Ibero-American Symposium on Computer Graphics - SIACG (2006)
P. Brunet, N. Correia, and G. Baranoski (Editors)

Kit Assembly Helper in Augmented Reality

M. V. Almeida[1] and J. M. S. Dias[2,3]

[1]*Teleca AB, Av. D. João II, Lote 1.022.2A, 1º Esq., Parque das Nações, 1990-095 Lisboa, Portugal*

[2]*MLDC, Microsoft Language Development Center, Lisbon, Av. Prof. Doutor Aníbal Cavaco Silva, Edifício Qualidade C1, C2, TagusPark, 2744-010 PORTO SALVO, Portugal,* `http://www.microsoft.com`

[3]*ADETTI/ISCTE, Edifício ISCTE, Av. das Forças Armadas, 1600-082 Lisbon, Portugal, http://www.adetti.pt*

Abstract

This paper describes an augmented reality application that aids users in assembling an object originally in a kit format, i.e. the object is received unassembled, with ordered instructions steps on how to connect each piece. We have used XML to define the objects, the object pieces and the assembly rules; thus, the Kit Assembly Helper (KAH) becomes flexible and can assemble any given 3D physical object. We have adopted a hierarchical structure similar to that used in a theatre, based on puppets and puppeteers. By using augmented reality and pre-recorded speech prompts, the cognitive distance between the user and the assembly instructions is reduced, allowing a better perception of the problem, as well as of the solution. The system is developed in MX Toolkit, an in-house C++ software platform oriented to the Augmented Reality/Mixed Reality application developer. The paper discusses system architecture aspects and provides a usability evaluation framework that shows the benefits of the approach for helping users in kit assembly tasks.

1. Introduction

The need to produce simple to use furniture assembly instructions is increasing with the trend of furniture kits [IKEA]. Instead of buying already assembled furniture, consumers are buying assembly kits which fit their cars and they can take easily home, where they have to start a sometimes long task of putting it all together. Because ordinary men and women aren't expert carpenters or skilled craftsman (although many think they are…), furniture assembly instructions have to be simple, straightforward and easy to follow. In order to solve this problem major furniture retailers [IKEA] have traditional means to help their clients, like hotlines, internet pages and assembly specialists who can go to client's home and take care of business. These traditional approaches cost time and money, both for companies and for consumers. The easier to follow and self explanatory the rules are, the easiest it is to assemble the object. Some furniture pieces are shipped with videos showing the assembly of the physical object; the client can watch the video and replicate the behaviour. But there is still a gap that is not filled: the distance between the object on screen and the real object is still present. With Augmented Reality - AR, it is possible to close the gap between the assembly instructions and the object, by registering a virtual representation of each assembly step, over already assembled pieces and providing 3D visual cues about the current assembly step. This can be done by superimposing virtual furniture pieces over the real physical pieces, demonstrating clearly where the pieces will fit, what type of tools are needed and how and where the consumer has to use them.

1.1 Requirements of a Kit Assembly Helper in AR

Assembly instructions are often a problem to the end user; sometimes they are unclear, misunderstood or both. Many people have to contact help centres to finish the assembly job, while others actually build internet pages complaining about the assembly instructions that came with the package. The authors think that the major problem resides in the fact that some people can't close the gap between current problem solving maps, which use a 2D media (paper assembly instructions) and real life 3D objects. By using Augmented Reality (AR), we can link each assembly step with real world objects, shortening the cognitive distance: the end user does not need to switch from the 2D representation to the 3D representation of an object; the AR assembly rule does that for him/her.

Real furniture objects are composed by different type of pieces which require different types of interactions depending on the specific assembly instructions. Some pieces need to be combined with other pieces, some need only to be positioned, and others require specific tools, like hammers and nails, or screwdrivers and bolts, to be assembled. If we want to provide computational assistance to simplify this task, there is the need to design specific natural interaction paradigms in AR, that are adapted to the task at hand, in order to build easy and comprehensible instructions for the user to follow. This problem forces the construction of assembly instructions that are flexible and support the description of all the needed interactions for all needed pieces.

