



Article

X-Reality System Architecture for Industry 4.0 Processes

Bruno Simões ^{1,*}, Raffaele De Amicis ², Iñigo Barandiaran ¹ and Jorge Posada ¹

¹ Vicomtech, 20009 Donostia/San Sebastian, Spain; ibarandiaran@vicomtech.org (I.B.); jposada@vicomtech.org (J.P.)

² School of Electrical Engineering & Computer Science, Oregon State University, Corvallis, OR 97331, USA; raffaele.deamicis@oregonstate.edu

* Correspondence: bsimoes@vicomtech.org

Received: 6 August 2018; Accepted: 2 October 2018; Published: 15 October 2018



Abstract: Information visualization has been widely adopted to represent and visualize data patterns as it offers users fast access to data facts and can highlight specific points beyond plain figures and words. As data comes from multiple sources, in all types of formats, and in unprecedented volumes, the need intensifies for more powerful and effective data visualization tools. In the manufacturing industry, immersive technology can enhance the way users artificially perceive and interact with data linked to the shop floor. However, showcases of prototypes of such technology have shown limited results. The low level of digitalization, the complexity of the required infrastructure, the lack of knowledge about Augmented Reality (AR), and the calibration processes that are required whenever the shop floor configuration changes hinders the adoption of the technology. In this paper, we investigate the design of middleware that can automate the configuration of X-Reality (XR) systems and create tangible in-site visualizations and interactions with industrial assets. The main contribution of this paper is a middleware architecture that enables communication and interaction across different technologies without manual configuration or calibration. This has the potential to turn shop floors into seamless interaction spaces that empower users with pervasive forms of data sharing, analysis and presentation that are not restricted to a specific hardware configuration. The novelty of our work is due to its autonomous approach for finding and communicating calibrations and data format transformations between devices, which does not require user intervention. Our prototype middleware has been validated with a test case in a controlled digital-physical scenario composed of a robot and industrial equipment.

Keywords: X-Reality; mixed reality; projector-camera systems; projected augmented reality; applications of SAR; industry 4.0

1. Introduction

The manufacturing sector is in the midst of the “industry 4.0” revolution. It is migrating the layout of its shop floors to dynamic spaces that integrate both humans and intelligent machines. The industrial equipment being added to shop floors is more complex than ever and becoming part of more complex and intelligent production systems (e.g., feeding predictive maintenance services, being controlled by flow optimization systems, etc.). Commonly, industrial equipment and control devices (e.g., Programmable Logic Controllers, Sensors, and Actuators) lack the embedded storage and computational resources necessary to analyze data in real time and can neither be used to obtain data insights nor unlock new sources of economic value.

Industrial solutions based on Cloud computing can facilitate data flow within ecosystems characterized by high heterogeneity of technologies and enable data analysis of unprecedented

complexity and volume [1,2]. This way of processing information utilizes computational power and efficient visualization tools from Information Visualization and Visual Analytics. Information Visualization is “the use of computer-supported, interactive, visual representations of abstract data to amplify cognition” [3] (p. 7), while Visual Analytics is “the science of analytical reasoning facilitated by interactive visual interfaces” [4]. Unfortunately, the use of Information Technology (IT) architectures that rely only on Cloud platforms might fail to comply with stringent industrial requirements. For example, Cloud computing cannot effectively handle tasks such as the transfer of large data volumes in short time spans or the provision of in-site interactions with low data latency. Future industrial applications are now migrating towards a “Fog” [5] and “Edge” [6] infrastructure to streamline their utilization of their Industrial Internet of Things (IIoT) devices and deployment of processes that include multi-latency systems, delegating to Cloud platforms the computation of insights and optimization directives for multiple sites or long-term strategies. The dynamic nature of edge infrastructures raises communication issues that traditional applications do not consider, such as operating devices in separate networks.

In recent years, advances in immersive technology have resulted in new interactive visualization modalities with the potential to change the way we artificially perceive and interact with data [7]. X-Reality (XR) refers to all real-and-virtual combined environments and human-machine interactions generated by computer technology. The concept of XR is more extensive than virtual reality (VR), augmented reality (AR) and mixed reality (MR). XR technology mixes the perceived reality as MR does, but it is also capable of directly modifying real components [8,9]. This technology enables intelligence on the shop floor, where data sensing is taking place. The physical proximity of this intelligence to industrial equipment creates new possibilities for the support of human workers in human-machine hybrid environments. This intelligence is characterized by smart machines with mobility capabilities with new forms of visualization and control of sensors and robotic systems in these inherently collaborative work environments. However, the deployment of such augmented cognitive systems and the manipulation of real-time data in shop floors is not straightforward. There are challenges related to the security and complexity of the infrastructure on industrial shop floors, which includes: subsystems that communicate using different network protocols (e.g., Modbus [10], OPC-UA [11], and others), machines with prohibitive constraints (e.g., limited power energy, low power communications, and unreliable communications), and very strict security layers, among other things. To deploy these systems, companies have to invest in the development of new solutions that fit their infrastructure or they must modify their existing systems to meet the requirements of the new applications.

Current AR applications in literature either rely on centralized computers (e.g., Programmable Logic Controllers (PLCs), cloud platforms) to access sensors' data and control shop floor machines or are directly connected to target devices. (e.g., connected over TCP/IP in the network). This is usually not an issue when interacting with a limited number of devices, especially if they belong to the same hardware family. However, issues in communication become evident when interacting simultaneously with hardware from separate providers. Current workarounds require the development of ad hoc software packages that enable specific platforms to communicate, requiring investments that might become obsolete whenever the shop floor undergoes reconfiguration, such as for manufacturing a new family of products. Hence, the creation of immersive and personalized XR experiences requires solving basic communication problems between devices, and then developing an optimal approach for sharing and synchronizing data.

