Eric Bradley
Image Compression, Packet Video and Video Processing (525.759)
Final Project Report
Digital Reconstruction of Analog Video
12/10/2007

Abstract:

The National Television System Committee (NTSC) established the first standard for black & white television transmission in 1941. This was followed by the establishment of a fully backwards compatible color standard in 1953 – what is now called the NTSC standard. All of North America and many other countries have used this standard unchanged for the last 5+ decades. NTSC's life is limited because in February 2009 all over-the-air television is FCC-mandated to be digital only. This project pays tribute to the long-lived, but dying analog standard by digitally manipulating analog NTSC data to reconstruct the TV images and sound.

Problem Statement:

Any television set contains relatively simple electronics (it is 50 years old after all) to decode the fairly complex NTSC signal. This project will demonstrate simulation of that receiver hardware in software to recreate the signal in a non-real-time manner.

Objective:

The NTSC RF signal contains independent luminance video, color video, and audio signals multiplexed together with various modulation schemes and techniques. After obtaining sampled RF NTSC data, write appropriate software to analyze the content of the signal in both time and frequency domains. The ultimate goal is to extract audio and luminance components for playback. If possible, the quality of the extracted data will be compared to the original to determine the adequacy of the software extraction.

Approach:

Sampled RF analog NTSC data was obtained from two independent sources. Using MATLAB, this data was analyzed, downconverted as appropriate, filtered, and demodulated. Then the luminance and audio sequences were reconstructed into their original content. The color signal was analyzed, but not reconstructed accurately. Each data stream reconstructed was compared to one another and the original sources (where available) in terms of quality and ease of extracting the data.

Analysis:

Data Capture

For readers not familiar with the NTSC specification, it will be explained in the following pages as the processes to decode it are described. But first the data collection systems will be highlighted. Table 1 outlines the characteristics of the two datasets. Figures 1 and 2 show the data collection systems used to capture the VCR data. There are three difference of note between data sets in terms of the end product. First is that the IF ranges are different (0-6 MHz vs. 5-11 MHz). Consequently, the sampling frequencies are different (12 MS/s vs. 32 MS/s).

Lastly, due to the test setup and inexperience of the operator, dataset 1 consists of only 2 clips of 3- and 2-second durations compared to a full minute for dataset 2.

	Data Collector	Sample Rate	Frequency Band	Video Content	Clip Length	Source Video Available	Relative Signal Quality
Dataset 1	Eric Bradley	12.5 MS/s	0-6 MHz	Aspen Ski Promo	2 seconds & 3 seconds	Yes	Good
Dataset 2	Dr. Nicholas Beser	32 MS/s	5-11 MHz	Al Jazeera Newscast	60 seconds	No	Poor

