T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

GÜNCEL SEZGİSEL ALGORİTMALAR İLE VEKTÖR TABANLI GÖRÜNTÜ SIKIŞTIRMA

YÜKSEK LİSANS TEZİ

VEYSEL CAN DEMİR

DENİZLİ, HAZİRAN-2024

T.C.
PAMUKKALE ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

GÜNCEL SEZGİSEL ALGORİTMALAR İLE VEKTÖR TABANLI GÖRÜNTÜ SIKIŞTIRMA

YÜKSEK LİSANS TEZİ

VEYSEL CAN DEMİR

DENİZLİ, HAZİRAN-2024

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

Veysel Can DEMİR

ÖZET

GÜNCEL SEZGİSEL ALGORİTMALAR İLE VEKTÖR TABANLI GÖRÜNTÜ SIKIŞTIRMA
YÜKSEK LİSANS TEZİ
VEYSEL CAN DEMİR
PAMUKKALE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANABİLİM DALI

(TEZ DANIŞMANI:PROF.DR. CEYHUN KARPUZ)
(EŞ DANIŞMAN:DR. ÖĞR. Ü. İLKER KILIÇ)
DENİZLİ, HAZİRAN 2024

Günümüzde teknolojinin gelişmesiyle, oluşturulan görüntülerin kalitesi artmaktadır. Kalitesiyle birlikte bu görüntülerin boyutları ve yükleme/indirme süreleri de artmaktadır. Dolayısıyla bu durumda da daha fazla depolamaya ve zamana ihtiyaç vardır. Bu sorunlara çözüm olarak görüntülere çeşitli görüntü sıkıştırma teknikleri uygulanmaktadır.

Görüntü sıkıştırma teknikleri, kayıplı görüntü sıkıştırma ve kayıpsız görüntü sıkıştırma olmak üzere ikiye ayrılır. Kayıpsız görüntü sıkıştırma işleminde, görüntü kalitesi mümkün olduğunca değişmez fakat sıkıştırma oranı düşüktür. Kayıplı görüntü sıkıştırma işleminde görüntüde bulunan bazı piksellerin değiştirilmesi sayesinde daha fazla sıkıştırma yapılabilir fakat görüntünün kalitesi orijinaline göre daha düşük olacaktır.

Sezgisel algoritmalar, doğada yaşayan herhangi bir canlının yaşamsal faaliyetlerini baz alarak bu faaliyetlerin matematiksel olarak ifade edilmesiyle oluşmuştur. Bu yaşamsal faaliyetlere avlanma, keşif, kovalamaca, gözlemleme gibi faaliyetler örnek verilebilir. Bu çalışmada meyve sineklerinin besin arayışından esinlenilerek oluşturulmuş olan Meyve Sineği Optimizasyon Algoritması, ateş böceklerinin eş seçimlerinden esinlenilmiş olan Ateş Böceği Optimizasyon Algoritması, Kuşların uçuş hareketleri baz alınarak oluşturulmuş olan Parçacık Sürü Optimizasyon Algoritması ve yarasaların avlanma hareketlerinden ilham alınarak oluşturulmuş olan Yarasa Optimizasyon Algoritması incelenmiştir.

Güncel bir şekilde kullanılmaya devam eden bu algoritmaların desteği ile

literatürde yaygın bir şekilde kullanılmakta olan, çeşitli kontrastlara sahip olan, gri tonlamalı görüntüler üzerinde kayıplı görüntü sıkıştırma işlemi uygulanmıştır. Her algoritmanın çeşitli kontrastlardaki görüntüler üzerindeki performansları incelenip, hangi algoritmanın diğer algoritmalara kıyasla daha iyi sonuç verdiği incelenmiştir.

ANAHTAR KELİMELER: Sezgisel Algoritmalar, Görüntü Sıkıştırma, Parçacık Sürü Optimizasyon Algoritması, Ateş Böceği Optimizasyon Algoritması, K-Means Algoritması.

ABSTRACT

VECTOR-BASED IMAGE COMPRESSION BY CURRENT META-HEURISTIC ALGORITHMS
MSC THESIS
VEYSEL CAN DEMİR
PAMUKKALE UNIVERSITY INSTITUTE OF SCIENCE
ELECTRICAL AND ELECTRONICS ENGINEERING

(SUPERVISOR:PROF. DR. CEYHUN KARPUZ)
(CO-SUPERVISOR:ASSIST. PROF. DR. ILKER KILIÇ)
DENİZLİ, JUNE 2024

With the advancement of technology today, the quality of images created is increasing. Along with the quality, the sizes of these images and the time it takes to upload/download them are also increasing. Therefore, more storage and time are needed in this situation. Various image compression techniques are applied to images as a solution to these problems.

Image compression techniques are divided into two categories as lossless image compression and lossy image compression. In lossless image compression, the image quality remains as unchanged as possible, but the compression ratio is low. In lossy image compression, more compression can be achieved by changing some pixels in the image, but the quality of the image will be lower than the original.

Meta-heuristic algorithms have been developed based on the vital activities of any living creature in nature, expressing these activities mathematically. Activities such as hunting, exploration, chasing, and observation can be given as examples of these vital activities. In this study, the Fruit Fly Optimization Algorithm, inspired by the food search of fruit flies, the Firefly Optimization Algorithm, inspired by the mate selection of fireflies, the Particle Swarm Optimization Algorithm, based on the flight movements of birds, and the Bat Optimization Algorithm, inspired by the hunting movements of bats, have been examined.

With the support of these algorithms, which continue to be used

currently, lossy image compression has been applied to grayscale images that are widely used in the literature and have various contrasts. The performances of each algorithm on images with various contrasts have been examined, and it has been investigated which algorithm gives better results compared to other algorithms.

KEYWORDS: Meta-Heuristic Algorithms, Image Compression, Particle Swarm Optimization Algorithm, Firefly Optimization Algorithm, K-Means Algorithm.

İÇİNDEKİLER

Sayfa

ÖZET ..... **i**
ABSTRACT ..... **iii**
İÇİNDEKİLER ..... **v**
ŞEKİL LİSTESİ ..... **vi**
TABLO LİSTESİ ..... **viii**
SEMBOL LİSTESİ ..... **ix**
KISALTMALAR LİSTESİ ..... **x**
ÖNSÖZ ..... **xi**
1. GİRİŞ **1**
1.1 Tezin Amacı 2
1.2 Tezin Metodu 2
1.3 Tezin Kapsamı 3
2. LİTERATÜR ÖZETİ **4**
2.1 Problem Tanımı 5
3. VEKTÖR TABANLI GÖRÜNTÜLER **6**
4. SAYISAL GÖRÜNTÜLER VE ÖZELLİKLERİ **8**
4.1 Binary Görüntü 8
4.2 Gri Tonlamalı Görüntü 8
4.3 RGB Renkli Görüntü 9
4.4 Düşük Kontrastlı Görüntü 10
4.5 Yüksek Kontrastlı Görüntü 11
4.6 BMP (Bitmap) 12
4.7 PNG (Portable Network Graphics) 12
4.8 TIFF (Tagged Image File Format) 12
4.9 JPEG (Joint Photographic Expert Group) 12
5. GÖRÜNTÜ SIKIŞTIRMA TEKNİKLERİ **13**
5.1 Kayıpsız Görüntü Sıkıştırma Teknikleri 13
5.1.1 Run Length Kodlaması 13
5.1.2 Huffman Kodlaması 14
5.1.3 Lempel-Ziv-Welch Kodlaması (LZW) 14
5.2 Kayıplı Görüntü Sıkıştırma Teknikleri 15
5.2.1 Vektör Nicemleme (VN) (Vector Quantization (VQ)) 15
5.2.2 Ayrık Kosinüs Dönüşümü (AKD) 15
5.2.3 Ayrık Fourier Dönüşümü (AFD) 15
5.2.4 K-Means Algoritması 16
6. SEZGİSEL OPTİMİZASYON ALGORİTMALARI **19**
6.1 Meyve Sineği Optimizasyon Algoritması (MSOA) 19
6.2 Parçacık Sürü Optimizasyon Algoritması (PSOA) 22
6.3 Ateş Böceği Optimizasyon Algoritması (AOA) 23
6.4 Yarasa Optimizasyon Algoritması (YOA) 26
7. YÖNTEM **29**
8. BULGULAR **35**
9. SONUÇ VE ÖNERİLER **46**
10. KAYNAKLAR **49**
11. ÖZGEÇMİŞ **52**

ŞEKİL LİSTESİ

Sayfa

Şekil 3.1: Gri tonlamalı görüntünün vektörel gösterimi 6
Şekil 3.2: Kod kitabının oluşturulması..... 7
Şekil 4.1: Binary görüntü. 8
Şekil 4.2: Gri tonlamalı görüntü (Lena). 9
Şekil 4.3: RGB görüntü (Baboon)..... 9
Şekil 4.4: Gri tonlamalı düşük kontrastlı görüntü (Airplane) 10
Şekil 4.5: "Airplane" isimli görselin kontrast grafiği 10
Şekil 4.6: Gri tonlamalı yüksek kontrastlı görüntü (Barbara)..... 11
Şekil 4.7: "Barbara" isimli görselin kontrast grafiği..... 11
Şekil 5.1: Run-Length Tekniği..... 14
Şekil 5.2: Veri setinde rastgele merkezlerin seçimi 16
Şekil 5.3:Veri setinin kümelere ayrılması 17
Şekil 5.4: Yeni merkezlerin hesaplanması 17
Şekil 5.5: Kümelerin yeniden oluşturulması..... 18
Şekil 6.1: Beslenmekte olan bir meyve sineği 20
Şekil 6.2: PSOA'nın esinlenildiği canlı türlerinden biri 22
Şekil 6.3: Ateş böceği 24
Şekil 6.4: Bir mikro yarasa türü 26
Şekil 7.1: Kod kitabının oluşturulması (ikinci kez) 29
Şekil 7.2: Meyve Sineği Optimizasyon Algoritması işleyiş şeması 31
Şekil 7.3: Parçacık Sürü Optimizasyon Algoritması işleyiş şeması 32
Şekil 7.4: Ateş Böceği Optimizasyon Algoritması işleyiş şeması..... 33
Şekil 7.5: Yarasa Optimizasyon Algoritması işleyiş şeması..... 34
Şekil 8.1: Airplane görseli 35
Şekil 8.2: Barbara görseli..... 35
Şekil 8.3: Boat görseli..... 35
Şekil 8.4: Clock görseli..... 35
Şekil 8.5: Lena görseli..... 36
Şekil 8.6: Moon görseli..... 36
Şekil 8.7: MSOA'nın sonuçları a) Airplane, b) Barbara, c) Boat, d) Clock, e) Lena, f) Moon..... 38
Şekil 8.8: PSOA'nın sonuçları a) Airplane, b) Barbara, c) Boat, d) Clock, e) Lena, f) Moon..... 40
Şekil 8.9: AOA'nın sonuçları a) Airplane, b) Barbara, c) Boat, d) Clock, e) Lena, f) Moon..... 43
Şekil 8.10: YOA'nın sonuçları a) Airplane, b) Barbara, c) Boat, d) Clock, e) Lena, f) Moon..... 45
Şekil 9.1: Airplane için sonuçlar..... 46
Şekil 9.2: Barbara için sonuçlar 46
Şekil 9.3: Boat için sonuçlar 47
Şekil 9.4: Clock için sonuçlar 47
Şekil 9.5: Lena için sonuçlar..... 47

Şekil 9.6: Moon için sonuçlar 48

TABLO LİSTESİ

Sayfa

Tablo 7.1: Meyve Sineği Optimizasyon Algoritması sözde kodu 31
Tablo 7.2: Parçacık Sürü Optimizasyon Algoritması sözde kodu 32
Tablo 7.3: Ateş Böceği Optimizasyon Algoritması sözde kodu 33
Tablo 7.4: Yarasa Optimizasyon Algoritması sözde kodu..... 34
Tablo 9.1: Yeniden oluşturulan görüntülerin MSE değerleri..... 48
Tablo 9.2: Yeniden oluşturulan görüntülerin PSNR değerleri..... 48

SEMBOL LİSTESİ

X_i : Her bireyin x eksenindeki başlangıç konumu
X_{axis} : X eksenindeki o anki konum
Y_i : Her bireyin y eksenindeki başlangıç konumu
rand : Düzgün aralıkta dağıtılmış rastgele sayı
Y_{axis} : Y eksenindeki o anki konum
$Dist_i$: Her birey için hesaplanan mesafe değeri
S_i : Her bireyin koku konsantrasyon değeri
$Koku_i$: Her bireyin bireysel konumlarındaki koku konsantrasyonu değeri
S_i : Her bireyin koku konsantrasyonu değeri
Max(Koku) : En yoğun koku konsantrasyonu değeri
Kokubest : İterasyon sonundaki en yoğun koku konsantrasyonu değeri
p_i : Her bireyin kişisel en iyi konumu
g : Global en iyi konum
R_1 : Rastgele değerler alan düzenleme parametresi
R_2 : Rastgele değerler alan düzenleme parametresi
c_1 : Oran parametresi
c_2 : Oran parametresi
v_i : Her birey için hız parametresi
r_{ij} : İki bireyin arasındaki mesafe
R_1 : Rastgele değerler alan düzenleme parametresi
R_2 : Rastgele değerler alan düzenleme parametresi
$\beta(r)$: Çekicilik değeri
α : Yakınsama değeri
f_{min} : Minimum frekans değeri
f_{max} : Maximum frekans değeri
A_i : Her bireyin ses şiddeti değeri
β : Yakınsama değeri
A_0 : Ortalama ses şiddeti değeri
r_i : Her bireyin sinyal gönderme oranı
d : Belirli bir aralıkta değer alan rastgele değer
$\gamma^{iter\ no}$: Yakınsama değeri

KISALTMALAR LİSTESİ

MSOA: Meyve Sineği Optimizasyon Algoritması
PSOA : Parçacık Sürü Optimizasyon Algoritması
AOA : Ateş Böceği Optimizasyon Algoritması
VN : Vektör Nicemleme
YOA : Yarasa Optimizasyon Algoritması
MSE : Ortalama Karesel Hata
PSNR : Tepe Sinyal-Gürültü Oranı
AKD : Ayrık Kosinüs Dönüşümü
AFD : Ayrık Fourier Dönüşümü
BMP : Bitmap görüntü formatı
PNG : Portable Network Graphics
TIFF : Tagged Image File Format
JPEG : Joint Photographic Expert Group

ÖNSÖZ

Tez çalışmam boyunca karşılaştığım bütün zorluklarda yanımda olan, beni destekleyen aileme çok teşekkür ederim.

Çalışma süreci boyunca beni destekleyen, kendi bilgi ve deneyimleriyle bana yol gösteren değerli tez danışmanlarım saygı değer Dr. Öğr. Ü. İlker KILIÇ ve Prof. Dr. Ceyhun KARPUZ'a saygılarımı sunar, kendilerine çok teşekkür ederim.

1. GİRİŞ

Günümüzde internet ve uygulamalar sayesinde veriler kolayca ve hızlıca oluşturulabilmektedir. Oluşturulan verilerin arasında özellikle görüntüler diğer veri tiplerine göre daha fazla yer kaplar. Özellikle günümüzde oluşturulan görüntülerin kalitesinin artmasıyla birlikte indirme-yükleme süreleri ve hafızada kapladığı alan da artmaktadır. Bu yüzden görüntünün hafızada kapladığı alanın küçültülebilmesi için sıkıştırılması gerekmektedir. Sıkıştırma işlemi görüntüdeki fazlalıkları azaltır ve görüntünün mümkün olduğunca küçük boyutlu olmasını sağlar. Görüntü sıkıştırma işlemi, kayıplı ve kayıpsız olmak üzere ikiye ayrılır (Rahebi 2022). Kayıplı görüntü sıkıştırma işlemlerinden en yaygın ve pratik olanlarından biri K-Means Algoritmasıdır. Bu sıkıştırma yöntemi, sıkıştırma sonrasında bazı görüntü bilgilerinin kaybolmasından dolayı kayıplıdır. Bu sıkıştırma yönteminde bir görüntü rastgele büyüklükte kümelere bölünür ve her kümeye rastgele bir merkez atanır. Daha sonrasında belirlenen merkezler ile kümeler içerisinde bulunan verilerin Öklid mesafesi hesaplanır. Ardından her küme içerisinde ölçülen mesafelerin ağırlık ortalaması alınır ve kümelerin yeni merkezleri belirlenir. (Li ve diğ. 2012).Bu yöntem sayesinde, orijinal görüntüdeki ana hatların bozulmadan görülebilecek şekilde, orijinal görüntüye kıyasla daha düşük kalitede olan ancak hafızada orijinal görüntüden daha az yer kaplayan yeni bir görüntü oluşturulur (Yuan ve diğ. 2014). Sezgisel algoritmalar, doğadaki herhangi bir canlının hareketlerinin (besin arama, uçuş, gözlemleme, avlanma, keşif, kovalamaca vb.) gözlemlenip belirli matematiksel ifadeler ile bu hareketlerin simüle edilmesine dayalıdır. Bu algoritmalar sayesinde herhangi bir mühendislik alanındaki problem kolayca çözüme ulaşabilir, bir sistemdeki belirli bir parametrenin en ideal değeri hesaplanabilir. Bu çalışmada K-Means algoritmasının işleyiş mantığı kullanılarak kod defterleri oluşturulmuş, sezgisel algoritmaların desteği ile çeşitli kontrastlardaki görüntülere görüntü sıkıştırma işlemi uygulanmıştır. Çeşitli sezgisel algoritmaların görüntü sıkıştırma işleminde kullanılması sonucu algoritmaların performansları karşılaştırılmıştır. İncelenmiş olan algoritmalar şunlardır: Meyve Sineği Optimizasyon Algoritması (MSOA), Parçacık Sürü Optimizasyon Algoritması (PSOA), Ateşböceği Optimizasyon Algoritması (AOA) ve Yarasa Optimizasyon Algoritmasıdır (YOA).

