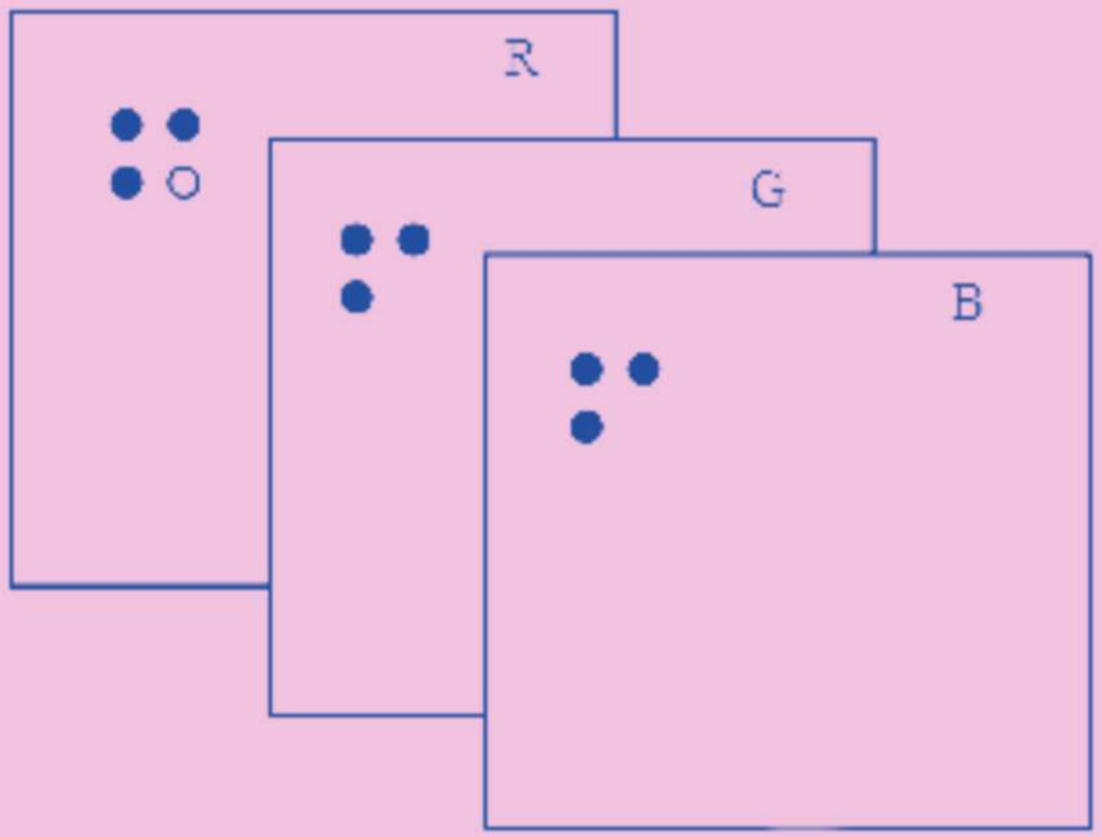
COLOR IMAGE PREDICTION CODING

IMA 4508

WHAT IS COLOR IMAGE PREDICTION CODING ?



- RGB color model
- Data compression
- Prediction error

$$(\mathrm{i,j}) = \sum_k \theta_k \cdot N_k(i,j)$$

- (i,j) a pixel
- θk prediction coefficient
- Nk neighbors

Why is is relevant ?

- Exploit spatial redundancy
- Lower Storage Costs

OUR PREDICTIVE MODEL

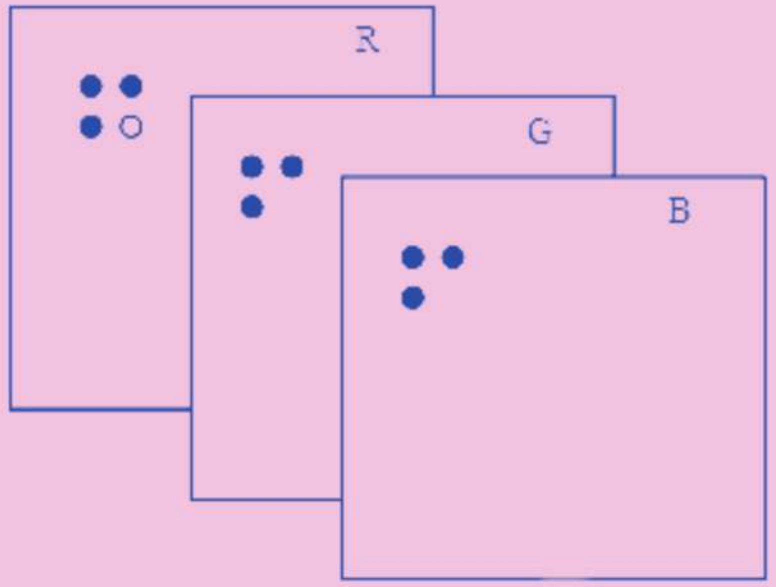
$$\hat{R}(i, j) = r_1 R(i-1, j) + r_2 R(i, j-1) + r_4 G(i-1, j) + r_5 G(i, j-1) + r_7 B(i-1, j) + r_8 B(i, j-1)$$

$$\hat{G}(i, j) = g_1 R(i-1, j) + g_2 R(i, j-1) + g_4 G(i-1, j) + g_5 G(i, j-1) + g_7 B(i-1, j) + g_8 B(i, j-1) + g_{10} R(i, j)$$

$$\hat{B}(i, j) = b_1 R(i-1, j) + b_2 R(i, j-1) + b_4 G(i-1, j) + b_5 G(i, j-1) + b_7 B(i-1, j) + b_8 B(i, j-1) + b_{10} R(i, j) + b_{11} G(i, j)$$