Typical assembly instructions come in leaflets with the kit object; they describe the several steps required to convert all the separate pieces in the final, and desired, object. Usually one cannot begin a new assembly step before all previous steps are completed. While designing our system, this gives us the need to provide a state-machine type of control for the assembly steps action and time-based evolution.

To create a richer user experience, and provide an analogy to the local presence and aid of an assembly expert, the authors are convinced that there is the need to use and register virtual helpers in AR, of humanoid or other representation. The humanoid helpers should demonstrate some parts or the entire required task, while other virtual helpers representations, should give help with the essential technical information about the assembly step.

It's important to set-up some sort of hierarchical structure to control pieces, virtual helpers and assembly instructions. In this respect, we think that the approach taken in [BPS04], which is based in puppet theatre storytelling is a valuable contribution to this problem. So we came out with the concepts of using Puppets to represent the furniture pieces and objects, and Puppeteers to control them. In an upper systemic level, we need to have something that interacts as a Choreographer / Director of Assembly Instructions, which chooses what is currently happening and what is going to happen next.

2. Related work

Augmented Reality techniques applied to Kit Assembly scenarios have raised the interest of the scientific community.

In [ZHB*03], the authors propose a Mixed Reality based step-by-step furniture assembly application. Usage of a mixture of traditional 2D and 3D technology leads the user through the assembly process. Objects and assembly instructions are described through XML representations which allow user defined objects and assembly instructions. This solution is based on marker detection, which makes small parts and round elements hard to use; this is solved by relating this with other flat surfaces with assigned visual markers.

In [BPS04], we find a AR helper system to aid the user in assembling LEGO® constructions. To verify the assembly result, tests need to be conducted; using a LEGO® Mindstorm robot, this system is able to receive commands and to respond mechanically; this is made possible by using sensors which inform the system state. An interesting hierarchical framework is used, based on theatre stage playing, that invokes the Director, Choreographer, Puppeteer and Puppet metaphors; this enables the separation between assembly instructions, the object to be assembled and the assembly visual helpers, allowing the construction of flexible helper systems.

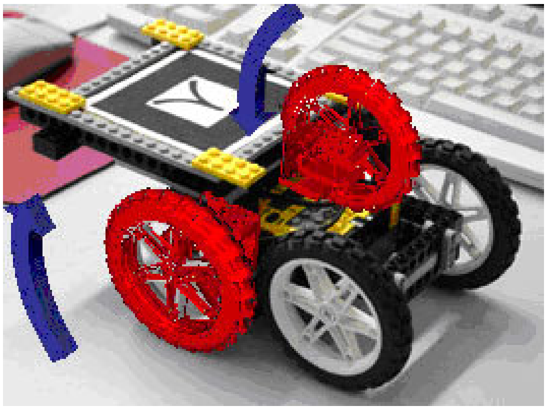

Figure 1: *Agents That Talk And Hit Back – Augmented LEGO® Mindstorm robot [BPS04]*

Evaluating the usability of KAH is important in order to understand its relevance. We consider [DJC*03], where is suggested a usability evaluation methodology for an authoring tool in mixed reality environments. Users are first introduced to the system, and have to perform pre-defined tasks both in an augmented reality system and in a conventional system. Questionnaires are used to define user profile and background, and to determine user opinion about the system.

3. Kit assembly concepts and requirements

3.1 Objects and pieces

For assembly purposes, we consider an object as a group of physical pieces. Each object is defined by its own components that, when assembled in the correct way, represent the final object. For each physical piece there is an assigned visual representation, consisting of a virtual 3D object, which allows visual user recognition of the correct piece to use in each assembly step.

As assembly steps are followed, the pieces start to fit together and to form a semi-assembled object; a semi-assembled object is an intermediary representation of an object that has not yet been fully assembled.

3.2 Assembly instructions and assembly steps

The goal of KAH is to assemble an object using the correct pieces, in the correct timely order and in the correct way. Therefor, assembly instructions should be a set of rules that show the user how to correctly assemble a given object.