Performans değerlendirmesi için MSE (Mean Squared Error) yani Ortalama Karesel Hata değeri incelenecektir. Bu değer ne kadar küçükse algoritmanın performansının o kadar iyi olduğu sonucuna ulaşılır. Yani amaç MSE değerinin minimize edilmesidir. Bu çalışmada işlem kolaylığı açısından 256x256 boyutunda, gri renk seviyeli, çeşitli kontrast oranlarına sahip görüntüler kullanılmıştır.

1.1 Tezin Amacı

Bu çalışmada literatürde yaygın bir şekilde kullanılmakta olan sezgisel algoritmalarıyla birlikte K-Means algoritmasından ilham alınarak oluşturulmuş olan çözüm kümeleri ile literatüre yaygın olarak kullanılan gri tonlamalı 256x256 boyutundaki yüksek ve düşük kontrastlı görüntülere görüntü sıkıştırma işlemi uygulanacaktır. Şekil 3.2'de de görüldüğü üzere uygulanacak olan görüntü sıkıştırma işlemi için görüntüyü en iyi şekilde temsil eden 8 adet, 4x4 boyutunda olan kod sözcüklerinden oluşan bir kod kitabının bulunması gerekmektedir. Bu çalışmada literatürde başarılı olarak kabul edilen güncel sezgisel algoritmalar kullanılarak en iyi kod kitabı bulunmuştur. Yapılan bu işlemin ardından, incelenen sezgisel algoritmaların performansları karşılaştırılıp, aralarından en verimli çalışan ve en iyi sonucu veren sezgisel algoritmanın hangisi olduğu belirlenecektir.

1.2 Tezin Metodu

Bu çalışmada görüntülerin üzerine görüntü sıkıştırma uygulanabilmesi için öncelikle çözüm kümesinin nasıl tasarlanması gerektiği düşünülmüş olup K-Means Algoritmasının çalışma biçiminden ilham alınmıştır. Borland C++ Builder ortamında çözüm kümesi tasarımı için uygun bir kod yazılıp, yazılan kodun çalışma mantığı doğrulandıktan sonra literatürde yaygın olarak kullanılan sezgisel algoritmalar hakkında araştırma yapılmıştır. Yapılan araştırmalar sonucunda bu çalışmada, Meyve Sineği Optimizasyon Algoritması, Parçacık Sürü Optimizasyon Algoritması, Ateş Böceği Optimizasyon Algoritması ve Yarasa Optimizasyon Algoritmasının kullanılmasına karar verilmiştir.

Yukarıda bahsi geçen algoritmalar için çeşitli çalışmalar incelenip, algoritmaların işleyiş mantıkları kod ortamında simüle edilmiştir. Mevcut algoritma için yapılan simülasyonun doğruluğu teyit edildikten sonra literatürde yaygın olarak kullanılan, çeşitli kontrastlardaki gri tonlamalı görüntüler üzerinde incelenen bu algoritmalarla birlikte görüntülere görüntü sıkıştırma işlemi uygulanmıştır. Çalışma sonucunda elde edilen sıkıştırılmış görüntülerin MSE değerleri bilgisayar ortamında kayıt altına alınmıştır. Ardından bir sonraki algoritmanın kod ortamında simüle edilmesi işlemine geçilip incelenen sezgisel algoritma ile görüntü sıkıştırma işlemi yapılmıştır. Hedeflenen sezgisel algoritmaların hepsiyle görüntü sıkıştırma işlemleri yapıldıktan sonra kayıt altına alınan MSE değerleri Octave programı yardımıyla oluşturulan çeşitli grafikler ile incelenmiştir. Dört farklı sezgisel algoritması kullanılarak yapılan görüntü sıkıştırma işleminden elde edilen sonuçlar görüntü sıkıştırma işlemi uygulanmış olan her bir görüntü için karşılaştırılmıştır.

1.3 Tezin Kapsamı

Bu çalışmada görüntü işleme alanında bulunan görüntü sıkıştırma teknikleri ve çeşitli güncel sezgisel algoritmalar incelenmiş olup, K-Means algoritmasından ilham alınarak görüntü sıkıştırma işlemi için çözüm kümeleri oluşturulmuş ve oluşturulan bu çözüm kümeleri kullanılarak Meyve Sineği Optimizasyon Algoritması, Parçacık Sürü Optimizasyon Algoritması, Ateş Böceği Optimizasyon Algoritması ve Yarasa Optimizasyon Algoritması ile düşük ve yüksek kontrastlı görüntülere görüntü sıkıştırma işlemi uygulanmış ve bu algoritmaların görüntü sıkıştırma işlemindeki performansları karşılaştırılmıştır.

2. LİTERATÜR ÖZETİ

Görüntü sıkıştırma işlemi, görüntünün boyutunun küçültülüp daha küçük depolama alanı gerektiren veya daha düşük bant genişliği kullanılarak iletilmesini sağlayan bir işlemdir. Görüntü sıkıştırma işlemi yüksek kalitedeki görüntülerin yüksek boyutlarda depolama alanı ve fazla bant genişliği gerektirmesi gibi problemlerin çözülmesinde önemlidir. Görüntü sıkıştırma işlemi sayesinde depolama alanının daha verimli bir şekilde kullanılması sağlanmış olur, veri paylaşımı hızlanır ve veri yönetimi süreçleri optimize edilmiş olur. Görüntü sıkıştırma işleminde algoritmaların kullanılmasıyla da büyük boyutlu veya yüksek çözünürlüklü görüntülerin depolanması ve taşınması daha verimli hale getirilmiş olur.

Vektör Nicemleme (Vector Quantization) yöntemi, görüntü sıkıştırma alanında yaygın olarak kullanılan, temel amacı verileri benzerliklerine göre sıralayıp gruplandırmayı ve bu grupların ortalamalarını belirleyerek sıkıştırmayı amaçlayan bir yöntemdir. Vektör Nicemleme yönteminde, görüntü öncelikle küçük bloklara bölünür ve her blok vektör olarak değerlendirilir. Bu vektörler kod kitabı adı verilen bir kümelerde gruplandırılırlar ve her bir vektör kod kitabından en yakın vektöre atanır. Böylece, görüntüdeki birçok blok aynı kod kitabındaki vektörlerle temsil edilebilir ve böylece veri miktarı azaltılabilir. Kısacası kod kitabı, orijinal görüntünün mümkün olduğunca benzer şekilde temsil edilmesini sağlar. En ideal kod kitabının üretimi, görüntü sıkıştırma işleminin verimliğini ve performansını iyileştirmek için önemli bir aşamadır.

Kod kitabının oluşturulması, sıkıştırma oranı ve kalitesi için önemlidir. Benzer özelliklere sahip olan blokların optimum düzeydeki bir kod kitabında daha iyi gruplandırılması sayesinde daha yüksek sıkıştırma oranları elde edilebilir (Kılıç ve Çetin 2024). Dolayısıyla herhangi bir görüntü için kullanılabilecek olan en ideal kod kitabını bulmak görüntü sıkıştırma işleminin en önemli kısımlarından biridir. En ideal kod kitabının bulunması sayesinde görüntü kalitesi korunur, Tepe sinyal-gürültü oranı (PSNR) yükselir, Ortalama Karesel Hata (MSE) değeri ve görüntünün boyutu azalır (Rahebi 2022). Kod kitabının oluşturulması sürecinde sezgisel algoritmaların kullanılmasıyla Vektör Nicemleme yöntemindeki verilerin kümelere

ayrılma işlemi daha verimli bir şekilde yapılabilir. Bu tür algoritmaların temel amacı, popülasyonu oluşturan bireylerin evrimleşmesiyle daha iyi bireylerin yaratılması ve bu sayede optimizasyon problemlerinin çözülebilmesidir (Kılıç ve Çetin 2024).

Literatürde, optimizasyon algoritmalarını kullanarak kod kitabı tasarımı yapılan birçok çalışma bulunmaktadır. Örneğin ,2008 yılında Yang tarafından kullandığı Genetik Algoritma, 2013 yılıda Tsai ve diğer çalışma arkadaşları tarafından kullanılan Karınca Kolonisi Optimizasyon Algoritması, 2007 yılında Feng ve diğer çalışma arkadaşları tarafından kullanılan Parçacık Sürü Optimizasyon Algoritması, 2011 yılında Horng ve Jiang tarafından kullanılan Bal Arısı Çiftleşme Optimizasyon Algoritması, 2012 yılında Horng tarafından kullanılan Ateş Böceği Optimizasyon Algoritması, 2007 yılında Pan ve Cheng tarafından kullanılan Tabu Arama Algoritması, 2019 yılında Kumar ve diğer çalışma arkadaşları tarafından kullanılan Yarasa Optimizasyon Algoritması, 2018 yılında Chiranjeevi ve Jena tarafından kullanılan Guguk Kuşu Arama Optimizasyon Algoritması, 2021 yılında Geetha ve diğer çalışma arkadaşları tarafından kullanılan Aslan Optimizasyon Algoritması, 2013 yılında Sanyal ve diğer çalışma arkadaşları tarafından kullanılan Bakteriyel Yiyecek Arama Optimizasyon Algoritması ve 2022 yılında Rahebi tarafından kullanılan Balina Optimizasyon Algoritması vb. çeşitli algoritmalar görüntü sıkıştırma işleminde, Vektör Nicemleme yöntemindeki kod kitabı tasarımı için kullanılabilir (Kılıç ve Çetin 2024).

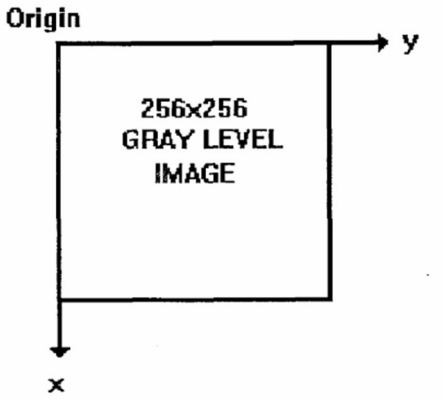
2.1 Problem Tanımı

Günümüzde teknolojinin gelişmesiyle, oluşturulan görüntülerin kalitesi artmaktadır. Kalitesiyle birlikte bu görüntülerin boyutları ve yükleme/indirme süreleri de artmaktadır. Dolayısıyla bu durumda da daha fazla depolamaya ve zamana ihtiyaç vardır. Süre ve alandan tasarruf edilmesi için görüntü sıkıştırma teknikleri kullanılmaktadır.

3. VEKTÖR TABANLI GÖRÜNTÜLER

Gri seviye görüntü veya basitçe görüntü terimi, x ve y'nin uzaysal koordinatları ve herhangi bir noktadaki (x,y) fonksiyon değerinin o noktadaki görüntünün parlaklığına (veya gri ton seviyesine) orantılı olan iki boyutlu bir ışık yoğunluğu fonksiyonunu ifade eder. Şekil 3.1, bu çalışma boyunca kullanılan eksen kuralını göstermektedir. Bazen bir görüntü fonksiyonunu üçüncü eksen parlaklık olacak şekilde perspektif olarak görüntülemek kullanışlı olabilir. Şekil x.x'e bu açıdan bakıldığında, parlaklık seviyelerinde çok sayıda değişiklik olan bölgelerde bir takım aktif tepe noktaları ve parlaklık seviyelerinin çok az değiştiği veya sabit olduğu yerlerde daha küçük tepeler olarak görünecektir. Daha parlak alanlara orantılı olarak daha yüksek değerler atamak, çizimdeki bileşenlerin yüksekliğini ilgili parlaklıkla orantılı hale getirecektir.

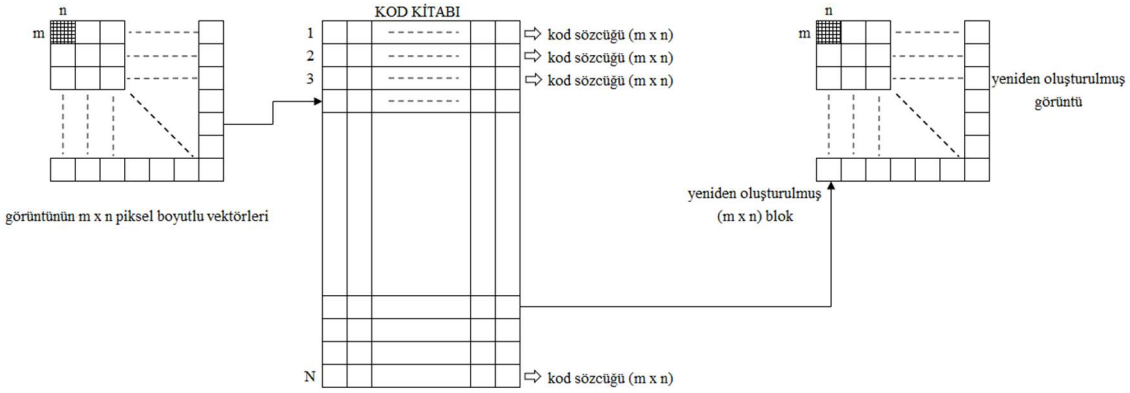
Dijital bir görüntü, satır ve sütun indisleri o noktada tanımlanan bir matris olarak düşünülebilir. Bu dijital dizinin elemanlarına piksel denir (Kılıç 1997).



Şekil 3.1: Gri tonlamalı görüntünün vektörel gösterimi (Kılıç 1997).

Şekil 3.1'de görüldüğü üzere bu çalışmada kullanılan görüntüler vektör tabanlı olarak incelenmiştir. Gri tonlamalı 256x256 görüntülerde vektörün sınırları 256 birimdir. Yani görüntü 256x256 boyutunda bir vektör olarak incelenmiştir.

Görüntünün vektör olarak değerlendirmesi sayesinde çözüm kümesi tasarımında pratiklik sağlanmıştır.



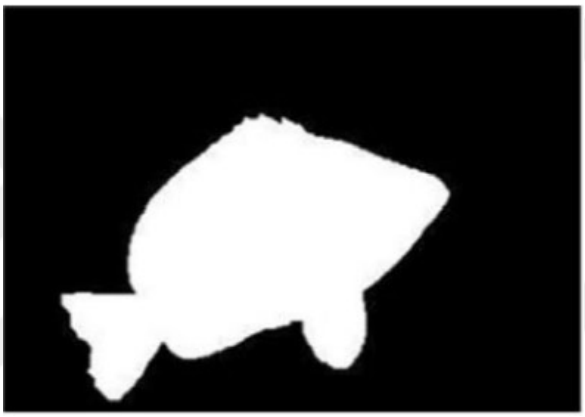
Şekil 3.2: Kod kitabının oluşturulması.

Şekil 3.2'de görüldüğü üzere herhangi bir görüntünün vektör olarak değerlendirilmesinin temsili hali gösterilmektedir. Şekil 3.2'de görüntüyü oluşturan her bir kare 4x4 boyutundadır yani her bir karenin içerisinde görüntüye ait olan 16 piksel mevcuttur. Yukarıdaki şekilde gösterilen kareler temsili olup orijinal görüntü 256x256 boyutunda bir vektör olarak ele alınıp 4096 adet kare elde edilmiştir. Görüntünün daha kolay bir şekilde işlenebilmesi için görüntü 4x4 boyutunda vektörlere bölünür. Bu vektörlere kod sözcüğü adı verilir. 256x256 boyutundaki bir görüntüde toplamda 4096 adet kod sözcüğü mevcuttur. Bu kod sözcüklerinin bir araya gelmesiyle de kod kitabı adı verilen vektör grupları oluşturulur. Bu çalışmada 8 adet kod sözcüğünden oluşan kod kitabı kullanılmıştır.

4. SAYISAL GÖRÜNTÜLER VE ÖZELLİKLERİ

4.1 Binary Görüntü

0 ve 1 olmak üzere yalnızca iki farklı piksel yoğunluğu değerine sahip görüntülere binary (ikili) görüntüler denir. Şekil 4.1'deki gibi görüntüler genellikle renkli bir görüntünün ayırt edici bir bölümünü vurgulamak için kullanılır (Dai ve diğ. 2015).



Şekil 4.1: Binary görüntü (Dai ve diğ. 2015).

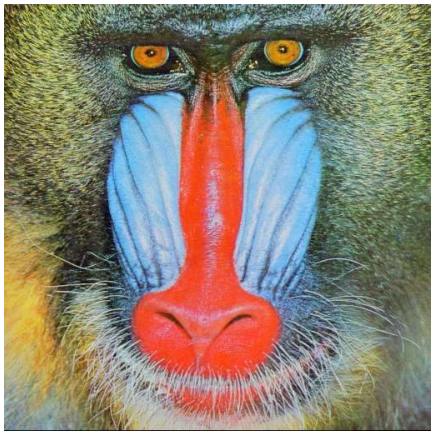
4.2 Gri Tonlamalı Görüntü

Gri tonlamalı veya 8 bit görüntüler, 256 farklı piksel değerinden oluşur; burada 0 piksel yoğunluğu siyah rengi ve 255 piksel yoğunluğu beyaz rengi temsil eder. Aradaki diğer 254 değerin tümü farklı gri tonlarıdır (Huang ve diğ. 2021). Şekil 4.2'de literatürde yaygın olarak kullanılan gri tonlamalı bir örnek görüntü gösterilmiştir.

Şekil 4.2: Gri tonlamalı görüntü (Lena).

