GAZİ ÜNİVERSİTESİ

1926

BİLGİSAYAR YAZILIMLARINDA FRAKTAL ÇİZİMLERİ

Sümeyye AKPINAR

YÜKSEK LİSANS TEZİ
MATEMATİK ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ŞUBAT 2022

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Sümeyye AKPINAR

11/02/2022

BİLGİSAYAR YAZILIMLARINDA FRAKTAL ÇİZİMLERİ
(Yüksek Lisans Tezi)

Sümeyye AKPINAR

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
Şubat 2022

ÖZET

Bu çalışmada bilgisayar yazılımlarından olan LaTeX, Mathematica ve diğer hazır paket programlarında fraktal çizim kodları araştırılarak yazılımlarda çizim kodları verilmektedir. Ayrıca fraktalın boyutunu hesaplayan bir yazılım MATLAB bilgisayar programı da verilmektedir. Fraktalların insan vücudundan kelebek kanatlarına ve çeşitli sanatsal alanlarda nasıl kullanıldığı da verilmektedir. Fraktala gönül veren matematikçi ve matematikçi olmayan insanların çalışmaları da bu tezde mevcuttur. Bu çalışmanın orijinal tarafı da LaTeX ile yazılan kodlarla Öklid geometrisinde bilinen üçgen, dörtgen, beşgen ve ongene kadar çokgenlerden fraktallar elde etmektir. Son olarak üçgenden ongene kadar olan çokgenlerden elde edilen fraktallar için bir LaTeX kodu yazıldı. Bu yazılımda yukarıda bahsedilen fraktalları bir arada görmek mümkündür.

Bilim Kodu	:	20402
Anahtar Kelimeler	:	Fraktal, Fraktal geometrik şekiller, Newton metodu, Latex, Mathematica
Sayfa Adedi	:	107
Danışman	:	Prof. Dr. Aysel TURGUT VANLI

FRACTAL DRAWINGS IN COMPUTER SOFTWARES
(M. Sc. Thesis)

Sümeyye AKPINAR

GAZİ UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
February 2022

ABSTRACT

In this study, fractal drawing codes are searched in computer softwares LaTeX, Mathematica and other ready-made package programs and drawing codes are given in the softwares. In addition, a software that calculates the dimension of fractal is given in the MATLAB computer program. How fractals are used from the human body to butterfly wings and in various artistic fields are also given. There are also studies of mathematicians and non-mathematicians who love fractals. The original side of this study is to obtain fractals from triangles, quadrilaterals, pentagons and polygons up to decagons known in Euclidean geometry with codes written in LaTeX. Finally, a LaTeX code are written for fractals obtained from polygons from triangle to decagon. In this software, it is possible to see mentioned all of the fractals mentioned above together.

Science Code	:	20402
Keywords	:	Fractal, fractal geometric shapes, Newton mehtod, Latex, Mathematica
Number of pages	:	107
Supervisor	:	Prof. Dr. Aysel TURGUT VANLI

TEŞEKKÜR

Bu çalışmanın gerçekleşmesinde emeği geçen, deneyim ve tecrübelerinden yararlandığım, desteğini esirgemeyen saygıdeğer hocam Prof. Dr. Aysel TURGUT VANLI'ya, eğitim hayatım boyunca emeğini ve sevgisini esirgemeyen sevgili eşim Murat AKPINAR'a ve sevgili aileme teşekkürlerimi sunuyorum.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılan simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
a	Ara boyutların ölçü birimi
$\mathbf{d_k}$	Kutu sayma boyutu
D	Dolambaçlık derecesi
Na	Toplam uzunluk
$\mathbf{p_n}$	Newton metodunun yaklaşık değeri

Kısaltmalar	**Açıklamalar**
GUI	Guide User Interface

1. GİRİŞ

Yüzyıllar boyu insanlar, doğayı tanıyıp gizemini çözmek için uğraşmışlardır. Fraktal, doğadaki karmaşık görünen mükemmelliğin tanımıdır. Karmaşıklıkların içinde sonsuza süren döngü ve uyumlardır. 20. yüzyıla kadar doğayı tanımlarken Öklid geometrisi kullanmışlardır. Dağ; bir üçgene, güneş; bir daireye, binalar; bir dikdörtgene benzetmişlerdir. Zamanla gelişen teknoloji sayesinde aslında bir dağın kenarının üçgen kadar düz bir çizgiye sahip olmadığı, pürüzlü bir yapıya sahip olduğu ortaya çıkmıştır.

Fraktal kavramını ortaya ilk atan kişi Benoit Mandelbrot'dur. Bu kavramı İngiltere kıyılarının kıvrımlı yapısını ölçmeyi denerken bulmuştur. Doğadaki bir çok yapı fraktala örnektir. Fraktal kavramı ortaya çıktıktan sonra bu yapılar daha çok anlaşılır olmuştur [1]. Bu alanda ilk çalışan kişiler Gaston Julia ve Pierre Fatou'dur. Fakat o yıllarda teknoloji yeterince gelişmediği için yapılan çalışmalar tamamlanıp görselleştirilmemiştir. Gelişen teknoloji sayesinde Mandelbrot bu çalışmaları görselleştirmiştir. Bu sebeple kendi adıyla anılmaktadır. Fraktal tanımları yapıldıktan sonra fraktal yapıların boyutları hesaplanmak istenmiş ve farklı metodlar geliştirilmiştir. Bu metodlardan biri kutu sayma metodudur. Bu metod sayesinde pürüzlü yapıların boyutları tam olarak hesaplanabilmektedir. Aynı zamanda fraktal çizimlerinde ve hesaplamalarında Newton metodu da çok sık kullanılmaktadır. Fraktal zamanla yalnızca matematik alanında değil resim, seramik, tıp gibi dallarında kullanım alanına da girmiştir.

Fraktal gelişen teknolojilerle birlikte bilgisayar ortamında görselleştirilmeye başlanmıştır. Geçmiş yıllarda bilgisayarlar bu kadar gelişmeden önce 1883'te Alman matematikçi Georg Cantor kendi adıyla anılan Cantor kümesini, 1942'de ise Hollandalı matematik öğretmeni Albert E. Bosman Pisagor ağacını oluşturmuştur [2]. Daha sonraki yıllarda matematikçiler tarafından sayısız fraktalın varlığı öne sürülmüştür. Böylece ilk fraktal üreten yazılım, Benoit Mandelbrot'un Julia kümesini genelleştirmek için başlayan arayışından dolayı ortaya çıkmıştır. Mandelbrot, 1979'da karmaşık düzlemin bir görüntüsünü yineleme ile oluşturabilceğini keşfetmiş ve bu sayede ilk temel fraktal çıktılar üretilmiştir. Daha sonraki yıllarda Loren Carperter, fraktallar hakkında Vol Libre adında iki dakikalık kısa bir film üretmiştir [3]. York Üniversitesi'nde Bilgisayar Bilimi Profesörü olan Susan Stepney ise

Acorn User dergisinin 1983 yılının ekim ayı sayısında fraktal şekiller üretmek için bir BBC BASIC listesi yayınlamıştır [4]. 1984 yılına gelindiğinde ise Rescue on Fractalus adında bir bilgisayar oyunu üretilmiştir. 1980'lerin başından yaklaşık 1995'e kadar yüzlerce farklı fraktal tip formüle edilmiştir. 3D fraktal üretimi ise 2009 yılı civarında ortaya çıkmıştır. Fraktal üreten yazılımların bir listesi de John Briggs tarafından 1992'de yayınlanan Fractals: The Patterns of Chaos adlı kitap için derlenmiştir [5]. Bu şekilde devam eden çalışmalarla bu alandaki yazılımlar ve geliştirmeler günümüze kadar devam etmektedir.

2. FRAKTAL

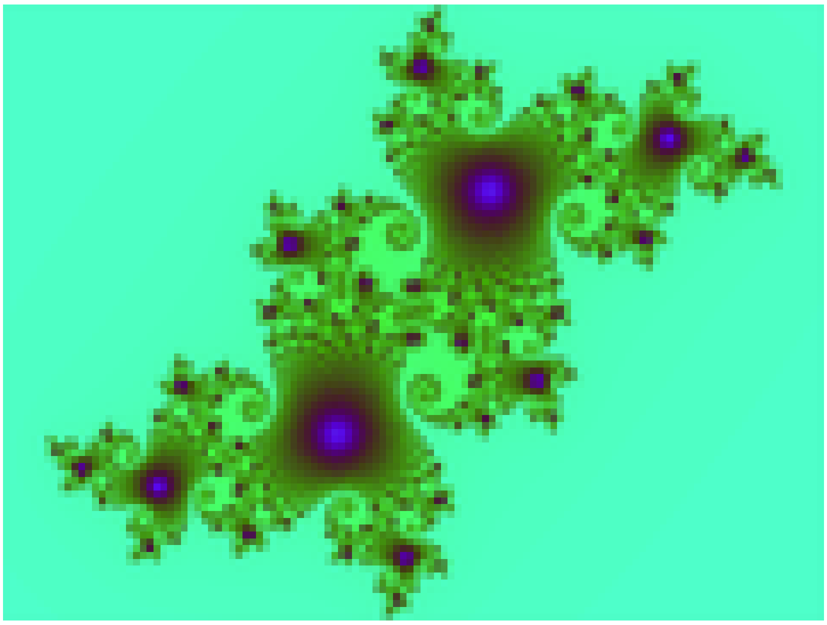
2.1. Fraktalın Doğuşu

Fraktal, Latince fractus kelimesinden türetilmiştir. "Fractus" kelime anlamı olarak kırılmış, parçalanmış anlamını taşımaktadır. 20. yüzyıla dek Öklid geometrisini kullanırken doğadaki şekiller için kare, çember, üçgen gibi geometrik şekillerin yeterli olmadığını farkedip daha net ölçümler yapabilmek adına Fraktal Geometri kullanılmaya başlanmıştır. Fraktal kelimesi ilk olarak 1975'te Polonya asıllı matematikçi Benoit Mandelbrot tarafından ortaya atılmıştır. Fraktal, bir geometri sistemidir ve fraktallar yakından incelendiğinde büyük şekli oluşturan ve orantılı olarak küçülerek oluşan küçük şekillerin büyük şekle benzediği bir sistemdir, bu kendini tekrar etme olayı sonsuza kadar uzar. Fraktalın en belirgin özelliği kendini tekrar etmesi ve kesilen her parçada büyük resim görülmesidir.

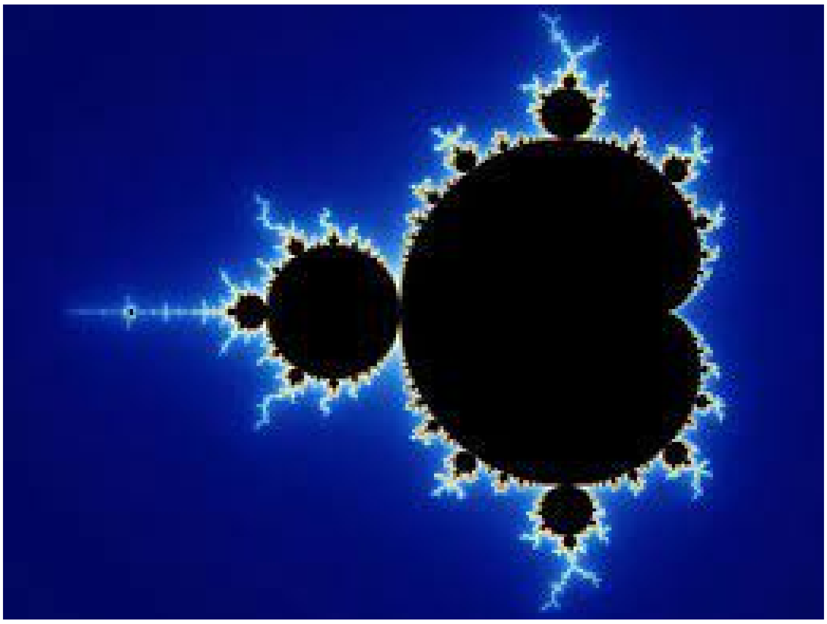
Fraktallarla ilgili ilk çalışmaları Fransız asıllı matematikçiler Gaston Julia (1893-1978) ve Pierre Fatou (1878-1929) tarafından yapılmış fakat onların yaşadığı zaman diliminde bilgisayarlar henüz bu fraktalları gösterebilecek kadar gelişmediğinden, Gaston Julia kendi oluşturduğu fraktal kümesinin (Julia kümesi) şeklini bilgisayarda görememiştir. Bunu yıllar sonra Mandelbrot göstermiştir.

Mandelbrot çocukluk yıllarında amcası sayesinde matematikle tanışmıştır. Yıllar sonra gençlik döneminde yaşadığı ülkeden savaş sebebiyle kaçıp Fransa'da yaşayan amcası Szolem Mandelbrot'un yanına taşınmıştır. Szolem, Benoit'in eğitimini üstlenmiştir. Bir matematikçi ve mühendis olan Benoit, amcasının önerisiyle 25 yıldır çözülemeyen bir problem ile tanışmıştır. Bu problem Julia ve Fatou'nun problemleridir. Bu sayede yeni bir kariyer imkanı karşısında belirmiştir. Mandelbrot da bu problemle ilgilenmeye başlamış ve gelişen bilgisayar teknolojileri sayesinde problemleri tamamlayıp görsel olarak sunmuştur. Mandelbrot'un geliştirdiği Mandelbrot kümesini, sanal karmaşık sayıların kullanılmasıyla elde edilen fonksiyonları bilgisayar ortamında muhteşem fraktallara dönüştürülebilen küme olarak tanımlamıştır [6].

Julia, formül olarak Julia Kümesi'ni $f(z) = z^2$ (Şekil 2.1) şeklinde tanımlarken Mandelbrot ise $f(z) = z^2 + c$ (Şekil 2.2) şeklinde tanımlamıştır.



Şekil 2.1. Julia seti



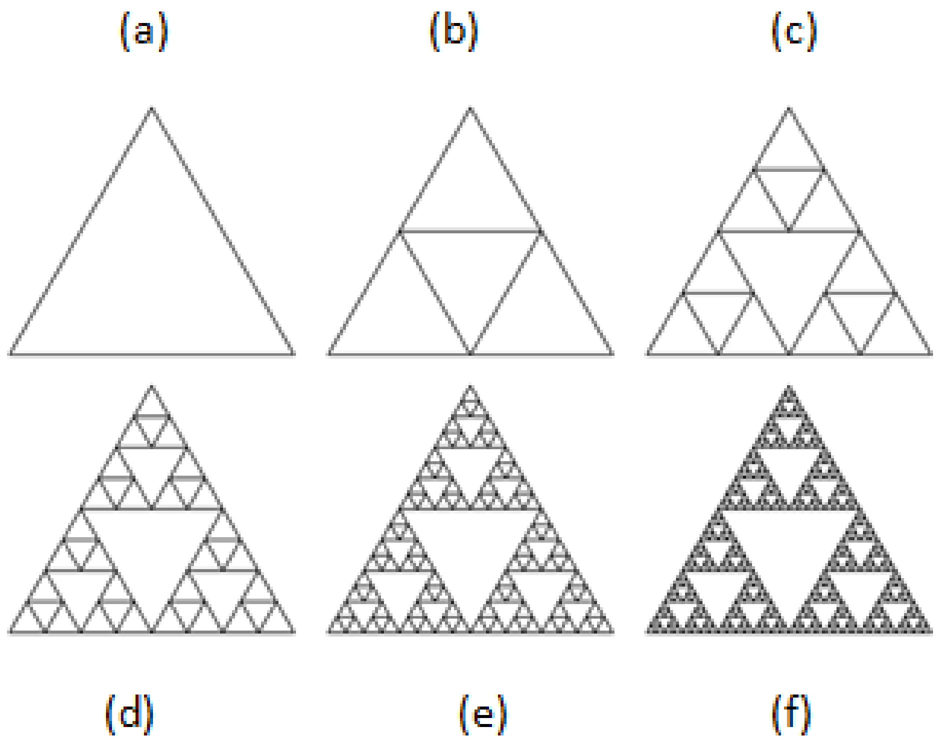
Şekil 2.2. Mandelbrot kümesi

2.2. Fraktalın Klasik ve Doğadaki Örnekleri

1. *Sierpinski Üçgeni:*

   Polonyalı matematikçi Waclaw Sierpinski (1882-1969) 1916 yılında, daha sonra kendi adıyla anılan ve Sierpinski Üçgeni (Sierpinski Triangle) veya Sierpinski Şapkası (Sierpinski Gasket) veya Sierpinski Kalburu (Sierpinski Sieve) da denen bir fraktal tanıtmıştır. Bu şeklin 12. yüzyılda bir kilisede süsleme olarak kullanıldığı da bilinmektedir.

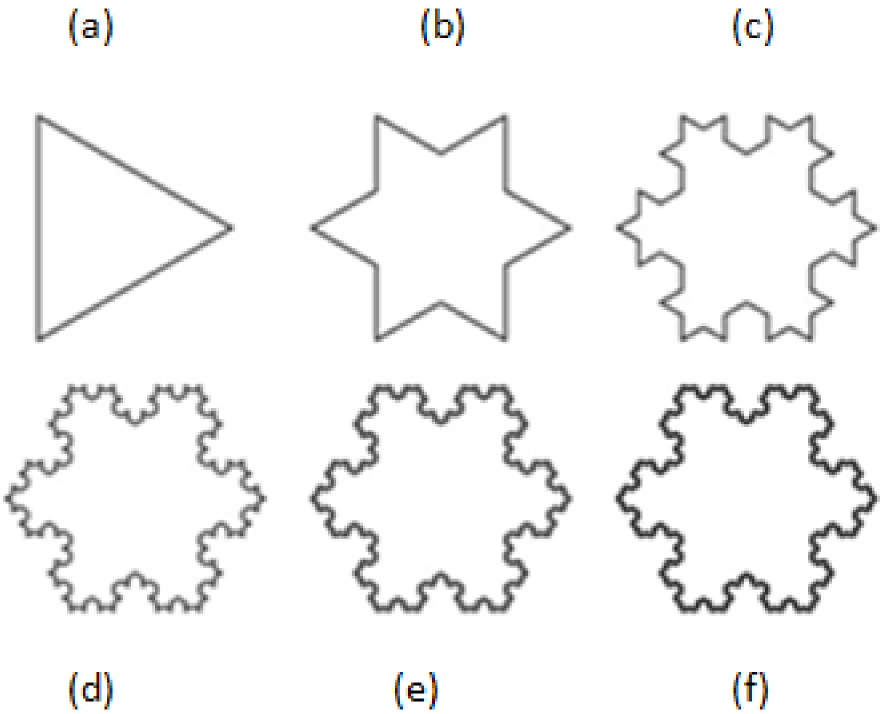
Sierpinski Üçgeni için kenar uzunluğu 1 olan bir eşkenar üçgen çizilir (Şekil 2.3(a)). Üçgenin orta noktaları birleştirilip ortada bir üçgen oluşturulur ve o üçgen kesip çıkarılır (Şekil 2.3(b)). Kalan büyük üçgenin kenarları 4 eş parçaya bölünür ve orta noktaları birleştirilir (Şekil 2.3(c)) ve bu küçük üçgenlerin kenar uzunlukları 1/2'dir. Geriye kalan şekil için aynı işlemler tekrar edilir ve daima 1/2 oranında küçültülmeyle oluşan çizgi modeli kendine benzeyen üçgenlerden oluştuğu için bir fraktal modeldir (Şekil 2.3).



Şekil 2.3. Sierpinski üçgeni

2. *Koch Kartanesi:*

   İsveçli matematikçi Niels Fabian Helge Von Koch tarafından bulunmuştur. İlk olarak bir eşkenar üçgen çizilir (Şekil 2.4(a)). Üçgenin bir kenarı üç eşit parçaya ayrılır ve ortadaki parça alınır. Boşta kalan iki uca alınan bu parçadan birer tane bağlanır ve uçlar üçgenin dışında birleştirilir. Kalan diğer iki kenar içinde aynı işlemler yapılır. Böylece altı köşeli bir yıldız elde edilir (Şekil 2.4(b)). Daima olarak her kenar için aynı işlemlerin tekrar edilmesiyle Koch kartanesi oluşturulur. Eşkenar bir üçgenin sürekli olarak uç kısımlarının, simetrik şekilde katlanmasıyla elde edilen bu şeklin kar tanesini andırmasından dolayı bu adı almıştır (Şekil 2.4).



Şekil 2.4. Koch kartanesi

3. *Doğadan örnekler:*

   Doğadaki hiçbir şey tek başına varolmamıştır ve hepsi arasında bir denge, düzen vardır. Öklid geometrisindeki gibi hiçbir şey kusursuz düzgünlükte değildir. Doğada kendiliğinden oluşan şekiller arasında da benzerlik görülmektedir. Bu karmaşık görünen benzerlik fraktalı ortaya koyar. Fraktalı doğanın her yerinde görmek mümkündür. Yenilen sebzelerde, çiçeklerde, ağaçlarda görülebilmektedir. Karmaşık görülen bu şekillerin aslında olan mükemmeliyeti herkes de hayranlık uyandırır. Fraktal tanımını farkında olmadan bakılan bu sanat eserleri doğanın sanatıdır [7, 8].

Şekil 2.5. Doğadan örnekler

Doğadan bir kaç fraktal örneği Şekil 2.5'deki gibidir.

2.3. Fraktal Boyut

Öklid geometrisinde boyutlar çoğunlukla tamsayıdır. Örneğin; bir doğru 1 boyutlu, düzlem 2 boyutlu, uzay ise 3 boyutludur. Fakat Felis Hausdorff (1886-1942), A.S. Besicovitch (1891-1970) ve A.N. Kolmogorov (1903-1987) gibi matematikçiler tamsayı olmayan boyutların varlığından bahsedip boyutu tamsayılardan farklı olarak tanımlamışlardır. Önermelerine göre; doğrular 1 boyuta, kareler 2 boyutla, birçok eğrilerin özelliklerine göre değişen ara boyuta sahiptir. Ara boyutlar a ile gösterilsin.

Ölçü birimini a olarak seçip daha sonra dolambaçlı çizginin uzunluğuda a cinsinden ölçülür. Buradaki dolambaçlı çizgi ölçülen şeklin kenarlarındaki pürüzlü yapının ölçülmesiyle ortaya çıkar. Kabul edilsin ki a ölçütü N defa kullanılmış olsun, yani toplam ölçü Na olsun. O halde Mandelbrot'un tanımına göre fraktal boyut;

$$D = \lim_{a \to 0} \frac{logN}{log(\frac{1}{a})} \tag{2.1}$$

ile verilir. Burada D' ye dolambaçlı çizginin "dolambaçlılık derecesi" denir. Burada $log(\frac{1}{a}) = -loga$ dır. log ile de 10 tabanında logaritma kastedilir, fakat başka bir taban da kullanılabilir. Uygulamada a ölçütünü küçük adımlarla, her defasında, herhangi bir sayının bir katı olarak alınır. $\frac{logN}{log(\frac{1}{a})}$ kesri, sabit bir değer olan D limitine yaklaşır. Bazı durumlarda kesrin değeri her adımda aynı olur. O halde (2.1) formülünü

$$D = \lim_{a \to 0} \frac{logN}{log(\frac{1}{a})}, \qquad logN = D.log(\frac{1}{a}) = log(\frac{1}{a})^D \tag{2.2}$$

biçiminde yazılabilir, buradan

$$N = (\frac{1}{a})^D \tag{2.3}$$

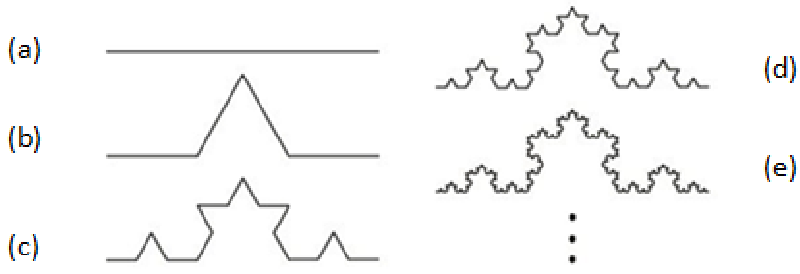
dir. Ölçülen toplam uzunluk Na ise

$$Na = (\frac{1}{a})^{D-1} \tag{2.4}$$

olur. Bu da ölçüt birimi olan a nın azalması halinde ölçülen Na uzunluğunun nasıl arttığını göstermektedir. Bu hesaplama yöntemiyle fraktal boyut hesaplanabilir [9].

Fraktalda çok bilinen ve çok kullanılan Koch kartanesinin fraktal boyutuna bakalım.

Örnek 2.3.1: Koch Eğrisi'nin Boyutunun Hesabı



Şekil 2.6. Koch eğrisi

Öncelikle Koch Eğrisi çizebilmek için bir doğru parçası (Şekil 2.6(a)) çizilerek başlanır, daha sonra doğru parçası üç eş parçaya ayrılır ve ortadaki alınır. Alınan parça, bir eşkenar üçgen şeklinde dışa doğru tamamlanıp dört eş doğru parçadan oluşan bir kırık çizgi elde edilir (Şekil 2.6(b)). Bu kırık çizgiden elde edilen şekle (Şekil 2.6(b)) motif denir. Eğer ilk doğru parçası (Şekil 2.6(a)) 1 br uzunluğunda seçilirse, oluşan motifte her biri $\frac{1}{3}$ uzunluklu dört parçadan oluşur. Dolayısıyla motifin toplam uzunluğu $\frac{4}{3}$ olur.

Benzer şekilde dört parçadan her bir doğru parçası için aynı işlemler tekrarlanarak her biri birer motif haline getirilir. Böylece (Şekil 2.6(c)) elde edilir. Bu oluşan şekilde $4 \times 4 = 16$ tane eş doğru parçası yer alır (Şekil 2.6(c)). Bu eğrinin toplam uzunluğu;

$$\frac{16}{9} = (\frac{4}{3})^2$$

olur. Benzer biçimde devam edilirse 3. adımda $4^3 = 64$ tane ve her birinin uzunluğu $(\frac{1}{3})^3 = \frac{1}{27}$ olan eş doğru parçalarından oluşan bir eğri elde edilir (Şekil 2.6(d)). Bu eğrinin toplam uzunluğu;

$$64\frac{1}{27} = (\frac{4}{3})^3$$

dir. O halde

$$\frac{log4}{log3}, \quad \frac{log4^2}{log3^2}, \quad \frac{log4^3}{log3^3}, \ldots$$

olduğundan (2.2) formülü uygulanır. Buradan

$$D = \frac{log4}{log3} = 1,26...$$

olarak bulunur ve bu da Koch kartesinin fraktal boyutudur [9].

2.3.1. Fraktal boyutta kutu sayma metodu

Bir eğrinin fraktal boyutunu ölçmenin yollarından biri de kutu sayma metodudur. Kutu sayma metodunda eğri, küçük karelerden oluşan bir kafes ile örtülür ve sonra eğrinin üstünden geçen kareler sayılır. Kareler giderek küçültülerek tekrarlanır ve küçüldükçe ölçüm hassaslaşır. Örneğin şekil, kenar uzunluğu r olan karelerle örtülür ve kaç tane kare kullanıldığı $N(r)$ ile gösterilir. $N(r)$'nin r'ye nasıl bağlı olduğuna bakılırsa yöntem sağlanır. Şekil bir doğru parçası gibi 1 boyutlu ise $N(r) = \frac{1}{r}$ olur. Burada r yerine $\frac{1}{r}$ alınmalıdır ki kareler küçüldükçe şekli örtmek için daha çok kare kullanılabilsin. Eğer şekil içi dolu bir kare ise parça sayısı $N(r) = (\frac{1}{r})^2$ olur.

Daha karmaşık şekiller için ise $N(r)$ ile $\frac{1}{r}$ arasındaki ilişkide daha karmaşıktır. Eğer $N(r)$'nin yaklaşık olarak $(\frac{1}{r})^d$ olacağı tahmin edilirse, $N(r) = (\frac{1}{r})^d$'nin logaritması alınarak $logN(r) = log(\frac{1}{r})^d$ bulunur ve bu eşitlik $logN(r) = dlog(\frac{1}{r})$'ye eşittir.

