4.3 Coding Of The Model Parameters

In the previous section, a method is introduced to extract the model parameters (I_L,C_L,C_R,W_L,W_R) by capturing the edge profiles $P_n(c_i)$ and $P_c(c_i)$. The parameters are modeled by polynomials and polynomial coefficients are coded. There are many advantages of coding with polynomials :

- Since parameters extracted from neighboring edge points are very similar and exhibit high correlation, it is therefore reasonable to use polynomials for exploiting the redundancy. Also, coding of all model parameters is a bit-consuming operation.
- Fitting polynomials to the parameters by minimizing least mean square error results in smoothing of the parameters. This may also smooth the error due to the parameter extraction process.
- It gives the ability of changing the compression ratio while allowing small amount of degradation. Lower the order of the polynomials, higher the compression ratio is.
- It is computationally easy to fit a polynomial to a curve.

In this section, curve fitting with polynomials is explained, then we proceed with the image reconstruction from model parameters by hybrid energy functional.

4.3.1 Curve Fitting with Polynomials

In curve fitting, we are given n points with pairs (x_1,y_1), ... , (x_n,y_n) and we want to determine a function f(x) such that $f(x_j) \cong y_j$, j=1,...,n. The type of function may be suggested by the nature of the problem. We offer polynomials of order n due to the reasons explained above. A widely used procedure of curve fitting is the method of *least squares*.

Let $P_n(x)$ be the nth order polynomial with the coefficients $(c_0,c_1,...,c_n)$ as given in (4.7)

$$P_n(x) = c_0 + c_1 x^1 + c_2 x^2 + \ldots + c_n x^n \tag{4.7}$$

M data pair is given in the form $(y_0, y_1, \ldots, y_M)$. We want such a polynomial with the coefficients $(c_0, c_1, \ldots, c_n)$ that it minimizes the quantity MSE denoted by Q :

$$Q = \sum_{i=1}^{M} [P(x_\mathrm{i}) - y_\mathrm{i}]^2 \tag{4.8}$$

A necessary condition for MSE to be minimum is

$$\frac{\partial Q}{\partial a_\mathrm{i}} = 0 \text{, i=0,...,n} \tag{4.9}$$

where $P(x_\mathrm{i}) = \sum_{k=0}^{n} c_\mathrm{k} x_\mathrm{i}{}^\mathrm{k}$. We define x_i as $x_\mathrm{i} = i$ for simplicity. In order to have a matrix form, we will define the following matrices

$$S_\mathrm{k} = \sum_{i=1}^{M} x_\mathrm{i}^\mathrm{k} = \sum_{i=1}^{M} i^\mathrm{k} \tag{4.10}$$

$$M_\mathrm{k} = \sum_{i=1}^{M} x_\mathrm{i}^\mathrm{k} y_\mathrm{i} = \sum_{i=1}^{M} i^\mathrm{k} y_\mathrm{i} \tag{4.11}$$

Solution of equation (4.9) becomes the solution of the following linear matrix equation

$$[S][A]=[M] \tag{4.12}$$

where

$$[S]=\begin{bmatrix} S_0 & S_1 & \dots & S_{n-1} \\ S_1 & S_2 & \dots & S_n \\ \vdots & \vdots & \dots & \vdots \\ S_{n-1} & S_n & \dots & S_{2(n-1)} \end{bmatrix},$$

$$A=\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_n \end{bmatrix} \quad \text{and} \quad M=\begin{bmatrix} M_0 \\ M_1 \\ \vdots \\ M_{n-1} \end{bmatrix}.$$

Note that this system is symmetric. To solve (4.12) for unknowns $(c_0, c_1, ..., c_n)$, first the matrix S is transformed into a tridiagonal form by Householder's method [36]. An algorithm for the curve fitting by polynomials is given in Table 4.1.

4.3.2 Image Reconstruction By Using The Hybrid Model

The following processes are applied to the image in given order at the transmitter

I. Edge map is obtained by generalized edge detector,

II. The edge map with the image is used to extract the model parameters,

III. The extracted parameters are modeled by polynomials,

IV. Polynomial coefficients are quantized and sent with the edge map coded by differential chain code followed by Huffman coding.

Receiver constructs two images which are

1. Edge map which is a binary image,

2. Intensity image is obtained from model parameters (I_L, C_L, C_R, W_L, W_R) which are constructed easily by evaluating the polynomial value at each edge point.

