YÜZ TANIMA SİSTEMİNİN PARALEL PROGRAMLAMA İLE OPTİMİZASYONU

2012
YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ

Kadriye ÖZ

YÜZ TANIMA SİSTEMİNİN PARALEL PROGRAMLAMA İLE OPTİMİZASYONU

Kadriye ÖZ

Karabük Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalında

Yüksek Lisans Tezi

Olarak Hazırlanmıştır.

KARABÜK

Haziran 2012

Kadriye ÖZ tarafından hazırlanan "YÜZ TANIMA SİSTEMİNİN PARALEL PROGRAMLAMA İLE OPTİMİZASYONU" başlıklı bu tezin Yüksek Lisans Tezi olarak uygun olduğunu onaylarım.

Yrd. Doç. Dr. Salih GÖRGÜNOĞLU .....
Tez Danışmanı, Bilgisayar Mühendisliği Anabilim Dalı

Bu çalışma, jürimiz tarafından oy birliği ile Bilgisayar Mühendisliği Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir. 28/ 06 / 2012

Ünvanı, Adı SOYADI (Kurumu)	İmzası
Başkan : Doç. Dr. Ahmet DEMİR (KBÜ)	.....
Üye :Yrd. Doç. Dr. Salih GÖRGÜNOĞLU (KBÜ)	.....
Üye :Yrd. Doç. Dr. Baha ŞEN (KBÜ)	.....

...../..../2012

KBÜ Fen Bilimleri Enstitüsü Yönetim Kurulu, bu tez ile, Yüksek Lisans derecesini onamıştır.

Prof. Dr. Nizamettin KAHRAMAN .....
Fen Bilimleri Enstitüsü Müdürü

"Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim."

Kadriye ÖZ

ÖZET

Yüksek Lisans Tezi

YÜZ TANIMA SİSTEMİNİN PARALEL PROGRAMLAMA İLE OPTİMİZASYONU

Kadriye ÖZ

Karabük Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Tez Danışmanı:
Yrd. Doç. Dr. Salih GÖRGÜNOĞLU
Haziran 2012, 49 sayfa

Yüz tanıma, kişi tanımada kullanılan yaygın biyometrik yöntemlerden birisidir. Tanınması istenen kişinin yüz resminden çeşitli yöntemlerle elde edilen özellikler, veritabanındaki yüz resimleri ile karşılaştırılmaktadır. Bu sayede kişinin doğrulanması ve tanınması mümkün olmaktadır. Bu çalışmada yüz tanıma için özyüzler ve dalgacık dönüşümü yöntemleri kullanılmıştır. Yüz resimlerinin işlenmesi kişi ve kişiye ait resim sayısının artmasıyla birlikte daha fazla zaman almaktadır.

Günümüzde, bu tür zaman alıcı uygulamalarda performans artırmak için paralel programlama tekniklerinden yararlanılmaktadır. Paralel programlama çok çekirdekli işlemciler ile gerçekleştirilebildiği gibi ekran kartlarının yapısındaki çok çekirdekli grafik işlemciler ile de gerçekleştirilebilmektedir. CUDA ekran kartlarının desteklediği genel amaçlı bir paralel programlama mimarisidir.

Bu çalışmada özyüzler yöntemi CUDA ile birlikte kullanılarak sistem performansı artırılmıştır. Ayrıca Matlab Parallel Computing Toolbox kullanılarak, dalgacık dönüşümü yöntemiyle geliştirilen yüz tanıma sisteminde performans analizi yapılmıştır. Bu yöntemler kullanılarak geliştirilen, yüz tanıma sistemlerine ait hata oranları ve zaman analizleri karşılaştırmalı olarak sunulmuştur.

Anahtar Sözcükler **:** Yüz tanıma, paralel programlama, PCA, özyüzler, dalgacık dönüşümleri, CUDA, matlab.

Bilim Kodu **:** 902.1.014

ABSTRACT

M.Sc. Thesis

OPTIMIZATION OF FACE RECOGNITION SYSTEM WITH PARALLEL PROGRAMMING

Kadriye ÖZ

Karabuk University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Thesis Advisor:
Assist. Prof. Dr. Salih GÖRGÜNOĞLU
June 2012, 49 pages

Face recognition is a biometric method which is used frequently for human recognition. The features which are going to be recognized from a choosen persons's face image are compared to face images in the database. Such that it will be possible verification and recognition of the person. In this study, eigenfaces and wavelet transform methods are used for face recognition. Processing the facial images takes more time with increasing number of person and person's face images.

Nowadays, in such time-consuming applications, parallel programming techniques are used to improve performance. Parallel programming can be performed with both multi-core processors and with multi-core graphics processors which are in the structure of graphics cards. CUDA is a general purpose parallel programming architecture which is supported by graphics cards.

In this study, system performance is improved by using CUDA together with Eigenfaces method. In addition, in face recognition system which is developed with Wavelet transform method using Matlab Parallel Computing Toolbox, performance analysis are implemented. In this study, the error rates and time analysis of face recognition systems are presented comparatively.

Keywords **:** Face recognition, parallel programming, PCA, eigenfaces, wavelet, CUDA, matlab**.**

Science Code **:** 902.1.014

TEŞEKKÜR

Bu tez çalışmasının planlanmasında, araştırılmasında, yürütülmesinde ve oluşumunda ilgi ve desteğini esirgemeyen, engin bilgi ve tecrübelerinden yararlandığım, yönlendirme ve bilgilendirmeleriyle çalışmamı bilimsel temeller ışığında şekillendiren sayın hocam Yrd. Doç. Dr. Salih GÖRGÜNOĞLU'na sonsuz teşekkürlerimi sunarım.

Sevgili aileme ve eşime maddi, manevi hiçbir yardımı esirgemeden yanımda oldukları için ve sevgili kızım ve oğluma varlıkları için tüm kalbimle teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

SİMGELER ve KISALTMALAR DİZİNİ

KISALTMALAR

AMD : Advanced Micro Devices
ARDB : AR Face Database
AŞM : Aktif Şekil Modeli
BBA : Bağımsız Bilesen Analizi
CPU : Merkezi İşlem Birimi (Central Processing Unit)
CUDA : Compute Unified Device Architecture
CWT : Sürekli Dalgacık Dönüşümü (Continuous Wavelet Transform)
DNA : Deoksiribonükleik Asit
DWT : Ayrık Dalgacık Dönüşümü (Discrete Wavelet Transform)
EER : Eşit Hata Oranı(Equal Error Rate)
FAR : Yanlış Kabul Oranı (False Acceptance Rate)
FMR : Yanlış Eşleştirme Oranı (False Match Rate)
FNMR : Yanlış Eşleşmeme Oranı (False Non-Match Rate)
GPU : Grafik İşleme Birimi(Graphics Processing Unit)
GPGPU : Grafik İşlemcisinde Genel Amaçlı Programlama (General-Purpose Computing on Graphics Processing Units)
OpenCL : Open Computing Language
ORL : Olivetti Research Laboratory
PCA : Principal Compenent Analysis
PCA-NN : Principal Compenent Analysis-Neural Networks
PCASVM : Principal Compenent Analysis Support Vector Machine
PICS : Psychological Image Collection at Stirling
RAM : Rastgele Erişimli Hafıza (Random Access Memory)
RGB : Red-Green-Blue

BÖLÜM 1

GİRİŞ

Teknolojideki gelişmelere paralel olarak kişi ve kurumların güvenlik ihtiyacının karşılanma yöntemleri de değişmektedir. Güvenliği artırmak için temel yöntemlerin yanında alternatif yöntemlere de ihtiyaç duyulmuştur. Her insanda birbirinden farklılık göstermesi nedeniyle güvenlik ihtiyacının karşılanma yöntemlerinden birisi de biyometrik özelliklerdir. Göz retinası, parmak izi, imza, yüz, ses, DNA gibi özellikler biyometrik veri olarak kabul edilmektedir. Biyometrik verilerin kaybedilme, unutulma ve çalınma risklerinin minimum düzeyde olması nedeniyle kişi ve kurumlar tarafından tercih edilmektedir [1,2].

Çizelge 1.1'de biyometrik sistemlerin karşılaştırılması verilmiştir [3]. Parmak izi en yaygın ve eski olarak kullanılan bir biyometrik özelliktir. Yaygın kullanım olarak ikinci sırada 15.4 % kullanım oranı ile yüz görüntüsü yer almaktadır [4]. Yüz tanıma diğer biyometriklerle karşılaştırıldığında farklılık ve performans açısından diğer biyometrik sistemlere göre düşük olsa da evrensellik, elde edilirlik, kabul edilirlik ve tuzağa düşürme bakımından yüksek niteliğe sahip olduğu görülmektedir.

Çizelge 1.1. Çeşitli biyometrik teknolojilerin karşılaştırılması [3].

Biometrik Özellik	Evrensellik	Farklılık	Performans	Elde Edilirlik	Performans	Kabul Edilirlik	Tuzağa Düşürme
Yüz	Y	D	O	Y	D	Y	Y
Parmak İzi	O	Y	Y	O	Y	O	O
El Geometrisi	O	O	O	Y	O	O	O
Iris	Y	Y	Y	O	Y	D	D
İmza	D	D	D	Y	D	Y	Y
Ses	O	D	D	O	D	Y	Y

Yüksek= Y, Orta=O ve Düşük=D

Yüz tanıma problemi statik veya video sahnesindeki bir ya da daha çok kişinin veritabanındaki yüzlerle karşılaştırılarak tanınması veya belirlenmesi olarak tanımlanmaktadır [5,6]. Yüz tanımanın kullanılabileceği pek çok alan bulunmaktadır. Bunlar kimlik saptama ve kimlik doğrulama şeklinde gruplandırılabilir. En yararlı olduğu uygulamalardan bazıları kalabalık izleme, video içerik indeksleme, kişisel kimlik (örneğin ehliyet) ve güvenlik olarak sıralanabilir [7].

Yüz tanıma sistemlerinin performansını etkileyen poz açısı, ışıklandırma koşulları, yüz ifadesi, yaşlanma ve kapanma etkileri olarak gruplandırılabilecek 5 temel etmen bulunmaktadır. Yüz 3 boyutlu bir nesne olduğu için kullanılan açı görüntüde değişikliklere yol açmaktadır. Farklı ortamlarda alınan görüntülerde ışıklandırmanın faklılığı da alınan görüntüyü etkilemektedir. İnsanın o anki yüz ifadesi, kullandığı mimikler görüntüyü değiştirmektedir. Yaşlanma ya da gözlük, sakal, bıyık gibi yüzde kapanma da görüntüyü ve tanıma işlemini etkilemektedir [8].

Yüz tanıma çalışmalarında iki temel yöntem izlendiği tespit edilmiştir. Bu yöntemlerden birincisi yüz üzerinde yer alan, göz, ağız, burun gibi organların geometrik özelliklerinden elde edilen bilgiler ile bir özellik vektörü oluşturulmasına tanıma işleminin de benzer şekilde elde edilmiş vektörlerin karşılaştırılması ile yapılması ilkesine dayanmaktadır. Yüz tanımada kullanılan ikinci yöntem ise temel bileşen analizi adı ile anılmaktadır. Bilgi teorisi temellerinin esas alındığı bu yaklaşımlarda, yüzleri en az bilgi ve en iyi şekilde ifade etme yoluna gidilmektedir. Yüzler ifade edilirken, göz, ağız, burun gibi organlar bağımsız olarak düşünülmemekte, elde edilen kodlamada, onlara ait bazı ayırt edici özellikler de yer alabilmektedir [9].

Yerel öznitelik tabanlı yüz tanıma sistemlerinde, tanıma başarımının artırılması için, yüzdeki belirgin noktaların bulunması gereklidir. Yüz tanımada kullanılan çoğu öznitelik çıkarımı algoritması her özniteliğin öneminin diğerlerinden bağımsızlığı ve öznitelik çıkarımının, yüzdeki göz, burun gibi organların belirli noktalarında yapılması varsayımına dayanır. Bu varsayımlar tanıma başarımını kısıtlamaktadır. Gökberk ve arkadaşları bu iki varsayımı gevşetmek için altküme seçim yöntemlerinin kullanıldığı çalışmalar yaptı [10].

Gümüş, yaptığı bir çalışmada özyüzler yöntemi için özyüz uzayı boyutunun, yapay sinir ağının saklı katmanındaki nöron sayısı ve eğitim hatasındaki değişimin ve kullanılan Kernel Fonksiyonlarının tanıma performansına etkilerini incelemiştir [11].

Ergezer, çalışmasında gerçek zamanlı yüz tanıma sistemlerinde kullanılabilirliğini test etmek amacı ile ARDB ve ORL veritabanında bu alanda en yaygın algoritmalar olan öz yüzler, sinir ağları, Gabor dalgacık yöntemlerini kullanmıştır [12].

Topkaya, yüz tanıma problemini imgeler yerine video görüntülerinde çalışmıştır. Sistem eğitiminde poz, acı ve dönme kısıdı olmadan yalnızca bir kişinin bulunduğu videolar kullanılmıştır. Tanıma işlemi de yine bir kişinin bulunduğu videolar üzerinden gerçekleşmiştir [13].

Yapılan bir başka çalışma ise haber videolarının etkin erişimi için, haberlerdeki en önemli öğe olan kişilerin videolarda bulunmasına yönelik yüz bulma yöntemlerinin sistematik bir değerlendirmesidir [14].

Referans pikseller, imgedeki pikseller üzerinde kaydırılarak yüz bölgesinin bulunabilmesi için tüm imge taranır. Yüz olmayan yerlerin de taranması sebebiyle özellikle büyük boyutlu imgelerde tarama zamanı çok fazla olabilir. Muhammad ve arkadaşları yaptıkları çalışmada yüz algılama veya tanıma algoritmalarından önce bir ön-tarama kullanarak yüz olmayan veya yüze az benzer olan yerlerin elenerek tarama zamanının azaltılmasını amaçlamıştır [15].

Anbarjafari, çalışmasında histogram eşleştirme tabanlı yeni bir yüz tanıma sistemi önermektedir. Önerilen sistem yüz imgelerine ait farklı renk uzaylarında elde edilmiş histogramları tanıma işleminde öznitelik vektörleri olarak kullanmaktadır [16,17].

Özkaya ve Sağıroğlu tarafından sunulan çalışmada parmak izi, yüz, iris, retina ve el geometrisi gibi biyometrik özellikler arasında olabilecek herhangi bir ilişkinin varlığı tartışılmakta ve kişilerin yalnızca parmak izini kullanarak yüzlerini tahmin etmeye yönelik yapay sinir ağları temelli yeni ve zeki bir sistem tanıtılmaktadır [18].

Özdemir, ön cepheden çekilmiş insan yüzü resimlerinin tanınmasında dalgacık dönüşümü kullanmıştır. Çalışma ile dalgacık dönüşümü yönteminin, resimlere herhangi bir dönüşüm uygulanmaması durumuyla karşılaştırıldığında, doğru tanıma oranını arttırken tanıma süresini azalttığı gözlenmiştir [19,20].

Gabor öznitelikleri tabanlı yaklaşımlar da yüz tanıma probleminin çözümünde kullanılmıştır. Kıraç, Gobor öznitelik vektörlerine En Yakın Komşu Ayrışım Analizi uygulayarak yeni bir yüz tanıma sistemi önermiştir [21].

Yapılan bazı çalışmalarda da yüz tanıma probleminin çözümü için alt uzay temelli yöntemler kullanılmıştır [22,23].

Parelel hesaplama yüksek işlem gücü ve veri boyutu gerektiren işlemlerin uygulanmasında kullanılan etkili yöntemlerden birisidir. Günlük hayatta kullanılan bilgisayarların paralel programlamaya uygun olması ve çeşitli programlama dilleri ve kütüphaneler yöntemin yaygınlaşmasını pozitif etkilemiştir. Paralel hesaplamaların gerçekleştirilmesinde kullanılabilecek yöntemlerden biri de MATLAB programı ve bu programa ait Paralel İşlem Araç Kutusudur (Parallel Processing Toolbox) [24].

Bilgisayar ekran kartları için tasarlanmış Grafik İşleme Birimi (GPU) başlangıçta yüksek performanslı Workstation için en güçlü yonga olarak ortaya çıkmıştır. Şu anda iki veya dört çekirdekli olan işlemci (CPU, Central Processing Unit) mimarilerinin aksine, GPU mimarisi çekirdeği binlerce threadi paralel çalıştırabilen yüzlerce çekirdeğe sahiptir [25].

Kolaylıkla programlanabilen manycore CPU ve GPU'ların varlığı, araştırmacıları bilimsel işlemler için bu muazzam hesaplama gücünü nasıl ortaya çıkarabilecekleri konusunda motive etmektedir. Boyer ve arkadaşları tarafından yapılan bir çalışmada beyaz kan hücrelerinin algılama ve izlenmesi için geliştirilen bir biyoloji sisteminin CUDA ile 200x hızlandırıldığı gösterilmiştir [26].

Tıbbi görüntüleme alanında, zaman-kritik uygulamalarda hızla değişen görüntüler için kayıt yöntemleri gerekir. Muyan-Özçelik ve arkadaşlarının bu alanda yaptıkları bir çalışma ile 55 kat hız (speedup) sağlanmıştır [27].

Toprak araştırmaları başlıca hedef olarak petrol ve gaz endüstrisinin dünyanın yeraltı yapısını inceleyerek petrol ve gazın nerede bulunabileceğini ve çıkarılabileceğini bulmak için yapılmaktadır. Sismik veri işlemede kullanılan algoritmalar hızla gelişmekte ve yapılması gereken hesaplamalar artmaktadır. Bu uygulamaları hızlandırmak için GPU ve NVIDIA CUDA programlama modeli uygulanmıştır [28].

Web sunucularının genellikle verilerin şifrelenmiş transferlerini yönetmesi gerekir. Şifreleme üzerine yapılan bir başka çalışmada da Cuda kullanılmıştır [29].

Yapılan bazı çalışmalarda ise CUDA programlama modelinin avantajları ve verimsizlikleri ortaya konmuştur. Paralel programlama ile sıralı programlama sonuçları karşılaştırılmış [30-33].

Yapılan başka bir uygulamada ise CPU paralel programlama ile GPU paralel programlaması karşılaştırılmıştır [34].

Hazırlanan bu tez çalışmasının ilk bölümünde literatür taraması ve çalışmanın kısa özeti verilmiştir. İkinci bölümde, yüz tanımada kullanılan metodlar anlatılmıştır. Üçüncü bölümde, performansı artırmak için kullanılan paralel programlama ve CUDA hakkında bilgi verilmiş olup dördüncü bölümde ise uygulamanın geliştirilmesi süresince kullanılan yöntemler detaylı olarak anlatılmıştır. Son bölüm olan beşinci bölümde bu çalışma sürecinde elde edilen sonuçlar sunulmaktadır.

BÖLÜM 2

YÜZ TANIMA

Yüz tanıma işlemi yüz görüntülerinin elde edilmesinden sonra iki temel aşama içermektedir. İlk aşama veritabanındaki yüzlerin özelliklerinin çıkarılmasıdır. İkinci aşama ise tanınması istenen kişinin yüz resminden elde edilen özelliklerin ilk aşamada elde edilen veritabanındaki yüz resimlerine ait özelliklerle karşılaştırılmasıdır.