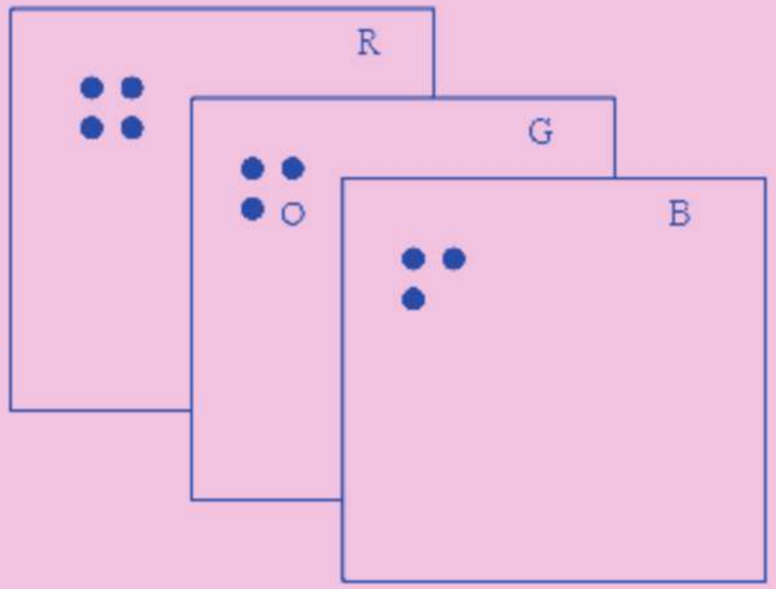
1

Predict the red pixel



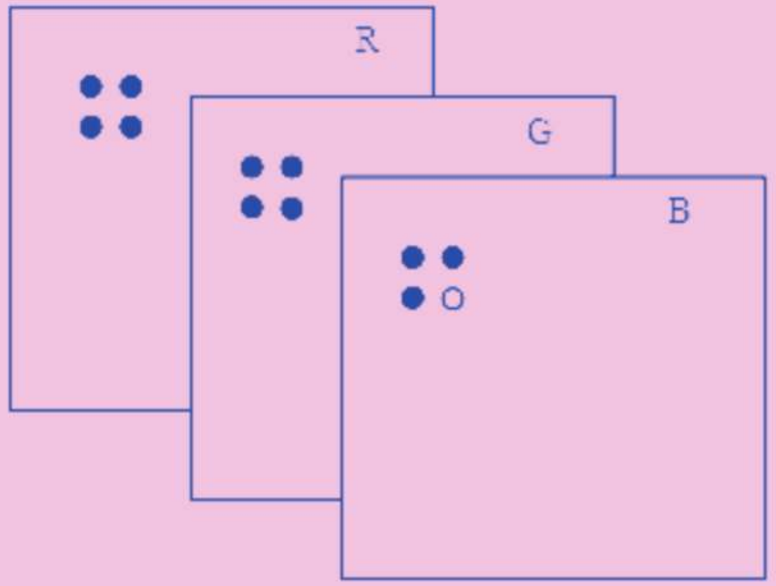
2

Predict the green pixel



3

Predict the blue pixel



THEORY

$\theta_g \quad \theta_r \quad \theta_b$

$$\downarrow$$

$$\boldsymbol{\theta} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{bmatrix}$$

Matrix form :

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\theta} + \boldsymbol{\varepsilon}$$

Y: true output value
X: Vector of neighboring pixels values
θ: coefficients to be estimated
ε : error

Optimal Solution :

given by the least squares method

$$\boldsymbol{\theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

$$\Downarrow$$

Problem :

$$\theta \;= \arg\min_{\theta} \|Y - X\theta\|_2^2$$

What we use in practise :

$$\boldsymbol{\theta} = \mathbf{K}^{-1}\mathbf{Y}_r$$

K : Correlation matrix between predictors
Yr : correlation vector

AR PREDICTIVE COEFFICIENTS

Image processing

For the three pixels,
extract the three RGB channels

Calculate covariance terms for R,G,B

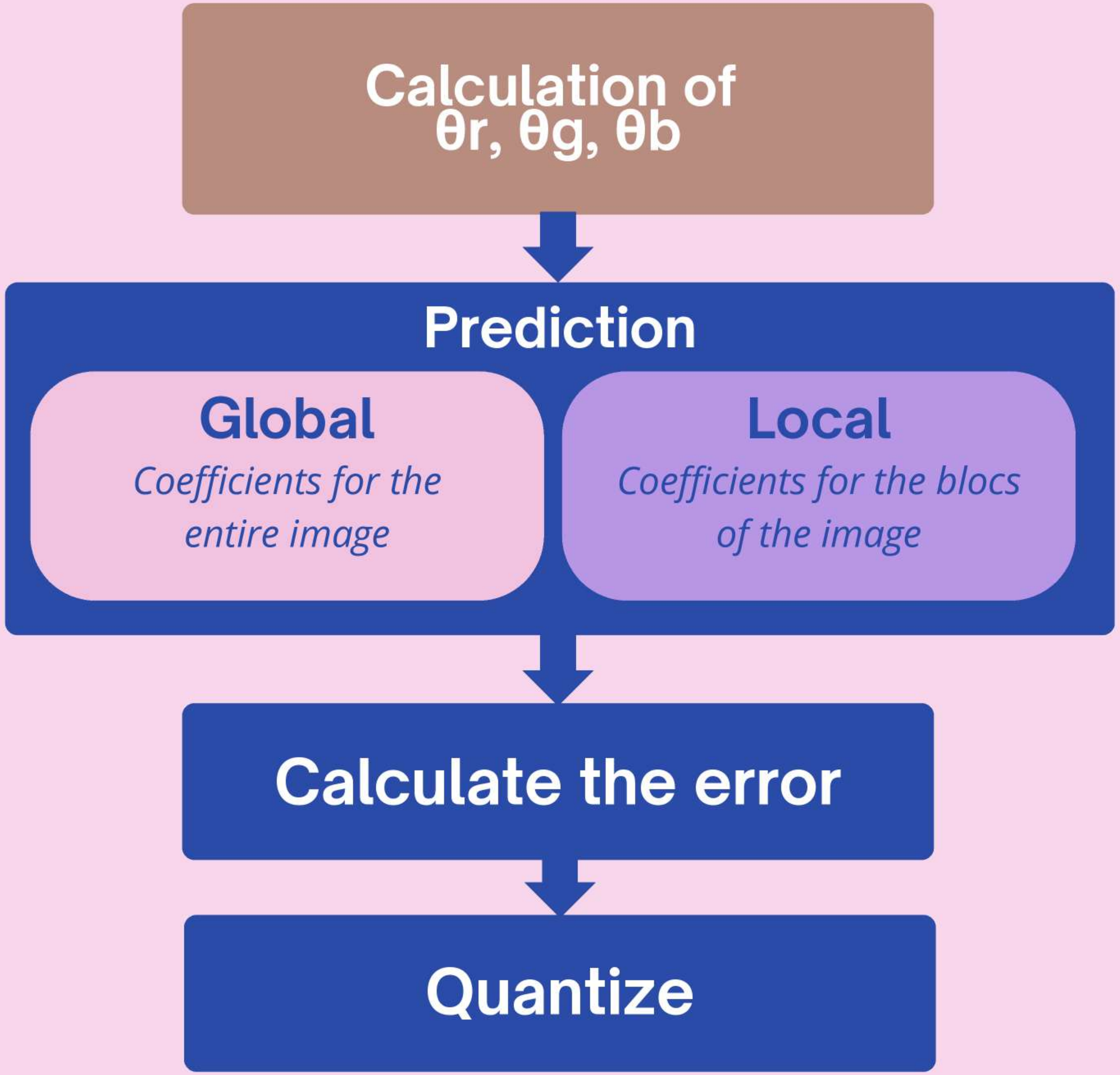
$$K = \begin{bmatrix} RR_{00} & RR_{11} & RG_{00} & RG_{11} & RB_{00} & RB_{11} \\ RR_{11} & RR_{00} & GR_{11} & RG_{00} & BR_{11} & RB_{00} \\ RG_{00} & GR_{11} & GG_{00} & GG_{11} & GB_{00} & GB_{11} \\ RG_{11} & RG_{00} & GG_{11} & GG_{00} & BG_{11} & GB_{00} \\ RB_{00} & BR_{11} & GB_{00} & BG_{11} & BB_{00} & BB_{11} \\ RB_{11} & RB_{00} & GB_{11} & GB_{00} & BB_{11} & BB_{00} \end{bmatrix} \qquad Y_r = \begin{bmatrix} RR_{10} \\ RR_{01} \\ RG_{10} \\ RG_{01} \\ RB_{10} \\ RB_{01} \end{bmatrix}$$