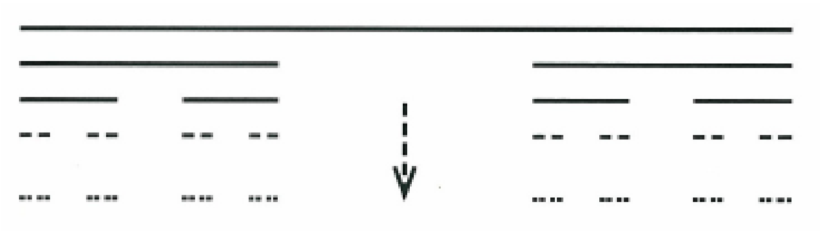
Buradan,

$$d = \lim_{i \to 0} \frac{log(N(r_i))}{log(\frac{1}{r_i})}$$

elde edilir. Eğer bu limit değeri varsa d sayısı fraktalın kutu sayma boyutudur [10].

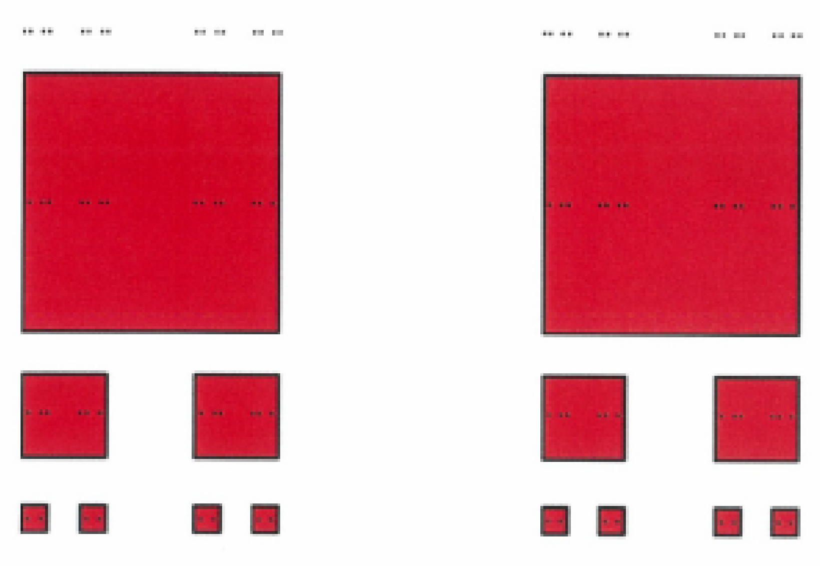
Kutu sayma metoduyla ilgili fraktalın bilinen bir kaç örneğini inceleyelim.

Örnek 1: Kantor Orta Üçlülerinin Cümlesi: İlk olarak Kantor Orta Üçlüsü oluşturulursa, birim doğru parçası üç eşit parçaya bölünür ve ortadaki üçte birlik parça atılır, daha sonra geriye kalan iki parçaya da aynı işlem uygulanıp ortalarındaki üçte birlik parçaların atılır ve tekrar geriye kalan dört parçanın her biri için aynı işlemler yapılıp sonra ortadaki parçaları atılır ve bu işleme devam edilmesiyle (Şekil 2.7) oluşturulur.



Şekil 2.7. Kantor orta üçlüleri

Kantor cümlesinin kutu sayma boyutunu hesaplamak için git gide küçülen kutularla Kantor cümlesi örtülür (Şekil 2.8).



Şekil 2.8. Kantor orta üçlülerinin örtülmesi

Burada kenar uzunluğu r_1 olan karelerle örtülür ve kaç tane kare kullanıldığı $N(r)$ ile gösterilir. Ardından doğru parçasının üç eşit parçaya bölünmesiyle devam edilir. Devamlı olarak 3'e bölünmeye devam eder.

Burada kareler küçüldükçe şekli örtebilmek adına daha çok kare kullanılması için r yerine $\frac{1}{r}$

alınmalıdır. İlk üç adım aşağıdaki gibidir;

$$r_1 = \frac{1}{3}, \quad N(r_1) = 2$$
$$r_2 = \frac{1}{9}, \quad N(r_2) = 4$$
$$r_3 = \frac{1}{27}, \quad N(r_3) = 8$$

$$N(1/3) = 2$$
$$N(1/9) = N((1/3)^2) = 4 = 2^2$$
$$N(1/27) = ((1/3)^3) = 8 = 2^3$$

dir ve genel olarak fraktal boyut

$$N((1/3)^n) = 2^n$$

olarak bulunur.

Bu fraktalın kutu sayma boyutu hesaplanırsa:

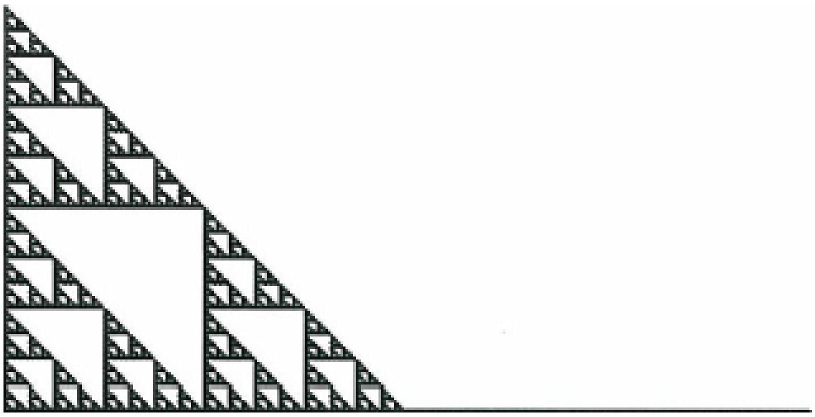
$$\begin{aligned} d_k = \frac{log(N((1/3)^n))}{log(1/(1/3)^n)} &= \frac{log(2^n)}{log(3^n)} \\ &= \frac{nlog2}{nlog3} \\ &= \frac{log2}{log3} \end{aligned}$$

dür. Buradan

$$\begin{aligned} d_k &= \lim_{n\to\infty} \frac{log(N((1/3)^n))}{log(1/(1/3)^n)} \\ &= \frac{log2}{log3} \\ &= 0.62989 \end{aligned}$$

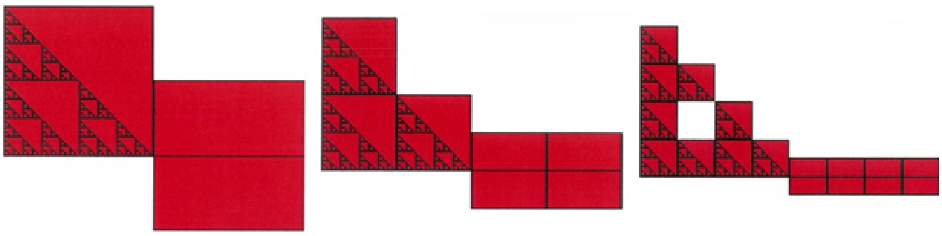
bulunur[11].

Örnek 2: Sierpinski Şapkası ve Doğru Parçasının Birleşimi: Şapka ve doğru parçası yanyana getirildiğinde oluşan (Şekil 2.9)'un kutu sayma boyutu hesaplanabilir.



Şekil 2.9. Sierpinski şapkası ve doğru parçasının birleşimi

Kutu sayma boyutunu hesaplamak için, bu şekil git gide küçülen kutularla örtülür (Şekil 2.10).



Şekil 2.10. Sierpinski şapkası ve doğru parçasının birleşiminin örtülmesi

Burada kenar uzunluğu r_0 olan karelerle örtülür ve kaç tane kare kullanıldığı $N(r_0)$ ile gösterilir. Ardından her adımda 2'ye bölünerek devam eder. Burada kareler küçüldükçe şekli örtebilmek adına daha çok kare kullanılması için r yerine $\frac{1}{r}$ alınmalıdır. Bu şekilde ilk 3 adım verilir ve genel bir formül aşağıdaki gibi elde edilir;

$$r_0 = 1, \quad N(r_0) = 2$$

$$r_1 = \frac{1}{2}, \quad N(r_1) = 5$$

$$r_2 = \frac{1}{4}, \quad N(r_2) = 13$$

$$N(1) = 2 = 1 + 1$$

$$N(1/2) = 5 = 3 + 2$$

$N(1/4) = ((1/2)^2) = 13 = 9+4 = 3^2+2^2$

ve genel olarak

$N((1/2)^n) = 3^n + 2^n$

bulunur. Bu bağlantıyla kutu sayma metodu hesaplanırsa;

$$3^n + 2^n = 3^n(1+(\frac{2}{3})^n)$$
$$log(3^n+2^n) = log(3^n(1+(\frac{2}{3})^n)) = log(3^n)+log(1+(\frac{2}{3})^n)$$
$$\lim_{n\to\infty} log(1+(\frac{2}{3})^n) = log1 = 0$$

olup

$$\begin{aligned} d_k &= \lim_{n\to\infty} \frac{log(N((1/2)^n))}{log(1/(1/2)^n)} \\ &= \lim_{n\to\infty} \frac{log(3^n+2^n)}{log2^n} \\ &= \lim_{n\to\infty} \frac{log(3^n(1+(\frac{2^n}{3^n}))}{2^n} \\ &= \lim_{n\to\infty} \frac{log3^n}{log2^n} \\ &= \lim_{n\to\infty} \frac{log3}{log2} \end{aligned}$$

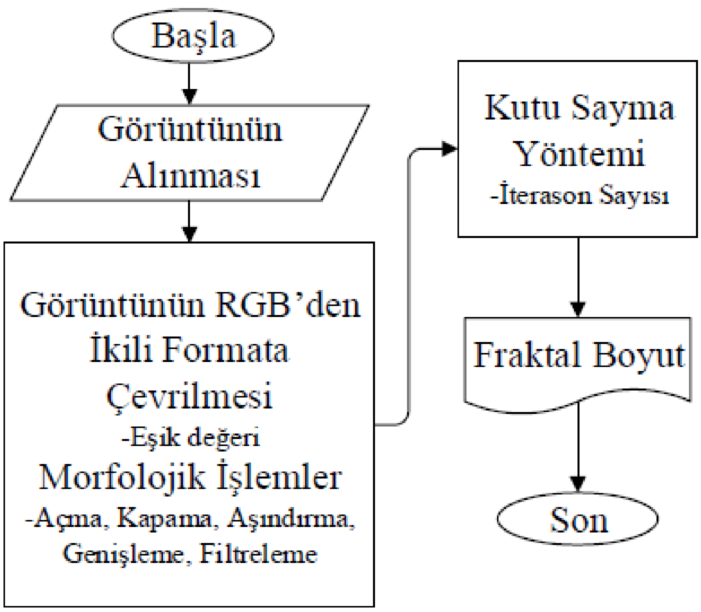
olarak bulunur [12].

2.3.2. MATLAB programında fraktal boyut hesabı

Fraktalın bir çok uygulama alanı ve tanımlaması olmasına rağmen en çok pürüzlü yapılarda kullanılmaktadır. Bu pürüzlü yapıların boyutlarının net olarak ölçülememesi ve yaklaşık değerler bulunabilmesi bu konunun gelişmesine yardımcı olmuştur. Matematiksel olarak boyut hesaplarken en çok kullanılan yöntem kutu sayma metodur. Bu yöntemden Bölüm 2.3.1 'de bahsedilmektedir.

Son yıllarda arayüz programları işlemleri kolaylaştırmak amacıyla çok yaygın bir şekilde kullanılmaktadır. Arayüz programı oluşturmak için MATLAB Guide User Interface (GUI) programı ile hazırlanan bir çok çalışma vardır. MATLAB GUI programı, MATLAB'ın gelişmiş analiz ve grafik özelliklerini kullanabildiği için çok esnek ve kullanışlı bir yapıya sahiptir. MATLAB GUI ile tasarlanan arayüz programının kullanılabilmesi için tasarlanan program deploytool paketi kullanarak *.exe dosyasına dönüştürülebilmektedir. Dolayısıyla oluşturulan *.exe dosyasını kullanıcı bilgisayarına kurduğunda ücretsiz olarak kullanabilecektir. Bu sebeplerle kutu sayma metodundan daha hızlı sonuç veren bir arayüz programı geliştiren çalışmalar ülkemizde de mevcuttur.

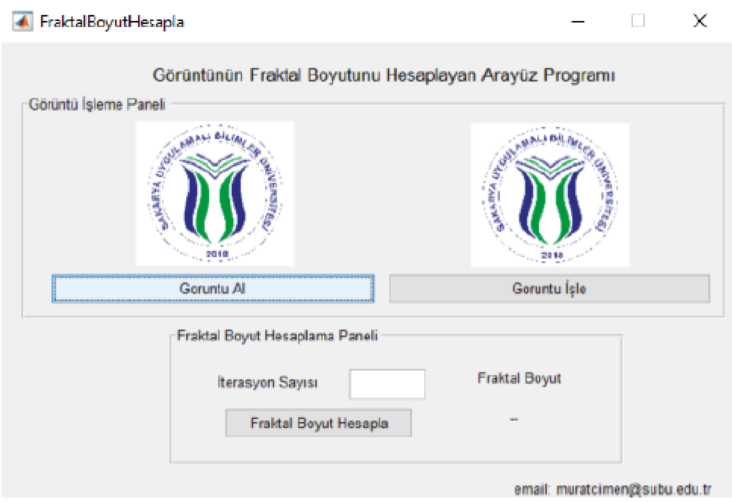
Burada anlatılan çalışmada, Matlab GUI ile hazırlanan kullanıcı dostu arayüz sayesinde kullanıcı, görüntü üzerinde ortalama filtre, medyan filtre, adaptif eşikleme, açma, kapama ve aşındırma gibi morfolojik işlemleri kullanarak görüntü üzerinde bulunan nesneleri rahatlıkla segmente edebilmektedir. Ardından görüntü üzerinden segmente edilen nesnelerin yine arayüz üzerinden kutu sayma yöntemi ile fraktal boyutu kolaylıkla hesaplanıp görüntülenebilmektedir.



Şekil 2.11. Akış şeması

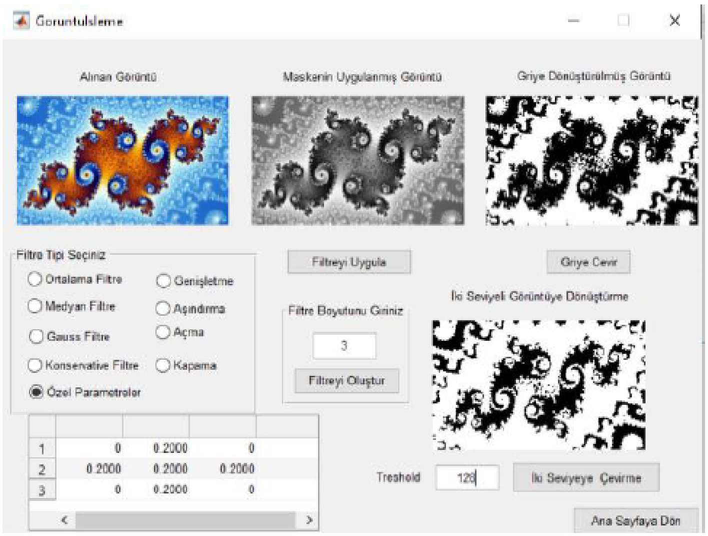
Şekil 2.11 'de verilen akış diyagramında görüldüğü üzere ilk olarak görüntü programa alınır. İkinci adımda fraktal boyutu hesaplanacak olan görüntü çeşitli ön ve ileri görüntü işleme tekniklerine tabi tutulur. Bu adımda alınan RGB formatındaki görüntünün ikili formata dönüştürülmesi veya alınan görüntü üzerinde gürültüleri gidermek, görüntüyü netleştirmek ve asıl işlenmek istenen detayları ortaya çıkarmak için morfolojik işlemler olarak adlandırılan görüntü işleme teknikleri uygulanmaktadır. Bu işlemler yapılıp fraktal nesne belirginleştirildikten sonra fraktal boyutunun hesaplanabilmesi için kutu sayma tekniği uygulanmakta ve sonucu görüntülenmektedir. Bununla ilgili Dr. Murat Çimen'in geliştirdiği programdan görsellerle aşağıdaki açıklanmaktadır [13].

İlk olarak arayüz programının başlangıç sayfası görülmektedir (Şekil 2.12).



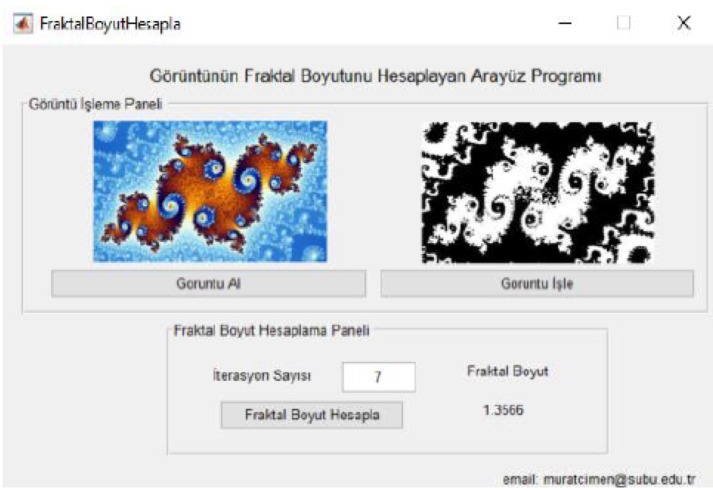
Şekil 2.12. Fraktal boyut arayüz programı başlangıç sayfası

Sonraki adımda fraktal boyutu hesaplanmak istenen görüntü için "Görüntü Al" tuşuna basılarak örnek bir görüntü eklenir. Görüntü işleme adımına geçilerek "Görüntü İşlem" arayüzü açılır. Görüntü üzerinde yapılmak istenen filtreleme ve morfolojik işlemler bu arayüzde gerçekleştirilir. Eklenen örnek görüntü üzerinde yapılacak olan işlem tipi seçilir ve "Filtre Boyutunu Giriniz" kısmının altında "edit text" içine girilen boyuta göre filtrenin boyutu belirlenir. Ardından "Filtreyi Oluştur" tuşuna basılır ve filtre tabloda görüntülenir. Bu işlemden sonra seçilen işleme göre "İşlemi Uygula" butonuna tıklanır ve görüntüye morfolojik işlem uygulanır. Ardından "axes"te görüntülenir. "Griye çevir butonuna" basıldığında görüntü gri seviyeye çevrilir ve görüntülenir. "İkili Seviyeye Çevir" butonuna basıldığında da "treshold" değerine göre görüntü ikili seviyeye çevrilir ve görüntülenir. Bu ekrandan tekrardan ana sayfaya geçilmek için "Ana Sayfaya Dön" butonuna basılır bu ekran kapanarak daha önceden açılan son penceredeki ana sayfa açılır. Fraktal boyutu hesaplanacak olan görüntüde kaç iterasyon yapılacak ise son pencerede görüldüğü gibi iterasyon sayısı girilir ve "Fraktal Boyut Hesapla" butonuna tıklandığında arayüzün arka tarafında fraktal boyutu hesaplanır. Görüntünün hesaplanan fraktal değeri, arayüzde bulunan "edit text" kısmına yazılmaktadır (Şekil 2.13).



Şekil 2.13. Fraktal boyut arayüz programı boyut hesabı

Arayüzde gerçekleştirilen örnek bir uygulama Şekil 2.14'deki gibidir. Şekil 2.13'te işlenen görüntü ve fraktal değeri hesaplanan görüntünün son hali Şekil 2.14'te verilmektedir. Farklı bir görüntünün fraktal değeri hesaplanmak istendiğinde yine "Görüntü Al" tuşuna basılarak yeni görüntü eklenir ve bu işlem adımları tekrarlanır.



Şekil 2.14. Fraktal boyut arayüz programı fraktal boyutu

Böylece istenilen fraktalın kutu sayma metoduyla fraktal boyutu hesaplanmış olur.

2.4. Fraktalda Newton Metodu

Newton metodu ya da diğer bilinen bir ismi ile Newton-Raphson metodu kök bulma probleminde kullanılan metotlardan biridir. Sürekli ve farklılaşabilir bir fonksiyonun, o fonksiyona teğet düz bir çizgi ile yaklaşabileceği fikrinden ortaya çıkar. Fikir, gerçek köke uygun bir şekilde yakın olan bir başlangıç tahmini ile başlar, daha sonra teğet çizgisine yaklaşıklaştırmak ve bu teğet çizginin x eksenini kesişimini temel cebir işlemi ile hesaplayarak devam eder. Bu x ekseninin kesme noktası orijinal fonksiyonun köküne ilk tahminden daha iyi bir yaklaşım olacaktır ve yöntem tekrarlanabilir [14]. Newton-Raphson yöntemi, Taylor serisinin ilk birkaç terimini kullanan bir kök bulma algoritmasıdır [15]. $f\varepsilon C^2[a,b]$ olsun. $f'(p_0) \neq 0$ ve $|p-p_0|$ farkı yeterince küçük olmak üzere p kök değerine $f\varepsilon[a,b]$ gibi bir yaklaşım yapılsın. $f(x)$ fonksiyonunun p_0 civarında birinci Taylor polinomunu $\xi(x)$ sayısı x ile p_0 arasında olmak üzere

$$f(x) = f(p_0) + (x-p_0)f'(p_0) + \frac{(x-p_0)^2}{2}f''(\xi(x))$$

şeklinde yazılabilir. Burada eğer $x = p$ alınırsa $\xi(p)$ sayısı p ile p_0 arasında olmak üzere

$$f(x) = f(p_0) + (p-p_0)f'(p_0) + \frac{(p-p_0)^2}{2}f''(\xi(p))$$

elde edilir. $f(p) = 0$ olduğundan

$$0 = f(p_0) + (x-p_0)f'(p_0) + \frac{(x-p_0)^2}{2}f''(\xi(x))$$

eşitliğine ulaşılır. Newton metodu $|p-p_0|$ farkının çok küçük olduğu varsayımı altında $(p-p_0)^2$ değerinin çok daha küçük olması olgusuna dayanır. Buna göre oluşan hata ihmal edilebilir bir büyüklüktedir.

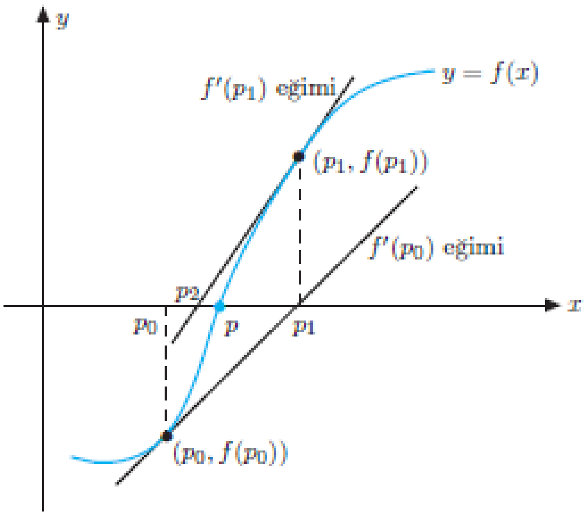
Dolayısıyla $0 \approx f(p_0) + (x-p_0)f'(p_0)$ yazılabilir. Bu ifade p'ye göre düzenlenirse

$$p \approx p_0 - \frac{f(p_0)}{f'(p_0)} \equiv p_1, f'(p_0) \neq 0$$

elde edilir. Özyinelemeli olarak $n \geq 1$ için $\{p_n\}_{n=0}^{\infty}$ dizisi

$$p_n = p_{n-1} - \frac{f(p_{n-1})}{f'(p_{n-1})} \tag{2.5}$$

şeklinde tanımlanırsa p_0 başlangıç yaklaşımı olmak üzere Newton metodu elde edilmiş olur (Şekil 2.15).



Şekil 2.15. Newton metodu

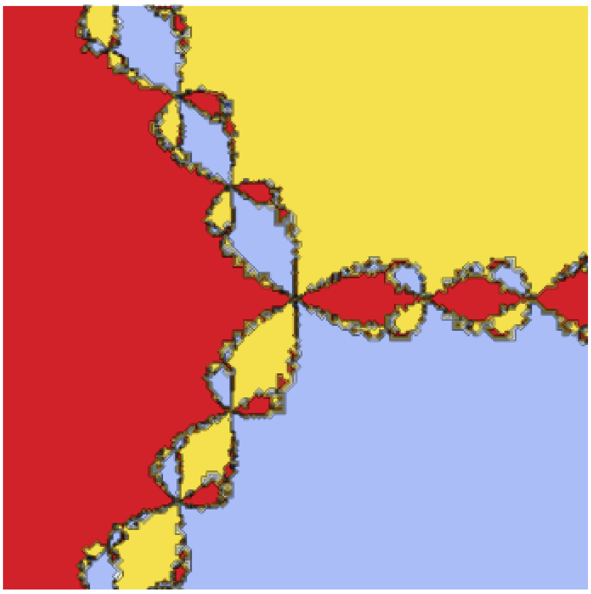
Fraktalda Newton Metodu ise Julia ve Mandelbrot'un başlangıç yollarıdır. Temel bir Newton metodu formülünü şekillendirip çoğaltarak sonsuza giden iterasyonlar elde etmek ve bunları şekil dökmek mümkündür [16, 17].

Newton metodunu Mathematica programında kullanarak oluşturulan fraktal örnekler aşağıdadır:

Örnek 1: İlk olarak bir fonksiyon tanımlanır. Bu örnekte fonksiyon $f(z) = z^3 + 2$ alınsın. Newton metodu fonksiyon üzerine uygulansın ve aralıklar $(x, -1.8, 1.8)$ ve $(y, -1.8, 1.8)$ olarak alınsın. Mathematica programında veilen fonksiyonlar ve aralıklar aşağıdaki gibi yazılırsa (Şekil 2.16) çizdirilir.

```
theFunct[z_] := z^3 + 2;
iterate[z_] :=
```

```
z -(theFunct[z] - 1)/theFunct'[z];
ContourPlot[Arg[FixedPoint
[iterate, x + I y, 25]],
{x, -1.8, 1.8},
{y, -1.8, 1.8},
ColorFunction -> "TemperatureMap",
Axes -> False,
Frame -> False]
```

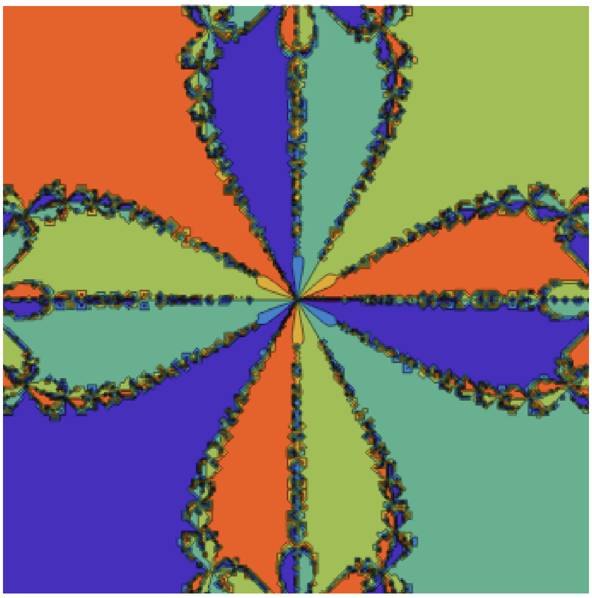


Şekil 2.16. Newton metodunun mathematica çizimi 1

Örnek 2: İlk olarak bir fonksiyon tanımlanır. Bu örnekte fonksiyon $f(z) = z^4 + 4$ alınsın. Newton metodu fonksiyon üzerine uygulansın ve aralıklar $(x, -1, 1)$ ve $(y, -1, 1)$ olarak alınsın. Mathematica programında veilen fonksiyonlar ve aralıklar aşağıdaki gibi yazılırsa (Şekil 2.17) çizdirilir.

```
f[z_] := z^4 + 4;
iterasyon[z_] :=
z - (f[z]/f'[z]);
ContourPlot[Arg[FixedPoint
[iterasyon, x + I y, 20]],
```

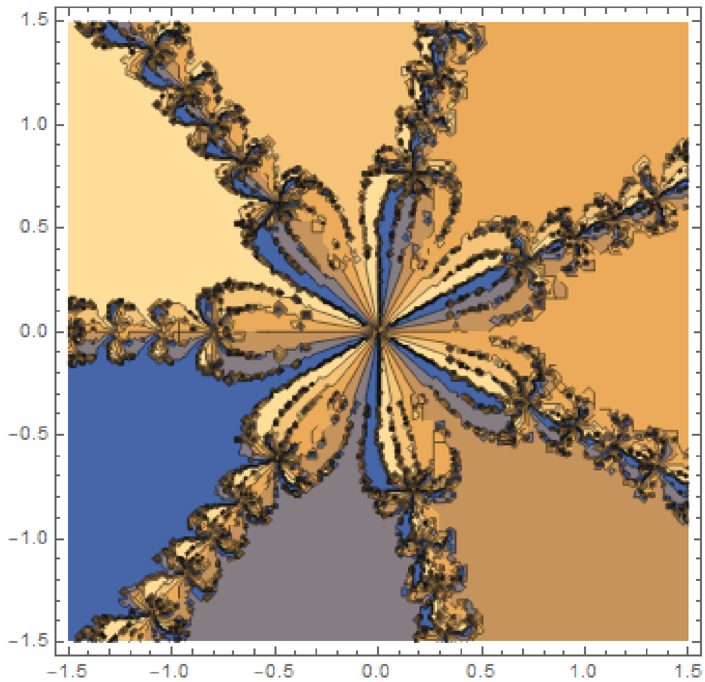
```
{x, -1, 1},
{y, -1, 1},
ColorFunction -> "Rainbow",
Axes -> False,
Frame -> False]
```



Şekil 2.17. Newton metodunun mathematica çizimi 2

Örnek 3: İlk olarak bir fonksiyon tanımlanır. Bu örnekte fonksiyon $f(z) = z^7$ alınsın. Newton metodu fonksiyon üzerine uygulansın ve aralıklar $(x, -1.5, 1.5)$ ve $(y, -1.5, 1.5)$ olarak alınsın. Mathematica programında veilen fonksiyonlar ve aralıklar aşağıdaki gibi yazılırsa (Şekil 2.18) çizdirilir.