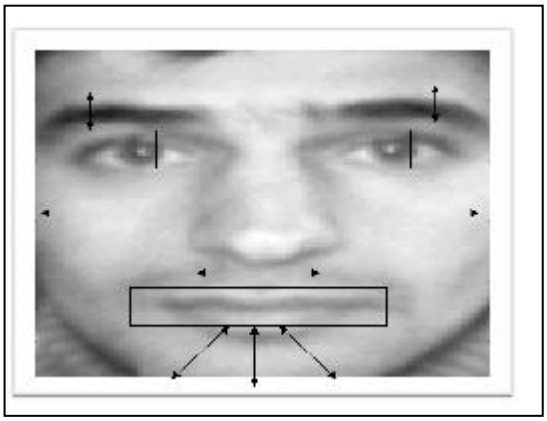
Yüz tanıma sisteminde özellik çıkarmada kullanılan yöntemlerden Temel Bileşen Analizi (Principal Compenent Analysis, PCA) istatiksel yolla özellik çıkartmada akla ilk gelen yöntemdir [35]. Karhunen;Loeve dönüşümü olarak da adlandırılan Temel Bileşen Analizi yönteminin yüz tanımadaki en önemli uygulaması özyüzler yöntemidir [36]. Özellik çıkartımı ve boyut indirgemede kullanılan bir diğer yöntem ise normal dağılımlı olmayan verinin istatiksel olarak bağımsız olmasını sağlayacak şekilde bir doğrusal dönüşüm yapılan Bağımsız Bilesen Analizidir (BBA) [37].

Dalgacık Dönüşümü yöntemi de boyut indirgemede kullanılabilecek bir diğer yöntemdir [38]. Resme ait özet ve detay bilgiler dalgacık dönüşümü ile elde edilebilmektedir.

Ön cepheden çekilen resimlere dalgacık dönüşümü uygulanarak algılama ve sınıflandırma yapılabilmektedir. Yüzün algılanmasının ardından yüz resmine kesikli dalgacık yöntemi uygulanarak yüze ait özet, çapraz , yatay ve dikey katsayılar özellik çıkartmada kullanılabilmektedir [19,39].

Geometrik özelliklere dayalı metodlar, göz, burun, ağız gibi bileşenlerin boyutu ve birbirlerine olan mesafelerinin kişiye özel farklılık göstermesini baz almaktadır.Bu sayede bu farklılıklar sınıflandırma amaçlı kullanılabilmektedir [19]. Şekil 2.1'de

CNNL veritabanındaki bir yüz resmine ait geometrik özelliklerin çıkartılması gösterilmektedir.



Şekil 2.1. CNNL veritabanındaki bir yüz resminde geometrik özelliklerin çıkartılması [19].

Brunelli ve Poggio yüz resminden 35 tane boyut ve mesafe ölçümü almıştır. Yüze ait özellik vektörünü belirten bu ölçümlerde çıkarılan bazı geometrik şekiller; kaş kalınlığı, göz merkezinin dikey konumu, burnun genişliği ve dikey konumu, ağzın dikey konumu, alt ve üst dudağının genişliği ve yüksekliği, çene şekli, yüzün burna olan genişliği ve burun ucu ile gözler arasındaki yüz genişliği olarak sıralanabilir [39].

Yüzün önemli görülen bölümleri ile bu bölümler arasındaki ilişkilerin incelendiği geometrik tabanlı özellik çıkartım yöntemlerine Aktif Şekil Modeli (AŞM) yöntemi örnek gösterilebilir [37].

Ayrıca Gabor süzgeçleri ve yapay sinir ağları da özellik çıkarımında kullanılan yaygın yöntemlerdir [10,12].

2.1. ÖZYÜZLER YÖNTEMİ (PCA)

İlk olarak Kirby ve Sirovich tarafından yüz resmini görüntülemek için kullanılan özyüzler yöntemi veriyi alt uzayda ifade ederek boyut indirgemeyi sağlayan PCA istatiksel yöntemine dayanmaktadır [19,40,41].

Turk ve Pentland, özyüzler yöntemini ilk kez tam otomatik bir siteme dönüştürerek özyüzlerin hesaplanması ve bir yüz resminin özyüzler kullanılarak sınıflandırılması hakkında detaylı bir çalışma sunmuşlardır [36].

2.1.1. Özyüzlerin Hesaplanması

PCA yöntemiyle yüz resimlerinden elde edilen koveryans matrisinin öz vektörleri elde edilerek yüz görüntülerinin yüz uzayı olarak adlandırılan alt uzayı elde edilir. Bu yöntemle elde edilen vektörlerin görünüm olarak yüze benzemesi ve yüz resimlerini işaret etmesi sebebiyle özyüzler olarak isimlendirilmiştir.

Özyüzler yöntemini detaylandırmak için eğitim setindeki her bir yüz resminin NxN boyutununda olduğunu varsayarsak, bir yüz resmi N^2 boyutunda Γ ile ifade edilen bir vektör olarak düşünülebilir.

M adet yüz resmi bulunan yüz görüntülerine ait bir eğitim setini $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$ şeklinde ifade ettiğimizde setin ortalaması

$$\Psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n \tag{2.1}$$

(2.1) denkleminde olduğu gibi tanımlanır. Her yüz ortalamadan

$$\Phi_i = \Gamma_i - \Psi \tag{2.2}$$

(2.2) denkleminde gösterilen vektör ile ayrılır.

Örnek eğitim setindeki bir kişiye ait yüz resimleri Şekil 2.2'de ve eğitim setindeki her kişiye ait 8 resim olmak üzere 30 kişinin ortalama yüz resmi ψ Şekil 2.3'de gösterilmektedir.

Şekil 2.2. Örnek eğitim setindeki bir kişiye ait resimler.

Şekil 2.3. Sisteme verilen 240 adet yüz resminden çıkarılan ortalama yüz resmi.

Eğitim setindeki her yüzün ortalamadan farkını ifade eden Φ vektörler setinin çok büyük boyutlarda olması sebebiyle, verinin dağılımını en iyi ifade edecek bir M orthonormal vektör setini, u_n , aramak için, bu çok büyük vektörler seti temel bileşen analizine tabi tutulur. Maksimum olacak şekilde k'nıncı u_k vektörü (2.3) ve (2.4) denklemlerinde olduğu gibi seçilir.

$$\lambda_k = \frac{1}{M}\sum_{n=1}^{M}(u_k^T \Phi_n)^2 \tag{2.3}$$

$$u_l^T u_k = \delta_{lk} = \begin{cases} 1, & if \quad l = k \\ 0, & otherwise \end{cases} \tag{2.4}$$

Burada u_k özvektörleri ve λ_k ise öz değerleri ifade etmek üzere (2.5) denklemindeki koveryans matrisine ait özvektörleri ve özdeğerleri göstermektedirler. .

$$C = \frac{1}{M}\sum_{n=1}^{M}(\Phi_n \Phi_n^T) = AA^T \tag{2.5}$$

Burada, matris $A = [\Phi_1 \Phi_2 ... \Phi_M]$ ve C koveryans matrisi ise N^2 x N^2 boyutlu bir simetrik matristir. C matrisi için N^2 öz vektörlerini ve öz değerlerini belirlemenin zorluğu, sayısal olarak fizibil bir metot ihtiyacını ortaya çıkarmaktadır.

Eğer kullanılan resim sayısı ,resmin boyutundan küçükse ($M < N^2$) N^2 yerine sadece $M - 1$ anlamlı öz vektörleri olacaktır. Bu durumda, N^2 boyutlu öz vektörleri önce bir M x M matrisinin öz vektörlerini çözüp, örneğin 256x256 boyutlu 16 resim için 16,384 x 16,384 matrisi yerine 16 x 16 matrisini çözerek, sonra yüz görüntülerinin Φ_i uygun lineer kombinasyonlarını alarak çözülebilir. Eğer v_i'yi A^TA'nın özvektörleri

$$A^T A v_i = \mu_i v_i \tag{2.6}$$

olarak düşünürsek ve eşitliğin iki tarafını da A ile çarparsak

$$AA^T A v_i = \mu_i A v_i \tag{2.7}$$

elde edilir. Burada Av_i'nin, $C=AA^T$ 'nin özvektörleri olduğunu görülür.

Bu analizin sonucunda $L_{mn} = \Phi^T_m \Phi_n$ eşitliği ile M X M boyutundaki matris $L = A^T A$ olduğu görülmektedir ve L'ye ait M tane özvektörü kapsayan v_i bulunur. Bu vektörler ile M adet yüz resmini içeren eğitim setinin lineer kombinasyonları, u_i özyüzlerini oluşturur.

$$u_l = \sum_{k=1}^{M}(u_{lk}\Phi_k) \quad l = 1,....., M \tag{2.8}$$

Eğitim setindeki resimlerin piksel sayısı (N^2) yerine resim sayısının (M) kullanılması hesaplamaları büyük ölçüde azaltır. Burada eğitim setindeki resim sayısının resim pixel sayısından nispeten küçük olacağı ($M<<N^2$) düşünülmektedir.Bu analizle elde edilen özdeğerler, görüntüler arasındaki değişikliği ifade etmede ki güçlerine göre, özvektörleri sıralama imkânı verir.

Bu aşamadan sonra L matrisinin özdeğer ve özvektörlerinin bulunması ve değerlendirilmesi önem taşımaktadır. $AX = \lambda X$ durumunda NxN boyutlu bir matris için A'nın λ özdeğerine ait bir X özvektörüne sahip olduğu söylenebilir. Ancak bu sahipliğin geçerli olması da

$$\det|A - \lambda I| = 0 \tag{2.9}$$

eşitliğinin geçerliliğine bağlıdır.

Büyük boyutlu matrislerde özdeğer ve özvektör hesaplama için kullanılan bazı metodlar geliştirilmiştir. Özvektör ve özdeğer hesaplamanın programlanmasında kullanılan yöntemlerden biri de QL metodudur [43]. İlk olarak, matrise Householder algoritmasi uygulanarak üç köşegenli şekle dönüştürülür. Householder algoritması, her transformasyonun tüm sütunun ve tüm ilgili sıranın ilgili kısmını sıfırladığı N-2 dikgen transformasyonlarla bir N x N simetrik A matrisini üç köşegenli şekle indirgemektedir. Bu dönüşümün ardından QL transformasyonları uygulanabilmektedir.

$$A = Q \cdot L \tag{2.10}$$

şeklinde ayrıştırılan matriste L alt üçgeni, Q ise dikgeni temsil etmektedir. QL algoritmasi için iterasyon başı iş yükü, yasaklayıcı bir matriste $O(N^3)$, üç köşegenli bir matris için O(N), ve Hessenberg matrisi için $O(N^2)$'dir [43].

2.1.2. Bir Yüz Görüntüsünü Sınıflandırmak İçin Özyüzlerin Kullanılması

Yüz görüntülerini ifade etmek amacıyla, L matrisinin öz vektörlerinden özyüz görüntüleri hesaplanmaktadır. Kafkas erkeklerinin M=115 görüntü takımıyla yapılan bir çalışmada yüz görüntülerinin çok iyi bir tarifi için, 40 özyüzün yeterli olduğu tespit edilmiştir [42].

Kontrollü koşullar altında yüz görüntüleri tanımlamak için özyüzler oldukça yeterli görünmesi sebebiyle yüz tanıma probleminde bir araç olarak kullanılmaktadır. Bu durumda yüz tanıma işlemi bir görüntü tanıma görevi haline gelmektedir. L matrisinin en önemli M' adet özvektörü en büyük ilgili öz değerlere sahip olanlar olarak seçilirler. Tanınması istenen yeni yüz görüntüsü için (Γ), özyüz bileşenleri elde edilerek tanıma işlemi gerçekleştirilir.

İlk aşamada Γ_T vektörünün eğitim setindeki yüzlerden elde edilen ortalama yüz bilgisinden farkı

$$\Phi_t = \Gamma_t - \Psi \tag{2.11}$$

denklemindeki gibi hesaplanmaktadır. Elde Φ_t fark vektörü $U = [u_1, u_2, u_3, ..., u_N]^T$ özyüz matrisi ile çarpılarak w_T özellik vektörü

$$w_t = u_t^T \Phi_t \qquad t = 1,2,....M' \tag{2.12}$$

denklemi ile elde edilir.

Hesaplanan w_T özellik vektörünün $W = [w_1, w_2, w_3, ..., w_N]$ matrisini olusturan w_i özellik vektörlerine yakınlığı veya uzaklığı tanınması istenen yüzün, o an özellik vektörü ile kıyaslaması yapılan kayıtlı yüze benzerliğini gösterir. Yapılan kıyaslama iki özellik vektörü arasındaki öklid mesafesine dayanmaktadır. Elde edilen sonuç, kullanıcı tarafından tanımlanan eşikten küçük ise yüz resmi bilinen olarak büyük ise

bilinmeyen olarak sınıflandırılır. Sistemin her kişinin tek resimle ifade edildiği durumlarda tanınan yüz olarak üreteceği yanıt, eşik değerinden küçük olmak şartıyla

$$d_i = \sum_{j=1}^{M} \left| w_t(j) - w_i(j) \right| \qquad i = 1,2,...N \tag{2.13}$$

denkleminde hesaplanan d_i değerlerinden en küçüğünü üreten i. resimdir. Öklid mesafesi eşik değerinden büyük çıkan yüz görüntüleri , sisteme dahil edilerek özyüzlerin yeniden hesaplanması ile sonraki kullanımlar için sistemin eğitilmesine imkan vermektedir.

2.2. DALGACIK DÖNÜŞÜMÜ

Bir sinyale ait bilgiyi farklı frekans bileşenlerina ayırarak, ölçekleme ile oluşturulan bileşenler üzerinde çalışan matemetiksel fonksiyolara dalgacıklar denir. Dalgacıklarda dikkat çeken husus, ölçekleme ve ölçeğe göre analizdir [44].

Dalgacık dönüşümü, sinyalin yüksek frekanslı ve alçak frekanslı bileşenlerini elde etmek amacıyla örnek fonksiyonun ölçekleme ve ötelemeleriyle oluşturulan taban fonksiyonlarına, sinyalin izdüşümünün alınmasıyla oluşturulur. Dalgacık dönüşümü değişken zaman-frekans çözünürlüğüne sahip olduğundan durağan sinyallerin analizinden çok ayrık zamanlı sinyalleri incelemede kullanılır. Dalgacık dönüşümleri için sonsuz bir kümeden bahsedilebilir. Farklı dalgacık aileleri ve tipleri vardır [44].

Dalgacık dönüşümleri analiz yöntemlerine göre ikiye ayrılmaktadır, sayısal uygulamalarda kullanılan Sürekli Dalgacık Dönüşümü (Continuous Wavelet Transform – CWT) ve anolog uygulamalarda kullanılan Ayrık Dalgacık Dönüşümü (Discrete Wavelet Transform–DWT) [45].

2.2.1. Sürekli Dalgacık Dönüşümü

Sürekli Dalgacık Dönüşümü (CWT), bir ana dalgacık $\psi(\tau, s)$ fonksiyonunun ötelenmesi ve genişletilmesi ile oluşturulan fonksiyonlarla sinyalin toplam zamanının çarpılması şeklinde tanımlanır [44].

$$CWT_x^w(\tau, s) = \frac{1}{\sqrt{|s|}} \int_s x(t)\varphi\left(\frac{t-\tau}{s}\right) d_t \tag{2.14}$$

$$\varphi_{a,b}(t) = \frac{1}{\sqrt{a}} \varphi\left(\frac{t-b}{a}\right) \tag{2.15}$$

Burada, x(t) sinyalinin y dalgacığı ile çarpılması ifade edilmektedir. Ayrıca $a \in R^+$ ve $b \in R$'dir. Dalgacık s ölçeğine göre t zamanında kaydırılır. τ parametresi ise her bir adım için dönüşüm parametresidir [19,44].

2.2.2. Ayrık Dalgacık Dönüşümü

CWT'de her ölçek için dalgacık katsayılarının hesaplanması, sinyal içindeki gereksiz bilgilerin saklanmasına sebep olduğu gibi analiz işlemini de gereksiz yere uzatmaktadır. İstenmeyen verilerin ayıklanması ve yapılan analizin etkili ve hızlı olması için, seçilecek ölçek ve pozisyonlar ikinin katları şeklinde düzenlenebilir. Bu şekilde yapılan dönüşüm Ayrık Dalgacık Dönüşümü (DWT) olarak adlandırılır [44].

DWT ayrık olarak isimlendirilmesine rağmen dönüşüm sürekli bir fonksiyondur. CWT'deki denklemde $a=2^{-j}$ ve $b = k.2^{-j}$ olarak ölçek katsayısı ve öteleme katsayısı ikinin katları şeklinde düzenlendiğinde ana dalgacık

$$\varphi_{a,b}(t) = \frac{1}{\sqrt{a}} \varphi\left(\frac{t-b}{a}\right) = \frac{1}{\sqrt{2^{-j}}} \varphi\left(\frac{t-k2^{-j}}{2^{-j}}\right) \tag{2.16}$$

$$\varphi_{j,k}(t) = 2^{j/2} \varphi\left(2^{-j}t - k\right) \qquad j,k \in Z \tag{2.17}$$

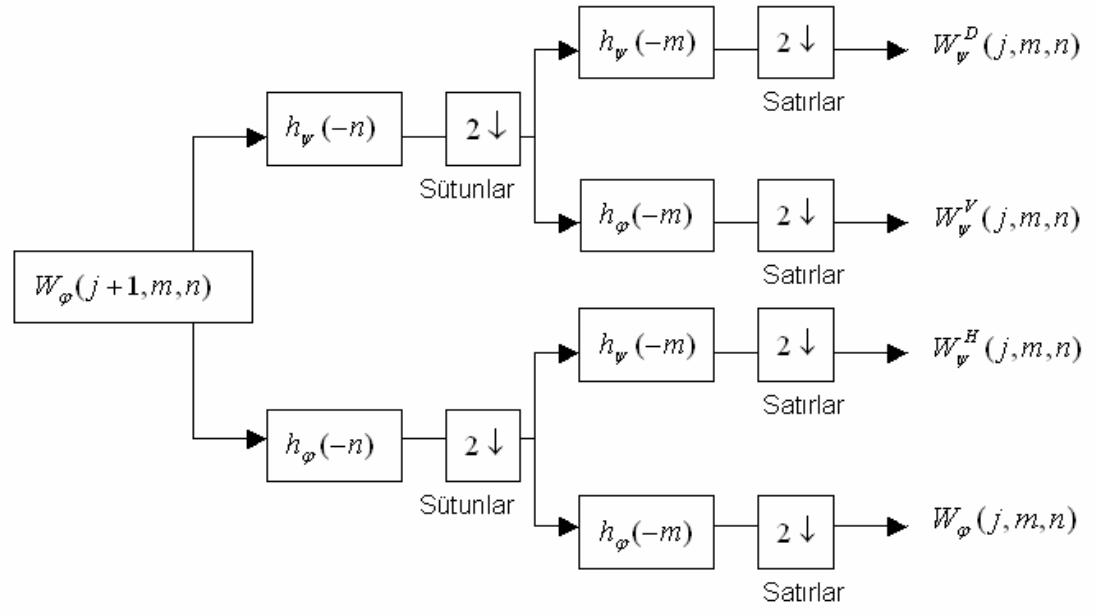
haline gelir. Burada j parametresinin dalgacığın ölçeğini ifade ettiğininden j'nin değeri arttıkça ölçeğin azalması söz konusudur.Seviye seviye yapılan dönüşüm j parametresi ile ters orantılı olarak ilerledikçe her seviyede ölçek iki kat artmaktadır.Böylece hesaplanacak katsayı miktarı da yarıya düşmektedir [45].