Table 1 - Dataset Specifications

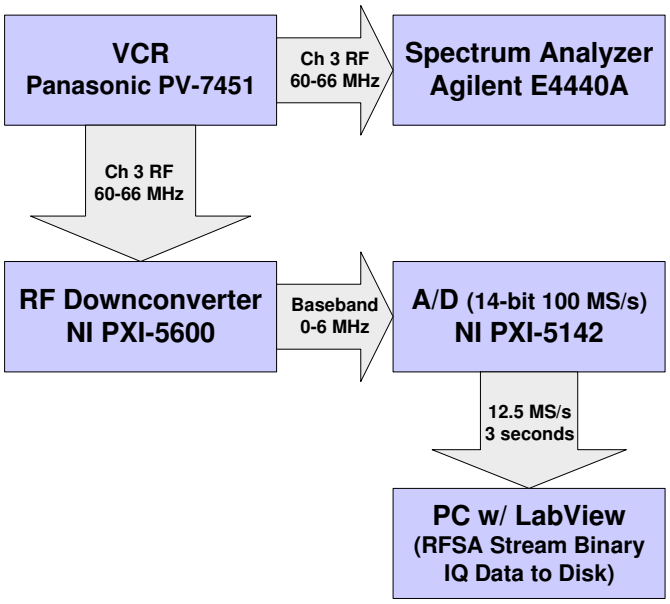

Figure 1 – Dataset 1 Data Collection System

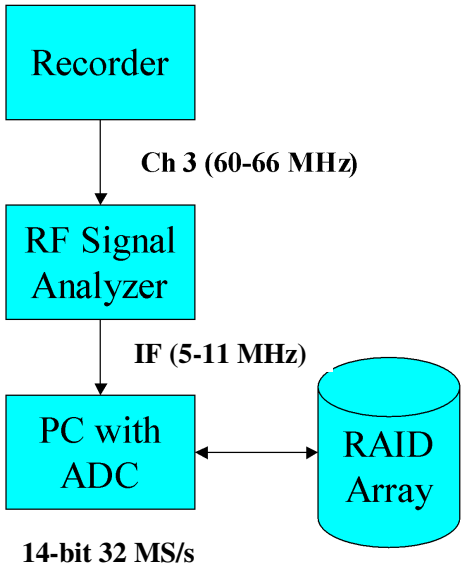

Figure 2 – Dataset 2 Data Collection System

The first step in processing the two data streams is to convert one or both datasets so that they are in the same frequency band with the same sampling period. Once this is accomplished, the processing can be essentially identical for both data sets assuming similar signal quality. The baseband (0-6MHz) is the optimal band in which to do the processing with the smallest sampling period possible to ease the processing requirements. The minimum sampling frequency, the Nyquist rate, for that data set is 12 MS/s.

Dataset 1 already exists at this frequency, so in theory it is already done. In reality, the Fourier transform of the raw data revealed that the data was downconverted an additional 1.25 MHz (for unknown reason). Therefore upconverting (to 5-11 MHz) and upsampling (to 32 MS/s) dataset 1 to look like dataset 2 was the simplest solution. From that point, both datasets can be treated as the same. Once at IF each dataset was bandpass filtered, then downconverted to the baseband, and lowpass filtered. Finally the data was resampled to the Nyquist rate to reduce the amount of data to process. Figure 3 shows the FFT of the baseband data with the obvious luminance, color, and audio carriers denoted.

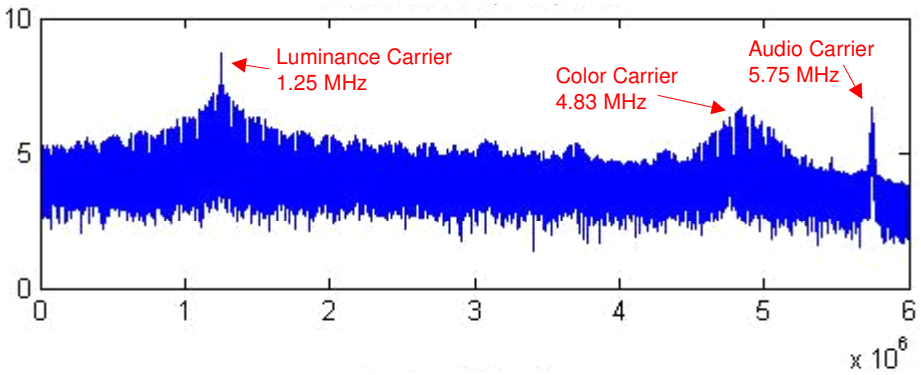

Figure 3 - Baseband Frequency Response

Audio Reconstruction

Of the three data stream components, the audio was by far the easiest to extract, and also the most uniform between data sets. The audio signal is frequency modulated at 4.5 MHz above the luminance carrier. Since FM is fairly tolerant of relatively low levels of noise, the processing was identical for each data set, even though the noise levels are much different (as we'll see shortly). Extracting the audio is simply a matter of a relatively tight band-pass filter (30 KHz passband) followed by FM demodulation; MATLAB provides functions for both. After that, the demodulated audio signal is drastically downsampled to the typical audio sampling rate of 44.1 KHz.

The quality of the audio from both data sets is reasonably good. The audio from dataset 1 is comparable to the original audio. The audio from dataset 2 is noisier than that of dataset 1, but still discernable (if you speak Arabic) and of much better relative quality than its extracted video.

The biggest challenge in processing either dataset is the sheer volume of data. At baseband frequencies every *second* of data is 12 million samples. Compound this with the overhead required to get to baseband and the overabundance of new data created for each uncompressed signal component, and this creates serious memory requirements. These constraints are met by only reading and processing one million samples at a time and then iteratively building up the entire signal. In addition, using the smallest possible data types in MATLAB (`uint8`) instead of the default `double` for image field storage saves a vast amount of memory.

Luminance Video Reconstruction

The NTSC color signal is composed of three separate components: luminance, chrominance I (orange/cyan), and chrominance Q (green/purple). The luminance signal is still being transmitted in the same format that it was with black and white only television in the 1940s (hence the backwards compatibility).

