TYPE Original Research
PUBLISHED 26 August 2022
DOI 10.3389/frvir.2022.955437

OPEN ACCESS

EDITED BY
Daniel Zielasko,
University of Trier, Germany

REVIEWED BY
Roberto Pierdicca,
Marche Polytechnic University, Italy
Derek Reilly,
Dalhousie University, Canada

*CORRESPONDENCE
Linda Rau,
linda.rau@hs-rm.de

SPECIALTY SECTION
This article was submitted to Technologies for VR, a section of the journal Frontiers in Virtual Reality

RECEIVED 28 May 2022
ACCEPTED 27 July 2022
PUBLISHED 26 August 2022

CITATION
Rau L, Bitter JL, Liu Y, Spierling U and Dörner R (2022), Supporting the creation of non-linear everyday AR experiences in exhibitions and museums: An authoring process based on self-contained building blocks. *Front. Virtual Real.* 3:955437. doi: 10.3389/frvir.2022.955437

Supporting the creation of non-linear everyday AR experiences in exhibitions and museums: An authoring process based on self-contained building blocks

Linda Rau*, Jessica L. Bitter, Yu Liu, Ulrike Spierling and Ralf Dörner

Design, Computer Science, Media, RheinMain University of Applied Sciences, Wiesbaden, Germany

The use of Augmented Reality (AR) has the potential to make everyday experiences exciting and educational. For example, AR can augment exhibits in museums with animated and interactive content. The creation of this content, however, is still facing challenges. To meet these, we employ reusable, pattern-based building blocks called AR nuggets. An AR nugget implements one application pattern in a small and self-contained piece of software to provide a complete solution for recurrent AR-based experiences. For example, in the application context of museums and exhibitions, we identify superimposition or visualization of echolocation as general patterns for AR use cases. AR nuggets encapsulate AR-specific knowledge and sophisticated functionalities to support authors and reduce the authoring task to tweaking existing templates to individual exhibits. To connect AR nuggets used in different exhibition rooms, we present novel AR nuggets that encapsulate the functionalities needed for creating a path between the exhibits. Additionally, we provide examples of AR nuggets that implement a virtual character that guides visitors to exhibits of interest. With this new set of AR nuggets, spatial connections can be authored, e.g., in the form of a guided tour with interactive narration. For this authoring task, we introduce an AR nugget manager that supports authors in creating and adapting multiple non-linear AR experiences. We illustrate our approach with the creation of an everyday AR application for a museum of natural history, share our experiences and discuss to what extent our approach can mitigate authoring challenges for everyday AR applications from a museum's point of view. This work contributes to the field of everyday AR with 1) a pattern-based authoring concept to create complex everyday AR experiences based on self-contained building blocks, 2) a set of patterns that allows for spatial connections of these to create non-linear AR content, 3) means for supporting this authoring process in the form of an AR nugget manager, 4) lessons learned in applying our authoring concept in a real application case in a museum, 5) our observation of hurdles that still prevent more widespread use of AR in everyday applications during the realization of this application case.

KEYWORDS

augmented reality, museum, authoring tools, non-linear, everyday experience, navigation, patterns, authoring

1 Introduction

Augmented Reality (AR) can be used to improve everyday experiences. In the last years, AR applications have been developed for diverse use cases, such as navigation (Bachras et al., 2019; Guarese and Maciel, 2019) or exhibitions (Hammady et al., 2020; Vlizos et al., 2021), where the technology enhanced the overall experience. However, including AR experiences in a museum exhibition can be a great challenge for museum curators, who typically are not experienced with AR applications and how to apply AR to their exhibition. To alleviate this, we strive to support authors with an educative, media, or design background, but without or little AR or programming experience, in creating AR applications for typical museum scenarios. Our authoring approach is based on self-contained building blocks that we call AR nuggets (Rau et al. (2021)). An AR nugget is a ready-to-use application that implements one specific AR pattern and can be executed on its own. It can be adapted by an author without the need for programming as described by Rau et al. (2022). For adapting AR nuggets, we provide authoring tools with a graphical user interface. Our authoring process consists of a process-specific and a location-specific phase. For both phases, we introduce and implement multiple authoring tools that support authoring AR nuggets.

The AR patterns at the core of our AR nuggets are tailored to museum use cases. In association with museum experts, we analyze and address typical challenges for AR in museums. For example, museums need to guide their visitors to AR experiences, as they may augment only a selection of all exhibits with AR features. These AR exhibits might be located in different departments or on different floors. To address this issue, one category of AR nuggets implements guiding solutions for visitors to provide spatial connections between several AR exhibits. Further, it is unlikely that all visitors stick to the exact same linear order of events. Although major flow paths may exist, individuals choose freely where to go next, based on interest, footfall, or other circumstances. Therefore, we support non-linear event sequences by defining pre- and postconditions for AR nuggets. Additionally, an AR nugget manager controls the AR nugget instances during runtime.

We evaluate our concept and implemented tools by applying them to a real application case in the field: Authors with a media and design background create a non-linear AR demonstrator fitting into a permanent exhibition in the Senckenberg Museum in Germany.

The contributions of this work are as follows.

- We elaborate the pattern-based authoring concept of AR nuggets to create complex everyday AR experiences.
- We describe how the concept of AR nuggets can support the authoring process of AR applications that are suited for the non-linear interactivity of museum visitors. For the control of these, we introduce an AR nugget manager and further tools based on the Game Engine Unity (Unity Technologies, 2021). The tools support authoring at an office workstation with Unity as well as on-site in the target immersive environment.
- We propose a set of patterns from the application domain that allow for spatial connections of AR nuggets to create interactive, non-linear AR content.
- We state lessons learned in applying our authoring concept to a real showcase in a museum, where non-programmers use our authoring approach and tools. Here, we describe our observation of hurdles that may still prevent more widespread use of AR in everyday applications.

In the following section, we analyze related work. Next, we describe the concept of our self-contained building blocks, AR nuggets, and how we apply them. Section 4 describes how we implemented our authoring tools. We report how we used these tools for the creation of a demonstrator in Section 5. Finally, we draw a conclusion and point to directions for future work in Section 6.

2 Related work

Current work (e.g., Rau et al. (2022); Geronikolakis et al. (2020); Apaza-Yllachura et al. (2019)) explores module-based authoring of AR applications and targets to lower the barriers to getting started with the development of AR applications. Recently, similar ideas are used the industry, for example, Power Apps by Microsoft, 2021) or the XRTY App, 2021). For Virtual Reality (VR), Horst and Dörner (2019) introduced building blocks that target to support persons without programming knowledge in creating VR applications. The authors transfer ideas from the educational approach of microlearning to their building blocks. Microlearning divides learning content into small units that each have a learning goal independent from other learning nuggets. Based on the concept of learning nuggets in microlearning, the authors call their building blocks VR nuggets. They define one VR nugget as a stand-alone VR application that reflects a pattern from the application domain.

Follow-up work by Rau et al. (2021) transfers this microlearning approach to AR and introduces building blocks that the authors call AR nuggets. AR nuggets are small AR applications that are independent of other applications. Rau et al. (2022) also introduce an authoring tool for creating AR nuggets that allow authors without programming knowledge to create their own AR applications. The authors implement three default AR nuggets that augment a real-world image with one or more virtual objects. To adapt their AR nuggets, a desktop computer or handheld AR device can be used. The authoring tool can be executed simultaneously on both, the computer and AR device, and changes to the AR nugget are synchronized between both devices. However, their authoring tool does not support other anchors in the real world than marker images and only supports the interaction possibilities of handheld devices.