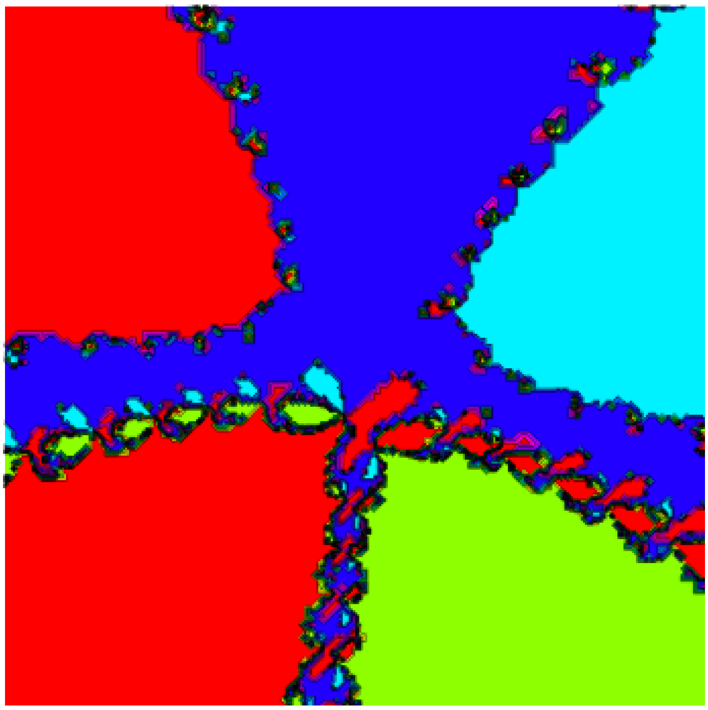
```
theFunc[z_] := z^7;
iterateThis[z_] :=
z - (theFunc[z] - 1)
/theFunc'[z];
ContourPlot[Arg[FixedPoint
[iterateThis, x + I y, 30]],
{x, -1.5, 1.5},
{y, -1.5, 1.5}]
```



Şekil 2.18. Newton metodunun mathematica çizimi 3

Örnek 4: İlk olarak bir fonksiyon tanımlanır. Bu örnekte fonksiyon $f(z) = z^5 + z^4 \times (-0.23 + 0.23I) + 0.12 \times z^3 + z^2 \times (-0.107 - 0.281I) + z \times (0.192 - 0.075I) - 0.123 - 0.17I$ alınsın. Newton metodu fonksiyon üzerine uygulansın ve aralıklar $(x, -1.8, 1.8)$ ve $(y, -1.8, 1.8)$ olarak alınsın. Mathematica programında veilen fonksiyonlar ve aralıklar aşağıdaki gibi yazılırsa (Şekil 2.19) çizdirilir.

```
P[z_] := z^5 + z^4*(-0.23 + 0.23 I)
+ 0.12*z^3 + z^2*(-0.107 - 0.281 I)
+ z*(0.192 - 0.075 I)
- 0.123 - 0.17 I;
Newt[z_] := z - (P[z]/P'[z]);
ContourPlot[Arg
[FixedPoint[Newt, x + I y, 25]],
{x, -1.8, 1.8},
{y, -1.8, 1.8},
ColorFunction -> "Rainbow",
Axes -> False,
Frame -> False]
```



Şekil 2.19. Newton metodunun mathematica çizimi 4

2.5. Bilgisayar Programlarında Fraktal Çizimler

2.5.1. Latex programında çizimler

Latex programını kullanılırken ilk olarak hangi ortamda yazılacağına karar verilmelidir. Genel olarak "documentclass{ortam}" yazarak başlanır. Daha sonra içeride komutları kullanabilmek için bir document girişi yapılır. Bu giriş "begin{document}" komutudur ve "end{document}" komutu arasına istenilen komutlar yazılır.

Bir görsel çizilmek istenirse komutlar "begin{tikzpicture}" ve "end{tikzpicture}" arasına yazılır. Ya da herhangi bir resim eklenmek istenirse "begin{figure}" ve "end{figure}" arasına yazılır ve "includegraphics{resimismi}" ile resim çağrılır. Burada önemli bir detay vardır ki oda resim ile yazılan latex dosyası aynı klasörde olmalı farklı bir klasörde olması halinda çağrılan resim gösterilemez.

Şekillerin altına yazılan "caption{isim}" komutuyla şekile bir isim ve bir numara tanımlanır. Bu resim ekleme komutunu yazılırken yanına köşeli parantez ile konumu belirtilebilir.

Bunun için kısayollar tablo halinde gösterilirse aşağıdaki gibidir;

Çizelge 2.1. Latex kısayolları

Veri Hizalama	Açıklama
h	buraya, olan yere hizalama
!	mutlaka diğer seçeneklerden birine hizalama
t	üste hizalama
b	alta hizalama
p	sayfaya hizalama

Programda herhangi bir döngü oluşturabilmek için verilen her değere kullanmak amacıyla "foreach" komutu kullanılır. "draw" komutu yazılan kodları çizdirmek için kullanılır. Genel olarak kullanılan bu komutlar dışında çizilmek istenen her şekil için ayrı bir komut dizisi kullanılmaktadır.

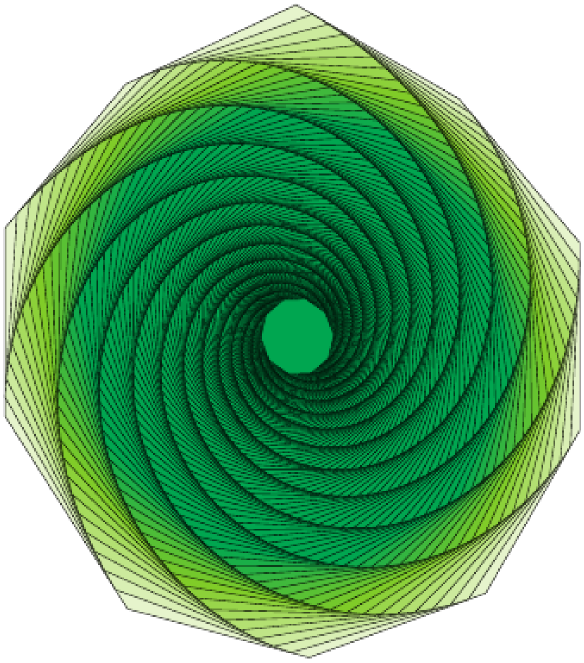
Latex programını kullanarak çizilen fraktal şekiller aşağıda verilmektedir [18, 19]. Bunun için önce latex programında kodlar yazılır ve pdf dosyası yapmak için F5 ’e tıklanıldığında şekili bilgisayar çizer.

Örnek 1:

```
\documentclass{article}
\usepackage[usenames,dvipsnames,pdftex]{xcolor}
\usepackage{tikz}
\usepackage{ifthen}
\begin{document}
\newcounter{density}
\setcounter{density}{20}
\begin{tikzpicture}
\def\couleur{LimeGreen}
\path[coordinate]
(0,0)  coordinate(A)
++(90:4cm) coordinate(B)
++(50:4cm) coordinate(C)
++(25:4cm) coordinate(D)
++(-30:4cm) coordinate(E)
```

```
++(-50:4cm) coordinate(F)
++(-90:4cm) coordinate(G)
++(-120:4cm) coordinate(H)
++(-160:4cm) coordinate(I)
++(1965:4cm) coordinate(J);
\draw[fill=\couleur!\thedensity]
(A) -- (B) -- (C) --(D)
 -- (E) -- (F)-- (G) --
 (H) -- (I) -- (J) -- cycle;
\foreach \x in {1,...,120}{
\pgfmathsetcounter{density}
{\thedensity+10}
\setcounter{density}{\thedensity}
\path[coordinate]
coordinate(X) at (A){};
\path[coordinate]
(A) -- (B)
coordinate[pos=.10](A)
-- (C) coordinate[pos=.10](B)
-- (D) coordinate[pos=.10](C)
-- (E) coordinate[pos=.10](D)
-- (F) coordinate[pos=.10](E)
-- (G) coordinate[pos=.10](F)
-- (H) coordinate[pos=.10](G)
-- (I) coordinate[pos=.10](H)
-- (J) coordinate[pos=.10](I)
-- (X) coordinate[pos=.10](J);
\draw[fill=\couleur!\thedensity]
(A)--(B)--(C)-- (D) --(E) -- (F)--
(G) -- (H) -- (I) -- (J) -- cycle;}
\end{tikzpicture}
```

```
\end{document}
```

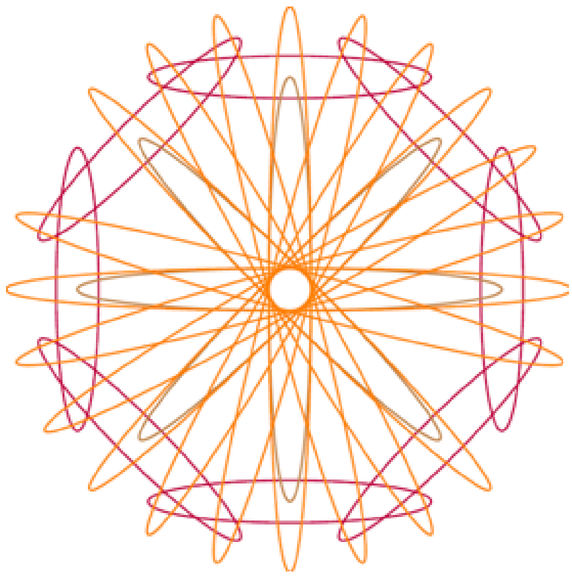


Şekil 2.20. Latex çizim örnekleri 1

Örnek 2:

```
\documentclass{standalone}
\usepackage{tikz}
\begin{document}
\begin{tikzpicture}[scale=1.25]
\foreach \angled
in {0,45,...,135}
\draw[rotate=\angled,line width =1pt, draw=purple]
(0,-3) ellipse(2 and 0.3);
\foreach \anglef in {0,45,...,135}
\draw[rotate=\anglef,line width =1pt, draw=purple]
(0,3) ellipse(2 and 0.3);
\foreach \angleg in {0,45,...,135}
\draw[rotate=\angleg,line width =1pt, draw=brown]
(0,0) ellipse(3 and 0.3);
\foreach \angleh in {-15,0,...,135}
```

```
\draw[rotate=\angleh,line width =1pt, draw=orange]
(0,0) ellipse(4 and 0.3);
\node[scale=0.5, text=red] at (-90:0.02) {};
\end{tikzpicture}
\end{document}
```

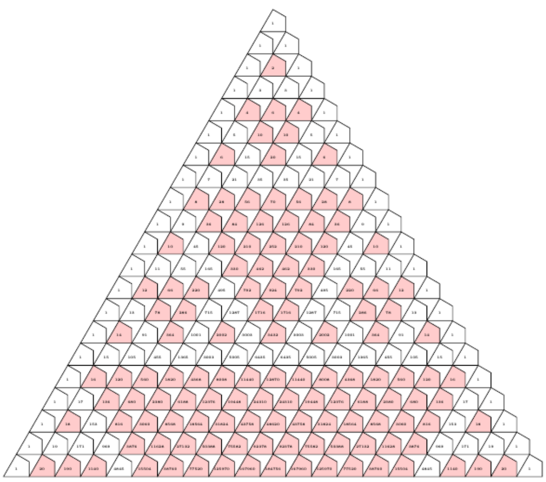


Şekil 2.21. Latex çizim örnekleri 2

Örnek 3:

```
\documentclass[tikz,border=0.25in]{standalone}
\usepackage{fp}
\newlength{\R}
\setlength{\R}{.75cm}
\newcommand\mycolor{white}
\newcommand{\modulo}[2]{%
\FPeval{\result}{trunc
(#1-(#2*trunc(#1/#2,0)),0)}\result%}
\newcommand{\makesierp}[2][2]{%
\begin{tikzpicture}[line width=.8pt]
\foreach \k in {0,...,#2}{
\begin{scope}[shift={(-60:{sqrt(3)*\R*\k})}]
```

```
\pgfmathtruncatemacro\ystart{#2-\k}
\foreach \n in {0,...,\ystart}{
\pgfmathtruncatemacro\newn{\n+\k}
\pgfmathsetmacro{\myvalue}{1}
\ifnum\k>0
\foreach \b in {1,...,\k}{
\FPeval\myvalue{trunc (\myvalue*(\newn-\b+1)/\b:0)}
\global\let\myvalue=\myvalue}
\fi \modulo{\myvalue}{#1}%
\ifnum\result=0 \def\mycolor{red}\fi%
\begin{scope}[shift={(-120:{sqrt(3)*\R*\n})}]
\draw[fill=\mycolor!20]
(30:\R) \foreach \x in {90,210,...,360}
{-- (\x:\R)} --cycle (90:0)node[font=\tiny]
{$\mathbf{\myvalue}$};
\end{scope}}
\end{scope}}
\end{tikzpicture}}
\begin{document}
\makesierp{20}
\end{document}
```

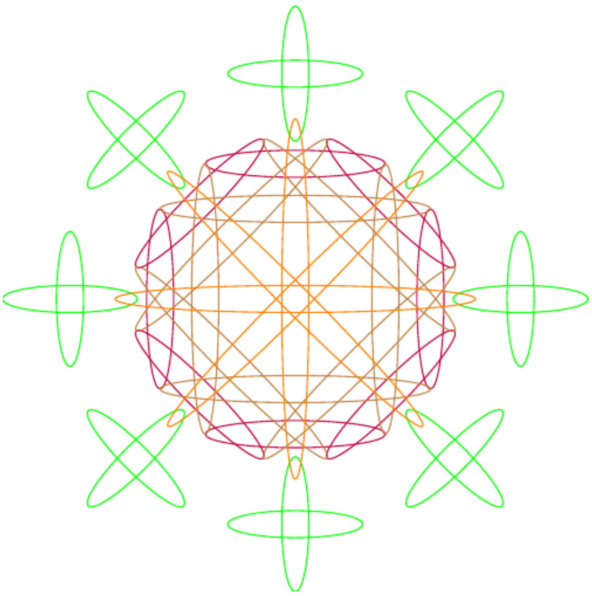


Şekil 2.22. Latex çizim örnekleri 3

Örnek 4:

```
\documentclass{standalone}
\usepackage{tikz}
\begin{document}
\begin{tikzpicture}[scale=1.25]
\foreach \angled in {0,45,...,135}
\draw[rotate=\angled,line width =1pt,
draw=purple] (0,-3) ellipse(2 and 0.3);
\foreach \anglee in {0,45,...,135}
\draw[rotate=\anglee,line width =1pt,
draw=green] (0,-5) ellipse(1.5 and 0.3);
\foreach \anglee in {0,45,...,135}
\draw[rotate=\anglee,line width =1pt,
draw=green] (0,5) ellipse(1.5 and 0.3);
\foreach \anglee in {0,45,...,135}
\draw[rotate=\anglee,line width =1pt,
draw=green] (-5,0) ellipse(1.5 and 0.3);
\foreach \anglee in {0,45,...,135}
\draw[rotate=\anglee,line width =1pt,
draw=green] (5,0) ellipse(1.5 and 0.3);
\foreach \anglef in {0,45,...,135}
\draw[rotate=\anglef,line width =1pt,
draw=purple] (0,3) ellipse(2 and 0.3);
\foreach \angleg in {0,45,...,135}
\draw[rotate=\angleg,line width =1pt,
draw=brown] (0,2) ellipse(3 and 0.3);
\foreach \angleg in {0,45,...,135}
\draw[rotate=\angleg,line width =1pt,
draw=brown] (0,-2) ellipse(3 and 0.3);
\foreach \angleh in {0,45,...,135}
\draw[rotate=\angleh,line width =1pt,
draw=orange] (0,0) ellipse(4 and 0.3);
```

```
\node[scale=0.5, text=red] at (-90:0.02) {};
\end{tikzpicture}
\end{document}
```



Şekil 2.23. Latex çizim örnekleri 4

Örnek 5:

```
\documentclass{minimal}
\usepackage{tikz}
\begin{document}
\begin{tikzpicture}[cap=round]
\colorlet{anglecolor}{yellow!50!black}
\colorlet{bordercolor}{black}
\def\alpha{6}
\def\layer{6}
\begin{scope}[scale=4]
\pgfmathsetmacro\sinTriDiff{sin(18-\alpha)}
\pgfmathsetmacro\cosTriDiff{1-cos(73-\alpha)}
\pgfmathsetmacro\radiusC{sqrt(\cosTriDiff
*\cosTriDiff + \sinTriDiff*\sinTriDiff)}
\pgfmathsetmacro\startAng{\alpha +
```

```
atan(\sinTriDiff/\cosTriDiff)}
\pgfmathsetmacro\al{\alpha*\layer}
\foreach \x in {0,\alpha,...,\al}
{
\pgfmathsetmacro\ang{\x + \startAng}
\pgfmathsetmacro\xRs{\radiusC*cos(\ang)}
\pgfmathsetmacro\yRs{\radiusC*sin(\ang)}
\pgfmathsetmacro\radiusLayer{\xRs+sqrt(1-\yRs*\yRs )}
\pgfmathsetmacro\angRs{acos(\yRs)}
\pgfmathsetmacro\angRss{acos(\radiusC*sin(\ang-\alpha))}
\colorlet{anglecolor}{yellow!\ang!red}
\pgfmathsetmacro\step{2*\alpha - 180}
\pgfmathsetmacro\stop{180-2*\alpha}
\foreach \y in
{-180, \step ,..., \stop}
{
\pgfmathsetmacro\deltaAng{\y-\x}
\filldraw[color=anglecolor,draw=bordercolor]
(\y-\x:\radiusLayer)
arc (-90+\angRs+\deltaAng :
\alpha-90+\angRss+\deltaAng :1)
arc (\alpha+90-\angRss+\deltaAng :
2*\alpha+90-\angRs+\deltaAng :1)
arc (\deltaAng+2*\alpha :\deltaAng:\radiusLayer);}
}
\pgfmathsetmacro\xRs{\radiusC*cos(\al+\startAng)}
\pgfmathsetmacro\yRs{\radiusC*sin(\al+\startAng)}
\pgfmathsetmacro\radiusLayer{\xRs+sqrt(1-\yRs*\yRs)}
\draw[line width=2, color=bordercolor]
(0,0) circle (.8*\radiusLayer);
\pgfmathsetmacro\radiusSmall{.7*\radiusLayer}
```

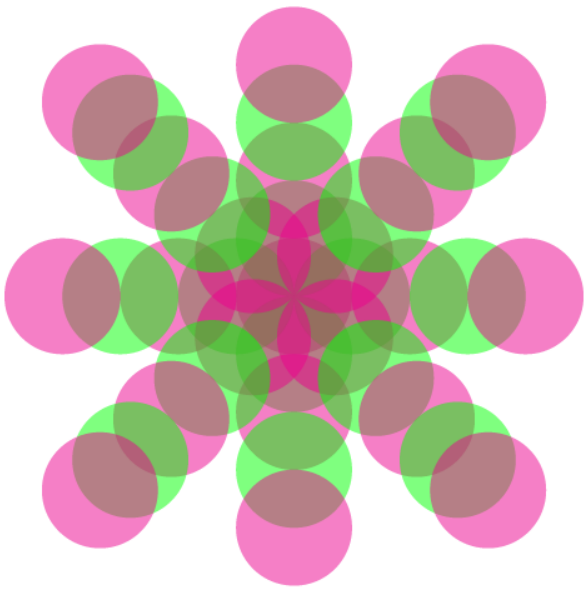
```
\foreach \x in {-60,0,...,240}
{\fill[color=anglecolor] (\x:\radiusSmall)
arc (-180+\x+60: -180+\x: \radiusSmall)
arc (0+\x: -60+\x: \radiusSmall)
arc (120+\x: 60+\x: \radiusSmall);
}
\foreach \x in {0, 4, ..., 360}
{\fill[color=anglecolor] (\x:1) -- (\x+3:1.05) --
(\x+5:1.05) -- (\x+2:1) -- cycle;
\fill[color=anglecolor] (\x+5:1.05) -- (\x+7:1.05) --
(\x+4:1.1) -- (\x+2:1.1) -- cycle;
}
\draw[line width=1, color=bordercolor]
(0,0) circle (1);
\draw[line width=1, color=bordercolor]
(0,0) circle (1.1);
\end{scope}
\end{tikzpicture}
\end{document}
```

Şekil 2.24. Latex çizim örnekleri 5

Örnek 6:

```
\documentclass{minimal}
\usepackage{tikz}
\begin{document}
\begin{tikzpicture}[fill opacity=0.5]
\fill[magenta] (0,0) circle (1);
\fill[green] (1,0) circle (1);
\fill[green] (-1,0) circle (1);
\fill[green] (0,1) circle (1);
\fill[green] (0,-1) circle (1);
\fill[magenta] (0.709,0.709) circle (1);
\fill[magenta] (0.709,-0.709) circle (1);
\fill[magenta] (-0.709,0.709) circle (1);
\fill[magenta] (-0.709,-0.709) circle (1);
\fill[magenta] (2,0) circle (1);
\fill[magenta] (-2,0) circle (1);
\fill[magenta] (0,2) circle (1);
\fill[magenta] (0,-2) circle (1);
\fill[green] (1.414,1.414) circle (1);
\fill[green] (1.414,-1.414) circle (1);
\fill[green] (-1.414,1.414) circle (1);
\fill[green] (-1.414,-1.414) circle (1);
\fill[green] (3,0) circle (1);
\fill[green] (-3,0) circle (1);
\fill[green] (0,3) circle (1);
\fill[green] (0,-3) circle (1);
\fill[magenta] (2.121,2.121) circle (1);
\fill[magenta] (2.121,-2.121) circle (1);
\fill[magenta] (-2.121,2.121) circle (1);
\fill[magenta] (-2.121,-2.121) circle (1);
\fill[magenta] (4,0) circle (1);
\fill[magenta] (-4,0) circle (1);
```

```
\fill[magenta] (0,4) circle (1);
\fill[magenta] (0,-4) circle (1);
\fill[green] (2.828,2.828) circle (1);
\fill[green] (-2.828,2.828) circle (1);
\fill[green] (2.828,-2.828) circle (1);
\fill[green] (-2.828,-2.828) circle (1);
\fill[magenta] (3.353,3.353) circle (1);
\fill[magenta] (-3.353,3.353) circle (1);
\fill[magenta] (3.353,-3.353) circle (1);
\fill[magenta] (-3.353,-3.353) circle (1);
\end{tikzpicture}
\end{document}
```

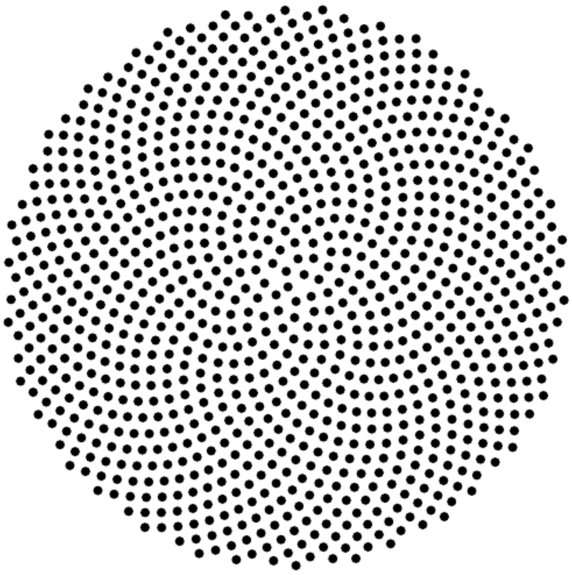


Şekil 2.25. Latex çizim örnekleri 6

Örnek 7:

```
\documentclass[tikz,border=2cm]{standalone}
\usepackage{tikz}
\usetikzlibrary{calc}
\begin{document}
\begin {tikzpicture}
```

```
\foreach \x in {0,...,1000}{
\fill ({mod((1+sqrt(5))*\x,2)*180}:
{0.1+sqrt(\x/pi)/2}) circle [radius=0.15];
}
\end{tikzpicture}
\end{document}
```

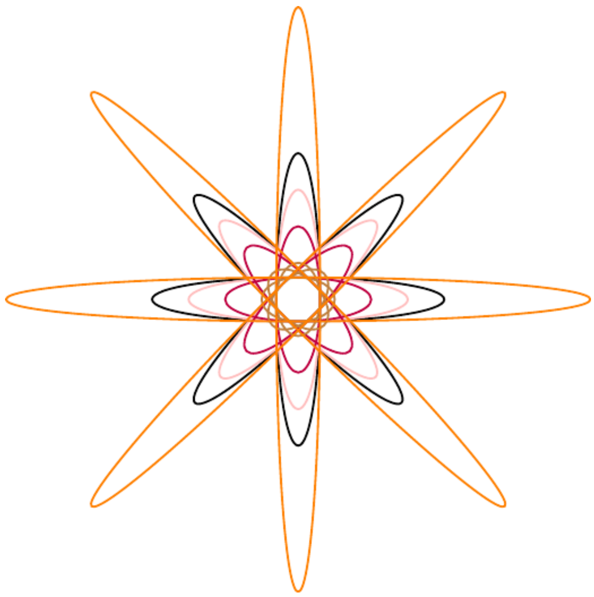


Şekil 2.26. Latex çizim örnekleri 7

Örnek 8:

```
\documentclass{standalone}
\usepackage{tikz}
\begin{document}
\begin{tikzpicture}[scale=1.25]
\foreach \angled in {0,45,...,135}
\draw[rotate=\angled,line width =1pt,
draw=black] (0,0) ellipse(2 and 0.3);
\foreach \anglee in {0,45,...,135}
\draw[rotate=\anglee,line width =1pt,
draw=pink] (0,0) ellipse(1.5 and 0.3);
\foreach \anglef in {0,45,...,135}
```

```
\draw[rotate=\anglef,line width =1pt,
draw=purple] (0,0) ellipse(1 and 0.3);
\foreach \angleg in {0,45,...,135}
\draw[rotate=\angleg,line width =1pt,
draw=brown] (0,0) ellipse(0.5 and 0.3);
\foreach \angleh in {0,45,...,135}
\draw[rotate=\angleh,line width =1pt,
draw=orange] (0,0) ellipse(4 and 0.3);
\node[scale=0.5, text=red] at (-90:0.02) {};
\end{tikzpicture}
\end{document}
```

