Synesthesia & FSO: Using frequency encoding to visually represent and communicate audio

Yvette Gerber, Hayden Somach

I. INTRODUCTION

The extension of our free-space optical (FSO) communication system introduces a function that maps the characteristics of an audio file into a visual image. This function is invertible, allowing the original audio to be reconstructed from the visual representation. In earlier versions of the FSO system, information was encoded by modulating amplitude, which enabled accurate data reconstruction but produced visual representations that bore no perceptual resemblance to the original audio signal. To create a stronger visual–auditory correspondence, we transitioned from amplitude-based encoding to frequency-based encoding and constrained the system's input to audio files.

II. METHODS

A. Hardware Setup

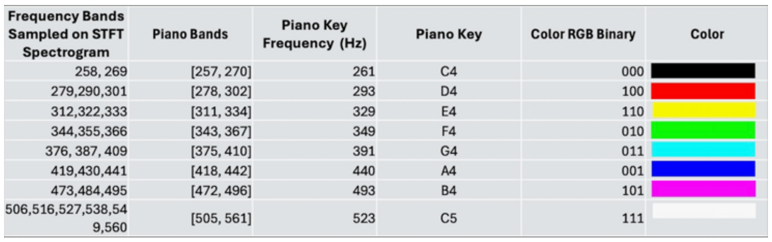
The setup can be split into a transmitter and a receiver. A desktop display monitor was used as our transmitter, and the camera recorded the data on the display.

- Camera: Teledyne FLIR Blackfly S USB 3.1 Color Camera (BFS-U3-16S2C-CS)
- Display: Dell Flat Panel Monitor P2422H

B. Algorithm

Input: The input is restricted to a .wav audio file containing a combination of piano notes ranging from C4 to C5.

Encoder: The encoder first reads the audio file and stores it in memory. Because the system uses frequency encoding rather than amplitude encoding, a spectrogram is generated from the audio signal. The spectrogram computes the frequency content of the audio within each time interval, making it well-suited for capturing musical notes. A mask is then applied to the frequency array to isolate frequencies between the lowest and highest piano keys used— 261 Hz (C4) and 523 Hz (C5), respectively. As shown in Fig. 1, the masked frequency band is subsequently divided into segments corresponding to the frequency ranges of individual notes.



Frequency Bands Sampled on STFT Spectrogram	Piano Bands	Piano Key Frequency (Hz)	Piano Key	Color RGB Binary	Color
258, 269	[257, 270]	261	C4	000	⬛
279,290,301	[278, 302]	293	D4	100	🟥
312,322,333	[311, 334]	329	E4	110	🟨
344,355,366	[343, 367]	349	F4	010	🟩
376, 387, 409	[375, 410]	391	G4	011	🟦
419,430,441	[418, 442]	440	A4	001	🟦
473,484,495	[472, 496]	493	B4	101	🟪
506,516,527,538,549,560	[505, 561]	523	C5	111	⬜

Fig.1: The image shows the process of taking piano keys and turning them into colors for the encoder.

Before encoding, a start word and an end word are appended to the beginning and end of the signal, respectively. Both words follow the color sequence Red–Green–Blue, shown in Fig. 7. These sequences allow the decoder to identify the boundaries of the transmitted data. To minimize false positives, black padding is added outside the start and end words. The resulting color triplets are then converted into RGB values and represented as colored squares, which are arranged row by row in a 20 × 20 grid. A video of the grid sequence is displayed on the desktop monitor at 50 frames per second (FPS).

Decoder: The decoder records a video of the transmitted signal displayed on the monitor. It reconstructs the encoded data by compensating for three main sources of error: color intensity, temporal drift, and spatial alignment.

Spatial Alignment: After the video is recorded, the encoded data is extracted by placing an ROI over each square and measuring its intensity. Before ROI analysis, the camera is aligned with the display to ensure accurate spatial correspondence. As shown in Fig. 6 in the appendix, once the video is captured, the camera's tilt angle relative to the display's vertical axis is determined by overlaying a reference square on the recorded frame. The tilt is corrected, and the video is manually cropped to isolate the region containing the encoded data.

