

14184

İSTANBUL TEKNİK ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

MARKOV ve GIBBS RASTLANTI ALAN MODELLERİ

İLE DOKU SENTEZLEME ve SINIFLANDIRMA

YÜKSEK LİSANS TEZİ

Müh. Erdoğan CAMCIOĞLU

Tezin Enstitüye Verildiği Tarih: 4 Haziran 1990

Tezin Savunulduğu Tarih : 27 Haziran 1990

Tez Danışmanı : Prof. Dr. Erdal PANAYIRCI

Diğer Jüri Üyeleri : Prof. Dr. Enise Erimez

Y. Doç. Dr. Ümit Aygözü

T. E.
Yükseköğretim Kurulu
Dokümantasyon Merkezi

HAZİRAN 1990

ÖNSÖZ

Doku analiz ve sentezindeki bir çok problemlerin çözümünde yaygın olarak kullanılan Markov ve Gibbs rastlantı alan modelleri gibi güncel bir konuda tez çalışması yapmamı sağlayan ve yardımlarıyla bu tezin hazırlanmasını yönlendiren Prof. Dr. Erdal PANAYIRCI'ya, tez çalışması boyunca konu ile ilgili kaynak bulmada ve elde edilen sonuçların değerlendirilmesinde yardımcı olan araştırma görevlisi Yük. Müh. Bilge GÜNSEL'e ve çalışmalarım boyunca bilgisayarlarını kullanmama izin veren ELEKTROMED Elektronik Medikal San. ve Tic. A. Ş. 'ne teşekkür ederim.

4-Haziran-1990
Erdoğan CAMCIOĞLU

İÇİNDEKİLER

ŞEKİL LİSTESİ	v
TABLO LİSTESİ	vii
ÖZET	viii
SUMMARY	ix
BÖLÜM 1. GİRİŞ	1
BÖLÜM 2. GIBBS ve MARKOV RASTLANTI ALANLARI	10
2.1. Giriş	10
2.2. Kafes Tabanlı Gibbs ve Markov Rastlantı Alanları	11
2.2.1. Ayrık Gibbs Rastlantı Alanı	12
2.2.2. Ayrık Markov Rastlantı Alanı	13
2.3. Geometrik Komsuluk Kavramı	14
2.4. Sürekli Rastlantı Alanları	17
2.5. Öz-Model (Auto-Model)	17
2.6. Gibbs ve Markov Rastlantı Alan Modelleri	18
2.6.1. Derin & Elliott Modeli	20
2.6.2. 'Auto-Binomial' Model	21
2.6.3. 'Auto-Poisson' Model	22
2.6.4. 'Auto-Normal' Model	22
2.7. Özet	24
BÖLÜM 3. GIBBS ve MARKOV RASTLANTI ALANLARININ ÖRNEKLENMESİ	25
3.1. Giriş	25
3.2. Ayrık GRA ve MRA'ların Örneklenmesi	25
3.3. Parametrelerin Üretilen Doku Üzerine Etkisi	35
3.4. Algoritma 1 ve 2 ile Üretilen İki-Renkli Dokular	35
3.5. Dört Renkli Doku Örnekleri	37
3.6. Özet	44
BÖLÜM 4. İSTATİSTİKSEL İNCELEME ve SONUÇ	45
4.1. Giriş	45
4.2. Parametre Kestirimi (Parameter Estimation)	46
4.2.1. 'Maximum Likelihood Estimation' (MLE)	49
4.2.2. 'Least Square Error Method' (LSQR) ...	54

4.2.3. 'Logit Model Fit Method' (Logit)	56
4.2.4. 'Minimum Logit Chi-Square Method' (Min-Chi-Square)	57
4.2.5. İrdeme	58
4.3. Uygunluk Testi (Goodness-of-Fit-Test) ..	59
4.3.1. Ki-Kare (χ^2) Testi (Chi-Square Test) .	59
4.3.2. Parametrik Olmayan Ranklama Testi (Nonparametric Ranking Test)	62
4.4. Özet	65
BÖLÜM 5. DOKU SINIFLANDIRMA	66
5.1. Giriş	66
5.2. Doku Sınıflandırma Sistemi	67
5.3. Özellik Belirleme ve Çıkarımı	68
5.3.1. SGLDM Oluşturulması ve Türetilen Özellikler	69
5.4. Doku Sınıflandırma Denemeleri	72
5.5. Özet	83
BÖLÜM 6. SONUÇLAR ve ÖNERİLER	84
EK A	86
EK B	88
EK C	89
EK D	94
KAYNAKLAR	155
ÖZGEÇMİŞ	159

SEKİL LİSTESİ

Figure A Example of structural texture	xii
Figure B Example of statistical texture	xii
Şekil 1.1 Yapısal dokuya örnek	5
Şekil 1.2 İstatistiksel dokuya örnek	5
Şekil 2.1 Geometrik komşular	16
Şekil 2.2 Komşuluk sistemlerinin klik tipleri	16
Şekil 3.1 Deneme 3.1 için üretilen dokular	32
Şekil 3.2 a parametresinin yakınsaması	33
Şekil 3.3(a) $b[1,1]$ parametresinin yakınsaması	34
Şekil 3.3(b) $b[1,2]$ parametresinin yakınsaması	34
Şekil 3.4 Parametrelerin doku üzerine etkisi	36
Şekil 3.5 Deneme 3.2 için üretilen dokular	38
Şekil 3.6 Deneme 3.2 için üretilen dokular	39
Şekil 3.7 Deneme 3.2 için üretilen dokular	40
Şekil 3.8 Deneme 3.3 için üretilen dokular	42
Şekil 3.9 Deneme 3.3 için üretilen dokular	42
Şekil 4.1 Birinci dereceden 'coding' bölgeleri	50
Şekil 4.2 İkinci dereceden 'coding' bölgeleri	51
Şekil 4.3 Üçüncü ve dördüncü dereceden 'coding' bölgeleri	51
Şekil 4.4 Deneme 4.2 için üretilen dokular	64
Şekil 5.1 Doku sınıflandırma sistemi	67
Şekil 5.2 Deneme 5.1 için üretilen dokular	75
Şekil 5.3 Deneme 5.1 için iki boyutta izdüşüm	76

Şekil 5.4 Deneme 5.2 için üretilen dokular	77
Şekil 5.5 Deneme 5.2 için iki boyutta izdüşüm	78
Şekil 5.6 Deneme 5.3 için üretilen dokular	79
Şekil 5.7 Deneme 5.3 için iki boyutta izdüşüm	80
Şekil 5.8 Deneme 5.4 için üretilen dokular	81
Şekil 5.9 Deneme 5.4 için iki boyutta izdüşüm	82



TABLO LİSTESİ

Tablo 3.1	Deneme 3.1 için bulunan parametreler	33
Tablo 3.2	Deneme 3.2 için bulunan nokta değişimleri ve aynı renge sahip noktaların sayısı	38
Tablo 3.3	Deneme 3.2 için bulunan parametreler	39
Tablo 3.4	Deneme 3.2 için bulunan parametreler	40
Tablo 3.5	Deneme 3.2 için bulunan parametreler	41
Tablo 3.6	Deneme 3.3 teki Derin & Elliott modeli için bulunan parametreler	43
Tablo 4.1	Deneme 4.1 için Ki-Kare testi örneği	61
Tablo 5.1	Deneme 5.1 için ortalama özellik vektörleri	75
Tablo 5.2	Deneme 5.2 için ortalama özellik vektörleri	77
Tablo 5.3	Deneme 5.3 için ortalama özellik vektörleri	79
Tablo 5.4	Deneme 5.4 için ortalama özellik vektörleri	81

ÖZET

Görüntü analizi, sentezi, yeniden oluşturma (reconstruction), iyileştirme (enhancement), kodlama (coding), sıkıştırma (compression) ve daha bir çok uygulamalarda Markov ve Gibbs rastlantı alanları ve doku (texture) kavramı sık sık kullanılmaktadır. Bu tez çalışmasındaki amaç, Markov ve Gibbs rastlantı alanları üzerine sistematik bir araştırma yapmak, varolan doku üretim algoritmalarını incelemek, bunları karşılaştırmak ve dokuyu tanımlayan özellik vektörlerini kullanarak dokuları sınıflandırmaktır.

Bu tezin bütününde, Markov ve Gibbs rastlantı alanları ve öz-model (auto-model) kavramı eşliğinde Markov ve Gibbs rastlantı alan modelleri anlatılmakta; görüntünün bütünsel özelliklerinin Gibbs dağılımı ile, yerel özelliklerinin de yerel koşullu olasılıklarla ifade edilebilme özelliği kullanılarak örnekleme algoritmaları ile, Gibbs ve Markov rastlantı alan modellerine uygun yapay dokular üretilmekte ve örnekleme algoritmalarından ikisinin karşılaştırılması yapılmaktadır.

Verilen dokuyu modelleyen parametreleri bulmak için kullanılan parametre kestirim (parameter estimation) yöntemleri, özellikle, 'coding' metoduna dayanan 'Maximum Likelihood Estimation' (MLE) yöntemi ayrıntılı olarak açıklanmakta ve bu yöntemle parametre kestirimi yapılmaktadır. Verilen bir doku ile, bir Markov rastlantı alan modeli kullanılarak yeniden üretilen bir doku arasındaki uygunluk ölçüsünü veren uygunluk testleri (goodness-of-fit) incelenmekte ve sonuçlar örneklerle gösterilmektedir.

Tanımlayıcı dokusal özelliklerden enerji, 'entropy', atalet (inertia), yerel homojenlik (local homogeneity) ve ilinti (correlation) değerleri kullanılarak yapay dokuların Mahalanobis uzaklığına göre sınıflandırılması üzerine bir çalışma yapıp özellik vektörlerinin 2-boyutlu izdüşümleri bulunmaktadır. Kullanılan bu yöntemde, parametreleri uygun seçilen dokuların hatasız sınıflandırıldığı, parametreleri birbirine yakın olan dokuların ise hatalı sınıflandığı gözlenmektedir.

TEXTURE SYNTHESIS AND CLASSIFICATION USING MARKOV AND GIBBS RANDOM FIELD MODELS

SUMMARY

In daily life, we have to make decisions by looking at the images in various areas. For example, in medical area, to classify normal and abnormal pulmonary patterns or to identify the illness by using medical images. In an airport, to control the air-traffic with radar signals. In the petroleum industry, to find an underground reservoir of crude oil via aerial photography. These or like these separate events can be collected under a subject named image analysis.

Image analysis is concerned with manipulation and analysis of pattern by computer. Its major subareas include image formation, enhancement, restoration, reconstruction, segmentation, coding and compression, texture analysis, shape analysis, representation, matching, description and recognition. There is no universal way to do image analysis because of both data-dependent and goal-oriented.

In this thesis, we deal with two-dimensional digital image that can be represented by an $N \times N$ matrix, or lattice, whose elements (called pixel) have integral values from 0 up to $G-1$ corresponding to the brightness levels (gray levels). In these digital image, the gray level of a pixel is highly dependent on its geometric neighboring pixels, but is nearly independent of remote pixels.

Markov random fields (MRFs) [3] have been found to be rich models for various areas in image analysis due to the dependency among pixels in spatial neighborhoods. Thus, the classical problems of texture synthesis [7], [13], texture classification [8], image segmentation, image restoration [4], [22], and image compression [14], have been attacked with MRFs.

The goal of this thesis is to make, under the theorem giving that MRFs and GRFs having the same neighborhood system are equivalent, a systematic study on the applications using these fields, and to examine carefully and compare the present sampling and parameter estimation algorithms, and finally, to classify textures by using the feature vectors. We designed a computer program for achieving all of them on personal computer.

In texture modelling using MRFs, a texture is assumed to be a realisation of a stochastic process which is governed by some parameters. In texture synthesis, an algorithm that generates a texture based on a few model parameters is desired. To represent a digital texture by a few model parameters is one goal of data compression. The establishment of an image model does not have a unique rule, but it should be at least based on some property of an image.

Markov random field models assume that the intensity at each pixel in an image depends on its "neighboring pixels" but is independent of other pixels. An image is viewed as a coloring of a lattice. An MRF is a probability space whose probability measure, defined on the sample space consisting of all possible colorings of a lattice, satisfies positivity, the Markov Property, and homogeneity [3]. Markov random fields have been extensively used for modeling images [7], [8], [12], [13], because of the natural coincidence between the local dependency of a model and the local dependency of neighboring pixels. Derin and Kelly [17] published a paper which describes Discrete-Index Markov-Type Random Processes.

The building of an MRF model should follow two principles. The MRF should naturally match the properties of a class of images and should be mathematically tractable. If an MRF generates images whose pixels have integral values, it is discrete, other-wise it is continuous. For example, an auto-binomial MRF [3] is a discrete MRF, and auto-normal MRF [3] is a continuous MRF. The studies in the literature show that discrete Markov random fields are more natural to digital textures but less mathematically tractable than continuous Markov random fields.

Texture is an important characteristic for the analysis of many types of images. It can be seen in all

images from multi-spectral scanner images to microscopic images of cell cultures or tissue samples. Despite a precise definition of texture, it can be defined a structural patterns of surfaces of objects such that wood, sand, grass. In the literature, textures are examined in two categories: structural textures, consisting of textons arranged according to certain placement rules, frequently appear in the studies of human vision. A structural texture is shown in Figure A. Statistical textures that we deal with in this thesis can be viewed as images in which pixels are obtained by some stochastic processes. A statistical texture is shown in Figure B.

Another basic problem in image synthesis is to extract textural features [24]. Textural features can be descriptive or generative. Also, Markov random field model parameters can be thought a feature vector describing the texture [9]. Descriptive textural features characterize some properties of textures corresponding to visual perception like coarseness, contrast, directionality, line-likeness, regularity and roughness [35]. Haralick [24], Weszka [37], and Connors and Harlow [11], survey and compare descriptive textural features. Descriptive textural features can not be used to produce a texture. Generative textural features can be used not only to describe a texture but also to generate a texture.

There are many applications in image processing in which texture synthesis is useful. For example, if a region of a picture is missing or highly corrupted by noise, an artificial texture can be generated to replace the missing data. In image coding applications, if a texture can be generated with a few parameters, we can send these parameters to remote places instead of the whole texture image. Texture synthesis means texture generation by computer. In the literature, many algorithms have been suggested for generating a variety of textures based on:

- 1) Mosaic models [1],
- 2) Fractal models [3],
- 3) Syntactic models [31],
- 4) Time series models [14], [27], [32],
- 5) Markov random field models [6], [12], [25].

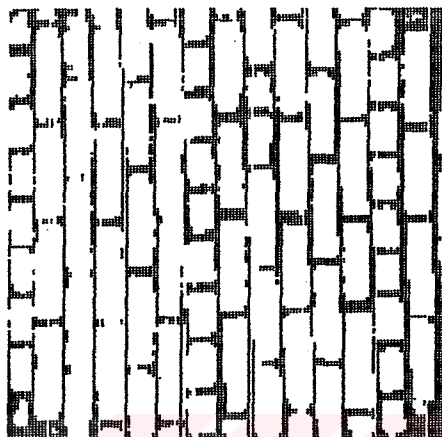


Figure A Example of structural texture

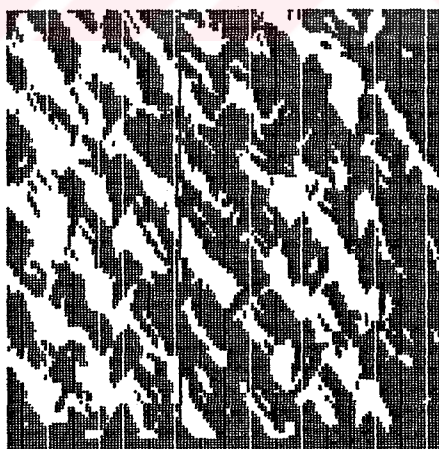


Figure B Example of statistical texture

Model-based texture synthesis involves three problems. First, develop an "efficient" algorithm based on the given model to generate textures. Second, develop a robust estimator for model parameters based on a single texture. Third, develop a statistic to measure the correspondence between the given texture and a model with estimated parameters.

Markov random fields were first proposed as texture models by Hassner and Sklansky [25]. They investigated Besag's auto-logistic MRFs [3] in binary texture synthesis. Cross [12] extended Hassner and Sklansky's work to model natural textures with eight colors, from Brodatz's book [5], by using the auto-binomial model [3]. Chellappa [6] used the auto-normal model (also referred to as a Gaussian Markov random field) and simultaneous autoregressive (SAR) model [3] to model natural textures. Recently, Derin and Elliott [15], [16], extended the Ising model [29] to model textures with more than two colors. In summary, Markov random fields are particularly useful for modelling homogeneous textures whose neighboring pixels have a high dependence but remote pixels are nearly independent.

Texture classification which is a classical problem of image analysis can be stated as follows. Given n_i texture samples (training textures) from texture class i , $1 \leq i \leq K$, extract textural features and establish a decision rule, based on the selected features, which can efficiently classify an unknown texture into one of the K categories.

Although a simple and computationally efficient method is desired, there is no universal way to select textural features for classification and recognition. Textural features can be derived from gray level co-occurrence matrices [23], [34], gray level run lengths [37], gray level differences [37], Fourier power spectra [37], spatial filtering [21], [30], and Markov random field model fitting [8], [9], [18], [26], [28]. Van Gool [36] gives a good survey of texture classification methods. Tamura [35] proposed six textural features, namely, coarseness, contrast, directionality, line-likeness, regularity, and roughness, which correspond to visual perception.

A classifier is a decision rule designed to efficiently assign an unknown texture to one of the known categories based on the extracted textural

features. The 1-NN, quadratic, Fisher's linear classifiers are commonly used in pattern recognition [20]. The design of a classifier should consider the relationship between the number of features and the number of training samples [20]. Recognition rate is a commonly used index to evaluate a classifier in pattern recognition. In texture classification, recognition rate is used not only to evaluate a classifier but also to evaluate the textural features.

The purpose of segmentation which we did not deal with in this thesis is to partition an image into "meaningful" components. The definition of a "meaningful component" is a function of the problem being considered. Segmentation algorithms based on Markov random fields have been investigated and developed [4], [15], [16], [22]. The segmentation problem can be stated as: Given an observed image y , find a coloring x , of pixels in y which optimizes specified criteria. There are two basic approaches:

- 1) Iterative pixel labeling approach : Individually label pixels one by one until some criteria are achieved. The most commonly used criterion is a maximum a posteriori (MAP) probability. The goal is to find an x which maximizes $P(x|y)=P(y|x)P(x)/P(y)$ over all possible x 's [22].
- 2) Region classification approach : An image is partitioned into several regions such that each region is assumed to be a realization of a single Markov random field model with known parameters or a combination of two Markov random fields models. How to partition on image into regions represented by Markov random fields is the most important problem. Derin [15], [16] developed a segmentation algorithm, called a dynamic programming on a strip, to labels pixels.

The final goal of image segmentation is to automatically recognize the regions in an image.

Image restoration is a process which attempts to reconstruct or recover an image that has been degraded by using some knowledge of the degradation phenomenon. It has many fields of applications, including space and biomedical imagery. Knowledge of the degradation phenomenon is crucial to restoration.

In this thesis, Chapter 2 gives a background of Gibbs Random field and Markov random field, Chapter 3 characterizes sampling algorithms for discrete GRFs and MRFs, Chapter 4 explains parameter estimation methods and goodness-of-fit problems, Chapter 5 represents the texture classification and recognition methods, and Chapter 6 summarizes and discusses results.

BÖLÜM 1

GİRİŞ

Günlük yaşamda çeşitli alanlarda elimizde bulunan görüntülere bakarak kararlar vermemiz gerekmektedir. Örneğin, tıbbi alanda, çekilen bir röntgen filmine bakarak hastalığı teşhis etmek; bir hava alanında radar sinyalleri ile hava trafiğini kontrol etmek; havadan çekilen resimleri kullanarak yeraltındaki petrol rezervlerini veya maden kaynaklarını bulmak. Bu ve bunlar gibi birbirinden farklı olayları tek bir konu altında, görüntü analizi (image analysis), toplamak olasıdır.

Görüntü analizi, bilgisayar yardımı ile resimlerin analizi ve onlar üzerinde çeşitli işlemler yapmak olarak düşünülebilir. Görüntü formu oluşturma (image formation), iyileştirme (enhancement), yeniden oluşturma (reconstruction), bölütleme (segmentation), kodlama ve sıkıştırma (coding and compression), doku analizi (texture analysis), şekil analizi (shape analysis), gösterim (representation), eşleme (matching), tanımlama (description) ve tanıma (recognition) görüntü analizinin en önemli alt bölümlerini oluşturmaktadır. Görüntü analizinde veri yapısı ve amaç çok önemli olduğundan görüntü analizinde izlenebilecek standart bir yöntem yoktur.

Bu tez çalışmasında 2-boyutlu, her bir elemanı (pixel), 0 ile G-1 değerleri arasında bir renge (gray level), (parlaklık seviyesi), sahip NxN lik bir kafes (lattice) ile gösterilebilen sayısal görüntülerle ilgilenilmektedir. Bu sayısal görüntülerde, kafes üzerindeki bir noktanın renginin (gray level) onun geometrik komşularından oluşan noktaların renklerine oldukça

bağımlı, diğer noktalara ise bağımlı olmadığı kabul edilmektedir [3], [12].

Uzamsal komşuluklar içindeki noktalar arasındaki bağımlılıktan dolayı, görüntü analizinde değişik sahalara için, Markov rastlantı alanlarının (MRA) çeşitli modelleri bulunmuştur [3]. Böylece, görüntü analizinin klasik problemleri olan doku sentezleme [7], [13] doku sınıflandırma (texture classification) [8], görüntü bölütleme (image segmentation), görüntü yenileme (image restoration) [4], [22] ve görüntü sıkıştırma (image compression) [14] konularında Markov rastlantı alanları ile büyük ilerlemeler kaydedilmiştir.

Bu tez çalışmasındaki amaç, aynı komşuluk sistemine sahip Markov rastlantı alanları ile Gibbs rastlantı alanlarının (GRA) eşdeğer olduğunu veren teorem ışığında, bu alanları kullanarak yapılan uygulamalar üzerine sistematik bir inceleme yapmak, varolan örneklemme algoritmaları ve parametre kestirim (parameter estimation) yöntemlerini karşılaştırmak ve son olarak da, ayrık MRA larında özellik vektörlerini (feature vectors) kullanarak dokuları sınıflandırmaktır. Bütün bu işlemleri kişisel bilgisayarlarla yapabilmek için bu tez çalışması boyunca geliştirilen bir paket program yazıldı (EK C ve EK D). Bu tez çalışmasının bir amacı da böyle bir paket program hazırlamaktır.

MRA lar ile görüntü modellemede, doku bazı parametrelerle ifade edilebilen bir stokastik sürecin gerçekleşmesi olarak düşünülür. Doku sentezlemede, bir kaç parametre ile sayısal bir doku üreten algoritmalar istenir. Sayısal dokuyu birkaç parametre ile gösterebilmek doku sıkıştırmanın da amaçlarından biridir. Görüntü modeli kurmanın temel bir kuralı yoktur. Ancak görüntünün bazı özelliklerinden yola çıkarak model kurulabilir. Markov rastlantı alan modellerinde, görüntü üzerindeki her bir noktanın parlaklığı onun komşu nok-

talarına (neighboring pixels) bağımlı, diğer noktalardan bağımsızdır. Görüntü kafes üzerindeki renklerin bileşiminden oluşan bir renk kümesi olarak düşünülür. Bir Markov rastlantı alanı, bir kafesin tüm olası gerçeklemelerinden oluşan örnek uzayı üzerinde tanımlanan pozitiflik, Markov özelliği ve homojenliği sağlayan olasılık ölçüsüne sahip bir olasılık uzayıdır [3]. Markov rastlantı alanları görüntü modellemeye yaygın olarak kullanılmıştır [7], [8], [12], [13]. Derin & Kelly [17] görüntü analizinde kullanılan Markov tipli rastlantı alanlarının geniş bir özetini yayınlamışlardır.

Bir MRA modeli kurmada iki özellik gözönüne alınır. MRA, bir sınıfa ait görüntünün özellikleri ile eşleşmeli ve matematiksel bir forma sokulabilmelidir. MRA ile üretilen görüntülerde, görüntüyü oluşturan noktaların değerleri tam sayı ise görüntü ayrık (discrete), değilse sürekli (continuous). Örneğin, auto-binomial MRA [3] ayrık bir MRA, auto-normal MRA [3] da bir sürekli MRA'dır. Literatürde yapılan çalışmalar, ayrık MRA'larının sayısal dokulara, sürekli MRA'lardan daha yatkın olduğunu fakat matematiksel bir forma sokmanın sürekli MRA'larından daha zor olduğunu göstermektedir.

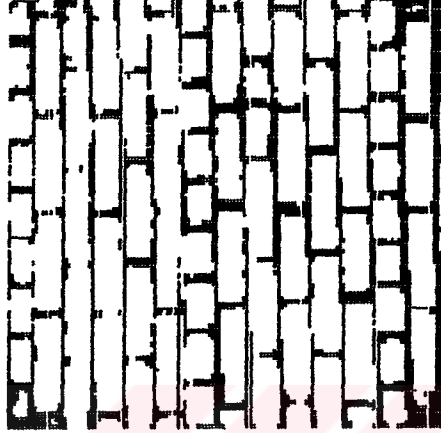
Doku, çeşitli tipteki görüntülerin analizi için önemli bir karakteristiktir. Dokular, 'multi spectral scanner' görüntülerinden, hücre kültürleri ve deri örneklerinin mikroskopik görüntülerine kadar her yerde karşımıza çıkabilir. Dokunun açık bir tanımı olmasa da, yüzeylerin ve objelerin yapısal görüntüleri olarak tanımlanabilir. Örneğin, ağaç kabuğu, kum, çakıl taşı gibi.

Literatürde dokular iki temel kategoride incelenirler: Yapısal (structural) ve istatistiksel (statistical) dokular. Yapısal dokular insan gözüyle kolayca farkedilebilirler. Örneğin, tuğla duvar gibi. Yapısal doku-

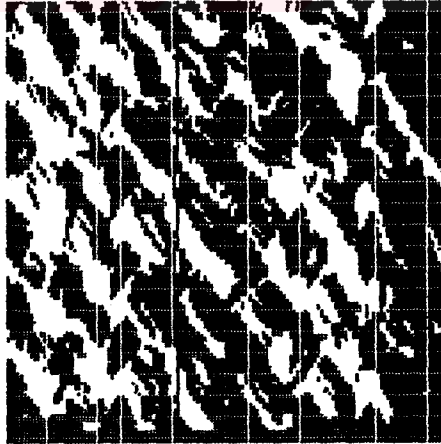
larda, görüntü, kesin çizgilerle ayrılmış bölgelerden oluşur. İstatistiksel dokular ise bazı stokastik işlemler ile elde edilen noktalardan oluşan görüntüler olarak düşünülebilir. Şekil 1.1 de yapısal bir dokuya, Şekil 1.2 de de istatistiksel bir dokuya örnek gösterilmektedir. Bu tez çalışmasında ise sadece istatistiksel dokularla ilgilenilmektedir.

Doku sentezlemedeki bir temel problem de dokusal özelliklerin elde edilmesidir [24]. Dokusal özellikler tanımlayıcı (descriptive) veya üretici (generative) özellikler olmak üzere ikiye ayrılabilir. Ayrıca MRA modelinin parametreleri de dokuyu tanımlayan bir özellik vektörü olarak düşünülebilir [9]. Tanımlayıcı dokusal özellikler, dokuların görüntüsü ile ilgili bazı özellikler ile karakterize edilebilir [35]. Örneğin, iri tanelilik (coarseness), zıtlık (contrast), doğrultululuk (directionality), çizgililik (line-likeness), düzenlilik (regularity) ve pürüzlülük (roughness) gibi. Haralick [24], Weszka [37] ve Connors & Harlow [11] tanımlayıcı dokusal özellikler üzerine bir araştırma yapmışlar ve dokusal özellik çıkarmak için kullanılan çeşitli yöntemleri birbirleriyle karşılaştırmışlardır. Dokusal özellikler bir dokuyu üretmek için kullanılamaz. Üretici dokusal özellikler ise yalnızca dokuyu tanımlamakla kalmayıp onun üretilmesinde de kullanılabilir.

Görüntü analizinde, doku sentezlemenin kullanışlı olduğu bir sürü uygulama vardır. Örneğin, resmin bir bölgesinin kayıp veya herhangi bir gürültü ile bozulmuş olduğunu düşünelim. Bu kayıp veriyi yerine koymak için, bir yapay doku üretebiliriz. Görüntü kodlama uygulamalarında, eğer doku bir kaç parametre ile üretilebiliyorsa, tüm doku yerine sadece bu parametreler kullanılabilir. Örneğin, uzak mesafelere sadece bu parametreleri göndermekle tüm görüntü gönderilmiş olur.



Şekil 1.1 Yapısal dokuya örnek



Şekil 1.2 İstatistiksel dokuya örnek

Doku sentezlemedeki amaç bilgisayar yardımı ile doku üretmektir. Literatürde her türlü doku modeli için bir üretim algoritması önerilmiştir. Doku üretmek ve göstermek için kullanılan doku modelleri şöyle sıralanabilir.

- 1) Mozaik modeller [1],
- 2) 'Fractal' modeller [33],
- 3) 'Syntaktik' modeller [31],
- 4) Zaman serisi modelleri [14], [27], [32],
- 5) Markov rastlantı alan modelleri [12], [6], [25].

Model tabanlı doku sentezlemede üç problemle karşılaşmaktadır. Birincisi, doku üretmek için, verilen modele uygun bir algoritma bulmak. ikincisi, tek bir dokunun model parametrelerinin kestirimi için bir yöntem bulmak. Üçüncüsü, verilen doku ile parametre kestirimi yapılmış bir model arasındaki ilişkinin istatistiksel ölçüsünü bulmak.

Markov rastlantı alanı ilk kez, Hassner & Sklansky [25] tarafından doku modelleri olarak önerildi. Hassner & Sklansky, Besag'ın auto-logistic MRA [3] modelini iki-renkli dokuların sentezinde kullandı. Cross [12], bu çalışmayı, auto-binomial MRA modeli kullanarak, Brodatz'ın kitabından [5] alınan sekiz-renkli doğal dokuları modellemek için genişletti. Chellappa [6] doğal dokuları modellemede auto-normal modeli (literatürde Gaussian Markov rastlantı alanı olarak bilinir) ve 'simultaneous autoregressive (SAR) modelini [3] kullandı. Derin & Elliott [15], [16], Ising modelini [29] iki renkten fazla dokuları modellemek için geliştirdi. Kısaca özetlemek gerekirse, Markov rastlantı alanları, özellikle, komşu noktalara bağımlı homojen dokuların modellenmesinde kullanışlıdır denilebilir.

Görüntü analizinin klasik sorunlarından biri olan doku sınıflandırma problemi aşağıdaki gibi ifade edilebilir. i doku sınıfına ait, $1 \leq i \leq K$, n_i doku örnekleri verilsin, bu örneklerden dokusal özellikler çıkarmak ve bu özelliklere dayanan bir karar kuralı bulmak ki, bu karar kuralı, K sınıftan hangisine ait olduğu bilinmeyen dokuyu doğru bir şekilde sınıflandırabilsin.

Basit ve kolayca hesaplanabilen bir yöntem istenmesine rağmen, sınıflandırma ve görüntü tanımda seçilecek dokusal özellikler için genel bir yol yoktur. Dokusal özellikler 'gray level co-occurrence matrices' [23], [34], 'gray level run lengths' [37], 'gray level differences' [37], 'Fourier power spectra' [37], 'spatial filtering' [21], [30] ve 'Markov random field model fitting' [8], [9], [18], [26], [28] lerden biri kullanılarak kolayca elde edilebilirler. Doku sınıflandırma üzerine Van Gool'un [36] güzel bir çalışması bulunmaktadır. Tamura [35], iri tanelilik, zıtlık, doğrultululuk, çizgililik, düzenlilik ve pürüzlülük olmak üzere görsel algılamaya dayalı altı dokusal özellik önermiştir.

Bir sınıflayıcı, dokusal özellikleri kullanarak, bilinmeyen bir dokuyu, bilinen sınıflardan birine atayan bir karar kuralından başka bir şey değildir. Görüntü tanımda 1-NN, 'quadratic' ve 'Fisher's Linear Classifier' yöntemleri yaygın olarak kullanılmaktadır [20]. Bu tez çalışmasında ise Mahalanobis uzaklığı (Mahalanobis Distance) sınıflayıcı karar kuralı olarak kullanılmaktadır (EK A). Sınıflayıcı, deneme örnekleri ve özellik sayısı arasındaki ilişki gözönüne alınarak tasarlanmalıdır. Literatürde genellikle özellik sayısının dört veya beş katı fazla örnek kullanılır [20]. Sınıflandırma hata olasılığı, sınıflayıcının kullanılabilirlik ölçütü olarak kabul edilir ve doku sınıflandırmada bu değer, dokusal özelliklerin geçerliliğinin bir

Bu tez çalışmasında üzerinde durulmayan bölütlemenin amacı, görüntüyü anlamlı parçalara ayırmaktır. Bu anlamlı parçalara ayırmanın tanımı günümüzde hala üzerinde düşünülen bir problemdir. Literatürde, MRA üzerine bölütleme algoritmaları araştırılmış ve geliştirilmiştir [4], [15], [16], [22]. Bölütleme problemi şu şekilde ifade edilebilir. Verilen bir görüntü, y , olsun, belirlenmiş bir kriterle bu y içindeki noktalardan bir x gerçeklemesi bul. Bunun için temel iki yaklaşım vardır.

- 1) 'Iterative pixel labeling approach' : Bu yöntemde kullanılan kriter bir 'maximum a posteriori' (MAP) olasılığıdır. Amaç, bütün olası x ler içinden $P(x|y)=P(y|x)P(x)/P(y)$ nin değerini maximum yapan x i bulmaktır [22].
- 2) 'Region classification approach' : Bu yöntemde görüntü birkaç bölgeye ayrılır ve her bölge parametreleri bilinen bir tek MRA modelinin bir gerçeklemesi veya iki MRA modelinin birleşimi olarak düşünülür. Derin bu yöntem için bir bölütleme algoritması geliştirmiştir [15], [16].

Bölütlemenin bir başka amacı da, görüntü içindeki bölgeleri otomatik olarak ayırt edebilmektir.

Görüntü yenilemedeki amaç ise, her hangi bir dış etki ile (örneğin, gürültü) bozulmuş görüntüleri, iyileştirmek veya aslına uygun olarak yeniden oluşturmaktır. Genellikle, uygulama alanları uzay ve biyomedikal görüntüler üzerinedir.

Bu tezde, Bölüm 2 de, Gibbs rastlantı alanları ve Markov rastlantı alanları karakterize edilip, GRA ve MRA'nın modelleri tanıtılmaktadır. GRA-MRA eşdeğerliliği bir teoremlle verilmekte ve öz-model kavramı açıklanmaktadır. Bölüm 3 te, MRA ve GRA ları için üç ayrı

Örnekleme algoritması verilmekte, ilk iki algoritmanın iki-renkli dokular için karşılaştırılması yapılmakta ve model parametrelerinin dokular üzerine etkileri incelenmektedir. Bölüm 4 te, parametre kestirimi ve uygunluk testi problemleri incelenmekte; Derin & Elliott ve auto-binomial model için 'Maximum Likelihood Estimation' yöntemi ayrıntılı olarak anlatılmaktadır. Bölüm 5 te, doku sınıflandırma ve tanıma problemleri üzerinde durulmakta ve dokusal özelliklerden enerji, 'entropy', atalet (inertia), yerel homojenlik (local homogeneity), ilinti (correlation) değerleri kullanılarak birbirine benzeyen ve benzemeyen aynı ve farklı modellere ait yapay dokular sınıflandırılmaktadır.

Bu tez çalışmasında kaynakça bölümünde verilen kaynakların birkaçını temin etmek mümkün olamamıştır. Bunların verilmesindeki amaç, konuyla ilgilenecek kişilere kaynak araştırması için yol göstermektir.

BÖLÜM 2

GIBBS VE MARKOV RASTLANTI ALANLARI

2.1. Giriş

Herhangi bir gürültü ile bozulmamış bir görüntü düşünelim, bu görüntü üzerindeki bir noktanın parlaklığının (renginin) onun çevresindeki noktaların parlaklıklarına oldukça bağımlı olacağı rahatça söylenebilir. Bu bölümde, Gibbs Rastlantı Alanları (GRA) ve Markov Rastlantı Alanları (MRA) kullanılarak, bu bağımlılığın açık bir şekilde modellenmesi ve tez boyunca kullanılacak teorik tanımlar açıklanacaktır. GRA ve MRA görüntülerinin modellenmesinde karşımıza iki sorun çıkmaktadır. GRA veya MRA nı örnekleme ve parametre değer biçimi. Bölüm 3' te örnekleme algoritmaları ve yöntemleri verilecek, Bölüm 4' te MRA ve GRA' dan parametre kestirim (=parametre değer çıkarım) yöntemleri anlatılacaktır.

Besag'ın "On Spatial Interaction and The Statistical Analysis of Lattice System" adlı makalesinden sonra MRA nı kullanarak görüntü modellemeye doğru bir eğilim oldu [3]. 1980 yılında Cross, Besag'ın bu makalesinde ilk kez öne sürdüğü 'Auto-Binomial' modelini kullanarak görüntü modellemeye çalıştı [12].

Görüntü modellemede, modellenecek görüntü, $N \times N$ boyutlu bir kafesi (lattice) oluşturan noktaların renk değerlerinin bir kümesi olarak düşünülmektedir. Bu görüntünün bütünsel özellikleri GRA ile, yerel karakteristikleri ise MRA ile uygun bir şekilde modellenebilmektedir.

Besag bu makalesinde Öz-model'leri (auto-model) ilk kez ortaya attığı gibi, Hammersley ve Clifford teoremini kullanarak GRA ile MRA arasındaki eşdeğerliliği de tanıtlamıştır [3]. Bu GRA-MRA eşdeğerliliği tez boyunca kullanılacaktır.

2.2. Kafes Tabanlı Gibbs ve Markov Rastlantı Alanları

Gibbs ve Markov rastlantı alanları ile görüntü modellemeye kullanacağımız kafes, L , $N \times N$ boyutlu noktalardan oluşan bir alandır.

$$L = \{ (i,j) \mid 1 \leq i \leq N, 1 \leq j \leq N \}. \quad (2.1)$$

Bir rastlantı alan modelinde genellikle kafes üzerindeki her bir nokta bir renge (gri seviyesine) karşı gelmekte ve bu renk değerini gösteren $X(i,j)$, bir Rastlantı Değişkeni olarak kabul edilmektedir [3]. $x(i,j)$ ile de bu rastlantı değişkenin aldığı renk değeri gösterilmektedir. Anlatımda kolaylık olması için bu tez çalışmasında aşağıdaki notasyon değişimi uygulanarak $X(i,j)$ rastlantı değişkeni X_t ile gösterilecektir.

$$t = j + N(i-1). \quad (2.2)$$

Eğer görüntü gri seviyelerinden oluşan ayrık (discrete) bir resim ise kafes üzerinde bulunan noktalara renk değerleri atama, sonlu bir küme olan $A=\{0, 1, 2, \dots, G-1\}$ kümesinden yapılmaktadır. Sürekli (continuous) bir resim ise A kümesi renk değerlerinden oluşan gerçel bir çizgiyi göstermektedir. (G en yüksek renk değeridir. iki-renkli (binary) görüntüler için $G=2$ alınır). Bizim çalışmalarımızda tüm renklerin gri seviyelerinden oluştuğu ayrık (discrete) görüntüler ele alınacaktır.

(2.2) deki gösterimle noktalar satır satır birbirinin peşine dizildiği varsayılmakta ve noktalar 1'den N^2 'ye kadar numaralanmaktadır. Yani, $M=N^2$ dersek,

$$t \in S = \{ 1, 2, 3, \dots, M \}. \quad (2.3)$$

$N \times N$ boyutlu her bir noktası bir renk değerine sahip kafesi, Rastlantı Vektörü, $\mathbf{X}=(X_1, X_2, \dots, X_M)$ olarak göstermek olasıdır ve bu rastlantı vektörü bir görüntüyü (gerçeklemeyi) ifade etmektedir.

