2016

Automatic generation of augmented reality guided assembly instructions using expert demonstration

Bhaskar Bhattacharya
Iowa State University

Automatic generation of augmented reality guided assembly instructions using expert demonstration

by

Bhaskar Bhattacharya

A dissertation submitted to the graduate faculty

in partial fulfilment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Major: Human Computer Interaction

Program of Study Committee:

Eliot Winer, Major Professor

Jim Oliver

Rafael Radkowski

Stephen Gilbert

Michael Dorneich

Iowa State University

Ames, Iowa

2016

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Augmented Reality (AR), the ability to present the real-world, overlayed with digital information, is quickly gaining popularity for a number of applications ranging from entertainment to manufacturing. Improved hardware and miniaturization along with robust sensor processing techniques have made AR a viable technology on a variety of high-end and commodity devices. The intention of any AR system is to "augment" a user's reality, with digital content, in order to aid him/her in a particular task. This task can vary from analyzing a factory layout in its intended physical space to following directions on your windshield while driving. Surgeons use AR technology to locate tumors on the fly during operations. The entertainment industry has used AR from hand-held games to major attractions. It has been used in all forms of product development from design to assembly to maintenance to guide users through complex processes. Particularly in manufacturing, these AR guided assembly viewers have been developed and found to benefit end-users considerably. It has reduced the cognitive load on the user, reduced assembly times, increased quality, and reduced training time. AR guided assembly viewers have also, in some cases, removed the advantage that experts have over non-experts.

However, authoring the content within an AR guided assembly viewer is the biggest obstacle to its widespread adoption. Issues hindering authoring include: 1) inadequate software skillset of the person responsible for authoring, 2) complex hardware and software systems, and 3) the process being time and resource consuming. To mitigate the various issues in authoring AR guided assembly content, researchers have proposed a number of solutions

without much success. Solutions include: 1) using markers placed in the real environment as reference for AR systems to reflect the necessary content, 2) creating content solely in a virtual environment, 3) performing gestures that facilitate content creation, and 4) using sensors (e.g., GPS) to locate the view of the user and overlay content.

Based on a literature review conducted of the numerous solutions available, various gaps were identified in the context of AR guided assembly and formed into three research issues. They are: 1) establishing automatic registration of parts during an expert demonstration, 2) ensuring sufficient domain level expertise is incorporated into AR guided assembly instructions, and 3) evaluating the instructions generated by an authoring tool for accuracy and quality.

To direct the creation of new AR guided assembly authoring methods that address the aforementioned research issues a classification technique, established by Hampshire et.al. was used (also known as Hampshire classification). Along with the Hampshire classification, the various factors necessary to successfully author AR content are categorized according to their specific environment, interaction metaphors, and author's domain knowledge. All of these were used to support the design choices made in the proposed authoring technique known as expert demonstration.

This technique automatically generates the various assembly steps and part movements by analyzing 3D point cloud data generated from an expert who performs the assembly (i.e. "demonstrates" it). Through this demonstration, a corresponding skin model is generated and used to detect and remove the expert's hands in the captured frames. Image analysis is also

performed to remove the scene background as well as identify and track each part in the assembly. The analysis results in the detection of each individual assembly step, each part location, and each assembly path. This information is used to generate AR guided assembly work instructions.

Skin detection is a critical component of the expert demonstration authoring technique, as without filtering out the hands detecting, identifying and tracking the parts becomes more difficult. Current skin detection techniques like Gaussian mixture models require clean data and are unable to handle noisy data without overfitting the model. This led to the development of a novel algorithm called particle swarm optimized Gaussian mixture model (PSOGMM). PSOGMM was designed to take only a single input variable describing the amount of noise in the system. It was compared to traditional GMM with two independent datasets.

To illustrate the authoring of AR guided assembly work instructions, AREDA (Augmented Reality via Expert Demonstration Authoring), an AR authoring tool was developed. It was divided into two phases known as the demonstration phase and the refinement phase. The demonstration phase consisted of determining the various calibration parameters (background, area, and skin) as well as recording and processing the assembly demonstration. The refinement phase consisted of taking these automatically generated AR work instructions and fixing mistakes during the demonstration phase, refining orientation and positions of the 3D models, and adding other forms of information like textual instructions or images. AREDA was tested on three assemblies of increasing complexity, namely DUPLO blocks, a 3D printed grip vise, and a laptop.

CHAPTER 1 INTRODUCTION

Central Processing Unit(CPU) and Graphics Processing Unit (GPU) technology have improved over the years at performance rates higher than predicted by Moore's Law [1]. A direct result of these performance rate increases is hardware miniaturization and convergence [2]. This allows for the creation of portable computers such as tablets and smartphones. The increased processing performance of these devices improve the frame rate for generating and rendering virtual content. Consequently, this performance increase has resulted in an overall push for the use and acceptance of augmented reality (AR) [3-6]. For example, Microsoft launched HoloLens (a standalone AR headset), Google invested over half a billion dollars in an AR company, Magic Leap and, according to Juniper research, the number of mobile AR users will grow from 60 million in 2013 to 200 million in 2018 [7]. Thus, augmented reality is fast emerging as a revolutionary new technology that will change the way we see and interact with the world around us.

1.1. Augmented Reality Defined

An augmented reality system is defined as "a system that supplements the real world with virtual (computer-generated) objects that appear to coexist in the same space as the real world. An AR system has the following characteristics: it combines real and virtual objects in a real environment, runs interactively, and in real time, and registers (aligns) real and virtual objects with each other in 3D." [8]

Users view the real world through a screen and the AR system overlays graphical objects including 2D textual instructions and animated 3D models. This seamless overlaying is made

possible by robust sensor processing techniques available today. Examples of sensors commonly used are camera images [9], RFID (Radio Frequency Identification) tags [10], GPS (Global Positioning System) [11] and infrared markers [12]. Figure 1 represents a snapshot of a commercial smartphone based AR system known as Yelp Monocle [13]. The system overlays 2D text and graphics onto the video of the outside world in real-time. These overlays provide the name of a location of interest, the type of location and the distance from the user. It also provides a virtual compass representing the current bearing of the user acquired from the GPS and location of the phone. The system has a database of locations with their corresponding GPS coordinates. As the user moves around changing his/her GPS coordinates and compass bearing, the various virtual objects update their information in real-time based on that information. This AR system combines a real world image feed with 2D text and graphics, updates in real-time and registers the real world and the various virtual objects based on the user's location and bearing.

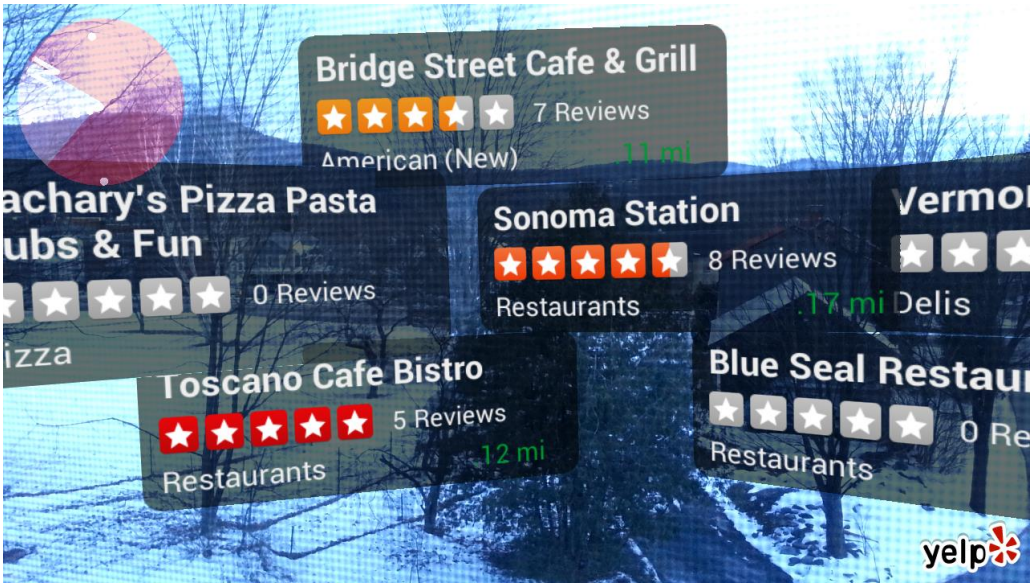


Figure 1. Yelp Monocle screenshot.

There are a number of other areas where AR technology has been used successfully. Virtual heritage [14] has been a proponent of AR, using it to display cultural heritages that may or may not be present today. It is able to educate the public and provide a more personable understanding of the local customs. Architects use AR to display their designs to interested parties [15]. It facilitates in assessing the reaction of the public before actually building a design. AR has also provided educators a new visual medium to teach students topics such as chemistry [16] and mathematics [17]. Researchers have reported enhanced learning effects by students compared to traditional teaching methods. With the increased use of AR, students have improved their spatial skills as well [18]. In medicine, researchers have used a Microsoft Kinect and aligned a virtual skeleton with a real person to teach anatomy [19]. AR has also been successfully used to assist laparoscopic surgery by providing specific tumor locations [20].

Commercially, AR has been used to move advertising beyond two dimensions. In the entertainment industry AR has made its mark from hand held console games like “Eyepet” and “Ghostwire” to large attractions like Disney’s “Toy Story Midway Mania” as shown in Figure 2. In addition to the entertainment industry, AR has caught the attention of the manufacturing sector. The automotive industry in general is using AR in all aspects of design, maintenance and production [21]. For example, General Motors is experimenting with an AR windshield to provide useful information to drivers to increase safety.

Figure 2. Disney's Midway Mania attraction. The arrows being shot are augmented.

1.2. AR Guided Assembly

AR technology is used by design and manufacturing companies [22] for: 1) evaluating interior vehicle designs [23], 2) creating an environment to facilitate collaboration [24], 3) providing assistance during assembly, disassembly, and maintenance of products [25], and 4) contributing relevant information for the safe and better use of a product to an end user. This dissertation focuses on the content that AR guided assembly viewers use to equip a user with well-defined, well-placed instructions in the real environment. Ideally, all the information necessary to perform a complex bench-top engineering assembly task would be available in an AR environment and a user would not have to refer to a manual.

Figure 3 represents an example of an AR guided viewer to aid in the assembly of a pump [26]. The system provides a top down view of the assembly area displayed on a screen. At each step the requisite textual instructions are provided with corresponding animated 3D models. These animated 3D models present specific visual instructions of the actual movements that must be performed to complete the assembly. The system combines the real world (assembly

area with various real parts) with the virtual world (textual instructions and animated 3D models), runs in real time, with both environments registered in 3D. This registration between the virtual and real environments is illustrated by the accurate size and location of the 3D models with respect to their real world counterparts. Without this registration, either the 3D models would just be floating with no apparent logical connection to the real world or the movement of the parts would be heavily constrained making the system inflexible.

Figure 3. Augmented reality system to aid user in assembling a pump. Textual instructions are provided on top while a virtual demonstration is presented [26]

Nilsson [3] studied the acceptance of AR instructions in real work settings and users preferred AR over conventional methods of instruction. Even with poor interaction most users had a positive experience using AR systems [27]. It is at least comparable to conventional techniques available and reduces the cognitive workload on a user [28]. In some examples it has allowed non-experts to perform complex tasks, previously accomplished only by

disciplinary experts [29]. However, this is not the case by simply implementing AR. Care must be taken as to the interface presented to a user and the content within.

Specifically in terms of AR guided assembly viewers, Boud [30] compared five guidance techniques on an assembly task: 1) Conventional 2D engineering drawings, 2) Desktop virtual reality (VR) using a 2D display , 3) Desktop VR using a stereoscopic display, 4) Immersive VR using a head mounted display and 5) AR guided assembly. The result was that the AR guided assembly viewers were the most effective. Users completed the studied assembly task approximately eight times faster than using conventional 2D engineering drawings as measured by mean completion time. The AR task also performed significantly better (30 secs vs. 52 secs) than the best VR condition (Desktop VR using stereoscopic display). There was no significant difference between the various VR tasks. In addition, Henderson [31] established that AR guided assembly significantly outperformed conventional instruction delivery with error reduction of over 60% and a 100% improvement in assembly speed.

1.3. Problems with Authoring AR Guided Assembly Content

AR has shown many benefits in a wide variety of areas but authoring AR guided assembly content brings with it significant challenges:

- Registering 3D models with the real world accurately is very difficult.
- Content creation is time consuming.
- The hardware used may not have sufficient processing power to run in real time.

At this point, the distinction needs to be made between an authoring tool and an AR guided assembly viewer. The responsibility of an authoring tool is in content creation. It should be able to load in virtual content (2D text, 3D models, etc.), perform 3D transformations on them (rotation, translation, and scaling), register the content and export it out along with the registration information. This registration information is the relative positions of the virtual content to processed sensor data available. An AR guided assembly viewer then displays this content to a user to allow them to complete the assembly process.

The system shown in Figure 3 is an example of an AR guided assembly viewer. Each step of the assembly process is shown to a user through text, animated 3D models, etc. For example, in the figure the 3D model of the cylinder block is registered correctly to the partial assembly. Thus, the user understands where this part must be before proceeding to the next step. As registration is a critical component to successfully using AR, it will be discussed in more detail. Proper registration of virtual content in AR requires a number of issues to be resolved:

1. Virtual models (i.e. parts) need to be created in a freeform 3D modeling package such as MAYA [32], Studio Max [33], or Blender [34].

2. Once the models are available, they need to be positioned in the real world with the correct graphical transformations (i.e. translation, rotation, and scaling).

3. These 3D models need to be exported from an AR authoring tool in a format acceptable by an AR guided assembly viewer. Improper conversion will introduce visual and animation path errors and lead to mistakes during assembly.

Registration is typically done by using external physical markers to establish a reference coordinate system in the real-world such as cards or paper with unusual markings on them. In most manufacturing sites, adding physical markers to an already crowded area of tools, parts, and personnel is not ideal. This often causes hindrances to the assembly [35]. Not adding markers requires the use of added computer vision techniques increasing the processing resources and complexity of the AR system. While these are complex issues, more exist when considering authoring AR guided assembly content. Tasks such as interaction methods also bring their own set of challenges that need to be addressed. Scale this example up to a real assembly line where there are hundreds of steps to generate and changes to these steps are done on a frequent basis, and the challenge to successful AR guided assembly authoring becomes enormous.

In addition to technical complexities, successfully authoring AR guided assembly content requires other disciplines as shown in Figure 4.

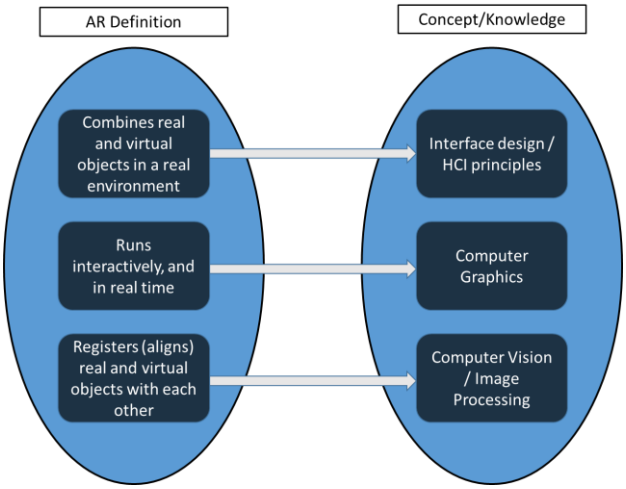


Figure 4. Correlation between various parts of AR definition and disciplinary expertise required to address them.

The figure presents the correlation between the basic elements of AR and the disciplinary expertise in which they fall under. Thus, authoring AR guided assembly content requires:

- A variety of disciplinary concepts to be understood
- The translation of assembly instructions to their corresponding graphical equivalent (e.g., tightening a bolt with a wrench would translate to an animated bolt somehow showing it is being tightened)
- Registering the graphical overlays appropriately with the real world
- Evaluating the resulting AR guided assembly instructions.

As a result, in most manufacturing scenarios, authoring this content is broken down into multiple responsibilities and delegated to different teams. One team may be responsible for creating the graphical instructions (3D modeling experience), another may be responsible for registration (computer vision experience) and another may be responsible for evaluation.

The work of this dissertation will focus on techniques to address the authoring challenges outlined. Fundamentally, the process for authoring content will be changed. The assembly process will be performed, or "demonstrated", by an expert in a setting similar to, if not exactly the same, as the bench top where they normally work. This demonstration will be recorded using a RGB+D (i.e. color and depth) camera. Using computer vision techniques, each individual step is identified, the path movement of the part is created, the location of the individual parts is registered, and sufficient textual feedback is created. This research also aims

to eliminate the use of artificial markers during the authoring process and automatically generate the AR guided assembly instructions from an expert demonstration.

The remainder of this dissertation is organized as follows: Chapter two presents a review of the state of the art AR viewers available and its impact on the various fields of study. Chapter three presents a classification and a categorization of the factors making up an AR authoring process. The gaps in research are presented along with the specific research issues that will be addressed. Together, they are used for guiding the design choices made and evaluating the proposed techniques for authoring AR guided assembly content. Chapter four describes the new skin detection algorithm PSOGMM that enables expert demonstration. Chapter five illustrates AREDA and the various steps involved to get the AR work instructions automatically. Chapter six and seven discuss AREDA, its limitations and the future direction that needs to be taken.

CHAPTER 2 LITERATURE REVIEW

This chapter presents a literature review of AR authoring tools and viewers used in a variety of disciplines and industries. The strengths and limitations of each are discussed.

Figure 5 presents an example from Scene Designer [36], an AR authoring tool with an AR previewer. Scene Designer uses an artificial marker for augmenting virtual content in the real environment. An AR previewer is an AR viewer built within an authoring tool and is used to preview the content in an AR environment. Thus, an AR viewer is a standalone application while an AR previewer is part of an authoring tool. A common theme among many of the viewers presented involve registering the virtual content with an artificial marker via GUI based interactions. These interactions involve the loading and placement (translation, rotation, and scale) of virtual content. Artificial markers typically consist of 2D AR Toolkit markers [36-37]. These markers are bi-tonal (black and white) and square shaped to facilitate ease of detection and registration even under poor lighting or in an image slightly out of focus. An example of this marker is shown in the middle of Figure 5. Other forms of artificial markers like optical trackers and RFID tags have also been used.

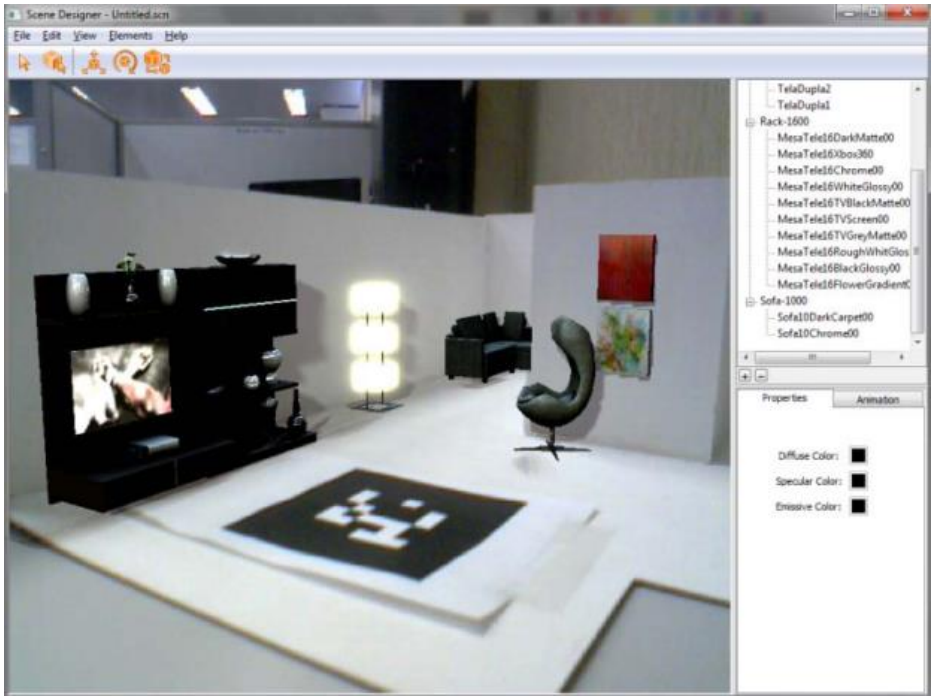


Figure 5. Scene Designer[39] uses an AR toolkit marker for content creation in an AR environment.

2.1. AR in Education

Education is an excellent example of the potential for improvement by AR [4]. Researchers have found that AR viewers have reduced cognitive load on users [40], maximized transfer of learning [18], and facilitated collaboration among peers [41].

Figure 6 presents an example of an AR viewer created to teach music [42]. Users wear a head mounted display allowing an augmented view of the keyboard showing the keys to press at the correct time. AR has also been used to teach math [17], art [41,42], astronomy [45], and chemistry [16]. Performance evaluations conducted on Augmented Chemisty (AC) instruction showed that students preferred it over conventional ball and stick models. Students reported

higher levels of satisfaction (59% vs 27%), visualization (50% vs 27%), content availability (45% vs 27%), and future use (40% vs 27%).

iFiction [46] is an authoring tool with an AR previewer that promoted literary creativity. Using smartphone sensors such as GPS and a compass, the viewer allowed students to author and share AR scenarios among themselves. AR GameBuilder [47] is a similar desktop based AR authoring tool that helped students create “serious games” intended to teach science.

Figure 6. AR viewer used to teach the piano [42]. AR Toolkit markers are used to register the virtual keys with the real keys.

Educ-AR [48] is an AR authoring tool with an AR previewer that facilitated content creation using a GUI. AR Toolkit markers were used to register the location of preloaded 3D models. The authoring tool allowed a teacher to load 3D models, manipulate them and register

them to a particular marker. Educ-AR was tested with 20 people and results showed that 90% of the subjects were able to use the markers and associate 3D models with them. Only 60% were able to load 3D models and whether it was bad interface design or improper 3D model formats, the researchers did not provide any detail as to why 40% of the users failed to load 3D models. Some teachers were unable to use the authoring tool due to poor lighting, which prompted the researchers to add a calibration phase. This addition of a calibration phase attempted to remove some of the complexity of authoring. Calibration involved finding a grayscale threshold value relative to the environment lighting to allow the marker to be distinguishable. This step introduced the concept of environment lighting and thresholding. These concepts are specifically computer vision based and are used in AR Toolkit marker based systems (in this case Educ-AR). Adding a calibration step allowed for human error as well as introduced more complex AR concepts that were not within the expected domain knowledge of educators. If for any reason Educ-AR doesn't work, the educators will not be able to debug the system. At best, they will restart the whole process and try again.

ARAS-SP [49], an authoring tool with an AR previewer, used a combination of AR Toolkit markers and manual file editing to realize an AR based educational game called Q&A-AR. Q&A-AR is a multiplayer car racing game that used question and answers as the mode for moving a virtual car. Markers were used to register the location of the game and also as a pointer in the AR environment during the authoring process. An alternative to authoring in the AR environment was to directly edit configuration files on a stationary computer. This authoring tool was assessed by a user study. Variables such as game interaction (3.8/5, 5 being the best), game interface (3.5/5), visual aspects (3.7/5) and audio aspects (4.3/5) were evaluated.

However, the participants of the study were undergraduates and not teachers. The researchers also do mention that in the future the author would have to use 3D modeling software to be able to create personalized 3D models if used for other applications.

Jee et. al. [50] provided the same functionality as the previous example but supplemented it with methods using natural features to register content during both the AR authoring and viewing process.. Rahman et. al. [51] extends Jee's visual feedback process, i.e. seeing the AR content displayed on the screen via the AR previewer, by including haptic (via a Falcon haptic device) as well as audio feedback. The haptic feedback actually provided an interesting method of physically realizing the location of content in 3D space.

De Lima [52] used AR Toolkit markers and sketches to create an AR based story. These markers were used to define characters while predefined sketches were used to display animated content. Using these two methods, stories were created by high school students who also evaluated the authoring tool based on a five point Likert scale compared against a menu-driven authoring tool. The sketch-based authoring tool was considered to be usable, met user needs, and was engaging. On the other hand, the interaction effectiveness was rated lower than the menu-driven authoring tool.

Teachers have been able to not only teach using AR but also create interactive examinations [53]. Using a GUI based authoring tool teachers were able to associate a list of questions and answers and corresponding 3D models with AR Toolkit markers. This data was imported by a student's AR viewer who then provided answers within the AR environment. A

qualitative analysis of user feedback showed that the authoring tool and AR viewer was well received by both students and teachers.

2.2. AR in Medicine

The medical field has also been an early adopter of AR. For example, it has been used successfully in the context of assisted surgery as shown in Figure 7. The surgeon has the image of a patient's MRI scan projected on top of their body during an operation to aid in incision location. Laparoscopic surgeons have also used AR to locate target tissue (e.g., tumors and nodules) while performing surgery [20]. Dentists used AR to visualize implants during oral surgeries [54].

Figure 7. MRI scan registered with patient body during an operation to aid in position of incisions.[20]

AR in a surgical environment is very difficult because there is a constant need for all the equipment to be sterilized. Almost all registration needs to be done off-line using optical trackers. Using artificial markers of any kind isn't very convenient as: 1) there is a lot of equipment to be marked, 2) there are not convenient locations to place markers that don't interfere with medical personnel and 3) occlusion of markers leads to improper registration. There is a lot of movement in a typical operation between the various medical personnel and equipment involved. Surgeries cannot be altered or stopped because of an AR system issue. Other AR assisted minimally invasive surgeries are mentioned in the Journal of Minimally Invasive Medical Technology [55]. For example, Computed Tomography (CT) scans were used to overlay information to a forensic expert to aid in judging cause of death [56].

AR has also been used in a teaching capacity for the medical profession. For example, the VirDent viewer used AR to teach students teeth reduction dental preparation [57]. Mirracle [19] used AR technology to teach anatomy. Using the Microsoft Kinect [58] (providing skeletal co-ordinates of a person in terms of image coordinates), the viewer is able to overlay a 3D model skeleton on a user's body. Other applications include a mobile application for teaching ethically sensitive topics in medicine [59], intravenous injection simulation in veterinary medicine [60], and surgical simulations [61].

Even with these advances, there aren't many authoring tools for using AR in medicine. Figure 8 provides one of the few, YouMove [62]. It was a GUI based authoring tool that records the skeletal movement of an author via a Kinect. The resulting recording was then modified similar to a video editing tool. Users viewed the recorded and edited skeletal movement of the author and tried to mimic him/her. The user movement was then evaluated based on similarity

scores between the user and recorded author movement. Though YouMove isn't strictly used for medical applications, it was used in providing patient movement feedback during physiotherapy.

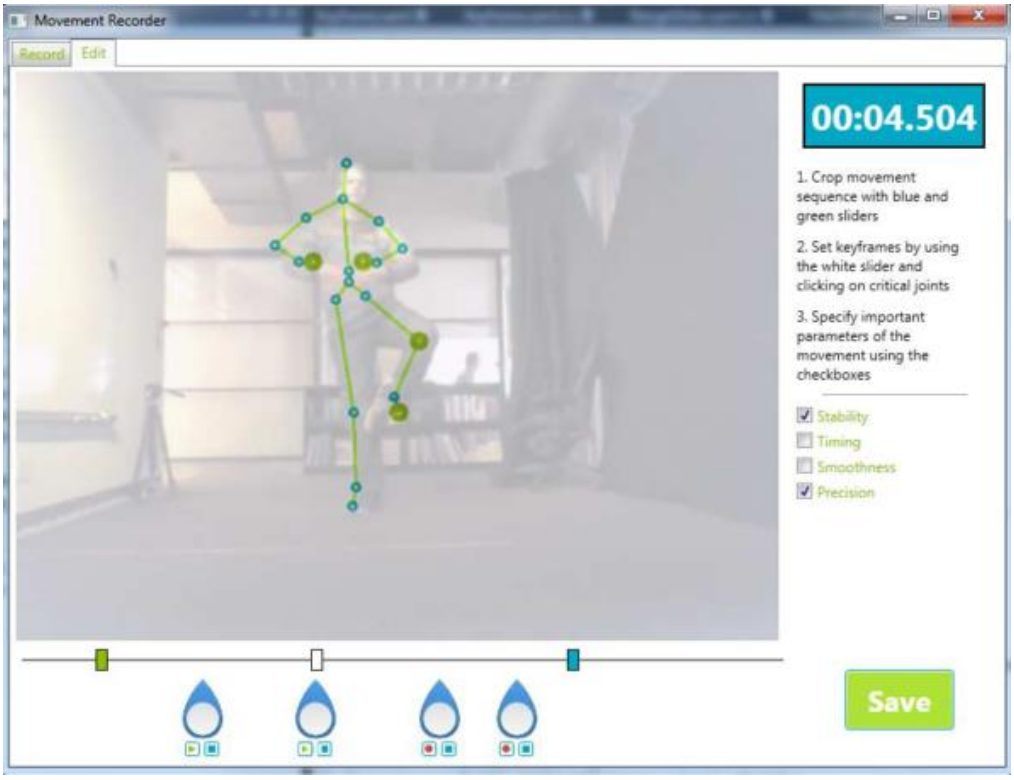


Figure 8. YouMove [62] authoring tool used to record and author skeletal movement.

2.3. AR in Entertainment

AR has been conceived in a variety of ways in the entertainment industry. Gaming, advertising, and multi-media attractions are just some of the areas in which AR has made an impact. It has become a "new media experience" [63] enjoyed by a multitude of people.

Massively multiplayer online role playing games (MMORPG) have been conceived in an AR environment where users are able to play games based on the real environment they are in [64]. CloudRidAR [65] had both single player functionality and multiplayer functionality.

Registration of virtual content was based on single line detection (i.e. virtual content is registered based on the location of a single line) and could only be played against high contrast regions, such as a whiteboard. The most prominent example of AR gaming was the development of ARQuake [66] in 2002. Using a combination of a backpack computer and HMD, researchers were able to register the real environment via GPS compass sensors and "play Quake in real life".

Wetzel et. al. [67] have established and evaluated guidelines for games created on mobile platforms using GPS and compass sensors to desktop based games using AR markers. Guidelines vary from the obvious (use the real environment and don't stay digital) to the unique (use potential technical problems as part of the game). For example, if tracking in dark places is difficult then create a situation where finding yourself in a dark place negatively affects the game. Similarly, Villalta [68] provided guidelines for classroom multiplayer presentational games.

There are also a number of commercial applications available in the market today. Google's Ingress [69], a GPS based smartphone game, transforms the user's real world into a point gathering game. Figure 9 provides a screenshot of Ingress, showing the various green and blue dots a user must use to score points. Peoleo Entertainment's Drakerz [70] augments popular card games with 3D animations, adding another dimension to conventional gameplay. Sony Playstation's PulzAR [71] involves moving cards augmented with 3D content to solve puzzles. As with most AR viewers, all of these games rely on either artificial markers or precise sensors.

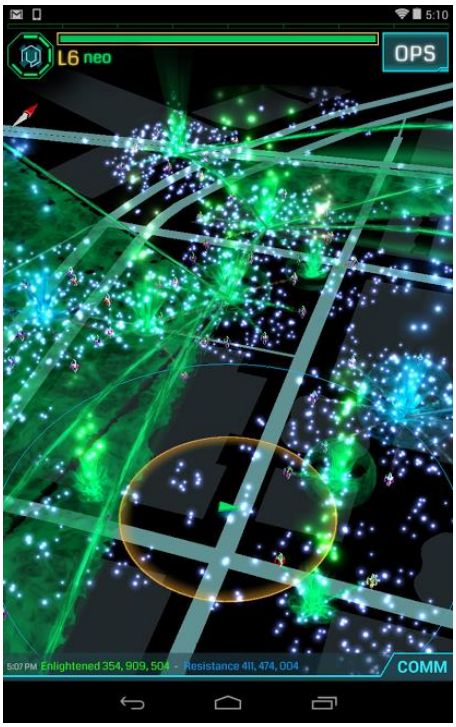


Figure 9. Screenshot of Google's Ingress[69]

In the realm of digital art, CollArt [72] is a tool which used natural features to create 3D AR based collages using virtual objects and primitives wrapped in user generated textures (e.g., a cube with photos wrapped on all sides). This is a smart phone application using Vuforia [73] as the 3rd party software for registration. Museums have used AR to augment experiences with textual information, audio, and video [72-74]. For example, Merges [77] showcased different forms of art and expression in a digital medium. Body performances, exhibitions, and sketches are just some of the conventional forms of art digitally enhanced with AR technology.

