TC
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ

BİYOMETRİK GÜVENLİK SİSTEMLERİ VE YÜZ TANIMAYA DAYALI ÇEVRİMİÇİ SINAV SİSTEMİ

YÜKSEK LİSANS TEZİ

Zihni KAYA

Bilgisayar Mühendisliği Ana Bilim Dalı

Bilgisayar Mühendisliği Programı

OCAK 2016

TC
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ

BİYOMETRİK GÜVENLİK SİSTEMLERİ VE YÜZ TANIMAYA DAYALI ÇEVRİMİÇİ SINAV SİSTEMİ

YÜKSEK LİSANS TEZİ

Zihni KAYA
(Y1313.010023)

Bilgisayar Mühendisliği Ana Bilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Ali GÜNEŞ

OCAK 2016

T.C.
İSTANBUL AYDIN ÜNİVERSİTESİ
FEN BİLİMLER ENSTİTÜSÜ MÜDÜRLÜĞÜ

Yüksek Lisans Tez Onay Belgesi

Enstitümüz Bilgisayar Mühendisliği Ana Bilim Dalı Bilgisayar Mühendisliği Tezli Yüksek Lisans Programı **Y1313.010023** numaralı öğrencisi **Zihni KAYA**'nın **"BİYOMETRİK GÜVENLİK SİSTEMLERİ VE YÜZ TANIMAYA DAYALI ÇEVRİMİÇİ SINAV SİSTEMİ"** adlı tez çalışması Enstitümüz Yönetim Kurulunun 19.01.2016 tarih ve 2016/03 sayılı kararıyla oluşturulan jüri tarafından oybirliği ile Tezli Yüksek Lisans tezi olarak kabul...edilmiştir.

Öğretim Üyesi Adı Soyadı **İmzası**

Tez Savunma Tarihi :09/02/2016

1)Tez Danışmanı: Prof. Dr. Ali GÜNEŞ .....

2) Jüri Üyesi : Yrd. Doç. Dr. Vassılya UZUN .....

3) Jüri Üyesi : Yrd. Doç. Dr. Ferdi SÖNMEZ .....

Not: Öğrencinin Tez savunmasında **Başarılı** olması halinde bu form **imzalanacaktır**. Aksi halde geçersizdir.

YEMİN METNİ

Yüksek Lisans / Doktora tezi olarak sunduğum "……..……
……
……" adlı çalışmanın, tezin proje safhasından sonuçlanmasına kadarki bütün süreçlerde bilimsel ahlak ve geleneklere aykırı düşecek bir yardıma başvurulmaksızın yazıldığını ve yararlandığım eserlerin Bibliyografya'da gösterilenlerden oluştuğunu, bunlara atıf yapılarak yararlanılmış olduğunu belirtir ve onurumla beyan ederim. (…/…/20…)

Aday / İmza

TEŞEKKÜR

Tezimin hazırlanması esnasında hiçbir yardımı esirgemeyen öğrencilerine büyük destek olan, bilimsel deney imkânlarını sonuna kadar bizlerin hizmetine veren, tez danışman hocam, Sayın Prof. Dr. Ali GÜNEŞ' e, tez çalışmalarım esnasında çok büyük fedakârlıklarla bana destek olan eşime teşekkür ederim.

ÖNSÖZ

Bilgisayar ve internet teknolojilerindeki gelişmelere bağlı olarak çevrimiçi eğitim sistemleri de hızla gelişmektedir. Her yenilikte olduğu gibi, bu gelişim de beraberinde birçok problemi getirmektedir. Bu sorunlardan birisi, ölçme-değerlendirme amacıyla yapılan çevrimiçi sınav uygulamalarında karşımıza çıkmaktadır. Sınava giren kişinin kimlik tespiti önemli bir güvenlik sorunudur. Bu çalışmada, çevrimiçi sınava giren kişinin kimlik tespitinde kullanılmak üzere en etkin biyometrik güvenlik sistemleri araştırıldı. Bu sistemlerden yüz tanıma teknolojisi kullanılarak güvenli sınav uygulaması gerçekleştirildi.

OCAK-2016 **Zihni KAYA**

Uzman

İÇİNDEKİLER

Sayfa

KISALTMALAR

OPENCV	Open Source Computer Vision
DNA	Deoksiribonükleik asit
USB	Universal Serial Bus
SQL	Structured Query Language
BLOB	Binary Large Object
YTT	Yüz Tanıma Teknolojisi
PHP	Hypertext Preprocessor
JPEG	Joint Photographic Experts Group
YSA	Yapay Sinir Ağları

ÇİZELGE LİSTESİ

SAYFA

ŞEKİL LİSTESİ

Sayfa

BİYOMETRİK GÜVENLİK SİSTEMLERİ VE YÜZ TANIMAYA DAYALI ÇEVRİMİÇİ SINAV SİSTEMİ

ÖZET

Bu çalışmada, çevrimiçi sınavlarda bir başkasının yerine sınava girişin önlenmesi amacıyla yüz tanımayla kimlik tespiti yapan bir uygulama geliştirilmiştir. Öncelikle ad, soyad ve yüz görüntüsü gibi verilerden oluşan kullanıcı bilgisi veri tabanına kaydedilmektedir. Sınav zamanında, kullanıcıdan web kamera yoluyla alınan yüz görüntüsü, veri tabanına daha önceden kaydedilen yüz görüntüsü ile karşılaştırılmaktadır. Karşılaştırma sonucuna göre kişi sınava alınmakta ya da alınmamaktadır.

Günümüzde, hemen hemen her bilgisayarda bir web kamera olması, yüz tanımanın en güvenli biyometrik sistemlerinden biri olması ve kullanımı kolay olması nedeniyle bu tez çalışmasında yüz tanıma sistemi kullanılmıştır.

Bu tezde, otomatik insan yüzü tanımak için Özyüz yöntemi kullanılmış olup oldukça yüksek başarım elde dilmiştir.

Anahtar Kelimeler: Biyometrik Güvenlik Sistemleri, Yüz Tanıma Sistemi, Çevrimiçi Sınav Sistemi

BIOMETRIC SECURITY SYSTEMS AND FACE RECOGNITION BASED ONLINE EXAM SYSTEM

ABSTRACT

In this study, an identification application based on face recognition is developed to prevent students from substituting others in online exams. Principally user information consisting data's such as name, surname and face image is saved on database. During the exam the face image of the user is taken by webcam and compared by the saved image data. Students' entry to exam will be accepted or denied according to the comparison result.

Face recognition system is used in this study because almost every PC have web cam, face recognition is one of the most reliable biometric systems and also it is easy to use.

In this study, faceplate method is used for automatic face recognition and considerably high success is achieved.

Keywords: Biometric Security Systems, Face Recognition System, Online Exam System

1. GİRİŞ

1.1 Çalışmanın Konusu

Bilgisayar tabanlı sistemlerdeki gelişim, çağımızı teknoloji ve internet çağı yapmıştır. Amaç hep daha hızlı, daha güvenli, daha akıllı ve daha kullanışlı sistemler oluşturmaktır. Yirmi otuz yıl önce gerçek zamanlı olarak yapılamayan bazı uygulamalar günümüzde kolaylıkla yapılabilmektedir. Teknolojideki bu hızlı gelişim çeşitli ihtiyaçları da beraberinde getirmiştir. Bunların başında sistemlerin güvenliği ve kullanışlılığı gelmektedir.

Genellikle, uygulamalardaki gereksinimler, kullanıcı tanımlama ve bu tanımlamaya göre kullanıcıya sistemde hak verme şeklinde olmaktadır. Günümüzde, yaygın olarak kullanıcı tanımlama, numara ve manyetik kart gibi kişiye verilen bir karakter dizisi anahtar ile yapılmaktadır. Bu anahtar değerlerinin unutulabilir, kaybedilebilir veya tahmin edilebilir bir yapıda olması bu sistemlerin en büyük açığıdır.

Biyometrik sistemler bu eksiklikleri büyük ölçüde ortadan kaldıran sistemlerdir. Kişinin ölçülebilen fizyolojik veya davranışsal özelliklerine biyometri denilmektedir. Başlıca biyometrik özellikler; yüz yapısı, parmak izi, avuç içi bilgisi, retina, iris, ses, yürüyüş, el yazısı, konuşma şekli ve DNA'dır. Biyometrik sistemlerin en büyük üstünlüğü, bu özelliklerin unutulamayan, kaybedilemeyen ve tahmin edilemeyen yapılar olmasından kaynaklanır.

Yüz tanıma; güvenlik sistemlerinde, suçluların izlenmesinde ve çok gizli yerlere giriş-çıkışların denetlenmesi gibi bir çok alanda kullanılmaktadır. Bir bilgisayarın yüz tanımayı gerçekleştirebilmesi yapay zeka araştırmacılarını ilgilendiren önemli konulardan biridir.

Bilgisayarlı yüz tanımanın uzun bir geçmişi vardır. Çoğu yüz tanıma yöntemi yüzleri; göz köşeleri, ağız kenarları ve burun ucu gibi özellik noktaları arasındaki normalize edilmiş uzunluklara ve oranlara göre modeller ve sınıflandırır (Nabiyev, 2012)

Yüz tanıma problemi üç grupta incelenebilir:

- Yüz Bulma
- Yüz Normalleştirme
- Yüz Onaylama

Yüz tanıma sisteminin başarımında sistemin hızı ve yanlış tanıma oranı dikkate alınmaktadır. Yüz tanıma sisteminde karşılaşılan sorunlardan aydınlatma problemi ve pozların çok çeşitli olması öne çıkmaktadır. Işıklandırma nedeniyle yüz tanınmayabilir. Bu sorunun çözümü için ışıklandırma konileri kullanılmaktadır (Nabiyev, 2012).

Yüz tanımada özellikler bilindiğinde; yapay sinir ağları, radyal alt fonksiyon ağları, çok katmanlı perspektif, doğruluk haritaları, esnek demet graf eşlemesi, en önemli parçaların farkını bulma ve bir çok temsilci gibi yaklaşımlardan herhangi birisi kullanılabilir. Yüz tanımada en başarılı yöntemlerden birisi iki boyutlu resimlerden global özelliklerin elde edilmesidir.

Yüz tanıma algoritmaları içinde en yaygın kullanılan yaklaşım özyüz (eigenfaces) tabanlı yöntemlerdir. Yüz görüntülerinin ayırt edici karakteristik özellikleri kullanılarak mümkün olduğunca küçük boyutlu bir uzay oluşturulur ve görüntüler bu yüz uzayında karşılaştırılır. Bu uzayın özyüzler olarak adlandırılan temel vektörleri, yüzlerin bazı yerel ve global özelliklerini meydana çıkarırlar ve yüz görüntülerinin içeriğiyle ilgili bilgi verirler (Turk & Pentland, 1991).

1.2 Tezin Amacı

Günümüzde kullanılan çevrimiçi sınav sistemlerinde kimlik tespiti genellikle kişiye daha önceden verilen bir anahtar (kullanıcı adı ve şifre gibi) ile yapılmaktadır. Bu anahtarı bilen herhangi bir kişi, o anahtarın gerçek sahibiymiş gibi sınava girebilmektedir. Bu yöntemde en büyük sorun bir başkasının yerine sınava girmedir.

Biyometrik güvenlik sistemlerinden yüz tanıma sistemi araştırılmış ve bir çok kullanım alanı tespit edilmiştir. Bu sistemin, çevrimiçi sınav sistemlerinde de kimlik tespiti için kullanılabileceği örnek uygulama ile kanıtlanmıştır.

2. BİYOMETRİK GÜVENLİK SİSTEMLERİ

Biyometri (Biometric) Yunanca bio (yaşam) ve metric (ölçüm) sözcüklerinin birleşiminden meydana gelmiştir. Bireyin ölçülebilir fizyolojik veya davranışsal özelliklerine biyometri denir.

Başlıca biyometrik özellikler; yüz yapısı, parmak izi, avuç içi bilgisi, retina, iris, ses, yürüyüş, el yazısı, konuşma şekli ve DNA'dır.

Biyometrik sistemlerin en büyük üstünlüğü, bu özelliklerin unutulamayan, kaybedilemeyen ve tahmin edilemeyen yapılar olmasından kaynaklanmaktadır.

Herkeste faklı olan biyometrik özellikler, biyometriyi güvenlik sistemlerinde en çok tercih edilen bir alan haline getirmiştir. Biyometrik problemlerinden bilgisayarlı kimlik tespiti yapay zeka araştırmalarının önemli alanlarından biridir.

Biyometrik özellikleri kullanan sistemlere biyometrik sistemler denir. Bankacılık, güvenlik, kriminoloji gibi bir çok alanda biyometrik sistemler kullanılmaktadır.

Biyometrik tanımlama yöntemlerinde kullanılan güvenlik sistemleri ile anlatılmak istenen şey aslında bilgi güvenliğidir. Bir bilginin güvenliği demek, bilginin gönderilmesi gereken kişiye değiştirilmeden, bozulmadan ve başka birisinin eline geçmeden ulaşması demektir. Bir bilgi gönderilmesi gereken kişi için gizli bir bilgi değilken 3.şahıslar için gizli bir bilgidir.

2.1 Biyometrik Sistem Çeşitleri

Biyometrik sistemler fizyolojik ve davranışsal biyometrik sistemler olmak üzere iki kısma ayrılır.

2.1.1 Davranışsal biyometrik sistemler

Davranışsal biyometrik sistemler imza, yazı dinamiği, yürüyüş şekli ve konuşma esnasındaki dudak hareketleri gibi belli zamanda belli amaçlar için gerçekleştirilmiş ve herkesin birbirinden farklı olarak gerçekleştirdiği davranışlar üzerine kurulmuştur.

2.1.1.1 İmza tanıma

İmza bir kişinin adını yazma şekli olarak tanımlanabilir. Bir belgenin altına atılan imza, kişinin o belgeyi okuduğunu, yazdığını ya da onayladığını gösterir.

İmza tanıma sistemlerinde iki tür bilgi kullanmaktadır. Birinci tipte, bir desen olarak imzaya ait özellikler, ikincisinde ise imzalama süresi, hızı ve ivmesi gibi imzaya ait özellikler kullanılır (Şamlı & M.Erkal, 2009).

