Alternate Realities for Mission Operations Plan Execution

Pete Bonasso*, Dave Kortenkamp*, Blair MacIntyre†, Bryn Wolfe*

*TRACLabs, Inc., bonasso@traclabs.com, †Georgia Tech, blair@cc.gatech.edu

Abstract

Procedures are a mechanism by which NASA crewmembers execute plans. Alternate reality systems can help replace some of the guidance that ground controllers offer to crewmembers during procedure execution. As space exploration missions take crews further away from Earth, new forms of procedure assistance will be necessary. This paper describes an early development of an alternate reality (AR) system called PRIDE-AVR. PRIDE-AVR is an integration of the PRIDE electronic procedure development and execution system with augmented, virtual and hybrid reality technologies. We describe the system architecture and three proofs of concept demonstrations that use these AR technologies.

Motivation

Standard operating procedures are the mechanism by which plans are executed during typical spacecraft operations. Execution of procedures on the International Space Station (ISS) is currently heavily dependent upon ground controllers assisting crewmembers in performing planned operations and maintenance as well as with responses to off-nominal situations. This close collaboration becomes more difficult in exploration missions that take human crews beyond the easy reach of Mission Control, so crewmembers will need to have more autonomy from ground controllers. Alternate realities – augmented, virtual or hybrid – can help replace some of the guidance that ground controllers offer to crewmembers during procedure execution (Tang et al., 2003).

The context of the current work is authoring and executing NASA procedures that are then used for plan execution. The on-board short-term plans (OSTPs) for the International Space Station (ISS) are carried out by executing pre-written procedures. We have developed a procedure authoring and executing system called PRIDE (Izygon et al., 2008) that is currently used by NASA to produce machine-readable procedures.

In support of NASA, TRACLabs and Georgia Tech's Augmented Environmental Lab are working to integrate our PRIDE procedure development system (Izygon et al., 2008) with augmented, virtual and hybrid reality technologies in a system called PRIDE Augmented and Virtual Reality (PRIDE-AVR).

PRIDE-AVR

Augmented reality is a live direct or indirect view of a physical, real-world environment whose elements are supplemented by computer-generated sensory input such as sound, video, graphics or geospatial data. Virtual Reality is a realistic and immersive simulation of a three-dimensional environment, created using interactive software and hardware, and experienced or controlled by movement of the body[1]. Hybrid reality, sometimes known as mixed reality (de Souza e Silva, 2009), is the merging of real and virtual

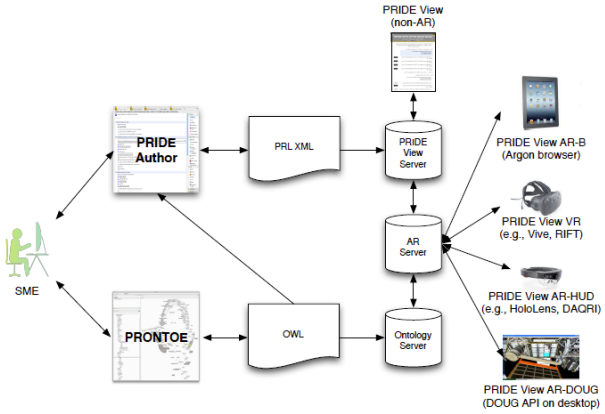

Figure 1 *The architecture of the PRIDE-AVR system.*

worlds to produce new environments and visualizations where physical and digital objects co-exist and interact in real time. With PRIDE-AVR we are investigating all three alternate realities in support of NASA procedure execution.

[1] http://www.dictionary.com/browse/virtual--reality

Our PRIDE-AVR design is shown in **Figure 1**. It builds on several existing components, including the PRIDE electronic procedure platform and a system ontology. The former includes a procedure authoring tool (PRIDE Author) and a server (PRIDE View Server) that shows procedures as web pages and supports crew member execution of procedures. The latter includes an ontology editor called PRONTOE (Bell et al., 2013) and an ontology server. Recently developed components are an alternate reality (AR) Server and new PRIDE View clients for the different alternate reality systems shown on the right of the figure.

