Static and Dynamic Hand-Gesture Recognition for Augmented Reality Applications

Stefan Reifinger, Frank Wallhoff, Markus Ablassmeier, Tony Poitschke, and Gerhard Rigoll

Technische Universität München, Institute for Man Machine Communication, Theresienstraße 90, 80333 Munich, Germany
{reifinger,wallhoff,ablassmeier,poitschke,rigoll}@mmk.ei.tum.de

Abstract. This contribution presents our approach for an instrumented automatic gesture recognition system for use in Augmented Reality, which is able to differentiate static and dynamic gestures. Basing on an infrared tracking system, infrared targets mounted at the users thumbs and index fingers are used to retrieve information about position and orientation of each finger. Our system receives this information and extracts static gestures by distance classifiers and dynamic gestures by statistical models. The concluded gesture is provided to any connected application. We introduce a small demonstration as basis for a short evaluation. In this we compare interaction in a real environment, Augmented Reality with a mouse/keyboard, and our gesture recognition system concerning properties, such as task execution time or intuitiveness of interaction. The results show that tasks executed by interaction with our gesture recognition system are faster than using the mouse/keyboard. However, this enhancement entails a slightly lowered wearing comfort.

Keywords: Augmented Reality, Gesture Recognition, Human Computer Interaction.

1 Introduction

Augmented Reality (AR) aims at a combination of reality and virtuality in a coordinated way. Major goal is the integration of virtual content embedded into the user's real environment as realistic as possible. Commonly, interaction between user and AR application occurs by use of non-natural interaction techniques (e.g. mice or keyboards). To achieve a fully immersive AR application, the system's output (e.g. visualization) as well as system's input has to adapt to the user's reality. Thus, AR applications have to comprehend the human's natural interaction techniques, e.g. speech or gestures. For this reason, our contribution focuses on the integration of a static and dynamic gesture recognition system for the use within AR applications.

2 Motivation and Previous Work

The most common way, which humans use for interaction in reality, is using speech or gestures. For the manipulation of a real object (e.g. translation) the user employs

© Springer-Verlag Berlin Heidelberg 2007

one or two hands. This natural technique directly associates user's action and object's reaction. Manipulation of virtual objects in commonly used AR systems is accomplished by non natural interaction techniques (using mice or keyboards). However, this artificial technique does not directly associate user's action and object's reaction. Thus the user has to transfer abstract paradigms to actions (e.g. predefined mouse and keyboard input combinations resulting to a specific interaction). For eliminating this abstract interaction, AR systems should be able to offer natural interaction techniques, such as interaction by hand reducing the user's cognitive load and increasing the immersive character of AR. Therefore, our contribution focuses the integration of an automatic hand gesture recognition system in Augmented Reality.

In general, there are two major approaches to implement automatic gesture recognition systems. Non-instrumented systems work with computer vision based algorithms, which extract information about the user's hand gestures from a visually captured stream (camera). This technique does not need additional hardware mounted at the user's hands, which have to be divided from the scene's background robustly. In a second step, the position of the hand and its fingers are calculated and used for recognizing predefined gestures by use of statistical methods. Instrumented systems on the other side need additional hardware mounted at the user's hand. This hardware as part of a tracking system provides information about the hand position and orientation from which gestures are calculated.

In [3], a non-instrumented gesture recognition interface, which differentiates pointing, clicking and five static gestures, has been developed. Advantage of this approach is the non-intrusive nature of recognition, as there is no need for the user to wear any hardware. On the negative side, the hand has to be permanently visible inside the user's personal field of view (FOV) for recognition. An instrumented gesture recognition system using optical markers mounted at the user's hand, as developed by [1] implies the same disadvantage. Instrumented systems basing on non-optical tracking systems, such as those used in [4], eliminate the necessity of visible hands, but entail that additional hardware has to be worn by the user.