SportVision [78] is a company which has revolutionized the way we see sports on television. Providing augmented information on screens like the first down line in American football, the offside line in soccer (Figure 10), and driver identification during NASCAR races.

Other commercial applications include the Hawkeye [79] in tennis and cricket used for detecting outside of a boundary line.

Figure 10. SportVision [78] uses AR technology to show the off-side line during a soccer match.

2.4. Miscellaneous Uses of AR

In addition to education, medicine, and entertainment there are many other areas where AR has been used.

In architecture and urban planning AR is used in a wide variety of applications. SiteLens [80] used AR Toolkit markers with a HMD to augment varying urban scenarios such as excessive pollution due to traffic congestion. Schattel et. al. [15] demonstrated an AR authoring tool that reflects annotations made in a real environment via a smartphone onto a virtual environment. Registration was done in an off-line manner mapping a pre-constructed virtual environment with location based sensors (GPS and WiFi) present on a phone. The authoring tool facilitates collaboration between authors in different physical spaces. For example, consider a fire safety manager of a particular building wanting a virtual representation of the location of all fire extinguishers available and their corresponding expiration dates. Dong et. al. [81] created a

mobile augmented reality system (backpack computer and HMD registered with the real environment) called ARMOR and used it for excavation collision avoidance. A similar AR viewer was demonstrated by Schall [27, 80], the only difference being the display used was on a mobile tablet. SquareAR [83] was a simplistic authoring tool for urban planning. Using AR Toolkit markers to augment a virtual grid and basic GUI interactions, the authoring tool allowed the user to generate a scene. Its ultimate use is to provide residents near empty or planned restoration areas the freedom to not only visualize the intended plan but also change things around as they deem fit.

In the realm of military training, the Battlefield Augmented Reality System (BARS) [82, 83] was a hands free system for running different types of training scenarios for soldiers. Command center [85] was an AR viewer using artificial markers to register virtual content. The AR viewer was created to visualize training scenarios and also allow on the fly changes in scenarios when required.

Advertising is another realm where AR is providing a new dimension. AR has allowed advertising to move from conventional 2D prints (images) to “interactive prints” usable with smartphones. Again, these prints become interactive by virtue of registering them in an offline manner with a video or a web link. Layar [86], Blippar [87], and Aurasma [88] are just a few examples of commercial software providing this functionality.

2.5. AR in Manufacturing

Manufacturing industries use AR technology to aid in design, assembly, and maintenance operations [20,21]. In the domain of maintenance and assembly, AR has been

used in a number of projects. Due to the similar nature of the maintenance domain (process of disassembly, possible part replacement, and reassembly) and the initial assembly domain, there are multiple publications which treat them as one entity. Thus, in this section, the reader may find discussions of research conducted in either domain treated as one.

In multiple AR guided maintenance viewers, registration of virtual content is typically done with artificial markers. Knopfle [89] used AR Toolkit markers for maintenance tasks on a BMW engine. De Marchi [90] demonstrated an AR guided maintenance viewer using Vuzix glasses and AR Toolkit markers to display maintenance tasks on an aircraft. Gimeno's SUGAR [91] used AR Toolkit markers to register virtual procedures for a motor engine. Schwald [92] tracked the user's HMD and real world objects of interest using optical infrared markers. With this information researchers registered the virtual content with the real world. AR puppet [93] revolves around the human computer interaction aspect of AR guided maintenance viewers. In a demonstration, researchers showcased a virtual repairman registered by an AR Toolkit marker in the real environment, talking to the end-user and explaining the steps involved in the maintenance task. Similar examples of AR guided maintenance/assembly viewers are presented in [20,26,93-95]. Webel et. al. [96,97] provided vibro-tactile feedback using a haptic bracelet along with an AR guided maintenance viewer. A user study was conducted on a typical maintenance task in an industrial context. Researchers claimed that adding this feedback improved performance time and decreased the number of unsolved errors when compared to conventional video instructions, but without use of the vibro-tactile feedback. Zhang et. al. [99] used RFID tags to mark the parts in two assembly processes: 1) assembly of a 3D puzzle and 2) assembly of a computer mouse. The RFID tags provided only the position and not the

orientation of the parts. The orientation was determined by using in-house image processing techniques based on 2D corner locations. Yuan et. al. [100] used markers to display instructions on screen and a virtual interactive tool to interact with the AR environment. The virtual interactive tool was essentially a single colored pen with a ball at its tip. The pen was distinguished from the background based on its unique color. Authoring was also conducted in the same environment with different options represented by a GUI within the AR environment. The pen was used to trigger buttons by hovering over them for two to three seconds. An example of the AR viewer was presented for a toy roller-coaster assembly.

Mobile and markerless AR viewers are dependent on natural features for registration and are typically displayed on tablet computers. Natural features are computed from representative images of the real environment at varying angles. This is a time-consuming and computationally intensive task that is difficult to achieve without errors. These viewers are easily affected by improper lighting or drift errors from the initial calibration. Platonov et. al. [101] showcased an example of a mobile and markerless viewer using features suggested by the Shi-Tomasi operator [102]. The researchers calibrated an AR viewer using images (projections) generated from CAD models of objects present in the natural scene. The advantage being that a lot of images can be generated from the model relatively quickly for registering the physical environment. However, no results were provided on the accuracy of the registration. The disadvantage was that the CAD models needed to be of sufficient fidelity to represent their physical counterparts accurately. Since the CAD model was an idealized version of the physical model, features of the real environment such as dirt and faded paint caused difficulty in accurate registration. Comport [103] used natural features (processed edges and

points) to track objects and provided example AR scenarios where different objects were tracked under varying background conditions. This ability to track objects in 3D space allowed for the automatic registration of virtual content in the AR environment. An experiment was performed where an object was held by a robot arm and tracked. Comparisons were made between the tracked readings to the ground truth robot odometer readings. Results indicate that there was less than 1mm translation error and less than 0.33 degrees rotational error. The disadvantage was that careful manual initialization was required to register the initial frame. Gene et. al. [104] used AR Toolkit markers as ground truth data to register natural features from varying angles of images. These natural features were used for the final registration of virtual content in the AR viewer.

The authoring process involves the creation and registration of virtual content for an assembly or maintenance step. This virtual content is typically authored independent of the location of the user's view in the AR environment. Context aware authoring tools allow the author to place rules on the virtual content depending on the location of the user's view. For example, Figure 11 shows an authoring environment where the same object is seen at two different angles. The author is able to provide a rule based on the distance of the end-user from the task to be performed (i.e. the context of the end user). In the left image, white text and a circle is displayed to let the user know that a task has to be performed in this location. As the user gets closer (in the right image) or the context changes the text and circle turns green providing information on not only what to do but also precisely where it needs to be done on the model. This change in adding rules to authoring content based on context makes the instructions more easily understandable. These rules are taken in by context aware AR viewers

that locate the global position of the camera viewing the assembly or maintenance task via calibrated sensors. Researchers have used AR Toolkit markers as well as natural features to develop context aware viewers [24,104-111].

Figure 11. Example scenario of a context-aware viewer [107] which provides different instructions at different times. In the left image a generic location and instruction is shown. In the right image much more specific location and color changes are used.

Wang et. al. developed a 3D hand interaction [113] technique to author virtual content in an AR environment. The researchers used this technique to design and assemble a toy car [114] for simulation purposes.

AR viewers in the manufacturing industry have shown to be intuitive and cause less strain, due to decreased neck movement, during maintenance tasks [115]. Tang et. al. [116] asked 75 participants to assemble a set of DUPLO blocks involving 56 steps. Participants were divided into four treatments (representing the instructional delivery system): 1) printed media, 2) 3D models in a virtual environment via LCD screen, 3) 3D models in a virtual environment via

HMD, and 4) AR viewer via HMD. Assembly time and assembly error rate were measured along with the cognitive load of each system via NASA's Task Load Index (NASA TLX) [117]. NASA TLX is a set of variables measured via a self-reporting questionnaire indicating the perceived workload of a task. Results showed that the AR viewer (treatment four) improved assembly time by four minutes over printed manuals, reduced assembly mean error rate from 12 to three over printed manuals, and reduced cognitive load with the lowest NASA TLX rating of 10.0/20.0 on the participant. Diaz [118] compared the learning rate of both virtual and augmented reality guided assembly viewers with the assembly of a milling machine. The researchers found that there was no significant difference in the learning rates of either of them against conventional learning techniques (eight training sessions using only the real components). Other statistics such as time of the assembly task were not discussed. Fiorentino [119] confirmed that using AR for maintenance tasks on a motorcycle engine versus paper based instructions significantly improved a user's execution time by 38% and decreased error rates by 92%. Henderson [31] provided a similar research experiment using a Rolls Royce Dart 510 turboprop engine as the maintenance task. Two methods of instruction delivery were tested and compared: 1) LCD with 2D instructions and 2) AR viewer. The AR viewer was 47% faster in mean execution time and mean accuracy was 34% better than the LCD method. A qualitative review conducted by Weidenhausen et. al. discussed the evaluation of ARKIVA [120]. They demonstrated that AR is very much liked by end-users and there is a strong demand for this by industry. Hou [121, 122] provided a summary of all the performance issues related to AR guided assembly viewers. The researchers found that users were initially disoriented with AR viewers due to the floating nature of objects and immersion was not fully achieved. Limited system lag is tolerable,

however, with increased jitter (caused by system lag) there is increased cognitive load and distraction AR viewers still require higher fidelity virtual content which seamlessly become part of the user environment i.e. are registered more robustly.

2.6. Authoring AR Guided Assembly

2.6.1. Priorities

The review of the various AR authoring tools discussed in the previous sections led to focusing the priorities of this dissertation towards developing techniques that facilitate intuitive authoring of AR guided assembly content. Based on industry preferences and gaps in current AR guided assembly viewers a checklist of priorities was developed. Any authoring techniques developed should be able to:

1) Register virtual content automatically and provide visual feedback of the AR environment in the same context as an end-user would view it.

2) Establish domain knowledge of the AR authoring process to consist of people with common skill sets.

3) Use appropriate markers available. Having to add external markers can be difficult and obtrusive.

4) Generate the AR guided assembly instructions in as fully an automated manner as possible.

Authoring techniques developed need to facilitate intuitive interaction between an author and the authoring tool.

2.6.2. Current AR guided assembly based authoring tools

A common solution to authoring AR guided assembly content is to use a virtual representation or simulation of the real world and author instructions through it. Knopfle [89] presented a desktop GUI that allowed an author to register virtual content (i.e. textual information and 3D models) with AR Toolkit markers in the real environment. The authoring tool provided the ability to load both the 3D content and the image of the AR Toolkit markers and link them accordingly. Thus, the AR guided assembly viewer was able to distinguish the artificial marker and display the appropriate 3D content. This AR guided assembly authoring tool was designed for technical writers. The researchers used a time driven format (like video editing) to represent the various instruction steps. However, most AR guided assembly viewers are event driven in that they perform actions once an event is triggered by the system. Since no evaluation of the authoring tool was performed it is unknown whether there was any disparity between the time driven concept of the authoring tool versus a more typical event driven concept. Espindola [123] and The *Authoring Wizard* [124] extended Knopfle's work with the added benefit of providing an AR previewer. This means that the virtual content being authored was immediately visualized in the context of an end-user allowing for corrections and on-the-fly changes during the authoring process. SUGAR [91] used an image of the assembly area as a base and allowed the author to generate and place artificial markers virtually on the image. Two concepts drove this work. First, the registration was done in the virtual environment and all the assembly steps were authored as such. Second, the author printed off the generated markers and placed them in the same location as he/she did in the corresponding image. It is unclear whether SUGAR provided real-time feedback via an AR previewer. The authoring tool

also captures the depth information of the assembly area, which was used to provide a more natural visualization of parts through occlusion to provide a sense of depth. When one object is hidden or occluded behind another then our brains perceive the hidden object to be farther away. Figure 12 shows a virtual dog placed behind a real cup. The fact that only part of the dog is visible provides a sense of depth in the scene. Further details on occlusion cues and rendering techniques to achieve this is in AR is described by Shah et. al. [125]

Figure 12. Example of occlusion cues used to provide depth. The virtual dog looks like it is behind the cup (courtesy Blair Macintyre).

Wang et al. [126] demonstrated a developed authoring process via a desktop software application. This authoring tool allowed the placement of virtual parts as well as other visual cues in the virtual environment mimicking Computer Aided Design (CAD) tools. It was designed for users well versed with CAD software. The registration of various virtual parts were based on

an off-line process using natural markers. This off-line process involved capturing certain representative images of the base. Natural features from these images were compared to the real time video feed to find the orientation of the camera and used to register virtual parts. The time taken for this off-line process, as well as the flexibility of the AR guided viewer (accuracy of registration), was not studied. Makris [127] showcased a GUI with virtual assembly parts constrained to being orthogonal (or collinear) to each other at each authoring step. No information was provided on the registration method used. A significant drawback to these approaches is lack of real-world behaviors. Physical constraints of the real world are not always accurately represented in a simulation, which can lead to oversights and mistakes in assembly sequences.

Ong and Zhu [105] tried to overcome this by creating an AR guided maintenance authoring tool with a number of different methods. First, instructions can be authored using a GUI to register textual information and 2D artifacts in a virtual environment. Registration occurs by processing raw data from a number of real-world sensors. In addition to the GUI based authoring method, another authoring method was also developed that provided visual feedback via an AR previewer. AR Toolkit markers were used to register the location of the maintenance site. Fingertip tracking and AR Toolkit markers were used to author the instructions in real time on the actual job site where the assembly would occur. The last authoring method was a GUI displaying the end-user's video feed with the author remotely providing 2D instructions in real-time. This design was used for real-time assistance during assembly if problems arose. The researchers used only 2D labels as augmentable content as 3D models were not available. This would not be suitable on large-scale processes, as a large

number of human assemblers would require a large number of subject matter experts ready to provide feedback. Similar research was conducted by Wang [128] but with the addition of animated 3D models.

Extending this work, Jo et. al. [95] used a knowledge base system (KBS) in conjunction with typical AR concepts. The KBS is a database of all instructions created by parsing available technical documentation of an assembly. The tool allowed an author to take representative images of an assembly area and annotate them based on the KBS available. Content authored for an AR guided assembly viewer was registered using natural markers. The advantage of using the KBS is that the author had all the appropriate technical documentation available. Authors did not have to refer to technical documentation and were able to avoid unfamiliar technical language. Evaluation of the performance of the natural marker tracking and ease of use by the end-users was not performed.

In addition, researchers have also used several commercial and open source tools and software libraries for AR authoring. Westerfield [129] used ASPIRE, an online intelligent tutoring authoring system, to create content for their in-house AR guided assembly viewer. Commercial software packages may be used together to author and register AR guided assembly content as in the case of Ramirez [130]. For example, Unity [131] and Vuforia [73] are commonly used together to create a functioning AR guided assembly viewer. Vuforia is responsible for all registration between the scene and the real world while Unity is used for authoring content. Touch designer [132], a 3D real-time modeling software, in conjunction with AR Toolkit markers "tries to bring AR out of the lab and into the hands of artists, designers, and other creative practitioners" [133,134].

Table 1. List of solutions and their management of priorities

AR guided assembly authoring solution	**Priority 1** ***Registration and real-time preview***	**Priority 2** ***Establish domain knowledge***	**Priority 3** ***Avoid artificial markers***	**Priority 4** ***Generate AR guided assembly instructions automatically***
Wang [126]		X	X	
Makris [127]		X	X	
Ramirez [130]	X			
Espindola [123]	X	X		X
Authoring Wizard [124]	X			X
Ong and Zhu [105]	X			X
Jo [95]		X	X	X

Table 1 provides an overall view of the various authoring solutions and whether they address the various priorities identified earlier. Priority one, registration and real time preview, has been addressed by a few solutions but only with artificial markers. Priority two, establish domain knowledge, is difficult to accomplish. A few of the current projects addressed it by using existing documentation and interviews with domain experts. This area is critical and requires advances in research to address it properly. Priority three, avoid artificial markers, was successfully done by only two projects. However, in both cases it required an off-line calibration step, which generally takes time and resources to complete. Priority four, automatic generation

of AR guided assembly instructions, is challenging. While several of the AR guided assembly viewers claim to do this, none were evaluated for speed, quality, and accuracy.

It is extremely challenging to create an authoring tool that addresses all of these priorities. In the referenced works, authors needed to complete steps such as content generation, calibration, and registration in the AR environment for the AR guided assembly instructions to be correctly generated.

However, another approach is now possible. In complex assembly tasks, novices are trained by experts typically through demonstration of individual assembly steps in a live setting or recorded for later viewing. Using computer vision, and other graphical, techniques these expert demonstrations can be captured and used to generate AR guided assembly content with little to no AR expertise required of a user. The method will take the demonstration and handle content placement, registration, etc. in as automated a manner as possible. In addition, the method needs to provide visual feedback via an AR previewer. The previewer would allow the author to review the assembly steps and make refinements, if needed. Such a method would successfully address all the aforementioned priorities.

2.7. Research Issues

Based on the literature review and analysis, the following research issues have been identified:

1. **Establish automatic registration of parts in an AR environment during an expert assembly demonstration.**

Registration of virtual 3D models to the actual assembly is a difficult task requiring expertise in a number of areas such as computer vision, computer graphics, and optics. Previous solutions with artificial markers were generally not preferred in the manufacturing industry as they interfered with parts or workers. There is a need to perform automatic registration of useful assembly information to reduce the cognitive load on the author and not interfere in the capture environment. This requires careful calibration, as well as, finding out the location of the various parts post demonstration.

2. **Identify, track and find the orientation of the part or tool within the AR environment.**

To successfully create an intuitive authoring tool, domain expertise must be integrated accurately. In the case of a bench top assembly, the expert demonstrator must be able to use the tools to which he/she is accustomed. To allow for such an eventuality, techniques need to be developed that differentiate between an assembly part, a tool used for assembly, and hands. This differentiation not only lies in the visual differences but also in their functional differences. For example, a screw is an assembly part that is fixed while the screwdriver is a tool used to fix the screw. To be able to track any part, it needs to be identified from a part library and its location needs to be found either on its own or via cues in the scene (relative location with hands).

3. Design the interface of the AR authoring tool.

The interface of the AR authoring tool is a key component to the feasibility of the tool. It is important to understand the factors that make up a good AR authoring tool. A proper

framework needs to be established that assesses the literature available on current AR authoring tools and guides future interface design principles. No such focus has been placed by researchers on the topic and requires proper study.

CHAPTER 3 DECOMPOSING AN AR AUTHORING PROCESS

3.1. AR Authoring Tools and Roles Within the Authoring Process

Morten [135] defines authoring tools as content creation tools specifically catered towards educational content. This dissertation argues that all AR viewers intend to educate the end-user. Whether it be AR guided assembly viewers that teach a user an assembly task or collaborative environments informing users or each other's intent, this definition holds true. To further refine the definition of AR authoring tools, the term "Mental Models" is used. This term is defined as an internal scale-model representation of an external model. A mental model must be "runnable" i.e., be able to provide conceptual feedback on the results [136]. Consider the task of hammering a screw into a wall. With the mind's eye, visualize using a hammer as opposed to a screwdriver to perform the task. It would be quite obvious as to which tool would be easier and more efficient to use. In other words, a person should be able to conceptually run tests on a mental model (hammering a screw) with differing parameters (hammer versus screwdriver) and analyze the results. This is the essence of a runnable mental model. Thus, an AR authoring tool is defined as a software application allowing authors to create content with the purpose of recreating the author's runnable mental model for an end-user using augmented reality as the intermediary.

Consider an industrial engineer (author) who wants to create an AR guided assembly viewer that conveys assembly instructions of a particular product to a factory worker (end-user). This viewer should provide step-by-step instructions in the form of graphical overlays. The industrial engineer has the domain expertise to assemble the appropriate parts and relays that information to the factory worker. He or she uses an AR guided assembly authoring tool

that provides a set of functions to load 3D models, registers the models at their correct locations, creates a sequence of steps, and prepares the overall set of assembly instructions. The person creating the authoring tool, the developer is responsible for making sure that the industrial engineer is able to use the tool effectively.

The author and the end-user operate using different mental models. Using our previous example, the goal of the industrial engineer is to author virtual content for an AR guided assembly viewer and effectively pass on the assembly instructions to the factory worker learning the assembly operations. The factory worker's goal is to assemble the product as efficiently as possible. Regardless of the final interface design for the end-user, the focus of this dissertation lies only in the challenges that need to be addressed when authoring AR guided assembly instructions.

3.2. Hampshire Classification

Virtual content used by AR viewers, in general, can be authored in several ways. A generalized method of classification for AR authoring tools is provided by Hampshire et. al. [137]. This classification method will be referred to as the Hampshire classification.

Figure 13 shows this classification as a graph with two attributes plotted. On the x-axis is concept abstraction defined as ease of understanding of what the software does. On the y-axis is application interface abstraction defined as ease of understanding of the interface. As both the concept abstraction and application interface abstraction increases, the authoring tool is able to progressively consolidate AR concepts. This consolidation of AR concepts enables the authoring tool to hide AR concepts behind more understandable approaches and terminology.

The programming framework (lower, gray translucent box shown in the figure below) is the layer in the graph where the application interface abstraction is low. Authors within this framework need to understand AR concepts and also be skilled in programming to fully utilize the functions available. Hampshire describes a low level programming framework (level one) to be those programming libraries limited to the core duties involved in AR such as computer vision used for registration and computer graphics for rendering. Subsequently, high level programming frameworks (level two) provide the same functionality as level one but through a common meta-architecture. For example, an author using OpenCV [138], Open Scene Graph [139] and the AR Toolkit to create content for an AR viewer would be considered to be using a level one authoring tool. The author using a Python based software library which encompasses all of the above library functions, and abstracts them from the developer, would be considered a level two authoring tool.

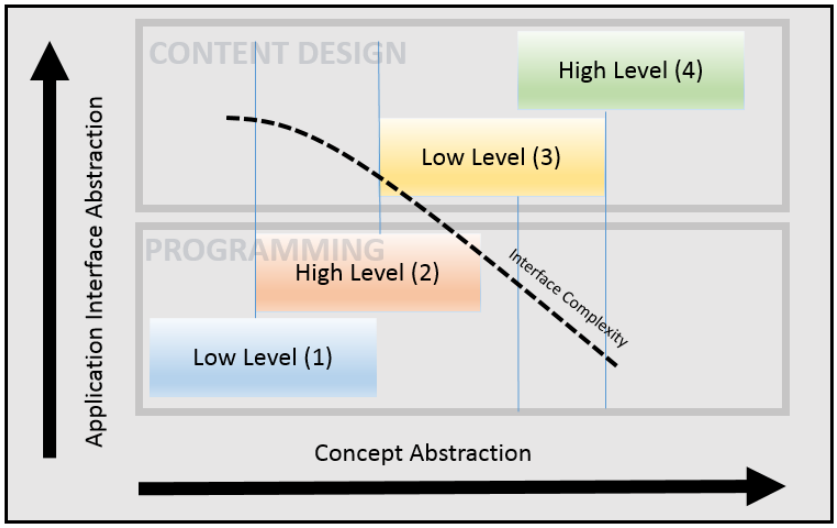


Figure 13. AR authoring tool classification level one (Low Level 1) to level four (High Level 4) (adapted from [137])

The content design framework (upper, gray translucent box in the figure above) is the layer in the graph where application interface abstraction is high. Authors within this framework typically use GUIs which translate complicated AR concepts into understandable button interactions. According to Hampshire the low level content design frameworks (level three) are content driven. In other words, the author need not be aware of the underlying AR concepts involved. The content is “visually programmable” by a graphical user interface. This provides easy abstraction between the tasks to be performed by the author and the content he/she chooses to incorporate in the AR viewer. It allows the author to easily translate his/her mental model into graphical content. Level four, or high level content design frameworks, are user-centric. The authoring tool is built on the domain in which the author is most familiar. Here the interaction between the author and the authoring tool is governed by the domain knowledge of the author.

Based on Hampshire’s research [137], Table 2 provides a summary of the different levels and expected domain knowledge of an author. Consider, the industrial engineer preparing the assembly instructions for the factory worker as discussed earlier. He/she would prefer an authoring interface that “speaks” his/her language. For example, if the industrial engineer wishes to place a virtual arrow in the real environment, it would require the author to understand the spatial orientation of the camera with respect to the real environment. Furthermore, the industrial engineer needs to understand that the arrow has to be placed with respect to a frame of reference and he/she may have to re-orient his/her view to properly place the arrow. For people with experience in 3D graphics or modeling packages this is a trivial task, but for someone without this expertise it is very difficult. A virtual arrow can represent pointing

behaviors, for showing direction while animations of the arrow can show expected path movement of the parts to be assembled. It presents multiple interpretations that the author needs to understand and decide how best to represent their mental model.

Table 2. Understanding the relationship between the expected author domain knowledge versus the complexity of the authoring tool [137]

Hampshire Classification	**Expected Author Domain Knowledge**
1	High Level Language such as C/ C++; Concepts in computer vision + computer graphics
2	Scripting Languages such as Python; Concepts in computer vision + computer graphics
3	GUI metaphors and interaction techniques; Concepts in Graphics
4	General Interface; Specific domain knowledge

Thus, placing a single arrow requires sizable considerations by the developer on its design and manipulation behavior, so that an author can understand the various ways to use it.. Authoring tools need to strike a balance between usability and functionality. An authoring tool shouldn't sacrifice important functionality to maintain ease of use. Conversely, it should not be overly complicated leading to many features going unused. These issues are all a reflection of the challenges involved in developing a level four authoring tool.

Creating an intuitive authoring tool would require it to intuitively call upon the experience of the industrial engineer. Imagine now, that the developer interviews some industrial engineers to understand their domain, terminologies and conventions. Using this information the developer makes an authoring tool that encompasses all of the terminologies. There are a number of problems which can occur –

- The industrial engineers interviewed are based in the aerospace industry whose use of terminology is different from someone say in the agricultural industry.
- Incorrect assumptions (based on the interviews) made by the developer misdirect the author during the authoring process.
- Underutilization of authoring tool features due to inadequate understanding of AR concepts by the author.

These problems are well represented in a ten year review [140] of the Designer's Augmented Reality Toolkit (DART) [141], an authoring tool (built on top of Adobe's Multimedia Director)with an AR previewer. In it, researchers reviewed the experiences authors had using DART. The researchers interviewed a broad group of authors that had used DART before. They varied in roles from researchers to artists to designers. The common themes within these interviews and their explanations are published. These researchers are the original developers of DART and as such no misinterpretations are expected. Before discussing the review, Figure 14 presents an example of the DART interface being used to author content for an AR application. Using DART, virtual content was loaded, edited, and registered with AR Toolkit markers placed around a room. The goal of this application was to create a scene where

prerecorded videos of jurors were talking among themselves around a real table in an AR environment. The interface shows four docked windows and a menu bar. The top left docked window is the AR previewer used to examine the authored content. The docked window on the bottom registers markers in the scene with particular virtual content.

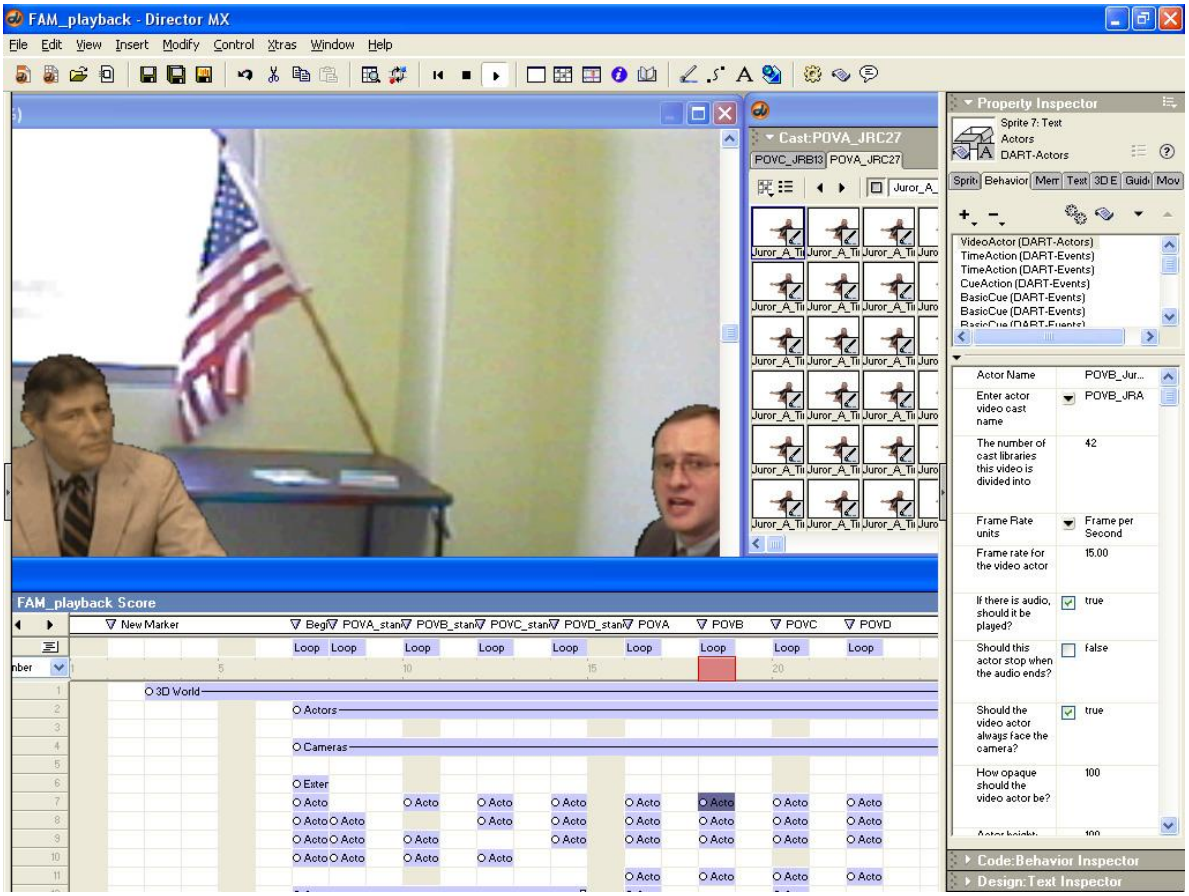


Figure 14. DART Interface [141] (used with permission from Blair MacIntyre)

The top right small docked window showcases a list of content available or used by the author in the AR environment. The right most docked window displays various properties of selected content that can be adjusted by an author.

