Preparing the HoloLens for user Studies: an Augmented Reality Interface for the Spatial Adjustment of Holographic Objects in 3D Indoor Environments

Julian Keil¹ · Dennis Edler¹ · Frank Dickmann¹

Received: 6 June 2019 / Accepted: 12 July 2019 / Published online: 25 July 2019
© The Author(s) 2019

Abstract

Augmented reality (AR), the extension of the real physical world with holographic objects provides numerous ways to influence how people perceive and interact with geographic space. Such holographic elements may for example improve orientation, navigation, and the mental representations of space generated through interaction with the environment. As AR hardware is still in an early development stage, scientific investigations of the effects of holographic elements on spatial knowledge and perception are fundamental for the development of user-oriented AR applications. However, accurate and replicable positioning of holograms in real world space, a highly relevant precondition for standardized scientific experiments on spatial cognition, is still an issue to be resolved. In this paper, we specify technical causes for this limitation. Subsequently, we describe the development of a Unity-based AR interface capable of adding, selecting, placing and removing holograms. The capability to quickly reposition holograms compensates for the lack of hologram stability and enables the implementation of AR-based geospatial experiments and applications. To facilitate the implementation of other task-oriented AR interfaces, code examples are provided and commented.

Keywords Augmented reality · AR cartography · HoloLens · Spatial cognition · Experimental methods

Zusammenfassung

Augmented reality (AR), die Erweiterung der realen physischen Welt durch Hologramme, bietet viele Möglichkeiten die menschliche Wahrnehmung von, und Interaktion mit, dem geographischen Raum zu beeinflussen. Solche holografischen Elemente könnten beispielsweise zu einer Verbesserung von Orientierung, Navigation, oder durch Interaktion mit der Umgebung erzeugter mentaler räumlicher Modelle führen. Da sich AR Hardware noch in einem frühen Entwicklungsstadium befindet, sind wissenschaftliche Untersuchungen der Effekte von holografischen Elementen auf räumliches Wissen und die räumliche Wahrnehmung für die Entwicklung von nutzerorientierten AR Anwendungen notwendig. Die exakte und replizierbare Positionierung von Hologrammen, welche eine maßgebliche Voraussetzung für standardisierte wissenschaftliche Experimente ist, ist durch den Stand der Technik jedoch noch nicht gegeben. In diesem Artikel werden die technischen Ursachen hierfür erläutert. Im Anschluss beschreiben wir die Entwicklung eines *Unity*-basierten Interface, welches das Hinzufügen, Auswählen, Platzieren, und Entfernen von Hologrammen durch den Nutzer ermöglicht. Die Möglichkeit Hologramme schnell neu zu positionieren dient der Kompensation mangelnder räumlicher Stabilität von Hologrammen und ermöglicht die

✉ Julian Keil
julian.keil@rub.de

Dennis Edler
dennis.edler@rub.de

Frank Dickmann
frank.dickmann@rub.de

¹ Geomatics/Cartography Group, Geography Department,
Ruhr University Bochum, Bochum, Germany

Umsetzung AR-basierter räumlicher Experimente und Anwendungen. Codebeispiele sind beigelegt und kommentiert, um die Entwicklung weiterer aufgabenorientierter AR Interfaces zu fördern.

Schlüsselwörter Augmented Reality · AR-Kartographie · HoloLens · Raumkognition · Experimentelle Methoden

1 Introduction

Orientation and navigation are fundamental aspects of everyday life. Estimating distances and angles, recalling and recognizing object locations and memorizing routes or landmarks are just a few examples of many basic tasks in outdoor as well as indoor environments (e.g., Keil et al. 2019; Plumert et al. 2005; Postma and De Haan 1996). To optimize the effectiveness and efficiency of these tasks, people often use geospatial media, such as maps or map-like representations. The academic fields of cartography and spatial cognition have increasingly addressed the question of how to integrate and use modern technologies—usually originating in IT- and entertainment industries—to develop proper geospatial media (e.g., Edler et al. 2019; Hruby 2019; Knust and Buchroithner 2014).

Going beyond traditional 2D print approaches, the portfolio of cartographic products was strongly influenced by digital techniques since the establishment of the computer as a mass media device and, in addition, as a tool to create other mass media (e.g., Clarke et al. 2019; Taylor and Lauriault 2007; Müller 1997). Multimedia cartography—sometimes also referred to as “cybercartography” (Taylor 2005)—lead to fundamental approaches of computer-based animated, interactive and multisensory (web) map applications (e.g., Kraak 1999; Peterson 1995; Krygier 1994). It is argued that animation techniques used in cartography have been influenced by the computer and video game industries (Edler et al. 2018a; Edler and Dickmann 2017; Ahlqvist 2011; Corbett and Wade 2005).

