Table of Contents

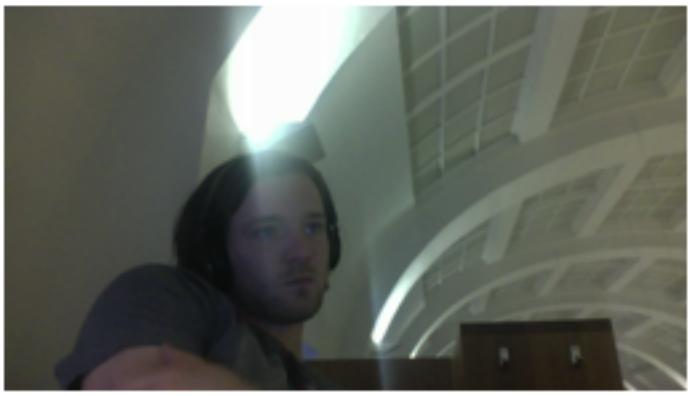
Convert Movie AVI to MPG and clock time

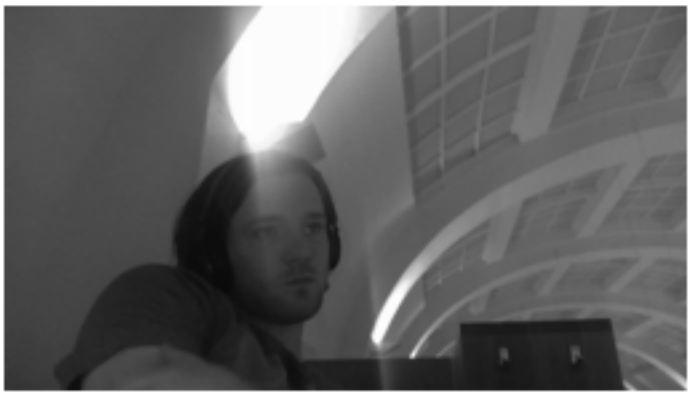
```
% First we open a Video reader/writer.
readerObj = VideoReader('testABCD.mp4');
writerObj = VideoWriter('testABCDAVI.avi', 'Uncompressed AVI');

% We scale down the original video to save computing time.
height = readerObj.Height * 0.2;
width = readerObj.Width * 0.2;
mov(1:readerObj.NumberOfFrames) = struct('data', zeros(height, width, 3, 'uint8'),

tic %Starts counting
open(writerObj);
for k = 1:readerObj.NumberOfFrames
    mov(k).data = imresize(read(readerObj,k), [height, width]);
    writeVideo(writerObj, mov(k).data);
end
close(writerObj);
disp('Time it took to convert MPG to AVI:');
disp(toc); %Display eleapsed time
```

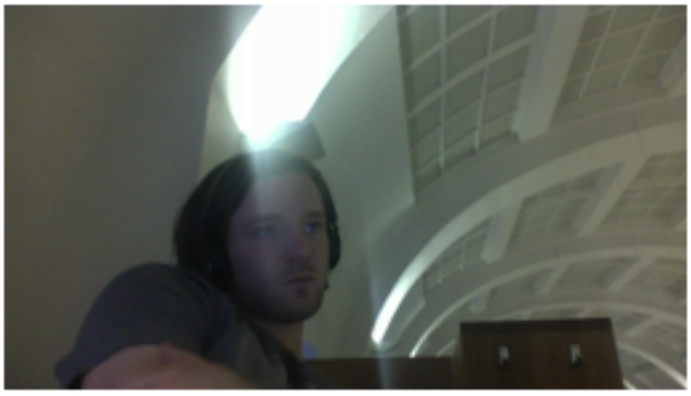
```
Time it took to convert MPG to AVI:
   25.2367
```

Show RGB -> YUV (Y) -> RGB

```
randomIndex = randi([1 readerObj.NumberOfFrames]);
randomFrameRGB = mov(randomIndex).data;
randomFrameYUV = frameRGB2YUV(randomFrameRGB);

randomFrameY = uint8(randomFrameYUV(:,:,1));
randomFrameBackToRGB = frameYUV2RGB(randomFrameYUV);

figure,
imshow(randomFrameRGB);
figure,
imshow(randomFrameY);
figure,
imshow(randomFrameBackToRGB);
```



