目录

[1. 背景 1](#_Toc433634151)

[2. 纹理理论和方法 2](#_Toc433634152)

[2.1 扩展纹理集 2](#_Toc433634153)

[2.2.1 均值和协方差矩阵 2](#_Toc433634154)

[2.2.2特征值和特征向量 3](#_Toc433634155)

[2.2.3 中心化 3](#_Toc433634156)

[2.2.4 随机生成 4](#_Toc433634157)

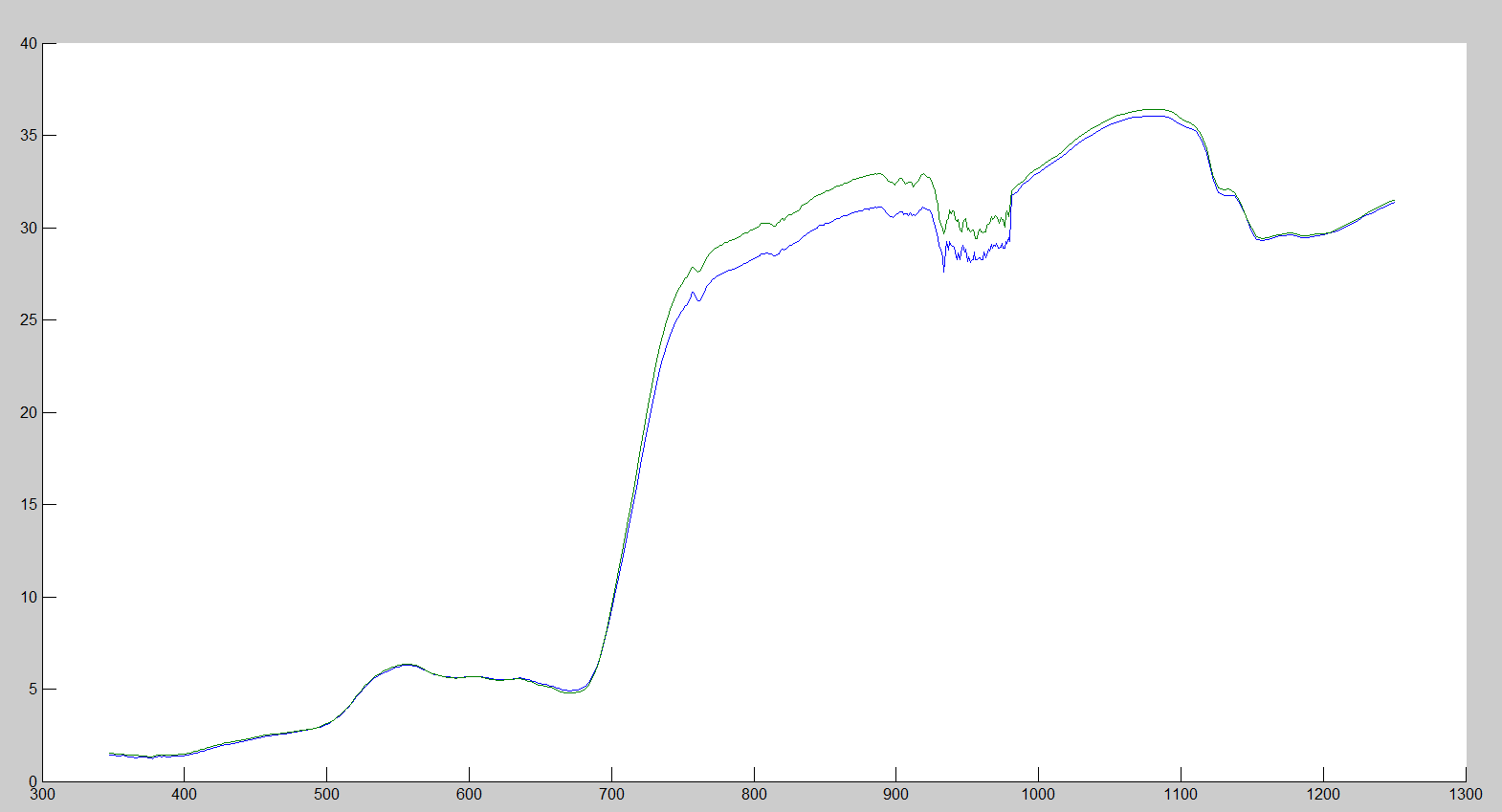
[2.2 选择纹理 4](#_Toc433634158)

