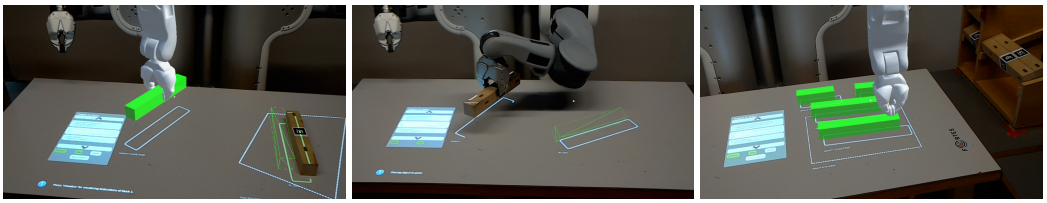
BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2018
http://excel.fit.vutbr.cz

Combining Spatial Augmented Reality Interface with Mixed Reality Head-mounted Display to Enhance User Understanding of Robotic Programs

Daniel Bambušek*



Abstract
This paper focuses on usability of mixed reality head-mounted display – Microsoft HoloLens – in human-robot collaborative workspace – ARTable. Use of the headset is demonstrated by created user interface which helps regular workers to better and faster understand the ARTable system. It allows to spatially visualize learned programs, without the necessity to run the robot itself. The user is guided by 3D animation of individual programs and by device voice, which helps him to get a clear idea of what will happen if he runs the program directly on the robot. This approach could be also helpful when the user wants to verify that he programed the robot as he wanted or just wants to quickly overview what will be the result of the current program. Using mixed reality displays enables to visualize valuable spatial information, such as robot perception.

Keywords: ARTable — Microsoft HoloLens — Augmented Reality — mixed reality head-mounted display — human-robot interaction — PR2 — program visualization — human-robot collaborative workspace

Supplementary Material: Demonstration Video — Downloadable Code

*xbambu04@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

1. Introduction

As industrial collaborative robots are getting more affordable, it is likely that small and medium enterprises will soon adopt such robots in order to increase productivity. However, in such enterprises, production batches are smaller and products may be customized for a specific contract. This requires reprogramming robots for particular tasks, which could be challenging due to necessity of robot-specific knowledge. Thus it would be beneficial to enable ordinary-skilled worker to program these robots easily. Therefore a prototype of a human-robot collaborative workspace – the ARTable was created (see Figure 1), which presents a novel approach to programming robots based on cognition, spatial augmented reality and multimodal input and output [1].

Programming of collaborative robots in this approach is based on setting up parameters for specific instructions which forms the whole program. These instructions are described only with text identifiers and as experiments showed up, users had difficulties in orienting in such program topology. There were uncertainties of how instructions follow, what exactly will specific program do or what is expected from user to program.

This work aims to clarify user's uncertainties in programming robots by providing program visualiza-

tion system in spatial augmented reality with use of Microsoft HoloLens. Being able to see what will happen when specific program runs, what specific instruction means and what is expected from the user to program should rapidly help in understanding of the whole ARTable system. Proposed program visualization is fully controllable with voice or projected buttons, so even a user familiar with the ARTable might find it usable, when he can validate correctness of a new program or quickly visualize an existing one and see it's outcome by skipping to the end, without the necessity of running it on the robot itself.

2. Background

There are several approaches of end-user robot programming. One of the main techniques are Programming by Demonstration and Visual Programming. Both approaches are combined in a robot programming system – *Code3* [2], which enables a non-roboticist programmers to create a complex robot programs using landmarks detection, programming by demonstration and visual drag-and-drop programming interface that lets users define control flow logic of their programs.

Usage of augmented reality (AR) in industrial robotics may lead to decreased workload throughout robot control by showing useful information to users [3]. A concept and architecture for programming industrial robots with use of AR is presented in [4]. In [5], authors developed an application that augments an industrial robot for shop floor tasks and evaluated different approaches of augmentation – using mobile phone or smart glasses (Epson Moverio). Glasses suffered from need of continuous marker tracking and limited field of view. However, usage of hand-held devices might be limiting because it prevents usage of both hands.

