İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

TİP İMAR YÖNETMELİĞİNE UYGUN VAZİYET PLANI ÜRETEN BİR YAPAY ZEKA DESTEK SİSTEMİ

DOKTORA TEZİ

Murat Çağlar BAYDOĞAN
(502032018)

Mimarlık Anabilim Dalı

Mimari Tasarım Programı

Tez Danışmanı: Prof. Dr. Sinan Mert ŞENER

OCAK 2013

İTÜ, Fen Bilimleri Enstitüsü'nün 502032018 numaralı Doktora Öğrencisi **Murat Çağlar BAYDOĞAN**, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı **"TİP İMAR YÖNETMELİĞİNE UYGUN VAZİYET PLANI ÜRETEN BİR YAPAY ZEKA DESTEK SİSTEMİ"** başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı : Prof. Dr. Sinan Mert ŞENER .....
İstanbul Teknik Üniversitesi

Jüri Üyeleri :Prof. Dr. Gülen ÇAĞDAŞ .....
İstanbul Teknik Üniversitesi

Prof. Dr. Arzu ERDEM .....
İstanbul Teknik Üniversitesi

Prof. Dr. Mehmet OCAKÇI .....
İstanbul Teknik Enstitüsü

Prof. Dr. Salih OFLUOĞLU .....
Mimar Sinan Güzel Sanatlar Üniversitesi

Teslim Tarihi : 15 Eylül 2011
Savunma Tarihi : 07 Ocak 2013

ÖNSÖZ

Bu tez çalışmasında, yapay zeka yardımıyla tip imar planlarına yardımcı bir program geliştirilmiştir. Günümüzde pek çok alanda kendini gösteren yapay zeka algoritmalarının optimizasyon ve mimari tasarım problemlerini çözme yeteneği çalışma kapsamında ele alınmıştır.

Tezimin bu aşamaya gelmesinde katkısı bulunan tez danışmanım Prof. Dr. Sinan Mert Şener'e, tez izleme komitelerimde değerli katkılar sağlayan Prof. Dr. Gülen Çağdaş ve Prof. Dr. Mehmet Ocakçı hocalarıma teşekkürü borç bilirim.

ABC algoritmasını dünyaya kazandıran, tezimde bana birçok noktada bilgisiyle, tecrübesiyle yardımcı olan Prof. Dr. Derviş Karaboğa hocama ve programlama konusunda destek veren ERÜ bilgisayar mühendisliği bölümü araştırma görevlileri Beyza Görkemli ve Selçuk Arslan'a, tezim konusunda yardımcı ve yapıcı olan çalışma arkadaşlarım, Yrd. Doç. Dr. Ceyhan Yücel, Yrd. Doç. Dr. Sema Serim, Yrd. Doç. Dr. Burcu Ceylan, Arş. Gör. Umut Doğan ve Yrd. Doç. Dr. Hale Kozlu'ya şükranlarımı sunarım.

Ayrıca, maddi, manevi her konuda, karşılıksız, desteklerini esirgemeyen biricik aileme teşekkür ederim.

Eylül 2011

Murat Çağlar BAYDOĞAN
(Yüksek Mimar)

İÇİNDEKİLER

Sayfa

ÖNSÖZ ..... **v**
İÇİNDEKİLER ..... **vii**
KISALTMALAR ..... **ix**
ÇİZELGE LİSTESİ ..... **xi**
ŞEKİL LİSTESİ ..... **xiii**
ÖZET ..... **xvii**
SUMMARY ..... **xix**
1. GİRİŞ **1**
1.1 Amaç 2
1.2 Konunun Ele Alınış Nedeni Ve Hedefler 3
1.3 Çalışmada İzlenen Yöntem 5
1.4 Çalışmanın Kapsamı 7
2. TASARIM SÜRECİ; KARŞILAŞILAN PROBLEMLER VE TİP İMAR YÖNETMELİĞİ **13**
2.1 Tasarım Problemi ve Tasarım Kriterleri 13
2.2 Tasarımcıyı Zorlayan ve Sınırlayan Etkenler 23
2.2.1 Yerleşim 23
2.2.2 Yönlenme 24
2.2.3 Kütle ölçüleri 25
2.2.4 Topografya 27
2.2.5 Yollar (yaya yolları, otoyollar) bina ve vaziyet planı üzerindeki girişler 28
2.2.6 İmar yönetmelikleri (tip imar yönetmeliği) 29
2.3 Tip İmar Yönetmeliği (TİY), Mimari Tasarım Problemi Ve TİY Arasındaki İlişki 31
2.4 Binaların Yerleşiminde Bilgisayar Programına Olan İhtiyaç 46
3. MİMARİ TASARIMDA OPTİMİZASYON PROBLEMLERİNE İLİŞKİN TEMEL KAVRAMLAR : YAPAY ZEKA, SÜRÜ ZEKASI, ABC ALGORİTMASI **51**
3.1. Yapay Zeka Ve Gelişimi 52
3.2 Optimizasyon, Optimizasyon Problemi Ve Çözümleri 55
3.3 Hesaplamalı Tasarım Yöntemleri 63
3.4. Sürü Zekası (Swarm Inteligence) 67
3.5 Yapay Arı Kolonisi (Artificial Bee Colony- ABC) Algoritması Ve Yapısı 70
3.6 Programlama Dili Olarak C# (.net) Seçilme Nedenleri 84
4. YAZILIMIN ÖZELLİKLERİ VE KULLANIMI (PROGRAMIN ÇALIŞMA SİSTEMİ) **87**
4.1 Yazılımın Amacı 97
4.2 Kurallar, Kısıtlamalar, İşlemler 97
4.3 Programın Arayüzü ve Kullanımı 102
4.4 Uygulamalar 119

4.5 Konvansiyonel Sistemlerle Üretilen Vaziyet Planları İle Karşılaştırma 124
5. SONUÇLAR VE ÖNERİLER..... 139
5.1 Avantajlar-Dezavantajlar ve Yazılımın Projeksiyonu..... 140
5.2 Öneriler..... 141
KAYNAKLAR..... 143
EKLER..... 151
ÖZGEÇMİŞ..... 175

KISALTMALAR

CAD : Computer Aided Design
CAAD : Computer Aided Architectural Design
TİY : Tip Imar Yönetmeliği
ABC : Artificial Bee Colony
BDT : Bilgisayar Destekli Tasarım
TOKİ : T.C. Başbakanlık Toplu Konut İdaresi Başkanlığı
KAKS : Kat alanları katsayısı
TAKS : Taban alanı katsayısı
E : Emsal
Hmax : Maksimum kütle yüksekliği

ÇİZELGE LİSTESİ

Sayfa

Çizelge 2.1: Tip imar yönetmeliği özet tablosu. 36
Çizelge 2.2: İmar planlarında parselle ilgili kuralların bulunduğu bilgilendirme kutusu 40
Çizelge 3.1: Optimizasyon yöntemlerinin sınıflandırılması 60
Çizelge 4.1: Yola bağlı hmax ve kat adetleri özet tablosu 95
Çizelge 4.2: İmar planlarında parselle ilgili kuralların bulunduğu bilgilendirme kutusu 96
Çizelge 4.3: Temel parametreler özet tablosu 97
Çizelge 4.4: Kare parsel 3 blok için vaziyet planı alternatifleri 119
Çizelge 4.5: Kare parsel 4 blok için vaziyet planı alternatifleri 121
Çizelge 4.6: Kare parsel 4 blok, 30 derece açı için vaziyet planı alternatifleri 122
Çizelge 4.7: Yamuk parsel 2-3-4 blok için vaziyet planı alternatifleri..... 123
Çizelge 4.8: Çorum Merkez için üretilen vaziyet planları 130
Çizelge 4.9: Mersin Silifke için üretilen vaziyet planları 132
Çizelge 4.10: İzmir Torbalı için üretilen vaziyet planları 133

ŞEKİL LİSTESİ

Sayfa

Şekil 1.1: Yoğunluk ve kütle büyüklüğü (geometri) (taks-kaks ve kütle biçimi) (Tekeli, 1985) 3
Şekil 1.2: Problemin tanımı 9
Şekil 1.3: Çalışmanın şehircilik ve mimarlık ekseninde yeri (Günay (2006) 'dan değiştirilip, düzenlenerek alınmıştır.) 10
Şekil 2.1: "Analiz-Sentez-Değerlendirme" modeli (Gero, 1996) 17
Şekil 2.2: "Fonksiyon-Davranış-Yapı" modeli (Gero, 1999) 18
Şekil 2.3: Revize edilmiş "Analiz-Sentez-Değerlendirme" modeli (Gero, 1999) 19
Şekil 2.4: Kütle-yönlenme 25
Şekil 2.5: Kişi başına düşen yaşam alanı ve kütle büyüklüğü 25
Şekil 2.6: Toki'nin ürettiği konut tiplerinden çıkarılan tipler ve boyutları 26
Şekil 2.7: Bodrum kat - yol ilişkisi 27
Şekil 2.8: Dikdörtgen parsel ve yol durumları 28
Şekil 2.9: Bina girişi ve yol durumu 28
Şekil 2.10: Bitişik nizam. 33
Şekil 2.11: Blok nizam 33
Şekil 2.12: Ayrık nizam 34
Şekil 2.13: Serbest nizam. 34
Şekil 2.14: Karma nizam. 34
Şekil 2.15: Nizamlar ve bahçeler (Tekeli, 1985) 35
Şekil 2.16: Vaziyet planında kütle ve parselle ilgili tanımlamalar 38
Şekil 2.17: Perspektif (Çizim: Ç. Baydoğan) 38
Şekil 2.18: Parselin tek kenarına komşu yol 40
Şekil 2.19: Parselin iki kenarına komşu yol 40
Şekil 2.20: Parselin iki kenarına komşu yol (köşe parsel) 41
Şekil 2.21: Parselin üç kenarına komşu yol 41
Şekil 2.22: Parselin dört tarafına komşu yol 41
Şekil 2.23: 4 katlı kütleler için farklı yerleşim şemaları 42
Şekil 2.24: 8 katlı kütleler için farklı yerleşim şemaları 43
Şekil 2.25: 12 katlı kütleler için farklı yerleşim şemaları 44
Şekil 2.26: 15 katlı kütleler için farklı yerleşim şemaları 45
Şekil 2.27: 75 konut/hektar yoğunluğundaki bir konut adası için farklı çözüm önerileri (Biddulph, 2007) 46
Şekil 2.28: Konut adası ve yoğunluk ilişkisi (Cheng ve ark., 2006) 48
Şekil 3.1: Gerçek hayat problemlerine sistematik çözüm yaklaşımı (Yaman, 2005) 57
Şekil 3.2: Optimizasyon sürecinin temel aşamaları (Yaman, 2005 uyarlanmıştır). .. 57
Şekil 3.3: Sürü yaşam biçimlerine örnekler 67
Şekil 3.4: Basit optimizasyon işlemi (Karaboğa ve Akay, 2010'dan düzenlenerek alınmıştır.) 71
Şekil 3.5: Sezgisel yöntemlerin diğer yöntemlerle performans karşılaştırması (Karaboğa ve Akay, 2010'dan düzenlenerek alınmıştır.) 71

Şekil 3.6: Evrimsel hesaplama modelleri ve tarihleri (Karaboğa ve Akay, 2010'dan düzenlenerek alınmıştır.) 72
Şekil 3.7: Bal Arılarının Yem Arama Davranışı 74
Şekil 3.8: ABC algoritmasında kaşif arıların gösterimi (Karaboğa ve Akay, 2010 dan düzenlenerek alınmıştır) 77
Şekil 3.9: ABC algoritmasında görevli arıların gösterimi (Karaboğa ve Akay (2010)' dan düzenlenerek alınmıştır) 77
Şekil 3.10: ABC algoritmasında görevli arıların nektar seçimi (Karaboğa, Akay, 2010' dan düzenlenerek alınmıştır) 78
Şekil 3.11: ABC algoritmasında yemek kaynağını terk etme işlemi (Karaboğa, Akay, 2010' dan düzenlenerek alınmıştır) 78
Şekil 3.12: ABC algoritmasında hafızadaki yemek kaynağı ve rastgele bulunan arasında karılaşırma(Karaboğa, Akay, 2010'dan düzenlenerek alınmıştır) .. 79
Şekil 3.13: Yapay arı kolonisi (ABC) (Tsai ve ark.,2009). 82
Şekil 3.14: Yapay arı kolonisi algoritması genel işleyişi (Karaboğa ve Akay, 2007) 83
Şekil 4.1: Olası çözüm kümeleri – arama/çözüm uzayı**.** 88
Şekil 4.2: Bir bloğa ait bilgilerin arı olarak kodlanmış gösterimi. 89
Şekil 4.3: Tip imar yönetmeliğine uygun çözümler üretebilen yazılım için çözüm kümesi (string) şematik gösterimi. 90
Şekil 4.4: Binalara ilişkin basit parametreler (Pearlmutter, 1998). 91
Şekil 4.5: Yazılım temel adımlar (akış şeması). 98
Şekil 4.6:. Program arayüzü 103
Şekil 4.7: Parsel koordinatlarını belirleme işlemi adımları. 104
Şekil 4.8: Parsel köşe noktaları girişi adım 1 105
Şekil 4.9: Parsel köşe noktaları girişi adım 2 105
Şekil 4.10: Parsel köşe noktaları girişi adım 3 105
Şekil 4.11: Parsel köşe noktaları girişi adım 4 105
Şekil 4.12: Parsel içine yerleştirilecek blok sayısı ve TAKS- KAKS- Emsal bilgilerinin girildiği kutular. 106
Şekil 4.13: Blok sayısı girişi 106
Şekil 4.14: Emsal parametresine göre giriş 106
Şekil 4.15: TAKS-KAKS parametrelerine göre giriş 106
Şekil 4.16: TAKS-KAKS değerleri değişikliği 107
Şekil 4.17: Parsel 107
Şekil 4.18: Toplam parsel, max. inşaat, max. taban alanları bilgilendirme kutuları 108
Şekil 4.19: Kenar sayısına bağlı yol-komşu parsel durumunun seçilme işlemi 108
Şekil 4.20: Kot alınacak kenar 109
Şekil 4.21: Bloklara ilişkin kısıtlamalar 1. adım 109
Şekil 4.22: Bloklara ilişkin kısıtlamalar 2. adım 110
Şekil 4.23: Bloklara ilişkin kısıtlamalar 3. adım 110
Şekil 4.24: Bloklara ilişkin kısıtlamalar 4. adım 110
Şekil 4.25: Bir blok için yapılan değişikliğin tüm bloklara uygulanması. 111
Şekil 4.26: Açı girilmesi 111
Şekil 4.27: Parsel için eğim yönü belirlenmesi işlemi. 111
Şekil 4.28: Eğim, kot farkı belirleme işlemi. 112
Şekil 4.29: Üretme - yerleştirme butonu 112
Şekil 4.30: Üretilmiş vaziyet planı alternatifi 1 113
Şekil 4.31: Üretilmiş vaziyet planı alternatifi 2 113
Şekil 4.32: Üretilmiş vaziyet planı alternatifi 3 114
Şekil 4.33: Farklı kaydet penceresi 114

Şekil 4.34: Kaydedilen imaj (vaziyet planı alternatifi)..... 115
Şekil 4.35: Alternatif özet bilgisi..... 115
Şekil 4.36: Raporlama butonu. 116
Şekil 4.37: Raporlama penceresi 116
Şekil 4.38: Violation (İhlal)..... 117
Şekil 4.39: Raporlama kısmında violation (ihlal)..... 117
Şekil 4.40: ABC kontrol paneli 118
Şekil 4.41: TOKİ, Manisa, Merkez (Akhisar), 3. Etap (Anonim). 125
Şekil 4.42: TOKİ, Aydın, Merkez, sadece konut (Anonim)..... 125
Şekil 4.43: TOKİ, Bitlis, Rahva (Anonim)..... 126
Şekil 4.44: TOKİ, Diyarbakır, Silvan (Anonim) 126
Şekil 4.45: TOKİ, Eskişehir, Tepebaşı, Aşağısöğütönü, vaziyet planı 1 (Anonim).127
Şekil 4.46: Çorum merkez 2 için yapılan künye örneği. 128
Şekil 4.47: Çorum merkez 1 için yapılan künye örneği. 128
Şekil 4.48: Çorum merkez için üretilen vaziyet planları örnek tablosu. 128
Şekil 4.49: ABC standart ayarları..... 129
Şekil 4.50: Çorum Merkez vaziyet planı detaylı künyesi..... 129
Şekil 4.51: Mersin Silifke vaziyet planı detaylı künyesi. 131
Şekil 4.52: Programda Mersin Silifke parsel..... 131
Şekil 4.53: İzmir Torbalı vaziyet planı detaylı künyesi..... 133
Şekil 4.54: Üç boyutlu vaziyet planı örneği 1. 135
Şekil 4.55: Üç boyutlu vaziyet planı örneği 2. 136
Şekil 4.56: Üç boyutlu vaziyet planı örneği 3. 136
Şekil 4.57: Üç boyutlu vaziyet planı örneği 4. 136
Şekil 4.58: Üç boyutlu vaziyet planı örneği 5. 137
Şekil A.1: TOKİ, Eskişehir, Tepebaşı, Aşağısöğütönü, vaziyet planı 2..... 153
Şekil A.2: TOKİ,Eskişehir, Çankaya, Vadişehir, 2. Etap..... 153
Şekil A.3: TOKİ,Eskişehir, Çankaya, Vadişehir, 3. Etap, vaziyet 1 154
Şekil A.4: TOKİ, İzmir, Torbalı..... 154
Şekil A.5: TOKİ, Konya, Ereğli, alt gelir grubu. 155
Şekil A.6: TOKİ, Manisa, Akhisar..... 155
Şekil C.1: Program çıktıları 1..... 161
Şekil C.2: Program çıktıları 2..... 161
Şekil C.3: Program çıktıları 3..... 162
Şekil C.4: Program çıktıları 4..... 162
Şekil C.5: Program çıktıları 5..... 162
Şekil C.6: Program çıktıları 6..... 163
Şekil C.7: Program çıktıları 7..... 163
Şekil C.8: Program çıktıları 8..... 164
Şekil C.9: Program çıktıları 9..... 164
Şekil E.1: 3 katlı kütleler için farklı yerleşim şemaları..... 165
Şekil E.2: 5 katlı kütleler için farklı yerleşim şemaları..... 166
Şekil E.3: 6 katlı kütleler için farklı yerleşim şemaları..... 167
Şekil E.4: 7 katlı kütleler için farklı yerleşim şemaları..... 168
Şekil E.5: 9 katlı kütleler için farklı yerleşim şemaları..... 169
Şekil E.6: 10 katlı kütleler için farklı yerleşim şemaları..... 170
Şekil E.7: 11 katlı kütleler için farklı yerleşim şemaları..... 171
Şekil E.8: 13 katlı kütleler için farklı yerleşim şemaları..... 172
Şekil E.9: 14 katlı kütleler için farklı yerleşim şemaları..... 172

TİP İMAR YÖNETMELİĞİNE UYGUN VAZİYET PLANI ÜRETEN BİR YAPAY ZEKA DESTEK SİSTEMİ

ÖZET

Günümüz koşullarında, Türkiye'de Tip İmar Yönetmeliğinin (TİY) kent planlamaya ve mimari tasarıma olan etkileri tartışmaya açık bir konudur. Tip imar yönetmeliğinin Türkiye'deki kentleşmeye olan etkileri, uygulama biçimi, işleyişi, geçerliliği bu tez çalışması kapsamında tartışılmayacaktır. Çalışma günümüzde var olan bu durumun değerlendirilmesini kolaylaştırmak amacındadır. TİY' in ve türevi olan imar planı yönetmeliklerinin tasarımcılara (mimarlar ve şehirciler) getirdiği yükü hafifletmek, hesaplardan kurtarmak, farklı çözüm olasılıklarını görmelerini sağlamak, zaman kaybını azaltmak ve en önemlisi; daha sonra oluşacak yapılaşma şartlarını ve alternatiflerini tasarımcılara göstermek çalışmanın hedefleri kapsamındadır.

Konutlar için yerleşim planları için tip imar yönetmeliği şartlarında, doğru yerleşimi bulacak, yoğunluk hesaplarını- (E (emsal), Hmax, TAKS ve KAKS hesaplarını) yapabilecek ve sonuçta çıkan kütle büyüklüğü ve kat adedini gösterebilecek bir program ortaya çıkarmak ve çalışırlığını örnek alan üzerinde sınamak tezin bilimsel yöntemini oluşturmaktadır. Ayrıca yukarda sayılan matematiksel problemlerin optimizasyonunu sağlayan program oluşturmak günümüzdeki "*imar*"la ilgili durumun pozitif yönde geliştirilmesi için gereklidir.

Çalışmanın içeriği tasarım, TİY, ve yapay zeka eksenindedir. Tasarım süreci, bu süreçte karşılaşılan güçlükler, TİY'nin getirdiği şartlar ve kısıtlamalar ikinci bölümde yer almaktadır.

Üçüncü bölümde; mimari tasarımda optimizasyon problemlemi çözümlerine ilişkin temel kavramlar; yapay zeka, sürü zekası, ABC algoritmasına ilişkin bilgiler verilmiştir.

Optimizasyon (en iyileme) yapan ABC algoritmasının oluşturulma süreci, oluşturulması için gerekli bilgiler ve programın mantığı dördüncü bölümde aktarılmıştır. Program içerisinde yer alan; kurallar, kısıtlamalar, işlemler, fonksiyonlar açıklanmıştır. Diğer bir deyişle; problemin ABC algoritması kullanılarak çözümüne ve oluşturulan programa ilişkin tüm bilgiler bu bölüm içerisinde yer almaktadır.

Oluşturulan programın çıktılarının konvansiyonel sistemlerle üretilmiş vaziyet planları ile karşılaştırılması, çıkan sonuçların değerlendirilmesi, genel sonuçlar ve geleceğe dönük çalışmaların neler olabileceği konusu sonuç ve değerlendirme bölümü içerisinde ele alınmıştır.

Tip imar yönetmeliği uygulaması üzerine oluşturulan programın üretkenliği (çeşitliliği, farklı alternatiflere ulaşma, matematiksel hesapların doğruluğu) ve hızı konvansiyonel yöntemlere kıyasla daha yüksek bulunmuştur.

Bilimsel anlamda, Türkiye'de günümüze kadar kentsel ölçekteki alanlara ilişkin sorunların veya olanakların keşfedilmemesi, Türkiye'de hala tartışılan TİY'nin yerleşim alanlarında oluşturduğu durumun korunması/düzeltilmesi/değiştirilmesi üzerine geliştilen fazla çalışma olmaması da TİY üzerine oluşturulan tezin geçerliliğini ve gerekliliğini dikkat çekmektedir.

Sonuç itibariyle bu tez çalışması, imar sistemi ve kuralları üzerine oluşturulan algoritmanın üretkenlikliğini test etmiştir. Gelecekte, kentsel alan ve imar üzerine daha geniş ölçekte ve teknolojilerin değişmesiyle daha farklı alternatiflerin oluşturulan programa eklenmesi kaçınılmazdır.

AN ARTIFICIAL SUPPORT SYSTEM TO GENERATE PROPER SITE PLANS FOR THE IMPLEMENTATION OF STANDARDIZED BUILDING AND ZONING REGULATION

SUMMARY

In Turkey, the effects of the Standardised Building and Zoning Regulation (SBZR) on the urban planning and architectural design have been very arguable recently. The questionable aspects of the SBZR (its effects on urbanisation, way of practise, mechanism, effectiveness etc.) are left out of the framework of the thesis; instead, providing an easier way to evaluation of current situation is the main goal of the thesis.

With the computers and technological advances designers can collect data easily, able to operate them without leeway and offer their designs. Nowadays computer programs are beyond being a helpful tool in designing process, have a major role on developing designing alternatives. In architectural design process computer programs have a supporting role in designing in several ways. If it is considered that the design process is a complex process it's obvious that computer technologies and programs contribute "designing" in terms of time.

The effects of the Standardized Building and Zoning Regulation (SBZR) on the urban planning and architectural design have been very arguable recently based on current building conditions in Turkey. The effects of the Standardized Building and Zoning Regulation to the urbanization, its mode of application, and validity are not within the scope of this study. The questionable aspects of the SBZR (its effects on urbanization, way of practice, mechanism, effectiveness etc.) are left out of the framework of the thesis; instead, providing an easier way to evaluation of current situation is the main goal of the study. The designed algorithm would provide a vision for practical results of Standardized Building and Zoning Regulation through manipulations on the parameters, contribute to the end product and accelerate the design-calculation processes. The method of the study is to design the software which finds the right place for placements of enclave plans which is according with SBZR, makes density calculations- (E(emsal, Hmax, FSI, GSI calculations) and presents the resulted block size and the number of storeys.

Design is a problem with many inputs. In architectural design, there are a lot of data areas which provides input for building shape formation.

Architectural design is; unimaginable without objective criteria such as environment criteria, user requests, zoning regulations, density, building type, building class (m^2-based), largeness of the mass and placement effects the decisions. Because of this it becomes necessary for a system which supports the site plan decisions. In taking placement decisions process (when starting the design), designer inputs, criteria and

working with a support system has transparent for data it will be possible for getting successful results.
In this study the main goal is to design the software on site-plan placements. Guide are for island scale, on the area to be used as a residential area, site plan of spatial organization, placement and depending zone regulations on the area, the algorithm has been developed to produce appropriate solutions. When making architectural designs the most difficult problem is site-plan decisions, with this software purposes to help designers with alternative site plan decisions.

With the software prepared by this study, the goals are;

- Directing designers with the mass largeness,

- Providing advice to suitable mass calculations, E (emsal) (density regulations), Hmax (height zoning), KAKS and TAKS which is a very time-consuming process.

- Showing the area calculation, elevation and similar calculations which must be written on placement plan.

- Helping the designer to see plan alternatives.

- Showing different site plan schemas in a short time.

- With the program making mass placement or planning in an order, obtaining two-dimensional suggestions in a short time

- Suggesting suitable placement order and storey heights to the designers

- Making changes easily in distances between units and neighbors

The thesis aims to ease planning and design process determined by the SBZR and other regulations, to get rid of quantitative works, to see alternative solutions, to reduce time looses, and the most importantly to present the building and constructing instructions and alternative solutions for new buildings to the designers.

To do this, designing and operating a computer programme that is expected

-to define the suitable location in the plot,

-to determine the dimensions and the number of storeys due to the building limitations of the SBZR (maximum size of the building on ground level, maximum size of construction, maximum height etc.) and

-finally to show the building proposals with the size and heights

is targeted for the situation plans of housing areas in urban settlements. A programme that necessitates the optimising of the mathematical problems mentioned above is needed to improve the current problematic situation of the building and construction area.

Design, SBZR and artificial inteligence are the main contents of the thesis. Design process, the difficulties that are faced in design process and the instructions and limitations of the SBZR take place in 2nd chapter. The 3rd chapter consist of the

knowledge on artificial intelligence, swarm intelligence, ABC Algorithm that are the basic concepts of optimising problem-solving in architectural design. In 4th Chapter, the generation process of artificial inteligence that is able to optimism, the necessary information and the logic of the programme are given together with rules, limitations, progress and functions of the programme. In other words, all information about solution of the problem and the proposed programme using the ABC algorithm is provided in this chapter.

The thesis is concluded with;

-the comparison of the outputs of the proposed a programme to the situation plans produced by conventional systems,

-evaluation of results,

-formulating the results and

-arguing about the further studies

in last chapter.

According to the comparison, the productivity (variety, reaching more alternatives, accuracy of the mathematical operations) of the algorithm based on the SBZR is found higher than the conventional methods.

In scientific manner, by means of the facts that the problems and potentials of the urban areas have not been analysed and discovered enough and there is not satisfactory researches on protecting/correcting/changing the situation of the housing areas mainly shaped by SBZR, the demand for and validity of the thesis is approved.

This thesis tests the productivity of the artificial inteligence mainly formed according to the building and construction system and its instructions. In the future, It is obvious that new options and alternatives will be improved for this programme by the innovations in technologies focusing on larger-scale urban and development problems.

1. GİRİŞ

Bilgisayarlar ve teknolojik gelişmeler sayesinde tasarımcılar verileri kolay toplayabilmekte, bunları zaman kaybı olmadan işleyebilmekte ve çıkan tasarımlarını iyi bir şekilde sunmaktadırlar. Günümüzde bilgisayar programları, tasarım sürecinde yardımcı bir araç olmanın ötesinde, tasarım alternatiflerinin geliştirilmesinde önemli bir rol üstlenmiştir. Mimari tasarım süreci içerisinde birçok noktada bilgisayar programları ve teknoloji ürünleri tasarıma destek veren bir rol üstlenmişlerdir. Tasarımın karmaşık bir süreç olduğu düşünülecek olursa bilgisayar teknolojileri ve programlarının "tasarlama" sürecine zaman açısından katkı sağladığı açıktır.

Geliştirilen yazılımların ve mimari paket programların (en çok kullanılan programlar: autodesk revit, archicad, allplan, sketchup, idemimari ve benzerleri gibi) tasarımcılara birçok noktada yardım edecek ve işlerini kolaylaştıracak modüller barındırdığı görülmektedir. Fakat genel anlamda paket programların vaziyet planı için çok fazla destek verdiğini söylenemez. Bu noktada çalışma, Türkiye özelinde eksikliği hissedilen vaziyet planı yerleşim kararlarına, hesaplara ve kütle büyüklüklerinin seçimine destek verecek yapay zekayı ortaya çıkarma ve bu yazılımın çalışma sistemini, bulguları, sonuçlarını ortaya koyma iddiasındadır.

Günümüz koşullarında, Türkiye'de Tip İmar Yönetmeliğinin (TİY) kent planlamaya ve mimari tasarıma olan etkileri tartışılmaktadır. Tip imar yönetmeliğinin Türkiye'deki kentleşmeye olan etkileri, uygulama biçimi, işleyişi, geçerliliği çalışma kapsamı içerisinde yer almamaktır. Tip imar yönetmeliğinin kurallarını esas alan çalışmanın amacı tip imar planlarına eleştiri getirmek veya savunmak değildir. Tip imar yönetmeliğinin hangi durumları ortaya çıkardığını tasarımcı ve plancıya göstermek üzere bu yazılım geliştirilmiştir. Oluşturulan program, TİY'nin uygulama üzerindeki sonuçlarını parametreler üzerinde yapılan manipülasyonlarla görme, sonuç ürüne katkı ve tasarım-hesap süreçlerine hız kazandıracaktır.

Çalışmanın içerisinde, konut tasarımları için yoğunluk tabanlı yerleşim-büyüklük plan kararlarını vermeye yardımcı olan sistem için kriterlerin seçimi ve nedenleri,

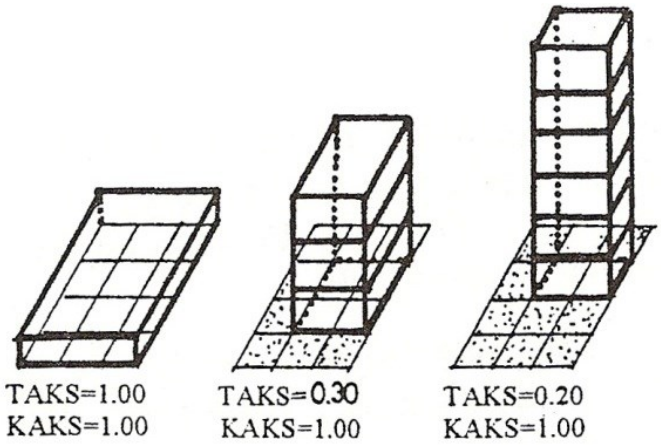
programın yazılım süreci, farklı yollarla üretilen yerleşim planları ile karşılaştırma ve sonuçlar ana başlıklar olacaktır.

1.1 Amaç

Konut yerleşim planları için tip imar planı ve yönetmeliklerinin şartlarını sağlayabilecek doğru yerleşimi bulacak, yoğunluk hesaplarını, (E(emsal), Hmax, TAKS ve KAKS hesaplarını) yapabilecek ve sonuçta çıkan kütle büyüklüğü ve kat adedini tasarımcıya gösterebilecek yardımcı programın oluşturulması çalışmanın ana amacını oluşturmaktadır. Diğer bir anlatımla; tip imar yönetmeliğinin, parseller üzerine yapılacak binaların özelliklerine (kat adedi, kütle büyüklüğü, çekme mesafeleri, vb. gibi) getirdikleri kısıtlamaları uygulayacak ve kütle yerleşimi önerecek bilgisayar programını geliştirmek çalışmanın ana hedefidir.

Tasarım, birçok girdisi olan bir problemdir. Mimari tasarlamada, bina biçimi oluşumuna girdi sağlayan birçok veri alanı bulunmaktadır. Bu alanlar genel başlıklar olarak, "işlevsel", "yapısal", "biçimsel", "ekonomik" ve "bağlamsal" olarak gruplanabilir. Günümüzde, belirtilen bu genel veri alanlarında bilginin çoğalması ve veri girişinin artması, mimari tasarlamada bina biçimi kararlarını karmaşık bir olgu haline getirmiştir. Mimari tasarımın çevre kriterlerinden, kullanıcı istek ve ihtiyaçlarından, imar yönetmelikleri, strüktür vb. gibi nesnel kriterlerden ayrı düşünülememesi, yoğunluk, bina tipi, bina sınıfı (m^2 bazlı) kavramların, kütle büyüklüğünü ve yerleşimi kararlarını etkileyen veriler içine girmesi, tasarım sürecinin en önemli ve temel aşamalarından biri olan vaziyet planı kararlarına destek verecek bir sistemi gerekli kılmaktadır. Yerleşim kararları alma sürecinde (tasarıma başlarken), tasarımcı girdileri, kriterleri ve verilerin çoğunu görebileceği destek sistem ile çalıştığında başarılı sonuçların ortaya çıkması mümkün olabilecektir.

Aşağıda örnek olarak verilen çizim (Şekil 1.1) basit yoğunluk hesabını (program kaks ve taks değişkenleri ile kütlenin büyüklüğünün nasıl değiştiğini) göstermektedir. Kütlenin zeminde kapladığı (parsel üzerinde kapladığı alan) ve kütle boyutları (en, boy ve yükseklik) arasındaki matematiksel ilişki, TİY (tip imar yönetmeliğinde) 'de yoğunluk bölgelemesi için kullanılan TAKS (taban alanı katsayısı), KAKS (kat alanı katsayısı) ve EMSAL ile sağlanmaktadır.



Şekil 1.1 : Yoğunluk ve kütle büyüklüğü ilişkisi (geometri-taks-kaks ve kütle biçimi) (Tekeli, 1985).

Özetle; çalışmada, tip imar yönetmeliği sonucu itibariyle, vaziyet planı yerleşimleri üzerine yazılım geliştirmek amaçlanmıştır. Kılavuz alan olarak ada ölçeğinde, konut alanı olarak kullanılacak alan üzerinde, vaziyet planında mekansal örgütlenme, yerleşim ve alana bağlı imar planı yönetmelikleri çerçevesinde, uygun çözümler üretmek için algoritma geliştirilmiştir. Mimari tasarımın temel belirleyicilerinden olan vaziyet planı kararlarının alternatif bir şekilde geliştirilmesini sağlayan yazılım, tasarımcıların başlangıç etütleri için ön fikir verecektir.

1.2 Konunun Ele Alınış Nedeni ve Hedefler

Günümüz koşullarında, Türkiye'de tip imar yönetmeliklerinin kent planlamaya, kentsel tasarıma getirdikleri ya da götürdükleri tartışmaya açık bir konudur. Çalışma, tip imar planı yönetmeliğinin tasarımcılara (mimarlar ve şehirciler) getirdiği yükü hafifletmek, hesaplardan kurtarmak, farklı alternatifleri görmelerini sağlamak üzerine yoğunlaşmıştır.

Tasarım sürecinin birçok evresinde bilgisayarların ve programların katkı sağladığı açıktır. Plan alternatifleri üretmekten 3 boyutlu animasyonlara, sunumlara kadar tasarımın birçok evresinde, bilgisayarlar ve programlar yer almaktadırlar. Fakat özellikle mimari çizim ve sunum için hazırlanmış paket programlar, vaziyet planı çözümü, hesapları, yönlenme alternatifleri için eksik kalmış ya da tam anlamıyla tasarımcılara ulaşamamışlardır. Genelde bu programlar, tasarımcılar ve mimarlar için vaziyet-yerleşim planlarına hiçbir katkı sağlayamamaktadırlar. Diğer yandan, yasaların, belediyelerin, imar mevzuatlarının, hesapların ve birçok etkenin, yerleşim

planları, kütle büyüklükleri, yönlenme, vb. gibi tasarımı etkilediğini söyleyebiliriz. Yoğunluk hesaplarının (E (emsal), TAKS, KAKS) yapılması, kütleleri belli bir düzene göre düzenleme ve 3 boyutlu olarak görme tasarımcıların zamanlarını almaktadır. Bu alanda daha önce yapılmış fazla çalışma olmaması da konunun özgünlüğünü ve gerekliliğini göstermektedir. Yazılan program ile bu alanda eksikliğin giderileceği düşünülmektedir.

Hedefler

Çalışma kapsamında hazırlanacak yazılım ile;

- Tasarımcıları gerçekçi kütle büyüklükleri üzerinden tasarıma yöneltmek,
- Çok zaman alan bir işlem olan, E (emsal) (yoğunluk düzenleme), Hmax (yükseklik bölgeleme), kat alanı katsayısı (KAKS) ve taban alanı katsayısına (TAKS) uygun kütle hesapları için önerilerde bulunmak,
- Daha önce belirlenmiş kütle formunu istenilen sayıda çoğaltma, çekme mesafelerine uygun bir biçimde yerleştirme ve yön için döndürme işlemlerini otomatik olarak yapmak,
- Yerleşim planın üzerinde alan hesabını, kotları ve benzeri hesapları göstermek,
- Tasarımcıyı verilerin ve mevcut uygulamadaki esnek olmayan kısıtlamaların sınırladığı ortamdan uzaklaştırmak (tasarımcının plan alternatiflerini görebilmesine yardımcı olmak),
- Farklı vaziyet planı şemalarını kısa süre içerisinde görebilmek,
- Belirli bir düzen veya kurala göre kütle yerleşimi veya planlama yapabilen bilgisayar programı ile iki boyutlu önerileri tasarımcılar için hızlı bir şekilde bilgisayar ortamında elde etmek,
- Tasarımcılara, girilen kriterlere uygun yerleşim düzeni ve kat yükseklikleri önermek,
- Birimlerin birbirleri ile, komşu parselle ve yolla mesafelerinin değişimlerini kolay yapabilmek,
- Kat alanı büyüklüğünü ve toplam alan büyüklüğünü olarak tasarımcılara göstermek,

- Vaziyet planında, kütlelerin kapladığı alan dışında kalan alanların (kentsel boşlukların) yüz ölçümünü vermek;
- Acil durumlar için (afet, savaş, vb.) yazılımın plan üretme kapasitesini araştırmak (Tip yapıların oluşturacağı yerleşimlerin ana planlama kararlarını verebilmek),
- Plan notlarına ve bu notların yönetmeliklerde, planlarda ve uygulamalarda yarattıkları değişikliklere-manüpilasyonlara da cevap verebilecek bir bilgisayar programı oluşturmak. Diğer bir deyişle gerektiğinde yönetmelik dışı çalışmalara da hizmet edecek (esnek) bir algoritma meydana getirmek.
- Gelecekte, farklı tip parametrelerin yazılımı nasıl etkileyeceğini araştırmak, hedeflenmiştir.

1.3 Çalışmada İzlenen Yöntem

Yazılımın yukarıda sayılan hedefleri gerçekleştirebilecek ve belirlenen kriterleri sağlayabilecek, problem olarak görülen uygulamaları yapabilecek bir sistem olarak çalışırlığının sağlanması, yöntemi belirleyecektir. Bir başka deyişle, “yazılımın, tasarım problemi çözme süreci içerisinde, çalışmalara destek verebilecek "yapay zeka" olduğunun denenerek ispatlanması” çalışmanın yöntemini ortaya koymaktadır. Yazılımın ürettiği çalışmaların konvansiyonel yöntemlerle üretilen çalışmalarla karşılaştırılması, üstün yönlerinin, avantajlarının, dezavantajlarının ortaya konulması çalışma yöntemi içerisinde yer almaktadır. Zaman-hız, doğruluk-güvenilirlik, erişilebilirlik, kolay kullanılırlılık, karşılaşılan problemlerin çözüm yöntemi, hesapların kontrolü çalışmanın sınanmasına yardımcı olacak kavramlar olarak seçilmiştir.

Çalışmanın içerisindeki aşamalar;

1. Literatür araştırması: Tasarım, ada ölçeğinde tasarım, tasarıma girdi sağlayan kriterler, yerleşim planı kararları, bu kararları etkileyen nedenler, vaziyet planlarını etkileyen nesnel veriler, (yoğunluk, çekme mesafesi vb. gibi), toplu konutlar, uygulanmış toplu konut örnekleri, uygun programlama dilleri, programlama dili seçim nedenleri, yurt içi ve dışında yapılan örneklere ve veri kaynaklarına ulaşmak, programlama dili için örnek kaynak kodlarına ulaşmak çalışmanın 1. aşamasında

yapılmıştır. Bu aşamada belirlenen ve yapılan diğer çalışmalar aşağıda maddelenmiştir.

- Benzer amaçlar için yazılmış programların (yazılı metinlerinin (amaç-yöntem-bulgu) incelenmesi,
- Temel bilgisayar bilgileri-modellerinin araştırılması ve çalışmaya uygun olan yaklaşımın irdelenmesi (matematik kökenli yaklaşımlar, dilbilim kökenli yaklaşımlar, biçimbilimi kökenli yaklaşımlar, biliş bilimi kökenli yaklaşımlar, mantık kökenli yaklaşımlar, bilgisayar bilimi kökenli yaklaşımlar; karar destek sistemleri, tasarımda yapay zeka yaklaşımları, uzman sistemler),
- Formun bilgisayara tanımında kullanılan modeller, tanım teknikleri ve uygulamaları,
- Mimari veya kentsel formun bilgisayarda kavramsal olarak modellenmesine ilişkin kuramsal bilgi ile kullanılan yöntem ve uygulanan stratejilere ilişkin bilgi edinmek,
- Bilgisayar programlamaya ilişkin temel bilgiler; algoritma kurma ve geliştirme; akış diyagramı yapılarını incelemek,
- Yapılandırılmış programların genel yapısını ve program oluşturma sürecini araştırmak,
- Bilgi tabanlı mimari tasarım sistemleriyle mekan planlaması yaklaşımları, modelleri ve kent ölçeğinde bu sistemlerin kullanılmasına ilişkin bilgiye ulaşmak,
- Bilimsel alan olan morfolojinin orijini, gelişimi, amacı, kavramı ve teknikleri, bu alandaki kaynakların kullanılması ve veri tabanının oluşturulması, vaziyet planı-bina biçimi, arazi-bina kullanımları arasındaki ilişkilerin açıklanması,
- Tip İmar Yönetmeliği'ni tüm yönleriyle incelemek, tasarımcı-plancılara getirdiği kıstlamaları ve avantajlarını detaylı bir biçimde araştırmak;

Çalışmanın kapsamına ve yöntemine uygun tüm bilgiler üzerinden çalışmanın problemi tanımlanmıştır.

2. Tanımlanan probleme ışık tutacak programın yazımı: Çalışmanın amacı olan uzman sistemi oluşmak, çalışırlığını sınamak bu aşamada yer alacaktır. Bilgisayar ortamında, program yazım aşamasında karşılaşılacak problemler, çözümler ve önerileri için yurt dışında yapılan çalışmalara, yazılan benzer programlara internet yolu ile-araştırma yaparak ulaşmak hedeflenmiştir. Programın yazımında izlenecek yöntem ve yollar aşağıda belirtilmiştir.

- Probleme dönük algoritma geliştirmek ve programlama uygulamaları denemek; problemin çözümünde kullanılabilecek benzetim modelleri geliştirmek, algoritmalarını kurmak,
- İki ve üç boyutlu grafik modeller; elemanlar üzerinde işlem yapma olasılıklarının denemesi (ölçekleme, döndürme, yansıtma ve öteleme işlemleri; şekiller üzerinde küme işlemleri),
- Formların bilgisayarda grafik temsiline ve tasarım amaçlı modellere ilişkin uygulamaları araştırmak.

Sonuç itibariyle çalışmanın bu aşamasında; programın mantığına ve çalışmanın kapsamına uyacak yazılımı geliştirmek amaçlanmıştır.

3. Yazılımın sınanması ve konvansiyonel sistemlerle karşılaştırılması : Bu aşamada değerlendirme, yorumlama, karşılaştırma ve ortaya konan veriler üzerinden öneri geliştirme işlemleri yapılacaktır. 1. ve 2. aşama sonucunda elde edilen bulgular ve ürünler doğrultusunda, çalışmanın sınanmasına yönelik önerilerin geliştirilecektir. Konvansiyonel sistemlerle üretilmiş ve üretilen çalışmalar ile karşılaştırma, yazılımın gelecekte geliştirilmesi için öneriler bu aşamada değerlendirilecektir.

1.4 Çalışmanın Kapsamı

Çalışma, bulunulan ortam gereği üzerinde durulması gereken birçok kavramı barındırmaktadır.

- Tasarım: Temel kavramlar, konut tasarımı, yer- çevre ilişkisi, toplu konutlar, konut büyüklükleri, yönlenme, yapılaşma, fonksiyon ve tasarım kriterleri, yerleşim-vaziyet planlarıı mevzuatlar, yoğunluk hesapları ve mekân ilişkisi, vb.

- Bilgisayar destekli tasarım: Mimari tasarımda yapay zeka, bilgi tabanlı tasarım sistemleri, bilgi tabanlı mimari tasarım sistemlerinin yapıları ve özellikleri, bilgisayarla mimari tasarımda bilginin temsili ve yorumlanması; us yürütme, mimari tasarımda yapay zeka kullanımı, sürü zekası ve uygulama alanları; çeşitli algoritmalarla üretilen mimari tasarımlar, evrimsel sistemlerle mekan planlaması yaklaşımları ve modelleri, olanakları, vb.
- Bilgisayar programlama: Programlama dilleri, OOP (Object Oriented programming), java, pascal, c# ve diğerleri, maxscript, programlama dili olarak kısıtlar, olanaklar-imkanlar, erişilebilirlikler, avantajlar, dezavantajlar, vb.değerlendirilip çalışma için uygun programlama dilleri seçilecektir.

Diğer bir yandan kent planlama alanında; tip imar yönetmeliği, uygulanması ve kent mekânını etkileyen diğer konular da çalışmanın kapsamına girmektedir. Tip imar yönetmeliğinin, Türkiye'de hiyerarşik kurallar bütününde hangi noktada bulunduğu aşağıda gösterilmiştir.

Anayasa—Kanun—Tüzük—Yönetmelik

Bu hiyerarşik durum imar planları, plan notları, belediye yönetmelikleri vb. gibi kural ve açıklamalarla kentsel mekan kontolü için kullanılmaktadır. Değerlendirilecek, gözden geçirilmesi gereken birçok konunun varlığı çalışmanın asıl problem olarak görülen noktaya eğilmesini zorunlu kılmıştır.

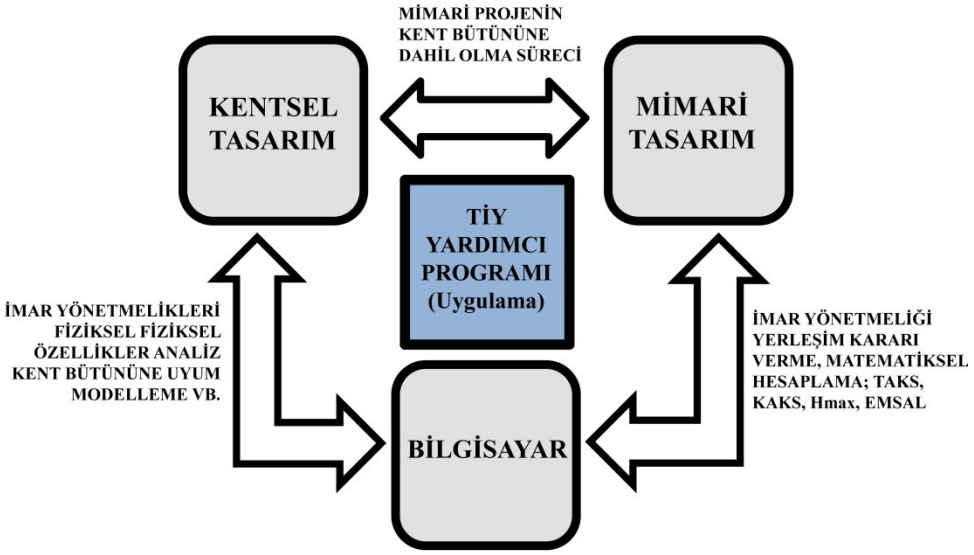
Bu bağlamda çalışma, ada ölçeğinde ve konut alanı olarak belirlenmiş bir alanın yerleşim planlarının 3194 nolu imar kanunu ve yönetmeliği çerçevesinde bilgisayar programı ile yapılabileceğini ve tasarımcıya yardımcı bir araç olarak çalışan bir sistem olduğunu ispatlamak amacındadır. Program (yoğunluk hesabı, yoldan ve komşu parselden uzaklık, çekme mesafeleri, vb. gibi imar mevzuatına konu olan ve tasarımı etkileyecek nesnel veriler üzerinden) vaziyet planı oluşturmak amacındadır. Ada ölçeğinde bir alanın vaziyet planı kararlarını almak, plan iç çözümlerine göre daha nesnel ve kesin kriterleri olan bir alan olarak düşünülebilir. Ada ölçeğinde bir alanın vaziyet planı kararları verilirken, uyulması zorunlu kuralların kesin ve nesnel olması tasarımcıyı sınırlamaktadır.

Yazılım, KAKS ve TAKS hesabını, KAKS ve TAKS hesabına göre kütle büyüklüklerini, yerleşim alternatiflerini, tasarımcıya katkı sağlayabileceği düşünülen yönlenme alternatiflerini sunabilecektir. Bilgisayar ortamında çizimlerin çekme

mesafeleri, yol büyüklükleri vs. sayısal verilere dayanmaktadır. Türkiye özelinde, bina yükseklikleri ve karakterleri gün ışığı ve yoğunluk hesabına dayanmaktadır. Yapı adalarının biçimleniş özellikleri (doluluk-boşluk, yapı yoğunluğu, yapı adasında konumlanma, arsa-yapı ilişkisi, 3. boyut özellikleri) program yazımında kullanılabilecek nesnel kriterlerden sayılabilir.

Yukarıda açıklandığı üzere, çalışma farklı disiplinler arasında yer almaktadır. Çalışma Türkiye'de 3194 sayılı imar yasası ile tip imar yönetmeliğinin mimarlara, kentsel tasarımcılara getirdiği kısıtlamaları parsel içerisinde uygulayacak, farklı çözüm alternatifleri sunacak yardımcı bilgisayar programının oluşturulmasını içermektedir.

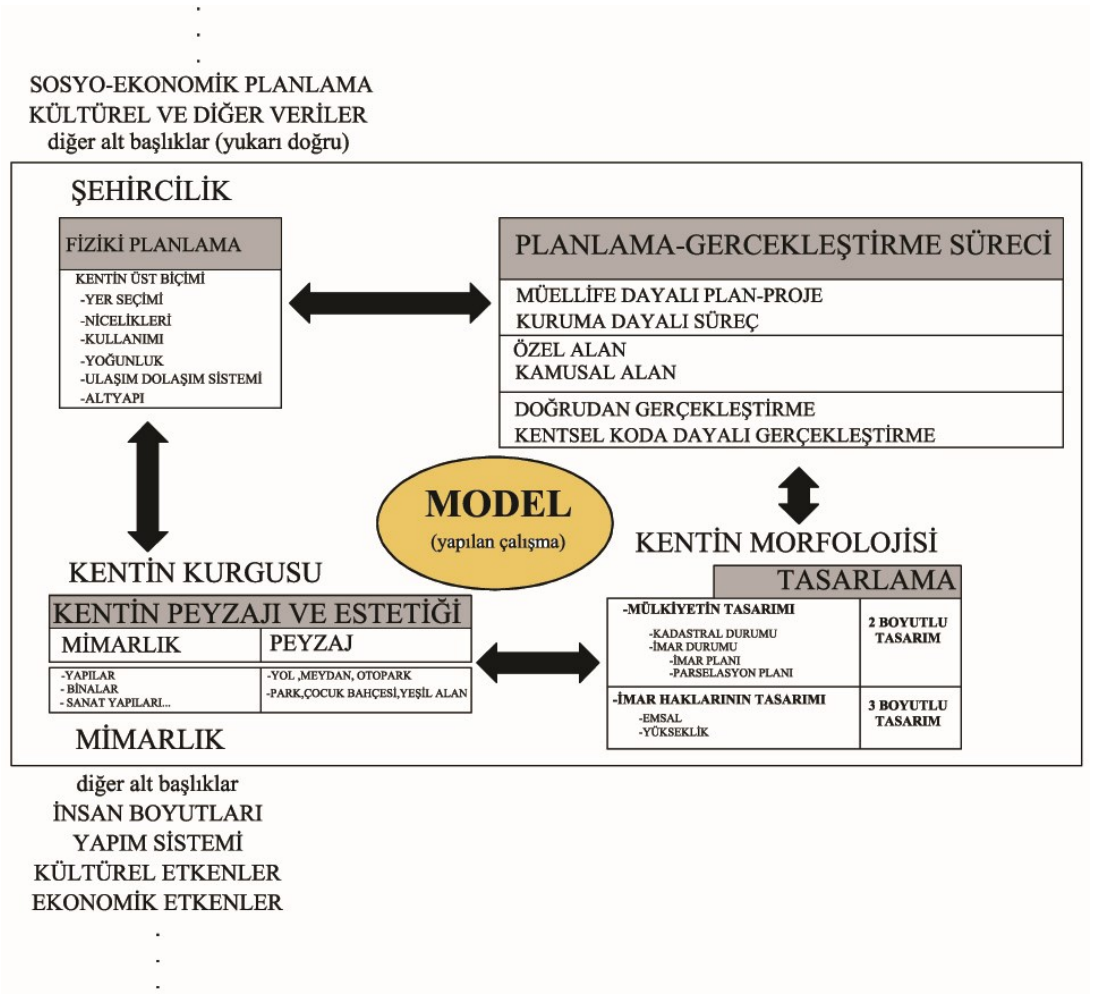
Disiplinler arası konumundan ötürü, yazılımın kapsamı tip imar yönetmeliği, kısıtlamaları ve kurallarıyla ile sınırlandırılmıştır. İleriye dönük geliştirme sürecinde programın yine bu çalışma alanında fakat biraz daha mimari tasarıma dönük bir noktada bulunacağı öngörülmektedir. Konunun yeraldığı teorik alt yapı aşağıdaki şekilde görülmektedir (Şekil 1.2).



Şekil 1.2 : Problemin tanımı.

Kent bütününe dahil olma sürecinde mimari kütlenin oluşma, yerleşme aşamalarını ve çalışmanın hangi alanlar çevresinde geliştiği aşağıdaki model çiziminde daha detaylı gösterilecektir. Şekil 1.3 de şehircilik ve mimarlık alanları arasında var olan döngü ve modelin yeri, fiziki planlama, planlama gerçekleştirme süreci, kentin morfolojisi, kentin kurgusu gibi genelde şehircilik bilgileri ekseninde yer almaktadır.

Bu sürecin içerisinde model- çalışma (TİY uygulama yardımcı programı) mimarlık alanında; bina tasarımı, ölçek-oran, büyüklük, parsel içerisinde yerleşim, vb. konularla, şehircilik ve kentsel tasarım alanında; yönetmeliklere uyum, yoğunluk hesapları değişimlerine uyum, vaziyet planı ölçeğinde dokuya, kent yapılı fiziki çevresine uyum gibi konulara yardımcı olmak için hazırlanmıştır. Şekil 1.3 de genelde şehircilik ve kentsel tasarım alanında genelde şekillenen işlem sırası ve şuan bulundukları durum aşağıdaki gibi kabul edilmiştir.



Şekil 1.3 : Çalışmanın şehircilik ve mimarlık ekseninde yeri (Günay (2006) 'dan değiştirilip, düzenlenerek alınmıştır.).

1. Fiziki planlama,
2. Planlama gerçekleştirme süreci,
3. Kentin morfolojisi,
4. Kentin kurgusu

Bu döngü içerisinde mimarlık; kentin kurgusu üst başlığında, kentin estetiği ve peyzajı içerisinde yer almaktadır. Bunun sebebi; şehirciliğin temel hedeflerinden biri olan "yaşanılabilir sağlıklı kentler ve çevreler üretmek ancak doğru bir planlama ile

olur" anlayışıdır. Genel ve kaba bir tabirle, mimarlık alanının uğraşısı bu plan ve gerçek hayatta kent içerisinde yer alan bir obje ve objelerdir. Çalışma şehircilik ve mimarlık arasındaki tasarım süreci boşluğunu doldurmak ve bu iki disiplinin arasında imar yönetmelikleri, imar planları vb. kurallarla tanımlı tasarım sürecinin doğru işlemesine yardımcı olmak amacıyla yapılmıştır.

2. TASARIM SÜRECİ; KARŞILAŞILAN PROBLEMLER VE TİP İMAR YÖNETMELİĞİ

Tasarım süreci içerisinde, kütle boyutları, parsel içerisine yerleşim, vaziyet planı kararları ve imar planlarına bağlı kısıtlamalar, kurallar mimarları zorlayabilmektedir. Tasarım sonuç itibariyle bu zorlukları çözme sürecidir. Bu süreç içerisinde, problemleri/zorlukları çözmeye yardımcı, alternatifleri görmeye destek sağlayan bir sistemin tasarımcılara katkı da bulunacağı açıktır.

Çalışmanın bu bölümünde, tasarımcıların tasarım problemi çözümünde izledikleri yöntemler, sürecin kendisi, bu süreçte karşılaşılan zorluklar açıklanacaktır. Tasarım ve tasarlama eylemi ile ilgili birçok çalışma vardır. Konunun, özellikle kentsel-mimari tasarım, bilgisayar ile tasarım konusuna odaklanması, üzerine hala tartışılan ve geniş bir konu olan tasarım süreci ile verilecek bilgilerin sınırlı, konunun kapsamına uygun bir şekilde verilmesini gerektirmiştir. Tasarım problemine bilgisayar ve algoritmik düşünce kavramları çerçevesinde bakılmaya çalışılmıştır.

Çalışmanın asıl odaklandığı TİY (tip imar yönetmeliği) ve TİY'nin tasarımcılara getirdiği kısıtlamalar, olanaklar ve olumsuzluklar detaylı olarak incelenecektir.

2.1 Tasarım Problemi ve Tasarım Kriterleri

Tasarım etimolojik olarak "Tasa-r-ı" "tasa", "tasar" ve "tasarı" ile ortak bir kökeni paylaşmaktadır. Tasa kaygı, dert olarak açıklanabilir. Tasarım kavramı, bir dertlenme olarak değerlendirilebilir. Tasar ise "bir iş, bir düşünce sırasını, düzeyini gösteren resim, yazı, plan" olarak belirtilmektedir. Tasarı ise "olması veya yapılması istenen bir şeyin zihinde aldığı biçim, proje"dir. "Bir araştırma sürecinin çeşitli dönemlerinde izlenecek yol ve işlemleri tasarlayan çerçeve"ye ise tasarım denilmektedir (Url-1). Tasarım; yapılacak eylem için izlenecek yolun, düşünce sırasının biçimlendirme kaygısıdır, denebilir. Tasarım kavramının, algoritma kavramına olan etimolojik yakınlığı ise dikkat çekicidir.

Larousse'un tanımına göre; "Tasarı, ulaşılmak istenen amaç, niyet, plan, proje, kabataslak çizgileriyle ortaya konan bir şeyi gerçekleştirme düşüncesidir" (Büyük Larousse Ansiklopedisi, 1986,). Dilimizde tasarım ve tasarlama kavramlarına benzer anlamlarda yabancı kökenli "Dizayn" kelimesi de kullanılmaktadır. Dizayn, latince "designaro", "designum" kökenli bir kelime olarak, "belirlemek", "işaretlemek" anlamındadır.

Araştırmacı ve teorisyenler, tasarım yeteneğinin, sistematiğini çözme çabasındadırlar. Tasarım üzerinde yapılan yöntem çalışmalarının tasarımı açıklayabilmesi için çeşitli araştırmalar yapılmıştır. "Ancak, giderek tasarıma bilim açısından yaklaşma gerekliliği duyulmaya başlanmıştır. Tasarımın büyük ölçüde insan beyninde oluşan bir süreçle ortaya çıkması nedeniyle, insanın psikolojik ve toplumsal açıdan dünyaya bakışı, çevresini etkileyişi ve etkilenişi tasarıma yansır. Bu nedenle, insanın tasarlama tasarım yapma esnasındaki davranışları tasarlama metotlarını tanımlamada ve ortaya koymada önem kazanmaktadır. Bu da ancak, tasarımcı üzerinde yapılacak bilimsel çalışmalarla sağlanabilmektedir (Bayazıd, 2004) ve sonuçları da tasarımcıdan tasarımcıya değişmektedir.

Terzidis (2006), tasarım teriminin genellikle planlama terimiyle karıştırıldığı söylemektedir. Planlama, önceden belirlenen bir amacı yerine getirebilmek için yapılan şema, program ya da metodu ortaya koymak için gösterilen çabadır. Tasarım ise kavramsal ve soyut bir eylemdir; bir fikri formüle etmek, görünür bir form veya eyleme dönüştürme isteğidir. Tasarım kavramsallaştırma, kurgulama, hayal etme, yorumlamadır. Planlama ise gerçekleştirme, organizasyon, uygulamadır. Tasarım belirsiz, tartışmaya açık bir oluşumdur. Tasarım, olagelir ya da başlangıç noktası genellikle kesin olmayan bir kavramın zihinde belirmesidir. Tasarım planlamanın çerçevesini çizen ilk aşamadır. Tasarım her zaman planlamadan önce gelir (Terzidis, 2006).

Diğer bir yandan, "tasarım" kelimesi birden fazla anlamda kullanılır. Tasarımcı tarafından hazırlanan çizimler, belgeler, sunumlar, şartnameler, yönerge ve hesapların hepsi "tasarım" olarak isimlendirilir. Tasarım aynı zamanda mimarın yaşadığı/deneyimlediği düşünsel süreç ve farklı eylemlerin de adıdır. Geleneksel anlamda, zanaatın ön planda olduğu endüstri çağı öncesi toplumlarda tasarım ve yapım kavramlarını birbirinden ayrılması mümkün değildir. Yapma eyleminden önce yapılacak veya inşa edilecek olanın planlanması düşünülemezdi. Buna karşılık

modern endüstri toplumlarında "yapmak" ve "tasarlamak" ayrı eylemler halini almıştır; genelde bir nesnenin yapımına tasarımı bitmeden başlanılması doğru bir eylem olarak değerlendirilmez. Günümüzde her tasarımcı (mimar, kentsel tasarımcı, endüstri ürünleri tasarımcısı veya mühendis) üretim sürecine bir katkı sağlar.

Tasarım ve yapım işleri arasındaki bu temel ayrım söz konusu olduğunda tasarım sürecinin en az bir amacı da açıklığa kavuşturulmuş olur: "Tasarım, yapılacak olanın açık ve net bir tanımıdır" (Bridges, 2007). Bu tanım, söz konusu objenin nasıl yapılacağına ilişkin tariflerin bütünüdür. Mimari tasarım açısından ele alınacak olursa vaziyet planı, hesaplamalar, uygulama çizimleri, üç boyutlu modeller, sistem detayları, nokta detayları bu tarifin birer parçasıdır.

Yukarıda aktarılanlara ek olarak, tasarım kavramının içerdiği soru veya probleme ilişkin, tasarımın eksik tanımlanmış bir problem yapısı içerdiği söylenebilir. Eksik tanımlanmış bir problem olarak (Mitchell,1985);

- Başlangıçta belirgin ve açıkça ortaya konmuş bir yöntem yoktur; fazlaca ve çok karışık bilgi yığının içerisinden bu yöntem çıkarılır. Pek çok durum ve kriter açıkça ortaya koymak yerine farz edilir.
- Uygulanabilir bir model verilmez; tasarımcı problemin çözümünü kolaylaştırmak için bazı değişkenleri ve operasyonları kendisi tanımlar. Tanımlanmış bir değişkenler / operasyonlar setinin de söz konusu probleme uygun olacağı hatta tasarımcının doğruproblemi çözmeye çalışıp çalışmadığı garanti edilemez.
- Ulaşılan yöntem katı değildir; süreç içerisinde keşfedilecek potansiyel çözümler ve çözüm kriterleri tamamen ve açıkça ortaya konmak yerine informel olarak karakterize edilebilir.
- Yöntem stabil değildir; şartlar ve kriterler tasarım süreci ilerledikçe edinilen sezgiler ve süreç boyunca kazanılan bilginin durumuna göre gelişebilir.
- Yöntemin kendisi süreç içinde çözülmesi gereken çelişkiler ve tutarsızlıklar yaratabilir; dahası bu çelişki ve tutarsızlıklar ancak tasarım sürecinin bitiminde açıklığa kavuşabilir.
- Tasarımcı tasarımını yapmak için gerekli olan bütün bilgiye aynı anda hakim olamaz; potansiyel çözümler veya çözüm kriterleri unutulabilir, değişik aşamalarda geliştirilen diğer çözümler ortaya çıkabilir (Mitchell,1985).