2.2.3. İki Boyutlu Ayrık Dalgacık Dönüşümü

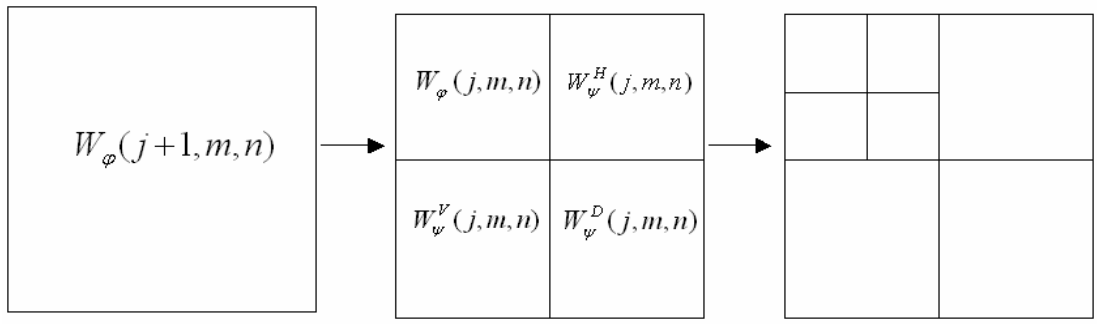
Ayrık Dalgacık Dönüşümü bir fonksiyonun değişik çözünürlüklere ayrılarak dalgacık katsayılarının belirlenmesi ile elde edilir. Dalgacık dönüşümünde bir adet alçak ve bir adet yüksek geçiren filtre bulunmaktadır.

İki boyutlu resimlerde dalgacık dönüşümü ise imgenin tekrar tekrar alçak ve yüksek geçiren filtreden geçirilmesi ile gerçekleştirilir. Her filtreleme işlemi sonrasında imgenin düşük çözünürlükte özet ve detay katsayıları elde edilir İmge tek piksel kalana dek filtreleme yapılabilir.

İki boyutlu ayrık dalgacık dönüşümü tek boyutlu ayrık dalgacık dönüşümünün iki boyuta uygulanışı olarak düşünülürse, her filtreleme işlemi sonrasında imge özet, yatay, dikey ve diyagonal katsayılar şeklinde 4 alt banda ayrılır. Özet dalgacık katsayılarını imgenin düşük çözünürlüklü bir kopyası olarak, yatay , dikey ve diyagonal katsayıları ise imgenin yatay, dikey ve diyagonal özelliklerini gösteren birer imge olarak düşünülebilir. Herhangi bir çözünürlükteki filtreleme işlemi aşağıdaki şekilde olduğu gibidir [19].



(a)



(b)

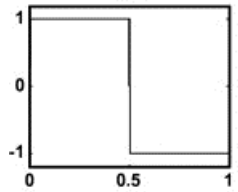
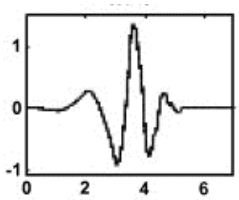
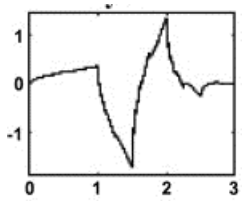
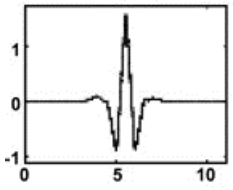
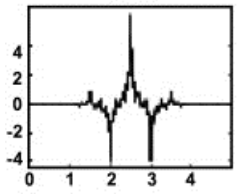
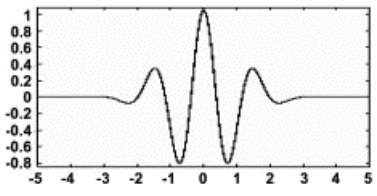
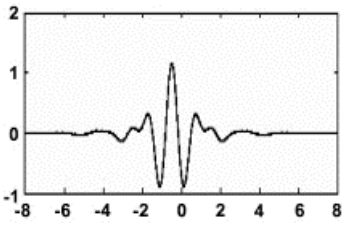
Şekil 2.4. Filtreleme İşlemi [19]. a) Alçak ve yüksek geçiren filtreler kullanılarak bir resmin herhangi bir *j* çözünürlüğündeki dalgacık katsayılarının bulunması. b) Elde edilen katsayıların özet, yatay, dikey ve diyagonal katsayılar olarak gösterimi**.**

Şekil 2.4'de gösterilen W_φ ,W_ψ^H ,$W_\Psi{}^D$,$W_\Psi{}^V$ değerleri resmin *j* çözünürlüğündeki özet, yatay, diyagonal ve dikey dalgacık katsayılarına karşılık gelmektedir.

2.2.3. Yaygın Dalgacık Ailesi Çeşitleri

Kullanım alanına göre dalgacık ailesi çeşitlerinden biri seçilebilir. En yaygın kullanılan ve Matlab wavemenude de seçenek olarak sunulan dalgacık ailesi çeşitlerinin bir kısmı Çizelge 2.1'de verilmiştir [19,45,46].

Çizelge 2.1. Dalgacık ailesi çeşitleri.

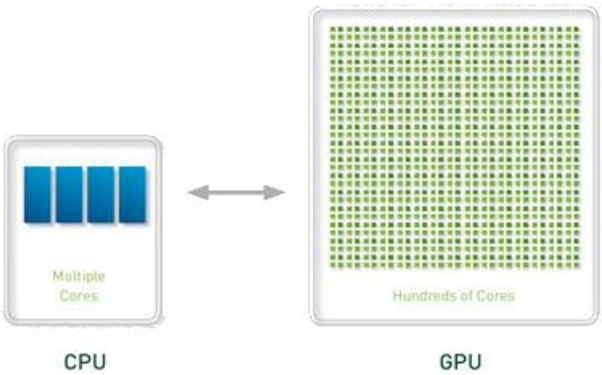
Kısa İsmi	İsmi	Şekli
haar	Haar Dalgacığı	
db	Daubechies Dalgacığı	
sym	Symlets Dalgacığı	
coif	Coiflets Dalgacığı	
bior	Biorthogonal Dalgacığı	
rbio	Reverse Biorthogonal Dalgacığı	
meyr	Meyer Dalgacığı	

BÖLÜM 3

CUDA İLE PARALEL PROGRAMLAMA

Bir ağ üzerinde birleştirilmiş ve birbiriyle iletişim halinde olan bilgisayarlar birlikte tek bir bilgisayar gibi kullanılabilmektedir. Böylelikle ağ üzerinde dağıtık halde bulunan bilgisayarların bellekleri ortak olarak kullanılabildiği gibi yapılan işlem tüm bilgisayarlara paylaştırılarak işlem sonucu normalden çok daha kısa sürede elde edebilir. Ayrıca bilgisayar mimarisinde çok çekirdekli yapıya geçilmiştir. Böylelikle bilgisayarlar aynı anda birden fazla işi yapabilecek duruma gelmiştir. Donanım olarak bu mimarinin kullanılabilmesi için paralel programlamaya uyumlu yazılımlar gerekmektedir. Sadece paralel programlama ile hazırlanmış programlar birden fazla çekirdeği aynı program için kullanma imkanı sunmaktadır. Böylelikle program parçalara ayrılarak çekirdeklerde eş zamanlı olarak çalıştırılır. Bu yöntemle günlerce sürecek büyük hesaplamaların yapıldığı programlar saatler içerisinde halledilebilir hale gelmiştir.

Son yıllarda kişisel bilgisayarlar üzerinde de paralel sistemler gelişmeye başlamıştır. Grafik kartlarının CPU'lara göre daha hızlı bir gelişme göstermesi sonucu GPGPU (General-Purpose Computing on Graphics Processing Units) paralel programlamada önemli bir çalışma alanı halen gelmiştir.



Şekil 3.1. CPU ve GPU arasındaki çekirdek karşılaştırması [47].

Şekil 3.1'de de görüleceği üzere grafik kartların çekirdek sayısında işlemci cekirdek sayısına göre muazzam bir artış söz konusudur. Bu GPUların muazzam paralel mimarisi yüksek işlem performansı sağlamaktadır [47]. Grafik kartların genel amaçlı programlanmasında üç temel teknoloji bulunmaktadır [48]. Advanced Micro Devices (AMD) BROOK+ teknolojisini geliştirmiştir [49]. ATI ekran kartları üzerinde parelel programlama yapılabilmesine olanak sağlanmaktadır. Open Computing Language (OpenCL), Khronos Group tarafından geliştirilmektedir [50]. OpenCL ise hem ATI hemde NVIDIA ekran kartlarında parelel programlamayı desteklemektedir. Böylece heterojen sistemler üzerinde paralel programlama yapılabilmektedir. Compute Unified Device Architecture (CUDA) ise, NVIDIA'nın sunduğu genel amaçlı bir paralel programlama mimarisidir. CUDA, NVIDIA tarafından geliştirilmiştir ve çalışması için NVIDIA ekran kartlarına ihtiyaç vardır.

3.1. CUDA

CUDA, GPU gücünü kullanarak bilgisayar performansında dramatik bir artış sağlayan grafik kartı üreticisi NVIDIA tarafından 2006 yılının sonunda yayınlanmış genel amaçlı bir paralel veri işleme mimarisidir [51].

3.1.1. Programlama Modeli

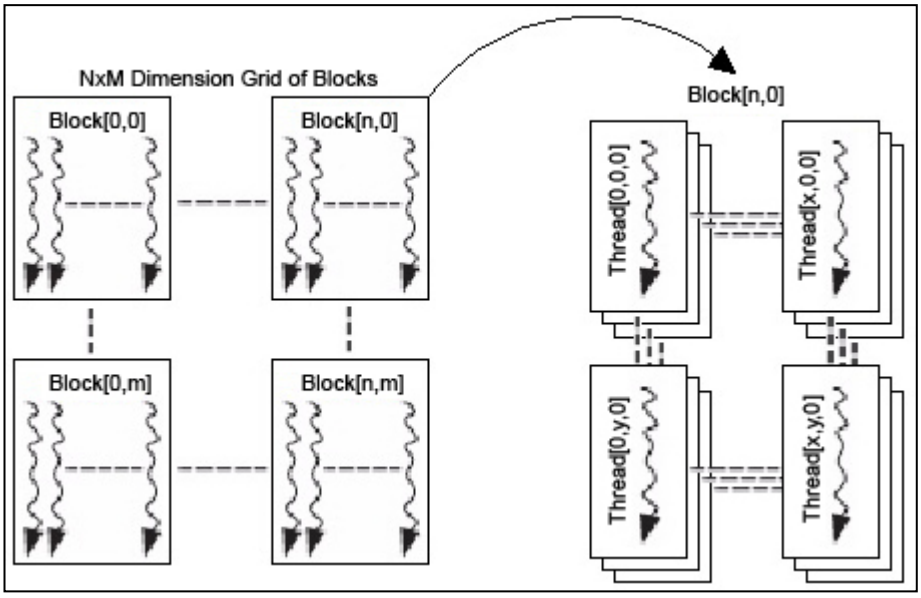
GPU mimarisi donanıma göre farklı özellikler göstermesine ve CUDA ile geliştirilen uygulamaları etkilemesine rağmen, programlama modeli açısından temel çalışma prensipleri aynıdır.

CUDA; Fortran, OpenCL ve DirectCompute gibi farklı dil ve programlama arabirimlerini belirli ek komut ve kısıtlamalarla desteklemektedir [51]. CUDA mimarisini programlamada kullanılabilecek dillerden biri de Cuda C'dir. CUDA C ile programlamayı kolaylaştırmak için kullanılabilecek hazır kütüphaneler geliştirilmiştir [52, 53]. Ayrıca Nawata ve Suda, C kodunu hiçbir direktif olmadan Cuda C koduna çeviren, Auto Parallelizing Translator from C to CUDA(APTCC) çalışması ile karmaşık GPU mimarisini gözardı ederek algoritmaya odaklanmıştır [54].

CUDA C, C programlama dilinden türetilmiştir. Programcının C fonksiyonlarına benzer biçimde kodlamasına olanak sağlanmaktadır. Paralel şekilde çalıştırılan Cuda iş parçaçıkları oluşturularak uygulamada gerekli yerlerde çağrılmaktadır.

Uygulama, CPU'nun kontrolünde çalıştırılır. Cuda paralel programlama modelinde, ana işlemci (CPU) host olarak isimlendirilirken, ana işlemciye yardımcı olan GPU device olarak adlandırılır. İşlenecek verinin host-device, device-host aktarımı ve GPU'ların yönetimi CPU tarafından gerçekleştirilir. CUDA, Şekil 3.2.'de görüldüğü üzere uygulamaların CPU ve GPU arasında karışık olarak çalıştırılmasına imkan sağlamaktadır.

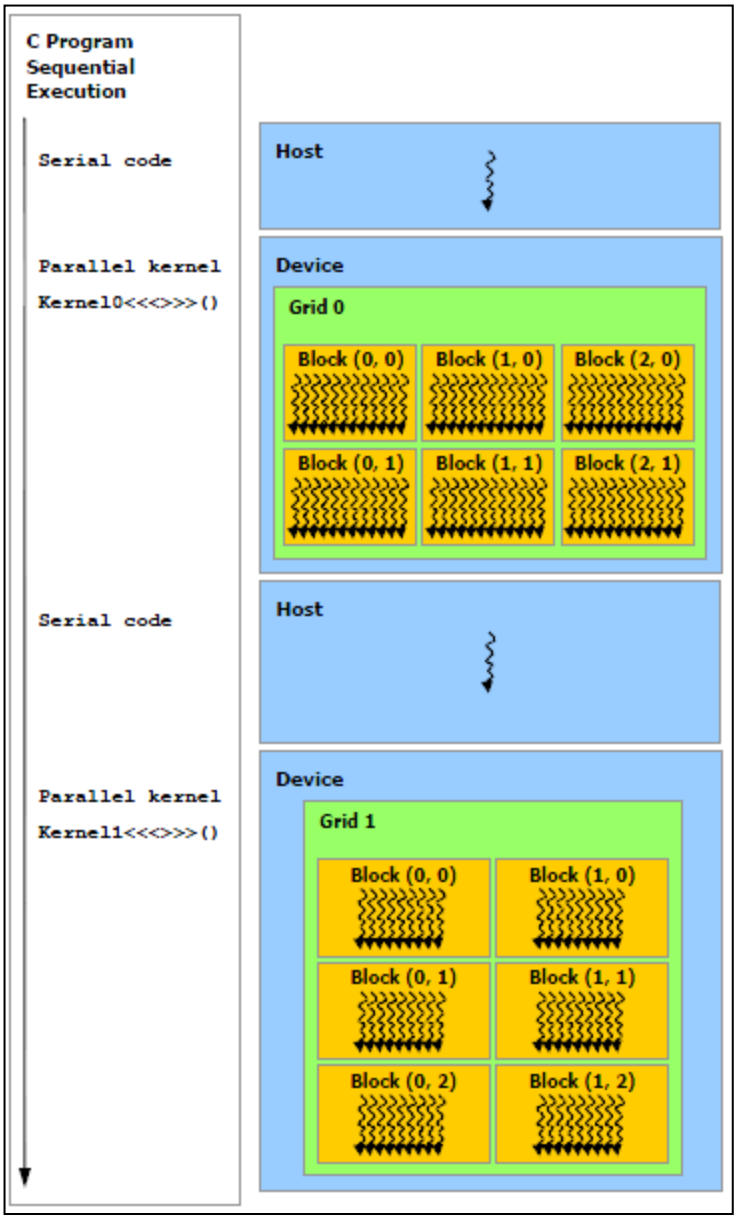
Koşut işlem yapılması istendiğinde, kernel olarak isimlendirilen, özel tanımlanmış bir C işlevi ile aygıta bildirilmektedir. *KernelName <<<gridsize,blocksize>>>(...)* şeklinde koşut işlev çağrılarak ,çalıştırılacak topoloji belirtilmektedir.



Şekil 3. 2. CUDA iş parçacığı diagramı.

Koşut programlama için işlevler, ızgara (*grid*), blok (*block*) ve iş parçacığı (*thread*) olarak isimlendirilmiş üç yapılı bir sistem ile gerçekleştirilmektedir. Şekil 3.2'de Cuda iş parçacıklarına ait diagram görünmektedir. Thread adı verilen iş parçacıkları 1,2 veya 3 boyutlu diziler şeklinde blokları oluşturur. Bir veya iki boyutlu blok dizileri

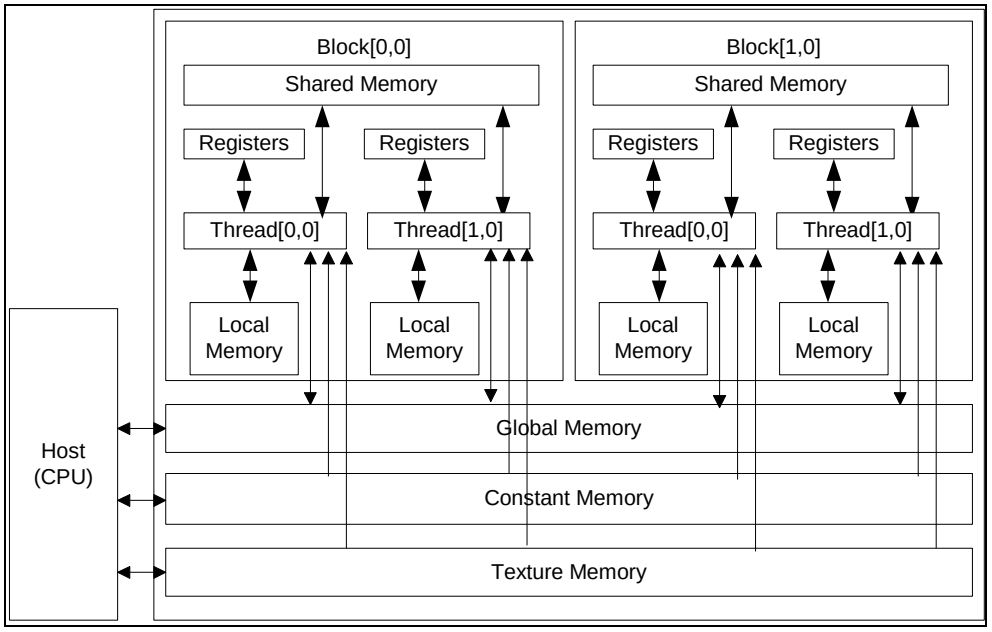
ise ızgarayı meydana getirmektedir. Şekil 3.3'de CPU ve GPU üzerinde karışık programlanmış bir uygulamanın çalışma şeması görünmektedir.



