TÜRKÇE KOMUTLARI TANIYAN SES TANIMA SİSTEMİ GELİŞTİRİLMESİ

Pamukkale Üniversitesi
Fen Bilimleri Enstitüsü
Yüksek Lisans Tezi
Bilgisayar Mühendisliği Anabilim Dalı

Murat Kemal BAYGÜN

Danışman: Yard. Doç. Dr. A. Kadir YALDIR

Şubat 2006
DENİZLİ

YÜKSEK LİSANS TEZİ ONAY FORMU

Murat Kemal BAYGÜN tarafından Yard. Doç. Dr. A. Kadir YALDIR yönetiminde hazırlanan "**Türkçe Komutları Tanıyan Ses Tanıma Sistemi Geliştirilmesi**" başlıklı tez tarafımızdan okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisans Tezi olarak kabul edilmiştir.

Yard. Doç. Dr. Serdar İPLİKÇİ
Jüri Başkanı

Yard. Doç. Dr. A. Kadir YALDIR
Jüri Üyesi (Danışman)

Yard. Doç. Dr. Sezai TOKAT
Jüri Üyesi

Pamukkale Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun
..../..../..... tarih ve ..... sayılı kararıyla onaylanmıştır.

Prof. Dr. Mehmet SARIGÖL
Müdür

Bu tezin tasarımı, hazırlanması, yürütülmesi, araştırmaların yapılması ve bulgularının analizlerinde bilimsel etiğe ve akademik kurallara özenle riayet edildiğini; bu çalışmanın doğrudan birincil ürünü olmayan bulguların, verilerin ve materyallerin bilimsel etiğe uygun olarak kaynak gösterildiğini ve alıntı yapılan çalışmalara atfedildiğini beyan ederim.

İmza : 
Öğrenci Adı Soyadı : Murat Kemal BAYGÜN

TEŞEKKÜR

Bu tez çalışmasında öncelikle beni yetiştiren ve üzerimde haklarını ödeyemeyeceğim en büyük emekleri bulunan annem Aynur BAYGÜN'e ve rahmetli babam Mehmet Özhan BAYGÜN'e (Nur içinde yatsın); her zaman bana güvenmiş ve destek olmuş olan ablam Arzu BAYGÜN'e; çalışmalarım sırasında gösterdiği sabır ve anlayış, verdiği moral ve manevi destek için eşim Sibel BAYGÜN'e ve çalışmalarıma destekten çok engel olmasına rağmen, bir küçük gülücükle de olsa moralimi yükseltmiş olmasından dolayı biricik kızım Ceren BAYGÜN'e; yardımlarını benden esirgemeyen Tez Danışmanım Yard. Doç. Dr. A. Kadir YALDIR'a; öneri ve katkılarından dolayı Yard. Doç. Dr. Serdar İPLİKÇİ'ye ve Yard. Doç. Dr. Sezai TOKAT'a; yine manevi destekleri için Egecom®'daki tüm çalışma arkadaşlarıma ve burada adını saymadığım ancak üzerimde emeği bulunan herkese gönülden teşekkür ederim.

ÖZET

TÜRKÇE KOMUTLARI TANIYAN SES TANIMA SİSTEMİ GELİŞTİRİLMESİ

Baygün, Murat Kemal
Yüksek Lisans Tezi, Bilgisayar Mühendisliği ABD
Tez Yöneticisi: Yard. Doç. Dr. A. Kadir YALDIR

Şubat 2006, 69 Sayfa

İnsanlar arası iletişim, hızlı ve etkin şekilde sesli olarak sağlanmaktadır. Ses tanıma sistemleri de, buradan yola çıkarak, insan – bilgisayar arası iletişimi daha etkin sağlayabilmek için konuşma dilini kullanmayı amaç edinir. Son yıllarda ses tanıma teknolojileri büyük önem kazanmıştır. Ses tanıma teknolojileri kullanılarak çoğunlukla kişi tanımlama ve konuşma dilini anlama olmak üzere birçok uygulama üzerinde çalışılmaktadır. Ancak Türkçe ses tanıma üzerine çok fazla çalışma bulunmamaktadır. Bu çalışmada ses tanıma sistemleri genel olarak incelenmiş, kullanılan yöntemler araştırılmış ve ses tanıma teknolojileri kullanılarak Türkçe komutları tanıyan bir Ses Tanıma Sistemi geliştirilmeye çalışılmıştır.

Ses tanıma süreci, sesin kaydedilmesi ve ifadenin saptanması; sesin işlenmesi; karşılaştırma ve eşleştirme yapılması; son olarak saptanan ifadeye karşılık gelen işlevin gerçekleştirilmesi aşamalarından oluşur. Bu aşamalardan her biri için, geliştirilmiş ve kullanılan birçok teknik bulunmaktadır. Bu çalışmada sesin kaydedilmesi aşamasında sözcük kesimlerinin saptanması için bir çerçevedeki sıfırı geçiş sayısı ve RMS (Root Mean Square) değerinden faydalanılmıştır. Sesin işlenmesi aşamasında kodlayıcı olarak LPC (Linear Predictive Coding) kullanılmış olup karşılaştırma ve eşleştirme aşamasında ise LPC parametreleri üzerinde DTW (Dynamic Time Warping) uygulanmıştır.

Geliştirilen bu ses tanıma sistemi, kişiye bağımlı, sözcük tabanlı bir komut – kontrol sistemi olarak oluşturulmuştur. Sözlük kapasitesi, toplam 15 kelime ile sınırlandırılmış olup rakamlar ve beş komuttan oluşmaktadır. Öncelikle, bu kelimeler için ses kaydı yapılıp, LPC ile kodlanmıştır. Her bir kelime için bu şekilde şablonlar oluşturulmuştur. Çalışma anında, kayıt esnasında saptanan ses, LPC ile kodlanarak, tüm kayıtlı şablonlarla DTW algoritması kullanılarak karşılaştırma gerçekleştirilmiştir. Karşılaştırma sonucunda elde edilen en yakın şablon ile kelime eşleştirilmesi gerçekleştirilerek algılanan komut, geliştirilen uygulama ara yüzüne yansıtılmaktadır.

Anahtar Kelimeler: Ses Tanıma, Sözcük Tabanlı Ses Tanıma, Türkçe Ses Tanıma, LPC, DTW

ABSTRACT

DEVELOPMENT OF A SPEECH RECOGNITION SYSTEM RECOGNISING TURKISH COMMANDS

Baygün, Murat Kemal
M. Sc. Thesis in Computer Engineering
Supervisor: Assoc. Prof. Dr. A. Kadir YALDIR

February 2006, 69 Pages

Communication between human beings is done quickly and more efficiently by voice. Speech recognition systems therefore, aim to improve the human - computer interaction by using spoken language. In recent years speech recognition techniques gained more importance. By using speech recognition techniques, many applications are being studied on for speaker identification and speech recognition. However, there are not so many studies on Turkish speech recognition. With this study, speech recognition systems have been examined generally, the methods used have been investigated and a Speech Recognition System has been tried to being developed for recognizing Turkish commands by using speech recognition techniques.

Speech recognition process consists of speech recording and determining utterance, signal processing, comparison and matching; and finally processing the function to response the recognized word. For all these phases, there are many techniques that have been investigated and are being used. In this study, zero pass count and RMS (Root Mean Square) are used to determine the utterances in speech recording phase. LPC (Linear Predictive Coding) is used in signal processing phase, and DTW (Dynamic Time Warping) is used in comparison and matching phase on LPC coefficients.

The speech recognition system that has been developed is designed as a speaker dependent word based command – control system. The dictionary is limited by 15 words and consists of numbers and five commands. First; speech recording is done and speech is coded with LPC for these words. For all these words, patterns are generated. During run-time, the determined utterance while recording is being coded with LPC and compared by using DTW algorithm with the all word patterns saved. After comparison, the nearest matching pattern is selected, and the recognized command is reflected on the developed application interface.

Keywords: Speech Recognition, Word Based Speech Recognition, Turkish Speech Recognition, LPC, DTW

Asst. Prof. Dr. Serdar İPLİKÇİ
Asst. Prof. Dr. A. Kadir YALDIR
Asst. Prof. Dr. Sezai TOKAT

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGE VE KISALTMALAR DİZİNİ

ADPCM	Adaptive Differential Pulse Code Modulation
APCM	Adaptive Pulse Code Modulation
DTW	Dynamic Time Warping
HMM	Hidden Markov Model
LPC	Linear Predictive Coding
PCM	Pulse Code Modulation
PLP	Perceptual Linear Prediction
YSA	Yapay Sinir Ağı

1. GİRİŞ

İnsanlar, günlük hayatta birçok iletişim şekli kullanırlar. Günümüzde en yaygın iletişim şekilleri; elektronik posta ve kısa mesaj servisleri gibi yazılı iletişimler ile cep telefonları, telefonlar ve yüz yüze yapılan sesli iletişimlerdir. Bu iletişim yöntemleri incelendiğinde en hızlı, kolay ve karşılıklı daha etkin iletişim sağlaması bakımından en önemli yöntem sesli iletişimdir. Bunun nedeni ise kişinin karşısındaki kişi ile her ikisinin de bildiği doğal konuşma dilini kullanarak etkileşim sağlıyor olmasıdır. Bir diğer neden ise karşılıklı sürekli bir etkileşim halinde iletişimin sağlanmasıdır.

Ses tanıma sistemlerinin, amaçlarından önemli bir tanesi, insan - bilgisayar iletişimini, kullanıcının en yaygın olarak kullandığı, sesli iletişimle sağlamak ve insanların, işlerini birçok alanda kolaylaştıran bilgisayar sistemlerinin daha yaygın kullanımına imkan tanımaktır. Bilgisayar kullanımı, öğrenildiğinde ne kadar kolay olsa da, bazıları için hala korkutucudur, kullanımı çok zordur ve öğrenilmesi imkansız görünmektedir. Ses tanıma sistemlerinin kullanılması ile, insan - bilgisayar iletişimi için, kullanıcının zaten alışkın olduğu en yaygın iletişim aracını, yani doğal konuşma dilini kullandırması bakımından bilgisayar kullanımı daha kolay olacaktır. Bunun dışında; hızlı ve uzaktan veri girişi, hareket serbestliği gibi faydalarının yanında çalışma kıyafetleri ve kullanılması zorunlu araç - gereç yüzünden veya tıbbi yetersizliklerinden dolayı klavye, fare, tablet gibi veri giriş cihazları kullanma imkanı olmayan kişilere bilgisayar kullanabilme imkanı tanıması ses tanıma sistemlerinin önemini arttırmaktadır.

İnsanlar arası sesli iletişim beyinde, konuşma seslerini üreten sinirsel hareketleri aktive eden bir düşünce ve niyetle başlar. Dinleyici, konuşmayı beynin anlayacağı sinirsel sinyallere dönüştüren işitme sistemi yardımıyla alır (Huang vd 2001). Ses tanıma sistemleri, insanlar arası sesli iletişim modelinde yer alan dinleyicinin yaptığı işlemleri yapay olarak gerçekleştirme prensibi ile çalışmaktadırlar.

Tüm bunlara ileriki bölümlerde değinilecek olup, bunun öncesinde çalışmanın anlaşılabilirliğinin artması bakımından, aşağıdaki terimlerin verilmesi ve açıklanması yerinde olacaktır.

İnsan Sesi – Konuşma: Basit olarak ses, akciğerlerden havanın dışarı atılması sonucunda oluşan hava akımının, ses sisteminde bir yerlerde sıkıştırılarak karıştırılmasından yayılan akustik dalgalardır (Rabiner ve Schafer 1978).

Fonem: Anlam içeren ve değişmesi ile dildeki bulunduğu bir kelimenin anlamını değiştiren, en küçük ses birimidir.

Ses Tanıma: Mikrofon ya da telefon tarafından alınmış akustik bir sinyalin, kelime kümesine olan çevrim işlemi olarak tanımlanabilir (Cole vd 1995).

Ses Analizi: Üretim metodunun da dikkate alınmasıyla, sesin özelliklerinin çıkarılabilmesi için yapılan işlemlerdir.

Ses Kodlama: İşlem sırasında gözlenebilir bir sinyal kalitesi kaybı olmadan, sinyalin sayısal gösterimi olan bit oranını azaltmayı amaç edinen kodlama algoritmaları.

Ses Sentezi: Metinden yapay olarak ses üretme işlemi olarak tanımlanabilir.

2. SES TANIMA SİSTEMLERİ

Çalışmanın bu bölümünde ses tanıma sistemlerinin sağladığı faydalardan bahsedilmiştir. Ardından çalışma ve uygulama alanlarına yer verilmiş, sonrasında ise ses tanıma sistemlerindeki temel sorunlara değinilmiştir.

2.1. Ses Tanıma Sistemlerinin Sağladığı Faydalar

Ses tanıma sistemleri, içlerinde üretim sektörü de bulunan birçok alanda; başta kullanım kolaylığı olmak üzere birçok fayda sağlar. Bu faydalar aşağıda özetlenmektedir:

- Kullanım kolaylığı: Ses tanıma sistemleri, veri giriş aracı olarak mikrofon ve kimi zaman da telefon kullanırlar. Veri kaynağı, insanın alışkın olduğu özel çaba gerektirmeyen konuşma olduğundan veri girişi ve kullanımı oldukça kolaydır. Klavye, fare ve tablet gibi veri girişi için kullanılan geleneksel yöntemlerde olduğu gibi özel çaba gerektirmez.

- Veri toplama hızı: Ses tanıma sistemleri ile veri girişi, konuşma dilini kullandırması bakımından, klavyeyle veri girişine oranla daha hızlı olacaktır. Doğal olarak, hızlı veri girişi sonucunda veri girişi işlemi daha kısa sürelerde tamamlanacaktır. Bu da özellikle üretim sektörü göz önüne alındığında; asıl işi veri giriş operatörlüğü değil de üretmek olan personelin üretime daha çok vakit ayırmasına olanak sağlayacaktır.

- Hareket serbestliği: Zaman zaman, yaptığı iş ve mesleği bakımından gerek giyimi, gerekse mesleğini icra ederken kullanılması zorunlu araç - gereç nedeniyle, çalışma anında elleri serbest olmayan kişilerin veri girişine ihtiyaç duyulabilmektedir. Bu tür durumlarda, klavye ve fare ile veri girişi gibi yaygın yöntemlerin kullanımı genelde mümkün olmamaktadır. Veri girişi çok zorunlu olduğunda ise bu yaygın yöntemler kullanılarak işe ayrılması gerekli olan önemli bir zaman veri girişi için kullanılmaktadır. Bu da doğal olarak iş kaybına yol açmaktadır. Ses tanıma sistemleri, konuşma ile veri girişi sağladığından;

yaka mikrofonu veya kulaklıklı mikrofonlar kullanılması ile bu tür personele hareket serbestliği sağlayacaktır.

- Uzaktan veri giriş imkanı: Telefonun, uzak bir nokta ile sesli iletişim sağlaması avantajı kullanılarak, ses tanıma sistemlerinde veri giriş aracı olarak kullanılabilmektedir. Bu sayede uygun sistemlerle veri girişi uzaktan yapılabilmektedir. Bu tür veri girişleri, telefonla destek ve servis hizmeti veren birçok firma tarafından gün geçtikçe yaygınlaşarak kullanılmaktadır.

2.2. Kullanım Alanları

Ses tanıma sistemlerinin başlıca uygulama alanları Cook (2002) tarafından; dikte, komut - kontrol, telefonla hizmet, giyim kısıtlamaları, tıbbi yetersizlikler ve gömülü uygulamalar olarak verilmiştir. Bu uygulama alanları aşağıda özetlenmiş ve çalışma prensipleri aktarılmıştır.

2.2.1. Dikte – yazdırma

Ses tanıma sistemlerinin en yaygın uygulama alanlarından biridir. Bilindiği gibi bilgisayar ortamına veri girişi için yaygın olarak klavye, fare ve tablet gibi cihazlar kullanılmaktadır. Özellikle uzun metin girişlerinde, (sözleşme, anlaşma metni, toplantı tutanakları vs.) veri girişi uzun zaman almaktadır. Kimi zaman bu tür girişlerde dikte yöntemi kullanılmakta ve bir personel bu iş için görevlendirilmektedir. Çok hızlı el yazısı yazan veya klavye kullanan kişi de olsa çoğu zaman toplantı akışında tüm konuşulanları dikte etme imkanı bulamamaktadır. Bu tür yazdırma işlevlerinde ek araç ve yöntemlere ihtiyaç duyulmaktadır. Konuşma esnasında bir kayıt cihazı çalıştırılmakta veya dikte eden kişi stenografi gibi kısaltmaya dayalı metotlarla bu işi gerçekleştirmektedir. Ancak stenografi için özel eğitim gerekmektedir. Her iki yöntemde de, asıl döküman sonradan oluşmaktadır. Kayıt cihazından tekrar tekrar dinlenerek, yazıya dökülmekte veya stenografi ile yazılmış metin dökümana çevrilmektedir.

Ses tanıma sistemlerinin çalışma ve kullanım alanlarından ilki olarak bu konu benimsenmiştir. Bu alanda özellikle İngilizce konuşma dili için başarılı sayılacak, ticari birkaç uygulama da mevcuttur (Ör. Microsoft® Dictation, DragonDictate®, IBM® ViaVoice gibi).

Temel olarak bu tür ses tanıma sistemlerinde amaç; mikrofondan alınan sesli ifade verilerini, metin düzenleyicisine yazdırmaktır.

2.2.2. Komut – kontrol

Ses tanıma sistemlerinin, bir diğer önemli çalışma ve uygulama alanı olarak komut – kontrol sistemleri görülmektedir. Bu tür ses tanıma sistemlerinde bir komut kümesi ve bu kümedeki her bir komuta karşılık gelen işlevler kümesi mevcuttur. Amaç mikrofonla alınan sesli ifade verilerini komutlar içinde tespit etmek ve tespit edilen bu komuta karşılık gelen işlevi gerçekleştirmektir.

2.2.3. Telefonla hizmet

Ses tanıma sistemleri, telefon bankacılığı gibi telefonla hizmet ve servis veren şirketlerce de kullanılmaktadır ve gün geçtikçe de bu alanda kullanımı yaygınlaşmaktadır. Telefonla hizmet veren firmalarda, klasik yöntemde; genelde müşteriye uzun uzun menüler dinletilmekte, her bir işlev için tuş karşılıkları verilmekte ve çok işlevli menülerde tuş seçimine karşılık yeni menüler ve tuşlar şeklinde birkaç sıralı işlem ardı ardına uygulanmaktadır. Bu da müşteriye hizmet süresini uzatmakta ve servis alacak kişinin bazen menüler içinde kaybolmasına yol açmaktadır.

Bu tür ses tanıma sistemlerinin amacı; komut – kontrol sistemlerindeki temele dayanır ve telefon aracılığı ile servis almak isteyen kişiyi, aldığı komutlar ile doğru kanala ya da kişiye yönlendirmektir.

Yine benzer şekilde, santrallerde kullanılan otomatik cevaplama yapan ses robotunun, ses tanıma sistemi ile desteklenerek daha akıllı hale getirilmesi de mümkün kılınabilir. Burada genel senaryo şu şekildedir: Ses robotu telefonu yanıtlar, kısa açılış konuşmasından sonra, ilk olarak arayan kişinin ismini, daha sonra görüşmek istediği kişinin ismini alır. Arayan kişinin doğal konuşması ile sesli olarak verdiği isimleri ses tanıma sisteminden geçirerek, aranan kişiyi tespit eder, dahili telefonunu rehberden (veritabanından) bulur. Aranan kişiye telefonu aktararak, arayan kişinin ismini iletir ve telefonu aktarır.

2.2.4. Donanım – giyim kısıtlamaları

Yapılan meslek, çalışma mekanı gibi ellerin serbestliğinin olmadığı durumlarda, veri girişi zorunlu, fakat klasik yöntemlerle veri giriş imkanı bulunmadığı zamanlar olabilmektedir. Klasik yöntemlerle veri girişi, iş kaybına yol açabilmektedir. Bu durumlarda, hareket serbestliği avantajı ile ses tanıma sistemleri veri giriş amacıyla kullanılabilmektedir.

2.2.5. Tıbbi yetersizlikler

Tıbbi yetersizliklerden dolayı ellerini ve dolayısıyla klasik yöntemlerle veri girişi amacıyla klavye, fare, tablet gibi cihazları kullanma imkanı bulunmayan kişilere, ses tanıma sistemlerinden faydalanılması suretiyle veri girişi ve bilgisayar kullanma yeteneği sağlanabilmektedir. Bu tür kişilere, bazı ihtiyaçlarını bu türde özel sistemler tasarlamak suretiyle giderebilme imkanı verilebilmektedir. Özel donanımlar ve kurulabilecek sistemler sayesinde, bu durumda olan bir kişi sesli komutlarla, klimanın ısısını ayarlayabilmektedir, televizyonu açıp kapayabilmektedir, müzik setini kumanda edebilmektedir.

2.2.6. Gömülü uygulamalar

Bu tür uygulamaların en belirgin örneği, sesli arama imkanı sağlayan cep telefonları olarak verilebilir. Sesli arama sistemi, telefon rehberindeki bir kişinin telefon numarası ile kaydedilmiş bir ses etiketinin ilişkilendirilmesi prensibine göre çalışmaktadır. Çalışma anında ses etiketi seslendirildiğinde, kayıtlı ses etiketleri taranarak en uygunu seçilip ilişkilendirilmiş numara rehberden alınarak sesli arama işlevi gerçekleştirilmiş olur.

2.3. Ses Tanıma Sistemlerinin Sınıflandırılması

Ses tanıma sistemlerinin sınıflandırılması, sistemin yapısına göre farklı açılardan yapılabilmektedir.

2.3.1. Sesin sürekliliğine göre

İlk sınıflandırma sesin, yani konuşmanın sürekli olup olmadığına göre yapılmaktadır. Genelde ayrık ve sürekli konuşma olarak iki kısımda sınıflandırılmasına

rağmen: bazen bu sınıflandırmaya, bağlı kelimeler ve doğal konuşma da dahil edilmektedir. Cook (2002) tarafından sınıflandırma şu şekilde yapılmıştır:

- Ayrık kelimeler: Bu tür ses tanıma sistemlerinde, sistem tarafından kelimeler arası kısa boşluklar beklenir.