Jumping back to the review of DART. DART was specifically created for designers (people experienced in Adobe's Multimedia Director), however, a number of problems were highlighted which illustrate the issues when authoring AR content. Authors unfamiliar with the domain of AR struggled to use DART and the researchers reviewing DART contributed it to incomplete documentation of the authoring tool. Helping a user correct errors was another challenge for DART. Authors were unable to narrow down the cause of defective behavior in the AR previewer (e.g. bad registration or jittery rendering). For example, an author was puzzled with the tracker errors the viewer was receiving until an expert realized the cause of the problem was malfunctioning tracking hardware and not from the authoring tool. There wasn't sufficient feedback to allow an author to diagnose the problem and call an expert if necessary. Thus, AR expertise was still required to properly use DART and new designers would have a difficult time using it. Many authors did not realize/understand the concept of shifting from a time driven authoring tool (based on fixed time-lines) to an event driven (based on user interaction) authoring tool. The similarity of the interface as shown in the figure to other time driven authoring tools led to this confusion. Though only one use case is referenced, it represents the complexity and challenges involved in creating a level four authoring tool.

3.3. Functional Categorization of AR Authoring Tools

Based on the literature review, the different functions necessary for an effective AR authoring tool have been defined and categorized. This categorization is presented in Figure 15. This provides a more organized manner to establish the critical functionality needed for AR authoring, but also the priority of each and its potential relationship to other functionalities.

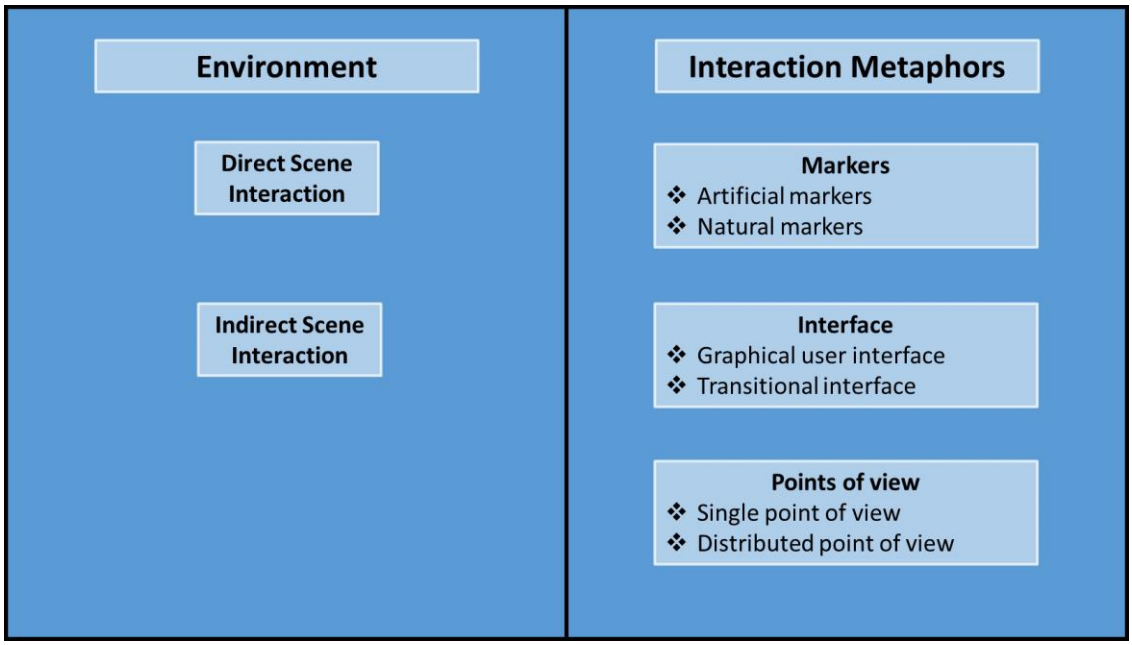


Figure 15. Functional Categorization of AR Authoring Tools

3.3.1. Environment

The environment category can be broken into two sub-categories: indirect and direct scene interaction.

Indirect scene interaction

The author is able to create content in the AR environment indirectly through a layer of abstraction typically in some form of GUI. Development of the AR viewer in this manner takes considerably longer because the author needs to design the content for the AR viewer (create 3D models or 2D textual instructions) and register it via the authoring tool. During the authoring process the author is unable to see how the content would appear in the AR environment. If an AR viewer is available for testing, the authoring process would further involve exporting the content into a format acceptable by the AR viewer, then importing it into

that viewer for testing. Thus, fixing problems, refining content location, etc. will require considerable time and effort from the author.

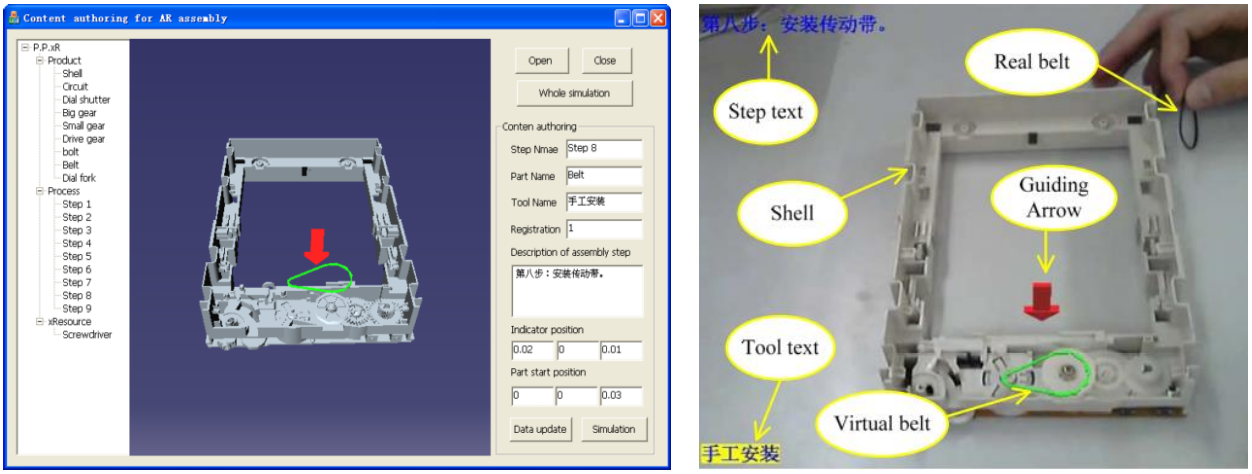


Figure 16. Authoring tool [126] presents a virtual 3D representation of the actual authoring scenario (left). AR viewer uses this information to overlay augmenting content (right) (used with permission from Junfeng Wang).

Direct scene interaction

In this sub-category, the author is able to view the augmented scene in the context of the end-user with an AR previewer. He/she can directly populate the physical scene with virtual content and also manipulate all objects in this scene. Thus, the author experiences the scene in a more similar manner as the end-user [142]–[144] and potentially mitigate problems as instructions are authored. Creating such an authoring tool is difficult as it requires registration of objects to the scene and provide interaction metaphors as well as real-time feedback to the author.

3.3.2. Interaction metaphors

Interaction metaphors are pre-defined design choices that facilitate the author being able to create registered augmentable content in an AR viewer. For example, in indirect scene interaction environments the translation of a 3D model could be based on typical mouse click and drag motion (as seen in 3D modeling software). The mouse click and drag motion is an interaction metaphor which facilitates translation of a 3D model. In the case of direct scene interaction environments, the same translation could be accomplished by the gesture of a finger swipe. Thus, the gesture is the interaction metaphor. Both metaphors are very different from each other but in the right context provide the same function. Interaction metaphors allow communication between the author and the authoring tool and typically mimic a real-world metaphor in order to promote intuitive interaction.

Based on the type of environment the developer has chosen he/she needs to understand the various interaction metaphors available. Choosing a badly represented interaction metaphor will cause poorly created authoring instructions and incorrect interpretation when viewed. For example, in a direct scene interaction environment, if the metaphors used are based on finger gestures then the right gestures needs to be chosen. Using a circular gesture for translation and straight line gesture for zooming into the scene is less intuitive and will cause frustration among users of the authoring tool.

Markers

In the case of a direct scene interaction environment, where visual feedback of the authored AR content is available, it is crucial that the right type of markers be used. They are used widely to provide the following properties:

- Detection - The authoring tool detects these markers and associates them to a particular augmenting element.
- Registration – The markers are used as a reference to provide the appropriate orientation of the augmenting elements.

Markers are divided into two sub-categories: natural and artificial. These markers were compared and contrasted by Zhang [145] based on the following questions:

- How quickly was the marker registration done?
- How accurate was the position of the marker?
- How easily could a marker be distinguished from other markers under varying ambient conditions?

Natural Markers

Natural markers are all geometrical, chromatic, or photometric features that are part of the physical environment including the user. For example, a user's hand may be modeled as a "device" whose input is understood by the computer using different natural features such as skin color [146,147] or shape [148-152], both of which allow for capturing hand pose. Subsequent tracking and recognition methods can facilitate hand gesture recognition [112] [153-155] for interaction. However, slight variability in gesture movement from one human to another can cause issues with accuracy. For example, in the case of HandyAR [156] or Wang's 3D hand interaction [113], the fingertips of the hand are used for augmenting the scene.

However, this requires a user to keep their fingers spread for long periods of time, which was uncomfortable and hindered use.

Natural features have been realized in multiple authoring tools for registering augmenting elements to predefined natural images. Drummond [157] used edges detected in outdoor environments compared to a 3D model of the same environment to register the orientation and location of the camera. Inaccuracies in the 3D model (missing edges) led to inaccuracies in the registration results. Pressigout [103] used edges also for registration in an indoor environment. The errors were very small comparatively (< 5mm) and the frame rate was real-time (50Hz) suggesting a low load on the processor. However, it required manual initialization of the correspondence between the object and edge model of the object. Gene [104] used point features selected by Shi-Tomasi , Klein [158] used FAST-10 corner detectors, Lepetit [159] used eigenimages, Ferrari [160] used image patches and Lee [161] used Scale Invariant Feature Transform (SIFT) features to register the orientation of the camera with respect to the real world. All of these techniques are affected by parameters such as lighting, camera resolution, and contrast. Care has to be taken to make sure these, and other real environment parameters, are consistent. Lastly, since natural markers rely on computer vision techniques to identify the markers it adds significant processing load on the viewer, which could lead to poor performance such as a low frame rate.

Artificial Markers

Artificial Markers (also known as fiducial markers) are man-made features incorporating a template that can be recognized by a computer. Due to their ease of use and registration they

have been used to realize several interaction metaphors including: pointers [20,21], multi-functional props [9,162,163], and location registration [155]. DWARF (Distributed Wearable Augmented Reality Framework) [164] is one such authoring tool where AR Toolkit [37] markers are heavily used. Predefined tools such as pens [100], personal Interaction panels [165], data gloves [166] or RFID tags [167] are also other examples of artificial markers.

Artificial markers are better than natural markers because they are designed to excel in all evaluative indicators. For example, as discussed earlier, AR Toolkit markers are bi-tonal (black and white) and square shaped to facilitate ease of detection and registration under poor conditions. Natural markers, on the other hand, need more precise ambient conditions to be reliably recognized by an AR author or viewer application. The biggest problem with artificial markers is that they need to be attached to a surface of the tracked object. Tying into our example of the industrial engineer authoring AR guided assembly content for a factory worker, the industrial engineer would need to place artificial markers on all parts in their correct location before the assembly. Placing them would be a tedious and time-consuming task, not to mention possibly hindering the assembly process. Overall, an AR authoring and viewing application that does not require artificial markers, while still keeping the frame rate of the visualization at an acceptable level, is desired.

Interfaces

An interface facilitates the communication between the author and the authoring tool. It provides methods to register virtual content available in the authoring tool with the AR environment. In the context of this research, interfaces are divided into two sub-categories: 1) graphical user interfaces and 2) transitional interfaces.

Graphical User Interface

Using a graphical user interface (GUI) is a visual way to interact with a computer. It uses icons, photos, text, and other graphical elements to accept user input or to provide data to a user. For three decades the world has been using GUIs and at this juncture it is safe to assume that people are familiar with common GUI concepts such as icon clicks, and drag and drop features. AR authoring tools with GUIs utilize an abstraction of the physical world to facilitate content creation and scene design in one of the following ways:

- 3D Model: In this case the environment is represented by 3D models and the augmenting elements are directly registered on the model [126].
- 2D Image: A snapshot of the environment is taken and all augmented objects are registered to that image [168,169]
- Location based: GPS coordinates are used to directly register the location with certain augmented objects [170].

AR authoring tools incorporating multiple of these features are also available. For example, the application introduced by Lee et. al. [171] provides both GPS and 2D Images for authoring.

Transitional Interfaces

Transitional interfaces use abstract concepts conceived by way of non-traditional computer inputs (i.e. not using keyboard and mouse input) to create content for an AR viewer. Concepts such as storyboards [172] and sketches [173] have been used to author AR work

instructions. In both of these projects, a specialized AR viewer was developed that interpreted the unique inputs. The viewer was then able to transform these into virtual augmented content. These type of interfaces are not used generally as they are subject to an author's ability to draw sketches of pre-defined templates. In addition, sketches and storyboards had to be considerably different from one another so that the viewer was able to distinguish between them. Lastly, these sketches and storyboards could not overlap each other.

Points of view

A developer may choose to provide multiple methods by which an author can preview content. This may be done to facilitate collaboration or user-feedback. Within the realm of interaction metaphors, "points of view" is defined as the number of authors that each have a separate hardware link to the AR previewer.

If an authoring tool allows for one author to use a Head Mounted Display (HMD) and another author to use a desktop application with a fixed camera to view the same AR environment, then it is considered to have distributed points of view. DWARF [164] is an example of an authoring tool that uses a distributed point of view. In its previewer a central AR environment (on a table) is viewed by a laptop, mobile device, and head mounted display.

Conversely, one or more authors viewing the same AR environment with the same interface is considered to have a single point of view. One example is when authors together view the environment through a single large screen [174] or individually through the same type of HMD [162]. The developer has to choose whether authors are allowed to use single or distributed points of view. Allowing for a distributed view means the developer provides an

author multiple experiences via different hardware interfaces. This leads to additional system complexity.

3.4. Author

The domain experience gathered by an author prior to using an authoring tool is their knowledge base. Establishing a knowledge base and creating a structured set of rules to follow for a particular domain is a specialized branch of software engineering and is not expected of the developer. It is up to the developer to find out the author's knowledge base by either assuming certain criteria, asking the authors questions or by using knowledge engineers who are experienced in creating rules from experts' ill-structured domain knowledge. The goal is to keep the author's domain knowledge in mind when deciding the environment and interaction metaphors. Once a knowledge base has been established for an author, the developer can begin to call upon this base to form intuitive links between the author and the authoring tool.

Early AR authoring tools like PowerSpace [175] used Powerpoint as the base interface. The author's domain knowledge was technical documentation editing and at the time the primary tool used was PowerPoint. It allowed a transition from "traditional" work to AR-based guides. Another example, AR Paint [174] was an authoring tool based on the Scratch programming language [176]. Scratch is a visual programming language intended for children developed at the Massachusetts Institute of Technology. The authoring tool allowed children to author AR content and preview it at the same time (direct scene interaction environment) using simplified logic available in Scratch.

3.5. AR Authoring Via Expert Demonstration

In this chapter a number of components have been introduced. The Hampshire classification seeks to address the capabilities of an AR authoring tool. A level four authoring tool is –

1. Domain specific

The interaction between the author and the authoring tool is governed by the domain knowledge of the author. Complex AR concepts are translated into language understandable by the author.

2. Content design driven

The aim of the author is to design the content for the AR system using the AR authoring tool. Design of content involves making decisions within the capabilities of the AR authoring tool. He/she wishes to translate his/her mental model into an AR based representation that can be mimicked by the end user of the AR system.

Based on the expectations of a level four authoring tool, this dissertation proposes a novel authoring process. The authoring process is divided into two steps, namely the expert demonstration step and the refinement step.

The novel idea behind this authoring process is that physical expert demonstrations are recorded and analyzed to provide AR based graphical components. Consider the example of the industrial engineer wanting to pass on the assembly information of a product to a factory worker using AR. It is safe to say that the industrial engineer will not have expertise in the realm of AR. The intention of this authoring process is for the industrial engineer to be able to

demonstrate the assembly of the product and automatically generate the instruction in an AR compatible format. The mental model of the industrial engineer's assembly is now a set of AR based instructions and forces him/her to a specific domain, in this case the domain is that of expert assembly. The expert demonstration step is a direct environment and natural marker based authoring step that involves physically demonstrating the assembly. In the next section, the dissertation provides a published version of a proof of concept idea of a new authoring process.

The refinement step is a GUI based indirect environment authoring step that seeks to refine the demonstration step. This refinement is done by enabling the author to set certain features that cannot be easily set in the demonstration step. For example, changing the colors of the virtual objects, adding textual instructions, and changing animation paths.

Due to these two separate steps the authoring process is considered to be a hybrid approach. It consists of, both direct and indirect environments, both GUI based and transitional based interfaces, and use of natural markers (as much as possible) and artificial markers. In conclusion, this authoring process seeks to remove challenges associated with having to understand complex AR concepts and still provide basic functionality to modify instructions. This chapter seeks to partly address the third research issue by providing a clear set of design principles to be used by developers when making an AR authoring tool. The authoring process is a direct outcome of that and the actual user interface is yet to be fully designed.

CHAPTER 4 USING A PARTICLE SWARM OPTIMIZED GAUSSIAN MIXTURE MODEL FOR SKIN DETECTION DURING AUTOMATED AR AUTHORING

B. Bhattacharya, E. Winer, “Using a Particle Swarm Optimized Gaussian Mixture Model for Skin Detection During Automated AR Authoring”, Computers in Industry, In Review (2016)

The expert demonstration authoring technique at this stage requires the use of hands during the manual assembly sequence. Thus, a critical component required prior to the processing of the part is the filtering or removal of the skin pixels from the scene. This paper submitted to Computers in Industry describes the PSOGMM algorithm, how it overcomes traditional GMM, and its necessity in the expert demonstration authoring technique.

Abstract

Augmented Reality (AR) for guided assembly has shown great promise in the past decade displaying increased efficiency, better performance and reduced errors among assembly operators in product manufacturing environments. However, the challenge for AR guided assembly lies mainly in the authoring of content as it requires people uninitiated in AR to understand complex concepts in computer vision and computer graphics. To mitigate this challenge, expert demonstration, a novel authoring approach, is introduced in this paper. It enables the author to demonstrate an assembly sequence that is recorded and processed to generate AR work instructions automatically. This assembly sequence is recorded via an RGB-D sensor (in this case a Microsoft Kinect) and processed to provide automatic AR work instructions. For this research, expert demonstration is required to be a manual bench top

assembly process. Due to this requirement, tracking the hands via skin detection is a critical component of expert demonstration based authoring. To create a skin detection model, training data was generated by providing prerecorded images of the expert's hand. The background from these images was filtered out and the expert's skin pixels remained. However, the skin data contained noise (i.e. was not clean) and there were pixels that were part of the background. Popular skin detection techniques such as the Gaussian Mixture Model (GMM) rely on clean training data. To address this, a new algorithm, termed the Particle Swarm Optimized Gaussian Mixture Model (PSOGMM) was developed. PSOGMM takes in a single input variable representing the noise threshold and generates a skin model. Unlike GMM, PSOGMM does not require the number of Gaussians as input to the algorithm, a significant achievement. PSOGMM was compared to GMM on two independent datasets. It was seen that PSOGMM performed 98% faster for training and 36% faster for testing on the first dataset and 98% faster for training and 43% faster for testing on the second dataset. Based on the skin model generated from the training data, the number of steps in the assembly can be automatically identified through a user input threshold. Results for three test cases for expert demonstration are also presented.

4.1. Introduction

An augmented reality (AR) system is defined as "a system that supplements the real world with virtual (computer-generated) objects that appear to coexist in the same space as the real world. An AR system has the following characteristics: it combines real and virtual objects in a real environment, runs interactively, and in real time, and registers (aligns) real and virtual objects with each other in 3D." [8]

AR users view the real world via a screen and graphical objects (e.g., 2D text and animated 3D models) are overlaid by the AR system. Robust sensors provide information that register (align) the graphical objects with the 2D camera images of the real world. Examples of sensors commonly used are camera images [9], RFID tags [10], GPS [11] and infrared markers [12]. Figure 17 represents a snapshot of the popular commercial smartphone based AR game known as Pokémon Go. This mobile AR system combines a real-world image feed with 2D text and graphics, updates in real-time and registers the real world and the various virtual objects based on the user's location and bearing.

Figure 17. Pokémon Go

Location and bearing information is based on GPS and compass sensors that have been well researched [4,6,7,8]. However, these are large scale applications that allow registration errors of several meters. In small scale applications like assembly, GPS and compass sensors are not accurate enough, thus requiring the use of near range sensors like cameras, RFID, infrared markers.

AR technology used in the assembly context is known as AR guided assembly. Near range sensors are used to provide accurate AR work instructions at each assembly step. For

example, Zhang et. al. [99] describes an RFID-assisted assembly guidance system that enables a user to provide AR work instructions for assembling a mouse. Radkowski et. al. [26] developed and evaluated an AR system that enables a user to assemble a pump.

Figure 18. Augmented reality system to aid user in assembling a pump. Textual instructions are provided on top while a virtual demonstration is presented [10]

A key component to an AR system is the registration of the virtual content with the real-world (i.e. showing a virtual part in the correct location on the actual physical assembly). This can be done in several ways using tracking systems or physical markers. Markers are physical objects, typically pictures or symbols on flat surfaces, present in the environment identifiable by sensors. The resulting sensor information is processed to enable the registration of virtual objects. Figure 18 illustrates one step of the assembly of a pump. There are various markers present in the environment (black and white square images) that are used by the AR system to register the appropriate 3D models and their animations. The works of Zhang and Radkowski clearly show the use and potential of AR technology in assembly. All the required information is available to the user and ideally no reference to manuals have to take place. However, in the above examples, the researchers do not discuss the time and effort required to create (i.e.

"author") the virtual content. For example, consider the assembly step shown in Figure 18. The person authoring the content needs to model the cylinder block in a CAD package such as SolidWorks [180] and export that model to a format readable by an AR authoring system and viewer. The appropriate 3D transformations then need to be applied with reference to the correct marker in the scene. If that is not within the capabilities of the AR system, then a third-party software tool needs to be used that is capable and requires further conversion between formats. Finally, there would require iteration between applying the 3D transformations and verifying them in an AR viewer to make sure the model appears in the correct location in the real environment. Part, or all, of this process is then repeated for every piece of virtual content in the scene.

"Expert demonstration" seeks to allow the author to demonstrate the assembly while recording it using a computer vision system. The recording is then processed using computer vision techniques. Due to the manual nature of the demonstration, detecting the hands is a critical component for expert demonstration authoring to work. Thus, skin detection is an important aspect of expert demonstration and requires analysis of the various algorithms available. The focus of this paper is in describing the deficiencies of popular skin detection algorithms, specifically, Gaussian mixture models (GMM). With the development of a new algorithm called Particle Swarm Optimized Gaussian Mixture Model (PSOGMM), it seeks to address the challenges associated with GMMs and provide a good model for skin detection.

4.2. Background

4.2.1. Skin detection

Skin detection is used to detect and process the hand pixels and is an important aspect of computer vision research. Through skin detection, hand and body movements can be determined to address challenges in a variety of areas. Hurst and Dekker [181] used skin detection to enable object creation in a mobile AR application. Ong et. al. [113] used skin detection to find particular hand interactions to create assembly simulations in AR. Skin detection was also used as a preprocessing step to detect hand pixels and identify hand gestures [182]. Platzer et. al. [183] developed an algorithm using skin detection to gauge the probability of an explicit image.

There has been plenty of research conducted in skin detection and it is an important part of computer vision research [146]. Kakumanu et al.[146], in their survey, illustrates that researchers have used a variety of color spaces and modeling techniques to achieve robust and accurate skin detection.

Popular methods of skin detection

A key concept to successful skin detection is determining the probability of a pixel being skin colored. The probabilities for a color are typically found from training data and further thresholding results in skin classification. Argyros et. al. [147] used this method in the YUV color space but with hysteresis thresholding to classify the skin. Baltzakis [184] used it to track hands, faces, and facial features of multiple people. A major disadvantage with this type of method is that finding the correct threshold is a matter of trial and error and will most likely change with different training data. Other skin detection methods include hue thresholding [155], [185],

statistical color models [186], Dempster–Shafer theory based models [187], and background subtraction based models [188]. It is important to note that the methods discussed previously are pixel wise classification models, that is, models based on one pixel's available information. Other methods implemented by Ruiz-Del-Solar et. al. [189] and Mathias et. al. [152] used neighborhood or local pixel information to improve their models. However, the time for training and testing were both greatly increased. Although these methods exist they each have their limitations and do not work well with noisy, constantly changing training data. Another popular method used for skin detection, Gaussian Mixture Models, sought to address these challenges.

Gaussian mixture models

Gaussian mixture models (GMM) [190] were developed to model the distribution of data as a weighted sum of Gaussian densities. Jones et. al. [186] created a generalized detector for skin using over a billion pixels of training data with an accuracy of 80%. Handy AR [156] used this generalized detector with an adaptive component for lighting issues. Park et. al. [191] used it to detect the skin among other things and identify the different poses of people.

$$p(x \mid \lambda) = \sum_{i=1}^{K} w_i g(x \mid \mu_i, \Sigma_i) \quad (1)$$

Equation 1 defines the GMM as a weighted (w_i) sum of K Gaussian densities described by its parameters (mean (μ) and covariance (Σ)). Typically, this equation is fitted through the Expectation-Maximization method [192] using the weights and Gaussian variables as the input parameters to be fit.

Finding the optimal number of Gaussians for a particular dataset is a challenge and is generally done via K-means [146]. Thus, there is a need to know the number of Gaussians before the computation of the model can commence. This is a major drawback in GMMs as in most cases knowing the number of Gaussians is difficult. There are many methods introduced by researchers to find the optimum K. The Gap statistic [193], Information Theoretic approach [194] and image processing techniques [195] are just some of the methods researchers have used to determine K. These techniques are slow and work well on synthetic data [196], [197] that is devoid of noise. In real-life (or natural) datasets, there is noise that cannot be removed easily, and these methods struggle for accuracy.

To mitigate this challenge, researchers allowed K to be part of the input parameters to be fit. However, in most real-world cases the data was overfit. In other words, the noise became part of the model and led to poor performance in terms of accuracy.

Particle swarm optimization

Particle Swarm Optimization (PSO) is an optimization technique that owes its roots to the behaviors of bird flocking, fish schooling and swarming insects [198]. Consider, a flock of birds within a forest trying to find a tree with the most food to support the flock. They initially spread out randomly across the forest and begin to communicate with each other. As they explore, the birds eventually begin to move towards the tree with the most food. Similar behaviors are seen in schools of fish or swarms of insects.

PSO attempts to model these behaviors in the following way:

1. The flocks are represented mathematically by a swarm of particles each having a position and velocity.

2. The forest is called the design space defined by an objective function with limits in each dimension.

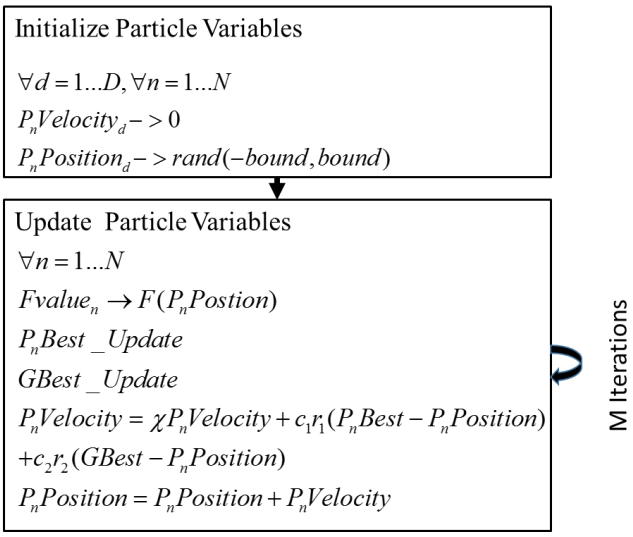


Figure 19. Inertial PSO algorithm

Figure 19 provides a more formal mathematical description of the Inertial PSO algorithm (a variant of the basic PSO algorithm). Initially, the particle positions are randomly found within certain bounds of the design space and the particle velocity is set to zero. The particles are then updated and this update step consists of the following:

1. Evaluate the objective function (Fvalue) for each particle.

2. Based on the objective function evaluations, update GBest (the global best objective value seen across the swarm) and PBest (the local best objective value seen by a particle in its history).

3. Update the particle velocity (PVelocity)

4. Update the particle position (PPosition)

The update step is run M times until some termination criteria is reached. This criteria can be a limit to M or a minimum change in the global best or a combination of the two. Once the algorithm is terminated the value of the global best is the result.

The advantages of PSO for solving optimization problems are:

1. The implementation has a small memory footprint.

2. It is simple to implement and requires no time-consuming equations.

3. The method does not require the computation of derivatives.

4. In a single threaded system, for every particle the most computational resources are spent in evaluating the objective function. Thus, in terms of time taken per particle, there is a single point of optimization.

5. Objective function evaluations for each particle can be parallelized on different computational threads or processors using SIMD architectures such as CUDA [36–42].

For a more detailed description of the PSO algorithm, and its applications, please refer to Eberhart et. al. [206].

4.2.2. Particle swarm optimized Gaussian mixture model

The intent of expert demonstration is to enable the author to create AR work instructions without having computer vision and 3D graphics expertise. So, a skin detection algorithm was required that was able to train and detect skin pixels quickly and provide an easy abstraction to the author. As shown in the background section, GMM, used in many popular algorithms, was either inaccurate or required intimate knowledge of computer vision. Abstracting the GMM parameters to make it easier for the author to understand presented significant challenges. For example, in depth analysis of the domain knowledge of the author and iterations of user studies would be required. Based on this, using GMM was ruled out.

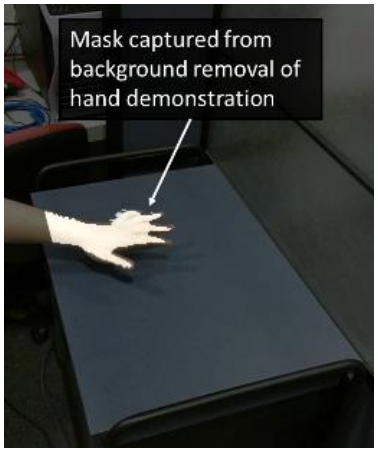


Figure 20. Skin pixels captured for training and modeled using PSOGMM

Figure 20 shows a frame of a hand demonstration captured in a controlled environment. The background pixels (i.e. the table) were removed and the skin pixels remained. However, edge cases around the hand corrupted the data and made it noisy. In the above figure, it is

clearly seen that pixels around the edge and in between the fingers are captured as part of the skin. Adding methods like erosion and dilation that filter out this noise would mean burdening the author with complicated computer vision concepts. This was against the intention of expert demonstration authoring. To address these issues, a new algorithm was developed called Particle Swarm Optimized Gaussian Mixture Model (PSOGMM). As the name implies, PSOGMM builds off of GMMs.

