T.C.
SÜLEYMAN DEMİREL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

İKİ BOYUTLU DÜZENSİZ ŞEKİLLERİ YERLEŞTİRME OPTİMİZASYONU

Fatih Ahmet ŞENEL

Danışman
Prof. Dr. Tuncay YİĞİT
II. Danışman
Dr. Öğr. Üyesi Asım Sinan YÜKSEL

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
ISPARTA - 2019

TEZ ONAYI

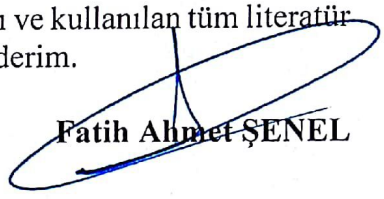
Fatih Ahmet ŞENEL tarafından hazırlanan **"İki Boyutlu Düzensiz Şekilleri Yerleştirme Optimizasyonu"** adlı tez çalışması aşağıdaki jüri üyeleri önünde Süleyman Demirel Üniversitesi Fen Bilimleri Enstitüsü **Bilgisayar Mühendisliği Anabilim Dalı**'nda **DOKTORA TEZİ** olarak başarı ile savunulmuştur.

Danışman	**Prof. Dr. Tuncay YİĞİT** Süleyman Demirel Üniversitesi	.....
Jüri Üyesi	**Doç. Dr. Gültekin ÖZDEMİR** Süleyman Demirel Üniversitesi	.....
Jüri Üyesi	**Doç. Dr. Ali Hakan IŞIK** Mehmet Akif Ersoy Üniversitesi	.....
Jüri Üyesi	**Dr. Öğr. Üyesi Fatih GÖKÇE** Süleyman Demirel Üniversitesi	.....
Jüri Üyesi	**Dr. Öğr. Üyesi Mehmet Fatih DEMİRAL** Mehmet Akif Ersoy Üniversitesi	.....

Enstitü Müdürü **Doç. Dr. Şule Sultan UĞUR** .....

TAAHHÜTNAME

Bu tezin akademik ve etik kurallara uygun olarak yazıldığını ve kullanılan tüm literatür bilgilerinin referans gösterilerek tezde yer aldığını beyan ederim.



Fatih Ahmet ŞENEL

İÇİNDEKİLER

Sayfa

ÖZET

Doktora Tezi

İKİ BOYUTLU DÜZENSİZ ŞEKİLLERİ YERLEŞTİRME OPTİMİZASYONU

Fatih Ahmet ŞENEL

Süleyman Demirel Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Prof. Dr. Tuncay YİĞİT

Bu tez çalışmasında, deri ayakkabı sektörünün kullanımına uygun otomatik yerleşim yapabilen bir yazılım geliştirilmiştir. Geliştirilen yazılım için, yeni bir hibrit optimizasyon yaklaşımı sunulmuştur. Bu hibrit yaklaşım Parçacık Sürü Optimizasyonu (PSO) ve Gri Kurt Optimizasyonu (GKO) algoritmalarının kullanılması ile geliştirilmiştir.

Geliştirilen algoritma (HPSGKO), sekiz farklı optimizasyon yöntemi ile karşılaştırılmıştır. Bu yöntemler; Yapay Arı Kolonisi (YAK), Tavlama Benzetimi (TB), Genetik Algoritma (GA), PSO, Sosyal Örümcek Algoritması (SÖA), GKO, Diferansiyel Gelişim (DG), Lineer popülasyon büyüklüğü azaltmalı başarılı parametrelerin kayıt altına alındığı hafıza tabanlı DG (L-SHADE) algoritmalarıdır. Tüm optimizasyon yöntemleri, iki boyutlu düzensiz şekilleri yerleştirme probleminin çözümünde kullanılmıştır. YAK, SÖA ve GKO algoritmaları literatürde yeni olmaları nedeniyle iki boyutlu dizilim problemine ilk defa bu tez çalışması ile birlikte uyarlanmıştır. Aynı şekilde sürekli gelişmekte olan DG ve L-SHADE algoritmaları da bu probleme ilk defa uyarlanmıştır.

Geliştirilen yazılım, yukarıda sayılan dokuz adet optimizasyon yöntemlerinden istenilenin seçilmesi ile dizilim işlemini gerçekleştirebilmektedir. Ayrıca, her bir yöntemin seçilen materyal ve kalıp örnekleri ile otomatik olarak 10'ar defa çalıştırılması sonucunda elde edilen analiz sonuçları, grafiksel olarak incelenebilmektedir.

Sonuç olarak, yapılan analiz işlemleri sonucuna göre, problemin çözümüne en uygun yöntem HPSGKO algoritmasıdır. Deri ayakkabı üretiminde kullanılabilecek, yerli bir yazılım geliştirilmiştir. Bahsedilen problemin çözümü başarılı bir şekilde sonuçlandırılmıştır.

Anahtar Kelimeler: Deri ayakkabı, dizilim, hibrit algoritma, iki boyutlu yerleşim, optimizasyon.

2019, 148 sayfa

ABSTRACT

PhD. Thesis

PLACEMENT OPTIMIZATION OF TWO-DIMENSIONAL IRREGULAR PATTERNS

Fatih Ahmet ŞENEL

Süleyman Demirel University
Graduate School of Natural and Applied Sciences
Department of Computer Engineering

Supervisor: Prof. Dr. Tuncay YİĞİT

In this thesis, a software that can make automatic placement suitable for the use of leather shoe-making industry is developed. We introduce a novel hybrid algorithm for the developed software. This hybrid approach is developed using Particle Swarm Optimization (PSO) and Grey Wolf Optimizer (GWO) algorithms.

The developed algorithm (HPSGWO) is compared with eight different optimization methods. These methods are Artificial Bee Colony (ABC), Simulated Annealing (SA), Genetic Algorithm (GA), PSO, Social Spider Algorithm (SSA), GWO, Differential Evolution (DE) and Improving the Search Performance of SHADE Using Linear Population Size Reduction (L-SHADE) algorithm. All optimization methods are used to solve the two-dimensional placement problem. ABC, SSA and GWO algorithms are new in the literature, and these algorithms have been adapted to the problem of two-dimensional placement for the first time with this thesis. Similarly, DE and L-SHADE algorithms, which are constantly evolving, have also been adapted to this problem for the first time.

The developed software can perform the nesting process by selecting the desired one of the nine optimization methods mentioned above. In addition, the results of the analysis obtained with the selected material and pattern samples of each method can be examined graphically after running 10 times automatically.

As a result, according to the results of the analysis, the most suitable method for solving the problem is the HPSGWO algorithm. A native software is developed for leather shoe-making sector. The solution of the mentioned problem is successfully concluded.

Keywords: Leather shoes, nesting, hybrid algorithm, two-dimensional placement, optimization.

2019, 148 pages

TEŞEKKÜR

Bu araştırma için beni yönlendiren, karşılaştığım zorlukları bilgi ve tecrübesi ile aşmamda yardımcı olan değerli danışman hocam Prof. Dr. Tuncay YİĞİT'e ve yine tezimde ikinci danışmanlığımı yapan her türlü bilgi ve tecrübesini benden eksik etmeyen Dr. Öğr. Üyesi Asım Sinan YÜKSEL'e teşekkürlerimi sunarım.

Araştırmanın yürütülmesinde maddi ve manevi yardımlarını gördüğüm ve tez izleme komitesinde bulunan Doç. Dr. Gültekin ÖZDEMİR ve Dr. Öğr. Üyesi Fatih GÖKÇE'ye teşekkür ederim.

4252-D2-15 No'lu Proje ile tezimi maddi olarak destekleyen Süleyman Demirel Üniversitesi Bilimsel Araştırma Projeleri Yönetim Birimi Başkanlığı'na teşekkür ederim.

Tezimin her aşamasında beni yalnız bırakmayan eşime ve aileme sonsuz sevgi ve saygılarımı sunarım.

Fatih Ahmet ŞENEL
ISPARTA, 2019

ŞEKİLLER DİZİNİ

Sayfa

ÇİZELGELER DİZİNİ

Sayfa

SİMGELER VE KISALTMALAR DİZİNİ

CNC	Computer Numerical Control
DG	Diferansiyel Gelişim Algoritması
DXF	Drawing eXchange Format
EPGL	Hewlett-Packard Graphics Language
ESICUP	The EURO Special Interest Group on Cutting and Packing
GA	Genetik Algoritma
GPU	Graphics Processing Unit
GKO	Gri Kurt Optimizasyonu
HPGL	Hewlett-Packard Graphics Language
HPSGKO	Hibrit PSO-GKO algoritması
L-SHADE	Lineer popülasyon büyüklüğü azaltmalı başarılı parametrelerin kayıt altına alındığı hafıza tabanlı DG
MD	Mesafe Dönüşüm
PLT	HPGL plotter - Vector graphics (AutoCAD Plot)
PSO	Parçacık Sürü Optimizasyonu
SKP	Stok Kesim Problemi
SÖA	Sosyal Örümcek Algoritması
SYP	Stok Yerleşim Problemi
TB	Tavlama Benzetimi
YAK	Yapay Arı Kolonisi

1. GİRİŞ

Günümüz teknolojisinin gelişmesi sonucunda artık sanayinin birçok alanında bilgisayarla kontrol edilebilen endüstriyel sistemler kullanılmaktadır. Özellikle üretim ile ilgili sanayileşmede, insan gücü neredeyse kullanılmamaktadır. Çünkü insan faktörü, yorgunluk, dikkat eksikliği, iş kazaları vs. gibi dezavantajlar barındıran faktörleri içermektedir. Durum böyle olunca, çalıştıkça yorulmayan, dikkat eksikliği olmayan ve iş kazalarını en aza indirebilen sistemlerin kullanılması zorunlu hale gelmiştir.

Bu tez çalışmasının konusu; deri, kâğıt, kumaş gibi malzemelerle üretim yapan firmaların üretecekleri ürünler için gerekli olan parçaları, kullanılacak malzemeden otomatik olarak en az fire verecek şekilde kesmelerini sağlayacak bir optimizasyon yazılımı geliştirmektir. Deri, kâğıt, kumaş gibi sektörlerde faaliyet gösteren firmaların ellerinde bulunan materyalden en az fire verecek şekilde kullanılacak alanı otomatik olarak en iyi biçimde belirleyen ve bulduğu kesim bölgelerini materyal üzerine çizen/kesen bir yazılım geliştirilmiştir. Çalışma, doğal deriden ayakkabı kalıplarının en az fire ile çıkarılmasını kapsamaktadır. Bu nedenle yapılan tüm çalışmalar deri ve ayakkabı modelleri ile gerçekleştirilmiştir. Deri materyalin diğer materyallerden farklı olarak geri dönüşüm işlemi olmayan bir madde olması, bu tezde konu olarak seçilmesini ön plana çıkarmıştır. Yukarıda bahsedilen diğer materyallerin hepsi geri dönüşümü mümkün olan materyallerdir. Her ne kadar geri dönüşüm işlemi de bir maliyet getirse de derinin geri dönüşümünün olmamasından daha avantajlıdır. Bu nedenle en uygun yerleşim işlemleri gerektiren üretim tezgâhlarında en fazla dikkat gereksinimi duyulan ham madde deridir.

Ülkemizde, deri ayakkabı üretimi yapan firmalara yapılan ziyaretlerde görülmüştür ki, çok az firma otomatik dizilim ve kesim yapan makineleri kullanmaktadır. Genel olarak dizilim işlemi için uzun yıllar tecrübesi olan ustalar kesim işlemi için baskı makineleri kullanmaktadırlar (Şekil 1.1). Bu işlemlerin başlıca dezavantajları dizilim işlemi yapılırken tecrübeli ustaların baskı makinesinin tablasına sığacak kadarını tablaya sermesi ve sadece tabla üzerinde gördüğü alandaki en uygun yere kalıbı yerleştirmesidir. Ancak tablaya o anda serilmemiş olan derinin bir başka bölümünde çok daha uygun bir yerleşim alanı varsa bunu görememektedir. Bu nedenle verilen

fireler özellikle orta ve büyük ölçekli firmalarda büyük zararlara yol açmaktadır. Kesim işleminin dezavantajı ise, her ayakkabı modeli için özel olarak metal kesim kalıplarının yaptırılması zorunluluğudur. Her ne kadar metal kesim kalıplarının geri dönüşü mümkün olsa da ciddi maliyetler doğurmaktadır. Denizli Organize Sanayi Bölgesinde yer alan Sarıkaya Ayakkabı San. Tic. Ltd. Şti. yetkilileriyle yapılan görüşme ve fabrika içi yapılan incelemelerde (2014 yılında) yıllık ortalama 50.000 ila 60.000 TL kalıp masrafının olduğu rapor edilmiştir.

Şekil 1.1. Deri Baskı Makinesi

Geliştirilecek olan yazılım ile hem kalıp masrafları ortadan kalkacak hem de elde mevcut olan kalıplarla sınırlı olan üretim kısıtlaması ortadan kalkacaktır. Çünkü kullanıcı, istediği ayakkabı modelini bilgisayar ortamına aktardığı an, materyal üzerine dizilimini hemen yaptırıp kesim işlemini gerçekleştirebilecektir. Metal kesim kalıplarının hazırlanması aşaması otomatik olarak devre dışı kalacaktır. Orta ölçekli bir firmada yıllık olarak 50.000 ila 60.000 TL arasında bir maliyete gerek kalmayacak ve firma daha fazla kâr elde edebilecektir. Ayrıca insan kaynaklı problemlerin de ortadan kaldırılmış olması amaçlanmaktadır.

Ülkemizde her ne kadar otomasyon sistemleri kullanarak üretim yapan firmalar bulunsa da çoğunluğu hala otomasyon sistemlerini değil insan gücü ve becerisini

kullanmaktadır. Özellikle deri kesiminde, kesim noktalarına insanlar tarafından karar verilmekte ve bu işe yıllarını vermiş çalışanlar dahi, belli bir yorgunluk seviyesinden sonra hatalar yapabilmekte ve istenmeyen fireler verebilmektedirler.

Bahsedilen bu firelerin önüne geçmek için uzun hesaplamalar yapılması gerekmekte ve bu hesap için tek bir kişinin yerine birden fazla kişinin istihdam edilmesi gerekmektedir. Böyleyken dahi, hatalar yapılabilmekte ve yapılan hatalardan da geri dönüş olamayacağı için çok büyük miktarda zarara neden olabilmektedir. Ayrıca hesaplamaların fazla zaman harcaması nedeniyle işverenler, verilen fireleri göz ardı etmektedirler.

En önemli hata nedeni insanlardaki yorgunluk faktörüdür. Yorgunluğun da belli başlı nedenleri;

- Uzun süre aralıksız çalışma,
- Sürekli benzer renk tonlarına odaklanma,
- Uykusuzluk,
- Aşırı stres gibi daha sayılabilecek birçok neden bulunmaktadır.

Bu nedenlerin birçoğu da normal yaşamını sürdüren her insanda görülebilecek durumlardır. Fakat bu durumlardan bir ya da bir kaçı ile karşılaşan insanlar Stok Yerleşim Problemi (SYP)'ne ihtiyaç duyulan sektörlerde (deri, kâğıt, metal vb.) çalıştığında bu hesaplamaları yanlış yapabilmekte ve büyük maddi zarara yol açabilmektedirler.

Bir başka problem ise kişisel farklılıklardır. En uygun kesim bölgesine kişiler kendi yaptıkları hesaplamalar ile ya da görsel olarak bakıp karar vermektedirler. Doğal olarak böyle bir durumda bir kişinin bulduğu en uygun kesim bölgesi ile başka bir kişinin bulduğu en uygun kesim bölgesi birbirinden farklı olabilmektedir.

Üretim yapan her hangi bir sektörde ne kadar malzemeden ne kadar ürünün çıkacağının hesabı yapılabilmektedir. Fakat konu deri ayakkabı sektörü olunca bahsedilen bu kişisel farklardan dolayı ne kadar girdiyle ne kadar ürün çıkacağının hesabı tam olarak yapılamamaktadır.

Bu nedenle, bahsedilen üretim işletmelerinde; çalıştıkça yorulmayan, aynı materyal için her zaman aynı en iyi sonucu garantileyen, hızlıca sonuca götürebilecek bir otomatik yerleştirme işlemi yapan yazılımın kullanılması zorunlu bir ihtiyaçtır.

Ayrıca, ülkemizde makine sanayi olarak yeterince imkân bulunmaktadır. Otomatik kesim işlemini yapabilen makineler rahatlıkla ülkemizde üretilebilmektedir. Yine Denizli Organize Sanayi Bölgesinde yer alan Filiz Makine yetkilileri ile yapılan görüşmede, makine üretimi konusunda hiçbir zorluğun olmadığı ancak yazılım tedariki konusunda sıkıntılar olduğu anlaşılmıştır. Ayakkabı üretimi yapan firmalar için baskı makinesi üreten Filiz Makinenin, ayrıca otomatik kesim yapan makinelerin de tedariki konusunda bulunduğu bölgede pazar payının büyük oranda sahibi olduğu bilinmektedir. Makine üretimi konusunda yerli imkânların mevcut olduğu ancak yazılımların yurt dışından alınma zorunluluğu olduğu yapılan görüşmeler sonucunda anlaşılmıştır.

Tüm bu bahsedilen durumlar dikkate alındığında otomatik yerleştirme işlemini en az fire verecek şekilde yapan bir yazılıma olan ihtiyacın oldukça fazla olduğu ortadadır.

Bu tezin konusu literatür kapsamında incelenecek olursa, literatürde SYP ya da Stok Kesim Problemi (SKP) olarak bilinen, düzenli ya da düzensiz geometrik şekillere sahip bir materyalden yine düzenli ya da düzensiz geometrik şekle sahip kalıpların en az fire verecek şekilde kesilerek çıkarılması problemlerine dâhil olduğu görülmektedir. Bu işlem gerçekleştirilirken firmalar için maliyeti azaltabilecek bazı önlemler yer almaktadır. Bu önlemleri başlıca şu şekilde sıralayabiliriz;

- Yerleşim işlemi sonunda en az fire verilmeli,
- En fazla kalıp sığacak şekilde en uygun yerleşim hesaplanmalı,
- Deforme olan bölgeler içerebilen materyallerde (örneğin deri) arızalı bölgeler yerleşim işlemine dâhil edilmemeli,
- Yerleşim işlemi sonunda üst üste çakışan kalıplar olmamalı,
- Tüm yerleşim, işlemi verilen materyalin sınırları içinde olmalı,
- İstenilen tüm parçalar verilen hammaddeden çıkarılabilmeli, gibi durumlardır.

SYP; deri, metal, tekstil, kâğıt vb. hammaddeleri kullanarak üretim yapan sektörlerde sıkça rastlanılan bir kaynak yönetimi problemidir. SYP, kendi içinde alt problemlere ayrılmaktadır;

- 1 boyutlu – çapı sabit, uzunluğu değişken (tel, boru vs.)
- 1.5 boyutlu – eni sabit, uzunluğu değişken (tekstil kumaş vs.)
- 2 boyutlu – eni ve boyu değişken (metal, cam, deri vs.)
  - Düzensiz şekil (deri vs.)
  - Düzenli şekil (metal, kâğıt vs.)
- 2.5 boyutlu – eni ve yüksekliği sabit, uzunluğu değişken (Konteyner vs.)
- 3 boyutlu – en, boy ve yükseklik değişken (Palet vs.)
  - Düzensiz şekil
  - Düzenli şekil
- 3'den fazla boyutlu – en, boy, yükseklik, ağırlık, öncelik vs. değişken.

Bu tez çalışmasında, şekil ve özellikleri birbirinden farklı olabilen deri materyal üzerine, istenilen ayakkabı modellerini en uygun şekilde dizebilen bir yazılım geliştirilmiştir. Geliştirilen yazılımda dokuz farklı optimizasyon yöntemi kullanılmıştır. Kullanılan bu yöntemler farklı materyal ve kalıp modelleri ile test edilerek birbirleriyle karşılaştırılmıştır. Kullanılan optimizasyon yöntemleri; Yapay Arı Kolonisi (YAK), Tavlama Benzetimi (TB), Genetik Algoritma (GA), Parçacık Sürü Optimizasyonu (PSO), Sosyal Örümcek Algoritması (SÖA), Gri Kurt Optimizasyonu (GKO) algoritması, Diferansiyel Gelişim (DG) algoritması, Lineer popülasyon büyüklüğü azaltmalı başarılı parametrelerin kayıt altına alındığı hafıza tabanlı DG (L-SHADE) algoritması ve son olarak PSO ve GKO algoritmalarının birlikte kullanılarak bu tez kapsamında geliştirilen Hibrit PSO-GKO (HPSGKO) algoritmalarıdır. YAK, SÖA ve GKO algoritmaları literatürde yeni olmaları nedeniyle tezin konusu olan probleme uyarlanmalarına rastlanmamıştır. Aynı şekilde, sürekli gelişmekte olan ve bu tezde kullanılan güncel DG ve L-SHADE algoritmaları da bu probleme ilk defa uyarlanmıştır. HPSGKO algoritması bu tez kapsamında geliştirilmiş, PSO ve GKO algoritmalarından daha başarılı sonuçlar elde edilmiştir. Böylelikle, yeni olan bu yöntemler, tezin konusuna özgü bir probleme uyarlanmış ve literatüre kazandırılmıştır.

Geliştirilen yazılım, proje kapsamında alınan bir çizici ile testlere tabi tutulmuş ve kesim işlemi yapan makinelere doğrudan çıktı verebilecek düzeye getirilmiştir. Normal bir kesim makinesi maliyeti yüksek olduğundan ve tez çalışmasında satın alınma imkânı olmamasından dolayı daha az maliyetli bir çizim makinesi ile testler gerçekleştirilmiştir (Şekil 1.2). Kesim işlemi yapan makineler büyük çoğunlukla *.DXF dosya formatlarını tanımakta, az bir kısmı da *.PLT uzantılı dosyalar ile kontrol edilmektedir. Tezde kullanılan çizici cihazı da *.PLT uzantılı dosyalar ile kontrol edilmektedir. Geliştirilen yazılım hem *.PLT hem de *.DXF uzantılı dosyaları çıktı olarak verebilecek özellikte tasarlanmıştır.

Şekil 1.2. Üç Eksen Çizim Makinesi

Tez aşamaları gerçekleştirilirken, ilk olarak en basit örneklerden yola çıkılarak gerçeklenmeye başlanmış ve düzenli şekle sahip bir alan üzerine yine düzenli şekle sahip şekillerin dizilmesi amaçlanmıştır. Yeterince başarılı sonuçlar alındıktan sonra kalıpların şekilleri değiştirilerek düzensiz şekilli kalıplar kullanılmıştır. Son aşamada materyal olarak deri kullanılarak problemin çözümü elde edilmiştir. Bu problem, literatürde iki boyutlu stok kesim problemi olarak kategorize edilmektedir (Wäscher vd., 2007). Stok kesim problemi de kendi içinde dört farklı gruba ayrılmaktadır (H.-H. Yang ve Lin, 2009);

1. Düzenli şekle sahip materyal üzerine, düzenli şekle sahip kalıpların dizilmesi
2. Düzenli şekle sahip materyal üzerine, düzensiz şekle sahip kalıpların dizilmesi
3. Düzensiz şekle sahip materyal üzerine, düzenli şekle sahip kalıpların dizilmesi

4. Düzensiz şekle sahip materyal üzerine, düzensiz şekle sahip kalıpların dizilmesidir.

Bu tez çalışması, yukarıda söylenen dördüncü maddeye konu olmaktadır. Çünkü kullanılan deri materyaller doğal deri olduğu için hem şekilleri hem de kalitesi birbirlerinden farklıdır. Düzenli bir şekle sahip değildir. Aynı şekilde kullanılan ayakkabı kalıp modelleri de düzenli şekillere sahip değillerdir.

Sonuç olarak amaçlanan hedefler maddeler halinde yazılacak olursa;

1. Kesim işleminde, insan faktöründen kaynaklanan hataları en aza indirgeyerek işletmeye ve ülke ekonomisine katkı sağlamak,
2. Kesim işleminde standart bir kaliteyi sağlamak ve sağlanan bu kaliteyi kayıt altına almak,
3. Ülkemizde üretilen ancak yazılımları dışardan ithal edilen makinelerin yazılım ihtiyaçlarını ülkemizden karşılamak,
4. Bu tez ile birlikte akademik olarak literatüre yeni sonuçlar kazandırmaktır.

Bu problem, dokuz farklı optimizasyon yöntemi ile çözüme kavuşturulmuştur. Sonuç olarak en başarılı olanlar ve bu probleme uygun olmayan optimizasyon yöntemleri belirlenmiştir. Problemin çözümünü sağlayacak yazılım için Microsoft Visual Studio platformu ve C# programlama dili kullanılmıştır. Yukarıda anlatılan ve problemin çözüm aşamalarına konu olan tüm detaylar ilerleyen bölümlerde anlatılacaktır.

Bu tezin birinci bölümünde, teze konu olan problem, geliştirilmiş olan yazılıma olan ihtiyaç, tezde kullanılmış olan yöntemler kısaca açıklanmıştır. İkinci bölümde literatürde yer alan düzenli ve düzensiz şekillere sahip iki boyutlu dizilim konusunu içeren çalışmaların sonuçları verilmiştir. Tez konusu ile ilgili çalışmalar da yine bu bölümde belirtilmiştir. Üçüncü bölümde kullanılan materyal ve yöntem hakkında bilgiler verilmiştir. Dizilim işlemlerinin test edilebileceği bir çizici cihazı tanıtılmış, kullanılan optimizasyon yöntemleri detaylı olarak açıklanmıştır. Dördüncü bölümde tezin gerçekleştirme aşamaları ele alınmış, kullanılan maliyet fonksiyonları detaylı olarak anlatılmıştır. Ayrıca kullanılan optimizasyon yöntemlerinin kodlanması aşaması da yine bu bölümde anlatılmıştır. Geliştirilen yazılımın detaylı olarak açıklanması, kullanılan optimizasyon yöntemlerinin karşılaştırılması işlemleri de bu

bölümde anlatılmıştır. Beşinci bölümde ise elde edilen tez çıktıları, sonuçlar ve gelecekte yapılması hedeflenen çalışmalar sunulmuştur.

2. KAYNAK ÖZETLERİ

Albano ve Sapuppo (1980), yaptıkları çalışmada, genişliği sabit uzunluğu değişken olan bir yerleşim alanı üzerinde çalışarak, parçaları sırayla yerleşim alanına sol alttan başlayarak yerleştirmişlerdir. Her yerleşim sonucunda yerleşimi çevreleyen dikdörtgen alan hesaplanıp boşta kalan fire alanları hesaplanmıştır.

Rutenbar (1989), yaptığı çalışmada, elektronik devre tasarımında kullanılan yongaların en az yer ve en kısa yol yapılacak şekilde bakır plaka üzerine dizilimini gerçekleştirmiştir. Çalışmada maliyet fonksiyonu olarak tüm yongalar yerleştikten sonra, tüm alanı çevreleyen dikdörtgenin alanı ve yongaların arasına yapılan bakır yolların uzunlukları toplamı dikkate alınmıştır. Sezgisel yöntemlerden olan TB yöntemi kullanılmıştır.

Dagli ve Hajakbari (1990), TB yöntemini kullanarak özellikle tekstilde ihtiyaç duyulan iki boyutlu şekillerin en uygun şekilde belirlenen bir alana sığdırılması üzerinde çalışmışlardır.

Marques vd. (1991), yaptıkları çalışmada, TB yöntemini kullanarak, tekstil üretiminde kumaşların en az fire olacak şekilde, düzensiz şekle sahip alanların kesilme noktalarının tespit edilmesi işlemini gerçekleştirmişlerdir. Maliyet hesaplaması olarak, tüm parçaları çevreleyen dikdörtgenin alanı ve her bir parçanın merkez noktasının yerleşim alanının merkez noktasına olan uzaklıkları toplamını kullanmışlardır.

Gomes ve Oliveira (2002), yaptıkları çalışmada, düzensiz şekle sahip nesneleri sabit genişlikli ve sonsuz uzunluklu düzenli bir alana en kısa uzunluk olacak şekilde yerleştirmişlerdir. Nesneleri ikişerli gruplar şeklinde düşünerek en uygun birleşen nesneleri öncelikli olarak birleştirmişlerdir. Bu işlemi yaparken birleşen parçaların minimum alan, minimum düzensiz şekil ve dikdörtgene en yakın şekli oluşturması kontrol edilmiştir. No-fit-polygon (NFP) ve Inner-fit-rectangle (IFR) yöntemlerini kullanmışlardır.

Gomes ve Oliveira (2004), yaptıkları çalışmada No-Fit Polygon (NFP) tekniğini ve TB algoritmasını kullanarak iki boyutlu düzensiz şekle sahip giysi kalıplarını en uygun

olacak şekilde dizmişlerdir. Yerleşim alanı olarak sabit genişlik ve sonsuz uzunluğa sahip bir alan seçmiş ve maliyet olarak en az uzunluk olacak şekilde hesaplamalar yapmışlardır.

Mahmoud vd. (2004), yaptıkları çalışmada, GA kullanarak iki boyutlu nesnelerin yerleştirilmesi işlemini gerçekleştirmişlerdir. Üç farklı yöntem denemişlerdir; ilk olarak her bir parçayı çevreleyen dikdörtgeni bulup dikdörtgenleri sırayla dizmişlerdir, ikinci olarak her bir parçayı kapsayacak en küçük alana sahip bir geometrik şekil hesaplayıp sırayla dizmişlerdir ve son olarak her parçayı olduğu gibi kabul ederek sırayla dizmişlerdir. Maliyet hesaplaması olarak, yerleşim alanındaki boşluklar ve üst üste çakışan nesnelerin çakışma miktarları hesaplanmıştır.

Whitwell (2004), yaptığı doktora tezi çalışmasında, iki boyutlu stok kesim problemini ele almıştır. Düzenli şekle sahip materyal ve kalıpların en uygun yerleştirilmesini farklı sezgisel yöntemler kullanarak gerçekleştirmiştir. Yine aynı şekilde, düzensiz şekle sahip kalıpların da dizilimini farklı sezgisel yöntemler kullanarak gerçekleştirmiştir.

Crispin vd. (2005), yaptıkları çalışmada, deri ayakkabılarda kullanılan kalıpları, doğal deri üzerine dizmeyi amaçlamışlardır. GA kullanarak yaptıkları çalışmada ayrıca NFP yöntemini kullanarak kalıpların en uygun yerleşeceği konumu tespit etmişlerdir. Ayrıca kalıpları ikili resim olarak temsil etmişler ve en uygun konum bulunduktan sonra çokgen olarak yerleşim işlemini gerçekleştirmişlerdir.

Liu ve He (2005), yaptıkları çalışmada NFP tekniğini kullanarak iki boyutlu düzensiz şekle sahip nesneleri en uygun şekilde dizmişlerdir. Her bir nesnenin ağırlık merkezini hesaplayarak, yerleşim alanının en altına en yakın olacak şekilde nesneleri döndürerek dizmişlerdir. Ayrıca GA kullanarak nesnelerin dizilim sırasını da optimize etmişlerdir.

Vassiliadis (2005), yaptığı çalışmada ikili arama ağacı yöntemini kullanarak, iki boyutlu düzenli şekle sahip nesneleri düzenli şekle sahip bir platform üzerine yerleştirmeyi amaçlamıştır. Çalışmasında TB ve Threshold Accepting (TA) optimizasyon yöntemlerini kullanmıştır.

Yuping vd. (2005), yaptıkları çalışmada deri materyal üzerine, düzensiz şekilli kalıpları dizmeyi amaçlamışlardır. Kalıpları yerleştirirken, kaydırma işlemini kullanarak çakışma ve materyalden taşmaları önleyecek şekilde çalışmışlardır. Kalıplar için üçer derecelik döndürme açılarına izin vermişlerdir. Algoritmalarını TB kullanarak yazmışlardır. TB algoritmasında gerekli durumlarda sıcaklık düşürme işlemini bir adım geriye alarak, yeniden soğutma işlemini tekrarlamışlardır.

Wong vd. (2008), yaptıkları çalışmada GA kullanarak nesneleri dikdörtgen şekilde düşünüp yerleşim alanına sol en alttan başlayarak dizmiştir. GA ile parçaların dizilim sırası optimize edilmeye çalışılmıştır.

Lee vd. (2008), yaptıkları çalışmada iki boyutlu düzensiz nesneleri düzensiz bir platforma yerleştirmeyi amaçlamışlardır. Çalışmalarında düzensiz şekle sahip nesneleri, nesneleri çevreleyecek şekilde en küçük çokgenlere dönüştürerek yerleşim işlemlerini gerçekleştirmişlerdir. Çokgenleri oluştururken hassasiyet olarak farklı açı değerleri ile çalışmışlardır. Açı hassasiyeti arttıkça çokgenin kenar sayısı artmaktadır. C++ ile geliştirdikleri yazılımda Auto-CAD programı formatında nesneleri veri girdisi olarak alıp sonucu yine aynı formatta dışarı aktarmışlardır.

Ramakrishnan vd. (2008), yaptıkları çalışmada Tabu Search (TS), GA ve TB yöntemlerini kullanmışlardır. İki farklı yöntemle (Iterative construction heuristics ve Searching over layout) çalışmış ve sonuçları karşılaştırmışlardır. NFP tekniği ile arama uzayını daraltmış ve süreyi azaltmışlardır. Maliyet fonksiyonu olarak üst üste çakışmaları ve materyalin uzunluğunu kullanmışlardır.

Elamvazuthi vd. (2009), yaptıkları çalışmada, mobilya endüstrisinde kullanılan deri materyallerin kesimi için el ile yerleşim ve otomatik yerleşim işlemlerini süre ve verim olarak karşılaştırmışlardır. Sonuçta hem süre hem de verim açısından otomatik dizilim işlemi yapan yazılımların, el ile dizilim işleminden daha iyi olduğunu göstermişlerdir.

Imamichi vd. (2009), yaptıkları çalışmada NFP tekniğini ve kendi geliştirdikleri doğrusal olmayan bir yöntem kullanmışlardır (Iterated local search algorithm-ILSQN). Maliyet fonksiyonunda materyalin sınırlarından taşan miktarı ve üst üste çakışan

miktarları hesaplanmış ve maliyet minimize edilecek şekilde yerleşim işlemini gerçekleştirmişlerdir.

Yang ve Lin (2009), yaptıkları çalışmada GA kullanarak deri ayakkabı için gerekli kalıp şekillerini en az fire olacak şekilde yerleştirmişlerdir. Çalışmalarında, düzensiz şekillerden oluşan kalıpları, düzenli bir şekle sahip suni deri üzerine yerleştirmişlerdir. Maliyet fonksiyonu olarak sabit genişliğe sahip olan suni derinin uzunluğunu minimum yapmayı amaçlamışlardır. Ayrıca kullandıkları kalıplar bir ya da iki farklı şekle sahiptir.

Zhang ve Yang (2009), yaptıkları çalışmada GA ve TB yöntemini birlikte kullanan bir yöntem geliştirmişlerdir. GA ile nesnelerin dizilim sırasını optimize etmeye çalışmışlar ve TB yöntemi ile de nesneler yerleştikçe mutasyon oranını değiştirerek her yerleşimde en iyi mutasyon oranını hesaplamışlardır. Çünkü geliştirdikleri yöntemde mutasyon oranı en önemli faktördür ve çok büyük ya da çok küçük olması iyi sonuçlar vermemekte, her yerleşim işleminde farklı mutasyon oranları ile daha iyi sonuçlar elde etmektelerdir. Bu nedenle TB yöntemi ile mutasyon oranını optimize etmeye çalışmışlardır.

Adakcı (2010), yaptığı çalışmada, alüminyum sektöründe kullanılan tek boyutlu kalıpların yine düzenli şekle sahip materyalden kesilmesi problemini ele almıştır. Problemin çözümünde algoritmik, sezgisel ve metasezgisel yöntemlerin kullanımını araştırmıştır. Metasezgisel yöntemlerde, karınca kolonisi algoritması, TB, tabu araştırması gibi algoritmaları kullanmıştır. ICRON isimli programla kendi çözüm algoritmasını geliştirmiş ve elde ettiği sonuçları karşılaştırmıştır.

Weng ve Kuo (2011) AutoCAD programı ile NEST isimli bir otomatik dizilim yazılımı geliştirmişlerdir. Geliştirdikleri yazılım, düzenli şekle sahip materyal üzerine düzenli ve düzensiz şekilli kalıpları dizmeyi amaçlamaktadır. Dizilim işlemi için üç parametre kullanmışlardır. Bunlar; hassasiyet, döndürme açısı ve hata oranıdır. Döndürme açısı belirlendikten sonra, her bir kalıbın tüm açılarını kapsayacak şekilde yeni bir kalıp seti oluşturulmuştur. Daha sonra Sol-Alt yaklaşımı ile en uygun yerleşen açı değeri bulunmuştur. Ayrıca çalışmalarında dikdörtgen şekilli materyallerin tek bir

kenarının düzgün olmadığı örneklerle denemeler yaparak kısmen düzensiz şekilli materyallerle de çalışmışlardır.

Akbulut (2012), yaptığı tez çalışmasında, dikdörtgen şekle sahip materyallerden düzensiz şekle sahip kalıpların en uygun şekilde dizilerek yerleştirilmesini amaçlamıştır. Problemin çözümünde kenar yaslama olarak isimlendirdiği yöntemi kullanmıştır. Bu yöntemle kalıplar arası boşlukları minimize etmiş ve kalıpları istenilen açılarda döndürülebilmesini sağlamıştır.

Bayır (2012), yaptığı tez çalışmasında, iki boyutlu düzensiz şekle sahip kalıpları eni sabit boyu değişken olan düzenli şekle sahip materyal üzerine dizmeyi amaçlamıştır. Bayır, bu problemin çözümü için GA kullanmıştır. Ayrıca problemin çözümü için Microsoft Visual Studio C# kullanarak bir uygulama geliştirmiştir. Bu tez çalışmasından farkı olarak düzenli şekle sahip materyal kullanmıştır.

Sato vd. (2012), yaptıkları çalışmada TB yöntemini kullanarak sabit genişlik değişken uzunluğa sahip bir yerleşim alanına düzensiz şekilleri yerleştirmişlerdir. Her bir nesneyi daha önce yerleştirilmiş olan nesnelerin kenarı doğrultusunda hareket ettirerek yerleşebileceği en uygun noktayı bulmuşlardır. En uygun noktaya karar verirken Convex Hull (CH) alanına bakmışlardır.

Elkeran (2013), yaptığı çalışmada, Cuckoo Search (CS) algoritmasıyla sabit genişlik değişken uzunluğa sahip yerleşim alanına düzensiz şekilleri en uygun şekilde yerleştirmiştir. Nesneleri ikili olarak gruplayarak en uygun grup oluşturan ikilileri tespit ederek devam ettirmiştir.

Martinez Sykora (2013), yaptığı doktora tezi çalışmasında, deri materyallere düzensiz şekilli kalıpları dizmiştir. Çalışmasında üç farklı kategori kullanmıştır. İlk olarak kalıpların döndürülmesine izin vermeyecek şekilde dizilim yapmıştır. Daha sonra kalıplara kısmi olarak döndürme açılarına izin vermiştir. Son olarak kalıpları giyotin ile kesilecek şekilde yerleşim işlemini gerçekleştirmiştir.

Shalaby ve Kashkoush (2013), yaptıkları çalışmada PSO algoritmasını kullanarak yerleşim optimizasyonunda parçaların geliş sıralarını optimize etmişlerdir. İki

aşamada çalışmalarını gerçekleştirmişlerdir. İlk aşamada parçaların geliş sırasını, ikinci aşamada ise parçaların yerleşim açısını optimize etmişlerdir. Geliştirdikleri çalışmalarını literatürde en fazla kullanılan 31 adet veri seti ile test etmiş ve 31 adet veri setinin 10 tanesinde daha iyi sonuca ulaşabilmişlerdir.

Timmerman (2013), yaptığı çalışmada DG algoritmasını kullanarak, düzensiz şekle sahip kalıpları düzenli bir alana dizmeyi amaçlamıştır. Bununla birlikte karınca kolonisi algoritması, tabu arama, TB ve GA'ları da kullanarak sonuçları karşılaştırmıştır. Kalıpları çevreleyen dikdörtgenler kullanarak çakışmaları, yakınlıkları hesaplamıştır. Sol-Alt yaklaşımını kullanarak kalıpları düzenli alana dizmiştir. Ayrıca kalıpları şekillerine göre gruplara ayırarak ta dizilim işlemi yaparak sonuçları gözlemlemiştir.

Dalalah vd. (2014), yaptıkları çalışmada, Convex Hull (CH) tekniği kullanarak nesneleri ikişerli gruplayarak yerleşim alanına dizmişlerdir. Maliyet hesaplaması olarak her nesnenin CH alanı ile kapladığı alanın farkını hesaplayıp o nesnenin CH alanına oranına bakmışlardır. Ayrıca maliyet hesabında nesnenin ağırlık merkezinin başlangıç noktasına olan uzaklığını da dikkate almışlardır.

Domovic vd. (2014), yaptıkları çalışmada üç farklı yöntem kullanarak iki boyutlu düzensiz şekle sahip tekstilde kullanılan elbise parçalarını sabit genişlik değişken uzunluklu bir platforma en uygun biçimde dizmişlerdir. "Random Search", "Greedy Algortihm" ve GA yöntemlerini kullanarak en uygun sonuçlar elde etmeye çalışmışlardır. Maliyet fonksiyonu olarak Convex Hull (CH) alanlarını hesaplamışlardır. Parçaları sırayla dizerek devam etmişlerdir. Tüm parçalar dizildikten sonra işlemi durdurmuşlardır.

Rocha (2014), yaptığı çalışmada, düzensiz şekle sahip kalıpları düzenli bir alana dizmeyi amaçlamıştır. Kalıpları literatürde var olan bir çok yöntemle temsil ederek sonuçları incelemiştir. Kalıpların temislinde farklı olarak çemberleri kullanmıştır. Çemberler vasıtasıyla çakışmaları ve kalıplar arasındaki yakınlık mesafelerini ayarlamıştır. Farklı stratejiler izleyerek dizilim işlemlerini gerçekleştirmiştir. Düşük çözünürlükle çalışarak kalıpları kabaca yerleştirmiş daha sonra çözünürlüğü artırarak hassas yerleşim işlemini tamamlamıştır. Diğer bir yöntem olarak iki faz yaklaşımını

kullanmıştır. İlk fazda büyük kalıpları çemberler vasıtasıyla birbirlerine yakınlaştırmıştır. İkinci fazda ise arada oluşan boşlukları daha küçük kalıplarla doldurmayı amaçlamıştır. Son olarak katman stratejisini kullanmıştır. Kalıp sayılarının çok fazla olduğu durumlarda işlem çok daha karmaşık olduğundan kalıpları gruplara bölerek, önce kendi içinde dizilim işlemini yapmıştır. Grupların her birini katman olarak isimlendirmiştir. Daha sonra tüm katmanları ard arda birleştirerek toplam yerleşim işlemini elde etmiştir.

Sato vd. (2014), çalışmalarında Voronoi Mountain yöntemini ve NFP tekniğini bir arada kullanarak düzensiz şekillere sahip kalıpları dikdörtgen bir alana dizmeyi amaçlamışlardır. Piksel (Raster yöntemi – Yani kalıpların piksellerle ifade edilmesi) tekniği kullanarak algoritmalarının hızlı olmasını sağlamışlardır. Tangram puzzle ve Fu problemlerine uyguladıkları yöntemleri iyi sonuçlar vermiştir.

Sherif vd. (2014), TB algoritması ile düzensiz şekle sahip kesim kalıpların düzenli şekle sahip bir platforma dizilim işlemini gerçekleştirmişlerdir. TB ile kalıpların dizilim sırasını optimize ederken, maliyet fonksiyonu olarak kesim başlığının, kesim sırasındaki izleyeceği yolun uzunluğunu dikkate almışlardır. Böylelikle parçalar mümkün olduğunca birbirine yakın olacak şekilde yerleştirme işlemi gerçekleştirilmiştir. Her bir kalıbı kapsayan minimum alanlı dikdörtgenler kullanarak algoritmalarını daha verimli hale getirmişlerdir.

Siasos ve Vosniakos (2014), yaptıkları çalışmada GA kullanarak tekstil ürünlerine ait kalıpları sabit genişlikli ve sonsuz uzunluklu materyale en az fire verecek şekilde dizilim yapmışlardır. Üç aşamalı bir çözüm üretmişlerdir. İlk olarak kalıpları piksel tabanlı olarak temsil etmişler ve Sol-Alt yaklaşımını kullanarak kabaca kalıbın yerini tespit etmişlerdir. Daha sonra sadece köşe noktalarının temsil edildiği düşük çözünürlüklü çokgenler kullanarak en sonda kalıpların tüm sınırları boyunca temsil edildiği yüksek çözünürlüklü çokgenler olarak temsil edip hassas yerleşim noktasını tespit etmişlerdir.

Yang (2014), yaptığı çalışmada, düzensiz şekle sahip kıyafet parçalarını Hybridizing Ant Algorithms tekniği kullanarak dizmiştir. Elde ettiği sonuçlar tekstilde fire oranını ve hesaplama zamanını kısaltacak şekildedir.

Anand ve Babu (2015), yaptıkları çalışmada iki boyutlu dikdörtgen şekle sahip kalıpların en uygun şekilde, sabit genişlik değişken uzunluklu bir platforma dizmişlerdir. İki bölümden oluşan çalışmalarının ilk bölümünde, parçaların dizilim sırasını ve dönme açılarını (0-90^o) GA ile optimize etmişlerdir. Daha sonra Sol-Alt yaklaşımı ile parçaları geliş sırasıyla dizmişlerdir.

Rocha vd. (2015), yaptıkları çalışmada yerleşim problemi için yeni bir yöntem geliştirmişlerdir. Problemi iki fazlı olarak çözmeyi amaçlamışlardır. Öncelikli olarak dizilim işlemine tabi tutulacak olan parçaları el ile boyutlarına göre büyük parçalar ve küçük parçalar olmak üzere iki gruba ayırmışlardır. Daha sonra ilk fazda büyük parçaları yerleşim alanının sınırlarından başlayarak merkeze doğru dizmişlerdir. İkinci fazda ise küçük gruptaki parçaları merkez etrafında oluşan boşluklara dizip, çakışmaları tamamen ortadan kaldırıncaya kadar küçük parçalar üzerinde döndürme işlemi uygulamışlardır. Herhangi bir dönme açısı kısıtlaması olmadan en uygun dönme açısını tespit ederek dizilim işlemini tamamlamışlardır.

Sato vd. (2015), yaptıkları çalışmada yerleşim probleminde çakışmaları azaltmayı amaçlamışladır. Piksel tabanlı olarak çalışmışlar ve çakışma miktarlarını azaltmak için çok boyutlu çözünürlük ile işlem yapmışlardır. İlk olarak düşük çözünürlükte kabaca yerleşecek bölgeyi tespit etmişler, daha sonra yerel bölgede yüksek çözünürlükle çakışmayı azaltacak şekilde hassas yerleşim işlemi yapmışladır. Distance Transform (DT) yöntemi ile parçaların sınırlarından merkezine doğru piksel değerleri artan matrisler elde etmişlerdir. Çakışan parçaların, çakıştığı bölgedeki piksel değerlerini toplayarak maliyet elde etmişlerdir. Maliyeti minimize edecek şekilde sonuca ulaşmışlardır. Kullandıkları yöntem çözünürlük artıkça yavaşlamaktadır ancak DT yöntemi ile hızlanma kat edilmiştir.

Valvo (2017), yaptığı çalışmada, yedi farklı meta sezgisel yöntem kullanarak, dikdörtgen şekilleri, sabit genişlik sonsuz uzunluklu bir şerit materyale uzunluğu minimum yapacak şekilde dizmiştir. Sol-Alt yaklaşımını ve NFP yöntemini kullanarak, en uygun yerleşim sırasını optimize etmiştir. Literatürde kullanılan farklı veri setleri ile yöntemleri test etmiştir.

Abeysooriya vd. (2018) yaptıkları çalışmada Jostle sezgisel algoritması ile iki boyutlu paketleme problemini çözmüşlerdir. İlk olarak sabit genişlik değişken uzunluklu bir şerit üzerine optimum dizmeyi amaçlamışlardır. Daha sonra eni ve boyu minimize etmeyi amaçlayan probleme uyarlamışladır. Kalıpların dizilim sıralarını optimize ederek işlemlerini gerçekleştirmişlerdir.

Paquay vd. (2018), üç boyutlu konteyner yükleme problemini ele almışlardır. Kübik şekle sahip yükleme nesnelerini çeşitli şekil be boyutlarda ki konteynerlere en optimum şekilde yerleştirmeyi amaçlamışlardır. İki fazlı bir sezgisel yaklaşım kullanmışlardır.

