SSTV Transmission Methodology

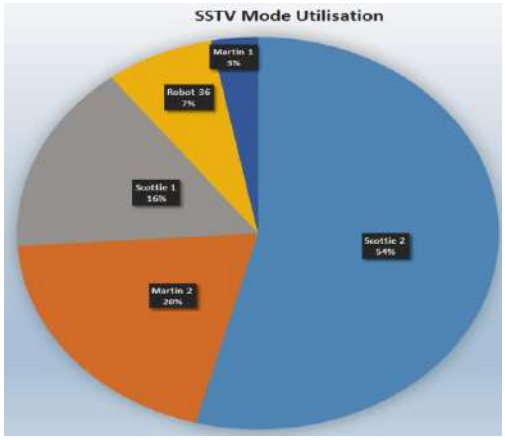
Slow Scan TV (SSTV) is a video mode which uses analog frequency modulation. Every different brightness in the image is assigned a different audio frequency.

The modulating frequency ranges from 1500 to 2300 Hz, corresponding to the brightness of the color component. The frequency of video signal varies from black by shades of gray to white.

Color is achieved by sending the brightness of each RGB color component separately. The original image is scanned from left to right, in green, blue and red order for Martin and Scottie modes. To help simplify this complex article, we will focus only on these two types of SSTV modes.

I recently conducted a limited study of 285 continuous images received at WB9KMW on 14.230 MHz during November 24-26, 2014 and determined from this sample the distribution of SSTV modes. 89 percent were in Scottie 1 or 2 and Martin 1 or 2 modes.

- Scottie 2: 52%
- Martin 2: 19%
- Scottie 1: 15%
- Robot 36: 7%
- Martin 1: 3%



This article about SSTV Transmission Methodology is written for personal edification. I have examined numerous Internet sources on the subject. I have completed considerable fundamental research. And I have been aided by ideas from Laurel Eihusen KD9AJT, Gary Peach G7SLL and Martin Bruchanov OK2MNM.

General SSTV Signal Processing

This is a process of taking a digital image, converting it to an audio frequency via a sound card, often in a ham's computer, and modulating an RF transmitter. In the United States, the FCC has authorized several emission codes for SSTV. Most commonly, hams use J3F. J for SSB modulation, 3 for one channel with audio information, and F for television signal. For FM transmissions, the emission code is F3F. For AM, it is A3F.

This modulated signal is transmitted at Radio Frequency via an antenna, often by sky way, sometimes by ground wave and sometimes by line-of-site.

At the receiving end the RF signal is detected and converted by the receiver to audio frequency modulation. That sound is sent to a sound card where it is converted back to a digital signal, displayed on a computer screen and recorded as a JPG or BMP file, depending on how the ham has configured his software. So we have a digital to frequency to digital process.

Opportunities for Analog Signal Degradation

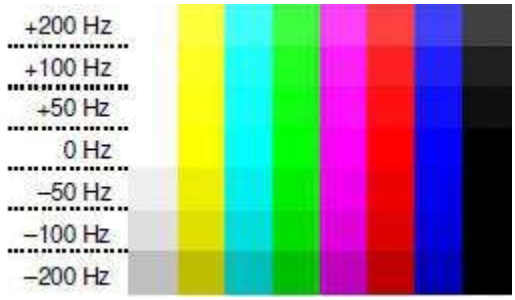
Degradation of the image occurs in many locations along this path. Here are some of the opportunities for error:

- MMSSTV software takes the original picture and converts it before sending it to the soundcard. That converted picture may be viewed if Internal Loop Back has been configured in the MMSSTV software.

- Distortions can occur in the soundcard on both ends. Many hams do not properly align their soundcard for Rx and Tx, so slanted images are transmitted and/or received.

- Problems arise in modulation and demodulation of RF signals since this is not a completely linear process. Transmitters can be overdriven with too much microphone gain or by having the speech processor engaged.

- The transmitter and receiver may not be on the exact same frequency, and one or more may exhibit frequency drift during the process.