- Sürekli konuşma: Ses tanıma sistemlerinin bu türünde, konuşmacının neredeyse en doğal halleriyle konuşmaları beklenir. Genelde dikte – yazdırma uygulamaları bu sınıflandırmaya girer.

- Bağlı kelimeler: Ayrık kelimelere benzer. Ayrık kelimelere göre kelimeler arası boşluklar daha kısadır. Konuşmacının daha kısa boşluklarla kelimeleri seslendirmesini destekleyen sistemler bu sınıfa girer.

- Doğal konuşma: Sürekli konuşma sınıflandırmasına çok benzer. Fakat insan doğasından kaynaklanan, doğal konuşma özelliklerini de algılayabilecek düzeyde geliştirilmişlerdir. Bu tür doğal konuşma özelliklerine, kelimeler arasında, düşünürken ağızdan çıkan 'mm', 'hımmm' tarzı beklemeler örnek olarak verilebilir.

2.3.2. Konuşmacıya bağımlılığına göre

Bu türde sınıflandırma, kişiye bağımlı ve kişiden bağımsız olarak iki şekilde yapılmaktadır:

- Kişiye bağımlı: Bu tür ses tanıma sistemlerinde farklı bir konuşmacının sesinin tanınması gerektiğinde, sistemde kayıtlı bulunan ve tanıma için temel olarak alınan referans şablonlarının güncellenmesi gerekmektedir.

- Kişiden bağımsız: Bu tür ses tanıma sistemlerinde ise herhangi bir şablon güncellenmesine gerek duymaksızın herhangi bir kişinin sesinin tanınmasına imkan verilmektedir.

Kişiden bağımsız ses tanıma sistemleri, kullanışlılık olarak daha etkin görünmesine karşın; bu tür sistemlerin geliştirilmesi, kişiye bağımlı sistemlerin geliştirilmesine kıyasla daha zordur. Ses, kişiden kişiye, hatta aynı kişinin farklı zamanlarda seslendirmesiyle bile değişiklik gösterebilmektedir. Bu nedenle farklı kişilerin seslerini

tanıma amacıyla genel şablonların hazırlanması ve genelleştirilmiş bir sistemin tasarımı uzun uğraşlar gerektirmektedir.

2.3.3. Ses tanıma sisteminde temel alınan birime göre

Bu sınıflandırma türünde, ses tanıma sistemleri, tanıma için temel alınan en küçük birim büyüklüğüne göre değerlendirilirler.

- Sözcük tabanlı: Bu tür ses tanıma sistemlerinde tanımanın en küçük birimi olarak sözcükler alınmıştır. Oluşturulan referans şablonları sözcüklerdir. Doğruluk derecesi daha yüksek olmasına rağmen şablonlar için en küçük birim sözcükler olarak alındığından sistem gereksinimleri daha yüksektir. Sözcük kapasitesinin sınırlı olduğu komut – kontrol gibi sistemlerde daha çok tercih edilirler.

- Fonem tabanlı: Bu tür ses tanıma sistemlerinde tanımanın en küçük birimi olarak fonemler alınır. En küçük birim fonem olarak alındığından şablon sayısı, oluşturulması, saklanması ve işlenmesi çok az bir sistem gereksinimiyle karşılanabilir. Fonemler arası geçişlerin, fonemlerin başlangıç ve bitişinin tespitinin zor olması; bu tür sistemlerdeki en büyük problemdir. Doğruluk oranı sözcük tabanlı sistemlere göre daha düşük olmasına rağmen; sistem gereksinimi daha az olduğundan, hata durumlarında geri dönüşlerle hata düzeltmeleri yapılabilecek zaman vardır.

2.4. Ses Tanıma Sistemlerindeki Temel Sorunlar

Ses tanıma sistemlerinin, yukarıda bahsedilmiş faydaları, çalışma ve uygulama alanları, hali hazırda mevcut ve öngörülüyor ise de temel bazı sorunlar da mevcuttur. Bunlardan en belirgini, veri giriş kaynağı olarak alınan sesin ortam gürültüsünden çok kolay etkilenip bozulabilmesidir. Bu bozulmanın nedeni ortam gürültüsü, telefon hatları için hattaki parazit ve çeşitli dış kaynaklar olarak verilebilir.

Bu problemlerin giderilmesi için çeşitli teknikler geliştirilmiş ve geliştirilmeye devam edilmektedir. Ses sinyali üzerindeki gürültünün temizlenmesi için filtreler geliştirilmiştir. Filtreler yardımıyla sinyal düzeltmeleri gerçekleştirilebilmektedir.

Diğer bir problem ise; sesin, yani konuşmanın kişiden kişiye, hatta aynı kişi için de olsa farklı zamanlarda seslendirilişinde, duygu durumuna göre bile değişkenlik göstermesidir. Aynı zamanda şive ve aksan farklılıkları da bu değişkenliğin bir etkenidir.

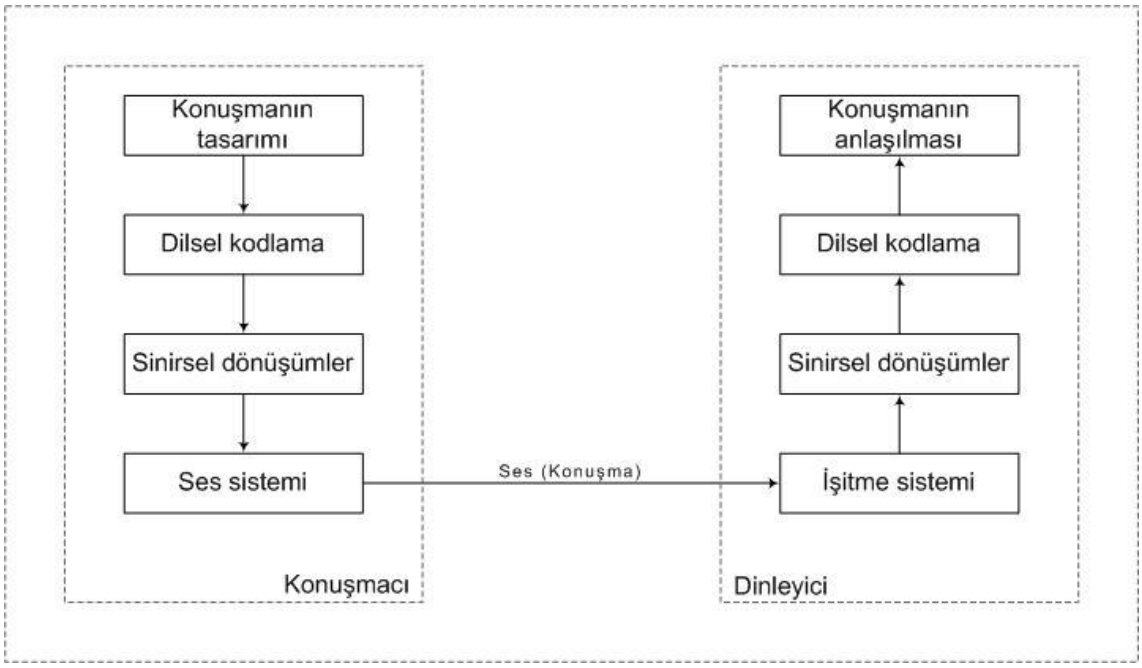
Sözlük kullanımında, sözlük dışı kelimeler ve sistem kaynakları nedeniyle sözlük kapasitesinin sınırlandırılma zorunluluğu da bir diğer temel sorun olarak görülür (Ibarra ve Curatelli 2000). Türkçe gibi türetilebilir dillerde ise tüm sözcüklerden oluşan bir sözlük hazırlamak ve bu sözlükle çalışmak neredeyse imkansızdır. Bu tür problemlerin giderilmesi için ise bazı ses tanıma sistemleri, kullanılan dil için bir kural tabanı eklentisi ile çözüm geliştirmeye çalışmaktadırlar. Alınan kelimeler, dilsel kurallara göre analiz edilmekte ve ses tanıma sisteminin başarımı arttırılmaya çalışılmaktadır.

3. SES TANIMA SİSTEMLERİNE GENEL BİR BAKIŞ

Önceki bölümde ses tanıma sistemlerine bir giriş yapılmış, sağladığı faydalar, kullanım alanları ile sınıflandırılmasına yer verilmiş ve temel sorunlarına değinilmiştir. Bu bölümde ise ilk olarak insanlar arası sesli iletişim modeli aktarılmış ve buradan yola çıkılarak ses tanıma sisteminin bir modeli verilmiştir. Sonrasında ise ses tanıma süreci ve aşamaları aktarılmıştır.

3.1. İnsanlar Arası Sesli İletişim

Sesli iletişim insanlar arası en temel iletişim şeklidir. Şekil 3.1'de insanlar arası sesli iletişimin bir modeli görülebilir.



Şekil 3.1 İnsanlar arası sesli iletişim (Huang vd 2001)

İnsanlar arası sesli iletişim beyinde, konuşma seslerini üreten sinirsel hareketleri aktive eden bir düşünce ve niyetle başlar. Dinleyici, konuşmayı beynin anlayacağı sinirsel sinyallere dönüştüren işitme sistemi yardımıyla alır (Huang vd 2001). İnsanlar arası sesli iletişim, konuşmacı ve dinleyicideki dönüşüm işlemleri süreç zinciri ile gerçekleşir.

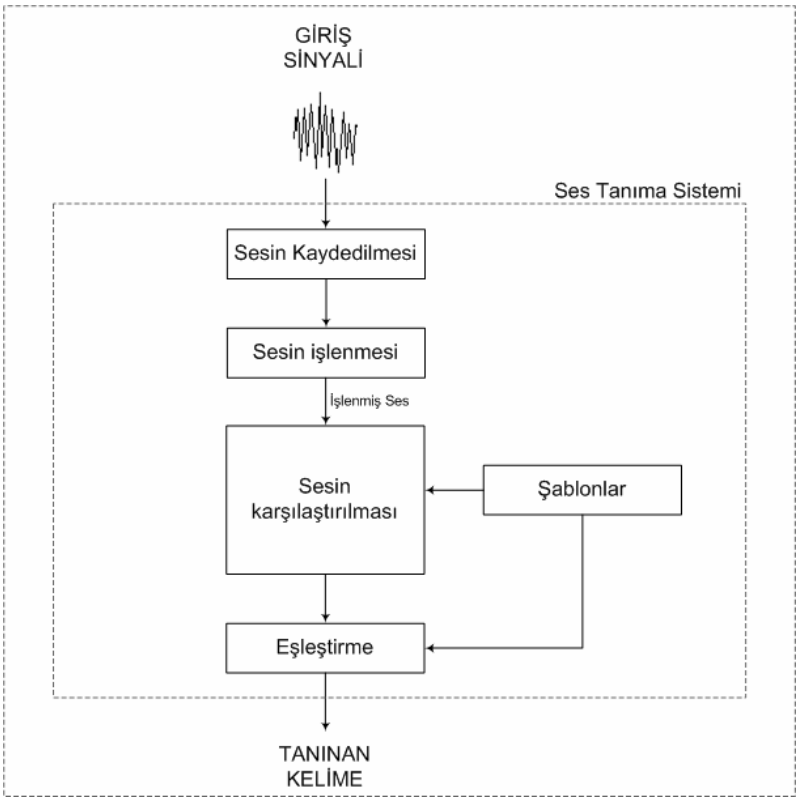
İnsanlar arası sesli iletişimdeki süreçler, konuşmacıda ve dinleyicide şu şekilde özetlenebilir:

- Konuşmacı: Konuşmanın oluşturulması süreci bir kişinin beyninde dinleyiciye konuşma yoluyla iletilecek bir mesajın düşüncesi ile başlar. Bu mesaj bir seri kelimeye dönüştürülür. Dilsel kodlama yardımıyla, bu kelime serisi, kelimelerin telaffuzuna karşılık gelen bir seri foneme dönüştürülür. Bu eşleştirme sonrasında mesajın konuşma şeklinde dinleyiciye aktarılması için ses sistemine gönderilecek sinirsel dönüşümler gerçekleştirilir. Ses sistemi bu sinirsel dönüşümler sayesinde uyarılarak konuşma gerçekleştirilir.

- Dinleyici: Konuşmanın anlaşılması süreci ise dinleyicideki, işitme sistemi ile başlar. Huang vd (2001) tarafından aktarıldığına göre; ses ilk olarak, bir filtreler bankası şeklinde frekans analizi gerçekleştiren iç kulaktaki *salyangoz*'a iletilir. Bunu izleyen sinirsel dönüşüm süreci ile spektral sinyaller, işitsel sinirlere iletilen aktivite sinyallerine dönüştürülür. Sinirsel dönüşümlerin, dilsel sistemde nasıl eşleştirildiği ve beyinde anlamanın nasıl gerçekleştiği henüz tam olarak açıklanamamıştır.

Ses tanıma sistemlerinin çalışması, insanlar arası sesli iletişim sürecinde dinleyicinin yaptığı işlevleri yapay olarak gerçekleştirme prensibine dayanmaktadır. Dinleyici ve ses tanıma sistemlerinde basit bir eşleştirme yapılırsa; sırasıyla dinleyicideki işitme sisteminin yerini, sesin kaydedilmesi alır; sinirsel dönüşümlerin yerini, alınan sesin işlenmesi; dilsel kodlamanın yerini, işlenmiş sesin dilsel karşılıkların belirlenmesi veya mevcut şablonlarla karşılaştırılması; konuşmanın anlaşılmasının yerini ise eşleştirme alır.

Şekil 3.1'deki örnek sesli iletişim modelinden yola çıkılarak, ses tanıma sistemi için örnek bir model oluşturulabilir. Sesli iletişim modelinden yola çıkılarak ve yukarıda verilen eşleştirmeden yararlanmak suretiyle oluşturulan örnek bir ses tanıma sistemi modeli, Şekil 3.2'de verilmektedir.



Şekil 3.2 Örnek bir ses tanıma sistemi modeli

3.2. Ses Tanıma Süreci

Ses tanıma, Cole vd (1995) tarafından, mikrofon veya telefonla alınmış ses sinyalinin kelimeler kümesine çevrim süreci olarak tanımlanmıştır. Tanınmış kelimelerin komut – kontrol, veri giriş ve döküman hazırlama gibi uygulamalar için sonuçlar olabileceğini ifade etmişlerdir.

Mikrofon veya telefonla alınmış ses sinyalinin kayıt işlemi ile başlayıp, ses tanıma sisteminin sonucu kelimeye karşılık gelen işlevin gerçekleştirilmesi ile sonuçlanan ses tanıma süreci aşamaları; genel olarak şu şekilde özetlenebilir:

- Sesin kaydedilmesi ve ifadenin saptanması
- Sesin işlenmesi
- Karşılaştırma ve eşleştirme
- İşlevin gerçekleştirilmesi

3.2.1. Sesin kaydedilmesi ve ifadenin saptanması

Bu aşama, sesin ses kayıt cihazı aracılığı ile kaydının yapılması ile başlar. Kayıt cihazı amaçlanan sisteme göre değişiklik gösterecektir. Genel olarak kullanılan kayıt cihazları, mikrofon ve telefondur. Telefon, daha çok, uzaktan erişimli sistemler için tercih edilmektedir.

Bu aşamada amaç, sesin kaydedilmesi; ses kaydının yapılırken, konuşma, yani sesin bulunduğu kısımların saptanmasıdır. Bu aşamada ses tanıma sistemi, kayıt cihazı yardımıyla ses kaydını gerçekleştirir, kayıtta bulunan konuşmayı başlangıç ve bitişi ile birlikte saptar, sonrasında konuşmanın bulunduğu kısmı işlenmesi için bir sonraki aşamaya verir. Konuşmanın başlangıç ve bitişinin saptanması, kısaca ifadenin saptanması olarak ifade edilebilir.

İfadenin saptanması ile ses tanıma sistemi tarafından, tüm kayıt yerine sadece konuşma geçen bölümlerin işlenmesi sağlanmaktadır. Bu sayede ses tanıma sisteminin, konuşma içermeyen ve sadece gürültü içeren ses kayıtlarını işlemekle boşa vakit kaybetmesi önlenerek performansı arttırılabilmektedir.

İfadenin saptanmasında en sık kullanılan teknikler ise *sıfırı geçiş sayısının* hesaplanması ve bir çerçevedeki enerjilerin *RMS (Root Mean Square - Karelerin aritmetik ortalamasının kökü)* hesabı olarak verilebilir.

3.2.2. Sesin işlenmesi

Kayıt aşamasında saptanmış bulunan ve ifade içeren ses sinyali bu aşamaya giriş olarak alınmaktadır. Bu giriş sinyali, ses tanıma sistemi tarafından karşılaştırma ve eşleştirmeye hazır hale getirilmesi için bir seri işlemden geçirilir. Ses sinyaline uygulanan bu hazırlık işlemleri, ses tanıma sisteminin tasarımına ve uygulamaya göre değişkenlik göstermektedir.

Bu aşamada genel olarak yapılan işlemler ise; sesin pencereleme fonksiyonundan geçirilmesi, normalizasyonların yapılması, sesin filtrelenmesi, yani ses sinyalinden gürültünün çıkarılması, sesin frekans analizlerinin gerçekleştirilmesi, sesin kodlanması, ses sinyalinin zamana göre yayılması olarak sayılabilir.

Ses tanıma sürecinde sesin işlenmesi, en önemli aşamalardan biri olarak karşımıza çıkar. Bu aşamada amaç kaydedilen sesin özelliklerini bozmadan, ses hakkında bilgi sahibi olmaktır.

Bu aşamada kullanılan tekniklerden bazıları şu şekilde özetlenebilir:

- Pencereleme için genel olarak kullanılan teknikler, *dikdörtgensel pencereleme* ve *Hamming pencereleme* fonksiyonlarıdır.

- Filtreleme için en sık olarak *doğrusal filtreler* kullanılmaktadır. Doğrusal filtrelerden ise geliştirilmesinin basit olması ve çalıştırılması sırasında çok az sistem kaynağı gerektirmesi nedeniyle *FIR filtreler* ve *IIR filtreler* sıklıkla kullanılmaktadır.

- Sesin frekans analizlerinin gerçekleştirilmesi ve kodlanmasında ise kullanılan çok fazla teknik bulunmaktadır. Bunlardan bir kısmı; PCM, APCM, DPCM, DM, ADPCM, spektrum analizleri, Filtreler bankası, LPC, PLP, CELP, VSELP, RASTA-PLP olarak sayılabilir. Bu tekniklerden bazıları bir sonraki bölümde daha detaylı olarak aktarılacaktır.

3.2.3. Karşılaştırma ve eşleştirme

Bu aşamada yapılan işlem, işlenmiş sesin, bilinen örneklerle karşılaştırılması ve eşleşenlerin saptanması ile kaydı gerçekleşen sesin tanınmasıdır. Bu aşama için kullanılan çok fazla teknik bulunmaktadır. Bunlardan bazıları: Time Warping – Dynamic Time Warping (DTW), Hidden Markov Model (HMM), Frekans Analizi, Lineer Cebir Teknikleridir. Tüm bu teknikler, bir olasılık ya da kesinlik eşleşmesi oluşturmak için kullanılmaktadır.

Bu aşamada kullanılmakta olan bir diğer teknik ise Yapay Sinir Ağlarıdır. Bu modellere ileriki bölümlerde değinilecektir. Ayrıca, HMM ve Yapay Sinir Ağlarının beraber kullanıldığı bazı sistemlerde bulunmaktadır ve bunlar ‘Hibrit Modeller’ olarak anılmaktadırlar.

3.2.4. İşlevin gerçekleştirilmesi

Ses tanıma sürecinde, en son aşama işlevin gerçekleştirilmesidir. Bu aşamada, ses tanıma sisteminde giriş olarak alınan ses sinyalinden eşleştirilen kelimeye karşılık gelen işlev gerçekleştirilmektedir. Gerçekleştirilen işlev, ses tanıma sistemi tasarımına göre değişkenlik gösterecektir. Dikte sistemleri için tanınan kelimenin, metin düzenleyicisine yazdırılması, komut - kontrol sistemi için ise tanınan kelimeye karşılık gelen komutun işlenmesi bu aşamaya karşılık gelmektedir.

4. SES TANIMA SÜRECİNDE KULLANILAN TEKNİKLER

Bir önceki bölümde insanlar arasındaki sesli iletişim modelinden yola çıkılarak ses tanıma sistemi modeline, çalışma prensibine ve ses tanıma süreci aşamalarına yer verilmiştir. Bu bölümde ise; ses tanıma sürecinin her bir aşaması için kullanılan tekniklere yer verilerek bu teknikler açıklanmıştır.

4.1. Sesin Kaydedilmesi ve İfadenin Saptanması

Bu aşamada ilk olarak sesin kaydedilmesi gerçekleştirilmektedir. Ses kaydedildikten sonra çeşitli süreçlerden geçecek ve işlenecektir. Bu işlemlerin yapılabilmesi için ise sesin sayısallaştırılması gerçekleştirilmektedir.

4.1.1. Sesin sayısallaştırılması

Analog sinyal, sayısallaştırılması için önce filtreden geçirilmekte, ardından örnekleme yapılmaktadır. Şekil 4.1'de örnek bir sayısallaştırma işlevi gösterilmektedir. Burada *x(t)* analog sinyali ifade ederken, *x(nT)* sayısallaştırılmış sinyali ifade etmektedir. Sesin sayısallaştırılması aşamasında, Şekil 4.1'de görülen filtre, analog filtreyi ifade etmektedir. Analog filtreleme ve örnekleme sesin kaydedilmesi aşamasında gerçekleştirilmektedir.