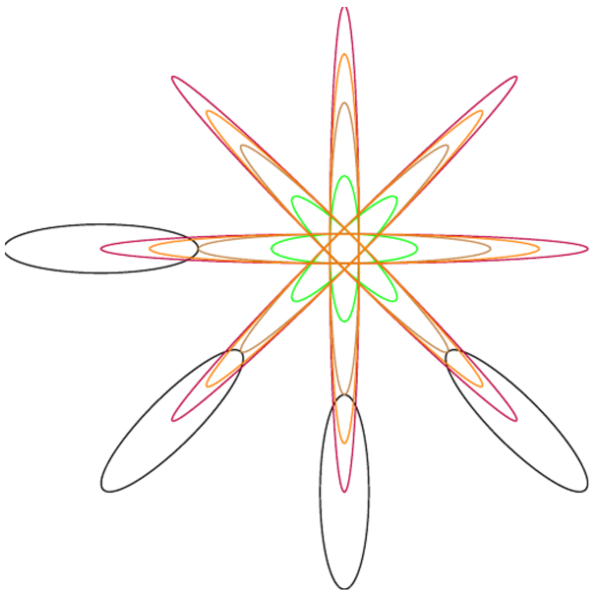


Şekil 2.27. Latex çizim örnekleri 8

Örnek 9:

```
\documentclass{standalone}
\usepackage{tikz}
\begin{document}
\begin{tikzpicture}[scale=1.25]
\foreach \angled in {0,45,...,135}
\draw[rotate=\angled,line width =1pt,
```

```
draw=black] (-5,0) ellipse(2 and 0.5);
\foreach \anglee in {0,45,...,135}
\draw[rotate=\anglee,line width =1pt,
draw=green] (0,0) ellipse(1.5 and 0.3);
\foreach \anglef in {0,45,...,135}
\draw[rotate=\anglef,line width =1pt,
draw=purple] (0,0) ellipse(5 and 0.3);
\foreach \angleg in {0,45,...,135}
\draw[rotate=\angleg,line width =1pt,
draw=brown] (0,0) ellipse(3 and 0.3);
\foreach \angleh in {0,45,...,135}
\draw[rotate=\angleh,line width =1pt,
draw=orange] (0,0) ellipse(4 and 0.3);
\node[scale=0.5, text=red] at (-90:0.02) {};
\end{tikzpicture}
\end{document}
```

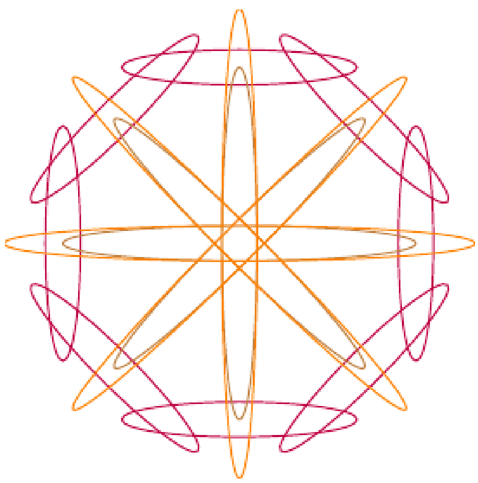


Şekil 2.28. Latex çizim örnekleri 9

Örnek 10:

```
\documentclass{standalone}
```

```
\usepackage{tikz}
\begin{document}
\begin{tikzpicture}[scale=1.25]
\foreach \angled in {0,45,...,135}
\draw[rotate=\angled,line width =1pt,
draw=purple] (0,-3) ellipse(2 and 0.3);
\foreach \anglef in {0,45,...,135}
\draw[rotate=\anglef,line width =1pt,
draw=purple] (0,3) ellipse(2 and 0.3);
\foreach \angleg in {0,45,...,135}
\draw[rotate=\angleg,line width =1pt,
draw=brown] (0,0) ellipse(3 and 0.3);
\foreach \angleh in {0,45,...,135}
\draw[rotate=\angleh,line width =1pt,
draw=orange] (0,0) ellipse(4 and 0.3);
\node[scale=0.5, text=red] at (-90:0.02) {};
\end{tikzpicture}
\end{document}
```



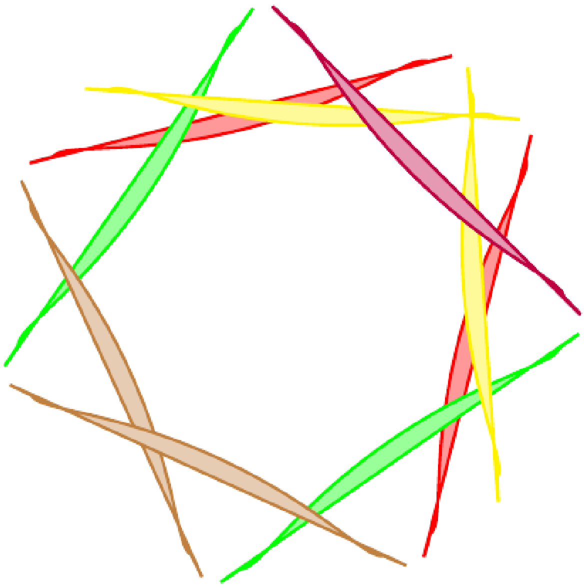
Şekil 2.29. Latex çizim örnekleri 10

Örnek 11:

```
\documentclass{beamer}
```

```
\beamertemplatenavigationsymbolsempty
\usepackage{tikz}
\tikzset{pics/spiro/.style={code={
\tikzset{spiro/.cd,#1}
\def\pv##1{\pgfkeysvalueof{/tikz/spiro/##1}}
\draw[trig format=rad,pic actions]
plot[variable=\t,domain=0:pi/2*\pv{nRotations},
samples=90*\pv{nRotations}+1, smooth cycle]
(
{(\pv{R}+\pv{r})*cos(\t)+\pv{p}
*cos((\pv{R}+\pv{r})*\t/\pv{r})},
{(\pv{R}+\pv{r})*sin(\t)+\pv{p}
*sin((\pv{R}+\pv{r})*\t/\pv{r})}
);
}},
spiro/.cd,R/.initial=6,r/.initial=-1.5,
p/.initial=1,nRotations/.initial=1}
\begin{document}
\begin{frame}[t]
\begin{center}
\begin{tikzpicture}
\draw[line width=.04cm,looseness=1]
foreach \i/\clr in
{3/red,1/green,2/brown,4/yellow,0/purple} {
pic[draw/.expanded=\clr!100,fill
/.expanded=\clr!40,scale=2/3,
rotate=-\i*100.25]{spiro}
pic[draw/.expanded=\clr!100,fill
/.expanded=\clr!40,scale=2/3,
rotate=\i*100.25]{spiro}
};
```

```
\end{tikzpicture}
\end{center}
\end{frame}
\end{document}
```

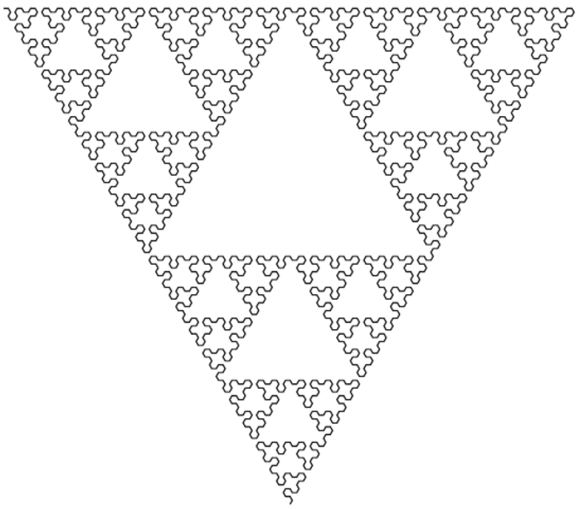


Şekil 2.30. Latex çizim örnekleri 11

Örnek 12:

```
\documentclass[border=8,tikz]{standalone}
\usepackage{tikz}
\usetikzlibrary{lindenmayersystems}
\begin{document}
\pgfdeclarelindenmayersystem{SierpArr}{
\symbol{X}{\pgflsystemdrawforward}
\symbol{Y}{\pgflsystemdrawforward}
\rule{X -> Y-X-Y}
\rule{Y -> X+Y+X}
}
\foreach \k in {7}
{
\tikz\draw[lindenmayer system={SierpArr,angle=60,
```

```
axiom=X,step=200pt/2^\k,order=\k}]
lindenmayer system;
}
\end{document}
```



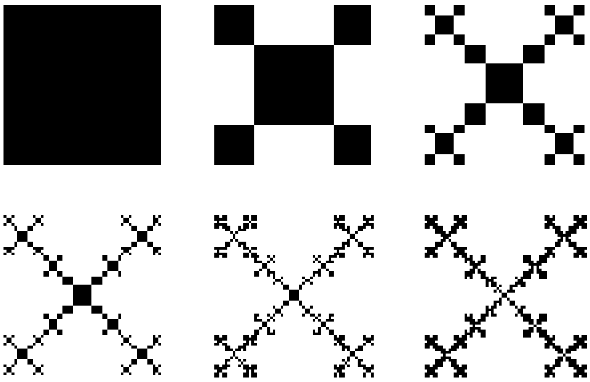
Şekil 2.31. Latex çizim örnekleri 12

Örnek 13:

```
\documentclass{article}
\usepackage{tikz}
\usetikzlibrary{calc}
\newcommand\DrawFracSquare[4]
{{% {Current number}{Side Length}{X}{Y}
\ifnum#1=0
\fill[black] ($(#3,#4)-(#2/2,#2/2)$)
rectangle +(#2,#2);
\else
\pgfmathsetmacro\NewNumber{int(#1-1)}
\pgfmathsetmacro\NewSideLength{#2/2}
\edef\NewRec{\noexpand\DrawFracSquare
{\NewNumber}{\NewSideLength}{#3}{#4}}
\NewRec
```

```
\pgfmathsetmacro\NewSideLength{#2/4}
\pgfmathsetmacro\NewX{#3+3*#2/8}
\pgfmathsetmacro\NewY{#4+3*#2/8}
\edef\NewRec{\noexpand\DrawFracSquare
{\NewNumber}{\NewSideLength}{\NewX}{\NewY}}
\NewRec
\pgfmathsetmacro\NewX{#3-3*#2/8}
\pgfmathsetmacro\NewY{#4+3*#2/8}
\edef\NewRec{\noexpand\DrawFracSquare
{\NewNumber}{\NewSideLength}{\NewX}{\NewY}}
\NewRec
\pgfmathsetmacro\NewX{#3-3*#2/8}
\pgfmathsetmacro\NewY{#4-3*#2/8}
\edef\NewRec{\noexpand\DrawFracSquare
{\NewNumber}{\NewSideLength}{\NewX}{\NewY}}
\NewRec
\pgfmathsetmacro\NewX{#3+3*#2/8}
\pgfmathsetmacro\NewY{#4-3*#2/8}
\edef\NewRec{\noexpand\DrawFracSquare
{\NewNumber}{\NewSideLength}{\NewX}{\NewY}}
\NewRec
\fi
}}
\begin{document}
\begin{tikzpicture}
\DrawFracSquare{0}{3}{0}{4}
\DrawFracSquare{1}{3}{4}{4}
\DrawFracSquare{2}{3}{8}{4}
\DrawFracSquare{3}{3}{0}{0}
\DrawFracSquare{4}{3}{4}{0}
\DrawFracSquare{5}{3}{8}{0}
```

```
\end{tikzpicture}
\end{document}
```

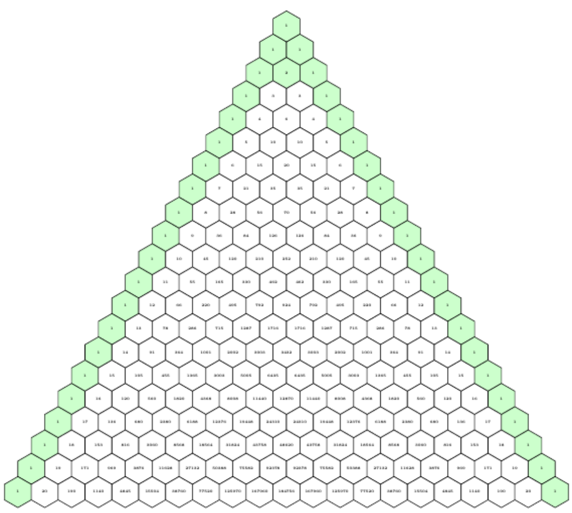


Şekil 2.32. Latex çizim örnekleri 13

Örnek 14:

```
\documentclass[tikz,border=0.25in]{standalone}
\usepackage{fp}
\newlength{\R}
\setlength{\R}{.75cm}
\newcommand\mycolor{white}
\newcommand{\modulo}[2]{%
\FPeval{\result}{trunc(#2-(#1*trunc(#2/#1,0)),0)}
\result% }
\newcommand{\makesierp}[2][2]{%
\begin{tikzpicture}[line width=.8pt]
\foreach \k in {0,...,#2}{
\begin{scope}[shift={(-60:{sqrt(3)*\R*\k})}]
\pgfmathtruncatemacro\ystart{#2-\k}
\foreach \n in {0,...,\ystart}{
\pgfmathtruncatemacro\newn{\n+\k}
\pgfmathsetmacro{\myvalue}{1}
\ifnum\k>0
\foreach \b in {1,...,\k}{
```

```
\FPeval\myvalue{trunc(\myvalue*
(\newn-\b+1)/\b:0)}
\global\let\myvalue=\myvalue}
\fi \modulo{\myvalue}{#1}%
\ifnum\result=0 \def\mycolor{green}\fi%
\begin{scope}[shift={(-120:{sqrt(3)*\R*\n})}]
\draw[fill=\mycolor!20]
(30:\R) \foreach \x in {90,150,...,330} {
-- (\x:\R)} --cycle (90:0)node[font=\tiny]
{$\mathbf{\myvalue}$};
\end{scope}}
\end{scope} }
\end{tikzpicture} }
\begin{document}
\makesierp{20}
\end{document}
```



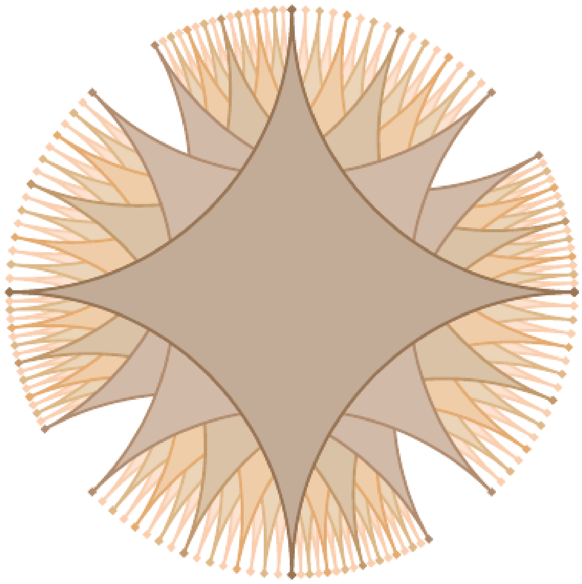
Şekil 2.33. Latex çizim örnekleri 14

Örnek 15:

```
\documentclass{beamer}
\beamertemplatenavigationsymbolsempty
```

```
\usepackage{tikz}
\begin{document}
\begin{frame}[t]
\definecolor{camel}{rgb}{0.76, 0.6, 0.42}
\definecolor{apricot}{rgb}{0.98, 0.81, 0.69}
\definecolor{burlywood}{rgb}{0.87, 0.72, 0.53}
\definecolor{fawn}{rgb}{0.9, 0.67, 0.44}
\definecolor{lighttaupe}{rgb}{0.7, 0.55, 0.43}
\definecolor{palebrown}{rgb}{0.6, 0.46, 0.33}
\begin{tikzpicture}
[
pics/spiro/.style={code={
\draw[line width=.04cm,
looseness=1,pic actions]
foreach \X in {0,90,180,270} {(\X:2)
node [circle, draw, scale=0.2]{}}
arc (180:90:2) arc (270:180:2) arc
(360:270:2) arc (90:0:2);
}
}]
\foreach \i/\clr in {1/apricot, 3/apricot,
5/apricot, 7/apricot,9/apricot,
11/apricot, 13/apricot,
15/apricot, 2/burlywood,  6/burlywood,
10/burlywood, 14/burlywood, 12/fawn,
4/fawn, 8/camel,
16/lighttaupe, 0/palebrown}
{
\pic[draw=\clr!100,fill=\clr!60,scale=2,rotate
=\i*1.8125]{spiro};
\pic[draw=\clr!100,fill=\clr!60,scale=2,rotate
```

```
=-\i*2.8125]{spiro};}
\end{tikzpicture}
\end{frame}
\end{document}
```

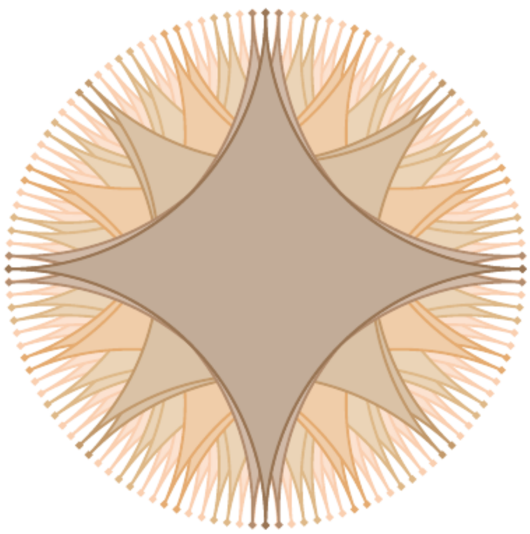


Şekil 2.34. Latex çizim örnekleri 15

Örnek 16:

```
\documentclass{beamer}
\beamertemplatenavigationsymbolsempty
\usepackage{tikz}
\begin{document}
\begin{frame}[t]
\definecolor{camel}{rgb}{0.76, 0.6, 0.42}
\definecolor{apricot}{rgb}{0.98, 0.81, 0.69}
\definecolor{burlywood}{rgb}{0.87, 0.72, 0.53}
\definecolor{fawn}{rgb}{0.9, 0.67, 0.44}
\definecolor{lighttaupe}{rgb}{0.7, 0.55, 0.43}
\definecolor{palebrown}{rgb}{0.6, 0.46, 0.33}
\begin{tikzpicture}
```

```
[pics/spiro/.style={code=
{\draw[line width=.04cm,looseness=1,
pic actions]
foreach \X in {0,90,180,270} {(\X:2) node
[circle, draw, scale=0.2]{}}
arc (180:90:2) arc (270:180:2)
arc (360:270:2) arc (90:0:2);}
}]
\foreach \i/\clr in {1/apricot, 3/apricot, 5/apricot,
7/apricot, 9/apricot, 11/apricot, 13/apricot,
15/apricot, 2/burlywood, 6/burlywood,
10/burlywood, 14/burlywood, 12/fawn,
4/fawn, 8/camel, 16/lighttaupe, 0/palebrown}
{\pic[draw=\clr!100,fill=\clr!60,scale=2,rotate
=\i*5.8125]{spiro};
\pic[draw=\clr!100,fill=\clr!60,scale=2,rotate
=-\i*5.8125]{spiro};}
\end{tikzpicture}
\end{frame}
\end{document}
```

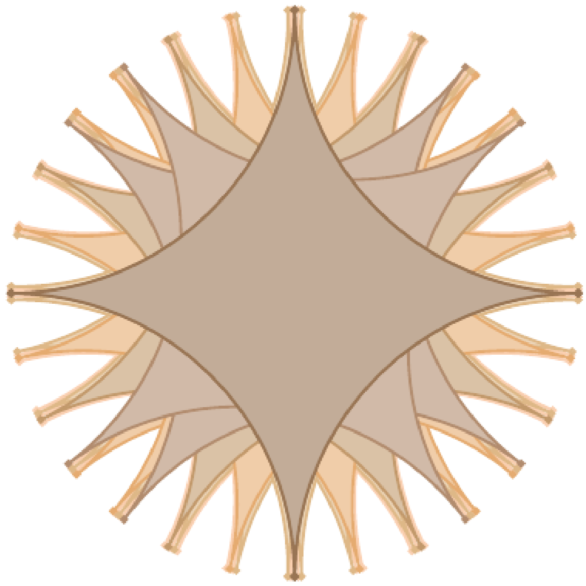


Şekil 2.35. Latex çizim örnekleri 16

Örnek 17:

```
\documentclass{beamer}
\beamertemplatenavigationsymbolsempty
\usepackage{tikz}
\begin{document}
\begin{frame}[t]
\definecolor{camel}{rgb}{0.76, 0.6, 0.42}
\definecolor{apricot}{rgb}{0.98, 0.81, 0.69}
\definecolor{burlywood}{rgb}{0.87, 0.72, 0.53}
\definecolor{fawn}{rgb}{0.9, 0.67, 0.44}
\definecolor{lighttaupe}{rgb}{0.7, 0.55, 0.43}
\definecolor{palebrown}{rgb}{0.6, 0.46, 0.33}
\begin{tikzpicture}
[pics/spiro/.style={code=
{
\draw[line width=.04cm,looseness=1,pic actions]
foreach \X in {0,90,180,270} {(\X:2)
node [circle, draw, scale=0.2]{}}
arc (180:90:2) arc (270:180:2)
arc (360:270:2) arc (90:0:2);
}
}]
\foreach \i/\clr in {1/apricot, 3/apricot, 5/apricot,
7/apricot, 9/apricot, 11/apricot, 13/apricot,
15/apricot, 2/burlywood, 6/burlywood,
10/burlywood, 14/burlywood, 12/fawn,
4/fawn, 8/camel, 16/lighttaupe, 0/palebrown}
{
\pic[draw=\clr!100,fill=\clr!60,scale=2,rotate
=\i*25.8125]{spiro}; \pic[draw=\clr!100,
fill=\clr!60,scale=2,rotate=-\i*25.8125]{spiro};
}
```

```
\end{tikzpicture}
\end{frame}
\end{document}
```

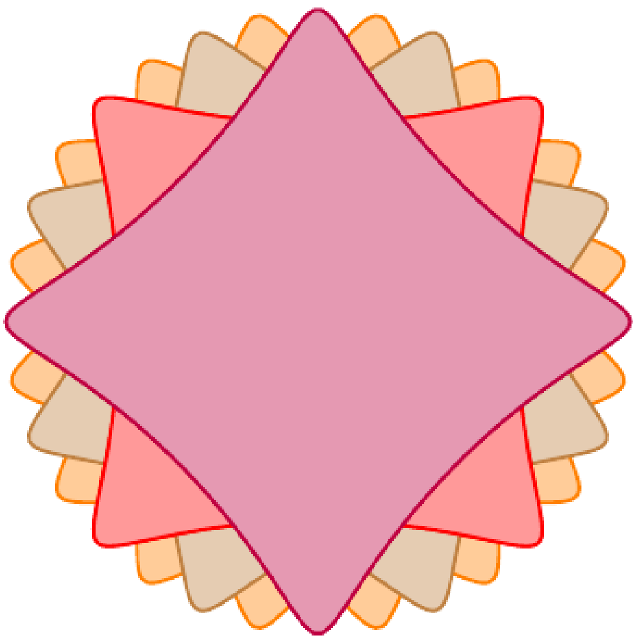


Şekil 2.36. Latex çizim örnekleri 17

Örnek 18:

```
\documentclass{beamer}
\beamertemplatenavigationsymbolsempty
\usepackage{tikz}
\tikzset{pics/spiro/.style={code=
{\tikzset{spiro/.cd,#1}
\def\pv##1{\pgfkeysvalueof{/tikz/spiro/##1}}
\draw[trig format=rad,pic actions]
plot[variable=\t,domain=0:2*pi*\pv{nRotations},
samples=90*\pv{nRotations}+1, smooth cycle]
( {
(\pv{R}+\pv{r})*cos(\t)+\pv{p}*
cos((\pv{R}+\pv{r})*\t/\pv{r})},
{(\pv{R}+\pv{r})*sin(\t)+\pv{p}*
sin((\pv{R}+\pv{r})*\t/\pv{r})});
```

```
}},
spiro/.cd,R/.initial=6,r/.initial=-1.5,
p/.initial=1,nRotations/.initial=1}
\begin{document}
\begin{frame}[t]
\begin{center}
\begin{tikzpicture}[line width=.04cm,
looseness=1]
\foreach \i/\clr in {3/orange,1/orange,
2/brown,4/red,0/purple}
{\pic[draw/.expanded=\clr!100,fill
/.expanded=\clr!40,scale=2/3,rotate=\i*11.25]{spiro};
\pic[draw/.expanded=\clr!100,fill
/.expanded=\clr!40,scale=2/3,rotate=-\i*11.25]{spiro};}
\end{tikzpicture}
\end{center}
\end{frame}
\end{document}
```

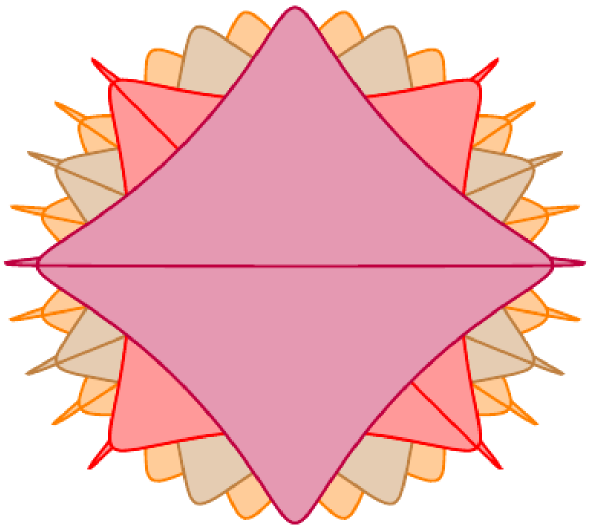


Şekil 2.37. Latex çizim örnekleri 18

Örnek 19:

```
\documentclass{beamer}
\beamertemplatenavigationsymbolsempty
\usepackage{tikz}
\tikzset{pics/spiro/.style={code={
\tikzset{spiro/.cd,#1}
\def\pv##1{\pgfkeysvalueof{/tikz/spiro/##1}}
\draw[trig format=rad,pic actions]
plot[variable=\t,
domain=0:3*pi*\pv{nRotations},
samples=90*\pv{nRotations}+1, smooth cycle]
(
{(\pv{R}+\pv{r})*cos(\t)+\pv{p}
*cos((\pv{R}+\pv{r})*\t/\pv{r})},
{(\pv{R}+\pv{r})*sin(\t)+\pv{p}
*sin((\pv{R}+\pv{r})*\t/\pv{r})});
}},
spiro/.cd,R/.initial=6,r/.initial=-1.5,p
/.initial=1,nRotations/.initial=1}
\begin{document}
\begin{frame}[t]
\begin{center}
\begin{tikzpicture}
[line width=.04cm,looseness=1]
\foreach \i/\clr in {3/orange,1/orange,
2/brown,4/red,0/purple}
{
\pic[draw/.expanded=\clr!100,fill
/.expanded=\clr!40,scale=2/3,rotate=\i*11.25]{spiro};
\pic[draw/.expanded=\clr!100,fill
/.expanded=\clr!40,scale=2/3,rotate=-\i*11.25]
{spiro};}
```

```
\end{tikzpicture}
\end{center}
\end{frame}
\end{document}
```

