Contents

Todo list

Chapter 1

Theory

Here is the theory. [1]

Table 1.1: The beginnings of the Golomb and Rice codes for a few parameter values. The midpoint (·) separates the high-order (unary) part from the low-order (binary) part of the codewords. The codes can be extended to all values of $n \geq 0$.

Golomb	$m = 1$	$m = 2$	$m = 4$	$m = 8$
Rice	$k = 0$	$k = 1$	$k = 2$	$k = 3$
$n = 0$	**0·**	**0·0**	**0·00**	**0·000**
1	**10·**	**0·1**	**0·01**	**0·001**
2	**110·**	**10·0**	**0·10**	**0·010**
3	**1110·**	**10·1**	**0·11**	**0·011**
4	**11110·**	**110·0**	**10·00**	**0·100**
5	**111110·**	**110·1**	**10·01**	**0·101**
6	**1111110·**	**1110·0**	**10·10**	**0·110**
7	**11111110·**	**1110·1**	**10·11**	**0·111**
8	**111111110·**	**11110·0**	**110·00**	**10·000**
9	**1111111110·**	**11110·1**	**110·01**	**10·001**
⋮	⋮	⋮	⋮	⋮

$P - L$	0	1	2	3	4
codeword	**111**	**10**	**00**	**01**	**110**

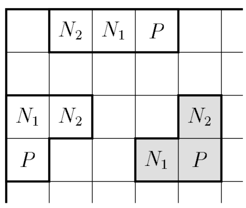


Figure 1.1: Nearest neighbors N_1 and N_2 of pixel P.

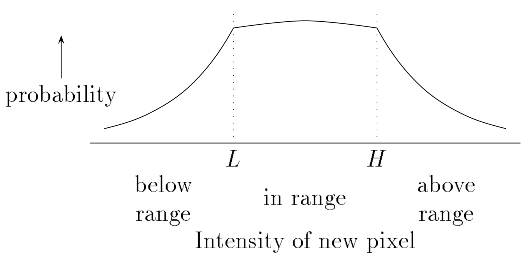
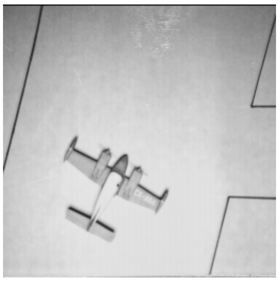


Figure 1.2: Scematic probability distribution of pixel values of P given L and H.

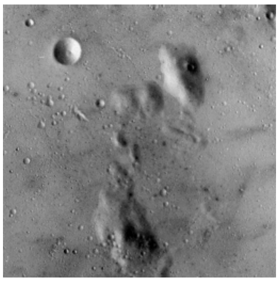
(a) Aerial



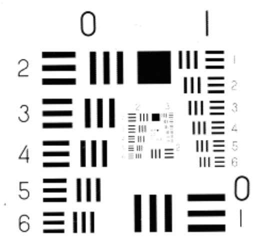
(b) Airplane

(c) Chemical plant

(d) Clock



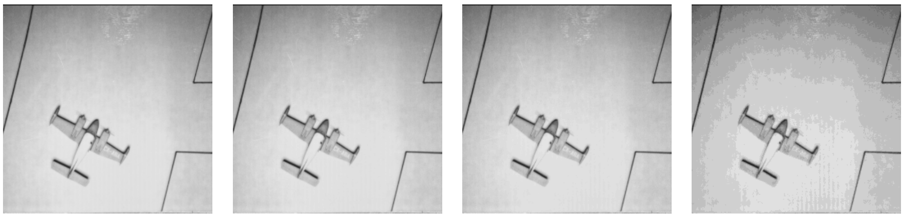
(e) Moon surface



(f) Resolution chart

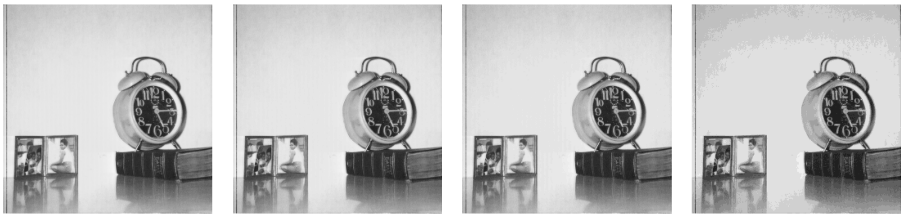
Figure 1.3: 256x256 pixel 8-bits grayscale test images [2]