4.3 RGB Renkli Görüntü

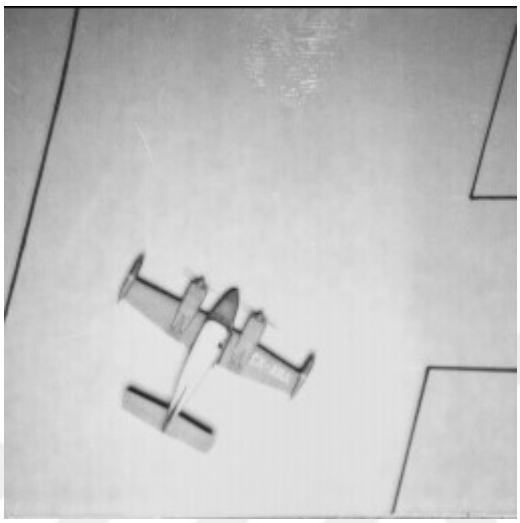
Modern dünyada alıştığımız görüntüler RGB veya bilgisayarlara 16 bitlik matrisler olan renkli görüntülerdir. Yani her piksel için 16777216 farklı renk mümkündür. "RGB" bir görüntünün Kırmızı, Yeşil ve Mavi tonlarını temsil eder. Bir piksel, piksel değeri (0, 0, 0) olduğunda siyah, (255, 255, 255) olduğunda beyaz olacaktır. Aradaki herhangi bir sayı kombinasyonu, doğada var olan tüm farklı renkleri meydana getirir. Örneğin, (255, 0, 0) kırmızı renktir (çünkü bu piksel için sadece kırmızı kanal aktiftir). Benzer şekilde (0, 255, 0) yeşil ve (0, 0, 255) mavidir (Daga ve diğ. 2012). Şekil 4.3'de literatürde yaygın olarak kullanılan bir RGB örnek görüntü gösterilmiştir.



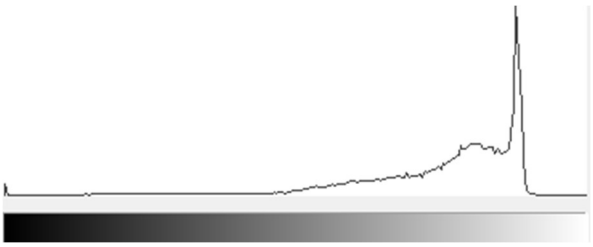
Şekil 4.3: RGB görüntü (Baboon) (Daga ve Yusiong 2012).

4.4 Düşük Kontrastlı Görüntü

Görüntüyü oluşturan piksellerdeki renk değişimlerinin daha az olduğu görüntülerdir. Bu tip görüntülerde çok fazla renk tonu bulunmaz. Şekil 4.4'teki görüntü düşük kontrastlı ve gri tonlamalı "Airplane" görselidir.



Şekil 4.4: Gri tonlamalı düşük kontrastlı görüntü (Airplane).



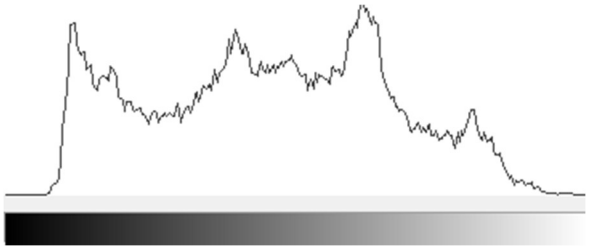
Şekil 4.5: "Airplane" isimli görselin kontrast grafiği.

Şekil 4.5'te görüldüğü gibi düşük kontrastlı görüntülerde piksellerin renk yoğunluğu, belirli bir gri ton aralığında toplanmıştır.

4.5 Yüksek Kontrastlı Görüntü

Görüntüyü oluşturan piksellerdeki renk değişimleri daha fazladır. Şekil 4.6'daki gibi bu tür görüntülerde düşük kontrastlı görüntülere kıyasla daha fazla detay ve renk tonu bulunur.

Şekil 4.6: Gri tonlamalı yüksek kontrastlı görüntü (Barbara).



Şekil 4.7: "Barbara" isimli görselin kontrast grafiği.

Şekil 4.7'de görüldüğü gibi yüksek kontrastlı görüntülerde piksellerin renk yoğunluğu, bütün ton skalasına yayılmış bir durumdadır.

Bu çalışmada literatürde yaygın bir şekilde kullanılan yüksek ve düşük kontrastlı, gri tonlamalı ve 256x256 boyutta görüntüler kullanılmıştır.

4.6 BMP (Bitmap)

BMP görüntü, pratik bir görüntü dosyası formatıdır. Görüntünün ham hali olup sıkıştırılma uygulanmamış halidir. Bu çalışmada BMP formatındaki literatürde yaygın olarak kullanılan görüntüler kullanılacaktır.

4.7 PNG (Portable Network Graphics)

PNG formatındaki görüntüler kayıpsız veri sıkıştırması kullanılarak oluşturulmuştur. LZ77 algoritması ve Huffman kodlamasının bir arada kullanan DEFLATE sıkıştırma algoritması kullanılarak oluşturulmuştur. Görüntülerin internette dağıtımı için tasarlanmıştır.

4.8 TIFF (Tagged Image File Format)

Fotoğraflar ve çizimleri depolamak için kullanılan kayıpsız sıkıştırma yapan bir dosya formatıdır. JPEG ve PNG'ye kıyasla daha yüksek renk derinliğine sahip görüntüler için kullanılan bir formattır. Bu format görüntü işleme uygulamaları ve fakslama gibi alanlarda kullanılır (Kaur & Choudhary, 2016).

4.9 JPEG (Joint Photographic Expert Group)

Renkli veya gri tonlamalı görüntüleri sıkıştırmayı amaçlayan bir formattır. Kayıplı sıkıştırma yapan bir formattır. Sıkıştırma oranının ayarlanabilir olması bu formatı popüler kılmaktadır. Sıkıştırma oranı arttıkça görüntüdeki kaybolan verilerin miktarı artar fakat yeniden oluşturulan görüntü, orijinaline kıyasla daha küçük boyutta olur. JPEG formatı, görüntülerin internette depolanabilmesi ve iletilmesi için kullanılan yaygın bir formattır.

5. GÖRÜNTÜ SIKIŞTIRMA TEKNİKLERİ

Dijital görüntülemede görüntünün temsil edilebilmesi için gerekli olan bilgi miktarının azaltılması sorununu aşabilmek için görüntüye, görüntü sıkıştırma işlemi uygulanır. Görüntünün kompakt bir temsilini oluşturmayı ve bu sayede görüntü depolama boyutlarını ve iletim sürelerini azaltmayı hedefleyen bir işlemdir. Her görüntü içerisinde gereksiz veriler mevcuttur ve bu veriler fazladan depolama alanı kullanımına sebep olur. Bu tip gereksiz veriler, görüntü boyunca tekrar eden pikseller veya çok sık tekrarlanmış olan desenler olabilir. Görüntü sıkıştırma işlemi ile bu fazlalıkların azaltılması veya ortadan kaldırılması depolama alanından tasarruf edilebilmesini sağlar.

Görüntü sıkıştırma işlemi kayıplı ve kayıpsız olmak üzere iki farklı türde incelenebilir. Kayıpsız sıkıştırma işlemiyle yeniden oluşturulan görüntü, sayısal değerler bakımından orijinal görüntü ile aynıdır. Fakat kayıpsız sıkıştırma işlemiyle çok az miktarda sıkıştırma uygulanabilir. Kayıplı sıkıştırma işlemiyle yeniden oluşturulan görüntüde ise orijinal görüntüye göre bozulmalar görülür. Bunun nedeni, görüntüdeki gereksiz bilgilerin ortadan kaldırılmasıdır. Ancak kayıplı sıkıştırma işlemi sayesinde oldukça yüksek oranlarda sıkıştırma yapabilmek mümkündür (Vijayvargiya ve diğ. 2013).

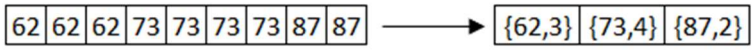
5.1 Kayıpsız Görüntü Sıkıştırma Teknikleri

Kayıpsız görüntü sıkıştırma işleminde orijinal görüntüdeki tüm bilgilerin kodlanması ile görüntü yeniden oluşturulur. PNG ve GIF, kayıpsız görüntü sıkıştırmaya örnek olarak gösterilebilir.

5.1.1 Run Length Kodlaması

Run-length kodlama (RLK), Şekil 5.1'de görüldüğü üzere orijinal görüntüdeki tekrar eden verileri birleştirilip tek veri değeri olarak düzenleyip

görüntüyü yeniden oluşturan bir tekniktir. Sıralı ve tekrar eden veriler için kullanıldığında oldukça verimli bir tekniktir. Gri tonlamalı görüntü için $\{V_i,R_i\}$ şeklinde belirtilir. "V_i" değeri piksel değeri, "R_i" ise o pikselden kaç adet bulunduğunu gösteren bir değerdir. Bu işlem düşük kontrastlı görüntülerde verimli bir şekilde çalışırken yüksek kontrastlı görüntülerde sıkıştırma yapamaz ve orijinal görüntüden daha yüksek boyutlu bir görüntü ortaya çıkar (Kaur & Choudhary, 2016). Şekil 5.1'de Run-length Tekniği'nin çalışma mantığı temsili olarak gösterilmiştir.



Şekil 5.1: Run-Length Tekniği.

5.1.2 Huffman Kodlaması

Huffman tarafından geliştirilmiş kayıpsız görüntü sıkıştırma yöntemlerinden birisidir. Görüntüdeki pikseller sembol olarak değerlendirilir. Daha sık görünen piksellere daha az sayıda bit ataması yapılırken, daha seyrek görünen piksellere daha fazla sayıda bit ataması yapılır (Vijayvargiya ve diğ. 2013).

5.1.3 Lempel-Ziv-Welch Kodlaması (LZW)

Abraham Lempel, Jacob Ziv ve Terry Welch tarafından geliştirilmiş, evrensel kayıpsız veri sıkıştırma algoritmasıdır. LZW sözlük tabanlı kodlamadan oluşturulmuş bir algoritmadır. Sözlük tabanlı kodlama statik veya dinamik olabilir. Statik sözlük kodlamada, kodlama ve kod çözme işlemleri yapılırken sözlük sabit kalır. Dinamik sözlük kodlamasında ise sözlük anlık olarak güncellenir. Uygulama kısmı basittir ve yüksek verimlilik oranlarına sahiptir, bu yüzden de bilgisayarlarda yaygın olarak kullanılan evrensel bir veri sıkıştırma algoritmasıdır (Vijayvargiya ve diğ. 2013).

5.2 Kayıplı Görüntü Sıkıştırma Teknikleri

Kayıplı görüntü sıkıştırma işlemi, depolama boyutunun azaltılabilmesi için görüntüdeki bazı verilerin silinip görüntünün yeniden oluşturulmasıdır. Oluşturulan yeni görüntü orijinal görüntüye benzer fakat birebir aynı görüntü değildir. Veri kaybı olduğundan dolayı oluşturulan yeni görüntü orijinal görüntünün daha düşük kalitedeki halidir. Böylece kaliteden ödün verilerek kayıpsız görüntü sıkıştırma işlemine oranla daha yüksek sıkıştırma oranları elde edilebilir.

5.2.1 Vektör Nicemleme (VN) (Vector Quantization (VQ))

Vektör nicemleme işlemi, büyük bir veri kümesini gruplara bölerek çalışır. Her grubun ortalaması alınır ve bir merkez belirlenir. Artık gruplara ayrılmış olan her veri merkez noktanın değeri ile gösterilir. Böylece birbirlerine yakın değerde olan veriler tek bir değer ile gösterilip veri boyutundan kazanç sağlanmış olur. Bu tekniğin yoğunluk eşleştirme özelliği sayesinde büyük boyutlu verileri kolayca gruplayabilir.

5.2.2 Ayrık Kosinüs Dönüşümü (AKD)

Görüntü sıkıştırma teknikleri içinde en popüler olan tekniklerden biridir. AKD tekniği görüntü ve video işlemede, sinyal işlemede kullanılmaktadır. Görüntüdeki piksel değerlerini iki boyutlu matris olarak değerlendirirsek AKD, frekans bileşenlerinin eşdeğer matrisine dönüştürülmesi işlemidir. Görüntü uzayının belirli bir frekansa eşlenmesi için kullanılır. Bu sayede görüntü sıkıştırılmış olur (Raid ve diğ. 2014).

5.2.3 Ayrık Fourier Dönüşümü (AFD)

Ayrık Fourier Dönüşümü (AFD), sonlu alana sahip olan ayrık zamanlı sinyallerin Fourier analizinin incelenmesi tekniğidir. Görüntü, video ve ses bilgilerinin sıkıştırılmasının analizi gibi birçok alanda kullanılan bir tekniktir. AFD

tekniği hesaplama yapılabilmesi için fazla sayıda çarpma ve toplama işlemi gerektirir. Bu yüzden bu teknik yerine Hızlı Fourier Dönüşümü de kullanılabilir (Ismaili ve diğ. 2012).

5.2.4 K-Means Algoritması

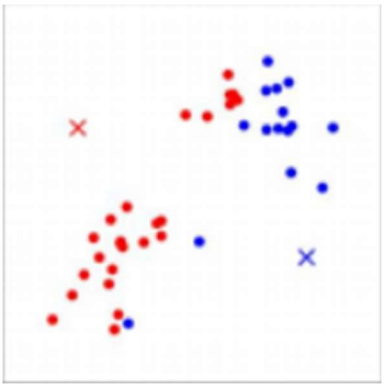
K-Means algoritması, J.B.MacQueen tarafından önerilen, bölmeye dayalı bir tür küme algoritmasıdır. Örüntü tanıma ve veri madenciliğinde sıklıkla kullanılan bir algoritmadır. Temel mantığı, kümelenmiş verilerin küme merkezlerine olan uzaklığın hesaplanıp her iterasyonda merkezin güncellenmesine dayalıdır (Li ve Wu 2012). Algoritma şu şekilde çalışmaktadır:

Adım 1: Bir K sayısı belirlenir. Bu K sayısı merkez sayısını yani küme sayısını belirler.

Adım 2: Belirlenen K sayısı kadar veri setinde rastgele küme merkezleri Şekil 5.2'de gösterildiği gibi belirlenir.

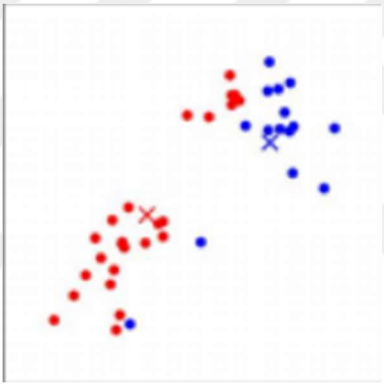
Şekil 5.2: Veri setinde rastgele merkezlerin seçimi (Piech 2013).

Adım 3: Veri setindeki verilerin, Şekil 5.3'teki gibi belirlenen merkezlere göre Öklid mesafesi hesabı yapılır ve her veri hangi merkeze yakın ise o merkezin kümesine ait kabul edilir.



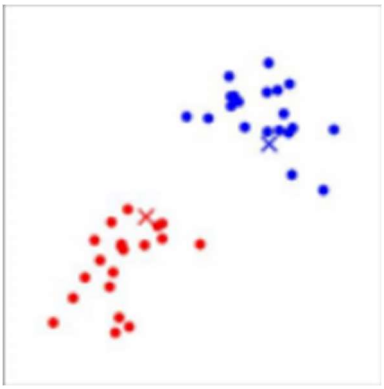
Şekil 5.3: Veri setinin kümelere ayrılması (Piech 2013).

Adım 4: Her kümede, küme merkezleri ile kümedeki verilerin arasındaki Öklid mesafesi hesaplanır ve ortalama bir değer hesaplanır.



Şekil 5.4: Yeni merkezlerin hesaplanması (Piech 2013).

Adım 5: Hesaplanan bu ortalama değer kümenin yeni merkezini oluşturur ve 3. adımdan itibaren bu işlemler belirli kriterler sağlanana kadar devam eder.



Şekil 5.5: Kümelerin yeniden oluşturulması (Piech 2013).

Bu çalışmada K-Means algoritmasının küme seçim mantığından esinlenilerek, literatürde yaygın olarak kullanılan gri tonlamalı görüntülerdeki rastgele kod sözcüklerinin (code word) alt alta sıralanması ile kod defterleri (code book) oluşturulmuştur. Oluşturulan bu kod defterleri ile birlikte çeşitli algoritmaların yardımı ile görüntüler yeniden oluşturulup, görüntü sıkıştırma işlemi gerçekleştirilmiştir.

6. SEZGİSEL OPTİMİZASYON ALGORİTMALARI

Son yıllarda artan doğanın işleyişini inceleyen araştırmalar sayesinde, doğanın karmaşık problemleri çözmek ve bir sistem geliştirmek için elverişli olduğu öne sürülüyor. Örneğin doğada, hayvanları ele alacak olursak hayvanların, besin arayışı, bölgelerini koruma ve eş bulma gibi özellikleri incelenmiştir. Bu davranışlar optimizasyon algoritmaları tarafından olarak simüle edilmiş ve bu algoritmalarda kullanılan parametreler, çözülmek istenen probleme göre ayarlanabilmektedir.

Çeşitli bilim dallarındaki problemlerin çözümünde ortaya çıkan karmaşık durum ve çoklu bilinmeyenler bugüne kadar yaklaşık metodlar ile çözülüyordu. Fakat son yıllardaki sezgisel algoritmaların çoklu bilinmeyen parametrelerin bulunmasındaki başarıları görüldü (Genetik algoritma gibi). Böylece sezgisel algoritmaların başarılı bir şekilde yaklaşık yöntemler (Newton Rapson gibi) yerine kullanılabileceği görüldü. Yeni sezgisel algoritmaların ortaya çıkması ile son 20 yılda da önerilen yöntemlerin sayısı hızlı bir şekilde artışa geçti.