Polyakovskiy ve Hallah (2018) yaptıkları çalışmada, teslim tarihi kısıtını da dikkate alarak iki boyutlu yerleşim optimizasyonu işlemini gerçekleştirmişlerdir. Mixed-Integer Programming ve Constraint Programming yöntemlerini içeren bir hibrit iki fazlı yaklaşım önermişlerdir.

3. MATERYAL VE YÖNTEM

Optimizasyon gerektiren problemlerin çözümünde kullanımı basit olmasından dolayı MATLAB programı tercih edilmektedir. Eğer optimizasyon problemi resim içeren, matrislerle işlem yapılmasını gerektiren bir problemse MATLAB programı çok daha cazip hale gelmektedir. Ancak geliştirilen bir yazılımın ticari değerinin olması, kullanıcı arayüz tasarımı gerektirmesi gibi nedenlerden dolayı yazılımın son halinin MATLAB programından başka bir dil ile (C#, java, C++ gibi) yazılması neredeyse zorunludur. Çünkü MATLAB programı kısıtlı bir arayüz tasarım aracına sahiptir. Ayrıca MATLAB programı ile geliştirilen yazılımların çalıştırılması için gerekli gereksinimler diğer diller ile geliştirilen yazılımlara nispeten bir hayli fazladır. Diğer diller için gerekli altyapılar (.NET için framework, java için JVM vb.) zaten günümüz bilgisayarlarının hemen hemen tamamında hazırdır. Hiçbir ek araca gerek duyulmadan rahatlıkla istenilen bilgisayarlarda kullanılabilmektedir.

Yukarıda sayılan nedenlerle, MATLAB yerine Microsoft Visual Studio platformu ve C# programlama dili tercih edilmiştir. Böylece geliştirilen yazılım rahat bir kullanıcı arayüzüne sahip, hızlı çalışabilen bir yazılıma dönüştürülmüştür.

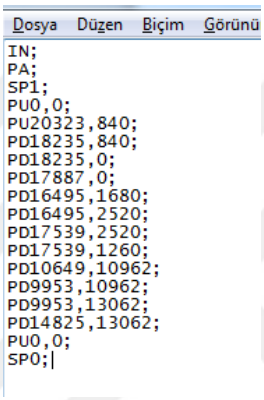
3.1. Çizici Cihazı Yazılımının Hazırlanması

Süleyman Demirel Üniversitesi Bilimsel Araştırma Projeleri koordinasyon birimi tarafından 4252-D2-15 nolu proje ile tez desteklenmiş ve üç eksen çizim yapacak olan makine tedarik edilmiştir (Şekil 1.2).

Şekil 1.2'de gösterilen makine Corel Draw, Photoshop, ArtCut gibi popüler çizim programları ile doğrudan çalışabilmektedir. En fazla tercih edilen program ArtCut programıdır. Bu program ile makine, kolay bir şekilde haberleşebilmekte ve çizim, kesim işlemleri yapılabilmektedir. Gerekli araştırmalar yapıldığında kesim işlemlerinde kullanılan CNC tezgâhlarının vektörel olarak temsil edilen dosyalarla çalıştığı görülmüştür. Vektörel tabanlı çizim programları farklı uzantılarda dosya üretebilmektedir ve en fazla tercih edilenleri *.dxf ve *.plt uzantılı dosyalardır. ArtCut programı ile herhangi bir *.BMP uzantılı dosya okunabilmekte ve "Image Tracing"

özelliği ile resim vektörel formata dönüştürülebilmektedir. Tez kapsamında geliştirilen yazılım, her iki dosya türünde de çıktı verecek şekilde geliştirilmiştir.

Satın alınan ve tezde kullanılacak olan çizici cihazı Hewlett-Packard Graphics Language (HPGL) dili ile haberleşmektedir. HPGL ise *.plt uzantılı dosyaları kabul etmektedir. Ayrıca makine seri port ve USB üzerinden haberleşebilmektedir. Geliştirilen yazılım seri port ile haberleşecek şekilde tasarlanmıştır. Gerekli bağlantı ayarlarının yapılmasından sonra elde edilen çizimler seri port ile doğrudan HPGL dili ile çizici cihazına gönderilmektedir. Şekil 3.1'de örnek bir çizim için HPGL dili komutları gösterilmiştir.



Şekil 3.1. Örnek Bir Yerleşim için HPGL Komut Dosyası

Geliştirilen yazılım, kullanılacak olan makinelerin özelliğine göre *.plt ya da *.dxf olarak elde edilen çizimleri çıktı olarak verebilmektedir. Geliştirilen yazılımın kullanım detayları ile ilgili bilgiler ilerleyen başlıklarda verilecektir.

3.2. Kullanılan Optimizasyon Yöntemleri

Bu tezde dokuz adet farklı optimizasyon (YAK, TB, GA, PSO, SÖA, GKO, DG, L-SHADE, HPSGKO) yöntemi kullanılmıştır. Her bir yöntem probleme uyarlanmış, geliştirilen yazılıma seçenek olarak eklenmiştir. Her bir yöntem detaylı olarak ilerleyen başlıklarda anlatılmıştır.

3.2.1. Yapay arı kolonisi

YAK, bal arılarının doğadaki besin arayışları sırasında göstermiş oldukları zeki davranışlarından esinlenilerek geliştirilmiş sürü tabanlı bir optimizasyon yöntemidir

(Koç, 2016.; Özcan, 2016.; Karaboga ve Basturk, 2007). Bal arıları mevcut yiyecek kaynaklarının azalması ya da tükenmesiyle birlikte yeni yiyecek kaynakları arayışına girerler. YAK, arıların bu yiyecek ararken ki durumlarını, bazı şartlarla birleştirerek optimizasyon problemlerinin çözümünde kullanılır. Bu optimizasyon yönteminde üç farklı arı türü vardır; işçi arılar, kâşif arılar ve gözcü arılar.

İşçi arılar, yiyecek kaynağından kovana nektar taşımakla görevli arılardır. Aynı zamanda işçi arılar, yiyecek kaynağı hakkında bilgileri diğer arılara aktarma görevini de yerine getirmektedirler. Kaynağın konumu ve yiyecek miktarı hakkında bilgileri işçi arılar iletmektedir. Bu bilgiyi işçi arılar kovan içinde yaptıkları dans ile diğer arılara aktarmaktadırlar. Görevi biten işçi arılar kâşif arı olarak görevlerine devam ederler.

Kâşif arılar, arama uzayı içerisinde rasgele olarak yiyecek kaynağı aramakla görevli arılardır. Buldukları yiyecek kaynağını diğer arılara bildirmekle görevlidirler. Kâşif arılar eğer yiyecek kaynağı bulurlarsa, buldukları kaynağın işçi arısı olarak görev yapmaya başlarlar.

Gözcü arılar ise kovanda bekleyerek, kovana gelen bilgiler doğrultusunda hareket etmekle görevlidirler. Yiyecek kaynakları bilgilerinden yola çıkarak nektarı fazla olan kaynağa doğru hareket etmekle görevlidirler.

YAK algoritmasına göre, toplam yiyecek kaynağı sayısı ile toplam görevli arı sayısı birbirine eşittir. Ayrıca işçi arı sayısı ile gözcü arıların sayısı da birbirine eşit olarak kabul edilmektedir. YAK algoritmasında başlangıç popülasyonu Denklem 3.1'de gösterildiği gibi oluşturulmaktadır.

$$P_{i,j} = lb_j + rand(0,1) \times (ub_j - lb_j) \tag{3.1}$$

Burada; j, değişken sayısını, i, yiyecek kaynağı sayısını göstermektedir. lb_j ve ub_j ise j. Değişkenin alt ve üst sınırlarını ifade etmektedir. Son olarak P ise popülasyonu temsil etmektedir.

İşçi arılar sürekli olarak komşuluklarında daha iyi bir uygunluk değerine sahip noktanın olup olmadığını arar. Önceki en uygun sonuç sürekli hafızada tutulur. Denklem 3.2, aday komşu noktaların bulunmasını ifade etmektedir.

$$C_{i,j} = P_{i,j} + rand(-1,1) \times (P_{i,j} - P_{k,j}) \tag{3.2}$$

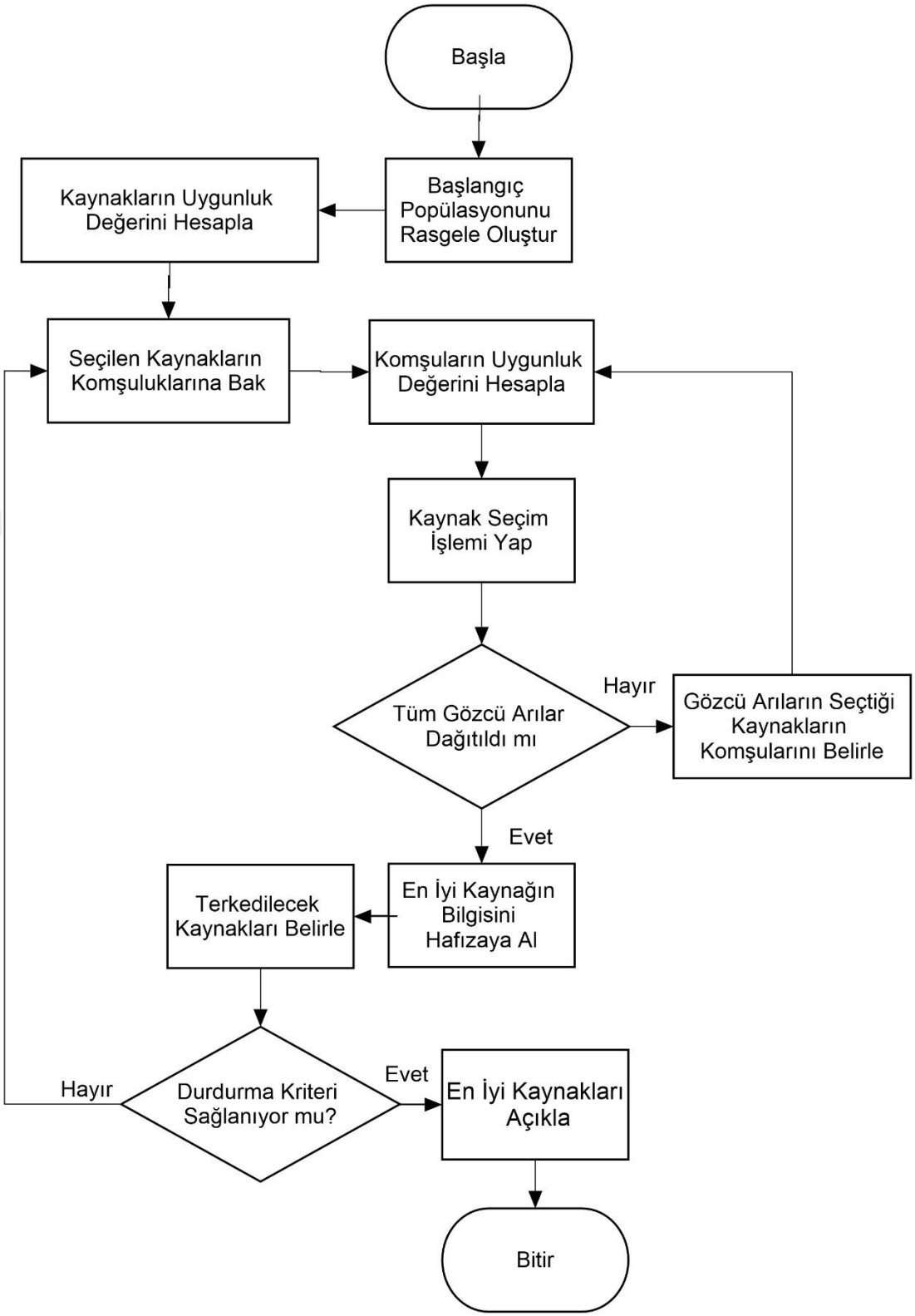
Burada; *C* komşu aday noktaları, *k* ise *k.* aday çözümü temsil etmektedir. Eğer komşuluklarda daha iyi bir nokta bulunursa işçi arılar kovanda dans ederek diğer arılara bilgi aktarımı yapmaktadır.

Gözcü arılar, yiyecek kaynaklarının uygunluk değerine göre yiyecek kaynağı seçimi işlemini yaparlar. Seçim işlemini rulet tekerleği yöntemine göre yaparlar. Uygunluk değeri bilinen bir yiyecek kaynağının seçilmesi Denklem 3.3 ve 3.4'te gösterildiği gibi hesaplanır.

$$Fit_i = \begin{cases} \dfrac{1}{f_i} & f_i \geq 0 \\ 1 + abs(f_i) & f_i < 0 \end{cases} \tag{3.3}$$

$$Prob_i = \frac{Fit_i}{\sum_{i=1}^{FN} Fit_i} \tag{3.4}$$

Burada; *f,* uygunluk fonksiyonunu, Fit_i, *i.* kaynağın uygunluk değerini (besin miktarını), *FN*, toplam besin kaynağı sayısını ve $Prob_i$ ise *i.* besin kaynağının seçilme olasılığını ifade etmektedir. Denklem 3.3 ve 3.4'ten anlaşıldığı üzere besin miktarı fazla olan kaynağın seçilme ihtimali daha yüksektir. Eğer zamanla besin kaynağında ki nektar miktarı artmıyor ve azalıyorsa o zaman bu kaynak terk edilecek ve o kaynakta görevli işçi arılar kâşif arı konumuna geçeceklerdir. YAK algoritmasının akış diyagramı Şekil 3.2'de verilmiştir.



Şekil 3.2. YAK algoritması akış diyagramı

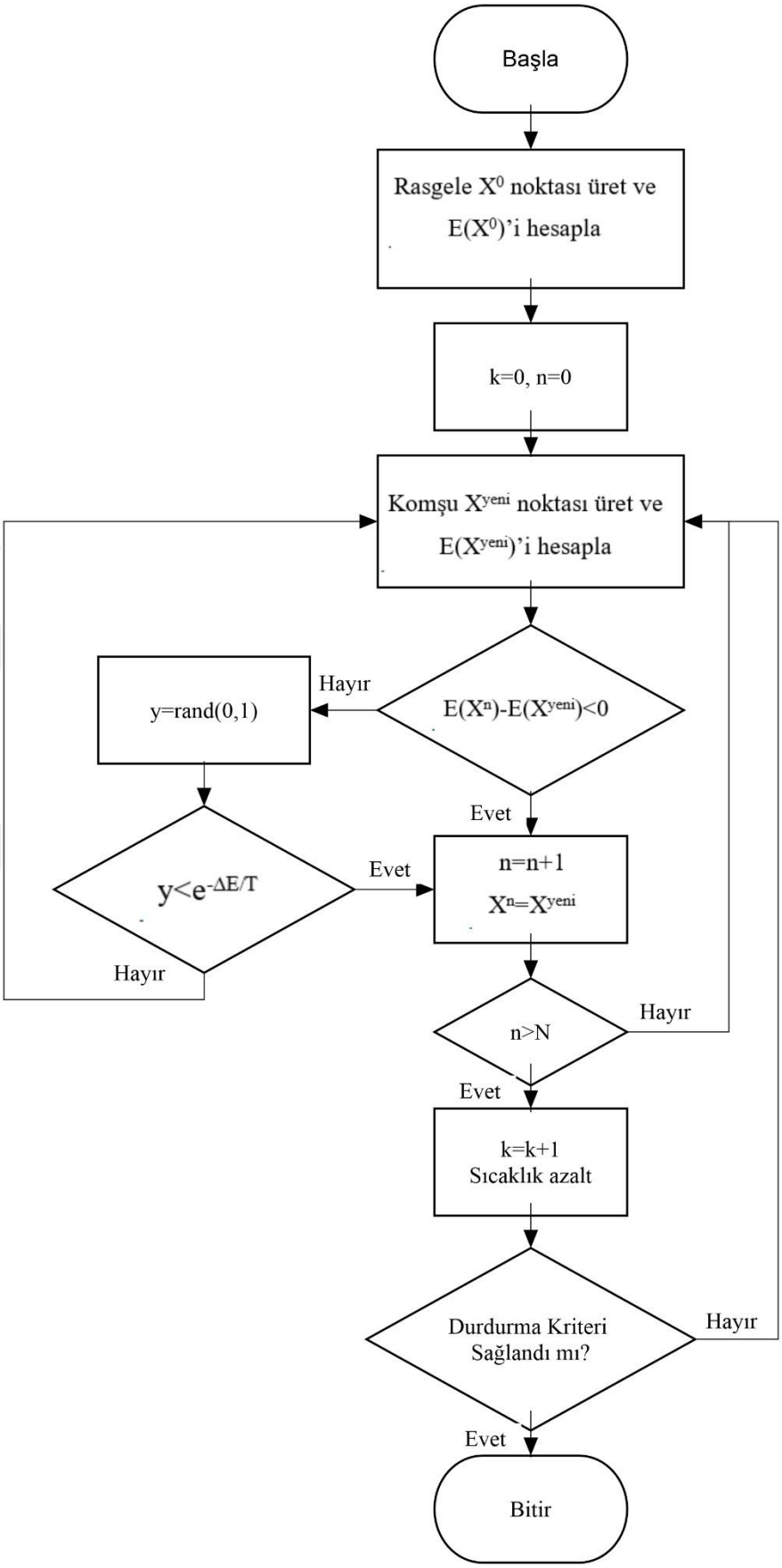
3.2.2. Tavlama benzetimi

Tavlama işlemi, bir katının ısı banyosunda düşük enerjili durumlarının elde edilebilmesi için geçirilen süreç olarak adlandırılmaktadır. Bir katı erime sıcaklığına kadar ısıtılıp daha sonra düşük enerjili durum elde edilinceye kadar sıcaklığının kontrollü bir şekilde düşürülmesi ile süreç gerçekleşir (Kirkpatrick vd., 1987; Yiğit, 2004). Bu süreçte, katının yapısal özelliğinde soğutma hızına bağlı olarak farklılaşma meydana gelmektedir. Isıtma işleminde katının kristal yapısı sıvı faza göre kendi ayarlamaktadır. Soğutma işlemi eğer çok uygun şekilde yavaşça yapılırsa, süreç sonunda katının kristal yapısı çok düzenli bir hal alır.

TB algoritması metropolis algoritmasını kullanır. Bu algoritmaya göre sürekli iyi çözümler seçilerek ilerleme kaydedilir. Ancak bazı durumlarda kötü çözümlerde seçime dâhil edilerek yerel minimumlara takılmalar önlenmekte ve daha iyi çözümlere doğru ilerlenebilmektedir. Başlangıçta çözüm uzayında rasgele bir noktadan başlanarak uygunluk değeri yani enerjisi hesaplanır. Üretilen bu noktanın komşuluklarına bakılarak daha iyi bir uygunluk değerine sahip nokta varsa güncelleme yapılır. Daha kötü komşular belli bir olasılığa göre tercih edilebilmektedir. Bu olasılık değeri için Boltzmann sabiti kullanılmaktadır. Denklem 3.5'te Boltzmann sabiti hesaplaması verilmiştir.

$$B = e^{-\Delta E/T} \tag{3.5}$$

Burada; B, Boltzmann sabitini, ΔE, mevcut nokta ile komşu noktanın uygunluk değerleri farkını ve T ise mevcut sıcaklık değerini ifade etmektedir.

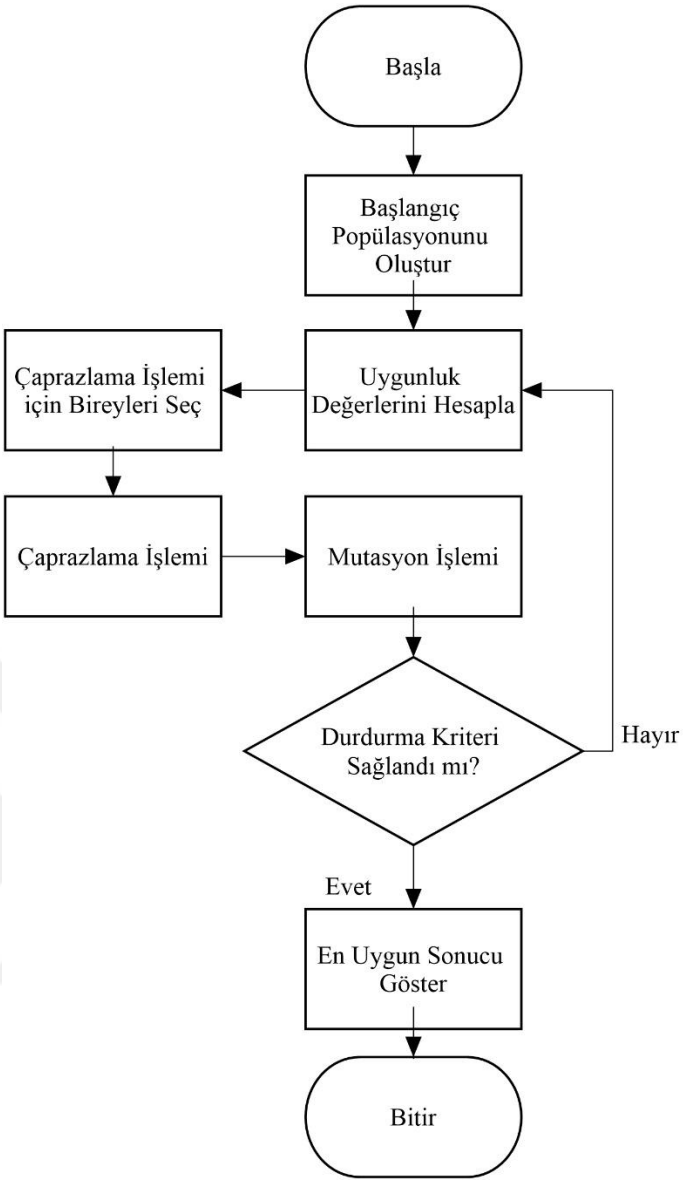


Şekil 3.3. TB algoritması akış diyagramı

Her sıcaklık değeri belli bir iterasyon sayısı kadar işletilir. Daha sonra sıcaklık değeri düşürülerek bir sonraki iterasyona geçilir. Durdurma kriteri sağlanıncaya kadar tüm bu adımlar devam ettirilir. TB algoritmasının akış diyagramı Şekil 3.3'te verilmiştir.

3.2.3. Genetik algoritma

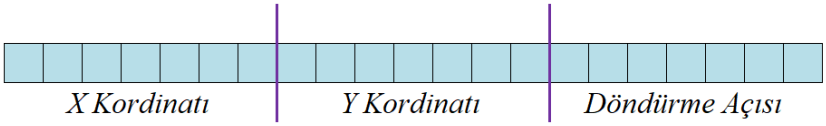
GA, genetik ve evrim teorisi birleştirilerek oluşturulmuş bir optimizasyon yöntemidir. Holland, yayınladığı "Doğal ve Yapay Sistemlerin Uygulanması" adlı kitabında GA'yı duyurmuştur (Holland, 1975). GA'da her bir canlı kromozomlar olarak temsil edilirler. Her bir kromozom bir ya da daha fazla optimize edilmek istenen değişkenlerin birleşimini ifade etmektedir. Başlangıçta belirli bir sayıda popülasyon oluşturulur ve her bir kromozomun uygunluk değeri hesaplanır. Kromozomların uygunluk değerlerine göre seçilim, çaprazlama ve mutasyon gibi genetik işlemler sonucunda yeni popülasyonlar oluşturulur. İstenilen bir durdurma kriteri sağlanıncaya kadar yeni popülasyonlar oluşturulmaya devam edilir. Şekil 3.4'te GA akış diyagramı verilmiştir.



Şekil 3.4. GA akış diyagramı

GA'da amaç, popülasyondaki yeterince iyi olmayan bireylerin elenmesi ve iyi bireylerin hayatlarını devam ettirmesidir. Böylece, zamanla iyi bireyler kendi arasından daha iyi yeni bireyler oluşturarak her nesilde bir iyileşme kat edilecektir.

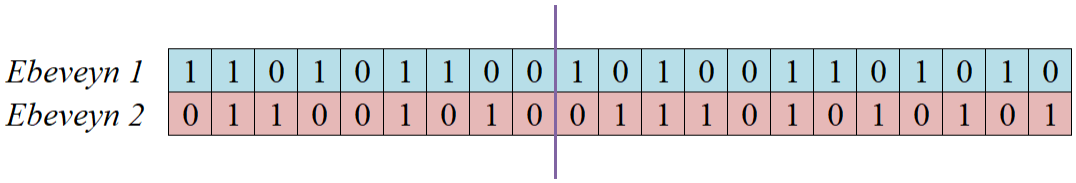
Kodlama işlemi: Kromozomlar farklı şekillerde ifade edilebilmektedir. İkili kodlama, değer kodlama, permütasyon kodlama gibi farklı çeşitler mevcuttur. İkili kodlama işlemi tezde tercih edilmiştir. Bu tezde optimize edilmek istenen problem üç parametrelidir. Bunlar; X, Y koordinat bilgisi ve döndürülecek açık bilgileridir. Tüm bilgiler yedi bit ile kodlanmaktadır. Şekil 3.5'te kromozom yapısı gösterilmiştir.



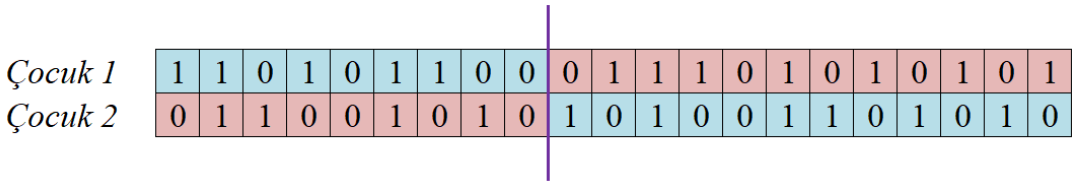
Şekil 3.5. Kromozom yapısı

Seçilim işlemi: GA'da yeni nesillerin oluşturulması için mevcut nesilden çaprazlama ve mutasyon işlemlerine tabi tutulacak bireylerin seçilmesi gerekmektedir. İyi olan bireylerin bir sonraki nesile aktarılması ve böylece oluşacak yeni bireylerinde iyi bireyler olması amaçlanmaktadır. GA'da birçok seçim işlemi kullanılmaktadır. Turnuva seçimi, rulet tekerleği seçimi, sıralama seçimi gibi farklı yöntemler bulunmaktadır. Bu tez çalışmasında turnuva seçimi yöntemi kullanılmıştır. Turnuva seçimi işleminde, popülasyondan iki adet birey rasgele seçilir. Uygunluk değeri yüksek olan birey turnuvanın kazananı olur ve seçilmiş olur.

Çaprazlama işlemi: Çaprazlama işlemi seçilen iki bireyin yani kromozomun rasgele bir ya da birden fazla noktadan karşılıklı olarak çaprazlanmasıdır. Bu tezde tek noktadan çaprazlama işlemi kullanılmıştır. Şekil 3.6'da örnek iki bireyin rasgele olarak çaprazlanması ve oluşan yeni bireyler gösterilmiştir.



(a)



(b)

Şekil 3.6. Çaprazlama işlemi (a) Çaprazlama işleminden önce ebeveynler (b) Çaprazlama işleminden sonra oluşan yeni bireyler

Mutasyon işlemi: Mutasyon işlemi, popülasyondaki bireylerin yeni nesile aktarılırken kromozom yapısında meydana gelen değişiklik olarak adlandırılmaktadır. İkili kodlama da mutasyona uğrayan genin tümleyeni alınarak işlem gerçekleştirilir. Yani 0 olan genin 1'e dönüşmesi, 1 olan genin ise 0'a dönüşmesidir. Mutasyon doğada da

çok küçük olasılıklarla meydana gelmektedir. Şekil 3.7'de örnek bir kromozom yapısındaki mutasyon işlemi gösterilmiştir.

Mutasyonsuz Birey	0	1	1	0	0	1	0	1	0	0	1	1	1	0	1	0	1	0	1	0	1
Mutasyonlu Birey	0	1	1	0	0	1	0	1	0	1	1	1	1	0	1	0	1	0	1	0	1

Şekil 3.7. Mutasyon işlemi

3.2.4. Parçacık sürü optimizasyonu

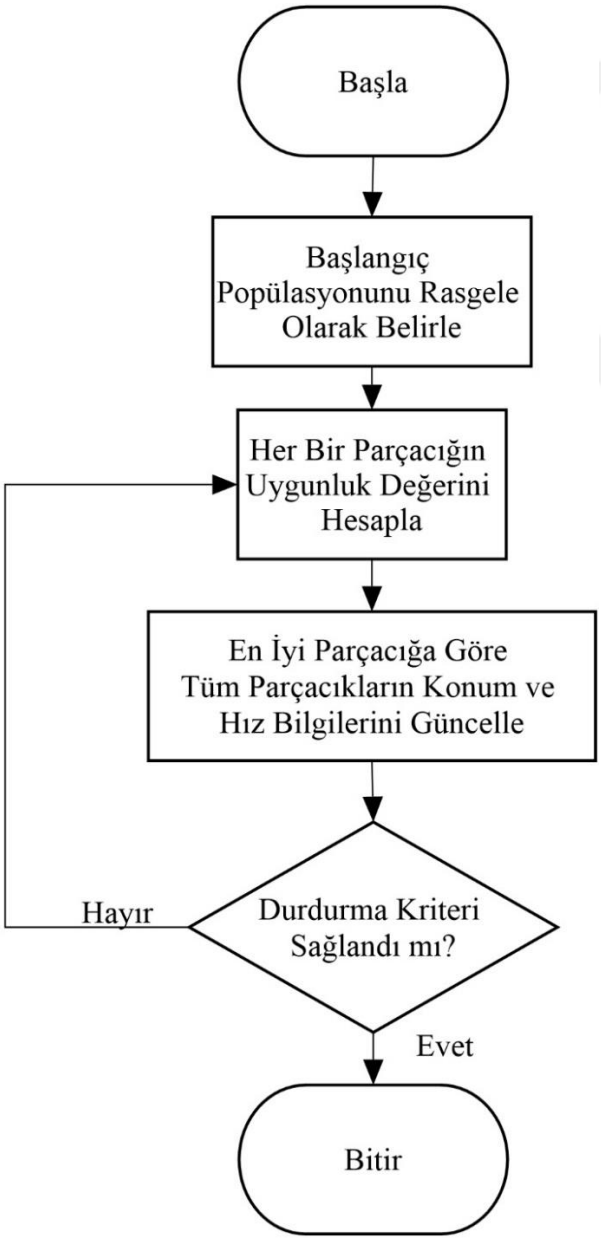
PSO, doğadaki kuş ve balık sürülerinin avlanma stratejilerinden esinlenilerek geliştirilmiş bir optimizasyon yöntemidir (Kennedy ve Eberhart, 1995). PSO algoritmasında her bir kuş ya da balık parçacık olarak ifade edilmektedir. Her bir parçacığın kendine ait bilgileri vardır. Bu bilgiler; parçacığın pozisyon değeri, hız değeri, kendi en uygun değeri ve sürünün en uygun değeridir. Her bir parçacık problemin boyutu olan *d* boyutlu bir başlangıç konum ve hız vektörü ile arama işlemine başlamaktadır. PSO'da her bir parçacık en iyi konuma sahip olan parçacığa doğru hareket etmektedir. Bu nedenle her bir parçacığın bilgilerinin tüm sürü içinde paylaşıldığı varsayılmaktadır (Barak, 2016; Durak, 2016.) Bilgi paylaşımı iki şekilde yapılmaktadır. Bunlar; en iyi konumu bulan parçacık (gbest) bilgilerinin, tüm sürü ile paylaşılması ve yerel olarak en iyi konumda bulunan parçacık (lbest) bilgilerinin, sadece yakınlarındaki parçacıklarla paylaşılmasıdır. PSO'da parçacıkların en iyi konumdaki parçacığa göre hareket etmeleri Denklem 3.6 ve 3.7'de verilmiştir.

$$\bar{v}_{n+1}^{i} = w\,\bar{v}_{n}^{i} + c_1 r_1\left(\bar{p}_{n}^{i} - \bar{x}_{n}^{i}\right) + c_2 r_2\left(\bar{p}_{n}^{g} - \bar{x}_{n}^{i}\right) \tag{3.6}$$

$$\bar{x}_{n+1}^{i} = \bar{x}_{n}^{i} + \bar{v}_{n+1}^{i} \tag{3.7}$$

Burada; *i*, i. parçacığı, *n*, şu anki iterasyon sayısını, r_1 ve r_2, [0, 1] arasında rasgele sayıları, c_1 ve c_2, PSO algoritmasındaki öğrenme faktörü sabit sayılarını ve *w* ise durgunluk katsayısını ifade etmektedir. Bu değerlerin küçük olması hassas adımlarla ilerlemeyi sağlamakla birlikte öğrenme sürecini uzatmaktadır. Bununla birlikte büyük seçilmesi ise öğrenme süresini kısaltmakta ancak bazı durumlarda sürüden kopmalara neden olmaktadır. Bu nedenle c_1 ve c_2 değerlerinin [0, 2] arasında seçilmesi uygun sonuçlar vermektedir. Ayrıca *x*, konum vektörünü, *v* ise hız vektörünü ifade etmektedir (Rezai, 2017).

PSO algoritmasında, sürü içinde sadece bir tane en iyi konumda olan parçacık bulunmaktadır. Tüm parçacıklar, en iyi parçacığa doğru hareket etmektedirler. Eninde sonunda tüm parçacıkların en iyi parçacığa ulaşacağı kabul edilmektedir. Her bir parçacığın hızı için bazı sınırlamalar getirilmiştir. Aksi halde parçacıklar arama uzayı dışına çıkabilmektedir. Parçacıkların hızları arama uzayı sınırları içinde olacak şekilde sınırlandırılarak, parçacıkların herhangi bir anda arama uzayı dışına çıkmasının önüne geçilmektedir. Şekil 3.8'de PSO algoritması için akış diyagramı verilmiştir.



Şekil 3.8. PSO algoritması akış diyagramı

3.2.5. Sosyal örümcek algoritması

SÖA, bir ağ üzerindeki örümceklerin avlanma stratejilerinden esinlenilerek geliştirişmiş bir meta sezgisel optimizasyon yöntemidir (Yu ve Li, 2015). SÖA algoritmasında, arama uzayı çok boyutlu bir örümcek ağı gibi düşünülmektedir. Örümcek ağı aynı zamanda örümcekler arası haberleşmeyi sağlayan titreşimler için bir iletim ortamı görevi görmektedir. Örümceklerin her biri ajan olarak isimlendirilir ve hepsi anlık olarak bulunduğu konumu, uygunluk değerini ve titreşim bilgilerini hafızada tutarlar.

Örümcekler, ağ üzerinde meydana gelen titreşimlerin yönünü ve şiddetini çok iyi tanımlayabilmektelerdir. Ağ üzerindeki her örümcek, konum değiştirdiği zaman, yeni konumu ve uygunluk değeri bilgilerini, titreşimler vasıtasıyla ağ üzerindeki diğer örümcekler arasında paylaşırlar. Sosyal bir bilgi paylaşım platformunu andıran bu durumdan dolayı algoritma Sosyal Örümcek Algoritması olarak isimlendirilmiştir.

SÖA algoritmasının en önemli bölümü, titreşimlerin olduğu bölümdür. Titreşimler, bulunan yiyecek kaynağının yönünü ve uzaklığını ifade etmektedir. Titreşimin şiddeti, kaynaktan uzaklaştıkça giderek azalmaktadır. Ağ üzerindeki her örümcek, her hareketi sonucunda elde ettiği yeni bilgileri titreşimler ile ağ üzerinde paylaşırlar. Titreşimlerin şiddeti $[0, +\infty)$ aralığında değişmektedir ve Denklem 3.8'de gösterildiği gibi hesaplanmaktadır.

$$I(s) = \log\left(\frac{1}{f(s) - C} + 1\right) \tag{3.8}$$

Burada, $I(s)$ titreşimin şiddetini, $f(s)$ uygunluk fonksiyonunu, C arama uzayının alabileceği minimum değeri ifade etmektedir (Yu ve Li, 2014).

Titreşimler, mesafe arttıkça gücünü kaybetmektedir. SÖA algoritmasında, iterasyon adımı ilerledikçe veya uzaklık arttıkça titreşim şiddeti azaltılabilmektedir. Denklem 3.9'da iterasyon arttıkça titreşim azaltma formülü verilmiştir.

$$I(t+1) = I(t) * r_a \tag{3.9}$$

Burada t, iterasyon sayısını, r_a ise kullanıcı tanımlı titreşim azaltma katsayısını ifade etmektedir. Denklem 3.10'da ise, mesafe arttıkça titreşim şiddetinin azalma formülü verilmiştir.

$$I(p) = I(s) * \exp(-\frac{D(s,p)}{\sigma * r_a}) \tag{3.10}$$

Burada, s ve p sırasıyla kaynağın pozisyonunu ve titreşimin alındığı pozisyonu ifade etmektedir. $I(s)$, s pozisyonundaki titreşim şiddetini, $D(s,p)$, s ve p pozisyonları arasındaki mesafeyi ve son olarak σ ise, ağ üzerindeki örümceklerin birbirlerine olan uzaklıklarının standart sapmasını temsil etmektedir. Bu tez çalışmasında Denklem 3.10 kullanılarak titreşim azaltma işlemi gerçekleştirilmiştir (Yu ve Li, 2014).

SÖA algoritması üç fazdan oluşmaktadır. Bunlar; başlatma, iterasyon ve sonlandırma fazlarıdır. Başlatma fazında, uygunluk fonksiyonu, arama uzayı ve optimizasyon algoritması parametreleri tanımlanmaktadır. Daha sonra, ağ üzerindeki örümcek popülasyonunun her bir bireyi (örümcek) rasgele olarak konumlandırılmaktadır. Her bir örümceğin başlangıç konumu rasgele belirlenmektedir ve başlangıç titreşim bilgileri sıfır (0) olarak atanmaktadır. Başlatma fazı bu işlemlerin ardından tamamlanmış olmaktadır.

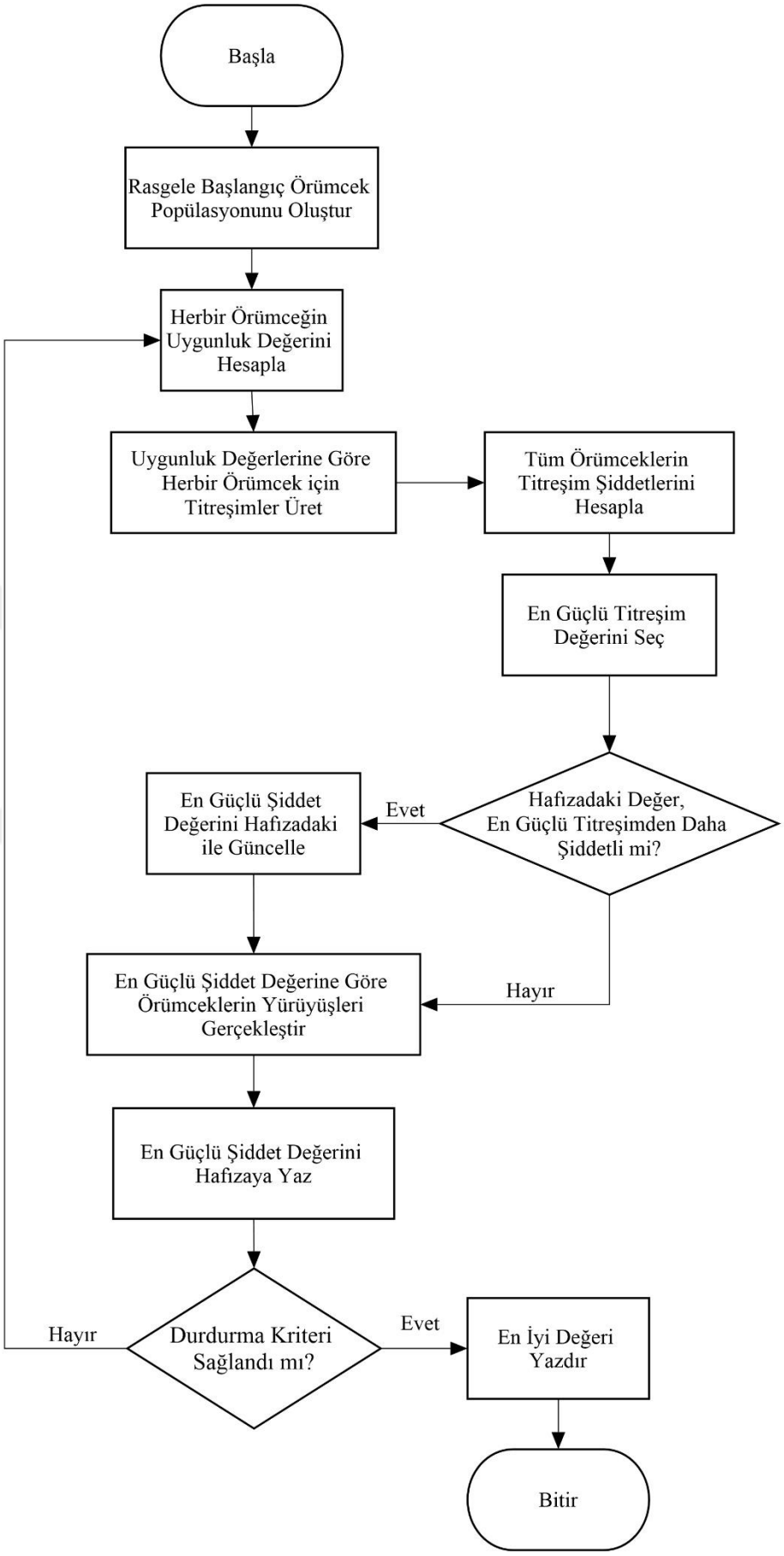
Başlatma fazının tamamlanmasının ardından iterasyon fazına geçilmektedir. İterasyon fazı gerçekleştirilirken, örümceklerin konumlarının her değişmesinden sonra yeni konumlarına göre uygunluk değerleri hesaplanmaktadır. Daha sonra her bir örümcek yeni uygunluk değerini ve konum bilgisini titreşim olarak ağ ile paylaşmaktadır. Ağ üzerindeki tüm örümcekler farklı yön ve şiddette olan tüm titreşimleri alarak değerlendirirler. En büyük şiddete sahip olan titreşimi (V_{best}), önceki en büyük titreşimle (V_{prev}) karşılaştırırlar. Eğer daha güçlü bir titreşim algılamışlarsa, hafızadaki titreşim bilgisini güncellerler. Daha sonra Denklem 3.11 kullanılarak tüm örümceklerin konumları güncellenir.

$$P_s(t+1) = P_s(t) + (P_{best}(t) - P_s(t)) \odot (1 - R \odot R) \quad (3.11)$$

Burada $P_s(t)$, t. iterasyonda popülasyondaki s. örümceğin konumunu ifade etmektedir. $\odot$ işlemi matris ve vektörlerin birebir çarpımını temsil etmektedir. P_{best}, en iyi titreşim kaynağının pozisyonunu, R ise [0-1] aralığında düzenli dağılıma sahip rasgele oluşturulmuş bir vektörü ifade etmektedir. Bu vektör problem boyutuna eşit uzunluktadır. Aynı zamanda Denklem 3.11'de görülen "1" değeri de yine problem boyutuna eşit uzunlukta "1"'lerden oluşan vektörü ifade etmektedir. Örümceklerin yeni konumları belirlendikten sonra bir sonraki iterasyona geçilmektedir.

İterasyon işlemleri durdurma kriteri sağlanıncaya kadar veya kullanıcı tarafından belirlenen bir iterasyon sayısına ulaşıncaya kadar devam eder. Durdurma kriteri ile sonlandırma fazı da tamamlanmış olmaktadır.

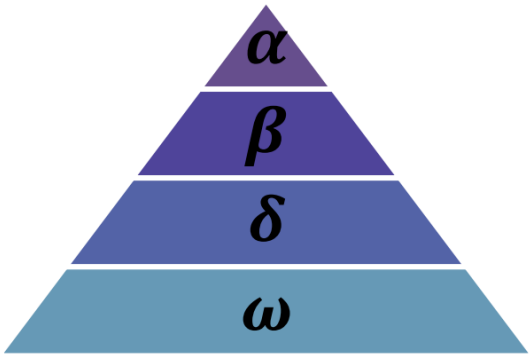
Şekil 3.9'da SÖA algoritması adımlarını gösteren akış diyagramı verilmiştir.



Şekil 3.9. SÖA algoritması akış diyagramı

3.2.6. Gri kurt optimizasyonu

GKO, gri kurtların doğadaki yaşam hiyerarşisinden esinlenilerek geliştirilmiş ve matematiksel olarak modellenmiş bir meta sezgisel optimizasyon algoritmasıdır (S. M. Mirjalili ve Lewis, 2014); (S. Mirjalili, 2015); (Song vd., 2015). Gri kurt hiyerarşisi en üstte Alfa kurtların bulunduğu ve sırasıyla Beta, Delta ve Omega kurtların olduğu bir hiyerarşiden oluşmaktadır (Şekil 3.10).



Şekil 3.10. GKO algoritması hiyerarşisi (S. M. Mirjalili ve Lewis, 2014)

Alfa kurtlar Şekil 3.10'dan görüldüğü üzere en baskın kurtlardır. Sürünün avlanması dinlenmesi gibi sürü ile ilgili önemli kararları Alfa kurtlar vermektedir. Beta kurtlar bu hiyerarşide Alfa kurtların yardımcıları olarak tanımlanır. Alfa kurdun emirlerini diğer kurtlara bildirme görevini yapmaktadırlar. Beta kurtlar, Alfa kurdun yerine geçecek kurtları temsil etmektedir. Omega kurtlar Alfa kurt tarafından seçilen en düşük seviyeli kurtlardır. Yemek işlemi sırasında her zaman en son yemek yiyen kurtlardır. Alfa, Beta veya Omega grubunda olmayan kurtlar Delta kurt olarak adlandırılırlar. Delta kurtlar Alfa ve Beta kurtlar tarafından seçilen ve Omega kurtlarına baskın olan kurtlardır (Tunç, 2016).

Gri kurtlar avlanırken üç aşamayı takip ederler. Bunlar;

- İzleme, takip ve ava yaklaşma
- Avı çevreleme ve av yorulup duruncaya kadar avı hareket ettirme
- Ava saldırma

GKO algoritmasında, Alfa kurt en iyi çözümü temsil etmektedir. Beta ve Delta kurtlar ikinci ve üçüncü en iyi çözümü temsil etmektedir. Son olarak Omega kurtlar ise aday çözümleri ifade etmektedirler.

GKO'da gri kurtlar avı çevrelerken Denklem 3.12 ve 3.13'u kullanmaktadırlar.

$$D = |C * X_p(t) - X(t)| \tag{3.12}$$

$$X(t+1) = X_p(t) - A * D \tag{3.13}$$

Burada, t anlık olarak iterasyon sayısını, X_p avın konumunu, X bir gri kurdun konum vektörünü tutmaktadır. A ve C vektör katsayılarını ifade etmektedirler ve Denklem 3.14 ve 3.15'de gösterildiği şekilde hesaplanmaktadırlar.

$$A = a * (2 * r_1 - 1) \tag{3.14}$$

$$C = 2 * r_2 \tag{3.15}$$

Burada, r_1 ve r_2 [0,1] arasında rasgele sayıyı ve a ise 2'den 0'a doğru iterasyon ilerledikçe doğrusal olarak azalan sayıyı ifade etmektedir.