The PRIDE viewer maintains the state of the executing procedure, that is, the current instruction and the success or failure of an instruction and/or of the procedure as a whole. As the user executes a procedure, the AR server accesses that information, tracks the progress and looks in the current instruction's PRL for references to objects in the domain ontology. The AR server queries the ontology server for any alternate reality attributes of those objects then sends the information to the AR system involved in the procedure execution.

NASA procedures can have conditional branching, instructions that are coordinated across multiple procedures, and instructions that invoke other procedures. For this preliminary work, we have used only linear procedures.

This paper describes the development of three proof-of-concept demonstrations using PRIDE-AVR in support of NASA missions.

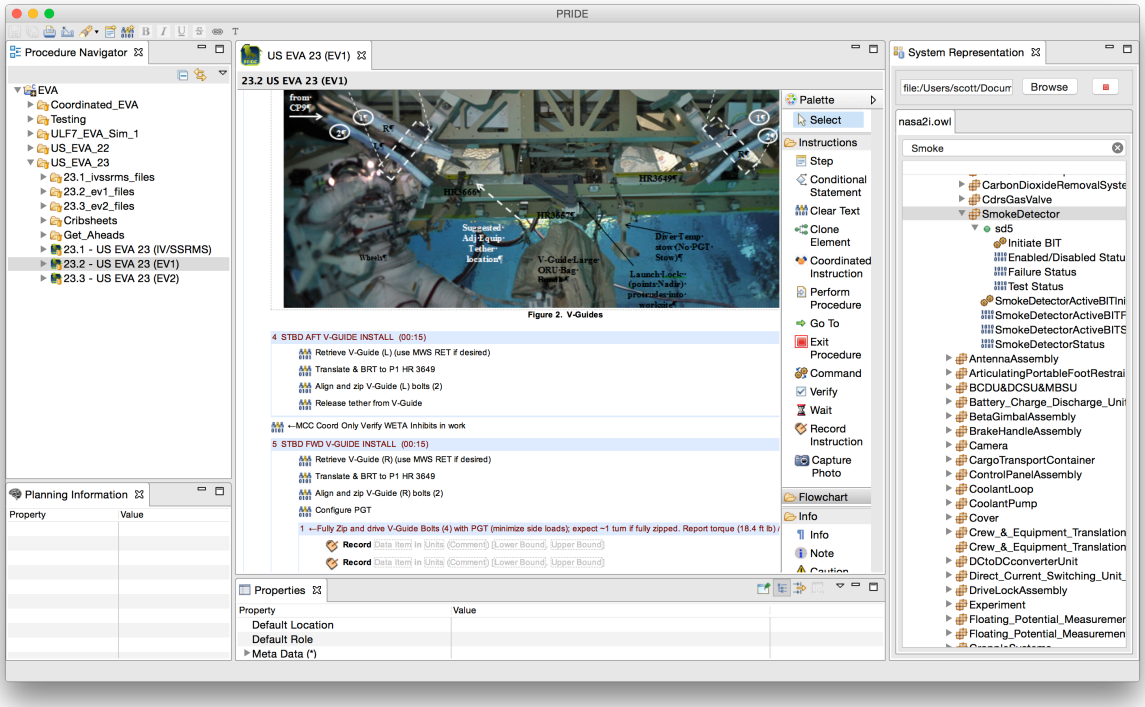

Figure 2 *A screenshot of an EVA procedure being developed in the PRIDE Author application.*

Preparing the Procedures for AR Support

Figure 2 shows the PRIDE Author interface. Users drag instruction types from the palette and drops them onto the center canvas, where they can edit the details. Users can also drag and drop entities from a domain ontology (Bonasso et al., 2013)(the System Representation pane in the figure) into an instruction where their URIs are embedded in the resulting Procedure Representation Language (PRL) XML file (Kortenkamp et al., 2008). Any of these