Bir kişinin imzasının taklit edilmesi oldukça zordur. İmza desen olarak taklit edilse bile atış şeklinin taklit edilmesi oldukça güçtür.

Şekil 2.1: İmza Tanıma Örneği

2.1.2 Fizyolojik biyometrik sistemler

Fizyolojik biyometrik sistemler yüz, parmak izi, el geometrisi, ses, iris ve retina olmak üzere, bir kişinin diğer kişiden ayrılmasını sağlayan değişmez özellikler üzerine kurulmuştur.

2.1.2.1 Parmak izi tanıma sistemi

En çok kullanılan biyometrik bilgilerden biri parmak izidir. Parmak izi tanıma genellikle parmak izinde bulunan özellik noktalarının ve bunlara ait parametrelerin karşılaştırılması esasına dayanır (Şamlı & M.Erkal, 2009).

Şekil 2.2: Parmak İzi Örneği

Parmak izi yöntemi polis tarafından kimlik saptama ve pasaport başvurularında yaygın olarak kullanılmaktadır.

Bu sistemin en önemli problemlerinden biri parmak izinin taklit edilmesidir. Bir diğeri ise organ eksikliği veya deri hastalıkları gibi sebeplerden ötürü parmak izlerinin olmamasıdır. İlk sorun parmak izi alınan kişinin o anda yaşıyor olduğunun kanıtlanması ile çözülmektedir. Diğer sorunun ise çözümü bulunmadığından, bu kişilerde bu yöntem uygulanmaz.

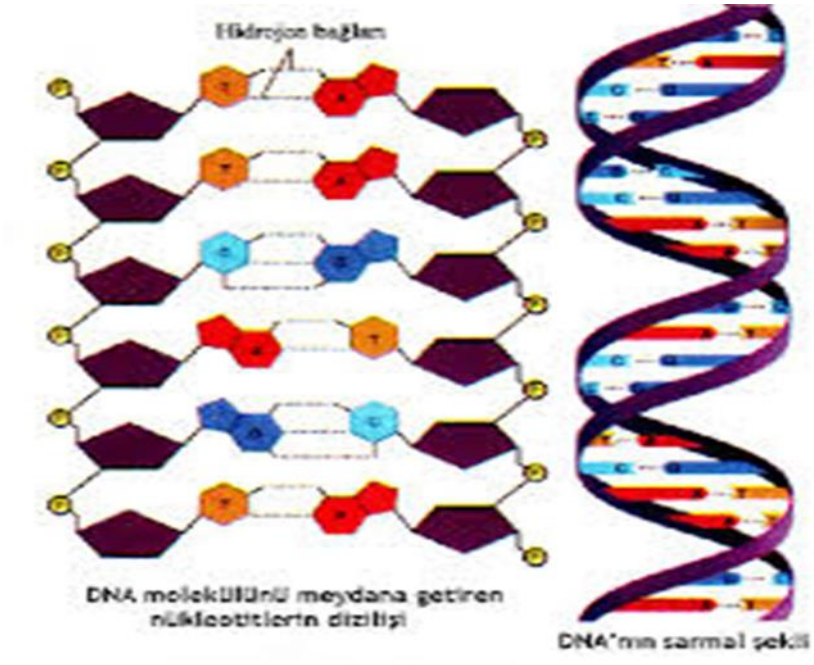
2.1.2.2 DNA tanıma sistemi

Günümüzde kullanılan en güvenilir kimlik tanıma yöntemlerinden biridir. Bireyin kan, saç gibi herhangi bir biyolojik materyali incelenir.

Bu yaklaşımda hücre nukleuslarındaki kromozomlarda saklanan DNA molekülleri kullanılmaktadır. DNA moleküllerinin dizilim eşleşmesine göre doğruluk kontrol edilir (Şamlı & M.Erkal, 2009).

Doğruluğu çok yüksek bir yaklaşım olmasına rağmen bazı dezavantajları vardır.

- Diğer kimyasal ve biyokimyasal analizlerde olduğu gibi sistemin doğruluğu örnek kalitesine bağlıdır.
- Örneklerin bozulması, karışması gibi örnek kalitesini düştüğü durumlarda sistemin başarısı düşmektedir.
- DNA analizi diğer biyometrik yöntemlere göre maliyeti yüksek bir yöntemdir.
- İşlem süresinin 24 saat gibi bir zaman gerektirmesi bu yöntemi bazı durumlarda kullanışsız yapmaktadır.



Şekil 2.3: DNA Örneği

2.1.2.3 İris tanıma sistemi

1990'ların başında geliştirilen bu sistem havaalanları gibi giriş çıkış kontrol noktalarında kullanılmaktadır. Parmak izi tanıma yönteminde 60 veya 70 karşılaştırma noktası kullanılırken iris tanıma yönteminde 200 civarı referans noktası karşılaştırılmaktadır (Şamlı & M.Erkal, 2009).

Bu yaklaşımda gözleri görmeyen, irisi bulunmayan ya da göz titremesi hastalığına sahip olan kişiler kimliklendirilemez. İris resmi alınırken gözlerin durumu, kirpiklerin ve/veya göz kapaklarının iris görüntüsünü bozması yöntemin olumsuz yönleri arasında yer alır.

s 

Şekil 2.4: İris Tanıma Örneği

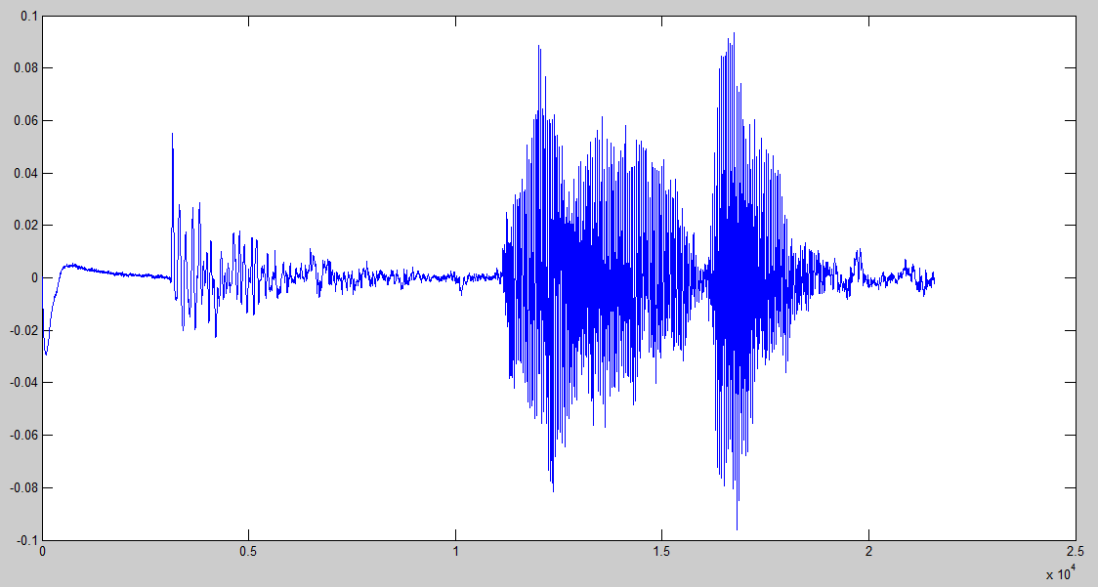
2.1.2.4 Ses tanıma sistemi

Ses tanıma sistemi ile ilgili araştırmalar 1950'lerden beri yapılmaktadır (Yalçın, Konuşma Tanıma Teorisi ve Teknikleri, 2008). Konuşma sesleri, farklı frekans değerlerine sahip sinüs dalgalarının doğrusal birleşiminden oluşur.

Ses tanıma sistemi ile konuşma sesi olarak ne söylendiği tahmin edilmeye çalışılır. Yani, ses sinyallerin ait unsurlar, ses işleme yöntemleri kullanılarak tespit edilir.

Ses tanıma sistemleri konuşma tanıma, konuşmacının tanınması, ayrık kelime tanıma, anahtar kelime bulma ve sürekli ses tanıma gibi birçok problemin çözümünde kullanılır.

Sayısal veriye dönüştürme işlemi sırasıyla örnekleme, nicelendirme ve kodlama aşamaları uygulanarak yapılır. (Yalçın, Konuşma Tanıma Teorisi ve Teknikleri, 2008)



Şekil 2.5: Ses Tanıma Örneği

2.1.2.5 Yüz tanıma sistemi

Yüz tanıma sistemi, kişilerin yüz özelliklerini göz önüne alarak yürütülen kimlik belirleme çalışmaları olarak tanımlanır.

Bu çalışmanın konusu olmasından dolayı yüz tanıma sistemi 4. bölümde detaylı olarak anlatılacaktır.

2.2 Biyometrik Güvenlik Sistemlerinin Karşılaştırılması

Çizelge 2.1'de biyometrik güvenlik sistemlerinin kullanılabilirlik özellikleri yüksek, orta ve düşük olarak sınıflandırılmıştır. Bu çizelgede; Y=Yüksek, O=Orta, D=Düşük olarak ifade edilmektedir.

Çizelge 2.1: Biyometrik Güvenlik Sistemlerinin Kullanılabilirlik Özelliklerine Göre Sınıflandırılması **(Yalçın & Gürsel, Biyometrik Güvenlik Sistemlerinin İncelenmesi, 2015)**

	Evrensellik	Eşsizlik	Süreklilik	Elde Edilebilirlik	Performans	Kabul Edilebilirlik	Yaygınlık
İmza	D	D	D	Y	D	Y	Y
Parmak izi	O	Y	Y	O	Y	O	O
DNA	Y	Y	Y	D	Y	D	D
İris	Y	Y	Y	O	Y	D	D
Ses	O	D	D	O	D	Y	Y
Yüz	Y	D	O	Y	D	Y	Y

Çizelge 2.1'de imza biyometriğinin evrensel özelliği düşük olarak görülmektedir. Organ kaybı gibi sebeplerden dolayı imza atma yeteneğinin olmaması veya okuma yazma bilmeyenlerin kendilerine ait bir imza biçiminin bulunmaması nedeniyle bu özellik düşüktür. DNA, iris ve yüz biyometriklerinin sahip oldukları özellikler yönünden evrensellikleri yüksek olarak sınıflandırılmıştır

Her insanda farklı olması yani eşsizlik özelliği yönünden baktığımızda DNA, parmak izi ve iris biyometrikleri aynı yumurta ikizlerinde bile farklı olduğundan yüksek olarak sınıflandırılmıştır.

Yaşam boyunca değişmeyen yani süreklilik özelliği bakımından değerlendirildiğinde DNA, parmak izi ve iris biyometrikleri yüksek olarak sınıflandırılmıştır. İmza ve ses biyometrikleri ise neşe, heyecan, korku ve soğuk hava gibi değişimlere bağlı olarak değiştikleri için düşük olarak sınıflandırılmışlardır.

Gözün narin yapısı ve kullanılan cihazın verdiği rahatsızlık gibi nedenlerden dolayı iris biyometriğinin elde edilebilirliği düşüktür. Yüz ve imza biyometriklerinin elde edilebilirliği ise yükseltir.

Doğruluk ve hız yönünden DNA, parmak izi ve iris biyometriklerinin performansı yüksektir.

Biyometrik verilerin ölçüm ve toplanması yönünden yani kabul edilebilirlik bakımından yüz, ses ve imza biyometrikleri yüksek sınıflandırılmıştır.

Son olarak yüz, imza ve ses biyometrikleri yaygınlı bakımından yüksel olarak sınıflandırılmıştır.

.

3. ÇEVRİMİÇİ SINAV SİSTEMLERİ

Çevrimiçi eğitim sistemlerinde ölçme ve değerlendirme, genellikle, çevrimiçi sınav sistemleri ile yapılmaktadır. Sınav soruları çoktan tek seçmeli, çoktan çok çekmeli, doğru-yanlış, kısa cevap, boşluk doldurma, numaralı vb. türünde olmaktadır.

Sınav soruları ve cevapları ilgili öğretmen veya sistem görevlisi tarafından soru bankasına girilir. Bu sorular ve cevapları web sunucudaki bir veri tabanında tutulur.

Daha önceden belirlenen gün ve saatte, sınav sistemine başarıyla giriş yapan öğrencilerin ekranına, soru bankasından rastgele seçilen sorular gelir. Soru bankasından her öğrenciye eşit sayıda, aynı zorluk seviyesinde, rastgele sorular seçilir. Öğrencinin ekranına sorular ve cevap seçenekleri rasgele sıralamada çıkartılır. Böylece, birbirine bakarak kopya çekmenin önüne geçilmeye çalışılır. Bu bağlamda soru bankasında ne kadar çok soru varsa, birbirine bakarak kopya çekme o kadar zor olur.

Öğrencilerin sınav süresi içinde cevapladığı sorular, sınav uygulaması tarafından sunucudaki veri tabanına kaydedilir.

Sınav sistemine giriş için öğrenciler kendilerine daha önceden verilen bir kullanıcı adı ve şifre kullanırlar. Bir kişi diğer kişinin kullanıcı adı ve şifresini biliyorsa onun yerine sınava girebilir. Bu çalışmada, bu sorunun çözümü için otomatik kimlik tanımlama yöntemlerinden yüz tanıma sistemi araştırılmıştır

4. YÜZ TANIMA YÖNTEMLERİ

Bilgisayarda yüz tanıma sistemiyle ilgili araştırmaların uzun bir geçmişi bulunmaktadır (Bledsoe, 1964; Goldstein, Harmon, & Lesk, 1971). Başlıca biyometrik teknolojilerden biri olan yüz tanıma, görüntü yakalama cihazlarındaki hızlı gelişmeler ve yüksek güvenliğe olan ihtiyaçlar sonucunda, giderek daha önemli bir hale gelmiştir (Varol & Cebe, 2011).

Senelerce görmediğimiz insanları tanıyoruz. Acaba, bunlar beyinde nasıl tutuluyor? Tanıma işlemi nasıl gerçekleştiriliyor?