# 背景

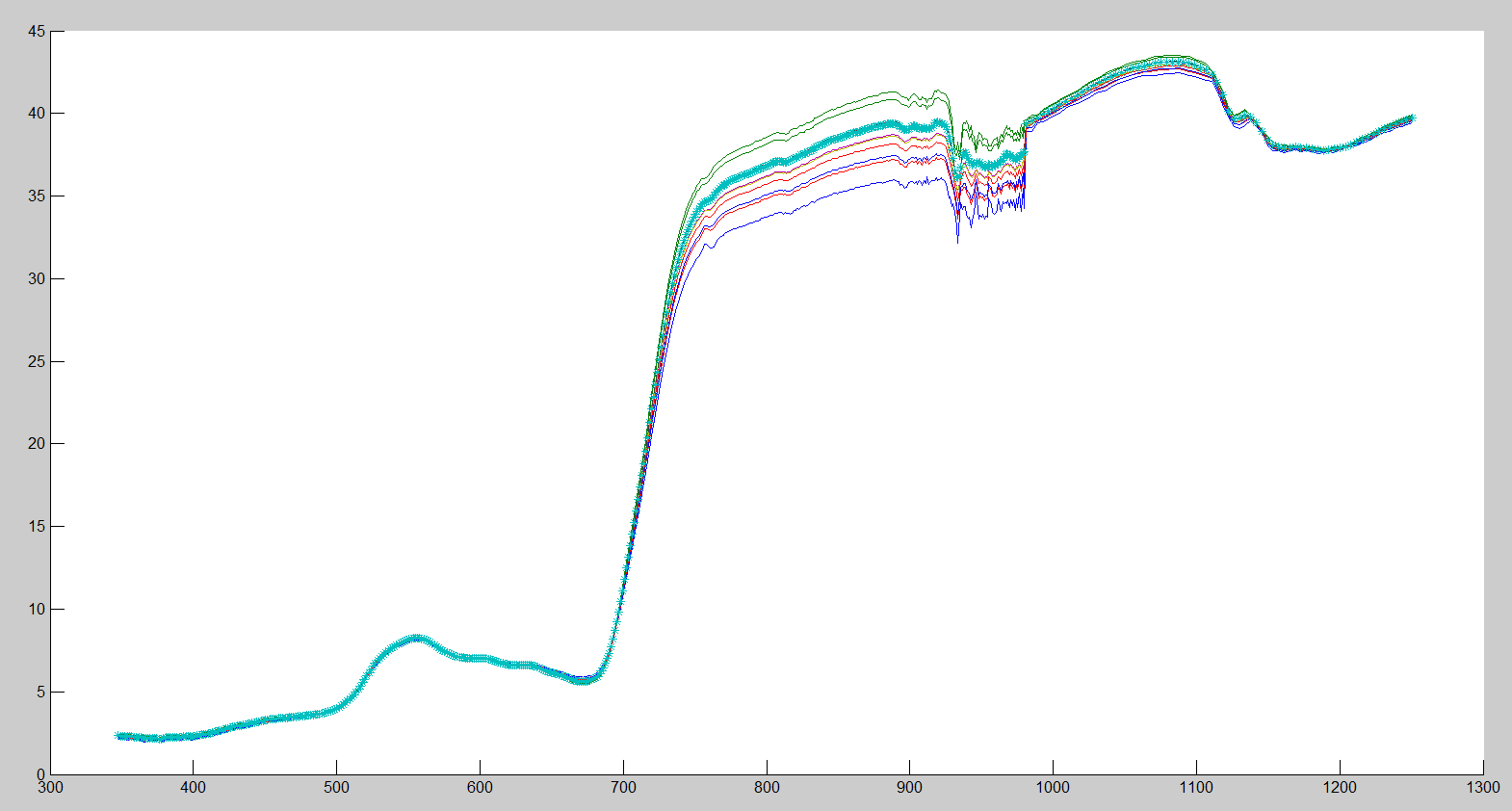
大草场 长密狗尾巴草 照片：DSC00975



光谱：0039~0040



随机生成10条光谱曲线：



# 2. 纹理理论和方法

## 2.1 扩展纹理集

### 2.2.1 均值和协方差矩阵

从一个实际地表类型所对应的光谱（假设它服从正态分布 ，进行次采样需要条原始曲线）对应的均值向量和协方差矩阵（注意：不同维度之间的协方差）分别为





这里， 和 是每一类的均值向量和协方差矩阵， 是光谱上第个点的均值， 是这个类中第个光谱均值和第个光谱均值之间的协方差。

### 2.2.2特征值和特征向量

计算这个协方差矩阵的特征值值和特征向量，分别为





这里满足：并且的每一列都是特征值相对应的归一化特征向量。

### 2.2.3 中心化

通过特征向量将原始的光谱数据转化为以零为中心的光谱非相关的数据



这个非相关的数据服从，且其协方差矩阵 满足



### 2.2.4 随机生成

用非相关的协方差矩阵生成多变量随机变量分布 。首先产生一组满足 的高斯随机数 组成的向量

对方程进行逆变换得到



为了服从多变量随机变量的分布，需要用到许多多变量统计数据来代表反射率曲线（注意：任何非零的值都要截断成零）。

为了获得更快的计算机运行速度，先优先用某种材料（这里使用草地）的特定像素进行一致抽样来产生一组曲线（比如100条曲线）。

## 2.2 选择纹理

对于某种特定材质类型的光谱数据库的每一条曲线，都可以得到每条曲线在这一组曲线中的评定，先计算曲线反射率的均值



这里，是第条曲线在波长为时的光谱反射率，是第条曲线在光谱条带上的采样数目，时曲线数据库中第条曲线从到之间反射率的均值。所有曲线的均值和标准差分别为





一旦确定光谱数据，这组光谱数据中每条曲线的-值的计算为



这个-值将会用来选择纹理图像中与特定亮度相匹配的曲线。

Matlab程序：

clear

load HR.090715.0000\_moc.txt

load HR.090715.0001\_moc.txt

load HR.090715.0002\_moc.txt

load HR.090715.0003\_moc.txt

load HR.090715.0004\_moc.txt

load HR.090715.0005\_moc.txt

load HR.090715.0006\_moc.txt

load HR.090715.0007\_moc.txt

load HR.090715.0008\_moc.txt

load HR.090715.0009\_moc.txt

load HR.090715.0010\_moc.txt

load HR.090715.0011\_moc.txt

load HR.090715.0012\_moc.txt

load HR.090715.0013\_moc.txt

load HR.090715.0014\_moc.txt

load HR.090715.0015\_moc.txt

load HR.090715.0016\_moc.txt

load HR.090715.0017\_moc.txt

load HR.090715.0018\_moc.txt