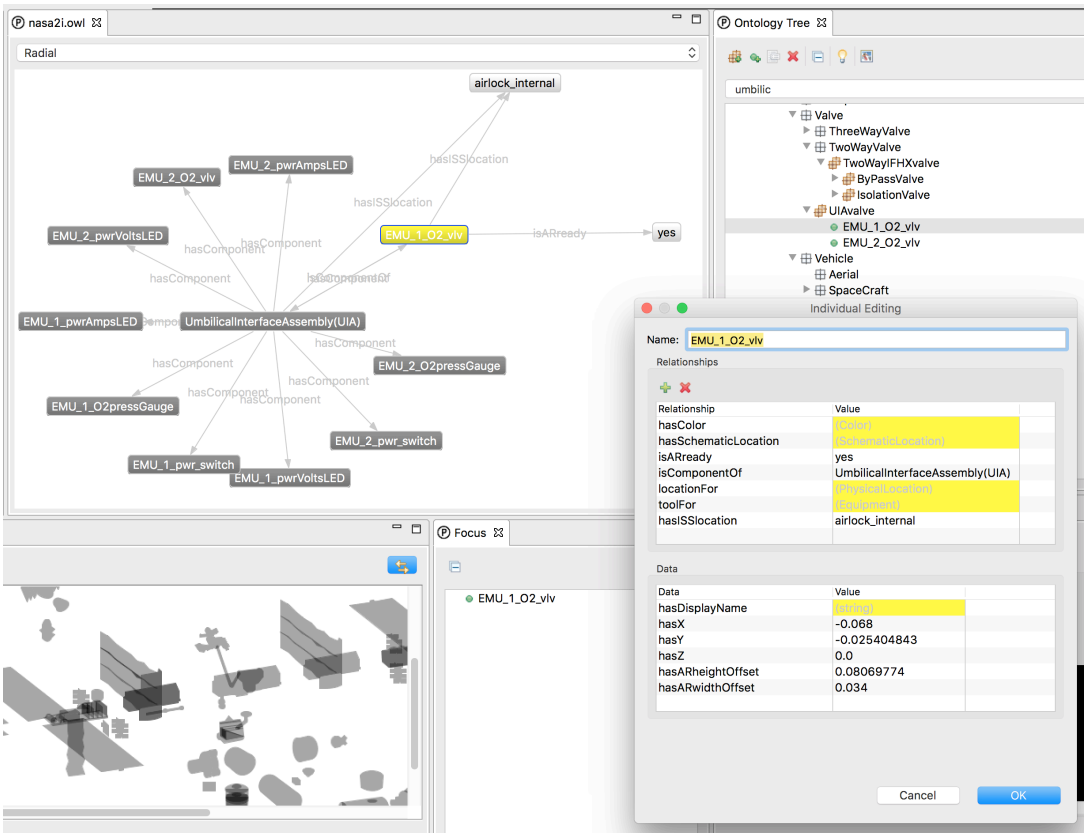

Figure 3 *A screenshot of PRONTOE showing a valve with its AR display offsets.*

entities can be AR objects, that is, they have properties germane to viewing in an AR system (e.g., see **Figure 3**).

Integrating Procedure Execution with Alternate Realities

As a procedure is executed, the AR server monitors its progress via a REpresentational State Transform (REST) interface (Fielding, 2000) to the PRIDE View server. If it detects the URI of any entity in the ontology, it queries the ontology server for that entity's properties, and if there are AR properties, it communicates them along with the instruction text to the AR system being used. For example, **Figure 3** shows coordinate offsets for a valve switch on a control panel. **Figure 4** shows that offset being used to draw a rectangle around the appropriate switch on the control panel.

Three Demonstrations[2]

The critical function of the AR server is to decide on an appropriate AR cue to send to the AR viewers. We developed three demonstrations that make use of the following AR cues:

Figure 4 *An iPad view of a control panel with the oxygen valve connected to EMU (spacesuit) 1.*

[2] Go to https://traclabs.com/projects/alternate-realities/ to see videos of these demonstrations.