Görme olayında, nesnelerin ne olduklarının belirlenmesine görüntü tanıma denir. İnsan beyninde her bölgenin kendi görevi olduğu bilinmektedir. Görme beyin zarındaki işbölümü, yerleştirme ve tanıma arasındaki bölünmeyle sona ermez. Tanımada kullanılan renk, biçim ve doku gibi bilgi türlerinin, beyin zarının tanıma dalının, farklı alt bölgelerinde incelendiği görülür (Nabiyev, 2012). Bilgi, ilksel alıcı alandan görme zarının öteki alanlarına doğru hareket ederken uzmanlaşma daha belirgin hale gelir. Bölgelerden biri sadece biçime, diğer bir bölge sadece renge, bir başka bölge ise sadece harekete ayrılır (Nabiyev, 2012).

Bir nesneyi tanımak, onu bir sınıflandırma içerisinde değerlendirmek anlamına gelir. Tanıma yoluyla nesnenin bir çok özelliğini öğreniriz. Nesnelerin renk, doku, büyüklük ve biçim gibi bir çok özellikleri vardır. Ama bunlardan tanımada önemli olan nesnenin biçimi ile bileşenleri arasındaki ilişkilerdir. (Nabiyev, 2012) Örneğin, biz bir balığı büyüklüğüne ve rengine bakmadan biçimine göre kolayca tanımaktayız. Bir çok nesneyi, sadece onların biçimini gösteren basit çizgilerden tanımamız, tanımanın biçim üzerinden gerçekleştiğini göstermektedir.

Bir nesneyi tanımak için onu biçimine uygun sınıfa nasıl koyacağız. Örneğin, insan bir harfi nasıl tanır. Önce, algısal sistem nesneyi çizgiler. Açılar ve kanarlar gibi bileşenlere göre tanımlamak için retina üzerindeki bilgileri kullanır. Sistem bu bileşenleri nesnenin kendisine ilişkin bir tasvirini çizmek için kullanır. Daha sonra, nesnenin bu tasvirini, görsel hafıza depolanmış nesne türlerinin biçim tanımlarıyla karşılaştırır ve en uygun olanı seçer (Nabiyev, 2012). Örneğin, bir Z harfini tanımak, bu harfin biçiminin en çok Z'ye uygun olduğunu söylemektir.

Bir insanın yüzü kendine hastır. Dolayısıyla, kimliğinin tespitinde bu karakteristik özellikleri kullanılabilir. Yüz tanıma teknolojisinin çalışma mantığı yüz görüntülerinden karakteristik özelliklerin çıkarılmasına dayanır. Sonrasında ise, iki görüntü karşılaştırılır. (Varlık & Çorumluoğlu, 2011)

Yüz tanıma süreci yüz görüntülerinden çeşitli özelliklerin çıkarılmasıyla başlar. Aynı kişinin yüz görüntülerinin özellikleri arasında bir benzerlik olurken, farklı kişilerin yüz görüntülerinin özellikleri arasında farklılık olur. Bütün bunlar dikkate alındığında yüz tanıma sistemleri iki kısıma ayrılabilir.

1. Özellik tabanlı yüz tanıma sistemleri: Geometrik özellik tabanlı, şablon eşlemeli yöntem, yapısal eşleme, gizli Markov modeli, dalgacık dönüşüm ve elastik demet grafi eşleme gibi yüz tanıma yöntemleri vardır.

2. Görünüm tabanlı yüz tanıma sistemleri: Özyüz yaklaşımı (Turk & Pentland, 1991), Fisher yaklaşımı (Belhumeur, Hespanha, & Kriegman, 1997) ve yapay sinir ağı gibi yüz tanıma yöntemleri vardır.

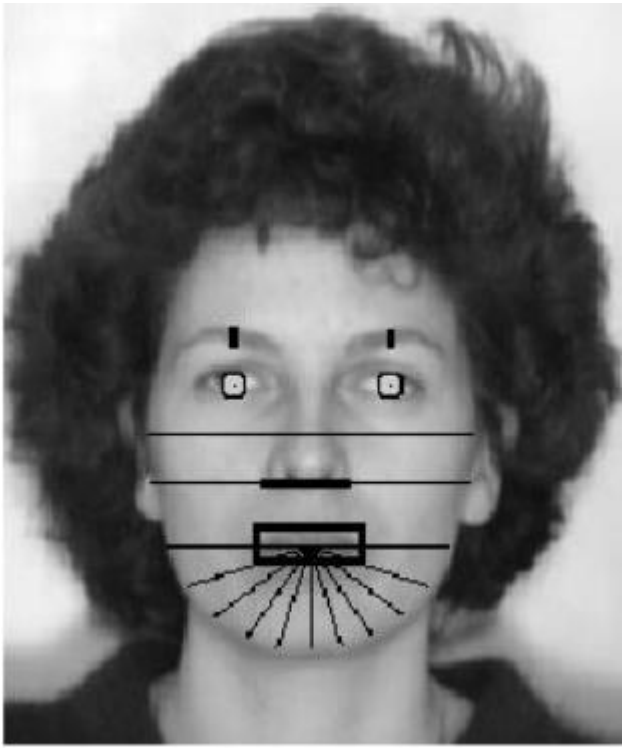
4.1 Geometrik Özellik Tabanlı Yöntem

İlk yüz tanıma yöntemi olan geometrik tabanlı yüz tanıma sistemi Goldstein tarafından 1964 yılında kullanılmıştır. Yüzde belirleyici özelliğe sahip bölgelerin boyutlarının birbirleri ile arasındaki uzaklıkların ve geometrisinin hesaplanması ile yapılan tanımlama yöntemidir.

Kaya ve Kobayashi (1972) yüz üzerindeki belirli noktaların belirlenmesinde birtakım önemli etmenler belirlemiş ve bu noktalar arasındaki öklit uzaklığını hesaplamışlardır (Brunelli & Poggio, 1992).

1. Kolay tahmini edilebilmeli
2. Işığa az duyarlı olmalı
3. Mimikten mümkün olduğunca bağımsız olmalı
4. Ayırt edici özelliğe sahip bilgi mümkün olduğunca çok olmalı

Yüz üzerindeki tanımlayıcı özelliğe sahip noktalara örnek olarak göz, ağız, kaş, çene ve burun örnek olarak verilebilir.



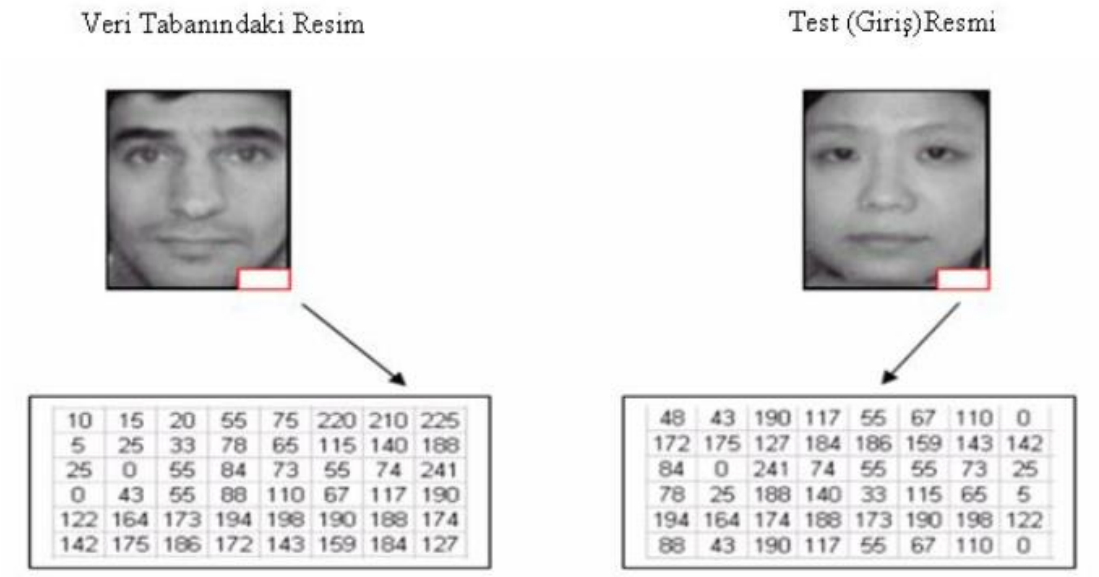
Şekil 4.1:Yüz Tanımada Kullanılabilecek 22 Geometrik Nokta **(Brunelli & Poggio, 1992)**

Geometrik özellik tabanlı yüz tanıma yöntemleri, daha önceden tanımlanmış kaş kalınlığı ve dikey konumu, burun kalınlığı ve konumu gibi ayırdedici özellikleri kullanır. Bulunan değerler özellik vektörü olarak kaydedilir. Brunelli ve Poggio 1992 yılında, yüzün ayırdedici özellikleri olarak gördükleri 22 geometik noktayı Şekil 4.1'de görüldüğü gibi belirlemişlerdir.

Bu geometrik noktalardan elde edilen 22 tane değer en yakın komşuluk algoritması kullanılarak karşılaştırılır ve en küçük değere sahip veri tabanı resmi ile eşleştirilir. (Brunelli & Poggio, 1992)

4.2 Şablon Eşlemeli Yöntem

Bu yöntemde, iki farklı desenin tümü veya belirli özeliklerdeki parça şablonları en yakın komşuluk algoritması ile karşılaştırılır (Brunelli & Poggio, 1993).



Şekil 4.2: Şablon Eşlemeyle Yüz Tanıma **(Brunelli & Poggio, 1995)**

Şablon eşlemeli yöntemde görüntüler gri renk düzlemindedir. Eşleme algoritması, eğitim ve test veri kümeleri arasındaki farklıkları ve benzerlikleri hesaplar. Eşleme, görüntünün belirli kısmlarına veya bütün görüntüye uygulanmalıdır. Şablonlar, görüntüler arasındaki benzerliklerin en çok olduğu veya farklılıkların en çok olduğu bölümlerden seçilir (Brunelli & Poggio, 1995). Şablon desenleri, eğitim verilerinden alınan referans desenleri olarak kabul edilir. Bir test resmi, bu referans desenleri kullanılarak tanımlanmaya çalışılır. Sonuçta, eğitim ve test desenleri arasındaki benzerlikler ve farklılıklar hesaplanır (Brunelli & Poggio, 1993).

4.3 Yapay Sinir Ağı Yöntemi

Yapay sinir ağları insan beyninin bilgi işleme tekniğinden esinlenerek geliştirilmiş bir teknolojidir. Öğrenme yetisine sahip olan bu yöntem, kişinin yüzünü tanımak için kullanılır.

Yapay sinir ağları bilgi işleme gücünü paralel dağılmış yapısından, öğrenebilme ve genelleme yapabilme yeteneğinden almıştır. Genelleme yeteneği ile YSA'nın öğrenme sürecinde karşılaşmadığı girişlere uygun tepkiler vermesi anlatılmak istenmektedir. YSA, aşağıda verilen özellikleri nedeniyle yüz tanıma da olmak üzere bir çok alanda kullanılmaktadır.

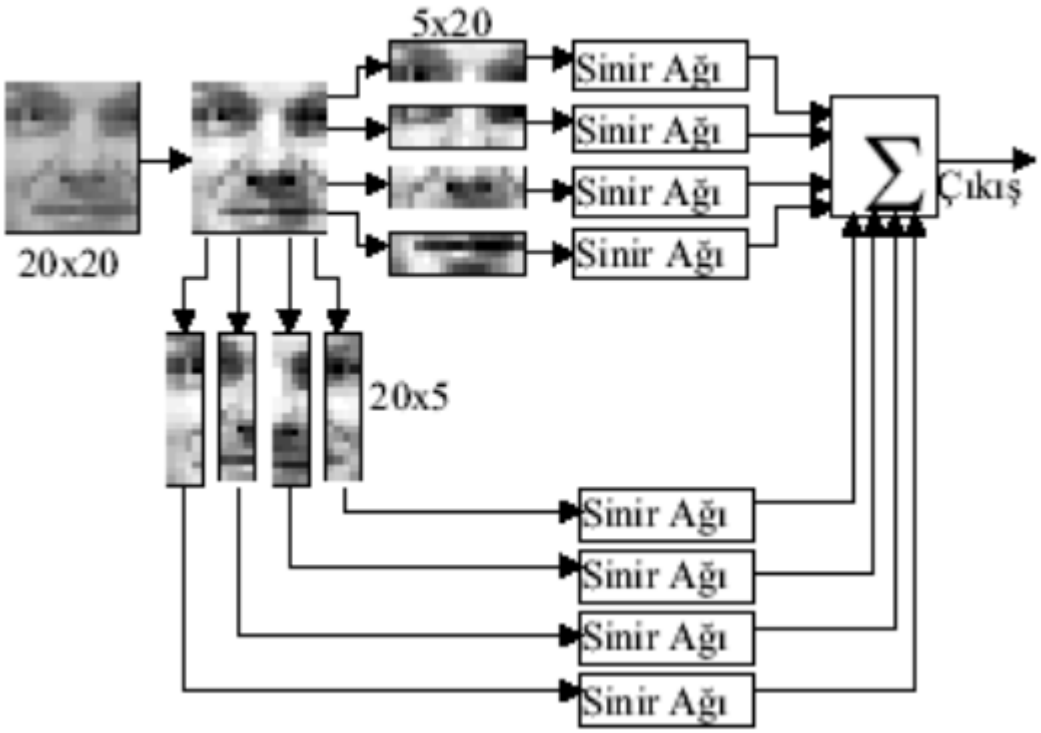
Doğrusal Olmama: YSA'nın temel işlem elemanı olan hücre doğrusal değildir. Bu nedenle hücrelerin birleşmesinden oluşan YSA'da doğrusal değildir. Bu özelliği nedeniyle doğrusal olmayan problemlerin çözümünde etkili bir araç olmuştur.