As the non-instrumented approach does not depend on additional hardware, worn by the user, the interaction with the AR system is not limited. But this approach leads to the necessity of the visibility of the user's hand during interaction. When the user's hand within the camera's view is lost, no gesture information is available, which disables the user to interact with the AR system. This disadvantage entails big computational efforts and a limited radius of action for the user. Additionally, such systems are sensitive for changing environmental parameters such as lighting conditions or objects similar to a human hand.

An instrumented approach limits the user by the necessity of wearing additional hardware. But this hardware increases the recognition rate of the user's hand, because tracking systems are optimized to track this hardware and gather information about their position and orientation. Those systems are more resistant for changing environmental parameters. Instrumented approaches, using non vision based tracking systems offer the advantage that the hand has not to be visible for a camera permanently. Such non vision based systems come with larger hardware cutting down the radius of interaction and wearing comfort of the user.

3 Implementation

To avoid occlusions endangering the continuous visibility of user's hands, our approach bases on an instrumented infrared (IR) tracking system (ITS) containing a six camera array [2]. Originally, this tracking system has been implemented for measuring the user's position and orientation for AR applications. Now this ITS setup enables additional tracking of the user's hands at the same time. To keep down the intrusive character of instrumented systems, we use two light-weighted IR-tracking targets (IRTTs). These IRTTs are mounted at the user's finger to receive the position and orientation. Our ITS delivers tracking data in temporally and spationally high resolution. For a more flexible interaction, both hands are used for gesture recognition. Due to the human's way of gesturing, static (e.g. pointing) as well as dynamic gestures (e.g. clapping) will be recognized.

For our gesture recognition system, we differentiate static and dynamic gestures. Static and dynamic gestures differ in the angle between the user's fingers. Static gestures are defined by the angle between the fingers and do not vary in time, e.g. pointing or grasping. Dynamic gestures are marked by changing angles between fingers during elapsed time, e.g. waving or drawing letters in the air. The position of the hand can vary both in static and dynamic gestures during the time.

Our gesture recognition system is able to differentiate static and dynamic gestures. In our demonstration application, the user is able to point, grasp and scale by using static gestures. Pointing and grasping is performed by using one hand (either right or left), scaling by using both hands. Pointing implies a right angle between user's thumb and index finger (index finger points on the object), while other fingers are angled. Thumb and index finger are formed to an "O" (tips of both fingers are touching) for grasping an object, other fingers are angled again. For scaling, the user grasps the object with both hands in a defined minimum distance and pulls them apart.

Currently our demonstration application only recognizes two dynamic gestures, which are performed by either right or left hand drawing a symbol "X" or "O" in the air.

The subsequent gesture recognition systems bases on a master-client architecture and consists of three main parts (see Fig. 1.).

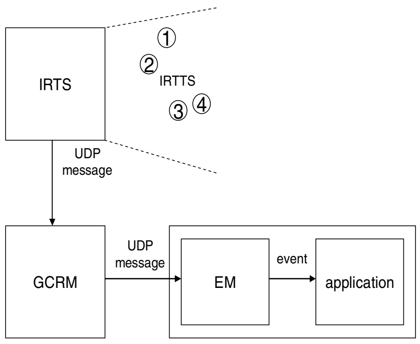

Fig. 1. Schematic Overview

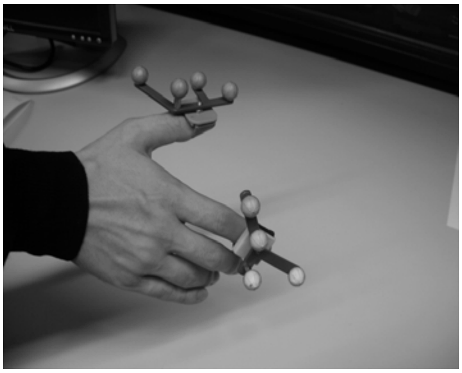

Fig. 2. IRTTs mounted at user's fingers

The ITS delivers information (position and orientation) about IRTTs worn by the user via UDP messages to the client. These IRTTs consist of four reflecting spheres fixed at a flexible tape. This tape ensures wearing comfort for the user. The first IRTT is worn at the thumb and the second at the index finger of the right hand (see Fig. 2.). The third and fourth is worn at the left hand arranged in the same way.