Use of the mixed reality head-mounted displays frees user's hands and as Rosen et al. [6] proved, 3D visualization of valuable information is more effective than 2D visualization. They used Microsoft HoloLens to visualize robot arm motion intent and asked participants to determine which robot motions are going to collide with obstacles on the table. They reached up to 16% increase in accuracy and 62% decrease in time it took participants to complete the task when using HoloLens.

3. The ARTable – Augmented Reality Collaborative Workspace

As the title suggests, the term "ARTable" stands for a prototype of a human-robot collaborative workspace

Figure 1. Setup of the human-robot collaborative workspace – the ARTable. In this example the user sets program parameters using robot's arm and gestures.

based on augmented reality. It was created by the research group Robo@FIT[1] at Brno University of Technology. The prototype focuses on small and medium enterprises where it should enable ordinary-skilled worker to program a robot on a high level of abstraction and perform collaborative tasks effectively and safely. It aims to show possibilities of the collaboration between human and robot in near future [1].

The source code and technical documentation of this prototype is available at github[2].

3.1 Core Components

The ARTable setup uses the PR2 robot from Willow Garage, which is located behind the table (from user's perspective) and serves as a demonstrator of a near-future collaborative robot. However, the setup is robot independent.

The table itself has a touch-sensitive layer onto which an interface is projected using Acer P6600 projector mounted on the construction above the table. The projected interface contains various elements to visualize state of the robot and current task. Together with the touch-sensitive layer, the table serves as the main source of input for the system and feedback for the user. Another possible way of passing input into the system is by direct manipulation with robotic arm. The table is complemented with speakers on each sides, which serves as supplementary source of feedback for the user.

[1] http://www.fit.vutbr.cz/research/groups/robo/index.php.en

[2] https://github.com/robofit/artable

The system uses two Kinects version 2, mounted on a stable tripod on each side of the table and one Kinect version 1, mounted on the head of the PR2, for a detection and tracking of objects in the scene. Each tripod has it's own processing unit (Intel NUC) where the projector and sensors are connected. This units are connected to the central computer. AR codes, attached to objects, are used for better tracking. They are also used for calibration of the whole system. Regarding to software setup, the ARTable uses the Robot Operating System (ROS).

3.2 Supported Programs and Instructions

Programs are sets of instructions which are collected into blocks. These instructions are linked together based on the result of specific instruction (success or failure). Currently, the system supports parametric instructions as *pick from polygon* (to pick an object from specified polygon on the table), *pick from feeder* (to pick an object from gravity feeder), *place to pose* (to place previously picked object to selected place on the table) and *apply glue* (to simulate gluing). User sets parameters in these instructions, such as object type to pick, place position of picked object or robots gripper position for picking objects from feeder.

Besides mentioned instructions, there are also few non-parametric instructions: *get ready* (which moves robot arms to a default position), *wait for user* (which pauses the program execution if user moves away from the table) and *wait until user finishes* (which pauses the program execution until user finishes current interaction with objects on the table).

4. Integration of Microsoft HoloLens into the ARTable System

Microsoft HoloLens is a headset for mixed reality, developed by Microsoft. It's wireless, runs Windows 10, has precise spatial mapping [7], and is capable of 3 ways of interaction – gaze, gestures and voice. Device draws computer generated holograms into the real world and has a potential to be used in many scenarios, for instance in medicine, to provide doctors with "X-ray vision" during surgery [8], in education, in architecture, to overview buildings in 3D model size [8] or just to augment skilled workers and technicians in the field to help the users with tasks in which they lack experiences [9]. However, the device is suffering with really small field of view – 30° x 17.5° degrees (aspect ratio 16:9) [10], which is a limitation that negatively affects experience with use in ARTable system.

In order to successfully integrate the HoloLens into the ARTable system there were two major steps to take

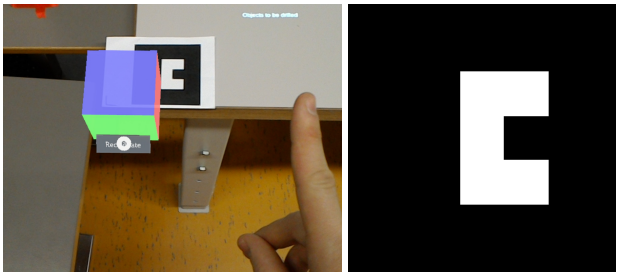


Figure 2. Left: Calibration of the HoloLens with respect to the ARTable. Colored cube visualizes how well the calibration was done and has an option button for recalibration. Red side of the cube represents positive x axis, green side represents positive y axis and blue positive z axis. **Right:** Marker used for calibration acquired from ARToolKit.