load HR.090715.0019\_moc.txt

load HR.090715.0020\_moc.txt

load HR.090715.0021\_moc.txt

load HR.090715.0022\_moc.txt

load HR.090715.0023\_moc.txt

load HR.090715.0024\_moc.txt

load HR.090715.0025\_moc.txt

load HR.090715.0026\_moc.txt

load HR.090715.0027\_moc.txt

load HR.090715.0028\_moc.txt

load HR.090715.0029\_moc.txt

load HR.090715.0030\_moc.txt

load HR.090715.0031\_moc.txt

load HR.090715.0032\_moc.txt

load HR.090715.0033\_moc.txt

load HR.090715.0034\_moc.txt

load HR.090715.0035\_moc.txt

load HR.090715.0036\_moc.txt

load HR.090715.0037\_moc.txt

load HR.090715.0038\_moc.txt

load HR.090715.0039\_moc.txt

load HR.090715.0040\_moc.txt

load HR.090715.0041\_moc.txt

load HR.090715.0042\_moc.txt

load HR.090715.0043\_moc.txt

load HR.090715.0044\_moc.txt

load HR.090715.0045\_moc.txt

load HR.090715.0046\_moc.txt

load HR.090715.0047\_moc.txt

load HR.090715.0048\_moc.txt

load HR.090715.0049\_moc.txt

load HR.090715.0050\_moc.txt

load HR.090715.0051\_moc.txt

load HR.090715.0052\_moc.txt

load HR.090715.0053\_moc.txt

load HR.090715.0054\_moc.txt

load HR.090715.0055\_moc.txt

load HR.090715.0056\_moc.txt

load HR.090715.0057\_moc.txt

load HR.090715.0058\_moc.txt

load HR.090715.0059\_moc.txt

load HR.090715.0060\_moc.txt

load HR.090715.0061\_moc.txt

load HR.090715.0062\_moc.txt

load HR.090715.0063\_moc.txt

load HR.090715.0064\_moc.txt

load HR.090715.0065\_moc.txt

x = HR\_090715\_0000\_moc(:, 1);