GKO algoritmasında, arama işlemi başlangıçta rasgele olarak başlar. Daha sonra her bir kurdun uygunluk değeri maliyet fonksiyonuna göre hesaplanır. En iyi üç kurt sırasıyla, alfa, beta ve delta olarak adlandırılırlar. Avlanma işlemi alfa kurt tarafından koordine edilir ve bazen bu işleme beta ve delta kurtlar da katılabilir. En iyi üç kurt olan alfa, beta ve delta kurtların pozisyonları, Denklem 3.16, 3.17 ve 3.18'te gösterildiği gibi güncellenmektedir.

$$\begin{aligned} D_\alpha &= |C_1 * X_\alpha - X(t)| \\ D_\beta &= |C_2 * X_\beta - X(t)| \\ D_\delta &= |C_3 * X_\delta - X(t)| \end{aligned} \tag{3.16}$$

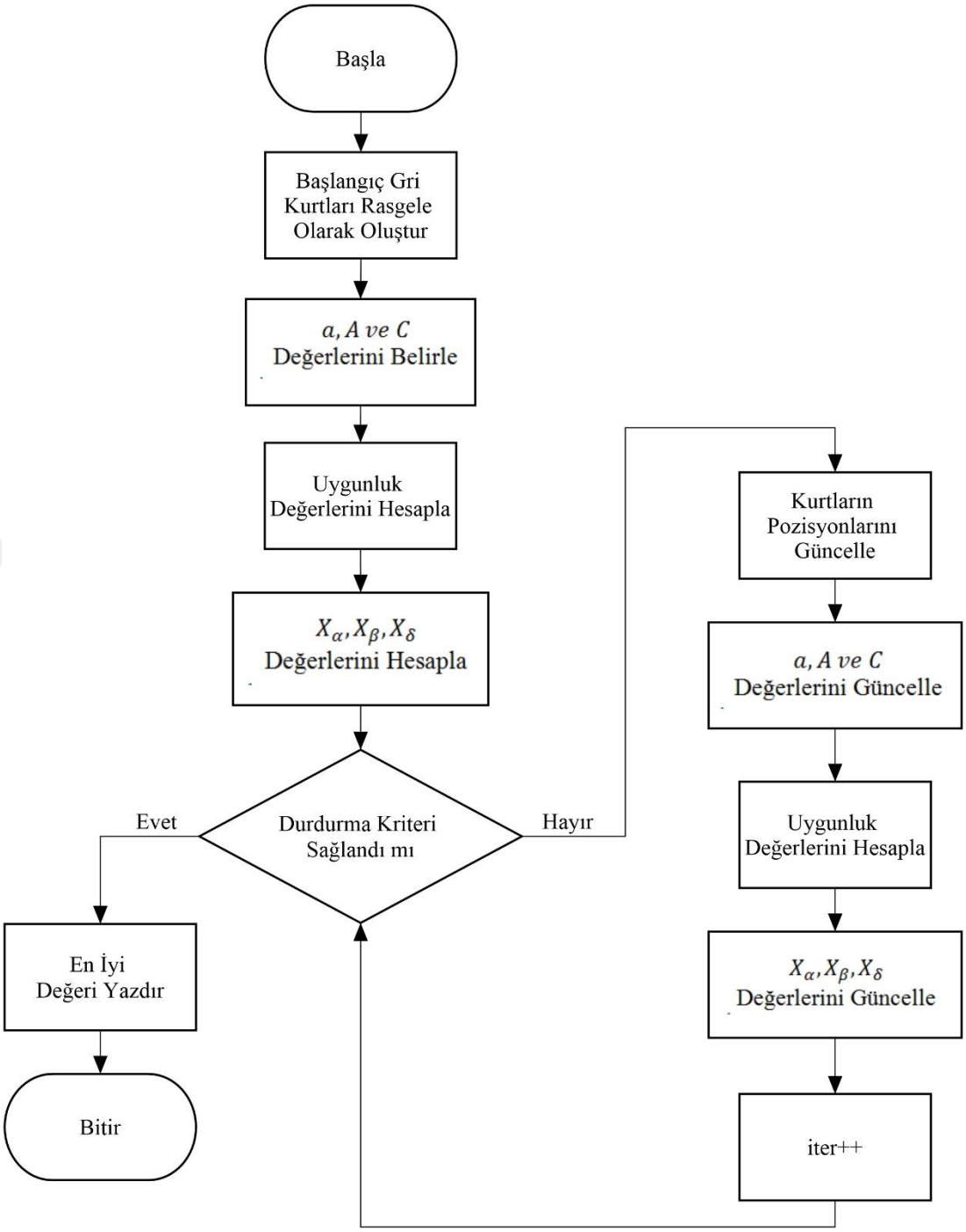
Burada, X_α, X_β ve X_δ sırasıyla alfa, beta ve delta kurtların pozisyonlarını temsil etmektedir. Her iterasyonda en iyi üç kurt sürekli güncellenmektedir.

$$X_1 = |X_\alpha - a_1 D_\alpha|$$
$$X_2 = |X_\beta - a_2 D_\beta| \quad (3.17)$$
$$X_3 = |X_\delta - a_3 D_\delta|$$

$$X_p(t+1) = \frac{X_1 + X_2 + X_3}{3} \quad (3.18)$$

$X_p(t+1)$, avın yeni konumunu ifade etmektedir.

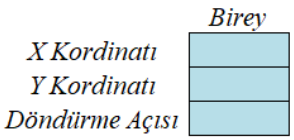
GKO'da avın yeri belirlendikten sonra son aşama olarak ava saldırı işlemi gerçekleştirilmektedir. Ava saldırı işlemi, av hareketini durdurduktan sonra gerçekleşmektedir. Matematiksel olarak modelleme işlemi dikkate alındığında, Denklem 3.14'te belirtilen *A* değerine göre saldırı işlemi gerçekleşmektedir. *A* değeri 2'den 0'a doğru r_1 rasgele değişkenine bağlı olarak azalmaktadır. Bu durumda *A* değişkeni $[-2a, 2a]$ aralığında değerler almaktadır. *A* değeri 1'den büyükse gri kurtlar avdan uzaklaşmakta ve daha uygun bir av aramaya başlamaktadırlar. Eğer 1'den küçükse gri kurtlar ava saldırmaya zorlanmaktadırlar. Bu işlem yerel minimumlara takılmaları engellemektedir. GKO'da avlanma işlemi durdurma kriteri sağlanıncaya kadar veya belirlenen iterasyon sayısına ulaşılıncaya kadar devam ettirilmektedir. Şekil 3.11'da GKO algoritması için akış diyagramı verilmiştir.



Şekil 3.11. GKO algoritması akış diyagramı

3.2.7. Diferansiyel gelişim algoritması

DG algoritması, Storn ve Price tarafından geliştirilmiş bir evrimsel optimizasyon algoritmasıdır (Storn ve Price, 1997). Başarılı sonuçları nedeniyle gerçek yaşam problemleri de dâhil olmak üzere bir çok optimizasyon problemine uyarlanabilmektedir. GA'da olduğu gibi çaprazlama, mutasyon ve seçme işlemleri, DG algoritmasında da vardır. DG algoritmasının en büyük avantajı, optimize edilmek istenen değişkenlerin gerçek değerlerinin tutulabildiği vektörler ile temsil edilmesidir. Şekil 3.12'de örnek bir bireyin temsili gösterilmiştir. DG algoritmasında üç çeşit birey bulunmaktadır. Bunlar; hedef (x) birey, mutasyonlu (v) birey ve test (u) bireydir. Popülasyon büyüklüğünün NP olduğu ve parametre sayısının D olduğu bir problem için, i. birey $x_i = (x_{1i}, x_{2i}, \dots, x_{Di})$ olarak gösterilir.



Şekil 3.12. DG algoritması bireylerin temsili

DG algoritmasının mutasyon stratejileri değişiklik göstermekte ve yöntemin başarısını doğrudan etkilemektedir. Bu nedenle DG algoritmasında her geçen gün yeni stratejiler geliştirilmekte ve sürekli olarak gelişim sürecini devam ettirmektedir. Standart DG algoritmasında mutasyon işlemi için, mevcut bireyin dışında birbirinden farklı en az üç bireye de daha ihtiyaç vardır. Bu nedenle NP en az dört değerini alabilmektedir.

Popülasyonu başlatma: Başlangıç popülasyonu, her bir parametre için belirlenen aralıklarda rasgele olarak oluşturulmaktadır. Denklem 3.19'da başlangıç popülasyonunun rasgele oluşturulması verilmiştir.

$$x_{j,i} = {x_j}^{min} + rand(0,1) \cdot ({x_j}^{max} - {x_j}^{min}) \tag{3.19}$$

Burada; $i = \{1, \dots, NP\}$, popülasyondaki bireyleri, $j = \{1, \dots, D\}$, problem boyutunu, ${x_j}^{min}$ ve ${x_j}^{max}$ değerleri sırasıyla, j. parametrenin minimum ve maksimum aralık değerlerini ifade etmektedir.

Mutasyon işlemi: Standart DG algoritmasında, mutasyon işlemi için birbirinden farklı üç adet bireye ihtiyaç vardır. Standart mutasyon stratejisi DE/rand/1 olarak isimlendirilmektedir ve Denklem 3.20'de gösterilmiştir.

$$v_i = x_{r_1} + F \times \left(x_{r_2} - x_{r_3}\right) \qquad r_1 \neq r_2 \neq r_3 \neq i \tag{3.20}$$

Burada; F değeri $[0, 2]$ aralığında seçilen bir katsayıyı (Storn ve Price, 1997), r_1, r_2 ve r_3 ise popülasyondan seçilen rasgele bireyleri ifade etmektedir. DG algoritması sürekli olarak gelişmeye devam etmekte ve başarısı geliştirilen mutasyon stratejilerine göre değişmektedir. Bu tez çalışmasında, en güncel olan bir DG algoritması mutasyon stratejisi kullanılmıştır. Mohamed ve Suganthan tarafından geliştirilen mutasyon stratejinde, popülasyondan oluşturulan üç farklı birey alt kümesi içinden, en iyi, iyi ve kötü bireyler belirlenmektedir (Mohamed ve Suganthan, 2017). Denklem 3.21, 3.22 ve 3.23'te uygulanan mutasyon stratejisi gösterilmiştir.

$$v_i^{G+1} = x_c^{-G} + F_1 \times \left(x_{en\ iyi}^G - x_{iyi}^G\right) + F_2 \times \left(x_{en\ iyi}^G - x_{kötü}^G\right) + F_3 \times \left(x_{iyi}^G - x_{kötü}^G\right) \tag{3.21}$$

Burada; G, iterasyon adımını, F_1, F_2 ve F_3 katsayılarının hesabı Denklem 3.22'de gösterilmiştir. x_c^{-G} ise, seçilen üç bireyin birleşimini ifade etmektedir (Denklem 3.23).

$$F_i = rand(0, k_i), i = 1, 2, 3$$

$$k_1 = \begin{cases} \left|\frac{f(x_{better})}{f(x_{best})}\right| + \varepsilon & if \left|\frac{f(x_{better})}{f(x_{best})}\right| < 1 \\ \left|\frac{f(x_{best})}{f(x_{better})}\right| + \varepsilon & otherwise \end{cases}$$

$$k_2 = \begin{cases} \left|\frac{f(x_{worst})}{f(x_{best})}\right| + \varepsilon & if \left|\frac{f(x_{worst})}{f(x_{best})}\right| < 1 \\ \left|\frac{f(x_{best})}{f(x_{worst})}\right| + \varepsilon & otherwise \end{cases} \tag{3.22}$$

$$k_3 = \begin{cases} \left|\frac{f(x_{worst})}{f(x_{better})}\right| + \varepsilon & if \left|\frac{f(x_{worst})}{f(x_{better})}\right| < 1 \\ \left|\frac{f(x_{better})}{f(x_{worst})}\right| + \varepsilon & otherwise \end{cases}$$

$$x_c^{-G} = w_1^* \times x_{en\ iyi} + w_2^* \times x_{iyi} + w_3^* \times x_{kötü} \tag{3.23}$$

w_i^* değerleri, $w_i^* \geq 0$ ve $\sum_{i=1}^{3} w_i^* = 1$ şartlarını sağlayacak şekilde Denklem 3.24'te gösterildiği şekilde hesaplanmaktadır.

$$w_i = 1 - \left| \frac{f(x_{en\ iyi}) - f(x_i)}{f(x_{en\ iyi}) - f(x_{max})} \right|, i = 1,2,3 \tag{3.24}$$

Burada; $f(x_{en\ iyi}) = f(x_{min})$ değeri $min\{f(x_i)\}$ değerini ifade etmektedir. $f(x_{max})$ ise bir iterasyonda gözlemlenen maksimum değeri ifade etmektedir. Denklem 3.25'te ise, bu mutasyon stratejisinin uygulanma olasılığı gösterilmiştir. Denklem 3.25'ten anlaşıldığı üzere iterasyon ilerledikçe yeni mutasyon stratejisinin uygulanma ihtimali artmaktadır. İterasyon ilk adımlarında ise standart DE/rand/1 mutasyon stratejisi uygulanmaktadır.

$$\begin{aligned} &if\left(u(0,1) \geq \left(1 - \frac{G}{GEN}\right)^2\right) Then \\ &\quad v_i^{G+1} = x_c^{-G} + F_1 \times \left(x_{en\ iyi}^G - x_{iyi}^G\right) + F_2 \times \left(x_{en\ iyi}^G - x_{kötü}^G\right) + F_3 \times \left(x_{en\ iyi}^G - x_{kötü}^G\right) \\ &Else \\ &\quad v_i^{G+1} = x_{r_1}^G + F \times \left(x_{r_2}^G - x_{r_3}^G\right) \\ &end \end{aligned} \tag{3.25}$$

Burada; G, iterasyon adımını ve GEN ise toplam iterasyon sayısını ifade etmektedir.

Böylece, DG algoritmasının, optimum bölgeye yaklaştıkça daha hassas aramalar yapabilmesi sağlanmış olmaktadır. İlk başlarda DG algoritması hızlı bir şekilde kabaca optimum noktaya yakınsamakta ve iterasyon ilerledikçe daha hassas bir şekilde ilerleme gerçekleştirmektedir.

Çaprazlama işlemi: Çaprazlama işleminde, oluşturulmuş mutasyonlu birey ve belirlenen bir çaprazlama oranı (CR) dikkate alınarak, test bireyi oluşturulur. Bu işlem, parametrelerin her biri için gerçekleştirilir. Eğer rasgele üretilen [0,1] aralığındaki değer, CR değerinden küçükse, test bireyin ilgili parametresi mutasyonlu bireyden aksi durumda hedef bireyden kalıtılır. Denklem 3.26'da bu işlem ifade edilmiştir.

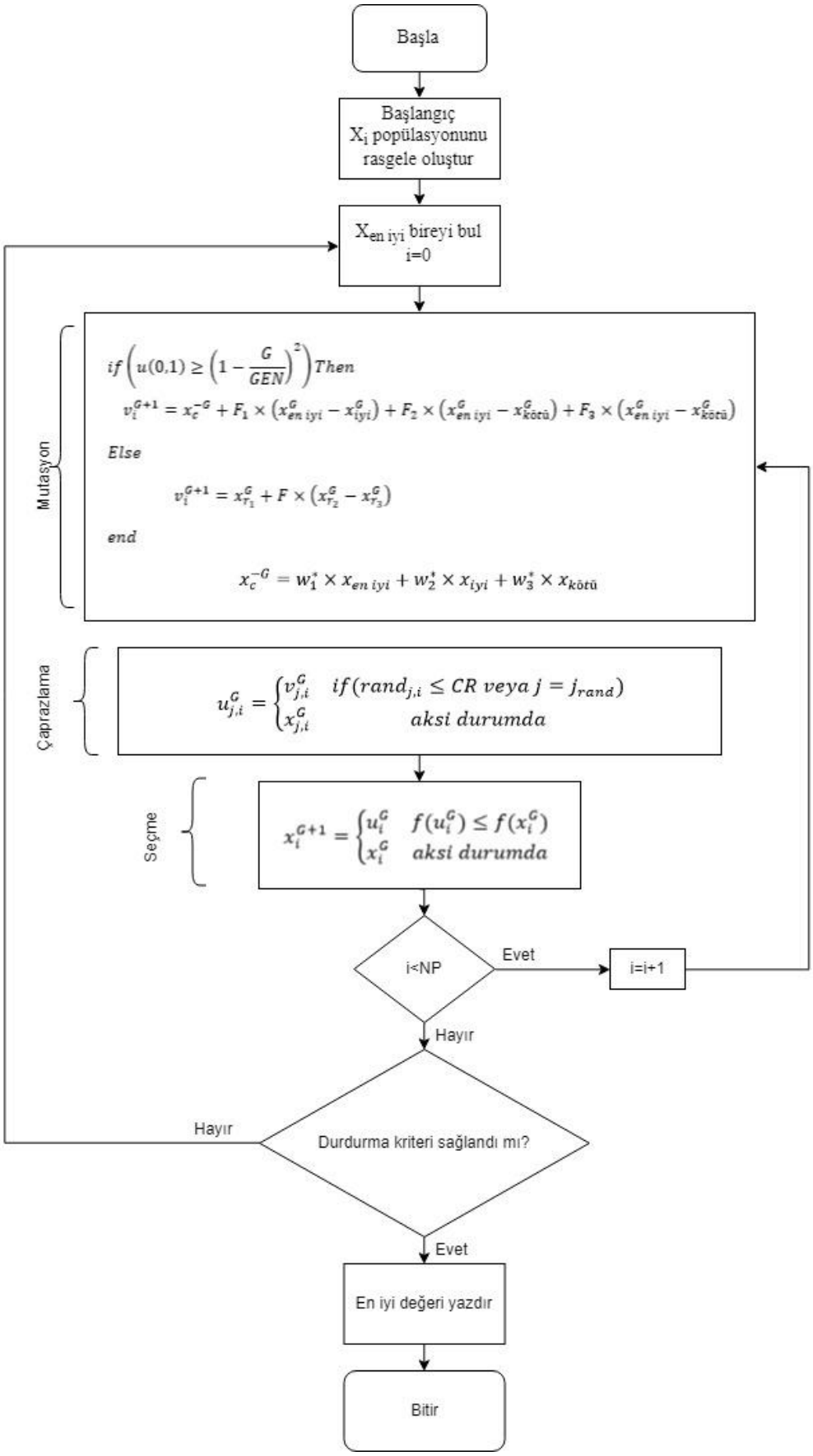
$$u_{j,i}^G = \begin{cases} v_{j,i}^G & if(rand_{j,i} \leq CR\ veya\ j = j_{rand}) \\ x_{j,i}^G & aksi\ durumda \end{cases} \tag{3.26}$$

Burada; $j = j_{rand}$ şartı ile en az bir parametrenin mutasyonlu bireyden kalıtılması garantilenmiş olmaktadır.

Seçme işlemi: DG algoritmasında seçme işlemi Greedy seçme algoritmasına göre yapılmaktadır. Yani, eğer yeni oluşturulan test bireyi, hedef bireyden daha iyi ise gelecek nesle test bireyi, aksi durumda hedef birey aktarılır. Denklem 3.27'de seçme işlemi gösterilmiştir.

$$x_i^{G+1} = \begin{cases} u_i^G & f(u_i^G) \leq f(x_i^G) \\ x_i^G & aksi\ durumda \end{cases} \tag{3.27}$$

Şekil 3.12'de bu tezde kullanılan DG algoritmasının akış diyagramı gösterilmiştir.



Şekil 3.12. DG algoritması akış diyagramı

3.2.8. L-SHADE algoritması

DG algoritmasının bir çok optimizasyon problemine başarılı bir şekilde uygulanabildiği bir önceki bölümde anlatıldı. DG algoritmasının bu başarısı dikkate alındığında, araştırmacılar DG algoritması üzerinde yoğunlaşarak daha da iyiye götürme çabası içine girmektedirler. DG algoritmasının, kontrol parametrelerinin seçilmesi işlemi, performansı doğrudan etkilemektedir. Bu nedenle, DG algoritması ile yapılacak iyileştirme çalışmalarının, öncelikle kontrol parametrelerinin optimum bir şekilde seçilmesi üzerine olması gerekmektedir. Tanabe ve Fukunaga (2013) yaptıkları çalışmada, SHADE algoritmasını geliştirerek, DG algoritmasının iterasyonlar boyunca en iyi sonuçları elde ettiği kontrol parametrelerini hafızada tutan bir yapı geliştirmişlerdir. Kullanıcı tarafından belirlenen sayıda (H) kontrol parametresi hafızada tutularak, her bir iterasyonda (G), CR ve F kontrol parametreleri, hafızada tutulan M_{CR} ve M_F değerleri kullanılarak elde edilmektedir. Başlangıçta hafızada ki tüm M_{CR} ve M_F değerleri 0.5 değerine ayarlanmaktadır. Denklem 3.28 ve 3.29'da, CR ve F değerlerinin elde edilme formülleri verilmiştir.

$$CR_i = \begin{cases} 0 & if\ M_{CR,r_i} = \perp \\ randn_i(M_{CR,r_i}, 0.1) & aksi\ durumda \end{cases} \tag{3.28}$$

$$F_i = randc_i(M_{F,r_i}, 0.1) \tag{3.29}$$

Burada, $randn(\mu, \sigma^2)$ ve $randc(\mu, \sigma^2)$ fonksiyonları, μ ortalama değerli ve σ^2 varyanslı rasgele Normal ve Cauchy dağılımlarını göstermektedir. Eğer hafızada tutulan M_{CR,r_i} değeri sınır değere ($\perp$) ulaşırsa tekrar başa döndürülerek 0 değerinden başlatılır. Ayrıca CR_i değeri [0,1] aralığı dışında bir değer alırsa, 0 ve 1 değerlerinden yakın olanına ayarlanır. F_i değeri de istenen sınırların dışına çıkması durumunda, istenen aralık değeri elde edilinceye kadar Denklem 3.29 uygulanmaya devem eder. Kontrol parametrelerinin ayarlanmasından sonra, Denklem 3.30'da gösterilen current-to-*p*best/1 mutasyon stratejisi ile mutasyonlu birey oluşturulmaktadır.

$$v_i^G = x_i^G + F_i \times \left(x_{en\ iyi}^G - x_i^G\right) + F_i \times \left(x_{r_1}^G - x_{r_2}^G\right) \tag{3.30}$$

Burada, $x_{en\ iyi}^{G}$ rasgele oluşturulan bir birey havuzundaki en iyi bireyi temsil etmektedir. Oluşturulan mutasyonlu bireyin herhangi bir parametresi, istenen aralıklar dışında ise Denklem 3.31, istenen aralık değerleri sağlanıncaya kadar uygulanır.

$$v_{j,i}^{G} = \begin{cases} (x_j^{min} + x_{j,i}^{G})/2 & if\ v_{j,i}^{G} < x_j^{min} \\ (x_j^{max} + x_{j,i}^{G})/2 & if\ v_{j,i}^{G} > x_j^{max} \end{cases} \tag{3.31}$$

Çaprazlama ve seçme işlemleri Denklem 3.26 ve 3.27'de gösterildiği gibi gerçekleştirilir.

Çaprazlama işlemi sonunda, eğer test birey, hedef bireyden daha iyi ise, mevcut CR ve F değerleri, S_{CR} ve S_F olarak kayıt altına alınır ve Denklem 3.32 ve 3.33 kullanılarak, mevcut $M_{CR,k}$ ve $M_{F,k}$ değerlerinde güncelleme işlemi yapılır.

$$M_{CR,k}^{G+1} = \begin{cases} mean_{WA}(S_{CR}) & if\ S_{CR} \neq \emptyset \\ M_{CR,k}^{G} & aksi\ durumda \end{cases} \tag{3.32}$$

$$M_{F,k}^{G+1} = \begin{cases} mean_{WL}(S_F) & if\ S_F \neq \emptyset \\ M_{F,k}^{G} & aksi\ durumda \end{cases} \tag{3.33}$$

$k \in \{1, \dots, H\}$ değeri hafızadaki tutulan kontrol parametrelerinin mevcut indis değerini ifade etmektedir. Her bir güncelleme işleminden sonra k değeri bir artırılarak bir sonraki indise ayarlanır. Hafıza boyutunu geçince tekrar başa döndürülür. Eğer test birey, hedef bireyden daha iyi bir sonuç üretemezse, $S_{CR} = S_F = \emptyset$ ataması yapılarak hafızada herhangi bir güncelleştirme işlemi yapılmaz. Denklem 3.34'te $mean_{WA}$, Denklem 3.35'te ise $mean_{WL}$ fonksiyonlarının hesaplanışı gösterilmiştir.

$$\begin{aligned} mean_{WA}(S_{CR}) &= \sum_{k=1}^{|S_{CR}|} w_k \cdot S_{CR,k} \\ w_k &= \frac{\Delta f_k}{\sum_{k=1}^{|S_{CR}|} \Delta f_k} \\ \Delta f_k &= |f(u_k^G) - f(x_k^G)| \end{aligned} \tag{3.34}$$

$$mean_{WL}(S) = \frac{\sum_{k=1}^{|S|} w_k \cdot S_k^2}{\sum_{k=1}^{|S|} w_k \cdot S_k}$$

$$w_k = \frac{\Delta f_k}{\sum_{l=1}^{|S_{CR}|} \Delta f_l} \tag{3.35}$$

$$\Delta f_k = |f(u_k^G) - f(x_k^G)|$$

Burada S değeri, hem S_{CR} hem de S_F değerlerini ifade etmektedir.

Tüm bu işlemlerden sonra, Tanabe ve Fukunaga (2014), SHADE algoritmasına, iterasyon sayısı ilerledikçe popülasyon büyüklüğü azalan bir özellik ekleyerek L-SHADE algoritmasını geliştirmiş ve algoritmalarının daha da hızlı çalışmasını amaçlamışlardır. Denklem 3.36'da, popülasyon büyüklüğünün lineer olarak azaltılması işlemi gösterilmiştir.

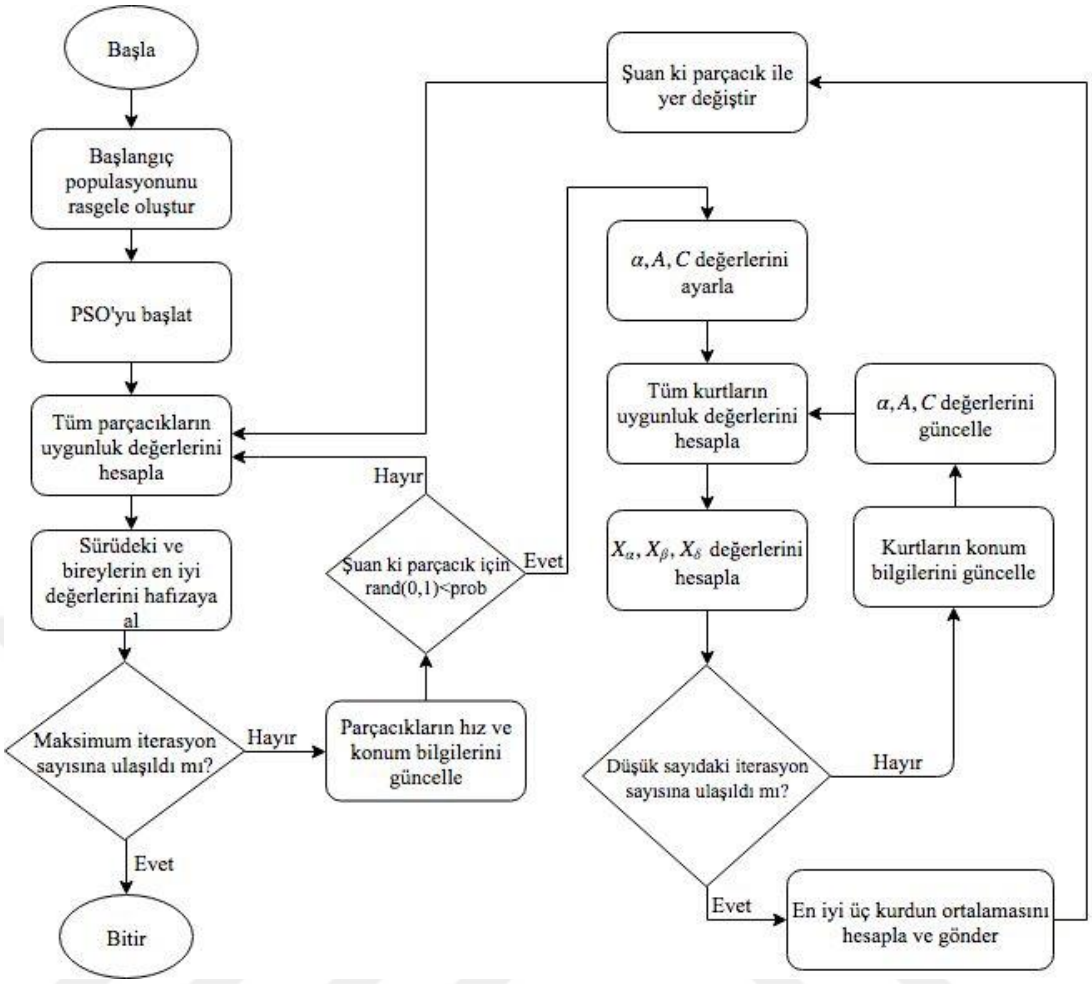
$$N_{G+1} = round\left[\left(\frac{N^{min} - N^{init}}{MAX\_NFE}\right) \times NFE + N^{init}\right] \tag{3.36}$$

Burada N^{min}, popülasyonun alabileceği minimum değeri (mutasyon işleminde mevcut birey haricinde en az üç bireye daha ihtiyaç olduğu için N^{min} değeri minimum dört olabilir.), N^{init} populasyonun başlangıç değerini, MAX_NFE maksimum iterasyon sayısını ve son olarak NFE ise anlık iterasyon adımını ifade etmektedir. Popülasyon azaltılması sürecinde, $N_{G+1} < N_G$ durumu gerçekleşirse populasyondaki kötü bireyler populasyondan çıkarılarak istenen popülasyon büyüklüğü elde edilmektedir.

3.2.9. Hibrit PSO-GKO algoritması

Bu tez çalışmasında yeni bir hibrit yaklaşım geliştirilmiştir. Amaçlanan bu yaklaşım PSO ve GKO algoritmalarının birlikte kullanıldığı ve literatürde daha önceden çalışılmamış bir yaklaşımdır (HPSGKO). Literatürde PSO ve GKO algoritmalarının hibritleştirilmesi ile ilgili yapılmış çok az sayıda çalışma bulunmaktadır (Chopra vd., 2016); (Kamboj, 2016); (Singh ve Singh, 2017). Yapılan bu çalışmalardan farklı bir yöntem amaçlanmıştır.

Geliştirdiğimiz HPSGKO algoritmasında, PSO algoritmasının global arama özelliğinin geliştirilmesi amaçlanmıştır. Bu amaçla, PSO algoritmasına her iterasyonda, kullanıcı tarafından belirlenen bir olasılık değeri ile GKO algoritması tarafından düşük sayıda iterasyon sonucu elde edilen bireyler PSO popülasyonunda ilgili parçacık ile yer değiştirmek suretiyle dâhil edilmektedir (Şenel vd., 2018). GKO algoritması düşük sayıda iterasyon ve düşük sayıda popülasyon büyüklüğü ile çalıştırılmaktadır. Böylece GKO algoritması tarafından kısmi olarak iyileştirilmiş bir birey elde edilmektedir. GKO algoritmasının düşük sayıdaki parametrelerle çalıştırılması, GKO algoritmasının olası yerel minimumlara takılmış bir bireyi PSO'ya aktarmasının önüne geçilmesini sağlamaktadır. PSO algoritmasının ana akışı bozulmadan belirli bir olasılıkla GKO algoritması çalıştırılarak kısmi iyileştirilmiş bireyler elde edildiği için PSO algoritmasının çalışma süresinde uzamalar meydana gelmektedir. Fakat deneysel çalışmaların anlatıldığı bir sonraki bölümde görüleceği üzere geliştirilen HPSGKO algoritması hem PSO hem de GKO algoritmalarından daha başarılı sonuçlar vermektedir. Elde edilen verimlerin daha yüksek olması, özellikle deri gibi geri dönüşü olmayan materyaller için hayati önem arz etmekte ve toplamda zaten dakikalar içinde sonuçlanan yerleşim işlemleri için, bir kaç dakikalık gecikmeler, elde edilen verimler nedeniyle göze alınabilmektedir. Geliştirilen HPSGKO algoritmasının akış diyagramı Şekil 3.13'te verilmiştir.



Şekil 3.13. HPSGKO algoritması akış diyagramı

4. ARAŞTIRMA BULGULARI

Bu tez çalışmasında uygulanan yöntemler aşağıda başlıklar halinde detaylı olarak anlatılmıştır.

4.1. Optimizasyon Probleminin Tanımlanması ve Modellenmesi

Bu tez çalışmasında, deri ayakkabı üretiminde kullanılan derilerden, en az fire verecek şekilde en uygun kesim işlemlerinin yapılması ele alınmıştır. Her ayakkabı birden fazla sayıda kalıptan meydana gelmektedir. Bir ayakkabı modelini oluşturan kalıpların, deri materyalinden en az fire verilecek şekilde kesilerek çıkarılması işlemi bu tezin optimizasyon problemini oluşturmaktadır.

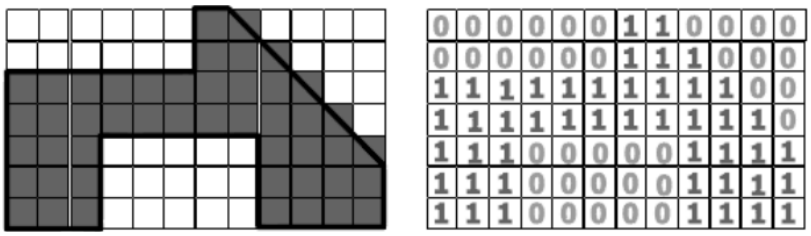
Optimizasyon işlemleri temelde matematiksel olarak ifade edilmektedir. Dolayısıyla her hangi bir optimizasyon problemi ele alındığında, ilk olarak tüm verilerin sayısallaştırılması gerekmektedir. Optimizasyon problemlerinde çözümlerin kabul veya ret edilmesi, problemi ifade eden uygunluk fonksiyonuna göre karar verilmektedir. Uygunluk fonksiyonu hesaplarında ise problem, matematiksel olarak ele alınmalıdır.

4.1.1. Kesim kalıplarının temsili

Deri ayakkabı üretiminde kesim kalıplarının en uygun yerleşimi probleminde ilk olarak ayakkabı modelini oluşturan kalıpların ve deri materyalin sayısal olarak ifade edilmesi gerekmektedir. Literatür incelendiğinde, iki boyutlu kesim optimizasyonu problemlerinde (deri, kâğıt, cam, tekstil vb.) kalıplar ve materyal genel olarak iki farklı şekilde temsil edilmişlerdir. Bunlardan ilki; piksel (raster) yöntemi diğeri ise çokgen (vektörel) yöntemidir.

4.1.1.1. Piksel yöntemi ile temsili

Piksel yöntemi, kalıp ve materyalin bir resim olarak ifade edildiği temsil yöntemidir. Yani her bir kalıp ve materyal mxn boyutundaki bir görüntü matrisi ile ifade edilmektedir (Şekil 4.1).

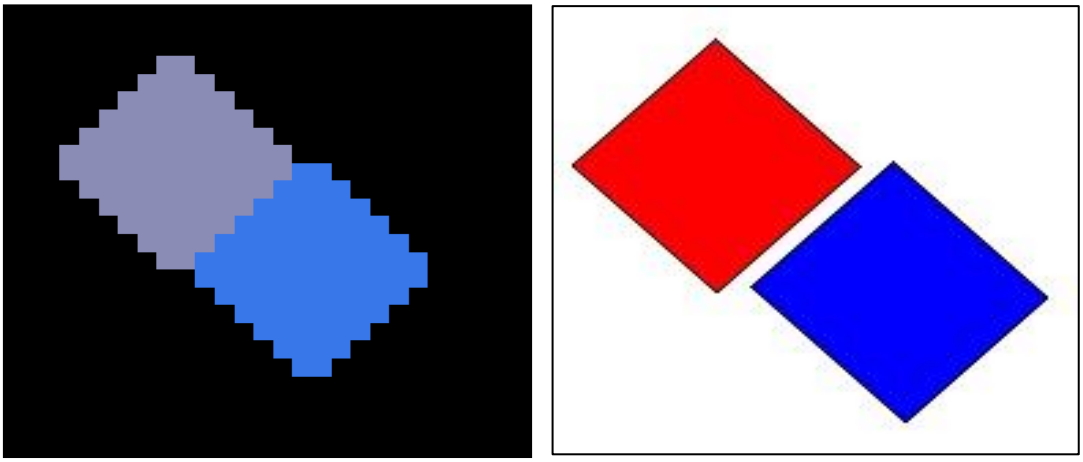


Şekil 4.1. Piksel yöntemi ile örnek bir kalıbın ikili olarak temsili

Piksel yönteminin anlaşılmasının ve optimizasyon probleminin uygunluk fonksiyonunda maliyet hesabının kolay olmasından dolayı bir çok çalışmada tercih edilmektedir. Piksel yöntemi ile maliyet hesabı yaparken sadece basit olarak matrislerde dört işlem gerekmektedir. Matematiksel olarak çok karmaşık işlemlerden kurtarmaktadır. Piksel yöntemi kesim kalıplarının düzenli şekillere sahip olduğu problemlerde kullanmaya çok uygundur. Kalıpların şekillerindeki düzensizlik arttıkça piksel yönteminin dezavantajları ortaya çıkmaktadır. Bu tez çalışmasının ilk dönemlerinde, literatürde kabul görmüş ESICUP veri setleri ile çalışılmıştır. Bu veri setinin içinde bulunan örneklerin çoğunluğu düzenli şekle sahip kalıplardır. Bu nedenle piksel yöntemi ile temsil edilmeye uygundur. Tez çalışması belirli bir olgunluğa gelinceye kadar piksel yöntemi ile çalışmalar devam ettirilmiştir. Ancak gerek deri ayakkabı kesim optimizasyonu problemindeki kalıpların düzenli şekle sahip olmayışı gerek piksel yönteminin bazı dezavantajları bu tez çalışmasında çokgen (vektörel) yöntemi ile çalışmaya zorlamıştır.

Şekil 4.1'den görüldüğü üzere, örnek bir kalıp piksel yöntemi ile temsil edilmiştir. Fakat kalıbın düzgün kenarları çok iyi temsil edilebilmişken, düzgün olmayan yani eğimli kenarları çok iyi temsil edilememiştir. Kalıp olduğundan daha fazla alana taşmıştır. Piksel yöntemi ile bu problem büyük oranda aşılabilmektedir. Bunun için kalıbı temsil eden resmin çözünürlüğünün artırılması yeterli olacaktır. Fakat bu durum hesaplama işlemlerinde her ne kadar dört işlemde kullanılsa da yavaşlamalara neden olmaktadır. Optimizasyon problemlerinde hız en önemli kriterlerden biri olduğu için mümkün olduğunca gereksiz hesaplamalardan kaçınmak gerekmektedir.

Piksel yönteminin düzgün şekilli olmayan kalıpları temsil etmekteki bir diğer dezavantajı ise Şekil 4.2'de gösterilmiştir.



(a) (b)

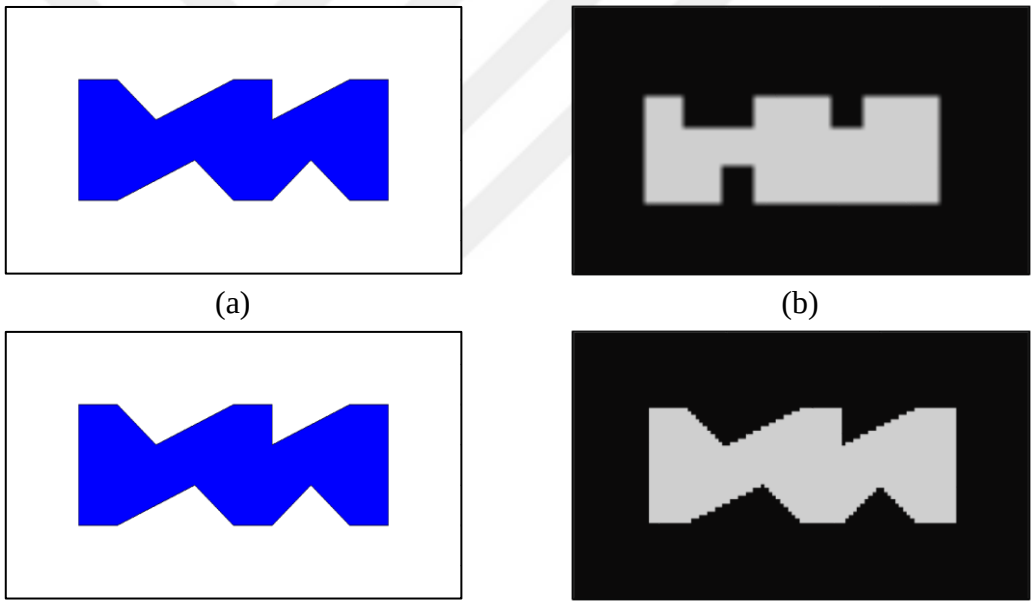
Şekil 4.2. Kalıpların piksel ve çokgen yöntemleri ile temsili (a) Piksel yöntemi ile temsili (b) Çokgen yöntemi ile temsili

Şekil 4.2.a ve Şekil 4.2.b'de gösterilen örnek kalıplar, her iki resimde de aynı büyüklükte ve aynı koordinata yerleştirilmiştir. Fakat piksel yöntemi ile temsil edildiğinde, iki kalıp birbirine yapışık olmakta ve daha fazla birbirine yaklaşamamaktadır. Şekil 4.1'de belirtilen durumdan dolayı kalıplar olduğundan büyük olarak ifade edilmektedir. Ancak Şekil 4.2.b'de aynı kalıplar çokgenler ile ifade edildiğinde aradaki boşluk net bir şekilde görülmekte ve kalıplar birbirine daha da yaklaşabilme imkânına sahip olmaktadır. Burada görüldüğü üzere piksel tekniği düzensiz şekillerle çalışıldığı zaman problemlere neden olmaktadır. Kullanılan çözünürlük problemin boyutunu küçültse de fazla işlem zamanı gerektireceğinden dolayı tercih edilmemektedir. Bu nedenle bu tezin gerçekleştirilmesinde çokgenlerle temsil yöntemi tercih edilmiştir. Ancak piksel yöntemi ile optimizasyon problemi yüksek oranda sonuca bağlanmıştır. Piksel tekniğinin hesaplama işlemleri kolaylığından yararlanılarak, optimizasyon problemi çözülmüş ve nihayetinde çokgenler ile temsil yöntemine geçilerek problem sonuçlandırılmıştır.

4.1.1.2. Çokgen yöntemi ile temsili

Bu tez çalışmasının ilk dönemlerinde piksel yöntemi ile çalışmalara başlanmış ve yeterli düzeyde ilerleme kaydedilmiştir. Düzensiz geometrik şekillere sahip kalıpların, yine düzenli veya düzensiz, içinde boşluk olan veya olmayan materyallerden kesilerek çıkarılması işleminde ciddi maliyet gerektiren optimizasyon algoritmaları kullanılmaktadır. Hesaplama maliyetini olabildiğince azaltabilmek için çözüm arama

uzayını mümkün olduğunca daraltmak gerekmektedir. Maliyet hesaplaması bakımından piksel yöntemi ile temsil edilmenin avantajı diğer yöntemlere göre ön plandadır. Ancak günlük yaşamda kullanılan ayakkabı model kalıplarının dijital ortama alınması sürecinde piksel yöntemi ile temsil edilmesinde bazı sorunlar meydana gelmektedir. Bu sorunların en önemlisi, bir pikselin sadece köşe ucundan geçen bir kenar nedeniyle o pikselin tamamının kullanılıyor olarak algılanmasıdır. Çünkü piksel değeri ya dolu ya da boş olarak işaretlenmektedir. Bu sorundan kaynaklı diğer bir problem ise düşük çözünürlüklü bazı çokgenlerin piksellere dönüştürüldüğünde şeklin tamamen değişmesidir. Şeklin bozulmaması için çokgenin alanı artacak şekilde ölçeklenerek çözünürlüğünün artırılması gerekmektedir. Çözünürlüğün artması ise ekstra işlem ve zaman kaybı gibi olumsuzlukları doğurmaktadır.



Şekil 4.3. Çokgenlerin Piksel Olarak Temsil Edilmesi Örneği (a) Orijinal Çokgen (b) Piksele Dönüştürme İşlemi Sonunda (c) 10 Kat Büyütülen Çokgen (d) 10 Kat Büyütülen Çokgenin Piksele Dönüştürme İşlemi Sonunda

Ayrıca piksel yöntemi ile çalışma sırasında en uygun yerleşim açısının tespiti yapılırken resmin farklı açılarda döndürülmesi ile de resmin şekli bozulmakta ve hesaplamalarda yanlışlıklara sebep olmaktadır.

Bu nedenle bazı problemlerin çözümünde piksel tabanlı çalışmak yerine çokgenlerle çalışmanın tercih edilmesi bir zorunluluk olmaktadır. Ayakkabı model kalıplarının düzenli ya da düzensiz geometrik şekillerle temsil edilmesi mümkündür. Her bir geometrik şekil ikiden fazla eğrinin birleşmesiyle elde edilmektedir. Bu eğriler doğrusal olabildiği gibi farklı şekillerde de olabilmektedir. Çokgenlerle çalışmanın avantajları olduğu kadar dezavantajları da bulunmaktadır. En büyük dezavantajı çok fazla ve karmaşık matematiksel hesaplamaların kullanılması gerektiğidir. Hesaplama maliyetlerinin artması, toplam maliyeti de artırdığı için en uygun çözüme götürecek olan maliyet hesabının çok iyi seçilmesi kaçınılmaz olmaktadır. Aşağıda çokgenlerle çalışma sırasında ihtiyaç duyulan bazı matematiksel hesaplamalar açıklanmıştır.

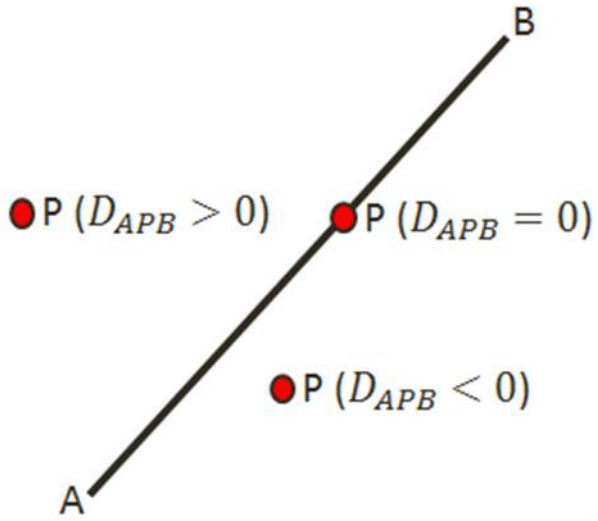
D-Fonksiyonu (Direct Trigonometry)

Çokgenlerle çalışıldığı zaman, çözüm uzayındaki bir noktanın herhangi bir çokgenin içinde mi, dışında mı veya üzerinde mi olduğunun araştırılması gerekmektedir. D-Fonksiyonu, çözüm uzayındaki herhangi bir noktanın iki noktası verilen bir doğrunun üzerinde mi, sağında mı yoksa solunda mı olduğu bilgisini vermektedir (Timmerman, 2013). Denklem 4.1'de noktanın doğruya göre konumunu hesaplayabilmek için gerekli olan ara hesap işlemi ifade edilmiştir. Denklem 4.2'de ise noktanın konumu hakkında bilgi veren formül ifade edilmiştir.

$$D_{APB}=(X_A-X_B)*(Y_A-Y_P)-(Y_A-Y_B)*(X_A-X_P) \tag{4.1}$$

$$D=\begin{cases} sol & D_{APB}>0 \\ sağ & D_{APB}<0 \\ üzerinde & D_{APB}=0 \end{cases} \tag{4.2}$$

Şekil 4.4'te D-Fonksiyonu yönteminin, verilen bir $|AB|$ doğrusuna göre P noktasının sonucu verilmiştir. D-Fonksiyonu yöntemi, yerleşimi yapılan bir çokgenin arama uzayının sınırları ile veya daha önceden yerleşimi yapılan çokgenlerle temas halinde (touch) olup olmadığının kontrolü sırasında kullanılmaktadır.



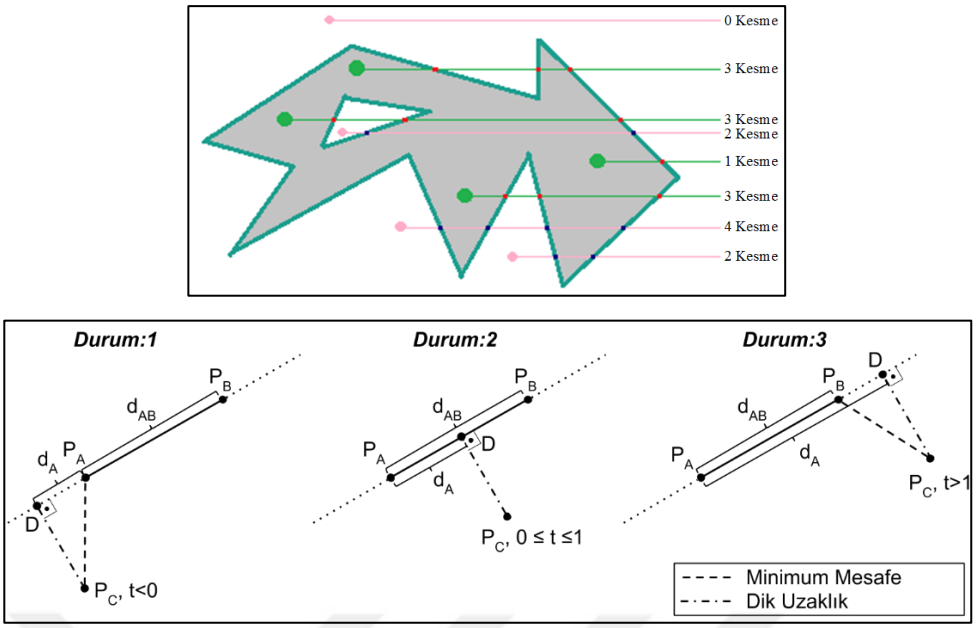
Şekil 4.4. D-Fonksiyonu yöntemi

Verilen Bir *P* Noktasının Çokgene Göre Konumu

Çokgenlerle çalışılırken maliyet hesabında en fazla kullanılan matematiksel işlem, arama uzayındaki bir noktanın, verilen bir çokgenin içinde olup olmadığının hesaplanması işlemidir. Bu işlem hem çokgenin en uygun yere yerleşip yerleşmediğinin kontrolü sırasında hem de iki çokgen arasında çakışma (overlap) olup olmadığının kontrolü sırasında kullanılmaktadır. Bir noktanın bir çokgen içinde olup olmadığının kontrolü için farklı yöntemler geliştirilmiştir. Bunlardan en yaygın olanı, verilen bir P noktasının sağına ve soluna doğru sonsuz bir doğru çizilmesi yöntemidir. Doğrunun test edilen çokgeni kestiği noktaların toplamına bakılır. Eğer tek sayıda kesiyorsa, P noktası çokgenin içinde, çift sayıda kesiyorsa çokgenin dışında olduğu anlaşılır. Şekil 4.5.a'da verilen bir P noktasının, verilen bir C çokgeni içinde ve dışında olduğu durumlar örnekle ifade edilmiştir.