Color Intensity: To ensure that the colors captured by the camera accurately represent those displayed on the monitor, a crosstalk matrix is applied to transform the recorded camera intensities back into their corresponding display values. The calibration data used to generate this matrix are derived from the known color values of the start word.

Module 1

$$c = \begin{bmatrix} c_{RR} & c_{RG} & c_{RB} \\ c_{GR} & c_{GG} & c_{GB} \\ c_{BR} & c_{BG} & c_{BB} \end{bmatrix}; \qquad \begin{bmatrix} m_R \\ m_G \\ m_B \end{bmatrix} = c^{-1} * \begin{bmatrix} s_R \\ s_G \\ s_B \end{bmatrix}$$

Eq. 1: C represents the crosstalk matrix. m is the displayed values vector. s is the sampled vector.

Additionally, the algorithm converts the measured intensities back into bits using an adaptive threshold. This variable threshold is determined by taking the midpoint between the maximum intensity of the clock signal and the minimum intensity of the clock signal over the entire video.

Temporal Alignment: To ensure sufficient sampling for accurate signal reconstruction, the camera recorded at 150 FPS, providing a 3:1 frame rate ratio between the camera and the display (50 FPS). To compensate for frame rate drift that may occur during longer recordings, the algorithm detects the rising and falling edges of a clock signal that alternates between black and white with each frame change on the display. An array is then generated to record the frame indices corresponding to each detected edge, allowing the algorithm to align and measure intensities precisely between transitions.

$$m_i = floor(\lfloor (c_i + c_{i+1})/2 \rfloor), \quad i = 1,2,\ldots,N-1$$

Eq. 2: The equations used to find the frames in between transition frames.

Output: The output of the system is a reconstruction of the initial audio and a spectrogram of the audio. The audio is found by inverting the previous frequency to bit mapping, storing an array of the frequencies played. These frequencies are converted to a stream of bits to play out loud.

C. Methods to improve bandwidth and accuracy

Spatial Alignment: To increase the bandwidth of the algorithm, a test was developed to find the number of ROIs per frame that would yield the highest bandwidth without sacrificing accuracy. Starting from an ROI of 2 x 2, keeping the width and height equivalent, we incrementally increased the dimensions and observed the ROI placed within an image of the same number of squares.

Color Intensity: To ensure accuracy within our algorithm, the decoded colors must align with the encoded colors. Two quantitative measurements ensure high color accuracy: a high dynamic range, and maximum and minimum intensities at around 0 and 255, respectively. The following four scenarios were considered.

1. The raw display intensities (without the camera)
2. The camera intensities without the crosstalk correction
3. The camera intensities with the crosstalk correction matrix
4. The camera intensities with the matrix with higher gain (high enough gain such that colors appear saturated)

Temporal Alignment: A major change from the previous algorithm was the implementation of a clock. Our previous algorithms averaged frames and used the FPS of the camera display to mitigate transition frames. A shortcoming was the inability to deal with time drift. To compare the accuracy of the methods, we calculated the bit error rate with the averaging algorithm and the clock algorithm, keeping the length of the signal consistent.

D. Methods to characterize audio-visual mapping

The following methods were designed to evaluate the performance and robustness of our audio–visual mapping function as a communication scheme. To explicitly assess robustness, we systematically varied input parameters and measured the resulting spectrogram correlation between the encoded and decoded signals.

First, we documented the encoding and decoding of an ascending C4–C5 octave played at 1 beat per second (BPS) for 8 s to verify that the core components of the system behaved as intended. The input was a .wav file generated by ChatGPT to satisfy the system's input constraints. To quantify the bit reduction introduced by our encoding design, we compared the number of bits in the original audio file to the number of bits represented in the encoded visual signal.

