EE384B: Multimedia Networking and Communications

Quiz #1 Solutions
Closed Book
Time: 30 minutes

April 19, 2001

Score: ___ / 36

Name:_____

E-mail:_____

☐ SITN student ☐ On-campus student

I agree to abide by the Stanford Honor Code:_____
(signature)

Question 1: Audio Compression Techniques (5 points)

There are several methods for coding and compressing voice (PCM, ADPCM, CELP). The MPEG standards also address audio compression, using completely different techniques. Why are the techniques used in MPEG audio compression so different from the voice compression techniques?

Voice is a "special case" of audio, with very well defined characteristics and with limited bandwidth (4 kHz). The voice coding methods are specifically tailored for the particular characteristics of voice, for example, the fact that, due to its limited bandwidth, it is slowly-varying signal (temporal redundancy), or the fact that voice is produced by the human vocal tract (and thus can be transmitted by modeling it and transmitting the model parameters rather than the actual signal). MPEG Audio is designed for compression of more general audio signals, such as music, with up to 24 kHz of bandwidth, and makes use of psycho-acoustic models for compression.

In summary, the differences come from the fact that they are intended for different kinds of audio signals.

Question 2: Latency in Video Encoding (5 points)

In general, when encoding video, there is a tradeoff between latency and quality. Why? Explain why latency suffers if you want the best possible quality, and why you need to give up quality to get low latency.

Fundamentally, to get the best possible quality for a given bit rate, the encoder needs to be able to extract the temporal correlation between frames, both present and future. However, waiting for future frames to encode the current frame at hand adds latency. This is the primary issue that controls the tradeoff between latency and quality; to achieve low latency, the encoder gives up the ability to use future information that could either reduce the bit rate for the same quality or increase the quality for the same bit rate.

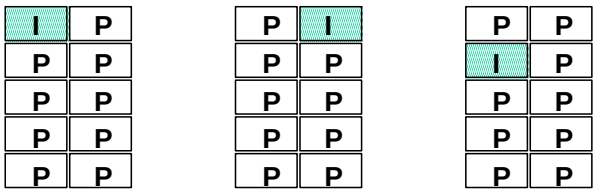
Other issues, not as fundamental:

- *Technology: real encoders have non-zero processing time. They need time to search for the best match in a P or B frame (this comes as a pipelining delay); if the time is limited, the quality of the match may not be optimal.*
- *Buffering: if the encoder is outputting to a constant bitrate channel, there is a delay on the VBV. Reducing the VBV reduces this delay, but the quality of the image is also reduced since the encoder reacts "too fast" to increases in scene complexity by going to coarser quantization.*

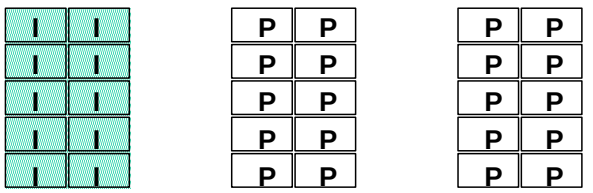
Question 3: Latency in H.261 (8 points)

The following two options are available in H.261:

Technique 1: Rotational intracoding of GOBs:



Technique 2: One completely intracoded picture followed by a number of predictive pictures:



a) Technique 1 above is used to reduce latency, as compared with technique 2. Why does it reduce latency? (4 points)

Intracoded GOBs have more bits than predictive GOBs. Therefore, the number of bits per frame of video in technique 2 varies with the frame type; a completely intra-coded frame is much larger than a predictive frame. In technique 1, all the frames are approximately of the same size. The required decoder buffer to display the stream encoded with technique 1 is then smaller than that required for technique 2, leading to a shorter buffering delay.

b) Assume that the video encoded using techniques 1 and 2 is transmitted over a channel with bit errors. Qualitatively compare the two techniques in regards to glitch duration. For technique 2, consider that the number of P frames between I frames is a design value, and discuss the effect it has on the glitch duration. (4 points)

When there is a bit error in the channel, that bit error leads to a glitch in the frame to which the errored bits belong. However, that frame is a reference for the subsequent frames, and they will also be "damaged". This continues until an intra-coded frame or macroblock is received; at that point, the glitch ends because the intra-coded block is not dependent on past frames. This is true for both techniques.

In technique 1, the glitch can potentially extend for a large number of frames, until that region of the image is refreshed by an intra-coded GOB. Note that, because of motion compensation, this may take a while and will be a function of the image. In technique 2, the next I-frame will clear the glitch. Therefore, the average glitch duration is known and is equal to (n+1)/2, where n is the number of P-frames between I-frames.

Question 4: Delay Jitter in MPEG (5 points)

Explain what is the effect of delay jitter in a live MPEG transmission system (i.e., a system where the output of a real-time encoder is being transmitted through a network to a real-time decoder).