– make communication between these two systems work and calibrate the HoloLens with respect to the ARTable. For development, the Unity[3] was chosen, which is a cross-platform game engine used mainly for developing 3D and 2D video games or simulations.

4.1 Communication via rosbridge

The ARTable system runs on ROS with Ubuntu but HoloLens runs on Windows. So there is a problem of how to transport data from one operating system to another. This problem is solved by ROS tool – *rosbridge*, which is a tool that provides a JSON API to ROS functionality for non-ROS programs.

For such communication I used and extended existing Unity library – *ROSBridgeLib*[4] (author Michael Jenkin, edited by Mathias Ciarlo Thorstensen). This library uses *SimpleJSON*[5] parser for parsing JSON messages. However, due to differences in scripting backends of Unity editor (Mono) and UWP applications (.NET), functionality of this library had to be extended to support the UWP format which is used by HoloLens. With inspiration from Unity library *holoROS*[6], created by Gabriel Santos Solia, who successfully integrated Microsoft HoloLens with ROS via *rosbridge*, in order to simulate a holographic *turtlesim*[7] environment, I was able to extend the *ROSBridgeLib* functionality to support communication between the ARTable (ROS) and Unity along with HoloLens (Windows).

Having this type of communication enables to connect multiple headsets with the ARTable, where all of them receive same data.

[3] https://unity3d.com/
[4] https://github.com/MathiasCiarlo/ROSBridgeLib
[5] https://github.com/Bunny83/SimpleJSON
[6] https://github.com/soliagabriel/holoROS
[7] http://wiki.ros.org/turtlesim

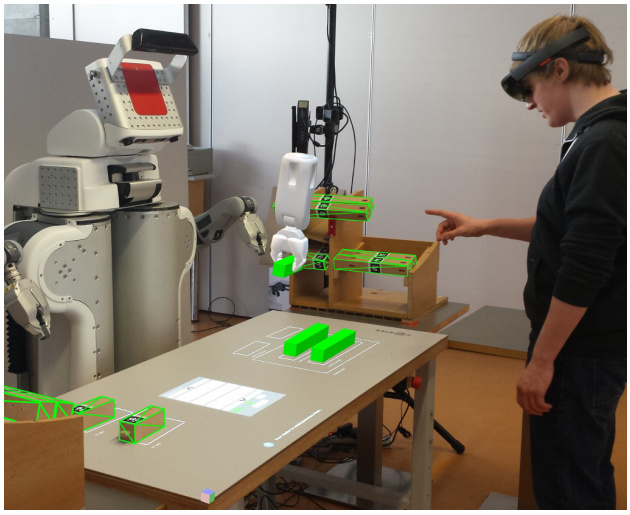


Figure 3. Spectator's view of the workspace extended by virtual holograms seen through the headset. In this situation the user observes visualization of "Stool assembly" program.

4.2 HoloLens Calibration with Respect to the ARTable

The ARTable is calibrated with use of three markers placed into corners of the table. Origin of coordinate system is estimated in bottom left corner of the table. The HoloLens also needs to know where that corner is. This is accomplished by detection of a marker placed in that spot with use of HoloLens world-facing camera (see Figure 2). For detection purposes, the *HoloLens-ARToolKit*[8] was used, which is a Unity library that integrates ARToolKit[9] functionality with UWP applications, created by Long Qian.

If marker detection is successful, colored cube is drawn on this marker spot which helps to visualize how well the calibration was done. This cube is afterwards wrapped with an empty Unity *gameObject* which is then stored as a spatial anchor and set as a parent to every other visual object in the scene. Thus it represents origin of the table. Spatial anchors are persistent during sessions, so calibration process needs to be done only once. There is also possibility of recalibration by clicking the *Recalibration* button as can be seen in Figure 2.

After calibration, one last step has to be made. There is a conflict in coordinate systems, the ARTable uses right-handed while Unity and therefore HoloLens uses left-handed. To solve this problem, every object position and rotation that came from ARTable needs to invert it's y axis and Euler x and z angle.