Matematiksel anlamda bir rastlantı alanı şu üçlüden ibarettir;

$$\langle \Omega, \Phi, P \rangle. \quad (2.4)$$

Ω , kafes üzerindeki noktaların renklerinden oluşan örnek uzayını, Φ bu örnek uzayındaki Borel alt kümesinin tümünü ve P ise Φ domenindeki olasılık değerini göstermektedir. Yani,

$$\begin{aligned} \Omega &= \{(x_1, x_2, \dots, x_M) \mid x_t \in A \text{ bütün } t \in S\}, \\ \Phi &= \{W \mid W \subseteq \Omega\}. \end{aligned} \quad (2.5)$$

2.2.1. Ayrık Gibbs Rastlantı Alanı

Ayrık Gibbs rastlantı alanı, Gibbs dağılımı kullanılarak, aşağıdaki olasılık yoğunluk işlevi ile tanımlanabilir. Bu alan ile görüntüye ait bütünsel özellikleri modellemek olasıdır.

$$P(\mathbf{X}=\mathbf{x}) = \exp(-U(\mathbf{x}))/Z, \quad (2.6)$$

$$Z = \sum_{\mathbf{y} \in \Omega} \exp(-U(\mathbf{y})).$$

Burada, $U(\mathbf{x})$ enerji işlevini, normalize sabiti Z ise bölümlendirme işlevini göstermektedir.

Tanım 2.1 : Eğer $P(X_t=u \mid X_s=v) = P(X_t=u)$ bütün (u,v) ise, kafes üzerindeki s noktası t noktasının bir olasılıksal komşusu değildir. Diğer bir deyişle, eğer $P(X_t=u \mid \text{bütün } X_s=v, t=s) = P(X_t=u \mid X_s=v)$ ise s noktası t noktasının bir olasılıksal komşusudur.

Tanım 2.2 : Klik (clique), kafes üzerindeki bir nokta-
nın olasılıksal komşuları ile oluşturduğu bütün ikili,
üçlü, dördü vs. nokta çiftlerinin kümesidir. Kafes
üzerindeki her bir nokta da birer kliktir.

Tanım 2.3 : Q_i , i noktasını içeren kliklerin kümesi
olsun. Bu durumda sıralanmış Q_i ler kümesi $\{Q_1, Q_2, \dots, Q_M\}$ bir Komşuluk Sistemini (Neighborhood System)
oluşturur. Bir komşuluk sistemindeki tüm kliklerin
kümesine Q diyelim. $V_c(x)$, Q kümesi içindeki kliklerin
işlevi olsun. Bu durumda yukarıda tanımladığımız enerji
işlevi, $U(x)$, klik işlevlerinin toplamı olarak ifade
edilebilir. Klik işlevi, potansiyel işlev olarak da
bilinir ve işlevin aldığı değer, c numaralı kliki
oluşturan noktaların renklerine bağlıdır.

$$U(x) = \sum_{c \in Q} V_c(x). \quad (2.7)$$

2.2.2. Ayırık Markov Rastlantı Alanı

Gibbs rastlantı alanının, kafes üzerindeki tüm
noktaların renklerinden oluşan bir görüntünün bütün
özellikleri ile tanımlandığını belirttik. Markov rast-
lantı alanı ise bu görüntünün sadece yerel özellikleri
ile tanımlanan bir rastlantı alanıdır. Eğer komşuluk
sisteminin geçerli olduğu bir rastlantı alan aşağıdaki
şartları sağlıyorsa, Markov Rastlantı Alanı dır.

- a) **Pozitiflik** : $P(X=x) > 0$ bütün $x \in \Omega$.
- b) **Markov Özelliği** : Bütün $t \in S$ için,

$$P(X_t=x_t \mid X_1=x_1, X_2=x_2, \dots, X_{t-1}=x_{t-1}, X_{t+1}=x_{t+1}, \dots, X_M=x_M) \\ = P(X_t=x_t \mid \text{bütün } X_r=x_r, r \in N_t). \quad (2.3)$$
 N_t , t noktasına ait komşuların kümesini göstermektedir. Ufak bir uyarılama ile,

$$P(X_t=x_t \mid \text{bütün } X_r=x_r, r \in N_t) = P(x_t \mid R_t) \quad (2.9)$$
 şeklinde yazılabilir.
- c) **Homojenlik** : $P(x_t \mid R_t)$ koşullu olasılığı sadece t noktasının değerine bağımlı değil komşu noktaların renk değerlerine de bağlıdır.

Bölüm 2.2.1 ve 2.2.2 den anlaşılabileceği gibi, belirlenmiş klik işlevleri ile bütünsel, belirlenmiş koşullu olasılıklar ile de yerel özellikleri modellemek olasıdır.

Teorem 1 : Belirlenmiş bir komşuluk sisteminde, her MRA için tek bir GRA ve her GRA için de tek bir MRA bulunmaktadır.

Tanıt : Bak. Besag, [3].

Bu teorem, bileşke olasılık değerine $P(X=x)$ (2.6) veya yerel koşullu olasılık değerine $P(x_t \mid R_t)$ (2.8) dayalı bir rastlantı alanı tanımlayabileceğimizi ifade etmektedir. Anlaşılabileceği gibi GRA ve MRA modelleri aynı komşuluk sistemi söz konusu ise birbirlerine eşdeğer olacaklardır. Aksi halde her MRA'nın bir GRA na veya her GRA'nın bir MRA na eşdeğer olduğunu söylememiz olası değildir.

2.3. Geometrik Komşuluk Kavramı

Görüntü analizinde kullanılan genel komşuluk sisteminde, komşuluk derecesi kavramı çok önemlidir. Bu

komşuluk derecesi o noktaya ait komşu noktaların sayısını bildiren bir ölçüttür [3], [12].

Notasyonlarda kolaylık olması için yaptığımız dönüşüm komşu noktaları göstermek için de kullanılacaktır. Şekil 2.1(a) komşuluk derecelerine göre t noktasının geometrik komşularını göstermektedir. Örneğin, t noktasının birinci dereceden komşuları "1" ile ikinci dereceden komşuları "2" ile gösterilmektedir. Şekil 2.1(a) da beşinci dereceye kadar olan komşular görülmektedir. Şekil 2.1(b) de ise, t noktasının geometrik komşuları için beşinci komşuluk derecesine kadar, göreceli bir numaralandırma yapılmıştır. Şekil 2.2 de birinci ve ikinci dereceden komşuluklara ait klik tipleri gösterilmektedir.

Bir Gibbs rastlantı alanında, tek bir noktadan oluşan klik dışında her bir klik tipine bir parametre atanır.

$$\begin{array}{ccccccc} & * & & * & & * & \\ [**, \theta_1]; & [* , \theta_2]; & [* , \theta_3]; & [* , \theta_4]; & & & \\ & ** & & * & & ** & & * & & ** \\ [* , \theta_1]; & [**, \theta_2]; & [* , \theta_3]; & [**, \theta_4]; & [**, \Gamma_1]. \end{array}$$

Bu klik potansiyellerinin değeri, genellikle,

$$V_c(x) = \begin{cases} -\gamma & \text{klik c içindeki tüm noktalar aynı renk ise} \\ \gamma & \text{tersi durumunda} \end{cases}$$

şeklinde tanımlanmaktadır. Tek noktalardan oluşan klik potansiyelinin değeri ise,

$$V_c(x) = \alpha_k \quad X_t = x_t \text{ için,} \quad (2.10)$$

olarak belirtilmiştir [16].

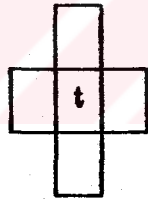
5	4	3	4	5
4	2	1	2	4
3	1	t	1	3
4	2	1	2	4
5	4	3	4	5

(a)

t:-11	t:-7	t:-6	t:+8	t:+12
t:-9	t:-3	t:-2	t:+4	t:+10
t:-5	t:-1	t	t:+1	t:+5
t:-10	t:-4	t:+2	t:+3	t:+9
t:-12	t:-8	t:+6	t:+7	t:+11

(b)

Şekil 2.1 t noktasının geometrik komşularının dereceleri ve yerleri.

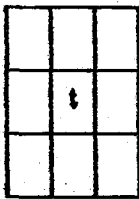


birinci dereceden
komşuluk sistemi

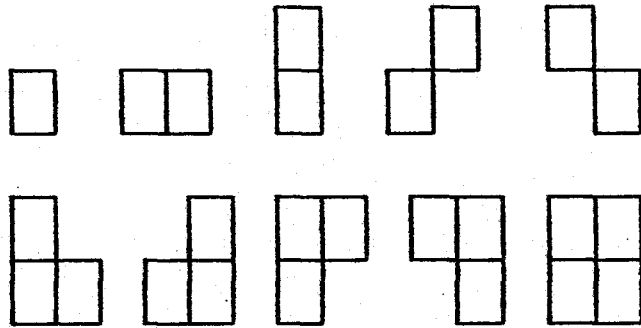


(a)

kık tipleri



ikinci dereceden
komşuluk sistemi



(b)

kık tipleri

Şekil 2.2 Komşuluk sistemleri ve onlara ait kık tipleri

2.4. Sürekli Rastlantı Alanlar

Bir rastlantı alanının olasılık yoğunluk işlevi aşağıdaki gibi ifade edilebiliyorsa, bu bir Sürekli Gibbs Rastlantı Alanıdır.

$$f_x(x) = K \exp(-U(x)) \quad \text{bütün } x \in \Omega. \quad (2.11)$$

Sürekli rastlantı alanlarında, Gaussian rastlantı değişkeni en uygun rastlantı değişkenidir ve bu tür rastlantı alanları görüntü modellemede Gaussian Markov Rastlantı Alanları olarak bilinir.

2.5. Öz-Model (Auto-Model)

M noktadan oluşan ve komşuluk ilişkileri belirli bir alanı stokastik bir süreç olarak ele almak olanaklıdır (Hammersley & Clifford teoremi olarak bilinir). Bu stokastik süreç de bir Markov alanı olacaktır [3]. Hammersley & Clifford teoreminin tanıtında, a) Kafes üzerindeki her bir noktanın sonlu bir renk değerine sahip olduğu ve b) pozitiflik özelliği gözönüne alınarak kafes üzerindeki bütün noktaların değerinin 'sıfır' olduğu bir gerçeğin (görüntünün) bulunduğu kabul edilmektedir. ($P(0) > 0$, 0, bütün noktaları sıfır değerli olan rastlantı vektörünü göstermektedir). Bu iki düşünceden yola çıkarak,

$$Q(X) = \ln\{P(X)/P(0)\} \quad \text{bütün } x \in \Omega \quad (2.12)$$

şeklinde bir $Q(X)$ tanımlanabilir [3].

$Q(X)$ aşağıdaki gibi açık bir şekilde yazılabilir.

$$Q(X) = \sum_{1 \leq i \leq M} x_i F_i(x_i) + \sum_{1 \leq i, j \leq M} x_i x_j F_{i,j}(x_i, x_j) + \sum_{1 \leq i, j, k \leq M} x_i x_j x_k F_{i,j,k}(x_i, x_j, x_k) + \dots + x_1 x_2 \dots x_M F_{1,2,\dots,M}(x_1, x_2, \dots, x_M). \quad (2.13)$$

Tanıt : Bak. Besag, [3].

Bu teorem ışığında (2.13) teki F işlevindeki karmaşıklığı basite indirgemek için (Besag'ın öne sürdüğü 'Öz-Model' lere geçmek için) iki varsayımda bulunulacaktır.

Varsayım 1 : ikiden fazla argümana sahip olan F işlevleri yok olur. (Birli ve ikili klikler dışındaki diğer klik tiplerinin enerjileri sıfır olur).

Varsayım 2 : Kafes üzerindeki her bir noktanın koşullu olasılık dağılımı üsteldir.

Bu varsayımlar ışığında GRA ve MRA öz-model olarak tanımlanır ve $Q(X)$ işlevi,

$$Q(X) = \sum_{1 \leq i \leq M} x_i F_i(x_i) + \sum_{1 \leq i, j \leq M} \beta_{i,j} x_j \quad (2.14)$$

olur.

Bu durumda, (2.12) de verilen eşitlik,

$$P(x_i | \dots) / P(0 | \dots) = \exp[x_i \{F_i(x_i) + \sum_{j=1}^M \beta_{i,j} x_j\}] \quad (2.15)$$

şeklinde yeniden yazılabilir.

$\beta_{i,j}$ modele ait parametreleri göstermektedir ve,

- i) $\beta_{i,j} = \beta_{j,i}$
- ii) $\beta_{i,j} = 0$, eğer i ve j noktaları birbirlerine komşu değilse.

2.6. Gibbs ve Markov Rastlantı Alan Modelleri

Gibbs rastlantı alanlarının potansiyel işlevleri ile (kliklerin enerjileri ile) tamamıyla karakterize edilebildiğini gördük (Bölüm 2.3). Sadece tek nokta ve

nokta çiftlerinden oluşan kliklerle enerji işlevini tanımlamak olasıydı. Öz-model özelliğini kullanarak enerji işlevini yazarsak,

$$U(x) = \sum_{t=1}^M F(x_t) + \sum_{t=1}^M \sum_{r=1}^c H(x_t, x_{t+r}) \quad (2.15)$$

$H(a,b) = H(b,a)$. c değeri komşuluk derecesine bağlı bir değerdir. Örneğin, birinci, ikinci, üçüncü, dördüncü ve beşinci dereceden komşulara göre, sırasıyla $c=2, 4, 6, 10, 12$ 'dır (Şekil 2.1). F işlevi tek noktalardan, H işlevi ise nokta çiftlerinden oluşan kliklerin potansiyel işlevleridir.

Teorem 2 : Enerji işlevi $U(x)$ ile tanımlanmış bir GRA (2.11) aşağıdaki yerel koşullu olasılık değerine sahiptir.

$$P(x_t | R_t) = \frac{\exp(-F(x_t) - \sum_{r=1}^c [H(x_t, x_{t+r}) + H(x_t, x_{t-r})])}{\sum_{s \in A} \exp(-F(x_s) - \sum_{r=1}^c [H(x_s, x_{t+r}) + H(x_s, x_{t-r})])} \quad (2.16)$$

Tanıt : Bak. Chen, [9].

(2.15) ve (2.16) daki eşitliklerde bulunan F ve H işlevlerini değiştirerek çeşitli GRA ve MRA modelleri tanımlamak olasıdır.

GRA ve MRA nın modellere ayrılmasının kuşkusuz ki çeşitli avantajları vardır. Bunları kısaca şöyle açıklayabiliriz.

1) GRA ve MRA nın bir modeline ait örnekleme algoritmaları bir diğer GRA ve MRA örneklemelerine uyarlanabilir.

ii) GRA ve MRA nın bir modeli için kullanılan parametre kestirim algoritmaları bir diğer GRA ve MRA modeline uyarlanabilir.

iii) Görünüşte doğal dokulara benzer yapay dokular GRA ve MRA nın bu modelleri kullanılarak üretilebilir.

iv) Örnekleme ve parametre kestirim yazılımları, doku sınıflamada, görüntü onarımında (restoration) ve görüntü bölütlemeye (segmentation) doğrudan veya kolayca uyarlama yapılarak kullanılabilir.

2.6.1. Derin & Elliott Modeli

$$A = \{0, 1, \dots, G-1\}$$

$$F(x_t) = \alpha x_t,$$

$$H(x_t, x_{t+r}) = \theta_r I(x_t, x_{t+r}).$$

$$I(a, b) = -1 \quad \text{eğer } a = b$$

$$I(a, b) = 1 \quad \text{eğer } a \neq b.$$

θ_r , modele ait parametreleri göstermektedir. Bu tez için yapılan bilgisayar programlarında bu parametreler ile aşağıdaki parametreler eşdeğer tutulmaktadır.

$$\alpha x_t = a,$$

$$\theta_1 = b[1,1], \quad \theta_3 = b[1,2], \quad \theta_5 = b[3,1],$$

$$\theta_2 = b[2,1], \quad \theta_4 = b[2,2], \quad \theta_6 = b[3,2].$$

Derin & Elliott modeli bir öz-model değildir. Bu model, ilk ortaya atıldıktan sonra [22] doku sentezleme ve bölütlemeye sık sık kullanılmıştır [4], [16]. Bu model, $G=2$ seçildiğinde Ising modeline eşdeğer olmaktadır [29].

Eğer $a_0=a_1, \dots, a_{G-1}$ ise, olasılık yoğunluk işlevi, renklerin herhangi bir dağılımında değişmez ve, $P(X_t=g) = 1/G, \quad 0 \leq g \leq G-1,$ olur.

Tanıt : Bak. Chen [9].

Yerel koşullu olasılık değerleri ile GRA,

$$P(x_t|R_t) = \frac{\exp(-\alpha x_t - \sum_{r=1}^c \theta_r [I(x_t, x_{t:+r}) + H(x_t, x_{t:-r})])}{\sum_{s \in A} \exp(-\alpha s - \sum_{r=1}^c \theta_r [I(s, x_{t:+r}) + I(s, x_{t:-r})])} \quad (2.17)$$

biçiminde yazılabilir.

2.6.2. 'Auto-Binomial' Model

$$A = \{0, 1, \dots, G-1\}$$

$$F(x_t) = \alpha x_t - \ln[\text{Conb}(G-1, x_t)],$$

$$H(x_t, x_{t:+r}) = \theta_r x_t x_{t:+r},$$

$$\text{Conb}(m, n) = \frac{m!}{(m-n)!n!}.$$

$G=2$ alındığında, bu model 'auto-logistic' modele karşı gelmektedir [3]. Bu model, Besag tarafından ilk kez ortaya atıldıktan sonra yapay (synthetic) ve doğal dokuların modellenmesinde başarıyla kullanılmıştır [12].

Yerel koşullu olasılık değerleri ile GRA,

$$T = \alpha + \sum \theta_r (x_{t:+r} + x_{t:-r}),$$

$$p = \exp(-T)/(1+\exp(-T)),$$

$$P(x_t|R_t) = \text{Conb}(G-1, x_t) p^{x_t} (1-p)^{G-1-x_t}. \quad (2.18)$$

Auto-binomial MRA modeli için [3],

$$T = \alpha + \sum \theta_r (x_{t:+r} + x_{t:-r}),$$

$$p = \exp(T)/(1+\exp(T)),$$

$$P(x_t|R_t) = \text{Conb}(G-1, x_t) p^{x_t} (1-p)^{G-1-x_t}. \quad (2.19)$$

2.6.3. 'Auto-Poisson' Model

$$\begin{aligned} A &= \{0, 1, \dots, \infty\} \\ F(x_t) &= \alpha x_t - \ln(x_t!), \\ H(x_t, x_{t:+r}) &= \theta_r x_t x_{t:+r}. \end{aligned}$$

Bu model ilk kez Besag [3] tarafından önerilmiş, fakat renk değerleri kümesi sonsuz elemanlı olduğundan ayrık görüntü analizinde henüz kullanılmamıştır.

Yerel koşullu olasılık değerleri ile GRA,

$$P(x_t | R_t) = \exp(-k) k^{x_t} / x_t!, \quad (2.20)$$

$$k = \exp\left(-\alpha - \sum_{r=1}^c \theta_r (x_{t:+r} + x_{t:-r})\right).$$

2.6.4. 'Auto-Normal' Model

'Auto-normal' model, enerji işlevi aşağıdaki gibi karakterize edilen bir sürekli GRA dır.

$$F(x_t) = \frac{(x_t - \mu_t)^2}{2\sigma^2},$$

$$H(x_t, x_{t:+r}) = -\theta_r \frac{(x_t - \mu_t)(x_{t:+r} - \mu_{t:+r})}{\sigma^2}.$$

Bu model için A kümesi, gri seviyelerinden oluşan gerçel bir çizgidir. Bu model ilk kez Besag tarafından, bitki ekolojisinde, ekin ürünlerinin modellenmesi için önerilmiştir [3]. Bu model doğal dokuların [8] ve yapay radar görüntülerinin modellenmesinde de kullanılmıştır.

'Auto-normal' modellemede $\mathbf{X}$ rastlantı değişkenlerinin dağılımı, ilinti (correlation) matrisi $\mathbf{B}^{-1}$ olarak tanımlanmış çok değişkenli bir normal dağılımdır [3].

Bu modele ilişkin olasılık yoğunluk işlevi aşağıdaki gibidir.

$$f_{\mathbf{x}}(\mathbf{x}) = \frac{|\mathbf{B}|^{\frac{1}{2}}}{(2\pi\sigma^2)^{N/2}} \exp\left[-\frac{(\mathbf{x}-\mu)^T \mathbf{B}(\mathbf{x}-\mu)}{2\sigma^2}\right] \quad (2.21)$$

$\mathbf{B}^{-1}$ matrisi $N \times N$ boyutlu blok-çevrimsel bir matristir ve N^2 tane blok da kendi içlerinde çevrimseldir. ilinti matrisinin tersi aşağıdaki gibi tanımlanır.

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \dots & \mathbf{B}_{1N} \\ \mathbf{B}_{1N} & \mathbf{B}_{11} & \dots & \mathbf{B}_{1N-1} \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}_{12} & \mathbf{B}_{13} & \dots & \mathbf{B}_{11} \end{bmatrix} \quad (2.22)$$

Birinci dereceden komşuluk için,

$\mathbf{B}_{11} = \text{çevrimsel}(1, -\theta_1, 0, \dots, 0, -\theta_1),$

$\mathbf{B}_{12} = \text{çevrimsel}(-\theta_2, 0, 0, \dots, 0, 0),$

$\mathbf{B}_{1N} = \text{çevrimsel}(-\theta_2, 0, 0, \dots, 0, 0),$

$\mathbf{B}_{1j} = 0, \quad 2 < j < N.$

ikinci dereceden komşuluk için,

$\mathbf{B}_{11} = \text{çevrimsel}(1, -\theta_1, 0, \dots, 0, -\theta_1),$

$\mathbf{B}_{12} = \text{çevrimsel}(-\theta_2, -\theta_3, 0, \dots, 0, -\theta_4),$

$\mathbf{B}_{1N} = \text{çevrimsel}(-\theta_2, -\theta_4, 0, \dots, 0, -\theta_3),$

$\mathbf{B}_{1j} = 0, \quad 2 < j < N.$

Bu modele en uygun rastlantı değişkeni Gaussian rastlantı değişkeni olduğundan bu model literatürde Gaussian Markov Rastlantı Alanı olarak bilinir. Yerel koşullu olasılık değeri aşağıdaki şekilde yazılır [3].

$$P(x_t | R_t) = (2\pi\sigma^2)^{-\frac{1}{2}} \exp \left[-\frac{[x_t - \mu_t - \sum_{r=1}^c \theta_r ((x_{t-r} - \mu_{t-r}) + (x_{t-r} - \mu_{t-r}))]^2}{2\sigma^2} \right].$$

2.7. Özet

Bu bölümde, klikler tanıtıldı ve bu klikler üzerinden tanımlanan potansiyel işlevlerinin toplamı olarak enerji işlevi, $U(x)$, hesaplandı. Geometrik anlamda komşuluk ve öz-model kavramı açıklandı. GRA ve MRA modelleri incelendi.

Sonuç olarak, enerji işlevi ile karakterize edilen ayrık Gibbs rastlantı alanının olasılık yoğunluk işlevinde bulunan Z sabitinin hesaplanmasının zaman alması ve zor olması nedeniyle GRA'nın örneklenmesinin istatistiksel olarak kolay olmadığı, yerel koşullu olasılığa dayanarak ifade edilen MRA'nın istatistiksel olarak daha kullanışlı olduğu söylenebilir.

BÖLÜM 3

GİBBS ve MARKOV RASTLANTI ALANLARININ ÖRNEKLENMESİ

3.1. Giriş

GRA veya MRA modelleri kullanarak doku sentezinde yapılan, bu alanlara uygun bir olasılık dağılımını örneklemektir. Bu bölümde 3 ayrı örnekleme algoritması verilecektir. (Bunlardan sadece ilk ikisi bu tezde kullanılmıştır). Birinci algoritma, Cross'un kullandığı değiş-tokuş algoritması [3]; ikinci algoritma, Chen'in kullandığı C Algoritması [9]; üçüncü algoritma da Gibbs Sampler olarak bilinen [22] örnekleme algoritmasıdır. Birinci ve ikinci algoritmalarda G^M tane olası gerçeklemenin söz konusu olduğu bir Markov Zinciri simule edilmektedir.

3.2. Ayırık GRA ve MRA'larının Örneklenmesi

Bir rastlantı alanının örneklenmesinde bize olası bir gerçeklemenin olasılık değeri gerekmektedir. Yani, $P(X=x)$ değeri tam olarak hesaplanabilmelidir. Pratikte bunu hesaplamak olanaksızdır (olasılık yoğunluk işlevinde bulunan Z sabitinin hesaplanmasının zaman alması ve zor olması nedeniyle). Bundan dolayı örnekleme algoritmalarında genellikle iki ayrı gerçeklemenin oranı gözönüne alınır. Yani,

$P(X=x)$ birinci gerçeklemeyi,

$P(Y=y)$ ikinci gerçeklemeyi gösterebilirsin,

bu durumda bir GRA veya MRA rastlantı alanını örneklemek için,

$r=P(X=x)/P(Y=y)$ oranını bilmek yeterli olacaktır.

Bu oranı hesaplamak çok kolaydır. Olasılık yoğunluk işlevlerinin (Bölüm 2.3) paydalarında bulunan Z normalize sabitleri bölüm dolayısıyla yok olacak, geriye sadece iki üstel işlevin birbirine oranı kalacaktır. Eğer iki gerçekleştirme arasında sadece bir noktada değişiklik yapılmışsa, bu oran tek bir üstel işleve dönüşecek ve üs bölümünde de sadece farklı olan bu iki noktaya ait klik potansiyelleri bulunacaktır.

Tanıt :

$P(\mathbf{X})$ birinci gerçekleştirme,

$P(\mathbf{Y})$ ikinci gerçekleştirme olsun,

birinci ve ikinci gerçekleştirme arasında tek bir nokta farklı olsun,

$$P(\mathbf{X}) = P(x_1, x_2, x_3, \dots, x_M) = \exp[-U(\mathbf{x})] / Z, \quad (3.1)$$

$$P(\mathbf{Y}) = P(x_1, y_2, x_3, \dots, x_M) = \exp[-U(\mathbf{y})] / Z. \quad (3.2)$$

Bu durumda $P(\mathbf{Y})/P(\mathbf{X})$ oranını yazarsak,

$$\frac{P(\mathbf{Y})}{P(\mathbf{X})} = \frac{\exp(-U(\mathbf{y}))}{\exp(-U(\mathbf{x}))}. \quad (3.3)$$

Enerji işlevleri açık bir şekilde yazılırsa,

$$U(\mathbf{x}) = - \sum_t^M F(x_t) - \sum_t^M \sum_r^c \theta_r I(x_t, x_{t+r}), \quad (3.4.a)$$

$$U(\mathbf{y}) = - \sum_t^M F(y_t) - \sum_t^M \sum_r^c \theta_r I(y_t, y_{t+r}), \quad (3.4.b)$$

olur. Tek bir nokta dışındaki tüm nokta değerleri her iki gerçekleştirmede de aynı olduğundan, bu oran,

$$\frac{P(Y)}{P(X)} = \exp \left[-F(y_2) - \sum_r^c \theta_r I(y_2, y_2 : +r) + F(x_3) + \sum_r^c \theta_r I(y_2, y_2 : +r) \right] \quad (3.5)$$

olur. Bu durumda hesaplama tüm noktalar üzerinden değilde sadece ilgili nokta ve onun komşularına indirgenmiş olmaktadır. Diğer GRA modelleri için de aynı durum geçerlidir. Örneğin, auto-binomial modelde,

$$\frac{P(Y)}{P(X)} = \frac{\text{Conb}(G-1, y_3)}{\text{Conb}(G-1, y_3)} \exp(R_3(y_3 - x_3)), \quad (3.6)$$

$$R_3 = \alpha + \sum_r^c \theta_r [x_3 : +r + x_3 : -r]$$

şeklinde olacaktır.

Aynı şekilde yerel koşullu olasılıklarla tanımlanan MRA için de bu oranı basit bir şekilde hesaplamak olasıdır. MRA larında elde edilen iki gerçekelemenin olasılık yoğunluk işlevlerinin oranı o gerçekelemelere ait koşullu olasılıklar biçiminde yazılabilir [3].

$$\frac{P(X)}{P(Y)} = \prod_{i=1}^M \frac{P(x_i | x_1, \dots, x_{i-1}, y_{i+1}, \dots, y_M)}{P(y_i | x_1, \dots, x_{i-1}, y_{i+1}, \dots, y_M)}. \quad (3.7)$$

Tanıt : Bak. Besag, [3].

Yukarıda olduğu gibi burada da iki gerçekeleme arasında bir nokta farklı ise,

$$\begin{aligned} P(X) &= P(x_1, x_2, x_3, \dots, x_M) \\ &= P(x_2 | x_1, x_3, \dots, x_M) \cdot P(x_1, x_3, \dots, x_M), \end{aligned}$$

$$\begin{aligned} P(Y) &= P(x_1, y_2, x_3, \dots, x_M) \\ &= P(y_2 | x_1, x_3, \dots, x_M) \cdot P(x_1, x_3, \dots, x_M), \end{aligned}$$

$$\frac{P(Y)}{P(X)} = \frac{p(y_2 | x_1, x_3, \dots, x_M) \cdot P(x_1, x_3, \dots, x_M)}{p(x_2 | x_1, x_3, \dots, x_M) \cdot P(x_1, x_3, \dots, x_M)},$$

$$\frac{P(Y)}{P(X)} = \frac{p(y_2 | x_1, x_3, \dots, x_M)}{p(x_2 | x_1, x_3, \dots, x_M)},$$

Komşuluk sistemi söz konusu olduğunda,

$$\frac{P(Y)}{P(X)} = \frac{p(y_2 | R_3)}{p(x_2 | R_3)} \quad (3.8)$$

Birinci algoritmada bu oran hesaplanırken iki gerçekteleme arasında sadece iki tane farklı noktanın olduğu düşünülmektedir. Bu algoritmada ikinci gerçekteleme, görüntü üzerinde aynı renk değerine sahip olmayan iki noktanın seçilip, kendi aralarında yerlerinin değiştirilmesi ile oluşturulur.

Algoritma 1

```
/* Ayrık MRA örnekleme */
/* Girişler, MRA na ait parametreler */
/* Sonuç, MRA nın bir gerçektelemesi */
(1) NxN boyutlu kafes üzerindeki her noktaya
    A={0,1,2,...,G-1} kümesinden rastgele seçilen
    bir renk ata. Bu görüntü örnekleme
    algoritmamız için başlangıç durumu olacaktır.
(2) Kafesteki ilk noktadan başlayarak M. noktaya
    kadar, (For s=1 to M Do)
    a) Kafes üzerinde xs<>yt noktasını belirle.
    b) r=P(Y)/P(X) oranını hesapla ve p=min{1,r}.
    c) p olasılıkla xs ile yt nin yerlerini
        değiştir.
(3) Bu işlemi Kaşngu kez tekrarla.
```

Algoritma 2

```

/* Ayrık MRA veya GRA örnekleme */
/* Girişler, MRA veya GRA na ait parametreler */
/* Sonuç, MRA veya GRA nın bir gerçeklemesi */
(1) NxN boyutlu kafes üzerindeki her noktaya
    A={0,1,2,...,G-1} kümesinden rastgele seçilen
    bir renk ata. Bu görüntü örnekleme
    algoritmamız için başlangıç durumu olacaktır.
(2) Kafesteki ilk noktadan başlayarak M. noktaya
    kadar, (For s=1 to M Do)
    a) Bütün t = s için  $y_t = x_t$ . A kümesinden
        rastgele bir g rengi seç ( $g \in A$ );  $y_s = g$ .
    b)  $r = P(Y)/P(X)$  oranını hesapla ve  $p = \min\{1, r\}$ .
    c) p olasılıkla  $X \leftarrow Y$ .
(3) Bu işlemi Kaşgu kez tekrarla.

```

Algoritma 3

```

/* Ayrık MRA veya GRA örnekleme */
/* Girişler, MRA veya GRA na ait parametreler */
/* Sonuç, MRA veya GRA nın bir gerçeklemesi */
(1) NxN boyutlu kafes üzerindeki her noktaya
    A={0,1,2,...,G-1} kümesinden rastgele seçilen
    bir renk ata. Bu görüntü örnekleme
    algoritmamız için başlangıç durumu olacaktır.
(2) Kafesteki ilk noktadan başlayarak M. noktaya
    kadar, (For s=1 to M Do)
    a)  $g=0,1,2,...,G-1$  için  $\{p_g\}$  olasılıklarını
        hesapla.  $p_g = P(x_s = g | R_s)$ .
    b)  $p_g$  olasılıkla  $x_s \leftarrow g$ .
(3) Bu işlemi Kaşgu kez tekrarla.

```

Bu üç algoritmayı kısaca karşılaştıralım. Algoritma 1 de MRA nın pozitiflik koşulu gözardı edilmektedir. Döngü sayıları sabit tutulduğunda ikinci algoritma en hızlısıdır. Üçüncü algoritma da hesaplanacak üstel

terimler fazla olduğundan en yavaşıdır. Örneklem algoritmalarını etkileyen faktörleri şöyle sıralayabiliriz:

- i) Başlangıç durumu,
- ii) Parametreler,
- iii) Renk sayısı,
- iv) Görüntünün boyutları,
- v) Döngü sayısı.

Yukarıda verilen örneklem algoritmaları hangi başlangıç durumundan başlarsa başlasın sonuçta bir gerçeklemeye yakınsar. Başlangıç durumu sadece yakınsama süresini etkiler. Döngü sayısının, Kaşgu, kesin bir değeri yoktur. Derin & Elliott modeli için yapılan denemelerde [9] en uygun sayı 50 olarak bulunmuştur. Algoritma 1 için ise, (auto-binomial modelde) bu sayının 10 olması yeterlidir [12].

Döngü sayısını bulmanın bir yolu da, doku örneklerken her döngüden sonra parametre kestirimi yapıp gerçek parametreler ile karşılaştırmaktır. Yani, parametrelerin yakınsamasını incelemektir. Eğer parametre kestiriminde elde edilen parametreler gerçek parametrelerin %5-10 civarında ise bu döngü sayısı yeterlidir denilebilir [12]. Bu tez çalışmasında birinci ve ikinci algoritmalar ve bu döngü sayıları kullanılmıştır.

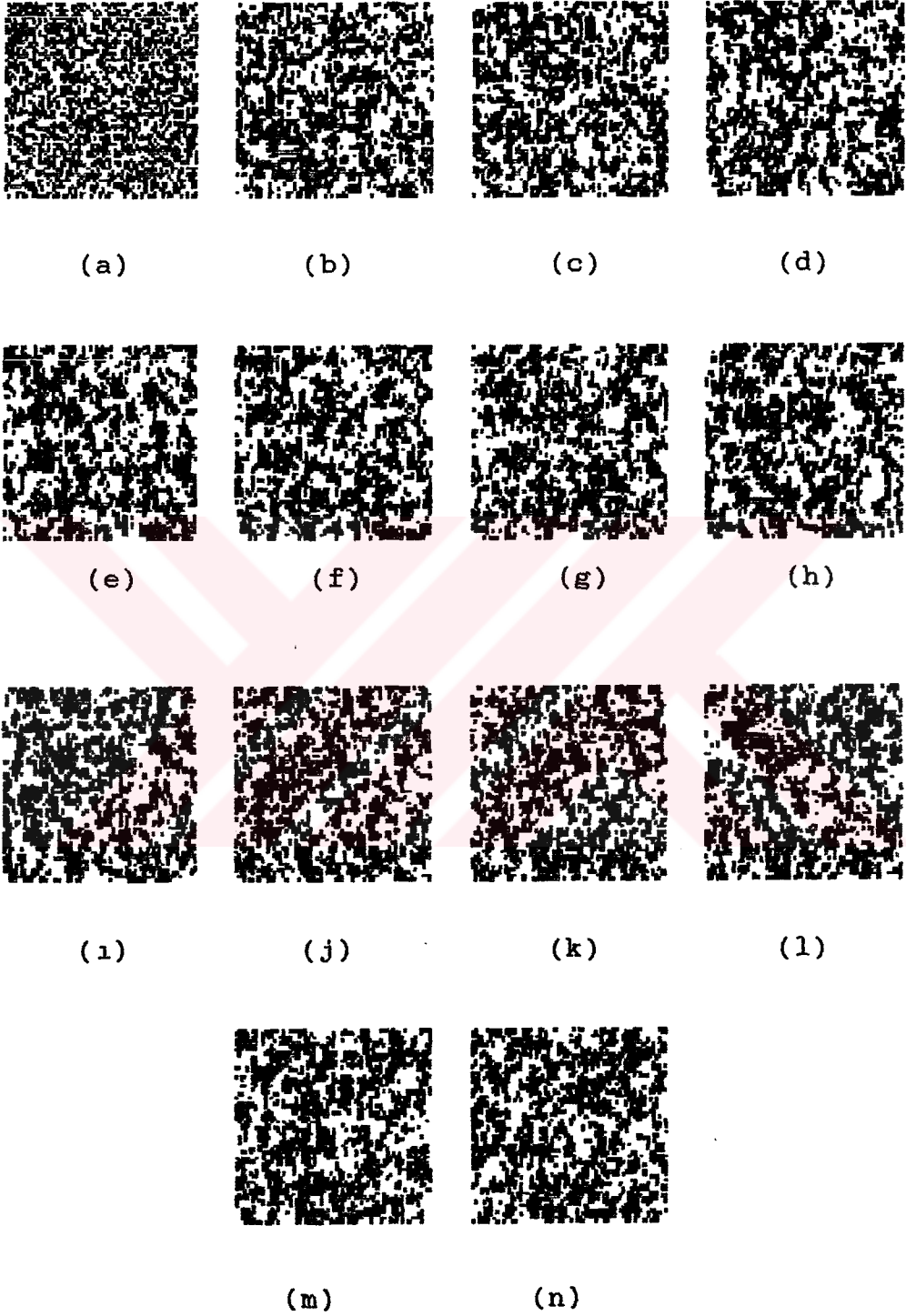
Örneklem algoritmalarının geçerliliğini, birinci algoritma için Cross, bir döngüdeki nokta değişimlerinin oranını kullanarak [12], ikinci ve üçüncü algoritmalar için ise Chen [9] incelemiştir. Chen geçerlilik testi, K döngü sonucu elde edilen gerçeklemeye ait, aşağıdaki şıklarda belirtilen değerlerden birinin ortalaması ve standart sapmasının birbirlerine oranının belirli bir değerden küçük olup olmadığını, $|\sigma/\mu| < \delta$, kontrol ederek yapmaktadır [9].

- 1) Her döngüdeki nokta değişimlerinin sayısı $\{K, K+1, \dots, K+h-1\}$.
- 2) Her döngüdeki enerji miktarı $\{K, K+1, \dots, K+h-1\}$.

Yani, $|\sigma_1/\mu_1| < \delta$ veya $|\sigma_2/\mu_2| < \delta$ ise örnekleme algoritması geçerlidir. Bu test için $K=50$, $h=10$ ve $\delta=10\%$ olarak alınabilir.

Deneme 3.1 : Bu denemede, birinci dereceden, auto-binomial ve $a=-2$, $b[1,1]=1$, $b[1,2]=1.5$ parametrelerine sahip bir doku üretip parametrelerin yakınsaması incelenmektedir. Şekil 3.1 başlangıç durumu ve her döngü sonunda elde edilen doku gerçeklemelerini, Tablo 3.1 her döngü sonunda örneklenen dokudan çıkarılan parametreleri, Şekil 3.2 de bu parametrelerin döngü sayısına bağlı olarak değişimini göstermektedir.

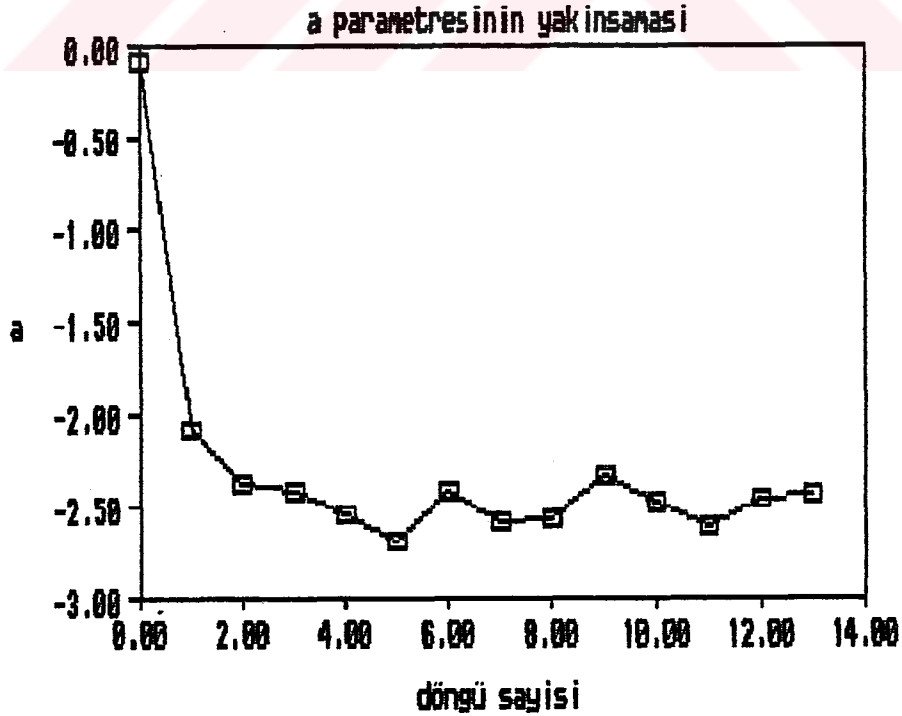
Sonuç olarak, grafikten de görüleceği gibi doku örneklerken, model parametreleri belirli bir döngüden sonra gerçek değerler etrafında salınmaktadır.