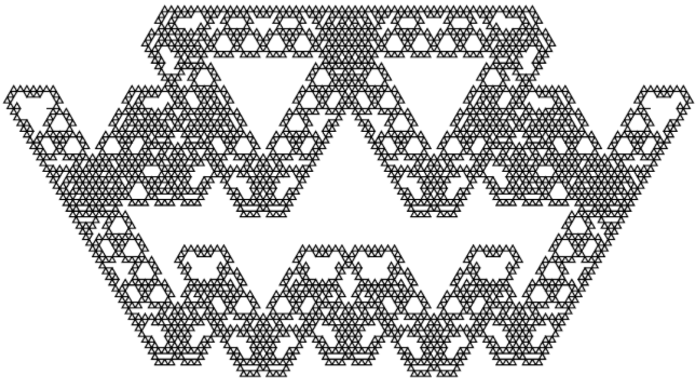


Şekil 2.38. Latex çizim örnekleri 19

Örnek 20:

```
\documentclass[border=5,tikz]{standalone}
\usepackage{tikz}
\usetikzlibrary{lindenmayersystems}
\begin{document}
\pgfdeclarelindenmayersystem{sekil}
{ \symbol{X}{\pgflsystemdrawforward}
\symbol{Y}{\pgflsystemdrawforward}
\rule{X -> Y-X+Y+X-Y}
\rule{Y -> X+Y+X} }
\foreach \k in {7}
{\tikz\draw[lindenmayer system=
{sekil,angle=120,axiom=X,
step=300pt/2^\k,order=\k}]
lindenmayer system; }
```

```
\end{document}
```

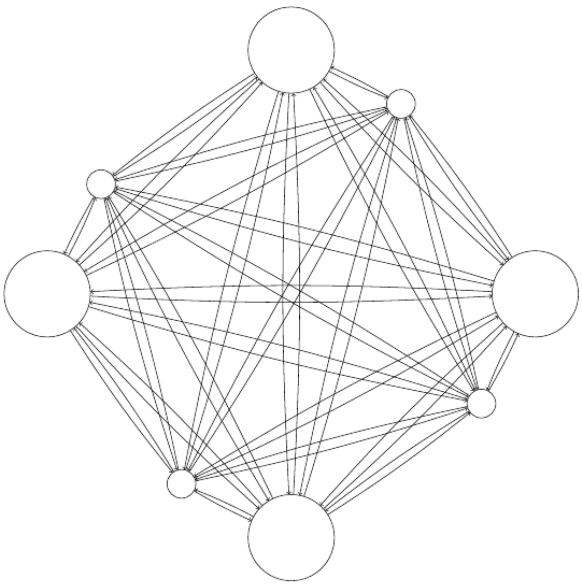


Şekil 2.39. Latex çizim örnekleri 20

Örnek 21:

```
\documentclass{article}
\usepackage{tikz}
\usepackage{verbatim}
\usepackage{verbatim}
\usepackage[active,tightpage]{preview}
\PreviewEnvironment{tikzpicture}
\setlength\PreviewBorder{5pt}%
\begin{comment}
\end{comment}
\usetikzlibrary[topaths]
\newcount\mycount
\begin{document}
\begin{tikzpicture}[transform shape]
\foreach \number in {1,...,4}{
\mycount=\number
\advance\mycount by -1
\multiply\mycount by 90
\advance\mycount by 0
\node[draw,circle,inner sep=0.75cm]
(N-\number) at (\the\mycount:6cm) {};}
```

```
\foreach \number in {5,...,8}{
\mycount=\number \advance\mycount by -1
\multiply\mycount by 90 \advance\mycount by 60
\node[draw,circle,inner sep=0.25cm]
(N-\number) at (\the\mycount:5.4cm) {};}
\foreach \number in {1,...,7}{
\mycount=\number \advance\mycount by 1
\foreach \numbera in {\the\mycount,...,8}{
\path (N-\number) edge[->,bend right=3]
(N-\numbera) edge[<-,bend left=3] (N-\numbera);}}
\end{tikzpicture}
\end{document}
```



Şekil 2.40. Latex çizim örnekleri 21

Örnek 22:

```
\documentclass{article}
\usepackage{tikz}
\usetikzlibrary{shapes.geometric}
\usepackage{caption}
\tikzset{myStyle/.style={
```

```
draw,
regular polygon,
regular polygon sides=3,
minimum size=20em}, }
\begin{document}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=purple]{};
\end{tikzpicture}
\caption{1}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=purple] (A) {};
\foreach \mycorner in {1,2,3}
{\node[myStyle, fill=blue,scale=.33] (A\mycorner) at
(A.corner \mycorner) {};}
\end{tikzpicture}
\caption{2}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=purple] (A) {};
\foreach \mycorner in {1,2,3}
{\node[myStyle, fill=blue,scale=.33] (A\mycorner) at
(A.corner \mycorner) {};}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
```

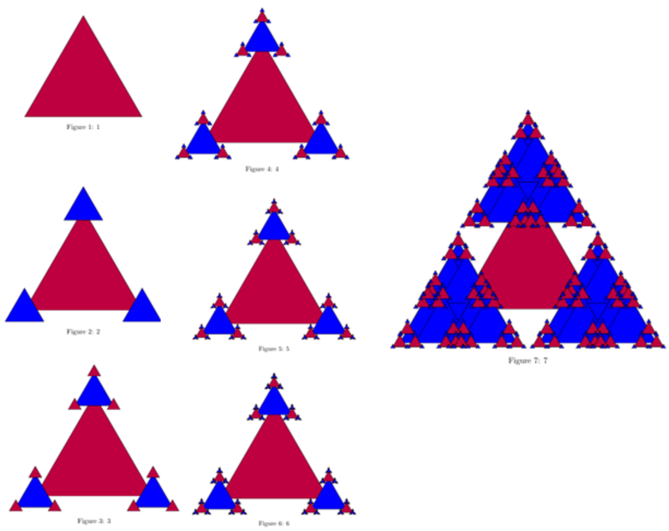
```
{\node[myStyle, fill=purple,scale=.11] at
(A\mycornerone.corner \mycornertwo) {};}}
\end{tikzpicture}
\caption{3}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=purple] (A) {};
\foreach \mycorner in {1,2,3}
{\node[myStyle, fill=blue,scale=.33] (A\mycorner)
at (A.corner \mycorner) {};}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\node[myStyle, fill=purple,scale=.11]
(A\mycornerone\mycornertwo) at
(A\mycornerone.corner \mycornertwo) {};}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\node[myStyle, fill=blue,scale=.0367] at
(A\mycornerone\mycornertwo.corner
\mycornerthree) {};}}}
\end{tikzpicture}
\caption{4}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=purple] (A) {};
\foreach \mycorner in {1,2,3}
```

```
{\node[myStyle, fill=blue,scale=.33]
(A\mycorner) at (A.corner \mycorner) {};}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\node[myStyle, fill=purple,scale=.11]
(A\mycornerone\mycornertwo) at
(A\mycornerone.corner \mycornertwo) {};}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\node[myStyle, fill=blue,scale=.0367]
(A\mycornerone\mycornertwo\mycornerthree) at
(A\mycornerone\mycornertwo.corner
\mycornerthree) {};}}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\foreach \mycornerfour in {1,2,3}
{\node[myStyle, fill=purple,scale=.0150] at
(A\mycornerone\mycornertwo\mycornerthree.corner
\mycornerfour) {};}}}}
\end{tikzpicture}
\caption{5}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=purple] (A) {};
\foreach \mycorner in {1,2,3}
{\node[myStyle, fill=blue,scale=.33] (A\mycorner) at
(A.corner \mycorner) {};}
```

```
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\node[myStyle, fill=purple,scale=.11]
(A\mycornerone\mycornertwo) at
(A\mycornerone.corner \mycornertwo) {};}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\node[myStyle, fill=blue,scale=.0367]
(A\mycornerone\mycornertwo\mycornerthree) at
(A\mycornerone\mycornertwo.corner
\mycornerthree) {};}}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\foreach \mycornerfour in {1,2,3}
{\node[myStyle, fill=purple,scale=.0150]
(A\mycornerone\mycornertwo\mycornerthree\mycornerfour)
at (A\mycornerone\mycornertwo\mycornerthree.corner
\mycornerfour) {};}}}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\foreach \mycornerfour in {1,2,3}
{\foreach \mycornerfive in {1,2,3}
{\node[myStyle, fill=blue,scale=.008] at
(A\mycornerone\mycornertwo\mycornerthree
\mycornerfour.corner \mycornerfive) {};}}}}}
\end{tikzpicture}
\caption{6}
\end{figure}
```

```
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=purple] (A) {};
\foreach \mycorner in {1,2,3}
{\node[myStyle, fill=blue,scale=.33]
(A\mycorner) at (A.corner \mycorner) {};}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\node[myStyle, fill=purple,scale=.11]
(A\mycornerone\mycornertwo) at
(A\mycornerone.corner \mycornertwo) {};}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\node[myStyle, fill=blue,scale=.4]
(A\mycornerone\mycornertwo\mycornerthree) at
(A\mycornerone\mycornertwo.corner
\mycornerthree) {};}}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\foreach \mycornerfour in {1,2,3}
{\node[myStyle, fill=purple,scale=.1]
(A\mycornerone\mycornertwo\mycornerthree\mycornerfour)
at (A\mycornerone\mycornertwo\mycornerthree.corner
\mycornerfour) {};}}}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\foreach \mycornerfour in {1,2,3}
```

```
{\foreach \mycornerfive in {1,2,3}
{\node[myStyle, fill=blue,scale=.03]
(A\mycornerone\mycornertwo\mycornerthree
\mycornerfour\mycornerfive) at
(A\mycornerone\mycornertwo\mycornerthree
\mycornerfour.corner \mycornerfive) {};}}}}}
\foreach \mycornerone in {1,2,3}
{\foreach \mycornertwo in {1,2,3}
{\foreach \mycornerthree in {1,2,3}
{\foreach \mycornerfour in {1,2,3}
{\foreach \mycornerfive in {1,2,3}
{\foreach \mycornersix in {1,2,3}
{\node[myStyle, fill=purple,scale=.001] at
(A\mycornerone\mycornertwo\mycornerthree
\mycornerfour\mycornerfive.corner \mycornersix)
{};}}}}}}
\end{tikzpicture}
\caption{7}
\end{figure}
\end{document}
```



Şekil 2.41. Latex çizim örnekleri 22

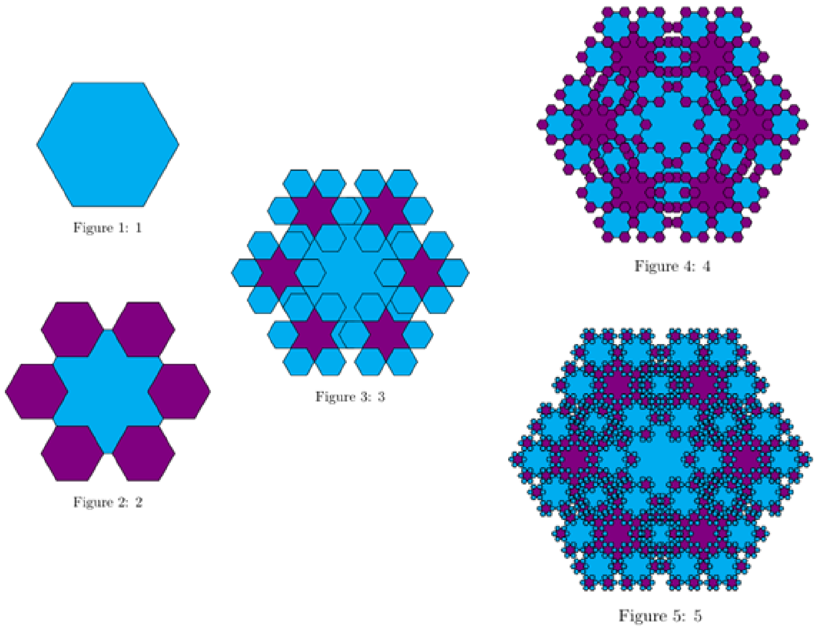
Örnek 23:

```
\documentclass{article}
\usepackage{tikz}
\usetikzlibrary{shapes.geometric}
\usepackage{caption}
\tikzset{
myStyle/.style={
draw,
regular polygon,
regular polygon sides=6,
minimum size=20em
},
}
\begin{document}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=cyan, scale=.50]{};
\end{tikzpicture}
\caption{1}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=cyan, scale=.50] (A) {};
\foreach \mycorner in {1,2,3,4,5,6}
{\node[myStyle, fill=violet,scale=.22]
(A\mycorner) at (A.corner \mycorner) {};}
\end{tikzpicture}
\caption{2}
\end{figure}
\begin{figure}
```

```
\centering
\begin{tikzpicture}
\node[myStyle, fill=cyan, scale=.50] (A) {};
\foreach \mycorner in {1,2,3,4,5,6}
{\node[myStyle, fill=violet,scale=.22]
(A\mycorner) at (A.corner \mycorner) {};}
\foreach \mycornerone in {1,2,3,4,5,6}
{\foreach \mycornertwo in {1,2,3,4,5,6}
{\node[myStyle, fill=cyan,scale=.11]
at (A\mycornerone.corner \mycornertwo)
{};}}
\end{tikzpicture}
\caption{3}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=cyan, scale=.50] (A) {};
\foreach \mycorner in {1,2,3,4,5,6}
{\node[myStyle, fill=violet,scale=.22]
(A\mycorner) at (A.corner \mycorner) {};}
\foreach \mycornerone in {1,2,3,4,5,6}
{\foreach \mycornertwo in {1,2,3,4,5,6}
{\node[myStyle, fill=cyan,scale=.11]
(A\mycornerone\mycornertwo) at
(A\mycornerone.corner \mycornertwo) {};}}
\foreach \mycornerone in {1,2,3,4,5,6}
{\foreach \mycornertwo in {1,2,3,4,5,6}
{\foreach \mycornerthree in {1,2,3,4,5,6}
{\node[myStyle, fill=violet,scale=.0367] at
(A\mycornerone\mycornertwo.corner \mycornerthree)
```

```
{};}}}
\end{tikzpicture}
\caption{4}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\node[myStyle, fill=cyan, scale=.50] (A) {};
\foreach \mycorner in {1,2,3,4,5,6}
{\node[myStyle, fill=violet,scale=.22]
(A\mycorner) at (A.corner \mycorner) {};}
\foreach \mycornerone in {1,2,3,4,5,6}
{\foreach \mycornertwo in {1,2,3,4,5,6}
{\node[myStyle, fill=cyan,scale=.11]
(A\mycornerone\mycornertwo) at
(A\mycornerone.corner \mycornertwo) {};}}
\foreach \mycornerone in {1,2,3,4,5,6}
{\foreach \mycornertwo in {1,2,3,4,5,6}
{\foreach \mycornerthree in {1,2,3,4,5,6}
{\node[myStyle, fill=violet,scale=.0367]
(A\mycornerone\mycornertwo\mycornerthree) at
(A\mycornerone\mycornertwo.corner \mycornerthree)
{};}}}
\foreach \mycornerone in {1,2,3,4,5,6}
{\foreach \mycornertwo in {1,2,3,4,5,6}
{\foreach \mycornerthree in {1,2,3,4,5,6}
{\foreach \mycornerfour in {1,2,3,4,5,6}
{\node[myStyle, fill=cyan,scale=.0150] at
(A\mycornerone\mycornertwo\mycornerthree.corner
\mycornerfour)
{};}}}}
```

```
\end{tikzpicture}
\caption{5}
\end{figure}
\end{document}
```



Şekil 2.42. Latex çizim örnekleri 23

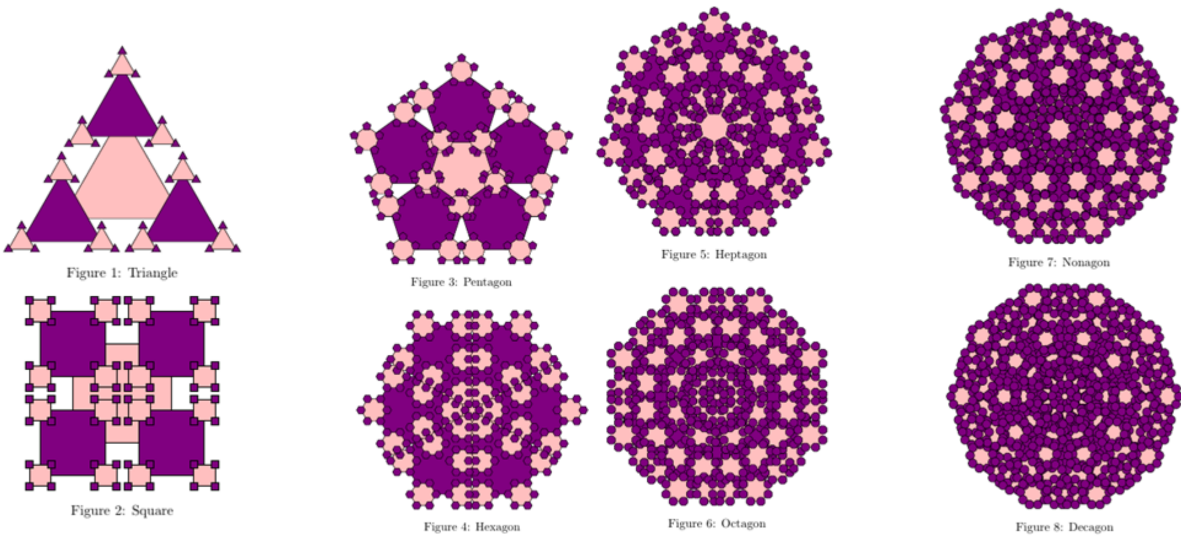
Örnek 24:

```
\documentclass{article}
\usepackage{tikz}
\usetikzlibrary{shapes.geometric}
\usepackage{caption}
\tikzset{
myStyle/.style={
draw,
regular polygon,
regular polygon sides=#1,
minimum size=20em
}
}
```

```
\newcommand{\myPoly}[1]{\node[myStyle=#1,
fill=pink, scale=.50] (A) {};
\foreach \mycorner in {1,2,...,#1}
{\node[myStyle=#1, fill=violet,scale=.33]
(A\mycorner) at (A.corner \mycorner) {};}
\foreach \mycornerone in {1,2,...,#1}
{\foreach \mycornertwo in {1,2,...,#1}
{\node[myStyle=#1, fill=pink,scale=.11]
(A\mycornerone\mycornertwo) at
(A\mycornerone.corner \mycornertwo) {};}}
\foreach \mycornerone in {1,2,...,#1}
{\foreach \mycornertwo in {1,2,...,#1}
{\foreach \mycornerthree in {1,2,...,#1}
{\node[myStyle=#1, fill=violet,scale=.0367] at
(A\mycornerone\mycornertwo.corner
\mycornerthree) {};}}}
}
\begin{document}
\begin{figure}
\centering
\begin{tikzpicture}
\myPoly{3}
\end{tikzpicture}
\caption{Triangle}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\myPoly{4}
\end{tikzpicture}
\caption{Square}
```

```
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\myPoly{5}
\end{tikzpicture}
\caption{Pentagon}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\myPoly{6}
\end{tikzpicture}
\caption{Hexagon}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\myPoly{7}
\end{tikzpicture}
\caption{Heptagon}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\myPoly{8}
\end{tikzpicture}
\caption{Octagon}
\end{figure}
\begin{figure}
\centering
```

```
\begin{tikzpicture}
\myPoly{9}
\end{tikzpicture}
\caption{Nonagon}
\end{figure}
\begin{figure}
\centering
\begin{tikzpicture}
\myPoly{10}
\end{tikzpicture}
\caption{Decagon}
\end{figure}
\end{document}
```



Şekil 2.43. Latex çizim örnekleri 24

2.5.2. Mathematica programında çizimler

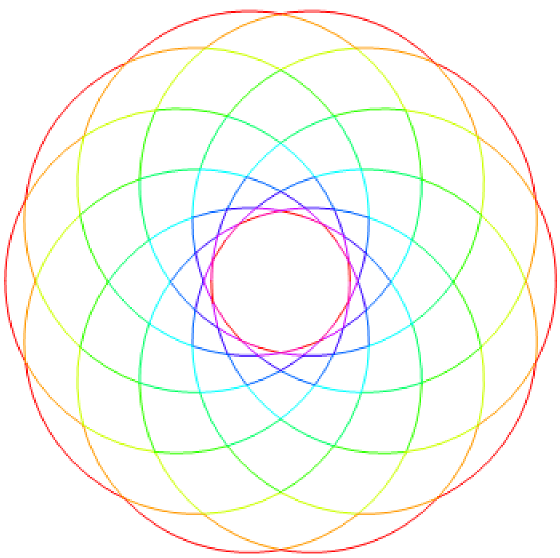
Mathematica programı kullanılırken ilk olarak bir fonksiyon tanımlanır. Fonksiyon tanımlanırken bir fonksiyon ismi belirlenir ve değişkenler "[x,y]" verilip ":=" eşitliğiyle fonksiyon yazılır. Tanımlanan değişkenlerin tanımlandığını renk değiştirip yeşil olmasından anlaşılır, mavi olarak kaldığında ise tanımlanmamış bir değişken olduğu gösterilir.

"Plot" komutuyla fonksiyonların grafikleri çizdirilir. "{x, aralık}" yazılarak grafikte değişkenin hangi aralıkta olacağı belirlenir. "Plot3D" komutu kullanılırsa üç boyutlu grafik çizdirilir. Grafiği renklendirmek için "ColorFunction" komutu kullanılır. "ContourPlot" komutu ise 3 boyutlu grafikleri 2 boyutlu olarak gösterir. Genel olarak kullanılan bu komutlar dışında çizilmek istenen her şekil için ayrı bir komut dizisi kullanılmaktadır.

Mathematica programıyla ilgili örnekler aşağıda eklenmiştir [18, 19].

Örnek 1:

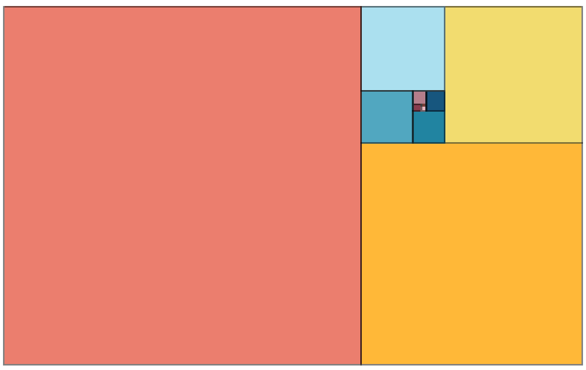
```
Graphics[Table[{Hue[t/40],
Circle[{3 Sin[2 Pi (t - 1)/40],
3 Cos[2 Pi (t - 1)/40]}, 5]},
{t, 40}]];
n = 10; R = 5; r = 3;
Graphics[Table[{Hue[Max[k, -1 - k]/n],
Circle[{r Cos[2 Pi t/n], r Sin[2 Pi t/n]},
R, {Pi (2 t + k)/n + ArcSin[r/R Sin[k Pi/n]],
Pi (2 t + k + 1)/n +
ArcSin[r/R Sin[(k + 1) Pi/n]]}]}],
{t,n}, {k, 1 - n, n}]]
```



Şekil 2.44. Mathematica çizim örnekleri 1

Örnek 2:

```
gr[0] := {{0, 0}, {1, -1}};
gr[n_] := Module[
{\[CurlyPhi]=GoldenRatio,
m = Mod[n, 4], a, b, c, d},
{{a, b}, {c, d}} = gr[n - 1];
Switch[Mod[n, 4],
0, {{a, d}, {a+\[CurlyPhi]^-n,
d-\[CurlyPhi]^-n}},
1, {{c, d+\[CurlyPhi]^-n},
{c+\[CurlyPhi]^-n, d}},
2, {{c-\[CurlyPhi]^-n,
b+\[CurlyPhi]^-n}, {c, b}},
3, {{a-\[CurlyPhi]^-n,b},
{a, b-\[CurlyPhi]^-n}}]];
Graphics[{EdgeForm[Opacity[.5]],
Table[{ColorData[24, k + 1],
Rectangle @@ gr[k]}, {k, 0, 10}]}]
```

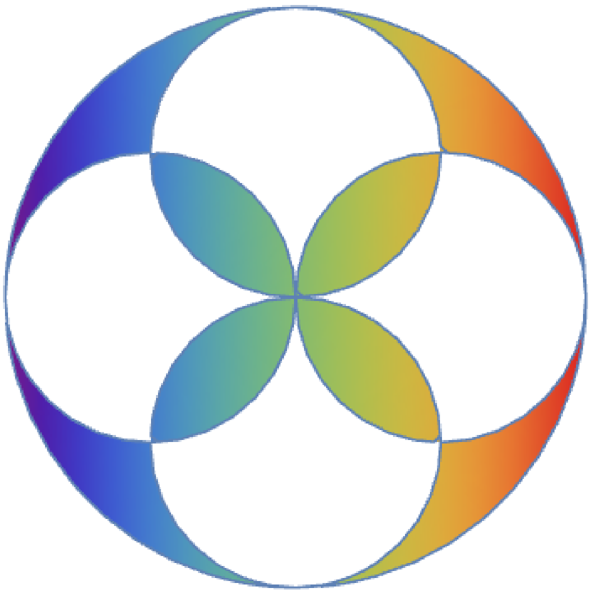


Şekil 2.45. Mathematica çizim örnekleri 2

Örnek 3:

```
RegionPlot[region = RegionUnion
[Sequence @@ RegionIntersection @@@
Subsets[{Disk[{-1, 0}], Disk[{0, -1}],
```

```
Disk[{1, 0}],
Disk[{0, 1}]}, {2}],
Fold[RegionDifference,
{Disk[{0, 0}, 2],
Disk[{-1, 0}],
Disk[{0, -1}], Disk[{1, 0}],
Disk[{0, 1}]}]],
ColorFunction -> "Rainbow",
Frame -> False]
```

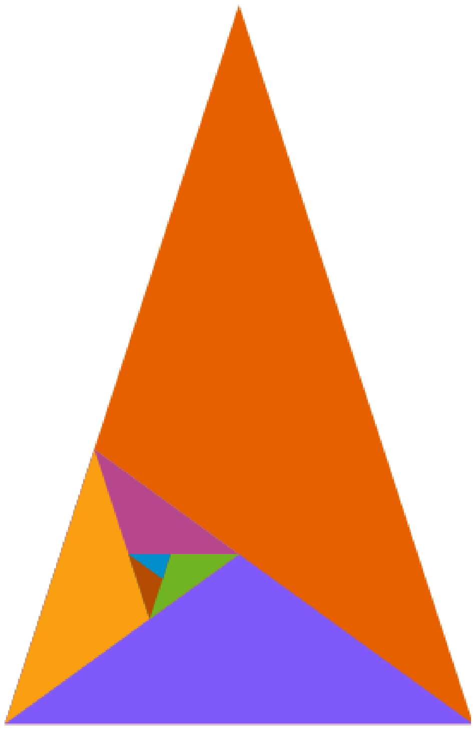


Şekil 2.46. Mathematica çizim örnekleri 3

Örnek 4:

```
With[{n = 6}, Graphics[MapIndexed
[{ColorData[103] @@ #2,#1} &,
NestList[MapAt[Composition[
TranslationTransform[AngleVector
[2 \[Pi]/5]/GoldenRatio],
RotationTransform[-3 \[Pi]/5],
ScalingTransform[GoldenRatio
- {1, 1}]],#, 1] &,
```

```
Polygon[{{0, 0}, {1, 0},
{1/2, Sqrt[5 + 2 Sqrt[5]]/2}}],
 n]]]]
```