Öğrenme: YSA'nın istenen çıktıyı verebilmesi için uygun bir şekilde ayarlanması gerekir. Buda, hücreler arasında bağlantıların doğru yapılmasını ve bağlantıların uygun ağırlıklara sahip olmasını gerektirir. YSA'nın yapısı nedeniyle bağlantılar ve ağırlıkları önceden ayarlanamaz. Bu nedenle YSA, ilgilendiği problemin eğitim örneklerini kullanarak bu bağlantılar ve ağırlıkları hesaplamalıdır.

Genelleme: YSA, bir problemi öğrendikten sonra eğitim esnasında karşılaşmadığı test verileri içinde istenen çıktıyı verebilmelidir.

Uyarlanabilirlik: YSA, problemdeki değişikliklere göre ağırlıklarını ayarlar. YSA'nın ilgilendiği problemde bir değişim olduğunda YSA bu değişimlere göre kendini tekrar eğitebilir.

Hata Toleransı: YSA, bir çok hücrenin çeşitli biçimlerde bağlanmasından meydana geldiği için paralel dağılmış bir yapıya sahiptir. Ağın sahip olduğu bilgi, ağdaki tüm bağlantılara dağıtılır. Bundan dolayı, YSA'nın bazı hücrelerinin veya bağlantılarının etkisiz hale gelmesi ağın doğru çıktı üretmesini pek etkilemez.



Şekil 4.3: Yapay Sinir Ağı Eğitimi **(Erdoğan, 2010)**

Şekil 4.3'te görüldüğü gibi, yüz görüntüleri 20x20 piksel boyutlarında sisteme girilmiştir. Her görüntü 5x20 piksel boyutlarında 4 tane satır matrisi ile 20x5 piksel boyutlarında 4 tane kolon matrisine dönüştürülür. Bu alt matrisler sinir ağına girer ve ağırlık katsayıları çarpımlarının çıkışları toplanarak 0 ile 1 arasında bir çıkış elde edilir. Son katmanda çıkışlar toplanarak sonuç çıkış elde edilir. Bu değerin eşit değerini aşıp aşmadığına bakılarak yüz görüntüsünün sahibi bulunur (Erdoğan, 2010).

5. ÖZYÜZ ALGORİTMASI

Gerçek hayatta, bir görüntü iki boyutlu bir f(x,y) fonksiyonu olarak tanımlanır. Burada x ve y düzlemsel koordinatlarıdır. f fonksiyonunun herhangi bir (x,y) koordinatındaki genliği görüntünün o noktadaki yeğinlik (örneğin parlaklık gibi) veya gri seviyesi olarak adlandırılır. x, y, f'in yeğinlik değerleri hep birlikte sonlu ve ayrık büyüklükte olduğunda bu görüntüye sayısal görüntü denir. (Gonzalez & Woods, 2014)

Sayısal görüntü sonlu sayıda bileşenden oluşur. Her bir bileşen belli bir görüntünün değeridir. Bu bileşenlere resim elemanı, görüntü elemanı veya piksel denir.

Yüz tanıma algoritmaları içinde en yaygın kullanılan yaklaşım özyüz (eigenfaces) tabanlı yöntemlerdir. Yüz görüntülerinin ayırt edici karakteristik özellikleri kullanılarak mümkün olduğunca küçük boyutlu bir uzay oluşturulur ve görüntüler bu yüz uzayında karşılaştırılır. Bu uzayın özyüzler olarak adlandırılan temel vektörleri, yüzlerin bazı yerel ve global özelliklerini meydana çıkarırlar ve yüz görüntülerinin içeriğiyle ilgili bilgi verirler (Turk & Pentland, 1991).

Özyüz algoritması (Temel bileşenler analizi) boyut azaltmak ve görüntüler arasındaki temel farklılıkları bulmak için kullanılan bir yöntemdir (Turk & Pentland, 1991).

Temel bileşen analizi sınıf bilgisini kullanmadığı için gözetimsiz bir öğrenme yöntemidir. Ölçüt, verideki varyanstır. Veri noktaları arasındaki farkın en iyi ortaya çıktığı, yani varyansın en yüksek olduğu vektör temel bileşeni oluşturur (Alpaydın, 2013, s. 90-92).

Özyüz algoritmasında her biri N x N boyutunda olan görüntüler N^2 uzunluğunda vektörlere dönüştürülür.

		$\begin{bmatrix} I1_{11} \dots\dots\dots\dots\dots & I1_{1N} \\ I1_{21} \dots\dots\dots\dots\dots & I1_{2N} \\ . & . \\ . & . \\ . & . \\ I1_{N1} & I1_{NN} \end{bmatrix}$		$\begin{bmatrix} I1_{11} \\ . \\ I1_{1N} \\ I1_{21} \\ . \\ I1_{2N} \\ . \\ I1_{N1} \\ . \\ I1_{NN} \end{bmatrix}$

Şekil 5.1: N x N tipinde görüntü örneği ve dönüştürülmüş hali

Şekil 6.1'de I veri tabanındaki yüz görüntülerini N ise piksel sayısını göstermektedir.

Daha sonra, veri tabanındaki tüm görüntülerden ortalama görüntü vektörü elde edilmektedir.

$$\Psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n \qquad \textbf{(6.1)}$$

Yukarıdaki denklemde, Ψ ortalama görüntü vektörü, Γ görüntü vektörü, M'de toplam görüntü sayısıdır. AT&T yüz veri tabanında bulunan 40 kişiye ait 10 farklı yüz görüntüsünden elde edilen ortalama yüz görüntüsü Şekil 6.2'de verilmiştir.

	$= \frac{1}{M}$	$\begin{bmatrix} I1_1 + \dots\dots\dots + & IM_1 \\ I1_2 + \dots\dots\dots + & IM_2 \\ . & . \\ . & . \\ . & . \\ I1_{N^2} + \cdots\dots\dots\dots + & I1_{N^2} \end{bmatrix}$

Şekil 5.2: Ortalama görüntünün hesaplanması

Şekil 6.2'de I yüz veri tabanındaki görüntüleri, N piksel sayısını, M ise toplam görüntü sayısını belirtmektedir. Sütun vektörüne dönüştürülen görüntülere ait pikseller toplanıp, toplam görüntü sayısına bölünür. Eş. 6.1'de verilen ortalama görüntüyü yeniden elde etmek için kolon matrisi NxN'lik matrise çevrilir.

Ortalama görüntü vektörünün her bir görüntü vektöründen çıkarılmasıyla sıfır ortalamaya sahip olan bir veri tabanı matrisi (A) elde edilir (Eş. 6.2 ve Eş. 6.3).

$$\theta_i = \Gamma_i - \Psi \quad \textbf{(6.2)}$$

Eş. 6.2'de θ ortalama görüntü vektöründen çıkarılmış görüntü vektörünü, i=1, 2, 3, … , M ise görüntü sayısını göstermektedir.

$$A = [\theta_1\ \theta_2\ \theta_1 \dots \theta_M] \quad \textbf{(6.3)}$$

Eş. 6.3'te A: $N^2\ x\ M$ boyutundaki görüntü veri tabanı matrisini ifade eder.

$$\vec{I}1_m = \begin{bmatrix} I1_1 - m_1 \\ I1_2 - m_2 \\ . \\ . \\ . \\ I1_{N^2} - m_{N^2} \end{bmatrix} \dots \vec{I}M_m = \begin{bmatrix} IM_1 - m_1 \\ IM_2 - m_2 \\ . \\ . \\ . \\ IM_{N^2} - m_{N^2} \end{bmatrix}, \quad A = \begin{bmatrix} & & \\ n & \dots & \vec{I}M_m \end{bmatrix}$$

Şekil 5.3: Ortalama merkezli görüntüler matrisi

Veri tabanındaki görüntülerden öz yüzlerin elde edilmesi için A matrisinin kovaryans matrisi oluşturulmaktadır (Eş. 6.4).

$$\mathrm{C} = \frac{1}{M}\sum_{n=1}^{M} \theta_n {\theta_n}^T = AA^T \quad \textbf{(6.4)}$$

Eş. 6.4'te C: kovaryans matrisini ifade eder.

Eş. 6.4'le elde edilen kovaryans matrisinin boyutu, görüntü vektörünün uzunluğu ile ilişkilidir. N^2'lik görüntü vektörlerinden elde edilecek olan kovaryans matrisinin boyutu N^2 x N^2'dir. Görüntü vektör boyutunun, toplam görüntü sayısından büyük olduğu durumlarda (N^2 >> M) bu büyüklükteki bir matrisin elde edilmesi, özdeğer ve özvektörlerinin oluşturulması oldukça güçtür. N^2 x N^2'lik kovaryans matrisinden elde edilecek olan N^2 adet özvektörden sadece M-1 tanesi anlamlı olacaktır. Anlamlı olan özvektör sayısı veri tabanındaki görüntü sayısı ile ilişkilidir.

Oluşturulan diğer özvektörler, sıfır değerine sahip özdeğerlere karşılık gelir. Bundan dolayı, N^2 x N^2'lik bir matrisin çözümlenmesi yerine M x M boyutundaki bir matrisin çözümlenmesi daha etkin olacaktır.

Kovaryans matrisinin daha küçük boyutta elde edilebilmesi için Eş. 6.4 yerine Eş. 6.5 kullanılabilir (Turk & Pentland, 1991).

$$\mathrm{C} = \frac{1}{M}\sum_{n=1}^{M} {\theta_n}^T \theta_n = A^T A \qquad \textbf{(6.5)}$$

Eş. 6.5 ile elde edilen kovaryans matrisinin özdeğer ve özvektörlerinden, Eş. 6.4'ün kovaryans matrisinin özdeğer ve özvektörlerine basit bir matris uzayı değişimi ile ulaşılabilmektedir. Eş. 6.5'deki kovaryans matrisinin özvektörleri v_i olarak gösterilerek,

$$A^T A v_i = \mu_i v_i \qquad \textbf{(6.6)}$$

Eş. 6.5'te μ_i: özdeğerleri, v_i'de özvektörleri ifade etmektedir.

Eş. 6.6'daki eşitliğin her iki tarafı A ile çarpılarak Eş. 6.7 elde edilmektedir.

$$AA^T A v_i = \mu_i A v_i \qquad \textbf{(6.7)}$$

Eş. 6.7'den AA^T matrisinin özvektörleri Av_i olarak bulunur.

İstenen özvektörlere ulaşmak için elde dilen özvektörler A matrisi ile çarpılır. Böylece, işlem yükü N^2 x N^2 boyutunda eşdeğişinti matrisi elde etmekten M x M boyutunda eşdeğişinti matrisi elde etmeye düşürülür. Ortaya çıkan M tane özvektörden M' tanesi uzay enerjisini en iyi görüntüleyecek şekilde özyüz değerlerine göre saklanmaktadır.

Yaratılan özvektörlere özyüz, özyüzlerin oluşturduğu matrise ise özyüz uzayı denilmektedir. Tanımlamanın yapılabilmesi için, her bir görüntünün, görüntü uzayındaki izdüşümünün bulunması ve ağırlık vektörünün çıkarılması gerekmektedir.

$$\omega_k = {V_k}^T \mathrm{A} \qquad \textbf{(6.8)}$$

Eş.5.8'de ω: 1 X M boyutunda ağırlık vektörünü, k=1, 2, 3, … , M' özvektörleri ifade eder.

Tanımlama için kullanılacak olan Q matrisinde, veri tabanında saklanacak olan görüntülerin ağırlık vektörleri yer almaktadır.

$$\Omega = [\omega_1{}^t \; \omega_2{}^t \; \ldots \; \omega_{M'}{}^t] \qquad \textbf{(6.9)}$$

Eş 6.9'da Ω: (MxM') boyutunda ağırlık matrisidir

Sisteme bilinmeyen bir yüz görüntüsü girildiğinde, eğitim seti için yapılan işlemler bu kez tek bir görüntü için yapılır. Girilen yüz görüntüsü $N^2 \; x \; 1$ uzunluğunda vektöre dönüştürülür.

$$= \begin{bmatrix} T_{11} \ldots\ldots\ldots\ldots\ldots & T_{1N} \\ T_{21} \ldots\ldots\ldots\ldots\ldots & T_{2N} \\ . & . \\ . & . \\ . & . \\ T_{N1} & T_{NN} \end{bmatrix} \quad t= \begin{bmatrix} T_{11} \\ . \\ T_{1N} \\ T_{21} \\ . \\ T_{2N} \\ . \\ T_{N1} \\ . \\ T_{NN} \end{bmatrix}$$

Şekil 5.4: Bilinmeyen yüz görüntüsü ve sütun vektörüne dönüştürülmesi

Şekil 6.5'te T test görüntüsüdür.

Bilinmeyen görüntü vektöründen, eğitim setinde bulunan ortalama görüntü vektörü çıkartılarak ortalama bilinmeyen görüntü vektörü elde edilir.

$$\vec{t}_m = \begin{bmatrix} t_1 - m_1 \\ t_2 - m_2 \\ . \\ . \\ . \\ t_{N^2} - m_{N^2} \end{bmatrix}$$

Şekil 5.5: Bilinmeyen Görüntünün Ortalama Görüntüden Farkı

$$\omega_k = V_k{}^T \theta_T = V_k{}^T (\Gamma_T - \Psi) \quad \textbf{(6.10)}$$

Eş 6.10'da ω: her bir yüz görüntüsüne ait ağırlıkları simgeler, k=1, 2, 3, … , M yüz görüntüsü sayısıdır.