The development of computer-based animation techniques came along with appropriate software and hardware solutions. Higher performances of software and hardware also allowed the further development of stable and detailed 3D visualization methods and techniques. For example, autostereoscopic displays allowed to generate 3D depth effects, which was explored in several studies on visualization and user experiments in cartography (e.g., Edler and Dickmann 2015; Bröhmer et al. 2013; Buchroithner 2007). Moreover, the open availability of game engines, such as *Unity* and *Unreal Engine*, supports the creation of individual 3D landscapes that can be accessed with virtual reality (VR) headsets, in real-time and from the ego perspective—thus, creating an impression of immersion. The potentials of VR-based visualization are currently under study (e.g., Cöltekin et al. 2019; Edler et al. 2018b; Hruby et al. 2019; Kersten et al. 2018).

Closely related to 3D visualization in VR are 3D visualizations in augmented reality (AR). AR techniques allow to project static or animated objects into real environments, thus extending real physical environments. Representing an early development stage, AR visualization techniques can be based on so called mid-air displays, sometimes also referred as free-space displays (Dickmann 2013). A mid-air displays projects graphical objects on free projection surfaces, such as a hardly visible wall of fog (“fog screen”) created by an installed blower (DiVerdi et al. 2008).

A famous example of an AR application interacting a lot with space is the gaming app “Pokémon GO” (Zhao and Chen 2017). In this smartphone- or tablet-based game, users interact with audiovisually animated game characters that can be found in the real environment. In this way, the whole logic and process of the game is added like an additional information layer into the physical landscape. The smartphone or tablet is the ‘physical gateway’ to this augmentation. The camera of the device is used to record the area in front of the user and the recordings are augmented with virtual objects in real time. As demonstrated by de Almeida Pereira et al. (2017), this technique can also be used to augment physical 2D maps with 3D geographic information as height maps.

Two head-mounted display (HMD) devices representing the state of the art of current AR are the *Microsoft HoloLens* (version 2 recently introduced) and the *HTC Vive Pro*. The *Microsoft HoloLens* (Fig. 1) uses a pair of *smart glasses*. Stereoscopic images (holograms) are projected onto two small lenses in front of the eyes (Noor 2016). As these lenses are see-through (Gruenfeld et al. 2017), the projected holograms merge with the real environment. The *HTC Vive Pro* (Fig. 1) is a VR headset with AR capability. The environment in front of the user can be recorded with two cameras inside the HMD and rendered on two displays located in front of the eyes of the user (HTC 2019). Additionally, the camera recordings can be augmented by artificial elements acting as stereoscopic holograms.

1.1 The Potential of AR Techniques for Experiencing Space

As both the *Microsoft HoloLens* and the *HTC Vive Pro* are capable of tracking head movements, they make it possible to create an impression of permanent presence of holographic geospatial objects. Even if the user walks around in a defined area, commonly indoor area, holograms remain

Fig. 1 The Microsoft HoloLens (top) and the HTC Vive Pro (bottom) represent the state of the art of augmented reality HMDs

and adopt to the user location and viewing perspective. This permanent and adaptable holographic projection may lead to visualization approaches that bring additional advantages for the cognitive processing of the geospatial area experienced.

Empirical research of cartography, spatial cognition and experimental psychology has recently led to some recommendations for the construction of user cognition-oriented cartographic media. For example, it was reported in user experiments that an additional layer of square grids increases the performances in memory of object locations (Bestgen et al. 2017; Kuchinke et al. 2016; Edler et al. 2014) and in estimating longer linear distances in maps (Dickmann et al. 2019). The grid-based memory effect also occurs if the grid structure is physically reduced to indicated (“illusory”) lines (Dickmann et al. 2017), gets a depth offset (Edler et al. 2015) or is changed to a hexagonal pattern (Edler et al. 2018c). Other studies reported that reducing the visibility of some map areas can direct visual attention towards other map areas (Keil et al. 2018), and that the display of landmarks can improve route knowledge (Ruddle et al. 2011) and orientation (Li et al. 2014).

The above mentioned cognition-based effects on spatial performance measures in maps are promising results indicating that an extended communication of spatial information can bring advantages for the map user in terms of map perception, orientation, navigation and the formation of spatial

knowledge. Similar effects will likely occur in real 3D environments augmented by holographic spatial objects. These holographic layers could offer an additional (geometrical) structure to support the cognitive processing of object locations, distance estimations and relative directions between objects.

To investigate possible effects for the perception of spatial information, new methodological challenges occur. The possibility to implement spatial models in AR applications has already been investigated and described (Wang et al., 2018). However, to take full advantage of the possibilities of AR for geospatial applications, technical limitations of the current available AR devices must be faced. These include the precise placement and stability of holograms in the three-dimensional space, a crucial quality criterion for AR applications (Harders et al. 2008). Having found stable solutions that guarantee a high spatial precision, the AR devices can become valuable methodological tools in geospatial experiments focused on fundamental questions of spatial cognition in 3D environments. In user studies, they could be used to project holographic objects in the environment. Moreover, AR devices could assist experimental investigators to arrange the spatial layout of movable real world objects used in their study. The projection of ‘virtual place markers’ can increase the precision of identical spatial object arrangements, which—from a methodological perspective—increases the comparability of acquired user data (between participants). Moreover, projected ‘virtual markers’ can support the analysis of user tasks, such as the identification/measurement of distortion errors, for example, in location memory tasks.