[8] https://github.com/qian256/HoloLensARToolKit

[9] https://github.com/artoolkit/artoolkit5

5. User Interface for Robot Programs Visualization

Current 2D projected user interface is limited in visualization of spatial information. For instance, it draws bounding boxes around detected objects placed on the table, that indicates what robot actually sees. However, sensors are able to detect these objects even if they are not placed directly on table, e.g. when they are in gravity feeder. Also the way it displays set parameters for a given program is not intuitive enough. Users were often uncertain in how instructions follow, what exactly will specific program do or what they are supposed to program.

Provided solution eliminates limitations of 2D projected interface. With use of the Microsoft HoloLens, drawing of 3D spatial bounding boxes around detected objects is easily solved. In order to eliminate user's uncertainties in programming the robot, programs visualization mode was implemented. This mode extends current 2D projected interface by *Visualize* button which starts the visualization itself. Only programs with set parameters are able to visualize (see Figure 4). After hitting this button, the interface checks if HoloLens are running, otherwise visualization won't start. Visualization is fully controllable with provided buttons – *Pause/Resume*, *Stop* and *Replay* (as shown in Figure 4).

As described in subsection 3.2, ARTable programs are compounded of various instructions that are linked together. Instruction set of currently visualized program is loaded from ROS environment into HoloLens which then dynamically builds this program from internally implemented classes representing these instructions. With this approach any program variations built from supported instructions should visualize correctly. After build, execution starts.

Example usage of the headset is shown in Figure 3, where the user observes visualization of "Stool assembly" program. Visualization of this program is further demonstrated in Figure 5. Whole process visualizes individual instructions which are connected together in a context of the program. For example, when visualizing *pick from feeder* instruction, virtual object is created, which is then picked by a virtual robot gripper from the feeder and placed on the preset position on the table. This object then stays active in the scene in case that another instruction could want to manipulate with it (e.g. to apply glue to it or to move it somewhere else). Whole process is commented by speech synthesizer. Besides provided buttons for controlling the visualization, user can use his voice. Introduced system recognizes couple of basic keywords: "pause"/"resume",

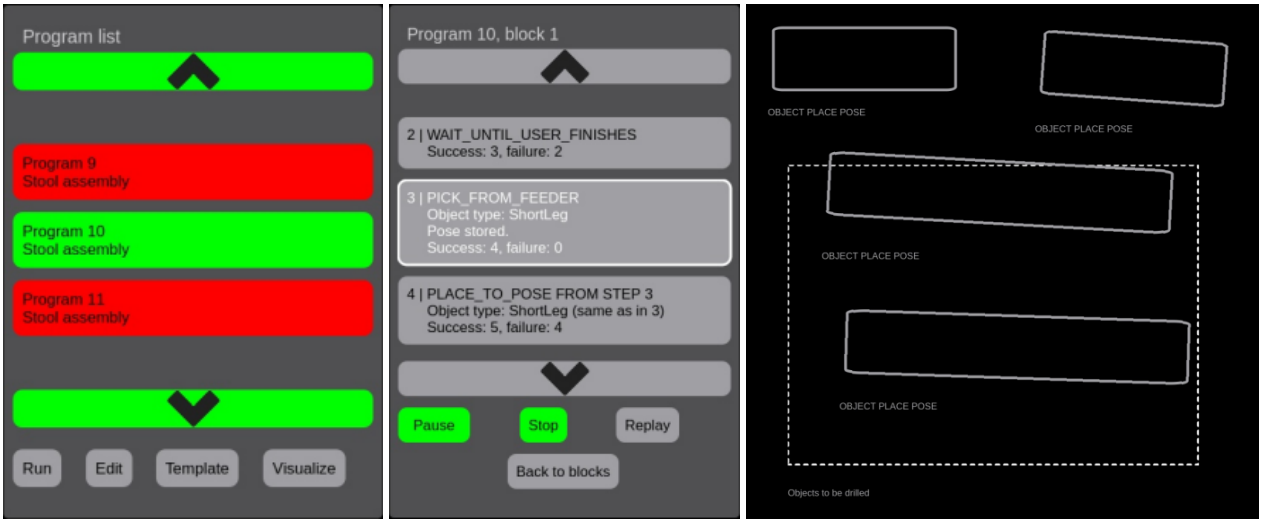
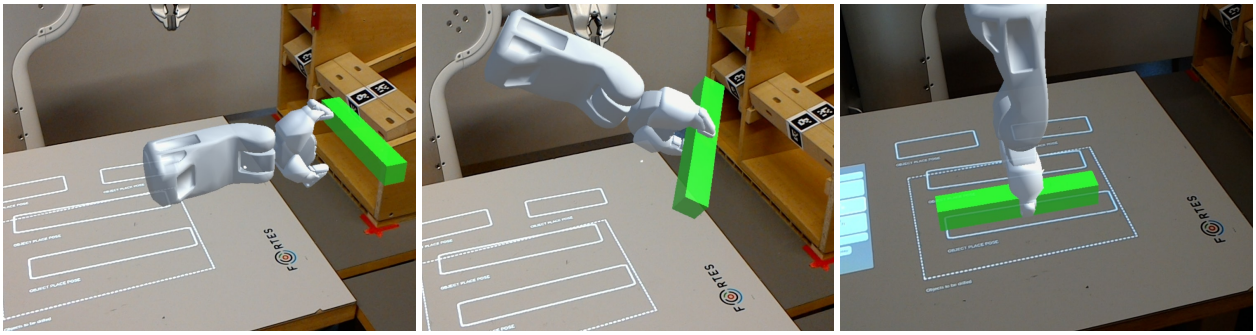


