Template-Based Continuous Speech Recognition

Mathias De Wachter, Mike Matton, Kris Demuynck, Patrick Wambacq, *Member, IEEE*, Ronald Cools, and Dirk Van Compernolle, *Member, IEEE*

***Abstract*—Despite their known weaknesses, hidden Markov models (HMMs) have been the dominant technique for acoustic modeling in speech recognition for over two decades. Still, the advances in the HMM framework have not solved its key problems: it discards information about time dependencies and is prone to overgeneralization. In this paper, we attempt to overcome these problems by relying on straightforward template matching. The basis for the recognizer is the well-known DTW algorithm. However, classical DTW continuous speech recognition results in an explosion of the search space. The traditional top-down search is therefore complemented with a data-driven selection of candidates for DTW alignment. We also extend the DTW framework with a flexible subword unit mechanism and a class sensitive distance measure—two components suggested by state-of-the-art HMM systems. The added flexibility of the unit selection in the template-based framework leads to new approaches to speaker and environment adaptation. The template matching system reaches a performance somewhat worse than the best published HMM results for the Resource Management benchmark, but thanks to complementarity of errors between the HMM and DTW systems, the combination of both leads to a decrease in word error rate with 17% compared to the HMM results.**

***Index Terms*—Dynamic time warping (DTW), episodic modeling, example-based recognition.**

I. Introduction

THE ADVANCES in large-vocabulary speech recognition systems over the past two decades were largely due to the power and flexibility of the HMM framework. The increases in available training data and the exponential growth in available computing power could be converted directly into more accurate models, essentially by scaling two parameters: number of Gaussians (refining the observation probability estimates) and number of HMM states (reducing granularity of the acoustic–phonetic units). Apart from scaling, other modifications were introduced in an effort to overcome the intrinsic model deficiencies due to the first-order Markov assumption, e.g., the addition of derivatives to the input features and the use of context dependent phones. A last line of major improvements was the development of discriminative training procedures. Whereas in the 1990's HMMs could turn every extra MIPS into lower error rates without much redesign, this is no longer true today. This might be an indication that it is time to switch to a new paradigm capable of exploiting the ever-increasing computational power.

Instead of looking for further modifications to the HMM model, we take a drastically different approach and return to template (example)-based recognition by dynamic time warping (DTW). We discard the model and training procedures altogether and perform recognition straight from the data. DTW was long ago abandoned in favor of HMMs for good reason. The main drawbacks of DTW were the explosion of the search space for continuous recognition tasks and poor speaker-independent performance. An intrinsic advantage of template-based recognition is that we do not need to model the speech process. This is very convenient, since our understanding of speech is still limited, especially with respect to its transient nature. Some aspects of modern HMMs (e.g., context-dependent models and time derivatives) have proven indispensable but are not completely understood [1]. Theoretically solid and complex new designs such as trajectory models (e.g., [2]), dynamic Bayesian networks or graphical models (e.g., [3]) have not produced sufficiently convincing results to supersede standard HMMs. Other clear evidence of our poor understanding of speech and transients in particular is the quality jump achieved by fully concatenative speech *synthesis* by applying the motto: "no modeling, just data." So why not do the same in recognition in a much broader scope? "No modeling of acoustic-phonetics, speakers, acoustic environments, etc.— just data."

When considering the concrete implementation of template-based recognition, it quickly becomes apparent that the classical DTW algorithm with the Euclidean distance used as local distance metric, combined with a simple beam search will not do the job, neither from a performance nor from a computational point of view. Despite our aim to eliminate the acoustic models, just using a (weighted) Euclidean distance in the feature space is not powerful enough given the sparseness of the data in the high-dimensional acoustic space. Consequently, we use class-dependent local distance measures in our system. Using them is straightforward, but estimating suitable parameters proves to be a challenge. The most immediate problem is to keep the search space of a template-based system within bounds. Contemporary reference databases are gigantic, and the DTW search space based on such databases is even more so. Gigabytes of memory and teraflops will help, but will not do the job alone. Our solution is to rely heavily on bottom-up (data-driven) template selection. In psycho-acoustic and physiological literature, bottom-up selection followed by competition (cf. beam search) is the accepted view of human

Manuscript received March 4, 2005; revised December 5, 2006. This work was supported in part by the Fund for Scientific Research Flanders (FWO) under Projects G.0249.03 and G.0260.07 and in part by the IWT in the GBOU program under Project 020192. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mari Ostendorf.

M. De Wachter, K. Demuynck, P. Wambacq, and D. Van Compernolle are with the Speech Processing Research Group, Electrical Engineering Department (ESAT), Katholieke Universiteit Leuven, 3000 Leuven, Belgium (e-mail: mathias.dewachter@esat.kuleuven.be; kris.demuynck@esat.kuleuven.be; patrick.wambacq@esat.kuleuven.be; dirk.vancompernolle@esat.kuleuven.be).

M. Matton and R. Cools are with the Numerical Integration, Nonlinear-Equations, and Software (NINES) Group, Department of Computer Science, Katholieke Universiteit Leuven, 3000 Leuven, Belgium (e-mail: mike.matton@cs.kuleuven.be; ronald.cools@cs.kuleuven.be).

Digital Object Identifier 10.1109/TASL.2007.894524

speech recognition [4], thus validating our approach, but even with considerable bottom-up pruning at decoding time (possibly augmented with extensive pruning of the reference database), we are faced with a search space that is considerably larger than that of a comparable HMM system. Nevertheless, we will show that it is feasible on today's hardware.

The paper is structured as follows. In Section II, we describe the global architecture of our template-based recognition system. In the subsequent sections, we describe in detail those components that are substantially different from commonplace HMM technology. In Section III-A, we develop class-dependent local distance measures, and Section III-B explores speaker and environment adaptation. Section IV discusses the bottom-up template selection procedure, a novel component in the search procedure. Next, Section V is devoted to novel possibilities with respect to unit selection offered by the template framework. Experiments and recognition results on a medium vocabulary task obtained with the template-based system described in this paper are presented in Section VI. Section VII compares the template-based system with HMM recognizers, pointing out strong and weak points and their effect on recognition performance. Section VIII discusses how the system scales to modern large vocabulary tasks. Finally, we draw some conclusions in Section IX.

II. An Architecture for Template-Based Recognition

A. What is a Template?

The term *template* is often used for two fundamentally different concepts: either for the representation of a *single* segment of speech with a known transcription, or for some sort of *average* of a number of different segments of speech. Both types of templates can be used in the DTW algorithm to compare them with a segment of input speech.

Using the latter type has the obvious advantage of reducing the number of templates and being more robust to outliers [5]. However, the averaging is a model building step, which makes it more akin to HMMs than to true example-based recognition.

As we will show later on, our approach is crucially dependent on the fact that a template is not a model but a real occurrence of speech. We will therefore use "template" only in the first meaning, and define it as follows.

Definition 1: A ***template*** is the representation of an actual segment of speech. It consists of the following:

- a sequence of consecutive acoustic feature vectors (or *frames*);
- a transcription of the sounds or words it represents (typically one or more phonetic symbols);
- knowledge of neighboring templates (a template number if no templates overlap);
- a tag with meta-information.

Definition 2: ***Meta-information*** or ***nonverbal information*** is any information we have about the segment of speech apart from the actual acoustic realization, transcription, and acoustic context. Examples are

- speaker characteristics such as gender, dialect region, age, etc.;
- environmental characteristics such as signal-to-noise ratio, type of noise, type of microphone, etc.;
- (especially prosodic) information derived from the sentence or paragraph level such as speaking rate, stress, position in the sentence, intonation, etc.

B. Template-Based Recognition—A Bayesian Approach

Hidden Markov models and the Bayesian recognition paradigm allow the merging of scores from different knowledge sources (acoustic–phonetic and linguistic) in a straightforward and convenient way. While abandoning the HMM framework as such, we do not want to abandon the advantages given by the Bayesian framework in the formulation of our example-based approach. In our further developments, we will frequently rely on the well-known similarities between the Viterbi and DTW recognition paradigms (cf. the Appendix) and use terminology and symbols that will allow for a simple comparison with HMM systems.

In an HMM-based Bayesian recognizer, the goal is to find the most likely string of words $\hat{\mathbf{W}}$ given the data X. This results in the following basic recognition equation:

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmax}} P(\mathbf{W}|X) = \underset{\mathbf{W}}{\operatorname{argmax}} f(X|\mathbf{W})P(\mathbf{W}). \quad (1)$$

In this formula, $P(\mathbf{W})$ is given by the language model, while $f(X|\mathbf{W})$ is given by the acoustic likelihood.

In a Bayesian approach to example-based recognition, both the language model and the acoustic model scores play similar roles as in HMMs. There is a significant difference, however. Making abstraction of pronunciation variants, a word sequence defines a unique HMM state sequence against which the input needs to be scored. In example-based recognition, each sentence can be explained by many different template concatenations $\mathbf{T}$, and hence the recognition equation becomes

$$\hat{\mathbf{W}} = \underset{\mathbf{W}}{\operatorname{argmax}} \sum_{\mathbf{T}} P(\mathbf{W}, \mathbf{T}|X). \quad (2)$$

Again, applying Bayes' rule and dropping the denominator splits the equation in its basic knowledge sources

$$(\hat{\mathbf{W}}, \hat{\mathbf{T}}) = \underset{\mathbf{W}}{\operatorname{argmax}} \sum_{\mathbf{T}} f(X|\mathbf{W}, \mathbf{T})P(\mathbf{T}|\mathbf{W})P(\mathbf{W}). \quad (3)$$

Since summing over all these possible template strings to find the best word sequence is computationally infeasible, we use a Viterbi approximation (cf. [6]), replacing the sum by a maximum.[1] This leads to the following equation:

$$(\hat{\mathbf{W}}, \hat{\mathbf{T}}) = \underset{\mathbf{W}}{\operatorname{argmax}} \underset{\mathbf{T}}{\operatorname{argmax}} f(X|\mathbf{W}, \mathbf{T})P(\mathbf{T}|\mathbf{W})P(\mathbf{W}). \quad (4)$$

The term $f(X|\mathbf{W}, \mathbf{T})$ in (4) expresses the acoustic likelihood of the input given a template string and a word string. The word string $\mathbf{W}$ in the latter term can be dropped, since $\mathbf{W}$ may be assumed conditionally independent of X, given $\mathbf{T}$. The resulting

[1]An alternative approach is to use the sum over a small number of well-matching templates to increase robustness against the stochastic variability within a group of templates. This was not investigated in this paper.