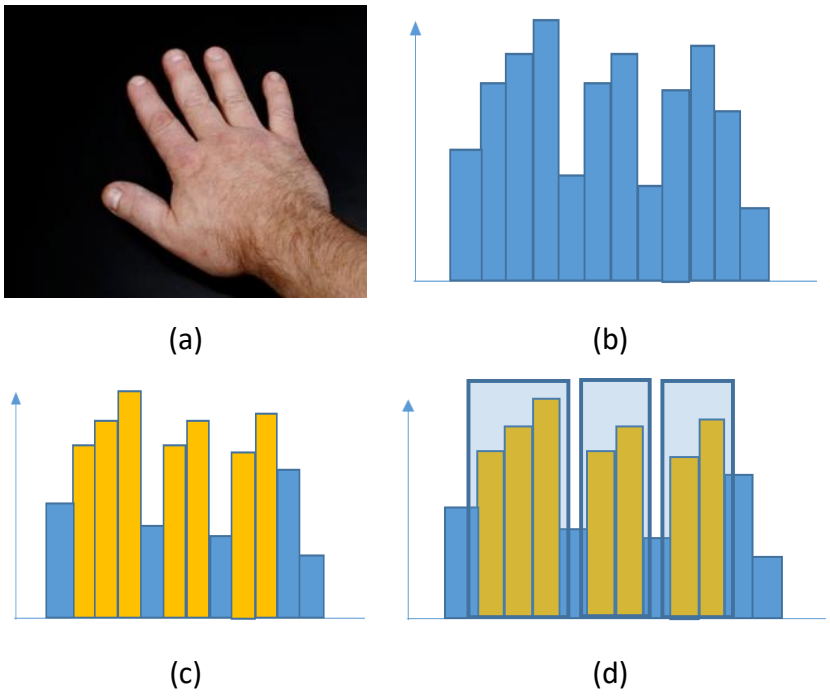


Figure 21. (a)-(d) Various steps involved in the improved PSOGMM algorithm; 21(a) Skin data is extracted from the hand demonstration; 21(b) Histogram of bin size one is created from the data; 21(c) The top contributors to the model are found and arranged according to their bin value; 21(d) The contributors are clustered and is used as the initial data for PSOGMM.

Once the skin data was extracted from the hand demonstration (as shown in Figure 21(a)), it was converted from the RGB color space to the YCbCr color space (using OpenCV [138]). The Y dimension represents the illuminance of the captured environment and can be

safely ignored from the skin model, hence, reducing the dimensionality of the model. This conversion was effectively used for skin detection by Kakumanu et. al. [146] and Argyros et. al. [147]. A histogram with bin size one was calculated from the skin data. Figure 21(b) illustrates an example of the histogram with the X axis being the color value (or bin value) and the Y axis describing the number of points within the dataset that were of that color (or bin amplitude).

Each bin's contribution to the training data was calculated by dividing the bin amplitude by the total number of training samples. The bin with the maximum contribution was added to a set G and this process iterated until the total contribution of the set was greater than a user set threshold (*Threshold*). This method is described in Table 3 (Step 1 – 6) and the result is illustrated in Figure 21(c). The yellow bars shown in Figure 21(c) are the highest contributors to the skin data distribution, now part of the set G, and the sum of their contributions is above *Threshold*.

The set G was clustered to extract the Gaussians that would eventually be used to initialize the PSOGMM. Table 3 (Step 7 – 12) describes how each individual Gaussian was extracted from the set G. Set G was first sorted by the bin value (i.e. the color value). Bins that were close to each other (within a user specified neighborhood value, in this case three) were grouped together as shown by the bounded bars in Figure 21(d). These groups were transformed into the Gaussians (i.e. mean and standard deviation parameters were calculated for each group) and used to initialize the PSOGMM. The algorithm described in Table 3 will be known as the initialization step. It is important to note that though only one dimension is described the same method was applied independently across all dimensions.

Table 3. Setting up the initial data for PSOGMM

Step	Description
1	Convert data *D* to histogram *H* with bin size 1
2	H' ← H / N (*N* – Number of data points)
3	G = Ø (G is a new set)
4	Add highest contributor (bin amplitude) from *H'* to G
5	Total Contribution = Σ contribution(g); g ε G
6	If Total Contribution < Threshold GOTO step 4
7	Sort *G* by bin value
8	I = 0, LB = G(I), UB = G(I), U = Ø
9	Loop I from 1 to length(G)
10	If (G(I) ≠ Ø && I – UB < 3) UB = I
11	Else Add new Gaussian (UB , LB) to set U
12	End Loop

$$\forall i \in [1, D],\ j \in [1, n(G_i)]$$
$$\mu(g_{ij}) - c_{ij} \bullet \sigma(g_{ij}) < p_i < \mu(g_{ij}) + c_{ij} \bullet \sigma(g_{ij}) \qquad (2)$$
$$\text{If satisfied } Match(p) = 1 \text{ else } Match(p) = 0$$

Equation 2 (also known as *Match(p)*) describes the skin model used to classify whether a pixel *p* is a skin pixel or not. *i* describes the ith dimension in the data with a total of *D* dimensions. Within the ith dimension, *j* describes the jth Gaussian available from the initialization step.

$$\min \left| \frac{\sum_{p \in P}^{n(P)} Match(p)}{n(P)} - Threshold \right| \tag{3}$$

Equation 3 describes the objective function for the PSO algorithm. *n(P)* is the total number of pixels available for training the model. *Threshold* is the user input value set to find out how much noise to remove. This is the same user set threshold as in the initialization step. The PSO algorithm optimizes c_{ij} until it converges. A termination criteria of a user defined error value (in this case 0.01 was used) was placed on the PSO algorithm. In other words, the PSO algorithm would terminate when the ratio of the skin pixels to the total number of pixels were within a certain error of the *Threshold*. For example, if the *Threshold* value was 0.9 then the PSO algorithm would continue to run until the PSOGMM model represents 89.99% to 90.01% of the skin pixels in the dataset.

The advantages of PSOGMM are as follows:

1. The number of Gaussians were automatically calculated unlike GMM.

2. There was only one user input variable that needs to be set. In this paper's evaluations, using a Threshold value between 0.85 - 0.95 worked well.

3. Due to the initialization step, the design space was well bounded and hence more likely to find the global minimum.

4. Overfitting of data was easily identifiable and to a certain extent rectifiable. For example, if a skin model detects noise pixels as skin, then it means that overfitting has occurred. In a GMM, typically the user would need to modify the number of Gaussians and the threshold requiring a more intimate understanding of the GMM algorithm. However, in a PSOGMM model, the user would only have to decrease the value of Threshold since it represents how much data is considered clean.

With the PSOGMM developed, it was used to detect and filter skin pixels in the expert demonstration authoring approach described in the next section. PSOGMM was also evaluated against traditional GMM on independent datasets as described in the Results section. It is important to note that the required hand demonstration allows the author to be more generic. Some manufacturing requirements may not allow the use of gloves, hence, the author has to use their bare hands. In other situations, gloves may be mandated. In either case, PSOGMM will capture the appropriate model and filter the skin pixels accordingly.

4.3. Expert Demonstration

4.3.1. Setup

To mimic a bench top assembly setup, a flat table with a back wall was used and depth + RGB camera (Microsoft Kinect) was used to capture the author's hand movement and part

assembly as shown in Figure 22. The 3D point cloud data generated by the Kinect was processed to get the individual assembly steps using the PSOGMM method of skin detection. It is important to note that the focus of this paper is in the critical step of skin detection. Further details regarding other processing steps involved with expert demonstration (e.g., finding the transformation of the various parts) is out of the scope of this paper.

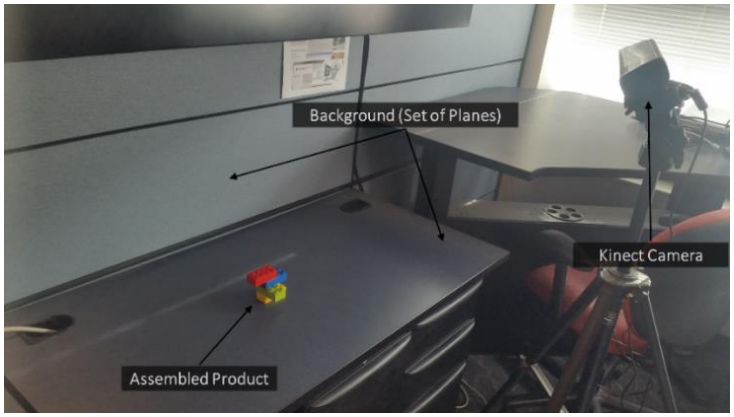


Figure 22. Capture Setup

Figure 23 displays the different 3D frames (Kinect sensor based RGB point clouds) of a typical assembly process using DUPLO blocks.

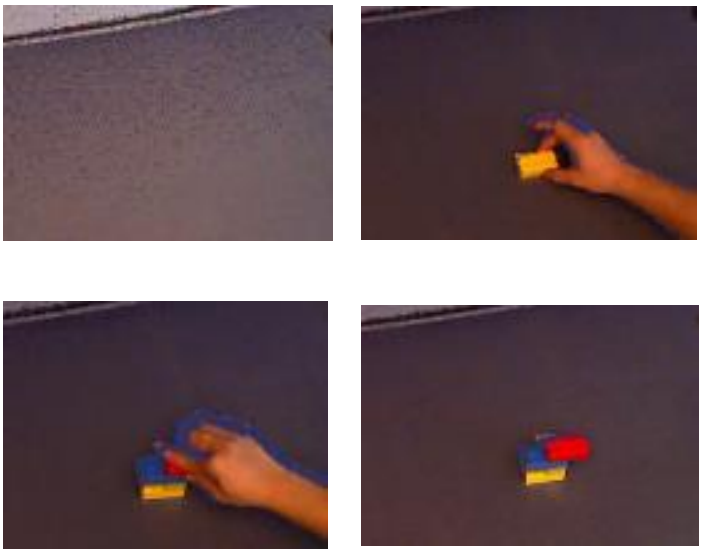


Figure 23. Demonstration of DUPLO blocks assembly (using Kinect to capture). Order is from top left to bottom right.

All the processing was done on a Dell Inspiron with an i7 processor at 2.10 GHz, 8GB RAM and NVIDIA GeForce GT 640M graphics processor.

4.3.2. Background processing

A set of orthogonal planes as shown in Figure 22 are assumed to be the background. They are identified and removed using PCL's (Point Cloud Library [207]) RANSAC plane removal technique. Noise from the Kinect sensor caused multiple points (x_0, y_0, z_0) to lie very close to, but not exactly on the planes (expressed as ax + by + cz + d = 0). These erroneous points are removed based on Equation 4.

$$\frac{ax_0 + by_0 + cz_0 + d}{\sqrt{a^2 + b^2 + c^2}} < BGThreshold \tag{4}$$

Equation 4 states that any point whose distance from the plane is less than *BGThreshold* is removed. Limited empirical testing showed that a distance of 2 mm was sufficient for this example. Further work is necessary to more accurately determine this distance.

4.3.3. Skin detection and finding the number of steps

Once the background was removed from the recorded demonstration, the data was available to create the skin model using the PSOGMM model. Based on the number of skin pixels, the number of assembly steps can be automatically found using the algorithm described in Table 4.

Table 4. Finding the number of steps

Step	Description
0	State = NoHand
1	For each frame F
2	Find the number of skin pixels (N_{SP}) from PSOGMM based skin model
3	If N_{SP} > Hand-NoHandThreshold place frame State = Hand
4	If N_{SP} < Hand-NoHandThreshold place State = NoHand
5	Increment Number_Of_Steps by 1 if state changes from NoHand to Hand
6	End Loop

Table 4 illustrates that each frame was evaluated to find skin pixels. The number of skin pixels (N_{SP}) was compared to a threshold (*Hand-NoHandThreshold*) to determine whether a hand was present in the frame or not. If the frame contained a hand it means that a part of the assembly was in the process of being placed. If the frame did not contain a hand, then it meant that the assembly step was concluded. The state of the frame was defined by whether a hand was present (Hand) or not (NoHand). If at a particular frame the state changed from NoHand to Hand, it indicated the start of a new assembly step. Consequently, if the state changed from Hand to NoHand, it indicated the end of an assembly step.

Accurately finding the number of steps was an important part of expert demonstration based AR authoring. It was essential for the reliability of the authoring approach and at this stage required the author to set the *Hand-NoHandThreshold*. Finding the number of steps allowed for the appropriate separation of the dynamic frames (that represent part movement) from the static frames (representing the final position of the part in the assembly step). If a hand had to enter the scene multiple times during a single step, a user can easily merge frames together if needed.

4.4. Results

4.4.1. Skin detection

In the first set of results this work compared the PSOGMM against traditional GMM. The independent variables used in both were color space (RGB, YCbCr, HSV, and HLS), tolerance (0.7 to 0.95 in steps of 0.05). For traditional GMM an additional number of Gaussians parameter (2 – 4) was necessary.

The dataset used was by Bhatt et. al. [208] that consisted of a combination of the FERET Database [209] and PAL Database. The dataset comprised of multiple people of varying ethnicities, poses, emotions and image backgrounds along with their corresponding masks as shown in Figure 24. Out of 1118 samples, 112 were used for training (every tenth image to make sure of the variability in the training set) and 1006 were used for testing.

Figure 24. Example of the dataset used; original image captured (left), manually labeled mask for corresponding image (right)

A full factorial of all variable combinations was tested resulting in 24 runs (4 color spaces x 6 tolerance levels) for PSOGMM and 72 runs (24 x 3 choices for number of Gaussians) for traditional GMMs.

To evaluate PSOGMM against traditional GMM two metrics were used, namely, precision and recall.

$$Precision = \frac{tp}{tp + fp} \tag{5}$$

$$Recall = \frac{tp}{tp + tn} \tag{6}$$

Equation 5 defines precision as the ratio between the true positive (*tp*) to the sum of the true positive and false positive(*fp*). In other words, precision measures the ratio of the number of actual skin values to the number of skin values identified by the skin model. Equation 6 defines recall as the ratio between the true positive to the sum of the true positive and true negative. In other words, recall measures the ratio of the number of actual skin values to number of skin values available in the testing dataset.

Table 5. PSOGMM vs traditional 2 GMM on Bhatt dataset

Step	**Traditional GMM**	**PSOGMM**	**Improvement**
Time for training	230.6 s	4.6 s	98%
Time for testing	456.9 s	290.7 s	36%
Precision	0.78	0.77	-1%
Recall	0.76	0.80	5%

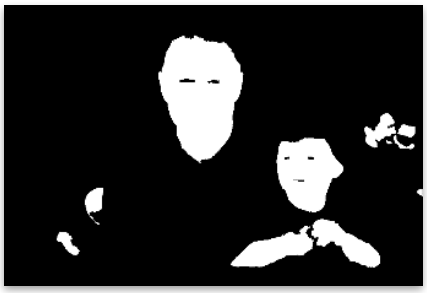
Table 5 presents the results of the best PSOGMM model compared to the best traditional 2 GMM model (2 represents the number of Gaussians). In terms of training and testing PSOGMM outperforms traditional GMM. PSOGMM has higher recall and less precision, which means that it can detect more skin pixels but also has a slightly higher rate of false positives.

Table 6. PSOGMM vs traditional 3 GMM on Ruiz-Del-Solar dataset

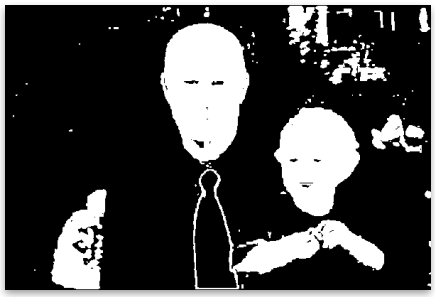
Step	**Traditional GMM**	**PSOGMM**	**Improvement**
Time for training	170.6 s	3.1 s	98%
Time for testing	180.9 s	102.4 s	43%
Precision	0.68	0.64	-1%
Recall	0.77	0.89	16%

Table 6 shows the results of PSOGMM vs traditional 3 GMM (3 represents the number of Gaussians) on a different database created by Ruiz-Del-Solar et. al. [189]. This dataset was primarily for multi-pixel wise classification (i.e. using neighborhood information). From the above table, it can be seen that PSOGMM consistently improves over traditional GMM with only a small loss in precision.

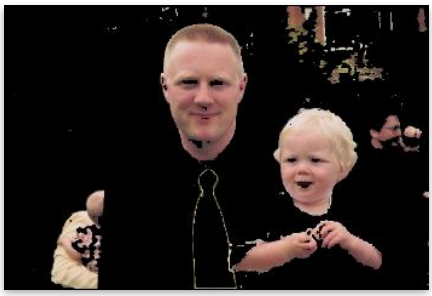
(a) Original image



(b) Manual mask



(c) PSO GMM based mask



(d) Final output

Figure 25. Example result of the PSO GMM based skin model

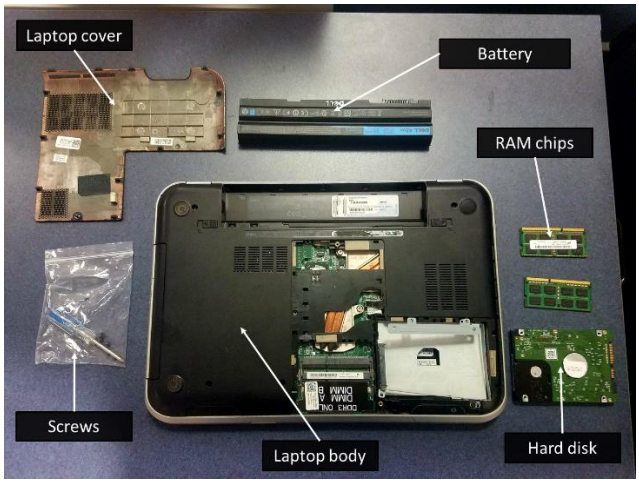
Figure 25 illustrates a result visually comparing PSOGMM with a mask created manually. PSOGMM can be seen to perform comparably in detecting skin. The differences seen would not cause problems in detecting hands in an assembly environment.

4.4.2. Expert demonstration dataset

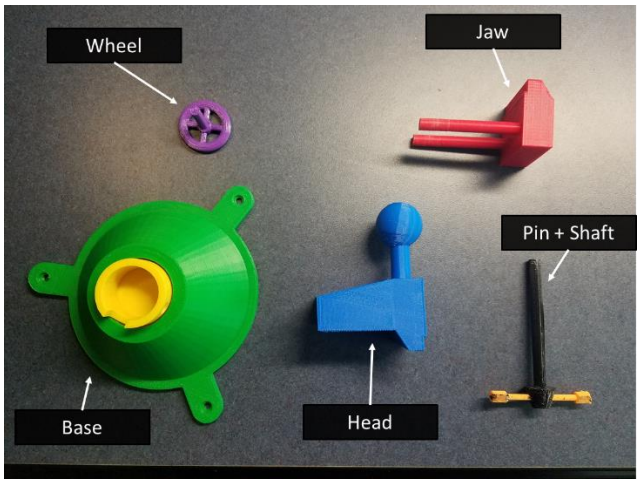
Three types of assembly sequences were tested for the expert demonstration authoring approach. They were:

1. DUPLO blocks
2. Vise grip
3. Laptop

These assemblies were chosen due to their increasing complexity from simple consistent DUPLO blocks to hard to identify and track flat parts of a laptop.



(a) Laptop assembly parts



(b) Vise grip assembly parts

Figure 26. Laptop (top) and vise assembly(bottom) parts

Figure 26 illustrates the various parts involved in the laptop and vise grip assembly sequences. The DUPLO blocks assembly has been discussed and will not be shown here again in the interest of space.

The DUPLO block assembly consists of four DUPLO blocks of different colors. This assembly was chosen because: 1) the DUPLO blocks are designed to lock into place with each other, 2) they are easy to identify, and 3) only one hand is needed for the assembly (less skin pixels to remove).

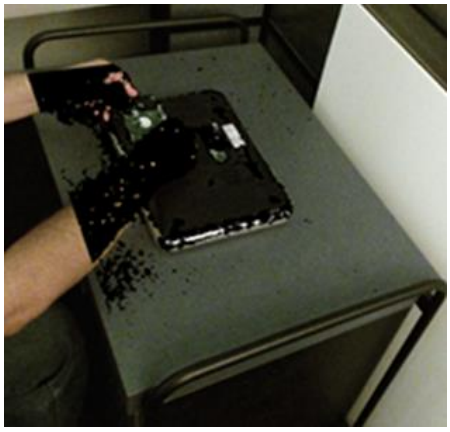
The vise grip assembly consists of five parts and all of them were 3D printed. The 3D CAD models were taken from the Cad software SolidWorks. The vise grip assembly was chosen because: 1) the parts are more complex, 2) the parts were easily available, and 3) it required two hands to assemble the grip making it more complex for skin detection as well as finding the number of steps.

The laptop assembly consists of seven parts and closely resembled a real life assembly sequence. The laptop assembly sequence was chosen because: 1) the parts were sufficiently complex for a bench top assembly, 2) tools were required in some steps, and 3) both hands were necessary.

It is important to note that for all assembly sequences the number of parts translated to the number of assembly steps. Assembly sequences that require sub-steps for each assembly step are not within the scope of this paper.

(a) DUPLO block assembly skin detection

(b) Vise grip assembly skin detection



(c) Laptop assembly skin detection

Figure 27. Example of vise PSOGMM based skin detection in the three assemblies

Figure 27 illustrates one of the frames in each of three test assemblies. It is seen that the majority of the skin pixels are removed while the part is clearly unaffected.

Table 7. List of assembly sequences and metrics

Assembly	Threshold	Total Steps (Actual)	Total Steps (Identified)	Frames	Time(s)
DUPLO blocks	15,000	4	4	205	1.7
Vise Grip	45,000	5	5	271	12.9
Laptop	49,000	7	7	679	69.23

Table 7 describes the metrics of the various assembly sequences. It is seen that the number of assembly steps are accurately identified. “Time” in the table represents the total time taken to process the skin pixels. The time taken for each assembly is different due to two reasons: 1) the number of frames are different and 2) the number of pixels to check are different for each assembly.

Since the same setup was used, the resolution was uniform across the three assembly sequences. Hence, assemblies with larger parts have less background filtered out and require more time to check for skin pixels. Thus, it can be seen that DUPLO blocks (which are small) was processed an order quicker than the laptop assembly (which is relatively large).

Considering the size and length of the assembly sequences the time taken for each assembly was determined to be sufficient, being approximately a minute or less. The algorithm, by using the hands entering and exiting the frame, correctly identified the number of frames for each test assembly. The algorithm is a major step towards enabling expert demonstration

authoring. The algorithm provided a good starting point for the subsequent necessary stages of processing i.e. part identification, animation, and possible refinement (position, orientation or color changes).

4.5. Summary

Currently, authoring AR guided assembly steps is a time-consuming task. Graphics expertise, use of multiple software packages, and significant time is often required to create or modify AR content for an AR viewer. Expert demonstration is an authoring approach that enables the author to physically demonstrate and record the assembly sequence. Ideally, this recorded demonstration is processed to automatically find the AR work instructions.

A critical component of the AR authoring approach is to detect and filter the hand pixels. Since current skin detection techniques are lacking when it comes to handling noisy data a novel method was established known as particle swarm optimized Gaussian mixture model. This is the main contribution of the paper. Through this skin detection method, an algorithm was developed to accurately find the number of steps automatically from the recorded demonstration.

PSOGMM was evaluated against GMM on two datasets and for three assembly sequences the number of steps were correctly found. Time taken for the skin detection method and the apparent mismatch across the three assemblies was determined to be a result of the size of the physical part being captured.

4.6. Future Work

A limitation of the PSOGMM method is that it assumes the noisy pixels to be the low occurring pixels in the histogram. In the datasets dealt with in this paper, this is a reasonable assumption. However, if there are datasets with significant noise then PSOGMM would not be able to work. There needs be more analysis to find the appropriate point at which the method fails.

Finding the number of steps required a user input threshold. This may require the author to run the processing for multiple variations of the threshold to find the appropriate value leading to added time waste. There needs to be further study into trying to find an automated way of discovering this threshold. One possible method is to use K-means that classifies the hand and non-hand frames based on the number of skin pixels.

4.7. Acknowledgment

This research was funded by The Digital Manufacturing and Design Innovation Institute, a federally-funded research and development organization of UI Labs (Award No. 015336).

CHAPTER 5 AUGMENTED REALITY VIA EXPERT DEMONSTRATION AUTHORING (AREDA)

B. Bhattacharya, E. Winer, “Augmented Reality via Expert Demonstration Authoring (AREDA)”, IEEE Access, In Review (2016)

Augmented Reality via Expert Demonstration Authoring (AREDA) is the name of the AR authoring tool developed to illustrate the potential of expert demonstration. Submitted to IEEE Access Journal, the paper below describes AREDA and its workflow from converting a demonstration of an assembly sequence to the automatically generated AR work instructions.

Abstract

Augmented Reality (AR) is an exciting new technology that has been gaining popularity over the past decade. AR has successfully been incorporated into the commercial market with mobile games like Pokemon Go. In academic research, AR has been applied to a variety of areas including medicine, sports, entertainment and manufacturing. The research presented in this paper centers around manufacturing, specifically AR guided assembly. This is the use of AR to provide product assembly instructions to factory workers. AR guided assembly has been shown to outperform conventional (paper based) and virtual based methods of instruction delivery in terms of accuracy and time. However, creating the AR content displayed to the worker (i.e. authoring) is difficult and time consuming, often involving multiple software packages and expertise in several disciplines. The main challenge for an AR authoring tool is to allow the author the ability to register the content intuitively, such that the author does not have to comprehend complicated AR concepts rooted in computer vision and computer graphics. Many

authoring tools, like the Designer's Augmented Reality Toolkit (DART), have tried to overcome this challenge but have been unable to sufficiently create a good working abstraction for the novice author. The work presented in this paper focused on creating AR work instructions for bench top product assembly using a novel form of authoring termed expert demonstration. The working concept is that an expert performs the assembly steps while being recorded by a computer vision system. The recording is then automatically processed to generate AR work instructions. Augmented Reality via Expert Demonstration Authoring (AREDA) is the name of the representative software developed in this research. AREDA is divided into two phases: the demonstration phase and refinement phase. Using established computer vision algorithms along with a newly developed skin detection algorithm known as Particle Swarm Optimized Gaussian Mixture Model (PSOGMM), the demonstration phase was created. This phase takes recorded 3D point clouds of assembly sequences and converts them into ordered 3D parts along with their transformations. The refinement phase then allows a user to modify the automatically generated AR work instructions to fix errors as well as add useful virtual content such as textual instructions. Finally, three use cases, of differing complexity, are presented using AREDA to generate the work instructions. Through these use cases, this paper shows the significance of the expert demonstration authoring approach.

5.1. Introduction

An augmented reality system is defined as "a system that supplements the real world with virtual (computer-generated) objects that appear to coexist in the same space as the real world. An AR system has the following characteristics: it combines real and virtual objects in a

real environment, runs interactively and in real time, and registers (aligns) real and virtual objects with each other in 3D." [8]

Users view the real world through a screen and the AR system overlays graphical objects including 2D textual instructions and animated 3D models. Figure 28 represents a snapshot of a commercial smartphone based AR system known as Yelp Monocle [13]. This mobile AR system combines a real-world image feed with 2D text and graphics, updates in real-time and registers the real world and the various virtual objects based on the user's location and bearing.

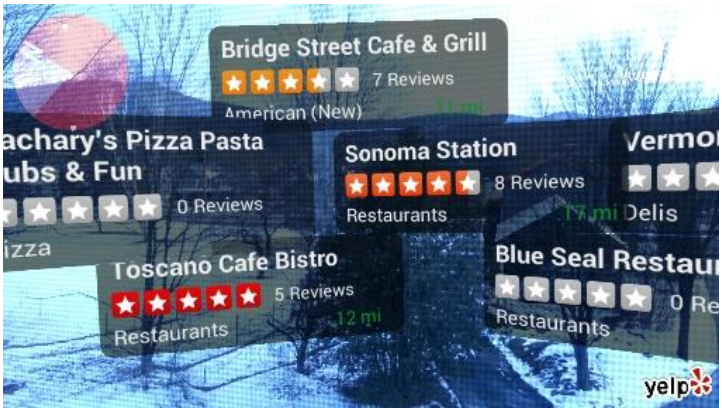


Figure 28. Yelp Monocle screenshot

Consumer Augmented Reality (AR) applications, like Pokemon Go, Ingress, and StarWalk, have made AR a popular form of contemporary media. Zugara provided a list of corporations, including Google, Qualcomm, and Microsoft, that have invested heavily in AR based startups [210], indicating the potential of AR. Juniper research has predicted that by 2018 there will be 200 million users of AR [7]. Ever since Azuma's seminal survey in AR [8], [211], AR research has been conducted in a variety of areas: manufacturing [212], [213], medicine [55], [214], entertainment [46], and education [40], [215].

Thanks to inexpensive modern technology and easily available sensors and sensor information, registration techniques have improved. In many AR applications like Pokemon Go, the primary sensor used for registration is a combination of the camera, GPS, and compass in a smartphone or tablet device. Considerable research [11], [170], [177]–[179] has been done using these types of sensors and have become quite robust and accurate for very large spaces where errors on the order of a few meters are negligible.

As an application's area of interest becomes smaller, GPS and compass sensors are not accurate enough for correct registration between virtual content and the real environment, leading to the necessity of near-range sensors such as camera images [9], RFID tags [10], GPS [11] and infrared markers [12]. Many such sensors have been used in applications such as maintenance [105], [216], [217], assembly [26], [129], [218], and simulation [219], [220]. In these works the AR application is discussed but not the authoring of the content within. Typically, these systems were created and the content was authored by first creating and then manually placing (through third-party softwares) the virtual content with respect to a particular marker. A marker is a physical object placed in the scene picked up by a sensor and used for quick detection and registration. Markers are often pictures or symbols such as AR Toolkit markers [221]. Consider, an industrial engineer wanting to create AR work instructions for the assembly of a product. One of the assembly steps requires the use of a fastener placed in a sub-assembly. Typically, the industrial engineer would have to first model the fastener and the sub-assembly, most likely in CAD software and then apply appropriate colors and textures. This may require the export of the model and import into a 3D modeler such as Maya or Studio Max. Transformations then have to be applied to the 3D models with respect to a marker. This

marker is captured by the camera or other near-range sensor and translated to information used for registration. The industrial engineer will need to understand this and apply the appropriate transformation either from within the 3D modeler or through another software tool designed for this purpose. The final 3D models are again exported for use in the AR viewer. For a novice user, all these steps, conversions, and transformations lead to errors and difficulty in creating usable AR work instructions. Thus, content authoring for AR is a significant challenge.

A popular method of authoring AR applications [46], [130], [222] is using a combination of a game engine like Unity3D [131] that handles the display, graphics, and interaction while a third party library like Vuforia [73] handles the registration. While usable instructions can be created, it requires considerable technical knowledge in computer vision and 3D graphics. In the realm of product assembly, not many workers have this, so methods need to be created that make authoring AR work instructions much easier.

To put this into context, Designer's Augmented Reality Toolkit (DART) [141] was a toolkit developed as a plugin for Macromedia Director for designers interested in creating AR content. In a user review [140], ten years after its initial launch, the developers found a number of challenges with DART. People uninitiated in the concepts of AR did not know how to begin using DART, users were unaware of external issues such as faulty tracking systems, and many of the tools available went unused due to the lack of abstraction between the feature and the user intention.

This paper showcases a novel approach to authoring AR work instructions using a demonstration phase and a refinement phase. This novel approach is known as expert demonstration authoring and subsequently the software used to illustrate this is called AREDA (Augmented Reality via Expert Demonstration Authoring). The initial development of AREDA was designed for manual assembly tasks in a bench top area (i.e. approximately a 1-2 meter work area).

In the demonstration phase, a content expert is asked to demonstrate an assembly sequence that is recorded in 3D and processed to provide automatically generated AR work instructions. In the refinement phase, these work instructions are further augmented by providing additional virtual content like textual instructions, animations, additional graphics, etc.

5.2. AR Guided Assembly

AR technology has been used by design and manufacturing companies [22] for: 1) evaluating interior vehicle designs [23], 2) creating an environment to facilitate collaboration [24], 3) providing assistance during assembly, disassembly, and maintenance of products [25], and 4) contributing relevant information for the safe and better use of a product to an end user. AREDA focuses on the content that AR guided assembly systems use to equip a user with well-defined, well-placed instructions in the real environment. Ideally, all the information necessary to perform a complex bench-top engineering assembly task would be available in an AR environment and a user would not have to refer to a manual.

