Asian Journal of Information Technology 6 (3): 288-295, 2007

Speech Signal Compression Using 2D Ridgelet Transform

W.A. Jassim
Department of Computer Engineering, Baghdad University, Iraq

Abstract: Speech Coding or Speech Compression is the field concerned with obtaining compact digital representations of voice signals for the purpose of efficient transmission or storage. This study presents new method for speech compression techniques using new mathematical transform, namely, the 2D Ridgelet Transform. Recently in last years, this new transform is widely used in applications of image signal processing like image enhancement, image compressions and other applications. The mathematical result of this study give us good quality reconstructed speech signal after compressed with different compression ratios as shown in next pages.

Key words: Speech compression, Ridgelet Transform, radon transform, speech coding, wavelets

INTRODUCTION

Speech signals are non-stationary and at best they can be considered as quasi-stationary over short segments, typically 5-20 ms. The statistical and spectral properties of speech are thus defined over short segments. Speech can generally be classified as voiced phonemes (e.g., /a/, /i/, etc), unvoiced phonemes (e.g., /sh/), or mixed. Time and frequency domain plots for sample voiced and unvoiced segments are shown in Fig. 1. Voiced speech is quasi-periodic in the time-domain and harmonically structured in the frequency-domain while unvoiced speech is random-like and broadband. In addition, the energy of voiced segments is generally higher than the energy of unvoiced segments.

Speech coding involves sampling and amplitude quantization. While the sampling is almost invariably done at a rate equal to or greater than twice the bandwidth of analog speech, there has been a great deal of variability among the proposed methods in the representation of the sampled waveform. The objective in speech coding is to represent speech with a minimum number of bits while maintaining its perceptual quality.

The quantization or binary representation can be direct or parametric. Direct quantization implies binary representation of the speech samples themselves while parametric quantization involves binary representation of speech model and/or spectral parameters (Spanias, 1994).

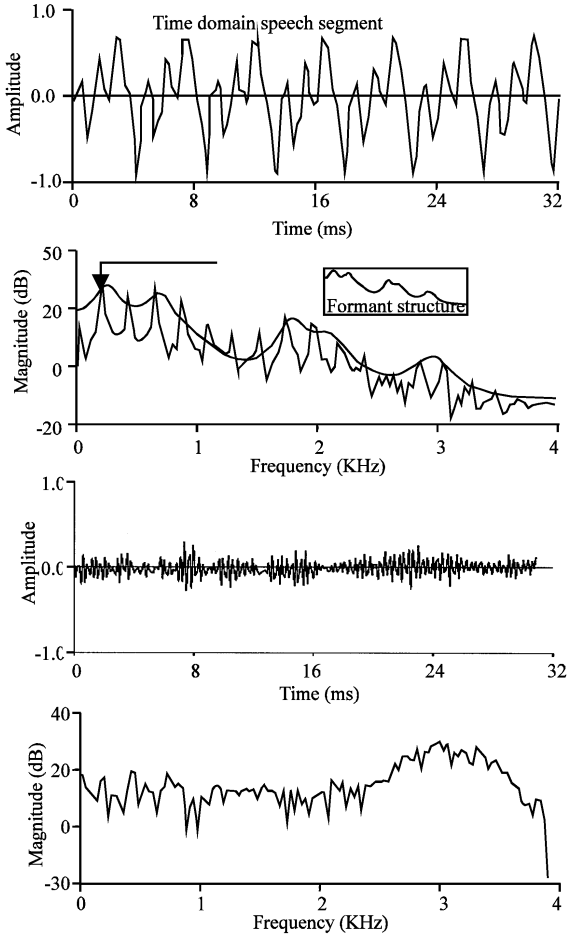


Fig. 1: Voiced and unvoiced segments and their short-term spectra

CLASSICAL SPEECH COMPRESSION SYSTEM

A speech coder consists of two components: - the encoder and the decoder. Speech is a time varying waveform. The analog speech signal S(t) is first sampled at the rate Fs≥2 Fmax, Where Fmax is the maximum frequency content of S(t). The sampled discrete time signal is denoted by S(n). This signal is then encoded