Şekil 3.3. CPU ve GPU üzerinde karışık programlama [51].

3.1.2. Bellek Çeşitleri

Nvidia ekran kartlarında global, paylaşımlı ve yerel bellek olmak üzere üç farklı bellek türü tanımlanmıştır. Şekil 3.4'de host ,device ,blok ve iş parçacıklarının belleklere erişim modeli görülmektedir.



Şekil 3.4. CUDA device memory space diagram [48].

Global bellek, ekran kartının RAM'idir. Izgara, blok ve iş parçaçıkları tarafından erişilebilmektedir. Erişim hızı yavaştır. Her bloğun okuma ve yazma yapılabildiği bu bellek host ile device arasında veri aktarılmasında ve bloklar arası iletişimde kullanılmaktadır.

Paylaşımlı bellek, global bellekten hızlıdır fakat sadece aynı blok içindeki iş parçaçıkları tarafından erişilebilmektedir. Gerektiğinde blok içinde iletişimde kullanılabilmektedir.

Cuda programlama modelinde, her thread için sadece kendisine ait bir yerel bellek vardır. Erişim hızı yavaş olduğu için ekran kartlarında hızlı erişim yapılabilen çok sayıda yazmaç (register) bulunmaktadır.

Ekran kartlarında genel amaçlı programlama dışında grafik uygulamalarında kullanılmak üzere sabit (*constant*) ve doku (*texture*) bellek olarak isimlendirilen iki çeşit bellek daha bulunmaktadır. Host constant and texture belleğe yazabilir ve okuyabilirken GPU daki threadlerin sadece okuma izni bulunmaktadır.

3.1.3. CUDA C Dili

Cuda C ile yazılan işlevler için işlevin host ya da device da çalışacağını belirtmek üzere işlev niteleyicileri kullanılmaktadır.

__device__ ,önüne getirildiği işlevin device da yani ekran kartı üzerinde çalışacağını ve sadece deviceda işlenen yordamlar tarafından çağırılabileceğini belirtmektedir.
__global__ , önüne getirildiği işlevin device üzerinde çalışacağını ve ancak host yani işlemcide çalışan yordamlar tarafından çağırılabileceğini belirtmektedir.
__host__, önüne getirildiği işlevin işlemci üzerinde çalışacağını ve işlemcide çalışan yordamlar tarafından çağırılabileceğini belirtmektedir.

Bir işlev için niteleyici tanımlanmadığında CUDA işlevin __host__ niteleyicisine sahip olduğunu varsayar ve işlemci üzerinde çalıştırır [51].

Kod 1. Aygıt Kodu

```
1.  __global__ void calculateA(float* d_A,float* d_face,float* d_meanface, int facecount,int pixel)
2.  {
3.      int i=threadIdx.x;
4.      int j = blockIdx.x;
5.      int index=i+j*facecount;
6.      d_A[index] = d_face[index] - d_meanface[j];
7.  }
```

Şekil 3.5. Aygıt kodu.

ThreadID bir bloktaki threadin indeksini verir. 1, 2 veya 3 boyutlu olabilir. Tek boyutlu bir blokta threadID dizinin indexi iken, iki boyutlu bir blok *(Dx, Dy) için*, thread indexi *(x + y Dx)* şeklinde; üç boyutlu blok*(Dx, Dy, Dz)* içinse, thread indexi *(x + y Dx + z Dx Dy*) şeklinde hesaplanabilir.

Bir grideki blok indexi ise blockID ile ifade edilir. 1 veya 2 boyutlu olabilir. blockIdx.x ve blockIdx.y şeklinde ilgili değerlere ulaşılabilir. Bir blocktaki thread sayısı blockDim ile 1, 2 veya 3 boyutlu ifade edilebilir . Compute Capacity 1.x olan

GPU lar için bu değer maximum 512 iken Compute Capacity 2.x olan GPU lar için maximum 1024 dür. Bir grideki block sayısı ise gridDim terimiyle ile 1 veya 2 boyutlu olarak ifade edilebilir. Gridde maximum 65535 block bulunabilir.

Kod 2. Yer ayırma ve aygıta kopyalama

```
1.  unsigned int size_face = faceCount * pixel;
2.  unsigned int mem_size_face = sizeof(float) * size_face;
3.  unsigned int mem_size_meanface = sizeof(float) *pixel;
4.  float* d_A;
5.  float* d_face;
6.  float* d_meanface;
7.  cutilSafeCall(cudaMalloc((void**) &d_A, mem_size_face));
8.  cutilSafeCall(cudaMalloc((void**) &d_face, mem_size_face));
9.  cutilSafeCall(cudaMalloc((void**) &d_meanface, mem_size_meanface));
10. cutilSafeCall(cudaMemcpy(d_face,face, mem_size_face, cudaMemcpyHostToDevice) );
11. cutilSafeCall(cudaMemcpy(d_meanface,meanface,mem_size_meanface,
    cudaMemcpyHostToDevice) );
```

Şekil 3.6. Yer ayırma ve aygıta kopyalama.

Host aynı zamanda GPU belleğini de yönetir. Device tarafında kullanılacak veriler için host, önce alan ayırarak verileri device belleğine kopyalar.

Kod 3. İşlev çağırma ve hosta kopyalama

```
1.  int block =pixel;
2.  int thread=faceCount;
3.  calculateA <<< block, thread >>>(d_A,d_face,d_meanface,faceCount,pixel);
4.  cudaThreadSynchronize();
5.  cutilSafeCall(cudaMemcpy(A, d_A, mem_size_face,cudaMemcpyDeviceToHost) );
6.  cutilSafeCall(cudaFree(d_A));
7.  cutilSafeCall(cudaFree(d_face));
8.  cutilSafeCall(cudaFree(d_meanface));
```

Şekil 3.7. İşlev çağırma ve hosta kopyalama.

Host, grid ve block size bilgilerinide belirleyerek kernelın kaç kez çalışacağını ayarlamalıdır. Kernel çalışmayı bitirdiğinde host,device belleğindeki veriyi host belleğine aktararak,device üzerinde ayrılan alanı serbest bırakmalıdır.

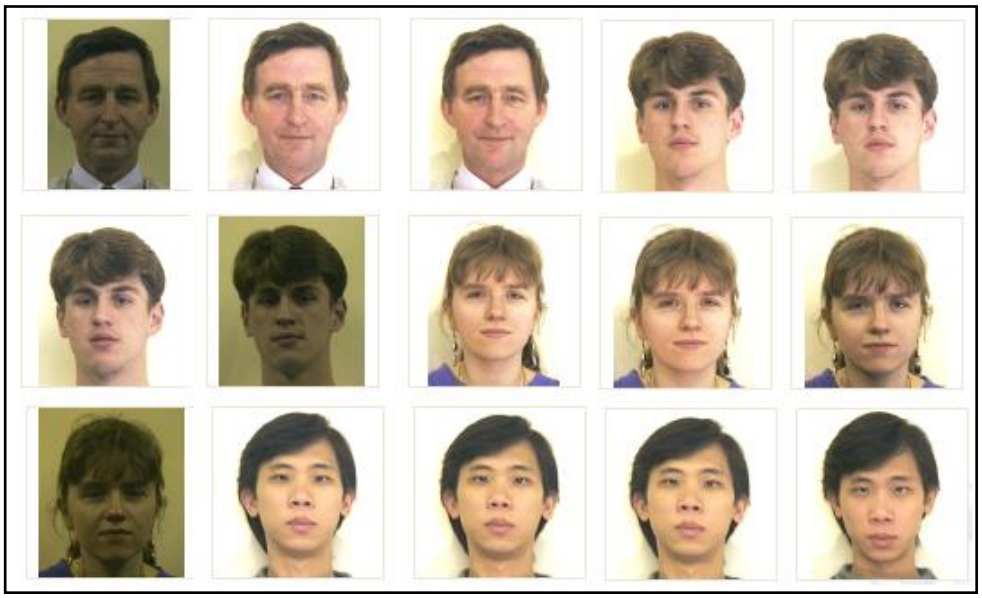
Host tarfından bir kez *KernelName <<<gridsize,blocksize>>>(...)* ifedesi şeklinde çağrılan kernel device üzerinde paralel işlenen bir dizi thread tarafından çalıştırılır. Bütün threadler aynı kodu çalıştırmasına rağmen kendilerine ait unique threadIdleri sayesinde alınacak kararlar ve yapılacak hesaplamalarda erişilecek bellek adresi kontrolü sağlanmış olur.

BÖLÜM 4

YÜZ TANIMA SİSTEMİNİN GERÇEKLEŞTİRİLMESİ

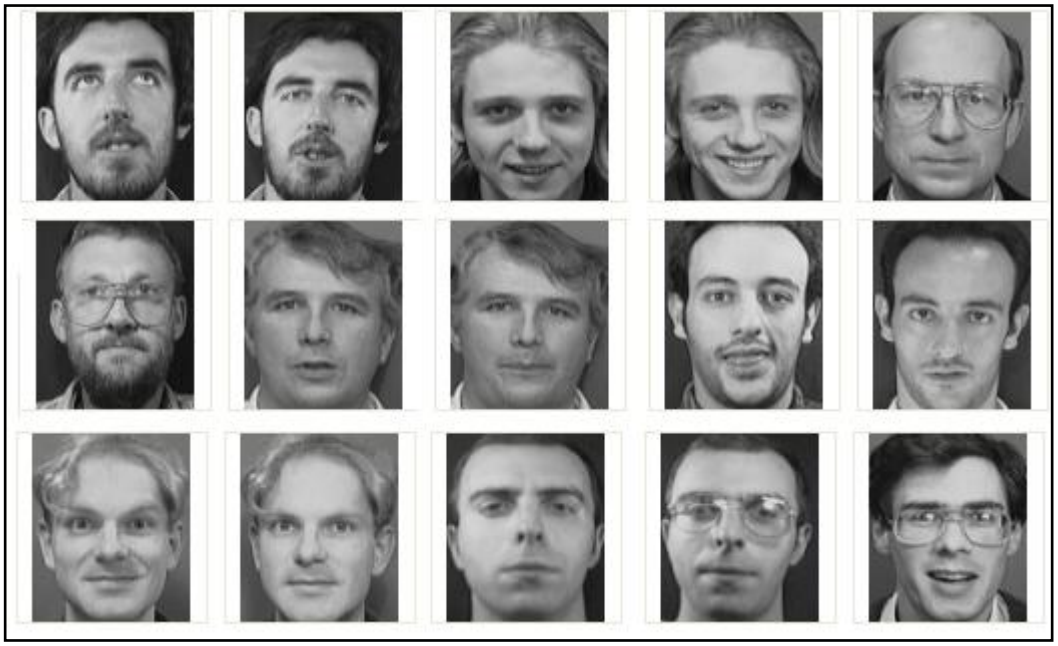
Tez kapsamında paralel programlama ve özyüzler yöntemi ile geliştirilen Yüz Tanıma Sisteminde, Visual Studio .NET ortamında C# programlama dili ve Cuda C kullanılmıştır. Ayrıca 2 boyutlu dalgacık dönüşümü yöntemi ile Matlab'da yüz tanıma sistemi uygulanmıştır.

Bu çalışmada ORL veritabanı ve PICS veritabanı kullanılmıştır [55,56]. ORL veritabanında her kişiye ait 10 olmak üzere 40 kişi için toplam 400 yüz resmi bulunmaktadır. Bu resimler normal pozdan gülme ve değişik poz açıları gibi farklı pozları kapsar. Resimler 92x112 çözünürlüğündedir. PICS veritabanında ise 288x384 boyutlarında her kişi için 7 resim olmak üzere 66 kişiye ait resimler kullanılmıştır. Şekil 4.1.a'da PICS veritabanından ve Şekil 4.1.b'da ORL veritabanından seçilmiş örnek resimler görülmektedir.



a) PICS

Şekil 4.1. Veritabanı örnek resimler.



b) ORL

Şekil 4.1. (devam ediyor).

4.1. ÖZYÜZLER YÖNTEMİ İLE YÜZ TANIMA SİSTEMİ

Yüz Tanıma Sistemi temel olarak 4 ana modülden oluşur. Bu modüller sırası ile kullanılacak veri tabanının seçilmesini sağlayan Veritabanı modülü, tanıma uzayını oluşturacak eğitim yüz görüntüleri üzerinden, özgün yüz bilgileri olan temel özellik değerlerinin çıkarılması ve özyüz (eigenface) uzayının oluşturulmasını içeren Yüz Özellik Çıkarımı modülü, sisteme yüz tanıma girişi olarak verilen test görüntüsü üzerinden, özyüz uzayı değerlerine göre tanıma işleminin yapıldığı Tanıma modülü ve CPU ve GPU tabanlı işlemlerin zaman analizi ile tanıma oranlarının analizlerini yapan Analiz modülüdür.

Kullanıcı yüz resimlerinin ait olduğu klasörü işaretleyerek buradaki resimleri kullanabilir. Resimlerin gri tonlu ya da renkli olabileceği göz önünde bulundurularak ilk olarak gri tonlu hale dönüştürülür. Burada gri tonlu resimlerin bozulmaması için RGB (Red-Green-Blue) parlaklık değerleri eşit katsayılarla çarpılır.

Sistemin farklı ışıklandırmaları tolere etmesi için normalizasyon kullanılmıştır. Her resim için gri tonlu skala maksimum ve minimum değerleri bulunmuştur. Min-max normalizasyon yöntemi kullanılarak resim değerleri yeniden hesaplanmıştır.

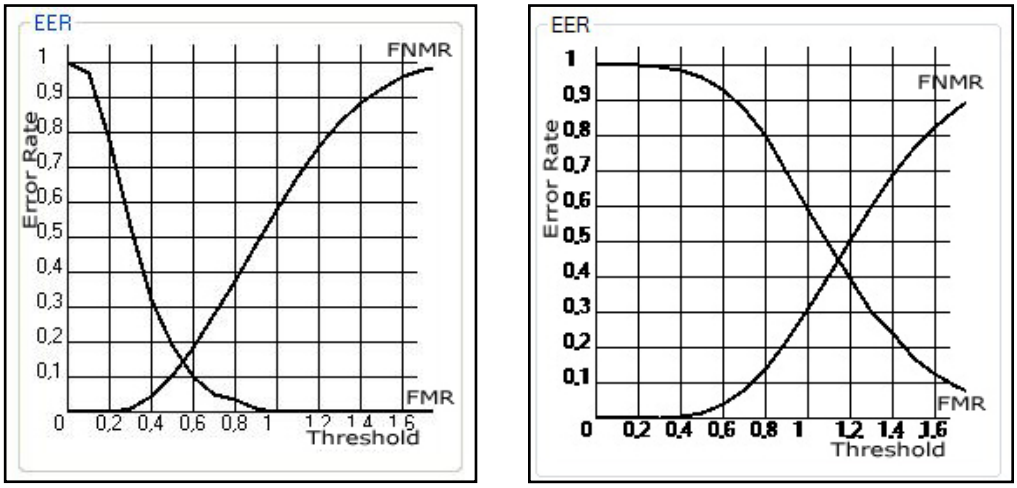
$$x' = \frac{x - \min}{\max - \min} \tag{4.1}$$

Ön işlemlerden geçen resimler PCA yöntemi ile daha düşük boyutlara indirgenmektedir. PCA algoritma adımları kullanılarak veritabanındaki yüzleri ifade edecek en iyi M' adet özyüz araştırılmaktadır. Özyüzlerin hesaplanmasında resim değerlerinin saklandığı matrise ait özdeğer ve özvektörlerin bulunması gerekmektedir. Bu çalışmada özdeğer ve özvektörlerin hesaplanmasında QL Metodu kullanılmıştır[38]. Özvektörler karşılık geldikleri özdeğerlere göre büyükten küçüğe sıralanmıştır. Bu özyüzlere ait ağırlıklar yeni yüzün tanınmasında kullanmaktadır.

Yüz tanımadaki ikinci aşama olan yeni bir yüzün tanınması, kişinin bilenen hangi kişi olduğu ya da bilinen kişilerden biri olmadığıdır. Tanıma işlemi için tanınacak yüz gri tonlu hale dönüştürülür. Işıklandırma etkilerini en aza indirgemek için min-max normalisazyonu yapılır. PCA algoritma adımları uygulanarak ağırlıklar elde edilir.

$$\varepsilon_k = \left\| (\Omega - \Omega_k) \right\| \tag{4.2}$$

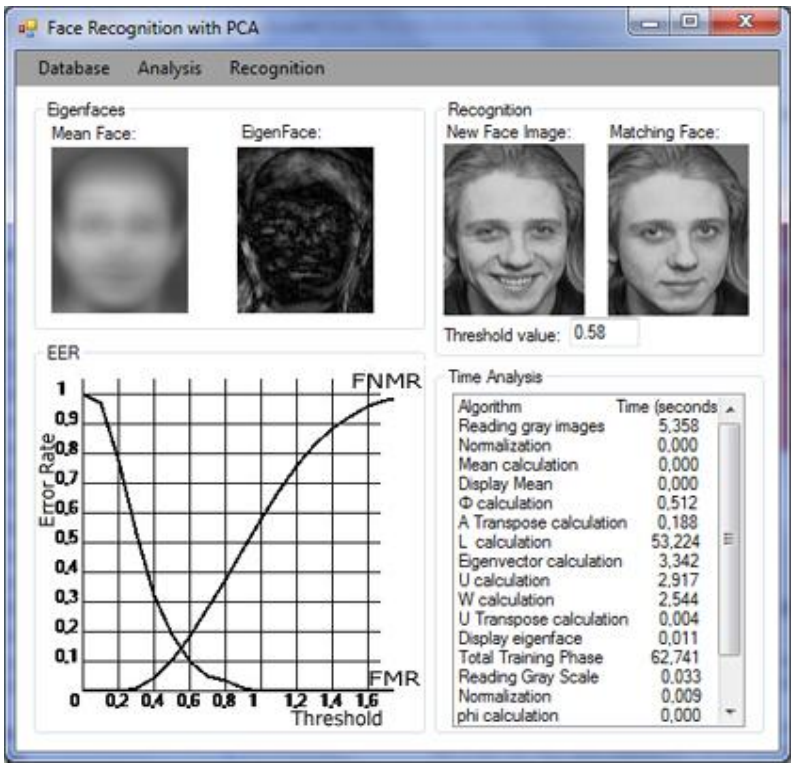
Öklid mesafesi (4.2) denklemi yöntemiyle ağırlıkların karşılaştırma işleminde eşik değeri önem kazanmaktadır. Eşik değeri seçilirken Yanlış Eşleme Oranı (False Match Rate-FMR) ve Yanlış Eşlememe Oranı (False Non-Match Rate-FNMR), oranları kullanılmaktadır. Eşit Hata Oranı (Equal Error Rate-EER), FNMR(T)=FMR(T) şartının gerçekleştiği noktada elde edilen hata oranını ifade etmektedir [52]. Sistem eşit sayıda kişiyi yanlış olarak kabul veya yanlış olarak reddedecek şekilde eşik değeri ayarlanır. Şekil 4.2'de ORL ve PICS veritabanları için Eşit Hata Oranı grafikleri görülmektedir.



a) ORL Veritabanı Hata Oranları b) PICS Veritabanı Hata Oranları

Şekil 4.2. Özyüzler yöntemi veritabanı hata oranları.

