HIGH-LEVEL AUTHORING TOOLS IN AUGMENTED REALITY: ANALYSIS AND CASE STUDY

STUDENT: ROBERTA MOTA
SUPERVISOR: VERONICA TEICHRIEB
CO-SUPERVISOR: RAFAEL ROBERTO

RECIFE – BRAZIL, DECEMBER 15, 2014

FEDERAL UNIVERSITY OF PERNAMBUCO

CENTER OF INFORMATICS

UNDERGRADUATE PROGRAM IN COMPUTER SCIENCE

HIGH-LEVEL AUTHORING TOOLS IN AUGMENTED REALITY: ANALYSIS AND CASE STUDY

ROBERTA CABRAL MOTA
(rcm4@cin.ufpe.br)

A monograph submitted to the Center of Informatics of the Federal University of Pernambuco in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.

SUPERVISED BY VERONICA TEICHERIEB, PH.D.
CO-SUPERVISED BY RAFAEL ROBERTO, M.SC.

"*Witchcraft to the ignorant, simple science to the learned*"

- Leigh Douglass Brackett

ACKNOWLEDGEMENTS

Este trabalho é fruto de quatro anos dedicados à graduação em Ciência da Computação na UFPE. Assim, gostaria primeiramente de enfatizar que finalizo esta etapa com grande orgulho de ter sido estudante neste centro e nesta universidade. Ambos me proporcionaram a infraestrutura, a motivação e a orientação necessária para concluir a graduação nesta área com a certeza de que nada foi em vão. Ressalto particularmente a importância dos professores Sílvio Barros de Melo, Fernando Castor, Carlos Alexandre Barros de Melo e Paulo André da Silva Gonçalves. Não apenas agradeço a estes professores pelas aulas inspiradoras e pelos ensinamentos, mas também por lecionarem de forma apaixonada.

No Centro de Informática da UFPE, tive também a oportunidade de fazer parte de três projetos: o PET-Informática sob a supervisão do prof. Fernando da Fonseca de Souza, o BlackBerry Tech Center Recife sob a supervisão do prof. Cristiano Coelho de Araújo e, por fim, o Voxar Labs sob a supervisão da prof. Verônica Teichrieb. Agradeço aos três supervisores e grupos dos quais fui parte por todas as experiências proporcionadas.

À professora Verônica Teichrieb, particularmente, agradeço pela orientação nesta monografia. Durante suas aulas tive o primeiro contato com realidade aumentada, o que me impulsionou a querer fazer parte do Voxar Labs e a escrever a monografia a respeito deste tema. Ao doutorando Rafael Roberto, agradeço pela co-orientação, pelas conversas e pelos conselhos que muito influenciaram a estrutura deste trabalho.

Durante estes quatro anos de graduação, pude também contar com um apoio pessoal inestimável. Em nome deste apoio, agradeço o carinho e o amor da minha família. Um “obrigada” enorme pela convivência diária cheia de apoio a Fernando Mota, Fátima Guedes, Cristina Guedes e Renan Mota. Aos meus pais, agradeço particularmente por todos os esforços empreendidos no decorrer destes anos. Sei que ambos se dedicam aos filhos de forma integral e agradeço por sempre me encorajarem a buscar o melhor profissionalmente. Um “obrigada” pelos constantes incentivos a sempre crescer também à família que se tornou minha: Francisco, Rinalda e Marilia Ramos. A Francisco, agradeço pelo exemplo prático que tem sido e pela orientação na vida acadêmica e profissional. A Rinalda, agradeço pelo apoio incondicional e pelo carinho cuidado de mãe. Por fim, a Mariana, dedico meus agradecimentos pela paciência, companheirismo e, ainda, por dividir comigo de perto as alegrias e aflições desta graduação e dos últimos seis anos e quatro meses. A todos aqui citados, todo o meu amor.

ABSTRACT

Due to its recent developments, Augmented Reality (AR) technology started to be widely used in various application domains, such as advertising, medicine, education, robotics, entertainment, tourism, and others. However, the time and technical expertise needed to create AR applications has prevented widespread use. In this sense, authoring tools have become a largely used solution to boost mainstream use of AR, since they facilitate the development of AR experiences.

Augmented reality authoring tools can provide various levels of abstraction in application development, establishing different levels of interface complexity, concept abstractions, and technical skills demands. Particularly, those categorized as content design tools present a high abstraction layer, which substantially simplifies the user interface to a point that even people with few or no technical expertise can create and deploy AR content. Therefore, these tools are a critical component to the success and growth of augmented reality since they allow it to be explored by ordinary users, which have expertise only in their actuation areas.

Due to their increasing relevance, this monograph aims at conducting a trend analysis in order to understand the current tendencies in the area of content design tools: the authoring paradigms and the distribution strategies of AR solutions that have been used. For this purpose, both commercial and academic tools were analyzed concerning their dataflows, which describe the end-to-end scenario that outlines from the creation of AR semantic through authoring tools to its visualization by end-users. In turn, the identification of AR authoring and deployment trends allowed the translation of project-specific dataflows, observed in the analyzed content design tools, into the creation of general dataflow models. In this sense, a minimum number of combinations of trends were performed in order to elaborate four generic models in which all of the content design tools could fit into.

Furthermore, in order to investigate and illustrate all the theoretical foundations approached in this work, a case study was performed as a complement to this monograph. The main purpose was to analyze whether a high-level content design tool is, indeed, easy to use and permits a diverse audience to build AR applications. Hence, the Augie Studio tool was created and further evaluated by programmers and non-programmers. By measuring the amount of time spent to realize specific tasks using the tool, examining answers obtained from an evaluation questionnaire, and analyzing additional commentaries collected from a non-structured interview, it was confirmed that the tool meets its intended purpose.

Resumo

Devido às suas recentes evoluções, Realidade Aumentada (RA) começou a ser amplamente utilizada em vários domínios de aplicação, como publicidade, medicina, educação, robótica, entretenimento, turismo, dentre outros. No entanto, o tempo e conhecimentos técnicos necessários para criar aplicações de RA têm impedido o amplo uso desta tecnologia. Neste contexto, ferramentas de autoria tornaram-se uma solução bastante utilizada para impulsionar o uso generalizado de RA, uma vez que facilitam o desenvolvimento de experiências de realidade aumentada.

Ferramentas de autoria de RA podem fornecer várias camadas de abstração para o desenvolvimento de aplicações, estabelecendo diferentes níveis de complexidade de interface, abstrações de conceitos e habilidades técnicas requeridas. Particularmente, aquelas classificadas como ferramentas de design de conteúdo apresentam um alto grau de abstração, o qual simplifica substancialmente a interface de usuário a ponto de que mesmo pessoas com pouca ou nenhuma experiência de programação podem criar e distribuir conteúdo de RA. Portanto, essas ferramentas são um componente crítico para o sucesso e crescimento de RA uma vez que permitem que esta tecnologia seja explorada por usuários comuns, os quais têm *expertise* somente em suas áreas de atuação.

Devido à sua crescente relevância, esta monografia tem como objetivo a realização de uma análise de tendências a fim de entender os direcionamentos atuais no campo de ferramentas de design de conteúdo: os paradigmas de autoria e as estratégias de distribuição de soluções de RA que vêm sendo utilizadas. Para este propósito, tanto ferramentas comerciais quanto acadêmicas foram analisadas com relação aos seus *dataflows*, os quais descrevem um cenário de fim-a-fim que esboça desde a criação da semântica de RA, através de ferramentas de autoria, até a sua visualização por usuários finais. Por sua vez, a identificação de tendências em autoria e distribuição de RA permitiu a tradução de *dataflows* específicos de projeto, observados nas ferramentas de design de conteúdo analisadas, na criação de modelos genéricos de *dataflow*. Neste sentido, foi realizado um número mínimo de combinações entre as tendências identificadas de modo a elaborar quatro modelos genéricos, nos quais todas as ferramentas de design de conteúdo poderiam enquadrar-se.

Além disso, na intenção de investigar e ilustrar todos os fundamentos teóricos abordados neste trabalho, um estudo do caso foi realizado como complemento a esta monografia. O principal objetivo deste estudo foi analisar se uma ferramenta de design de conteúdo de alto nível é, de fato, fácil de usar e permite que um público diversificado possa construir aplicações de RA. Neste contexto, a ferramenta Augie Studio foi implementada e avaliada por programadores e não programadores. Ao medir a quantidade de tempo gasto para realizar tarefas específicas usando a ferramenta, examinar respostas obtidas a partir de um questionário de avaliação e analizar comentários adicionais recolhidos em uma entrevista não estruturada, confirmou-se que a ferramenta atende a sua finalidade.

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Figure 1: augmented reality has been applied in a number of domain areas. Yet, a critical component for its advancement is allowing non-technologists to engage directly with the AR creation medium, with the goal of accelerating its adoption and evolution.

Augmented Reality (AR) supplements the physical world with virtual content that appears to coexist in the same space as the real world. Figure 1 illustrates an example. Thus, it enhances the user's perception of and interaction with the real environment. The AR technology has existed for just over two decades, but the growth and progress in the past years has been remarkable [**1**].

In this sense, the increasing advances of both hardware and software have been allowing for richer experiences in augmented reality. For instance, the creation of mobile devices with stronger processing power, graphic resources, and a mix of sensors have made possible the diffusion of applications for mobile AR. Also, recent improvements in tracking and visualization techniques have enabled the creation of applications that combine information from multiple sources, such as camera, compass, gyroscope and GPS, to display AR contents based on those wellspring, such as geo-location applications [**2**]. As a result of these recent developments, AR has been applied in several domain areas, including advertising, medicine, education, robotics, entertainment, tourism, and others [**3**].

The increasing group of application domains resulted in the adoption of AR by the general public, thus including marketers, designers, doctors, teachers, game and web developers. Hence, there has been a gradual demand for tools that can facilitate the development of AR applications in the sense of making this technology accessible to this diverse audience [**4**]. In this context, AR authoring tools comprise a wide range of software products with capabilities for composing, editing, and managing AR experiences.

AR authoring tools can be classified according to their programming characteristics and content design from low to high-level approaches. Therefore, distinctive authoring approaches have different concept abstractions

and interface complexity, and hence address audiences that do not necessarily have the same technical expertise [**5**]. Still, it is important to observe that authoring approaches are constructed hierarchically: abstraction is added gradually, while low-level features and concepts are hidden.

Among the approaches of AR authoring tools, it is important to note the relevance of those categorized as content design tools. They completely remove the dependency of knowing a programming language, replacing it for graphical user interfaces to describe the virtual content and its relationship with the real scene. These tools are particularly relevant because they leverage the widespread adoption of AR, since they highly simplify the authoring process and allow the development of applications and content by ordinary users, which have expertise only in their application areas.

1.1. Purpose

In this sense, this monograph presents the findings from a trend analysis conducted in the field of content design authoring tools. This investigation attempted at understanding the current patterns regarding the authoring paradigms and deployment strategies of AR experiences that have been used in both commercial and academic realms.

Also, this monograph presents a case study in which a high-level content design tool was implemented and an usability test was performed in order to evaluate whether this category of authoring tools are, indeed, easy to use and allow a diverse public to create AR solutions, including people with no technical expertise.

1.2. Monograph Outline

The remainder of this monograph will proceed as follows. Chapter 2 provides an overview of the field of AR and demonstrates how this technology has been applied in different application domains. Chapter 3 approaches authoring tools and their relevance in the AR field. It also defines different categories of AR authoring tools and gives illustrative examples, focusing on those categorized as content design tools.

The following two chapters approach the purpose of this monograph, which was previously mentioned. Chapter 4 describes the methodology and results obtained from the tendency analysis performed in the field of content design tools. Chapter 5 approaches the case study by describing the development of a high-level content design tool and, also, the methodology and results collected from usability testing.

Chapter 6, finally, concludes the monograph and discusses future works focused on improving the ease of use of the implemented authoring tool, mentioned in the previous chapter.

2. Augmented Reality

Figure 2: the iOnRoad application measures the vehicle's headway distance, using Augmented Reality to alert the driver about his speed and proximity to the traffic ahead.

Augmented Reality supplements reality by displaying consistently virtual elements with real ones, creating a mixed environment where the user can interact in real time with both the real and virtual world [**6**]. The superimposed virtual imagery, sound or other sensorial enrichments upon the real world enhances user's perception of and interaction with the real world.

In order to illustrate this concept, iOnRoad [**7**] is a mobile application which strives to help prevent driving collisions. It alerts drivers when they are too close to the car in front or when they are straying outside their lane. Firing up iOnRoad starts the phone camera to show the surrounding area on the display. A green overlay outlines the lane that the user is currently driving in and an icon indicates that the car ahead had been detected, as can be seen in Figure 2. Within that icon, either the distance, reaction time, or vehicle speed can be displayed in numerical form. If the time gap between the user and the car ahead drops below 0.5 second, an audible chime sounds from the speaker and a text warning alert appears on the phone screen.

Additionally, AR is not limited to the sense of sight [**6**]. Indeed, despite visual augmented reality be the most common type of AR, this technology can potentially be used to all senses, augmenting smell, touch, taste, and hearing as well.

One example suggests AR to be applied to augment the sight of poor vision users by the use of audio cues. Figure 3 illustrates a portable device that helps vision-impaired people to "read", named OrCam [**8**]. A tiny eyeglasses-mounted camera performs text recognition in real time and provides audio response to the user through a bone-conduction earpiece. The OrCam device responds to a finger-pointing gesture to a specific item to be identified, such as product labels, newspaper text, bus numbers, among others. For this purpose, live video from the camera is processed and an audio snippet translating the detected text is sent to the bone-conduction speaker in the user's ear.

Furthermore, [**6**] also considers diminished reality as a subset of AR. This is not surprising

as removing an object from the real world corresponds to covering it with virtual information that matches the background, thus giving the user the impression that the object is not there.

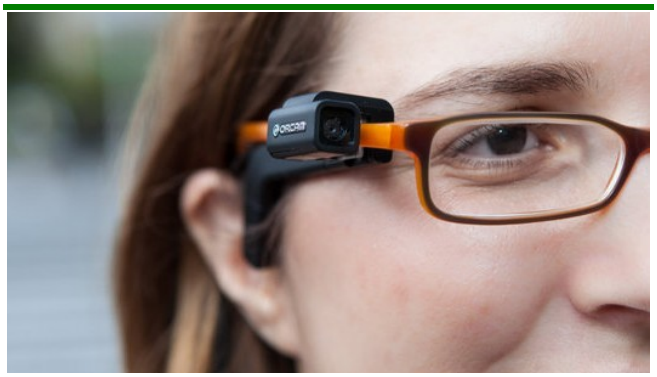


Figure 3: OrCam is a novel assistance device for visually impaired people. It is a camera-based system intended to give the user the ability to both "read" and move freely.

An augmented reality system uses different technologies depending on its intended application. As can be observed in the two previously mentioned examples, the OrCam and iOnRoad systems greatly differ in terms of technologies being used to compose each AR application. The former uses an unobtrusive camera and bone-conduction speaker attached to eyeglasses. It also responds to simple finger-pointing gesture with audio cues. The latter utilizes the smartphone's native camera and sensors to detect traffic ahead and audiovisual warnings pop up whenever the driver exceeds a minimum safe proximity of the vehicle in front. However, it is possible to outline three basic components needed to build compelling AR environments.

One major component is a precise tracking system which provides accurate environmental information to keep virtual elements registered with the real ones. In the iOnRoad app, for example, the tracking solution is responsible for detecting vehicles ahead, continuously monitoring the headway distance from the car, and calculating the correct locations of the green lane and icon overlays at the real scene.

Another required component is the use of a display for viewing the merged virtual and real environments. There are a number of different display hardware which can be used in AR applications. In this sense, regular monitors and mobile device screens offer a fairly low-cost alternative. Therefore, they are the most common platforms used to visualize augmented content.

