AR4CAD
a framework for the creation of assembly assistants based on augmented reality

Luca Piccinini, Claudio Salati
T3LAB, Bologna, Italy
Email: claudio.salati@t3lab.it
Web: www.t3lab.it

Summary
Applying augmented reality (AR) techniques in the industrial environment raises several issues. First of all, the creation of an AR based assistant may not be too complex (e.g. should not involve an experienced programmer but rather a process expert) and may not take too much time. Second, the creation and execution of the assistant should be integrated with the overall industrial process, e.g. with the design of the equipment that we want to assemble and with the quality inspections that we want to apply during the assembly process. Finally, the use of AR techniques, and in particular of artificial vision, should not require a particular instrumentation of component parts and of the environment, such as the widespread introduction of markers. AR4CAD means to be a proof of concept that all these issues can be tackled based on existing technology.

Keywords
Augmented reality, CAD, assembly, machine vision

Introduction
AR4CAD makes use of the CAD description of a complex object and of CAD tools to support the efficient design of the storyboard of the virtual assembly of that object; then, starting from this storyboard, a specific assembly assistant, based on augmented reality (AR) techniques, is automatically generated.

The assistant supports the operator during the assembly by:

- guiding him through the appropriate sequence of steps of the process;
- recognizing the individual objects being handled (without the need of specific markers) and eventually checking their size;
• providing the information required to mount component parts in the right place and position. This is done primarily by properly overlaying virtual companion parts aside the image of an anchor object being captured by the camera (Figure 4, left) or by overlaying a completely virtual assembly animation over this same anchor (Figure 4, right);
• verifying (to some extent) that component parts being used are the right ones and that assembly steps have been performed correctly.

The overall structure of the AR4CAD framework (built on top of the Autodesk CAD suite, and in particular on Autodesk Publisher [1]) is shown in the following Figure 1.

![Figure 1: overall structure of the AR4CAD framework](image)

The system, developed so far in prototype form, makes use of a camera observing the work space (in practice a table). An additional camera located at a known distance from the bench surface is used for precise measurements (Figure 2).

![Figure 2: assembly station setup](image)
the other based on the storyboard (Figure 4).

The interaction with the operator is largely voice based, in both directions: instructions are normally provided through on-line voice synthesis, though the operator may ask to see them in written form, and the operator issues commands and requests verbally, thanks to voice analysis.

A text based contextual interaction menu is continuously overlaid on the scene image captured by the camera and presented to the operator (currently on a PC screen, Figure 4).

Menu requests can then be issued not only using the voice but also the standard keyboard (selection of requests through the movement of hands has been conceived but not yet implemented).

Additional interactive, contextual helps may be requested in this ways, such as the display of manual pages, videos and detail information that is automatically extracted from the CAD database (e.g. physical dimensions, part number, warehouse info, …).

The assembly workflow

The assembly workflow is represented as a tree (Figure 3):

- leaves represent the single, atomic components,
- intermediate nodes represent subassemblies at different levels,
- the root is the final assembly.

![Figure 3: overall assembly workflow and one of its steps](image)

The workflow of our application, for a single assembly step, is as follows (along a fetch-check-mount-check pattern):

1. part A is shown to the operator for him to fetch;
2. the operator takes part A and shows it to the camera;
3. the system verifies whether the part that has been shown is correct;
4. part B is shown to the operator for him to fetch;
5. the operator takes part B and shows it to the camera;
6. the system verifies whether the part that has been shown is correct;
7. the system knows that all required parts are available and is now able to display the instructions
for the assembly task (A, B → A+B). As already stated, instructions are presented primarily verbally, and by properly overlaying virtual companion parts aside the image of a real anchor object that the operator is requested to show to the camera (Figure 4):

8. the operator shows the assembly to the camera and tells the system to check it;
9. the system verifies whether the assembly step has been completed correctly.

The overall assembly process is shown in Figure 5. In order to complete the assembly, the tree (Figure 3) must be traversed in some form of post-order.