In this manuscript, we propose a middleware architecture for industrial XR systems. First, we describe the architecture for publishing and subscribing to event messages, distributing system configurations, performing real-time data transformations, and visualizing and interacting with industrial assets. Afterwards, we introduce the concept of communication hyperlinks, a mechanism we coined for data sharing that facilitates data visualization and cross-technology interaction. Communication hyperlinks provide automation and mechanisms to reduce data latency on a human level and to map local interactions (relative to the source device) to an application-level interaction

space. Hence, they enable us to embed different devices' interaction modalities into a seamless experience. The pervasiveness of this new interaction metaphor empowers users with new ways to share, analyze, and present in hybrid digital-physical environments.

Finally, we describe a validation case that uses our middleware to create immersive and personalized data exploratory experiences for industrial assets. In particular, we demonstrate how interfaces like Microsoft's HoloLens (Redmond, WA) can interweave commercial projectors, industrial machines (e.g., arm robots and sensors) and RGB-D cameras. We highlight the advantages of our device pairing feature and the potential it creates in terms of user-machine interaction. Aside from their the real-time visualization of data, users can act on projected elements and have their surroundings further enhanced with 3D features delivered by MR headsets. As digital elements are interlinked with physical objects, their manipulation might also imply remote control.

This manuscript is structured as follows: first, we review literature related to the use of augmented reality and sensor fusion visualization in industrial environments. Next, we describe the middleware architecture and the concept of communication hyperlinks. Afterwards, we describe a validation use case that enables pervasive interaction between industrial sensors and visualization devices. We conclude this manuscript with final remarks and future directions.

2. State of the Art

With increasing interest in in-site interaction, data visualization has become a key component of spatial augmented reality (SAR) [7]. However, most existing AR literature attempts to solve problems caused by data overlaying and cluttering in real scenes [12–16]. The implications of this effect both information understanding and perception [17–19]. Other authors have investigated methods to allow users to interact with physical and digital objects. Mayer and Soros described the use of smart glasses and watches to interact with objects using selection in the real world [20]. In their experiment, a smartwatch would be used to present a tactile interface for objects recognized with Google glasses, i.e., to activate a music interface when looking at an audio player. The communication between devices was mediated by a smartphone, which implemented an application to control the detected devices. Hence, their work was limited to the use pre-configured devices and to the capacity of a central node (smartphone) that was responsible for executing the interaction logic. Considering recent trends in mobile computing, it would be more practical to support dynamic connections between mobile and pre-installed devices. Rekimoto and Saitoh described Hyperdragging [21], which is a pointing device for dragging objects on a computer to different devices. When the cursor reaches the edge of the screen, it “jumps” to the device that is spatially related to the position. The authors make use of cameras and markers to spatially locate devices. The devices are preconfigured with the same Java application, which allows them to serialize and exchange data using Java Remote Method Invocation (Java RMI). The biggest limitation in their system is its design focus on the exchange of multimedia content; it can not handle data streams from sensors or additional input devices without modifying the core application. The authors of XDKinect applied Microsoft Kinect to interactions that aim at enabling user interaction with multiple devices [22]. Their work describes an application programming interface (API) for accessing events and gestures recognized by one or more Kinects attached to a central computation node. Recognized events are broadcasted with websockets to multiple clients. A study presented by [23] extended the previous work by adding multiple modalities for manipulating or interacting with specific properties of virtual objects. However, it did not overcome the aforementioned limitations. The most similar research to that which is presented in this manuscript is HoloR [24] and SoD [25]. The first is an AR system which uses projectors, goggles, and a Kinect. The former is a distributed architecture that relies on websockets to enable the interaction of HoloLens with Kinects. SoD unfortunately suffers from scalability issues despite using consumer accessible Kinect sensors, and as such is infeasible to scale to a large room. This is because each Kinect is limited to optimal tracking of “a range of 1.2 to 3.5 m” [26].

The application of SAR technology to industrial manufacturing is not a new endeavor, and, in particular, the augmentation of physical workspaces has been investigated well. At the middleware level, there is a lack of tools that allow for a high degree of customization, yet are easy to use and to extend. Some notable exception are OpenTracker [27] from Vienna University of Technology, the MR toolkit of the University of Alberta, and the OpenXR initiative of Khronos (a standard for VR and AR applications and devices that aims to eliminate industry fragmentation by enabling applications to be written once to run on any VR system). What literature is missing is a system that enables communication and interaction across different technologies without manual configuration or calibration.

Exhaustive reviews of works applying augmentation to manufacturing have been reported in [28–33]. Ong et al. [28], which studied AR in manufacturing, and Fite-Georgel [30], which reviewed the state-of-art of industrial augmented reality (IAR). Nee et al. [32] reported a survey of AR applications in design and manufacturing, and meanwhile Leu et al. [31] reviewed research methodologies for developing systems for assembly simulation, planning, and training based on computer-aided design (CAD) models. More recently, Wang et al. surveyed previous works which used AR for assembly tasks [29,33]. All of these systems were designed to provide a specific interaction with families of devices where communications were unrestricted. To our knowledge, there are no works addressing ways to integrate these tools with industrial infrastructures that contain Programmable Logic Controllers (PLCs), robots, and other complex networking elements. The topic has been researched in other scientific disciplines to ameliorate accessing and managing industrial data [34], but there is a knowledge gap in transferring those lessons to XR systems.