Şekil 4.1 Sesin sayısallaştırılması (Robinson 1998)

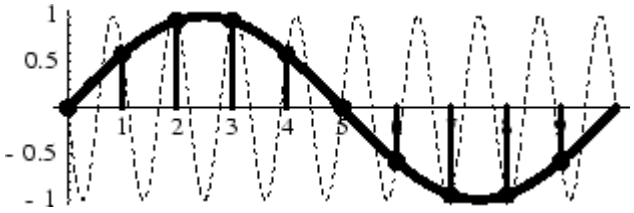
4.1.1.1. Örnekleme

Sayısal sinyal işleme tekniklerinin kullanılabilmesi için analog sinyalin, yani konuşmanın, bir seri sayılar şeklinde gösterimi gerekmektedir (Rabiner ve Schafer 1978). Bu, analog sinyalin örneklenmesi ile yapılmaktadır.

Sesin özelliklerinin korunması için, örnekleme, dönüşümü yapılacak ses sinyalinin içerdiği en yüksek sıklıktaki frekansın en az iki katı sıklıkta gerçekleştirilir. Bu, literatürde örnekleme teoremi olarak geçmektedir. Rabiner ve Schafer tarafından (1978) ise şu şekilde aktarılmıştır: eğer bir sinyal $x_u(t)$; $\Omega >= 2\pi F_N$ olmak üzere $X_u(j\Omega) = 0$ olacak şekilde bant sınırlı bir Fourier dönüşümüne $X_u(j\Omega)$ sahipse; $x_u(t)$, $1/T > 2F_N$ olacak şekilde ve $-\infty < n < \infty$ olmak üzere $X_u(nT)$ şeklinde eşit olarak yerleştirilmiş örneklerle yeniden inşa edilebilir.

Burada F_N, Nyquist Frekansı olarak ifade edilir. Analog ses sinyalini tekrar oluşturabilecek şekilde; en uygun örnekleme frekansı F_N'den en az iki katı olacak şekilde seçilmesi gerektiği elde edilir (Coleman 2005).

Yüksek frekanslı bir sinyal, çok düşük oranla örnekleme yapılmış olsun. Düşük oranla alınan bu örneklerden dalga yaklaşımı yapılırsa, elde edilen sinyal, ilk örnekleme yapılan sinyale göre daha düşük frekanslı bir sinyal gibi görünecektir. Yani bu şekilde örnekleme sonucu, sinyalin özelliği korunamamış olacaktır. Aşağıda, yüksek frekanslı bir sinyalin, çok düşük sıklıkta örneklenmesi sonucu, Şekil 4.2'de verilmiştir (Smith 2003).



Şekil 4.2 Yüksek frekanslı bir sinyalin, düşük oranla örneklenmesi (Smith 2003)

4.1.2. İfadenin saptanması

Sesi oluşturan ses dalgası, iki önemli özellik içermektedir. Bu özellikler genlik ve frekanstır (Huang vd 2001). Frekans, sesin tizlik ve peslik özelliklerini belirlerken; genlik, sesin şiddetini ve taşıdığı enerjiyi belirlemektedir. Ses tanıma sistemleri, ses sinyallerinin analizi ve ayrıştırılmasından faydalanmaktadır. Sesteki frekans ve genlik değerleri, fonem bazında farklılık göstermektedir. İşte ses tanıma sistemleri bu farklılıklardan faydalanmak suretiyle, fonemlerin veya bir seri fonemden oluşan kelimenin saptanmasına ve sınıflandırılmasına çalışır.

Sesin kaydedilmesi ve ifadenin saptanması aşamasında kullanılan genel teknik, konuşmanın geçtiği kısımların sesin genliği ile saptanmasıdır. Belli bir çerçevedeki toplam genlik hesabından faydalanılarak konuşmanın geçtiği kısımların saptanması gerçekleştirilebilmektedir. *Toplam Genlik (TG)* hesabı için eşitlik 4.1 verilmiştir.

$$TG = \sum_{t=1}^{n} x(t) \qquad (4.1)$$

Bu eşitlikte $x(t)$; t anındaki genlik; yani ses dalgasının t anındaki taşıdığı enerjiyi ifade etmektedir.

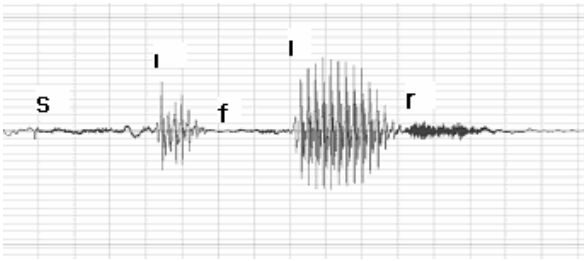
Bu yöntem ile hesaplanan toplam genlik değeri, yani enerji toplamı belirli bir değerin üstündeyse bu çerçevede sesin, yani konuşmanın başladığı anlamı çıkarılır. Konuşmanın başlangıç ve bitişinin saptanabilmesi için kullanılan bir diğer yöntem yine genlikten faydalanılarak elde edilmektedir. Bu yöntem genliklerin *RMS* (Root Mean Square) değerinden faydalanmaktadır. Bir çerçevedeki genliklerin karelerinin aritmetik ortalamasının karekökü; genliklerin RMS değeri olarak ifade edilmektedir. RMS hesabı için eşitlik 4.2 verilmiştir.

$$RMS = \sqrt{\frac{\sum_{t=1}^{n} x^2(t)}{n}} \qquad (4.2)$$

Aynı zamanda bir çerçevedeki toplam sıfırı geçiş sayısının hesaplanması da konuşma başlangıç ve bitişinin saptanmasına yardımcı olan bir diğer tekniktir.

Bu tez çalışmasında sesin kaydedilmesi ve ifadenin saptanması aşamasında *RMS hesabı* ve *toplam sıfırı geçiş sayısı* beraberce kullanılmıştır. Sesin kaydı aşamasında, kaydedilen ses kısımları sistemden alındıkça, kayıt süresince kuyruğa atılmaktadır. Oluşturulan ses tanıma sistemindeki kuyruk analizcisi yardımıyla kuyruktan bu ses parçacıkları alınmakta ve çerçeve bazında sıfırı geçiş miktarları ve RMS değerleri analiz edilmektedir. Analiz sonucunda, içinde konuşma kısmı içerdiği saptanan çerçeveler, birbiri ardı sıra eklenerek sesli ifadeler oluşturulmakta ve ifade kuyruğuna atılmaktadır.

Şekil 4.3'te, geliştirilen uygulama kullanılarak 8000 Hz'de örneklenerek kaydedilmiş 'sıfır' kelimesi için, RMS ve sıfırı geçiş sayısı ile başlangıç ve bitişi saptanarak belirlenmiş ses sinyali görülmektedir.



Şekil 4.3 Başlangıç ve bitişi ile belirlenmiş ‘sıfır’ kelimesi ses sinyali

4.2. Sesin İşlenmesi

Sesin işlenmesi aşamasında genel olarak yapılan işlemler pencereleme, sayısal filtreler ile filtreleme, normalizasyon ve sesin kodlanması, yani özelliklerinin çıkarılmasıdır. Bu aşamada özellikle sesin kodlanması aşamasında kullanılan pek çok teknik bulunmaktadır.

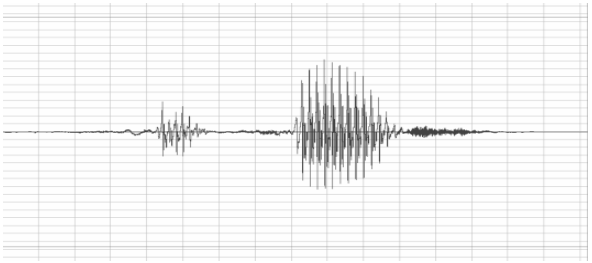
4.2.1. Pencereleme

Sesin işlenmesi aşamasında, ses sinyalinin pencereleme fonksiyonundan geçirilmesi sıkça kullanılan bir yöntemdir. Sıklıkla kullanılan iki pencereleme fonksiyonu, Hamming pencereleme ve dikdörtgensel pencereleme (Rectangular Window) fonksiyonlarıdır.

Kullanılan diğer pencereleme fonksiyonları şu şekilde verilebilir:

- Barlett Window
- Hanning Window
- Blackman Window
- Kaiser Window

Geliştirilen uygulamada Hamming penceresi kullanılmıştır. Şekil 4.4’te, Şekil 4.3’te verilmiş olan ‘sıfır’ ses sinyalinin Hamming pencereleme fonksiyonundan çıkışı görülmektedir.



Şekil 4.4 Hamming penceresinden geçirilmiş ‘sıfır’ kelimesi için ses sinyali

Hamming penceresi için fonksiyon Eşitlik 4.3’te verilmektedir. Hamming penceresi yardımıyla ses sinyalinin merkezi belirginleştirilmektedir.

$$w(n) = 0.54 - 0.46\cos(\frac{2\pi n}{N-1}) \qquad (4.3)$$

Hamming pencereleme fonksiyonu çıkışı *y(n)*, eşitlik 4.4 ile verilmiştir.

$$y(n) = w(n) \cdot x(n) \qquad (4.4)$$

Hamming penceresi de bir nevi filtre etkisi yaratmaktadır. Şekil 4.3 ve Şekil 4.4 beraberce incelendiğinde, Hamming pencereleme fonksiyonu çıkışında sinyalin daha düzgün olduğu, kelime başlangıç ve bitişinin daha belirgin hale geldiği açıkça görülmektedir.

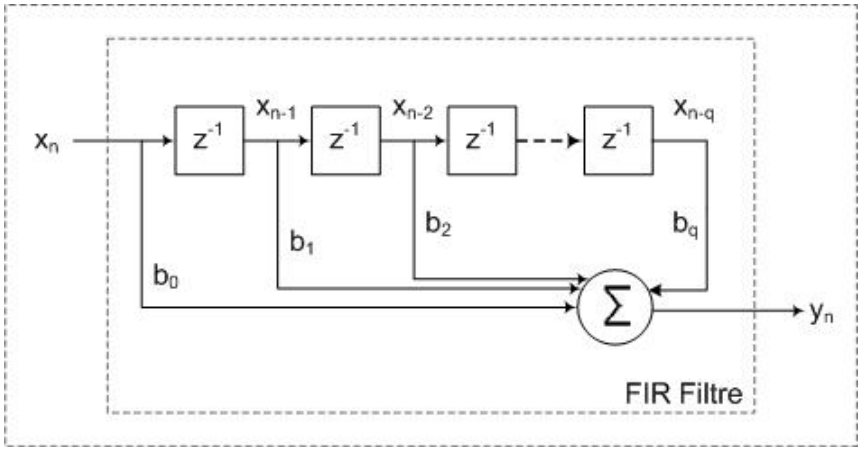
4.2.2. Doğrusal filtreler

Sesin işlenmesi aşamasında filtrelerin iki kullanım amacı vardır. Bunlar ses sinyalinin ayrılması ve ses sinyalinin düzeltilmesidir. Smith (1999), sinyal ayırmanın, sinyalin; bozulma, gürültü veya diğer sinyallerle kirlenmesinde gerekli olduğunu; sinyal düzeltilmesinin ise sinyalin herhangi bir biçimde bozulmasında kullanıldığını söylemiştir. Sayısal filtrelerden FIR (Finite Impulse Response) Filtre ve IIR (Infinite Impulse Response) Filtre; geliştirilmesi basit olması ve çalışması esnasında çok düşük sistem gereksinimi ile karşılanabilmesi nedenleriyle sıklıkla kullanılmakta olan iki doğrusal filtredir.

4.2.2.1. FIR filtreler

Giriş sinyali, *x(n)*’e karşılık, o anki ve önceki girişlerin ağırlıklı toplamı olan *y(n)* çıkışını oluşturur.

Şekil 4.5'te örnek bir FIR Filtre modeli gösterilmiştir. Bu filtrenin matematiksel gösterimi ise 4.5 nolu eşitlikle verilmektedir.

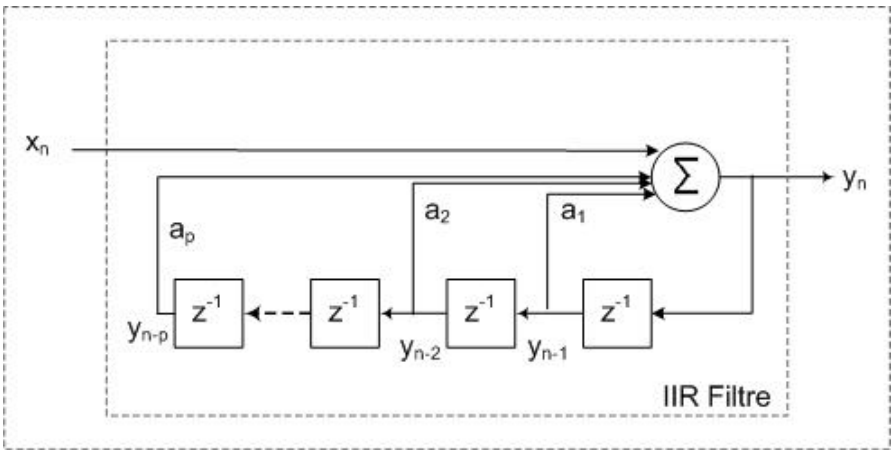


Şekil 4.5 Örnek bir FIR filtre modeli

$$y_n = b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} + \ldots + b_q x_{n-q} \qquad (4.5)$$

4.2.2.2. IIR filtreler

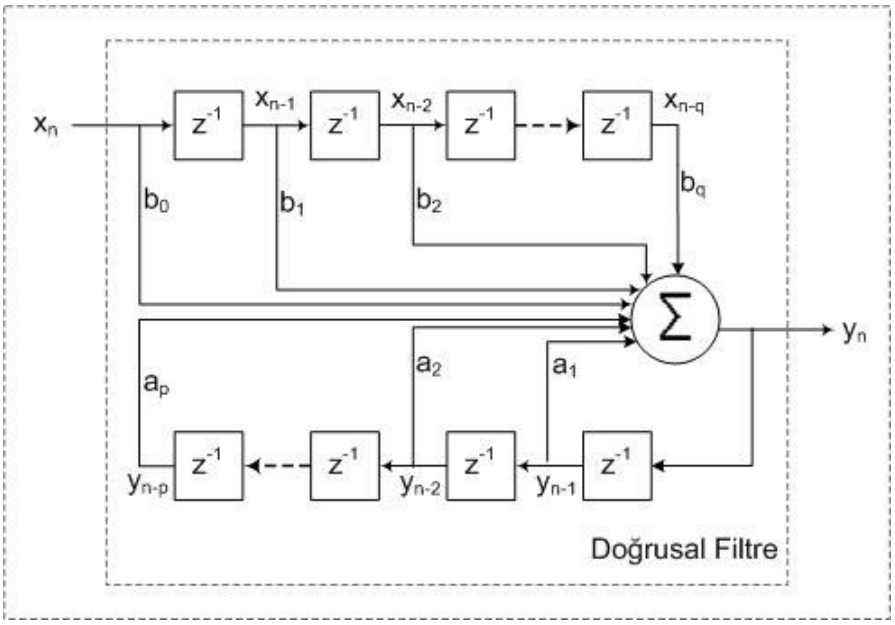
Giriş sinyali, *x(n)*'e karşılık, o anki ve önceki girişlerin ağırlıklı toplamları ile birlikte, önceki çıkışların da ağırlıklı toplamını ifade eden *y(n)* çıkışını oluşturur. Şekil 4.6'da örnek bir IIR modeli verilmiştir.



Şekil 4.6 Örnek bir IIR filtre modeli

Bu modelde *x(n)* girişi ile birlikte, önceki *p* adet çıkışın ağırlıklı toplamı filtre çıkışı *y(n)*'i vermektedir. Önceki girişlerin bu örneğe özel olarak ağırlıklı toplama dahil edilmediği görülmektedir.

Şekil 4.7'de genelleştirilmiş bir doğrusal filtre modeli verilmiştir. Bu modelde önceki çıkışların, ağırlıklı toplama girmediği durumlarda FIR filtre, önceki çıkışların da ağırlıklı toplama dahil edildiği durumda ise IIR Filtre elde edilir.



Şekil 4.7 Genelleştirilmiş doğrusal filtre

Genelleştirilmiş doğrusal filtre modeli eşitlik 4.6 ile ifade edilir.

$$y_n = \sum_{i=1}^{p}(a_i \cdot y_{n-i}) + \sum_{j=0}^{q}(b_j \cdot x_{n-j}) \qquad (4.6)$$

Bu eşitlikte $p = 0$ olduğu durumlarda FIR filtre, $p \neq 0$ olduğu durumlarda ise IIR filtre sistemi elde edilebilmektedir.

4.2.3. Sesin kodlanması

Sesin sayısallaştırılmasından sonra sesin kodlanması gerçekleştirilir. Sesin kodlanması genel olarak iki sınıfta incelenmektedir. İlki; sesin tekrar oluşturulması amacını taşıyan dalga biçimi kodlaması (waveform coding), ikincisi ise kodlanmış sinyaldeki konuşma özelliklerini korumayı amaçlayan ses kodlayıcıları (vocoder) olarak ifade edilirler.

PCM, APCM, DPCM, DM ve ADPCM, dalga biçiminin bir yaklaşımını vermeye çalışan tekniklerdir. Yeterli oranlarda kimi zaman dalga biçimini aynen elde etmeye çalışırlar (Huang vd 2001).

4.2.3.1. PCM

PCM, doğrusal kuantizasyon (niceleme - nicelik çıkarma) tekniklerinin en basit olanıdır. B bit ile 2^B kuantizasyon seviyesi elde edilmektedir.

Bilgisayarlarda kayıtlı Windows® WAV, Apple® AIF, Sun® AU ve SND formatındaki dijital sesler, temel olarak 16-bit doğrusal PCM kullanmaktadır. Yoğun diskler de (CD) 16 bit doğrusal PCM kullanmaktadırlar (Huang vd 2001). 64 kbps'de telefon konuşmaları için gereken anlaşılırlıkta kodlama sağlayabilmektedir.

4.2.3.2. ADPCM

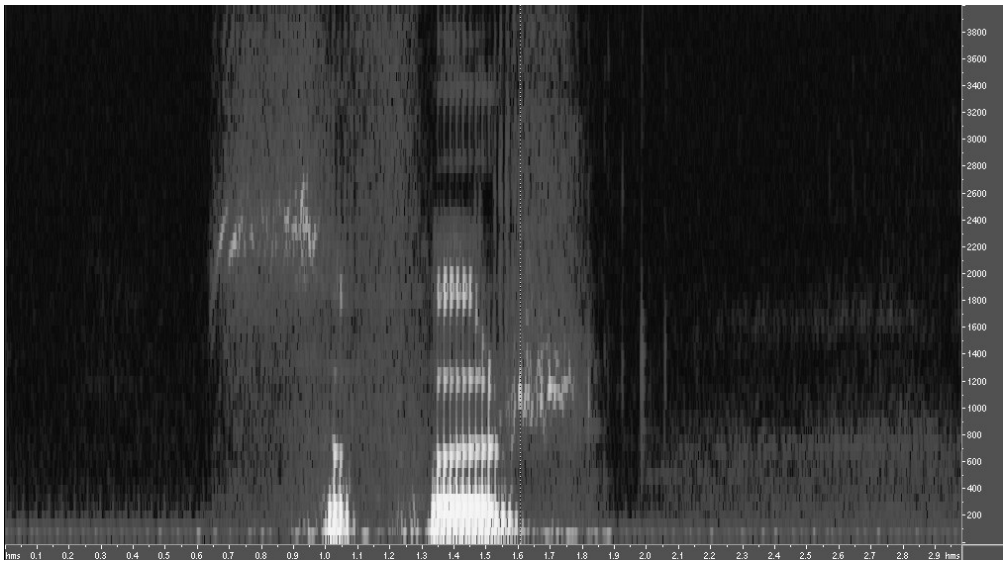
32 kbps'de yüksek kalitede konuşma sağlamaktadır. PCM gibi doğrudan ses sinyalini kuantize etmek yerine, ses sinyali ile ses sinyalinden yapılan önkestirimin farkını kuantize eder. ADPCM genellikle, özel ağlarda ve uluslararası çevrimlerde ses kanallarının iki kata genişletilmesi ile genel kullanıma sahiptir.

16 kbps ve daha az sıklıklarda, yüksek ses kalitesi, daha karmaşık uyarlamalı önkestirim (adaptive prediction) ile elde edilebilir.

4.2.3.3. Filtreler bankası

Sesin, analiz için bir seri filtreden geçirilmesidir. Bu teknikte her frekans bir filtre tarafından işlendiğinden bilgi kaybı yoktur. Bu teknik ile, sesin çeşitli frekanslardaki özellik çıkarımları yapılıp, değerlendirme yapılmasına imkan tanınmaktadır. Filtreler bankası verilmeden önce spektrum ve Fast Fourier Transform – FFT tanımlarını vermek yerinde olacaktır.

Spektrum, ses sinyalindeki çerçeveler arasındaki kısa süreli değişimlere duyarlı olmaktadır. Hesaplanması, sesin belirli aralıklarla frekans yoğunluğunun alınması ile yapılmaktadır. Şekil 4.8'de 8000 Hz ile örneklenmiş 'sıfır' kelimesi ses sinyalinin spektrumu gösterilmektedir.



Şekil 4.8 8000 Hz ile örneklenmiş 'sıfır' kelimesi ses sinyalinin spektrumu

Ses tanıma sistemlerinde sıklıkla kullanılan bir yöntem, sesli ifadenin spektrum içeriğinden faydalanmaktır. Spektrum hesaplamak için kullanılmakta olan en etkin yöntemlerden biri Fast Fourier Transform – FFT olarak bilinmektedir. FFT, Discrete Fourier Transform – DFT olarak adlandırılan dönüşüm işleminin daha etkin yapılabilmesi için geliştirilmiş bir biçimdir.