Rowe (1987)'a göre de söz konusu tasarım problemleri üç grupta ele alınabilir:

- İyi tanımlanmış problemler; amaçlar önceden açıkça verilmiştir, çözüm uygun araçların kullanılmasına kalmıştır. Örneğin bir arsa ve bunun içinde yer alacak bina mekanları ve bunlar arasındaki ilişkiler açıklıkla verilebilir ve tatminkar bir kombinasyon kolaylıkla oluşturulur.
- Yeterli tanımlanmamış problemler; problem çözmenin başında ne sonuçlar, ne de çözümün araçları bilinmektedir.
- Kesin formülasyonu olmayan, daima ek sorular sorulabilen, sürekli yeninden formüle edilebilen, buna bağlı olarak da, önerilen çözümlerin daima daha ileriye götürülebildiği problemler; ayrıca farklı problem formülasyonlarının, farklı çözümler verdiği ve son olarak önerilen çözümün doğru veya yanlış olmasının gerekmediği, dolayısıyla alternatif çözümlerin daima olabildiği durum (Rowe, 1987).

Bu durum için, mimarların tasarlayarak ortaya koyulabilen, net bir tanımı olmayan bir problemden, çok iyi tanımlanmış bir çözüme vardıkları söylenebilir. Bridges (2007)' a göre; tasarım süreci incelendiğinde genel anlamda tasarımcıların kullandığı stratejiler çok kaba olarak ikiye ayrılabilir. Ya önceden oluşturulmuş modellerin aşama aşama modifikasyonu tasarımı oluşturur ya da tasarımcı her aşamada ana eylemin kökeni olan "keşfetme" işini yeniden yapar. Bu noktada CAD in olanaklarını yeniden göz önüne getirmek faydalı olacaktır. Bilgisayar destekli tasarım doğal olarak yeni fikirler oluşturmak söz konusu olduğu zaman yetersizdir fakat önceden oluşturulmuş bazı yapıları (modelleri, kompozisyonları) belli amaçlar doğrultusunda ve sınırlar dahilinde değiştirmekte çok kullanışlı olabilir (Bridges, 2007).

Tasarımın doğasını anlamaya yönelik pek çok çalışma tasarımcıların tasarlarken her ne kadar benzer aşamalardan geçseler de kişisel metodlar kullanma eğiliminde olduğunu ortaya çıkarmıştır. Diğer bir deyişle her tasarımcı farklı teknikler kullanarak doğaçlama yapar, probleme göre çeşitli teknikler geliştirir. Bu bulgular göz önünde bulundurulduğunda hesaplamalı tasarım yöntemlerinin sundukları çarpıcıdır. Bu avantajları yüzeysel olarak sıralamak gerekirse; hazır formların erişilebilirliği, kısa zamanda çok fazla potansiyel çözüme ulaşabilme, üzerinde tasarım geliştirilen medyaya ait olan verilerin aynı zamanda sunum olarak kullanılabilme olanağı, oluşturulmuş kavramsal çizimler, modeller üzerinde tasarımcı

tarafından belirlenen sınırlar ve değişkenlerle oluşturulmuş yazılımları kullanarak manipülasyon yapma sayılabilir. Mimarların ve kentsel tasarımcıların, tasarımlarını yaparken kullanacakları araçları (yazılımlar) da geliştirme şansına sahip olmaları, tasarım stratejilerini de tasarlamalarına olanak sağlayabilmektedir.

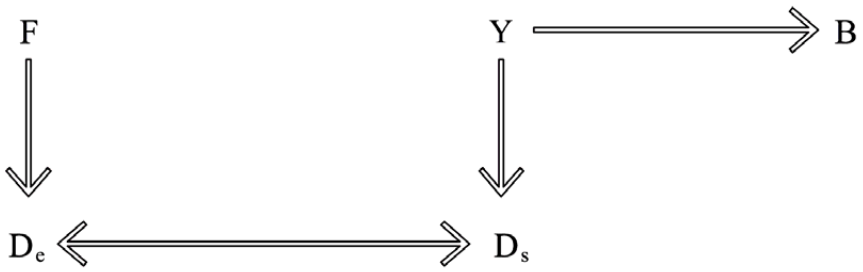
Tasarıma düşünsel bir süreç olarak bakılacak olursa; Tasarımcının tam tanımlanmamış sorundan iyi tanımlanmış çözüme varmak için ne gibi metotları kullandığı merak edilen bir araştırma konusudur. Tasarım kavramına ürüne yönelik pratik bir süreç olarak bakmak mümkün olduğu gibi tasarımcının tasarlarken içinden geçtiği düşünsel süreç açısından da bakılabilir. Dahası bilgisayarların tasarım sürecinde kullanımına yönelik bir model oluşturmak, tasarımın nasıl bir düşünsel süreç olduğu anlaşılmadan gerçekleştirilemez. Çeşitli araştırmacılar bu düşünsel süreç için birbirinden farklı ama özde tutarlı, karmaşık bir yapının daha kolay anlaşılmasını sağlayan çözümlemeler sunmuşlardır. Tasarım sürecinin düşünsel yanına değişik açılardan bakmak konunun daha anlaşılabilir kılacaktır. "Tasarım, amaçlı, sınırlandırılmış, karar almayı gerektiren, keşfetmeyi ve öğrenmeyi gerektiren bir eylem olarak kabul edilebilir. Karar alma, içinden tercih yapılması gerektiren bir takım değişkenler ve değerleri vurgular. Arama, karar alırken kullanılan temel eylemdir. Keşif ile kastedilen, alınan her karardan sonra yeni bir sorunlar bütününü çözmeye çalışmaktır. Öğrenme bilinenlerin yeniden yapılandırılmasıdır" (Gero, 1996). Gero'nun ortaya koyduğu tanım bizi klasik analiz-sentez-değerlendirme yorumuna yönlendirir (Şekil 2.1).



Şekil 2.1 : "Analiz-Sentez-Değerlendirme" modeli (Gero, 1996).

Analiz-Sentez-Değerlendirme modeli tasarımcıların tasarım yaparken nasıl düşündüklerini anlamaya yönelik en eski sınıflamalardan biridir. Bu modele göre tasarımcı önce karşılaştığı problemin analizini yapar sonra analizini dış veriler, tecrübe, alışkanlık gibi girdilerle sentezler ve bunu bir değerlendirme evresi izler. Süreç sürekli devinim halindedir. Buna karşın son yıllarda ortaya atılan bir takım

yeni öneriler de bulunmaktadır. Analiz terimi yerine formülasyon kavramı geçmiş, analiz daha çok değerlendirmeden önceki evre için kullanılmıştır. Bu sınıflamadan farklı olarak ortaya atılan farklı modellerden biri de "Fonksiyon-Davranış-Yapı" modelidir (Şekil 2.2).



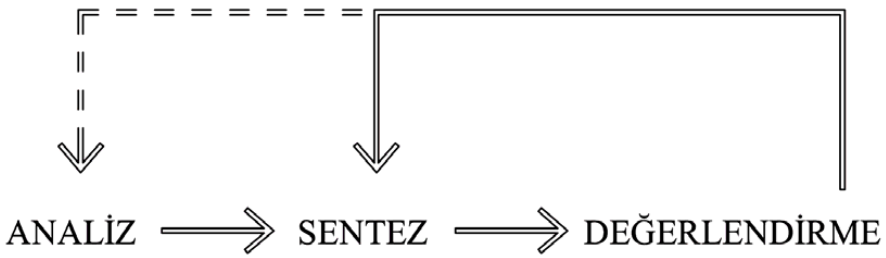
Şekil 2.2 : "Fonksiyon-Davranış-Yapı" modeli (Gero,1999).

F, fonksiyonu, Y yapıyı simgeler. Davranış iki kola ayrılmıştır. De beklenen davranışı, Ds yapıdan çıkarılan davranış veya aktüel davranışı, B tasarımın sonucu olan dokümantasyonu simgeler. → transformasyon, ↔ karşılaştırma için kullanılmıştır."Fonksiyon-Davranış-Yapı" modeli sekiz adet tasarım süreci şeklinde de görülebilir;

1) formülasyon: F → De
2) sentez: De → B (Ds üzerinden)
3) analiz: Y → Ds
4) değerlendirme: De ↔ Ds
5) belgeleme: Y → B
6) yeniden formülasyon - 1: Y → Y'
7) yeniden formülasyon - 2: Y → De
8) yeniden formülasyon - 2: Y → F De (F üzerinden)

1 ila 5 nolu süreçler önceki analiz sentez değerlendirme modeli ile bir bakıma örtüşmesine karşın 6 ila 8 nolu süreçler genel olarak aynı amaçlı yapılmış çözümlemelerde bilimsel açıklanabilirlikten uzak bir yapı sergiledikleri için çok net açıklanamamışlardır. Analiz-sentez-değerlendirme modeli tasarımcının temel eylemlerini tam anlamıyla sınıflamamaktadır. Yapılan çalışmalarda görülmüştür ki tasarımcılar değerlendirme sürecinde sadece tekrar analiz yapmakla kalmaz aynı zamanda sentez sürecini de tekrar ele alır. Bu yeni bakış açısı ile yeniden bir

değerlendirme yapıldığında analiz-sentez-değerlendirme modeli aşağıdaki gibi anlaşılabilir (Şekil 2.3).



Şekil 2.3 : Revize edilmiş "Analiz-Sentez-Değerlendirme" modeli (Gero,1999).

Tasarım sürecini inceleme kriterleri çeşitlendirilerek bu ve benzeri çözümlemeler artırılabilir. Daha önce de belirtildiği gibi tasarım, her ne kadar bilimsel çalışmalarda belli aşamalar ve süreçlere ayrılmaya çalışılarak anlaşılmaya çalışılsa da bir yandan da tasarımcıya özgü, daha çok bireyin alışkanlıkları ile biçimlenen sübjektif bir süreçtir. Dahası birey bu süreçte sürekli yaratıcı kimlikli eylemlerle biçimler, ilişkiler ve kararlar oluşturur. Bunun yanında sözlü ve grafik ifadeler mimari tasarım süreci için bir temel teşkil eder. "Olgun bir tasarımcı zihinsel ve dışsal sunumların arasındaki bağlantıyı iyi çözümler ve iki olgu arasında ustaca gidip gelir. Tasarımı geliştirme sürecinde tasarımcı söz konusu projenin kendisini nasıl yönlendirmesi gerektiği konusunda güçlü sezgilere sahiptir ve her aşamada düşündüklerini iki veya üç boyutlu eskizler halinde belgeleyerek süreçte yol alır. Bu eskizler sürecin "birlikte düşünülecek objeler"i haline gelir ve her bir sonraki adım için bir temel oluşturur. Bu kararlar veya kurallar böylece sürecin bir parçası olarak meydana gelmiş olur. Söz konusu meydana gelmiş kurallar hangi medya kullanılmış olursa olsun tasarım sürecinin bir sonucudur."(Yakeley, 2000).

Diğer bir yandan tasarımın sistematiği ve tasarımın doğası üzerine; J. Christopher Jones (1970), sistemli hale getirilmemiş tasarım süreçlerinin yarattığı tatminsizlikler ve yüksek maliyetli tasarım hataları sebebiyle bu süreçlerin çözüme kavuşturulması gerektiğini vurgulamıştır. Özellikle tasarım süreçlerinin dışa vurulması gerektiğini belirtmektedir. Bu yolla her çizgisi yüksek maliyetlerle hayata geçirilen mimar ve mühendislerin daha az hata yapacaklarını söylemektedir. Jones'un (Jones, 1970) tanımladığı iki tür tasarımcı tipi vardır. Bunlar "kara kutu" başka bir deyişle "sihirbaz tasarımcı" ve "cam kutu" yani "bilgisayar tasarımcı"dır. Kara kutu tasarımcılar, tasarım sürecinin hiçbir şekilde irdelenemeyeceğini tamamıyla içten

gelen, soyut bir işlem olduğunu savunmaktadırlar. Tasarımın psikolojik bir olgu olduğunu ve anlık sıçramalarla yapıldığını ya da ilhamla oluştuğunu aktarmaktadırlar. Tasarımcıyı hızlandıran etkenin ise tasarımcının deneyimi olduğunu savunmaktadırlar. Yaratıcılığın dışa vurulmasının mümkün olmadığını dışa vurulmaya çalışılan tasarımın da yaratıcılıktan uzak sıradan ve basit ürünler vereceğini iddia etmektedirler (Jones, 1970).

Cam kutu tasarımcılar ise bir bilgisayar gibi çalışmaktadırlar ve soyut, gizemli çözümler yerine rasyonel çözümleri kullanmaktadırlar. Bu tasarımcılar tasarım sürecinin tamamıyla açıklanabilecek bir süreç olduğunu belirtmektedirler. Sistematik tasarım süreçlerini kullanan tasarımcılar, bu süreçte, neyi ne için yaptıkları bildiklerini vurgularlar. Fakat sonuç olarak cam kutu tasarımcılar farklı tasarım süreçlerinde kara kutu sürecinden daha iyi sonuçlar elde etse de farklı durumlarda cam kutu tasarımcılar da kara kutu tasarımcı davranışlarını sergilemektedirler (Jones, 1970).

Bir tasarım kendi içinde bir yapıya ve bu yapı arkasında bir planlamaya sahip olmalıdır. Bütün sanatların temelinde bir tasarım olgusu bulunmaktadır. Tasarlama eylemi, oluşturulacak yapının organizasyonu ile ilgili her türlü faaliyeti içine almaktadır.

Tasarım süreci; aşağıdaki aşamalardan geçerek çözüme ulaşır;

1) Problemin tanımı

2) Bilgi toplama

3) Yaratıcılık ve buluş süreci

4) Çözüm bulma

5) Uygulama

Mimari tasarım, tasarım, tasarlama eyleminden farklı bir eylem değildir."Mimari tasarım" kavramı; A. Öke'ye göre; "Bina kavramına giren somut nesnelerin çözümlendiği bileşenlerden her birinin gelecekteki durumlarının ve bu durumlara ulaşılması için gereken eylemlerin kararlaştırılmasıdır" (Öke ve Ark., 1978) şeklinde açıklanabileceği gibi, kısaca; bina biçimine, yerine ve benzeri seçimlere ve tasarımın her aşamasında karşılaşacak bütün sorunların çözümüne karar verme süreci olarak tanımlanabilir.

Mimari tasarlama eylemine ilişkin farklı yaklaşımlar aşağıda verilmiştir.

-*Tasarlama olgusunu çözümleyici yaklaşımlar:* Mimari tasarlama kavramını farklı yönlerden açıklamayı amaçlayan bilimsel yaklaşımların varlığı bilinmektedir. Bu konuda yapılan çalışmalar incelendiğinde; bunların, tasarlama sürecini analiz eden yaklaşımlar, tarihi süreç içinde biçimsel üsluplar bağlamında yaklaşımlar, felsefi paradigmalar, teoriler ve akımlar bağlamında yaklaşımlar olarak gruplanabilir. Bu çalışmada diğer yaklaşımlar konu dışında bırakılarak; tasarlama süreci bileşenleri ile bilgisayar yazılımları ilişkisi ele alınmaktadır.

-*Tasarlama Süreci Bağlamında Yaklaşımlar:* Norbert Wiener'in (1950) geliştirdiği ve "Cybernetics" (yönetim bilimi) olarak isimlendirdiği, haberleşme ve kontrol genel teorisi günümüzün "Sistem Analizi" ve "Sistem Mühendisliği" bilgi disiplinlerine temel oluşturmuştur. Sistem kavramının daha sonraki yıllarda mimari tasarlama olgusunu da etkileyerek "Sistemli Tasarlama-Çözümleyici Tasarım Metotları" kavramlarının ortaya çıktığı görülmektedir (Öke ve Ark., 1978). Disiplinlerarası ilişkiler mimari tasarlama eylemini karmaşık bir olgu haline getirmektedir.

Mimari tasarlamayı bu veriler ile ele aldığımızda; tasarım sürecine ilişkin yaklaşımlar şu başlıklar halinde görülmektedir.

• Tasarlama dışı süreçlerin entegrasyonu bağlamında:

-Bina elde etme süreci yaklaşımları

• Tasarım içi süreçler bağlamında:

-Tasarımcı psikolojisi ve zihinsel süreç yaklaşımları,

-Tasarımda süreç stratejileri yaklaşımları,

-Tasarımda süreç strüktürleri yaklaşımları,

-Süreç organizasyonları yaklaşımları.

Tasarlama sürecinde, biçimlendirme eylemi ile düşünme biçimleri arasında yakın ilişki olduğu söylenebilir. Yaratıcı düşünce ile, tümdengelimci problem çözme stratejisi; algoritmik düşünce ile, tümevarımcı strateji arasında bağlar görülmektedir. Tasarlama sürecinde bu açılımlardan hangisinin öncül olarak seçileceğini, her tasarım sorunsalının kendine özgü veri seti belirlemektedir. Örneğin; sağlık veya endüstri yapıları ile; sembolik değer taşıyacak anıtsal-analojik yapı problemlerinde

tasarıma başlama noktaları farklı olacaktır. Ancak; tasarlama sürecinde bu tercihler tek başına kullanılmamakta, karşı etkileşimli olarak gidip gelmeler olmaktadır. Özellikle tasarım geliştirme aşamasında, bina alt eleman ve sistemlerinin (mekan eleman etütleri, malzeme, taşıyıcı sistem tercihleri gibi) bazılarına odaklanıldığında diğer ön aşamalara dönüşler olmaktadır (Gürer, 1976; Bayazıt, 1978).

Mimari tasarım, yukarıda aktarıldığı üzere; bilgi toplama, analiz, sentez ve değerlendirme aşamalarından sonra yeni ihtiyaçları karşılamak için çözümler önerilen bir süreçtir. Tasarım aşamasında, mimarın kullandığı araçlar çeşitlidir. Geleneksel mimari tasarım yönteminde karşılaşılan tasarım problemi için öncelikle bir araştırma yapılır. Gerekli bilgiler toplandıktan sonra, imar yönetmeliklerine göre büyüklük, oturum alanı, toplam inşaat alanı, kat sayısı, vb. tasarım için önemli noktalar belirlenir. Eskizler ve çalışma maketleri hazırlanır. Bu aşamadan sonra çizim bilgisayara geçirilir. Günümüzde çoğu tasarımcı bilgisayarları çizim kalitesini yükselten; 3 boyutlu görselleştirme sağlayan bir çizim aracı olarak kullanmaktadır. Fakat bilgisayar teknolojilerinin sırf çizim ve sunum için kullanılması bu karmaşık süreç için yeterli olmamaktadır. Tasarım sürecini ve çözüm yolarını bilgisayar teknolojileriyle geliştirmek için disiplinler arası gruplar oluşturup, projelere göre hesaplamalı tasarım yöntemlerini geliştirmek ve tasarım sürecini planlamak bilgisayarların tasarıma katılmasını sağlayacaktır.

Mimarlık alanında bilgisayarın rolü, sunuma yardımcı olmak şeklindeyken; süreç içersinde sunuma yardımcı olma işlevinden, tasarımı tetikleyici, yazılan algoritmalarla da tasarlayıcı olarak değişmiştir. Bilgisayarın rolünün değişmesi, mimarın rolünün değişmesine de neden olmuştur. “Hesaplamalı Tasarım” yöntemlerini kullanan mimarlar, öncelikle tasarım süreçlerini tasarlayarak işe başlamaktadır. Projeye göre kullanacakları “Hesaplamalı Tasarım” yöntemine karar vermekte, diğer disiplinlerden insanlarla beraber çalışarak, gerekli sistemi oluşturarak, işe başlamaktadırlar. Günümüzde mimarlar, bilgisayar programcıları, matematikçilerden oluşan tasarım ekipleri ile karşılaşılmaktadır. Bilgisayar teknolojileriyle tasarlanan projelerde sayısal parametreler, algoritmik olarak tanımlanmış ilişkiler üzerinden tasarım geliştirilir.

2.2 Tasarımcıyı Zorlayan ve Sınırlayan Etkenler

Tasarım sürecinde mimarların kent parçası içerisine yapacakları binalar için yaşadıkları süreç yukardaki bölümde aktarılmıştır. Sürecin içerisinde veri toplamadan sonra gelen aşamalar ve tasarımı-tasarımcıyı zorlayan etkenlerden bazıları aşağıda verilmiştir. Özellikle mimarların tasarım yaparken karşılaştıkları, çalışmanın kapsamına giren problemler:

- Yerleşim
- Yönlenme
- Kütle ölçüleri
- Topografya
- Yaya yolları, otoyollar
- İmar yönetmeliklerine uyum,

olarak belirlenmiş ve çalışma genelinde bu zorlukların üstesinden gelebilecek bir bilgisayar programını ortaya çıkarmak amaçlanmıştır.

2.2.1 Yerleşim

Yapının konumu ve yerleşimi ayrı bir öneme sahiptir. Yapı tasarımında yer seçimi şansı çok az olduğundan tasarımcının yapı için önceden belirlenmiş arazi parçasının var olan özelliklerini (topoğrafik düzeni, eğimi, yönü vs) veri olarak değerlendirerek güneş ışınımı açısından en uygun konumu belirlemesi gerekir. Bu belirlemenin kentsel yerleşme ölçeğinde mevcut yerleşme dokusuna uyması gerekir.

Mimari proje pratiğinde, projenin kendi iç felsefesinin yalınlığını ve tutarlılığını sağlamak kadar, parsel içerisinde doğru noktaya yerleşim de önemlidir. Örneğin; toplu konutlar için hazırlanan bir projede, konut planları ne kadar iyi çözülürse çözülsün, vaziyet planında, konutların bulundukları yerler, yoldan, kenarlardan uzaklıkları, birbirleri arasındaki ilişkileri, bütün blokların çevre yapılarla ilişkileri, kentsel ölçekteki parsele yerleşimleri, topografyaya uyumları uygun değilse, ortaya çözümsüz kente-çevreye uyum sağlayamayan ve sorunlu kentsel mekanlar çıkacaktır. Kent bütününde mimarların ve şehircilerin ilk başta çözmesi gereken durum; doğa ile bütünleşme, togografyaya uyumdur. Bu ise iki çalışma alanı için de doğru yer seçiminden geçmektedir.

2.2.2 Yönlenme

Vaziyet planında kütlelerin bir referansa göre yönlenmesi kastedilmiştir. Bu referanslar iklime ve güneşlenmeye göre kuzey-güney, sirkülasyona göre ana yol, manzaraya göre, vb. alınabilir.

Güneş enerjisinden mimaride tasarımda alınan yönlenme önlemleriyle yararlanmak olanaklıdır. Günümüzde güneş mimarisi, yeşil mimari ve enerji etkin mimari uygulamaları oldukça güncel olmakla birlikte; tarihsel perspektif, insanlığın binlerce yıldır güneşten gelen yaşam ve enerji akışının bilincinde olduğunu göstermektedir.

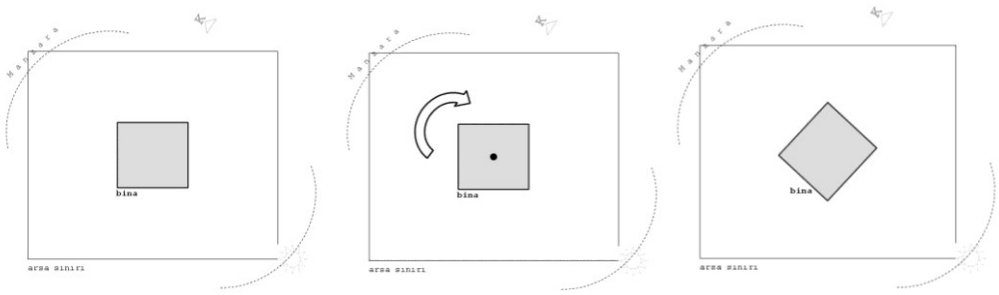
Güneş ışınlarının dünyaya yıl boyunca değişik açılarda ulaşması nedeniyle, mimari tasarımda ve kütle yerleşiminde yönlenme dikkatli kullanılırsa, sıcak ve soğuk dönemlerde ısıl açıdan binalarda en uygun şartları oluşturmak mümkün olabilmektedir. Kuzey yarım kürede, Türkiye genelinde, Orta ve Doğu Anadolu için; yaz güneşinin geliş açısının yüksek, kış güneşinin ise daha düşük olması, güneye bakan yüzeylerin kışın daha fazla güneş ışınımı almasını olanaklı kılmakta, dolayısı ile mimarlıkta güneye bakan cepheler önem kazanmaktadır. Verilen örnekler ve fiziksel şartlar gereği mimari tasarım probleminde yönlenmenin ne kadar önemli olduğu görülmektedir.

Yönlenmenin diğer bir önemi de kullanım kaynaklıdır. Yerleşilecek parsel içersinde ana yola ve manzara yönüne cephe vermesi genelde; kullanım değerini ve mülkiyetin değerini arttırmaktadır. Binaların, genelde plan çözümleri gereği önemli mekânlarının (salon, oturma odası, vb.) manzara yönüne bakma durumları, uzun cephelerinin ya da bir başka deyişle en fazla yüzeylerinin bu yöne bakması gerekliliğini getirebilir.

Tasarımcı döndürme işlemini yaparken, yukarıda sayılan kriterlerden (yol, manzara, kuzey, güney, vb.) birini veya birkaçını dikkate alarak mimari tasarımına ve dolayısıyla vaziyet planına yansıtır. Topografya durumu diğer bir deyişle, eğim döndürme – yönlenme için bu sebeplerden biri olarak sayılabilir. Eğim yönüne göre mühendislik hesapları (statik, mekanik, vb.) ve mali sebeplerden dolayı binayı (mimari kütleyi) döndürme işlemi yapılabilmektedir.

Sonuç olarak; herhangi bir sebepten ötürü, kütlelerin z eksenlerinden belli bir açıda döndürülmesine “yönlenme” denmektedir. Kartezyen koordinat sisteminde kütlenin

kenar uzantılarının x,y eksenlerine olan açısı, döndürülme açısını/yönlenme açısını vermektedir (Şekil 2.4).



Şekil 2.4 : Kütle-yönlenme.

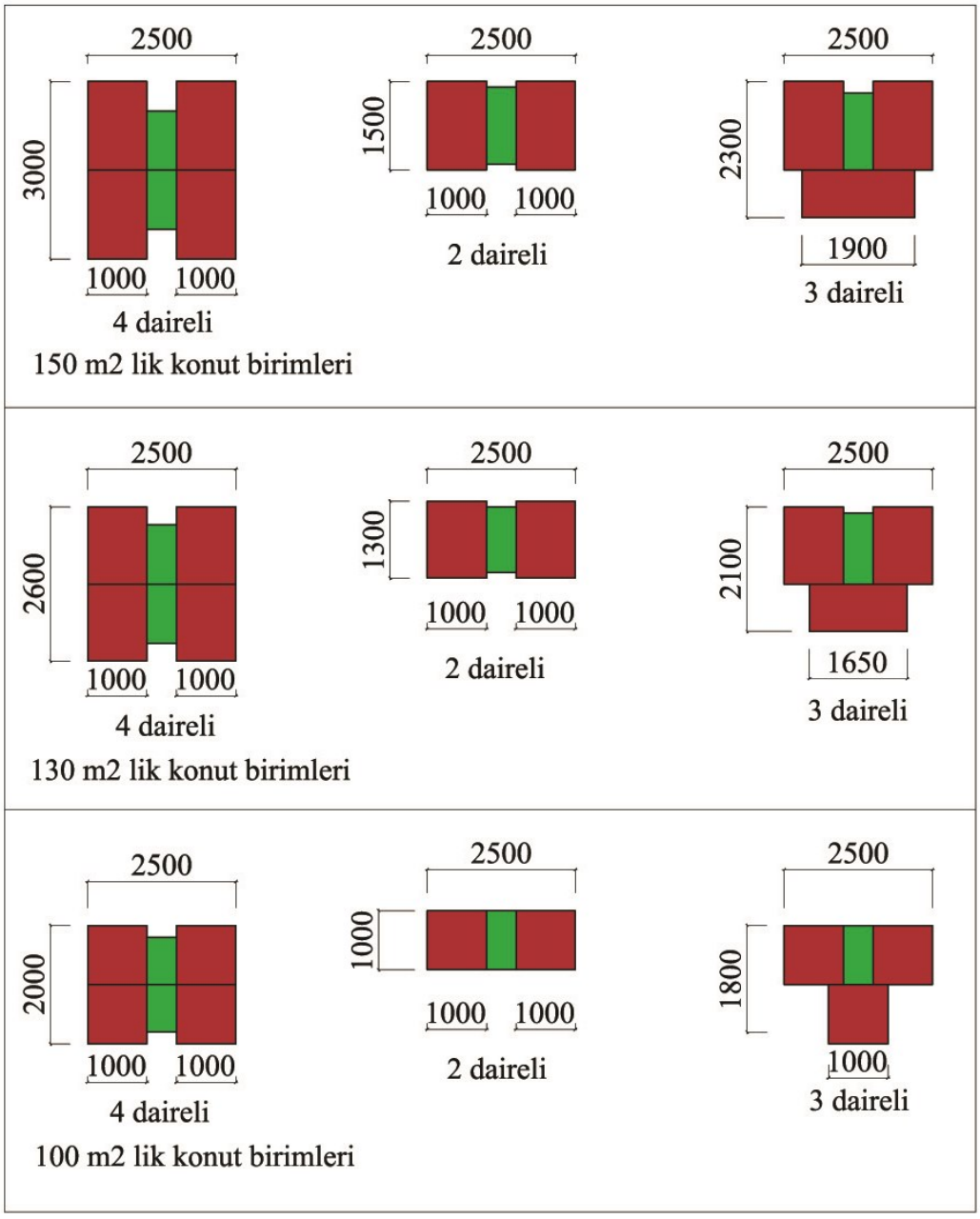
2.2.3 Kütle ölçüleri

Türkiye genelinde, son 30 yılın konut tipolojisi genelde yüksek katlı, kat yükseklikleri 3 metreden oluşan dikdörtgenler prizması şeklindedir. Konut büyüklüklerine bağlı olarak taban alanı yani kütlenin zeminde kapladığı alan değişmektedir. Konut büyüklükleri, toplam sayıları, her kattaki sayıları ve kütlelerin boyutları genelde benzemektedir. Konutta, bina girişinde, kat hollerinde ve diğer mekanlarda kullanılmayan, kayıp alanların bulunması ekonomik olmayan bir çözümü gösterir. TOKİ 'nin genelde ürettiği konut tipleri büyüklükleri kişi başına konut içinde düşen alanla ilişkilidir (Şekil 2.5).

	15 m2 (kişi başı)	20 m2 (kişi başı)	25 m2 (kişi başı)	30 m2 (kişi başı)	35 m2 (kişi başı)
3 kişi	45 m2 5x9 m.	60 m2 6x10 m.	75 m2 7,5x10 m.	90 m2 9x10 m.	105 m2 10x10,5 m.
4 kişi	60 m2 6x10 m.	80 m2 8x10 m.	100 m2 10x10 m.	120 m2 10x12 m.	140 m2 10x14 m.
5 kişi	75 m2 7,5x10 m.	100 m2 10x10 m.	125 m2 10x12,5 m.	150 m2 10x15 m.	175 m2 10x17,5 m.
6 kişi	90 m2 9x10 m.	120 m2 10x12 m.	150 m2 10x15 m.	180 m2 10x18 m.	210 m2 10x21 m.

Şekil 2.5 : Kişi başına düşen yaşam alanı ve kütle büyüklüğü (konutların boyutları ideal –mimari olarak çözülebilir boyutlarda, dikdörtgen olarak gösterilmiştir).

Başbakanlık Toplu Konut İdaresi Başkanlığı ’nın (Url-2) ve birçok konut yapan kooparatifin 2000‘li yıllarda yaptığı ve yapmakta olduğu bloklarının geneli, her katta 2, 3 ve 4 daireli yapılardır. Bunun nedeni Türkiye’de ekonomik, düşük maliyetli konuta olan talepdir. Şekil 2.6’ da TOKİ’nin genelde ürettiği konut tiplerinin kişi başı alana bağlı boyutları ve üretilen konutların boyutlarını gösteren özet tabloda yer almaktadır.



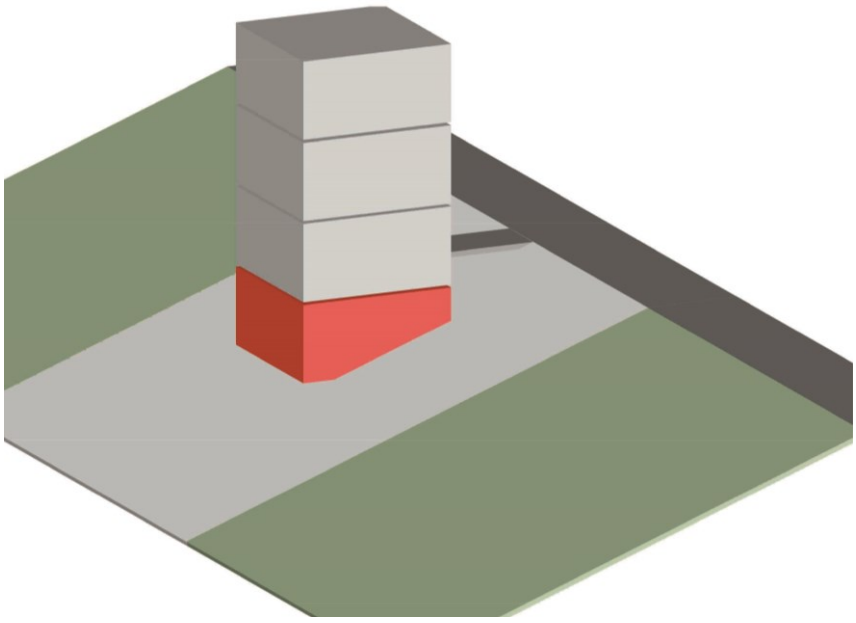
Şekil 2.6 : Toki’nin ürettiği konut tiplerinden çıkarılan tipler ve boyutları.

Konut kütle ölçülerini sadece; ekonomik daire çözümü ile ilişkilendirmek yanlış olur. Binaların coğrafi konumlarının, iklim ile ilişkili geometrilerilerinin de kütle

ölçülerini değiştiren etkenlerden sayılması mümkündür. Binalar soğuk hava şartlarında minimum ısı kaybı, sıcak hava şartlarında ise minimum ısı kazancı sağlayacak şekilde biçimlendirilmelidir. Bina boylarının uzunluğu iklim tipine göre değişir. Soğuk ve kuru iklimlerde kompakt yapı oluşturulması oldukça yararlıdır. Bu tür yapılar sert çevre etkilerine karşı minimum yüzey oluşturur. Kompakt yapıların boyları ile enleri arasındaki fark küçüktür.

2.2.4 Topografya

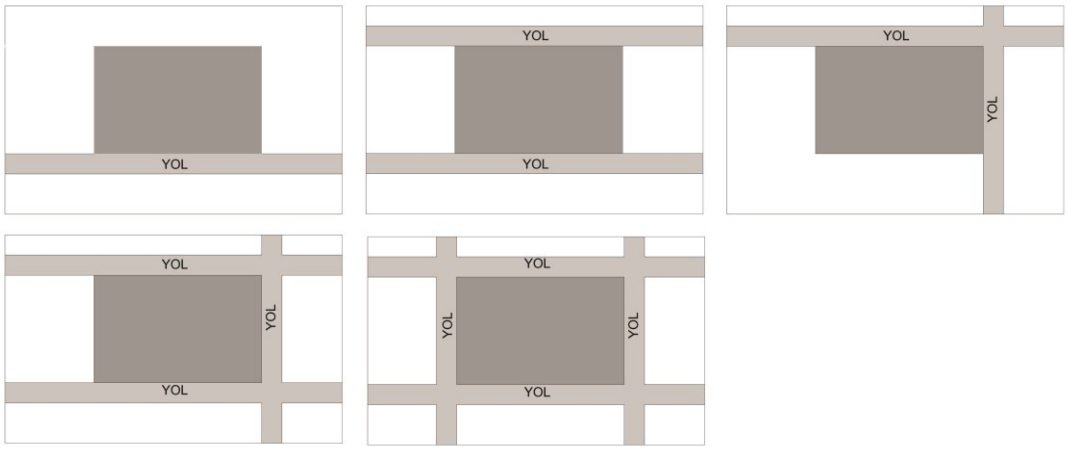
Topoğrafya tasarımcıları sınırlayan ve tasarımı etkileyen verilerden sayılabilir. Tasarımcılar (mimarlar, şehir plancıları) binanın hangi kota oturacağı, bodrum kat seviyesi vb. eğimle alakalı bilgileri kullanmak zorundadırlar. Bina giriş kotu, ve giriş kotu altında kalan katlar bodrum kat olarak ifade edilir. Diğer bir deyişle; bodrum kat, zemin katın altındaki katlardır. İmar yönetmeliklerine göre bina girişinin baz alındığı yol kotunun aşağısında kalan katların alanları KAKS (emsal) a dahil edilmemektedir. Bodrum katta kullanılan alanların tasarımı, hesabı, arsa alanı içerisinde (x,y) binanın oturduğu noktanın kotu (z koordinatı), blokların herbiri için oturdukları kotların ayarlanması (z), vaziyet planı oluşturma işlemleri içerisinde ve dolayısıyla tasarım sürecinde yer almaktadır. Tasarım problemi ve TİY kuralları, yukarıda belirtilen tüm noktaları içermektedir. Aşağıdaki şekilde (Şekil 2.7) yoldan kot alan ve tek yöne eğimli arsa üzerinde bulunan bloğun altında oluşan bodrum kat gösterilmektedir.



Şekil 2.7 : Bodrum kat - yol ilişkisi.

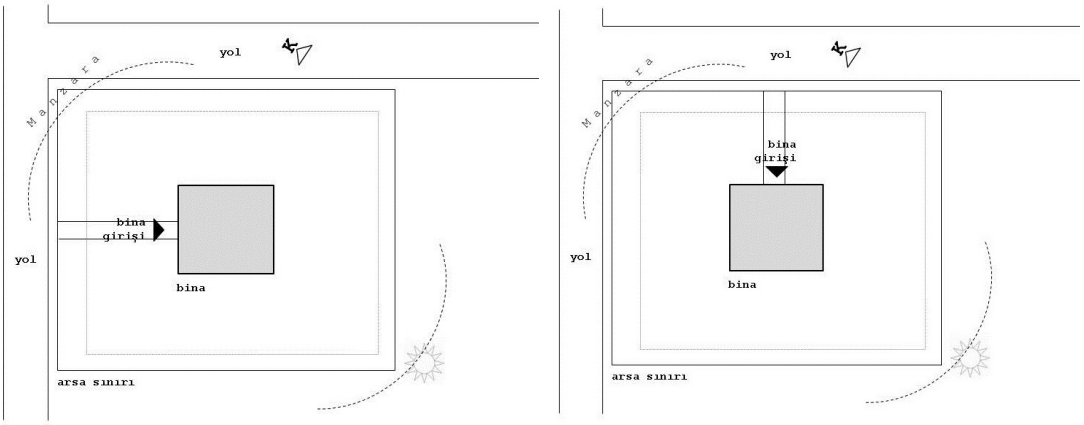
2.2.5 Yollar (yaya yolları, otoyollar) bina ve vaziyet planı üzerindeki girişler

Tasarım sürecinde; imar planı üzerindeki yollardan alınan servis yolları ve otopark çözümleri, binalara yollardan yaklaşım, vaziyet planı üzerinde binaların giriş çözümleri tasarımcıların alternatiflerini düşünürken zorlanabildikleri noktalardır. Parsele komşu yolların varlığı parsel içerisindeki kütle düzeni de değiştirebilir. Kütlelerin oto yollarına göre konumları, açıları mimari tasarım ve kentsel tasarım için çözülmesi gereken bir problemdir. Türkiye imar şartlarında dikdörtgen bir parselin yolla ilişkisi 5 temel şekilde oluşmaktadır. Bu durumlar; parselin bir kenarından geçen yol, parselin paralel iki kenarından geçen yol, parselin iki kenarından geçen yol (köşe parsel), parselin üç kenarından geçen yol ve parselin dört kenarından geçen yol (imar adası) şeklindedir (Şekil 2.8).



Şekil 2.8 : Dikdörtgen parsel ve yol durumları.

İmar adası kenarındaki yol/lar ve 3194 İmar Kanununa göre girişlerinin hangi yol üzerinden olacağına ilşikin bilgi mimari projeyi (tasarımı) değiştirebilir (Şekil 2.9).



Şekil 2.9 : Bina girişi ve yol durumu.

2.2.6 İmar yönetmelikleri (tip imar yönetmeliği)

Keleş (1996) imar planlarını, yönetmeliklerin ne amaçla kullanıldığını planlama üzerinden şu şekilde açıklamaktadır: "…alternatifler arasından yapılan seçim sonucu, alınan karar ile tamamlanan şehir planlama, bir karar verme sürecidir (Keleş, 1996).". İmar planları ise bu karar verme sürecinde bir araçtır. Açıklayıcı raporları ile bir bütündür ve uygulama süreci ile ilişkilerinin kurulması beklenir (Keleş, 1996). İmar planlamasından amaç, düzenli bir kentsel gelişimi ve toplum yararına uygun bir toprak kullanımını sağlamaktır. Çoğu kez, imar planının yapılmasıyla bu amaçlara ulaşıldığı ve her işin bitmiş olduğu düşüncesi de yaygındır. Oysa, imar planlarını uygulamaya olanak sağlayan koşulların gerçekleştirilmesi, kent planlamasının ana sorununu oluşturmaktadır.

Bu açıdan denebilir ki, imar planlarının uygulanması, bu sürecin en güç aşamasıdır (Yavuz, vd. 1978, ss. 344). Dolayısıyla, imar planları ile benimsenen arazi kullanış kararlarının (konut, ticaret, yol, park gibi işlevlerin) sınırlarının arazide (zeminde) işaretlenmesi ve böylelikle imar planının arazi ile ilişkisinin kurulması ve planın yaşama geçirilmesi başarılmalıdır (Gürler ,1983, ss. 139). Bu faaliyetlerin, bir imar planının yürürlüğe girmesinin hemen ardından yapılması zorunlu kılınırsa, kentsel gelişmeye yön verilebilir ve kentsel toprak (arsa) politikası kontrol altında yürütülebilir. Tersi durumda, imar planının araziye (zemine) uygulanması sağlanmadan, buna bağlı gerçekleştirilen diğer (ruhsat, teknik altyapı tesisleri, yol yapımı ve diğer pek çok denetim faaliyetleri gibi) işler başarılamaz. Türk hukuk sisteminde, imar planı, gerek imar programının ve gerekse arsa düzenlemesinin yasal dayanağı olarak kabul edilmektedir (Danıştay 6. Dairesi, 21.03.1988, E.1987/1369, K.1988/416 sayılı karar).

Çalışmanın genelinin ve bu bölüm alt başlığının Türkiye' de uygulanan imar programları ile çok yakın ilişki olması nedeniyle, Türkiye'de imar programının tarihsel evriminine çalışma içerisinde yer verilmesinde yarar görülmektedir.

"İmar programı, kent planlarını uygulayıcı makamların imar planlarını uygularken yararlandıkları en önemli yetkilerden biri olarak görülmektedir (Keleş 1996,ss. 130)." Cumhuriyetten sonra çıkarılan imar, belediye ve yapı ile ilgili tüm yasalarda da, imar programına, bu programı kentlere uygulama araçları olan imar planlarına ve

yönetmeliğe ilişkin ayrıntılı kurallara yer verilmiştir. Bunlar kronolojik olarak aşağıdaki gibidir:

1. 03.04.1930 tarihli ve 1580 sayılı Belediye Yasası
2. 10.06.1933 tarihli ve 2290 sayılı Belediye Yapı ve Yollar Yasası
3. 09.07.1956 tarihli ve 6785 sayılı İmar Yasası
4. 6785 sayılı İmar Yasası'nın imar programına ilişkin 30. maddesini değiştiren 11.07.1972 tarihli ve 1605 sayılı Yasa,
5. 03.05.1985 tarihli ve 3194 sayılı İmar Yasası

Çalışmada, kent bütünleri ve parçaları yerine bunları oluşturan bileşenleri, özellikle de ülkemizde parsel ölçeğine ve yönetmeliğin belirlediği yapı-imar tipolojilerine temellendirilmiş ve kentsel mekânı biçimlendiren yol-parsel-bina üçlüsünün oluşturduğu kentsel mekân birimleri arasındaki ilişkileri en uygun yolla çözümlemek, kentsel düzeninin ve mimari çözümün optimum dengesini yakalamak, istenen amaç olarak saptanmıştır. Dolayısıyla, aza indirgenmiş bu değişkenlerin birbirleri ile ilişkilerinden elde edilecek göstergeleri kent fiziki mekânının bütününe doğru giderek genellemek çalışmanın hedefleri için yanıltıcı olmaktadır.

Tasarlama eylemi özünde yaratıcılıktır. Tasarımın mekânın özüne ve biçimine ilişkin olarak elde edilmesine yönelik kurallarının olması zorunludur. Ancak ülkemizde bu kuralları kapsayan mevcut yasal düzenlemeler, tasarımı belirli kalıplar içerisinde ele almakta, dolayısı ile yaratıcılık eylemini kısıtlamakta, tanım ve yaptırım belirsizlikleri içermektedir. Bu belirsizlikler içerisinde gerçekleşen uygulamaların yaşanabilir çevreler oluşturulmasında yetersiz kaldıkları açıktır. Aynı belirsizlikler planlama ve mimariyi yönlendiren yönetmeliklerde de vardır. Planlama ve mimari arasında bir köprü niteliği taşıyan kentsel tasarım ise bu yönetmeliklerde, sadece çerçeve tanımlamalar ile yer almaktadır. Yönetmeliklerde kentsel tasarımın kavram, kapsam ve sürecine ilişkin herhangi bir değerlendirme bulunmamaktadır (Bayraktar, Kaplan, Tekel, 2006).

Yürürlükteki imar yönetmeliklerinin mimarlığı ve özellikle de planlamayı, sınırlayıcı ‘şablonlar’ içine sokan kuralları ile ‘anlam ve yaptırım belirsizlikleri’ içeren kuralları iç içedir (Karaman, 2003). Bunun yanı sıra kuralların ve kural uygulayıcılarının birbirleri ile çelişen yaklaşımları, amaçlanan süreçlerin ve hedeflerin dışında

uygulamalara ve kentsel tasarıma ilişkin eksik ve yetersiz değerlendirmeler yapılmasına neden olmaktadır.

Bu konuda, 1. Kentsel Tasarım ve Uygulamalar Sempozyumu Sonuç Bildirgesi'nin 18. maddesi, "Kentlerde imar yönetmelikleri yerine tasarımı yönlendiren kentsel tasarım rehberleri" geliştirilmesi gereğine dikkat çekmektedir (Çubuk, 1992). Aynı bildirgenin 26. maddesi planlama kademesi içinde sürecin bir parçası olarak kentsel tasarımın yer almasını ve bu düzenlemeye olanak sağlayacak bir Şehircilik Çerçeve Yasası'nın düzenlenmesini önermektedir. 9. Kentsel Tasarım ve Uygulamalar Sempozyumu Sonuç Bildirgesi'nin I.1.10. maddesi ise, yönlendirici yönetmeliklerin olması gerekliliğini; I.3.1. maddesi kentsel tasarımın yasal çerçevesinin çizilmesinin önemini; I.3.2.maddesi kentsel tasarımın planlamanın bir girdisi olması gerekliliğini vurgulamaktadır (Çubuk, 1997). Anlaşılacağı üzere imar yönetmelikleri ve kuralları tasarımcılar için çok kısıtlayıcı durumdadır. Mimari proje ölçeğinde özellikle konut tasarımında yaratıcılığı ve özgünlüğü etkilemektedir. Son dönemlerde bunun bilincinde olan ülkelerde mimari tasarım sürecini etkilemeden, yaratıcılığı engellemeden, kent dokusunu bozmadan farklı yöntemler kullanılarak kent içi yapıları denetleme yolları araştırılmaya başlanmıştır.

Günümüzde kentler imar yönetmelikleri ve plan kararları ile oluşmaya, eşit yükseklikli bloklar halinde gelişmeye devam etmektedir. Şehir planlama eğitimi ve süreci de, bu pratiğe dayanarak iki boyutlu yapı adaları ve alanları tasarlamakla sınırlı kaldı. Bu durum, mimari projelere daha çeşitli biçimlenme olanakları tanımaktan uzak, daha seçenekli ve esnek çözümler bulunmasına engel teşkil etmektedir (Acar,2006).

2.3 Mimari Tasarım Problemi ve TİY Arasındaki İlişki

Konu başlığı genel anlamda mimari tasarım problemi ile yönetmelik ilişkisini kapsamaktadır. Fakat çalışmanın bu bölümünde özellikle vaziyet planlarının oluşturulması, parsel içi kütle düzenleri, kent bütününe dahil olacak yapıların büyüklükleri, imar planları-vaziyet planları ilişkisi üzerinde durulacaktır.

Tip imar yönetmeliği üzerine literatürde birçok kaynak olmasına rağmen parsel büyüklüğü–kütle arasındaki ilişkiye dair, imar yönetmeliklerinin mimari tasarım üzerine getirdiği kısıtlamalar üzerine fazla kaynağa rastlanamamıştır. Bu ikili ilişki

üzerine çalışması olan Aysu (1992)'ya göre; Tip imar yönetmelikleri için; biçimsel olarak yasaların yönetmeliklerin belirlediği ve aynen uygulandıklarında da sağlıklı çevreler oluşturulacağına inandığımız imar kanunlarının getirdiklerini çözümleyerek bilmek, tanımak ve tüm sapmaları ile birlikte anlamak, açıklamak olanakların ve hem de kıt toprak kaynağının verimli bir biçimde kullanılmasını sağlamak, sağlıklı kentsel çevreler için gereklidir (Aysu, 1992).

Parsel ölçeğinde, kent mekânını düzenlemeye indirgenecek olan planlama çalışmalarında sağlıklı gelişmeleri denetlemek plancılar ve uygulamacılar için büyük bir önem taşımaktadır. Kent fiziki mekânının bileşenleri ve alt bileşenleri arasındaki ilişkileri yapıların, yapılar grubunun ya da kent mekânının bir parçasının içerdiği potansiyel ve performanslarını araştıran yöntemler, tip imar yönetmeliğinin geçerliliğini sorgulamaktadır. Burada potansiyel ve performans kavramları, yapısı belirlenen mekânsal dizgenin ya da alt dizgenin hangi koşullarda ya da girdiler çerçevesinde saptanan ya da beklenen amacı ne düzeyde gerçekleştirebileceğinin ölçüsü olarak tanımlanmaktadır (Aysu, 1992).

Çok sayıda değişkenin ya da girdinin etkilediği kentsel biçimlenmenin yapısını, geniş kapsamlı çerçeve içersinde çözümlemek ve belirlemek gerekmektedir (Aysu, 1992). Ancak insanın kavrama yapısına bağlı olarak amaçlanan olguyu daha iyi anlamak, açıklamak ve aktarmak için girdi oluşturan, bazı değişkenlerin sayısının oluşumdaki önemlerini unutmaksızın azaltmak ve sınırlamak gereklidir. İmar yönetmelikleri bir anlamda en küçük mekân biriminden (parsel-bina ölçeği) hareket ederek kent fiziki mekân bütünleri ve diğer yerleşme birimleri arasındaki bağlantıyı kurmayı da amaç edinmektedir.

Genel anlamda mimari bir tasarımın kent ile ilişkisini kurmaya yarayan, mimari nesnenin kent ölçeğinde anlaşılmasını sağlayan vaziyet planları tasarım için çok önemli bir noktadadır. Kenti şehircilik ölçeğinden alıp mimari ölçeğe getiren imar planları mimari tasarımın kente nasıl adapte olacağını kesin kurallar ile göstermektedir. Vaziyet planları imar planında gösterilen kurallara ve imar planı, yönetmelik hükümlerine uymak zorundadırlar. Özet olarak; her noktada imar planı kurallarına uygun, mimari nesne için optimum, tasarımcı için kolay üretilebilecek vaziyet planı üretmek. Bu sebeple programa girdi oluşturacak nesnel kurallar içeren imar planları, üretilme süreci ve kuralların detaylı şekilde incelenmesi çalışma için kaçınılmaz bir durumdur.

İmar planlarındaki bazı kavramlar için aşağıda kısa bilgiler verilmiştir;

Yapı düzenleri: 3194 sayılı imar kanunu ile belirlenmiş yapıların bir araya gelişlerine ilişkin düzenler aşağıdaki gibidir.

1. Bitişik nizam

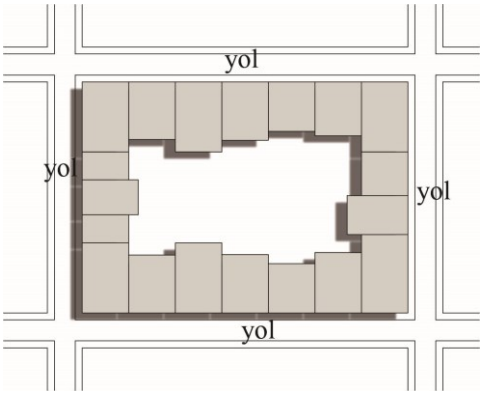
2. Blok nizam

3. Ayrık nizam

4. Serbest nizam

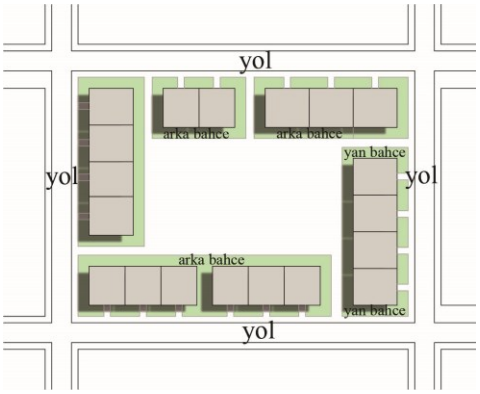
5. Karma nizam

1. Bitişik nizam: Hiçbir koşulda yan bahçesi bulunmayan düzendir.(Şekil 2.10) Dörtgen bir imar adasında yapılar birbirine bitişik olarak, ada çevresinde sıralanırlar. Eğer imar adası tek cepheli ise, yapılar bu cephe üzerinde sıralanırlar, ada bitimlerinde yan bahçe yoktur. Ön bahçe planda öngörülüyor ise vardır.



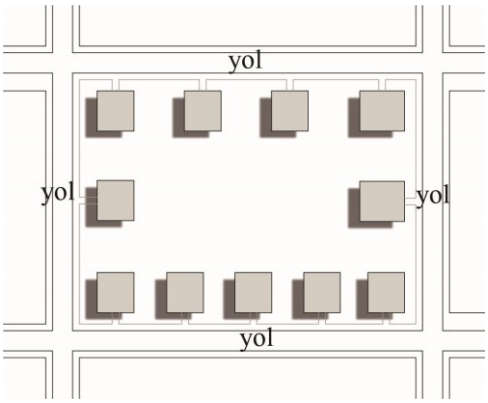
Şekil 2.10 : Bitişik nizam.

2. Blok nizam: Plan ya da yönetmelikte cephe uzunluğu, derinliği ve yüksekliği belirlenmiş tek yapı kütlesinin bir ya da birden fazla parsel üzerine oturduğu bahçeli yapı nizamıdır (Şekil 2.11). Yalnız blok sonlarında yan bahçe vardır.



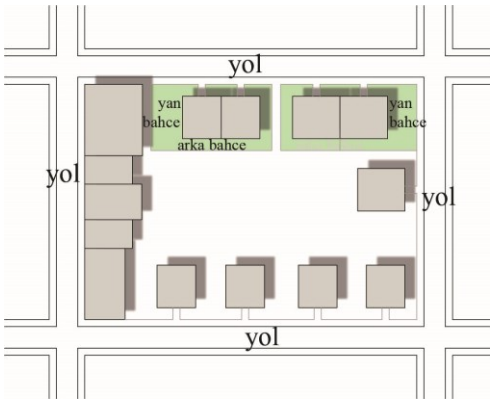
Şekil 2.11 : Blok nizam.

3. Ayrık nizam: Hiçbir yandan komşu parsele bitişik olmayan bahçeli yapı düzenidir (Şekil 2.12).



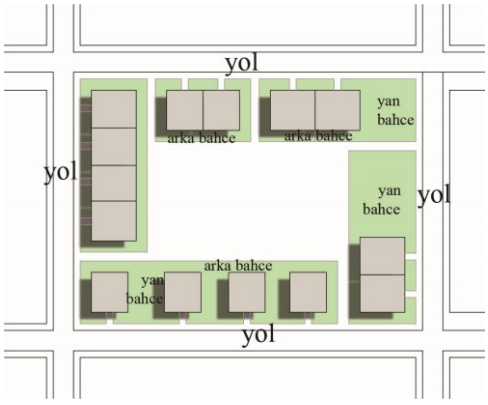
Şekil 2.12 : Ayrık nizam.

4. Serbest nizam: İstenilen düzenin parselde kullanılması (Şekil 2.13).



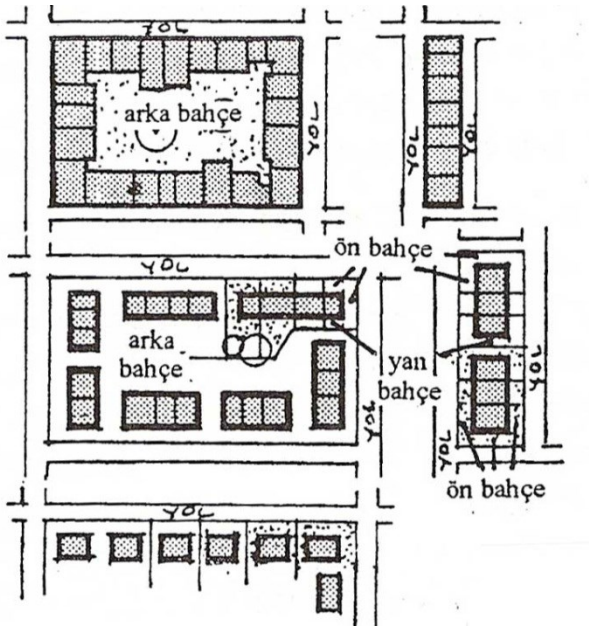
Şekil 2.13 : Serbest nizam.

5. Karma nizam: Üç ayrı nizamın aynı parsel içerisinde kullanılması (Şekil 2.14).



Şekil 2.14 : Karma nizam.

Yukarıda aktarılan yapı nizamlarıyla ilgili Tekeli'nin yaptığı çizim aşağıda (Şekil 2.15) verilmiştir.



Şekil 2.15 : Nizamlar ve bahçeler (Tekeli, 1985).

Diğer bir yandan, imar kuralları içinde yoğunluk bölgelemesine ve/veya nüfus yoğunluğu ayarlamasına yarayan araçlar başlıklar halinde açıklanacak olursa;

E (emsal): Bir yapının toplam inşaat alanının arsa alanına oranıdır. Genelde emsal E izin verilen en fazla yapı yüksekliği Hmax ile kullanılır.

KAKS hesabı ile aynıdır. Kat alanlarının toplamının arsanın alanına oranlanması ile belirlenir.

TAKS (taban alanı katsayısı): yapı taban alanının arsa alanına oranıdır.

KAKS (kat alanları kat sayısı): Bir yapının toplam inşaat alanının, kat alanları toplamının arsa alanına oranıdır. (örneğin KAKS 1 ise parsele yapılacak yapının toplam alanı arsa alanına eşittir.)

İmar planı düzenleme araçları; Plan yapımında yararlanılan düzenleme araçlarından başlıcaları bölgeleme, parsellemenin denetimi ve normlar/standartlardır. Konu kapsamında; araçlardan en çok yoğunluk bölgelemesi kullanılır. Program bu araç ve onun alt başlıklarında ki kısıtlamaları kullanacaktır.

Yoğunluk bölgelemesi; Yapı yoğunluğunu denetleyerek ışık, güneş ve hava sağlamak amacı ile kentsel arsa kullanımına getirilen sınırlamalardır.

Yapı yapılmasına izin verilecek en küçük parsel boyutları parsel alanını kullanım türüne ve yapılanma biçimine göre belirlenmektir. Bu konuda, plan üzerinde ya da

plan notu olarak bir açıklama yoksa "Tip İmar Yönetmeliği" nin (TİY) 17. maddesi geçerlidir. TİY'de belirtilen bahçe mesafesi ve yapı boyutlarına ilişkin değerler olabilecek en az değerlerdir. Daha büyük alınabilirler. Ayrıca, belediyeler TİY'ni yörenin doğal, kültürel, sosyal yapısıyla uyumlu biçimde, iyileştirici yönde değiştirebilirler.

Tip imar yönetmeliğinin 17., 18., 26., 27., ve 28. maddeleri özet tablosu Çizelge 2.1'de verilmektedir.

Çizelge 2.1 : Tip imar yönetmeliği özet tablosu.

Konut bölgelerinde "yapı nizamı"/ "düzeni"	Yapı boyutu (metre) (en az ve en çok)		Bahçe mesafesi (metre)		
	Cephe uzunluğu	Cephe derinliği	Ön	Yan	Arka
Bitişikdüzen (B)					
1-4 kat	En az 6 m.	10-40 m.	olmayabilir	yok	Olması gereken H/2 (en az 3 metre)
5-9 kat	En az 9 m.				
10+ kat	En az 12 m.				
Blokdüzen (BL)					
1-4 kat 5-9 kat 10+ kat	En çok 50 metre	10-40 m.	En az 5 metre	Blok başında en az 3 m. + her bir kat için 0,5 m.	Olması gereken H/2 (en az 3 metre)
Ayrık düzen (A)					
1-4 kat	6-30 m.	10-40 m.	En az 5 metre	Blok başında en az 3 m. + her bir kat için 0,5 m.	Olması gereken H/2 (en az 3 metre)
5-9 kat	9-30 m.				
10+ kat	12-30 m.				
Yapılanma koşulu					
TAKS/KAKS	Min TAKS = 0,40		En az 5 metre	Yukardaki A, BL koşulları geçerli	Yukarıdaki koşullar geçerli
E (emsal)	Min TAKS = 0,40		En az 5 metre		

Bahçe mesafesi: Yapının parsel sınırına olan uzaklığıdır.

Arka bahçe mesafesi: TİY'in 28. maddesi gereği yapı yüksekliğinin (h) yarısından az olamaz (yapılar arası en küçük uzaklık yapı yükseklikleri toplamının yarısı

olmalıdır.) Ancak parsel derinliği az ise, yapı derinliği 10 m. altına düşmeyecek şekilde, komşunun rızası ile bahçe mesafesi 3 m. ye indirilebilir.

Bu imar planı yönetmelikleri kurallar şeklinde tanımlanırsa (programda kullanılacak kuralları formüle etmek için amacıyla);

Bitişik düzen için;

*Kütle ölçüleri (**n1**) ;*Boy; 50 > n1boy > 6 (tek kütle min. cephe uzunluğu), derinlik; 40 > n1en > 10.

*Bahçe mesafesi (**ln1** ön, yan, arka);* Ön; 33 (parsel derinliği - kütle derinliği minimum) > ln1ön > 0, yan; 0, Arka; hn1 / 2 (h=kütle yüksekliği)> ln1arka > 3.

Blok düzen için;

*Kütle ölçüleri (**n2**) ;*Boy; 50 > n2 boy, derinlik; 40 > n2 en > 10.

*Bahçe mesafesi (**ln2** ön, yan, arka);* Ön; 33 (parsel derinliği- kütle derinliği minimum) > ln2 ön > 5, yan; blok kat adedi X 0,5 > ln2 yan > 3, arka; hn1 / 2 (h=kütle yüksekliği) > ln2 arka > 3.

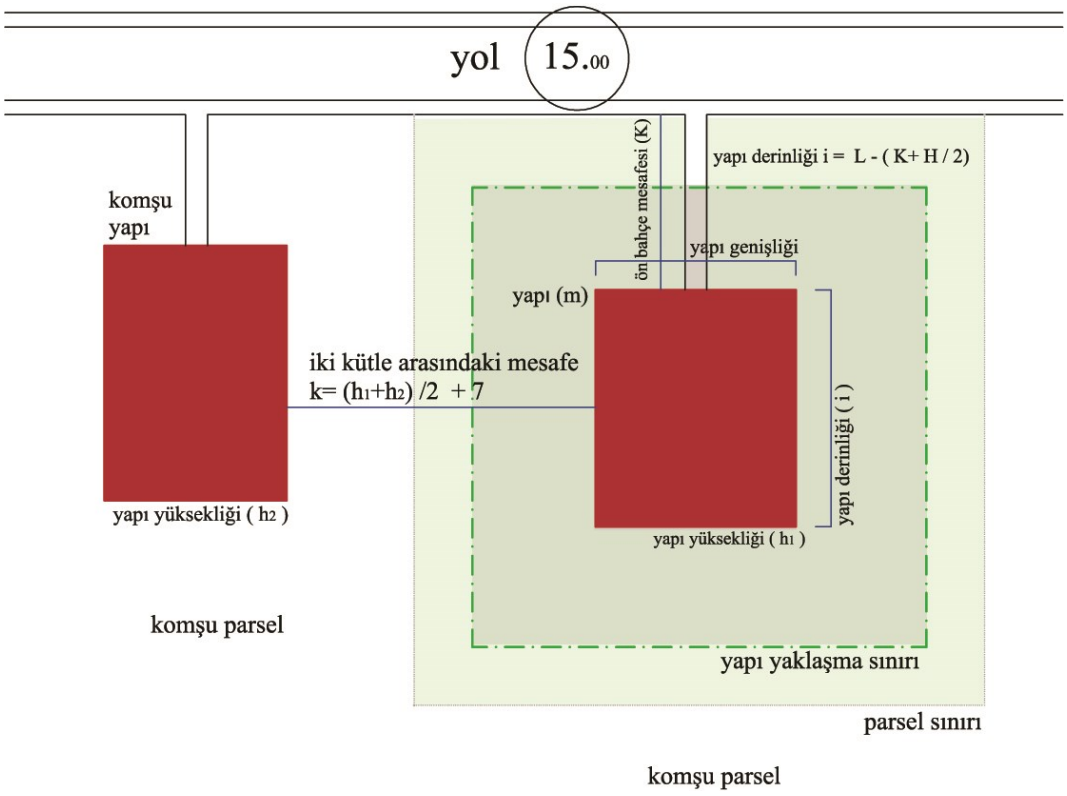
Ayrık düzen için;

*Kütle ölçüleri (**n3**)* (metre); Boy; 30 > n3 boy > 6, derinlik; 40 > n3 en > 10.