Geronikolakis et al. (2020) introduce their authoring tool called M.A.G.E.S. platform that allows generating gamified AR experiences for virtual exhibitions. It also illustrates a showcase of using various modes from their SDK to create AR experiences. The authors introduce a diagram of the interaction module with additions to support HoloLens gesture interactions. The authoring tool offers a module-based authoring method based on Unity and targets to help non-programmers in creating AR experiences for AR glasses. However, the above authoring tools are limited in their functionalities. For example, they do not support the creation of AR experiences for navigation or non-linear experiences. Therefore, more experienced authors might quickly get to the authoring tool's limits. As Ashtari et al. (2020) point out, if authors figure that they cannot implement the behavior that they intend, it is difficult to transition to more powerful tools like game engines. Nebeling and Speicher (2018) review existing AR and VR authoring tools and conclude that current tools either support only a low-level fidelity or require a high skill level. Krauß et al. (2021) analyze and discuss current challenges in creating AR applications. They point out that future AR authoring tools should be tailored to the target group and offer high-level authoring functionalities, e.g., specifying behavior in an AR application and process-specific authoring tasks.

For museums, not only creating AR experiences for exhibits can be challenging, but also guiding visitors to these exhibits to create a spatial connection. Here, a virtual guide can offer an entertaining alternative to human guides. Virtual guides are available on demand and can create a personalized experience for the visitor. Additionally, they may help with navigating pre-planned paths in an exhibition and reduce the need for visitors to be directed by human guides from one point of interest (PoI) to the next one ((Hammady et al., 2020)). Furthermore, navigation via a virtual guide can be faster than other approaches, as work by Campbell et al. (2014) indicates.

One application that includes virtual guides is introduced by Hammady et al. (2020). First, the authors observe museum visitors and categorize the visitors into four groups with different behaviors. Based on these, they identify 12 functions that they implement in their museum guide application called MuseumEye. At different PoI in a museum, the authors place virtual guides that explain the exhibit to the user and provide additional information on-site. In a user study, they show that AR-enhanced museum tours can add to a positive visitor experience. Their authoring pipeline includes various tools for 3D modeling, video editing, and programming. However, the authors do not describe if and how their museum application and its system design could be reused for other museums or how much programming knowledge is necessary to do so.

Work by Singh et al. (2021) implements a story authoring tool, called Story CreatAR, as a Unity plug-in. It targets locative AR experiences and their simulation in VR. Authors can create conversation nodes, human character agents, props such as tables or chairs, and 3D sound. The authoring tool offers an agent that moves in the scene based on the storylines of the experiences. The agent offers three different walking speeds with the goal to adapt to the visitor's speed. It calculates a pathway to its destination using the A* pathfinding algorithm. The authoring tool focuses on the spatial relationships between story elements and their creation. However, the authoring of the navigation process between these elements remains open.

Kampa and Spierling (2017) explored authoring processes for location-based handheld AR. Their aim is to enable storytelling by visual AR overlays of recorded video characters in front of real backdrops, such as ancient buildings, by displaying re-enactments of historical scenes. Their authoring process is twofold, with one part conceiving the exhibit and story at the office desk, and another locative part for matching AR content to the outdoor environment, employing a mobile authoring toolset. Their toolset allows to capture spatial preconditions and create placeholder content on site. Single AR scenes of an overarching story can have non-linear connections to each other by pre- and postconditions. The core for authoring is an XML-based template file that can be edited by non-programmers by adapting default structures.

Overall, there is a need for authoring tools that are targeted to museums' use cases and goals and that are suitable for non-programmers. For example, exhibition designers need tools that support the creation of non-linear AR experiences with high-level authoring functionalities. Besides the challenge of implementing AR experiences for their exhibits, museums also face the challenge to guide visitors to their AR-enhanced exhibits. Therefore, authoring tools for AR experiences that target museums should also support authors in implementing navigational functions.

3 Self-contained building blocks to facilitate AR authoring: AR nuggets

Our concept is based on self-contained building blocks that we call AR nuggets as introduced by Rau et al. (2021). Each AR nugget reflects one pattern from the application domain. A pattern, as described by Alexander et al. (1997), describes a solution for a reoccurring process. We identify reoccurring use cases from everyday scenarios where AR adds value and describe these in application patterns. AR nuggets provide solutions for these reoccurring use cases in form of ready-to-use applications. As a self-contained building block, one AR nugget includes all functionalities that it needs to be executed, i.e., all scripts, interactions, objects, and real-world anchors. We provide all mandatory components of an AR nugget as placeholders or default values. For example, an AR nugget that labels a virtual 3D model includes one placeholder label with a default text and one placeholder 3D model. Because the AR nugget already includes all mandatory functionalities, it can be executed right away within the development platform. In the authoring process, authors may replace the placeholder objects and adapt the default values.

Additionally, our AR nuggets include further functionalities that aim to facilitate the authoring process. We call these functional coatings. Multiple functional coatings can be applied to each AR nugget or parts of one AR nugget. Using these, authors can define how virtual content should appear, be anchored, and behave in the real world. Our *grabbable* tool allows authors or visitors to move, rotate, or scale a virtual object. Additionally, the virtual object can be moved, rotated, or scaled using Microsoft's Mixed Reality Toolkit's (MRTK) (Microsoft, 2020) far interactions, an air tap gesture, or a hand ray. Optionally, this may be combined with our *anchor* tool. It can be dragged to another virtual object to attach a spatial anchor to it. Virtual objects with a spatial anchor are anchored in the real-world and always appear at the anchored location, independently of when and where the AR device was started. The spatial anchor is updated automatically every time the virtual object is moved. In some cases, authors need to position virtual objects on real-world surfaces, e.g., indicator circles on the ground. Here, we support authors with our *tap to place* tool. This automatically rotates the virtual objects to align with the surfaces and eases the process of positioning them. Furthermore, we introduce a *rotate towards* tool, which rotates a virtual object to a defined position. For example, it can be used to rotate text to the user so that the text is easier to read. Another example is to rotate our agent in a guiding AR nugget in the direction it is currently navigating to. Lastly, we implement a *condition manager* that allows authors to work with conditions without programming or scripting. Authors may choose one or more conditions and one or more corresponding events that will be triggered when the condition(s) become(s) true. For example, if the user raises one hand, show a menu. Authors may use these functionalities to create a complex AR experience without the need for programming. We further extend the concept of AR nuggets with pre- and postconditions. When all postconditions of an AR nugget are met, it can be stopped and another AR nugget may start. Using preconditions, authors can define conditions that must be met before an AR nugget is started.

This work applies AR nuggets and their authoring to museums. As suggested by Liu et al. (2021), we see two categories for AR experiences in the context of a museum visit: navigating from one PoI to another and exploring a PoI. Typically, visitors start their museum tour with orientation and navigating to their first PoI. Here, we use an AR nugget from the navigating category to guide the visitor to a PoI. After arriving, visitors can start to explore the exhibit at the PoI, which we call the exploration use case, described in Section 3.2.

In the following, we present three AR nuggets that target navigating and further five AR nuggets that target exploring.

3.1 AR nuggets for navigation

Different approaches could be taken to realize a guiding function, e.g., arrows, direction signs, or an agent that the visitor can follow. As pointed out in Section 2, navigation with an agent is a practical way to guide visitors in a museum. Therefore, we realize the navigation process with an agent that leads the visitor to the desired PoI.

We elaborate a guiding logic that adapts to various visitor scenarios. For example, a visitor might want to change the destination, take a break, or skip the guiding process completely. We design a state machine with six states that control the behavior of the virtual agent. Authors can adapt the agent's behavior using four parameters $a - d$. Depending on these parameters and the visitor's movements during the guiding process, the agent adapts its states and behavior as visualized in Figure 1.