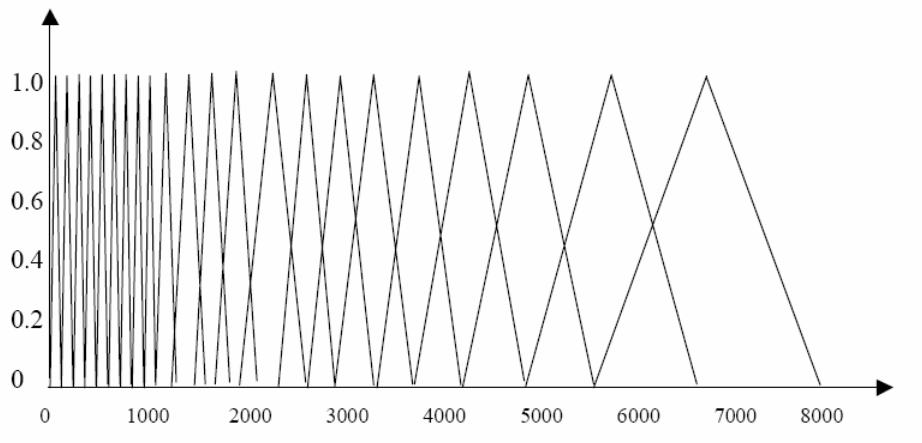
DFT için dönüşüm işlemi, f frekans, f_s örnek alınan frekans N_s ise pencere genişliği olmak üzere eşitlik 4.7 ile verilmektedir.

$$S(f) = \sum_{n=0}^{N_S-1} s(n) \cdot e^{-j(2\pi f / f_s)n} \qquad (4.7)$$

Spektrum analizi için *fourier* dönüşümü dışında bir dizi filtre bankasının kullanılması da mümkündür. Filtreler bankası bir seri bant geçişli filtreden oluşmaktadır. Bu şekilde filtreleme yapılması sonucu her çerçevenin o anki spektrumu alınmış olur. Bu filtreler serisinde bant genişliklerinin eşit alınma zorunluluğu bulunmamaktadır.

Filtreler bankası olarak sıklıkla kullanılanlardan ilki, *Mel-scale* filtre bankası olarak bilinmektedir. Bu filtre bankası üçgen şeklinde bir seri filtreden oluşmaktadır. Filtrelerin özellikleri ise düşük frekanslar için doğrusal iken, daha yüksek frekanslar için logaritmik artan şeklindedir (Rabiner ve Juang 1993).

Mel-scale filtre bankası Şekil 4.9'da verilmektedir.



Şekil 4.9 Mel-scale filtre bankası

Mel-scale filtreler bankası için kullanılan mel frekansı eşitlik 4.8 ile ifade edilmektedir.

$$f_{Mel} = 2595 \log 10(1 + f/700.0) \qquad (4.8)$$

Sık kullanılan bir diğer filtreler bankası ise *Bark scale* filtreler bankasıdır. Bark-scale frekansı eşitlik 4.9 ile verilmektedir.

$$f_{Bark} = 13 \cdot \arctan\left(\frac{0.76 \cdot f}{1000}\right) + 3.5 \cdot \arctan\left(\frac{f^2}{(7500)^2}\right) \qquad (4.9)$$

Mel-scale ve bark-scale filtre bankalarını oluşturabilmek için kritik bant genişliklerinin belirlenmesi gerekmektedir. Kritik bant genişliklerinin belirlenmesi işlemi eşitlik 4.10 ile yapılmaktadır.

$$BG_{kritik} = 25 + 75 \cdot \left[1 + 1.4 \cdot (f/1000)^2\right]^{0.69} \qquad (4.10)$$

4.2.3.4. LPC

Bu yöntem, insan gırtlağı ve ağız yapısı özelliklerinin yanı sıra, ses özelliklerini de dikkate alır. Doğrusal önkestirim temel olarak, sesin, periyodik dürtü veya rasgele gürültü ile uyarılan, doğrusal ve zamana göre değişen bir sistemin çıktısı ile modellenebileceği prensibine dayanır. Bu model doğrusal bir filtre olarak Eşitlik

4.11'deki transfer fonksiyonu ile ifade edilmektedir. Burada *p*, LPC kodlayıcının seviyesi olarak ifade edilir (Huang vd 2001).

$$H[z] = \frac{X[z]}{E[z]} = \frac{1}{1 - \sum_{i=1}^{p} a_i \cdot z^{-i}} \qquad (4.11)$$

4.11 nolu eşitlikte, ters z-dönüşümü uygulandığında, 4.12 nolu eşitlik elde edilmektedir.

$$x[n] = \sum_{i=1}^{p} a_i \cdot x[n-i] + e[n] \qquad (4.12)$$

LPC, sıradaki örneğin, önceki bir seri örnekten yaklaşık olarak elde edilebileceği prensibiyle çalışır (Eşitlik 4.13).

$$\hat{x}[n] = \sum_{i=1}^{p} a_i \cdot x[n-i] \qquad (4.13)$$

Tahmin sonucu elde edilen örneğin asıl örnekle olan farkının; yani hatanın kareleri toplamının minimizasyonu için bir seri parametre hesaplanmaktadır (Eşitlik 4.14).

$$e[n] = x[n] - \hat{x}[n] = x[n] - \sum_{i=1}^{p} a_i \cdot x[n-i] \qquad (4.14)$$

Eşitlik 4.14'ün çözümü ile *p* sayıda LPC parametresi hesaplanmaktadır. Burada *p*, LPC kodlayıcı seviyesi; $a_1, a_2, \ldots, a_p$ ise LPC Parametreleri olarak ifade edilir.

Tahmin sonucu elde edilen örneğin asıl örnekle olan farkının; yani hatanın en küçük yapılabilmesi için bir seri parametre güncellenmektedir. Bu, eşitlik 4.14'ün minimize edilmesi yani, optimizasyonu problemidir. Bu eşitliğin optimizasyonu ile hesaplanan *p* sayıda parametre, LPC kodlayıcıya olan bir çerçeve örnek girişine karşılık çıkışı verip, kodlama sonucuna karşılık gelmektedir.

4.2.3.5. PLP

LPC'nin bir varyasyonudur ve ilk olarak 1990'da Hermansky tarafından ortaya atılmıştır. Bu teknikteki temel fikir, insan kulağının işitilebilir aralıkla ilgili, fiziki özelliklerinden türetilen bazı karakteristikleri dikkate almasıdır. PLP'de de; LPC'de olduğu gibi bir seri parametre hesaplanmaktadır. PLP parametreleri, DFT ve LP (doğrusal önkestirim) tekniklerinin birleştirilmesi ile hesaplanmaktadır (Hermansky 1990).

4.2.4. Karşılaştırma ve eşleştirme

Modern ses tanıma sistemlerinde genel olarak kullanılan mimari, ses sinyalinden, olması muhtemel kelimeler serisi oluşturan yazılım mimarisi şeklindedir. Bu mimarideki en popüler algoritmalar istatistiksel metotlara dayanır.

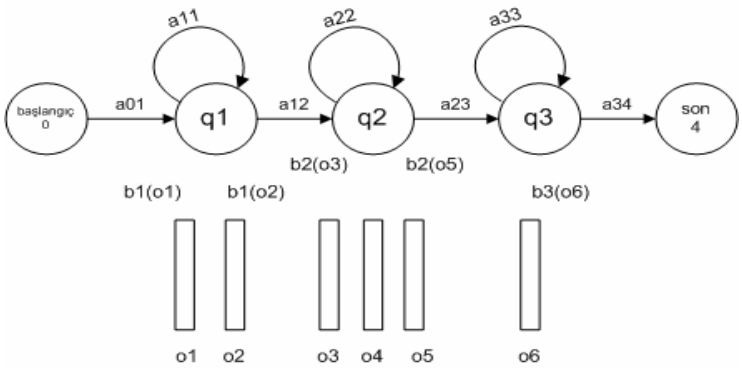
Karşılaştırma ve eşleştirme aşamasında en çok kullanılan teknikler; HMM (Hidden Markov Model), DTW (Dynamic Time Warping) ve Yapay Sinir Ağlarıdır.

4.2.4.1. Hidden Markov Model

HMM, ayrık zamanlı serilerin, gözlenmiş veri örneklerinin sınıflandırılması için güçlü bir istatistiksel yöntemdir. Ses tanıma sistemlerinde çok sıklıkla kullanılır. Bu kadar popüler olmasının nedeni ise, zengin bir matematiksel yapıya sahip olması ve uygun olarak uygulandığında başarılı sonuçlar elde edilmesini sağlamasındandır.

Ses tanıma sistemlerinde, HMM'deki gözlem serileri, ses sinyalini simgeleyen özellik vektörleridir. Modelde görülen durumlardan kendisine geçişler de, fonem tabanlı bir ses tanıma sistemi için, bir fonemin uzun seslendirilmesinde aynı durumun tekrarlarını ifade etmek içindir. Kısa bir deyişle, ses tanıma sistemlerinde HMM'deki durumlar fonemlere karşılık gelir.

Ses tanıma sistemlerinde kullanılan, HMM'in amacı da bilinmeyen (saklı) durum dizisini gözlemlere dayanarak bulmaktır. Şekil 4.10'da örnek bir HMM gösterimi verilmektedir.



Şekil 4.10 Örnek bir HMM gösterimi

Şekil 4.10'da örnek bir gösterimi verilen HMM'i tanımlayan parametreler, Jurafsky ve Martin (2000) tarafından şu şekilde aktarılmıştır:

- Durumlar: durumların bir kümesidir. $Q = q_1q_2...q_N$
- Geçiş olasılıkları: Olasılıkların bir kümesidir. $A = a_{01}a_{02}...a_{N1}...a_{NN}$ Burada her a_{ij}, durum i'den durum j'ye geçiş olasılığını gösterir. Bunların tümü, geçiş olasılıklarının matrisini oluşturur.
- Gözlem olabilirlikleri: gözlem olabilirliklerinin bir kümesidir. $B=b_i(o_t)$ ile gösterilir. Her biri, durum i'den bu o_t gözleminin oluşturulabilme olasılığını ifade eder.
- İlk dağılma: π_i , HMM'in i durumundan başlama olasılığı olarak ifade edilir. Tabii ki, bazı durumlar doğal olarak 0 olabilecektir. Bu da bu durumun ilk dağılma olamayacağı anlamına gelir.
- Kabul durumları: Kabul edilebilir durumların bir kümesidir.

Gözlem serisi, $O = (o_1o_2o_3...o_T)$, olarak ifade edilir.

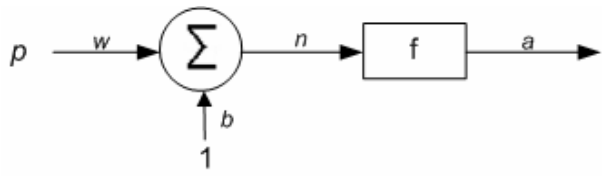
HMM, geliştirilirken üç alt probleme ayrılabilir.

- Hesaplama problemi: Verilen bir model ve gözlemler serisi için, modelin bu gözlemleri oluşturma olasılığının hesabıdır.
- Çözümleme problemi: Verilen bir model ve gözlemler serisi için, gözlemleri oluşturan, en olası durum serisinin hesaplanmasıdır.

- Öğrenme Problemi: Verilen bir model ve gözlemler serisi için, bu gözlemleri oluşturabilme olasılığını en yüksek yapmak için modelin parametrelerinin ayarlanmasıdır.

4.2.4.2. Yapay sinir ağları

Yapay sinir ağlarının kullanım alanı oldukça geniştir. Kullanım prensibi ise bilinmeyen bir fonksiyonun modellemesini yapmaya dayanır. Bunu yapabilmek için ilk olarak yapay sinir ağı modeli oluşturulmaktadır. Bilinmeyen fonksiyona ait gözlenen giriş ve çıkışlar, veri serisi olarak yapay sinir ağında eğitim amacıyla kullanılır. Yapay sinir ağının eğitimi tamamlandıktan sonra amaç, bu bilinmeyen fonksiyonun çıkışlarını üretebilme yeteneğine kavuşturulmasıdır. Yapay sinir ağına temel olarak insan beyninin yapısı, nöronlar ve aralarındaki bağlar alınmıştır. Yapay sinir ağlarına, insan beyninin basit bir modelidir denebilir. Şekil 4.11'de tek girişli bir nöron modeli verilmiştir (Hagan 1995).



Şekil 4.11 Tek girişli nöron modeli

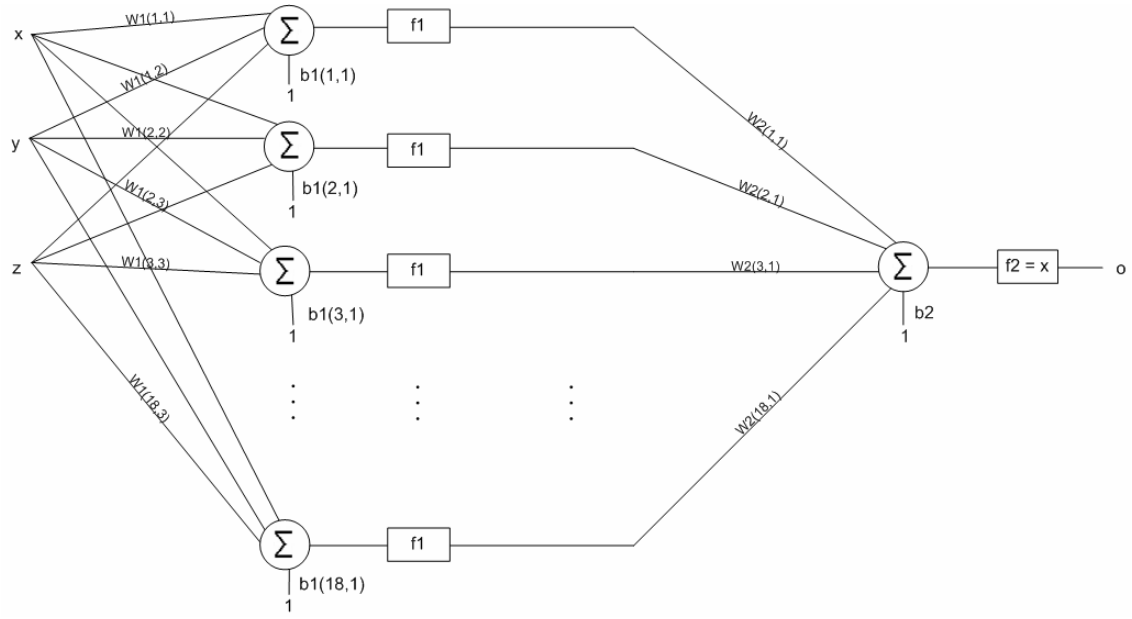
Bu nöron modelinde: *p* bu nörona olan giriş, *a* ise bu nörondan olan çıkışı ifade etmektedir. Burada *w* ağırlığı (*weight*) ve *b* ise öngerilimi (*bias*) göstermektedir. Nöron çıkışı *a*, *p* girişinin bu nörona olan ağırlığı ile çarpımının yine bu nörondaki öngerilim ile toplamından çıkan sonucun (yani *n*), *f* fonksiyonuna girişi karşılığı elde edilen çıkıştır.

Şekil 4.11'de verilen tek girişli nöron modelinin çıkış fonksiyonu 4.7 nolu eşitlikle verilmektedir.

$$a = (f.w + b) \qquad (4.15)$$

Bu şekildeki birçok nöronun, birbirine bağlı olması Yapay Sinir Ağını oluşturur. Her bir nöron, başka nöronlardan gelen çıkış verilerinin ve bu nörondaki öngerilimin

ağırlıklı toplamını bir fonksiyondan geçirerek diğer nöronlara iletir. Şekil 4.12'de 3 girişli, 1 çıkışlı ve 2 katmanlı bir yapay sinir ağı modeli görülmektedir.



Şekil 4.12 Örnek bir yapay sinir ağı modeli (3 girişli, 1 çıkışlı ve 2 katmanlı)

Yapay sinir ağı, Şekil 4.12'deki örnek modelde görüldüğü gibi, üzerindeki katsayılar ve fonksiyonlar belirli olduğunda bir fonksiyonu temsil etmektedir. Belli girişler için, belli çıkışları üretir. Buradaki amaç, bir fonksiyonu modellemek olduğuna göre bazı parametrelerin baştan belli olmaması doğaldır. Yapay Sinir Ağının eğitimi ile kastedilen de bu parametrelerin belirlenmesi aşamasıdır. Bunun için girişlere karşılık gözlenen çıkışların bir serisine ihtiyaç vardır. Amaç bilinmeyen fonksiyona, yapay sinir ağının yaklaştırılması, hatta aynı girişlere karşılık, aynı çıkışları üretebilmesini sağlamaktır.

Bilinmeyen fonksiyonumuza $f(x)$, Yapay sinir ağının çıkışına da $f'(x)$ denirse, eşitlik 4.16 ile verilen e, hata olarak ifade edilir.

$$e = f'(x) - f(x) \qquad (4.16)$$

Amaç, hataların minimize edilmesidir. Bunun yöntemi ise, tüm eğitim verileri için oluşan hataların karelerinin toplamını minimize etmektir. Bu bir optimizasyon problemidir. Bu konuda birçok optimizasyon yöntemi geliştirilmiş ve uygulanmaktadır. Sonuç olarak yapay sinir ağının eğitimindeki amaç, ağırlıkların ve katsayıların bu optimizasyon problemi çözümüyle belirlenmesidir.

Bunun yapılabilmesi için, ağırlıklar ve öngerilimler, ilk aşamada belli aralıktaki rasgele değerler olarak atanır. Adım adım, optimizasyon problemi çözümüyle, ağırlıklar ve öngerilimler yeni değerlerle güncellenir. Sonuç olarak bu ağırlıklar ve öngerilimlerin değerleri belirlenir ve fonksiyon modellenmiş olur.

Yapay sinir ağları, giriş ve çıkışlar arası bir ilişki oluşturabilmek üzere eğitilebilmesinden faydalanılarak örüntü tanıma ya da sınıflandırma amacıyla sıklıkla kullanılmaktadırlar. Ses tanıma sistemlerindeki amaç ta, ses sinyallerine karşılık gelen fonem ve buradan kelimelerin belirlenmesi olduğundan, ses sinyalinin işlenmesi sonucu, ses sinyaline karşılık çıkarılan özellik vektörlerinin yapay sinir ağları yardımıyla sınıflandırılması sağlanabilir. Ses tanıma sistemlerinde Yapay sinir ağlarının kullanımı da bu şekilde gerçekleştirilmektedir.

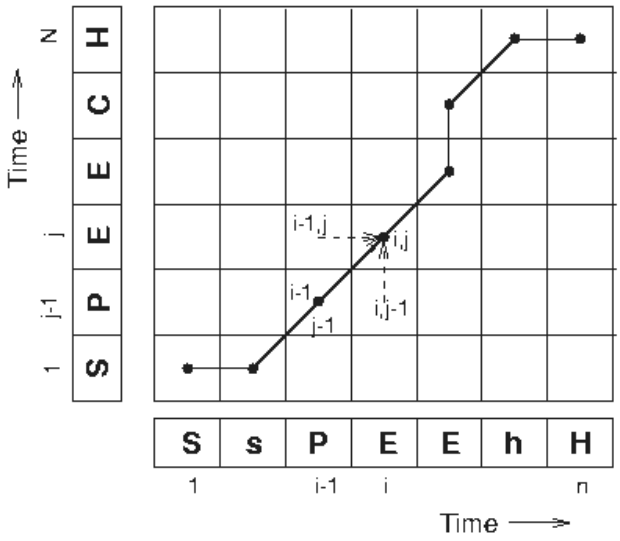
4.2.4.3. Dynamic Time Warping

Belirli bir sözcüğün seslendirilmesi, aynı kişi seslendirse bile zaman içinde farklılık gösterebilmektedir. Aynı sözcüğün seslendirilmesi, bir seslendirmede uzun, bir seslendirmede ise daha kısa zamanda gerçekleştirilebilir. Dynamic Time Warping algoritması yardımıyla, bu iki seslendirme, zaman içinde yayılarak ya da daraltılarak birbirine yaklaştırılmaya çalışılır. Yani bu iki seslendirmenin, zaman olarak örtüştürülmesi işlevi gerçekleştirilir.

DTW algortiması, sözcük tabanlı ses tanıma sistemlerinde etkin ve sıkça kullanılan bir yöntemdir. Bu yaklaşımla, çalışma anında saptanan sözcük kesimlemesi, sistemde kayıtlı sözcük şablonları ile seslendirme zamanları örtüştürülerek karşılaştırılması gerçekleştirilebilir.

Bu algoritmanın uygulanması dinamik programlama teknikleri kullanımı ile gerçekleştirilmektedir. İki ses sinyalinin arasındaki fark ve benzeşme çıkarılırken, büyük bir problem olan iki seri farkının hesaplanması, daha küçük bir problem olan seriden ikişer ikişer alınan örneklerin farklarının hesaplanması üzerine kurularak bu farklardan yararlanmaktadır.

Şekil 4.13'te iki ses sinyali arasında DTW algoritmasının zaman ekseninde uygulanması gösterilmektedir.