Since both images are sparse, we use hybrid energy functional to span a surface through these points. Hybrid energy functional tries to find such a function f(x,y) which minimizes

$$E(f) = \iint_{\Omega} (f(x,y) - d(x,y))^2 + \lambda\,[(1-\tau)(f_x^2 + f_y^2) + \tau(f_{xx}^2 + 2f_{xy}^2 + f_{yy}^2)]\,dxdy \tag{4.13}$$

where λ controls the smoothness of the surface and τ controls the continuity of the surface. In the functional, first term on the right hand side is a measure of the closeness of the solution *f(x,y)* to the data *d(x,y)*, and the second and the third terms are stabilizers on the solution as the first and second order derivatives. The Euler-Lagrange equation associated with this hybrid functional is

$$\lambda\tau\,[f_{xxxx} + 2f_{xxyy} + f_{yyyy}] - \lambda(1-\tau)\,[f_{xx} + f_{yy}] + f = d \tag{4.14}$$

Properties of the hybrid model and λτ–space can be found in [2]. The minimization problem can be solved by discretizing the partial differential equation (4.14) or directly discretizing the hybrid energy functional by using finite difference approximation of derivatives.

$$\begin{aligned} E(u;\lambda,\tau) = {} & \sum_{i}\sum_{j} \beta_{i,j}(u_{i,j} - d_{i,j})^2 + \\ & \sum_{i}\sum_{j} \lambda\tau((u_{i,j} - u_{i-1,j})^2 + (u_{i,j} - u_{i,j-1})^2) + \\ & \sum_{i}\sum_{j} \lambda(1-\tau)\,((u_{i+1,j} + u_{i-1,j} - 2u_{i,j})^2 + \\ & (u_{i,j+1} + u_{i,j-1} - 2u_{i,j})^2 + 2(u_{i+1,j+1} + u_{i,j} - u_{i+1,j} - u_{i,j+1})^2) \end{aligned}$$

An iterative solution to (4.15) is obtained by the Successive-Over Relaxation iterations as

$$u_{\mathrm{i,j}}^{(n+1)} = u_{\mathrm{i,j}}^{(n)} - \frac{w}{T}\frac{\partial E(u)}{\partial u_{\mathrm{i,j}}} \tag{4.16}$$

$$\begin{aligned}\frac{\partial E(u)}{\partial u_{\mathrm{i,j}}} = & -\beta_{\mathrm{i,j}} \times d_{\mathrm{i,j}} + [\beta_{\mathrm{i,j}} + 4\lambda(1+4\tau)] \times u_{\mathrm{i,j}}^{(n)} \\ & -\lambda(1+7\tau)[u_{\mathrm{i\text{-}1,j}}^{(n+1)} + u_{\mathrm{i,j\text{-}1}}^{(n+1)} + u_{\mathrm{i+1,j}}^{(n)} + u_{\mathrm{i,j+1}}^{(n)}] \\ & + 2\lambda\tau[u_{\mathrm{i\text{-}1,j\text{-}1}}^{(n+1)} + u_{\mathrm{i\text{-}1,j+1}}^{(n+1)} + u_{\mathrm{i+1,j\text{-}1}}^{(n)} + u_{\mathrm{i+1,j+1}}^{(n)}] \\ & + \lambda\tau[u_{\mathrm{i\text{-}2,j}}^{(n+1)} + u_{\mathrm{i,j\text{-}2}}^{(n+1)} + u_{\mathrm{i+2,j}}^{(n)} + u_{\mathrm{i,j+2}}^{(n)}]\end{aligned} \tag{4.17}$$

where $\beta_{\mathrm{i,j}}$ handles the sparseness of data and it is equal to 1 if data is available at the point (i,j), and 0 otherwise. In Figure 4.3,4.4,4.5 (c) and (d) represents $\beta_{\mathrm{i,j}}$ and $d_{\mathrm{i,j}}$.The iterations terminate when the condition $\left\| u_{\mathrm{i,j}}^{(n+1)} - u_{\mathrm{i,j}}^{(n)} \right\|_{\infty} < \varepsilon$ is satisfied for a prespecified ε. The iteration given by (4.16) is an interpolation by iterative over relaxation governed by the heat diffusion equation. Such iterative solutions require long convergence time. Theoretically, the number of iterations for convergence is N^2 . One way to accelerate the convergence is to use multigrid technique [41]. Recently, Salembier et al. [30] has presented a new interpolant based on morphological operations such as dilation and erosions which do not require any multiplication. They have reported 13 iterations equivalent to 376 iterations of multigrid diffusion and 2980 iterations of linear diffusion.

We examine how the quality of reconstructed image by using the hybrid model is affected by the order of the polynomial and the block length in terms of nmse. It is

Table 4.1 Curve fitting by polynomials

- Construction of S and M matrices

```
for k:=1 to M
 for i:=0 to n
    for j:=0 to n
         begin
           S[i][j] = S[i][j] + power(k,i+j)      ;
          M[i]    = power(k,i) * y[i]        ;
       end
```

- Solution of linear matrix equation [S] [A] = [M]

Symmetric linear system [S] [A] = [M] is transformed into tridiagonal form by Householder's method. Then unknows can easily found as follows