Finally, one important aspect of augmented reality is to create appropriate techniques for intuitive interaction between the user and the virtual content of AR applications in real time. For example, the OrCam system responds whenever the user points his finger to a specific item to be recognized.

In summary, to provide an effective AR experience three factors must be developed: a tracking technique of the real scene, display hardware for viewing virtual elements over the real world, and interaction techniques so the user can manipulate the AR virtual content. This chapter details each of these topics. It also demonstrates how AR technology can be used in different domains by describing some AR applications.

2.1. Tracking Techniques

Tracking, also known as registration, is the component responsible for "understanding" the surrounding environment so virtual elements can be coherently inserted into the real world.

According to a research conducted by [**9**], until 2008, tracking had been the most popular topic for research in ten-year development of ISMAR international academic conference. This result perceives tracking as one of the most challenging technologies for AR.

According to the authors, tracking technologies can be grouped into three major categories: sensor-based, vision-based, and hybrid tracking techniques. Figure 4 presents a generic taxonomy for the available augmented reality tracking techniques.

2.1.1. Sensor-based Tracking Techniques

Sensor-based tracking techniques are based on sensors such as magnetic, acoustic, inertial, optical and mechanical sensors [**9**]. Each of these technologies has different levels of accuracy, advantages, weaknesses, and depends greatly on the type of system being developed. For example, magnetic sensors are lightweighted, low-cost, and have a high update rate; however, these sensors suffer in terms of jitter and the accuracy degrades with distance and electromagnetic noise.

There have been a significant number of researches published on tracking using non-camera based sensors. One example is the work of [**10**], whose ultrasonic tracking technique is used to provide a wide area indoor tracking. In this system, the user wears ultrasound emitters and sensors are fixed in the environment. Based on the period of time taken for sound to reach the sensors, the position and orientation of the user is calculated.

2.1.2. Vision-based Tracking Techniques

Vision-based tracking techniques use computer vision methods to calculate the camera pose relative to real world objects, with which augmented content can be coherently added to the real scene.

According to [**11**], most of the available tracking techniques can be divided into two classes: marker-based and markerless tracking. In the former case, visual markers are used to estimate the camera pose. In the latter case, features of the real environment are used to perform this computation.

2.1.2.1. Marker-based Tracking

Marker-based systems utilize artificial markers, which are elements with previously known information to estimate camera pose. This tracking approach reduces the requirements for computation while provides robust solutions. Additionally, marker-based systems are low-cost, easy to implement, and a number of well-known marker-based toolkits are available.

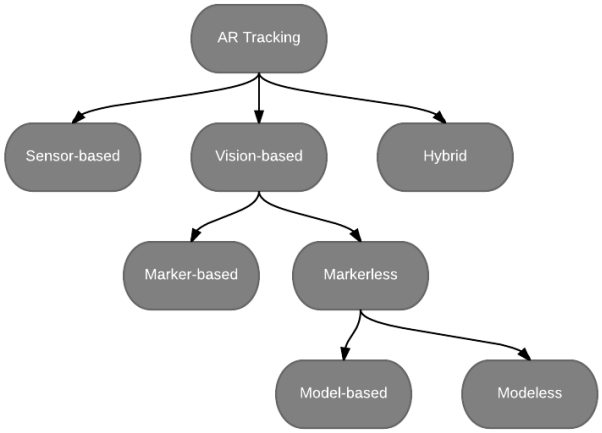


Figure 4: the infographic above presents a general taxonomy for augmented reality tracking techniques.

In this sense, marker-based approaches gained popularity and several types of markers were created to meet specific needs. Some common types include template-based, ID-based, DataMatrix, split and frame markers [**12**].

2.1.2.2. Markerless Tracking

Marker-based augmented reality systems require intrusive artificial markers to be placed at the environment. Additionally, this tracking approach is not suitable in large scale outdoor environments.

Rather than using visual markers, camera pose can also be determined from naturally occurring features, such as points, edges, corners and textures. In this sense, markerless tracking utilizes the surrounding environment as a marker.

One major advantage of using this approach is the increase in immersion degree in a manner closer to the one imagined since the conception of Augmented Reality [**13**]. However, markerless techniques are much more complex and demand higher computational power.

According to [**13**], techniques developed for real-time markerless augmented reality systems can be broken down into two categories: model-based and modeless. In the former case, the system utilizes a three-dimensional model of (part of) the scene. It deduces camera pose

based on correspondences between the model and the environment. In the latter case, the camera movement is estimated without any prior knowledge about the scene.

2.1.3. Hybrid Tracking Techniques

Hybrid tracking techniques combine both sensor and vision-based tracking. They use multiple measurements to produce robust tracking solutions. Typically, AR applications employ hybrid techniques to exploit strengths and compensate weaknesses of each tracking technology.

For example, [**2**] use a sensor-fusion-based approach for tracking in outdoor environments. It combines panorama-based vision tracking, accelerometer and compass data to track rotational movements, whereas the GPS sensor is used to estimate the current user position. Particularly, the panorama-based tracking and the sensors were fused as an attempt to combine the accuracy and robustness of vision tracking and absolute orientation of the inertial and magnetic sensors.

2.2. Display Techniques

Along with an accurate tracking system, consistent display hardware is also an essential factor to provide an immersive and realistic augmented reality experience.

According to [**9**], the augmented view of the world can be presented to the user via three major types of display technologies: head mounted displays (HMDs), handheld displays and spatial displays.

2.2.1. See-through HMDs

See-through HMDs require the user to wear the display system on his head. They can be fundamentally divided into two categories: optical see-through (OST) and video see-through (VST). Both systems have two image sources, the real world and the computer-generated world, which are to be merged.

As can be seen in Figure 5b, optical see-through displays take what might be called a "minimally obtrusive" approach. In other words, they allow the user to see the real world with their natural eyes and attempt to augment it by merging a reflected image of the computer-generated scene into the view of the real world. The OST display overlays virtual content onto the user's view by using optical combiners, such as half-silvered mirrors, transparent LCD displays, or similar technology [**14**].

Video see-through displays are typically more obtrusive in the sense that they block out the real-world view in exchange for the ability to merge the real and computer-generated views more convincingly, as illustrated in Figure 5a. Therefore, one advantage of VST HMDs is consistency between real and synthetic views. However, they are more demanding than OST systems as they require the user to wear two cameras on his head and require the processing of both cameras to provide both the real and augmented views with unmatched resolution [**15**].

Some technical issues for HMD systems include (i) difficulty in supporting multiple users, (ii) cumbersome when users have to wear them continuously after a long time, (iii) adverse effects such as eyestrain and nausea, and (iv) the latency of the system. The latter case, in particular, addresses a gap in time, referred to as "lag", between the moment when the HMD position is measured and the moment when the virtual image for that position is fully rendered to the user. During this period of time, the user can move his head and, thus, the discrepancy can destroy the main goal of AR, which is to keep users from perceiving the difference between the real world and the virtual augmentation of it.

Figure 5: (a) Visette 45 SXGA [16] is a video see-through HMD and (b) Vuzix Wrap 920AR eyewear [17] is an optical see-through HMD.

2.2.2. Handheld Displays

Handheld displays employ small computing devices with a display that the user can hold in their hands. Particularly, recent advancements in mobile phones present a versatile platform with enough processing power for tracking operations, graphics capabilities for 3D rendering, and a combination of sensors, such as build-in camera, GPS, accelerometer and magnetometer. Additionally, smartphones and tablets have become an inexpensive technology and they are being produced for a massive market. This scenario ensures a large-scale number of users and broad geographic coverage. Therefore, smartphones and tablets are considered a very promising platform for AR.

As an example, a recent commercial application called Nokia City Lens [**18**] uses GPS, compass and gyroscope information to present location-based AR content on mobile devices, such as restaurants, museums, and hotels, as can be observed in Figure 6.

Unfortunately, smartphones have small display sizes, which are not ideal for 3D user interfaces. In comparison with HMDs, these devices provide less degree of immersion. Furthermore, mobile phones can present a limited field of view (FOV), and so the user may need to move the device in order to visualize the augmented scene.

Figure 6: Nokia City Lens application incorporates GPS-tagged content rendered using screen-aligned icons.

2.2.3. Spatial Displays

In contrast to body-attached displays (head-attached and hand-held), spatial displays detach most of the technology from the user. They present the computer-generated content directly onto physical objects without requiring the user to wear or carry the display [**15**]**.**

There are different spatial approaches, which mainly differ in the way they augment the environment: video see-through, optical see-through and direct augmentation.

2.2.3.1. Screen-based Video See-through Displays

As can be observed in Figure 7, this approach makes use of video-mixing and displays the augmented scene on a regular monitor [**14**]. Screen-based augmented reality is sometimes referred to as a *window on the world*, since it may create the impression that the user observes the augmented world through a "window" – which is the traditional monitor [**19**].

In the case of screen-based displays, it represents a cost efficient AR approach, since only off-the-shelf hardware components and standard PC equipment is required [**14**]. However, one major disadvantage is the limited field of view, which is restricted to the monitor size. As a result, this approach provides a low degree of immersion.

Figure 7: a screen-based display terminal being used as a virtual fitting room.

2.2.3.2. Spatial Optical See-through Displays

As illustrated in Figure 8, similarly to head-attached OST displays, spatial OST displays enable the user to directly see the real scene and visualize the virtual content through an optical surface, known as *optical combiner*. Optical combiners are typically transparent screens and half-silvered mirror beam combiners. These components are responsible for mixing the real environment with an image source that displays the rendered graphics [**14**].

Figure 8: Microsoft HoloDesk [20] is a spatial optical see-through display. A virtual image of a 3D scene is rendered through a silvered mirror and spatially aligned with the real-world for the viewer.

Note that, spatial optical see-through displays do not follow the entire user body's movement. Therefore, they are comparable with spatial projection-based displays, but do not share the opaque characteristic of such displays.

The general advantages of these systems are easier eye accommodation, larger and scalable FOV. However, they do not support mobile applications because of the spatially aligned optics and display technology. Furthermore, due to the limited size of the optical combiner, virtual objects that are outside of the display area can be unnaturally cropped.

2.2.3.3. Projection-based Spatial Displays

In projector-based augmentation, the user's physical environment is augmented with images projected directly onto physical object's surfaces. Note that the projection surface does not need to be neither a rectangular plane nor planar, as exemplified in Figure 9. Another compelling example is an architect augmenting a tabletop scaled model of a building using a projector. The augmented model could have virtual objects, such as doors, windows, and chimneys. It could also show underground water pipes or another support structure inside the building [**14**].

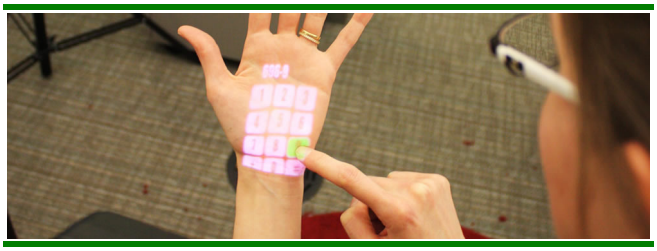


Figure 9: Microsoft OmniTouch [21] utilizes a body-worn projection and sensing system. It allows users to manipulate interfaces projected onto the environment, held objects and their own bodies.

Some benefits achieved by the use of projection-based displays include (i) easy eye accommodation, since the observer does not need to switch focus between the image plane and the real environment, (ii) the capability of generate images that are larger than the actual display device and (iii) create augmentation visible to several users. However, one main disadvantage of these systems is that they commonly lack mobility, since the setups for most projection-based displays are fixed to the surface for which they are installed.

2.3. User Interfaces and Interaction Techniques

Formerly, most AR prototypes development effort was concentrated on displaying information that was registered with the real world and did not concern with how users would

interact with these systems [**1**]. Meanwhile, creating appropriate techniques for intuitive interaction between the user and the virtual content of AR applications is becoming one essential aspect of augmented reality.

According to [**9**], there are three main ways of interaction in AR applications: tangible, collaborative, and hybrid AR interfaces.

2.3.1. Tangible AR Interfaces

The concept of tangible user interfaces (TUIs) denotes interfaces where physical objects are used to represent and control digital information [**22**].

The main advantage of this interaction approach is that using manageable objects as keywords eliminates the language barrier of conventional graphical interfaces. Thus, tangible interfaces can be extremely intuitive, as the user has an inherent knowledge of the manipulated objects. Furthermore, the tangible approach can also provide a more interactive environment, support collaborative activities, and two-handed interactions.

This same TUI metaphor can be applied to AR interfaces in order to combine the intuitiveness of physical input devices with enhanced display possibilities provided by virtual overlays. For example, Letters Alive is a supplemental reading program [**23**]. It includes flash cards of animal and word designed to encourage students to create sentences, as seen in Figure 10. Meanwhile, AR technology prompts the animals to answer and react to student's questions and statements formed by manipulating the cards.

2.3.2. Collaborative AR Interfaces

Although single user AR applications were studied for decades, there have been a number of researches focusing on collaborative AR applications. The main goals of these works are exploring AR technology to support remote and co-located activities.

Figure 10: in Letters alive application, as children form words or sentences with physical cards, three-dimensional animals respond to student's actions and answer their questions.

For co-located collaboration, also known as face-to-face collaboration, AR can be used to enhance a shared physical workspace [**24**]. For example, [**25**] described an AR tennis game as an early collaborative AR application for mobile phones. When the players point their cameras at ARToolkit markers, they see a virtual tennis court superimposed over the real world. The players move their phones to hit the virtual ball to each other. Each time the ball is hit, a sound is played and the phone vibrates, providing multi-sensory cues to help the players.

For remote collaboration, AR can enhance communication systems. For example, [**26**] constructed an infrastructure to facilitate communication of situational and navigational information between outdoor and indoor users. In this system, indoor users can place physical objects as well as portions of their bodies onto a tabletop projected display. Physical objects can be captured as the table surface is continuously scanned by a series of cameras, as can be observed in Figure 11. Next, the data is sent over a wireless network and is reconstructed at a real world location for outdoor users, which are equipped with mobile AR systems.

2.3.3. Hybrid AR Interfaces

Hybrid interfaces combine an assortment of heterogeneous, but complementary, displays with which users interact through a potentially equally varied range of interaction devices. Furthermore, these systems may be used in unplanned environments, where it is not known

in advance which type of interaction techniques and devices will be used [**15**]. Therefore, they must provide a flexible infrastructure to automatically accommodate a changing set of input and output device and the interaction techniques with which they are used.

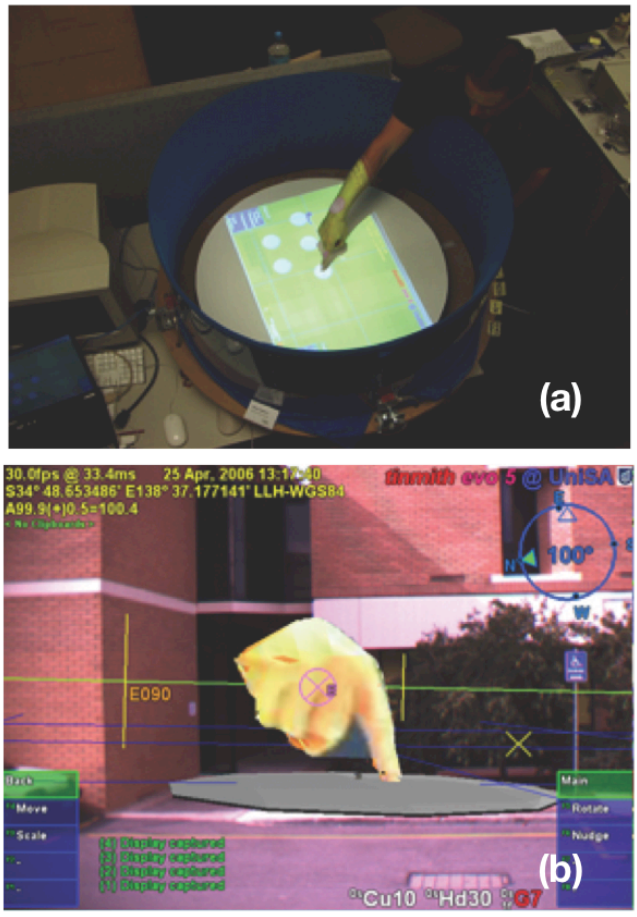


Figure 11: (a) indoor users provide navigational instructions by simply pointing to an area on the table surface and (b) the outdoor users, equipped with mobile AR systems, see the changes in real-time.