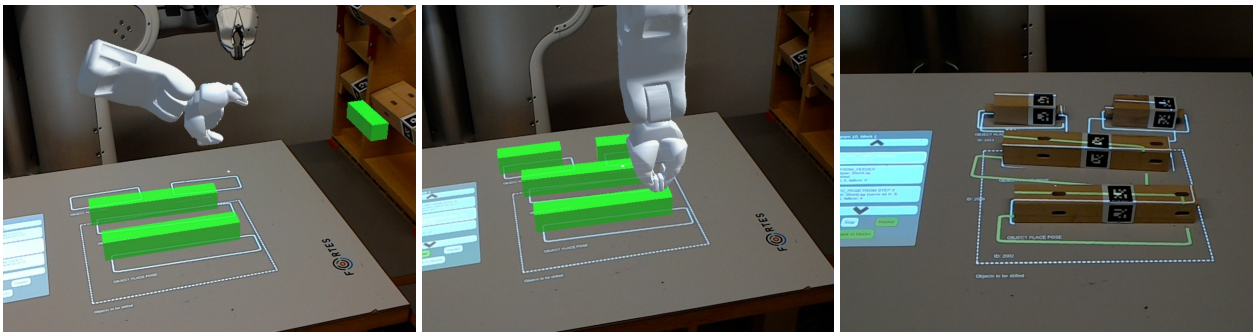
Figure 4. Widgets of projected ARTable GUI extended by visualization mode. **Left:** List of programs, where green programs are ready to visualize or run and red programs need to set parameters. **Middle:** List of instructions of currently visualized program. Visualization is fully controllable with provided buttons – "Pause"/"Resume", "Stop", "Replay". Visualization can be also controlled with voice by saying same keywords as are names of those buttons. **Right:** Rectangles defining areas on the table onto which the grabbed objects of different type will be placed. Dashed polygon defines area in which the robot will apply glue to corresponding objects. All these supplementary elements of projected GUI are displayed at the same time when program visualization runs.



(a) Virtual robot gripper is grabbing virtual object from the feeder.

(b) The robot gripper moves grabbed object to specified place pose.

(c) The robot is placing virtual object into preset position.



(d) The robot goes for smaller virtual objects – stretchers.

(e) Simulation of glue application to virtual objects within defined area.

(f) Result of "Stool assembly" program running on the real robot.

Figure 5. An example of program visualization. In this case – first block of "Stool assembly" program which consists of a total of three blocks. In this block the robot will pick the objects from the feeder, place them onto the table and apply glue into the holes in order to prepare these components for user to assembly. This block consists of eleven instructions – "wait until user finishes", four "pick from feeder" and consequent "place to pose" instructions, "apply glue" and lastly "get ready".