(a)

(b)



Şekil 4.5. Verilen noktaların, çokgen ve doğru parçalarına göre durumları (a) Farklı P noktalarının C çokgenine göre durum örnekleri (b) Örnek bir P_C noktasının, $|P_A P_B|$ doğru parçasına olan uzaklığının hesaplanması

Bir P Noktasının Çokgene Olan Minimum Uzaklığının Hesaplanması

Verilen bir P_C noktasının, bir çokgene olan uzaklığı hesaplanırken bir noktanın bir doğruya olan uzaklığından yararlanılabilmektedir. Çünkü çokgenlerin her kenarı bir doğru olarak düşünülebilir. P_C noktasının tüm kenarlara olan uzaklığı hesaplanarak minimum olan değer istenilen mesafe olarak tanımlanabilir. Bir doğru parçasına, bir noktanın minimum mesafesinin bulunması için, noktanın doğruya olan dik uzaklığının bulunması gerekmektedir. Çünkü en kısa mesafe dik uzaklıktır. Fakat doğru parçaları sonsuz uzunlukta olmadığı için her zaman dik uzaklık bulunmayabilir. Bunun için öncelikle, noktanın doğrunun sınırları içinde olup olmadığının kontrol edilmesi gerekmektedir. Eğer doğrunun sınırları dâhilinde ise dik uzaklık minimum mesafeyi vermektedir. Eğer doğrunun sınırları dışında ise, doğru parçasının iki uç noktasından yakın olanına olan uzaklığı minimum mesafeyi vermektedir. Şekil 4.5.b'de üç farklı durum için görsel bir örnek verilmiştir. Minimum mesafenin hesaplanması işlemi yapılırken, P_C noktasının, $|P_A P_B|$ doğrusuna olan minimum uzaklığının hesaplanması için türevden yararlanılmaktadır. Çünkü minimum mesafenin bulunması için P_C noktasının, $|P_A P_B|$ doğrusunu kestiği noktanın bulunması gerekmektedir. Bu ise türev ile basitçe hesaplanabilecek bir optimizasyon problemidir. Bu işlem yapılırken ilk olarak, P_C noktasının, $|P_A P_B|$ doğrusunu kestiği noktanın (D) konumunu ifade etmek

gerekmektedir. D noktası, P_A noktasından P_B noktası yönünde t ile ifade edilen bir oranda P_A noktasından uzakta yer aldığı varsayılmaktadır. Yani, D noktasının P_A noktasına olan uzaklığının, $|P_A P_B|$ doğru parçasının uzunluğuna oranı t değerini vermektedir. t değeri bilindikten sonra D noktasının X ve Y koordinatlarının değerleri Denklem 4.3 ve 4.4'te sırasıyla verilmiştir.

$$D_X = P_{A_X} + t * (P_{B_X} - P_{A_X}) \tag{4.3}$$

$$D_Y = P_{A_Y} + t * (P_{B_Y} - P_{A_Y}) \tag{4.4}$$

D noktasının koordinatları belirlendikten sonra, P_C noktasına olan uzaklığı ($|DP_C|$) iki nokta arasındaki uzaklık formülü ile hesaplanır. Denklem 4.5'te $|DP_C|$ uzaklığının hesaplanması gösterilmiştir.

$$|DP_C| = \sqrt{\left(\left(P_{A_X} + t * \left(P_{B_X} - P_{A_X}\right)\right) - P_{C_X}\right)^2 + \left(\left(P_{A_Y} + t * \left(P_{B_Y} - P_{A_Y}\right)\right) - P_{C_Y}\right)^2} \tag{4.5}$$

$|DP_C|$ değerini minimum yapacak olan t değerinin bulunması gerekmektedir. Bu işlem, tek değişkenli bir optimizasyon problemi olup türevi alınarak sıfıra eşitlenmesi ile uç noktası bulunur. İşlemlerin kolay halledilebilmesi için karekök içindeki ifadenin minimum yapılması hedeflenmektedir. Çünkü karekök içindeki ifadenin minimum olması aynı zamanda tüm ifadeyi de minimum yapmaktadır. Karekök içindeki ifadenin t'ye göre türevi alındıktan sonra Denklem 4.6 elde edilmektedir. t değerinin eşitliğin bir tarafına çekilmesi ise Denklem 4.7'de gösterilmiştir.

$$\begin{aligned}\frac{d}{dt}|DP_C|^2 = {} & 2 * \left(\left(P_{A_X} + t * \left(P_{B_X} - P_{A_X}\right)\right) - P_{C_X}\right) * \left(P_{B_X} - P_{A_X}\right) \\ & + \; 2 * \left(\left(P_{A_Y} + t * \left(P_{B_Y} - P_{A_Y}\right)\right) - P_{C_Y}\right) * \left(P_{B_Y} - P_{A_Y}\right)\end{aligned} \tag{4.6}$$

$$t = \frac{\left(P_{C_X} - P_{A_X}\right) * \left(P_{B_X} - P_{A_X}\right) + \left(P_{C_Y} - P_{A_Y}\right) * \left(P_{B_Y} - P_{A_Y}\right)}{\left(P_{B_X} - P_{A_X}\right)^2 + \left(P_{B_Y} - P_{A_Y}\right)^2} \tag{4.7}$$

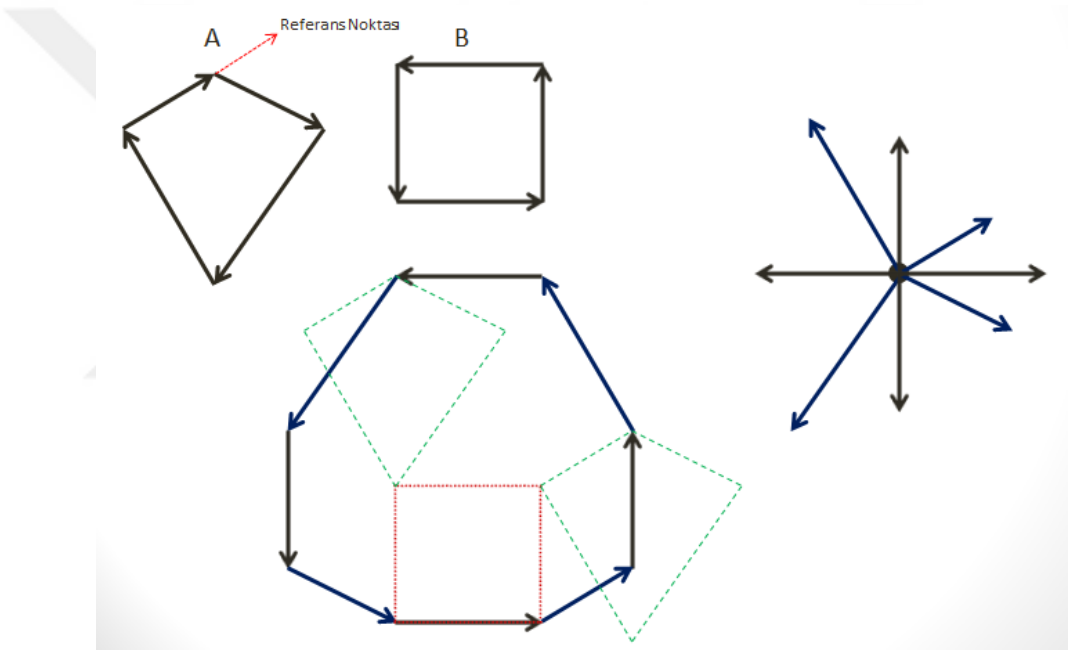
Denklem 4.7 sıfıra eşitlenerek t değeri bulunur. t değerinin durumuna göre en kısa mesafe hesaplanır. t değeri P_A'dan P_B'ye doğru pozitif yönde değer alırken tersi yönde negatif değerler almaktadır. t değeri $[0-1]$ arasında ise dik uzaklık doğru parçası üzerinde, 1'den büyük olduğu durumda P_B noktasının ötesinde, 0'dan küçük olduğu durumlarda ise P_A noktasının gerisinde bulunmaktadır. Doğru parçasının dışında yer aldığı durumlarda ise yakın olan uç hangisi ise o noktaya olan uzaklık minimum değeri vermektedir. Denklem 4.8'de minimum mesafenin (d_{min}) hesaplanması gösterilmiştir.

$$d_{min} = \begin{cases} \sqrt{\left(P_{A_X} - P_{C_X}\right)^2 + \left(P_{A_Y} - P_{C_Y}\right)^2} & t < 0 \\ \sqrt{\left(\left(P_{A_X} + t * \left(P_{B_X} - P_{A_X}\right)\right) - P_{C_X}\right)^2 + \left(\left(P_{A_Y} + t * \left(P_{B_Y} - P_{A_Y}\right)\right) - P_{C_Y}\right)^2} & 0 \leq t \leq 1 \\ \sqrt{\left(P_{B_X} - P_{C_X}\right)^2 + \left(P_{B_Y} - P_{C_Y}\right)^2} & t > 1 \end{cases} \quad (4.8)$$

No-Fit Polygon

En uygun yerleşim yerinin tespit edilmesi sırasında en büyük problem hız olarak karşımıza çıkmaktadır. Maliyet hesabı yaparken olabildiğince basit matematiksel ifadelerle çalışmak ve olabildiğince arama uzayını daraltmak en uygun yerin tespitinde hızlı bir şekilde sonuca varmamızı sağlamaktadır. Çokgenler ile yapılan çalışmalar incelendiğinde literatürde en fazla karşılaşılan yöntemin NFP yöntemi olduğu anlaşılmaktadır. NFP yöntemi; verilen iki çokgenden biri sabit diğeri hareketli olmak üzere, hareketli çokgenin sabit olan çokgenin etrafında sürekli temasta (touch) olacak şekilde döndürülmesiyle elde edilen yörüngenin hesaplanması olarak açıklanabilir (Whitwell, 2004). Hareketli çokgenin herhangi bir noktası referans noktası olarak seçilir ve yörünge bu referans noktasının izlediği yol olarak tanımlanır. Böylece arama uzayında, materyalin tüm alanı değil, sadece yörünge üzerindeki noktalardan hangisinin en uygun yerleşimi verdiği hesaplanır. Hareketli çokgenin seçilen referans noktası, NFP yörüngesinin içinde kalıyorsa çakışma (overlap) olduğu, dışında kalıyorsa temasın (touch) olmadığı, üzerinde ise iki çokgenin temasta (touch) olduğu anlaşılmaktadır. En uygun yerleşim bulunurken çokgenlerin birbirine en yakın olacak şekilde yerleşmesi materyalden verilecek fire miktarını azaltmaktadır. Bu nedenle

çokgenler mümkün olduğunca birbirine yakın yerleştirilmelidir. En uygun yerleşim ise çokgenlerin temas (touch) halinde olduğu yerleşimlerdir. NFP yörüngesinin ve alanının hesaplanmasında farklı yöntemler geliştirilmiştir. Kullanımı en basit olan yöntem, verilen çokgenlerin her bir kenarını bir doğru vektör olarak kabul edilerek yapılan yöntemdir. Bu yöntemde sabit olan çokgenin kenarlarını saat yönünün tersinde, hareketli olan çokgenin yönünü ise saat yönünde vektörlere dönüştürerek işlem yapılır. Vektörlere dönüşüm işlemi tamamlandığında tüm vektörler ortak bir O merkez noktasında başlangıç noktaları birleşecek şekilde yerleştirilir ve tüm vektörler uç uca eklenerek yörünge hesaplanır. Şekil 4.6'da örnek bir NFP hesaplama işlemi gösterilmiştir.



Şekil 4.6. Örnek NFP Yöntemi Hesabı

Çakışma Alanlarının Hesaplaması

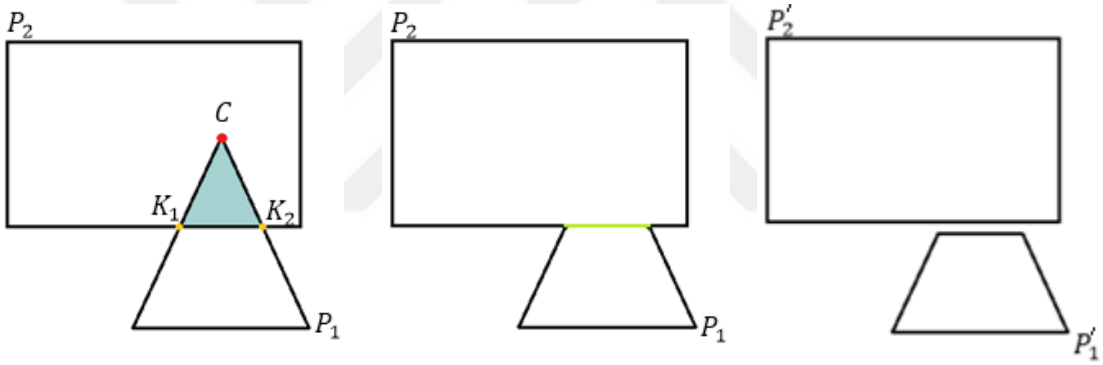
Çokgenlerle çalışıldığında karşılaşılan en büyük problem, çokgenler arasındaki çakışma alanlarının hesaplanması zorluğudur. Maliyet hesaplamasında büyük önem arz eden çakışma alanları hesabı için olası durumlar göz önünde bulundurularak gerekli hesaplamalar gerçekleştirilmiştir.

Literatürde çokgenler arası çakışmaların hesaplanması için kullanılan farklı yöntemler bulunmaktadır. Bu tezde, çokgenler arası çakışmaların hesabı için olası durumlar tespit edilmiş ve gerekli kodlamalar yapılarak işlem gerçekleştirilmiştir.

Çakışma alanları hesaplanması için gerekli durumlar incelendiğinde aşağıda belirtilen üç farklı durumun olduğu tespit edilmiştir. Geliştirilen kod tüm durumları kapsayacak şekilde yazılmıştır.

Durum 1:

Şekil 4.7'de görüldüğü gibi bir P_1 çokgenin herhangi sadece bir köşesinin, diğer P_2 çokgenin içinde kaldığı durumdur. Öncelikli olarak algoritma, her iki çokgeninde tüm köşe noktalarını test ederek herhangi bir köşe noktasının, diğerinin içinde olup olmadığını kontrol etmektedir. Herhangi bir köşe noktası, diğer çokgenin içinde olması durumunda, iki çokgen arasında çakışmanın olduğu anlaşılmaktadır.



(a) (b) (c)

Şekil 4.7. Durum 1 için örnek resim (a) Çakışma noktalarının ve içteki noktanın tespiti (b) İçte kalan noktanın çokgenden çıkarılması (c) Çokgenlerin çakışma boyunca birbirinden ayrılması

İçte kalan nokta (C) tespit edildikten sonra C noktasının bir öncesi ve bir sonrasında bulunan nokta takip edilerek içte kalan noktanın tek olduğundan emin olunur. Daha sonra C noktasına bağlı olan kenarların, diğer çokgen ile çakıştığı noktalar (K_1 ve K_2) tespit edilmektedir. Kenar tespit işlemi için çokgenlerin tüm kenarları bir doğru gibi kabul edilerek, P_2 çokgeninin tüm kenarları arasında ortak bir kesişim noktası aranmaktadır. Böylece ortak kesişim noktası olan kenarların birbiriyle çakıştığı tespit edilmektedir. Kesişim noktası aynı zamanda çakışma noktalarının koordinat değerlerini ifade etmektedir. Farklı yöntemlerle kesişim noktası tespit edilebilir. En basit yolu, doğru denklemlerinde x ve y değerleri eşitlenerek ortak çözüm yapılır.

Ortak x ve y değerleri kesişim noktasının koordinatlarını ifade etmektedir. Kesişim noktaları tespit edildikten sonra, bir köşesi diğerinin içinde kalan çokgenden, içerde kalan nokta çıkarılarak tespit edilen kesişim noktaları sırasıyla çıkarılan yere eklenir. Böylece sonuçta iki adet çakışmayan çokgen (P_1' ve P_2') elde edilmiş olmaktadır. Son durumda çokgenlerin köşe noktaları Çizelge 1'de verildiği şekilde olmaktadır.

Çizelge 4.1. Çokgenlerin ayrılmadan önceki ve sonraki köşe noktaları

Ayrılmadan Önceki Durum		Ayrılmadan Sonraki Durum	
P_1	P_2	P_1'	P_2'
x_1, y_1	x_1, y_1	x_1, y_1	x_1, y_1
x_2, y_2(C noktası)	x_2, y_2	K_{1_x}, K_{1_y}	x_2, y_2
x_3, y_3	x_3, y_3	K_{2_x}, K_{2_y}	x_3, y_3
	x_4, y_4	x_3, y_3	x_4, y_4

Son olarak başlangıçtaki çokgenlerin alanları toplamından son durumdaki çokgenlerin alanları toplamı çıkarılarak toplam çakışma bölgesinin alanı hesaplanmış olmaktadır (Denklem 4.9).

$$Alan_{Çakışma} = (Alan_{P_1} + Alan_{P_2}) - (Alan_{P_1'} + Alan_{P_2'}) \tag{4.9}$$

Durum 1 incelendiğinde P_1 çokgeninin bir köşe noktasının P_2 çokgeni içinde olduğu ancak P_2 çokgeninin hiçbir köşe noktasının P_1 çokgeni içinde kalmadığı Şekil 4.7'den görülmektedir.

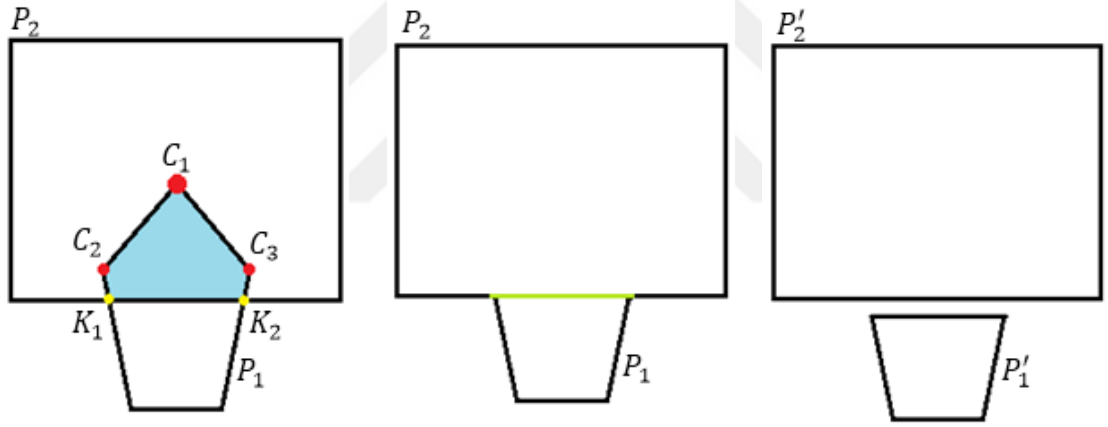
Çokgenlerin alan hesabı yapılırken Denklem 4.10'da belirtilen formül kullanılmıştır. Denklem 4.10'a göre, ilk olarak çokgenin tüm köşe noktaları ($x\ ve\ y$) sırasıyla alt alta yazılır. İlk sıraya yazılan koordinat tablonun en altına tekrar eklenir. Daha sonra, bir önceki köşe noktasının x koordinat değeri ile bir sonraki köşe noktasının y koordinat değerleri çarpılarak toplanır. Bu toplamdan, bir önceki köşe noktasının y koordinat değeri ile bir sonraki köşe noktasının x koordinat değerleri çarpımları toplamı çıkarılır. Son olarak elde edilen değer iki ile bölünerek çokgenin alanı hesaplanır.

$$Alan = \frac{\sum_{i=1}^{s} x_i * y_{i+1} - \sum_{i=1}^{s} x_{i+1} * y_i}{2} \tag{4.10}$$

Burada; s, koordinatların alt alta yazıldığı tablodaki satır sayısı (çokgenin köşe nokta sayısı + 1, ilk sıraya yazılan koordinat en alta tekrar yazıldığı için bir eklendi), x ve y, köşe noktasının koordinatlarını ifade etmektedir.

Durum 2:

Çakışma bölgesi tespit edilirken diğer bir durum, çakışma bölgesinde birden fazla köşe noktasının kaldığı durumdur. Durum 1'de olduğu gibi sadece P_1 çokgeninin köşe noktaları P_2 çokgeni içinde yer almaktadır. P_2 çokgeninin herhangi bir köşe noktası P_1 çokgeni içinde yer almamaktadır. Şekil 4.8'de görüldüğü üzere eğer çakışma bölgesinde birden fazla köşe noktası varsa içerde kalan köşe noktalarının bir önceki ve bir sonraki noktaları çokgen dışında kalıncaya kadar takip edilir. İçerde kalan köşe noktalarının en baş ve en sondaki noktaları tespit edilir. Durum 1'dekine benzer şekilde yine kesişim noktaları hesaplanır. Daha sonra içerde kalan köşe noktaları P_1 çokgeninden çıkarılarak kesişim koordinatları P_1 çokgenine sırasıyla eklenir.



(a) (b) (c)

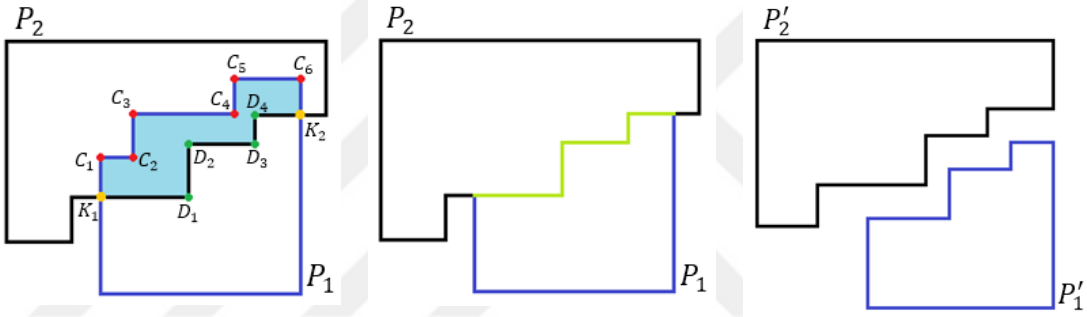
Şekil 4.8. Durum 2 için örnek resim (a) Çakışma noktalarının ve içteki noktaların tespiti (b) İçte kalan noktaların çokgenden çıkarılması (c) Çokgenlerin çakışma boyunca birbirinden ayrılması

İçte kalan noktalar (C_1, C_2 ve C_3) tespit edildikten sonra noktaların başlangıç (C_2) ve bitiş (C_3) noktaları takip edilerek tespit edilir. İçerde kalan nokta sayısından bağımsız olarak sadece başlangıç ve bitiş noktalarının tespit edilmesi önem arz etmektedir. Başlangıç ve bitiş noktalarının bağlı olduğu ve içerde kalmayan kenarlar P_2 çokgeni ile çakışmaktadır. Bu nedenle sadece bu kenarların, P_2 çokgeninin tüm kenarlarıyla çakışması test edilerek kesişim noktalarının koordinatları bulunmaktadır. Durum 1'de olduğu gibi içerde kalan noktalar P_1 çokgeninden çıkarılarak yerine sırasıyla kesişim

noktalarının koordinatları K_1 ve K_2 eklenir. Son olarak Denklem 4.9'da verilen formül ile çakışma bölgesinin alanı hesaplanır.

Durum 3:

Durum 1 ve Durum 2'den farklı olarak çakışma durumunda her iki çokgeninde birbirine içinde kalan köşe noktalarının olduğu durumlar mevcuttur (Şekil 4.9). Bu durumda öncelikle çakışan çokgenlerden herhangi biri seçilerek işleme başlanır. Seçilen çokgenin, diğer çokgen içinde kalan köşe noktaları tespit edilir. Daha sonra içerde kalan köşe noktaları takip edilerek başlangıç ve bitiş köşe noktaları bulunur. Durum 1 ve Durum 2'den farklı olarak burada, içerde kalan noktalar çıkarıldıktan sonra kesişim noktalarının koordinatları ile bu noktalar arasında kalan diğer çokgenin köşe noktaları da (-ki bu noktalar seçilen çokgenin içinde kalan noktalar) sırasıyla seçilen çokgene eklenir.



(a) (b) (c)

Şekil 4.9. Durum 3 için örnek resim (a) Çakışma noktalarının ve içteki noktaların tespiti (b) İçte kalan noktaların çokgenden çıkarılması (c) Çokgenlerin çakışma boyunca birbirinden ayrılması

Şekil 4.9'da görüldüğü gibi öncelikle P_1 çokgeni seçilmektedir. Daha sonra P_1 çokgeninin P_2 çokgeni içinde bulunan noktalar $(C_1, C_2, C_3, C_4, C_5, C_6)$ tespit edilmektedir. Daha sonra başlangıç ve bitiş noktaları olan C_1 ve C_6 noktalarının bağlı olduğu kenarlar ile P_2 çokgeninin kenarları arasındaki çakışma noktalarının koordinatları bulunmaktadır. P_1 çokgeninin içerde kalan noktaları çıkarılarak sırasıyla öncelikle K_1 noktası daha sonra P_2 çokgeninin P_1 içinde kalan noktaları (D_1, D_2, D_3, D_4) son olarak da K_2 noktası eklenerek çokgenler birbirinden ayrılmış olmaktadır. Denklem 4.9 kullanılarak çakışma alanı hesaplanmaktadır.

4.1.2. Uygunluk fonksiyonunun tanımlanması

Optimizasyon problemleri ile çalışılırken, problemin önemli aşaması tanımlanan uygunluk fonksiyonudur. Çünkü kullanılan optimizasyon yöntemi ne kadar başarılı olursa olsun, eğer uygunluk fonksiyonu yeterince iyi seçilmemişse problemin çözümüne istenilen düzeyde ulaşılamaz. Bu nedenle, uygunluk fonksiyonu çok iyi bir şekilde tanımlanmalıdır. Uygunluk fonksiyonunun olabildiğince sade matematiksel ifadelerden oluşması gerekmektedir. Aksi halde problemin çözümü çok zaman alacaktır. Uygunluk fonksiyonu, optimizasyon yöntemi problemi çözerken defalarca kullanılmaktadır. Optimizasyon probleminin en fazla zaman harcadığı bölüm uygunluk fonksiyonu ile maliyetin hesaplandığı bölümdür. Bu nedenle olabildiğince sade ve hızlı çalışacak bir uygunluk fonksiyonunun seçilmesi büyük önem arz etmektedir.

Bu tez çalışmasında ele alınan problemin bazı kısıtlar altında çözüme ulaşması gerekmektedir. Bu kısıtlar aynı zamanda optimizasyon yöntemi için kullanılacak uygunluk fonksiyonunun tanımlanmasında etkili olan faktörlerdir. Bu faktörleri sıralayacak olursak;

- Materyal sınırlarının dışına taşmamak,
- Üst üste çakışmaların olmaması,
- Deforme olmuş bölgelere yerleşim yapılmaması,
- Uygun olmayan açıda yerleşim yapılmaması,
- Daha uygun yer varken başka bir yere yerleşim yapılmaması vs. sıralanabilir.

Bu tezde ele alınan problemin çözümünde, yukarıda sıralanan tüm maddelerin göz önünde bulundurulması gerekmektedir. Tüm bu durumlar dikkate alındığında birçok farklı maliyet fonksiyonu oluşturulabilir. Ancak en uygun çözüme en yakın ve kısa sürede götürebilen uygunluk fonksiyonu tercih edilmelidir. Uygunluk fonksiyonu Denklem 4.11'de gösterildiği şekilde hesaplanabilmektedir.

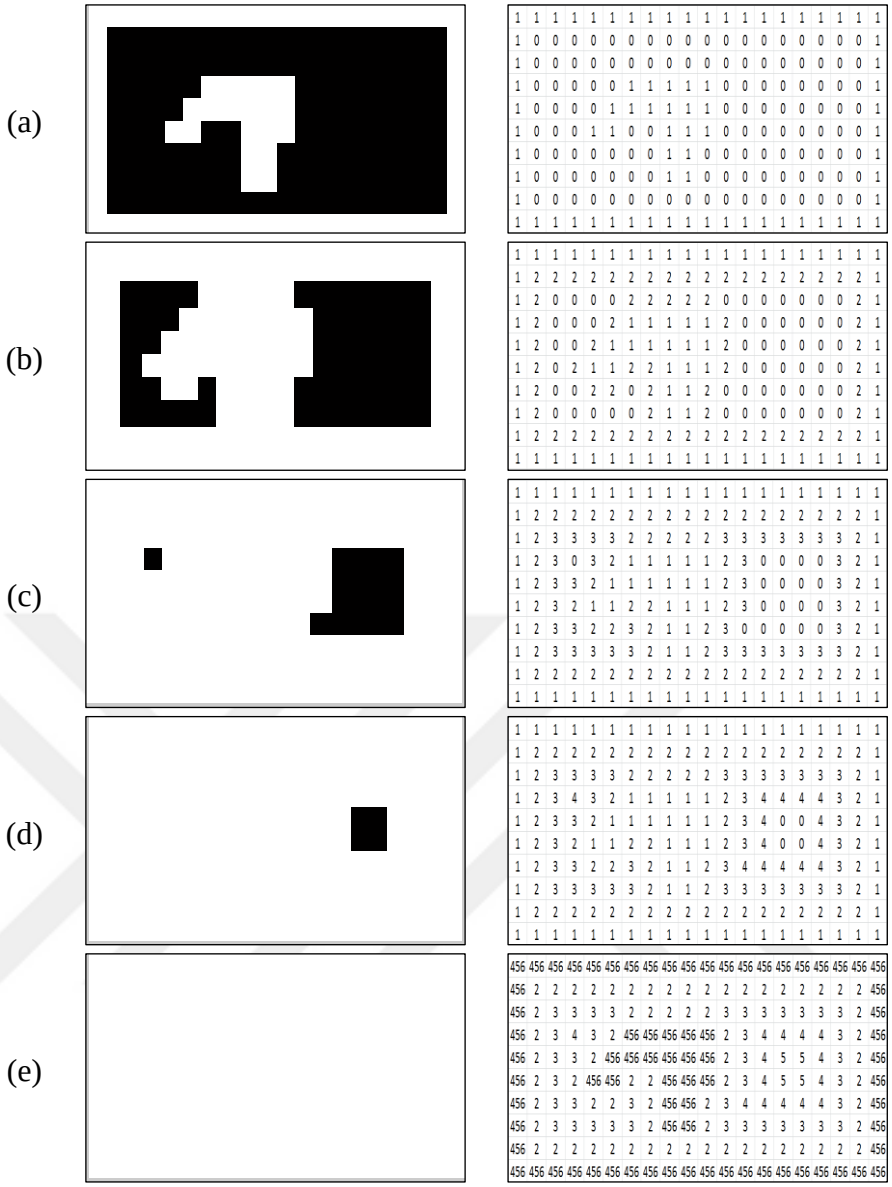
$$Maliyet = A * \sum_{i,j}^{N} uzaklik(P_i, P_j) + B * \sum_{i}^{N} sınıruzaklik(P_i) + C * \sum_{i,j}^{N} çakışma(P_i, P_j) + D * \sum_{i}^{N} taşma(P_i) + E * \sum_{i}^{N} uygunluk(P_i) \quad (4.11)$$

Denklem 4.11'de P, yerleşimi yapılan çokgenleri temsil etmektedir. Denklem 4.11 incelendiğinde, yerleşimi yapılan çokgenlerin, birbirlerinden uzak noktalara yerleştirilmesi, materyalin sınırlarından uzağa yerleştirilmesi, çokgenlerin birbirleriyle çakışma içinde olması, çokgenlerin materyalden dışarıya taşması ve uygun olmayan açılarda yerleştirilmesi gibi durumların önüne geçildiği görülmektedir. Denklem 4.11'de her bir ifadenin başında bulunan katsayılar ise her bir ifadenin önem derecesini belirtmekte kullanılmaktadır. Örneğin, çokgenlerin, birbirinden uzak noktalara yerleştirilmesi bazı durumlarda kabul edilebilir ancak, çokgenlerin birbirleriyle çakışma içinde olması veya materyal sınırından taşması hiçbir zaman kabul edilememektedir. Bu nedenle C ve D katsayılarının diğer katsayılardan büyük seçilerek, çakışma ve taşma durumlarında maliyeti hızlı bir şekilde yükseltmesi sağlanmaktadır.

Bu tez çalışması boyunca birçok maliyet fonksiyonu ile çalışılmıştır. Problemin çözümü için en iyi sonuçları veren uygunluk fonksiyonlarının gelişim aşamaları aşağıda anlatılmıştır.

4.1.2.1. Morfolojik işlemler ile maliyet hesabı

Bu tez çalışmasının ilk aşamalarında piksel yöntemi ile çalışıldığı yukarıdaki bölümlerde anlatılmıştı. Piksel yöntemi ile problemin çözümü istenilen düzeye gelince çokgenlerle çalışılmaya geçilmiştir. Dolayısıyla, tezin ilk dönemlerinde kullanılan uygunluk fonksiyonu, kalıpların birer resim matrisi olarak temsil edilmesi (piksel yöntemi) esasına göre tanımlanmıştır. Morfolojik işlemler ile tanımlanan uygunluk fonksiyonu aşağıda Şekil 4.10 üzerinden anlatılmıştır.



Şekil 4.10. Maliyet hesabı için matrisin hazırlanması

Şekil 4.10.a'da ortasında daha önceden başka bir kalıbın yerleşmiş olduğu temsili bir resim görülmektedir. Morfolojik işlemlerden genişletme işlemi adım adım uygulanmakta ve her adımda yeni genişleyen pikseller tespit edilerek adım sayısı piksel değeri olarak atanmaktadır. Böylece materyalin sınırlarından ve daha önceden yerleşimi yapılan kalıplardan uzaklaştıkça maliyetin artması ve yeni gelen kalıpların sınıra ya da daha önceden yerleşimi yapılan kalıplara yakın bir şekilde yerleşiminin yapılması sağlanmaktadır. Tüm resim genişletme işlemi ile tamamlanıncaya kadar bu işlem devam etmektedir. İşlem bittiğinde materyalin sınırları olan ve önceden yerleşmiş kalıpların bulunduğu yerlere, maliyeti çok yükseltecek büyük bir sayı atanmaktadır. Böylece yerleşimi yapılan kalıp, materyalden taşma veya başka bir

kalıpla çakışması durumuna denk geldiğinde, maliyet bir anda yükselecek ve o bölgelerden kaçınılacaktır. Şekil 4.11'da maliyet hesabı anlatılmıştır.

456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456
456	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	456
456	2	3	3	3	3	2	2	2	2	2	3	3	3	3	3	3	3	2	456
456	2	3	4	3	2	456	456	456	456	456	2	3	4	4	4	4	3	2	456
456	2	3	3	2	456	456	456	456	456	456	2	3	4	5	5	4	3	2	456
456	2	3	2	456	456	2	2	456	456	456	2	3	4	5	5	4	3	2	456
456	2	3	3	2	2	3	2	456	456	2	3	4	4	4	4	4	3	2	456
456	2	3	3	3	3	3	2	456	456	2	3	3	3	3	3	3	3	2	456
456	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	456
456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456

(a) 68 Maliyetli yerleşim

456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456
456	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	456
456	2	3	3	3	3	2	2	2	2	2	3	3	3	3	3	3	3	2	456
456	2	3	4	3	2	456	456	456	456	456	2	3	4	4	4	4	3	2	456
456	2	3	3	2	456	456	456	456	456	456	2	3	4	5	5	4	3	2	456
456	2	3	2	456	456	2	2	456	456	456	2	3	4	5	5	4	3	2	456
456	2	3	3	2	2	3	2	456	456	2	3	4	4	4	4	4	3	2	456
456	2	3	3	3	3	3	2	456	456	2	3	3	3	3	3	3	3	2	456
456	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	456
456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456

(b) 2761 Maliyetli yerleşim

456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456
456	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	456
456	2	3	3	3	3	2	2	2	2	2	3	3	3	3	3	3	3	2	456
456	2	3	4	3	2	456	456	456	456	456	2	3	4	4	4	4	3	2	456
456	2	3	3	2	456	456	456	456	456	456	2	3	4	5	5	4	3	2	456
456	2	3	2	456	456	2	2	456	456	456	2	3	4	5	5	4	3	2	456
456	2	3	3	2	2	3	2	456	456	2	3	4	4	4	4	4	3	2	456
456	2	3	3	3	3	3	2	456	456	2	3	3	3	3	3	3	3	2	456
456	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	456
456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456

(c) 41 Maliyetli yerleşim

Şekil 4.11. Örnek maliyet hesaplamaları

Şekil 4.11'de temsili olarak kırmızı çizgi ile sınırları belirlenen ve içinin dolu olarak kabul edildiği bir dikdörtgenin, yerleşim alternatifleri gösterilmiştir. Maliyet hesabı

yapılırken, yerleşimi yapılan kalıbın altında kalan tüm piksellerin değerleri toplamı maliyet olarak atanmaktadır. Şekil 4.11.b'de görüldüğü gibi önceden yerleşim yapılmış piksele denk geldiği anda maliyet çok yüksek çıkmakta ve o bölgeden kaçınılmaktadır. En uygun yerleşim olan Şekil 4.11.c optimizasyon yöntemi ile birkaç adımda hızlı bir şekilde bulunabilmektedir. Maliyet hesabı için kullanılan bu yöntem Denklem 4.11'de verilen maliyet hesaplama fonksiyonunun yaptığından daha fazlasını çok daha hızlı sonuç verecek şekilde yerine getirebilmektedir. Sadece matematiksel olarak toplama işlemi kullanılarak çok sade bir şekilde hesaplama yapılabilmektedir.

4.1.2.2. Mesafe dönüşüm yöntemi (Distance Transform) ile maliyet hesaplama

Morfolojik işlemler kullanılarak her ne kadar en uygun çözüm çok hızlı bir şekilde bulunabilse de resmin çözünürlüğü arttıkça işlem süresi çözünürlükle doğru orantılı olarak artmaya devam etmektedir. Bu sorunun üstesinden gelmek için araştırmalar yapılmış ve literatürde Distance Transform olarak adlandırılan mesafe dönüşüm yöntemi (MD) kullanılmıştır. MD yöntemi; pikseller arasındaki uzaklıkların hesaplanması temeline dayanmaktadır. Referans olarak belirlenen piksel ya da pikseller temel alınarak geriye kalan piksellerin, referans piksellere olan uzaklıkları hesaplanarak yeni piksel değeri olarak atanması yapılmaktadır. Uzaklık değerleri ondalıklı olarak çıkabilmektedir. Bu değerler tam sayıya yuvarlanabilmektedir. Ancak maliyet hesaplamasında ondalıklı olup olmamasının herhangi bir avantajı ya da dezavantajı olmamaktadır. MD yönteminde farklı şekillerde hesaplamalar yapılabilmektedir. Bu tezde yapılan çalışmalarda Öklid uzaklıkları temel alınarak işlem yapılmıştır.

MD yöntemi, morfolojik işlemler kullanılarak yapılan maliyet hesabı ile aynı sonucu vermektedir. Ayrıca morfolojik işlemlerle yüksek çözünürlükte uzun hesaplama zamanı gerekmesine karşılık, MD yönteminde çözünürlükten bağımsız olarak hesaplama süresi neredeyse fark yok denecek kadar azdır. Durum böyle olunca, morfolojik işlemlerle yapılan sonuçlara çok daha hızlı bir şekilde ulaşmak mümkün olmaktadır. Şekil 4.12'de MD yöntemi ile elde edilen örnek sonuç matrisleri gösterilmiştir.

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1
1	0	0	0	1	1	0	0	1	1	1	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

(a) Başlangıç durumu

456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456
456	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	456
456	1	2	2	2	1,4	1	1	1	1	1	1,4	2	2	2	2	2	2	1	456
456	1	2	2,2	1,4	1	456	456	456	456	456	1	2	3	3	3	3	2	1	456
456	1	2	1,4	1	456	456	456	456	456	456	1	2	3	4	4	3	2	1	456
456	1	2	1	456	456	456	456	456	456	456	1	2	3	4	4	3	2	1	456
456	1	2	1,4	1	1	1,4	1	456	456	1	1,4	2,2	3	3	3	3	2	1	456
456	1	2	2	2	2	2	1	456	456	1	2	2	2	2	2	2	2	1	456
456	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	456
456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456	456

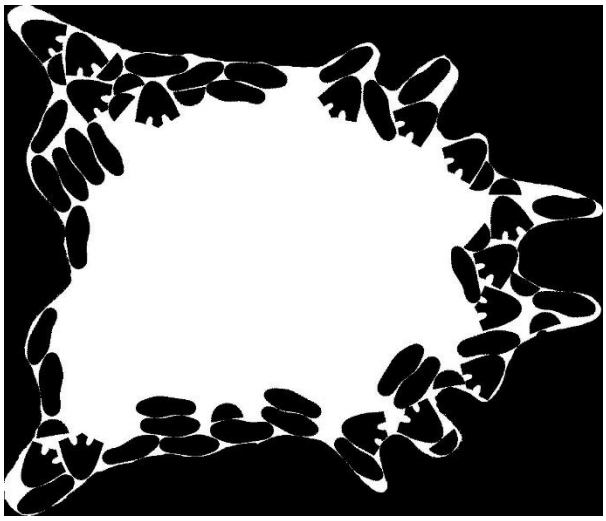
(b) MD yöntemi ile sonuç durumu

Şekil 4.12. MD yöntemi ile başlangıç ve son durumlar

Şekil 4.11.a ve Şekil 4.12.a'da verilen aynı başlangıç durumları ve Şekil 13'te verilen örnek bir yerleşim için genişletme işlemi ve MD yöntemi hesaplama zamanı olarak karşılaştırılmış ve Çizelge 4.2'de verilen sonuçlar gözlemlenmiştir. Hesaplama işlemleri Intel Core2Quad 2.66 GHz ve 3GB RAM olan masaüstü bir bilgisayarda gerçekleştirilmiştir.

Çizelge 4.2. Genişletme işlemi ve MD yöntemi zaman karşılaştırması

Şekiller	Şekil 4.10.a ve Şekil 4.10.a	Şekil 4.12
Çözünürlük	10x20	959x1123
Morfolojik İşlem	0.009515 sn	6.493558 sn
MD Yöntemi	0.001192 sn	0.117970 sn



Şekil 4.13. Örnek bir yerleşim görüntüsü

Bu yöntemle morfolojik işlem kullanılarak yapılan işlemin aynısı yapılabilmektedir. Çizelge 4.2'den de görüldüğü üzere düşük çözünürlükte hesaplama zamanı açısından çok fazla bir fark olmamasına rağmen çözünürlük artıkça mesafe dönüşüm yönteminin çok daha performanslı olduğu anlaşılmaktadır.

Piksel yöntemi ile çalışmanın yukarıda bahsedilen dezavantajlarının yanında avantajları da bulunmaktadır. Matematiksel işlemlerle çok karmaşık ve zaman alıcı işlemler, piksel yöntemi ile çalışıldığında çok daha kısa sürede ve çok daha basit bir şekilde yapılabilmektedir. Örnek olarak en uygun yerleşim yeri tespit edilirken;

1. Çokgenler arasındaki çakışmanın tespiti
2. Çokgenler arasındaki mesafelerin tespit edilerek mümkün olduğunca minimize edilmesi
3. En uygun açının hesaplanması vb.

gibi işlemler matematiksel olarak çok fazla maliyetli olmasına karşın, piksel yöntemi ile çalışıldığında çok basit bir şekilde gerçekleştirilebilmektedir.

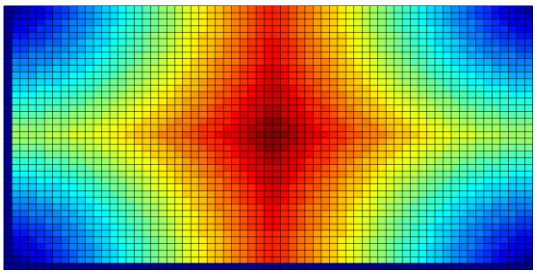
Piksel yöntemi ile çalışmanın bu avantajından yola çıkarak, çokgenler ile temsil yöntemi uygulanırken, piksel mantığına benzer şekilde çalışılmıştır. Her bir X-Y koordinat değerine karşılık gelen bir Z_{XxY} maliyet matrisi tanımlanmıştır. Böylece maliyet hesabı yapılırken; çokgenlerin birbirlerine olan uzaklıkları, materyalin sınırlarına olan uzaklıkları, çakışmaların hesabı vs. gibi matematiksel işlemlerini basit bir şekilde yapılabilmesi sağlanmıştır.

Maliyet hesaplaması yapılırken yukarıda bahsedildiği üzere, X-Y koordinat sistemi bir ızgara (grid) gibi düşünülerek her bir nokta için maliyet değerlerini tutan bir Z matrisi tanımlanmıştır. Mesafe Dönüşüm (Distance Transform) yöntemi maliyet hesabında kullanılmıştır (Sato vd., 2015). Mesafe Dönüşüm yöntemi resimler üzerinde uygulanan bir yöntem olduğu için doğrudan çokgenlere uygulanamamıştır. Denklem 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7 ve 4.8'de verilen matematiksel ifadeler yardımıyla X-Y koordinat düzlemindeki her bir noktanın kendisine en yakın olan materyal sınırına ve önceden yerleşmiş olan çokgenlere olan uzaklıkları hesaplanmıştır. Hesaplanmış bu uzaklıklar küçükten büyüğe doğru sıralanarak en küçük mesafe Z matrisindeki ilgili noktanın maliyeti olarak tanımlanmıştır. Ayrıca her bir çokgen yerleşimi yapıldığında, çokgenin içinde kalan tüm noktalara önceden belirlenen bir maksimum değer ataması yapılarak, optimizasyon algoritmasının daha önceden yerleşim yapılan bölgelerden kaçınması sağlanmıştır. Yani en uygun noktalar aranırken, herhangi bir çakışma olursa, daha önceden farklı bir çokgenin yerleştiği noktanın maliyetleri yüksek olduğundan çakışmaların önüne geçilmiş olacaktır. Aynı şekilde materyal sınırlarının dışı ve varsa deforme olmuş bölgelerin maliyeti de maksimize edilerek materyal sınırları dışına çıkılması ve deforme olmuş bölgelerin üzerine yerleşim yapılması engellenmiş olmaktadır. Şekil 4.14'te dikdörtgen şekle sahip bir materyal için örnek başlangıç Z maliyet matrisi verilmiştir.

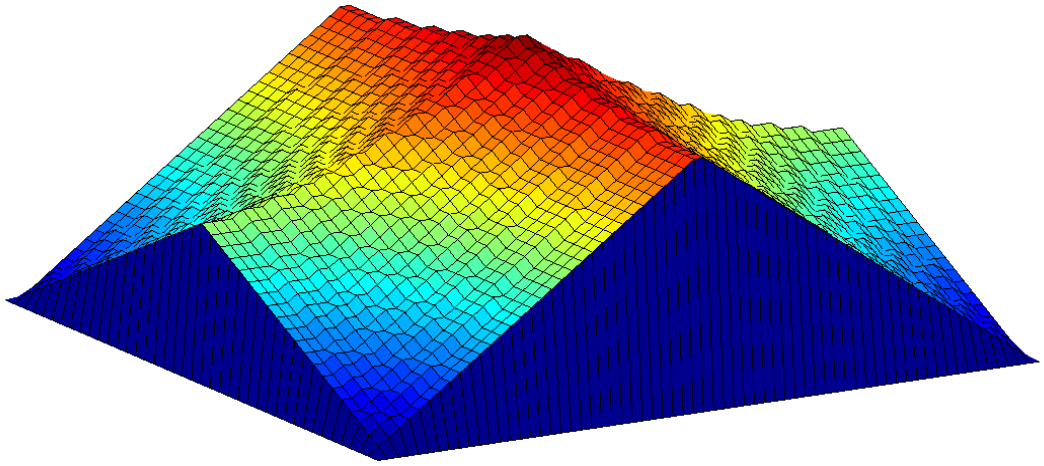
Şekil 4.14'te her bir noktanın, köşelere olan uzaklıkları dikkate alınarak hesaplanmış Z matrisi gösterilmiştir. Bu durumda yerleşim işlemine materyalin köşe noktalarından başlanarak yerleşim işleminin yapılması sağlanmıştır. Özellikle Şekil 4.14.c'de üç boyutlu olarak matris gösterilmiştir. Üç boyutlu gösterilim incelendiğinde, yeni yerleşecek olan bir kalıbın, çok basit bir şekilde materyalin neresinden bırakılırsa bırakılsın, doğrudan köşelere yerleşeceği görülmektedir. Şekilden de anlaşıldığı gibi tezde kullanılan maliyet fonksiyonu çok hızlı olarak sonuca götürmektedir.