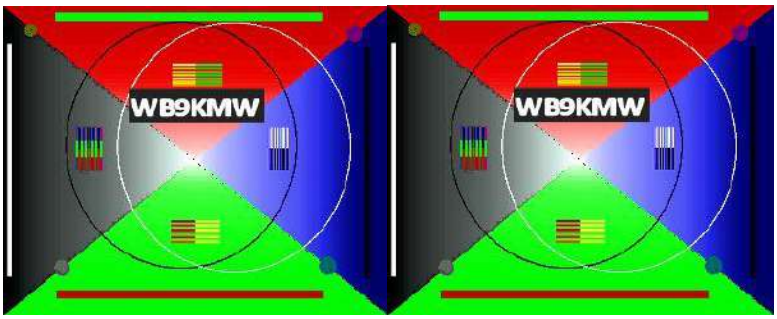


- Path loss creates a situation where the signal-to-noise ratio (SNR) leads to reception error, and this noise problem is compounded when man-made noise is introduced, such as QRM from other SSTV transmissions or SSB voice transmissions on frequency or on a nearby frequency that is not eliminated by the receiver.

- Each time that a received picture is stored on a computer, there may be further degradation of the image. Care should, therefore, be taken when repeatedly re-storing the image if it is to be used for forensic analysis. The way one stores the picture is critical.

For example, this picture was saved and then the copy saved with the Adobe Photoshop 'Save As' process for 50 generations of duplicates, without any editing of the original image. I studied for a precise pixel by pixel match with ImageMagick. The 50th copy matches only 17.27% of the original (on left).

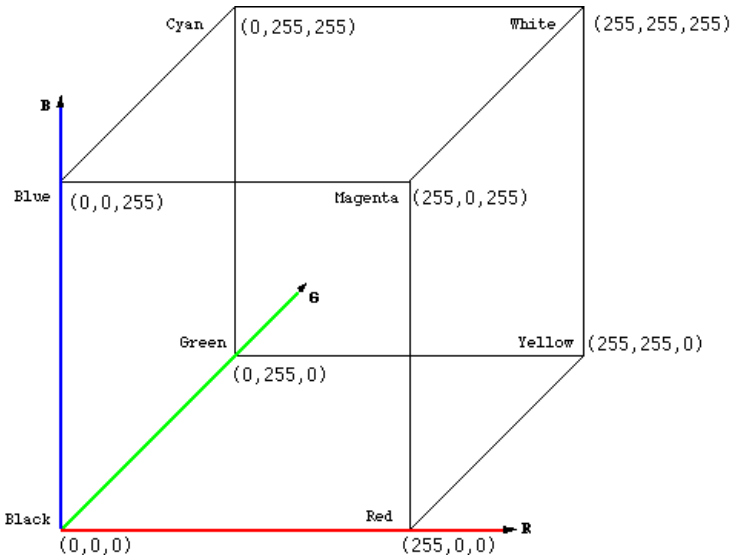
However, if one simply copies an original and pastes it in a computer directory, and does this over and over for many generations of recopies of the copy, there is no quality degradation. ImageMagick returns a 100.00% perfect match of the 40th copy against the original (on left which is an SSTV test pattern created for me by Gary Peach G7SLL).



Vertical Sync

Every mode uses a digital header for identification. The rest of the transmission is analog and is accomplished by frequency modulation to convey colors in RGB space (for Martin & Scottie modes) according to various levels of color brightness between 1500 and 2300 Hz. Because it is analog, it cannot transfer images without loss.

It is this RGB in space information that needs to be converted to an audio signal for transmission.



The SSTV transmission begins with a calibration header. This starts with a 1900 Hz tone for 300 ms, a 10 ms break at 1200 Hz, and another 1900 Hz tone for 300 ms. Next comes the vertical synchronization information. Once finished, the image scan can begin.

