Cooperation in Human-Robot Teams

Torben Cichon and Jürgen Roßmann
Institute for Man-Machine Interaction (MMI)
RWTH Aachen University
52074 Aachen, Germany
{cichon,rossmann}@mmi.rwth-aachen.de

***Abstract*— To create an intuitive access to the robotic system at hand, we propose to use its Digital Twin (DT)—a virtual copy of the used robot with all necessary details of the mechatronic system, its construction, function, use and application. To do so, these DTs are embedded into a Virtual Testbed (VTB) (a software framework for cross-system, -discipline, and -application development on a system level) to gain insight into complex system by having a bidirectional online data stream and interaction between human, DT, and Real Twin (RT). Intuitive user interfaces, the DT, and hardware interfaces are the requirements for this and can be used to fuse reality and virtuality to develop, monitor, and control the robotic system through the DT. Such DTs can then be used in various applications, here focusing on mobile robots for catastrophic scenarios where an effective coupling of human and robot skills is inevitable for a successful mission.**

I. INTRODUCTION

Manufacturers worldwide are building sensors and communications into their devices to collect online real-time data. The most advanced are feeding this data into so-called “Digital Twins” (DTs), creating feedback loops between in-use devices and the simulation tools used to create them. This idea of using the already implemented virtual footprint of the system not only during the development but also in production of the real system shows great potential.

Applications arise from various areas like standard production environments up to holistic mobile robotic systems. Although DTs are being advertised in all engineering domains they are still limited to predefined animations. Augmented data of a system is based on static information (like a handbook) and changes of the system’s setup require a complete redesign of the DT. One central aspect of our methodology is *adaptation*. The DT of the robot itself has to adapt to changes of the RT, its environment and vice versa. The run-time environment needs to be flexible enough allowing multiple different interacting DTs.

II. THE DIGITAL TWIN

While the term Digital Twin (DT) is often confused with a 3D Computer-Aided Design (CAD) model, in reality, the DT is significantly more complex. The fallacy here lies in confusing the twin model with a simulation model. For a DT to count as such, it needs a *physical counterpart*—the Real Twin (RT)—with which the DT can *interact*. DTs are being developed still on the basis of CAD models and animation or single-domain simulation software tools. A holistic solution which connects domains, application, and most importantly the virtual and real system is still missing. Thus, we propose embedding the DT in a Virtual Testbed (VTB) and utilize this framework then online to assist and support the human working cooperatively with robotic systems side by side, as well as spatially separated. In this contribution the central application is a teleoperated mobile robot, accompanied by its DT, to assist the operator in his decision making process and to support an efficient exploitation of skills in this cooperative hybrid team of human and robot (see Fig. 1).

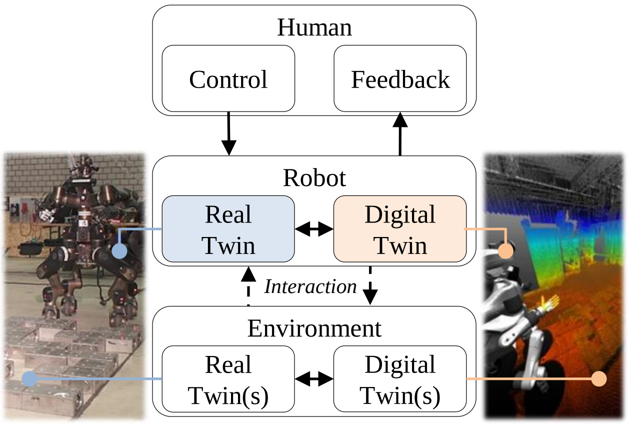


Fig. 1: Using the Digital Twin online to support human-robot cooperation: Interfacing the robot (in terms of direct control and feedback), no matter if real or digital, interacting with Real or Digital Twins of the environment