Testler sonucu bulunan eşik değerinin altında bir fark değeri elde edilemezse "Tanınamadı" uyarısı verilir, aksi durumda en düşük farka ait resim ekrana bastırılır. Uygulamaya ait ekran çıktısı şekil 4.3 de görülmektedir.



Şekil 4.3. Uygulama arayüzü.

Bu çalışma ile ORL veritabanında yüz tanımada bilinen resimler testinde %80 doğru tanıma ve %20 tanıyamama olmuştur. Bilinmeyen yüz testinde ise %4 yanlış tanıma meydana gelmiştir. Tüm testler bir arada düşünüldüğünde uygulamada %2 hatalı tanıma, %8 tanıyamama ve %90 oranında başarı elde edilmiştir.

4.1.1. Deneysel Sonuçlar Ve Değerlendirme

Geliştirilen yüz tanıma sisteminin CPU ve GPU çalışma süreleri ölçülerek sistemin zaman analizi yapılmış ve paralel programlamanın performans üzerindeki etkileri incelenmiştir. Çalışma sürelerinin ölçüldüğü bilgisayarlar Çizelge 4.1.'de ve grafik kartları Çizelge 4.2.'de gösterilmektedir.

Çizelge 4.1. Kullanılan bilgisayar özellikleri.

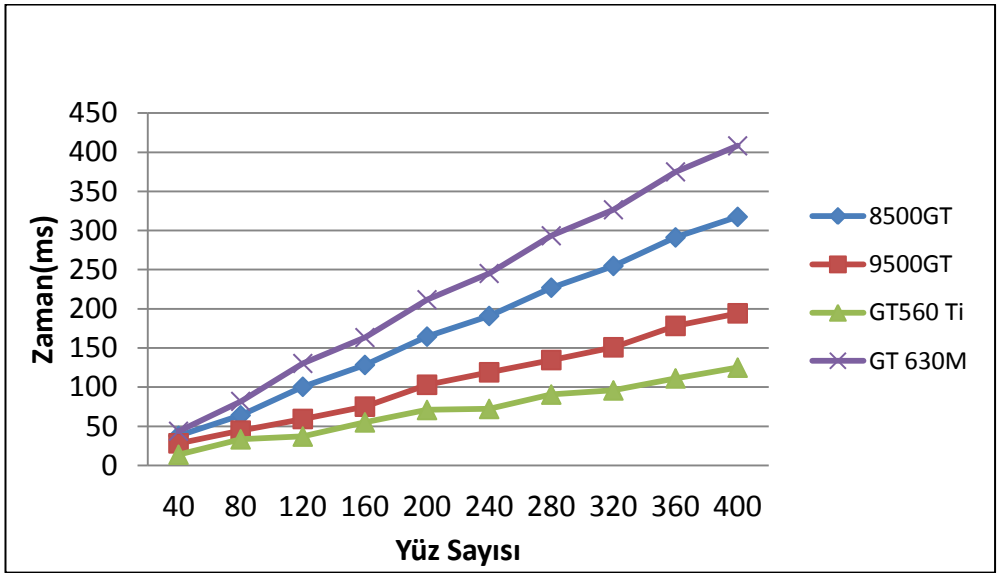
Algorithms	1	2	3
İşlemci	Intel(R) Core(TM) 2 Duo E6550 2.33 GHz	Intel(R) Core(TM) 2 Quad Q8300 @ 2.50GHz	Intel(R) Core(TM) i7-26700QM @ 2.20 GHz
Ram Bellek	2 GB	4 GB	8 GB
İşletim sistemi	Windows XP	Windows 7	Windows 7 Home Premium
Sistem Türü	32 bit	32 bit	64 bit

Çizelge 4.2. Kullanılan GeForce grafik kartı özellikleri.

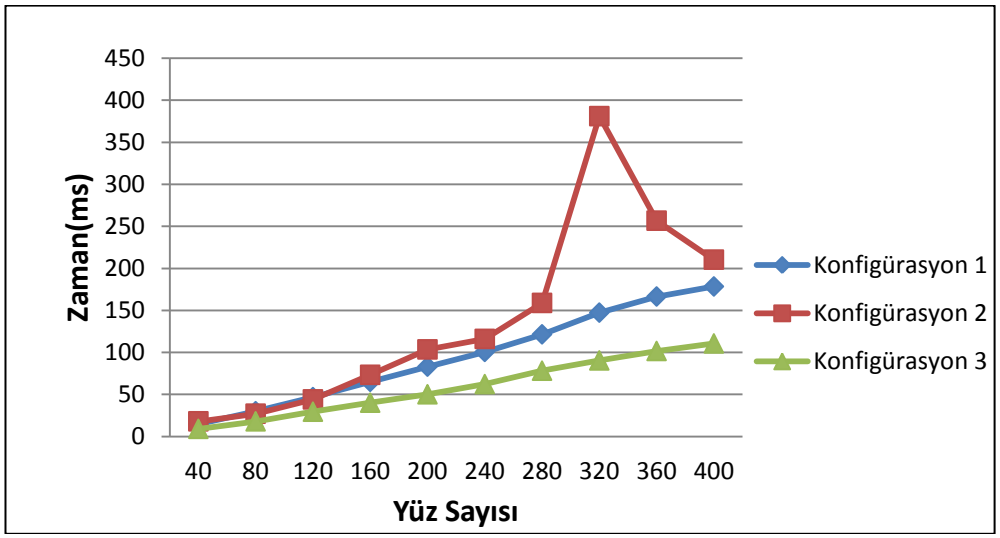
Grafik Kartı	8500GT	9500GT	GT560	GT630M
Core Sayısı	16	32	384	96
Compute Capability	1.1	1.1	2.1	2.1

Yüz tanıma sisteminde paralel programlanabilen algoritmalar Cuda C ile gerçekleştirilerek dll şeklinde sisteme entegre edilmiştir. İşlemci ve grafik kartta algoritmaların çalışma süreleri her algoritma 10 defa çalıştırlıp ortalaması alınarak

oluşturulmuştur. Algoritmalara ait zaman analizleri ve hızlanma durumları grafiklerle sunulmaktadır.

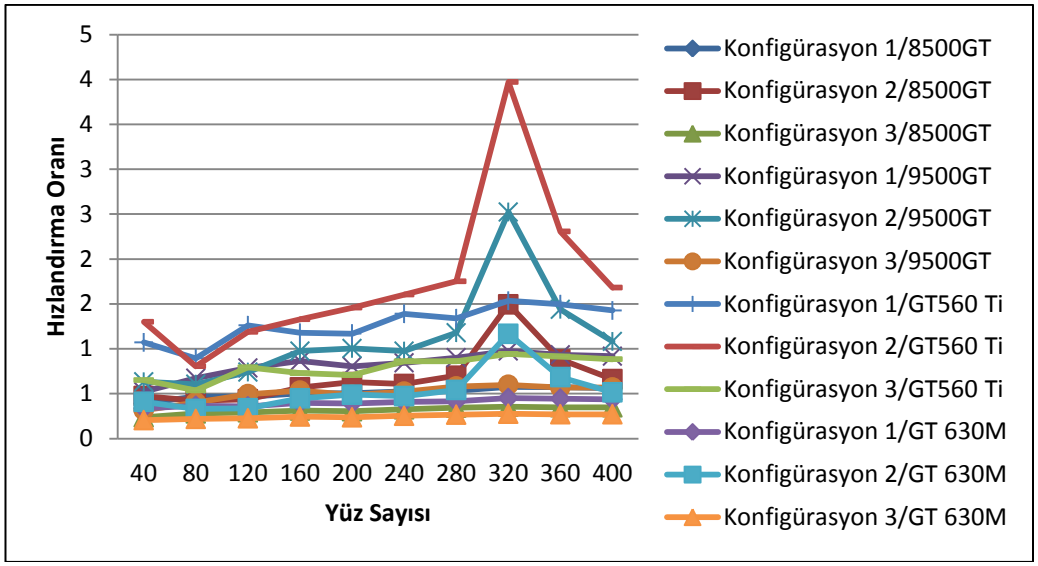


a) Normalizasyon GPU zaman analizi.



b) Normalizasyon CPU zaman analizi.

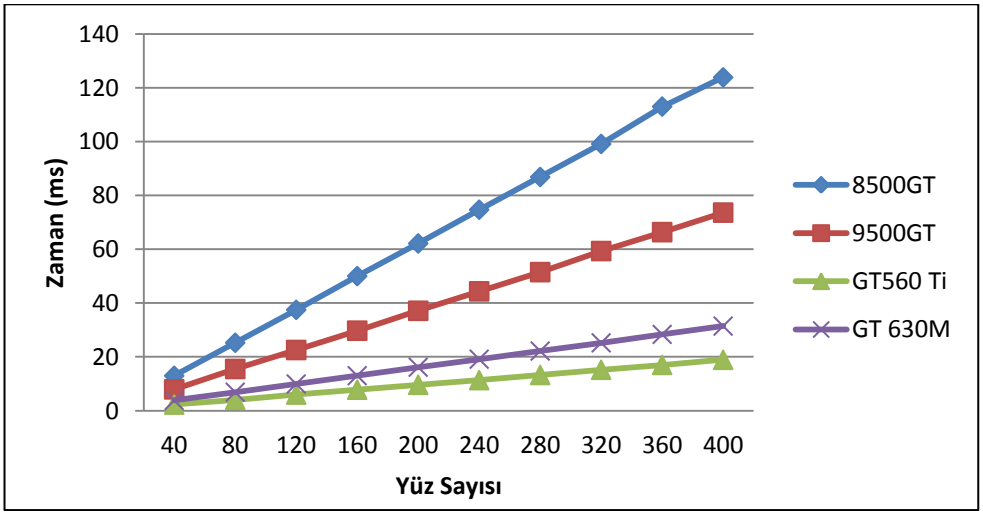
Şekil 4.4. Normalisazyon fonksiyonu analizi.



c) Normalizasyon hızlandırma oranları analizi.

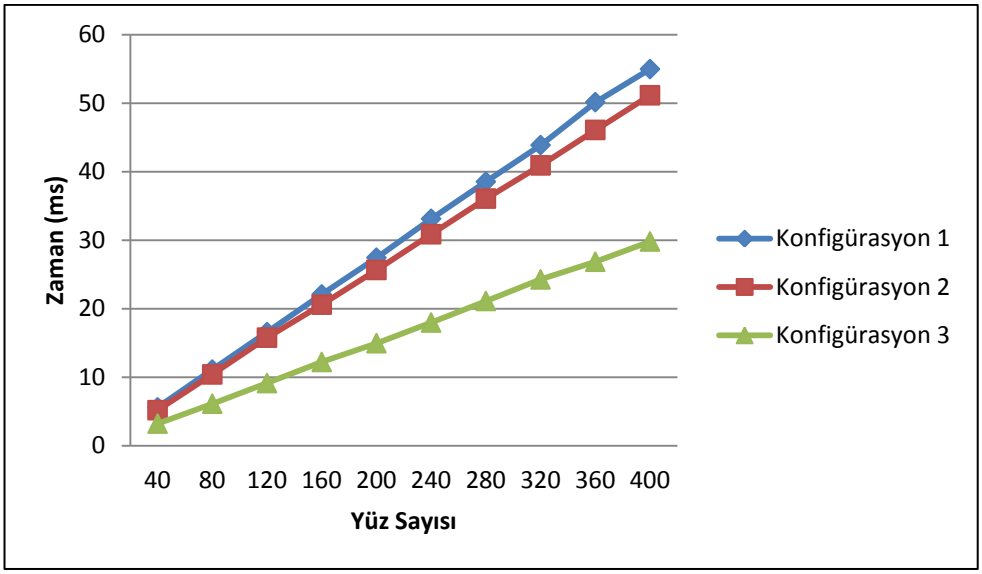
Şekil 4.4. (devam ediyor).

Uygulanan min-max normalizasyonunda her bir iş parcacığı bir yüz resminin pixellerini dolaşarak minimum ve maximum değerleri bulmaktadır.Aynı iş parcacığı tekrar yüz resmine ait tüm pixelleri dolaşarak değerleri 0-1 aralığına indirgemektedir. Bu durumda tek bir iş parcacığına düşen yük çok fazla olmaktadır. İstenilen hızlandırma oranları elde edilemediğinden normalizasyon algoritması önerilen uygulamada kullanılmamıştır.

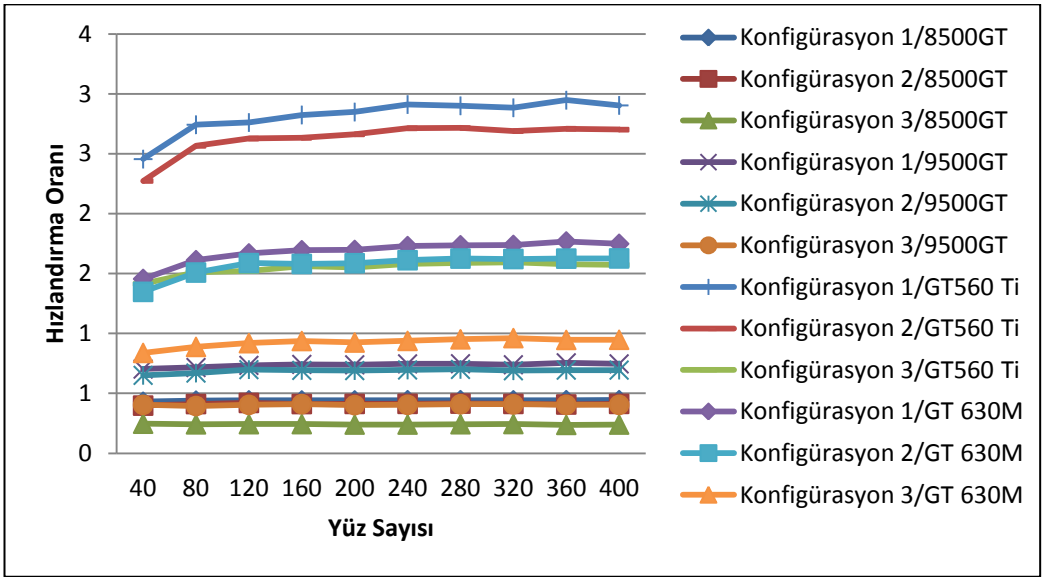


a) Ortalama hesaplama GPU zaman analizi.

Şekil 4.5. Ortalama hesaplama analizi.



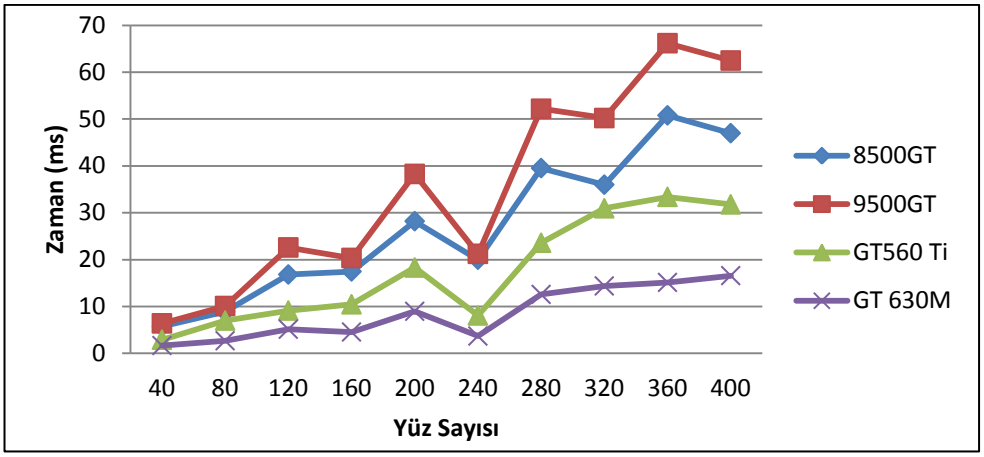
b) Ortalama hesaplama CPU zaman analizi.



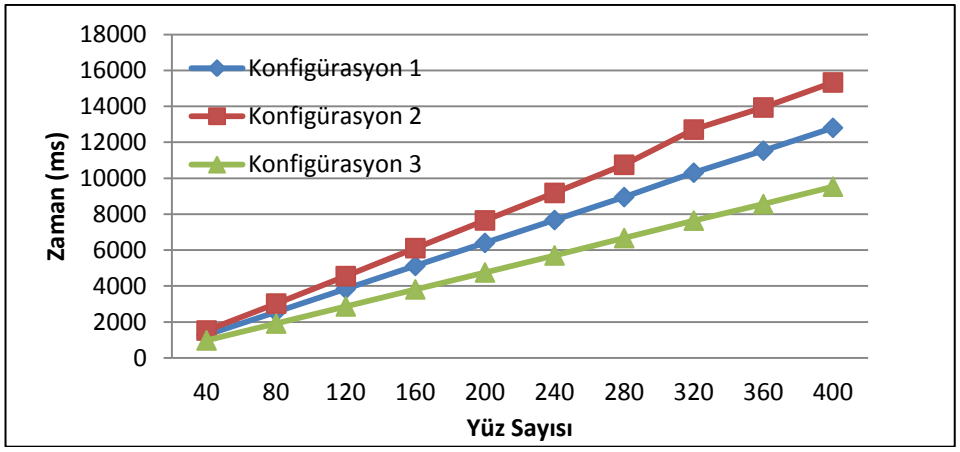
c) Ortalama hesaplama hızlandırma oranları analizi.

Şekil 4.5. (devam ediyor).