3. System Architecture

Self-configurable devices are the key feature that enables every technological ecosystem to adapt to new operational scenarios and to respond adequately to unplanned tasks. In this paper, we define a self-configurable device as a component in a technological ecosystem that observes the network and can establish communication rules that enable interaction with other devices. Our self-configuration process of industrial hardware operates at two levels: data transformation and communication. The operation of data transformation converts data from one device reference system into another one so to be mutually understandable by both devices. The operation of data communication establishes the communication rules for transferring data between devices.

Our architecture uses self-configuration and edge computing to perform as many computations as possible while data is held in devices' memory. Afterwards, it transfers the data according to the requirements of the network. In our context, where real-time processing and streaming of data is an essential requirement, load balancing between edge computational resources is essential to reduce overall data communication latency. Figure 1 provides an overview of the proposed system architecture.

In our architecture, we categorize devices based on their functionalities. We coin the term “coordination node” for devices that implement network security protocols or handle communications between different devices. At higher levels, coordination nodes are responsible for event broadcasting, maintaining network consistency, and optimizing network flow. We use the term “computational nodes” to refer to devices that extend the base behavior of our middleware, i.e., devices that implement services to retrieve data from other sources, transform it as necessary, process it, and then transfer it through a limited network bandwidth to other endpoints. Computational nodes are a local solution to cloud and are well suited for extreme-scale IIoT data applications which are becoming more common due to industry 4.0. The term “devices” refers to network members that provide interactivity. Devices are belong to at least one of three categories: those capable of displaying data, sensing the environment, and/or streaming contextual data.

Devices always have a direct physical counterpart, unlike coordination and computational nodes that represent a virtual version of an entity. Nodes can be replicated and deployed in strategic parts

of the network, while the devices lack such flexibility. Our load-balance and replication is ensured with HAProxy [35]. Additionally, a Dynamic Service Registration (DSR) is integrated to facilitate the service modularity. The DSR is composed by two components: the first component “Consul” [36] provides a distributed service discovery that interfaces with the second component “Registrar” [37], to automatically register and deregister computational nodes as they come online or offline.

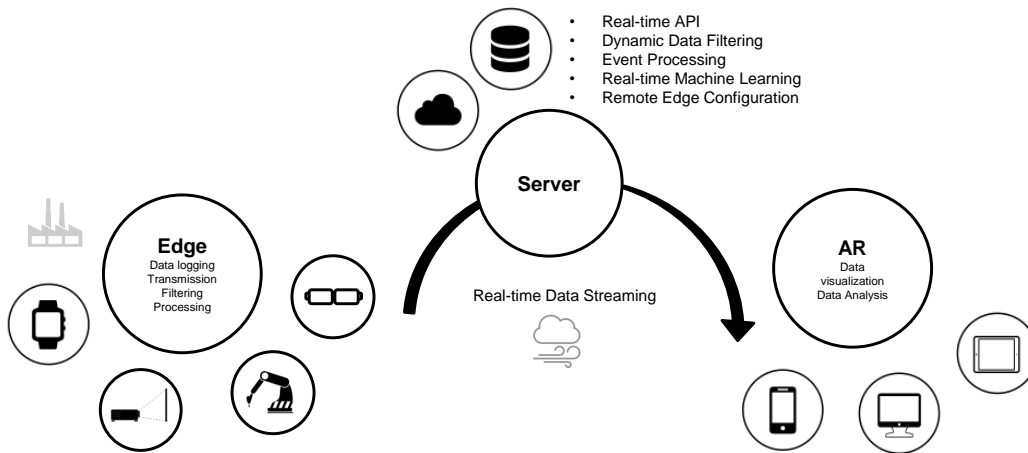


Figure 1. Middleware system architecture.

Each coordination node exposes two communication layers throughout its Service Exposure Gateway (SEG): workflow and subscription. When new devices connect to the network they initiate their self-calibration process by broadcasting a JavaScript Object Notation (JSON) message to coordination nodes in the same subnet. Messages are processed by the workflow or subscription layer depending upon the messages’ structure.

3.1. Self-Calibration Process

The extension of the middleware with new service interaction modalities requires the deployment of new devices or computational nodes into the network. The configuration of these new assets must proceed such that it is compatible with all other networked devices. In the first step of self-calibration, overall network information is requested from coordination nodes. Pertinent data about the self-configuring asset is included in this request, e.g., a description of the hardware, functionalities, an inventory of available data, and an overview of processes running on the device. The information shared with the network enables coordination nodes to determine the capabilities of the asset and to activate relevant calibration processes. There are two major calibration processes implemented in our framework: The first is related to devices that need to determine how data must be transformed prior to being exchanged with other devices, while the second process considers those assets that might want to track or detect the presence of other devices. Figure 2 describes several calibration steps that are required in the handshaking process.

