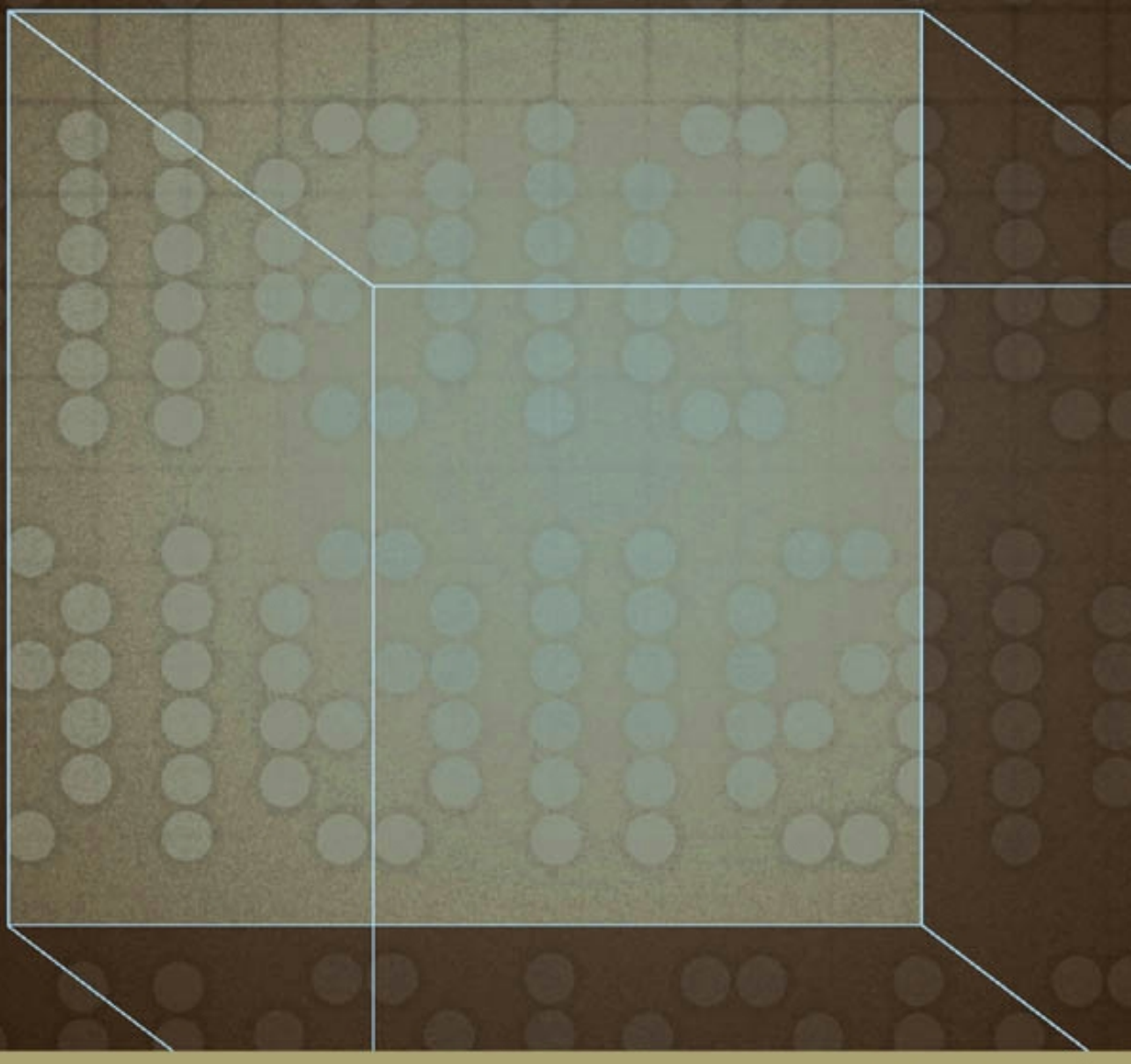
PEARSON NEW INTERNATIONAL EDITION

Speech and Language Processing

Daniel Jurafsky James H. Martin

Second Edition



ALWAYS LEARNING

PEARSON

Pearson New International Edition

Speech and Language Processing
Daniel Jurafsky James H. Martin
Second Edition

PEARSON

8.4 Diphone Waveform Synthesis

We are now ready to see how the internal representation can be turned into a waveform. We present two kinds of **concatentative** synthesis: **diphone synthesis** in this section, and **unit selection synthesis** in the next section.