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	2	4	6	8	10	12	14	14	12	10	8	6	4	2	0
0	4	6	8	8	10	12	14	14	12	10	8	8	6	4	0
0	6	8	8	10	12	14	16	16	14	12	10	8	8	6	0
0	8	8	10	12	12	14	16	16	14	12	12	10	8	8	0
0	10	10	12	12	14	16	18	18	16	14	12	12	10	10	0
0	8	8	10	12	12	14	16	16	14	12	12	10	8	8	0
0	6	8	8	10	12	14	16	16	14	12	10	8	8	6	0
0	4	6	8	8	10	12	14	14	12	10	8	8	6	4	0
0	2	4	6	8	10	12	14	14	12	10	8	6	4	2	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

(a)



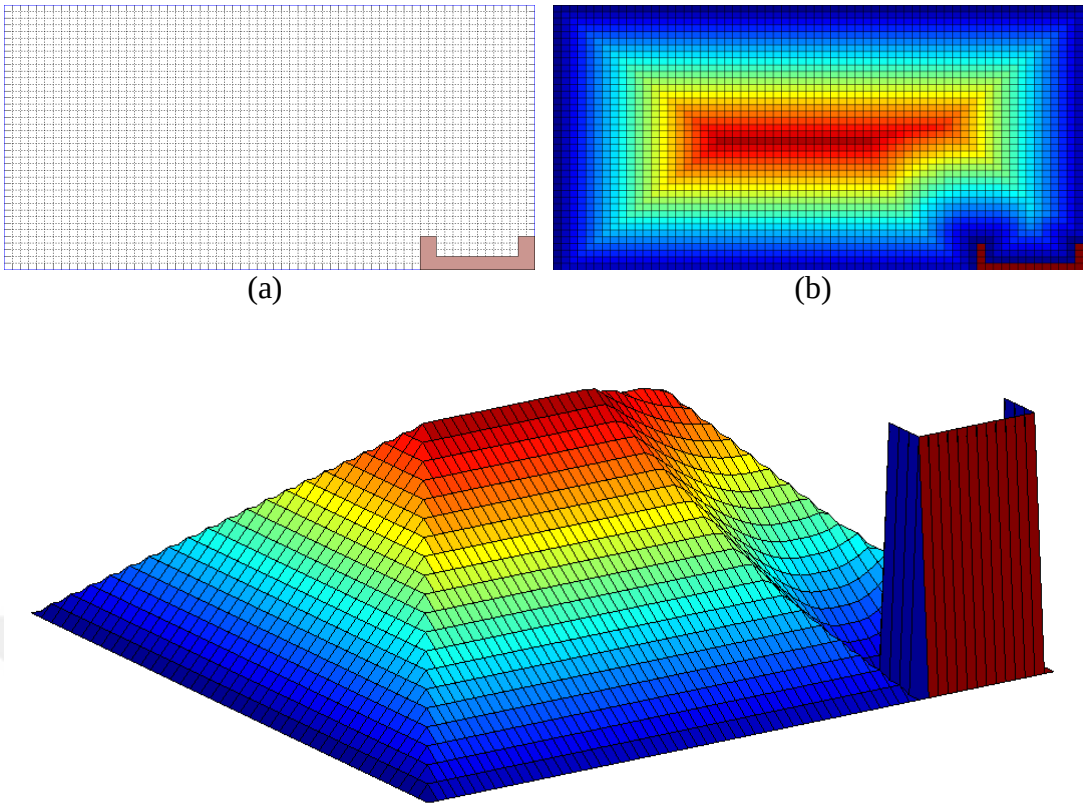
(b)



(c)

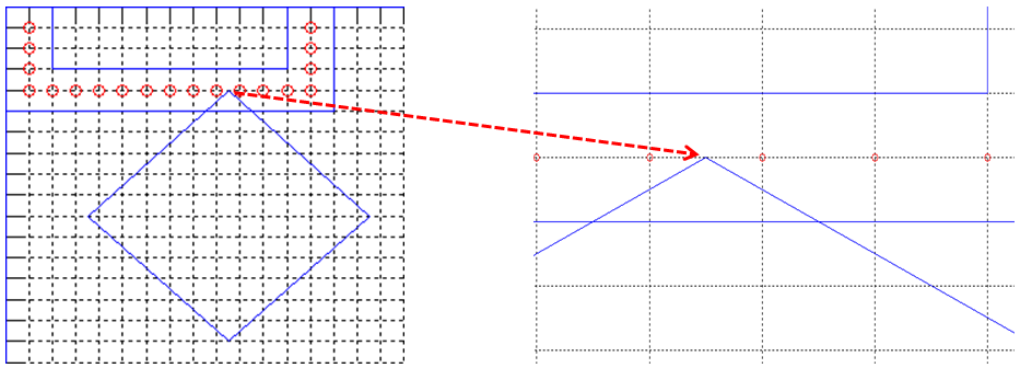
Şekil 4.14. 11x16 boyutlarında örnek *Z* başlangıç maliyet matrisi (a) Maliyet değerleri (b) Yukarıdan yüzey görüntüsü (c) Üç boyutlu görünüm

Ancak bazı durumlarda, materyalin köşe noktalarını değil, tüm sınırlarını referans alarak maliyet matrisi hesaplaması gerekmektedir. Özellikle düzensiz bir şekle sahip materyal kullanılıyorsa, materyalin sınırlarını referans almak gerekmektedir. Şekil 4.15'te her bir noktanın materyal sınırlarına ve daha önceden yerleşimi yapılmış olan bir kalıba olan en yakın uzaklıkları hesaplanarak elde edilen *Z* matrisi gösterilmiştir. Şekil 4.15.a'da sağ alt köşeye önceden yerleşmiş bir kalıp görülmektedir. Yeni yerleşecek olan kalıpların, önceden yerleşen kalıplarla çakışmalarını önlemek için, önceden yerleşen kalıpların bulunduğu koordinatların maliyet değeri yüksek bir değer olarak seçilmektedir. Böylece, yeni gelecek olan bir kalıp, çakışma veya taşma olmaksızın, materyal üzerine yerleşebileceği en kötü konuma yerleşse bile, her hangi bir çakışma ya da taşma durumundaki maliyetten daha düşük maliyete sahip olacaktır.



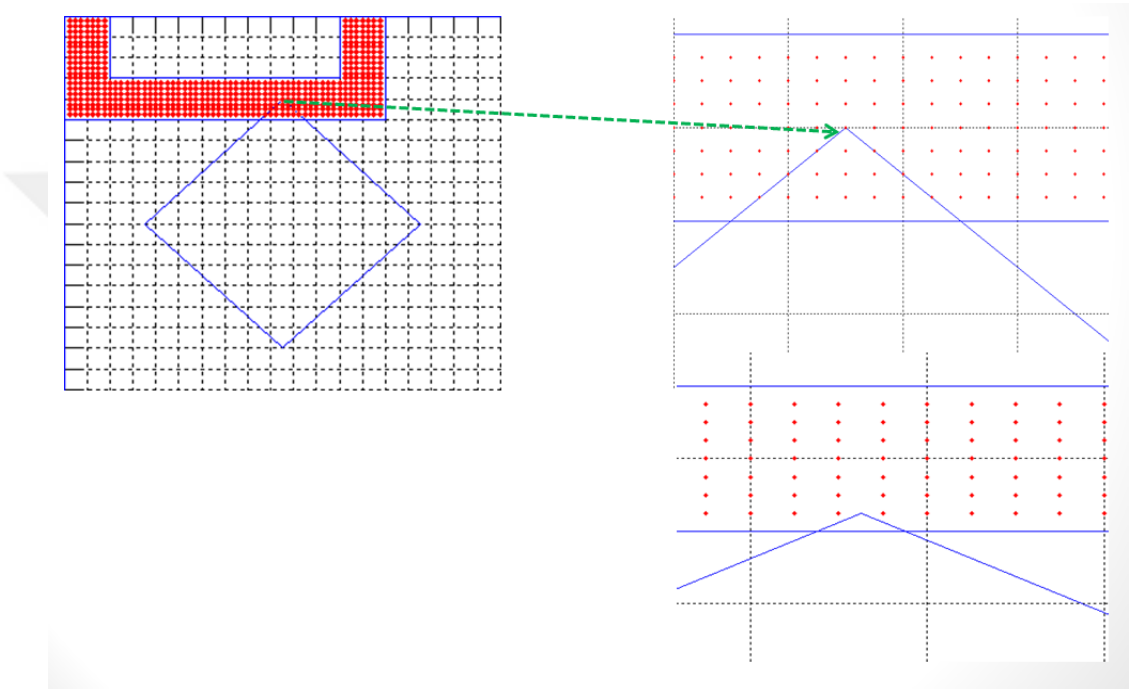
Şekil 4.15. Önceden çokgen yerleşimi yapılmış materyalin *Z* maliyet matrisi (a) Çokgenin yerleşimi (b) Yukarıdan yüzey görüntüsü (c) Üç boyutlu görünüm

Bu yöntemle çalışmalar yapıldığında, tıpkı piksel yöntemi ile temsil edildiğinde olduğu gibi *Z* matrisinin çözünürlüğünün büyük önem arz ettiği anlaşılmıştır. *Z* matrisi bir birim çözünürlüğe sahip olduğunda bazı durumlarda çakışmaların tespit edilemediği görülmüştür. Şekil 4.16'da bu durum gösterilmiştir.



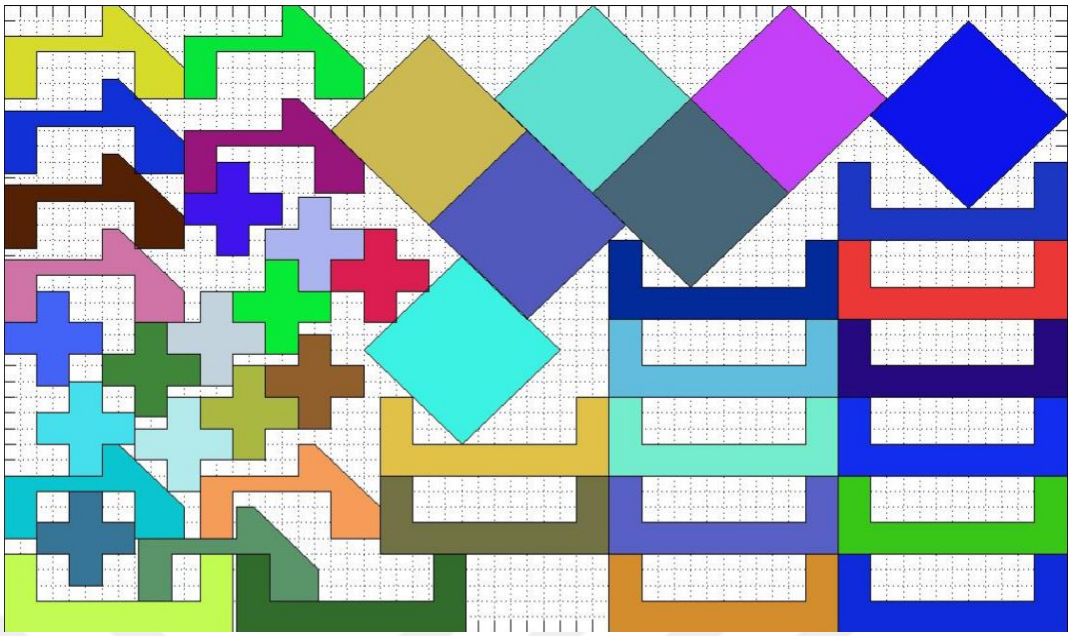
Şekil 4.16. Bir Birim Çözünürlüklü *Z* Maliyet Matrisi Görüntüsü

Şekil 4.16'da görüldüğü üzere bir birim çözünürlük ile çalışıldığında, yukarıdaki yerleşimde çakışma olmasına rağmen baklava dilimi olan çokgenin içinde kalan bölgeye değeri maksimum olan hiçbir nokta denk gelmemektedir. Bu nedenle çakışma tespit edilememekte ve yanlış bir yerleşim işlemi yapılmaktadır. Bu sorunu çözmek için Z maliyet matrisinin çözünürlüğünün artırılması denenmiştir. Şekil 4.17'de dört birim hassasiyete sahip olan bir yerleşim örneği gösterilmektedir. Yani her bir birimlik kare 4x4 adet daha küçük noktaya bölünerek çakışmanın tespit oranı artırılmıştır.



Şekil 4.17. Dört Birim Çözünürlüklü Z Maliyet Matrisi Görüntüsü

Şekil 4.16'da verilen aynı yerleşim noktaları, dört birim çözünürlüklü Z maliyet matrisi ile yerleştiğinde, Şekil 4.17'de gösterildiği gibi çakışma bölgesi içine dokuz adet değeri maksimum olan nokta denk gelmektedir. Böyle olunca yerleşimin yapıldığı noktaların maliyeti maksimize olmakta ve o noktalardan kaçınılmaktadır. Şekil 4.17'nin sağ alt tarafında maliyeti maksimum olmayan ancak çakışmanın olduğu farklı bir görüntü verilmiştir. Bu nedenle olabildiğince çözünürlüğü yüksek Z maliyet matrisi ile çalışmak sıfıra yakın bir çakışma ile sonucu elde etmemizi sağlayacaktır. 10 birim çözünürlüğe sahip Z maliyet matrisi ile elde edilen yerleşim örneği Şekil 4.18'de gösterilmiştir.



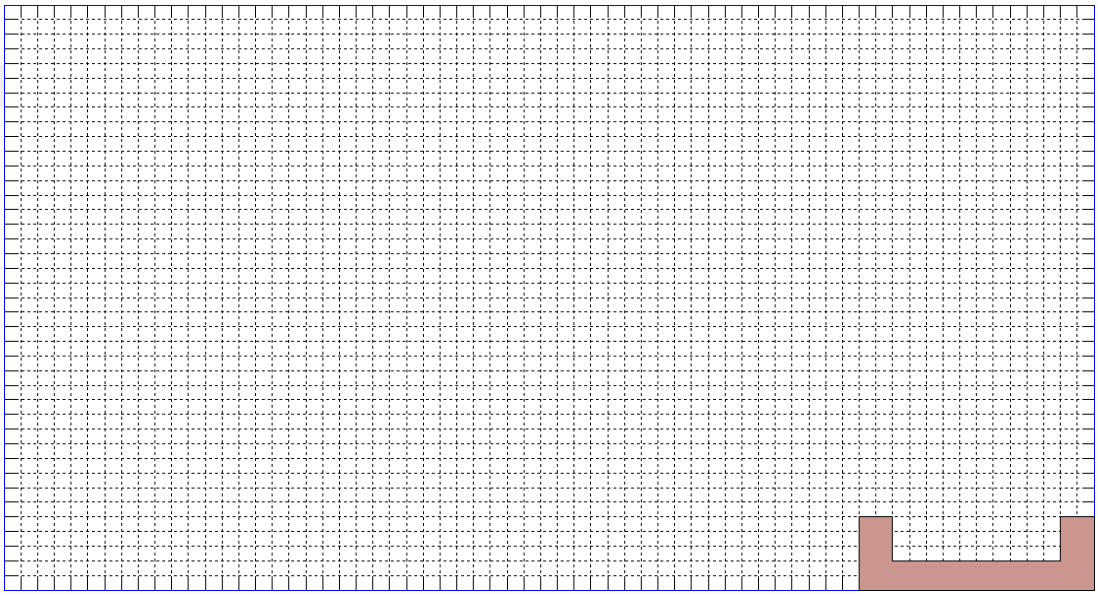
Şekil 4.18. 10 birim çözünürlüklü Z maliyet matrisi ile yerleşim görüntüsü

Şekil 4.18'de çözünürlüğü yüksek olan bir matrisle hesaplamalar yapılarak yerleşim işlemi gerçekleştirilmiştir. Çözünürlük yüksek olduğu için, Şekil 4.17'den anlaşılacağı üzere, çakışmalar sıfıra yakındır. Şekil 4.18 dikkatlice incelendiğinde küçük çakışma bölgelerinin olduğu anlaşılmaktadır.

Çözünürlüğün artırılması doğruluk oranını artırdığı kadar hesaplama zamanını da artırmaktadır. Şekil 4.18'de gösterilen yerleşim, yaklaşık 4.5 saat sonunda özellikleri 2,66GHz, 3GB RAM'i olan 32 bitlik bir masaüstü bilgisayardan alınmıştır.

Yukarıda bahsedilen yöntemle, maliyet fonksiyonu çakışmayı minimize etmeyi amaçlamakta ancak çakışmanın olmamasını garanti etmemektedir. Ayrıca maliyet hesaplamasında kullanılan çözünürlük hız problemlerine yol açmakta ve düşük çözünürlükte çakışmaların meydana gelmesi, yüksek çözünürlükte ise algoritmanın yavaşlaması gibi sorunlarla karşılaşılmaktadır. Yukarıda anlatılan yöntemle, maliyet hesabı yapılırken materyalin tüm noktaları, belli bir çözünürlükte koordinat düzlemine dönüştürülmekte ve her bir koordinatın MD yöntemi ile maliyeti hesaplanmaktadır. Ancak bu işlem yapılırken, yerleşim sırasında hiç kullanılmayan ve yaklaşık materyalin %95'ini kapsayan bölgeler gereksiz yere hesaplanmakta ve işlem zamanı artmaktadır. Yani sadece, kalıbın yerleşeceği aday koordinatların maliyet değerleri gerekli iken, tüm materyalin, tüm koordinatlarının maliyet değerlerinin hesaplanmasına gerek yoktur.

Gereksiz işlemleri azaltmak için, maliyet fonksiyonunda bir takım değişiklikler yapılarak hızlanma sağlanmıştır. Ayrıca, çakışan bölgelerin alanları da maliyet hesabına katılarak çakışmaların önüne geçilmiştir. Geliştirilen uygunluk fonksiyonunda tüm materyalin değil, sadece yerleşimi yapılan kalıbın sınırları boyunca, tüm kenarları belli bir hassasiyet ile noktalara bölünerek, bu noktaların MD işlemine göre maliyeti dikkate alınmaktadır. Önceki kullanılan maliyet hesabı ile gereksiz hesaplamalardan kurtularak geliştirilen maliyet hesaplama yöntemleri Şekil 4.19 ve Şekil 4.20'de gösterilmiştir.

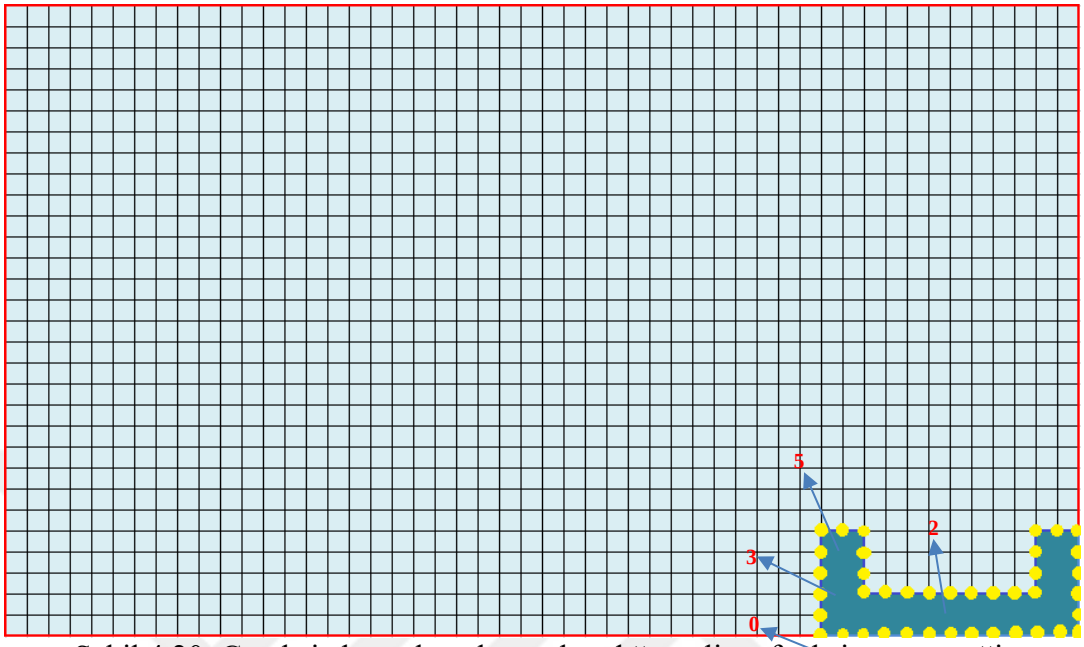


Şekil 4.19. Dört birim çözünürlüklü önceki dönemde kullanılan Z maliyet matrisi görüntüsü

Şekil 4.19'da görülen zeminin tüm koordinatlarının MD yöntemine göre maliyet değeri her yerleşim işleminden önce hesaplanmaktadır. Oysa yerleşimi yapılan kalıp küçük ve hesabı yapılan birçok koordinatın gereksiz olduğu görülmektedir. Çünkü maliyete sadece kalıbın altında kalan koordinatlar dâhil edilmektedir. Diğer koordinatların hiçbir etkisi olmamaktadır.

Şekil 4.20'de görüldüğü gibi, yerleşimi yapılmakta olan kalıbın tüm kenarları belli bir hassasiyetle aralıklara bölünmektedir. Daha sonra aday yerleşim noktalarına yerleştiğinde, tüm kenarlarında bulunan koordinatların maliyetleri MD yöntemi ile

hesaplanmaktadır. En yakın materyal sınırı veya en yakın önceden yerleşmiş kalıplar temel alınarak yerleşim işleminin maliyeti hesaplanmaktadır.



Şekil 4.20. Gereksiz hesaplamaların olmadığı maliyet fonksiyonu örneği

Şekil 4.20'de yerleşimi yapılan bir kalıbın sarı noktalarla tüm sınır boyunca koordinatları gösterilmiştir. Örnek olarak dört farklı koordinat değerinin maliyeti gösterilmiştir. Örnek noktaların en yakın olduğu yer materyal sınırları olduğu için maliyet hesaplarken en yakın materyal sınırı dikkate alınmıştır. Bu yöntemde sınır boyunca koordinat hesabı yapılırken hassasiyet artırılabilir veya düşürülebilir. Önceki maliyet hesaplamasının aksine burada yüksek doğruluk için yüksek çözünürlük ihtiyacı ortadan kaldırılmış olmaktadır.

Ayrıca bu yöntemle çakışma olup olmadığı da kontrol edilerek çakışmaların önüne geçilmiş olunur. Çünkü herhangi bir çakışma durumunda, çokgenin en az bir köşe noktası başka bir çokgenin içinde olacak ve MD yöntemi ile o noktanın maliyeti maksimize edilecektir. Geliştirilen bu maliyet hesaplamasıyla, gereksiz koordinat maliyeti hesaplamasının da önüne geçilerek algoritmada hızlanma sağlanmıştır.

Deri ayakkabı kesim optimizasyonu için gerekli olan uygunluk fonksiyonu istenilen düzeyde başarı elde etmiştir. Optimizasyon probleminin çözüm süreci tamamlandığından, kullanıcı arayüzü geliştirme sürecine geçilmiştir.

4.2. Geliştirilen Yazılım

Bu tezde, çalışmalara ilk olarak MATLAB programı ile başlanmıştır. Özellikle ilk olarak optimizasyon probleminin geliştirilmesi aşamasında, piksel yöntemi ve resim matrisleri ile çalışıldığı için MATLAB programı kolaylıklar sağlamıştır. Ancak gerek çokgenlerle temsil yöntemine geçilmesi, gerek gelişmiş bir kullanıcı arayüzünün tasarlanması gerekmesi gibi nedenlerden dolayı, kullanıcı arayüzü Microsoft Visual Studio platformunda C# programlama dili ile yazılmıştır. Ayrıca problemin çözümünde hızın da büyük önemi olmasından dolayı MATLAB programı istenilen düzeyde ihtiyacı karşılamamıştır. Bu nedenle hesaplama işlemlerinde makine diline en yakın olan C programlama dili tabanlı olan C# programlama diline geçilmiştir.

MS Visual Studio platformu kullanıcı arayüzü tasarımında kolaylıklar sağlayan güçlü bir kod yazma editörü sunmaktadır. Bu tez çalışmasında kullanıcı arayüzü tasarımı çok önemli bir aşamadır. Çünkü tezin amacında, kullanıcı dostu kullanımı kolay olan bir otomatik yerleşim optimizasyonu yazılımının geliştirilmesi hedeflenmiştir. Tüm bu ihtiyaçlar göz önünde bulundurulduğunda, MS Visual Studio C# bu tez için çok uygun bir tercihtir.

Tezin gereksinimleri incelendiğinde, geliştirilecek olan yazılımda aşağıdaki özelliklerin olması gerekmektedir;

1. Yazılım, kalıp ve materyali vektörel formatta içe aktarabilmelidir.
2. Yerleşim işleminin sonucunu yine vektörel formatta dışa aktarabilmelidir.
3. Kullanıcı istediğinde manuel olarak dizilim işlemini gerçekleştirebilmelidir.
4. Kullanıcı bilgisayarın faresi ile kalıpları istediği konuma istediği açıyla yerleştirebilmelidir.
5. Yerleşim işlemi tamamlanınca, kesim işlemi yapan makineye doğrudan yerleşimi gönderebilmelidir.
6. Yerleşim işlemini daha sonra kesim makinesine göndermek üzere kayıt altına alıp, istediği zaman makineye gönderebilmelidir.
7. Farklı optimizasyon yöntemleri ile otomatik dizilim işlemi yapılabilmelidir.
8. Farklı optimizasyon yöntemlerinin karşılaştırılabilmesi için analiz işlemleri yapılabilmelidir.

Yukarıda sayılan tüm özellikleri içinde barındıran yazılım bu tez kapsamında geliştirilmiştir.

4.2.1. Optimizasyon yöntemlerinin kodlanması

Bu tezde önceki bölümlerde anlatılan dokuz farklı optimizasyon yöntemi kullanılmıştır. Yazılımın geliştirilmesine ilk olarak optimizasyon yöntemlerinin her biri, ayrı bir sınıf yapısı kullanılarak kodlanmıştır. Aşağıda her bir yöntem için yazılan sınıflar, kurucu metotlarının açıklanması ile ifade edilmiştir.

4.2.1.1. Yapay arı kolonisi

YAK için MS Visual Studio C#'ta YAK.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan "Solver" fonksiyonu ile optimizasyon problemi çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. YAK sınıfı toplamda 16 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.

CS: YAK yönteminde kullanılacak kolonideki arı sayısını ifade etmektedir.

limit: bu değişken optimizasyon döngüsünü kırmak için kabul edilebilir fark değerini ifade etmektedir.

maxCycle: yöntemin çalıştırılacağı maksimum döngü sayısını ifade etmektedir.

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise

problemin maksimizasyon, “false” ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer “sıfır” ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

4.2.1.2. Tavlama benzetimi

TB için MS Visual Studio C#’ta TB.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan “Solver” fonksiyonu ile optimizasyon problemi çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. TB sınıfı toplamda 16 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.

temperature: TB yönteminde kullanılacak başlangıç sıcaklık değerini ifade etmektedir.

iteration: yöntemin her bir sıcaklık değerinde çalıştırılacağı maksimum döngü sayısını ifade etmektedir.

epsilon: bu değişken optimizasyon döngüsünü kırmak için kabul edilebilir fark değerini ifade etmektedir.

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise problemin maksimizasyon, "false" ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer "sıfır" ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

4.2.1.3. Genetik algoritma

GA için MS Visual Studio C#'ta GA.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan "Solver" fonksiyonu ile optimizasyon probleminin çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. GA sınıfı toplamda 18 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.

PS: GA yönteminde kullanılacak popülasyondaki birey sayısını ifade etmektedir.

r_c: GA yöntemindeki çaprazlama oranını,

r_m: Mutasyon oranını,

r_ch: GA yöntemindeki yeni nesile aktarılacak birey oranını,

iter: Her nesilde çalıştırılacak iterasyon sayısını ifade etmektedir,

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise problemin maksimizasyon, "false" ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer "sıfır" ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

4.2.1.4. Parçacık sürü optimizasyonu

PSO için MS Visual Studio C#'ta PSO.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan "Solver" fonksiyonu ile optimizasyon probleminin çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. PSO sınıfı toplamda 16 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.
numParticles: PSO yönteminde kullanılacak popülasyondaki parçacık sayısını ifade etmektedir.

epoch: yöntemin çalıştırılacağı maksimum döngü sayısını ifade etmektedir.

target: bu değişken optimizasyon döngüsünü kırmak için kabul edilebilir fark değerini ifade etmektedir.

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise problemin maksimizasyon, "false" ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer "sıfır" ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

4.2.1.5. Sosyal örümcek algoritması

SÖA için MS Visual Studio C#'ta SÖA.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan "Solver" fonksiyonu ile optimizasyon problemi çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. SÖA sınıfı toplamda 18 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.

PS: SÖA yönteminde kullanılacak popülasyondaki birey sayısını ifade etmektedir.

r_a: SÖA yöntemindeki titreşim zayıflama oranını oranını,

p_c: SÖA yöntemindeki örümceğin rasgele yürüyüşündeki değişiklik yapma oranını,

p_m: SÖA yöntemindeki bir bitin 1 olma ihtimalini ifade etmektedir.

iter: Her nesilde çalıştırılacak iterasyon sayısını ifade etmektedir.

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise problemin maksimizasyon, "false" ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer "sıfır" ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

4.2.1.6. Gri kurt optimizasyonu

GKO için MS Visual Studio C#'ta GKO.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan "Solver" fonksiyonu ile optimizasyon problemi çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. GKO sınıfı toplamda 15 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.

PS: GKO yönteminde kullanılacak popülasyondaki birey sayısını ifade etmektedir.

iter: Her nesilde çalıştırılacak iterasyon sayısını ifade etmektedir.

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise problemin maksimizasyon, "false" ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer "sıfır" ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

4.2.1.7. Diferansiyel gelişim algoritması

DG için MS Visual Studio C#'ta DG.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan "Solver" fonksiyonu ile optimizasyon problemi çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. DG sınıfı toplamda 16 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.

PS: GKO yönteminde kullanılacak popülasyondaki birey sayısını ifade etmektedir.

iter: Her nesilde çalıştırılacak iterasyon sayısını ifade etmektedir.

target: bu değişken optimizasyon döngüsünü kırmak için kabul edilebilir fark değerini ifade etmektedir.

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise problemin maksimizasyon, "false" ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer "sıfır" ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

4.2.1.8. L-SHADE algoritması

L-SHADE için MS Visual Studio C#'ta LSHADE.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan "Solver" fonksiyonu ile optimizasyon problemi çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. LSHADE sınıfı toplamda 17 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.

PS: GKO yönteminde kullanılacak popülasyondaki birey sayısını ifade etmektedir.

iter: Her nesilde çalıştırılacak iterasyon sayısını ifade etmektedir.

target: bu değişken optimizasyon döngüsünü kırmak için kabul edilebilir fark değerini ifade etmektedir.

M: bu değişken L-SHADE algoritmasının çaprazlama ve mutasyon oranlarının tutulacağı hafızanın büyüklüğünü ifade etmektedir.

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise problemin maksimizasyon, "false" ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer "sıfır" ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

4.2.1.9. Hibrit PSO-GKO algoritması

HPSGKO için MS Visual Studio C#'ta HPSGKO.cs adında bir sınıf yazılmıştır. Yazılan sınıf bir adet kurucu metottan oluşmaktadır. Kurucu metot ile optimizasyon için gerekli olan tüm parametreler sınıfa gönderilmektedir. Son olarak sınıfta bulunan "Solver" fonksiyonu ile optimizasyon problemi çözümü yapılmakta ve geriye bulunan noktalar ve maliyet değeri döndürülmektedir. HPSGKO sınıfı toplamda 16 parametre almaktadır. Bu parametreler aşağıda belirtildiği gibi tanımlanmıştır;

D; optimizasyon probleminde optimize edilecek değişken sayısını ifade etmektedir.

PS: GKO yönteminde kullanılacak popülasyondaki birey sayısını ifade etmektedir.

iter: Her nesilde çalıştırılacak iterasyon sayısını ifade etmektedir.

target: bu değişken optimizasyon döngüsünü kırmak için kabul edilebilir fark değerini ifade etmektedir.

rang: n adet değişken için nx2 boyutlu bir float türünde değişkendir. Her bir satırı her bir değişken için minimum ve maksimum değerlerini ifade etmektedir.

minmax: Bu değişken bool türünde olup optimizasyon probleminin minimizasyon veya maksimizasyon olduğunu tanımlamak için kullanılmaktadır. Eğer "true" ise problemin maksimizasyon, "false" ise minimizasyon problemi olduğu ifade edilmektedir.

resolution: Bu değişken, parametreler için ondalık hane sayısının kaç olacağını ifade etmektedir. Eğer "sıfır" ise tam sayılarla çalışılacağı anlaşılmaktadır. Aksi halde parametreler için ondalık hassasiyet sayısını ifade etmektedir.

TempPolygon: anlık olarak yerleşim işlemine tabi tutulan kalıbı ifade etmektedir.

Polygons: Bu değişken, daha önceden materyal üzerine yerleşimi yapılmış tüm kalıpları temsil etmektedir. Maliyet hesabı yapılırken çakışmalar hesaplanmakta ve maliyet hesabında ihtiyaç duyulmaktadır.

material: Bu değişken yerleşimin yapıldığı materyali temsil etmektedir.

dir: Bu değişken yerleşimin dizilim yönünü ifade etmektedir. Sırasıyla; 0: Soldan Sağa, 1: Sağdan Sola, 2: Aşağıdan Yukarı, 3: Yukarıdan Aşağıya olarak ifade edilmektedir.

angle: Dizilim işlemindeki kalıpların izin verilen döndürme açısını ifade etmektedir.

X_Eksenmin, X_Eksenmax, Y_Eksenmin, Y_Eksenmax: Bu değişkenler yerleşimin yapıldığı materyalinin sınırlarını temsil etmektedir.

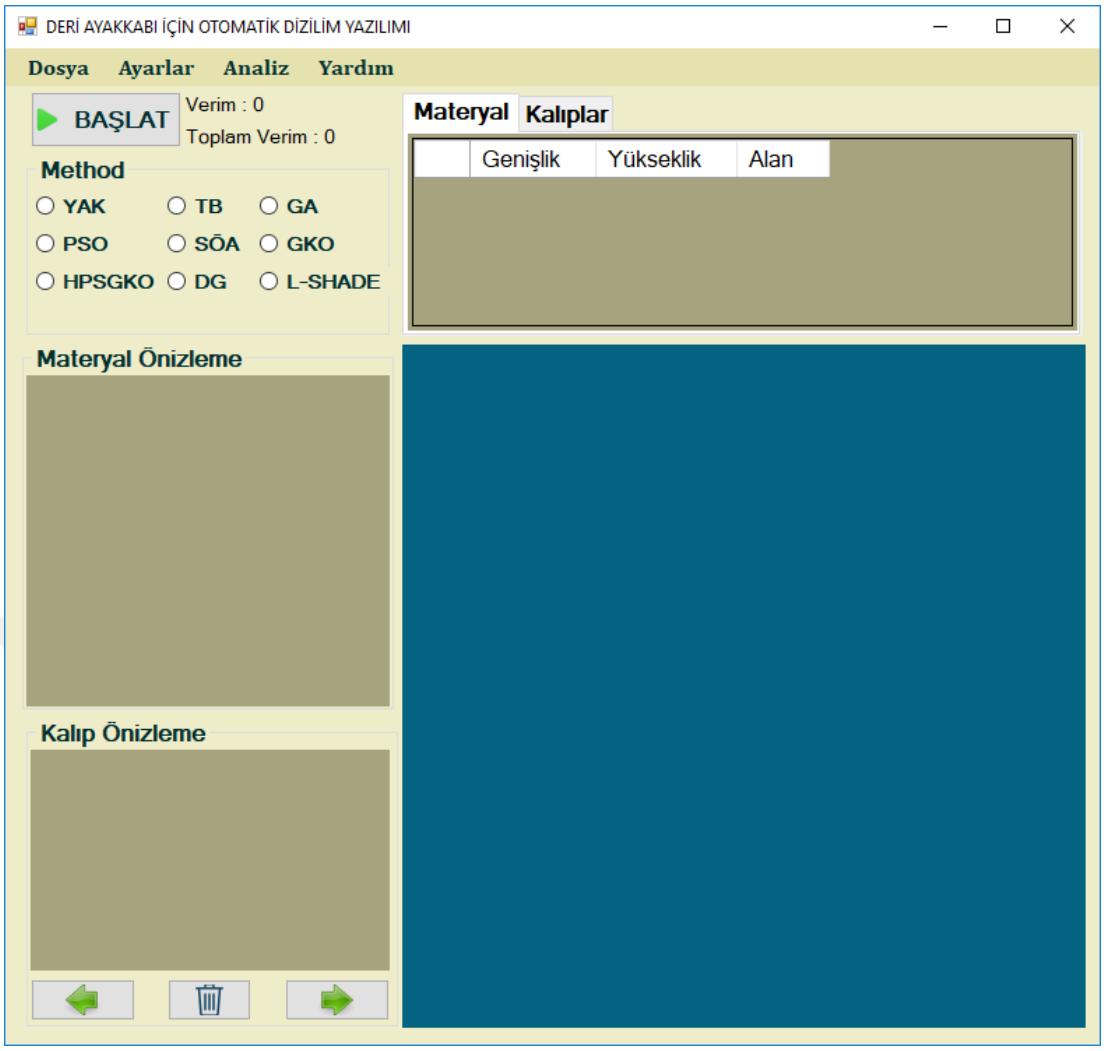
4.2.2. Maliyet Sınıfı

Tüm optimizasyon yöntemlerinde kullanılan maliyet fonksiyonu aynıdır. Problem değişmediği için maliyet fonksiyonunun da sabit olması gerekmektedir. Her bir yöntemin maliyet fonksiyonu, yöntemlerin bulunduğu sınıflar içerisine yazılabilir ancak maliyet fonksiyonunda bir değişiklik gerektiğinde dokuz yöntemin her birinde ayrı ayrı değiştirmek zorluklara neden olmaktadır. Bu zorluktan kaçınmak için maliyet işlemlerinin yapıldığı ve tek bir noktadan kontrol edilebildiği Maliyet isimli bir sınıf yazılmıştır. Tüm optimizasyon yöntemlerinde maliyet hesabı yapılırken Maliyet sınıfından bir nesne üretilerek işlemler gerçekleştirilir. Böylece tek bir noktadan kontrol sağlanabilmektedir. Farklı maliyet fonksiyonları ile hızlıca optimizasyon yöntemleri test edilebilmektedir. Maliyet hesabı yapan fonksiyonun sözde kodu aşağıda verilmiştir.

1. Başla
2. Dizilimi yapılacak olan kalıbın, dizilim yönü dikkate alınarak başlangıç noktasına olan uzaklığını hesapla
3. Dizilimi yapılacak olan kalıbın, daha önceden dizilen kalıplarla olan çakışma alanlarını hesapla
4. Dizilimi yapılacak olan kalıbın, materyal sınırından taşan alanını hesapla
5. Dizilimi yapılacak olan kalıbın, her bir noktasının kendisine en yakın olan materyal sınırına veya önceden dizilen kalıplara olan uzaklığını hesapla
6. Hesaplanan alan ve uzaklık değerlerini topla
7. Toplam değeri maliyet değeri olarak geri döndür
8. Bitir

4.3. Kullanıcı Arayüzünün Tasarlanması

Şekil 4.21'de geliştirilen programın ana giriş ekranı verilmiştir. Program temel olarak altı bölümden oluşmaktadır. Bunlar; menüler, dizilim işlemi için kullanılacak yöntemler, dizilim işleminin yapılacağı materyal, materyal üzerine dizilecek olan kalıplar, materyal ve kalıplar hakkında bilgilerin verildiği bölüm ve son olarak dizilim işleminin anlık görüntüsünün verildiği bölümdür.



Şekil 4.21. Ana Ekran Görüntüsü

4.3.1. Menüler

Programda dört adet ana menü bulunmaktadır. Menüler vasıtasıyla programa girdiler verilip çıktılar alınabilmektedir. Aynı zamanda program içinde kullanılan optimizasyon yöntemleri, optimizasyon yöntemlerinin analizi ve program ayarları yapılabilmektedir.

4.3.1.1. Dosya

Şekil 4.22'de dosya menüsünde bulunan alt menüler gösterilmiştir. Her bir alt menünün detaylı açıklaması aşağıda verilmiştir.

Şekil 4.22. Dosya Menüsü

Dizilimi Temizle: Bu menü seçimi ile programda elde edilen yerleşim işlemi temizlenerek yerleşim işleminin tekrar başa döndürülmesi sağlanmaktadır.

İçe Aktar: İçe aktar menüsü, yerleşimin yapılacağı materyal ve yerleştirilecek kalıpların DXF formatında programa aktarıldığı bölümdür. İlk olarak materyal içe daha sonra materyal üzerine yerleşecek kalıplar içe aktarılmalıdır.

Dışa Aktar: Çizim işleminin tamamlanmasından sonra elde edilen yerleşimin kesim makinesine gönderilmesi için iki farklı şekilde dışa aktarabilme özelliği kazandırılmıştır. DXF ve PLT uzantılı olarak program çıktıları alınabilmektedir. Bu tez çalışması kapsamında satın alınan ve denemeler yapılan çizim ve kesim makinesi PLT dosya uzantısı ile çalışmaktadır. Bu nedenle programa PLT uzantılı çıktı alabilme özelliği eklenmiştir. Böylece yerleşim işlemi istenildiğinde daha sonra makineye gönderilecek şekilde kaydedilip istenilen zamanda makineye gönderilebilir. Piyasada kullanılan kesim makineleri DXF formatında dosyalar kabul etmekte olduğundan program DXF formatında çıktı da verebilmektedir.

DXF Görüntüleyici: Bu menü ile daha önceden yerleşimi yapılmış ve kaydedilmiş olan DXF uzantılı dosyaların görüntülenmesi işlemi gerçekleştirilmektedir. Ayrıca görüntüleme işlemi ile yazıcıya doğrudan yerleşim işlemi gönderilebilmektedir.

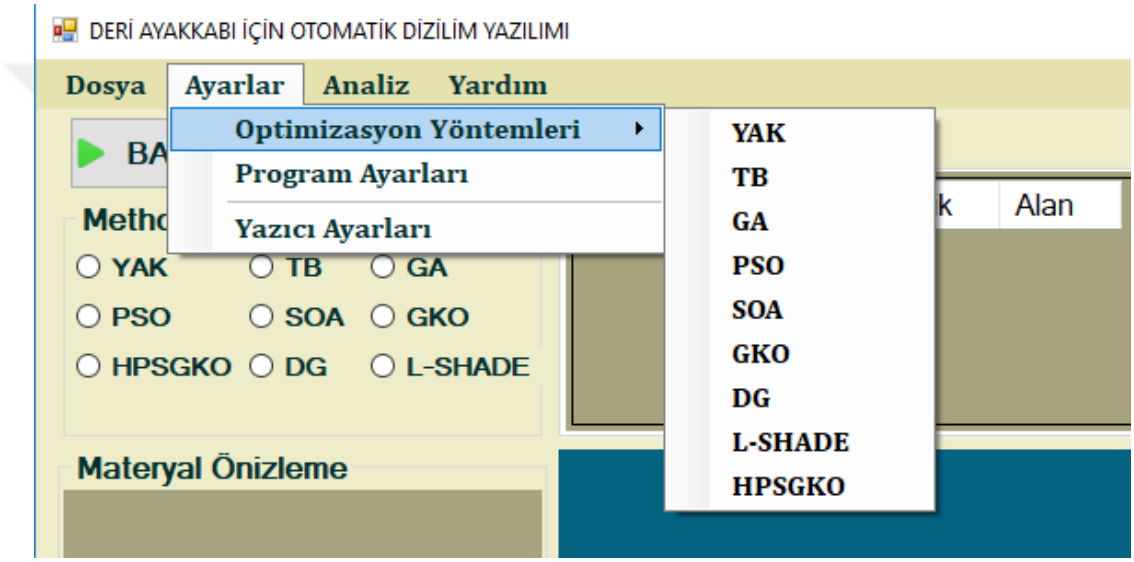
Yazdır: Yerleşim işleminin tamamlanmasından sonra istenirse yerleşim işlemi doğrudan makineye gönderilebilir. Ayarlar menüsünden makinenin bağlı olduğu seri

haberleşme portu ve bağlantı hızı ayarlanarak yerleşim işlemi doğrudan makineden çıktı alınabilmektedir.

Çıkış: Programdan tamamen çıkma işlemini gerçekleştirmektedir.

4.3.1.2. Ayarlar

Şekil 4.23'te dosya menüsünde bulunan alt menüler gösterilmiştir. Her bir alt menünün detaylı açıklaması aşağıda verilmiştir.

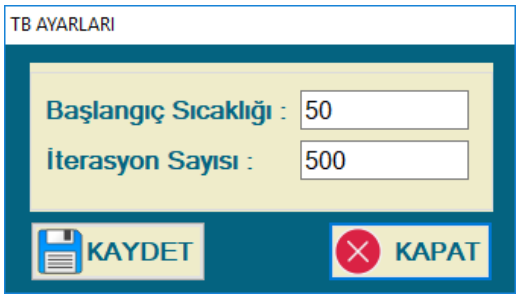


Şekil 4.23. Ayarlar Menüsü

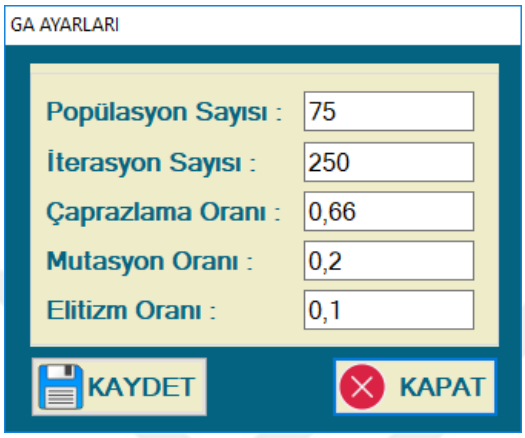
Optimizasyon Yöntemi: Bu programda yerleşim işleminin yapılabilmesi için dokuz farklı optimizasyon yöntemi kullanılmıştır. Tez çalışması olması nedeniyle farklı optimizasyon yöntemleri ile çalışılmış ve sonuçlar karşılaştırılmıştır. Program kullanılırken farklı yöntemlerin seçilmesiyle yerleşim işlemleri yapılıp sonuçlar gözlemlenebilmektedir. Şekil 4.24'te kullanılan optimizasyon yöntemlerinin gerekli ayar parametreleri görülmektedir.