To exploit the possibilities of AR systems for geospatial user experiments, it is necessary to create technical methods to establish controlled procedures and to standardize the placement of holographic objects in a real 3D setting. In the following sections, we describe the functionality of current AR systems, how technical factors of these AR systems affect the targeted placement of holograms, and additional requirements for geospatial applications and experiments. To address and resolve the described discrepancy between requirements and limitations, we present a self-developed AR interface application capable of (re-)placing holograms highly accurately during runtime. Code examples are provided for transparency, a better understanding, and replicability.

2 Hologram Placement and Display

In many VR and monitor-based 3D applications, positions of 3D objects are ‘hard coded’, i.e., their positions are predefined and cannot be changed. The advantage is that all participants see exactly the same arrangement of visual stimuli,

which is often a highly relevant precondition for geospatial experiments. However, in AR-based applications, several technical limitations demand a more flexible approach for the placement of visual stimuli.

One of the greatest technical differences between the two most popular AR headsets, the *Microsoft HoloLens* (including version 2) and the *HTC Vive Pro*, lies in the tracking system used to register head movements and to adjust the displayed holograms. The *Microsoft HoloLens* uses an *inside-out* tracking system, i.e., the sensors used to track the position and rotation of the head are mounted to the HMD. Six cameras integrated in the headset scan the environment (Evans et al. 2017). The camera recordings are then used to build a spatial representation of all perceived objects (Liu et al. 2018). As this representation can be exported as a 3D model, the *HoloLens* can also be used to visualize 3D space (see Fig. 2).

Head movements are registered by matching objects recorded by the cameras in real-time to objects already represented in the spatial model. Calculating the relative position towards these objects then allows to triangulate the current head position and rotation inside the 3D space. The advantage of pure inside-out tracking is that no additional hardware is required. Given that the *Microsoft HoloLens* is a standalone device (Evans et al. 2017), people can walk freely and use the device seamlessly in different rooms or even floors. However, inside-out tracking also has some serious disadvantages concerning the placement of static holograms. First, image analysis, the precondition for image-based tracking, requires a lot of processing power (Liu et al. 2013). As the processing power of a standalone device is naturally limited, this leads to only moderate tracking accuracy and occasional tracking lags. Second, the cameras need to identify at least some reference objects within the spatial

range of the cameras, which according to our experience is approximately 5 m. Therefore, tracking lags regularly occur in large empty spaces, especially outdoors. Additionally, poor lighting conditions may negatively affect the capability to identify referenced objects (Loesch et al. 2015). The mentioned tracking lags often lead to a distorted or shifted internal coordinate system. In these occasions, the positions of all ‘hard-coded’ holograms relative to real world objects are also distorted or shifted and need to be readjusted. A third limitation of the *Microsoft HoloLens* concerning the placement of static holograms is rather code-based than tracking-based. Each time an application is started, the internal coordinate system is set relative to the position of the headset. As it is almost impossible to place the headset at the exact same position when an application is started, ‘hard coded’ static holograms cannot be placed reliably at the same real world position twice.

In contrast to the *Microsoft HoloLens*, the *HTC Vive Pro* uses a combination of inside-out and *outside-in* tracking system. In contrast to inside-out tracking, outside-in tracking uses stationary sensors located around the tracked space to register head movements. In the case of the *HTC Vive Pro*, two *SteamVR tracking* infrared base stations (previously called *Lighthouse*) are placed in opposite corners of a tracked space with a maximum diagonal size of 5 m (HTC 2019). These base stations interact with photo sensors built into the headset and hand controllers. By comparing the times different sensors perceive the signals of the base stations, the positions of the HMD and the controllers in 3D space can be triangulated. Additionally, inside-out tracking is used to generate a 3D model of the objects inside the tracked area. Similar to the *Microsoft HoloLens*, the 3D model is generated based on two cameras in the headset. Combining *inside-out* and *outside-in* tracking limits the

Fig. 2 Picture of a room (left side) and the 3D model of this room built by the *Microsoft HoloLens* based on camera recordings (right side). Notice how static objects (e.g., the doorframe and shelves) are represented very accurately whereas moveable objects (e.g., the mon-

itor) are somewhat bulky. This bulkiness represents the attempt of the *HoloLens* to account for the modified arrangement of spatial objects after the objects have been slightly repositioned

mobility of the device, as the tracking area is confined to the range of the base stations. However, this technique has clear advantages in terms of tracking accuracy. First, the use of infrared base stations provides a high tracking accuracy and precision (Ng et al. 2017). Second, the static positioning of base stations also allows for much more accurate resets of the internal coordinate system after short tracking losses or lags. Therefore, the positions of static holograms relative to real world objects need to be readjusted less often.