Recall that for diphone synthesis, our internal representation is as shown in Fig. 8.13 and Fig. 8.14 and consists of a list of phones, each phone associated with a duration and a set of F0 targets.

Diphone

The diphone concatenative synthesis model generates a waveform from a sequence of phones by selecting and concatenating units from a prerecorded database of **diphones**. A diphone is a phone-like unit going from roughly the middle of one phone to the middle of the following phone. Diphone concatenative synthesis can be characterized by the following steps:

Training:

1. Record a single speaker saying an example of each diphone.
2. Cut each diphone from the speech and store all diphones in a database.

Synthesis:

1. Take from the database a sequence of diphones that corresponds to the desired phone sequence.
2. Concatenate the diphones, with slight signal processing at the boundaries.
3. Use signal processing to change the prosody (f0, duration) of the diphone sequence to the desired prosody.

Coarticulation

We tend to use diphones rather than phones for concatenative synthesis because of the phenomenon of **coarticulation**. In Chapter 7 we defined **coarticulation** as the movement of articulators to anticipate the next sound or perseverating movement from the last sound. Because of coarticulation, each phone differs slightly depending on the previous and following phone. Thus if we just concatenated phones we would have very large discontinuities at the boundaries.

In a diphone, we model this coarticulation by including the transition to the next phone inside the unit. The diphone [w-eh], for example, includes the transition from the [w] phone to the [eh] phone. Because a diphone is defined from the middle of one phone to the middle of the next, when we concatenate the diphones, we are concatenating the middle of phones, and the middle of phones tends to be less influenced by the context. Figure 8.15 shows the intuition that the beginning and end of the vowel [eh] have much more movement than the center.

8.4.1 Steps for Building a Diphone Database

Building a diphone database requires six steps:

1. Create a **diphone inventory**.
2. Recruit a speaker.
3. Create a text for the speaker to read for each diphone.

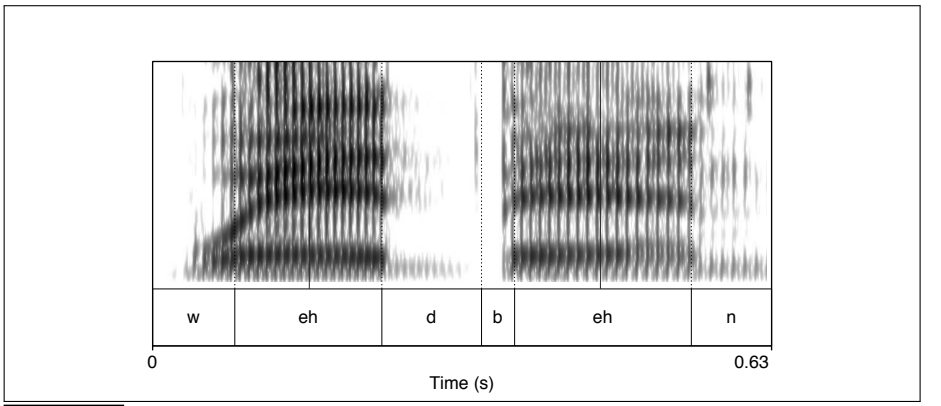


Figure 8.15 The vowel [eh] in different surrounding contexts, in the words *wed* and *Ben*. Notice the differences in the second formants (F2) at the beginning and end of the [eh], but the relatively steady state portion in the middle at the center mark.

4. Record the speaker reading each diphone.
5. Segment, label, and pitch-mark the diphones.
6. Excise the diphones.

What is the inventory of diphones that we need for a system? If we have 43 phones (like the AT&T system of Olive et al. (1998)), there are $43^2 = 1849$ hypothetically possible diphone combinations. Not all of these diphones can actually occur. For example, English **phonotactic** constraints rule out some combinations; phones like [h], [y], and [w] can only occur before vowels. In addition, some diphone systems don't bother storing diphones if there is no possible coarticulation between the phones, such as across the silence between successive voiceless stops. The 43-phone system of Olive et al. (1998) thus has only 1162 diphones rather than the 1849 hypothetically possible set.

Voice talent
Voice

Next, we recruit our speaker, often called a **voice talent**. The database of diphones for this speaker is called a **voice**; commercial systems often have multiple voices, such as one male and one female voice.