Şekil 4.13 İki ses sinyaline DTW algoritmasının uygulanması (Kale 2002)

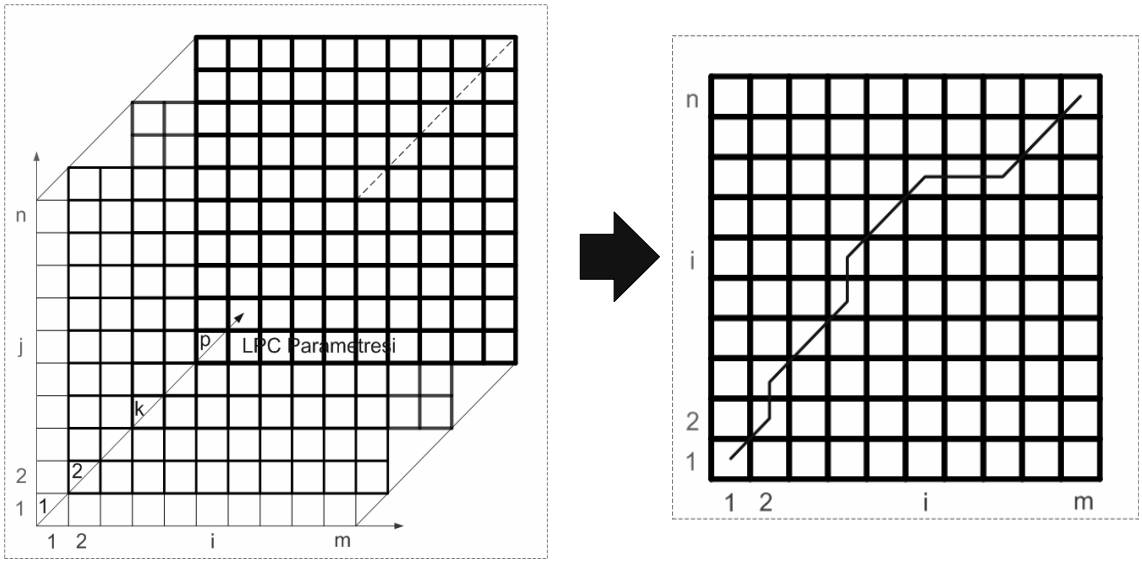
Şekil 4.13'te görüldüğü gibi, *N* sayıda örnekten oluşan ve y ekseninde yerleştirilen bir ses sinyali ile *n* sayıda örnekten oluşan ve x ekseninde yerleştirilmiş bir diğer sinyalin karşılaştırılması ve en yakın benzeşme maliyetinin belirlenmesi için daha küçük problemlerin çözümünden faydalanılmaktadır (Coleman 2005). Amaç; iki serinin birbirine dik olarak yerleştirilmesi ile oluşturulan düzlemde *(1, 1)* koordinatından, *(n, N)* koordinatına en düşük maliyetle ulaşabilmektir. Bunun için de, dinamik programlama teknikleri ile iki komşu koordinat arası geçiş maliyetlerinden yararlanılarak toplama ulaşılmaktadır.

Bu amaçla; ilk olarak x eksenindeki her bir örnek için, y eksenindeki örneklerin her biriyle olan farklar hesaplanarak matriste (ya da dinamik olarak oluşturulan iki boyutlu dizide) ilgili koordinata yazılmaktadır. Sonrasında *(1, 1)* koordinatından başlamak kaydıyla geliş yönleri aşağıdan, sol-aşağıdan veya soldan olmak kaydıyla diğer hücrelere geçiş maliyetleri her bir hücre için ayrı bir matriste hesaplanmaktadır. Bu hesaplama için bir hücreye gelinebilecek üç farklı nokta için maliyetler hesaplanıp en düşük olanı belirlenerek bu hücreye yazılmakta ve geliş yönü de ayrı bir matriste tutulmaktadır. Örnek olarak: Şekil 4.13'te *(i, j)* koordinatına geçiş için geçerli üç yön olan *(i, j-1), (i-1, j-1)* ve *(i-1, j)* koordinatlarından en düşük maliyetli olanı *(i-1, j-1)* koordinatı olarak hesaplanmış ve bu iki sinyalin bu kısmı için benzeşme bu yönde çıkarılmıştır.

Bu şekilde tüm matris doldurulduğunda; matriste *(n, N)* koordinatında bulunan değer, iki sinyal arası geçiş maliyetini ve Şekil 4.13'teki koyu siyah çizgi ile gösterilen geçişler ise, iki sinyalin ne şekilde benzeştiğini, yayıldığını göstermektedir.

Bu algoritmanın bir diğer kullanılma şekli ise; şablon olarak kayıtlı birden çok parametrenin ayrı ayrı DTW algoritması yardımıyla karşılaştırma işlemi için hazırlanması ve beraberce değerlendirilmesidir.

Şekil 4.14'te, LPC parametreleri üzerine DTW algoritmasının uygulanması gösterilmektedir.



Şekil 4.14 LPC parametreleri üzerine DTW algoritmasının uygulanması

Bu çalışma ile kullanılan uygulama şekli ise şöyledir: şablon olarak kaydedilmiş sözcüklerin LPC parametre değerleri ile çalışma anında ses kaydı ile alınmış sözcükten hesaplanan LPC parametre değerleri, LPC Analizcisi yardımıyla zaman içinde örtüştürülür. Bu örtüştürme sayesinde kayıtlı tüm şablonlar ile karşılaştırma sağlanarak her şablon için benzeme maliyetleri hesaplanmaktadır. Hesaplanan yakınlık maliyetleri yardımıyla en yakın şablona olan yakınlaşma oranı yüzde olarak hesaplanmakta ve eğer bu yakınlaşma oranı, tanımlanan eşik değerin üstünde ise eşleştirme gerçekleştirilmektedir.

LPC kodlayıcı çıkışında her bir çerçeve karşılığında dönüş değeri olarak *p* adet LPC parametresi alınmaktadır. İfade kuyruğu analizcisi, ifadeleri ifade kuyruğundan çekerek LPC kodlayıcısına kodlama için çerçevelere ayırarak göndermektedir. Kuyruktan

çekilen ifadenin *m* adet çerçeveden oluştuğu durumda: bunun sonucu olarak kodlanmış ifade boyutları *p* ve *m* olan iki boyutlu bir dizidir. Sistemde *n* adet çerçeveden alınmış ifadenin LPC ile kodlanmış karşılığı şablon olarak kayıtlı bulunsun. Bu durumda kayıtlı şablon, boyutları *p* ve *n* olan iki boyutlu bir dizi olacaktır.

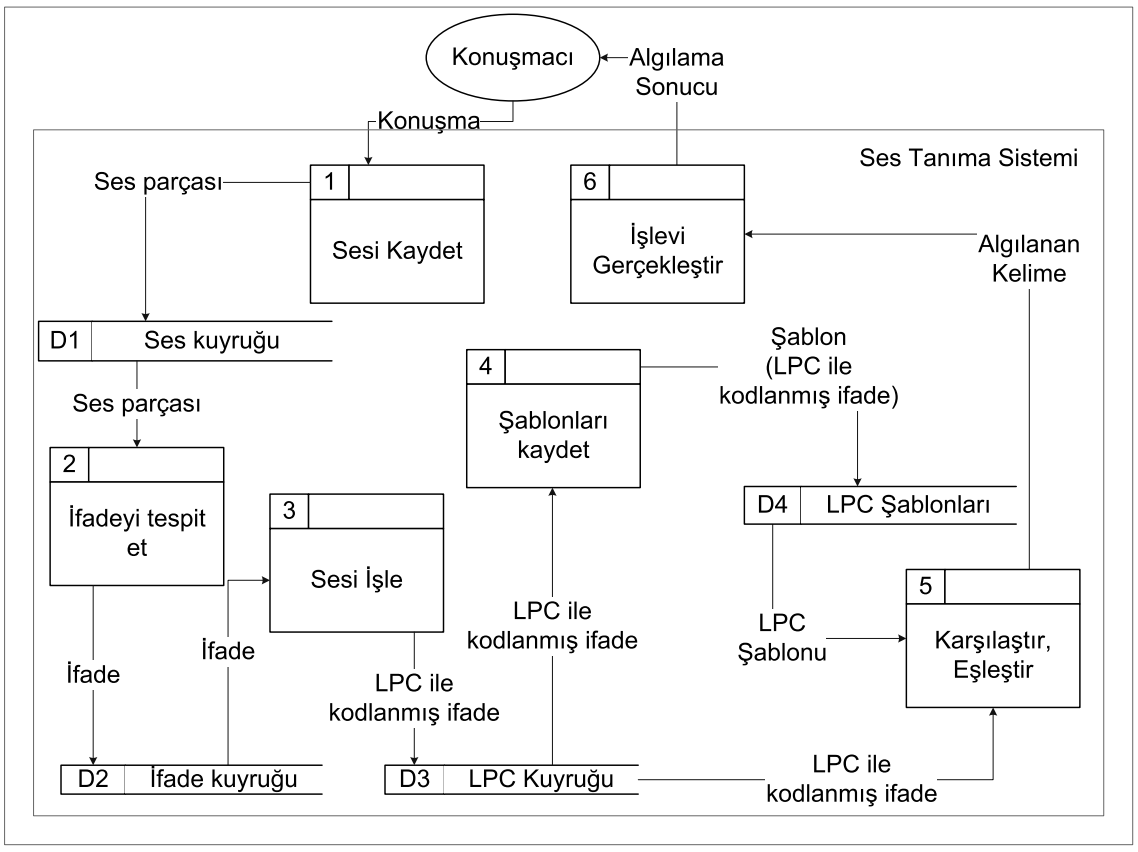
Bu iki dizi; üç boyutlu uzayda Şekil 4.14'te görüldüğü gibi birbirine dik olarak yerleştirilerek, *1*'den *p*'ye kadar görülen her bir LPC parametre düzleminde farklar hesaplanmakta, sonrasında her LPC parametre düzlemi hücre bazında ortalama farklar hesaplanarak tek bir düzleme indirgenmektedir. Sonrasında DTW algoritmasının uygulanması, iki ses sinyaline doğrudan DTW algoritmasının uygulanmasında olduğu gibi yapılmakta ve bir durumdan diğerine geçiş maliyetleri çıkarılarak karşılaştırma gerçekleştirilmektedir.

5. TÜRKÇE KOMUTLAR İÇİN SES TANIMA SİSTEMİ GELİŞTİRİLMESİ

Önceki bölümde ses tanıma sürecinin her bir aşaması için kullanılan tekniklere yer verilerek bu teknikler açıklanmıştır. Bu bölümde geliştirilen ses tanıma sisteminin modeli verilmiştir. Ardından geliştirilen ses tanıma sisteminde kullanılan temel sınıflar ve metotlara yer verilerek açıklanmış, sonrasında çalışma prensibi aktarılmıştır. Son olarak ise, geliştirilen ses tanıma sisteminde süreç bazında uygulanan teknikler ve sistem içindeki yerleri aktarılmıştır.

5.1. Geliştirilen Ses Tanıma Sistemi Modeli

Geliştirilen ses tanıma sistemi modeli, birinci seviye veri akış diyagramı yardımıyla Şekil 5.1'de verilmiştir.



Şekil 5.1 Geliştirilen ses tanıma sistemi modeli

Geliştirilen ses tanıma sisteminde ana işlevler; sesin kaydedilmesi, ifadenin saptanması, sesin işlenmesi, şablonların kaydedilmesi, karşılaştırma ve eşleştirme yapılması, son olarak ta işlevin gerçekleştirilmesi olarak belirlenmiştir. Bu işlevlerden şablonların kaydedilmesi, sadece sistemin eğitimi esnasında çalışmaktadır. Geliştirilen ses tanıma sistemi, kişiye bağımlı olarak tasarlanmıştır. Şablonların kişiye bağlı olarak güncellenmesi, şablonların kaydedilmesi işleviyle sağlanmaktadır.

Geliştirilen ses tanıma sistemi, iki farklı kipte çalışmaktadır. Bunlar Eğitim ve Uygulama kipi olarak değerlendirilebilir. Eğitim kipi, konuşmacının ses şablonlarının, sisteme tanıtılması için gerekli olmaktadır. Bu nedenle, sistem ilk çalıştırıldığında ve konuşmacı değiştikten sonra ilk çalıştırılışında, eğitim kipinde olması gerekmektedir. Geliştirilen ses tanıma sisteminin, Eğitim ve Uygulama kiplerinde çalışması ileriki kısımlarda verilecektir.

Geliştirilen ses tanıma sisteminde sürekli bir ses kayıt işlemi gerçekleştirilmektedir. Sesin kaydedilmesi sürerken, bir yandan da, kaydedilen sesteki ifadelerin saptanması gerekmektedir. Sonrasında saptanan ifadelerin işlenmesi ve şablonlarla karşılaştırılması, karşılaştırma sonucu eşleştirme yapılması da paralel olarak yürütülmektedir. Belirlenen kelimeye karşılık gelen işlevin gerçekleştirilmesi ve konuşmacıya algılama sonucunun yansıtılması da süreklilik gerektirmektedir.

Geliştirilen ses tanıma sisteminde, bütün işlevler süreklilik gerektirdiğinden, sistem çoklu kullanımlı (multi-threaded) olarak tasarlanmıştır. Sistem için gerekli ana işlevler belirlenmiş ve bu işler ayrıştırılarak iş parçacıklarına (*thread*) bölünmüştür. Geliştirilen ses tanıma sistemi, toplam dört ayrı iş parçacığına bölünmüştür. Geliştirilen ses tanıma sistemindeki iş parçacıkları ve görevleri Tablo 5.1'de verilmiştir.

Tablo 5.1 Geliştirilen ses tanıma sistemindeki iş parçacıkları ve görevleri

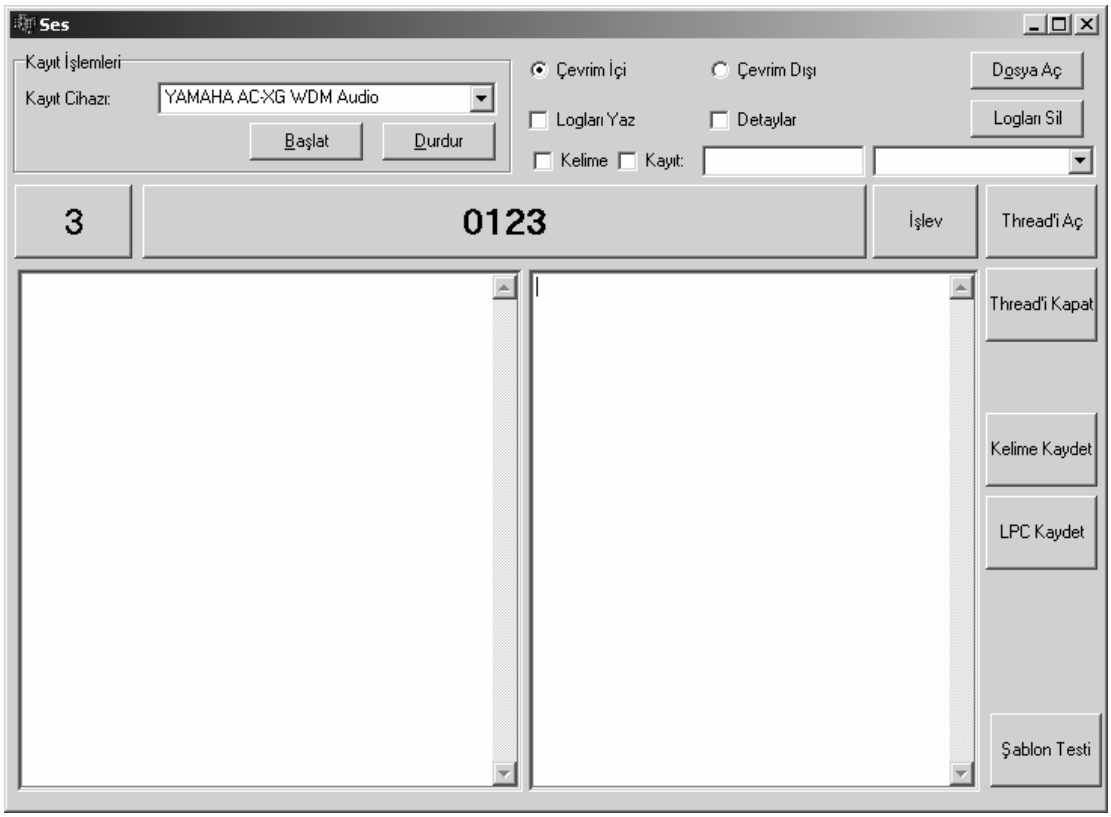
İŞ PARÇACIĞI	GÖREVİ
Ana İş Parçacığı	Kullanıcı grafik ara yüzü (sesin kaydedilmesi, işlevin gerçekleştirilmesi, şablonların kaydedilmesi)
Kuyruk Analizcisi	Ses kuyruğunun izlenmesi (ifadelerin saptanması)
İfade Kuyruğu Analizcisi	İfade kuyruğunun izlenmesi (sesin işlenmesi)
LPC Kuyruğu Analizcisi	LPC kuyruğunun izlenmesi (karşılaştırma ve eşleştirme yapılması)

Geliştirilen ses tanıma sistemi temel olarak dört veri kaynağı kullanmaktadır. Bu veri kaynakları ve her bir veri kaynağında tutulan veriler Tablo 5.2'de verilmiştir.

Tablo 5.2 Geliştirilen ses tanıma sistemindeki veri kaynakları

VERİ KAYNAĞI	VERİ KAYNAĞINDA TUTULAN VERİLER
Ses Kuyruğu	Sistemden gelen kayıtlı sesler
İfade Kuyruğu	Başlangıcı ve bitişi ile birlikte saptanmış sesli ifade
LPC Kuyruğu	LPC ile kodlanmış sesli ifadenin LPC parametreleri
LPC Şablonları	Eğitim kipiyle kaydedilmiş ve kelimelerle ilişkilendirilmiş LPC şablonları

Uygulama geliştirilirken kodlama için Borland® C++ Builder™ 6.0 kullanılmıştır. Şekil 5.2'de geliştirilen ses tanıma sisteminin kullanıcı grafik ara yüzüne ait ekran görüntüsü verilmektedir.



Şekil 5.2 Geliştirilen ses tanıma sistemi grafik ara yüzü

5.1.1. Geliştirilen ses tanıma sisteminde kullanılan temel sınıflar ve metotlar

Geliştirilen ses tanıma sisteminde kullanılan iş parçacıklarına geçilmeden önce; bu iş parçacıkları içinde gerek üye olarak tanımlı ve kullanılan, gerekse iş parçacığının metotları içinde kullanılmış temel sınıf ve metotları aktarmak yerinde olacaktır.

Bu sınıfların tanımları temel sınıflar olması nedeniyle ayrılmış ve "*UClasses.h*" dosyasında tanımlanmıştır. Bu temel sınıflara ait gerek kurucu, gerekse diğer metotların uygulaması; "*UClasses.cpp*" dosyasında yer almaktadır.

5.1.1.1 Ses parçası sınıfı - TSpeechBuffer

Kullanılan temel sınıflardan ilki ses parçası sınıfı - TSpeechBuffer olup ses kuyruğu ve ifade kuyruğu veri kaynaklarında kullanılmaktadır. Bu sınıfın tanımı, üyeleri ve metotları Şekil 5.3'te verilmektedir. Şekil 5.3'ün en altında, TSpeechBuffer sınıfının bir liste tanımı tip tanımlaması (*typedef*) komutu ile TSpeechBufferList olarak yapılmaktadır.

```
class TSpeechBuffer
{
public:
  unsigned int SBufferLength;
  short * SBuffer;

  TSpeechBuffer();
  TSpeechBuffer(int SBLength);

  ~TSpeechBuffer();

  TSpeechBuffer& operator =(const TSpeechBuffer& Copy);

  TSpeechBuffer * Add(TSpeechBuffer *NewBuffer);
};

typedef std::list<TSpeechBuffer *> TSpeechBufferList;
```

Şekil 5.3 Ses parçası sınıfı (TSpeechBuffer) tanımı, üye ve metotları

Bu sınıfa ait üyeler ve metotlar özet olarak şu şekilde tanımlanabilir.

- SBufferLength: Bu üye tanımı; işaretsiz tamsayı, yani *unsigned int* olarak yapılmıştır ve ses parçacığının uzunluğunu tutmaktadır.

- SBuffer: Bu üye tanımı kısa tamsayı işaretçisi, yani *short* * olarak yapılmıştır ve ses parçacığını tutmaktadır.

- TSpeechBuffer(): Bu metot kurucu metottur, sınıfın parametre girilmeksizin ilk oluşturulduğu anda çalışır ve sınıfa ait temel işlemler bu metot içinde yer almaktadır. İlk tanımların yapılması, bellek alanlarının ayrılması gibi bazı işlemler temel işlemler olarak nitelendirilmektedir.

- TSpeechBuffer(int SBLength): Bu metot ikinci bir kurucu metottur, sınıfın *SBLength* tamsayı parametresi ile birlikte ilk olarak oluşturulduğunda çalışır ve sınıfa ait temel işlemler bu metot içinde yer almaktadır. *SBLength* ile sınıf oluşturulurken uzunluğu biliniyorsa bu kurucu metot kullanılır.

- ~TSpeechBuffer(): Sınıfın yok edilmesinden hemen önce çalıştırılan metodudur ve sınıfın kullandığı bellek alanlarının bırakılması işlemleri bu metot içinde yer almaktadır.

- TSpeechBuffer& operator = (const ...): Bu metot ile *TSpeechBuffer* sınıfı için eşittir operatörünün, yani = kullanımı halinde sınıfın nasıl davranacağı tanımlanmaktadır.

- TSpeechBuffer * Add(TSpeechBuffer *NewBuffer): Bu metot parametre olarak kendi sınıfından bir işaretçi, yani *TSpeechBuffer* * alır ve yine kendi sınıfından bir işaretçi, yani *TSpeechBuffer* * döndürür. Yaptığı işlem ise, *NewBuffer* ses parçacığı sınıfı işaretçisi içindeki ses parçacığının, çağrıldığı ses parçacığı sınıfının ses parçacığına eklenmesi olarak özetlenebilir.

- typedef std::list<TSpeechBuffer *> TSpeechBufferList: Bu satır bir tip tanımlamasını ifade etmektedir. C++ standart kütüphanesindeki liste yapısı yardımı ile *TSpeechBuffer* * sınıfının, *TSpeechBufferList* ismiyle bir liste tanımlaması oluşturulmaktadır. Ses kuyruğu ve ifade kuyruğu veri kaynakları bu tipten birer sınıf listesi ile tanımlanmaktadır.

5.1.1.2 LPC parametre sınıfı - TLPCBuffer

Kullanılan temel sınıflardan bir diğeri, LPC kuyruğu veri kaynağının bir parçası olup kelime analizcisi ve LPC kuyruğu analizcisi iş parçacıklarında kullanılmaktadır. Bu sınıfın tanımı, üyeleri ve metotları Şekil 5.4'te verilmektedir.

```
class TLPCBuffer
{
public:
  unsigned int LPCCodeLength;
  short ** LPCCode;

  TLPCBuffer(int LPCCLength);

  ~TLPCBuffer();

  TLPCBuffer& operator =(const TLPCBuffer& Copy);
};

typedef std::list<TLPCBuffer *> TLPCCodeList;
```

Şekil 5.4 LPC parametre sınıfı (TLPCBuffer) tanımı, üye ve metotları

Bu sınıfa ait üyeler ve metotlar özet olarak şu şekilde tanımlanabilir.