Figure 29 represents an example of an AR guided assembly system to aid in the assembly of a pump [26]. The system provides a top down view of the assembly area displayed on a screen. At each step the requisite textual instructions are provided with corresponding animated 3D models. These animated 3D models present specific visual instructions of the actual movements that must be performed to complete the assembly. The system combines the real world (assembly area with various real parts) with the virtual world (textual instructions and animated 3D models), runs in real time, with both environments registered in 3D. This registration between the virtual and real environments is illustrated by the accurate size and location of the 3D models with respect to their real-world counterparts. Without this registration, either the 3D models would just be floating with no apparent logical connection to the real world or the movement of the parts would be heavily constrained making the system inflexible.

Figure 29. Augmented reality system to aid user in assembling a pump. Textual instructions are provided on top while a virtual demonstration is presented.

Nilsson [3] studied the acceptance of AR instructions in real work settings and found users preferred AR over conventional methods of instruction. Even with poor interaction most users had a positive experience using AR systems [27]. It was at least comparable to conventional techniques available and reduced the cognitive workload on a user [28]. In some examples it has allowed non-experts to perform complex tasks previously accomplished only by disciplinary experts [29]. However, this is not the case by simply implementing AR. Care must be taken as to the interface presented to a user and the content within.

Specifically in terms of AR guided assembly systems, Boud [30] compared five guidance techniques on an assembly task: 1) conventional 2D engineering drawings, 2) desktop virtual reality (VR) using a 2D display , 3) desktop VR using a stereoscopic display, 4) immersive VR using a head mounted display and 5) AR guided assembly. The result was that the AR guided assembly systems were the most effective. Users completed the studied assembly task approximately eight times faster than using conventional 2D engineering drawings as measured by mean completion time. The AR task also performed significantly better than the best VR condition (Desktop VR using stereoscopic display). There was no significant difference between the various VR tasks. In addition, Henderson [31] established that AR guided assembly significantly outperformed conventional instruction delivery with error reduction of over 60% and a 100% improvement in assembly speed.

5.3. AR Authoring Tools

Morten [135] defines authoring tools as content creation tools specifically catered towards educational content. This work argues that all AR viewers intend to educate the end-

user. Whether it be AR guided assembly viewers that teach a user an assembly task or collaborative environments informing users or each other's intent, this definition holds true.

All the AR systems described till now are termed as AR viewers. These AR viewers take pre-defined virtual content and register them appropriately based on pre-defined sensor readings. Consider, the example of the AR guided assembly viewer of the hand pump shown in Figure 29. Before the viewer can do anything, the following information is at least necessary: 1) define the list of instructions and 3D models, 2) define the list of available sensor readings (in this case the 2D black and white markers present in the real environment), 3) the linking between the 3D models and the 2D markers, and 4) the appropriate 3D transformations of the 3D models. The AR viewer typically already has this information available and uses it to generate the appropriate AR work instructions. This information is generated by AR authoring tools and the author, the person using the AR authoring tool, is responsible for creating the information correctly. On the other hand, the end-user, the person using the AR viewer, is responsible for following the instructions provided.

A good analogy is that of a video editor and video viewer. The video editor is responsible for cutting the video, adding sound and video effects, etc. that is then rendered into a final composite video. This final video is then distributed and viewed by the users through a video player such as a desktop computer or in a movie theater. Similarly, an AR authoring tool takes the various textual, graphical and sensory inputs and creates a final composite of the AR data. This AR data is then read and displayed to the user by an AR viewer on various hardware platforms.

5.3.1. Challenges of authoring AR content

AR authoring is heavily dependent on the content and the real environment used during authoring (authoring context). Simple content (e.g., purely textual instructions) registered in a simple environment (e.g., 2D artificial markers like AR Toolkit markers) are more easily authored. In comparison, using complex content (e.g., multiple 3D models with relevant animations) within a complex environment (e.g., physical assembly with homogeneous color) is much harder to author. Thus, it is important to recognize every authoring tools' authoring context.

The Orion project [223] was a EU led initiative to "forming a network of leading experts to create an informed and authoritative research roadmap for the development of 3D technology, literacy and usage in the framework of archaeology, and specifically in the context of the archaeological museums." This project led to the development of virtual heritage [75], [224]–[226]. The AR section of this project's intention was to develop a system that people used to create awareness of various archaeological sites. The project's needs analysis and research activities will be used as an example of the challenges in AR authoring:

- 3D Scanning
- Quality of 3D representation
- Object registration and content management
- Cost of systems, tracking, and maintenance

Owing to this needs analysis the research activities of the project was divided as follows:

Computer graphics - Computer graphics research was to be conducted to find optimal methods of rendering high quality data in real-time, understand 3D object and scene modeling, and animation and behavior modeling within user constraints.

Scene authoring - Understanding how to create a scene, how the various 3D models (scanned and artist generated) were part of the environment, and how the content is accessed.

Scene experiencing - Questions such as: How do users interact with the scene? What was their experience and understanding of the content available? How immersed were they? What are the current tools and systems available to develop these experiences?

Content management - Since this was a large project intended to provide access to content all over the world, a robust content management systems had to be established.

The needs analysis of the Orion project provides a good representation of the challenges involved in AR authoring. Computer graphics, scene authoring, and scene experiencing are still relevant challenges according to surveys of AR [227]–[231]. Content generation and management may or may not be an issue depending on the field. For example, in the manufacturing industry 3D CAD models of each part used are generally modeled and stored appropriately. On the other hand, chemistry education does not have much content and more likely needs 3D modeling. In this work, content generation for bench top assembly is not considered a bottleneck to the authoring process and the required 3D models are easily available.

5.3.2. List of AR authoring tools

There have been many AR authoring tools developed each with their strengths targeted to solving a challenge. In this section, the authoring tools are categorized in terms of their intended design.

Linking system

A linking system allows the author to connect virtual content with artificial or natural markers, typically through a GUI. In this system, there is a need for the author to test the authored content through a separate AR viewer. Thus, the author constantly needs to iterate by authoring the content, exporting it to an acceptable format, and viewing it in AR for accuracy. Other graphics based concepts, such as rotation and translation of the 3D models may be present. Examples include: Powerspace [175], an authoring tool that takes Microsoft Powerpoint slides and converts them to AR content and AMIRE (Authoring MIxed REality), an authoring tool [232] that links Mixed Reality (MR) Gems (basic content components) to AR Toolkit markers [233].

AR previewer

An AR previewer allows an author to preview content with respect to the expected AR environment within the authoring tool in real-time. This allows for quicker content authoring as there is less need for the author to export their content and view in a separate AR viewer. Haller's authoring wizard [234] registers each step of an assembly with a particular artificial marker. Lee et. al. [235] used artificial markers to register the content as well as author the content. This is an instance of directly authoring content within the AR environment.

Virtual registration

The virtual registration category requires the construction of a virtual environment that mirrors the real environment. The author is then able to place content virtually in the constructed environment. By registering with markers reflected in both environments a 1:1 correspondence between the two environments is established.

Knopfle [89] uses virtual registration to author 3D work instructions and registers the final 3D content using artificial markers. SceneDesigner [36] applies virtual registration along with other computer graphics techniques to develop photorealistic virtual environments that are registered to the real environment using artificial markers.

Hybrid methods

In this category, the authoring tool provides a method to use virtual registration as well as directly authoring on site in the real environment. Ong et. al. [105] used a desktop GUI based application to virtually register content to the real environment as well as use a number of tracking methods (AR Toolkit markers, Parallel Tracking and Mapping, and template matching) to provide real time on site authoring capability.

Context aware

Context aware AR authoring tools apply static (motion of the user invariant) or dynamic (motion of the user dependent) rules to the content during the authoring phase. In many publications, the concept of context aware is assumed to mean that when a user is in view of a particular environment only the relevant content shows up. Since the tracking of a particular marker is linked to specific content, calling it context aware is not novel. At most, it can be termed as implicit context awareness, which is present in all AR systems. Explicit context

awareness where the rules are explicitly authored is demonstrated by Ong et. al. [25], [107]. The researchers provided a method to define rules based on distance of the user to the task location.

Knowledge base

Knowledge base authoring tools are used to incorporate the domain knowledge of the author during the authoring phase. Jo et. al. [95] created a knowledge based system (KBS) based on manuals, technical reports, and other documentation relevant to the author's domain, in this case aircraft maintenance. The author uses this information to create the content.

3rd party packages

Libraries, packages or software based authoring tools are used in conjunction to create AR systems. For example, Unity [131], a 3D game engine, in conjunction with Vuforia [73], a library for handling registration based on template images has been used man times to create AR systems [130], [222], [236].

In these categories there are several issues: 1) the author is expected to know the various computer vision and computer graphics based concepts to properly use the AR authoring tool, 2) some interactions involve a level of abstraction that is not intuitive to the author, 3) modifying any of the work instructions requires a lot of effort on the part of the author, and 4) most of the AR authoring tools do not provide the ability to use any of the legacy information available.

5.3.3. Interfaces involved in AR authoring tools

In this section, several authoring tools with context to assembly are presented to the reader for a detailed understanding of the state of the art. In these AR authoring tools, an AR previewer is assumed to be present. This is critical to enable novice users to author content. The next few sub sections detail various authoring tools classified based on how the virtual content is authored.

Desktop GUI based authoring

This represents the set of AR authoring tools that use a desktop GUI to author content. This is typically the most common form of authoring available as the hardware interaction (mouse/keyboard) is commonplace. Within these set of authoring tools are many approaches to authoring AR content. Some authoring tools allow the development of a graph where each element/node of the graph describes a specific form of virtual content. Ideally, the graph is understood by the AR viewer and displays the appropriate virtual content.

Anisetti [237] presents CoolTour, an AR authoring tool that allows for simple and easy authoring of virtual 2D content in the real environment. The researchers present no user study nor explanation of how the content is registered to the real environment. Only a high level overview of the 2D authored content is presented.

Mobile AR (MAR) authoring

This represents the set of AR authoring tools that use some sort of mobile device (smartphones, tablets, etc.) for authoring.

Markouzis et. al. [238] looked into a number of Interactive Storytelling MAR applications (Mentira, Alien Contact, Mad City Mystery, etc.) with the perspective of trying to find a rapid prototype design. The researchers concluded that user friendly authoring tools are already available, a conclusion that is heavily biased towards GPS + Compass sensors based AR applications. For small scale applications like bench top assembly it is heavily lacking.

A typical method of using MAR is discussed in Yang's Mobile Augmented Reality Authoring Tool [239]. Here the researchers present a smartphone application where all the interaction is done on the smartphone touch interface and is reflected in the previewer. The advantage the smartphone has over regular desktop systems is that it incorporates standard touch interactions and gestures (pinch zoom, touch drag, etc.) that are more commonplace today.

HMD with 2D/3D camera sensor

This represents a set of AR authoring tools that use a Head Mounted Display (HMD) with a 2D or 3D camera sensor. The 2D/3D camera sensor definition is expanded to include regular cameras (e.g., Logitech webcam), near-range depth sensors (e.g., Microsoft Kinect, Intel RealSense, Point Grey Bumblebee), and specialized camera-projector systems [240]. The difference between HMD based authoring tools and mobile based authoring tools is the ability to now use both hands during the authoring process.

In Wang et. al. [219] the user wears an HMD with a regular 2D camera sensor to view the AR assembly simulation. The researchers use feature points selected by the author to

initialize and track the pose of the component(s). Along with a set of markers, the author is able to move and place 3D virtual objects during the assembly planning process.

Hybrid authoring

This represents the set of AR authoring tools that use a combination of previously discussed concepts.

Woo and Ha [241] use a combined GUI and TUI (Tangible User Interface) to author virtual content. A tangible user interface is an interface where the manipulations and selections of the virtual object are done typically through a marker object and the visual feedback is in real-time.

Yu et. al. [242] uses a hybrid approach of MAR and HMD authoring tools. HMD and hand interactions are used to provide a general location of the virtual objects in the real environment while the MAR authoring tool is used to allow for more refined translations and rotations along with other possible refinements.

Demonstration based authoring

In desktop GUI and MAR authoring, there is an abstraction between the AR environment and AR authoring tool. The known keyboard-mouse interactions for desktop GUIs and touch interactions for mobile devices are leveraged to provide more natural abstractions. For example, “pinch zoom” interaction in mobile devices is used to scale the 3D model. These types of interactions are useful only if the author is well aware of the AR concepts involved during authoring. Novice authors unfamiliar with AR concepts would not be able to use these AR authoring tools efficiently.

HMD with 2D/3D camera sensors based authoring try to overcome this challenge by allowing the authoring of content directly in the AR environment. However, this requires the author to learn new gestures or interaction techniques that may or may not be intuitive to use.

Expert demonstration is a novel technique that seeks to address the challenges associated with AR authoring tools mentioned above. In essence, expert demonstration based AR authoring allows the author to demonstrate an assembly sequence in a similar work environment. This demonstration is recorded and processed via the AR authoring tool and appropriate AR work instructions are found. This form of AR authoring has never been done before and is an important contribution to this work.

The research presented in the following sections describes the development of AREDA. As mentioned earlier, AREDA consists of a demonstration phase and a refinement phase. The demonstration phase builds off demonstration based authoring tools and the refinement phase builds off desktop GUI based authoring tools. AREDA attempts to mitigate the challenge of the author having to comprehend complex AR concepts through the demonstration phase and provides quick and easy abstraction to an AR environment through the desktop GUI in the refinement phase.

5.4. AREDA Methodology – Demonstration phase

5.4.1. Overall view

Before going into the details of the methodology an overall view of the various steps is discussed to give the reader a holistic understanding of what AREDA does.

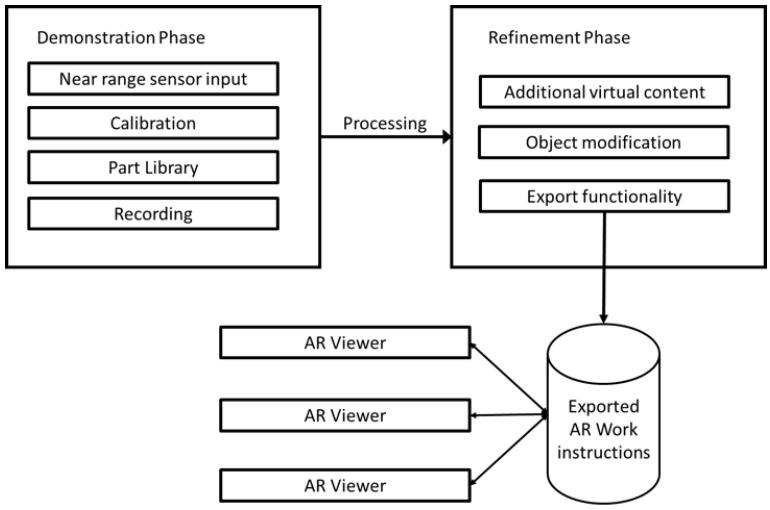


Figure 30. Overview of AREDA system

In Figure 30 there are two main phases, namely the demonstration phase and refinement phase. The demonstration phase is responsible for capturing from the near range sensor, calibration, uploading 3D models to the parts library, and processing the input assembly demonstration to generate AR assembly work instructions. At the completion of the demonstration phase, the AR work instructions generated are only the 3D representations and the transformations captured for each part during each assembly step. These AR work instructions require additional information like textual instructions, color changes and corrections to the object transformations leading to the need of a refinement phase. This phase is responsible for providing additional virtual information and correcting any mistakes made during processing (e.g., improperly detected parts or transformation). Finally, the AR work instructions are exported to a database for use by an AR viewers. Ideally, the end user could use any device (smartphone, HMD, or tablet), launch the AR viewer on the chosen device, and load the work instructions from the database accordingly.

5.4.2. Capture setup

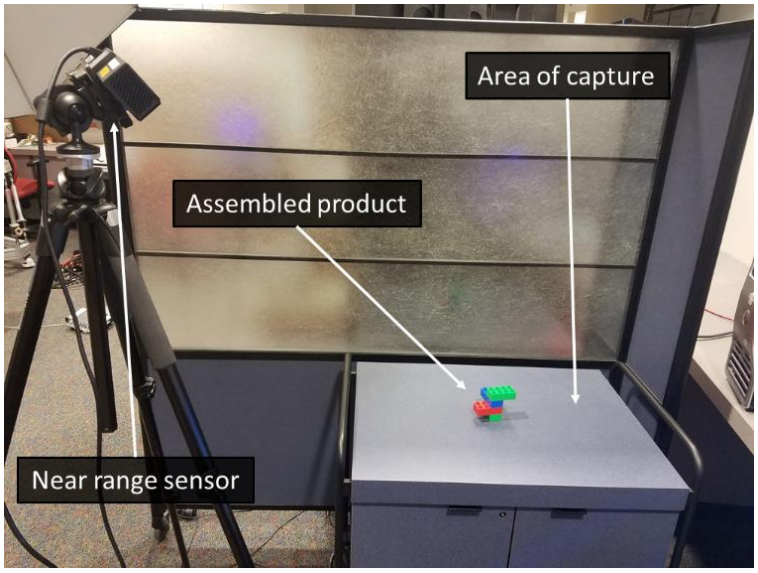


Figure 31. Capture setup

Figure 31 presents the capture setup used in testing AREDA. The near-range 3D sensor is used to capture the assembly demonstration. This research used a Microsoft Kinect, but a wide variety of 3D depth cameras can be integrated into the system. The area shown of a table with a wall represents a typical bench top assembly area. Captures were recorded and processed on a computer with the following specifications: Intel i7 2.6 GHz processor, 16GB RAM, and an Nvidia GeForce 960M graphics card.

5.4.3. Background calibration

Before the actual assembly demonstration is recorded several calibration steps need to be performed. These steps enable AREDA to process and convert the recorded demonstration to AR work instructions. The first is background calibration necessary for: 1) cropping the capture frames to the area of interest and 2) generating a model for the bench plane.

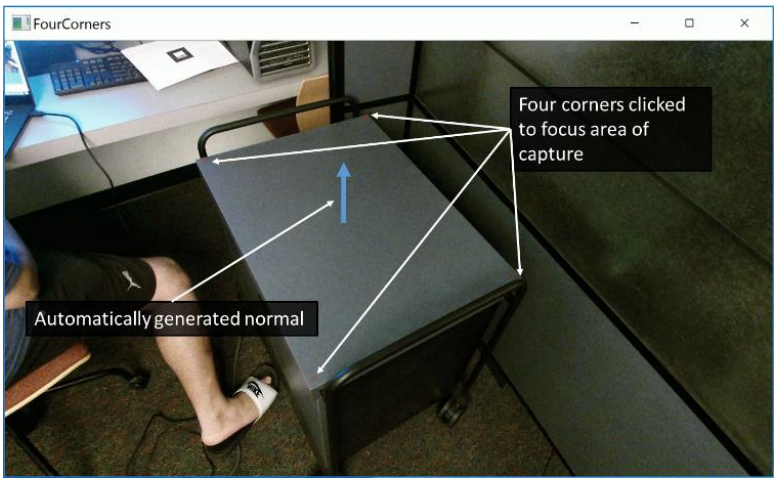


Figure 32. Background calibration

Figure 32 describes the user input required to calibrate the background. The four points clicked by the author are used to crop unwanted regions. The information left is then used to find the equation of the resulting plane using the RANSAC plane fitting algorithm available in the Point Cloud Library (PCL) [207]. Points that are within a user specified distance threshold are removed.

If the bench top is not a perfectly flat plane, the background is saved as a KDTree and background removal is done by comparing the distance of input point clouds with the background point cloud. Points that are within a user specified distance from the background are removed. For the test cases in this research, a distance of 2 mm was used as determined by ad-hoc experimentation.

5.4.4. Area calibration

The next step is area calibration. This calibrates the 3D information from the near-range 3D sensor to the marker that will be used in an eventual AR viewer. This step is not directly used in the processing of the AR work instructions during authoring but is important towards

efficiently deployment to end users. In the AR authoring tools surveyed the correspondence between the virtual objects and the external markers was established manually, typically by trial and error. By adding this area calibration step, AREDA is able to automatically find the proper correspondence between the virtual objects and markers.

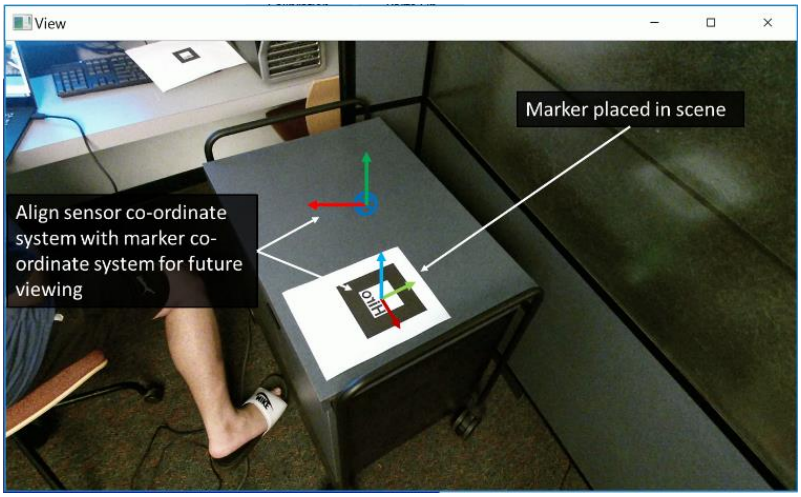


Figure 33. Area calibration

Figure 33 provides an example scene of the area calibration. The image is captured by the Kinect and displays two coordinate systems. An AR Toolkit marker is used, however, this step is not restricted to fiducial markers and any type of marker (fiducial, natural, optical, etc.) can be used.

There are two coordinate systems present, the marker coordinate system and the sensor coordinate system. The marker coordinate system is a Z Up coordinate system while the sensor coordinate system is Z In. In the example shown in the above figure, the image is captured by the sensor and the center of the marker is a point available in both coordinate

systems. By applying the appropriate transformation, the points in the sensor coordinates are converted to marker coordinates using this change of origin transformation.

Consider the center of the marker to be $Marker_{XYZ}$ in the sensor coordinate system (subscript XYZ marks the sensor coordinate system). In the marker coordinate system, the center of the marker is considered at the origin or (0,0,0). Consider, converting a point P_{XYZ} from the scene to the marker coordinate system.

$$P'_{XYZ} = P_{XYZ} - Marker_{XYZ} \tag{7}$$

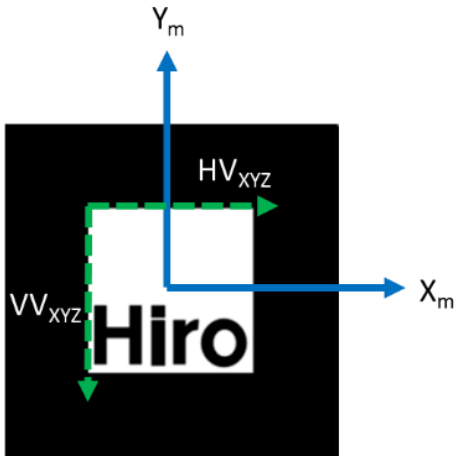


Figure 34. Finding the rotation between the sensor coordinate system and the marker coordinate system

At this stage, the points are correctly translated (described in equation 7), however, the correct rotation is required to complete the change of coordinate system. Figure 34 illustrates the various vectors available during the capture of the marker. The center of the marker as well as the coordinates of the four corners of the marker are available. From these four coordinates

the two vectors HV_{XYZ} (horizontal edge vector) and VV_{XYZ} (vertical edge vector) are found. These vectors run parallel to X_m and Y_m, respectively, in the marker coordinate system. Using this information, the cross product of HV_{XYZ} and VV_{XYZ} gives ZV_{XYZ} (Z vector or the vector normal to the marker) as shown in equation 8. Using equation 9, the rotation matrix is available and the change of coordinate system from sensor to marker is complete.

$$ZV_{XYZ} = HV_{XYZ} \times -VV_{XYZ} \tag{8}$$

$$P_M = \begin{bmatrix} HV_{XYZ} \\ -VV_{XYZ} \\ ZV_{XYZ} \end{bmatrix}^{-1} * P'_{XYZ} \tag{9}$$

Not all sensors will have the same orientation of coordinate systems. However, simple modification of these equations will allow the calibration to work for generically.

5.4.5. Skin calibration

Identifying skin pixels in the assembly steps allows AREDA to: 1) segment and process the assembly parts, 2) find part movements, and 3) distinguish between the various assembly steps. Thus, to enable skin calibration the author provides a demonstration of the hand (as seen in Figure 35). This demonstration is used to form the training data to develop the skin model.

Lighting is an important factor to consider in computer vision algorithms. Lighting changes will affect the skin model and thus it is assumed to constant between the calibration and the recorded demonstration. Significant changes would require recalibration of the skin

model. Further study needs to be conducted in the effectiveness of the current lighting models and in terms of the required calibration steps to be set by author.

Figure 35. Demonstration of hand

Current popular skin detection techniques like Gaussian mixture models (GMM) are lacking because: 1) they are not designed for unclean data, 2) they require the author to understand complicated algorithms to set the parameters appropriately, and 3) complicated computer science concepts like overfitting cannot be easily understood. Hence, a new algorithm known as particle swarm Optimized Gaussian mixture model (PSOGMM) was developed [243].

Figure 36. Skin calibration using PSOGMM based skin model

Figure 36 illustrates the results of using the PSOGMM skin detection model. In this figure, the skin pixels in the capture area have been identified and removed. The advantages of PSOGMM over GMM are that: 1) it has only one user input threshold variable that describes how much of the training data is considered skin and how much is considered background, 2) there is no need to understand complicated algorithms as all the required parameters are set automatically and can be treated as a black box, and 3) PSOGMM is faster than GMM in training by 97% and by 49% in testing while comparable in precision and recall. For complete details on the algorithm, please see [243].

5.4.6. Processing the recorded demonstration

Before processing the recorded demonstration, AREDA requires the 3D model representation of the parts (mesh model and point cloud) to be used in the assembly sequence to be loaded into a part library. Once all the calibration steps are completed, the demonstration of the assembly sequence is recorded and processed. The algorithm for processing and finding the individual parts is presented in Table 8.

Table 8. Processing the recorded demonstration to find individual parts

Step	Description
1	For all frames F
2	F' ← F – B (*background subtraction*)
3	F" ← F' – S (*skin removal*; Keep track of F")
4	If the number of skin pixels removed > *start_of_skin* start keeping track of the number of skin pixels in vector V_{SP}
5	End For frames F
6	Use Kmeans to divide V_{SP} into hand (with more skin pixels) and non-hand (with noisy skin pixels) frames, consequently get each step
7	For every step
8	For all non-hand frames
9	Filter F"
10	Remove pixels that are present within previous frame buffer
11	Pixels left are accumulated into buffer B
12	Push F" to previous frame buffer
13	End for non-hand frames
14	Match buffer B with parts library using PCA + ICP; Save model
15	End of step

5.4.7. Finding the number of steps

Initially, the background and skin pixels are removed using the methods described in sections Background Calibration and Skin Calibration. For every frame, the number of skin pixels is compared to a user defined threshold '*start_of_skin.'*(15,000 in this case). Matching the first occurrence of the hand within this threshold marks the beginning of the assembly sequence and once the assembly begins the number of skin pixels per frame is recorded into a vector V_{SP}.

In the previous version of this algorithm [243] finding the number of steps was based on a threshold value '*hand/non-hand*' for the number of skin pixels that constitute a hand. Getting the appropriate '*hand/non-hand*' threshold was a matter of trial and error and constantly changed with different input parameters. This constant trial and error method contributed significantly to the time of processing. To mitigate this problem, a K-means classification algorithm (K = 2) was used on V_{SP} to classify hand and non-hand frames. This proved to be a simple and effective way to indirectly find out the threshold and classify those frames as hand or non-hand frames accordingly. Figure 37 illustrates an example dataset of skin pixels recorded per frame. It is clearly seen that there is a distinction between the hand frames (upper regions of the blue line) and the non-hand frames (lower regions of the blue line). Figure 37 also describes how the frames based on the number of skin pixels detected were classified into hand and non-hand frames using K-means as shown by the orange line. It is important to note that the value of the label is either zero (non-hand frame) or one (hand frame). The K-means was scaled for illustration purposes to make it visible on graph.

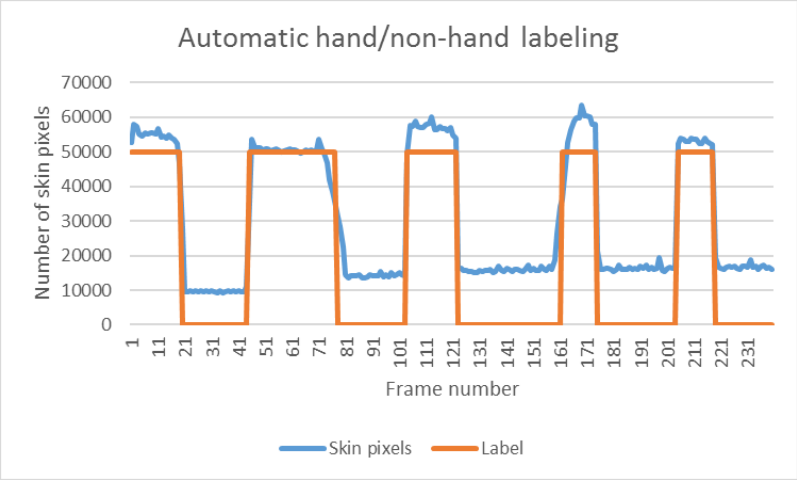


Figure 37. Automatic hand/non-hand labeling; the frame (blue line) are properly classified as shown by the label (orange line)

5.4.8. Finding the part for each step

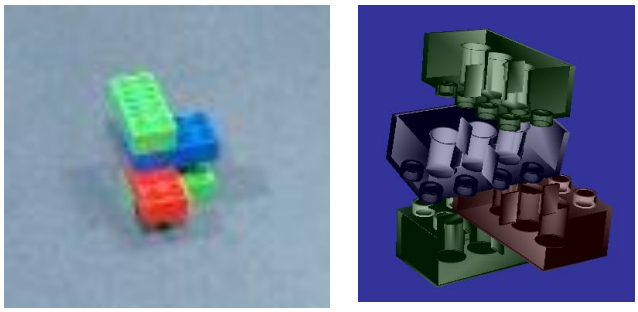


Figure 38. Recorded example of DUPLO block assembly(left); Converted virtual assembly to be used later for AR instructions (right)

Figure 38 presents an example of the conversion that takes place from a step of a recorded assembly demonstration.

The process for finding each part is as follows. The frames between non-consecutive hand frames (i.e. non-hand frames) were accumulated into a point cloud. These point clouds represent the sub-assembly of this step. Sub-assemblies found in previous steps are then

filtered out by finding the spatial overlap in point cloud data between the current point cloud and previous point clouds. For the initial version of AREDA, it was assumed that the sub-assembly does not move and the parts are rigid.

Once the points from the previous step were removed, what remains is the new part added and its location. The number of points in both the scene and model point cloud were made equal. This was done by randomly sampling the larger (by number of points) point cloud. Using PCA (Principal Component Analysis) to find an initial estimation of the transformation required, the Iterative Closest Point algorithm (ICP) was then used to refine the final transformation of the part. Then, the number of points within a user defined threshold between the model point cloud and scene point cloud were counted. The ratio between the counted points and total points was then used as a metric to identify the part available in the parts library.

ICP was used because: it is robust and always converges monotonically and its transformation results provide a metric of distinguishing parts. However, it must be noted that ICP can fall into local minima and can be computationally slow.