As an example, [**27**] constructed a hybrid interface in which a user interacts with different physical input devices and virtual objects are drawn on several desktop monitors. Each input device performs a simple transformation (translation, rotation and scale) in one specific three-dimensional object. The user handles a tracked wand with which he can reconfigure these three-element-based relationships: an input device, a virtual object, and a transform operation. Additionally, the user can use a head-tracked, see-through, head worn display to visualize augmented reality overlays related to the system's current connections.

2.4. Application Domains

Augmented Reality has existed for almost twenty years and early researches were mainly concerned to establish the technical aspects. Since then, AR technology has begun to mature as advancements were made in hardware, software, and techniques capabilities and costs. These improvements reached a level where feasible AR experiences can be delivered. As a result, in a 20-year period, AR has gone from being viewed as a restricted technology, only appropriate for industrial and military applications, to a wide range of domain areas, including advertising, medicine, manufacturing, aeronautics, robotics, entertainment, tourism, social network and education [**3**].

Augmented reality is widely used for advertising and commercial purposes. It enables marketers to reach shoppers via outdoors ads, billboards, magazine, newspapers, among others communication channels. AR technology has become an advertising trend, since it can attract, retain and engage consumers through immersive experiences. For instance, in summer of 2013, IKEA launched their augmented reality catalogue [**28**] to enable consumers to visualize how certain pieces of furniture could look inside their home, as seen in Figure 12. Also, the app measures the size of the products against the surrounding room to offer a realistic experience.

Figure 12: using smartphones or tablets, users can view and place selected 3D representations of IKEA products in their own rooms.

There are also AR applications for entertainment or educational purposes. For example, the ColAR Mix [**29**] is an augmented reality coloring app for children. First, some

picture drawings are downloaded and printed out from their website. Next, parents can take the printed pages to their children and let them paint the pictures with regular crayons. Then, with the app running and hold above the drawing, the colored pictures turn into three-dimensional models. These models are animated and have the exact color and details that the students put on their coloring drawings, as can be seen in Figure 13.

Figure 13: ColAR Mix turns colorized pages into hand-animated 3D worlds.

Another example is the ARBlocks, a dynamic blocks platform for educational activities based on projective augmented reality and tangible user interfaces [**30**]. The blocks are the projection surface in which various multimedia content can be presented. Different applications can be built for the ARBlocks platform based on the teacher needs. For example, one application may request the child to manipulate the blocks to combine letters and form a word related to an image, as can be seen in Figure 14. In another application, the students may have to move the blocks following specific movements. ARBlocks also offers visual feedbacks to guide students during the activities.

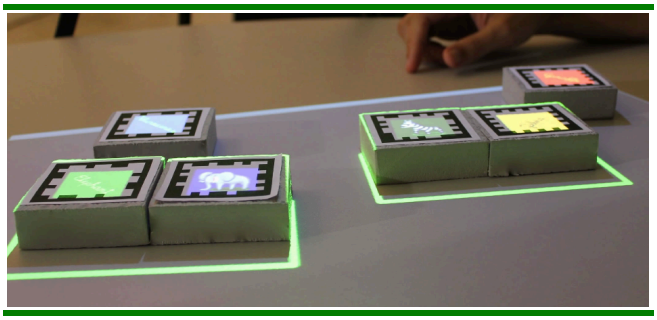


Figure 14: in ARBlocks, the content is projected on the blocks empty area. The green line denotes a positive feedback when the activity is properly executed.

AR technology has also started to be incorporated into games. This type of games provides interesting gameplay options, allowing a direct interaction rather than using controllers or keyboards. A novel example is Ingress, an immersive, geo-location based augmented reality game [**31**]. In this game, the players are divided into two factions battling to collect and control a powerful 'Exotic Matter', which is virtually located in real-life locations worldwide, as illustrated in Figure 15.

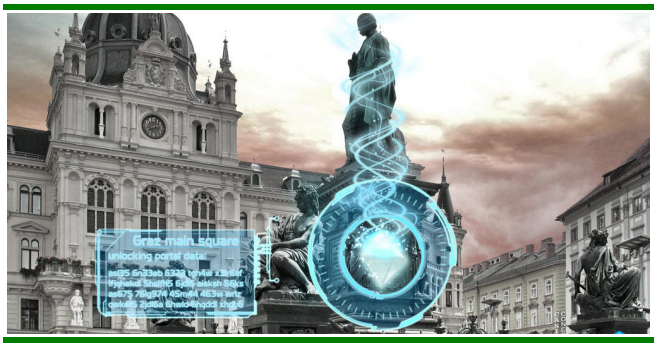


Figure 15: the Ingress mobile game uses augmented reality and GPS sensor. It enables players to see invisible portals and other virtual structures and artifacts overlaid on the real world.

In order to leverage a widespread usage of AR, companies have been proposing an equivalent of desktop or mobile Web browsers for the physical world, commonly referenced as AR browsers, in the smartphone marketplace. AR browsers like Junaio [**32**], Layar [**33**], and Wikitude [**34**] permit users to publish and access GPS-tagged augmented interactive content on the real world. They may also link virtual content to physical objects by using computer vision-based recognition and tracking techniques, as can be seen in Figure 16.

Figure 16: Junaio is a commercial AR browser application, which allows users to browse digital information connected to real products, locations, newspapers and billboards.

AR browsers is a promising approach since mobility advancements offered the possibility to deploy AR applications in a global scale by using a low-cost platform that combines AR display, tracking and processing. Additionally, this type of application is today downloaded or preinstalled on more than 50 million smartphones [**35**]. This search data, in turn, shows how the interest towards AR browsers has been growing.

Similarly to augmented reality browsers, authoring tools have also become a largely used solution to increase social acceptance and usage of AR. They attempt to support improving AR technologies, and to encourage AR development in different domain applications.

3. Authoring Tools in Augmented Reality

Since the first steps in the 1990s, augmented reality has undergone considerable advances in AR research and adoption by the growing community of users [**4**]. In this sense, both academia and industry now believe that there is a huge potential for AR technology in a broad range of areas. Therefore, more researchers and companies are developing solutions focused on the mainstream adoption of AR technology.

For example, the previously mentioned AR browsers are an attempt to arrange AR as a noticeable player by building a structured software platform and a gateway to an anywhere AR user experience. Today the AR Browser is being used mainly to annotate objects and environments with public or private information. Taking into account the number of publicly available databases an AR Browser can draw upon, it is possible to see how useful this tool could be for several situations. In fact, as mentioned in [**2**], mobile AR is now experienced primarily through AR browsers, which achieved million downloads from mobile app stores, and some are even preinstalled on smartphones.

In addition, companies and businesses are continuously trying to incorporate smart-glasses into industrial and consumer markets. These displays provide most of the available features and capabilities of a modern smartphone in a hands-free wearable device. For example, Google's Glass [**36**] allows users to either use voice commands or simple hand gestures for operating controls, such as to capture video, share contents in social networks, and run third-party applications. Furthermore, the sound creation takes place with the help of bone-induction speakers, which is considered as less abrasive as compared to headphones. Other brand releases include DAQRI Smart Helmet [**37**], Epson's Moverio BT-200 [**38**], ORA by Optinvent [**39**], and others.

These devices offer extended human-computer interfaces, cutting edge technologies, and a new range of applications. As a result, organizations are investing in smart-glasses to be overtaken in terms of popularity, thus reinforcing mainstream AR. In fact, a report by Juniper Research predicts that AR users worldwide will increase from 60 million to 200 million over the next five years. This estimative is based on the expectation that Google Glass will take off [**40**].

In a similar way, authoring tools have also become a largely used solution to boost mainstream use of AR. In this sense, this chapter describes AR authoring tools and how they may leverage the evolution and success of augmented reality. It also conceptually categorizes different AR authoring approaches by providing a generic taxonomy.

3.1. Definitions, Drifts and Challenges

According to [**41**], the term *authoring tool* denotes a wide range of software products with capabilities for composing, editing, and managing software or digital content. Moreover, the term *authoring system* refers to a subset of these products, which generally provides high-level visual tools that enable content development without writing any programming code.

In this sense, design frameworks like Twitter Bootstrap [**42**] and Zurb Foundation [**43**] are web authoring tools, which allow developers to code fully functional web templates. These frameworks include a collection of CSS and HTML components to be added into the webpage source code, such as forms, buttons, tables, and stylish typography. On the other hand, Adobe products like InDesign [**44**] and Muse [**45**] are authoring systems designed as a way for graphic designers to build websites without touching code, as seen in Figure 17.

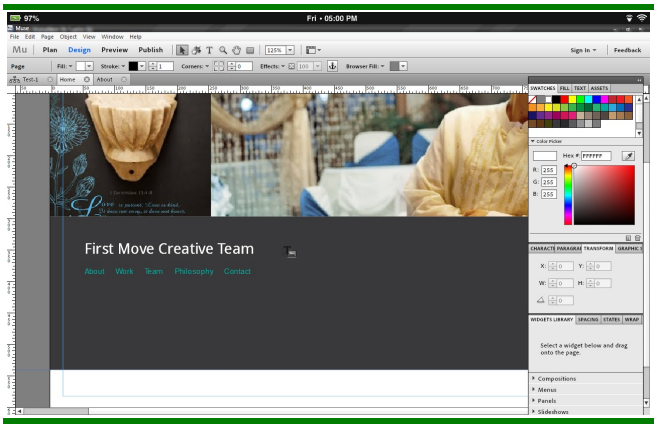


Figure 17: Adobe Muse is an authoring system in which graphical elements can be dropped into a blank canvas editor, such as texts, images, slideshows, boxes, etc.

Similarly, JavaScript libraries like Tracking.js [**46**] and Headtrackr [**47**] are AR authoring tools, which provide different computer vision algorithms and tracking techniques to help developers implement AR applications. On the other hand, Layar Creator [**48**] is an online authoring tool that enables the creation of AR content with no programming skills requirements. As can be observed in Figure 18, the author simply uploads trackable images and drags-and-drops pre-made buttons onto them. These buttons act as interactive augmented content, which perform different actions whenever clicked such as opening Youtube video, sharing content through social media, calling a predefined number, and others.

It is important to note that, for simplicity reasons, in this work both nomenclatures are indistinctly referenced as AR authoring tools regardless the need for programming skills.

Authoring tools are critical to the success and growth of AR. Despite the advancements reached by its technology, there are still issues to be addressed to ensure a wider use of AR. In this sense, authoring tools may (i) support research of AR to overcome technological limitations and (ii) facilitate the development of AR solutions. Each of these topics is detailed below.

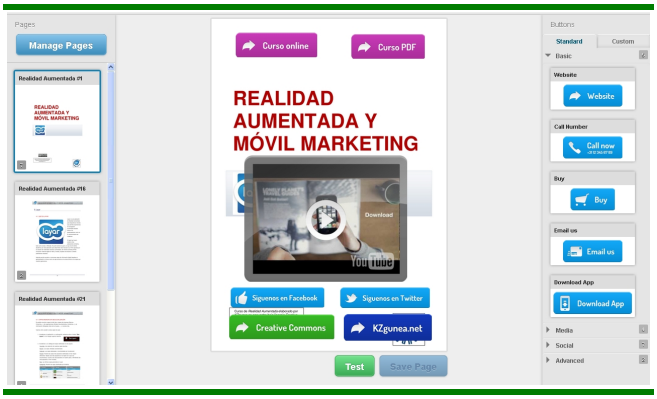


Figure 18: Layar Creator is an AR authoring tool in which the user uploads trackable images (shown on the left panel) and drops digital buttons onto them. These buttons, in turn, can be chosen from the right panel to act as interactive augmented content.

3.1.1. Research Maturation

There are still low-level software and hardware limitations that need to be overcome so they can be able to guarantee that AR technology reaches a high level of maturity, thus delivering compelling AR experiences. These technical challenges comprise a broad range of AR research fields, such as performance, mobility, and alignment concerning the proper placement of virtual content to real world objects. As a result, researchers developed programming authoring tools focused on supporting AR research goals.

For example, ARToolkitPlus (ARTK+) [**49**] is based on ARToolkit (ARTK) [**50**], which is a software library used to calculate camera position and orientation relative to physical black-square markers in real time. The ARToolkitPlus has an optimized pose tracking library for the usage on mobile devices. It also extended ARToolkit's vision algorithm with features such as automatic thresholding, a more stable pose estimation algorithm, and the

possibility to switch to ID-encoded marker detection instead of the built-in template matching.

In turn, Studierstube Tracker [**51**] is a successor to the ARToolkitPlus library. It includes many different marker types, such as template, ID, frame, and grid markers. It also includes performance improvements: its memory requirements are very slow (it is about twice as fast as ARTK+), its processing is very fast (it is only 5-10% of the memory usage of the ARTK+), and while ARTK+ follows a monolithic approach, Studierstube Tracker is highly modular.

3.1.2. Application Building

In addition, it is important to observe that authoring tools are commonly used to simplify the development of AR applications and content. In this sense, they may provide software infrastructure addressing capabilities such as tracking, graphics, and interaction techniques.

For example, the previous mentioned Tracking.js, Headtrackr, ARTK, ARTK+, and Studierstube Tracker support programmers to create augmented reality solutions. In this sense, they provide a tracking infrastructure, comprised by computer vision algorithms, to calculate the camera pose relative to physical markers.

Also, other tools may provide further features, such as Metaio SDK [**52**], which offers advanced tracking, rendering, and interaction infrastructures, including tracking of 3D objects and environments, built-in renderer of three-dimensional geometries, and multi-touch gestures sensor support.

All of the tools mentioned above facilitate developers to deploy AR applications. However, they require the application developer to work at a fairly low level, with languages like C or C++ [**53**], and are targeted at AR technologists. Despite of that, the recent exploration of application domains has prompted the adoption of AR applications by the general public. AR has currently been used in marketing, maintenance, games, education, training, architecture, home shopping and other application domains.

Hence, the AR operation field is now made up of all types of researchers and general public [**4**]. As a result, there is an increased demand for making these technologies accessible to a diverse audience, with the goal of supporting AR experience design and development that fully address the needs of these non-technologists. Therefore, recent AR projects also confronted authoring from the other end of the design spectrum, creating environments to simplify the creation of AR systems with minimal or no programming.

For example, the previous mentioned Layar Creator fits this authoring paradigm, since it enables the creation of AR content through a graphical user interface, a drag-and-drop mechanism, and no coding skills requirements.

3.2. Authoring Approaches

AR authoring tools can provide different levels of abstraction in application development and the correlated amount of technical expertise required to deal with the tasks at each of these levels. Similar to conventional digital content creation, a trade-off had to be made between low-level, programming driven systems and high-level content based systems, affecting various aspects of the actual application product [**5**]. Figure 19 illustrates a taxonomy of development tools for AR applications. This theoretical view establishes different levels of concept abstractions, interface complexity, and technical skills demands by predefining underlying concepts.

Development tools for building AR applications can be broadly organized into two different approaches: AR authoring for programmers and non-programmers [**54**]. In the former case, tools are typically code libraries that require programming knowledge to author the application. In this work, we address this approach as *programming tools*. In the latter

case, abstraction is added and low level programming capability is removed or hidden. Thus, tools for non-programmers are content driven and commonly include graphical user interfaces for building applications without writing any lines of code. Here, we address this approach as *content design tools*.

