Multi-touch Gestural Interaction in X3D using Hidden Markov Models

Sabine Webel* Jens Keil† Michael Zoellner‡

Fraunhofer Institut für Graphische Datenverarbeitung
Darmstadt, Germany

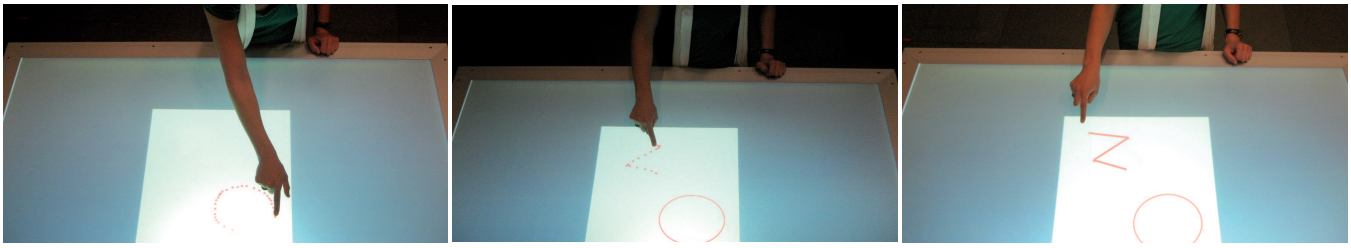

Figure 1: *Painting application in X3D: recognized paintings created with gestural input were redefined.*

Abstract

Multi-touch interaction on tabletop displays is a very active field of todays HCI research. However, most publications still focus on tracking techniques or develop gesture configurations for a specific application setup using a small set of simple gestures. In this work we present a new approach to easily set up the recognition of even complex gestures for multi-touch applications. Our gesture recognition module is based on Hidden Markov Models, which offer a robust recognition of multiple gestures in real-time. An X3D interface of the recognition module is provided to qualify designers and other non-programmers to apply gesture recognition functionalities to multi-touch applications in an easy and straightforward manner.

CR Categories: H.5.2 [Information Systems]: INFORMATION INTERFACES AND PRESENTATION—User Interfaces

Keywords: multi-touch, multi-user, tabletop interaction, gesture recognition, X3D

1 Introduction

Interactive direct-touch tabletop displays have been the focus of numerous research projects and become increasingly important in the area of Human-Computer-Interfaces (HCI). They were utilized in various application areas and have to comply with different requirements, such as the handling of multi-touch and multi-user functionalities. However, it is obvious that also the corresponding applications may become increasingly complex and different subjects have to be taken into account, like e.g. finger and gesture tracking and recognition, software setup and implementation, and also more sophisticated graphical interfaces and interaction principles as well. Previous research on gestural interaction in the area of virtual environments and pen-based input has either focused on single-user or even single-touch, or apply multi-touch by using a very limited set of simple ”common” gestures, such as moving an object around by touching it and moving the finger, scaling the object by stretching two fingers apart and rotating it by describing an angle with the two touching fingers [Ringel et al. 2004].

This work focuses on the gesture recognition of complex gestures by using Hidden Markov Models (HMMs) [Rabiner 1989]. We present a single and multi-user interface to recognize multiple gestures simultaneously; the existing X3D ISO standard is extended to enable less experienced users to apply gesture recognition functionalities in single and multi-touch environments. It is also shown that our approach can identify non-meaningful input gestures.

2 HMM-based Gesture Module

Although originally developed for problems in speech recognition, HMMs have become a popular approach to gesture recognition [Kim et al. 2007; Liu and Lovell 2003] as well. In general, gestures performed by human typically vary in speed, intensity of the movement, and exact location of gesture performance; even if one person repeats the same gesture several times. Thus, the accurate repetition of a movement is practically impossible. HMMs provide a probabilistic framework that can account for dynamically time-varying gesture sequences. Several topologies of HMMs exist under which the so called left-right model or Bakis model [Bakis 1976] is particularly suited for gesture recognition. As time increases the state index of a left-right model increases (or stays the same), i.e. the states proceed from left to right. Hence, left-right models are well suited for the modelling of signals whose properties change over time, what applies to gestures. Every gesture is represented by a discrete left-right HMM. During a training phase example sequences are collected and used to create these models and to determine the model parameters. Thus, a trained model represents a gesture as it is most likely performed by a human. Each HMM is composed of several states which represent different sections of the underlying observation sequences. A gesture in this work is described as a spatiotemporal sequence of feature vectors, which specify the moving direction of the gesture, no matter of how the gesture is processed (e.g. by mouse, finger, etc.). For each gesture, we choose a segmentation depending on the complexity of the

*e-mail:sabine.webel@igd.fraunhofer.de
†e-mail: jens.keil@igd.fraunhofer.de
‡e-mail:michael.zoellner@igd.fraunhofer.de

gesture described by directional and motional changes. This segmentation is used as a first indication to determine the number of states of the gesture HMM and to estimate the initial model parameters. The final number of states is ascertained experimentally.

2.1 Filtering, Training and Recognition

To smooth the input sequences and to eliminate outliers caused by closely consecutive observation symbols, first of all a proximity filter, that filters objects whose distance to the antecessor is under a specified threshold, is applied to every sequence. Since we use HMMs with a discrete alphabet of observation symbols, we need a vector quantizer to map each input sequence onto discrete codebook indeces. The underlying codebook consists of 16 vectors, what is enough to largely cover every moving direction of a gesture with passable computational costs. Using the well known *k-means clustering algorithm* the codebook is trained with the collected training input gestures. Once the codebook has been obtained, the mapping between the input gestures represented as angle sequences and the codebook indices become a simple nearest neighbour computation.