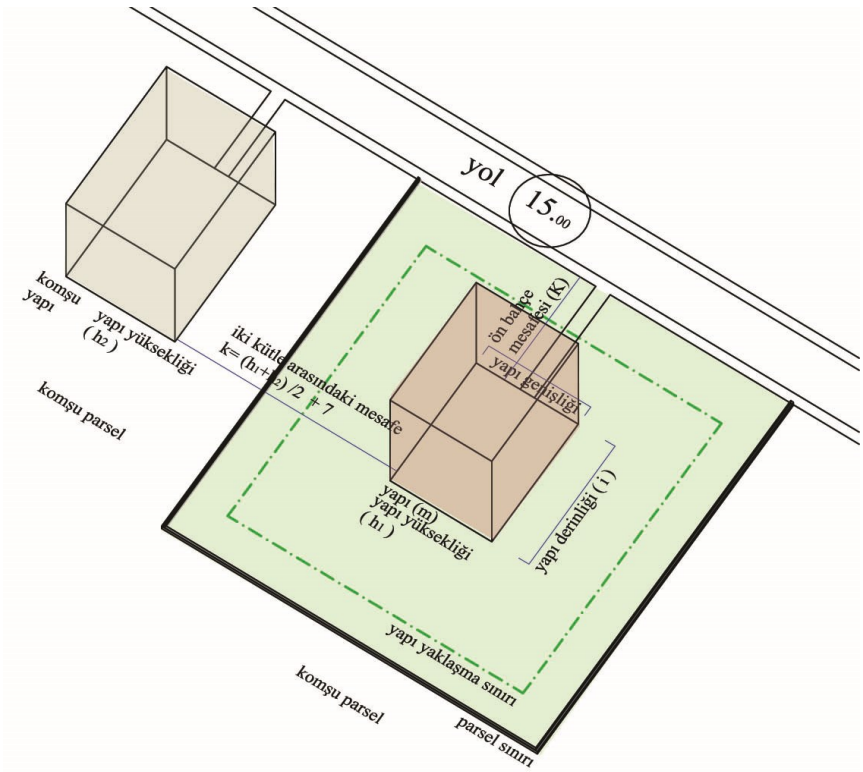
*Bahçe mesafesi (**ln3** ön, yan ,arka);* Ön; 33 (parsel derinliği- kütle derinliği minimum) > ln3 ön > 5, yan; blok kat adedi X 0,5 > ln3 yan > 3, arka; hn1 / 2 (h=kütle yüksekliği) > ln3 arka > 3.

Bunların dışında; imar planı yapı yükseklikleri ve çekme mesafeleri ile ilgili bilgiler, emsal ve maksimum kütle yüksekliği hmax, nizamlar, blok, ayrık, bitişik ve karma tip imar yönetmeliği girdileri çalışmada yer alacak konulardandır.

Tip imar planına ilişkin diğer bilgiler aşağıda (Şekil 2.16) verilmiştir. Konunun eksiksiz açıklanabilmesi için aynı çizimin üç boyutlu hali de Şekil 2.17'da verilmiştir. İki bina arasındaki mesafelerle ve yapı derinliği ile ilgili parametrik bağıntı şekillerin üzerinde görülmektedir.



Şekil 2.16 : Vaziyet planında kütle -parselle ilgili tanımlamalar (çizim: Ç. Baydoğan).



Şekil 2.17 : Perspektif (çizim: Ç. Baydoğan).

Yol genişliğine bağlı olarak değişen yapı yüksekliği; (3194 imar kanunu ve yönetmelikleri, 29. madde) Belirli bir yol aksında bulunan parsellere eşit hak tanımak ve gerekli ışık ve havayı sağlamak için arsa kullanımına getirilen sınırlamadır. Bu konuda, planda/plan notunda bir açıklama yoksa TİY nin 29. maddesi uygulanır.

- genişliği 7.00 m. den dar yollarda bodrum hariç iki kattan (6.50 m.),
- genişliği 7.00-9.50 m. yollarda bodrum hariç üç kattan (9.50 m.),
- genişliği 9.50-12.00 m. yollarda bodrum hariç dört kattan (12.50 m.),
- genişliği 12.00-15.50 m. yollarda bodrum hariç beş kattan (15.50 m.),
- genişliği 15.50 m.den geniş yollarda bodrum hariç altı kattan (18.50 m.),
- genişliği 17.00 m.den geniş yollarda bodrumhariç yedi kattan(21.50 m.),
- genişliği 19.50 m. den geniş yollarda bodrum hariç sekiz kattan (24.50 m.) den daha yüksek yapı yapılamaz (ön bahçe, yeşil alan, meydan, yol genişleme ve daralmaları, vb. unsurlar yol genişliğine dahil değildir.) (3194 imar kanunu ve yönetmelikleri, 29. maddeden düzenlenerek alımıştır.)

Yukarıda verilen TİY de belirtilen bahçe mesafeleri, yapı boyutlarına ilişkin değerler olabilecek en az değerlerdir. Daha büyük alınabilirler. Yükseklikler ise maksimum değerlerdir (Belediye ve mücavir alan sınırları içinde ve dışında planı bulunmayan alanlarda uyulacak imar yönetmeliği).

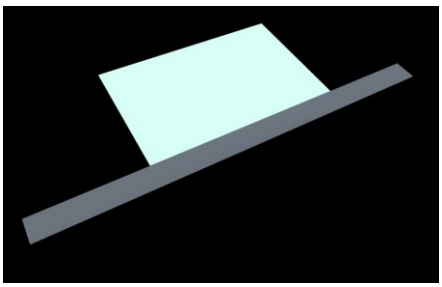
Bahçe mesafeleri ve kat yüksekliği ile arsa kullanımına getirilen sınırlamalar, kentte birbirine benzer yapılar dizisi oluşturduğu, monotonluğa neden olduğu için yoğun biçimde eleştirilmiştir (Aydemir, 1999, 2004). 1970'li yıllarda parselin esnek kullanımına olanak veren TAKS, KAKS, daha sonraları ise E (emsal)-Hmax (maksimum yapı yüksekliği) ile yapılanma yöntemi getirilmiştir. Ancak toplumdaki eğilimin, genelde parseldeki yapılanma hakkını en çoğa çıkarma yönünde olması, bu yoğunluk sınırlama yönteminin de kimlikli kentsel çevre oluşumunda başarısız kalmasına neden olmuştur/olmaktadır. Zaten, parsel ölçeğinde yapılandırmayı yönlendiren en önemli etmen emlak sahibinin tutumu ve ekonomik koşullardır (Duyguluer, 1989).

Plan üzerinde, genellikle gelişme alanlarında, parseldeki yapılaşmaya ilişkin tüm sınırlama aşağıda görülen tek formatla (Çizelge 2.2) ifade edilmektedir. Mevcut dokuda ise yapılanma koşulu ve yapı düzeni ya birlikte ya da ayrı ayrı kullanılmaktadır.

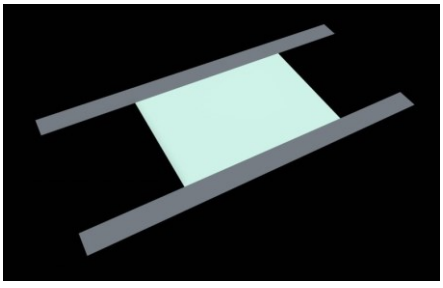
Çizelge 2.2 : İmar planlarında parselle ilgili kuralların bulunduğu bilgilendirme kutusu.

Arka bahçe derinliği	Yan bahçe derinliği	Ön bahçe derinliği
Kat sayısı	Yapı düzeni	Min. Parsel cephesi
Min. Parsel alanı	KAKS	TAKS

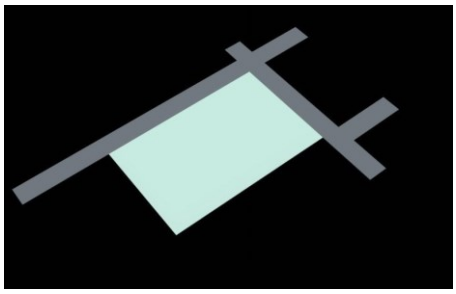
Parselin yola göre yerleşim alternatifleri aşağıdaki gibidir. Türkiye imar şartlarında dikdörtgen parselin yola göre 5 durumu vardır. Bunlar: Parselin tek kenarına komşu yol (şekil 2.18), parselin iki kenarına komşu yol (Şekil 2.19), parselin iki kenarına komşu yol - köşe parsel (Şekil 2.20), parselin üç kenarına komşu yol (Şekil 2.21) ve parselin dört tarafına komşu yol- ada parsel (Şekil 2.22). Parselin köşe sayısı arttıkça artan kenar sayısı kadar yol geçme olasılığı vardır. Büyük parseller (bir kenarı 80-100 metre ve daha büyük) veya yapı adaları için ulaşım-erişilebilirlik açısından durum farklı olabilir. Fakat bu durum küçük parseller için pratikte diğer bir deyişle imar planlarında kadastral duruma bağlı olarak mümkün değildir. Maksimum kenar sayısı (temel poligon'un) 5 i geçmez. Küçük bir imar parseli için durumlar:



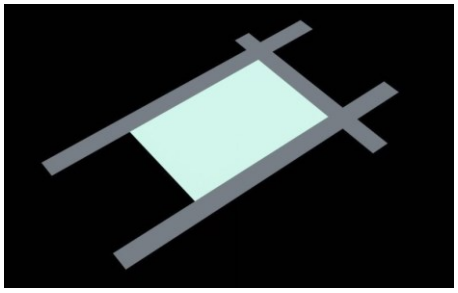
Şekil 2.18 : Parselin tek kenarına komşu yol.



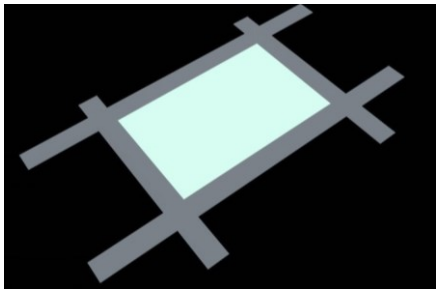
Şekil 2.19 : Parselin iki kenarına komşu yol.



Şekil 2.20 : Parselin iki kenarına komşu yol (köşe parsel).

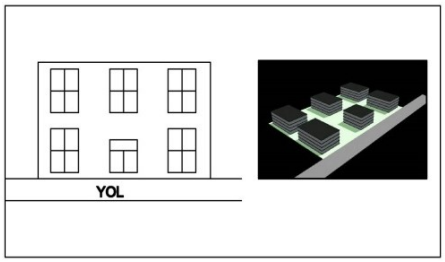
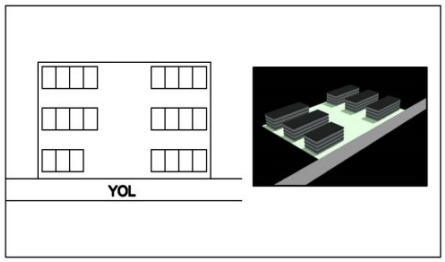
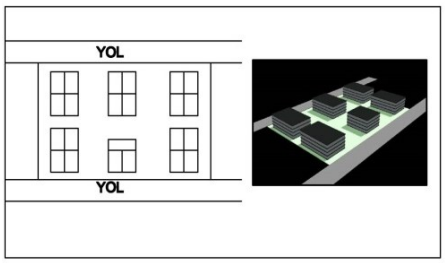
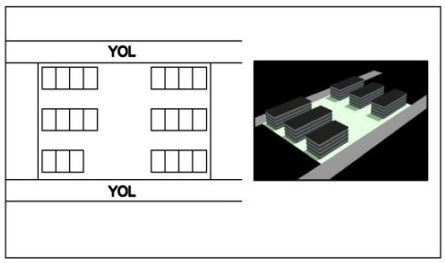
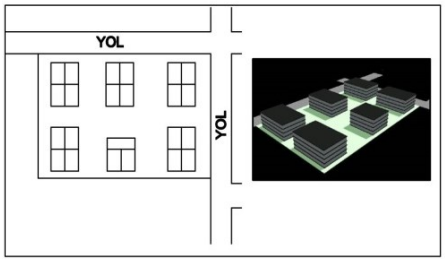
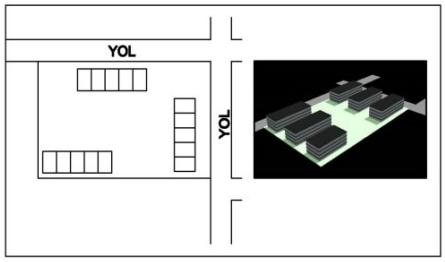
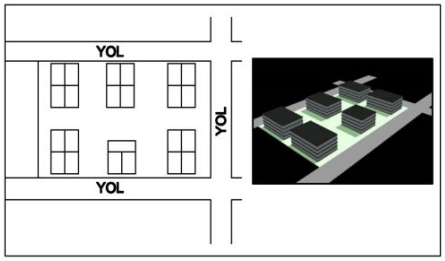
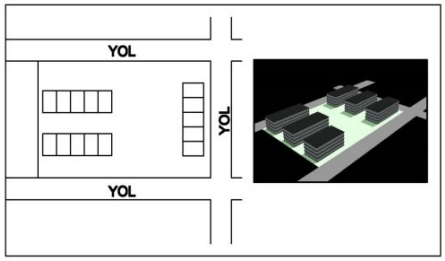
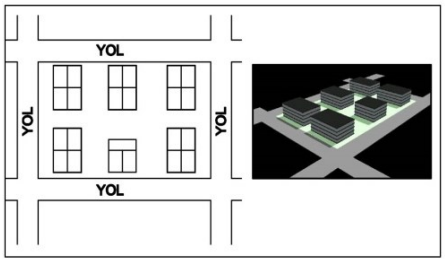
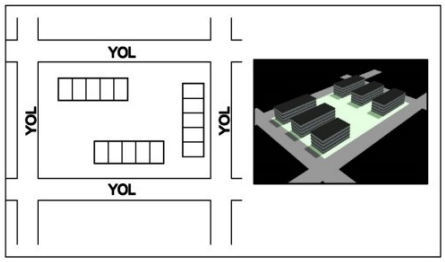


Şekil 2.21 : Parselin üç kenarına komşu yol.

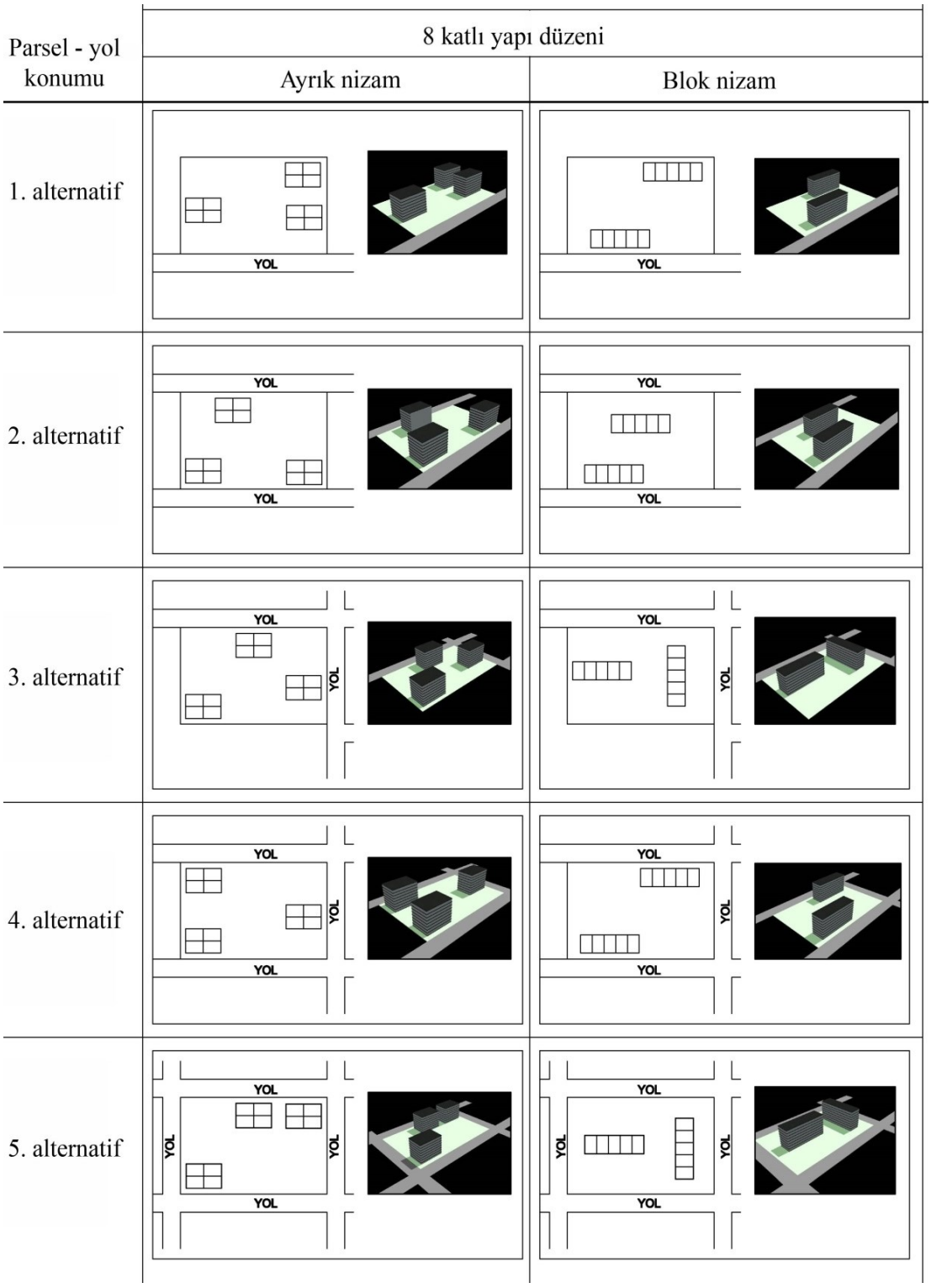


Şekil 2.22 : Parselin dört tarafına komşu yol.

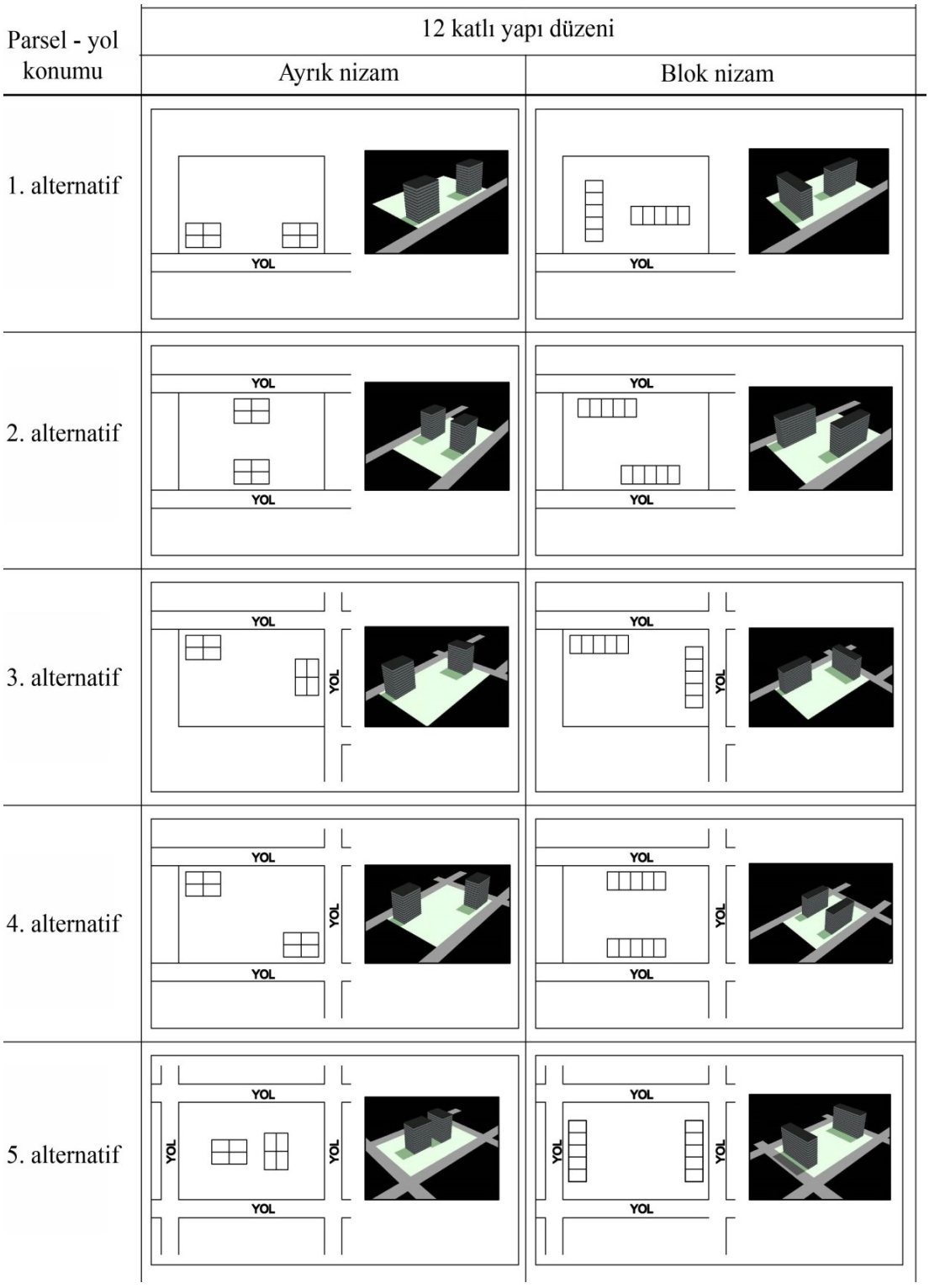
Parsel-yol komşuluğu ile ilgili olasılıklar kenar sayısı ile sınırlıdır. Türkiye imar şartlarında, geometrik olarak dikdörtgen bir yapı adası-parsel içerisinde tasarlanacak yapı grubunun çeşitlenişine ilişkin tablolar aşağıda verilmiştir. Parsel içinde yapılacak yapı grubunun yapı nizamına, kütle büyüklüğüne (en-boy), kat sayısına (yükseklik), yola göre yerleşim durumuna ilişkin alternatifler gösterilmiştir. Alternatifler 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14 ve 15 katlı olarak seçilmiştir. Aşağıdaki örnekler Türkiye şartlarında en çok kullanılan yapı nizamları olan ayrık ve blok düzene göre ayrılmıştır (Şekil 2.23, Şekil 2.24, Şekil 2.25, Şekil 2.26, EK.E). Taks ve kaks sabit kabul edilmiştir. Yapıların yerleşmelerine ilişkin öneriler temel sıralanma, yönler ortografik (0, 90 derecelik) olasılıklar içinden rastgele seçilmiştir. Amaç büyüklükleri aynı yol durumları farklı olan parsel üzerinde yapılacak yapıların sabit kısıtlar, değişmeyen kurallar ile, 90 derecelik alternatiflerin çeşitliliğini göstermektir.

Parsel - yol konumu	4 katlı yapı düzeni	
	Ayrık nizam	Blok nizam
1. alternatif		
2. alternatif		
3. alternatif		
4. alternatif		
5. alternatif		

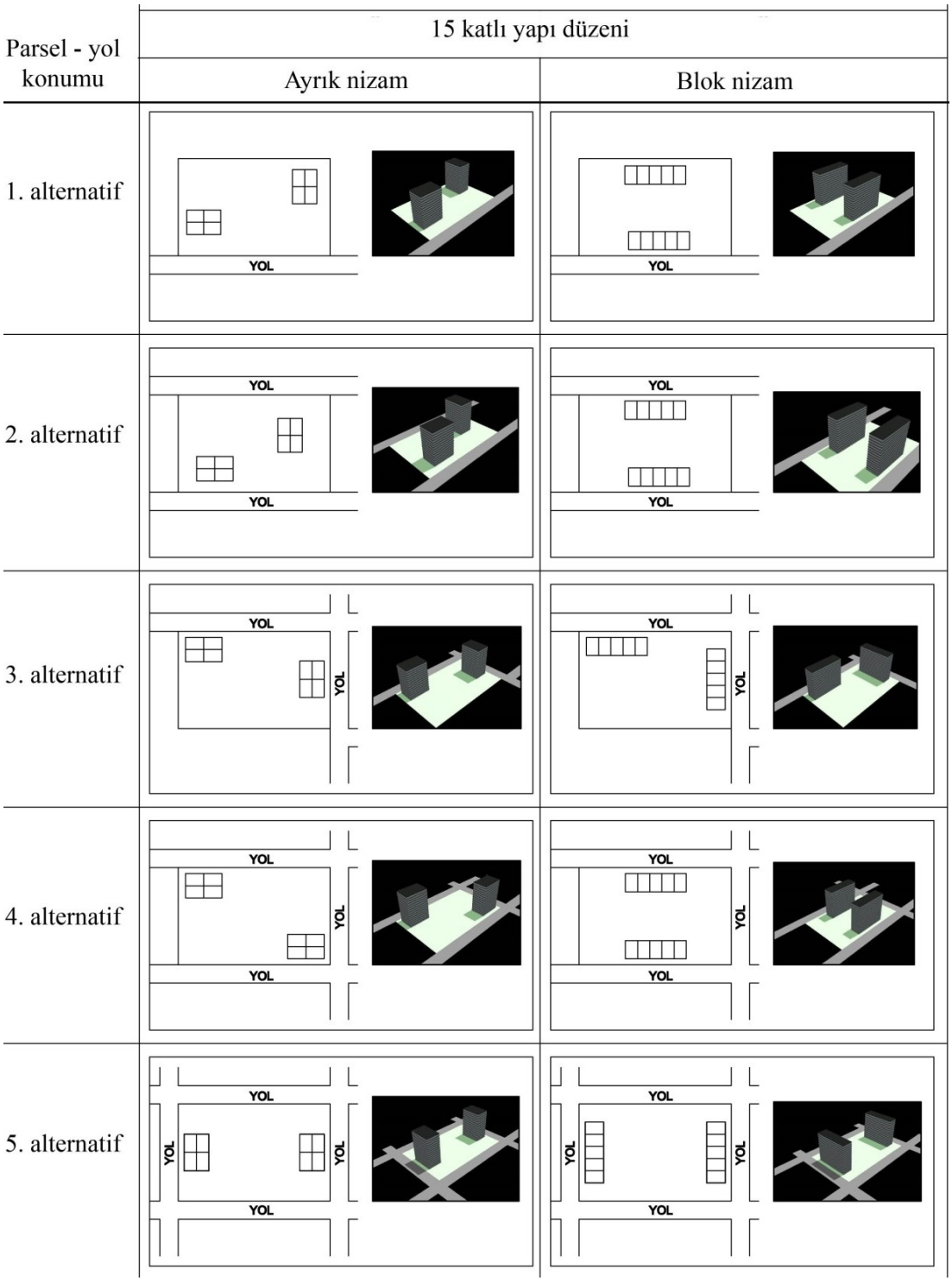
Şekil 2.23 : 4 katlı kütleler için farklı yerleşim şemaları.



Şekil 2.24 : 8 katlı kütleler için farklı yerleşim şemaları.



Şekil 2.25 : 12 katlı kütleler için farklı yerleşim şemaları.



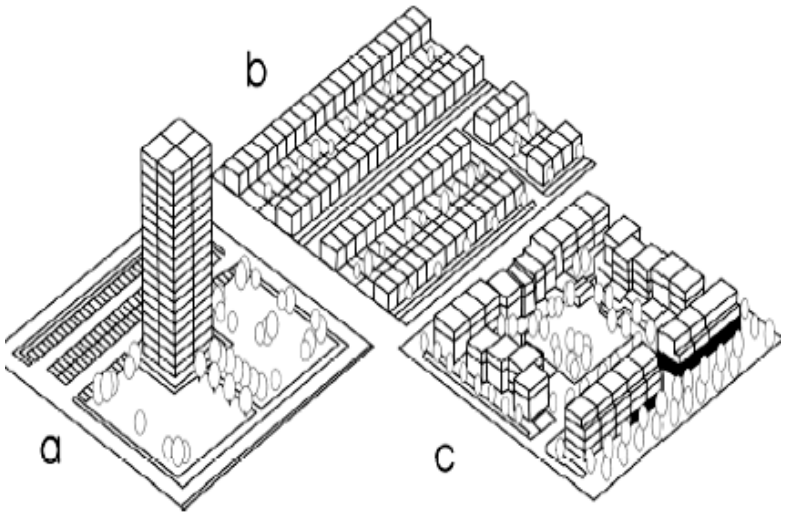
Şekil 2.26 : 15 katlı kütleler için farklı yerleşim şemaları.

Yukarıdaki alternatif vaziyet planlarından görüleceği üzere sabit taks ve kaks için çok farklı alternatifler vardır. Olası durumların çok azı yukarıdaki şekilerle aktarılmıştır. Parsel ve yol durumuna göre katsayıları baz alınarak çizilen bu kısıtlı alternatiflerden bile, parametrik bağıntıların sayısının fazlalığını ve olasılıkların çokluğunu anlamamız mümkündür. Tasarımcıların bu parametrik bağlantıların

hepsini aynı anda kontrol etmesi ve farklı alternatifleri denemesi uzun zaman alacaktır.

2.4. Binaların Yerleşiminde Bilgisayar Programına Olan İhtiyaç

Tip İmar Yönetmelikleri (TİY)'ne, kentsel ölçekte, binaların kent bütününe hangi düzenle, hangi büyüklükte entegre olacaklarına dair bir tasarım optimizasyon problemi olarak bakmamız mümkündür. Bunun sebebi bir imar parselinde tanımlanmış sınırlamaların çoğunun, birbirine bağlı matematiksel hesap işlemi ve yerleştirme problemi olarak tanımlanabilmesidir. Bu probleme mimari ölçekten bakacak olursak, imar planlarına verilen büyüklük diğer bir deyişle yoğunluk kısıtlamalarını kütle büyüklüklerine uygulamak, blok sayısı, kat adedi, en, boy, yükseklik gibi niceliksel boyutları hesaplamak ve son olarak bu kütleleri uygun çözümde arsa üzerine yerleştirmek kapsamlı matematik hesapları barındıran, parametrik bir tasarım optimizasyonudur. Bu durumu, Biddulph (2007) 'ın 75 konut/hektar yoğunluktaki alan için farklı alternatiflerin neler olabileceğine dair çizimi (şekil 2.27) üzerinden görebilmek mümkündür.



Şekil 2.27 : 75 konut/hektar yoğunluğundaki bir konut adası için farklı çözüm önerileri (Biddulph, 2007).

Genel olarak, mimari ölçekten bu soruna bakacak olursak; tasarımcılar, tasarım aşamasına gelmeden önce, yoğun olarak veri toplamaktadırlar. Toplanan veri, belirli kurallar çerçevesinde değerlendirilip, belirli sonuçlar çıkarıldıktan sonra tasarım aşamasına geçilmekte ve belirli araçlar kullanılarak, tasarlanan çalışma alanı sunulmaktadır. Yukarıda kısaca özetlenen, bu süreci dört ana başlık altında

inceleyebiliriz. Veri toplama, sentez çalışmaları, tasarım ve sunum. Tasarım sürecinde ilk aşama veri toplamaktır. Analiz çalışmaları, kullanıcı bilgileri, görsel tespitleri imar planları ve mevzuat bilgileri, veri toplamanın alt başlıklarıdır. Toplanan veriler, daha sonra tasarımcı tarafından, daha önceki deneyimleri ve genel kentsel tasarım prensipleri doğrultusunda, bir senteze ulaşmak üzere değerlendirilir.

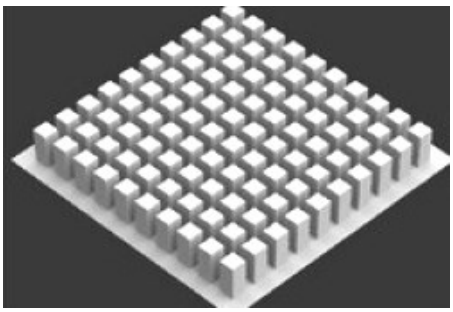
Bütün bu aşamalardan sonra, kullanıcı veya belediyeden gelecek, yeni öneriler ve istekler sonucunda, yeni sentezler yapılacak, tasarımlar yenilenecek ve yeniden sunulacaktır. Tüm bu aşamaları eğer en ideal tasarıma ulaşmak istenirse daha birçok kez tekrarlanacaktır. Bu süreç içerisinde tasarımcı, genelde zaman ve iş gücü açısından; hiç bir zaman en ideal tasarıma ulaşmak konusunda bir çaba göstermeyecek ve ortalama çözümü bulmaya çalışacaktır.

Kullanıcı, işveren, belediye isteklerine göre değişen, kütle boyutları, yeniden tekrarlanan, zahmetli hesapların ve çizimin yapıldığı bu süreçde optimizasyon problemini çözecek bir algoritma sayesinde, tasarım ve tip imar yönetmelikleri konrolü daha ideal bir noktaya ulaşacaktır.

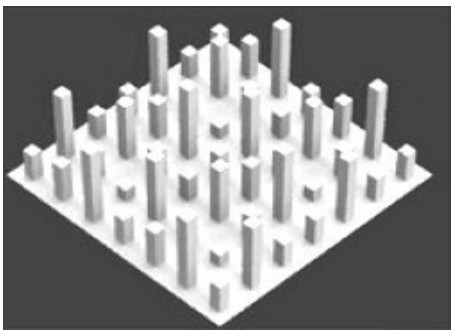
İmar planı, imar yönetmeliklerinin kent içine uygulama konusunu ele aldığımızda ise, gerçekte çok kompleks bir yapının varlığı ortaya çıkar. Tasarımcı kent mekanı içerisinde oluşturacağı bina, binaları varederken bir çok konuya karar vermek zorundadır. Vereceği kararlarda kenti ilgilendiren, kullanıcıları, işvereni, kontrol mekanizmalarını ve diğer çatışan istekleri çözüme kavuştururken gözden kaçırabileceği noktalar olabilir. Örneğin; dikkat ve titizlik gerektiren alan hesabını yaparken, Türkiye özelinde işverenin isteği sonucu toplam inşaat alanını genelde maximize, imar kanunu ve belediye yönetmelikleri gereği hataları minimize etmek zorundadır.

Şekil 2.28 ‘de konut adası (Cheng ve ark., 2006) üzerinde imar şartlarında yoğunluk bölgelemesi için çok kullanılan TAKS, KAKS ‘ın alacağı matematiksel değerlere göre kent içi imar parselinin yoğunluğunun nasıl değişeceği örneklenmiştir. Plancıların ve tasarımcıların bu matematik bağıntılardan çıkacak sonuçları tecrübeleri vasıtasıyla tahmin edebilmeleri mümkündür. Fakat çok küçük matematiksel değer değişikliklerin kent içinde yaratacağı sorun ve problemler göz ardı edilmemelidir. Aşağıda aynı kare parsel üzerine farklı yoğunluk değerlerinde konut kütleleri alternatifleri görülmektedir. Bu parametrik ilişkilerin yapay zeka ile

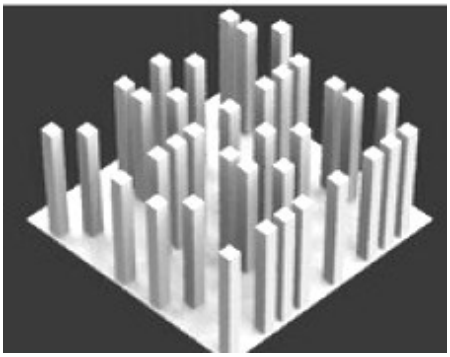
hesaplanması ve konunun kapsamı gereği alternatiflerin oluşumununun tasarımcılar tarafından algılanılması için matematiksel bağlantıların bilinmesi gerekmektedir.



TAKS:2,7 KAKS: 3,6



TAKS:1,1 KAKS:1,4



TAKS:1,1 KAKS:3,6

Şekil 2.28 : Konut adası ve yoğunluk ilişkisi (Cheng ve ark., 2006).

Yukarıda anlatılan etkenlerden dolayı, tek bir tasarımcı yerine, değişik konularda uzmanlaşmış bir tasarım grubu tarafından tasarımın yapılması gerekliliği ortaya çıkmaktadır. Ayrıca, meslek etiği açısından tasarımcılar diğer tasarım süreçlerinde olduğu gibi; arsa üzerine uygulanacak projenin bütün kontrol ve uygulama aşamalarını yerine getirmek zorundadır. Bu durumda mimarlar için TİY lerini oluşturacakları binalar için, kontrol edebilecek, gerektiğinde parsel içerisine tanımlanmış problemi çözebilecek bir bilgisayar programının varlığına ihtiyaç duyulmaktadır.

Kentsel tasarım, nazım imar planı (1/5000 ölçek) oluşturma, yoğunluk bölgeleme sürecine bakıldığında, tasarımcı yerine, tasarım sürecinde tasarımcıya destek verip tasarlanan kenti, düşünülen yoğunluk hesabını iki veya üç boyutlu olarak kullanıcıların anlayabileceği gibi gösterip, istenilen anda uygulamaya yönelik planları oluşturacak, bir araca ihtiyaç olduğu ortaya çıkmaktadır. Bu sürecin neredeyse tamamında günümüz olanakları ile bilgisayar ve teknolojilerinin kullanılması mümkündür (Elliiki, 1996).

Bilgisayar teknolojileri (GIS, vb.) ve programlar kentsel tasarım sürecine enaz, tasarımcıların yapmak zorunda oldukları yoğunluk, yerleşim ve boyut kararlarını verdikleri aşamada katkı sağlamaktadır. Fakat tasarımcıların tasarım süreci içerisinde oldukça fazla zamanlarını alan bu aşama ve yapılan kentsel tasarımın gelecekte karşılaşacağı durumlar ve performansları için test edilmesi işlemleri kentsel tasarım açısından kritik bir noktada bulunmaktadır.

Bu tez kapsamında yapılan çalışmaya benzer-yakın çalışmalar, bu konu üzerine yapılmış farklı yazılımlar, yerleşim kararına destek olan bilgisayar programlarının sayısı dünya genelinde ve Türkiye özellinde çok azdır. Genelde Avrupa ve Amerika'da da matematiksel hesaplamalamalar-nesnel verilerle yapılan kentsel tasarım ve imar yasaları-yönetmeliklerine uyum aşamalarının tanımlanmasına, kentsel, imarla ilgili önerilerin çözümüne, üretimine ilişkin fazla çalışma olmaması, bu çalışmanın doğru bir alanda olduğunu, bu eksikliği kapatabileceğini, kompleks olduğu düşünülen bir problemin çözümüne ilişkin doğru kararlar verebilecek bir algoritma geliştirdiğini ispatlamaktadır.

Bu alanda yapılan çalışmalardan olan Modelur; Skechup üzerinde çalışan, kentsel tasarım yardımcı programıdır. Kentsel ölçekte, imar planı-halihazır plan altlığı üzerinde yönetmeliklere uygun üç boyutlu mimari kütleler oluşturmaya yardımcı olmaya çalışan bir programdır. Oluşturulan kütleleri analiz edebilmektedir. Yoğunluk, kullanım, yükseklik, otopark hesabı, yeşil alan hesabı vb. gibi hesapları kütle üzerinden yapıp detaylı bilgi verebilmektedir. Geleneksel programlama yöntemleriyle yapılmış skechup üzerinde çalışan bir script (plug-in) dir. Matematiksel olarak doğru, imar yönetmeliğine uygun alternatifleri üretme kapasitesi yoktur. Skechup programının 3. boyuta hakimiyeti sayesinde, Modelur kolaylıkla 3 boyutlu olarak kullanılabilmektedir.

3. MİMARİ TASARIMDA OPTİMİZASYON PROBLEMLERİNE İLİŞKİN TEMEL KAVRAMLAR: YAPAY ZEKÂ, SÜRÜ ZEKÂSI, ABC ALGORİTMASI

Bu bölümde; çalışmanın alt yapısını oluşturan ve bilgi ve çalışma alanı olan bilgisayar programlamaya ait temel kavramlar sunulmuştur. Ayrıca bu kavramların tasarım sürecine katkıları, bu katkıların gelişimi, nedenleri vb. konular aktarılacaktır. Yapay zeka ile ilgili özellikle mühendislik alanında bir çok çalışma vardır. Konunun çok kapsamlı olması ve içerisinde bir çok farklı sistemi barındırması ayrıca çalışmanın kapsamı nedeniyle, yapay zeka başlığı altında bulunan; optimizasyon, popülasyon temelli algoritmalar, sürü algortiması, konularına mimari tasarım çerçevesinden bakılmaya çalışılmıştır.

Çalışmanın içerisinde yer alan bilgisayar programlama sürecine temel oluşturacak, ana kavramlar ve gelişimleri aşağıda tanımlanmıştır. Tasarımcılara katkı sağlayacak programın bu bölümde aktarılan, açıklanan ve genelden özele doğru sıralanan konu başlıklarıyla, nasıl bir algortima olduğu anlaşılabilmektedir. Çalışmanın teorik alt yapısı 3. bölüm genelinde genelden özele, hiyerarşik bir düzene göre aktarılacaktır. Çalışmanın bilgisayar bilimleri içerisinde nerede yer aldığı aşağıda hiyerarşik olarak belirtilmiştir, genelden özele doğru yapılan belirleme işlemi

Yapay zeka genel başlığı altında;

- *Optimizasyon problemi çözümleri*
  - *Evrimsel hesaplamalı yaklaşımlar (popülasyon tabanlı metotlar) (Evolutionary Comptuation Approaches)*
    - *Sürü zekası (Swarm Intelligence)*
      - *Arı davranışlarına dayalı algoritmalar (Bee Algorithms)*
        - *Yiyecek arama davranışına dayalı algoritmalar*
          - *ABC (Artificial Bee Colony)*
            - *Çalışma*

3.1 Yapay Zeka ve Gelişimi

MIT Bilgisayar Bilimleri Laboratuarı uzmanlarından Edward Fredkin'e göre *"Tarihte üç büyük olay vardır. Bunlardan ilki evrenin oluşumudur, ikincisi ise yaşamın başlangıcıdır. Bu ikisiyle eşdeğer önemli olan üçüncüsü ise yapay zekanın ortaya çıkışıdır."* (İnam, 2001, s. 38-45).

Yapay zeka ile ilgili düşüncelerin veya çalışmaların tarihçesi 1920'lere kadar dayanmaktadır. 1921 yılında Çek yazar Karel Capek'in Rossum'un Evrensel Robotları (R.U.R) oyunu birçok dile salt "Robot" sözcüğünü değil, günümüzde de hala geçerliliğini koruyan "robot" kavramını tanıştırmıştır (Capek, 2001). 1942 yılında Isaac Asimov 'Kovalamaca (Runaround)' (Asimov, 1942) adlı kısa hikayesinde de "robot" ve "robotik" sözcüklerini kullanmış, daha sonra hem bu hikayesini hem de birçok başka robot hikayeleri içeren 'Ben-Robot' (I-Robot) adlı kitabı yazmış ve günümüzde de bir ütopya olan "Pozitronik Beyin" fikrini ortaya atarak ilk kez elektronik olarak tasarlanmış ve "yapay zekaya" sahip bir beyin fikrini ortaya atmıştır (Asimov, 1991). 1950'de, bir İngiliz mantık ve matematikçisi olan Alan Turing, Mind adlı felsefe dergisinin Ağustos sayısında, "Computing Machinery and Intelligence" adlı bir makale yayımlayarak "Makineler düşünebilir mi?" sorusunu gündeme getirmiştir (Penrose, 1989.). 1969 yılına geldiğimizde ise Stanley Kubrick tarafından çekilen "2001 Uzay Efsanesi" (2001: A Space Odyssey) film bu bağlamda özel bir ilgi uyandırmıştır. Özel bir görev için Mars'a gönderilen ekibin uzay aracını yöneten "HAL-9000" adlı sistem, tıpkı insan gibi konuşma, görme, algılama gibi yeteneklere sahip, düşünebilen, duygusal tepkiler verebilen insansı bir makine olarak karşımıza çıkmıştır (Topçuoğlu, 2001, s. 38-45).

Robot sözcüğünün sözlük tanımında "yeniden programlanabilen ve ancak insanların yapabileceği düşünülen, özelleşmeyi gerektiren birçok işlevi çeşitli biçimlerde programlanarak yapabilen otomatik manipülatörler yada araçlar, veya insansı makineler" biçiminde olduğu görülür (Url-3). Yapay zeka araştırmacılarının en başından beri ulaşmak istediği amaç aslında insan gibi düşünen insan gibi davranan sistemler yaratmaktır. İnsansı makineler kavramı ve bu kavramla özdeşleşmiş olan "Yapay Zeka" kavramı hem sanatçılara hem de birçok farklı alanda araştırmalar yapan bilim insanlarına esin kaynağı olmuş ve günümüzde bilişim teknolojilerinin de katkısı ile daha şimdiden birçok yaşam ve araştırma alanının içinde sürekli gelişen ve

daha iyileşen biçimde yer almaya başlamışlardır. Yapay zekâ, akıllı bilgisayar programları ve akıllı makinenin tasarımı üstüne odaklanmış bir mühendislik bilimidir (McCarthy, 2004). Yapay zekâ günümüzde birçok aktif kullanım alanına sahiptir. Yapay zekâ, çeşitli sorunların çözümünün hızlandırılmasına da yardımcı bir araç olarak kullanılmaktadır.

Yapay Zeka (AI-Artificial Intelligence); öğrenme, gerekçeleme, problem çözme, yabancı bir dili alma v.b. gibi insanoğlunun davranışlarını gösterebilen sistemlerle ilgilenen bir bilgisayar bilimidir. Yapay Zeka'nın ana amacı insanların davranışlarının ve sezgisel yeteneklerinin bilgisayar üzerinde benzetimidir. İnsanoğlu esas olarak bilgi'yi kullanmakta ve onu işlemektedir. Bu yüzden bilgi ve bilginin kullanımı yapay zeka'nın da anahtar karakteristikleridir.

Yapay zekanın gelişimi

"Yapay Zeka"yı Minsky "İnsanlar tarafından yapıldığı takdirde zeka gerektirecek şeyleri gerçekleştiren makineleri yapış bilimi" olarak tanımlar (Minsky, 1968). Yapay zeka konusunda gerçek problem insan zekasını tanımlamanın zorluğudur. Belirlenmiş herhangi bir amaç için programlanan çözüm, hatta program insandaki zekanın bir göstergesi olarak kabul edilen bazı şeyleri yapmış olsa bile, yine de insan zekasının yansıması ya da bir ürünüdür. Programın davranışı zeki olarak kabul edilmez; çünkü onun, insanın üretmiş olduğu bir algoritmayı çalıştırmakta olduğu bilinir. Program kendi kendine algoritma üretmediği için, programın insanın uyguladığı aynı yöntemle problemi çözmediği bilinir ve bu nedenle onun davranışı zeki olarak tanımlanmak istenmez (Yazdani, 1986).

Yapay Zeka'dan ne zaman ve hangi durumlarda bir bilim olarak söz edilebilir?

- Bir programda denenebilir model fikri güçlüyse,
- Her olay hiyerarşik olarak açıklanamıyorsa,
- Heuristikler ve sistematik çalışma uygun bulunuyorsa,
- Heuristiklerin kullanımı "zeki davranış" kriterine uyuyorsa,
- Otomatize araştırma işlemine yönelme söz konusuysa,

bu durumda yapay zeka bir bilim olarak değerlendirilebilir.

Yukarıda sayılan koşullar yapay zekanın ilk ana hedefinin başarısıyla ilgili prensiplerdir (Yazdani, 1986, s.28).

Yapay zeka araştırmalarının ilk aşamaları; dünyaya ilişkin bilginin, genellikle, "zeki davranış" olarak bilinen insanın becerisinin ardında olduğu gerçeğine önderlik etmiştir. Yapay zeka, bilgisayarda zeki davranış üretme girişiminde, insanların genel bilgi birikimine, ilişkin bilgi miktarı ile sınırlıdır (Yazdani, 1986). Burada "zeki" den kasıt şudur: Eğer kolaylıkla bir sınıfta oturup bir hocanın konuşmasını dinleyerek ve yüzlerce kitap okuyarak bir dalda uzman olabilen bir bilgisayar düşünülürse, ancak o zaman bir bilgisayar insan seviyesinde zeki kavramına yaklaşır. Böyle bir bilgisayar, etkin şekilde ona verilen bilgiye dayanarak, kendi kendini programlayacaktır (Schank, Childers, 1984). Öyleyse bilgisayar bu programlamayı nasıl yapabilir? Gerçekte donanımın kendince bir fikri yoktur, fikri insanlar verirler. "Bilgisayarlar zeki olabilir mi ?" sorusunu sormak, "Bu makineyi zeki davranır gibi göstermeyi başarabilecek herhangi bir yazılım var mı? " sorusuna eşdeğerdir (Schank, Childers, 1984). Bir başka deyişle, makineyi düşünülen anlamda "zeki" kılabilmek için bilişsel açıdan insan beyninin işleyişine benzeyen programlar yapılmalıdır.

Bilişsel benzerliğe sahip programları üretme girişimi en azından bazı avantajlar getirir. Birincisi, insan "Yapay Zeka" için sahip olunan en iyi modeldir. İkincisi insan bilişini (algısını) taklit etme yaklaşımları, ona ait açıklamaları detaylandırmaya olanak verebilir (Yazdani, 1986, s.267-268). Programlar, hiç insan-benzeri olmayan bir şekilde insan-benzeri davranış üretebilir (Yazdani, 1986, s.278).

Özetle, insan zekasıyla bir düşünceyi kendiliğinden üretebilir, oysa yapay zeka, ancak üretilmiş bir düşünceyi taklit edebilir. Bilişsel anlayış yeteneğine sahip bir bilgisayar aşağıdakilerden bazılarını yapabilecektir (Schank, Childers, 1984).

- Deneyimlerinin sonucu olarak değiştirebilir ya da öğrenebilir.
- Geçmiş deneyimlerle, mevcut deneyimler arasında zekice ilişki kurabilir.
- Kendisi için yeni olan enformasyonu formüle edebilir ve deneyime dayanan çıkarımlara kendi kendine varabilir.

- Kendi kendine bir sonuca ulaşmada hangi düşünce işlemini oluşturduğunu ve kurduğu bağıntıların nedenini açıklayabilir (Schank, Childers, 1984).

3.2 Optimizasyon, Optimizasyon Problemi ve Çözümleri

Bu tez çalışması genel itibariyle, yerleştirme ve hesaplama işlemlerini barındırmaktadır. Optimizasyon problemi olarak tanımlanan bu işlemleri, problem çözme, optimize etme sürecindeki aşamaları anlamak için; genel anlamda optimizasyon işlemini ve ne amaçla kullanıldığını bilmekte yarar vardır. Tip imar yönetmeliklerine uygun yerleştirme, düzenleme ve hesap etme işlemlerinin niye bir problem sayıldığına ilişkin cevap, konunun anlaşılabilmesi için verilen tanımlar, sistemin mantığı ve süreç aşağıda verilmiştir.

Optimizasyon mevcut koşullar altında en iyiyi arama sürecidir. Daha teknik bir ifadeyle, belli kısıtlar altında (eğer varsa) bir fonksiyonun maksimumunu ya da minimumunu bulmayla ilgilenen bir matematiksel disiplindir. Optimizasyon modelleri ise bir sistem çıktısını en iyilemek için, sistemin ilişkilerinin matematiksel ifadelerle tanımlanmış biçimidir. Optimizasyonun ilk ortaya çıkışını, 1940'larda George Dantzig'in (Dantzig, 1991, 2002), askeri uygulamalar için (eğitim programları ve zaman çizelgesi) matematiksel teknikleri kullanmasıyla göstermiştir. O tarihten bu yana, doğrusal programlama teknikleri ve uzantıları üretim araçlarını programlamadan hava yolları yönetimine geniş bir alanda uygulana gelmiştir. Bugün, optimizasyon yöneylem araştırması, yapay zeka ve bilgisayar bilimleri gibi bir çok alanda karar vermenin nesnel aracıdır (Bergey ve Ragsdale, 2005.).

Kimya, biyoloji, fizik, astronomi gibi temel bilim dalları ile elde edilen bilgilerden faydalanarak, araçların, eşyaların, objelerin, ürünlerin diğer bir deyişle sistemlerin tasarımı ve gerçekleştirilmesi ise tasarım-mühendislik bilim dallarının görevidir. Var olan veya tasarlanan sonradan üretilmiş/üretilen/üretilecek sistemlerin belirli şartlar altında nasıl davranış sergilediklerinin incelenmesine, ortama ve sisteme ilişkin bilgilerin ortaya konmasına "analiz" denir (Karaboğa, 2004). Mühendislikte yapılan bilimsel çalışmaları kabaca iki ana gruba ayırabiliriz: Analiz ve tasarım. Mühendislikte ne tür çalışmalar olduğuna bakılacak olursa; tasarımın içinde var olmadığı bir alt dalın ya da bölümün mevcut olmadığı görülebilmektedir.

Tasarlanan bir sistemin istekleri ne kadar sağlayıp sağlayamadığı elbette tasarımın ne kadar iyi yapıldığına bağlıdır. Tasarlanacak sistemden beklentiler arttıkça daha karmaşık hale gelmekte ve problemin çözümü de o derece zorlaşmaktadır. Bir sistem için istekleri doğru şekilde sağlayan çok sayıda olası alternatif çözüm bulunabilir, bu çözümler içerisinden en iyisinin (optimum), en kısa zamanda bulunması önemli bir mühendislik çalışmasıdır. Bu ve benzeri çalışmalar "optimizasyon problemi" olarak adlandırılır (Karaboğa, 2004).

Tasarım sürecinde problemler ve çözüm yolları birçok kez değiştirilir, geliştirilir ve böylece daha iyiyi, daha ucuza bulma yolları aranır. 1950'den günümüze geliştirilen optimizasyon teknikleri, birçok tasarım problemlerinin çözümünde kullanılmıştır. Özellikle bilgisayar teknolojisindeki gelişmelere paralel olarak bu uygulamalar çok daha kolaylaşmış ve yaygınlaşmıştır.

Optimizasyon işlemlerinde uygun bir çözüm alanını elde etmek için çok çeşitli arama teknikleri kullanılmaktadır. Bunların çoğu üç temel sınıfta gruplandırabilir. Bunlardan birincisi hesaplama tabanlı tekniktir. İkincisi, çözüm uzayında her noktayı ele aldığımız ve bunları değerlendirdiğimiz tekniktir. Hesaplama yoğunluğu nedeniyle bu türden bir yaklaşım, özellikle basit problemler haricinde sınırlı fayda sağlamaktadırlar. Üçüncüsü ise yönlendirme yaparak rasgele arama yapabileceğimiz bir tekniktir. Bu tekniği kullanırken, arama işlemlerinde daha iyiye gitmek için sezgisellikten faydalanılmaktadır. Gelişime Dayalı (GD) algoritmalar bu gruba dahil edilen algoritma türleri olarak karşımıza çıkmaktadır (Bergey and Ragsdale, 2005).

Gerçek hayat problemlerinin çözümleri için karar vermemiz gerekir. Karar verme sürecinde düşünülen çözümlerin kısıtlı olması veya çözüm arayanın bakış açısının yeterince geniş olamaması nedeniyle önerilen veya gerçekleştirilen çözümler ancak uygun çözüm veya olası çözüm olarak nitelenebilir.

Zorlaşan hayat koşulları nedeniyle uygun veya olası çözümlerin yetersiz olduğu ve bu çözümler arasında en iyi bulunamadığı sürece başarının ve rekabetin kısıtlı olduğu açıktır. Başarının arttırılması amacıyla karar verme sürecine sistematik yaklaşmak akılcı olup, bu sistematik yaklaşımın doğru uygulandığı birçok durumda başarı seviyesi artmaktadır (Sinanoğluve Börklü, 2002).

Optimal tasarım metodları;

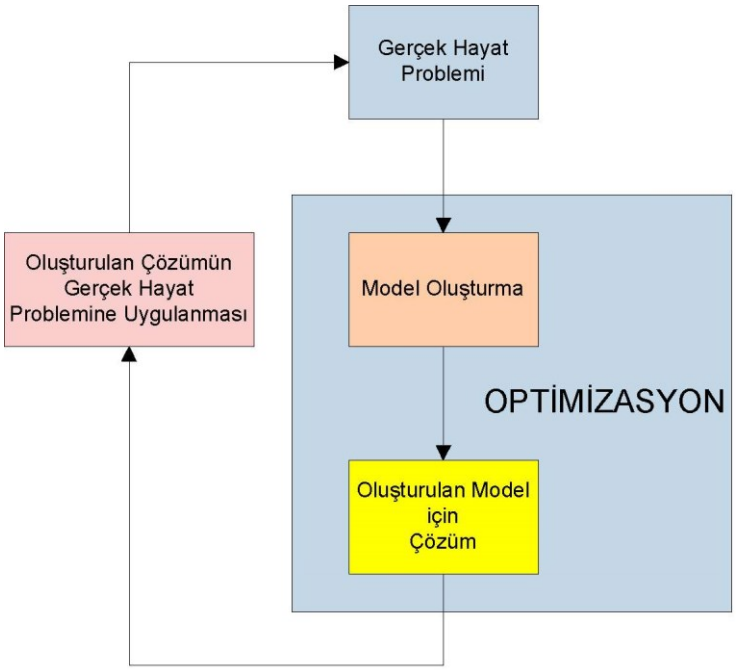
1. Evrimle optimizasyon: Benzer tasarım veya tabii sistemlerin incelenmesiyle tasarımın geliştirilmesi.

2. Sezgiyle optimizasyon: Sezgi veya ilhama dayalı tasarım çözümleri bulma.

3. Deneme yanılma yöntemi ile optimizasyon: Gerçekte, optimizasyona dayalı tasarım olarak adlandırılmaz.

4. Nümerik algoritmalarla optimizasyon: Bu, mevcut gelişme alanıdır.

Gerçek hayat probleminin optimizasyonla çözümü şekil 3.1'de şematik olarak aktarılmıştır. Bu çözüm şemasının basitleştirilmiş hali, optimizasyon işleminin temel aşamaları olarak şekil 3.2' de verilmiştir.



Şekil 3.1 : Gerçek hayat problemlerine sistematik çözüm yaklaşımı **(**Yaman, 2005).



Şekil 3.2 : Optimizasyon sürecinin temel aşamaları **(**Yaman, 2005' den uyarlanmıştır).

Optimizasyonun tercih edilme sebepleri ise;

- Matematik, mühendislik, tasarım ve gerçek hayat problemleri arasındaki benzerliklerin ve geçişlerin varlığı,

- Optimizasyonun mevcut durum ve şartlarda en iyiyi belirleme ve seçme anlamına gelmesidir (Yaman, 2005).

Gerçek hayat problemleri ve tasarım gerektiren (yerleştirme, düzenleme, vb.) hesaplamalar optimizasyon yardımıyla çözümünde öncellikle bu problemleri ifade edebilen matematik modeller oluşturulur. Böyle bir modelde ise bir amaç fonksiyon ve kısıtlar vardır. Bu amaç fonksiyonunun alabileceği değerler (kısıtlar dahilinde) uygun çözümü verirler. Uygun çözümde en iyiyi sağlayan değer optimum olarak bilinmektedir. Uygulamalı bilimler, mühendislik, ekonomi, tıp ve istatistik alanlarındaki problemler optimizasyon problemleri olarak değerlendirilebilir. Özellikle matematiksel modeller problemleri anlayabilmek ve analiz edebilmek açısından oluşturulurlar. Optimizasyon da modelin karakteristiklerinin gerçeklerle ilişkili olduğu savunularak en iyi veya en uygun değeri belirlemek için kullanılır.

Optimizasyon ile ilgili bilgi, formül olarak ifade edilecek olursa; bir dizi terimi yeniden açıklamak gerekir.. Sonuç olarak; optimal tasarım, mümkün tasarımların en iyisi olarak tanımlanabilir. Optimizasyonu ise, arzulanan bir çokluğun maksimum yapılırken; arzulanmayanların minimum yapılması işlemi olarak tanımlamak mümkündür. Optimizasyon teorisi, maksimum ve minimum özelliklerini ele alan ve bunları bulmaya yarayan bir matematik dalıdır. Genel bir optimizasyon durumunda tasarımcı, bağımsız değişkenlerin nümerik değerleri sabit olmayan genel bir oluşumunu yaratmaktadır. Burada bağımsız değişkenler cinsinden tasarımın değerlerini tanımlayan bir amaç fonksiyon saptanır (Sinanoğlu ve Börklü, 2002, Yaman, 2005). Yaygın amaç fonksiyonlar; maliyet, ağırlık, güvenilirlik ve üretilebilirlik olabilir. Amaç fonksiyon, belirli sınırlayıcılara maruz bulunmaktadır. Sınırlayıcılar, fiziki kanunlar ve sınırlandırmalar veya bağımsız değişkenler üzerindeki uyumluluk şartlarından çıkmaktadır.

Tasarım uygulamalarında tabii olarak oluşan bir bölgesel sınırlandırma çeşidi, tasarım tanımlarına dayanmaktadır. Tasarım tanımları, sistemin diğer parçaları ile etkileşim noktalarıdır.

Genelde optimizasyon problem formülasyonunda bazı özellikler amaç fonksiyon olarak seçilirken; diğer bazı özellikler, sınırlayıcılar konumuna düşürülmektedir (Url-4).

Optimizasyon diğer bir deyişle eniyileme probleminin çözümünün daha önce nasıl yapıldığına ilişkin; Karaboğa (2004) "Optimizasyon problemlerinin çözümüyle ilgili bilimsel çalışmalar oldukça eskidir ve bu çalışmalar tüm mühendislik alanlarını ilgilendirdiğinden önemli bir bilim alanı haline gelmiştir. Pozitif bilimlerin etkisiyle önceleri bu tür problemler için geliştirilen çoğu yöntemler, çeşitli kabullerle elde edilen matematiksel bağıntılara dayalı belirgin yöntemlerdir" (Karaboğa, 2004) demektedir. Klasik yöntemler olarak adlandırılan matematiksel bağıntılara dayalı bu yöntemlerin, problemlere özel olması, başka tür veya benzer problemlerin çözümünde kullanılamaması, diğer bir deyişle genel (esnek) olmaması, problemin kesinlikle matematiksel fonksiyonlarla tanımlanma gerekliliği gibi, birçok dezavantajları vardır (Karaboğa, 2004).

Günümüze kadar, özellikle son 30-40 yıldır, optimizasyon işlemleri ve süreçleri üzerine çalışan bazı bilim adamları genel amaçlı, esnek ve performansı yüksek yöntemler geliştirmek için dikkatlerini doğada var olan sistemlere ve bu sistemlerde meydana gelen olaylara yöneltmişlerdir. Sebebi; doğadaki karmaşık, zor optimizasyon problemleri yine tabiatta var olan doğal yöntemlerle çözülmektedir. Doğada var olan sistemleri ve olayları temel alarak oluşturulan optimizasyon yöntemlerine sezgisel yöntem, zeki yöntem veya yapay zeka yöntemi gibi çeşitli isimler verilmektedir. Yapay ısıl işlem algoritması, tabu araştırma algoritması, genetik algoritma, karınca koloni algoritması, yapay bağışıklık algoritması, diferansiyel gelişim algoritması, yapay arı koloni algoritması vb. gibi algoritmalar bu sınıfa giren, sık kullanılan algoritmalardır. Günümüzde bu algoritmalar oldukça farklı mühendislik sahalarında çok sayıda zor problemlerin çözümünde kullanılır hale gelmişlerdir (Karaboğa, 2004).

Aşağıdaki çizelgede (Çizelge 3.1) optimizasyon arama metotları düzenli ve hiyerarşik halde görülmektedir. Çalışma kapsamında yapılacak programın modern sezgisel tekniklerden, olasılık temelli, popülasyon tabanlı diğer bir deyişle evrimsel hesaplamalı ve alt başlıklardan olan sürü zekası temelli algoritmalardan, yapay arı kolonisi algoritması (ABC) ile tip imar yönetmeliğine uygun vaziyet planı geliştirmeye yani optimizasyon problemi çözmeye çalıştığı anlaşılmaktadır.

Çizelge 3.1: Optimizasyon yöntemlerinin sınıflandırılması (Akay, 2009' dan (Akay, 2009, ss.11) düzenlenerek alınmıştır.)