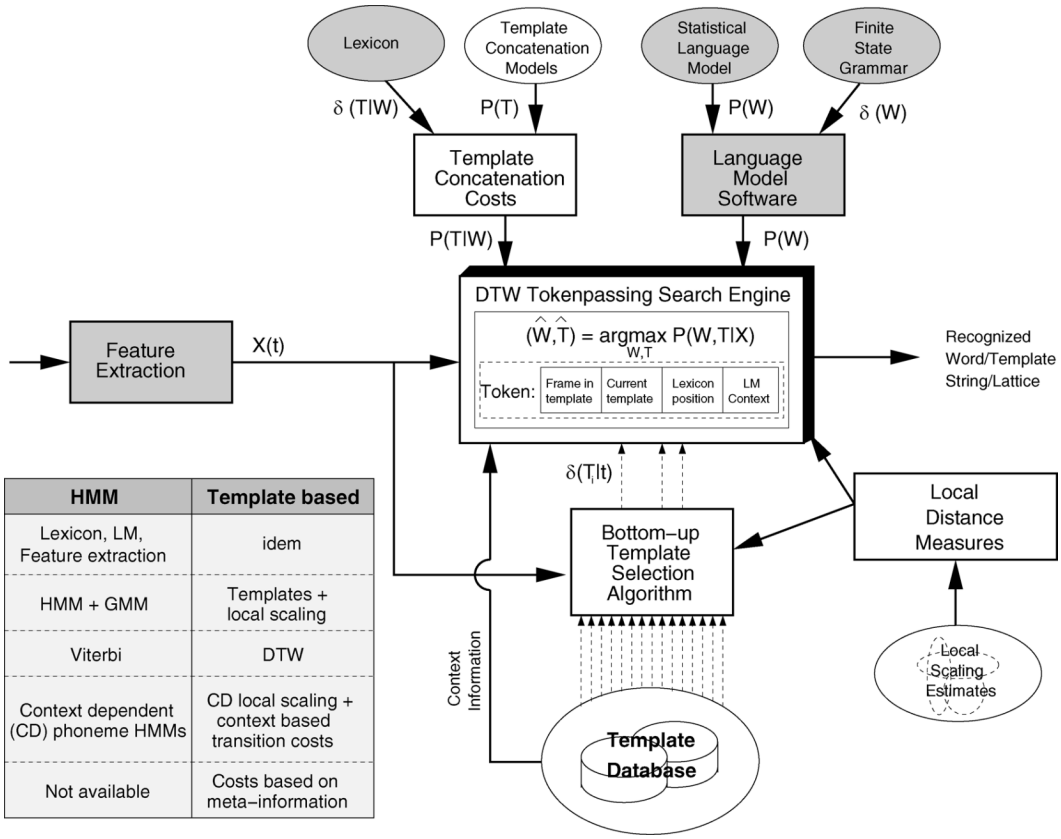

Fig. 1. Architectural overview. The shaded boxes and ellipses correspond to components that have identical counterparts in typical HMM speech recognizers. The others are discussed in the text. The table on the bottom left summarizes the correspondence between HMM and template-based recognizers.

term $f(X|\mathbf{T})$ is a similarity score between the input and a concatenation of templates of real speech, and is easily calculated using the DTW algorithm.

As discussed in the Appendix , in order to compare DTW and HMM systems, it is useful to take the negative logarithm of all probability terms, and it is reasonable to write $-\log(f(X|\mathbf{T}))$ as $D_{\text{dtw}}(X;\mathbf{T})$. The resulting practical basic equation for our system thus becomes

$$(\hat{\mathbf{W}},\hat{\mathbf{T}}) = \underset{\mathbf{W},\mathbf{T}}{\operatorname{argmin}}\{D_{\text{dtw}}(X;\mathbf{T}) - \log(P(\mathbf{T}|\mathbf{W})) - \log(P(\mathbf{W}))\}. \quad (5)$$

While acoustic ($D_{\text{dtw}}(X;\mathbf{T})$) and language model ($P(\mathbf{W})$) scores are present as in a typical HMM recognizer, a third term has appeared. $P(\mathbf{T}|\mathbf{W})$ is the (*a priori*) probability of the template string, given the word transcription. Section V discusses this further.

C. Knowledge Sources

Fig. 1 shows a schematic overview of our example-based recognition system. Where applicable, the arrows are labeled with the information based on (5). A further distinction is made between the truly probabilistic terms (denoted with the usual P) and zero/one decisions (denoted with δ). If a hypothesis gets a zero from one of the knowledge sources, it is not considered in the search. We will discuss the knowledge sources (ellipses) in this section, while the computational components (rectangular boxes) are discussed in Section II-D. The shaded boxes and ellipses correspond to components that have identical counterparts in typical HMM recognizers and will not be discussed. The other system components and their HMM counterparts (if any) are introduced here and, when relevant, they will be examined closely later in the paper. The figure also contains a table with a short comparison of HMM and template-based recognizers. Note that a detailed comparison of the DTW and Viterbi algorithms is given in the Appendix . For reference, Fig. 2 gives a short step-by-step summary of how to "train" all the necessary knowledge sources of the template-based recognizer.

```
1. Segment and transcribe the training
   database. The result is a segmentation
   file with phoneme transcriptions and
   phonetic boundary timings.
2. Merge consecutive segments at will to
   produce supra-phonemic templates.
3. Determine meta-information for each segment.
4. Determine suitable transition costs
   to each pair of possible meta-tags.
5. Assign suitable acoustic scaling matrices
   for each frame in the database.
6. Calculate an indexing structure for fast
   k-nearest neighbours selection.
7. Compute weights to combine the different
   knowledge sources.
```

Fig. 2. Cookbook description of how to train the knowledge sources of the template-based recognizer.

1) Template Database: All acoustic knowledge of the recognizer is contained in the template database. The raw data is segmented and each segment is labeled with its unit identity. With *unit* we mean the acoustic-phonetic unit used during recognition. Optionally, successive basic units can be concatenated to form syllable, word, or even word group examples. This merging is performed only at the level of the database segmentation, and can be made completely transparent to the rest of the system by labeling each extended unit by a combination of basic labels.

Finally, meta-information is added to each template. Currently, we opt to store the entire database in the original order, with the advantage that the *complete* acoustic-phonetic context of each template remains available.

2) Local Scaling Estimates: The calculation of the distance between an input frame and a reference frame uses a set of weights dependent on the reference frame, similar to the covariance matrices used in HMMs.

3) Template Concatenation Models: These models assign a prior probability to each pairwise concatenation of templates. The prior probability of a template *string* is a product of all the pairwise probabilities. The prior probability estimates can be based on a multitude of template features such as phonetic context, acoustic trajectory, and meta-information.

D. Computational Components

1) Local Distance Measures: The local distance measures in DTW alignment measure the acoustic similarity between a frame in the observation and a frame in a template. Hence, they play the same role as the observation probability density functions (pdf's) in HMM systems.

2) Bottom-Up Template Selection: The bottom-up template selection is a fast algorithm that matches templates to segments of the input, in order to prune the set of templates used in further processing. For each input frame, the algorithm selects a small number of templates that can start at that point in time, based only on acoustic resemblance.

3) Template Concatenation Costs: This module returns the (negative logarithm of the) prior probability of a template string. It combines scores based on concatenation models and lexicon information.

4) DTW Token Passing Search Engine: The search engine combines scores from all the knowledge sources and performs a time-synchronous token passing DTW alignment [7]. Each token (hypothesis) consists of a score and a node in the search space matching the input sequence up to the current input frame. The score is a weighted sum of the acoustic distance (DTW score) between the partial input sequence and the hypothesized template string, the negative logarithm of the *a priori* likelihood of that template string and the negative logarithm of the hypothesized word string likelihood. The search space is a dynamically constructed combination of the knowledge sources [8]. Each node in the search space is determined by a language model context, a current position in the lexicon network, the current template and the frame in the current template. Tokens for which these four references are identical are recombined.

When a token reaches the end of a template, all the bottom-up selections for the current input frame number are considered for expansion: the token is propagated to the first and second frame of each selected template if the concatenation of the current template, and the selected template is allowed by the lexicon and the language model. Apart from the bottom-up selections, the *natural successor template* (i.e., the template that follows the current template in the original recording) is also considered. This adaptation was made to favor longer strings of natural templates (see Section V). Given the vast search space, beam pruning has to be used. The actual DTW alignment uses Itakura constraints [9], and uses fixed additive skipping and stalling costs (see also the Appendix for a formal description).