Within the encoder, we plotted both the spectrogram and the frequency corresponding to peak power at each sampled time bin. The encoded signal was rendered as a video and passed directly into the decoder. In the decoder, we plotted the decoded amplitudes associated with each bit triplet and verified correct sampling by recording the X and Y bounds of each frequency band when simplifying the 800-length frequency array to an 8-element array. Finally, we generated and compared the spectrograms of the encoded and decoded signals to compute their cross-correlation.

Test 1 — Tempo Tolerance (Without Camera): We used ChatGPT to generate an 8-second baseline .wav melody that ascended and descended across the C4–C5 octave, and produced additional versions at 2, 4, 10, and 50 BPS. For each version, we computed the spectrograms of the input and reconstructed signals and recorded the correlation coefficient between them to quantify reconstruction accuracy as a function of tempo.

Test 2 — Tempo and Duration Tolerance (Without Camera): To examine how tempo and signal duration jointly affect temporal accuracy, we generated two .wav audio files.

1. A C4–C5 melody at 2 BPS with durations of 8 s and 16 s.

Module 1

2. An ascending C4–C5 octave sweep at 1 BPS, also with durations of 8 s and 16 s.

These audio patterns, previously validated in Test 1, ensured that content did not act as a confounding variable. Each file was processed through the same encode-to-decode pipeline. We plotted spectrograms of the encoded and decoded signals and evaluated temporal reconstruction by comparing the expected and reconstructed durations and alignments.

Test 3 — Temporal Alignment (With Camera): Finally, to assess temporal alignment when using optical transmission, we repeated Test 2 while capturing the encoded video with a camera before decoding. This allowed us to isolate camera-induced timing effects while keeping all other parameters identical to the previous test.

III. RESULTS

A. *Results from bandwidth and accuracy tests*

Spatial Alignment: After incrementally increasing the ROI dimensions from 2 × 2, we found that a 20 × 20 grid of ROIs provided the largest configuration that maintained reliable alignment. Confidence in ROI placement was assessed visually, based on the proportion of ROI centers that correctly aligned with their corresponding squares in the encoded image.

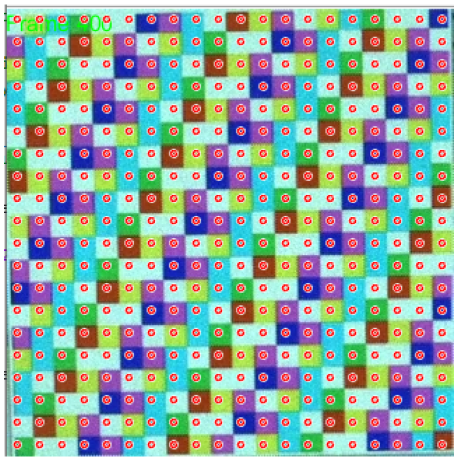


Fig. 2: 20 x 20 ROI placement

Color Intensity: Testing the effectiveness of the system with and without the crosstalk correction matrix demonstrated that applying the matrix under a high-gain setting most effectively increased the dynamic range. Fig. 3 shows that the measured dynamic ranges were 18.37 dB, 18.55 dB, and 29.86 dB for the no-matrix, regular-gain matrix, and high-gain matrix configurations, respectively. Based on these results, we concluded that implementing a high-gain video with a crosstalk correction matrix provides the best overall performance.

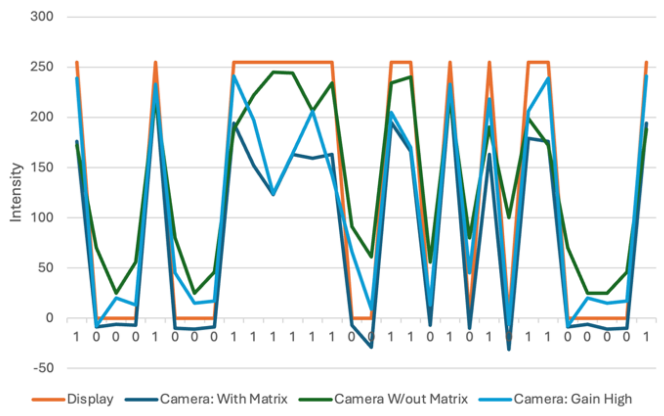


Fig 3. A graph of each possible binary triplet and the intensities measure through the camera for four different algorithms.

