EUROGRAPHICS Workshop on Multimedia (2004)
N. Correia, J. Jorge, T. Chambel, Z. Pan (Editors)

Simultaneous Tracking of Multiple Objects for Augmented Reality Applications

C. Yuan[†]

Fraunhofer-Institut für Angewandte Informationstechnik FIT
Schloss Birlinghoven, 53754 Sankt Augustin, Germany
chunrong.yuan@fit.fraunhofer.de

Abstract

This paper presents an appearance–based image processing and tracking algorithm which is applied in a distributed Augmented Reality (AR) system. The tracker is computer vision based and is capable of simultaneous tracking of multiple objects. These objects are called place holder objects (PHOs), as they are used as interface elements and act as tangible interfaces for handling and interacting with virtual artifacts. The tracking system uses a fix mounted camera viewing at the workspace — a normal round table. All the PHOs are placed on the table and can be moved arbitrarily around, allowing both in–plane and out–of–plane rotations. In order to track and differentiate the PHOs in real–time, we apply an appearance–based object modeling. The utilization of appearance–based method for object recognition and tracking gives the system a distinct advantage in that it is computationally less expensive and it can be easily adapted to work with arbitrary PHOs by simply using an off-line training process.

Categories and Subject Descriptors (according to ACM CCS): I.4.8 [Image Processing and Computer Vision]: Object Recognition

1. Introduction

Detecting, recognizing and tracking of moving objects in images and/or videos can have a wide variety of applications in computer graphics and vision tasks, e.g., image coding and transmission, video surveillance, robotics and gaming. During the last few years, it becomes an important research topic in the field of augmented reality [ABB*01]. In AR applications, the augmented virtual objects can be assigned to be related to some real world objects so that the virtual world can be manipulated by an AR system user operating on those real objects. Hence the success of an AR system depends among others largely on an optical processing and tracking system. Via such tracking system, the position and orientation parameters of the real objects can be calculated. Based on these parameters, specific virtual objects can then be visualized and overlaid properly in the real world.

During the last decade, a large variety of motion detection and tracking algorithms have been proposed, using either geometrical or textural properties of the object to be tracked. One widely adopted approach applies boundary–based features and employs active contour models [CKS95], like snakes [KWT87], balloons [Coh91] or deformable templates [ZJDJ00]. These models are energy–based or geometric–based minimization approaches and they require an accurate initialization step for the algorithm to work properly.

Another approach is region based, applying the optical flow [MS98, NH98], a spatial-temporal motion estimation technique. In this case, a correspondence between the associated target regions in different frames must be established. Since point-to-point feature matching is required, the process is very time-consuming, not to say that the detection of reliable features is itself an unsolved problem in vision community. To simplify the matching problem, some tracking systems introduce constraints on the objects used, resulting in that only objects with suitable geometry

[†] This work is partially supported by the European Commission in project AUTHUR under the contract number IST-2000-28559

or distinct colors can be tracked [SSN01a, SSN01b]. Others attach markers (fiducials) on objects for easy registration purpose [CLN98, vLM03]. Since markers are generally undesirable and can be even impossible in real–world scenario, a more general–purpose tracking strategy should be approached.

In AR applications, tracking goes further than object and motion detection. When multiple moving objects coexists, all of them should be segmented and recognized. Furthermore, the six DOF (degree of freedom) pose of each object should be computed, in order that the corresponding virtual objects they are standing for can be rendered properly. Since model–based pose estimation depends heavily on the availability of reliable feature points, it has difficulties when there are a large number of object models of different objects. Also due to interaction of the user in the AR environment, object will be occluded frequently by the user, e.g. through hand movement. Therefore, we address in this paper the problem using an appearance–based approach and propose a unified approach for the detection and tracking (simultaneous recognition and localization) of several moving objects.