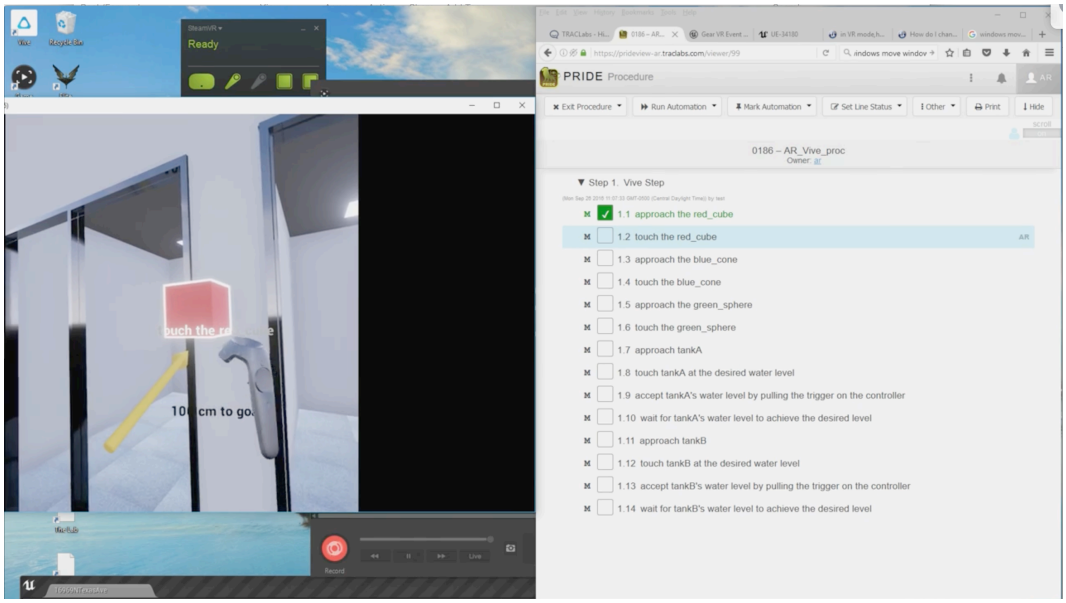

Figure 5 *View from HTC Vive headset alongside an external view of the executing procedure.*

• Show: highlight an object (outline, glow, flash, etc.), or provide additional details (e.g., wiring diagrams) as insets or overlays
• Instruct: Place a text instruction in a specific place in the field of view or when the user looks at a specific location
• Locate/Find: Show an object's location, for example, by displaying an arrow pointing to the object if it is in the field of view or pointing in the object's direction if it is not
• Data: show (live) data related to an object near that object or near an instruction referencing that object, for example, when a piece of telemetry needs to be verified in a procedure.

Each of these demonstrations uses the same PRIDE-AVR architecture as shown in **Figure 1**. The only changes are to the individual procedures, the ontology, and the hardware output device. It is important to remember that the procedure author does not need to do anything special to create augmented and virtual reality procedures. They simply drag objects from the ontology into the procedure and the AR server automatically turns them into AR cues. Moreover, PRONTOE ensures that the ontology can also be maintained by subject matter experts who need no programming experience. Thus, PRIDE-AVR allows flight controllers and other experts to create and change alternate reality systems with no coding or knowledge of those systems.

Augmented Reality Browser

We used PRIDE Author to create a portion of the Extravehicular Mobility Unit (EMU), or spacesuit, checkout procedure. This procedure has a significant number of instructions that refer to different components of the EMU and of the Umbilical Interface Assembly (UIA) to which it is connected. These components were modeled in the ontology (**Figure 3**). We printed a 2x3 foot image of the UIA and used it to create a Vuforia1 Image Target[3]. We then used the Argon browser (MacIntyre et al., 2011) and JavaScript framework developed at Georgia Tech's Augmented Environments Lab to create a web server that combined the

Figure 6 *View of the Real-world Fluid Transfer System (FTS)*

Vuforia image tracking technology with the data coming from the AR server to provide an augmented reality view of the EMU checkout procedure in a browser running on an iPad (**Figure 4**).