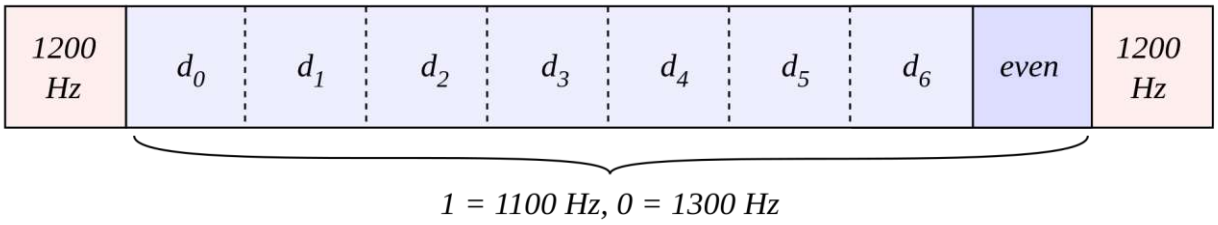
Robot Research developed VIS (Vertical Interval Signaling). VIS is a digital code. The first and last bits are start and stop indicators at 1200 Hz. The remaining eight bits identify the mode. It takes 300 ms to transmit these ten bits (30 ms each). Immediately after the last bit (stop bit), the horizontal sync tone is sent at 1200 Hz for Martin mode (see next section for horizontal sync).

Each bit is 30 ms and a frequency of 1300 Hz expresses a logical state of 'zero.' A logical state of 'one' is at 1100 Hz.

The last bit of eight is a parity bit for error checking, using even parity. If the sum of 'one' bits is odd, the parity bit is set to 'one,' otherwise 'zero' for even.

So we have:

- Calibration header: 610 ms
- VIS code: 300 ms

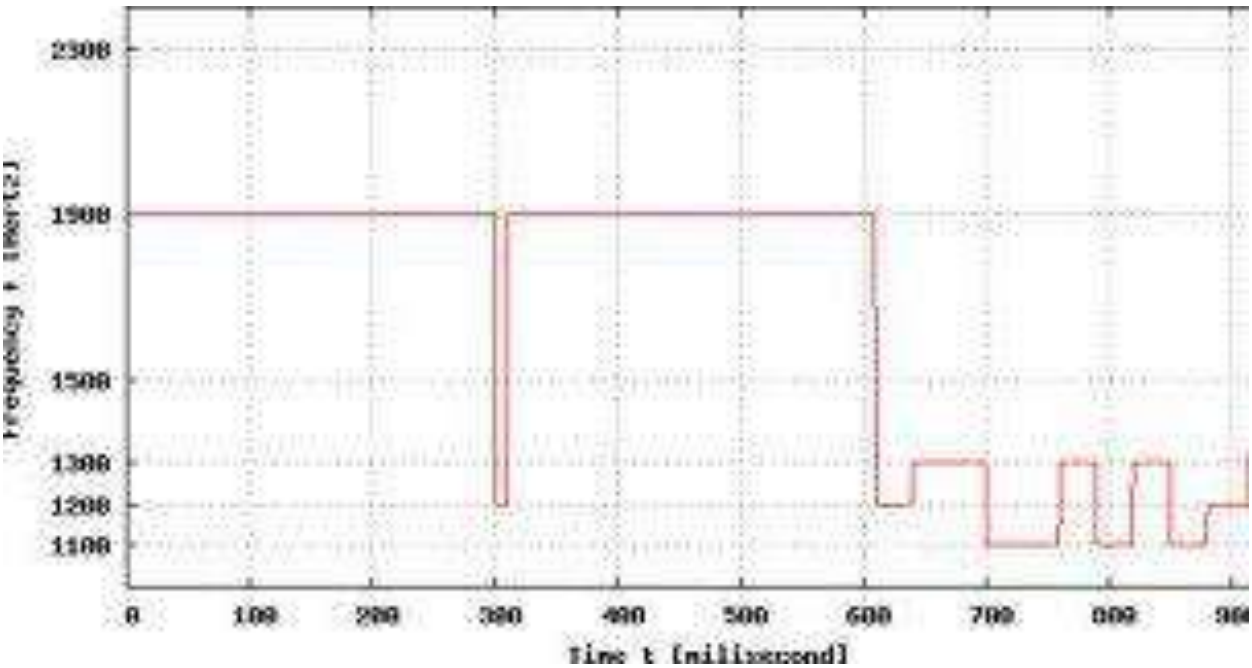


Here are the VIS codes for Martin and Scottie modes. The data is arranged from the least significant bits to the most significant bits when transmitted.