Karınca kolonileri, kuş sürüleri, arı sürüleri gibi canlıların kolektif akıllı davranışlarını inceleyen araştırmacılar, son yıllarda bu canlıların sergilediği sürü davranışlarından ilham alıp bu davranışları çeşitli matematiksel denklemler ile ifade etmiş ve o canlının adını taşıyan optimizasyon algoritmalarını ortaya çıkarmıştır (Yuan, X. ve diğerleri 2014). Bu algoritmalara örnek olarak, Parçacık Sürü Optimizasyon Algoritması (PSOA), Ateşböceği Optimizasyon Algoritması (AOA) Meyve Sineği Optimizasyon Algoritması (MSOA) ve Yarasa Optimizasyon Algoritması (YOA) gösterilebilir. Giriş kısmında da belirtildiği gibi bu çalışmada sürü hareketlerinden ilham alınmış olan bu algoritmalar üzerinde çalışmalar yapılacaktır.

6.1 Meyve Sineği Optimizasyon Algoritması (MSOA)

Meyve sinekleri, Şekil 6.1'de gösterilen kırmızı gözlere sahip küçük sineklerdir. Genellikle olgunlaşmış yiyeceklere ilgi duyarlar. Osphresis organları

sayesinde koku alma kabiliyetleri çok yüksek olan bu canlılar yaklaşık 40 km uzaklıktaki bir besinin kokusunu rahatlıkla alabilir (Yuan ve diğ. 2014). Ayrıca içinde 760 birim göz barındıran bir bileşik göze sahiptirler. Bu bileşik göz sayesinde besinlerin yerini rahatça saptayabilmektedir. Meyve sineklerinin besin arayış süreci şu şekilde gerçekleşir: ilk önce osphresis organıyla besini koklar ve besine doğru uçar. Besin konumuna yaklaştıktan sonra, hassas görüş özelliği ile besini ve diğer meyve sineklerinin konumu saptar (Xing ve Gao 2014).

Algoritmanın işleyiş adımları aşağıdaki gibidir:

Şekil 6.1: Beslenmekte olan bir meyve sineği (Wikipedia contributors 2024).

Adım 1: Meyve sineklerinin başlangıç konumu rastgele olacak şekilde belirlenir.

Adım 2: Yiyecek aranması için meyve sineklerinin konumlarına rastgele değerler eklenir.

$$X_i = X_{axis} + rand \tag{6.1}$$

$$Y_i = Y_{axis} + rand \tag{6.2}$$

Buradaki i değeri o anki meyve sineğinin indis değeridir. “rand” ifadesi de sineklerin dağılımından esinlenilen rastgele bir vektördür. Belirlenen yarıçap değeri kadar meyve sineklerinin konum bilgilerinin üzerinde ilave veya azaltma işlemi yapılır.

Adım 3: Yiyeceğin konumu tam olarak bilinmediği için başlangıç için orijine olan mesafe "Dist" tahmin edilir, ardından koku konsantrasyonu değerlendirme değeri olan "S" hesaplanır. Hesaplanan bu değer mesafe değerinin tersine eşittir.

$$Dist_i = \sqrt{X_i^2 + Y_i^2} \tag{6.3}$$

$$S_i = \frac{1}{Dist_i} \tag{6.4}$$

Adım 4: Meyve sineklerinin bireysel konumlarının koku konsantrasyon değerini "Koku" bulabilmek için her meyve sineğinin koku konsantrasyon değerlendirme değerleri "S" uygunluk fonksiyonunda yerine yazılır.

$$Koku_i = Fonksiyon(S_i) \tag{6.5}$$

Adım 5: Meyve sinekleri arasında en yoğun koku konsantrasyonuna sahip olanı bulunur.

$$[bestKoku, bestIndex] = \max(Koku) \tag{6.6}$$

Adım 6: En iyi koku konsantrasyon değeri ve koordinat değerleri x, y saklanır. Bu andan itibaren bütün meyve sinekleri bu konumdan harekete geçecektir.

$$Kokubest = bestKoku \tag{6.7}$$

$$X_{axis} = X(bestIndex) \tag{6.8}$$

$$Y_{axis} = Y(bestIndex) \tag{6.9}$$

Adım 7: Adım 2 – Adım 5 arası döngü devam eder. Sonraki iterasyonlarda eğer daha iyi değerler hesaplandıysa **Adım 6**'ya gidilir ve en iyi değerler güncellenir. Bu döngü belirli bir iterasyon sayısı kadar veya başka bir kriter sağlanana kadar devam eder.

6.2 Parçacık Sürü Optimizasyon Algoritması (PSOA)

Parçacık Sürü Optimizasyon Algoritması (PSOA), Eberhart ve Kennedy tarafından 1995 yılında önerilen sürüye bağlı değişken bir algoritmadır. PSOA, böcekler, kuşlar, balıklar ve diğer hayvan sürülerini de kapsayacak şekilde hayvanların sosyal davranışlarını taklit eder (Marini, F. & Walczak, B. 2015). Bu sürüler göç etmek veya besin aramak için Şekil 6.2'deki örnek görseldeki gibi birlikte hareket ederler, böylece sürüdeki her üye, diğer üyeleri ve kendisinin deneyimlerine göre arama modelini güncellemeye devam eder (Wang ve diğ. 2017).

Algoritmanın işleyiş adımları aşağıdaki gibidir:

Şekil 6.2: PSOA'nın esinlenildiği canlı türlerinden biri (Seferovic 2023).

Adım 1: Sürüdeki her parçacığın konumu rastgele belirlenir. Ardından bütün parçacıkların kişisel en iyi konumu "p_i" değerleri başlangıçta belirlenen konumlara eşitlenir.

$$p_i(0) = x_i(0) \tag{6.10}$$

Adım 2: Her parçacık için uygunluk fonksiyonu hesaplanır. Her parçacığın uygunluk fonksiyonu değerleri birbirleri ile karşılaştırılır. Hangi değer diğerlerine göre daha iyiyse o değer global değer "g" olur.

$$f\left(x_j(0)\right) \geq f\left(x_i(0)\right) \text{ ise } g = x_j(0) \tag{6.11}$$

Adım 3: Her parçacık için hız denklemi hesaplanır. Bu hesaplamada "c_1" ve "c_2" parametreleri dışarıdan belirlenir, "R_1" ve "R_2" parametreleri ise [0,1] arasında rastgele değerlere sahip tamsayılardır. "$v_i(t)$" ifadesi parçacıkların ilk hızı olup ilk iterasyonda sıfırdır. "ω" katsayısı ise probleme göre değişkenlik gösteren [0,1] arası değişebilen bir tamsayıdır.

Gerçek değerli ve genellikle $0 \leq (c_1, c_2) \leq 4$ aralığında olan "c_1" ve "c_2" ivme katsayıları "bilişsel katsayı" veya "sosyal katsayı" olarak isimlendirilir ve parçacığın kişisel ve global en iyi parçacıktan etkilenme miktarını belirler. "R_1" ve "R_2" değerleri, [0,1] arasında düzgün bir dağılımdan üretilmiş olan rastgele sayılardır. Hem bu iki değişken hem de sosyal ve bilişsel değişkenler (6.12)'deki hız denklemi üzerinde stokastik bir etkiye sahiptir (Marini ve Walczak 2015).

$$v_i(t+1) = \omega * v_i(t) + c_1 * \big(p_i - x_i(t)\big) * R_1 + c_2(g - x_i(t)) * R_2 \quad (6.12)$$

Adım 4: Hız denklemi hesaplanan parçacıkların konum verileri güncellenir.

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (6.13)$$

Adım 5: Güncellenen konumların uygunluk fonksiyonları yeniden hesaplanır, "p_i" ve "g" değerleri güncellenir.

Adım 6: Adım 3 – Adım 5 arası döngü devam eder. Bu döngü belirli bir iterasyon sayısı kadar veya başka bir kriter sağlanana kadar devam eder. Problemin çözümü "g" değeridir.

6.3 Ateş Böceği Optimizasyon Algoritması (AOA)

Ateşböceği Optimizasyon Algoritması, 2008 yılında Yang tarafından geliştirilen, sürü zekâsı yöntemini kullanan, doğadan esinlenilmiş, sezgisel ve değişken (stokastik) bir algoritmadır. Zor olarak kabul edilebilecek optimizasyon problemlerinde kullanılabilecek verimliliği yüksek bir algoritmadır.

Ateş böcekleri, Şekil 6.3'te görüldüğü gibi biyolüminesans adı verilen biyokimyasal bir süreç sayesinde etrafa yanıp sönen ışık yayabilirler. Bu türde bir yanıp sönme durumunda olan bir ışık kaynağı sayesinde karşı tarafa kur yapmak

istediğini belirtebilmesinin yanında potansiyel yırtıcıları da uyarabilir. Dişi ateşböcekleri eşlerini seçerken yanıp sönen ışığın parlaklığına önem verir. Yani hangi erkek ateşböceği daha parlak ışık saçıyor ise dişi o ateşböceğine doğru yönelecektir. Algoritmada da bu olaylardan esinlenilmiştir. Burada ateşböcekleri mevcut sistem için birer çözümü temsil etmektedir (Iztok ve diğ. 2013). Literatürde, bu algoritma kullanılırken 3 kurala dikkat edilir:

- Tüm ateşböcekleri cinsiyetsizdir. Böylece her ateşböceği birbiri ile etkileşim kurabilecektir.
- Çekicilik faktörü ışık şiddeti ile doğru orantılı olduğu için daha parlak ışık saçan ateşböceği diğerini çekecektir. Eğer tüm ateş böcekleri aynı ışık şiddetine sahip ise bütün ateş böcekleri rastgele hareket edecektir.
- Her ateş böceğinin parlaklığı hedef fonksiyonu ile doğru orantılıdır.

Algoritmanın işleyiş adımları aşağıdaki gibidir:

Şekil 6.3: Ateş böceği (Girard 2018).

Adım 1: Popülasyon rastgele değerler alacak şekilde belirlenir. Ardından popülasyondaki her ateş böceğinin uygunluk fonksiyonu değeri hesaplanır.

Adım 2: Her defasında bir çift ateş böceği eşleşebileceğinden eşleşen çiftin arasındaki mesafe (4.14)'teki gibi hesaplanır. Eşleşme sayısı popülasyon büyüklüğü yardımı ile hesaplanır. Buradaki $x_{i,k}$ ifadesi i. ateş böceğinin k. bileşeni iken $x_{j,k}$ ise j.

ateşböceğinin k. bileşenini göstermektedir. x_i, x_j i. ve j. ateş böceğini ifade etmektedir.

$$eşleşme = popülasyon(popülasyon - 1)/2 \quad (6.13)$$

$$r_{ij} = \left|\left|x_i - x_j\right|\right| = \sqrt{\sum_{k=1}^{n}(x_{i,k} - x_{j,k})^2} \quad (6.14)$$

Adım 3: Ardından her ateş böceği için çekicilik faktörü hesaplaması yapılır. Bu hesaplamada kullanılan "γ" ve "β_0" değeri [0,1] arasında değişebilen tamsayılar olup her probleme göre farklı değer alabilmektedir.

$$\beta(r) = \beta_0 . e^{-\gamma r_{i,j}^2} \quad (6.15)$$

Adım 4: Karşılaşma esnasında hangi ateşböceği diğerinden daha parlak ışık saçıyor (uygunluk fonksiyonunda daha iyi sonuç elde edilmiş) ise daha az parlak olan ateş böceği ona doğru çekilecektir. i. ateş böceğinin j. ateş böceği tarafından çekildiğini düşünürsek i. ateş böceğinin yeni koordinatı (6.16)'daki gibi hesaplanır. β_0 ifadesi genellikle 1 olarak alınır. Buradaki "α" değeri rastgele olup, [0,1] arasında değişebilen bir tamsayıdır.

Rastgeleleştirme parametresi olan "α" parametresi [0,1] aralığında rastgele üretilen bir değer alır. "β_0" parametresi sıfıra eşit olduğunda ateş böcekleri rastgele hareket edecektir. "γ" parametresinin yakınsama hızı üzerinde önemli bir etkisi vardır. Değeri teorik olarak $\gamma \in [0,\infty)$ aralığındaki herhangi bir değer olabilir. Bu değer optimizasyon probleminde göre değişiklik gösterir. Tipik olarak 0.1 ile 10 arasında değişir. "∈i" değeri Gauss dağılımından çekilen rastgele bir sayıdır (Fister ve diğ. 2013). "rand-1/2" şeklinde de ifade edilebilir. "rand" değeri [0,1] arasında düzgün bir aralıkta dağıtılmış rastgele bir sayı üretecidir (Arora ve Singh 2013).

$$x_i = x_i + \beta_0 e^{-\gamma r_{i,j}{}^2} . \left(x_i - x_j\right) + \alpha(rand - \tfrac{1}{2}) \quad (6.16)$$

Adım 5: Konumları güncellenen ateşböceklerinin uygunluk fonksiyonları yeniden hesaplanır.

Adım 6: Belirli bir iterasyon sayısına veya herhangi bir kritere ulaşılana kadar **Adım 2- Adım 5** arasında döngü devam eder.

6.4 Yarasa Optimizasyon Algoritması (YOA)

Yarasa Optimizasyon Algoritması, 2010 yılında Yang tarafından geliştirilmiş, yarasaların yön bulmak ve avlanmak için geliştirdiği sistemden esinlenilmiş bir algoritmadır. Doğada 1000 farklı yarasa türü olduğu düşünülmektedir. Mega yarasa adı verilen türler haricinde bütün yarasalar ekolokasyon özelliğini kullanabilirken mikro yarasa türleri bu özelliği daha sık kullanabilmektedir.

Ekolokasyon özelliği bir tür sonar tipidir. Çok yüksek bir ses darbesi yayıp çevredeki nesnelerin yankısını dinleyebilirler. Şekil 6.4'te gösterilen mikro yarasa türü ve diğer mikro yarasa türleri bu özellik sayesinde karanlıkta bile engellerden kaçınabilir, avlarının yerini ve tünedikleri bölgenin yerini saptayabilmektedir. Yang bu algoritmayı şu üç kurala uyarak geliştirmiştir:

- Tüm yarasalar mesafe ölçümü için ekolokasyon özelliğini kullanır. Bütün yarasaların av veya besin ve arka planda var olan engeller arasındaki farkı bildikleri varsayılır.
- Yarasalar avlarını tespit edebilmek için rastgele bir şekilde "x_i" konumundan, "v_i" hızla, "f_{min}" frekansıyla, değişen "λ" dalga boyuyla ve "A_0" ses yüksekliğiyle uçarlar.
- Ses yüksekliği değeri farklı yönlerden dolayı değişebilse bile, bu değerin minimum sabit değere kadar değişebildiği varsayılır. Yani ses yüksekliği belirli bir minimum düzeyin altına düşemez. (Yang ve He 2013)

Şekil 6.4: Bir mikro yarasa türü (Green 2022).

Algoritmanın işleyiş adımları aşağıdaki gibidir:

Adım 1: Yarasa popülasyonu rastgele konumlardan başlatılır, ilk hız değeri olan v_i değeri bütün popülasyon için sıfıra eşitlenir.

Adım 2: Her yarasa için [1,2] sayıları arasında rastgele değerde ses şiddeti değeri "A_i", frekans değeri "f_{min}" ve "f_{max}", [0.5,1] sayıları arasında rastgele değerde sinyal gönderme oranı "r_i" belirlenir. Frekans sınırları olan "f_{min}" ve "f_{max}" değerleri [0,1] aralığında düzgün şekilde rastgele dağıtılmış olan parametrelerdir ve optimizasyon probleminin türüne göre değişebilir (Kılıç 2022).

Adım 3: Tüm yarasaların ses şiddetinin ortalaması olan "A_0" değeri hesaplanır.

Adım 4: Popülasyondaki her yarasa için uygunluk fonksiyonu hesaplanır. Ardından her bir yarasa için [0,1] arasında rastgele bir değer alan "d" değişkeni atanır (Kılıç 2022). Bu "d" değeri ile her yarasanın "r_i" değeri karşılaştırılır. Eğer d değeri daha küçük ise yarasa lokal araştırma yaparak avına daha yaklaşacaktır, aksi halde yarasa rastgele araştırma yaparak avını tespit etmeye çalışacaktır.

"β" parametresi [0,1] aralığında düzgün dağılımdan elden edilen rastgele bir değere sahiptir. "α"ve "γ" parametreleri genellikle [0,1] arasında değerler alabilen, optimizasyon probleminin türüne göre değişebilen değerlere sahiptir (Yang ve He 2013).

- **Rastgele Uçuş:**

$$f_i = f_{min} + (f_{max} - f_{min}).\beta \tag{6.17}$$

$$v_i^t = v_i^{t-1} + \left(x_i^{t-1} - x_g\right).f_i^t \tag{6.18}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{6.19}$$

Buradaki "x_g" değeri yarasaların arasında en iyi sonucu veren yarasanın konumudur. "t" ifadesi o anki değeri, "t-1" ifadesi ise yarasanın bir önceki

iterasyondaki değerini temsil etmektedir. (6.18)'deki ifade, her yarasanın kendine ait hız denklemidir. β değeri [0,1] arasında rastgele değer alan bir tamsayıdır.

- **Lokal Araştırma Uçuşu:**

$$x_i^t = x_i^{t-1} + \varepsilon_k A_{ort}^{t-1} \tag{6.20}$$

Buradaki "ε_k" değeri [-1,1] arasında düzgün bir aralıkta dağıtılmış rastgele bir değerdir (Kılıç 2022).