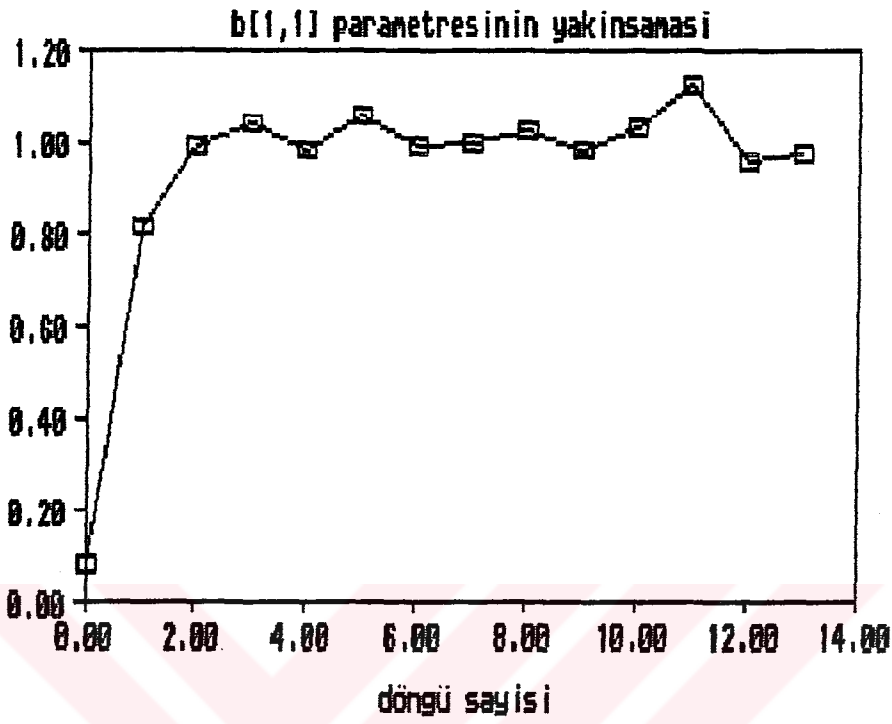
Şekil 3.1 (a) Başlangıç, (b) ... (n) 1. döngüden 13. döngüye kadar dokunun aldığı formlar.

Tablo 3.1 $a=-2.5$, $b[1,1]=1.0$, $b[1,2]=1.5$ olan auto-binomial bir dokunun her döngü sonunda MLE ile bulunan parametreleri.

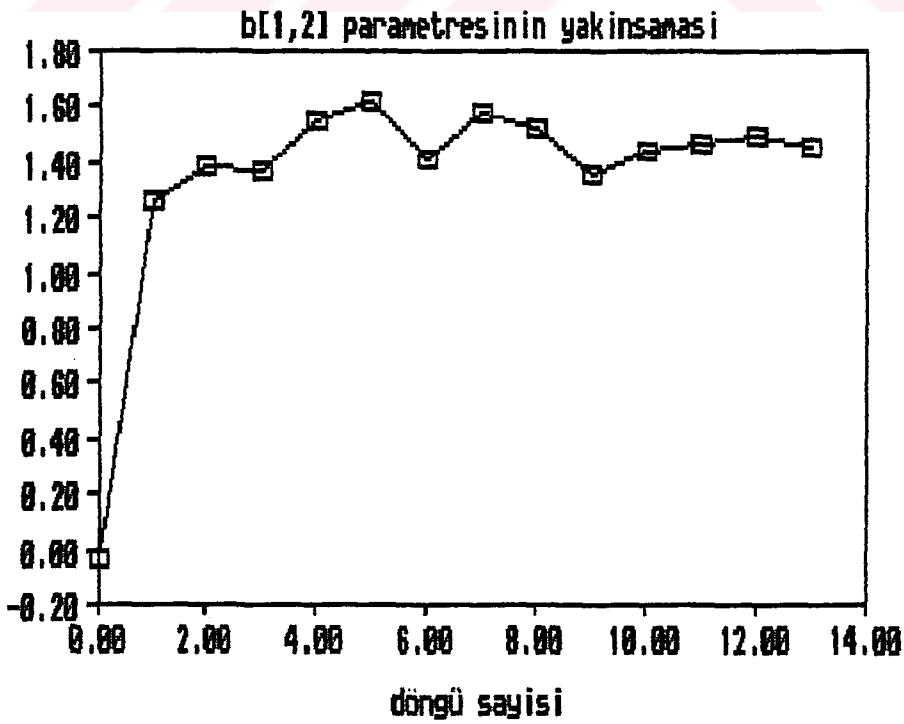
Döngü Sayısı	a	b[1,1]	b[1,2]
Başlangıç	-0.0777	0.0832	-0.0330
1	-2.0747	0.8136	1.2557
2	-2.3804	0.9929	1.3863
3	-2.4208	1.0437	1.3723
4	-2.5352	0.9875	1.5552
5	-2.6805	1.0625	1.6144
6	-2.4131	0.9888	1.4090
7	-2.5765	1.0006	1.5709
8	-2.5577	1.0289	1.5195
9	-2.3419	0.9806	1.3578
10	-2.4750	1.0349	1.4399
11	-2.5990	1.1251	1.4713
12	-2.4552	0.9613	1.4886
13	-2.4347	0.9711	1.4568



Şekil 3.2 a parametresinin yakınsaması



Şekil 3.3.(a) b[1,1] parametresinin yakinsanasi



Şekil 3.3.(b) b[1,2] parametresinin yakinsanasi

3.3. Parametrelerin Üretilen Doku Üzerine Etkisi

Yapay doku üretilirken kullanılan parametrelerin herbiri dokunun görüntüsünü farklı biçimlerde etkiler. θ_1 ($b[1,1]$), yatay yönde kümelenmeyi; θ_2 ($b[1,2]$), dikey yönde kümelenmeyi; θ_3 ($b[2,1]$), 45° lik yönde kümelenmeyi; θ_4 ($b[2,2]$), 135° lik yönde kümelenmeyi kontrol etmektedir. Bu parametrelerin pozitif ve negatif olmaları da dokunun görüntüsünü etkilemektedir. Örneğin pozitif değerler çekme (attraction), negatif değerler ise itme (repulsion) veya dama tahtası (checker-board) etkisine neden olurlar.

Tanım 3 : Eğer $b[1,1]=b[1,2]$ ise, bu MRA veya GRA i. dereceden yönsemezdir (isotropic). Tersî durumda ise yönserdir (anisotropic). ($b[i,j]$ parametrelerindeki en büyük i değeri o alanın derecesini gösterir).

Şekil 3.4 te parametreleri (a) $a=-5$, $b[1,1]=2.5$, $b[1,2]=2.5$; (b) $a=0.2$, $b[1,1]=2.1$, $b[1,2]=2$, $b[2,1]=0.1$, $b[2,2]=0.02$; (c) $a=-2$, $b[1,1]=1.95$, $b[1,2]=0.1$, $b[2,1]=0.05$, $b[2,2]=0.02$; (d) $a=-2$, $b[1,1]=-0.15$, $b[1,2]=0.15$, $b[2,1]=1.95$, $b[2,2]=-0.08$ olan iki-renkli auto-binomial modeline göre üretilmiş dokular (ikinci algoritma kullanılarak) gösterilmektedir.

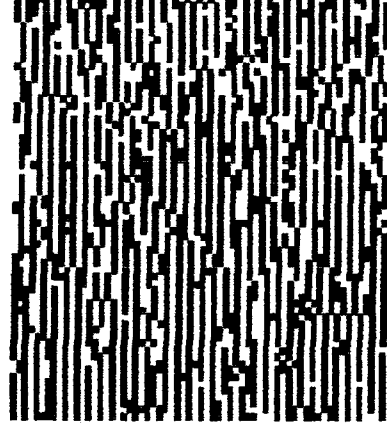
3.4. Algoritma 1 ve 2 ile Üretilen İki-Renkli Dokular

Birinci algoritma (değiş-tokuş yöntemi) ile MRA örneklerken pozitiflik koşulu gözardı ediliyordu. İkinci algoritma ile bu olay ortadan kaldırılmış görünmektedir. Bu bölümde aynı parametreler kullanılarak, bu iki algoritma ile doku üretilip karşılaştırılması yapılacaktır.

Deneme 3.2 : Birinci dereceden, parametreleri $a=-3.2$, $b[1,1]=1.8$, $b[1,2]=1.8$ olan auto-binomial modeline



(a)



(b)



(c)



(d)

Şekil 3.4 Parametrelerin doku üzerine etkisi

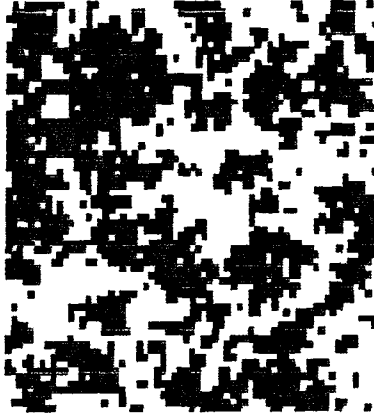
uygun bir doku örneğimiz olsun. Şekil 3.5 te 1. ve 2. algoritmalar kullanarak üretilen bu dokular, Tablo 3.3 de MLE yöntemi ile bu dokulardan çıkarılan parametre değerleri verilmektedir. Şekil 3.6 parametreleri $a=4.6$, $b[1,1]=-2.2$, $b[1,2]=2.1$ birinci dereceden, Şekil 3.7 parametreleri $a=0.15$, $b[1,1]=2.05$, $b[1,2]=2$, $b[2,1]=-2.1$, $b[2,2]=-2$ olan bir dokuyu, Tablo 3.4 ve 3.5 bu dokulardan MLE yöntemi ile çıkarılan parametreleri göstermektedir. Tablo 3.2 ise her iki algoritma ile doku örneklerken dokuyu oluşturan olası renklerin sayısını göstermektedir.

Sonuçta yukardaki denemelerden de anlaşıldığı gibi iki renkli dokular her iki algoritma ile rahatça üretilir. Birinci algoritmada aynı renkten olan noktaların sayısı değişmemekte (sadece noktaların yerleri değiştiğinden), ikinci algoritmada ise her döngüde bu sayı değişmektedir.

3.5. Dört Renkli Doku Örnekleri

Binom dağılımında genelde en açık ve en koyu renge doğru bir eğilim vardır. Eğer doku örneklerken uygun parametreler seçilmezse, birkaç döngüden sonra doku tek bir rengin hakim olduğu bir hale dönüşür. Bu da 'faz geçişi' olarak bilinir. Faz geçişine neden olan en büyük etmen sınır koşullarının (boundary conditions) seçimidir. Cross, değiş-tokuş yöntemini kullanarak bu olayın önüne geçmiştir [12]. Auto-binomial modelin bir diğer özelliği de doku derecesi yükseldikçe yüksek dereceden parametre değerlerinin küçülmesidir.

Derin & Elliott modelinde de bazı parametre değerlerinde faz geçişi olduğu Chen tarafından gözlenmiştir [9], [19].



(a)



(b)

Şekil 3.5 (a) Birinci algoritma ile (b) ikinci algoritma kullanılarak aynı parametrelerle üretilen dokular.

Tablo 3.2 (a) Birinci (b) ikinci algoritma ile üretilen Şekil 3.5 teki dokuların her döngüdeki nokta değişimleri ve aynı renge sahip noktaların sayısı.

(a)

Değişim		0	1
Başlangıç		1990	2106
1	1920	1990	2106
2	1130	1990	2106
3	989	1990	2106
4	915	1990	2106
5	879	1990	2106
6	833	1990	2106
7	755	1990	2106
8	737	1990	2106
9	712	1990	2106
10	722	1990	2106

(b)

Değişim		0	1
Başlangıç		2034	2062
1	2342	2042	2054
2	1815	2115	1981
3	1658	2103	1993
4	1614	1989	2107
5	1522	2001	2095
6	1364	2055	2041
7	1363	1954	2142
8	1377	1937	2159
9	1357	1814	2282
10	1334	1872	2224

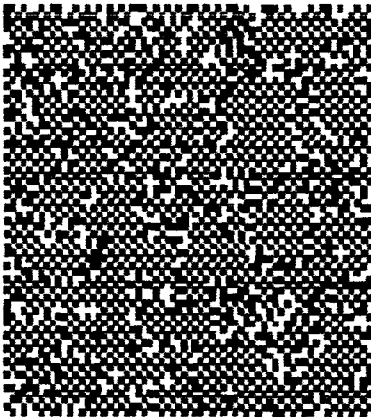
Tablo 3.3 (a) Birinci (b) ikinci algoritma ile üretilen Şekil 3.5 teki dokuların MLE yöntemi ile parametre değeri çıkarımı.

(a)

'coding'	a	b[1,1]	b[1,2]
1. bölge	-3.3452	1.7315	1.6307
2. bölge	-3.3010	1.7317	1.5411
Ortalama	-3.3231	1.7316	1.5859

(b)

'coding'	a	b[1,1]	b[1,2]
1. bölge	-2.8186	1.5240	1.3856
2. bölge	-3.2031	1.7022	1.4302
Ortalama	-3.0109	1.6131	1.4079



(a)



(b)

Şekil 3.6 (a) Birinci algoritma ile (b) ikinci algoritma kullanılarak aynı parametrelerle üretilen dokular.

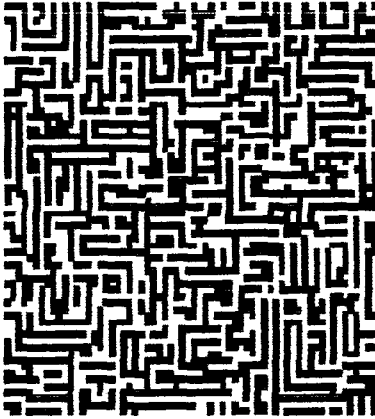
Tablo 3.4 (a) Birinci (b) ikinci algoritma ile üretilen Şekil 3.6 daki dokuların MLE yöntemi ile parametre değeri çıkarımı.

(a)

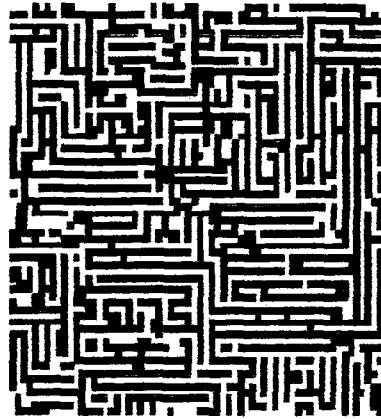
'coding'	a	b[1,1]	b[1,2]
1. bölge	4.1005	-2.0326	-1.9273
2. bölge	4.3574	-2.0027	-2.1728
Ortalama	4.2290	-2.0177	-2.0501

(b)

'coding'	a	b[1,1]	b[1,2]
1. bölge	4.6345	-2.0892	-2.1738
2. bölge	4.4399	-2.0076	-2.0102
Ortalama	4.5372	-2.0484	-2.0920



(a)



(b)

Şekil 3.7 (a) Birinci algoritma ile (b) ikinci algoritma kullanılarak aynı parametrelerle üretilen dokular.

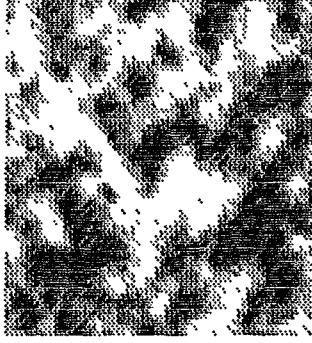
Tablo 3.5 (a) Birinci (b) ikinci algoritma ile üretilen Şekil 3.7 deki dokuların MLE yöntemi ile parametre değer çıkarımı.

(a)

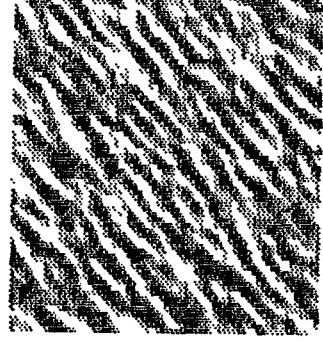
'coding'	a	b[1,1]	b[1,2]	b[2,1]	b[2,2]
1. bölge	0.0952	1.8143	1.8705	-1.7716	-2.1796
2. bölge	-0.0865	1.9174	1.9141	-2.0668	-1.7671
3. bölge	0.4450	1.6870	1.7313	-2.1845	-1.7886
4. bölge	0.0729	1.9148	1.8573	-1.8277	-2.1205
Ortalama	0.1317	1.8334	1.8433	-1.9627	-1.9639

(b)

'coding'	a	b[1,1]	b[1,2]	b[2,1]	b[2,2]
1. bölge	0.2134	1.8818	1.8716	-1.8195	-2.0782
2. bölge	-0.2967	2.2483	2.0559	-1.9302	-1.8056
3. bölge	-0.1149	2.1900	2.2843	-1.8199	-2.4031
4. bölge	0.3238	2.1426	2.3000	-2.3826	-2.3641
Ortalama	0.0314	2.1157	2.1279	-1.9881	-2.1627

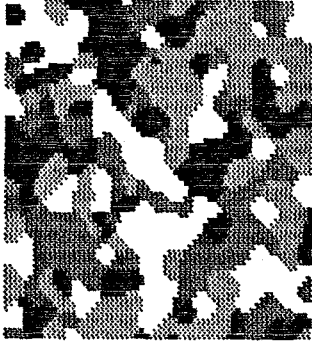


(a)

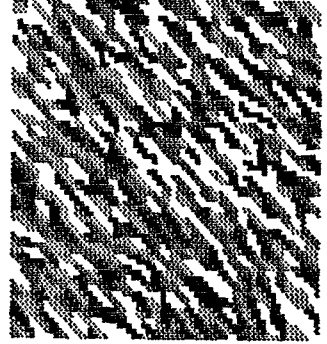


(b)

Şekil 3.8 Auto-binomial model 4-renkli doku örnekleri



(a)



(b)

Şekil 3.9 Derin & Elliott model 4-renkli doku örnekleri

Tablo 3.6 Deneme 3.3 teki Derin & Elliott modeli ile üretilen dokuların parametrelerinin MLE ile bulunması.

(a)

'coding'	b[1,1]	b[1,2]	b[2,1]	b[2,2]
1. bölge	1.0287	1.1666	0.9473	1.0001
2. bölge	1.1305	1.1978	0.9855	0.9319
3. bölge	1.1571	0.9607	0.8369	1.0126
4. bölge	1.1652	0.9910	1.0140	0.8003
Ortalama	1.1204	1.0790	0.9459	0.9362

(b)

'coding'	b[1,1]	b[1,2]	b[2,1]	b[2,2]
1. bölge	0.9598	1.0593	1.0048	-1.1720
2. bölge	1.0015	1.0236	0.9551	-1.1553
3. bölge	1.0530	0.9656	0.9215	-1.1342
4. bölge	1.1369	0.9090	0.9670	-1.1031
Ortalama	1.0378	0.9893	0.9621	-1.1411

Deneme 3.3 : Bu denemede Şekil 3.8 de görülen, 4-renkli, ikinci dereceden, auto-binomial, parametreleri (a) $a=-2$, $b[1,1]=1$, $b[1,2]=1$, $b[2,1]=1$, $b[2,2]=1$; (b) $a=-2$, $b[1,1]=1$, $b[1,2]=1$, $b[2,1]=1$, $b[2,2]=-1$ olan dokular ve Şekil 3.9 da görülen, 4-renkli, ikinci dereceden, Derin & Elliott model, parametreleri (a) $b[1,1]=1$, $b[1,2]=1$, $b[2,1]=1$, $b[2,2]=1$; (b) $b[1,1]=1$, $b[1,2]=1$, $b[2,1]=1$, $b[2,2]=-1$ olan dokular örnek olarak üretilmektedir.

3.6. Özet

Bu bölümde, doku üretiminde kullanılan algoritmaların karşılaştırılması yapılmakta, dokulardan çıkarılan parametrelerin yakınsamaları ve ilk iki algoritma kullanılarak üretilen dokular incelenmektedir. Birinci ve ikinci algoritmalarla üretilen aynı parametrelere sahip iki-renkli dokuların görsel olarak birbirlerine benzedikleri ve MLE ile bulunan parametre değerlerinin de gerçek değerlere yakın oldukları görüldü. Parametrelerin dokular üzerine etkileri örneklerle incelendi.

BÖLÜM 4

İSTATİSTİKSEL İNCELEME VE SONUÇ

4.1. Giriş

Bundan önceki bölümlerde, GRA ve MRA'nın olasılıksal yapısı ve bu alanlardan nasıl örnekler üretileceği üzerinde duruldu. Bu algoritmaları kullanarak ürettiğimiz dokularda parametrelerin önceden bilindiği varsayıldı. Eğer bize GRA veya MRA'na uygun bir doku verilip de, bu dokuya ait parametreler sorulursa ne yapacağız? İşte bu bölümde bu sorunun yanıtı verilmeye çalışılacaktır.

Doku sentezinde asıl amaç, görüntüyü birkaç parametre ile tanımlayabilmektir. Dokular arasındaki benzerlikler sentezlemede ciddi bir sorun olmamaktadır. Amaçlanan, GRA veya MRA'na uygun verilen bir dokuya, görünüşte benzer bir doku üretebilmektir. Bunun için de, bize verilen dokunun parametrelerinin en uygun bir şekilde kestirimi için çeşitli yöntemler gerekmektedir.

1974'de Besag, ilk defa, herhangi bir ayrık GRA veya MRA'da parametre kestiriminde kullanılabilecek bir yöntem önerdi [3]. Bu yöntemde Besag, ayrık GRA veya MRA ile modellenebileceği düşünülen $N \times N$ lik bir dokuyu 'coding' bölgelerine ayırmayı ve her 'coding' bölgesi için ayrı ayrı parametreleri bulup, her bir parametre için bulunan değerlerin ortalamasını alarak dokuya ait parametreleri saptamayı önermektedir.

Derin & Elliott [16] ise, kendi modelleri için bu yöntemin tam güvenilir olmadığını açıklayıp, yeni bir yöntem öne sürmüşlerdir.

4.2. Parametre Kestirimi (Parameter Estimation)

Bu bölümde GRA veya MRA na uygun bir dokudan parametre kestirimi için kullanılabilecek dört temel yöntemden bahsedilecektir. Bunlardan Besag'ın 'coding' yöntemine dayanan 'Maximum-Likelihood' (ML) metodu, tez çalışmasında kullanıldığından, genişçe verilecek, diğer metodlar ise kısaca anlatılacaktır.

Bu metodlar sırasıyla, Derin & Elliott tarafından önerilen 'Least Square Error Method' (LSQR); 'Standard Logit Method' (Logit) ve 'Minimum Logit Chi-Square' (Min-Chi-Square) dır. Bu üç metodu iki-renkli dokular dışında kullanmak, çok büyük bir görüntü hacmi ve dolayısıyla da büyük bellekli sistemler gerektirdiğinden pek kullanışlı değildir. En büyük sorun ise renk sayısı arttığında elde edilen denklem sayısının çok fazla olmasıdır. Parametre kestirim yöntemlerinde enerji işlevi, $U(x)$, aşağıdaki tanımlamalardan sonra vektörel olarak gösterilecek ve öyle kullanılacaktır.

R_t , t noktasına en yakın komşulardan oluşan noktalar kümesi olsun (Şekil 2.1).

Birinci dereceden komşulara sahip alanlar için,

$$R_t = \{x_t: -1, x_t: +1, x_t: -2, x_t: +2\}. \quad (4.1)$$

ikinci dereceden komşulara sahip alanlar için,

$$R_t = \{x_t: -1, x_t: +1, x_t: -2, x_t: +2, x_t: -3, x_t: +3, x_t: -4, x_t: +4\}. \quad (4.2)$$

$w(x_t, R_t)$, t noktası ve ona komşu noktaların renklerine bağımlı bir vektör olsun.

Birinci dereceden komşulara sahip alanlar için,

$$w(x_t, R_t) = \begin{bmatrix} w_0(x_t) & w_1(x_t) & w_2(x_t) \end{bmatrix}^T. \quad (4.3)$$

ikinci dereceden komşulara sahip alanlar için,

$$w(x_t, R_t) = \begin{bmatrix} w_0(x_t) & w_1(x_t) & w_2(x_t) & w_3(x_t) & w_4(x_t) \end{bmatrix}^T. \quad (4.4)$$

w vektöründeki $w_0(x_t)$ değerleri sadece binom modeli için geçerlidir. Çünkü, Derin & Elliott modelinde $\alpha=0$ olarak kabul edilmektedir. Bundan dolayı w vektörü, w_0 değeri olmayan dört parametrelili bir vektör olacaktır.

Birinci dereceden komşulara sahip alanlar için,

$$w(x_t, R_t) = \begin{bmatrix} w_1(x_t) & w_2(x_t) \end{bmatrix}^T. \quad (4.5)$$

ikinci dereceden komşulara sahip alanlar için,

$$w(x_t, R_t) = \begin{bmatrix} w_1(x_t) & w_2(x_t) & w_3(x_t) & w_4(x_t) \end{bmatrix}^T. \quad (4.6)$$

w vektöründe bulunan w_1 yatay klikleri, w_2 dikey klikleri, w_3 135° deki klikleri, w_4 45° deki klikleri, w_0 ise tek noktadan oluşan klikleri göstermektedir.

Derin & Elliott modeli için w vektörünü oluşturan w_1 değerleri,

$$\begin{aligned} w_1(x_t) &= I(x_t, x_t: -1) + I(x_t, x_t: +1), \\ w_2(x_t) &= I(x_t, x_t: -2) + I(x_t, x_t: +2), \\ w_3(x_t) &= I(x_t, x_t: -2) + I(x_t, x_t: +2), \\ w_4(x_t) &= I(x_t, x_t: -2) + I(x_t, x_t: +2), \\ I(a, b) &= -1 \quad \text{eğer } a=b, \\ I(a, b) &= 1 \quad \text{eğer } a \neq b. \end{aligned} \quad (4.7)$$

Auto-binomial model için w vektörünü oluşturan w_i değerleri,

$$\begin{aligned} w_0(x_t) &= ax_t, \\ w_1(x_t) &= x_t(x_t:-1+x_t:+1), \\ w_2(x_t) &= x_t(x_t:-2+x_t:+2), \\ w_3(x_t) &= x_t(x_t:-2+x_t:+2), \\ w_4(x_t) &= x_t(x_t:-2+x_t:+2). \end{aligned} \quad (4.8)$$

Parametre kestiriminde bulunacak parametre vektörünü θ ile gösterirsek,

Derin & Elliott modeli için,

$$\theta = [\theta_1 \quad \theta_2]^T \quad (1. \text{ dereceden}), \quad (4.9)$$

$$\theta = [\theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4]^T \quad (2. \text{ dereceden}), \quad (4.10)$$

Auto-binomial model için,

$$\theta = [\theta_0 \quad \theta_1 \quad \theta_2]^T \quad (1. \text{ dereceden}), \quad (4.11)$$

$$\theta = [\theta_0 \quad \theta_1 \quad \theta_2 \quad \theta_3 \quad \theta_4]^T \quad (2. \text{ dereceden}), \quad (4.12)$$

olacaktır. Burada θ_1 yatay komşuları, θ_2 dikey komşuları, θ_3 135° deki komşuları, θ_4 ise 45° deki komşuları etkileyen parametrelerdir.

Bu gösterimlerden sonra, (2.15) de tanımlanan enerji işlevi, $U(x)$, aşağıdaki gibi yazılabilir.

$$U(x) = h + \frac{1}{2} \sum_{t=1}^M w(x_t, R_t)^T \theta \quad (4.13)$$

Derin & Elliott modeli için, $h=0$;

Auto-binomial model için, $h = - \sum_{t=1}^M \ln[\text{Conb}(G-1, x_t)]$,

$\text{Conb}(m, n) = m! / (m-n)! n!$.

Bu gösterimlerden sonra, (2.16) da tanımlanan yerel koşullu olasılık aşağıdaki gibi yazılabilir.

$$P(x_t | R_t) = P(x_t, R_t) / P(R_t) = \exp(f(x_t) - w(x_t, R_t)^T \theta) / Z_t,$$

$$Z_t = \sum_{g=0}^{G-1} \exp(f(x_t) - w(x_t, R_t)^T \theta), \quad (4.14)$$

Derin & Elliott modeli için, $f(g) = 0$;

Auto-binomial model için, $f(g) = -\ln[\text{Conb}(G-1, g)]$.

Bu tanımlamalardan sonra takip eden bölümlerde bahsedilen dört temel parametre kestirim yöntemleri anlatılmaktadır.

4.2.1. 'Maximum-Likelihood Estimation' (MLE)

MLE yöntemi, kafes üzerindeki noktaların yerel koşullu olasılık değerlerinin çarpımının maximum olmasını sağlayan parametreleri bulmaktır. $P(x_i | R_i)$ yerel koşullu olasılık değerini gösterir. ML yöntemini kullanırsak, M noktadan oluşan bir sayısal doku için,

$$q(\theta) = \prod_i^M P(x_i | R_i) \quad (4.15)$$

'likelihood' işlevini elde ederiz. Bu işlevi maximum yapan θ_i değerleri de bizim aradığımız parametreler olacaktır. $q(\theta)$ 'yı (4.15) maximum yapmak $L(\theta)$ 'yı (4.16) maximum yapmakla eşdeğerdir ve bu 'log-likelihood' olarak bilinir.

$$L(\theta) = \sum_i^M \ln[P(x_i | R_i)] \quad (4.16)$$

(4.16) da tanımlanan bu eşitliği maximum yapmanın zor olduğu apaçık ortadadır. Çünkü toplamı oluşturan koşullu olasılık değerleri birbirinden olasılıksal anlamda bağımsız değildir. Besag, [3] bu soruna bir

çözüm olarak 'coding' yöntemini önermiştir. Bu yöntemde tümü noktalardan oluşan $N \times N$ boyutlu kafes, 'coding' diye isimlendirilen birbirinden ayrık noktalar kümesine ayrılır. Herbir 'coding' bölgesi, kendini oluşturan noktalar birbirinden olasılıksal anlamda bağımsız olacak şekilde seçilir. Eğer t ve s aynı 'coding' bölgesine ait iki nokta ise, Bayes kuralı ve Markov özelliği (2.8) kullanılarak, $P(x_t, x_s \mid R_t, R_s) = P(x_t \mid R_t)P(x_s \mid R_s)$ yazılabilir. Bu bize 'coding' bölgesindeki noktaların olasılıksal anlamda birbirinden bağımsız olduğunu göstermektedir. Yani, 'coding' içindeki herhangi iki nokta birbirinin olasılıksal komşusu değildir. Bu da 'coding' bölgesine ait herhangi bir t noktası etrafında seçilen modelin derecesine uygun bir boşluk verilerek sağlanır.

Başka bir deyişle, gerekli 'coding' sayısı komşuluk derecesine bağlıdır. Birinci dereceden modeller için en az iki 'coding' bölgesine (Şekil 4.1), ikinci dereceden modeller için dört 'coding' bölgesine (Şekil 4.2), üçüncü ve dördüncü dereceden modeller için ise dokuz 'coding' bölgesine (Şekil 4.3) ihtiyaç vardır.

```

1 2 1 2 ..... 1 2 1 2
2 1 2 1 ..... 2 1 2 1
1 2 1 2 ..... 1 2 1 2
2 1 2 1 ..... 2 1 2 1
1 2 1 2 ..... 1 2 1 2
2 1 2 1 ..... 2 1 2 1

```

Şekil 4.1 Birinci dereceden komşuluk için 'coding' bölgeleri. "1" ve "2" şeklinde numaralandırılmış iki ayrı bölge.

```

1 2 1 2 ..... 1 2 1 2
3 4 3 4 ..... 3 4 3 4
1 2 1 2 ..... 1 2 1 2
3 4 3 4 ..... 3 4 3 4
1 2 1 2 ..... 1 2 1 2
3 4 3 4 ..... 3 4 3 4

```

Şekil 4.2 ikinci dereceden komşuluk için 'coding' bölgeleri. "1", "2", "3" ve "4" şeklinde numaralandırılmış dört ayrı bölge.

```

1 2 3 1 2 3 ..... 1 2 3 1 2 3
4 5 6 4 5 6 ..... 4 5 6 4 5 6
7 8 9 7 8 9 ..... 7 8 9 7 8 9
1 2 3 1 2 3 ..... 1 2 3 1 2 3
4 5 6 4 5 6 ..... 4 5 6 4 5 6
7 8 9 7 8 9 ..... 7 8 9 7 8 9

```

Şekil 4.3 Üçüncü ve dördüncü dereceden komşuluk için 'coding' bölgeleri. "1" den "9" a kadar numaralandırılmış dokuz ayrı bölge.

S_j , aynı numaralardan oluşan noktalar kümesini yani, 'coding' bölgelerini gösterebilir. Bu durumda, her bir 'coding' bölgesindeki yerel koşullu olasılıkların çarpımını maximum yapan θ_i değerlerini bulmak için kullanacağımız ML yöntemi için gerekli olan $q(\theta)$ (4.15) ve $L(\theta)$ yı (4.16) yeniden yazarsak,

$$q(\theta) = \prod_{i \in S_j}^M P(x_i | R_i) \quad (4.17)$$

$$L(\theta) = \sum_{i \in S_j}^M \ln[P(x_i | R_i)] \quad (4.18)$$

olur. Sonuçta her bir 'coding' bölgesi için bulunan parametrelerin ortalaması alınarak tüm dokuyu modelleyen parametre değerleri bulunmuş olur.

$$\hat{\theta} = \frac{1}{n} \sum_{j=1}^n \theta_j \quad n : \text{'coding' sayısı} \quad (4.19)$$

Doğrusal olmayan bir işlev olan $L(\theta)$ yı (4.18) maximum yapan θ_k değerleri, bu işlevin θ_k lara göre alınan kısmi türevlerini sıfır yapan değerlere eşittir. Yani,

$$\frac{\delta L(\theta)}{\delta \theta_k} = 0 \quad (4.20)$$

yapan θ_k değerleri $L(\theta)$ yı (4.18) maximum yapan değerlerdir. Kuşkusuz ki bu değerleri bulmak için çok çeşitli sayısal yöntemler vardır. Bu tez çalışmasında, parametrelere göre kısmi türev alarak elde edilen bu doğrusal olmayan eşitlikler (parametre sayısı kadar) Newton-Raphson metodu kullanılarak çözülecektir. Aşağıda her iki model için de kullanılan kısmi türev eşitlikleri verilmiştir.

Auto-binomial model için,

$$\begin{aligned} L(\theta) &= \sum_{i \in s_j} \ln[P(x_i | R_i)] \\ &= \sum_{i \in s_j} \ln \left[\frac{\text{Conb}(G-1, x_i) \cdot \exp(x_i T)}{(1 + \exp(T))^{G-1}} \right], \end{aligned} \quad (4.21)$$

$$T = ([w(x_i, R_i)]^T \hat{\theta}) / x_i .$$

Birinci dereceden auto-binomial model için kısmi türevler:

$$F_1 = \frac{\delta L(\theta)}{\delta \theta_0} = \sum_{i \in s_j} \frac{x_i + (x_i - (G-1)) \exp(T)}{(1 + \exp(T))}, \quad (4.22)$$

$$F_2 = \frac{\delta L(\theta)}{\delta \theta_2} = \sum_{i \in S_j} \frac{[w_2(x_i)/x_i][x_i + (x_i - (G-1))\exp(T)]}{(1 + \exp(T))}, \quad (4.23)$$

$$F_3 = \frac{\delta L(\theta)}{\delta \theta_3} = \sum_{i \in S_j} \frac{[w_3(x_i)/x_i][x_i + (x_i - (G-1))\exp(T)]}{(1 + \exp(T))}. \quad (4.24)$$

ikinci dereceden auto-binomial model için kısmi türevler, T değeri ikinci dereceden komşuluk sistemine göre hesaplanan yukardaki F_1 , F_2 , F_3 ve aşağıda verilen F_4 , F_5 işlevleridir.

$$F_4 = \frac{\delta L(\theta)}{\delta \theta_4} = \sum_{i \in S_j} \frac{[w_3(x_i)/x_i][x_i + (x_i - (G-1))\exp(T)]}{(1 + \exp(T))}, \quad (4.25)$$

$$F_5 = \frac{\delta L(\theta)}{\delta \theta_5} = \sum_{i \in S_j} \frac{[w_4(x_i)/x_i][x_i + (x_i - (G-1))\exp(T)]}{(1 + \exp(T))}. \quad (4.26)$$

Derin & Elliott model için,

$$L(\theta) = \sum_{i \in S_j} \ln[P(x_i | R_i)]$$

$$= \sum_{i \in S_j} \ln \left[\frac{\exp(-[w(x_i, R_i)]T\theta)}{\sum_{g=0}^{G-1} \exp(-w[g, R_i])T\theta)} \right]. \quad (4.27)$$

Birinci dereceden Derin & Elliott modeli için kısmi türevler:

$$F_1 = \frac{\delta L(\theta)}{\delta \theta_1} = \sum_{i \in S_j} \left[\frac{\sum_{g=0}^{G-1} [w_1(g) - w_1(x_i)] \exp(-[w(x_i, R_i)]T\theta)}{\sum_{g=0}^{G-1} \exp(-[w(g, R_i)]T\theta)} \right] \quad (4.28)$$

$$F_2 = \frac{\delta L(\theta)}{\delta \theta_2} = \sum_{i \in s_j} \left[\frac{\sum_{g=0}^{G-1} [w_2(g) - w_2(x_i)] \exp(-[w(x_i, R_i)]^T \theta)}{\sum_{g=0}^{G-1} \exp(-[w(g, R_i)]^T \theta)} \right] \quad (4.29)$$

ikinci dereceden Derin & Elliott model için kısmi türevler, $[w(x_i, R_i)]^T$ değeri ikinci dereceden komşuluk sistemine göre hesaplanan yukardaki F_1 , F_2 ve aşağıda verilen F_3 , F_4 işlevleridir.

$$F_3 = \frac{\delta L(\theta)}{\delta \theta_3} = \sum_{i \in s_j} \left[\frac{\sum_{g=0}^{G-1} [w_3(g) - w_3(x_i)] \exp(-[w(x_i, R_i)]^T \theta)}{\sum_{g=0}^{G-1} \exp(-[w(g, R_i)]^T \theta)} \right] \quad (4.30)$$

$$F_4 = \frac{\delta L(\theta)}{\delta \theta_4} = \sum_{i \in s_j} \left[\frac{\sum_{g=0}^{G-1} [w_4(g) - w_4(x_i)] \exp(-[w(x_i, R_i)]^T \theta)}{\sum_{g=0}^{G-1} \exp(-[w(g, R_i)]^T \theta)} \right] \quad (4.31)$$

Bu işlevlerden Newton-Raphson yöntemi için gerekli olan 'Jacobian Matrisi' de kolayca hesaplanıp uygun başlangıç koşulları verilerek çözüm hesaplanır.

4.2.2. 'Least Square Error Method' (LSQR)

Bu metod ilk kez Derin & Elliott tarafından kendi modelleri için önerildi [16]. Derin & Elliott bu yön-

temle parametre kestirim sorununu doğrusal denklem takımlarından oluşan (4.32) bir sistem çözümüne indirgemişlerdir.

$$C\theta = q \quad (4.32)$$

Burada, θ parametre vektörü, C ve q de verilen görüntüden hesaplanan değerler olmaktadır.