(a) Aerial **(b)** **(c)** **(d)**

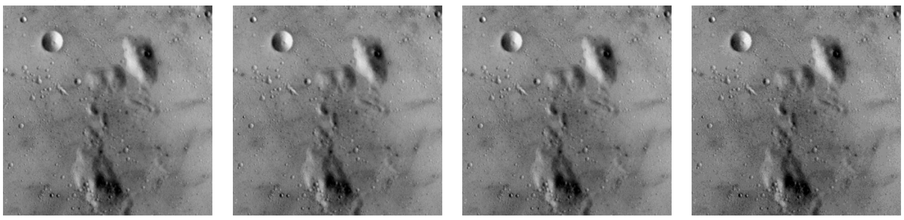


(e) Airplane **(f)** **(g)** **(h)**

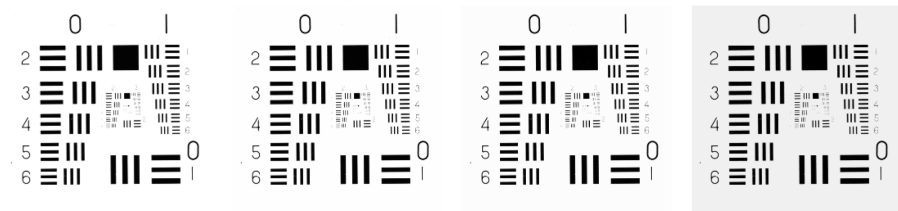
(i) Chemical plant **(j)** **(k)** **(l)**



(m) Clock **(n)** **(o)** **(p)**



(q) Moon surface **(r)** **(s)** **(t)**



(u) Res. chart **(v)** **(w)** **(x)**

Figure 1.4: Restored images after lossy compression. Left pictures are the originals, right of then are the results of the Truncate1 compresseion, then Truncate2 and on the right Truncate4.

$$
\begin{array}{cccccccc}
a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\
b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \\
c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_0 \\
d_7 & d_6 & d_5 & d_4 & d_3 & d_2 & d_1 & d_0 \\
e_7 & e_6 & e_5 & e_4 & e_3 & e_2 & e_1 & e_0 \\
f_7 & f_6 & f_5 & f_4 & f_3 & f_2 & f_1 & f_0 \\
g_7 & g_6 & g_5 & g_4 & g_3 & g_2 & g_1 & g_0 \\
h_7 & h_6 & h_5 & h_4 & h_3 & h_2 & h_1 & h_0
\end{array}
$$

(a) 8 uncompressed bytes.

$$
\begin{array}{cccccccc}
a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & \_ \\
b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & \_ \\
c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & \_ \\
d_7 & d_6 & d_5 & d_4 & d_3 & d_2 & d_1 & \_ \\
e_7 & e_6 & e_5 & e_4 & e_3 & e_2 & e_1 & \_ \\
f_7 & f_6 & f_5 & f_4 & f_3 & f_2 & f_1 & \_ \\
g_7 & g_6 & g_5 & g_4 & g_3 & g_2 & g_1 & \_ \\
h_7 & h_6 & h_5 & h_4 & h_3 & h_2 & h_1 & \_
\end{array}
$$

(b) Compressing 8 to 7 bytes.

$$
\begin{array}{cccccccc}
a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & h_1 \\
b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & h_2 \\
c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & h_3 \\
d_7 & d_6 & d_5 & d_4 & d_3 & d_2 & d_1 & h_4 \\
e_7 & e_6 & e_5 & e_4 & e_3 & e_2 & e_1 & h_5 \\
f_7 & f_6 & f_5 & f_4 & f_3 & f_2 & f_1 & h_6 \\
g_7 & g_6 & g_5 & g_4 & g_3 & g_2 & g_1 & h_7
\end{array}
$$

(c) 7 compressed bytes.

Figure 1.5: Truncate1 compression algorithm.

$$
\begin{array}{cccccccc}
a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\
b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \\
c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_0 \\
d_7 & d_6 & d_5 & d_4 & d_3 & d_2 & d_1 & d_0
\end{array}
$$

(a) 4 uncompressed bytes.

$$
\begin{array}{cccccccc}
a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & \_ & \_ \\
b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & \_ & \_ \\
c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & \_ & \_ \\
d_7 & d_6 & d_5 & d_4 & d_3 & d_2 & \_ & \_
\end{array}
$$

(b) Compressing 4 to 3 bytes.

$$
\begin{array}{cccccccc}
a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & d_3 & d_2 \\
b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & d_5 & d_4 \\
c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & d_7 & d_6
\end{array}
$$

(c) 3 compressed bytes.

Figure 1.6: Truncate2 compression algorithm.

$$\begin{array}{cccccccc} a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\ b_7 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \end{array}$$

(a) 2 uncompressed bytes.

$$\begin{array}{cccccccc} a_7 & a_6 & a_5 & a_4 & \_ & \_ & \_ & \_ \\ b_7 & b_6 & b_5 & b_4 & \_ & \_ & \_ & \_ \end{array}$$

(b) Compressing 2 to 1 bytes.

$$\begin{array}{cccccccc} a_7 & a_6 & a_5 & a_4 & b_7 & b_6 & b_5 & b_4 \end{array}$$

(c) 1 compressed bytes.

Figure 1.7: Truncate4 compression algorithm.

Bibliography

[1] Paul G. Howard and Jeffrey Scott Vitter. Fast and efficient lossless image compression. In *in Proc. 1993 Data Compression Conference, (Snowbird)*, pages 351–360, 1993.

[2] USC Viterbi School of Engineering. The usc-sipi image database. `http://sipi.usc.edu/database/`. [Online; accessed May 4, 2016].