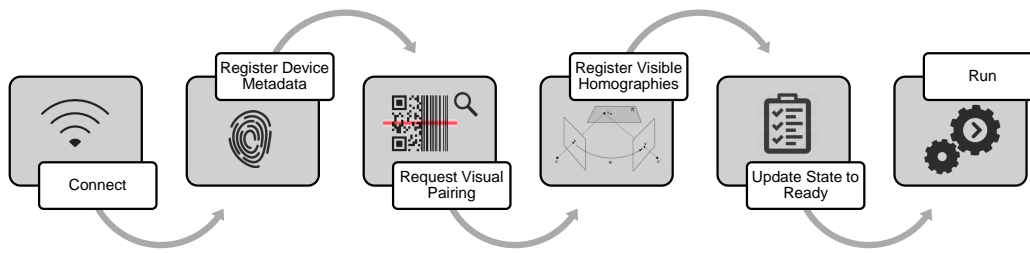


Figure 2. Process that establishes the ground rules for managing the flow of data between the two devices.

If the device is equipped with optical cameras, it is possible to compute a homography. Homography is a matrix that relates two images of the same planar surface in space (assuming a pinhole camera model). In this case, the coordination node instructs the other devices to activate their visual calibration procedure that uses planar chess-board patterns [38]. Vice versa, the display devices will compute homographies and share them either with the optically equipped one or with the computational nodes responsible for handling data transformation. Due to the fact that all homographies are known to coordination nodes, the aforementioned process allows the self-calibration of devices that are not visible to each other, as long as there is a valid transformation graph between the two endpoints.

Knowing the location of a physical device creates new possibilities both in terms of data communication, data analysis, and user interaction. The handshaking process deals with the information on tracking mechanisms and assets, such as 3D models and markers, that can be used by other devices for keeping track of the device. At the moment, our middleware implements two tracking methods. One uses the OpenCV library to support marker tracking. The second uses 3D matching in point clouds and thus requires the device to share a 3D representation of its physical appearance.

3.2. Interaction Hyperlinks

Our middleware defines an Event-Driven Architecture (EDA) which implements mechanisms to produce, detect, consume, and react to events. A JSON scheme is used for the exchange of data and actions as well as to ensure deterministic communication. Each JSON message has a well defined structure and represents a communication node within a hierarchical interaction graph. Event triggers (or constraint-based events) are groups of messages that define higher levels of interaction between users and the system as well as offer the possibility to describe parallel workflows. They do not represent the communication of data in the interaction graph. Instead, they represent how data should be handled in terms of interaction.

Hardware specifications set constraints on how data can be visualized and animated and the process of data communication between devices requires near real-time conversion of raw data into a data format and reference system that supports inverse transformations. Our proposed middleware offers a cross-device communication mechanism for translating messages, as well as mechanisms to trigger the render and streaming of data to multiple devices. Our conversion of raw data into a format within a specific reference system is extensively scalable and can be applied efficiently to a high device, sensor, and user traffic. As such, it fulfills the requirement of industrial environments for near real-time data exchange.

Interaction “hyperlinks” are visual interaction mechanisms for depicting devices’ data inputs and outputs into “docking spaces” (Images superimposed upon the surface surrounding the device with the support of projectors and/or AR headsets, see Figure 3). Docking spaces provide access to data and different visual representations thereof provided by devices. Docking spaces are designed to be manipulated, physically and digitally, to facilitate network communication and link streams of data across different devices.

The advantage of the proposed design stems from its abstract interaction mechanisms that enables the user to manipulate how devices interact and behave using physically meaningful techniques. For example, if the device supports the visualization of 3D models and scientific visualization algorithms, there are two docking areas are displayed around the device. When an object with digital data attached to it is moved to the scientific visualization docker, a list of possible visualization algorithms is shown. Then, the user can move the object to the desired algorithm. In the case of the HoloLens, a holographic menu located next to the device is used to switch the active visual representation. The menu is visible only when the device is connected to a dataset and shows only data visualization techniques that are relevant to the data been visualized. Pulling back the data-bearing object from a docking area removes the data from the target device.



Figure 3. Representation of Hyperlinks in HoloLens (3D cubes) and projected surfaces (red circles).

4. Validation Scenario

In this section, we illustrate how the proposed middleware enables seamless interaction between devices. We tested the framework with the following hardware: a Microsoft HoloLens, two RGB-D cameras, two projectors, a sensor rack with one PLC, a Universal Robot (UR), and one tablet. The objective of our experiment is to combine the AR features of Microsoft's HoloLens with the functionalities of other devices such as RGB-D cameras and to streamline development of opportunities in information handling and manipulation. The validation test aimed to assess users' abilities to create contextually immersive data exploratory experiences without programming skills.

The rationale for the selection of the hardware incorporated into the tests is two twofold. Firstly, Microsoft HoloLens has a limited set of gestures and low tracking accuracy of hands, which is a limitation we can overcome with more reliable hardware like industrial RGB-D cameras. Secondly, although projectors are more suitable for displaying information to groups of people, they lack direct user interaction mechanisms. Our architecture maintains the advantage of Microsoft's HoloLens allowing users to explore content in 3D and to manipulate 3D objects with visual feedback. Current architectures require that, when designing an interaction system, these hardware limitations are kept in mind.

4.1. System Components

Prior to our end-user test, we installed the following devices in the workspace: a content delivery network (running multiple instances of the Nginx web server) to deliver static resources (e.g., models, videos, images, etc), a Microsoft HoloLens, a UR, a PLC that provided access to temperature sensors, two projectors with independent processing unities, and four computational nodes: one to handle AR assembly instructions, another to detect objects in point cloud streams (captured by an Ensensio N20 industrial camera (Freiburg im Breisgau, Germany) and Kinect), a third to analyse video streams and track devices' locations, and one to manipulate data and transfer it to projectors. Each device had its processing unit with a computational node installed.