Readjustments of displayed holograms may not only be required based on the described technical limitations of AR systems. Some experimental designs or other AR applications may pose requirements to change the real world positions of holograms, or to add (*spawn*), select, or remove (*despawn*) holograms during runtime. For example, a small scale holographic visualization of a construction project (see example in Fig. 3) would benefit from the ability to easily modify building positions to visualize different architectural layouts. Although natural interaction like dragging objects with a controller is theoretically possible, this does not apply for the highly accurate and standardized placement of holograms required in scientific experiments used to investigate cognitive effects of holograms on the perception of geographic space. Additionally, the lack of a multi button controller for the Microsoft HoloLens demands other ways of controlling hologram display and interaction effectively. Therefore, we decided to develop a holographic AR interface application that allows spawning, selection, despawning and highly accurate (re-)positioning of holograms during runtime. As visibility of the developed interface may unintentionally affect the behavior of participants in scientific experiments, we also provided a function to reduce interface visibility. To

Fig. 3 Holographic representation of the old industrial landscape Zeche Holland in Bochum Wattenscheid displayed by the Microsoft HoloLens

support the development of other demand oriented AR interfaces, the workflow and functions of the developed interface are described in detail.

3 Implementing Interactivity of Holograms

We chose to develop the AR interface application with the *Unity* game engine (version 2018.3), as it supports both the *Microsoft HoloLens* and the *HTC Vive Pro*. The code was written in the *C#* programming language. The holographic interface consists of UI (user interface) elements provided by *Unity*, namely *Buttons* and *Texts*. Buttons are used to trigger actions by user input and Texts are used to provide information about the Buttons. The UI elements are arranged on a *Canvas*, a layer required to display the UI elements. Additionally, an *Event System* is used to send input from the user to the UI elements. Figure 4 illustrates a minimalistic UI hierarchy in *Unity*.

3.1 User Input

The mode of user input is determined by the used AR hardware. The *HTC Vive Pro* uses hand controllers that send a ray similar to a laser pointer. The user can use this ray to aim at a Button of the holographic interface and to interact with it by clicking on a trigger button on the hand controller (see Fig. 5). The *Microsoft HoloLens* uses an invisible ray that is casted forward from the front of the headset. When the invisible ray hits a hologram, a circular cursor is displayed at the hit point. To aim at a Button, the user needs to turn the headset towards the Button. In other words, the Button needs to be in the center of the field of view. The user can then interact with it using the *air tap* gesture, which includes lifting and then bending the index finger (see Fig. 5).

Fig. 4 Minimalistic example of a user interface in Unity. The button and text elements are set as children of the canvas. The automatically added event system handles user input

Fig. 5 The upper picture shows a HTC Vive Controller and the ray used to aim at UI buttons. The two bottom pictures demonstrate the air tap gesture used by the Microsoft HoloLens to interact with UI buttons

3.2 Spawning

As adding spawnable holograms or removing them should be as easy as possible, we created an empty array of spawnable *Prefabs*. Prefabs are *Unity* elements that act like a copy template. They are objects with defined characteristics, such

```
// spawn a new object
public void SpawnObject (GameObject objectToSpawn) {
    // spawn the object and add it to the list of spawned objects
    spawnedObjects.Add(Instantiate (objectToSpawn,
                                    new Vector3 (0, -0.5f, 3.5f),
                                    Quaternion.identity) as GameObject);
    // refresh the list of spawned object buttons
    RefreshButtons ();
}
```

Fig. 6 The function “SpawnObject” creates a new hologram. The spawn position is set relative to the position of the interface (0.5 m below and 3.5 m in front of the interface). Subsequently, the list of spawned buttons gets refreshed to add a button for the newly spawned hologram

```
// select a spawned object
public void SelectObject (GameObject selectedObject, int objectId) {
    // set the active object to the passed selected object
    activeObject = selectedObject;
}
```

Fig. 7 When a button of a spawned object is pressed, the associated object is defined as the active object

as shape, size or color. In this specific case, Prefabs are the spawnable holograms. The size of the Prefab array can be changed and Prefabs can be added or removed. When the application is started, a button is added to the AR interface for each element in the Prefab array (Fig. 12, area 1). Pressing one of these buttons creates an *instance* (a copy) of the associated Prefab and adds it to the list of spawned objects (see Fig. 6). By pressing the same button again, additional instances of the same hologram Prefab can be spawned. For each spawned hologram, a button is added to the list of spawned object buttons (Fig. 12, area 2).

3.3 Selection

To select a spawned hologram, the user has to interact with the button representing the spawned object (Fig. 12, area 2). The button then invokes a function that defines the associated hologram as the active object (Fig. 7), which can then be despawned, moved or rotated. The associated button of the selected hologram is highlighted with a red background.