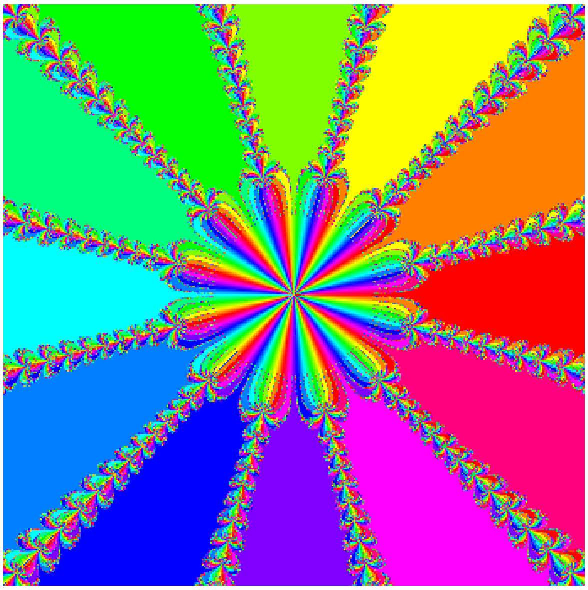


Şekil 2.47. Mathematica çizim örnekleri 4

Örnek 5:

```
newt[n_, z_] := Arg[FixedPoint[# - (#^n - 1)/
(n #^(n - 1)) &, N[z], 50]]/(2 Pi);
newtC = Compile[{{n, _Integer},
{z, _Complex}},
Arg[FixedPoint[# - (#^n - 1)/(n #^(n - 1)) &,
N[z], 50]]/(2 Pi)];
newtonplot[n_, npoint_, xmin_, xmax_, ymin_, ymax_]:=
DensityPlot[newtC[n, x + I*y],
{x, xmin, xmax},
{y, ymin, ymax},
Mesh -> False, PlotPoints -> npoint,
Frame -> False,
```

```
ColorFunctionScaling -> False,
ColorFunction -> (Hue[#] &)]
{newt[3, 2], newt[3, +2 + 2*I], newt[3, -2 - 2*I]}
{0, 0., -0.333333}, newtonplot[12, 300, -2, 2, -2, 2]
```

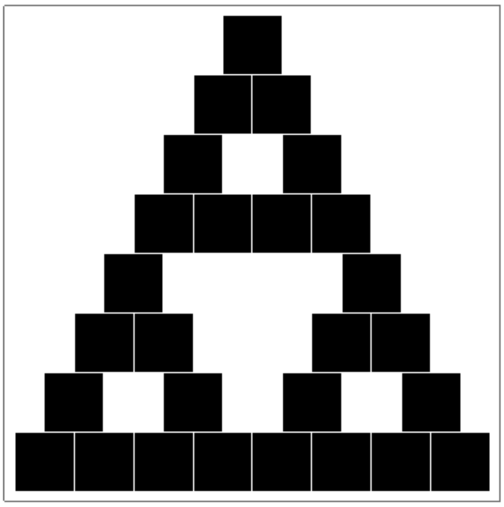


Şekil 2.48. Mathematica çizim örnekleri 5

Örnek 6:

```
pascalMod2b[n_, rr_: 0, cf_: Automatic,
tc_: Automatic,
fs_: Automatic][opts : OptionsPattern[]] :=
Graphics[{EdgeForm[White],
Table[MapIndexed[
Module[{b = Binomial[i, #2[[1]] - 1]},
{Mod[b, 2] /. {1 -> (cf /. Automatic ->
(Black &))@b, 0 -> White},
Rectangle[{#, -i}, RoundingRadius -> rr],
Text[Style[b, fs /. Automatic -> 12,
(tc /. Automatic -> (Opacity[0] &))@
b], {#, -i} + .5]}] &,
Range[-i/2, i/2]], {i, 0, 2^n - 1}]}, opts];
```

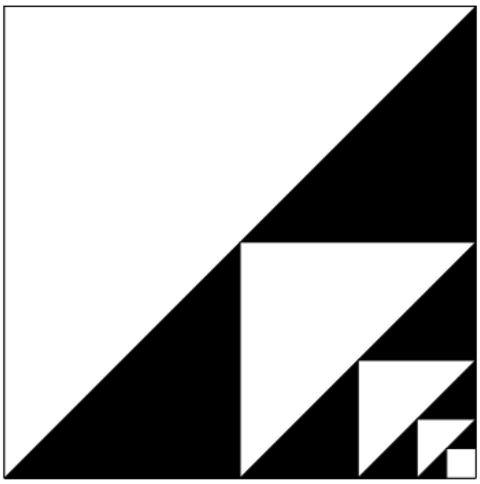
```
pascalMod2b[3][Frame -> True,
FrameTicks -> None]
```



Şekil 2.49. Mathematica çizim örnekleri 6

Örnek 7:

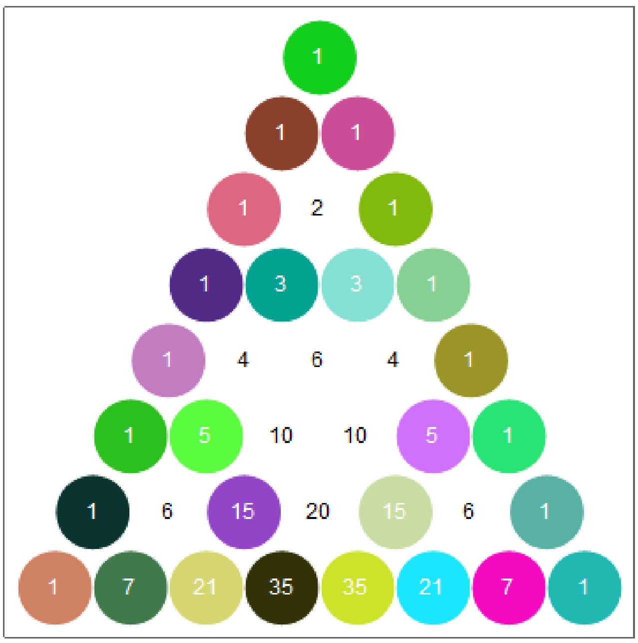
```
coords = {{0, 0}, {0, 1}, {1, 1}, {1, 0}};
tf = Composition[TranslationTransform[{1/2, 0}],
ScalingTransform[{1/2, 1/2}]]
n = 4; rects = NestList[tf /@ # &, coords, n];
Graphics[{EdgeForm[Black], Rectangle[],
White, Polygon[Most /@ rects], Polygon@Last@rects}]
```



Şekil 2.50. Mathematica çizim örnekleri 7

Örnek 8:

```
pascalMod2b[n_, rr_: 0, cf_:
Automatic, tc_: Automatic,fs_: Automatic]
[opts : OptionsPattern[]]:=
Graphics[{EdgeForm[White],
Table[MapIndexed[
Module[{b = Binomial[i, #2[[1]] - 1]},
{Mod[b, 2]/. {1 -> (cf /. Automatic ->
(Black &))@b, 0 -> White},
Rectangle[{#, -i}, RoundingRadius -> rr],
Text[Style[b, fs /. Automatic -> 12,
(tc /. Automatic -> (Opacity[0] &))@
b], {#, -i} + .5]}] &,
Range[-i/2, i/2]], {i, 0,
2^n - 1}]}, opts];
pascalMod2b[3, .5, RandomColor[] &,
Mod[#, 2] /. {0 -> Black, 1 -> White} &]
[Frame -> True,
FrameTicks -> None]
```

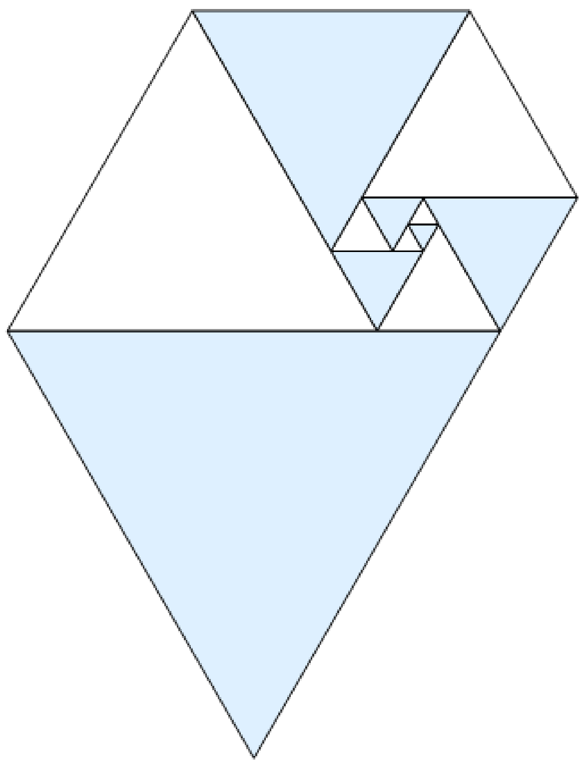


Şekil 2.51. Mathematica çizim örnekleri 8

Örnek 9:

```
nextTriangle[oppositept_, firstedge_] :=
Module[{f = firstedge, p},
p = {{(f[[1, 1]] + f[[2, 1]] + Sqrt[3.]
(f[[1, 2]] - f[[2, 2]]))/
2, (f[[1, 2]] + f[[2, 2]] - Sqrt[3.]
(f[[1, 1]] - f[[2, 1]]))/
2}, {(f[[1, 1]] + f[[2, 1]] - Sqrt[3.]
(f[[1, 2]] - f[[2, 2]]))/
2, (f[[1, 2]] + f[[2, 2]] + Sqrt[3.]
(f[[1, 1]] - f[[2, 1]]))/
2}};
{firstedge[[1]], firstedge[[2]],
Chop[First[
Sort[p, EuclideanDistance[#1, oppositept] >
EuclideanDistance[#2, oppositept] &]]]}]
n = 12;
triangles = {{{0, Sqrt[3.]}, {-1, 0}, {1, 0}}};
Do[{t = Last[triangles];
nextedge = t[[{1, 3}]];
edgefit = Fit[nextedge, {1, x}, x];
allpts = Flatten[triangles, 1];
colinearpos =
Boole[Chop[edgefit /. x -> #[[1]]] == #[[2]]
& /@ allpts];
colinearpts = Cases[Transpose
[{allpts, colinearpos}],
{x_, 1} -> x]; line = {First[
Sort[colinearpts,
EuclideanDistance[#1, t[[3]]] >
EuclideanDistance[#2, t[[3]]] &]], t[[3]]};
nextt = nextTriangle[t[[2]], line];
```

```
AppendTo[triangles, nextt];}, {i, 1, n - 1}]
Graphics[Table[{If[EvenQ[n], LightBlue, White],
EdgeForm[Thin], Polygon[triangles[[n]]]},
{n, 1, Length[triangles]}]]
```

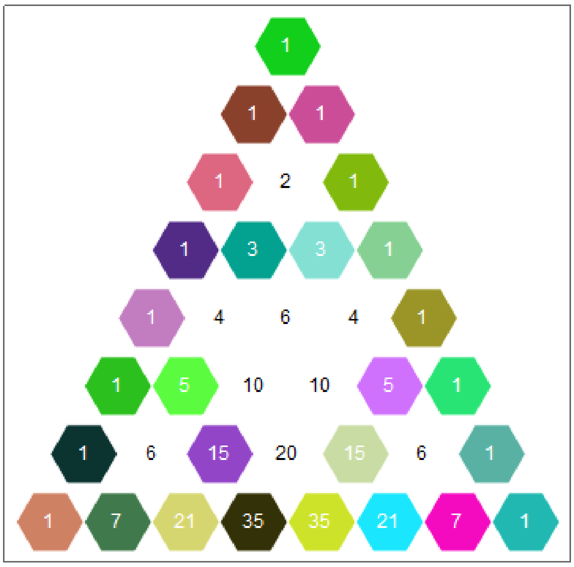


Şekil 2.52. Mathematica çizim örnekleri 9

Örnek 10:

```
pascalMod2b[n_, rr_: 0, cf_:
Automatic, tc_: Automatic,
fs_: Automatic][opts : OptionsPattern[]] :=
Graphics[{EdgeForm[White], Table[MapIndexed[
Module[{b = Binomial[i, #2[[1]] - 1]},
{Mod[b, 2]
/. {1 -> (cf /. Automatic ->
(Black &))@b, 0 -> White},
Rectangle[{#, -i}, RoundingRadius -> rr],
Text[Style[b, fs /. Automatic -> 12, (tc /.
```

```
Automatic -> (Opacity[0] &))@ b],
{#, -i} + .5]}]
&, Range[-i/2, i/2]], {i, 0,2^n - 1}]},
opts]; % /. Rectangle[a_, ___] :>
Polygon[CirclePoints[a + .5, .5, 6]]
```

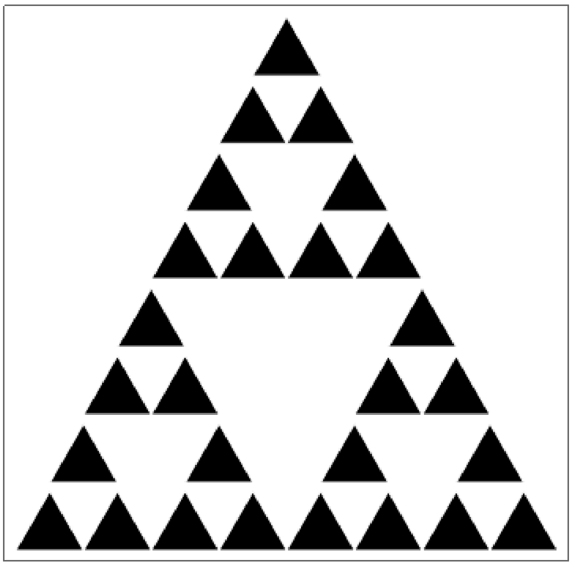


Şekil 2.53. Mathematica çizim örnekleri 10

Örnek 11:

```
pascalMod2b[n_, rr_: 0, cf_:
Automatic, tc_: Automatic,
fs_: Automatic][opts : OptionsPattern[]] :=
Graphics[{EdgeForm[White],
Table[MapIndexed[
Module[{b = Binomial[i, #2[[1]] - 1]},
{Mod[b, 2] /.{1->(cf/.
Automatic->(Black &))@b,0->White},
Rectangle[{#, -i}, RoundingRadius -> rr],
Text[Style[b, fs /. Automatic -> 12,
(tc /. Automatic -> (Opacity[0] &))@
b], {#, -i} + .5]}] &, Range[-i/2, i/2]],
```

```
{i, 0,2^n - 1}]}, opts]; pascalMod2b[3]
[Frame -> True, FrameTicks -> None]/.
Rectangle[a_, ___] :> Translate
[SSSTriangle[1, 1, 1], a + .5]
```

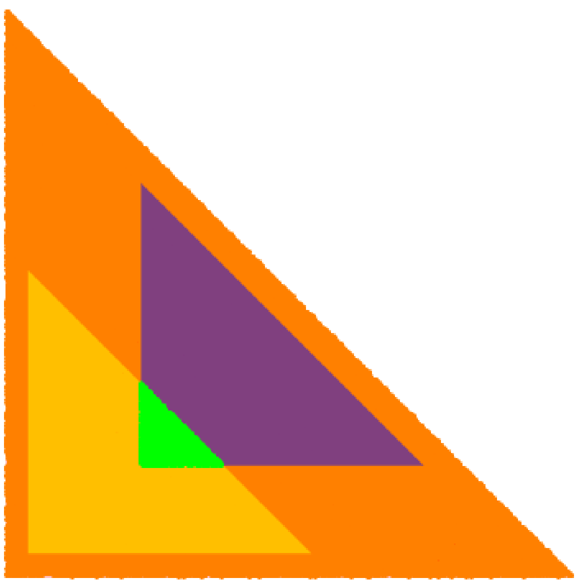


Şekil 2.54. Mathematica çizim örnekleri 11

Örnek 12:

```
n = 1*^5;(*number of sample points*)
big = {{0, 0}, {1, 0}, {0, 1}};
SeedRandom[42, Method -> "Legacy"];
(*for reproducibility*)While[
t1 = {#2, 1 - #1 - #2} & @@
RandomVariate[DirichletDistribution[{1, 1, 1}]];
! (-1/2 <= #1 < 1/2 && 0 <= #2 <= 1/2 - #1)
& @@ t1];
While[t2 = {#2, 1 - #1 - #2} & @@
RandomVariate[DirichletDistribution[{1, 1, 1}]];
! (-1/2 <= #1 < 1/2 && 0 <= #2 <= 1/2 - #1)
& @@ t2];
ts1 = {t1, t1 + {1/2, 0}, t1 + {0, 1/2}};
```

```
ts2 = {t2, t2 + {1/2, 0}, t2 + {0, 1/2}};
tsi = Graphics`PolygonUtils
`PolygonIntersection[Polygon[ts1],
Polygon[ts2]][[1, 1]];
rmf = RegionMember[Triangle[tsi]];
pts = {#2, 1 - #1 - #2} & @@@
RandomVariate[DirichletDistribution[{1, 1, 1}], n];
inside = Select[pts, rmf];
outside = Complement[pts, inside];
Graphics[{{Red, Triangle[big]}, {Orange,
Point[outside]}, {Opacity[1/2, Yellow],
Triangle[ts1]}, {Opacity[1/2, Blue],
Triangle[ts2]},
{Green,Point[inside]}}, PlotLabel ->
Row[{"p=", N[Length[inside]/n]}]]
```

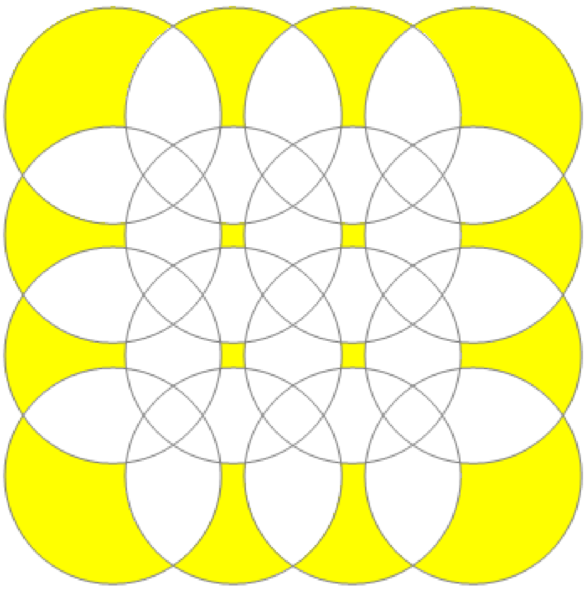


Şekil 2.55. Mathematica çizim örnekleri 12

Örnek 13:

```
uples = Tuples[Range@4, 2];
disks = Disk[#, 9/10] & /@ tuples;
```

```
circles = Circle[#, 9/10] & /@ tuples;
nF[x_] := Module[{d = DeleteCases[disks, x]},
Pick[d, RegionDisjoint[#, x] & /@ d, False]]
boolReg[n_] := Module[{bCF =
BooleanCountingFunction[{n},
Length@nF@#]}, DeleteCases[
RegionIntersection[#,
BooleanRegion[bCF, nF@#]],
_EmptyRegion]] &r1 =
Show[Region[#, BaseStyle -> Yellow]
& /@ boolReg[0] /@ disks,
Graphics[{Gray, circles}]]
```

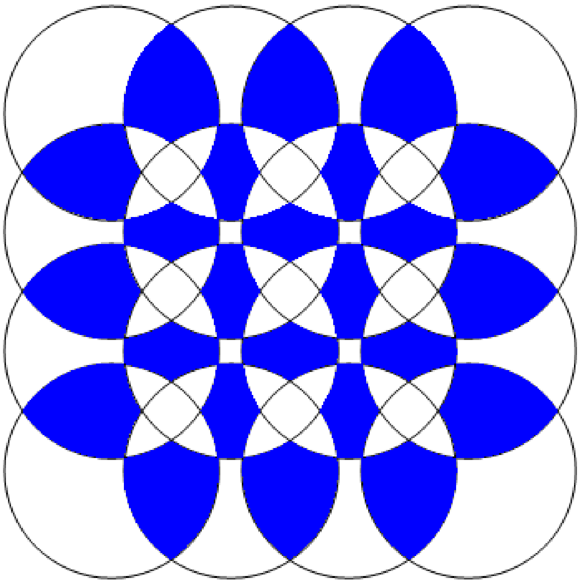


Şekil 2.56. Mathematica çizim örnekleri 13

Örnek 14:

```
tuples = Tuples[Range@4, 2];
disks = Disk[#, 9/10] & /@ tuples;
circles = Circle[#, 9/10] & /@ tuples;
nF[x_] := Module[{d = DeleteCases[disks, x]},
Pick[d, RegionDisjoint[#, x] & /@ d, False]]
```

```
boolReg[n_] := Module[{bCF =
BooleanCountingFunction[{n},
Length@nF@#]}, DeleteCases[RegionIntersection[#,
BooleanRegion[bCF, nF@#]],_EmptyRegion]] &
r2 = Show[Graphics[circles],
Region[#, BaseStyle -> Blue] &
/@ boolReg[1] /@ disks]
```

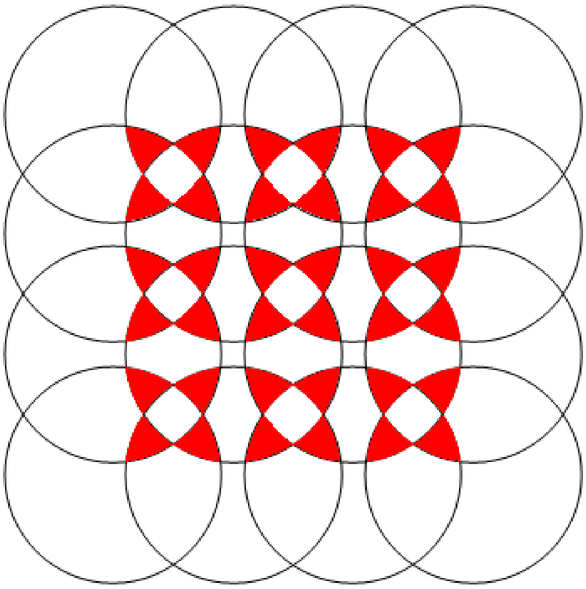


Şekil 2.57. Mathematica çizim örnekleri 14

Örnek 15:

```
tuples = Tuples[Range@4, 2];
disks = Disk[#, 9/10] & /@ tuples;
circles = Circle[#, 9/10] & /@ tuples;
nF[x_] := Module[{d = DeleteCases[disks, x]},
Pick[d, RegionDisjoint[#, x] & /@ d, False]]
boolReg[n_] := Module[{bCF =
BooleanCountingFunction[{n}, Length@nF@#]},
DeleteCases[RegionIntersection[#,
BooleanRegion[bCF, nF@#]], _EmptyRegion]] &
r3 = Show[Graphics[circles],
```

```
Region[#, BaseStyle -> Red] &
/@ boolReg[2] /@ disks]
```

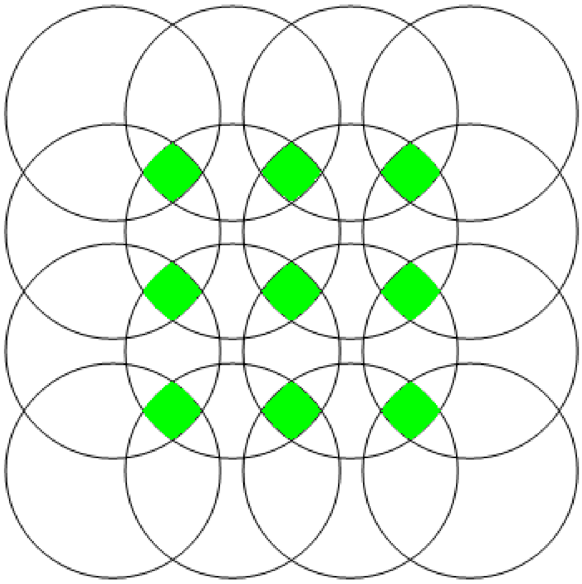


Şekil 2.58. Mathematica çizim örnekleri 15

Örnek 16:

```
tuples = Tuples[Range@4, 2];
disks = Disk[#, 9/10] & /@ tuples;
circles = Circle[#, 9/10] & /@ tuples;
nF[x_] := Module[{d =
DeleteCases[disks, x]},
Pick[d, RegionDisjoint[#, x]
& /@ d, False]]
boolReg[n_] := Module[{bCF =
BooleanCountingFunction[{n},
Length@nF@#]}, DeleteCases[
RegionIntersection[#,
BooleanRegion[bCF, nF@#]],
_EmptyRegion]] &
r4 = Show[Graphics[circles],
Region[#, BaseStyle -> Green]
```

```
& /@ boolReg[3] /@ disks]
```

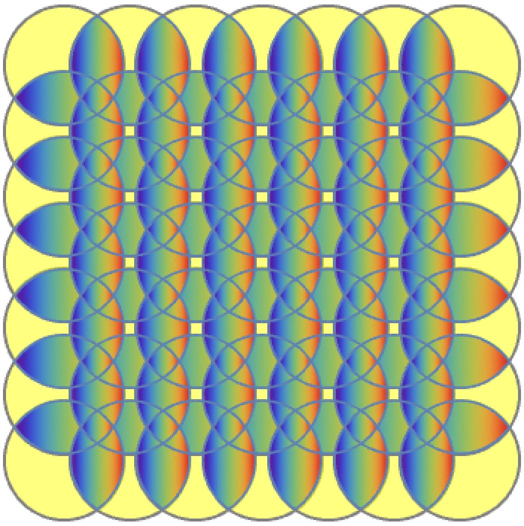


Şekil 2.59. Mathematica çizim örnekleri 16

Örnek 17:

```
tuples = Tuples[Range@4, 2];
disks = Disk[#, 9/10] & /@ tuples;
circles = Circle[#, 9/10] & /@ tuples;
nF[x_] := Module[{d = DeleteCases[disks, x]},
Pick[d, RegionDisjoint[#, x] & /@ d, False]]
boolReg[n_] := Module[{bCF =
BooleanCountingFunction[{n},
Length@nF@#]}, DeleteCases[RegionIntersection[#,
BooleanRegion[bCF, nF@#]], _EmptyRegion]] &
intersections = DeleteCases[RegionIntersection @@@
Subsets[(Disk[#, 9/10] & /@
Tuples[Range[7], 2]),{2, 4}], _EmptyRegion];
Show[Graphics[{Opacity[.5, Yellow],
EdgeForm[{Gray, Thick}], Disk[#, 9/10]
& /@ Tuples[Range[7], 2]}],
RegionPlot[#, PlotStyle -> RandomColor[]]
```

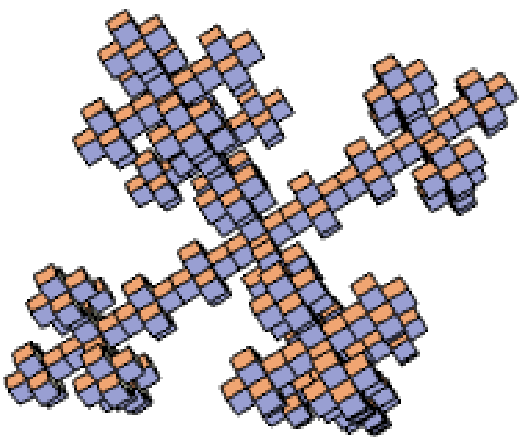
```
& /@ intersections]
```



Şekil 2.60. Mathematica çizim örnekleri 17

Örnek 18:

```
p = {{-1, 0, 0}, {0, -1, 0}, {0, 0, -1},
{0, 0, 0}, {0, 0, 1}, {0, 1, 0}, {1, 0, 0}};
f[x_] := Scale[Translate[x, p], 1/3]
Graphics3D[Nest[f, Cuboid[], 3], Boxed -> False]
```