The agent starts in an idle state. When the guiding process is started, the state machine continuously calculates the agent's state based on the current distance between it and the visitor (d_a2v). The agent starts to move fast if this distance is greater than the value specified in a. If that distance is smaller than the value specified in a but greater than the one specified in b, the agent starts to move slowly. If the distance is smaller than the value specified in b, the agent waits for the visitor to catch up. These three states allow the agent to adapt to the visitor's walking pace or to pause. From the "waiting" state, if the distance between agent and visitor becomes larger and exceeds the value specified in c, the agent moves back towards the visitor. For example, if the visitor walks away from the agent and their distance becomes greater than the value in c, the agent moves back to the visitor. This can help to avoid that the agent vanishes behind exhibits and also to help the visitor to not lose eye contact to the agent. If the agent is configured to include spatial data from the real world for the path calculation, it casts a ray in the forward direction while guiding. If the ray hits an obstacle, a new path can be calculated. Similar to the calculation of the states and the distance between agent and visitor, the distance between the agent and the current node (d_a2n) is continuously calculated. If this distance becomes smaller than the value specified in d, the

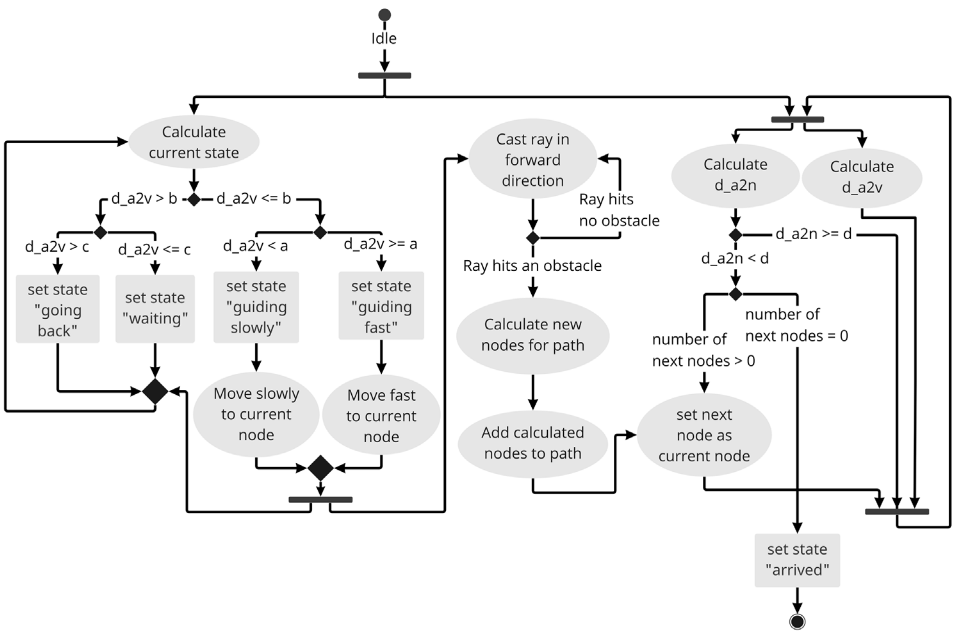

FIGURE 1
State diagram of the guiding state machine, showing how the agent behaves based on the distance between it and the visitor (*d_a2v*) and the distance between it and the current node (*d_a2n*). Authors can configure the agent's behavior using the parameters *a* – *d*. The states "going back", "waiting", "guiding slowly", and "guiding fast" are accessedd based on *a*, *b* and *d_a2v*. If the agent detects an obstacle on the path, it calculates new nodes and adds them to the path. When it arrives at a node, it starts to move to the next node. If there are no more next nodes, the state is set to "arrived".

agent starts moving to the next node, or, if the current node was the last node of the path, it switches to the "arrived" state.

Our default AR nugget for navigation with an agent includes one default agent and implements a pathfinding algorithm. The default agent can be customized and its visual representation, the avatar, can be replaced during the authoring process. For example, for navigating to a dinosaur exhibit, we could use an agent that is embodied by a dinosaur, and for navigating to a deep-sea exhibition, one could have the agent embodied by a marine creature. This AR nugget also includes interactions to start, pause, or resume the guiding process. Additionally, it includes information about the path where the visitor should be guided. We identify three different ways to implement a path within an AR nugget for navigation with an agent and describe these in the following.

Node-based navigation: One option to create a path from one location to another is based on nodes. An author may create a path by placing nodes at each corner of the path. Each node is connected to the prior node. Together, these nodes form a path that leads from the first node as starting position to the last node, which could be the next PoI. Nodes can be anchored in the real world using spatial anchors. Each AR nugget for node-based navigation with an agent includes one determined path without conjunctions. Therefore, the paths are not connected and visitors may only be guided on pre-authored pathways. This contributes to the AR nuggets' independence because each AR nugget has its own nodes. If one AR nugget and its nodes are deleted, it is ensured that all other AR nuggets are still functional.

Navigation based on pre-processed scan data: Our second option is calculating a pathway based on pre-processed scan data. This can simplify authoring a path because authors would only need to define the path's goal. Then, the system can calculate a path that starts at the visitor's current position and targets the path's goal at run-time. However, further knowledge about the museum's area is necessary to calculate this path. Therefore, it is required to scan and pre-process the museum's area beforehand. Then, the AR nugget can use this scan data for pathway calculations. This option calculates a pathway when the guiding process is started and navigates visitors to the pathway's goal starting from any location. It does not require to pre-define the pathway beforehand. Thus, this approach is more flexible than the node-based option. If the visitor leaves the pathway, the AR nugget can navigate them back to it. If the visitor deviates from the path so far that another path may be easier from the visitor's current position, the AR nugget automatically detects this and continues to navigate the visitor based on the newly calculated easier path. Because this option requires the author only to place one goal for one path and not multiple nodes, authoring pathways can be faster with this option compared to our option based on nodes. However, the scanning task as well as pre-processing the scan makes authoring this AR nugget more complex. Whether this option or our node-based option should be preferred likely depends on how many paths are supposed to be included in the whole museum application. If only a few paths are needed and the area is rather large, using the node-based option would be recommendable. In contrast, if numerous paths will be authored, it can be worth doing the scanning and pre-processing to save time and effort when authoring the paths.

Spatial mapping information for pathway calculations: Our third option is calculating pathways based on spatial real-world data. Usually, in a museum or other everyday environments, we are not alone but there are other people around. If a person or any physical object is detected blocking the path, this AR nugget type should detect this to re-calculate a new sub-path and the agent guide should navigate around the obstacle, back to the original path. For example, if a person is standing in front of the visitor, the agent guide would navigate around that person and then continue navigating following the original path. To detect this, the AR hardware needs to continuously gather spatial data about the real world. While this pattern can be combined with the other introduced patterns for navigation, it can also be implemented as a stand-alone AR nugget. In the latter case, an author would be required to place the path's goal and could optionally place further nodes. Then, the guide would navigate straight towards the path's goal and automatically take turns to avoid walls or other physical objects blocking the path. However, this pattern might not always find the fastest path. In some cases, it might not be able to find a path at all, e.g., if the path has a dead end and the guide gets stuck there.

Indicators: In a big museum, it is probable that not every sample, artifact or specimen in one exhibition room offers an AR option. Therefore, visitors need further cues that indicate possible AR experiences at a PoI, also after successfully being guided to a room. For example, visitors may see an outline around AR-enhanced PoI, or search the environment with help of a 2D stencil displayed in the viewport, similar to described by Spierling et al. (2016). In small rooms, this could also serve as the only orientation, because a guide and navigation are more suitable for large distances. Our default Indicators AR nugget contains the following components. First, it needs one or more visual hints to indicate that certain PoI are augmented. These hints may be any 3D models that fit the target exhibition, for example, arrows, circles, or any hand-modeled geometries. For linear visiting sequences, a visual hint can also be a number. This is useful if several indicators are placed in the visible vicinity to indicate a suggested order. As default geometries, we use a circle on the floor around the position of the exhibit. For placing indicators, including their elements at proper positions in the exhibition, each indicator needs an anchor in the real world that authors can adapt. An optional text shows the exhibit's title or a label. This AR nugget also includes a trigger area (for example, a box collider). In case a visitor enters that area, another AR nugget from the exploring category can be started.

3.2 AR nuggets for exploring

Based on a typical museum visit, we identify patterns that serve as a basis to create default AR nuggets for exploring exhibits. At each PoI, an AR nugget from the exploring category can enhance the experience, e.g., by providing further information. We discussed these patterns with museum experts to validate the patterns' and AR nuggets' values for a museum context. However, we think that the patterns and AR nuggets can also be applied to further use cases.