Our method is based directly on the incoming image stream, neither optical flow estimation nor camera calibration is required. Initially, a statistical analysis is performed and is used to provide the motion information. The interframe difference density function is considered as a two–component model corresponding to the static (background) and the moving objects (foreground). Based on this model, the input frame can be further analyzed to identify moving object regions. Using a neural appearance–based object recognition and localization approach [YN01, YN03], multiple PHOs can be tracked with six DOF pose information.

2. Approach

2.1. Detection of moving objects

Let $\boldsymbol{f}(x,y,t)$ be the current and $\boldsymbol{f}(x,y,t-1)$ be the previous input frame and let $\boldsymbol{f}_d$ be the interframe gray level difference, i.e.:

$$\boldsymbol{f}_d(x,y,t) = \boldsymbol{f}(x,y,t) - \boldsymbol{f}(x,y,t-1) \tag{1}$$

The motion detection problem can be viewed as a decision for each pixel as *static* or *mobile*. Since a static camera is used, *static* pixels correspond to the background in both frames and the *mobile* ones belong to the moving objects in the current or the previous frame. Let p be the probability density function of the observed interframe difference image. This density function can be assumed to be a mixture model of a static and a mobile component. Using the histogram of the interframe difference as a measurement, the two components can be separated using statistical decision rules.

Let Ω_0 be the background class and Ω_1 be the moving object class. Though there can be different moving objects, all of them belonging to Ω_1. The histogram of the interframe difference can be given as

$$p_i = \frac{n_i}{N} \tag{2}$$

where N is the total number of pixels in the frame, and n_i is the number of pixels whose gray–level value is equal to i. Now the problem becomes one of separating the histogram at an optimum value k so that all the pixels belonging to Ω_0 has the value smaller than k and all the pixels belonging to Ω_1 has the value greater than k. As a consequence, the probability density function of each class can be expressed as

$$P(\Omega_0) \;=\; \sum_{i=0}^{k} p_i \tag{3}$$

$$P(\Omega_1) \;=\; \sum_{i=k+1}^{m} p_i \tag{4}$$

where the value m is the maximal gray–level value of the interframe difference, usually equal to 256.

From the above, the mean of each class can be given as

$$\mu(\Omega_0) \quad = \textstyle\sum_{i=0}^{k} \frac{ip_i}{P(\Omega_0)} \tag{5}$$

$$\mu(\Omega_1) \;= \textstyle\sum_{i=k+1}^{m} \frac{ip_i}{P(\Omega_1)} \tag{6}$$

And the mean of the whole image can be given as

$$\mu = \sum_{i=0}^{m} ip_i \tag{7}$$

According to pattern recognition rules, an optimum separation should maximize the mean square difference between two different classes, namely to maximize the following function:

$$J_i = P(\Omega_0)(\mu(\Omega_0)-\mu)^2 + P(\Omega_1)(\mu(\Omega_1)-\mu)^2 \tag{8}$$

Hence the optimum value k can be obtained as

$$k = \underset{i}{\operatorname{argmax}}\{J_i\} \tag{9}$$

Using this optimum value k, the regions corresponding to possible moving objects in the current frame can be detected. In some situations, the region may not hold the whole object if the object moves slowly, or the region may contain some objects we are not looking for, e.g. background objects due to user entering the scene and/or user hand movements. Some post–processing steps have been done on these candidate regions.

On the current image $\boldsymbol{f}(x,y,t)$, within all the candidate regions, we first perform a corner detection which is similar to the Harris corner detection method. If a region contains too many or two few corners as required, it can be removed from the candidate region set. After this operation, areas corresponding to background objects or occluded regions can