[3] www.vuforia.com

PTC's Vuforia image tracking software (http://www.ptc.com/en/about/history/vuforia), embedded in the Argon4 browser, has a 2D coordinate system for the photograph. The position and size of the instruments on the panel, in the coordinate system of the image used for tracking, were stored in our ontology and passed by the AR server to the Argon4 web application as parameters to the augmentation http command. These values are entirely dependent on the system used for tracking the position and orientation of the display device relative to the control panel. The next version of our implementation will be to express these locations relative to the parent component (e.g., Umbilical Interface Assembly (UIA)), so that when we are tracking the location of the component relative to the display, we can render the augmentations appropriately in 3D, just as we are doing in the VR system.

If the UIA (or a portion of it) is in the camera field of view as reported by Vuforia, then the current procedure instruction is displayed at the bottom of the live camera image and any UIA component that is referenced in that instruction is outlined. A Done button is also displayed. When the Done button is displayed and clicked on by the user, a step-completed message is passed to the AR server, which instructs the PRIDE View server to automatically advance to the next instruction and the process repeats.

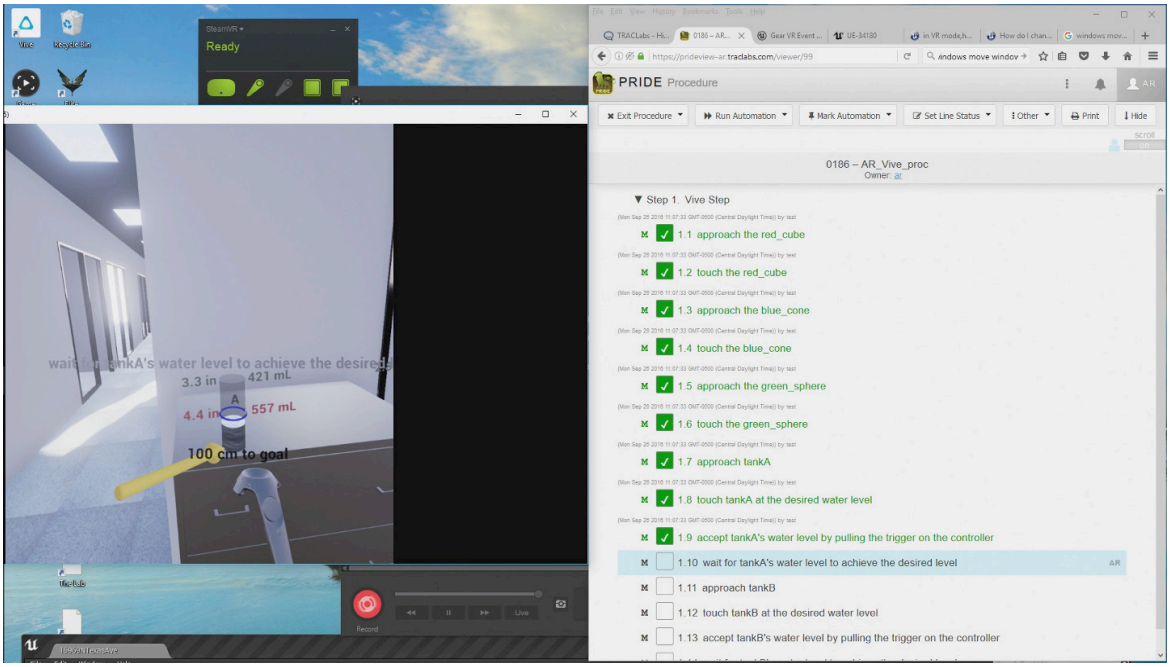

Figure 7 *View of the Virtual World Fluid Transfer System and the attendant procedure.*

Virtual/Hybrid Reality Demonstration

Our VR demo test bed consists of an HTC Vive connected to a computer in an open area of our facility. The Vive is a stereoscopic immersion platform that provides both visual and auditory information to the user, with hand controllers for interacting with the virtual environment. Multiple illuminators positioned around the open area paint patterned light on the user's headset and hand controllers, which then interpret those light patterns to extract location and orientation. That information updates the virtual environment, allowing free exploration of the virtual world within the confines of the Vive system's real-world arena.