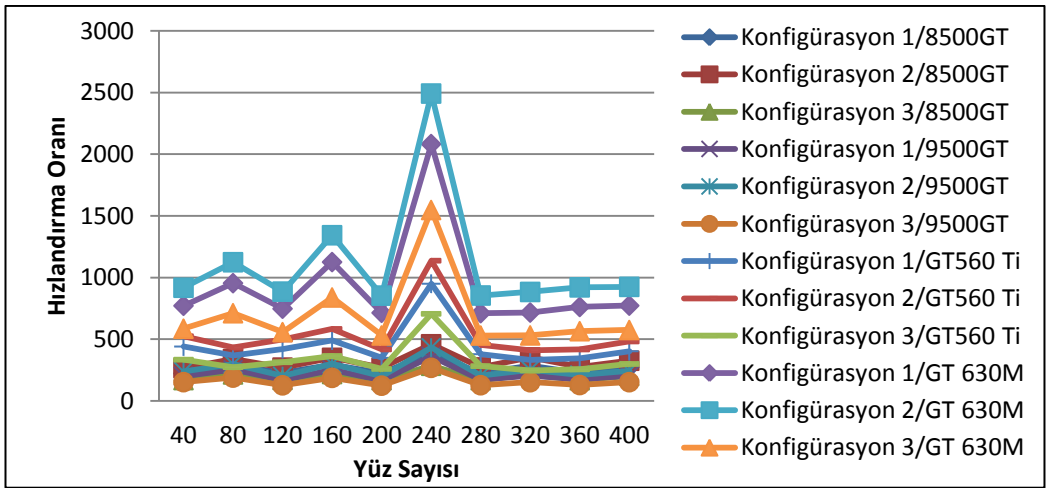
Ortalama hesaplama algoritması her bir pixel için tüm yüz resimlerinin ortalamasını bulmaktadır. Çekirdek sayısı sırasıyla 96 ve 384 olan GT630M ve GT560 Ti 2,3 katlık bir hızlandırma sağlamıştır. Önerilen uygulamada çekirdek sayısının yüksek olduğu grafik kartlarda kullanılması öngörülmüştür.



a) A (fark) hesaplama GPU zaman analizi.



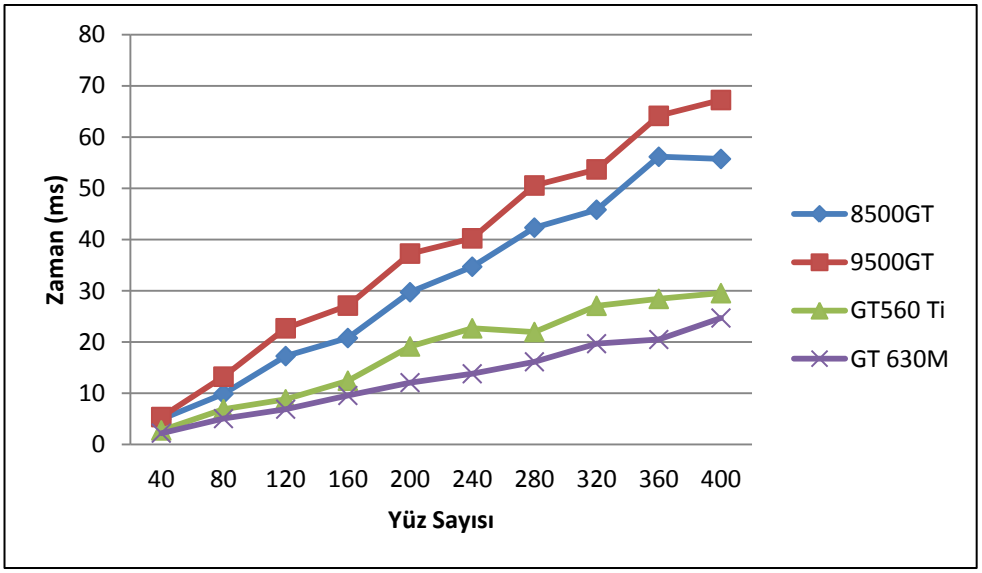
b) A (fark) hesaplama CPU zaman analizi.



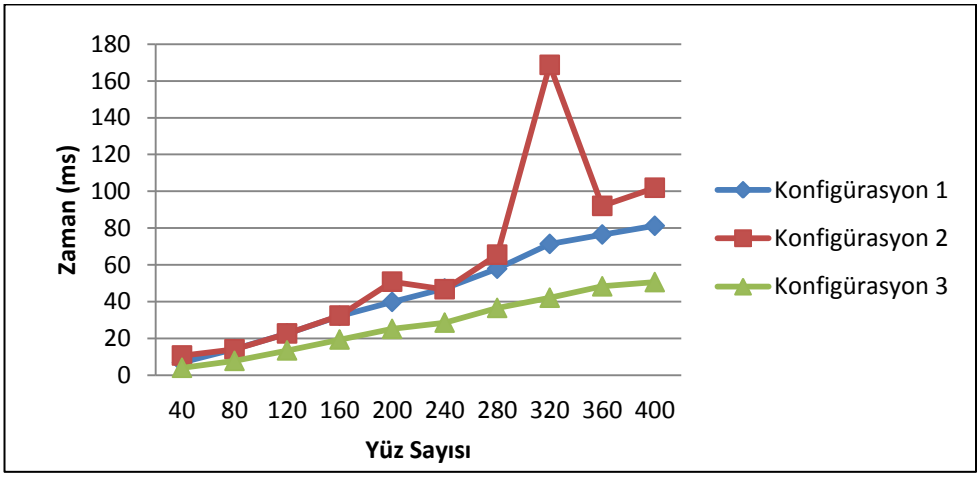
c) A (fark) hesaplama hızlandırma oranları analizi.

Şekil 4.6. A (fark) hesaplama analizi.

A(fark) hesaplama algoritması her yüz resminin ortalamadan farkını hesaplamaktadır. Bir iş parçacığı yalnızca bir pixelin ortalamadan farkını hesapladığı için paralel programlama ile 150 ile 2500 aralığında hızlandırma oranları elde edilmiştir.

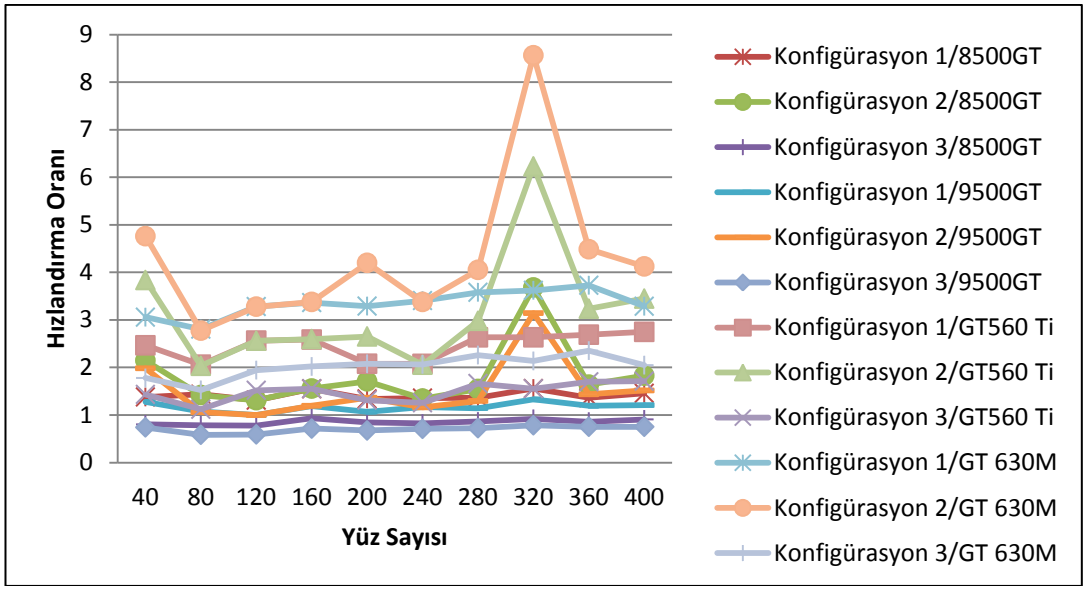


a) A transpose hesaplama GPU zaman analizi.



b) A transpose hesaplama CPU zaman analizi.

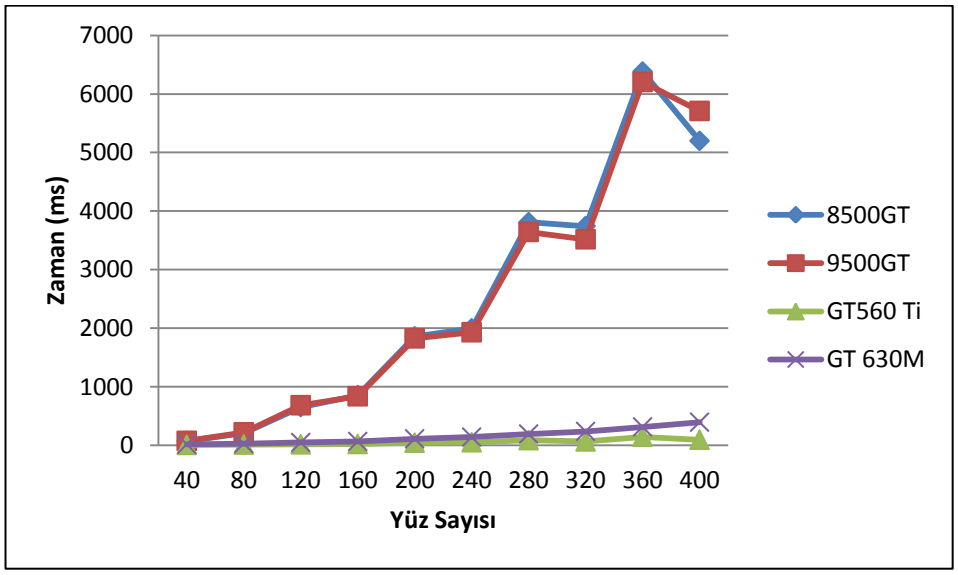
Şekil 4.7. A transpose hesaplama analizi.



c) A transpose hesaplama hızlandırma oranları analizi.

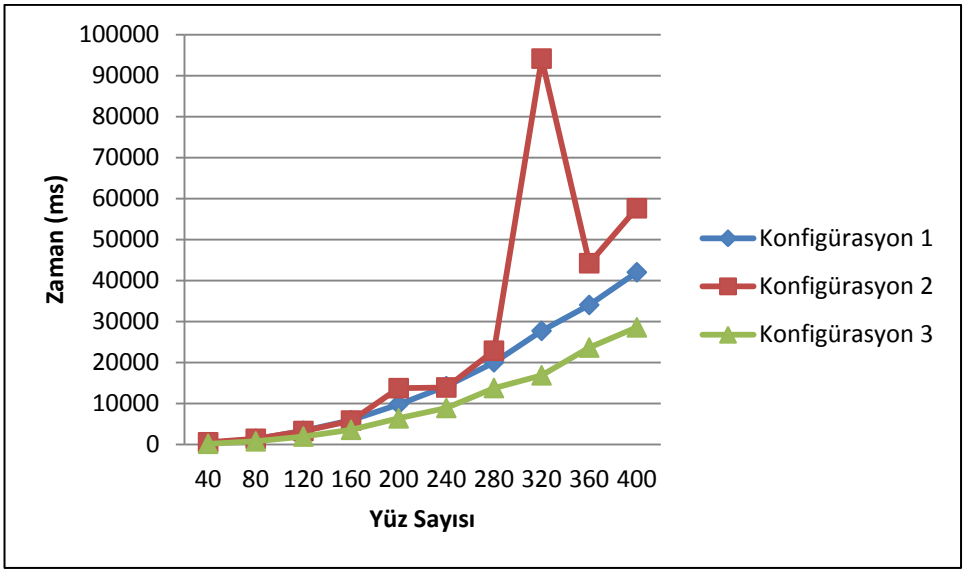
Şekil 4.7. (devam ediyor).

Transpose hesaplama algoritmasındaki hız artışı beklenilen oranların altında kalmıştır. Bu duruma verilerin kopyalanma hızının etki ettiği görülmekle birlikte grafik kartın belleğine okuma ve yazma işleminin yavaşlığıda dikkat çelicidir.

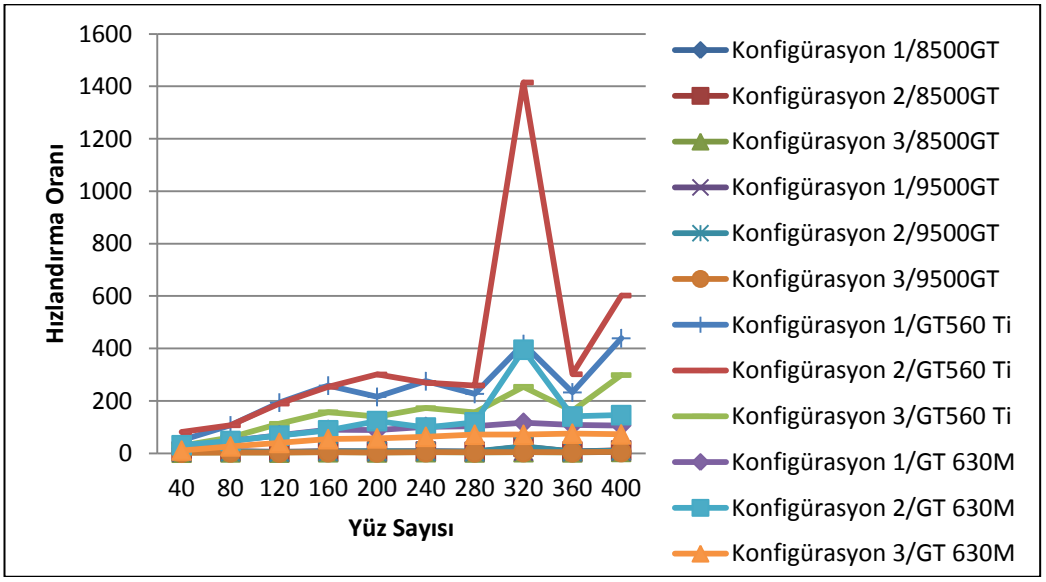


a) L kovaryans hesaplama GPU zaman analizi.

Şekil 4.8. L kovaryans hesaplama analizi.



b) L kovaryans hesaplama CPU zaman analizi.

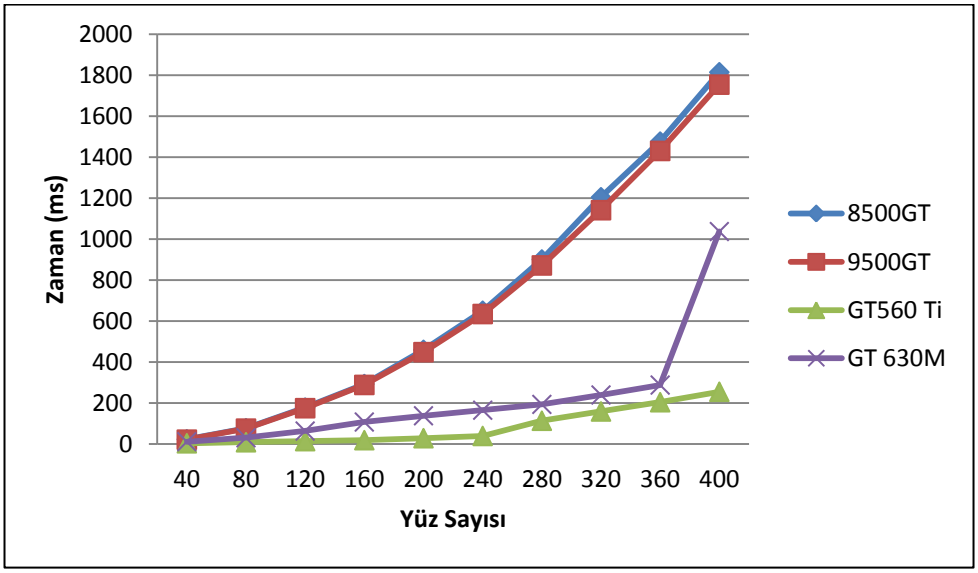


c) L kovaryans hesaplama hızlandırma oranları analizi.

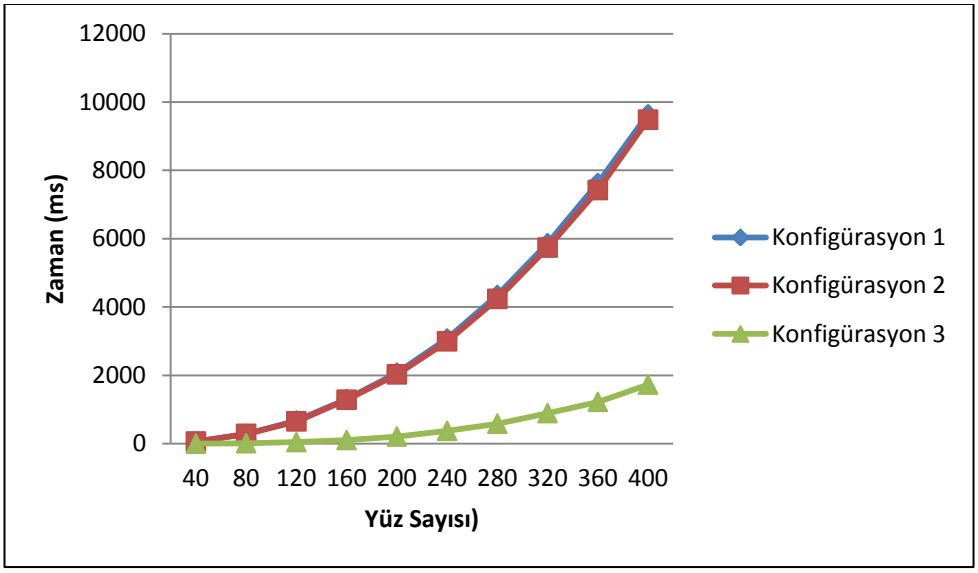
Şekil 4.8. (devam ediyor).

Kovaryans hesaplama özyüzler yöntemi ile yüz tanımada en fazla zaman alan algoritmadır. 3 ile 1415 aralığında hızlandırma oranları elde edilmiştir. Çekirdek sayısı arttıkça hızlandırma oranlarının arttığı gözlemlenmiştir. Ayrıca kullanılan yüz resmi sayısı arttıkça da sonuçlar memnuniyet vermektedir. 96 çekirdekli GT630M grafik kartında 400 yüz resmi için ortalama108 kat , 384 çekirdekli GT560 Ti grafik kartında 400 yüz resmi için ortalama 450 kat hızlanma oranı gözlemlenmiştir.

Kovaryans hesaplama algoritmasına uygulanacak paralel programlama uygulamaya zaman kazandıracaktır.

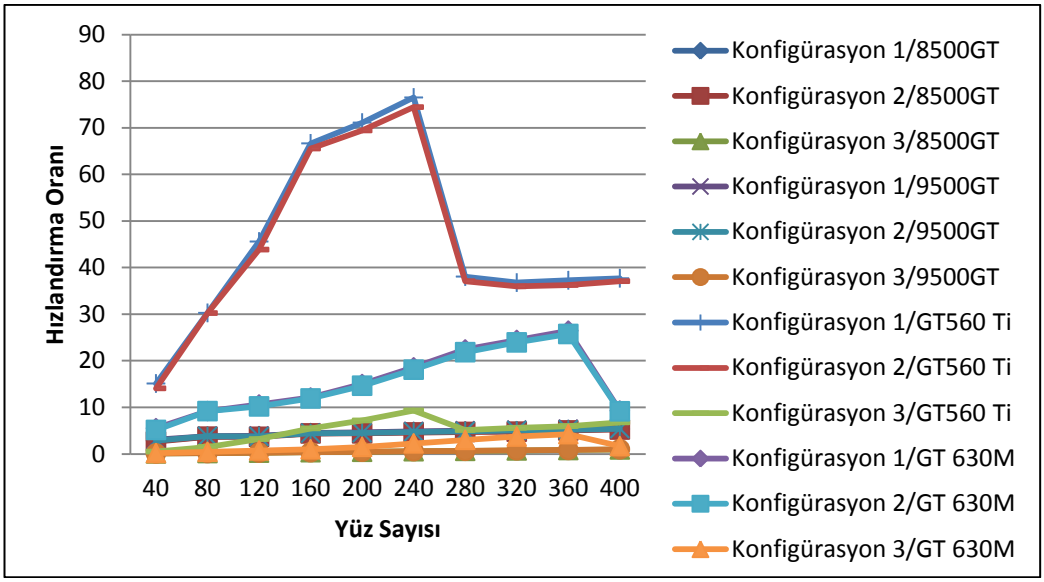


a) Özyüz hesaplama GPU zaman analizi.



b) Özyüz hesaplama CPU zaman analizi.