3.4 Despawning

Once a hologram has been selected, pressing the despawn button (Fig. 12, area 3) despawns the object using the *Unity* function “*Destroy()*” and removes it from the list of spawned objects (Fig. 8). Simultaneously, all spawned object buttons are removed and a new button is added for each element in the updated list of spawned objects (Fig. 12, area 2).

```

// despawn a spawned object
public void DespawnObject () {
    // remove the buttons of all spawned objects
    RemoveDynamicButtons ();
    // remove the currently selected object from the list of spawned objects
    spawnedObjects.Remove (activeObject);
    // despawn the currently selected object
    Destroy (activeObject);
    // add a button for each object in the list of spawned objects
    RefreshButtons ();
}

```

Fig. 8 The function “DespawnObject” despawns the currently active object, removes the reference to this object from the list of spawned objects and removes the associated object button

3.5 Placement

To enable movement and rotation of spawned holograms, eight buttons were added to the interface (Fig. 12, area 4). Six buttons allow positive or negative movement along the three spatial axes. Two buttons allow clockwise or counter-clockwise rotation. Pressing one of these buttons moves or rotates the currently selected hologram by a value defined in a scale variable (see Fig. 9).

As the required accuracy and the distance between the original and desired location may vary, four buttons used to adjust the scale variable were added (Fig. 12, area 5). Pressing these buttons allows to set the scale to 1 mm, 1 cm, 10 cm or 1 m (see Fig. 10). The corresponding button of the currently selected scale is highlighted with a red background. Switching the scale enables quick movement of holograms across long distances, as well as accurate placement in the millimeter range.

3.6 Reducing Interface Visibility

A relevant precondition for the ability to use the developed interface for scientific experiments is the possibility to optionally reduce its visibility, as it could distract participants’ attention and thereby affect the experimental results. Therefore, we implemented a Button that can toggle the visibility of all other Buttons and the Texts on or off (Figs. 11 and 12, area 6). The invoked function checks whether all other UI elements are currently visible (“*UiVisible*”). Then it loops through all static UI elements (“*objectsToToggle*”) and the dynamically generated Buttons of the spawnable and spawned objects (Fig. 12, area 1 and 2) and either hides or unhides them using the *Unity* function “*SetActive()*”.

```

// change the x position of the selected object
public void setX (int direction){
    // if an object has been selected
    if (activeObject != null) {
        // move the selected object on the x axis in the passed direction
        activeObject.transform.position = new Vector3 (
            activeObject.transform.position.x + (scale * direction),
            activeObject.transform.position.y,
            activeObject.transform.position.z);
    }
    // if no object has been selected yet
} else {
    // return a warning
    Debug.Log ("No Object selected yet.");
}
}

```

Fig. 9 Pressing the left or right arrow in the interface (see Fig. 12, area 4) invokes the function “setX”. This function checks if an object has been selected and, if this is the case, moves the object

along the x axis. The scale variable defines the distance that the object is moved along the x axis


```

// set the scale for object replacement
public void SetScale (float newScale, int buttonId){
    // set the scale to the passed scale value
    scale = newScale;
    // loop through the scale button list
    for (int i = 0; i < 4; i++) {
        // set the color of the currently selected button to white
        scaleButtons[i].GetComponent<Image>().color = Color.white;
    }
    // set the scale of the passed (pressed) button to red
    scaleButtons[buttonId].GetComponent<Image>().color = Color.red;
}

```

Fig. 10 The function “SetScale” changes the length by which objects are moved in one step. Additionally, it highlights the button linked to the currently selected scale value

```

// hide/show the UI
public void toggleUi () {
    // if the UI is currently not hidden
    if (UiVisible) {
        // loop through the static UI elements
        for (int i = 0; i < objectsToToggle.Length; i++) {
            // deactivate visibility of the current UI element
            objectsToToggle [i].SetActive (false);
        }
        // loop through the buttons of currently spawned objects
        for (int i = 0; i < spawnedObjectsButtons.Count; i++) {
            // deactivate visibility of the current button
            spawnedObjectsButtons[i].SetActive (false);
        }
        // loop through the buttons of the spawnable objects
        for (int i = 0; i < spawnableObjectsButtons.Count; i++) {
            // deactivate visibility of the current button
            spawnableObjectsButtons[i].SetActive (false);
        }
        // set the UI control variable to 'hidden'
        UiVisible = false;
    }
    // if the UI is currently hidden
    } else {
        // loop through the static UI elements
        for (int i = 0; i < objectsToToggle.Length; i++) {
            // activate visibility of the current UI element
            objectsToToggle [i].SetActive (true);
        }
        // loop through the buttons of currently spawned objects
        for (int i = 0; i < spawnedObjectsButtons.Count; i++) {
            // activate visibility of the current button
            spawnedObjectsButtons[i].SetActive (true);
        }
        // loop through the buttons of the spawnable objects
        for (int i = 0; i < spawnableObjectsButtons.Count; i++) {
            // activate visibility of the current button
            spawnableObjectsButtons[i].SetActive (true);
        }
        // set the UI control variable to 'visible'
        UiVisible = true;
    }
}

```

Fig. 11 The function “toggleUi” either hides or displays all UI elements except the Button used to invoke the function

Fig. 12 Overview of the built AR interface. The green rectangles and numbers are inserted to visualize the different areas and are not part of the actual interface. Area one contains a Button for each spawnable object. Pressing one of the Buttons spawns an instance of the associated spawnable object. For each spawned object, an associated Button is added to area two. These are used to select spawned objects.

