Journal of Computational Science 10 (2015) 66–76

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

A new approach for touch gesture recognition: Conversive Hidden non-Markovian Models

Tim Dittmar*, Claudia Krull, Graham Horton

Otto-von-Guericke-University Magdeburg, P.O. Box 4120, 39016 Magdeburg, Germany

ARTICLE INFO

Article history:
Received 15 March 2014
Received in revised form 3 November 2014
Accepted 5 March 2015
Available online 16 March 2015

Keywords:
Touch gesture recognition
Hidden Markov models
Conversive Hidden non-Markovian models
Proxel

ABSTRACT

With the current boom of touch devices the recognition of touch gestures is becoming an important field of research. Performing such gestures can be seen as a stochastic process, as there can be many little differences between different executions. Therefore stochastic models like Hidden Markov Models have already been applied to gesture recognition. Although the modelling possibilities of Hidden Markov Models are limited, they achieve an acceptable recognition quality. But they have never been tested with gestures that only differ in execution speed.

We propose the use of Conversive Hidden non-Markovian Models for touch gesture recognition. This extension of Hidden Markov Models enhances the modelling possibilities and adds timing features. In this paper, two touch gesture recognition systems were developed and implemented based on these two model types. Experiments with a set of similar gestures show that the proposed model class is a good and competitive alternative to Hidden Markov Models.

1. Introduction

1.1. Background

Due to the big success of smartphones and tablets, a ubiquitous presence of multi-touch devices is establishing itself around the world. While the multi-touch input method offers manifold possibilities for controlling devices, almost all of them are usually controlled by using a fixed set of simple gestures like tap, drag and pinch. Such systems can be realized quite easily using heuristics, but they are not very flexible.

To create a gesture recognition system using heuristics requires that the system has all of the gestures implemented *a priori*. The subsequent addition of a new gesture could make code adaptations of the previously implemented gestures necessary, especially when the gestures are very similar. In order to create a more flexible gesture recognition system, other methods need to be used that enable the definition of a gesture by performing several examples. This way the user of a touch device could define personalized gestures for each action.

A flexible multi-touch gesture recognition system was presented by Damaraju and Kerne [1]. This system is based on Hidden Markov Models (HMM) and creates one HMM for each gesture using sample inputs. While there are other pattern recognition methods that are deployed for touch gesture recognition, the current work focuses on hidden models, namely HMM and a quite new model class: Conversive Hidden non-Markovian models (CHnMM).

This new model class is an extension of HMMs and therefore should also be able to be the basis of a flexible gesture recognition system with an open gesture set. But in contrast to HMMs, CHnMMs are able to incorporate explicit timing information. As a result, it should be possible to create more precise models of gestures, so that even very similar gestures can be distinguished. However, their computation is more complex and could render real-time applications infeasible. With this work, we want to evaluate whether CHnMMs are applicable for touch gesture recognition and how they perform in comparison to HMMs.

1.2. Motivation and goals

The idea to use non-Markovian Models for gesture recognition was first brought into consideration by Bosse et al. [2]. The goal was to show that a system based on Hidden non-Markovian models (HnMM) could distinguish gestures that are similar in shape

* Corresponding author. Tel.: +49 15114441120.
E-mail addresses: tim.dittmar@ovgu.de (T. Dittmar), claudia.krull@ovgu.de (C. Krull), graham.horton@ovgu.de (G. Horton).

http://dx.doi.org/10.1016/j.jocs.2015.03.002
1877-7503/

but differ in execution speed. This feature had not been considered for HMM gesture recognition systems before. For that reason an HMM- and an HnMM-system were developed to recognize gestures performed with a Nintendo Wiimote and both systems were compared in their recognition quality.

Inspired by this, we attempt to apply a similar approach to touch devices. The general goal of this work is to find out whether CHnMMs are applicable for touch gesture recognition. Therefore the CHnMM-system needs to reach similar or better recognition rates than the HMM-system. In addition, the time needed for recognizing the gesture has to be competitive, so that the system could be used in real-time scenarios.

In order to evaluate the recognition quality, the metrics Precision and Recall are used. These metrics give a measure of the quality of the recognition results and a measure for the completeness. These metrics are appropriate, because it is possible that a performed gesture is not classified by the recognition system. A high level of Precision will be needed to support the claim that a CHnMM-system can distinguish between similar gestures. We therefore aim for a Precision that is at least 85% and that exceeds the HMM-system Precision value. Furthermore, a minimum Recall value of 70% should be reached.

