PAPER VI

Augmented interiors with digital camera images

In: Piekarski, W. (Ed.) 7th Australasian User Interface Conference (AUIC); CRPIT. Vol. 50. ACS. Pp. 33–36.

Augmented Interiors with Digital Camera Images

Sanni Siltanen, Charles Woodward
VTT Information Technology
PO Box 1203, FI-02044 VTT, Finland
sanni.siltanen@vtt.fi, charles.woodward@vtt.fi

Abstract

In this paper, we present a system for Augmented Reality interior design based on digital images. The system can be used with an ordinary PC and a digital camera: no special equipment is required. Once placed in the image, virtual objects may be scaled, moved and rotated freely. In addition, the layout can be stored in file for later adjustment. We describe various user interface details and implementation issues, including a useful marker erasure method for general AR applications.

Keywords: User Interfaces, Augmented Reality, Virtual Furniture, Consumer Applications.

1 Introduction

We all know how difficult it is to choose the right sofa at the store, let alone taking it out and trying it at home. Thus, Augmented Reality (AR) technology has been proposed for interior design applications by several authors, see for example Koller et al. (1997). The related devices typically include data glasses hooked to a portable PC. A more light weight solution is to use a PDA, e.g. as proposed by Pasman and Woodward (2003). However, these devices are not commonly available for ordinary consumers.

In this paper, we present a solution for augmented interior design using just very basic home equipment, i.e. PC, digital camera and printer. A marker is placed on the floor to define the scale and coordinate system of the room. Subsequently the software system allows for a choice of virtual furniture to be placed into a room and viewed in a sequence of still images, taken from different view angles.

While a similar system has earlier been presented by the company, Augmented Solutions (2005), here we have added more functionality to the user interface and improved implementation issues, e.g. managing objects and projects, and adding lighting and shadows. Also, we present a simple method for removing the marker from the final images. This method has potential for being useful to many other AR applications too. Other issues covered include notes on Internet portal implementation of the system.

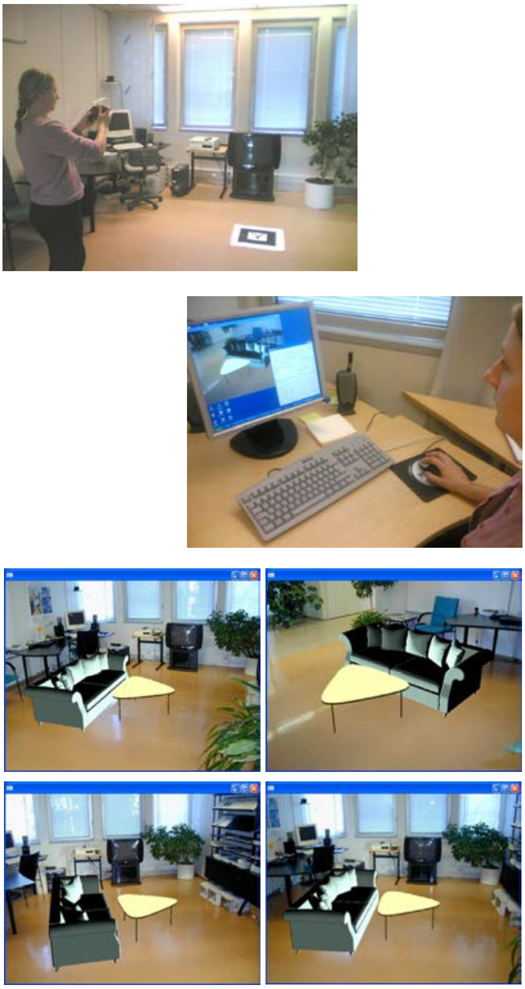


Figure 1: Images are taken, transferred to the computer and augmented with virtual furniture.

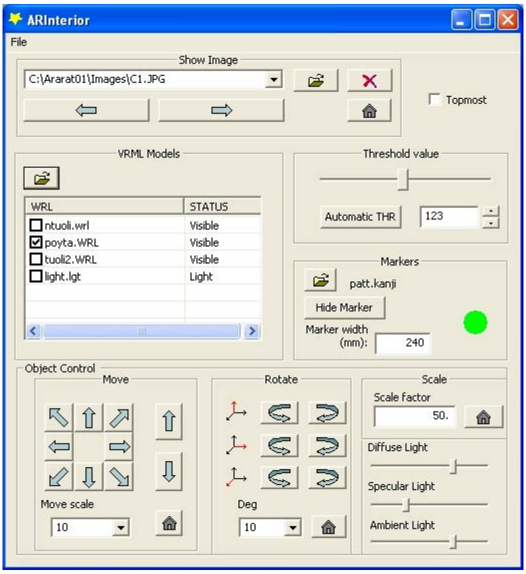
2 Operation and user interface

The operation of the system is described in following paragraphs (see Figure 1). First, the user prints out a marker that comes with the system. The style and size of the marker can be defined from the user interface in order to adapt it to the environment (viewing distance and size of the room). The marker is then placed on the floor of the room to be decorated.