<table>
<tr><th colspan="8">Optimizasyon Metotları</th></tr>
<tr><th colspan="2">Klasik Metotlar</th><th colspan="6">Modern Sezgisel Teknikler</th></tr>
<tr><th>Analitik Metotlar</th><th>Çözüm Yapılandıran Metotlar</th><th rowspan="2">Deterministik Metotlar</th><th colspan="5">Olasılık Temelli Metotlar</th></tr>
<tr><td rowspan="2">Doğrusal Programlama</td><td rowspan="2">Dinamik Programlama</td><td rowspan="2">Tek Çözümün Geliştirildiği Metotlar</td><td colspan="4" rowspan="2">Popülasyon Tabanlı Metotlar (Evrimsel hesaplamalı yaklaşımlar)</td></tr>
<tr><td rowspan="6">Tabu Araştırma</td></tr>
<tr><td>Bölgesel Arama</td><td>Ayır ve Bağla Metodu</td><td>Isıl İşlem</td><td>Evrimsel (gelişime dayalı) Algoritmalar</td><td colspan="2">Sürü Zekası Temelli Algoritmalar</td></tr>
<tr><td rowspan="4">Türeve Dayalı Metotlar</td><td rowspan="4">Böl ve Keşfet Metodu</td><td rowspan="4">Stokastik Hill Climber</td><td>Genetik Algoritma</td><td colspan="2">Karınca Koloni Optimizasyon Algoritması (1991)</td></tr>
<tr><td>Genetik programlama</td><td colspan="2">Parçacık Sürüsü Optimizasyon Algoritması (1995)</td></tr>
<tr><td>Evrimsel programlama</td><td colspan="2">Yapay Arı Kolonisi Algoritması (2005)</td></tr>
<tr><td>Diferansiyel Gelişim Algoritması</td><td colspan="2"></td></tr>
</table>

Optimizasyon problemlerini çözmek için, bir çok algoritma bulunmaktadır. Yöntemlerin çoğu sadece belli tipteki problemler için uygundur. Dolayısıyla, uygun bir çözüm tekniğini tespit edebilmek için problemin özelliklerini tanıyabilmek önemlidir. Her bir problem grubu için hesaplama gereksinimi, yakınsama özellikleri, vb. açılardan farklılaşan değişik algoritmalar vardır.

Sezgisel Teknikler (algoritmalar) : Herhangi bir amacı gerçekleştirmek veya hedefe varmak için çeşitli alternatif hareketlerden etkili olanlara karar vermek amacı ile tanımlanan kriterler veya bilgisayar metotlarıdır. Bu algoritmalar, çözüm uzayında optimum çözüme yakınsaması ispat edilemeyen algoritmalar olarak da adlandırılır.

Bu tür algoritmalar yakınsama özelliğine sahiptir, ama kesin çözümü garanti edemezler ve sadece kesin çözüm yakınındaki bir çözümü garanti edebilirler (Karaboğa, 2004).

Optimizasyon problemlerinde sezgisel algoritmalara gerek duyulmasının sebepleri

- Optimizasyon problemi, kesin çözümü bulma işleminin tanımlanamadığı bir yapıya sahip olabilir.
- Anlaşılabilirlik açısından sezgisel algoritmalar karar verici için çok daha basit olabilir.
- Sezgisel algoritmalar, öğrenme amaçlı ve kesin çözümü bulma işleminin bir parçası olarak kullanılabilir.
- Matematik formülleriyle yapılan tanımlamalarda genellikle gerçek dünya problemlerinin en zor tarafları (hangi amaçlar ve hangi sınırlamalar kullanılmalı, hangi alternatifler test edilmeli, problem verisi nasıl toplanmalı) ihmal edilir. Model parametrelerini belirleme aşamasında kullanılan verinin hatalı olması, sezgisel yaklaşımın üretebileceği alt optimal çözümden daha büyük hatalara sebep olabilir (Karaboğa, 2004).

Optimizasyon problemlerinde Sezgisel algoritmaların değerlendirilmesi için kriterler

Çözüm kalitesi ve hesaplama zamanı: Çözüm kalitesi ve hesaplama zamanı herhangi bir algoritmanın etkinliğinin değerlendirilmesi için önemli kriterlerdir. Bundan dolayı iyi bir algoritma, ayarlanabilir parametreler setine sahip olmalı ve bu parametreler kullanıcıya önemlilik açısından hesaplama maliyeti ile çözüm kalitesi arasında bir vurgulamanın yapılabilmesine imkan vermelidir. Yani, çözüm kalitesi ve hesap zamanı arasındaki ilişki kontrol edilebilmelidir.

Kod basitliği ve gerçeklenebilirlik: Algoritma prensipleri basit olmalı ve genel olarak uygulanabilir olmalıdır. Bu durum, problem yapısı ile ilgili başlangıçta çok az bilgiye sahip olunması halinde bile algoritmanın yeni alanlara kolaylıkla uygulanabilmesini sağlar (Karaboğa, 2004).

Esneklik: Algoritmalar modelde, sınırlamalarda ve amaç fonksiyonlarında yapılacak değişiklikleri kolayca karşılayabilmelidir.

Dinçlik (Robustness): Başlangıç çözümünün seçimine bağlı olmaksızın her zaman yüksek kaliteli, kabul edilebilir çözümleri üretebilme kabiliyetine sahip olmalıdır.

Basitlik ve Analiz Edilebilirlik: Karmaşık algoritmalar, esneklik ve çözüm kalitesi açısından basit algoritmalardan daha zor analiz edilebilmektedir. Algoritma kolayca analiz edilebilir olmalıdır.

Etkileşimli Hesaplama ve Teknoloji Değişimleri: Algoritma içinde insan- makine etkileşimini kullanma fikri çoğu sistemde yaygın olarak gerçekleştirilmektedir. Herkesçe bilindiği gibi iyi bir kullanıcı ara-yüzü herhangi bir bilgisayar sistemini veya algoritmayı daha çekici yapmaktadır. Bunun en önemli avantajı çözümlerin grafiksel olarak sergilenebilmesidir(Karaboğa, 2004).

Bazı sezgisel-bölgesel araştırma algoritmaları başlangıç çözümüne bağlı olarak bölgesel optimum çözümler üretirler. Bu, genellikle iteratif gelişme metotlannda karşılaşılan bir durumdur. Bölgesel optimal çözüm küresel optimal çözümden çok uzak olabilir ve yine çoğu ayrık optimizasyon problemlerinde uygun bir başlangıç çözümünün seçimi için genel bilgiler mevcut olmayabilir. Sezgisel-bölgesel araştırma metotların dezavantajlarının bazılarını ortadan kaldırmak için basitlik ve genellik muhafaza edilmek kaydıyla aşağıda sıralanan fikirler değerlendirilebilir (Karaboğa, 2004):

- Çok sayıda başlangıç çözümleri ile algoritmanın tekrar tekrar koşulması. Ancak bu durumda rastgele farklı başlangıç çözümleriyle hesaplama işlemi oldukça maliyetlidir ve hiçbir zaman optimal çözümün bulunması garanti edilemez.
- Çok daha karmaşık komşuluk yapısının tanımlanması suretiyle çok daha iyi komşu çözümlerin üretilmesinin sağlanabilir.
- Algoritmanın koşulması esnasında bilgi toplanmasını sağlayan karmaşık öğrenme stratejilerinin kullanılması ve bu bilginin her bir koşma sonunda belirli bölgeleri veya çözümleri cezalandırmak amacıyla kullanılır.
- Bölgesel araştırma metotlarında amaç, sadece fonksiyon değerini azaltan çözümlerin kabul edilmesi olmasına rağmen, bölgesel optimallikten kaçmak için bazı gelişmeleri önleyen hareketlerin kabullenilmesi gerekmektedir. (Karaboğa, 2004).

Çalışmanın kapsamında, optimizasyon problemine ilişkin bilgi aktarılacaktır. Ayrıca, genelde optimizasyon problemi olarak düşünülen ve yapılan çalışmaya yakın bir konu olan “tesis yerleşim problemi” için kısa bir bilgi verilecektir.

Tesis Yerleşim Problemi

Tesis yerleşim problemleri araç/gereçleri, tesisleri ve kaynakları belirli bir kritere göre optimum performans sağlayacak şekilde yerleştirme kararını içermektedir. Bu gibi kararlar, araç/gereçlerin genellikle farklı ürünleri üretme esnasında kullanılmasından dolayı karmaşık hale gelmektedir. Her ürünün kendine özgü gereksinimleri olabilir ve tüm ürünler için toplam üretim maliyetinin optimum olması sağlanacak şekilde yerleşim tasarlanabilir. Yerleşim kararları hızlı ve doğru verilmelidir. Çünkü kararların zayıflığı üretim esnasında ortaya çıkmakta ve bu da artı maliyetlere yol açmaktadır. Tesis yerleşim problemleri bunun gibi bir çok zorluğu içermektedir. Bu tip problemlerin çözümünde uygun bir çözüm yöntemi olabilmektedir (Yeniay, 2001: 43, Emel, Taşkın ,2002).

3.3 Hesaplamalı Tasarım Yöntemleri

Tasarıma dönük optimizasyon teknikleri ve arama yöntemi olarak popülasyona dayalı algoritmalara ilişkin teorik bilgi verilmeden önce tasarımda bilgisayar kullanımı, gelişimi, hesaplamalı tasarım yöntemleri, vb. temel bilgiler kısaca açıklanacaktır. Konunun kapsamı gereği hesaplamalı tasarım yöntemlerinin gelişimi de bu bölüm içerisinde yer almaktadır. Diğer alt bölümlerde, optimizasyon problemi, yapay yaşam biçimleri optimizasyon problemlerinde kullanımı, sürü zekasına dayalı algoritmaların tasarım-optimizasyon problemlerinde kullanımları detaylı şekilde aktarılacaktır.

Tasarımda bilgisayar kullanımı, hesaplamalı tasarım: Bilgisayar teknolojilerinin mimarlıkta kullanımı üzerine ilk araştırmalar, 1960'lı yıllarda, mimarlığın mühendislik problemlerine benzer alt problem alanlarında çözüm üretebilmek, mimari tasarımın çizim ve sunum aşamalarında kullanabilmek amacıyla yapılmıştır. Tasarımın sistematikleştirilmesi alanında yapılan bu çalışmalar bilgisayar bilimlerindeki gelişmelerle ilerlemiştir. Bilgisayar Destekli Mimari Tasarım kavramı, 1970'li yıllarda "Computer Aided Architectural Design-CAAD" (Bilgisayar Destekli Mimari Tasarım) adlı kitabıyla William Mitchell tarafından ortaya atılmıştır (Çağdaş, 2002).

Bilgisayarla tasarım 1970'lerden günümüze kadar bir araştırma konusu olarak bilinmektedir. 1970'li yıllarda Carnegie Mellon, Ohio State gibi bazı üniversitelerde bilgisayar bilimleri ve de mimarlık bölümlerinde bir arada görev yapan Charles

Eastman gibi isimlerin öncülüğünde çalışmalar yapılmıştır. 1980'lerde UCLA'da William Mitchell, Ohio State'de Chris Yessios, Suny Buffalo'da Yehuda Kalay, University of Michigan'da Harold Borkin, Cornel Üniversitesi'nde Donald Greenberg araştırma çalışmalarını yürütmüşlerdir (Akipek, 2004). Bu yıllarda üniversitelerde geliştirilen çizim, 3-boyutlu modelleme ve mimari modelleme programları 1980'lerin ortalarından beri profesyonel hayatta kullanılmaya başlanmıştır. İlk kullanımlarda büyük şirketlerde projeler CAD operatörleri tarafından bilgisayar ortamına aktarılmıştır. Bu dönemde tasarımcılar bilgisayar ortamına aktarma sürecinin tamamen dışındadır. Macintosh bilgisayarlarının çıkması ve bilgisayar kullanabilen mimarların artmasıyla bilgisayar operatörlerinin yerini mimarlar almıştır. Bu durumda bilgisayar kullanan mimarların bilgisayar kullanamayan tasarımcılara bilgisayarla çizim konusunda aracılık ettiği bir süreçten geçilmiştir. Günümüzde hemen herkes bilgisayar aracılığıyla çizim yapmaktadır. T-cetveli ve rapido gibi çizim araçlarının yerini tamamen bilgisayarlar almıştır. Baykan daha önce kullanılan çizim araçlarının yerini tamamen bilgisayar teknolojilerinin aldığını ve bu sürecin bundan sonra da teknoloji ile hızla gelişerek daha farklı tasarım ortamlarına dönüşeceğini vurgulamıştır (Baykan, 2001).

Mimarlık alanında bilgisayar destekli tasarım, birçok farklı kullanımının yanı sıra yapı üretimi sürecinin farklı aşamalarında da kullanılmaktadır. Tasarım üretiminde; çizim, 3 boyutlu modelleme, canlandırma, yapı tasarımının sanal ortamda deprem dayanımı, aydınlatma, ısı korunumu gibi performanslarının simule edilmesi, metraj-keşif çıkarma, çizimin şantiyeye anında iletimi, işveren ile eş zamanlı toplantı yapılması gibi çağdaş olanaklar sunmaktadır. Yapı üretiminde ise; malzeme stok kontrolu, metraj, işçi takibi, iş akış programlaması, maliyet analizi, malzeme siparişi, teknik çizimlere şantiyeden kolay ulaşım, ses ve görüntülü haberleşme gibi alanlarda yoğun olarak kullanılmaktadır.

Bilgisayarın çizim, canlandırma, sunum ve benzeri işleri yaparken kullanılacak bir araç ve birçok hesaplama işlemini otomatik yapan yardımcı olmasının dışında, farklı tasarımlar yapan ve yenilikçi düşünceler oluşturmaya yardımcı üretken sistemlere dönüşmesi gerekmektedir (Çağdaş, 2004).

Tasarım alanında bilgisayar kullanımının tasarımın her aşamasında ve her yönüyle kullanılması gerekliliği konusunda Çağdaş, bilgisayar ortamının, yalnızca çizim, modelleme ve canlandırma ortamı olarak değil, aynı zamanda bir tasarım ortamı

olarak kullanılmasının tasarımcıya, tasarımlarının kapsam ve sınırlarını genişletme olanağı verebileceğini, enformasyon ve iletişim teknolojilerinin, bilgi mühendisliği aracılığıyla elde edilen ve bilgisayar ortamında temsil edilebilen yönüyle tasarımcıların tasarım bilgisini kullanarak ve uzman bilgisi katkısıyla, tasarımcı ve bilgisayarın işbirliği yapabilecekleri ortam olarak kullanılması gerekliliğini vurgulamıştır (Çağdaş, 2004).

Yakın zamanımızda ise sayısal teknolojiler; hem tasarımcıları hem de tasarım eğitimcilerini, yeni tasarım yöntemleri geliştirmeye, farklı tasarım problemlerinin çözümünde yeni buluşlar ortaya koymaya ve bu problemleri formüle etmeye zorlamaktadır.

Bu noktada hesaplamalı tasarım kavramlarına genel bir şekilde bakacak olursak;

1990'lardan günümüze kadar gelen süreçte ise teorik ve felsefi yönlerden hesaplamalı tasarım birçok araştırmacı tarafından incelenmiştir. Kalay (2004) ve Szalapaj (2001) gibi araştırmacılar CAD prensipleri, teorileri ve yöntemleri üzerinde araştırmalarını yoğunlaştırmış; Kolarevic (2003) ve Oxman (2006) gibi araştırmacılar, Hesaplamalı Tasarım yöntemlerini sınıflandırmaya, hesaplamalı tasarım yöntemlerini çözümlemeye çalışmışlardır (Turan, 2009).

Günümüzde hesaplamalı tasarım araçları mimaride sadece sunum aracı olma görevlerinden sıyrılıp, üretken araç olma işlevini yerine getirmektedirler. Farklı hesaplamalı türetici süreçlerle farklı biçim üretme, tasarlama teknikleri gelişmiştir.

Hesaplamalı tasarım teknolojilerini kullanan mimarlar, her projede problemin yaklaşımına uygun olan teknolojiyle çalışır. Önce tasarım süreci tasarlanır. Geleneksel tasarım teknikleri yardımcı araçlar olarak kullanılmaya devam etmektedir. Bilgisayar, tasarıma yaklaşımda yeni olasılıklar sunmakta, mekânsal araştırmaları desteklemektedir (Akipek, 2004).

Tasarımın erken evresindeki problemlerin çözümünde bile bilgisayar ortamının kullanıldığı, tasarım kararlarının verildiği bir ortam oluşmaya başlanmıştır. Hesaplamalı tasarım yöntemlerinde programlama dilleri kullanılarak yazılımlar geliştirilmekte; proje bazlı olarak scriptler, algoritmalar oluşturulmaktadır. Bu algoritmaların ürettiği tasarımlar yeni başlayan tasarım sürecinin devamında tasarımcılara yeni ufuklar açmaktadır.

Branko Kolarevic'e göre hesaplamalı tasarım, hesaplama esaslı biçim oluşum ve dönüşüm süreçlerinden oluşur. Hesaplamalı türetici süreç; topolojik uzay, izomorfik yüzeyler, kinetik ve dinamik, animasyon, parametrik tasarım ve genetik algoritmalardır (Kolarevic, 2000).

Kolarevic, hesaplamalı tasarım yöntemlerini,

-Topolojik Mimarlık (topological architecture)

-İzomorfik Mimarlık (isomorphic architecture)

-Animasyona Dayalı Mimarlık (animate architecture)

-Metamorfik Mimarlık (metamorphic architecture)

-Parametrik Mimarlık (parametric architecture)

-Performansa Dayalı Mimarlık (performative architecture)

-Gelişime Bağlı Algoritmalar, Evrimsel Mimarlık (evolutionary architecture)

olarak sınıflandırmıştır (Kolarevic, 2000).

Bilgisayar destekli tasarım tanımına yapay zeka çerçevesinden bakacak olursak; mimarlıkta da yapay zeka alanında gerçekleşen gelişmelerle yeni araştırma konuları oluşmaktadır. İnsanın düşünme sürecini çözümlemeye çalışan yapay zeka çalışmaları, tasarım sürecinde mimarın bunu nasıl gerçekleştirdiği ve bu sürecin bir yapay zeka sistemi ile nasıl simüle edilebileceği üzerine yoğunlaşırken diğer taraftan insanın görme, konuşma, algılama, dış dünya ile etkileşime girme gibi özelliklerini simüle etme çalışmaları "interaktif mimarlık", "enformasyon mekanı", "akıllı mekanlar" gibi yeni mimari kavramların oluşmasına neden olmuştur (Çağdaş, 2004).

Mimarlık alanında gerçekleşen bu gelişmelerin uzantısında ortaya çıkan bir diğer kavram da birçok farklı alanda görmekte olduğumuz "akıllı" olarak nitelenen sistem ve tasarımlardır. Her geçen gün günlük hayatta daha çok karşımıza çıkan bu kavramın tanımının net olarak yapıldığını söylemek çok kolay olmamakla beraber bu kavramla nitelenen sistem ve tasarımlardan kullanıcıların beklentisi çok fazla olmaktadır. Ancak burada altı çizilmesi gereken her bilgi alanının kendi "akıllı" tanımlarını yapması ve bu doğrultuda bilgi alanlarının kendi "akıllı tasarım yada sistemlerini" üretmesi gerekliliğidir. Bu bağlamda mimarlık alanında bu tanımların nasıl şekillenmekte olduğu ve buna bağlı olarak mimarlıkta tasarım biçim ve

süreçlerinin ve bunlara bağlı olarak üretim biçimlerinin sorgulanması ve araştırılması kaçınılmazdır.

Mimarlık alanındaki yapay zeka çalışmalarının genel amaçları tasarımcıyı destekleyecek, tasarım sürecini simüle edecek ve son nokta olarak tasarımcının yerini alabilecek sistemlerin yaratılabilmesidir (Url-5).

Mimari tasarlamada, biçime karar verme-uygulama süreci boyunca, her aşamada bilgisayar teknolojileri yoğun olarak kullanılmaktadır. Çalışmada, mimari kütlenin biçimlendirilmesi tasarım sürecinin önemli bir evresi olarak ele alınmıştır. Yoğunluk hesabı-büyüklüğü-yüksekliği ve kütlelerin yerleşiminde uzmanlaşmış sistemin varlığıyla tasarım sürecinin hızlanacağı ve hassaslaşacağı düşünülmektedir.

3.4 Sürü Zekası (Swarm İnteligence)

Sürü; birlikte yaşayan hayvan topluluğudur (Url-1). Yetenekleri tek başlarına sınırlı olan bireylerin-canlıların bir araya gelerek oluşturduğu topluluk anlamında kullanılmaktadır.



Şekil 3.3 : Sürü yaşam biçimlerine örnekler (Web-Anonim).

Hayvanlar niçin sürü halinde dolaşır? Sorusuna verilecek cevap sürü zekasının üstün olan taraflarını oluşturmaktadır.

1.Düşmanlardan korunmak için

- Düşman tehditlerini önceden algılamayı geliştirmek
- Düşmanlar tarafından ele geçirilme olasılığını azaltmak

2. Yemek arama başarısını geliştirmek

3. Daha iyi eş bulma şansı yakalamak

4. Enerji tüketimini azaltmak

Bu sosyal böceklerin (karıncalar, termitler, arılar…) çok uzun süredir başarılı olma sebepleri de;

- etkili ve verimli olmaları,
- esnek olmaları,
- dayanıklı-güçlü olmaları,
- birçok problem çözmeleri yemek bulmak, yuvayı savunmak,yuva inşa etmek, vb. gibi

Sürü zekası ile ilgili; Bonabeau :"… herhangi bir algoritma tasarımı girişiminde veya yayımlanmış problem çözme cihazlarını tasarımda sosyal böcek kolonilerinin ve diğer hayvan toplumların kolektif davranışlarından esinlenilmiştir.**…**" (Bonabeau ve diğ.,1999).

"Sürü davranışı" genel anlamda topluluk halinde birbirleriyle etkileşen ajanlar veya bireylerin denetimli-kontrollü davranışına işaret etmekte kullanılır. Sürünün zekası; bir fonksiyonu optimize (en iyi hale getirmeye çalışan) eden veya yerel / genel bir çevreye uyum sağlamayı amaç edinmiş birbirleriyle etkileşen bir popülasyon olarak tanımlanabilir (Karaboğa ve Akay, 2010).

sürü zekası ≅ toplu uyum

Bu benzerliğe örnek verilirse; Arı kolonilerinin kovan etrafında dolaşarak birbirlerine bilgi aktarımları, karıncaların geçtikleri yollara kimyasal madde bırakarak diğer karıncalara bilgi aktarımları, kuş sürülerinin ve balık sürülerinin konum ve hızlarını ayarlayarak ilerlemeleri sürü zekasına temel teşkil eden "zeki davranış"lardır.

Bir sürünün içinde, uzman olan bireyler tarafından aynı anda gerçekleştirilen farklı görevler vardır. Bu tür bir olaya işbölümü denir. Sürü tanımından yola çıkarak açıklanacak olursa; Sürü zekası (kolektif zeka) termitler, arılar, karıncalar, kuşlar, balık sürüleri gibi aralarında etkileşim olan böceklerin veya diğer sosyal hayvanların topluluk halindeki davranışlarını örnek alarak, problemlere çözüm getirmeyi amaçlayan bir yapay zeka tekniğidir.

Sürü zekasının temel fikri; toplu performans için büyük önem taşıyan iki temel kavram vardır:

1. Kendi kendine organize olmak;

   Kendi kendini organize etmenin dayandığı dört temel özellik

   1. pozitif geri besleme
   2. negatif geri besleme
   3. değişim, dalgalanma
   4. çoklu karşılıklı etki, etkilenme

2. İş bölümü

Bir sürüde yukarıda belirtildiği üzere iki önemli işlev vardır: a) Kendi başına organize olabilme (self-organization) b) İş bölümü. Tekrarlanan bu işlevler daha detaylı incelendiğinde; Kendi başına organize olabilme; bir sistemdeki temel birimlerin, diğer birimlerle etkileşimden aldıkları bilgileri kullanarak kendi başlarına işlev görerek sistemin bütününü etkilemeleridir. Sistemin diğer birimleri ile etkileşiminde temel komşuluk bilgilerinden faydalanılır. Yani sistemin bir bütün olarak genel başarımı ile ilgili bilgiler söz konusu değildir. Bonabeau ve arkadaşları kendi başına organize olabilmeyi pozitif geri besleme (positive feedback), negatif geri besleme (negative feedback), salınımlar (fluctuation) ve çoklu etkileşim olmak üzere dört özellik ile karakterize etmişlerdir (Bonabeau ve ark., 1999; Karaboğa, 2005, Karaboğa ve Akay, 2007)

Pozitif geri besleme, daha uygun yapıların bulunmasına sağlayan davranışlardır. Karıncalardaki kimyasal maddenin bırakılması ve diğer karıncaların bu maddeyi takibi ile daha uygun yolların bulunması, arıların dans etmeleri ile zengin nektar kaynakları hakkındaki bilgileri diğer arılara iletmeleri pozitif geri beslemeye örnek verilebilir. Negatif geri besleme ise toplanan bilgilerin kararlı hale gelebilmesi için çalışır. Tüm bireylerinin aynılaşarak topluluğun doyuma ulaşmasını engeller. Salınımlar da yeni kaynak keşiflerinin yapılabilmesi için rastgele dolaşımlar gibi düşünülebilir. Çoklu etkileşim ile de bir bireyin diğer bireye ait bilgiyi kullanabilmesi ifade edilmektedir. İş bölümü, topluluktaki bireylerin eş zamanlı olarak farklı işleri gerçekleştiriyor olmasıdır. Özelleşmiş bireylerin bir arada çalışarak gösterdikleri performans, bu şekilde bir iş bölümüne tabi olmayan

bireylerin gösterdikleri performanstan daha etkili olmaktadır ve bu özellik araştırma uzayındaki değişimlere cevap verebilmeyi sağlamaktadır.

Yapay yaşam terimi 1980'lerin sonunda Amerikalı biyolog Chirstopher Langton tarafından ortaya atılmıştır. Yapay yaşam düşü; bir bilgisayarda biyolojik yaşamınkilere benzer kurallara uygun olarak evrimleşme yetisine sahip matematik organizmaları üretmek; yaşamın maddeselliğine sahip olmak; yaşamı şekillendirmek düşüdür (Garassini, 2007).

Yapay yaşam biçimlerinde yapay beyinler, davranış yöntemleri, haberleşme metotları, problem çözme durumları gibi konularda evrimsel araştırma teknikleri kullanılmaktadır (Url-6).

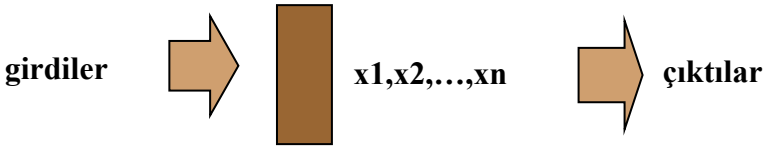
Belirli bir problemin çözümü için zeki bireylerden oluşmuş bir topluluğun kullanılması, tek bir bireyin kullanılmasına göre daha etkin bir yöntemdir (Russell ve Norvig, 2003)

Bazı sistemler ise topluluk zekâsına benzeyen sürü zekâsını particle swarm optimization (Settles, 2005) ve karınca kolonisi optimizasyonunu (ant colony optimization) Dalkılıç ve Türkmen, 2002) kullanmaktadır. Bu sistemler genel olarak birden fazla bireyin bir araya gelmesi ve bulundukları ortam içinde birbirleri ile haberleşme halinde olması ile oluşturulur. Bu gibi sistemlerde herhangi bir bireyin hareketi diğer bireylerin hareketlerini tetikleyebilir. Örnek olarak, bir canlı sürüsünü oluşturan bireylerden birinin tehlikeyi sezmesini ve bu tehlikeye karşı verdiği tepkinin sürü içinde ilerleyip tüm bireylerin tehlikeye karışı ortak bir davranış sergilemesini verebiliriz.

3.5 Yapay Arı Kolonisi (Artificial Bee Colony- ABC) Algoritması ve Yapısı

Yapay arı kolonisi algoritması tarifine, ne amaçla kullanıldığına, özelliklerine geçmeden önce "algoritma" nın ne anlamda kullanıldığını kısaca açıklamakta fayda vardır. Algoritma bir problemi çözmek için tekrar edilen tarifler kümesidir. Prosedür (uyulması gereken kurallar bütünü) olarak da düşünülebilir.

Yapay arı kolonisinin (ABC) anlatımına girilmeden daha önce kısmen aktarılan bilgileri gözden geçirmekte ve ABC özelinde genişletilmesinde yarar görülmüştür. Şekil 3.4'de basit optimizasyon işleminin temel adımları gösterilmiştir.



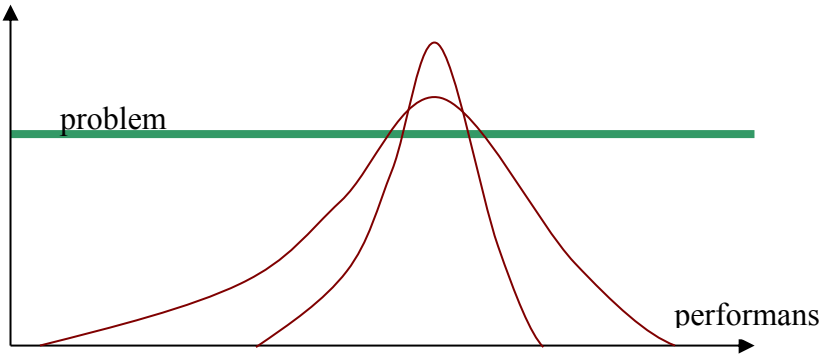
Şekil 3.4 : Basit optimizasyon işlemi (Karaboğa ve Akay, 2010 dan düzenlenerek alınmıştır).

Amaç Fonksiyon → f(x1,x2,…,xn)

Amaç Fonksiyon → (arzu edilen çıktılar-gerçek çıktılar)

- Hesaplamalı tasarım
- Nümerik tasarım
    - sınırlamasız
    - sınırlamalı

Teknik sistemler gün geçtikçe büyük ve kompleks hale gelmektedir, bu sistemler için çok fazla sayıda tasarım parametresi ve doğrusal olmama durumu söz konusu olmaktadır.



Şekil 3.5 : Sezgisel yöntemlerin diğer yöntemlerle performans karşılaştırması (Karaboğa ve Akay, 2010 dan düzenlenerek alınmıştır).

Optimizasyon problemleri için yeni yaklaşımlar kategorize edilirse;

- Doğal hesaplama (programlama)
- Hesaplamalı zeka
- Yumuşak programlama

1) Beyin benzetimli yaklaşımlar

- Sinir ağları
- Bulanık sistem

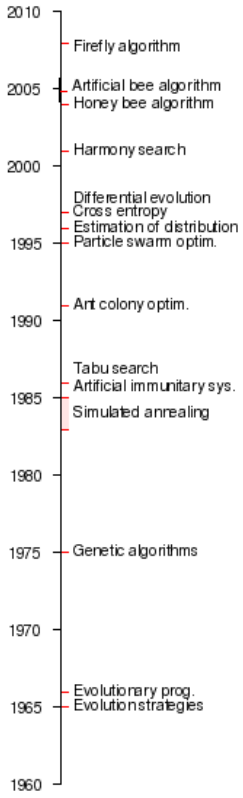
2) Evrimsel hesaplama yöntemleri

- Genetik Algoritmalar
- Genetik Programlama

- Evrimsel Stratejiler
- Evrimsel Programlama
- Sürü zekası
  * Karınca koloni Optimizasyon
  * Parçacık sürü optimizasyonu
  * Arı kolonisi Optimizasyonu

Evrimsel hesaplama modelleri ve tarihleri ise Karaboğa (2010)' a göre;

- evrimsel programlama (EP) – L. Fogel -1960.
- evrimsel stratejiler (ES) - I. I. Rechenberg-1960.
- genetik algoritmalar (GAs) – J. Holland-1975.
- genetik programlama (GP) – J. Koza -1992.
- karınca koloni optimizasyonu (ACO)- M. Dorigo-1992.
- parçacık sürü optimizasyonu (PSO) - Kennedy & Eberhart-1995.
- difarensiyel gelişim (evrimi) (DE)- R. Storn and K. Price-1995.
- barışıklık(uyum) araması (HS)-Z. W. Geem, 2001.
- yapay arı kolonisi (ABC) - D. Karaboga-2005.
- ateş böceği algoritması (FA)- ;X-S. Yang-2008.



Şekil 3.6 : Evrimsel hesaplama modelleri ve tarihleri (Karaboğa ve Akay, 2010’ dan alınmıştır).

Gerçek Arı Davranışı

Bal arılarında amaç kovan etrafında bulunan nektar kaynaklarından toplanan balı kovana getirmektir. Bu işi yerine getirmek için arılar ikiye ayrılmaktadırlar: çalışanlar ve çalışmayanlar.

Çalışanlar*:* Bir nektar kaynağından kovana nektar getiren arıları,

Çalışmayanlar (unemployed bees): Bir nektar kaynağını terk etmiş veya herhangi bir nektar kaynağına gitmemiş arıları ifade eder.

Yiyecek kaynakları, kovanın bulunduğu çevredeki nektar kaynaklarıdır. Yiyecek kaynakları bal arılarında birçok faktöre bağlıdır; yuvaya yakınlık, zenginlik, enerji gibi. Bir nektar kaynağının değerini nektar kaynağından getirilen nektarın harcanan enerjiye oranı belirler.

Çalışan toplayıcılar, belirli bir yiyecek kaynağından nektar getiren arılardır. Mevcut durumda nektar kaynağından faydalanmaya ve çalışmaya devam ederler. Bu arılar kovana kaynağın değerini ve kovan-kaynak arasındaki uzaklık bilgisini taşırlar.

Çalışmayan toplayıcılar, yararlanılacak yiyecek kaynağı ararlar. Kaşif ve seyirci olmak üzere iki tiptir. Kaşif arılar kovan etrafında yeni yiyecek kaynağı ararken kovanda bekleyen seyirci arılar çalışan arayıcılar tarafından taşınan bilgiyi beklerler. Bir kovandaki ortalama kaşif arıların sayısı uygun koşullarda %5-10 civarındadır (Karaboğa, 2005).

Arılar arasındaki bilgi değişimi ortak bilginin oluşmasında en önemli olaydır. Genellikle kovanın bazı bölümlerini ayırt etmek mümkündür. Bir kovan için en önemli bölüm bilgi değişiminin gerçekleştiği dans edilen (dancing area) alandır.

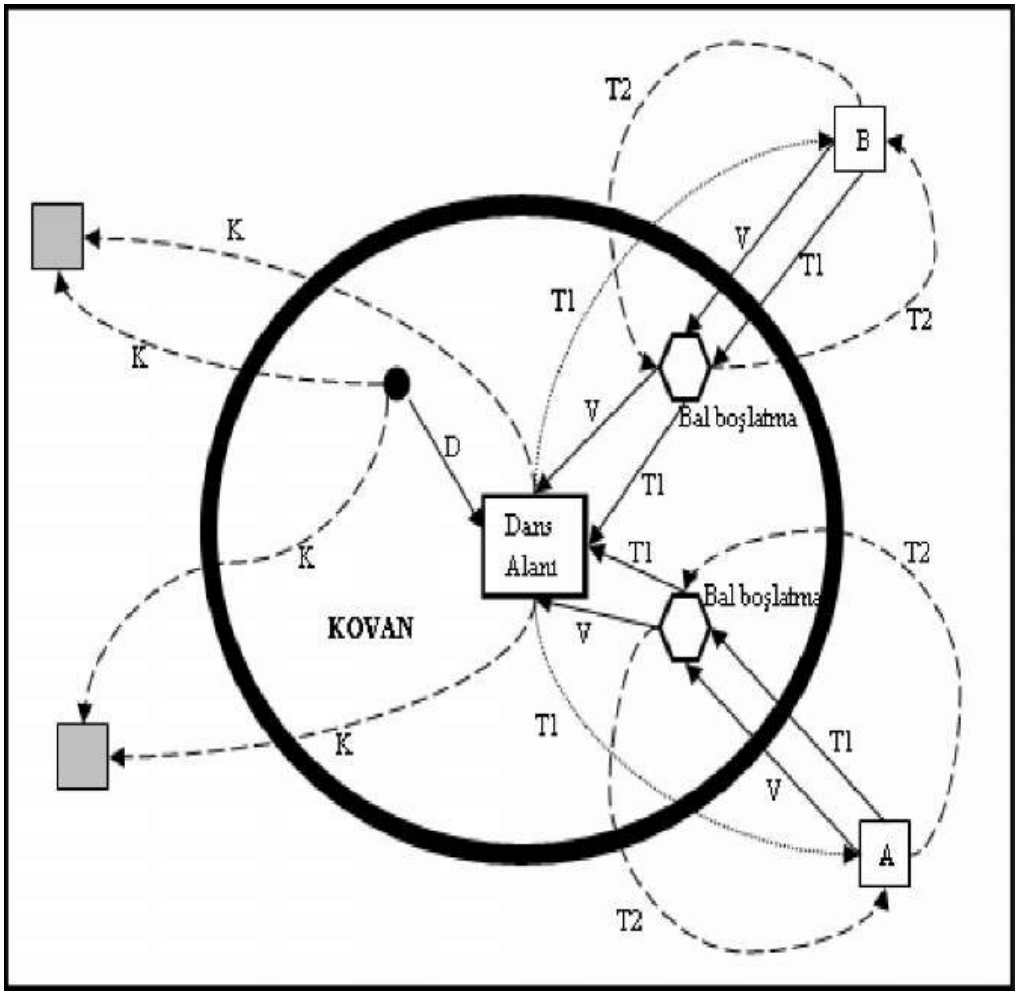
Arılar arasında bilgi değişimi dansla olur ve bu dans "dancing area"da yapılır. Bal arıları bilgi değişimi sayesinde kaliteli yeni yiyecek kaynaklarını keşfeder. Bal arılarında kaynağın yeri ve karlılığı "Waggle Dance" denilen bir dans ile diğer arılarla paylaşılır (Gruter ve Farina, 2009). Keşfedilen iki yiyecek kaynağı A ve B olsun. Başlangıçta muhtemel bir toplayıcı işsiz bir toplayıcı gibi başlayacaktır ve bu toplayıcı yuva etrafındaki herhangi bir yiyecek kaynağı bilgisine sahip değildir. (Karaboğa, 2005, Karaboğa ve Akay 2008)

Bir arı için iki seçenek vardır. Bir kaşif olabilir ve içgüdüyle veya yüzeysel ipuçlarıyla yuva etrafındaki yiyecekleri kendiliğinden aramaya başlar. (Şekil-3. 7 'deki K)

Boş olabilir ve sallanış dansını (waggle dance) izler ve yiyecek kaynağını aramaya başlar. (Şekil 3.7 'deki D)

Yiyecek kaynağı bulunduktan sonra arı kaynağın pozisyonunu hafızasına alır ve hemen nektar kaynağından yararlanmaya başlar. Bu yararlanmadan dolayı bu arı çalışan toplayıcı (employed forager) olur. Toplanan nektar kovana götürülür ve depolanır. Depolama işleminden sonra arının takip edebileceği üç yol vardır.

- Mevcut kaynağı terk edip bağımsız hale gelebilir, (Şekil 3.7 'deki V)
- Aynı kaynağa dönmeden önce bilgi paylaşımında bulunabilir (Şekil 3.8'deki T1),
- Bilgi paylaşımında bulunmadan kaynaktan nektar getirmeye devam edebilir.(Şekil 3.7 'deki T2)



Şekil 3.7 : Bal Arılarının Yem Arama Davranışı.

Son seçenek bir arı için sosyal davranış olarak görünmese de aslında kaynaktan nektar getiren yeteri kadar arı varsa geçerli bir yoldur. Sürü zekası alt bölümünde

açıklanan; Kendi kendini organize etmenin temel özellikleri arılarda aşağıdaki adımlara dayanır (Karaboğa, 2005, Karaboğa ve Akay 2008):

1. Pozitif Geribesleme (Positive Feedback): Yiyecek kaynaklarındaki nektar miktarı gözcü arıların onları ziyaret sayısı ile gelişir.
2. Negatif Geribesleme (Negative Feedback): Bir yiyecek kaynağının araştırma süreci terk edildiğinde arılar durdurulur.
3. Dalgalanmalar (Fluctuations): Kaşifler, yeni yiyecek kaynaklarının keşfi için rast gele bir arama sürecini tamamlar.
4. Çoklu Etkileşim (Multiple Interactions): Arılar dans alanında yiyecek kaynağı pozisyonları hakkındaki bilgilerini paylaşırlar.

Arı Sistemi

"Arı Sistemi" kavramı Tomoya Sato ve Masafumi Hagiwara (Karaboğa, 2004) tarafından ortaya konulmuş gerçek arıların davranışlarından esinlenerek oluşturulan bir bir sezgisel algoritmadır.

Arama süreci başında tüm arılar kovandadır ve arama süresince arılar direk olarak birbirleriyle haberleşebilirler. Her arı kendine ait bir veya daha fazla mümkün yolu oluştururken aynı zamanda genel arama sürecinin de bileşenlerini hazırlamaktadır. Arama süreci tekrarlardan oluşur. Birinci iterasyon, arılar bir veya daha fazla uygun çözümü oluşturduğuna sona erer. Bulunan en iyi çözüm birinci iterasyon esnasında kaydedilir ve ikinci çevrim başlar. İkinci çevrimde arılar adım adım çözüm yollarını oluşturur. Her çevrimin sonunda bir veya daha fazla kısmî çözüm vardır. İterasyonların toplam sayısını karar verici tayin eder.

Yapay arılar uçtuğunda ileri veya geriye geçiş işlemini gerçekleştirir. İleri geçiş esnasında arılar, geçmişten gelen kolektif deneyim ve bireysel inceleme ile çeşitli kısmi çözümler oluşturur. Bundan sonra geri geçiş işlemlerinde tüm arılar kovana döner ve karar-üretim sürecine katılır. Her arı diğer tüm arıların elde ettiği kaliteli çözümler hakkında bilgi sahibi olur. Bu yolla arılar oluşturulan kısmî çözümlerin kalitesi hakkında bilgi sahibi olurlar. Üretilen kısmi çözümlerin kalitesine dayanarak kendi kısmî çözümünü terk etmeye veya terk etmemeye karar verir. Terk ederse bağımsız bir izleyici olur, terk etmezse kovandan işçi arı almaksızın kendi çözümüne geliştirmeye devam eder veya dans ederek kendi kısmî çözümüne dönmeden önce işçi arı alır. Üretilen kısmi çözümlerin kalitesine bağlı olarak her arı kesinlikle önceki

kısmi çözüme bir sadakati vardır. İkinci ileri geçişte önceki oluşturulan kısmi çözümü geliştirmeye çalışır ve sonra tekrar geri geçiş gerçekleştirilir ve kovana dönülür. Kovanda arılar bilgi değişimi ve karar-üretim sürecine katılır, üçüncü ileri geçiş gerçekleştirilir. İterasyon bir veya daha fazla uygun çözüm oluşturulduğunda bitirilir (Lucic ve Teodorovic, 2002, Teodorovic ve ark., 2006).

ABC (artificial bee algorithm)

Yapay arı kolonisi algoritmasında, bir kolonide üç grup arı bulunmaktadır: işçi arılar, gözcü arılar ve kaşif (scout) arılar. Modelde, koloninin yarısı işçi, yarısı gözcü arı olarak seçilmiştir. Her bir nektar kaynağı için sadece bir işçi arı bulunmaktadır. Yani işçi arıların sayısı nektar kaynağı sayısına eşittir. Diğer bütün genetik algoritmalarda benzeri olduğu gibi; yapay arı algoritmasının temel adımları ise şu şekildedir (Karaboğa, 2005, Karaboğa ve Akay 2008):

Initialization

REPEAT

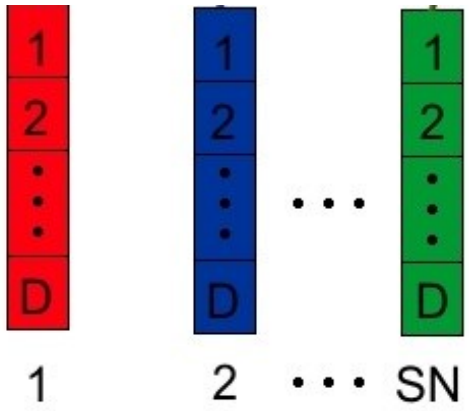
- İşçi arıları kaynaklara gönder ve nektar miktarlarını hesapla
- Gözcü arıları kaynaklara gönder ve nektar miktarlarını hesapla
- Rasgele yeni kaynaklar bulmaları için kaşif arıları gönder
- O ana kadarki en iyi kaynağı hafızada tut

UNTIL (durma kriteri sağlanana kadar): Her bir çevrim üç adımdan oluşmaktadır.

1. İşçi ve gözcü arıların kaynaklara gönderilmesi,
2. Gidilen kaynakların nektar miktarlarının hesaplanması,
3. Kaşif arının belirlenerek yeni bir kaynağa rasgele konumlanması (Karaboğa 2005, Karaboğa ve Akay, 2007).

Yukarıda aktarılan işlemler basit ve sıralı olarak aşağıda resimlerle tekrardan ele alınmıştır.

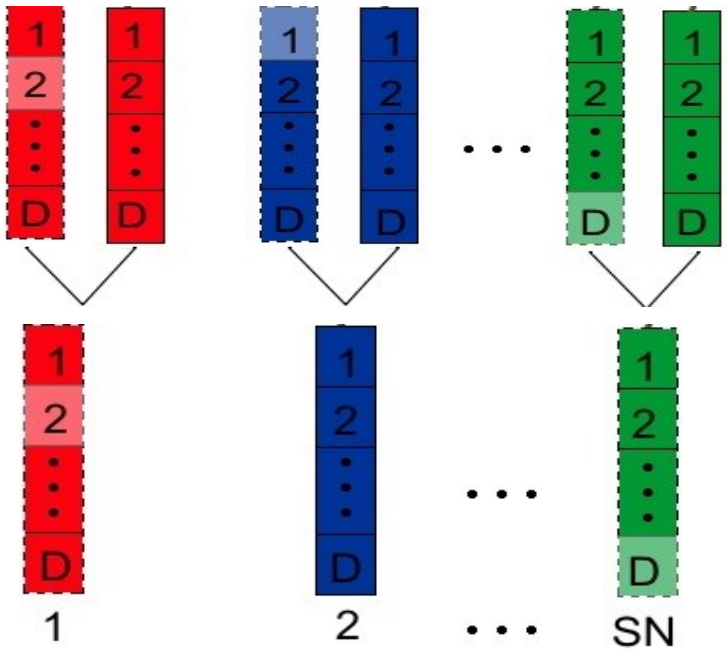
Basamak 1 : Kaşif arıları yemek kaynaklarının nektar büyüklüklerini belirlemek için yolla (Şekil 3.8):



Şekil 3.8 : ABC algoritmasında kaşif arıların gösterimi (Karaboğa ve Akay, 2010' dan düzenlenerek alınmıştır).

Programdaki karşılığı: Uygun çözümler için arama yap.

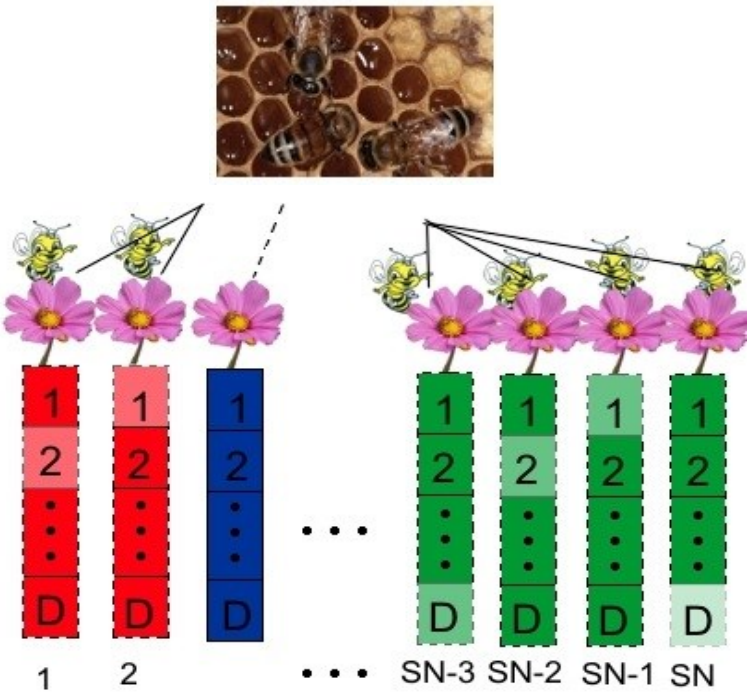
Basamak 2 : Görevli arıları yemek kaynaklarına yolla ve nektar büyüklüklerini hesapla (Şekil 3.9):



Şekil 3.9 : ABC algoritmasında görevli arıların gösterimi (Karaboğa ve Akay, 2010' dan düzenlenerek alınmıştır).

Programdaki karşılığı: Diğer uygun çözümler için arama yap ve uygun olduğu düşünülen çözümlerin büyüklüklerini değerlendir.

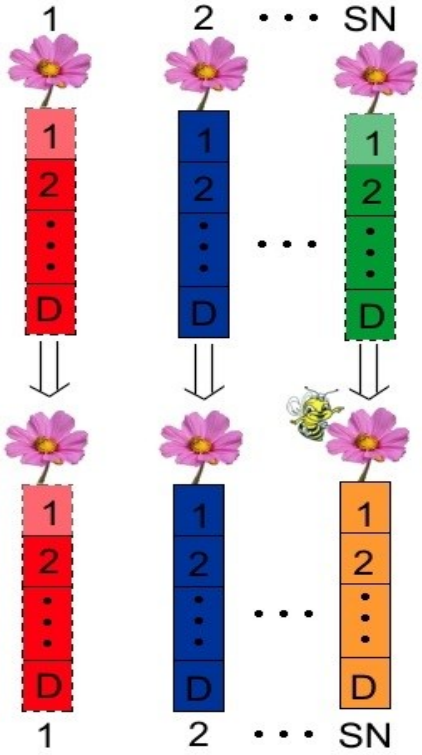
Basamak 3 : Yeni kaynaklar ile dans alanında gözcü arılar tarafından tercih edilen kaynaklar arasında olasılık hesabı yap ve gözcü arıları yolla (Şekil 3.10).



Şekil 3.10 : ABC algoritmasında görevli arıların nektar seçimi (Karaboğa, Akay, 2010' dan düzenlenerek alınmıştır).

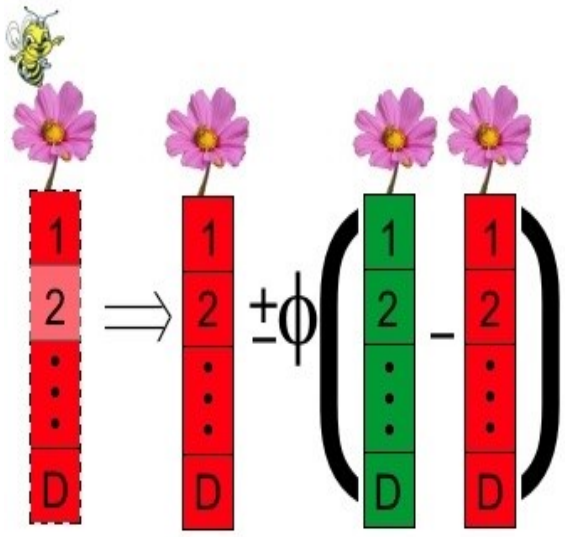
Basamak 4 : Tükenen kaynakları sömürmeyi bırak (terket).

Basamak 5 : Kaşif arıları yeni çözüm olasılıkları bulmak için arama alanına rastgele gönder (Şekil 3.11).



Şekil 3.11 : ABC algoritmasında yemek kaynağını terk etme işlemi (Karaboğa, Akay, 2010' dan düzenlenerek alınmıştır)

İşlem, gözcü arı veya görevli arının hafızasındaki komşu yemek kaynağı bulmak için kullanılır (Şekil 3.12).



Şekil 3.12 : ABC algoritmasında hafızadaki yemek kaynağı ve rastgele bulunan arasında karılaşırma (Karaboğa, Akay, 2010' dan düzenlenerek alınmıştır).

Karaboğa ve Akay, 2007' ye göre ABC Kontrol parametreleri:

- Koloni büyüklüğü (number of food sources-NS)
- Maksimum döngü sayısı (Maximum Cycle Number -MCN)
- Terketme kriteri (limit) olarak belirlenmiştir.

Diğer bir yandan, çözüm uzayında başka denetleyiciler de bulunmaktadır. Bunlar;

- Kaşif arılar keşfetme süresini control etmektedir. (*the exploration process)*
- Gözcü arılar ve görevli arılar işletme-sömürme işlemini gerçekleştirmektedir (*the exploitation process)*

Yiyecek kaynakları optimize edilmeye çalışılan problemin olası çözümlerine karşılık gelmektedir. Bir kaynağa ait nektar miktarı, o kaynakla ifade edilen çözümün kalite değerini ifade etmektedir. Gözcü arılar rulet tekerleği prensibine (Goldberg, 1989) göre gidecekleri kaynakları belirlemektedirler. Her kolonide rastgele araştırma yapan kaşif arılar bulunmaktadır. Bu arılar yiyecek ararken herhangi bir ön bilgi kullanmamakta, tamamen rastgele araştırma yapmaktadırlar. Dolayısıyla arama maliyetleri düşüktür ve de buldukları kaynağın ortalama kalite değeri düşüktür. Zengin nektar kaynağına sahip keşfedilmemiş kaynakları bulmaları da olasıdır. ABC algoritmasında işçi arılardan biri seçilerek kaşif arı haline gelmektedir. Bu seçme işlemi "limit" parametresine göre yapılmaktadır. Bir kaynağı ifade eden çözüm belli sayıdaki deneme ile geliştirilememişse bu kaynak terk edilir ve bu kaynağa gidip

gelen işçi arı kaşif arı haline gelir. Kaynağın terk edilmesi için belirlenmiş deneme sayısı "limit" parametresi ile belirlenmektedir. Gürbüz bir arama sürecinde keşif (exploration) ve keşfedilenden faydalanma (exploitation) aynı anda gerçekleşmelidir. ABC algoritmasında gözcü ve işçi arılar keşfedilen kaynaklardan faydalanma işleminde, kaşif arılar ise keşif sürecinde görev alırlar. Gerçek arılarda taşıma hızı koloninin bir kaynağı bulması ve onu kovana getirmesi ile belirlenirken, yapay arılar durumunda bulunan çözümün kalite değeri yani uygunluğu ile belirlenir. Diğer sosyal yiyecek arayıcıları gibi, arılar E/T değerini yani birim zamanda yuvaya getirilen yiyecek miktarını belirten enerji fonksiyonunu maksimize etmek için çalışırlar. Bir maksimizasyon probleminde de amaç fonksiyonunun ( ) i F θ , p i θ ∈ R , maksimize edilmesi işlemi gerçeklenir. iθ , i. kaynağın pozisyonu olmak üzere ( ) i F θ bu nektar miktarına karşılık gelir ve ( ) i E θ ile orantılıdır. c çevrim sayısı (cycle), S: kovan etrafındaki nektar kaynağı sayısı olmak üzere (Karaboğa ve Akay, 2007)(3.1);

$$P(c)\ \{\theta\ (c)i\ 1,2,\ ,S\} = i = K \qquad \textbf{(3.1)}$$

P(c) : olasılık kümesi

S : kovan etrafındaki nektar kaynağı sayısı

c : çevrim sayısı

i : kaynağın pozisyonu

K : tüm kaynakların pozisyon bilgilerini içeren nektar kaynağı popülasyonu

Daha önce belirtildiği gibi gözcü arıların bir kaynağı seçmeleri F(θ) değerine bağlı idi. Kaynağın nektar miktarı ne kadar fazla olursa, bu kaynağı bir gözcü arı tarafından seçilme olasılığı o kadar fazla olmaktadır. Yani i θ pozisyonundaki bir kaynağın seçilme olasılığı şu şekildedir (Karaboğa ve Akay, 2007)(3.2):

$$P_i = \frac{F(\theta_i)}{\sum_{k=1}^{S} F(\theta_k)} \qquad \textbf{(3.2)}$$

P_i : i . Bir nektar kaynağının seçilme olasılığı

S : işçi arıların sayısı

θ_i : işçi arıların i . pozisyonu

$F(\theta i)$: uygunluk değeri

$F(\theta k)$: diğer kaynakların uygunluk değeri

Gözcü arı, işçi arıların dansını izledikten ve formül 1 eşitliğindeki olasılık değeri ile i θ konumundaki kaynağı seçtikten sonra, bu kaynağın komşuluğunda bir kaynak belirler ve kaynağın nektarını almaya başlar. Yani i θ civarındaki kaynaklar arasında bir kıyaslama yapar. Seçilen komşuya ait pozisyon bilgisi şu şekilde hesaplanmaktadır (3.3):

$$x_{ij}(t+1) = \theta_{ij}(t) + \phi\left(\theta_{ij}(t) - \theta_{kj}(t)\right) \quad \textbf{(3.3)}$$

x i : gözcü arının pozisyonu (The position of the onlooker bee.)

t : iterasyon sayısı (The iteration number)

θ k : rastgele seçilen işçi arı (The randomly chosen employed bee.)

j : çözümün mesafesi (The dimension of the solution)

$\phi(\bullet)$: [-1,1] aralığında çeşitlilik serisi (A series of random variable in the range [-1,1])

(t) i φ , i θ civarında daha fazla nektara sahip bir kaynak bulabilmek için kullanılan, rastgele üretilen adım büyüklüğüdür. (t) i ϕ , k i'den farklı rastgele üretilen popülasyondaki bir çözüme ait indis olmak üzere (t) i θ ve (c) k θ çözümlerinin bazı bölümlerinin farkının alınması ile hesaplanır. (t + 1) i θ e ait nektar miktarı $F((t+1))$ i θ , (t) i θ konumundaki kaynağa ait nektar miktarından daha fazla ise arı kovana giderek bu bilgisini diğerleri ile paylaşır ve yeni pozisyon olarak (t + 1) i θ aklında tutar, aksi durumda (t) i θ yi hafızasında saklamaya devam eder. i θ konumundaki nektar kaynağı "limit" parametresi sayısınca gelişememiş ise i θ deki kaynak terk edilir ve o kaynağın arısı kaşif arı haline gelerek rastgele araştırma yapar, yeni bulduğu kaynak i θ ye atanır. Denklemde kaşif arıların hareketi ise (Karaboğa 2005, Tsai ve ark.,2009)(3.4);

$r \in [0,1]$ olmak üzere;

$$\theta_{ij} = \theta_{j\min} + r \cdot \left(\theta_{j\max} - \theta_{j\min}\right) \quad \textbf{(3.4)}$$

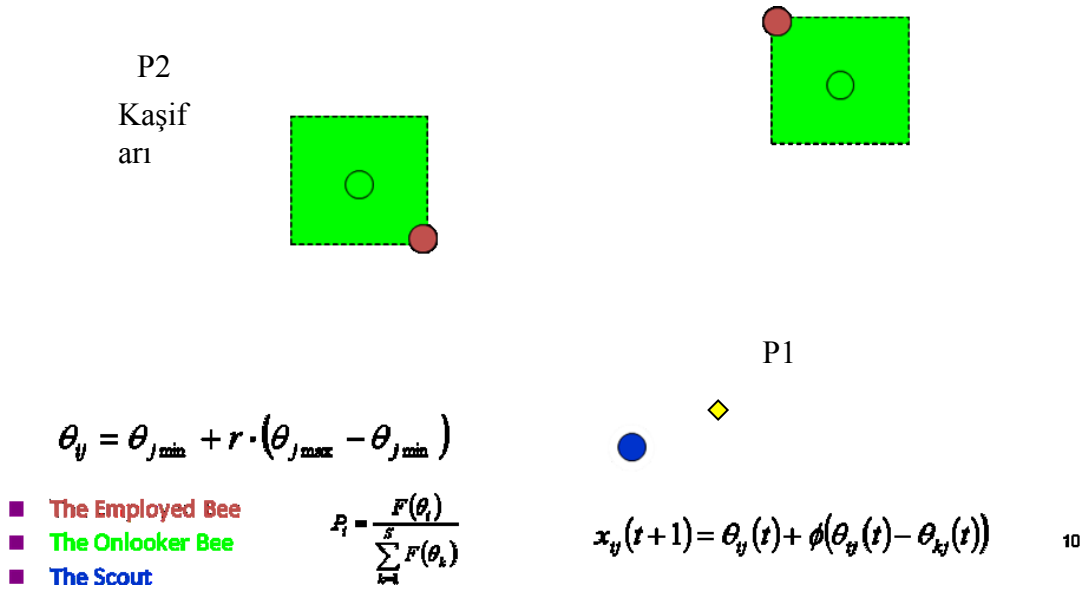
θ ij : yeni kaynağın eski kaynağa mesafesi

θ jmin : yeni kaynağın minumum mesafesi

r : rastgele bir sayı

θ jmax : yeni kaynağın maksimum mesafesi

Yukarıda aktarılan formüllerin şekilsel anlatımı Tsai ve ark. (2009) yaptıkları şemada (Şekil 3.13) açıklanmaya çalışılmıştır.



Şekil 3.13 : Yapay arı kolonisi (ABC) (Tsai ve ark.,2009).

ABC (Artifical Bee Colony)'nin Temel Özellikleri

ABC algoritması;

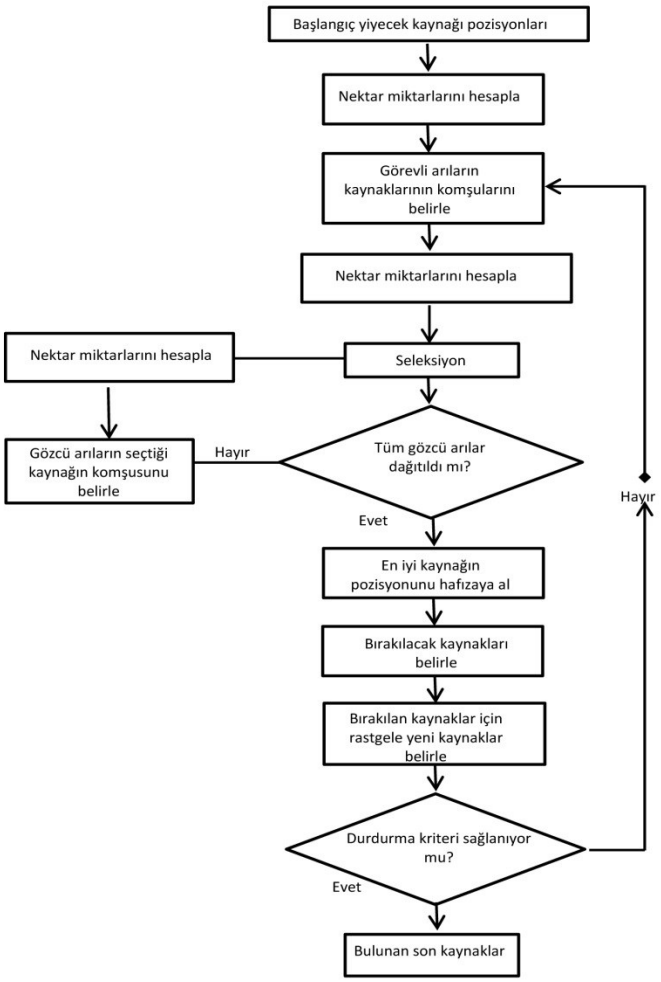
- oldukça basit ve esnektir,
- gerçek yiyecek arayıcı arıların davranışlarına oldukça yakın şekilde simüle eder,
- sürü zekasına dayalı bir algoritmadır,
- nümerik problemler için geliştirilmiştir ama ayrık problemler için de kullanılabilir,
- oldukça az kontrol parametresine sahiptir,

- kaşif arılar tarafından gerçekleştirilen küresel ve görevli ve gözcü arılar tarafından gerçekleştirilen bölgesel araştırma kabiliyetine sahiptir ve ikisi paralel yürütülmektedir (Karaboğa, 2005, Karaboğa ve Akay, 2007) .

Kullanım alanları ise;

- mekanik tasarım problemleri
- mühendislik problemleri
- dijital filtre tasarımı
- yapay sinir ağları eğitimi
- tıbbi data madeciliği
- doku sınıflaması gibi gerçek hayat problemleridir.

ABC (yapay arı kolonisi algoritması) nin temel basamakları, işleyişi diğer algoritmalara benzer şekildedir. Ara adımları ve akışı gösterir Karaboğa ve Akay' ın şeması Şekil 3.14' de sunulmuştur.



Şekil 3.14 : Yapay arı kolonisi algoritması genel işleyişi (Karaboğa ve Akay, 2007).

Sınırlamalı problemlerde kullanım yöntemleri:

- çözümlerin uygunluklarını korumalarına dayalı metotlar;
- ceza fonksiyonuna dayalı metotlar;
- uygun ve uygun olmayan çözümler arasında açık ayrım yapan metotlar ;
- farklı hibrit metotlar.