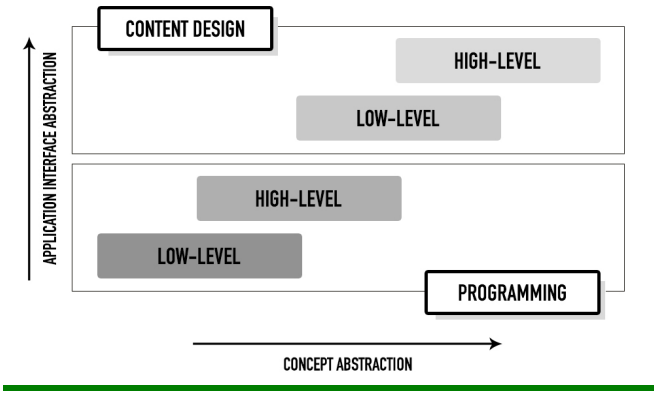


Figure 19: schematic view on digital media authoring.

These generic categories can be further organized into low-level and high-level application builder tools. Low-level tools require coding or scripting skills, while high-level ones use high-level libraries or visual authoring techniques.

Note that, all of these authoring approaches are built upon each other. Abstraction is gradually added and low-level functionalities and concepts are removed or hidden. Also, different abstraction levels address different target audiences with different technical expertise. Based on this general context, these four different approaches on digital media authoring can be conceptually described as follows.

3.2.1. Low-Level Programming Tools

Low-level programming tools consist of APIs that provide core tasks, such as computer vision based tracking, visual and spatial registration of objects, and generic 3D rendering. Their thin level of abstraction yields a high degree of performance and flexibility. On the other hand, it requires the developer to manually define the interaction techniques, visualization and simulation aspects [**5**].

An example of a low-level programming tool is the previous mentioned ARToolkit. This open source library provides a set of features focused on specific tasks in AR, such as vision based tracking of black square markers. However, to develop with ARToolkit requires further code for 3D content loading, interaction techniques and other utility functions. Therefore, it cannot be considered a framework, but rather a software library.

3.2.2. High-Level Programming Tools

High-level programming tools offer a generalized meta-structure with services to provide general functionality needed by most AR systems. This authoring approach minimizes development time and effort by helping programmers to focus more on the application than low-level problems. However, the application developer still has to deal with programming aspects.

Examples of high-level programming tools are Studierstube ES [**55**], osgART [**56**], and Metaio SDK, which provide a broad foundation for developing AR applications. The latter case is a modular framework that includes the entire infrastructure needed for building a mature and complete AR solution: a capturing component, a sensor interface component, a rendering component, a tracking component, and an interface to provide interaction between the application and the four modular components.

3.2.3. Low-Level Content Design Tools

Low-level content design tools provide another layer of abstraction in which the content itself becomes an aspect of authoring. This authoring model removes dependency on a programming language, replacing it by the description of virtual content and its relation to each other within an AR environment. Thus, the development process is still programmatic but relates to content [**5**].

AMIRE [**57**], DART [**58**], and trackman [**59**] are component oriented frameworks with graphical user interfaces designed to facilitate the process

of creating augmented reality applications. These tools provide a set of components which perform common AR tasks such as interaction IO hardware, data processing operations, and virtual content output. These components interfaces consist of a set of properties that can be accessed through property editors. trackman applications are developed by directly connecting components, as illustrated in Figure 20. AMIRE and DART applications are constructed by connecting properties of different components. Hence, the application designer first selects the components that are needed to build the AR solution. After various components have been added, they can be combined to construct an AR application.

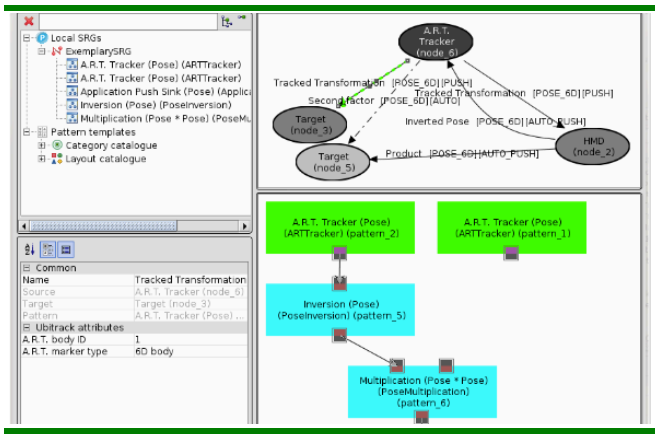


Figure 20: the trackman is a low-level content design tool that allows visual programming of AR applications by linking software components.

AR Scratch [**60**] is an augmented reality authoring tool for children, illustrated in Figure 21. When the user opens the program, it is split in three parts. On the left, all sort of pre-made programming blocks are listed within categories like Controls, Variables, Sensing, etc. Children can drag and drop each of these blocks to the center part and change their parameters to create the desired code block. In turn, the center part is the scripting tab, in which blocks are linked to one another to make the desired animation story. On the right, children can preview the results through a video camera, as they control the virtual world due to interactions between physical objects like cards and knobs.

Figure 21: AR Scratch is an authoring environment that uses a visual programming approach to allow children to shape augmented reality experiences.

3.2.4. High-Level Content Design Tools

A high-level content design tool provides an additional abstraction layer that fully hides the programmatic description of content and the need of scripting skills. Furthermore, this authoring approach concentrates on providing optimal support for a particular domain, thus the concepts become domain specific. This approach introduces the lowest complexity degree in the user interface and, as a result, these tools are the most intuitive and suitable for laymen.

For example, Powerspace [**61**] is an authoring tool built on top of Microsoft PowerPoint, which is used as a placeholder for the real world. First, images are placed on master slides to define real world references. Annotations like text, pictures and videos are then arranged on slides, thus defining their 2D relations to the reference elements. When the presentation is completed, its file is exported to the Powerspace Editor, in which slides are displayed as planes in the 3D world. In this editor, annotations can be spatially arranged to define their positions and orientations. In turn, the 2D reference images are replaced with a 3D geometry of the real world reference. Figure 22 illustrates the first steps of the AR authoring by the Powerspace tool.

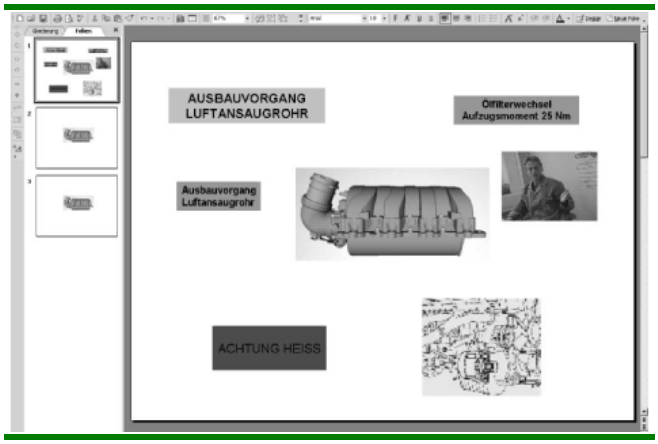


Figure 22: in Powerspace system, the user first defines 2D reference elements and annotations using PowerPoint slides.

As another example, SquareAR [**62**] is a desktop tool that allows authors to elaborate virtual restorations and planning of public spaces. The authoring tool is based on the video see-through approach, in which a monitor displays an aerial photo of the public space as a reference to create the virtual model of it. As can be observed in Figure 23, through an icon menu and a drag-and-drop mechanism, the authors place and manipulate virtual models over the real scene, such as threes, bus stops, and ground materials.

Figure 23: in SquareAR environment, the lateral menu provides three-dimensional models to be superimposed on the scene.

Also, Metaio Creator [**63**], Wikitude Studio [**64**], and the previous mentioned Layar Creator are desktop tools based on a GUI that allow to create a full AR experience without programming. The author first uploads trackables, which are presented in the scenario area. Through a resource menu and a drag-and-drop interface, resources are then added and manipulated in the scene, such as videos, three-dimensional models, texts, etc. Figure 24 presents the Metaio Creator's graphical user interface.

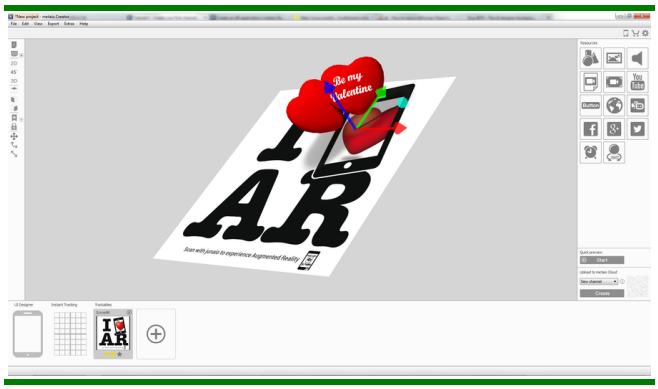


Figure 24: in Metaio Creator environment, the central panel comprises the AR scenario editing area. On the right, the resource bar enables to load the augmented digital content. On the left, the toolbar gives functionalities to switch between 2D, 45, and 3D view and to configure the scenario, such as translate, rotate, and scale the content.

Despite the relevance of programming tools, this work emphasizes content design tools in augmented reality. It then analyzes several dataflow of authoring tools within this category.

4. Content Design Tools: an Analysis

The advent of Web 2.0 substantially changed the way people use the Internet because it allowed users to create web content. Instead of only retrieving it, they had the opportunity to engage in creating and modifying web material. As a result, web interfaces have become simplified to a point that even people with few or no technical expertise could create and deploy content [**65**].

Interestingly, these developments in web technologies played an important role to envision and implement high-level authoring environments targeted at people with few or no technical expertise. Therefore, recent AR projects confronted authoring from the other end of the design spectrum, creating high-level tools to simplify the creation of AR systems with minimal or no programming.

Such advancements in the creation of web content led to the increasing relevance of non-expert users in the AR authoring field. During the analysis carried out in this monograph, two types of users were defined: *designers* and *end-users*. Both participate in the processes of AR content authoring and delivery. Designers are the ones who make the AR experience possible for the end-user. Although they have no technical expertise or programming abilities, they make use of high-level AR authoring tools to create the final AR solution. End-users, in turn, distinguish themselves by being those that access the AR experience after it has been fully developed and marketed.

High-level authoring tools in augmented reality, as can be observed, have greater relevance when we take into account the potential amount of people that can use AR solutions in the future. Indeed, assuming the wide acceptance of people using AR applications, the amount of content should reflect the massive user base and cannot be only created by a few AR technologists. Opportunely, the rise of Web 2.0 demonstrated how to create content for millions of users: user participation.

Due the relevance of high-level AR authoring tools, this chapter seeks to understand the current trends of such tools regarding the following fields:

1. The authoring paradigms that have been used in order to understand how they can improve AR content creation by the designer;
2. The strategies of AR content distribution, that have been applied in order to observe how AR experiences can be reached by a greater or smaller number of end-users.

For the purpose of respecting the taxonomy presented in the last chapter, the high-level AR authoring tools mentioned above will be referenced as content design tools from now on. The concepts of designer and end-user will also be explored in order to distinguish the important actor roles in the development and delivery of AR solutions. In this sense, this chapter describes the analysis of the current trends mentioned above regarding content design tools.

4.1. Methodology

Four steps were taken to explore the trends regarding the two fields mentioned in the introduction of this chapter. The first of them was the selection of content design tools available in the marketplace and literature. Then, as a second step, the analysis relied on

observing the dataflow of development and access to the AR content of each of the selected tools. Based on this individual analysis, the third step consisted in the identification of AR authoring and deployment trends, and consequently their benefits and limitations. Finally, the last step regarded the elaboration of general dataflow models through different combinations of the authoring and deployment trends previously identified. A minimum number of combinations were performed, so generic models could be elaborated in a manner that they could represent all of the project-specific dataflow previously analyzed. This section explores each of these steps and how they may help us understanding content design tools.

4.1.1. Selecting Content Design Tools

Initially, the keywords and expressions ("*authoring tool*" AND "*augmented reality*") were used in IEEE and ACM digital libraries in order to find relevant papers concerning authoring tools in augmented reality. That allowed for an investigation over important publications from 2001 to the present (2014). During this examination, only authoring tools classified through content design tools were selected for a deeper analysis, thus eliminating those categorized as programming tools.

4.1.2. Understanding Content Design Tools

Following the selection in the first step, a deeper analysis was performed regarding each of the selected authoring tools in order to understand the dataflow for development and access to the AR content. On a high level, this dataflow describes the end-to-end scenario that outlines the authoring and deployment of AR experiences, from the creation of AR semantic through authoring tools to its visualization by end-users. As an example, on the Metaio Creator, the designer first designs and builds the AR experience: trackables are uploaded, to which digital resources will be added, such as videos, three-dimensional models, hyperlinks, among others.

After the authoring process, the tool provides several project deployment alternatives, as can be observed in Figure 25. One of these alternatives is to export the AR project as an AREL package to be published in Junaio, Metaio's augmented reality browser. In this sense, Metaio automatically provides a hosting service. However, there is also the possibility for the designer to host the project on his own FTP server. A second alternative is to export the solution as a code package to be bundled into an iOS/Android application that makes use of Metaio platform specific SDK. A third alternative, in turn, allows to export the project as an executable package that will run as a desktop application that also uses Metaio SDK - *.exe* extension format for Windows and *.app* format for Mac computers. Therefore, the AR solutions may be accessed by end-users through Junaio platform, mobile apps that may be downloaded from Google Play and Apple Store, or executable desktop applications.

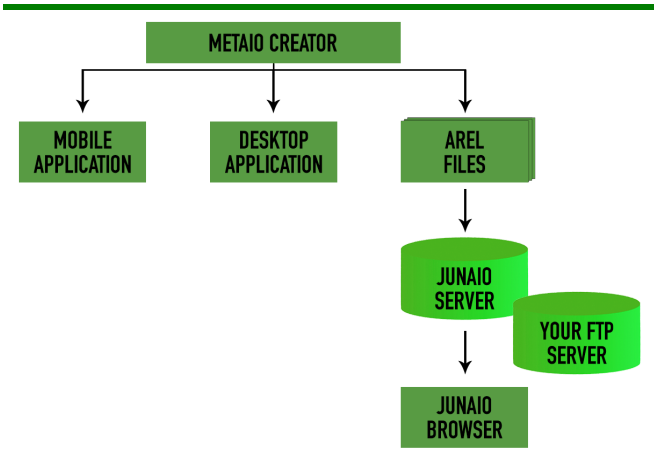


Figure 25: as can be observed, the Metaio Creator offers different hosting and delivering services. AR scenarios may be deployed as cloud-based apps that run on Junaio AR browser, or as mobile or desktop applications.

As another example, in the k-MART [**66**] authoring system, an AR solution is modeled as a collection of context-behavior pairs. According to the authors, some typical real-world contexts include the detection of a physical target object, the arrival of the user at a predefined location, a time-based event, an environment condition (e.g., brightness value), among others. On the other hand, examples of behaviors include time-

based display of text, three-dimensional objects, animations, videos, among others. Hence, in sum, mappings between various contexts and behaviors constitute the basic building blocks of AR solutions. Following this, the content is exported in a declarative structure which extends the standard X3D file format. Finally, the generated data are executed in a separate content browser.

Just as the two examples mentioned above, analyses were performed in each of the previously selected content design tools.

4.1.3. Categorizing Content Design Tools

The deeper analysis performed on each of the selected content design tools made possible the observation of trends regarding authoring and distribution strategies of AR experiences that have been used.