Our VR proof of concept demonstration illustrates many of the concepts necessary to integrate electronic procedures into a VR display system. As before, the AR server tracks the executing procedure and queries the ontology server for AR properties of objects contained in the instruction. This information is presented to the VR system through a JSON RESTful interface, supplying current instruction information and accepting "instruction complete" commands for advancing to the next instruction. The VR rendering system parses current instruction information and renders guidance cues to the user in the form of textual instructions, visual guidance indicators, and spatial audio cues as directed by the AR server.

We implemented a procedure that walks the user through a set of manual tasks such as approaching and

touching various 3D virtual objects in the virtual environment (see **Figure 5**).

The TRACLabs Vive test bed also includes real-world devices with which the user can interact, thus implementing a form of hybrid reality. The second half of the demo has the user interacting with a real-world fluid transfer system (FTS) consisting of two cylindrical glass tanks with fluid level sensors, two fluid pumps, and a controller (see **Figure 6**). The FTS monitors fluid levels and accepts commands to move water from one tank to the other. The FTS provides a JSON RESTful interface for querying and commanding tank levels for both tanks. In the virtual environment, the tanks are modeled to mimic their physical characteristics in both appearance and location (see **Figure 7**). Fluid levels are reproduced in the virtual environment and the user can command new fluid levels by interacting with the virtual model.

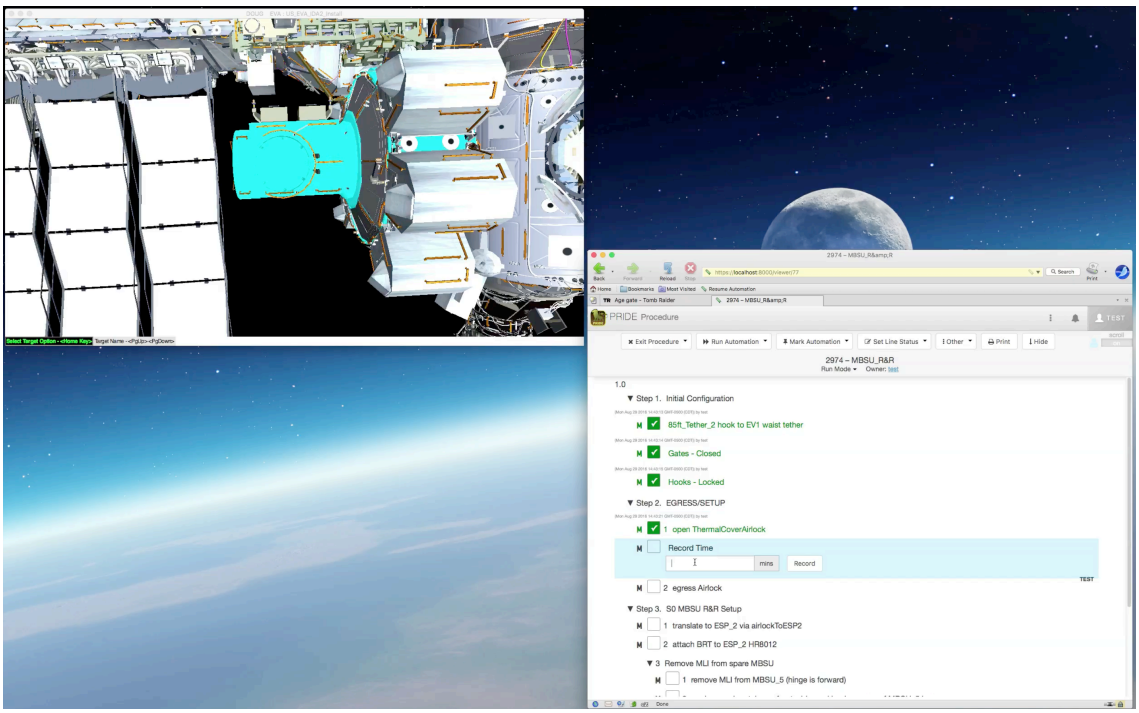

Figure 8 *View of an executing EVA procedure and the resulting DOUG display with the referenced ISS object highlighted in blue.*