An additional observation from this experiment was that the (1, 1, 1) triplet—representing white—consistently produced the least accurate decoding, even under high gain. Since the white square originally denoted the end word, this finding suggests that the end signal should not be represented by white if the goal is to maximize dynamic range and decoding accuracy.

Temporal Alignment: The most significant improvement to the algorithm resulted from replacing frame averaging with clock-based monitoring. As shown in Table 1, while maintaining reasonable accuracy, the averaging-based approach was only able to capture less than 50% of the transmitted signal before terminating prematurely. In contrast, the clock-based algorithm shown in Table 2 successfully captured 100% of the signal, achieving solid bit error rates (BER).

Table 1: Bit error rate with the averaging system

Trial	Bits Decoded per Trial	BER (Percentage of bits sensed from total sent)	BER (Accuracy of bits decoded)
1	20,430	36%	0%
2	11,197	20%	3%
3	25,323	45%	0%

Table 2: Bit error rate with edge detection system.

Trial	Bits Decoded per Trial	BER (Percentage of bits sensed from total sent)	BER (Accuracy of bits decoded)
1	105,000	100%	0.20%
2	105,000	100%	0.50%
3	105,000	100%	1.70%
4	105,000	100%	1.70%
5	105,000	100%	0.03%

It is important to note that, for both algorithms, the methods used to detect the end word were the same. With this in mind, we can determine that the most crucial improvement between the average frame and the clock was the end-of-word detection. This confirmed our suspicions that the averaging algorithm could not handle time drift, resulting in the eventual measurement of transition frames.

B. *The characterization of the audio-to-visual function*

Fig. 4 shows the spectrogram generated by the encoder, with the regions of highest power corresponding to the frequency bands of the C4–C5 octave in increasing order. Each note exhibits the expected temporal duration of approximately 1 second. The frequency associated with the peak power at each sampled time bin is consistent with the spectrogram and is shown in Fig. 4.

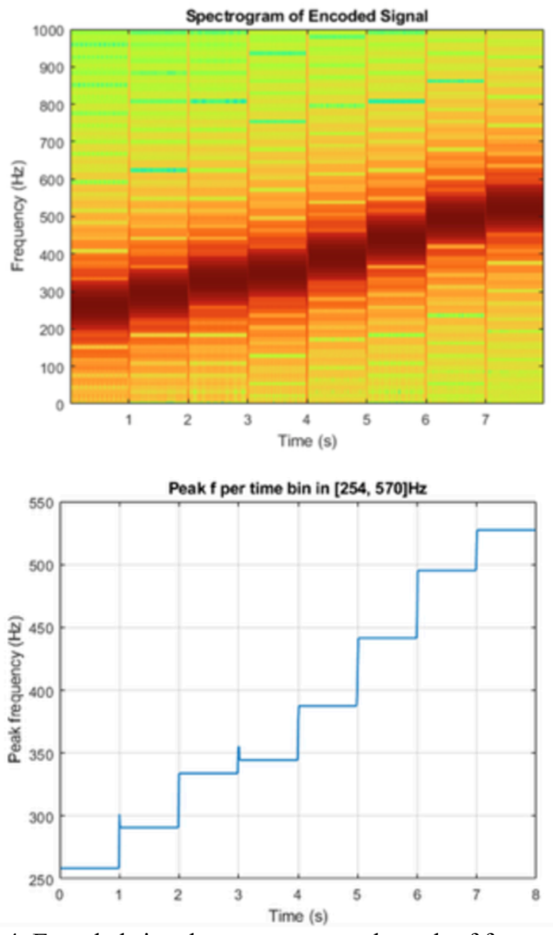


Fig 4. Encoded signal spectrogram and graph of frequencies.