Build coefficient matrices K and Yr

Solving to find θr, θg, θb

$$\boldsymbol{\theta} = \mathbf{K}^{-1}\mathbf{Y}_r$$

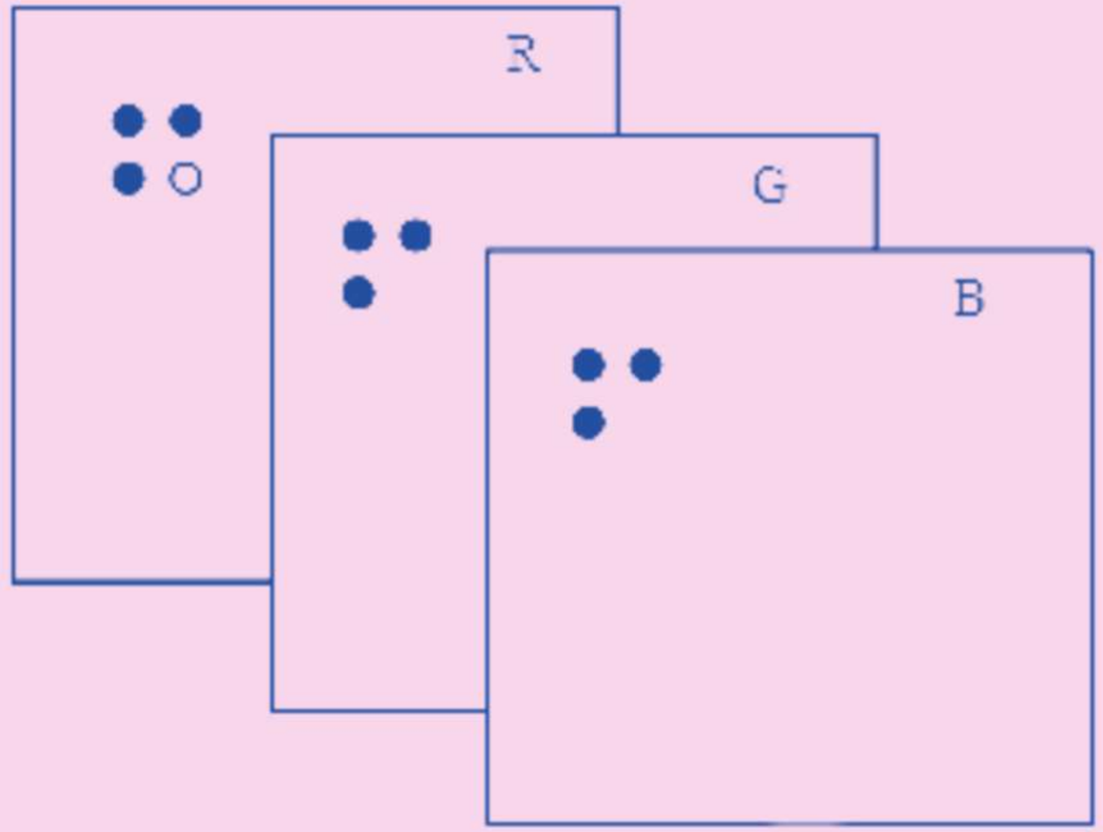
GLOBAL VS LOCAL STRATEGY



HOW WE CODE - GLOBAL

Image processing

For the three pixels, extract the three RGB channels



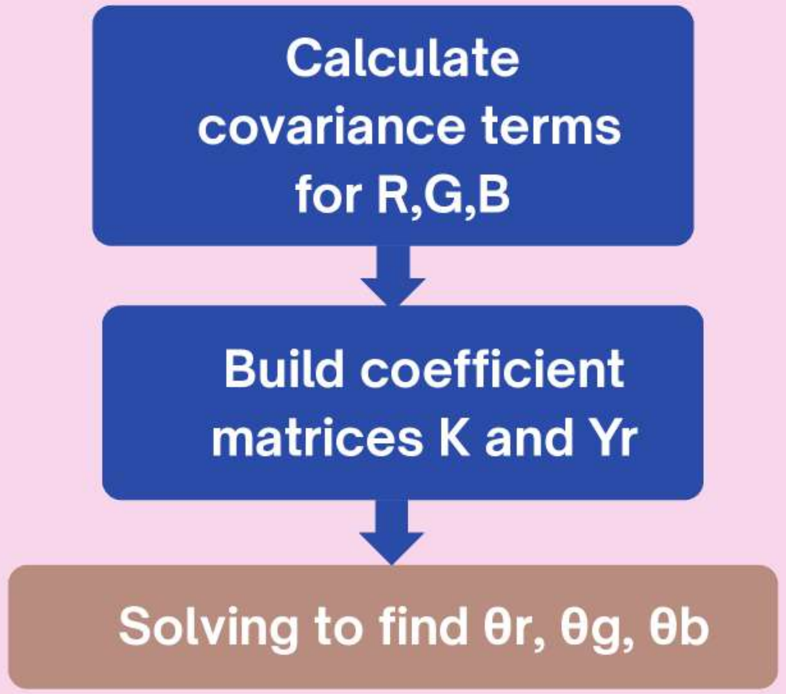

```
% calculate Coefficient
R = img(:,:,1);
G = img(:,:,2);
B = img(:,:,3);
[M,N] = size(R);
Rp = padarray(R, [1,1], 'symmetri');
Gp = padarray(G, [1,1], 'symmetri');
Bp = padarray(B, [1,1], 'symmetri');
R_left = Rp(2:M+1, 1:N);
R_top = Rp(1:M,2:N+1);
G_left = Gp(2:M+1, 1:N);
G_top = Gp(1:M,2:N+1);
B_left = Bp(2:M+1, 1:N);
B_top = Bp(1:M,2:N+1);
```

```
function [r,g,b] = Cal_para(filename)

function [err_r, err_g, err_b,Rmed, Gmed, Bmed] = predictionRGB(filename, r, g, b, delta)

function RGB_rec = predictionRGB_inv(err_r, err_g, err_b, r, g, b, delta, Rmed, Gmed, Bmed)
```

Calculate covariance terms




```
function [r,g,b] = Cal_para(filename)

% Correlation
RR01 = mean(mean(R.*R_top));

% Correlation matrix
Kr = [RR00,RR11,RG00,RG11,RB00,RB11;
     RR11,RR00,GR11,RG00,BR11,RB00;
     RG00,GR11,GG00,GG11,GB00,GB11;
     RG11,RG00,GG11,GG00,BG11,GB00;
     RB00,BR11,GB00,BG11,BB00,BB11;
     RB11,RB00,GB11,GB00,BB11,BB00];
```

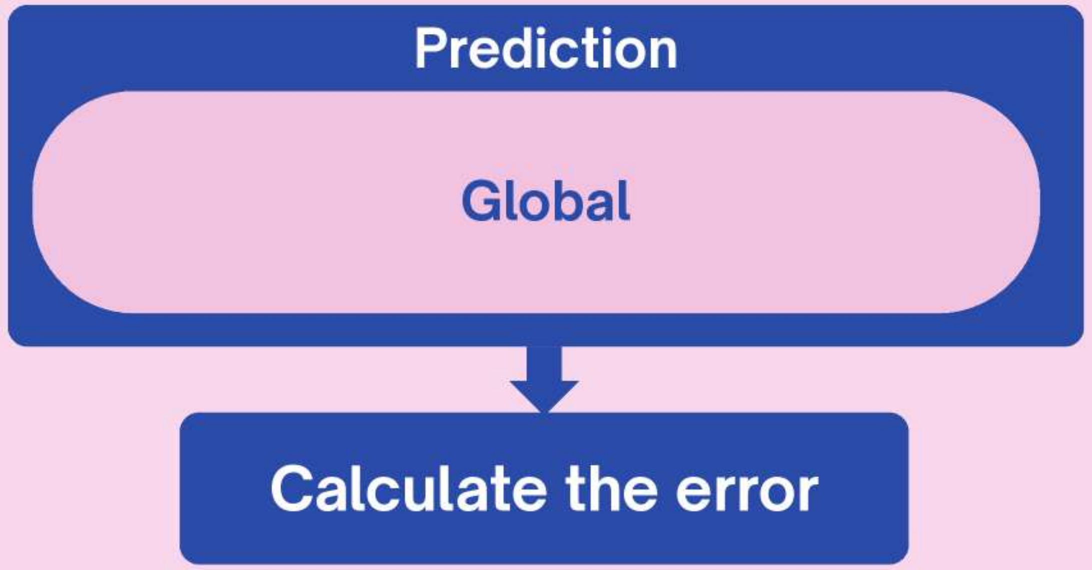
$$\begin{aligned}\hat{R}(i,j) = {} & r_1 R(i-1,j) + r_2 R(i,j-1) \\ & + r_4 G(i-1,j) + r_5 G(i,j-1) \\ & + r_7 B(i-1,j) + r_8 B(i,j-1)\end{aligned}$$

$$\begin{aligned}\hat{G}(i,j) = {} & g_1 R(i-1,j) + g_2 R(i,j-1) \\ & + g_4 G(i-1,j) + g_5 G(i,j-1) \\ & + g_7 B(i-1,j) + g_8 B(i,j-1) \\ & + g_{10} R(i,j)\end{aligned}$$

$$\begin{aligned}\hat{B}(i,j) = {} & b_1 R(i-1,j) + b_2 R(i,j-1) \\ & + b_4 G(i-1,j) + b_5 G(i,j-1) \\ & + b_7 B(i-1,j) + b_8 B(i,j-1) \\ & + b_{10} R(i,j) + b_{11} G(i,j)\end{aligned}$$