Figure 2: *Pieces, semi-assembled object and object*

Assembly instructions are separated in several assembly steps that show the user how to assemble given pieces. Each assembly step is only responsible for a small group of similar pieces (usually only one piece); this top-down methodology simplifies the assembly procedure, reducing the solution to specific problems that lead the user to focus only on simple tasks.

The assembly steps are executed in step-by-step fashion, and are ruled by a state-machine that controls which is the current assembly step and when it is time to move on to the next one.

Each assembly step defines what piece has to be assembled and the correct way to do so; it has information about the correct location and orientation for placing the piece in the semi-assembled object, allowing a correct display of visual information to the user. This correct location and orientation of a given assembly piece is defined by spatial relations, using visual markers.

3.3 Visual markers

As the assembly steps are being completed, the user has to physically interact with the semi-assembled object; therefore its physical location may change. The system needs to perceive the location of the semi-assembled object in order to place the needed assembly piece in the correct position of the world. Thus some sort of object tracking technology needs to be adopted. In this paper we have selected marker-based tracking. Visual markers are used to set spatial references to correctly place the visual representation of the assembly piece regarding the semi-assembled object. These relations are defined in the assembly instructions for each assembly step.

For the first assembly step we consider a special visual marker that defines the beginning of the object assembly; the user can set it wherever he/she likes, and the object assembly will start from the reference frame defined by that marker. But as the assembly steps are followed, there is the need to have visual markers attached to the semi-assembled object. Thus, we have connected visual markers with some of the object pieces; as the object is being assembled, the number of attached visual markers grows, allowing more flexible spatial referencing for the object pieces that are yet to be assembled. While assembling the pieces, the user may occlude some visual markers, so we needed a multi-marker tracking system: for each step, there are several assigned markers, each one with a set priority. Because of their relative position and orientation in relation to the camera, some visual markers may induce larger errors when calculating the pose of a given assembly piece; priorities are defined based on visual marking positioning in relation to the intended assembly piece position. The system enables spatial tracking using the highest priority visible visual marker, setting its state to active.

Visual markers have an on/off state. In an assembly step, there are visual markers used for spatial reference and others that are left unused. As pieces are assembled, there is a switch in the useful visual markers; for instance, while hammering some nails on the back of a drawer, there is no need to activate visual markers that are on the front of the drawer (but still one can do so if one chooses).

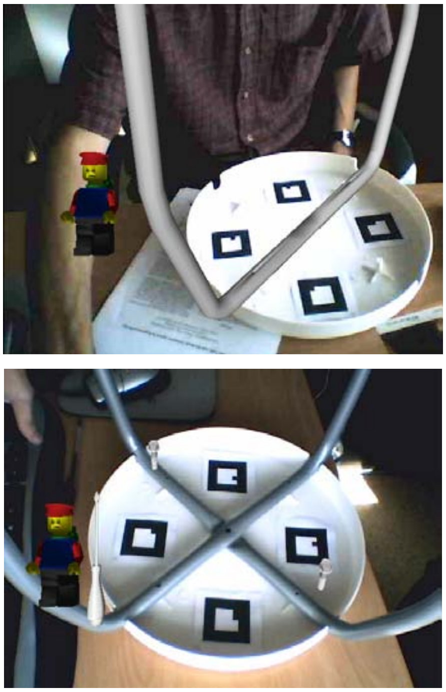

Figure 3: *Using several visual markers to place object pieces*

3.4 Puppets, puppeteers and master of puppets

A puppet is a controlled character operated from above with strings by a puppeteer. A puppeteer is a person who manipulates a puppet. The master of puppets follows the story and controls the actions of several puppeteers, functioning as choreographer and stage director. These definitions were the starting point to build our assembly helper system.

To guide the user through the assembly process, we need helpers that can represent simple tasks such as hammering, attaching and rotating objects. A humanoid character is best for enhancing the user understanding of some simple tasks, as it replicates the behavior of the human assembler; visual add-ons (arrows, 3D object representations, icons) are good tools to aid the user on its present action, enabling a quick perception of what is needed to solve a problem or how to solve it. Puppets are these visual helpers: they can perform several actions, but they have to be told what and when to do something, since they don't control the timing of each action. An understandable command is, for example, *"screw bolt"*.