Adım 5: Tüm popülasyonun sinyal gönderme oranları, ses şiddeti değeri ve ses şiddetinin ortalama değeri güncellenir.

$$A_i^{t+1} = \alpha A_i^t \tag{6.21}$$

$$r_i^{t+1} = r_{max}^t . (1 - 0.5. \gamma^{iter\ no}) \tag{6.22}$$

Yukarıdaki denklemlerde "α" ve "γ" değeri sistemin hızlı yakınsamaması için [0.9,1] arasında rastgele tamsayı olacak şekilde seçilir. Böylece "γ^{iterno}" değeri gittikçe sıfıra yaklaşacaktır. "iter no " o anda kaçıncı iterasyonda olduğunu belirtir. "r_{max}" ifadesi ise genellikle 1 olarak seçilmektedir.

Adım 6: Tüm popülasyon için uygunluk fonksiyonu yeniden hesaplanır.

Adım 7: Belirli bir iterasyon sayısına veya herhangi bir kritere ulaşılana kadar **Adım 4-Adım 6** arasında döngü devam eder.

7. YÖNTEM

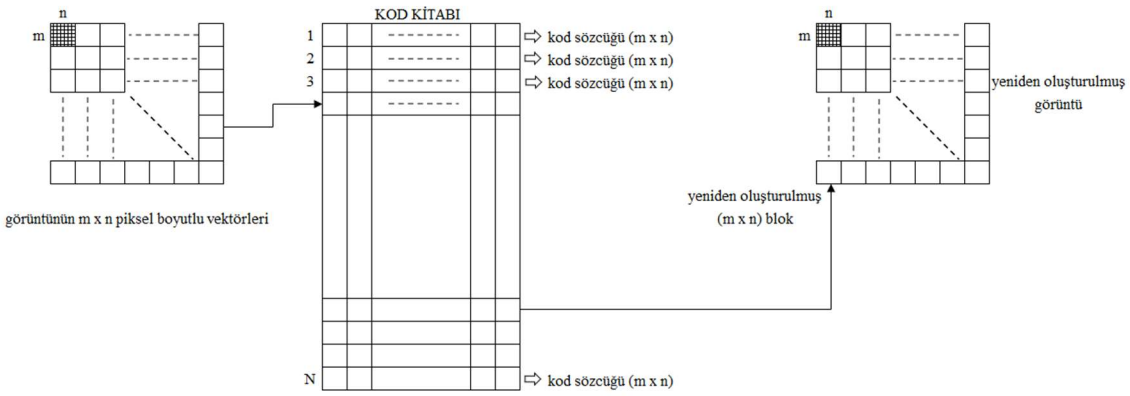
Bu çalışmada yukarıdaki kısımlarda anlatıldığı gibi K-Means algoritmasının işleyiş biçimi ile görüntü sıkıştırma işlemi için kod kitapları oluşturulmuştur. Bu kod kitapları görüntünün içerisinden rastgele konumlardaki 4x4 blokların alt alta yazılması ile oluşturulmaktadır. Kullanılan görüntüler 256x256 boyuttadır. Kod kitapları, Şekil 7.1'de gösterildiği üzere şu şekilde oluşturulmaktadır:

1. 256x256 boyutunda olan orijinal görüntü 4x16384 boyutunda vektöre dönüştürülür. Bu dönüşümde piksellerin yerleri sıralıdır, yani rastgele bir dönüşüm gerçekleşmemektedir.

2. Elde edilen 4x16384 boyutundaki vektörden rastgele olacak şekilde 4x4'lük parçalar alınır, bu parçalara kod sözcüğü adı verilir.

3. Alınan 8 kod sözcüğü alt alta veya yan yana yazılarak 4x32 boyutunda bir vektör elde edilir. Bu vektöre kod kitabı adı verilir.

Bu çalışmada sezgisel algoritmalarda çözüm kümesi olarak kod kitapları kullanılmıştır. 8 kod sözcüğünden oluşan bir çözüm kümesi tercih edilmesinin sebebi, farklı boyuttaki diğer kod kitabı tasarımlarına kıyasla daha verimli çalışmasıdır. Sezgisel algoritmalarla çalışırken popülasyonlardaki her bir birey bir kod kitabını temsil etmektedir.



Şekil 7.1: Kod kitabının oluşturulması (ikinci kez).

Her bir kod kitabı ile orijinal görüntü yeniden oluşturulur. Ardından orijinal görüntü ile yeniden oluşturulan görüntünün karşılaştırılması yapılır. Bunun için Mean Squared Error (MSE) yani "Ortalama Karesel Hata" değeri hesaplanır. MSE değeri, yeniden oluşturulan görüntünün orijinal görüntüden ne kadar farklı olduğunun sayısal olarak ifade edilmesini sağlar. Çözüm kümelerinin sonuçları karşılaştırıldığında en düşük MSE değerine sahip olan çözüm kümesi en iyi çözüm kümesi olarak kabul edilmektedir. Bu çalışmada da çeşitli sezgisel algoritmalar sonucu oluşturulan görüntülerin MSE değerleri incelenerek hangi sezgisel algoritmanın daha iyi sonuç verdiği gözlemlenecektir.

$$MSE = \left|\frac{1}{n}\sum_{i=1}^{n}(Y - Y')^2\right| \tag{7.1}$$

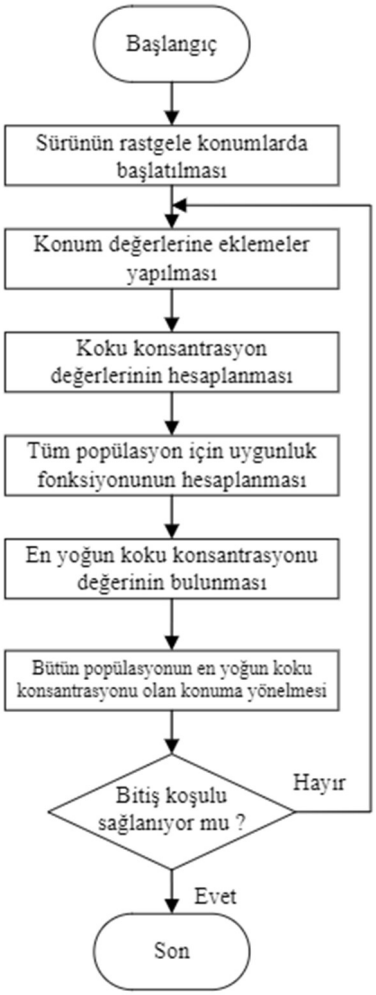
Buradaki "n" değeri veri sayısıdır. Görüntü üzerinde çalışma yapıldığı ve 256x256 boyutta görüntüler kullanıldığı için bu değer 65536'dır. Y değeri orijinal görüntüdeki piksel değeri, Y' değeri, yeniden oluşturulan görüntüdeki piksel değeridir.

Tepe Sinyal-Gürültü Oranı (PSNR), sinyal gürültüsünün gücü ile sinyalin maksimum gücü arasındaki oran olarak ölçülen niceliktir. PSNR, genellikle kayıplı sıkıştırma tekniği uygulanmış görüntülerin yeniden yapılandırma kalitesinin desibel (dB) değerini ölçmek için kullanılır.

Gri seviye (8 bit) görüntüler için m×n (256x256) boyutunda olan bir referans görüntü Y' ve bir test görüntü Y verildiğinde, Y' ve Y arasındaki PSNR şu şekilde tanımlanır:

$$PSNR = 10log_{10}(\frac{255^2}{MSE}) \tag{7.2}$$

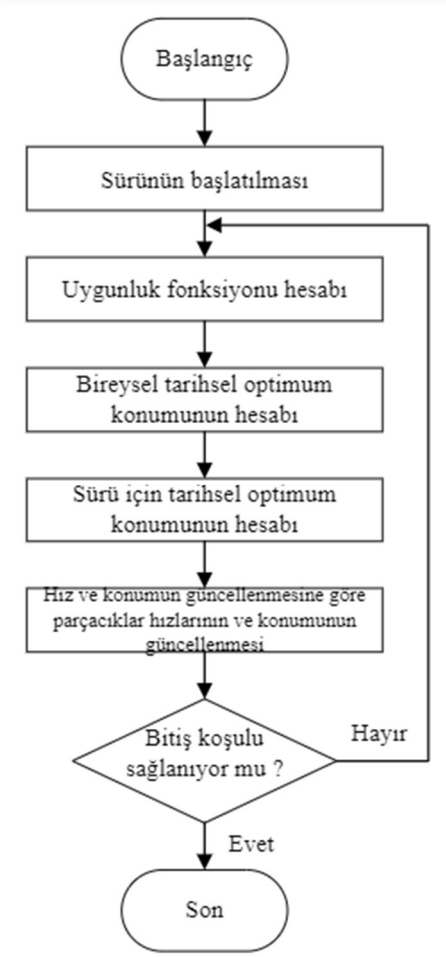
Buradaki 255 değeri gri tonlamadaki en yüksek değeri ifade etmektedir.



Şekil 7.2: Meyve Sineği Optimizasyon Algoritması işleyiş şeması.

Tablo 7.1: Meyve Sineği Optimizasyon Algoritması sözde kodu.

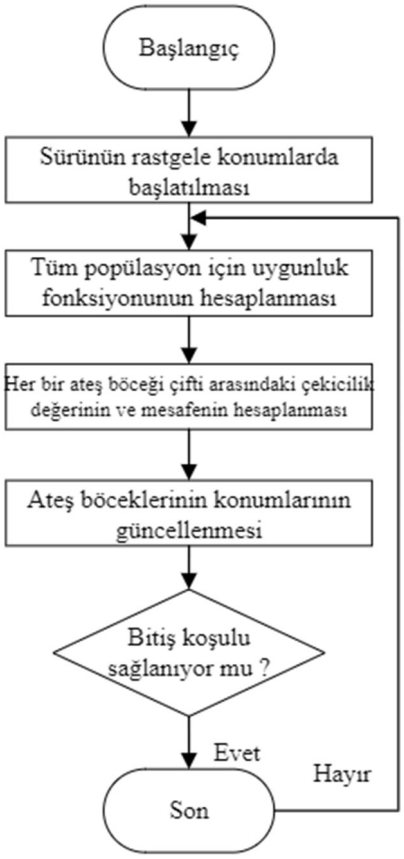
Meyve Sineği Optimizasyon Algoritması
Rastgele popülasyon üretimi
while (t<Maximum iterasyon sayısı)
Tüm popülasyonun uygunluk fonksiyonunu hesapla
En iyi değeri veren sineğin fonksiyon değeri ve konum bilgilerini (elit) sakla
Bütün popülasyonun konumunu elit sineğin konumuna eşitle.
Tüm popülasyonun konum bilgilerine rastgele değerler ekle veya çıkar



Şekil 7.3: Parçacık Sürü Optimizasyon Algoritması işleyiş şeması.

Tablo 7.2: Parçacık Sürü Optimizasyon Algoritması sözde kodu.

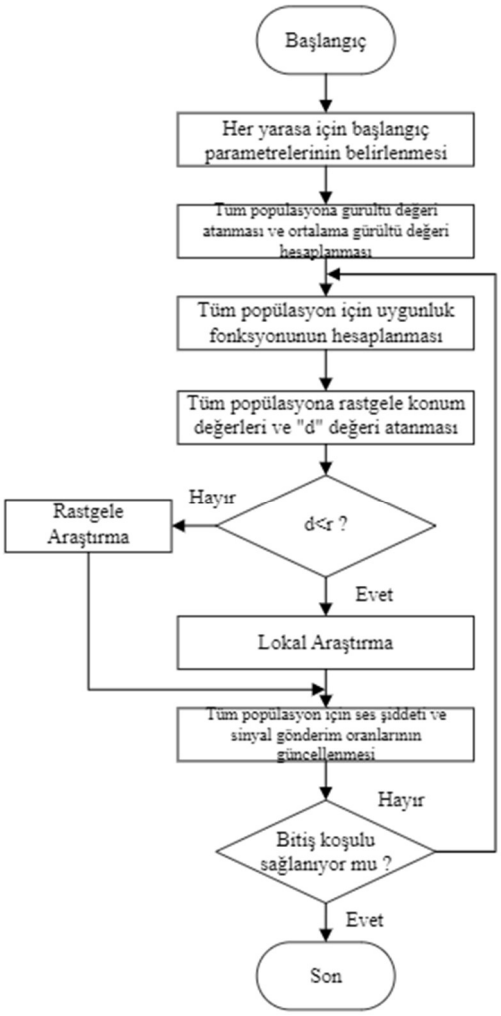
Parçacık Sürü Optimizasyon Algoritması
Rastgele popülasyon üretimi
Tüm popülasyonun konum bilgilerini p_i içine kaydet
while (t<Maximum iterasyon sayısı)
Tüm popülasyon için uygunluk fonksiyonunu hesapla
En iyi uygunluk fonksiyonu değerine sahip bireyin konum bilgilerini g içine kaydet
p_i içindeki konum bilgilerini tüm popülasyon için güncelle
Popülasyondaki her birey için hız denklemi hesapla
Hesaplanan hız denklemlerini popülasyona ekle



Şekil 7.4: Ateş Böceği Optimizasyon Algoritması işleyiş şeması.

Tablo 7.3: Ateş Böceği Optimizasyon Algoritması sözde kodu.

Ateş Böceği Optimizasyon Algoritması
Rastgele popülasyon üretimi **while** (t<Maximum iterasyon sayısı) Tüm popülasyon için uygunluk fonksiyonunu (parlaklık değeri) hesapla **while** (k<popülasyon*(popülasyon-1)/2) Popülasyon içerisinden rastgele iki birey seç **(6.14)**'teki denklemi kullanarak iki birey arasındaki mesafeyi hesapla **(6.15)**'teki denklemi kullanarak çekicilik faktörünü hesapla Uygunluk fonksiyonu daha kötü olan bireyi diğer bireye yaklaştırmak için **(6.16)**'teki denklemi kullan



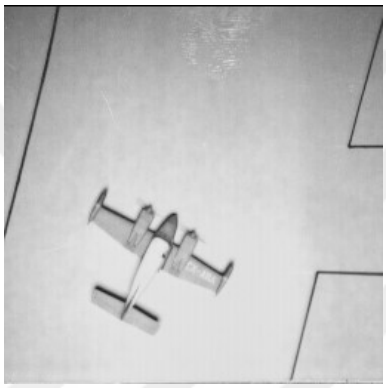
Şekil 7.5: Yarasa Optimizasyon Algoritması işleyiş şeması.

Tablo 7.4: Yarasa Optimizasyon Algoritması sözde kodu.

Yarasa Optimizasyon Algoritması
Rastgele popülasyon üretimi Tüm popülasyon için v_i=0 Tüm popülasyon için f_{min}, f_{max}, [1,2] arasında rastgele A_i ses şiddeti ve [0.5,1] arasında rastgele r_i sinyal gönderme oranı belirle Ortalama ses şiddeti değeri A_0 hesapla **while** (t<Maximum iterasyon sayısı) Tüm popülasyon için uygunluk fonksiyonunu hesapla [0,1] arasında rastgele d değeri ata ve o anki yarasanın sinyal gönderme oranı r ile karşılaştır **if** (d<r) **(6.17)**'deki denklemi hesapla, burada bulunan değer ile **(6.18)**'i kullanarak yarasa için hız değerini hesapla ve **(6.19)**'deki denklem ile yarasanın konumunu güncelle **else** [-1,1] arasında rastgele ε_k belirle ve **(6.20)**'deki denklemi kullanarak yarasanın konumunu güncelle **(6.21)** ve **(6.22)**'deki denklemleri kullanarak her yarasa için sinyal gönderme oranını, ses şiddetini ve ortalama ses şiddetini güncelle

8. BULGULAR

Bu çalışmada 4 farklı optimizasyon algoritması ile, popülasyon sayıları, hedef iterasyon sayıları eşit olacak şekilde farklı kontrastlarda, gri tonlamalı, 256x256 boyutta, BMP formatındaki 6 adet görüntüye sıkıştırma işlemi uygulanmıştır. Sonuçları sayısal olarak gözlemleyip karşılaştırma yapabilmek için yeniden oluşturulan her görüntü için MSE değeri hesaplanmıştır. Kullanılan örnek görüntüler aşağıdadır:



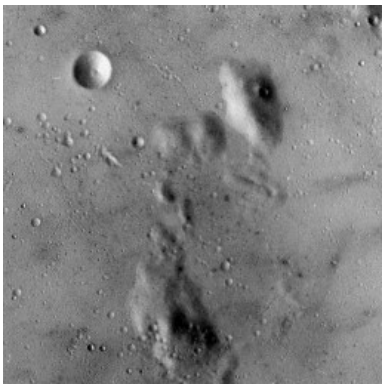
Şekil 8.1: Airplane görseli.

Şekil 8.2: Barbara görseli.

Şekil 8.3: Boat görseli.