Fig. 7 shows the start word and the encoded signal. The color bands are arranged with equal spatial dimensions in increasing order of their associated frequencies, noting that this test was run with magenta and cyan switched in the frequencies they represent, which was later changed to match the final representation shown in Fig. 1. The visual signal aligns closely with the expected encoding behavior, confirming that the function correctly mapped the audio data into a visual representation based on the spatial and color associations corresponding to the temporal and frequency characteristics of the sound.

Table 3: Total Bits in Input Audio Signal vs Communicated Visual Signal

Audio Input Signal (bits)	Frequency Banded, Communicated Signal (bits)
5,644,800	2,400

The frequency-banded version of the audio input signal represents only 0.425% of the bit number in the original signal, demonstrating an extreme reduction in information. This reduction is an intentional design characteristic of the function, offering significant advantages in communication scenarios where the signal either benefits from frequency filtering or where the information lost through reduction is negligible. In such cases, this function provides an efficient means of transmitting only the essential data, thereby enabling the communication of more information within a fixed bandwidth.

This design aligns directly with our goal of reconstructing pure audio, as the reduced representation preserves only the most salient frequency components. Fig. 10 illustrates this effect: while the input spectrogram contains multiple resonant frequencies, the spectrogram of the reconstructed audio displays only the pure, desired tones.

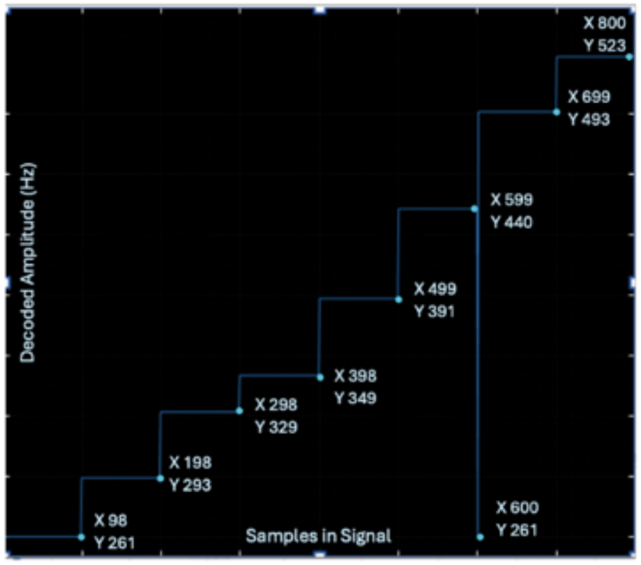


Fig. 5: Continuous Line Connecting Samples Decoded From Bit Triplets to Associated Frequency, Showing Number of Samples within Each Frequency Bin.

Each frequency band was set to 100 ± 2 samples, a small enough margin to ensure accurate sampling near the center of each window, assuming each window contains exactly 100 samples.

Module 1

<u>Test 1 — Tempo Tolerance (Without Camera):</u> Spectrogram correlation coefficients at 2, 4, 10, and 50 bps were 0.6901, 0.6267, 0.6714, and 0.4559, respectively (see Figs. 8–11). Accuracy remained comparable across 2–10 bps but decreased significantly at 50 bps. The slightly lower correlation at 4 bps (0.6267) is attributed to resonance components visible in the input spectrogram but absent in the reconstruction, rather than to a systematic failure. Overall, the method demonstrates robustness to moderate tempo increases, with performance degradation only at very high bit rates.

<u>Test 2 — Tempo and Duration Tolerance (Without Camera):</u> Both test patterns reconstructed with correct durations at 8 s and 16 s, indicating robustness to both content and duration, as shown in Figs. 12–15. However, inputs longer than 16 s could not be processed due to a fixed decoder design with 800 time bins, which is insufficient for longer audio sequences.