9 x 8 / 2+3-6=33

```
for i = 1:M
    for j = 1:N
        % boundary
        Rl = Rrec(max(i-1,1), j);
        Rt = Rrec(i, max(j-1,1));
        Gl = Grec(max(i-1,1), j);
        Gt = Grec(i, max(j-1,1));
        Bl = Brec(max(i-1,1), j);
        Bt = Brec(i, max(j-1,1));
        % predict
        R_pred = r(1)*Rl + r(2)*Rt + r(3)*Gl + r(4)*Gt + r(5)*Bl + r(6)*Bt;
        G_pred = g(1)*Rl + g(2)*Rt + g(3)*Gl + g(4)*Gt + g(5)*Bl + g(6)*Bt + g(7)*R_pred;
        B_pred = b(1)*Rl + b(2)*Rt + b(3)*Gl + b(4)*Gt + b(5)*Bl + b(6)*Bt + b(7)*R_pred + b(8)*G_pred;
        r_val = Rz(i,j);    g_val = Gz(i,j);    b_val = Bz(i,j);
        % calculate error and quantize
        err_r(i,j) = round((r_val - R_pred) / delta);
        err_g(i,j) = round((g_val - G_pred) / delta);
        err_b(i,j) = round((b_val - B_pred) / delta);
        % update the rec
        Rrec(i,j) = R_pred + delta * err_r(i,j);
        Grec(i,j) = G_pred + delta * err_g(i,j);
        Brec(i,j) = B_pred + delta * err_b(i,j);
    end
end
```



```
for i = 1:M
    for j = 1:N
        % neighborhood
        Rl = Rrec(max(i-1,1), j);
        Rt = Rrec(i, max(j-1,1));
        Gl = Grec(max(i-1,1), j);
        Gt = Grec(i, max(j-1,1));
        Bl = Brec(max(i-1,1), j);
        Bt = Brec(i, max(j-1,1));
        % predict
        R_pred = r(1)*Rl + r(2)*Rt + r(3)*Gl + r(4)*Gt + r(5)*Bl + r(6)*Bt;
        G_pred = g(1)*Rl + g(2)*Rt + g(3)*Gl + g(4)*Gt + g(5)*Bl + g(6)*Bt + g(7)*R_pred;
        B_pred = b(1)*Rl + b(2)*Rt + b(3)*Gl + b(4)*Gt + b(5)*Bl + b(6)*Bt + b(7)*R_pred + b(8)*G_pred;
        % dequantize + rebuild
        Rrec(i,j) = R_pred + delta * err_r(i,j);
        Grec(i,j) = G_pred + delta * err_g(i,j);
        Brec(i,j) = B_pred + delta * err_b(i,j);
    end
end
```

```
%decentralization
R = Rrec + Rmed;
G = Grec + Gmed;
B = Brec + Bmed;
```

From the error

reconstruction

Global

Using centralization

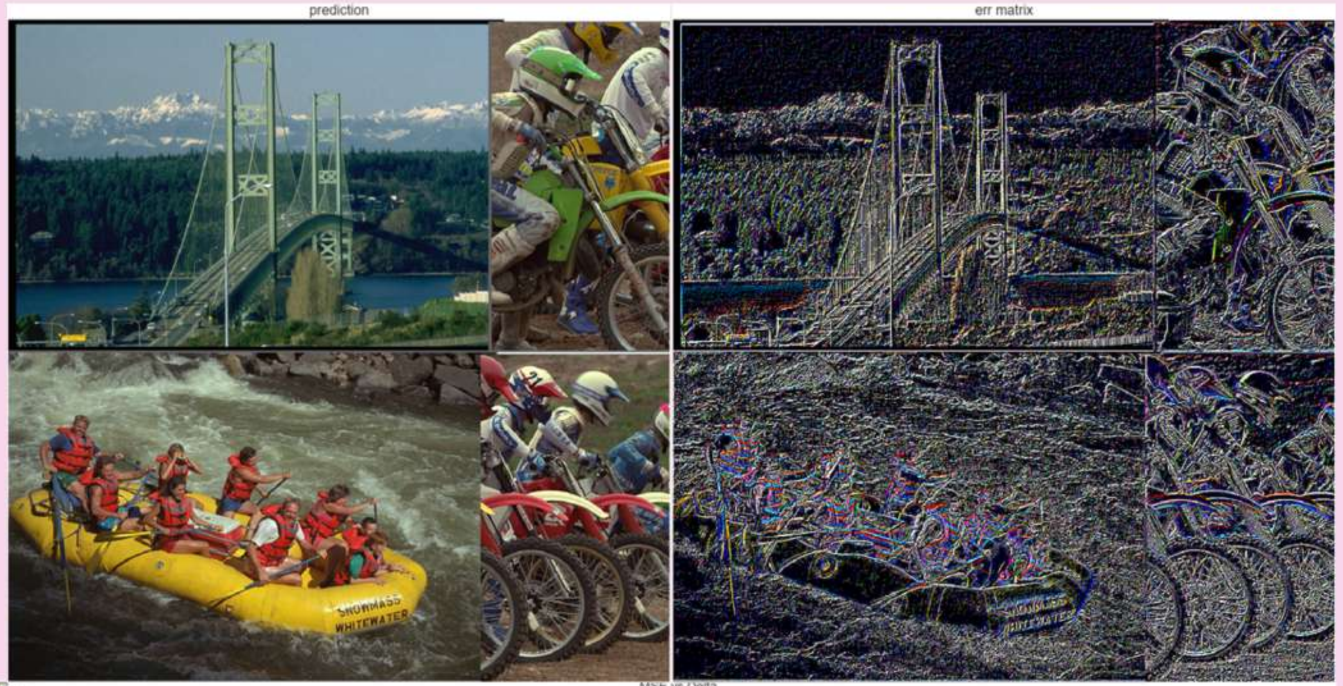
MSE: R=35.79, G=33.24, B=50.94
PSNR: R=32.59 dB, G=32.91 dB, B=31.06 dB
Entropy,R=0.19, G=0.16, B=0.95