III. Feature Transformations

A. Local Distance Measures

1) Introduction: DTW traditionally relies on a distance metric that is global and symmetric between reference and test frames, whereas HMMs rely on a state specific –i.e., local– pdf. The use of class-dependent pdf's is one of the aspects in which the HMM framework improves upon the classical DTW approach. In this section, we show how most of the beneficial properties of the HMM "distance calculation" can be transferred to the DTW approach.

2) Local Mahalanobis Distance Measure: Suppose we have a set $\mathcal{D}$ of M-dimensional vectors. A global distance metric uses a set of weights $\Lambda \in \mathbb{R}^{M\times M}$ to compute the distance between two (column-)vectors $\mathbf{x}$ and $\mathbf{y}$ as

$$d(\mathbf{x};\mathbf{y}) = (\mathbf{x}-\mathbf{y})^T \Lambda (\mathbf{x}-\mathbf{y}). \tag{6}$$

For $\Lambda = I^{M\times M}$, this is the Euclidean distance. With $\Lambda = \Sigma^{-1}$, the inverse of the covariance matrix of the data, the Mahalanobis distance is obtained. As shown by Bocchieri *et al.* [10], such global distance metrics are often too limited to cope with the complexity of the space of speech frames.

This global distance measure can be improved upon by making Λ dependent on the feature vectors that are being compared. If $\mathcal{D}$ is partitioned into a set of mutually disjoint classes, the *class-dependent* distance measure is defined as

$$d(\mathbf{x};\mathbf{y}) = (\mathbf{x}-\mathbf{y})^T \Lambda_{k(\mathbf{y})} (\mathbf{x}-\mathbf{y}) \tag{7}$$

with $k(\mathbf{y})$ being the class to which $\mathbf{y}$ belongs. For simplicity of notation, the $(\mathbf{y})$ in $k(\mathbf{y})$ is dropped in the remainder of this paper; k always denotes the class to which the second argument of the distance measure belongs. Such local distance measures can adopt themselves better to the data. On the other hand, they are not metrics, since both the symmetry and the triangular inequality properties are lost.[2]

An obvious candidate for a local distance measure is an adaptation of the Mahalanobis distance

$$d(\mathbf{x};\mathbf{y}) = (\mathbf{x}-\mathbf{y})^T \Sigma_k^{-1} (\mathbf{x}-\mathbf{y}) + \log|\Sigma_k| \tag{8}$$

where the extra additive bias term compensates for the transformations towards the different classes (cf. Jacobian needed when transforming a stochastic variable).

[2]Under the assumption of a correct alignment, $\mathbf{x}$ and $\mathbf{y}$ belong to the same class, and (7) is a metric.

As in [11], a version with diagonal covariance matrices will be used

$$d(\mathbf{x};\mathbf{y}) = \sum_{l=1}^{M} \left(\frac{x_l - y_l}{\hat{\sigma}_{k,l}} \right)^2 + \log \left(\prod_{l=1}^{M} \hat{\sigma}_{k,l}^2 \right). \qquad (9)$$

In the Appendix , we show that using such a distance measure in a DTW framework leads to a natural HMM interpretation of the recognition system.

Since the local distance measure corresponds to the acoustic likelihood [see (4) and (5)], we could also make the link with nonparametric density estimation techniques such as kernel/Parzen estimates [12]. Assuming a Gaussian kernel $\mathcal{N}(x;0,\Sigma_k)$ with Σ_k the covariance matrix of (8), the following density estimator is obtained:[3]

$$\hat{f}_{Parzen}(\mathbf{x}) = \frac{1}{|k|h^M} \sum_{y_i \in k} \mathcal{N}\left(\frac{\mathbf{x} - \mathbf{y}_i}{h}; 0, \Sigma_k \right) \qquad (10)$$

where h is the *bandwidth*. It is clear that the local Mahalanobis distance is equivalent to the Parzen density estimate, but using only the kernel of $\mathbf{y}$ instead of an average over all $\mathbf{y}_i$.

3) Tail Distribution Modeling: Apart from providing an additional motivation for the local Mahalanobis distance, other parallels with Parzen density estimation provide further insight. It is known that classical Parzen estimation performs quite poorly in the tails of the distribution [12]. A good practical solution is the use of *adaptive kernel estimates*, where the bandwidth h is a function of $\mathbf{y}$. The idea is to use wider kernels in low-density regions to avoid under-smoothing. A "pilot distribution," i.e., a rough first guess of the distribution, is used to calculate the local bandwidth [12].

When this idea is applied to the local Mahalanobis distance, we obtain

$$d(\mathbf{x};\mathbf{y}) = \sum_{l=1}^{M} \left(\frac{x_l - y_l}{\alpha_y \hat{\sigma}_{k,l}} \right)^2 + \log \left(\prod_{l=1}^{M} (\alpha_y \hat{\sigma}_{k,l})^2 \right) \qquad (11)$$

where α_y, the local bandwidth, has become dependent on $\mathbf{y}$. We used Gaussian mixture models (GMMs) with diagonal covariance matrices as a pilot distribution to calculate the α_y values. We used the same approach as in [12], but limited the range of α_y to 1 ± 0.33. As shown in Section VI, the variable bandwidth local Mahalanobis distance outperforms the fixed-bandwidth version.

4) Discriminative Local Distance Measures: An alternative to the local Mahalanobis distance is to consider discriminatively trained local distance measures. While discriminative training has proven advantageous in HMM systems [14] and initial results for template-based recognition were promising [15], [16], in the presented template-based recognizer, discriminatively trained distance measures do not improve recognition accuracy.

[3]If the underlying distribution is also assumed Gaussian with covariance matrix Σ_k, it can be shown that the choice of Σ_k for the kernel's covariance matrix is optimal [13].

B. Adaptation Methods

A major advantage of our approach is the availability of all information about the training data. If the database is large and diverse enough, we can match the input speech with examples of the closest speaker/environment, and award costs for switching speaker/environment in the hypothesis. This idea, dubbed *implicit adaptation*, is discussed in Section V-C.III. However, if the input speaker/environment deviates strongly from all available training data, explicit adaptation or normalization methods are needed.

Normalization techniques such as VTLN plus speaker adaptive training can easily be incorporated into our system by warping all the training data as well as the input speech. An extra advantage is that we can store the warping factor used on each training template as extra information, and give preference to the templates that have been warped with about the same factor as the input.

For speaker or environment adaptation, applying existing techniques is feasible but not always straightforward. For the actual "model transformation," we gain extra options because all information is at hand. For example, the data (and variance estimates) can be transformed *per speaker*, while this is impractical in HMM systems. Adapting to noise is also easier because we have to calculate the effect of the estimated noise on actual acoustic data rather than on models. On the other hand, using different transformations per state as in some HMM adaptation techniques leads to discontinuities in the reference templates, while matching with real—and thus continuous—templates instead of flawed models is a large part of the motivation to use example-based recognition. Also, estimating the most likely transformation is easier on an explicit acoustic state model than on a set of templates.

IV. Bottom-Up Template Selection

A. Motivation

One of the major differences between HMMs and DTW lays in the number of acoustic units (models or templates). This has strong implications for the decoder. Let us compare both systems for a time-synchronous token passing decoder [7], using phonemes as acoustic units.

As long as we only consider transitions within a single HMM or template, both decoders are almost identical, as shown in the Appendix . However, at phoneme boundaries, tokens in an HMM system are only propagated to the first state of all possible subsequent phonemes according to the lexicon. Hence, the basic branching factor is determined by the lexicon, irrespective of the number of HMM states.[4] In the template-based system, each template represents a distinct hypothesis. As a result, the branching factor at phoneme boundaries is *multiplied by the number of templates for each phoneme*. Even for small template databases, the decoder will not be able to handle the resulting search space [17].

[4]Note that while the introduction of context dependent phonemes increases the number of states with an order of magnitude, only those states with matching phonemic context can be activated. As such, the introduction of context dependent phonemes has little effect on the branching factor.

Our solution is to use a bottom-up selection procedure, designed to suggest acoustically interesting templates to the top-down token passing decoder.[5] For each input frame, a list of templates with high enough probability to match the following segment of input is returned by an efficient *acoustic look-ahead* module. The first part of the look-ahead module, a *fast K-nearest neighbor selection*, finds a small number of vectors in the neighborhood of the input vector. The second part, which we dubbed *time filter*, finds template candidates based on the time evolution of the nearest neighbors.

B. Fast K-Nearest Neighbor Selection

Formally, K-nearest neighbors (K-NN) selection is the problem of finding the K closest points in a set of points $\mathcal{D}$, given a query point $\mathbf{q}$ according to some distance measure. Our setup uses a criterion on the NN that is less strict: we are satisfied if a certain fraction (e.g., 90%) of the real NN is selected by the NN search algorithm. While we could be missing a small number of points that belong to the true K-NN, we have not seen any significant impact of this as the next step—the time filter—can fill in occasional missing elements in the K-NN.

To check the difference in acoustic generalizing ability between HMMs and the example-based approach, we performed a set of experiments classifying each frame of the testset independently. The HMM classifier used the sum of the likelihoods of all states of a certain phoneme, multiplied with the prior probability of the state as estimated on the training set. The K-NN classifier uses majority voting. The HMM classifier achieves a performance of 70%, the 1-NN classifier 54%, the 3-NN classifier 65% and the 10-NN classifier 67%. Furthermore, the reference segmentation was made with an HMM system. For words with multiple possible transcriptions, this favors the HMM classifier. It can be concluded that the acoustic generalizing ability at the frame level is comparable for both systems.