Furthermore, this observation also tried to understand (a) how the different authoring paradigms may support AR content development, and (b) how the deployment strategies seek to reach a larger number of end-users. In this sense, the pros and cons of these trends were analyzed by discussing the approaches through themes such as interoperability, maintenance cost, and offline capability.

4.1.4. Elaborating General Models

Finally, the identification of AR authoring and deployment trends allowed the translation of the project-specific dataflow, observed in the selected content design tools, into the creation of general dataflow models. In this sense, a minimum number of combinations of trends were performed in order to elaborate generic models, in which all of the content design tools could fit into.

4.2. Results

Once the publications from 2001 to 2014 were investigated, 6 papers were chosen among the 20 results provided in the IEEE research database, and 8 papers in the ACM repository among the 114 available outcomes. In sum, there were 14 tools selected in the literature. Beside, 7 commercial tools that are well consolidated or relevant in the market were chosen. Thus, taking into account both academic and commercial realms, there were culled 21 content design tools.

Thereafter, a dataflow analysis was performed in each of the selected content design tools. This concluded the first and second steps described in the methodology section.

Once individual analyses were performed in each of the previous selected content design tools, it was observed that two authoring paradigms have been used to create AR solutions, which this monograph refers as *stand-alone* and *AR plug-in* approaches. It was also noticed that two deployment strategies have been applied to make these AR experiences available for end-users, which this monograph refers as *platform-specific* and *platform independent* methods. These AR authoring and distribution trends are described below and are further compared, one trend against its alternative, in order for us to discuss their benefits and shortcomings.

4.2.1. AR Authoring Paradigms

4.2.1.1. Stand-alone

Historically, the term *stand-alone* may refer to a software program that does not require any auxiliary software other than the operating system to run. Thus, software such as plug-ins and expansion packages are not considered stand-alone programs since they will not run unless a certain hosting program is already installed.

Similarly, this monograph references stand-alone augmented reality authoring tools as software with all the necessary components for the development of complete AR experiences,

as can be seen in Figure 26. In turn, these components may include a graphical user interface, a series of importers, sensor interfaces, tracking and rendering engines, among others. In this sense, each stand-alone content design tool is a new software that allows designers to create their custom AR projects with more or less ease.

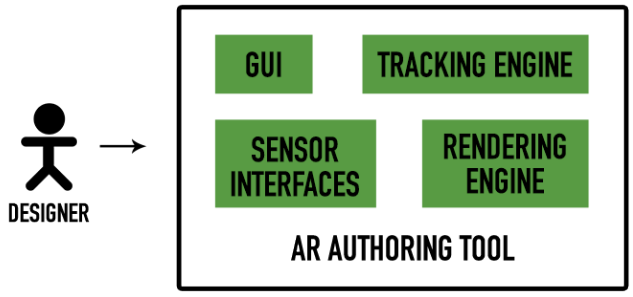


Figure 26: stand-alone AR authoring tools enable building entire AR experiences. In order to provide AR capabilities, these tools integrate components such as sensor interfaces, tracking and rendering engines.

As an example, the Metaio Creator provides a complete set of features along the entire creation workflow, such as graphical user interface including drag and drop to ease the scenario creation, tracking solutions for 2D and 3D trackables, and content importer for 2D, 3D, video and audio files.

4.2.1.2. AR Plug-in

Similar to conventional digital plug-ins, AR plug-ins are third-party software components installed on host applications in order to enable additional features, as illustrated in Figure 27. In this sense, these authoring tools provide augmented reality capabilities to traditional software applications, such as tracking techniques, access to physical sensors, three-dimensional rendering engine, among others.

It is relevant to note that, from the practical point of view, an AR plug-in instance will appear in the target software as a set of GUI elements, such as one or more menu items and toolbar buttons. Therefore, the whole AR authoring process occurs within the hosting environment, as the designer completely configures the desired AR experience by means of those elements along with the ones already provided by the target software.

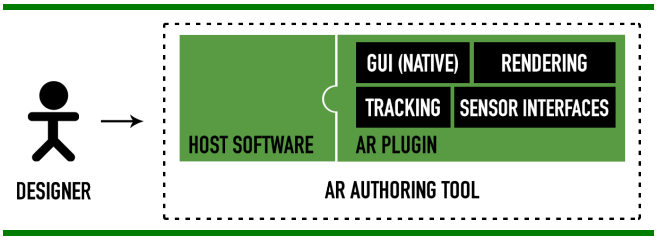


Figure 27: AR plug-ins provide AR functionalities for non-AR authoring environments. The designer interacts directly with the hosting software in order to create AR experiences.

As an example, the DART system is built as a collection of extensions to the Adobe Director [**67**], a widely used environment for multimedia content creation, to support the development of a variety of AR applications. DART [**4**] leverages the features that already existed in Director, including support for physical and virtual content in the design environment, sketch-based animatic content, fast video-mixed AR, marker tracking and other sensor data.

4.2.1.3. Stand-alone vs. AR Plug-in

Reusability. Stand-alone authoring tools generally offer a smaller set of features when they are compared to full-fledged hosting software. In the plug-in approach, the designer may create the AR experience by using not only the extra functionalities provided by the plug-in, but also mature, already existing features in the target software. On the other hand, it would require strong effort and time to implement these features in a stand-alone authoring tool.

For instance, DART is implemented on top of Adobe Director due to the fact that this is a full-featured tool for multimedia content creation, with a robust debugging and design environment, an active designer community, and cross platform support. Also, a guiding principle of DART work is that it should build on the existing design practices and tools used by experienced designers, rather than expect these designers to adopt radically new practices and tools [**58**].

Domain specificity. Another benefit of the AR plug-in over the stand-alone approach is that

each created plug-in can be specialized for a specific application domain. By using the plugin approach, one can propose a set of functionalities tailored to one specific application domain. Thus, implementing only the required features and not every possible one that usually a stand-alone tool must provide.

Learning curve. In addition, given a situation in which stand-alone tools are focused on specific tasks or application areas, the learning curve and the time employed to learn how to use each new tool, with different interfaces, is longer if compared to integrated tools. Since several plug-ins may be implemented to the same environment, each focusing on a specific task or application domain, the designer would not need to interact with a new interface.

Integration restraints. Despite the advantages of using the plug-in approach, bringing AR capabilities inside non-AR authoring tools is neither straight forward nor trivial. The first factor that must be considered is the availability of an SDK for the target software: if the target software is a closed system, it is extremely hard to manage the integration.

Another major factor to take into account is the graphical user interface. When designing an integrated AR-plugin for a host software, it is mandatory to create GUI elements with the same look-and-feel as the target software's. Moreover, the authoring metaphor must be used in a coherent way with the target software. In this sense, obstacles may arise during the development of functionalities that follow the traditional authoring style and, at the same time, enable AR content creation with ease.

As an example, DART is designed to complement the common development style used in the Director environment, which is based on a stage production metaphor. As illustrated in Figure 28, the environment includes a *stage* (where the content is placed), multiple *casts* (where all content elements are stored, including images, video, text and so forth), a *score* (the timeline of the experience) and *sprites* (cast members that have been placed on the stage or in the score). The main structuring mechanism in Director is the score. Traditionally, when an application runs, the "play-head" moves across the score from left to right. However, since interactive applications do not typically follow a fixed linear script, the timeline-based structuring metaphor might not be a natural metaphor for interactive content.

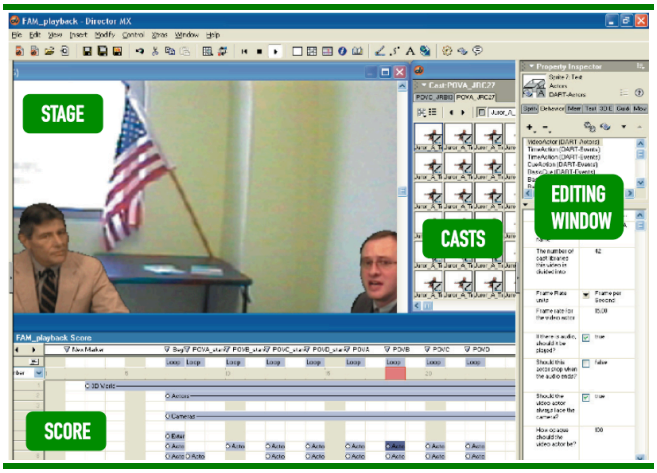


Figure 28: a work session on DART. The score includes various sprites and scenes (each scene is a column in the score). The stage presents the running experience and is currently showing part of a video content.

Hence, the challenge in designing DART was to create a system that matches the authoring style commonly used for complex Shockwave [**68**] content by experienced Director designers, yet is appropriate for AR content creation [**4**]. The struggle of this project was to expose complex AR topics like tracker transforms, cameras, and physical objects for occlusion via already understood authoring metaphors.

4.2.2. AR Deployment Strategies

4.2.2.1. Platform-specific

In the platform-specific (PS) approach, AR projects built through authoring tools are exported to archive files to be independently distributed. Some common archive file formats include *.exe* in Windows, *.dmg* or *.app* in Mac OS, *.apk* in Android, and *.ipa* for iOS operating systems. Note that these archive files are software packages used to distribute and install native application software (commonly

shortened as *apps*). A native app, in turn, is considered a stand-alone program itself since it is a self-contained program that does not require any auxiliary software on which must be executed, as can be observed in Figure 29. Native apps are usually available through application distribution platforms, such as App Store, Google Play, and Windows Phone Store. However, they must be downloaded from the platform to the end-user devices, such as iPhone, Android, Windows phones, or even laptops or desktop computers.

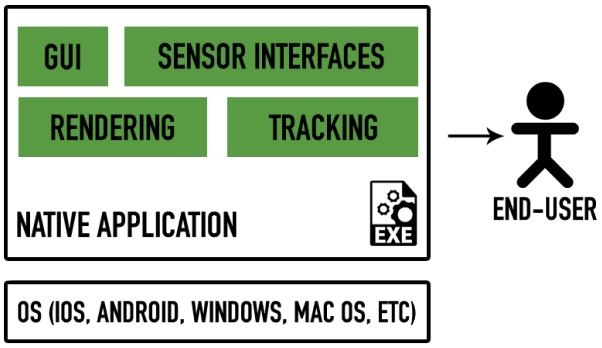


Figure 29: a native app includes all required elements to execute AR experiences, thus can be considered a stand-alone software itself. Also, the term *native* comprises applications compiled at runtime, such as an Android app, or precompiled executable programs.

As an example, the Metaio Creator supports deployment options to mobile applications for iOS/Android platforms, and to executable programs for Windows/Mac OS computers.

4.2.2.2. PLATFORM-INDEPENDENT

As it was mentioned, the previous approach distributes the created AR experiences as stand-alone, native applications installed on end-user devices. The platform-independent (PI) approach, on the other hand, delivers the AR solutions as data files read and executed on a software platform running on the end-user device. Also, it is worth pointing out that, after the authoring process, the generated content requires a platform on which it must be executed. Therefore, the content cannot be considered a stand-alone program; rather, it comprehends data files (commonly structured in XML-based formats) that are interpreted by the software platform, as illustrated in Figure 30.

As an example, the previous mentioned k-MART system allows designers to export AR solutions as X3D-based data files. In turn, these files are later executed on a separate content browser.

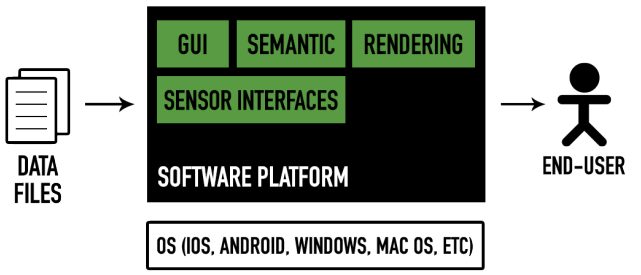


Figure 30: data files are interpreted by a native shell, which provides the required infrastructure to present AR experiences.

Furthermore, since the content does not need to be installed in the device, a major advantage is the possibility of implementing a cloud-based deployment service. This increasingly popular variant approach uses a server infrastructure as a backbone. The remote server is responsible for content storage and retrieval as requested by the clients. The clients, in turn, are responsible for presenting the retrieved content on end-user devices. Also, a client comprehends a cloud-based software platform that reaches into the cloud for contents on demand. In turn, all data files remotely accessed are here referenced as cloud-based applications.

As an example, leader companies in AR technology like Metaio, Layar, and Wikitude developed Junaio, Layar App, and Wikitude World AR browsers, respectively. To the end-user, an AR browser looks very similar to a typical native app: it is downloaded from an app store, stored on the mobile device, and launched just like a native app. However, the most prominent advantage of AR browsers is that end-users need only one app for multiple content. Once installed, they pull new cloud-based apps on demand from the cloud.

4.2.2.3. PS VS. PI

Portability. Since native apps are built using the device's native programming language, they only run on their designated platform. This means that the same app cannot be re-used between

platforms. Thus, deploying a native app to Android, iOS and Windows Phone would require creating three different applications to run on each platform.

Contrastingly, one major promise about platform independence is that designers only have to write the application once and then it will be able to run *anywhere*, without having to be recreated by the designer for each separate platform. *Anywhere*, that is, all major operating systems that support the software platform. For instance, the previous mentioned augmented reality browsers are cross-platform applications. Junaio allows cloud-based apps to run identically in iOS, Android, Windows, and Mac OS platforms. Layar App and Wikitude World support iOS, Android, and BlackBerry 10 operating systems.

Maintenance cost. Maintaining native applications is also expensive for the designer. As a native app is built for a particular device and its operating system, whenever new OS versions are rolled out, native apps may require considerable updates to work on these newer versions. Consequently, changes have to be packaged in a new version of the app and placed in the app store.

On the other hand, data files run independent on a platform-specific shell that operates as an abstraction layer that encapsulates the underlying hardware and software updates. Hence, the designer does not have to worry about updating and resubmitting apps.

Offline functionality and speed. This is a clear advantage for native apps. Since the application remains installed on the device from the original download, depending on the nature of the app, no internet connection may be required. Another area where native apps have a clear advantage is speed. These apps, by definition, run at native speed. Cloud-based apps run on top of additional layers, which consume computing resources and can therefore decrease the speed degree. In this sense, end-users get peak performance at all times regardless the limited internet access.

4.2.3. General Models

Given the authoring and deployment trends explored in the previous subsection, it was possible to elaborate four general dataflow models that represent all of the content design tools' dataflow analyzed in this monograph. These four general models are described and discussed below.

4.2.3.1. Model 1: Stand-alone PS Model

As can be observed in Figure 31, this dataflow model embodies a stand-alone authoring approach combined with a native distribution strategy. In this sense, the designer first creates AR experiences through stand-alone content design tools. Then, the designer may eventually export the project as platform-specific archive files, which are used to deliver stand-alone, native applications for Android, iOS, BlackBerry 10, Windows, or other operating system.

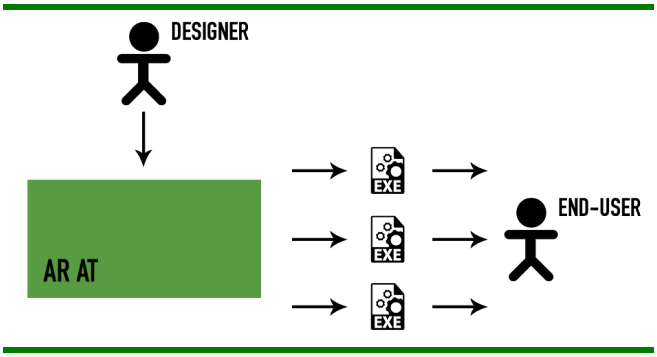


Figure 31: this model combines a stand-alone authoring with a platform-specific distribution. Therefore, each generated native application is individually installed and accessed by end-users.