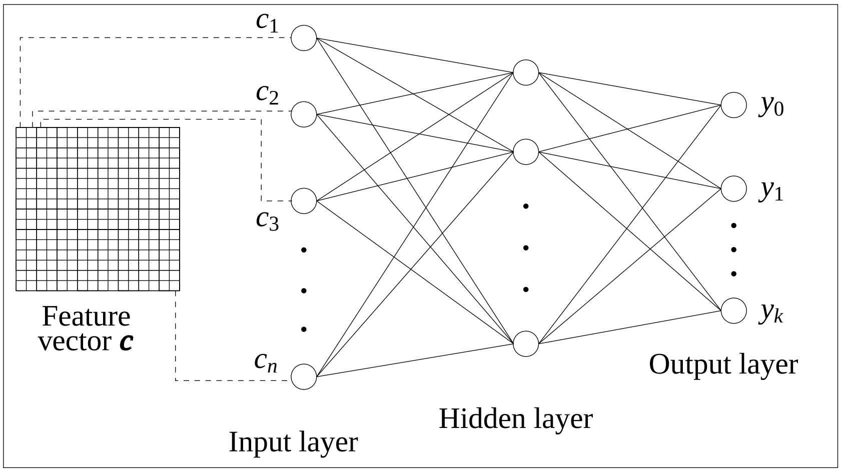


Figure 1: *A three–layer neural net for object identification.*

be removed. On the left candidate regions of $\boldsymbol{f}_d(x,y,t)$, we conduct a boundary extraction step to retrieve complete 8–connected object regions. The advantage of this processing is that it can enlarge the size of a candidate region in case it doesn't hold an entire object.That means, if the candidate region contains already a whole object, the region is kept as it is. If the candidate region contains only part of a moving object, after the boundary completion step, an enlarged region that contains the entire object boundary can be obtained.

On each of the remaining regions, either enlarged or not enlarged, we extract a square region of interest (ROI). Each ROI is actually the bounding box which contains the boundary of those regions.

2.2. Object recognition

After the detection of moving objects, the extracted ROIs in the current input frame are further analyzed to output object identity and location. In order to identify the different moving objects, we need to classify each area that contains moving objects as belonging to one class Ω_κ out of λ object classes Ω_i, $i = 0,\ldots,\lambda$. Ω_0 is the class of background objects. If an object is detected to belong to Ω_0, we don't need to compute the pose parameters since it is not a place holder object. Objects belonging to Ω_1 to Ω_λ are those place holder objects.

A classification of the ROIs into one of those classes is usually done based on some feature vector $\boldsymbol{c}$ representing the ROIs. With the purpose of a good and compact representation of each area, we use wavelet transform for feature extraction.

Given an image area $\boldsymbol{f}_0(x,y)$ with $x \in \{0,1,...,D_x - 1\}, y \in \{0,1,...,D_y - 1\}$ (D_x/D_y is the width/height of a ROI), the 2–D discrete wavelet transform is computed by applying a separable filterbank to the area repeatedly [Mal89]:

$$\boldsymbol{f}_n(x,y) = [\boldsymbol{H}_x * [\boldsymbol{H}_y * \boldsymbol{f}_{n-1}]_{|2,1}]_{|1,2}(x,y) \quad (10)$$

$$\boldsymbol{D}_{n_1}(x,y) = [\boldsymbol{H}_x * [\boldsymbol{G}_y * \boldsymbol{f}_{n-1}]_{|2,1}]_{|1,2}(x,y) \quad (11)$$

$$\boldsymbol{D}_{n_2}(x,y) = [\boldsymbol{G}_x * [\boldsymbol{H}_y * \boldsymbol{f}_{n-1}]_{|2,1}]_{|1,2}(x,y) \quad (12)$$

$$\boldsymbol{D}_{n_3}(x,y) = [\boldsymbol{G}_x * [\boldsymbol{G}_y * \boldsymbol{f}_{n-1}]_{|2,1}]_{|1,2}(x,y) \quad (13)$$

where * denotes the convolution operator, $|2,1(|1,2)$ means subsampling along the rows (columns), $\boldsymbol{H}$ and $\boldsymbol{G}$ are a low and bandpass filter, respectively. $\boldsymbol{f}_n$ is obtained by lowpass filtering and is therefore referred to as the low resolution image at scale n. The details images $\boldsymbol{D}_{n_i}$ are obtained by bandpass filtering in a specific direction (horizontal, vertical and diagonal) and contain directional detail information at scale n.