y0 = HR\_090715\_0000\_moc(:, 2);

y1 = HR\_090715\_0001\_moc(:, 2);

y2 = HR\_090715\_0002\_moc(:, 2);

y3 = HR\_090715\_0003\_moc(:, 2);

y4 = HR\_090715\_0004\_moc(:, 2);

y5 = HR\_090715\_0005\_moc(:, 2);

y6 = HR\_090715\_0006\_moc(:, 2);

y7 = HR\_090715\_0007\_moc(:, 2);

y8 = HR\_090715\_0008\_moc(:, 2);

y9 = HR\_090715\_0009\_moc(:, 2);

y10 = HR\_090715\_0010\_moc(:, 2);

y11 = HR\_090715\_0011\_moc(:, 2);

y12 = HR\_090715\_0012\_moc(:, 2);

y13 = HR\_090715\_0013\_moc(:, 2);

y14 = HR\_090715\_0014\_moc(:, 2);

y15 = HR\_090715\_0015\_moc(:, 2);

y16 = HR\_090715\_0016\_moc(:, 2);

y17 = HR\_090715\_0017\_moc(:, 2);

y18 = HR\_090715\_0018\_moc(:, 2);

y19 = HR\_090715\_0019\_moc(:, 2);

y20 = HR\_090715\_0020\_moc(:, 2);

y21 = HR\_090715\_0021\_moc(:, 2);

y22 = HR\_090715\_0022\_moc(:, 2);

y23 = HR\_090715\_0023\_moc(:, 2);

y24 = HR\_090715\_0024\_moc(:, 2);

y25 = HR\_090715\_0025\_moc(:, 2);

y26 = HR\_090715\_0026\_moc(:, 2);

y27 = HR\_090715\_0027\_moc(:, 2);

y28 = HR\_090715\_0028\_moc(:, 2);

y29 = HR\_090715\_0029\_moc(:, 2);

y30 = HR\_090715\_0030\_moc(:, 2);

y31 = HR\_090715\_0031\_moc(:, 2);

y32 = HR\_090715\_0032\_moc(:, 2);

y33 = HR\_090715\_0033\_moc(:, 2);

y34 = HR\_090715\_0034\_moc(:, 2);

y35 = HR\_090715\_0035\_moc(:, 2);

y36 = HR\_090715\_0036\_moc(:, 2);

y37 = HR\_090715\_0037\_moc(:, 2);

y38 = HR\_090715\_0038\_moc(:, 2);

y39 = HR\_090715\_0039\_moc(:, 2);

y40 = HR\_090715\_0040\_moc(:, 2);

y41 = HR\_090715\_0041\_moc(:, 2);

y42 = HR\_090715\_0042\_moc(:, 2);

y43 = HR\_090715\_0043\_moc(:, 2);

y44 = HR\_090715\_0044\_moc(:, 2);

y45 = HR\_090715\_0045\_moc(:, 2);

y46 = HR\_090715\_0046\_moc(:, 2);

y47 = HR\_090715\_0047\_moc(:, 2);

y48 = HR\_090715\_0048\_moc(:, 2);

y49 = HR\_090715\_0049\_moc(:, 2);

y50 = HR\_090715\_0050\_moc(:, 2);

y51 = HR\_090715\_0051\_moc(:, 2);

y52 = HR\_090715\_0052\_moc(:, 2);

y53 = HR\_090715\_0053\_moc(:, 2);

y54 = HR\_090715\_0054\_moc(:, 2);

y55 = HR\_090715\_0055\_moc(:, 2);

y56 = HR\_090715\_0056\_moc(:, 2);

y57 = HR\_090715\_0057\_moc(:, 2);

y58 = HR\_090715\_0058\_moc(:, 2);

y59 = HR\_090715\_0059\_moc(:, 2);

y60 = HR\_090715\_0060\_moc(:, 2);

y61 = HR\_090715\_0061\_moc(:, 2);

y62 = HR\_090715\_0062\_moc(:, 2);

y63 = HR\_090715\_0063\_moc(:, 2);

y64 = HR\_090715\_0064\_moc(:, 2);

y65 = HR\_090715\_0065\_moc(:, 2);