The Gesture Caption and Recognition Module (GCRM) acting as client receives this information and classifies observed data into static and dynamic gestures. Static gestures, such as pointing, can efficiently be classified by applying an Euclidean Distance measure. If the absolute mean difference of at least two position vectors is above a certain threshold, the recording of a dynamic observation is started. It is stopped if this difference is under the threshold again. In order to smooth the observation with variable length, a spline interpolation is applied. Hereafter an unknown gesture can be identified by finding the Hidden Markov Model (HMM) with the highest likelihood [5].

For the training phase of the above mentioned reference vectors and HMMs, a set of several samples from ten people has been gathered. Thus, the resulting gesture recognition system can be considered as person independent. Due to its low dimensionality and marginal preprocessing the entire recognition is running in real-time and has a very low latency so that it can be used on-line within the demonstrator. Any recognized gesture is sent as an UDP message to the Event Manager (EM) acting as master, implemented as a C# class. Depending on to recognized gesture, the EM raises an event, not only containing information about the gesture but also its confidence. This event can be processed by any connected application.

3.1 Gesture Caption and Recognition Module

The GCRM is the core recognition module receiving its data from the ITS tracking system and sending the decoded gestures to the EM via network (see Fig. 1).

As a consequence of the obeyed IRTT system, the fingers' positions and angles can be observed directly, i.e. the x-, y- and z coordinate as well as the three angles roll, azimuth and elevation. Thus no feature extraction technique has to be applied enabling a fast preprocessing. However, aiming at robust features including position and velocity, a Hermitian Spline Interpolation (HSI) is performed on the measured tracking data.

Besides a feature smoothing, this step is motivated in order to fill an invalid or missing tracking feature $P_{\text{interp.}}$ mainly caused by occluded infrared markers. These faulty observations arise when not both fingers' targets are visible in more than one camera. Otherwise these faulty observations would harm the confidence of the subsequent recognizers.

Aiming at filling a hole by reconstructing a valid gesture trajectory, a curve between the ends of the last visible observations is computed on the basis of the two points (P_1, P_2) and their tangents (T_1, T_2).

The velocity can also be reconstructed correctly, since a distance change is represented by the lengths of the tangents.

The HSI bases on a linear combination of four cubical elemental functions with start and end points and can be expressed in the following matrix expression, where s ranges from 0 in P_1 to 1 in P_2 :

$$P_{\text{interp.}} = \begin{pmatrix} s^3 \\ s^2 \\ s \\ 1 \end{pmatrix} \bullet \begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \bullet \begin{pmatrix} P_1 \\ P_2 \\ T_1 \\ T_2 \end{pmatrix} \text{ with } s \in [0.0 - 1.0]$$

Since gestures are represented by their dynamic properties, the absolute position or rotation of the finger or hand is not significant and carries no further information. Therefore the feature vector has to be normalized by transforming it into the global origin. This can be achieved by deriving the relative position and angle change of all further samples with respect to the first sample.

In a next step this improved feature stream has to be segmented and classified into static and dynamic gestures. In order to decide if a relevant dynamic gesture starts, respectively ends, a threshold based decision is derived from the continuous data stream. If the magnitude of the difference of two previous positions is above the speed threshold $T_{\text{speed}} = 18$ cm/s at least the next $T_{\text{duration}} = 350$ ms are considered to be a separate feature. When the gesture velocity is below the threshold T_{speed} the segment is assumed to have ended.

3.1.1 Dynamic Gestures

Continuous classical left to right Hidden Markov Models (HMMs) with their excellent dynamical time warping capabilities and recognition performance are utilized to handle dynamic gestures [6]. With this paradigm the robust recognition of gestures is guaranteed no matter how fast or slow they are expressed.