The Button in area three is used to despawn the selected object. With the arrow Buttons in area four, the selected object can be moved and rotated. The Buttons in area five set the scale for the object positioning. Pressing the Button in area six toggles the visibility of all other UI elements on or off

4 Summary

Despite the massive leaps forward during the last years, AR hardware has not yet been developed far enough to be classified as a consumer-friendly technology. In terms of Gartner's Hype Cycle Approach (Jarvenpaa and Makinen 2008), one could say that AR passed the peak of inflated expectations. Initial ideas for AR applications have been confronted with technical limitations still to be resolved, as localization of the HMD position, processing power of untethered devices, and hologram placement. In this paper, the last mentioned limitation has been addressed. In the context of geospatial experiments and applications, we specified the requirements for hologram positioning and display. These include precise and standardized hologram placement and repositioning of holograms in the real world. The presented AR interface (Fig. 13) addresses these requirements and contributes to an optimized experimental user testing in a real 3D spatial layout. The interface is specifically designed for user-studies that focus on the cognitive processing of 3D spatial arrangements, such as object locations, distances and relative directions. Additionally, the presented solution can act as a template for the development of other task-oriented AR interfaces for geographic and cartographic applications in *Unity*. Application examples would be to display holographic models of 3D environments with adjustable scale or selectively overlaying 3D height maps with additional

Fig. 13 The finalized AR interface captured with the screenshot function of the Microsoft HoloLens

information as population density (cf. de Almeida Pereira et al. 2017) or annual precipitation.

Furthermore, we illustrated which technical characteristics of current AR devices are in conflict with the identified requirements. Especially the stability of holograms was argued to be affected by tracking issues of current AR headsets. As a workaround, we described the development process of an AR interface capable of adding, removing and

placing holograms precisely in real world space. This interface allows to perform standardized scientific experiments using AR hardware by correcting false hologram positions manually. To reduce interference with experimental visual stimuli, visibility of the AR interface can be reduced to a minimum when it is currently not required. However, our proposed solution addresses only some limitations of current AR devices. The most crucial limitation, the incapability to use current AR devices in large scale and outdoor environments, still remains. As long as highly accurate and reliable tracking cannot be provided by AR hardware (e.g., realized by a combination of inside-out and satellite tracking), the use of AR devices will be limited to spatially confined environments.

Acknowledgements This study was supported by grants of the Deutsche Forschungsgemeinschaft (DFG) to FD (DI 771/11-1; Project 218150251). We would like to thank Melvin Sossna for supporting the development of the 3D model presented in Fig. 3.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