Superimposition: One use case is to overlay a physical exhibit with a fitting hologram. It targets exhibits where only parts or inner structures, e.g., bones, are displayed physically. Here, photo-realistic 3D models can be superimposed onto physical bones, skeletons, objects, etc. This is feasible for extinct species, where the superimposition can show different theories about what the creature may have looked like. Similarly, it is also an interesting feature for living species, to show how the hide or fur wraps around the skeleton. Based on this use case, we identified the necessary components to implement a superimposition AR nugget. First, the superimposition AR nugget needs a 3D model that fits the physical exhibit and serves as a hologram. It should have a similar size and shape as the physical object to create the best possible AR experience. If the 3D model does not fit, it might induce confusion or irritation in the visitor. Authors need to be able to align the 3D model with the real exhibit. Then, the 3D model's position in the real world must be saved. For the visitor, we make this AR nugget non-interactive to prevent the visitors from inadvertently adjusting the position of the 3D models.

Transparency: Additionally to the superimposition AR nugget described above, it can be feasible to give visitors the ability to change the degree of occlusion from the superimposing 3D model. For example, this could show an animal's skin as the outside layer and the inner organs of the animal as the inside layer. Visitors can control what layer they want to view using a virtual slider UI element. Our transparency AR nugget includes this slider with the functionality to control the degree of occlusion from the outside layer. Additionally, the AR nugget includes at least one 3D model for the outside and inside layers each. These 3D models are placed in the real world by authors and anchored there, similar to the 3D models in our superimposition AR nugget.

Visualizing echolocation: Echolocation is a specific way of localization via sonar waves and is used by some animals, like whales and bats. Echolocation is also used in technology, e.g., submarines use sonar to detect other vessels or ships. We identify a pattern that shows how these animals or technology localize their surroundings by illustrating the sonar waves and how a sonar wave moves through space. Visitors should have real-time feedback on their interactions, i.e., when they hold their hand into the animated waves, the virtual sonar waves should behave similarly to how real sonar waves would when encountering an obstacle. The AR nugget contains 3D models and animations for the sonar waves and a trigger system for this interactive part. Authors need to be able to define an anchor in the real world that serves as the origin of the sonar waves.

Explosion: Providing diagrams to illustrate sub-parts and their connections is a practical method in data visualizations (Li et al., 2004; Luboschik and Schumann, 2007). Exhibits in museums may need explanation methods for complex structures and connections. For example, most ancient animal fossils are only found with partial pieces because partial limbs are missing. Here, an explosion diagram can help visitors to understand the structural relations between a physical fossil and its missing parts. Thus, we provide an explosion AR nugget to illustrate the internal structures of an artifact via pieces of linear motion animations. The AR nugget includes a virtual default object, that represents one sub-part of the fossil, with a linear motion animation that sequentially shows its connection. Each of these sub-parts can be anchored in the real world. Additionally, the AR nugget provides an interface to expand or fold the artifact structure.

Sequential explanation: Commonly, further information about an exhibit in museums is provided by boards or signs with images and text close to it. Although there are projects that explored AR-related solutions to give further information on exhibits ((Tarantino et al., 2019; Huang et al., 2020)), visitors should still find the related points from the audio information with their eyes. We introduce a sequential explanation AR nugget to provide exhibit explanations. Three sections are included in this AR nugget. Firstly, it contains several pieces of audio information. Secondly, each piece of audio information relates to a visual object for supporting visitors in searching for the mentioned key points. Authors can attach visual objects to positions in the real world. Lastly, it provides an interface for visitors to pause, skip, and replay the audio explanation.

3.3 Authoring process and tools

Our conceptualized authoring process aims at supporting authors of location-dependent, immersive AR experiences within a non-linear museum tour. Because of the AR content's dependence on precisely fitting to a real, physical exhibition, this creation process has at least two phases (similar as described by Kampa and Spierling (2017)). At first, authors concentrate on authoring tasks that are not location-specific, but process-specific. Typically, authors prefer to work at a desk or workstation during this phase. The second phase needs to take place at the final location, to anchor virtual objects into the real world or to adjust proper sizes within the environment. This phase is most suitable to be performed with an AR device, preferably the target device, through which authors can directly see and inspect the fitting overlay. These two phases may need to be repeated in several iterations. We visualize these two phases in Figure 2.

In the first authoring phase, authors choose, initialize and arrange AR nuggets that fit their ideas. These small and self-contained AR applications stay independent from each other and can be re-arranged, exchanged, or removed anytime during the process. Furthermore, parameter values can be adapted, assets within the AR nuggets can be replaced, and functional coatings can be added. To support authors in this first phase, we propose an AR nugget manager tool that aims to 1) ensure that the AR nuggets are independent of each other, 2) provide authors with an overview of their AR nuggets, 3) take care of controlling which nugget is executed at which time, and 4) allow references and connections between AR nuggets by defining pre- and postconditions. We visualize how the AR nugget manager controls the execution of AR nuggets in Figure 3. If one application includes more than one AR nugget, the application automatically adds an AR nugget manager. Then, when the application is started, the AR nugget manager starts the first AR nugget. By default, this is the first AR nugget that the author created. However, authors may adapt which AR nugget is executed first. While the application and the AR nugget manager are running, every subsequent AR nugget then starts and stops based on its defined pre- and postconditions.

For example, an AR nugget may be triggered to start by distance to the visitor, or when a previous AR nugget has ended. The list of conditions is expandable based on newly emerging use cases in everyday scenarios. We provide a default set of pre- and postconditions that we think is versatile for museum scenarios. Because there is always exactly one AR nugget running at a time, the currently executed AR nugget's postconditions must be met before the next AR nugget may start. Once its postconditions are met, the AR nugget manager checks if preconditions for another AR nugget are met. If yes, the AR nugget manager starts that AR nugget. To control this, the AR nugget manager includes a list of all AR nuggets that have their preconditions met. Each AR nugget checks if its preconditions are met and adds or removes itself from the list. The AR nugget manager only starts the next AR nugget at the same time as it ends the previous AR nugget. This ensures that there is always only one active AR nugget. Throughout the experience, this process is repeated every time a nugget ends to ensure that the next nugget can start. For example, a precondition for the first AR nugget in a museum visit could be that the user is located close to the starting point of the visit. Its postcondition could be that the AR nugget was fully experienced. For a subsequent AR nugget, the precondition could be that the previous AR nugget was fully experienced and again that the user is close to the subsequent AR nugget. In this case, the AR nugget manager would start the first AR nugget and the visitor can experience it. Then, once the postcondition of this AR nugget is met, the AR nugget manager would start to check if there are AR nuggets where the preconditions are met. In this case, when the user is close enough to the subsequent AR nugget, the AR nugget manager would end the first AR nugget and start the subsequent one. If the preconditions for multiple AR nuggets are met, the AR nugget manager checks if one of those was not experienced before and

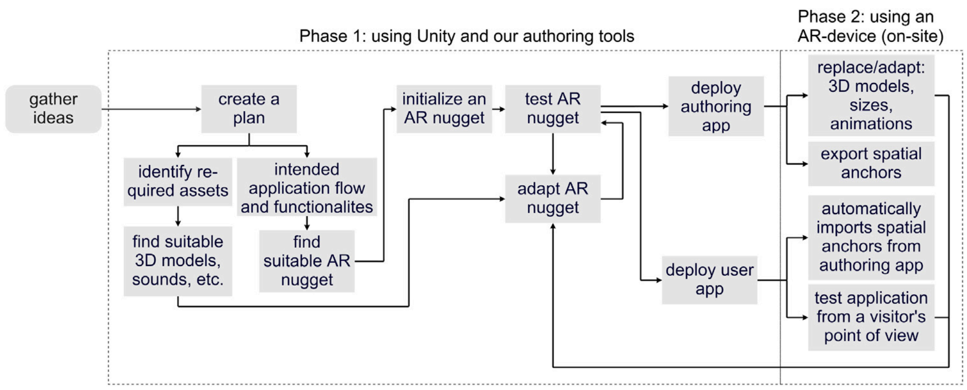
Phase 1: using Unity and our authoring tools
Phase 2: using an AR-device (on-site)
gather ideas
create a plan
identify required assets
intended application flow and functionalites
find suitable 3D models, sounds, etc.
find suitable AR nugget
initialize an AR nugget
test AR nugget
adapt AR nugget
deploy authoring app
deploy user app
replace/adapt: 3D models, sizes, animations
export spatial anchors
automatically imports spatial anchors from authoring app
test application from a visitor's point of view