An arbitrary HMM λ representing one certain gesture class is completely described by its number J of internal emitting states q_j, a state transition matrix (a_{ij}) including the non emitting start and end state (q_0 and q_{J+1}):, and the (continuous) production probability vector $b = [b_1 ... b_J]^T$.

The elements of the matrix A, $a_{q_j q_{(j+1)}}$ represent the probabilities of being in state $q_{(j+1)}$ after having been in state q_j (1st order Markov Model).

The elements b_j in a certain state j for a D-dimensional observation $\vec{x}_j$ are given by a multivariate Gaussian distribution consisting of a mean value vector $\vec{\mu}_j$ and a covariance matrix $\sum_j$: $b_j\left(\vec{x}_j, \vec{\mu}_j \sum_j\right) = \frac{1}{\sqrt{(2\pi)^D \left|\sum_j\right|}} e^{-\frac{1}{2}(\vec{x}_j - \vec{\mu}_j)^T \sum_j^{-1} (\vec{x}_j - \vec{\mu}_j)}$, describing the probability of a given observation or feature $\vec{x}$ being in a certain state q_j.

In our case, a dynamic gesture is represented by an observation sequence X . This feature sequence X has to be at least a piecewise stationary signal and consists of the single observations or feature elements $X = [\vec{x}_1 ... \vec{x}_T]$.

The unknown parameters in A and $\vec{b}$ have to be estimated prior to the recognition process. For this purpose the well-known Baum-Welch-Estimation procedure [6] can be applied together with an appropriate amount of positive examples, here 30 from 10 different subjects for each class.

An unknown gesture can be classified by the following maximum-likelihood decision using all previously trained models λ: $\lambda^* = \underset{\lambda \in \text{GESTURES}}{\arg\max}\ P(X|\lambda)$.

Herein X represents the unknown gesture, λ one class out of the set of all prior trained gestures and λ^* the best matching model out of this set, which is the winner.

3.1.2 Static Gestures

Static gestures, such as pointing or grasping, are represented by the distance and the angle between the thumb and the index finger. They can efficiently be classified by applying the Euclidean Distance measure.

In order to train the prior defined set of classes, the static finger positions of several subjects are captured. The class is then represented by the mean of these reference examples. Unknown vectors can be classified by finding the class with the minimal distance to it.

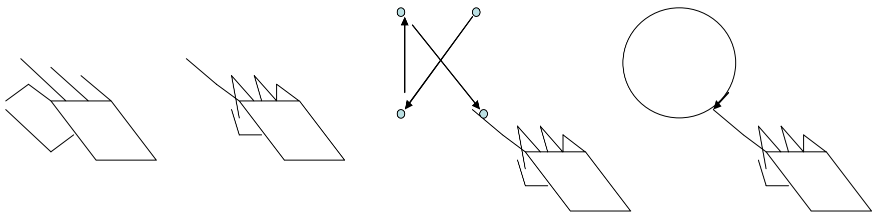

Fig. 3. Gestures used in our recognition system (grasp, point, dynamic X, dynamic O)

3.2 Event Manager

The EM is implemented in form of a C# class, which can be integrated in any application. The EM provides an event driven architecture as well as general functions and properties concerning the gesture recognition. High level functions, such as connecting to the GCRM, are provided for easy use of any developer. If a UDP connection between EM and GCRM is established, any information of the GCRM is sent by a UDP message to the EM.

Thus, the application is able to connect to events raised by the EM, if any gesture is recognized by the GCRM. Any gesture of the left or right hand (static and dynamic) as well as any position and orientation of any finger can be retrieved by the application in real-time.

Basing on this data the subsequent application logic can be controlled by our gesture recognition system.

4 Evaluation

Aiming at evaluating our system we performed a short system evaluation, which focused on the comparison of three interaction paradigms: interacting in the real world, AR with a mouse/keyboard, and AR with our gesture recognition system. Our main attention was to compare the execution time of similar tasks, the intuitiveness of the underlying system and the interaction comfort. In order to keep the evaluation straightforward, we decided, to only examine interaction by using the grasping gesture.