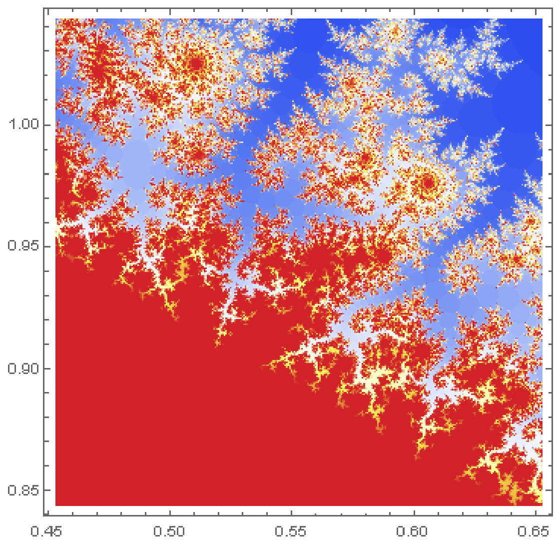


Şekil 2.61. Mathematica çizim örnekleri 18

Örnek 19:

```
cosineEscapeTime :=
```

```
Compile[{{c, _Complex}},
Block[{z = c, n = 2,
escapeRadius = 10
\[Pi], maxIterations = 100},
While[And[Abs[z] <= escapeRadius,
n < maxIterations],
z = Cos[z] + c; n++]; n]]
Block[{center =
{0.5527, 0.9435}, radius = 0.1},
DensityPlot
[cosineEscapeTime[x + I y],
{x, center[[1]] - radius,
center[[1]] + radius},
{y, center[[2]] - radius,
center[[2]] + radius},
PlotPoints -> 250, AspectRatio -> 1,
ColorFunction -> "TemperatureMap"]]
```

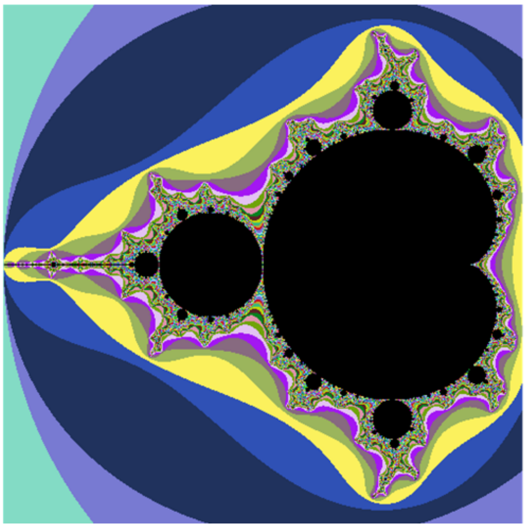


Şekil 2.62. Mathematica çizim örnekleri 19

Örnek 20:

```
mlf = LibraryFunctionLoad
```

```
["demo_numerical",
"mandelbrot", {Complex}, Integer];
n = 501; samples =
Table[mlf[x + I y],
{y, -1.25, 1.25, 2.5/(n - 1)},
{x, -2., .5, 2.5/(n - 1)}];
colormap =
Function[If[# == 0,{0., 0., 0.},
Part[r, #]]] /.
r -> RandomReal[1, {1000, 3}];
Graphics[Raster
[Map[colormap, samples,
{2}]], ImageSize -> 512]
```



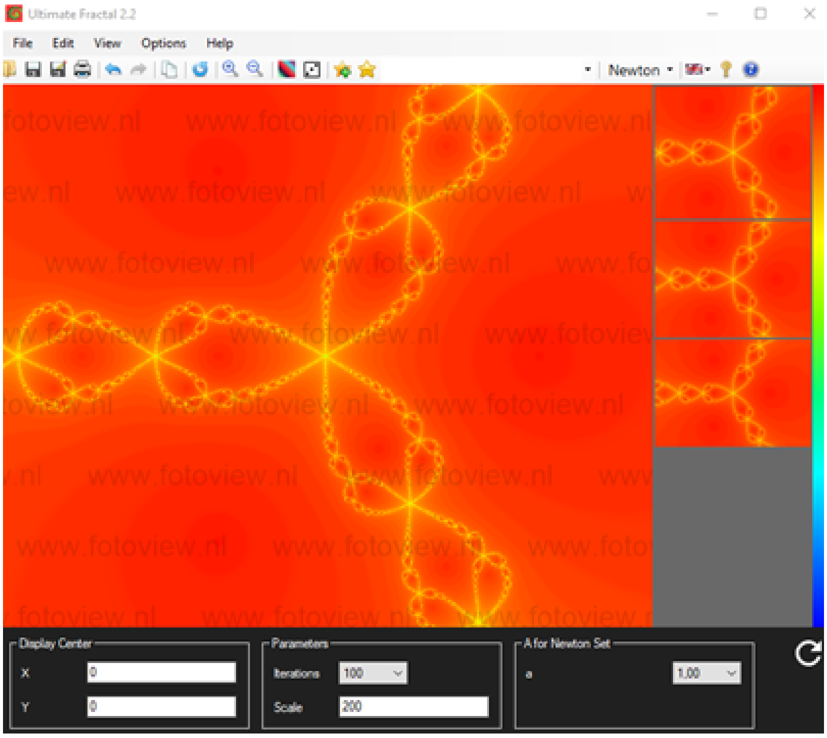
Şekil 2.63. Mathematica çizim örnekleri 20

2.5.3. Diğer paket programlarda çizimler

Aşağıda listelenen programlar isim olarak aratıldığında her bilgisayara indirilip kullanılabilir. Programlar sırasıyla açıklanmakta ve örneklerle desteklenmektedir.

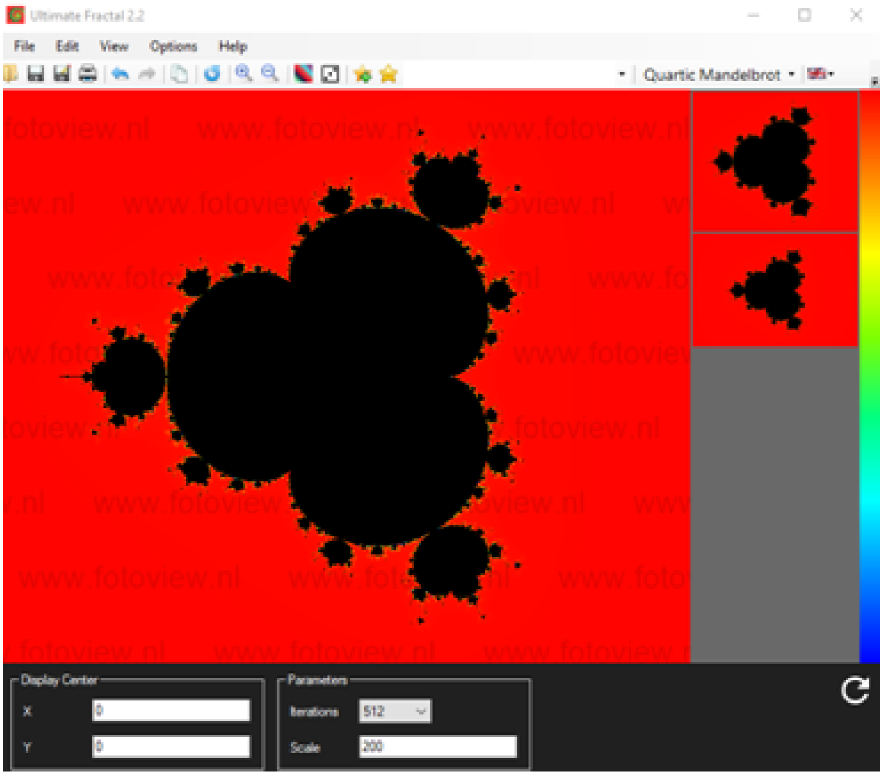
1) Ultimate Fraktal 2.2: Program *https://ultimate-fractal.soft112.com/* sitesinden online olarak indirilir. Programda herşey hazır paket halindedir. En bilinen fraktallar kayıtlıdır. Yalnızca tek tek seçilip üzerlerinde renk, boyut değişimi yapılabilmektedir.

Örnek 1:



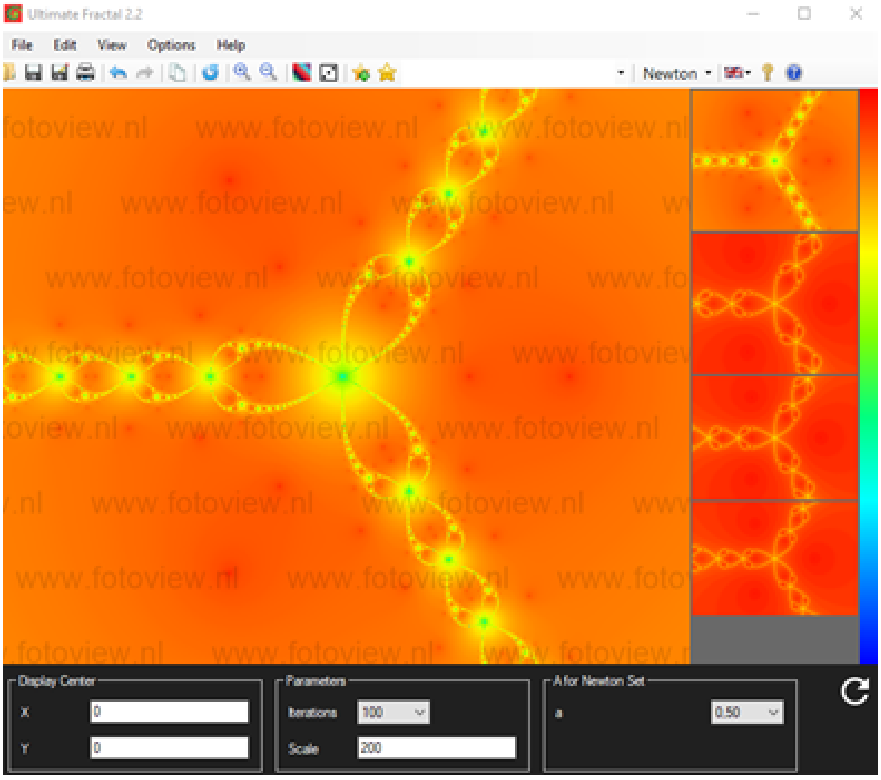
Şekil 2.64. Ultimate fraktal çizim örnekleri 1

Örnek 2:



Şekil 2.65. Ultimate fraktal çizim örnekleri 2

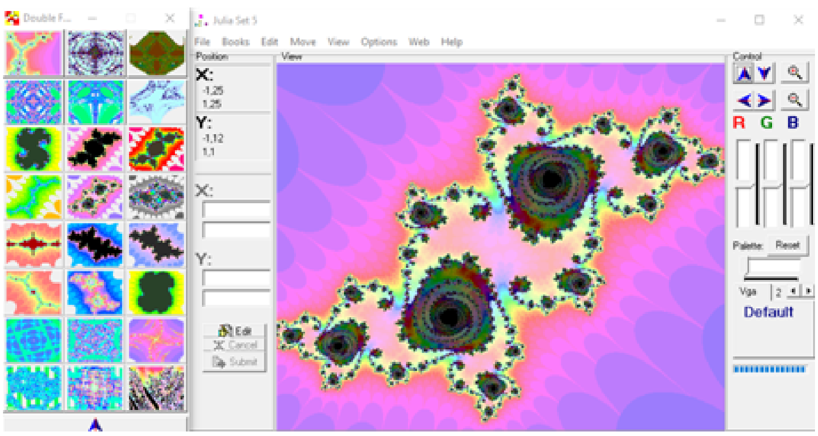
Örnek 3:



Şekil 2.66. Ultimate fraktal çizim örnekleri 3

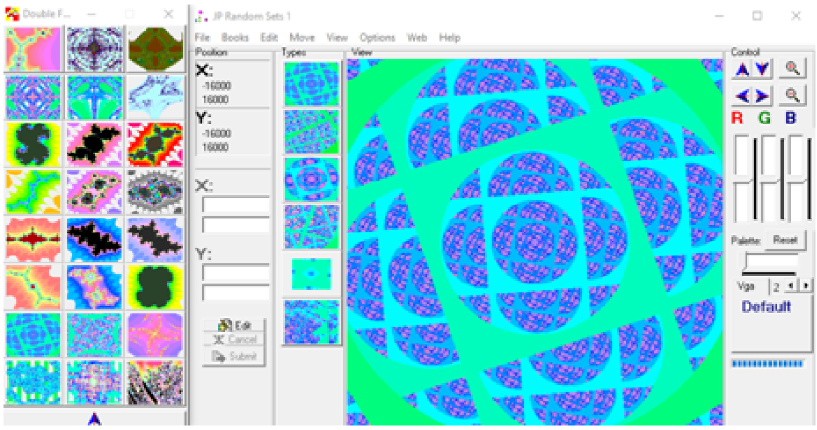
2) Double Fractal Exe: Program *https://double-fractal.soft32.com/* sitesinden indirilir. Programda herşey hazır paket halindedir. En bilinen fraktallar kayıtlıdır. Tek tek seçilip üzerlerinde renk, boyut değişimi yapılabilmektedir. Bunlar dışında bir değişime izin verilmemektedir.

Örnek 1:



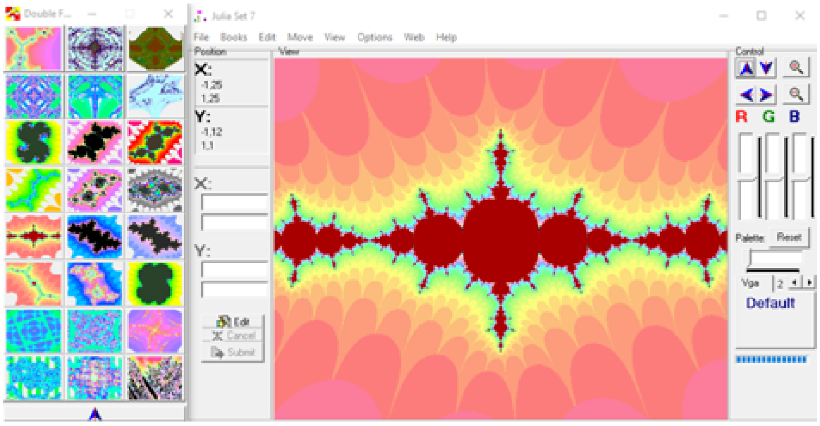
Şekil 2.67. Double fraktal çizim örnekleri 1

Örnek 2:



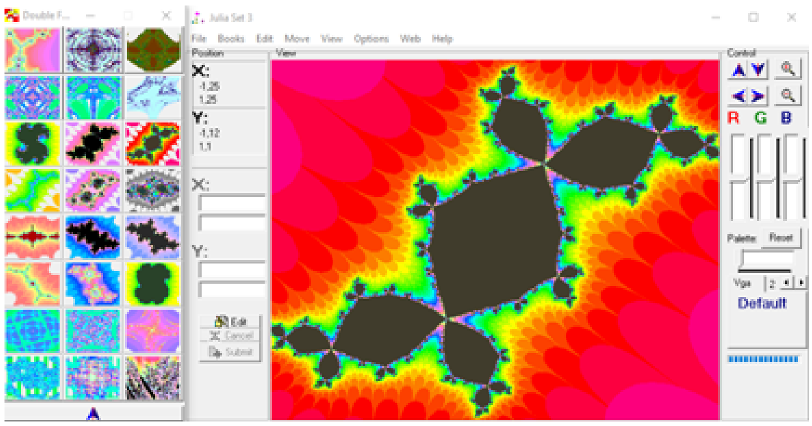
Şekil 2.68. Double fraktal çizim örnekleri 2

Örnek 3:



Şekil 2.69. Double fraktal çizim örnekleri 3

Örnek 4:

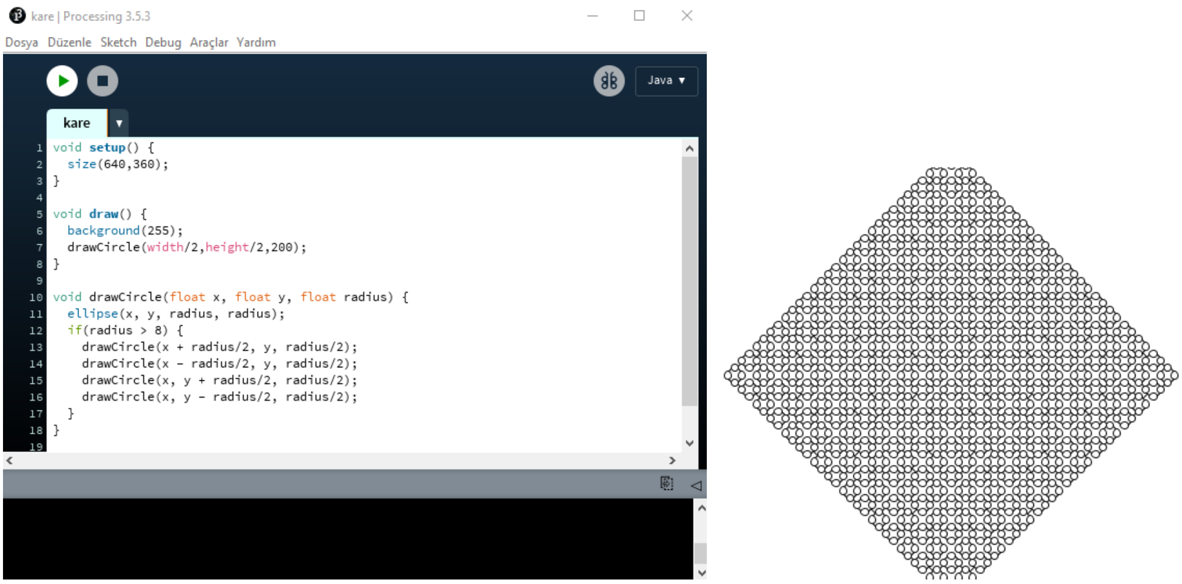


Şekil 2.70. Double fraktal çizim örnekleri 4

3) Proccessing 3.5.3: Programı *https://processing.org/download* sitesinden indirilir. Program kendi programlama diline sahiptir. Kodları, bilinen Cplus dilleriyle benzerlik gösterir. Kullanımı tamamiyle kendine özgüdür.

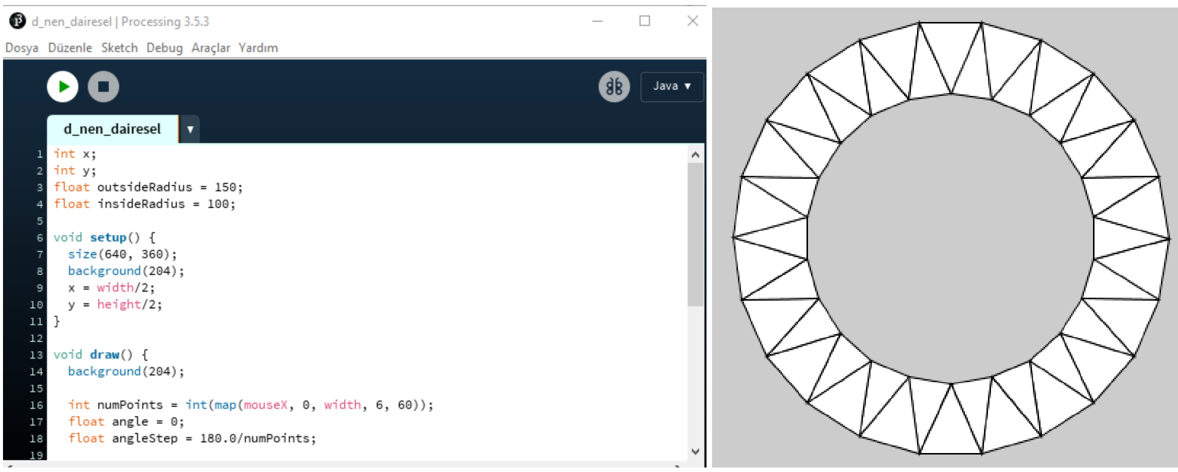
"Proccessing" ismiyle aratıldığında hazır kodları bulunmaktadır. Kendi uygulaması içinde de hazır kodlar vardır.

Örnek 1:



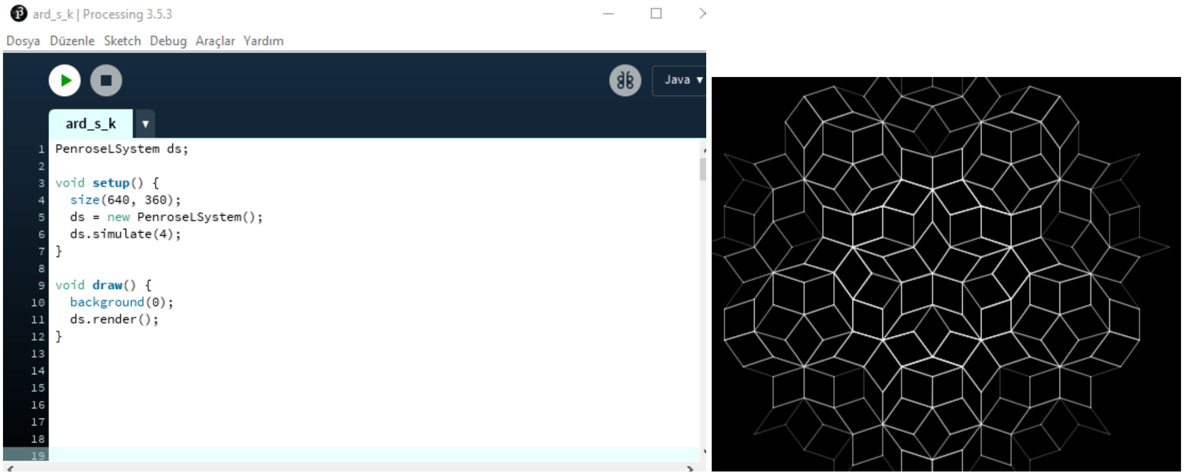
Şekil 2.71. Proccessing Çizim Örnekleri 1

Örnek 2:



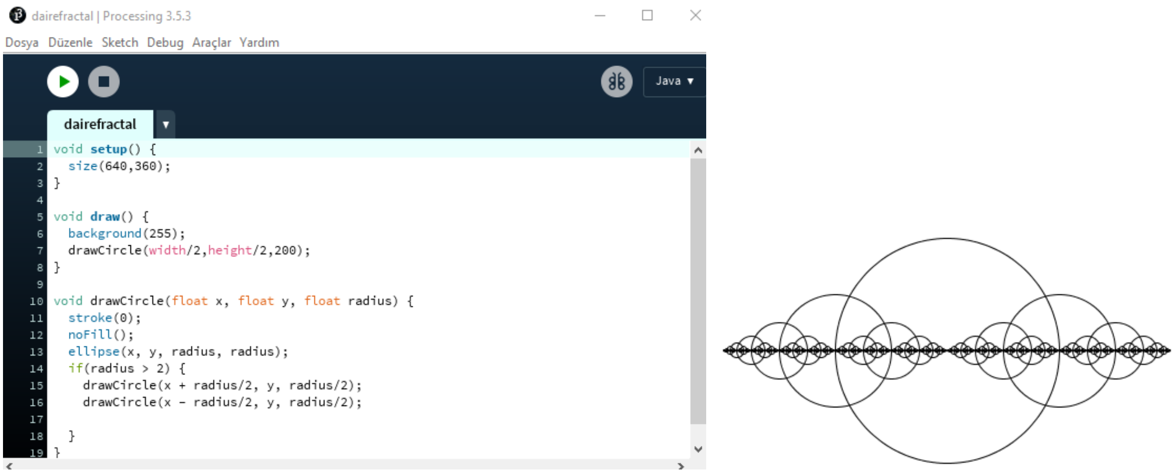
Şekil 2.72. Proccessing Çizim Örnekleri 2

Örnek 3:



Şekil 2.73. Proccessing Çizim Örnekleri 3

Örnek 4:



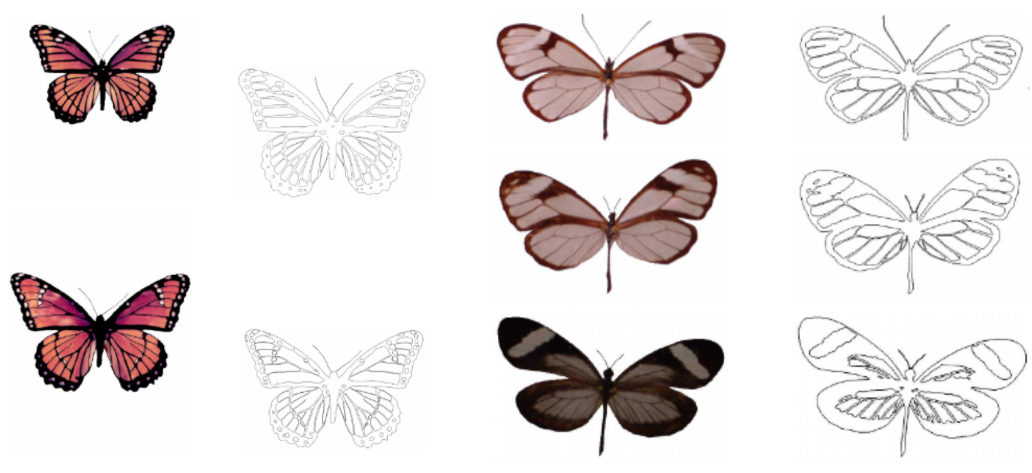
Şekil 2.74. Proccessing Çizim Örnekleri 4

3. FRAKTALIN VARLIĞI VE UYGULAMA ALANLARINDAN BAZILARI

Fraktal, belirli oranlarda küçülerek veya büyülerek oluşturulan şekilin her bir parçasında büyük şekili görmektir. Fraktal birbirini tekrar eden ve kendine benzerlik gösteren bir yapıya sahiptir. Öyle ki bazı yapılarda fraktalı görmek mümkündür. Doğa, bilim sayesinde daha anlaşılabilirdir. Doğadaki şekilleri tanımlarken matematiğin bir kolu olan geometrinin büyük bir rolü vardır. Çünkü geometri, insanların doğayı nasıl algıladığı ile doğrudan ilişkilidir. Bunu sağlayanda Öklid geometrisidir. Öklid geometrisiyle doğadaki şekiller üçgenler, daireler, küreler, doğrulardır ancak bu şekiller doğada var olan karmaşık yapıları anlamak için yeterli değildir. İncelendiğinde doğadaki şekillerin Öklid geometrisindeki şekillere benzemediği, daha karmaşık ve düzensiz olduğu görülmektedir. Güneş bir küre şeklinde değil veya dağlar bir üçgen şeklinde değildir. Bazen bir kelebeğin kanadında, bir çiçekte, beyin kıvrımlarında bakıldığında fraktal yapıların varlığından bahsedilebilir ve fraktal kavramı bu sayede bir çok uygulama alanı bulmuştur [20, 21].

3.1. Kelebek Kanatlarında Fraktal

Kelebek kanatları hem çok karmaşık hemde çok büyüleyici yapılardır. Bu doğa harikası yapılar fraktalında geliştirilmesiyle fraktal ile bağdaştırılmıştır. Kelebek kanatlarındaki desen tıpkı fraktal gibi kendi tekrar eden bir yapıya sahiptir. Genel olarak, kelebeğin kanatlarındaki sol ve sağ desenler simetriktir. Bununla ilgili çeşitli çalışmalar vardır [22, 23].