In this work, only $\boldsymbol{H}$ filter is used for computational efficiency and the lowest scale is at n=4. That means we only need to compute the $\{\boldsymbol{f}_n\}_{n=1,2,3,4}$ and the computation of $\{D_{n_i}\}_{n=1,2,3,4,i=1,2,3}$ is eliminated for simplicity.

After feature extraction, the identification of multiple moving objects with each enclosed in a ROI can be viewed as a mapping from a set of input variables represented by $\boldsymbol{c} = \boldsymbol{f}_4$ to a set of output variables representing the class labels. Suppose the output variables are denoted by y_j, with $j = 0,\ldots,\lambda$. The mapping can be modeled in terms of some mathematical functions which contain a number of adjustable parameters:

$$y_j = y_j(\boldsymbol{c};\boldsymbol{w}) \quad (14)$$

where $\boldsymbol{w}$ is a vector which embraces in it the parameters whose value can be determined with the help of the training data. A three layer neural net whose number of input neurons is equal to the dimension of $\boldsymbol{c}$ and whose number of output neurons is equal to λ can be applied to form a model for the classification. Shown in Figure 1 is the network we configured for the recognition task.

This kind of multi–layer perceptron has proven to be able to approximate arbitrarily well any functional mapping from one space to another, provided that the number of hidden units is sufficiently large [HSW89]. Also as verified in [Bis95], the output of the net y_j can be interpreted as measuring the posterior probability function $P(\Omega_j|\boldsymbol{c})$ for each class. According to Bayes rule, the area represented by vector $\boldsymbol{c}$ should be classified as coming from class Ω_κ with

$$\kappa = \underset{j}{\mathrm{argmax}}\{y_j\} \tag{15}$$

2.3. Object localization

For object localization, we are aiming at inferring 3–D properties from a single 2–D image. Making such an inference requires building the relationship between the 2–D image and the 3–D world, i.e., the mapping of the image features to object pose parameters. In order to infer object pose $\boldsymbol{p}$ from a single input image $\boldsymbol{f}_0$, we must process some form of knowledge regarding the variation of its feature vector $\boldsymbol{c}$ as a function of its pose $\boldsymbol{p}$: $\boldsymbol{c} = g(\boldsymbol{p})$. All pose estimation schemes employ some model of this image formation process. The two basic approaches to approximating $g(\boldsymbol{p})$ are model–based and appearance–based, which can be regarded respectively as analytical and empirical. In the analytical approach, an explicit and object specific model, which is the geometric configuration of some image feature points on a particular 3–D object with regard to its pose, must be computed in advance [Fau93]. Pose is then approximated by applying the model knowledge to those feature points extracted from a 2–D image of the object and solving some system of equations [DD95]. In the appearance–based approach, we dispense with geometric object models and directly approximate $g(\boldsymbol{p})$ from empirical measurements. Training samples are acquired by placing the object in a known pose $\boldsymbol{p}$ and acquiring N images from the pose space. These N samples, when combined with a method for interpolating between them, can yield an implicit pose estimation model $\boldsymbol{p} = f(\boldsymbol{c}) = g^{-1}(\boldsymbol{c})$. This is exactly the function one can approximate with neural models. Similar to object classification, the object pose can be computed as:

$$\boldsymbol{p} = \mathrm{argmax}\{f_{\boldsymbol{p}}(\boldsymbol{c},\boldsymbol{w})\}. \tag{16}$$

