ARTiSt — An Augmented Reality Testbed for Intelligent Technical Systems

Bassem Hassan[1], Jörg Stöcklein[2](✉), and Jan Berssenbrügge[2]

[1] Fraunhofer Institute for Production Technology IPT, Aachen, Germany
Bassem.Hassan@ipt.fraunhofer.de

[2] Heinz Nixdorf Institute at University of Paderborn, Paderborn, Germany
{Joerg.Stoecklein,Jan.Berssenbruegge}@hni.upb.de

Abstract. This paper describes a simulation and visualization environment called **ARTiSt** (**A**ugmented **R**eality **T**estbed for **i**ntelligent technical **S**ys**t**ems), which serves as a tool for developing extension modules for the miniature robot BeBot. It allows developers to simulate, visualize, analyze, and optimize new simulated components with existing, real system components. In ARTiSt real BeBots combined with virtual prototypes of a lifter- and a transporter-module, which are attached on top of the real BeBot. The simulation of the virtual components and the management of real BeBots are realized with MATLAB/Simulink. The determination of important parameters for the simulation of the real BeBots, such as real-world position and -rotation, is done using an Augmented Reality tracking system. A camera, installed on top of the testbed, continuously captures the testbed and determines the real-world transformation of the BeBots. The calculated transformations are the basis for further pathfinding within the simulation in MATLAB/Simulink.

1 Introduction

Virtual Prototyping is a technique, which applies *Virtual Reality*-based product development for the engineering of mechanical and mechatronic systems. Virtual prototyping is based on the modeling, design, and analysis of *Virtual Prototypes*, e.g. computer-internal models, which are executed and analyzed in a Virtual Environment.

Virtual Prototypes are typically developed prior to physical prototypes (or mock-ups), which are mainly profitable for relatively small subsystems. Compared to physical prototypes, the development of Virtual Prototypes is less expensive and time-consuming, and Virtual Prototypes provide a significantly higher flexibility for change requests and variant management. Moreover, due to the virtualization of the prototype and the environment, Virtual Prototypes facilitate the early evaluation of the final product. All experiments can be conducted under controlled conditions of a well structured *Virtual Test Bench* and, for instance, can easily be repeated for regression testing.

R. Shumaker and S. Lackey (Eds.): VAMR 2015, LNCS 9179, pp. 457–469, 2015.
DOI: 10.1007/978-3-319-21067-4_47

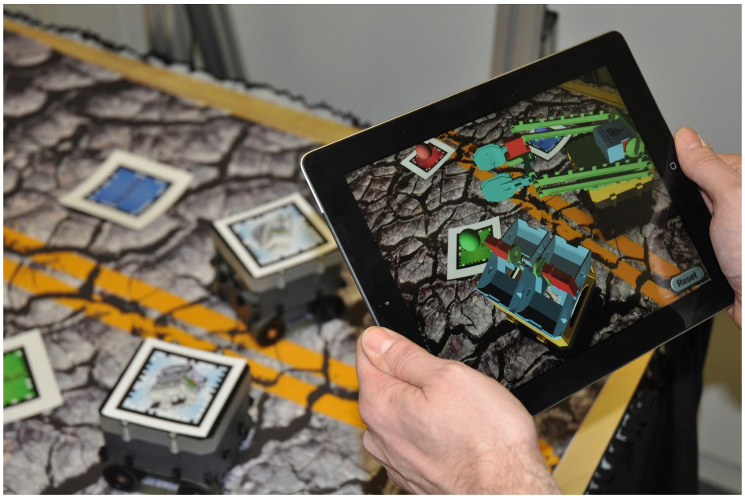

Fig. 1. Visualization of virtual modules on real BeBots using augmented reality.

Rapid prototyping in the product development process, uses Virtual Prototypes in conjunction with direct manufacturing methods, e.g. 3D printing, is successfully applicable nowadays. Products can thereby be virtually designed and tested and are manufactured to successful optimization and validation in the direct manufacturing process. Thus, a product component by component developed and virtual and real components are evaluated simultaneously with the help of *Augmented Reality*.