(4.14) deki eşitlikten,

$$\frac{\exp(-[w(x_t, R_t)]^T \theta)}{P(x_t, R_t)} = \frac{Z_t}{P(R_t)} \quad (4.33)$$

yazılabilir. Bu eşitliğin sağ tarafının t noktasının komşularının renklerine, eşitliğin sol tarafının ise x_t değerine bağımlı olduğu açıkça görülmektedir. Dolayısıyla (4.33) deki eşitlik,

$$\frac{\exp(-[w(x_t, R_t)]^T \theta)}{P(x_t, R_t)} = \frac{Z_t}{P(R_t)} = \frac{\exp(-[w(x_s, R_s)]^T \theta)}{P(x_s, R_s)}, \quad (4.34)$$

$R_s = R_t$ ise,

şeklinde yazılabilir. Ufak bir sadeleştirme ile,

$$[w(x_t, R_t) - w(x_s, R_s)]^T \theta = \ln[P(x_s, R_s)/P(x_t, R_t)], \quad (4.35)$$

$R_s = R_t$ ise,

eşitliğini elde ederiz.

(4.34) ve (4.35) daki t ve s noktaları, aynı komşu kümesine sahip, $R_s = R_t$, olacak şekilde rastgele seçilirler. Yeni bir düzenleme ile, R_s ve R_t yi D , s ve t yi de u ve v ile gösterirsek, (4.35) teki eşitlik,

$$[w(u, D) - w(v, D)]^T \theta = \ln[P(v, D)/P(u, D)] \quad (4.36)$$

şeklinde yazılabilir.

Bu durumda bütün doğrusal sistemi oluşturan eşitlikler (4.36) formunda olacaktır. $P(v,D)/P(u,D)$ ise, ikinci dereceden bir komşuluk sistemi için v renginde ve D komşuluğuna sahip 3×3 lük blokların sayısının, u renginde ve yine aynı D komşuluğuna sahip 3×3 lük blokların sayısına oranı olarak hesaplanmaktadır [16]. İkinci dereceden iki-renkli bir GRA veya MRA için $2^8=256$ tane eşitlik vardır.

Eğer $w(u,D)=w(v,D)$ olursa, $P(u,D)=P(v,D)$ olacaktır. Bu durumdan sakınmak için,

$$[w(u,D) - w(v,D)]^{T\theta} = \ln[(P(v,D)+\epsilon)/(P(u,D)+\epsilon)], \quad (4.37)$$

$$\epsilon=1/(2 \times 256)=1/256$$

şeklinde bir küçük değer bu olasılıklara eklenir.

Derin & Elliott, doğru sonuca varmak için, görüntü üzerinde en sık bulunan blok tiplerini, (u,D) ve (v,D) gibi, almanın yeterli olacağını söylemektedirler [16]. Böylece doğrusal sistem için elde edilen eşitlikler daha az olacaklardır. Fakat buna rağmen, en sık bulunan blok tiplerini seçmek veya ihmal edilecek blokları belirlemek için henüz uygun bir yol önerilmemiştir. 'Perturbation' faktörü ϵ 'nın seçimini için de, çok önemli olmasına rağmen, herhangi bir teorik öneri yoktur. Literatürde bu değer yukarıdaki gibi seçilmektedir.

4.2.3. 'Logit Model Fit Method' (Logit)

Bu metod, LSQR metodunun benzeridir [9]. Bu yöntem, parametre kestirim sorununu bir Logit model uygunluk problemine dönüştürmektedir. İkinci dereceden modellenmiş iki-renkli bir doku düşünelim (0 ve 1 lerden oluşmuş). t noktasının sekiz komşusunun renklerinden oluşan R_t lerin tüm olası kümelerinin toplamını $\bigwedge$ ile gösterelim. $|\bigwedge|=2^8=256$. Eğer $w(0,E)=w(0,F)$ ise $E \approx F$

diyebiliriz ($E, F \in \wedge$). Bu durumda $|\wedge|=81$ olacak ve sorun sadece aşağıdaki sistem çözümüne indirgenecektir.

$$[w(1,D) - w(0,D)]T\theta = \ln[(P(0,D)+\epsilon)/(P(1,D)+\epsilon)], \quad (4.38)$$

$$\epsilon=1/(2 \times 81)=1/162 .$$

Bu eşitlikler de 'Least Square Error Method' kullanılarak rahatça çözülür.

4.2.4. 'Minimum Logit Chi-Square Method' (Min-Chi-Square)

Bu metod ikili 'binary' değişkenlere bir Logit model uydurmak için Berkson'un minimum χ^2 (Ki-Kare) işlemine dayanmaktadır [2]. Bir önceki bölümde ikinci dereceden iki-renkli bir dokuda 81 tane birbirinden farklı 3×3 lük blokların olduğunu gördük. Bu yöntemde 3×3 lük blokların herbirinin merkezinde bulunan noktanın 0 veya 1 değeri aldığı düşünülmektedir.

$r_i = i$ tipinde merkezindeki nokta değeri 1 olan blokların sayısı.

$s_i = i$ tipinde merkezindeki nokta değeri 0 olan blokların sayısı.

$n_i = r_i + s_i$, $i=1,2,\dots,81$.

$p_i = r_i/n_i$, $q_i = s_i/n_i$, $l_i = \ln(p_i/q_i)$, $\bar{l}_i = \ln(P_i/Q_i)$,

$i=1,2,\dots,81$, $P_i = P(1 | R_i)$, $Q_i = P(0 | R_i)$,

R_i : i tipindeki komşulukta bulunan noktaların kümesi.

Önerilen bu metodda, bulunacak parametreler aşağıdaki eşitliği minimum yapan parametrelerdir.

$$W(\theta) = \sum_{i=1}^{81} \left[\frac{(r_i - n_i P_i)^2}{n_i P_i} + \frac{(s_i - n_i Q_i)^2}{n_i P_i} \right] \quad (4.39)$$

$p_i \approx P_i$ ve $q_i \approx Q_i$ dersek,

$$W(\theta) = \sum_{i=1}^{81} n_i p_i q_i (l_i - \hat{l}_i)^2 \quad (4.40)$$

olur.

(4.40) teki eşitliğin kısmi türevleri alındığında karşımıza parametre sayısı kadar doğrusal eşitlik çıkmaktadır. Bu eşitlikler de çeşitli sayısal yöntemlerle rahatça çözülebilir.

4.2.5. İrdelene

Doğrusal olmayan denklem takımını çözmek için kullanılan Newton-Raphson yönteminde, çözüm ve yakınsama koşulu için bir başlangıç değerine ihtiyaç vardır. Aksi takdirde uygun bir çözüme ulaşılamadan yerel maximum veya minimum değerlere varılabilir. Bunun için değişik başlangıç koşulları için parametre kestirim programı tekrarlanmalı ve çözüm vektörü civarındaki $L(\theta)$ nın değeri kontrol edilmelidir. (Derin & Elliott modelinde MLE ile parametre kestirimi $L(\theta)$ değerleri kontrol edildi). Bizim çalışmalarımızda başlangıç değeri için bütün parametreler 0 (sıfır) olarak alındı.

MLE yöntemini anlatırken iki-renkli doku gibi bir kısıtlama yapılmadı. Fakat diğer metodlar için böyle bir kısıtlama yapmak gerekecektir. LSQR ve Logit metodunun sezgisel olarak çok-renkli dokulara uyarlanabileceği görülse bile, büyük görüntü hacmi gerektirdiğinden ve renk sayısı arttıkça blok tiplerinin sayısının hızla artması nedeniyle kullanılamıyacağı, Min-Chi-Square metodunun ise sadece iki renkli dokular üzerine uygulanabileceği apaçık ortadadır.

4.3. Uygunluk Testi (Goodness-of-Fit-Test)

Aşağıda verilen 'Null' hipotezi test etmek için, θ parametreleri ile bir MRA modeli ve bir tek X dokusunun verildiğini düşünelim.

H_0 : Verilen doku, θ parametreleri ile verilen MRA veya GRA nın bir örneğidir.

Bir önceki bölümde, GRA veya MRA nından parametre kestirim yöntemleri üzerinde duruldu. Bulunan parametrelerin doğruluğu, bu parametrelerle üretilen doku ile verilen dokuyu karşılaştırmakla test edilebilir. Bu karşılaştırmamanın bir dezavantajı, insan görme sisteminin aldanma olasılığının yüksek olmasıdır. Çünkü üretilen doku ile verilen doku gözle birbirinden farklı olsa bile, gerçekte üretilen doku verilen dokuya çok iyi uyan bir model olabilir.

MRA veya GRA doku modeli ile verilen doku arasındaki uygunluğu karakterize edecek bir ölçüye ihtiyaç vardır. Literatürde, sadece Cross [12], [13], Besag'ın önerilerini [3] dikkate alarak böyle bir ölçümü dokular üzerine uyguladı. Cross, verilen doku ile auto-binomial modele göre üretilen dokular arasındaki uygunluğu χ^2 (Ki-kare) testi uygulayarak ölçmeye çalıştı. Bu yöntemde, özellikle iki-renkli olmayan dokular için, görüntü hacminin çok büyük olması gerektiğinden hesaplama karmaşıklığı ile karşılaşilmektedir.

4.3.1. Ki-Kare (χ^2) Testi (Chi-Square Test)

$N \times N$ lik kafes, L , üzerinde renk kümelerinden oluşan bilinmeyen $P(X)$ dağılımına sahip bir görüntü düşünelim. Koşullu olasılığı bulunacak x noktası olarak merkezi bir nokta seçilirse ve onun her bir gözlenebilen komşuluk yerleşkesine bir örnek noktası dersek, $M = N^2/k$ tane,

$p(x | \cdot)$, koşullu olasılık yoğunluk değerine sahip örnek noktalarımız olur (k , kafes üzerindeki 'coding' sayısı). Bu durumda, biz, her bir merkezi noktanın beklenen sıklığı ile, gözlenen sıklığı arasındaki uygunluğu Ki-Kare testini uygulayarak ölçebiliriz. Burada beklenen sıklık değerleri, kestirim ile bulunan parametreler kullanılarak hesaplanmaktadır. Örneğin, 64×64 boyutlarında birinci dereceden komşuluk sistemine ve $\theta = [-3 \quad 1 \quad 1.5]$ parametrelerine sahip, merkez noktası $x_t = 1$ ve komşu noktaların $x_{t:-1} = 1$, $x_{t:+1} = 0$, $x_{t:-2} = 1$, $x_{t:+2} = 0$ olan bir iki-renkli dokuda, bu komşuluk yerleşmesine sahip ve merkez noktası 1 olan örneklerin sayısı ($k=2$, k : 'coding' bölgesi),

$$\text{beklenen sıklık sayısı} = (\text{toplam nokta sayısı} / k) * P(x_t | R_t)$$

$$\text{b.s.s.} = (N^2 / k) * \frac{\exp[x_t(\theta_0 + \theta_1(x_{t:-1} + x_{t:+1}) + \theta_2(x_{t:-2} + x_{t:+2}))]}{1 + \exp[x_t(\theta_0 + \theta_1(x_{t:-1} + x_{t:+1}) + \theta_2(x_{t:-2} + x_{t:+2}))]}$$

$$= 2048 * \frac{\exp(-3 + 1*1 + 1.5*1)}{1 + \exp(-3 + 1*1 + 1.5*1)}$$

$$= 2048 * 0.37754$$

$$= 677.28.$$

Deneme 4.1 : $a = \theta_0 = -3.01$, $b[1,1] = \theta_1 = 1.61$, $b[1,2] = \theta_2 = 1.41$ verilen dokudan çıkarılan parametre değerleri olsun. Tablo 4.1 de bu dokuya ait komşuluk yerleşmeleri, beklenen ve gözlenen sıklık değerleri her bir 'coding' bölgesi için ('coding' siz, 1. ve 2. 'coding') görülmektedir. Beklenen sıklık değerleri, yukarıda ki örnekte anlatıldığı gibi parametre kestirim yöntemlerinden biri ile bulunan parametreler ile hesaplanan koşullu olasılık değerleri kullanılarak hesaplanır. Ki-Kare testinde uygunluğa karar vermek için gerekli olan serbestlik derecesi de Cross tarafından [12] şu şekilde verilmiştir.

Tablo 4.1 2-renkli 64x64 boyutlarındaki bir dokunun Ki-Kare testi örneği. (a) 'coding' siz, (b) birinci 'coding' bölgesi, (c) ikinci 'coding' bölgesi için. 6 df serbestlik derecesine göre bulunan Ki-Kare değerleri (a) 8.59769960, (b) 13.10600970, (c) 7.20332038. ($u1=x_t:-1$, $u2=x_t:+1$, $t1=x_t:-2$, $t2=x_t:+2$)

(a)

u1+u2	t1+t2	x=0 Göz.	x=0 Bekl.	x=1 Göz.	x=1 Bekl.
0	0	702	695	27	34
0	1	308	320	77	65
0	2	38	40	35	33
1	0	274	277	71	68
1	1	318	303	291	306
1	2	81	81	337	337
2	0	27	33	47	41
2	1	76	79	401	398
2	2	48	45	938	941

(b)

u1+u2	t1+t2	x=0 Göz.	x=0 Bekl.	x=1 Göz.	x=1 Bekl.
0	0	364	359	13	18
0	1	153	163	43	33
0	2	14	19	20	15
1	0	130	135	38	33
1	1	156	159	164	161
1	2	34	36	152	150
2	0	12	17	25	20
2	1	35	37	188	186
2	2	30	23	477	484

(c)

u1+u2	t1+t2	x=0 Göz.	x=0 Bekl.	x=1 Göz.	x=1 Bekl.
0	0	338	335	14	17
0	1	155	157	34	32
0	2	24	21	15	18
1	0	144	142	33	35
1	1	162	144	127	145
1	2	47	45	185	187
2	0	15	17	22	20
2	1	41	42	213	212
2	2	18	22	461	457

$$df=(G-1) * N_c - E$$

Burada,

G : Maximum renk sayısı,

N_c: Olası komşuluk yerleşkesi sayısı,

E : Kestirimde bulunan parametre sayısı.

Bizim denememizde G=2, N_c=9, E=3 olduğundan,

$$df=(2-1)*9-3=6$$

olarak bulunur. Bu testte her bir komşuluk yerleşkesinde, en az bir tane beklenen ve gözlenen sıklık bulunduğu kabul edilmektedir. Ki-Kare testi verilen doku için tekrar tekrar uygulanır ve uygunluğun kabulü veya reddi araştırılır [12]. Örneğin, bu denemede %5 lik Ki-Kare sınırına göre (a) ve (c) kabul (b) ise reddedilir.

4.3.2. Parametrik Olmayan Kerteleme Testi 'Nonparametric Ranking Test'

Bu yöntem gerçekte bir Monte-Carlo benzetiminden (simulation) başka bir şey değildir. Belirlenmiş parametreler ile elde edilen bir GRA veya MRA'nın gerçeklemesi gözlenen doku ise, aynı parametrelerle üretilmiş görüntülerden alınan bir dokuyu gözlenen doku ile birbirinden ayırmanın oldukça zor olacağı apaçık ortadadır. Kısaca şu şekilde ifade edecek olursak; verilen görüntü, X , ve bilinen parametrelerle birbirinden bağımsız olarak üretilen (GRA veya MRA) n tane görüntü de y_i , $1 \leq i \leq n$, olsun. Bizim amacımız da, verilen duyarlılık seviyesinde X in $\{y_i\}$ kümesindeki görüntülere uygun olup olmadığına karar vermek olacaktır [9].

$v_0=(v_{01}, v_{02}, \dots, v_{0k})$, verilen görüntü X i karakterize eden ölçü veya özelliklerden oluşan vektör (feature vector), v_i de y_i , $1 \leq i \leq n$, görüntü kümesini karakterize eden özellik vektörleri olsun. v_j , j . görüntüyü içermeyen ortalama özellik vektörü olarak tanımlansın.

$$\bar{v}_j = \frac{1}{n} \sum_{i=0}^n v_i, \quad j=0,1,\dots,n \quad (4.41)$$

u_j , j . görüntü ile diğer görüntüler arasındaki fark ölçüsü olarak tanımlansın.

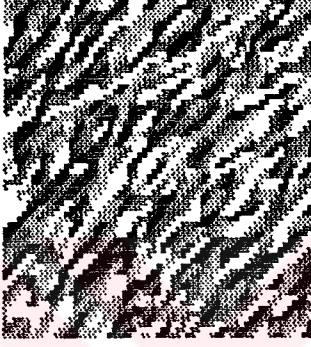
$$u_j = |v_j - \bar{v}_j|, \quad j=0,1,\dots,n \quad (4.42)$$

Kerte (Rank) $\{u_0, u_1, \dots, u_n\}$, küçükten büyüğe doğru sıralansın, $u(0) \leq u(1) \leq \dots \leq u(n)$.

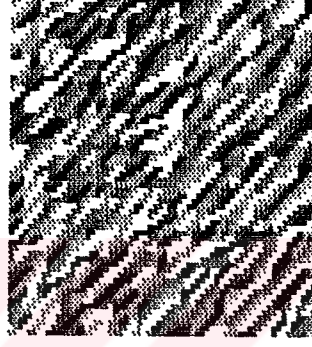
Bütün bu tanımlamalardan sonra H_0 testi, aşağıdaki gibi ifade edilebilir.

Eğer $u_0 > u(n)$ ise, % $100(1-t/(n+1))$ duyarlılıkta H_0 reddet. Genellikle $n=99$ ve $t=95$ olarak seçilir. Görüntüyü karakterize eden özellik vektörlerini seçmek için çeşitli yollar vardır. Örneğin, 'co-occurrence' matrisinden türetilen vektörler gibi (Bölüm 5 de anlatılacak). Biz ise deneme 4.2 için v_i , i . yönde bulunan aynı renge sahip nokta çiftlerinin sayısı olmak üzere $v=(v_1, v_2, v_3, v_4)$ şeklinde bir özellik vektörü kullandık.

Literatürde, uygunluk testi problemi hesaplama zorluğundan ve araştırmacıların verilen görüntü ile bu görüntü için bir model gerçekleştirme arasındaki uygunluğa göz ile karar vermeyi yeterli bulduklarından çoğunlukla ihmal edilir [19]. Uygunluk testi, özellikle rastlantı alan modellerinin doğal dokulara [5] uygunluğunu kontrol için önemlidir. Örneğin, Cross [12] ikinci derece auto-binomial modelleri, Chellappa [8] dördüncü dereceden Gaussian modelleri Brodatz'ın kitabındaki [5] dokulara uyarlamıştır.



(a)



(b)

Şekil 4.4 (a) verilen doku, (b) yeniden üretilen doku

Deneme 4.2 : $\theta_1=1$, $\theta_2=1$, $\theta_3=-1$, $\theta_4=1$ parametrelerine sahip 2. dereceden Derin & Elliott GRA düşünelim ve bu parametrelerle 1. algoritmayı kullanarak 64x64 boyutlarında 4-renkli yapay dokuyu üretelim. Bu dokudan MLE ile elde edilen parametreler: $\theta_1=1.0017$, $\theta_2=1.0007$, $\theta_3=-1.1043$, $\theta_4=0.9636$. Bu parametreleri kullanarak 99 tane doku üretelim. Yukarıda önerildiği gibi v_1 , i. yönde bulunan aynı renge sahip nokta çiftlerinin sayısı olmak üzere $v=(v_1, v_2, v_3, v_4)$ şeklinde bir özellik vektörü belirleyip, üretilen her doku için değerlerini bulalım. Parametrik olmayan kerteleme testinde önerildiği gibi i. görüntüyü içermeyen ortalamaları hesaplayalım. Bu ortalamaların kertesini hesaplandığında (kerteleme işleminde, birbirine eşit veya çok yakın olan ortalama değerlerinden sadece biri alınır) kerte 95 ten küçük çıktığından H_0 hipotezi kabul edilir. Şekil 4.1 de verilen ve çıkarılan parametrelerle yeniden üretilen doku görülmektedir.

4.4. Özet

Bu bölümde, GRA ve MRA parametreleri için kullanılabilecek parametre kestirim yöntemleri anlatıldı ve bunların teorik olarak karşılaştırması yapıldı. Verilen doku ile yeniden üretilen doku arasındaki uygunluğun ölçümü için kullanılabilecek iki ayrı uygunluk testi verildi. (Bunlardan parametrik olmayan kerteleme testinin 'nonparametric ranking test' çok-renkli dokular için de rahatça kullanılabileceği açıkça görülmektedir).

BÖLÜM 5

DOKU SINIFLANDIRMA

5.1. Giriş

Doku sınıflandırma, günümüzde, tıbbi görüntü analizinde (akciğere ait normal ve normal olmayan röntgen filmlerinin ayırılabilmesi gibi), uzaktan algılamada (radar görüntüleri kullanılarak ekin tiplerinin tanımlanması gibi), petrol endüstrisinde (havadan çekilen resimlerle yeraltındaki petrol rezervlerini bulmada), halı endüstrisinde ('carpet wear assesment' [34] ve kalite kontrolunda) yaygın bir şekilde kullanılmaktadır.

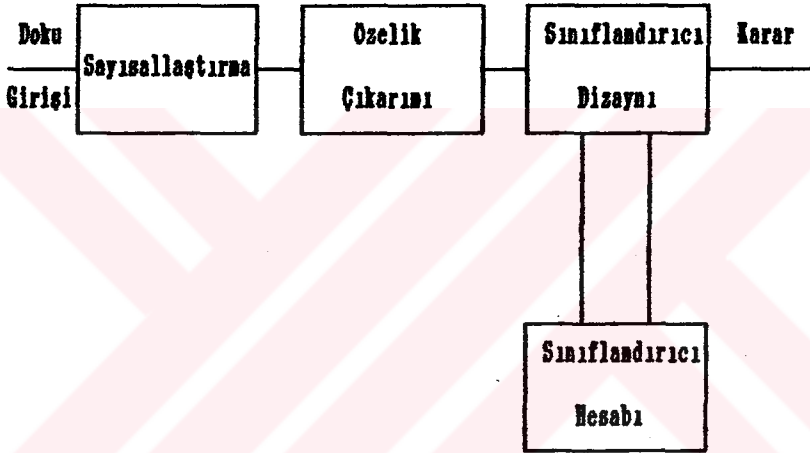
Doku sınıflandırma problemi bir görüntü tanıma (pattern recognition) sorunu olarak ele alınabilir [20]. Bu durumda doku sınıflandırma problemini aşağıdaki şekilde ifade etmek olasıdır.

Elimizde i sınıfına ait, $1 \leq i \leq K$, dokudan n_i tane olsun, öyle bir doku özellik vektörleri ve karar kuralı seçelim ki, hangi sınıfa ait olduğu bilinmeyen doku doğru bir şekilde sınıflandırılsın.

Bu bölümde yukarıda tanımlandığı gibi bir doku sınıflandırma sistemi karakterize edilip, doku sınıflandırma için literatürdeki teknikler kısaca verilecek ve Derin & Elliott modeline uygun üretilen dokular ve bu model ile auto-binomial modele uygun üretilen dokular arasında bir sınıflandırma yapılacaktır.

5.1. Doku Sınıflandırma Sistemi

Şekil 5.1 de kabaca bir doku sınıflandırma sistemi gösterilmektedir. Bu doku sınıflandırma sisteminin girişine verilecek görüntü, $N \times N$ boyutlarında her bir noktası olası renk değerlerine sahip bir matris olarak düşünülen sayısallaştırılmış bir dokudur. (Biz doku sınıflandırma denemelerimizde kendi ürettiğimiz yapay dokuları kullandık).



Şekil 5.1 Doku Sınıflandırma Sistemi

Sınıflandırma sistemindeki en önemli iki adım, **Özellik çıkarımı** (feature extraction) ve **sınıflandırıcı tasarımı** (classifier design) dir. Özellik çıkarımındaki amaç, $N \times N$ boyutlu dokuyu mümkün olduğu kadar çok bilgi içeren bir özellik vektörü ile belirleyebilmektir. Yani, $N \times N$ lik dokunun boyutlarını bilgi kaybı olmadan en aza indirmektedir. Amaçlanan, aynı doku sınıfına ait dokulardan elde edilen özellik vektörlerinin, özellik uzayında belirli bir bölgede yığılması ve bu yığılmaların değişik doku sınıfları için farklı farklı yerlerde bulunmasıdır. Sınıflayıcı, ayırdetme işlevine (discriminant function) göre, özellik uzayını K ayırık bölgeye ayırır ve bilinmeyen bir dokudan hesaplanan özellik vektörünü K bölgelerinden birinin içine atar.

Bir sınıflayıcı kendisine ait sınıflama hata olasılığı ile değerlendirilir [20]. Doku sınıflama probleminde, sayısal bir dokudan çıkarılan özelliklerin sınıflayıcı tasarımını da optimize etmesi istenir. Dolayısıyla sınıflama hata olasılığı özelliklerin geçerliliğinin de bir ölçütü olacaktır.

5.3. Özellik Belirleme ve Çıkarımı

Dokusal özellikler, dokunun karakteristik özelliklerini içerecek şekilde seçilirler. Dokusal özelliklerin çıkarımı, doku sınıflandırmanın en önemli kısmını oluşturur. İki tip dokusal özellik vardır. Tanımlayıcı (descriptive) ve üretici (generative). Tanımlayıcı dokusal özellikler, dokunun gözle de algılanabilecek tipik karakteristiklerinden oluşur. Örneğin, iri tanelilik (coarseness), doğrultululuk (directionality), zıtlık (contrast), çizgililik (line-likeness), düzenlilik (regularity), pürüzlülük (roughness) gibi. Fakat bu tanımlayıcı özellikler dokuyu yeniden üretmek için kullanılamaz. Ancak üretici dokusal özellikler dokunun karakteristik özelliklerini belirlemelerinin yanı sıra, dokuyu yeniden üretmek için de kullanılabilir. Doku sınıflamadaki özellik çıkarımının amacı, maximum ayırtedici bilgiyi içeren minimum bir özellik kümesi bulmaktır.

Tanımlayıcı dokusal özellikler, 'Spatial Gray Level Dependence Matrices' (SGLDM), 'Fourier Power Spectrum' (FPS), 'Gray Level Run Length Matrices' (GLRLM) ve 'Gray Level Difference Matrices' (GLDM) yöntemlerinden biri kullanılarak bulunan 'co-occurrence' matrisinden türetilir [37], [34]. Literatürde yapılan araştırmalar ve teorik çalışmalar SGLDM metodunun diğerlerinden daha iyi olduğunu göstermiştir [37], [34]. Elimizdeki doku örnekleri ne kadar sınırlı olsa da SGLDM, GLRLM, GLDM veya FPS den binlerce özellik üretmek olası olduğundan en iyi özellikleri seçmek de ayrı bir sorun oluşturmaktadır.

Faugeras ve Pratt [21], Üretici dokusal özellikleri bulmak için 'decorrelation' metodunu önermişlerdir. Üretici doku özelliklerine örnek verecek olursak, görüntüye beyazlatma (whitening) operatörü uygulayarak elde edilen artıklardan (residual) hesaplanan ortalama (mean), sapma (deviation), 'skewness', 'kurtosis' gibi özellikler sayılabilir.

Özellik vektörü çıkarmanın bir başka yolu da, MRA na uygun üretilmiş dokuların parametrelerini kullanmaktır. Çünkü bu parametreler dokuya ait temel karakteristikleri içermektedirler (Bölüm 3.3). Bu yöntem renk sayısı az olan ayırık MRA ları ve renk sayısı çok fazla olan sürekli MRA ları için uygundur [9].

Bu bölümün devamında, verilen bir dokudan SGLDM nin oluşturulması ve özelliklerin türetilmesi üzerinde durulacaktır.

5.3.1. SGLDM Oluşturulması ve Türetilen Özellikler

Uzamsal gri düzeyi bağıllık matrisi 'spatial gray level dependence matrix' (SGLDM) sayısal bir dokudan elde edilen $p(i,j,d,a)$ değerleri kullanılarak oluşturulur. Her bir $p(i,j,d,a)$, verilen iki nokta arası uzaklık, d , ve belirlenmiş bir a açısı yönünde, i renginden j rengine geçiş olan nokta çiftlerinin sayısıdır. Genellikle $a=\{0^\circ, 45^\circ, 90^\circ, 135^\circ\}$ ve $d=1$ olarak alınır. Eğer görüntü G tane renge sahipse, oluşturulan SGLDM bir $G \times G$ simetrik matris olur. Örneğin, dört farklı gri düzeyi içeren bir sayısal görüntü,

0	0	1	1
0	0	1	1
1	2	2	2
2	2	3	3

ve $d=1$, $a=0^\circ$ (yatay yönde) olsun, o zaman $P(i,j,1,0)$ lardan oluşan SGLDM matrisi,

	0	1	2	3
0	4	2	0	0
1	2	4	1	0
2	0	1	6	1
3	0	0	1	2

değerlerine sahip olacaktır.

Bu matristen özellik türetirken, i renginden j rengine geçiş olan (i,j) çiftlerinin olasılıklarının hesaplanması gerekmektedir. Eğer,

$$R = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} P(i,j,d,a) \quad \text{dersek,}$$

$$f(i,j,d,a) = P(i,j,d,a)/R, \quad (5.1)$$

bize bu olasılığı verir.

Bir SGLDM den bir çok dokusal özellik çıkarmak olasıdır. En uygun özellikleri çıkarmak ve seçmek için değişmez bir yol yoktur. Haralick [23], bu matrislerden kullanışlı dokusal özellikleri bulmak için kullanılabilir bir özellikler kümesi önermiştir.

Bu tez çalışmasında aşağıda verilecek olan beş özellik doku sınıflandırmada kullanılmaktadır.

Enerji (E), görüntünün homojenliğinin bir ölçüsüdür. Homojen görüntülerde farklı renklerden birbirine geçiş çok az olduğundan matris büyük değerlere sahip bir kaç elemandan oluşacaktır. Homojen olmayan görüntülerde ise matris, küçük değerli bir çok elemandan oluşacak ve dolayısıyla enerji özeliği (energy feature) küçük değerli olacaktır. 'Entropy' (ENT), görüntünün karma-

sıklığının (complexity) bir ölçüsüdür. Karmaşık bir (complex) görüntüde bu değer, basit (simple) bir görüntününkinden daha büyüktür. Atalet (inertia) (I), görüntüde bulunan yerel değişimlerin miktarının bir ölçüsüdür. Yerel Homojenlik (Local Homogeneity) (LH), komşu olan benzer renklerin derecesine bir ölçüdür. İlinti (correlation) (COR), görüntüdeki renklerin doğrusal bağımlılıklarının bir ölçüsüdür. μ_x ve σ_x matrisin satır toplamalarının, μ_y ve σ_y de sütun toplamalarının ortalaması ve standart sapmasıdır. Aşağıda bu özelliklerin hesaplanması için gerekli olan formüller verilmiştir.

$$E = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} [f(i,j,d,a)]^2, \quad (5.2)$$

$$ENT = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} [f(i,j,d,a) \ln[f(i,j,d,a)]], \quad (5.3)$$

$$I = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} [(i-j)^2 f(i,j,d,a)], \quad (5.4)$$

$$LH = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} [f(i,j,d,a)/(1+(i-j)^2)], \quad (5.5)$$

$$COR = \sum_{i=0}^{G-1} \sum_{j=0}^{G-1} [(i-\mu_x)(j-\mu_y)f(i,j,d,a)/(\sigma_x\sigma_y)], \quad (5.6)$$

$$\mu_x = \sum_{i=0}^{G-1} i \sum_{j=0}^{G-1} [f(i,j,d,a)], \quad (5.7)$$

$$\mu_y = \sum_{j=0}^{G-1} j \sum_{i=0}^{G-1} [f(i,j,d,a)], \quad (5.8)$$

$$\sigma_x = \sum_{i=0}^{G-1} (i-\mu_x)^2 \sum_{j=0}^{G-1} [f(i,j,d,a)], \quad (5.9)$$

$$\sigma_y = \sum_{j=0}^{G-1} (j-\mu_y)^2 \sum_{i=0}^{G-1} [f(i,j,d,a)]. \quad (5.10)$$

5.4. Doku Sınıflandırma Denemeleri

Aşağıda SGLDM den türetilmiş özellikler kullanılarak yapılan bir dizi deneme verilecektir. Bu denemelerde 64x64 boyutlarındaki yapay dokular ikişer ikişer karşılaştırılmaktadır. Yani, iki durumlu bir sınıflandırma yapılmaktadır. Her bir sınıf için 25 doku örneği alındı (görüntü tanımada özelliklerin sayısının enaz dört veya beş katı örnek alınması gerekmektedir [20]) ve her bir sınıfa ait dokular için dört ayrı yöndeki SGLDM lerden yirmi tane özellik (E, ENT, I, LH ve COR dan oluşan) hesaplandı ve bunlar o dokuya ait özellik vektörünü oluşturdu. Bu özelliklerin ortalamaları alındı ve bu ortalamaların uzamda iki ayrı sınıfa karşı düştüğü düşünüldü. Karar kuralı için Mahalanobis uzaklığı (Mahalanobis Distance) kullanıldı (EK A). Görsel olarak karar verebilme açısından özelliklerin iki boyutlu (2-B) izdüşümü bulundu (EK B).

Deneme 5.1 de görsel olarak birbirine benzeyen, Deneme 5.2 de görsel olarak birbirine benzese de uygun parametre seçimi yapılmış, Deneme 5.3 ve 5.4 te birbirinden farklı, Deneme 5.5 te de görsel olarak birbirine benzeyen fakat ayrı modellere ait dokuların sınıflandırılmasına çalışıldı.

Deneme 5.1 : 64x64 boyutlarında 4-renkli parametreleri (a) (1, 1, 1, 1) ve (b) (0.1, 0.1, 0.1, 0.1) olan iki ayrı sınıftan 2. algoritma kullanılarak Derin & Elliott modeline uygun dokular üretilip bunları sınıflandırıldı.

Şekil 5.2 de gösterilen bu dokuları göz ile ayırdetmek pek olanaklı değildir. Sınıflandırma işleminde örnek dokulardan hesaplanan ortalama özellik vektörleri bizim karar kriterimiz olmaktadır (Tablo 5.1).

Bu sınıflandırma denemesinde her bir dokuya ait kovaryans (covariance) matrisinin en büyük iki özdeğeri (a) için (0.0019054, 0.0150292); (b) için (0.004048, 0.015056) dir. Sınıflayıcıya önce birinci sınıfa ait olduğu bilinen dokular gönderildi ve hata olasılığı bulundu, $P(\epsilon | C_1)=8/50$. Sonra da ikinci sınıfa ait olduğu bilinen dokular gönderildi ve hata olasılığı bulundu, $P(\epsilon | C_2)=6/50$. Bu durumda bu sınıflayıcının bu iki dokuyu sınıflandırırken yaptığı hata olasılığı (sadece iki sınıf olduğu için, $P(w_1)=P(w_2)=1/2$),

$$P(\epsilon) = P(w_1)P(\epsilon | C_1) + P(w_2)P(\epsilon | C_2) \quad (5.11)$$

$$P(\epsilon) = (1/2)(8/50) + (1/2)(6/50)$$

$$P(\epsilon) = 7/50.$$

olarak hesaplandı. Şekil 5.3 bu dokulara ait özellik vektörlerinin 2-boyutlu izdüşümlerini göstermektedir. Parametreleri birbirine çok yakın olan (a-(1, 1, 1, 1) ve b-(0.5, 0.5, 0.5, 0.5)) iki doku için ise $P(\epsilon)=29/50$ olarak bulunmaktadır.

Deneme 5.2 : Bu denemede 64x64 boyutlarında 4-renkli parametreleri (a) (-1, 1, 1, 1) ve (b) (1, -1, 1, 1) olan iki ayrı sınıftan ikinci algoritma kullanılarak Derin & Elliott modeline uygun üretilen dokular (Şekil 5.4) sınıflandırıldı. Bulunan en büyük iki öz değer (a) için (0.008178, 0.045592) (b) için (0.005915, 0.037947).

Sınıflayıcıya birinci sınıfa ait olduğu bilinen dokular gönderildiğinde hata olasılığı, $P(\epsilon | C_1)=0/25$, ikinci sınıfa ait olduğu bilinen dokular gönderildiğinde hata olasılığı, $P(\epsilon | C_2)=0/25$, olarak bulundu. Bu

durumda bu sınıflayıcının bu iki dokuyu sınıflandırırken yaptığı hata olasılığı,

$$P(\epsilon) = P(w_1)P(\epsilon | C_1) + P(w_2)P(\epsilon | C_2)$$

$$P(\epsilon) = 0/25$$

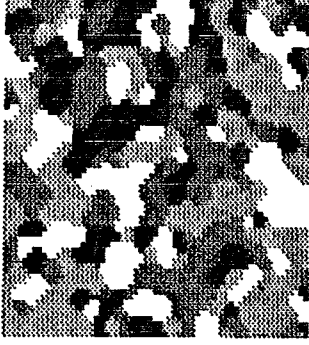
olur. Yani, bu iki doku görsel olarak birbirine benzese de rahatlıkla birbirinden ayırdedilebilir. Tablo 5.2 de bu iki dokuya ait ortalama özellik vektörleri, Şekil 5.5 te de bu özellik vektörlerinin 2-boyutlu izdüşümü gösterilmektedir.

Deneme 5.3 : Bu denemede de 64x64 boyutlarında 4-renkli parametreleri (a) (1, 1, 1, -1) ve (b) (1, 1, -1, 1) olan iki ayrı sınıftan ikinci algoritma kullanılarak Derin & Elliott modeline uygun üretilen dokular (Şekil 5.6) sınıflandırıldı. Bulunan en büyük iki öz değer (a) için (0.001716, 0.027671) (b) için (0.001757, 0.015994).

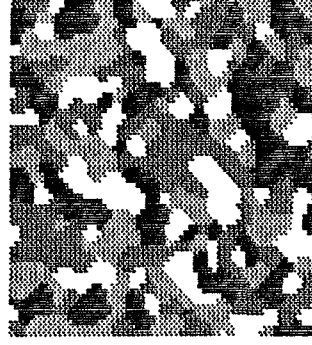
Bu denemede, $P(\epsilon | C_1)=0/25$ ve $P(\epsilon | C_2)=0/25$ olarak bulundu. Dolayısıyla da bu sınıflayıcı için $P(\epsilon)=0/25$ olarak hesaplandı. Tablo 5.3 te bu iki dokuya ait ortalama özellik vektörleri, Şekil 5.7 ta da bu özellik vektörlerinin 2-boyutlu izdüşümü verilmektedir.

Deneme 5.4 : Bu denemede ise 64x64 boyutlarında 4-renkli iki ayrı MRA modeline ait parametreleri (a) auto-binomial (-2, 0.4, 0.4, -0.4, 0.4) ve (b) Derin & Elliott (1, 1, -1, 1) olan iki ayrı modelden dokular (Şekil 5.8) sınıflandırıldı. Bulunan en büyük iki öz değer (a) için (0.001716, 0.027671) (b) için (0.001757, 0.015994).

Bu denemede, $P(\epsilon | C_1)=0/25$ ve $P(\epsilon | C_2)=0/25$ olarak bulundu. Dolayısıyla da bu sınıflayıcı için $P(\epsilon)=0/25$ olarak hesaplandı. Tablo 5.4 te bu iki dokuya ait ortalama özellik vektörleri, Şekil 5.9 da da bu özellik vektörlerinin 2-boyutlu izdüşümü verilmektedir.



(a)



(b)

Şekil 5.2 Görsel olarak birbirine benzeyen dokular.
(Deneme 5.1 için).

Tablo 5.1 Deneme 5.1 deki özellik vektörlerinin ortalaması.