The user walks along in the room and, on the way, takes a series of snapshot images with the digital camera. The images are uploaded to the PC using well established methods. Next, the furniture augmenting system is started.

Figure 2 shows the current user interface of the system. It includes functions for handling images, moving wrml-models and lights, defining marker properties and threshold values and for manipulating objects and lights. There are actually more options that would be in the real consumer application. Which of the options should be available depends on the eventual application.

The user may select different pieces of (virtual) furniture from the object list at left, then add, delete, change and hide them as required. Each piece of furniture first appears on the marker. The user may then move it to the desired position by dragging it with the mouse, or using the arrow/spin keys with adjustable step sizes. The arrow keys move the objects in the image always to the natural direction, e.g. the left arrow always moves the object to the left in the image. Many AR applications use fixed directions in marker coordinates. Accordingly, when looking from opposite direction, the object would move to unnatural direction. Our approach is more natural for the user, as he/she does not need to know anything about the marker coordinates.



C:\Ararat01\Images\C1.JPG

WRL	STATUS
□ ntuoli.wrl	Visible
☑ poyta.WRL	Visible
□ tuoli2.WRL	Visible
□ light.lgt	Light

Threshold value: 123

patt.kanji ●

Marker width (mm): 240

Scale factor: 50.

Move scale: 10

Deg: 10

Figure 2: User interface of the system.

Once the virtual furniture has been arranged the user may scan through the digital images one by one from the top left pull-down menu, or by using the next/prev buttons. In each image, the virtual furniture stays in the relative position where it was placed before. The user may also save the augmented views in several image formats and print them out. In the near future, we intend to create a "pseudo video" function where the view point automatically moves through the set of augmented images. Also, we intend to enable looking at the virtual part of the scene from any view point, for example from the ceiling.

Our system is able to handle images of any (practical) size. While the image's size may be changed on screen as pleased, actual augmenting is performed on the original size images in order to obtain full resolution and antialiasing on the virtual model textures.

All the images and virtual models are loaded to the system dynamically. Furthermore, the state of the virtual furnishing application can be saved in a project file which can be loaded later as the user desires to continue with furnishing. Also, several project files can be combined together.

The marker detection may sometimes require adjusting the threshold value by which the markers are recognised. When required, the threshold value can be defined for each image separately. The system also keeps the threshold values in store between sessions. Traffic lights in the user interface indicate how well markers are identified. Green value means that the correct marker was found, orange that some marker was found but the marker ID is unsure, and red means that no marker was detected. This guided our test users to understand when the marker was poorly detectable in the images, or when the marker defined from the user interface was different from the one shown in the images. In future versions, we will add an auto threshold function to find an adequate threshold value automatically, but manual threshold adjusting may still be required in case the automatic method fails.

3 Implementation issues

We use the ARToolKit software (version 4.0) as the engine for marker detection and 3D rendering. Some modifications to the basic ARToolKit code were required to make it work more flexibly with still images instead of video, and to enable the marker definitions (pattern file name, physical size of the marker, etc.) through the user interface. Application programming involved first of all the transformation operations to release the virtual objects from the markers.

4 Erasing the marker from images

The appearance of the relatively large size marker can be somewhat disturbing in the augmented images. Therefore, we implemented a method for automatically erasing the marker from the images that are presented to the user; see Figure 3.

The method assumes the marker is placed on a relatively uniform colour background, such as room floors typically are. Thus, after we retrieve the marker corner points from ARToolKit, we expand them to cover also the marker's white surroundings, and fill in the interior pixels by bi-linear interpolation of pixels next to the white boundary of the marker.

The method has general value for many other AR applications too, as markers are most typically placed on uniform colour walls, tables, etc. The method is fast enough even for real-time implementation, it produces very good results in most cases, and in our application it is practically always better than showing the marker.