In our Project *ARTiSt* physical miniature robots BeBots work together with virtual prototypes of a lifter- and a transporter-module, which are attached on top of the real BeBot, in real-time. In order to evaluate both, the real miniature robots and the virtual extension modules, we use Augmented Reality to correctly combine both prototypes together. A simulation is used for calculating the optimal motion paths of the BeBots and the controlling commands for lifting and transporting the objects.

The visualization of the virtual advanced modules (lifter, transporter) attached to the top of the real BeBots is realized by means of an Augmented Reality application for mobile devices (see Fig. 1). This allows an intuitive visual analysis of the interaction of all the components using the magic lens metaphor. The Augmented Reality application is connected via wireless network to the simulation engine developed in MATLAB/Simulink. That enables the direct visualization of the calculated parameters of the simulation in the virtual model.

Objective of the simulation is to demonstrate cooperation of two BeBots, were one BeBot acts as a transporter, carrying collected objetcts to its destination, and cooperate with a lifter, collecting objects and puts them on the transporter. Both robots work efficiently in collecting and transporting virtual objects which were distributed interactively on the testbed. Therefore, the lifter and the transporter need to coordinate themselves autonomously. This process can be analyzed using the Augmented Reality application.

After the validation of the simulation, the optimized structure of advanced modules can be produced in the direct manufacturing process. This approach reduces the number of real samples and prototypes considerably.

2 Essentials

This paper combines technologies from the domains of *Virtual Reality*, *Virtual Prototyping*, and *Augmented Reality* with methods developed for mechatronic system. In the following paragraphs, the basic principles of these technologies as well as the test platform used will be described in short.

2.1 Virtual Reality, Augmented Reality and Virtual Prototype

Virtual Reality (VR) provides methods for the analysis and evaluation of test results. Using VR during the product development process is not unusual. For example, VR is applied in the field of mechanical engineering and plant engineering in order to plan and evaluate technical systems [11].

Augmented Reality (AR), in comparison to VR, is still a novel technology. AR is a human-computer-interface, which superimposes the perception of reality with computer-generated information. Today, most AR applications can be found in niches. Cutting-edge fields are automotive development and marketing. For instance, automotive development uses AR for the evaluation of new automotive prototypes and for the preparation of experiments [3,10]. In the marketing sector AR often used for product presentation. Recently, a manufacturer of toy building blocks tested AR, in order to present the final product on top of the package [5].

A *Virtual Prototype* (VP) is the computer-internal representation of a real prototype of a product [4]. The VP is based on the *Digital Mock-Up* (DMU). The DMU represents the shape and the structure of the product. The foundations of the DMU are composed of two types of models: 3D-CAD for shape and the logical structure of the product. The VP extends the DMU because further aspects are taken into account which are aspects like the kinematics, dynamics, strength, or information processing. A computer-internal model represents each of these aspects.

A *Virtual Environment* (VE) is a synthetic, computer generated environment, which presents a visual, haptic, and auditive simulation of a real world [1]. A VE for developing, analyzing, improving, and testing VPs is composed of one or multiple VPs integrated into the VE. A detailed explanation can be found in [8].

2.2 Specification of the Miniature Robot BeBot

The miniature robot *BeBot* (see Fig. 2) has a size of approximately 9 by 9 cm and a height of about 5 cm. Its chassis uses **MID** (**M**olded **I**nterconnect **D**evices)