Luminance is modulated onto a 1.25 MHz carrier using vestigial sideband modulation – a form of single sideband AM where the lower sideband is only partially suppressed. Fortunately very reasonable demodulation is possible with the MATLAB `demod` function using normal single sideband demodulation. From here a low pass filter of 4.2 MHz is applied to remove any remaining high frequency content. Lastly, the waveform is normalized to values from 0 to 100 so that features can be identified and acted upon. The resulting normalized luminance waveform is shown on different time scales in Figures 4 and 5.

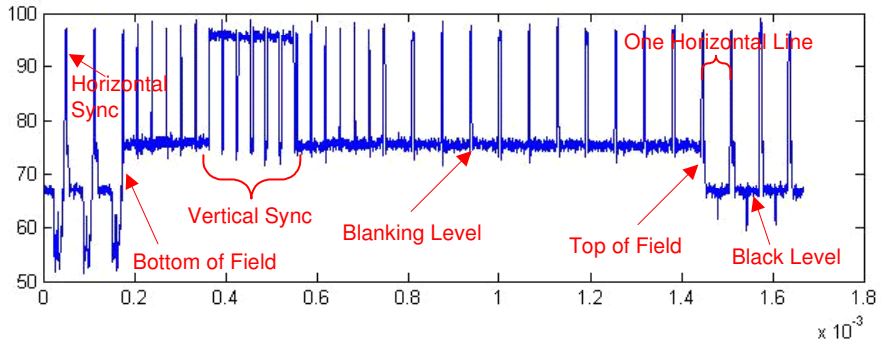

Figure 4 - Luminance Waveform - Vertical Sync

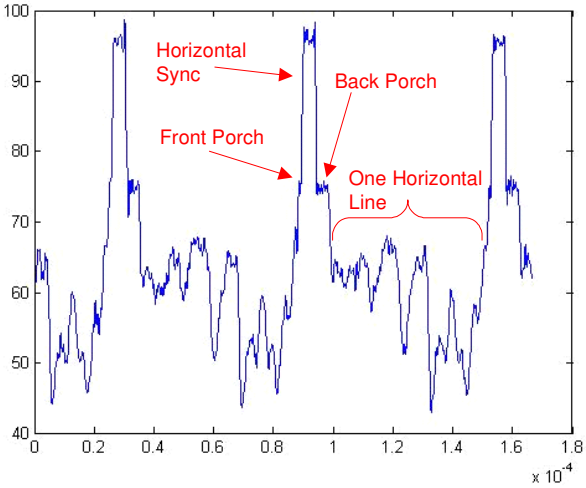

Figure 5 - Luminance Waveform - Field Line

The vertical sync pulses in Figure 4 are used to identify the start of a new field at 59.94 Hz. Each horizontal sync pulse is used to identify the start of one of the 262.5 lines per field. Each video frame is composed of two fields.

In order to properly start the capture of a field, the MATLAB code identifies and measures 6 consecutive vertical sync pulses within the predefined levels shown above. If those levels are not properly behaved or are very noisy, vertical sync detection becomes difficult or impossible. Dataset 1 has very well behaved levels through the duration of the recording – sync detection is not a problem.

Dataset 2, however, is a different story. Not only do levels vary considerably, prohibiting synchronization, but by several seconds into the video, the luminance carrier frequency appears to shift enough that demodulation is also thrown askew. Figures 6 through 8 show dataset 1's clean waveform and dataset 2 waveforms with good and poor demodulation. All are in roughly the safe timescale. Figure 7 illustrates how relative level fluctuations can throw off the vertical sync detection. Figure 8 shows how critical it is to demodulation with the exact frequency in order achieve proper results. The workaround for the inability to detect vertical syncs for dataset 2 is simply to ignore them completely and read the fields strictly based on timing. This, of course, results in a rolling image when played back. When the waveform starts to look like Figure 8, the algorithm starts to throw out many lines per field.