Puppeteers control puppets telling them exactly what type of action to do, as well as where to do it. They don't have a complete vision of the problem, and work separately from assembly step to assembly step. For instance, a puppeteer knows how to solve the problem *"screw bolt type 1 on piece type 4"*; it commands its puppets to do the required actions needed for completing each assembly step. There are puppet puppeteers and object puppeteers. Puppet puppeteers control the humanoid characters as well as the visual add-ons. Object puppeteers control the assembled object.

Figure 4: *Puppet puppeteers control the lego character and the hammer; object puppeteers control the nails*

While there can be several puppets and puppeteers at work at the same time, there is only one active master of puppets; it follows the assembly instructions, dividing them in each assembly step. According with each assembly step, it determines what must happen to solve each problem; as each assembly step is completed, it resets its state and advances to the next step. It knows at all times what available puppeteers are under its command, and performs commands such as *"Connect piece 1 on semi-assembled object at a given location, showing visual representation of piece 1 and hammer visual helper"*. It functions as an action director, controlling what is happening in the user display due to the current stage of the assembly process.

4. Piece-marker group puppeteer

The need to control the relation between several visual markers with several assembly pieces at different times creates the need for a structure that binds pieces and markers dynamically. We have adopted the same approach by building a puppeteer that controls these relations. This is needed since the piece-marker group needs to be controlled through each assembly step. This group doesn't have the knowledge of the current assembly situation neither it has capacity of deciding what pieces to link to what object; instead, its state is controlled by the master of puppets, that instructs it to create the needed groups for each assembly step, keep track of possible changes and determine the visual marker priority for each assembly piece.

For each assembly instruction, this puppeteer turns on the needed visual markers and links them with the piece(s) being currently assembled. In marker-based Augmented Reality, it is usual to have a visual marker assigned only to a specific object during execution. In our case this is not acceptable, as the need to bind several pieces to the same visual markers arises. As stated before, the visual markers are already attached to the assembled pieces so, when a new piece is to be assembled, KAH uses the already existent visual markers in the semi-assembled object; but when assembling the next piece it might use the same markers as previously.

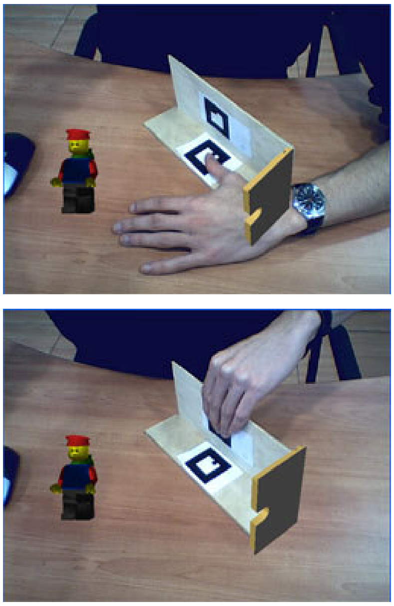

Figure 5: *Using multi-markers in order to limit occlusion problems*

This not only allows a flexible use of visual markers, as well as limits the number of needed visual markers. There is no need to have a visual marker for each piece, providing there are already useful markers in place.

Figure 6: *Using the same visual markers to place a new piece*

Group markers are sets of visual markers which are dynamic through the assembly of a given object. Several assembly steps may use the same group markers, but they also may need new group markers. They are built on run time and a visual marker may be present in several group markers. This enables us to use a dynamic multi-marker system.