Most existing fast K-NN algorithms are based on an indexing structure that is trained offline. The approaches can be divided in *hierarchical* indexing stuctures and *graph-based* indexing structures. Well-known tree-based K-NN algorithms are kd-trees [18] or bbd-trees [19]. Examples of graph-based structures for K-NN search are the randomized neighborhood graph [20] and the roadmap-algorithm, which has been developed for fast selection of Gaussians [21] and has been adapted for a K-NN search in speech recognition feature space [11].

Most K-nearest neighbor algorithms have a time complexity that is exponential in the number of dimensions. This is an instance of the so called "curse of dimensionality" [22]. With increasing dimensionality, such algorithms will perform as bad as, or even worse, than a brute force K-NN search. A brute force K-NN search has a time complexity of $O(M|\mathcal{D}|\log_2 K)$ whereas kd-trees are logarithmic in $|\mathcal{D}|$ but exponential in the feature dimensionality M. This exponential factor dominates very quickly.

[5]The distinction between *top-down* and *bottom-up* or *data driven* can easily be understood from a schematic notation of the search space, where the larger concepts (e.g., sentences) are put above their composing parts (e.g., words). A top-down search strategy will build hypotheses from top to bottom, always expanding into valid lower level representations, while a bottom-up strategy will select probable elementary parts and try to compose them into a valid higher level concept.

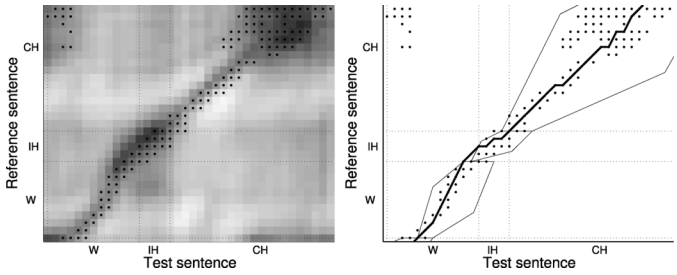

Fig. 3. Time filter example. On the left a local distance matrix with for each column the K-NN (black dots). On the right the time filter's activations (hexagons) and the best DTW path (dash-dot line).

Another disadvantage of most K-NN algorithms, in particular the tree-based ones, is their inability to cope with nonsymmetric distance measures such as those from (7). Graph-based algorithms are somewhat more robust as long as the violation is not too large. Therefore, in the architecture described in this paper, the roadmap algorithm (described in detail in [11]) is used to compute the nearest neighbors.

C. Time Filter

The time filter is based on the idea that it should be possible to quickly detect templates that resemble the input, based on a sequence of K-NN. When the input and the training sequence are highly similar, we can expect successive input frames to have nearest neighbors in successive training frames. The time filter uses that principle on the complete training database and adds some robustness to account for different nonlinear alignments. Hence, it searches for *approximately diagonal patterns* in the sparse local distance matrix made up of the K-NN. The same idea is used in the popular BLAST algorithms for similarity search in DNA or protein databases [23].

We presented the time filter algorithm in detail in [17]. Fig. 3 shows a real-life example. The left part of the figure shows the local distance matrix for a segment of the same sentence spoken by two different speakers. The phoneme boundaries (obtained through HMM Viterbi alignment) are given by dotted lines, and for each column, the K-NN are labeled with dots in the matrix. The right figure shows only the K-NN, the activations calculated by the time filter (hexagons in full line) and, for reference, part of the optimal DTW alignment path (dash-dot line) for the complete files.

A database template only gets activated when a sufficiently good path can be found in the K-NN matrix from its first to its last frame. In the figure, this corresponds to a path of K-NN between two horizontal dotted lines with a slope of approximately 45° and sufficiently small gaps.

It should be noted that we do not use the ending windows of the activations in our recognizer. In other words, DTW alignments are legal if the path "enters" a template through the activation's start window (base of the hexagon), even if the path does not "exit" the template through the ending window (top of the hexagon).

The time filter results in a very drastic reduction of the search space at a low computational cost. This is shown clearly by some statistics taken from the experiments presented in Section VI on the Resource Management benchmark: The number of nearest

neighbors presented as input to the time filter was less than 1% of the total number of vectors in the reference database. Based on these NN, the time filter on average selected less than 0.1% of all (phoneme) templates in the database, using about 1% of the total processing time. Still, given these activations, the correct hypothesis was always part of the search space.

D. Scaling Behavior

In [17], we showed that the time filter can easily be implemented with complexity $O(K)$, i.e., linear in K. In [11], we showed that in practice, the Roadmap algorithm on speech data will scale with complexity $O(K\log(K))$.

However, how does the number of nearest neighbors K scale with the database size $\Gamma\mathcal{D}\Gamma$? We believe that it will scale sublinearly since the acoustic space will be more densely populated with a growing database size, and hence the bottom-up selection will become more accurate with a relatively smaller K. We have no experimental results on this issue, but it seems safe to say that K will scale linearly with $\Gamma\mathcal{D}\Gamma$ at worst.

Given the fact that for our current experimental setup, the computation time for the top-down search dominates that of the bottom-up component, and that, unlike the top-down search, the bottom-up component is very suitable for massive parallelization, we are confident that even when K scales linearly to $\Gamma\mathcal{D}\Gamma$, the bottom-up component will remain practically feasible for very large training databases.

V. Template Concatenation

A. Choice of Basic Unit

The traditional choice of the phoneme as the basic modeling unit in speech recognition is based on pragmatics and intuition far more than on its status of elementary acoustic unit. Longer units such as syllables would be superior for modeling speech dynamics, but require far more training data. A decade ago this was enough reason to stick to (context-dependent) phonemes. Today, interest in speech recognition with longer units is rising [24]. In template-based recognition, longer templates severely limit the search space and hence seem an ideal way to achieve large speedups.

Most phoneme boundaries are places of high acoustic discontinuity. In speech synthesis, it was established that it is desirable to segment the training data at points of maximal acoustic stability to achieve smooth concatenation. Hence, we should raise the question of where to put the segment (template) boundaries: in-between all phonemes (the traditional approach), or only at acoustically stable parts, for example at prosodic boundaries or in the middle of stationary sounds? In the latter case, templates are centered around a transient part. Since most information may well be contained in the transient part [25], this option might be preferable.

However, using boundaries in the middle of stationary sounds causes templates of different length (in terms of number of phonemes). For our system, this is no problem, since the choice of the basic unit merely defines *possible* places where concatenations of different templates can occur. Therefore, we can easily combine templates of different length in a single recognizer. It is sufficient to have the transcription of all longer templates written as the concatenation of elementary transcriptions (e.g., when the basic unit is the phoneme, syllable templates can be combined with phoneme templates by labeling them with their phonetic transcription). Using two symbols per phoneme as elementary transcription unit allows segment boundaries in the middle of sounds. However, since longer templates require more data, and for pragmatic reasons, the experiments in Section VI all use phoneme templates.

B. Template Concatenation Model: Motivation

The concatenation model $P(\mathbf{T}|\mathbf{W})$ in (4) describes the appropriateness of combining templates in a hypothesis. Intuitively, this template concatenation likelihood will reflect the smoothness of the matched template sequence. Obviously, the smoothness will be maximal for templates that were successors in the original recording. To reflect that maximal smoothness, the template concatenation probability score is normalized to 1 (or 0 in the log domain, see Section II-D.IV) for these natural successors. Hence, even when short basic units such as phonemes are used as templates, optimal hypotheses will very often consist of multiple phonemes in their original context since "natural" concatenations have the highest prior probability.

C. Template Concatenation Costs: Implementation

Using the explicit notation $T_1^{N_T}$ to indicate that $\mathbf{T}$ is a sequence of N_T templates T_i, we can rewrite the template string probability of (4) as

$$P(\mathbf{T}|\mathbf{W}) = P(T_1^{N_T}|\mathbf{W}) \approx \left\{\prod_{i=2}^{N_T} P(T_i|T_{i-1},\mathbf{W})\right\} P(T_1|\mathbf{W}) \quad (12)$$

where the last line approximates the probability of a template given its predecessor string by the probability of the template given only its immediate predecessor. This first-order approximation may well be criticized, but is necessary for the efficiency of the Dynamic Programming decoder. When templates are sufficiently long, the influence of the direct predecessor template will dominate the influence of the more distant predecessors.

The factor $P(T_i|T_{i-1},\mathbf{W})$ in the last line of (12) expresses the probability that a certain template T_i follows a given template T_{i-1}, when making up part of the word string $\mathbf{W}$. While it might be conceivable that in some words abruptness in concatenations is more important than in others, it is far more practical to have a 0/1 dependency, reducing the conditioning on $\mathbf{W}$ to a mere lexical lookup function. Hence, we drop $\mathbf{W}$ in our further discussion, assuming that probabilities are only assigned when the concatenation is lexically allowed.

Determining the transition probabilities $P(T_i|T_{i-1})$ directly from the training database (cf. N-gram distributions) is not an option since each template occurs exactly once in the training database by definition. A more appealing alternative is to base the template transition probabilities on a coarse set of presumed independent features. In general, the probability estimates can be based on the following basic knowledge sources.

1) The ***phonetic context*** of the template offers crucial information about coarticulation effects on the template.

2) A ***smoothness measure*** may be calculated from the combined acoustic trajectory of the templates.
3) A match or mismatch in ***meta-information*** (e.g., concatenating templates of the same or opposite gender) will clearly contribute to the naturalness or unnaturalness of template sequences.

In the following discussion, we will use "concatenation *cost*" instead of "probability" since in practice we are working in the log domain and we do not normalize all estimates.

1) Context Dependency for Template-Based Recognition: For HMM systems, context dependency is part of the acoustic model construction, while in example-based recognition it becomes a separate knowledge base. Because of this separation, we gain flexibility (soft decisions as opposed to predefined state tying), and open possibilities to use long-term context effects.