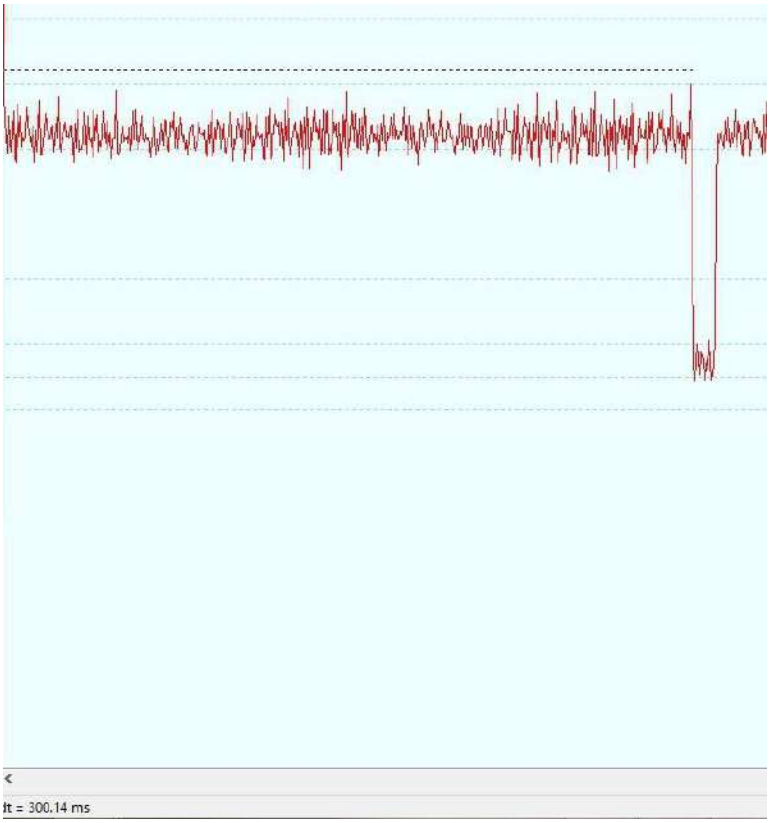
	d6	d5	d4	d3	d2	d1	d0
Martin 1	0	1	0	1	1	0	0
Martin 2	0	1	0	1	0	0	0
Scottie 1	0	1	1	1	1	0	0
Scottie 2	0	1	1	1	0	0	0

d0/d1 indicate color composite video for Martin & Scottie modes
d2 denotes horizontal resolution: 0 = 160 px and 1 = 320 px
d3 is the vertical resolution: 0 = 128 lines and 1 = 256 lines
d4 to d6 collectively provide the added binary coding for the mode

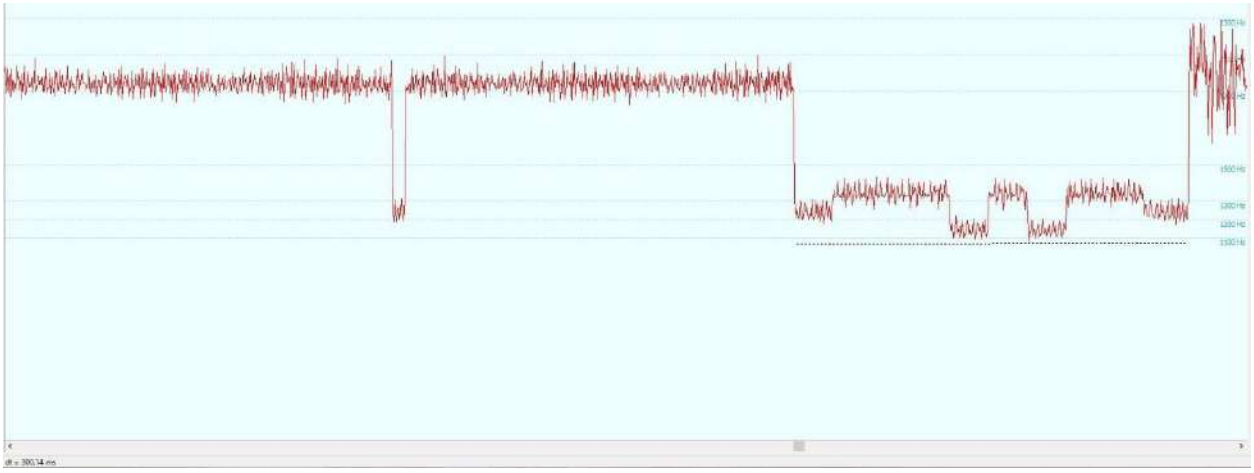
Here is an example of the calibration header at 1900 Hz/1200 Hz/1900 Hz, followed by the 1200 Hz start bit, then d0 to d6 bits (0011010), the parity bit of 1, and a 1200 Hz stop bit for Martin 1.