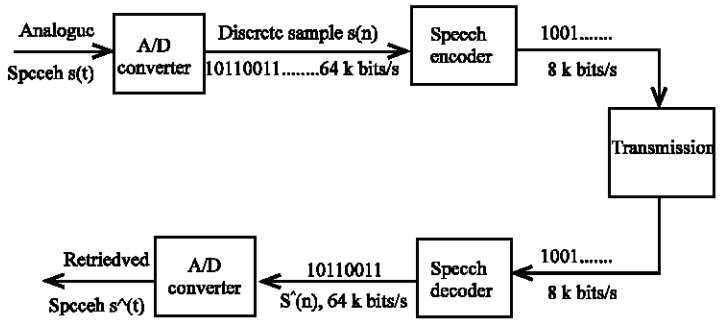


Fig. 2: Process of speech signal

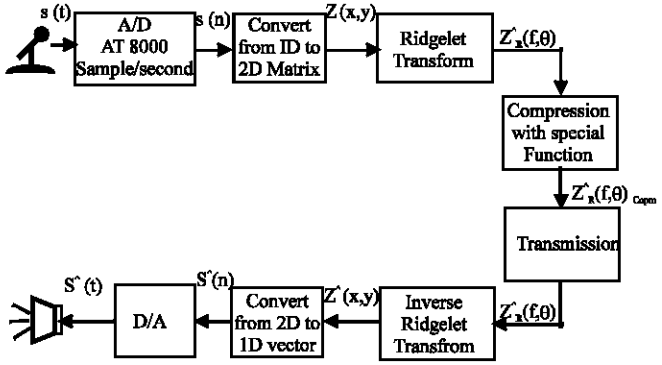


Fig. 3: The speech siginal converted to image signal

using one of several coding schemes such as PCM (Pulse Code Modulation) or predictive coding. The decoder reconstructs the speech signal from those transmitted parameters. The whole process is illustrated in Fig. 2, (Tamanna, 2000; Wissam, 2002).

THE PROPOSED SPEECH COMPRESSION SYSTEM USING RIGHLET TRAMSFORM

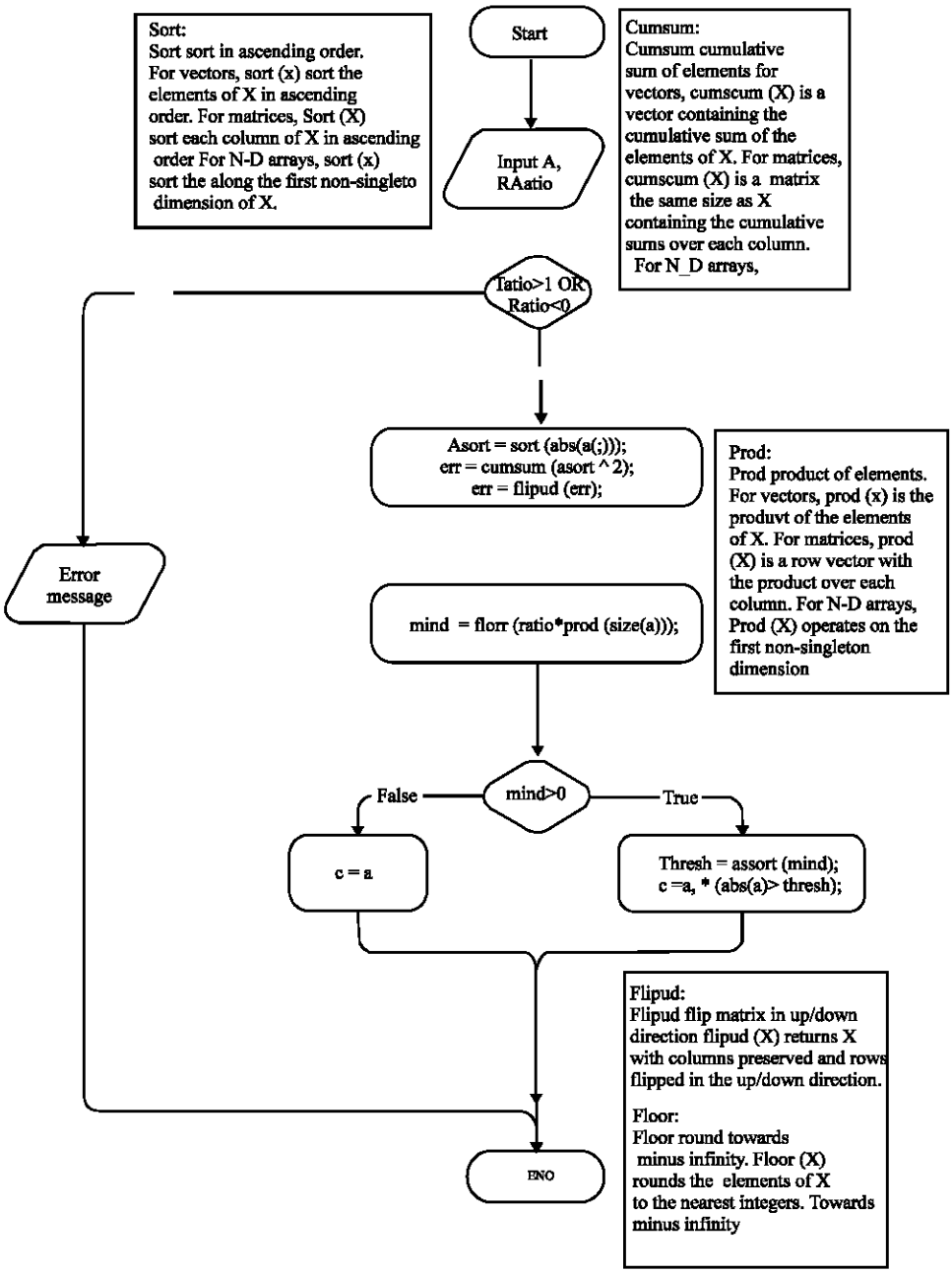
The new system contains same main components block diagrams of classical system but with main difference that the speech signal output from A/D converter is converted from 1D signal or vector to 2D matrix or image signal. This image is then entered to system for image compression using Ridgelet transform and then transmits the compressed image to receiver side. At, receiver side, the same system with inverse operation is used to reconstruct the original speech signal as shown in Fig. 3. Also, the above procedure can be summarized by the following steps:

- Record original speech signal via MIC for period like 9.5 second.
- Convert analog signal to sampled signal with sampling frequency like 8000 Hz (sample/second) to get speech vector of 75625 samples.
- Convert 1D vector signal S(n) into 2D matrix with size of (275*275) samples to get Z(x,y).
- Now, convert image signal from spatial domain to Ridgelet domain using Ridgelet transform to get $Z_R(f,\theta)$.
- Apply compression techniques on signal in Ridgelet domain with different ratios of compression to get compressed image and then transmit to receiver side as image signal $[ZR(f,\theta)]_{Comp.}$
- At receiver side, take received signal and apply inverse Ridgelet transform to retune signal to original domain and get $\hat{Z}_R(f,\theta)$.
- Convert image signal to 1D vector which represented the reconstructed speech signal $\hat{S}(n)$

RIDGELET TRAMSFORM

This transform consists of manly steps. The first step is to compute the Radon transform $Rf(t,\theta)$ and second, to apply a 1-D wavelet transform to the slices $Ef(\bullet,\theta)$. Radon transform can be summarized according to following steps (Donoho, 1997; Digital Ridgelet Transform, 1998):

- 2-D FFT. Compute the 2-D FFT of giving the array (input image).



Flow chart 1: Shows compression function

- Cartesian to Polar Conversion. Using an interpolation scheme, substitute the sampled values of the Fourier transform obtained on the square lattice. That is, on a lattice where the points fall on lines going through the origin.
- 1-D IFFT. Compute the 1-D IFFT on each line, i.e., for each value of the angular parameter.

Figure 4, shows the flow graph of Ridgelet traansfrom (Jean *et al*., 2002).

COMPRESSION FUNCTION

As shown in Fig. 3, the speech signal converted to image signal form output from Ridgelet transform; i.e., in

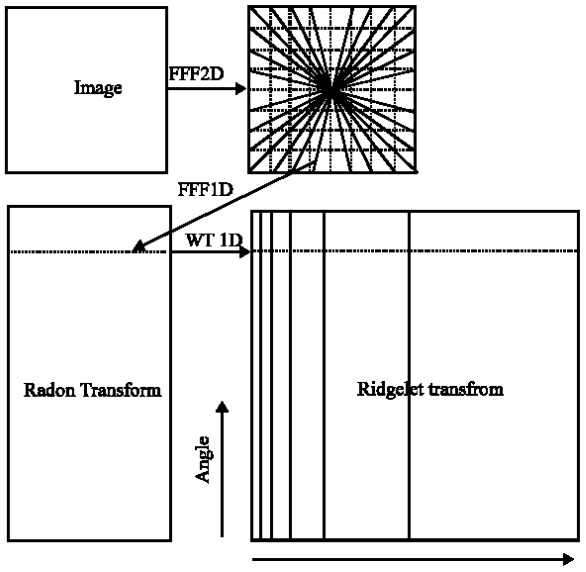


Fig. 4: The flow graph of Ridgelet Transform

Ridgelet domain; is represented the input to special function of compression. This function has two states depends on compression ratio value entered by user. The two states is one of the following:

- Zero value: - If the sample (of image in Ridgelet domain) is less than or equal to threshold value, then the sample value is equals to zero. Threshold value is equal to the value of sample which has index equal to mind value. Mind value is calculated depends on compression ratio entered by user.
- Same sample value if its value is greater than threshold value.

The all flow chart of compression function is shown in flow chart 1.