(a)

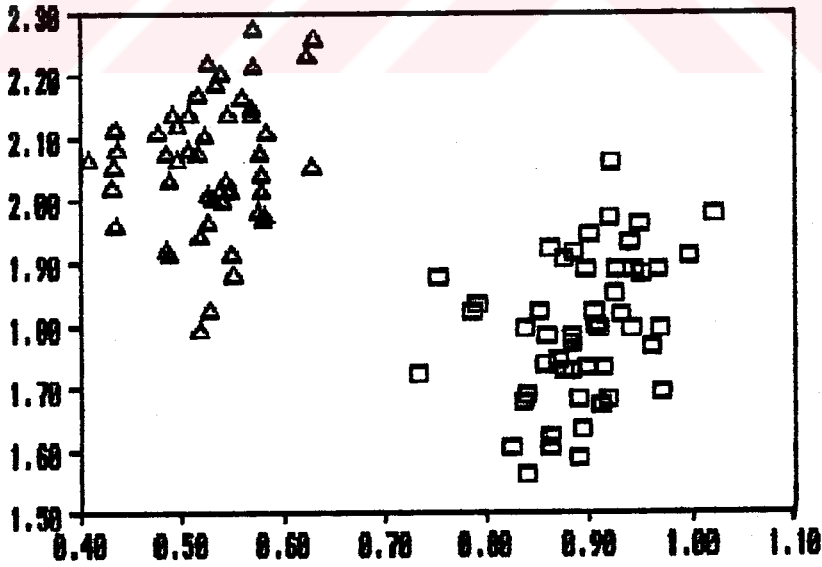
d=1 Aç1=0°	d=1 Aç1=45°
E.....: 0.17410060	E.....: 0.15199852
ENT...: 2.04765230	ENT...: 2.19477841
I.....: 0.60213294	I.....: 0.79585790
LH...: 0.87912401	LH...: 0.83880574
COR...: 0.75238675	COR...: 0.67255551

d=1 Aç1=90°	d=1 Aç1=135°
E.....: 0.17508707	E.....: 0.15196705
ENT...: 2.04107282	ENT...: 2.19417754
I.....: 0.58802579	I.....: 0.81004787
LH...: 0.88130853	LH...: 0.83827765
COR...: 0.75777342	COR...: 0.66689380

(b)

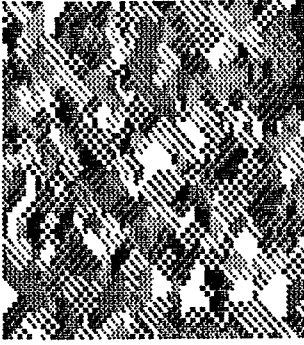
d=1 Aç1=0°	d=1 Aç1=45°
E.....: 0.16268234	E.....: 0.13871368
ENT...: 2.12064907	ENT...: 2.27756461
I.....: 0.69972222	I.....: 0.93482489
LH....: 0.86035317	LH....: 0.81331922
COR...: 0.71647970	COR...: 0.62162443

d=1 Aç1=90°	d=1 Aç1=135°
E.....: 0.16198502	E.....: 0.13819164
ENT...: 2.12507077	ENT...: 2.28033236
I.....: 0.70254960	I.....: 0.94032754
LH....: 0.85924901	LH....: 0.81191030
COR...: 0.71572529	COR...: 0.61917012

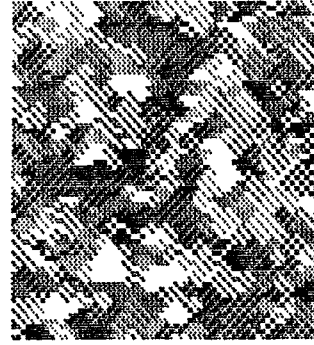


□ Δ

Şekil 5.3 Deneme 5.1 deki özellik vektörlerinin 2-boyutlu izdüşümü.



(a)



(b)

Şekil 5.4 Deneme 5.2 için üretilen dokular.

Tablo 5.2 Deneme 5.2 deki özellik vektörlerinin ortalaması.

(a)

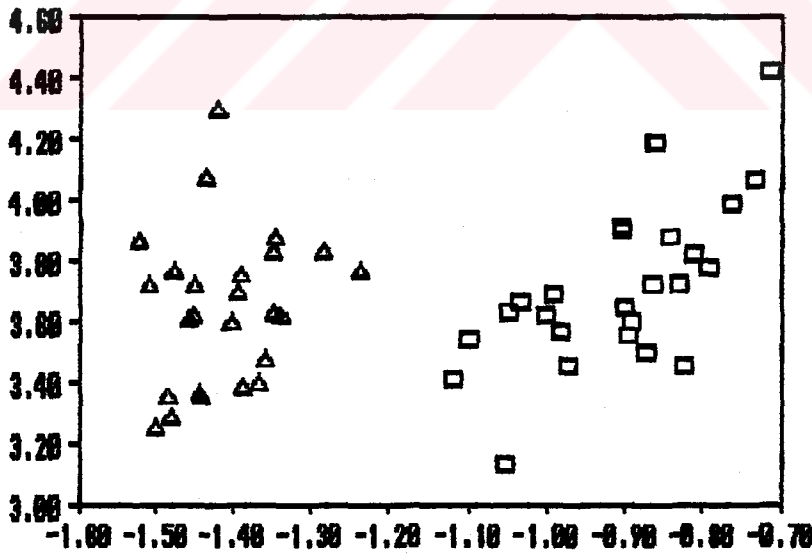
d=1 Aç1=0°	d=1 Aç1=45°
E....: 0.06400800	E....: 0.12282774
ENT...: 2.76010861	ENT...: 2.37607606
I....: 2.55891865	I....: 1.09363568
LH...: 0.48992163	LH...: 0.78061980
COR...: -0.02681582	COR...: 0.56138534

d=1 Aç1=90°	d=1 Aç1=135°
E....: 0.06690738	E....: 0.12269554
ENT...: 2.73996710	ENT...: 2.37687688
I....: 2.20892857	I....: 1.08670194
LH...: 0.56110714	LH...: 0.78065205
COR...: 0.11332107	COR...: 0.56421920

(b)

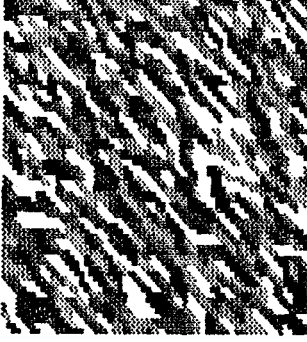
d=1 Aç1=0°	d=1 Aç1=45°
E.....: 0.06677509	E.....: 0.12361760
ENT...: 2.74000766	ENT...: 2.37136695
I.....: 2.19471230	I.....: 1.08527085
LH...: 0.55901290	LH...: 0.78209322
COR...: 0.11728118	COR...: 0.56359879

d=1 Aç1=90°	d=1 Aç1=135°
E.....: 0.06437888	E.....: 0.12377202
ENT...: 2.75730079	ENT...: 2.37044121
I.....: 2.55100198	I.....: 1.08429327
LH...: 0.48786806	LH...: 0.78253364
COR...: -0.02615473	COR...: 0.56388436

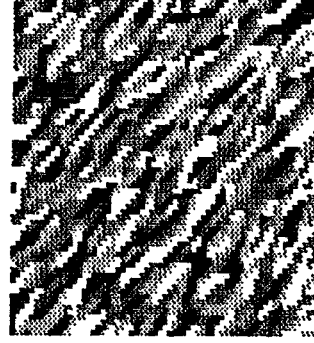


□ ■ Δ b

Şekil 5.5 Deneme 5.2 deki özellik vektörlerinin 2-boyutlu izdüşümü.



(a)



(b)

Şekil 5.6 Deneme 5.3 için üretilen dokular.

Tablo 5.3 Deneme 5.3 deki özellik vektörlerinin ortalaması.

(a)

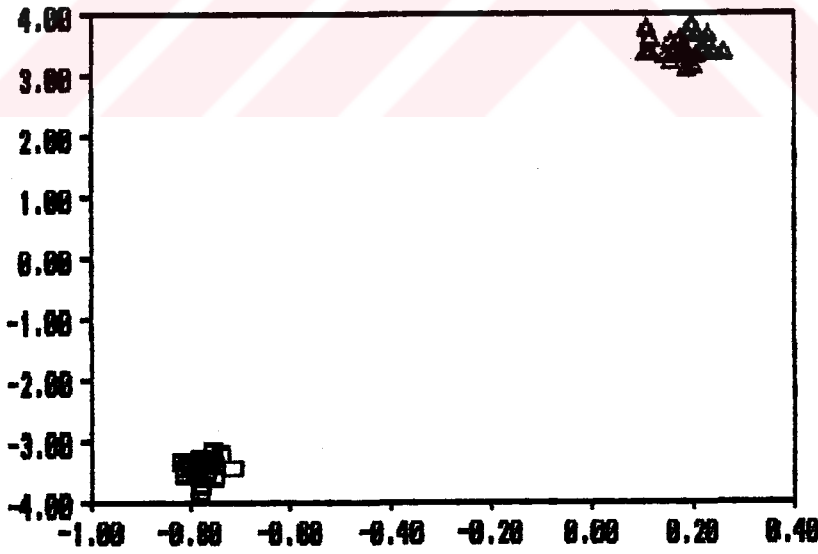
d=1 Aç1=0°	d=1 Aç1=45°
E....: 0.09506822	E....: 0.06425732
ENT...: 2.55254506	ENT...: 2.75796141
I....: 1.47252976	I....: 2.68244898
LH...: 0.70587798	LH...: 0.46476190
COR...: 0.41070390	COR...: -0.07321954

d=1 Aç1=90°	d=1 Aç1=135°
E....: 0.09457680	E....: 0.14122878
ENT...: 2.55550084	ENT...: 2.25555621
I....: 1.47886905	I....: 0.88510960
LH...: 0.70458929	LH...: 0.82245503
COR...: 0.40814167	COR...: 0.64571805

(b)

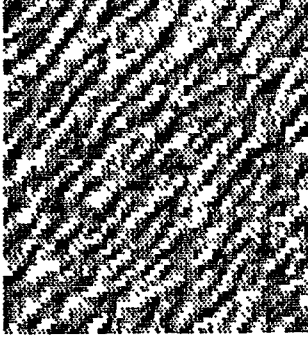
d=1 Aç1=0°	d=1 Aç1=45°
E....: 0.09447697	E....: 0.14135918
ENT...: 2.55647236	ENT...: 2.25471401
I....: 1.46501984	I....: 0.88276140
LH....: 0.70504960	LH....: 0.82275838
COR...: 0.40993279	COR...: 0.64447103

d=1 Aç1=90°	d=1 Aç1=135°
E....: 0.09539041	E....: 0.06410784
ENT...: 2.55053106	ENT...: 2.75905605
I....: 1.45771825	I....: 2.65642731
LH....: 0.70781944	LH....: 0.46687629
COR...: 0.41278098	COR...: -0.06962646

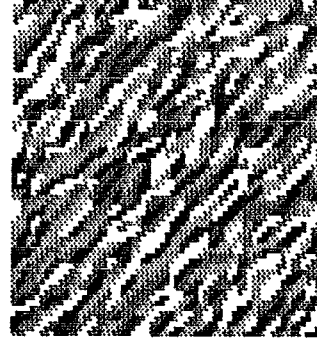


□ a Δ b

Şekil 5.7 Deneme 5.3 deki özellik vektörlerinin 2-boyutlu izdüşümü.



(a)



(b)

Şekil 5.8 Deneme 5.4 için Üretilen dokular.

Tablo 5.4 Deneme 5.4 deki özellik vektörlerinin ortalaması.

(a)

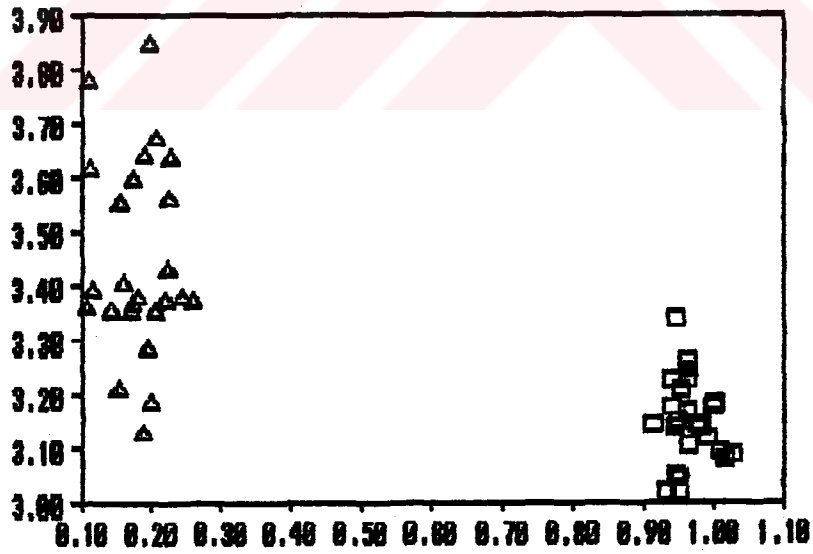
d=1 Aç1=0°	d=1 Aç1=45°
E.....: 0.07101115	E.....: 0.08681179
ENT...: 2.70146478	ENT...: 2.56456609
I.....: 1.59754960	I.....: 1.01519778
LH...: 0.59753472	LH...: 0.68297405
COR...: 0.36312507	COR...: 0.59539728

d=1 Aç1=90°	d=1 Aç1=135°
E.....: 0.07097890	E.....: 0.06545971
ENT...: 2.70160947	ENT...: 2.74826997
I.....: 1.59749008	I.....: 3.03590829
LH...: 0.59700496	LH...: 0.44298917
COR...: 0.36271762	COR...: -0.20988944

(b)

d=1 Aç1=0°	d=1 Aç1=45°
E....: 0.09447697	E....: 0.14135918
ENT...: 2.55647236	ENT...: 2.25471401
I....: 1.46501984	I....: 0.88276140
LH....: 0.70504960	LH....: 0.82275838
COR...: 0.40993279	COR...: 0.64447103

d=1 Aç1=90°	d=1 Aç1=135°
E....: 0.09539041	E....: 0.06410784
ENT...: 2.55053106	ENT...: 2.75905605
I....: 1.45771825	I....: 2.65642731
LH....: 0.70781944	LH....: 0.46687629
COR...: 0.41278098	COR...: -0.06962646



□ Δ

Şekil 5.9 Deneme 5.4 deki özellik vektörlerinin 2-boyutlu izdüşümü.

5.5 Özet

Bu bölümde, dokuları sınıflandırmak için dokusal özellik çıkarma metodundan bahsedildi. Bu dokusal özellikler kullanılarak 2 durumlu bir sınıflayıcı tasarlandı ve bu sistem kullanılarak çeşitli doku örnekleri sınıflandırılmaya çalışıldı. Yapılan deneyler sonunda, eğer parametreler uygun seçilmişse, bu sınıflayıcının bu dokuları büyük bir doğrulukla ait oldukları sınıflara atadığı görüldü. Birbirine yakın parametreler ile üretilmiş dokularda ise hata olasılığının arttığı görüldü.



BÖLÜM 6

SONUC ve ÖNERİLER

Markov rastlantı alanlarının uygulamalarında (doku sentezi, doku sınıflandırma, görüntü sıkıştırma, yenileme ve bölütleme) sıkça karşılaşılan, MRA örnekleme, MRA model parametrelerinin kestirimi, verilen bir doku ile bir MRA modeli arasındaki uygunluk testi problemleridir. Bu tezde bu konular üzerinde sistematik bir çalışma yapıldı. Varolan örnekleme algoritmalarından ikisi karşılaştırıldı, yine varolan uygunluk testlerinden ikisi tanıtıldı (Bölüm 3). Verilen bir dokuyu modelleyen parametreleri bulmak için kullanılan parametre kestirim yöntemlerinden MLE yöntemi ayrıntılı olarak açıklandı (Bölüm 5).

Karşılaştırılan birinci ve ikinci örnekleme algoritmalarının iki-renkli dokular için aynı sonucu verdiği görüldü. Fakat renk sayısı artığında büyük parametre değerleri için ikinci algoritmanın tek bir rengin hakim olduğu bir gerçeklemeye yakınsadığı gözlemlendi. (Birinci algoritmada Cross [12], doku örneklerken değiş-tokuş yöntemini kullanarak bu olayın önüne geçmiştir. Fakat bu sefer de MRA'nın pozitiflik özelliği göz ardı edilmektedir).

Bölüm 4 de anlatılan parametre kestirim yöntemlerinden Besag'ın 'coding' metoduna dayanan MLE, bu tez çalışmasında kullanıldı.

Bölüm 5 te dokusal özellikler kullanılarak dokular sınıflandırılmaya çalışıldı. Parametreleri uygun seçilen

dokuların büyük bir doğrulukla sınıflandığı, parametreleri birbirine yakın olan dokular için ise hata olasılığının büyüdüğü görüldü.

Varolan parametre kestirim yöntemleri çok-renkli dokulara uygulandığında büyük bellekli ve çok hızlı çalışan sistemlere ihtiyaç duyulmaktadır. Çünkü renk sayısı arttıkça, uygun parametre değerleri için hem görüntünün boyutlarının büyümesi hem de çözüm için elde edilen denklem sayılarının çok fazla olması gerekmektedir. Yeni ve doğru sonuç veren bir parametre kestirim yöntemi hala araştırılan bir konudur. Yine aynı şekilde birbirine yakın parametre değerlerine sahip dokuların da doğru bir şekilde sınıflandırılması istenmektedir.



Ek A

MAHALANOBIS UZAKLIĞI ve KARAR KURALI

C_i sınıfına ait N tane deneme vektörümüz olsun
 $\{V_1^{(i)}, V_2^{(i)}, \dots, V_N^{(i)}\}$, $1 \leq i \leq K$. i 'inci görüntü sınıfına ait ortalama vektör μ_i ve kovaryans 'covariance' matrisi Σ_i aşağıdaki gibi hesaplanır.

$$\mu_i = \frac{1}{N} \sum_{j=1}^N V_j^{(i)}$$

$$\Sigma_i = \frac{1}{N} \sum_{j=1}^N [V_j^{(i)} - \mu_i][V_j^{(i)} - \mu_i]^T \quad i=1,2,\dots,K$$

x , hangi sınıfa ait olduğu bilinmeyen, görüntü vektörü ve C_i sınıfına ait ortalama vektörü arasındaki Mahalanobis uzaklığı,

$$g_i(x) = (x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i) \quad i=1,2,\dots,K$$

olarak verilir. Bu uzaklığı göre bu tezde kullanılan iki durumlu sınıflayıcının karar kuralı aşağıdaki gibidir.

μ_1 ve μ_2 iki ayrı sınıfın ortalama vektörleri ve x de bu iki sınıftan birine ait olan görüntü vektörü olsun. Bu görüntünün her iki ortalama vektörüne olan Mahalanobis uzaklıkları,

$$g_1(x) = (x - \mu_1)^T \Sigma_1^{-1} (x - \mu_1)$$

$$g_2(x) = (x - \mu_2)^T \Sigma_2^{-1} (x - \mu_2)$$

olarak hesaplanırsın. Eğer,

$g_1(x) \leq g_2(x)$ ise x , C_1 sınıfına ait,
aksi ise C_2 sınıfına aittir.



EK B

İKİ BOYUTA İNDİRGEME

C_i sınıfına ait N tane deneme vektörünün (EK A) kovaryans 'covariance' matrisi Σ_i'inin özdeğerleri ve özvektörleri hesaplanır. En büyük iki özdeğere ait özvektörler, K boyuttan 2 boyuta indirgenecek vektörler için dönüşüm matrisini oluşturur. y₁ ve y₂, 2 boyutlu uzayımıza ait noktalar ise,

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} A_i \end{bmatrix}_{K \times 2}^T \begin{bmatrix} V_j^{(i)} \end{bmatrix}_{K \times 1}$$

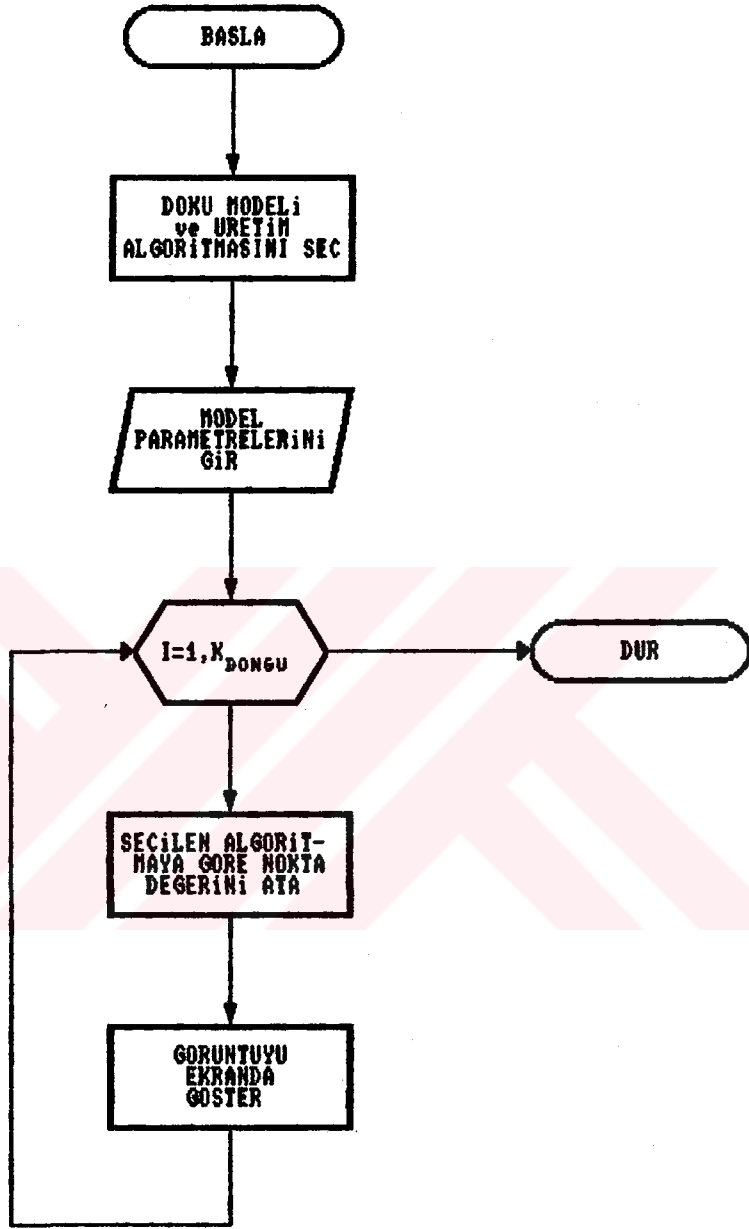
A_i : i. sınıfa ait dönüşüm matrisi.

V_j⁽ⁱ⁾ : i. sınıfa ait j. görüntü vektörü.

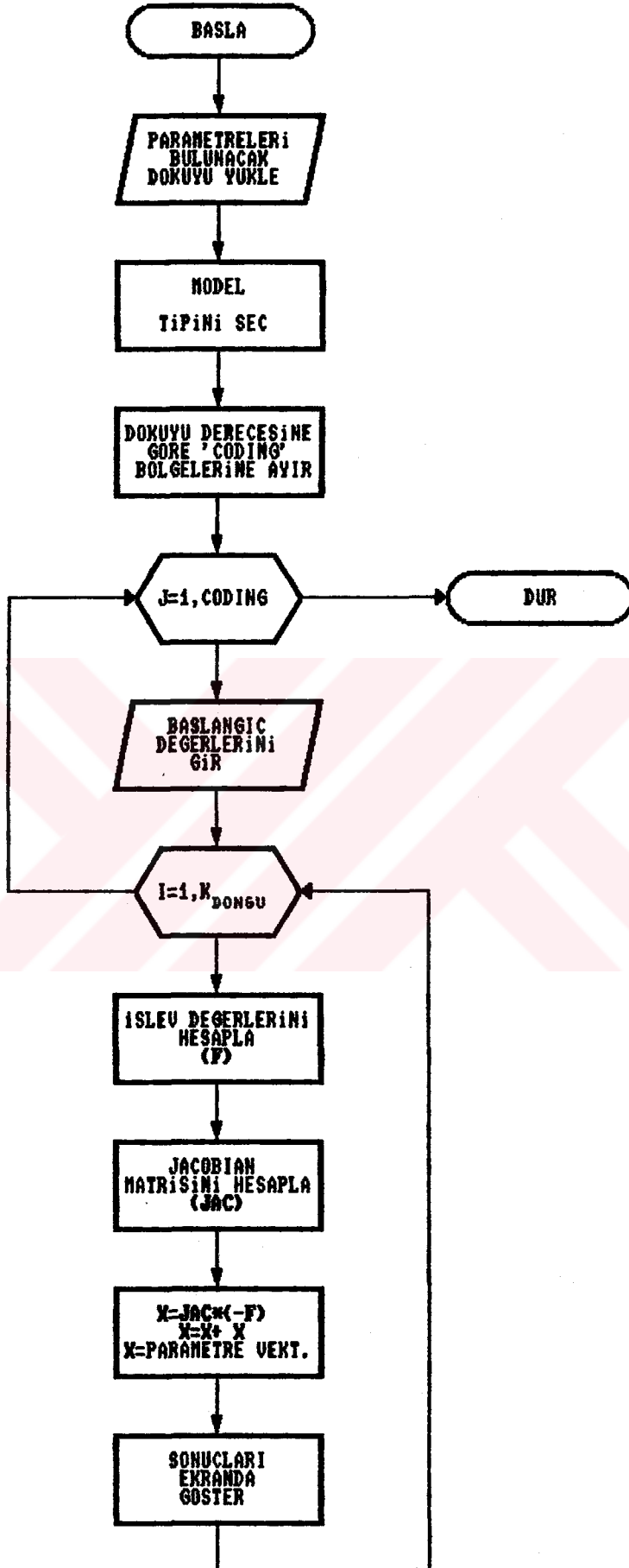
EK C



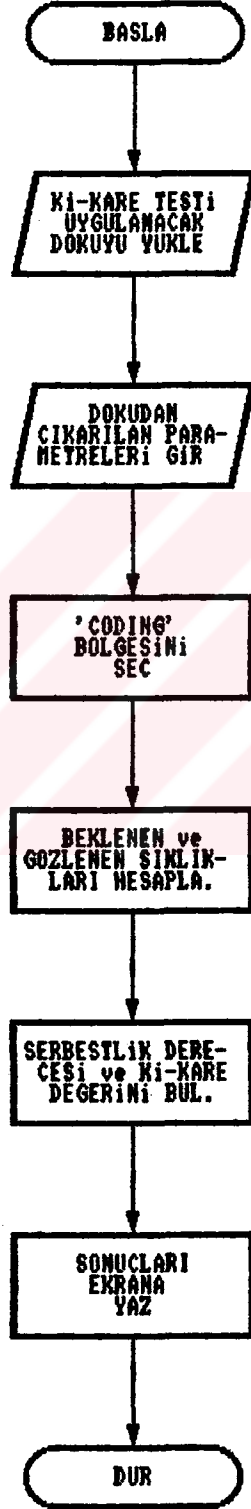
AKİS ÇİZELGELERİ

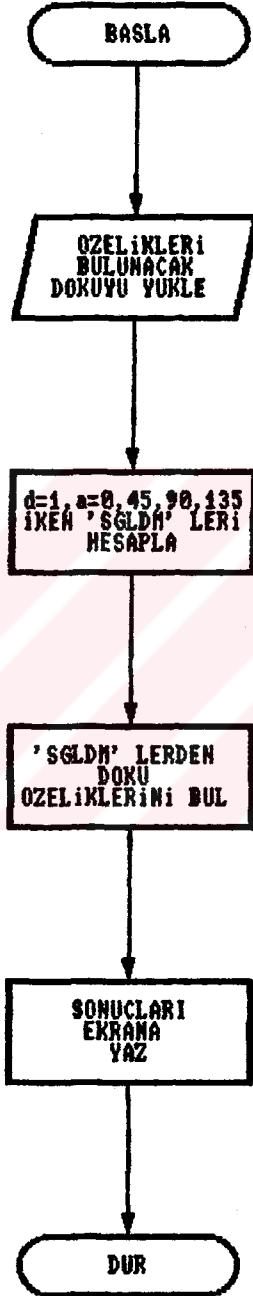


Doku Üretimi



Parametre Kestirimi





Özelik Çıkarma

E K D



PROGRAMLAR

```
/* Ana Program ve Doku Üretim Bölümü */
```

```
Program Regeneration_of_Textures;
```

```
{%N+}
```

```
{%N $8000,10000,64000}
```

```
uses crt,dos,printer,psrvbox,  
  menubox,Graph,setupbox,estbox,chibox,classify,srvbox;
```

```
Procedure Texture_Generation(var m_dot : m_lattice);  
  /* Doku Üretimi */
```

```
  var x,y,k,color,temp : integer;  
      u,t                : extended;  
      r, prob_x,prob_y,prob_x2,prob_y2 : extended;  
      Theta : extended;  
      s      : str2;  
      erd1,erd2,e1,e2:integer;  
      iteration,chan : str5;
```

```
Procedure take_value;
```

```
  /* Doku parameterelerinin girilmesi */  
  type ucl=array [1..12] of byte;  
  var ly,l : integer;  
      d     : double;  
  const len : ucl=(8,1,4,8,8,8,8,8,8,8,8,8);  
        y_pos: ucl=(9,10,11,13,14,15,16,17,18,19,20,21);
```

```
begin
```

```
  Screset;  
  Clear_Buffer;  
  Res(Model_Name+' Model',1);  
  Hid_Frame(23,5,58,13,'Parameters',Palet[8]);  
  PlaceStr(26,9,Dots('Number of Gray Level',23));  
  PlaceStr(26,10,Dots('Order Level',23));  
  PlaceStr(26,11,Dots('Number of Iteration',23));  
  l:=1;  
  spkeys(25);  
  Repeat  
    CON:=TRUE;  
    inputstr(BUFFER[l],len[l],50,y_pos[l],Term,TC);  
    Case l Of  
      1 : Intcon(CON,2,255,BUFFER[l],k);  
      2 : if sec=1 then Intcon(CON,1,4,BUFFER[l],k) else Intcon(CON,1,4,BUFFER[l],k);  
      3 : begin  
          Intcon(CON,1,9999,BUFFER[l],k);  
          if (TC='N') or (TC='X') Then  
            begin  
              store_scr;  
              val(BUFFER[2],k,1);  
              Mes_Frame(44,12,65,14+2*k,Palet[5]);  
              PlaceStr(47,13,Dots('s',8));
```

```

For i:=1 to k do
begin
  str(1,s);
  PlaceStr(47,12+2*i,Dots('b('+s+',1)',8));
  PlaceStr(47,13+2*i,Dots('b('+s+',2)',8));
end;
ly:=1;
Repeat
  CON:=TRUE;
  inputstr(BUFFER[3+ly],len[ly+3],56,y_pos[ly+3],Term,TC);
  Case ly Of
    1..8 : doublecon(CON,-10.0,10.0,BUFFER[3+ly],d);
  end;
  If CON Then Incdec(2*k+1,ly,TC);
  Until (TC='Z') OR (((TC='N') AND (ly=1)) AND CON);
  if TC='Z' Then begin Load_Scr; TC:=' '; end;
end;
end;

End;
If CON Then Incdec(3,L,TC);
Until (TC='Z') OR (((TC='N') AND (l=1)) AND CON);

if TC<>'Z' then
begin
  DEC(SCR_LEVEL);
  val(BUFFER[1],k,1); gray_level:=k;
  val(BUFFER[2],k,1); order:=k;
  val(BUFFER[3],k,1); iter:=k;
  val(BUFFER[4],d,1); a:=d;
  for i:=1 to order do
  begin
    val(BUFFER[3+2*i],d,k); b[i,1]:=d;
    val(BUFFER[4+2*i],d,k); b[i,2]:=d;
  end;
end;
end;
end;

```

Procedure Initial_Texture;

```

/* Başlangıç durumunun elde edilmesi */
var xx,yy : integer;
begin
  ini_zero:=0; ini_one:=0; ini_two:=0; ini_three:=0;
  ini_four:=0; ini_five:=0; ini_six:=0; ini_seven:=0;
  for yy:=-1 to 66 do
    for xx:=-1 to 66 do
      begin
        color:=random(gray_level);
        m_dot[xx,yy]:=color; { column, row }
        if (xx>0) and (xx<65) then
          if (yy>0) and (yy<65) then
            begin
              Put_Pixel(xx,yy,m_dot[xx,yy],100,120);
              if color=0 then inc(ini_zero);
              if color=1 then inc(ini_one);
              if color=2 then inc(ini_two);
              if color=3 then inc(ini_three);
            end;
          end;
        end;
      end;
    end;
  end;
end;

```

```

        if color=4 then inc(ini_four);
        if color=5 then inc(ini_five);
        if color=6 then inc(ini_six);
        if color=7 then inc(ini_seven);
    end;
end;
waitkey;
end;

```

```

Procedure Calculate_7(x1,y1 : integer);
/* Komşuluk yerleşmesine göre elde edilen değer */
var X1,X2,X3,X4,X5,X6,X7,X8 : integer;
    I1,I2 : integer;
begin
    case sec of
        1,2: begin
            X1:=m_dot[x1-1,y1]+m_dot[x1+1,y1];
            X2:=m_dot[x1,y1+1]+m_dot[x1,y1-1];
            X4:=m_dot[x1-1,y1+1]+m_dot[x1+1,y1-1];
            X3:=m_dot[x1+1,y1+1]+m_dot[x1-1,y1-1];
            X6:=m_dot[x1,y1+2]+m_dot[x1,y1-2];
            X5:=m_dot[x1-2,y1]+m_dot[x1+2,y1];
            X8:=m_dot[x1-1,y1+2]+m_dot[x1+2,y1-1]+m_dot[x1-2,y1+1]+m_dot[x1+1,y1-2];
            X7:=m_dot[x1+1,y1+2]+m_dot[x1-2,y1-1]+m_dot[x1+2,y1+1]+m_dot[x1-1,y1-2];
        end;
        3 : begin
            if m_dot[x1,y1]=m_dot[x1+1,y1] then I1:=-1 else I1:=1;
            if m_dot[x1,y1]=m_dot[x1-1,y1] then I2:=-1 else I2:=1;
            X1:=I1+I2;
            if m_dot[x1,y1]=m_dot[x1,y1+1] then I1:=-1 else I1:=1;
            if m_dot[x1,y1]=m_dot[x1,y1-1] then I2:=-1 else I2:=1;
            X2:=I1+I2;
            if m_dot[x1,y1]=m_dot[x1+1,y1-1] then I1:=-1 else I1:=1;
            if m_dot[x1,y1]=m_dot[x1-1,y1+1] then I2:=-1 else I2:=1;
            X4:=I1+I2;
            if m_dot[x1,y1]=m_dot[x1+1,y1+1] then I1:=-1 else I1:=1;
            if m_dot[x1,y1]=m_dot[x1-1,y1-1] then I2:=-1 else I2:=1;
            X3:=I1+I2;
            if m_dot[x1,y1]=m_dot[x1+2,y1] then I1:=-1 else I1:=1;
            if m_dot[x1,y1]=m_dot[x1-2,y1] then I2:=-1 else I2:=1;
            X6:=I1+I2;
            if m_dot[x1,y1]=m_dot[x1,y1+2] then I1:=-1 else I1:=1;
            if m_dot[x1,y1]=m_dot[x1,y1-2] then I2:=-1 else I2:=1;
            X5:=I1+I2;
            if m_dot[x1,y1]=m_dot[x1+1,y1+2] then I1:=-1 else I1:=1;
            if m_dot[x1,y1]=m_dot[x1-1,y1-2] then I2:=-1 else I2:=1;
            X7:=I1+I2;
            if m_dot[x1,y1]=m_dot[x1-1,y1+2] then I1:=-1 else I1:=1;
            if m_dot[x1,y1]=m_dot[x1+1,y1-2] then I2:=-1 else I2:=1;
            X8:=I1+I2;
        end;
    end;
end;

case order of
    1 : t:=a+b[1,1]*X1+b[1,2]*X2;
    2 : t:=a+b[1,1]*X1+b[1,2]*X2+b[2,1]*X3+b[2,2]*X4;

```

```

3 : t:=a+b[1,1]*X1+b[1,2]*X2+b[2,1]*X3+b[2,2]*X4+b[3,1]*X5+b[3,2]*X6;
4 : t:=a+b[1,1]*X1+b[1,2]*X2+b[2,1]*X3+b[2,2]*X4+b[3,1]*X5+b[3,2]*X6+b[4,1]*X7+b[4,2]*X8;
end;
end;

```

```
{Procedure MAIN}
```

```

begin
  Sub[1]:='Auto Binomial (Cross)';
  Sub[2]:='Auto Binomial (Chen)';
  Sub[3]:='Derin & Elliott';
  W_DIR:=True;
  sec:=0;
  MidMenu(sec,3,22,16,24,'Texture Model');
  if sec=0 then EXIT;
  if sec=1 then Model_Name:='Binom. (Cross)';
  if sec=2 then Model_Name:='Binom. (Chen)';
  if sec=3 then Model_Name:='Derin & Elliott';

  Take_Value;

  if TC<>'Z' then
    begin
      Change_to_Graphics;
      outtextxy(1,1,'Model Name_: '+Model_Name);
      outtextxy(1,10,'Gray Level_: '+BUFFER[1]);
      outtextxy(180,10,'Order_: '+BUFFER[2]);
      outtextxy(1,20,'Iteration_: '+BUFFER[3]);
      outtextxy(180,20,'a__ '+BUFFER[4]);

      for i:=1 to order do
        begin
          str(i,s);
          outtextxy(c1[i],c2[i], 'b['+s+',1]: '+BUFFER[3+2*i]);
          outtextxy(c1[i],c2[i]+10, 'b['+s+',2]: '+BUFFER[4+2*i]);
        end;

      Randomize;

      Initial_Texture;

      for i:=1 to iter do
        begin
          changes[i]:=0; sifir[i]:=0; bir[i]:=0; iki[i]:=0;
          uc[i]:=0; dort[i]:=0; bes[i]:=0; alti[i]:=0; yedi[i]:=0;
        end;

      for k:=1 to iter do
        begin
          str(k:2,iteration);
          for erd2:=60 to 70 do
            for erd1:=1 to 20 do putpixel(erd1,erd2,0);
            outtextxy(1,60,iteration+' . iteration');
            for y:=1 to 64 do
              for x:=1 to 64 do
                begin

```

```

Calculate_T(x,y);
temp:=m_dot[x,y];
case sec of
1 : begin /* Auto-Binomial (Cross) */
    color:=m_dot[x,y];
    while color=m_dot[x,y] do
        begin
            e1:=random(65);
            e2:=random(65);
            color:=m_dot[e1,e2];
        end;
{x1/t1} prob_x:=conb(gray_level-1,m_dot[x,y])*exp(m_dot[x,y]*(T)); {first state}
{x2/t1} prob_y:=conb(gray_level-1,color)*exp(color*(T)); {next state}
    Calculate_T(e1,e2);
{x1/t2} prob_y2:=conb(gray_level-1,m_dot[x,y])*exp(m_dot[x,y]*(T)); {next state}
{x2/t2} prob_x2:=conb(gray_level-1,color)*exp(color*(T)); {first state}
    r:=(prob_y*prob_y2)/(prob_x*prob_x2);
    if r>=1.00 then
        begin
            m_dot[x,y]:=color; m_dot[e1,e2]:=temp;
        end else begin
            u:=random;
            if r>u then
                begin
                    m_dot[x,y]:=color; m_dot[e1,e2]:=temp;
                end;
            end;
        Put_Pixel(e1,e2,m_dot[e1,e2],400,120);
    end;

2 : begin /* Auto-Binomial (Chen) */
    color:=m_dot[x,y];
    while color=m_dot[x,y] do color:=random(gray_level);
    if gray_level<>2 then
        begin
            prob_x:=Conb(Gray_level-1,m_dot[x,y]);
            prob_y:=Conb(Gray_level-1,color);
            r:=(prob_y/prob_x)*exp(T*(color-m_dot[x,y]));
        end
    else begin
        r:=exp(T*(color-m_dot[x,y]));
    end;
    if r>=1.00 then m_dot[x,y]:=color
    else begin
        u:=random;
        if r>u then m_dot[x,y]:=color;
    end;
    end;

3 : begin /* Derin & Elliott */
    color:=random(gray_level);
    m_dot[x,y]:=color;
    prob_x:=T;
    Calculate_T(x,y);
    m_dot[x,y]:=temp;
    r:=exp(T-prob_x);
    if r>1.00 then r:=1.00;

```

```

        if r<=1.00 then
            begin
                u:=random;
                if r<=u then m_dot[x,y]:=color;
            end;
        end;
    end;
    Put_Pixel(x,y,m_dot[x,y],400,120);
    if m_dot[x,y]=0 then inc(sifir[k]);
    if m_dot[x,y]=1 then inc(bir[k]);
    if m_dot[x,y]=2 then inc(iki[k]);
    if m_dot[x,y]=3 then inc(uc[k]);
    if m_dot[x,y]=4 then inc(dort[k]);
    if m_dot[x,y]=5 then inc(bes[k]);
    if m_dot[x,y]=6 then inc(alti[k]);
    if m_dot[x,y]=7 then inc(yedi[k]);

    if temp<>m_dot[x,y] then inc(changes[k]);
end;
{ if changes[k]<205 then k:=iter;}
end;
outtextxy(80,70,'The End of Generation');
waitkey;
closegraph;
Show_Number_Of_Gray_Value;
end;
end;

```

```
{ MAIN }
```

```

begin
    Init_Screen;
    Hide_Cursor;
    sec:=0;
    f_name:='';
    Screset; LX:=0; LL:=1;
    DetectGraph(GraphDriver,GraphNode);

    Repeat
        tc:=' '; HEP:=0;
        window(1,1,80,25);
        HIDE_CURSOR;
        MX[1]:='Show Screen Status';
        MX[2]:='Setup Graphics Mode';
        MX[3]:='Texture Generation';
        MX[4]:='Parameter Estimation';
        MX[5]:='File Processing';
        MX[6]:='Chi Square Test';
        MX[7]:='Feature Extraction';

        CON:=TRUE;
        if LX=0 then HainMenu(LL,7,'Texture Regeneration','Designed by E. CANGIOGLU')
            else LL:=LX;
    
```

Case LL of

```

1 : Show_Screen_Status(GraphDriver,GraphMode);
    /* Ekranın grafik özelliklerini gösterir. */
2 : Setup_Graphics(GraphDriver,GraphMode);
    /* Çeşitli grafik ekran tiplerinin seçimi */
3 : Texture_Generation(Markov_Dots);
    /* Doku Üretimi */
4 : Parameter_Estimation(Markov_Dots);
    /* Parametre Kestirimi */
5 : File_Processing(Markov_Dots);
    /* Dosya işlemleri */
6 : Chi_Square_Test(Markov_Dots);
    /* Ki-Kare Testi */
7 : Feature_Extraction(Markov_Dots);
    /* Özellik Çıkarımı */
end;
Until LL=0;
Show_Cursor;
screset;
end.