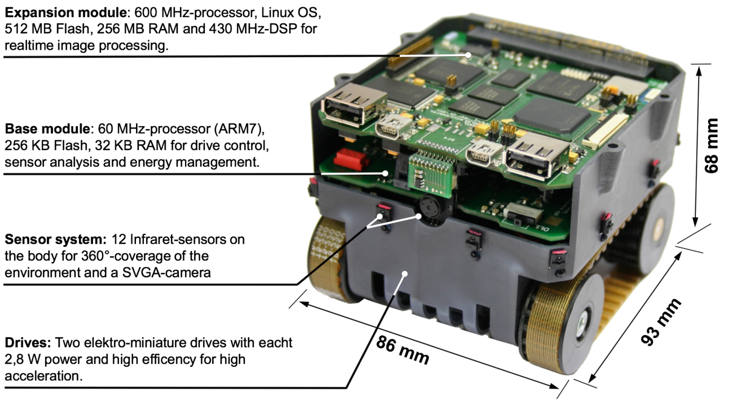
Expansion module: 600 MHz-processor, Linux OS, 512 MB Flash, 256 MB RAM and 430 MHz-DSP for realtime image processing.
Base module: 60 MHz-processor (ARM7), 256 KB Flash, 32 KB RAM for drive control, sensor analysis and energy management.
Sensor system: 12 Infraret-sensors on the body for 360°-coverage of the environment and a SVGA-camera
Drives: Two elektro-miniature drives with eacht 2,8 W power and high efficency for high acceleration.
68 mm
86 mm
93 mm

Fig. 2. BeBot with hardware components.

technology, where printed circuit board tracks directly applied on the body surface. MID offers new possibilities for the synergistic integration of mechanics and electronics [2]. This technology is used for mounting 12 infrared sensors, a micro controller, several transistors, and resistors for preprocessing directly on the robot chassis. The drive of the robot consists of a chain drive. Together with two 2.8 W DC gear motors with built-in encoders the robot offers robust motion even on slightly rough ground. The complete system is supplied by a 3.7 V/3900 mAh lithium-ion accumulator. The BeBot uses a modular concept of information processing with two board slots. The lower board implements basic functions like motor control and power supply. An *ARM 7* based micro controller provides enough computational power to implement low-level behavior. The module also contains a three axis acceleration sensor, a yaw rate gyroscope, and a sensor for battery monitoring. The upper slot provides more powerful information processing and wireless communication. It is equipped with a low power system-on-chip (SoC) with package-on-package memory and provides an *ARM Cortex-A8 600* MHz high performance processor, a TIC64x+ digital signal processor, 256 MB main and 512 MB flash memory. This allows the computation of complex algorithms and image processing direct-on the system. The integrated wireless communication standards Bluetooth and IEEE 802.11 wireless network offer communication with various bandwidth and power consumption. The board is equipped with a small camera and two microphones for a substantial perception of the environment. Different techniques for energy saving like dynamic frequency and voltage scaling as well as dynamic power-down of non-used hardware components including RF processing, combining powerful computation capability with long battery life time. Additional I2C, UART, USB, MMC/SDcard, camera and memory interfaces as well as a small module slot provides great expansion capabilities. An infrared communication interface allows the cordless equipment with mechanical extension modules. The software environment uses *OpenRobotix*, which is based on OpenEmbedded and allows the generation of a fully customize Linux operating system. It generates cross-compiled software packages and images for the embedded target. The existing software branch was extended to contain the robot-specific information, patches, and additional software like drivers for the robot hardware and the Player network server.

3 BeBot Application Scenario

The miniature robot BeBot was developed as a demonstration platform to evaluate different advanced algorithms and technologies such as: swarm intelligent algorithms, dynamic reconfiguration, multi agent systems, molded interconnected devices as well as self-optimization functionalities. In order to demonstrate, investigate, and evaluate these algorithms and technologies with the help of the miniature robot BeBot, application scenarios have to be designed and developed.

For ARTiSt we have developed an application scenario which focuses on two different parts in the development process: The product development of external modules attached to the BeBot in order to fulfill a specific task on the one hand. On the other hand the algorithmic development of strategies which enable two or more BeBots to work together on a global task but with different subtasks for each BeBot.