FIGURE 2
Flowchart of the authoring process and its two phases. Authors start with process-specific tasks in phase 1. They use Unity and our authoring tools to adapt their AR nuggets and to deploy an authoring as well as a user application. In phase 2, authors use an AR device to perform location-specific authoring tasks like placing and scaling augmentations or testing the AR experience from a visitor's point of view. The authoring process is iterative and authors might go back to phase 1 to further adapt their AR nuggets.

starts that one. For example, if a guiding AR nugget to one POI and another one to another POI are implemented and both AR nuggets have their preconditions met, the AR nugget manager would choose the guiding AR nugget that was not experienced before and guide visitors to the POI that they have not visited yet. If more than one were not experienced before, the AR nugget manager checks which AR nugget the visitor is closer to, and then starts that AR nugget. Here, we find it suitable to check for the closest AR nugget because visitors can hardly experience an AR nugget if they are in another room.

In the second authoring phase, the author may build the application to an AR device and bring it to the exhibition on-site. The precise anchoring of the assets and components of each nugget in the real-world environment takes place here. We create on-site authoring tools that make assets and their parameters as well as anchors in the real-world exchangeable by storing them in exchange files (*load assets*, *save and load values*, and *anchor ex-/import* tools). This allows authors to exchange assets and adapt parameters without having to go back to the first phase at their desk or having to build and deploy the application again. During this development phase, authors need to switch between authoring and testing their application from a visitor's point of view. Therefore, we introduce a further functional coating that we call *mode switcher*. This tool switches between authoring mode and user mode to support authors in creating one authoring and one visitor application in only one authoring workflow.

4 Implementing authoring tools, AR nuggets, and functional coatings

We implement our authoring tools, AR nuggets, and their functional coatings with the Game Engine Unity (Unity Technologies, 2021). Our authoring tools can be used directly in Unity, which authors would use as an authoring environment. This has the advantage that the tools can draw from Unity functions like its included build pipeline to deploy applications to devices. Also, authoring tools can easily be extended or updated and authors with more experience are not restricted by the potential limitations of an AR-specific authoring environment. Unity also includes several options for target devices so that our AR nuggets and tools are not committed to only one specific device. However, for our museum use case, we target the Microsoft HoloLens 2.

Unity allows creating so called Prefabs. A Prefab can store components, properties, and values and works as a template. We implement our default AR nuggets as Unity Prefabs. Our authors may then choose one of the prefabs and create their default AR nugget from it via drag and drop. With drag and drop, we strive to make the authoring process suitable for usage in everyday scenarios for persons with no or little programming knowledge. Authors may create an application with multiple AR nuggets simply by using drag and drop to create an AR nugget multiple times.

We implement one AR nugget for each navigation type. In our system, each AR nugget implements one pathway with one goal. Multiple Navigation AR nuggets may be implemented to connect multiple pathways or to give a museum visitor a range of destinations to choose from. All Navigation AR nuggets include one agent and a menu to control the agent. In user mode, this menu includes start, pause, and resume guiding. In authoring mode, further menu elements allow to create a path or define a goal for one path. The *Navigation Based on Scan Data AR nugget* furthermore implements one pre-scanned default environment as well as one default goal within this environment. Authors may change the pathway's goal using the menu elements. Vuforia and their Area Target approach (PTC, 2021b) allow to tracking environments based on scan data. So, we scanned a default

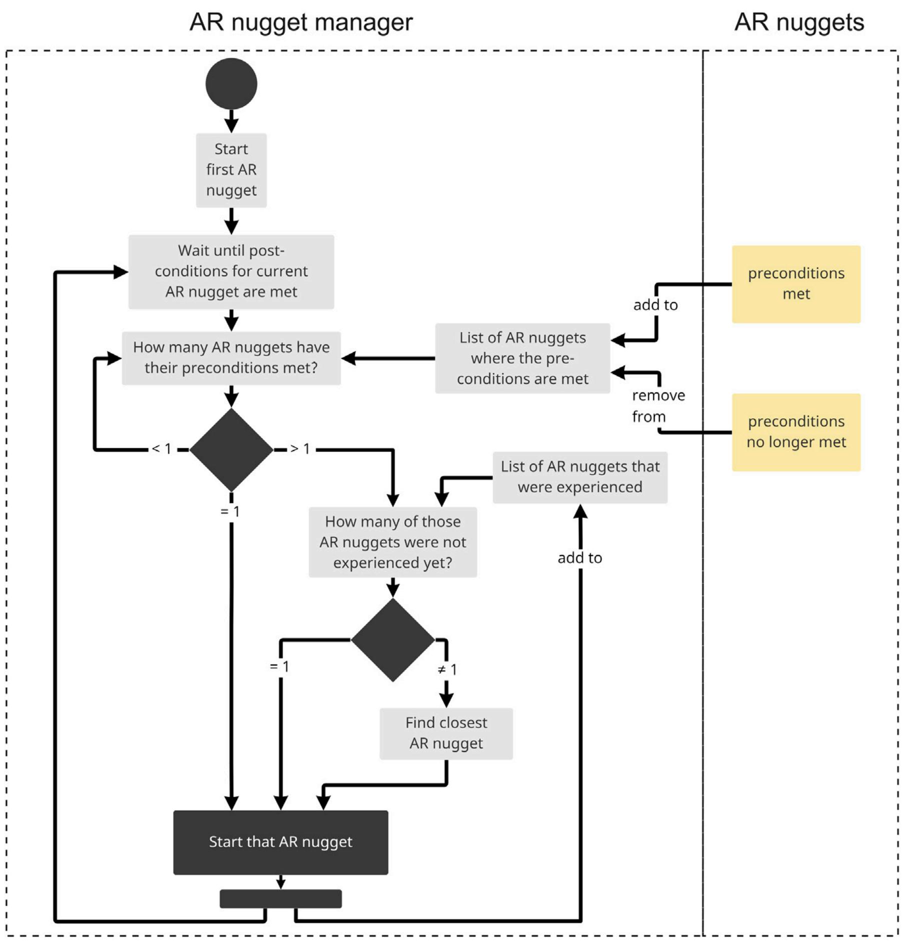
AR nugget manager
AR nuggets
Start first AR nugget
Wait until post-conditions for current AR nugget are met
How many AR nuggets have their preconditions met?
List of AR nuggets where the pre-conditions are met
add to
remove from
preconditions met
preconditions no longer met
< 1
> 1
= 1
List of AR nuggets that were experienced
How many of those AR nuggets were not experienced yet?
add to
= 1
≠ 1
Find closest AR nugget
Start that AR nugget

FIGURE 3
Activity diagram of the AR nugget manager and AR nuggets. After starting the first AR nugget, the AR nugget manager waits until the AR nugget's postconditions are met. Then, it searches a list for other AR nuggets where the preconditions are met. All AR nuggets add or remove themselves from this list when their preconditions are met or no longer met. If there is only one AR nugget in the list, the AR nugget manager starts it. Otherwise, it checks if multiple AR nuggets were not experienced yet. In that case, it starts the AR nugget that is closest to the visitor. Otherwise, it starts the AR nugget that was not experienced yet. The AR nugget manager adds the AR nugget that it just started to the list of experienced AR nuggets and then waits again until the AR nugget's postconditions are met.

environment using a NavVis M5 indoor mapping system. Then, using the Area Target Generator tool from Vuforia (PTC, 2021a), we create an Area Target based on the scan data. The *Navigation Based on Spatial Mapping AR nugget* implements one goal, but no scan data. Again, authors can move or replace this goal using the menu elements. The *Navigation Based on Nodes* AR nugget implements further menu elements to support authors can in creating nodes for the pathway. The *Indicator* AR nugget implements two virtual circles and one virtual line that connects both. Each of these three objects has its own spatial anchor. In the authoring mode, we give the objects a Tap2Place functionality that allows authors to place these objects directly on the ground or any other surface.