Based on [1] one can define the transition from simulation model to the DT model as follows. Let the Digital Twin model M have an input $\underline{\mathbf{u}}(t)$ and an output $\underline{\mathbf{y}}(t)$ then there is a mapping in terms of $\underline{\mathbf{y}}(t) = M(\underline{\mathbf{u}}(t))$. Then the naive goal of simulation would be that the real system ($\underline{\mathbf{y}}^{\mathrm{RT}}(t) = M^{\mathrm{RT}}(\underline{\mathbf{u}}^{\mathrm{RT}}(t))$) and the virtual system ($\underline{\mathbf{y}}^{\mathrm{DT}}(t) = M^{\mathrm{DT}}(\underline{\mathbf{u}}^{\mathrm{DT}}(t))$) yield the same result:

$$||\underline{\mathbf{y}}^{\mathrm{RT}}(t) - \underline{\mathbf{y}}^{\mathrm{DT}}(t)|| \rightarrow 0 \tag{1}$$

But more importantly, it is necessary that algorithms (A) using data from the DT ($\underline{\mathbf{y}}^{\mathrm{DT}}(t)$) yield the same result as using data from the RT ($\underline{\mathbf{y}}^{\mathrm{RT}}(t)$). Thus, the main intention for the DTs is that for all algorithms $\underline{\mathbf{z}}(t) = A(\underline{\mathbf{y}}(t))$:

$$||A(\underline{\mathbf{y}}^{\mathrm{RT}}(t) - A(\underline{\mathbf{y}}^{\mathrm{DT}}(t))|| \rightarrow 0 \tag{2}$$

The important message of Eq. (2) is that for processing modules using the robot as a data source it does not matter any longer if the modules are connected to the DT or the RT. Same goes for control modules when using the same input interface to DT and RT.

With this approach of a hybrid system of human and robot, the Virtual Testbed becomes the run-time environment for Digital Twins. Within the VTB the DTs can interact with each other and their real counterparts becoming a network of interacting DTs. These DTs can then interact within the VTB on a multi-domain basis (cf. [2]).

III. HUMAN ASSISTANCE SYSTEMS

For an intelligent hybridization of man and machine capabilities assistance systems have been developed to compensate for the weaknesses of both. There are five main types of support: *physical*, *psychological*, *cognitive*, *organizational*, and *communicative*. Support systems can assist in each of these domains. Additionally, there are three determinants of classifying support systems: *tempo-spatial*, *type of coupling*, and *supervision* (Fig. 2), whereas the type of coupling can be *haptic*, *visual*, or *acustic*.

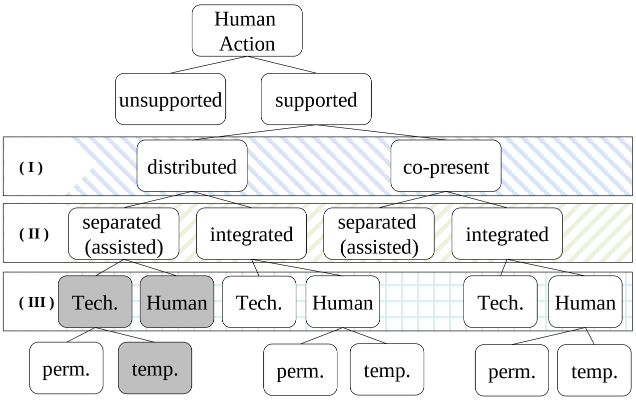


Fig. 2: The three determinants of classifying support systems (based on [3]): (I) tempo-spatial relation, (II) type of coupling, and (III) supervision (who is in control?). Highlighted is the option set of the application presented in Fig. 4.

The human is one essential part in this system of Real and Digital Twins. With the model of human information processing (Fig. 3), supporting a human should focus on the *perception*, *cognition*, or *action*. As one can see, support systems are spanning across various physiological and scientific disciplines.

A. Motoric/Physical Support

In human-robot cooperation a co-present and integrated solution to combine human intelligence and robot power efficiently are so-called exoskeletons. Directly attached to the human, the exoskeleton supports the human muscles in various disciplines. A general overview of these hybrid human robot teams is given in [6] and [7]. Specific support of the lower extremities is given in [8], where the human is supported in squatting, bending, swinging from side to side, twisting, and walking as well as stepping while carrying equipment and supplies. Support of the upper extremities is presented in [9] or [10] for either physically impaired people or as a support of manual assembly tasks. Instead of this direct physical coupling one can also support the human through such exoskeletons in a distributed and separated manner. Utilizing exoskeletons as force feedback input devices to couple human actions with teleoperated robotic systems (or their DTs) is presented in [11].