The ARTiSt application scenario is shown in Fig. 3. The objective is to collect three different-colored objects within the testbed and place them in their corresponding drop areas. The objects are randomly placed within the testbed at the beginning by the user. Then a group of BeBots (in our case two) should cooperatively collect these objects. Each BeBot has one of two pre-defined roles, though: one can act like a transporter, which collects all objects and transports

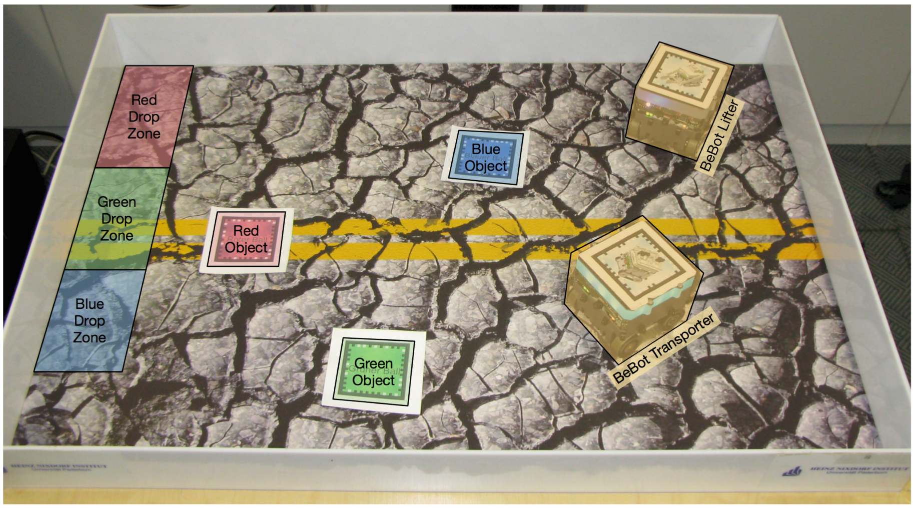

Fig. 3. Application scenario.

them to the drop zones. The other BeBot works as a lifter, which picks each objects and puts it into a free storage of the transporter. Both BeBots are working independently but in cooperation with the other group members.

3.1 Development of BeBots Extension Modules

In order to realize the two pre-defined roles of the BeBot (transporter and lifter) two extension modules have to be developed and attached with the existing main vehicle. The development of these extension modules has been done with the help of virtual prototyping and simulation.

Today, virtual prototyping is an integral part of the product development process. In virtual prototyping, a computer model of the product under development is generated and subsequently tested, just like a real prototype. This way, design errors can already be detected in the early phases of the product development process and alternative designs of a product can be virtually examined, without having to build a real prototype. This reduces time and costs in product development and raises product quality.

Each of the two extension modules has particular requirements according to the specific task in the application scenario. The requirements and are described as following:

Requirements of the Lifter Extension Module: The lifter extension module should be able to pick up and lift different object types from the ground and load them into the transporter extension module. Therefore, it should have a gripping unit to do the picking. The gripping unit should move in a vertical direction in order to lift the object from the ground and put it into the transporter extension module. Additionally, for sensing the dimensions of the picked object, the gripping unit should include an infrared sensor placed between the two gripping arms. This also ensures that the object is located correctly between the gripping arms. Moreover, the gripping unit should be equipped with a pressure sensor to measure the gripping power applied on the object. According to the modular concept of the BeBot, the lifter module should be controlled by a separate circuit board and should have its own power supply.

Requirements of the Transporter Extension Module: The transporter module should be able to carry and transport at least four different objects at once, each object on a separated cavity. The load operation should be done with the help of the lifter module, but the transporter module can independently transport and unload the objects in their drop areas. The unload process could be done for each object individually. According to the modular concept of the BeBot, the transporter module should be controlled by a separate circuit board and should have its own power supply.

3.2 Virtual Prototyping of the Extension Modules

In order to develop the extension modules for the BeBots virtual prototyping and simulation techniques have been used. Both extension modules are *mechatronic systems* and therefore consist of the three main components:

Mechanical Component: Each extension module includes mechanical parts, e.g. the lifting unit or the unload mechanic.

Electronic Components: Each extension module includes sensors and actuators, e.g. a infrared sensor in the lifter or a servo motor in the transporter, and a control circuit board.

Software Components: For each extension module a customized control software is developed to operate all functions.