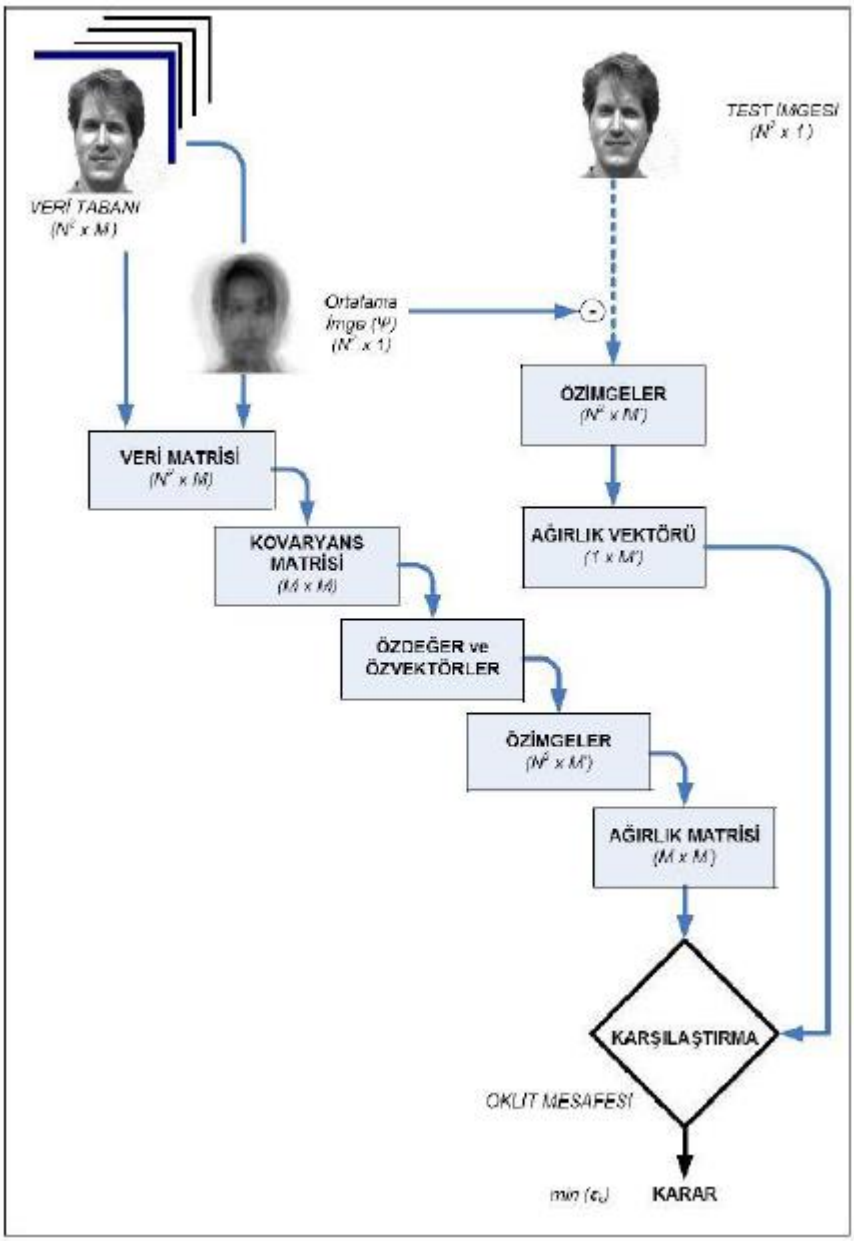
$$\theta_T = [\omega_1\ \omega_2\ \omega_3 \ldots\ \omega_M] \quad \textbf{(6.11)}$$

Eş 6.11'de θ_T: 1 x M boyutunda test görüntüsüne ait ağırlık vektörüdür.

Yüz görüntüsü uzayına düşürülen her bir yüz görüntüsünün, ağırlık matrisinden öklid uzaklığına bakılarak tanımlama gerçekleştirilmektedir.

$$\varepsilon_k^2 = P(\theta - \theta_K)P^2 \quad \textbf{(6.12)}$$

Eş 6.12 ile ağırlık matrisine göre, test yüz görüntüsünün öklid uzaklığı bulunmaktadır. Bu mesafenin, ağırlık matrisi ile yüz görüntüleri arasındaki en küçük öklid mesafesi olduğu kabul edilerek tanımlama geçekleştirilir. Özyüz algoritmasının işelevsel blok şeması Şekil 5.6'da verilmiştir (Erdoğan, 2010).



Şekil 5.6: Özyüz Algoritması İle Sınıflandırma İşlevsel Blok Şeması

6. YÖNTEM

6.1 Yüz Tanıma Teknolojisiyle Çevrimiçi Sınav Giriş Projesi

Bu çalışmada yüz tanıma teknolojisi kullanılarak güvenli çevrimiçi sınav uygulaması geliştirilmiştir.

Merkezi bir veri tabanı ve yüz tanıma sistemi ile otomatik olarak sınav girişinin kontrol edilmesi ve işletilmesi gerçekleştirilmiştir. Her bilgisayarda bulunan ve donanım olarak kullanılan web kamera ile kişilerin görüntüleri alınmakta, sonrasında ise bu görüntü ile veri tabanındaki yüz görüntüleri karşılaştırılarak sınav giriş kontrolü yapılmaktadır. Böylece, klasik olarak yapılan kimlik tespit yöntemlerine alternatif olarak çevrimiçi çalışan, kontrolü yapılabilen, otomatik olarak kimlik tespiti yapabilen ve yüz görüntülerinden kimlik oluşturabilen öğrenci tanıma uygulaması geliştirilmiştir.

Bu projede, kimlik tespiti internet üzerinden veya yerel ağ üzerinden yapıldığından zamandan tasarruf sağlayacaktır. Ayrıca, çok fazla öğrencinin bulunduğu sınavlarda, gözetmenler tarafından tanınmayan öğrenciler, sistem tarafından otomatik olarak tanınacağından sınav güvenliği de artacaktır.

Biyometrik tanımlama yoluyla yüz görüntülerinden oluşturulan kimlik bilgisi sisteme eklenecek ve bir merkezi yönetim ile öğrenci girişleri otomatik olarak yapılacaktır. Veri tabanında kimlik bilgisi olmayan kişilere sınav ekranı açılmayacak, böylece, bir başkasının yerine sınava girme şeklinden yapılan sınav hileleri ortadan kalkacaktır.

6.2 Yüz Tanıma İle Sınav Girişi Uygulamasında Kullanılan Yazılım Yapısı

Bu projede, Java programlama dili kullanılarak, sınav sisteminde tanımlanan kullanıcı bilgileri ile web kameradan gelen yüz görüntüsü bilgilerini yöneten bir sistem geliştirilmiştir. Görüntü işleme kütüphaneleri olarak OpenCV ve JavaCV kullanılmıştır.

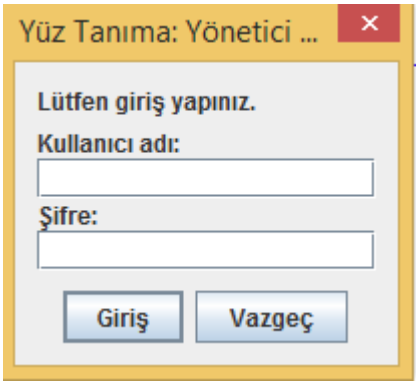
OpenCV Intel tarafından geliştirilerek BSD lisansı ile lisanslanmış, "Bilgisayarla Görü/ Görme" kütüphanesidir. Özellikle gerçek zamanlı uygulamalar hedef alınarak geliştirilmiş olması, ticari kullanımı dahil ücretsiz olması ve Windows, Linux, MacOS X gibi farklı platformlarda kullanılabilmesi bu kütüphaneyi diğer görüntü işleme araçlarından bir adım öne çıkarmaktadır (Erişti, 2010).

Dematrasyonda gösterdiğimiz sınav giriş ekranı Java programlama dili kullanılarak geliştirilmiştir. Bu ekranda yapılan kimlik tespitinden başarıyla geçen kullanıcılar PHP ile geliştirilen sınav uygulamasına yönlendirilirler. Veri tabanı olarak MySql sunucu kullanılmıştır.

Proje, yönetici ve kullanıcı ekranlarından oluşur.

6.3 Yönetici Giriş Arayüzü

Yöneticisi yetkisine sahip olan kullanıcıların sisteme giriş yaptığı ekrandır.



Şekil 6.1: Yönetici Giriş Arayüzü Görüntüsü

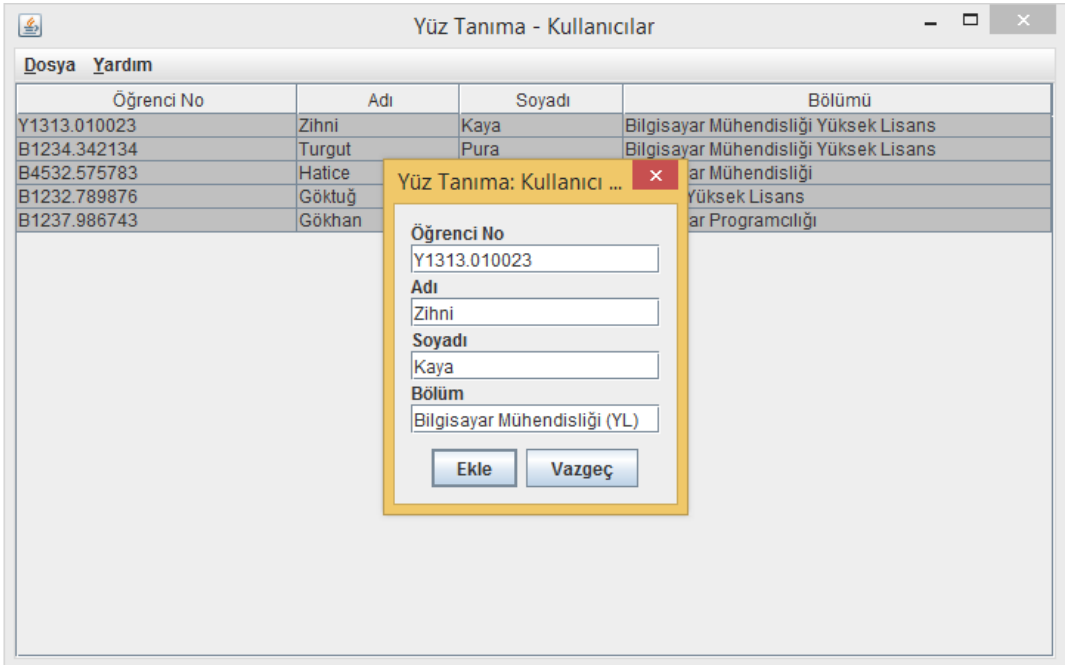
6.4 Yönetici Ana Ekranı Arayüzü

Kullanıcı ekleme, düzenleme, silme ve yüz görüntüsü ekleme gibi işlemlerin yapıldığı yönetici ana ekranıdır. Bu ekrana ilk giriş yapıldığında sistem veri tabanında kayıtlı olan tüm öğrenciler listelenir. Yönetici ana ekranı Şekil 7.2'de görülmektedir.

Şekil 6.2: Yönetici Ana Ekranı Arayüzü

6.5 Yeni Kullanıcı Ekleme Arayüzü

Sisteme yeni kullanıcı eklemek için kullanılan programın arayüzü Şekil 7.3'te görünmektedir. Buradaki bilgiler doldurularak kullanıcı sisteme eklenir.



Şekil 6.3: Kullanıcı Ekleme Arayüzü Örneği

6.6 Kullanıcı Yüz Bulma Arayüzü

Sistemde yer alan bir kullanıcı için yüz görüntüsü bulma arayüzü Şekil 7.4'te görülmektedir.

Şekil 6.4: Kullanıcı Yüz Görüntüsü Bulma Arayüzü

Bu arayüze giriş yapıldığında web kamera çalıştırılmakta, yakalanan görüntüdeki yüz bulunarak yeşil kenarlı dikdörtgen içine alınmaktadır. Bu işlemleri yapan kodlar aşağıda verilmiştir.

```
private static void yuzBul(Mat image) {
      MatOfRect faceDetections = new MatOfRect();
      faceDetector.detectMultiScale( image, faceDetections, 1.1,
7,0, new Size(150,40), new Size());
      Rect rectCrop=null;

      for (Rect rect : faceDetections.toArray()) {
          Imgproc.rectangle(image, new Point(rect.x, rect.y), new
Point(rect.x + rect.width, rect.y + rect.height), new Scalar(0, 255,
0));
          rectCrop = new Rect(rect.x, rect.y, rect.width,
rect.height);
      }

      if(rectCrop != null){
        yuzMat = new Mat(image,rectCrop);
        yuzMatResize = new Mat();
```

```
Size boyutlar = new Size(100,100);
Imgproc.resize( yuzMat, yuzMatResize, boyutlar);
                    yuzBufResize =
imageProcessor.toBufferedImage(yuzMatResize);

yuzMatGray = new Mat();
Imgproc.cvtColor(yuzMatResize, yuzMatGray, Imgproc.COLOR_RGB2GRAY);

            MatOfByte matOfByte = new MatOfByte();
            Imgcodecs.imencode(".jpg", yuzMatGray, matOfByte);
            matOfByteArr = matOfByte.toArray();
          }

}
```

"Kaydet" düğmesine tıklandığında, bulunan yüz görüntüsü 100x100 piksel boyutlarında görüntüye dönüştürülür. Sonrasında, RGB renk uzayındaki görüntü Gri seviye görüntüye çevrilir. Son olarak, JPEG görüntü formatında veri tabanına kaydedilir. Kaydetme işlemini gerçekleştiren kodlar:

```
private static void yuzKaydet(){
        Yuz yuz = new Yuz();
        yuz.goruntuVer(matOfByteArr);

        yuzDAO.ekle(yuz);
        yuzTableModel.yenile();
        yuzTableModel.fireTableDataChanged();

}
```

Ekranın sağ tarafında, ilgili kullanıcının veri tabanında bulunan tüm yüz görüntüleri listelenmektedir. Buradaki yüz görüntüleri kullanılarak kişinin kimlik bilgileri oluşturulur. Yüz tanımada kullanılmak istenmeyen yüz görüntüleri "Sil" düğmesine tıklanarak veri tabanından silinir.

6.7 Kullanıcı Giriş Arayüzü

Sisteme daha önceden kayıt olmuş öğrencilerin sınava girmek için kullandıkları arayüzdür.

Kullanıcı giriş arayüzü görüntüsü Şekil 7.5'te görülmektedir. Bu arayüz ile web kameradan gelen görüntüler alınmakta ve bu görüntülerdeki yüz bulunmaktadır.