4.2. System Self-Configuration

The first step of our case study was to initiate the HoloLens application we developed to interpret our middleware communications, hence making it capable of sensing supported devices connected to the network. The first message exchanged between the Microsoft HoloLens and the middleware was a description of the Microsoft HoloLens capabilities. The information was logged in a pool of active devices and shared with the rule-based decision engine. Adding to the list of existent facts, the middleware defined a list of devices that could interact with HoloLens based on their reported capabilities. In response, the coordination node iteratively dispatched requests to the computational nodes in the list ordered by affinity to find a candidate which could configure the new device. It is up to the implementation of each computational node to decide whether to act or not based on variables related to the machine. On rejected requests, the middleware communicated a new fact (i.e., it was rejected) to the rule-based decision engine and assessed the next alternative through the same process. In the diagram depicted in Figure 4 the next alternative was a service to provide assembly instructions.

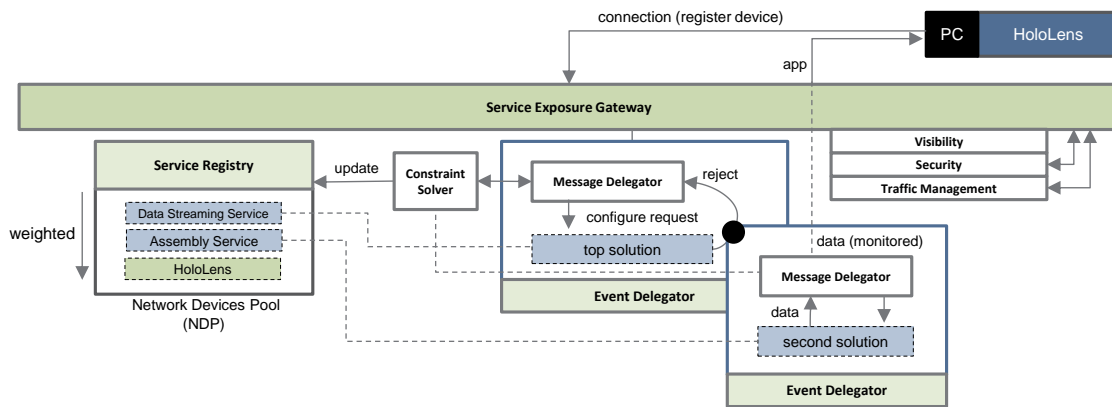


Figure 4. Self-configuration procedure for Microsoft HoloLens.

As our test system was designed with a computational node which would be able to act, the node accepts the request and exchanges a list of assembly instructions based on the hardware information and user. Because no digital assembly manual was specifically requested, the service sent the definition of the menu object containing the list of digital assembly manuals available and a list of event triggers defining the interaction techniques appropriate to its content. Upon receiving a user consent, the HoloLens loads the application and updates its metadata to reflect that which is active. Additional communications required by the application are processed directly by the workflow service and handled by the same computational node until a communication rejection is received.

Afterwards, we activate the first projector. The configuration workflow systemically identical to that of the Microsoft HoloLens, ending up with the “assembly service” sending an AR assembly

manual with no instructions to the projector (see Figure 5). The list of applications is not exchanged with the projector because it did not declare possible means of interacting with them in its capabilities.

As per our design, the assembly service is not restricted to a workspace, user or device. The digital manual was designed to provide instructions to both 3D and 2D devices. However, for our projectors, the location of digitally-projected artifacts was manually defined for their given perspectives (projection mapping to the workspace surroundings). Analogous to the previous scenario, once the projector-controller device receives the AR data, it renders its first instruction with no visual elements and notifies the network about this new active process.

In a last step, we plugged into the network the following devices: RGB-D cameras, a second projector, and a UR. The only devices our middleware automatically paired were the RGB-D cameras and the two projectors that were visible to each other.

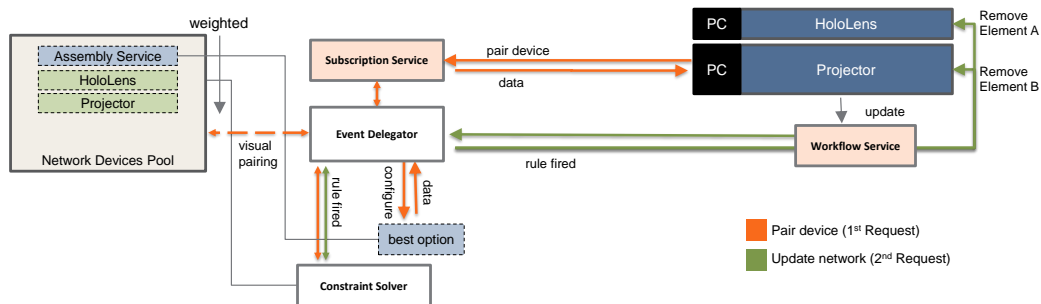


Figure 5. Self-configuration procedure for the first projector.

4.3. End-User Process

We applied our architecture in a specific use case related to the assembly of an electrical cabinet. In the first step, the user wears the HoloLens and from a provided application menu selects the application “Assemble Electric Cabinet”. A system of cameras has a field of view encompassing the workspace. When the physical body of the electric cabinet is detected in the field of view, the application projects’ assembly instructions onto it (see Figure 6).

Afterwards, the user drags the hyperlink interface from the location of the RGB-D camera (detected by HoloLens using computer vision and the tracking model specified in its capabilities) to the table where content is projected and where the electric cabinet will be placed. Because of this, the middleware detects a connection between two devices that can communicate: the camera and the projector. Upon recognizing this connection, it updates the registered capabilities of the projector with those of the camera. Upon the registry of capabilities changing, the projector requests a new application feature from the assembly service (which it receives): A menu with the list of functionalities that can be selected by touching the projected area.