YAK AYARLARI
Arı Sayısı :
100
İterasyon Sayısı :
100
KAYDET
KAPAT

(a)


TB AYARLARI
Başlangıç Sıcaklığı :
50
İterasyon Sayısı :
500
KAYDET
KAPAT

(b)


GA AYARLARI
Popülasyon Sayısı :
75
İterasyon Sayısı :
250
Çaprazlama Oranı :
0,66
Mutasyon Oranı :
0,2
Elitizm Oranı :
0,1
KAYDET
KAPAT

(c)


PSO AYARLARI
Parçacık Sayısı :
50
İterasyon Sayısı :
150
KAYDET
KAPAT

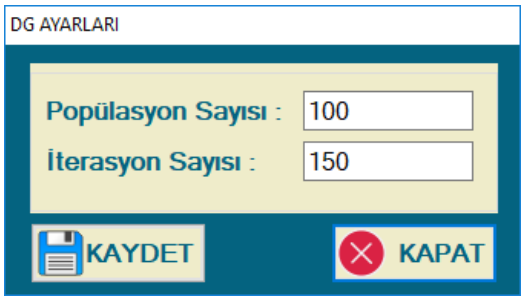
(d)


SOA AYARLARI
Popülasyon Sayısı :
25
İterasyon Sayısı :
500
r_a Değeri :
1
p_c Değeri :
0,7
p_m Değeri :
0,1
KAYDET
KAPAT

(e)


GKO AYARLARI
Popülasyon Sayısı :
50
İterasyon Sayısı :
150
KAYDET
KAPAT

(f)


DG AYARLARI
Popülasyon Sayısı :
100
İterasyon Sayısı :
150
KAYDET
KAPAT

(g)

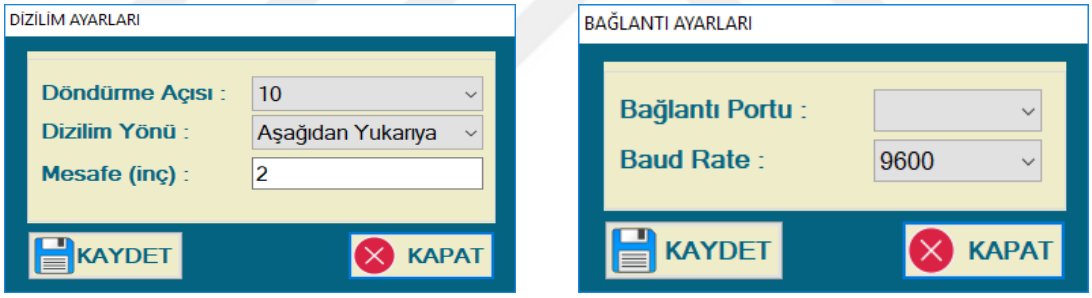

L-SHADE AYARLARI
Popülasyon Sayısı :
1000
İterasyon Sayısı :
500
Hafıza Boyutu :
10
KAYDET
KAPAT

(h)

(i)

Şekil 4.24. Optimizasyon yöntemleri ayarları (a) YAK yöntemi ayarları (b) TB yöntemi ayarları (c) GA yöntemi ayarları (d) PSO yöntemi ayarları (e) SÖA yöntemi ayarları (f) GKO yöntemi ayarları (g) DG yöntemi ayarları (h) L-SHADE yöntemi ayarları (i) HPSGKO yöntemi ayarları

Program Ayarları: Program ayarları menüsünden yerleşim işlemi ile ilgili ayarlar yapılabilmektedir. Bu menüden yerleşim işlemi yapılırken hangi yönden yerleşim işlemine başlanacağı, kaç derecelik dönmelere izin verileceği ve yerleşimi yapılacak olan kalıpların aralarında ne kadar boşluk bırakılacağı ayarları yapılmaktadır (Şekil 4.25.a).

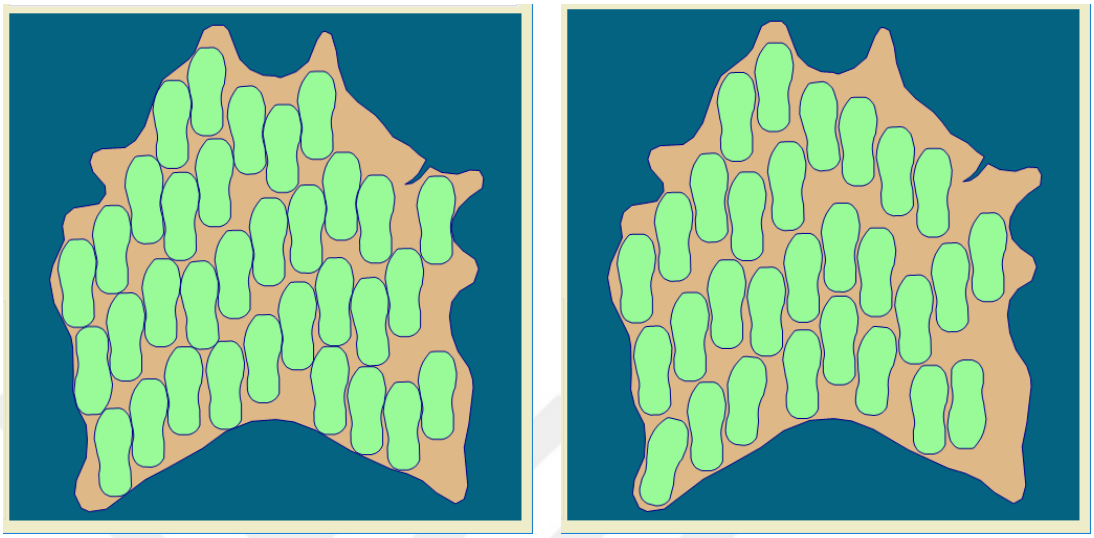


(a) (b)

Şekil 4.25. Yerleşim işlemleri ve bağlantı ayarları (a) Yerleşim işlemleri ayarları (b) Yazdırma işlemi için bağlantı ayarları

Kalıplar yerleştirilirken, aralarına belli miktarda boşluk bırakmak gerekmektedir. Çünkü kesim işlemini yapacak olan makinenin kullandığı bıçak kalınlığı, lazerli ise lazerin çapı gibi parametreler kalıpların sınırlarından küçük miktarda aşındırma yapmaktadır. Bu nedenle, kesim işleminden sonra kalıpların gerçek boyutundan küçük olmaması için, kesim işlemini yapan makinenin özelliğine göre kalıplar arasındaki boşluğun ayarlanması gerekmektedir. Günümüzde kullanılan makineler incelendiğinde makine uçlarının yaklaşık olarak bir ile iki inç arasında bir genişliğe sahip olduğu görülmektedir. Şekil 4.26'da kalıplar arası mesafe ayarlarının

ayarlanabildiğini göstermek amacıyla, mesafenin 1,2 inç ve 10 inç olarak ayarlanarak örnek yerleşim görüntüleri verilmiştir.



(a) (b)

Şekil 4.26. Farklı mesafeler ile dizilim örnekleri (a) Mesafe 1,2 inç için dizilim (b) Mesafe 2 inç için dizilim

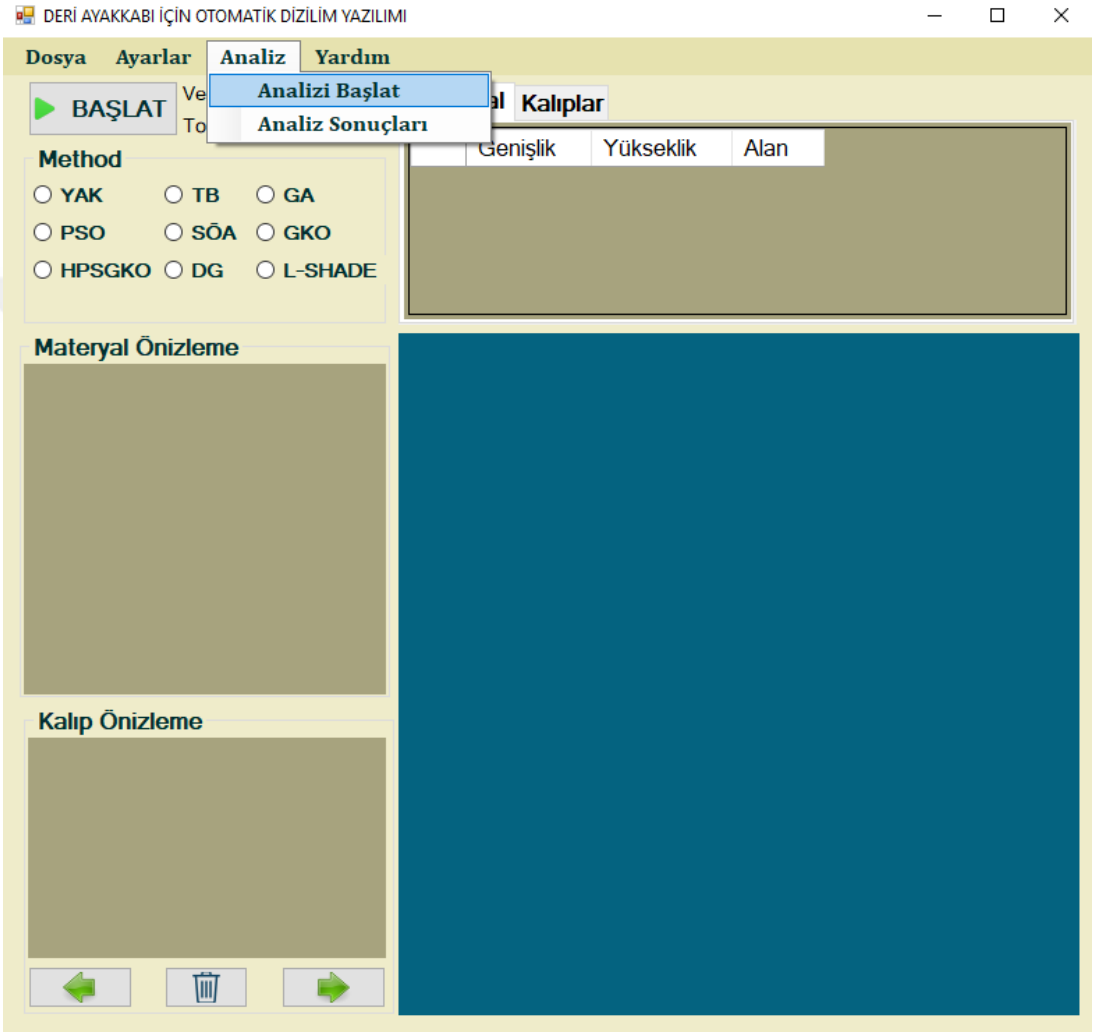
Ayrıca kullanılan materyalin şekline bağlı olarak dizilimin başlanacağı yönünde ayarlanması gerekmektedir. Materyalin düzensiz şekillere sahip olduğu yönünden, küçük boyutlu kalıplarla dizilim yapılması sonuçtaki verimi etkilemektedir. Bu nedenle geliştirilen programa dizilimin başlanacağı yön parametre olarak eklenmiştir.

Yazıcı Ayarları: Bu menü kullanarak, yazılım ile kesim işlemini yapacak olan makinenin bağlantı ayarları yapılmaktadır. Connection Port, yerleşim işleminin doğrudan makineye gönderilebilmesi için makinenin bağlı olduğu seri portun isminin ayarlandığı seçenektir (Şekil 4.25.b). Baud Rate ise seri port bağlantı noktasının hızının ayarlandığı bölümdür (Şekil 4.25.b).

4.3.1.3. Analiz

Geliştirilen programa "Analiz" adında bir ana menü eklenmiştir (Şekil 4.27). Bu menü vasıtasıyla analiz işlemleri gerçekleştirilmektedir. Ayrıca yine bu menü vasıtasıyla elde edilen analiz sonuçlarının grafiksel olarak sonuçları gözlemlenebilmektedir.

Analiz işlemi toplam dokuz adet optimizasyon yöntemi için programa eklenen materyal ve kalıpların yerleşimini yapmaktadır. Her bir yöntem için 10'ar defa çalıştırılarak sonuçlar yerleşim işleminin her bir adımını kapsayacak şekilde *.DXF formatında kayıt altına alınmaktadır. Kullanıcı analiz bittikten sonra istediği dizilimin görüntüsü açabilmekte ve isterse yazıcıya gönderebilmektedir.



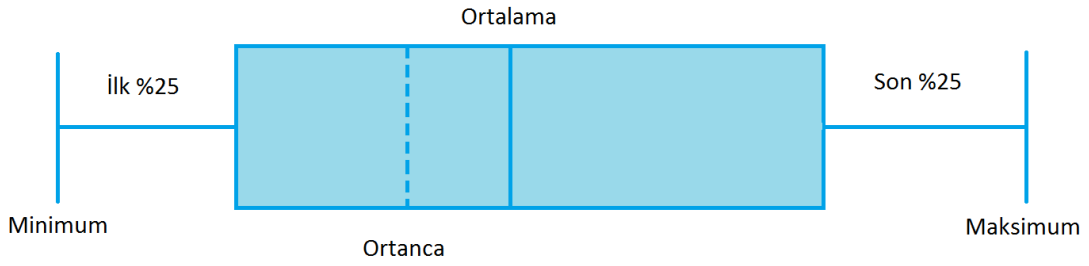
Şekil 4.27. Analiz işlemleri menüsü

Analiz işlemi başlatıldıktan sonra istenirse kullanıcı tarafında iptal edilebilmektedir. Ancak tekrar kaldığı yerden devam edilememektedir. Çünkü analiz işleminde geçen sürede analiz için kullanılmaktadır. Bu nedenle analiz işlemi başladıktan sonra bitinceye kadar durdurulmaması gerekmektedir. Sadece istenirse iptal edilerek tekrar baştan başlatılabilmektedir.

Analiz işlemleri bittikten sonra “Analiz” menüsünden “Analiz Sonuçları” seçeneği ile elde edilen analiz sonuçları gözlemlenebilmektedir. Analiz sonuçları için dört farklı grafikte sonuçlar kullanıcıya gösterilmektedir (Şekil 4.28).

Şekil 4.28. Analiz sonuçları için grafik menüsü

Verim analiz grafiğinde ve Süre analiz grafiğinde BoxPlot-Kutu Bıyık grafiği kullanılmıştır. Bu grafik türünde kullanıcıya birçok bilgi çok daha basit bir gösterilimle verilebilmektedir.



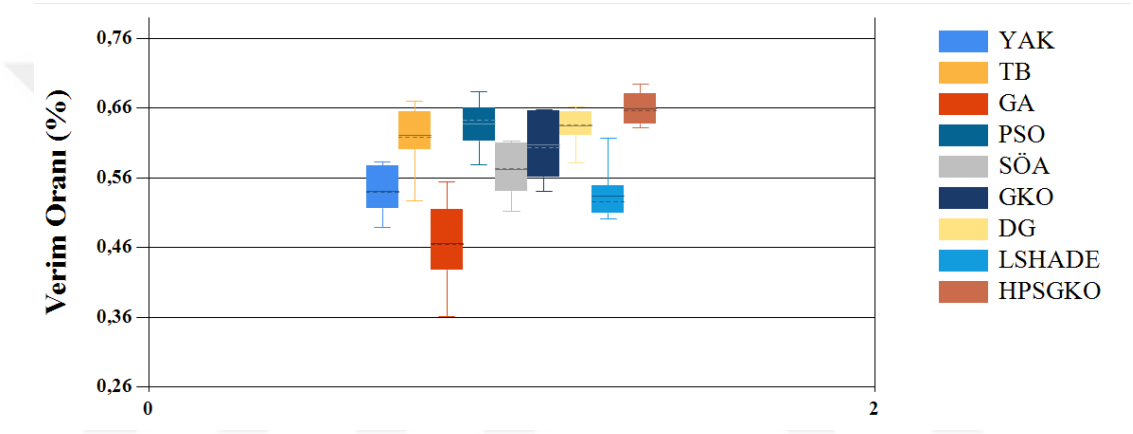
Şekil 4.29. Kutu Bıyık Gösterilimi

Şekil 4.29’da Kutu Bıyık grafiği gösterilmiştir. Grafiğin bir ucu verilerin minimum değerini, diğer ucu maksimum değerini göstermektedir. Verilerin ilk ve son yüzde 25’lik bölümleri çizgi ile gösterilmektedir. Kutu ile gösterilen bölüm verilerin yüzde 50’lik bölümünü ifade etmektedir. Kutu içerisindeki düz çizgi tüm verilerin ortalamasını, kesikli çizgi ise verilerin küçükten büyüğe doğru sıralanmasındaki

ortanca değeri göstermektedir. Bu çizgilere bakarak verilerin dağılımı hakkında bilgi sahibi olunabilmektedir.

Verim Analiz Grafiği

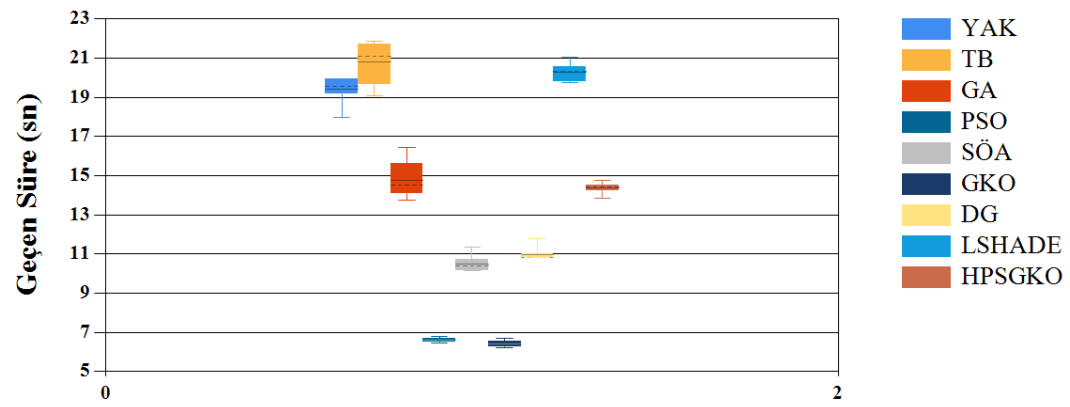
Verim analiz grafiğinde, her bir yöntemin 10'ar kez çalıştırılması sonucunda elde edilen verimlerin karşılaştırılması kutu bıyık grafiği ile gösterilmiştir. Son kalıbın yerleştirilmesi sonucunda elde edilen verimlerin referans alındığı grafik sonucu örnek bir analiz için Şekil 4.30'da gösterilmiştir.



Şekil 4.30. Örnek bir yerleşim sonucu için verim analiz grafiği

Süre Analiz Grafiği

Süre analiz grafiğinde, her bir yöntemin 10'ar kez çalıştırılması sonucunda elde edilen sürelerin karşılaştırılması kutu bıyık grafiği ile gösterilmiştir. Her bir çalıştırılma sonucunda toplam geçen süre ile toplam yerleşen kalıp sayısı dikkate alınarak ortalama bir kalıbın yerleştirilme süreleri karşılaştırılmıştır. Şekil 4.31'de örnek bir yerleşim işlemi için süre analiz grafiği sonucuna bakılabilir.

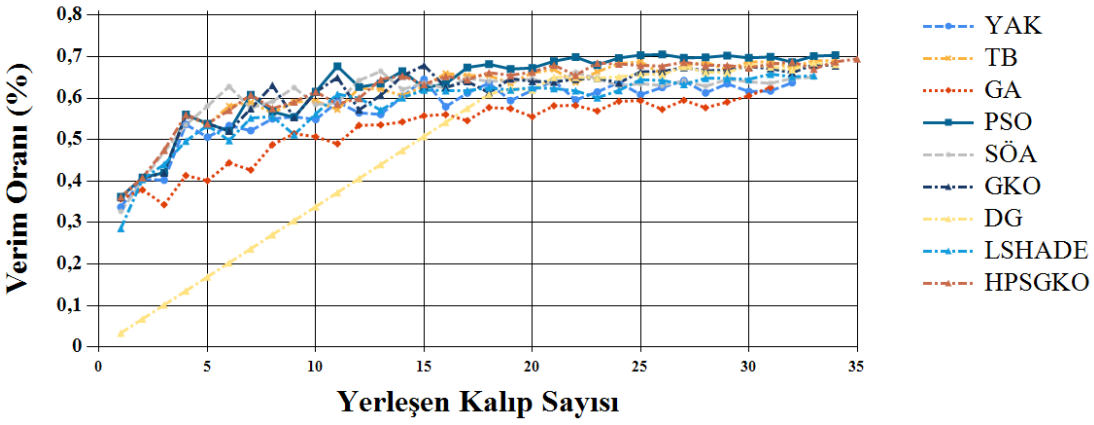


Şekil 4.31. Örnek bir yerleşim sonucu için süre analiz grafiği

Şekil 4.31 incelendiğinde PSO ve GKO yöntemlerinin ortalama 1 saniyede yerleşim yaptığı ve en hızlı oldukları görülmektedir.

Kalıp Verim Grafiği

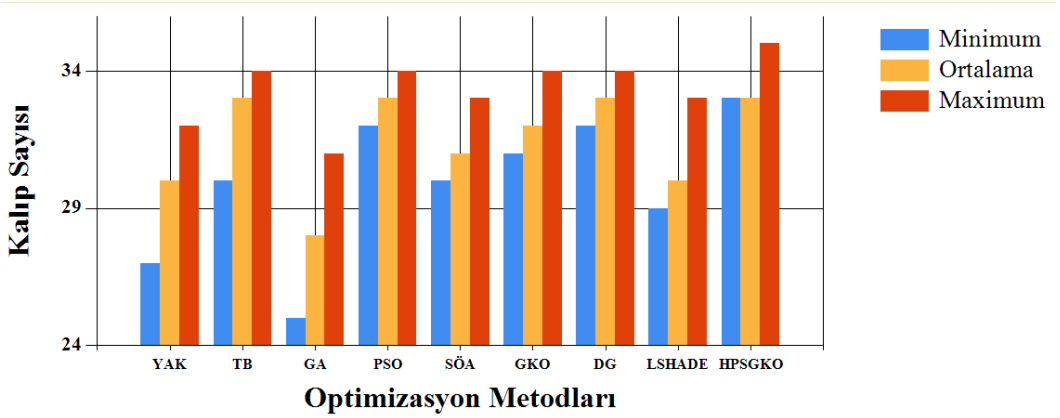
Kalıp verim grafiğinde, çalıştırılan 10 sonucun en iyisi dikkate alınarak her bir kalıbın yerleşiminden sonraki verimleri grafik üzerinde gösterilmiştir. Şekil 4.32'de kalıp verim karşılaştırmaları gösterilmiştir.



Şekil 4.32. Örnek bir yerleşim sonucu için kalıp verim grafiği

Kalıp Analiz Grafiği

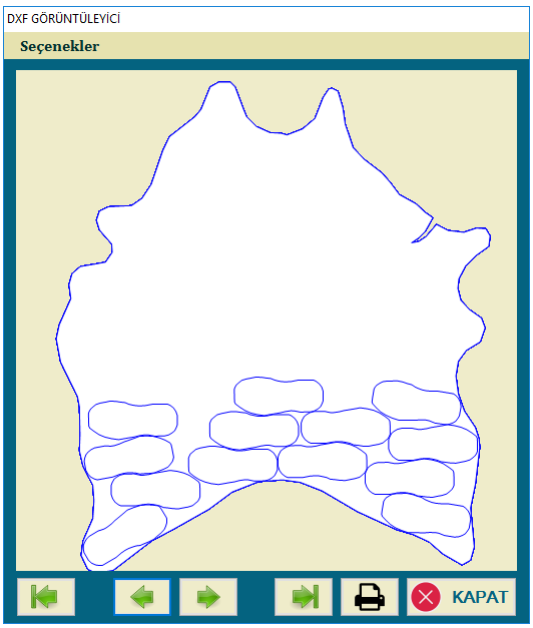
Kalıp analiz grafiğinde, 10 deneme için en az ve en fazla sayıda yerleştirilen kalıp sayıları gösterilmektedir. Ayrıca her bir yöntem için ortalama yerleşen kalıp sayısı da yine bu grafikte gösterilmektedir (Şekil 4.33).



Şekil 4.33. Örnek bir yerleşim sonucu için kalıp analiz grafiği

Analiz Ara Çıktılarının Görüntülenmesi

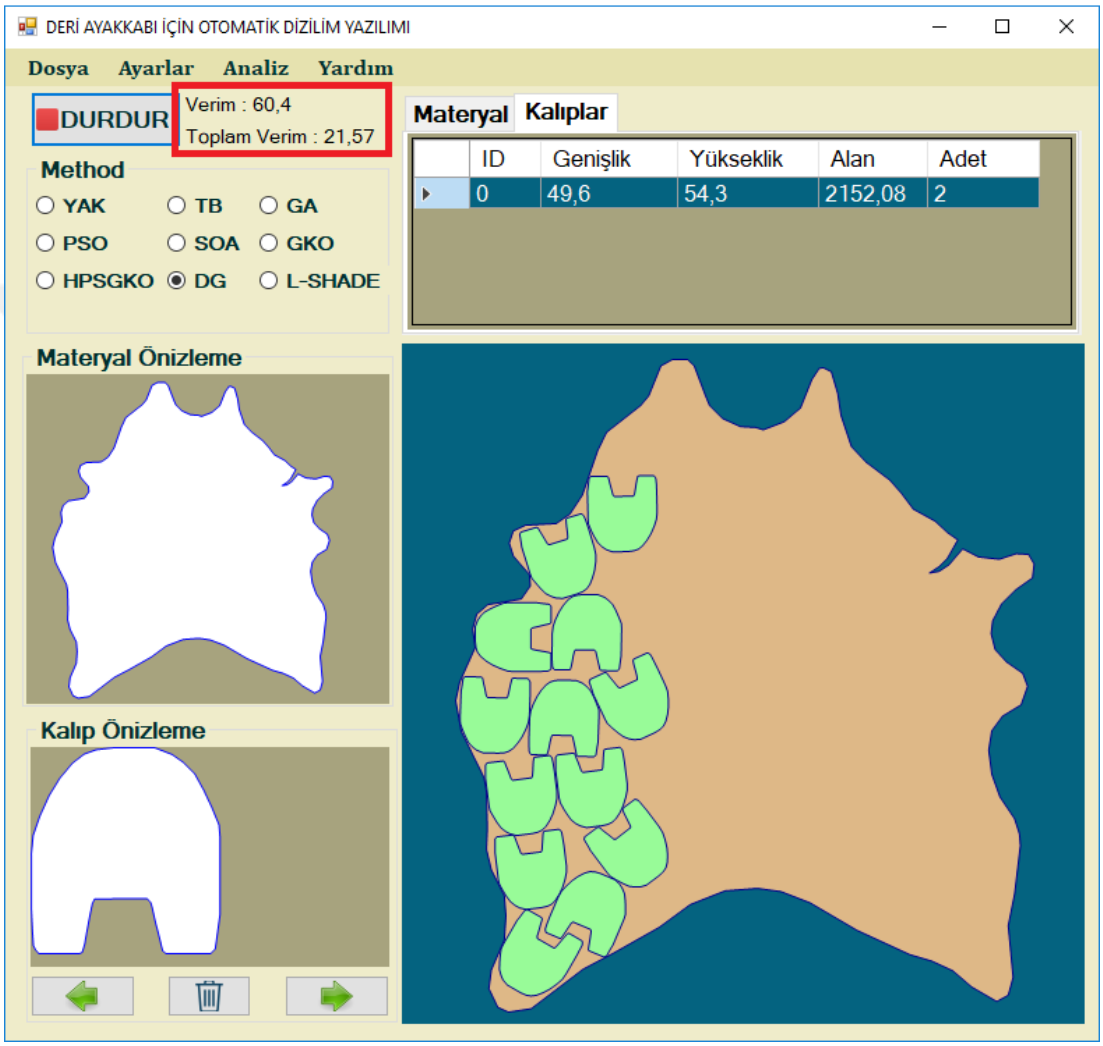
Analiz işlemleri yapılırken, her bir dizilim işleminde elde edilen görüntü *.DXF formatında kaydedilmektedir. Böylece istenildiği zaman adımların nasıl ilerlediği kullanıcı tarafından görüntülenebilmektedir. Ayrıca istenilen bir yerleşim örneği doğrudan çizici vasıtasıyla çıktı alınabilmektedir. Kaydedilen dizilimlerin kullanıcı tarafından görüntülenebilmesi için ana ekranda "Dosya" menüsü altında bulunan "DXF Görüntüleyici" menüsü kullanılmaktadır. Şekil 4.34'te kaydedilen dizilimlerin görüntülendiği form ekranı gösterilmiştir. Form üzerindeki ileri ve geri tuşları ile dizilim işlemi sırasıyla görüntülenebilmektedir.



Şekil 4.34. DXF görüntüleyici ekran görüntüsü

Verim Oranın Gösterilmesi

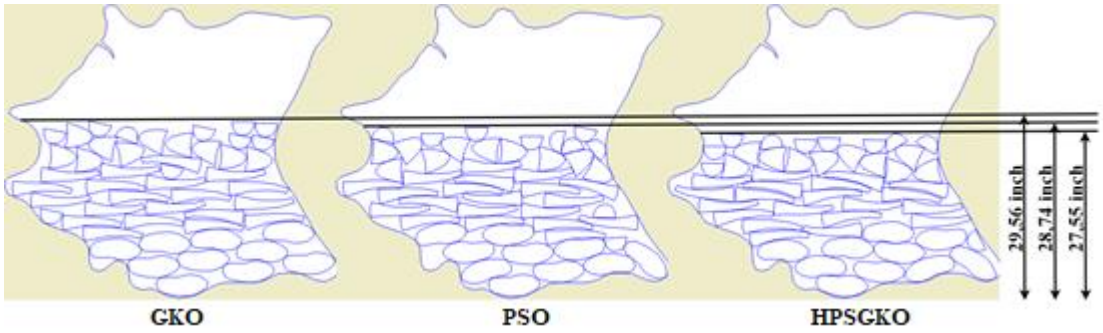
Programa eklenen bir diğer özellik ise her bir kalıbın yerleştirilmesi sonucunda elde edilen verimin kullanıcıya anlık olarak gösterilmesi özelliğidir. Kullanıcı böylece verimi anlık olarak takip edebilmektedir (Şekil 4.35).



Şekil 4.35. Verim yüzdesinin gösterilmesi

Verim hesabı yapılırken, dizilim yönünden en uzağa yerleştirilen kalıbın en uç noktası referans alınır. Bu en uç noktadan itibaren dizilim yönü başlangıcına paralel olarak materyalin kesildiği varsayılmaktadır. Literatür incelemeleri sonucunda verim hesabının bu şekilde yapıldığı gözlendiği için aynı yöntemle verim hesaplanmıştır. Ayrıca kullanıcı, toplam verim oranını da görmek isteyebilir. Yani toplam materyalin yüzde kaçının kullanıldığı bilgisini görmek isteyebilir. Toplam Verim bilgisi ile de, anlık olarak materyalin kullanım yüzdesi kullanıcıya gösterilmektedir. Şekil 4.36'da

örnek yöntemler ve verim hesabı yapılırken kalıpların en uç noktalarının referans alındığı durumlar gösterilmiştir.



Şekil 4.36. Verim hesabı için kalıpların en uç noktalarının görüntüleri

4.3.1.4. Yardım

Hakkında: Program hakkında bilgilerin verildiği menü seçeneğidir. Bu form ile programın bir doktora projesi kapsamında geliştirildiği gibi bilgiler kullanıcıya verilmektedir (Şekil 4.37).

Şekil 4.37. Program hakkında bilgilerin verildiği ekran

4.3.2. Programın kullanılması

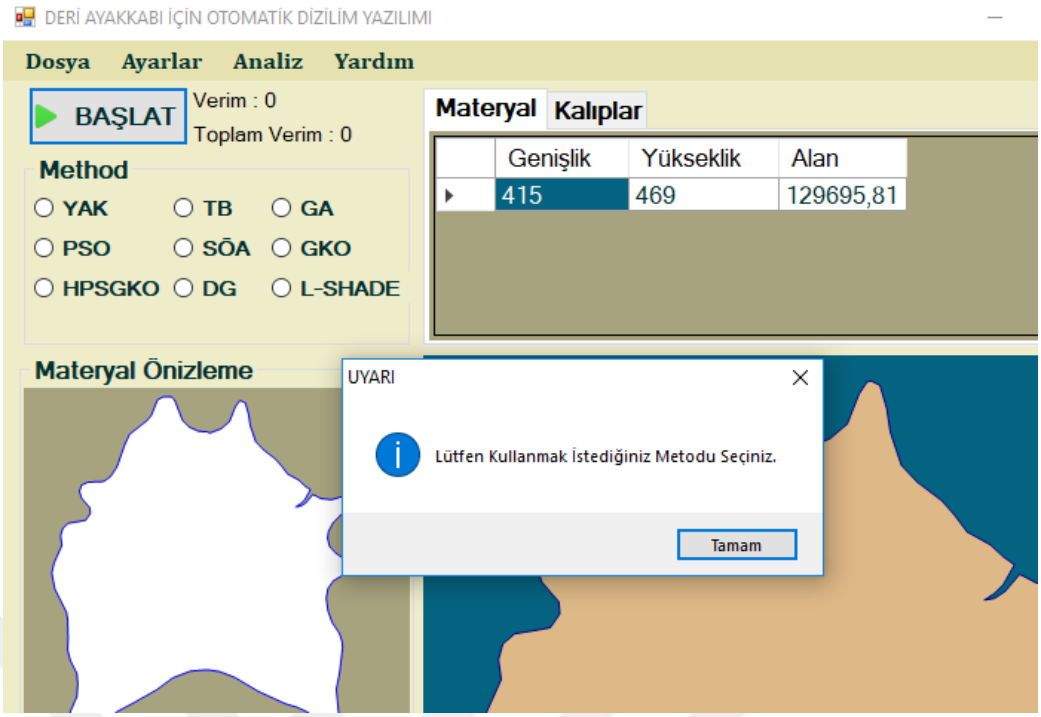
Bu tez çalışmasında geliştirilen yazılım hem el ile hem de otomatik olarak dizilim işlemine izin vermektedir. Tez çalışmasının ilk dönemlerinde yapılan fabrika gezilerinde, el ile dizilim işleminin de yazılımda olması gerektiği anlaşılmıştır. Çünkü bu işlerle uğraşan insanlar zaman zaman dizilim işlemine müdahale edebilmeyi

istemektedirler. Piyasada kullanılan programların tamamına yakını ya tam otomatik ya da tamamen el ile dizilim yapacak şekilde çalışmaktadır. Bu çalışmada diğerlerinden farklı olarak hem el ile hem de otomatik dizilime izin veren bir yazılım geliştirilmiştir.

El ile dizilim için bilgisayarın faresi çok etkin bir şekilde kullanılabilmektedir. Farenin sol tuşu, kalıpları yerleşim alanında hareket ettirme işlemini, sağ tuşu kalıpları izin verilen açılarda döndürme işlemini ve farenin orta tuşu da yerleşimi yapılan kalıbı silme işini yapmaktadır. Bu işlemler yapılırken, çalışma alanında kalıpların herhangi bir kalıpla çakışma olması veya materyal sınırından dışarı taşması gibi durumlar renkli olarak kullanıcıya gösterilmektedir. Yerleşimi başarıyla yapılan yani herhangi bir hatası olmayan kalıplar yeşil renkte, hatalı olarak yerleşen yani çakışma ve taşma durumunda olan kalıplar ise kırmızı renkte gösterilmektedir. Böylece kullanıcı kalıpları el ile dizerken herhangi bir hata durumunu doğrudan fark edebilmekte ve gerekli düzeltmeleri anında yapabilmektedir.

Ayıraca, otomatik dizilim işlemi devam ederken, kullanıcı her hangi bir anda dizilim işlemine müdahale etmek isteyebilir. Yani kullanıcının çok daha verimli olduğunu düşündüğü bir yerleşim işlemi için dizilim işlemini duraksatıp, kendisi birkaç kalıp yerleştirdikten sonra otomatik dizilim işlemini devam ettirebilir. Kullanıcı istediği bir anda dizilim işlemini durdurup müdahale edip ve tekrar başlatabilir. Böylece otomatik dizilim işlemi kullanıcı açısından çok daha esnek hale getirilmiştir. Geliştirilen yazılım bu yönleri dikkate alındığında kullanıcıyla çok iyi bir etkileşim içinde olduğu görülmektedir.

Bu tez çalışmasında geliştirilen yazılım el ile dizilim sunarak kullanıcı dostu bir uygulama olsa da yazılımın en büyük özelliği otomatik dizilim işlemini başarılı bir şekilde gerçekleştirebiliyor olmasıdır. Programda otomatik yerleşim işlemine başlamadan önce hangi yöntemle yerleşim yapılacağı seçimi yapılmalıdır. Böylece dokuz farklı yöntemin yerleşim işlemindeki başarısı karşılaştırılabilmektedir. Kullanıcı otomatik yerleşim işlemine başlamadan önce istediği yöntemi seçebilmektedir. Herhangi bir seçim yapmadan başlatılmaya çalışılan dizilime, uygulama uyarı vererek kullanıcıyı istediği yönteme seçmeye yönlendirmektedir. (Şekil 4.38).



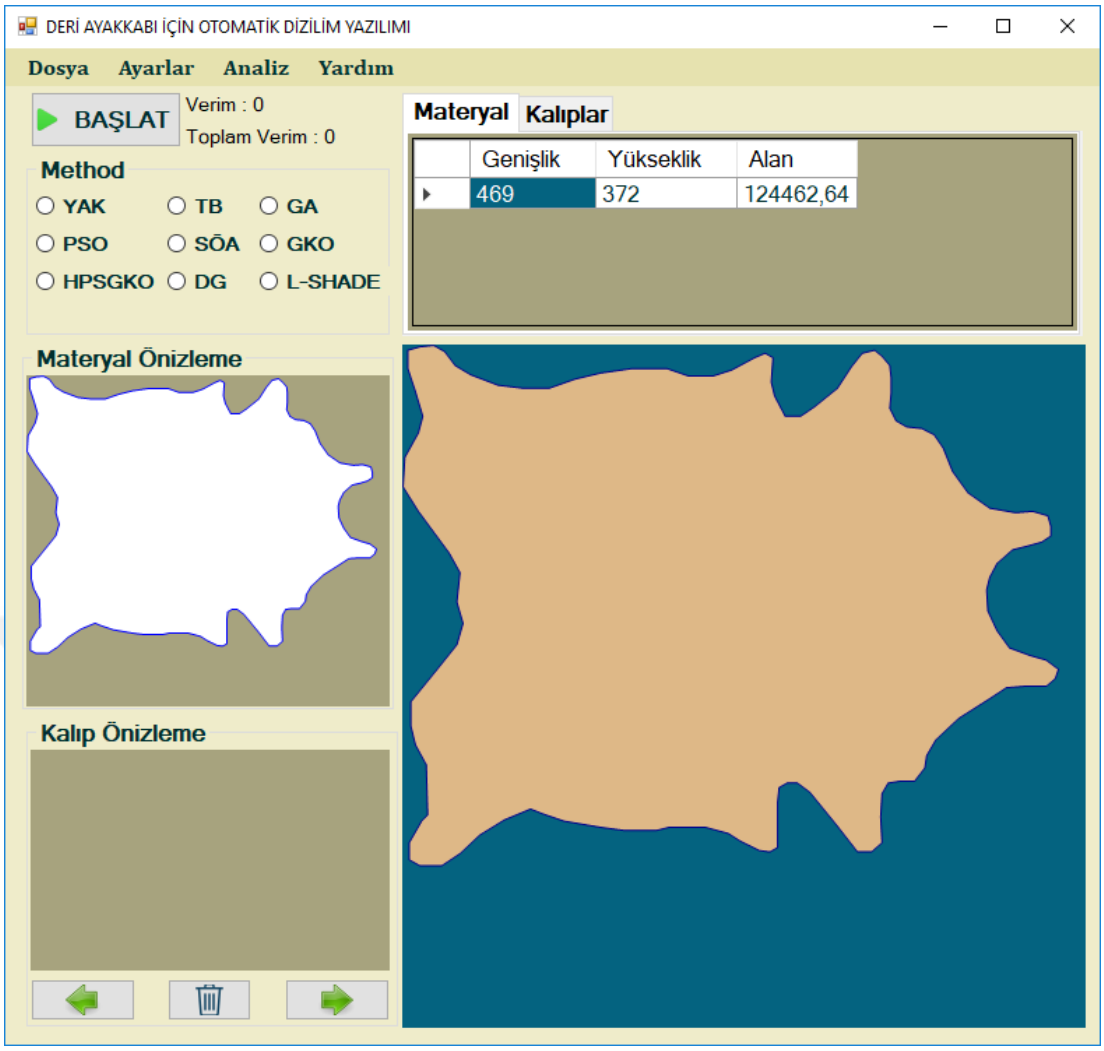
Şekil 4.38. Optimizasyon yöntemi seçimi ekranı

Kullanılan optimizasyon yöntemlerinin parametreleri Ayarlar menüsünden değiştirilebilmektedir.

4.3.2.1. Kullanılacak materyalin seçimi

Yerleşim işleminin yapılacağı materyal seçimi Dosya->İçe Aktar->Materyal yolu izlenerek gerçekleştirilmektedir. Bu yolla DXF formatındaki materyal programa aktarılabilmektedir. Programa aktarılan materyalin görüntüsü Materyal Önizleme bölümünde gösterilmektedir. Aynı zamanda yerleşim yapılacağı zemine materyal görüntüsü yerleştirilmektedir. Materyal yerleştirilirken zemine sığacak şekilde ölçeklenmektedir. Hesaplanan ölçek oranı kalıplar içe aktarılırken de kullanılacağından, öncelikle programa materyal dâhil edilmelidir.

İçe aktarma işlemi yapılırken dosya seçimi penceresinde sadece DXF uzantılı dosyalar gösterilmektedir. Programa aktarılan materyalin ölçeklendikten sonraki genişlik ve uzunluk ölçüleri ile toplam alanı hesaplanarak ekranda kullanıcıya bilgi amaçlı olarak gösterilmektedir. Şekil 4.39'da örnek bir deri görüntüsü programa aktarıldığı ve deri ile ilgili sayısal bilgilerin gösterildiği ekran görüntüsü verilmiştir.

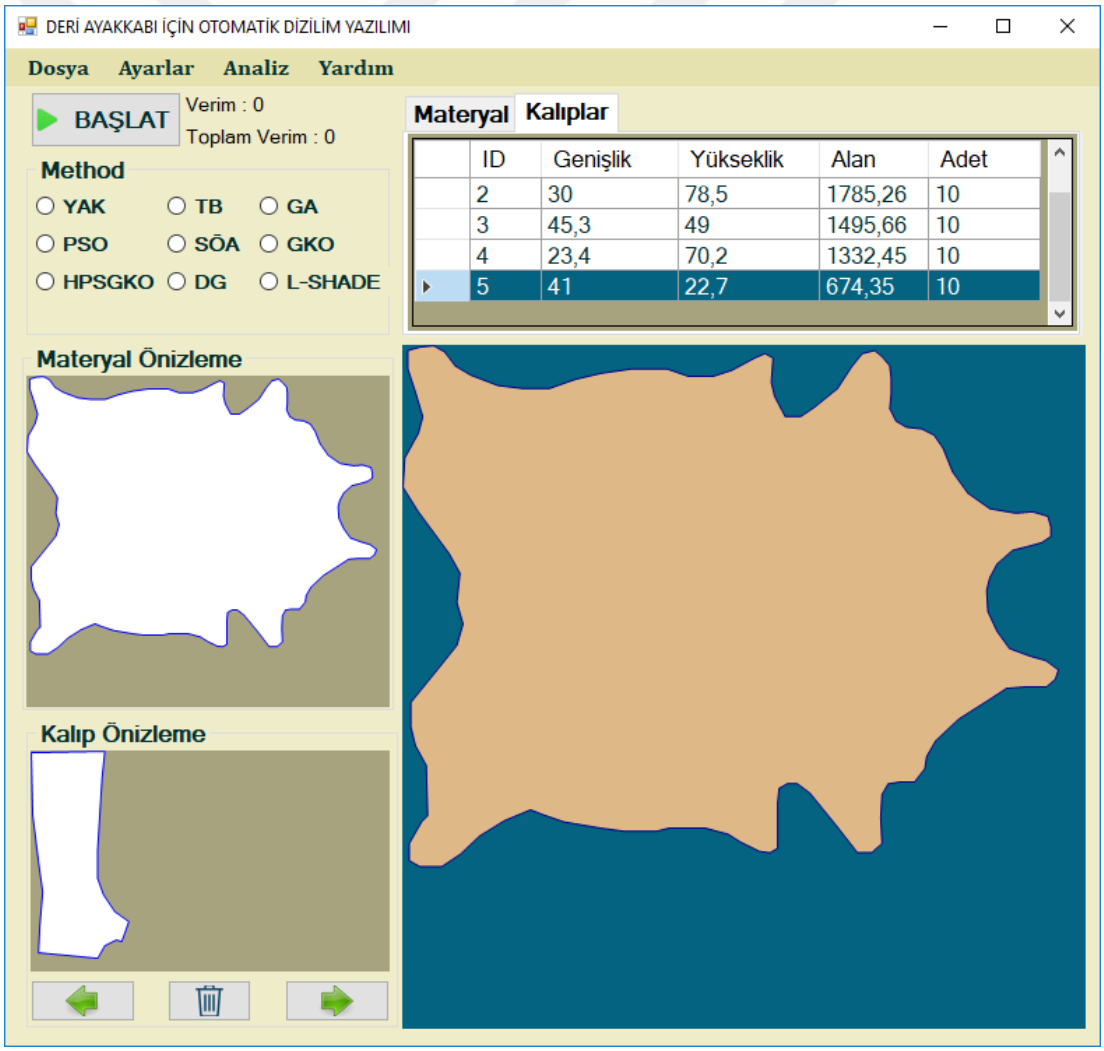


Şekil 4.39. Materyalin programa aktarılması

4.3.2.2. Kullanılacak kalıpların seçimi

Yerleşim işleminin yapılacağı kalıpların seçimi Dosya->İçe Aktar->Kalıplar yolu izlenerek DXF formatındaki programa aktarılabilmektedir. Programa yerleşim işlemi yapılırken istenen kalıplar sayı sınırı olmaksızın eklenebilmektedir. Eklenen kalıplar, materyal eklenirken elde edilen ölçekleme oranına göre ölçeklenerek yerleşim işleminde kullanılmaktadırlar. Ana ekranın sol alt bölümünde içe aktarılan kalıplar arasında "İleri-Geri" tuşları ile gezilebilmektedir. Aynı şekilde Kalıp Önizleme bölümünde o anda gösterilen kalıp hangisi ise o kalıp fare yardımıyla materyal üzerine yerleştirilebilmektedir. Programa aktarıldıktan sonra silinmek istenen kalıp varsa çöp kutusu simgeli tuş yardımıyla programdan silinebilir. Kalıplar eklendikçe materyalde olduğu gibi DataGrid nesnesine de eklenmektedirler. Genişlik, uzunluk, alan ve adet bilgileri DataGrid nesnesinden takip edilebilmektedir.

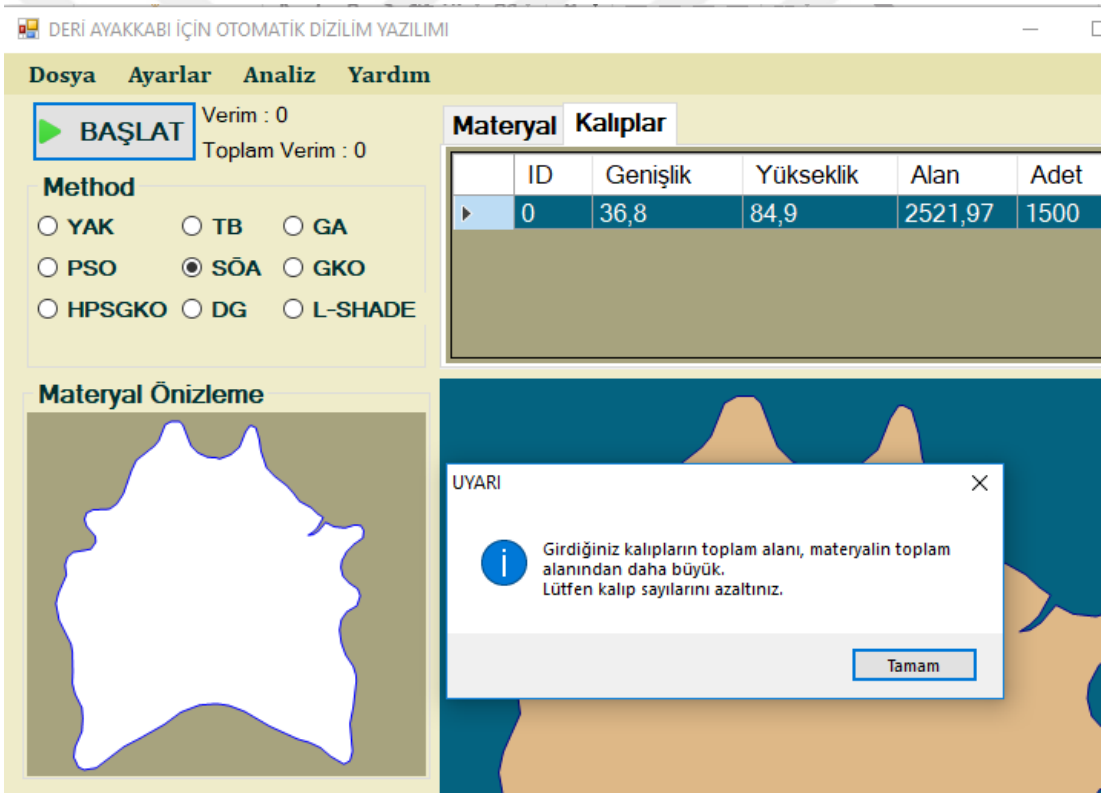
Kalıplar DataGrid nesnesine alanlarına göre büyükten küçüğe doğru sıralı olacak şekilde eklenmektedir. Yerleşim işlemi yapılırken de kalıplar öncelikle büyükten başlar ve küçük kalıplara doğru ilerler. Kullanıcı eğer farklı bir sıralamada dizilim yapmak isterse kalıpları teker teker ekleyerek dizilimi otomatik yaptırabilir. Böylece kullanıcı kalıpları istediği sırayla da yerleştirebilmiş olmaktadır. Kalıpların kaç adet yerleşeceği kullanıcı bilgisine bırakıldığından, kullanıcı DataGrid nesnesi üzerinden sayılarda güncelleme yapabilmektedir. "İleri-Geri" tuşlarıyla yapılan işlev DataGrid nesnesi üzerinde de yapılabilmektedir. Yani DataGrid nesnesinde seçilen kalıp hangisi ise, Kalıplar bölümünde gösterilmektedir. Şekil 4.40'ta iki adet örnek ayakkabı kalıbının eklendiği bir ekran görüntüsü verilmiştir.