- LPCCodeLength: Bu üye tanımı işaretsiz tamsayı, yani *unsigned int* olarak yapılmıştır ve LPC parametre serisinin vektör uzunluğunu tutmaktadır.

- LPCCode: Bu üye tanımı iki boyutlu kısa tamsayı işaretçisi, yani *short* ** olarak yapılmıştır ve LPC parametre vektör serisini tutmaktadır. İki boyutlu bu özelliğin bir boyutunun uzunluğu LPC parametre sayısı olup, *"GConstants.h"* dosyasında tanımlıdır ve kullanılan LPC kodlayıcısının çerçeve başına 12 LPC parametresi göndermesi nedeniyle sabit *12* olarak tanımlanmıştır.

- TLPCBuffer(int LPCCLength): Bu metot kurucu metottur, sınıfın tamsayı parametre ile birlikte ilk olarak oluşturulduğunda çalışır ve sınıfa ait temel işlemler bu metot içinde yer almaktadır. İlk tanımların yapılması, bellek alanlarının ayrılması gibi işlemler temel işlemler olarak nitelendirilmektedir.

- ~TLPCBuffer(): *TLPCBuffer* sınıfının yok edilmesinden hemen önce çalıştırılan metodudur ve bu sınıf tarafından kullanılan bellek alanlarının serbest bırakılması işlemleri bu metot içinde yer almaktadır.

- TLPCBuffer& operator = (const ...): Bu metot ile *TLPCBuffer* sınıfı için eşittir operatörünün, yani = kullanımı halinde *TLPCBuffer* sınıfının nasıl davranacağı tanımlanmaktadır.

- typedef std::list<TLPCBuffer *> TLPCBufferList: Bu satır *TSpeechBufferList* tanımında da olduğu gibi bir tip tanımlamasını ifade etmektedir. C++ standart kütüphanesindeki liste yapısı yardımı ile *TLPCBuffer* * sınıfının, *TLPCBufferList* ismiyle bir liste tanımlaması oluşturulmaktadır. LPC kuyruğu veri kaynağı bu tipten bir sınıf listesi ile tanımlanmaktadır.

5.1.1.3 LPC şablon sınıfı - TLPCPattern

Kullanılan temel sınıflardan en sonuncusu, LPC şablonları veri kaynağında kullanılmaktadır ve sistemde kayıtlı LPC parametre şablonlarının program açıldığında LPC şablonları veri kaynağına yüklenmesine yardımcı olmaktadır. Bu sınıfın tanımı, üyeleri ve metotları Şekil 5.5'te verilmektedir.

```
class TLPCPattern
{
public:
  short ActionCode;
  unsigned int LPCPatternLength;
  short * LPCPattern;
  AnsiString PatternName;

  TLPCPattern(int LPCPLength);

  ~TLPCPattern();

  TLPCPattern& operator =(const TLPCPattern& Copy);
};

typedef std::list<TLPCPattern *> TLPCPatternList;
```

Şekil 5.5 LPC şablon sınıfı (TLPCPattern) tanımı, üye ve metotları

Bu sınıfa ait üyeler ve metotlar özet olarak şu şekilde tanımlanabilir.

- ActionCode: Bu üye tanımı kısa tamsayı, yani *short int* olarak yapılmıştır ve LPC şablonunun hangi kelimeye ait olduğunu tutmaktadır. Geliştirilen ses tanıma sistemi, komut – kontrol sistemi olarak tasarlandığından; bu üye, aynı zamanda çalıştırılacak komutun dizin numarasını ifade etmektedir.

- LPCPatternLength: Bu üye tanımı işaretsiz tamsayı, yani *unsigned int* olarak yapılmıştır ve LPC parametrelerinin bir serisinden oluşan şablon uzunluğunu tutmaktadır.

- LPCPattern: Bu üye tanımı kısa tamsayı işaretçisi, yani *short* * olarak yapılmıştır ve LPC parametre serisini yani şablonu tutmaktadır. Bu üye tanımı aslında iki boyutlu bir veri içermesine karşın, sisteme kayıt ve sistemden geri yükleme işlemleri için kolaylık sağlaması bakımından tek boyutlu olarak tanımlanmıştır. LPC parametre serisi sisteme şablon olarak kaydedilirken, iki boyutlu seriden tek boyutlu seriye çevrim işlemi gerçekleştirilmekte ve kayıt işlemi yapılmaktadır.

- TLPCPattern(int LPCPLength): Bu metot kurucu metottur, *TLPCPattern* sınıfının *LPCPLength* tamsayı parametresi ile birlikte ilk olarak oluşturulduğunda çalışır ve sınıfa ait temel işlemler bu metot içinde yer almaktadır. İlk tanımların yapılması, bellek alanlarının ayrılması vb. işlemler temel işlemler olarak nitelendirilmektedir.

- ~TLPCPattern(): *TLPCPattern* sınıfının yok edilmesinden hemen önce çalıştırılan metodudur ve bu sınıf tarafından kullanılan bellek alanlarının serbest bırakılması işlemleri bu metot içinde yer almaktadır.

- TLPCPattern& operator = (const ...): Bu metot ile *TLPCPattern* sınıfı için eşittir operatörünün, yani = kullanımı halinde *TLPCPattern* sınıfının nasıl davranacağı tanımlanmaktadır.

- typedef std::list<TLPCPattern *> TLPCPatternList: Bu satır *TSpeechBufferList* ve *TLPCBufferList* tanımlarında da olduğu gibi sınıf listesinin bir tip tanımlamasını ifade etmektedir. C++ standart kütüphanesindeki liste yapısı yardımı ile *TLPCPattern* * sınıfının, *TLPCPatternList* ismiyle bir sınıf listesi

tanımlaması oluşturulmaktadır. LPC şablonları veri kaynağı bu tipten bir sınıf listesi ile tanımlanmaktadır.

5.1.1.4 Temel metotlar – UUtils.cpp

Geliştirilen ses tanıma sisteminde kullanılan ve temel metotlardan bir kısmını içeren bir dosya *"UUtils.cpp"*ye ait başlık dosyası *"UUtils.h"* Şekil 5.6'da bir kısmı gösterilen başlık dosyası içinde tanımlı metotları ile birlikte verilmektedir.

```
double RMSEnergy(short *Frame, int FrameLength);

short ZeroPassCount(short *VoiceIn, int Length);

short * Hamming(short *VoiceIn, int Length);
```

Şekil 5.6 UUtils.h'de tanımlı genel metotlar

"UUtils.h" dosyasında yer alan metotlar ve bu metotlarda yapılan işlemler şu şekilde tanımlanabilir.

- RMSEnergy(short *Frame, int FrameLength): Bu metot, *FrameLength* parametresi ile verilen uzunluktaki, *Frame* işaretçi parametresi ile verilen ses örneklerini içeren çerçeveye ait RMS değerinin hesaplanmasını gerçekleştirmektedir. Bu metot kuyruk analizcisi iş parçacığı, yani *TQueueAnalyser* tarafından ifade başlangıç ve bitişinin saptanmasında kullanılmaktadır.

- ZeroPassCount(short *VoiceIn, int Length): Bu metot, *Length* parametresi ile verilen uzunluktaki, *VoiceIn* işaretçi parametresi ile verilen ses örneklerini içeren çerçeveye ait sıfırı geçiş sayısının hesaplanmasını gerçekleştirmektedir. Bu metot kuyruk analizcisi iş parçacığı, yani *TQueueAnalyser* tarafından ifade başlangıç ve bitişinin saptanmasında *RMSEnergy* metodu ile birlikte kullanılmaktadır.

- Hamming(short *VoiceIn, int Length): Bu metot, *Length* parametresi ile verilen uzunluktaki, *VoiceIn* işaretçi parametresi ile verilen ses örnekleri serisini Hamming pencereleme fonksiyonundan geçirerek *short* * cinsinden Hamming

Pencereleme fonksiyonu çıkışını gerçekleştirmektedir. Bu metot ifade kuyruğu analizcisi iş parçacığı, yani *TWordAnalyser* tarafından LPC kodlaması öncesinde Hamming pencereleme fonksiyonunun gerçekleştirilmesi amacıyla kullanılmaktadır.

5.1.1.5 Temel metotlar – GRoutines.cpp

Geliştirilen ses tanıma sisteminde kullanılan ve temel metotlardan bir kısmını içeren bir dosya *"GRoutines.cpp"*ye ait başlık dosyası *"GRoutines.h"*nin bir kısmı Şekil 5.7'de başlık dosyası içinde tanımlı metotları ile birlikte verilmektedir.

```
#include "UClasses.h"

extern TSpeechBuffer * PopFromQueue();
extern TSpeechBuffer * PopFromWordQueue();
extern TLPCBuffer * PopFromLPCQueue();

extern short * LPCToShort(TLPCBuffer *LPCBuffer);

extern TLPCBuffer * CodeWithLPC(TSpeechBuffer * NewWordBuffer);
```

Şekil 5.7 GRoutines.h'de tanımlı genel metotlar

"GRoutines.h" dosyasında yer alan metotlar ve bu metotlarda yapılan işlemler şu şekilde tanımlanabilir.

- PopFromQueue(): Bu metot; geliştirilen ses tanıma sistemindeki ses kuyruğundan, yani sistemde tanımlı *SpeechBufferList* listesinden ilk sıradaki *TSpeechBuffer* nesnesini çekerek çağıran satıra döndürme işlevini gerçekleştirmektedir. Bu metot; kuyruk analizcisi iş parçacığı, yani *TQueueAnalyser* tarafından ses parçacıklarının ses kuyruğundan çekilmesi amacıyla kullanılmaktadır.

- PopFromWordQueue(): Bu metot; geliştirilen ses tanıma sistemindeki ifade kuyruğundan, yani sistemde tanımlı WordBufferList listesinden ilk sıradaki *TSpeechBuffer* nesnesini çekerek, çağıran satıra döndürme işlevini gerçekleştirmektedir. Bu metot; ifade kuyruğu analizcisi iş parçacığı, yani *TWordAnalyser* tarafından ifadelerin ifade kuyruğundan çekilmesi amacıyla kullanılmaktadır.

- PopFromLPCQueue(): Bu metot; geliştirilen ses tanıma sistemindeki LPC kuyruğundan, yani sistemde tanımlı *LPCCodeList* listesinden ilk sıradaki *TLPCBuffer* nesnesini çekerek, çağıran satıra döndürme işlevini gerçekleştirmektedir. Bu metot; LPC kuyruğu analizcisi iş parçacığı, yani *TLPCAnalyser* tarafından, LPC ile kodlanmış ifadelerin LPC kuyruğundan çekilmesi amacıyla kullanılmaktadır.

- LPCToShort(TLPCBuffer *LPCBuffer): Bu metot, *LPCBuffer* sınıfının nesne işaretçisi parametresi ile verilen LPC ile kodlanmış sesli ifade verisini *short** cinsine çevirmektedir. Bu metodun geliştirilen ses tanıma sisteminde kullanılma amacı ve yeri; şablon kaydedilmesi esnasında LPC parametrelerinin, LPC kuyruğundan çekildikten sonra kaydedilmesi ve yine karşılaştırma için şablonların şablon dosyalarından okunması işlemlerini kolaylaştırmaktır. Bu metot, şablonların kaydedilmesinde ana iş parçacığı tarafından ve karşılaştırma öncesinde LPC kuyruk analizcisi *TLPCAnalyser* tarafından kullanılmaktadır.

- CodeWithLPC(TSpeechBuffer *NewWordBuffer): Bu metot, *TSpeechBuffer* işaretçi parametresi ile verilen *TSpeechBuffer* sınıfından *NewWordBuffer* nesnesinin LPC ile kodlanmasını gerçekleştirmektedir. Bu metot; ifade kuyruğu analizcisi iş parçacığı, yani *TWordAnalyser* tarafından LPC kodlaması amacıyla kullanılmaktadır.

5.2. Geliştirilen Ses Tanıma Sistemi İş Parçacıkları

Bu kısımda geliştirilen ses tanıma sistemindeki iş parçacıkları ve bu iş parçacıklarının gerçekleştirdiği işlemler aktarılmıştır.

5.2.1. Ana iş parçacığı

Geliştirilen ses tanıma sistemindeki ana iş parçacığı, kullanıcı grafik ara yüzüdür. Kullanıcı grafik ara yüzü yardımıyla ses tanıma sisteminin, çalışma kipi belirlenmekte, kayıt cihazı seçilebilmekte ve ses tanıma sisteminin yönetimi sağlanabilmektedir.

Ana iş parçacığı, geliştirilen ses tanıma sisteminin işlemlerinden, sesin kaydedilmesi, algılanan kelimeye karşılık gelen işlevin gerçekleştirilmesi ve şablonların kaydedilmesi işlemlerini içinde barındırmaktadır.

Bu iş parçacığı yardımıyla sistemde kayıtlı şablonların yüklenmesi de gerçekleştirilir.

Şekil 5.8'de Ana iş parçacığına ait public değişken ve fonksiyonlar verilmektedir.

```
public:   // User declarations
  __fastcall TfrmSpeech(TComponent* Owner);

  TQueueAnalyser *QAAnalyser;
  TWordAnalyser *WordAnalyser;
  TLPCAnalyser *LPCAnalyser;

  TLPCPattern *LPCPattern;

//  TLPCPatternList LPCPatternList;

  void __fastcall TfrmSpeech::ProcessAction(int ActionToExecute);

  void __fastcall CMWaveData(TMessage &Message);
  void __fastcall CMWaveOpen(TMessage &Message);
  void __fastcall CMWaveClose(TMessage &Message);
  void __fastcall CMQANewWord(TMessage &Message);
  void __fastcall CMNewLog(TMessage &Message);
```

Şekil 5.8 Ana iş parçacığı (TfrmSpeech) public değişken ve fonksiyonları

Ana iş parçacığının sınıf ve değişken tanımlarına geçilmeden önce bu iş parçacığında kullanılmakta olan *TMessage* yapısının verilmesi yerinde olacaktır. *TMessage*, Borland C++ Builder kütüphanesinde *"Messages.hpp"* dosyasında bir yapı olarak tanımlıdır. *TMessage* yapısı tanımı, Şekil 5.9'da gösterilmektedir.

```
struct TMessage
{
  Cardinal Msg;
  union
  {
     struct
     {
        Word WParamLo;
        Word WParamHi;
        Word LParamLo;
        Word LParamHi;
        Word ResultLo;
        Word ResultHi;

     };
     struct
     {
        int WParam;
        int LParam;
        int Result;
     };

  };
} ;
```

Şekil 5.9 TMessage yapısı

- ProcessAction(int ActionToExecute): *ActionToExecute* parametresi ile verilen işlevi gerçekleştirir. İşlevler *"GConstants.h"* dosyasında tanımlanmaktadır.

- CMWaveData(TMessage &Message): Kayıt cihazından veri geldiğinde yapılması gerekli işlevleri gerçekleştirir. Sisteme kayıt amacıyla boş tampon bellek alanının gönderilmesi, gelen dolu tampon bellek alanının, yani ses parçacığının ses kuyruğuna atılması bu fonksiyon içinde çağrılır. Ara yüze *CM_WAVE_DATA* mesajının gelmesi ile çağrılan fonksiyondur. Çağrılması waveInProc içinden yapılmaktadır. Kayıt cihazından sistem tarafından o anda kayıt için kullanılan tampon bellek bölgesi dolduğunda *WIM_DATA* mesajının waveInProc fonksiyonuna geçilmesi sonucunda çağrılmaktadır.

- CMWaveOpen(TMessage &Message): Kayıt cihazının, kayıt amacıyla açılması gerçekleştiğinde, yapılması gereken işlemler; bu metot içinde yer almaktadır. Bu işlemler; sisteme kayıt amacıyla ilk boş tampon bellek alanlarının gönderilmesi, gönderim öncesi bu tampon alanların hazırlanması olarak sayılabilir. Ara yüze *CM_WAVE_OPEN* mesajının gelmesi ile çağrılan fonksiyondur. Çağrılması waveInProc içinden yapılmaktadır. Kayıt cihazından, kayıt için kayıt cihazının başarıyla açıldığı mesajı *WIM_OPEN* mesajının *waveInProc* fonksiyonuna geçilmesi sonucunda çağrılmaktadır.

- CMWaveClose(TMessage &Message): Kayıt cihazında, kayıt işlemi bittiğinde cihaza kapama mesajı gönderilmesi gerekmektedir. Bu mesaj gönderilip kayıt cihazı kapama işlevi başarıyla gerçekleştiğinde, yapılması gereken işlemler bu metot içinde yer almaktadır. Ara yüze *CM_WAVE_CLOSE* mesajının gelmesi ile çağrılan fonksiyondur. Çağrılması *waveInProc* içinden yapılmaktadır. Kayıt cihazından, kayıt cihazının başarıyla kapatıldığını ifade eden *WIM_CLOSE* mesajının *waveInProc* fonksiyonuna gelmesi sonucunda bu metot çağrılmaktadır.

- CMQANewWord(TMessage &Message): Geliştirilen ses tanıma sisteminin seslendirilen komutu, yani kelimeye karşılık işlevleri gerçekleştirebilmesi için yazılmış olan metottur. LPC kuyruğu analizcisi tarafından ara yüze, yani ana iş parçacığına, *CM_QA_NEWWORD* mesajının gönderimi ile birlikte çalıştırılmaktadır. Ana iş parçacığına bu mesaj geçilmeden önce, kuyruk

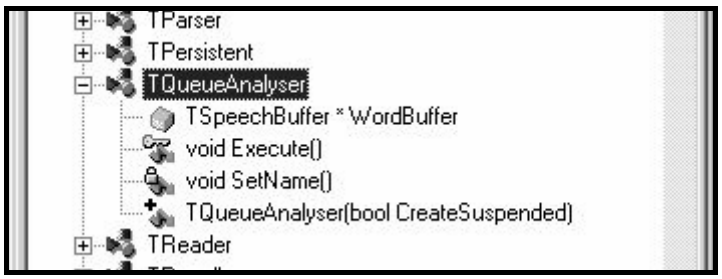
analizcisi tarafında, bu yapıya ait *WParam* özelliğine işlenecek komutun dizin numarası ataması yapılır.

- CMNewLog(TMessage &Message): Kullanıcı grafik ara yüzünde *'Logları yaz'* onay kutusu seçildiğinde, çalışma anında ara yüzün sağ altındaki kısımda loglar yazılmaktadır. Geliştirilen ses tanıma sisteminin çalışmasındaki sürekliliği sağlamak amacıyla, kullanıcı grafik ara yüzüne bu logların yazdırılması, herhangi bir iş parçacığı tarafından, log kuyruğuna logların eklenmesi, ardından eklenen log satırı sayısının *TMessage* yapısının *WParam* özelliğine atanması ve ana iş parçacığına *CM_NEWLOG* mesajı gönderilmesi ile sağlanmaktadır. Bu mesajın ana iş parçacığına ulaşması sonucu bu metot çağrılmakta ve log kuyruğundan sayısı *WParam* parametresinde belirtilmiş miktarda log satırı çekilerek ara yüze yazdırılmaktadır.

5.2.2. Kuyruk analizcisi

Kuyruk analizcisi iş parçacığı yardımıyla, ses kuyruğu izlenmektedir. Kuyruk analizcisi iş parçacığı, geliştirilen ses tanıma sistemi işlemlerinden ifadenin saptanması işlemini gerçekleştirmektedir.

Şekil 5.10'da Kuyruk analizcisi iş parçacığına ait sınıf yapısı verilmektedir.



Şekil 5.10 Kuyruk analizcisi iş parçacığı (TQueueAnalyser) sınıf yapısı

- WordBuffer: Bu üye, *Execute* metodu içinde kullanılmaktadır. *Execute* metodunun her bir döngüsünde, ses kuyruğundan sıradaki ses parçası çekilmektedir. Konuşma başlangıcı içeren ilk çerçeve ile WordBuffer sınıfının bir nesnesi, çerçeve boyutu uzunluğunda oluşturulmakta ve konuşma bitişi tespit edilene kadar olan tüm çerçeveler bu nesnenin *Add* metodu ile bu nesnenin *SBuffer* özelliğinin sonuna eklenerek ifadeler oluşturulmaktadır. Bu şekilde her

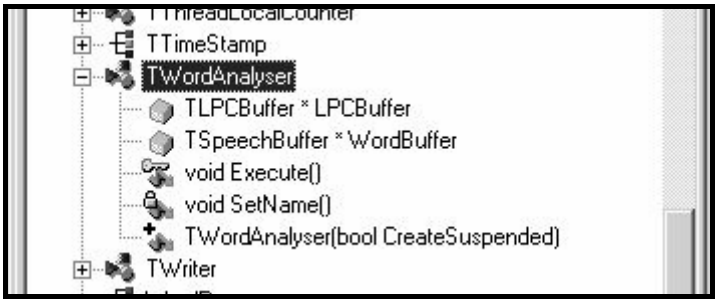
ifade tamamlandığında, İfade kuyruğunun sonuna eklenmekte ve sıradaki çerçeveden itibaren kuyruk analizi devam etmektedir.

- Execute(): Bu iş parçacığının çalışması, iş parçacığının sistemde yaratılması ile birlikte *Execute* metodu ile başlar ve sonlandırana kadar kuyruk izlenmesi işlevini yapar. Bu metot içinde ses parçacıkları ses kuyruğundan alınır, çerçeveler halinde RMS değeri ve sıfırı geçiş sayıları hesaplanarak konuşma geçen çerçeveler tespit edilir. Bu şekilde, ses parçacıklarından konuşma içeren çerçeveler birleştirilerek, konuşmanın başlangıç ve bitişi ile birlikte ifadeler oluşturulur. Oluşturulan ifadeler, ifade kuyruğuna atılırlar.

5.2.3. İfade kuyruğu analizcisi

İfade kuyruğu analizcisi, ifade kuyruğunu izlemekte ve ifade kuyruğundan alınan sesli ifadenin işlenmesini gerçekleştirmektedir. Geliştirilen ses tanıma sistemindeki sesin işlenmesi aşaması bu iş parçacığı yardımıyla gerçekleştirilmektedir.

