Multimedia Systems assignment
GSM 06.10 vocodec

Thodoris Tyrovouzis 9369

March 8, 2022

1 Introduction

This document provides some general guidance for the codec's source code. My goal in this assignment was to follow the specification up to **Παραδοτέο Επιπέδου 3** as closely as possible, using simple and readable code. The source code for this assignment is hosted on GitHub at `https://github.com/Thodoris1999/GSMVocodec`.

2 Source code description

The source code contains the following utility functions.

packFrmBitStrm, unpackFrmBitStrm

Pack and unpack coded frame packets to and from the frame parameters.

lar, lar_inv

Transform LPC reflection coefficients to and from log area ratios

quantize_lars, dequantize_lars

Quantize/Dequantize log-area ratios.

LTP_gain_code, LTP_gain_decode

Code and decode long term prediction gains b_j.

preproc, postproc

Preprocess and postprocess procedures (paragraphs 3.1.1, 3.1.2, 3.2.4 of the standard).

acf

Estimates the autocorrelation function from the samples.

block_weight_filter

Applies the block weighting filter specified in the RPE encoding section.

RPE_decimation_selection

Downsample LTP residuals with grid selection

quantize_xprime, dequantize_xprime

Quantize/Dequantize LTP residuals.

ST_filtering

Short term analysis filtering (with interpolated log-area ratios)

ST_synthesis

Short term synthesis filtering (with interpolated log-area ratios)

The rest of the functions, **RPE_frame_coder**, **RPE_frame_decoder**, **RPE_frame_ST_coder**, **RPE_frame_ST_decoder**, **RPE_frame_SLT_coder**, **RPE_frame_SLT_decoder** are implemented as described in the assignment description.

The main function to test the codec is the **encode_audio(file)** function. It takes an audio file as an argument and plays its audio after it has been coded and decoded, plots the samples before and after, and writes the output audio to **out.wav**. In the project folder, there is already a speech sample, so the project can be tested by running.

```
encode_wav('OSR_us_000_0010_8k.wav')
```

or just **demo** on the MATLAB interpreter.

3 Implementation notes

The project repository has 4 branches. The main branch contains the 3rd level assignment which follows the implementation specified in the standard as closely as possible, including the interpolation of log-area ratios. The level3_no_lar_interp branch targets the 3rd level as well, but does not interpolate the log-area ratios. The stage2 and stage2 branches isolate the assignment levels 1 and 2 respectively, and can be used to verify that the first two levels are working correctly and their coding quality.

Both level3 branches are nicely intelligible when using speech samples, even though it is easily identifiable that they are different from the original (unlike the first 2 assignment levels). I implemented the log-area ratio interpolation expecting to get even clearer results, but hardly any difference can be noticed. As mentioned in the introduction, I wanted to follow the specification very closely, so I considered any further modifications "cheating", but some of the

mid-high frequency buzzing can potentially be reduced with a low pass filter at the decoder or a more aggressive RPE weighting filter.

The implementation in the stage2 branch contains only the first two levels, so no coding/quantization of the long term analysis residuals (RPE parameters) is applied. This produces results identical to the original audibly (to my ear at least), which makes sense since we are using exactly the original residuals which were obtained using the same quantized-dequantized parameters. The majority of the compression occurs on the residuals (instead of 160 floats we only use 188 bits in each frame for the RPE parameters!). The stage1 branch, which only uses short term prediction is even closer to other original, with barely any divergence from it even when zooming to the output signal plot very closely. Again, this happens because we are using the original uncompressed short term residuals.

4 Conclusion

Overall this was an interesting assignment that produced a very efficient but still easily intelligible vocal codec. It can be used to transmit 100% of the verbal information, while only using $260/160 = 1.625$ bits per sample from the original 13 bits.