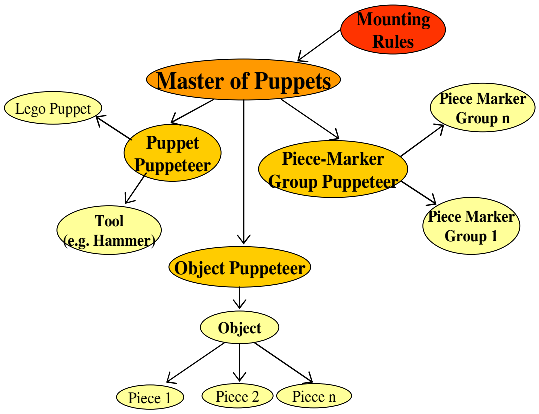

Figure 7: *Kit Assembly Helper hierarchy*

5. System architecture

5.1 Component definition using XML

To make KAH a flexible system, it has to be capable of supporting the assembly process of several different physical objects: a normalized representation of the objects and of the assembly instructions is needed. Since we want to have the capability of loading new objects without having to recompile the system code, we chose to work with configuration files with XML definitions for the objects to be assembled. Each object requires 2 XML configuration files: a file that specifies the given object and another that sets the assembly instructions.

```
<Puppet>
  <Object>
      <filename>boneco.3ds</filename>
      <xscale>6</xscale>
      <yscale>6</yscale>
      <zscale>6</zscale>
      <xtransl>-20</xtransl>
      <ytransl>-10</ytransl>
      <ztransl>-50</ztransl>
      <angle>-1,57</angle>
      <xrot>1</xrot>
      <yrot>0</yrot>
      <zrot>0</zrot>
  </Object>
  <Object>
      <filename>hammer.3ds</filename>
      …
  </Object>
  <Object>
      <filename>seta.3ds</filename>
      …
  </Object>
</Puppet>
```

Figure 8: *XML representation of an Object*

The XML representation of an object is, in fact, equivalent to a given chapter in kit assembly manuals where it is specified what comes inside the kit box (e.g. 4 bolts, 2 sets of legs, 1 instructions manual …). The difference is that KAH doesn't enumerate each of the components; instead it gives them a name, which will be used throughout the assembly process, and specifies the basic scale, translation and rotation transformations for each one of them, in relation to the corresponding marker reference frame. These are used as the basic transformations for each type of piece during the assembly process, where other transformations are combined with the base ones. Furthermore, the XML representation defines the filename for the visual 3D representation of each piece, so it can load and register the 3D object in AR.

The assembly instructions file is composed by all the needed assembly steps. Each assembly step has four elements: the path to a prompt sound file to be played, the type of action to be performed by the puppet, the wait state and information about what pieces should be used and how, including transformations and relations with the visual markers reference frames. The sound files enable the possibility of voice instructions, which will explain in detail what to do, to correctly assemble the object. The wait state turns on/off the need to wait for a user input to continue to the next assembly step. The type of action defines what is to be done in this assembly step (hammer, assemble, screw, etc) and the action to be performed by character puppeteers and visual add-ons.; this provides visual assembly help and reflects the actions users must execute in order to correctly finalize the assembly step.

As stated before, each assembly step needs to correctly register the position and orientation of the piece that is being assembled. Working with several visual markers at the same instant requires the definition of different piece transformations for each visual marker, because their spatial location is different, therefore effectively defining a multi-marker. Depending on the active visual marker, the correct transformation must be loaded. Thus, the XML file defines what pieces should be visualized and sets the transformation with each marker group. This can lead to large assembly instructions files, if one uses many visual markers for a given assembly step; but it enables a better solution for the problem of visual marker occlusion by other pieces and by the user.

```
<Rule center_on_mark="1">
<init_sound_filename>lado_1Drawer</init_sound_filename>
  <puppet_action>1</puppet_action>
  <wait>1</wait>
  <pieces exist="1" show_all="0">
      <piece_marker_connection name="lado"
        multi_marker="0" group_marker_id="1">
          <marker marker_num="5">
              <xscale>3</xscale>
              <yscale>3</yscale>
              <zscale>3</zscale>
              <xtransl>0</xtransl>
              <ytransl>0</ytransl>
              <ztransl>0</ztransl>
              <angle>0</angle>
              <xrot>1</xrot>
              <yrot>0</yrot>
              <zrot>0</zrot>
          </marker>
      </piece_marker_connection>
  </pieces>
</Rule>
```

Figure 9: *XML representation of an Assembly Step*

All markers and group markers are numbered and have information about the visual object with whom they can be registered. An assembly step may have one or more piece-marker groups defined for each piece, and a piece-marker group can be linked to only one piece at a given assembly step.