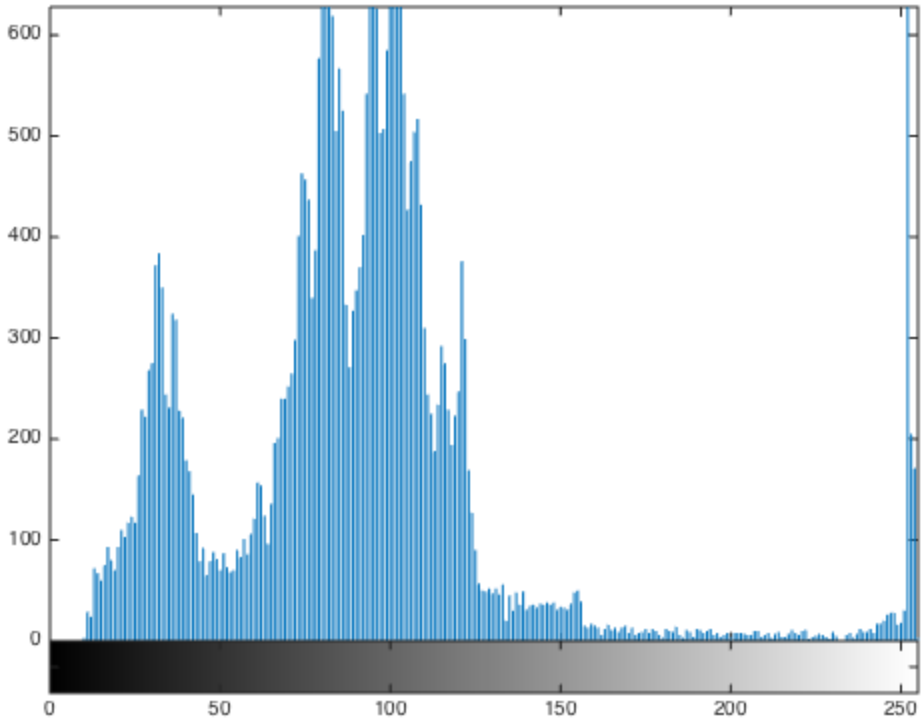




Huffman encoding / decoding

```
figure, imhist(randomFrameY);
disp('Histogram of frame used for Huffman Codebook:');
disp(entropy(randomFrameY));

codeBook = huffmanCodebook(randomFrameY, 0, 256);

randomIndex = randi([1 readerObj.NumberOfFrames]);
randomFrameYUV = frameRGB2YUV(mov(randomIndex).data);
randomFrameY = uint8(randomFrameYUV(:,:,1));

disp('Entropy before huffman encoding');
disp(entropy(randomFrameY));

randomFrameYHuffmanEncoded = huffmanEncoder(randomFrameY, codeBook);
disp('Entropy after huffman encoding:');
disp(entropy(randomFrameYHuffmanEncoded));

randomFrameYHuffmanDecoded = huffmanDecoder(randomFrameYHuffmanEncoded, codeBook,

if isequal(randomFrameY, randomFrameYHuffmanDecoded)
    disp('No data lost');
end
```

```
Histogram of frame used for Huffman Codebook:
    6.8281

Entropy before huffman encoding
    6.8166

Entropy after huffman encoding:
    0.9981

No data lost
```



Differential encoding/decoding

```
% First we make a random 200 x 200 matrix.
frame = uint8(rand(200, 250) * 255);
[height, width] = size(frame);

% Then we encode it with differential-encoding
frameDiffEncoded = diffEncoder(frame);

% Then we decode it with differntial-decoding.
frameDiffDecoded = diffDecoder(frameDiffEncoded, height, width);

% And to prove that the encoding works without data loss:

if isequal(frame, frameDiffDecoded)
    disp('No data loss');
end
```

```
No data loss
```

Published with MATLAB® R2014b