Available online at www.sciencedirect.com

ScienceDirect

Speech Communication 49 (2007) 317–330

SPEECH COMMUNICATION

www.elsevier.com/locate/specom

Multisyn: Open-domain unit selection for the Festival speech synthesis system

Robert A.J. Clark *, Korin Richmond, Simon King

CSTR, The University of Edinburgh, 2 Buccleuch Place, Edinburgh EH8 9LW, UK

Received 12 June 2006; received in revised form 25 January 2007; accepted 25 January 2007

Abstract

We present the implementation and evaluation of an open-domain unit selection speech synthesis engine designed to be flexible enough to encourage further unit selection research and allow rapid voice development by users with minimal speech synthesis knowledge and experience. We address the issues of automatically processing speech data into a usable voice using automatic segmentation techniques and how the knowledge obtained at labelling time can be exploited at synthesis time. We describe target cost and join cost implementation for such a system and describe the outcome of building voices with a number of different sized datasets. We show that, in a competitive evaluation, voices built using this technology compare favourably to other systems.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Speech synthesis; Unit selection

1. Introduction

Over the last decade, the Festival speech synthesis system (Taylor et al., 1998) has become the de facto standard free toolkit for speech synthesis research. It has also formed the starting point for at least three leading commercial systems.[1]

Until recently, Festival offered two distinct methods for concatenative synthesis: a conventional single-instance diphone-based method using an inventory containing one recording of each diphone type, and the "clunits" method (Black and Taylor, 1997) which uses an inventory of units recorded in natural sentences and performs a restricted form of unit selection.

In this paper, we introduce a third method: a general-purpose unit selection algorithm, along with the tools for building voices. The method is general-purpose because it is capable of realising open-domain voices ("clunits" performs best in limited domains, where the recordings in the inventory are from the same domain – e.g. use the same limited vocabulary and constrained syntax – as the utterances to be synthesised). We call this method "Multisyn" and it can be downloaded as part of Festival 1.95 and above from https://www.cstr.ed.ac.uk.

Unit selection speech synthesis (Black and Campbell, 1995; Hunt and Black, 1996) was proposed as a way to solve some of the problems of unnaturalness introduced by the signal processing techniques needed to produce convincing synthetic speech from a database consisting of a single example of each diphones that occurs in a language. Instead of having one example of each diphone, a number of examples in different contexts are included, and the synthesis process is formulated as a search problem. A search is performed to find the best sequence of diphones (or potentially other sized units). The goal of unit selection speech synthesis is to select a sequence of diphones which requires much less signal processing than standard diphone synthesis, or ideally no signal processing at all.

* Corresponding author. Tel.: +44 131 6511767.
E-mail addresses: robert@cstr.ed.ac.uk, Rob.Clark@ed.ac.uk (R.A.J. Clark), korin@cstr.ed.ac.uk (K. Richmond), Simon.King@ed.ac.uk (S. King).

[1] From Rhetorical Systems (now Nuance), AT&T and Cepstral.

0167-6393/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.specom.2007.01.014

There are a number of important issues to be addressed in a robust and efficient implementation of unit selection, and recent advances have lead to an improved understanding of the process. The first of these involves designing the recording script. Much of this work discusses the use of greedy algorithms to optimally select a script from a very large text corpus, examples include the work by van Santen and Buchsbaum (1997), Bozkurt et al. (2003) and Kominek and Black (2004), whilst other work discusses the theoretical and practical problems of recording the ideal dataset (Möbius, 2001).

Once a dataset has been recorded, it needs to be searched efficiently. The general search method (Hunt and Black, 1996) has been refined (e.g. Conkie, 1999; Taylor, 2000; Bulyko and Ostendorf, 2001) and complemented by other procedures for specific tasks such as limited domain speech synthesis (Black and Lenzo, 2000).

The primary goal of our Multisyn engine is to provide state-of-the-art unit selection speech synthesis within a framework that makes it easy to (semi-automatically) develop new voices, with only limited speech synthesis knowledge.

1.1. Unit selection speech synthesis

A full tutorial on unit selection speech synthesis is beyond the scope of this paper; we refer the reader to (Hunt and Black (1996)). However, we will define the terminology to be used in the rest of this paper.

Unit selection speech synthesis uses a recorded *database* (sometimes called the *inventory*) of speech. This usually consists of recordings of isolated, naturally occurring sentences (e.g. from newspaper text). The inventory along with its associated linguistic annotation is called the *voice*. *Units* are extracted from this database and concatenated to synthesise novel utterances. The unit type may be the same throughout the database (e.g. diphones), or variable (e.g. a mixture of phones, diphones, syllables, etc.). The database should contain multiple examples of each unit type.

To synthesise a novel utterance, a *target* utterance is constructed, which consists of the desired linguistic specification of the utterance: the words, the phone sequence, the syllable boundaries, placement of accents, optionally a pitch contour and segment durations, and so on. The target is constructed from the input text by the language processing front end, which is usually using some combination of rules and statistical models.

A sequence of units taken from different places in the database is then found which best matches this target. This task is performed by the unit selection *engine*. "Best matching" is measured by two costs, summed over the unit sequence. The chosen unit for a given position in the target utterance is selected from a set of available *candidate units* which may be all matching diphones (regardless of context) in the inventory, or may be a subset of those (after some pre-selection has been applied – Section 3.6).

The *join cost* estimates how well two consecutive units will join together in the large number of cases where they were not contiguous in the database and is commonly computed using only acoustic features. The *target cost* measures how well a unit matches part of the target specification, for example in terms of the constituent phones, within-syllable or within-phrase position, and is commonly computed using linguistic features. Since the join and target costs are locally computed, a Viterbi search can be used to efficiently search for the unit sequence that minimises the total cost. The details of how the join and target costs are computed vary from system to system.

1.2. Structure of this paper

Since Festival is primarily a research toolkit, this paper concentrates on explaining how Multisyn satisfies two design goals. The first goal is to provide a stable general-purpose unit selection implementation that is suitable for carrying out further research into unit selection and related techniques. The second goal is to provide the end user with a simple, mostly automatic mechanism to build their own voice for the system, requiring only limited specialist knowledge. As we shall see, this second goal means that there are times when we have employed a simple but robust technique instead of a *potentially* better, but more complex, technique. Particular attention is given to the design decisions and procedures required to build new voices.

In Section 2 we describe the design and implementation of the Multisyn unit-selection engine. The front end processes used with this engine are simply a subset of those used in the standard diphone system so are not described in this paper. We also compare and contrast the Multisyn approach to other approaches. Sections 3 and 4 discuss the requirements for the database and the process of building a voice from it respectively. In Section 5, we address the issue of automatic segmentation to phonetically label recorded speech databases. In Section 6 we discuss speech synthesis evaluation techniques and recent evaluation in which the Multisyn engine has been involved.

2. Multisyn design and implementation

The Multisyn unit selection algorithm implemented in Festival is conventional and reasonably straightforward, and follows the description in Section 1.1.

2.1. Festival's architecture

Festival is modular and uses a simple framework, commonly known as a "blackboard architecture". The system is centred on a common data structure, called the Utterance, which is passed from module to module within the system. Modules either modify existing parts – called Relations – of this Utterance structure, or add new Relations. This architecture allows users to control easily both the sequence of processes in the pipeline of modules (perhaps