<u>Test 3 — Temporal Alignment (With Camera):</u> In the 8 s trials (2 bps melody and 1 bps octave), reconstruction was fully successful, but extending these inputs to 16 s yielded only approximately 8 s of recovered audio, as shown in Figs. 16–19. Because these same inputs reconstructed correctly when the encoder's video output was passed directly to the decoder (bypassing the camera), the discrepancy likely stems from a camera-related timing issue within the encode/decode pipeline. We hypothesize that the problem originates in the clock synchronization, which was not evident during long-duration camera tests but becomes apparent when transmitting very short audio segments, in this case as a few bits across only two frames.

IV. CONCLUSION

Overall, this extension aimed to create a function that mapped audio data to visual data such that the visual data showed certain characteristics of the audio itself. Additionally, it was imperative to be able to reconstruct the audio from the image, making the function invertible. For future work, we would explore adaptive frequency windowing dependent on the signal's spectral complexity, enhancing the capabilities of the algorithm. We would also implement brightness and a larger range of colors to produce an image that is more visually representative of the audio.

V. APPENDIX

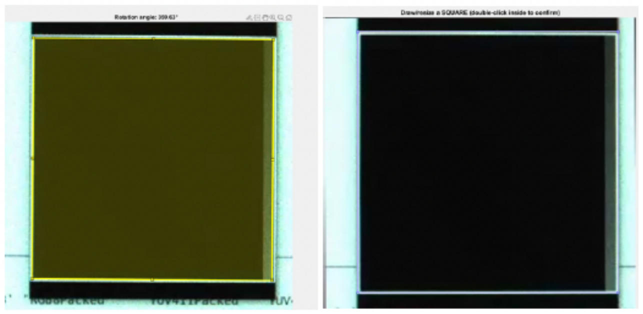


Fig 6: The image on the left is a screenshot of the processing of finding the angle from the center. The image on the right is the cropping mechanism.

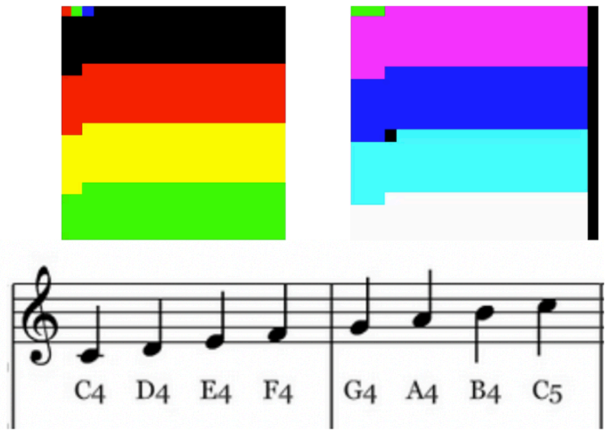
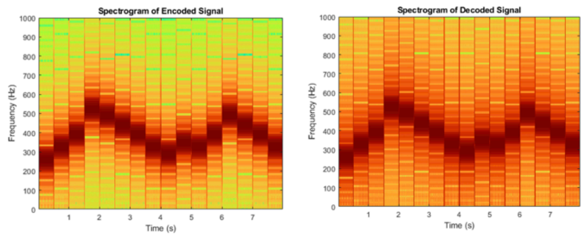
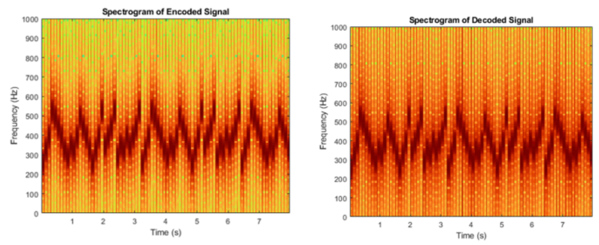


Fig 7: Encoded Visual of C4-C5 Increasing Octave, 8s, 1bps Audio Input (Top) and Corresponding Piano Music. Start word Red-Green-Blue shown in upper left.