B. Perceptual Support

Besides the physical support there are also various ways of a perceptual support. Haptic assistance can be achieved by using force feedback input devices as presented before. Although there are five human senses, 80% of all information is derived from vision [12]. As the human sight is limited in wavelength and light requirements one can overcome this shortcomings by robotic cognition. Two mentionable examples are the compensation of the sense of hearing by visualizing sound sources and the visualization of infrared heat vision both presented in [5].

C. Cognitive Support

To support the cognition of the human operator the main intention is to reduce his workload and to give him easy access to everything he demands. Thus, the cognitive support is also connected to the aforementioned *visual preparation* of incoming data streams. Of course, there are a manifold of examples of visual support systems in the automotive sector like blind spot monitoring, distance control, navigation system, traffic-sign recognition, speed control, various warning systems etc. These can of course be transferred to mobile robots and their control by Augmented Reality (AR) metaphors [13]. Another way of cognitive support is *autonomy*, either permanent or temporary. In teleoperative systems it is often helpful to have a set of (semi-)autonomous predefined motion patterns like grasping or maneuvering operations from point a to point b. Instead of permanent autonomy these actions are often supervised by the human as presented in [14] or [15].

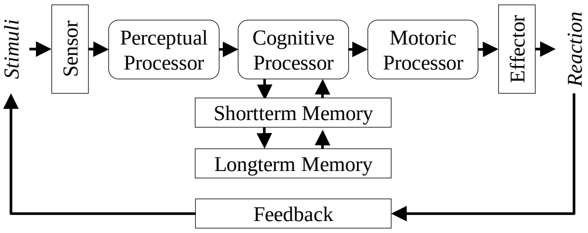


Fig. 3: Model of the human information processing (based on [4], [5]): *perception*, *cognition*, and *action*. In all these dimensions the human can be supported/assisted.

IV. REQUIREMENTS

To establish the DT as the central mediator for assistance systems there are several requirements the DT has to fulfill:

- Holistic system's life cycle support, based on the DT,
- data flow from RT to DT and from DT to RT,
- (bi-directional) connection of operator and robot and
- intuitive user interfaces.

Additionally, the interaction of the robot with its surroundings has to be taken into account in terms of environment models (respectively DTs of the environment).

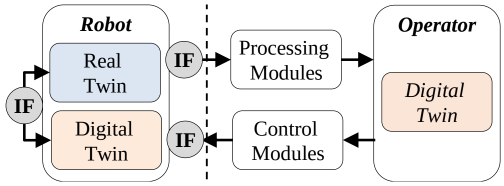
By utilizing 3D simulation technology within a VTB we already have a profound basis for the development of holistic system models especially in the field of industrial or mobile robots. Embedding the DT in these VTBs then has to fulfill all requirements mentioned above and is an improvement of the state of the art.

V. IMPLEMENTATION

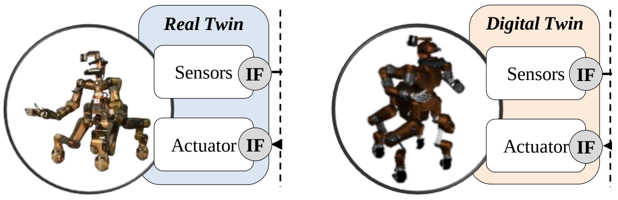
For embedding the DT in the VTB we use the simulation system VEROSIM (Virtual Environments and Robotics Simulation System) [16], which is co-developed at our institute. Prior developments have been presented for robotic teleoperation [17] and simulation-based user interfaces [18] which are the basis for these implementations.

A. Digital Twins

The DT is implemented in a modular way. Thus, it is possible to use it with reduced complexity to just visualize the current state of the system. On the other hand, adding more and more complexity to the DT leads to the required switch between real and digital robot. It then encompasses everything necessary to be "equivalent" to the real system especially with respect to Eq. (2).



(a) Directly controlling the robot, no matter if real or digital, visualized by means of a DT as well. Additionally, the required direct connection of RT and DT.