The AR nugget *Superimposing the Shell* includes one default 3D model with one spatial anchor. For the authoring mode, we restrict rotations to only around the up-axis. This aims to make placing the virtual 3D model easier. However, we also allow authors to deactivate this option in case they want to tilt the virtual object. Our *Transparency* AR nugget implements one default object and a slider. When users interact with the slider, the object's transparency changes. The AR nugget already implements different types of interactions based on the MRTK to allow moving the slider.

The default AR nuggets already include typical pre- and postconditions. Their default value depends on the AR nugget type. For AR nuggets for navigation, the default precondition is that the visitor is close to the pathway's starting point. Their default postcondition is that the guiding process is completed, i.e., the agent's state is "arrived". For other AR nuggets, the default precondition is that the visitor is close to the AR nugget's content's location and the postcondition is met when the visitor has moved away from the AR nugget. We equip each AR nugget with dropdown-menus where authors can select their desired pre- or postconditions. Programming experts can add further pre- and postcondtions to the system by extending the code. We provide the following options for pre- and postconditions within our AR nuggets: the visitor collides/does not collide with another virtual object, the visitor is closer/further than a certain value to another object, a specific other AR nugget was experienced, another virtual object is visible, and an agent's states from a navigation AR nugget. Also, authors can select and define multiple pre- and postconditions for one AR nugget. For example, an author could define that the preconditions of an AR nugget are met, and therefore the AR nugget is ready to start when the visitor is close to a defined location and the agent's state is "arrived". We implement the AR nugget manager to access the pre- and postconditions from the AR nuggets and to control them based on that information. Authors can configure one starting AR nugget that is executed at the beginning without checking whether its preconditions are met.

Similar to our AR nuggets, we implement our functional coatings for AR nuggets from Section 3 as Unity Prefabs. This way, authors can add functional coatings to a whole AR nugget or to parts of one AR nugget by choosing one or more from a list and adding them with drag and drop. For example, an author could drag and drop a *grabbable* coating to a default object to make it grabbable. We base our functional coatings *grabbable*, *anchor*, *tap to place*, and *rotate towards* on functionalities from MRTK. In contrast to the functionalities of MRTK, we provide these functional coatings as prefabs to support using them by drag and drop and without programming. In addition to the MRTK-based scripts, our prefabs include functionalities to ensure that the scripts are executed correctly, without the need to define or add possible needed components. For example, virtual objects need to define a so-called collider in order to detect collisions, which is required for *grabbable* or *tap to place*. Therefore, these functional coatings check if the virtual object defines a collider and adds one if necessary. For the *condition manager*, we use drop-down menus to choose conditions, similar to choosing pre- and postconditions for AR nuggets. To define what happens under the chosen condition, authors can choose from a list of events, again using a drop-down menu. We also implement a Unity event that allows more experienced authors to define any event that they like.

We also use Unity to implement our authoring tools. For our *save and load values* tool, we use JSON as a readable exchange format. We use JsonUtility, Unity's utility functions for JSON data, to save values to a JSON string. Then, using our implemented functions, we can save the JSON string to a file and store it on the HoloLens 2. To *load assets*, we utilize Unity's assetBundle function. One assetBundle includes one or more assets, e.g., 3D objects. Authors can export an assetBundle from Unity and then replace one existing virtual object in their AR nugget with one virtual object from the assetBundle. To anchor virtual objects or other virtual content in the real world, we implement an *anchor ex-/import* tool and use spatial anchors. Spatial anchors are a feature of MRTK. One spatial anchor represents one point in the real world. Spatial anchors can be accessed from different devices. Authors can create spatial anchors using an AR device, e.g., a HoloLens or a smartphone. Then, the spatial anchors are stored within the AR application. We implement one tool that enables authors to export these spatial anchors from the application to a file. Then, authors can click a button in their AR application to export the spatial anchors. Accordingly, we create another tool that imports the spatial anchors from a previously exported file. Authors can choose whether spatial anchors are imported by clicking a button or automatically at the start of the application. Using both tools, authors may use one authoring application to place their virtual content and export the spatial anchors while museum visitors may use another application that does not allow any authoring actions. Additionally, we implement the *mode switcher* tool that divides authoring functionalities from visitor functionalities. With this tool, authors only need one click to switch from one mode to the other one and can then build and deploy their authoring or visitor application. Besides restricting authoring functionalities from visitors, the museum would likely restrict them from using other applications than the designed museum tour application. For this, the Microsoft HoloLens 2 uses a kiosk mode. When enabled, the HoloLens 2 automatically starts the chosen application and it is not possible to go to the home menu to start another application. If an application is started in visitor mode, we automatically import the spatial anchors that were exported by the same application in authoring mode.

5 Making of an integrated demonstrator

We created and deployed an integrated demonstrator with immersive AR experiences for a museum tour at the Senckenberg Museum in Germany. The demonstrator combines visitor navigation with local exhibit augmentations. The creation of this AR application also served as a formative evaluation of our authoring concepts, processes, and tools presented above. Several iterations of producing test results with corresponding refinements directly optimized our authoring tools during their implementation phase. The process also showed that the tools can support non-programming authors to complete the authoring tasks without coding, as they have been used by two media designers of our team. The resulting application is suited for a natural history exhibition about whales, to be experienced with a HoloLens 2 as the target device. In the following, we describe this making of an AR museum application and report our findings.

The creative process started with visiting the museum, researching necessary information, and looking for suitable objects as well as places to apply augmentations. Figure 4 shows the resulting rough draft for the first designed visitor journey. After visitors receive the head-mounted AR device with instructions, a virtual avatar–in this case, a blue butterfly–shall guide them from the entrance to the whale department. This virtual butterfly can appear to sit on a physical pillar, one at the

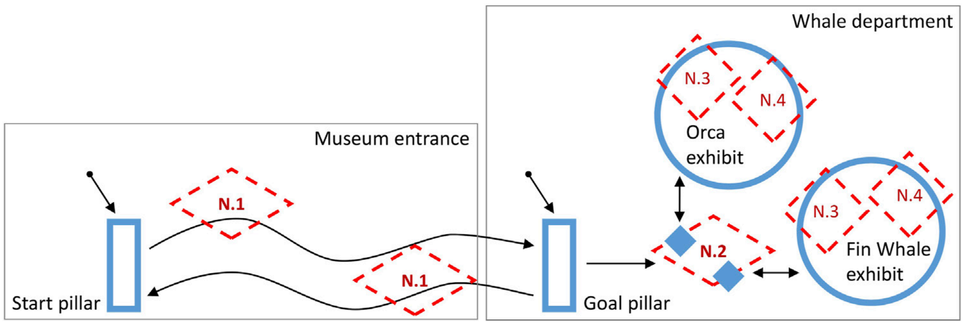

FIGURE 4
Abstract concept of the museum tour. N.1: Node-based navigation AR nuggets; N.2: Indicator AR nugget; N.3: Superimposition AR nuggets; N.4: Transparency AR nuggets. The museum tour starts at a physical start pillar close to the museum's entrance. Visitors are guided from there to the whale department, where an indicator AR nugget indicates which exhibits offer AR experiences. Visitors can experience the AR nuggets in the whale exhibition and return to the physical goal pillar to be guided back to the museum's entrance.

entrance booth and another one at the end point of the navigation path, also easing orientation for visitors. On arrival at the whale room, the butterfly disappears and visitors see diamonds floating above two points of interest, with circles appearing on the floor close to the positions of two whale skeletons that offer augmentations. It is then the visitors' choice to decide which of the exhibits to approach first. After walking close enough to the specimen, stepping into one circle, further visualizations can be triggered, for example, to illustrate how these mammals have looked alive, by superimposing a 3D model of their skin shape over the real skeleton. Anytime, for example, after exploring the two exhibits, users can approach the pillar, where they meet the butterfly again that guides them back towards the museum's entrance.

