Question 1

- The quantization steps are: [-3.75, -3.5, -3.25, -3.0, -2.75, -2.5, -2.25, -2.0, -1.75, -1.5, -1.25, -1.0, -0.75, -0.5, -0.25, 0.0, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0, 2.25, 2.5, 2.75, 3.0, 3.25, 3.5, 3.75, 4.0].
  The quantized values are: 1.75, 2.25, 2.25, 3.25, 3.25, 3.25, 2.5, 2.75, 2.75, 2.75, 1.5, 1.0, 1.25, 1.25, 1.75, 2.25, 2.25, 2.25, 2.0, 2.25, 1.25, 0.25, -1.25, -1.25, -1.75, -1.0, -2.25, -1.5, -1.5, -0.75, 0.0, 1.0.
  The corresponding quantization levels are: 22, 24, 24, 28, 28, 28, 25, 26, 26, 26, 21, 19, 20, 20, 22, 24, 24, 24, 23, 24, 20, 16, 10, 10, 8, 11, 6, 9, 9, 12, 15, 19.

- 32 levels, so $log_2(32) = 5$ bits.

Question 2

- 4:2:0 chroma subsampling with 8 bits per channel means 12 bits per pixel on average, so bitrate is $450 * 520 * 25 * 12 = 70200000bps$.

- 4:2:0 chroma subsampling with 6 bits per channel means 9 bits per pixel on average, so new bitrate is $450*520*25*9 = 52650000bps$. Number of bits for $10min = 600s$ is $52650000*600 = 31590000000b \approx 3.9GB$.

Question 3

- Assume the white mark is at zero angle at the start of the recording; at time of first frame $\frac{1}{25}s$, the white mark is at $\frac{1}{25} * 2\pi * 20 = \frac{8\pi}{5}rad$. However, because of discontinuity, the frame is referred to the previous frame via the shortest path, so the observed angle is $\frac{8\pi}{5} - 2\pi = -\frac{2\pi}{5}rad$. Therefore over the 25 frames in 1 second, it would appear that the white mark has moved $-\frac{2\pi}{5} * 25 = -10\pi\ rad = -5\ cycles$, i.e. the observed speed of rotation is 5 $rotations/sec$ in the reverse direction.

- Angle at time of first frame is $\frac{1}{25} * 2\pi * 10 = \frac{4\pi}{5}rad$, and in this case the shortest path of reference is still $\frac{8\pi}{5} - 0 = \frac{4\pi}{5}rad$, so the observed speed of rotation is the same as actual speed at 10 $rotations/sec$.

Notes on the programming part

Hi! I wish to clarify on how I implemented logarithmic quantization, just in case the quantized image doesn't look quite the same as reference (this might be the case due to, for example, choice of parameters or the original image). In short, I essentially applied the inverse of mu-law to the uniform quantization steps normalized to [0,1]. I first divided the total number of steps between the two halves [0,pivot] and [pivot,255], proportional to their respective interval sizes and rounded to the nearest integers. Then, using the numbers of steps for each half, I generated two sets of uniform steps quantizing [0,pivot] and [pivot,255] respectively. For each set of uniform steps, I normalized all its steps to [0,1], applied to the normalized steps s the inverse of mu-law $\frac{2^{8s}-1}{256}$, and scaled them back to get the logarithmic quantization steps. I chose log base 2 and μ value 255 in this procedure, which may be tweaked to get stronger or weaker bias. Finally, I quantized all pixels by rounding to the middle between two adjacent steps, as in uniform quantization.
For example, the steps generated for $q = 3$ and $m = 0$ are:

$$0.000, 0.996, 2.988, 6.973, 14.941, 30.879, 62.754, 126.504, 254.004$$

and steps generated for $q = 3$ and $m = 128$ are:

$$0.500, 96.500, 120.500, 126.500, 128.000, 129.488, 135.441, 159.254, 254.504$$