(b) Direct control of the RT w.r.t.(a) (c) Direct control of the DT w.r.t.(a)

Fig. 4: Operating a mobile robot: Monitor, control and visualize the robot real or digital

1) Robot/ Robot Components: The three main components in a (mobile) robotic system are: sensors, actors, and interfaces (see Fig. 4). Thus, the DT has to mimic these components in a holistic way. For sensors we use the integrated sensor framework of the VTB which can be adopted towards new types of sensors very easily. For actors we use the integrated rigid body framework of the VTB which can be extended towards application specific requirements, like soft body grippers, water simulation, detailed contact models for wheels, etc. Interfaces have been established on both sides: the operator and the real system (cf. Fig. 4a).

2) User Interface (Human-DT): The interface from operator to DT is bidirectional: Control input and (mostly audio-visual) output. We implemented a broad range of input devices (controller-based, optical, haptic, etc.) to generate the most intuitive control experience for each operator. For visualization we implemented Virtual Reality (VR), Augmented Reality (AR), Mixed Reality (MR), and classical monoscopic (screen-based) visualization. Although one might think that an immersive experience in VR, projecting the user to the scene of operation, should be most suitable, the human habit and long usage of classical "flat" screens is often most appreciated. Additionally, we implemented these flat visualizations as virtual screens in VR (cf. Fig. 5) as well and established a stereoscopic rendered Head-Up Display (HUD).

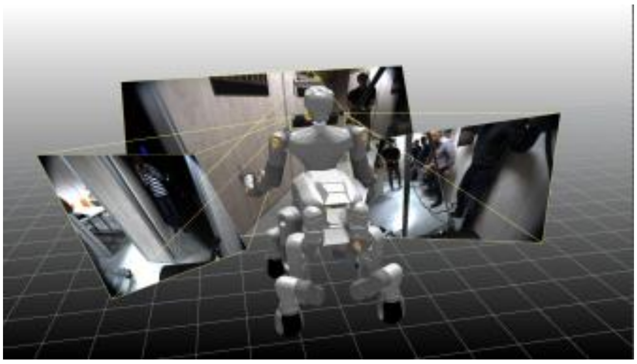


Fig. 5: User interface: Virtual screens in VR

3) Hardware Interface (DT-RT): The interface to the robot can have various possibilities. We decided to mainly support classical UDP (User Datagram Protocol) communication as well as the Robot Operating System (ROS) [19] including its model exchange format of the Unified Robot Description Format (URDF) [20]. Using this standardized, heavily used middleware it is easily possible to connect internal and external processing modules (of control, sensor data, etc.) to our DT. The DT speaks the same protocol as the RT—it e.g. provides the same ROS nodes and topics.

Combining the DT with its interfaces leads to the possibility to control the real robot through its DT, enabling the full spectrum of options in terms of input/output devices, visualizations, control schemes, and fusion of virtuality and reality.

4) Environment DTs: For the DT to be a functional representation of the real system the interaction with its environment is one central component. If one defines every object (including terrain, tasks etc.) as DTs as well, then one gets a network of interacting DTs. Thus, you either need all these object predefined (similar to the robot's DT) or you have to dynamically create the DTs. We propose to use a modular combined approach. On the one hand, we use predefined environments and objects (like a driller, knife, valves, switches, etc.). If the scenario is known prior to the mission this is the easiest and fastest way to generate all DTs necessary. On the other hand, we use dynamically created objects. These can either be meshes, gathered by the sensors and combined to different objects by using meshing algorithms (like splatting-based surface rendering techniques as presented in [21]), or we use template-based model insertion. In this case we use available deep learning methods for pose estimation (yielding a frame $F(p, O)$) and object detection (yielding a name string s). Using this tupel $t(F(p, O), s)$ one can morph predefined objects (matched via s) into the detected size of the object and insert it at F.

Whereas the meshing is especially used for generating terrain, the model insertion is mainly used on object level.

5) Interaction: Interaction of all these DTs can happen either in reality or in virtuality. Nevertheless, one can also use the twins either as a data sink or data source. Different approaches are shown in Fig. 6 where we hybridly mix and match real and virtual data. Still, robotic interaction will only happen either in (a) and (b) but the aforementioned online creation of DTs can bridge this gap by instantiating perceived objects as DTs and thus making them part of the virtual world.