Delay jitter means that the packet arrives either before or after its nominal arrival time. If a packet arrives early, this could, in theory, cause the decoder buffer to overflow, but in practice it is not an issue because networked decoders typically have large buffers. A packet that arrives late may cause the decoder to underflow (run out of data) and stutter. To avoid this, it suffices to have the decoder pre-buffer enough data prior to start of play to ensure that this condition will not happen. Pre-buffering more data means increased latency - in other words, the additional buffering causes the worst-case delay jitter to be added to the end-to-end latency.

Question 5: Encoder/Decoder Synchronization (5 points)

In a live MPEG transmission system (a real-time encoder transmitting to a real-time decoder), why is it necessary to synchronize the encoder and the decoder? Briefly explain two synchronization techniques.

The video frames enter the encoder at a rate determined by the video source. They are transmitted through the network, and leave the system (i.e., are played by the decoder) at a rate determined by a local clock in the decoder. These two rates are nominally the same. However, if they are not exactly synchronized, the decoder will eventually overflow (if it is running at a slightly lower rate than the encoder) or underflow (if it is running slightly faster). To synchronize them, the following techniques can be used:

- *Use a PLL to lock the decoder clock with the encoder clock. This PLL is driven by the time stamps present in the MPEG signal. The decoder will have a local timestamp register, which will be compared to the incoming timestamps to derive an error signal; this error signal is then used to adjust the decoder clock.*
- *Use the decoder buffer fullness to adjust the clock. If the buffer is going dry, slow down the clock; if it is getting full, speed it up.*
- *If the decoder clock is not adjustable, the solution is to have the decoder drop or repeat frames as appropriate (drop frames if the buffer is getting too full, repeat frames if it is getting close to empty).*

Question 6: Scalable Video (8 points)

Using MPEG-4, it is possible to generate ***scalable video***. A scalable video stream has a ***base layer***, which provides a certain minimum quality, and a number of ***enhancement layers***. Combining the base layer with the enhancement layers provides increasing video quality. Consider a scenario where MPEG-4 is being used in a live point-to-multipoint video distribution system, where there are heterogeneous receivers. Receivers vary in their connection speed to the network; some have low speed links, some have very fast connections, and others have available something in between. The backbone links between the receivers and the video source also vary. Explain how you would use the following network features to get the best possible video quality for each receiver:

- IP Multicast.
- IP Precedence.

Justify any statement you make.

The first obvious use of IP Multicast is to support the basic point-to-multipoint operation: the server transmits a single copy of the stream, and the network replicates it as needed. This way, an arbitrary number of receivers can be supported without any additional load on the server. The functionality can be further improved by assigning ***different*** *multicast IP addresses to the base layer and each of the enhancement layers. This way, each receiver can join only the number of layers that its MPEG-4 decoder and/or network attachment can support.*

One issue with the implementation proposed above is the situation where the receiver does not know the available bandwidth. It could try adding layers one by one until it detects packet loss, and then back off. However, when packet loss due to congestion happens, it is spread uniformly all over the layers. If many receivers are doing this at different times on a shared link, quality may be degraded. Adding the IP Precedence can solve this problem – each enhancement layer gets a different precedence level, with the base at highest priority. This way, the packet loss is concentrated in the lower-priority layers.

Note that one could use a single IP Multicast address for the whole stream, and differentiate the enhancement layers with priorities. This causes each decoder to get the most from its network connection, but it will saturate the network links whose capacity is lower than the overall stream data rate and will cause unnecessary transmission in the network.

BONUS QUESTION (Extra Credit): (4 extra points)

When coding P or B macroblocks, the encoder finds the best match block in the reference frame, computes the difference between that block and the block being coded, and takes the DCT of this difference. Alternatively, it could take the DCT of the reference block and the block being coded, and compute the difference in the frequency domain by subtracting the coefficients. What is the effect of doing this? Is there any advantage of doing it one way or another? Justify.

Since this is a closed-book test, here is the DCT equation in case you need it:

$$S_{vu} = \frac{1}{\sqrt{N1N2}} c_u v_u \sum_{x=0}^{N1-1} \sum_{y=0}^{N2-1} s_{yx} \cos\frac{(2x+1)u\pi}{2N1} \cos\frac{(2y+1)v\pi}{2N2}$$

$$c_u, v_u = \frac{1}{\sqrt{2}}, u = 0, v = 0$$

$$c_u, v_u = 1, else$$

From the DCT equation, you can see that the operation is linear; just replace s_{yx} by ($s1_{yx}$ - $s2_{yx}$). You get the same result by taking the DCT of the difference or by computing the difference of the DCTs. So, there is no difference from a math point of view!

*Now, from an implementation point of view, there is a big difference in number of operations to reach the result. If you take the DCT of the difference, you do **one** matrix subtraction and **one** DCT; if you take the difference of the DCTs, you do **two** DCTs and **one** matrix subtraction.*