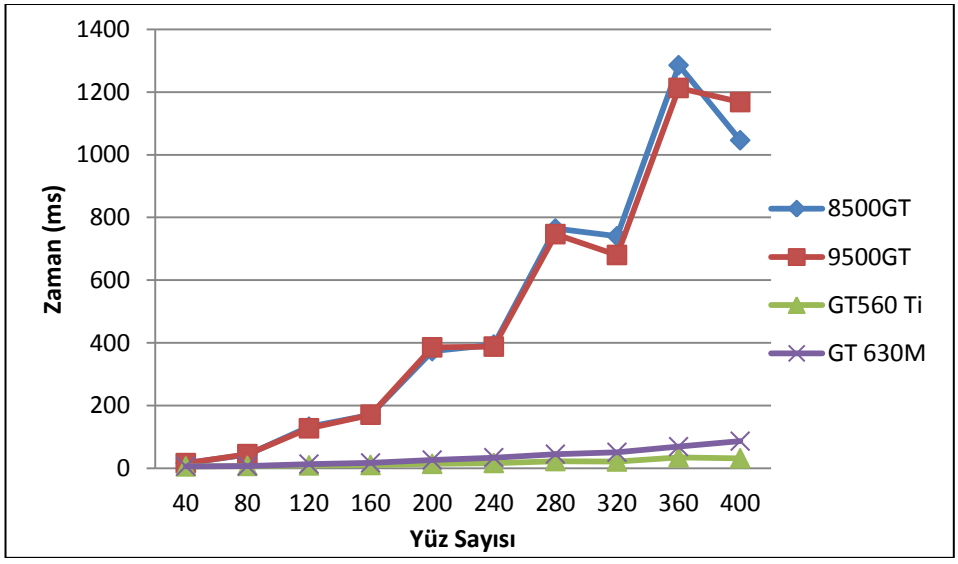
Şekil 4.9. Özyüz hesaplama analizi.



c) Özyüz hesaplama hızlandırma oranları analizi.

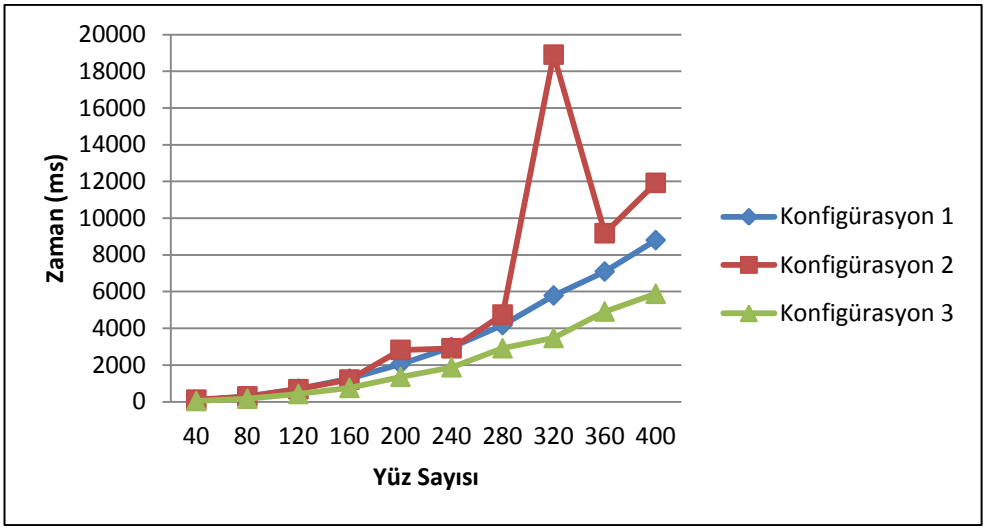
Şekil 4.9. (devam ediyor).

Özyüz hesaplama algoritmasında yüksek çekirdekli grafik kartlarda Konfigurasyon 1 ve 2 bilgisayarlarına göre 75 kata kadar hızlandırma oranları gözlemlenmiştir. Ancak Konfigurasyan 3 bilgisayarına göre hızlandırma oranları oldukça düşüktür.

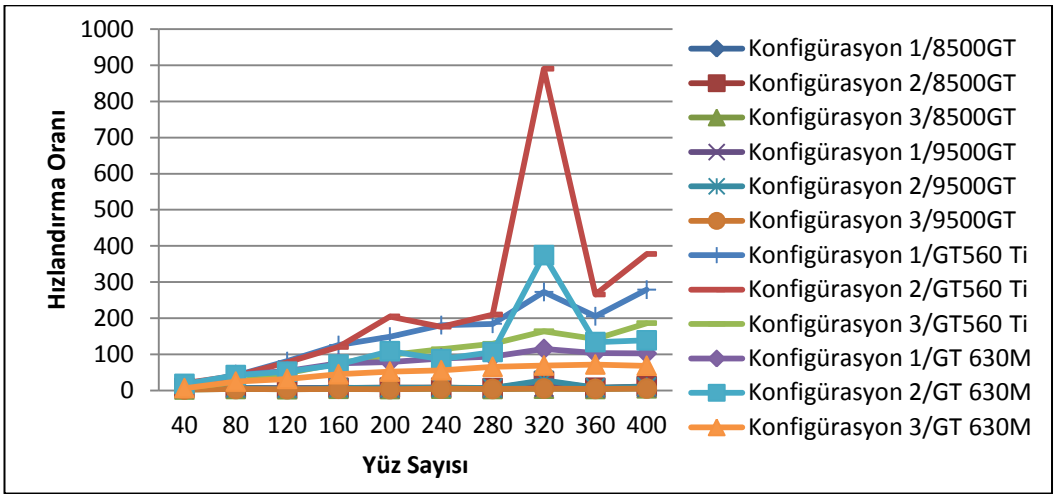


a) Özellik hesaplama GPU zaman analizi.

Şekil 4.10. Özellik hesaplama.



b) Özellik hesaplama CPU zaman analizi.



c) Özellik hesaplama hızlandırma oranları analizi.

Şekil 4.10. (devam ediyor).

Özellik hesaplama algoritmasında artan yüz resmi sayısı ile performansda istenen düzeylerde artışlar gözlenmiştir.

Çizelge 4.3 ve Çizelge 4.4'de özyüzler yöntemi için her bir algoritma adımının sistemin çalıştırıldığı bilgisayar ve grafik kartlar için süreleri verilmiştir. Çizelgeler ve grafiklerin ışığında özyüzler yöntemi ile yüz tanıma sistemi için grafik işlemci ile heterojen çalışan yeni bir sistem önerilmiştir.

Çizelge 4.3. Özyüzler yöntemi eğitim aşaması CPU zaman çizelgesi (400x92x112).

Algoritmalar	Konfigürasyon 1	Konfigürasyon 2	Konfigürasyon 3
Normalizasyon	178	210	111
Ortalama Hesaplama	55	51	30
A(Fark) Hesaplama	12809	15332	9535
A Transpose Hesaplama	81	102	51
L (Kovaryans) Hesaplama	42016	57628	28587
Özvektör Hesaplama*	-	-	-
U (Özyüz)Hesaplama	9641	9481	1731
W(Özellik) Hesaplama	8802	11921	5880
Toplam Süre (ms)	73582	94725	45925

Çizelge 4.4. Özyüzler yöntemi eğitim aşaması GPU zaman çizelgesi (400x92x112).

Algoritmalar	8500 GT	9500 GT	GT 560	GT 630M
Normalizasyon	318	194	125	408
Ortalama Hesaplama	124	74	19	31
A (Fark) Hesaplama	47	62	32	17
A Transpose Hesaplama	56	67	30	25
L (Kovaryans) Hesaplama	5198	5709	96	395
Özvektör Hesaplama*	-	-	-	-
U (Özyüz) Hesaplama	1814	1754	256	1037
W (Özellik) Hesaplama	1046	1168	32	86
Toplam Süre (ms)	8603	9028	590	1999

* Özvektör hesaplama işleminde QL algoritması uygulanmış olup bu algoritmanın paralel programlamaya uyarlanması mümkün olmamıştır. Bu nedenle algoritmaya ait süre çizelgelerde verilmemektedir.

Önerilen sistemde Grafik işlemci ile ana işlemciden maksimum seviyede faydalanılmaya çalışılmıştır. Önerilen sistem için çalışma süreleri ve hızlandırma oranları Çizelge 4.5'de görülmektedir.

Çizelge 4.5. Özyüzler yöntemi önerilen sistem analizi

Algoritmalar	Kullanılan İşlemci
Normalizasyon	CPU
Ortalama Hesaplama	CPU
A (Fark) Hesaplama	GPU
A Transpose Hesaplama	GPU
L (Kovaryans) Hesaplama	GPU
Özvektör Hesaplama	CPU
U (Özyüz) Hesaplama	GPU
W (Özellik) Hesaplama	GPU
Toplam Süre (ms)	4402

Önerilen sistem konfigürasyon 3 bilgisayarında GeForfe GT 630M ekran kartında çalıştırılmıştır. Çalışma süresi 400 yüz resmi için 10 çalışmanın ortalaması alınarak elde edilmiştir. Önerilen sistem ile CPU çalışma süreleri karşılaştırılan algoritmalar için incelendiğinde konfigürasyon 3 bilgisayarı ve GeForfe GT 630M ekran kartında 27x hızlandırma oranı elde edildiği görülmektedir.

4.2. DALGACIK DÖNÜŞÜMÜ YÖNTEMİ İLE YÜZ TANIMA SİSTEMİ

Geliştirilen yüz tanıma sistemi, İki Boyutlu Ayrık Dalgacık Analizi yöntemi ile öznitelik çıkarımını ve regresyon yöntemi kullanarak ise tanıma işlemini gerçekleştirmektedir. Tanınması istenen bir resmin özniteliklerinin çıkarılarak veritabanı içerisindeki resimlere ait özniteliklerle ile karşılaştırılması ve en benzer

olan resmin sonuç olarak döndürülmesi esasına dayanmaktadır. Matlab ortamında geliştirilen yüz tanıma sisteminde matlab wavelet fonksiyonları kullanılmıştır.

Matlab Wavemenu Araç Kutusu ile verilen bir resmin dalgacık katsayıları elde edilebilmektedir. Şekil 4.4'de Wavemenu ile gerçekleştirilen dalgacık dönüşümlerine ait ekran görüntüleri gösterilmektedir.

a) 2. seviye haar dalgacık dönüşümü.

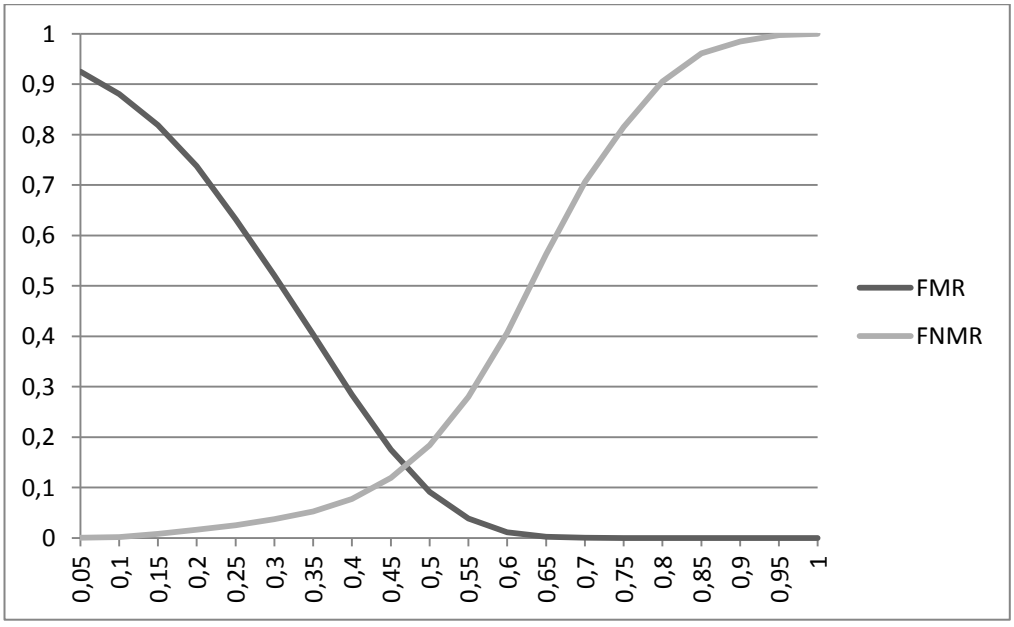
b) 4. seviye bior 3.7 dalgacık dönüşümü.

Şekil 4.11. Matlab wavemenu ile iki boyutlu ayrık dalgacık analizi.

Yüz veritabanı her kişiye ait resimlerin o kişiye ait klasörde depolanması şeklinde oluşturulmuştur. Kullanılan ORL veritabanı için her klasörde 10 resim olmak üzere toplam 40 klasör olarak dizayn edilmiştir.

Sistem ilk olarak veritabanındaki yüz resimleri için 2. seviyeden İki Boyutlu Ayrık Dalgacık Analizi kullanarak veritabanındaki resimler için özet katsayıları oluşturur. Sisteme tanınması istenen yüz resmi verildiğinde yeni yüz için de 2. seviye İki Boyutlu Ayrık Dalgacık Analizi ile resmin özet katsayıları elde edilir. Tanıma işlemi tanınacak resmin özet bilgileri ile veritabanındaki resimlerin özet bilgileri arasındaki regresyon değerlerine göre sonuçlandırılır. Regresyon değerleri -1 ile +1 aralığındadır. İki resim arasındaki benzerlik arttıkça elde edilen regresyon değeri de +1'e yaklaşmaktadır.

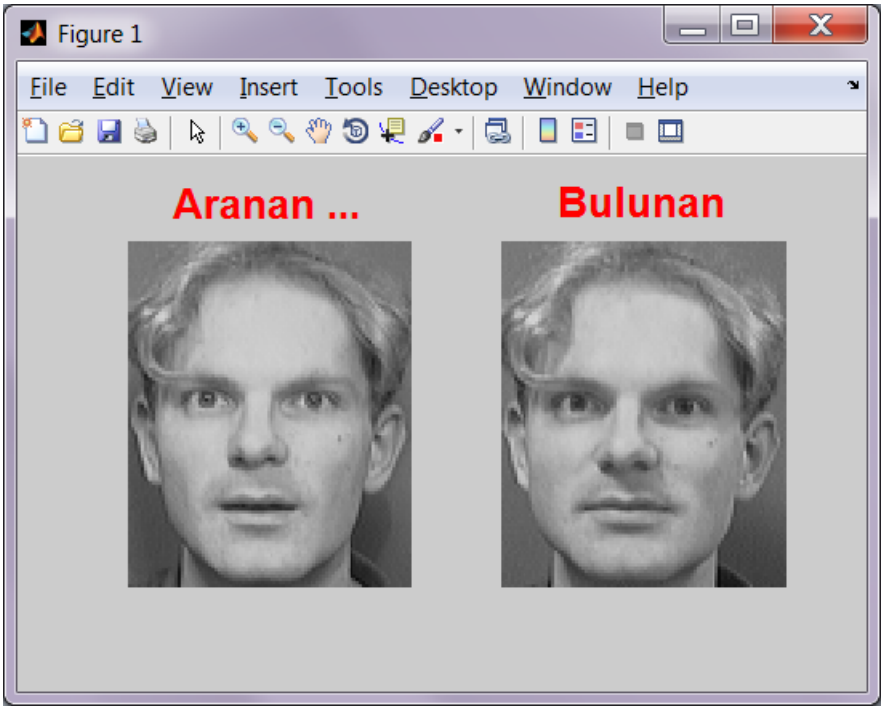
Tanıma işlemine ait sonuca karar vermekte eşik değeri önem kazanmaktadır. Eşik değerine Eşit Hata Oranı yöntemiyle karar verilmiştir. Şekil 4.5'de ORL veritabında elde edilen hata Yanlış Eşleme ve Yanlış Eşlememe hata oranlarına ait grafik görülmektedir. Yanlış eşleme oranı 0,6 değerinden itibaren sıfıra çok yaklaşmasına rağmen tam olarak 0,8 değeriyle sıfır olmaktadır. Bu nedenle yanlış eşlemeyi minumum seviyede tutmak için eşik değeri 0,72 seçilmiştir.



Şekil 4.12. Dalgacık dönüşümü yöntemi ORL veritabanı eşit hata oranı.

ORL veritabanında yapılan testlerde 30 kişinin 8'er resmi eğitim seti olarak kullanılmıştır. 30 kişinin kalan 2'şer resmi bilinen yüz testinde, diğer 10 kişiye ait toplam 100 resim bilinmeyen yüz testinde kullanılmıştır. Tüm testler bir arada düşünüldüğünde uygulamada %2 hatalı tanıma, %9 tanıyamama ve %89 oranında başarı elde edilmiştir.

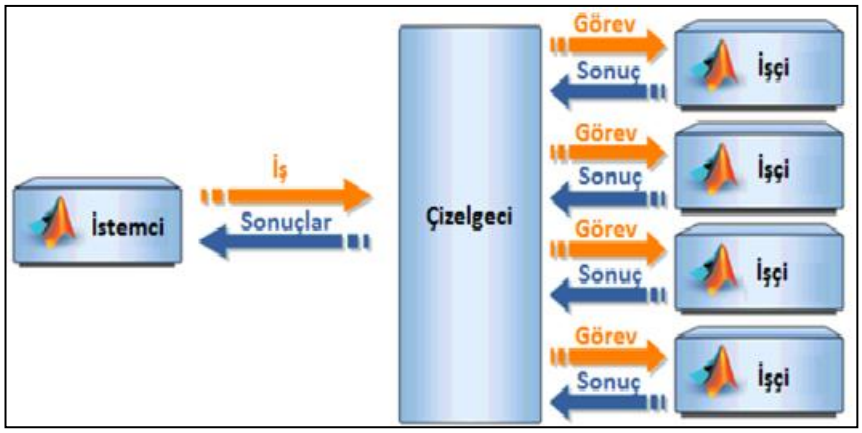
Karşılaştırmaların sonucunda elde edilen en yüksek regresyon değeri belirlenen eşik değerinin altında kalırsa sistem "tanınmadı" uyarısını verecektir. Elde edilen sonuç eşik değerinin üstünde ise sistem tanınan kişiyi ekranda görüntülemektedir. Şekil 4.13'de yapılan çalışmaya ait arayüz görünmektedir.