%%

sum0 = y0 + y1 + y2 + y3 + y4 + y5 + y6 + y7 + y8 + y9;

sum1 = y10 + y11 + y12 + y13 + y14 + y15 + y16 + y17 + y18 + y19;

sum2 = y20 + y21 + y22 + y23 + y24 + y25 + y26 + y27 + y28 + y29;

sum3 = y30 + y31 + y32 + y33 + y34 + y35 + y36 + y37 + y38 + y39;

sum4 = y40 + y41 + y42 + y43 + y44 + y45 + y46 + y47 + y48 + y49;

sum5 = y50 + y51 + y52 + y53 + y54 + y55 + y56 + y57 + y58 + y59;

sum6 = y60 + y61 + y62 + y63 + y64 + y65;

mean = (sum0 + sum1 + sum2 + sum3 + sum4 + sum5 + sum6) / 66;

A0 = [y0'; y1'; y2'; y3'; y4'; y5'; y6'; y7'; y8'; y9'];

A1 = [y10'; y11'; y12'; y13'; y14'; y15'; y16'; y17'; y18'; y19'];

A2 = [y20'; y21'; y22'; y23'; y24'; y25'; y26'; y27'; y28'; y29'];

A3 = [y30'; y31'; y32'; y33'; y34'; y35'; y36'; y37'; y38'; y39'];

% A3 = [y30'; y31'; y32'; y33'; y34'; y36'; y37'; y38'; y39'];

A4 = [y40'; y41'; y42'; y43'; y44'; y45'; y46'; y47'; y48'; y49'];

A5 = [y50'; y51'; y52'; y53'; y54'; y55'; y56'; y57'; y58'; y59'];

A6 = [y60'; y61'; y62'; y63'; y64'; y65'];

% A6 = [y63'; y64'; y65'];

% A = [A0; A1; A2; A3; A4; A5; A6];

A = [y39'; y40'];

%%

% A = [A; A; A; A; A; A; A; A; A; A; A; A; A; A];

covarianceMatrix = cov(A);

% [D, p] = chol(covarianceMatrix);

d = eig(covarianceMatrix);

[eigenVector, eigenValue] = eig(covarianceMatrix);

% sort eigenValue(with eigenVector)

n = length(d);

i = 1; j = 1;

min = 1e-10;

for i = 1 : n

if d(i) < min

d(i) = 0;

end

end

% for j = 1 : n

% for i = 2 : n

% if d(i-1) < d(i)

% temp = d(i);

% d(i) = d(i-1);

% d(i-1) = temp;

% for k = 1 : n % exchange eigenVector according eigenValue

% t = eigenVector(k, i);

% eigenVector(k, i) = eigenVector(k, i-1);

% eigenVector(k, i-1) = t;

% end

% end

% end

% n = n-1;

% end

% normalize eigenVector

[rowNum colNum] = size(A);

for i = 1 : colNum

eigenVector(:, i) = eigenVector(:, i) / norm(A(:, i));

end

% d = sort(d, 'descend');

% for i = 1 : 548

% if d(i) < 0

% d(i) = 0;

% end

% end

% d = (d - min(d))/(max(d)-min(d));

% for i = 0 : 100

% yi = normrnd(0, d');

% resulti = inv(eigenVector) \* yi' + mean;

% plot(x, resulti, '-');

% end

% Y1= normrnd(0,d');

% result1 = inv(eigenVector) \* Y1' + mean;

hold all;

rouSum = 0;

N = 100; % the sum of line

for i = 1 : N

yi = normrnd(0, d');

yi = yi / 300;

resulti = inv(eigenVector') \* yi' + mean;

plot(x, resulti, '-');

% compute the z-score

rouAvgI(1, i) = sum(resulti) / 61;

rouSum = rouSum + rouAvgI(1, i);

end

rouAvg = rouSum / N;

deltaRou = std(rouAvgI);

zi = (rouAvgI - rouAvg) / deltaRou;

%%

plot(x, mean, '\*');