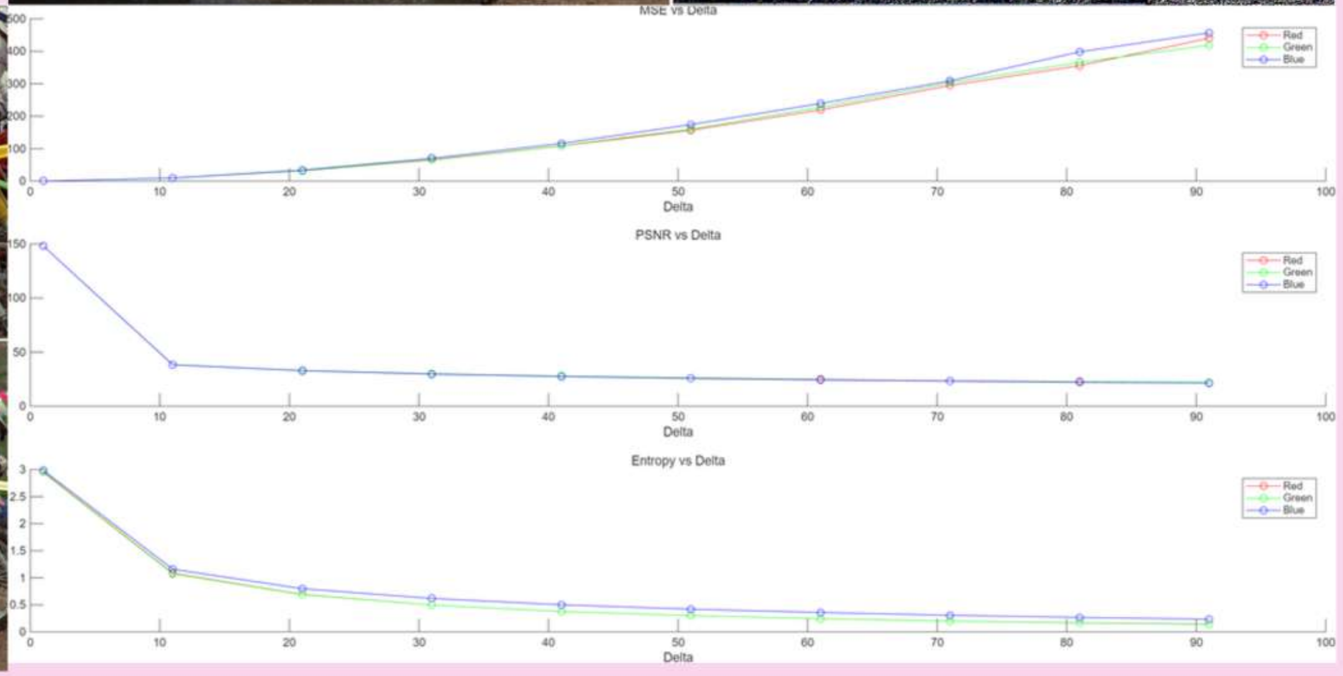
Without centralization

MSE: R=36.43, G=34.49, B=50.96
PSNR: R=32.52 dB, G=32.75 dB, B=31.06 dB
Entropy,R=0.19, G=0.16, B=0.95

% set delta = 25

PARAME TERSAND RESULTS





% set delta = 2

r =	g =	b =
		-0.8535
0.5350	-0.5527	-0.0229
0.4614	-0.4153	0.6998
-0.0086	0.5701	-0.7107
0.0135	0.4303	0.1926
0.0002	0.0008	0.7900
-0.0045	-0.0014	0.8586
	0.9680	0.0434

%Original

PSNR: R=148.13 dB, G=148.13 dB, B=148.13 dB

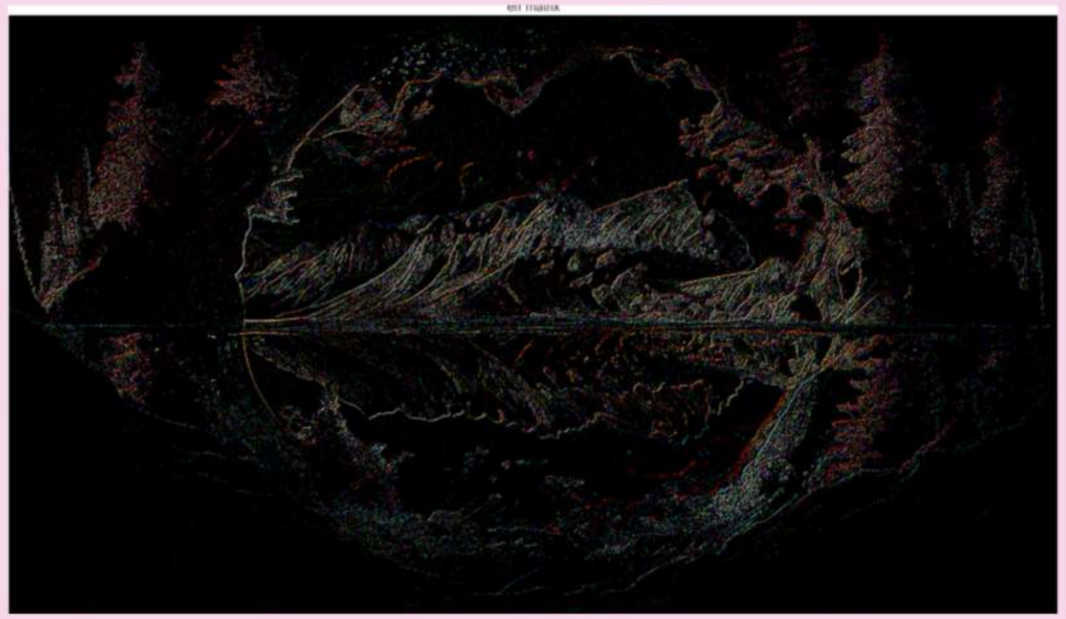
%Prediction

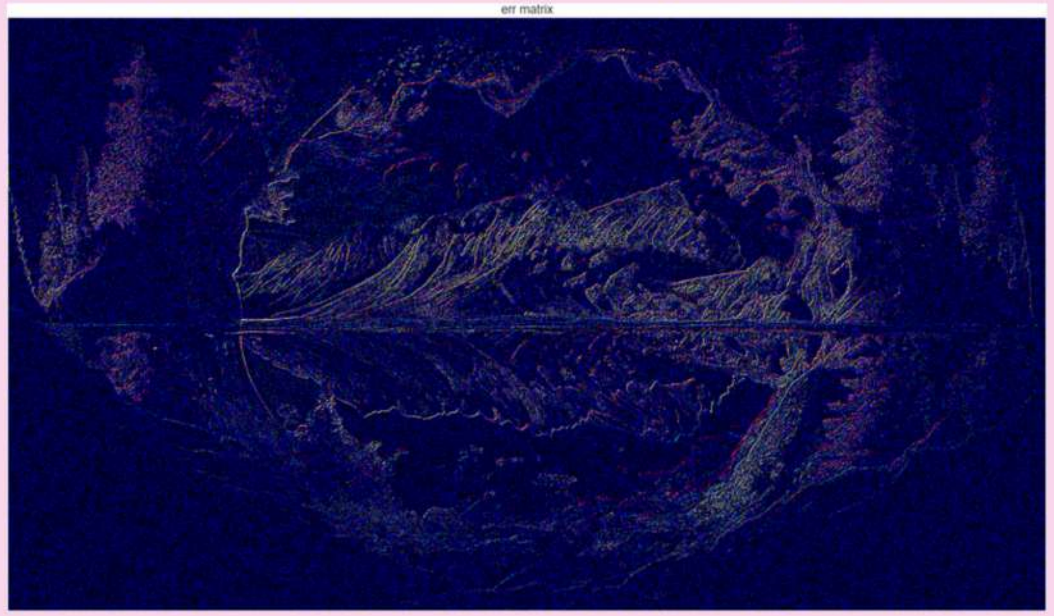
MSE: R=0.50, G=0.50, B=0.50

PSNR: R=51.16 dB, G=51.16 dB, B=51.18 dB

Entrpy,R=2.39, G=2.38, B=2.42

CROSS-CHANNEL VS SAME-CHANNEL

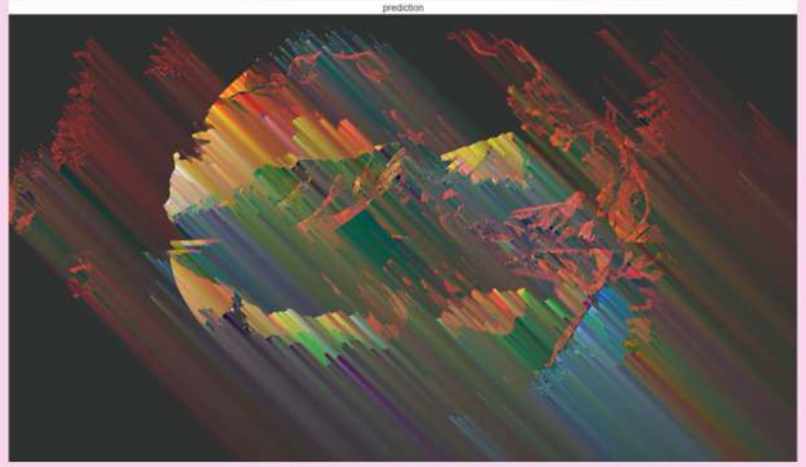




MSE: R=26.33, G=26.07, B=24.58
PSNR: R=33.93 dB, G=33.97 dB, B=34.23 dB
Entropy,R=0.23, G=0.20, B=0.18