For a 3–D object, its pose parameter is six–dimensional, which consists of the rotation

$$\boldsymbol{R} = \boldsymbol{R}_z\boldsymbol{R}_y\boldsymbol{R}_x \in \Re^{3\times 3} \tag{17}$$

and the translation

$$\boldsymbol{t} = (t_x, t_y, t_z)^T \in \Re^3. \tag{18}$$

Here $\boldsymbol{R}_x, \boldsymbol{R}_y, \boldsymbol{R}_z$ are rotation matrices with rotation angle ϕ_x, ϕ_y and ϕ_z around the x–, y– and z–axis, respectively:

$$\boldsymbol{R}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi_x & \sin\phi_x \\ 0 & -\sin\phi_x & \cos\phi_x \end{bmatrix} \tag{19}$$

$$\boldsymbol{R}_y = \begin{bmatrix} \cos\phi_y & 0 & \sin\phi_y \\ 0 & 1 & 0 \\ -\sin\phi_y & 0 & \cos\phi_y \end{bmatrix} \tag{20}$$

$$\boldsymbol{R}_z = \begin{bmatrix} \cos\phi_z & \sin\phi_z & 0 \\ -\sin\phi_z & \cos\phi_z & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{21}$$

As object translation and rotation are unrelated, one can use one neural model for translation parameter estimation and one for rotation parameter estimation. The weight parameters in each network can be regarded as modeling $p(\boldsymbol{c}|\boldsymbol{t})$ and $p(\boldsymbol{c}|\boldsymbol{R})$, respectively. And object pose can be computed as

$$\boldsymbol{t} = \underset{\boldsymbol{t}}{\mathrm{argmax}}\{f_{\boldsymbol{t}}(\boldsymbol{c},\boldsymbol{w})\}, \tag{22}$$

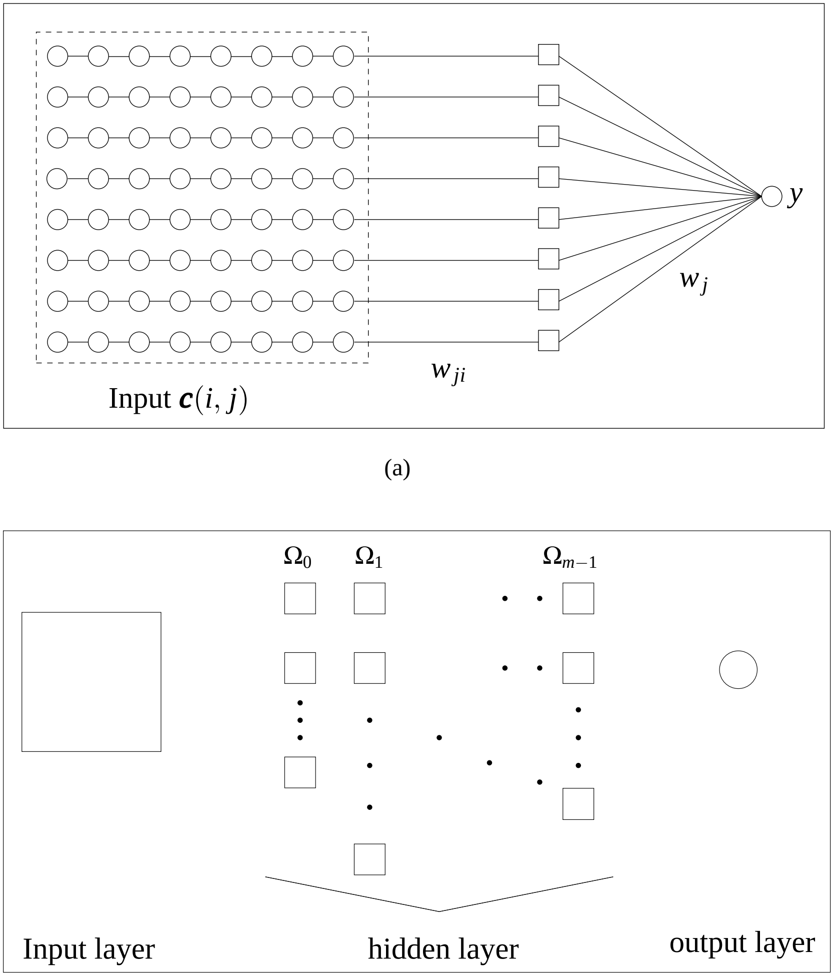
$$\boldsymbol{R} = \underset{\boldsymbol{R}}{\mathrm{argmax}}\{f_{\boldsymbol{R}}(\boldsymbol{c},\boldsymbol{w})\}. \tag{23}$$