4.2.3.2. Model 2: Stand-alone PI Model

Similarly to the previous model, the designer first builds AR experiences through stand-alone content design tools; however, these models greatly differ in their content delivery services. On the one hand, the stand-alone PS model uses a platform-specific policy to distribute AR solutions with offline capabilities and native speed. On the other hand, this model applies a platform independent strategy for reaching interoperability and maintainability. In this sense, the designer exports the authored AR solutions

as data files that run on a separate software platform.

Note that, a content design tool can generate one or more data files which can be interpreted and run in a single software platform, in different periods of time (Figure 32). Yet, since each stand-alone content design tool is a brand new software, the data files created by distinct tools generally differ in their structures, logics, and formats. For instance, the previous mentioned k-MART system describes AR projects as custom XML files that extend the standard file format X3D; while Metaio Creator exports AR experiences as AREL packages, which are based on technologies such as XML for static content definition, and HTML5 to provide the graphical user interface and application logic. Hence, the *software platform* mentioned above may comprehend a number of distinct software engines, including augmented reality browsers, WebGL-enabled browsers, or other proprietary software that may be adequate for parsing and executing specific data files.

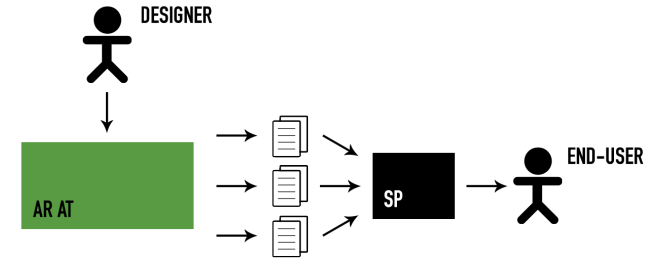


Figure 32: this model unites a stand-alone authoring with a platform-independent distribution.

4.2.3.3. Model 3: All-in-one Model

As illustrated in Figure 33, in this model, both designers and end-users utilize the same environment to build and access AR solutions, respectively. In the sense, the designer creates and saves AR solutions as data files. Eventually, these files are read and executed within the same environment in order to present the AR experience to end-users.

Hence, similarly to the previous model, the all-in-one model comprehends a stand-alone authoring approach combined with a platform-independent distribution. However, the major difference resides in the fact that production and delivery services are merged within a single ambient.

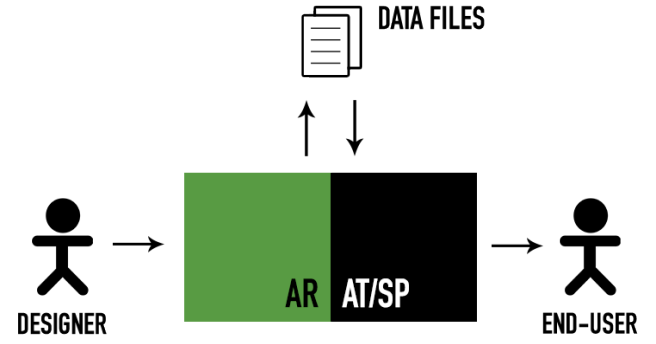


Figure 33: this model combines a stand-alone authoring with a platform-independent distribution. Yet, both designers and end-users utilize the same ambient to create and visualize AR solutions.

4.2.3.4. Model 4: AR Plug-in PI Model

In this dataflow model, the designer first builds AR projects through hosting software integrated with AR plug-ins. Then, these projects are saved as data files that are later retrieved and executed on a separate software application. In other words, this model includes a plug-in approach combined with a PI deployment strategy, as can be observed in Figure 34.

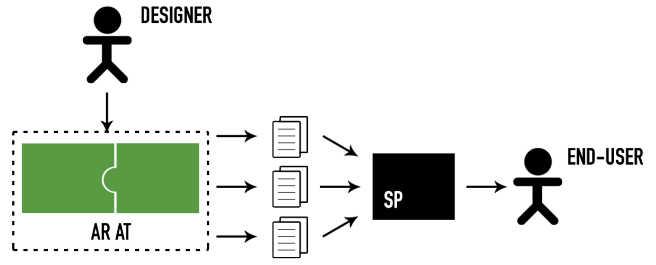


Figure 34: this model merges an AR plug-in authoring with a platform-independent distribution.

All the content design tools that were selected and analyzed in this monograph are listed in the tables below (Table 1 and Table 2). The tables divide the commercial and academic tools. In the latter case, the works are organized according to their year of publication ending with the most recent one. Note that, works in which the authors did not name the tools, these will be called as *tool + reference number*.

The tables indicate to which of the four general dataflow models each tool belongs. It is important to keep in mind that it is not

mandatory for a tool to be categorized into only one general model. After all, as it has been discussed in this chapter, a single content design tool can provide different distribution approaches and, consequently, different dataflow models.

The next chapter integrates all the topics explored in the previous ones. It describes a case study in which a content design tool has been implemented, so it can be observed if it indeed facilitates AR content creation by non-programmers. Besides, the generated AR solutions are loaded and run in a separate software, which is responsible for presenting the content to end-users. In this sense, the next chapter also describes the composition of one of the general dataflow models mentioned above. This allows a deeper understanding of this model and its particularities.

Commercial Content Design Tools	**Model 1**	**Model 2**	**Model 3**	**Model 4**
Metaio Creator [**63**]	■	■		
Metaio AR Creator Plugin [**69**]				■
Wikitude Studio [**64**]	■	■		
Layar Creator [**48**]		■		
Build AR [**70**]		■		
trackman [**59**]		■		
AR-media Plugin [**71**]				■

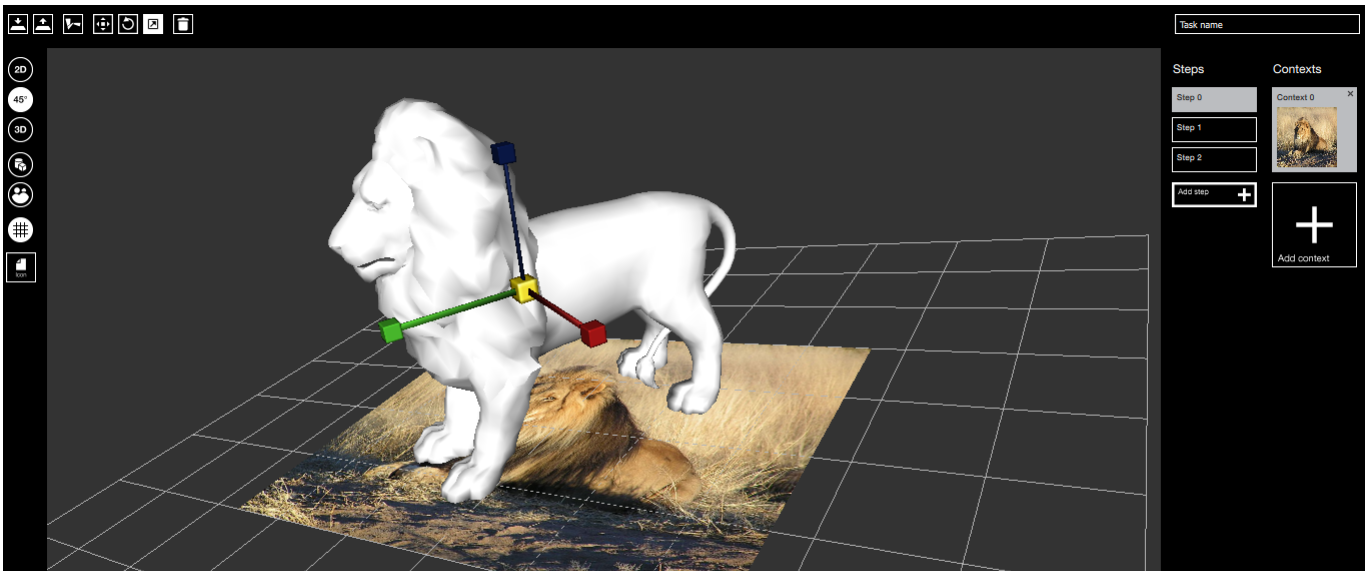
Table 1: classification of each commercial tool according to the general dataflow models.

Academic Content Design Tools	**Year**	**Model 1**	**Model 2**	**Model 3**	**Model 4**
Powerspace [**61**]	2002		■		
Authoring Wizard [**72**]	2003		■		
AMIRE authoring environment [**57**]	2004		■		
DART [**58**]	2004				■
Tool 1 [**73**]	2004			■	
ComposAR [**74**]	2008			■	
VREditor [**75**]	2009		■		
ARBookCreator [**76**]	2009			■	
AR Scratch [**60**]	2009			■	
k-MART [**66**]	2010		■		
SquareAR [**62**]	2011			■	
Tool 2 [**77**]	2012		■		
AVATAR [**78**]	2012				■
Tool 3 [**79**]	2013		■		

Table 2: classification of each academic tool according to the general dataflow models.

5. Content Design Tools: a Case Study

Figure 35: the image below presents a high-level content design tool, called Augie Studio, through which AR instructors can be created and further executed on Augie platform. In this environment, the augmented content can be manipulated through colored manipulator widgets (the arrows for translation, sphere-shaped for rotation, and cube-shaped for scale).



Because of the increasing relevance of high-level authoring tools in the AR field, this chapter presents a case study in which a high-level authoring tool was implemented and a usability test was performed in order to evaluate the abilities of these tools to be intuitive, efficient, and pleasant to interact with during the development of AR experiences by both experts and laymen.

Due to the availability of an augmented reality software platform, which was developed in a prior work, it was chosen to implement a tool through which AR experiences can be created and further executed on this platform. In this sense, it was envisioned the development of one of the general dataflow models described in the previous chapter, the stand-alone PI model, since this would allow a richer description of this model and its particularities.

Hence, this chapter integrates all the foundations acquired from the previous chapters. Moreover, it provides a deeper understanding of these theoretical foundations while demonstrating how theory can be introduced into practice. The following sections describe the ready-made software platform and the implemented high-level content design tool, named Augie and Augie Studio, respectively. Besides, it is reported the elaboration and execution of a usability testing focused on evaluating the tool's capacity to meet its intended purpose.

5.1. Augie Platform

Augie is an Android application that makes use of Metaio platform-specific SDK. This application executes AR instructors, which, in a similar manner to traditional instructions, provide step-

by-step guidance about performing specific tasks or functions. Yet, they use AR technology to incorporate relevant instructions directly within their contexts of use, overlaying layers of digital information to the physical scene at which the user is looking at, as can be seen in Figure 36.

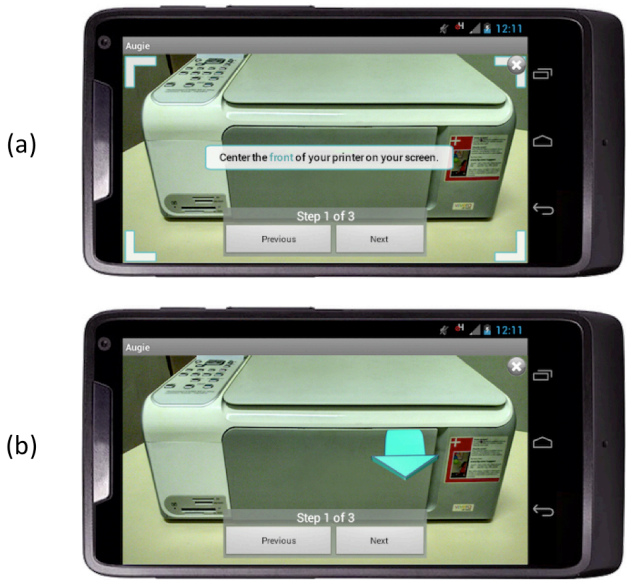


Figure 36: the Augie interface (a) before and (b) after detecting a target image.

Once the application is launched, a menu icon presents all the available instructors. The end-user can then select the desired instructor, which causes the application to show a second menu listing all the tasks contained in the chosen instructor. The end-user can then choose a specific task to be executed, which starts the device's camera.

In turn, a task consists of a series of steps to complete an action. In turn, each step is composed of one or more contexts. A context is tied to a single image target and equipped with virtual content such as images, three-dimensional models, animations, and sounds. To unlock this virtual information on top of the real world, the user simply points the device's camera at the tracking target.

Furthermore, the end-user can navigate from one step to another using the forward and backward buttons. It is important to mention that, during the reproduction of one step, multiple image patterns can be tracked simultaneously. Indeed, the actual amount of image targets corresponds to the number of contexts contained in this step. Hence, in case one of these target images is recognized, the corresponding virtual content is rendered on the device screen.

Additionally, above the navigation buttons, it is presented the current and the total number of steps. After passing through all the steps, the user can return to one of the previous menus and choose a different task or instructor.

5.1.1. Instructor: Logical Structure

From the explanation above, it can be noticed that AR instructors have a hierarchical structure. Also, they can conceptually be interpreted as a tree data structure in which the root node symbolizes the instructor with its subtrees of children representing tasks. Figure 37 describes in detail the logical structure of an instructor.

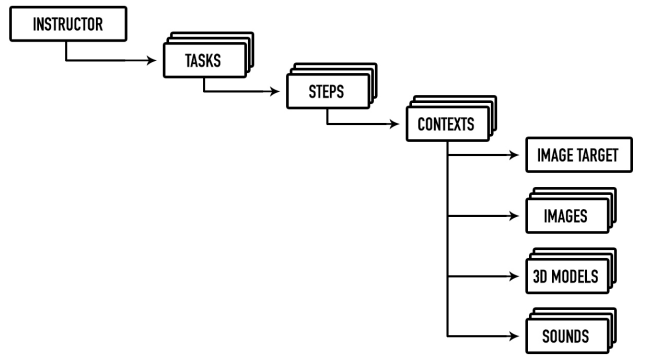


Figure 37: logical structure of an instructor.

As it was previously mentioned, an instructor consists of a series of tasks, and each task can perform a certain sequence of steps. In turn, each step is composed of one or more contexts. Each context, in contrast, is tied to a single image target, yet it may contain several sounds, images and three-dimensional models spatially arranged – that is, their positions, orientations, and scales - according to the trackable image.

5.1.2. Instructor: Physical Structure

Technically, an instructor consists of a set of data files organized according to the folder structure illustrated in Figure 38. In turn, these

files define configuration information of the instructor, such as its logical structure, the asset mapping list, and the tracking setup. This structure is detailed as follows.

AssetsSample. This XML file lists the resource files used while executing the instructor – i.e. the three-dimensional models, images, target images, and sounds. In this list, each asset is structured as a key-value pair. The key is a unique identifier, while the value contains the path to the asset file. In turn, these keys are referenced in the *xmlConfig* and *InstructorSample* files.

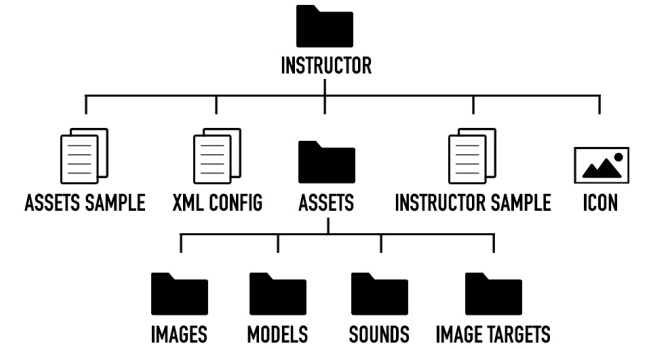


Figure 38: physical structure of an instructor.

xmlConfig. This XML file corresponds to the tracking configuration file commonly set in applications that utilize the tracking strategies provided by the Metaio SDK. Hence, this file determines the tracking configurations used while executing the instructor: the tracking technique such as marker-based, markerless, or sensor-based; the reference image targets that should be tracked; the maximum number of target images to be tracked in parallel; the similarity threshold for specifying whether the template tracking was successful or failed; among others.

InstructorSample. This XML file defines the instructor by following the same hierarchical structure mentioned in the previous subsection.