Şekil 5.11'de İfade kuyruğu analizcisi iş parçacığına ait sınıf yapısı verilmektedir.



Şekil 5.11 İfade kuyruğu analizcisi iş parçacığı (TWordAnalyser) sınıf yapısı

- WordBuffer: Bu üye, *Execute* metodu içinde kullanılmaktadır. *Execute* metodunun her bir döngüsünde, ifade kuyruğundan sıradaki ifade çekilmektedir. Konuşma başlangıcı ve bitişi ile birlikte ifade kuyruğundan çekilen her bir ifade, bu özelliğe atanır ve takip edilir. Yeni bir *TSpeechBuffer* nesnesi oluşturulmadan, kuyruktan çekilen nesne bu özelliğe atanır. Bu nesne içindeki ses parçası *SBuffer*, önce *Hamming* fonksiyonundan geçirilir, ardından bu nesne *CodewithLPC* fonksiyonu ile kodlanarak LPC parametre serisine çevrilir.

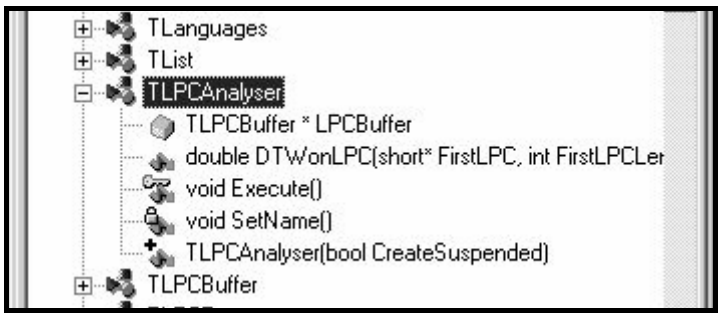
- LPCBuffer: Bu üye, *Execute* metodu içinde kullanılmaktadır. Execute metodunun her bir döngüsünde, *CodewithLPC* fonksiyonuna kodlama amacıyla gönderilen her bir *TWordBuffer* nesnesine karşılık, *TLPCBuffer* sınıfına ait yeni bir nesne oluşturulur ve bu özelliğe atanır. Bu şekilde oluşturulmuş her bir nesne bir işleme tabii olmadan LPC kuyruğunun sonuna eklenmektedir.

- Execute(): Bu iş parçacığının çalışması, iş parçacığının sistemde yaratılması ile birlikte *Execute* metodu ile başlar ve iş parçacığı sonlandırılana kadar ifade kuyruğunun izlenmesi işlevini yapar. Bu metot içinde Kuyruk Analizcisi yardımıyla oluşturulmuş ve ifade kuyruğuna eklenmiş ifadeler alınır, Hamming pencereleme fonksiyonundan geçirilirler, LPC ile kodlanarak, LPC kuyruğuna eklenirler.

5.2.4. LPC kuyruğu analizcisi

LPC kuyruğu analizcisi; LPC kuyruğunu izlemekte, LPC kuyruğundaki her bir LPC ile kodlanmış ifadenin kayıtlı LPC şablonları ile karşılaştırılmasını gerçekleştirmekte ve en yakın şablonu seçerek eşleştirme yapmaktadır. Bu iş parçacığı ile, geliştirilen ses tanıma sistemindeki karşılaştırma ve eşleştirme işlemi gerçekleştirilmektedir

Şekil 5.12'de LPC kuyruğu analizcisi iş parçacığına ait sınıf yapısı verilmektedir.



Şekil 5.12 LPC kuyruğu analizcisi iş parçacığı (TLPCAnalyser) sınıf yapısı

- LPCBuffer: Bu üye, *Execute* metodu içinde kullanılmaktadır. *Execute* metodunun her bir döngüsünde, LPC kuyruğundan çekilen *LPCBuffer* nesnesi bu özelliğe atanır. Bu şekilde alınan her bir *LPCBuffer* nesnesinin, sistemde kayıtlı tüm şablonlarla karşılaştırılmak üzere takip edilmesi amacıyla bu üye kullanılmaktadır.

- Execute(): Bu iş parçacığının çalışması, iş parçacığının sistemde yaratılması ile birlikte *Execute* metodu ile başlar ve iş parçası sonlandırılana kadar LPC kuyruğunun izlenmesi işlevini yapar. Bu metot içinde İfade kuyruğu analizcisi yardımıyla oluşturulmuş ve LPC kuyruğuna eklenmiş LPC ile kodlanmış ifadeler sırayla alınır ve sistemde kayıtlı tüm şablonlarla LPC parametreleri üzerinde DTW algoritmasının uygulanması yardımıyla karşılaştırılmaktadır. Hesaplanan yakınlık maliyetleri yardımıyla en yakın şablona olan yakınlaşma oranı yüzde olarak hesaplanmakta ve eğer bu yakınlaşma oranı, tanımlanan eşik değerin üstünde ise eşleştirme gerçekleştirilmektedir. Varsa eşleşen kelime dizin numarası ya da eşleştirme gerçekleştirilemediği bilgisi; yani *-1*, Ana iş parçacığına işlevin gerçekleştirilmesi için iletilir.

- DTWonLPC(short* FirstLPC, int FirstLPCLength,short* SecondLPC, int SecondLPCLength): Bu metot yardımıyla uzunlukları ile birlikte verilen iki LPC parametre serisi DTW algoritması yardımıyla dinamik veri yapıları kullanılarak zamana göre yayılarak karşılaştırılır ve bir yakınlık maliyeti hesaplanır. Dönüş değeri; *double* cinsindendir.

5.3. Geliştirilen Ses Tanıma Sistemindeki İşlemler

Bu kısımda geliştirilen ses tanıma sistemindeki işlemler ses tanıma sistemi sürecine bağlı olarak süreç sırasıyla aşama aşama verilmiş ve çalışma prensibi aktarılmıştır.

5.3.1. Sesin kaydedilmesi

Sesin kaydedilmesi; ses kayıt cihazı seçimi sonrasında, çalışma kipi seçildikten sonra, *'Başlat'* düğmesine basılarak kullanıcı tarafından başlatılmaktadır. Sesin kaydedilmesi, kullanıcı tarafından *'Durdur'* düğmesine basılana dek sürekli olarak devam etmektedir.

Sesin kaydedilmesinde, Microsoft®'un dalga biçimi ses servisleri için olan Uygulama Programlama Ara yüzü (Microsoft® Waveform Audio Services API) kullanılmıştır. Bu uygulama programlama ara yüzünde yer alan, ses kaydı esnasında kullanılan yapı ve fonksiyonlarla ilgili bilgi Ek-1'de verilmektedir.

Sesin kaydedilmesi esnasında, ses kayıt cihazından gelen ses verisi, gelir gelmez *Ses kuyruğu*'na atılmakta ve boş bir bellek bölgesi, sisteme kayıt için gönderilmektedir. Bu sayede kayıt sürekliliği sağlanmakta ve konuşma kayıpları önlenmektedir.

Ses kayıt biçimi, yani ses kaydının kaç kanal olacağı, örnek başına bit sayısı, saniyedeki örnek sayısı vb. parametreler, kullanıcı ara yüzü ilk açıldığında uygulama tarafından varsayılan olarak atanmakta ve değiştirilememektedir.

5.3.2. İfadenin saptanması

Bu işlemin amacı, ifadenin saptanmasını gerçekleştirmektir. Bu işlemin çalışma prensibi şu şekilde açıklanabilir; Ses kuyruğu sürekli olarak izlenmekte ve sesin kaydedilmesi işlemi tarafından ses kuyruğuna atılan ses parçacıkları sırasıyla alınarak, her bir parça çerçevelere bölünmektedir. Çerçeve boyutu sistemde sabit olarak belirlenmiştir. Alınan her çerçevede RMS değeri ve sıfırı geçiş sayısı hesaplanmaktadır. Hesaplanan bu değerler, sistemde sabit olarak tanımlı eşik değerlerle karşılaştırılmak suretiyle, bu çerçevede konuşma olup olmadığı belirlenmektedir.

Bu çerçevenin konuşma içerdiği saptandığında, daha önceden konuşma başlamış ise bu ifade verisinin sonuna eklenmektedir. Konuşma bu çerçeve ile başlıyorsa, boş bir ifade verisi oluşturulmakta ve bu çerçeve ifade verisine eklenmektedir.

Bu işlem esnasında, konuşma kayıplarını en aza indirgemek için konuşma yeni başlamış ise; varsa bir önceki çerçeve ifade verisinin ilk çerçevesi olarak, konuşma bitişi tespit edildiğinde ise son çerçeve ifade verisinin son çerçevesi olarak eklenmektedir. Bu şekilde konuşma başlangıcı ve bitişi ile belirlenen sesli ifade, ifade kuyruğuna eklenmektedir.

5.3.3. Sesin işlenmesi

Bu işlemde, ifade kuyruğu izlenmektedir. İfadenin saptanması işleminde, belirlenen ifadelerin atıldığı ifade kuyruğundan sırasıyla alınan ifade verileri önce bir pencereleme fonksiyonundan geçirilmektedir. Bu sayede ifade verisinin orta kısımlarının belirginleşmesi sağlanmaktadır.

Pencereleme fonksiyonundan geçirilmiş ifade verisi; LPC çerçevesi boyutuyla belirlenen çerçeve boyutlarında alınarak, LPC kodlayıcı fonksiyona gönderilirler. Her

bir çerçeveye karşılık gelen LPC parametreleri LPC kodlayıcısında hesaplanır. LPc kodlayıcısndan alınan bu parametreler birleştirilerek kodlanmış ifade verisi oluşturulur. Sonuç olarak ifade kuyruğundan alınan her bir ifade verisi, LPC parametrelerinden oluşan bir seri LPC parametresi verisine dönüştürülmektedir.

Bu şekilde ifade verisinin tamamı, LPC ile kodlandıktan sonra oluşan LPC verisi, LPC kuyruğuna atılmaktadır.

5.3.4. Şablonların kaydedilmesi

Bu işlem, sadece sistem eğitim kipinde başlatıldığında çalıştırılabilmektedir. Sistem ilk çalıştırıldığında sistemde kayıtlı şablon bulunmadığından, ilk olarak LPC şablonlarının oluşturulması gerekmektedir. Bu işlemin sağlanması için kullanıcı grafik ara yüzünde *'Kayıt'* onay kutusu seçili olması gerekmektedir. Ses kayıt işlemi bu şekilde başlatıldığında, sistem tarafından LPC kuyruk analizcisi çalıştırılmamakta, bu sayede sesin işlenmesi aşaması ile oluşturulmuş LPC parametre verilerinin LPC kuyruğunda birikmesi sağlanmaktadır.

Sistemde kelimeler ile şablonların ilişkilendirilmesi, şablon kayıt işlemi esnasında şablona verilen isim ile yapılmaktadır. İsim yapısı sistemde aşağıdaki şekilde belirlenmiştir.

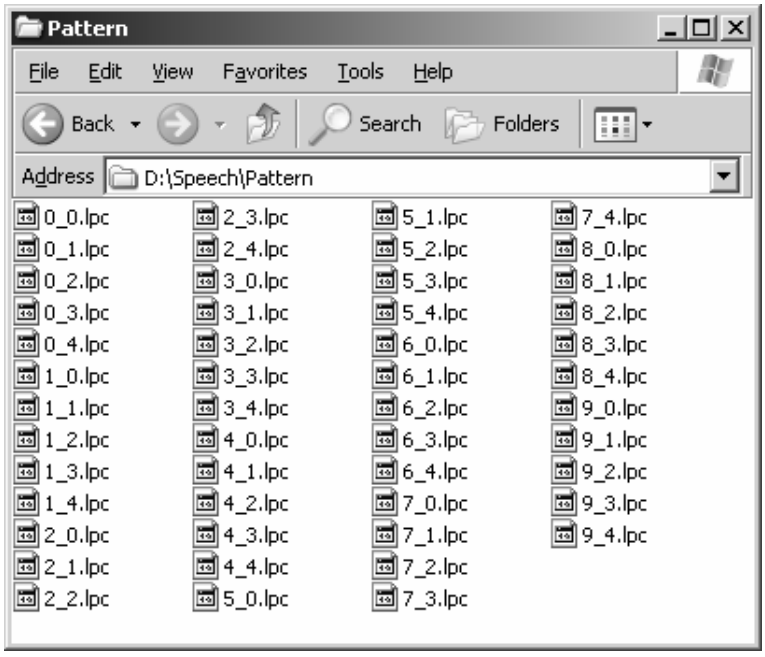
ŞABLON_İSMİ = ŞABLON_DİZİN_ETİKETİ + "_" + *ŞABLON_NUMARASI*

Burada *ŞABLON_DİZİN_ETİKETİ*, sistemde tanınması beklenen sayılardan her biri için rakam karşılığı olarak verilmesi gerekmektedir. Yani *'Sıfır'* için *0*, *'Bir'* için *1*, benzer şekilde *'Dokuz'* için *9* olarak verilir. *'Aç'*, *'Kapat'*, *'Sil'*, *'İptal'* ve *'Ara'* komutları için ise *10 - 14* arası şablon etiketi olarak belirlenmiştir. Tablo 5.3'te her bir komuta karşılık gelen şablon etiketi, etiket dizin sırasıyla verilmiştir. Şablonlara *ŞABLON_NUMARASI* ve dosya uzantısı kayıt esnasında otomatik olarak verilmektedir.

Tablo 5.3 Komutlara karşılık gelen şablon etiketleri

Şablon Etiketi	Komut	Şablon Etiketi	Komut	Şablon Etiketi	Komut
0	Sıfır	5	Beş	10	Aç
1	Bir	6	Altı	11	Kapat
2	İki	7	Yedi	12	Sil
3	Üç	8	Sekiz	13	İptal
4	Dört	9	Dokuz	14	Ara

Şekil 5.13'te kayıtlı şablonlar için örnek bir dizin verilmiştir. Bu örnekte şablonların isim yapısı da görülmektedir.



Şekil 5.13 Kayıtlı şablonlar örnek dizin

Sistemde kayıtlı şablonların uygulama dizini içindeki *'\Pattern'* dizininde bulunması gerekmektedir ve uygulamanın açılışında şablon isimleri yukarıdaki yapıda beklenmektedir. Bu nedenle şablon kayıt işlemi sırasında her kelime ayrı ayrı seslendirilmeli ve kaydedilmelidir. Bir kelimenin seslendirme işlemi gerçekleştirildikten sonra kayıt durdurularak, *'Kayıt'* onay kutusu yanında bulunan metin kutusuna *ŞABLON_DİZİN_ETİKETİ* + "_" şeklinde şablon ön eki yazılması gerekmektedir. Sonrasında *'LPC Kayıt'* düğmesine basılarak, LPC kuyruğundaki veriler şablon olarak kaydedilebilmektedir. Örneğin *'Sıfır'* şablonları kaydedilirken metin kutusuna *'0_'*, *'Bir'* şablonları kaydedilirken ise *'1_'* yazılması yeterlidir.

Şablonlar sistem tarafından otomatik olarak uygulama dizinine kaydedilmektedir. Şablonların buradan olması gereken *'\Pattern'* dizinine, kullanıcı tarafından taşınması gerekmektedir. Doğrudan şablon dizinine kayıt işleminin yapılmamasının nedeni, başarılı şablonların üzerine yanlışlıkla kaydetme işlemi yapılmasını önlemektir. Başarılı şablonların silinmesi, sistem başarımını düşürecektir.

5.3.5. Karşılaştırma ve eşleştirmenin gerçekleştirilmesi

Bu işlemde; LPC kuyruğundan LPC ile kodlanmış ifadeler sırasıyla alınarak, uygulama açılırken yüklenen tüm şablonlarla karşılaştırılmaktadır. Karşılaştırma işlemi hesaplanan LPC parametreleri üzerinde DTW algoritmasının uygulanması ile gerçekleştirilmektedir.

Bu işlem esnasında, LPC_i; LPC ile kodlanmış ifade ve $LPC_{j;}$ sıradaki LPC şablonu, karşılaştırma yapan fonksiyona gönderilmektedir. Bu fonksiyonda LPC_i ve LPC_j bir biri üzerinde zamana göre yayılarak LPC_i'nin, LPC_j'ye ne şekilde benzediği ve bu benzeşmenin maliyeti hesaplanmaktadır. Bu maliyet değeri, fonksiyonun geri dönüş değeridir.

Tüm şablonlarla bu şekilde benzeme maliyeti adım adım hesaplanarak, en düşük ve en yüksek benzerlik maliyetleri her bir adımda saklanmaktadır. Tüm şablonlarla karşılaştırma tamamlandıktan sonra, en düşük ve en yüksek benzerlik maliyetleri yardımıyla en yakın şablona olan yakınlaşma oranı yüzde olarak hesaplanmaktadır. En yakın şablon için hesplanan bu yakınlaşma oranı, tanımlanan eşik değerin üstünde ise eşleştirme gerçekleştirilmektedir.

Eşleşme var ise, eşleşen kelime için *ŞABLON_DİZİN_ETİKETİ*, eşleşme yok ise eşleştirme gerçekleştirilemediği bilgisi; yani -1, Ana iş parçacığına işlevin gerçekleştirilmesi için iletilmektedir.

5.3.6. İşlevin gerçekleştirilmesi

İşlevin gerçekleştirilmesi, geliştirilen ses tanıma sistemindeki son işlem olarak yer almaktadır. Bu işlemde algılanan kelime, karşılaştırma ve eşleştirme aşamasında iletilen *ŞABLON_DİZİN_ETİKETİ* yardımıyla alınarak işleve dönüştürülmektedir.

Algılanan son kelime sol üstte, *'Kayıt İşlemleri'* başlıklı grup kutusu altında bulunan küçük pano üzerine yazılmaktadır. Aynı zamanda, bu küçük pano yanında bulunan, büyük panonun sonuna eklenerek, algılanan rakamların bir serisi görüntülenmektedir. Komutlara özel olarak ise; *'Sil'* komutu geldiğinde, büyük panoya yazılan en son rakam buradan silinmekte; *'İptal'* komutu geldiğinde ise, büyük panoya yazılmış rakam serisi var ise, bu rakam serisi tamamıyle silinmektedir. *'Aç'*, *'Kapat'* ve *'Ara'* komutları için

ise sadece sol üstteki küçük panoya komut yazdırılmakta, ayrıca bir işlem yürütülmemektedir.

Ana iş parçacığına sıradaki işlev olarak *-1* geldiğinde, en son seslendirilen komutun, sistemde kayıtlı şablonlara benzemediği; yani komutun anlaşılmadığını ifade etmek üzere sol üstteki küçük panoya ve hemen yanındaki büyük panonun sonuna *'?'* yazdırılmaktadır.

6. SONUÇ VE ÖNERİLER

Bir önceki bölümde geliştirilen ses tanıma sisteminin modeli verildi, sonrasında çalışma prensibi aktarıldı, geliştirilen ses tanıma sisteminde süreç bazında uygulanan teknikler ve sistem içindeki yerleri aktarıldı. Bu bölümde ise, geliştirilen ses tanıma sistemi değerlendirilmiştir. Geliştirilen ses tanıma sisteminin sonuçlarına ve bu sonuçlara bağlı olarak ileriki çalışmalar için önerilere yer verilmiştir.

6.1 Sonuç

Bu çalışma ile, kişiye bağımlı sözcük tabanlı bir komut – kontrol sistemi geliştirilmiştir. Geliştirilen uygulamanın program kodları ve çalıştırılabilir versiyonu, CD ortamında Ek-2 olarak verilmiştir.

Geliştirme esnasında, ifade tesbiti için *sıfırı geçiş sayısı* ve *RMS* değeri hesabı, pencereleme için *Hamming* pencereleme fonksiyonu, kodlayıcı olarak *LPC* kullanılmış, karşılaştırma için ise LPC parametreleri üzerinde *DTW* algoritması uygulanmıştır.

Yapılan araştırmalar ve çalışmalar doğrultusunda başarımın zamana bağlı olarak değişkenlik gösterdiği saptanmıştır. Bunun en büyük nedeninin ise sesin, bir seslendirilişi ile bir diğer seslendirilişi arasında, aynı kişi seslendirse dahi çokça farklılık göstermesi olarak görülmüştür. Diğer bir nedenin ise, ortam gürültüsü olduğu çalışmalar sırasında tespit edilmiştir.

Çalışma sonrasında varılan bir diğer sonuç ise, çalışma kapsamında öngörülen LPC parametreleri üzerinde DTW uygulanması ile kelimelerin en yakın olanının tespit edilmesi tekniğinin, kelime tanıma konusunda tek başına çok başarılı olmamasıdır.

Varılan bu sonuçlar ışığında, önerilen yaklaşımlar, sonraki kısımda öneriler başlığı altında verilmiştir.

Bu çalışma sonucunda, mevcut tekniklerden yararlanmak suretiyle ses tanıma sistemi oluşturulmuş ve elde edilen bu sistem, bu konuda çalışma yapacak kişilere bu

tekniklerin uygulama içinde kullanımı ve ilişkilendirilmesi hakkında örnek oluşturacaktır.

Geliştirilen bu sistemin başarımının arttırılması ile Türkçe uyumlu telefon otomasyon uygulamalarında, fiziksel engelli kişilerin bilgisayar kullanımlarının sağlanmasında, araç sürücülerinin belli işlevleri sesli komutlarla gerçekleştirmelerinde ve üretim sahasından veri toplama amacıyla kullanımı sağlanabilir.

6.2 Öneriler

Sesin, zaman içinde seslendirilmesinde görülen değişkenlik nedeniyle, en yakın şablonun bulunması ve doğru kelimenin her zaman tespit edilmesi pek mümkün olmamaktadır. Bu nedenle önerilebilecek yaklaşım, farklı zamanlarda kayıtlar sonrası ortaya çıkan şablonlardan, ortalama bir şablon hesaplanması ve tanıma için bu şablonun kullanılması başarımı arttırabilir.

Sadece LPC parametreleri üzerinde DTW uygulanması yerine, kaydedilmiş sesin farklı tekniklerle başka özellikleri de çıkarılarak LPC'ye ek parametreler olarak saklanabilir ve bu özelliklerin tümünün üzerinde DTW algoritmasının uygulanması ile başarım arttırılabilir.