I used my 10m beacon to transmit a signal and record in mmv audio format with MMSSTV. Then I analyzed time elapsed frequency patterns with the SSTV Signal Viewer and captured these snippets to illustrate a real SSTV signal. This is the first of two 300 ms 1900 Hz signals from the Calibration Header, with a single 1200 Hz break for 10 ms. The dotted line measures the approximate duration of this 300 ms signal.



In this picture, we see the 300 ms of VIS code, in digital format. It starts with a 1200 Hz start bit, then the d0 through d6 bits for Martin 2 (0001010) at 1300 and 1100 Hz, followed by the parity bit (0 for an even sum of 1's), and concludes with another 1200 Hz tone for the stop indicator.



G-B-R Horizontal Processing for Martin & R-G-B for Scottie Modes

Martin Emmerson G3OQD developed the Martin mode. Eddie Murphy GM3SBC developed the Scottie mode. They have much in common. Both use the RGB color format.

A most important feature of these newer SSTV modes is the use of only one line sync pulse at the start of each color sequence. This makes it impossible for the receiving scan convertor to get confused as to which color is being tranmitted, as only one line sync pulse is sent for each color line. The time intervals where line sync pulses are no longer transmitted are filled with reference black level at 1500 Hz.

There is a single horizontal sync tone sent for Martin modes before each scan line begins. This lasts for 4.862 ms at 1200 Hz. Then the color components for that horizontal line are sent in this sequence: **green**, **blue** and lastly **red**. Between each color component, there is a short gap in which a 1500 Hz black reference level signal is sent for 0.572 ms.

The scan-line composition and scan timing are different for Scottie modes. Following vertical sync, the scan lines appear in this order: a 1.5 ms gap at 1500 Hz, then the **red** component, a 1.5 ms gap at 1500 Hz, then the **green** component, then a horizontal snyc tone at 1200 Hz, another 1.5 ms gap at 1500 Hz and finally, the **blue** component. In addition, for the first line only, a 1200 Hz starting sync pulse begins the image transmission.

The specifics of the Martin and Scottie transmission modes are provided by Martin Bruchanov OK2MNM.

	Tx	Color			Scan line time (ms)			
Mode	Time (s)	Resolution	Sequence	Sync	G/R*	B/G*	R/B*	Speed (l/m)
Martin 1	114	320x256	G-B-R	4.862	146.432	146.432	146.432	134.3947532
Martin 2	58	160x256	G-B-R	4.862	73.216	73.216	73.216	264.5525975
Scottie 1	110	320x256	G-B-R	9.0	138.240	138.240	138.240	140.1148942
Scottie 2	71	160x256	G-B-R	9.0	88.064	88.064	88.064	216.0667214

*G-B-R sequence for Martin modes and R-B-G sequence for Scottie modes

For example, with Martin 1, the following occurs:

- Horizontal sync for 4.862 ms
- Black reference: 0.572 ms
- Green component: 146.432 ms
- Black reference: 0.572 ms
- Blue component: 146.432 ms
- Black reference: 0.572 ms
- Red component: 146.432 ms
- Black reference: 0.572 ms