MSE: R=24.76, G=22.85, B=33.07
PSNR: R=34.19 dB, G=34.54 dB, B=32.94 dB
Entropy,R=0.23, G=0.20, B=0.98

When we enlarge in on the step size



>> Cross-Channel
MSE: R=1646.04, G=645.75, B=2264.01
PSNR: R=15.97 dB, G=20.03 dB, B=14.58 dB
Entropy,R=0.01, G=0.00, B=0.78

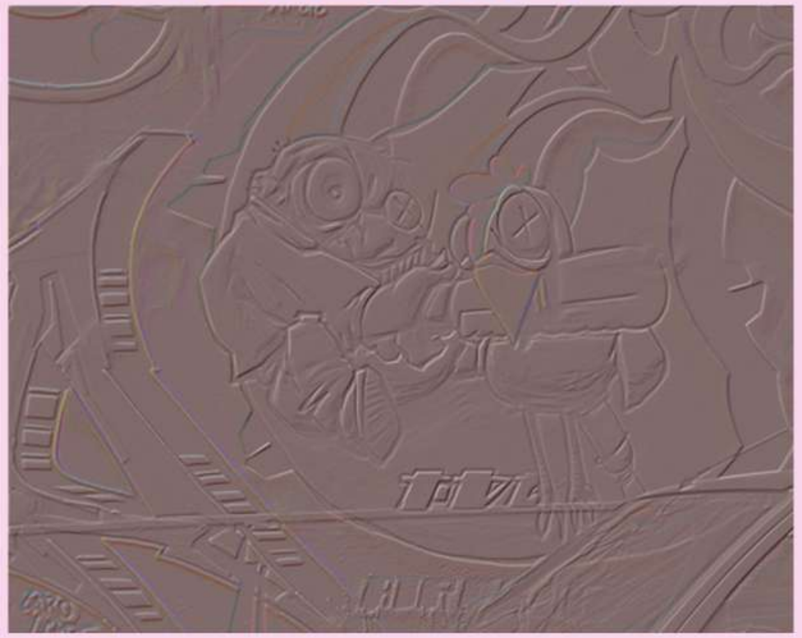
>> Same-Channel
MSE: R=2155.06, G=1529.47, B=1183.04
PSNR: R=14.80 dB, G=16.29 dB, B=17.40 dB
Entropy,R=0.01, G=0.00, B=0.00