Şekil 4.40. Kalıpların programa aktarılması

4.3.2.3. Materyal ve kalıp alan kontrolü

Geliştirilen programda, otomatik dizilim için materyal ve kalıplar seçildikten sonra, yerleşim alanının yeterli olup olmaması kontrolü eklenmiştir. Böylece kullanıcı otomatik dizilime başlamadan önce eğer materyale sığmayacak kadar fazla miktarda kalıp seçmişse program tarafından uyarılacak ve kalıp sayısında düzenlemeye gidilecektir. Bu özellik vasıtasıyla kullanıcı, eğer materyale sığmayacak kadar fazla kalıp seçmişse programın sona geldiğinde gereksiz yere dizilim işlemini devam ettirmesinin önüne geçmiş olacaktır. Alan hesabı yaparken materyalin tamamının dolabilme ihtimali göz önünde bulundurulmuştur. Şekil 4.41'de örnek bir uyarı mesajı gösterilmiştir.



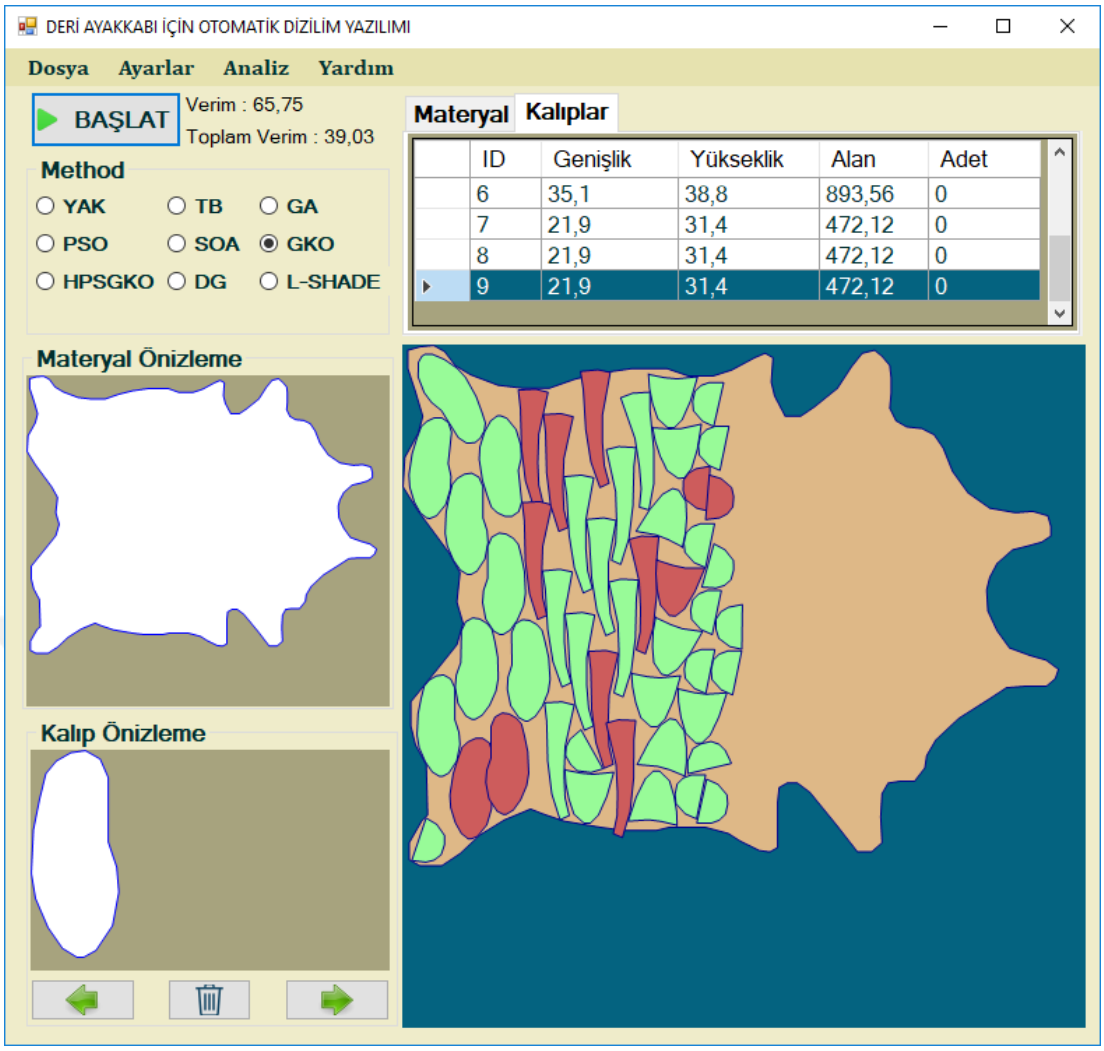
Şekil 4.41. Materyal-Kalıp alan kontrolü

4.3.2.4. Materyal ve kalıp bilgileri (DataGrid Nesnesi)

Bölüm 4.3.2.1 ve Bölüm 4.3.2.2'de DataGrid nesnesi hakkında bilgiler verilmiştir. DataGrid nesnesi temel olarak programa aktarılan materyal ve kalıplar hakkında genişlik, uzunluk, alan ve adet bilgilerinin gösterildiği bölümdür.

4.3.2.5. Yerleşim işlemi alanı

Programa aktarılan materyal üzerine kalıpların yerleşimini anlık olarak gösteren ve programın görsel olarak en önemli bölümü Yerleşim İşlemi Alanı'dır. Her hangi bir çakışma ya da materyalin sınırlarından taşma durumunda olan kalıplar varsa farklı bir renkte gösterilerek kullanıcıyı uyarma özelliği eklenmiştir. Şekil 4.42'de örnek bir yerleşim görüntüsü verilmiştir. Bazı kalıplar bilinçli olarak çakışma ve materyalden taşmalara maruz bırakılarak problemli olan yerlerin farklı bir renkte gösterilmesi sağlanmıştır. Böylece kullanıcı çakışma ve taşmaların olduğu bölümleri doğrudan görerek el ile müdahale edebilmektedir. Böylece kullanıcı için hız kazandırılmış olmaktadır.



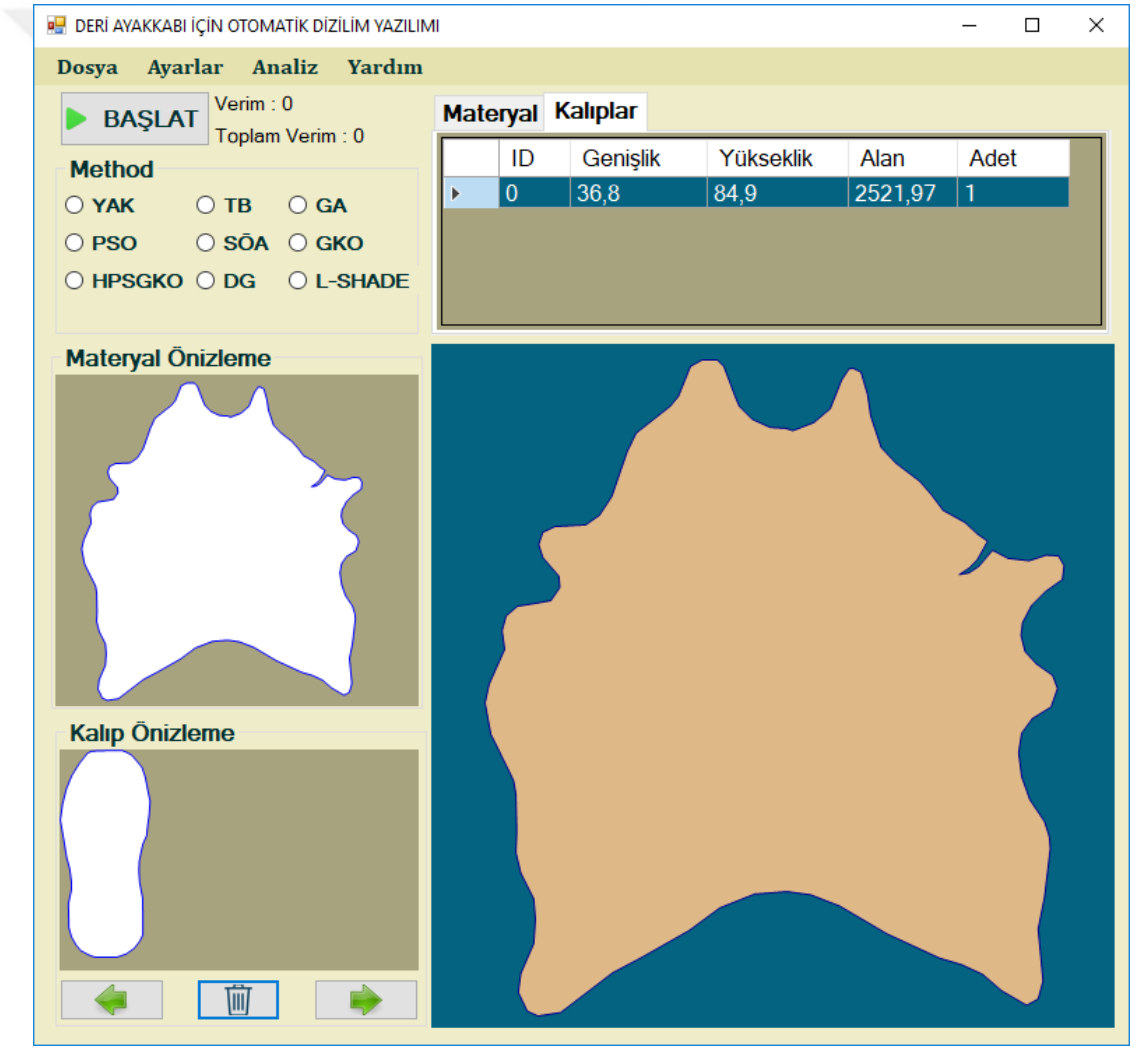
Şekil 4.42. Çakışma ve taşmaların olduğu örnek bir yerleşim görüntüsü

4.4. Deneysel Sonuçlar

Bu tez çalışmasında, hem el ile hem de otomatik dizilim yapabilen bir uygulama geliştirilmiştir. Otomatik dizilim işlemi yapılırken dokuz farklı optimizasyon yöntemi kullanılmıştır. Tüm yöntemler, ilk olarak tek bir materyal ve tek bir kalıp örneği ile daha sonra da dört farklı materyal için dört farklı ayakkabı modeli kalıpları ile test edilmiştir. Ayrıca dizilim işlemi için ayar parametreleri değiştirilerek hangi parametrelerin hangi durumlarda dikkate alınması gerektiği gibi durumlar tespit edilmiştir. İlerleyen başlıklarda bu tez kapsamında gerçekleştirilmiş olan deneysel sonuçlar anlatılmıştır.

4.4.1. Tek materyal ve tek kalıp örneği ile yöntemlerin karşılaştırılması

Dokuz adet optimizasyon yöntemi programın farklı parametreleri için çalıştırılarak sonuçlar karşılaştırılmıştır. Programın ayar parametrelerine bakıldığında, dizilim yönü ve izin verilen döndürme açısı parametrelerinin ayarlanabildiği görülmektedir. Bu bölümde, aynı veri seti için her bir yöntem farklı ayarlarla çalıştırılarak sonuçlar incelenmiştir. Karşılaştırma yaparken süre ve verim olarak iki farklı değerlendirme kriteri ele alınmıştır. Sonuçlar ilk olarak tek bir materyal ve tek bir kalıp örneği ile karşılaştırılmıştır. Şekil 4.43'te karşılaştırma işleminin yapıldığı tek bir materyal ve tek bir kalıbın görüntülerine yer verilmiştir.

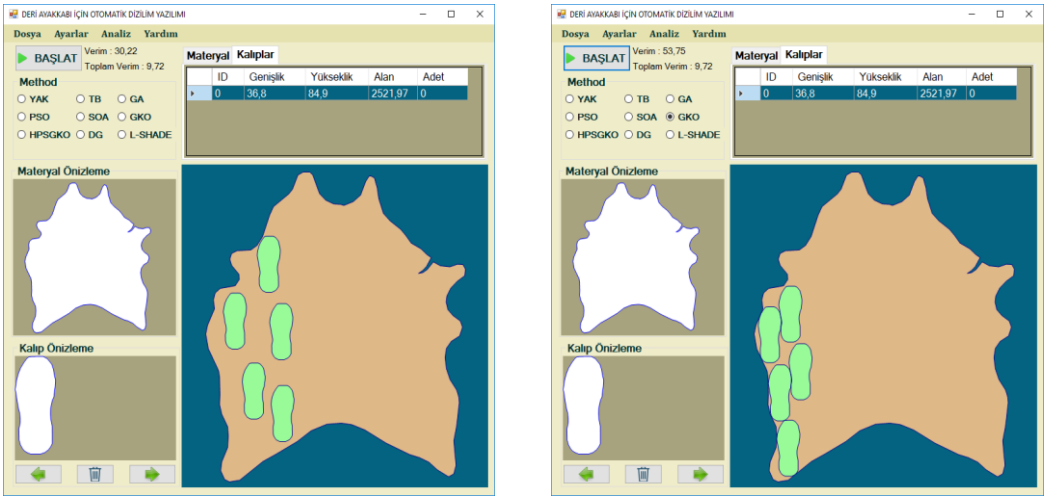


Şekil 4.43. Sonuçların karşılaştırılması için kullanılan materyal ve kalıp görüntüsü

4.4.1.1. Farklı dizilim yönlerinin karşılaştırılması

Geliştirilen yazılımın Analiz menüsü kullanılarak, dokuz adet optimizasyon yönteminin karşılaştırılması yapılabilmektedir. Kullanılan optimizasyon yöntemlerinin, dizilim işlemine başladıkları yönlere göre süre ve verim karşılaştırılması Çizelge 4.3'te gösterilmiştir. Karşılaştırma işlemi yapılırken döndürme açısının 0^{o} olduğu durum değerlendirilmiştir. Aynı zamanda karşılaştırma işlemi yapılırken, yerleştirilen kalıpların sayıları da hesaplamaya dâhil edilmiştir. Aynı analiz sonucu için en fazla sayıda kalıp yerleştiren yöntemin, yerleştirdiği kalıp sayısı ile daha az sayıda kalıp yerleştiren yöntemlerin yerleştirdiği kalıp sayıları orantılanarak hesaplamalar yapılmıştır. Örnek olarak 25 kalıp yerleştiren bir yöntemin elde ettiği verim sonucu, 26 kalıp yerleştiren yöntemin elde ettiği verim sonucundan iyi olabilmektedir. Fakat tüm materyalin verim oranı dikkate alındığında daha fazla kalıp yerleştiren yöntem daha da avantajlı olacaktır. Geliştirilen arayüz üzerinde görülen Verim ve Toplam Verim sonuçları bu durumu açıklamaktadır (Şekil 4.35).

Şekil 4.44'te Toplam Verim değerleri aynı olan iki farklı yerleşim örneği gösterilmiştir.

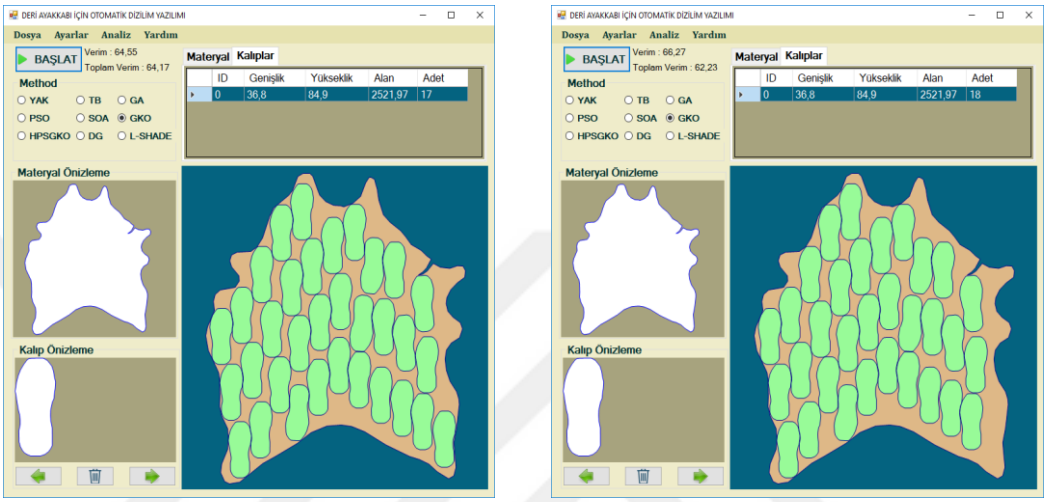


(a) (b)

Şekil 4.44. Toplam Verim ve Verim değerlerinin karşılaştırılması (a) Örnek kötü yerleşim (b) Örnek iyi yerleşim

Şekil 4.44'ten görüldüğü üzere, Toplam Verim değerleri %9.72 olan fakat Verim değerleri farklı olan örnek yerleşimler gösterilmiştir. Yani eğer yerleşimi yapılmak istenen kalıplar materyal alanını doldurmayacaksa Toplam Verim değerlerine göre

karşılaştırma yapılması yanlış sonuçlar vermektedir. Çünkü aynı verim değerlerini veren farklı iki yerleşim aynı başarıyı elde etmiş gibi yanlış sonuçlar verebilmektedir. Benzer şekilde materyalin tamamının kullanılmaya zorlandığı yerleşim işlemlerinde de sadece Verim değerine bakılarak karşılaştırma işleminin yapılması da sonuçlarda yanlışlıklara neden olabilmektedir. Şekil 4.45'te verilen örnek sonuç incelenirse, Verim değerlerine göre daha iyi olan fakat Toplam Verim değerinde kötü olan yerleşim sonuçları ile karşılaşılabilmektedir.



(a) (b)

Şekil 4.45. Toplam Verim ve Verim değerlerinin karşılaştırılması (a) Örnek 33 adet kalıp yerleşimi (b) Örnek 32 adet kalıp yerleşim

Şekil 4.45.b'de, Verim değeri %66.27 olan ve Şekil 4.45.a'da Verim değeri %64.55 olan örnek yerleşimler verilmiştir. Görüldüğü üzere Verim değeri dikkate alınarak karşılaştırma işlemi yapıldığı zaman Şekil 4.45.b'nin daha başarılı olduğu sonucu ortaya çıkmaktadır. Fakat Şekil 4.45.a'nın daha fazla kalıp yerleştirdiği ve Toplam Verim sonucunun daha iyi olduğu görülmektedir. Dolayısıyla, sadece Toplam Verim ve sadece Verim değerlerinin dikkate alındığı durumlarda yanlış sonuçlar elde edilebilmektedir. Bu nedenle, her ikisinin birden dikkate alınabildiği bir karşılaştırma yönteminin geliştirilmesi gerekmektedir. Denklem 4.12'de verilen işlem ile karşılaştırma işlemlerinin yapılabilmesi için kullanılan formül gösterilmiştir.

$$Karşılaştırma_{verimi} = \frac{Yerleşen_{kalıp}}{Maksimum_{kalıp}} \times Verim \tag{4.12}$$

$Yerleşen_{kalıp}$, karşılaştırma işlemine tabi tutulan yöntemin yerleştirdiği kalıp sayısını, $Maksimum_{kalıp}$, analiz işlemi sonucunda maksimum yerleştirilebilen kalıp sayısını ve $Verim$ değeri ise arayüzde gösterilen Verim değerini ifade etmektedir. Şekil 4.45.a'da en fazla kalıp yerleştirildiği için Verim değerinde bir değişiklik olmayıp, Şekil 4.45.b'nin ise Denklem 4.12'den belirtilen hesap sonucunda Verim değeri %64.26 olarak hesaplanmaktadır. Böylece beklenen sonuç yani Şekil 4.45.a'nın daha başarılı olduğu ortaya çıkmaktadır.

Farklı açı değerleri ile elde edilen verim karşılaştırma sonuçlarına Ek A'dan bakılabilir.

Çizelge 4.3. Dizilim yönlerine göre sonuçların karşılaştırılması

	Soldan Sağa		Sağdan Sola		Yukarıdan Aşağı		Aşağıdan Yukarı	
Yöntem	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)
YAK	54,60	22,50	53,62	24,81	**55,46**	22,45	53,88	22,28
TB	59,88	20,67	58,83	21,71	52,59	20,22	**62,67**	21,35
GA	49,13	14,52	46,75	14,63	**50,03**	15,13	48,12	14,94
PSO	60,50	6,43	60,78	6,39	63,42	6,65	**71,67**	6,69
SÖA	**57,82**	11,05	54,60	10,43	55,93	10,02	53,92	9,96
GKO	61,28	6,29	60,59	6,28	59,58	6,24	**70,64**	6,38
DG	59,66	10,75	59,06	10,63	56,33	10,79	**68,54**	11,19
L-SHADE	**55,99**	19,79	53,44	20,26	46,61	18,82	54,69	19,65
HPSGKO	59,91	14,47	60,10	14,28	65,79	14,99	**71,69**	15,41

Çizelge 4.3 incelendiğinde, yerleşim işleminin en başarılı yöntemlerinin aşağıdan yukarı doğru dizilim yönünde en iyi sonuçlar aldığı görülmektedir. Materyal incelendiğinde ise, materyalin alt kısmının diğer yönlere göre daha düzenli şekle sahip olduğu görülmektedir. Yani dizilim işlemine, materyalin en düzenli şekle sahip olduğu yönden başlanması verimi artırmaktadır. Kullanılan kalıp şekillerinin de büyük etkisi bulunmaktadır. Küçük şekilli kalıpların kullanılması durumunda, materyalin ilk olarak düzensiz şekilli yönünden başlanması verimi artıracaktır. Çünkü ilk dizilim aşamalarında materyalin düzensiz şekle sahip olduğu girintili çıkıntılı bölümler

kalıplarla doldurularak, dizilim ilerledikçe daha düzenli alanların açılmasını sağlamaktadır.

4.4.1.2. Farklı açı değerleri ile elde edilen sonuçların karşılaştırılması

Bu tezde kullanılan optimizasyon yöntemleri karşılaştırılırken dikkate alınan bir diğer dizilim ayarı parametresi döndürme açı değeridir. Kalıplar için döndürülmeye izin verilen açı değerleri dizilim işleminde verimi büyük oranda etkilemektedir. Ancak süre olarak bakıldığında toplam süreyi uzattığı görülmektedir. Kullanıcıların kendi istekleri doğrultusunda yani eğer süre olarak kısıt yoksa döndürme açılarını değiştirerek en fazla verimi elde ettiği dizilim işlemlerini kullanabilmektedir. Burada dikkat edilmesi gereken nokta, dizilim işlemine tabi tutulan kalıpların şekilleridir. Çünkü kalıpların düzenli şekle sahip olması, döndürme açısından çok az etkilenmesini sağlamaktadır. Kalıplar ne kadar düzensiz şekle sahipse, döndürme açısının değeri sonucu büyük oranda etkilemektedir. Bu nedenle kullanıcı dizilim işlemine başlamadan önce kalıbın şekline bakarak döndürme açısının değerini ayarlamalıdır. Böylece kullanıcılar, gereksiz zaman kayıplarından kurtulmakla birlikte elde edilen verimi de artırmış olacaktırlar. Çizelge 4.4'te 0^o, 1^o, 30^o ve 90^o değerleri için elde edilen sonuçlar aşağıdan yukarı doğru dizilim yönüne göre verilmiştir.

Çizelge 4.4. Döndürme açısı değerlerine göre sonuçların karşılaştırılması

	0^o		1^o		30^o		90^o	
Yöntem	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)
YAK	53,88	22,28	54,96	22,23	57,27	19,79	**57,89**	19,62
TB	62,67	21,35	61,45	22,44	**65,14**	22,57	61,45	20,78
GA	**48,12**	14,94	42,50	15,12	42,20	14,53	40,43	14,42
PSO	**71,67**	6,69	66,70	6,81	66,31	6,59	67,05	6,58
SÖA	**53,92**	9,96	47,87	10,94	48,82	9,82	49,76	9,98
GKO	**70,64**	6,38	65,95	6,55	64,09	6,29	63,63	6,27
DG	68,54	11,19	65,70	11,31	**69,94**	13,81	67,87	11,03
L-SHADE	54,69	19,65	48,70	20,03	52,49	19,90	**56,69**	20,11
HPSGKO	**71,69**	15,41	67,33	12,82	70,70	14,96	67,50	15,38
Ortalama	61,76		57,91		59,66		59,14	

Bu tez çalışmasında yapılan analizler ile daha fazla açı değerleri için sonuçlar elde edilmiştir. Bu sonuçlar; Ek A'da verim karşılaştırması olarak, Ek B'de ise süre karşılaştırması olarak detaylı bir şekilde verilmiştir.

Çizelge 4.4'ten görüldüğü üzere, yarıdan fazla durumlarda, en iyi sonuçlar 0^o döndürme açısıyla elde edilmiştir. Kullanılan kalıpların şekli ne kadar düzensiz ise daha hassas açı değerleri ile döndürmelere izin verilmeli, ne kadar düzenli ise daha az açı değerlerine izin vermek süreden avantaj elde etmemizi sağlayacaktır. Kullanılan kalıbın daha küçük olduğu durumlarda elde edilen verim oranı daha da artmaktadır. Çünkü daha küçük kalıplar daha fazla ayrıntılara yerleşebildiği için daha az fire verilmektedir. Özellikle materyallerin düzensiz şekilli yönlerinden daha küçük kalıplarla dizilime başlanması toplam verimi artırmaktadır. Çizelge 4.5'te, Çizelge 4.4'te kullanılan aynı materyale daha küçük boyutlu kalıp yerleştirilerek elde edilen sonuçlar gösterilmiştir.

Çizelge 4.5. Daha küçük kalıp kullanılarak elde edilen sonuçların karşılaştırılması

	0^o		1^o		30^o		90^o	
Yöntem	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)
YAK	55,72	17,25	**61,65**	17,53	59,77	17,74	59,50	17,60
TB	60,66	9,27	**68,01**	9,32	63,78	9,12	64,88	9,20
GA	50,18	6,61	**52,37**	6,77	51,30	6,75	50,89	6,64
PSO	59,54	2,72	65,45	2,76	**67,58**	2,82	65,56	2,78
SÖA	53,26	4,40	55,45	4,46	52,81	4,60	**55,91**	4,51
GKO	60,38	2,57	**63,85**	2,63	61,00	2,67	63,59	2,63
DG	57,25	6,62	**63,77**	6,61	62,34	6,50	60,41	6,65
L-SHADE	47,42	2,41	**66,10**	7,34	65,29	7,47	60,44	7,24
HPSGKO	56,08	7,33	62,30	2,69	**64,94**	2,70	63,60	2,71
Ortalama	55,61		62,11		60,98		60,53	

Çizelge 4.5'te daha küçük kalıp örneği için elde edilen sonuçlar gösterilmiştir. Çizelge 4.5 incelendiğinde, Çizelge 4.4'e göre daha fazla verimlerin elde edildiği durumlar görülmektedir. Ortalama olarak tüm yöntemler tarafından elde edilen sonuçlar, çizelgelerinde en alt satırlarına eklenmiştir. Ortalama değerler karşılaştırıldığında,

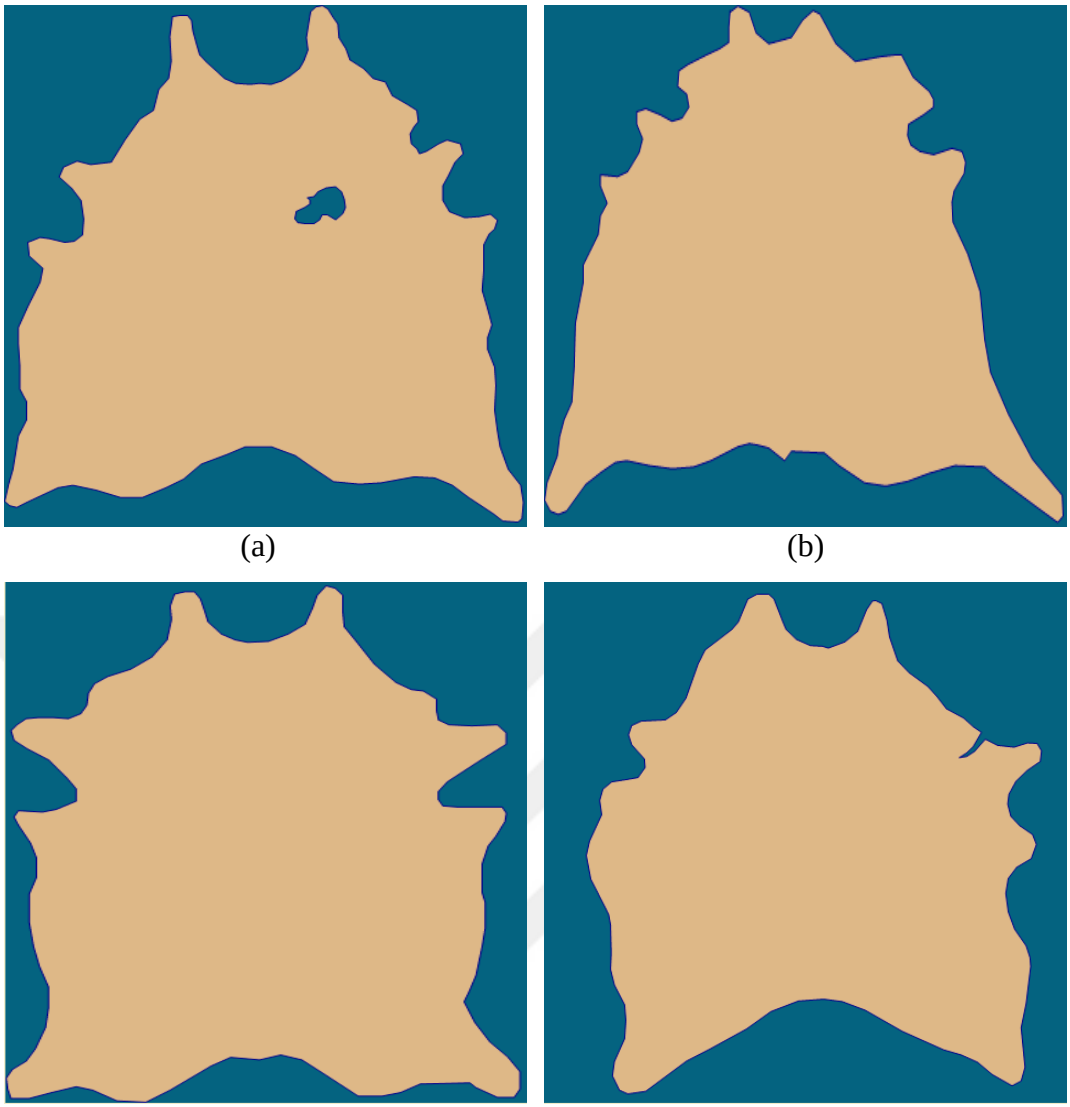
sadece 0^o döndürme açısı olan durumda büyük kalıplar daha iyi sonuç vermiştir. Ancak döndürmelere izin verilen diğer tüm açı değerlerinde, materyalin tüm girinti ve çıkıntılarına küçük parçalar döndürülerek yerleştirilebildiği için daha fazla verimler elde edilmiştir. Kalıpların boyutları küçüldükçe ve şekilleri düzgünleştikçe elde edilen verimler de artmaktadır.

4.4.2. Birden fazla materyal ve kalıp örneği ile yöntemlerin karşılaştırılması

Tek bir materyal ve tek bir kalıp örneği ile yöntemlerin karşılaştırılması bir önceki başlıkta verilmiştir. Bu başlık altında birden fazla materyal ve kalıp örnekleri için kullanılan optimizasyon yöntemlerinin karşılaştırılması sonuçları verilmiştir. Şekil 4.46'da kullanılan materyal örnekleri, Şekil 4.47'de ise kullanılan kalıp modelleri verilmiştir.

Ayrıca Çizelge 4.6, 4.7, 4.8, 4.9, 4.10, 4.11, 4.12, 4.13 ve 4.14'te sırasıyla, YAK, TB, GA, PSO, SÖA, GKO, DG, L-SHADE ve HPSGKO yöntemleri için elde edilen sonuçlar gösterilmiştir. Çizelgelerde yer alan verim oranları, son kalıbın yerleştirilmesi sonucunda, materyalin son kalıbın bittiği hizaya kadar ki bölümünün kullanım yüzdesini, süreler ise ortalama olarak her bir kalıbın dizilme süresini vermektedir.

Şekil 4.47'de gösterilen kalıp örnekleri, bir çift ayakkabı için gerekli olan model kalıplarıdır. Yani her çift ayakkabı için bir adet sağ ve bir adet sol ayak için olmak üzere her kalıp modeli ikişer adetten oluşmaktadır. Test işlemleri için, kullanılan materyallerden kaç çift ayakkabı diziliminin yapıldığı çizelgelerde belirtilmiştir.

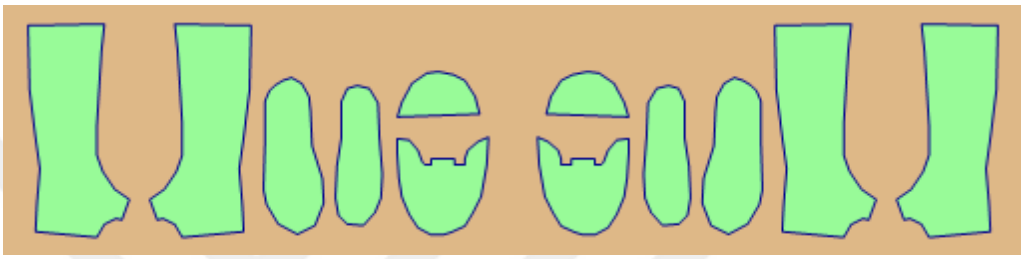


Şekil 4.46. Kullanılan materyal örnekleri (a) 1 nolu materyal (b) 2 nolu materyal (c) 3 nolu materyal (d) 4 nolu materyal

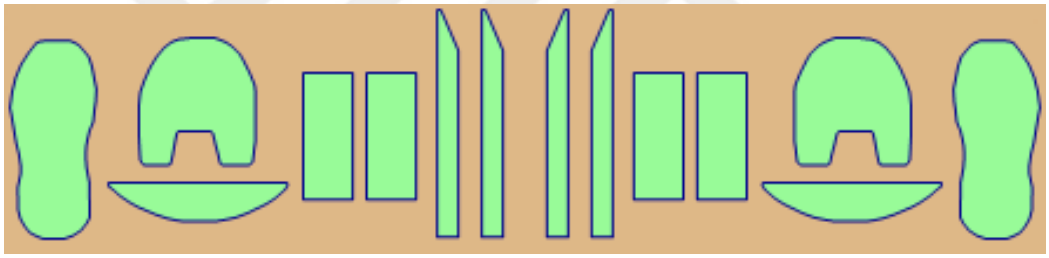
Şekil 4.46'da gösterilen materyaller, olabildiğince karşılaşılabilecek materyal örneklerinden seçilmiştir. Yani hiçbir sorunu olmayan materyal (Şekil 4.46.b ve Şekil 4.46.c), ortasında bir yırtık ya da deformeden dolayı delik olan model (Şekil 4.46.a) ve her hangi bir kenarında yırtık olan model (Şekil 4.46.d)'lerden oluşmaktadır.

(a)

(b)



(c)



(d)

Şekil 4.47. Kullanılan ayakkabı modelleri örnekleri (a) 1 nolu model kalıpları (b) 2 nolu model kalıpları (c) 3 nolu model kalıpları (d) 4 nolu model kalıpları

Örnek materyal ve kalıp örnekleri için elde edilen sonuçlar aşağıda gösterilmiştir. Belirtilen süreler, toplam dizilim için geçen süreyi ifade etmektedir.

Çizelge 4.6. YAK yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	114,00	106,60	105,00	110,60
			Verim (%)	46,79	46,10	43,23	45,86
Model 2	10	5	Süre (sn)	353,00	349,50	360,00	291,50
			Verim (%)	51,46	52,94	50,51	52,71
Model 3	12	2	Süre (sn)	158,64	134,40	166,32	127,20
			Verim (%)	55,04	57,22	51,96	53,16
Model 4	14	3	Süre (sn)	285,18	271,32	300,30	301,14
			Verim (%)	47,91	53,51	51,81	50,40

Çizelge 4.7. TB yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	264,60	218,80	226,80	193,20
			Verim (%)	53,69	54,02	52,02	54,14
Model 2	10	5	Süre (sn)	620,00	574,50	661,00	616,50
			Verim (%)	64,88	62,36	67,52	**68,91**
Model 3	12	2	Süre (sn)	275,04	264,00	265,44	233,28
			Verim (%)	64,86	61,22	67,22	62,58
Model 4	14	3	Süre (sn)	548,52	472,92	482,16	472,50
			Verim (%)	60,93	**70,04**	63,35	65,16

Çizelge 4.8. GA yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	180,00	164,00	160,80	147,20
			Verim (%)	48,44	46,73	43,17	46,56
Model 2	10	5	Süre (sn)	494,50	507,50	462,00	437,50
			Verim (%)	57,04	62,92	59,38	58,11
Model 3	12	2	Süre (sn)	189,36	183,36	193,92	173,04
			Verim (%)	57,52	63,03	58,61	56,98
Model 4	14	3	Süre (sn)	421,26	416,64	370,44	352,38
			Verim (%)	52,53	57,02	54,57	53,42

Çizelge 4.9. PSO yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	70,20	62,00	63,00	62,80
			Verim (%)	50,46	55,41	54,86	54,48
Model 2	10	5	Süre (sn)	238,50	227,50	231,50	223,50
			Verim (%)	64,29	66,03	67,32	65,53
Model 3	12	2	Süre (sn)	100,08	93,36	99,36	98,64
			Verim (%)	61,85	66,11	65,78	64,08
Model 4	14	3	Süre (sn)	161,70	151,20	168,00	165,06
			Verim (%)	63,79	64,16	65,54	64,75

Çizelge 4.10. SÖA yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	125,00	117,80	102,80	105,60
			Verim (%)	49,01	49,45	51,22	48,76
Model 2	10	5	Süre (sn)	342,50	305,00	360,00	350,50
			Verim (%)	57,79	54,62	59,19	57,46
Model 3	12	2	Süre (sn)	150,96	133,44	151,44	118,32
			Verim (%)	59,60	63,24	59,31	58,50
Model 4	14	3	Süre (sn)	301,98	264,60	310,38	265,02
			Verim (%)	53,39	56,87	60,68	56,28

Çizelge 4.11. GKO yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	71,60	61,00	64,60	59,60
			Verim (%)	50,66	53,70	52,99	54,36
Model 2	10	5	Süre (sn)	238,00	236,50	241,00	222,50
			Verim (%)	62,23	66,54	65,80	65,26
Model 3	12	2	Süre (sn)	99,84	91,20	96,72	96,24
			Verim (%)	63,69	66,51	67,14	65,86
Model 4	14	3	Süre (sn)	155,82	150,36	156,24	149,52
			Verim (%)	63,28	63,20	66,60	66,49

Çizelge 4.12. DG yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	145,40	131,20	138,80	132,40
			Verim (%)	55,09	**57,41**	**55,95**	55,27
Model 2	10	5	Süre (sn)	371,50	357,00	359,50	336,50
			Verim (%)	61,57	64,57	66,24	61,17
Model 3	12	2	Süre (sn)	156,24	146,88	172,80	141,84
			Verim (%)	63,03	63,20	69,48	65,05
Model 4	14	3	Süre (sn)	343,56	341,04	328,44	333,90
			Verim (%)	63,82	65,73	66,94	66,54

Çizelge 4.13. L-SHADE yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	71,60	65,40	78,60	65,00
			Verim (%)	39,42	43,88	42,14	41,16
Model 2	10	5	Süre (sn)	182,00	179,00	213,50	167,50
			Verim (%)	46,45	47,99	46,39	46,64
Model 3	12	2	Süre (sn)	94,80	67,68	76,56	83,04
			Verim (%)	50,39	53,78	53,34	53,06
Model 4	14	3	Süre (sn)	157,92	166,32	152,46	142,80
			Verim (%)	50,53	50,09	48,29	49,59

Çizelge 4.14. HPSGKO yöntemi ile elde edilen sonuçlar

Model	Kalıp Sayısı	Çift Sayısı	Kriter	Materyal 1	Materyal 2	Materyal 3	Materyal 4
				Performans			
Model 1	4	5	Süre (sn)	193.98	142.18	150.24	142.39
			Verim (%)	**55,12**	56,29	55,45	**55,74**
Model 2	10	5	Süre (sn)	395.50	371.67	386.32	371.66
			Verim (%)	**65,97**	**67,66**	**69,00**	68,24
Model 3	12	2	Süre (sn)	182.50	158.38	168.52	159.91
			Verim (%)	**66,27**	**68,98**	**69,78**	**67,10**
Model 4	14	3	Süre (sn)	382.70	358.40	374.23	358.81
			Verim (%)	**67,85**	69,64	**69,52**	**69,34**

Yukarıda verilen çizelgelerde, en iyi elde edilen sonuçlar **bold** olarak işaretlenmiştir. Çizelgeler incelendiğinde, HPSGKO algoritmasının en fazla sayıda iyi sonuç verdiği görülmektedir. Toplam 16 farklı testin 12'sinde HPSGKO algoritması en iyi sonucu bulmuştur. DG ve TB algoritmaları da, HPSGKO algoritmasını takip etmektedir. Ancak özellikle TB algoritmasının hız olarak çok geride kalması elde ettiği başarıları gölgede bırakmaktadır. Çünkü GKO ve PSO algoritmaları her ne kadar DG ve TB algoritmalarının gerisinde kalsa da çok yakın sonuçlar elde edebilmişlerdir. TB yöntemi geliştirilerek çok daha hızlı çalışması sağlanabilirse hem PSO hem de GKO yöntemlerine alternatif olabilecektir. DG ve HPSGKO algoritmaları süre olarak birbirlerine yakın çalışmaktadırlar. Ancak HPSGKO algoritmasının verim bakımında elde ettiği başarılar dikkate alınırsa, DG algoritması HPSGKO algoritmasına rakip olamamaktadır.

GKO ve PSO yöntemlerinin sonuçları incelenirse, verim açısından büyük farkların olmadığı görülmektedir. Birbirlerini geçtikleri durumların çoğunda küçük farklar ile ön plana çıkmaktadırlar. Süre dikkate alındığında, GKO yöntemi PSO yönteminden küçük farklarla önde olduğu görülmektedir. HPSGKO algoritması tüm test sonuçlarında, GKO ve PSO algoritmalarından daha iyi sonuç vermiştir. GKO ve PSO algoritmaları her ne kadar daha hızlı çalışsa da HPSGKO algoritmasına alternatif olamayacaklardır.

EK A'da verilen detaylı analiz işlemleri incelenirse, toplam altı farklı açı değeri ve dört farklı yön olmak üzere 24 adet test işlemi gerçekleştirilmiştir. Bu test işlemleri sonucunda, algoritmalar tarafından elde edilen en iyi değerler tablolarda **bold** olarak belirtilmiştir. Tüm testler göz önünde bulundurulduğunda, HPSGKO algoritmasının 24 testin, 13 tanesinde en iyi sonuçları bulmuştur. DG ve PSO algoritmaları ise dörder testte en iyi sonuçları bularak HPSGKO algoritmasının arkasından gelmektedir. Son olarak GKO algoritması ise üç testte en iyi sonucu bulmuş, DG ve PSO algoritmalarına yakın olarak üçüncü sırada yer almıştır. Diğer yöntemler ise hiçbir testte, HPSGKO, DG, PSO ve GKO algoritmalarını geçemeyerek, bu problem için en son sırada yer almaktadırlar.

4.4.3. El ile dizilim ve otomatik dizilimin karşılaştırılması

Geliştirilen yazılımın, hem el ile hem de otomatik olarak dizilim işlemi yapabildiği önceki başlıklarda belirtilmişti. Geliştirilen uygulamada aynı veri seti için hem el ile olarak, hem de otomatik olarak dizilim işlemleri gerçekleştirilerek sonuçlar karşılaştırılmıştır. Çizelge 4.15'te kullanılan optimizasyon yöntemlerine karşılık el ile dizilim için elde edilen sonuçlar verilmiştir.

Çizelge 4.15. El ile dizilim ve otomatik dizilimin sonuçlarının karşılaştırılması

	0^o		1^o		30^o		90^o	
Yöntem	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)	Verim (%)	Süre (sn.)
YAK	53,88	22,28	54,96	22,23	57,27	19,79	57,89	19,62
TB	62,67	21,35	61,45	22,44	65,14	22,57	61,45	20,78
GA	48,12	14,94	42,50	15,12	42,20	14,53	40,43	14,42
PSO	71,67	6,69	66,70	6,81	66,31	6,59	67,05	6,58
SÖA	53,92	9,96	47,87	10,94	48,82	9,82	49,76	9,98
GKO	70,64	6,38	65,95	6,55	64,09	6,29	63,63	6,27
DG	68,54	11,19	65,70	11,31	69,94	13,81	67,87	11,03
L-SHADE	54,69	19,65	48,70	20,03	52,49	19,90	56,69	20,11
HPSGKO	71,69	15,41	67,33	12,82	70,70	14,96	67,50	15,38
Manuel	**63,62**	**7,40**	**64,53**	**17,40**	**64,48**	**6,59**	**64,39**	**7,52**

Çizelge 4.15'te el ile dizilim ile elde edilen sonuçlar verilmiştir. Çizelgeden görüldüğü gibi el ile dizilim, optimizasyon yöntemlerinin yarıdan fazlası ile elde edilen verim değerlerini hiçbir hassasiyet değerinde geçememiştir. Ortalama sürelerde dikkate alındığında iyi sonuçlar veren optimizasyon yöntemlerinin çok gerisinde kaldığı görülmektedir. Önceki çizelgelerde göz önünde bulundurularak en performanslı algoritmaların HPSGKO, DG, PSO ve GKO yöntemleri olduğu anlaşılmaktadır. El ile dizilim, HPSGKO, DG, PSO ve GKO yöntemleri ile karşılaştırıldığında hem süre hem de verim açısından geride kalmaktadır. Geliştirilen yazılımın günümüz fabrikalarında el ile yapılan işlemleri hem hızlandıracağı hem de daha verimli hale getireceği çizelgelerden anlaşılmaktadır.

5. TARTIŞMA VE SONUÇLAR

Bu tez çalışmasında, deri ayakkabı üretiminde kullanılmak üzere, iki boyutlu çokgenlerle ifade edilen ayakkabı kalıplarının, yine iki boyutlu çokgenle ifade edilen deri materyali üzerine en az fire verecek şekilde en uygun yerleştirilmesi işlemi gerçekleştirilmiştir.

En uygun yerleştirme işlemi yapılırken dokuz adet farklı optimizasyon yöntemi kullanılmıştır. Bu yöntemlerde DG (güncel versiyonu), L-SHADE, GKO ve SÖA yöntemleri literatürde yeni oldukları için daha önce iki boyutlu yerleşim optimizasyonu problemine uyarlanmamıştır. Bu tez çalışmasıyla birlikte bu yöntemlerin iki boyutlu yerleşim optimizasyonu problemlerinde kullanılabilirliği gösterilmiştir. Bu yönüyle literatüre yeni bir katkı yapılmış olmaktadır.

Ayrıca bu tez kapsamında GKO ve PSO algoritmaları kullanılarak yeni bir hibrit algoritma (HPSGKO) geliştirilmiştir. Geliştirilen algoritmada, PSO algoritmasının global arama özelliğinin iyileştirmesi amaçlanmıştır. Yerel arama işleminde başarılı olan PSO algoritmasına, global arama özelliğinde başarılı olan GKO algoritması entegre edilerek hibrit bir yaklaşım geliştirilmiştir. HPSGKO algoritmasının test sonuçları dikkate alındığında GKO ve PSO algoritmalarından çok daha başarılı olduğu görülmüştür. Literatüre GKO ve PSO algoritmalarından daha iyi sonuçlar üretebilen bir hibrit algoritma kazandırılmıştır.