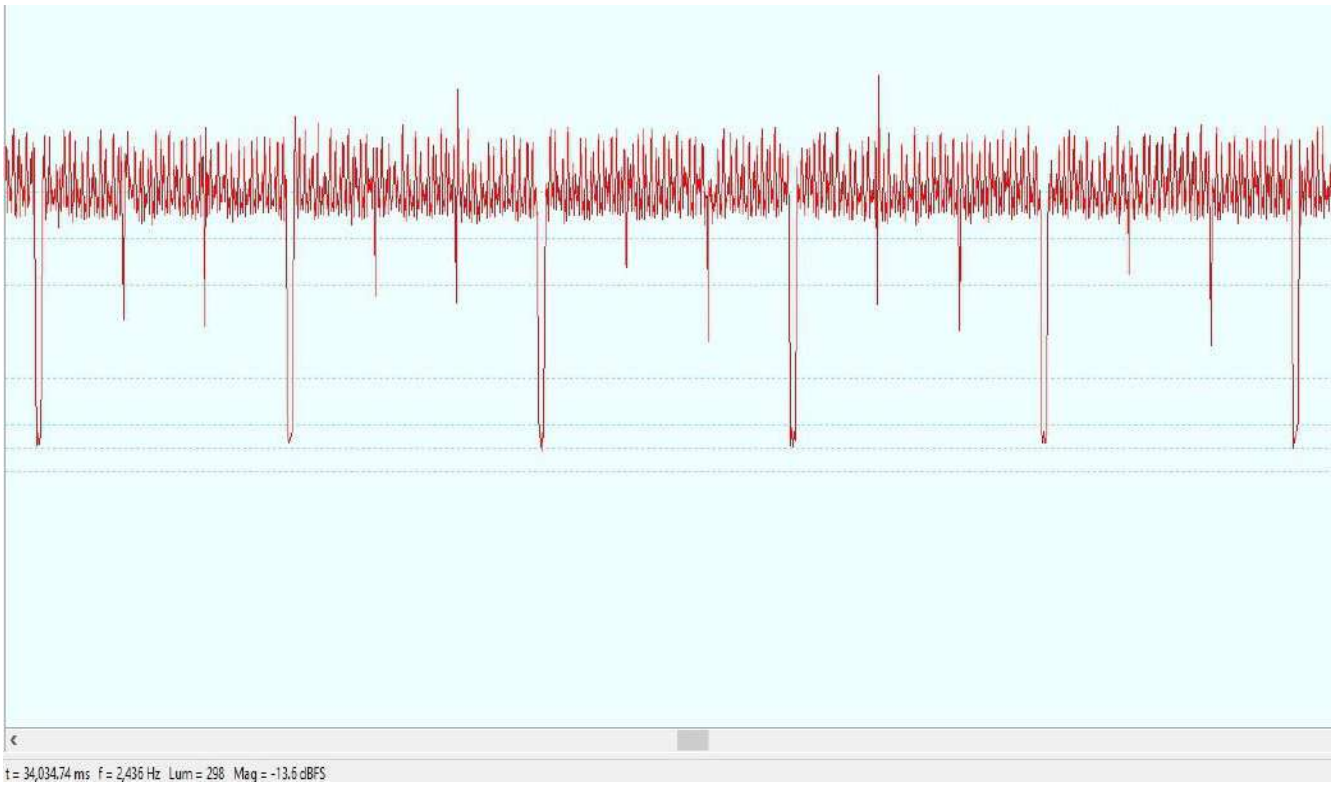
This takes 446.446 ms per line. There are 256 lines, for a total elapsed time of 114290.176 milliseconds, or approximately **114** seconds. If the line processing time is divided into 60000 ms, the number of milliseconds in one minute, we get the line speed of 134.3947532 lines/minute.

Now let's carefully examine what happens with Scottie 1.

- Black reference tone for 1.5 ms
- Red component: 138.240 ms
- Black reference tone for 1.5 ms
- Green component: 138.240 ms
- Horizontal sync for 9.0 ms
- Black reference tone for 1.5 ms
- Blue component: 138.240 ms

This takes 428.22 ms per line. There are 256 lines, for a total elapsed time of 109624.32 milliseconds, or approximately **110** seconds. If the line processing time is divided into 60000 ms, the number of milliseconds in one minute, we get the line speed of 140.1148942 lines/minute.