As an example, suppose the transcription of a hypothesis T_1^6 is given by /E X A M P L/. Each of the T_i is a fragment of natural speech, and hence has an *original context*, which can be found in the template database. The third template "A" could originally have been recorded as part of /…Z A N…/. The original left context Z and right context N are the cause of most actual coarticulation effects in template A. However, in the hypothesis, X and M determine the desired coarticulation effects. It is then natural to partly base the concatenation cost $-\log(P(T_3|T_2))$ on the difference between the original left context Z of template T_3 and the identity of its left neighbor in the hypothesis, X. Similarly, $-\log(P(T_4|T_3))$ can partly be based on the difference between N and M.

Many possibilities (e.g., phonetically relevant decision tree clustering as in HMM allophone clustering [26]) are available to structure and estimate the concatenation costs. However, for the results reported in Section VI, only a single shared context mismatch cost was used.

2) Smoothness Cost: The success of the use of acoustic smoothness estimation in speech synthesis should not immediately convince us of the viability of the approach in template-based speech recognition. In speech synthesis, the acoustic concatenation cost is used to choose between *all* examples that sufficiently match the "target" [27]. How well an example matches the target can depend on many factors, such as phonetic context and prosody. These factors are not acoustic, however. In template-based recognition, the first and foremost selection is based on acoustic match–match between template and input signal. When two adjoining templates closely match the input signal, they are already likely to have an acoustically smooth combined trajectory. Therefore, the effect of acoustic smoothness for concatenation probability estimation may well be much smaller for recognition than for synthesis.

Preliminary experiments with a simple acoustic smoothness measure did not show any improvement in recognition performance. Therefore, the results presented in this paper did not use an acoustic smoothness cost.

3) Feature Discontinuity Cost: Implicit Adaptation: Most meta-features (see Section II-C.I) are not relevant to speech synthesis: typically, the complete example database is produced by one speaker in one location. For speaker-independent speech recognition, however, a whole range of features may be considered. Using costs for concatenations of templates with different meta-information leads to an increased consistency in the choice of the concatenated templates, which provides *implicit adaptation* [17] of the recognizer w.r.t. the speaker, recording condition, etc.

As an example, take implicit gender adaptation. We add a cost each time a hypothesis concatenates two templates of opposite gender. This way, hypotheses with consistent gender tags are preferred. The beam search algorithm still investigates all plausible alternatives, so both predominantly male and predominantly female hypotheses will be considered in parallel and the best one will be used.

VI. Experiments

A. Experimental Setup

In this section, we present a series of experiments on the Resource Management (RM) benchmark. All four testsets (feb89, oct89, feb91, and sep92) are used. For HMM baseline results on this task, see Section VI-D. Our implementation of the DTW decoder runs at about 10 times real time on a dual processor workstation, using 256 MB of internal memory. We use the complete training database, labeled with gender and speaker ID information. Phonetic segmentations are made starting from sentence-level transcriptions, using ESAT's in-house HMM system.[6] This HMM system uses cross-word context-dependent modeling with 693 tied states.[7] Apart from a phoneme-level segmentation, a state-level segmentation is also obtained. This state-level segmentation is used for the definition of class-dependent local distance measures, i.e., the state-level segmentation together with the gender information for each template specifies $k(\mathbf{y})$ from Section III-A. Hence, the total number of classes used for local distance calculation is 1386.

The preprocessing transforms the 16–kHz audio into a 25-dimensional feature vector each 10 ms of speech using overlapping frames of 30 ms. The features are based on 24 mel-scaled filterbank coefficients, and their first and second time derivatives. MIDA –an improved LDA algorithm– is used to transform the space and keep the 25 most informative directions [28].

The database consists of only single-phoneme templates. This setup allows for easy comparison with similar HMM systems and a detailed analysis of the lengths of the chosen "natural template concatenations."

The language model used is the word-pair grammar defined for the RM benchmark. We use the CMU v0.4 phonetic lexicon with a set of 47 phonemes (including stressed and unstressed variants, and with the addition of "syllabic l, m and n"). The lexicon proved to be too crude on several occasions: especially a lack of vowel-reduced variants was noticed.

B. Comparison of Different Local Distance Measures

The first part of Table I shows the results (word error rates) for different local distance measures. The setup includes template concatenation costs based on phonetic context and gender tags (see Section VI-C). Rows one through three list results for,

[6]The DTW system can be used to segment the entire training database using the HMM segmentation (or a human segmentation) as a bootstrap. This produced no significant improvement.

[7]Using a simpler (context independent) HMM system for segmenting, the training database produces no significantly different results.

TABLE I
WORD ERROR RATES FOR DIFFERENT DTW SETUPS ON THE DIFFERENT TEST SETS. FIRST PART: DIFFERENT LOCAL DISTANCE MEASURES. SECOND PART: EFFECT OF TEMPLATE CONCATENATION COSTS

setup	feb89	oct89	feb91	sep92	avg
DTW0: EUCL	4.18	4.32	3.70	7.07	**4.82**
DTW1: LMH	3.63	4.28	3.58	5.00	**4.13**
DTW2: VBLMH	3.08	4.10	3.02	5.00	**3.80**
DTW3: VBLMH NG	3.32	4.28	2.82	6.02	**4.12**
DTW4: VBLMH NG NC	5.08	6.18	5.43	8.83	**6.39**

respectively, a Euclidean local distance measure (EUCL), the local Mahalanobis distance measure (LMH) from (9), and the local Mahalanobis distance with variable bandwidth (VBLMH) from (11), where the bandwidths were calculated from a GMM. The variable-bandwidth approach improves recognition accuracy with 8% relative. The total improvement caused by using local distance measures instead of a Euclidean is 21%.

C. Effect of Concatenation Costs

Row four in Table I (VBLMH NG) shows the result for a setup identical to row three, but without using gender costs. It can be seen that implicit adaptation based on gender information improves recognition accuracy with 8%. It should be noted that even when no gender transition cost is used, about 90% of all template transitions have matching gender labels. The gender transition cost makes practically all recognition results completely gender consistent.

Experiments with additional costs based on the speaker id showed no further improvement. Using a VTL estimate per speaker or per file, and using a distance between these estimates as a cost has the same effect as simply applying a gender switch cost.

Rows one to four in Table I use the simplest concatenation cost model by assigning a single fixed cost for all switches in acoustic context. The context is one phoneme wide, and forward and backward costs are applied independently. Row five (VBLMH NG NC) shows the result when neither context-dependent, nor gender-dependent concatenation costs are used. Comparing rows four and five shows that even trivial context dependency results in a relative recognition improvement of 35%. So far, more elaborate cost structures based on the phonetic context have not shown significant further improvements.

Each type of template concatenation cost is expected to have a different effect on the recognizer's behavior. Since natural concatenations never get a cost, higher concatenation costs will result in a preference for longer natural template sequences. A single natural concatenation is the same as using a biphone template, two successive natural concatenations equal a triphone template, etc. The difference with using longer templates at the level of the database segmentation, is that a nonnatural concatenation *can* take place after each phoneme template.

Fig. 4 shows the correlation between the average natural template sequence length, and recognition performance for the oct89 test set using the fixed kernel local Mahalanobis distance. Other setups and other test sets show similar behavior. The curve is obtained by varying the fixed cost for a nonnatural concatenation in a setup with no other template concatenation costs. It can be seen that the best recognition results are achieved with natural template sequence lengths of about 2. Fig. 4 shows a rather broad minimum, indicating that the system is only moderately sensitive to the choice of the fixed transition cost. The subfigure is a histogram of natural sequence lengths at the minimum of the curve. The few very long natural sequences are caused by the match between training and test material in the RM task.

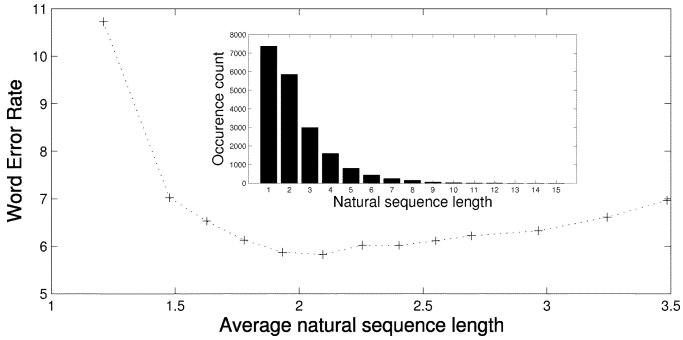

Fig. 4. Effect of average natural sequence length on word error rate and histogram of chosen natural sequence lengths.

TABLE II
WORD ERROR RATES FOR HMM SYSTEMS AND THE COMBINATION OF HMMS AND DTW. (DTW2 REFERS TO THE THIRD ROW IN TABLE I)

setup	feb89	oct89	feb91	sep92	avg
HMM1	2.46	3.32	2.58	5.78	**3.54**
HMM2	2.38	2.91	2.13	5.16	**3.15**
HMM1+HMM2	2.23	2.46	2.01	4.65	**2.84**
DTW2+HMM1	1.99	2.53	1.81	4.49	**2.71**
DTW2+HMM2	2.07	2.53	1.97	3.91	**2.62**
DTW2+HMM1+HMM2	1.99	2.20	1.85	3.75	**2.41**

The different concatenation costs in the experiments described in this section used fixed values for each mismatch. The exact value of the costs needs to be set relative to the size of the acoustic distances, which depends on the preprocessing and the local scaling. To get an idea of the size of the concatenation costs, we can compare them with the average between-frame acoustic distance along the winning path. For the "default" setup, i.e., row three in Table I, the gender mismatch cost was about 2.5 times the average local distance, and both left and right context mismatch costs were about 2 times that distance. Apart from those costs, all nonnatural concatenations got an extra cost of about twice the average local distance.