SIMULATION RESULTS

All the techniques described in the previous Section have been implemented using MATLAB (version 6.5) package and tested over different kinds of speech signals. The following steps show more details about simulation results:

Figure 5, shows the original test speech signal S(n) which contains 66049 samples at sampling frequency of 8000 Hz (sample/sec), i.e., 8.256 sec.

Figure 6, shows the original test speech signal after conversion into 2D image signal Z(x,y) and the size is (257*257) samples.

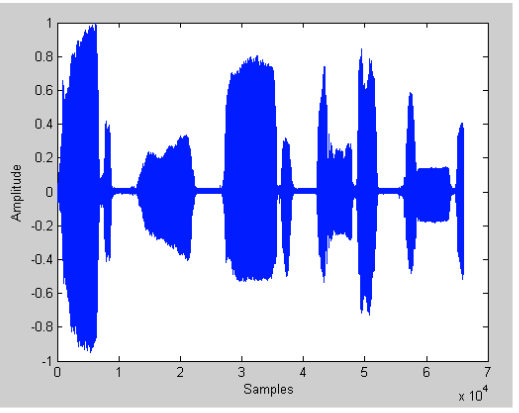


Fig. 5: Original speech signal

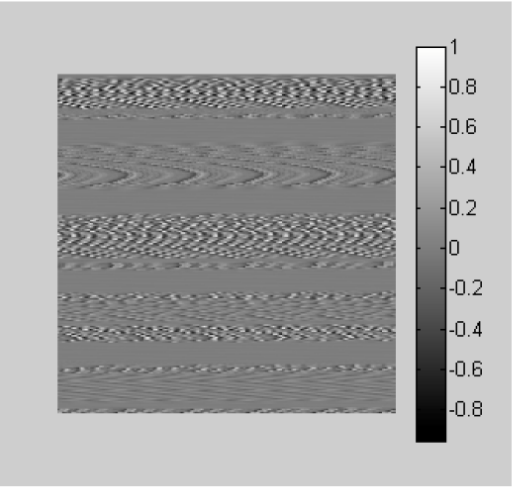


Fig. 6: Original speech singnal

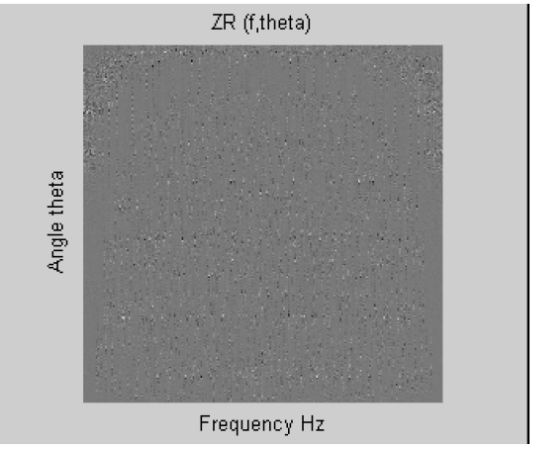


Fig. 7: Z_R (f,θ)

Table 1: Shows the first (50*10) values of $Z_R$matrix, so that we can compare its values after compression function