Şekil 6.5: Kullanıcı Kimlik Tespiti Arayüzü Görüntüsü

Kullanıcı "Yüz Tanı" düğmesine tıklayarak yüz tanıma işlemini başlatır. Yüz tanıma JavaCV kütüphanesi kullanılarak gerçekleştirilmiştir (JavaCV, OpenCV için bir Java arayüzüdür). Bulunan yüz görüntüsü veri tabanında kayıtlı olan yüz görüntüleriyle eşleştirilmektedir. Eşleştirme sonucuna göre, öğrencinin sınava girişine izin verilmekte ya da verilmemektedir.

Eşleştirme sonucunu gösteren ekran görüntüsü Şekil 7.6'da görülmektedir. Burada yer alan "Sınava Başla" düğmesine tıklayarak sınav uygulamasına geçiş yapılır.

Şekil 6.6: Kullanıcı Kimlik Tespiti Sonuç Ekranı

Yüz tanıma işlemini yapan kodlar aşağıda verilmiştir:

```
package com.zk.yuz_tanima;

import java.nio.IntBuffer;
import java.util.List;
import java.util.ListIterator;

import com.zk.App;
import com.zk.kullanici.Kullanici;
import com.zk.kullanici.KullaniciDAO;
import com.zk.yuz_bul.Yuz;
import com.zk.yuz_bul.YuzDAO;

import static org.bytedeco.javacpp.opencv_face.*;
import static org.bytedeco.javacpp.opencv_core.*;
import static org.bytedeco.javacpp.opencv_imgcodecs.*;
```

```
import static org.bytedeco.javacpp.opencv_imgproc.*;

public class YuzTaniyici {
    public static void tani() {
       Mat testYuz = imdecode(new Mat(App.matOfByteArr),
CV_LOAD_IMAGE_UNCHANGED);

        KullaniciDAO kulDAO = new KullaniciDAO();

        YuzDAO yuzDAO = new YuzDAO();
        List<Yuz> yuzler = yuzDAO.bul();
       ListIterator<Yuz> yuzlerIter = yuzler.listIterator();

        MatVector goruntuler = new MatVector(yuzDAO.topYuz());

        Mat labels = new Mat(yuzDAO.topYuz(),1,CV_32SC1);
        IntBuffer labelsBuf = labels.getIntBuffer();

       int sayac=0;
       while(yuzlerIter.hasNext()){
             Yuz yuz = yuzlerIter.next();
             Mat mat = imdecode(new Mat(yuz.goruntuAl()),
CV_LOAD_IMAGE_UNCHANGED);
             goruntuler.put(sayac, mat);
             labelsBuf.put(sayac, yuz.kulIdAl());
             sayac++;
        }

        FaceRecognizer faceRecognizer = createEigenFaceRecognizer();
        faceRecognizer.train(goruntuler, labels);
        int predictedLabel = faceRecognizer.predict(testYuz);

        Kullanici kul = kulDAO.idDenBul(predictedLabel);
        App.tanimaSonucu.setText("Merhaba "+kul.adAl()+"
"+kul.soyadAl());
        App.taniBut.setVisible(false);
        App.taniBut.removeAll();
        App.sinGirButGoster();
    }
}
```

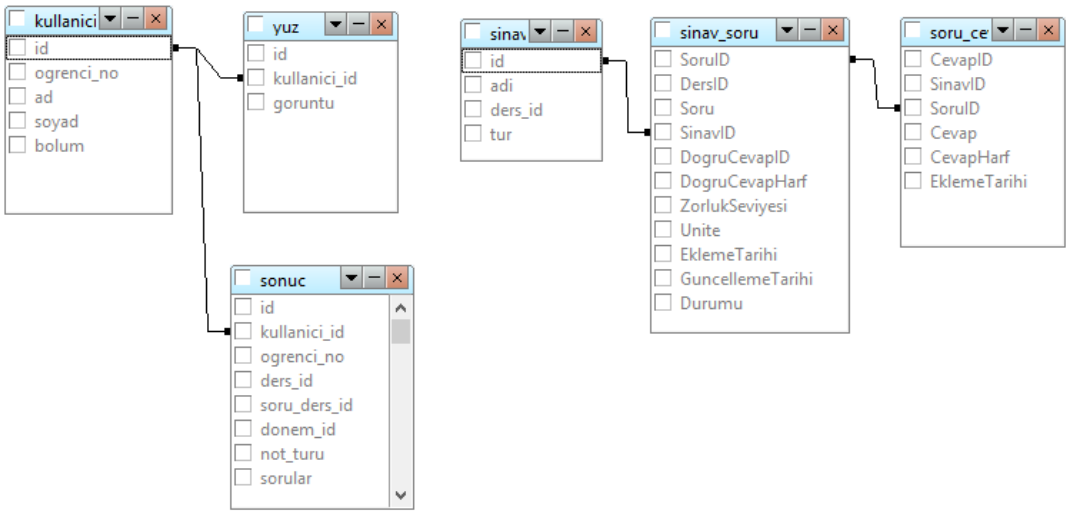
6.8 Sınav Arayüzü

Kimlik tespitinden başarıyla geçen öğrencilerin sınav oldukları web tabanlı uygulamadır. Uygulamanın ekran görüntüsü Şekil 6.7'de görülmektedir.

Şekil 6.7: Sınav Arayüzü Görüntüsü

6.9 Sınav Sisteminin Veri tabanı Yapısı

Sınav sisteminde kullanılan veri tabanı yapısı Şekil 7.6'da görülmektedir. "yuz" tablosu, 100x100 piksel boyutlarındaki gri seviye yüz görüntülerinin ikili olarak saklandığı BLOB tipinden bir alandır.



Şekil 6.8: Sınav Sistemi Veritabanı Yapısı Görüntüsü

7. SONUÇ VE ÖNERİLER

Bu tezin amacı, bir kişinin başka bir kişinin yerine sınava girerek sahtekarlık yapmasını önlemektir.

Bu çalışmada, biyometrik güvenlik sistemlerinden yüz tanıma sistemi araştırılmış ve bir çok kullanım alanı tespit edilmiştir. Bu sistemin, çevrimiçi sınav sistemlerinde de kimlik tespiti için kullanılabileceği örnek uygulama ile kanıtlanmıştır.

Bilgisayar tabanlı sistemlerde yapılan sınavlarda, özellikle gözetmenler tarafından tanınmayan öğrencilerin sınavlara girişinde problemler yaşanabilmektedir. Bu sınav sistemlerinde genellikle, sınava giriş hakkı, kullanıcıya daha önceden verilen bir anahtar (kullanıcı adı ve şifre gibi) doğrulanarak verilmektedir. Bu anahtarı bilen herhangi bir kişi, o anahtarın gerçek sahibiymiş gibi sınava girebilmektedir. Buda, başkasının yerine sınava girme problemini ortaya çıkarmaktadır. Bu çalışma ile bu problemin önüne geçilmiştir.

Öncelikle, sınav sistemine giriş yapacak kişilerin ad, soyad, öğrenci numarası ve yüz görüntü bilgilerinden oluşan kimlik bilgileri veri tabanına kaydedilmektedir. Yüz görüntüleri 100x100 piksel boyutlarında ve gri ölçekte tutulur.

Sınav zamanında, web kamera yoluyla alınan yüz görüntüsü veri tabanındaki yüz görüntüleri ile karşılaştırılır. Bu karşılaştırma sonucuna göre öğrenciye sınava giriş hakkı verilmekte ya da verilmemektedir.

Elli öğrenci sisteme tanımlanmış ve her bir öğrenci için bir tane yüz görüntüsü veri tabanına kaydedilmiştir. Sistem, bu 50 kişiden 40'ının kimlik tespitini doğru olarak yapmıştır.

Kalan 10 kişi için sisteme farklı pozlarda 5 yüz görüntüsü daha eklemiştir. Sonrasında, yüz tanıma sistemi bu kişilerden 8'inin kimlik tespitini doğru olarak yapmıştır. Geri kalan 2 kişi yanlış tanınmıştır.

KAYNAKLAR

Alpaydın, E. (2013). *Yapay Öğrenme* (2 b.). İstanbul: Boğaziçi Üniversitesi Yayınevi.

Belhumeur, P., Hespanha, J., & Kriegman, D. (1997). Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. *IEEE Transaction On Pattern Analysis and Machine Intelligence, 19*(7), 711 - 720.

Bledsoe, W. W. (1964). The model method in facial recognition. Technical report pri. *Panoramic Research, Inc*.

Brunelli, R., & Poggio, T. (1992). Face recognition through geometrical features.

Brunelli, R., & Poggio, T. (1993). Face recognition: Features versus templates. *IEEE Transaction on Pattern Analysis and Machine Intelligence, 15*, 1042-1053.

Brunelli, R., & Poggio, T. (1995). Template Matching: Matched Spatial Filters and Beyond. *IRST Tech. Rep.*

Erdoğan, A. Y. (2010). Yüz Tanımada Özyüz ve Fisher Yüz Algotimalarının İncelenmesi. *Yüksek Lisans Tezi*, 28. Ankara: Ankara Üniversitesi, Fen Bilimleri Enstitüsü.

Erişti, E. (2010). Görüntü İşlemede Yeni Bir Soluk, OPENCV. *Akademik Bilişim'10*, (s. 223-228). Muğla.

Goldstein, A. J., Harmon, L. D., & Lesk, A. B. (1971). Identification of Human Faces. *59*(5), 748-760.

Gonzalez, R., & Woods, R. (2014). *Sayısal Görüntü İşleme* (3.Baskıdan Çeviri b.). Ankara: Palme Yayıncılık.

Nabiyev, V. V. (2012). *Yapay Zeka.* Ankara: Seçkin Yayıncılık.

Şamlı, R., & M.Erkal, Y. (2009). Biyometrik Güvenlik Sistemleri. Şanlıurfa: XI Akademik Bilişim Konferansı.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience, 3*(1), 71-86.

Varlık, A., & Çorumluoğlu, Ö. (2011). Dijital Fotogrametri Teknikleri İle Kişi Tanıma. *Harita Teknolojileri Elektronik Dergisi, 3*(2), 1-24.

Varol, A., & Cebe, B. (2011). Yüz Tanıma Algoritmaları. *International Computer & Instructional Technologies Symposium.* Elazığ: Firat University.

Yalçın, N. (2008, Mart). Konuşma Tanıma Teorisi ve Teknikleri. *Kastamonu Eğitim Dergisi*, s. 249-266.

Yalçın, N., & Gürsel, F. (2015). Biyometrik Güvenlik Sistemlerinin İncelenmesi. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, 398-413.