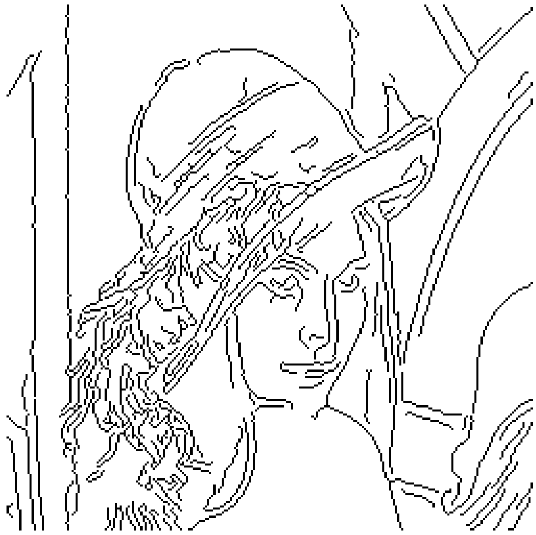
```
for i:= 0 to n-2
for j:= i+1 to n-1
    begin
       pivot   := - S[i][j]/ S[i][i]           ;
     S[j][i]  :=  0                   ;
     for k:= j+1 to n-1
         S[j][k] = S[j][k] + pivot × S[i][k]   ;
    M[j]    = pivot × M[i]              ;
   end

A[n-1] := M[n-1] / S[n-1][n-1]            ;
for k:= n-2 to 0
   begin
     A[k] := M[k]                     ;
  for i:= n-1 to k+1
     A[k] := A[k] - S[i] × A[i]           ;
end
```

evident that the nmse increases as the order decreases and block length increases. Besides that, we have also observed that there exist an order and a block length such that nmse of the reconstructed image starts to remain almost constant. Quality of the reconstructed image is affected by the order of the polynomial for intensity and contrast much more than the one for width.

Reconstructed images by the centipede model with varying polynomial order are given in Figure 4.6. For Lenna image and its edge map given in Figure 4.6 (a) and (b) when the block length is equal to 10. Since nmse remains the same for higher order approximations at about 37, the polynomial order is $(Order_{contrast}, Order_{intensity}, Order_{width})=(3,4,1)$ enough to represent the model parameters.

Quality of the reconstructed images by the centipede model with varying block length is given in Table 4.2 for Lenna image. It has been observed that nmse for block length of 12 is still close to the nmse’s for shorter block lengths.

(a)



(b)

(c)

(d)

(e)

(f)

Figure 4.6 (a) Original Lenna Image
(b) Edges detected by GED for (λ=0.5,τ=0.5)
(c) Reconstructed Lenna Image for (1,1,1)
nmse=41.34, snr=17.66, psnr=49.35
(d) Reconstructed Lenna Image for (2,2,1)
nmse=39.15, snr=18.75, psnr=50.44
(e) Reconstructed Lenna Image for (3,4,1)
nmse=38.83, snr=19.17, psnr=50.86
(f) Reconstructed Lenna Image for (4,5,2)
nmse=37.09, snr=19.40, psnr=51.09
where ($Order_{contrast}$, $Order_{intensity}$,$Order_{width}$) stands for polynomial orders

Table 4.2 Quality of Image Reconstructed by Centipede Model with respect to Block Length

Block Length	NMSE	SNR (dB)	PSNR
1	29.53	24.39	56.08
2	33.67	21.70	53.45
4	36.43	20.19	51.88
6	37.96	19.36	50.05
8	39.03	18.81	50.50
10	39.36	18.64	50.33
12	40.41	18.12	49.80
Contour Length	48.04	14.66	46.35

4.4 Experimental Results

We have used two methods to increase the quality of the reconstructed image while decreasing the compression ratio little:

- Adaptively varying the order of polynomial in a block with a constant length,
- Mean coding.

Different contours need different order of approximation for the same error measure. By adaptively changing the order of the polynomial, it is possible that the reconstructed image will have higher quality for the same compression ratio. The results from the previous section also suggest this conclusion.

Since the surface reconstruction does not guarantee that the mean of the image will be preserved. In mean coding, the mean of the image in a block of size W is coded and sent to the receiver. Receiver side uses the mean information in the reconstruction level in such a way that the mean of the reconstructed image is closer to the received mean in that block. Sending the mean of the image in a block does not only increase the quality of the reconstructed image but also speed the reconstruction process up.

Compression results are given in Figure 4.7 with the improvements stated above. We have applied the centipede model on various types of images from artificial images to real images having completely different features. The results are summarized in Table 4.3.

Table 4.3 Compression Results for various type of images
(CR stands for Compression Ratio)

Image	**Size** (bytes)	**CR Differential Chain Code**	**CR Polynomial Coefficients**	**CR**	**NMSE**	**SNR (dB)**
che.hips	16465	210:1	230:1	**157:1**	14.56	63.26
bars.hips	16465	160:1	606:1	**127:1**	10.28	65.73
mouse.hips	250149	186:1	251:1	**107:1**	16.25	36.33
house.hips	65106	86:1	107:1	**48:1**	38.56	19.05
camera.hips	262225	70:1	140:1	**43:1**	36.29	20.26
lena.hips	86561	48:1	123:1	**35:1**	35.95	20.45
brain.hips	30706	23:1	30:1	**13:1**	46.27	15.41