3.6 Programın Dili Olarak C# (. Net) Seçilme Nedenleri

C# dili farklı bir bakış açısıyla C ve C++ programlama dillerinin uzantısıdır. Bilgisayarları programlamak için programlama dillerine ihtiyaç duyulur.Bu dillerden en çok bilinenleri Basic, C, C++, Pascal,Java ve Assembler 'dır. Bazı programlama dilleri derleyicilere ihtiyaç duymasına karşın bazıları ise yorumlayıcılara ihtiyaç duyarlar, mesela bir C++ programını çalıştırabilmek için C++ derleyicisine ihtiyaç duyulurken, Perl ile yazılmış bir CGI scripti için komut yorumlayıcısına ihtiyaç vardır. Derleyiciler programı çalıştırmadan önce kodları makina komutlarına çevirirler fakat yorumlayıcılar bir grup kodu satır satır ya da bloklar halinde yorumlayarak çalıştırırlar. Sonuç itibariyle; derleyiciler de, komut yorumlayıcıları da birer bilgisayar programlarıdır. C ve C++ dilleri bir giriş bekleyen ve çıkış veren birer bilgisayar programları olarak düşünülebilir. Giriş olarak kaynak kodu veren bu programlar çıkış olarak ise makina kodu üretirler. C ve C++ dillerinin ortaya çıkışına kısaca bakılırsa; C dili en çok kullanılan yapısal programlama dillerinden biridir. C dili Dennis Ritchie tarafından, Martin Richards ve Ken Thompson tarafından geliştirilen BCBL ve B dillerinin temelleri üzerine kuruldu. C dili "The C Programming Language by Brian Kernighan and Dennis Ritchie" kitabıyla büyümüştür (Kernighan, Ritchie, 1978, Stroustrup B.,1994). 1983 yılında ANSI standartlar komitesi C standartları için toplanmıştır. Bu standartlaşma süreci tam 6 yıl sürmüştür. Ve tabi ki şu anki standartların oluşumuna katkıda bulunan ANSI 99 standartları da diğer önemli bir gelişmedir (Stroustrup B.,1994). C programlama dili, programcılar tarafından herhangi bir tür program geliştirmek için yazılmış genel amaçlı bir dildir. C ile bir düşük seviyeli sistem için program yazabileceği gibi, yüksek seviyeli bir GUI (Grafik Arabirimi) tasarımını oluşturmak da mümkündür. Günümüz programcıları çeşitli amaçlar için programlarını geliştirirken C dili ile yazılmış kaynak kodlarını kullanırlar. Bjarne Stroustrup (1980) 1980 yıllında C++ dilini ortaya çıkarmıştır. C++ dili C temelli ve C nin bir üst kümesi olarak

düşünülebilir (Stroustrup ,1994). C++ en popüler nesne temelli programlama dilidir. C++ dilinin ilk ismi "C with Classes" şeklindedir (Stroustrup, 1980b). C++ dili C diline nazaran daha etkili ve güçlüdür. Ve en önemli özellği ise C 'den farklı olarak nesne temelli bir dildir. Şu anda C++ dili ANSI ve ISO kuruluşları tarafından standartlaştırılmıştır. Bu standartların son versiyonu 1997 yılında yayınlanmıştır.

C ve C++ dillerinden türetilmiş C# programlama dili; güçlü, modern, nesne tabanlı ve aynı zaman type-safe (tip-güvenli) bir programlama dilidir. Aynı zamanda C#, C++ dilinin güçlülüğünü ve Visual Basic' in ise kolaylığını sağlar. C# dilinin çıkması Java dilinin çıkmasından bu yana programlama dilleri adına yapılan büyük gelişmelerden biridir. C#, C++ 'ın gücünden, Visual Basic (ara yüz) 'in kolaylığından ve Java programlama dilinin özelliklerinden faydalanarak tasarlanmış bir dildir. Fakat, Delphi ve C++ Builder 'daki bazı özellikler C# programlama dilinde de bulunmaktadır.. Ama Delphi ya da C++ Builder hiçbir zaman Visual C++ ya da Visual Basic 'in kullanım kolaylığını yakalayamamıştır. C# aynı ortamda ister alt seviyede ister yüksek seviyede program geliştirebilmeye olanak sağlamaktadır. C# dili Microsoft tarafından geliştirilen ".NET" platformunun en temel ve resmi dili olarak lanse edilmiştir. C# dili Turbo Pascal derleyicisini ve Delphi 'yi oluşturan Anders Heljsberg ve Microsoft'da Visual J++ takımında çalışan Scott Wiltamuth tarafından geliştirilmiştir. .NET framework'ünde bulunan CLR (Common Language Runtime), JVM (Java Virtual Machine)' ye, garbage collection, güvenilirlik ve JIT (Just in Time Compilation) bakımından çok benzerlikler taşır. CLR, ".NET" Framework yapısının servis sağlama ve çalışma zamanının kod organizasyonu yapan ortamıdır. CLR, ECMA standartlarını destekler (Url-7). Kısaca C# kullanmak için ".CLR" ve ".NET" Framework sınıf kütüphanesine ihtiyaç bulunmaktadır. Bu da, C# programlama dili, JAVA, VB ya da C++ programlama dillerinden değildir. C, C++ ve JAVA 'nın iyi özelliklerini barındıran yeni bir programlama dilidir. Sonuç olarak; C# ile kod yazmak avantajlı ve daha kolaydır (Url-8).

Dot net (.NET)

Dotnet bir platformdur. ".NET" ile Microsoft, Microsoft Intermediate Language (MSIL) standardını tanımlayarak tüm programlama dilleri ve derleyici firmaları tarafından kaynak kodun standart bir Ara Kod'a çevrimini sağlayacak alt yapıyı kurmuştur (Url-9). Böylece temel olarak iki hedef başarılmış olmuştur. İlki, birbirinden farklı programlama dilleri kullanarak tip uyuşmazlığı gibi problemlerin

olmadığı bir zeminde standart ara kodun oluşturulması, ikincisi, standart yapıda olan ara kodun optimizasyonunun ve hedef dile çevriminin tek noktadan yapılabilmesidir (Url-9). Bütün bu işlemlerin sağladığı olanaklar ise; Daha güvenli işletim sistemi, dil bağımsızlığı, donanım bağımsızlığı olarak özetlenebilir. ".NET" ile gerek derleyici firmaları, gerek programcıların temel alışkanlıkları radikal olarak değişmektedir. Buna en güzel örnek kullanılan dilin ve derleyicinin önemini büyük ölçüde yitirmiş olmasıdır. Dolayısıyla ister C#, ister şu ana kadar desteklenen programlama dillerinden herhangibiri ile uygulamalar geliştirmiş olsun, öncekine göre kullanılan derleyici ve dilin etkisi çok daha az olacaktır (Url-10).

4. YAZILIMIN ÖZELLİKLERİ VE KULLANIMI (PROGRAMIN ÇALIŞMA SİSTEMİ)

Tasarım süreci kompleks bir süreçtir. Tasarım sürecinin de, diğer kompleks sistemler[1]de olduğu gibi, çözümlenemeyecek kadar çok girdisi olduğu açıktır. Günümüzde kompleks sistemler üzerine yapılan çözümleme çalışmaları algoritmik düşünce baz alınarak yürütülmüştür. Yazılım üretmek veya bir yazılımı manipüle edebilmek için, sadece yazılım dilini bilmek yeterli değildir. Bilgisayarın çalışma mantığını bilmek gerekmektedir. Bu fikir algoritmik düşüncedir. Algoritmayı anlamak, sayısal tasarım için kritik önem taşımaktadır. Tasarım problemini optimize etme, çözüm alternatiflerini bulma tamamen algoritmik düşüncenin ürünüdür. Bu sebeple tasarım kavramı ve kompleks sistemlerin çözümünü öngören algoritmik altyapı, algoritma geliştirme ve sisteminin üzerinde durulacaktır. Bu anlamda; yapılan çalışmada daha önceki bölümde aktarılan teorik alt yapı üzerine pratiği geliştirmek diğer bir deyişle tip imar yönetmeliklerini temel alan yazılımın üretilmesi süreci, program mantığı, işleyişi bu bölümün konularıdır.

Tip imar yönetmeliği uygulaması üzerine hazırlanan, diğer bir deyişle optimizasyon işlemini gerçekleştiren algoritma, ABC algoritmasını kullanan sürü algoritmasıdır. Sonuç itibariyle evrimsel algoritma kullanan yapay zekadır.

Programa ilişkin bilgiler : Program optimizasyon işlemi yapmaktadır. Optimizasyon işlemlerinde amaç fonksiyonun matematiksel bağıntılar ile tanımlı olması gerekir.

Programın amaç fonksiyonu (4.1);

$$|TA| - |ta| \geq 0 \qquad \textbf{(4.1)}$$

TA : Emsal (KAKS) ile parsel alanının çarpımına eşit olan ideal çözüm (alan)

ta : bulunan en yakın sonuç (komşuluk) (alan)

---

[1] Kompleks sistemler; karmaşık anlamına *gelmemektedir.* Belli bir sistem içerisinde, birçok girdi ve çıktı alternatifi olan sistem anlamındadır.

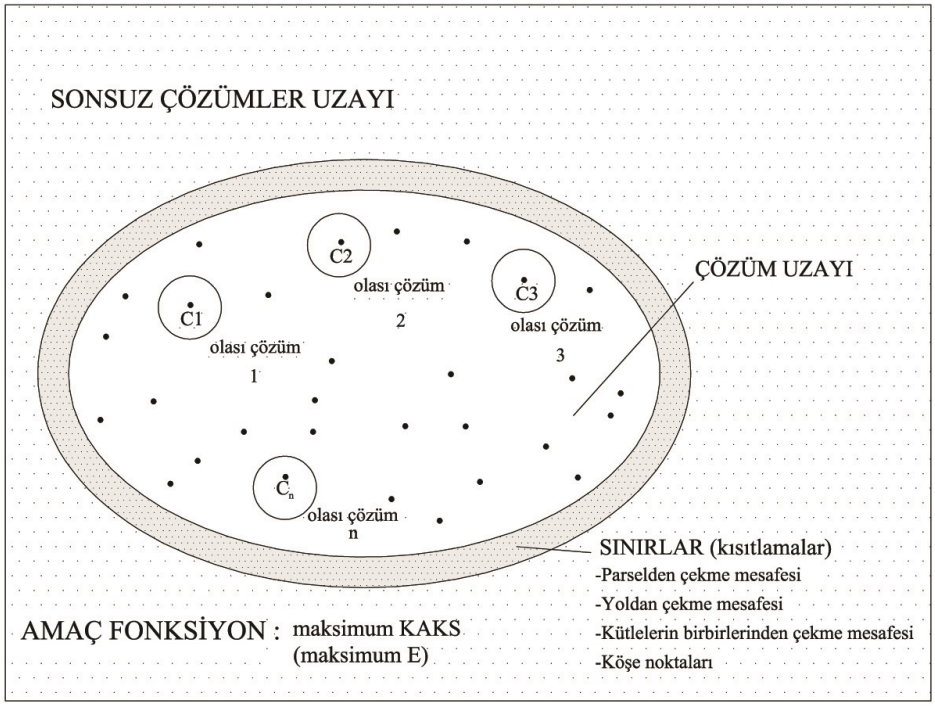
Formülden de anlaşılacağı üzere yazılım için amaç fonksiyon; "hatayı", diğer bir deyişle "tam doğru çözümle bulunan çözümler arasındaki farkı" minimize etmektir.

D tasarım parametreleri (D- design parameters) yani arama uzayı boyutu; kütle için; 1. En, 2. Boy, 3. Açı, 4. Pozisyon (x, y), 5. Kat sayısı, 6. Hesaba dahil edilecek kat sayısı, 7. Oturduğu kot, 8. Kütle merkez piksel (x, y) olarak belirlenmiştir.

Uzayı sınırlamayı sağlayan kısıtlamalar ise;

1. Parsel sınırları, 2. Komşu parselden çekme mesafesi, 3. Yoldan çekme mesafesi, 4. Kütlelerin birbirinden çekme mesafesidir.

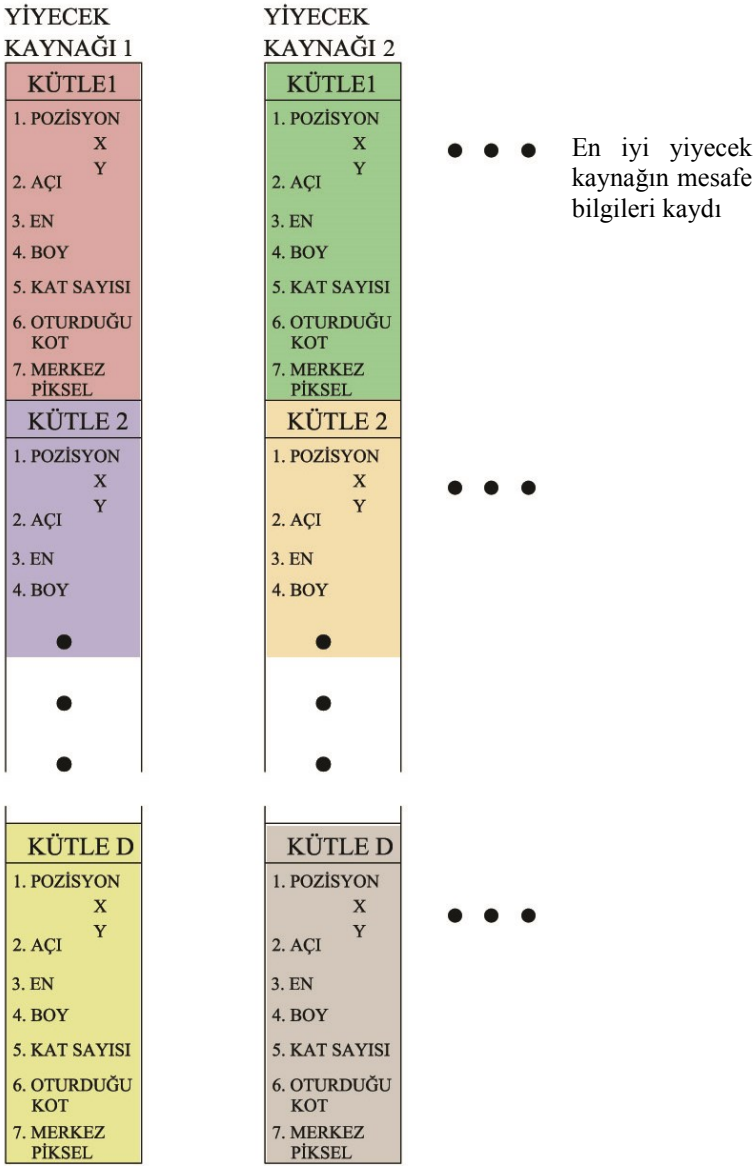
Şekil 4.1 de bu optimizasyon işleminin kavramsal çizimi yer almaktadır.



Şekil 4.1: Olası çözüm kümeleri – arama/çözüm uzayı.

Şekil 4.1 de C1, C2, …, Cn ile ifade edilen "olası çözüm" kümeleri D tasarım parametreleri (design parameters)-arama uzay boyutu büyüklüğünde kümelerdir. ABC algoritması çözüm uzayı, D tasarım parametreleri içerisinde uygun çözümler için arama yapan algoritmadır. Bu yüzden arama ve optimizasyon işleminin tüm adımları ABC algoritması çalışma mantığına ve kurallarına uygun yürümektedir.

ABC içerisinde, tasarım parametrelerinin diğer bir deyişle kütle bilgilerinin oluşturulması işlemi (blok bilgilerine ait bilgilerinin şematik gösterimi) şekil 4.2 de gösterilmiştir.



Şekil 4.2 : Bir bloğa ait bilgilerin arı olarak kodlanmış gösterimi.

Yiyecek kaynağı 1

Bloğun kat sayısı (adet)

Bloğun en ölçüsü (n.)

Bloğun boyu (m.) → Bloğun taban alanı (en x boy)

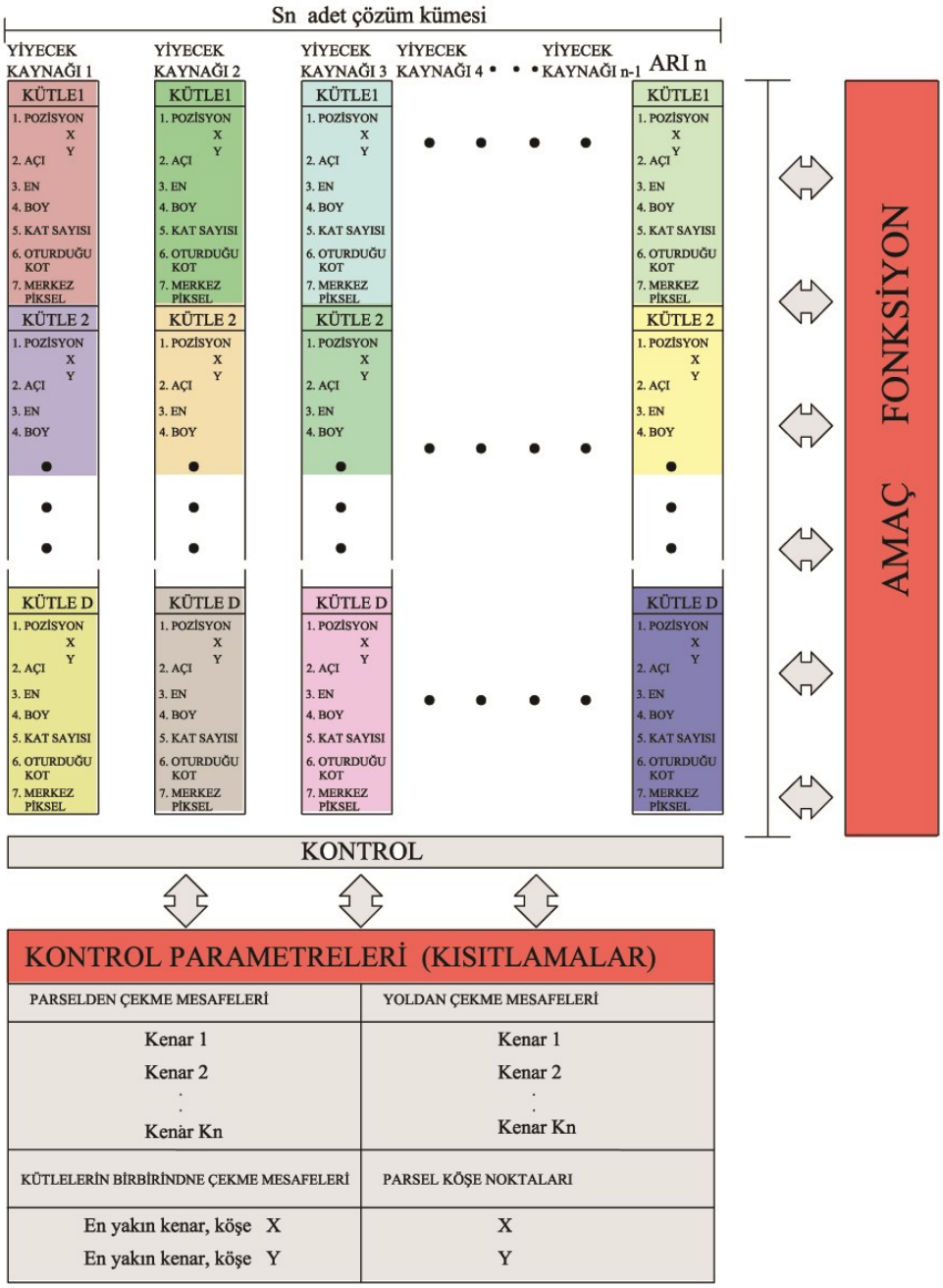
Bloğun pozisyonu (x,y)

Bloğun oturduğu kot (z)

Bloğun x koordinatıyla yaptığı açı

Merkez piksel (x, y)

Diğer bloklara ilişkin parametreler yiyecek kaynağı (kütle 1) in arkasına eklenir. n blok sayısı kadar çözüm "string"i (çözüm kümesi) oluşur. String içerisinde yer alan çözümler peş peşe parsel içerisine yerleştirilecek toplam blok sayısı kadardır.



Şekil 4.3 : Tip imar yönetmeliğine uygun çözümler üretebilen yazılım için çözüm kümesi (string) şematik gösterimi.

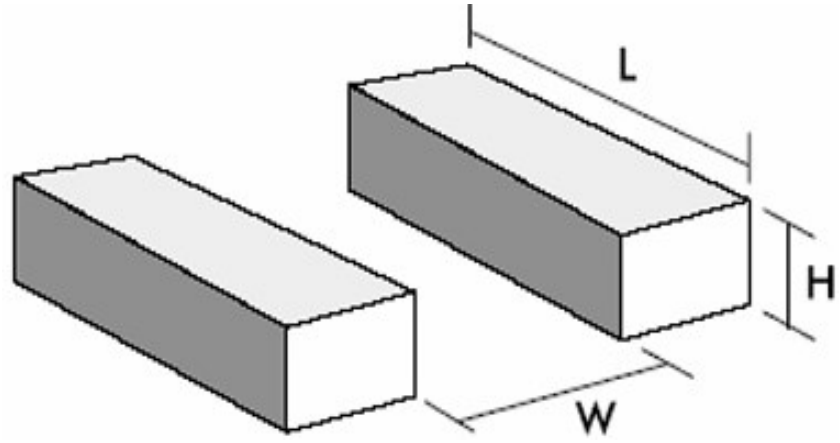
Problem için kullanılan ABC algoritmasının sağlıklı çalışması için, parsel içerisine yerleştirilmesi gereken binaların kütle ölçüleri bilgileri ile birlikte koloni büyüklüğü, limit, çevrim sayısı, MR, Epsilon, SSP gibi temel algoritma ayarlarını da programa girmek gerekmektedir.

Bunun dışında diğer bilgiler için detaylı bilgiler aşağıda verilmiştir.

a. Arsa-parsel parametreleri

1. parsel eni (pw)
2. parsel boyu (pl)
3. parsel alanı (pw X pl))
4. parsel kenarları

1. kenar komşusu= yol-parsel
2. kenar komşusu= yol-parsel …
n. kenar komşusu= yol-parsel



Şekil 4.4 : Binalara ilişkin basit parametreler (Pearlmutter, 1998).

b. Bina parametreleri

1. bina pozisyonu (x,y) (parsel büyüklüğü ile değişebilir)
2. kat sayısı $(2 \leq k \leq 15)$
3. kat alanı (m2) = mw X ml--- kat sayısı X kat alanı = toplam inşaat alanı
4. bina eni (mw)
5. bina boyu (ml)
6. açı (derece)

c. kurallara ilişkin parametreler

mesafeler

a. binalar arası minimum mesafe (m)

b. binaların parsel kenarlarına uzaklığı

Bina biçimi ile ilgili veriler

İhtiyaca cevap verebilecek yazılımın, çalışma kapsamına ve amacına uygun olabilmesi için; iki tür bilgi kümesine ihtiyaç duyacağı öngörülmüştür.

- Vaziyet planı olarak belirlenen alanı tanıtmaya-tanımlamaya (düzlemi oluşturmaya) yönelik girdiler.
- Binaya (yapılara) yönelik girdiler

Kullanıcıdan oluşturulacak düzlem için, vaziyet planı alternatifleri üretilecek alanın imar planındaki genişlik-uzunluk, çalışılacak alanın-parselin şekli, imar planı tipi(kentlere göre), emsal ve maksimum kütle yüksekliği gibi özellikleri istenir. Bu verilerle beraber bina büyüklüğü için uygun olan büyüklüklerdir.

Parsele ait bilgiler (emsal, hmax, vb.) arayüzde bulunan ikonlar ve kutucuklar sayesinde programa girilir. Yaklaşık yapı büyüklükleri, yapılar arasındaki düzene ilişkin bilgiler (ayrık, bitişik, blok, vb.) girildikten sonra programın istenilen özelliklere yapı yaklaşma sınırı içersinde (yapı yapılabilecek alan-yollar, yan parselden olan uzaklıklara bağlı olarak) bir başka deyişle kısıtlamalara bağlı olarak üreteceği alternatif kütleler 2 boyutlu, vaziyet planı şeklinde görülebilecektir.

Kullanılacak (binaların kaplayacağı alan) kapladığı büyüklüğü saptamak için emsal ve hmax sayısal büyüklüklerinin girilmesi gerekmektedir. Doluluk oranı (yoğunluk- binaların vaziyet planı üzerinde kaplayacağı alanların toplam alana oranı -TAKS) her kattaki yüzey alanı ile çarpılarak bulunan sonuç KAKS ve/veya emsali verecektir. Bina formu ve alternatifleri ilişkin tanımlar ve girdiler çalışmanın ileri safhalarında değerlendirilecektir.

Tanımlamalar

Yapılara (kütle) ilişkin tanımlar:

Kütlelerin boyutları

- Yükseklik (300cm X kat adedi)
- En
- Boy

Parsele ilişkin tanımlar:

- Parselin şekli, boyutu, yapı yapılabilecek alan boyutları (çekme mesafeleri)
- Parselin çevresinde bulunanlara ilişkin bilgiler; komşu parsel, yol, vb.
- Emsal
- H max

Yapıların yükseklikleri-İki kütle arasındaki mesafe: Kütleler arasında kalması

gereken mesafe k= (h1+h2) /2 + 7 formülüyle bulunabilir.

Yapıların yükseklikleri-en-boy ilişkileri: Emsale bağlı olarak; kütle hacmi aynı kalmak durumundadır. Dolayısıyla yapı en ve boy ölçüleri küçüldükçe yükseklik artacak, yapıların en ve boy büyüklükleri arttıkça yapı yüksekliği azalacaktır. Kısaca programın, imar yönetmeliklerine uygun, maksimum yapı yoğunluğunda, kütle biçimi ve yükseklikleri için sunacağı alternatiflerin hacimleri birbirine eşit olmak durumundadır. Bu çalışmanın-yazılımın hacim (en-boy-yükseklik) tabanlı çalışabileceğinin bir göstergesi olabilir. Kısıtlamalar ve kurallar, kütlelere ilişkin kurallar aşağıda daha kapsamlı bir biçimde verilecektir.

Parsele ve kütle sayısına ilişkin kurallar, ise sadeleştirilmiş bir biçimde şu şekildedir.

-Parsel büyüklükleri : Konut alanlarında min. 6 metredir. $6\ m. \leq P \leq 900\ m.$

-Kütlelerin komşu parselden ve yoldan uzaklığı: Kütlelerin sınırlara olan uzaklıkları (bahçe mesafeleri):

-Ön ve yol kenarına rastlayan yan bahçe mesafeleri minimum 5 metredir. Ancak mevcut düzende bu mesafeler 5 metreden fazla ise bu düzene uyulur.

-Bitişik olmayan yan cephelerde minimum 3 m. mesafe bırakılması şarttır.

-Ön bahçesiz bitişik veya blok olarak yapılacak binalarda blok boyu 40 metreyi aşamaz.

-Arka bahçe mesafesi bina yüksekliğinden az olamaz.

Parsel içerisine yerleşecek kütle sayısı ve büyüklüğü: Parsel içerisine yerleşecek kütlelerin sayısı imar planları yoğunluk hesaplarından (emsal) çıkacaktır. Yapıda kullanılacak toplam alan kütle büyüklüğünü ve sayısını belirleyecektir. Mimari projede ihtiyaç programından bulunan birim (daire büyülüğü) alanı da kütle sayısını belirleyecek diğer bir faktördür. Türkiye genelinde, ekonomik sebeplerle, imar planında yazılan yoğunluk hesabı sonucu çıkan alanın tümü kullanılmaya çalışılır. Bu sebeple, yazılım maksimum emsal (yoğunluk) hesabına başka bir deyişle maksimum alan kullanımına göre çalışacaktır.

Tip imar yönetmeliği kısıtlama olarak yazılacak program içerisinde yer alacağı için; yoğunluğu, yapı düzeni, yerleşimini ve kütle formunu etkileyebilecek bilgilerin tamamı 3194 sayılı imar kanunu (03/05/1985 (kabul tarihi) ve değişiklikleriyle) ve mevzuatlarından alınmıştır.

İşlemler :

-Parsele ilişkin işlemler:

-Parselin köşe koordinatlarına göre tamamı çizilecek

-Parsele komşu yollar ve parseller belirlenecek

-Eğim yönü gösterilecek

-Parsel kotu belirlenecek (giriş kotu)

-En üst kot ile en alt kot arasındaki yükseklik belirtilecek

-Kütleye ilişkin işlemler:

-KAKS, TAKS, Emsal, hmax seçilecek

-Kütle büyüklükleri belirtilecek ve kütle büyüklükleri seçilecek (en-boy, her katta bulunan daire sayısı, standart kütle büyüklükleri için seçim yapılacak)

-Kat sayısı belirtilecek

-Kat yüksekliği 300 cm standart.

-Kütleler arası organizasyona ilişkin işlemler:

Çekme mesafeleri

Kenar yoldan çekme mesafesi

Komşu parselden çekme mesafeleri

-Genel yerleşime ilişkin işlemler:

Kısıtlamalar :

-Parsele ilişkin kısıtlamalar:

1000 m2 <Parsel alanı < 200.000 m2

2 < Parsel köşe sayısı < 19

20 m <Parsel kenar uzunluğu< 999 m

%0 ≤ Eğim < %30 (yaklaşık)

Eğim yönü tek yön

0 m ≤ En üst kot ile en alt kot arasındaki fark < 24 m

-Kütleye ilişkin kısıtlamalar:

0,4 ≤ KAKS ≤ 4

0,4 ≤ TAKS ≤ 1

0,4 ≤ Emsal ≤ 4

6 m ≤ hmax ≤ 50 m.

Kütle büyüklükleri

6 m ≤ en ≤ 20 m, 10 m ≤ boy ≤ 50 m

2 ≤ Kat sayısı ≤ 15

3 m ≤ Kat yüksekliği ≤ 4 m

3 m ≤ zemin kat yüksekliği ≤ 6 m

-Kütleler arası organizasyona ilişkin kısıtlamalar:

-Genel yerleşime ilişkin kısıtlamalar:

Kurallar :

-Parsele ilişkin kurallar:

Yolun genişliğine bağlı kaldırım genişliği (tablodan-matristen oku)

Örn: 15 m. lik yolda kaldırım genişliği (tek taraf) 2 m.

30 m. lik yolda kaldırım genişliği (tek taraf) 4 m.

-Kütleye ilişkin kurallar:

Bitişik düzen için; Kütle ölçüleri (n) ;

Boy; 50 m >nboy> 6 m (tek kütle min.cephe uzunluğu)

Derinlik; 40 m > n en > 10 m

Blok düzen için; Kütle ölçüleri (n) ;

Boy ; 50 m > n boy Derinlik ; 40 m > n en > 10 m

Ayrık düzen için; Kütle ölçüleri (n) ;

Boy ; 30 m > n boy > 6 m Derinlik ; 40 m > n en > 10 m

-Kütleye ilişkin kurallar:

Kütle yükseklikleri yol genişlikleriyle değişmektedir. (ön bahçe, yeşil alan, meydan, yol genişleme ve daralmaları, vb. unsurlar yol genişliğine dahil değildir.)

Çizelge 4.1 : Yola bağlı hmax ve kat adetleri özet tablosu.

	YOL GENİŞLİĞİ	Hmax	KAT ADEDİ
yol-hmax değişim tablosu	7.00 m.	6.50 m.	2
	7.00-9.50 m.	9.50 m.	3
	9.50-12.00 m.	12.50 m.	4
	12.00-15.50 m.	15.50 m.	5
	15.50 m.den geniş	18.50 m.	6
	17.00 m.den geniş	21.50 m.	7
	19.50 m. den geniş	24.50 m.	8

Kütleler arası organizasyona ilişkin kurallar:

Kütlelerin aralarındaki mesafe k: (h1+h2)/2 +7

Kütlelerin yoldan mesafesi min 500 cm olmak zorundadır.

Kütlelerin komşu parselden mesafesi min. 300 cm olmak zorundadır. Her kat için 50 cm eklenecektir.

Daha sonra aşamalarda üzerinde çalışılması düşünülen genel yerleşime ilişkin kurallar:

- Kütlelerin büyük yüzeylerinin manzara yönüne bakacak şekilde yerleştirilmesi.
- Kütlelerin birbirlerini kapatmayacak şekilde yerleştirilmesi.
- Büyük yüzeylerinin (kütlelerin boyu) istenilen yöne bakacak şekilde yerleştirilmesi.
- Kütlelerin dar kenarlarının eğim çizgilerine dik olacak şekilde yerleştirilmesi.

Diğer Kurallar:

Kütlelerin hacimleri hmax ve E (emsal) e göre ayarlandığında aynı kalmak durumundadır.

Toplam V (hacim) : Sabit (Yazılım maksimum emsal (yoğunluk) hesabına başka bir deyişle maksimum alan kullanımına göre çalışacaktır.) Toplam V; Yoğunluğu (emsal) hmax belli bir parsel için her zaman aynı kalacaktır. Kat alanı arttıkça kat sayısı azalacak, kat alanı azaldıkça kat sayısı artacaktır.

A (bir kütle alanı) : kütle eni (i) X kütle boyu (l), H: kat adedi X300 cm (sabit), Toplam A : kat sayısı X kütle sayısı = bütün kat alanları toplamı, Toplam A = E (emsal) X arsa alanı, Hmax: maksimum kütle yüksekliği (imar planına bağlı), E(emsal) : m2 ye dahil kullanılabilecek kat alanları toplamı / parsel alanı

Çizelge 4.2 : İmar planlarında parselle ilgili kuralların bulunduğu bilgilendirme kutusu.

Arka bahçe derinliği	Yan bahçe derinliği	Ön bahçe derinliği
Kat sayısı	Yapı düzeni	Min. Parsel cephesi
Min. Parsel alanı	KAKS	TAKS

Çizelge 4.3 : Temel parametreler özet tablosu.

KONUT BÖLGELERİNDE **"YAPI DÜZENİ"**	**Yapı boyutu (metre)**		**Bahçe mesafesi (metre)**		
	Cephe uzunluğu	Cephe derinliği	ön	yan	arka
Bitişikdüzen (B) 1-4 kat 5-9 kat 10+ kat	min 6 min 9 min 12	10-40	olmayabilir	yok	H/2 (min 3)
Blokdüzen (BL) 1-4 kat 5-9 kat 10+ kat	max. 50	10-40	min 5	Blok başında min 3 + her bir kat için 0,5	H/2 (min 3)
Ayrık düzen (A) 1-4 kat 5-9 kat 10+ kat	6-30 9-30 12-30	10-40	min 5	Blok başında min 3 + her bir kat için 0,5 m.	H/2 (min 3)
Yapılanma koşulu					
TAKS/KAKS	Min TAKS = 0,40		En az 5	Yukardaki A, BL koşulları geçerli	Yukarıdaki koşullar geçerli
E (emsal)	Min TAKS = 0,40		En az 5		

4.1 Yazılımın Amacı

Geliştirilen yazılım yukardada anlatıldığı üzere; arsa üzerinde tip imar planı ile ilgili tüm bilgilerin Taks, Kaks, Emsal, Hmax, kütle, en, boy, yükseklik, kat adedi, parsel kenarın çekme mesafesi, komşu binalardan çekme mesafesi gibi matematiksel bağıntılar içeren optimizasyon problemini çözmek amacındadır.

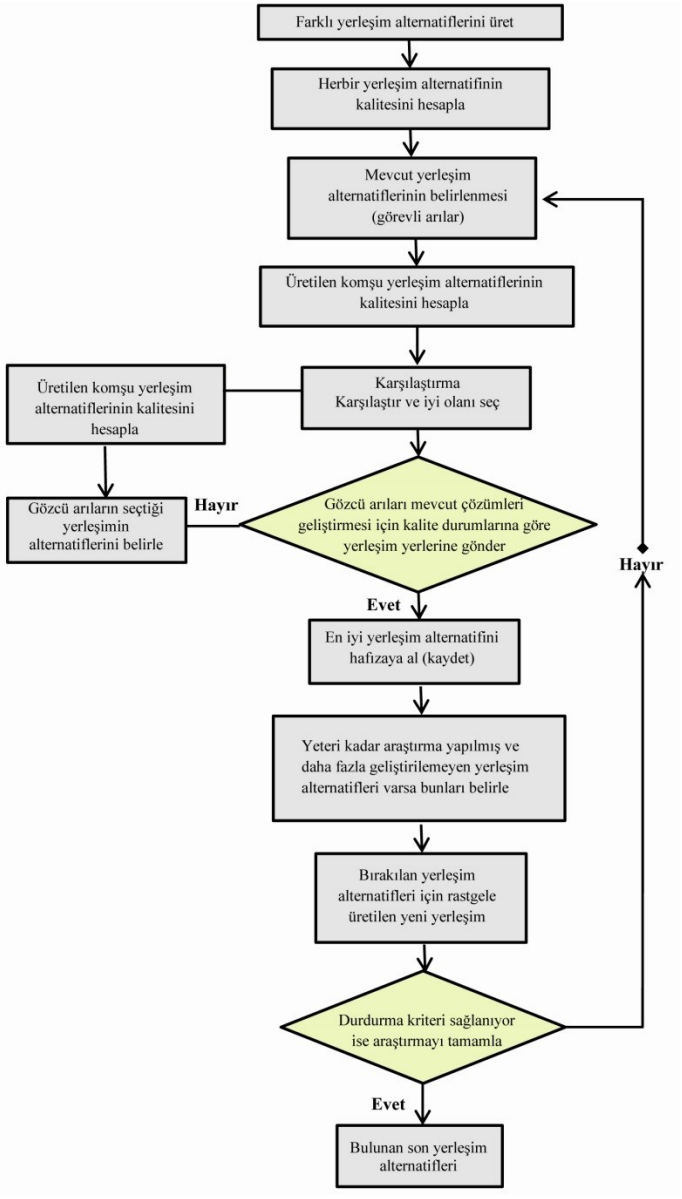
4.2.Kurallar, Kısıtlamalar, İşlemler

Kuralların tamamı yukarıda anlatıldığı üzere TİY (tip imar yönetmeliğin) den birebir alınmıştır.

Çalışma kapsamında oluşturulan yazılım ABC algoritmasına benzer çalışma adımları takip ederek TİY'e uygun vaziyet planları üretir. Program için temel adımlar;

- Farklı yerleşim alternatiflerini üret
- Herbir yerleşim alternatifinin kalitesini hesapla
- Mevcut yerleşim alternatiflerinin belirlenmesi (görevli arılar)

- Üretilen komşu yerleşim alternatiflerinin kalitesini hesapla
- Karşılaştırma- karşılaştır ve iyi olanı seç
- Gözcü arıları mevcut çözümleri geliştirmesi için kalite durumlarına göre yerleşim yerlerine gönder
- En iyi yerleşim alternatifini hafızaya al (kaydet)
- Yeteri kadar araştırma yapılmış ve daha fazla geliştirilemeyen yerleşim alternatifleri varsa bunları belirle
- Bırakılan yerleşim alternatifleri için rastgele üretilen yeni yerleşim alternatifleri belirle
- Durdurma kriteri sağlanıyor ise araştırmayı tamamla; şeklindedir.



Şekil 4.5 : Yazılım temel adımlar (akış şeması).

Bu temel adımlar içerisinde kütle boyutları, parsel bilgileri, vb. kısıtlamaların-kuralların ve temel ABC algoritması çalışma sistemine (mantığına) tasarım probleminin uygun hale getirilmesi için yapılan kodlamalar C# programa diliyle, Microsoft Visual Studio 2010 , ".NET" ortamında kodlanmıştır.

Arsaya ilişkin bilgilerin algoritma içerisinde nasıl kullanıldığı ve yapılan işlemler ve kütleye ilişkin bilgilerin kodlanmasına ilişkin bilgiler aşağıda verilmiştir.

Programda kullanılan kısıtlamaların program içerisinde nasıl yer aldığına, arı algoritmasının nasıl çalıştığına vb. işlemlere dair yazılan program kodlarına bir bakarsak;

C#, Microsoft Visual Studio 2010 , ".NET" çalışma mantığında belli görevleri yaptırmak üzere class (sınıf)' lar oluşturulur. Bu farklı classlar içinden elenerek çıkarılan program kodlarından bazı parçaları çalışmanın anlaşılabilmesi mümkün olacaktır. Kütle özelliklerinin bulunduğu "class kütle" içinde yer alan kısıtlamalar;

```
namespace konutyerlestirme
{
   public class kutle:ICloneable
   {
      public int id = 0;
      public PointF konum = new PointF();
      public double en = 0;
      public double boy = 0;
      public int katSayisi = 0;
      public int gKatSayisi = 0;
      public int aci = 0;
      public double kot = 0;
      public PointF merkezpiksel = new PointF();
      public PointF[] pt=new PointF[4];
      public void ptGuncele()
```

Parsel içine yerleştirilecek bir kütleye ait merkez piksel belirleme işlemi aşağıdaki gibidir;

```
 //merkez pikselin belirlenme işlemi
         PointF[] pt0 = new PointF[4];
         PointF p1 = new PointF(this.konum.X, this.konum.Y);
         pt0[0] = new PointF(p1.X, p1.Y);
         PointF p2 = new PointF(Convert.ToInt32(this.konum.X + this.en), this.konum.Y);
         pt0[1] = new PointF(p2.X, p2.Y);
         PointF p3 = new PointF(Convert.ToInt32(this.konum.X + this.en), Convert.ToInt32(this.konum.Y +
this.boy));
         pt0[2] = new PointF(p3.X, p3.Y);
         PointF p4 = new PointF(this.konum.X, Convert.ToInt32(this.konum.Y + this.boy));
         pt0[3] = new PointF(p4.X, p4.Y);
         this.pt = pt0;
```

Aşağıda bir kütleye ilişkin kısıtlamalar verilmiştir. "kütle işlemleri" sınıfı (class) ile tanımlanmıştır. Her kütle için bir "id", ekranda görülen kütlenin numarası, en, boy,

açı, konum x, konum y, katsayısı, görünen katsayısı, oturduğu kot, kütle merkezi (merkez piksel) x konumu, kütle merkezi (merkez piksel) y konumu değişken olarak program içinde kodlanmıştır.

```
    public object Clone()
    {
      kutle s = new kutle();
      s.id = this.id;
      s.en = this.en;
      s.boy = this.boy;
      s.aci = this.aci;
      s.konum.X = this.konum.X;
      s.konum.Y = this.konum.Y;
      s.katSayisi = this.katSayisi;
      s.gKatSayisi = this.gKatSayisi;
      s.kot = this.kot;
      s.merkezpiksel.X = this.merkezpiksel.X;
      s.merkezpiksel.Y = this.merkezpiksel.Y;
      for (int i = 0; i < pt.Length; i++)
      {
        if (this.pt[i] != null)
        {
          PointF pfs = new PointF(this.pt[i].X,this.pt[i].Y);
          s.pt[i] = pfs;
        }
      }
      return s;
    }
  }
```

Abc algoritması “ABC genel” class ‘ı içinde kodlanmıştır.

Program yazımında kullanılan ABC algoritması içerisine yerleştirilen “tip imar yönetmeliği” ne uygun başlangıç atamasına ve ABC çalışma mantığına örnek kod;

```
//ABC genel
      ari[] arilar1 = new ari[kaynakS];//ari sınıfından arilar1 adında nesne dizisi tanımlama işlemi
      for (int k = 0; k < kaynakS; k++)
      {
        ari ari1 = (ari)metotlar.baslagicAtamasiYap(parselP,rn, xKKisitlariMn, xKKisitlariMx,
yKKisitlariMn, yKKisitlariMx, bEnKisitlariMn, bEnKisitlariMx, bBoyKisitlariMn, bBoyKisitlariMx,
bAciKisitlari, bKatKisitlariMn, bKatKisitlariMx).Clone();
        double g1k = metotlar.g1KisitiHesapla(ari1, PKomsuBilgileri,parselP);
        double h1k = metotlar.h1KisitiHesapla(ari1);
        double tkk = metotlar.tkKisitiHesapla(ari1, TA, TTabA,epsilon);
        ari1.violationHesapla(h1k, g1k,tkk);//abc fonksiyonu
        if (ari1.violation == 0)
        {
          ari1.fitnessHesapla(TA, TTabA,merkezDKot,kot1,kot2,cS);
        }
          arilar1[k] = (ari)ari1.Clone();
        }
      best1 = (ari)metotlar.enIyiAriBul(arilar1).Clone();//arilar1 kolonisi icindeki en iyi ariyi
bulduk(fitness degerine göre)
      for (int c = 0; c < iterasyon; c++)
            {
```

İşçi arılar (employed bees) için metodlar:

```
//employed bees
```

```
            for (int p = 0; p < kaynakS; p++)
            {
               ari komsu = (ari)arilar1[p].Clone();
               komsu.ariGonder(arilar1,paralel,parselP, MR, kaynakS, xKKisitlariMx, xKKisitlariMn,
yKKisitlariMx, yKKisitlariMn, bKatKisitlariMn, bKatKisitlariMx, bEnKisitlariMx, bEnKisitlariMn,
bBoyKisitlariMx, bBoyKisitlariMn, p, rn);
               //Deb in kuralına göre seçim yap
               double g1k = metotlar.g1KisitiHesapla(komsu, PKomsuBilgileri, parselP);
               double h1k = metotlar.h1KisitiHesapla(komsu);
               double tkk = metotlar.tkKisitiHesapla(komsu, TA, TTabA,epsilon);
               komsu.violationHesapla(h1k, g1k, tkk);

               if (arilar1[p].violation > 0 && komsu.violation > 0)
               {
                  if (komsu.violation < arilar1[p].violation)
                  {
                     arilar1[p] = (ari)komsu.Clone();
                     arilar1[p].gelisim = 0;
                  }
               }
               else if (arilar1[p].violation <= 0 && komsu.violation <= 0
               {
                  komsu.fitnessHesapla(TA, TTabA,merkezDKot,kot1,kot2,cS);
                  if (komsu.fitness < arilar1[p].fitness)
                  {
                     arilar1[p] = (ari)komsu.Clone();
                     arilar1[p].gelisim = 0;
                  }
               }
               else if (arilar1[p].violation > 0 && komsu.violation <= 0)
               {
                  komsu.fitnessHesapla(TA, TTabA, merkezDKot, kot1, kot2, cS);
                  arilar1[p] = (ari)komsu.Clone();
                  arilar1[p].gelisim = 0;
               }
               else
               {
                  arilar1[p].gelisim++;
               }
            }
```

Gözcü arılar (Onlooker bees) için ihtimal hesabı :

```
//onlooker lar için ihtimal hesabı yap

            double tViolaiton = 0;
            double tFitness = 0;
            for (int z = 0; z < arilar1.Length; z++)
            {
               if (arilar1[z].violation > 0)
               {
                  tViolaiton += arilar1[z].violation;
               }
               else
               {
                  tFitness += arilar1[z].fitness;
               }
            }
            for (int z = 0; z < arilar1.Length; z++)
            {
              if (arilar1[z].violation > 0)
              {
              arilar1[z].normfit=(1-arilar1[z].violation/tViolaiton)* 0.5;
               }
```

```
            else
            {
            arilar1[z].normfit=0.5+(arilar1[z].fitness/tFitness)* 0.5;
            }}
```

Gözcü arılar için;

```
//onlooker bees

            int t = 0;
            int m = 0;
            while (t < arilar1.Length)
            {
               if (rn.NextDouble() < arilar1[m].normfit)          {
                  t++;
                  ari onlooker = (ari)arilar1[m].Clone();
                  onlooker.ariGonder(arilar1,paralel,parselP, MR, kaynakS, xKKisitlariMx, xKKisitlariMn,
yKKisitlariMx, yKKisitlariMn, bKatKisitlariMn, bKatKisitlariMx, bEnKisitlariMx, bEnKisitlariMn,
bBoyKisitlariMx, bBoyKisitlariMn, m, rn);
                  //Deb in kuralına göre seçim yap
                  double g1k = metotlar.g1KisitiHesapla(onlooker, PKomsuBilgileri, parselP);
                  double h1k = metotlar.h1KisitiHesapla(onlooker);
                  double tkk = metotlar.tkKisitiHesapla(onlooker, TA, TTabA,epsilon);
                  onlooker.violationHesapla(h1k, g1k, tkk);

                  if (arilar1[m].violation > 0 && onlooker.violation> 0)
                  {
                     if (onlooker.violation < arilar1[m].violation)
                     {
                        arilar1[m] = (ari)onlooker.Clone();
                        arilar1[m].gelisim = 0;              }   }
                  else if (arilar1[m].violation <= 0 && onlooker.violation <= 0)
                  {
                     onlooker.fitnessHesapla(TA, TTabA, merkezDKot, kot1, kot2, cS);
                     if (onlooker.fitness < arilar1[m].fitness)
                     {arilar1[m] = (ari)onlooker.Clone();
                        arilar1[m].gelisim = 0;
                     }                  }
                  else if (arilar1[m].violation > 0 && onlooker.violation <= 0)
                  {
                     onlooker.fitnessHesapla(TA, TTabA, merkezDKot, kot1, kot2, cS);
                     arilar1[m] = (ari)onlooker.Clone();
                     arilar1[m].gelisim = 0;          }                else                    {
                     arilar1[m].gelisim++;
                  }
               }
               m++;
               if (m == (kaynakS))
               { m = 0; }          }
```

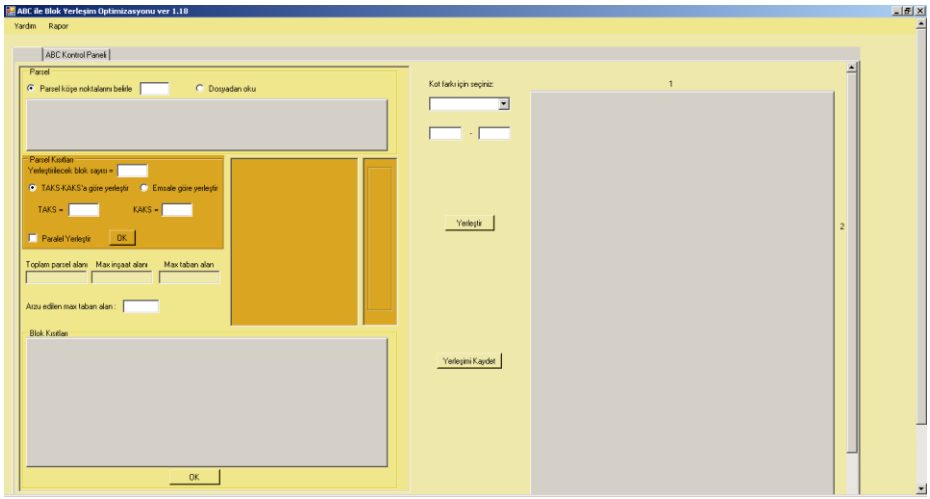
4.3 Programın Arayüzü ve Kullanımı

Oluşturulan bilgisayar programının arayüzü ve kullanımı için temel bilgiler bu bölümde anlatılacaktır. Arayüz temelde, 3 farklı panel yardımı ile oluşturulmuştur. Bunlar;

1. Temel parametreler
2. Rapor bilgileri paneli

3. ABC kontrol paneli

1. Program-temel parametreler: TİY ‘e ilişkin, parsel, kütle, eğim, yerseçimi, büyüklük, sayı, kat adedi vb. ayarların bulunduğu temel kısımdır. Program çalıştırıldığında aşağıdaki gibi (şekil 4.6) bir arayüzle karışılaşılacaktır.

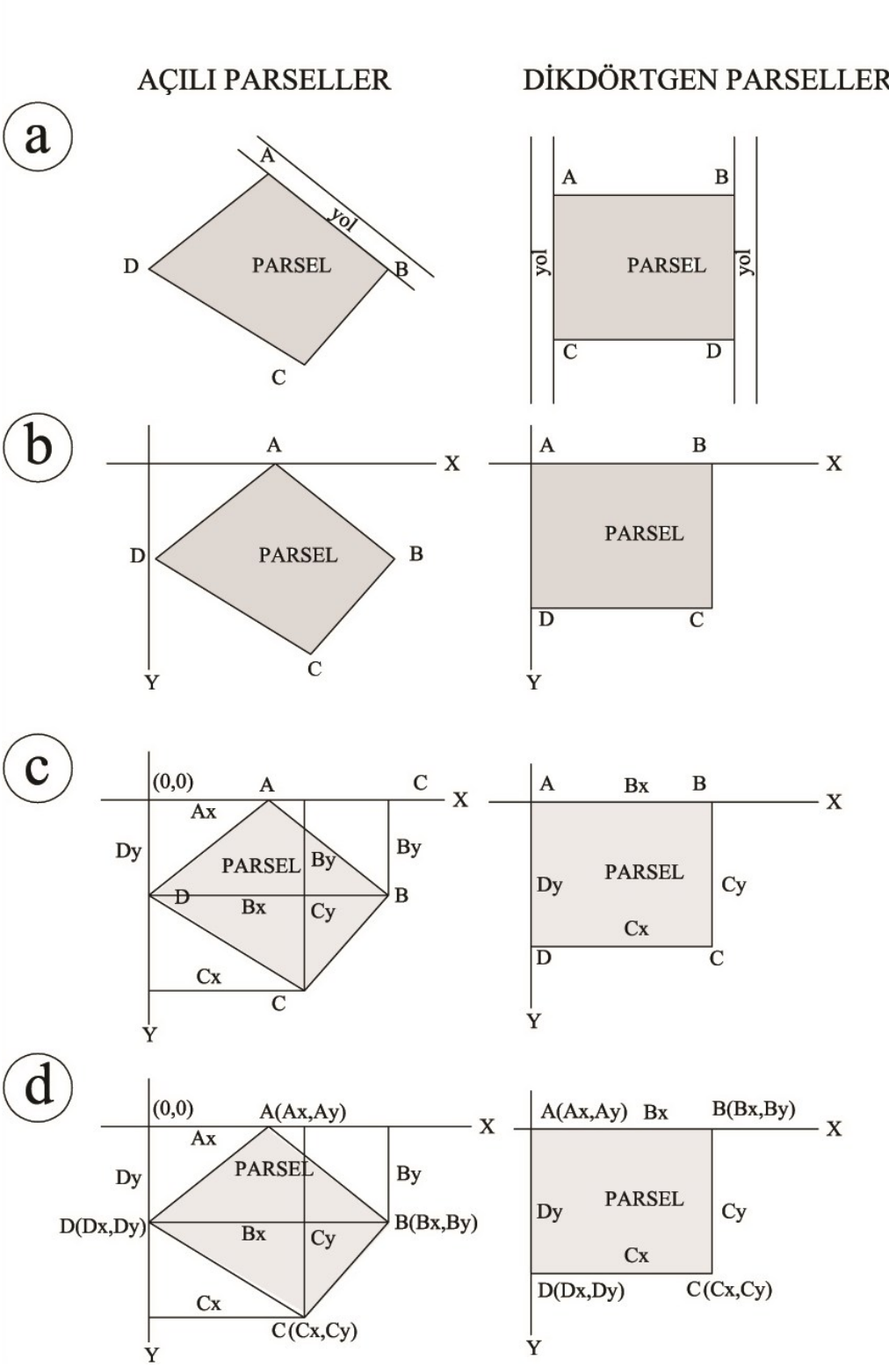


Şekil 4.6 : Program arayüzü.

Temel parametrelerden ilki, parsele ilişkin bilgilerin girildiği yada dosyadan (.txt) okultulduğu kısımdır. Aşağıda kartezyen koordinat sistemine uygun (vektörel) parsel köşe noktalarının girilmesi işlemi sırayla aktarılmıştır. Parsel köşe sayısı maksimum 19 olarak belirlenmiştir. Bu sınırın belirlenmesinde, Türkiye’deki genelde imar parsellerinin köşe sayıları baz alınarak ortalama bir değer olarak seçilmiştir. Parsel köşe sayısının belirlenmesi işleminden sonra, belirlenen köşe sayısı kadar koordinat bilgilerinin girilmesi gerekmektedir.

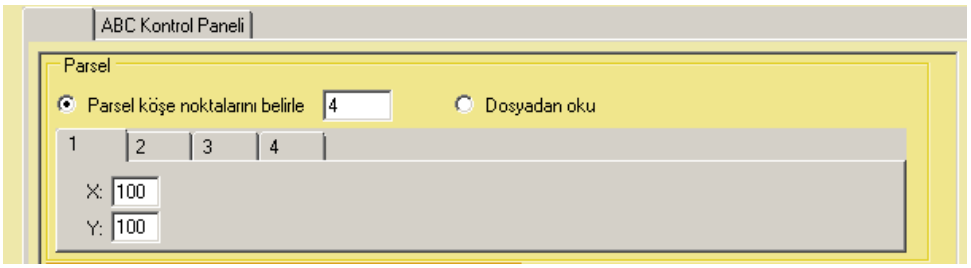
Köşe koordinat noktaları belli parseller için ve/veya köşe sayısı çok olan parseller için köşe koordinatlarının text (metin) belgesi, “.txt” dosya uzantılı belgeye yazılması işlemi aşağıda gösterilmiştir. Koordinatlar sırası ile girildiğinde yazılım bu koordinat belgesinde bulunan (x,y) noktalara göre parseli oluşturacak ve bu parsele ilişkin işlemleri yapacaktır. Koordinat belgeleri “dosyadan oku” seçeneği ile bulunduğu konumdan programa yüklenecektir.

Parsel köşe noktaları koordinatlarının programda kullanılabilir text (.txt) dosyasına dönüştürülme işlemi aşağıda şekillerle gösterilmiştir (4.7).

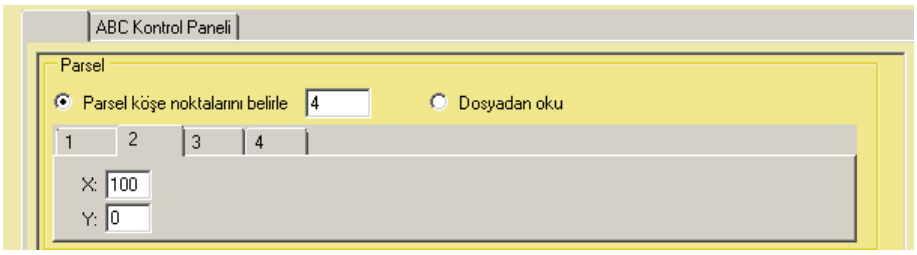


Şekil 4.7 : Parsel koordinatlarını belirleme işlemi adımları.

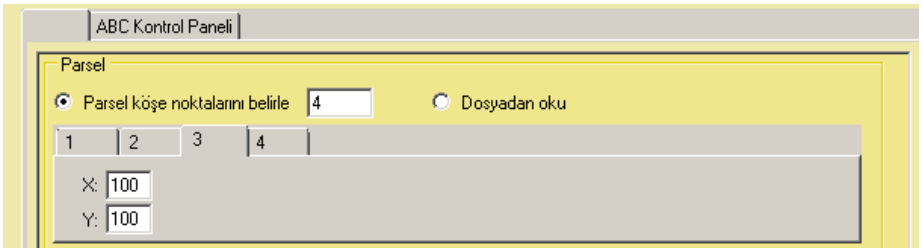
Aşağıda köşe sayısı 4 olarak belirlenmiş, bir kenarı 100 metre olan dikdörtgen bir parsel köşe noktaları koordinat girişi aktarılmaktadır (şekil 4.8, şekil 4.9, şekil 4.10, şekil 4.11) .



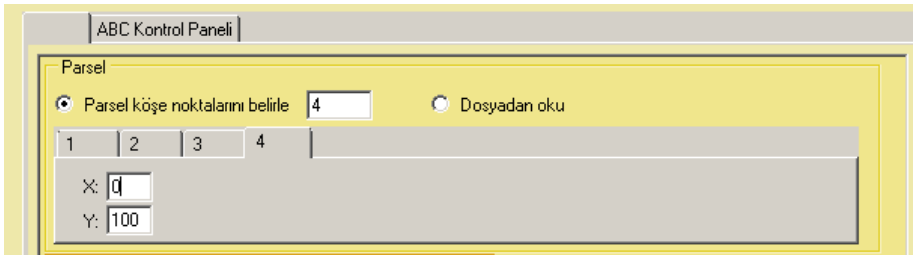
Şekil 4.8 : Parsel köşe noktaları girişi adım 1.



Şekil 4.9 : Parsel köşe noktaları girişi adım 2.

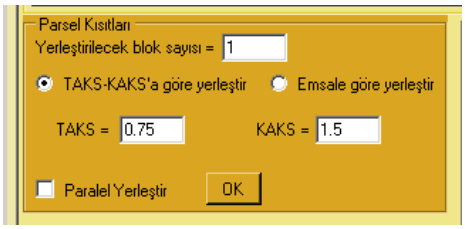


Şekil 4.10 : Parsel köşe noktaları girişi adım 3.



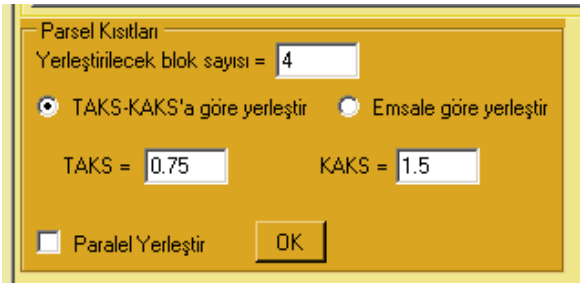
Şekil 4.11 : Parsel köşe noktaları girişi adım 4.

Temel parametreler 2. aşamada parsel içerisine yerleştirilecek blok sayısı ve TİY ‘de bulunan yerleştirilecek bloklara ilişkin kıstlamaların bulunduğu bölüm yer almaktadır. Yerleştirilecek blok sayısı 1 ile 999 arasında bir tam sayıdır. Bu blokların TİY ‘de yer alan yerleştirme-hesap işlemleri 2 türlü olabilmektedir. Bunlar TAKS – KAKS a göre yada yine TİY ‘de bulunan E (emsal) e göre hesap ve yerleştirme işlemleridir. Emsale göre yerleştirme işlemi seçildiğinde TAKS otomatikman devre dışı kalacaktır. Diğer bir deyişle tabanda kullanılacak alan serbest olacaktır.

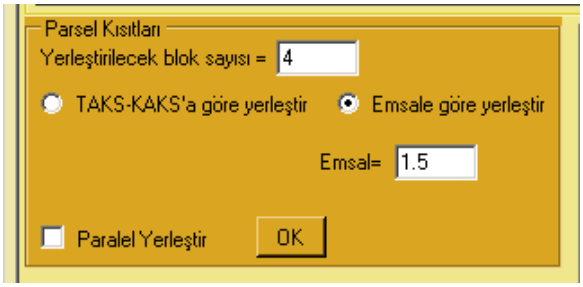


Şekil 4.12 : Parsel içine yerleştirilecek blok sayısı ve TAKS- KAKS- Emsal bilgilerinin girildiği kutular.

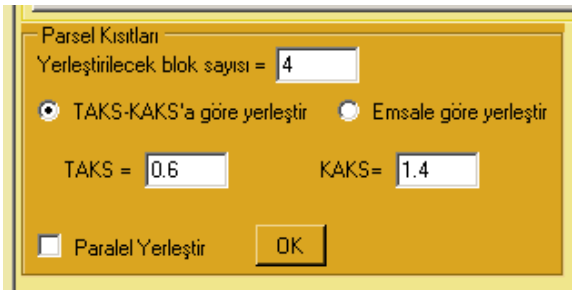
Aşağıda 100x100 metre boyutlarındaki parsele yerleştirilecek blok sayısı ve diğer işlemlere ilişkin bilgilerin girildiği kısım basamaklarıyla anlatılmıştır.



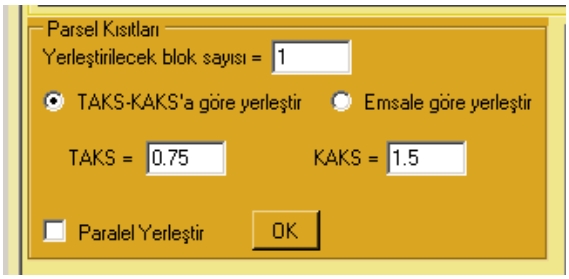
Şekil 4.13 : Blok sayısı girişi.



Şekil 4.14 : Emsal parametresine göre giriş.

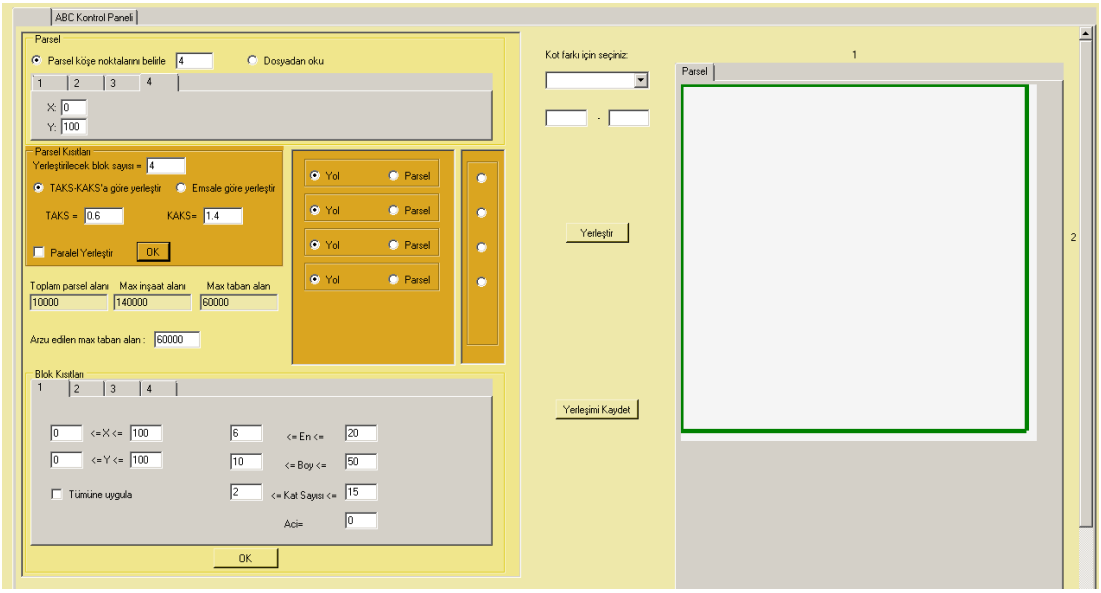


Şekil 4.15 : TAKS-KAKS parametrelerine göre giriş.