Due to ICP's prevalence to fall into local minima, it does not do well with symmetrical or near symmetrical objects. Near symmetrical objects are objects that have very subtle changes in shape and due to poor resolution or occlusion these changes are not identified leading to incorrect results. If the object is symmetrical then it does not matter what the orientation is. However, for near symmetrical objects this is a difficult problem to contend with and instead of trying to use high end hardware or complicated algorithms that increase the processing time,

this is pushed to the refinement stage where the author can manually fix the orientation problems. Further analysis of the ICP algorithm is presented in the Results section.

5.4.9. Finding the part movement

The part for an assembly step has already been identified in the previous section and will be known as the identified part. Table 9 describes the algorithm involved in finding the identified part's movement also known as the part animation.

Table 9. Processing the recorded demonstration to find part movement

Step	Description
1	For all steps
2	For all hand frames F
3	F' ← F - B(*background subtraction*)
4	F" ← F' - S(*skin removal*; Keep track of F")
5	Find average location of part
6	Find closest location of part from final positon
7	End of hand frames
8	For all hand frames beginning from closest location to start of initial movement of part (backwards)
9	Use PCA + ICP to find the final transformation of the part
10	Record as way point in animation
11	End hand frames
12	End step

Generally, traditional animation is defined as a simulation of a movement created by displaying a series of computer generated images or frames. A key frame (as related to animation) is described as the image that is the start or end of a movement. Key frame animation is an animation technique where the key frames are authored and the computer interpolates between them accordingly to create the resulting animation. Similarly, in AREDA, part animation is defined in this manner where each key frame is described by the part position and orientation. OpenSceneGraph [139], the graphics library used by AREDA, takes these key frames and renders the animation accordingly. The hand frames previously unused provide useful information for finding and defining the appropriate key frames.

A typical assembly step has the following progression: 1) the part is manually moved to its appropriate position in the sub-assembly, 2) the part is manipulated with or without a tool(s) to fit within the sub-assembly, and 3) the hands along with the tool(s) are removed and the part for the next assembly step is chosen. Consider an example of assembling DUPLO blocks as shown in Figure 38. A green block is placed, followed by a red block, a blue block, and another green block. At each assembly step, the author chooses the part, manually fits the block with the sub-assembly with their hand, and removes their hand.

Within this progression, it is important to process the first two points but ignore the third one since it does not contribute to the part animation. This is done by processing the hand frames and, like the non-hand frames, the background and skin pixels are removed. The remaining points constitute the part's point cloud. For each non-hand frame, the mean position of the part is compared to the final position of the identified part and the frame that is closest

to this final position is calculated. Steps 2-7 from the algorithm in Table 9 describe the method to find those frames that are involved in the part animation.

From this closest frame, the point cloud, found from the scene, is matched with identified part using ICP to find the intermediate orientation of the part. Similarly, the frames (going backwards) leading up to the start of the assembly step are processed to get the entire part animation.

To say it another way, the part detected in the scene has already been matched to a part in the library and identified. The hand frames now constitute the motion of the identified part. These hand frames are filtered for background and skin and the part found in the scene is matched with the identified part to find the orientation and position. The various orientations and positions are now reconstructed as a key frame animation.

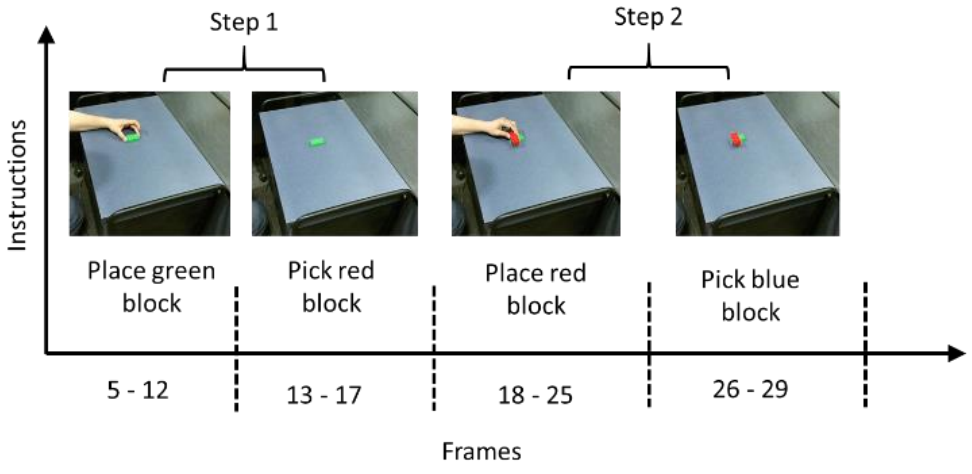


Figure 39. DUPLO block assembly sequence

Figure 39 describes the first two steps of the DUPLO assembly sequence as a graph of frames vs. instructions. Frame 5 marks the start of the assembly i.e. the frame when the green

block's movement is discovered. The hand is removed and the red block is chosen. The red block is placed and subsequently removed from the scene. Then the second step, where the red block is placed, occurs. Frames 18 to 25 are the hand frames. Within those frames only 18-22 contribute to the red block's movement (found via steps 2-7 in Table 9). From the previous section, the final position of the red block is known and is considered the final position in the animation. Frame 22 is processed accordingly and is considered the penultimate positon for the animation. This processing continues backwards to Frame 18 which marks the beginning of the red block's assembly step. Thus, the part animation of the red block is found

5.5. AREDA Methodology - Refinement phase

The AR work instructions automatically generated in the previous phase consist of a virtual representation of the part (i.e. 3D model available from the parts library) and its corresponding 3D transformations. In this section, details are provided regarding other virtual information added to the work instructions and the design choices made in the interface.

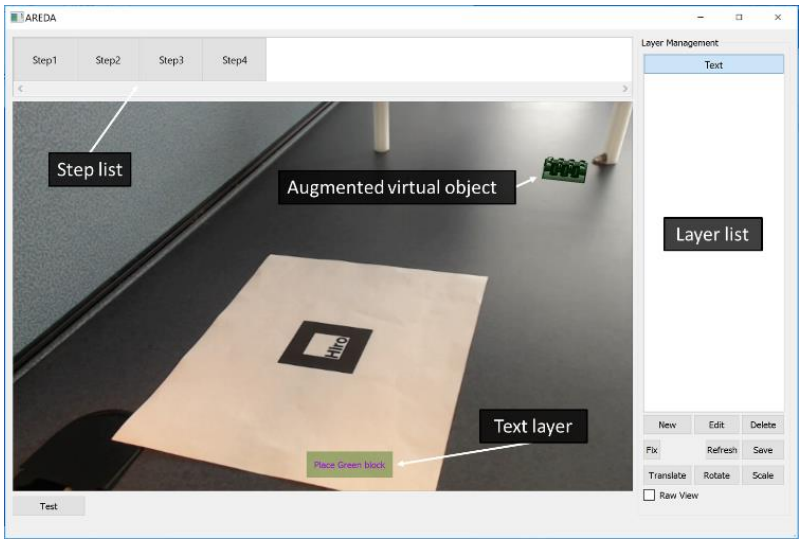


Figure 40. Example of AR previewer

Once the demonstration is recorded and processed, the automatically generated AR work instructions are viewable in the refinement phase. Figure 40 presents an example of the refinement phase interface available in AREDA. It consists of a step list, an AR previewer and layer list. The step list allows a user to view any automatically generated AR work instruction in the AR previewer. For example, the green DUPLO block visible in the figure has been rendered based on the processing of the four-part DUPLO assembly described in the previous sections. The number of steps are also automatically found.

The AR previewer is responsible for displaying any changes made to the AR work instructions during the refinement phase. In the middle is the preview window that displays all the AR content in real time. An author can view the current instructions and changes are they are made in real-time. It is noteworthy that all the information initially populated in the refinement phase was automatically generated form the demonstration phase. To this point, the user has only had to click four points for area calibration and then place their hand in the scene for skin detection calibration. Then, they proceeded with the assembly. This clearly shows how AREDA is able to function without requiring a user to have expertise in computer vision and 3D graphics.

In Figure 40, the layer list is responsible for adding, deleting or editing any additional layers to each AR work instruction.

5.5.1. Layers

Before diving into the details of the types of virtual content available, it is important to understand the way these virtual objects are setup. Each work instruction is composed of layers and each layer defines the AR graphical overlay involved.

Default layer

This layer is real-time video input. All virtual content is placed on this layer. AREDA assumes access to a camera is present and uses it for the AR previewer.

Part layer

The part layer is the generated part 3D model and its transformations. This layer is responsible for any effects required on the part. This was made a separate layer since part modifications were a necessary step into improving the overall capabilities of AREDA. Fixing mistakes in part selection and transformation, improving 3D model visualization with highlight effects or occlusion are just some of the possible improvements that can be part of this layer. The part layer is also capable of creating assembly steps without the addition of an actual part. For example, consider if one of the assembly step's was to perform intermediate quality analysis (QA) by viewing the sub-assembly at a complicated viewing angle. The expert during the demonstration would illustrate this and even though this viewing angle isn't automatically generated by AREDA, by modifying the part layer the author can quickly describe the step easily.

2D text, image and video layers

The purpose of this layer is to allow the author to enhance the AR work instructions with 2D textual, image or video instructions. This enables the author to repurpose previously

generated paper based instructions and videos. 2D text and image layers can be placed on top of the AR video input and freely moved around. However, the video layer is placed outside the AR video input by design to not distract the end-user while performing an assembly step. These layers are visible in the layer list.

5.5.2. Managing the layers

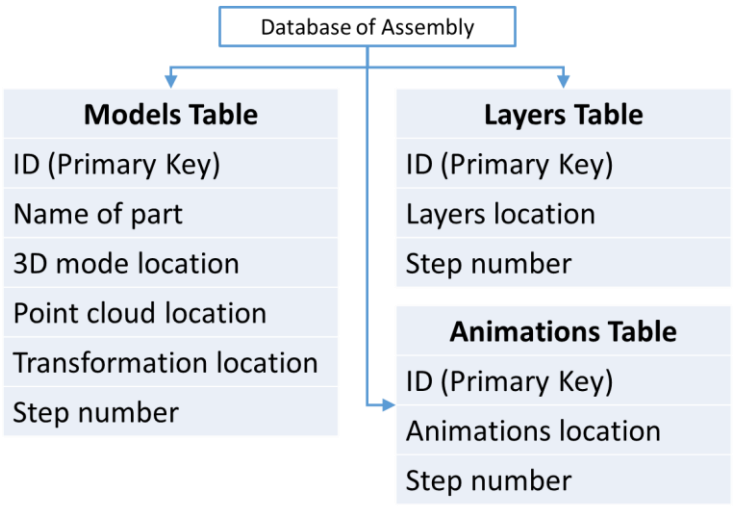


Figure 41. Database structure of AREDA

As each layer is added, they are locally stored in a SQLITE database. Figure 41 presents a view of the internal format used to save the various layers. There are three folders along with the database named “Models,” “Animations,” and “Layers.” There are three tables in the database named in the exact same way. The Models table keeps a link of the various 3D models used at each step as well the location of the 3D model file in the Models folder. The models database includes the changes made in the parts layer. Thus, any changes made in the parts layer are reflected and saved through the models database. The Animations table links the various steps with the animations i.e. the transformations involved in the part movement. Each

animation is stored in the Animations folder and the Animations table has a link of the location of each animation file. Animations are stored through the internal OpenSceneGraph file format. Similarly, the Layers table keeps a record of each externally added layer (2D text, image, video) and their corresponding step while the actual layer file, written using an internal file structure, is saved in the Layers folder.

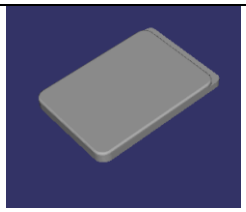
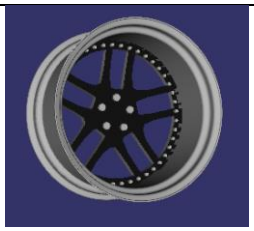
In general, all actions performed by the author are auto-saved. This database can be in a central repository that is then used by other different AR viewers that can parse the work instructions and display them appropriately or exported into a file format for input into an AR viewer.

5.6. Results

5.6.1. ICP results

To verify the efficacy of the ICP algorithm for part identification an experiment was setup. A random list of 12 3D models were downloaded from TurboSquid [244]. Care was taken that the 3D models had differing levels of size and symmetry (4 of the 12 models are shown in Table 10 for illustrative purposes) and the 3D models were converted to point clouds using a uniform density sampling method within the Cloud Compare [245] software.

Table 10. Subset of model list

Part	Preview	Axes of symmetry	File Size
Chair		2	40 KB
Chip		3	112KB
Scraper		1	1,413KB
Wheel		5	1,486KB

The different point clouds were then synthetically processed to add three levels of Gaussian noise (mean of zero and standard deviation of 0.1 ,0.5, and 0.7). Further, these noisy point clouds were transformed about each of their X, Y, and Z axes by four angles 0, 10, 30, and 45 degrees. This resulted in 12 synthetically created point clouds for each individual 3D model.

This was to create representative point clouds that might be found during an expert demonstration and have to be matched to a part library.

These representative point clouds were then matched to their original counterparts with two things being evaluated, the angle and final ICP result. The angle was deemed to be correct if the final transformation was within five degrees of error of the actual. The 12 variants for each of the 12 3D models were matched to original 3D model list for a total of 144 runs of the matching algorithm.

Table 11 describes the various models and their respective correct recognition and orientation percentages averaged across the 12 results found for each model. It shows that recognition was quite accurate averaging 95% (standard deviation 4.4) across the models. In terms of the orientation, the average was 81% (standard deviation 12). This loss in performance in orientation happened because there were symmetrical axes that the ICP algorithm was unable to handle. In the case of the Demon part that had no axes of symmetry, the correct orientation was found every time.

Table 11. Results of ICP comparison

Model	**Avg. Correct recognition (%)**	**Avg. Correct orientation (%)**
Chair	98	67
Chip	92	97
Scraper	100	83
Wheel	94	59
Demon	100	100
Gears	97	86
Chair 2	86	90
Milestone	93	87
Boxcar	99	73
DUPLO block	99	74
Laptop battery	95	83
Laptop microchip	100	74

Based on these results the part matching algorithm was deemed effective. Of note in this approach was that the ICP algorithm was used to match models that were complete. In the case of the Kinect capture since only one direction of information is available, the part found from the scene is often incomplete. Further tests need to be done on the efficacy of the method by mimicking the Kinect capture and synthetically cropping the dataset at various

random orientations. However, as seen in the test assemblies, the method performed accurately so this issue was minimal at best.

5.6.2. AREDA results

In this section, the results of AREDA on three different assembly sequences, DUPLO blocks, vise grip, and laptop, are discussed.

The assembly product (DUPLO blocks) was used for three reasons:

1. Due to the connective nature of the blocks, issues like part slippage was not a concern.
2. Each block is easily identifiable.
3. Modeling the 3D blocks was considerably easy.

The vise grip assembly was used for the following reasons:

1. It is a more complicated assembly than the DUPLO blocks.
2. Modeling of the vise was not required as it was part of the example assemblies in the SolidWorks software tool.
3. Relative sizes between the various parts was not too large.
4. Some of the parts would be partially occluded during assembly.

The laptop assembly (see Figure 42) was used for the following reasons:

1. Some parts were flat and small and would likely be difficult to identify.

2. Relative sizes between the parts were very different. The laptop body is considerably bigger than the screws involved in the assembly.

3. The assembly required the parts to be placed within the laptop body which would make it difficult to extract the point cloud.

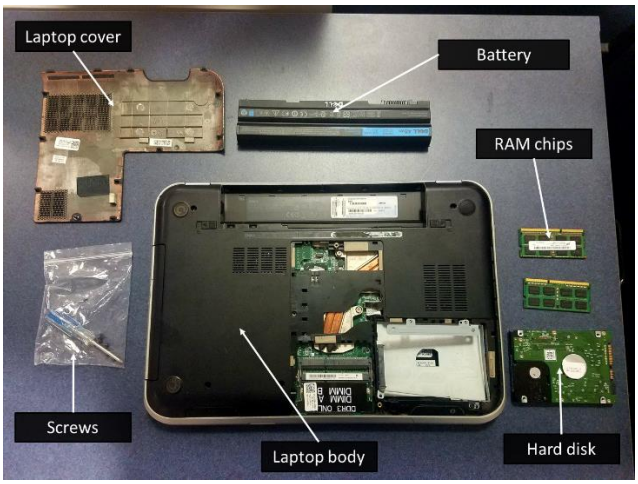


Figure 42. Laptop assembly parts

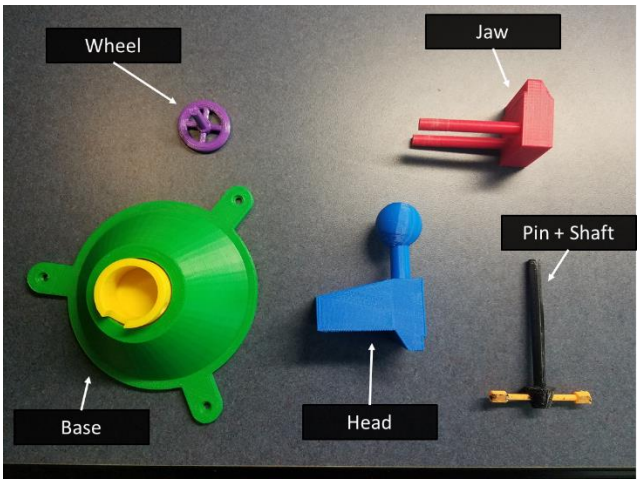


Figure 43. Vise grip assembly parts

Figure 43 illustrates the various parts involved in the vise assembly. The vise assembly is used to illustrate the various steps involved in the demonstration step.

Figure 44 shows the vise assembly available in Solidworks [180] and its 3D printed equivalent using a Makerbot [246]. These parts were then exported to an STL format that was then imported into Cloud Compare. Cloud Compare provides options for sampling a mesh. The same sampling density for the models in the ICP results section was used. This sampled point cloud was then exported to a binary PCD file format (part of PCL's readable files) that was imported by AREDA.

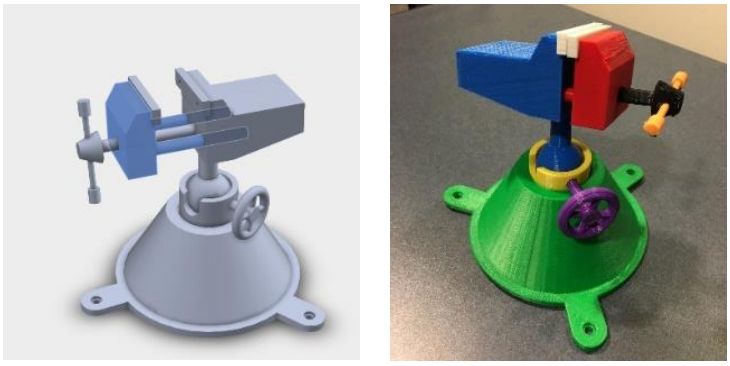


Figure 44. Vise assembly modeled in Solidworks (left); the 3D printed equivalent (right)

Table 12 provides a list of the vise parts and their equivalent point cloud sizes. The order that is displayed is also the order of the assembly sequence. First the base is placed on the table, second the head is fixed on top of the base, third the jaw is attached to the head, pin + shaft is fixed through the jaw to the head, and finally the wheel is attached to the base.

Table 12. Vise grip parts and point cloud sizes

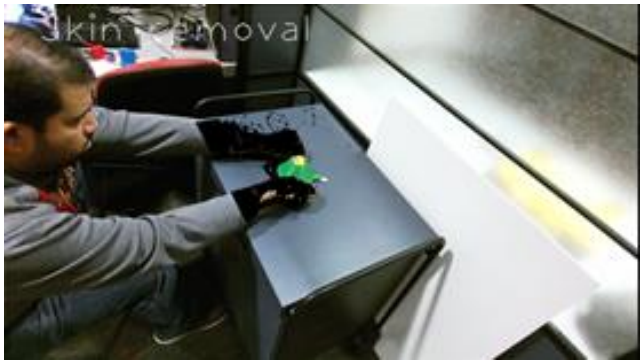
Part	Point cloud size
Base	89260
Head	33923
Jaw	15694
Pin + Shaft	6756
Wheel	4695

After the calibration was completed, the processing began with finding the number of steps in the assembly. Based on the background calibration, the area of interest and a plane model is generated that is compared to the scene and the background (or the bench top) is filtered out. The skin calibration allowed the user to generate a PSOGMM based skin model and used to filter the skin pixels.

Figure 45 illustrates the background removal and skin removal methods in the first and last step of the assembly. The above images are what the author sees during processing. This is done to identify any errors that may have to be corrected by a re-calibration or during the refinement phase. For example, during the authoring of the vise assembly using AREDA, the hand demonstration was accidently performed with separate lighting to the actual assembly. This was immediately realized during this initial processing step and processing was terminated. A new skin model was then generated using the same lighting as per the assembly demonstration. Since the skin model was based on the PSOGMM, the training time was considerably reduced and generation of the new skin model did not hamper the overall time of

the calibration. The entire process of recording the hand, generating the skin model, and start testing it against the recorded data took less than two minutes.

(a) Background removal



(b) Skin removal

(c) Final result

Figure 45. Vise assembly processing begins with removal of background and skin

After the background and skin processing was complete, what remained was the individual point clouds of the actual parts. During this point of processing, the number of steps were found and the frames that involve dynamic hand motion (for finding animation of parts) were separated from the static non-hand frames (for finding the individual parts).

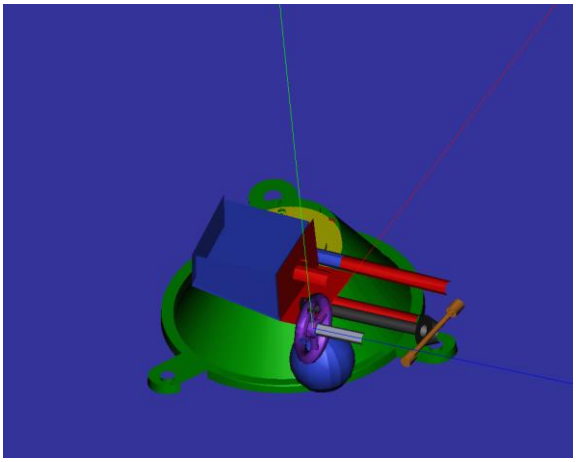
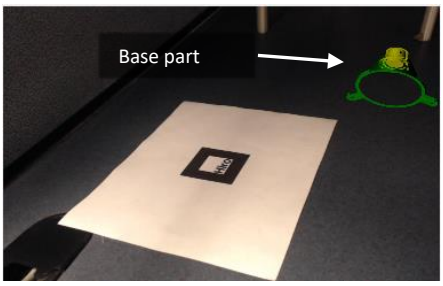
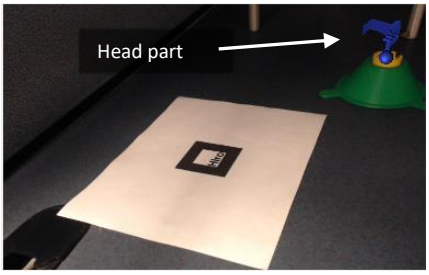


Figure 46. Virtual conversion of the vise assembly

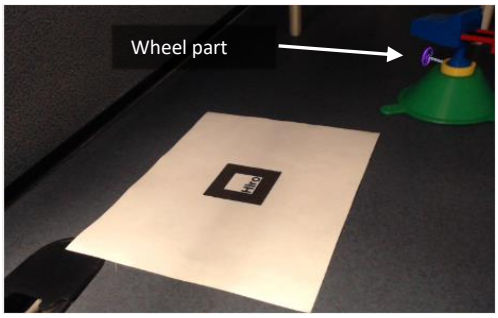
Figure 46 showcases the virtual representation of the recorded vise assembly demonstration after processing. It is clear that some of the parts are not clearly aligned. For example, the jaw (red part) is flipped 180 degrees about the Z axis. However, these were easily fixable in the refinement step and could be done while viewing each step in AR previewer.



(a) Step 1 – Place base part



(b) Step 2 – Place head part



(c) Step 3 – Place wheel part

Figure 47. Vise work instructions; Step 1,2 and 5 are shown

Figure 47 illustrates three steps of the vise assembly namely Step 1) Placing the base part, Step 2) Placing the head part, and Step 5) Placing the wheel. These instructions are generated after further refinement of the positions of the parts. Other features such as adding textual instructions have not been displayed.

Table 13. Time taken (secs) for individual processing for each assembly

Processing step	DUPLO blocks	Vise grip	Laptop
Calibration	21.45	37.91	39.66
- Rec. BG	0.15	0.36	0.22
- Rec. 4 corners	8.93	9.14	8.46
- BG Model	0.28	0.38	0.50
- Rec. Area	0.07	0.17	0.11
- Area calibration	0.24	0.28	0.27
- Rec. Skin	1.72	6.28	6.62
- Skin Calibration	5.72	11.67	13.95
- Test Skin	4.31	9.60	9.50
Main Recording	11.25	23.70	66.92
Number of Steps	13.92	60.93	187.35
- Process BG	0.87	4.32	11.86
- Process skin	1.78	12.91	69.23
- K-Means	0.00047	0.007	0.01
- Visualization	11.26	43.68	106.24
Step 1	174.40	405.67	219.48
- Preprocessing	4.12	49.31	40.17
- Matching	170.27	356.35	179.30
Step 2	170.66	170.12	375.47
- Preprocessing	3.57	51.28	153.23
- Matching	167.08	118.82	222.21
Step 3	228.49	193.05	301.45
- Preprocessing	5.96	51.01	159.18
- Matching	222.51	142.01	142.23
Step 4	153.88	175.99	576.15
- Preprocessing	1.64	49.90	576.01
- Matching	152.23	126.07	0.058
Step 5	NA	197.53	2257.93
- Preprocessing	NA	83.64	2119.51
- Matching	NA	113.86	138.28
Step 6	NA	NA	3122.07
- Preprocessing	NA	NA	3121.83
- Matching	NA	NA	0.049
Step 7	NA	NA	1647.15
- Preprocessing	NA	NA	1482.32
- Matching	NA	NA	164.80

Table 13 displays the time taken to complete each step of the AREDA authoring process for the three assemblies. The processing steps marked by dashes are sub-processing steps of the nearest processing step marked in bold above. For example, Process BG (processing background), Process skin (processing skin), K-means and Visualization are all sub-processes of the main processing step **Number of Steps**. The process can be broadly decomposed into two sets, one for preparing the captured data to determine the number of steps. This includes calibration, main recording, and number of steps and is represented by the blue shaded cells in the table. The other is matching the parts and their positions in the steps themselves and is represented by the green shaded cells in the table.

The calibration step for each of the three test cases took less than one minute. This is an important result illustrating how quickly this can be done regardless of the assembly to be recorded. The actual assembly recoding time is dependent on the size of the assembly, complexity of operations, etc. It is assumed that a user knows ahead of time how much time the assembly will take to complete. Finally, finding the number of steps is dependent on the following points:

1. The setup's area of interest i.e. the area within the four points chosen during the background calibration. The larger the area of interest, the more time will be taken in the background processing.

2. The size of the parts involved. If the parts cover a majority of the area of interest, then there will be more pixels to process during the skin processing.

3. Feedback to the user in terms of the visualization involved.

Point 1 can be verified as a logical result. Since background processing checks for every pixel whether it is part of the background or not. If the area of interest increases, the number of pixels to process will increase and subsequently the time to process the background. Point 2 can be verified by the results seen from Table 13. The setup is identical in the three assemblies, hence the background processing is relatively the same. The DUPLO blocks were small and only one hand was used for assembly so the skin processing took the shortest amount of time (1.78 secs). The vise assembly parts were small but both hands were used in assembly so it took more time in skin processing (12.9 secs). The laptop assembly parts are relatively large, both hands were used during the assembly and so it required the most time during the skin processing (69.2 secs). These three values are highlighted in the table for easy reference. Visualization at this stage of processing is a necessary step for AREDA as the author can determine if the calibration was incorrect and stop processing accordingly. Without visualization, it would be challenging to determine any mistakes made in the calibration from the final processed results. Hence, it was important to analyze the effect visualization had on processing the number of steps. Point 3, verified by Table 13, illustrates that finding the number of steps was dominated (over 50% contribution to the processing time of **Number of Steps**) by the visualization feedback provided to the author. These values are also highlighted in the table. However, it had a small effect on the overall processing time (less than 1%). Thus, it can be interpreted as an apt place for visualization, as it provides useful information to the author with minimal impact on the overall processing time.

At this point, AREDA has added a minimal amount beyond what the actual assembly time is. Now, each part must be determined in each step (green cells in the table). The processing of each step is based on two main metrics:

Preprocessing – Filtering the data for noise, removing overlapping point clouds from the previous steps (so as to get the new part only) and clustering it to find the largest point cloud (assumed to be the part).

Matching – Performing PCA + ICP against the parts library to find the best match.

In terms of the preprocessing metric, the DUPLO block assembly had a mean time of 3.75 secs with a standard deviation of 0.6 secs, the vise grip assembly had a mean time of 57.02 secs with a standard deviation of 14.89 secs and the laptop assembly had a mean time of 1093 secs with a standard deviation of 1186 secs. This was computed by averaging the preprocessing times for each step for a specific test case. For example, the DUPLO blocks averaged 4 numbers as there were 4 steps in this assembly. It was seen that the preprocessing time for each assembly step was increasing quite dramatically for the laptop assembly. This is attributed to the fact that the assembly consisted of small flat parts compared to the base so at every step, the base had to be removed, which was a significant number of points. Hence, parts that are relatively small and comparable to each other perform better during preprocessing as evident from the mean time and standard deviation of the three assemblies.

Moving onto the matching sub-processing for each step, the DUPLO block assembly had a mean time of 178.03 secs with a standard deviation of 30.68 secs, the vise grip assembly had a mean time of 171.42 secs with a standard deviation of 103.92 secs and the laptop assembly

had a mean time of 120.98 secs with a standard deviation of 87.14 secs. This is an interesting result as the laptop had more parts in the library to match yet the mean time for matching is lower. Digging deeper, it was seen that some parts were very small (like the screw) so the part from the scene was composed of a few points, hence matching time was essentially negligible. On other hand, the DUPLO blocks were all the same size and matching required more iterations in ICP to arrive at a conclusion. Thus, DUPLO blocks took more time to match but they were consistent across the steps (i.e. they had the smallest standard deviation across the assembly steps).

In terms of the accuracy of the matching, AREDA automatically detected all the parts in test cases 1 and 2. However, in the laptop test case, the system struggled to identify the parts. Further analysis showed that because one part (a screw) was very small it was being matched to every point cloud found in the scene in every step. By removing it from the part library and reprocessing the individual parts in the laptop assembly were identified correctly. This is an important result as there is a contingency established in case this happens on other parts. There is a need to study the ratio between the largest and smallest parts in a part library and see at what point does AREDA fail.

5.7. Conclusion

In this paper a method is proposed to automatically generate AR work instructions using a newly developed software system, AREDA. AREDA is an AR authoring tool with two phases: the demonstration phase and the refinement phase. The demonstration phase consists of several in-house developed algorithms that enabled automatic generation of AR work

instructions from a recorded demonstration. A workflow had to be generated that provided comprehensible abstraction to the author.

This paper also described the different forms of calibration necessary while keeping in mind ease of use for the author. Area calibration is a new concept introduced that enables easy conversion of a real (sensor) coordinate system to a virtual (marker) coordinate system.

An improved skin detection algorithm known as Improved Particle Swarm Optimized Gaussian Mixture Model was developed and compared to traditional PSOGMM.

Processing time was measured for the various steps leading to the conclusion that AREDA is optimal for bench top assemblies with relatively small parts. The moment the parts are too big processing slows down during matching of parts against the library.

The refinement phase is used to provide additional information to the end-user as well as correct mistakes during the demonstration phase. During this phase, all the information viewable and modifiable by the author was represented as layers. A database is used to save other necessary information that the refinement phase later provides through the various layers. An example of this entire process was shown with three different assemblies.