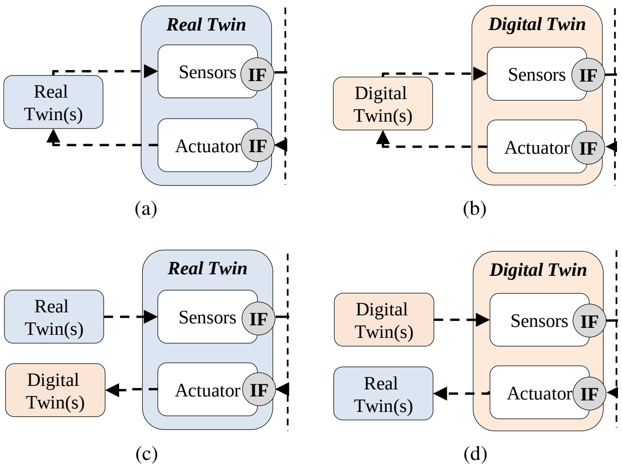


Fig. 6: Interacting DTs (w.r.t. Fig. 4): Interaction can happen either in reality (a) or in virtuality (b). Mixed approaches can use real sensor data to actuate the virtual robot (c) or virtual sensor data to actuate the real robot (d).

B. Data Representations / Views

As the human is mainly based on his visual sense this is also represented by different visualization possibilities of the DT, physical parameters, or even scenario variants.

1) Representations of the DT: For now, we presented the DT mainly as a 3D representation of the real robot (cf. Fig. 7(a)) which is only one possible view of a DT. Besides this visual 3D representation, the VTB offers additional structural view in a hierarchical tree (cf. Fig. 7(b)) describing components and functions of the system at hand. Even more important is the system-theoretical view on all in- and outputs of the DT's components. Here, an input/output board (cf. Fig. 7(c)) in the VTB shows the functional flow of data within the internal components and the connection to external frameworks like ROS. Additional tooling in the ROS context can then also be used externally.

Fig. 7: Representations of the DT: (a) three dimensional view, (b) hierarchical tree of system components (equivalent to an extensible markup language (XML) file), (c) functional flow of data

2) Parameter visualization: For an intuitive access of internal system parameters it is essential to visualize required data in an intuitive way to the user. HUD visualizations can be modularly combined and connected to system outputs to generate a customized user interface. Exemplary, these can be used to visualize critical system parameters like joint motor heat, connection signal strength, or battery life directly in the view of the operator. Besides these "flat" visualization one can also project this information into the 3D scene. Virtual screens can be rendered at the site of occurrence giving the data the additional information of space.

3) Option visualization: Due to the connection to the 3D simulation backbone one can also utilize this link to evaluate different possibilities in simulation first before executing one of them in reality. The DT can try and show possible options and outcomes to the user who can decide which to choose.

Thus, visualizing all the aforementioned data leads to an easy way of generating customized user interfaces to control and visualize the robot's state.

C. Operational Modes

In terms of human-robot cooperation one can either use the DT to visualize the current robot's state or directly control the robotic system. Here, the direct control of the RT and the DT is identical due to the use of the same interfaces and input devices. Thus, the level of detail varies between variants of the DT (see Fig. 4). For visualization the DT just reflects the current state of the system and has no inherent intelligence (in terms of special data processing capabilities), an optimal user experience is the main intention. The DT replacing the RT uses the rigid body simulation and sensor simulation to mimic the behavior of the real system producing output data for the processing modules and accepting input control from the operator's control modules (according to Eq. (2)). Of course, these modes can be used in parallel, such as actions can be evaluated controlling the DT first, before using the RT.

VI. THE SYSTEM AT WORK

The system is used in the CENTAURO project [22] to develop and operate a Centaur-like robotic system teleoperated in highly unstructured and unknown environments to fulfill a given set of tasks necessary in catastrophic scenarios. These scenarios consist of the robot's DT, the environments' DTs (like ground, buildings etc.) and the objects' DTs (like rubbish, valves, drills etc.). Combining all these interacting DTs into one scenario allows all aforementioned assistance for the operator during the missing. Furthermore, it allows to generate benchmark scenarios for testing the robot prior to its mission.