EKLER

EK A: Program kodları

EK A

```
package com.zk;

import java.awt.Color;
import java.awt.Component;
import java.awt.Dimension;
import java.awt.GridBagConstraints;
import java.awt.GridBagLayout;
import java.awt.GridLayout;
import java.awt.Image;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;

import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.SwingConstants;
import javax.swing.event.ListSelectionEvent;
import javax.swing.event.ListSelectionListener;

import org.opencv.core.Mat;
import org.opencv.core.MatOfRect;
import org.opencv.core.Point;
import org.opencv.core.Rect;
import org.opencv.core.Scalar;
import org.opencv.core.Size;
import org.opencv.imgcodecs.Imgcodecs;
import org.opencv.imgproc.Imgproc;
import org.opencv.objdetect.CascadeClassifier;
import org.opencv.videoio.VideoCapture;
import org.opencv.videoio.Videoio;

import com.zk.giris.GirisController;
import com.zk.kullanici.Kullanici;
import com.zk.util.ImageProcessor;
import com.zk.yuz_bul.Yuz;
import com.zk.yuz_bul.YuzDAO;
import com.zk.yuz_bul.YuzTableModel;
import com.zk.yuz_tanima.YuzTaniyici;

import org.opencv.core.*;
public class App {
	static{ System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
	}

	public static JDialog dialog;
	private static JLabel imageView = new JLabel();;
	private static CascadeClassifier faceDetector;
	public static YuzTableModel yuzTableModel;
```

```
public static JTable yuzTable;
public static int secilenGoruntuId;
public static JPanel adKaydetPanel = new JPanel();
public static JLabel tamAdi = new JLabel();
public static JButton kaydetButton = new JButton("Kaydet");
public static JScrollPane yuzScrollPane;
public static Mat yuzMat;
public static Mat yuzMatGray;
public static byte[] yuzByte;
public static byte[] yuzByteGray;
public static BufferedImage yuzBuf;
public static BufferedImage yuzBufGray;
public static Mat yuzMatResize = new Mat();
public static BufferedImage yuzBufResize;
public static byte[] matOfByteArr;
public static ImageProcessor imageProcessor = new ImageProcessor();
public static JLabel tanimaSonucu;
public static JButton sinGirBut;
public static JButton taniBut;

public static YuzDAO yuzDAO;

public static Kullanici kullanici;

public static final String APP_NAME = "Yüz Tanıma";
public static final String APP_VERSION = "1.0";

public static void main(String[] args) {
kullaniciGirisi();
GUIYuzBulma();
cascadeYukle();
kamCalistir(args);
}

private static void cascadeYukle() {
String cascadePath =
"src/main/resources/cascades/haarcascade_frontalface_alt.xml";
    faceDetector = new CascadeClassifier(cascadePath);
}

   private static void kullaniciGirisi(){
      GirisController giris = new GirisController();
      giris.kimlikSor();
   }

public static void GUIYuzBulma() {
JFrame sahipsiz = null;
dialog = new JDialog(sahipsiz,"Yüz Bulma", true);
dialog.setLayout(new GridBagLayout());
dialog.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
dialog.setSize(400,400);
setupImage();
}
```

```
public static void GUIYuzTanima() {
            JFrame sahipsiz = null;
            dialog = new JDialog(sahipsiz,"Yüz Tanıma", true);
            dialog.setLayout(new GridBagLayout());

      dialog.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
            dialog.setSize(400,400);
            setupImage();
      }

      private static void setupImage() {
            GridBagConstraints c = new GridBagConstraints();
            c.fill = GridBagConstraints.WEST;
            c.gridx = 0;
            c.gridy = 0;
            dialog.add(imageView,c);
      }

      public static void setupButtons() {
            GridBagLayout gbl = new GridBagLayout();
            adKaydetPanel.setLayout(gbl);
            GridBagConstraints gbc = new GridBagConstraints();
            gbc.fill=GridBagConstraints.HORIZONTAL;
            gbc.gridx = 0;
            gbc.gridy = 0;
            //gbc.fill=GridBagConstraints.WEST;
            gbc.ipadx=250;
            try{
                  tamAdi.setText(kullanici.adAl()+'
'+kullanici.soyadAl());
            }catch(NullPointerException ex){
                  System.out.println(ex.getMessage());
            }
            tamAdi.setAlignmentX(Component.LEFT_ALIGNMENT);
            tamAdi.setOpaque(true);
            gbl.setConstraints(tamAdi, gbc);
            adKaydetPanel.add(tamAdi, gbc);

            kaydetButton.addActionListener(new ActionListener() {
                  public void actionPerformed(ActionEvent event) {
                        yuzKaydet();
                  }
            });

            gbc.gridx = 1;
            gbc.gridy = 0;
            gbc.ipadx=0;
            gbl.setConstraints(kaydetButton, gbc);
            adKaydetPanel.add(kaydetButton, gbc);

            GridBagConstraints c = new GridBagConstraints();
            c.gridx = 0;
            c.gridy = 1;
            c.anchor = GridBagConstraints.WEST;
            dialog.add(adKaydetPanel,c);
      }

      public static void kulYuzleriGoster() {
```

```
try{
 yuzDAO = new YuzDAO(kullanici);
 yuzTableModel = new YuzTableModel(kullanici);
 yuzTable = new JTable(yuzTableModel);
 yuzTable.setBackground(Color.LIGHT_GRAY);
 yuzTable.getColumnModel().getColumn(0).setPreferredWidth(20);
 yuzTable.getColumnModel().getColumn(1).setPreferredWidth(100);
 yuzTable.setRowHeight(100);
 yuzTable.setFillsViewportHeight(true);
 yuzTable.setAutoResizeMode(JTable.AUTO_RESIZE_OFF);
 yuzScrollPane = new JScrollPane();
 yuzScrollPane.getViewport().add(yuzTable);
 yuzScrollPane.setPreferredSize(new Dimension(140, 480));
 GridBagConstraints c = new GridBagConstraints();
 c.gridx = 1;
 c.gridy = 0;
 dialog.add(yuzScrollPane,c);

 yuzTable.getSelectionModel().addListSelectionListener(new
ListSelectionListener(){
         public void valueChanged(ListSelectionEvent event) {
             if ( event.getValueIsAdjusting() ){
                 return;
              }
             secilenGoruntuId = (int)
(yuzTable.getValueAt(yuzTable.getSelectedRow(), 0));
             System.out.println("yuz id:"+secilenGoruntuId);
         }
     });

 }catch(NullPointerException ex){

 }
 }

 public static void silEgitButtonGoster() {
 JButton silButton = new JButton("Sil");
 silButton.addActionListener(new ActionListener() {
 public void actionPerformed(ActionEvent event) {
 yuzSil();
 }
 });
 silButton.setAlignmentX(Component.RIGHT_ALIGNMENT);
 GridLayout gridRowLayout = new GridLayout(1,0);
 gridRowLayout.setVgap(5);
 JPanel buttonsPanel = new JPanel(gridRowLayout);

 buttonsPanel.add(silButton);

 GridBagConstraints c = new GridBagConstraints();

                    c.gridx = 1;
                    c.gridy = 1;
                    dialog.add(buttonsPanel,c);
             }

 private static void yuzSil(){
 Yuz yuz = new Yuz();
```

```
yuz.idVer(secilenGoruntuId);
            yuzDAO.sil(yuz);

        yuzTableModel.fireTableRowsDeleted(secilenGoruntuId,
secilenGoruntuId);
        yuzTableModel.yenile();
        yuzTableModel.fireTableDataChanged();
      }

public static void taniButtonuGoster() {
            taniBut = new JButton("Yüz Tanı");
            taniBut.addActionListener(new ActionListener() {
                  public void actionPerformed(ActionEvent event) {
                        tanimaSonucu.setText("Kimlik tespiti
yapılıyor. Lütfen bekleyiniz...");
                        YuzTaniyici.tani();
                  }
            });
            taniBut.setAlignmentX(Component.CENTER_ALIGNMENT);

            GridLayout gridRowLayout = new GridLayout(3,0);
            JPanel buttonsPanel = new JPanel(gridRowLayout);
            buttonsPanel.add(taniBut);
            tanimaSonucu = new JLabel("",SwingConstants.CENTER);
            tanimaSonucu.setPreferredSize(new Dimension(200, 5));
            buttonsPanel.add(tanimaSonucu);
            buttonsPanel.add(taniBut);
            GridBagConstraints c = new GridBagConstraints();
            c.gridx = 0;
            c.gridy = 1;
            dialog.add(buttonsPanel,c);
      }

public static void sinGirButGoster() {
            sinGirBut = new JButton("Sınava Başla");
            sinGirBut.addActionListener(new ActionListener() {
                  public void actionPerformed(ActionEvent event) {

                  }
            });
            sinGirBut.setAlignmentX(Component.CENTER_ALIGNMENT);

            GridLayout gridRowLayout = new GridLayout(3,0);
            JPanel buttonsPanel = new JPanel(gridRowLayout);
            buttonsPanel.add(sinGirBut);
            GridBagConstraints c = new GridBagConstraints();
            c.gridx = 0;
            c.gridy = 2;
            dialog.add(buttonsPanel,c);
      }


private static void kamCalistir(String[] args) {
            Mat webcamMatImage = new Mat();
            Image tempImage;
            VideoCapture capture = new VideoCapture(0);
            capture.set(Videoio.CV_CAP_PROP_FRAME_WIDTH,640);
            capture.set(Videoio.CV_CAP_PROP_FRAME_HEIGHT,480);
```

```
if( capture.isOpened()){
while (true){
 capture.read(webcamMatImage);
 if( !webcamMatImage.empty() ){
 yuzBul(webcamMatImage);
 tempImage= imageProcessor.toBufferedImage(webcamMatImage);

 ImageIcon imageIcon = new ImageIcon(tempImage, "Alınan görüntü");
 imageView.setIcon(imageIcon);
 dialog.pack();
 }
 else{
 System.out.println("Görüntü alınamadı!");
 break;
 }
 }
 }
 else{
 System.out.println("Görüntü okunamadı!");
 }
 }

 private static void yuzBul(Mat image) {
     MatOfRect faceDetections = new MatOfRect();
     faceDetector.detectMultiScale( image, faceDetections, 1.1,
7,0, new Size(150,40), new Size());
     Rect rectCrop=null;

     for (Rect rect : faceDetections.toArray()) {
         Imgproc.rectangle(image, new Point(rect.x, rect.y), new
Point(rect.x + rect.width, rect.y + rect.height), new Scalar(0, 255,
0));
         rectCrop = new Rect(rect.x, rect.y, rect.width,
rect.height);
     }

     if(rectCrop != null){
       yuzMat = new Mat(image,rectCrop);
       yuzMatResize = new Mat();
 Size boyutlar = new Size(100,100);
 Imgproc.resize( yuzMat, yuzMatResize, boyutlar);
 yuzBufResize = imageProcessor.toBufferedImage(yuzMatResize);

       yuzMatGray = new Mat();
       Imgproc.cvtColor(yuzMatResize, yuzMatGray,
Imgproc.COLOR_RGB2GRAY);

       MatOfByte matOfByte = new MatOfByte();
       Imgcodecs.imencode(".jpg", yuzMatGray, matOfByte);
       matOfByteArr = matOfByte.toArray();
     }
 }
 private static void yuzKaydet(){
               Yuz yuz = new Yuz();
               yuz.goruntuVer(matOfByteArr);

               yuzDAO.ekle(yuz);
               yuzTableModel.yenile();
```

```
                yuzTableModel.fireTableDataChanged();
}

        }



        package com.zk.yuz_tanima;

        import java.nio.IntBuffer;
        import java.util.List;
        import java.util.ListIterator;

        import com.zk.App;
        import com.zk.kullanici.Kullanici;
        import com.zk.kullanici.KullaniciDAO;
        import com.zk.yuz_bul.Yuz;
        import com.zk.yuz_bul.YuzDAO;

        import static org.bytedeco.javacpp.opencv_face.*;
        import static org.bytedeco.javacpp.opencv_core.*;
        import static org.bytedeco.javacpp.opencv_imgcodecs.*;
        import static org.bytedeco.javacpp.opencv_imgproc.*;

        public class YuzTaniyici {
            public static void tani() {
              Mat testYuz = imdecode(new Mat(App.matOfByteArr),
        CV_LOAD_IMAGE_UNCHANGED);

                KullaniciDAO kulDAO = new KullaniciDAO();

                YuzDAO yuzDAO = new YuzDAO();
                List<Yuz> yuzler = yuzDAO.bul();
              ListIterator<Yuz> yuzlerIter = yuzler.listIterator();

                MatVector goruntuler = new MatVector(yuzDAO.topYuz());

                Mat labels = new Mat(yuzDAO.topYuz(),1,CV_32SC1);
                IntBuffer labelsBuf = labels.getIntBuffer();

              int sayac=0;
              while(yuzlerIter.hasNext()){
                    Yuz yuz = yuzlerIter.next();
                    Mat mat = imdecode(new Mat(yuz.goruntuAl()),
        CV_LOAD_IMAGE_UNCHANGED);
                    goruntuler.put(sayac, mat);
                    labelsBuf.put(sayac, yuz.kulIdAl());
                    sayac++;
                }

                FaceRecognizer faceRecognizer = createEigenFaceRecognizer();
                faceRecognizer.train(goruntuler, labels);
                int predictedLabel = faceRecognizer.predict(testYuz);

                Kullanici kul = kulDAO.idDenBul(predictedLabel);
                App.tanimaSonucu.setText("Merhaba "+kul.adAl()+"
        "+kul.soyadAl());
                App.taniBut.setVisible(false);
```

```
            App.taniBut.removeAll();
            App.sinGirButGoster();
        }
}
```

```
package com.zk.yuz_tanima;

import com.zk.kullanici.Kullanici;
import com.zk.main.MainWindow;
import com.zk.util.Util;
import com.zk.yuz_bul.Yuz;

import java.nio.charset.Charset;
import java.nio.charset.StandardCharsets;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.util.ArrayList;
import java.util.Collections;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.logging.Logger;

public final class YuzTanimaDAO {

   static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
   static final String DB_URL = "jdbc:mysql://localhost/yuz_tanima";

   static final String USER = "root";
   static final String PASS = "kaya";

   Connection conn = null;
   Statement stmt = null;

   private static final Map<String, Yuz> table = new
LinkedHashMap<>();
   private static final String NULL = "NULL";
   private final static Charset ENCODING = StandardCharsets.UTF_8;

   private Kullanici kullanici;

   public YuzTanimaDAO(Kullanici kullanici){
     this.kullanici=kullanici;
   }

   List<Yuz> list() {
   System.out.println(this.kullanici);
   bul();
   List<Yuz> result = new ArrayList<>(table.values());
   return result;
          }
```

```
public void bul() {
     try{
        Class.forName("com.mysql.jdbc.Driver");

        conn = DriverManager.getConnection(DB_URL, USER, PASS);

        stmt = conn.createStatement();
        String sql;
        sql = "SELECT id, kullanici_id, goruntu FROM yuz ORDER BY
kullanici_id";
        ResultSet rs = stmt.executeQuery(sql);
        while(rs.next()){
         int id = rs.getInt("id");
         String kullanici_id = rs.getString("kullanici_id");
       byte[] goruntu = rs.getBytes("goruntu");
         Yuz yuz = new Yuz(goruntu);
       yuz.idVer(id);
        }
        rs.close();
        stmt.close();
        conn.close();
     }catch(SQLException se){
     }catch(Exception e){
     }finally{
        try{
           if(stmt!=null)
              stmt.close();
        }catch(SQLException se2){
        }
        try{
           if(conn!=null)
              conn.close();
        }catch(SQLException se){
           se.printStackTrace();
        }
     }
  }

   private void appendTo(StringBuilder aText, Object aField, String
aAppend) {
     if (Util.textAra(Util.format(aField))) {
       aText.append(Util.format(aField));
     }
     else {
       aText.append(NULL);
     }
     aText.append(aAppend);
   }

   private static String maybeNull(String aText) {
     return NULL.equals(aText) ? null : aText;
   }


}
```

```
package com.zk.kullanici;

import java.util.*;

import com.zk.exception.InvalidInputException;
import com.zk.util.Util;

public final class Kullanici implements Comparable<Kullanici>{

  // PRIVATE
  private String fId;
  private final String fOgrNo;
  private final String fAd;
  private final String fSoyad;
  private final String fBolum;
  private static final int EQUAL = 0;
  private static final int DESCENDING = -1;

  public Kullanici(String aId, String aOgrNo, String aAd, String