No.	1	2	3	4	5	6	7	8	9	10
1	0.0378	0.0658	0.0693	-0.0092	0.0277	0.0613	-0.0599	-0.0178	-0.0423	-0.0188
2	0.0037	-0.025	0.0125	-0.0083	0.0247	-0.0346	0.0028	-0.0107	-0.005	-0.0046
3	0.0226	0.0691	0.0629	-0.0122	0.0318	0.0345	-0.0528	-0.0252	-0.0109	-0.0327
4	0.0174	-0.0046	-0.0208	0.0198	0.0103	0.012	-0.0509	-0.0366	-0.0205	-0.1635
5	-0.0135	0.0706	0.0349	-0.0287	0.0082	0.0464	-0.0792	-0.0264	-0.0491	0.0377
6	0.0244	-0.0009	0.0263	0.0294	-0.0549	0.0034	-0.05	-0.019	0.0178	0.0615
7	0.0238	0.1154	0.1157	-0.1325	-0.0064	-0.0067	-0.1591	-0.0837	-0.0423	0.0365
8	0.0118	0.1077	0.1277	-0.1399	0.261	-0.0992	0.0478	-0.0162	-0.0927	-0.2701
9	0.0017	0.0291	-0.1493	0.0547	0.1372	0.115	0.1716	-0.0199	-0.2249	0.1741
10	0.0127	0.0226	-0.0148	0.1241	-0.2084	0.0197	-0.0381	-0.1561	0.0596	-0.1649
11	0.0133	0.0288	0.1673	0.0273	-0.0139	0.1659	-0.1713	0.0221	0.1125	0.0907
12	-0.0194	-0.0467	-0.0403	0.1262	0.0424	-0.0371	-0.0535	-0.0321	-0.0788	-0.0581
13	0.0083	0.1556	0.1647	0.1384	0.0481	0.1896	-0.0444	-0.0112	1.00E-04	0.2179
14	0.0016	-0.0717	-0.0366	-0.0903	-0.0312	0.0115	0.0185	-0.0127	-0.0018	0.4208
15	-0.0191	0.0438	-0.0173	-0.1226	-0.0517	0.2837	-0.1343	-0.1226	0.0119	-0.1134
16	0.0219	-0.0657	0.1057	-0.0148	-0.1729	0.0145	0.0363	0.1415	0.0876	-0.2676
17	-0.0023	0.0438	0.042	0.2574	0.1265	-0.1339	-0.1671	-0.0748	-0.228	-0.1859
18	0.0049	-0.3684	-0.311	-0.0428	0.4704	0.4129	-0.0389	0.1713	0.0952	-0.2699
19	0.009	-0.7374	-0.165	-0.0779	-0.0595	0.0839	-0.1003	0.1657	0.1761	-0.0019
20	-0.0351	0.7913	-0.0182	-0.378	-0.3411	-0.1192	-0.1265	-0.3709	-0.1426	0.5036
21	0.0095	0.5568	-0.2622	-0.1324	0.1587	0.3803	0.0562	0.2457	0.1523	0.2113
22	-0.0068	-0.0166	-0.2754	-0.1411	-0.1546	0.0337	-0.2512	0.2285	0.189	-0.2624
23	0.0078	0.3651	0.0786	-0.1944	0.1539	0.0421	0.0152	-0.0656	0.0307	-0.4323
24	0.0106	-0.0353	-0.0281	-0.1241	0.1028	-0.0254	-0.0375	-0.0459	0.1194	0.7328
25	-0.0045	0.0908	-0.0591	-0.0764	-0.0194	0.2068	0.1844	-0.2473	0.2764	0.0478
26	0.0026	-0.0876	-0.0767	0.3182	-0.2654	-0.0127	0.0176	-0.1086	-0.0587	-0.1511
27	0.0043	-0.0309	0.0826	0.0584	0.0526	0.1805	0.087	-0.1042	0.1423	0.1187
28	0.0068	-0.0573	-0.337	-0.1172	0.1036	-0.0224	0.2201	-0.0248	-0.0579	-0.241
29	0.0029	0.4401	-0.3134	-0.3617	-0.0169	0.1142	-0.5192	-0.207	0.1958	0.3595
30	0.0049	0.2003	-0.3276	0.2563	-0.2027	-0.4544	0.0896	-0.1073	0.4114	0.0185
31	-0.0026	0.355	0.0695	-0.236	-0.223	0.2565	0.2396	0.1597	-0.0076	-0.03
32	0.005	-0.0571	-0.1611	-0.261	-0.0928	-0.3827	0.0262	-0.0378	0.0449	0.0006
33	0.0037	0.2503	0.364	-0.0142	0.0866	-0.049	0.0319	-0.232	-0.0488	0.3693
34	0.0109	-0.1481	-0.025	0.0079	-0.0304	0.1487	-0.0207	0.1096	0.0454	-0.4115
35	0.0315	0.0535	0.1485	-0.2942	0.0897	0.0651	-0.0556	-0.0247	0.1871	0.0195
36	0.0176	-0.1491	-0.1113	-0.0622	0.1633	0.1312	-0.058	0.3503	0.0643	-0.0365
37	-0.0429	0.5102	0.1789	0.2668	-0.0207	-0.1666	0.1912	-0.0846	-0.0261	-0.161
38	-0.051	0.2346	0.4586	0.1914	-0.5558	0.1027	0.1251	-0.1097	0.416	-0.2428
39	0.0172	0.2124	0.0855	-0.1931	-0.044	0.2055	-0.3198	-0.0568	0.1474	0.2322
40	-1.00E-04	-0.1745	0.0194	-0.3824	0.0709	-0.1364	0.3416	0.0107	0.187	-0.0934
41	0.0052	0.7284	0.0129	-0.6321	-0.094	0.6551	0.0415	-0.334	-0.1972	0.3351
42	0.0097	0.1283	-0.3862	-0.1262	-0.3414	-0.2703	0.0987	-0.5752	-0.3267	0.176
43	0.0171	0.3447	-0.6321	-0.337	-0.0344	0.2884	-0.0302	0.0418	0.2078	0.0445
44	0.0186	0.1997	0.1107	0.0464	-0.3602	-0.216	0.0427	-0.7055	-0.2506	-0.1414
45	-0.0003	1.3727	0.3104	-0.8755	0.5907	0.4859	0.2893	-0.0774	-0.2642	-0.5496
46	-0.0028	0.1024	-0.2709	-0.0764	-0.2862	-0.572	0.3052	0.6946	-0.0879	0.001
47	0.0043	0.7818	0.0125	-0.5392	0.8697	0.2491	-0.1449	-0.6653	-0.2625	-0.2089
48	0.0136	0.3745	-0.8528	1.4131	0.5333	0.2024	0.8909	-0.344	-0.7219	-0.0646
49	-0.0027	-1.6585	0.1586	1.2733	-0.5828	-1.0141	0.4601	0.2936	0.7431	-0.1556
50	-0.0293	-0.2935	0.2266	-0.0753	-0.1034	-0.0673	0.2935	-0.0541	-0.292	0.0001