```

```
/* Parametre Kestirim Bölümü */
```

```
unit EstBox;
```

```
{N+}
```

```
interface
```

```
uses crt,psrvbox,menubox,SrvBox;
```

```
procedure Parameter_Estimation(m_dot : m_lattice);
```

IMPLEMENTATION

```
Procedure Parameter_Estimation(m_dot : m_lattice);
```

```
/* Parametre Kestirimi */
```

```
type besli = array [1..5] of extended;
```

```
  d_besli = array [1..5] of besli;
```

```
var a_jac,temp      : d_besli;
```

```
  b_est,f,delta,ave : besli;
```

```
  max_iter,m_din,
```

```
  max_code,m2,code,k : integer;
```

```
Procedure Fonksiyon_first_zero;
```

```
var x,y,X1,X2,e1,e2,i1,i2,i,j : integer;
```

```
  Z1,DUNNY,DUNNY_JAC,Sun_Y : extended;
```

```
  s_ : array [0..64] of array [1..4] of integer;
```

```
  Y_ : array [0..64] of extended;
```

```
  DUN,dm_ : besli;
```

```
  dun_ : d_besli;
```

```
begin
```

```
  F[1]:=0.0; F[2]:=0.0; F[3]:=0.0;
```

```
  for y:=1 to m_din do
```

```
    for x:=1 to m_din do a_jac[y,x]:=0.0;
```

```
  for y:=1 to 64 do
```

```
    for x:=1 to 64 do
```

```
      begin
```

```
        case m2 of
```

```
          1 : begin
```

```
            X1:=(m_dot[x+1,y]+m_dot[x-1,y]);
```

```
            X2:=(m_dot[x,y+1]+m_dot[x,y-1]);
```

```
            Z1:=exp(b_est[1]+b_est[2]*X1+b_est[3]*X2);
```

```
            e2:=gray_level-1;
```

```
            e1:=m_dot[x,y]-e2;
```

```
            DUNNY:=(m_dot[x,y]+e1*Z1)/(1+Z1);
```

```
            DUNNY_JAC:=(e2*Z1/sqr(1+Z1));
```

```
            F[1]:=F[1]+DUNNY;
```

```
            F[2]:=F[2]+X1*DUNNY;
```

```
            F[3]:=F[3]+X2*DUNNY;
```

```
            a_jac[1,1]:=a_jac[1,1]-DUNNY_JAC;
```

```
            a_jac[2,2]:=a_jac[2,2]-sqr(X1)*DUNNY_JAC;
```

```
            a_jac[3,3]:=a_jac[3,3]-sqr(X2)*DUNNY_JAC;
```

```
            a_jac[1,2]:=a_jac[1,2]-X1*DUNNY_JAC;
```

```
            a_jac[1,3]:=a_jac[1,3]-X2*DUNNY_JAC;
```

```

a_jac[2,1]:=a_jac[1,2];
a_jac[3,1]:=a_jac[1,3];
a_jac[2,3]:=a_jac[2,3]-X1*X2*DUNNY_JAC;
a_jac[3,2]:=a_jac[2,3];
end;
2 : begin
  if m_dot[x,y]=m_dot[x+1,y] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x-1,y] then I2:=-1 else I2:=1;
  X1:=I1+I2;
  if m_dot[x,y]=m_dot[x,y+1] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x,y-1] then I2:=-1 else I2:=1;
  X2:=I1+I2;
  for i:=0 to gray_level-1 do
    begin
      if m_dot[x+1,y]=1 then I1:=-1 else I1:=1;
      if m_dot[x-1,y]=1 then I2:=-1 else I2:=1;
      s[i,1]:=I1+I2;
      if m_dot[x,y+1]=1 then I1:=-1 else I1:=1;
      if m_dot[x,y-1]=1 then I2:=-1 else I2:=1;
      s[i,2]:=I1+I2;
    end;
  for i:=0 to gray_level-1 do
    begin
      dunny:=(b_est[1]*s[i,1]+b_est[2]*s[i,2]);
      T[i]:=exp(-dunny);
    end;
  SUN_T:=0;
  for i:=0 to gray_level-1 do SUN_T:=SUN_T+T[i];

  for i:=1 to m_dim do begin dum[i]:=0.0; DM[i]:=0.0; end;

  for i:=1 to m_dim do
    for j:=1 to m_dim do dum[i,j]:=0.0;

  for i:=0 to gray_level-1 do
    begin
      DUN[1]:=DUN[1]+((s[i,1]-X1)*T[i]);
      DUN[2]:=DUN[2]+((s[i,2]-X2)*T[i]);
    end;
  F[1]:=F[1]+DUN[1]/SUN_T;
  F[2]:=F[2]+DUN[2]/SUN_T;

  for i:=0 to gray_level-1 do
    begin
      DUN[i,1]:=DUN[i,1]-(s[i,1]*(s[i,1]-X1)*T[i]);
      DUN[i,2]:=DUN[i,2]-(s[i,2]*(s[i,1]-X1)*T[i]);

      DUN[i,2,1]:=DUN[i,2,1]-(s[i,1]*(s[i,2]-X2)*T[i]);
      DUN[i,2,2]:=DUN[i,2,2]-(s[i,2]*(s[i,2]-X2)*T[i]);
    end;

  for i:=0 to gray_level-1 do
    begin
      DM[i]:=DM[i]-(s[i,1]*T[i]);
      DM[i,2]:=DM[i,2]-(s[i,2]*T[i]);
    end;

```

```

a_jac[1,1]:=a_jac[1,1]+(DUN[1,1]*Sun_T-(DUN[1]*DM[1]))/sqr(SUN_T);
a_jac[1,2]:=a_jac[1,2]+(DUN[1,2]*Sun_T-(DUN[1]*DM[2]))/sqr(SUN_T);

a_jac[2,1]:=a_jac[2,1]+(DUN[2,1]*Sun_T-(DUN[2]*DM[1]))/sqr(SUN_T);
a_jac[2,2]:=a_jac[2,2]+(DUN[2,2]*Sun_T-(DUN[2]*DM[2]))/sqr(SUN_T);
end;
end;
end;
end;

```

Procedure Fonksiyon_first;

```

var x,y,X1,X2,e1,e2,t2,I1,I2,i,j : integer;
    Z1,DUNNY,DUNNY_JAC,Sun_T : extended;
    s_ : array [0..64] of array [1..4] of integer;
    T_ : array [0..64] of extended;
    DUN,dm_ : besli;
    dum_ : d_besli;

```

begin

```

F[1]:=0.0; F[2]:=0.0; F[3]:=0.0;
for y:=1 to m_dim do
  for x:=1 to m_dim do a_jac[y,x]:=0.0;
  for y:=1 to 64 do
    for x:=1 to 64 do
      begin
        if (code=1) then if (y mod 2)=1 then t2:=1 else t2:=0;
        if (code=2) then if (y mod 2)=1 then t2:=0 else t2:=1;
        if (x mod 2)=t2 then
          begin
            case m2 of
              1 : begin
                X1:=(m_dot[x+1,y]+m_dot[x-1,y]);
                X2:=(m_dot[x,y+1]+m_dot[x,y-1]);
                Z1:=exp(b_est[1]+b_est[2]*X1+b_est[3]*X2);
                e2:=gray_level-1;
                e1:=m_dot[x,y]-e2;
                DUNNY:=(m_dot[x,y]+e1*Z1)/(1+Z1);
                DUNNY_JAC:=(e2*Z1/sqr(1+Z1));
                F[1]:=F[1]+DUNNY;
                F[2]:=F[2]+X1*DUNNY;
                F[3]:=F[3]+X2*DUNNY;
                a_jac[1,1]:=a_jac[1,1]-DUNNY_JAC;
                a_jac[2,2]:=a_jac[2,2]-sqr(X1)*DUNNY_JAC;
                a_jac[3,3]:=a_jac[3,3]-sqr(X2)*DUNNY_JAC;
                a_jac[1,2]:=a_jac[1,2]-X1*DUNNY_JAC;
                a_jac[1,3]:=a_jac[1,3]-X2*DUNNY_JAC;
                a_jac[2,1]:=a_jac[1,2];
                a_jac[3,1]:=a_jac[1,3];
                a_jac[2,3]:=a_jac[2,3]-X1*X2*DUNNY_JAC;
                a_jac[3,2]:=a_jac[2,3];
              end;
            end;
          end;
        if m_dot[x,y]=m_dot[x+1,y] then I1:=-1 else I1:=1;
        if m_dot[x,y]=m_dot[x-1,y] then I2:=-1 else I2:=1;
        X1:=I1+I2;
        if m_dot[x,y]=m_dot[x,y+1] then I1:=-1 else I1:=1;

```

```

if m_dot[x,y]=m_dot[x,y-1] then I2:=-1 else I2:=1;
X2:=I1+I2;
for i:=0 to gray_level-1 do
begin
  if m_dot[x+1,y]=1 then I1:=-1 else I1:=1;
  if m_dot[x-1,y]=1 then I2:=-1 else I2:=1;
  s_[i,1]:=I1+I2;
  if m_dot[x,y+1]=1 then I1:=-1 else I1:=1;
  if m_dot[x,y-1]=1 then I2:=-1 else I2:=1;
  s_[i,2]:=I1+I2;
end;
for i:=0 to gray_level-1 do
begin
  dummy:=(b_est[1]*s_[i,1]+b_est[2]*s_[i,2]);
  T_[i]:=exp(-dummy);
end;
SUM_T:=0;
for i:=0 to gray_level-1 do SUM_T:=SUM_T+T_[i];

for i:=1 to m_dim do begin dum[i]:=0.0; DH_[i]:=0.0; end;

for i:=1 to m_dim do
  for j:=1 to m_dim do dum_[i,j]:=0.0;

for i:=0 to gray_level-1 do
begin
  DUN[1]:=DUN[1]+((s_[i,1]-X1)*T_[i]);
  DUN[2]:=DUN[2]+((s_[i,2]-X2)*T_[i]);
end;
F[1]:=F[1]+DUN[1]/SUM_T;
F[2]:=F[2]+DUN[2]/SUM_T;

for i:=0 to gray_level-1 do
begin
  DUN_[1,1]:=DUN_[1,1]-(s_[i,1]*(s_[i,1]-X1)*T_[i]);
  DUN_[1,2]:=DUN_[1,2]-(s_[i,2]*(s_[i,1]-X1)*T_[i]);

  DUN_[2,1]:=DUN_[2,1]-(s_[i,1]*(s_[i,2]-X2)*T_[i]);
  DUN_[2,2]:=DUN_[2,2]-(s_[i,2]*(s_[i,2]-X2)*T_[i]);
end;

for i:=0 to gray_level-1 do
begin
  DH_[1]:=DH_[1]-(s_[i,1]*T_[i]);
  DH_[2]:=DH_[2]-(s_[i,2]*T_[i]);
end;

a_jac[1,1]:=a_jac[1,1]+(DUN_[1,1]*SUM_T-(DUN[1]*DH_[1]))/sqr(SUM_T);
a_jac[1,2]:=a_jac[1,2]+(DUN_[1,2]*SUM_T-(DUN[1]*DH_[2]))/sqr(SUM_T);

a_jac[2,1]:=a_jac[2,1]+(DUN_[2,1]*SUM_T-(DUN[2]*DH_[1]))/sqr(SUM_T);
a_jac[2,2]:=a_jac[2,2]+(DUN_[2,2]*SUM_T-(DUN[2]*DH_[2]))/sqr(SUM_T);
end;
end;
end;
end;
end;
end;

```

```

Procedure Fonksiyon_second_zero;
var x,y,X1,X2,X3,X4,i,j,e1,e2,l1,l2 : integer;
    Z1,DUNNY,DUNNY_JAC,Sum_T : extended;
    a_ : array [0..64] of array [1..4] of integer;
    T_ : array [0..64] of extended;
    DUN,dm_ : besli;
    dum_ : d_besli;

begin
  for y:=1 to m_dim do F[y]:=0.0;
  for y:=1 to m_dim do
    for x:=1 to m_dim do a_jac[y,x]:=0.0;
  for y:=1 to 64 do
    for x:=1 to 64 do
      begin
        case m2 of
          1 : begin
            X1:=(m_dot[x+1,y]+m_dot[x-1,y]);
            X2:=(m_dot[x,y+1]+m_dot[x,y-1]);
            X4:=(m_dot[x-1,y+1]+m_dot[x+1,y-1]);
            X3:=(m_dot[x+1,y+1]+m_dot[x-1,y-1]);
            Z1:=exp(b_est[1]+b_est[2]*X1+b_est[3]*X2+b_est[4]*X3+b_est[5]*X4);
            e2:=gray_level-1;
            e1:=m_dot[x,y]-e2;
            DUNNY:=(m_dot[x,y]+e1*Z1)/(1+Z1);
            DUNNY_JAC:=(e2*Z1/sqr(1+Z1));
            F[1]:=F[1]+DUNNY;
            F[2]:=F[2]+X1*DUNNY;
            F[3]:=F[3]+X2*DUNNY;
            F[4]:=F[4]+X3*DUNNY;
            F[5]:=F[5]+X4*DUNNY;
            a_jac[1,1]:=a_jac[1,1]-DUNNY_JAC;
            a_jac[2,2]:=a_jac[2,2]-sqr(X1)*DUNNY_JAC;
            a_jac[3,3]:=a_jac[3,3]-sqr(X2)*DUNNY_JAC;
            a_jac[4,4]:=a_jac[4,4]-sqr(X3)*DUNNY_JAC;
            a_jac[5,5]:=a_jac[5,5]-sqr(X4)*DUNNY_JAC;
            a_jac[1,2]:=a_jac[1,2]-X1*DUNNY_JAC;
            a_jac[1,3]:=a_jac[1,3]-X2*DUNNY_JAC;
            a_jac[1,4]:=a_jac[1,4]-X3*DUNNY_JAC;
            a_jac[1,5]:=a_jac[1,5]-X4*DUNNY_JAC;
            a_jac[2,1]:=a_jac[1,2];
            a_jac[3,1]:=a_jac[1,3];
            a_jac[4,1]:=a_jac[1,4];
            a_jac[5,1]:=a_jac[1,5];
            a_jac[2,3]:=a_jac[2,3]-X1*X2*DUNNY_JAC;
            a_jac[3,2]:=a_jac[2,3];
            a_jac[2,4]:=a_jac[2,4]-X1*X3*DUNNY_JAC;
            a_jac[4,2]:=a_jac[2,4];
            a_jac[2,5]:=a_jac[2,5]-X1*X4*DUNNY_JAC;
            a_jac[5,2]:=a_jac[2,5];
            a_jac[3,4]:=a_jac[3,4]-X2*X3*DUNNY_JAC;
            a_jac[4,3]:=a_jac[3,4];
            a_jac[3,5]:=a_jac[3,5]-X2*X4*DUNNY_JAC;
            a_jac[5,3]:=a_jac[3,5];
            a_jac[4,5]:=a_jac[4,5]-X3*X4*DUNNY_JAC;
            a_jac[5,4]:=a_jac[4,5];
          end;
        end;
      end;
    end;
  end;
end;

```

```

2 : begin
  if m_dot[x,y]=m_dot[x+1,y] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x-1,y] then I2:=-1 else I2:=1;
  X1:=I1+I2;
  if m_dot[x,y]=m_dot[x,y+1] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x,y-1] then I2:=-1 else I2:=1;
  X2:=I1+I2;
  if m_dot[x,y]=m_dot[x+1,y-1] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x-1,y+1] then I2:=-1 else I2:=1;
  X4:=I1+I2;
  if m_dot[x,y]=m_dot[x+1,y+1] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x-1,y-1] then I2:=-1 else I2:=1;
  X3:=I1+I2;
  for i:=0 to gray_level-1 do
    begin
      if m_dot[x+1,y]=i then I1:=-1 else I1:=1;
      if m_dot[x-1,y]=i then I2:=-1 else I2:=1;
      s[i,1]:=I1+I2;
      if m_dot[x,y+1]=i then I1:=-1 else I1:=1;
      if m_dot[x,y-1]=i then I2:=-1 else I2:=1;
      s[i,2]:=I1+I2;
      if m_dot[x+1,y-1]=i then I1:=-1 else I1:=1;
      if m_dot[x-1,y+1]=i then I2:=-1 else I2:=1;
      s[i,4]:=I1+I2;
      if m_dot[x+1,y+1]=i then I1:=-1 else I1:=1;
      if m_dot[x-1,y-1]=i then I2:=-1 else I2:=1;
      s[i,3]:=I1+I2;
    end;
  for i:=0 to gray_level-1 do
    begin
      dummy:=(b_est[1]*s[i,1]+b_est[2]*s[i,2]+b_est[3]*
        s[i,3]+b_est[4]*s[i,4]);
      T[i]:=exp(-dummy);
    end;
  SUN_T:=0;
  for i:=0 to gray_level-1 do SUN_T:=SUN_T+T[i];

  for i:=1 to m_dim do begin dum[i]:=0.0; DH[i]:=0.0; end;

  for i:=1 to m_dim do
    for j:=1 to m_dim do dum[i,j]:=0.0;

  for i:=0 to gray_level-1 do
    begin
      DUN[1]:=DUN[1]+((s[i,1]-X1)*T[i]);
      DUN[2]:=DUN[2]+((s[i,2]-X2)*T[i]);
      DUN[3]:=DUN[3]+((s[i,3]-X3)*T[i]);
      DUN[4]:=DUN[4]+((s[i,4]-X4)*T[i]);
    end;
  F[1]:=F[1]+DUN[1]/SUN_T;
  F[2]:=F[2]+DUN[2]/SUN_T;
  F[3]:=F[3]+DUN[3]/SUN_T;
  F[4]:=F[4]+DUN[4]/SUN_T;

  for i:=0 to gray_level-1 do
    begin
      DUN[i,1]:=DUN[i,1]-(s[i,1]*(s[i,1]-X1)*T[i]);

```

```

DUN_1,2]:=DUN_1,2)-(s_1,2)*(s_1,1)-X1)*T_1));
DUN_1,3]:=DUN_1,3)-(s_1,3)*(s_1,1)-X1)*T_1));
DUN_1,4]:=DUN_1,4)-(s_1,4)*(s_1,1)-X1)*T_1));

DUN_2,1]:=DUN_2,1)-(s_1,1)*(s_1,2)-X2)*T_1));
DUN_2,2]:=DUN_2,2)-(s_1,2)*(s_1,2)-X2)*T_1));
DUN_2,3]:=DUN_2,3)-(s_1,3)*(s_1,2)-X2)*T_1));
DUN_2,4]:=DUN_2,4)-(s_1,4)*(s_1,2)-X2)*T_1));

DUN_3,1]:=DUN_3,1)-(s_1,1)*(s_1,3)-X3)*T_1));
DUN_3,2]:=DUN_3,2)-(s_1,2)*(s_1,3)-X3)*T_1));
DUN_3,3]:=DUN_3,3)-(s_1,3)*(s_1,3)-X3)*T_1));
DUN_3,4]:=DUN_3,4)-(s_1,4)*(s_1,3)-X3)*T_1));

DUN_4,1]:=DUN_4,1)-(s_1,1)*(s_1,4)-X4)*T_1));
DUN_4,2]:=DUN_4,2)-(s_1,2)*(s_1,4)-X4)*T_1));
DUN_4,3]:=DUN_4,3)-(s_1,3)*(s_1,4)-X4)*T_1));
DUN_4,4]:=DUN_4,4)-(s_1,4)*(s_1,4)-X4)*T_1));
end;

for i:=0 to gray_level-1 do
begin
  DN_1]:=DN_1)-(s_1,1)*T_1));
  DN_2]:=DN_2)-(s_1,2)*T_1));
  DN_3]:=DN_3)-(s_1,3)*T_1));
  DN_4]:=DN_4)-(s_1,4)*T_1));
end;

a_jac[1,1]:=a_jac[1,1]+(DUN_1,1)*Sun_T-(DUN_1]*DN_1))/sqr(SUN_T);
a_jac[1,2]:=a_jac[1,2]+(DUN_1,2)*Sun_T-(DUN_1]*DN_2))/sqr(SUN_T);
a_jac[1,3]:=a_jac[1,3]+(DUN_1,3)*Sun_T-(DUN_1]*DN_3))/sqr(SUN_T);
a_jac[1,4]:=a_jac[1,4]+(DUN_1,4)*Sun_T-(DUN_1]*DN_4))/sqr(SUN_T);

a_jac[2,1]:=a_jac[2,1]+(DUN_2,1)*Sun_T-(DUN_2]*DN_1))/sqr(SUN_T);
a_jac[2,2]:=a_jac[2,2]+(DUN_2,2)*Sun_T-(DUN_2]*DN_2))/sqr(SUN_T);
a_jac[2,3]:=a_jac[2,3]+(DUN_2,3)*Sun_T-(DUN_2]*DN_3))/sqr(SUN_T);
a_jac[2,4]:=a_jac[2,4]+(DUN_2,4)*Sun_T-(DUN_2]*DN_4))/sqr(SUN_T);

a_jac[3,1]:=a_jac[3,1]+(DUN_3,1)*Sun_T-(DUN_3]*DN_1))/sqr(SUN_T);
a_jac[3,2]:=a_jac[3,2]+(DUN_3,2)*Sun_T-(DUN_3]*DN_2))/sqr(SUN_T);
a_jac[3,3]:=a_jac[3,3]+(DUN_3,3)*Sun_T-(DUN_3]*DN_3))/sqr(SUN_T);
a_jac[3,4]:=a_jac[3,4]+(DUN_3,4)*Sun_T-(DUN_3]*DN_4))/sqr(SUN_T);

a_jac[4,1]:=a_jac[4,1]+(DUN_4,1)*Sun_T-(DUN_4]*DN_1))/sqr(SUN_T);
a_jac[4,2]:=a_jac[4,2]+(DUN_4,2)*Sun_T-(DUN_4]*DN_2))/sqr(SUN_T);
a_jac[4,3]:=a_jac[4,3]+(DUN_4,3)*Sun_T-(DUN_4]*DN_3))/sqr(SUN_T);
a_jac[4,4]:=a_jac[4,4]+(DUN_4,4)*Sun_T-(DUN_4]*DN_4))/sqr(SUN_T);
end;
end;
end;

end;

end;

end;

Procedure Fonksiyon_second;
var x,y,X1,X2,X3,X4,i,j,
    e1,e2,t2,t3,l1,l2 : integer;
    Z1,DUNNY,DUNNY_JAC,Sun_T : extended;

```

```

s_ : array [0..64] of array [1..4] of integer;
T_ : array [0..64] of extended;
DUN,dm_ : bes1;
dum_ : d_bes1;

```

```

begin
  for y:=1 to m_dim do F[y]:=0.0;
  for y:=1 to m_dim do
    for x:=1 to m_dim do a_jac[y,x]:=0.0;
  for y:=1 to 64 do
    for x:=1 to 64 do
      begin
        if (code=1) then begin t3:=1; t2:=1; end;
        if (code=2) then begin t3:=1; t2:=0; end;
        if (code=3) then begin t3:=0; t2:=1; end;
        if (code=4) then begin t3:=0; t2:=0; end;
        if ((y mod 2)=t3) and ((x mod 2)=t2) then
          begin
            case m2 of
              1 : begin
                  X1:=(m_dot[x+1,y]+m_dot[x-1,y]);
                  X2:=(m_dot[x,y+1]+m_dot[x,y-1]);
                  X4:=(m_dot[x-1,y+1]+m_dot[x+1,y-1]);
                  X3:=(m_dot[x+1,y+1]+m_dot[x-1,y-1]);
                  Z1:=exp(b_est[1]+b_est[2]*X1+b_est[3]*X2+b_est[4]*X3+b_est[5]*X4);
                  e2:=gray_level-1;
                  e1:=m_dot[x,y]-e2;
                  DUNNY:=(m_dot[x,y]+e1*Z1)/(1+Z1);
                  DUNNY_JAC:=(e2*Z1/sqr(1+Z1));
                  F[1]:=F[1]+DUNNY;
                  F[2]:=F[2]+X1*DUNNY;
                  F[3]:=F[3]+X2*DUNNY;
                  F[4]:=F[4]+X3*DUNNY;
                  F[5]:=F[5]+X4*DUNNY;
                  a_jac[1,1]:=a_jac[1,1]-DUNNY_JAC;
                  a_jac[2,2]:=a_jac[2,2]-sqr(X1)*DUNNY_JAC;
                  a_jac[3,3]:=a_jac[3,3]-sqr(X2)*DUNNY_JAC;
                  a_jac[4,4]:=a_jac[4,4]-sqr(X3)*DUNNY_JAC;
                  a_jac[5,5]:=a_jac[5,5]-sqr(X4)*DUNNY_JAC;
                  a_jac[1,2]:=a_jac[1,2]-X1*DUNNY_JAC;
                  a_jac[1,3]:=a_jac[1,3]-X2*DUNNY_JAC;
                  a_jac[1,4]:=a_jac[1,4]-X3*DUNNY_JAC;
                  a_jac[1,5]:=a_jac[1,5]-X4*DUNNY_JAC;
                  a_jac[2,1]:=a_jac[1,2];
                  a_jac[3,1]:=a_jac[1,3];
                  a_jac[4,1]:=a_jac[1,4];
                  a_jac[5,1]:=a_jac[1,5];
                  a_jac[2,3]:=a_jac[2,3]-X1*X2*DUNNY_JAC;
                  a_jac[3,2]:=a_jac[2,3];
                  a_jac[2,4]:=a_jac[2,4]-X1*X3*DUNNY_JAC;
                  a_jac[4,2]:=a_jac[2,4];
                  a_jac[2,5]:=a_jac[2,5]-X1*X4*DUNNY_JAC;
                  a_jac[5,2]:=a_jac[2,5];
                  a_jac[3,4]:=a_jac[3,4]-X2*X3*DUNNY_JAC;
                  a_jac[4,3]:=a_jac[3,4];
                  a_jac[3,5]:=a_jac[3,5]-X2*X4*DUNNY_JAC;
                  a_jac[5,3]:=a_jac[3,5];

```

```

a_jac[4,5]:=a_jac[4,5]-X3*X4*DUMMY_JAC;
a_jac[5,4]:=a_jac[4,5];
end;
2 : begin
  if m_dot[x,y]=m_dot[x+1,y] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x-1,y] then I2:=-1 else I2:=1;
  X1:=I1+I2;
  if m_dot[x,y]=m_dot[x,y+1] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x,y-1] then I2:=-1 else I2:=1;
  X2:=I1+I2;
  if m_dot[x,y]=m_dot[x+1,y-1] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x-1,y+1] then I2:=-1 else I2:=1;
  X4:=I1+I2;
  if m_dot[x,y]=m_dot[x+1,y+1] then I1:=-1 else I1:=1;
  if m_dot[x,y]=m_dot[x-1,y-1] then I2:=-1 else I2:=1;
  X3:=I1+I2;
  for i:=0 to gray_level-1 do
    begin
      if m_dot[x+1,y]=1 then I1:=-1 else I1:=1;
      if m_dot[x-1,y]=1 then I2:=-1 else I2:=1;
      s[i,1]:=I1+I2;
      if m_dot[x,y+1]=1 then I1:=-1 else I1:=1;
      if m_dot[x,y-1]=1 then I2:=-1 else I2:=1;
      s[i,2]:=I1+I2;
      if m_dot[x+1,y-1]=1 then I1:=-1 else I1:=1;
      if m_dot[x-1,y+1]=1 then I2:=-1 else I2:=1;
      s[i,4]:=I1+I2;
      if m_dot[x+1,y+1]=1 then I1:=-1 else I1:=1;
      if m_dot[x-1,y-1]=1 then I2:=-1 else I2:=1;
      s[i,3]:=I1+I2;
    end;

    for i:=0 to gray_level-1 do
      begin
        dummy:=(b_est[1]*s[i,1]+b_est[2]*s[i,2]+b_est[3]*
          *s[i,3]+b_est[4]*s[i,4]);
        T[i]:=exp(-dummy);
      end;
    SUM_T:=0;
    for i:=0 to gray_level-1 do SUM_T:=SUM_T+T[i];
    for i:=1 to m_dim do begin dum[i]:=0.0; DUM[i]:=0.0; end;

    for i:=1 to m_dim do
      for j:=1 to m_dim do dum[i,j]:=0.0;

    for i:=0 to gray_level-1 do
      begin
        DUM[1]:=DUM[1]+((s[i,1]-X1)*T[i]);
        DUM[2]:=DUM[2]+((s[i,2]-X2)*T[i]);
        DUM[3]:=DUM[3]+((s[i,3]-X3)*T[i]);
        DUM[4]:=DUM[4]+((s[i,4]-X4)*T[i]);
      end;

    F[1]:=F[1]+DUM[1]/SUM_T;
    F[2]:=F[2]+DUM[2]/SUM_T;
    F[3]:=F[3]+DUM[3]/SUM_T;
    F[4]:=F[4]+DUM[4]/SUM_T;

```

```

for i:=0 to gray_level-1 do
begin
  DUN_1,1:=DUN_1,1-(s_1,1)*(s_1,1)-X1)*T_1));
  DUN_1,2:=DUN_1,2-(s_1,2)*(s_1,1)-X1)*T_1));
  DUN_1,3:=DUN_1,3-(s_1,3)*(s_1,1)-X1)*T_1));
  DUN_1,4:=DUN_1,4-(s_1,4)*(s_1,1)-X1)*T_1));

  DUN_2,1:=DUN_2,1-(s_1,1)*(s_1,2)-X2)*T_1));
  DUN_2,2:=DUN_2,2-(s_1,2)*(s_1,2)-X2)*T_1));
  DUN_2,3:=DUN_2,3-(s_1,3)*(s_1,2)-X2)*T_1));
  DUN_2,4:=DUN_2,4-(s_1,4)*(s_1,2)-X2)*T_1));

  DUN_3,1:=DUN_3,1-(s_1,1)*(s_1,3)-X3)*T_1));
  DUN_3,2:=DUN_3,2-(s_1,2)*(s_1,3)-X3)*T_1));
  DUN_3,3:=DUN_3,3-(s_1,3)*(s_1,3)-X3)*T_1));
  DUN_3,4:=DUN_3,4-(s_1,4)*(s_1,3)-X3)*T_1));

  DUN_4,1:=DUN_4,1-(s_1,1)*(s_1,4)-X4)*T_1));
  DUN_4,2:=DUN_4,2-(s_1,2)*(s_1,4)-X4)*T_1));
  DUN_4,3:=DUN_4,3-(s_1,3)*(s_1,4)-X4)*T_1));
  DUN_4,4:=DUN_4,4-(s_1,4)*(s_1,4)-X4)*T_1));
end;

for i:=0 to gray_level-1 do
begin
  DN_1:=DN_1-(s_1,1)*T_1));
  DN_2:=DN_2-(s_1,2)*T_1));
  DN_3:=DN_3-(s_1,3)*T_1));
  DN_4:=DN_4-(s_1,4)*T_1));
end;

a_jac[1,1]:=a_jac[1,1]+(DUN_1,1)*Sun_T-DUN[1]*DN_1)/sqr(SUN_T);
a_jac[1,2]:=a_jac[1,2]+(DUN_1,2)*Sun_T-DUN[1]*DN_2)/sqr(SUN_T);
a_jac[1,3]:=a_jac[1,3]+(DUN_1,3)*Sun_T-DUN[1]*DN_3)/sqr(SUN_T);
a_jac[1,4]:=a_jac[1,4]+(DUN_1,4)*Sun_T-DUN[1]*DN_4)/sqr(SUN_T);

a_jac[2,1]:=a_jac[2,1]+(DUN_2,1)*Sun_T-DUN[2]*DN_1)/sqr(SUN_T);
a_jac[2,2]:=a_jac[2,2]+(DUN_2,2)*Sun_T-DUN[2]*DN_2)/sqr(SUN_T);
a_jac[2,3]:=a_jac[2,3]+(DUN_2,3)*Sun_T-DUN[2]*DN_3)/sqr(SUN_T);
a_jac[2,4]:=a_jac[2,4]+(DUN_2,4)*Sun_T-DUN[2]*DN_4)/sqr(SUN_T);

a_jac[3,1]:=a_jac[3,1]+(DUN_3,1)*Sun_T-DUN[3]*DN_1)/sqr(SUN_T);
a_jac[3,2]:=a_jac[3,2]+(DUN_3,2)*Sun_T-DUN[3]*DN_2)/sqr(SUN_T);
a_jac[3,3]:=a_jac[3,3]+(DUN_3,3)*Sun_T-DUN[3]*DN_3)/sqr(SUN_T);
a_jac[3,4]:=a_jac[3,4]+(DUN_3,4)*Sun_T-DUN[3]*DN_4)/sqr(SUN_T);

a_jac[4,1]:=a_jac[4,1]+(DUN_4,1)*Sun_T-DUN[4]*DN_1)/sqr(SUN_T);
a_jac[4,2]:=a_jac[4,2]+(DUN_4,2)*Sun_T-DUN[4]*DN_2)/sqr(SUN_T);
a_jac[4,3]:=a_jac[4,3]+(DUN_4,3)*Sun_T-DUN[4]*DN_3)/sqr(SUN_T);
a_jac[4,4]:=a_jac[4,4]+(DUN_4,4)*Sun_T-DUN[4]*DN_4)/sqr(SUN_T);
end;
end;
end;
end;
end;

```

```

Procedure Inverse_Matris(var inv_a : d_besli);
var k,i,j : integer;
begin
  for k:=1 to m_dim do
    begin
      for i:=1 to m_dim do
        for j:=1 to m_dim do
          if not ((i=k) or (j=k)) then
            inv_a[i,j]:=inv_a[i,j]-inv_a[i,k]*inv_a[k,j]/inv_a[k,k];
          inv_a[k,k]:=-1.0/inv_a[k,k];
          for i:=1 to m_dim do if not (i=k) then inv_a[i,k]:=inv_a[i,k]*inv_a[k,k];
          for j:=1 to m_dim do if not (j=k) then inv_a[k,j]:=inv_a[k,j]*inv_a[k,k];
        end;
      for i:=1 to m_dim do
        for j:=1 to m_dim do inv_a[i,j]:=-inv_a[i,j];
      end;
    end;
  end;
end;

```

```

Procedure Newton_Raphson;
var k1,say,i,j : integer;
    dummy : extended;
    s : str3;
    NOC : boolean;
begin
  for k1:=1 to max_iter do
    begin
      CON:=FALSE;
      NOC:=FALSE;
      case max_code of
        1 : case order of
              1 : Fonksiyon_first_zero;
              2 : Fonksiyon_second_zero;
            end;
        2 : Fonksiyon_first;
        4 : Fonksiyon_second;
      end;
      str(k1:5,s);
      PlaceStr(14,9,s);
      PlaceStr(41,9,s);
      PlaceStr(68,9,s);
      for say:=1 to m_dim do
        begin
          str(b_est[say]:2:8,BUFFER[15]);
          PlaceStr(14,9+say,cLen(BUFFER[15],12));
          str(f[say]:2:8,BUFFER[15]);
          PlaceStr(41,9+say,cLen(BUFFER[15],12));
        end;
      Inverse_Matris(a_jac);
      for i:=1 to m_dim do
        begin
          delta[i]:=0.0;
          for j:=1 to m_dim do
            begin
              DUNNY:=a_jac[i,j]*(-F[j]);
              delta[i]:=delta[i]+DUNNY;
            end;
          end;
        end;
      end;
    end;
  end;
end;

```

```

for say:=1 to m_dim do
begin
  str(delta[say]:2:8,BUFFER[15]);
  PlaceStr(58,9+say,clen(BUFFER[15],12));
end;
case m2 of
1 : for i:=1 to m_dim do if (abs(delta[i])>0.000001) then CON:=TRUE;
2 : for i:=1 to m_dim do if (abs(F[i])>2.000001) then CON:=TRUE else NOC:=TRUE;
end;
if CON then for i:=1 to m_dim do b_est[i]:=b_est[i]+delta[i];
if ((not CON) or NOC) then k1:=max_iter;
end;
end;

```

```

Procedure Condition_Value;
/* Başlangıç koşulları girilir. */
type uclu=array [1..12] of byte;
var ly      : integer;
    d       : double;
    s       : str3;
    i,ii,k   : integer;
const len   : uclu=(2,3,5,8,8,8,8,8,8,8,8,8);
    y_pos   : uclu=(9,10,11,13,14,15,16,17,18,19,20,21);

```

```

begin
  Screset;
  Clear_Buffer;
  Mes(Model_Name+' Model',1);
  Hid_Frame(23,5,59,13,'Parameters',Palet[9]);
  PlaceStr(26,9,Dots('Order Level',23));
  PlaceStr(26,10,Dots('Which Coding',23));
  PlaceStr(26,11,Dots('Maximum Iteration',23));
  l:=1;
  spkeys(25);
  Str(order,BUFFER[1]);
  BUFFER[2]:=BUFFER[1];
  BUFFER[3]:='50';
  for ii:=4 to 14 do BUFFER[ii]:='0';
  Repeat
    CON:=TRUE;
    InputStr(BUFFER[1],len[1],50,y_pos[1],Term,TC);
    Case L Of
      1 : Intcon(CON,1,4,BUFFER[L],k);
      2 : Intcon(CON,0,2,BUFFER[L],k);
      3 : begin
          Intcon(CON,1,10000,BUFFER[L],k);
          if (TC='N') or (TC='X') Then
            begin
              store_scr;
              val(BUFFER[1],k,1);
              if m2=1 then ii:=2*k else ii:=2*k-1;
              Mes_Frame(44,12,66,14+ii,Palet[5]);
              For i:=1 to ii+1 do
                begin
                  str(i,s);

```

```

        PlaceStr(47,12+i,Dots('b_est['+s+'],'9));
    end;
    ly:=1;
    Repeat
        CON:=TRUE;
        inputstr(BUFFER[3+ly],len[ly+3],57,y_pos[ly+3],Term,TC);
        Case ly Of
            1..8 : doublecon(CON,-10.0,10.0,BUFFER[3+ly],d);
        end;
        If CON Then Incdec(ii+1,ly,TC);
        Until (TC='Z') OR (((TC='N') AND (ly=1)) AND CON);
        if TC='Z' Then begin Load_Scr; TC:=' '; end else DEC(SCR_LEVEL);
    end;
end;
End;
If CON Then Incdec(3,L,TC);
Until (TC='Z') OR (((TC='N') AND (l=1)) AND CON);
if TC<>'Z' then
begin
    val(BUFFER[2],k,1);
    if k=0 then max_code:=1 else max_code:=2*k;
    val(BUFFER[1],k,1);
    n_dim:=li+1;
    val(BUFFER[3],k,1); max_iter:=k;
end;
end;

var say,ek : integer;
    d : extended;
    s : str3;

begin
    if sec=0 then
        begin
            Screset;
            Mes_Frame(20,10,60,14,PALET[5]);
            PlaceStr(30,11,'!!! W A R N I N G !!!');
            PlaceStr(30,12,'No Texture in Memory');
            PlaceStr(23,13,'Please, Load or Generate a Texture');
            Waitkey; Exit;
        end;
    m2:=0; TC:=' ';
    W_DIR:=TRUE;
    SUB[1]:='Auto Binomial Model';
    SUB[2]:='Derin & Elliot Model';
    Hidmenu(m2,2,20,16,28,'Parameter Estimation');
    if m2=0 Then Exit;
    Condition_Value;
    if TC<>'Z' then
        begin
            Screset;
            if max_code=4 then ek:=2 else ek:=0;
            Mid_Frame(1,6,26,13+ek,'Estimating Value',PALET[8]);
            Mid_Frame(28,6,53,13+ek,'Function Value',PALET[8]);
            Mid_Frame(55,6,80,13+ek,'Difference Value',PALET[8]);
            PlaceStr(3,9,Dots('Iteration',10));