In the training phase sample gestures performed by 8 people at a multi-touch table were collected. Every person performed each gesture about 20 times. The collected sample gestures are used to train the codebook and the gesture HMMs. In the HMM training, the model parameters are adjusted to maximize the probability of the observation sequences of the sample gestures. To make the resulting gesture HMMs accessible within an X3D environment we introduce a novel node called "Gesture" with the following interface:

```
Gesture : X3DNode {
  SFString  [in,out]  label  ""
  SFInt32   []        numStates  0
  SFInt32   []        vocabSize  0
  MFInt32   []        vocab  []
  SFInt32   []        delta  -1
  MFDouble  []        initialStateProbs  []
  MFDouble  []        transitionProbs  []
  MFDouble  []        emissionProbs  []
  MFInt32   [in]      sequence
  SFDouble  [out]     forwardProb
}
```

The *label* field defines a label for the gesture. The possible output symbols are given by *vocab*. *Delta* defines the number of jumpable states in the HMM, a value of *-1* denotes no restriction. *ForwardProb* returns the forward probability, i.e. the likelihood, of the observation sequence applied to *sequence*. The remaining fields contain the reestimated model parameters.

Besides the creation of the gesture HMMs, the design of the recognition process is the second important point in gesture recognition. Since we have only a limited set of relevant gestures and an infinite large amount of irrelevant input, we need a recognition model that can identify non-meaningful sequences as well. Due to this reason our gesture recognition model is setup on an HMM-based threshold model introduced by [Lee and Kim 1999], that yields the likelihood value of a gesture model to be used as a threshold. This approach uses the *internal segmentation property*[1] of HMMs. The threshold model is an ergodic HMM that consists of state copies of all trained gesture models. It yields positive matching results with the patterns generated by the combination of subsequences of the predefined gestures in arbitrary order. However, the likelihood of a correctly performed gesture is higher for the dedicated gesture HMM because the temporal order of subpatterns is better described in this model (see [Lee and Kim 1999] for more details). Hence, the likelihood of the threshold model can be utilized as an adaptive threshold. To make the gesture recognition functionality accessible for X3D application developers, we introduce the "GestureRecognitionModel" node with the following interface:

[1]The state transitions of a trained HMM represent subpatterns of the proper gesture and their sequential order.

```
GestureRecognitionModel : Gesture {
  SFFloat   []        proxFilterThreshold  1
  MFDouble  []        codebook  []
  MFNode    []        gestures  []
  SFFloat   [in,out]  modelTransitionProb  1
  SFBool    [in,out]  rotationAllowed  TRUE
  MFVec3f   [in]      data
  MFString  [out]     result
  ...
}
```

ProxFilterThreshold defines the threshold for the proximity filter, *codebook* contains the codebook used for the vector quantization. The "Gesture" nodes of the gestures to be recognized must be placed in *gestures*. *RotationAllowed* defines, whether the gestures are rotation invariant. Multiple gesture sequences can be passed to the "GestureRecognitionModel" node using the *data* field. The node can be placed directly inside the X3D scene. On initialization, a recognition model described above is created and the recognition process starts for all incomming input sequences. How the input sequences are generated remains to the application developer.

3 Conclusion

We have implemented the presented HMM-based approach for multi-touch gesture recognition for X3D environments within the context of the *instantReality* framework [Behr et al. 2007]. Two new nodes have been introduced, which provide a simple and clear defined interface that even qualifies non-programmers to access the gesture recognition module within an X3D scene. To demonstrate the functionality we created an X3D painting application. The user or multiple users, can paint on a canvas and if the resulting gesture shape is recognized, the drawn shape is smoothed. This application is tested on a computer's desktop and on a multi-touch table as well. Recognition rates of about 93% with mouse input and 97.5% for multi-touch table input were achieved; notice that all training data was collected on the touch table. We expect the recognition rates to be further improved by training the gesture models with data collected using various devices.

References

BAKIS, R. 1976. Continuous speech word recognition via centisecond acoustic states. In *ASA Meeting*.

BEHR, J., DÄHNE, P., JUNG, Y., AND WEBEL, S. 2007. Beyond the web browser - x3d and immersive vr.

KIM, D., SONG, J., AND KIM, D. 2007. Simultaneous gesture segmentation and recognition based on forward spotting accumulative hmms. *Pattern Recogn. 40*, 11, 3012–3026.

LEE, H.-K., AND KIM, J. H. 1999. An hmm-based threshold model approach for gesture recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 21*, 10, 961–973.

LIU, N., AND LOVELL, B. C. 2003. Gesture classification using hidden markov models and viterbi path counting. In *DICTA*, 273–282.

RABINER, L. R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE 77*, 2, 257–286.

RINGEL, M., RYALL, K., SHEN, C., FORLINES, C., AND VERNIER, F. 2004. Release, relocate, reorient, resize: fluid techniques for document sharing on multi-user interactive table. ext. In *Abs. of the ACM Conference on Human Factors in Computing Systems*, ACM Press, 1441–1444.

Citation

Webel, Sabine; Keil, Jens; Zöllner, Michael:
Multi-touch Gestural Interaction in X3D Using Hidden Markov Models.
In: ACM SIGCHI u.a.:
Proceedings VRST 2008 : ACM Symposium on Virtual Reality Software and Technology.
New York : ACM, 2008, pp. 263-264