Figure 7 shows Z(x,y) matrix after conversion to Ridgelet domain to get $Z_R(f,\theta)$.

Table 1, show the frist (50*10) values of Z_R matrix, so that we can compare its values after compression function.

Figure 8 shows $Z_R(f,\theta)$ matrix after compression function with compression ratio of 0.6 to get $[Z_R(f,\theta)]_{Comp}$.

Table 2 shows the first (50*10) values of $[Z_R(f,\theta)]_{Comp}$ matrix, so that we can compare its values before compression function. Most of these values are zero because of threshold value.

Figure 9 shows $\hat{Z}_R(f,\theta)$ matrix after applying Inverse Ridgelet transform to get $\hat{Z}(x,y)$.

Figure 10 shows the reconstructed test speech signal $\hat{S}(n)$ which contains 66049 samples at sampling frequency of 8000 Hz (sample/sec), i.e., 8.256 sec.

Now, to calculate the reduction in storage size and SNR before and after compression:

- Number of total samples before compression is 257*257 = 66049 values.
- Number of zeros in $[Z_R(f,\theta)]_{Comp}$ is 39628 while (66049 - 39628 = 26421) samples has the same value without any change.
- New storage size after compression for transmission is: -

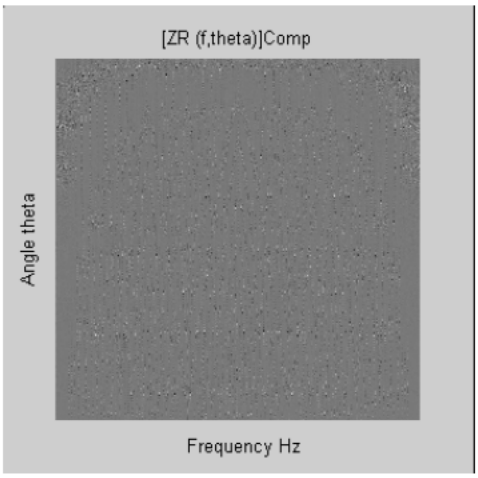


Fig. 8: $[Z_R (f,\theta)]_{comp}$ matrix

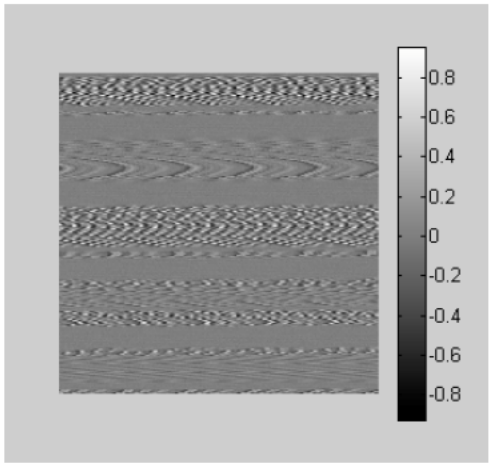


Fig. 9: $[Z^{\wedge}_R (f,\theta)]_{comp}$ matrix

Table 2: First 50*10 of $[Z_R(f\grave{e})]_{Comp}$ matrix

No.	1	2	3	4	5	6	7	8	9	10
1	0	0.0658	0.0693	0	0	0.0613	-0.0599	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0.0691	0.0629	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	-0.1635
5	0	0.0706	0	0	0	0	-0.0792	0	0	0
6	0	0	0	0	0	0	0	0	0	0.0615
7	0	0.1154	0.1157	-0.1325	0	0	-0.1591	-0.0837	0	0
8	0	0.1077	0.1277	-0.1399	0.261	-0.0992	0	0	-0.0927	-0.2701
9	0	0	-0.1493	0	0.1372	0.115	0.1716	0	-0.2249	0.1741
10	0	0	0	0.1241	-0.2084	0	0	-0.1561	0.0596	-0.1649
11	0	0	0.1673	0	0	0.1659	-0.1713	0	0.1125	0.0907
12	0	0	0	0.1262	0	0	0	0	-0.0788	0
13	0	0.1556	0.1647	0.1384	0	0.1896	0	0	0	0.2179
14	0	-0.0717	0	-0.0903	0	0	0	0	0	0.4208
15	0	0	0	-0.1226	0	0.2837	-0.1343	-0.1226	0	-0.1134
16	0	-0.0657	0.1057	0	-0.1729	0	0	0.1415	0.0876	-0.2676