Since CHnMMs have never been used in practical applications before, our goal is to merely to establish the plausibility of our approach, not to develop a ready-to-use touch recognition system.

2. Background information

2.1. Previous work

In this section, we summarise the methods on which our algorithm is based and introduce the terms and models that will be used.

In the context of this work, the term hidden models stands for all model classes that are able to infer conclusions from so-called partially observable discrete stochastic (PODS) systems [3, p. 1]. This is a special class of real-world problems where a stochastic description of the system behaviour is known, but the discrete states of the system cannot be directly observed. Instead, observable signals are created by the system that give clues about its inner state.

2.1.1. Hidden Markov models

One of the first types of hidden model that was formalized and applied to recognition problems is the Hidden Markovian model (HMM). A very good overview of its elements and how it can be applied to speech recognition is given by Rabiner in [4]. Today, HMMs are used in many fields of pattern recognition such as speech recognition, handwriting and gesture recognition and DNA analysis [5, pp. 9–29]. An HMM can be described mathematically and the notation is quite consistent across the literature (e.g. [3–6]), with the exception of some subtleties. The following paragraphs define the notations that are used in this article.

The two basic elements of an HMM are:

- a finite set of N possible states $S = \{s_1, s_2, \ldots, s_N\}$, $N \in \mathbb{N}$
- a finite set of discrete outputs (also called symbols) $V = \{v_1, v_2, \ldots, v_M\}$

For this work, a *trace* is defined as a simple sequence of T observations $O = o_1, o_2 \ldots o_T$ with $o_t \in V$. Furthermore, $q_t \in S$ denotes the state of the system after the tth symbol emission and a *path* $Q = q_1, q_2 \ldots q_T$ is a possible sequence of traversed internal states of an HMM. With these definitions the additional elements of an HMM can be defined:

- a quadratic matrix of state-transition probabilities

$$A = \{a_{ij} | P(q_{t+1} = s_j | q_t = s_i)\}, A \in \mathbb{R}^{N \times N}$$

- a *initial probability vector* Π of start probabilities

$$\Pi = (\pi_0, \pi_1, \ldots, \pi_N), \pi_i = P(q_0 = s_i), \Pi \in \mathbb{R}^N$$

- a N-by-M matrix of state specific output probability distributions

$$B = \{b_i(v_k) | b_i(v_k) = P(o_t = v_k | q_t = s_i)\}$$

As a result, an HMM – usually denoted by λ – can be represented as a tuple $\lambda = (S, V, A, B, \Pi)$, or $\lambda = (A, B, \Pi)$ for short, since the sets S and V are merely names for states and symbols that do not affect the model behaviour.

The sets S and A define a Markov chain that fulfils the Markov property, i.e. the behaviour of a system or process at a given time t depends only on the previous state. This can be expressed as follows:

$$P(q_t | q_1, q_2, \ldots, q_{t-1}) = P(q_t | q_{t-1}), q_t \in S$$

The internal states are assumed to be unobservable, thus the Markov chain represents the *Hidden* part of a *Hidden* Markov model. The only part that is observable are the outputs or symbols of a system or process o_t. The HMM gives a stochastic description of how a PODS system or process creates these symbols by assuming that in every state a certain probability for creating a certain symbol is present, which is expressed in matrix B. Since an HMM is processed in a step-wise manner, it is also assumed that the system or process creates a symbol in each step. In most cases, the step size is a discrete interval δt that describes the time between two outputs/symbols emitted by the process or system. When such a system is observed, these signals are collected in a trace O that could also be interpreted as a log of the system.

There are three basic problems that are of interest when using HMMs: Evaluation, Decoding and Training [3–5], which all involve an observed trace O of a PODS system or process and an HMM λ that represents that system or process.

The evaluation problem is the determination of the probability that a given trace O is the result of a system or process described by the HMM λ or formally $P(O|\lambda)$. It “is the most widely used measure for assessing the quality with which an HMM describes the statistical properties of certain data” [5, p. 78].

The calculation of this measure could be naively done with a “brute force” approach where every possible *path* is determined and the probability of emitting the *trace O* along each *path* is calculated and summarized. Since this approach has an exponential complexity of $O(TN^T)$, it is impractical for most use cases. Instead, a more efficient method is used: the *forward algorithm*. This is based on the *forward variable* $\alpha_t(s_i)$ which represents the probability that a given model λ generated the given trace up to o_t and that state i has been reached:

$$\alpha_t(s_i) = P(o_1 o_2 \ldots o_t \cap q_t = s_i | \lambda)$$