5.8. Future Work

One of main limitations of AREDA is the fact that sub-assemblies cannot be moved at any time during the assembly demonstration step. Using ICP in its current state would possibly work but with the sheer volume of data involved it would add additional processing. Adding some form of marker to the base sub assembly would help but it would require the base to be

marked in the final AR viewer as well. Finding better natural feature techniques to track the base is an option to be explored.

Creating a separate AR viewer that takes in the database and converts it to visible AR work instructions is another direction that needs to be considered. Ideally what can be done is create a suite of AR viewers on different devices or via Unity since it already has capabilities to export to different devices. This would provide a good metric in understanding the in between work necessary for the author or end-user to convert the AREDA based work instructions to the AR viewer of choice.

5.9. Acknowledgements

This research was funded by The Digital Manufacturing and Design Innovation Institute, a federally-funded research and development organization of UI Labs (Award No. 015336).

5.10. Additional Results

Due to space considerations in the previous paper not all the results were illustrated.

Figure 48 (next page) ilustrates the various stages of processing involved for the DUPLO blocks assembly and laptop assembly. As described in the paper, they successfully remove the background and skin pixels and provide the ICP matching with the appropriate part point clouds from the scene.

(a) Background removal of DUPLO block assembly (left) and laptop assembly (right)

(b) Skin removal of DUPLO block assembly (left) and laptop assembly (right)

(c) Final point cloud of DUPLO block assembly (left) and laptop assembly (right)

Figure 48. Various processing stages of DUPLO blocks and laptop assembly

Figure 49 illustrates two AR work instructions in the laptop assembly, namely the first step of placing the laptop body and the second step of fixing the hard disk in the correct place.

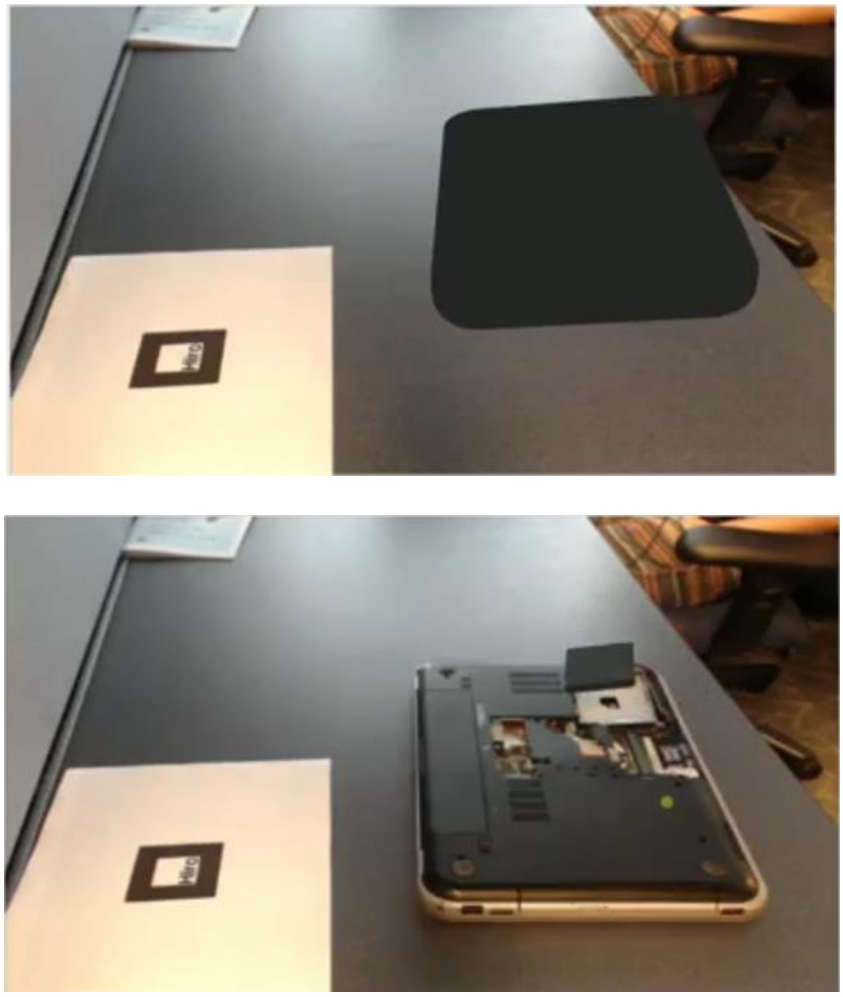


Figure 49. AR work instructions for laptop

CHAPTER 6 SUMMARY

Augmented reality is a technology that has evolved over the past decade. In the context of assembly, AR guided assembly viewers have shown great promise in increasing efficiency, reducing errors and reducing the overall time taken by assembly workers. To bring AR to the commercial market, the challenge lies in efficient authoring of virtual content. Nowadays, authoring is limited to the AR experts who understand the various AR concepts and are able to create AR work instructions. To try and remove this gap in knowledge between the AR expert and the novice author a new authoring technique known as expert demonstration was proposed. Based on a review of the various authoring techniques and technologies available, expert demonstration was determined to mitigate the following challenges: 1) establish automatic registration of parts in an AR environment during an expert assembly demonstration, 2) identify, track and find the orientation of the part or tool within the AR environment, and 3) design the interface of the AR authoring tool.

AREDA was developed as an AR authoring tool that identified the key technological challenges involved in mitigating the research challenges. It was divided into two phases namely the demonstration phase and the refinement phase. The demonstration phase consisted of recording the demonstration and processing to get the AR work instructions. The refinement phase consisted of taking these automatically generated AR work instructions and modifying them to fix errors possibly present in the previous phase, adding text, images or videos, and making sure that view is correct.

A workflow was established that required the author to set calibration parameters, namely background, area and skin before processing the recorded demonstration of the assembly. Care was taken that the various calibration parameters were designed to ease the author into the authoring process and not overload them with too many concepts. This lead to the development of a novel algorithm known as particle swarm optimized Gaussian mixture model (PSOGMM) and enabled the author to quickly generate skin models. PSOGMM was shown to perform at par with traditional GMM on independent datasets, however, it outperformed in both training and testing speed. The processing of the recorded demonstration lead to the development of the automatic detection of the number of assembly steps, ICP based part matching, and animation generation.

AREDA was tested across three assemblies, namely DUPLO blocks, vise grip, and laptop, and results provided show that the various assembly parts were identified and registered as viewed by the AR previewer in the refinement phase. Time analysis was also conducted on the various stages of processing to see the effect of the types of assembly parts involved.

CHAPTER 7 FUTURE WORK

This work presented AREDA, an AR authoring tool that uses the expert demonstration authoring technique. AREDA is divided into two phases namely the demonstration phase and the refinement phase. AREDA can record and process a demonstration of a manual assembly sequence and automatically convert it into AR work instructions. However, there are several limitations that need to be addressed:

1) **The base of the assembly does not move or parts once fixed remain in the same position throughout the assembly**

This is a limitation that restrains the type of assembly can be used. To remove this constraint, it is necessary to be able to track the base of the assembly. This can be done either by using a marker attached to base and tracking it. Further calibration has to be done to properly register the base with the marker. Natural features can also be used to track the base however processing time will increase.

2) **ICP does not handle flat parts and small parts very well**

This is a challenge that is still an open problem in research. Short of using expensive high end hardware with improved spatial resolution that can pick out the subtleties in part shape, the other possible route is to use added computer vision based preprocessing techniques in object identification in both 2D images and 3D point clouds (e.g., SIFT, HoG or SURF based feature matching).

3) **Area calibration is specific between the sensor coordinate system and the marker coordinate system**

Area calibration needs to extended to generalize the conversion between the two coordinate systems. A possible solution is to allow the user to manually transform the coordinate system. In AREDA, the ARToolkit marker and the Kinect are used. Now consider, that 3D point cloud data of the AR Toolkit marker is captured and visualized in a 3D virtual environment with the center of the marker as the origin. Now the author moves the point cloud data such that marker coordinate system matches up with the virtual coordinate system (in this case it represents the sensor coordinate system) displayed. This way a general method of calibration is established between any two sensors. However, the burden of calibration is now placed on the author who needs to know the coordinate systems for both sensor and marker.

4) **Processing is time consuming**

AREDA at its core has many CPU based image processing algorithms that can be parallelized. Analysis can be done to find the bottle necks of processing and reimplemented for maximum throughput. For example, the entire background calibration and skin detection can be parallelized where the frame and masks are sent to the GPU, appropriate processing takes place on each pixel and the new mask generated is returned.

It is clear at this stage, that expert demonstration is a valid method of authoring AR work instructions. AREDA at this point is limited to bench top assemblies and single point of view sensors. Assembly is a massive component of the manufacturing industry and ranges from small assemblies like soldering microchips to large scale fuselage assemblies. To be able to

cover all forms of assembly more work is needed in expanding the types of sensors, the methods of calibration, and speeding up processing algorithms. However, the fundamental algorithm of filtering out the background and skin to get the part that is then used to do part matching and generate the AR work instructions will not change. For example, consider the case of the fuselage and adding the interior parts to the fuselage (chairs etc.). Let us assume, that we are able to track the person either by MOCAP (Motion Capture) technology or by image processing if we force the author to wear a single colored suit. As the author is placing parts, sensors either just outside the fuselage looking in or on the body of the author are recording the demonstration of the assembly. Using the same algorithm as before and with all the sensor information available, the author should be able to convert the demonstration to AR work instructions. It goes to show that the versatility of AREDA remains only in the type of information it able to capture and process. The underlying algorithms will be able to handle the appropriate conversions.

ACKNOWLEDGMENTS

I'd like to thank the The Digital Manufacturing and Design Innovation Institute (Award No. 015336) for giving me the opportunity to do this research.

Also, I'd like to thank Dr. Winer for his unencumbered support and constant vigilance to push me in a positive direction. He is a busy man but always manages to find time for me for which I am deeply grateful. His advice throughout my PhD on both professional and personal matters has always steadied the ship. Further, I'd like to thank the committee for their invaluable guidance throughout my PhD.

I'd like to thank my parents for their constant support and always trying to comprehend my research while trying to understand why it took so long.

Finally, I'd like to thank Sreeja for always being by my side these past two years while I ranted and raved on matters ranging from insignificant to urgent.

REFERENCES

[1] J. D. Owens, “Streaming Architectures and Technology Trends,” **GPU Gems 2 Program. Tech. High-Performance Graph. Gen. Comput., pp. 457–470, 2005.**

[2] **M. Doi, J. Howell**, and S. Hirakawa, “Personal and home electronics and our changing lifestyles,” **Proc. IEEE, vol. 100, no. SPL CONTENT, pp. 1646–1656, 2012.**

[3] S. Nilsson and B. Johansson, “Acceptance of augmented reality instructions in a real work setting,” **Proc. ACM CHI 2008 Conf. Hum. Factors Comput. Syst., vol. 2, pp. 2025–2032, 2008.**

[4] D. D. Sumadio and D. R. a. Rambli, “Preliminary Evaluation on User Acceptance of the Augmented Reality Use for Education,” **Comput. Eng. Appl. (ICCEA), 2010 Second Int. Conf., vol. 2, pp. 461–465, 2010.**

[5] N. A. M. El Sayed, H. H. Zayed, and M. I. Sharawy, “ARSC: Augmented reality student card,” **Comput. Educ., vol. 56, no. 4, pp. 1045–1061, May 2011.**

[6] R. Wojciechowski and W. Cellary, “Evaluation of learners’ attitude toward learning in ARIES augmented reality environments,” **Comput. Educ., vol. 68, pp. 570–585, Oct. 2013.**

[7] “Juniper Research.” [Online]. Available: http://www.juniperresearch.com/press-**release/augmented-reality-pr1.**

[8] **R. T. Azuma, Y. Baillot, R. Behringer, S. K. Feiner, S. Julier, B. MacIntyre, Y.** Baillot, R. Behringer, S. K. Feiner, S. Julier, B. MacIntyre, and Y. Baillot, “Recent

Advances in Augmented Reality," **IEEE Comput. Graph. Appl., vol. 21, no. December, pp. 34–47, 2001.**

[9] **T. Ha, W. Woo, Y. Lee, J. Lee, J**. Ryu, H. Choi, and K. Lee, "ARtalet: Tangible **User Interface Based Immersive Augmented Reality Authoring Tool for Digilog** Book," **2010 Int. Symp. Ubiquitous Virtual Real., pp. 40–43, Jul. 2010.**

[10] **E. Ginters and J. Martin**-Gutierrez, "Low Cost Augmented R**eality and RFID** Application for Logistics Items Visualization," **Procedia Comput. Sci., vol. 26, pp. 3–13, Jan. 2013.**

[11] D. Grüntjens, S. Groß, D. Arndt, and S. Müller, "Fast Authoring for Mobile Gamebased City Tours," **Procedia Comput. Sci., vol. 25, pp. 41–51, Jan. 2013.**

[12] **T. Starner, S. Mann, B. Rhodes, J. Levine, J. Healey, D. Kirsch, R. Picard, and A.** Pentland, "Augmented Reality Through Wearable Computing."

[13] "Yelp Monocle Article." [Online]. Available: http://mashable.com/2009/08/27/yelp-**augmented-reality.**

[14] A. C. Addison, "Emerging Trends in Virtual Heritage," pp. 22–**25, 2000.**

[15] D. Schattel, T. Marcus, G. Klinker, G. Schubert, and F. Petzold, "[Demo] On-**Site** Augmented Collaborative Architecture Visualization," pp. 369–**370, 2014.**

[16] **M. Fjeld, J. Fredriksson, M. Ejdestig, F. Duca, K. Bötschi, B. Voegtli, and P.** Juchli, "Tangible User Interface for Chemistry Education : Comparative Evaluation **and Re**-Design," pp. 805–**808, 2007.**

[17] H. Kaufmann and D. Schmalstieg, "Mathematics and geometry education with

collaborative augmented reality," **Comput. Graph., vol. 27, no. 3, pp. 339–345, Jun. 2003.**

[18] **H.-K. K. Wu, S. W.-Y. Y. Lee, H.-Y. Y. Chang, and J.**-C. C. Liang, "Current status, opportunities and challenges of augmented reality in education," **Comput. Educ., vol. 62, pp. 41–49, Mar. 2013.**

[19] T. Blum, V. Kleeberger, C. Bichlmeier, and N. Navab, "Mirracle: An augmented reality magic mirror system for anatomy education," **Proc. - IEEE Virtual Real., pp. 115–116, 2012.**

[20] **F. Volonté, F. Pugin, P. Bucher, M. Sugimoto, O. Ratib, and P. Morel,** "Augmented reality and image overlay navigation with OsiriX in laparoscopic and robotic surgery: Not only a matter of fashion," **J. Hepatobiliary. Pancreat. Sci., vol. 18, no. 4, pp. 506–509, 2011.**

[21] **H. Regenbrecht, G. Baratoff, a**nd W. Wilke, "Augmented reality projects in the automotive and aerospace industries," **IEEE Comput. Graph. Appl., vol. 25, no. 6, pp. 48–56, 2005.**

[22] A. Y. C. Y. C. Nee, S. K. K. Ong, G. Chryssolouris, and D. Mourtzis, "Augmented **reality applications in d**esign and manufacturing," **CIRP Ann. - Manuf. Technol., vol. 61, no. 2, pp. 657–679, Jan. 2012.**

[23] G. J. M. C. R. R. Fründ J., "Using augmented reality technology to support the automobile development," **Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 3168 LNCS, pp. 289–298, 2005.**

[24] M. Fiorentino, R. de Amicis, G. Monno, and A. Stork, “Spacedesign: a mixed reality workspace for aesthetic industrial design,” **Proceedings. Int. Symp. Mix. Augment. Real., pp. 86–318, 2002.**

[25] J. Zhu, S. K. Ong, and a. Y. C. Nee, “An authorable context-**aware augmented** reality system to assist the maintenance technicians,” **Int. J. Adv. Manuf. Technol., pp. 1699–1714, Aug. 2012.**

[26] **R. Radkowski, J. Herrema, and J. Oli**ver, “Augmented Reality-**Based Manual** Assembly Support With Visual Features for Different Degrees of Difficulty,” **Int. J. Hum. Comput. Interact., vol. 31, no. 5, pp. 337–349, May 2015.**

[27] M. Dias, J. Jorge, J. Carvalho, P. Santos, and J. Luzio, “Usability **evaluation of** tangible user interfaces for augmented reality,” **2003 IEEE Int. Augment. Real. Toolkit Work., pp. 54–61, Oct. 2003.**

[28] **D. Markov**-Vetter and O. Staadt, “A pilot study for Augmented Reality supported **procedure guidance to operate payload racks on-board the International Space** Station,” in **Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on, 2013, pp. 1–6.**

[29] G. Schall, S. Zollmann, and G. Reitmayr, “Smart Vidente: advances in mobile **augmented reality for interactive vi**sualization of underground infrastructure,” **Pers. Ubiquitous Comput., vol. 17, no. 7, pp. 1533–1549, Sep. 2012.**

[30] a. C. Boud, D. J. Haniff, C. Baber, and S. J. Steiner, “Virtual reality and augmented reality as a training tool for assembly tasks,” **1999 IEEE Int. Conf. Inf. Vis. (Cat. No. PR00210), pp. 32–36, 1999.**

[31] S. J. Henderson and S. K. Feiner, “Augmented reality in the psychomotor phase of a procedural task,” in **2011 10th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2011, 2011, pp. 191–200.**

[32] “Maya.” [Online]. Available: www.autodesk.com/products/maya.

[33] “Studio Max.” [Online]. Available: www.autodesk.com/products/3ds-**max.**

[34] “Blender.” [Online]. Available: www.blender.org/.

[35] **D. G. Ullman, The Mechanical Design Process, vol. 3. ed. 2003.**

[36] **G. D. S. Moura, S. a. Pessoa, J. P. S. D. M. Lima, V. Teichrieb, and J. Kelner,** “RPR-SORS: An authoring toolkit for photorealistic AR,” in **Proceedings - 2011 13th Symposium on Virtual Reality, SVR 2011, 2011, pp. 178–187.**

[37] D. Wagner and D. Schmalstieg, “ARToolKitPlus for Pose Tracking on Mobile Devices ARToolKit,” 2007.

[38] M. Haller, W. Hartmann, T. Luckeneder, and J. Zauner, “Combining ARToolKit with scene graph libraries,” in **Augmented Reality Toolkit, The First IEEE International Workshop, 2002, p. 2 pp.-pp.**

[39] **S. A. Pessoa, G. de S. Moura, J. P. S. do M. Lima, V. Teichrieb, and J. Kelner,** “RPR-**SORS: Real-time photorealistic rendering of synthetic objects into real** scenes,” **Comput. Graph., vol. 36, no. 2, pp. 50–69, Apr. 2012.**

[40] **M. E. C. M. E. C. M. E. C. Santos, A. Chen, T. Taketomi, G. Yamamoto, J.** Miyazaki, and H. Kato, “Augmented reality learning experiences: Survey of prototype design and evaluation,” **IEEE Trans. Learn. Technol., vol. 7, no. 1, pp.**

38–56, Jan. 2014.

[41] L. Ştefan, “Immersive Collaborative Environments for Teaching and Learning Traditional Design,” **Procedia - Soc. Behav. Sci., vol. 51, pp. 1056–1060, 2012.**

[42] J. Chow, R. Amor, and C. W. Burkhard, “Music Education using Augmented **Reality with a Head** Mounted Display,” vol. 139, no. February, 2013.

[43] **K.-E. Chang, C.-T. Chang, H.-T. Hou, Y.-T. Sung, H.-L. Chao, and C.-M. Lee,** “Development and behavioral pattern analysis of a mobile guide system with **augmented reality for painting appreciation instruct**ion in an art museum,” **Comput. Educ., vol. 71, pp. 185–197, Feb. 2014.**

[44] V. Geroimenko, “Augmented reality technology and art: The analysis and visualization of evolving conceptual models,” in **Proceedings of the International Conference on Information Visualisation, 2012, pp. 445–453.**

[45] J. Zhang, Y. T. Sung, H. T. Hou, and K. E. Chang, “The development and **evaluation of an augmented reality-based armillary sphere for astronomical** observation instruction,” **Comput. Educ., vol. 73, pp. 178–188, Apr. 2014.**

[46] W. Chinthammit and A. Thomas, “IFiction: Mobile technology, new media, Mixed Reality and literary creativity in English teaching,” in **11th IEEE International Symposium on Mixed and Augmented Reality 2012 - Arts, Media, and Humanities Papers, ISMAR-AMH 2012, 2012, pp. 39–46.**

[47] E. Klopfer and J. Sheldon, “Augmenting your own reality: student authoring of **science**-based augmented reality games.,” **New Dir. Youth Dev., vol. 2010, no.**

128, pp. 85–94, 2010.

[48] L. Farias, R. Dantas, and A. Burlamaqui, "E**duc-AR: A tool for assist the creation** of augmented reality content for education," in **Virtual Environments Human-Computer Interfaces and Measurement Systems (VECIMS), 2011 IEEE International Conference on, 2011, pp. 1–5.**

[49] C. Kirner and T. G. Kirner, "**Development of an Educational Spatial Game using** an Augmented Reality Authoring Tool," **Int. J. Comput. Inf. Syst. Ind. Manag. Appl., vol. 3, pp. 602–611, 2011.**

[50] H. Jee, S. Lim, J. Youn, and J. Lee, "An Immersive Authoring Tool for Augmented **Reality-Based E**-Learning Applications," in **Information Science and Applications (ICISA), 2011 International Conference on, 2011, pp. 1–5.**

[51] A. M. M. Rahman, J. Cha, and A. E. Saddik, "Authoring edutainment content **through video annotations and 3D model augmentatio**n," in **Virtual Environments, Human-***Computer Interfaces and Measurements Systems, 2009. VECIMS '09.* **IEEE International Conference on, 2009, pp. 370–374.**

[52] **E. S. De Lima, B. Feijó, S. D. J. Barbosa, A. L. Furtado, A. E. M. Ciarlini, and C.** T. Pozzer, "Draw your own story: Paper and pencil interactive storytelling," **Entertain. Comput., vol. 5, no. 1, pp. 33–41, Jan. 2014.**

[53] M. Wang, C. Tseng, and C. Shen, "An Easy to Use Augmented Reality Authoring Tool for Use in Examination Purpose," **Human-Computer Interaction, no. 7, pp. 285–288, 2010.**

[54] O. Ploder, A. Wagner, G. Enislidis, and R. Ewers, “[Computer-**assisted** intraoperative visualization of dental implants. Augmented reality in medicine],” **Radiologe, vol. 35, no. 9, pp. 569–572, 1995.**

[55] O. Wieben, “Virtual and Augmented Reality in Medicine,” **Minim. Invasive Med. Technol. Ser. Ser. Med. Phys. Biomed. Eng. ISBN 978-0-7503-0733-8. Taylor Fr. Ed. by John Webster, pp. 176-194, vol. 1, pp. 176–194, 2001.**

[56] **T. Kilgus, E. Heim, S. Haase, S. Prüfer, M. Müller, A. Seitel, M. Fangerau, T. Wiebe, J. Iszatt, H.**-P. Schlemmer, and others, “Mobile markerless augmented reality and its application in forensic medicine,” **Int. J. Comput. Assist. Radiol. Surg., pp. 1–14, 2014.**

[57] **A. Corneliu, D. Mihaela, P. Mircea-Do**rin, B. Crenguta, and G. Mircea, “Teeth **reduction dental preparation using virtual and augmented reality by Constanta** dental medicine students through the VirDenT system,” in **The International Conference Development, Energy, Environment, Economics , Puerto De La Cruz, Tenerife, Spain, 2011.**

[58] “Kinect.” [Online]. Available: https://www.microsoft.com/en-**us/kinectforwindows/.**

[59] **U.**-V. Albrecht, B. Häussermann, H. K. Matthies, and U. von Jan, “MOBILE **AUGMENTED REALITY APPS FOR TEACHING ETHICALLY SENSITIVE** TOPICS IN MEDICINE,” **Mob. Learn. 2012, p. 247, 2012.**

[60] **S. Lee, J. Lee, A. Lee, N. Park, S. Song, A. Seo, H. Lee, J.-I. Kim, and K. Eom,** “Augmented reality intravenous injection simulator based 3D medical imaging for veterinary medicine,” **Vet. J., vol. 196, no. 2, pp. 197–202, 2013.**

[61] **D. J. Scott, J. C. Cendan, C. M. Pugh, R. M. Minter, G. L. Dunnington, and R. A.** Kozar, “The changing face of surgical education: simulation as the new paradigm.,” **J. Surg. Res., vol. 147, no. 2, pp. 189–93, Jun. 2008.**

[62] F. Anderson, T. Grossman, J. Matejka, and G. Fitzmaurice, “YouMove: enhancing movement training with an augmented reality mirror,” in **Proceedings of the 26th annual ACM symposium on User interface software and technology** - *UIST ’13*, **2013, pp. 311–320.**

[63] B. MacIntyre, J. D. D. Bolter, E. Moreno, and B. Hannigan, “Augmented reality as a new media experience,” in **Augmented Reality, 2001. Proceedings. IEEE and ACM International Symposium on, 2001, pp. 197–206.**

[64] **T. Lang, B. MacIntyre, and I. J**. Zugaza, “Massively Multiplayer Online Worlds as a Platform for Augmented Reality Experiences,” in **Virtual Reality Conference,** *2008. VR ’08. IEEE*, **2008, pp. 67–70.**

[65] Z. Huang, W. Li, P. Hui, and C. Peylo, “CloudRidAR : A Cloud-**based Architecture for Mo**bile Augmented Reality,” pp. 29–**34, 2014.**

[66] Piekarski, Wayne, and B. H. Thomas, “ARQuake: the outdoor augmented reality gaming system.,” **Commun. ACM, vol. 45, no. 1, pp. 36–38, 2002.**

[67] R. Wetzel, R. Mccall, A. Braun, and W. Broll, “Guidelines for Des**igning** Augmented Reality Games,” pp. 173–**180, 2006.**

[68] **M. Villalta, I. Gajardo, M. Nussbaum, J. J. Andreu, A. Echeverría, and J. L. Plass,** “Design guidelines for Classroom Multiplayer Presential Games (CMPG),”

Comput. Educ., vol. 57, no. 3, pp. 2039–2053, Nov. 2011.

[69] “Ingress.” .

[70] “Drakerz.” [Online]. Available: www.drakerz.com.

[71] “PulzAR.” [Online]. Available: https://www.playstation.com/en-**us/games/pulzar-psvita/.**

[72] A. Marzo, “CollARt : a Tool for Creating 3D Photo Collages Using Mobile Augmented Reality,” pp. 585–**588, 2013.**

[73] “Vuforia.” [Online]. Available: https://www.qualcomm.com/products/vuforia.

[74] **R. Wojciechowski, K. Walczak, M. White, and W. Cellary,** “Building virtual and augmented reality museum exhibitions,” vol. 1, no. 212, pp. 135–**145, 2004.**

[75] M. Carrozzino and M. Bergamasco, “Beyond virtual museums: Experiencing immersive virtual reality in real museums,” **J. Cult. Herit., vol. 11, no. 4, pp. 452–458, Oct. 2010.**

[76] O. Bimber, L. M. Encarnação, and D. Schmalstieg, “The virtual showcase as a new platform for augmented reality digital storytelling,” in **Proceedings of the workshop on Virtual environments 2003 - *EGVE ’03*, 2003, pp. 87–95.**

[77] **C. M**. Merges and A. F. Marcos, “Digital Art: When Artistic and Cultural Muse Merges with Computer Technology,” **Comput. Graph. Appl. IEEE, vol. 27, no. 5, pp. 98–103, Sep. 2007.**

[78] “SportVision.” [Online]. Available: www.sportvision.com.

[79] P. McIlroy, “Hawk-Eye: Augmented reality in sports broadcasting and officiating,” **in Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on, 2008, pp. xiv–xiv.**

[80] S. White and S. Feiner, “SiteLenss: Situated Visualization Techniques for Urban Site Visit,” in **Proceedings of the 27th international conference on Human factors in computing systems - *CHI ’09*, 2009, pp. 1117–1120.**

[81] S. Dong and V. R. Kamat, “Robust mobile computing framework for visualization of simulated processes in augmented reality,” in **Proceedings - Winter Simulation Conference, 2010, pp. 3111–3122.**

[82] **G. Schall, E. Mendez, E. Kruijff, E. Veas, S. Junghanns, B. Reitinger, and D.** Schmalstieg, “Handheld Augmented Reality for underground infrastructure visualization,” **Pers. Ubiquitous Comput., vol. 13, no. 4, pp. 281–291, 2009.**

[83] K. Anagnostou and P. Vlamos, “Square AR: Using Augmented Reality for Urban Planning,” in **Games and Virtual Worlds for Serious Applications (VS-GAMES), 2011 Third International Conference on, 2011, pp. 128–131.**

[84] S. Julier, Y. Baillot, M. Lanzagorta, D. Brown, and L. Rosenblum, “BARS : Battlefield Augmented Reality System,” **NATO Symp. Inf. Process. Tech. Mil. Syst., no. Code 5580, pp. 1–7, 2000.**

[85] **R. Tache, H. A. Abeykoon, K. T. Karunanayaka, J. P. Kumarasinghe, G. Roth, O.** N. N. Fernando, and A. D. Cheok, “Command Center: Authoring tool to supervise augmented reality session,” in **Virtual Reality Short Papers and Posters (VRW), 2012 IEEE, 2012, pp. 99–100.**

[86] “Layar.” [Online]. Available: https://www.layar.com/.

[87] “Blippar.” [Online]. Available: https://blippar.com/.

[88] “Aurasma.” [Online]. Available: http://www.aurasma.com/.

[89] C. Knopfle, J. Weidenhausen, L. Chauvigne, and I. Stock, “Template based authoring for AR based service scenarios,” **IEEE Proceedings. VR 2005. Virtual Reality, 2005., pp. 237–240, Mar. 2005.**

[90] L. De Marchi, a. Ceruti, N. Testoni, a. Marzani, and a. Liverani, “Use of augmented reality in aircraft maintenance operations,” vol. 9064, p. 906412, 2014.

[91] **J. Gimeno, P. Mori**llo, J. M. Orduña, and M. Fernández, “A new AR authoring tool using depth maps for industrial procedures,” **Comput. Ind., vol. 64, no. 9, pp. 1263–1271, Dec. 2013.**

[92] B. Schwald, B. Schwald, B. DeLaval, and B. DeLaval, “An Augmented Reality **System for Tra**ining and Assistance to Maintenance in the Industrial Context,” in **The 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, 2003.**

[93] I. Barakonyi, T. Psik, and D. Schmalstieg, “Agents That Talk And Hit **Back:** Animated Agents in Augmented Reality,” **Third IEEE ACM Int. Symp. Mix. Augment. Real., no. Ismar, pp. 141–150, 2004.**

[94] M. Hincapié, A. Caponio, H. Rios, and E. González Mendívil, “An introduction to **Augmented Reality with applications in aeronautic**al maintenance,” in **International Conference on Transparent Optical Networks, 2011.**

[95] G. Jo, K. Oh, I. Ha, K. Lee, M. Hong, U. Neumann, and S. You, "A Unified Framework for Augmented Reality and Knowledge-Based Systems in Maintaining Aircraft," **Proc. Twenty-Sixth Annu. Conf. Innov. Appl. Artif. Intell., pp. 2990–2997, 2014.**

[96] **J. Serván, F. Mas, J. L. Menéndez, and J. Ríos,** "Using augmented reality in AIRBUS A400M shop floor assembly work instructions," vol. 633, no. 2012, pp. **633–640, 2012.**

[97] N. Gavish, T. Gutierrez, S. Webel, and J. Rodriguez, "Design Guidelines for **the Development of Virtual Reality and Augmented Reality Training Systems for** Maintenance and Assembly Tasks," in **The International Conference SKILLS, 2011, vol. 29, pp. 1–4.**

[98] **S. Webel, U. Bockholt, T. Engelke, N. Gavish, M. Olbrich, and C. Preusche,** "An augmented reality training platform for assembly and maintenance skills," **Rob. Auton. Syst., vol. 61, no. 4, pp. 398–403, Apr. 2013.**

[99] J. Zhang, S. K. Ong, and a. Y. C. Nee, "RFID-**assisted assembly guidance system in an augmented reality environmen**t," **Int. J. Prod. Res., vol. 49, no. 13, pp. 3919–3938, Jul. 2011.**

[100] M. L. Yuan, S. K. Ong, and A. Y. C. Nee, "Augmented reality for assembly guidance using a virtual interactive tool," **International Journal of Production Research. 2008.**

[101] **J. Plato**nov, H. Heibel, P. Meier, and B. Grollmann, "A mobile markerless AR system for maintenance and repair," in **Proceedings - ISMAR 2006: Fifth IEEE**

and ACM International Symposium on Mixed and Augmented Reality, 2007, pp. 105–108.