Creation of an AR assistant

The creation of the storyboard (http://www.youtube.com/watch?v=y7NJG1_C1M8) takes an amount of time comparable to that required for the execution of the described physical operation (in our experience, 3 to 5 times as long, once all material is available and the AR4CAD programmer knows exactly what she wants to do) and, as already stated, takes advantage of available CAD tools, in our case Autodesk Publisher [1]:
The AR4CAD programmer (user) produces step by step the exploded view from the CAD project (Figure 6). Notice that the relative pose of the objects appearing in the frames of the storyboard is semantically relevant and taken into account in later processing and operations (e.g. for object recognition and for the presentation of instructions).

Figure 6: stepwise exploded view drawing

The explosion storyboard is reversed and this makes up the sequence of steps of the assembly storyboard (definition of the explosion sequence is the responsibility of the AR4CAD programmer).

The assembly storyboard is instrumented with suitable AR4CAD pragmas and additional information, such as the contextual interactive helps and commands that will be available to the operator in each step (e.g. links to manual pages, duration of an assembly step, ...).

The instrumented storyboard is compiled by the AR4CAD compiler into an executable image (in practice, an XML file) that describes, step by step, the assisted assembly procedure. Compilation is a complex process since it automatically produces a lot of additional information that is all linked into the executable image; most of this information is meant to be shown to the operator during the execution of the physical assembly (virtual model of CAD objects, virtual animation of assembly steps, detail information about objects that can be extracted from the CAD database -e.g. part number, description, location in the warehouse-, relative position of component objects during each assembly step), but the AR4CAD compiler produces also the image vision models that will be used by the object recognition software that is part of the AR4CAD interpreter and all that is needed to perform requested visual inspections (e.g. the dimensions of objects and their type).

Finally, the executable image is run by the AR4CAD interpreter, which guides an operator through the physical assembly (http://www.youtube.com/watch?v=LGFBReqK2bs).

Multidisciplinary integration

Using a machine vision library

Different from most AR systems AR4CAD can work completely markerless (even though DPM, Direct Part Marking, is supported) and can handle untextured objects. In order to achieve this result a state of the art machine vision library (MVTec HALCON [2]) is used. To our knowledge it is the first time that standard machine vision toolkits are used to recognize and track objects in AR systems: this type of toolkits was considered unsuited due to the related computational complexity and the resulting difficulties in tracking objects. We think that we have proven that this is not the case any longer and
that, with the use of some heuristic optimization techniques, it is possible to combine the object detection and localization capability of industrial machine vision with the efficiency requirement of AR. Currently AR applications built with AR4CAD can run at an average frame rate of 10fps using VGA cameras.

Having machine vision tools available has allowed us to introduce inspection steps in the storyboard (Figure 7), checking both the dimensional characteristics of the objects (e.g. diameter and length of bolts) and the correctness of the activity of the operator (e.g. all bolts are actually present where they should in the assembly). Again, to our knowledge this is new in AR systems.

Completeness of an assembly  
Dimensional checks of parts

Figure 7: automatically generated visual inspections

In principle, visual inspection procedures are also meant to be supported automatically by the system. Consider the presence of a component part in an assembly: once it has recognized the assembly, the system knows where to look for the part, and it knows also its correct pose, so that invoking a highly constrained object detection procedure with a high score request should be enough to perform the check. Unfortunately, the component part and its expected pose may be such (e.g. the head of a screw) that additional, ad hoc image processing operations are necessary to avoid false results (the resolution of the camera and environmental condition can also play a role): some work is still necessary to completely automate the support of visual inspections (e.g. using additional information, such as color, that appears in the CAD database).

Other technologies

The realization of the AR4CAD project has implied the integration of several different technologies, each of them intrinsically complex: voice processing, modeling, visualization, geometry recognition. We think that the relevance of AR4CAD is not related to particular advances in any of the technologies it is using, but in their integration in a framework for the rapid development of specialized AR based assistants. It was actually an Autodesk interface person that commented about AR4CAD that “Seeing voice recognition, modeling, visualization, geometry recognition working all together creating an augmented reality ‘experience’ is ‘enlightening’.”