We'll now create a text for the voice talent to say, and record each diphone. The most important thing in recording diphones is to keep them as consistent as possible; if possible, they should have constant pitch, energy, and duration, so that they are easy to paste together without noticeable breaks. We promote consistency by enclosing each diphone to be recorded in a **carrier phrase**. By surrounding the diphone with other phones, we keep utterance-final lengthening or initial phone effects from making any diphone louder or quieter than the others. We'll need different carrier phrases for consonant-vowel, vowel-consonant, phone-silence, and silence-phone sequences. For example, a consonant vowel sequence like [b aa] or [b ae] could be embedded between the syllables [t aa] and [m aa]:

Carrier phrase

pause t aa b aa m aa pause
pause t aa b ae m aa pause
pause t aa b eh m aa pause
...

If we have an earlier synthesizer voice lying around, we can use that voice to read the prompts out loud and have our voice talent repeat after the prompts. This is another way to keep the pronunciation of each diphone consistent. It is also important to use a high-quality microphone and a quiet room or, better, a sound booth.

Once we have recorded the speech, we need to label and segment the two phones that make up each diphone, usually by running a speech recognizer in **Forced alignment** mode. In forced alignment mode, a speech recognition is told exactly what the phone sequence is; its job is just to find the exact phone boundaries in the waveform. Speech recognizers are not completely accurate at finding phone boundaries, so usually the automatic phone segmentation is hand-corrected.

Forced alignment

We now have the two phones (for example, [b aa]) with hand-corrected boundaries. There are two ways we can create the /b-aa/ diphone for the database. One method is to use rules to decide how far into the phone to place the diphone boundary. For example, for stops, we place the diphone boundary 30% of the way into the phone. For most other phones, we place the diphone boundary 50% into the phone.

A more sophisticated way to find diphone boundaries is to store all of both phones and wait to excise the diphones until we are know what phone we are about to concatenate with. In this method, known as **optimal coupling**, we take the two (complete, uncut) diphones we need to concatenate and check every possible cutting point for each diphones, choosing the two cutting points that would make the final frame of the first diphone acoustically most similar to the end frame of the next diphone (Taylor and Isard, 1991; Conkie and Isard, 1996). Acoustic similarity can be measured by **cepstral similarity**, as defined in Section 9.3.

Optimal coupling

8.4.2 Diphone Concatenation and TD-PSOLA for Prosody

We are now ready to see the remaining steps for synthesizing an individual utterance. Assume that we have completed text analysis for the utterance, have arrived at a sequence of diphones and prosodic targets, and have also grabbed the appropriate sequence of diphones from the diphone database. Next, we need to concatenate the diphones and then adjust the prosody (pitch, energy, and duration) of the diphone sequence to match the prosodic requirements from the intermediate representation.

Given two diphones, what do we need to do to concatenate them successfully? If the waveforms of the two diphones edges across the juncture are very different, a perceptible **click** will result. Thus, we need to apply a windowing function to the edge of both diphones so that the samples at the juncture have low or zero amplitude. Furthermore, if both diphones are voiced, we need to ensure that the two diphones are joined **pitch-synchronously**. This means that the pitch periods at the end of the first diphone must line up with the pitch periods at the beginning of the second diphone; otherwise, the resulting single irregular pitch period at the juncture is perceptible as well.

Click

Pitch-synchronous

Now given our sequence of concatenated diphones, how do we modify the pitch and duration to meet our prosodic requirements? It turns out there is a simple algorithm for doing this, called **TD-PSOLA** (**Time-Domain Pitch-Synchronous OverLap-and-Add**).

TD-PSOLA

As we just said, a **pitch-synchronous** algorithm is one in which we do something at each pitch period or **epoch**. For such algorithms it is important to have accurate pitch markings: measurements of exactly where each pitch pulse or **epoch** occurs. An epoch can be defined by the instant of maximum glottal pressure, or alternatively by the instant of glottal closure. Note the distinction between **pitch marking** or **epoch detection** and **pitch tracking**. Pitch tracking gives the value of F0 (the average cycles per second of the glottis) at each particular point in time, averaged over a neighborhood. Pitch marking finds the exact point in time at each vibratory cycle at which the vocal folds reach some specific point (epoch).

Pitch marking
Pitch tracking

Epoch labeling can be done in two ways. The traditional way, and still the most accurate, is to use an **electroglottograph** or **EGG** (often also called a **laryngograph** or **Lx**). An EGG is a device that straps onto the (outside of the) speaker's neck near the larynx and sends a small current through the Adam's apple. A transducer detects whether the glottis is open or closed by measuring the impedance across the vocal folds. Some modern synthesis databases are still recorded with an EGG. The problem with using an EGG is that it must be attached to the speaker while he or she is recording the database. Although an EGG isn't particularly invasive, it is still annoying, and the EGG must be used during recording; it can't be used to pitch-mark speech that has already been collected. Modern epoch detectors are now approaching such a level of accuracy that EGGs are no longer used in most commercial TTS engines. Algorithms for epoch detection include Brookes and Loke (1999), and Veldhuis (2000).