- Ahlqvist O (2011) Converging themes in cartography and computer games. *Cartogr Geogr Inf Sci* 38(3):278–285. <https://doi.org/10.1559/15230406382278>
- Bestgen AK, Edler D, Müller C, Schulze P, Dickmann F, Kuchinke L (2017) Where is it (in the map)? Recall and recognition of spatial information. *Cartographica* 52(1):80–97. <https://doi.org/10.3138/cart.52.1.3636>
- Bröhm K, Knust C, Dickmann F, Buchroithner MF (2013) Z-Axis based visualization of map elements—cartographic experiences with 3D monitors using lenticular foil technology. *Cartogr J* 50:211–217
- Buchroithner MF (2007) Echtdreidimensionalität in der Kartographie: Gestern, heute und morgen. *Kartogr Nachrichten* 57(5):239–248
- Clarke KC, Johnson MJ, Trainor T (2019) Contemporary American cartographic research: a review and prospective. *Cartogr Geogr Inf Sci* 46(3):196–209. <https://doi.org/10.1080/15230406.2019.1571441>
- Cöltekin A, Oprean D, Wallgrün JO, Klippel UA (2019) Where are we now? Re-visiting the digital earth through human-centered virtual and augmented reality geovisualization environments. *Int J Digit Earth* 12(2):119–122. <https://doi.org/10.1080/17538947.2018.1560986>
- Corbett J, Wade K (2005) Player perspective: using computer game engines for 3D cartography. *Cartographica* 40(3):113–120. <https://doi.org/10.3138/8443-505Q-M8T1-1774>
- de Almeida Pereira GH, Stock K, Stamato Delazari L, Centeno JAS (2017) Augmented reality and maps: new possibilities for engaging with geographic data. *Cartogr J* 54(4):313–321. <https://doi.org/10.1080/00087041.2017.1411417>
- Dickmann F (2013) Freiraum-displays—Ein neues Medium für die Kartographie? *Kartogr Nachrichten* 63(2/3):89–92
- Dickmann F, Edler D, Bestgen AK, Kuchinke L (2017) Exploiting illusory grid lines for object-location memory performance in urban topographic maps. *Cartogr J* 54(3):242–253. <https://doi.org/10.1080/00087041.2016.1236509>
- Dickmann F, Keil J, Kuner J, Edler D (2019) Quadratische Gitterzellen in Topographischen Karten erhöhen die Genauigkeit von Distanzschätzungen. *KN J Cartogr Geogr Inf* 69(2):109–120. <https://doi.org/10.1007/s42489-019-00014-2>
- DiVerdi S, Olwal A, Rakkolainen I, Höllerer T (2008) An immaterial pseudo-3D display system with 3D interaction. In: Ozaktas HM, Onural L (eds) *Three-dimensional television: capture, transmission, and display*. Springer, Heidelberg, pp 505–528
- Edler D, Dickmann F (2015) Elevating streets in urban topographic maps improves the speed of map-reading. *Cartographica* 50(4):217–223. <https://doi.org/10.3138/cart.50.4.3131>
- Edler D, Dickmann F (2017) The impact of 1980s and 1990s video games on multimedia cartography. *Cartographica* 52(2):168–177. <https://doi.org/10.3138/cart.52.2.3823>
- Edler D, Bestgen AK, Kuchinke L, Dickmann F (2014) Grids in topographic maps reduce distortions in the recall of learned object locations. *PLoS ONE* 9(5):e98148. <https://doi.org/10.1371/journal.pone.0098148>
- Edler D, Bestgen AK, Kuchinke L, Dickmann F (2015) True-3D accentuating of grids and streets in urban topographic maps enhances human object location memory. *PLoS ONE* 10(2):e0116959. <https://doi.org/10.1371/journal.pone.0116959>
- Edler D, Husar A, Keil J, Vetter M, Dickmann F (2018a) Virtual reality (VR) and open source software: a workflow for constructing an interactive cartographic VR environment to explore urban landscapes. *Kartogr Nachrichten* 68(1):3–11
- Edler D, Keil J, Bestgen AK, Kuchinke L, Dickmann F (2018b) Hexagonal map grids—an experimental study on the performance in memory of object locations. *Cartogr Geogr Inf Sci*. <https://doi.org/10.1080/15230406.2018.1496035>
- Edler D, Keil J, Dickmann F (2018c) Varianten interaktiver Karten in Video- und Computerspielen - eine Übersicht. *Kartogr Nachrichten* 68(2):57–65
- Edler D, Kühne O, Keil J, Dickmann F (2019) Audiovisual cartography: established and new multimedia approaches to represent soundscapes. *KN J Cartogr Geogr Inf*. <https://doi.org/10.1007/s42489-019-00004-4>
- Evans G, Miller J, Pena MI, MacAllister A, Winer EH (2017) Evaluating the Microsoft HoloLens through an augmented reality assembly application. In: Sanders-Reed JN and Arthur JJ (eds) *Degraded environments: sensing, processing, and display*. SPIE, Bellingham, Washington: Proceedings of SPIE 10197. <https://doi.org/10.1117/12.2262626>
- Gruenefeld U, Hsiao D, Heuten W, Boll S (2017) EyeSee: beyond reality with Microsoft HoloLens. In: Simeone AL (ed) *Proceedings of the 5th symposium on spatial user interaction*. ACM, New York, p 148. <https://doi.org/10.1145/3131277.3134362>
- Harders M, Bianchi G, Knoerlein B, Székely G (2008) Calibration, registration, and synchronization for high precision augmented reality haptics. *IEEE Trans Vis Comput Graph* 15(1):138–149. <https://doi.org/10.1109/TVCG.2008.63>
- Hruby F (2019) The sound of being there—audiovisual cartography with immersive virtual environments. *KN J Cartogr Geogr Inf*. <https://doi.org/10.1007/s42489-019-00003-5>