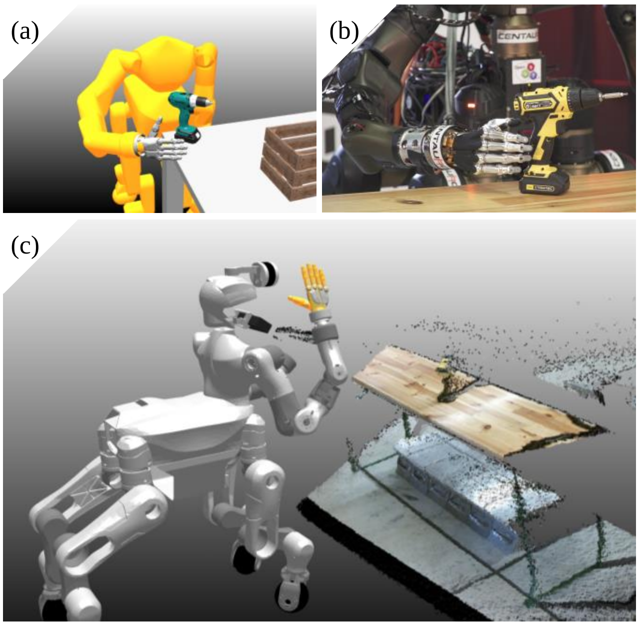


Fig. 8: Interacting with the Real or Digital Twin, exemplary shown on "grasping a drilling tool": (a) DT-DT, (b) RT-RT, (c) DT-RT

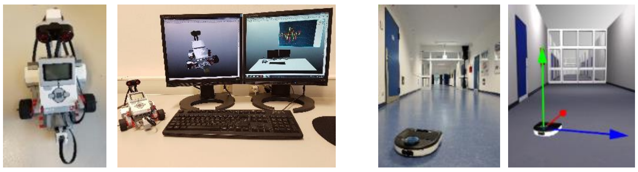
The modularity of virtual and real objects and modules (cf. Fig. 1) lead to DT-driven development of the holistic system. Each task can then be handled in four different interaction modes, exemplary described by "grasping a drilling tool" (cf. Fig. 8):

(a) Based on virtual sensor data the virtual actuator is grasping a virtual tool,
(b) Based on real sensor data the real arm is grasping a real tool.
(c) Based on real sensor data a virtual drilling tool model is instantiated and can be seen equivalent to (a),

Besides (c) there are of course also mixed options of using the data source from either the RT or DT as described in Section V-A.5 which were not used in this example. In the end, a successful virtual execution based on (c) could be directly executed/ mirrored onto the RT of course.

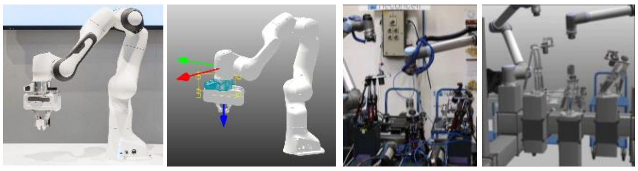
VII. APPLICATIONS

Applications for using the Digital Twin as a central compartment of the system are numerous (see also Fig. 9). First thing that comes to mind is the classical component-wise *development* of systems where most of the overall system is not available yet. Here, the DT can be used as a substitute which evolves with the real system in parallel. Additionally, interfaces allow the *monitoring* of systems online through the eyes of the DT. These two aspects can be seen as the basis for any developments where the DT is used *online*.



(a) Ground robots: *LEGO Mindstorms EV3* and *Neato Botvac D85*

(b) Unmanned aerial (UAV) and underwater vehicles/ robots (UUV)



(c) Industrial Robots: *FRANKA Emika Panda* [23] and *Universal Robots UR10* [24] (used in the project ReconCell [25])

Fig. 9: Applications besides the main developments in the Centauro system (cf. Fig. 4): Digital Twins can be used for a variety of applications for mobile ground, aerial and underwater robots, as well as industrial robots.

Due to the presented framework, not only the *fusion* of human and machine skills but also virtual and real data sinks or sources become possible to assist and support the human in such cooperative development, integration, and work. Thus, one can combine real and virtual data sources to e.g. navigate a real robot through an empty room filled with virtual obstacles recognized via virtual sensor data. On the other hand, we can generate virtual worlds based on the robot's percepts to finally evaluate different choices by using the DT before executing an operation in reality.