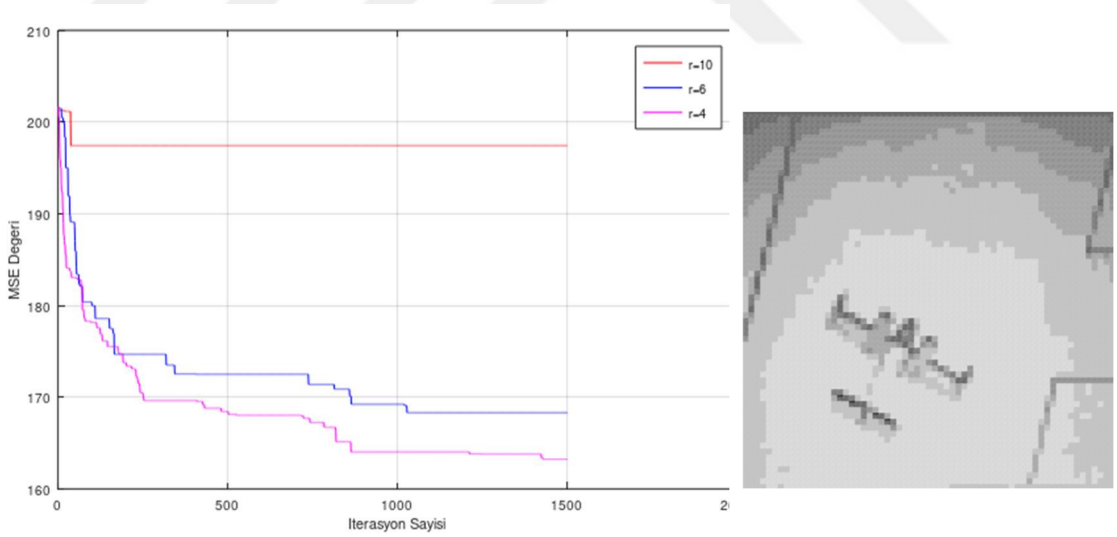
Şekil 8.4: Clock görseli.

Şekil 8.5: Lena görseli.

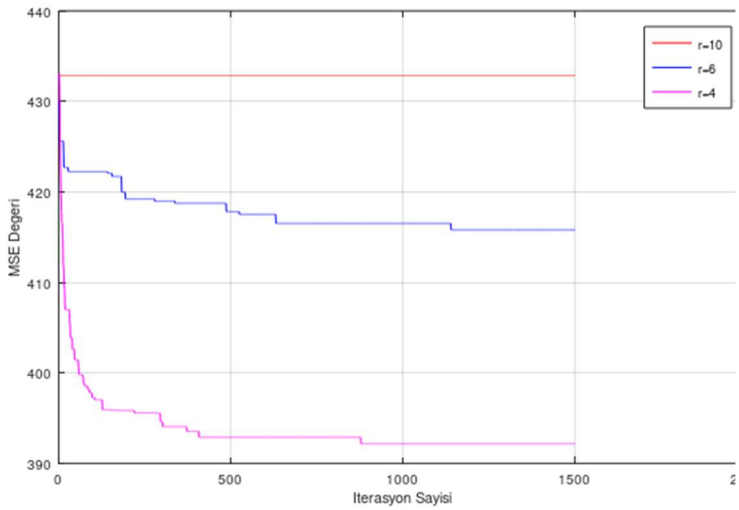


Şekil 8.6: Moon görseli.

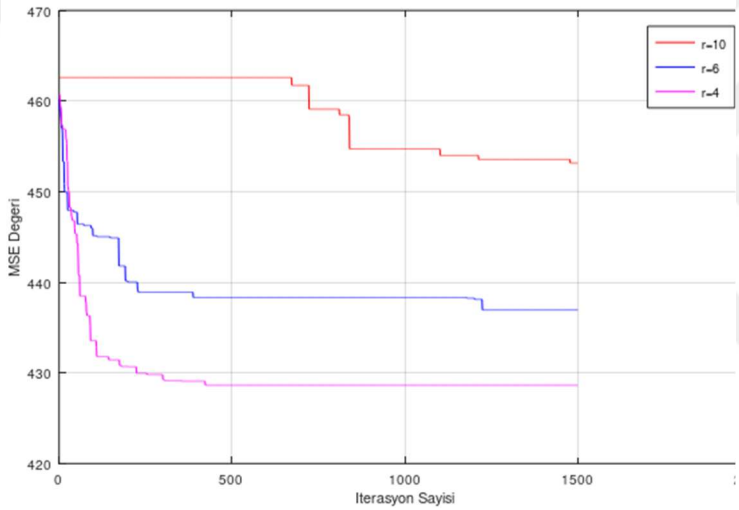
Meyve Sineği Optimizasyon Algoritmasını çalıştırırken meyve sinekleri için çeşitli arama yarıçapları denenmiştir. Bu yarıçap değerleri her bir sineğin konum bilgilerine ilave edilecek olan sayıyı göstermektedir. Yarıçap değeri küçüldükçe meyve sineklerinin hata değerinin azaldığı görülmüştür. Bu çalışmada inceleme için r=10, r=6 ve r=4 yarıçapları kullanılmıştır. En iyi değeri veren r=4 yarıçaplı çalıştırmanın sonucunda yeniden oluşturulan görüntüler ve yarıçapların performansının karşılaştırıldığı grafikler aşağıda verilmiştir:



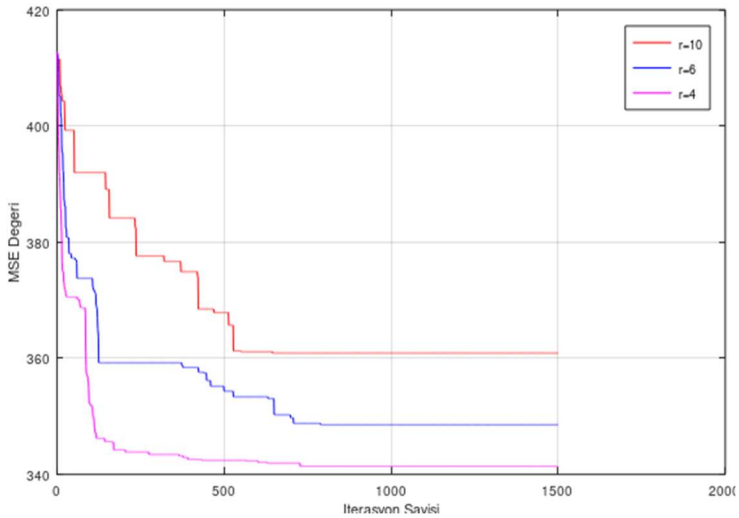
(a)

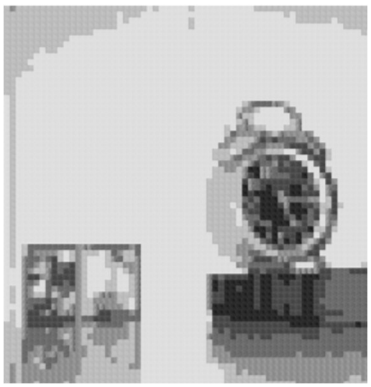

r=10
r=6
r=4
MSE Degeri
Iterasyon Sayisi

(b)

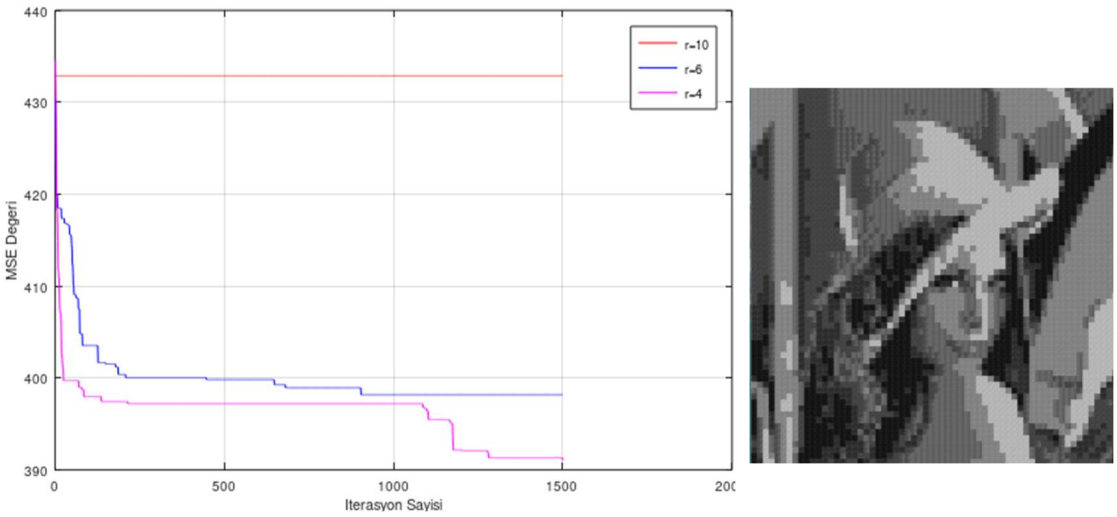

r=10
r=6
r=4
MSE Degeri
Iterasyon Sayisi

(c)

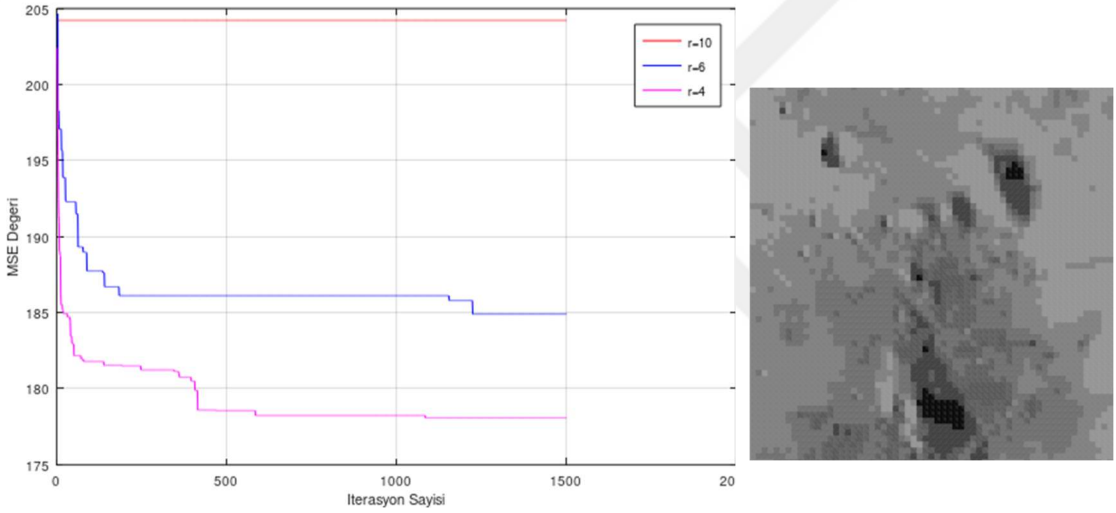

r=10
r=6
r=4
MSE Degeri
Iterasyon Sayisi



(d)



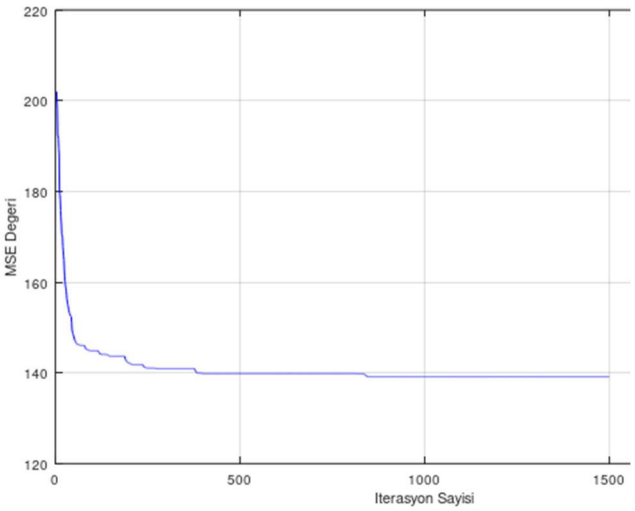
(e)



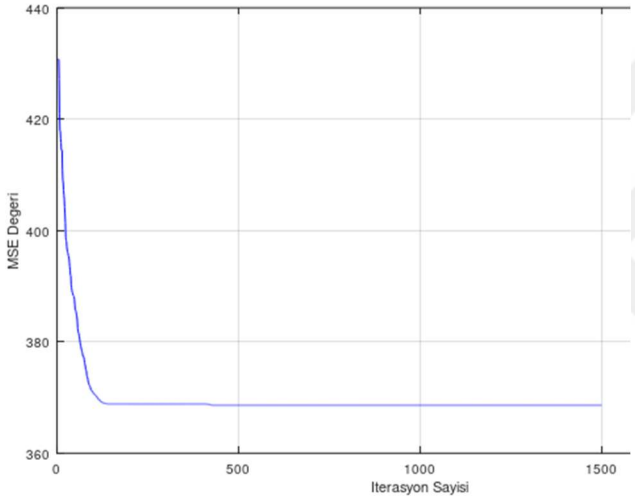
(f)

Şekil 8.7: MSOA'nın sonuçları a) Airplane, b) Barbara, c) Boat, d) Clock, e) Lena, f) Moon.

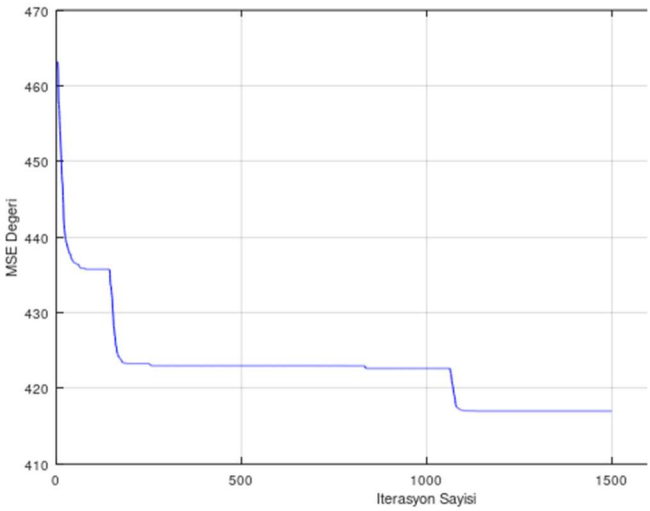
Parçacık Sürü Optimizasyon Algoritmasında c_1=1, c_2=2 ve ω=0.2 olarak alınmıştır. Denenen diğer katsayılar arasında en iyi sonucu bu değerler vermiştir. Algoritmanın uygulanması sonucu sayısal değerler ve yeniden oluşturulan görüntüler aşağıdadır:


MSE Degeri
Iterasyon Sayisi
220
200
180
160
140
120
0
500
1000
1500

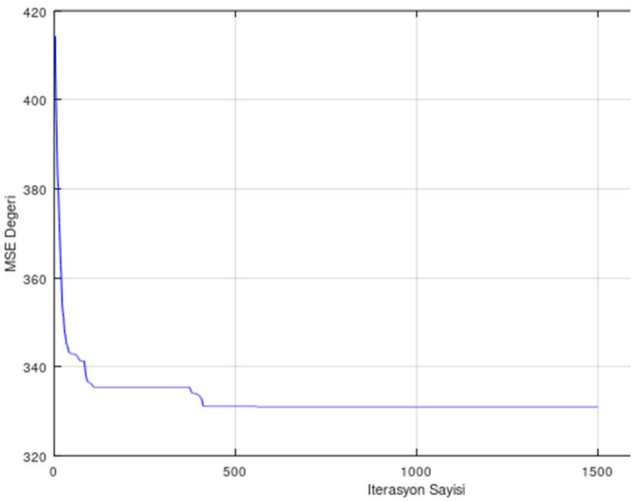
(a)

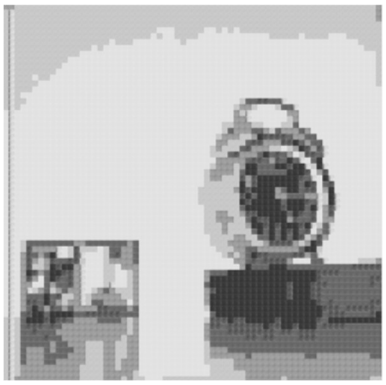

MSE Degeri
Iterasyon Sayisi
440
420
400
380
360
0
500
1000
1500

(b)

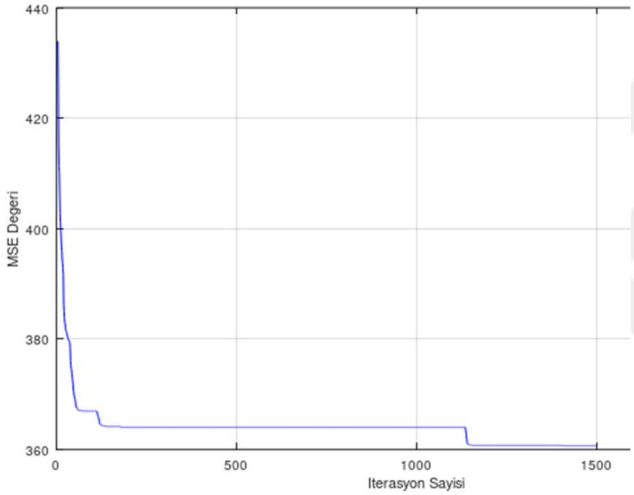

MSE Degeri
Iterasyon Sayisi
470
460
450
440
430
420
410
0
500
1000
1500

(c)

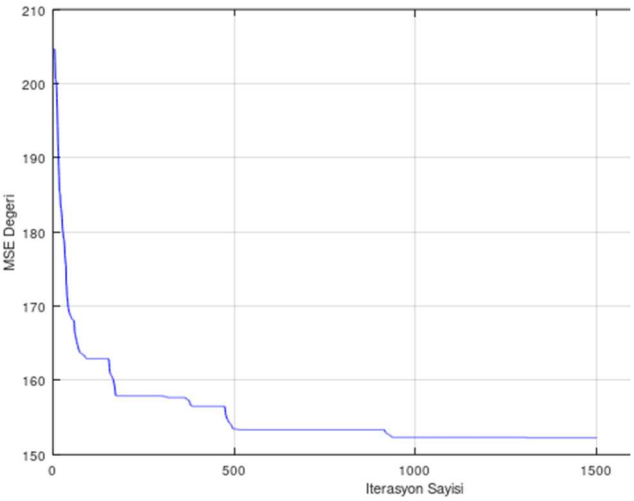




(d)