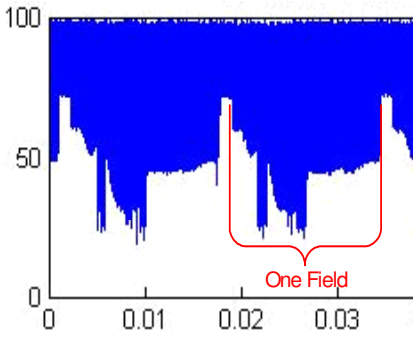

Figure 6 - Dataset 1 Luminance

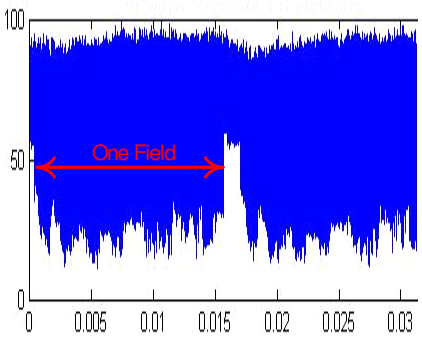

Figure 7 - Dataset 2 Luminance Proper Demod

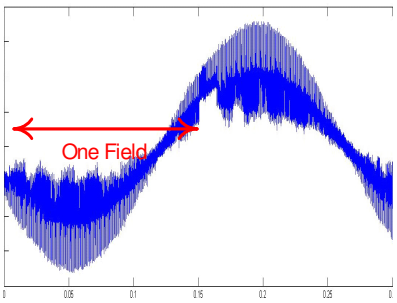

Figure 8 - Dataset 2 Luminance Carrier Shifted ~30 Hz

After a new field has been identified from its vertical sync, the algorithm begins to look for horizontal syncs to find the first line. Fortunately horizontal sync detection is more tolerant than vertical sync detection. As shown in Figure 4, the horizontal sync has a larger delta height. If the algorithm does happen to miss a horizontal sync, the image is only missing a line, which is less noticeable than the rolling frame effect that results from missing vertical syncs. Once the sync has been identified, the back porch is found, and the line is read in. Finding the end of the back porch is tricky with noisy data (dataset 2). In noisy data, a line can be slightly offset horizontally creating a noisy, but still readable image.

Each line that is read in must be scaled and normalized to 256-byte grayscale. Once 262 lines have (the extra ½ is discarded) been read in, it is stored off as a field. This is repeated once, and the two fields are interlaced together to form a frame. Interlacing generally did not work well for dataset 2 due to the lack of vertical synchronization. Repeat the entire process until the end of the data stream to build up all frames in the video.

Examples of the resulting frames are shown in Figures 9 through 11. Compare the a typical extracted frame from dataset 1 in Figure 9 to its original frame (captured via commercial frame capture hardware) in Figure 10 and note the similarities. The main difference between the two is the existence of high frequency noise in the extracted frame. This is due to the computationally cheap lowpass filter that was used in place of a more complicated comb or trap filter to remove the color harmonics after demodulation. Figure 11 shows one of the highest quality frames from dataset 2. Most frames in dataset 2 exhibit poor interlacing (due to vsync), and dropped lines (due to carrier drift).

Figure 9 - Digitally Reconstructed Frame from Dataset 1

Figure 10 - Original Frame from Dataset 1

Figure 11 - Digitally Reconstructed Frame from Dataset 2

Color Video Reconstruction

As mentioned above, the color component is actually two components – chrominance I and Q. These signals are modulated together using QAM at ~3.58 MHz (above luminance carrier). But before demodulating, the signal must first be bandpass filtered with a passband of 2.1 to 4.1 MHz. After filtering, QAM demodulation is applied to retrieve the 90° phase-offset signals. Chrominance I is then lowpass filtered by 1.5 MHz. Chrominance Q is less perceivable by the human eye, so is lowpass filtered even further by 0.5 MHz. After filtering, the I & Q signals resemble luminance in timing minus the sync pulses. Luminance sync pulses are used to find the timing in the chrominance signals. Finally the YIQ triplet is converted into RGB for digital display and storage.

The crux, however, in reconstructing color video is capturing and synchronizing to the color burst signal that is present in the back porch of the luminance signal. This color burst provides a phase reference so that QAM demodulation will return consistent phase results and ultimately consistent color. Unfortunately in both datasets, the color burst signal is mysteriously absent, but that did not prevent an attempt at reconstructing the color video in dataset 1 only! Without the color burst one would expect the relative color to be changing constantly. From Figure 12, this is clearly the case. This is a sequence of 5 fields output while the VCR was not played a video; in other words, just a constant blue screen. The levels are constantly changing due to lack of synchronization.