After these ideas were sketched, it was necessary to transfer them to a more formal plan for technical authoring with the tools. We started this process by planning what components our AR experience should be composed of. The plan includes two lists. The first one records required elements - such as assets - for our envisioned story, and the second documents needed functionalities - such as conditions and triggers - for the flow of the expected events. It was essential to group these elements and conditions into sequential building blocks that each form one pattern. Using the authoring concept of AR nuggets, ideally, each of these patterns corresponds with an existing nugget as a template. In our research project, the concept and implementation of the initial set of AR nuggets for exhibitions and the design of the museum visit have been done in parallel. Therefore, when using the tools to author the demonstrator, the next step towards authoring was to search the AR nugget pool for suitable AR nuggets. Figure 4 also indicates the AR nuggets that the authors have chosen to create the tour with the tools. We used two AR nuggets for node-based navigation to create two pathways. We let the navigation AR nuggets start when the visitor is near the physical pillar and end when the guiding process is finished. At the two whale exhibits, we need an indicator AR nugget that contains visible range markers and options for further preconditions and post-conditions, defining events to start and stop further nuggets, such as for superimposition.

After creating this plan, we could begin with the first phase of the technical authoring process described in Section 3.3. At our workplace, we opened the Unity project where our authoring tools and pool of default AR nuggets are implemented. In Unity's editor, we opened a default scene that already includes MRTK and an AR nugget manager. Next, we selected the node-based navigation AR nugget and added it to the hierarchy of the scene by drag and drop. Automatically, the placeholder elements were established. We checked whether all of the functionalities that we planned for were covered by the AR nugget. After confirming that this was the case, we added a second instance of this AR nugget. Because one node-based AR nugget implements only one pathway, we need one AR nugget that navigates a visitor towards the exhibition, and another one that later navigates the visitor back towards the entrance.

We also selected an indicator and a superimposition AR nugget and dragged them into the hierarchy. We duplicated the latter, which is another method of creating more copies of the same AR nugget. For better facility of inspection, we renamed the AR nuggets according to their purpose, e.g., "Guiding Back", and "Superimposition Fin Whale". Within an AR nugget, it is possible to define the conditional behavior of the included elements. We selected predefined conditions without typing any code, for example, to let a feedback element appear when the user collides with the range marker.

The resulting application could be tested in Unity's Play Mode. Here, the visitor's journey can be simulated. Existing logical mistakes in the pre- and postconditions of the AR nuggets could be identified and iteratively adapted by comparing the behavior of the AR nuggets with the application plan described above. When the application performed as intended, it was deployed to a HoloLens 2. With the running app on the HoloLens 2, we were ready for the second

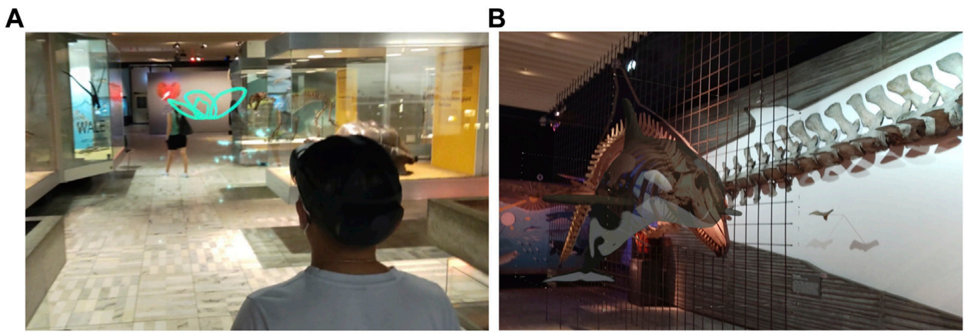

FIGURE 5
AR nuggets in the Senckenberg Museum. **(A)**: A node-based navigation AR nugget guides a visitor to the whale exhibition. The guiding agent is embodied by a blue butterfly. **(B)**: Making of a superimposition AR nugget augmenting a whale skeleton. The virtual geometry does not fit the physical exhibit accurately. Using our authoring tools, authors can quickly exchange the virtual 3D model with an updated version.

locative phase of the authoring process described in Section 3.3, which took place on-site. After opening the application on the device, we grabbed the virtual 3D models with our hands and placed them precisely within the real exhibition, saving their anchors by using the menus. One example of a superimposition AR nugget is shown if Figure 5B. We proceeded similarly with authoring the navigation path by placing path node markers at corners and turns. The navigation process is shown in Figure 5. To be able to reuse the positions in the real world in the visitor application later, we exported the spatial anchors to the device. We could also have exchanged geometries at this stage, but the virtual objects, including the avatar, fitted our ideas. Further adjustments needed to be made at the museum regarding parameters like distances, sizes, and speed of the guiding agent. We saved these values in a JSON file to be imported to the visitor application later. It has been hard to guess and set up the proper distances while sitting at the desktop. Especially the huge sizes of the whales led to unsuitable estimations of viewpoint distances, which needed correction.

On-site, we noted that some further tools could contribute to simplifying the authoring process. The authoring process for the node-based navigation AR nugget could be accelerated if the guiding process could be stopped and a node could be inserted anywhere in the path, not only at the end. Moreover, it proved to be exhausting to author in a big space like a museum. It is necessary to walk back and forth when authoring the navigation path, and grabbing nodes to position them. It is difficult to heed passing visitors when a grabbed node is blocking the view-port, hence the author may stretch their arm out, which again may cause problems for other visitors. Finally, we realized that the process of going on-site and returning to the desktop should be iterated more than once. Authors may realize on-site that the concept they implemented at the desktop may be insufficient or off for the real exhibits. In the following, they need to adjust their AR nuggets at their workplace, and then return on-site to verify the concept again. Especially the last adjustment steps in the second phase are designed to be performed by non-programming domain experts in the museum. For this, basic instructions on how to use the authoring menus are needed. These instructions should also appear during the authoring process in the authoring app on the HoloLens 2.

6 Conclusion

This work explored how AR nuggets can support the authoring of non-linear AR experiences for an everyday context. AR nuggets are self-contained building blocks based on patterns that fit the application domain. We identified five reoccurring use cases in museums for which AR can contribute to an educational and exciting experience and reflected them in AR nuggets. Additionally, we use a virtual agent to guide visitors from one exhibit to another using navigation AR nuggets. These allow to spatially connect AR nuggets from location-specific exhibits in a museum and enable the creation of one larger AR experience based on multiple AR nuggets. For these, we proposed and implemented three distinguished ways to author the AR nugget's pathway: based on nodes, based on pre-processed scan data, and based on real-time spatial mapping information. To define which AR nugget visitors experience at what time, we introduced pre- and postconditions to AR nuggets. This is supported by an AR nugget manager that controls which AR nugget is executed at which time. Furthermore, the AR nugget manager ensures that there is always exactly one AR nugget executed at one time. Executing only one AR nugget at a time contributes to the AR nuggets' independence from each other and helps to create an AR application that runs smoothly on AR devices that typically have limited computing power. This was suitable for our scenario in a museum. However, there might be other scenarios where it is more suitable to execute two or more AR nuggets simultaneously.

To further support the nugget-based authoring process, we introduced and implemented five functional coatings that hat can be used via drag and drop to extend an AR nugget or parts of it with functionalities. Additionally, we introduced and implemented further authoring tools to support creating non-linear AR experiences. Both, the functional coatings and our authoring tools, do not require programming knowledge. Our authoring process, AR nuggets, and tools are based on the Game Engine Unity. On the one hand, this allows to easily update them or to add further ones and also offers more possibilities to experienced authors. For example, if authors without programming knowledge need a specific AR nugget that is not implemented yet, a programming expert can extend the existing pool of AR nuggets with a new one. Also, further tools or functional coatings can be added to extend and update our authoring system. On the other hand, it could be a hurdle for authors to install and become familiar with Unity and its basic functions, like managing prefabs, in order to use them.