D. Combining Template-Based Recognition and HMMs

The first two rows of Table II show the results obtained on the same benchmark with two different HMM systems. The first system, "HMM1," uses the same features as used for the DTW experiments and the same number of states as used for clustering the local distance measures. The second line, "HMM2," uses a different frame shift, longer feature vectors (size $= 36$) and more states (824) than HMM1. Its result equals the best reported HMM result that we know of. Beyond that, further optimization is possible by tweaking the lexicon and by using phonological rules, which were however not used in our experiments [29].

Rows three to five show the results of a simple pairwise combination technique: both systems contribute their best sentence hypothesis and corresponding score. Each system then rescores the other's hypothesis through forced alignment. The resulting scores for the hypotheses are a weighted average of both systems' scores. The best of the two hypotheses is then chosen.

Obviously, this method of combination can only improve the WER if the systems are sufficiently different and make different errors. The DTW system shares about 30% to 35% of all errors with each of the HMM systems, while the two HMM systems share over 60% of their combined errors. The results in Table II show that the combination achieves a large performance gain. Combining DTW with the best HMM results in a 17% relative improvement, while the combination with the other HMM achieves a 23% relative improvement. Combining the two HMM systems also results in a system that is better than either of them alone, but is worse than the combination with DTW.

Finally, row six shows the same combination technique, but with both HMM systems and the best DTW system. It can be seen that adding DTW still improves over the combination of the two HMM systems with 15% relative.

E. Significance of the Results

The Bootstrap method of Bisani *et al.* [30] was used to perform nonparametric pairwise significance tests. These tests show that all results in Tables I and II are significantly different at a 95% confidence level, except for the difference between "DTW VBLMH" and "DTW LMH" (P-value 94%, hence almost significant), the difference between "HMM1" and "DTW VBLMH," and the difference between each of the pairwise combinations. Combining all three systems, however, improves significantly over all the pairwise combinations.

VII. Discussion

Although several times we have stressed the similarity between our template-based approach and HMMs, the fact that they produce different errors and the performance gain in the combination experiments show that template-based recognition is not merely "yet another not so perfect HMM." Instead, it indicates that template-based recognition is a valid and different model. This section repeats and highlights some important differences between example-based recognition and HMMs. We outline the potential benefits/shortcomings and discuss how they influence or could influence recognition results.

A. Differences in Favor of Template-Based Recognition

1) Acoustic Trajectory Modeling: Almost as soon as HMMs became established as the *de facto* standard in speech recognition, proposals were being made to improve the within-model trajectory modeling [31]. DTW deals with this issue thoroughly. Especially when long templates (or natural sequences) are used, the acoustic trajectory of complex coarticulation is modeled more accurately than can be done in HMM variants. The fact that optimal recognition accuracy is obtained with rather long natural sequences supports this claim. For a further discussion on this issue, see [32].

2) Flexible Context Dependency: HMMs put context dependency in the acoustic models. Therefore, enough acoustic data has to be available to estimate models for every context dependent variant. At the frame level, our system uses similar context dependent acoustic scalings, but additionally, context dependency is handled by the template concatenation costs. Our experimental results show that even trivial context-dependent concatenation costs significantly help performance.

3) Flexible Recognition Units: The flexibility of example-based recognition is also apparent in the ease with which recognition units can be defined. A single recognizer can use templates of different logical length. The flexibility is a result of the decoupling of frame-based acoustic scaling, context dependency, and the recognition unit. In HMM systems, those aspects are tightly coupled within the individual HMMs. The template-based recognizer can take advantage of this property by merging frequent consonant clusters, function words, or even frequent word groups into a single template.

4) Use of Meta-Information: Certainly an advantage of example-based recognition over any model-based technique is the possible use of meta-information. The experiment we used to show the advantage of implicit adaptation, i.e., gender switch costs, may not show this convincingly, since gender-dependent HMMs can easily be trained and used. However, training a separate HMM for each phoneme for many possible values of a composite tag of meta-information will almost always be impossible. Currently, we have no experimental results to show the true potential of the use of meta-information. However, other recent research shows that the use of meta-information is promising [33].

B. Differences in Favor of HMMs

1) Acoustic Generalization: The most common objection to all example-based techniques is a lack of generalization. Asymptotically in the number of examples, example-based classification and density estimation can generally be shown to be nearly optimal [13]. However, with finite sample sizes and especially for high-dimensional data, performance can decrease dramatically.

For the case of template-based speech recognition, it is very difficult to estimate the necessary amount of data, and to assess exactly how well the system generalizes. In Section IV-B, we showed that frame-based acoustic generalization for the small RM training database is comparable to HMMs. However, the bottom-up selection procedure could fail to generalize from the set of templates to an unusual input. For the test sets of the RM task, this was not the case, since the bottom-up search space included the correct transcription of all test sentences.

2) Influence of Acoustic Outliers: Example-based techniques are also prone to overrepresentation of outliers. In our case, outliers can stem from uncommon pronunciation or speaker characteristics, or because of wrong labeling or segmentation of the training data. Although adding more data should diminish the effect of outliers, a system that explicitly penalizes unusual examples could improve recognition accuracy.

3) Efficiency: Even for the small RM training database, our template-based recognizer is 10 to 100 times less efficient than typical HMM recognizers. HMMs combined with GMMs are

a very compact representation of large amounts of data. Therefore, although many optimizations (e.g., using longer templates) are still possible, template-based recognition is very unlikely to ever achieve the same efficiency as HMM recognizers.

4) Bottom-Up Versus Top-Down: Combining bottom-up acoustic selection with top-down language information intrinsically carries a certain amount of risk. When the respective knowledge sources "disagree," the search path can get stuck at a dead end. For our current system, this is the case when the bottom-up part selects templates with a different phoneme identity than the ones predicted by the lexicon. In a classical top-down HMM recognizer, the acoustic models for the phonemes predicted by the lexicon are always evaluated and used, even when their acoustic probability is very low. Of course, in that case the probability of a recognition mistake becomes high for top-down HMM systems as well.

The blame for mismatch between top-down and bottom-up information can obviously lay with either component. In Section VII-B.I, we concluded that it is too early to make a definitive judgment on the generalizing ability of the bottom-up component. We do know, however, that mistakes or omissions in the lexicon occur frequently. Especially the lexicon we used for our experiments is of rather poor quality. It is known that the use of a better lexicon and phonological rules has a strong positive influence on benchmarks on the RM task [29]. To have an idea about the influence of the lexicon quality for our recognizer, we did some exploratory tests by manually adding, e.g., a reduced version of some vowel in the lexicon. As this would often eliminate a whole string of errors in a given sentence, improving the lexicon could indeed yield further improvements. However, all the presented results were obtained with the standard lexicon without manual additions.

VIII. Future Directions: Towards Large-Vocabulary Recognition

Considering the experimental results and the list of improvements that are still feasible in the template-based approach, we believe that example-based recognition will be able to compete (performance-wise) with state-of-the-art HMMs, also when moving to tasks more complex than RM. Larger training databases should help acoustic generalizing ability, and a more diverse set of templates will favor more complex template concatenation models. Moreover, we do not see any great fundamental reasons why the positive experimental results on the combined HMM+DTW systems should not be obtained on larger systems, except for the fact that the low branching factor in the RM task may be favorable to the context concatenation model we implemented.

Computationally, moving towards large databases presents a few challenges. The first scaling problem is quite obvious. Does the acoustic data of a large training database fit in memory? While the RM training database contains about 3.8 h of acoustic data, large training databases may have up to 1000 times as much. Although even the largest databases will fit in the main memory of today's large servers, and the increase in available memory size is likely to outpace the growth of training material in our opinion, entirely storing those gigantic databases is not an efficient use of resources. Therefore, informative data selection techniques (i.e., database pruning techniques) are probably an essential step for true large vocabulary template-based recognition.

The scaling behavior of the bottom-up component was discussed in Section IV-D. We showed that the worst case scaling behavior of the bottom-up component is linear in database size. However, the bottom-up component is very easy to parallelize. With multi(-core) processor systems rapidly becoming mainstream, the bottom-up component should not present serious problems when scaling to large tasks.

The top-down search will suffer both from a larger lexicon and a higher number of templates for each phoneme. Scaling the lexicon does not cause a very large increase in the branching factor of the search space: a good lexicon network will ensure that most of the increase goes to shared arcs [8]. The increase in available templates is a more serious problem. At this point, the relationship between database size and the number of templates that should be selected by the bottom-up component is not clear. In any case, units larger than phonemes should be considered when using large databases. We suspect that the gain in efficiency by using longer units should largely offset the increase in the number of selected templates.

Using longer LM contexts slows down the search because tokens cannot be recombined as quickly. On the other hand, a more accurate LM probability estimate will allow better pruning, and in our experience very often leads to *faster* recognition. Therefore, we are confident that using N-grams will present no real problems.

Finally, modern large vocabulary tasks contain a few "practical" problems. However, they occur mostly at the level of the language model (e.g., repetitions) or the lexicon (e.g., out of vocabulary words), and not at the level of the acoustic modeling. Since our top-down decoder is structurally similar to HMM decoders, the same solutions will apply.

All in all, the move to true large vocabulary tasks is far from trivial. A number of new techniques (e.g., informative data selection) as well as experience with new modeling issues (e.g., use of longer units) need to be developed. However, we are confident that it is feasible, and that the ever-increasing hardware capabilities are a necessary and essential catalyst for research in large vocabulary template-based recognition.