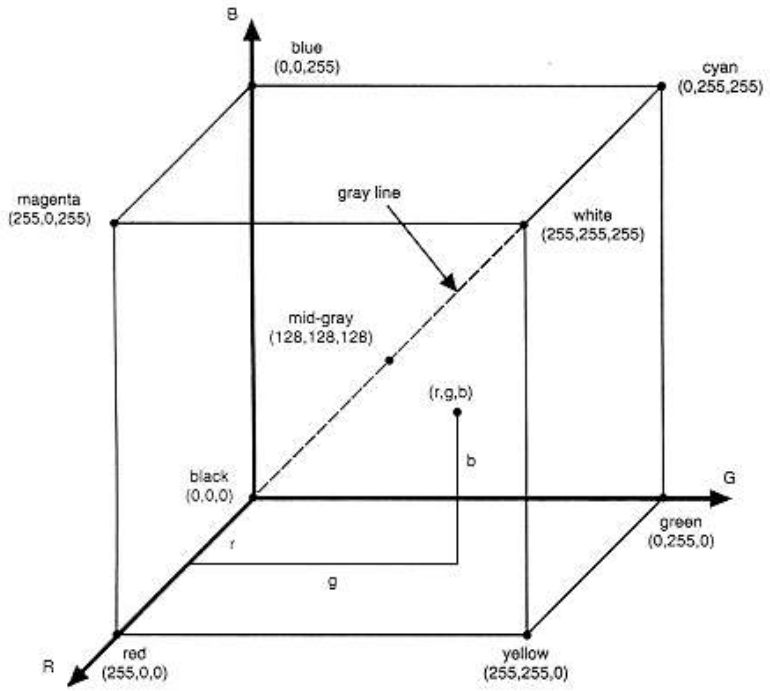
Here is the 1200 Hz horizontal sync pulse that begins each new row of pixels in Martin 2, and lasts for 4.862 ms. These are the six lowest lying frequency traces for these five rows of transmission.



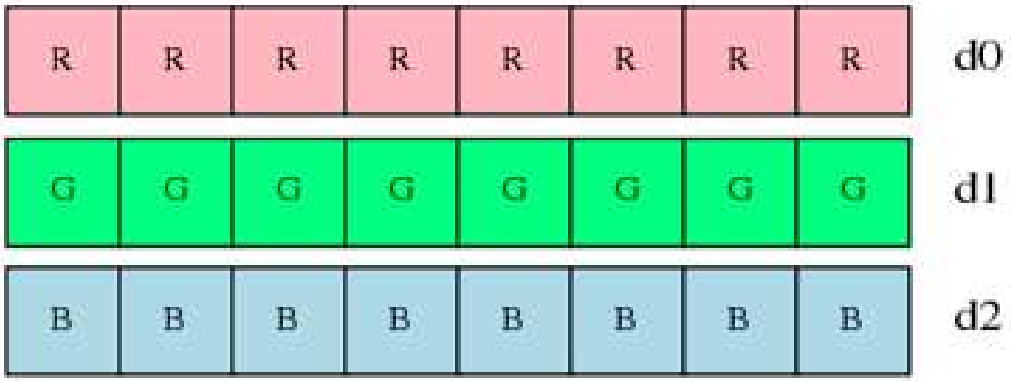
The green, blue and red components of RGB image space are then transmitted, row by row. Here is the first row of information encoded in audio frequency to denote brightness of color. I used a near-white color so that the black reference markers at 1500 Hz could be easily observed. They last for only 0.572 ms, and the mmv audio file doesn't quite capture their actual frequency of transmission. The color segments are shown here in order: green, blue and finally, red.