Icon. A regular image that represents the instructor icon. Once the Augie application is launched, this image is presented in the initial menu icon.

Assets folder. This folder contains the raw files used while executing the instructor. Yet, these resource files are further organized according to their formats and usage, in subfolders named Images, Models, Image targets, and Sounds.

It is worth pointing out that each instructor folder is locally stored on the SD card in an Android device, under a directory named Augie. Eventually, once the Augie application is running, it reads all available instructors from this directory, interprets and presents them to end-users through the initial menu icon.

5.2. Augie Studio Tool

Before executing augmented reality instructors on Augie platform, they must first be created. In this sense, this section describes a high-level content design tool, named Augie Studio, in which instructors can be created and further executed on Augie platform.

Augie Studio was developed using the Qt [**80**] application and UI framework, and it is written with a combination of QML [**81**] and C++ languages. Some factors were considered when choosing this framework. One factor taken into account was that Qt allows the developers to write multi-platform applications across all supported platforms, such as Windows, Mac OS X, Linux, Android, iOS, among others. Note that, although Augie Studio is a cross-platform application, this monograph mainly used the Windows platform.

A second factor considered was that Qt framework includes the Qt Quick module, which provides a declarative scripting language called Qt Modeling Language (QML). Due to its similarities to CSS and JavaScript technologies, QML language eases the design and implementation of fluid user interfaces.

A third factor considered was that Qt provides support for integration with OpenGL [**82**] library, which is used for rendering 3D graphics. For this purpose, it was also used the

Figure 39: the graphical user interface of Augie Studio comprises four major areas.

libQGLViewer [**83**] rendering library. This is an open source library based on Qt that provides functionalities to accelerate the creation of OpenGL three-dimensional viewers.

5.2.1. User Interface

Augie Studio is entirely based on a GUI which allows the designer to create a full AR experience without having to program. The included buttons enable all needed actions to create, manage, and deploy augmented reality instructors.

As illustrated in Figure 39, the application interface can be divided into four areas. The first one is the AR scenario editing area on the center of the GUI. This is the graphical work area that designers use to visualize and configure the currently selected AR scene.

The second area is the toolbar on the left of the GUI. It gives the designer the possibility to upload and include virtual content to the AR scenario, set the instructor icon, or switch between 2D, 45º, and 3D view. The third area is the toolbar on top of the GUI. It provides access to additional features such as to translate, rotate, scale, or delete virtual content. It also permits the designer to open/save the AR project, or export the project as data files to be read and executed on Augie platform.

The fourth area, finally, is the trackables bar on the right of the GUI. This area includes two vertical lists. The one on the left displays all the existing steps, and the designer creates new ones by pressing the plus button placed at the bottom of this list. By selecting one step from the left list, all its contexts are automatically displayed on the right list. In turn, each item from the right list holds a thumbnail image, which acts as a button to set the target image of the related context. Yet, in a similar manner to the left list, the designer includes new contexts by pressing the plus button at the end of the right list. In turn, a context selected from this list is automatically loaded in the AR scenario editing area.

From the explanation above, it is worth pointing out that the current version of the tool suffers from some limitations. In this sense, the tool neither supports creating multiple tasks per instructor nor setting sound resources to the AR scenario.

5.3. Usability Testing

Since the purpose of this case study was to analyze the ease of use of high-level content design tools by both experts and laymen, we conducted a usability test with the Augie Studio tool to evaluate the following attributes:

- Learnability: how easy is it for users to learn how to use the tool?
- Efficiency: how quickly a user can construct an instructor?
- Satisfaction: how pleasant is it to use the interface design?

For this purpose, we recruited six representative users that were divided into two groups, with each group containing three people. One group was composed of people with no programming or scripting skills. Contrastingly, the other group contained developers with prior experience in AR application development. The tests were conducted individually with each of these participants, and consisted in six steps that are described as follows.

First, the basics regarding Augie and the concept of instructors were explained to the participant. During the explanation, an instructor was executed on Augie platform so the participant could better understand the software and the notion of instructors within a practical view. Second, we covered the authoring process in Augie Studio, presenting each of the user interface components and how it could be used during the creation of AR instructors. The third step consisted in letting the participant freely use the authoring environment during a small period of time, so he could become familiar with it. To summarize, the three previous steps consisted in a theoretical and practical training. In turn, this was important since most of the participants did not have prior comprehension of both the Augie platform and the Augie Studio tool. This brief training lasted for about 10 to 15 minutes.

In regards to the three next steps, the aim was to observe people using the authoring environment in a realistic manner, so that we could gather information on the strong and weak aspects of the usability of the interface design. For this purpose, we prototyped two animal encyclopedias, with pages containing pictures and information about different animals, as can be observed in Figure 40.

In this sense, as the fourth step, we requested the participant to create two instructors, in which three-dimensional animal models should be displayed whenever the user points the device's camera at the corresponding pictures in the encyclopedias. Hence, both tasks consisted in spatially arrange the 3D models - that is, their positions, orientations and scales - according to the real target images contained in the animal encyclopedias. At the end, once the instructors were created, the participant was able to see their AR applications actually working on Augie platform.


Voxar Labs November 18, 2014

MAMMALS

Quick Guide of Mammals Animals

Biology

The average life expectancy of a camel is 40 to 50 years. A full-grown adult camel stands 1.85 m (6 ft 1 in) at the shoulder and 2.15 m (7 ft 1 in) at the hump. Camels can run at up to 65 km/h (40 mph) in short bursts and sustain speeds of up to 40 km/h (25 mph). Bactrian camels weigh 300 to 1,000 kg (660 to 2,200 lb) and dromedaries 300 to 600 kg (660 to 1,320 lb).

The male dromedary camel has in its throat an organ called a dulla, a large, inflatable sac he extrudes from his mouth when in rut to assert dominance and attract females. It resembles a long, swollen, pink tongue hanging out of the side of its mouth. Camels mate by having both male and female sitting on the ground, with the male mounting from behind. The

Camel

A camel is an even-toed ungulate within the genus Camelus, bearing distinctive fatty deposits known as "humps" on its back. The two surviving species of camel are the dromedary, or one-humped camel (C. dromedarius), which inhabits the Middle East and the Horn of Africa; and the bactrian, or two-humped camel (C. bactrianus), which inhabits Central Asia. Both species have been domesticated; they provide milk, meat, hair for textiles or goods such as felted pouches, and are working animals with tasks ranging from human transport to bearing loads.

Figure 40: one of the pages contained in the animal encyclopedia.

During the execution of this step, it was measured the amount of time participants took to create each of the required instructors. Thus, it could be determined the average time per group and the general average time spent on creating the instructors. In turn, these quantitative results can strongly indicate if the tool is learnable and efficient in reducing the amount of time needed for the authoring process. Note that, despite the encyclopedia similarity, the sequence in which the user created the instructors was randomly defined in order to avoid tendentious results.

In the fifth step, the participant answered a post-test questionnaire to evaluate efficiency, satisfaction, and ease of understanding information about the Augie Studio, wherein the participant rated the measures on a Likert scale [**84**], as can be seen in Appendix A. As the last

step, a non-structured interview was conducted to gather additional comments about the authoring experience, the potential of use of this tool by non-technologists, and changes to improve user satisfaction.

5.3.1. Results

During the execution of the tests, we considered some variables for measuring the usability of the Augie Studio tool, such as the time spent for creating the instructors, the answers obtained from the post-test questionnaire, and the comments collected during the non-structured interview.

The time results proved that the Augie Studio tool meets the requirements to be easy to understand and manage by both programmers and non-programmers. All participants were able to complete creating the two required instructors in a significant short period of time. Furthermore, these results demonstrated that the tool, indeed, greatly simplifies the authoring stage, which leads to more efficiency and increasing productivity in the AR content creation. This can be observed on Table 3, which presents the average time (in minutes and seconds) spent to elaborate each of the two instructors, divided per group.

As can be noticed on Table 3, the non-programmer group spent about 8 m 44 s to elaborate an instructor. This result has greater relevance when we take into account that non-experts are people with no programming skills, no previous knowledge in creating AR experiences, and have had no previous contact with neither Augie platform nor Augie Studio tool. Therefore, this observation is a strong indicator that the tool is intuitive, as, after all, after a short 10 to 15-minute training, these participants were able to create AR applications in less than 10 minutes.

The information on Table 3 also communicate that, in general, the amount of time participants spent on creating the instructors has decreased from the first to the second one. Actually, while conducting two test cases, during the development of the second instructor, we observed that the participants felt more familiar with the tool and aimed at exploring it a little further. Thus, the time used to build the second instructor was increased and so negatively affected the previously mentioned fall time between the two instructor buildings. Yet, this behavioral component demonstrates user motivation and engagement in using the tool, which is a valuable aspect of the product design.

Table 3: each column illustrates the average time spent on creating the two required instructors, organized per group. In turn, the last column indicates the general results.

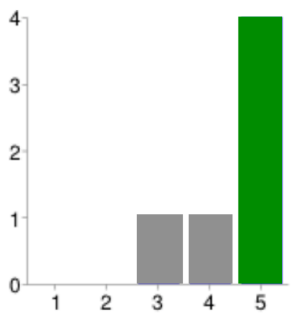
	Experts	Non-experts	General
Instr. 1	4 m 46 s	8 m 48 s	6 m 47 s
Instr. 2	3 m 18 s	8 m 41s	5 m 59 s

After building the instructors, the participants were requested to evaluate their experience according to four affirmatives. Each of the sentences could be answered with numbers ranging from 1 (deeply disagree) to 5 (completely agree), as can be observed in Figure 41. The purpose of this survey was to gather some feedback regarding the efficiency, satisfaction, and ease of understanding of the Augie Studio tool. In turn, the participants were encouraged to enter into a more detailed conversation, giving further comments about these issues during the non-structured interview, which followed the questionnaire. This dialogue aimed to collect information that could be relevant to consider design changes on the Augie Studio tool in order to improve its usability, and also the potential use of the tool for a diverse public, in different application domains.

In this sense, the participants mentioned that the Augie Studio tool indeed simplifies the authoring process, allowing them to create AR applications spending low time and effort. This declaration was reaffirmed in the answers obtained from the questionnaire. Chart 1 at Figure 41 presents the number of participants who agreed that it is possible to rapidly build an

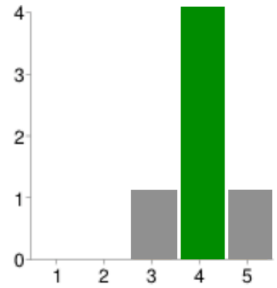
instructor. Even though one user responded with level 4 and another replied with level 3, most of the users strongly agreed with the affirmative.

1. It is possible to rapidly create an instructor.



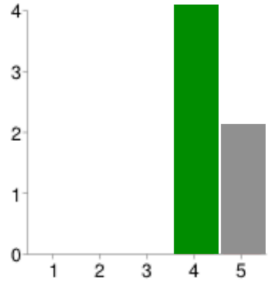
1	**0**	0%
2	**0**	0%
3	**1**	17%
4	**1**	17%
5	**4**	67%

2. It is possible to easily interpret the functionalities of the buttons.



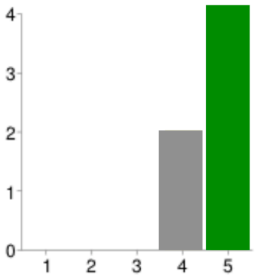
1	**0**	0%
2	**0**	0%
3	**1**	17%
4	**4**	67%
5	**1**	17%

3. It is possible to easily manipulate the virtual content.



1	**0**	0%
2	**0**	0%
3	**0**	0%
4	**4**	67%
5	**2**	33%

4. The forms of visualization were enough for creating the instructor.



1	**0**	0%
2	**0**	0%
3	**0**	0%
4	**2**	33%
5	**4**	67%

Figure 41: the questionnaire results. The y-axis represents the number of participants, while the x-axis denotes the level of agreement.

Particularly, all programmers affirmed that creating an AR application through the authoring tool indeed decreases time when compared to program the application from scratch. Note that, as it is illustrated in Table 3, the expert group spent less than 5 minutes to create an instructor.

However, some experts mentioned that the authoring tool limits the application structure. That is, it must be an instructor composed of a sequence of steps, each one containing a series of contexts, and so forth. Although it is possible to create a number of different AR applications using this structure, it is still restricted. This is due to the fact that, the more interfaces are abstracted, the lower the level of control authors have over the final application. This trade-off against a loss of flexibility and control between tools for expert versus novice is well known in the human-computer interface community [**85**]. Experts want tools that provide more features and finer-grain control over the presentation and behavior of the content being created. However, more features and finer-grain control typically leads to more complex user interfaces, which makes the authoring tool more difficult for less experienced people to learn and use. On the other hand, a tool designed for inexperienced people might be missing features required for an expert.

Furthermore, the participants suggested some design adjustments in order to boost the usability of the Augie Studio tool. One recommendation is to enable the manipulation of the virtual content in other perspectives besides 2D and 45. Some participants mentioned that it would be interesting to be possible to instantly define the perspective view during the content editing. Yet, they affirmed that this viewing aspect did not make serious implications concerning the authoring process. In turn, this can be verified from the answers obtained in the questionnaire. Chart 4 at Figure 41 presents the percent of users who agreed that the forms of visualization were enough to construct the instructor. In this sense, two users responded with level 4 and four users totally agreed with the statement.

A second recommendation regards at modifying some symbols used at the button icons in order to get the proper metaphor for each action, thus allowing a meaningful

interpretation even for a diverse public. Some participants mentioned that an alternative idea would be to insert “hint texts” so users could get short additional information about the use of a button whenever the mouse hover over it. Yet, the participants did not consider the buttons design a major obstacle for building the instructor. In this sense, Chart 2 at Figure 41 approaches the possibility of effortlessly understand the functionalities of the buttons. Despite the fact that one participant answered with level 5 and another with level 3, most of the users chose level 4 to express that they agreed with the sentence.

A third recommendation refers to the mouse interaction metaphor used to manipulate the virtual content – in other words, to translate, rotate, or scale it. For some participants, rather than first selecting the content with the left mouse button and then manipulating it using the right button, it would be more natural to use the same mouse button to realize both actions. Despite this, Chart 3 at Figure 41 approached the easiness with which participants could manipulate the virtual content. As can be noticed, this chart presents good evaluation results, in which four users replied their level of agreement as 4 and two users responded that they completely agreed with the sentence.

To summarize, the findings obtained from the time measurement, the post-test questionnaire, and the non-structured interview made it possible to validate that the Augie Studio tool fulfills the requirement to be easy to use and, also, to allow a diverse audience to create AR applications. Interestingly, some non-experts mentioned that they could envision developing a number of AR applications that could be useful in their own expertise area. This consideration demonstrates that authoring tools, indeed, have the capacity of making the use of AR more popular in different application domains.

6. Conclusion

Figure 42: in the real state sector, augmented reality can be used to project information about the properties that are available for sale over the image of the houses. In this sense, content design tools may help architects to create AR architectural model buildings as an innovative solution to display homes haven't been built yet, thus helping them to generate new clients and set up a faster communication with their customers.

The research conducted for this monograph made it possible to observe that authoring tools help improving the maturation of AR technology in a diverse set of AR research fields. Moreover, it was possible to understand how these tools facilitate the development of AR applications, thus being a commonly used solution to increase the use of AR. In this sense, augmented reality authoring tools can provide several levels of abstraction, thus targeting audiences within a range of different technical expertise. Particularly, those categorized as content design tools allow non-technologists, such as marketers, architects, doctors, and professors to explore the AR creation medium, as illustrated in Figure 42. Therefore, these tools are an essential component for helping AR technology to gain popularity in different application domain.