```

```

PlaceStr(30,9,Dots('Iteration',10));
PlaceStr(57,9,Dots('Iteration',10));
for say:=1 to m_dim do
begin
  str(say,s);
  PlaceStr(3,9+say,Dots('b_est['+s+']',10));
end;
for say:=1 to m_dim do
begin
  str(say,s);
  PlaceStr(30,9+say,Dots('Func['+s+']',10));
end;
for say:=1 to m_dim do
begin
  str(say,s);
  PlaceStr(57,9+say,Dots('delta['+s+']',10));
end;
Mid_Frame(1,15+ek,80,23+ek,'',PALET[5]);
for say:=1 to m_dim do
begin
  str(say,s);
  PlaceStr(10+14*(say-1),16+ek,'b_est['+s+']');
end;
PlaceStr(2,22+ek,'Result:');
{ begin iteration }
for code:=1 to max_code do
begin
  str(code,s);
  Mes_Frame(1,1,80,4,PALET[5]);
  PlaceStr(3,2,Dots('Coding Order',20)+' '+BUFFER[2]);
  PlaceStr(27,2,Dots('Coding Region',20)+' '+s);
  PlaceStr(53,2,Dots('Maximum Iterasyon',20)+' '+BUFFER[3]);
  for say:=1 to m_dim do
begin
  str(say,s);
  PlaceStr(3+16*(say-1),3,Dots('b_est['+s+']',9)+BUFFER[say+3]);
end;
for say:=1 to m_dim do
begin
  val(BUFFER[3+say],d,k);
  b_est[say]:=d;
end;
Newton_Raphson;
for say:=1 to m_dim do temp[say,code]:=b_est[say];
for i:=1 to code do
begin
  str(i:2,s);
  PlaceStr(3,17+ek+i,s+'');
  for say:=1 to m_dim do
begin
  str(temp[say,i]:2:8,BUFFER[15]);
  PlaceStr(10+14*(say-1),17+ek+i,clen(BUFFER[15],12));
end;
end;
end;
for say:=1 to m_dim do ave[say]:=0.0;
for say:=1 to m_dim do

```

```
    for code:=1 to max_code do ave[say]:=ave[say]+(temp[say,code]/max_code);
  for say:=1 to m_dim do
    begin
      str(ave[say]:2:8,BUFFER[15]);
      PlaceStr(10+14*(say-1),22+ek,clen(BUFFER[15],12));
    end;
    waitkey;
  end;
end;
end.
end.
```

```
/* Ki-Kare Testi */
```

```
unit ChiBox;
```

```
{$N+}
```

```
interface
```

```
uses crt,psrvbox,menubox,SrvBox;
```

```
Procedure Chi_Square_Test(m_dot : m_lattice);
```

IMPLEMENTATION

```
Procedure Chi_Square_Test(m_dot : m_lattice);
```

```
  type tip = array [1..2] of array [1..81] of integer;
  tip2 = array [1..2] of array [1..81] of double;
```

```
  var exp_dot : tip2;
      obs_dot : tip;
      k,coding,ord_,x,y : integer;
```

```
Procedure First_Order_Zero_Coding(var dott : tip);
```

```
  var x,y,i,e1,e2 : integer;
```

```
  begin
```

```
    for y:=1 to 2 do
```

```
      for x:=1 to 10 do dott[y,x]:=0;
```

```
    for y:=1 to 64 do
```

```
      for x:=1 to 64 do
```

```
        begin
```

```
          e1:=m_dot[x+1,y]+m_dot[x-1,y];
```

```
          e2:=m_dot[x,y+1]+m_dot[x,y-1];
```

```
          for i:=0 to 1 do
```

```
            if m_dot[x,y]=i then
```

```
              begin
```

```
                if (e1=0) and (e2=0) then inc(dott[i+1,1]);
```

```
                if (e1=0) and (e2=1) then inc(dott[i+1,2]);
```

```
                if (e1=0) and (e2=2) then inc(dott[i+1,3]);
```

```
                if (e1=1) and (e2=0) then inc(dott[i+1,4]);
```

```
                if (e1=1) and (e2=1) then inc(dott[i+1,5]);
```

```
                if (e1=1) and (e2=2) then inc(dott[i+1,6]);
```

```
                if (e1=2) and (e2=0) then inc(dott[i+1,7]);
```

```
                if (e1=2) and (e2=1) then inc(dott[i+1,8]);
```

```
                if (e1=2) and (e2=2) then inc(dott[i+1,9]);
```

```
              end;
```

```
            end;
```

```
          end;
```

```
Procedure First_order_coding(var dott : tip);
```

```
  var x,y,i,e1,e2,t2 : integer;
```

```
  begin
```

```
    for y:=1 to 2 do
```

```
      for x:=1 to 10 do dott[y,x]:=0;
```

```
    for y:=1 to 64 do
```

```

for x:=1 to 64 do
begin
  if (coding=1) then if (y mod 2)=1 then t2:=1 else t2:=0;
  if (coding=2) then if (y mod 2)=1 then t2:=0 else t2:=1;
  if (x mod 2)=t2 then
begin
  e1:=m_dot[x+1,y]+m_dot[x-1,y];
  e2:=m_dot[x,y+1]+m_dot[x,y-1];
  for i:=0 to 1 do
    if m_dot[x,y]=i then
begin
  if (e1=0) and (e2=0) then inc(dott[i+1,1]);
  if (e1=0) and (e2=1) then inc(dott[i+1,2]);
  if (e1=0) and (e2=2) then inc(dott[i+1,3]);
  if (e1=1) and (e2=0) then inc(dott[i+1,4]);
  if (e1=1) and (e2=1) then inc(dott[i+1,5]);
  if (e1=1) and (e2=2) then inc(dott[i+1,6]);
  if (e1=2) and (e2=0) then inc(dott[i+1,7]);
  if (e1=2) and (e2=1) then inc(dott[i+1,8]);
  if (e1=2) and (e2=2) then inc(dott[i+1,9]);
end;
end;
end;
end;

```

Procedure First_Order_Second_Order_Spacing_Zero_Coding(var dott : tlp);

var x,y,i,e1,e2,t2,t3 : integer;

begin

for y:=1 to 2 do

for x:=1 to 10 do dott[y,x]:=0;

for y:=1 to 64 do

for x:=1 to 64 do

begin

e1:=m_dot[x+1,y]+m_dot[x-1,y];

e2:=m_dot[x,y+1]+m_dot[x,y-1];

for i:=0 to 1 do

if m_dot[x,y]=i then

begin

if (e1=0) and (e2=0) then inc(dott[i+1,1]);

if (e1=0) and (e2=1) then inc(dott[i+1,2]);

if (e1=0) and (e2=2) then inc(dott[i+1,3]);

if (e1=1) and (e2=0) then inc(dott[i+1,4]);

if (e1=1) and (e2=1) then inc(dott[i+1,5]);

if (e1=1) and (e2=2) then inc(dott[i+1,6]);

if (e1=2) and (e2=0) then inc(dott[i+1,7]);

if (e1=2) and (e2=1) then inc(dott[i+1,8]);

if (e1=2) and (e2=2) then inc(dott[i+1,9]);

end;

end;

end;

Procedure First_Order_Second_Order_Spacing_Coding(var dott : tlp);

var x,y,i,e1,e2,t2,t3 : integer;

begin

for y:=1 to 2 do

```

for x:=1 to 10 do dott[y,x]:=0;
for y:=1 to 64 do
  for x:=1 to 64 do
    begin
      if (coding=1) then begin t3:=1; t2:=1; end;
      if (coding=2) then begin t3:=1; t2:=0; end;
      if (coding=3) then begin t3:=0; t2:=1; end;
      if (coding=4) then begin t3:=0; t2:=0; end;
      if ((y mod 2)=t3) and ((x mod 2)=t2) then
        begin
          e1:=m_dot[x+1,y]+m_dot[x-1,y];
          e2:=m_dot[x,y+1]+m_dot[x,y-1];
          for i:=0 to 1 do
            if m_dot[x,y]=i then
              begin
                if (e1=0) and (e2=0) then inc(dott[i+1,1]);
                if (e1=0) and (e2=1) then inc(dott[i+1,2]);
                if (e1=0) and (e2=2) then inc(dott[i+1,3]);
                if (e1=1) and (e2=0) then inc(dott[i+1,4]);
                if (e1=1) and (e2=1) then inc(dott[i+1,5]);
                if (e1=1) and (e2=2) then inc(dott[i+1,6]);
                if (e1=2) and (e2=0) then inc(dott[i+1,7]);
                if (e1=2) and (e2=1) then inc(dott[i+1,8]);
                if (e1=2) and (e2=2) then inc(dott[i+1,9]);
              end;
            end;
          end;
        end;
      end;
    end;
  end;
end;

```

```

Procedure Second_Order_Zero_Coding(var dott : tip);
var x,y,i,e1,e2,e3,e4,t2,t3 : integer;
begin
  for y:=1 to 2 do
    for x:=1 to 81 do dott[y,x]:=0;
  for y:=1 to 64 do
    for x:=1 to 64 do
      begin
        e1:=m_dot[x+1,y]+m_dot[x-1,y];
        e2:=m_dot[x,y+1]+m_dot[x,y-1];
        e3:=m_dot[x+1,y-1]+m_dot[x-1,y+1];
        e4:=m_dot[x+1,y+1]+m_dot[x-1,y-1];
        for i:=0 to 1 do
          if m_dot[x,y]=i then
            begin
              if (e1=0) and (e2=0) and (e3=0) and (e4=0) then inc(dott[i+1,1]);
              if (e1=0) and (e2=0) and (e3=0) and (e4=1) then inc(dott[i+1,2]);
              if (e1=0) and (e2=0) and (e3=0) and (e4=2) then inc(dott[i+1,3]);
              if (e1=0) and (e2=0) and (e3=1) and (e4=0) then inc(dott[i+1,4]);
              if (e1=0) and (e2=0) and (e3=1) and (e4=1) then inc(dott[i+1,5]);
              if (e1=0) and (e2=0) and (e3=1) and (e4=2) then inc(dott[i+1,6]);
              if (e1=0) and (e2=0) and (e3=2) and (e4=0) then inc(dott[i+1,7]);
              if (e1=0) and (e2=0) and (e3=2) and (e4=1) then inc(dott[i+1,8]);
              if (e1=0) and (e2=0) and (e3=2) and (e4=2) then inc(dott[i+1,9]);
              if (e1=0) and (e2=1) and (e3=0) and (e4=0) then inc(dott[i+1,10]);
              if (e1=0) and (e2=1) and (e3=0) and (e4=1) then inc(dott[i+1,11]);
              if (e1=0) and (e2=1) and (e3=0) and (e4=2) then inc(dott[i+1,12]);
            end;
          end;
        end;
      end;
    end;
  end;
end;

```

[illegible]

```

        if (e1=2) and (e2=1) and (e3=2) and (e4=0) then inc(dott[i+1,70]);
        if (e1=2) and (e2=1) and (e3=2) and (e4=1) then inc(dott[i+1,71]);
        if (e1=2) and (e2=1) and (e3=2) and (e4=2) then inc(dott[i+1,72]);
        if (e1=2) and (e2=2) and (e3=0) and (e4=0) then inc(dott[i+1,73]);
        if (e1=2) and (e2=2) and (e3=0) and (e4=1) then inc(dott[i+1,74]);
        if (e1=2) and (e2=2) and (e3=0) and (e4=2) then inc(dott[i+1,75]);
        if (e1=2) and (e2=2) and (e3=1) and (e4=0) then inc(dott[i+1,76]);
        if (e1=2) and (e2=2) and (e3=1) and (e4=1) then inc(dott[i+1,77]);
        if (e1=2) and (e2=2) and (e3=1) and (e4=2) then inc(dott[i+1,78]);
        if (e1=2) and (e2=2) and (e3=2) and (e4=0) then inc(dott[i+1,79]);
        if (e1=2) and (e2=2) and (e3=2) and (e4=1) then inc(dott[i+1,80]);
        if (e1=2) and (e2=2) and (e3=2) and (e4=2) then inc(dott[i+1,81]);
    end;
end;
end;

```

Procedure Second_Order_Coding(var dott : tip);

var x,y,i,e1,e2,e3,e4,t2,t3 : integer;

begin

for y:=1 to 2 do

for x:=1 to 81 do dott[y,x]:=0;

for y:=1 to 64 do

for x:=1 to 64 do

begin

if (coding=1) then begin t3:=1; t2:=1; end;

if (coding=2) then begin t3:=1; t2:=0; end;

if (coding=3) then begin t3:=0; t2:=1; end;

if (coding=4) then begin t3:=0; t2:=0; end;

if ((y mod 2)=t3) and ((x mod 2)=t2) then

begin

e1:=m_dot[x+1,y]+m_dot[x-1,y];

e2:=m_dot[x,y+1]+m_dot[x,y-1];

e3:=m_dot[x-1,y+1]+m_dot[x+1,y-1];

e4:=m_dot[x-1,y-1]+m_dot[x+1,y+1];

for i:=0 to 1 do

if m_dot[x,y]=i then

begin

if (e1=0) and (e2=0) and (e3=0) and (e4=0) then inc(dott[i+1,1]);

if (e1=0) and (e2=0) and (e3=0) and (e4=1) then inc(dott[i+1,2]);

if (e1=0) and (e2=0) and (e3=0) and (e4=2) then inc(dott[i+1,3]);

if (e1=0) and (e2=0) and (e3=1) and (e4=0) then inc(dott[i+1,4]);

if (e1=0) and (e2=0) and (e3=1) and (e4=1) then inc(dott[i+1,5]);

if (e1=0) and (e2=0) and (e3=1) and (e4=2) then inc(dott[i+1,6]);

if (e1=0) and (e2=0) and (e3=2) and (e4=0) then inc(dott[i+1,7]);

if (e1=0) and (e2=0) and (e3=2) and (e4=1) then inc(dott[i+1,8]);

if (e1=0) and (e2=0) and (e3=2) and (e4=2) then inc(dott[i+1,9]);

if (e1=0) and (e2=1) and (e3=0) and (e4=0) then inc(dott[i+1,10]);

if (e1=0) and (e2=1) and (e3=0) and (e4=1) then inc(dott[i+1,11]);

if (e1=0) and (e2=1) and (e3=0) and (e4=2) then inc(dott[i+1,12]);

if (e1=0) and (e2=1) and (e3=1) and (e4=0) then inc(dott[i+1,13]);

if (e1=0) and (e2=1) and (e3=1) and (e4=1) then inc(dott[i+1,14]);

if (e1=0) and (e2=1) and (e3=1) and (e4=2) then inc(dott[i+1,15]);

if (e1=0) and (e2=1) and (e3=2) and (e4=0) then inc(dott[i+1,16]);

if (e1=0) and (e2=1) and (e3=2) and (e4=1) then inc(dott[i+1,17]);

if (e1=0) and (e2=1) and (e3=2) and (e4=2) then inc(dott[i+1,18]);

[illegible]

```

        if (e1=2) and (e2=2) and (e3=1) and (e4=0) then inc(dott[i+1,76]);
        if (e1=2) and (e2=2) and (e3=1) and (e4=1) then inc(dott[i+1,77]);
        if (e1=2) and (e2=2) and (e3=1) and (e4=2) then inc(dott[i+1,78]);
        if (e1=2) and (e2=2) and (e3=2) and (e4=0) then inc(dott[i+1,79]);
        if (e1=2) and (e2=2) and (e3=2) and (e4=1) then inc(dott[i+1,80]);
        if (e1=2) and (e2=2) and (e3=2) and (e4=2) then inc(dott[i+1,81]);
    end;
end;
end;
end;

```

```

Procedure Condition_Value;
type uclu=array [1..2] of byte;
var l,i : integer;
const len : uclu=(2,2);
      y_pos : uclu=(11,12);

begin
  Screset;
  Clear_Buffer;
  Mes(Model_Name+' Model',1);
  Hid_Frame(23,5,60,14,'Parameters',Palet[9]);
  PlaceStr(26,9,Dots('Order Level',23));
  PlaceStr(26,11,Dots('Which Order Spacing',23));
  PlaceStr(26,12,Dots('Which Codings',23));
  l:=1;
  spkeys(25);
  Str(order,BUFFER[1]);
  PlaceStr(50,9,BUFFER[1]);
  BUFFER[2]:=BUFFER[1];
  BUFFER[3]:='1';
  Repeat
    CON:=TRUE;
    inputstr(BUFFER[l+1],len[l],50,y_pos[l],Term,TC);
    Case L Of
      1 : Intcon(CON,1,4,BUFFER[L+1],k);
      2 : Intcon(CON,0,4,BUFFER[L+1],k);
    End;
    If CON Then Incdec(2,L,TC);
  Until (TC='Z') OR (((TC='N') AND (l=1)) AND CON);
  if TC<>'Z' then
    begin
      val(BUFFER[2],k,1);
      ord_:=k;
      val(BUFFER[3],k,1);
      coding:=k;
    end;
  end;
end;

```

```

Procedure Calculate;
var x,y,l : integer;
begin
  case order of
    1 : case ord_of

```

```

1 : case coding of
  0 : First_Order_Zero_Coding(obs_dot);
  1..2 : First_Order_Coding(obs_dot);
end;
2 : case coding of
  0 : First_Order_Second_Order_Spacing_Zero_Coding(obs_dot);
  1..2 : First_Order_Second_Order_Spacing_Coding(obs_dot);
end;
end;
2 : case ord_of
  2 : case coding of
    0 : Second_Order_Zero_Coding(obs_dot);
    1..4 : Second_Order_Coding(obs_dot);
  end;
end;
end;
end;
end;

```

```

Procedure Take_Param(var a1,a2,a3 : double);
type uclu=array [1..3] of byte;
var l,i,k : integer;
    d : double;
const len : uclu=(8,8,8);
    y_pos : uclu=(9,11,13);

begin
  Screset;
  Clear_Buffer;
  Mes(Model_Name+' Model',1);
  Mid_Frame(25,5,55,15,'Parameters',Palet[9]);
  PlaceStr(28, 9,Dots('a',10));
  PlaceStr(28,11,Dots('b[1,1]',10));
  PlaceStr(28,13,Dots('b[1,2]',10));
  l:=1;
  spkeys(25);
  Repeat
    CON:=TRUE;
    inputstr(BUFFER[l],len[l],39,y_pos[l],Term,TC);
    Case L Of
      1..3 : doublecon(CON,-10,10,BUFFER[L],d);
    End;
    If CON Then Incdec(3,L,TC);
  Until (TC='Z') OR (((TC='H') AND (l=1)) AND CON);
  if TC<>'Z' then
    begin
      val(BUFFER[1],d,1);
      a1:=d;
      val(BUFFER[2],d,1);
      a2:=d;
      val(BUFFER[3],d,1);
      a3:=d;
    end;
  end;
end;

```

```

var k1_ : array [1..2] of double;
k1_kare, k1, k2, k3 : double;
sum      : array [1..81] of integer;
freq,prob : tip2;
u,t,z,w : array [1..81] of integer;
para, degree_of_freedom : integer;

begin
  if sec=0 then
    begin
      Screset;
      Mes_Frame(20,10,60,14,PALET[5]);
      PlaceStr(30,11,'!!! W A R N I N G !!!');
      PlaceStr(30,12,'No Texture in Memory');
      PlaceStr(23,13,'Please, Load or Generate a Texture');
      Waitkey; Exit;
    end;
    Condition_Value;
    Screset;
    Take_Param(k1,k2,k3);
    if TC<>'Z' then
      begin
        screset;
        Calculate;
        if order=1 then
          begin
            u[1]:=0; u[2]:=0; u[3]:=0;
            u[4]:=1; u[5]:=1; u[6]:=1;
            u[7]:=2; u[8]:=2; u[9]:=2;
            t[1]:=0; t[2]:=1; t[3]:=2;
            t[4]:=0; t[5]:=1; t[6]:=2;
            t[7]:=0; t[8]:=1; t[9]:=2;

            para:=3;
            for y:=1 to 2 do
              for x:=1 to 10 do if obs_dot[y,x]=0 then inc(obs_dot[y,x]);

              PlaceStr(1,1,'u1 + u2' + t1 + t2' + x=0 Obsr.' + x=0 Expt.'
                + x=1 Obsrv.' + x=1 Expt. ');
              PlaceStr(1,2,'-----' + '-----' + '-----' + '-----'
                + '-----' + '-----');
              for x:=1 to 9 do
                begin
                  str(obs_dot[1,x],BUFFER[15]);
                  str(obs_dot[2,x],BUFFER[14]);
                  str(u[x]:1,BUFFER[13]);
                  str(t[x]:1,BUFFER[12]);
                  PlaceStr(1,2+x,' '+c1en(BUFFER[13],9)+BUFFER[12]+rset(BUFFER[15],14)
                    +c1en('',11)+rset(BUFFER[14],13));
                end;
              PlaceStr(1,3+x,'-----' + '-----' + '-----' + '-----'
                + '-----' + '-----');
              for x:=1 to 9 do prob[1,x]:=1/(1+exp(k1+k2*u[x]+k3*t[x]));
              for x:=1 to 9 do prob[2,x]:= (exp(k1+k2*u[x]+k3*t[x]))/(1+exp(k1+k2*u[x]+k3*t[x]));

```

```

for y:=1 to 9 do sum[y]:=obs_dot[1,y]+obs_dot[2,y];

for x:=1 to 9 do exp_dot[1,x]:=sum[x]*prob[1,x];
for x:=1 to 9 do exp_dot[2,x]:=sum[x]*prob[2,x];

for y:=1 to 2 do
  for x:=1 to 10 do if exp_dot[y,x]<1 then exp_dot[y,x]:=1.0;

for x:=1 to 9 do
  begin
    str(exp_dot[1,x]:4:0,BUFFER[15]);
    str(exp_dot[2,x]:4:0,BUFFER[14]);
    PlaceStr(29,2+x,rset(BUFFER[15],11));
    PlaceStr(53,2+x,rset(BUFFER[14],11));
  end;

for y:=1 to 2 do
  for x:=1 to 9 do freq[y,x]:=0.0;

for y:=1 to 2 do
  for x:=1 to 9 do freq[y,x]:=obs_dot[y,x]-exp_dot[y,x];

Ki_1]:=0;
Ki_2]:=0;
for y:=1 to 2 do
  for x:=1 to 9 do Ki_1y]:=Ki_1y]+(sqr(freq[y,x])/exp_dot[y,x]);
Ki_Kare:=Ki_1]+Ki_2];
str(Ki_Kare:5:6,BUFFER[15]);
Degree_of_Freedom:=(Gray_level-1)*9-para;
str(Degree_of_Freedom,BUFFER[14]);
PlaceStr(10,23,' Degree of Freedom : '+BUFFER[14]);
PlaceStr(10,18,' Chi Square Value : '+BUFFER[15]);
end
else
begin
  for y:=1 to 27 do begin u[y]:=0; u[y+27]:=1; u[y+54]:=2; end;
  for y:=1 to 9 do
    begin
      t[y]:=0; t[y+9]:=1; t[y+18]:=2;
      t[y+27]:=0; t[y+36]:=1; t[y+45]:=2;
      t[y+54]:=0; t[y+63]:=1; t[y+72]:=2;
    end;
  for y:=1 to 3 do
    begin
      s[y]:=0; s[y+3]:=1; s[y+6]:=2;
      s[y+9]:=0; s[y+12]:=1; s[y+15]:=2;
      s[y+18]:=0; s[y+21]:=1; s[y+24]:=2;
      s[y+27]:=0; s[y+30]:=1; s[y+33]:=2;
      s[y+36]:=0; s[y+39]:=1; s[y+42]:=2;
      s[y+45]:=0; s[y+48]:=1; s[y+51]:=2;
      s[y+54]:=0; s[y+57]:=1; s[y+60]:=2;
      s[y+63]:=0; s[y+66]:=1; s[y+69]:=2;
      s[y+72]:=0; s[y+75]:=1; s[y+78]:=2;
    end;

  for y:=1 to 81 do
    begin

```

```

    if (y mod 3)=1 then w[y]:=0;
    if (y mod 3)=2 then w[y]:=1;
    if (y mod 3)=0 then w[y]:=2;
end;

para:=5;
for y:=1 to 2 do
  for x:=1 to 81 do if obs_dot[y,x]=0 then inc(obs_dot[y,x]);

PlaceStr(1,1,' u   t '+'   z   w '+'   x=0 Obsr.'+'   x=0 Expt.'
          '+'   x=1 Obsrv.'+'   x=1 Expt. ');
PlaceStr(1,2,'---  --- '+'  ---  --- '+'  ----- '+'  -----'
          '+'  ----- '+'  ----- ');

  for x:=1 to 81 do
    prob[1,x]:=1/(1+exp(a+b[1,1]*u[x]+b[1,2]*t[x]+b[2,1]*z[x]+b[2,2]*w[x]));

  for x:=1 to 81 do
    prob[2,x]:=(exp(a+b[1,1]*u[x]+b[1,2]*t[x]+b[2,1]*z[x]+b[2,2]*w[x]))
              /(1+exp(a+b[1,1]*u[x]+b[1,2]*t[x]+b[2,1]*z[x]+b[2,2]*w[x]));

  for y:=1 to 81 do sum[y]:=obs_dot[1,y]+obs_dot[2,y];

  for x:=1 to 81 do exp_dot[1,x]:=sum[x]*prob[1,x];
  for x:=1 to 81 do exp_dot[2,x]:=sum[x]*prob[2,x];

  for y:=1 to 2 do
    for x:=1 to 81 do if exp_dot[y,x]<1 then exp_dot[y,x]:=1.0;

gotoxy(1,3);
for x:=1 to 81 do
begin
  y:=x mod 21; if y=0 then inc(y);
  str(obs_dot[1,x],BUFFER[15]);
  str(obs_dot[2,x],BUFFER[14]);
  str(u[x]:1,BUFFER[13]);
  str(t[x]:1,BUFFER[12]);
  str(z[x]:1,BUFFER[11]);
  str(w[x]:1,BUFFER[10]);
  str(exp_dot[1,x]:4:0,BUFFER[9]);
  str(exp_dot[2,x]:4:0,BUFFER[8]);

  writeln(' '+cLen(BUFFER[13],4)+cLen(BUFFER[12],4)+cLen(BUFFER[11],4)
          +cLen(BUFFER[10],4)+rset(BUFFER[15],14)+RSET(BUFFER[9],11)
          +rset(BUFFER[14],13)+RSET(BUFFER[8],11));

  if (x mod 21)=0 then
  begin
    waitkey;
    region_color(1,3,80,22,0);
    gotoxy(1,3);
  end;
end;

PlaceStr(1,3+x,'---  --- '+'  ---  --- '+'  ----- '+'  -----'
          '+'  ----- '+'  ----- ');

for y:=1 to 2 do
  for x:=1 to 81 do freq[y,x]:=0.0;

```

```

for y:=1 to 2 do
  for x:=1 to 81 do freq[y,x]:=obs_dot[y,x]-exp_dot[y,x];

  Ki_1]:=0;
  Ki_2]:=0;
  for y:=1 to 2 do
    for x:=1 to 81 do Ki_1]:=Ki_1]+(sqr(freq[y,x])/exp_dot[y,x]);
  Ki_2]:=Ki_1]+Ki_2];
  str(Ki_2:=5:8,BUFFER[15]);
  Degree_of_Freedom:=(Gray_level-1)*81-para;
  str(Degree_of_Freedom,BUFFER[14]);
  PlaceStr(10,23,' Degree of Freedom : '+BUFFER[14]);
  PlaceStr(10,24,' Chi Square Value : '+BUFFER[15]);
end;
waitkey;
end;
end;
end.

```

```
/* Özellik Çıkarım Bölümü */
```

```
unit Classify;
```

```
{%N+}
```

```
interface
```

```
uses crt, psvrbox, menubox, srvbox;
```

```
procedure Feature_Extraction(m_dot : m_lattice);
```

```
IMPLEMENTATION
```

```
Procedure Feature_Extraction(m_dot : m_lattice);
```

```
/* Özellik Çıkarımı */
```

```
var pair : array [0..3] of array [0..3] of integer;
```

```
f : array [0..3] of array [0..3] of double;
```

```
Energy, Entropy, Inertia,
```

```
Local_Homogeneity, Correlation,
```

```
Hx, Hy, Sigma_x, Sigma_y, Cor2, Cor3 : extended;
```

```
R, x, y, i, G_level : integer;
```

```
choice : Char;
```

```
val_unknown : array [1..20] of double;
```

```
bfile : text;
```

```
Procedure Calculate_R_Theta_0;
```

```
var x, y : integer;
```

```
begin
```

```
  R:=0;
```

```
  for y:=0 to G_level do
```

```
    for x:=0 to G_level do pair[y,x]:=0;
```

```
  for y:=1 to 64 do
```

```
    for x:=1 to 63 do
```

```
      case m_dot[x,y] of
```

```
        0 : case m_dot[x+1,y] of
```

```
          0 : begin inc(pair[0,0]); inc(pair[0,0]); end;
```

```
          1 : begin inc(pair[0,1]); inc(pair[1,0]); end;
```

```
          2 : begin inc(pair[0,2]); inc(pair[2,0]); end;
```

```
          3 : begin inc(pair[0,3]); inc(pair[3,0]); end;
```

```
        end;
```

```
        1 : case m_dot[x+1,y] of
```

```
          0 : begin inc(pair[1,0]); inc(pair[0,1]); end;
```

```
          1 : begin inc(pair[1,1]); inc(pair[1,1]); end;
```

```
          2 : begin inc(pair[1,2]); inc(pair[2,1]); end;
```

```
          3 : begin inc(pair[1,3]); inc(pair[3,1]); end;
```

```
        end;
```

```
        2 : case m_dot[x+1,y] of
```

```
          0 : begin inc(pair[2,0]); inc(pair[0,2]); end;
```

```
          1 : begin inc(pair[2,1]); inc(pair[1,2]); end;
```

```
          2 : begin inc(pair[2,2]); inc(pair[2,2]); end;
```

```
          3 : begin inc(pair[2,3]); inc(pair[3,2]); end;
```

```
        end;
```

```

3 : case m_dot[x+1,y] of
  0 : begin inc(pair[3,0]); inc(pair[0,3]); end;
  1 : begin inc(pair[3,1]); inc(pair[1,3]); end;
  2 : begin inc(pair[3,2]); inc(pair[2,3]); end;
  3 : begin inc(pair[3,3]); inc(pair[3,3]); end;
end;
end;
for y:=0 to G_level do
  for x:=0 to G_level do R:=R+pair[y,x];

  for y:=0 to G_level do
    for x:=0 to G_level do f[y,x]:=pair[y,x]/R;
  end;
end;

```

Procedure Calculate_R_Theta_45;

```

var x,y : integer;
begin
  R:=0;
  for y:=0 to G_level do
    for x:=0 to G_level do pair[y,x]:=0;
    for y:=2 to 64 do
      for x:=1 to 63 do
        case m_dot[x,y] of
          0 : case m_dot[x+1,y-1] of
            0 : begin inc(pair[0,0]); inc(pair[0,0]); end;
            1 : begin inc(pair[0,1]); inc(pair[1,0]); end;
            2 : begin inc(pair[0,2]); inc(pair[2,0]); end;
            3 : begin inc(pair[0,3]); inc(pair[3,0]); end;
          end;
          1 : case m_dot[x+1,y-1] of
            0 : begin inc(pair[1,0]); inc(pair[0,1]); end;
            1 : begin inc(pair[1,1]); inc(pair[1,1]); end;
            2 : begin inc(pair[1,2]); inc(pair[2,1]); end;
            3 : begin inc(pair[1,3]); inc(pair[3,1]); end;
          end;
          2 : case m_dot[x+1,y-1] of
            0 : begin inc(pair[2,0]); inc(pair[0,2]); end;
            1 : begin inc(pair[2,1]); inc(pair[1,2]); end;
            2 : begin inc(pair[2,2]); inc(pair[2,2]); end;
            3 : begin inc(pair[2,3]); inc(pair[3,2]); end;
          end;
          3 : case m_dot[x+1,y-1] of
            0 : begin inc(pair[3,0]); inc(pair[0,3]); end;
            1 : begin inc(pair[3,1]); inc(pair[1,3]); end;
            2 : begin inc(pair[3,2]); inc(pair[2,3]); end;
            3 : begin inc(pair[3,3]); inc(pair[3,3]); end;
          end;
        end;
      end;
    end;
    for y:=0 to G_level do
      for x:=0 to G_level do R:=R+pair[y,x];

      for y:=0 to G_level do
        for x:=0 to G_level do f[y,x]:=pair[y,x]/R;
      end;
    end;
  end;
end;

```

Procedure Calculate_R_Theta_90;

```

var x,y : integer;
begin
  R:=0;
  for y:=0 to G_level do
    for x:=0 to G_level do pair[y,x]:=0;
  for y:=2 to 64 do
    for x:=1 to 64 do
      case m_dot[x,y] of
        0 : case m_dot[x,y-1] of
          0 : begin inc(pair[0,0]); inc(pair[0,0]); end;
          1 : begin inc(pair[0,1]); inc(pair[1,0]); end;
          2 : begin inc(pair[0,2]); inc(pair[2,0]); end;
          3 : begin inc(pair[0,3]); inc(pair[3,0]); end;
        end;
        1 : case m_dot[x,y-1] of
          0 : begin inc(pair[1,0]); inc(pair[0,1]); end;
          1 : begin inc(pair[1,1]); inc(pair[1,1]); end;
          2 : begin inc(pair[1,2]); inc(pair[2,1]); end;
          3 : begin inc(pair[1,3]); inc(pair[3,1]); end;
        end;
        2 : case m_dot[x,y-1] of
          0 : begin inc(pair[2,0]); inc(pair[0,2]); end;
          1 : begin inc(pair[2,1]); inc(pair[1,2]); end;
          2 : begin inc(pair[2,2]); inc(pair[2,2]); end;
          3 : begin inc(pair[2,3]); inc(pair[3,2]); end;
        end;
        3 : case m_dot[x,y-1] of
          0 : begin inc(pair[3,0]); inc(pair[0,3]); end;
          1 : begin inc(pair[3,1]); inc(pair[1,3]); end;
          2 : begin inc(pair[3,2]); inc(pair[2,3]); end;
          3 : begin inc(pair[3,3]); inc(pair[3,3]); end;
        end;
      end;
    end;
  for y:=0 to G_level do
    for x:=0 to G_level do R:=R+pair[y,x];

    for y:=0 to G_level do
      for x:=0 to G_level do f[y,x]:=pair[y,x]/R;
    end;
  end;

```

Procedure Calculate_R_Theta_135;

```

var x,y : integer;
begin
  R:=0;
  for y:=0 to G_level do
    for x:=0 to G_level do pair[y,x]:=0;
  for y:=2 to 64 do
    for x:=2 to 64 do
      case m_dot[x,y] of
        0 : case m_dot[x-1,y-1] of
          0 : begin inc(pair[0,0]); inc(pair[0,0]); end;
          1 : begin inc(pair[0,1]); inc(pair[1,0]); end;
          2 : begin inc(pair[0,2]); inc(pair[2,0]); end;
          3 : begin inc(pair[0,3]); inc(pair[3,0]); end;
        end;
      end;
    end;
  end;

```

```

1 : case m_dot[x-1,y-1] of
  0 : begin inc(pair[1,0]); inc(pair[0,1]); end;
  1 : begin inc(pair[1,1]); inc(pair[1,1]); end;
  2 : begin inc(pair[1,2]); inc(pair[2,1]); end;
  3 : begin inc(pair[1,3]); inc(pair[3,1]); end;
end;
2 : case m_dot[x-1,y-1] of
  0 : begin inc(pair[2,0]); inc(pair[0,2]); end;
  1 : begin inc(pair[2,1]); inc(pair[1,2]); end;
  2 : begin inc(pair[2,2]); inc(pair[2,2]); end;
  3 : begin inc(pair[2,3]); inc(pair[3,2]); end;
end;
3 : case m_dot[x-1,y-1] of
  0 : begin inc(pair[3,0]); inc(pair[0,3]); end;
  1 : begin inc(pair[3,1]); inc(pair[1,3]); end;
  2 : begin inc(pair[3,2]); inc(pair[2,3]); end;
  3 : begin inc(pair[3,3]); inc(pair[3,3]); end;
end;
end;
for y:=0 to G_level do
  for x:=0 to G_level do R:=R+pair[y,x];

  for y:=0 to G_level do
    for x:=0 to G_level do f[y,x]:=pair[y,x]/R;
  end;
end;

```

```

Procedure Cal_Energy;
var x,y : integer;
begin
  Energy:=0.0;
  for y:=0 to G_level do
    for x:=0 to G_level do Energy:=Energy+sqr(f[y,x]);
  end;
end;

```

```

Procedure Cal_Entropy;
var x,y : integer;
begin
  Entropy:=0.0;
  for y:=0 to G_level do
    for x:=0 to G_level do if f[y,x]>0 then Entropy:=Entropy-(f[y,x]*ln(f[y,x]));
  end;
end;

```

```

Procedure Cal_Inertia;
var x,y : integer;
begin
  Inertia:=0.0;
  for y:=0 to G_level do
    for x:=0 to G_level do Inertia:=Inertia+(sqr(y-x)*f[y,x]);
  end;
end;

```

```

Procedure Cal_Local_Homogeneity;
var x,y : integer;
begin

```

```

Local_Homogeneity:=0.0;
for y:=0 to G_level do
  for x:=0 to G_level do Local_Homogeneity:=Local_Homogeneity+(f[y,x]/(1+sqr(x-y)));
end;

```

```

Procedure Cal_Hx;
var x,y : integer;
    dummy : extended;
begin
  Hx:=0.0;
  for y:=0 to G_level do
    begin
      dummy:=0.0;
      for x:=0 to G_level do dummy:=dummy+f[y,x];
      Hx:=Hx+y*dummy;
    end;
  end;
end;

```

```

Procedure Cal_Hy;
var x,y : integer;
    dummy : extended;
begin
  Hy:=0.0;
  for x:=0 to G_level do
    begin
      dummy:=0.0;
      for y:=0 to G_level do dummy:=dummy+f[y,x];
      Hy:=Hy+x*dummy;
    end;
  end;
end;

```

```

Procedure Cal_Sigma_x;
var x,y : integer;
    dummy : extended;
begin
  Sigma_x:=0.0;
  for y:=0 to G_level do
    begin
      dummy:=0.0;
      for x:=0 to G_level do dummy:=dummy+f[y,x];
      Sigma_x:=sigma_x+sqr(y-Hx)*dummy;
    end;
  end;
end;

```

```

Procedure Cal_Sigma_y;
var x,y : integer;
    dummy : extended;
begin
  Sigma_y:=0.0;
  for x:=0 to G_level do
    begin
      dummy:=0.0;
      for y:=0 to G_level do dummy:=dummy+f[y,x];

```

```

        Sigma_y:=Sigma_y+sqr(x-Ny)*dummy;
    end;
end;

```

```

Procedure All_Properties;
begin
    Cal_Mx;
    Cal_My;
    Cal_Sigma_x;
    Cal_Sigma_y;
end;

```

```

Procedure Cal_Correlation;
var x,y : integer;
begin
    Correlation:=0;
    All_Properties;
    for y:=0 to G_level do
        for x:=0 to G_level do Correlation:=Correlation+(y-mx)*(x-
my)*(f[y,x]/(sqr(sigma_x)*sqr(sigma_y)));
    end;
end;

```

```

Procedure All_Features(ofs : integer);
begin
    Cal_Energy;      val_unknown[1+ofs]:=Energy;
    Cal_Entropy;     val_unknown[2+ofs]:=Entropy;
    Cal_Inertia;     val_unknown[3+ofs]:=Inertia;
    Cal_Local_Homogeneity; val_unknown[4+ofs]:=Local_Homogeneity;
    Cal_Correlation; val_unknown[5+ofs]:=Correlation;
end;