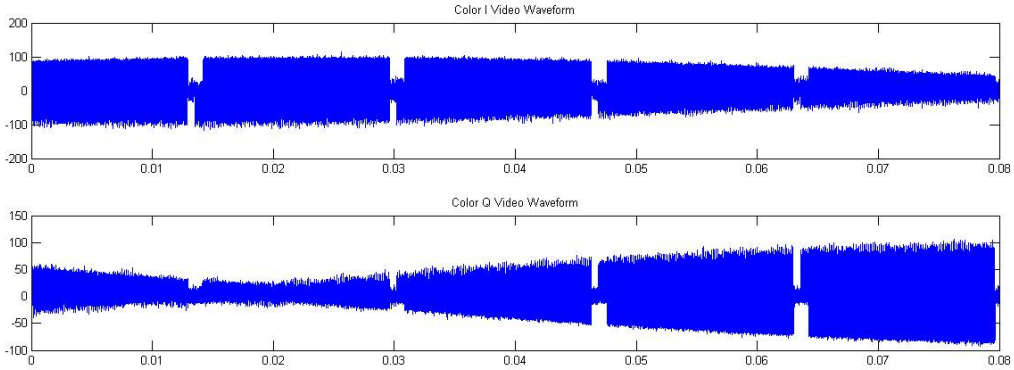

Figure 12 - I & Q Unsynchronized without Color Burst

Of course, occasionally the proper colors will be displayed as the levels sweep back and forth. Figure 13 shows the reconstruction from dataset 1 in one of these cases. Compare this to the original image in Figure 14. Not to get ones hopes up, however, because Figure 15 shows what most frames look like. This is in fact the very next reconstructed frame.

Figure 13 - Reconstructed Proper Color Image

Figure 14 - Original Color Image

Figure 15 - Reconstructed Typical Color Image

Possible Algorithm Improvements

Several possibilities exist to improve upon the results obtained. The first is most obvious from the previous section. Find the color burst and use it to synchronize the color subcarrier. Color video is fairly worthless and actually detracting without it – truly Never The Same Color!

To put the finishing touches on the already decent data reconstruction of clean data (e.g. dataset 1), addition filtering should be added. Ideally interlaced luminance and color comb filters should be used to remove the harmonics of the other. An easier implementation would be a 3.58 MHz trap filter on the demodulated, filtered luminance data. That should remove most of the high frequency noise in the picture.

The vertical sync detector and normalizer in the current implementation are relatively vulnerable to relative changes in amplitude. Improving its robustness would solve some of the rolling video issues in the reconstruction of dataset 2.

Lastly, dataset 2 has a nasty issue of a roving carrier frequency. The reconstructed video clip reveals that it may only change once at around 6 seconds. If this were the case, the new carrier frequency could be manually determined through trial and error. FFT analysis by itself did not provide for enough resolution – the change in carrier frequency is probably around only 30 Hz. A more elegant and significantly more complex solution is to somehow accurately measure the peak frequency for each iteration and adjust the demodulation frequency accordingly. Either of these solutions will hopeful eliminate the unreadable signal seen in Figure 8, and return data resembling Figures 6 or 7.

Performance

The algorithms described above are very processor and memory intensive. It can take up to 1000 times longer to reconstruct the data digitally using MATLAB than the length of the actual clip reconstructed. For example it took about an hour to reconstruct the 3 seconds of audio and video from dataset 1. Much of that time was spent looking for sync pulses. Consequently reconstructing dataset 2 (which ignores vertical syncs) isn't as time consuming, taking "only" 285 minutes to reconstruct 60 full seconds of audio and video.

Conclusion:

Extracting luminance and audio from clean data was a challenge, but the end result is very promising. As the data quality starts to degrade, however, the benefits of actual hardware become very apparent. Reconstructing color accurately and consistently would also prove difficult, but certainly not impossible in software.

The NTSC standard is truly an amazing feat of engineering – adding respectable color data in a fixed bandwidth without significantly corrupting the existing luminance data. Besides that, its longevity alone speaks volumes. It has served us well.

Supporting Software:

MATLAB code, original data, original video clips (w/ audio), reconstructed video clips (w/ audio), reconstructed audio, project presentation, and this project report can be found online at:

http://ericbradley.net/imagecompproj/

References:

Beser, N. 525.759 Image Compression and Packet Video. Class Notes. "Lecture 1 – Introduction to Video." Slides 44-69.

Carlson, A.B., Crilly, P.B, and Rutledge J.C. *Communication Systems: An Introduction to Signals and Noise in Electrical Communications.* 2002. McGraw Hill. Pgs 170-172, 286-299.

Jack, K. *Video Demystified.* CD-ROM Supplemental. LLH Technology Publishing.

Van Valkenburg, M.E. *Reference Data for Engineers: Radio, Electronics, Computer & Communications.* 2001. Newnes. Pgs 35-12 – 35-20.

Wang Y., Ostermann J., and Zhang Y. *Video Processing and Communications.* 2002. Prentice Hall. Pgs 12-22.

http://en.wikipedia.org/wiki/NTSC