To develop each of the three main components different approaches are used, which will be described below:

Mechanical Construction of the Extension Modules: Based on each extension module requirements, its mechanical construction has to be developed. Nevertheless, there are numerous designs of each extension module fulfilling the predefined requirements. Therefore, the innovation funnel model has been used in order to select the best mechanical construction of the extension modules.

The best mechanical construction selection process is consisting of the following four phases:

1. **Brainstorming:** In this phase, the involved design group has to generate many possible concepts of the extension modules' mechanical construction.
2. **Documentation:** In this phase, the generated mechanical constructions in the brainstorming phase have to be documented by means of their mechanisms and components.
3. **Evaluation:** In this phase, the documented mechanical constructions have to be evaluated with the help of the following criteria: cost, weight, needed space, dimensions, efficiency, and simplicity of control.
4. **Selection:** In this phase, a mechanical construction has to be selected based on the evaluation of the different mechanical constructions.

A 3D model of the selected mechanical construction of each extension module has been developed with help of a CAD tool. In this work, we used CATIA to construct the 3D models. extension module.

Additionally, a physical model of each module has to be modeled. The model of the extension module represents the mechanical construction as a multi body system, sensors, actuators and the control algorithm. In this work we, used Matlab/Simulink to build up the physical models of the extension modules.

Controlling Circuit Boards for the Extension Modules: The development of the two controlling circuit boards for the extension modules was done in three phases.

The first phase was the schematic design. In this phase, the electronic components have been selected and connected to each others. Moreover, the interaction between the different electronics components have been simulated and tested with the help of a schematic editor tool. In this work, we used the tool EAGLE for the development of the electronic circuit board.

The second phase was the layout design. In this phase, layouts of the circuit boards have been carried out. The layout design has been done by positing the electronic components within the predefined dimensions of the circuit board (5 by 7 cm).

The third phase was the conducting path design. In this phase, conducting paths of each copper connection as well as its thickness have to be created with the different layers of the circuit. Our circuit boards have a compact size. Therefore, each circuit board has 7 connection layers. Nowadays, electronic circuits design tools e.g. Eagle can create the routing plane automatically within the different layers of the electronic circuit board.

Controlling Software for the Extension Modules: In order to control the extension modules, a control algorithm has to be programmed for each module. The control algorithm should collect signals from the different sensors and calculates the control signal for each actuator. The control algorithm of each extension module will be executed with the help of a micro controller on the control circuit.

The controlling software program is written in ANSI-C programming language. It has the advantage testing the control algorithm virtually by embedding the code in the Matlab/Simulink environment by using s-functions.

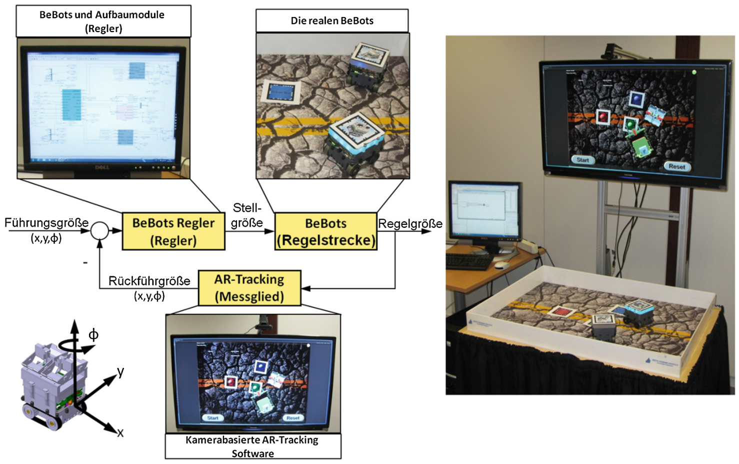

Fig. 4. Structure of the augmented reality testbed.

4 The Augmented Reality Testbed

In this section, we describe the structure of our augmented reality testbed ARTiSt by means of the specific scenario described in Sect. 3. However, ARTiSt can also be used in other scenarios, e.g. evaluating other extension modules for the BeBot or examine cooperative and competitive team strategies for other tasks. However, to be able to use ARTiSt in other scenarios, the structure of the testbed and the use of different components to set up an own scenario has to be understood.