Electroglotto-graph
EGG
Laryngograph
Lx

Given an epoch-labeled corpus, the intuition of TD-PSOLA is that we can modify the pitch and duration of a waveform by extracting a frame for each pitch period (windowed so that the frame doesn't have sharp edges) and then recombining these frames in various ways by simply overlapping and adding the windowed pitch period frames (we introduce the idea of windows in Section 9.3.2). The idea that we modify a signal by extracting frames, manipulating them in some way, and then recombining them by adding up the overlapped signals is called the **overlap-and-add** or **OLA** algorithm; TD-PSOLA is a special case of overlap-and-add in which the frames are pitch synchronous and the whole process takes place in the time domain.

Overlap-and-add
OLA

For example, to assign a specific duration to a diphone, we might want to lengthen the recorded master diphone. To lengthen a signal with TD-PSOLA, we simply insert extra copies of some of the pitch-synchronous frames, essentially duplicating a piece of the signal. Figure 8.16 shows the intuition.

TD-PSOLA can also be used to change the F0 value of a recorded diphone to give a higher or lower value. To increase the F0, we extract each pitch-synchronous frame from the original recorded diphone signal, place the frames closer together (overlapping them), with the amount of overlap determined by the desired period and hence frequency, and then add up the overlapping signals to produce the final signal. But note that by moving all the frames closer together, we make the signal shorter in time! Thus to change the pitch while holding the duration constant, we need to add duplicate frames.

Figure 8.17 shows the intuition; in this figure we have explicitly shown the extracted pitch-synchronous frames that are overlapped and added; note that the frames moved closer together (increasing the pitch) while extra frames have been added to hold the duration constant.

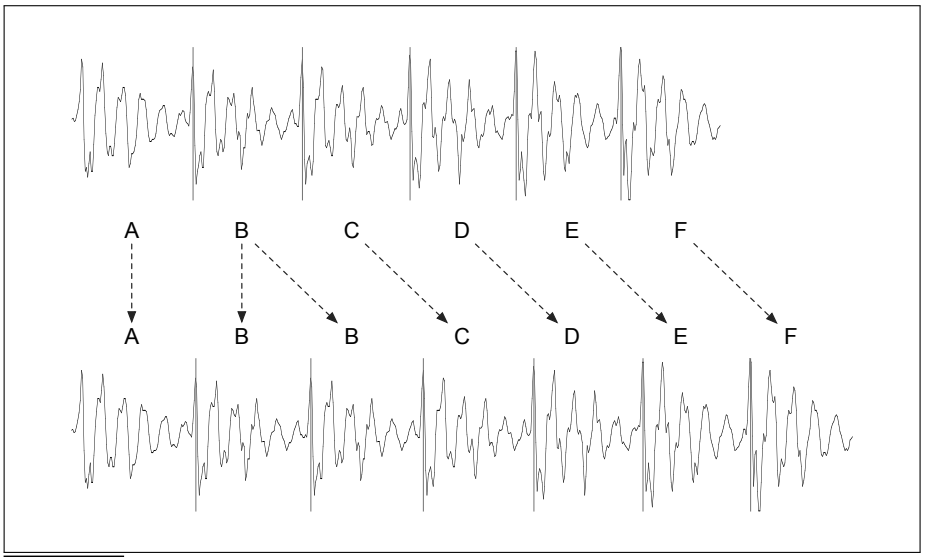


Figure 8.16 TD-PSOLA for duration modification. Individual pitch-synchronous frames can be duplicated to lengthen the signal (as shown here), or deleted to shorten the signal.

8.5 Unit Selection (Waveform) Synthesis

Diphone waveform synthesis suffers from two main problems. First, the stored diphone database must be modified by signal process methods like PSOLA to produce the desired prosody. Any kind of signal processing of the stored speech leaves artifacts in the speech that can make the speech sound unnatural. Second, diphone synthesis captures only the coarticulation due to a single neighboring phone. But there are many more global effects on phonetic realization, including more distant phones, syllable structure, the stress patterns of nearby phones, and even word-level effects.