In contrast to single domain developments, like e.g. presented for unmanned aerial vehicles (UAVs) [26], our approach is more generic. Using VTBs and ROS as middleware, our framework can be used with any ROS-based robot (or system in general) no matter if it is on ground, air, or underwater. Also, industrial or automotive applications are possible. Besides this, it is also possible to use the power of scale shown in Fig. 9a. Developing and testing algorithms in the lab using miniaturized (for example LEGO based) robot mock-ups can be easily transferred to real systems (like cars in unknown environments). Based on this framework and the DTs developed for ground robots, everything can be transferred into other mobile robotic systems (on the ground, in the air, or even underwater), or also into the context of industrial robotics and close safety relevant human-robot collaboration (Fig. 9b and Fig. 9c respectively).

The DT then supports the user in his *perception* (via diverse sensor data processing modules, deep learning, etc.), in his cognitive decisions (by visual, audio and haptic feedback being "present" at the mission site), and in his motoric processing (getting access to dangerous environments via teleoperation or scaling movements for high precision tasks). Exemplary, for Centauro we use exoskeletal teleoperational control with force feedback [11], external modules for stepping [27] and sensor data processing in terms of object detection [28] or terrain classification [29].

VIII. CONCLUSIONS AND FUTURE WORKS

Putting the human in control of the Digital Twin leads to numerous prospects in term of an efficient support. As the DT is more than just CAD data, interfaces and integrated functionalities have been established to put the DT in between the human and the real system. Assisting in perception, cognition, and action in various applications has been presented. Especially the mobile Centauro robot with its inherent complexity and the dynamic unknown environment it is operating in shows the possibilities the DT offers for visualization on the one hand, and the direct control of the DT on the other hand. Thus, allowing development, monitoring, and control on a system level. The system at work can then use the fusion of virtuality and reality to generate the best suited man-machine interaction for the given scenario. Overall, this leads to an effective coupling of skills from the human and the robot to cooperatively execute a given task/mission.

As an outlook, one could include more interface options for the DT, like ROS2 for mobile systems or OPC Unified Architecture (OPC UA) [30] for industrial applications, and continuously integrate specialized simulation frameworks, like FEM simulation for structure mechanics, into the VTB for specific application. Due to the modularity of the presented framework and the embedding of the DT in the VTB it is possible to do so without huge efforts.

IX. ACKNOWLEDGMENTS

This project has received funding from the European Unions Horizon 2020 research and innovation program under grant agreement No 644839.

REFERENCES

[1] L. Atorf, M. Schluse, and J. Rossmann, "Simulation-based optimization, reasoning, and control: The eRobotics approach towards intelligent robots," in *Int. Symp. on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS)*, 06 2014.

[2] M. Rast, "Domänenübergreifende Modellierung und Simulation als Grundlage für Virtuelle Testbeds," Ph.D. dissertation, Aachen, 2014.

[3] R. Weidner, T. Redlich, and J. P. Wulfsberg, *Technische Unterstützungssysteme*. Springer-Verlag, 2015.

[4] S. K. Card, *The psychology of human-computer interaction*. CRC Press, 2017.

[5] R. Weidner and F. Steinicke, "Human enhancement technology," *Technische Unterstützungssysteme, die die Menschen wirklich wollen*, p. 365, 2016.

[6] M. P. de Looze, T. Bosch, F. Krause, K. S. Stadler, and L. W. OSullivan, "Exoskeletons for industrial application and their potential effects on physical work load," *Ergonomics*, vol. 59, no. 5, pp. 671–681, 2016.

[7] H. Lee, W. Kim, J. Han, and C. Han, "The technical trend of the exoskeleton robot system for human power assistance," *International Journal of Precision Engineering and Manufacturing*, vol. 13, no. 8, pp. 1491–1497, 2012.

[8] H. Kazerooni, "Exoskeletons for human power augmentation," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 3459–3464.

[9] K. Kiguchi, M. H. Rahman, M. Sasaki, and K. Teramoto, "Development of a 3dof mobile exoskeleton robot for human upper-limb motion assist," *Robotics and Autonomous systems*, vol. 56, no. 8, pp. 678–691, 2008.

[10] R. Weidner, N. Kong, and J. P. Wulfsberg, "Human hybrid robot: a new concept for supporting manual assembly tasks," *Production Engineering*, vol. 7, no. 6, pp. 675–684, 2013.

[11] T. Cichon, C. Loconsole, D. Buongiorno, M. Solazzi, C. Schlette, A. Frisoli, and J. Roßmann, "Combining an exoskeleton with 3d simulation in-the-loop," in *9th International Workshop on Human Friendly Robotics*, 2016, pp. 31–34.