For each of the three translation parameters t_x, t_y, t_z, we use one neural model as shown in Figure 2 (a) for the computation. In order to compute each of the three rotation parameters ϕ_x, ϕ_y, ϕ_z, we build the neural model as shown in Figure 2 (b). Different from the neural model shown in Figure 1, they are not neural classifier, but neural estimators.

3. Experimental evaluation

The proposed appearance–based approach is developed for an AR system named AUTHUR (Augmented Round Table for Architecture and Urban Planning). In order to facilitate collaborative meeting and discussions which take place frequently in the architectural planning and construction process, an indoor AR environment is set up. Users are gathered at a round table, over which a micro head camera (ELMO CC-491) is fix mounted. Eight PHOs have been used in the current AUTHUR system. As the proposed approach can handle arbitrary 3–D objects, it can handle objects with fewer degrees of freedom as well. Six of the PHOs are used as normal interaction units which can move, scale or rotate virtual artifacts. Two PHOs are pointing devices. One can point them to other PHOs to trigger some actions, e.g. changing the color or texture of the virtual objects, adding bounding box to the selected objects etc.

At the beginning all the place holder objects are placed on the table and made available to the users, as is shown



(b)

Figure 2: *Neural estimator for the computation of pose parameter.*

in Figure 3. A separate program can establish a proper relation between those PHOs and the virtual objects on the fly. Each user of the AR system wears a see–through head mounted Display (HMD) and can see the virtual objects in his HMD and can interact with the virtual objects by operating on the PHOs. Once an object has been moved by the user, the system changes the virtual object pose based on the current PHO pose. If an object stops moving for sometime, the system just keeps its old pose. If an object is outside of the camera field of view, the corresponding virtual object will be removed from the augmented scene as well.

An off–line strategy is applied to train the system to recognize and localize the eight PHOs. We collect an image data set consisting of the upper half of a sphere spanning 360^o in longitude and 90^o in latitude, with a sampling interval of 3^o. During the capture process, several illumination conditions are used. For each object, there is a sequence which consists of 3720 images with different viewpoints. Using a training set that is less than one–tenth of the whole data, the system achieved a recognition rate of 98%.

After the offline training procedure, the tracking system can work in real time with a tracking rate of 20 images/s. For the object detection part, very satisfactory results have been obtained. Based on the proposed histogram–based statistical model, mobile areas can be captured very easily and quickly. Applying the appearance–based approach, the corresponding moving object can be identified and their pose parameters can be tracked in six DOF. The average localization error is 1.3 cm for translation parameter and 1.8^o for rotation parameter.

After the pose parameters of the recognized objects have been computed from each frame, they are sent via TCP/IP to the AR system. The virtual objects can then be visualized and certain interactions can take place (See Figure 4).