5.2 Hierarchy

The top of our hierarchy is controlled by the master of puppets; it parses and interprets the XML assembly instructions, controlling the evolution of the assembly steps. It then defines which actions each of the middle level puppeteers must perform.

The puppeteers can interpret simple action commands defined by the master of puppets, deciding on what puppets must do to act accordingly. Puppet puppeteers control the assembly helpers, in order to provide visual aid to the user; piece-marker group puppeteers switch visual markers state between on and off; object puppeteers can set pieces' visible state through each assembly step.

Puppets are the lower level elements, performing simple actions when their puppeteer commands them.

See Figure 7 for a schematic representation of KAH's structure hierarchy.

5.3 MX Toolkit

As our base development kit, we use MX Toolkit [DSB03], a C++ software platform oriented to the Augmented Reality/Mixed Reality application developer. It is based in the AR Toolkit [Kat01], but provides a simplified initialization process of OpenGL and video capture modules. Its core class makes it easy to associate visual markers with 3D shapes, update marker's transformation matrix relative to the camera on each video frame (using AR Toolkit Marker Tracking Module) and easy to use methods to render object's 3D shape using OpenGL Render Module.

MX Toolkit is object oriented and provides a higher level abstraction layer, and is currently used by our development team in several other projects, as it has proven its utility before.

6. Usability evaluation

6.1 Methodology

The usability testing experiment was designed to assess the usefulness of an AR assembly helper, compared to traditional paper assembly instructions. For this we evaluated the way users performed the assembly of simple kit objects. The chosen object was a drawer assembly kit that has the following types of interaction:

- Combining object pieces
- Hammering
- Rotating semi-assembled objects

This same task had to be done in both paradigms – paper assembly instructions and AR assembly helper – so that they could be compared. For that we used a time metric, clocking each user's duration to complete the task in each paradigm, to give us a measure of how efficiently the assembly could be done.

The usability experiment was run on 11 unpaid users, which were in their late 20's and with different professional and academic backgrounds, such as educational science, history and arts. All of them had little or no experience with AR interfaces, but knew how kit assembly is usually performed.

The AR usability tests were run on Pentium IV 3.0 GHz machines, with 512 Mbytes of RAM and a Intel 82945G with 128Mbytes graphic card, using a common webcam and a monitor as visual feedback system. This configuration was chosen because it represents a standard personal computer, enabling tests to be performed by almost anyone; the use of other equipment, such as AR goggles or other head mounted displays, would limit the scope of these tests.

First, the group was introduced to basic AR concepts, and shown some AR applications and how they work. Then, a demo session was given, explaining the project, demonstrating its features and briefly showing what could be done with the system and how. The facilitator then showed the three different types of interaction needed to perform the proposed task. After the session, each subject was given five minutes to fill a questionnaire related to their profile and background. The test subjects were then evaluated one at a time in individual sessions. First they were given the paper assembly instructions and asked to assemble the object accordingly; then the same task would be performed using the AR assembly helper. Once finished, the user was asked to answer another questionnaire related to the experiment.

Figure 10: *Subject performing the usability test*

6.2 Evaluation results and analysis

All the test subjects were successful in completing the experiment with paper assembly instructions as well as with KAH. We identified several issues during the usability test and received several comments and suggestions, which we consider to be future work.

Subjects took 2 to 3 minutes completing the test following paper assembly instructions and 4 to 6 minutes performing the assembly tasks with KAH. This is derived of user familiarity with traditional paper assembly instructions, and their lack of experience using Augmented Reality environments. The drawer chosen to perform the tests is simple to assemble, and some users didn't follow all the advised steps in paper assembly instructions, jumping to what they knew would be the final assembly state.