The user drags the hyperlink of HoloLens to the table, causing the assembly instructions to be shared with the projector (see Figure 6). The representation rendered by the projector relies on the projection mapping defined by the digital assembly manual. The user is now capable of interacting on the projected surface with digital elements and on HoloLens with holograms in a parallel and synchronized manner. Moreover, these digital elements are constructed from the same virtually simulated entity for both systems. Additionally, within our test, the users were able to explore data from the robot and sensors (see Figure 3).

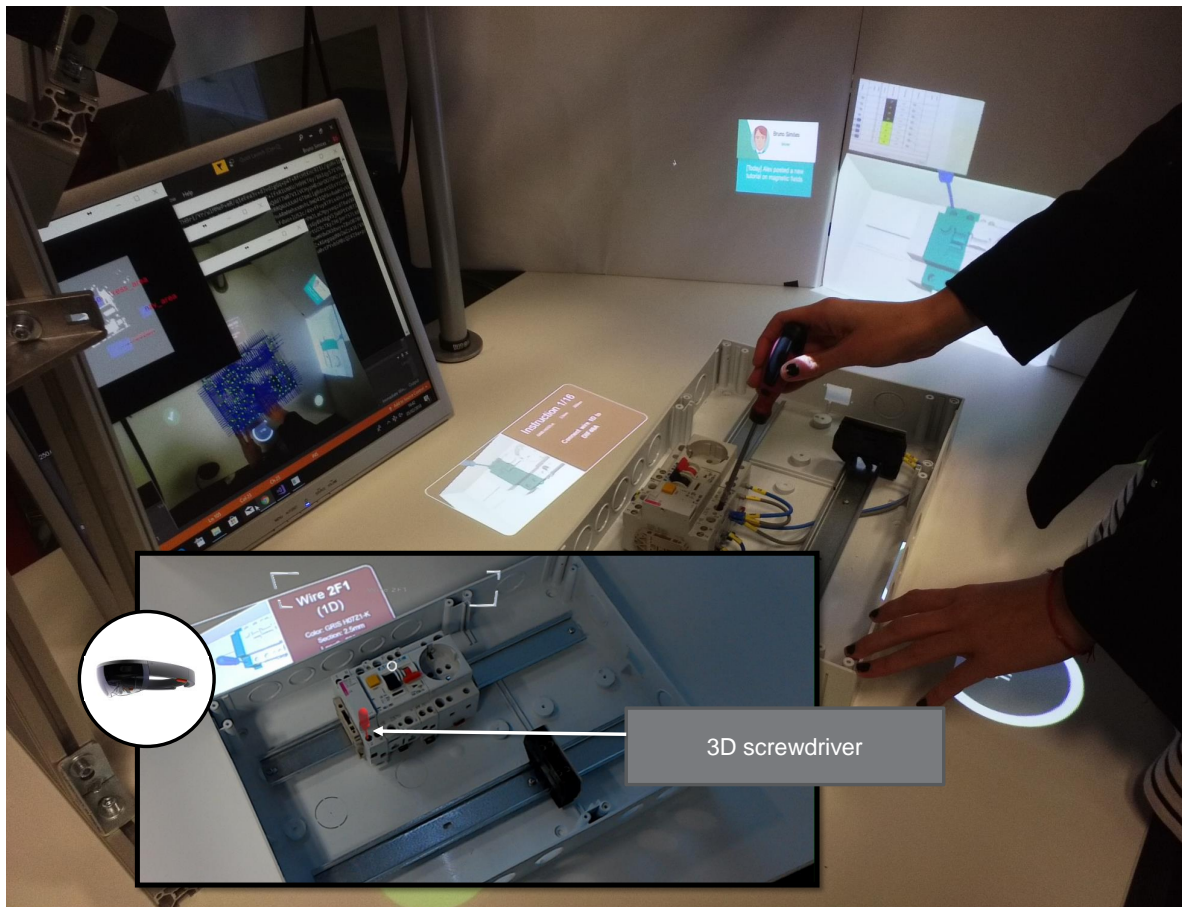


Figure 6. Multimodal use case.

5. Discussion

The primary objective of this validation is to test our hypothetical method for generating seamless interaction with different devices. To handle user interaction, we used a method called visual orchestration [39]. We did not evaluate the usability of the graphical elements representing our hyperlinks. To minimize the impact this would have on our use case, we kept the visual hyperlink representation simple: simple cubes that could be moved into visual surfaces defined as docks. We focused on the background communication concept that enables new ways of interaction with devices. Future work will explore if the proposed representations are suitable for the concept presented.

In addition to this pursuit, we applied a spatial perception algorithm to detect whenever either a physical or a digital object (holograms) collides with a virtual hyperlink. In future test scenarios, hyperlinks could be attached to printed images or objects without computational capabilities. These could be used to manipulate relations between different datasets.

The seamless interaction between cameras and projected content is remarkable. The middleware's calibration process requires an initial visual calibration process between the two devices (i.e., the device controlling the projector and the device processing images from the camera). Although the middleware could provide support for non-planar surfaces, the calibration procedure would be difficult to automate without the help of a human to move the calibration chess-board in 3D physical space. When thinking about future workspaces that integrate robots, the calibration task could be supported by a computational node that could control robots capable of moving the chessboard. Hence, the current implementation is limited to interaction in planar surfaces, but this is not an inherent property of the proposed architecture.