"stop", "replay", "next" (to immediately move to next instruction) and "back" (to immediately move to previous instruction). These keywords are insured against unexpected behavior. For example if users says "stop" when visualization is already stopped, he gets notified by speech synthesizer.

During whole visualization process, the HoloLens and the ARTable keeps communicating using aforementioned *rosbridge*. Both systems synchronizes their states about which instruction is visualizing, whether it's playing or is stopped and so forth. This continuous communication ensures scrolling down in instruction list of projected 2D interface or enabling and disabling projected buttons. Lastly, supplementary elements of projected interface – such as object place pose rectangles, polygons defining areas from which to pick or in which to apply glue to objects – are displayed at once to help the user get an instant overview of an outcome of current program.

6. Conclusions

In this paper, usage of mixed reality head-mounted display was integrated into a human-robot collaborative workspace – the ARTable. Current 2D projected interface was extended over the visualization mode, which adds new buttons and functionality in order to successfully communicate with the headset. A visualization system that enhances user understanding of robotic programs in the ARTable workspace was implemented. This system enables the user to visualize learned programs by a form of animation, allows him to control this animation by his voice or buttons of the projected interface and guides him by speech synthesizer through whole visualization process. With possibility of controlling the visualization, even an experienced user might find it usable, when he can quickly see program's outcome without the necessity of running the real robot. Use of the headset allows to visualize valuable spatial information, such as robot perception, which enhances human-robot collaboration.

Provided solution enables to use multiple headsets, where all of them are synchronized through the ARTable. Thanks to that, they use same information and render same holograms. Solution of the ARTable's part is headset independent.

For future plan, this work will be extended over a functionality, that will interactively guide a user through programming of the ARTable programs with use of speech synthesizer and 3D holograms. Further on, proper user testing will take place.

Acknowledgements

I would like to thank my supervisor Michal Kapinus for his help.

References

[1] Zdeněk Materna, Michal Kapinus, Vítězslav Beran, and Pavel Smrž. Using Persona, Scenario, and Use Case to Develop a Human-Robot Augmented Reality Collaborative Workspace. In *HRI 2017*, pages 1–2. Association for Computing Machinery, 2017.

[2] Justin Huang and Maya Cakmak. Code3: A System for End-to-End Programming of Mobile Manipulator Robots for Novices and Experts. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '17, pages 453–462, New York, NY, USA, 2017. ACM.

[3] Susanne Stadler, Kevin Kain, Manuel Giuliani, Nicole Mirnig, Gerald Stollnberger, and Manfred Tscheligi. Augmented Reality for Industrial Robot Programmers: Workload Analysis for Task-based, Augmented Reality-supported Robot Control. In *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 179–184, Aug 2016.

[4] Jan Guhl, Son Tung, and Jörg Kruger. Concept and Architecture for Programming Industrial Robots using Augmented Reality with Mobile Devices like Microsoft HoloLens. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4, Sept 2017.

[5] Ivo Malý, David Sedláček, and Paulo Leitão. Augmented Reality Experiments with Industrial Robot in Industry 4.0 Environment. In *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, pages 176–181, July 2016.

[6] Eric Rosen, David Whitney, Elizabeth Phillips, Gary Chien, James Tompkin, George Konidaris, and Stefanie Tellex. Communicating Robot Arm Motion Intent Through Mixed Reality Head-mounted Displays. *CoRR*, abs/1708.03655, 2017.

[7] Ivo Sluganovic, Matej Serbec, Ante Derek, and Ivan Martinovic. HoloPair: Securing Shared Augmented Reality Using Microsoft HoloLens. In *ACSAC*, pages 250–261, December 2017.

[8] Bonnie Christian. HoloLens trial gives doctors 'X-ray vision' to allow them to peer inside

patients during surgery. WIRED, March 2017. http://www.wired.co.uk/article/industries-using-microsoft-hololens.

[9] Laura Toler. Holographic Rovers: Augmented Reality and the Microsoft HoloLens. Technical report, NASA Kennedy Space Center; Cocoa Beach, FL United States, April 2017.

[10] Oliver Kreylos. HoloLens and Field of View in Augmented Reality, August 2015. http://doc-ok.org/?p=1274.