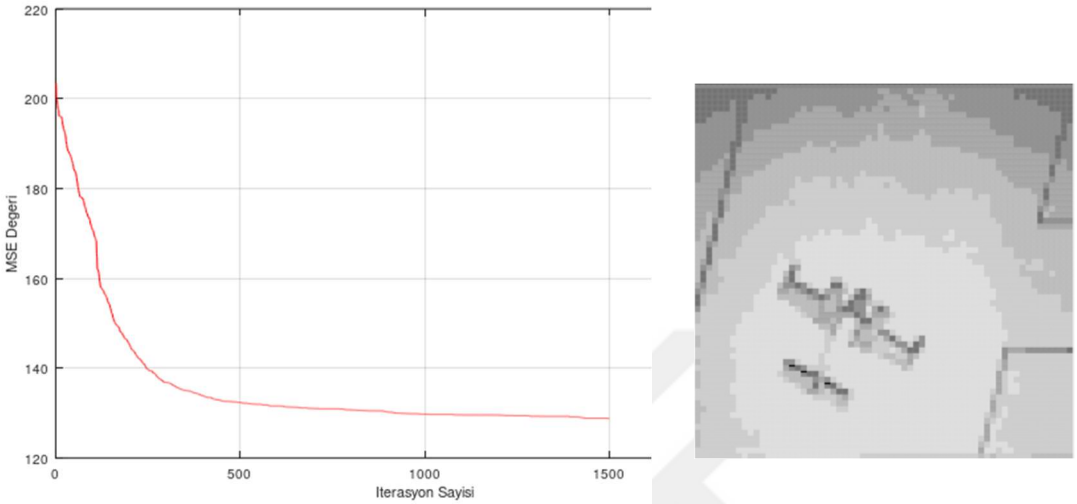
(e)



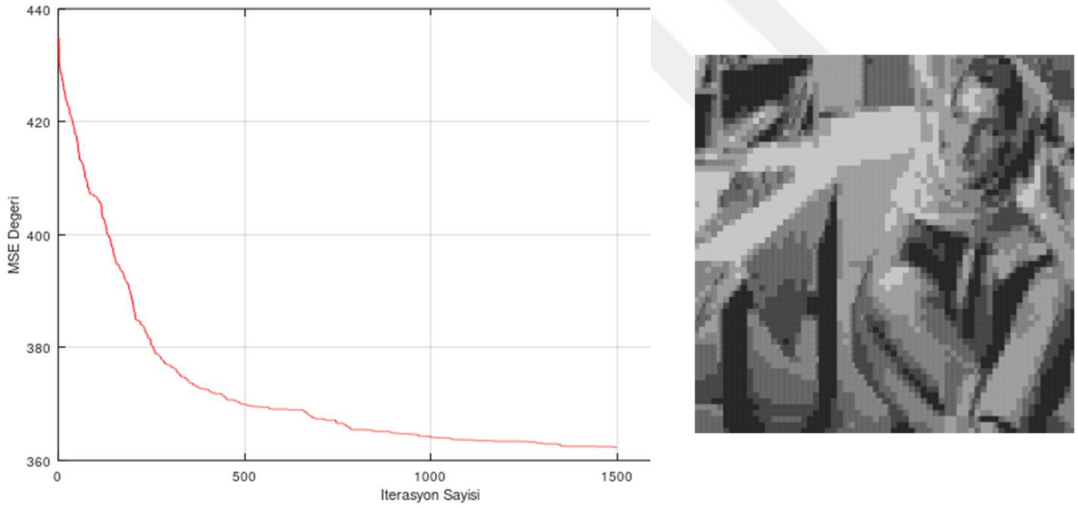
(f)

Şekil 8.8: PSOA'nın sonuçları a) Airplane, b) Barbara, c) Boat, d) Clock, e) Lena, f) Moon.

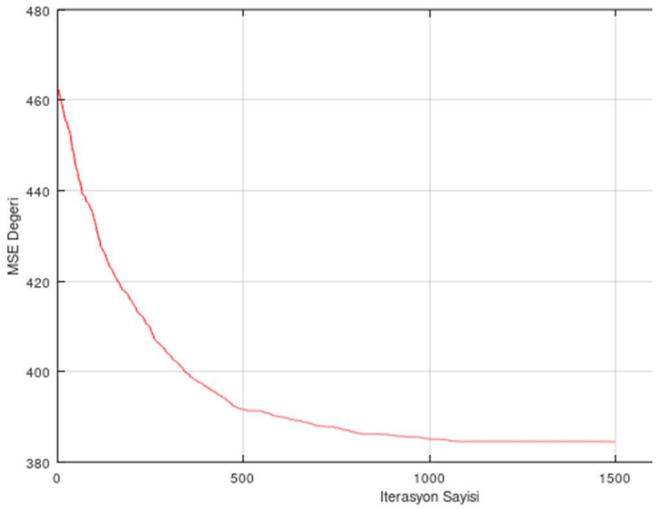
Ateş Böceği Optimizasyon Algoritmasında en verimli sonuçları elde edecek şekilde katsayı düzenlemesi yapılıp α=0.5, γ=0.01, β_0=0.9 olarak alınmıştır. Algoritmanın uygulanması sonucu sayısal değerler ve yeniden oluşturulan görüntüler aşağıdadır:



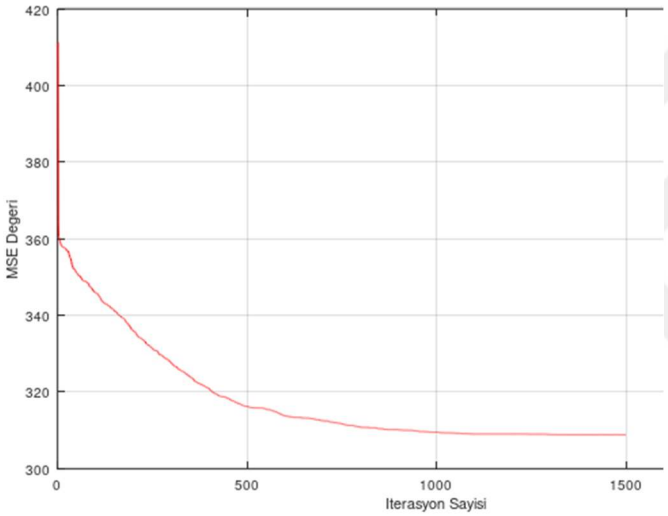
(a)

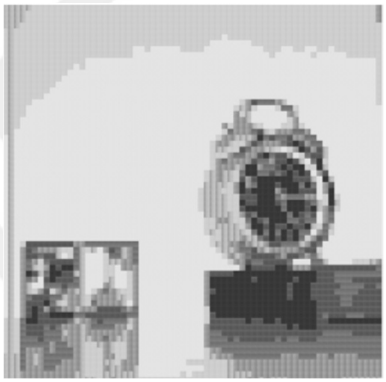


(b)

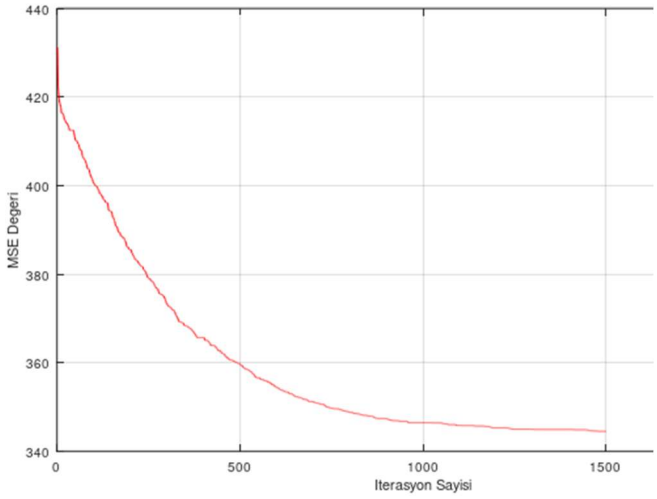

480
460
440
420
400
380
MSE Degeri
0
500
1000
1500
Iterasyon Sayisi

(c)

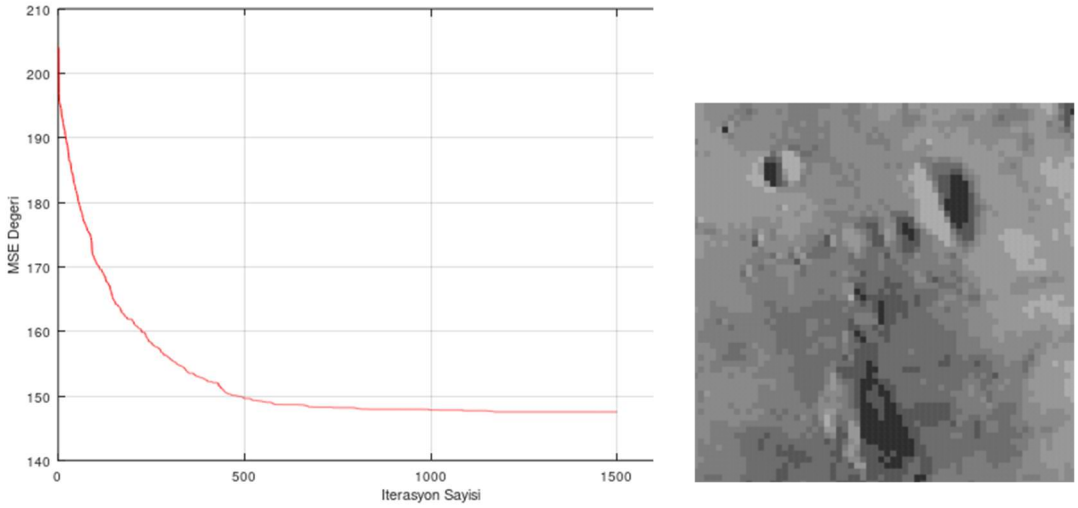

420
400
380
360
340
320
300
MSE Degeri
0
500
1000
1500
Iterasyon Sayisi



(d)


440
420
400
380
360
340
MSE Degeri
0
500
1000
1500
Iterasyon Sayisi

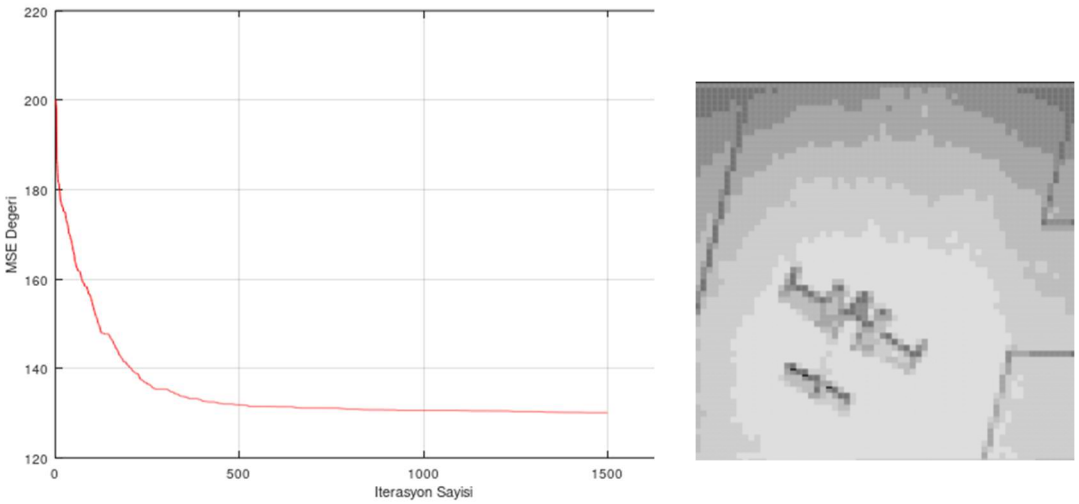
(e)



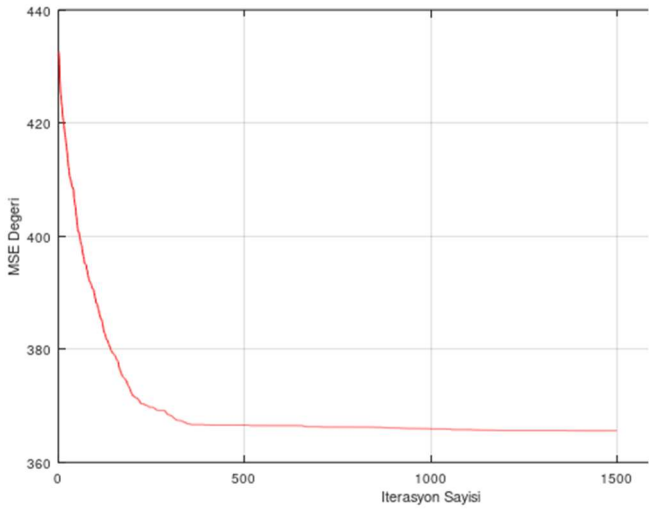
(f)

Şekil 8.9: AOA'nın sonuçları a) Airplane, b) Barbara, c) Boat, d) Clock, e) Lena, f) Moon.

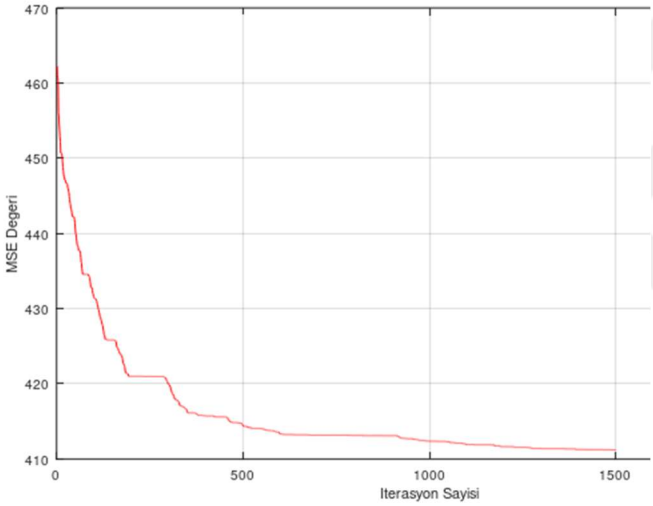
Yarasa Optimizasyon Algoritmasında en verimli sonuçlara ulaşabilmek için algoritmada bulunan f_{min}=0.0 ve f_{max}=0.3 olarak alınmıştır. γ değeri görüntüye göre değişiklik göstermekte olup 0.997 veya 0.998 değerinde seçilmiştir. Yüksek kontrastlı görüntülerde γ=0.998, düşük kontrastlı görüntülerde γ=0.997 olarak seçilmiştir. α değeri ise her görüntü için sabit olup α=0.998 olarak seçilmiştir.



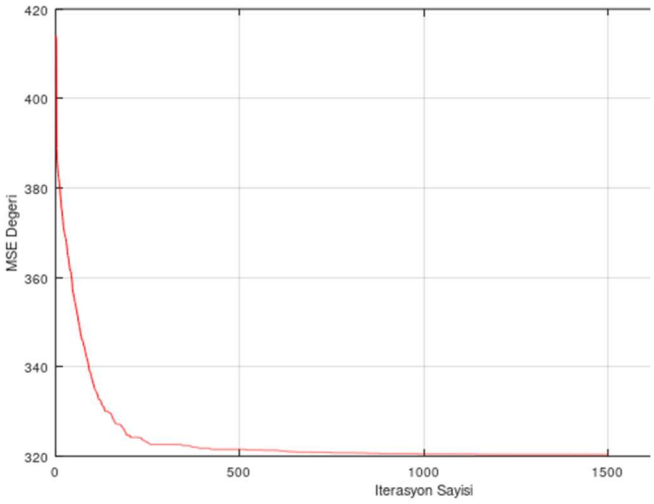
(a)

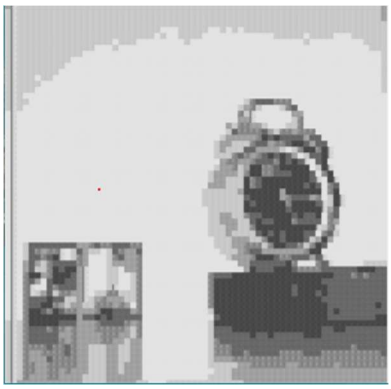

MSE Degeri
Iterasyon Sayisi
440
420
400
380
360
0
500
1000
1500

(b)

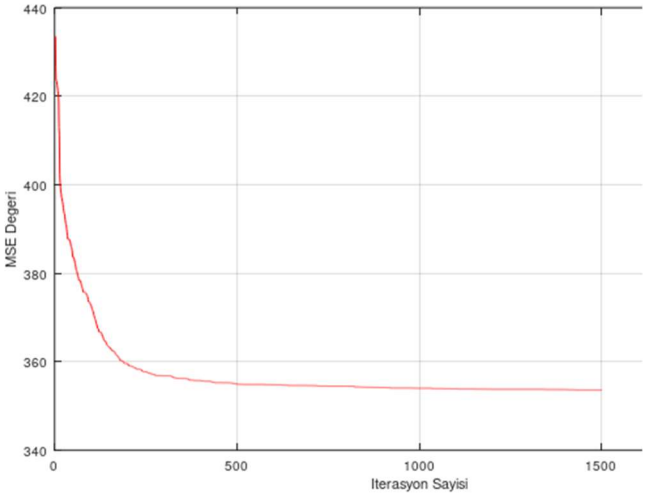

MSE Degeri
Iterasyon Sayisi
470
460
450
440
430
420
410
0
500
1000
1500

(c)