RESULTS: GLOBAL

Original

640 x 800 pixels

Global Prediction
PSNR: 12.28 dB



Global Error
MSE: 3849.76

Computation time

0.11s

Entropy reduction

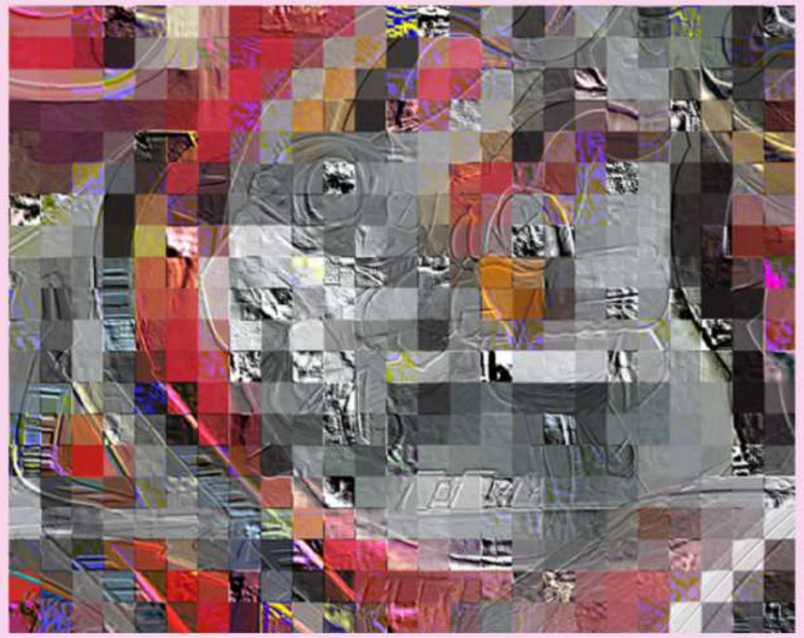
34.1%

RESULTS : LOCAL BLOCS 32

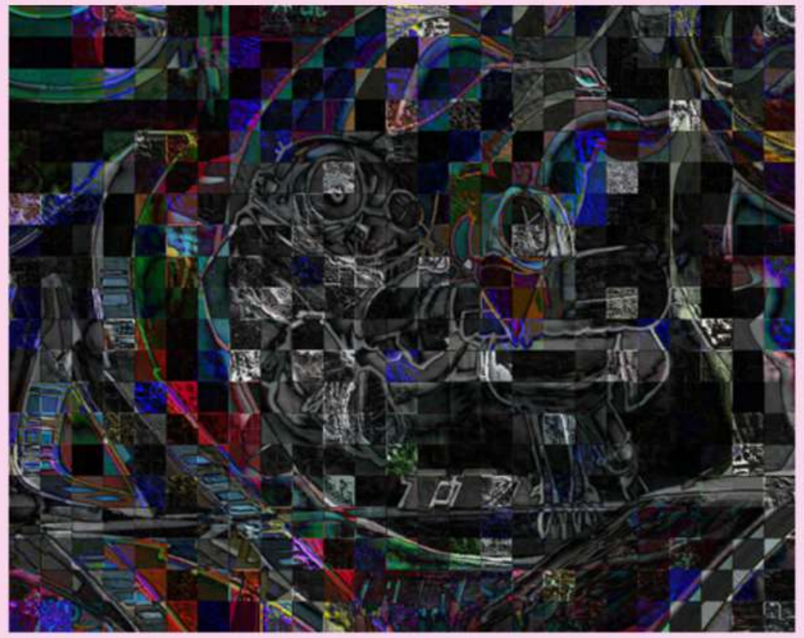
Original

640 x 800 pixels

Local Prediction
PSNR: 12.80 dB



Local Error
MSE: 3414.86



Computation time

93.16s

Entropy reduction

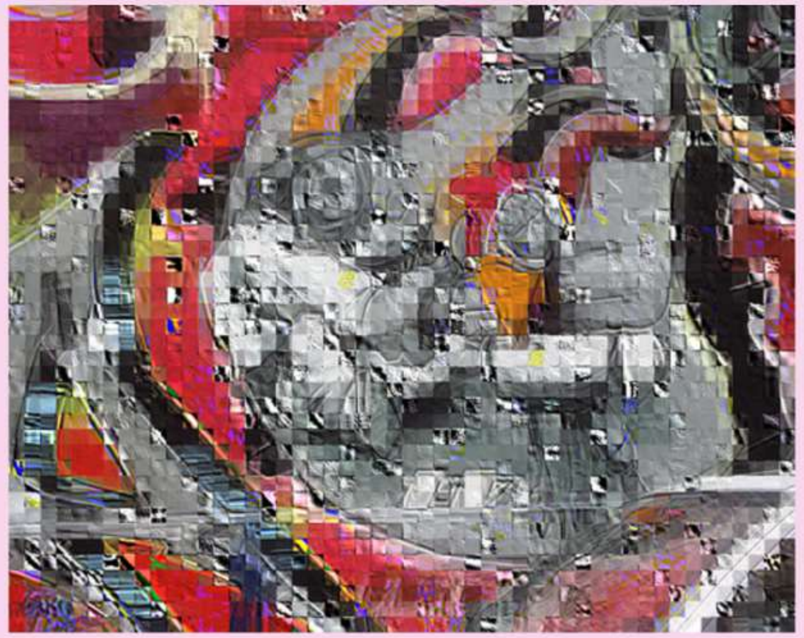
15.1%

RESULTS : LOCAL BLOCS 16

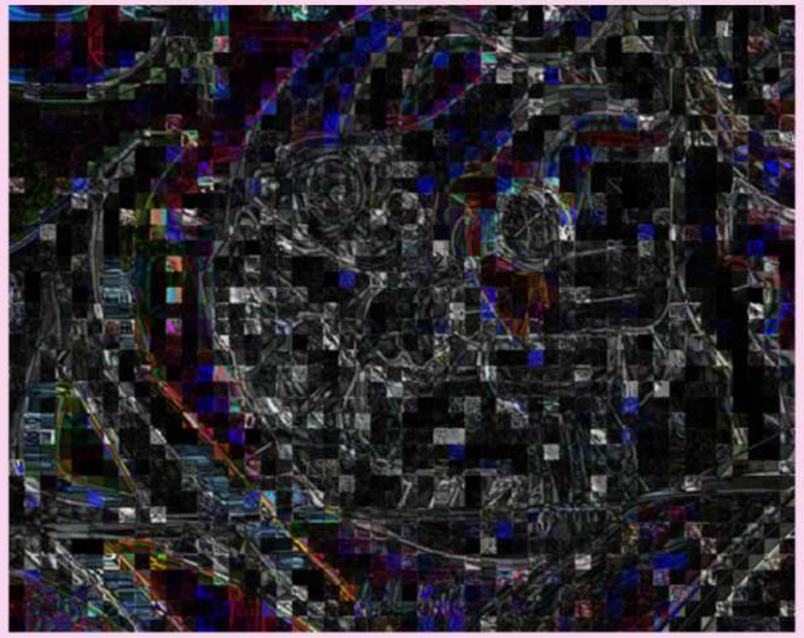
Original

640 x 800 pixels

Local Prediction
PSNR: 13.24 dB



Local Error
MSE: 3083.54



Computation time

178.20s

Entropy reduction

10.9%

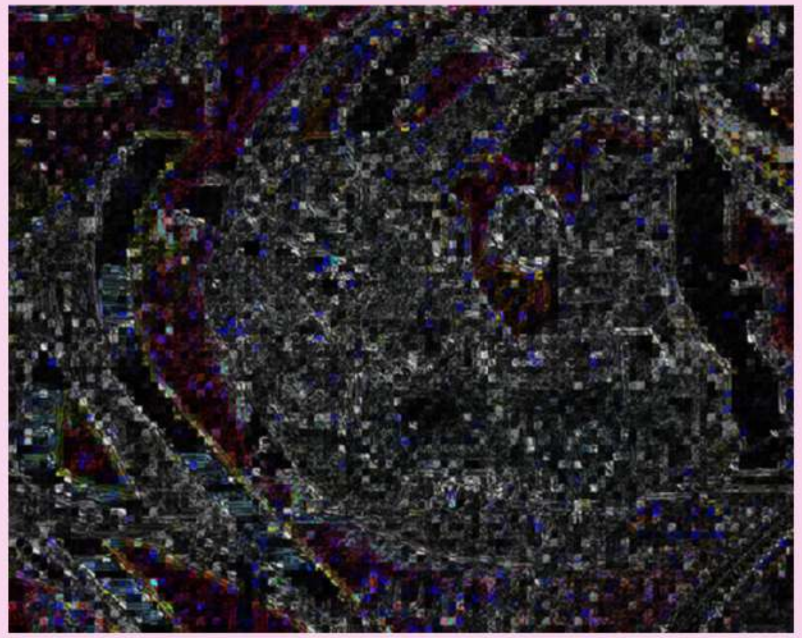
RESULTS : LOCAL BLOCS 8

Original

640 x 800 pixels

Local Prediction
PSNR: 13.17 dB

Local Error
MSE: 3134.52



Computation time

645.55s

Entropy reduction

7.4%

COMPARAISON

Methods	32	16	8	Global
Entropy quantized err (bits/pixel)	6.492	6.812	7.081	5.041
PSNR (dB)	12.80	13.24	13.17	12.28
MSE	3414.86	3083.54	3134.52	3849.78
Computation Time (en s)	93.16	178.20	645.55	0.11
Entropy reduction	15.1%	10.9%	7.4%	34.1%

APPLICATIONS

Methods	32	16	8	Global
Using	Economic Batch Processing	Professionnal archiving	Avoid	Real-time streaming/IOT

POSSIBLE OPTIMIZATION

Block parallelization
(÷4 to ÷8 time reduction)

Hybrid method
Global for smooth areas, 16×16 for details

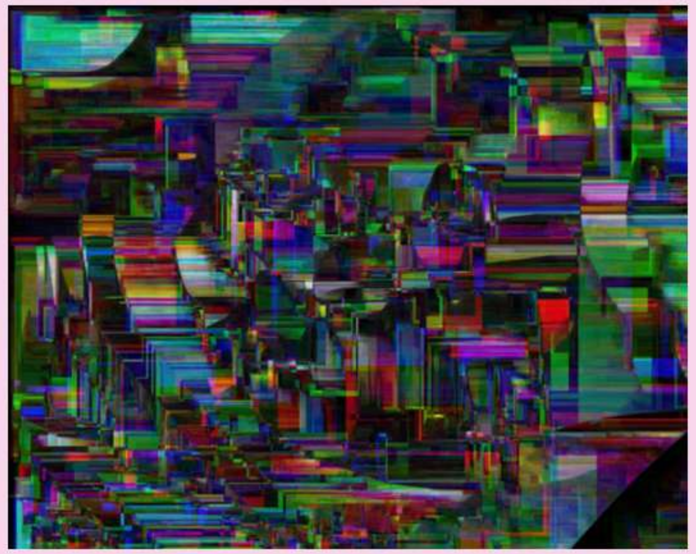
Dynamic adaptation of size based on local complexity