[102] C. Tomasi, "Good Features," **Image (Rochester, N.Y.), pp. 593–600, 1994.**

[103] A. I. Comport, E. Marchand, M. Pressigout, and F. Chaumette, "Real-**time** markerless tracking for augmented reality: The virtual visual servoing framework," **in IEEE Transactions on Visualization and Computer Graphics, 2006, vol. 12, no. 4, pp. 615–628.**

[104] Y. Gene, S. Riedel, F. Souvannavong, C. Akinlar, and N. Navab, "Marker-**less** Tracking for AR : A Learning-Based Approach," 2002.

[105] S. K. Ong and J. Zhu, "A novel maintenance system for equipment serviceability improvement," **CIRP Ann. - Manuf. Technol., vol. 62, no. 1, pp. 39–42, Jan. 2013.**

[106] J. T. Doswell, "Context-**Aware Mobile Augmented Reality Architecture for Lifelong** Learning," **Sixth IEEE Int. Conf. Adv. Learn. Technol., pp. 372–374, Jul. 2006.**

[107] **J. Zhu, S. K. K. Ong, & A. Y. C. Nee, and** a. Y. C. Y. C. Nee, "A context-**aware** augmented reality assisted maintenance system," **Int. J. Comput. Integr. Manuf., vol. 28, no. June, pp. 1–13, Feb. 2014.**

[108] J. Park, C. Kang, S. Oh, H. Lee, and W. Woo, "Context-**Aware Augmented Reality Authoring To**ol in Digital Ecosystem," **2010 Int. Symp. Ubiquitous Virtual Real., pp. 16–19, Jul. 2010.**

[109] J. Y. Lee, D. Seo, B. Y. Song, and R. Gadh, "Visual and tangible interactions with **physical and virtual objects using context**-aware RFID," **Expert Syst. Appl., vol.**

37, no. 5, pp. 3835–3845, May 2010.

[110] J.-M. Su and C.-F. Huang, “An easy-**to-use 3D visualization system for planning context**-aware applications in smart buildings,” **Comput. Stand. Interfaces, vol. 36, no. 2, pp. 312–326, Feb. 2014.**

[111] A. M. Demiris, V. Vlahakis, A. Makri, M. Papaioannou, and N. Ioannidis, “intGuide: **A platform for context-aware services featuring augmented-reality, based on the** outcome of European Research Projects,” **Signal Process. Image Commun., vol. 20, no. 9–10, pp. 927–946, Oct. 2005.**

[112] M. C. Rodriguez-Sanchez, J. Martinez-Romo, S. Borromeo, and J. A. Hernandez-Tamames, “GAT: Platform for automatic context-**aware mobile services for m-**tourism,” **Expert Syst. Appl., vol. 40, no. 10, pp. 4154–4163, Aug. 2013.**

[113] S. K. Ong and Z. B. Wang, “Augmented assembly technologies based on 3D **bare**-hand interaction,” **CIRP Ann. - Manuf. Technol., vol. 60, no. 1, pp. 1–4, Jan. 2011.**

[114] L. X. Ng, Z. B. Wang, S. K. Ong, and a. Y. C. Nee, “Integrated product design **and assembly planning** in an augmented reality environment,” **Assem. Autom., vol. 33, no. 4, pp. 345–359, 2013.**

[115] S. Henderson and S. Feiner, “Exploring the benefits of augmented reality documentation for maintenance and repair.,” **IEEE Trans. Vis. Comput. Graph., vol. 17, no. 10, pp. 1355–68, Oct. 2011.**

[116] A. Tang, C. Owen, F. Biocca, and W. Mou, “Comparative effectiveness of

augmented reality in object assembly," in **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2003, no. 5, pp. 73–80.**

[117] S. G. Hart and L. E. Staveland, "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research," **Adv. Psychol., vol. 52, no. C, pp. 139–183, 1988.**

[118] C. Diaz and G. Paramo, "Combining Virtual and Augmented Reality to Improve the Mechanical Assembly Training Process in Manufacturing."

[119] M. Fiorentino, A. E. Uva, M. Gattullo, S. Debernardis, and G. Monno, "Augmented reality on large screen for interactive maintenance instructions," **Comput. Ind., vol. 65, no. 2, pp. 270–278, Feb. 2014.**

[120] J. Weidenhausen, C. Knoepfle, and D. Stricker, "Lessons learned on the way to industrial augmented reality applications, a retrospective on ARVIKA," **Comput. Graph., vol. 27, no. 6, pp. 887–891, Dec. 2003.**

[121] L. Hou and X. Wang, "Using Augmented Reality to Cognitively Facilitate Product Assembly Process," no. January, 2010.

[122] L. Hou, X. Wang, L. Bernold, M. Asce, and P. E. D. Love, "Using Animated Augmented Reality to Cognitively Guide Assembly," **J. Comput. Civ. Eng., vol. 27, no. October, pp. 439–451, 2013.**

[123] D. B. Espíndola, L. Fumagalli, M. Garetti, C. E. Pereira, S. S. C. Botelho, and R. Ventura Henriques, "A model-based approach for data integration to improve maintenance management by mixed reality," **Comput. Ind., vol. 64, no. 4, pp.**

376–391, May 2013.

[124] J. Zauner, M. Haller, A. Brandl, and W. Hartman, “Authoring of a mixed reality assembly instructor for hierarchical structures,” **Second IEEE ACM Int. Symp. Mix. Augment. Reality, 2003. Proceedings., pp. 237–246, Oct. 2003.**

[125] M. M. Shah, H. Arshad, and R. Sulaiman, “Occlusion in Augmented Reality,” **8th Int. Conf. Inf. Sci. Digit. Content Technol. (ICIDT 2012), vol. 2, pp. 372–378, 2012.**

[126] J. F. WANG, C. ZENG, Y. Liu, and S. Q. LI, “Integrated Content Authoring for **Augm**ented Reality Based Product Manual Assembly Process Instruction,” in **43rd International Conference on Computers and Industrial Engineering, 2013, pp. 16–18.**

[127] S. Makris, G. Pintzos, L. Rentzos, and G. Chryssolouris, “Assembly support using **AR technology** based on automatic sequence generation,” **CIRP Ann. - Manuf. Technol., vol. 62, no. 1, pp. 9–12, Jan. 2013.**

[128] J. Wang, Y. Feng, C. Zeng, and S. Li, “An Augmented Reality Based System for **Remote Collaborative Maintenance Instruction of Complex Products**,” in **2104 IEEE International Conference on Automation Science and Engineering, 2014, pp. 1–6.**

[129] G. Westerfield, “Intelligent Augmented Reality Training for Assembly and Maintenance,” 2012.

[130] **H. Ramirez, E. G. Mendivil, P. R. Flores, and M. C. Gonz**alez, “Authoring **Software for Augmented Reality Applications for the Use of Maintenance and**

Training Process," **Procedia Comput. Sci., vol. 25, pp. 189–193, Jan. 2013.**

[131] "Unity." [Online]. Available: http://unity3d.com/unity.

[132] "Touch Designer." [On**line]. Available: https://www.derivative.ca.**

[133] R. Berry, N. Hikawa, M. Makino, M. Suzuki, and T. Furuya, "Authoring augmented **reality: a code**-free approach," in *SIGGRAPH '04: ACM SIGGRAPH 2004* **Posters, 2004, p. 43.**

[134] **R. Berry, M. Oikawa, J. Prasad, J. Unterberg, W. Liu, A. D. Cheok, and H. Kato,** "Augmented reality authoring for artists and designers," **ACM SIGGRAPH ASIA 2008 artgallery Emerg. Technol. -** *SIGGRAPH Asia '08***, p. 40, Dec. 2008.**

[135] B. Morten and F. Paulsen, "Online Education Systems : Discussion and Definition of Terms," pp. 1–**8, 2002.**

[136] **A. B. Markham, Knowledge Representation. 1999.**

[137] A. Hampshire, H. Seichter, R. Grasset, and M. Billinghurst, "Augmented reality **author**ing," in **Proceedings of the 20th conference of the computerhuman interaction special interest group CHISIG of Australia on Computerhuman interaction design activities artefacts and environments OZCHI 06, 2006, p. 409.**

[138] "OpenCV." [Online]. Available: h**ttp://opencv.org/.**

[139] "Open Scene Graph." [Online]. Available: **http://trac.openscenegraph.org/projects/osg/.**

[140] M. Gandy and B. MacIntyre, "Designer's augmented reality toolkit, ten years later," in **Proceedings of the 27th annual ACM symposium on User interface**

software and technology - *UIST '14*, **2014, pp. 627–636.**

[141] B. MacIntyre, M. Gandy, S. Dow, and J. D. Bolter, "DART: a toolkit for rapid design exploration of augmented reality experiences," **Proc. 17th Annu. ACM Symp. User interface Softw. Technol., vol. 6, no. 2, pp. 197–206, 2004.**

[142] B. G. A. Lee, G. J. Kim, and M. Billinghurst, "What You Experience Is What You Get," vol. 48, no. 7, 2005.

[143] C. Machinery, "Smartphone as an AR Authoring Tool via Multi-**Touch based 3D** Interaction Method," **vol. 1, no. 212, pp. 17–20, 2012.**

[144] Y. Baillot, D. Brown, and S. Julier, "Authoring of physical models using mobile computers," **Proc. Fifth Int. Symp. Wearable Comput., pp. 39–46, 2001.**

[145] X. Z. X. Zhang, S. Fronz, and N. Navab, "Visual marker detec**tion and decoding in** AR systems: a comparative study," **Proceedings. Int. Symp. Mix. Augment. Real., 2002.**

[146] P. Kakumanu, S. Makrogiannis, and N. Bourbakis, "A survey of skin-**color** modeling and detection methods," **Pattern Recognit., vol. 40, no. 3, pp. 1106–1122, Mar. 2007.**

[147] A. A. Argyros and M. I. A. Lourakis, "Real-**Time Tracking of Multiple Skin-Colored** Objects with a Possibly Moving Camera," in **ECCV, 2004, pp. 368–379.**

[148] J. Maccormickl and M. Isard, "Partitioned Sampling, Articulated Objects, **and Interface**-Quality Hand Tracking," pp. 3–**19, 2000.**

[149] B. Stenger, P. R. S. Mendonc, and R. Cipolla, "Model-**Based Hand Tracking**

Using an Unscented Kalman Filter," pp. 63–72.

[150] S. Sclaroff, "Automatic 2D Hand Tracking in Video Sequences 4 . Detecting Candidate Hand Loca- tions in a Single Frame."

[151] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Model-based hand tracking using a hierarchical Bayesian filter.," IEEE Trans. Pattern Anal. Mach. Intell., vol. 28, no. 9, pp. 1372–84, Sep. 2006.

[152] K. Mathias and M. Turk, "Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration," 2004.

[153] R. Radkowski and C. Stritzke, "Interactive Hand Gesture-based Assembly for Augmented Reality Applications," no. c, pp. 303–308, 2012.

[154] W. Hürst and C. Van Wezel, "Gesture-based interaction via finger tracking for mobile augmented reality," pp. 233–258, 2013.

[155] J. Shim, M. Kong, Y. Yang, J. Seo, and T.-D. Han, "Interactive features based augmented reality authoring tool," 2014 IEEE Int. Conf. Consum. Electron., pp. 47–50, Jan. 2014.

[156] T. Lee, H. Tobias, T. Hollerer, and H. Tobias, "Handy AR : Markerless Inspection of Augmented Reality Objects Using Fingertip Tracking," 2007 11th IEEE Int. Symp. Wearable Comput., vol. 2007, pp. 1–8, Oct. 2007.

[157] G. Reitmayr and T. Drummond, "Going out: robust model-based tracking for outdoor augmented reality," 2006 IEEE/ACM Int. Symp. Mix. Augment. Real., pp. 109–118, Oct. 2006.

[158] G. Klein and D. Murray, “Parallel Tracking and Mapping for Small AR Workspaces,” **2007 6th IEEE ACM Int. Symp. Mix. Augment. Real., pp. 1–10, Nov. 2007.**

[159] V. Lepetit, “Fully Automated and Stable Registration for Augmented Reality Applications,” 2006.

[160] V. Ferrari, T. Tuytelaars, and L. Van Gool, “Markerless Augmented Reality with a **Real-time Affine Region Tracker.**”

[161] T. Lee and H. Tobias, “Hybrid Feature Tracking and User Interaction for Markerless Augmented Reality,” pp. 145–**152.**

[162] I. Poupyrev, D. Tan, M. Billinghurst, H. Kato, H. Regenbrecht, N. Tetsutani, and D. Ag, “Tiles : A Mixed Reality Authoring Interface,” in **Conference on Human Computer Interaction, 2001.**

[163] M. Billinghurst, H. Kato, and I. Poupyrev, “The MagicBook - **Moving seamlessly** between reality and virtuality,” **IEEE Comput. Graph. Appl., vol. 21, pp. 6–8, 2001.**

[164] A. MacWilliams, C. Sandor, M. Wagner, M. Bauer, G. Klinker, and B. Bruegge, “Herding sheep: live system for distributed augmented reality,” **Second IEEE ACM Int. Symp. Mix. Augment. Reality, 2003. Proceedings., 2003.**

[165] Z. Szalavári and M. Gervautz, “The Personal Interaction **Panel - a Two-Handed** Interface for Augmented Reality,” **Comput. Graph. Forum, vol. 16, no. 3, pp. C335–C346, Jun. 2008.**

[166] B. H. Thomas and W. Piekarski, “Glove Based User Interaction Techniques for

Augmented Reality in an Outdoor Environment," **Virtual Real., vol. 6, no. 3, pp. 167–180, Oct. 2002.**

[167] J. Y. Lee, D. W. Seo, and G. W. Rhee, "Tangible authoring of 3D virtual scenes in dynamic augmented reality environment," **Comput. Ind., vol. 62, no. 1, pp. 107–119, Jan. 2011.**

[168] **S. Guven, S. Feiner, a**nd O. Oda, "Mobile augmented reality interaction **techniques for authoring situated media on**-site," **2006 IEEE/ACM Int. Symp. Mix. Augment. Real., pp. 235–236, Oct. 2006.**

[169] J. Barbadillo and J. R. Sánchez, "A Web3D authoring tool for augmented reality **mob**ile applications," **Proc. 18th Int. Conf. 3D Web Technol. -** *Web3D '13,* **p. 206, Jun. 2013.**

[170] A. Hill, B. MacIntyre, M. Gandy, B. Davidson, and H. Rouzati, "KHARMA: An open KML/HTML architecture for mobile augmented reality applications," **2010 IEEE Int. Symp. Mix. Augment. Real., pp. 233–234, Oct. 2010.**

[171] S. Lee, Y. Seo, and H. S. Yang, "Scalable Building Facade Recognition and Tracking for Outdoor Augmented Reality," pp. 923–**931.**

[172] M. Shin, B. Kim, and J. Park, "AR storyboard: an augmented reality **based** interactive storyboard authoring tool," **Fourth IEEE ACM Int. Symp. Mix. Augment. Real., pp. 198–199, Oct. 2005.**

[173] **N. Hagbi, R. Grasset, O. Bergig, M. Billinghurst, and J. El**-Sana, "In-**Place Sketching for content authoring in Augmented Reality ga**mes," **2010 IEEE Virtual**

Real. Conf., pp. 91–94, Mar. 2010.

[174] I. Radu, “Augmented-Reality Scratch : a Children ’ s Authoring Environment for **Augmented**-Reality Experiences,” pp. 3–**6, 2009.**

[175] M. Haringer and H. T. H. T. T. Regenbrecht, “A pragmatic ap**proach to augmented** reality authoring,” **Proceedings. Int. Symp. Mix. Augment. Real., no. c, pp. 237–245, Sep. 2002.**

[176] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. a Y. Silver, B. Silverman, and Y. Kafai, “Scratch: Programming for All.,” **Commun. ACM, vol. 52, pp. 60–67, 2009.**

[177] B. Thomas, B. Close, J. Donoghue, J. Squires, P. De Bondi, and W. Piekarski, “First person indoor/outdoor augmented reality application: ARQuake,” in **Personal and Ubiquitous Computing, 2002, vol. 6, no. 1, pp. 75–86.**

[178] B. Thomas, V. Demczuk, W. Piekarski, D. Hepworth, and B. Gunther, “A wearable computer system with augmented reality to supportterrestrial navigation,” **Dig. Pap. Second Int. Symp. Wearable Comput. (Cat. No.98EX215), 1998.**

[179] A. P. Gee, M. Webb, J. Escamilla-Ambrosio, W. Mayol-Cuevas, and A. Calway, “A topometric system for wide area augmented reality,” **Comput. Graph., vol. 35, no. 4, pp. 854–868, Aug. 2011.**

[180] “Solidworks.” [Online]. Available**: https://www.solidworks.com.**

[181] W. Hürst and J. Dekker, “Tracking-**based interaction for object creation in mobile** augmented reality,” in **Proceedings of the 21st ACM international conference on**

Multimedia - *MM '13*, **2013, pp. 93–102.**

[182] S. Mitra, S. Member, and T. Acharya, "Gesture Recognition : A Survey," vol. 37, **no. 3, pp. 311–324, 2007.**

[183] C. Platzer, M. Stuetz, and M. Lindorfer, "Skin Sheriff : A Machine Learning Solution for Detecting Explicit Images," *SFCS '14 Proc. 2nd Int. Work. Secur.* **forensics Commun. Syst., pp. 45–55, 2014.**

[184] H. Baltzakis, M. Pateraki, and P. Trahanias, "Visual tracking of hands, faces and facial features of multiple persons," **Mach. Vis. Appl., vol. 23, no. 6, pp. 1141–1157, Feb. 2012.**

[185] F. Dadgostar and A. Sarrafzadeh, "An adaptive real-**time skin detector based on** Hue thresholding: A comparison on two motion tracking methods," **Pattern Recognit. Lett., vol. 27, no. 12, pp. 1342–1352, 2006.**

[186] M. Jones and J. Rehg, "Statistical Color Models with Application to S**kin Detection** 2 Histogram Color Models," **Comput. Vis. Pattern Recognit., vol. 46, no. 1, pp. 1–23, 1999.**

[187] M. Shoyaib, M. Abdullah-Al-Wadud, and O. Chae, "A skin detection approach **based on the Dempster**–Shafer theory of evidence," **Int. J. Approx. Reason., vol. 53, no. 4, pp. 636–659, Jun. 2012.**

[188] S. J. D. Prince, "Computer vision : models , learning and inference," 2012.

[189] J. Ruiz-Del-Solar and R. Verschae, "Skin detection using neighborhood information," in **Proceedings - Sixth IEEE International Conference on Automatic**

Face and Gesture Recognition, 2004, pp. 463–468.

[190] D. Reynolds, “Gaussian Mixture Models,” **Encycl. Biometrics, no. 2, pp. 659–663, 2009.**

[191] Sangho Park and J. K. Aggarwal, “Segmentation and tracking of interacting human body parts under occlusion and shadowing,” in **Workshop on Motion and Video Computing, 2002. Proceedings., 2002, pp. 105–111.**

[192] S. Borman, “The Expectation Maximization Algorithm A short tutorial,” **Submitt. Publ., vol. 25, no. x, pp. 1–9, 2009.**

[193] R. Tibshirani, G. Walther, and T. Hastie, “Estima**ting the number of clusters in a** data set via the gap statistic,” **Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 63. pp. 411–423, 2001.**

[194] C. Sugar and J. Gareth, “Finding the number of clusters in a data set : An **in**formation theoretic approach,” **J. Am. Stat. Assoc., vol. 98, pp. 750–763, 2003.**

[195] L. Wang, C. Leckie, K. Ramamohanarao, and J. Bezdek, “Automatically determining the number of clusters in unlabeled data sets,” **IEEE Trans. Knowl. Data Eng., vol. 21, no. 3, pp. 335–350, 2009.**

[196] G. W. Milligan and M. C. Cooper, “An examination of procedures for determining the number of clusters in a data set,” **Psychometrika, vol. 50, no. 2, pp. 159–179, 1985.**

[197] A. Weingessel, E. Dimitriadou, and K. Hornik, “An exam**ination of indexes for** determining the number of clusters in binary data sets,” **Psychometrika, vol. 67,**

no. 1, pp. 137–159, 2003.

[198] J. Kennedy and R. Eberhart, “Particle Swarm Optimization,” **Eng. Technol., pp. 1942–1948, 1995.**

[199] **D. Zan and J. Jaros**, “Solving the Multidimensional Knapsack Problem using a CUDA accelerated PSO,” **Evol. Comput. (CEC), 2014 IEEE Congr., no. 2, pp. 2933–2939, 2014.**

[200] L. De and R. a. Krohling, “Swarm’s flight: Accelerating the particles using C-CUDA,” **2009 IEEE Congr. Evol. Comput. CEC 2009, pp. 3264–3270, 2009.**

[201] L. Mussi, S. Ivekovic, and S. Cagnoni, “Markerless articulated human body **tracking from multi-view video with GPU**-PSO,” **Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 6274 LNCS, pp. 97–108, 2010.**

[202] **D. L. Souza, G. D. Monteiro, T. C. Martins, V. A. Dmitriev, Ot, #225, and V. N.** Teixeira, “PSO-**GPU: accelerating particle swarm optimization in CUDA-based** graphics processing units,” **Proc. 13th Annu. Conf. companion Genet. Evol. Comput., pp. 837–838, 2011.**

[203] L. Mussi, Y. S. G. Nashed, and S. Cagnoni, “GPU-**based asynchronous particle** swarm optimization,” **Proc. 13th Annu. Conf. Genet. Evol. Comput.** - *GECCO ’11*, **p. 1555, 2011.**

[204] **Y. Zhou and Y. Tan,** “GPU-based parallel particle swarm optimization,” in **2009 IEEE Congress on Evolutionary Computation, 2009, no. 1, pp. 1493–1500.**

[205] S. Singh, J. Kaur, and R. S. Sinha, “A Comprehensive Survey on Various Evolutionary Algorithms on GPU,” **Icccs, 2014.**

[206] R. C. Eberhart and Y. Shi, “Particle swarm optimization: developments, applications and resources,” **Proc. 2001 Congr. Evol. Comput., vol. 1, pp. 81–86, 2001.**

[207] “Point Cloud Library.” [Online]. Available: http://pointclouds.org/.

[208] R. Bhatt and A. Dhall, “Skin Segmentation Dataset, UCI Machine Learning Repository.” [Online]. Available: **https://archive.ics.uci.edu/ml/datasets/Skin+Segmentation.**

[209] P. Jonathon Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, “The FERET **evaluation methodology for face**-recognition algorithms,” **IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 10, pp. 1090–1104, 2000.**

[210] “Startup investment statistics.” [Online]. Available: http://zugara.com/a-**list-augmented-reality-and-venture-capital-funding.**

[211] R. Azuma and R. Azuma, “A survey of augmented reality,” **Presence Teleoperators Virtual Environ., vol. 6, no. 4, pp. 355–385, 1997.**

[212] S. K. Ong, M. L. Yuan, and A. Y. C. Nee, “Augmented reality applications in manufacturing: a survey,” **Int. J. Prod. Res., vol. 46, no. 10, pp. 2707–2742, 2008.**

[213] X. Wang, S. K. Ong, and A. Y. C. Nee, “A comprehensive survey of augmented reality assembly research,” **Adv. Manuf., vol. 4, no. 1, pp. 1–22, Mar. 2016.**

[214] D. Hawkes, “Virtual Reality and Augmented Reality in Medicine,” **Percept. Vis.**

Inf., pp. 361–390, 1997.

[215] A. Dünser, R. Grasset, M. Billinghurst, H. Interface, and N. Zealand, “A survey of evaluation techniques used in augmented reality studies,” in **ACM SIGGRAPH ASIA 2008 courses, 2008, no. September, p. 27.**

[216] S. Benbelkacem, M. Belhocine, A. Bellarbi, N. Zenati-Henda, and M. Tadjine, “Augmented reality for photovoltaic pumping systems maintenance tasks,” **Renew. Energy, vol. 55, pp. 428–437, Jul. 2013.**

[217] S. J. Henderson and S. K. Feiner, “Augmented Reality for Maintenance and Repair (ARMAR),” **Distribution, 2007. [Online]. Available: http://graphics.cs.columbia.edu/projects/armar/.**

[218] M. Andersen, R. Andersen, C. Larsen, T. B. Moeslund, and O. Madsen, “Interactive Assembly Guide Using Augmented Reality,” pp. 999–**1008, 2009.**

[219] X. Wang, S. K. Ong, and A. Y. C. Nee, “Real-**virtual interaction in AR assembly** simulation based on component contact handling strategy,” **Assem. Autom., vol. 35, no. 4, pp. 376–394, 2015.**

[220] W. Xin, “AN INTE**GRATED AUGMENTED REALITY METHOD TO ASSEMBLY** SIMULATION AND GUIDANCE,” 2016.

[221] D. W. and D. Schmalstieg, “ARToolKitPlus for Pose Tracking on Mobile Devices,” **in** *12th Computer Vision Winter Workshop (CVWW’07)***.**

[222] A. Mossel and B. Venditti, “3DTouch an**d HOMER**-S : Intuitive Manipulation **Techniques for One**-Handed Handheld Augmented Reality,” 2013.

[223] L. Salgado, N. O. Connor, M. Tsapatori, and J. A. Soler, “The ORION Project A European Union Thematic Network,” pp. 103–**112, 2005.**

[224] F. Gîrbacia, S. Butnariu, and A. P. Orman, “VIRTUAL RESTORATION OF **DETERIORATED RELIGIOUS HERITAGE OBJECTS USING AUGMENTED** REALITY TECHNOLOGIES,” vol. 9, no. 2, pp. 223–**231, 2013.**

[225] A. D. Styliadis, I. I. Akbaylar, D. A. Papadopoulou, N. D. Hasanagas, S. A. Roussa, and L. A. Sexidis, “Metadata-**based heritage sites modeling with e-**learning functionality,” **J. Cult. Herit., vol. 10, no. 2, pp. 296–312, Apr. 2009.**

[226] Z. Noh, M. S. Sunar, and Z. Pan, “A review on augmented reality for virtual heritage system,” in **Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2009, vol. 5670 LNCS, pp. 50–61.**

[227] J. Grubert, T. Langlotz, and R. Grasset, “Augmented Reality Browser Survey,” **2011.**

[228] T. Olsson and M. Salo, “Online user survey on current mobile augmented reality applications,” **2011 10th IEEE Int. Symp. Mix. Augment. Real., no. July, pp. 75–84, 2011.**

[229] D. W. F. V. W. F. V. D. Van Krevelen, R. Poelman, D. W. F. Van Krevelen, R. Poelman, D. W. F. V. W. F. V. D. Van Krevelen, I. Rabbi, and S. Ullah, “A survey of Augmented Reality Technologies, Applications and Limitations,” **Int. J. Virtual Real., vol. 9, no. 2, pp. 1–20, 2010.**

[230] I. Rabbi and S. Ullah, “A Survey on Augmented Reality Challenges and Tracking,” **Acta Graph., vol. 24, no. 1–2, pp. 29–46, 2013.**

[231] M. Billinghurst, A. Clark, and G. Lee, “A Survey of Augmented Reality Augmented Reality (AR) Defini3on,” **Found. Trends Human-Computer Interact., vol. 8, no. 2–3, pp. 73–272, 2015.**

[232] **P. Grimm, M. Haller, V. Paelke, S. Reinhold, C. Reimann, R. Zauner, and P.** Grimin, “AMIRE - authoring mixed reality,” in **Augmented Reality Toolkit, The First IEEE International Workshop, 2002, p. 2 pp.-pp.**

[233] **D. Schmalstieg, A. Fuhrmann, G. Hesina, Z. Szalavári, L. M. Encarnação, M.** Gervautz, and W. Purgathofer, “The Studierstube Augmented Reality Project,” **Presence Teleoperators Virtual Environ., vol. 11, no. 1, pp. 33–54, 2002.**

[234] M. Haller, A. Brandl, and W. Hartmann, “Authoring of a Mixed **Reality Assembly** Instructor for Hierarchical Structures,” 2006.

[235] G. A. A. G. A. Lee, Ļ. C. Nelles, G. Jounghyun, C. Nelles, M. Billinghurst, J. Kim, Ļ. **C. Nelles, G. Jounghyun, C. Nelles, M. Billinghurst, J. Kim, and G. J. Kim,** “Immersive Authoring of Tangible Augmented Reality Applications,” in **Mixed and Augmented Reality, 2004. ISMAR 2004. Third IEEE and ACM International Symposium on, 2004, no. Ismar, pp. 172–181.**

[236] J. Park, S. Sil Kim, H. Park, and W. Woo, “‘DreamHouse’: NUI-**based Photo-realis**tic AR Authoring System for Interior Design,” in **Proceedings of the 7th Augmented Human International Conference 2016 on** - *AH ’16*, **2016, pp. 1–7.**

[237] M. Anisetti, V. Bellandi, P. Ceravolo, and E. Damiani, “Intelligent Interactive **Multimedia Systems and S**ervices,” **Smart Innov. Syst. Technol., vol. 6, pp. 251–260, 2010.**

[238] D. Markouzis and G. Fessakis, “Interactive Storytelling and Mobile Augmented **Reality applications for Learning and Entertainment - A rapid prototyping** perspective,” **Interact. Mob. Commun. Technol. Learn. (IMCL), 2015 Int. Conf., no. November, pp. 4–8, 2015.**

[239] **Y. Yang, J. Shim, S. Chae, and T.-**D. Han, “Mobile Augmented Reality Authoring Tool,” **2016 IEEE Tenth Int. Conf. Semant. Comput., pp. 358–361, 2016.**

[240] **S. Zhang, B. Li, and Y**. Xu, “Superfast 3D shape measurement with binary dithering techniques,” in **Recent advances in topography research, 2013, pp. 43–60.**

[241] T. Ha and W. Woo, “Graphical Tangible User Interface for a AR Authoring Tool in Product Design Environment,” **Design, pp. 3–4, 2007.**

[242] J. Yu, J. Jeon, G. Park, H. Kim, and W. Woo, “A Unified Framework for Remote Collaboration Using Interactive AR Authoring and Hands Tracking,” vol. 9189, N. **Streitz and P. Markopoulos, Eds. Cham: Springer International Publishing, 2016, pp. 132–141.**

[243] B. Bhattacharya and E. Winer, “A method for real-**time generation of augmented** reality work instructions via expert movements,” in **The Engineering Reality of Virtual Reality 2015, 2015, p. 93920G.**

[244] “Turbosquid.”

[245] “Cloud Compare.”

[246] “Makerbot.” .