RGB Frequency Encoding

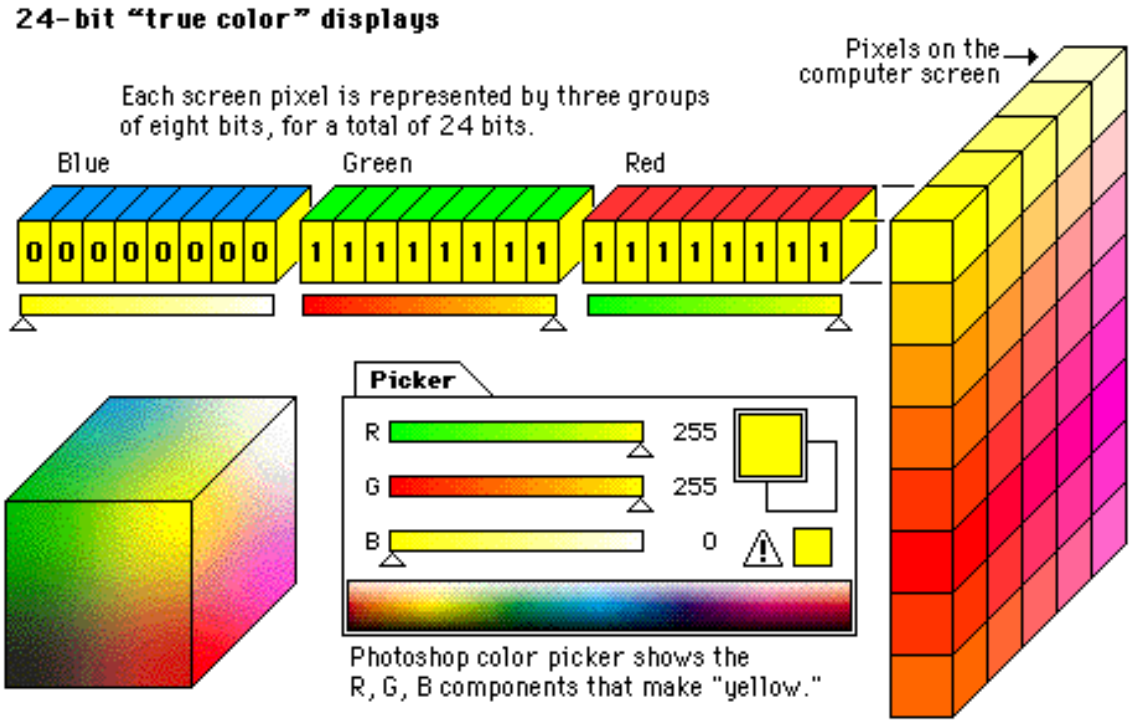
RGB color space constructs all the colors from a combination of red, green and blue colors. The red, green and blue uses eight bits each, which have integer values from 0 to 255. This makes 256x256x256 = 16,777,216 possible colors!



The RGB components must be unpacked from their 24-bit entity in 8-bit (0-255) color clusters.



When this is done and converted to a frequency the information can then be transmitted intelligently starting with position 0,0 for the first row and first column. Remember that for SSTV, the first row entries are greens, followed by blues for that same row and ending with the reds for each pixel position in the row.



Let's look at a complex color, azure.

The RGB color decimal values are:

RedByte = 51
GreenByte = 102
BlueByte = 153

The pixel value is given by the formula, RGB = (R x 65,536) + (G x 256) + B

It has a pixel value of 51 x 65,536 + 102 x 256 + 153 = 3,368,601

The hexadecimal value is 336699.

SSTV uses a frequency range of 1500 to 2300 Hz to represent the range of brightness values from pure black to pure white. Once the decimal value has been determined for each ColorByte, it may be inserted into this formula to determine which audio frequency to transmit.

Frequency = 1500 + (ColorByte x 3.1372549)

Notice that 800 Hz (ie, 2300 - 1500 Hz) divided by 255 = 3.1372549

So for the azure color, the frequency tones would be:

Green row:

1500 + (102 x 3.1372549) = 1,820 Hz

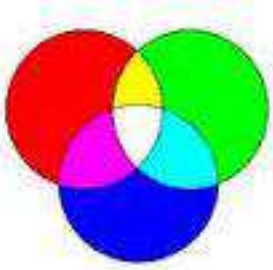
Blue row:

1500 + (153 x 3.1372549) = 1,980 Hz

Red row:

1500 + (51 x 3.1372549) = 1,660 Hz

At the receiving end this information is recombined through a decoding process and then delivered to the ham operator's computer screen in pixel form. Here is an example of how one pixel combines the red, green and blue to display the color for that screen location.



If you instead know the transmitted audio frequency, you may then calculate the ColorByte for the given row of pixels:

ColorByte = (Frequency – 1500)/3.1372549

Larry WB9KMW @ WB9KMW.com