- Hruby F, Ressler R, de la Borbolla del Valle G (2019) Geovisualization with immersive virtual environments in theory and practice. *Int J Digit Earth* 12(2):123–136
- HTC (2019) VIVE Pro HMD support. <https://www.vive.com/us/support/vive-pro-hmd/>. Accessed 12 Mar 2019
- Jarvenpää HM, Makinen SJ (2008) An empirical study of the existence of the Hype Cycle: a case of DVD technology. *IEEE international engineering management conference*. IEEE, Piscataway, pp 1–5. <https://doi.org/10.1109/IEMCE.2008.4617999>
- Keil J, Mocnik FB, Edler D, Dickmann F, Kuchinke L (2018) Reduction of map information regulates visual attention without affecting route recognition performance. *ISPRS Int J Geoinf* 7(12):1–13. <https://doi.org/10.3390/ijgi7120469>
- Keil J, Edler D, Dickmann F, Kuchinke L (2019) Meaningfulness of landmark pictograms reduces visual salience and recognition performance. *Appl Ergon* 75:214–220. <https://doi.org/10.1016/j.apergo.2018.10.008>
- Kersten T, Deggim S, Tschirschwitz F, Lindstaedt MU, Hinrichsen N (2018) Segeberg 1600—Eine Stadtrekonstruktion in virtual reality. *Kartogr Nachrichten* 68(4):183–191
- Knaust C, Buchroithner MF (2014) Principles and terminology of true-3D geovisualisation. *Cartogr J* 51(3):191–202. <https://doi.org/10.1179/1743277413Y.0000000038>
- Kraak MJ (1999) Cartography and the use of animation. In: Cartwright W, Peterson MP, Gartner G (eds) *Multimedia cartography*. Springer, Berlin, pp 317–326
- Krygier JB (1994) Sound and geographic visualization. In: Taylor DRF, MacEachren AM (eds) *Visualization in modern cartography*. Pergamon, Oxford, pp 149–166
- Kuchinke L, Dickmann F, Edler D, Bordewieck M, Bestgen AK (2016) The processing and integration of map elements during a recognition memory task is mirrored in eye-movement patterns. *J Environ Psychol* 47:213–222. <https://doi.org/10.1016/j.jenvp.2016.07.002>
- Li R, Korda A, Radtke M, Schwering A (2014) Visualising distant off-screen landmarks on mobile devices to support spatial orientation. *J Locat Based Serv* 8(3):166–178. <https://doi.org/10.1080/17489725.2014.978825>
- Liu F, Shu P, Jin H, Ding L, Yu J, Niu D, Li B (2013) Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications. *IEEE Wirel Commun* 20(3):14–22. <https://doi.org/10.1109/MWC.2013.6549279>
- Liu Y, Dong H, Zhang L, Saddik AE (2018) Technical evaluation of HoloLens for multimedia: a first look. *IEEE Multimed* 25(4):8–18. <https://doi.org/10.1109/MMUL.2018.2873473>
- Loesch B, Christen M, Wüest R, Nebiker S (2015) Geospatial augmented reality—Lösungsansätze mit natürlichen Markern für die Kartographie und die Geoinformationsvisualisierung im Außenraum. In: Kersten TP (ed) *Publikationen der Deutschen Gesellschaft für Photogrammetrie, Fernerkundung und Geoinformation e.V. DGPF, Münster*, pp 89–97
- Müller JC (1997) GIS, Multimedia und die Zukunft der Kartographie. *Kartogr Nachrichten* 47(2):41–51
- Ng AKT, Chan LKY, Lau HYK (2017) A low-cost lighthouse-based virtual reality head tracking system. In: *International conference on 3D immersion (IC3D)*. IEEE, Brussels, pp 1–5. <https://doi.org/10.1109/IC3D.2017.8251910>
- Noor AK (2016) The HoloLens revolution. *Mech Eng* 138(10):30–35. <https://doi.org/10.1115/1.2016-Oct-1>
- Peterson MP (1995) *Interactive and animated cartography*. Prentice Hall, Englewood Cliffs
- Plumert JM, Kearney JK, Cremer JF, Recker K (2005) Distance perceptions in real and virtual environments. *ACM Trans Appl Percept* 2(3):216–233. <https://doi.org/10.1145/1077399.1077402>
- Postma A, De Haan EHF (1996) What was where? Memory for object locations. *Q J Exp Psychol A* 49(1):178–199. <https://doi.org/10.1080/713755605>
- Ruddle RA, Volkova E, Mohler B, Bühlhoff HH (2011) The effect of landmark and body-based sensory information on route knowledge. *Mem Cogn* 39(4):686–699. <https://doi.org/10.3758/s13421-010-0054-z>
- Taylor DRF (2005) The theory and practice of cybercartography: an introduction. In: Taylor DRF (ed) *Cybercartography: theory and practice*. Elsevier, Amsterdam, pp 1–13
- Taylor DRF, Lauriault TP (2007) Future directions for multimedia cartography. In: Cartwright W, Peterson MP, Gartner G (eds) *Multimedia cartography*, 2nd edn. Springer, Berlin, pp 505–522
- Wang W, Wu X, Chen G, Chen Z (2018) Holo3DGIS: leveraging Microsoft HoloLens in 3D geographic information. *ISPRS Int J Geo Inf* 7(2):1–16. <https://doi.org/10.3390/ijgi7020060>
- Zhao B, Chen Q (2017) Location spoofing in a location-based game: a case study of Pokémon Go. In: Peterson MP (ed) *Advances in cartography and GI science. Selections from the international cartographic conference 2017*. Springer, Cham, pp 21–32