Most subjects preferred to perform the assembly tasks using KAH rather then following paper assembly instructions; the focus was on combining object pieces, as KAH had the best results when compared to paper assembly instructions. Visual information about where to place the piece was considered a good help in order to perceive the problem. The hammering and screwing task was considered well supported by KAH, as well as rotating semi-assembled objects. Overall, all subjects considered KAH to be better for reaching a solution, and all except one considered KAH to be more efficient.

The major problem we found in the usability tests was the lack of subject experience dealing with the system axis; as we concluded, subjects expect the images on the monitor to respond as a mirror, i.e. moving a real object to the left should also move the real object to the left in the screen. This led to assembly errors and discomfort during the experiment.

As KAH is a step-by-step wizard, some test subjects stated that paper assembly could give a better overall notion of the assembly process because it enables the simultaneous visualization of all the assembly steps.

In order to enhance identification of the current piece to be used, subjects suggested virtually numbering the pieces; this way, they could faster choose the correct piece to assemble. This would also be an advantage to show the current assembly step being performed.

On rotating semi-assembled objects, some subjects were confused by the axis and degrees of rotation, and suggested a creation of a better visual aid.

Background speech was considered a good add on, as it explained what to do in each step, helping user comprehension of the system and proposed assembly solution.

7. Conclusions and future work

We have created an augmented reality hierarchical framework called Kit Assembly Helper that provides a flexible platform to assemble objects originally in a kit format. By using a theatre stage playing metaphor, we allow the use of different types of visual aid experiences, as well as dynamic control of several visual markers. Objects and assembly rules are defined using XML schema files, making it easy to create assembly rules for new objects, and allowing the possibility of configuring and defining different kinds of visual helpers.

We have tested and performed usability evaluation of KAH with inexperienced users who compared it with paper assembly instructions, concluding that it can better lead the user through a kit assembly process. Based on the experience, suggestions were collected in order to define system expansion.

We are currently developing new 2D based visual assistants that replicate paper assembly instructions; the user can then walk though the assembly steps following both types of paradigms. A better piece identification system is being pursued, as well as a more accurate way to show the current overall assembly state.

In order to solve the inverted mirror problem stated by many users, an x-axis inverting system is being developed, so the user can choose the orientation of the augmented reality environment.

8. Bibliography

[BPS04] Barakonyi I., Psik T., Schmalstieg D.: Agents That Talk And Hit Back: Animated Agents in Augmented Reality. *ISMAR*, pp. 141-150, Third IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'04), 2004.

[DJC*03] Dias M., Jorge J., Carvalho J., Santos P., Luzio J.: Usability Evaluation of Tangible User Interfaces for Augmented Reality. *ART03* (2003), The Second IEEE International Augmented Reality Toolkit Workshop, Waseda University, Tokyo, Japan, 2003.

[DSB03] Dias J. M. S., Santos P., Bastos R.: Developing and Authoring Mixed Reality with MX Toolkit. *ART03,* 2003.

[IKEA] www.ikea.com, last visited in 29th March 2006.

[Kat01] Kato, H.: Developing Applications with ARToolkit. *SIGGRAPH 2001,* Course Notes 27, 2001.

[TOBM03] Tang A., Owen C., Biocca F., Mou W.: Comparative Effectiveness of Augmented Reality in Object Assembly. *ACM CHI '03*, Conference on Human Factors in Computing Systems, Ft. Lauderdale, Florida, USA, 2003.

[WOSL01] Wiedenmaier S., Oehme O., Schmidt L., Luczak H.: Augmented Reality (AR) for Assembly Processes – An Experimental Evaluation. *ISAR01* (2001). IEEE and ACM International Symposium on Augmented Reality, Columbia University, New York, USA, 2001.

[ZHB*03] Zauner J., Haller M., Brandl A., Hartmann W.: Authoring of a Mixed Reality Assembly Instructor for Hierarchical Structures. *ISMAR*, p. 237, The Second IEEE and ACM International Symposium on Mixed and Augmented Reality, 2003.