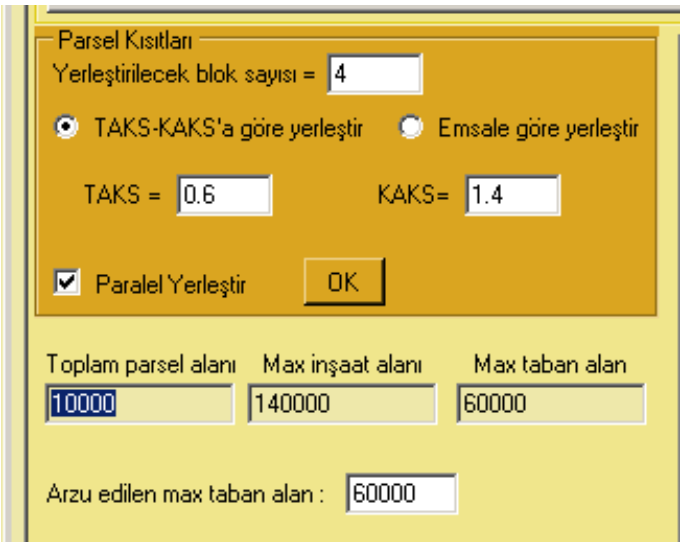
Şekil 4.16 : TAKS-KAKS değerleri değişikliği.

Burada yer alan TAKS değeri 0,4 ile 1 arasında ondalık bir sayıdır. KAKS ise TİY'de belirtildiği gibi 0,4 ile sonsuz arasında tam yada ondalık bir sayı olabilir. Yerleştirilecek blokları kenarlara paralel olarak yerleştirmeyi, parsel kısıtları bölümünün sol altında bulunan paralel yerleştirme butonunu tıklayarak sağlayabiliriz. Bu kısmı "ok" butonuna basarak geçtiğimizde, koordinatlarını girdiğimiz parseli sol tarafta görmemiz mümkündür.



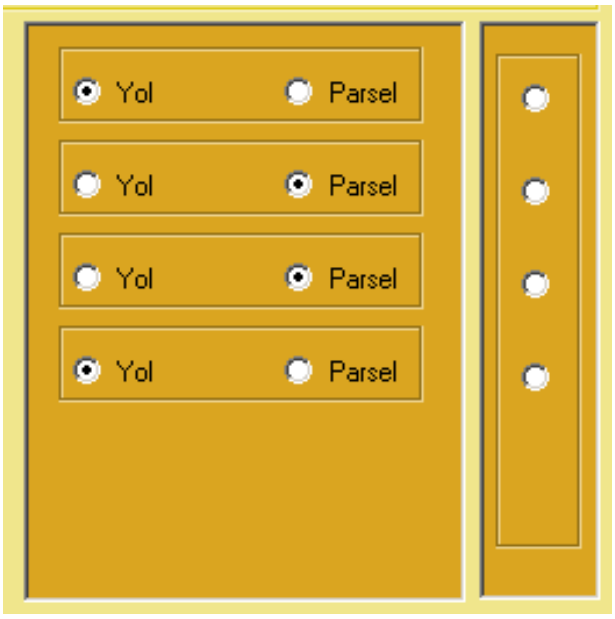
Şekil 4.17 : Parsel.

Parsel kısıtlarının altında bulunan bölümde toplam parsel alanı, maximum inşaat alanı ve maximum taban alanlarını görmemiz mümkündür. Arzu edilen taban alanı ile de tabanda kullanılacak alanı belirlememiz mümkündür. Parsel içerisinde tabanda farklı kullanımlar olabileceğini düşünerek bu seçenek kullanıcıya bırakılmıştır. Tabanda yeşil alan, otopark, çocuk oyun alnı, vb. kullanımların olabileceği göz önüne alınmıştır.



Şekil 4.18 : Toplam parsel, max. inşaat, max. taban alanları bilgilendirme kutuları.

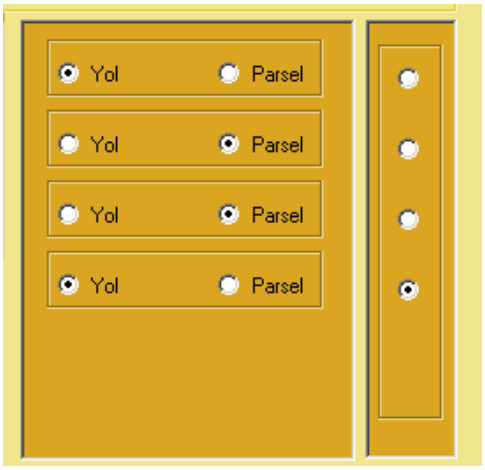
4 köşeli-4 kenarlı parselin komşusu olan; yol ve/veya komşu parsellerle ilgili seçimin yapılabileceği bölüm aşağıda gösterilmiştir. Bu durum sol kısımda bulunan parsel üzerine yol için turuncu çizgi, komşu parsel için yeşil çizgi ile gösterilecektir.



Şekil 4.19 : Kenar sayısına bağlı yol-komşu parsel durumunun seçilme işlemi.

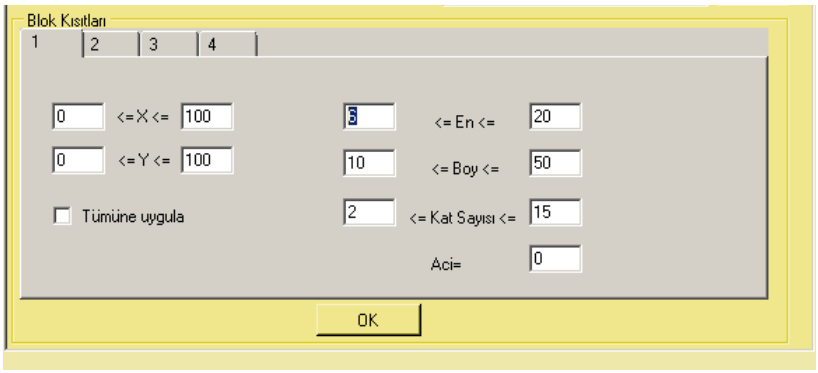
Parsel içerisinde bulunan eğime göre, TİY ‘e göre toplam inşaat alanı için baz alınacak kotun nereden alınacağına ilişkin seçimi mavi boyalı kısımdan seçmemiz mümkündür. Bu kot seçimi, bloklar içinde bulunacak kat sayısı ve metrekareye dahil edilmeyecek bodrum kat belirleme işlemi için gerekli bir adımdır.

Aşağıda (şekil 4.20) yolun bulunduğu kenardan alınacak kot noktasının nasıl işaretleneceği gösterilmiştir.

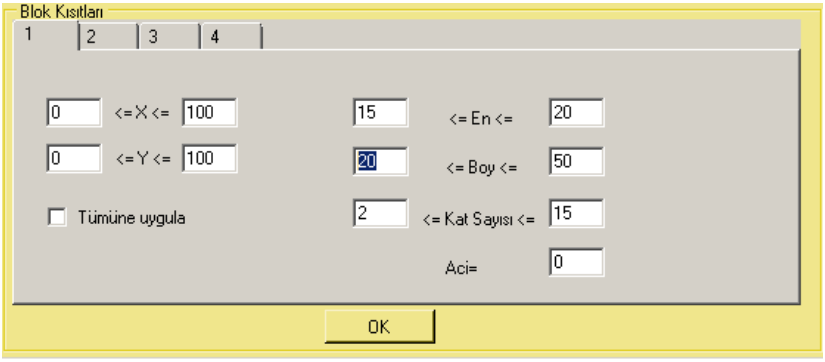


Şekil 4.20 : Kot alınacak kenar.

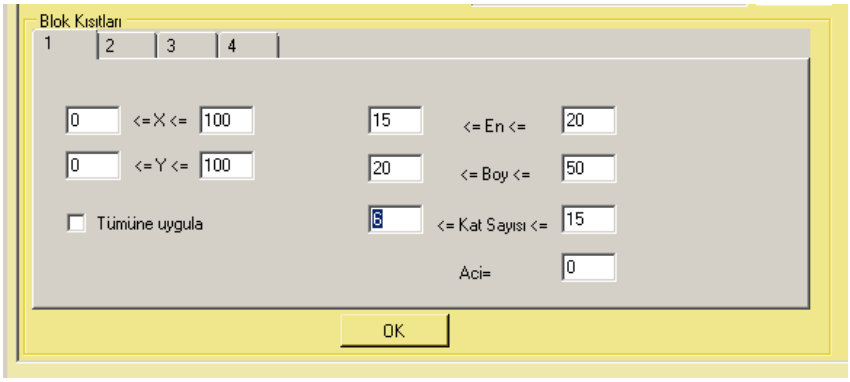
Temel parametreler 3. kısımda, bloklara ilişkin kısıtlamaların bulunduğu ayarlama kutucuğu bulunmaktadır. Bu kısımda yukarıda belirlediğimiz blok sayısı kadar bloğu sekme halinde görmek mümkündür. Her blok için ayrı ayrı pozisyon girilebilmesine olanak veren koordinat ekseni ayar kutuları ile blok/blokların hangi koordinatta bulunacağı da belirlenebilir. Bu kutularda belirlenen nokta (x,y) bloğun kütle merkezidir. Yerleşilebilecek koordinat parsel sınırı ile belirlenmiş olarak karşımıza çıkacaktır. Her bloğa farklı bir pozisyon verebileceğimiz gibi bu işlemi algoritmanın yapmasını da sağlayabiliriz.



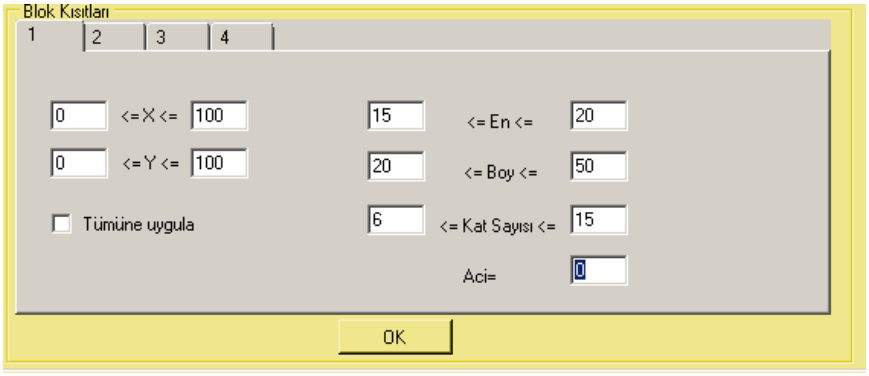
Şekil 4.21 : Bloklara ilişkin kısıtlamalar 1. adım.



Şekil 4.22 : Bloklara ilişkin kısıtlamalar 2. adım.



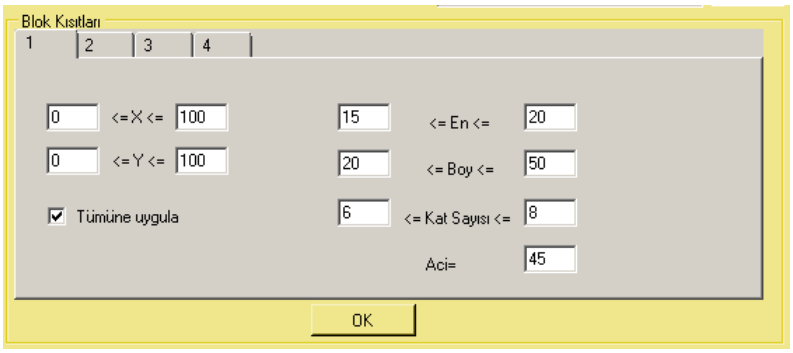
Şekil 4.23 : Bloklara ilişkin kısıtlamalar 3. adım.



Şekil 4.24 : Bloklara ilişkin kısıtlamalar 4. adım.

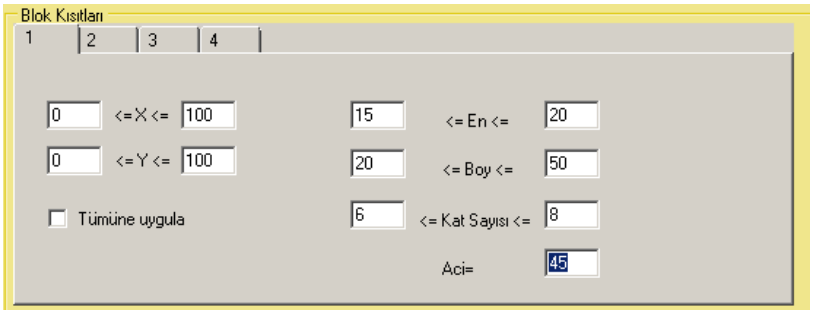
Bu kısımda yer alan blok parametreleri: en, boy, kat sayisi ve açi olarak belirlenmiştir. TİY içerisinde yapılacak bloklara ilişkin kısıtlamaların hepsi birebir programa yansıtılmıştır. Blok eni; 6 ile 40 metre arasında, boy 10 ile 50 metre arasında bir tam sayidir. Kat sayisi ise Türkiye şartları için 2 ile 30 arasında bir tamsayı olarak belirlenmiştir.

Bloklara ilişkin bilgileri tümüne uygula seçeneğiyle tüm bloklar için sabitlememiz mümkündür.



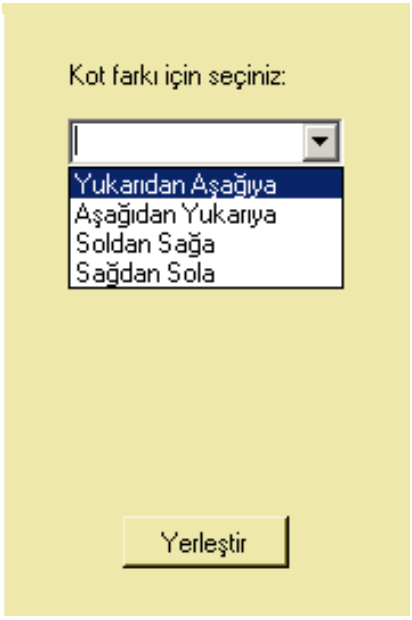
Şekil 4.25 : Bir blok için yapılan değişikliğin tüm bloklara uygulanması.

Ayrıca blok/bloklar için -360 ile 360 derece arasında değer girilebilmektedir.



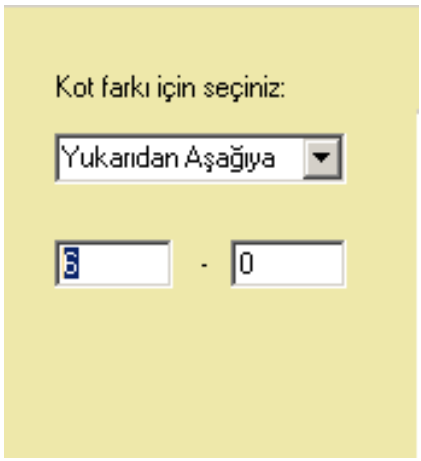
Şekil 4.26 : Açı girilmesi.

Parselde bulunan tek yönlü eğimin girilmesi programın geldiği bu aşamada sadece 4 yön ile sınırlıdır. Eğimin ve alınacak kotun önemi daha önceki bölümlerde anlatılmıştır.



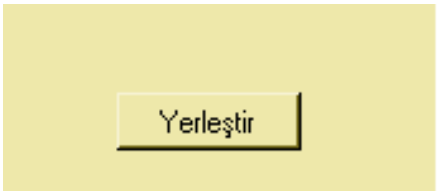
Şekil 4.27 : Parsel için eğim yönü belirlenmesi işlemi.

Eğim yönünün belirlenmesinden sonra, parsel içerisinde bulunan tek yönlü eğimin 4 temel yöndeki noktaları arasındaki farkın 0 ile 24 arasında tam sayı olarak girilmesi gerekmektedir.



Şekil 4.28 : Eğim, kot farkı belirleme işlemi.

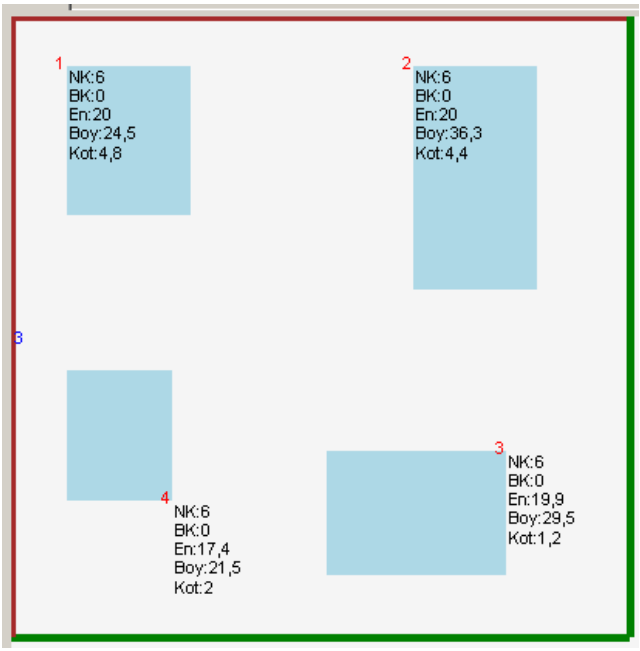
Bu işlemlerden sonra "yerleştir" butonu ile yazılımın, özellikleri belirlenen parsel, girilen TİY parametreleri, blok sayısı ve özelliklerine göre işlem yapması ve sonuç ürünleri göstermesi mümkündür.



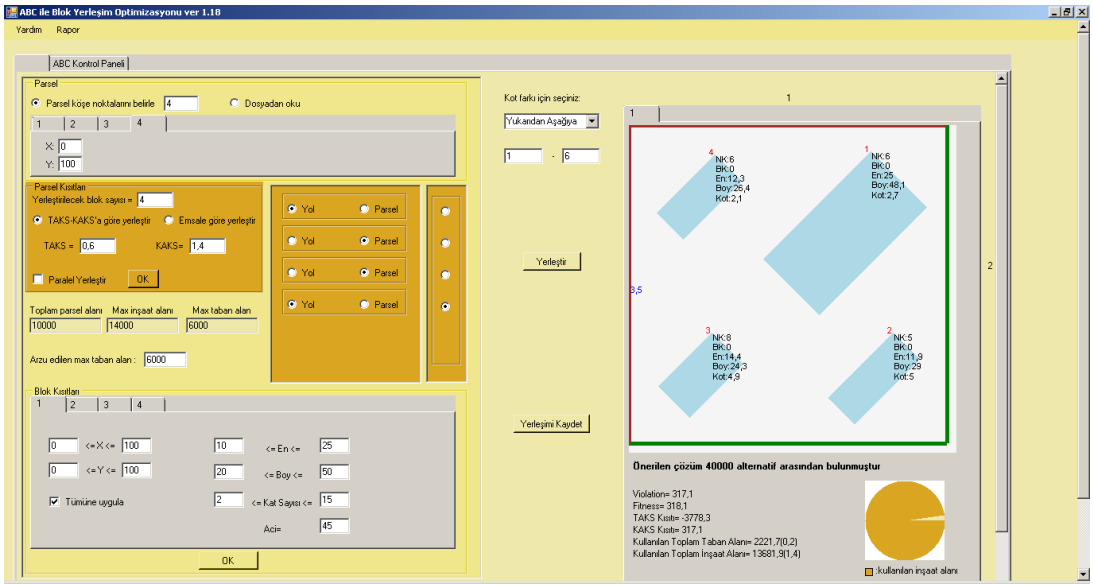
Şekil 4.29 : Üretme-yerleştirme butonu.

Arayüz çıktılarından anlaşılacağı üzere, yerleştirilen blokların üzerinde bloğun normal kat sayısı (NK), bodrum kat sayısı (BK), en, boy ve bloğun oturduğu kot yazılı bir şekilde kullanıcıya gösterilmektedir. Blokların numaraları, her bloğun sol üst kısmında yer almaktadır. Bloklar aldıkları numaralar değişmemek kaydı ile, yerleştirme (pozisyon atama) işleminde parsel içerisinde yapılaşma alanı içerisinde sıraya bağlı olmaksızın yer alabilirler. Blokların herbiri için değişikliğe ihtiyaç duyulursa, blok kısıtları bölümünde yer alan bloğun numarasının yer aldığı sekme tıklanarak blok üzerinde değişiklik yapılabilir. Aşağıda yukarıda belirlenen-seçilen özelliklere göre genetik algoritmanın oluşturduğu çözümlerden birkaç tanesi yer almaktadır (şekil 4.30, 4.31, 4.32). Blokların oturduğu kotlar ve kotun alındığı nokta gösterilmektedir. Aşağıdaki örnekte görülen, parselin sol kısmında, mavi renkli

rakam kot farkını belirtmektedir. Kütle numaraları kırmızı renktedir. Kütlelerin normal kat sayısı (NK), bodrum kat sayısı (BK), eni, boyu, oturdukları kot siyah yazıyla kütle (açık mavi) üzerinde bulunmaktadır (şekil 4.30).

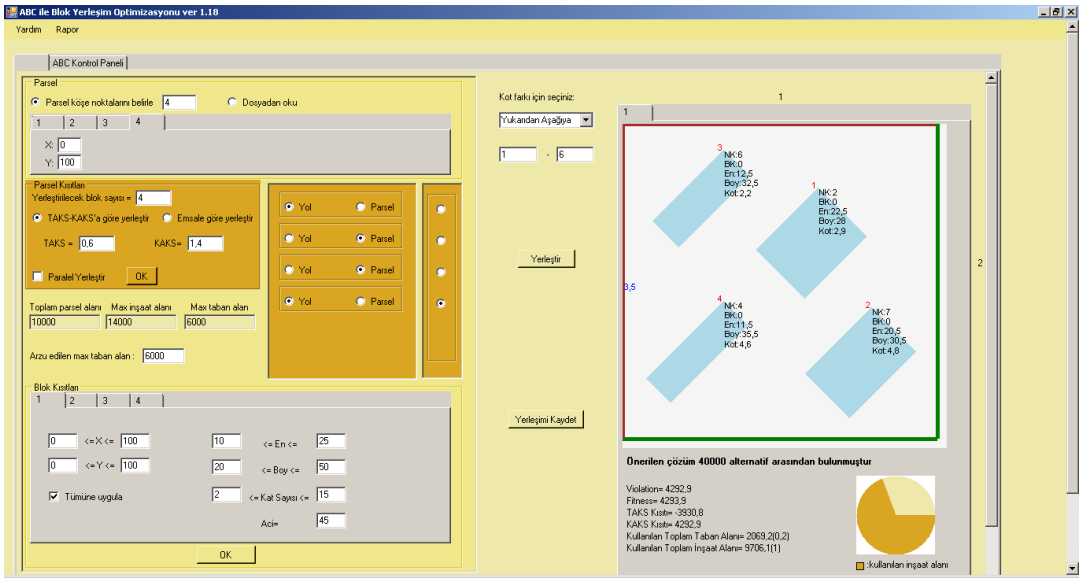


Şekil 4.30 : Üretilmiş vaziyet planı alternatifi 1.



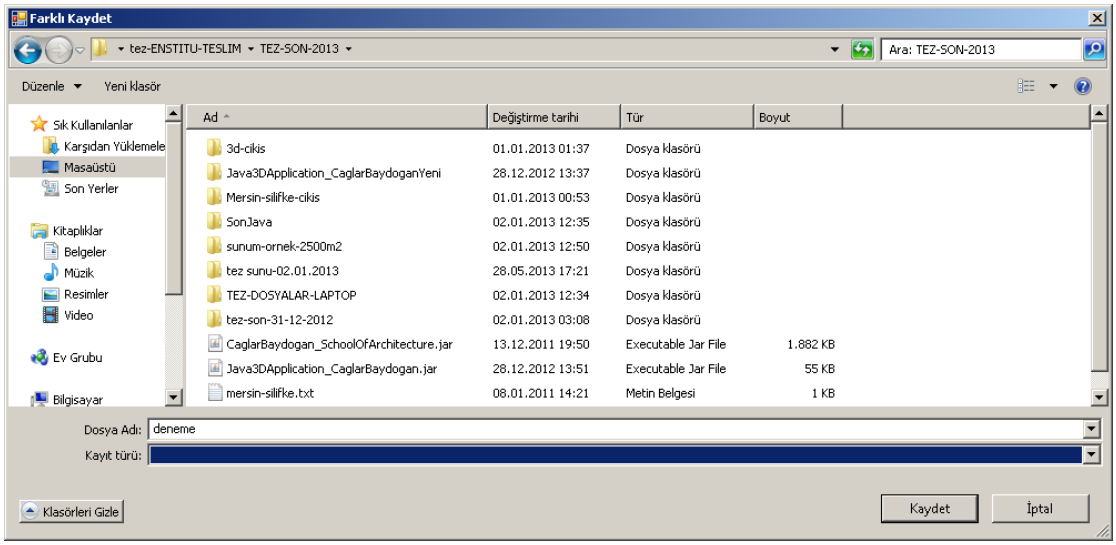
Şekil 4.31 : Üretilmiş vaziyet planı alternatifi 2.

Aşağıdaki alternatifi kaydetme işlemini "yerleşimi kaydet" butonu aracılığı ile yapmamız mümkündür. Kaydetme işlemi bu aşamada görüntü (jpeg) formatında olmaktadır.



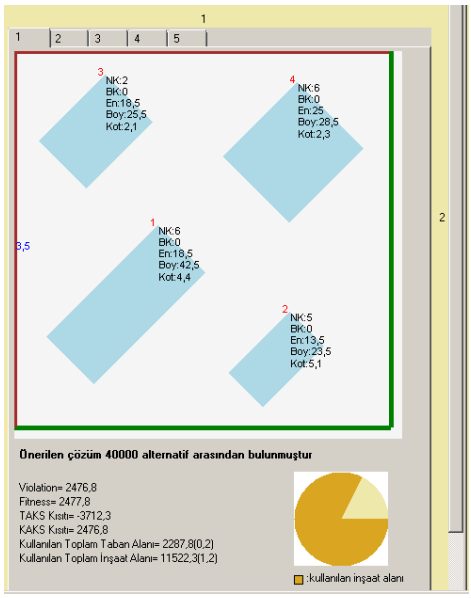
Şekil 4.32 : Üretilmiş vaziyet planı alternatifi 3.

Bir parsel için yerleşim alternatiflerini görmek ve bunları saklamak kullanıcı için önem taşımaktadır. Yerleşimi kaydet sekmesine tıkladıktan sonra çıkan farklı kaydet penceresi ile belirtilen konuma yerleşim alternatifini kaydetmek mümkündür (şekil 4:33).



Şekil 4.33 : Farklı kaydet penceresi.

Program istenilen sayıda koşma yaparak farklı vaziyet planı alternatiflerini tek seferde üretmektedir. Şekil 4.34 ‘de 5 adet koşma yapılarak üretilen vaziyet planı alternatiflerinden biri görülmektedir.



Şekil 4.34 : Kaydedilen imaj (vaziyet planı alternatifi)

Programda şu an gelinen aşamada sadece üretilen vaziyetle ilgili özet bilgilerin yeraldığı kısım aşağıda aktarılacaktır.

Programın ürettiği alternatifler için; yapılaşılan alan içerisindeki koşulları, alternatifin uygunluk, ihlal, TAKS, KAKS değerlerini üretilen vaziyet planı altındaki bölümde görebilmek mümkündür. Bu bilgilere ek olarak şekil 4.35'de üretilen alternatif planda kullanılan inşaat alanını ve önerilen çözümün kaç adet farklı alternatif arasından bulunduğunu gösterir bilgilendirme bölümü içerisinde yer almaktadır.



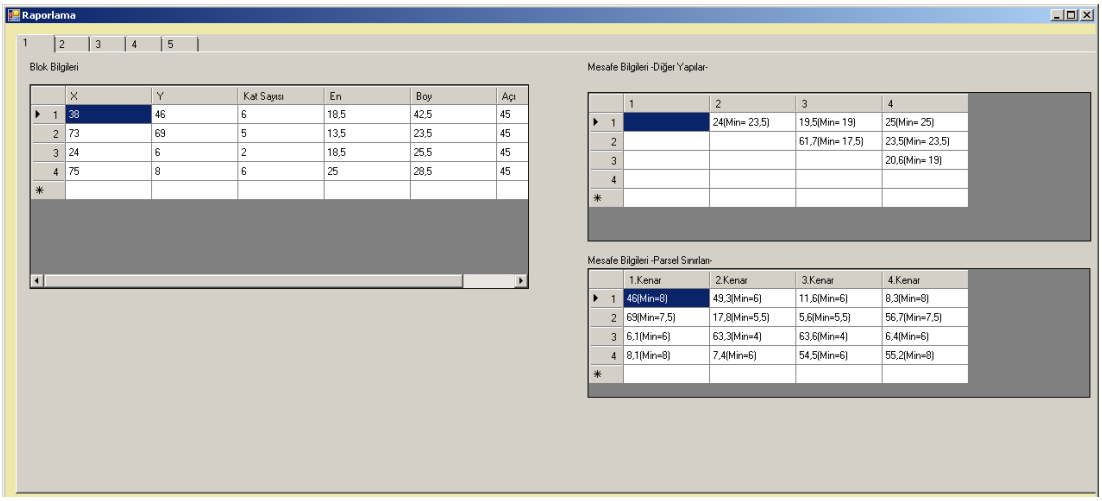
Şekil 4.35 : Alternatif özet bilgisi.

2. Rapor bilgileri paneli: bu panel sayesinde üretilen alternatifler içindeki mesafe bilgilerine, bloklara ait bilgilere tablo olarak ulaşılabilmektedir. Her bloğun kenarlara olan mesafeleri, minimum mesafeleri, birbirleriyle olan mesafeleri, aralarında olması

gereken minumum mesafeler ve bloklara ait en, boy, kat adedi, açı ve pozisyon bilgileri tablolar ile kullanıcıya sunulmaktadır (şekil 4.36, şekil 4.37).

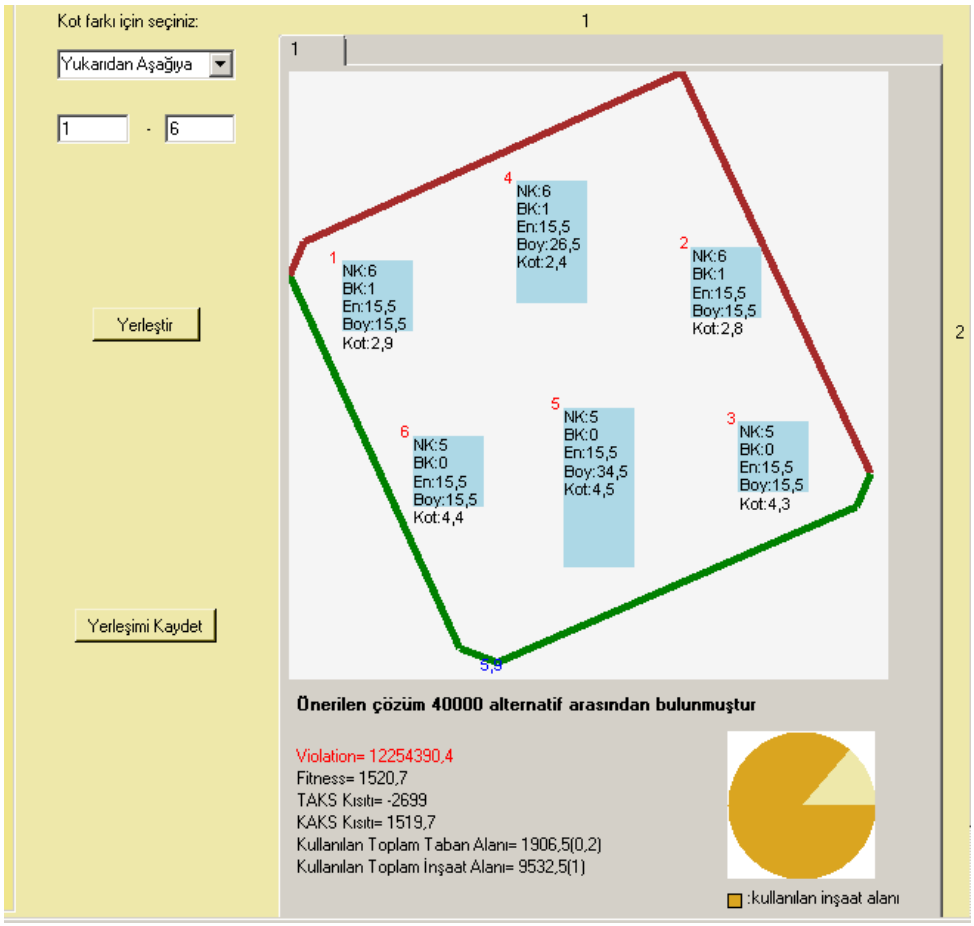


Şekil 4.36 : Raporlama butonu.



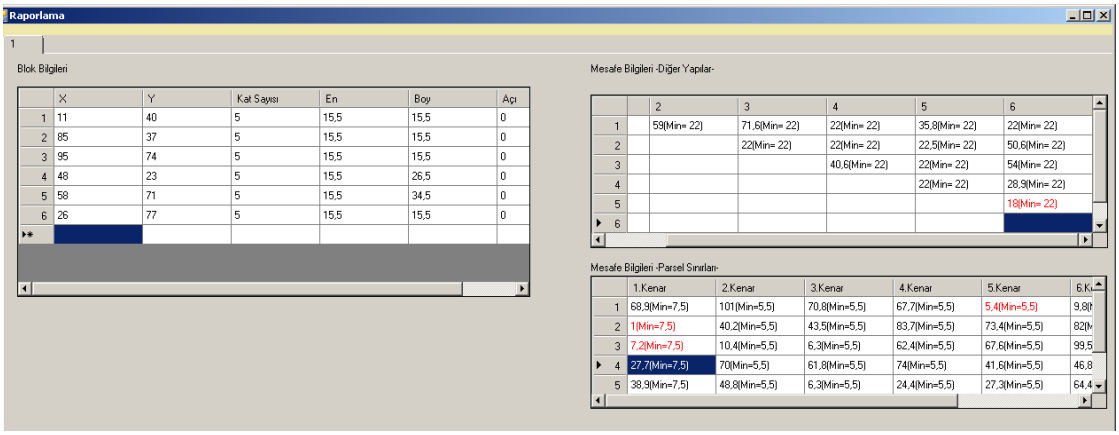
Şekil 4.37 : Raporlama penceresi.

Üretilen vaziyet planı aletrnatifinde "violation" ihlal/ler varsa özet bilgi kısmında kırmızı renkle, raporlama içerisinde yine kırmızı renkle gösterilmektedir. Aşağıdaki şekilde uygun olmayan alternatif için violation (ihlal) değeri kırmızı renkle gösterilmiştir. İhlal nekadar büyükse programdaki violation değeri o kadar büyük bir değer almaktadır. Kısıtlayıcıların çok olması bazen çözüm uzayının çok daralmasına sebep olmaktadır. Bu durumda ihlal-violation sıkça görülebilir (Şekil 4.38). Kısıtlamalara uyan çözümün diğer bir deyişle ihlal-violation değerinin 0 (min) olduğu çözümün bulunması için algoritmanın birden fazla aynı ayarlarla çalıştırılması ("run" işlemi yapılması) yeterli olabilmektedir.



Şekil 4.38 : Violation (ihlal).

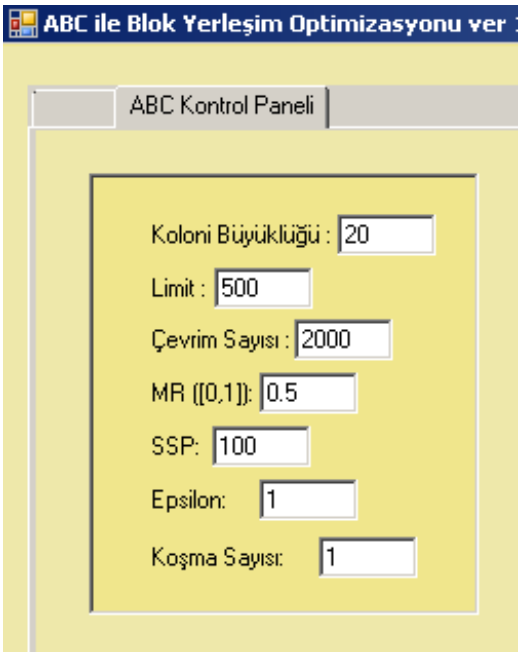
Bu ihlalin alternatif üzerinde, nerede/nerelerde olduğunu bulmak için raporlama sayfasında kırmız renkle gösterilen kısma (şekil 4.39) bakmak yeterli olacaktır. Matriste ihlalin (hatanın) nerede olduğu; "mesafe bilgileri diğer yapılar", "mesafe bilgileri parsel sınırları" kısımlarında kullanıcıya sunulmaktadır.



Şekil 4.39 : Raporlama kısmında violation (ihlal).

3. ABC algoritması ara yüzü: Artificial Bee Colony algoritmasının işleyişine ait parametrelerin kontrolünün yapıldığı kısımdır. diğer bir deyişle; sürü temeli diğer algoritmalara benzer olan ABC algoritması için populasyon, döngü, değişme oranı, olasılığı, vb. ayarların yapıldığı arayüzdür.

Bu panelde bulunan parametrelerin ABC algoritmasında ne olarak ifade edildiği aşağıda açıklanmıştır.



Şekil 4.40 : ABC kontrol paneli.

Koloni büyüklüğü: Kaynakların pozisyon bilgilerini içeren nektar kaynağı popülasyonudur. Popülasyon büyüklüğü, popülasyon içerisinde yer alan bireylerin sayısı.

Limit: Kaynağın terk edilmesi için belirlenmiş deneme sayısı "limit" parametresi ile belirlenmektedir. (çözüm uzayının büyüklüğü =D-string in boyu)

Çevrim sayısı: İterasyon sayısı, yineleme sayısı

MR ([0,1]): Alternatif çözümlerin, çözüm string boyu ne kadar büyük olursa olsun MR değeri ile bu çözümlerin en az 1 parametre için değişmesini sağlamaktadır. Genetik algoritmalarda, popülasyondaki farklılığı sağlayan çaprazlama operatörü ile aynı işlemi yapar. Değeri 0.1 ile 1 arasında bir değerdir.

SSP: Kaşif arının ortaya çıkma durumunu kontrol etmek için kullanılır. Diğer bir deyişle; kaynağın tükenip tükenmediğinin kontrolünün hangi çevrimde-iterasyonda yapılacağını kontrol etmek için kullanılmaktadır.

Epsilon: Matematiksel olarak hata payının nekadar olabileceğini gösterir bir sayıdır.

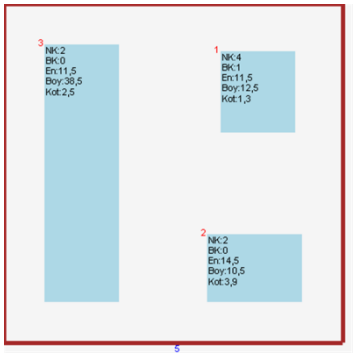
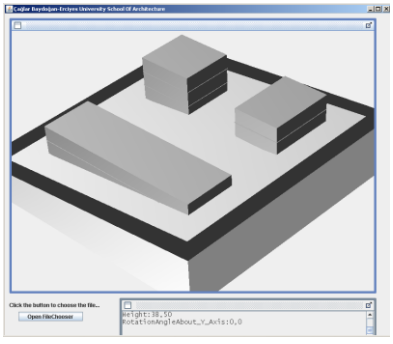
Örnek olarak; programın, üretilecek alternatifteki KAKS, girilen KAKS değeri oranını eşitlemeye çalışması, yani ideal olana eşitlemeye çalışması normaldir. Fakat 1 birim eksik olabilme durumu, olası ihlalleri ve bulunan çözümlerin sayısını çok etkilemektedir. Kesin doğru, net olan sonucu esnetmek, olası çözümleri artırmak için kullanılır. Diğer bir deyişle, aslında program girilen KAKS değeriyle kat alanları toplamına göre değil de var olan hata payına göre maksimize etmeye çalışır.

Koşma sayısı: Programın-algoritmanın tek seferde kaç tane vaziyet planı alternatifi üreteceğini belirlemek için kullanılır.

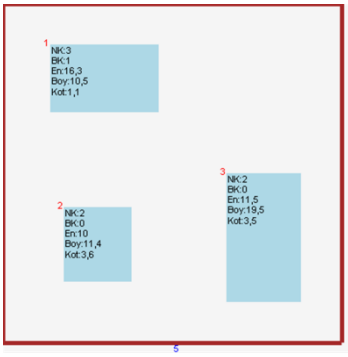
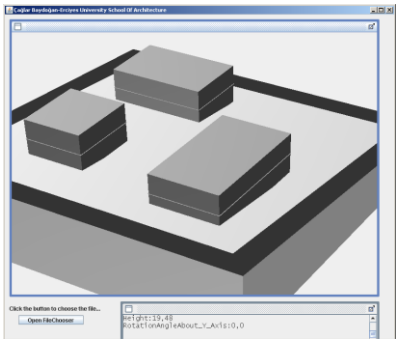
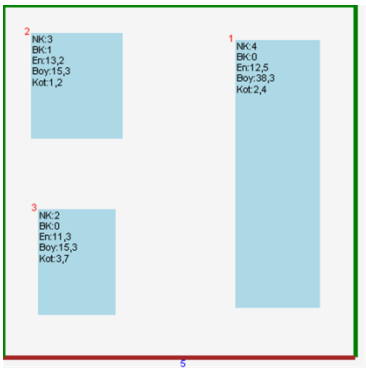
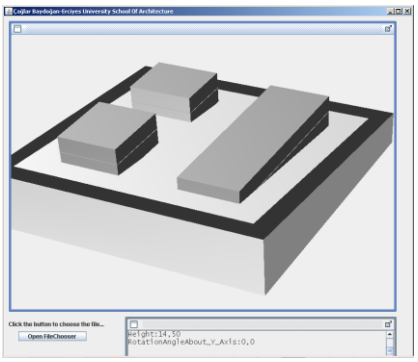
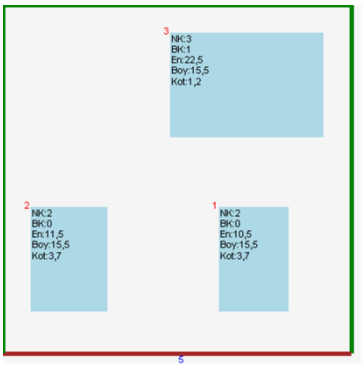
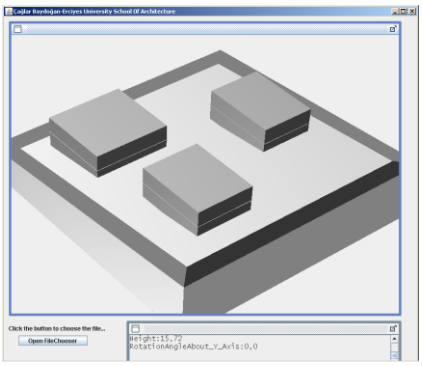
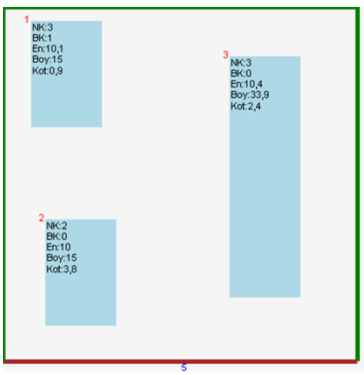
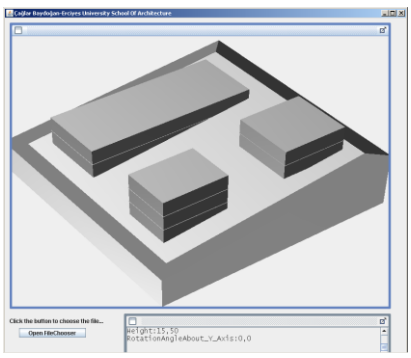
4.4. Uygulamalar

Algoritmanın farklı parsel ve değerler için ortaya çıkaracağı alternatifler bu bölümde değerlendirilecektir. Programın farklı tipte parseller ve farklı blok sayıları için geliştirdiği alternatifler uygulama örneği olarak verilmiştir. Çalışmanın bu kısmında, 2500 m2 lik arsa büyüklüğüne sahip kare ve yamuk parsel için vaziyet planı üretilmiştir. Yazılımın farklı geometrik şekillerdeki parseller için ürettiği vaziyet planları aşağıda verilmiştir.

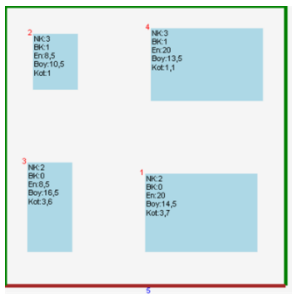
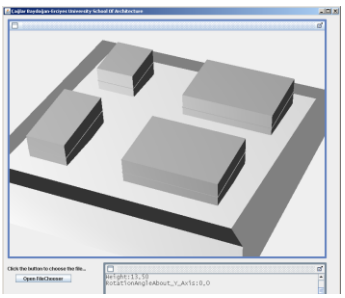
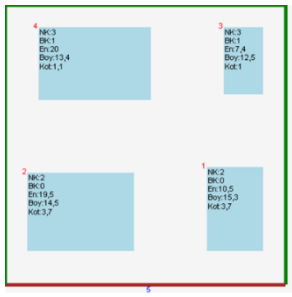
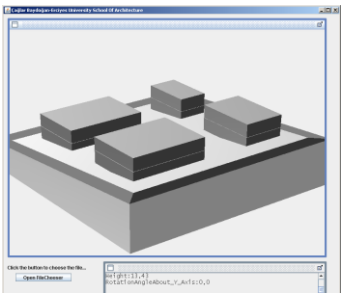
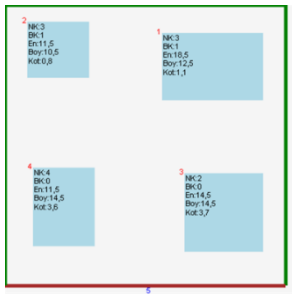
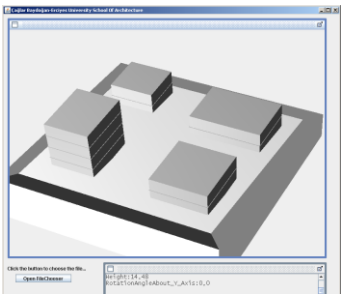
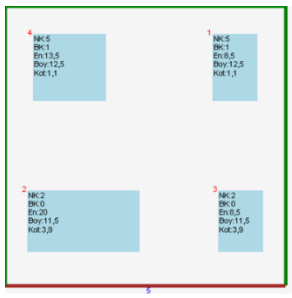
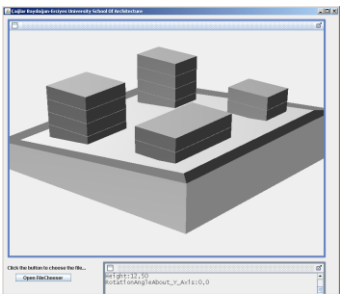
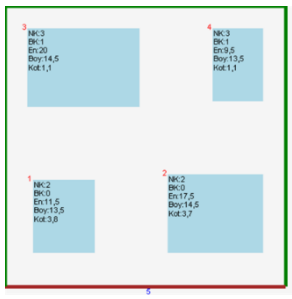
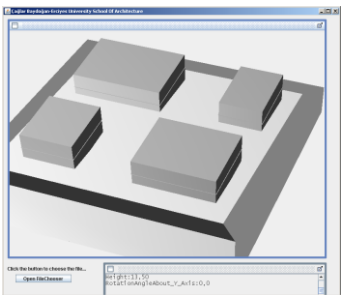
Çizelge 4.4 : Kare parsel 3 blok için vaziyet planı alternatifleri.

Alternatif no	Alternatif	3 boyut
1		

Çizelge 4.4 (devam) : Kare parsel 3 blok için vaziyet planı alternatifleri.

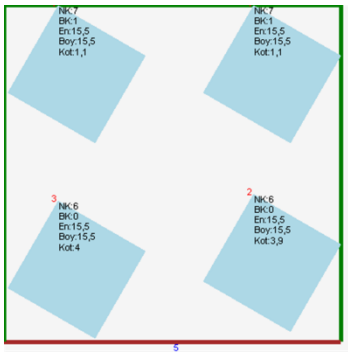
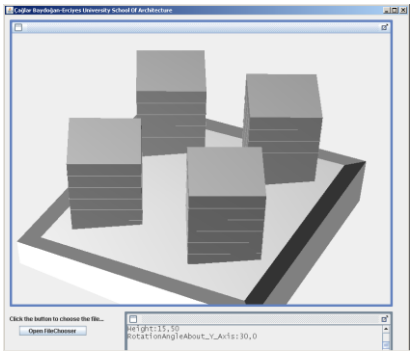
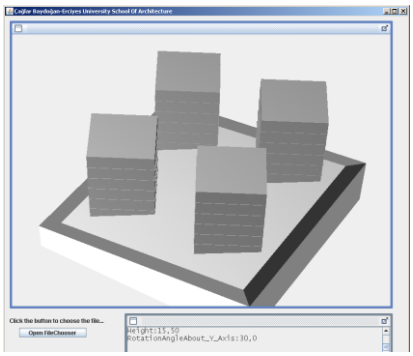
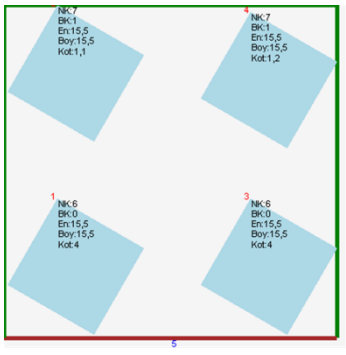
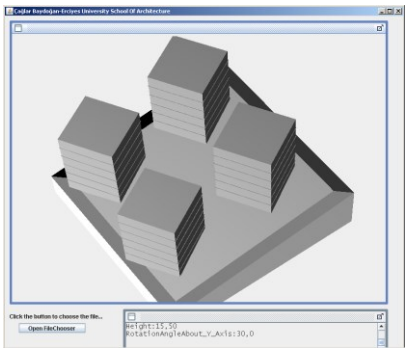
2		
3		
4		
5		

Çizelge 4.5 : Kare parsel 4 blok için vaziyet planı alternatifleri.

Alternatif no	Alternatif	3 boyut
1		
2		
3		
4		
5		

Kare parsel 4 blok için üretilen vaziyet planlarından bazıları yukarıdaki çizelgelerde (Çizelge 4.4, Çizelge 4.5) gösterilmiştir. Üretilen vaziyet planları için açı 0 derecedir. Aynı şartlarda 30 derece için üretilen vaziyet planı alternatiflerinden birkaçı Çizelge 4.6 da gösterilmiştir.

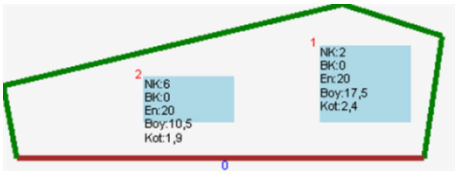
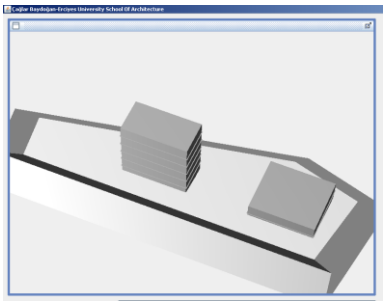
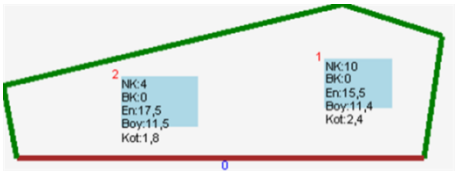
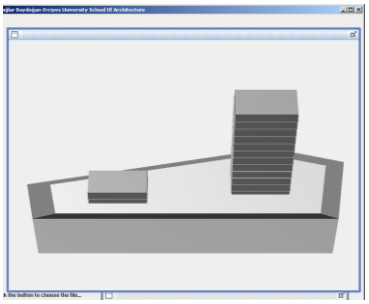
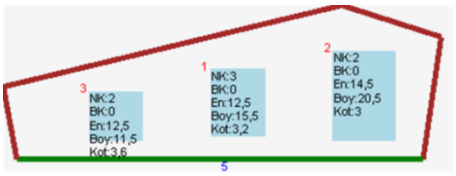
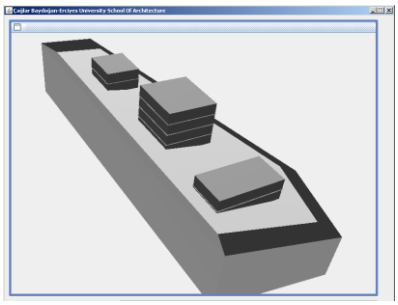
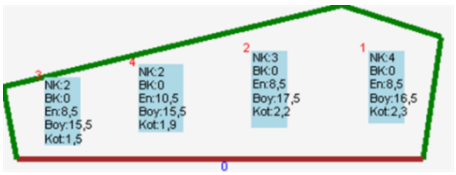
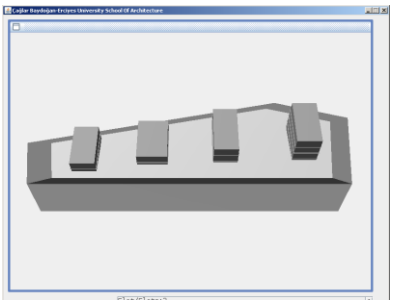
Çizelge 4.6 : Kare parsel 4 blok, 30 derece açı için vaziyet planı alternatifleri.

Alternatif no	Alternatif	3 boyut
1		
2		
3		

Yamuk parseller için üretilen vaziyet planları aşağıda verilmiştir. 2500 m2 lik yamuk parsel için üretilen 2-3-4 blok için vaziyet planları tek bir çizelgede (Çizelge 4.7) gösterilecektir. Üretilen vaziyet planlarında (her iki parsel şekli için kare-yamuk

parsel) TAKS: 0.7, KAKS :1.5 olarak sabitlenmiştir. Bloklar için en, boy boyutları; 14<en<21, 14<boy<36 şeklinde belirtilmiştir. Bu iki parsel şekli için kat sayısı belirtilmemiş olup, yazılımın kat adetlerini belirlemesi beklenmiştir. Diğer bir yandan eğimle ilgili kot farkı seçeneklerine; "yukarıdan aşağı" ve kot farkı olarak "5" değeri girilmiştir.

Çizelge 4.7 : Yamuk parsel 2-3-4 blok için vaziyet planı alternatifleri.

Alternatif no	Alternatif	3 boyut
1		
2		
3		
4		

4.5. Konvansiyonel Sistemlerle Üretilen Vaziyet Planları İle Karşılaştırma

Konvansiyonel sistemler, bilindik, üzerinde uzlaşılmış, daha önceden süregelen üretim şekilleri anlamında kullanılmaktadır.

Elde veya bilgisayar ortamında yapılan çizim, el ile yada hesap makinası kullanılarak yapılan hesaplar, TİY maddelerine uyularak üretilen vaziyet planları konvansiyonel, genelde kullanılan üretim şekilidir.

Çalışmanın sonuçlarını test etmek ve konvansiyonel sistemlerle karşılaştırma yapmak için; özellikle ada ölçeğinde konut alanı olarak belirlenmiş alanlar için, Türkiye şartlarında en fazla toplu konut üreten kurum olan TOKİ'nin toplu konutlar için ürettiği yerleşim planlarını; vaziyet planı ölçeğinde mekansal örgütlenme, yerleşim ve tip imar yönetmelikleri çerçevesinde, yeniden tez kapsamında yapılan program ile üretilmiştir. Karşılaştırma kriterleri; E(emsal), Hmax, TAKS ve KAKS, çekme mesafelerine ve diğer hesaplarına uygunluk, uygun çözüm- üretim süresi olarak belirlenmiştir.

TOKİ, Türkiye şartlarında konut üretimi programı dâhilinde bugün itibariyle, Türkiye'nin 540 ilçesinde ve 1200 şantiyesinde konut üretmektedir (Url-11). Görüldüğü üzere Türkiye'de konut üretimi ve toplu konut vaziyet planlarını 25 yıldan fazla süredir üreten, uygulayan bir kurumun elinden çıkan vaziyet planları ile oluşturulan yazılımın ürettiği vaziyet planları arasında karşılaştırma yapmak doğru olacaktır.

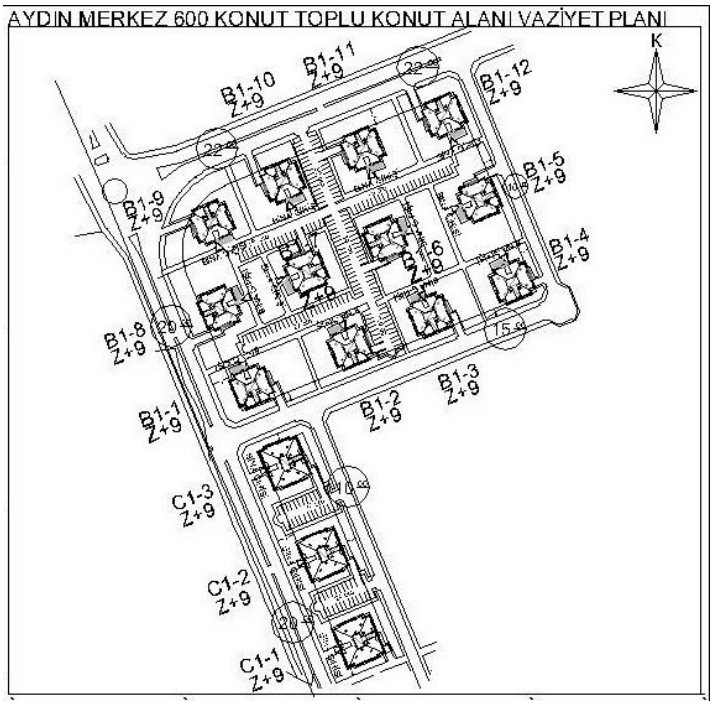
TOKİ 'nin son dönemde ürettiği vaziyet planlarına baktığımızda bunların birçoğunun mahalle ölçeğinde, alt yapı-kamusal alanları, ilkokul, alışveriş merkezi, dini yapı, kapalı spor alanları vb. barındıran ölçekte olduğunu görmekteyiz. Hazırlanan yazılımın sadece konut alanları için alternatifler üretmesi düşünüldüğünden, TOKİ'nin ürettiği vaziyet planlarından, işlev olarak sadece konut olarak düşünülenler incelenmiş ve test etmek için seçilmiştir.

TOKİ 'nin toplu konutlar için ürettiği vaziyet planı örneklerinden Manisa merkez'de toplu konutlar için üretilen vaziyet planı şekil 4.41. de verilmiştir.



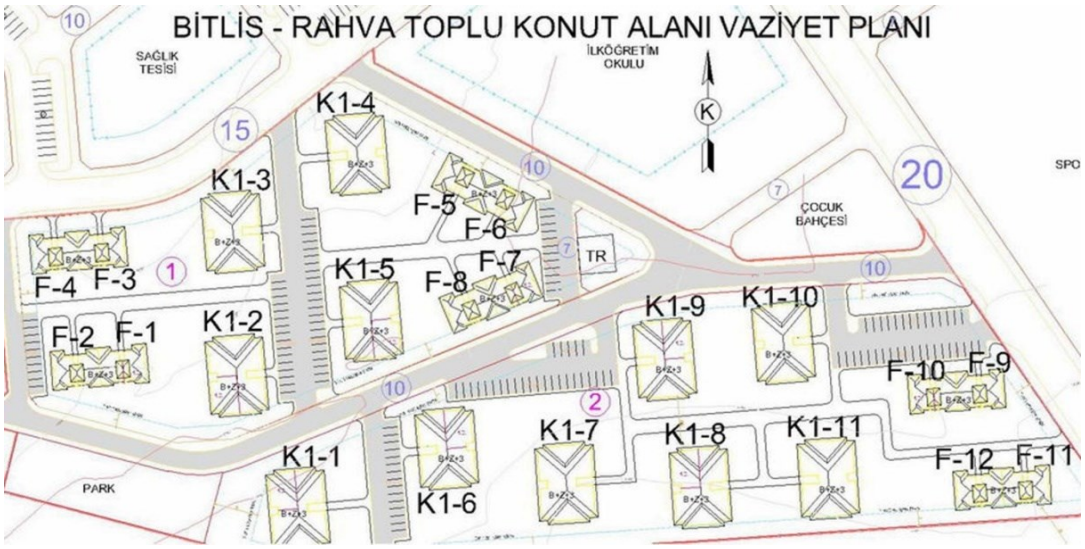
Şekil 4.41 : TOKİ, Manisa, Merkez (Akhisar), 3. Etap (Anonim).

TOKİ'nin ürettiği vaziyet planlarından birçoğunu çalışma kapsamında programın da üretebileceği varsayılmaktadır. Konvansiyonel yöntemlerle kıyaslama yapmak için eşit şartlarda, programın en uygunu bulma kapasitesini test etmek ve hali hazırda üretilen methodlarla yarıştırmak amacıyla içinde sadece "konut yerleşim çözümü", sunulmuş olan vaziyet planlarından seçilen vaziyet planları için üretilen alternatifler bölüm sonunda verilecektir. Aşağıda (Şekil 4.42) Aydın Merkez de TOKİ'nin ürettiği vaziyet planı bulunmaktadır.



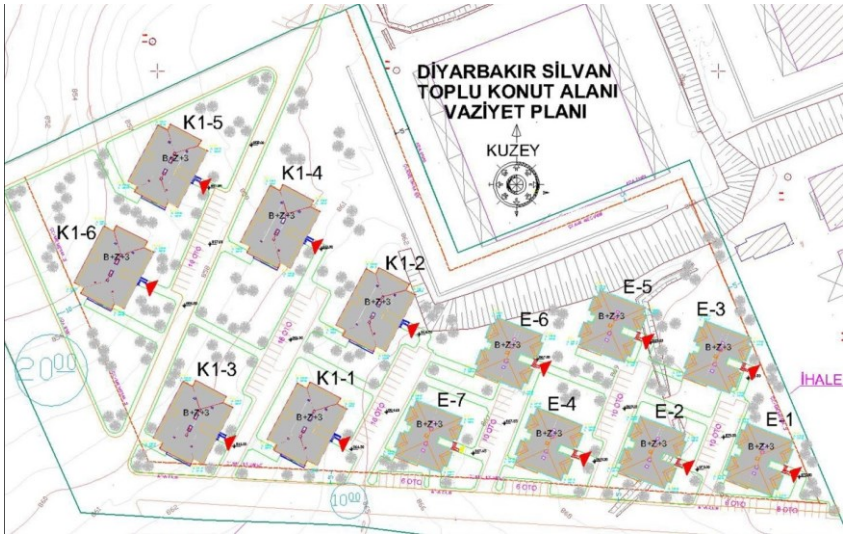
Şekil 4.42 : TOKİ, Aydın, Merkez, sadece konut (Anonim).

Genelde tek tip veya 2 tip konut büyüklüğü olan alternatifler seçilmiştir. Bunun sebebi test etme işlemleri için pratik kullanım ve kütle büyüklüklüklerini de her yönüyle kıyaslama şansı sağlamak içindir.



Şekil 4.43 : TOKİ, Bitlis, Rahva (Anonim).

TOKİ 'nin toplu konutlar için ürettiği çözümler rasyonel sayılmaktadır. Çevresel, coğrafik konum, topografya dikkate alınarak yapılan çalışmalardır. Dolayısıyla programın bu çözümlere yakın bir çözüm yapabilmesi demek tasarımcıların vaziyet planlarına sadece teknik olarak bakıldığı anlamına gelmektedir. Bu durum istenmeyen bir durumdur. Algoritmanın üretebileği tipte yerleşim planlarını uygulamanın tasarım etiği açısından hiçbir anlamı yoktur. Daha iyi olanın, çevre kriterlerine, doğaya, insana uygun olanın üretilmesi kaçınılmaz olandır.



Şekil 4.44 : TOKİ, Diyarbakır, Silvan (Anonim).

Bu örneklerden de anlaşılacağı üzere; standartlaşma, tipleşme özellikle konut sektöründe Türkiye ‘de son dönemde tasarlama eylemi ve tasarımcılar için bir açmazdır. Yeni konuta olan çok talep üretim sektörünü tetiklemiştir. Hızlı üretme ve kısa zamanda üretilenleri tüketme son dönemde çokça yaşanan bir olgudur.

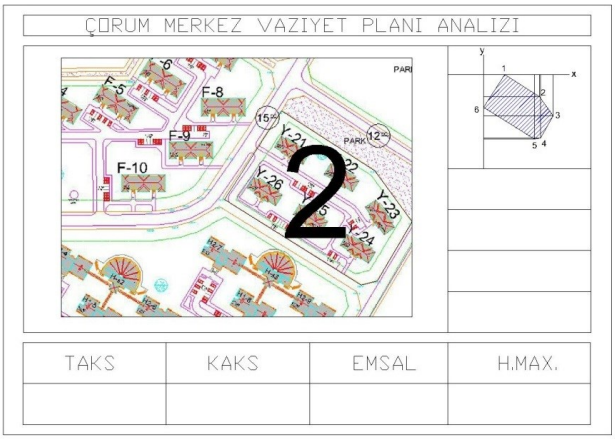


Şekil 4.45 : TOKİ, Eskişehir, Tepebaşı, Aşağısöğütönü, vaziyet planı 1 (Anonim).