```

```

Procedure Calculate;
begin
    Screset;
    Calculate_R_Theta_0;
    All_Features(0);
    Mid_Frame(2,2,39,11,'d=1 Theta=0',Palet[9]);
    str(Energy:2:8,Buffer[15]);
    PlaceStr(4,5,'Energy.....: '+buffer[15]);
    str(Entropy:2:8,Buffer[15]);
    PlaceStr(4,6,'Entropy.....: '+buffer[15]);
    str(Inertia:2:8,Buffer[15]);
    PlaceStr(4,7,'Inertia.....: '+buffer[15]);
    str(Local_Homogeneity:2:8,Buffer[15]);
    PlaceStr(4,8,'L_Homogeneity: '+buffer[15]);
    str(Correlation:2:8,Buffer[15]);
    PlaceStr(4,9,'Correlation...: '+buffer[15]);

    Calculate_R_Theta_45;
    All_Features(5);
    Mid_Frame(41,2,76,11,'d=1 Theta=45',Palet[9]);
    str(Energy:2:8,Buffer[15]);
    PlaceStr(43,5,'Energy.....: '+buffer[15]);

```

```

str(Entropy:2:8,Buffer[15]);
PlaceStr(43,6,'Entropy.....: '+buffer[15]);
str(Inertia:2:8,Buffer[15]);
PlaceStr(43,7,'Inertia.....: '+buffer[15]);
str(Local_Homogeneity:2:8,Buffer[15]);
PlaceStr(43,8,'L_Homogeneity: '+buffer[15]);
str(Correlation:2:8,Buffer[15]);
PlaceStr(43,9,'Correlation...: '+buffer[15]);

Calculate_R_Theta_90;
All_Features(10);
Mid_Frame(2,13,39,22,'d=1 Theta=90',Palet[9]);
str(Energy:2:8,Buffer[15]);
PlaceStr(4,16,'Energy.....: '+buffer[15]);
str(Entropy:2:8,Buffer[15]);
PlaceStr(4,17,'Entropy.....: '+buffer[15]);
str(Inertia:2:8,Buffer[15]);
PlaceStr(4,18,'Inertia.....: '+buffer[15]);
str(Local_Homogeneity:2:8,Buffer[15]);
PlaceStr(4,19,'L_Homogeneity: '+buffer[15]);
str(Correlation:2:8,Buffer[15]);
PlaceStr(4,20,'Correlation...: '+buffer[15]);

Calculate_R_Theta_135;
All_Features(15);
Mid_Frame(41,13,78,22,'d=1 Theta=135',Palet[9]);
str(Energy:2:8,Buffer[15]);
PlaceStr(43,16,'Energy.....: '+buffer[15]);
str(Entropy:2:8,Buffer[15]);
PlaceStr(43,17,'Entropy.....: '+buffer[15]);
str(Inertia:2:8,Buffer[15]);
PlaceStr(43,18,'Inertia.....: '+buffer[15]);
str(Local_Homogeneity:2:8,Buffer[15]);
PlaceStr(43,19,'L_Homogeneity: '+buffer[15]);
str(Correlation:2:8,Buffer[15]);
PlaceStr(43,20,'Correlation...: '+buffer[15]);
end;

begin
  if sec=0 then
    begin
      Screset;
      Mes_Frame(20,10,60,14,PALET[5]);
      PlaceStr(30,11,'!!! W A R N I N G !!!');
      PlaceStr(30,12,'No Texture in Memory');
      PlaceStr(23,13,'Please, Load or Generate a Texture');
      Waitkey; Exit;
    end;
    G_level:=Gray_level-1;
    Calculate;
    Waitkey;
    gotoxy(1,24); write(clen('',32),'W - Write to File');
    choice:=readkbd;
    case choice of
      'W','w' : begin
        buffer[15]:=f_name+'.1';

```

```
        assign(bfile,buffer[15]);  
        rewrite(bfile);  
        for i:=1 to 20 do writeln(bfile,val_unknown[i]);  
        close(bfile);  
    end;  
end;  
waitkey;  
end;  
end.
```



/* Dosya işlemleri ve Servis Programları Bölümü */

unit SrvBox;

{ \$N+ }

interface

uses crt,dos,psrvbox,menubox,Graph;

type

 m_lattice = array [-1..66] of array [-1..66] of integer;

 texture_rec = record

 mod_name : str15;

 g_level : integer;

 order_level : integer;

 iter_num : integer;

 param_alpha : double;

 param_beta : array [1..10] of double;

 texel : m_lattice;

 end;

texture_file = file of texture_rec;

special_rec = record

 tex : m_lattice;

end;

special_file = file of special_rec;

const c1 : array [1..4] of integer = (1,180,1,180);

 c2 : array [1..4] of integer = (30,30,50,50);

var

 texture : texture_rec;

 special : special_rec;

 file_var : texture_file;

 special_var : special_file;

 markov_dots : m_lattice;

 GraphDriver,

 GraphNode,gray_level,

 LX,LL,iter,

 order,sec : integer;

 bfile : text;

 f_name : str25;

 BUFFER : array [1..15] of str25;

 Model_name : str15;

 a : double;

 b : array [1..5] of array [1..2] of double;

 changes,sifir,bir,iki,uc,dort,bes,alti,yedi : array [1..100] of integer;

 ini_zero,ini_one,ini_two,ini_three,

 ini_four,ini_five,ini_six,ini_seven : integer;

```

Function Power(x : double; y : integer):extended;
Function Fact(x : integer):extended;
Function Conb(g,k : integer):extended;
Procedure File_Processing(var m_dot : m_lattice);
procedure Clear_Buffer;
procedure Change_to_Graphics;
procedure Put_Pixel(x1,y1 : integer; renk : word; base_x,base_y : integer);
Procedure Show_Number_Of_Gray_Value;

```

IMPLEMENTATION

```

procedure Change_to_Graphics;
begin
  InitGraph(GraphDriver,GraphMode,'');
end;

procedure Put_Pixel(x1,y1 : integer; renk : word; base_x,base_y : integer);
begin
  case GraphDriver of
    CGA,HCSA,ATT480,
    VGA           : if (GraphMode=4) or (GraphMode=5) then
                     begin
                       PutPixel(((2*x1-1)+base_x,y1+base_y,renk);
                       PutPixel(2*x1+base_x,y1+base_y,renk);
                     end
                     else PutPixel(x1+(base_x div 2),y1+base_y,renk);
    EGA,EGA64,EGANono,
    HercMono,PC3270 : begin
                       PutPixel(((2*x1-1)+base_x,y1+base_y*2,renk);
                       PutPixel(2*x1+base_x,y1+base_y*2,renk);
                     end;
  end;
end;

Procedure Show_Number_Of_Gray_Value;
var i : integer;

begin
  Hide_cursor;
  PlaceStr(1,1,
    '  Changes   0     1     2     3     4     5     6     7');
  PlaceStr(1,2,
    '-----');
  gotoxy(1,3);
  writeln('  Initial', ' ',ini_zero:6, ' ',ini_one:6, ' ',ini_two:6,
    ' ',ini_three:6, ' ',ini_four:6, ' ',ini_five:6,
    ' ',ini_six:6, ' ',ini_seven:6);
  PlaceStr(1,4,
    '-----');
  gotoxy(1,5);

  for i:=1 to iter do
    begin

```

```

        writeln(i:3, ' ', changes[i]:6, ' ', sifir[i]:6, ' ', bir[i]:6, ' ',
            iki[i]:6, ' ', uc[i]:6, ' ', dort[i]:6, ' ', bes[i]:6, ' ', alti[i]:6, ' ', yedi[i]:6);
        if (i mod 20)=0 then begin waitkey; region_color(1,5,80,25,0); gotoxy(1,5); end;
    end;
    waitkey;
end;

```

```

Function Power(x : double; y : integer):extended;
    var z : extended;
        i : integer;
    begin
        z:=1.0; i:=y;
        while i>0 do
            begin
                if Odd(i) then z:=z*x;
                i:=i div 2;
                x:=Sqr(x);
            end;
        Power:=z;
    end;
end;

```

```

Function Fact(x : integer):extended;
    var i : integer;
        temp : extended;
    begin
        temp:=1;
        if x<>0 then
            for i:=1 to x do temp:=temp*i;
        Fact:=temp;
    end;
end;

```

```

Function Comb(g,k : integer):extended;
    begin
        Comb:=Fact(g)/(Fact(k)*Fact(g-k));
    end;
end;

```

```

Procedure Clear_Buffer;
    begin for i:=1 to 15 do BUFFER[i]:=''; end;

```

```

Procedure File_Processing(var m_dot : m_lattice);
    /* Dosya islenleri */
    var select,x,y,rnd : integer;
        s : str2;

    procedure Creat_File(file_name : str10);
        begin
            assign(file_var,file_name);
            rewrite(file_var);
        end;

```

```
Procedure Write_File_Of_Gray_Value(dosya : str13);
```

```
var i : integer;
```

```
begin
```

```
  assign(bfile,dosya+'.CHG');
```

```
  rewrite(bfile);
```

```
  writeln(bfile,'  Changes    0    1    2    3    4    5    6    7');
```

```
  writeln(bfile,'-----');
```

```
  writeln(bfile,'  Initial', ' ',ini_zero:6, ' ',ini_one:6, ' ',ini_two:6,
```

```
    ' ',ini_three:6, ' ',ini_four:6, ' ',ini_five:6,
```

```
    ' ',ini_six:6, ' ',ini_seven:6);
```

```
  writeln(bfile,'-----');
```

```
  for i:=1 to iter do
```

```
    writeln(bfile,i:3, ' ',changes[i]:6, ' ',sifir[i]:6, ' ',bir[i]:6, ' ',iki[i]:6, ' ',uc[i]:6,
```

```
    ' ',dort[i]:6, ' ',bes[i]:6, ' ',alti[i]:6, ' ',vedi[i]:6);
```

```
  close(bfile);
```

```
end;
```

```
Procedure Display_Number_Of_Gray_Value(dosya : str13);
```

```
var line : str80;
```

```
  i : integer;
```

```
begin
```

```
  Screset;
```

```
  assign(bfile,dosya+'.CHG');
```

```
  reset(bfile);
```

```
  readln(bfile,line);
```

```
  PlaceStr(1,1,line);
```

```
  readln(bfile,line);
```

```
  PlaceStr(1,2,line);
```

```
  readln(bfile,line);
```

```
  gotoxy(1,3); writeln(line);
```

```
  readln(bfile,line);
```

```
  PlaceStr(1,4,line);
```

```
  gotoxy(1,5);
```

```
  i:=1;
```

```
  while not EOF(bfile) do
```

```
    begin
```

```
      readln(bfile,line);
```

```
      writeln(line);
```

```
      if (i mod 20)=0 then begin waitkey; region_color(1,5,80,25,0); gotoxy(1,5); end;
```

```
      inc(i);
```

```
    end;
```

```
  waitkey;
```

```
end;
```

```
Procedure Open_File(file_name : str13);
```

```
begin
```

```
  assign(file_var,file_name);
```

```
  reset(file_var);
```

```
end;
```

```

begin
  TC:='';
  Repeat
    Sub[1]:='Save Texture';
    Sub[2]:='Load Texture';
    Sub[3]:='View Texture';
    Sub[4]:='Show Changes';
    Sub[5]:='Special Save';
    Sub[6]:='Special load';
    Sub[7]:='Special View';
    Sub[8]:='Add Noise';
    W_DIR:=True;
    Store_SCR;
    HidMenu(select,8,35,19,18,'Files');
    case select of
      0 : begin DEC(SCR_LEVEL); EXIT; end;
      1 : begin
        Mes_Frame(42,15,74,17,Palet[5]);
        PlaceStr(44,16,Dots('File Name',11));
        inputstr(f_name,15,56,16,Tern,TC);
        if TC<>'Z' then
          begin
            if f_name='' then
              begin
                Mes_Frame(59,16,77,18,Palet[4]);
                PlaceStr(61,17,'Wrong File Name');
                waitkey; EXIT;
              end;
            Mes_Frame(65,16,77,18,Palet[4]);
            PlaceStr(67,17,'Saving...');
            Creat_File(f_name);
            texture.mod_name:=model_name;
            texture.g_level:=gray_level;
            texture.order_level:=order;
            texture.iter_sum:=iter;
            texture.param_alpha:=a;
            for i:=1 to order do
              begin
                texture.param_beta[2*i-1]:=b[i,1];
                texture.param_beta[2*i]:=b[i,2];
              end;
            for y:=-1 to 66 do
              for x:=-1 to 66 do texture.texel[x,y]:=m_dot[x,y];
            write(file_var,texture);
            Close(file_var);
            Write_File_Of_Gray_Value(f_name);
          end;
        end;
      2 : begin
        Mes_Frame(42,16,74,18,Palet[5]);
        PlaceStr(44,17,Dots('File Name',11));
        inputstr(f_name,15,56,17,Tern,TC);
        if TC<>'Z' then
          begin
            if f_name='' then
              begin
                Mes_Frame(59,17,77,19,Palet[4]);

```

```

        PlaceStr(61,18,'Wrong File Name');
        waitkey; EXIT;
    end;
    Mes_Frame(65,17,77,19,Palet[4]);
    PlaceStr(67,18,'Loading...');
    open_file(f_name);
    read(file_var,texture);
    model_name:=texture.mod_name;
    if Model_Name='Auto Binomial' then sec:=1 else sec:=2;
    gray_level:=texture.g_level;
    order:=texture.order_level;
    iter:=texture.iter_num;
    a:=texture.param_alpha;
    for i:=1 to order do
        begin
            b[1,1]:=texture.param_beta[2*i-1];
            b[1,2]:=texture.param_beta[2*i];
        end;
    for y:=-1 to 66 do
        for x:=-1 to 66 do m_dot[x,y]:= texture.texel[x,y];
        close(file_var);
    end;
end;
3 : begin
    Change_to_Graphics;
    outtextxy(1,1,'Model Name: '+model_name);
    str(gray_level:2,BUFFER[15]);
    outtextxy(1,10,'Gray Level: '+BUFFER[15]);
    str(order:2,BUFFER[15]);
    outtextxy(10,10,'Order: '+BUFFER[15]);
    str(iter:4,BUFFER[15]);
    outtextxy(1,20,'Iteration: '+BUFFER[15]);
    str(a:2:4,BUFFER[15]);
    outtextxy(10,20,'a: '+BUFFER[15]);
    for i:=1 to order do
        begin
            str(i,s);
            str(b[1,1]:2:4,BUFFER[15]);
            outtextxy(c1[i],c2[i],b['+s+',1]:'+BUFFER[15]);
            str(b[1,2]:2:4,BUFFER[15]);
            outtextxy(c1[i],c2[i]+10,b['+s+',2]:'+BUFFER[15]);
        end;
    for y:=1 to 64 do
        for x:=1 to 64 do Put_Pixel(x,y,m_dot[x,y],300,100);
        waitkey;
    CloseGraph;
    end;
4 : begin
    Screset;
    if sec=0 then
        begin
            Mes_Frame(20,10,60,14,PALET[5]);
            PlaceStr(30,11,'!!! W A R N I N G !!!');
            PlaceStr(30,12,'No Texture in Memory');
            PlaceStr(23,13,'Please, Load or Generate a Texture');
            Waitkey; Exit;
        end;

```

```

        if f_name<>' ' then Display_Number_Of_Gray_Value(f_name)
        else Show_Number_Of_Gray_Value;
    end;

5 : begin
    Mes_Frame(42,15,67,17,Palet[5]);
    PlaceStr(44,16,Dots('File Name',11));
    inputstr(f_name,8,56,16,Tern,TC);
    if TC<>'Z' then
        begin
            if f_name=' ' then
                begin
                    Mes_Frame(59,16,77,18,Palet[4]);
                    PlaceStr(61,17,'Wrong File Name');
                    waitkey; EXIT;
                end;
            Mes_Frame(65,16,77,18,Palet[4]);
            PlaceStr(67,17,'Saving...');
            assign(special_var,f_name);
            rewrite(special_var);
            for y:=-1 to 66 do
                for x:=-1 to 66 do special.tex[x,y]:=m_dot[x,y];
            write(special_var,special);
            Close(special_var);
            Write_File_Of_Gray_Value(F_name);
        end;
    end;

6 : begin
    Mes_Frame(42,16,67,18,Palet[5]);
    PlaceStr(44,17,Dots('File Name',11));
    inputstr(f_name,8,56,17,Tern,TC);
    if TC<>'Z' then
        begin
            if f_name=' ' then
                begin
                    Mes_Frame(59,17,77,19,Palet[4]);
                    PlaceStr(61,18,'Wrong File Name');
                    waitkey; EXIT;
                end;
            Mes_Frame(65,17,77,19,Palet[4]);
            PlaceStr(67,18,'Loading...');
            sec:=1;
            assign(special_var,f_name);
            reset(special_var);
            read(special_var,special);
            for y:=-1 to 66 do
                for x:=-1 to 66 do m_dot[x,y]:= special.tex[x,y];
            close(special_var);
        end;
    end;

7 : begin
    Change_to_Graphics;
    outtextxy(1,1,'Special ');
    for y:=1 to 64 do
        for x:=1 to 64 do Put_Pixel(x,y,m_dot[x,y],300,100);
    waitkey;
    CloseGraph;

```

```

end;
8 : begin
    Randomize;
    Mes_Frame(65,17,77,19,Palet[4]);
    PlaceStr(67,18,'Adding...');
    for y:=1 to 64 do
        for x:=1 to 64 do
            begin
                rnd:=round(random*2)-1;
                m_dot[x,y]:=m_dot[x,y]+rnd;
                if m_dot[x,y]<0 then m_dot[x,y]:=0;
                if m_dot[x,y]>gray_level-1 then m_dot[x,y]:=gray_level-1;
            end;
        end;
    end;
    Load_SCR
    Until (YC='Z');
end;
end.

```

```
/* Grafik Ekran Seçimi ile ilgili Bölüm */
```

```
unit SetupBox;
```

```
{$N+}
```

```
interface
```

```
uses crt,psrvbox,menubox,Graph;
```

```
Procedure Show_Screen_Status(Graph_Dr,Graph_Mod : integer);
```

```
Procedure Setup_Graphics(var Driver, Mode : integer);
```

IMPLEMENTATION

```
var gr_drv, gr_mode : str10;
    colors : array [1..16] of str20;
    ek,i,j : integer;
```

```
Procedure Take_Colors;
```

```
begin
    colors[1] := '[0] Black';
    colors[2] := '[1] Blue';
    colors[3] := '[2] Green';
    colors[4] := '[3] Cyan';
    colors[5] := '[4] Red';
    colors[6] := '[5] Magenta';
    colors[7] := '[6] Brown';
    colors[8] := '[7] LightGray';
    colors[9] := '[8] DarkGray';
    colors[10] := '[9] LightBlue';
    colors[11] := '[10] LightGreen';
    colors[12] := '[11] LightCyan';
    colors[13] := '[12] LightRed';
    colors[14] := '[13] LightMagenta';
    colors[15] := '[14] Yellow';
    colors[16] := '[15] White';
end;
```

```
Procedure Take_Mono_Color;
```

```
begin
    colors[1] := '[0] Black';
    colors[2] := '[1] White';
end;
```

```
Procedure Set_Status(Graph_Dr, Graph_Mod : integer);
```

```
begin
    if Graph_Dr=DETECT then DetectGraph(Graph_Dr,Graph_Mod);
    case Graph_Dr of
        CGA,HCGA,
        ATT400 : begin
            case Graph_Dr of
                1 : gr_drv:='CGA';
                2 : gr_drv:='HCGA';
                3 : gr_drv:='ATT400';
            end;
```

```

case Graph_Mod of
  0 : begin
    ek:=2; gr_mode:='320x200';
    colors[1]:='[0] Black';
    colors[2]:='[1] LightGreen';
    colors[3]:='[2] LightRed';
    colors[4]:='[3] Yellow';
  end;
  1 : begin
    ek:=2; gr_mode:='320x200';
    colors[1]:='[0] Black';
    colors[2]:='[1] LightCyan';
    colors[3]:='[2] LightMagenta';
    colors[4]:='[3] White';
  end;
  2 : begin
    ek:=2; gr_mode:='320x200';
    colors[1]:='[0] Black';
    colors[2]:='[1] Green';
    colors[3]:='[2] Red';
    colors[4]:='[3] Brown';
  end;
  3 : begin
    ek:=2; gr_mode:='320x200';
    colors[1]:='[0] Black';
    colors[2]:='[1] Cyan';
    colors[3]:='[2] Magenta';
    colors[4]:='[3] LightGray';
  end;
  4 : begin
    ek:=0; gr_mode:='640x200';
    Take_Mono_Color;
  end;
  5 : begin
    ek:=0;
    if Graph_Dr=HCGA then
      gr_mode:='640x400'
    else gr_mode:='640x400';
    Take_Mono_Color;
  end;
end;
end;
EGA : begin
  ek:=6; gr_drv:='EGA';
  case Graph_Mod of
    EGALo : gr_mode:='640x200';
    EGANi : gr_mode:='640x350';
  end;
  Take_Colors;
end;
EGA64 : begin
  gr_drv:='EGA64';
  case Graph_Mod of
    EGA64Lo : begin
      ek:=6; gr_mode:='640x200';
      Take_Colors;
    end;
  end;
end;

```

```

        EGA64Hi : begin
            ek:=2; gr_mode:='640x350';
            colors[1]:='{0}';
            colors[2]:='{1}';
            colors[3]:='{2}';
            colors[4]:='{3}';
        end;
    end;
end;
EGAMono : begin
    ek:=0; gr_drv:='EGAMono';
    gr_mode:='640x350';
    Take_Mono_Color;
end;
IBH8514 : begin
    ek:=1; gr_drv:='IBH8514';
    case Graph_Mod of
        IBH8514LO : gr_mode:='640x480';
        IBH8514HI : gr_mode:='1024x768';
    end;
    colors[1]:='256 color';
end;
HercMono : begin
    ek:=0; gr_drv:='HercMono';
    gr_mode:='720x348';
    Take_Mono_Color;
end;
VGA : begin
    ek:=6; gr_drv:='VGA';
    case Graph_Mod of
        0 : gr_mode:='640x200';
        1 : gr_mode:='640x350';
        2 : gr_mode:='640x480';
    end;
    Take_Colors;
end;
PC3270 : begin
    ek:=0; gr_drv:='PC3270';
    gr_mode:='720x350';
    Take_Mono_Color;
end;
end;
end;

Procedure Show_Screen_Status(Graph_Dr,Graph_Mod : integer);
begin
    Set_Status(Graph_Dr,Graph_Mod);
    Hid_Frame(14,8,44,14,'Current Status',Palet[8]);
    PlaceStr(17,11,Dots('GraphDriver',15)+gr_drv);
    PlaceStr(17,12,Dots('GraphNode',15)+gr_mode);
    case ek of
        0 : begin
            Hid_Frame(19,13,39,18,'Colors',Palet[5]);
            PlaceStr(21,16,colors[1]);
            PlaceStr(21,17,colors[2]);
        end;
        1 : begin

```

```

        Mid_Frame(19,13,39,17,'Colors',Palet[5]);
        PlaceStr(21,16,colors[1]);
    end;
2 : begin
        Mid_Frame(19,13,39,20,'Colors',Palet[5]);
        for i:=1 to 4 do PlaceStr(21,15+i,colors[i]);
    end;
6 : begin
        Mid_Frame(19,13,59,24,'Colors',Palet[5]);
        for i:=1 to 8 do
            begin
                PlaceStr(21,15+i,colors[i]);
                PlaceStr(40,15+i,colors[i+8]);
            end;
        end;
    end;
end;
waitkey;
end;

```

```

Procedure Setup_Graphics(var Driver, Mode : integer);
var select : integer;
begin
    Repeat
        Sub[1]:= 'Default Mode';
        Sub[2]:= 'Change Current Mode';
        W_DIR:=True;
        ClockON;
        MidMenu(select,2,62,13,15,'Setup');
        ClockOFF;
        Store_SCR;
        if select=0 then begin DEC(SCR_LEVEL); EXIT; end;
        if select=1 then begin Driver:=DETECT; Load_SCR; EXIT; end
        else begin
            if Driver=DETECT then DetectGraph(Driver,Mode);
            ClockON;
            Sub[1]:= 'CGA';
            Sub[2]:= 'HCGA';
            Sub[3]:= 'EGA';
            Sub[4]:= 'EGA64';
            Sub[5]:= 'EGAMono';
            Sub[6]:= 'IBM8514';
            Sub[7]:= 'HERCULES';
            Sub[8]:= 'ATT400';
            Sub[9]:= 'VGA';
            Sub[10]:= 'PC3270';
            MidMenu(Driver,10,46,20,18,'GraphDriver');
            inc(Mode);
            Case Driver of
                1 : begin
                    Sub[1]:= 'CGAC0';
                    Sub[2]:= 'CGAC1';
                    Sub[3]:= 'CGAC2';
                    Sub[4]:= 'CGAC3';
                    Sub[5]:= 'CGAH1';
                    MidMenu(Mode,5,55,23,18,'GraphNode');
                    if Mode<>0 then
                        begin

```

```

    dec(Node);
    Set_Status(Driver,Node);
    if (Node>=0) and (Node<4) then
    begin
        Hes_Frame(61,17,80,22,palet[4]);
        for i:=1 to 4 do Placestr(63,17+i,colors[i]);
    end else
    begin
        Hes_Frame(63,17,77,20,palet[4]);
        for i:=1 to 2 do Placestr(65,17+i,colors[i]);
    end;
    Waitkey;
end;
2 : begin
    Sub[1]:= 'MCGAC0';
    Sub[2]:= 'MCGAC1';
    Sub[3]:= 'MCGAC2';
    Sub[4]:= 'MCGAC3';
    Sub[5]:= 'MCGAMed';
    Sub[6]:= 'MCGAH1';
    HidMenu(Node,6,55,23,18,'GraphMode');
    if Node<>0 then
    begin
        dec(Node);
        Set_Status(Driver,Node);
        if (Node>=0) and (Node<4) then
        begin
            Hes_Frame(61,17,80,22,palet[4]);
            for i:=1 to 4 do Placestr(63,17+i,colors[i]);
        end else
        begin
            Hes_Frame(63,17,77,20,palet[4]);
            for i:=1 to 2 do Placestr(65,17+i,colors[i]);
        end;
        Waitkey;
    end;
end;
3 : begin
    Sub[1]:= 'EGALo';
    Sub[2]:= 'EGAH1';
    HidMenu(Node,2,55,20,18,'GraphMode');
    if Node<>0 then
    begin
        dec(Node);
        Set_Status(Driver,Node);
        Hes_Frame(63,17,77,19,palet[4]);
        Placestr(65,18,' 16 Color');
        Waitkey;
    end;
end;
4 : begin
    Sub[1]:= 'EGA64Lo';
    Sub[2]:= 'EGA64H1';
    HidMenu(Node,2,55,20,18,'GraphMode');
    if Node<>0 then
    begin

```

```

dec(Node);
Set_Status(Driver,Node);
if Node=EGA64Lo then
begin
  Mes_Frame(63,17,77,19,palet[4]);
  Placestr(65,18,' 16 Color');
end
else begin
  Mes_Frame(63,17,77,19,palet[4]);
  Placestr(65,18,' 4 Color');
end;
Waitkey;
end;
end;
5 : begin
  Sub[1]:='EGAMonoHi';
  MidMenu(Node,1,55,19,18,'GraphNode');
  if Node<>0 then
  begin
    dec(Node);
    Set_Status(Driver,Node);
    Mes_Frame(63,17,77,20,palet[4]);
    for i:=1 to 2 do Placestr(65,17+i,Colors[i]);
    Waitkey;
  end;
end;
6 : begin
  Sub[1]:='IBM8514Lo';
  Sub[2]:='IBM8514Hi';
  MidMenu(Node,2,55,20,18,'GraphNode');
  if Node<>0 then
  begin
    dec(Node);
    Set_Status(Driver,Node);
    Mes_Frame(63,17,77,19,palet[4]);
    Placestr(65,18,' '+colors[i]);
    Waitkey;
  end;
end;
7 : begin
  Sub[1]:='HercMonoHi';
  MidMenu(Node,1,55,19,18,'GraphNode');
  if Node<>0 then
  begin
    dec(Node);
    Set_Status(Driver,Node);
    Mes_Frame(63,17,77,20,palet[4]);
    for i:=1 to 2 do Placestr(65,17+i,Colors[i]);
    Waitkey;
  end;
end;
8 : begin
  Sub[1]:='ATT400C0';
  Sub[2]:='ATT400C1';
  Sub[3]:='ATT400C2';
  Sub[4]:='ATT400C3';
  Sub[5]:='ATT400Med';

```

```

Sub[6]:=‘ATT400Hi’;
MidMenu(Node,6,55,23,18,‘GraphNode’);
if Node<>0 then
begin
dec(Node);
Set_Status(Driver,Node);
if (Node>=0) and (Node<4) then
begin
Mes_Frame(61,17,80,22,palet[4]);
for i:=1 to 4 do Placestr(63,17+i,colors[i]);
end
else begin
Mes_Frame(63,17,77,20,palet[4]);
for i:=1 to 2 do Placestr(65,17+i,colors[i]);
end;
Waitkey;
end;
end;
9 : begin
Sub[1]:=‘VGALo’;
Sub[2]:=‘VGAMed’;
Sub[3]:=‘VGAMHi’;
MidMenu(Node,3,55,21,18,‘GraphNode’);
if Node<>0 then
begin
dec(Node);
Set_Status(Driver,Node);
Mes_Frame(63,18,77,20,palet[4]);
Placestr(65,19,‘ 16 Color’);
Waitkey;
end;
end;
10 : begin
Sub[1]:=‘PC3270Hi’;
MidMenu(Node,1,55,19,18,‘GraphNode’);
if Node<>0 then
begin
dec(Node);
Set_Status(Driver,Node);
Mes_Frame(63,17,77,20,palet[4]);
for i:=1 to 2 do Placestr(65,17+i,Colors[i]);
Waitkey;
end;
end;
end;
ClockOFF;
if Driver=0 then Load_SCR
else DEC(SCR_LEVEL);
end;
Until ((select in [1,2]) AND (Driver in [1..10]));
end;
end.

```

```
/* Yardımcı Programlar */
```

```
/* Yardımcı programlar ELEKTROHED A.Ş. ne ait olduğundan */  
/* yazılım ayrıntılı olarak verilmeyecektir. */
```

```
UNIT PdrvBox;
```

```
{ $N+ }
```

```
interface
```

```
uses crt,dos,printer;
```

```
CONST TERM : SET OF CHAR=['I','X','M','E','Z'];
```

```
type Anystr=String[255];
```

```
Str2=String[2];
```

```
Str3=String[3];
```

```
Str5=String[5];
```

```
Str8=String[8];
```

```
Str10=String[10];
```

```
Str13=String[13];
```

```
Str15=String[15];
```

```
Str20=String[20];
```

```
Str25=String[25];
```

```
Str40=String[40];
```

```
Str80=String[80];
```

```
CharSet=Set of Char;
```

```
VAR      renkli      : Boolean;  
        regset      : Registers;  
        tc          : Char;  
        con         : Boolean;  
        l           : LongInt;  
        SCR_LEVEL,  
        VDU_MODE    : Integer;  
        SCR_SEG     : 1..$FFFF;  
        W_DIR       : Boolean;  
        SCR         : Array[1..6,1..4000] Of Byte;  
        PALET       : Array[1..25] Of Integer;  
        LINE_ARRAY  : Array [1..160] Of Byte;  
        MX          : Array[1..10] Of String[30];  
        SUB         : Array [1..15] Of String[30];  
        l,att,nowx,nowy,choice,rep : Integer;
```

```
Procedure Disable;
```

```
Procedure Enable;
```

```
Procedure Hide_Cursor;
```

```
Procedure Show_Cursor;
```

```
Procedure Screset;
```

```
Procedure Waitkey;
```

```
Function Today:Str8;
```

```
Function FillStr(C:Char; N:Integer):Anystr;
```

```
Procedure Mes(A:Str80; Z:Integer);
```

```
Function MidSet(S:anystr; N:Integer):Anystr;
```

```
Function Clen(S:anystr; N:Integer):Anystr;
```

```

Function Dots(s:anystr; n:Integer):Anystr;
Procedure DoubleCon(var a:boolean; lbound,ubound:Double; str:Anystr; var b:Double);
Procedure IntCon(var A:Boolean; lbound,ubound:Integer; str:Anystr; var b:Integer);
Function ReadKbd:Char;
Procedure InputStr(var S:Str80; L,X,Y:Integer; Term :CharSet; var TC:Char);
Procedure IncDec(X:Integer; var LI:Integer; TC:Char);
Procedure Spkeys(L:Integer);
Procedure Frame(X1,Y1,X2,Y2:Integer);
Procedure Mid_Frame(X1,Y1,X2,Y2:Integer; str_:Str80; COLOR:Integer);
Procedure Mes_Frame(X1,Y1,X2,Y2,COLOR:Integer);
Procedure Scroll(DIR,X1,Y1,X2,Y2,LINES,ATTRIB:Integer);
Procedure ScrollUp(X1,Y1,X2,Y2,LINES,ATTRIB:Integer);
Procedure ScrollDown(X1,Y1,X2,Y2,LINES,ATTRIB:Integer);
Procedure Region_Color(X1,Y1,X2,Y2,COLOR:Integer);
Procedure PlaceStr(X,Y : Integer; c:Anystr);
Procedure Ch_Att(X,Y,LEN,ATTR:Integer);
Procedure PlaceC(X,Y:Integer; C:Char);
Procedure Init_Screen;
Procedure Store_Scr;
Procedure Load_Scr;
Procedure Load_Line(LINE:Integer);
Procedure Store_Line(LINE:Integer);
Procedure Clr_Line(X,Y:Integer);
Function Store_Att(x,y:Integer):Integer;
Function Rset(S :Anystr; N:Integer):Anystr;

```

implemmentation

```
/* Yardımcı Programlar */
```

```
/* Yardımcı programlar ELEKTRONİK A.Ş. ne ait olduğundan */  
/* yazılım ayrıntılı olarak verilmeyecektir. */
```

```
UNIT MenuBox;
```

```
{ $N+ }
```

```
Interface
```

```
uses crt,dos,printer,psrvbox;
```

```
Procedure ClockOn;
```

```
Procedure ClockOff;
```

```
Procedure MainMenu(Var sec: Integer; itemnum: Integer;  
                  s: Str40; name : Str40);
```

```
Procedure MidMenu(Var L: Integer; LINES,X,Y,LENGTH: Integer; STR_: Str80);
```

```
Implementation
```

KAYNAKLAR

- [1] AHUJA, N., ve ROSENFELD, A., Mosaic Models for Textures, IEEE Trans. on PAMI-3, pp. 1-10, 1981.
- [2] BERKSON, J., Maximum Likelihood and Minimum Chi-Square Estimates of Logistic Function, JASA, 50, pp. 130-162, 1955.
- [3] BESAG, J., Spatial Interaction and the Statistical Analysis of Lattice Systems, J.R. Statist. Soc. B., 36, pp. 192-236, 1974.
- [4] BESAG, J., On Statistical Analysis of Dirty Pictures, J.R. Statist. Soc. B., 48, pp. 259-302, 1986.
- [5] BRODATZ, P., Textures: A Photographic Album for Artist and Designers, New York: Dover, 1966.
- [6] CHELLAPPA, R., Stochastic Models in Image Analysis and Processing, Ph.D. Thesis, Purdue University, 1981.
- [7] CHELLAPPA, R., CHATTERJEE, S., BAGDAZIAN, R., Textures Synthesis and Compression Using Gaussian- Markov Random Field Models, IEEE Trans. on SMC-33, pp. 298-303, 1985.
- [8] CHELLAPPA, R., CHATTERJEE, S., Classification of Textures Using Gaussian-Markov Random Fields, IEEE Trans. on ASSP-33, 1985.
- [9] CHEN, C.C., Markov Random Fields in Image Analysis, Ph.D. Thesis, Michigan State University, 1988.
- [10] COGGINS, J.M., ve JAIN, A.K., Spatial Filtering Approach to Texture Analysis, Pattern Recognition Letters, 3, pp. 195-203, 1985.

- [11] CONNERS, R.W., ve HARLOW, C.A., A Theoretical Comparison of Texture Algorithms, IEEE Trans. on PAMI-2, pp. 204-222, 1980.
- [12] CROSS, G.R., Markov Random Field Texture Models, Ph.D. Thesis, Michigan State University, 1980.
- [13] CROSS, G.R., ve JAIN A.K., Markov Random Field Texture Models, IEEE Trans. on PAMI-5, pp. 25-39, 1983.
- [14] DELP, E.J., ve KASHYAP, R.L., Image Data Compression Using Autoregressive Times Series Models, Pattern Recognition, 11, pp. 313-323, 1979.
- [15] DERIN, H., ELLIOTT, H., CRISTI, R., ve GEMAN, D., Bayes Smoothing Algorithms for Segmentation of Binary Images Modeled by Markov Random Fields, IEEE Trans. on PAMI-6, pp. 707-720, 1984.
- [16] DERIN, H., ve ELLIOTT, H., Modeling and Segmentation of Noisy and Textured Images Using Gibbs Random Fields, IEEE Trans. on PAMI-9, pp. 39-55, 1987.
- [17] DERIN, H., ve KELLY, P.A., Discrete-Index Markov-Typed Random Process, Proc. IEEE, vol. 77, 10, pp. 1485-1509, 1989.
- [18] de SOUZA, P., Texture Recognition via Autoregression, Pattern Recognition, 15, pp. 471-475, 1982.
- [19] DUBES, R.C., ve JAIN, A.K., Random Field Models in Image Analysis, Journal of Applied Statistics, vol. 16, 2, 1989.
- [20] DUDA, R.O., ve HART, P.E., Pattern Classification and Scene Analysis, New York: Wiley, 1973.
- [21] FAUGERAS, O.D., PRATT, W.K., Decorrelation Methods of Texture Feature Extraction, IEEE Trans. on PAMI-2, pp. 323-332, 1980.

- [22] GEMAN, S., ve GEMAN, D., Stochastic Relaxation: Gibbs Distributions, and the Bayesian Restoration of Images, IEEE Trans. on PAMI-6, pp. 721-741, 1984.
- [23] HARALICK, R.M., SHANMUGAM, K., DINSTEIN, I., Textural Features for Image Classification, IEEE Trans. on SMC-3, pp. 610-621, 1973.
- [24] HARALICK, R.M., Statistical and Structural Approaches to Texture, Proc. IEEE, 67, pp. 786-804, 1979.
- [25] HASSNER, M., ve SKLANSKY, J., The Use of Markov Random Fields as Models of Texture, CGIP, 12, pp. 357-370, 1980.
- [26] KANEKO, H., ve YODOGAWA, E., A Markov Random Field Application to Texture Classification, Proc. IEEE Conf. PRIP, pp. 221-225, 1982.
- [27] KASHYAP, R.L., ve LAPSA, P.M., Synthesis and Estimation of Random Fields Using Long Correlation Models, IEEE Trans. on PAMI-6, pp. 800-809, 1984.
- [28] KHOTANZAD, A., ve KASHYAP, R., Feature Selection for Texture Recognition Based on Image Synthesis, IEEE Trans. on SMC, 17, pp. 1087-1095, 1987.
- [29] KINDERMAN, R., ve SNELL, J.L., Markov Random Fields and Their Applications, American Mathematical Society, Volume I, 1980.
- [30] LAWS, K.I., Textured Image Segmentation, Ph.D. Thesis, University of Southern California, 1979.
- [31] LU, S.Y., ve FU, K.S., Stochastic Tree Grammer Inference for Texture Synthesis and Discrimination, CGIP, 9, pp. 234-245, 1979.
- [32] McCORMICK, B.H., ve JAYARAMAMURTHY, S.N., Time Series Model for Texture Synthesis, Int. J. Com. & Infor. Sci., 3, pp. 329-343, 1974.

- [33] PENTLAND, A.P., Fractal-Based Description of Natural Scenes, IEEE Trans. on PAMI-6, pp. 661-674, 1984.
- [34] SIEW, L.H., HODGSON R.M., ve WOOD E.J., Texture Measures for Carpet Wear Assessment, IEEE Trans. on PAMI-10, pp. 92-105, 1988.
- [35] TAMURA, H., MORI, S., ve YAMAWAKI, T., Textural Features Corresponding to Visual Perception, IEEE Trans. on SMC-8, pp. 460-470, 1978.
- [36] VAN GOOL, L., DEWAELE, P. ve DOSTERINCK, A., Survey: Texture Analysis Anno 1983, CVGIP, 29, pp. 336-357, 1985.
- [37] WESZKA, J., DYER, C.R., ve ROSENFELD, A., A comparative Study of Texture Measures for Terrain Classification, IEEE Trans. on SMC-6, pp. 269-285, 1976.

OZGEÇMİŞ

1965 istanbul doğumludur. 1983 yılında MAÇKA TEKNİK LİSESİ Elektronik Bölümünü, 1987 yılında da HACETTEPE ÜNİVERSİTESİ Elektronik Mühendisliğini bitirmiştir. Özel bir şirkette çalışmaktadır.