IX. Conclusion

In this paper, we have introduced an extensive framework that makes template-based recognition viable for medium vocabulary continuous speech recognition. In the past, two major arguments were used to discard DTW for continuous speech recognition: bad computational behavior and poor speaker-independent performance. To keep the computational load within bounds, we introduced a bottom-up template selection procedure based on the calculation of distances to only a small percentage of all feature vectors in the database. To improve speaker-independent recognition performance, we extended the classical DTW algorithm with a number of ideas borrowed from state-of-the-art HMMs and concatenative speech synthesis.

As a first step, we introduced a refined model for template-based recognition that expands on the classical Bayesian recognition paradigm. While fitting in all typical knowledge sources as used in HMM recognizers, the new model also shows that template concatenation costs are an inherent feature of example-based recognition. The use of concatenation costs based on meta-information leads to a new form of adaptation. Also, context dependency, a major milestone in the evolution of HMM systems, can be expressed in terms of these template concatenation costs.

Our attention to the similarities with HMM systems also motivated extensive research concerning local distance measures, the equivalent in DTW for state-dependent covariance matrices in HMM systems. Class-dependent local distance measures proved to be essential for bridging the performance gap between DTW and HMMs. Further research will expand on the presented work, investigating suitable local scaling operations on the template level rather than the frame level. This is a departure from the HMM equivalent.

We performed comparative recognition experiments for our prototype system and a state-of-the art HMM system on the medium vocabulary Resource Management task. An HMM system using the same features and number of acoustic states as used in the example-based system did not produce statistically significant better results than the new system. A larger HMM system still outperformed the example-based recognizer, however.

So far, we have not focused on the possibilities of combining template-based recognition with HMMs, as suggested in [34] and [33]. However, a very simple sentence-level combination technique showed significant improvement over the best HMM results. Therefore, the practical relevance of our research should possibly be sought in combination methods.

To conclude, we believe that the presented system is an ideal platform for future research, as in its baseline implementation it already yields competitive results and the architecture opens lines of research that cannot be addressed with model-based systems.

APPENDIX
EQUIVALENCE AND DIFFERENCES BETWEEN DYNAMIC TIME WARPING AND VITERBI ALGORITHMS

The dynamic time warping and the Viterbi algorithms are two instances of dynamic programming in which a global score is found as a sum of local distances along the optimal alignment between input and reference. By definition, the optimal alignment is the one that yields the smallest distortion (or best match) between input and reference. The main conceptual difference is that in the case of DTW the reference is a data stream just like the input, while in the case of Viterbi, the reference is a set of HMM models. These differences primarily have an impact on the way the local distances are computed.

In this Appendix, we make a detailed analysis of similarities and differences between both approaches. We start from the HMM/Viterbi formulation and show how the HMM states can be interpreted as frames of a template and vice versa. For the sake of this analysis, we restrict ourselves to isolated word recognition and full word HMM models. We assume the HMM word model to be made up of N_S independent states $\{S(q), q = 1 : N_S\}$ with Bakis style state connections; i.e., in which the states are organized in a left-to-right fashion with the addition of single state skips.

Assuming an input stream of N feature vectors $\{\mathbf{x}_i\}$, the Viterbi algorithm finds an optimal alignment $\{q(i), i = 1 : N\}$ between input feature stream and model along the path that yields the highest probability

$$P_{vit} = \max_{q(.)} \prod_{i=1}^{N} \{P(\mathbf{x}_i | S(q(i))) \times P(S(q(i)) | S(q(i-1)))\}$$

where $q(.)$ is the set of all legal state alignments in which state transitions are constrained by $q(i) = q(i-1) + j$ with $j \in \{0, 1, 2\}$. As initial and final condition, we assume a dummy initial state $S(0)$, $P(S(1)|S(0)) = 1.0$, and $q(N) = N_S$.

In order to find the link to DTW, we need to work with continuous-density HMMs and constrain the HMM observation probabilities $P(\mathbf{x}_i|S(q))$ to single Gaussian densities $\mathcal{N}(\mathbf{x}; \mu_q, \Sigma_q)$. Furthermore, all transition probabilities are pooled to a single set P_T with only three different probabilities (for self-loops, successor transitions, and state skips). Under these constraints, we now compute the negative log-probability of the Viterbi path score, as shown by (13)–(15) at the bottom of the page. In (15), $d_M(.,.;\Sigma)$ stands for the Mahalanobis distance parameterized by the covariance matrix Σ.

In the constrained case, in which all states in the HMM are modeled by a state-specific mean and one pooled covariance, the Viterbi equation further reduces to

$$L_{vit} = \min_{q(.)} \sum_{i=1}^{N} \left\{ d_M\left(\mathbf{x}_i, \mu_{q(i)}; \Sigma\right) - 2 \log P_T(q(i) - q(i-1)) \right\}. \quad (16)$$

$$L_{vit} = \min_{q(.)} \sum_{i=1}^{N} \left\{ -\log \mathcal{N}\left(\mathbf{x}_i; \mu_{q(i)}, \Sigma_{q(i)}\right) - \log P\left(S(q(i)) | S(q(i-1))\right) \right\} \quad (13)$$

$$= \min_{q(.)} \sum_{i=1}^{N} \left\{ \left(\mathbf{x}_i - \mu_{q(i)}\right)^T \Sigma_{q(i)}^{-1} \left(\mathbf{x}_i - \mu_{q(i)}\right) + \log |\Sigma_{q(i)}| - 2 \log P_T\left(q(i) - q(i-1)\right) \right\} \quad (14)$$

$$= \min_{q(.)} \sum_{i=1}^{N} \left\{ d_M\left(\mathbf{x}_i, \mu_{q(i)}; \Sigma_{q(i)}\right) + \log |\Sigma_{q(i)}| - 2 \log P_T\left(q(i) - q(i-1)\right) \right\} \quad (15)$$

Finally, in the above equation, the Mahalanobis distance reduces to a Euclidean distance given an appropriate linear transformation of the input feature vector and all the means of the HMM states.

In the case of the DTW algorithm, the input sequence is matched against a template $\{t_j, j = 1 : N_t\}$, by finding the best alignment, i.e., the one that yields the smallest global distance, according to following most general equation:

$$D_{\text{dtw}} = \min_{j(.)} \sum_{i=1}^{N} \left\{ \alpha(j(i), j(i-1)) \times d\left(\mathbf{x}_i; t_{j(i)}\right) + \beta\left(j(i), j(i-1)\right) \right\} \tag{17}$$

with path constraining functions $\alpha(.,.)$, $\beta(.,.)$ and appropriate boundary conditions for $j(0)$ and $j(N)$.

A popular implementation of DTW uses a Euclidean local distance measure and Itakura path constraints where $\beta(.,.) = 0$, $\alpha(j, j+1) = 1.0, 1.0 \leq \alpha(j, j), \alpha(j, j+2) \leq 2.0$, and $\alpha(.,.) = +\inf$ for other values. The Itakura constraint also states that a single reference frame can be used at most twice. Experiments using a fixed additive cost (i.e., $\beta(j, j) = \beta(j, j+2) = ct$ and $\beta(j, j+1) = 0$) gave similar results as the Itakura approach and speed up the DTW calculation. Hence, we used the additive version in our software.

Now we may express the equivalence between DTW and Viterbi in greater detail.

- The equivalence is valid for continuous-density HMMs using single normal distributions.
- The frames of the reference template correspond to the means of states in the HMM.
- The negative log-prob of probabilities in Viterbi corresponds to local distances in the DTW framework.
  - — Using a Euclidean distance corresponds to using HMM states with pooled unity covariance.
  - — In the case of state-dependent covariances, the local distances should be written as a Mahalanobis distance (parameterized by the state covariance) **and** a bias term equal to $\log|\Sigma_q|$.
- State connections and the associated transition probabilities have a similar role as DTW path constraints.
- Typical implementations of Viterbi and DTW have the following significant differences.
  - — The number of states in an HMM is typically considerably smaller (factor 2–3) than the number of frames in the input, while in DTW the input and reference are roughly of equal length; this makes unusual paths (that stay for a long time in the same state and then rush through the other states) easier in the case of HMMs than in the case of template matching.
  - — In an HMM, the transition probabilities are state dependent, while in a classic DTW implementation with Itakura constraints, off-diagonal transitions are penalized by a fraction of the local distance. However, a shared additive cost performs just as well, further closing the gap between DTW and HMMs.

It should be remarked that the HMM and DTW equivalence holds for the "within template" scoring only. The interconnections between individual templates are drastically more flexible in our case than in typical HMM systems, though it could be mimicked to some extent by graphical models [3].

REFERENCES

[1] J. S. Bridle, "Towards better understanding of the model implied by the use of dynamic features in HMMs," in *Proc. Int. Conf. Spoken Lang. Process.*, Jeju Island, Korea, Oct. 2004, vol. I, pp. 725–728.

[2] M. Ostendorf, V. V. Digalakis, and O. A. Kimball, "From HMM's to segment models: A unified view of stochastic modeling for speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 4, no. 5, pp. 360–378, Sep. 1996.

[3] J. A. Bilmes, G. Zweig, T. Richardson, K. Filali, K. Livescu, P. Xu, K. Jackson, Y. Brandman, E. Sandness, E. Holtz, J. Torres, and B. Byrne, "Discriminatively structured graphical models for speech recognition," Report of the JHU 2001 Summer Workshop, 2001.