Gerek kodlayıcı, gerekse karşılaştırma ve eşleştirme için farklı tekniklerin kullanımı ile daha başarılı sonuçlar alınabilir. Bu tekniklerin karşılaştırılması yapılarak Türkçe Konuşma Dili için kullanabilecek uygun tekniklerin çıkarımı yapılabilir.

KAYNAKLAR

Cole, R. A., Mariani, J., Uszkoreit, H., Zaenen, A., and Zue, V. (1995) Survey of the State of the Art in Human Language Technology, Varile, G., and Zampolli, A., http://cslu.cse.ogi.edu/HLTsurvey/HLTsurvey.html (21.11.1995).

Coleman, J. (2005) Introducing Speech and Language Producing, Cambridge University Press, Cambridge, 301s.

Cook, S. (2002) Speech Recognition HOWTO, http://www.gear21.com/speech/index. html (19.04.2002).

Hagan, M. T., Demuth, H. B., and Beale, M. H. (1995) Neural Network Design, Brooks Cole, PWS Publishing, Boston, 736s.

Hermansky, H. (1990) Perceptual Linear Predictive (PLP) Analysis of Speech. Journal Of Acoustic Society of America, 87: (4) 1738-1752.

Huang, X., Acero, A., and Hon, H. W. (2001) Spoken Language Processing: A Guide to Theory, Algorithm and System Development (1st Ed.), Prentice Hall PTR, New Jersey, 980s.

Ibarra, O. M., and Curatelli, F. (2000) A Brief Introduction to Speech Analysis and Recognition, An Internet Tutorial., http://www.mor.itesm.mx/~omayora/Tutorial/ tutorial.html (05.05.2000)

Jurafsky, D., and Martin, J. H. (2000) Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition (1st Ed.), Prentice Hall, New Jersey, 934s.

Kale, K. R. (2002) Dynamic Time Warping, http://www.cnel.ufl.edu/~kkale/dtw.html (04.04.2002)

Rabiner, L., and Juang, B. (1993) Fundamentals of Speech Recognition, Prentice Hall, New Jersey, 496s.

Rabiner, L., and Schafer, R. W. (1978) Digital Processing of Speech Signals, Prentice Hall PTR, 512s.

Robinson, T. (1998) Speech Analysis Tutorial, http://svr-www.eng.cam.ac.uk/~ajr/ SpeechAnalysis/ (23.01.1998)

Smith, J. O. (2003) Mathematics of The Discrete Fourier Transform with Audio Applications, http://ccrma.stanford.edu/~jos/mdft/Alias_Operator.html (03.01.2006)

Smith, S. W. (1999) The Scientist's and Engineer's Guide to Digital Signal Processing (2nd Ed.), California Technical Publishing, San Diego, 650s.

EKLER

Ek-1 – Mıcrosoft® Dalga biçimi Ses Servisleri Uygulama Programlama Ara yüzü (Mıcrosoft® Waveform Audio Services API) İçinde Tanımlı Ses Kaydı İçin Kullanılan Yapılar Ve Fonksiyonlar

Aşağıda Microsoft® dalga biçimi ses servisleri uygulama programlama ara yüzünde tanımlı ses kayıt işlemi için kullanılan yapılar verilmektedir.

- WAVEFORMATEX: Bu yapı dalga biçimi ses verisinin biçimini tanımlamaktadır. Dalga biçimi ses veri biçiminde genel olan bilgi bu yapıya dahil edilmiştir. Bu yapı, üyeleri ile birlikte aşağıdaki şekilde tanımlanmaktadır.

```
typedef struct {
    WORD wFormatTag;
    WORD nChannels;
    DWORD nSamplesPerSec;
    DWORD nAvgBytesPerSec;
    WORD nBlockAlign;
    WORD wBitsPerSample;
    WORD cbSize;
} WAVEFORMATEX;
```

Bu yapıya ait üyeler, sırasıyla şu şekilde verilmektedir.

- *wFormatTag*: Bu üye, ses biçiminin tipini belirtmektedir.

- *cChannels*: Bu üye, ses verisindeki kanal sayısını belirtmektedir.

- *nSamplesPerSec*: Saniyedeki örnek sayısı (Hertz) cinsinden örnekleme oranını belirtmektedir. Aynı yapı için *wFormatTag* üyesi *WAVEFORMAT_PCM* olarak belirlenmiş ise bu üye için genel değerler 8.0 kHz, 11.025 kHz, 22.05 kHz ve 44.1 kHz olabilmektedir. Diğer PCM dışı biçimler için, bu üyenin değeri, bu biçim takısının üreticisinin belirlediği özelliklere göre hesaplanması gerekmektedir.

- *nAvgBytesPerSec*: Verilen biçim için, saniyedeki byte cinsinden gerekli ortalama veri transfer oranını belirtmektedir. Aynı yapı için *wFormatTag* üyesi *WAVEFORMAT_PCM* olarak belirlenmiş ise bu üye için değer *nSamplesPerSec*

ve *nBlockAlign* çarpımı olması gerekmektedir. Diğer PCM dışı biçimler için, bu üyenin değeri, bu biçim takısının üreticisinin belirlediği özelliklere göre hesaplanması gerekmektedir. Yürütme ve kaydetme yazılımları, bu üye yardımı ile tampon boyutlarını değerlendirmektedirler.

- *nBlockAlign*: Bu üye, byte cinsinden blok yerleşimini belirtmektedir. Blok yerleşimi, *wFormatTag* biçim tipi için, verinin en küçük birim değerini ifade etmektedir. Aynı yapı için *wFormatTag* üyesi *WAVEFORMAT_PCM* olarak belirlenmiş ise bu üye için değer *nChannels* ve *wBitsPerSample* çarpımının, byte'taki bit sayısına yani 8'e bölümü olması gerekmektedir. Diğer PCM dışı biçimler için, bu üyenin değeri, bu biçim takısının üreticisinin belirlediği özelliklere göre hesaplanması gerekmektedir.

- *wBitsPerSample*: Aynı yapı için verilen *wFormatTag* biçim tipinin saniyedeki örnek sayısını belirtmektedir. Aynı yapı için *wFormatTag* üyesi *WAVEFORMAT_PCM* olarak belirlenmiş ise bu üye için değer 8 ya da 16 olarak verilebilmektedir. Diğer PCM dışı biçimler için, bu üyenin değeri, bu biçim takısının üreticisinin belirlediği özelliklere göre hesaplanması gerekmektedir. Bazı sıkıştırma şemaları *wBitsPerSample* üyesi için bir değer tanımlayamamaktadırlar. Bu gibi durumlarda *wBitsPerSample* değerinin 0 olarak verilmesi gerekmektedir.

- *cbSize*: Bu üye, PCM dışı diğer biçimler için WAVEFORMATEX yapısının sonuna eklenmiş ilave biçim bilgisi için byte olarak boyutunu belirtmektedir. Aynı yapı içindeki *wFormatTag* için ek özelliklere ihtiyaç yoksa üye değerinin 0 olarak verilmesi gerekmektedir.

- WAVEHDR: Bu yapı dalga biçimi ses tampon belleğinin başlığını tanımlamaktadır. Bu yapı, üyeleri ile birlikte aşağıdaki şekilde tanımlanmaktadır.

```
typedef struct {
    LPSTR lpData;
    DWORD dwBufferLength;
    DWORD dwBytesRecorded;
    DWORD dwUser;
```

```
    DWORD dwFlags;
    DWORD dwLoops;
    struct wavehdr_tag * lpNext;
    DWORD reserved;
} WAVEHDR;
```

Bu yapıya ait üyeler, sırasıyla şu şekilde verilmektedir.

- *lpData*: Dalga biçimi tampon belleğinin adres bilgisini belirtmektedir.

- *dwBufferLength*: Tampon bellek için, byte olarak bellek uzunluğunu belirtmektedir.

- *dwBytesRecorded*: Tampon bellek, yani *WAVEHEADER* kayıt için kullanıldığı durumda, bu üye tampon bellek içinde byte cinsinden ne kadarlık kısmın dolu olduğunu belirtmektedir.

- *dwUser*: Kullanıcı için veri alanını belirtmektedir.

- *dwFlags*: Tampon bellek hakkında bilgi sağlayan bayrak alanı belirtmektedir. Bu üye için tanımlı değerle: *WHDR_BEGINLOOP* (yürütme tampon belleklerinde kullanılmaktadır ve bir çevrime ait ilk tampon bellek olduğunu bildirmektedir.), *WHDR_DONE* (cihaz sürücüsü tarafından, tampon bellek ile işinin bittiğini ve uygulamaya geri döndürdüğünü belirtmek üzere ayarlanmaktadır.), *WVHDR_ENDLOOP* (yürütme tampon belleklerinde kullanılmaktadır ve bir çevrime ait son tampon bellek olduğunu bildirmektedir.), *WVHDR_INQUEUE* (sistem tarafından, tampon belleğin yürütme amacıyla kuyruğa eklendiğini belirtmek amacıyla ayarlanmaktadır.), *WVHD_PREPARED* (sistem tarafından tampon belleğin hazırlanma işleminin yapıldığını belirtmek amacıyla ayarlanmaktadır.) olarak tanımlanmaktadır.

- *dwLoops*: Bu üye yürütme bellekleri tarafından kullanılmaktadır ve çevrimin kaç kere yürütüleceğini belirtmektedir.

- *wavehdr_tag*: Sistem tarafından ayrılmıştır.

- *Reserved*: Sistem tarafından ayrılmıştır.

- WAVEINCAPS: Bu yapı dalga biçimi ses kayıt cihazının yeteneklerini tanımlamaktadır. Bu yapı, üyeleri ile birlikte aşağıdaki şekilde tanımlanmaktadır.

```
typedef struct {
    WORD wMid;
    WORD wPid;
    MMVERSION vDriverVersion;
    CHAR szPname[MAXPNAMELEN];
    DWORD dwFormats;
    WORD wChannels;
    WORD wReserved1;
} WAVEINCAPS;
```

Bu yapıya ait üyeler, sırasıyla şu şekilde verilmektedir.

- *wMid*: Bu üye, ses kayıt cihazının, cihaz sürücüsü için üretici kimliğini belirtmektedir. Üretici kimlikleri, "*mmreg.h*" dosyasında tanımlanmaktadır.

- *wPid*: Bu üye, ses kayıt cihazının, cihaz sürücüsü için ürün kimliğini belirtmektedir. Ürün kimlikleri, "*mmreg.h*" dosyasında tanımlanmaktadır.

- *vDriverVersion*: Bu üye, ses kayıt cihazının cihaz sürücüsünün versiyon numarasını belirtmektedir. Yüksek değerlikli byte, ana versiyon numarasını, düşük değerlikli byte yan versiyon numarasını vermektedir.

- *szPname*: Bu üye ürün ismini belirtmektedir.

- *dwFormats*: Desteklenen standart biçimlerinin bir kombinasyonunu belirtmektedir. Desteklenen biçimler ise *WAVE_FORMAT_1M08* (11.025 kHz, mono, 8-bit), *WAVE_FORMAT_1M16* (11.025 kHz, mono, 16-bit), *WAVE_FORMAT_1S08* (11.025 kHz, stereo, 8-bit), *WAVE_FORMAT_1S16* (11.025 kHz, stereo, 16-bit), *AVE_FORMAT_2M08* (22.05 kHz, mono, 8-bit), *WAVE_FORMAT_2M16* (22.05 kHz, mono, 16-bit), *WAVE_FORMAT_2S08* (22.05 kHz, stereo, 8-bit), *WAVE_FORMAT_2S16* (22.05 kHz, stereo, 16-bit), *WAVE_FORMAT_4M08* (44.1 kHz, mono, 8-bit), *WAVE_FORMAT_4M16*

(44.1 kHz, mono, 16-bit), *WAVE_FORMAT_4S08* (44.1 kHz, stereo, 8-bit), *WAVE_FORMAT_4S16* (44.1 kHz, stereo, 16-bit) olarak verilmektedir.

- *wChannels*: Ses kayıt cihazının, mono (1), yoksa stereo (2) kaydı mı desteklediğini gösteren sayıyı belirtmektedir.

- *wReserved1*: Doldurmayı belirtmektedir.

Aşağıda Microsoft® dalga biçimi ses servisleri uygulama programlama ara yüzünde tanımlı ses kayıt işlemi için kullanılan fonksiyonlar verilmektedir.

- waveInGetNumDevs(): Bu fonksiyon sistemde bulunan dalga biçimi ses kayıt cihazlarının sayısını geri döndürmektedir. Sistemde kayıtlı hiçbir ses kayıt cihazı yok ise fonksiyon dönüş değeri 0 olmaktadır.

- waveInOpen(): Belirtilen ses kayıt cihazının kayıt işlemlerini yapabilmek amacıyla bir tipini oluşturmaktadır. Bir başka deyişle seçilen kayıt cihazını, kayıt işlemlerini yapabilmek amacıyla açar.

- waveInPrepareHeader(): Kayıt için sisteme gönderilen tampon bellek bölgesinin hazırlanması işlemini gerçekleştirmektedir. Toplam üç parametre almaktadır. İlk parametre ses kayıt cihazının tanıtıcısı olup, ikinci parametre hazırlığı yapılacak tampon bellek bölgesinin tanımını veren *WAVEHDR* yapısının adresi olmaktadır. En son parametre ise, ikinci parametre olarak adresi verilen *WAVEHDR* nesnesinin byte cinsinden toplam uzunluğudur. Bu fonksiyon dört farklı sonuç değeri döndürmektedir bunlar ise yapılan işlem başarılı, yani bir hata yoksa *MMSYSERR_NOERROR*, verilen cihaz tanıtıcısı geçerli bir cihaz tanıtıcısı değil ise *MMSYSERR_INVALHANDLE*, cihaz sürücüsü mevcut değil ise *MMSYSERR_NODRIVER* ve son olarak ta eğer bellek ayrımı yapılamıyor ise *MMSYSERR_NOMEM* sonucunu döndürmektedir.

- waveInUnprepareHeader(): Bu fonksiyon, daha önce waveInPrepareHeader fonksiyonunun çağrılması ile ayrımı yapılan bellek bölgesini temizler. Bu fonksiyonda üç adet parametre almaktadır. Bu parametrelerden ilki, ses kayıt cihazının tanıtıcısı; ikincisi, bellek bölgesi temizlenecek *WAVEHDR* yapısının adresi ve son parametre ise ikinci parametre olarak adresi verilen *WAVEHDR*

yapısının byte cinsinden boyutudur. Dönüş değerleri ise, eğer yapılan işlem başarılı, yani bir hata yoksa *MMSYSERR_NOERROR*, verilen cihaz tanıtıcısı geçerli bir cihaz tanıtıcısı değil ise *MMSYSERR_INVALHANDLE*, cihaz sürücüsü mevcut değil ise *MMSYSERR_NODRIVER*, eğer bellek ayrımı yapılamıyor ise *MMSYSERR_NOMEM* ve son olarak ta eğer *WAVEHDR* yapısının adresini tutan ve ikinci parametre olarak verilen adresteki bellek bölgesi halen kayıt kuyruğunda ise *WAVERR_STILLPLAYING* sonucunu döndürmektedir.

- waveInClose(): Verilen ses kayıt cihazını, kapatır. Ses kayıt cihazının tanıtıcısını gösteren tek bir parametre almaktadır. Sonuç olarak ise beş farklı sonuç değeri döndürebilmektedir. Bunlar: eğer yapılan işlem başarılı, yani bir hata yoksa *MMSYSERR_NOERROR*, verilen cihaz tanıtıcısı geçerli bir cihaz tanıtıcısı değil ise *MMSYSERR_INVALHANDLE*, cihaz sürücüsü mevcut değil ise *MMSYSERR_NODRIVER*, eğer bellek ayrımı yapılamıyor ise *MMSYSERR_NOMEM* ve son olarak ta eğer kayıt kuyruğunda halen tampon var ise *WAVERR_STILLPLAYING* sonucunu döndürmektedir.

- waveInReset(): Bu fonksiyon, verilen ses kayıt cihazındaki kayıt işlemini durdurulmasını ve şu anki pozisyonunun sıfıra çekilmesini sağlamaktadır. Bu fonksiyonun çağrılması sonucu, kayıt için bekleyen tüm tampon bellek bölgeleri tamamlandı olarak işaretlenmekte ve uygulamaya geri döndürülmektedir. Ses kayıt cihazının tanıtıcısını tek parametre olarak alırken, dört farklı dönüş değerinden birini sonuç olarak döndürmektedir. Dönebilecek sonuç değerleri: yapılan işlem başarılı, yani bir hata yoksa *MMSYSERR_NOERROR*; verilen cihaz tanıtıcısı geçerli bir cihaz tanıtıcısı değil ise *MMSYSERR_INVALHANDLE*; cihaz sürücüsü mevcut değil ise *MMSYSERR_NODRIVER* ve son olarak ta eğer bellek ayrımı yapılamıyor ise *MMSYSERR_NOMEM* sonuçlarından biri olabilmektedir.

- waveInStart(): Verilen ses kayıt cihazında ses kayıt işlemini başlatma, bu fonksiyon çağrılarak yapılmaktadır. Bu fonksiyon ses kayıt işlemi yapılacak ses kayıt cihazının tanıtıcısını tek parametre olarak alır ve sonuç olarak dört farklı dönüş değerinden birini döndürmektedir. Bu sonuç değerleri, eğer yapılan işlem başarılı ise *MMSYSERR_NOERROR*; verilen cihaz tanıtıcısı geçerli bir cihaz

tanıtıcısı değil ise *MMSYSERR_INVALHANDLE*; cihaz sürücüsü mevcut değil ise *MMSYSERR_NODRIVER* ve son olarak ta eğer bellek ayrımı yapılamıyor ise *MMSYSERR_NOMEM* sonuçlarından biri olabilmektedir.

- waveInStop(): Bu fonksiyon, ses kayıt işlemini durdurmaktadır. Yine tek parametre olarak ses kayıt cihazının tanıtıcısını alır, sonuç olarak dört farklı dönüş değerinden birinin döndürmektedir. Bu sonuç değerleri, işlem başarılı olarak gerçekleşmiş ise *MMSYSERR_NOERROR*; verilen cihaz tanıtıcısı geçerli bir cihaz tanıtıcısı değil ise *MMSYSERR_INVALHANDLE*; cihaz sürücüsü mevcut değil ise *MMSYSERR_NODRIVER* ve son olarak ta eğer bellek ayrımı yapılamıyor ise *MMSYSERR_NOMEM* sonuçlarından biri olabilmektedir.

- waveInAddBuffer(): Verilen ses kayıt cihazına, ses kayıt işlemi yapılması amacıyla bir bellek tampon bölgesinin gönderilmesi işlemi bu fonksiyonun çağrılması ile gerçekleştirmektedir. Bu fonksiyon toplam ilki ses kayıt cihazının tanıtıcısı, ikincisi gönderilecek tampon bellek bölgesini tanımını yapan *WAVEHDR* yapısının adresi ve sonuncusu da adresi ikinci parametre olarak gönderilen *WAVEHDR* yapısının byte cinsinden toplam boyutu olmak üzere toplam üç adet parametre almaktadır. Dönüş değeri olarak, işlem başarılı olarak gerçekleşmiş ise *MMSYSERR_NOERROR*; verilen cihaz tanıtıcısı geçerli bir cihaz tanıtıcısı değil ise *MMSYSERR_INVALHANDLE*; cihaz sürücüsü mevcut değil ise *MMSYSERR_NODRIVER*, eğer bellek ayrımı yapılamıyor ise *MMSYSERR_NOMEM* ve son olarak ta ikinci parametre olarak adresi verilen *WAVEHDR* yapısının tanımladığı tampon bellek bölgesinin ayrımı gerçekleştirilemiyor ise *WAVERR_UNPREPARED* sonuçlarından biri olabilmektedir.

- waveInGetDevCaps(): Bu fonksiyon ile belirtilen ses kayıt cihazının ses kayıt yetenekleri alınabilmektedir. Toplam üç adet parametre almaktadır. Bu parametrelerden ilki, ses kayıt cihazının tanımlayıcısı olup, bir cihaz tanıtıcısı olabileceği gibi, açık bir ses kayıt cihazının tanıtıcısı da olabilmektedir. İkinci parametre kayıt cihazının kayıt yetenekleri bilgisiyle doldurulacak *WAVEINCAPS* yapısının adresi olup, son parametre ise, ikinci parametre olarak adresi verilen *WAVEINCAPS* yapısının byte cinsinden toplam boyutudur. Bu fonksiyon dört farklı dönüş değerinden birini sonuç olarak döndürmektedir. Bu

sonuç değerleri, işlem başarılı olarak gerçekleşmiş ise *MMSYSERR_NOERROR*; verilen cihaz tanıtıcısı geçerli sınırlarda değil ise *MMSYSERR_BADDEVICEID*; cihaz sürücüsü mevcut değil ise *MMSYSERR_NODRIVER* ve son olarak ta eğer bellek ayrımı yapılamıyor ise *MMSYSERR_NOMEM* sonuçlarından biri olabilmektedir.

ÖZGEÇMİŞ

Murat Kemal BAYGÜN, 25 Temmuz 1975 tarihinde Balıkesir'de doğmuştur. Evli ve bir kız çocuğu babasıdır.

Eylül 1986 – Haziran 1993 tarihleri arasında Balıkesir Sırrı Yırcalı Anadolu Lisesi'nde hazırlık, ortaokul ve lise eğitimine devam etmiştir. Ekim 1993'te, İstanbul Üniversitesi Mühendislik Fakültesi Bilgisayar Bilimleri Mühendisliği'nde Lisans eğitimine başlamış ve Temmuz 1997'de Lisans eğitimini tamamlamıştır.

Ekim 2000'den bu yana Egekom® Ltd ve Egecom® Ltd. bünyesinde Yazılım Geliştirme üzerine çeşitli görevleri üstlenmiştir. Şu anda halen Egecom® Ltd. Şti.'nde Kıdemli Yazılım Geliştirme ve Ar-Ge Uzmanı olarak görevini sürdürmektedir.