In the proposed work, the movement of physical data representations and holograms that represent data triggers a change of representation in data based on the capabilities of the target device. This allows the user to quickly explore different data perspectives based on different hardware available.

6. Conclusions

In this manuscript, we explore the advantages of spatial augmented reality technology in terms of enhancing cognition by using new forms of data interaction. We describe a new form of interacting with peripherals and industrial devices that does not require the design of applications for specific technologies. Instead, our interactions are designed to be highly abstract. The objective of our work is to propose a middleware that enables seamless access to data (e.g., produced by devices) across different physically-located devices. Overall, this cross-platform support enables us to develop intuitive interaction techniques for data and objects and also enables multiple users to interact with the same data using different technologies. Unlike previous propositions, the interactions supported by our middleware go beyond individual devices, extending to the space between them (providing the network with spatial awareness). It specializes in versatile support for the use of projectors and mixed reality devices to explore and interact with data and physical devices. We also provide a video to describe this paper, which is available here: <https://drive.google.com/open?id=1rJopJb-sjIOKafLLOYO1nnvFwey9y2oy>.

Author Contributions: B.S. and I.B. devised the project, the main conceptual ideas, and technological outlines. B.S. worked out the technical details and carried out the implementation of the research. B.S. and R.D.A. designed the system architecture and contributed to the writing and structure of the manuscript. All authors discussed and revised the results presented in the manuscript.

Funding: This research was funded by the European Union's Horizon 2020 research and innovation programme under the grant agreement no 723711.

Acknowledgments: The authors would like to thank Eric Prather student of Oregon State University's College of Engineering for his support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dash, S.K.; Mohapatra, S.; Pattnaik, P.K. A survey on applications of wireless sensor network using cloud computing. *Int. J. Comput. Sci. Emerg. Technol.* **2010**, *1*, 50–55.
2. Parwekar, P. From Internet of Things towards cloud of things. In Proceedings of the 2011 2nd International Conference on Computer and Communication Technology (ICCCCT-2011), Allahabad, India, 15–17 September 2011; pp. 329–333.
3. Card, S.K.; Mackinlay, J.D.; Shneiderman, B. *Readings in Information Visualization: Using Vision to Think*; Morgan Kaufmann: San Francisco, CA, USA, January 1999.
4. Cook, K.A.; Thomas, J.J. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*; Technical Report; Pacific Northwest National Laboratory (PNNL): Richland, WA, USA, 2005.
5. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 17 August 2012; pp. 13–16.
6. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [[CrossRef](#)]
7. Chandler, T.; Cordeil, M.; Czauderna, T.; Dwyer, T.; Glowacki, J.; Goncu, C.; Klapperstueck, M.; Klein, K.; Marriott, K.; Schreiber, F.; et al. Immersive analytics. In Proceedings of the Big Data Visual Analytics (BDVA), Hobart, Australia, 22–25 September 2015; pp. 1–8.
8. Coleman, B. Using Sensor Inputs to Affect Virtual and Real Environments. *IEEE Pervasive Comput.* **2009**, *8*, 16–23. doi:10.1109/MPRV.2009.60. [[CrossRef](#)]
9. Mann, S.; Furness, T.; Yuan, Y.; Iorio, J.; Wang, Z. All Reality: Virtual, Augmented, Mixed (X), Mediated (X, Y), and Multimediated Reality. *arXiv* **2018**, arXiv:1804.08386.