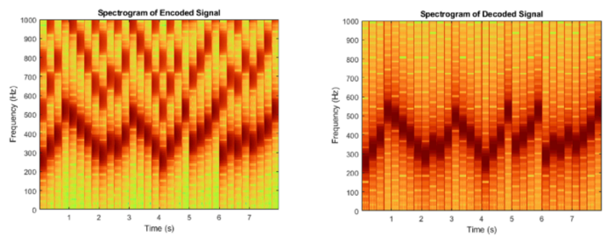
C4-C5 Melody, 2bps, 8s : Spectrogram correlation (dB): 0.6901

Fig 8: Encoder and decoder spectrograms for the 8-secondlong, 2 beats per second audio input. Without the camera.



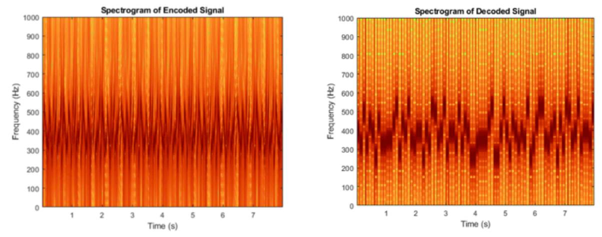
C4-C5 Melody, 10bps, 8s : Spectrogram correlation (dB): 0.6714

Fig 9: Encoder and decoder spectrograms for the 8-secondlong,10 beats per second audio input. Without the camera.



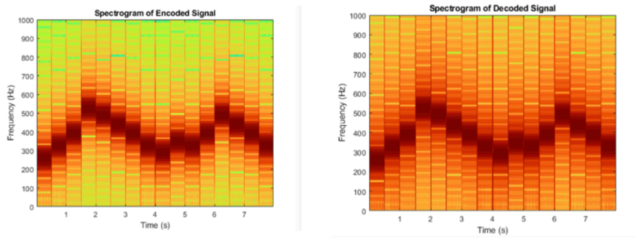
C4-C5 Melody, 4bps, 8s : Spectrogram correlation (dB): 0.6267

Fig 10: Encoder and decoder spectrograms for the 8-second long, 4 beats per second audio input. Without the camera.



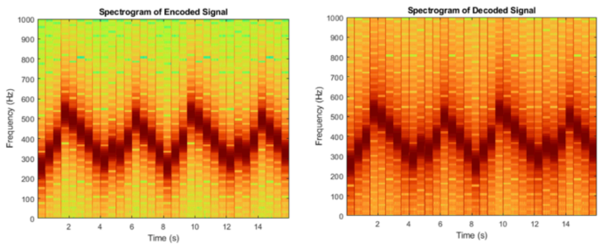
C4-C5 Melody, 50bps, 8s : Spectrogram correlation (dB): 0.4559

Fig 11: Encoder and decoder spectrograms for the 8-second long, 50 beats per second audio input. Without the camera.



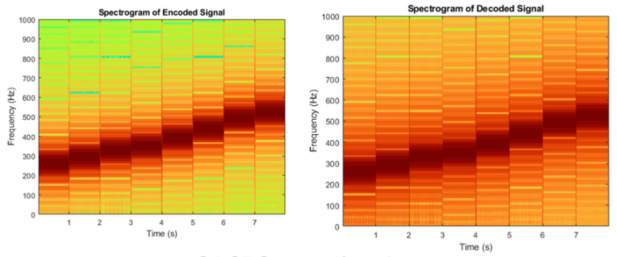
C4-C5 Melody, 2bps, 8s

Fig 12: Encoder and decoder spectrograms for the 8-second long, 2 beats per second audio input. Without the camera.