To enable the virtual extension modules to interact with the physical BeBots we used the AR technology (see Fig. 4). AR is used for registering virtual objects with real-world objects, in this case the virtual extension modules with the real BeBots. For the user, who looks through an AR device, the virtual objects seem to exist in the real world. For an operable simulation, the computer must know the exact position and orientation of the real world and all its objects.

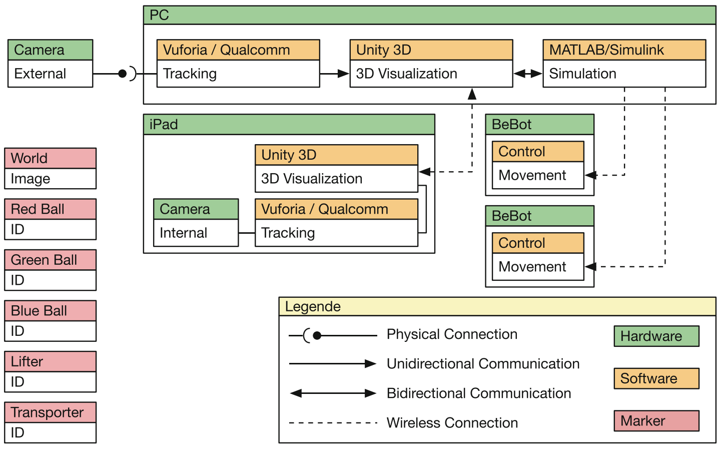

Fig. 5. Hardware and software structure of the ARTiSt-system.

Figure 5 illustrates the hardware and software structure of the ARTiSt system. A standard PC with an attached video camera serves as main component. With the help of the video system, all different markers are detected using a third-party tracking system. We use *Qualcomm Vuforia* [7] as an AR tracking system because of the easy usage and the possibility to track large images which can be partially occluded. The system has to track the following objects:

The BeBots: Two BeBots have to be tracked by the system to ensure that the virtual extension modules are properly registered in the visualization and to send the world coordinates to the simulation.

The Collectable Objects: The Marker representing the virtual objects, which will be picked up, must be tracked by the system. Each of the marker can be placed across the testbed by the user randomly before starting the simulation.

The World: The System has to know the origin of the world. This is vital for the calculation of the position of both BeBots and the user-placed objects. All objects and BeBots must be in world-space for proper simulation.

For the BeBots and for the objects, we used so-called ID-markers, as the tracking software is able to detect them even under bad light conditions. For tracking the origin of the world, we used an image marker, since ID-markers have the disadvantage that the tracker looses the detection, if parts of the marker are covered. Image markers can be partial occluded but will be still detected by the tracking system. Using an image marker, which covers the entire floor of the testbed, was sufficient to be able to detect the origin of the world in any situation.

The tracking system Vuforia was integrated as plugin in the 3D visualization system *Unity3D* [9]. Therefore, the position and orientation of the tracked markers is directly available as a transformation matrix inside the 3d visualization software. With the help of the supplied transformation matrices, the virtual objects can be registered in the correct position within the captured video frame. Overlaying the virtual 3D model on top of the captured video frame produces a correct 3D impression. As the simulation of the BeBots was realized with MATLAB/Simulink, the position and orientation had to be provided to it. This was done by a local UDP connection between Unity3D and MATLAB/Simulink, where Unity3D sends each tracked position. In MATLAB/Simulink the motion paths of each BeBot are calculated, according to their actual position and orientation, which results in the rotation speed of each individual DC motor of the BeBots. The rotation speeds are then sent as commands to the BeBots using a wireless lan connection. When approaching a virtual object, additional commands for lifting the objects are send to the transporter BeBot. But, as the lifter only exists as a virtual prototype, these commands are not executed on the real BeBot but in the 3D visualization system, which simulates the lifting process of objects. Therefore, the 3D visualization was able to control the lifter, grab the virtual objects, and drop it into an empty slot of the virtual transporter extension module.