COMPARAISON TO OTHER METHODS

Entropy = 5.185

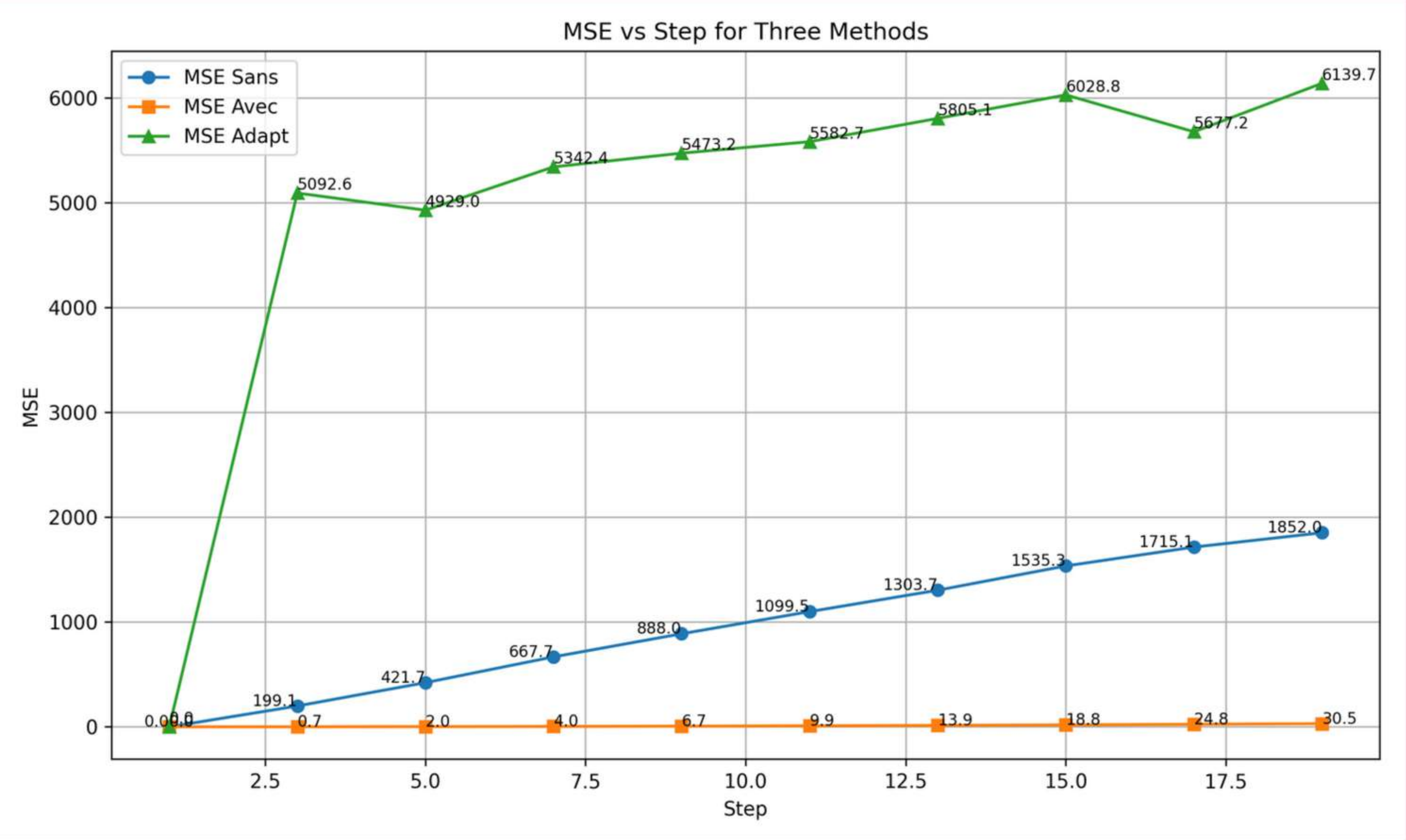
Entropy = 4.985

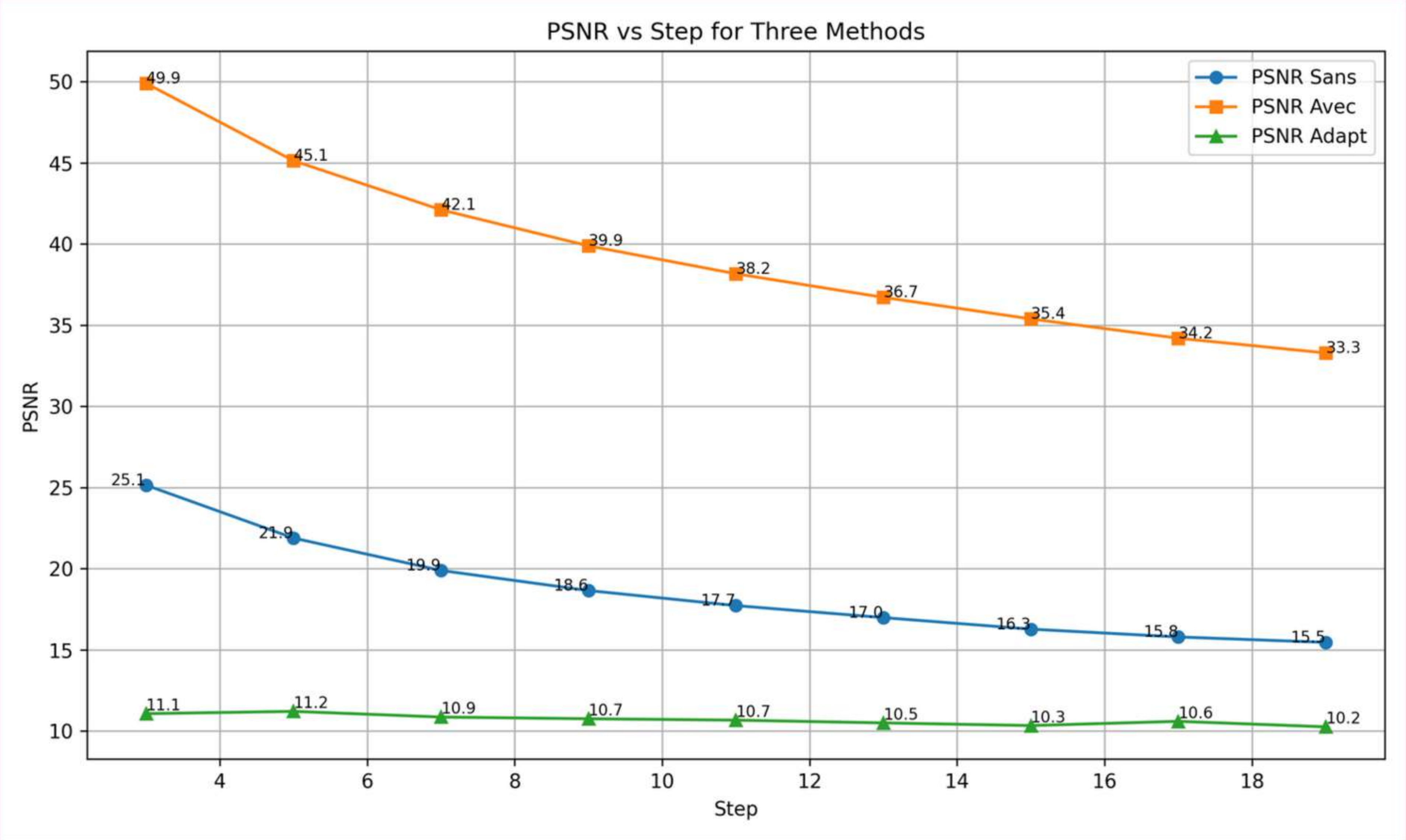
Entropy = 5.185



Entropy = 3.42

COMPARAISON TO OTHER METHODS





ACADEMIC RESSOURCES

Fundamental Theory

- Makhoul, J. (1975) - "Linear Prediction: A Tutorial Review", Proc. IEEE
- Jain, A.K. (1981) - "Image data compression: A review", Proc. IEEE
- Netravali, A.N. & Limb, J.O. (1980) - "Picture coding: A review", Proc. IEEE

RGB Predictive Coding

- "Interplane prediction for RGB video coding" - IEEE Conference, IEEE Xplore
- "High-Fidelity RGB Video Coding Using Adaptive Inter-Plane Weighted Prediction" - IEEE Journals, IEEE Xplore
- "A lossless image coding technique exploiting spectral correlation on the RGB space" - IEEE Conference, IEEE Xplore

Advanced Applications

- "Linear prediction image coding using iterated function systems" - ScienceDirect, ScienceDirect
- "Predictive Coding - Overview" - ScienceDirect Topics, ScienceDirect