AR-DOUG

In the third demonstration, the PRIDE-AVR system was used to drive NASA's Dynamic On-board Ubiquitous Graphics (DOUG) system[4], which is used to train astronauts for Extravehicular Activities (EVAs). Again, the AR server monitors a procedure executing in PRIDE View and parses the current instruction for references to objects in the ontology. For any references found, it queries the ontology to obtain DOUG info, such as the ISS location and the DOUG name. If it finds such information it commands DOUG to "fly to" the referenced object, and flashes the object (see **Figure 8**). We also added crew translation paths to the ontology, so, if the instruction being executed describes a translation action and references a translation path, the AR server will instruct DOUG to highlight the handrails and other hand holds as DOUG flies the view camera along the path.

Next Steps

Our work thus far shows that our PRIDE_AVR system architecture is feasible enough to support all three kinds of alternate reality viewers. Our next steps are to extend and enhance various components of the system to be able to address a larger set of requirements. For example, we will extend the AR server to reason over additional cues, such as audio cues, the use of countdown timers, and using additional media such as short movies or animations.

Our current system had a viewer plug-in for the HTC Vive using the Unreal engine and development environment. We will continue to expand that viewer plug-in with a goal of deploying it in the NASA Hybrid Reality Labora-

[4] https://vrlab.jsc.nasa.gov/

tory (HRL) HTC Vive environment. Used for inexpensive training of astronauts, the HRL has 3D models of the ISS (both interior and exterior) and allows users to move through the ISS and interact with objects in it. We are developing a tutorial procedure that guides the user through the various types of interactions in the HRL.

We will also extend our plug-in suite to include the Microsoft HoloLens augmented reality system.

References

Bell, S., Bonasso, R. P., Boddy, M., and Kortenkamp, D. 2013. PRONTOE: A case study for developing ontologies for operations. In *5th International Conference on Knowledge Engineering and Ontology Development (KEOD 13)*. Algarve, Portugal.

Bonasso, R. P., Kortenkamp, D., Boddy, M., and Bell, S. 2013. Ontological Models To Support Planning Operations. In *Proceedings of Internation Workshop on Planning and Scheduling in Space (IWPSS 13)*. Mountain View, CA.

de Souza e Silva, A. S., Daniel M. 2009. *Digital Cityscapes: merging digital and urban playspaces*. New York: Peter Lang Publishing, Inc.

Fielding, R. T. 2000. Architectural Styles and the Design of Network-based Software Architectures (chapter 5: Representational State Transfers). Computer Science, UC Ervine.

Izygon, M., Kortenkamp, D., and Molin, A. 2008. A procedure integrated development environment for future spacecraft and habitats. In *Space Technology and Applications International Forum (STAIF)*, vol. 969. Albuquerque, NM: American Instutute of Physics.

Kortenkamp, D., Bonasso, R. P., and Schreckenghost, D. 2008. A Procedure Representation Language for Human Spaceflight Operations. In *The 9th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS-08)*. Los Angeles, CA.

MacIntyre, B., Hill, A., Rouzati, H., Gandy, M., and Davidson, B. 2011. The argon ar web browser and standards-based ar application environment. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. Basel Basel, Switzerland: IEEE.

Tang, A., Owen, C., Biocca, F., and Mou, W. 2003. Comparative effectiveness of augmented
reality in object assembly. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. Fort Lauderdale, FL: ACM, 73-80.