Ayrıca bu tez kapsamında, sadece optimizasyon probleminin çözülmesi değil aynı zamanda kullanıcıların çok rahat kullanabileceği bir uygulama geliştirilmiştir. Ülkemizde deri ayakkabı üretimi yapan firmaların genel olarak yazılımı dış ülkelerden satın aldıkları giriş bölümünde anlatılmıştı. Bu yönüyle ülkemizde, deri ayakkabı üretiminde kullanılmak üzere yerli bir yazılım geliştirilerek bu alandaki açık büyük oranda kapatılmış olmaktadır.

Geliştirilen yazılım dokuz farklı optimizasyon yöntemi ile çalışabilme kapasitesine sahiptir. Her bir yöntem yapılan analizler sonucunda karşılaştırılmış ve aşağıda ki sonuçlar elde edilmiştir.

- HPSGKO algoritması bu tez kapsamında literatüre kazandırılan ve kullanılan tüm yöntemler içinde en başarılı olan algoritmadır. HPSGKO algoritmasının, optimum yerleşim problemlerinde rahatlıkla kullanılabileceği görülmüştür.
- GKO ve PSO algoritmaları, hem süre hem de verim açısından benzer sonuçlar üretmiştir. Süre olarak en hızlı çalışan algoritmalardandırlar. Fakat elde ettikleri verimler dikkate alındığında, özellikle HPSGKO algoritmasının gerisinde kaldıkları ve bu tez çalışmasına konu olan problem için HPSGKO ve DG algoritmalarından sonraki en iyi yöntemlerdir. HPSGKO algoritmasına alternatif olamayıp, DG algoritmasına hızlı çalıştıkları için alternatif olarak değerlendirilebilirler.
- SÖA ve TB algoritmaları verim bakımından az sayıda testte, en iyi sonuçlara ulaşabilmektedirler. PSO ve GKO algoritmalarının elde edemediği verimleri SÖA ve özellikle TB yöntemleri her zaman olmasa da elde edebilmektedirler. Ancak algoritmaların çalışma hızları çok yavaş olduklarından PSO ve GKO algoritmalarına bu problem için alternatif olamamaktadırlar.
- GA, YAK ve L-SHADE algoritmaları ile yapılan analizler diğer yöntemler ile karşılaştırıldıklarında çok geride kaldıkları görülmektedir. Bu problem için uygun algoritma olmadıkları analiz sonuçlarından anlaşılmaktadır.

Sonuç olarak, yukarıda bahsedilen tüm durumlar birlikte dikkate alındığında, bu tez çalışmasında ele alınan problem için en iyi yöntem, bu tez kapsamında geliştirilen HPSGKO algoritmasıdır. DG algoritması da elde ettiği verimler bakımından GKO ve PSO algoritmalarının önünde yer almaktadır. GKO ve PSO algoritmaları hızlı çalıştıkları için DG algoritmasına bir alternatif olarak düşünülebilir. GKO algoritması literatüre girdiği günden bugüne ilk defa yerleşim optimizasyonu problemine uyarlanmıştır. Aynı şekilde SÖA algoritması da literatüre yeni girmiş ve yerleşim optimizasyonu problemine ilk defa bu tez çalışmasıyla birlikte uyarlanarak, yerleşim optimizasyonu problemlerinde kullanılabileceği gösterilmiştir.

Gelecekteki çalışmalarda, HPSGKO algoritmasının daha hızlı çalışabilmesi için iyileştirmelerin yapılması ve her açıdan GKO ve PSO algoritmalarına bir alternatif olmasının sağlanması hedeflenmektedir. Ayrıca bu tez çalışmasının konusu olan yerleşim probleminin haricindeki optimizasyon problemlerinde de, HPSGKO algoritmasının daha iyi çalışabileceğinin gösterilmesi amaçlanmaktadır.

KAYNAKLAR

Abeysooriya, R. P., Bennell, J. A., Martinez-Sykora, A. , 2018. Jostle heuristics for the 2D-irregular shapes bin packing problems with free rotation. International Journal of Production Economics, 195, 12–26.

Adakcı, S. Stok Kesme Problemi: Alüminyum Sektöründe Uygulaması. İstanbul Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2010, 137s, İstanbul.

Akbulut, D. An edge matching approach for two-dimensional irregular shaped cutting stock problems. Çankaya Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2012, 93s, Ankara.

Albano, A., Sapuppo, G. , 1980. Optimal Allocation of Two-Dimensional Irregular Shapes Using Heuristic Search Methods. IEEE Transactions on Systems, Man, and Cybernetics, 10(5), 242–248.

Barak, M. Z. E. Sipariş Kabul ve Çizelgeleme Probleminin Parçacık Sürü Optimizasyonu ile Çözülmesi. Çukurova Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2015, 147s, Adana.

Bayır, F. Kesme problemine sezgisel bir yaklaşım (Doktora Tezi). İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü, Doktora Tezi, 2012, 162s, İstanbul.

Chopra, N., Kumar, G., Mehta, S. , 2016. Hybrid GWO-PSO algorithm for solving convex economic load dispatch problem. International journal of research in advanced technology, 4(6), 37–41.

Crispin, A., Clay, P., Taylor, G., Bayes, T., Reedman, D. , 2005. Genetic Algorithm Coding Methods for Leather Nesting. Applied Intelligence, 23(1), 9–20.

Dagli, C. H., Hajakbari, A. , 1990. Simulated annealing approach for solving stock cutting problem. 1990 IEEE International Conference on Systems, Man, and Cybernetics Conference Proceedings (ss. 221–223). IEEE.

Dalalah, D., Khrais, S., Bataineh, K. , 2014. Waste minimization in irregular stock cutting. Journal of Manufacturing Systems, 33(1), 27–40.

Domoviç, D., Rolich, T., Grundler, D., Bogoviç, S., Domovic, D., Rolich, T., Grundler, D., Bogovic, S. , 2014. Algorithms for 2D nesting problem based on the no-fit polygon. 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO 2014 - Proceedings, (May), 1094–1099.

Durak, E. S. Çok Seviyeli Eviricilerde Anahtarlama Açılarının Optimizasyonu. Karadeniz Teknik Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2016, 112s, Trabzon.

Elamvazuthi, I., Kamaruddin, S., Azmi, M. S. , 2009. Automation of Nesting and Cutting Processes of Leather Furniture Production: A Case Study. International Journal of Mechanical & Mechatronics Engineering IJMM E-IJENS, 09(10), 14–18.

Elkeran, A. , 2013. A new approach for sheet nesting problem using guided cuckoo search and pairwise clustering. European Journal of Operational Research, 231(3), 757–769.

Gomes, A. M., Oliveira, J. F. , 2002. A 2-exchange heuristic for nesting problems. European Journal of Operational Research, 141(2), 359–370.

Gomes, A. M., Oliveira, J. F. , 2006. Solving Irregular Strip Packing problems by hybridising simulated annealing and linear programming. European Journal of Operational Research, 171(3), 811–829.

Holland, J. , 1975. Adaptation in Natural and Artificial System. University of Michigan Press, 232s, Michigan.

Imamichi, T. Nonlinear Programming Based Algorithms to Cutting and Packing Problems. Kyoto University,Department of Applied Mathematics and Physics Graduate School of Informatics, PhD Thesis, 2009, 111s, Kyoto.

Kamboj, V. K. , 2016. A novel hybrid PSO–GWO approach for unit commitment problem. Neural Computing and Applications, 27(6), 1643–1655.

Karaboga, D., Basturk, B. , 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of Global Optimization, 39(3), 459–471.

Kennedy, J., Eberhart, R. , 1995. Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks (C. 4, ss. 1942–1948). IEEE.

Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P. , 1987. Optimization by Simulated Annealing. Readings in Computer Vision içinde (ss. 606–615).

Koç, İ. Sınıflandırma Problemlerinde Meta-Sezgisel Optimizasyon Yöntemlerinin Özellik Seçimi ve Ayrıklaştırma Amacıyla Kullanımı. Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2016, 191s, Konya.

Lee, W.-C., Ma, H., Cheng, B.-W. , 2008. A heuristic for nesting problems of irregular shapes. Computer-Aided Design, 40(5), 625–633.

Liu, H., He, Y. , 2006. Algorithm for 2D irregular-shaped nesting problem based on the NFP algorithm and lowest-gravity-center principle. Journal of Zhejiang University-SCIENCE A, 7(4), 570–576.

Mahmoud, A. F., Samia, M., Eid, S., Bahnasawi, A. , 2003. Genetic algorithms for solving 2D cutting stock problem. 2003 46th Midwest Symposium on Circuits and Systems (C. 2, ss. 956–960). IEEE.

Marques, V. M. M., Bispo, C. F. G., Sentieiro, J. J. S. , 1991. A system for the compaction of two-dimensional irregular shapes based on simulated annealing. Proceedings IECON '91: 1991 International Conference on Industrial Electronics, Control and Instrumentation (ss. 1911–1916). IEEE.

Martínez Sykora, A. Nesting Problems : Exact and Heuristic Algorithms. University of Valencia, Departamento de Estadística e Investigación Operativa, PhD Thesis, 2013, 187s, Valencia.

Mirjalili, S. , 2015. How effective is the Grey Wolf optimizer in training multi-layer perceptrons. Applied Intelligence, 43(1), 150–161.

Mirjalili, S. M., Lewis, A. , 2014. Grey Wolf Optimizer. Advances in Engineering Software, 69, 46–61.

Mohamed, A. W., Suganthan, P. N. , 2017. Real-parameter unconstrained optimization based on enhanced fitness-adaptive differential evolution algorithm with novel mutation. Soft Computing, 1–21.

Özcan, M. Atölye Tipi Çizelgeleme Problemlerinde Evrimsel Algoritmalar ile Yapay Arı Kolonisi Algoritmasının Bütünleşik Bir Yaklaşımı. Sakarya Üniversitesi, Fen Bilimleri Enstitüsü, Doktora Tezi, 2016, 192s, Sakarya.

Paquay, C., Limbourg, S., Schyns, M. , 2018. A tailored two-phase constructive heuristic for the three-dimensional Multiple Bin Size Bin Packing Problem with transportation constraints. European Journal of Operational Research, 267(1), 52–64.

Ramakrishnan, K., Bennell, J. A., Mohamed K., O. , 2008. Solving two dimensional layout optimization problems with irregular shapes by using meta-heuristic. 2008 IEEE International Conference on Industrial Engineering and Engineering

Management (ss. 178–182). IEEE.
Rezai, S. , 2017. Elektrostatik Tabanlı Çok Tepeli Parçacık Sürü Optimizasyon Algoritması ile Çoklu Daire Algılama. Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2017, 83s, Ankara.
Rocha, P. F. M. Geometrical Models and Algorithms for Irregular Shapes Placement Problems. Porto University, Informatics Engineering, PhD Thesis, 2014, 184s, Porto.
Rocha, P., Rodrigues, R., Gomes, A. M., Toledo, F. M. B., Andretta, M. , 2015. Two-Phase Approach to the Nesting problem with continuous rotations. IFAC-PapersOnLine, 48(3), 501–506.
Rutenbar, R. A. , 1989. Simulated annealing algorithms: an overview. IEEE Circuits and Devices Magazine, 5(1), 19–26.
Sato, A. K., Martins, T. C., Tsuzuki, M. S. G. , 2012. An algorithm for the strip packing problem using collision free region and exact fitting placement. Computer-Aided Design, 44(8), 766–777.
Sato, A. K., Tsuzuki, M. de S. G., Martins, T. de C., Gomes, A. M. , 2015. Multiresolution based overlap minimization algorithm for irregular packing problems. IFAC-PapersOnLine, 48(3), 484–489.
Sato, A. K., Tsuzuki, M. de S. G., Martins, T. de C., Gomes, A. M. da F. F. , 2014. Irregular Packing Overlap Minimization Using Discrete Voronoi Mountain. IFAC Proceedings Volumes, 19(1), 5241–5246.
Şenel, F. A., Gökçe, F., Yüksel, A. S., Yiğit, T. , 2018. A novel hybrid PSO–GWO algorithm for optimization problems. Engineering with Computers, In press, 1–15.
Shalaby, M. A., Kashkoush, M. , 2013. A Particle Swarm Optimization Algorithm for a 2-D Irregular Strip Packing Problem. American Journal of Operations Research, 03(02), 268–278.
Sherif, S. U., Jawahar, N., Balamurali, M. , 2014. Sequential optimization approach for nesting and cutting sequence in laser cutting. Journal of Manufacturing Systems, 33(4), 624–638.
Siasos, A., Vosniakos, G.-C. , 2014. Optimal directional nesting of planar profiles on fabric bands for composites manufacturing. CIRP Journal of Manufacturing Science and Technology, 7(3), 283–297.
Singh, N., Singh, S. B. , 2017. Hybrid Algorithm of Particle Swarm Optimization and Grey Wolf Optimizer for Improving Convergence Performance. Journal of Applied Mathematics.
Song, X., Tang, L., Zhao, S., Zhang, X., Li, L., Huang, J., Cai, W. , 2015. Grey Wolf Optimizer for parameter estimation in surface waves. Soil Dynamics and Earthquake Engineering, 75, 147–157.
Storn, R., Price, K. , 1997. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. Journal of Global Optimization, 11(4), 341–359.
Tanabe, R., Fukunaga, A. , 2013. Success-history based parameter adaptation for Differential Evolution. 2013 IEEE Congress on Evolutionary Computation (ss. 71–78). IEEE.
Tanabe, R., Fukunaga, A. S. , 2014. Improving the search performance of SHADE using linear population size reduction. 2014 IEEE Congress on Evolutionary Computation (CEC) (ss. 1658–1665). IEEE.
Timmerman, M. , 2013. Optimization methods for nesting problems. University West, Department of Engineering Science, Master Thesis, 2013, 49s, Trollhättan.

Tunç, A. , 2016. Finans Sektörü için Yapay Öğrenme Teknikleri Kullanarak Kredi Kullanabilirliğin Tespiti (Yüksek Lisans Tezi). Selçuk Üniversitesi, Fen Bilimleri Enstitüsü, Yüksek Lisans Tezi, 2016, 73s, Konya.

Valvo, E. Lo. , 2017. Meta-heuristic Algorithms for Nesting Problem of Rectangular Pieces. Procedia Engineering, 183, 291–296.

Vassiliadis, V. S. , 2005. Two-dimensional stock cutting and rectangle packing: binary tree model representation for local search optimization methods. Journal of Food Engineering, 70(3), 257–268.

Vijay Anand, K., Ramesh Babu, A. , 2015. Heuristic and genetic approach for nesting of two-dimensional rectangular shaped parts with common cutting edge concept for laser cutting and profile blanking processes. Computers & Industrial Engineering, 80, 111–124.

Wäscher, G., Haußner, H., Schumann, H. , 2007. An improved typology of cutting and packing problems. European Journal of Operational Research, 183(3), 1109–1130.

Weng, W.-C., Kuo, H.-C. , 2011. Irregular stock cutting system based on AutoCAD. Advances in Engineering Software, 42(9), 634–643.

Whitwell, G. , 2004. Novel Heuristic and Metaheuristic Approaches to Cutting and Packing. University of Nottingham, School of Computer Science and Information Technology, PhD Thesis, 2004, 314s, Nottingham.

Wong, W. K., Wang, X. X., Mok, P. Y., Leung, S. Y. S., Kwong, C. K. , 2009. Solving the two-dimensional irregular objects allocation problems by using a two-stage packing approach. Expert Systems with Applications, 36(2), 3489–3496.

Yang, H.-H., Lin, C.-L. , 2009. On genetic algorithms for shoe making nesting – A Taiwan case. Expert Systems with Applications, 36(2), 1134–1141.

Yang, Q. , 2014. No Fit Polygon for Nesting Problem Solving with Hybridizing Ant Algorithms. Journal of Software Engineering and Applications, 07(05), 433–439.

Yiğit, V. , 2004. Kapasite Kısıtsız Tesis Yerleşim Problemleri için Evrimsel Yaklaşımlı Tavlama Benzetimi Algoritması. Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Doktora Tezi, 2004, 240s, Ankara.

Yu, J. J. Q., Li, V. O. K. , 2014. Base station switching problem for green cellular networks with Social Spider Algorithm. 2014 IEEE Congress on Evolutionary Computation (CEC) (ss. 2338–2344). IEEE.

Yu, J. J. Q., Li, V. O. K. , 2015. A social spider algorithm for global optimization. Applied Soft Computing, 30, 614–627.

YuPing, Z., Caijun, Y. , 2009. A Generic Approach for Leather Nesting. 2009 Fifth International Conference on Natural Computation (ss. 303–307). IEEE.

Yuping, Z., Shouwei, J., Chunli, Z. , 2005. A very fast simulated re-annealing algorithm for the leather nesting problem. The International Journal of Advanced Manufacturing Technology, 25(11–12), 1113–1118.

EKLER

EK A. Farklı açı ve farklı dizilim yönleri ile analiz sonuçlarının verim karşılaştırma sonuçları

EK B. Farklı açı ve farklı dizilim yönleri ile analiz sonuçlarının süre karşılaştırma sonuçları

EK A. Farklı açı ve farklı dizilim yönleri ile analiz sonuçlarının verim karşılaştırma sonuçları

Tezin bu bölümünde, örnek bir materyal ve örnek bir kalıp için, geliştirilen programın analiz işlemleri gerçekleştirilmiştir. Her optimizasyon yöntemi 10'ar defa çalıştırılarak sonuçlar değerlendirilmiştir. Böylece optimizasyon yöntemlerinin detaylı olarak karşılaştırılması yapılabilmiştir. Ayrıca optimum yerleşim probleminde dizilime başlanan yönün sonuca olan etkisi de incelenmiştir. Materyale dört farklı yönden dizilim analizleri yapılmıştır. Her yön için sabit açılarla sonuçlar elde edilmiştir. Bu sonuçlar tablolar halinde aşağıda verilmiştir.

Çizelge Ek A.1. 0º, 1º ve 30º ve farklı dizilim yönleri için YAK, TB ve GA verim sonuçları

Yöntemler	Değer (%)	Döndürme Açısı											
		0º				1º				30º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
YAK	En iyi	58,29	60,01	59,98	56,72	57,53	53,84	59,11	61,87	59,54	54,48	60,75	61,53
	En kötü	48,68	48,34	51,56	47,99	50,73	47,81	47,61	48,74	49,58	47,60	30,94	52,54
	Ortalama	54,60	53,62	55,46	53,88	54,06	50,42	53,25	54,96	54,89	51,00	53,58	57,27
	Standart Sapma	3,71	3,90	2,91	2,71	2,17	2,03	3,06	4,29	2,80	2,53	8,23	2,58
TB	En iyi	65,79	63,07	64,57	70,97	62,27	61,62	63,05	68,14	64,46	65,76	68,37	70,68
	En kötü	52,39	55,03	15,86	58,75	29,48	24,71	55,11	57,25	39,35	22,34	46,98	55,52
	Ortalama	59,88	58,83	52,59	62,67	55,06	54,84	60,10	61,45	58,47	57,79	62,56	65,14
	Standart Sapma	4,64	2,72	13,54	3,41	9,36	10,87	2,97	3,42	7,54	12,71	6,12	4,28
GA	En iyi	57,25	51,94	54,19	53,46	55,70	45,00	46,02	50,96	52,18	53,61	51,24	47,87
	En kötü	43,43	39,32	46,50	42,10	42,83	32,85	32,91	36,04	40,92	45,37	39,06	38,61
	Ortalama	49,13	46,75	50,03	48,12	47,44	40,13	40,80	42,50	47,01	48,51	44,49	42,20
	Standart Sapma	3,89	3,75	2,50	3,78	3,53	4,24	3,59	5,15	3,64	2,79	3,96	2,84

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek A.2. 60º, 90º ve 180º ve farklı dizilim yönleri için YAK, TB ve GA verim sonuçları

Yöntemler	Değer (%)	Döndürme Açısı											
		60º				90º				180º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
YAK	En iyi	61,57	53,99	65,15	66,02	59,98	59,15	61,15	67,33	61,05	54,73	58,29	59,44
	En kötü	47,51	47,82	54,44	54,22	50,36	48,66	49,39	49,70	50,21	50,14	53,65	50,27
	Ortalama	54,63	51,21	58,99	61,19	55,67	52,59	56,58	57,89	54,81	52,08	55,47	56,62
	Standart Sapma	4,13	1,81	3,11	4,01	3,23	3,30	3,56	6,51	3,87	1,66	1,84	2,80
TB	En iyi	64,43	68,97	69,30	72,95	69,04	66,56	71,64	71,26	66,99	62,93	66,37	65,15
	En kötü	52,17	55,68	18,91	51,80	54,23	56,06	52,97	56,95	56,28	54,06	41,17	53,94
	Ortalama	59,72	61,72	62,59	68,35	63,94	61,06	60,70	61,45	62,62	58,26	57,49	60,63
	Standart Sapma	3,70	4,23	15,43	5,99	4,31	3,40	6,05	4,25	3,66	2,72	6,76	2,83
GA	En iyi	53,73	53,56	54,81	54,18	57,06	52,23	48,39	45,94	48,67	52,18	51,45	50,60
	En kötü	43,74	38,89	38,93	43,70	37,21	42,09	35,77	33,90	40,99	41,19	44,31	44,16
	Ortalama	46,75	42,92	48,49	49,06	47,92	46,80	40,15	40,43	45,92	46,93	48,41	47,06
	Standart Sapma	3,13	4,06	4,60	3,47	5,74	3,43	4,48	4,17	2,86	3,15	2,03	2,47

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek A.3. 0º, 1º ve 30º ve farklı dizilim yönleri için PSO, SÖA ve GKO verim sonuçları

Yöntemler	Değer (%)	Döndürme Açısı											
		0º				1º				30º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
PSO	En iyi	64,87	63,17	64,60	71,74	67,61	68,79	68,94	73,40	68,95	70,14	69,96	69,47
	En kötü	58,01	59,76	58,85	71,41	60,00	60,79	57,82	58,43	57,62	56,10	57,07	62,92
	Ortalama	60,50	**60,78**	63,42	71,67	63,58	**63,31**	63,67	66,70	63,57	64,08	65,60	66,31
	Standart Sapma	2,18	1,60	2,06	0,09	2,77	2,66	2,95	4,61	3,48	4,62	3,72	1,89
SÖA	En iyi	60,01	60,42	60,08	59,89	62,45	55,03	64,42	54,41	60,70	53,58	68,10	52,53
	En kötü	53,90	52,21	50,21	51,05	48,81	41,21	53,98	42,08	51,91	41,26	53,72	44,34
	Ortalama	57,82	54,60	55,93	53,92	58,62	46,08	58,13	47,87	56,85	48,99	59,05	48,82
	Standart Sapma	1,94	2,59	3,13	2,43	4,20	4,55	3,05	3,94	2,69	3,47	4,56	2,94
GKO	En iyi	68,06	63,28	68,58	71,72	61,61	61,35	65,92	72,96	65,19	62,80	67,42	69,02
	En kötü	58,97	55,35	15,85	62,47	56,64	54,32	57,81	58,46	54,39	57,48	59,78	58,59
	Ortalama	**61,28**	60,59	59,58	70,64	58,86	57,93	62,45	65,95	60,35	60,52	62,97	64,09
	Standart Sapma	2,88	2,45	15,74	2,87	1,65	2,34	2,96	3,64	3,19	1,98	2,44	3,29

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek A.4. 60º, 90º ve 180º ve farklı dizilim yönleri için PSO, SÖA ve GKO verim sonuçları

Yöntemler	Değer (%)	Döndürme Açısı											
		60º				90º				180º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
PSO	En iyi	71,29	72,28	76,41	77,72	70,35	66,50	73,59	71,93	66,89	64,59	62,82	69,70
	En kötü	61,85	61,22	68,23	70,35	59,66	58,65	62,25	63,53	59,15	54,01	54,84	60,45
	Ortalama	65,31	**64,35**	73,08	74,66	65,66	61,94	**68,23**	67,05	64,51	58,51	58,75	66,70
	Standart Sapma	2,77	3,66	2,17	2,37	3,22	2,33	3,94	2,90	2,63	3,44	2,84	3,93
SÖA	En iyi	62,10	55,14	65,80	61,92	63,13	55,15	65,14	54,07	67,19	60,77	62,58	60,55
	En kötü	51,10	41,15	57,22	50,58	52,70	42,18	54,78	46,41	54,47	50,18	50,66	50,28
	Ortalama	58,54	47,83	61,32	53,73	58,88	49,84	59,75	49,76	59,47	55,97	58,30	56,23
	Standart Sapma	3,52	4,52	2,56	3,39	3,62	4,91	3,60	2,79	4,04	3,03	3,39	3,04
GKO	En iyi	65,82	65,46	72,80	74,32	67,79	67,51	75,74	68,91	65,19	65,47	66,83	69,94
	En kötü	57,55	55,31	67,83	66,88	55,63	53,17	61,64	58,11	57,87	55,71	55,60	54,15
	Ortalama	61,61	61,38	70,81	71,79	62,52	59,87	67,52	63,63	60,84	**60,32**	**60,88**	65,22
	Standart Sapma	2,74	3,10	1,87	2,49	4,76	4,56	4,43	3,47	2,10	3,52	3,36	5,80

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek A.5. 0º, 1º ve 30º ve farklı dizilim yönleri için DG, L-SHADE ve HPSGKO verim sonuçları

Yöntemler	Değer (%)	Döndürme Açısı											
		0º				1º				30º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
DG	En iyi	63,35	63,11	60,71	71,70	73,50	65,41	66,32	69,80	70,09	68,43	68,32	72,53
	En kötü	55,59	51,94	51,70	59,16	59,66	57,60	61,07	61,84	60,78	60,63	58,74	65,82
	Ortalama	59,66	59,06	56,33	68,54	**67,30**	62,00	64,41	65,70	**66,59**	63,75	63,51	69,94
	Standart Sapma	2,31	2,91	2,62	5,18	4,09	3,24	1,61	2,80	3,07	2,28	3,02	2,24
L-SHADE	En iyi	61,23	56,39	60,24	62,88	49,86	51,23	53,93	51,79	54,46	53,99	56,91	57,97
	En kötü	52,01	49,25	13,23	48,89	43,51	44,40	44,35	45,92	46,43	19,59	48,37	48,26
	Ortalama	55,99	53,44	46,61	54,69	47,22	46,79	48,19	48,70	50,98	47,93	52,47	52,49
	Standart Sapma	3,47	2,60	17,62	4,85	1,97	2,24	2,97	2,18	2,94	10,24	2,39	3,26
HPSGKO	En iyi	62,73	63,10	68,58	71,69	69,97	69,17	68,59	70,18	69,20	72,83	68,25	73,79
	En kötü	59,11	59,69	60,65	71,69	58,90	60,80	63,17	61,55	64,10	65,19	58,62	65,89
	Ortalama	59,91	60,10	**65,79**	**71,69**	63,31	63,00	**66,70**	**67,33**	66,56	**68,64**	**65,87**	**70,70**
	Standart Sapma	1,44	1,05	3,24	0,00	3,35	2,77	1,81	2,62	1,72	2,61	3,10	2,63

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek A.6. 60º, 90º ve 180º ve farklı dizilim yönleri için DG, L-SHADE ve HPSGKO verim sonuçları

Yöntemler	Değer (%)	Döndürme Açısı											
		60º				90º				180º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
DG	En iyi	66,24	68,86	77,71	77,90	68,12	66,98	72,10	73,04	69,44	62,34	62,45	70,54
	En kötü	57,45	57,61	64,60	69,70	59,89	59,18	60,88	64,90	53,80	54,44	55,13	62,19
	Ortalama	63,51	62,21	70,89	73,62	65,39	**63,06**	65,20	**67,87**	62,87	57,93	58,89	67,81
	Standart Sapma	2,93	3,60	4,17	2,25	2,50	2,36	3,64	2,86	4,57	2,65	1,73	2,93
L-SHADE	En iyi	57,87	57,48	61,49	65,22	63,51	59,50	62,80	60,26	60,22	60,89	57,87	58,92
	En kötü	44,10	22,86	51,25	57,83	51,62	50,99	51,60	50,02	51,06	50,40	47,92	50,12
	Ortalama	50,98	49,90	57,22	60,46	54,99	54,92	56,15	56,69	55,05	54,46	53,50	54,47
	Standart Sapma	4,26	10,36	3,38	2,29	3,51	3,00	3,96	2,76	3,57	2,90	3,01	3,28
HPSGKO	En iyi	70,00	68,16	80,86	78,12	71,50	64,01	68,54	72,99	69,45	65,42	66,15	70,59
	En kötü	63,01	57,41	72,07	73,18	65,13	57,19	64,28	64,77	62,26	51,48	55,12	60,59
	Ortalama	**66,08**	62,80	**75,61**	**75,09**	**67,86**	60,95	66,14	67,50	**65,95**	59,47	59,81	**69,02**
	Standart Sapma	2,93	3,93	2,65	2,03	2,27	2,23	1,39	3,00	2,66	3,71	3,01	2,99

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

EK B. Farklı açı ve farklı dizilim yönleri ile analiz sonuçlarının süre karşılaştırma sonuçları

Tezin Ek A bölümünde yapılan tüm analizlere ilişkin verim karşılaştırma sonuçları verilmişti. Bu bölümde ise yapılan tüm analiz sonuçlarına ilişkin süre karşılaştırmaları yapılmıştır. Tüm sonuçlar tablolar halinde detaylı olarak verilmiştir. Tablolarda verilen süreler, dizilim süresi boyunca ortalama olarak tek bir kalıbın yerleştirilmesi için geçen süreyi ifade etmektedir.

Çizelge Ek B.1. 0º, 1º ve 30º ve farklı dizilim yönleri için YAK, TB ve GA verim sonuçları

Yöntemler	Değer (sn)	Döndürme Açısı											
		0º				1º				30º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
YAK	En iyi	23,02	25,47	23,39	23,56	23,49	24,62	23,38	23,47	21,78	20,83	20,78	20,16
	En kötü	21,97	24,18	21,88	21,51	19,58	20,34	19,60	19,65	18,78	19,70	16,03	19,16
	Ortalama	22,50	24,81	22,45	22,28	22,23	23,49	21,90	22,23	19,71	20,18	19,44	19,79
	Standart Sapma	0,36	0,42	0,44	0,56	1,38	1,52	1,53	1,32	0,80	0,40	1,29	0,29
TB	En iyi	21,72	23,35	22,84	22,15	22,30	24,57	23,96	23,21	22,55	23,06	24,01	23,88
	En kötü	19,39	20,43	13,85	20,57	15,41	14,71	21,23	21,76	16,54	13,98	18,62	20,94
	Ortalama	20,67	21,71	20,22	21,35	20,96	21,51	22,30	22,44	20,83	20,74	22,29	22,57
	Standart Sapma	0,79	0,82	2,41	0,46	2,02	2,62	0,85	0,51	1,79	2,51	1,46	0,86
GA	En iyi	14,99	15,42	15,81	16,26	15,76	15,74	16,66	16,58	15,01	16,48	15,91	15,37
	En kötü	14,18	14,15	14,81	14,24	14,50	13,50	13,96	14,37	13,90	14,41	14,04	14,03
	Ortalama	14,52	14,63	15,13	14,94	15,16	14,53	15,23	15,12	14,53	15,26	14,81	14,53
	Standart Sapma	0,23	0,39	0,35	0,61	0,39	0,74	0,72	0,63	0,36	0,68	0,59	0,50

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek B.2. 60º, 90º ve 180º ve farklı dizilim yönleri için YAK, TB ve GA verim sonuçları

Yöntemler	Değer (sn)	Döndürme Açısı											
		60º				90º				180º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
YAK	En iyi	21,81	24,96	20,66	21,11	19,97	25,12	20,36	20,51	20,21	25,41	22,72	23,16
	En kötü	18,73	21,71	19,14	19,27	17,97	21,79	18,56	18,84	19,12	22,19	19,25	18,99
	Ortalama	19,80	22,91	19,69	20,18	19,42	23,17	19,39	19,62	19,51	23,57	21,04	20,28
	Standart Sapma	0,87	0,95	0,42	0,63	0,59	1,27	0,51	0,60	0,39	1,12	1,56	1,49
TB	En iyi	22,68	22,24	25,16	24,40	21,86	23,17	23,96	23,82	23,17	18,73	28,27	23,29
	En kötü	20,08	20,41	15,26	18,96	19,09	18,07	18,20	17,71	17,25	17,10	17,97	18,24
	Ortalama	21,11	21,34	22,08	22,39	20,81	20,40	20,95	20,78	19,86	17,77	21,18	20,37
	Standart Sapma	0,94	0,59	2,65	1,42	1,00	1,60	1,80	1,96	2,30	0,49	3,18	1,46
GA	En iyi	16,43	15,38	16,75	16,03	16,44	15,73	15,85	15,21	15,56	16,02	16,17	15,95
	En kötü	14,22	13,94	14,21	14,65	13,77	13,96	13,49	13,53	14,19	14,42	14,84	14,68
	Ortalama	14,81	14,56	15,14	15,15	14,77	14,68	14,33	14,42	14,65	14,80	15,13	15,09
	Standart Sapma	0,71	0,41	0,71	0,44	0,85	0,51	0,66	0,55	0,45	0,49	0,39	0,44

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek B.3. 0º, 1º ve 30º ve farklı dizilim yönleri için PSO, SÖA ve GKO verim sonuçları

Yöntemler	Değer (sn)	Döndürme Açısı											
		0º				1º				30º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
PSO	En iyi	6,50	6,45	6,86	7,04	6,71	6,91	7,31	7,12	6,84	7,10	7,15	6,68
	En kötü	6,32	6,35	6,51	6,64	6,51	6,51	6,71	6,54	6,43	6,26	6,67	6,51
	Ortalama	6,43	6,39	6,65	6,69	6,63	6,65	6,90	6,81	6,56	6,61	6,82	6,59
	Standart Sapma	0,05	0,04	0,12	0,12	0,07	0,14	0,17	0,18	0,12	0,26	0,13	0,05
SÖA	En iyi	12,07	10,97	10,31	10,34	11,06	11,80	10,93	18,73	11,24	10,63	11,14	10,05
	En kötü	10,45	10,12	9,67	9,61	10,10	9,81	10,36	9,69	9,98	9,56	10,10	9,55
	Ortalama	11,05	10,43	10,02	9,96	10,57	10,28	10,56	10,94	10,33	10,13	10,35	9,82
	Standart Sapma	0,43	0,24	0,20	0,24	0,26	0,66	0,18	2,75	0,34	0,32	0,31	0,16
GKO	En iyi	6,57	6,35	7,06	6,57	6,58	6,55	6,87	6,97	6,40	6,35	6,75	6,40
	En kötü	6,21	6,17	4,24	6,20	6,25	6,28	6,46	6,27	6,13	6,25	6,33	6,16
	Ortalama	6,29	6,28	6,24	6,38	6,36	6,37	6,62	6,55	6,27	6,31	6,46	6,29
	Standart Sapma	0,11	0,06	0,73	0,09	0,09	0,08	0,15	0,23	0,07	0,04	0,12	0,08

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek B.4. 60º, 90º ve 180º ve farklı dizilim yönleri için PSO, SÖA ve GKO verim sonuçları

Yöntemler	Değer (sn)	Döndürme Açısı											
		60º				90º				180º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
PSO	En iyi	6,86	7,04	6,98	6,86	6,80	6,80	6,95	6,71	6,73	6,54	6,91	6,67
	En kötü	6,49	6,44	6,67	6,61	6,45	6,36	6,58	6,51	6,50	6,29	6,43	6,41
	Ortalama	6,58	6,61	6,86	6,76	6,61	6,50	6,77	6,58	6,59	6,40	6,60	6,60
	Standart Sapma	0,11	0,21	0,08	0,08	0,11	0,14	0,10	0,07	0,07	0,07	0,15	0,09
SÖA	En iyi	10,51	10,49	10,83	10,92	11,34	10,67	10,73	10,41	10,81	10,62	10,78	10,66
	En kötü	9,96	9,63	10,13	9,79	10,17	9,48	9,90	9,59	10,14	9,95	9,82	9,96
	Ortalama	10,31	9,99	10,36	10,20	10,49	9,96	10,30	9,98	10,38	10,28	10,27	10,24
	Standart Sapma	0,16	0,32	0,22	0,36	0,37	0,31	0,21	0,29	0,21	0,18	0,29	0,21
GKO	En iyi	6,70	6,59	7,18	6,81	6,70	6,42	6,70	6,38	6,47	6,69	6,49	6,40
	En kötü	6,28	6,15	6,44	6,32	6,20	6,05	6,36	6,12	6,29	6,17	6,22	5,94
	Ortalama	6,38	6,34	6,65	6,54	6,42	6,25	6,53	6,27	6,36	6,35	6,37	6,29
	Standart Sapma	0,12	0,12	0,21	0,19	0,15	0,12	0,11	0,08	0,06	0,14	0,09	0,16

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek B.5. 0^o, 1^o ve 30^o ve farklı dizilim yönleri için DG, L-SHADE ve HPSGKO verim sonuçları

Yöntemler	Değer (sn)	Döndürme Açısı											
		0^o				1^o				30^o			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
DG	En iyi	11,10	11,72	11,57	11,73	12,12	11,25	11,49	11,70	15,99	11,61	14,63	14,22
	En kötü	10,47	10,04	10,41	10,66	10,86	10,63	11,25	11,10	12,57	10,64	13,02	13,51
	Ortalama	10,75	10,63	10,79	11,19	11,36	10,87	11,39	11,31	14,15	10,91	13,68	13,81
	Standart Sapma	0,22	0,46	0,33	0,33	0,35	0,21	0,10	0,19	1,39	0,28	0,58	0,23
L-SHADE	En iyi	20,12	21,19	20,22	20,24	20,80	20,79	20,49	21,03	20,79	21,49	20,87	20,86
	En kötü	19,45	19,57	14,67	19,15	19,18	19,43	19,26	19,40	19,02	16,69	19,43	19,21
	Ortalama	19,79	20,26	18,82	19,65	19,87	19,78	19,69	20,03	19,85	19,42	19,97	19,90
	Standart Sapma	0,28	0,54	2,18	0,42	0,53	0,42	0,41	0,61	0,56	1,21	0,45	0,65
HPSGKO	En iyi	14,80	14,58	15,25	15,86	17,54	15,17	15,30	13,15	14,84	14,95	15,18	15,48
	En kötü	14,07	14,04	14,47	15,09	14,54	14,55	12,50	12,42	14,05	14,33	14,31	14,32
	Ortalama	14,47	14,28	14,99	15,41	16,15	14,78	13,29	12,82	14,39	14,57	14,76	14,96
	Standart Sapma	0,21	0,17	0,22	0,24	1,14	0,21	1,01	0,23	0,22	0,20	0,23	0,33

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

Çizelge Ek B.6. 60º, 90º ve 180º ve farklı dizilim yönleri için DG, L-SHADE ve HPSGKO verim sonuçları

Yöntemler	Değer (sn)	Döndürme Açısı											
		60º				90º				180º			
		Yön*				Yön*				Yön*			
		→	←	↓	↑	→	←	↓	↑	→	←	↓	↑
DG	En iyi	12,98	11,12	14,27	14,49	11,80	11,19	12,81	11,24	11,43	11,07	11,51	11,36
	En kötü	12,10	10,47	13,26	13,92	10,81	10,58	10,98	10,86	10,62	10,35	10,78	10,88
	Ortalama	12,52	10,73	13,62	14,14	10,98	10,82	12,13	11,03	11,02	10,71	10,95	11,10
	Standart Sapma	0,24	0,22	0,29	0,25	0,30	0,19	0,60	0,13	0,25	0,26	0,20	0,13
L-SHADE	En iyi	23,65	20,71	20,99	21,54	21,05	21,95	21,14	21,51	20,97	23,97	20,46	21,14
	En kötü	18,95	15,76	19,44	20,08	19,74	19,42	19,27	19,35	20,00	23,10	19,50	19,54
	Ortalama	20,88	19,37	20,20	20,37	20,27	20,41	19,99	20,11	20,57	23,33	19,95	19,98
	Standart Sapma	1,53	1,35	0,53	0,43	0,42	0,79	0,57	0,63	0,34	0,24	0,30	0,47
HPSGKO	En iyi	17,73	14,83	15,21	13,05	14,77	14,41	15,66	16,40	16,95	16,48	15,58	15,29
	En kötü	14,23	14,10	12,75	12,66	13,85	13,81	14,54	14,74	14,58	13,33	14,42	14,71
	Ortalama	15,75	14,43	13,39	12,81	14,38	14,08	15,04	15,38	15,90	14,68	15,13	15,09
	Standart Sapma	1,22	0,25	0,91	0,12	0,25	0,19	0,37	0,50	0,86	0,85	0,40	0,16

*→: Soldan sağa, ←: Sağdan sola, ↓: Yukarıdan aşağıya, ↑: Aşağıdan yukarıya

ÖZGEÇMİŞ

Adı Soyadı : Fatih Ahmet ŞENEL

Doğum Yeri ve Yılı : Malatya, 1988

Medeni Hali : Evli

Yabancı Dili : İngilizce

E-posta : fatihsenel@sdu.edu.tr

Eğitim Durumu

Lise : Malatya Org. Eşref Bitlis Lisesi, 2005

Lisans : Pamukkale Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği, 2009

Yüksek Lisans : Süleyman Demirel Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Ana Bilim Dalı, 2013

Mesleki Deneyim

SDÜ Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü 2011-……..(halen) Araştırma Görevlisi

Yayınları

SCI, SSCI ve AHCI dışındaki indeks ve özler tarafından taranan dergilerde yayımlanan teknik not, editöre mektup, tartışma, vaka takdimi ve özet türünden yayınlar dışındaki makale

1. ŞENEL, F. A., CETİŞLİ, B., "Görüntü İşleme ve Beş Eksenli Robot Kol ile Üretim Bandında Nesne Denetimi", Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, Sayı 21-5, Sayfa:158-161, 2015.
2. ŞENEL, B., ŞENEL, F. A., KAHRİMAN, M., "Linear Behaviour Modelling of SBB5089Z Power AmplifierModule with Data Mining," International Journal of Scientific and Technological Research, ISSN 2422-8702 (Online), Vol 3, No.5, 2017.

3. SAPLIOĞLU, K., ŞENEL, F. A., TOPÇU, F., "Yapay Arı Kolonisi ile Hazne Kapasitesinin Optimizasyonu: Köprüçay Örneği", Harran Üniversitesi Mühendislik Dergisi, Sayı:2-2, Sayfa:101-108, 2017
4. ŞENEL, B., KAHRİMAN, M., ŞENEL, F. A., "Besleme ve RF Giriş Gücü Şartlarının Wimax Güç Yükselteci Performans Parametrelerine Olan Etkisinin İncelenmesi", Süleyman Demirel Üniversitesi Mühendislik Bilimleri ve Tasarım Dergisi, Cilt:5 Sayı:3, 2017
5. YÜKSEL, A. S., ŞENEL, F.A., ÇANKAYA, İ.A., Yazma Davranışlarının Mobil Cihaz Sensörleri Kullanılarak Sınıflandırılması, Dicle Üniversitesi Mühendislik Fakültesi Dergisi, 9(1), 2018
6. ŞENEL, F. A., YÜKSEL, A. S., GÖKÇE, F., YİĞİT, T., "Gri kurt optimizasyon algoritması ile iki boyutlu dizilim yazılımının geliştirilmesi." Balıkesir Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 20(2), 2018.

SCI, SSCI ve AHCI tarafından taranan dergilerde yayımlanan teknik not, editöre mektup, tartışma, vaka takdimi ve özet türünden yayınlar dışındaki makale

1. PENÇE, İ., ŞİŞECİ ÇEŞMELİ, M., ŞENEL, F. A., CETİŞLİ, B., A new unconstrained global optimization method based on clustering and parabolic approximation, Expert Systems with Applications, Volume 55, 15 August 2016, Pages 493-507
2. ŞENEL, F.A., GÖKÇE, F., YÜKSEL, A.S., YİĞİT, T., A Novel Hybrid PSO-GWO Algorithm for Optimization Problems, Engineering with Computers, 2019
3. YÜKSEL, A.S., ŞENEL, F.A., ÇANKAYA, İ.A., Classification of Soft Keyboard Typing Behaviors Using Mobile Device Sensors with Machine Learning, Arabian Journal of Science and Engineering, 2019

Ulusal toplantıda sunularak tam metin olarak yayımlanan bildiri

1. ŞENEL, F. A., CETİSLİ, B., GÜVENÇ, S. A., TAŞCI, İ., Görüntü İşleme ve Optimizasyon ile Kesim Kalıplarının Deri Üzerine Optimum Yerleştirilmesi, 23. Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SİU 2015), 2015, Malatya
2. GÜVENÇ, S. A., ŞENEL, F. A., CETİSLİ, B., Bilgisayarlı Görü ile İşlenmiş İç Fındıkların Sınıflandırılması, 23. Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SİU 2015), 2015, Malatya
3. ŞENEL, F. A., TOKAT, S., "Görüntü İşleme Teknikleri Kullanılarak Bir Ortamın İnsan Yoğunluğunun Hesaplanması", s. 613-617, ELECO '2012 Elektrik - Elektronik ve Bilgisayar Mühendisliği Sempozyumu, 29 Kasım - 01 Aralık 2012, Bursa

Ulusal toplantıda poster, sözlü sunum ve gösterim

1. SAYGIN, M., ÖZTÜRK, Ö., ŞENEL, F. A., "Aile Hekimleri için OSAS Ön Tarama Testi Yazılımı", TÜSAD 40. Ulusal Kongresi - SOLUNUM 2018, 13-16 Ekim 2018, Antalya

2. SAYGIN, M., ÖZTÜRK, Ö., ŞENEL, F. A., "Vücut Analizi ile OSAS Tanısına Yeni Bir Yaklaşım Yapay sinir Ağı", Türk Toraks Derneği 21. Yıllık Kongresi, 11-15 Nisan 2018, Antalya
3. SAYGIN, M., ÖZTÜRK, Ö., ŞENEL, F. A., "OSAS Ön Tarama Testi Epworth için Yazılım geliştirme", Türk Toraks Derneği 21. Yıllık Kongresi, 11-15 Nisan 2018, Antalya
4. ÖZTÜRK, Ö., SAYGIN, M., ŞENEL, F. A., "OSAS TANISINA YENİ BİR YAKLAŞIM; YAPAY SİNİR AĞI", TÜSAD 39. Ulusal Kongresi - SOLUNUM 2017, 14-17 Ekim 2017, İzmir
5. METLEK, S., CETİSLİ, B., ŞENEL, A. F., Görüntü İşleme ile Traverten Plakaların Sınıflandırılması, 23. Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SİU 2015), 2015, Malatya
6. ŞENEL, F. A., CETİŞLİ, B., "ED 7220C Model Robot Kolunun Gömülü Sistemle Kontrolü", İnönü Üniversitesi, Otomatik Kontrol Ulusal Toplantısı (TOK'2013), Bildiri Kitabı, Sayfa 1219-1222, Eylül 2013, Malatya.
7. A. K. ŞEKER, M. F. ÇAĞLAR, F. A. ŞENEL, V. K. UZ, "Lazer Mesafe Sensörü ile Yüzey Tarama Sistemi", Fırat Üniversitesi, Elektrik-Elektronik Bilgisayar Sempozyumu (FEEB 2011), Bildiriler Kitabı II, Sayfa 56-60, Ekim 2011, Elazığ.

Uluslararası toplantıda sunularak tam metin olarak yayımlanan bildiri

1. ŞENEL, B., KAHRİMAN, M., ŞENEL, F. A., "2.45GHz LNA Design with Arduino Biasing Control Unit", International Advanced Technology Symposium (IATS), Fırat Üniversitesi, Elazığ, Türkiye, 2017
2. ŞENEL, B., ÇAĞLAR, M. F., ŞENEL, F. A., "Behavioral Modeling of 2.4GHz RF Power Amplifier Using Data Mining", 2nd International Conference on Engineering and Natural Sciences (ICENS 2016), Books of Abstracts, pp. 779, Sarajevo, Bosnia-Herzegovina, 24-28th May, 2016

Uluslararası toplantıda sunularak özet metin olarak yayımlanan bildiri

1. ŞENEL, F. A., PENÇE, İ., CETİŞLİ, B., A new global optimization method based on clustering and parabolic approximation for the shoe nesting on leather, 13th EURO Special Interest Group on Cutting and Packing (ESICUP), 18-20 Mayıs 2016, Sayfa 26, Ibiza, İspanya

Ulusal kuruluşlarca desteklenen projede görev alma

1. "Robot Kollar Kullanılarak Üretim Bandı Üzerindeki İşlenmeyen Parçaların Ayrıştırılması", 3216-YL1-12 nolu Süleyman Demirel Üniversitesi Bilimsel Araştırma Projesi