17	0	0	0	0.2574	0.1265	-0.1339	-0.1671	-0.0748	-0.228	-0.1859
18	0	-0.3684	-0.311	0	0.4704	0.4129	0	0.1713	0.0952	-0.2699
19	0	-0.7374	-0.165	-0.0779	-0.0595	0.0839	-0.1003	0.1657	0.1761	0
20	0	0.7913	0	-0.378	-0.3411	-0.1192	-0.1265	-0.3709	-0.1426	0.5036
21	0	0.5568	-0.2622	-0.1324	0.1587	0.3803	0	0.2457	0.1523	0.2113
22	0	0	-0.2754	-0.1411	-0.1546	0	-0.2512	0.2285	0.189	-0.2624
23	0	0.3651	0.0786	-0.1944	0.1539	0	0	-0.0656	0	-0.4323
24	0	0	0	-0.1241	0.1028	0	0	0	0.1194	0.7328
25	0	0.0908	-0.0591	-0.0764	0	0.2068	0.1844	-0.2473	0.2764	0
26	0	-0.0876	-0.0767	0.3182	-0.2654	0	0	-0.1086	-0.0587	-0.1511
27	0	0	0.0826	0.0584	0	0.1805	0.087	-0.1042	0.1423	0.1187
28	0	0	-0.337	-0.1172	0.1036	0	0.2201	0	0	-0.241
29	0	0.4401	-0.3134	-0.3617	0	0.1142	-0.5192	-0.207	0.1958	0.3595
30	0	0.2003	-0.3276	0.2563	-0.2027	-0.4544	0.0896	-0.1073	0.4114	0
31	0	0.355	0.0695	-0.236	-0.223	0.2565	0.2396	0.1597	0	0
32	0	0	-0.1611	-0.261	-0.0928	-0.3827	0	0	0	0
33	0	0.2503	0.364	0	0.0866	0	0	-0.232	0	0.3693
34	0	-0.1481	0	0	0	0.1487	0	0.1096	0	-0.4115
35	0	0	0.1485	-0.2942	0.0897	0.0651	0	0	0.1871	0
36	0	-0.1491	-0.1113	-0.0622	0.1633	0.1312	0	0.3503	0.0643	0
37	0	0.5102	0.1789	0.2668	0	-0.1666	0.1912	-0.0846	0	-0.161
38	0	0.2346	0.4586	0.1914	-0.5558	0.1027	0.1251	-0.1097	0.416	-0.2428
39	0	0.2124	0.0855	-0.1931	0	0.2055	-0.3198	0	0.1474	0.2322
40	0	-0.1745	0	-0.3824	0.0709	-0.1364	0.3416	0	0.187	-0.0934
41	0	0.7284	0	-0.6321	-0.094	0.6551	0	-0.334	-0.1972	0.3351
42	0	0.1283	-0.3862	-0.1262	-0.3414	-0.2703	0.0987	-0.5752	-0.3267	0.176

Table 2 continued

No.	1	2	3	4	5	6	7	8	9	10
43	0	0.3447	-0.6321	-0.337	0	0.2884	0	0	0.2078	0
44	0	0.1997	0.1107	0	-0.3602	-0.216	0	-0.7055	-0.2506	-0.1414
45	0	1.3727	0.3104	-0.8755	0.5907	0.4859	0.2893	-0.0774	-0.2642	-0.5496
46	0	0.1024	-0.2709	-0.0764	-0.2862	-0.572	0.3052	0.6946	-0.0879	0
47	0	0.7818	0	-0.5392	0.8697	0.2491	-0.1449	-0.6653	-0.2625	-0.2089
48	0	0.3745	-0.8528	1.4131	0.5333	0.2024	0.8909	-0.344	-0.7219	-0.0646
49	0	-1.6585	0.1586	1.2733	-0.5828	-1.0141	0.4601	0.2936	0.7431	-0.1556
50	0	-0.2935	0.2266	-0.0753	-0.1034	-0.0673	0.2935	0	-0.292	0