4.1 Evaluation Setup and Procedure

For testing our developed recognition system we designed a demonstration application, which integrates the system capabilities and acts as an evaluation procedure.

This demonstration application offers virtual building blocks integrated in AR (similar to kid's toys, see Fig. 4.). As in a real kit our virtual kit offers different building blocks, which can be used to compose a complex figure.

Manipulation of real building blocks is done by using hands, thus our system can be used in a reasonable way. All virtual building blocks can be manipulated by using the real hand simulating manipulation of a real kit in a realistic way.

This virtual kit consists of eight virtual building blocks separated in four different kinds of building blocks. At the beginning of the demonstration application these blocks are arranged side by side. All blocks can be grasped and displaced by using the grasping gesture of our system. At the moment of grasping a block is bound to the hand adopting movement and orientation of the hand. After releasing, the block is bound again to the world coordinate system. In that way, manipulation of each building block is possible. As the real kit, our virtual kit offers the opportunity to create complex models by using the hands only. All building blocks can be reset by use of the dynamic gesture “O”, in case of deleting all blocks, gesture “X” can be used. For a better depth perception the fingers are covered with occlusion models. In that way, the user is able to differentiate hand position before and behind the virtual object (see fig. 5.).

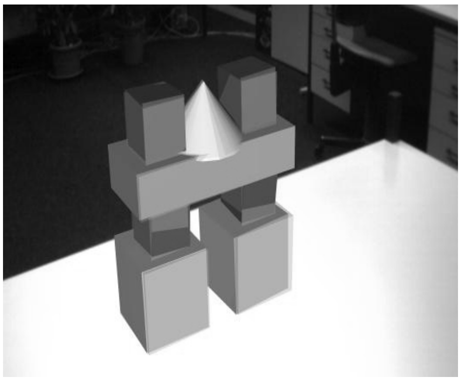
Fig. 4. Virtual building blocks composing a complex model

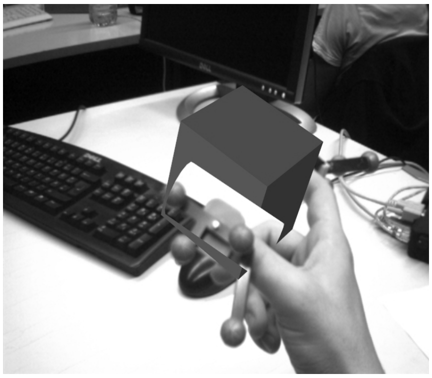
Fig. 5. Grasped virtual building block with occluding thumb

For our evaluation, the demonstration application was limited to grasping only, since there is no possibility for resizing the real building blocks or the manipulation by dynamic gestures (e.g. deletion).

For a comparison of the three interaction paradigms we have chosen the virtual building blocks application. Basing on a real kit with building blocks, used for the part of interaction in reality, virtual models of these blocks were created (see fig. 4.). These models were loaded into the AR application. The subject sat in front of a desk on which the real respectively the virtual blocks were arranged. During the start the position of the real and virtual blocks was the same in order to assure the same conditions for every test run. The first tested system was the interaction in reality, thus, no additional hardware had to be worn by the user. The second system was AR using a mouse/keyboard for interaction. This system offered grasping by clicking a button and translating by moving the mouse. For differentiation of the three axes, the mouse input was combined with keyboard input in order to distinguish all three axes. Visualization was done by a Head-Mounted Display (HMD) worn by the test person. The third system was our gesture system demanding additional targets, which have been placed on the thumb and index finger of the user. Visualization was also done using a HMD.

At the beginning of the test, a short general introduction to AR was given to each subject. The users had five minutes to experience the functionality of each system and get comfortable with wearing a HMD and the targets mounted at their fingers.

The second part was the core evaluation task, which consisted of arranging the building blocks to a predefined model. This task had to be performed consecutively with all three different systems by the user while the needed execution time was recorded. After each part, a short questionnaire concerning a rating of intuitiveness and comfort had to be filled out by the subject.