aSoyad, String aBolum ) throws InvalidInputException {
    fId = aId;
    fOgrNo = aOgrNo;
    fAd = aAd;
    fSoyad= aSoyad;
    fBolum = aBolum;
    validateState();
  }

  public String idAl(){
        return fId;
  }

  void idVer(String aId){
        fId = aId;
  }

  public String ogrNoAl(){
        return fOgrNo;
  }

  public String adAl(){
        return fAd;
  }

  public String soyadAl(){
        return fSoyad;
  }

  public String bolumAl(){
        return fBolum;
  }


  @Override public boolean equals(Object aThat){
    if ( this == aThat ) return true;
    if ( !(aThat instanceof Kullanici) ) return false;
    Kullanici that = (Kullanici)aThat;
    return
```

```
      areEqual(this.fOgrNo, that.fOgrNo) &&
      areEqual(this.fAd, that.fAd) &&
      areEqual(this.fSoyad, that.fSoyad)  &&
      areEqual(this.fBolum, that.fBolum)
    ;
  }

  @Override public int hashCode(){
    int result = 17;
    result = addHash(result, fOgrNo);
    result = addHash(result, fAd);
    result = addHash(result, fSoyad);
    result = addHash(result, fBolum);
    return result;
  }

  @Override public String toString(){
    return
      "Kullanici  Id:" + fId + " Öğrenci No:" + fOgrNo + " Adı" + fAd
+
      " Soyadı" + fSoyad + " BÖlÜmü:"+fBolum
    ;
  }

  @Override public int compareTo(Kullanici aThat) {
    if ( this == aThat ) return EQUAL;

    int comparison = DESCENDING*comparePossiblyNull(this.fAd,
aThat.fAd);
    if ( comparison != EQUAL ) return comparison;

    comparison = this.fOgrNo.compareTo(aThat.fOgrNo);
    if ( comparison != EQUAL ) return comparison;

    comparison = comparePossiblyNull(this.fSoyad, aThat.fSoyad);
    if ( comparison != EQUAL ) return comparison;

    comparison = comparePossiblyNull(this.fBolum, aThat.fBolum);
    if ( comparison != EQUAL ) return comparison;


    return EQUAL;
  }

  /** Sort by Title. */
  public static Comparator<Kullanici> TITLE_SORT = new
Comparator<Kullanici>(){
    @Override public int compare(Kullanici aThis, Kullanici aThat) {
      if ( aThis == aThat ) return EQUAL;

      int comparison = aThis.fOgrNo.compareTo(aThat.fOgrNo);
      if ( comparison != EQUAL ) return comparison;

      comparison = DESCENDING*comparePossiblyNull(aThis.fAd,
aThat.fAd);
      if ( comparison != EQUAL ) return comparison;

      comparison = comparePossiblyNull(aThis.fSoyad, aThat.fSoyad);
```

```
      if ( comparison != EQUAL ) return comparison;

      comparison = comparePossiblyNull(aThis.fBolum, aThat.fBolum);
      if ( comparison != EQUAL ) return comparison;

      return EQUAL;
    };
  };

  public static Comparator<Kullanici> RATING_SORT = new
Comparator<Kullanici>(){
    @Override public int compare(Kullanici aThis, Kullanici aThat) {
      if ( aThis == aThat ) return EQUAL;

      int comparison = DESCENDING*comparePossiblyNull(aThis.fSoyad,
aThat.fSoyad);
      if ( comparison != EQUAL ) return comparison;

      comparison = DESCENDING*comparePossiblyNull(aThis.fAd,
aThat.fAd);
      if ( comparison != EQUAL ) return comparison;

      comparison = aThis.fOgrNo.compareTo(aThat.fOgrNo);
      if ( comparison != EQUAL ) return comparison;



      return EQUAL;
    };
  };

  /** Sort by Comment. */
  public static Comparator<Kullanici> COMMENT_SORT = new
Comparator<Kullanici>(){
    @Override public int compare(Kullanici aThis, Kullanici aThat) {
      if ( aThis == aThat ) return EQUAL;

      int comparison = comparePossiblyNull(aThis.fOgrNo,
aThat.fOgrNo);
      if ( comparison != EQUAL ) return comparison;

      comparison = DESCENDING*comparePossiblyNull(aThis.fAd,
aThat.fAd);
      if ( comparison != EQUAL ) return comparison;

      comparison = comparePossiblyNull(aThis.fSoyad, aThat.fSoyad);
      if ( comparison != EQUAL ) return comparison;

      return EQUAL;
    };
  };


  private void validateState() throws InvalidInputException {
    InvalidInputException ex = new InvalidInputException();

    if( ! Util.textAra(fOgrNo) ) {
        ex.add("Örenci numarası boş bırakılamaz");
```

```
        }
        if( ! Util.textAra(fAd) ) {
            ex.add("Ad boş bırakılamaz");
        }
        if( ! Util.textAra(fSoyad) ) {
            ex.add("Soyad boş bırakılamaz");
        }

        if ( ex.hasErrors() ) {
          throw ex;
        }
      }

      private boolean areEqual(Object aThis, Object aThat){
        return aThis == null ? aThat == null : aThis.equals(aThat);
      }

      private int addHash(int aHash, Object aField){
        int result = 37*aHash;
        if (aField != null){
          result = result + aField.hashCode();
        }
        return result;
      }

      /** Utility method.  */
      private static <T extends Comparable<T>> int comparePossiblyNull(T
aThis, T aThat){
        int result = EQUAL;
        int BEFORE = -1;
        int AFTER = 1;

        if(aThis != null && aThat != null){
          result = aThis.compareTo(aThat);
        }
        else {
          //at least one reference is null - special handling
          if(aThis == null && aThat == null) {
            //do nothing - they are not distinct
          }
          else if(aThis == null && aThat != null) {
            result = BEFORE;
          }
          else if( aThis != null && aThat == null) {
            result = AFTER;
          }
        }
        return result;
      }

    }
```

```
package com.zk.kullanici;

import java.nio.charset.Charset;
import java.nio.charset.StandardCharsets;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.PreparedStatement;
import java.util.ArrayList;
import java.util.Collections;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.logging.Logger;

import com.zk.kullanici.Kullanici;
import com.zk.main.MainWindow;
import com.zk.util.Util;

public final class KullaniciDAO {

   static final String JDBC_DRIVER = "com.mysql.jdbc.Driver";
   static final String DB_URL =
"jdbc:mysql://localhost/yuz_tanima?useUnicode=true&characterEncoding
=UTF-8";

   static final String USER = "root";
   static final String PASS = "";

   Connection conn = null;
   Statement stmt = null;

  private static final Map<String, Kullanici> fTable = new
LinkedHashMap<>();
  private static final String NULL = "NULL";
  private final static Charset ENCODING = StandardCharsets.UTF_8;


  public void KullaniciDao(){
        bul();
  }

  List<Kullanici> list() {
      bul();
      List<Kullanici> result = new ArrayList<>(fTable.values());
    Collections.sort(result);
    return result;
  }

  public void ekle(Kullanici aKullanici) {
        try{
           Class.forName("com.mysql.jdbc.Driver");
           conn = DriverManager.getConnection(DB_URL, USER, PASS);
```

```
            PreparedStatement pstmt = conn.prepareStatement("INSERT
INTO kullanici (ogrenci_no, ad, soyad,bolum) VALUES (?, ?, ?, ?)",
PreparedStatement.RETURN_GENERATED_KEYS);
            pstmt.setString(1, aKullanici.ogrNoAl());
            pstmt.setString(2, aKullanici.adAl());
            pstmt.setString(3, aKullanici.soyadAl());
            pstmt.setString(4, aKullanici.bolumAl());
            pstmt.executeUpdate();

            ResultSet keys = pstmt.getGeneratedKeys();
            keys.next();
            bul(keys.getString(1));
         }catch(SQLException se){
            se.printStackTrace();
         }catch(Exception e){
            e.printStackTrace();
         }finally{
            try{
               if(stmt!=null)
                  conn.close();
            }catch(SQLException se){
            }// do nothing
            try{
               if(conn!=null)
                  conn.close();
            }catch(SQLException se){
               se.printStackTrace();
            }
         }
      }

  void degistir(Kullanici aKullanici) {
         try{
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            PreparedStatement pstmt = conn.prepareStatement("UPDATE
kullanici SET ogrenci_no=?, ad=?, soyad=?, bolum=? WHERE id=?");
            pstmt.setString(1, aKullanici.ogrNoAl());
            pstmt.setString(2, aKullanici.adAl());
            pstmt.setString(3, aKullanici.soyadAl());
            pstmt.setString(4, aKullanici.bolumAl());
            pstmt.setString(5, aKullanici.idAl());
            pstmt.executeUpdate();
            bul();
         }catch(SQLException se){
            se.printStackTrace();
         }catch(Exception e){
            e.printStackTrace();
         }finally{
            try{
               if(stmt!=null)
                  conn.close();
            }catch(SQLException se){
            }
            try{
               if(conn!=null)
                  conn.close();
            }catch(SQLException se){
```

```
                se.printStackTrace();
            }
        }
    }

    void sil(Kullanici aKullanici) {
        try{
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            PreparedStatement pstmt = conn.prepareStatement("DELETE
FROM kullanici WHERE id=?");
            pstmt.setString(1, aKullanici.idAl());
            pstmt.executeUpdate();
            fTable.clear();
            bul();
        }catch(SQLException se){
            se.printStackTrace();
        }catch(Exception e){
            e.printStackTrace();
        }finally{
            try{
                if(stmt!=null)
                    conn.close();
            }catch(SQLException se){
            }
            try{
                if(conn!=null)
                    conn.close();
            }catch(SQLException se){
                se.printStackTrace();
            }
        }
    }

    public List<Kullanici> bul() {
        List<Kullanici> kullar=null;
        try{
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            stmt = conn.createStatement();
            String sql;
            sql = "SELECT id, ogrenci_no, ad, soyad,bolum FROM
kullanici";
            ResultSet rs = stmt.executeQuery(sql);
            while(rs.next()){
             String id = rs.getString("id");
             String ogrNo = rs.getString("ogrenci_no");
                 String ad = rs.getString("ad");
                 String soyad = rs.getString("soyad");
                 String bolum = rs.getString("bolum");
                 Kullanici kullanici = new Kullanici(id, ogrNo, ad,
soyad, bolum);
                 fTable.put(kullanici.idAl(), kullanici);
                 kullar = new ArrayList<Kullanici>();
                 kullar.add(kullanici);
            }
            Collections.sort(kullar);
            rs.close();
```

```
            stmt.close();
            conn.close();
         }catch(SQLException se){
            se.printStackTrace();
         }catch(Exception e){
            e.printStackTrace();
         }finally{
            try{
               if(stmt!=null)
                  stmt.close();
            }catch(SQLException se2){
            }
            try{
               if(conn!=null)
                  conn.close();
            }catch(SQLException se){
               se.printStackTrace();
            }
         }
         return kullar;
      }

  public void bul(int id) {
         try{
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            stmt = conn.createStatement();
            String sql;
            sql = "SELECT id, ogrenci_no, ad, soyad, bolum FROM
kullanici WHERE id='"+id+"'";
            ResultSet rs = stmt.executeQuery(sql);
            rs.next();
        String kulId = rs.getString("id");
        String ogrNo = rs.getString("ogrenci_no");
            String ad = rs.getString("ad");
            String soyad = rs.getString("soyad");
            String bolum = rs.getString("bolum");
            Kullanici kullanici = new Kullanici(kulId, ogrNo, ad,
soyad, bolum);
              fTable.put(kullanici.idAl(), kullanici);
            rs.close();
            stmt.close();
            conn.close();
         }catch(SQLException se){
            se.printStackTrace();
         }catch(Exception e){
            e.printStackTrace();
         }finally{
            try{
               if(stmt!=null)
                  stmt.close();
            }catch(SQLException se2){
            }
            try{
               if(conn!=null)
                  conn.close();
            }catch(SQLException se){
               se.printStackTrace();
```

```
            }
        }
    }

  public Kullanici idDenBul(int id) {
        Kullanici kullanici=null;
        try{
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            stmt = conn.createStatement();
            String sql;
            sql = "SELECT id, ogrenci_no, ad, soyad, bolum FROM
kullanici WHERE id='"+id+"'";
            ResultSet rs = stmt.executeQuery(sql);
            rs.next();
            String kulId = rs.getString("id");
            String ogrNo = rs.getString("ogrenci_no");
            String ad = rs.getString("ad");
            String soyad = rs.getString("soyad");
            String bolum = rs.getString("bolum");
            kullanici = new Kullanici(kulId, ogrNo, ad, soyad,
bolum);
            rs.close();
            stmt.close();
            conn.close();
         }catch(SQLException se){
            se.printStackTrace();
         }catch(Exception e){
            e.printStackTrace();
         }finally{
            try{
               if(stmt!=null)
                  stmt.close();
            }catch(SQLException se2){
            }
            try{
               if(conn!=null)
                  conn.close();
            }catch(SQLException se){
               se.printStackTrace();
            }
         }
         return kullanici;
    }

  public void bul(String ogrNo) {
         try{
            Class.forName("com.mysql.jdbc.Driver");
            conn = DriverManager.getConnection(DB_URL, USER, PASS);
            stmt = conn.createStatement();
            String sql;
            sql = "SELECT id, ogrenci_no, ad, soyad, bolum FROM
kullanici WHERE ogrenci_no='"+ogrNo+"'";
            ResultSet rs = stmt.executeQuery(sql);
            rs.next();
            String id = rs.getString("id");
            String ad = rs.getString("ad");
            String soyad = rs.getString("soyad");
```

```
            String bolum = rs.getString("bolum");
            Kullanici kullanici = new Kullanici(id, ogrNo, ad,
soyad, bolum);
              fTable.put(kullanici.idAl(), kullanici);
            rs.close();
            stmt.close();
            conn.close();
         }catch(SQLException se){
            se.printStackTrace();
         }catch(Exception e){
            e.printStackTrace();
         }finally{
            try{
               if(stmt!=null)
                  stmt.close();
            }catch(SQLException se2){
            }
            try{
               if(conn!=null)
                  conn.close();
            }catch(SQLException se){
               se.printStackTrace();
            }
         }
      }
```

```
  private void appendTo(StringBuilder aText, Object aField, String
aAppend) {
    if (Util.textAra(Util.format(aField))) {
      aText.append(Util.format(aField));
    }
    else {
      aText.append(NULL);
    }
    aText.append(aAppend);
  }

  private static String maybeNull(String aText) {
    return NULL.equals(aText) ? null : aText;
  }


}
```

ÖZGEÇMİŞ

Ad-Soyad	**:**	Zihni Kaya
Doğum Yeri Ve Tarihi	**:**	İstanbul, 1977
E-posta	**:**	zihnikaya@gmail.com

ÖĞRENİM DURUMU **:**

- **Lisans** **:** 2004, Akdeniz Üniversitesi, İİBF, Maliye
- **Yüksek Lisans** **:** 2016, İstanbul Aydın Üniversitesi, Bilgisayar Mühendisliği, Bilgisayar Mühendisliği

TEZDEN TÜRETİLEN YAYINLAR, SUNUMLAR VE PATENTLER:

Kaya Z., 2nd International Congress on Education, Distance Education and Educational Technology, International Congress, Şubat 4-5, 2016 Antalya, Turkey