Unit selection synthesis

For this reason, modern commercial synthesizers are based on a generalization of diphone synthesis called **unit selection synthesis**. Like diphone synthesis, unit selection synthesis is a kind of concatenative synthesis algorithm. The word **unit** means any stored piece of speech that is concatenated together to form an output. The intuition of unit selection synthesis is that we can store units of different sizes, which can be much larger that diphones. Unit selection thus differs from classic diphone synthesis in two ways:

1. In diphone synthesis, the database stores one copy of each diphone; in unit selection, the database is many hours long, with many copies of each diphone.
2. In diphone synthesis, the prosody of the units is modified by PSOLA or similar algorithms; in unit selection no (or minimal) signal processing is applied to the concatenated units.

The strengths of unit selection are due to the large unit database. In a sufficiently large database, entire words or phrases of the utterance we want to synthesize may already be present in the database, resulting in an extremely natural waveform for these

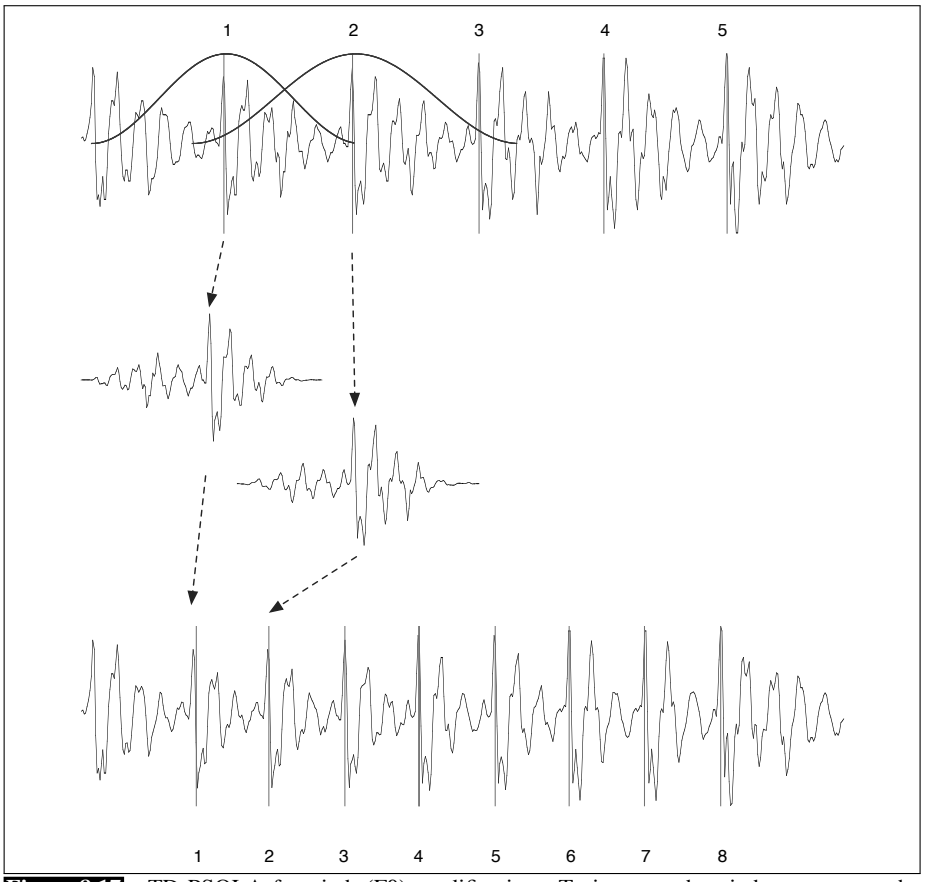


Figure 8.17 TD-PSOLA for pitch (F0) modification. To increase the pitch, we extract the individual pitch-synchronous frames, Hanning-window them, move them closer together and then add them up. To decrease the pitch, we move the frames further apart. Increasing the pitch will result in a shorter signal (since the frames are closer together), so we also need to duplicate frames if we want to change the pitch while holding the duration constant.

words or phrases. Thus we implicitly create larger units by selecting diphones that are consecutive in the database. In addition, in cases in which we can't find a large chunk and have to back off to individual diphones, many copies of each diphone make it more likely that we will find one that will fit in naturally.

The architecture of unit selection can be summarized as follows. We are given a large database of units; let's assume these are diphones (although it's also possible to do unit selection with other kinds of units such as half-phones, syllables, or half-syllables). We are also given a characterization of the target 'internal representation', that is, a phone string together with features such as stress values, word identity, F0 information, as described in Fig. 8.1 on page 250.

The goal of the synthesizer is to select from the database the best sequence of diphone units that corresponds to the target representation. What do we mean by the "best" sequence? Intuitively, the best sequence would be one in which