Table 3: Simulation resutles

Compression ratio	RatioPSNR in (dB)	Number of zeroes	New Storage size after compression (%)
0.1	43.34	6604	90 % of original size
0.15	37.96	9907	85 % of original size
0.2	34.06	13209	80 % of original size
0.25	31.0	16512	75 % of original size
0.3	28.45	19814	70 % of original size
0.35	26.25	23116	65 % of original size
0.4	24.29	26419	60 % of original size
0.45	22.5	29721	55 % of original size
0.5	20.83	33024	50 % of original size
0.55	19.23	36326	45 % of original size
0.6	17.69	39628	40 % of original size
0.65	16.15	42931	35 % of original size
0.7	14.61	46233	30 % of original size
0.75	12.69	49536	25 % of original size
0.8	11.17	52838	20 % of original size
0.85	9.18	56140	15 % of original size
0.9	6.98	59443	10 % of original size
0.95	4.42	62745	5 % of original size

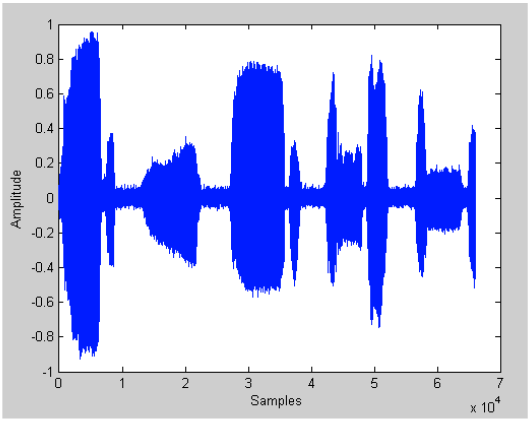


Fig. 10: Reconstructed speech signal $\hat{S}(n)$

$$\frac{\text{Total number of samples after compression}}{\text{Total number of samples before compression}} *100 = \frac{26421}{66049} *100 = 40\% \text{ of original size}$$

- This means that, the test speech signal after compression can be transmitted to receiver end with storage size is 40% less than the storage size without compression.
- Also, The PSNR is 17.69 dB.

Table 3 shows different values of compression ratio applied on the same test speech file with there corresponding PSNR values, number of zeroed samples, and the new storage size ratio of original size for transmission.

CONCLUSION

This research presented a new family of discrete transforms for speech signal processing merged with image processing based on the Ridgelet idea. The proposed Ridgelet transform is self-inverting - the inverse transform uses the same algorithm as the forward transform-and has excellent nucurves and there are texture regions (which generate point discontinuities), the Ridgelet transform is not optimal. Therefore, a more practical scheme in employing the Ridgelet transform is to first utilize a quad-tree division of images into suitable blocks where edges look straight and then apply the discrete Ridgelet transform to each block. Another scheme is to use the Ridgelet transform as the building block in a more localized construction such as the Curvelet transform.

Also, in this study, a computationally efficient approach to the construction of speech compression has been presented. These techniques may be useful in voice transmission systems with low storage size used and maintaining good voice quality. All simulation results

gained from this paper give good quality voice with very low size. This system can be used efficiently in offline meetings and conferences and also, in online transmission with no high perceptual delay due to collections speech data and converted in the buffer into image form.

Although, in this study, an attempt to treat speech signal with sampling frequency of 8000 Hz. So, we need approximately 8.256 sec to form image matrix of (257*257) samples in size. To reduce the delay time, we can increase sampling frequency to be like 48000 Hz. So, the time needed is 1.376 sec and so on. Also, another way to reduce delay time is by decreasing image size matrix input to Ridgelet transform to be like (5*5) samples or (17*17) samples which represent s the prime number limitations used in the program.

REFERENCES

Digital Ridgelet Transform via Rectopolar Coordinate Transform, 1998. Stanford University, Stanford, A, Technology Report.

Donoho, D.L., 1997. Fast ridgelet transforms in dimension 2, Stanford University, Department. Statistics, Stanford, CA, Technology Report.

Jean, L. Starck, Emmanuel J. Candès and David L. Donoho, 2002. The Curvelet Transform for Image Denoising IEEE Transactions on Image Processing, Vol. 11.

Spanias, A.S., 1994. Speech Coding: A Tutorial Review, A Tutorial Review of the IEEE., Vol. 82.

Tamanna Islam, 2000. Interpolation of Linear Prediction Coefficients for Speech Coding, M.Sc thesis, Department of Electrical Engineering, McGill University, Montreal, Canada,

Wissam, A. Jassim, 2002. Design and implementation of high speech quality vocoder system at 2400 bps using MELP model M.Sc thesis, Department of Electrical Engineering, University of Baghdad, Iraq.