4.2 Results

15 subjects, with 10 males among them, participated in our study. The average age was 25 years. The mean values of their ratings and task execution times are summarized in Tab. 1.

Table 1. Subjects' mean rating and task execution time

	Reality	Mouse/Keyboard	Gesture Recognition
Task Execution time in [s]	9	89	57
Intuitivness	5	1,8	4
Comfort	5	1,9	1,5

The results show that the fastest way to solve the given task is the interaction in reality. The average task duration time was 9 seconds. Interaction by mouse/keyboard turned out to be slower than interaction with our gesture recognition system. Average time using our system was 57 seconds, whereas mouse/keyboard required 89 seconds.

The test persons rated the interaction in reality as the most intuitive way to solve this task with a score of 5. Our gesture recognition system was rated with a score of 4, which states that gestures are more intuitive than using the mouse and keyboard.

Furthermore, reality turned out to be the most comfortable way of interaction. Interacting with mouse/keyboard and our gesture recognition system was rated rather low with 1.9 and 1.5 points.

These results show that using our gesture recognition system lowers the average task duration time by a third for this given task compared to the mouse/keyboard. However, this enhancement comes along with a lack of comfort, caused by additional hardware, which has to be worn by the user (IRTTs and HMD). As expected, the interaction in reality is still by factor six much faster than interaction in an augmented environment. Additionally it has turned out that the interaction in reality is the most intuitive and comfortable way of solving the given task.

5 Conclusion and Future Directions

In this contribution we presented an automatic gesture recognition system. This system is able to recognize static gestures (e.g. pointing or grasping) as well as dynamic gestures (e.g. drawing letters in the air). Basing on a master-client structure the gesture caption and recognition module receives tracking data of a connected infrared tracking system originated from Augmented Reality applications. This combination enables an easy integration into Augmented Reality. The user wears two light weighted infrared tracking targets at his thumb and index finger. Based on these captured data, which include the position and orientation of the targets, a feature vector is gained by a subsequent hermitian spline interpolation. The recognition module classifies unknown static gestures by calculating the nearest neighbors or Hidden Markov Models for the classification of predefined dynamic gestures.

Our presented system was benchmarked by a short evaluation procedure based on a construction task focusing on a comparison of the following three interaction paradigms: reality, Augmented Reality using a mouse/keyboard, and our developed system. The valuated parameters were average task execution time, intuitiveness, and comfort of interaction. As expected the results of this study proved that interaction in reality is the fastest, the most intuitive, and the most comfortable way of interaction. Using our gesture recognition system the average task duration time was lowered by a third compared to interaction by mouse/keyboard. It further increases the intuitiveness of the construction task. However, this enhancement comes with a slightly lowered wearing comfort, caused by additional hardware that has to be worn by the user. Therefore our presented way of human-machine interaction is the most preferable way to be used within AR applications.

References

1. Buchmann, V., Violich, S., Billinghurst, M., Cockburn, A.: FingARtips: gesture based direct manipulation in Augmented Reality. In: GRAPHITE ’04: Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia (2004)
2. Advanced Realtime Tracking GmbH: ARTtrack1 (2005) http://www.ar-tracking.de

3. Stoerring, M., Moeslund, T.B., Liu, Y., Granum, E.: Computer Vision-Based Gesture Recognition for an Augmented Reality Interface. In: 4th IASTED International Conference on VISUALIZATION, IMAGING, AND IMAGE PROCESSING (2004)
4. Kaiser, E., Olwal, A., McGee, D., Benko, H., Corradini, A., Li, X., Cohen, P., Feiner, S.: Mutual disambiguation of 3D multimodal interaction in augmented and virtual reality. In: ICMI '03: Proceedings of the 5th international conference on Multimodal interfaces (2003)
5. Wallhoff, F., Zobl, M., Rigoll, G.: Action Segmentation And Recognition in Meeting Room Scenarios. In: Proceedings on IEEE International Conference on Image Processing (2004)
6. Rabiner, L.R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In: Proceedings of the IEEE, vol. 77(2) (1989)