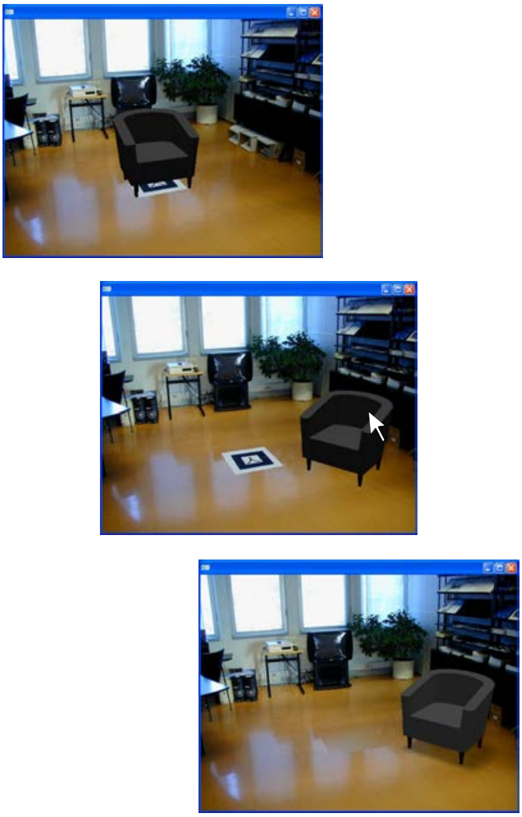


Figure 3: (Top) virtual object placed over the marker, (Middle) object moved and rotated to desired position (Bottom) marker erased and shadows added.

5 Lights and shadows

Light points can be inserted to the scene just like any other virtual object and, thereafter, manipulated with the object list. In the current implementation, we use a simple soft shadow algorithm where the shadows on the floor plane are presented using a semi-transparent alpha texture. The alpha texture is created by cumulating hard shadows based on a few jittered light points and smoothing the half-shadow (penumbra) area with an average filter. In the future, we might use a more sophisticated algorithm, like the one presented by Gibson et al. (2003).

Lights can be moved using the arrow keys just like the other objects. A red line from the marker centre to the light direction is shown for positioning aid (when the light is no more active in object list this line is not drawn). The user may change the intensity of the light source as well as the ambient lighting with the slider bars in the user interface. In this way, the brightness and shadows of the virtual objects can be adjusted to be near natural, in comparison to the eye-catchingly bright virtual objects that are seen in many other AR applications (see figure 4).

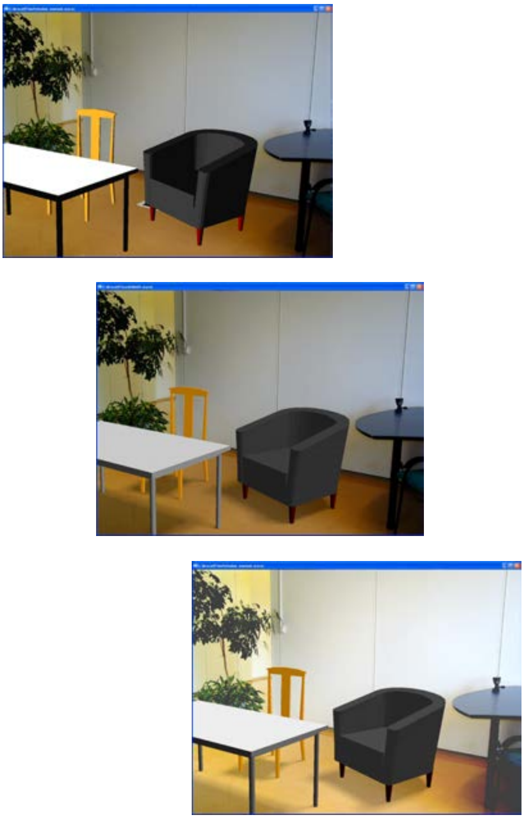


Figure 4: Without shadows, virtual objects seem to hang in the air, and if virtual lights are fixed, the brightness of objects is often unnatural (top). We augment virtual soft shadows underneath virtual objects and enable adjusting the virtual lights to the real room lighting (middle and bottom).

6 Increasing the accuracy

When using a single marker, the marker detection accuracy in the depth (z) component is much worse than in the image plane (x and y) components. This may cause the virtual objects to change their position when viewed from different directions. In a single image the object's position may look right while it is actually placed e.g. 5 cm above the floor. A second image from the side would then reveal the error and present the object in a drifted position.

Our near-future plan to improve on the depth accuracy (in single marker case) is to have an option indicating the camera's height relative to the floor/marker to the system. This would then enable numerically adjusting the detected marker position to match its known real world height. Typically, just a single height value would suffice for each user, assuming he/she stands straight when shooting the pictures.