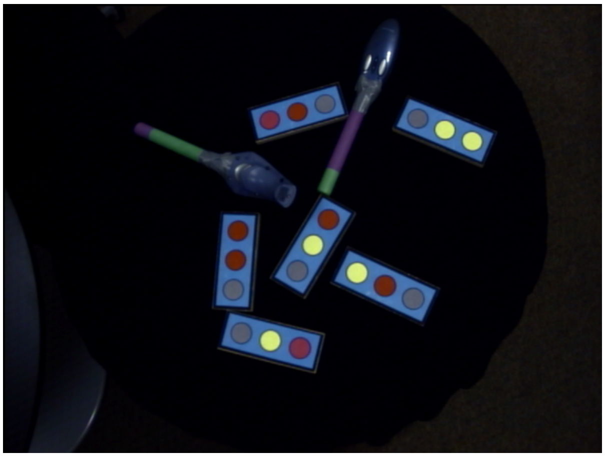


Figure 3: *Illustration of the eight place holder objects.*

In Figure 4 (a), a user is using the six PHOs arranging the six parallel beams of a construction. In the figure, the real world (table with the PHOs) are overlaid with virtual scenes. By placing the PHOs, the yellows beams can be moved towards or away from each other. In Figure 4 (b), two users sitting at the table are collaborating with each other in designing a new building. As can been seen, a large cityscape is augmented to their working environment — the round table. One user has just created the new building using the pointer PHO and is pointing to it. Because the scene graphs are synchronized, the other user can see the same virtual world and is using her finger pointing to the same object.

Up to now, dozens of users have tested the tracking system by using the eight PHOs to interact with the AR system. Although they point out that the HMD is a little heavy and some are not quite happy with the dark glasses of the HMD, every one likes the idea of using the PHOs to do interaction and is very satisfied with the tracking system. Even unexperienced users has no difficulty in using the pointers picking virtual objects.

4. Conclusion

This paper describes a new approach for the detection and tracking of multiple moving objects in video frames acquired by a static observer. Unlike many applications in AR field, the system described doesn't use extra tracking device, but is purely based on image and video processing algorithms. It works directly on the incoming image stream and can track multiple PHOs automatically. It doesn't require any manual initialization at all. With eight different PHOs, encouraging tracking rate (20 images/s) as well as tracking precision (within 2 cm in position and 2 degree in orientation parameters) have been achieved. Tests with different users have proven that, using the PHOs and the proposed optical tracking system, interaction within the AR environments can take place naturally and successfully. Currently we are planning to enlarge the number of PHOs and use multi–view video frames obtained from several cameras to enlarge the field of view and deal with occlusions. At the same time we are exploring computer vision based techniques for both PHO and head tracking in mobile AR systems, which can be used both in indoor (e.g.museums) and outdoor (e.g outdoor AR gaming) environments.

References

[ABB*01] AZUMA R., BAILLOT Y., BEHRINGER R., S. F., JULIER S., MACINTYRE B.: Recent advances in augmented reality. *IEEE Computer Graphics and Applications 21*, 6 (Nov.–Dec. 2001), 34–47.

[Bis95] BISHOP C. M.: *Neural networks for pattern recognition*. Oxford clarendon press, 1995.

[CKS95] CASELLES V., KIMMEL R., SAPIRO G.: Geodesic active contours. In *ICCV95* (1995), pp. 694–699.

[CLN98] CHO Y., LEE J., NEUMANN U.: A multi–ring fiducial system and an intensity–invariant detection method for scalable augmented reality. In *Int'l Workshop Augmented Reality (IWAR98)* (1998), Peters A., (Ed.), pp. 147–165.

[Coh91] COHEN L. D.: On active contour models and balloons. *Computer Vision, Graphics, and Image Processing. Image Understanding 53*, 2 (1991), 211–218.

[DD95] DEMENTHON D., DAVIS L.: Model–based object pose in 25 lines of code. *International Journal of Computer Vision 15* (1995), 123–141.