[4] W. D. Marslen-Wilson and A. Welsh, "Processing interactions and lexical structure of spoken language understanding," *Cognitive Psychol.*, vol. 10, pp. 29–63, 1978.

[5] L. R. Rabiner, J. G. Wilpon, A. M. Quinn, and S. G. Terrace, "On the application of embedded digit training to speaker independent connected digit recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 32, no. 2, pp. 272–280, Apr. 1984.

[6] H. Ney, "Modeling and search in continuous speech recognition," in *Proc. Eur. Conf. Speech Commun. Technol.*, Berlin, Germany, Sep. 1993, vol. I, pp. 491–498.

[7] S. J. Young, N. H. Russell, and J. H. S. Thornton, "Token passing: A simple conceptual model for connected speech recognition systems," Cambridge Univ. Eng. Dept., 1989, Tech. Rep. CUED/F-INFENG/TR38.

[8] K. Demuynck, J. Duchateau, D. Van Compernolle, and P. Wambacq, "An efficient search space representation for large vocabulary continuous speech recognition," *Speech Commun.*, vol. 30, no. 1, pp. 37–53, Jan. 2000.

[9] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 23, no. 1, pp. 67–72, Feb. 1975.

[10] E. L. Bocchieri and G. R. Doddington, "Frame-specific statistical features for speaker independent speech recognition," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 34, no. 4, pp. 755–764, Aug. 1986.

[11] M. De Wachter, K. Demuynck, P. Wambacq, and D. Van Compernolle, "A locally weighted distance measure for example based speech recognition," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Montreal, QC, Canada, May 2004, vol. I, pp. 181–184.

[12] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. London, U.K.: Chapman and Hall, 1986.

[13] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed. San Mateo, CA: Morgan Kaufmann, 1990.

[14] P. C. Woodland and D. Povey, "Large scale discriminative training of hidden Markov models for speech recognition," *Comput. Speech Lang.*, vol. 16, no. 1, pp. 25–47, 2002.

[15] M. Matton, M. De Wachter, D. Van Compernolle, and R. Cools, "A discriminative locally weighted distance measure for speaker independent template-based speech recognition," in *Proc. Int. Conf. Spoken Lang. Process.*, Jeju Island, Korea, Oct. 2004, pp. 429–432.

[16] ——, "Maximum mutual information training of distance measures for template based speech recognition," in *Proc. Int. Conf. Speech Comput.*, Patras, Greece, Oct. 2005, pp. 511–514.

[17] M. De Wachter, K. Demuynck, D. Van Compernolle, and P. Wambacq, "Data driven example based continuous speech recognition," in *Proc. Eur. Conf. Speech Commun. Technol.*, Geneva, Switzerland, Sep. 2003, pp. 1133–1136.

[18] J. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sep. 1975.

[19] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu, "An optimal algorithm for approximate nearest neighbor searching in fixed dimensions," *J. ACM*, vol. 45, no. 6, pp. 891–923, Nov. 1998.

[20] S. Arya and D. M. Mount, "Approximate nearest neighbor queries in fixed dimensions," in *Proc. 4th Annual ACM Symp. Discrete Algorithms*, Austin, TX, Jan. 1993, pp. 271–280.

[21] D. Povey and P. Woodland, "Frame discrimination training on HMMs for large vocabulary speech recognition," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Phoenix, AZ, Mar. 1999, pp. 333–336.

[22] R. Bellman, *Adaptive Control Processes*. Princeton, NJ: Princeton Univ. Press, 1961.

[23] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, "Gapped BLAST and PSI-BLAST: A new generation of protein database search programs," *Nucleic Acids Res.*, vol. 25, no. 17, pp. 3389–3402, 1997.
[24] A. Ganapathiraju, J. Hamaker, M. Ordowski, G. Doddington, and J. Picone, "Syllable-based large vocabulary continuous speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 9, no. 4, pp. 358–366, May 2001.
[25] S. Furui, "On the role of spectral transition for speech perception," *J. Acoust. Soc. Amer.*, vol. 80, no. 4, pp. 1016–1025, Oct. 1986.
[26] K.-F. Lee, S. Hayamizu, H.-W. Hon, C. Huang, J. Swartz, and R. Weide, "Allophone clustering for continuous speech recognition," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Albuquerque, NM, 1990, vol. II, pp. 749–752.
[27] A. J. Hunt and A. W. Black, "Unit selection in a concatenative speech synthesis system using a large speech database," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Atlanta, GA, May 1996, vol. I, pp. 373–376.
[28] K. Demuynck, J. Duchateau, and D. Van Compernolle, "Optimal feature sub-space selection based on discriminant analysis," in *Proc. Eur. Conf. Speech Commun. Technol.*, Budapest, Hungary, Sep. 1999, vol. III, pp. 1311–1314.
[29] J.-L. Gauvain, L. Lamel, G. Adda, and M. Adda-Decker, "Speaker-independent continuous speech dictation," *Speech Commun.*, vol. 15, no. 1–2, pp. 21–37, Oct. 1994.
[30] M. Bisani and H. Ney, "Bootstrap estimates for confidence intervals in ASR performance evaluation," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Montreal, QC, Canada, May 2004, pp. 409–412.
[31] S. Roucos, M. Ostendorf, H. Gish, and A. Derr, "Stochastic segment modeling using the estimate-maximize algorithm," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, 1988, pp. 127–130.
[32] H. Strik, "How to handle pronunciation variation in ASR: By storing episodes in memory?," in *Proc. Speech Recognition and Intrinsic Variation Workshop*, Toulouse, France, May 2006, pp. 33–38.
[33] G. Aradilla, J. Vepa, and H. Bourlard, "Using pitch as prior knowledge in template-based speech recognition," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Toulouse, France, May 2006, vol. I, pp. 445–448.
[34] S. Axelrod and B. Maison, "Combination of hidden Markov models with dynamic time warping for speech recognition," in *Proc. Int. Conf. Acoust., Speech, Signal Process.*, Montreal, QC, Canada, May 2004, pp. 173–176.

Mathias De Wachter received the M.Sc. degree in informatics from the Katholieke Universiteit Leuven, Leuven, Belgium, in 2001. He is currently pursuing the Ph.D. degree in the Speech Processing Research Group, Electrical Engineering Department (ESAT), Katholieke Universiteit Leuven.

His main research interests are template-based speech recognition, including acoustic modeling, search algorithms and system architecture, and automatic speech recognition in general.

Mike Matton received the M.Sc. degree in informatics and the Master in Artificial Intelligence Degree from the Katholieke Universiteit Leuven, Leuven, Belgium, in 2002 and 2003, respectively. He is currently pursuing the Ph.D. degree in computer engineering at the NINES Research Group, Department of Computer Science, Katholieke Universiteit Leuven.

His research interests currently include template-based speech recognition, distance measures for speech recognition, and parallel computing.

Kris Demuynck received the M.S. and Ph.D. degrees in electrical engineering from the Katholieke Universiteit Leuven, Leuven, Belgium, in 1994 and 2001, respectively.

He is a Senior Researcher at the Speech Processing Research Group, Electrical Engineering Department (ESAT), Katholieke Universiteit Leuven. His principal research interest is large-vocabulary continuous speech recognition (LVCSR), covering a variety of topics such as search algorithms, acoustic modeling, feature extraction, feature-based resynthesis, latent semantic analysis, and novel speech recognition architectures. He is also the Lead Software Architect of the ESAT LVCSR-toolkit.

Patrick Wambacq (M'87) received the M.S. and Ph.D. degrees in electrical engineering from the Katholieke Universiteit Leuven, Leuven, Belgium, in 1980 and 1985, respectively.

From 1980 to 1998, his main interests were image processing in general, and automatic visual inspection more specifically. Since 1998, he has been the head of the Speech Processing Research Group, Electrical Engineering Department (ESAT), Katholieke Universiteit Leuven, with research in the areas of robust speech recognition, spontaneous speech recognition, new architectures for recognition, speaker adaptation, clinical and educational applications of speech recognition, and speech and audio modeling.

Ronald Cools received the M.Sc. and Ph.D. degrees in computer science from the Katholieke Universiteit Leuven, Leuven, Belgium, in 1984 and 1989, respectively.

Since 1996, Ronald Cools has been the head of the Numerical Integration, Nonlinear-Equations, and Software (NINES) Group, Department of Computer Science, Katholieke Universiteit Leuven. Initially, his main interests were the construction of cubature formulas for the approximation of low-dimensional integrals, adaptive software for multivariate numerical integration, and the computation of all solutions of systems of polynomial equations. He codeveloped widespread software for solving problems in these areas. His interests in the area of approximating multivariate integrals shifted towards higher dimension, currently focussing on quasi-Monte Carlo methods. He has a broad interest in numerical software and is a member of IFIP Working Group 2.5 on Numerical Software.

Dirk Van Compernolle (M'85) received the Electrical Engineering degree from the Katholieke Universiteit Leuven (K.U. Leuven), Leuven, Belgium, in 1979 and the M.Sc. and Ph.D. degrees from Stanford University in 1982 and 1985, respectively.

From 1985 to 1987, he was with the IBM T. J. Watson Research Center, Yorktown Heights, NY. Since 1987, he has held positions at the Electrical Engineering Departement (ESAT), K.U. Leuven, where he has been a Professor since 1994. From 1994 to 2000, he was with Lernout and Hauspie Speech Products as Vice President of Research. His research interests have been in the areas of cochlear implants, microphone arrays, and noise robust speech recognition. Currently, his research is in the area of novel speech recognition architectures inspired by human speech recognition. He is the general chair of INTERSPEECH 2007.