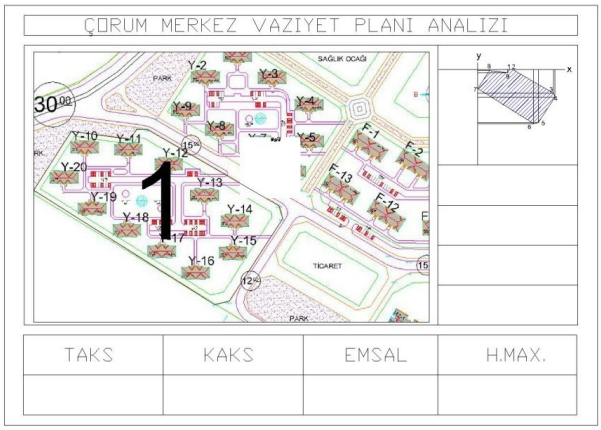
Yukarıdaki örneklerden anlaşılacağı üzere; TOKİ 25 seneyi aşan bir süredir genelde, Türkiye’ye için standart dikdörtgen prizması konutlar üretmiştir. Yapıldığı çevrenin imar durumuna ve TİY’e uyan 2-3-4 daireli konutlar vaziyet planı ölçeğinde parseller içerisine genelde standart farklı yapı düzeni arayışına girilmeden TOKİ tarafından yerleştirilmiştir. Sonuç itibariyle nerdeyse Türkiye’nin bütün kentlerinde imar kurallarına, diğer yapısal kurallara uyan birbirlerine çok benzeyen yapılardan oluşan siteler, uydu kentler TOKİ eliyle yapılmış ve yapılmaktadır. Çalışma kapsamında üretilen bilgisayar programı vasıtasıyla, TOKİ’nin yapmış olduğu “konut grupları vaziyet planları” için alternatifler üreterek bu alana (vaziyet planı üretmek) nasıl bir katkı koyulacağını araştırmasına da yardımcı olabilecektir.

TOKİ’nin ürettiği vaziyet planlarından 5 tanesi üzerinde analiz yapılmış ve konvansiyonel olarak üretilen bu vaziyet planlarıyla programın ürettiği vaziyet planları karşılaştırılmıştır. Bu analizlerden üç tanesi çalışma kapsamında sunulacaktır.

TOKİ’nin ürettiği vaziyet planlarından bazıları tüm bilgileriyle künyelenmiştir. Aşağıda bu künyeye ait imajlardan bulunmaktadır (şekil 4.46, 4.47).

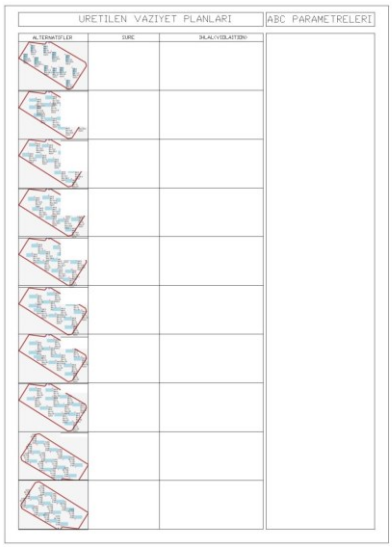


Şekil 4.46 : Çorum merkez 2 için yapılan künye örneği.



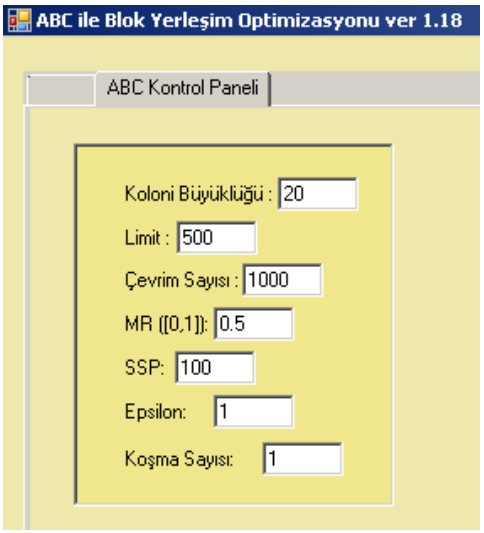
Şekil 4.47 : Çorum merkez 1 için yapılan künye örneği.

Programın ürettiği vaziyet planları da bütün bilgileriyle tablolara atılmıştır. Aşağıda bu tabloya bir örnek verilmiştir (4.47).



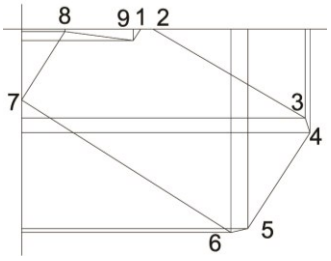
Şekil 4.48 : Çorum merkez 1 için üretilen vaziyet planları örnek tablosu.

Üretilen vaziyet planları standart ABC ayarlarıyla üretilmiştir.



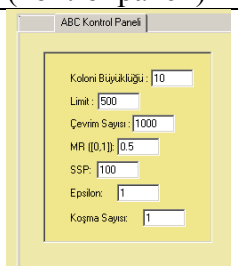
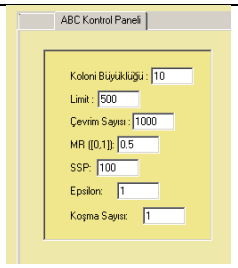
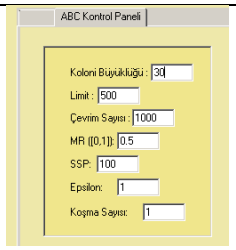
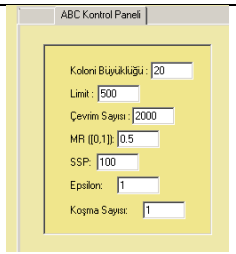
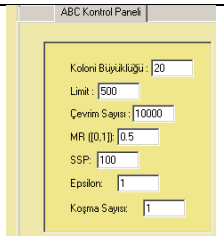
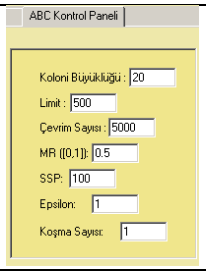
Şekil 4.49 : ABC standart ayarları.

TOKİ 'nin Çorum Merkez'de bulunan parsel için ürettiği vaziyet planı çalışma aşağıdaki (şekil 4.50) gibidir. Çorum Merkez vaziyet planından ve imar şartlarından programın alternatif üretirken kullanılabileceği bilgiler alınmıştır. Çalışma kapsamında sunulacak 3 adet vaziye planı için detaylı vaziyet planı künyeleri hazırlanmıştır. Parseller için koordinatlar, program için diğer parsel bilgileri ve imar şartları belirlenmiştir.

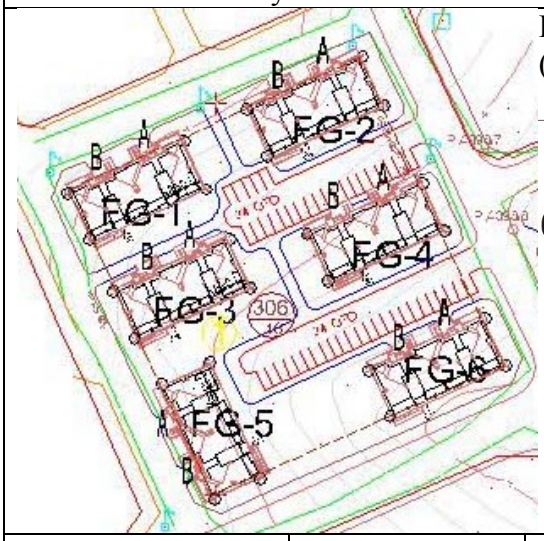
<table>
<tr><td colspan="4">Çorum Merkez vaziyet planı</td></tr>
<tr><td colspan="2">
</td><td>Parsel köşe koordinatları (vektörel)

</td><td>Koordinatlar
x y
106 0
117 0
252 79
256 92
201 177
186 180
0 63
38 2
99 10</td></tr>
<tr><td>TAKS</td><td>KAKS</td><td>EMSAL</td><td>HMAX/KAT</td></tr>
<tr><td>-</td><td>-</td><td>1</td><td>12,50/4</td></tr>
<tr><td>BLOK SAYISI</td><td>AÇI</td><td>ARSA KOT FARKI</td><td>BLOK</td></tr>
<tr><td rowspan="2">11</td><td rowspan="2">30°</td><td rowspan="2">0</td><td>EN</td><td>BOY</td></tr>
<tr><td>13 m.</td><td>30 m.</td></tr>
</table>

Şekil 4.50 : Çorum Merkez vaziyet planı detaylı künyesi.

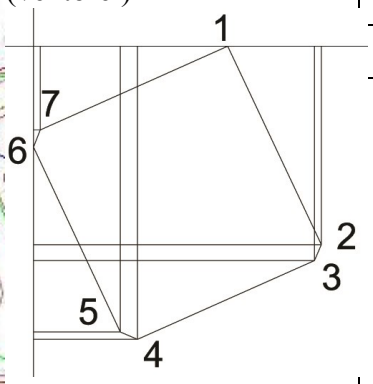
Çizelge 4.8 : Çorum Merkez için üretilen vaziyet planları.

Çorum Merkez için üretilen vaziyet planları			
Alternatifler	Süre	İhlal	Abc parametreleri (kontrol paneli)
1	267 sn.	var	
2	283 sn	var	
3	317 sn	var	
4	547 sn	var	
5	1121 sn	yok	
6	828 sn	yok	

Mersin Silifke Vaziyet Planı



Parsel köşe koordinatları (vektörel)

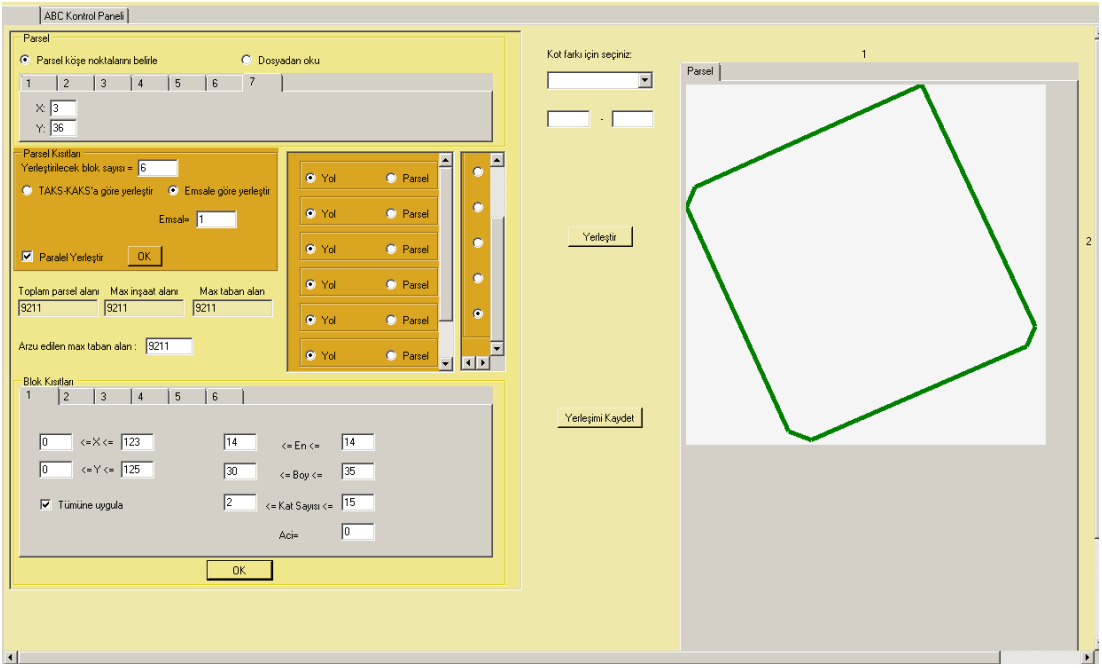


Koordinatlar	
x	y
83	0
123	85
120	92
44	125
36	12
0	43
3	36

TAKS	KAKS	EMSAL	HMAX/KAT	
-	-	1	-/6	
BLOK SAYISI	AÇI	KOT FARKI	BLOK	
6	0	Aşağıdan-yukarı 8 m	EN	BOY
			14	31

Şekil 4.51 : Mersin Silifke vaziyet planı detaylı künyesi.

Mersin Silifke için program içinde koordinatlar girilerek oluşturulan parsel ve program ayarları (şekil 4.52). Oluşturulan parselin şekli, kuzeye doğru yerleşmiş orijinal vaziyet planı yönünde ve şeklindedir.

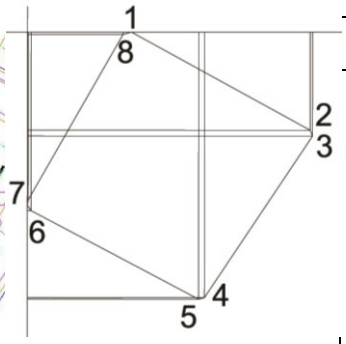


Şekil 4.52 : Programda Mersin Silifke parsel.

Çizelge 4.9: Mersin Silifke için üretilen vaziyet planları

Mersin Silifke için üretilen vaziyet planları			
Alternatifler	Süre	İhlal	Abc parametreleri (kontrol paneli)
1	312 sn	yok	
2	478 sn	yok	
3	391 sn	yok	
4	402 sn	yok	
5	267 sn	yok	

İzmir Torbalı'da TOKİ 'nin konut sitesine ürettiği vaziyet planı için yapılan künye aşağıdadır (şekil 4.53).

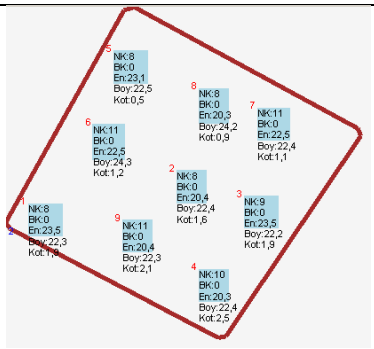
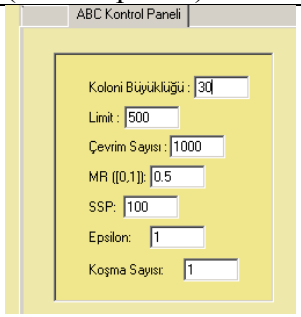
<table>
<tr><td colspan="4">İzmir Torbalı vaziyet planı</td></tr>
<tr><td colspan="2" rowspan="2">
</td><td>Parsel köşe koordinatları (vektörel)

</td><td>Koordinatlar
x | y
86 | 0
235 | 81
237 | 86
147 | 220
142 | 221
3 | 148
0 | 144
79 | 2</td></tr>
<tr><td>TAKS</td><td>KAKS</td><td>EMSAL</td><td>HMAX/KAT</td></tr>
<tr><td>-</td><td>-</td><td>4</td><td>30,50/10</td></tr>
<tr><td>BLOK SAYISI</td><td>AÇI</td><td>KOT FARKI</td><td colspan="2">BLOK</td></tr>
<tr><td>9</td><td>0</td><td>Yukarıdan-aşağıya
3 m.</td><td>EN
22</td><td>BOY
25</td></tr>
</table>

Şekil 4.53 : İzmir Torbalı vaziyet planı detaylı künyesi.

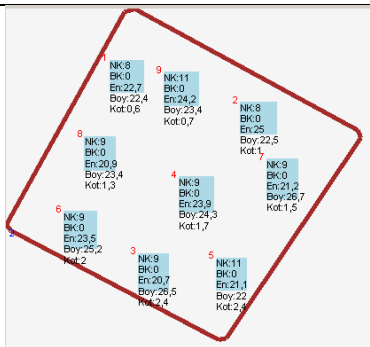
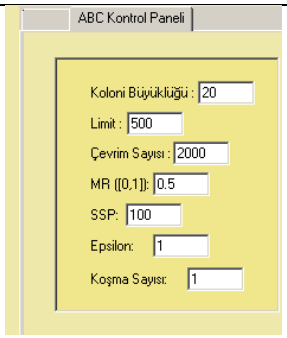
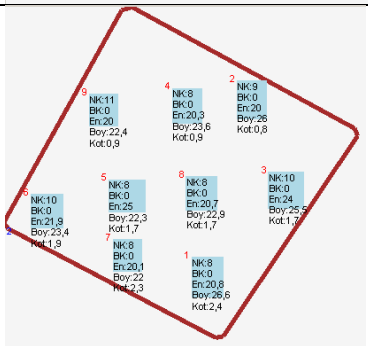
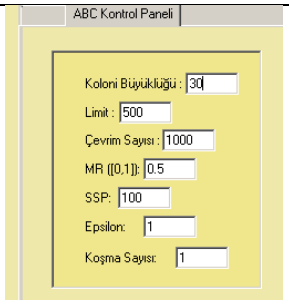
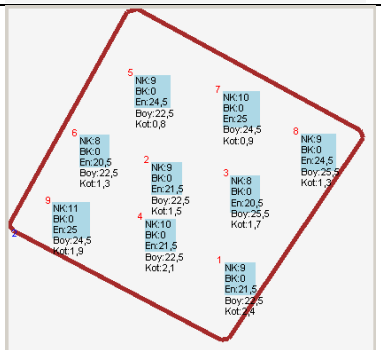
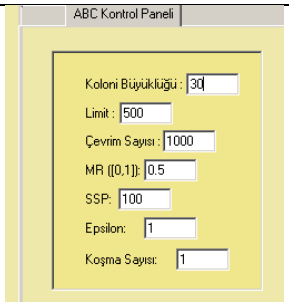
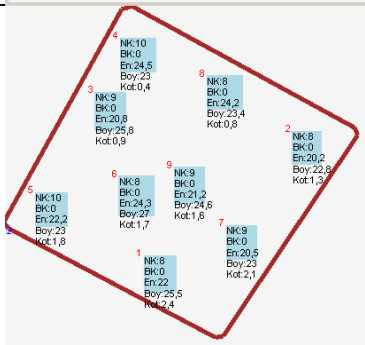
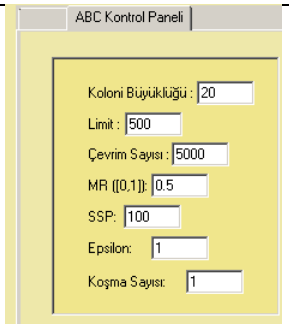
Üretilen vaziyet planlarından bazıları çizelge 4.10' da sunulmuştur.

Çizelge 4.10 : İzmir Torbalı için üretilen vaziyet planları.

İzmir Torbalı için üretilen vaziyet planları

Alternatifler	Süre	İhlal	Abc parametreleri (kontrol paneli)
1 	482 sn	var	

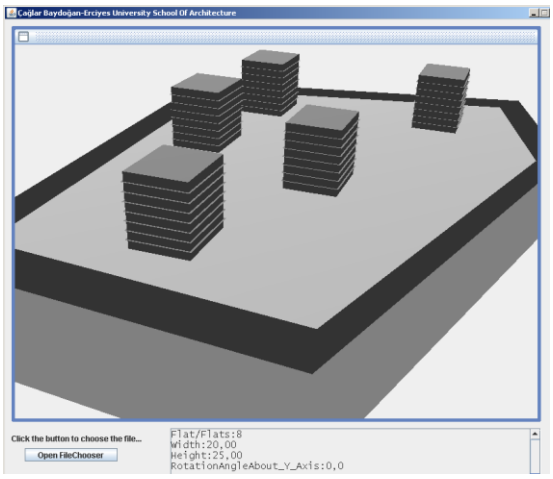
Çizelge 4.10 (devam) : İzmir Torbalı için üretilen vaziyet planları.

2		543 sn	var	
3		602 sn	var	
4		583 sn	var	
5		967 sn	var	

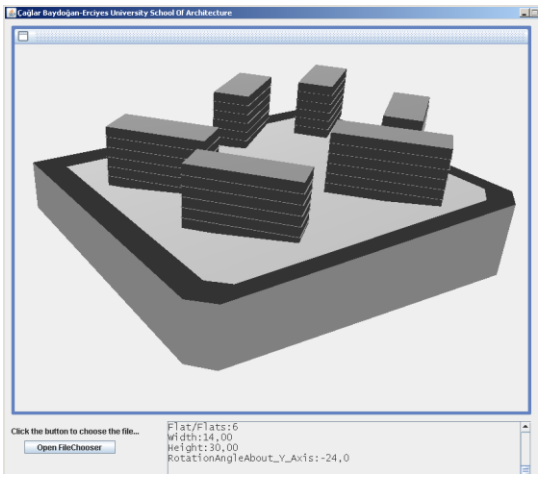
Farklı vaziyet planları için programın ürettiği vaziyet planı alternatifleri, süreleri, ayarları yukarıdaki çizelgelerde verilmiştir. Bu çizelgelerden anlaşılacağı üzere, bilgisayar programı ile üretilen vaziyet planı alternatifleri TOKİ'nin üretip, uyguladığı vaziyet planı çözümlerine benzemektedir. TİY'e uyum konusunda ise, üretilen vaziyet planı çözümlerinden bazıları uygun, bazıları ise uygun değildir. Fakat özellikle çevrim sayısı ve ABC parametreleri değiştirilerek uygun çözümlere

çizelgelerde gösterilen süreden biraz fazla zamanda ulaşmak genelde mümkün olmaktadır. TİY'e uymayan, hatalı çözümlerde ihlalin nereden kaynaklandığı belirlenebilmektedir. Program vasıtasıyla yapılan optimizasyonun, tamamen matematiksel veriler üzerinden yürüdüğünün unutulmaması, tasarımın bir tek matematiksel veriler üzerinden yapılamayacağının, bu programın tasarımcıya destek veren bir sisem olduğunun bir defa daha hatırlatılmasında fayda vardır.

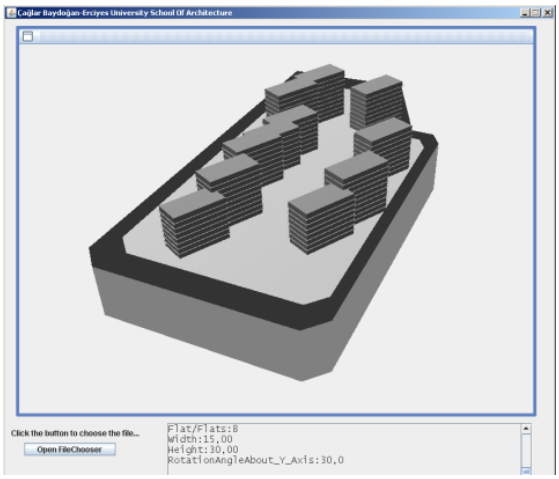
Giriş kısmında ele alınan, ayrıca çalışmanın hedeflerinden biri olan, programın oluşturulan vaziyet planlarını üç boyutlu gösterebilmesi konusu çalışma kapsamında değerlendirilmiştir. Java ortamında hazırlanan, Netbeans IDE 7.2 programı kullanılarak kodlanan bir program vasıtasıyla bu hedefe de ulaşılmıştır. Programın oluşturduğu çıkış dosyaları (.txt) java platformunda çalışan ayrı bir program vasıtasıyla üç boyutlu olarak oluşturulabilmektedir. Üç boyut için çalışma kapsamında oluşturulan program; ana programda oluşturulmuş vaziyet planında bulunan kütlelerin kat adet, büyüklük (en-boy), yerleşim için gereken koordinatları (x, y, z (merkez piksel noktası, kat adedi)) bilgilerini ve parselle ilgili sınır (komşu parsel-yol) bilgilerini alarak vaziyet planını 3 boyutlu oluşturmaktadır. Program bu bilgileri "currentfolder/konutyerlestirmeversion/konutyerlestirme/bin/debug/cikis" dosyasında oluşturduğu metin belgesinde (.txt) kayıt altına almaktadır. 3 boyuta ilişkin bilgiler "cikis1, cikis2, …" şeklinde oluşturulan txt dosyaları üç boyut programında okutularak alternatiflerin üç boyutlu görüntülerine ulaşmak mümkün olmaktadır. Aşağıda (şekil 4.54, 4.55, 4.56, 4.57, 4.58) çalışma kapsamında ve deneme sürecinde oluşturulan üç boyutlu vaziyet plan alternatiflerinin görülmesi mümkün olmaktadır.



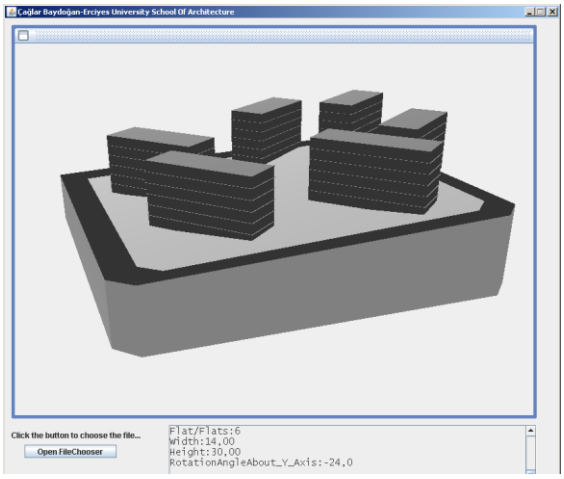
Şekil 4.54 : Üç boyutlu vaziyet planı örneği 1.



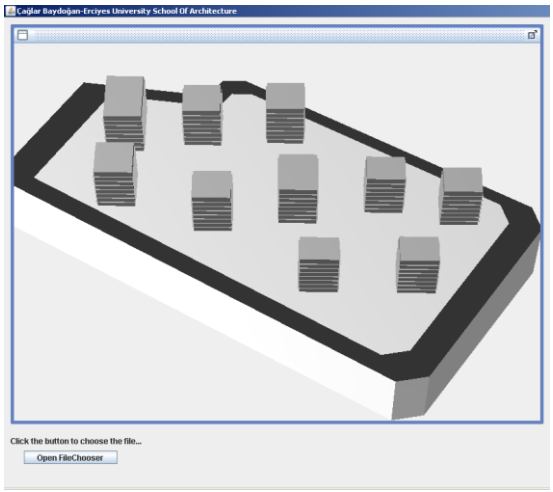
Şekil 4.55 : Üç boyutlu vaziyet planı örneği 2.



Şekil 4.56 : Üç boyutlu vaziyet planı örneği 3.



Şekil 4.57 : Üç boyutlu vaziyet planı örneği 4.



Şekil 4.58 : Üç boyutlu vaziyet planı örneği 5.

Çalışma kapsamında üretilen yazılımlar ile, konvansiyonel şekilde; el ile, çizim için paket bilgisayar programı, hesap makinesi, Tip İmar Yönetmeliği kitapçığı, vb. araçlar kullanılarak oluşturulan vaziyet planları arasında tasarım, yönetmeliğe uyum, yerleşim şekli, hesaplar açısından bariz bir fark yoktur. Baz alınan vaziyet planları bağlamında, üretilen vaziyet planları özellikle süre açısından, alternatif fazlalığı ve tasarımları üç boyutlu görmek açısından konvansiyel üretimden daha başarılı bulunabilir. İncelenen vaziyet planları özelinde (50 adet) genelde TOKİ'nin ürettiği toplu konut projelerine benzer üretimleri çalışma kapsamında oluşturulan bilgisayar programı gerçekleştirmiştir. Üç boyutlu ve alternatifli olarak bu üretimlerin etkin şekilde kullanılması tasarımcılar için, tasarım arayışını, daha iyi tasarımları olanaklı kılabilecektir.

5. SONUÇLAR VE ÖNERİLER

Yazılımın, tasarım problemi çözme süreci içerisinde, çalışmalara destek verebilecek yapay zeka olduğu örneklerle ortaya konulmaya çalışılmıştır. Yazılımın ürettiği çalışmaların konvansiyonel yöntemlerle üretilen çalışmalarla karşılaştırılması, yazılımın getirilerine işaret etmektedir. Zaman- hız, doğruluk- güvenilirlik, erişilebilirlik, kolay kullanılırlık, karşılaşılan problemlerin çözüm yöntemi, hesapların kontrolü, yazılımın konvansiyonel (bilgisayar yordamıyla veya elle üretilen) yollarla üretilen vaziyet planlarına karşı üstünlüğünün ortaya konmasında öne çıkan kavramlardır.

Üretilen yazılım çalışmanın başında hedeflenen amaçlara ulaşmıştır. TİY' leri günümüzde ve yakın geçmişte kent mekanlarının kötü şekilde değişmesine sebep olduğu için ve içindeki mevzuat açıkları nedeniyle eleştirilen, çoğu durumda değişik çözümlere olanak vermeyen ayrıca genelde doğru, yararlı kullanılamayan bir durumdadır. Yazılım, eleştirilen-tartışılan bir konu olan tip imar yönetmeliklerinin ortaya çıkardığı durumları tasarımcı lehine çevirmektedir. Daha sonraki kurallı, nesnel verilere dayalı yönetmelikler için de geliştirilebilir olması geleceğe dönük bir yol açmıştır. TİY, Türkiye'de belediyelerin hazırladığı imar planı yönetmeliklerinin hepsinde altlık, şablon başlangıç noktası olarak mevcuttur. Çalışma günümüzde hala yürürlükte olan sistemin hissedilen engellerinin azaltılması için başlangıçtır.

İmar parseli üzerinde, plan notları, yönetmeliklerin üzerinde olan kuralların ve bilgilerin ne sonuçlar doğurabileceğini öngörmek için üretilen yazılımın olanakları kullanılabilecektir.

Sonuç itibariyle geliştirilen program sayesinde;

- TİY e kolayca uyum
- Plancıların yoğunluk hesabı yaparken önkestirimde bulunabilmelerini kolaylaştırmak
- Karmaşık hesapların kolay yapılabilmesi

- Tasarıma başlarken sonuç ürüne yakın bir boyuttan çalışabilme olasılığı
- Hesapların doğruluğu,
- Birçok alternatifi hızlı bir biçimde görmek,
- Parsel için uygun alternatifleri saklamak, hesapları kaydedebilmek,
- Plan notları gibi değişiklikler için programa girilen kriterlerin-değerlerin manipülasyonu mümkündür.

5.1 Avantajlar-Dezavantajlar ve Yazılımın Projeksiyonu

Avantajlar: Yazılımın, ileriye dönük geliştirilebilme olanaklarının fazla olması, C# gibi güncel, birçok programlama diline çevrilmeye yatkın olan bir programlama dili kullanılması avantajlarından bazılarıdır. Asıl avantajlı olduğu durum kendi alanında özgün bir çalışma olmasıdır. Yönetmelikler, imar yasaları, hükümleri diğer bir deyişe kent mekanını etkiyecek kurallar üzerine, hesaplamalı tasarım yöntemleriyle geliştirilmiş akademik anlamda ve pratikte fazla çalışma bulunmaması nedeniyle; çalışmanın bu alana nasıl bir katkı koyduğunun ispatı niteliğindedir.

Kullanımı kolaydır. Tasarım, planlama konularıyla uğraşan uzman insanların zaten bildikleri terimlerin basit değişikliklerle yapıldığı bir ortamdır.

Asıl amaç olan farklı alternatiflerin görülmesi, daha iyi çözümlerin bulunabilmesi için bir araç olacaktır.

Dezavantajlar: Yazılım çok sabit kurallar dizisi olan TİY üzerine hazırlanmıştır. TİY üzerine karışık matematiksel işlemler, boyutlandırma, yerleştirme vb. işlemler yapan diğer bir anlatımla optimizasyon yapan bir yazılım üretilmiştir. Bu nedenle birçok kısıtlama ve parametrenin olduğu algoritma yazılımın yaratıcı yönünü engellemektedir. Algoritmanın sadece dikdörtgen bloklardan oluşan vaziyet planı alternatifleri üretmesi en büyük zaafıdır.

Yazılımın ileriye dönük projeksiyonları arasında; tip imar yönetmeliği çerçevesinde farklı geometride kütle alternatifleri üzerinden işlem yapması, farklı yaratıcı üretken sistemlerin program bünyesine dahil olması sayılabilir.

Tip imar yönetmeliği sınırlaması yaygın kullanımı engellemektedir. Programın daha sonra, şu an illerde ve/veya ilçelere yürürlükte olan imar yönetmeliklerine göre yeniden güncellenebilme olanağı sunması gerekmektedir.

Programın birçok kullanıcı tarafından test edilmesi, gelecek eleştiriler-öneriler ile geliştirilebilmesi erişilebilir olması ile ilgilidir. Programın açık kaynak kodlu, her kullanıcı tarafından geliştirilebilir ve geliştirmelerin paylaşılması ileriye dönük projeksiyonlar arasında yer almaktadır. Ayrıca yazılımın enteraktif olarak internet üzerinden kullanıcıların isteklerine cevap verecek bir duruma getirilmesi, diğer internet siteleri ile ortak çalışması mümkün olabilecektir. Örneğin; mimari proje bedel hesabı yapan bir siteyle ortak çalışması, parsel köşe koordinat bilgilerinin coğrafi bilgi sistemleri aracılığıyla belediye, tapu kadastro vb kurumlar tarafından sağlanması, vb. gibi disiplinler arası ortaklıklar kurulması programın geliştirilmesine katkı sağlayacaktır.

5.2 Öneriler

Çalışmanın şuan geldiği nokta bakımından değerlendirilmesi yanlış olur. Çalışma gelecekte gelinen aşamadan çok daha ileriye gidecek potansiyeli barındırmaktadır. Çalışmanın Türkiye şartlarında üstlenmiş olduğu görev belirlenmiş bir alanda, belli tip bir problemin çözülme ait optimizasyon işidir. Farklı boyutlarda, farklı şartlarda fakat yöntemi benzer olan gerçek hayat problemidir. Fakat çalışmanın üstlenmiş olduğu görev aslında bukadarla sınırlı değildir. Program yani algoritma olası bir çözümü bulmak için geniş, sonsuz alternatifli çözüm uzayında tarama yapar. Olası çözümlerin içerisinden en iyi olanı bulup çıkarmaya çalışır. Fakat tasarımın doğasında örneğin, mimari bir çözümde, ya da farklı kentsel alan problemi çözümünde “tek doğru” denilen bir çözüm yoktur. Dolayısıyla programın bir sonraki aşamada tasarım olasılıklarını değerlendirip, bulduğu olasılıkları derecelendirip kullanıcıya ard arda sunması beklentiler ve öneriler içerisinde yer alabilir.

Daha farklı bir öneri ise; programın üretken yönünün ön plana çıkarılması gereğidir. Programın, gerçek hayat problemi olan, vaziyet planı ölçeğinde; TİY e uygun yerleştirme, hesaplama, yapan program olarak adlandırılması yerine vaziyet planı tasarımı yapan üretken, tasarıma yatkın bir sistem olarak adlandırılması ve bu alandaki potansiyel için tanımlanan şekilde kullanılması daha iyi sonuçlar verecektir.

Tasarımcıların ve şehir bölge plancılarının programı kullanmaları ileriye dönük gelişme adına veya geri besleme adına iyi olacaktır. Programın yaygınlaşması gelişimini olumlu yönde etkileyecektir.

Tip imar yönetmeliği taban alınan bu çalışma, belediyelerin aktif olarak kullandıkları yönetmeliklere entegrasyonu sağlamalıdır.

Belediyelerle iş ortaklıkları yapılmalı, uygun tasarıma, daha iyi kentsel mekanlara ulaşmak, kent konforu ve estetiğini arttırmak için programı bu yönde geliştirmek ana hedefler arasında yer almalıdır.

Çok kullanılan paket çizim programları ile işbirliği yapmak da diğer hedeflerden biri olmalıdır.

KAYNAKLAR

Acar, E. (2006). Yerleşim ve Bina Tasarımı, *Planlama Dergisi, 2006/4,* 16, Ankara: Tmmob Şehir Plancıları Odası. Erişim adresi: http://www.spo.org.tr/resimler/ekler/8c41c4a18675a74_ek.pdf

Akipek, F. Ö.,(2004). Bilgisayar Teknolojilerinin Mimarlıkta Tasarımı Geliştirme Amaçlı Kullanımları (Yayınlanmamış Doktora Tezi). İstanbul: Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü.

Allahverdi, N. (2002). *Uzman Sistemler: Bir Yapay ZekaUygulaması*. İstanbul: Atlas.

Asimov, I. (1991). *I Robot*. New York: Bantam.

Aydemir, S. E. Aydemir Ş., Ökten, N., Öksüz, A. M., Sancar, C., Özyaba, M. (2004). *Kentsel Alanların Planlanması ve Tasarımı.* Trabzon: Akademi.

Baykan, C. A. (2001). Mimarlık eğitiminde bilgisayarla tasarım (CAD in architectural education). *TMMOB Mimarlar Odası Ankara Şubesi Mimarlıkta Cad Kullanımı Sempozyumu*, Ankara. Erişim adresi: http://www.metu.edu.tr/~baykan/publications-pdf/tmmob23kasim.pdf

Baykan, C. A. (1984). *Heuristic methods for structuring architectural design problems* (Unpublished report). Pittsburgh: Department of Architecture, Carnegie Mellon University.

Baykan, C. A., Eastman, C. M. (1984). *An expert system for window performance analysis* (Unpublished report). Pittsburgh: Institute of Building Science, Carnegie Mellon University.

Bayraktar N., Kaplan, H., Tekel, A. (2006). Kentsel Tasarım Yarışma Süreçleri ve Uygulama Örnekleri Üzerinden Kentsel Tasarım Tartışmaları ,. *Mimarlık Dergisi, 329, Mayıs-Haziran,* 14-19. Erişim adresi: http://www.mimarlikdergisi.com/index.cfm?sayfa=mimarlik&DergiSayi=42&RecID=1049

Bergey P.K. and Ragsdale C. (2005). Modified Differential Evolution: A Greedy Random Strategy For Genetic Recombination. *Omega, The International Journal of Management Science*, *33, 3*, 255-265. doi: 10.1016/j.omega.2004.04.009

Biddulph, M. (2007). *Introduction To Residential Layout***,** Oxford: Elsevier.

Bridges, A. (2007). Computational Support For Early Stage Architectural Design. *Proceedings of CIB 24th W78 Conference Maribor*, 755-761. Erişim adresi: http://itc.scix.net/data/works/att/w78-2007-079-049-Bridges-a.pdf

Bonabeau, E., Dorigo, M. and Theraulaz, G. (1999). *Swarm Intelligence: From Natural to Artificial Systems.* New York: Oxford University Press.

Bozdemir, M., Mendi, F. (2005). Yapay Zeka Destekli Sistematik Tasarım İçin Bilgi Yönetim Sistem Mimarisi, *Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi. 20, 2,* 267-274. Erişim adresi: http://www.mmfdergi.gazi.edu.tr/2005_2/267-274.pdf

Büyük Larousse Ansiklopedisi, (1986), 8. Baskı, *Cilt.18,* 121. Librairie Larousse, İstanbul: Interpress.

Capek, K. (2001). *R.U.R (Rossum's Universal Robots).* New York: Dover Publications.

Cheng, V., Steemers, K., Montavon, M. And Compagnon, R. (2006). Urban form, density-and solar potential, *PLEA2006 - The 23rd Conference on Passive and Low Energy-Architecture, Switzerland:Geneva 6-8 September, 1*, 701-706.. Erişim adresi: http://www.usp.br/fau/cursos/graduacao/arq_urbanismo/disciplinas/aut0221/Material_de_Apoio/Cidades/Cheng_Plea2006_Solar_Density.pdf

Chu, K. (2006). Metaphysics of Genetic Architecture and Computation, *Architectural Design*, *76, 4, July/August*, 38-45. Erişim adresi: http://onlinelibrary.wiley.com/doi/10.1002/ad.292/pdf

Çağdaş, G. (2002). Mimarlıkta Bilişim ve Eğitimi. *Yapı (Mimarlıkta Bilişim eki) 252,* 3-8.

Çağdaş, G. (2004). Enformasyon Teknolojilerindeki Evrimsel Sürecin Mimari Tasarım Eğitimine Yansımaları, Stüdyo, *Tasarım Kuram Eleştiri Dergisi*, *2, Güz,* 04-05.

Çolakoğlu, B., Yazar, T. (2007). An Innovatıve Design Education Approach: Computatıonal Desıgn Teachıng For Archıtecture, *Metu Jfa*, *2007/2*, 24:2. Erişim adresi: http://jfa.arch.metu.edu.tr/archive/0258-5316/2007/cilt24/sayi_2/159-168.pdf

Çubuk, M. (1992). *1.Kentsel Tasarım Ve Uygulamalar Sempozyumu,* 23-24 Mayıs İstanbul: Mimar Sinan Üniversitesi Mimarlık Fakültesi.

Çubuk, M. (1997). Kentsel Tasarım Formasyonu ve Eğitimi Sunuş Bildirisi ve Sonuç Metni. *VIII. Kentsel Tasarım ve Uygulamalar Sempozyumu*, *15-16 Mayıs*. İstanbul: MSÜ Mim. Fak.

Dalkılıç G., Türkmen F. *(*2002). Karınca Kolonisi Optimizasyonu, (Ed: Sevilgen, F. E., ve Sadıkoğlu, H.,), *YPBS2002 – Yüksek Performanslı Bilişim Sempozyumu, Ekim.* Kocaeli: GYTE Matbaası.

Dantzig, G. B. (2002). Linear Programming. Dantzig Subject classification: Professional: comments on. *Area of review: Anniversary Issue (Special), Operations Research, 50, 1, January–February, 1526-5463 electronic ISSN. 2002,* 42-47. Erişim adresi: https://www.cs.duke.edu/courses/spring07/cps296.2/papers/LinearProgramming_article.pdf

Duda, L., Erman D., Hayes-Roth F., Lesser V. R., Reddy, D. R. (1979). The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty. *ACM Computing Surveys (CSUR), 12, 2,* 213-253.New York: ACM.
doi: 10.1145/356810.356816

Duyguluer, F. (1989). *İmar Mevzuatı'nın Cumhuriyet Dönemi Mimarlığına ve Şehir Planlamasına Etkileri*. Ankara : TBMM Kültür, Sanat ve Yayın Kurulu Yayınları, 38.

Eastman, C. M. (1970). On the Analysis of Intuitive Design Processes, G. Moore *(*Ed.), *Emerging Methods in Environmental Design and planning, proceedings of the Design Methods Group First International Conference Cambridge*, *Massachusetts*, 21-37. Cambridge: M.I.T.

Elliiki, **O. B. (**1996). *Şehirsel Tasarıma Destek Veren Bir Bilgisayar Sisteminin Oluşturulması*, *Bilgisayar Destekli Şehirsel Tasarım* (Yayınlanmamış, yüksek lisans tezi). İstanbul: İTÜ, Fen Bilimleri Enstitüsü.

Frazer, J. (1995). *An Evolutinary Architecture*. Londra: Architectural Association Publications.

Garassini S. (2000). Between Nature and Artifice. Domus, *Domus M. December 2007, 2,*192-195. Erişim adresi:
http://www.flong.com/texts/press/domus_2007

Gero, J. S. (1996). Creativity, Emergence and Evolution in Design: Concepts and Framework. *Knowledge-Based Systems, 9(7),* 435-448. Erişim adresi:
http://mason.gmu.edu/~jgero/publications/1996/96GeroKBS.pdf

Gero, J.S. (1999). Constructive Memory in Design Thinking. Goldschmidt, G. and Porter, W. (eds.), *Design Thinking Research Symposium: Design Representation,* 29-35. Cambridge: MIT.

Goldberg, D. E., Lingle, R. (1985). Alleles, loci and the traveling salesman problem. Grefenstette, J. J. (ed.). *Proceedings of an International Conference on Genetic Algorithms and Their Applications,* 15–19. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning,* MA :Addison-Wesley. Erişim adresi:
http://link.springer.com/content/pdf/10.1023%2FA%3A1022602019183.pdf

Grüter, C., Farina, M. (2009). The Honeybee Waggle Dance: Can We Follow The Steps?, *Trends in Ecology and Evolution*, *24-5,* 242-247.
doi:10.1016/j.tree.2008.12.007

Gürer, B., Bayazıt N. (1976). *Mimarlıkta Sistemli Tasarlama Metotları* (Yayınlanmamış Ders Notu). İstanbul: İTÜ Mimarlık Fakültesi.

Gürler, M. (1983). *İmar Planları ve Uygulama Tekniği***.** Ankara: TMMOB Harita ve Kadastro Mühendisleri Odası Yayını.

Günay, B. (2006). Şehircilik-planlama-tasarlama-mimarlık-peyzaj, *Planlama*, *2006/4*, 38. Ankara: TMMOB Sehir Plancıları Odası. Erişim adresi:
http://www.spo.org.tr/resimler/ekler/3fe9d84310e77f1_ek.pdf

Harel, D. (1992). *Algorithmics: the spirit of computing (2nd ed.)*. Reading, MA: Addison Wesley.

Hayes, R. (1983). *Building Expert Systems,* 5-14. Reading: Mass Addison -Wesley Publishing.

Hensel, M. (2006a). Computing Self-Organisation: Environmentally Sensitive Growth, Modelling*, Architectural Design, 76, Issue 2*, *March/April*, 12-17.
doi: 10.1002/ad.235.
http://onlinelibrary.wiley.com/doi/10.1002/ad.235/pdf

İnam, A. (2001). İnsanın Mekanik Kopyası Yapılabilir mi?, *Bilim ve Teknik*, *sayı: 409, Aralık 2001*, s. 46.

İmar Kanunu: 3194 (1985). *Resmi Gazete* (Kabul Tarihi: 03/05/1985).

Jones, J. C. (1970). *Design Methods: seeds of human futures.* New York: Wiley-Interscience.

Kalay, Y. E. (2004). *Architecture's New Media: Principles, Theories, and Methods of Computer-Aided Design*, Cambridge: MIT Press.

Karaboğa, D. (2005). An idea based on honey bee swarm for numerical optimization. *Technical Report TR06.* Kayseri: Erciyes University, Engineering Faculty, Computer Engineering Department.

Karaboğa, D., Akay, B. B. (2007)**.** Artificial Bee Colony Algorithm on Training Artificial Neural Networks, Signal Processing and Communications Applications*. SIU 2007, IEEE 15 th. 11-13 June*, 1- 4.
doi: 10.1109/SIU.2007.4298679.

Karaboğa, D., Akay, B. B. (2010). *Yapay Zeka Optimizasyon Algoritmaları*, Yüksek lisans dersi 2010-2011 güz yarıyılı (Yayınlanmamış ders notları), Kayseri: Erciyes Üniversitesi, Mühendislik Fakültesi.

Karaboğa, D, Akay, B. B. (2008). On the Performance of Artificial Bee Colony (ABC) Algorithm, *Applied Soft Computing*, *vol.8,* 687-697.

Karaboğa, D. (2004). *Yapay Zeka Optimizasyon Algoritmaları*, İstanbul: Atlas Yayın Dağıtım.

Karaman, A. (2003). Dönüşüm Projelerinde Kentsel Tasarımın Rolü, Kentsel Yenileşme ve Kentsel Tasarım. *Uluslararası 14. Kentsel Tasarım ve Uygulamalar Sempozyumu Bildiriler Kitabı*. İstanbul: MSÜ Yay.

Keleş, R. (1996). *Kentleşme Politikası***.** (3. Baskı), ISBN 975-533053-4, Ankara: İmge Kitapevi.

Kernighan, B., Ritchie, D. (1978). *The C Programming Language*, ISBN 0-13-110163-3.NJ : Prentice-Hall, Englewood Cliffs.

Kolarevic, B. (2000). Digital Morphogenesis and Computational Architectures in J.R. Kos, A.P. Borde ve D.R. Barros (Eds.), *SIGRADI 2000 Proceedings, Rio de Janeiro: Constructing the Digital Space*, 1-6.

Kolarevic, B. (2003). *Architecture in The Digital Age: Design and Manufacture*, London: Spon Press.

Kristl, Z. And Krainer, A. (2001). Energy evaluation of urban structure and dimensioning-of building site using ISO-shadow method. *Solar Energy, 70,* 23-34.

Lucic, P., Teodorovic, D. (2002). Transportation Modelling: An Artificial Life Approach. *14. IEEE International Conference on Tools with Artificial Intelligent, ICTAI'02*, 216-223.
doi : 10.1109/TAI.2002.1180807

McCarthy, J. (2004). *What is Artificial Intelligence.* CS Department, Stanford University Computer Science Department.

Mitchell, W. J. (1985). Formal Represantations: A Foundation for Computer Aided Architectural Design, *Environment and Planning B: Planning and Design, 5,* 5-18.
doi:10.1068/b130133

Ousterhoud, J. K. (1998). Scripting: Higher Level Programming For The 21st Century, *Ieee Computer*. *Volume:31 , Issue: 3,* 23- 30.
doi: 10.1109/2.660187

Oxman, R., (2006). Theory and design in the first digital age. in *The international Journal of Design Studies*, *Elsevier, Vol 27 No 3*, 229-265. doi:10.1016/j.destud.2005.11.002

Öke A., Bayazıt N., İnceoğlu M., Tapan M. (1978). Mimari Tasarlama Ders Notları. İstanbul: İTÜ Mimarlık Fakültesi.

Pearlmutter, D. (1998). Street canyon geometry and microclimate: designing for urban-comfort under arid conditions. *Proceedings of PLEA '98*, *Lisbon,* 163-166.

Penrose, R. (1989). *Bilgisayar ve Zeka Kralın Yeni Usu*. İstanbul: TÜBİTAK yayınları.

Pirim, H. (2006). Yapay Zeka. *Journal of Yasar University,1(1), İzmir: Endüstri Mühendisliği Bölümü, Mühendislik Mimarlık Fakültesi, Yaşar Üniversitesi,* 81-93.

Piringer, M., Joffre, S. (2004). The urban surface energy budget and mixing height in-european cities: data, models and challenges for urban meterology and air quality. *Final Report of Working Group 2 of COST-715 Action*, *Central Institute for-Meterology and Geodynamics, Vienna***,** 2-198.
doi:10.1007/s10546-007-9170-0

Poyrazoğlu O., Gülesin M., Kayır Y. (1997). Uzman Sistem Tekniklerine Dayalı Alışılmamış İmalat Yöntemlerinin Tespiti. *Makina Tasarım ve İmalatları Yönetim* Konferansı, *Gazi Üniversitesi*, 80-89.

Rowe, P. G. (1987). *Design Thinking*. Cambridge: MA: MIT Press.

Russell S., Norvig P. (2003). *Artificial Intelligence A Modern Approach.* Prentice Hall.

Schank, R. C., Childers, P. (1984). *The cognitive computer on language, learning, and artificial intelligence.* United States:Addison-Wesley Pub. Co. Inc., Reading, MA.

Settles, M. *(*2005). *An Introduction to Particle Swarm Optimization*. USA: University of Idaho Computer Science Department.

Shortliffe, E. H. (1976). *Computer-based medical consultation*, *MYCIN*. New York: American EIsevier.

Sinanoğlu, C. ve Börklü, H. R. (2002). Montaj Sırası Planlama Sistemlerinde Bir Optimizasyon Yaklaşımı, *ODTÜ Makina Tasarım ve İmalat Dergisi*, *Cilt: 4, Sayı: 3, Mayıs, s.,* 155-172.

Stroustrup, B. (1994). *The Design and Evolution of C++*. Addison-Wesley, MA, ISBN 0-201-54330-3.

Stroustrup, B. (1980b). A Set of C Classes for Co−routine Style Programming. *Bell Laboratories Computer Science Technical Report CSTR−90*. November, 1980.

Szalapaj, P. (2001). CAD *Principles for Architectural Design, Analytical Approaches to Computational Representation of Architectural Form*. MA: Architectural Press.

Tekeli, İ. (1985). Ders Notları, Ankara: ODTÜ.

Teodorovic, D., Lucic P., Markovic, G., Dell' Orco M. (2006). Bee Colony Optimization: Principles and Applications. *8th Seminar On Neural Network Applications in Electrical Engineering IEEE NEUREL* 2006. *Belgrade, Serbia & Montenegro*, 151-156. doi:10.1109/NEUREL.2006.341200

Terzidis, K. (2006). *Algorithmic architecture*. Oxford:.Architectural Press.

Topçuoğlu, A. (2001). Yapay Zeka. *Bilim ve Teknik, 409*, 38-45.

Turan, B. O. (2009). *Dijital Tasarım Sürecinin Geleneksel Tasarım Stüdyosuna Etkileri*. İstanbul: YTÜ, Fen Bilimleri Enstitüsü.

Turban, E. (1990). *Decision Support and Expert Systems*. 2nd ed., New York: Macmillan Pub. Comp.

Url-1 <http://www.tdk.gov.tr>, erişim tarihi 03.06.2010.

Url-2 <http://www.toki.gov.tr**>**, erişim tarihi 03.06.2010.

Url-3 <http://www.webster.com**>**, erişim tarihi 03.06.2010.

Url-4 <http://websitem.gazi.edu.tr/rborklu/ DosyaIndir?DosyaNo =ca5d8fa5bde5f576c23a76308394b855>, erişim tarihi 18.04.2010.

Url-5 <http://tangible.media.mit.edu/projects/>, erişim tarihi 03.06.2010.

Url-6 <http://www.cs.ucl.ac.uk/staff/p.bentley/wc3paper.html**>,** erişim tarihi 18.04.2010.

Url-7 <http://msdn.microsoft.com/vstudio/nextgen/technology/csharpintro.asp>, erişim tarihi 03.06.2010.

Url-8 <http://ercangunbay.com/dersler/ CSharp.pdf>, erişim tarihi 18.04.2010.

Url-9 <http://msdn.microsoft.com/vstudio>, erişim tarihi 06.05.2010.

Url-10 <http://www.microsoft.com/> erişim tarihi 29.05.2010.

Url-11 <http://os.kolokyum.com/osfiles/ 2009-06/ 090626_ TOKI_ Kayabasi_ Bolgesi_ Yarismasi _ Kolokyum_ Desifresi.pdf> erişim tarihi 29.05.2010.

Url-12 <http://www.abc.net.au/arts/today/stories/s248345.htm>, erişim tarihi 03.06.2010.

Url-13 <http://www.metu.edu.tr/~baykan/publicationspdf/tmmob23kasim.pdf - Mimarlık Eğitiminde Bilgisayarla Tasarım Can A. Baykan>, erişim tarihi 03.06.2010.

Url-14 <http://www.generativeart.com/papers2005/27.NicolettaSala2005.htm>, erişim tarihi 10.03.2010.

Url-15 <http://bibliotecnica.upc.es/bib200/fractals/galeria.htm>, erişim tarihi 03.06.2010.

Url-16 <http://translate.google.com/translate?hl=en&sl=zh-CN&u=http://scholar.ilib.cn/Abstract.aspx%3FA%3Dcsgh200502008&sa=X&oi=translate&resnum=4&ct=result&prev=/search%3Fq%3Dbovill,%2Bc.fractal%2Bgeometry%26hl%3Den%26lr%3D%26sa%3DG>, erişim tarihi 18.03.2010.

Url-17 <http://www.bot.yildiz.edu.tr/bom/_frames/_displaythesis.php? changeval =23>, erişim tarihi 11.07.2010.

Url-18 <http://www.dexigner.com/jump/news9624.html>, erişim tarihi 03.06.2010.

Url-19 <http://www.archinf.tu-cottbus.de>, erişim tarihi 03.06.2010.

Url-20 <160_massgeschneidert/de>, erişim tarihi 10.03.2010.

Url-21 <http://www.archinf.tu-cottbus.de>, erişim tarihi 03.06.2010.

Url-22 <http://www.cc.utah.edu/~asn8200/rapid.html>, erişim tarihi 03.06.2010.

Url-23 <http://www.cs.hut.fi/~ado/rp/rp.html>, erişim tarihi 10.03.2010.

Url-24 <http://www.mit.edu>, erişim tarihi 03.06.2010.

Url-25 <http://www.3Dsystems.com>, erişim tarihi 03.06.2010.

Url-26 <http://www.glform.com>, erişim tarihi 03.06.2010.

Url-27 <http://www.oosterhuis.nl>, erişim tarihi 03.06.2010.

Url-28 <http://www.unstudio.com>, erişim tarihi 03.06.2010.

Url-29 <http://wikipedia.org/>, erişim tarihi 03.06.2010.

Url-30 <http://www.cs.ucl.ac.uk/staff/p.bentley/wc3paper.html>, erişim tarihi 03.06.2010.

Url-31 <http://journal.yasar.edu.tr/wp-content/uploads/2011/07/no1_vol1_07_ harun_pirim.pdf>, erişim tarihi 03.06.2010.

Url-32 <http://old.mo.org.tr/mimarlikdergisi/index.cfm?sayfa=mimarlik&Dergi Sayi =42&RecID=1049>, erişim tarihi 10.12.2009.

Url-33 <http://www.balikesir.edu.tr/~bauopt/index.php?sayfa=yayinlar>**,** erişim tarihi 19.09.2010.

Url-34 <http://iris.gmu.edu/~khoffman/papers/newcomb1.html>, erişim tarihi 19.09.2010.

Url-35 <http://citeseerx.ist.psu.edu/viewdoc/download?doi= 10.1.1.33.5371 &rep=rep1&type=pdf>**,** erişim tarihi 19.09.2010.

Url-36 <www2.research.att.com/~bs/hopl2.pdf>**,** erişim tarihi 22.09.2010.

Url-37 <http://msdn.microsoft.com/vstudio/nextgen/technology/ csharpintro.asp>, Anders Hejlsberg & Scott Wiltamuth, *C# Language Reference, Microsoft Corp, 2000.* erişim tarihi 22.09.2010.

Url-38 <http://www.javaworld.com/javaworld/jw-11-2000/jw-1122-csharp.html>**,** Mark Johnson,*C#: A language alternative or just J--?,*Java World, November/December 2000, erişim tarihi 22.09.2010.

Url-39 <http://www.oreillynet.com/pub/a/dotnet/2001/06/07/csharp_java.html>**,** Budi Kurniawan, *Comparing C# and Java,* Published on The O'Reilly Network, erişim tarihi 22.09.2010

Wolfram, S. (2006). How Do Simple Programs Behave?. *Architectural Design*, 76 *Issue 4 July/August*, 34-37.

Yakeley, M., (2000). Digitally Mediated Design: Using Computer Programming to Develop Personal Design Process. (Ph.D. Dissertation). MIT: Department of Architecture,

Yaman, R. (2005)**.** Optimizasyon. Balıkesir Üniversitesi Endüstri Mühendisliği, Mühendislik-Mimarlık Fakültesi Doğrusal olmayan Optimizasyon (Yüksek Lisans Dersi yayınlanmamış ders notları), Balıkesir.

Yavuz, F., Keleş R., Geray C. (1978). *Şehircilik (Sorunlar-Uygulama- Politika).* Ankara: AÜ, SBF, Yayın 415.

Yazdani, M. (1986). *Artificial intelligence: Principles and applications.* New York: NY: Chapman and Hall, Ltd..

EKLER

EK A : TOKİ' nin ürettiği çalışma kapsamında incelenen vaziyet planlarından örnekler**.**

EK B : Programın C# programında oluşturulmuş kaynak kodları.

EK C : Program çıktıları.

EK D : Bilgisayar programı CD'si.

EK E : Farklı yapı nizamları, yapı-yol durumu şemaları.

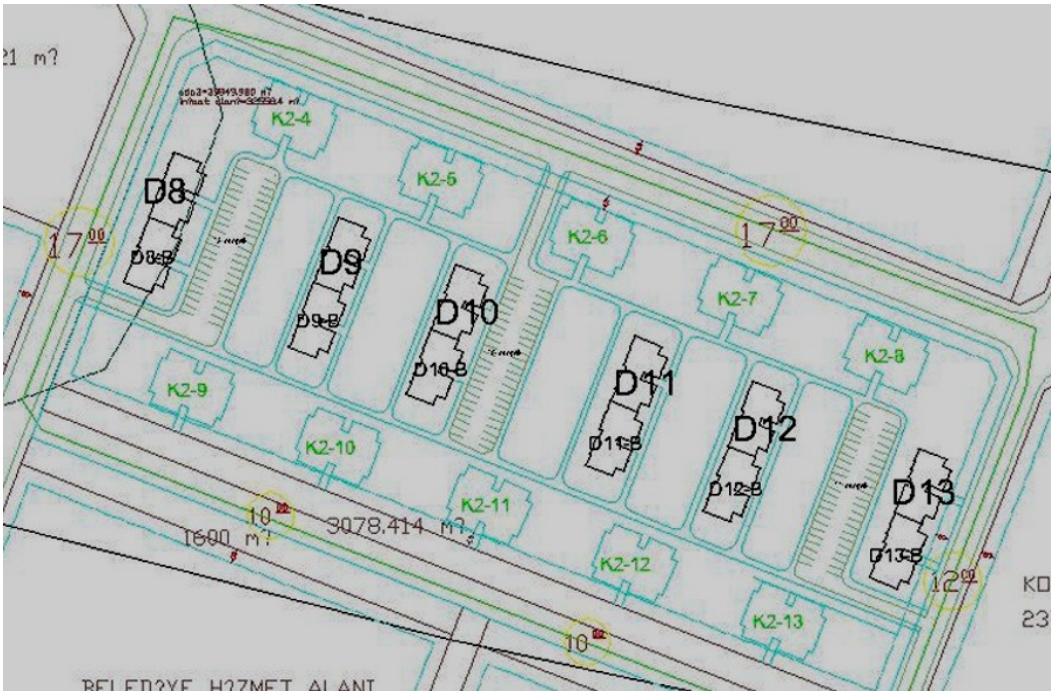
EK A.: TOKİ' nin ürettiği çalışma kapsamında incelenen vaziyet planlarından örnekler (bu örneklerden bazıları programın yeniden üretimini gerçekleştirdiği vaziyet planlarındandır.)



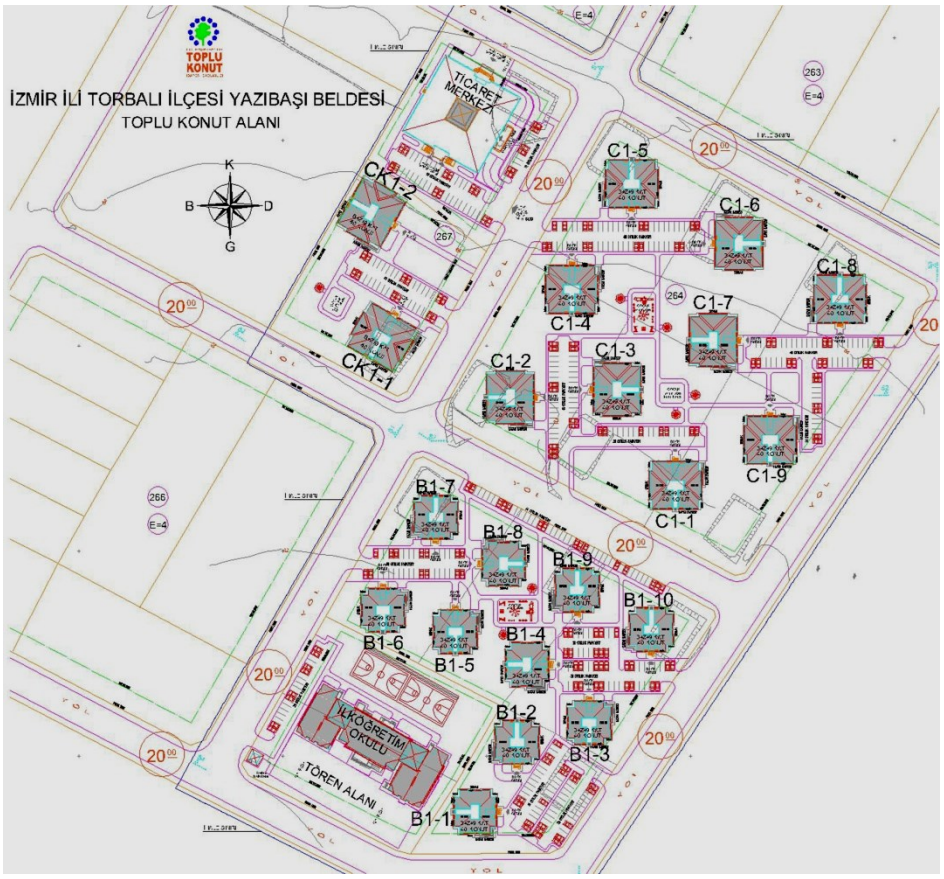
Şekil A.1 : TOKİ, Eskişehir, Tepebaşı, Aşağısöğütönü, vaziyet planı 2.