Our system is able to use several markers, mainly for the purpose of covering large room areas but this can also be exploited to improve accuracy. Assuming the relative position of two or more markers in the real world is known (i.e. they are placed in measured positions), the detection accuracy can be improved using probability estimation of the detection error. Our method finds the most probable position and pose of the camera, based on the notions that (a) the detection error is biggest in the direction perpendicular to the camera, c.f. Pyökkimies (2002), (b) the orientation of the far away markers is more unsure than that of relatively near ones, and (c) the markers are actually all located in the same plane. This approach presents an improvement to the marker board idea used in ARToolkit which only relies on confidence values of each marker. The model matrix used in positioning of virtual objects is in a way a weighted average of those matrices calculated using separate markers.

7 Future work

Our implementation is still in its prototype stage, and we have various improvements in mind for the near future. Increasing the marker pose detection accuracy with single markers was already mentioned in the previous section. Adaptive marker thresholding, e.g. based on the method presented by Pintaric (2003), would be better than using fixed threshold value in whole image area, especially when several markers are to be detected from one image. Further, the marker erasure method could be improved by taking into account wider texture areas near the marker boundaries. Note that in our application where the marker is erased just once per image, even quite sophisticated image processing methods could be applied for this.

Currently, the walls of the room are ignored, and users may move objects unintentionally out of the room. A way to determine the wall coordinates would be having the user draw lines on the boundary between the walls and floor. Another possible solution is to find the floor by using region growing techniques based on the texture and colour of the floor. The marker is situated on the floor, thus we know from where to begin. Besides assuring the objects to be inside the room, finding the walls would also enable casting shadows on them.

The current implementation of the system is intended to serve as a desktop PC solution which the furniture manufacturer could deliver to customers by Internet download or on a product CD. However, the virtual model file sizes could be a problem with Internet download. Hence, another direction for future work is to develop the system into an Internet portal solution. It would operate on uploading images from the user's home to the web service, and downloading augmented images back, with all the 3D computation, tracking and virtual models residing on the server. Note that only overlay images of the virtual objects need to be downloaded (smaller than full images). Also, real-time interaction for positioning the furniture could be done locally using perspective approximations before refreshing the image.

8 Conclusions

We have presented an Augmented Reality interior design system that is operated with digital camera images. The system presents various improvements on user interface and presentation quality compared to earlier solutions. Most importantly, the system is easy to use. In particular, it can be operated using just ordinary PC and digital camera equipment. Also, the Internet portal implementation of our system is intended to be as straightforward as possible, and we hope to see our solutions applied in commercial use soon.

Numerous previously presented AR applications operate with real-time video. However, from user's point of view, it may be questioned how important real-time video actually is for interior design applications. Consumers are not ready to invest in e.g. data glasses, but many of them already have digital cameras available. The way people evaluate real furniture is looking from a few positions, and pausing for a moment to contemplate on what they see, in a way of still images. This is what we believe is best also for viewing of augmented interiors. Nevertheless, our system could easily be modified to process video clips too. Using e.g. the first video frame as the reference image to place the furniture, the rest follows from the discussion above.

9 Acknowledgements

We gratefully thank Mr. Mika Hakkarainen for general software engineering, Mr. Petri Honkamaa for the soft shadow implementation and Prof. Hirokazu Kato for his helpful advise with ARToolKit programming.

10 References

Augmented Solutions (2005): *Augmented Furniture Client*. http://www.ar-solutions.de/scripts/l_afc.php. Accessed 24 Aug 2005.

Gibson S., Cook J., Howard T., Hubbold R. (2003): "Rapid shadow generation in real-world lighting environments". In *Rendering Techniques 2003 (Proc. of the Eurographics Symposium on Rendering 2003)*, Leuven, Belgium.

Koller D., Klinker G., Rose E., Breen D., Whitaker R., Tuceryan M.(1997): "Real-time vision based camera tracking for augmented reality applications", *Proc. VRTT-97*, Lausanne, Switzerland, pp. 87-94.

Pasman W., Woodward C. (2003): "Implementation of an augmented reality system on a PDA", *Proc. ISMAR 2003*, Tokyo, Japan, pp. 276-277.

Pintaric (2003): "An adaptive thresholding algorithm for the Augmented Reality Toolkit", in *Proc. Second IEEE International Augmented Reality Toolkit Workshop (ART03)*, Tokyo, Japan.

Pyökkimies E.-P. (2002): *Detection of Object's Distance and Orientation*, Research report, VTT Information Technology. http://www.vtt.fi/multimedia/. Accessed 24 Aug 2005.