The virtual objects can only be seen on a monitor attached to the testbed, as illustrated in Fig. 4, which restrict the viewing angle for the user. Therefore, we developed a 3D visualization for a mobile device, which is able to augment the real BeBots and the virtual objects as the 3D visualization running on the PC but can be used as a mobile 'window to the world' [6]. In fact, as we used Unity3D as the implementation platform, we can reuse the same project, we developed for the PC, also for the mobile device. This cuts the development of both version to the half. The mobile device uses its internal camera module to capture the real world environment and detects all markers inside. On the detected marker, the virtual 3D objects can be registered. As on the PC version of the visualization we use Vuforia as AR tracker. Nowadays, mobile devices are powerful enough to do the video capturing, the AR tracking, and the 3D visualization of the virtual 3D objects in real-time. Therefore, the mobile version of the visualization runs stand-alone. To enable a synchronous animation of the mobile application and the PC software, we synchronized both applications using a wireless network connection. Unity3D offers a suitable solution for this problem. Thereby, the commands sent by the MATLAB/Simulink simulation are only received by the

PC version of the visualization. The animations of the 3D objects are then synchronized, using the network synchronization option of Unity3D. We do not need to connect the mobile application to the MATLAB/Simulink simulation, thus saving network bandwidth, simulation execution time, and implementation effort.

All relationships between the different software applications we use in ARTiSt are illustrated in Fig. 5. Figure 5 illustrates how the PC and the mobile device is communicating, which channels for communication are used and what kind of markers are used for the different AR objects. As the MATLAB/Simulink simulation was part of a former project, we were able to build the 3D visualization and AR part of the testbed for both platforms in less then 2 weeks. This rapid development can only be achieved because of the use of a full-featured 3D visualization software platform like Unity3D.

5 Implementation of the Prototype Scenario

After starting the test scenario, the AR-Tracking software calculates the position (X, Y) and the orientation angle (ϕ) of each object regarding to the workspace coordinate system (origin point is on the top right corner). There are five objects marked by AR-Markers, as mentioned in section Sect. 4: both BeBots (lifter and transporter) and the three collectible objects. The AR-Tracking software send the detected position and orientation (X, Y, ϕ) of each object in tracked world coordinates to the scenario controller.

Based on the five object positions and orientations, the scenario controller calculates the object collection order depending on the pre-programmed strategy e.g. collect the objects as fast as possible or collect the objects with minimum energy consumption. The scenario controller sends the position and orientation of the next object to be collected to the lifter-BeBot controller.

Based on the current position and orientation of the lifter-BeBot and the next object position, the lifter-BeBot motion controller software calculates the shortest motion path reaching the next object. The motion path in this case contains the position (X, Y) of the target point as well as the target orientation (ϕ). The target angle is the desired lifter-BeBot orientation on the target point. The lifter-BeBot motion controller sends the path information to the lifter-Bebot collision avoidance controller.

Based on the Lifter motion path and the position and orientation of the others objects as well as the transporter-BeBot, the lifter collision avoidance has to ensure that there is no collision between the lifter-BeBot from one side and the other objects from the other side along the selected motion path. If a collision is detected, the collision avoidance controller has to define a maneuver in form of a new motion path in order to avoid the collision. The new motion path in this case contains more than one target position and orientation (typically three points) in order to achieve the collision avoidance maneuver. Therefore, the lifter-Bebot collision avoidance controller sends the new calculated motion path in the form of current point position and orientation and an array of target point positions and orientations to the motion controller. If no collision is detected, the originally calculated motion path in the form of current point position and orientation and target point position and orientation will be sent.

The lifter-BeBot motion controller is a fuzzy logic controller, which controls two parameters: the BeBot translation and the BeBot rotation by operating the two motors of the BeBot. As soon as the lifter-BeBot motion controller has reached the target point successfully, it sends a signal to the extension module controller to confirm that the target position and orientation has been reached. The lifter-BeBot extension module controller starts to operate the gripping mechanism in order to grip the object with help of the simulated infrared and gripping pressure sensors. As soon as the extension module controller successfully has gripped the object, it sends a confirmation signal to the scenario controller to confirm that the first object is in the gripper.