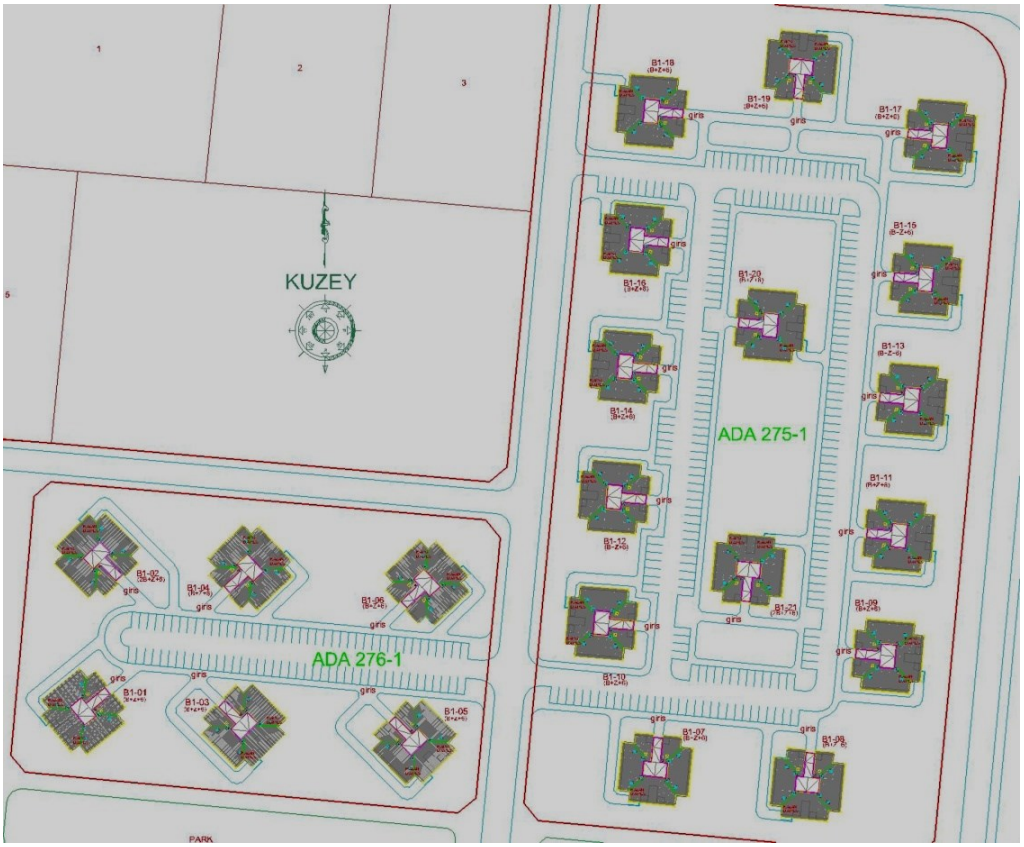
Şekil A.2 : TOKİ,Eskişehir, Çankaya, Vadişehir, 2. Etap.



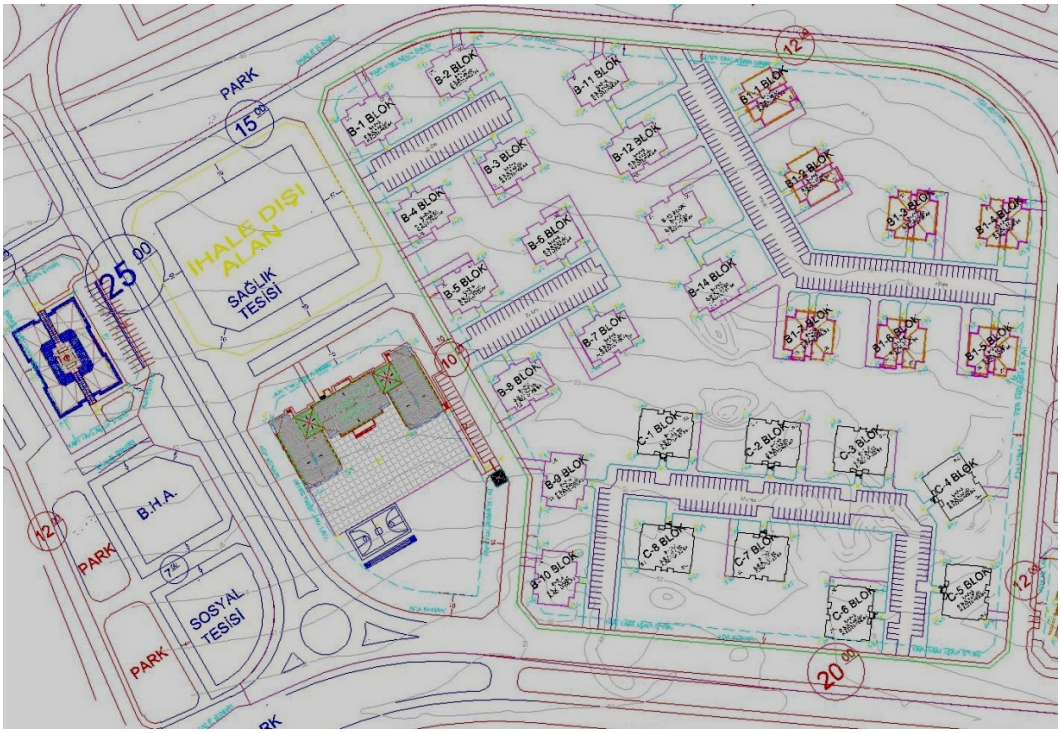
Şekil A.3 : TOKİ, Eskişehir, Çankaya, Vadişehir, 3. Etap, vaziyet 1.



Şekil A.4 : TOKİ, İzmir, Torbalı.



Şekil A.5 : TOKİ, Konya, Ereğli, alt gelir grubu.



Şekil A. 6 : TOKİ, Manisa, Akhisar.

EK B : Programın C#, .net ortamında oluşturulmuş kaynak kodları.

Program C# da oluşturulmuştur. C# da işleyiş classlar, formlar üzerindendir. Algoritmayı bu düzenle fakat eksiltilmiş olarak aşağıda görmek mümkündür. Algoritmanın genelde ABC (yapay arı kolonisi algoritmasıyla ilgili kısmi verilmiştir

Form 1.cs

```
namespace konutyerlestirme
{   public partial class Form1 : Form
   {     public Form1()
        {          InitializeComponent();       }
        int[] xKKisitlariMx;
        int[] yKKisitlariMx;
        double[] bEnKisitlariMx;
        double[] bBoyKisitlariMx;
        int[] bKatKisitlariMx;
        int[] xKKisitlariMn;
        int[] yKKisitlariMn;
        double[] bEnKisitlariMn;
        double[] bBoyKisitlariMn;
        int[] bKatKisitlariMn;
        int[] bAciKisitlari;
        public PointF[] parselP;
        PointF[] parseltasP;
        PointF[] parseltasP2;
        TextBox[] parselt;
        RadioButton[] komsuBilgiRB;
        RadioButton[] kotBilgiRB;
……….
          for (int i = 0; i < parselP.Length; i++)
          {   if (komsuBilgiRB[2 * i].Checked)
             { PKomsuBilgileri[i] = 1; }
             else if (komsuBilgiRB[2 * i+1].Checked)
             { PKomsuBilgileri[i] = 0; }
             if (kotBilgiRB[i].Checked)
             { kotBilgi = i; }          }
          kot1 = Convert.ToInt32(textBox4.Text);
          kot2 = Convert.ToInt32(textBox7.Text);
          //kotDegeri = Convert.ToDouble(textBox10.Text);
          //istenen yol için orta nokta pikseli ve kot değeri belirleme
          PointF merkezpd = new PointF();
          if (kotBilgi != parselP.Length - 1)
          {     merkezpd.X = ((float)(parselP[kotBilgi].X + parselP[kotBilgi + 1].X) / 2);
                merkezpd.Y = ((float)(parselP[kotBilgi].Y + parselP[kotBilgi + 1].Y) / 2);
          else
             merkezpd.X = ((float)(parselP[0].X + parselP[parselP.Length - 1].X) / 2);
             merkezpd.Y = ((float)(parselP[0].Y + parselP[parselP.Length - 1].Y) / 2);
          }
          double merkezDKot=0;
          Rectangle rcEB = metotlar.minBoundingBoxHesapla(parselP);
          int cS = comboBox1.SelectedIndex;
          if (cS == 0)
          {          merkezDKot = (merkezpd.Y / rcEB.Height) * (kot2 - kot1) + kot1; }
          else if (cS == 1)
          {   merkezDKot = (merkezpd.Y / rcEB.Height) * (kot1 - kot2) + kot2;          }
          else if (cS == 2)
          {  merkezDKot = (merkezpd.X / rcEB.Width) * (kot2 - kot1) + kot1;          }
          else if (cS == 3)
          {             merkezDKot = (merkezpd.X / rcEB.Width) * (kot1 - kot2) + kot2;
          koloni = Convert.ToInt32(textBox5.Text);
```

```
        int kaynakS = koloni / 2;
        limit = Convert.ToInt32(textBox8.Text);
        iterasyon = Convert.ToInt32(textBox6.Text);
        MR = Convert.ToDouble(textBox9.Text);
        SSP = Convert.ToInt32(textBox17.Text);
        epsilon = Convert.ToDouble(textBox18.Text);

        //Directory.CreateDirectory(Environment.CurrentDirectory + @"\çıkış");

        double ATabAalan = Convert.ToDouble(textBox14.Text);
        if (ATabAalan > TTabA)
        { MessageBox.Show("Maksimum taban alandan daha küçük bir değer girmelisiniz!"); }
        else
        { TTabA = ATabAalan; }
        //ABC
        ari[] arilar1 = new ari[kaynakS];
        for (int k = 0; k < kaynakS; k++)
        {           ari ari1 = (ari) metotlar.baslagicAtamasiYap(parselP,rn, xKKisitlariMn, xKKisitlariMx,
yKKisitlariMn, yKKisitlariMx, bEnKisitlariMn, bEnKisitlariMx, bBoyKisitlariMn, bBoyKisitlariMx,
bAciKisitlari, bKatKisitlariMn, bKatKisitlariMx).Clone();
          double g1k = metotlar.g1KisitiHesapla(ari1, PKomsuBilgileri,parselP);
          double h1k = metotlar.h1KisitiHesapla(ari1);
          double tkk = metotlar.tkKisitiHesapla(ari1, TA, TTabA,epsilon);
          ari1.violationHesapla(h1k, g1k,tkk);
          if (ari1.violation == 0)
          { ari1.fitnessHesapla(TA, TTabA,merkezDKot,kot1,kot2,cS); }
          arilar1[k] = (ari)ari1.Clone();          }
        best1 = (ari)metotlar.enIyiAriBul(arilar1).Clone();
        for (int c = 0; c < iterasyon; c++)
        {           //employed bees
          for (int p = 0; p < kaynakS; p++)
          {           ari komsu = (ari)arilar1[p].Clone();
            komsu.ariGonder(arilar1,paralel,parselP, MR, kaynakS, xKKisitlariMx, xKKisitlariMn,
yKKisitlariMx, yKKisitlariMn, bKatKisitlariMn, bKatKisitlariMx, bEnKisitlariMx, bEnKisitlariMn,
bBoyKisitlariMx, bBoyKisitlariMn, p, rn);
            //Deb in kuralına göre seçim yap
            doubleg1k=metotlar.g1KisitiHesapla(komsu, PKomsuBilgileri, parselP);
            double h1k = metotlar.h1KisitiHesapla(komsu);
            double tkk = metotlar.tkKisitiHesapla(komsu, TA, TTabA,epsilon);
            komsu.violationHesapla(h1k, g1k, tkk);
            if (arilar1[p].violation > 0 && komsu.violation > 0)
            {           if (komsu.violation < arilar1[p].violation)
              {           arilar1[p] = (ari)komsu.Clone();
                arilar1[p].gelisim = 0;          }          }
            else if (arilar1[p].violation <= 0 && komsu.violation <= 0)
            {     komsu.fitnessHesapla(TA, TTabA,merkezDKot,kot1,kot2,cS);
              if (komsu.fitness < arilar1[p].fitness)
              {           arilar1[p] = (ari)komsu.Clone();
                arilar1[p].gelisim = 0;          }          }
            else if (arilar1[p].violation > 0 && komsu.violation <= 0)
            {     komsu.fitnessHesapla(TA, TTabA, merkezDKot, kot1, kot2, cS);
              arilar1[p] = (ari)komsu.Clone();
              arilar1[p].gelisim = 0;
            } else    {           arilar1[p].gelisim++;          }          }
          //onlooker lar için ihtimal hesabı yap
          double tViolaiton = 0;
          double tFitness = 0;
          for (int z = 0; z < arilar1.Length; z++)
          {           if (arilar1[z].violation > 0)
            {           tViolaiton += arilar1[z].violation;
            }    else   {          tFitness += arilar1[z].fitness;          }          }
          for (int z = 0; z < arilar1.Length; z++)
```

```
{if (arilar1[z].violation > 0)
   {arilar1[z].normfit = (1 - arilar1[z].violation / tViolaiton) * 0.5;          }
   else
   {  arilar1[z].normfit = 0.5 + (arilar1[z].fitness / tFitness) * 0.5;}          }
//onlooker bees
int t = 0;
int m = 0;
while (t < arilar1.Length)
{              if (rn.NextDouble() < arilar1[m].normfit)
   {                  t++;
      ari onlooker = (ari)arilar1[m].Clone();
      onlooker.ariGonder(arilar1,paralel,parselP, MR, kaynakS, xKKisitlariMx, xKKisitlariMn,
yKKisitlariMx, yKKisitlariMn, bKatKisitlariMn, bKatKisitlariMx, bEnKisitlariMx, bEnKisitlariMn,
bBoyKisitlariMx, bBoyKisitlariMn, m, rn);
      //Deb in kuralına göre seçim yap
      double g1k = metotlar.g1KisitiHesapla(onlooker, PKomsuBilgileri, parselP);
      double h1k = metotlar.h1KisitiHesapla(onlooker);
      double tkk = metotlar.tkKisitiHesapla(onlooker, TA, TTabA,epsilon);
      onlooker.violationHesapla(h1k, g1k, tkk);
      if (arilar1[m].violation > 0 && onlooker.violation > 0)
      { if (onlooker.violation < arilar1[m].violation)
         {arilar1[m] = (ari)onlooker.Clone();
            arilar1[m].gelisim = 0;                    }}
      else if (arilar1[m].violation <= 0 && onlooker.violation <= 0)
      {onlooker.fitnessHesapla(TA, TTabA, merkezDKot, kot1, kot2, cS);
         if (onlooker.fitness < arilar1[m].fitness)
         {arilar1[m] = (ari)onlooker.Clone();
            arilar1[m].gelisim = 0;                    }              }
      else if (arilar1[m].violation > 0 && onlooker.violation <= 0)
      {onlooker.fitnessHesapla(TA, TTabA, merkezDKot, kot1, kot2, cS);
         arilar1[m] = (ari)onlooker.Clone();
         arilar1[m].gelisim = 0;                }
      else
      {                  arilar1[m].gelisim++;              }              }
   m++;
   if (m == (kaynakS))
   { m = 0; }              }
//run eniyisini güncelle
ari eniyi = (ari)metotlar.enIyiAriBul(arilar1).Clone();
if ((eniyi.fitness < best1.fitness) || (best1.fitness == 1000000000 && eniyi.fitness ==
1000000000 && eniyi.violation < best1.violation))
{  best1 = (ari)eniyi.Clone();              }
//scout bee aşaması
if (c % SSP == 0)              {
   int k = metotlar.maxFailure(arilar1);
   if (arilar1[k].gelisim > limit)
   {                  ari scout = (ari)metotlar.baslagicAtamasiYap(parselP,rn, xKKisitlariMn,
xKKisitlariMx, yKKisitlariMn, yKKisitlariMx, bEnKisitlariMn, bEnKisitlariMx, bBoyKisitlariMn,
bBoyKisitlariMx, bAciKisitlari,bKatKisitlariMn, bKatKisitlariMx).Clone();
      Double g1ks= metotlar.g1KisitiHesapla (scout, PKomsuBilgileri, parselP);
      double h1ks = metotlar.h1KisitiHesapla(scout);
      double tkks = metotlar.tkKisitiHesapla(scout, TA, TTabA,epsilon);
      scout.violationHesapla(h1ks, g1ks, tkks);
      if (scout.violation == 0)
      { scout.fitnessHesapla(TA, TTabA, merkezDKot, kot1, kot2, cS); }
      arilar1[k] = (ari)scout.Clone();
      arilar1[k].gelisim = 0;}              }
//if (checkBox1.Checked)
//{              //   best1.ciz(parselEn,parselBoy);
//     best1.bm.Save(Environment.CurrentDirectory + @"\çıkış/ABC " + r.ToString() + ", " +
c.ToString() + ", " + best1.fitness.ToString() + ".jpg");
//}
```

```
        best1.fitnessHesapla(TA, TTabA, merkezDKot, kot1, kot2, cS);
        best1.ciz(parselP,PKomsuBilgileri,merkezDKot,merkezpd);
        pictureBox1.Image = best1.bm;
        //sonuçların gösterilmesi
        double tKisiti = best1.tTA - TTabA;
        double kKisiti = Math.Abs(best1.tA - TA) - epsilon;
        if (best1.violation - kKisiti > 0.001 && best1.violation != 0)
        { label5.ForeColor = Color.Red; }
        else
        { label5.ForeColor = Color.White; }
        label5.Text="Violation= " + String.Format("{0:0.#}", best1.violation);
        label4.Text = ("Fitness= " + String.Format("{0:0.#}", best1.fitness));
        label10.Text = ("TAKS Kısıtı= " + String.Format("{0:0.#}", (tKisiti)));
        label15.Text = ("KAKS Kısıtı= " + String.Format("{0:0.#}", (kKisiti)));
        label16.Text = ("Kullanılan Toplam Taban Alanı= " + String.Format("{0:0.#}", best1.tTA) + "(" +
String.Format("{0:0.#}", (best1.tTA / (parselAlan))) + ")");
        label23.Text = ("Kullanılan Toplam İnşaat Alanı= " + String.Format("{0:0.#}", best1.tA) + "(" +
String.Format("{0:0.#}", (best1.tA / (parselAlan))) + ")");
```

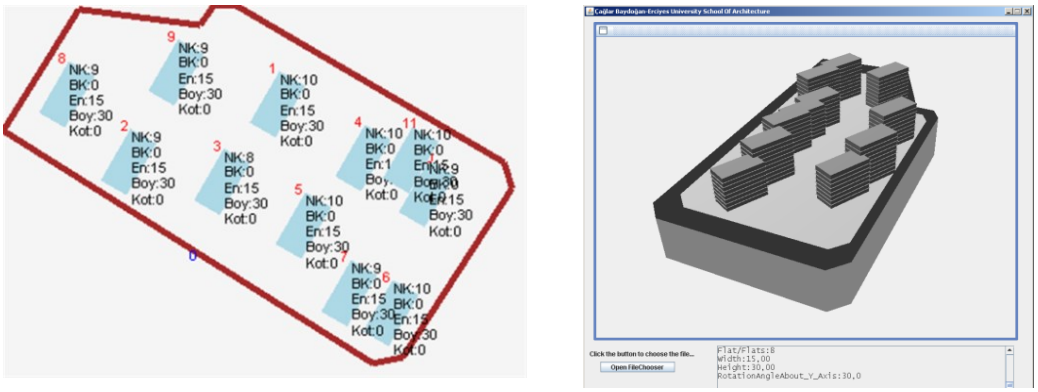
Kütle cs.

```
namespace konutyerlestirme
{   public class kutle:ICloneable
   {      public int id = 0;
      public PointF konum = new PointF();
      public double en = 0;
      public double boy = 0;
      public int katSayisi = 0;
      public int gKatSayisi = 0;
      public int aci = 0;
      public double kot = 0;
      public PointF merkezpiksel = new PointF();
      public PointF[] pt=new PointF[4];
      public void ptGuncele()       {
         PointF[] pt0 = new PointF[4];
         PointF p1 = new PointF(this.konum.X, this.konum.Y);
         pt0[0] = new PointF(p1.X, p1.Y);
         PointF p2 = new PointF(Convert.ToInt32(this.konum.X + this.en), this.konum.Y);
         pt0[1] = new PointF(p2.X, p2.Y);
         PointF p3 = new PointF(Convert.ToInt32(this.konum.X + this.en), Convert.ToInt32(this.konum.Y +
this.boy));
         pt0[2] = new PointF(p3.X, p3.Y);
         PointFp4=newPointF(this.konum.X,Convert.ToInt32(this.konum.Y+ this.boy ) );              pt0[3] =
new PointF(p4.X, p4.Y);
         this.pt = pt0;       }
      public object Clone()
      {          kutle s = new kutle();
         s.id = this.id;
         s.en = this.en;
         s.boy = this.boy;
         s.aci = this.aci;
         s.konum.X = this.konum.X;
         s.konum.Y = this.konum.Y;
         s.katSayisi = this.katSayisi;
         s.gKatSayisi = this.gKatSayisi;
         s.kot = this.kot;
         s.merkezpiksel.X = this.merkezpiksel.X;
         s.merkezpiksel.Y = this.merkezpiksel.Y;
         for (int i = 0; i < pt.Length; i++)
         {             if (this.pt[i] != null)             {
               PointF pfs = new PointF(this.pt[i].X,this.pt[i].Y);
               s.pt[i] = pfs;
            }
         }          return s; }   }}
```

EK C. : Program çıktıları

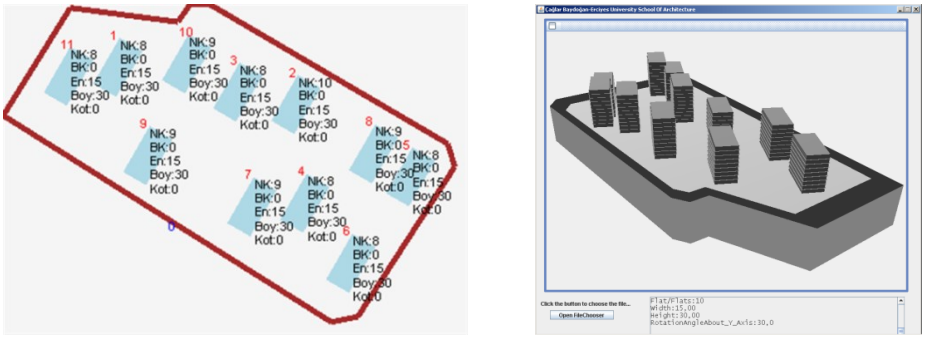


Şekil C.1 : Program çıktıları 1.

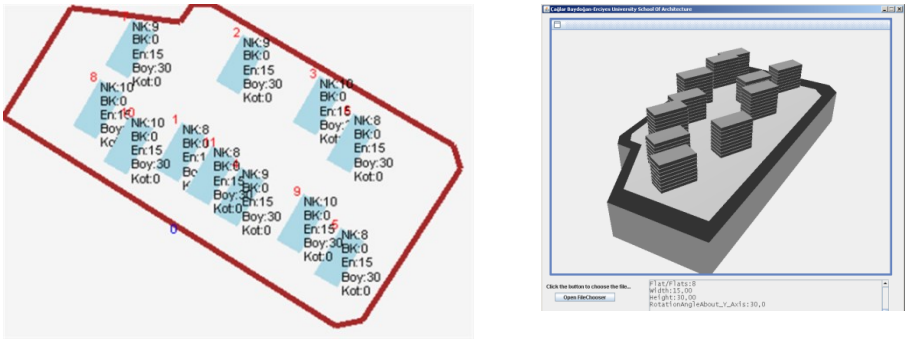


(a) **(b)**

Şekil C.2 : Program çıktıları 2.



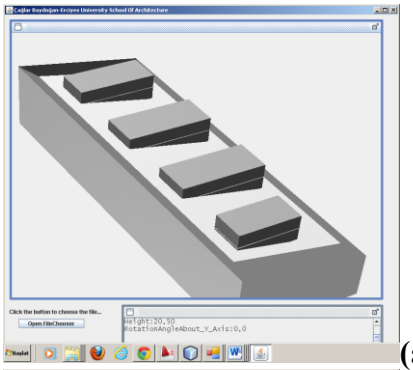
Şekil C.3 : Program çıktıları 3.

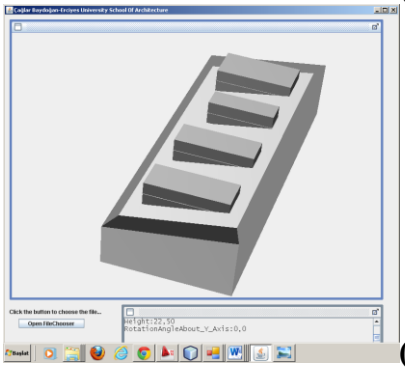


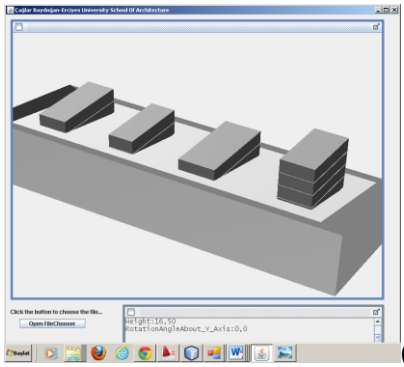
Şekil C.4 : Program çıktıları 4.



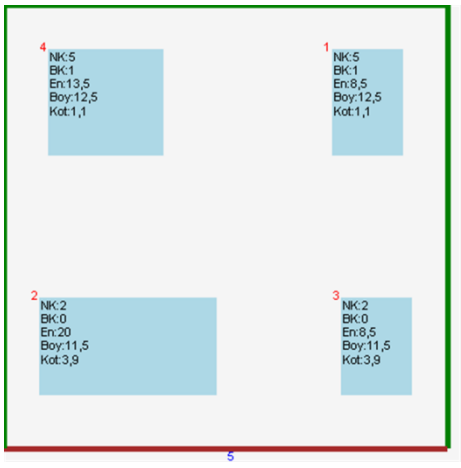
Şekil C.5 : Program çıktıları 5.


(a)

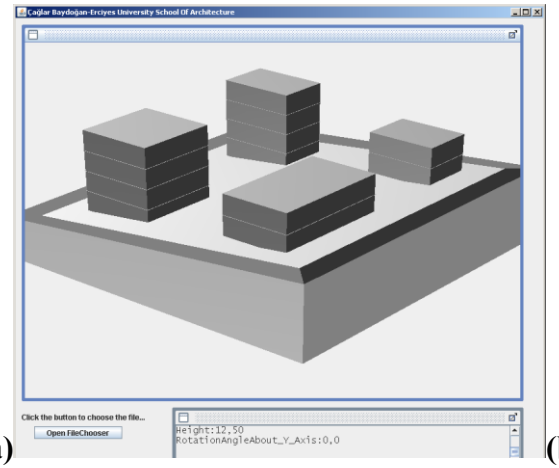

(b)


(c)

Şekil C.6 : Program çıktıları 6.

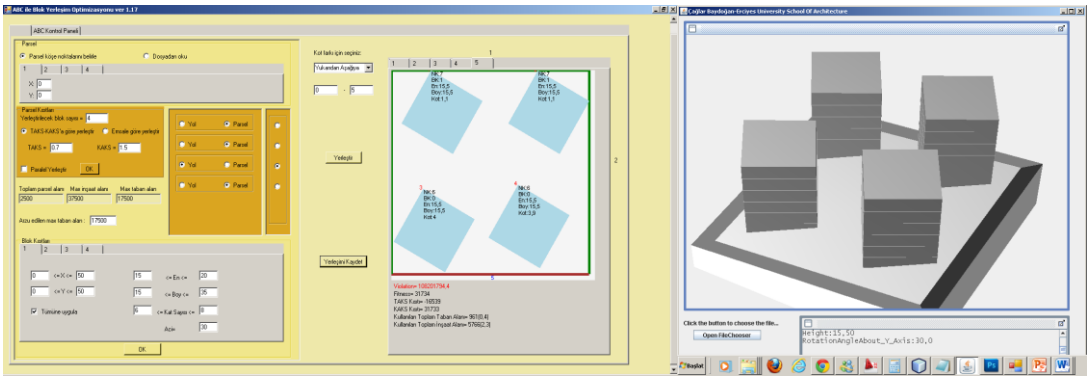


(a)

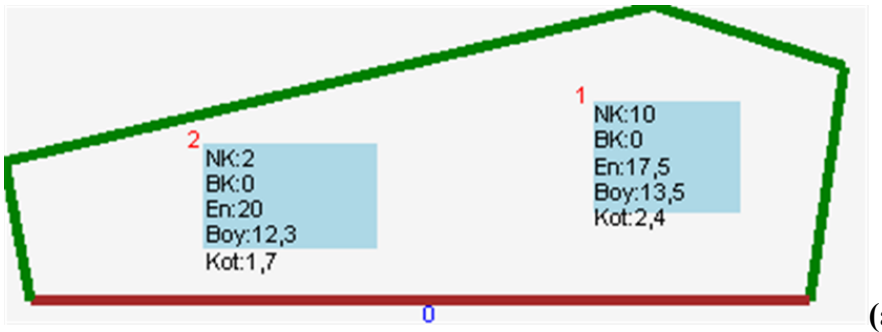


(b)

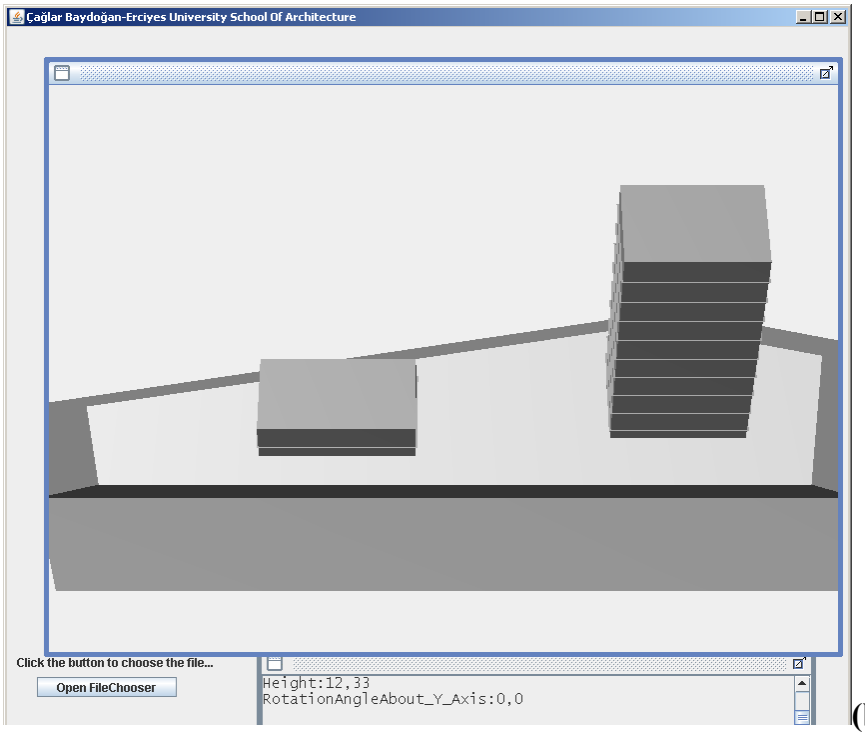
Şekil C.7 : Program çıktıları 7.



Şekil C.8 : Program çıktıları 8.



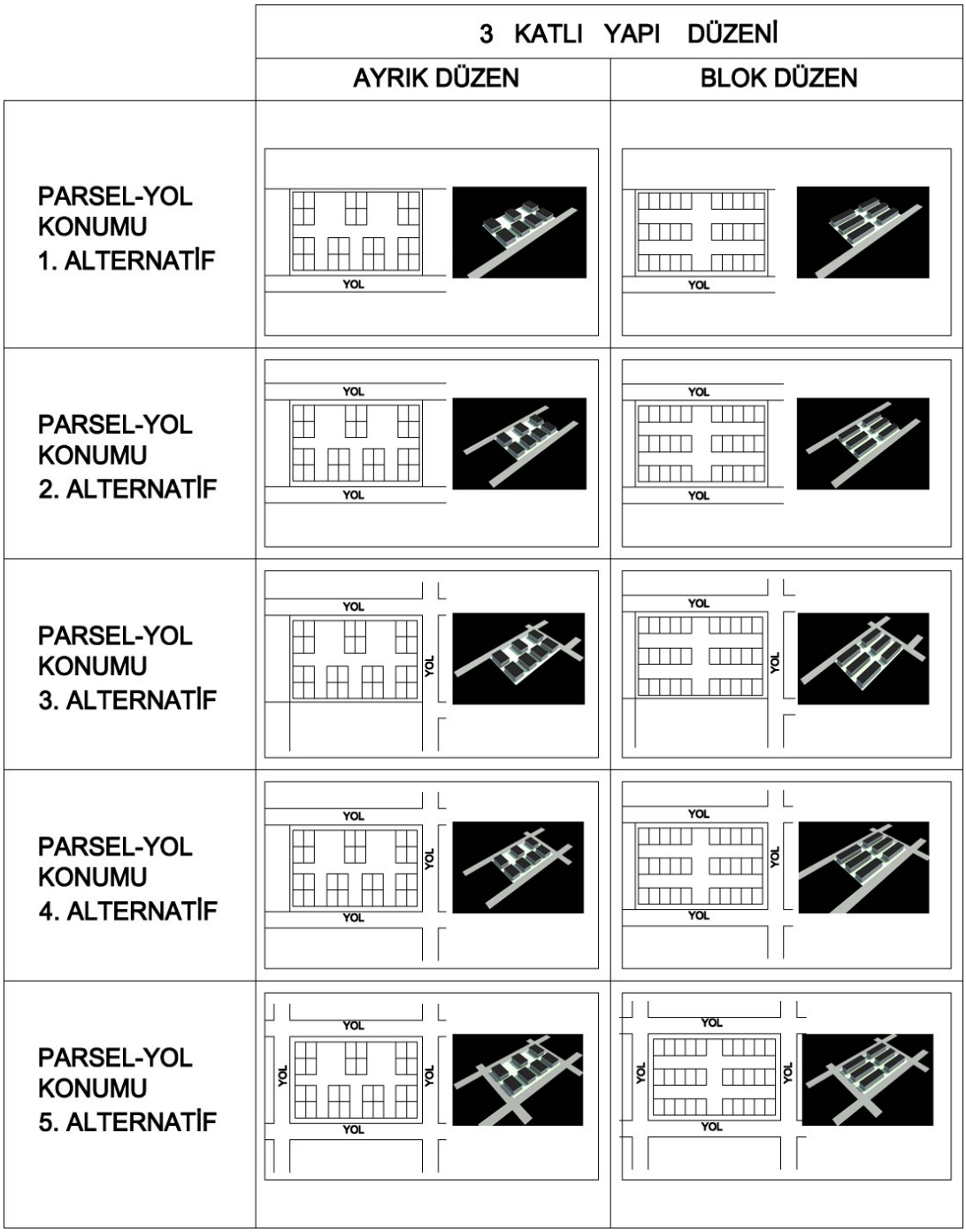
(a)



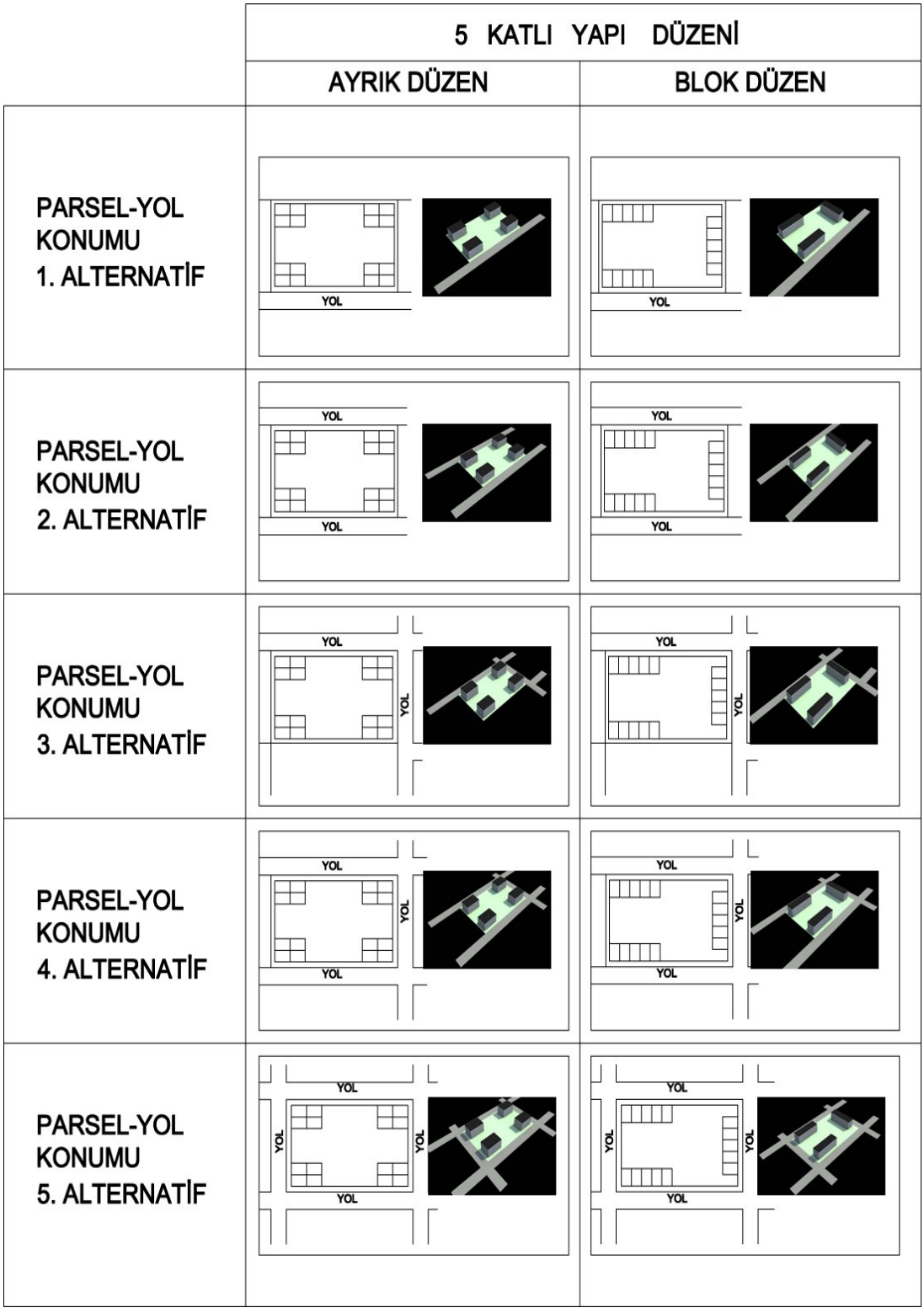
(b)

Şekil C.9 : Program çıktıları 9.

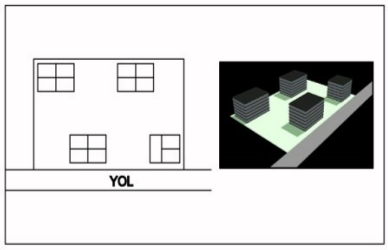
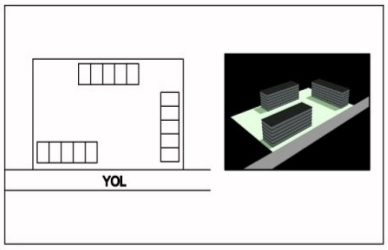
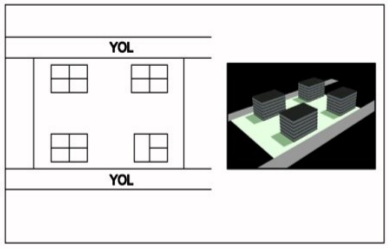
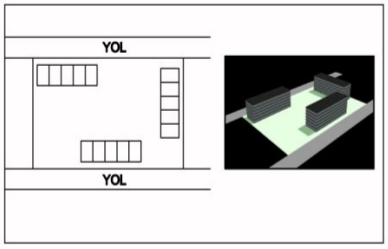
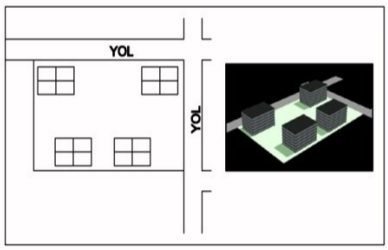
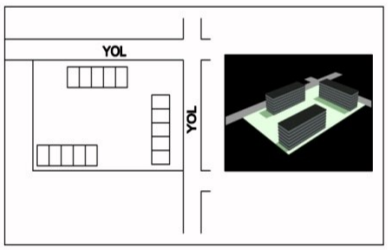
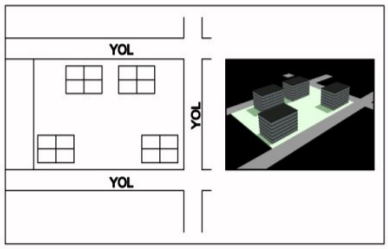
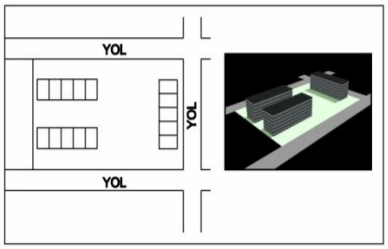
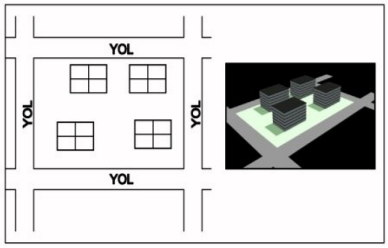
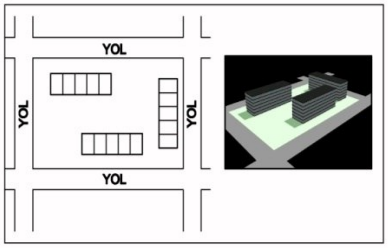
EK E : Farklı yapı nizamları, yapı-yol durumu şemaları.



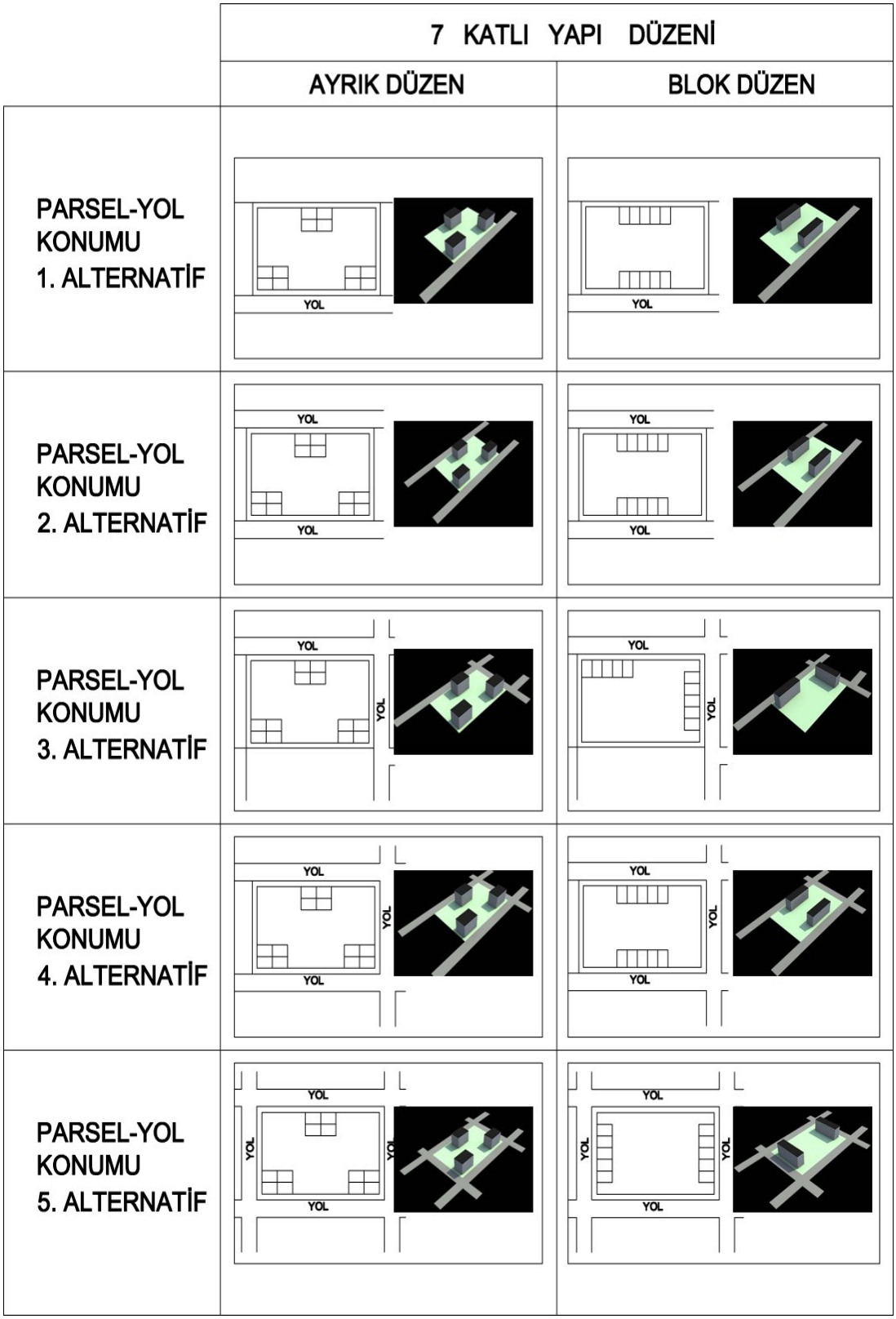
Şekil E.1 : 3 katlı kütleler için farklı yerleşim şemaları.



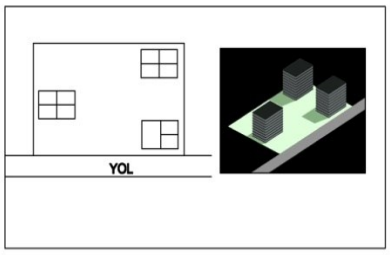
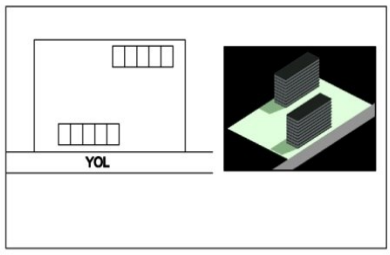
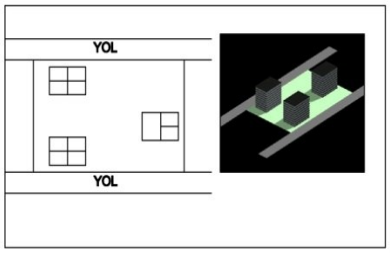
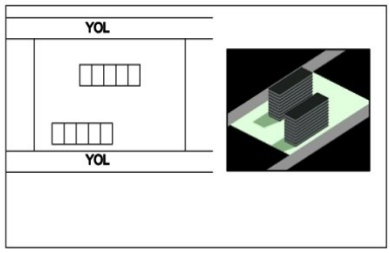
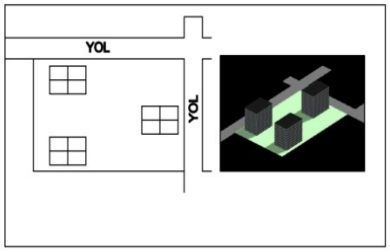
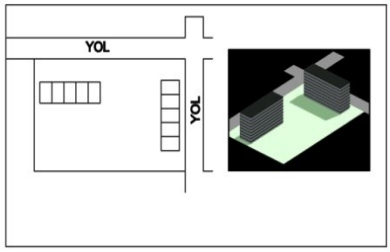
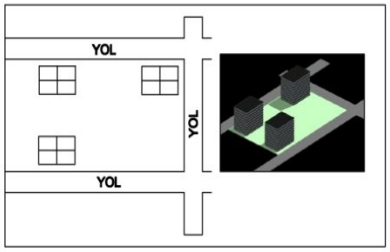
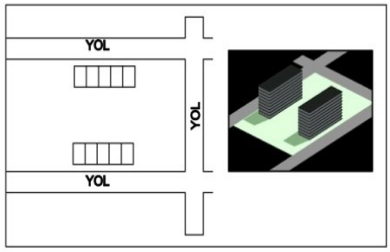
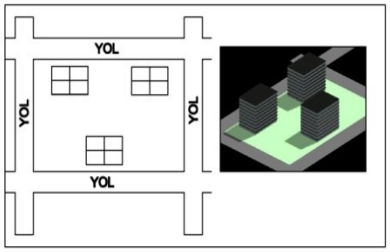
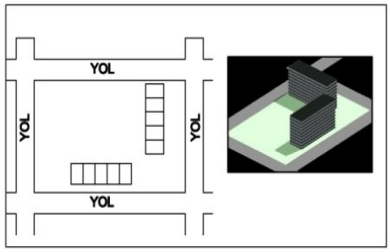
Şekil E.2 : 5 katlı kütleler için farklı yerleşim şemaları.

	6 KATLI YAPI DÜZENİ	
	AYRIK DÜZEN	BLOK DÜZEN
PARSEL-YOL KONUMU 1. ALTERNATİF		
PARSEL-YOL KONUMU 2. ALTERNATİF		
PARSEL-YOL KONUMU 3. ALTERNATİF		
PARSEL-YOL KONUMU 4. ALTERNATİF		
PARSEL-YOL KONUMU 5. ALTERNATİF		

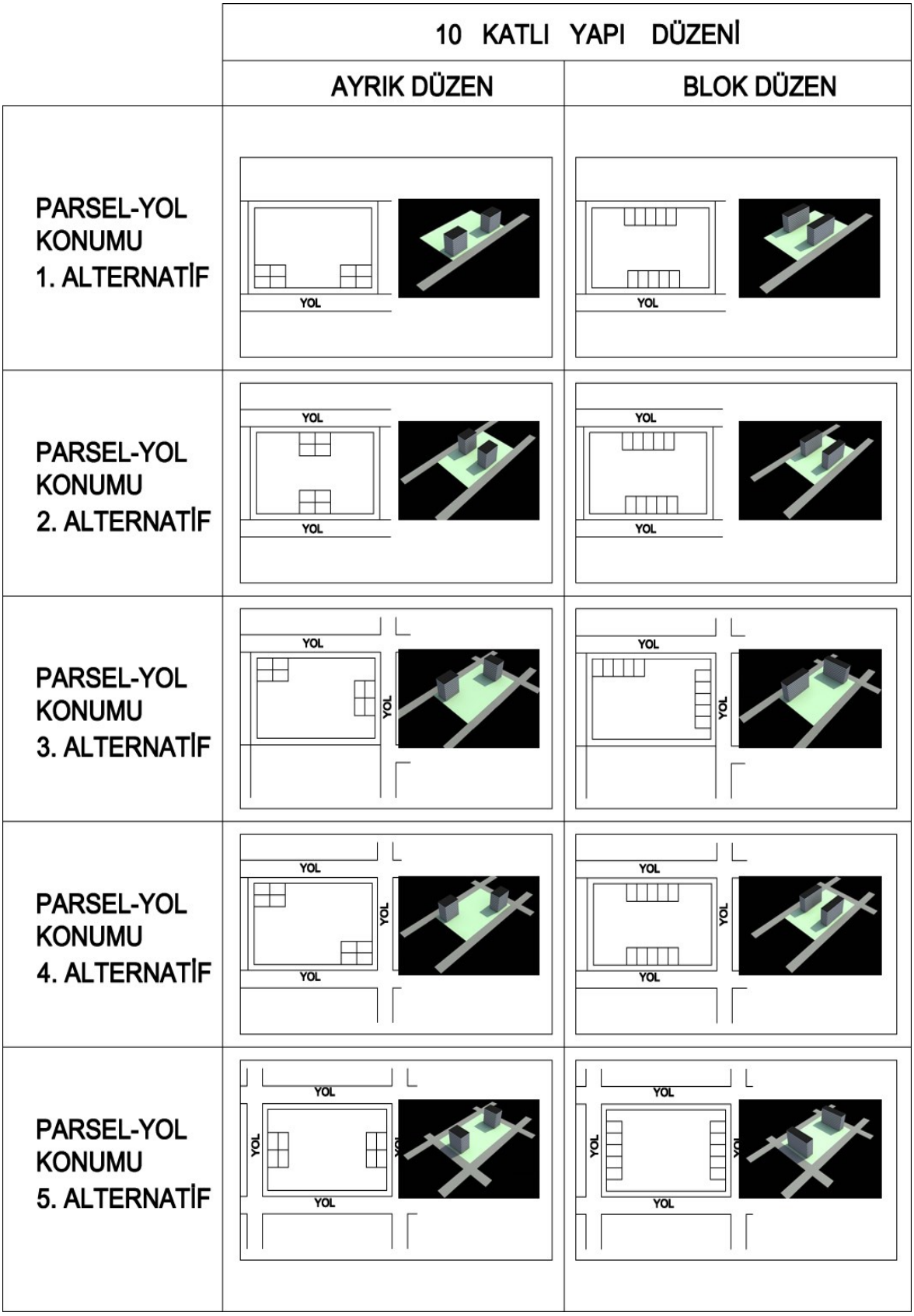
Şekil E.3 : 6 katlı kütleler için farklı yerleşim şemaları.



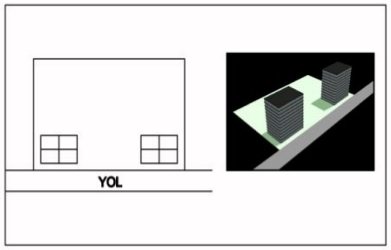
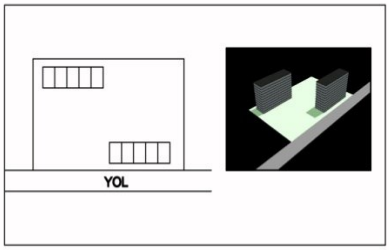
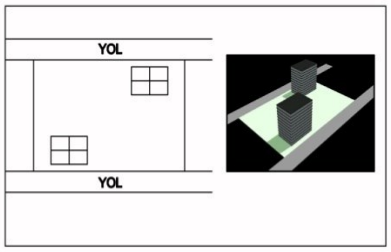
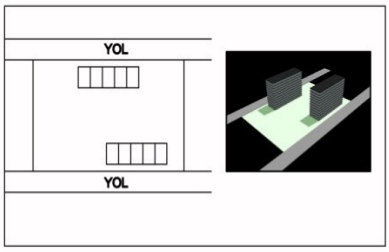
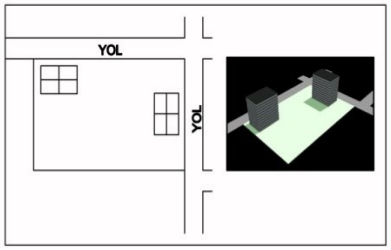
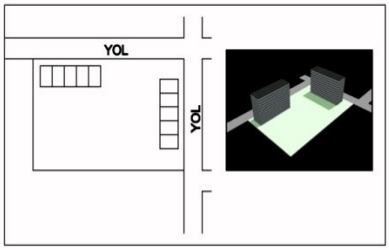
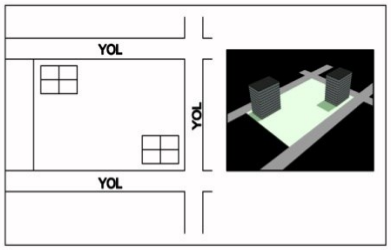
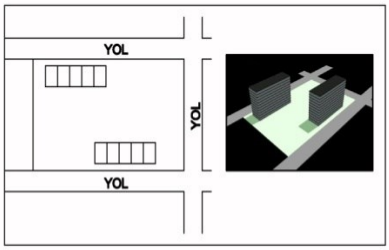
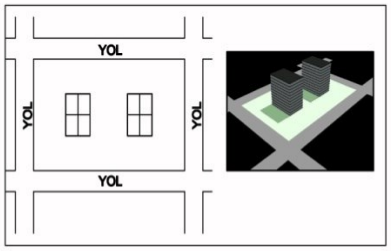
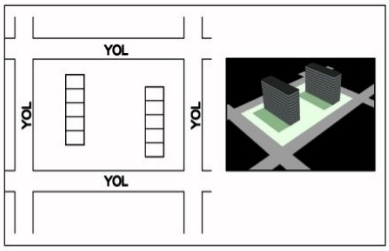
Şekil E.4 : 7 katlı kütleler için farklı yerleşim şemaları.

	9 KATLI YAPI DÜZENİ	
	AYRIK DÜZEN	BLOK DÜZEN
PARSEL-YOL KONUMU 1. ALTERNATİF		
PARSEL-YOL KONUMU 2. ALTERNATİF		
PARSEL-YOL KONUMU 3. ALTERNATİF		
PARSEL-YOL KONUMU 4. ALTERNATİF		
PARSEL-YOL KONUMU 5. ALTERNATİF		

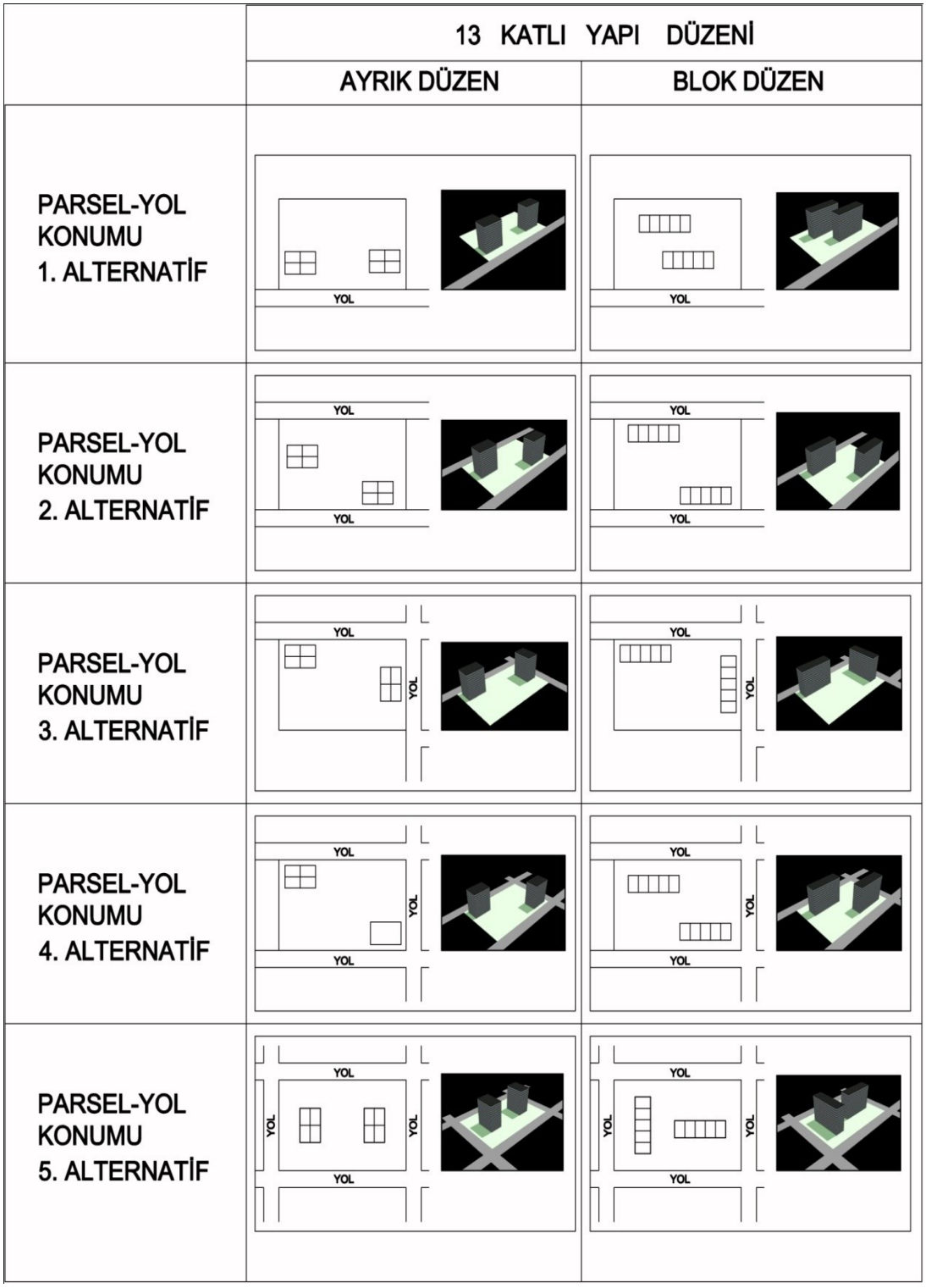
Şekil E.5 : 9 katlı kütleler için farklı yerleşim şemaları.



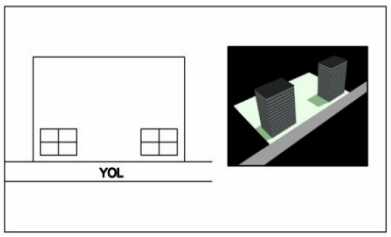
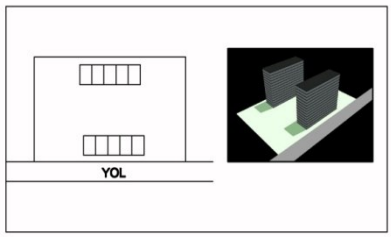
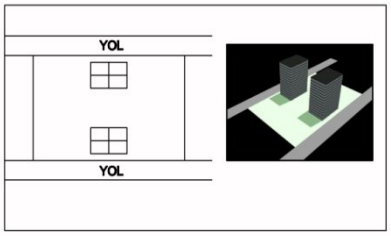
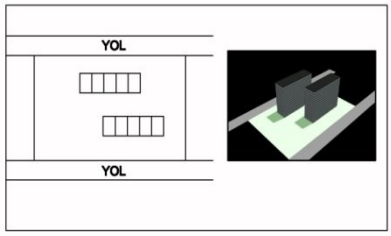
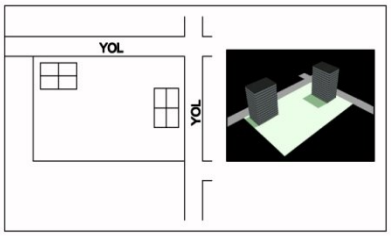
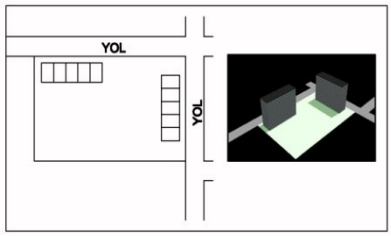
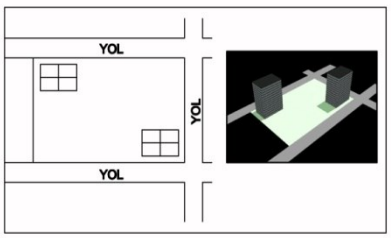
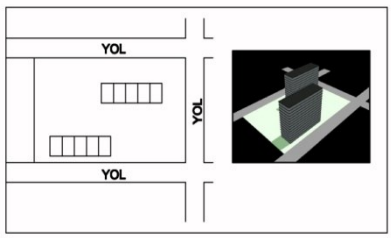
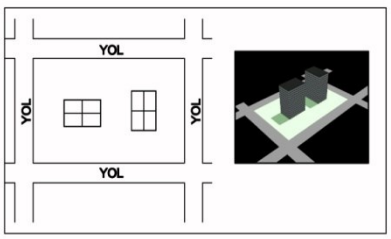
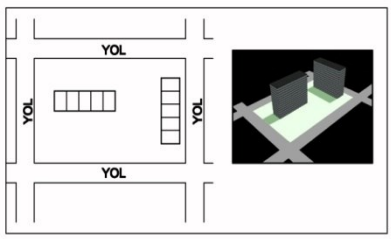
Şekil E.6 : 10 katlı kütleler için farklı yerleşim şemaları.

	11 KATLI YAPI DÜZENİ	
	AYRIK DÜZEN	BLOK DÜZEN
PARSEL-YOL KONUMU 1. ALTERNATİF		
PARSEL-YOL KONUMU 2. ALTERNATİF		
PARSEL-YOL KONUMU 3. ALTERNATİF		
PARSEL-YOL KONUMU 4. ALTERNATİF		
PARSEL-YOL KONUMU 5. ALTERNATİF		

Şekil E.7 : 11 katlı kütleler için farklı yerleşim şemaları.



Şekil E.8 : 13 katlı kütleler için farklı yerleşim şemaları.

	14 KATLI YAPI DÜZENİ	
	AYRIK DÜZEN	BLOK DÜZEN
PARSEL-YOL KONUMU 1. ALTERNATİF		
PARSEL-YOL KONUMU 2. ALTERNATİF		
PARSEL-YOL KONUMU 3. ALTERNATİF		
PARSEL-YOL KONUMU 4. ALTERNATİF		
PARSEL-YOL KONUMU 5. ALTERNATİF		

Şekil E.9 : 14 katlı kütleler için farklı yerleşim şemaları.

ÖZGEÇMİŞ



Ad Soyad **:** Murat Çağlar Baydoğan

Doğum Yeri ve Tarihi **:** Malatya- 06.06.1976

Adres **:** Erciyes Üniversitesi Mimarlık Fakültesi 38039 Kayseri

E-Posta **:**caglar.baydogan@gmail.com

Lisans Üniversite **:** Erciyes Üniversitesi Mimarlık Fakültesi 1999

Yüksek Lisans Üniversite **:** Gazi Üniversitesi Fen Bilimleri Enstitüsü 2002

Mesleki Deneyim ve Ödüller: (2010), Erciyes Üniversitesi Yerleşkesi Alt Merkez Mimari Proje Yarışması **1.Ödül**, Erciyes Üniversitesi Rektörlüğü, Kayseri.

Yayın ve Patent Listesi **:**

Yazılan ulusal kitaplar veya kitaplarda bölümler

Baydoğan M., Ç. (2012). 20.Yüzyılda Kayseri Cumhuriyet Meydanının Değişim Süreci, Kayseri'nin Yirminci Yüzyılı: Mimarlık. *Kent Tarihi ve Kültürü*, ed. Asiliskender, B., Ceylan, B., Tozoğlu, A. E., ss.103-117, ISBN 978-605-4658-00-8, Abdullah Gül Üniversitesi, Kayseri.

Ulusal bilimsel toplantılarda sunulan ve bildiri kitabında basılan bildiriler

Çoruh, L., **Baydoğan M., Ç.** (2011). Mimarlık Eğitiminde Etkileşimli Ve Üç Boyutlu Dijital Prototip Uygulamalarının Etkinliğinin Değerlendirilmesi. *1. Sanat Ve Tasarım Eğitimi Sempozyumu - Dün Bugün Gelecek*, Bildiri Kitabı. Sayfa 522-525.