Şekil 4.13. Dalgacık yöntemi ile yüz tanıma sistemi arayüzü

4.2.1. Deneysel Sonuçlar Ve Değerlendirme

İki Boyutlu Ayrık Dalgacık Analizi yöntemi ile geliştirilen yüz tanıma sistemi Matlab ortamında geliştirildiği için paralel programlama yöntemi olarak da Matlab Paralel İşleme Araç Kutusu (Parallel Computing Toolbox) kullanılmıştır. Paralel işleme araç kutusunun sağladığı görev paralel yapı Şekil 4.14'de görülmektedir.



Şekil 4.14. Matlab paralel işleme araç kutusunun sağladığı görev paralel yapı [46].

Yüz tanıma sisteminde kullanılan algoritma genel olarak iki aşama içermektedir. İlk aşama veritabanındaki tüm resimler için dalgacık dönüşümünün yapılmasıdır. İkinci aşama ise tanınması istenen yüz resmi ile veritabanındaki yüz imgelerinin regresyon değerlerininin hesaplanmasıdır.

Çizelge 4.6. İki boyutlu ayrık dalgacık analizi filtreleme seviyesine göre zaman analizi

	Filtreleme	Normal İşlenme Süresi (s)	Paralel İşlenme Süresi (s)	Hızlandırma Oranı
Özellik Çıkarımı	1. seviye	2,1367	0,64145	3,33
	2.seviye	2,4311	0,70635	3,44
	3.seviye	2,6842	0,76845	3,49
Regresyon Hesaplama	1. seviye	0,0405	0,1765	-4,36
	2.seviye	0,0291	0,1689	-5,80
	3.seviye	0,0263	0,1645	-6,27
Toplam	1. seviye	2,1772	0,8180	2,66
	2.seviye	2,4602	0,8752	2,81
	3.seviye	2,7105	0,9329	2,91

Dalgacık dönüşümünde kaç seviye filtreleme yapılacağına karar vermek için farklı seviyelerde denemeler yapılmış ve optimum performansa göre karar verilmiştir. Çizelge 4.5'de filtreleme seviyelerine göre işlem süreleri görülmektedir. Regresyon hesaplamada hızlandırma oranlarının negatif olduğu görülmektedir. Buradaki negatiflik paralel programlama ile performans artışı yerine düşüş gerçekleştiğini ifade etmektedir.

Önerilen sistemde iki boyutlu ayrık dalgacık analizi filtreleme seviyesi olarak 2. seviye kullanılmasına karar verilmiştir. Ayrıca regresyon hesaplamada paralel programlamanın negatif etkisi nedeniyle seri programlama tercih edilmiştir. Önerilen sistemin çalışma süresi incelendiğinde toplam 0,7354 saniye olduğu görülmektedir. Paralel programlamanin dalgacık dönüşümünde kullanılması ile 3x oranında bir performan artışı elde edilmektedir.

BÖLÜM 5

SONUÇ VE ÖNERİLER

Bu çalışmada özyüzler ve dalgacık dönüşümü yöntemlerine dayanan yüz tanıma sistemleri geliştirilmiştir. Özyüzler yöntemi kolay uygulanabilir olması nedeniyle hızlı bir çözüm yaklaşımı olarak tercih edilmiştir. Önerilen yüz tanıma sisteminin farklı ışıklandırmalar altında iyi sonuç verdiği görülmüştür. Ayrıca yüzün görünürlüğünü etkilemeyen poz değişimleri de sistem tarafından tolere edilmektedir. Resimler için yapılan ön işlemler, özyüzlerin hesaplanması ve tanınması istenen yüz resmi için yapılan işlemler zaman bakımından analiz edilip değerlendirilmiştir.

Zaman analizi sonuçları yüz tanıma sisteminin eğitim aşamasında kişi sayısındaki artışların sistemin hızına negatif etkilerini ortaya koymaktadır. Sisteme eklenecek her yeni kişi için hesaplamaların yeniden yapılması gerekliliği göz önünde bulundurularak özyüzler yöntemi ile geliştirilen yüz tanıma sistemine paralel pragramlama uygulanmıştır.

Özyüzler yönteminde paralel programlama yöntemi olarak CUDA seçilmiştir. Erişiminin ve maliyetinin uygun olması ve işlemci üzerine daha fazla yük binmesinin önleniyor olması bu seçimde etkili olmuştur. Dalgacık dönüşümü yöntemi ile yüz tanıma sistemi Matlab ortamında geliştirilmiştir. Uygulama kolaylığı ve ek maliyetinin olmaması sebebiyle Matlab Paralel İşlem Araç Kutusu paralel programlama yöntemi olarak tercih edilmiştir.

Bu çalışmada geliştirilen yüz tanıma sistemlerinde kullanılan paralel programlamanın her algoritmada aynı iyileştirmeyi göstermediği gözlenmiştir. Bu nedenle paralel programlamanın her adımda kulanılmasından ziyade kabul edilebilir performans artışı gösterdiği adımlarda kullanılmasına karar verilmiştir.

Sonuç olarak, bu tez kapsamında Özyüzler ve İki Boyutlu Ayrık Dalgacık Analizi yöntemleri ile geliştirilen yüz tanıma sistemlerine paralel programlama yöntemlerinden CUDA ve Matlab Paralel İşlem Araç Kutusu uygulanmıştır. Geliştirilen yüz tanıma sistemlerinin çalışma süreleri ölçülerek sistemin zaman analizi yapılmış ve paralel programlamanın performans üzerindeki etkileri incelenmiştir. Kullanılan her yöntem için, her bir algoritma adımında seri programlama ve paralel programlama karşılaştırması yapılmış, paralel programlamanın etkilerine göre hangi adımlarda kullanılacağına karar verilmiştir. Özyüzler yönteminde CUDA ile 23 kat ve İki Boyutlu Ayrık Dalgacık Analizi yönteminde Matlab Paralel İşlem Araç Kutusu ile 3 kat hızlanma elde edilmiştir.

KAYNAKLAR

1. Görgünoglu, S., " Parmakizi analizinde performans optimizasyonu", Doktora Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü,*** Ankara, 1-65 (2006).

2. Görgünoğlu, S., Öz, K. and Bayır, Ş., "Performance analysis of eigenfaces method in face recognition system", ***2. International Symposium on Computing in Science & Engineering***, Aydın, 136-142 (2011).

3. Uludağ, U., Pankanti, S., Prabhakar, S. and Jain, A. K., "Biometric cryptosystems: ıssues and challenges.", ***In Proceedings of the IEEE Special Issue on Enabling Security Technology for Digital Rights Management***, 92 (6): 948-960 (2004).

4. Sarma, G. and Singh, P. K., "Internet banking: risk analysis and applicability of biometric technology for authentication.", ***Int. J. Pure Appl. Sci. Technol***., 1 (2): 67-78 (2010).

5. Tolba, A. S., El-Baz, A.H. and El-Harby, A. A., "Face recognition: a literature review", ***International Journal of Signal Processing,*** 2 (2): 88-103 (2005).

6. Tan, X., Chen, S., Zhou, Z. and Zhang F., "Face recognition from a single image per person: A survey", ***Pattern Recognition,*** 39: 1725-1745 (2006).

7. İnternet: Boğaziçi Üniversitesi "Face Recognition Using Principle Component Analysis"**http://www.cmpe.boun.edu.tr/courses/cmpe360/spring2007/files/pca/project_spec.pdf** (2012)

8. Abate, A. F., Nappi , M. Riccio, D. and Sabatino, G., "2D and 3D face recognition: A survey", ***Pattern Recognition Letters,*** 28: 1885–1906 (2007).

9. Atalay, İ., "Face recognition using eigenfaces", M. Sc. Thesis, ***Istanbul Technical University Institute Of Science And Technology***, İstanbul, 10-20 (1996)

10. Gökberk, B., Irfanoglu, M. O., Akarun, L. and Alpaydın, E., "Optimal gabor kernel selection for face recognition", ***Proceedings of the IEEE International Conference on Image Processing***, Barcelona, 677-680 (2003).

11. Gümüş, E., "Yüz tanıma problemine karma yöntemlerin uygulanması", Yüksek Lisans Tezi, ***İstanbul Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul, 1-15 (2008).

12. Ergezer, H., “Yüz tanıma: Öz yüzler, yapay sinir ağları,gabor dalgacık dönüşümü yöntemleri”, Yüksek Lisans Tezi, ***Başkent Üniversitesi Fen Bilimleri Enstitüsü***, Ankara, 1-4 (2003).

13. Topkaya, İ. S., “Video görüntülerinden yüz tanıma”, Yüksek Lisans Tezi, ***Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul, 4-5 (2008).

14. Acar, C., Atlas, A., Çevik, K., Ölmez, İ., Ünlü, M., Özkan, D. and Duygulu , P., “Systematic evaluation of face detection algorithms on news videos”, ***IEEE 15. Sinyal İşleme ve İletişim Uygulamaları Kurultayı***, Eskişehir, 1200-1203 (2007).

15. Muhammad, İ., Telatar, Z. ve Tüzünalp, Ö., “Yüz algılama algoritmalarında tarama zamanının azaltılması için bir hızlı ön tarama algoritması”, ***IEEE 9. Sinyal Isleme ve Uygulamalari Kurultayi***, Gazimagusa-Kibris, 565-570 (2001).

16. Anbarjafari, G. "A new face recognition system based on colour statistics", Yüksek Lisans Tezi, ***Doğu Akdeniz Üniversitesi Fen Bilimleri Enstitüsü*** , Gazimağusa, 1-4 (2008).

17. Demirel, H. ve Anbarjafari, G. “Renkli histogram eşleştirme tabanlı yeni bir yüz tanıma sistemi”, ***IEEE 16. Sinyal İşleme ve İletişim Uygulamaları Kurultayı***, Didim, 609-612 (2008).

18. Özkaya, N. ve Sağıroğlu, Ş., “Parmak izinden yüz tanıma”, ***Gazi Üniv. Müh. Mim. Fak. Der.***, 23 (4): 785-793 (2008).

19. Özdemir, A., “Dalgacık dönüşümünü kullanarak ön cepheden çekilmiş insan yüzü resimlerinin tanınması”, Yüksek Lisans Tezi, ***Kahramanmaraş Sütçü İmam Üniversitesi Fen Bilimleri Enstitüsü,*** Kahramanmaraş, 5-32 (2007).

20. Özdemır, A. ve Artıklar, M., "İki boyutlu dalgacik dönüsümü kullanarak ön cepheden çekilmis ınsan yüzü resimlerini tanima üzerine yaklasimlar", ***KSÜ Mühendislik Bilimleri Dergisi***, 12 (1): 6-9 (2009)

21. Kırtaç, K., "Gabor feature based face recognition using nearest neighbor discriminant analysis", Yüksek Lisans Tezi, ***İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü,*** İstanbul, 1-14 (2008)

22. Gündüz, H., “Altuzay temelli yaklaşımlar kullanarak gerçek zamanlı yüz tanıma”, Yüksek Lisans Tezi, ***Eskişehir Osmangazi Üniversitesi Fen Bilimleri Enstitüsü***, Eskişehir, 11-22 (2010)

23. Kern, D., Ekenel, H.K. and Stiefelhagen, R., “Illumination subspaces based robust face recognition”,***14th IEEE Signal Processing and Communications Applications Conference,*** Antalya, 1-4 (2006).

24. Kaçar, S. ve Çankaya, G., "Paralel hesaplama kullanilarak dogrusal olmayan sistemlerin analizi", ***6th International Advanced Technologies Symposium***, Elazığ, 135-139 (2011).

25. Luebke ,D., “Cuda: Scalable parallel programming for high-performance scientific computing”, ***Biomedical Imaging: From Nano to Macro 5th IEEE International Symposium on***, Trondheim, 838-838 (2008).

26. Boyer, M., Tarjan, D., Acton, S.T. and Skadron, K., "Accelerating leukocyte tracking using CUDA: A case study in leveraging manycore coprocessors", ***IEEE International Symposium on Parallel&Distributed Processing***, Boston, 1-12 (2009).

27. Muyan-Özçelik, P., Owens, J. D., Xia, J. and Samant, S. S., “Fast deformable registration on the GPU: A CUDA ımplementation of demons”, ***International Conference on Computational Sciences and Its Applications***, Perugia, 223-233 (2008).

28. Deschizeaux, B. and Blanc, J., “Imaging earth’s subsurface using CUDA”, GPU Gems 3, Hubert Nguyen, ***NVIDIA Corporation,*** Boston, 831-850 (2007)**.**

29. Biagio, A. D., Barenghi, A., Agosta, G. and Milano, P., “Design of a parallel AES for graphics hardware using the CUDA framework”, ***IEEE International Symposium on Parallel & Distributed Processing,*** Rome,1-8 (2009).

30. Che, S., Boyer, M., Meng, J., Tarjan, D., Sheaffer, J. W. and Skadron, K., “A performance study of general-purpose applications on graphics processors using CUDA”, ***J. Parallel Distrib. Comput.***, 68: 1370-1380 (2008) .

31. Arora, R., Tulshyan, R. and Deb, K., “Parallelization of binary and real-coded genetic algorithms on CUDA”, ***Evolutionary Computation IEEE Congress on***, Barcelona , 1-8 (2010).

32. Komatitsch, D., Michéaa, D. and Erlebacher, G. “Porting a high-order finite-element earthquake modeling application to NVIDIA graphics cards using CUDA”, ***J. Parallel Distrib. Comput.***, 69: 451-460 (2009).

33. Marzat, J., Dumortier, Y. and Ducrot, A., “Real-time dense and accurate parallel optical flow using CUDA”, ***WSCG***, Bory, 105-111 (2009).

34. Tsutsui, S. and Fujimoto, N., “Solving quadratic assignment problems by genetic algorithms with GPU computation: A case study”, ***GECCO’09***, Montréal Québec, 2523-2530 (2009).

35. Riaz, Z., Mirza, S.M. and Gilgiti, A., “Face recognition: using principal components and independent components”, ***IEEE International Multitopic Conference***, Lahore, Pakistan, 14–19 (2004).

36. Turk, M., and Pentland, A., "Eigenfaces for recognition", ***Journal of Cognitive Neuroscience***, 3: 71-86 (1991).

37. Çelebi, A. T., “Biyometrik tanıma”, Yüksek Lisans Tezi, ***Kocaeli Üniversitesi Fen Bilimleri Enstitüsü,*** Kocaeli, 4-21 (2008).

38. Graps, A.L., "An introduction to wavelets", ***IEEE Computational Sciencess and Engineering***, 2 (2): 50-61 (1995).

39. Garcia, C., Zikos, G. and Tziritas, G., "Wavelet packet analysis for face recognition", ***Image and Vision Computing***, 18: 289-297 (2000).

40. Brunelli, R. and Poggio, T., “Face recognition features vs. templates”, ***IEEE Transaction On Pattern Analysis and Machine Intelligence***, 15 (10): 1042-1053 (1993).

41. Kirby, M. and Sirovich, L., "Application of Karhunen-Loeve procedure for characterization of human faces", ***IEEE Transaction On Pattern Analysis and Machine Intelligence***, 12 (1): 103-108 (1990).

42. Sirovich, L. and Kirby, M., "Low dimensional procedure for characterization of human faces", ***J. Opt. Cos. Amer***., 4: 519-524 (1987).

43. Preuss, W. H., Teukolsky, S. A., Vetterling, W.T. and Flannery, B. P., "Eigensystems", Numerical Recipes In C: The Art of Scientific Computing 3_{nd} ed, ***Cambridge University Press***, New York, 474-481 (2007).

44. Taşaner, M. M., "Çoklu silindirik hedeflerin sınıflandırılması", Yüksek Lisans Tezi, ***Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü***, İstanbul, 27-35 (2009).

45. Fidan, S., "Dalgakılavuzunda yayılan elektromanyetik dalganın dalgacık dönüşümü ile modellenmesi", Yüksek Lisans Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü***, Ankara, 43-62 (2006).

46. Nanni, L. and Lumini, A., "Wavelet decomposition tree selection for palm and face authentication", ***Pattern Recognition Letters***, 29 (3): 343–353 (2008).

47. İnternet: NVIDIA Corporation, "What Is GPU Computing?" **http://www.nvidia.com/object/what-is-gpu-computing.html** (2011).

48. Riha, L. and Smid, R., “Acceleration of acoustic emission signal processing algorithms using CUDA Standard”, ***Computer Standards & Interfaces***, 33: 389-400 (2011).

49. İnternet: Advanced Micro Devices Inc, "Ati Stream Computing-Technical Overview"**http://developer.amd.com/gpu_assets/Stream_Computing_Overview.pdf** (2009).

50. İnternet: Khronos Group, "TheOpenCL Specification" **http://www.khronos.org/registry/cl/specs/opencl-1.0.pdf** (10/6/09).

51. İnternet: NVIDIA Corporation, “ NVIDIA CUDA C Programming Guide”, **http://www.nvidia.com/** (22/10/2010).

52. İnternet: NVIDIA Corporation, “CUDA CUBLAS Library" **http://developer.download.nvidia.com/compute/cuda/3%202%20prod/toolkit/docs/CUBLAS Library.pdf** (2010)

53. İnternet: NVIDIA Corporation, “CUDA CUFFT Library" **http://developer.download.nvidia.com/compute/cuda/3%202%20prod/toolkit/docs/CUFFT Library.pdf** (2010)

54. Nawata, T. and Suda, R., “APTCC : Auto parallelizing translator from C to CUDA”, ***International Conference on Computational Science***, Amsterdam, 352-361 (2011)

55. İnternet: Cambridge University AT&T Laboratories, " The Database of Faces" **http://www.cl.cam.ac.uk/Research/DTG/attarchive:pub/data/att_faces.zip** (2011)

56. İnternet: University of Stirling, "Psychological Image Collection at Stirling (PICS) 2D face sets" **http://pics.psych.stir.ac.uk/2D_face_sets.htm** (2011)

ÖZGEÇMİŞ

Kadriye ÖZ 1982 yılında Karabük'de doğdu; ilk ve orta öğrenimini aynı şehirde tamamladı. Karabük 75. Yıl Anadolu Lisesi 'nden mezun oldu. 2000 yılında Ege Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü'nde öğrenime başlayıp, 2005 yılında mezun oldu. 2009 yılında Karabük Üniversitesi, Teknik Eğitim Fakültesi, Elektronik ve Bilgisayar Eğitimi Bölümü'nde Araştırma Görevlisi olarak göreve başlamıştır. Halen Karabük Üniversitesi, Teknik Eğitim Fakültesi, Elektronik ve Bilgisayar Eğitimi Bölümü'nde göreve devam etmektedir.

ADRES BİLGİLERİ

Adres : Karabük Üniversitesi
Fen Bilimleri Enstitüsü
Balıklarkayası Mevkii / KARABÜK

Tel : 0370 4338200 / 1015
E-posta : kadriyeoz@karabuk.edu.tr