As a matter of fact all of these technologies are addressed through the use of available toolkits: this is for instance the case for voice analysis where we are using CMU Sphinx [3]. In some cases the current state of the art of AR4CAD is far from satisfactory: this is the case, for instance, for visualization where no particular effort was spent to completely support a proper interaction between the images of virtual and real objects given their relative pose (e.g. occlusion of virtual objects by physical objects is not supported).
In fact even the AR4CAD programming IDE (Integrated Development Environment) is made only from an augmented version of Autodesk Publisher: in this prototypal implementation the goal was only to prove that an effective, sophisticated IDE could be implemented as a CAD/CAM plug-in.

Further developments

Extension to replacement assistants

When we considered the use of CAD based tools for the creation of assistants for industrial operations we first limited ourselves to assembly: this is because the storyboard of an assembly is straight and deterministic.

Maintenance looks far more complex: if we want to replace a component of an assembly, the number of storyboards we have to design is related to the number of parts we want to substitute; each storyboard is still deterministic, but we have to choose between them based on the part that we want to replace (and we have to design all of them!). And if we want to consider diagnosis, then our storyboard must allow for dynamically selectable, alternative branches.

In spite of this, we wanted to test whether our approach could be realistically extended to maintenance, and in order to do this we made the framework support the replacement of a known, defective part of an assembly.

First of all, we found many similarities between a replacement and an assembly task: what the operator has to do after the substitution is re-assemble all the parts he has removed to reach the defective one. So the second phase of the replacement process is a normal assembly task.

The first phase consists of removing all subassemblies and all parts that "occlude" the one that the operator has to replace: in this case we can see this first phase as an inversion of the assembly process.

With reference to Figure 3 and supposing that the part we want to substitute is B, the replacement workflow is as follows:

1. assembly at Level 5 is shown to the operator;
2. the operator takes the assembly and shows it to the camera;
3. the system verifies whether the assembly that is shown is correct;
4. the system is now able to display the instruction for the disassembly task (A+B → A, B);
5. the operator executes the task;
6. the system verifies the obtained subassembly;
7. if the tree Level is not the one of part B the workflow continues with step 1 from the new Level we have achieved;
8. the system is now able to display how to remove part B;
9. the operator removes part B;
10. the system verifies the obtained subassembly;
11. from this step a normal assembly workflow starts to complete the whole assembly.

Further improvements

Beside the weakness points that we have already mentioned, we want to highlight also the following ones, since they touch different, architectural aspects of the system:
1. AR4CAD is currently limited to 2D computer vision: that's for instance the reason why the secondary camera must be located at a known distance from the work bench. Using 3D stereo vision, especially when coupled with structured light, could significantly improve object recognition and measurement.

2. AR4CAD storyboard and pragmas do not support iteration of operations (for loops): iteration would be useful in case a same subassembly is present multiple times in the assembly. The IDE and the pragmas that are used to describe the storyboard should undergo a major redesign (moreover, in the meantime, support of Autodesk Publisher has been discontinued).

3. Voice processing has very poor performances, especially from the standpoint of analysis (poor recognition and few supported languages): cloud based voice processing systems are now available with much better performances, e.g. Google Cloud Speech API [4].

This is actually an example of a general issue, where a lot of the information that is currently linked in the AR4CAD executable program could better be allocated on the cloud thus, for instance, facilitating its maintenance.

4. The operator interface should also be revised taking into account tablets, wearable devices (e.g. smart glasses), virtual interfaces and gesture analysis.

Conclusions

The scientific challenges that AR4CAD wanted to face, and that we think it has successfully addressed are the following:

- It is possible to realize a CAD based framework that supports the rapid development of AR assistants for industrial applications. These assistants can automatically make use of all information that is present in the CAD system.
- Thanks to the use of a state of the art machine vision library, the AR based assistants that are generated using the framework can work without markers and can deal even with the highly 3-dimensional, highly directional, untextured objects of variable surface characteristics that are typical of mechanical assemblies.
- The use of a state of the art machine vision library does not significantly hamper the frame rate that can be achieved by the system nor the tracking of objects.
- Thanks to the use of a state of the art machine vision library, the AR assistants that are generated using the framework can effectively and automatically integrate visual inspections.

The AR4CAD project was partially funded by Regione Emilia-Romagna thanks to POR FESR 2007-2013 funds.

We also want to acknowledge the support of Autodesk, MVTec, and CMS.

References