[12] D. Stokes and S. Biggs, "The dominance of the visual," *Perception and its Modalities*, 2014.

[13] T. Cichon, C. Schlette, and J. Roßmann, "Towards a 3d simulation-based operator interface for teleoperated robots in disaster scenarios," in *Safety, Security, and Rescue Robotics (SSRR), 2016 IEEE International Symposium on*. IEEE, 2016, pp. 264–269.

[14] M. Schwarz, T. Rodehutskors, D. Droeschel, M. Beul, M. Schreiber, N. Araslanov, I. Ivanov, C. Lenz, J. Razlaw, S. Schüller *et al.*, "Nimbro rescue: Solving disaster-response tasks with the mobile manipulation robot momaro," *Journal of Field Robotics*, vol. 34, no. 2, pp. 400–425, 2017.

[15] M. Schwarz, J. Stückler, and S. Behnke, "Mobile teleoperation interfaces with adjustable autonomy for personal service robots," in *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*. ACM, 2014, pp. 288–289.

[16] J. Rossmann, M. Schluse, C. Schlette, R. Waspe, J. Van Impe, and F. Logist, "A new approach to 3d simulation technology as enabling technology for erobotics," in *1st International Simulation Tools Conference & EXPO*, 2013, pp. 39–46.

[17] T. Cichon and J. Rossmann, "Simulation-based user interfaces for digital twins: Pre-, in-, or post-operational analysis and exploration of virtual testbeds," in *Goncalves, P. J. S. (Ed.): The 31st European Simulation and Modelling Conference (ESM) - Modelling and Simulation, October 25 27, 2017, Lisbon, Portugal, pp. 384-389, ISBN 978-492859-00-6*. EUROSIS-ETI, Ghent, Belgium, 2017.

[18] ——, "Robotic teleoperation: mediated and supported by virtual testbeds," in *The 15th IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR), Shanghai, China, October 11-13, 2017, pp. 55-60, ISBN (USB): 978-1-5386-3922-1*, 2017.

[19] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[20] L. Kunze, T. Roehm, and M. Beetz, “Towards semantic robot description languages,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5589–5595.

[21] G. Guennebaud, L. Barthe, and M. Paulin, “Real-time point cloud refinement.” in *SPBG*, 2004, pp. 41–48.

[22] CENTAURO. (2014) Project website of Horizon2020 project CENTAURO. (accessed 2018-05-23). [Online]. Available: https://www.centauro-project.eu/

[23] FRANKA Emika. (2018) Product website of the Panda robot. (accessed 2018-05-23). [Online]. Available: https://www.franka.de/panda

[24] Universal Robots. (2018) Product website of the UR10 robot. (accessed 2018-05-23). [Online]. Available: https://www.universal-robots.com/products/ur10-robot/

[25] ReconCell. (2014) Project website of Horizon2020 project ReconCell. (accessed 2018-05-23). [Online]. Available: http://www.reconcell.eu/

[26] T. Sayre-McCord, W. Guerra, A. Antonini, J. Arneberg, A. Brown, G. Cavalheiro, Y. Fang, A. Gorodetsky, D. McCoy, S. Quilter, F. Riether, E. Tal, Y. Terzioglu, L. Carlone, and S. Karaman, “Visual-inertial navigation algorithm development using photorealistic camera simulation in the loop,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[27] T. Klamt and S. Behnke, “Anytime hybrid driving-stepping locomotion planning,” in *Int. Conference on Intelligent Robots and Systems (IROS)*, 2017.

[28] M. Schwarz, A. Milan, A. S. Periyasamy, and S. Behnke, “Rgb-d object detection and semantic segmentation for autonomous manipulation in clutter,” *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 437–451, 2018.

[29] F. J. Lawin, M. Danelljan, P. Tosteberg, G. Bhat, F. S. Khan, and M. Felsberg, “Deep projective 3d semantic segmentation,” in *International Conference on Computer Analysis of Images and Patterns*. Springer, 2017, pp. 95–107.

[30] S.-H. Leitner and W. Mahnke, “Opc ua–service-oriented architecture for industrial applications,” *ABB Corporate Research Center*, 2006.