Our authoring tools and AR nuggets with functional coatings were used by authors with a media and design background. Using our authoring tools and AR nuggets, the authors were able to create a non-linear demonstrator application for a real museum application for a natural history exhibition in the Senckenberg Museum in Germany. The authors worked iteratively in two phases. First, they worked at a desk to create an application based on AR nuggets with pre- and postconditions. Here, it is necessary that authors have a specific idea of their intended tour. They need to be able to translate their plan to our AR nuggets and the AR nugget manager. In a second authoring phase, the authors deployed their AR application to a HoloLens 2 to fine-tune the content and test it on-site.

In future work, we plan to develop improved user interfaces for the authoring tools and target to increase the system's usability. In our case, the authors with media and design backgrounds were already acquainted with the authoring tools and how to use them. However, for authors who are new to our tools, we see the requirement to provide short explanations for every authoring step. We plan to include them directly where the author needs to adjust values and geometries. In addition, we plan to enhance the guidance of the author's attention by highlighting which values should be adapted. For example, when an author defines that an AR nugget starts based on the precondition that the visitor is closer than a certain value to a location, the author should adapt that value. Here, it could be supportive if the user interface draws the author's attention to this value. Aside from that, we are in the process of creating further AR nuggets to broaden the scope of use cases that can be targeted with our nugget system.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Ethics statement

Written informed consent was obtained from the individual(s) for the publication of any potentially identifiable images or data included in this article.

Author contributions

All authors contributed to the work relative to the order of enumeration and agree to be accountable for the content of the work.

Funding

This work has been funded by the German Federal Ministry of Education and Research (BMBF), funding program Forschung an Fachhochschulen, contract number 13FH181PX8.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Alexander, C., Ishikawa, S., and Silverstein, M. (1997). *A pattern language: Towns, buildings, construction*.

Apaza-Yllachura, Y., Paz-Valderrama, A., and Corrales-Delgado, C. (2019). "Simplear: Augmented reality high-level content design framework using visual programming," in 2019 38th International Conference of the Chilean Computer Science (Society SCCC IEEE), 1–7.

Ashtari, N., Bunt, A., McGrenere, J., Nebeling, M., and Chilana, P. K. (2020). "Creating augmented and virtual reality applications: Current practices, challenges, and opportunities," in Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, 1–13.

Bachras, V., Raptis, G. E., and Avouris, N. M. (2019). "On the use of persistent spatial points for deploying path navigation in augmented reality: An evaluation study," in *Human-computer interaction – INTERACT 2019 of lecture notes in computer science*. Editors D. Lamas, F. Loizides, L. Nacke, H. Petrie, M. Winckler, and P. Zaphiris (Cham: Springer International Publishing), 11749, 309–318. doi:10.1007/978-3-030-29390-1_17

Campbell, A. G., Stafford, J. W., Holz, T., and O'Hare, G. M. P. (2014). Why, when and how to use augmented reality agents (auras). *Virtual Real.* 18, 139–159. doi:10.1007/s10055-013-0239-4

Geronikolakis, E., Zikas, P., Kateros, S., Lydatakis, N., Georgiou, S., Kentros, M., et al. (2020). "A true ar authoring tool for interactive virtual museums," in *Visual computing for cultural heritage* (Springer), 225–242.

Guarese, R. L. M., and Maciel, A. (2019). "Development and usability analysis of a mixed reality gps navigation application for the microsoft hololens," in *Advances in computer graphics of image processing, computer vision, pattern recognition, and graphics*. Editors M. Gavrilova, J. Chang, and N. M. Thalmann (Cham: Springer International Publishing), 11542, 431–437. doi:10.1007/978-3-030-22514-8_41

Hammady, R., Ma, M., Strathern, C., and Mohamad, M. (2020). Design and development of a spatial mixed reality touring guide to the egyptian museum. *Multimed. Tools Appl.* 79, 3465–3494. doi:10.1007/s11042-019-08026-w

Horst, R., and Dörner, R. (2019). "Mining virtual reality nuggets: A pattern-based approach for creating virtual reality content based on microlearning methodology," in IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE).

Huang, B.-C., Hsu, J., Chu, E. T.-H., and Wu, H.-M. (2020). Arbin: Augmented reality based indoor navigation system. *Sensors* 20, 5890. doi:10.3390/s20205890

Kampa, A., and Spierling, U. (2017). "Smart authoring for location-based augmented reality storytelling applications," in *INFORMATIK 2017*. Editors M. Eibl and M. Gaedke (Bonn): Gesellschaft für Informatik), 915–922. doi:10.18420/in2017_93

Krauß, V., Boden, A., Oppermann, L., and Reiners, R. (2021). "Current practices, challenges, and design implications for collaborative ar/vr application development," in CHI Conference on Human Factors in Computing Systems (ACM), 1–15.

Li, W., Agrawala, M., and Salesin, D. (2004). Interactive image-based exploded view diagrams. *Proc. Graph. Interface* 2004, 203–212.

Liu, Y., Spierling, U., Rau, L., and Dörner, R. (2021). "Handheld vs. head-mounted ar interaction patterns for museums or guided tours," in *Intelligent Technologies for interactive entertainment*. Editors N. Shaghaghi, F. Lamberti, B. Beams, R. Shariatmadari, and A. Amer (Cham: Springer International Publishing), 229–242.

Luboschik, M., and Schumann, H. (2007). "Explode to explain-illustrative information visualization," in 2007 11th International Conference Information Visualization (IV'07) (IEEE), 301–307.

Microsoft (2021). Power apps. Availableat: https://docs.microsoft.com/en-us/powerapps/powerapps-overview.

Microsoft (2020). *What is the mixed reality toolkit*. Availableat: https://docs.microsoft.com/en-us/windows/mixed-reality/mrtk-unity.

Nebeling, M., and Speicher, M. (2018). "The trouble with augmented reality/virtual reality authoring tools," in 2018 IEEE international symposium on mixed and augmented reality adjunct (ISMAR-Adjunct IEEE), 333–337.

PTC (2021a). Ptc. Availableat: https://library.vuforia.com/area-targets/create-area-targetsl.

PTC (2021b). Ptc. Availableat: https://library.vuforia.com/features/environments/area-targets.html.

Rau, L., Döring, D. C., Horst, R., and Dörner, R. (2022). Pattern-based augmented reality authoring using different degrees of immersion: A learning nugget approach. *Front. Virtual Real.* 3. doi:10.3389/frvir.2022.841066

Rau, L., Horst, R., Liu, Y., and Dorner, R. (2021). "A nugget-based concept for creating augmented reality," in 2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct IEEE), 212–217. doi:10.1109/ISMAR-Adjunct54149.2021.00051

Singh, A., Kaur, R., Haltner, P., Peachey, M., Gonzalez-Franco, M., Malloch, J., et al. (2021). "Story creatar: A toolkit for spatially-adaptive augmented reality storytelling," in 2021 IEEE Virtual Reality and 3D User Interfaces (New York City, United States: VR), 713–722. doi:10.1109/VR50410.2021.00098

Spierling, U., Kampa, A., and Stöbener, K. (2016). "Magic equipment: Integrating digital narrative and interaction design in an augmented reality quest," in Proceedings of International Conference on Culture & Computer Science ICCCS, 25–28.

Tarantino, E., De Falco, I., and Scafuri, U. (2019). A mobile personalized tourist guide and its user evaluation. *Inf. Technol. Tour.* 21, 413–455. doi:10.1007/s40558-019-00150-

Unity Technologies (2021). Unity game engine. Availableat: https://unity.com/.

Vlizos, S., Sharamyeva, J.-A., and Kotsopoulos, K. (2021). "Interdisciplinary design of an educational applications development platform in a 3d environment focused on cultural heritage tourism," in *Emerging Technologies and the digital transformation of museums and heritage sites*. Editors M. Shehade and T. Stylianou-Lambert (Cham: Springer International Publishing), 79–96.

XRTY App (2021). *Create augmented reality marketing campaign without code*. Availableat: https://xrty.app/.