Based on the current position of the lifter-Bebot and the current position of the transporter-BeBot, the scenario controller calculates in which transporter-carriage the object has to be loaded and also calculates the target position and orientation for the transporter-BeBot. The transporter-BeBot controllers (motion and collision avoidance) control the motors until it reach the target position and orientation. It works with the same procedure like the lifter-BeBot controllers. As soon as the transporter-BeBot successfully reached the target point, it sends a confirmation signal to the scenario controller. The scenario controller sends a signal to the gripper to release the object into the transporter-BeBot carriage. The pervious steps are repeated until all the objects have been collected. As soon as all objects have been collected, the scenario controller calculates for the transporters the unload path in form of three target points: red drop-point, green drop-point and blue drop-point. The scenario controllers send the target position and orientation of the target drop points one by one.

The transporter-BeBot controllers (motion and collision avoidance) control the transporter-BeBot motors until they reach the drop-point position and the target orientation. As soon as the transporter-BeBot has reached the drop-point successfully, it sends a confirmation signal to the scenario controller.The scenario controller sends a signal to the transporter to unload a specific object regarding to the drop-point color. After unloading all objects, the scenario controllers end the simulation.

6 Conclusion

This paper described an augmented reality testbed for the development of intelligent technical systems. The presented approach has been implemented and applied successfully during the development of extension modules for the miniature robot BeBot. The presented approach showed the significant strength of integrated modeling, simulation, and AR techniques together in an environment, which support the development of an extension for an existing systems. This approach has been validated with the help of the presented application scenario ‘extension modules development for BeBots’ but it can be modified and used in other application scenarios, e.g. development of existing machines extensions.

References

1. Gutiérrez, M., Vexo, F., Thalmann, D.: Stepping into Virtual Reality. Springer, London (2008)
2. Kaiser, I., Kaulmann, T., Gausemeier, J., Witkowski, U.: Miniaturization of autonomous robot by the new technology molded interconnect devices (mid). In: Proceedings of the 4th International AMiRE Symposium, Buenos Aires (2007)
3. Klinker, G., Dutoit, A.H., Bauer, M., Bayer, J., Novak, V., Matzke, D.: Fata morgana a presentation system for product design. In: ISMAR 2002: Proceedings of the 1st International Symposium on Mixed and Augmented Reality. IEEE Computer Society, September 2002
4. Krause, F.-L., Jansen, H., Kind, C., Rothenburg, U.: Virtual product development as an engine for innovation. In: Krause, F.-L. (ed.) The Future of Product Development, pp. 703–713. Springer, Berlin (2007)
5. Metaio. The LEGO group to boost retail with metaio. Press release, December 2008
6. Milgram, P., Takemura, H., Utsumi, A., Kishino, F.: Augmented reality: a class of displays on the reality-virtuality continuum (1995)
7. Qualcomm Connected Experiences, Inc., Qualcomm Vuforia Developer Portal (2015). https://developer.vuforia.com
8. Radkowski, R., Waßmann, H.: Software-agent supported virtual experimental environment for virtual prototypes of mechatronic systems. In: Proceedings of the ASME 2010 World Conference on Innovative Virtual Reality WINVR2010, Ames, Iowa, USA, 12–14 May 2010
9. Unity Technologies. Unity - Game engine, tools and mulitplatform (2015). http://www.unity3d.com
10. Wittke, M.: Ar in der pkw-entwicklung bei volkswagen. In: Schenk, M. (ed.) IFF-Wissenschaftstage -Virtual Reality und Augmented Reality zum Planen, Testen und Betreiben technischer Systeme, 4. Fachtagung zu Virtual Reality, Fraunhofer IFF, Magdeburg, 27–28 June 2007
11. Ye, J., Badiyani, S., Raja, V., Schlegel, T.: Applications of virtual reality in product design evaluation. In: Jacko, J.A. (ed.) HCI 2007. LNCS, vol. 4553, pp. 1190–1199. Springer, Heidelberg (2007)