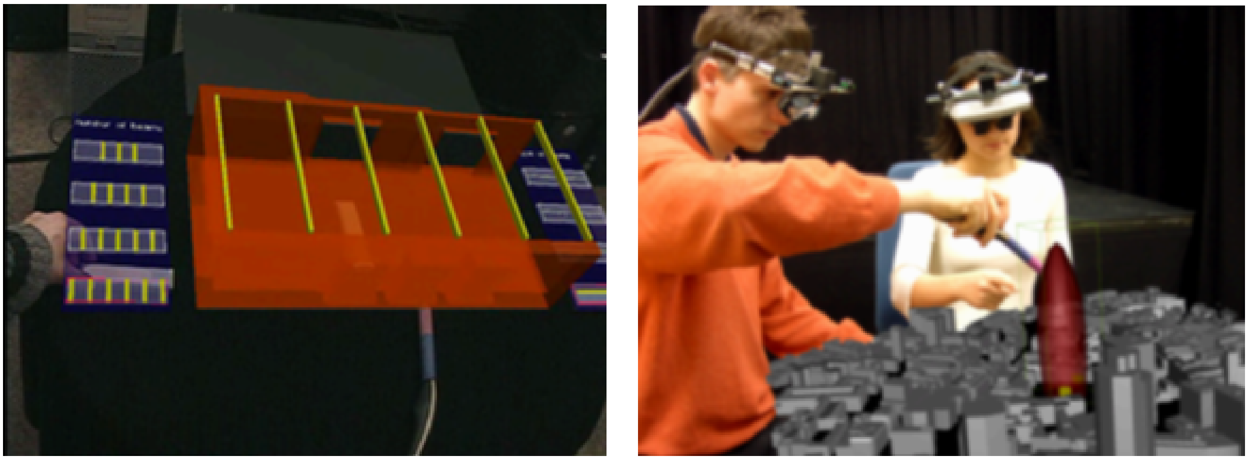
[Fau93] FAUGERAS O.: *Three–Dimensional Computer Vision – a Geometric View Point*. MIT Press, Cambridge, MA, 1993.

[HSW89] HORNIK A., STRINCHCOMBE M., WHITE H.: Multilayer feedforward networks are universal approximators. *Neural Networks 2* (1989), 359–366.

[KWT87] KASS M., WITKIN A., TERZOPOULOS D.: Snakes: Active contour models. In *IEEE Int. Conf. on Computer Vision* (IEEE London, 1987), Computer Society Press of the IEEE, pp. 259–268.

[Mal89] MALLAT S.: A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. PAMI 11* (1989), 674–693.

[MS98] MAE Y., SHIRAI Y.: Tracking moving object in 3-d space based on optical flow and edges. In *International Conference on Pattern Recognition* (1998), vol. 2, pp. 1439 –1441.



(a) (b)

Figure 4: *Using the tracking results to interact with the augmented round table.*

[NH98] NAGEL H.-H., HAAG M.: Bias-corrected optical flow estimation for road vehicle tracking. In *Sixth International Conference on Computer Vision* (Bombay, India, Jan. 4-7 1998), pp. 1006 – 1011.

[SSN01a] SCHMIDT J., SCHOLZ I., NIEMANN H.: Placing arbitrary objects in a real scene using a color cube for pose estimation. In *Pattern Recognition, 23rd DAGM Symposium* (Munich, Germany, September 12–14 2001), Radig B., Florczyk S., (Eds.), Springer-Verlag, Berlin, Heidelberg, New York, pp. 421–428. Lecture Notes in Computer Science 2191.

[SSN01b] SCHOLZ I., SCHMIDT J., NIEMANN H.: Farbbildverarbeitung unter Echtzeitbedingungen in der Augmented Reality. In *7. Workshop Farbbildverarbeitung* (Erlangen, Germany, 4.-5. Oktober 2001), Paulus D., Denzler J., (Eds.), Universität Erlangen-Nürnberg, Institut für Informatik, pp. 59–65.

[vLM03] VAN LIERE R., MULDER J.: Optical tracking using projective invariant marker pattern properties. In *the IEEE Virtual Reality Conference 2003* (2003), pp. 191–198.

[YN01] YUAN C., NIEMANN H.: Neural networks for the recognition and pose estimation of 3–D objects from a single 2–D perspective view. *International Journal of Image and Vision Computing 19* (August 2001), 585–592.

[YN03] YUAN C., NIEMANN H.: Neural networks for appearance–based 3–D object recognition. *Neurocomputing 51*, C (April 2003), 249–264.

[ZJDJ00] ZHONG Y., JAIN A., DUBUISSON-JOLLY M.-P.: Object tracking using deformable templates. *IEEE Trans. Pattern Anal. Machine Intell. 22*, 5 (May 2000), 544 –549.