10. Modbus, I. Modbus Application Protocol Specification v1. 1a. North Grafton, Massachusetts. 2004. Available online: www.modbus.org/specs.php (accessed on 14 June 2018).
11. OPC Foundation. OPC Unified Architecture Specification. 2018. Available online: opcfoundation.org (accessed on 14 June 2018).
12. Slay, H.; Phillips, M.; Vernik, R.; Thomas, B. Interaction modes for augmented reality visualization. In *APVis '01 Proceedings of the 2001 Asia-Pacific Symposium on Information Visualisation*; Australian Computer Society, Inc.: Sydney, Australia, 2001; Volume 9, pp. 71–75.
13. Langlotz, T.; Nguyen, T.; Schmalstieg, D.; Grasset, R. Next-generation augmented reality browsers: Rich, seamless, and adaptive. *Proc. IEEE* **2014**, *102*, 155–169. [[CrossRef](#)]
14. Bell, B.; Höllerer, T.; Feiner, S. An annotated situation-awareness aid for augmented reality. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology*, Paris, France, 27–30 October 2002; pp. 213–216.
15. Feiner, S.; MacIntyre, B.; Hollerer, T.; Webster, A. A touring machine: Prototyping 3D mobile augmented reality systems for exploring the urban environment. In *Proceedings of the First International Symposium on Wearable Computers*, Cambridge, MA, USA, 13–14 October 1997; pp. 74–81.
16. Höllerer, T.; Feiner, S. Mobile augmented reality. In *Telegeoinformatics: Location-Based Computing and Services*; Taylor & Francis Books: Haarlem, The Netherlands, 2004; pp. 221–256.
17. Drascic, D.; Milgram, P. Perceptual issues in augmented reality. In *Proceedings of the Electronic Imaging: Science & Technology*, San Jose, CA, USA, 28 January–2 February 1996; pp. 123–134.
18. Kruijff, E.; Swan, J.E.; Feiner, S. Perceptual issues in augmented reality revisited. In *Proceedings of the 2010 9th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Seoul, Korea, 13–16 October 2010; pp. 3–12.
19. Pirolli, P.; Card, S. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of the International Conference on Intelligence Analysis*, McLean, VA, USA, May 2005; Volume 5, pp. 2–4.
20. Mayer, S.; Sörös, G. User Interface Beaming–Seamless Interaction with Smart Things Using Personal Wearable Computers. In *Proceedings of the 2014 11th International Conference on Wearable and Implantable Body Sensor Networks Workshops (BSN Workshops)*, Zurich, Switzerland, 16–19 June 2014; pp. 46–49.
21. Rekimoto, J.; Saitoh, M. Augmented surfaces: A spatially continuous work space for hybrid computing environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Pittsburgh, PA, USA, 15–20 May 1999; pp. 378–385.
22. Nebeling, M.; Teunissen, E.; Husmann, M.; Norrie, M.C. XDKinect: Development Framework for Cross-device Interaction Using Kinect. In *Proceedings of the 2014 ACM SIGCHI Symposium on Engineering Interactive Computing Systems*, Rome, Italy, 17–20 June 2014; pp. 65–74. doi:10.1145/2607023.2607024. [[CrossRef](#)]
23. Fernandes, L.; Nunes, R.R.; Matos, G.; Azevedo, D.; Pedrosa, D.; Morgado, L.; Paredes, H.; Barbosa, L.; Fonseca, B.; Martins, P.; et al. Bringing user experience empirical data to gesture-control and somatic interaction in virtual reality videogames: An exploratory study with a multimodal interaction prototype. In *Proceedings of the SciTecIn15—Conferência Ciências e Tecnologias da Interação*, Coimbra, Portugal, 12–13 November 2015.
24. Schwede, C.; Hermann, T. HoloR: Interactive mixed-reality rooms. In *Proceedings of the 2015 6th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, Gyor, Hungary, 19–21 October 2015; pp. 517–522.
25. Seyed, T.; Azazi, A.; Chan, E.; Wang, Y.; Maurer, F. SoD-toolkit: A toolkit for interactively prototyping and developing multi-sensor, multi-device environments. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces*, Madeira, Portugal, 15–18 November 2015; pp. 171–180.
26. Satyavolu, S.; Bruder, G.; Willemsen, P.; Steinicke, F. Analysis of IR-based virtual reality tracking using multiple Kinects. In *Proceedings of the 2012 IEEE Virtual Reality Short Papers and Posters (VRW)*, Costa Mesa, CA, USA, 4–8 March 2012; pp. 149–150.
27. Reitmayr, G.; Schmalstieg, D. Opentracker—an open software architecture for reconfigurable tracking based on xml. In *Proceedings of the IEEE Virtual Reality*, Yokohama, Japan, 13–17 March 2001; pp. 285–286.
28. Ong, S.; Yuan, M.; Nee, A. Augmented reality applications in manufacturing: A survey. *Int. J. Prod. Res.* **2008**, *46*, 2707–2742. [[CrossRef](#)]

29. Wang, X.; Ong, S.; Nee, A. A comprehensive survey of ubiquitous manufacturing research. *Int. J. Prod. Res.* **2018**, *56*, 604–628. [[CrossRef](#)]
30. Fite-Georgel, P. Is there a reality in industrial augmented reality? In Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Basel, Switzerland, 26–29 October 2011; pp. 201–210.
31. Leu, M.C.; ElMaraghy, H.A.; Nee, A.Y.; Ong, S.K.; Lanzetta, M.; Putz, M.; Zhu, W.; Bernard, A. CAD model based virtual assembly simulation, planning and training. *CIRP Ann. Manuf. Technol.* **2013**, *62*, 799–822. [[CrossRef](#)]
32. Nee, A.Y.; Ong, S.; Chryssolouris, G.; Mourtzis, D. Augmented reality applications in design and manufacturing. *CIRP Ann. Manuf. Technol.* **2012**, *61*, 657–679. [[CrossRef](#)]
33. Wang, X.; Ong, S.K.; Nee, A.Y.C. A comprehensive survey of augmented reality assembly research. *Adv. Manuf.* **2016**, *4*, 1–22. doi:10.1007/s40436-015-0131-4. [[CrossRef](#)]
34. García, A.; Arbelaz, A.; Franco, J.; Oregui, X.; Simões, B.; Etxegoien, Z.; Bilbao, A. Technologies for Industry 4.0 Data Solutions. In *Technological Developments in Industry 4.0 for Business Applications*; IGI Global, Montclair State University: Montclair, NJ, USA, 2018; p. 71.
35. Technologies, H. HAProxy—The Reliable, High Performance TCP/HTTP Load Balancer. 2018. Available online: www.haproxy.org (accessed on 14 June 2018).
36. HashiCorp. Consul. 2018. Available online: www.consul.io (accessed on 14 June 2018).
37. Weave. Service Registry Bridge for Docker. 2018. Available online: github.com/gliderlabs/registry (accessed on 14 June 2018).
38. Zhang, Z. Flexible camera calibration by viewing a plane from unknown orientations. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–27 September 1999; Volume 1, pp. 666–673.
39. Simoes, B.; Conti, G.; Piffer, S.; de Amicis, R. Enterprise-level architecture for interactive web-based 3D visualization of geo-referenced repositories. In Proceedings of the 14th International Conference on 3D Web Technology, Darmstadt, Germany, 16–17 June 2009; pp. 147–154.



© 2018 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).