The trend analysis conducted in this monograph has presented that there are two authoring paradigms and two distribution strategies that have been widely used for content design tools, in both commercial and academic realms. Furthermore, these authoring and deployment trends can be combined to elaborate generic dataflow models, which represent all the project-specific dataflow models that have been investigated throughout this analysis.

This monograph has also attempted to illustrate the theoretical knowledge approached in this work through a practical case study, in which a high-level content design tool was implemented, named Augie Studio. Beyond that, this report has also conducted a usability test for the given tool and it concludes that, indeed, the Augie Studio tool is easy to use by both programmers and non-programmers. Therefore, we can assume that content design tools have the potential to encourage a greater use of AR technology in different domain areas.

6.1. Future Work

Despite of that, there are still issues that require a deeper investigation in this monograph. Some

of these affairs include the three design adjustments mentioned by the participants during the usability testing of the Augie Studio tool. As a first usability improvement, it is desirable that the tool supports creating new perspective views of the three-dimensional scenario area to work on editing the virtual content, besides the pre-defined 2D and 45º views.

A second design modification refers to creating stronger associations between the underlying functions of buttons and their graphical representation. For this purpose, we plan on replacing some button icons to more self-explanatory depictions, thus diminishing ambiguity and misinterpretation. The idea is to use consolidate concepts for actions and metaphors, such as the metaphor of a floppy disk for saving and a folder for opening project files.

As the third design alteration, modify the mouse interaction used on the selection and manipulation of the virtual content so it uses only one mouse button. In this sense, based on comments collected from the test participants, it would be more convenient for the designer to left-click the colored manipulator widgets, drag, and finish the transformation whenever he releases the left mouse button.

Furthermore, it remains for future works to add support in the Augie Studio tool for setting audio resources and creating multiple tasks for a single instructor, thus supporting full integration for the AR content executed on Augie platform.

Finally, it remains to implement a technique to calculate the relative pose between two or more image targets of existing contexts from the same step. This calculation is desirable when we take into consideration a situation where various images of the same three-dimensional scene, but from different viewpoints, are set as targeting images. In this sense, we want to avoid the redundant and cumbersome task of setting the same virtual content on every existing viewpoint. Therefore, we need to estimate the relative pose between each image and a specified based image, in which the designer may include the augmented content to be used while executing the referred step. Hence, the position and orientation of these virtual models in the other viewpoints can be automatically determined based on the calculated transformation.

APPENDIX A

Augie Studio: Evaluation Questionnaire

It is possible to rapidly create an instructor.

	1	2	3	4	5	
Totally disagree	◯	◯	◯	◯	◯	Totally agree

It is possible to easily interpret the functionalities of the buttons.

	1	2	3	4	5	
Totally disagree	◯	◯	◯	◯	◯	Totally agree

It is possible to easily manipulate the virtual content

	1	2	3	4	5	
Totally disagree	◯	◯	◯	◯	◯	Totally agree

The forms of visualization were enough for creating the instructor

	1	2	3	4	5	
Totally disagree	◯	◯	◯	◯	◯	Totally agree

Submit

REFERENCES

[1] Ronald Azuma et al., "Recent Advances in Augmented Reality," *IEEE Computer Graphics and Applications*, vol. 21, no. 6, pp. 34-47, November 2001.

[2] Tobias Langlotz et al., "Sketching up the world: in situ authoring for mobile Augmented Reality," *Personal and Ubiquitous Computing*, vol. 16, no. 6, pp. 623-630, August 2012.

[3] Matt Bower, Cathie Howe, Nerida McCredie, Austin Robinson, and David Grover, "Augmented Reality in Education - cases, places, and potentials," in *International Council for Educational Media*, Singapore, 2013, pp. 1-11.

[4] Maribeth Gandy Coleman. (2012, September) Georgia Tech Thesis and Dissertations. [Online]. https://smartech.gatech.edu/handle/1853/45845

[5] Alastair Hampshire, Hartmut Seichter, Raphaël Grasset, and Mark Billinghurst, "Augmented reality authoring: generic context from programmer to designer," in *OZCHIComputer-Human Interaction of Australia* , 2006, pp. 409-412.

[6] Ronald Azuma, "A Survey of Augmented Reality," *Presence: Teleoperators and Virtual Environments*, pp. 355-385, 1997.

[7] iOnRoad™. [Online]. http://www.ionroad.com

[8] Orcam. [Online]. http://www.orcam.com

[9] Feng Zhou, Henry Been-Lirn Duh, and Mark Billinghurst, "Trends in Augmented Reality Tracking, Interaction and Display: A Review of Ten Years of ISMAR," in *IEEE International Symposium on Mixed and Augmented Reality*, Cambridge, 2008, pp. 193-202.

[10] Joseph Newman, David Ingram, and Andy Hopper, "Augmented Reality in a Wide Area Sentient Environment," in *IEEE and ACM International Symposium on Augmented Reality*, New York, 2001, pp. 77-86.

[11] Ihsan Rabbi and Sehat Ullah, "A Survey on Augmented Reality Challenges and Tracking," *Acta Graphica* , pp. 29-46, 2013.

[12] Rafael Roberto, Daniel Freitas, João Paulo Lima, Veronica Teichrieb, and Judith Kelner, "ARBlocks: A Concept for a Dynamic Blocks Platform for Educational Activities," in *Symposium on Virtual Reality*, Uberlandia, 2011, pp. 28-37.

[13] Severino Paulo Gomes Neto et al., "Experiences on the Implementation of a 3D Reconstruction Pipeline," *International Journal of Modeling and Simulation for the Petroleum Industry*, vol. II, no. 1, pp. 7-15, 2008.

[14] Oliver Bimber and Ramesh Raskar, *Spatial Augmented Reality: Merging Real and Virtual Worlds*. Natick, MA, USA: A. K. Peters, Ltd., 2005.

[15] Julie Carmigniani and Borko Furht, *Handbook of Augmented Reality*. Boca Raton, Florida: Springer Science & Business Media, 2011.

[16] Visette 45 SXGA. [Online]. http://www.vrealities.com/products/head-mounted-displays/visette-45-sxga

[17] Vuzix Wrap 920AR. [Online]. http://www.vuzix.com/UKSITE/ar/products_wrap920ar.html

[18] Nokia City Lens. [Online]. http://www.windowsphone.com/en-us/store/app/nokia-city-lens/93301a45-5849-4aad-a68e-c7c95df83ca1

[19] Oliver Bimber and Raskar Ramesh, "Modern approaches to augmented reality," in *SIGGRAPH*, 2006.

[20] Microsoft HoloDesk. [Online]. http://research.microsoft.com/en-us/projects/holodesk/

[21] Chris Harrison, Hrvoje Benko, and Andrew D. Wilson, "OmniTouch: wearable multitouch interaction everywhere," in *User Interface Software and Technology (UIST)*, 2011, pp. 441-450.

[22] Hiroshi Ishii and Brygg Ullmer, "Tangible bits: towards seamless interfaces between people, bits and atoms," in *Proceedings of the ACM SIGCHI Conference on Human factors in computing systems*, New York, 1997, pp. 234-241.

[23] Letters Alive. [Online]. http://www.follettearlylearning.com/lettersalive

[24] Mark Billinghurst and Hirokazu Kato, "Collaborative augmented reality," *Communications of the ACM - How the virtual inspires the real*, vol. 45, no. 7, pp. 64-70, July 2002.

[25] Anders Henrysson, Mark Billinghurst, and Mark Ollila, "Face to Face Collaborative AR on Mobile Phones," in *International Symposium on Mixed and Augmented Reality*, Washington, 2005, pp. 80-89.

[26] Aaron Stafford, Wayne Piekarski, and Bruce Thomas, "Implementation of God-like Interaction Techniques for Supporting Collaboration Between Outdoor AR and Indoor Tabletop Users," in *International Symposium on Mixed and Augmented Reality*, Santa Barbard, 2006, pp. 165-172.

[27] Christian Sandor, Alex Olwal, Blaine Bell, and Steven Feiner, "Immersive Mixed-Reality Configuration of Hybrid User Interfaces," in *International Symposium on Mixed and Augmented Reality*, Washington, 2005, pp. 110-113.

[28] IKEA Catalog. [Online]. https://itunes.apple.com/us/app/ikea-catalog/id386592716?mt=8

[29] ColAR Mix. [Online]. http://colarapp.com

[30] Rafael Alves Roberto and Veronica Teichrieb, "ARBlocks: A Projective Augmented Reality Platform for Educational Activities," in *Virtual Reality Short Papers and Posters (VRW)*, Costa Mesa, 2012, pp. 175-176.

[31] Ingress. [Online]. https://www.ingress.com

[32] Junaio - Augmented Reality browser. [Online]. http://www.junaio.com

[33] Layar app. [Online]. https://www.layar.com/products/app/

[34] App - Wikitude. [Online]. http://www.wikitude.com/app/

[35] Tobias Langlotz and Dieter Schmalstieg, "Next-Generation Augmented Reality Browsers: Rich, Seamless, and Adaptive," in *Proceedings of the IEEE*, vol. 102, New York, 2014, pp. 155-169.

[36] Glass. [Online]. https://www.google.com/glass/start/

[37] DAQRI Smart Helmet. [Online]. http://hardware.daqri.com/smarthelmet/

[38] Epson Moverio BT-200. [Online]. http://www.epson.com/cgi-bin/Store/jsp/Landing/moverio-bt-200-smart-glasses.do

[39] Optinvent ORA. [Online]. http://optinvent.com/see-through-glasses-ORA

[40] Juniper Research, "Mobile Augmented Reality : Smartphones, Tablets and Smart Glasses 2013-2018," Networks & Technologies 2013.

[41] Craig Locatis and Hana Al-Nuaim, "Interactive technology and authoring tools: A historical review and analysis," *Educational Technology Research and Development*, vol. 47, no. 3, pp. 63-75, 1999.

[42] Bootstrap. [Online]. http://getbootstrap.com

[43] Zurb Foundation. [Online]. http://foundation.zurb.com

[44] Adobe InDesign CC. [Online]. http://www.adobe.com/br/products/indesign.html

[45] Adobe Muse CC. [Online]. http://www.adobe.com/br/products/muse.html

[46] tracking.js. [Online]. http://trackingjs.com

[47] Headtrackr. [Online]. https://github.com/auduno/headtrackr/

[48] Layar Creator. [Online]. https://www.layar.com/products/creator/

[49] ARToolKitPlus. [Online]. http://handheldar.icg.tugraz.at/artoolkitplus.php

[50] ARToolKit. [Online]. http://www.hitl.washington.edu/artoolkit/

[51] Studierstube Tracker. [Online]. http://studierstube.icg.tugraz.at/handheld_ar/stbtracker.php

[52] Metaio SDK. [Online]. http://dev.metaio.com/sdk/

[53] cplusplus.com - The C++ Resources Network. [Online]. http://www.cplusplus.com

[54] Hyung-Keun Jee, Sukhyun Lim, JinYoung Youn, and Junsuk Lee, "An Immersive Authoring Tool for Augmented Reality-Based E-Learning Applications," in *International Conference on Information Science and Applications (ICISA)*, 2011, pp. 1-5.

[55] Studierstube ES. [Online]. http://studierstube.icg.tugraz.at/handheld_ar/stbes.php

[56] osgART. [Online]. http://www.osgart.org/index.php/Main_Page

[57] Daniel F. Abawi and Ralf Dörner, "A component-based authoring environment for creating multimedia-rich mixed reality," in *Eurographics conference on Multimedia*, 2004, pp. 31-40.

[58] Blair MacIntyre, Maribeth Gandy, Steven Dow, and Jay David Bolter, "DART: A Toolkit for Rapid Design Exploration of Augmented Reality Experiences," in *User Interface Software and Technology (UIST)*, 2004, pp. 197-206.

[59] Ubitrack. [Online]. http://campar.in.tum.de/UbiTrack/WebHome

[60] Iulian Radu and Blair MacIntyre, "Augmented-reality scratch: a children's authoring environment for augmented-reality experiences ," in *Interaction Design and Children (IDC)*, 2009,

pp. 210-213.

[61] Matthias Haringer and Holger T. Regenbrecht, "A pragmatic approach to augmented reality authoring," in *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2002, pp. 237-245.

[62] Kostas Anagnostou and Panagiotis Vlamos, "Square AR: Using Augmented Reality for Urban Planning," in *Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)*, Athens, 2011, pp. 128-131.

[63] Metaio Creator. [Online]. http://www.metaio.com/creator/

[64] Wikitude Studio. [Online]. http://studio.wikitude.com

[65] Dieter Schmalstieg, Tobias Langlotz, and Mark Billinghurst, "Augmented Reality 2.0," in *Virtual Realities: Dagstuhl Seminar 2008*, Sabine Coquillart, Guido Brunnett, and Greg Welch, Eds.: SpringerWienNewYork, 2011, pp. 13-38.

[66] Jinhyuk Choi et al., "k-MART: Authoring Tool for Mixed Reality Contents," in *Mixed and Augmented Reality (ISMAR)*, Seoul, 2010, pp. 219-220.

[67] Adobe Director. [Online]. http://www.adobe.com/br/products/director.html

[68] Adobe Shockwave - Wikipedia. [Online]. http://en.wikipedia.org/wiki/Adobe_Shockwave

[69] Metaio AR Creator Plugin. [Online]. http://www.metaio.com/products/creator/ar-creator-plugin/

[70] Build AR. [Online]. http://www.buildar.co.nz

[71] AR-media. [Online]. http://www.armedia.it

[72] Jürgen Zauner, Michael Haller, Alexander Brandl, and Werner Hartmann, "Authoring of a Mixed Reality Assembly Instructor for Hierarchical Structures," in *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2003, pp. 237-246.

[73] Gun A. Lee, Claudia Nelles, Mark Billinghurst, and Gerard Jounghyun Kim, "Immersive Authoring of Tangible Augmented Reality Applications," in *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2004, pp. 172-181.

[74] Hartmut Seichter, Julian Looser, and Mark Billinghurst, "ComposAR: An Intuitive Tool for Authoring AR Applications," in *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2008, pp. 177-178.

[75] Nikoletta Mavrogeorgi, Stefanos Koutsoutos, Angelos Yannopoulos, and Theodora Varvarigou, "Cultural Heritage Experience with Virtual Reality according to User Preferences," in *Advances in Human-oriented and Personalized Mechanisms, Technologies, and Services*, Porto, 2009, pp. 13-18.

[76] Trien V. Do and Jong Weon Lee, "Creating 3D E-books with ARBookCreator," in *Advances in Computer Enterntainment Technology*, 2009, pp. 429-430.

[77] Tobias Langlotz et al., "Sketching up the world: in situ authoring for mobile Augmented Reality," *Personal and Ubiquitous Computing*, vol. 16, no. 6, pp. 623-630, August 2012.

[78] Guangzheng Fei, Xin Li, and Rui Fei, "AVATAR: Autonomous Visual Authoring of Tangible Augmented Reality," in *Virtual-Reality Continuum and its Applications in Industry*, 2012, pp. 21-24.

[79] Javier Barbadillo and Jairo R. Sánchez, "A Web3D authoring tool for augmented reality mobile applications," in *3D technologies for the World Wide Web* , 2013, pp. 206-206.

[80] Qt Project. [Online]. http://qt-project.org

[81] QML Applications | QtDoc 5.3 | Documentation | Qt Project. [Online]. https://qt-project.org/doc/qt-5-snapshot/qmlapplications.html

[82] OpenGL - The Industry Standard for High Performance Graphics. [Online]. https://www.opengl.org

[83] libQGLViewer. [Online]. http://www.libqglviewer.com

[84] Likert scale - Wikipedia. [Online]. http://en.wikipedia.org/wiki/Likert_scale

[85] Lawrence A. Rowe, "ACM SIGMM retreat report on future directions in multimedia researchACM SIGMM retreat report on future directions in multimedia research," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 1, no. 1, pp. 3-13, February 2005.