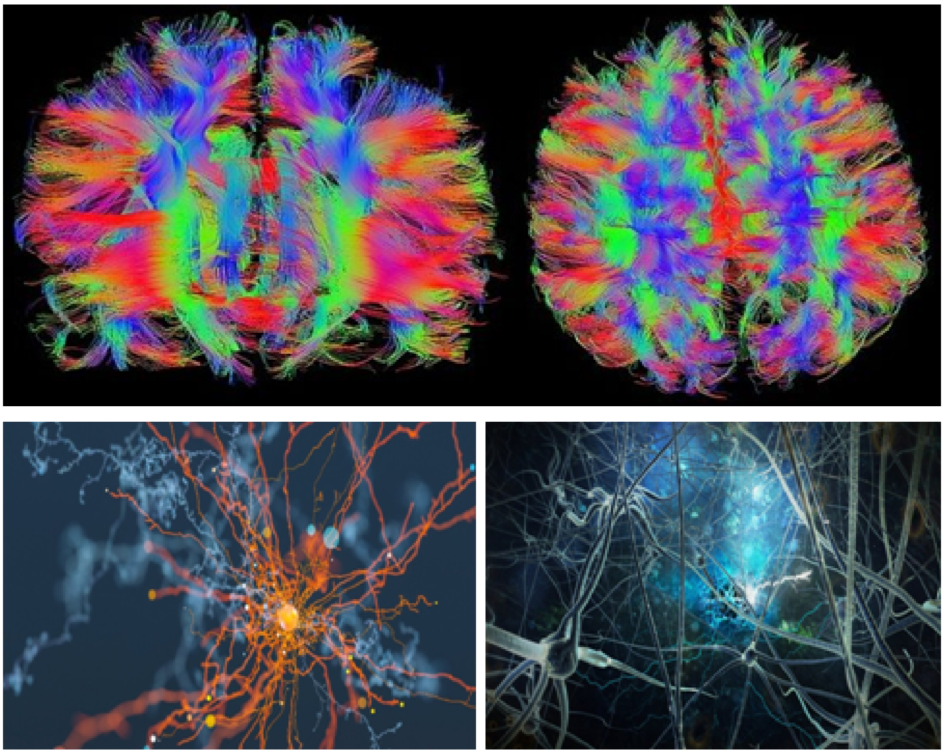


Şekil 3.1. Kelebek kanatları

Örnekleri yukarıda Şekil 3.1'de verilmiştir.

3.2. Beyin Hücrelerinde Fraktal

İnsan beyni, birbiriyle karmaşık yapıya sahip nöronlardan oluşur. Bu yapı aktiviteleri kontrol eder, zekayı, duyuları oluşturur. İnsan ve diğer canlılarda yaşamsal faaliyetleri yerini getirir. Nöronlar birbirleriyle çok fazla bağlantıya sahiptir ve bu bağlantılar sayesinde birbiriyle etkileşim halindedir. Gelişmiş bilgisayarlarla karşılaştırılamayacak şekilde karmaşık ve daha üstün yapıya sahiptir. Matematiksel olarak bakıldığında kaos teorisi olarak düşünülüp beynin üst düzey fonksiyonlarının modellenmesinde önemli bir rol oynamaktadır. Kaos teorisinin bir uzantısı olarak düşünülebilecek fraktal geometriyle bağlantısıda burada başlar. Nöronların bu karmaşık ama bir o kadar da düzenli yapısı fraktal geometriye örnek oluşturur. Aynı nöron yapılarının birbirini tekrar etmesi ve düzenli döngüye sahip olması en önemli faktördür [24, 25]. Bu nöron yapılarının farklı açılardan görselleri ve örnekleri aşağıda Şekil 3.2'de gösterilmektedir.



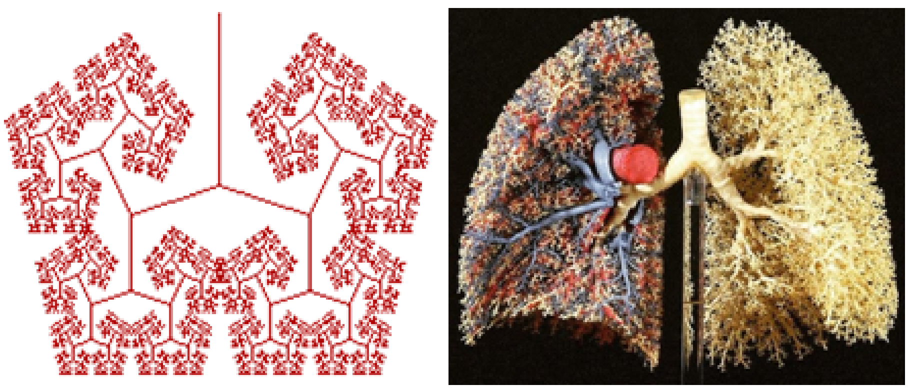
Şekil 3.2. Nöron hücreleri

Beynin yüzeyi manifold yapıdadır ve çok sayıda katlanmalardan oluşur. Öyle ki insan beyninin yüzeyi hayvanlara göre çok daha katlanmalara sahiptir. Yüzey diferensiyellenebilirdir ve fraktal boyutuda ikiden büyüktür. İnsanlarda bu boyut 2.73 ile 2.79 arasında değişkenlik gösterir [9].

3.3. Akciğer Hüclerinde Fraktal

Fraktallar için insan vücudundaki en belirgin örneklerden biri akciğer hücreleridir. Nefes alıp vermekte kullanılan bu sistem tüplerden oluşur. Alınan hava nefes borusundan geçip farklı iki boruya ayrılır. Daha sonrada bronş denilen en küçük torbalara geçip bir sonraki adımda dahada küçük gözelere ayrılarak sistemi devam ettirir. Bu şekilde sürekli ayrılarak devam eden sistem bir fraktal yapısıdır.

Akciğerlerin birer fraktal oluşturduğunun en belirgin kanıtlarından biri de gözelerin alanlarının ölçüsüdür. Işık mikroskobu ile ölçüldüğünde 80 m^2 ve elektron mikroskobu ile ölçüldüğünde ise 140 m^2 kadardır. Alanın gittikçe kusursuz şekilde büyümeside fraktalların bir örneğidir [9]. Akciğer gözelerinin farklı açılardan görselleri aşağıda Şekil 3.3'de gösterilmektedir.

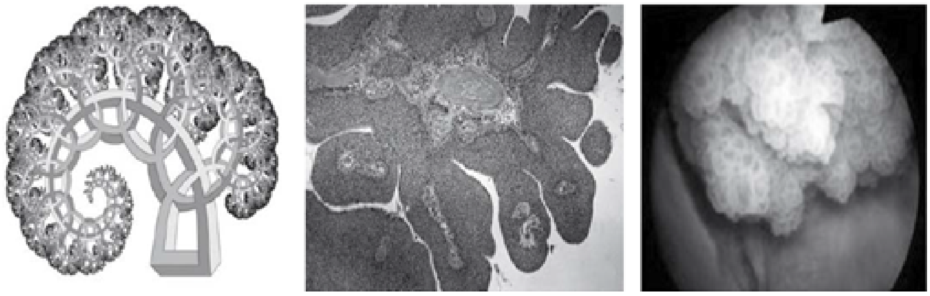


Şekil 3.3. Akciğer gözeleri

3.4. Kanser Hücrelerinde Fraktal

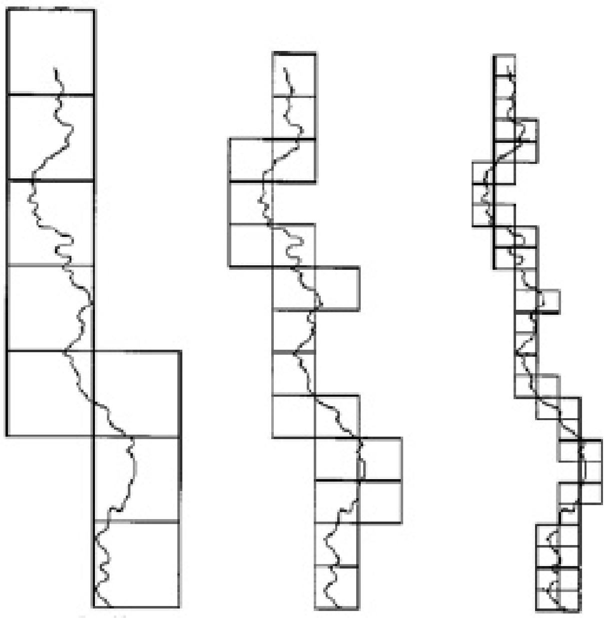
Kanser hücreleri de fraktallar gibi kaotik bir yapıdır, bu hücrelerinin düzensiz şekilleri, damarlanması bildiğimiz Öklid geometrisi (doğru, düzlem, küre gibi) ile tanımlanamaz. Fraktal geometride bu düzensizlikleri açıklayıp ölçmeyi sağlayacak en önemli kaynaklardan biridir [26].

Kanserin hakkındaki bilgi artışına rağmen çoğu zaman tanısı, radyolojik görsellerin incelenmesine, biyopsi örneklerinin mikroskop ile incelenmesine veya dokunun direkt gözlenmesine dayanmaktadır oysa daha net daha bilimsel bir yaklaşımla bilgisayar analizlerine ihtiyaç vardır. Fraktal boyut da bu konuda oldukça yardım sağlamaktadır. Hücrelerin fraktal yapıya benzerliği aşağıdaki Şekil 3.4'de görülmektedir;



Şekil 3.4. Kanser hücreleri

Geleneksel matematik yöntemleriyle net bilgi elde etmek pek mümkün değildir. Bu sebeple hücre kenarlarındaki dalgalanmaları hesaplamak için fraktal geometride sık kullanılan Kutu Sayma Metodu kullanılmaktadır. Kenardaki kıvrımlar kutucuklara bölünüp böylece iki boyutlu yüzeye geçirilip hesaplanmaktadır. Şekil 3.5'de gösterilir. Böylece kutular tam olarak şekli kaplar ve boyutu hesaplanır.



Şekil 3.5. Kıyı şeridinin kutu sayma metoduyla kaplanması

Kıvrımların kutularla kaplanması Şekil 3.5'te gösterilmiştir.

3.5. Seramikte Fraktal

Örnek 1) Hacettepe Üniversitesi Seramik Anasanat Dalı yüksek lisans öğrencisi Ceren Genç tarafından yayımlananan tez çalışmasında yaptığı seramikler yer almaktadır. Uygulamalarında doğanın en temel özelliklerinden olan süreç, denge, döngü, kaos, bağlılık, sarmal oluşum gibi kavramlar vardır. Çalışmalarında, doğanın fraktal yapısının özü olan ve aynı zamanda sanatta da birer değer olan denge, ritm, düzen form gibi ögeleri kullanarak sanatsal nesneler ortaya koymayı amaçlamıştır. Oluşturulan formlar keskinlikten uzak, yumuşak geçişler ile yapılmıştır.

Aşağıda Ceren Genç'in kişisel çalışmaları Şekil 3.6'de sıralanmıştır [27].

Şekil 3.6. Seramik çalışmaları 1

Örnek 2) Hacettepe Üniversitesi Seramik Anasanat Dalı yüksek lisans öğrencisi Aylin YILMAZ tarafından yayımlananan tez çalışmasında yaptığı seramikler yer almaktadır. Fraktal geometrinin keşfiyle kompozisyon kavramına yeni bir bakış açısı getirilmiş ve seramikte bundan etkilenmiştir. Birçok sanatçı çalışmalarında fraktaldan etkilenmiş farklı kompozisyonlarda işler yapmışlardır. Fraktalda boyut kavramı kompozisyon oluşturmada yeni alternatifler sunmuştur. Sanatın taklit ve duyusal yönüyle ilgilenen Platon gibi, seramik sanatçısı da doğadan aldığı malzemeyi yine doğayı taklit ederek şekillendirir.

Aşağıda Aylin YILMAZ'ın kişisel çalışmaları Şekil 3.7'de sıralanmıştır [28].

Şekil 3.7. Seramik çalışmaları 2

4. FRAKTALA GÖNÜL VERENLER

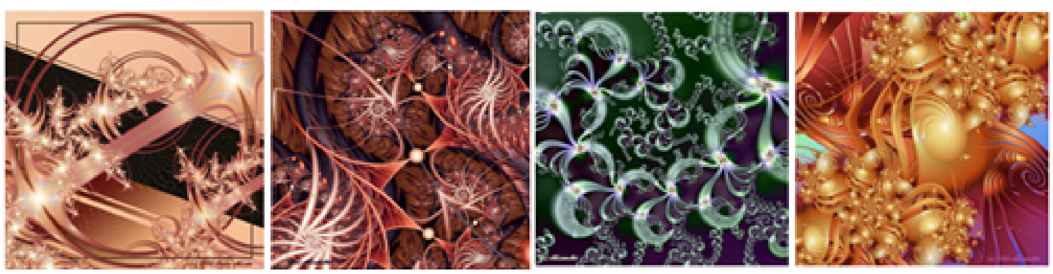
4.1. H.Hilmi Hacısalihoğlu

1942 yılı Trabzon doğumludur. 1963'de Ankara Üniversitesi Fen Fakültesi Matematik-Astronomi bölümü mezunudur. 1966'da doktorasını tamamlamıştır. 1972 yılında doçent ve 1976 yılında profesör olmuştur. 1969-1971 yılları arasında Brown Üniversitesi'nde (ABD) araştırmalar yapmıştır. Çeşitli üniversitelerde kurucu bölüm başkanlığı yapmış, bir çok çalışmaya öncülük etmiştir. Aynı zamanda Prof. Dr. Hilmi HACISALİHOĞLU Matematik bölümlerinin vazgeçilmez kitaplarının yazarıdır. 38 adet kitabı mevcuttur. Bunlardan biride Fraktal Geometri adlı kitabıdır. Ayrıca ingilizceden çevirdiği Gerald A. Edgar'ın Ölçü, Topoloji ve Fraktal Geometri kitabı da mevcuttur [29].

Ülkemize fraktal geometriyi tanıtan, sevdiren, öncüllük eden hocamızdır. Petek ve parmak izlerinin fraktal olarak incelemiş ve tezlere öncülük etmiştir [30, 31].

4.2. Linda Allison

ABD'nin Florida'da eyaletinde yaşayan engelli bir ev hanımı olan Linda 1994'ten beri boş zamanının bir kısmını fraktal görüntüler tasarlamaya adamıştır. Resmi bir matematik eğitimi olmamasına rağmen Linda, sonsuzluk kavramını renk paletleriyle görüntülerine yansıtma konusunda inanılmaz bir yeteneğe sahiptir [32]. Örnek görselleri Şekil 4.1'deki gibidir.

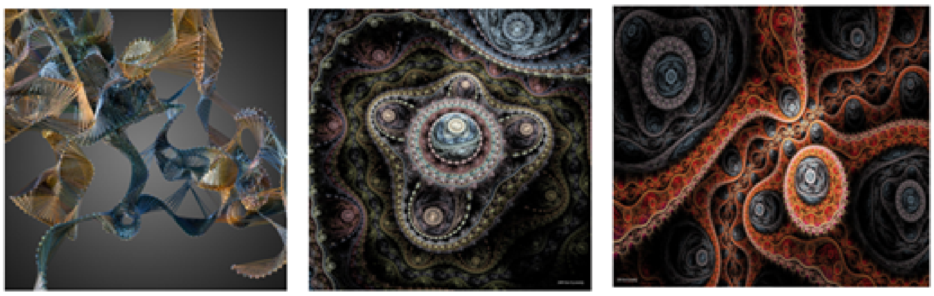


Şekil 4.1. Linda Allison

4.3. Dan Kuzmenka

Dan Kuzmenka, kimya alanıyla ilgilenen Amerikalı bir bilim insanıdır. Dan fraktal geometriyi 1985'te Scientific American dergisinde bir makale okuyarak keşfetmiştir ancak ilk fraktal görüntülerini 1999'da oluşturmaya başlamıştır. Sanatçı genellikle daha sıcak renkler ve toprak tonları kullanmaktadır [32].

Örnek görselleri Şekil 4.2'deki gibidir.

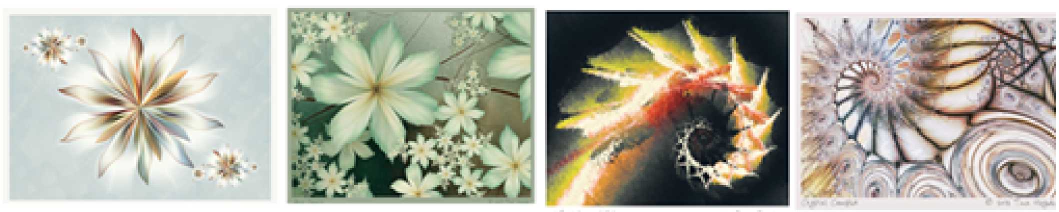


Şekil 4.2. Dan Kuzmenka

4.4. Tina Oloyede

Tina Oloyede, aslında tıp doktoru olan bir sanatçıdır. Çok sevdiği ve gönül verdiği fraktallara daha fazla vakit ayırmak için doktorluğu bırakmıştır. İngiltere'de yaşayan sanatçı geçimini bu sanatla sağlamaktadır. Tina Oloyede'nin sanatsal ifade kapasitesi tartışılmazdır ve oldukça yeteneklidir [32].

Örnek görselleri Şekil 4.3'deki gibidir.



Şekil 4.3. Tina Oloyede

4.5. Janet Parke

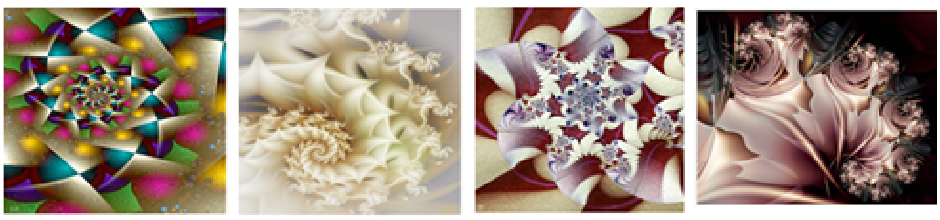
Birleşik Devletler'in güney doğusundaki Tennessee eyâletinde bulunan bir kent olan Memphis'te doğan Janet Parke, hayatının büyük bir bölümünü balerin, koreograf ve dans profesörü olarak geçirmiştir. 1999 yılında, o zamana kadar pek bilinmeyen yeteneğiyle fraktal sanatını sergilemeye ve satmaya başlamıştır [32]. Örnek görselleri Şekil 4.4'deki gibidir.

Şekil 4.4. Janet Parke

4.6. Joe Zazulak

Joe Zazulak, 55 yaşında Amerika Birleşik Devletleri Gazi İşleri Bakanlığı'ndan emekli olmuş ve bundan sonra kendisini fraktal sanatını sergilemeye adamıştır. Joe Zazulak, resimlerini önceden planlamaz ve yaratıcı süreçten sonra ne olacağını bilemeden tasarlar. Çalışmalarına çok basit bir yapıyla, neredeyse hiç renk vermeden başlar ve hoş bir sonuç elde edene kadar sezgisel olarak şekil varyasyonları ekler. En sonunda harika yapıtlar elde eder [32].

Örnek görselleri Şekil 4.5'deki gibidir.

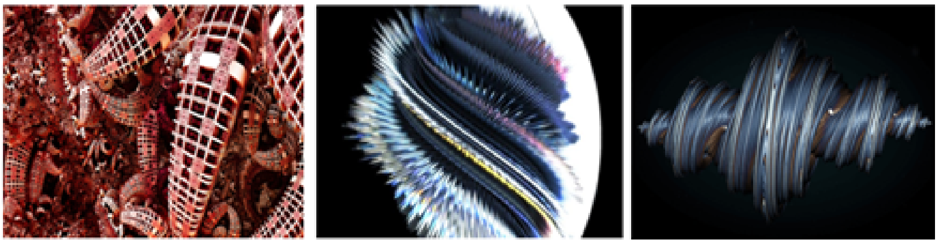


Şekil 4.5. Joe Zazulak

4.7. Ozan Türkkan

Tekirdağ doğumlu olan sanatçı ABD, Avrupa ve İstanbul'da çeşitli çalışmalarda bulundu. Türkkan, genel olarak deneysel medya ve dijital sanatlarda çalışmış olup, çalışmaları özellikle jeneratif sanat, algoritmik sanat, fraktal geometri, deneysel video ve enstalasyonlar üzerinde yoğunlaştırmaktadır [33].

Örnek görselleri Şekil 4.6'deki gibidir.

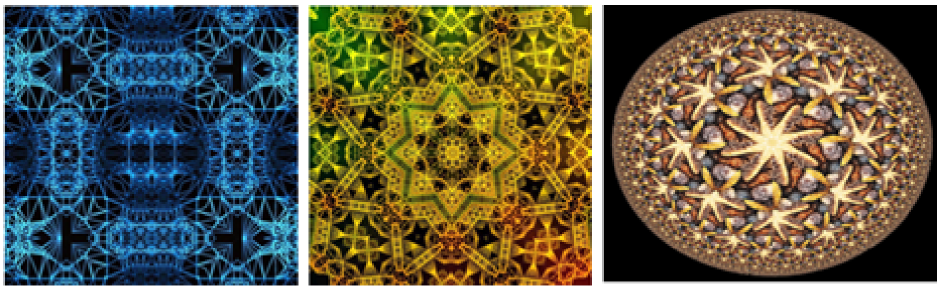


Şekil 4.6. Ozan Türkkan

4.8. Ali Öner

1958 yılında Ege'de doğan Ali Öner, 1981 yılında Hacettepe Üniversitesi'nden Kimya Mühendisi olarak mezun olmuştur. 1999 yılında matematiksel formüllerle üretilen "Fraktal" ile tanışıp dijital renkler ve soyut şekiller kullanıp sanatını yaratmaya başlamıştır [33].

Örnek görselleri Şekil 4.7'deki gibidir.



Şekil 4.7. Ali Öner

5. SONUÇ

Mandelbrot'un öncülük ettiği fraktal geometri son yıllarda ülkemizde de yaygın bir alan olarak çalışılmaktadır. Farklı bilim dallarına da öncülük edip yol açan bu bilim dalını dahada yaygınlaştırıp nerelerde ne şekilde kullanıldığının bilinmesi amaçlanmıştır. Doğada, bakılan bir resimde, heykelde hatta insan vücudunda bile rastlamak mümkündür [34].

Bu alanda kalıcı birşeyler bırakmayı amaçlayarak bu tezde fraktaldan, boyutundan, kullanım alanlarından ve son olarak da çeşitli programlarda görseller verilmektedir.

Mandelbrot'un bir sözüyle bitirmek gerekirse fraktal için şunlar söylenebilir; *İnsanlar ilk başta bunun tamamen dünya dışına ait bir şey olduğunu düşündüler ama sonra, çok kısa bir süre sonra geri gelip şöyle dediler: "Biliyor musunuz, bunlar bana bir şey hatırlatıyor. Bence bunlar doğal. Kâbus ya da rüya gibiler ama doğallar."*

KAYNAKLAR

1. Mahalu, G., Graur, A. (2015). The Fractal Techniques Applied in Pattern Recognition. *International Journal of Fuzzy Logic and Intelligent Systems*, 15(2), 121-125.

2. İnternet: Shiffman, D. (2012). Fractals The Nature of Code: Chapter 8. Web: https://natureofcode.com/book/chapter-8-fractals/. Erişim Tarihi: 18.12.2021

3. İnternet: Norman, J. (1980). Vol Libre: The First Fractal CGI Movie. Web: https://www.historyofinformation.com/detail.php?entryid=3690. Erişim Tarihi: 09.01.2022.

4. Stepney, S. (1983). Incredible fractals. *Acorn User*, 10.

5. Briggs, J. (1992). *Fractals: The Patterns of Chaos*. Simon and Schuster, 182.

6. Mandelbrot, B. B. (2010, 13 Şubat). *TedTalks'da Fraktallar ve Pürüzlülük Sanatı Üzerine Söyleşi*, Long Beach, California, 1-20.

7. İnternet: Cotak, G. (Temmuz, 2015). Doğada Fraktal ve Simetri Örnekleri. Web: https://ortaokulmatematik.com/dogada-fraktal-ve-simetri-ornekleri/ Erişim Tarihi: 11.10.2021.

8. Kavlak, İ. (2006). Doğadaki Fraktallar. *Journal of İstanbul Kültür Üniversitesi*, 3, 105-112.

9. Hacısalihoğlu, H. H. (2017). *Fraktal Geometri*. Ankara: Hacısalihoğlu Yayınları, 47.

10. Gök, İ. (2013). *Fraktal Geometri-1*, Ankara: Açık Ders, 117-149.

11. Hubbard, J., Schleicher, D., Sutherland, S. (2001). How to find all roots of complex polynomials by Newton's method. *Inventiones mathematicae*, 146, 1-33.

12. Barnsley M. (1993). *Fractals Everywhere*. Cambridge: Academic Press, 351.

13. Çimen, M. E., Boyraz, Ö. F., Garip, Z., Pehlivan, İ., Yıldız, M. Z., Boz, A. F. (2021). Görüntü işleme tabanlı kutu sayma yöntemi ile fraktal boyut hesabı için arayüz tasarımı. *Politeknik Dergisi*, 24(3), 867-878.

14. Duman, E. (2012). *Nümerik Analiz*, İstanbul: İstanbul Kültür Üniversitesi, 51-53.

15. Corte, J. (2003). Fractal Images Generated by Newton's Method. *Tennessee Research and Creative Exchange*, 8, 11-14.

16. Young, A. P. *Fractals from Newton's Method*. University of California, Santa Cruz, 115-120.

17. McClure, M. Newton's method for complex polynomials. *Mathematica in Education and Research*, 1-14.

18. İnternet: Fraktal örnekleri. Web: https://mathematica.stackexchange.com/ ve https://tex.stackexchange.com/. Erişim Tarihi: 25.12.2021.

19. İnternet: Tikz ve Pgf Örnekleri. Web: https://texample.net/tikz/examples/all. Erişim Tarihi: 13.01.2021.

20. Uyar, A., Öztürk, D. (2017). *Fraktal Analizin Yeryüzü Araştırmalarında Kullanılması*. Yüksek Lisans Tezi, Ondokuz Mayıs Üniversitesi Fen Bilimleri Enstitüsü, Samsun, 154-156.

21. İnternet: Sever, M. (Aralık, 2018). Fraktallar: Göz Kamaştıran Geometrik Şekiller. Web: https://evrimagaci.org/fraktallar-goz-kamastiran-geometrik-sekiller-7518. Erişim Tarihi: 08.06.2021.

22. Castrejon-Pita, A. A., Sarmiento-Galan, A., Castrejon-Pita, J R., Castrejon-Garcia, R. (2005). Fractal dimension in butterflies' wings: a novel approach to understanding wing patterns?. *Journal of Mathematical Biology*, 50(5), 94.

23. Dai, W., Chang, R., Shih, Z. (1995). Fractal pattern for a butterfly wing. *The Visual Computer*, 3, 177-187.

24. Leva, A. D. (2016). *The Fractal Geometry of the Brain*, Springer Science, New York, 65-79.

25. Losa, G. A. (2009). The Fractal Geometry of Life. *Rivista di Biologia*, 102(1), 29-59.

26. Narter, F. ve Köse, O. (2013). Kanser geometrisi ve mesane kanserinde fraktallar. *Üroonkoloji Bülteni*, 12(1), 11-17.

27. Genç, C. (2019). *Fraktal Geometri ile Sanatsal Pratikler*. Yüksek Lisans Tezi, Hacettepe Üniversitesi Güzel Sanatlar Enstitüsü, Ankara, 1-89.

28. Yılmaz, A. (2018). *Fraktal Seramik Yorumlamalar*. Yüksek Lisans Tezi, Hacettepe Üniversitesi Güzel Sanatlar Entitüsü, Ankara, 1-68.

29. Edgar, G. A. (2006). *Ölçü, Topoloji ve Fraktal Geometri*. (Çev. H. H. Hacısalihoğlu). Ankara: Nobel Akademik Yayıncılık, 150. (Eserin orjinali 1990'da yayımlandı).

30. Akpolat, F. (2014). *Parmak İzinin Fraktal Teorisi*. Yüksek Lisans Tezi, Bilecik Şeyh Edebali Üniversitesi ve Anadolu Üniversitesi Fen Bilimleri Enstitüsü, 7-10.

31. İnternet: H. Hilmi Hacısalihoğlu. Web: http://tonyaninsozu.net/. Erişim Tarihi: 05.04.2021.

32. İnternet: (2006). Fractal Art: Beauty and Mathematics. Web: http://www.ams.org/publicoutreach/math-imagery/mandelbrot. Erişim Tarihi: 05.04.2021.

33. Uyan, S. (2019). *Bir Tasarım Yöntemi Olarak Fraktal Tasarım*. Yüksek Lisans Tezi, Dokuz Eylül Üniversitesi Fen Bilimleri Enstitüsü, İzmir, 72-74.

34. Mandelbrot, B. B. (1983). *Fractal Geometry of Nature*. New York, ABD: W. H. Freeman and Company, 18.

GAZİ GELECEKTİR...