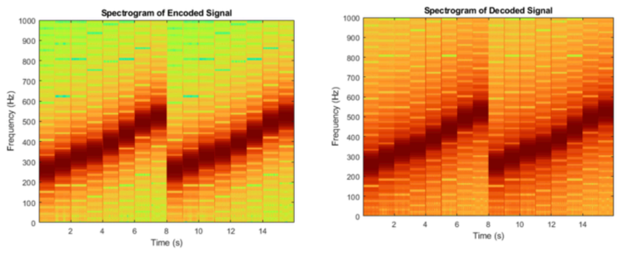
C4-C5 Melody, 2bps, 16s

Fig 13: Encoder and decoder spectrograms for the 16-second long, 2 beats per second audio input. Without the camera.



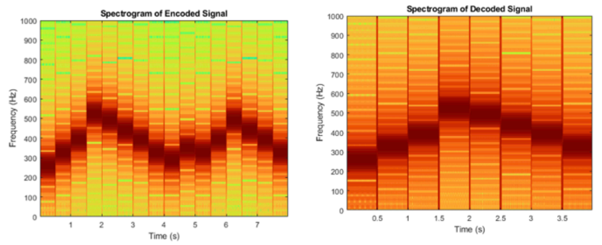
C4-C5 Octave, 1bps, 8s

Fig 14: Encoder and decoder spectrograms for the 8-second long, 1 beat per second audio input. Without the camera.



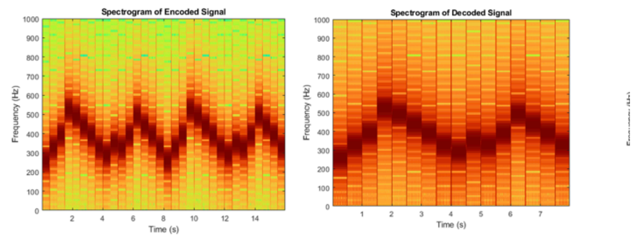
C4-C5 Octave, 1bps, 16s

Fig 15: Encoder and decoder spectrograms for the 16-second long,1 beat per second audio input.



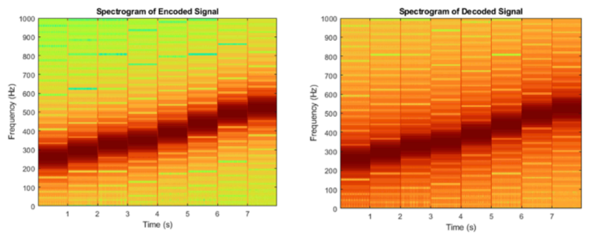
C4-C5 Melody, 2bps, 8s

Fig 16: Encoder and decoder spectrograms for the 8-second long, 2 beats per second audio input. Using the camera. Decoder spectrogram shows reconstruction of only half of encoding.



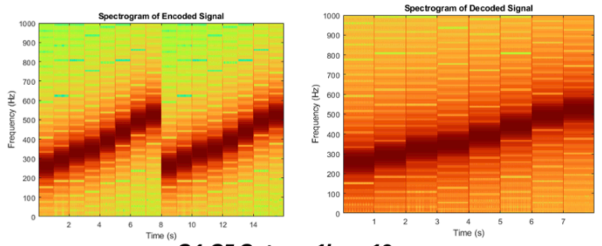
C4-C5 Melody, 2bps, 16s

Fig 17: Encoder and decoder spectrograms for the 16-second long, 2 beats per second audio input. Using the camera. Decoder spectrogram shows reconstruction of only half of encoding.



C4-C5 Octave, 1bps, 8s

Fig 18: Encoder and decoder spectrograms for the 8-second long, 1 beat per second audio input. Using the camera.



C4-C5 Octave, 1bps, 16s

Fig 19: Encoder and decoder spectrograms for the 16-second long, 1 beat per second audio input. Using the camera.

Module 1

Decoder spectrogram shows reconstruction of only half of encoding.