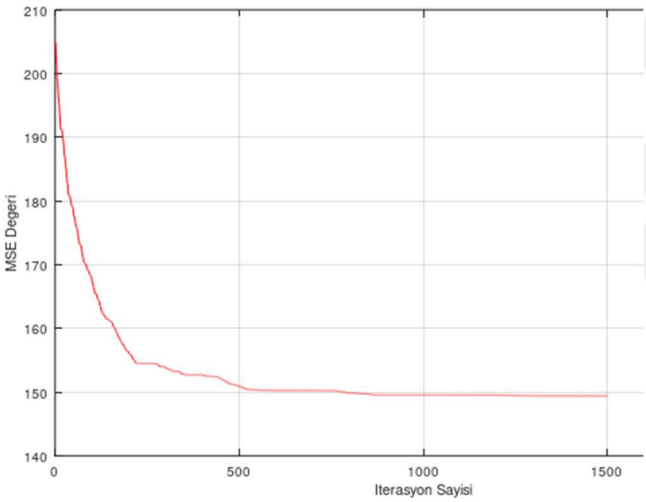

MSE Degeri
Iterasyon Sayisi
420
400
380
360
340
320
0
500
1000
1500



(d)



(e)

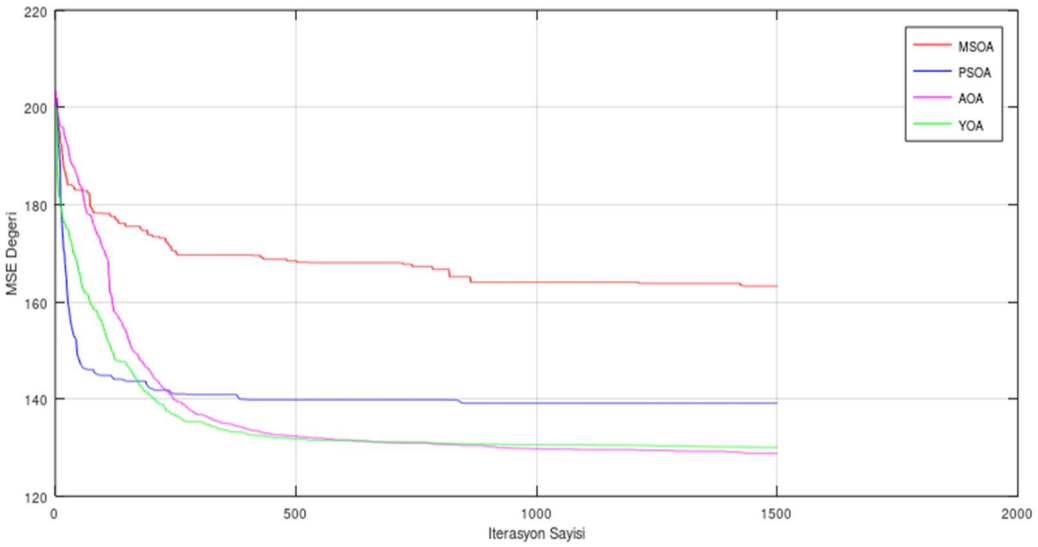


(f)

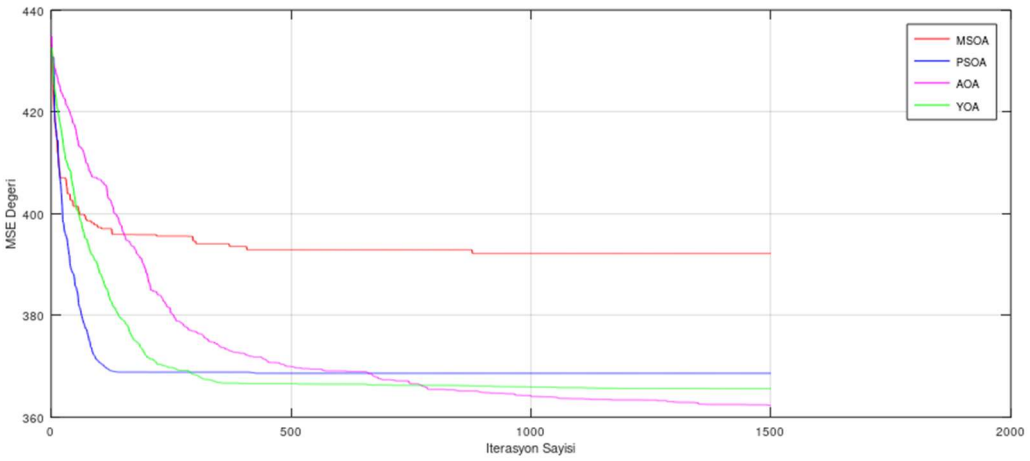
Şekil 8.10: YOA'nın sonuçları a) Airplane, b) Barbara, c) Boat, d) Clock, e) Lena, f) Moon.

9. SONUÇ VE ÖNERİLER

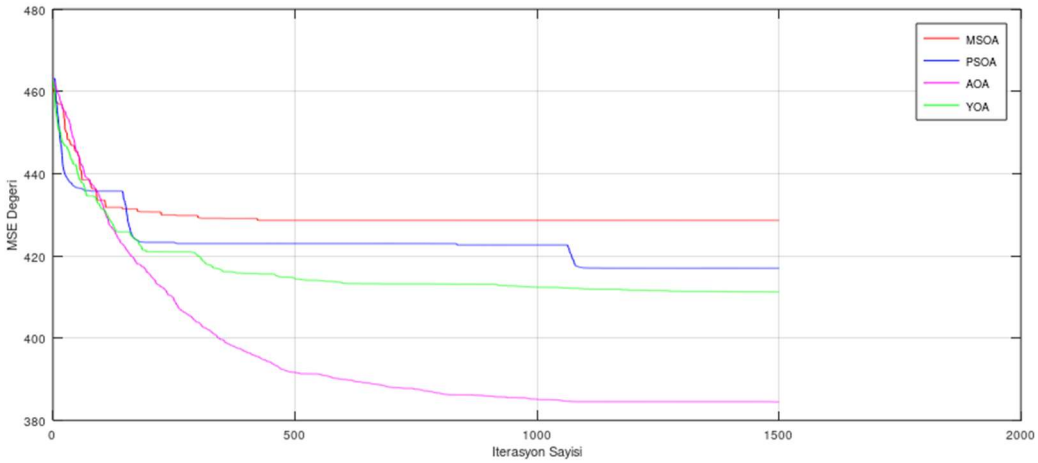
Bütün algoritmalar eşit şartlar altında çalıştırılıp incelendiğinde sonuçlar aşağıdaki grafiklerde ve tabloda görüldüğü gibidir:



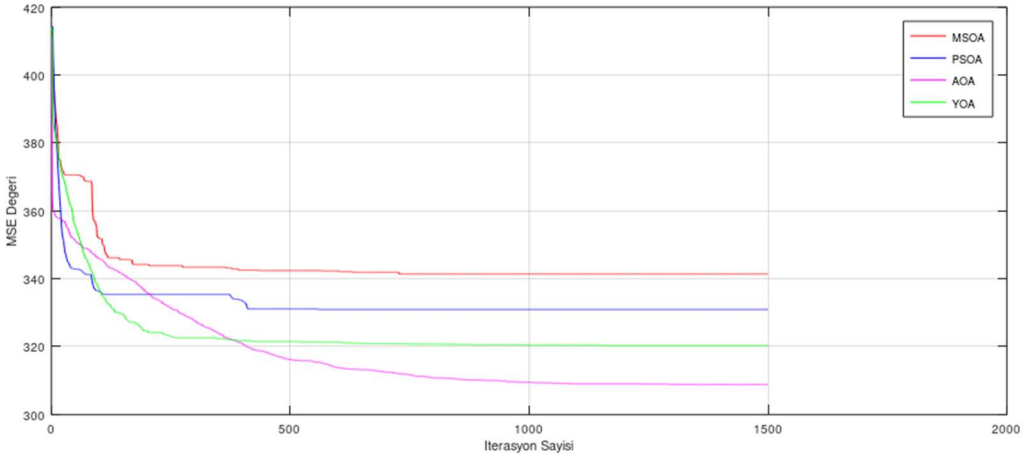
Şekil 9.1: Airplane için sonuçlar.



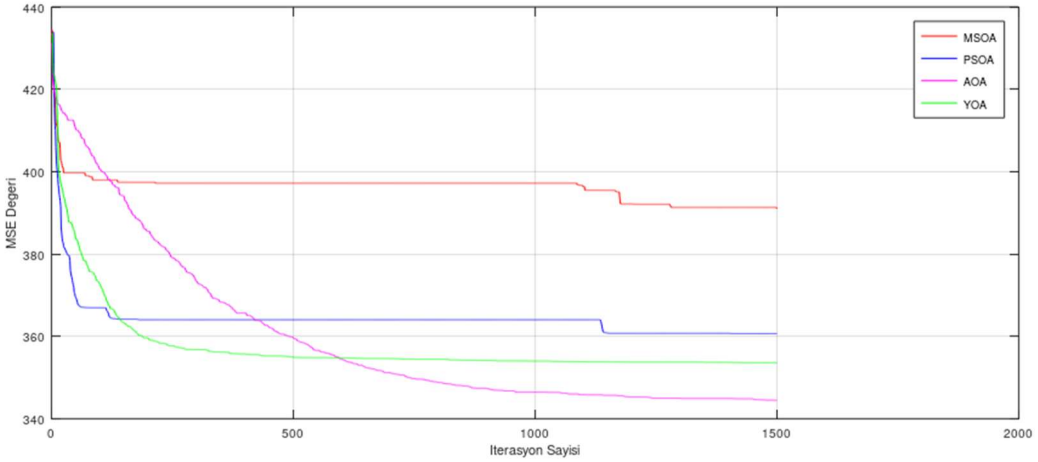
Şekil 9.2: Barbara için sonuçlar.



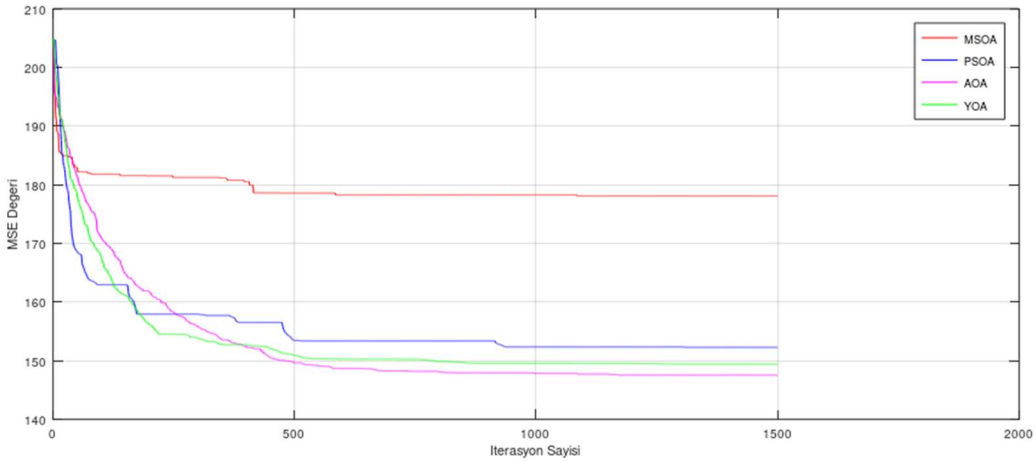
Şekil 9.3: Boat için sonuçlar.



Şekil 9.4: Clock için sonuçlar.



Şekil 9.5: Lena için sonuçlar.



Şekil 9.6: Moon için sonuçlar.

Tablo 9.1: Yeniden oluşturulan görüntülerin MSE değerleri.

MSE	Airplane	Barbara	Boat	Clock	Lena	Moon
MSOA	163.2	392.1	428.6	341.4	391.0	178.1
PSOA	139.1	368.6	417.0	331.0	360.7	152.3
AOA	128.8	362.3	384.5	308.9	344.5	147.5
YOA	130.1	365.6	411.2	320.4	353.6	149.4

Tablo 9.2: Yeniden oluşturulan görüntülerin PSNR değerleri.

PSNR	Airplane	Barbara	Boat	Clock	Lena	Moon
MSOA	26.0 dB	22.2 dB	21.8 dB	22.8 dB	22.2 dB	25.6 dB
PSOA	26.7 dB	22.5 dB	21.9 dB	22.9 dB	22.6 dB	26.3 dB
AOA	27.0 dB	22.5 dB	22.3 dB	23.2 dB	22.8 dB	26.4 dB
YOA	27.0 dB	22.5 dB	21.9 dB	23.0 dB	22.6 dB	26.4 dB

Yukarıdaki grafikler ve tablo incelendiğinde görüntü kontrastı fark etmeksizin AOA'nın en iyi sonucu verdiği gözlemlenmiştir. Bütün yeniden oluşturulan görüntüler 39:1 oranda sıkıştırılmıştır. Grafiklerden de anlaşılacağı üzere MSOA'nın performansı diğer algoritmalara göre düşük bir seviyede kalmıştır. Bunun gibi durumlarda algoritmaya bazı teknikler veya modeller eklenerek algoritmanın performansı iyileştirilebilir.

10. KAYNAKLAR

Arora, S., & Singh, S. “The firefly optimization algorithm: convergence analysis and parameter selection”, *International Journal of Computer Applications*, 69(3), (2013).

Daga, R. R. M., & Yusiong, J. P. T.,” Image compression using harmony search algorithm”, *International Journal of Computer Science Issues* (IJCSI), 9(5), 16, (2012).

Dai, L., Ding, J., & Yang, J., “Inhomogeneity-embedded active contour for natural image segmentation”, *Pattern Recognition*, 48(8), 2513-2529, (2015).

Fister, I., Fister Jr, I., Yang, X. S., & Brest, J., “A comprehensive review of firefly algalgorithms”, *Swarm and Evolutionary Computation*, 13, 34-46, (2013).

Girard, L., “How many evolutionary events can it take to screw in nature’s lightbulb?[online]”, (29 Nisan 2024), https://wi.mit.edu/news/how-many-evolutionary-events-can-it-take-screw-nature-s-lightbulb, (2018).

Green, S., “Microbats face extinction due to climate change, vegetation clearance[online]”, (29 Nisan 2024), https://www.abc.net.au/news/2022-06-22/micro-bats-face-extinction-climate-change/101170298, (2022).

Ismaili, I. A., Khowaja, S. A., & Soomro, W. J. (2013, July). “Image compression, comparison between discrete cosine transform and fast fourier transform and the problems associated with DCT.”, In International Conference on Image Processing, Computer Vision and Pattern Recognition (pp. 962-965), (2012).

Kaur, R., & Choudhary, P. “A review of image compression techniques”, *Int. J. Comput. Appl*, 142(1), 8-11, (2016).

Khaleel, S. I., “Image Compression Using Swarm Intelligence”, *International Journal of Intelligent Engineering & Systems*, 14(1), (2021).

Kilic, I. “A lévy flight based bat optimization algorithm for block-based image compression”, *Tehnički Glasnik*, 16(4), 477–483, (2022).

Kiliç, İ. “Digital image compresion techniques”, Yüksek Lisans Tezi, *Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü*, Elektrik Elektronik Mühendisliği Anabilim Dalı, İzmir, (1997).

Kumar, V., & Kumar, D. (2021). “A systematic review on firefly algorithm: past, present, and future”, *Archives of Computational Methods in Engineering*, 28, 3269-3291, (2021).

Li, Y., & Wu, H., “A clustering method based on K-means algorithm”, *Physics Procedia*, 25, 1104-1109, (2012).

Marini, F., & Walczak, B., “Particle swarm optimization (PSO). A tutorial”, *Chemometrics and Intelligent Laboratory Systems*, 149, 153-165, (2015).

Piech, C., “K Means[online]”, (29 Nisan 2024), https://stanford.edu/~cpiech/cs221/handouts/kmeans.html, (2013).

Rahebi, J., “Vector quantization using whale optimization algorithm for digital image compression”, *Multimedia Tools and Applications*, 81(14), 20077-20103, (2022).

Raid, A. M., Khedr, W. M., El-Dosuky, M. A., & Ahmed, W. (2014). “Jpeg image compression using discrete cosine transform-A survey.”, arXiv preprint arXiv:1405.6147, (2014).

Seferovic, M., “Collaborative leadership: lessons from bird flocks[online]”, (29 Nisan 2024), https://www.forbes.com/sites/forbesbusinesscouncil/2023/03/20/collaborative-leadership-lessons-from-bird-flocks/?sh=593b1ea530a9, (2023).

Vijayvargiya, G., Silakari, S., & Pandey, R., “A survey: various techniques of image compression”, arXiv preprint arXiv:1311.6877, (2013).

Wang, D., Tan, D., & Liu, L., “Particle swarm optimization algorithm: an overview”, *Soft Computing*, 22, 387-408, (2018).

Wikipedia contributors, “Drosophila melanogaster[online]”, (29 Nisan 2024), https://en.wikipedia.org/wiki/Drosophila_melanogaster#/media/File:Drosophila_melanogaster_Proboscis.jpg, (2024).

Xing, B., & Gao, W. J. “Innovative computational intelligence: a rough guide to 134 clever algorithms”, (Vol. 62, pp. 22-28). Cham: Springer International Publishing, 166-170, (2014).

Yang, X. S., & He, X., “Bat algorithm: literature review and applications”, *International Journal of Bio-Inspired Computation*, 5(3), 141-149, (2013).

Yuan, X., Dai, X., Zhao, J., & He, Q., “On a novel multi-swarm fruit fly optimization algorithm and its application”, *Applied Mathematics and Computation*, 233, 260-271, (2014).