Megan Rogge - A2

1. I tried static, adaptive, context adaptive, and finally landed on what I call PriorFrameAdaptive modeling. I spent a ton of time during the first few days following the assignment's release building models from scratch: a video model with 300 frame models with each frame model having 4096 pixel models (of a class PixelModel that implemented SourceModel<>). I spent alot of time trying to make these work. When I finally abandoned the custom models, things got easier.

2. Videos should be as smooth as possible when transitioning from frame to frame so as to make them appear continuous to the human eye. For this smooth transition to take place, changes must occur gradually, and any given pixel within a frame will likely have the same or nearly the same intensity as it had in the prior frame. My PriorFrameAdaptive scheme takes advantage of this inherent need for changes to take place slowly by using the pixel model of the prior frame as the corresponding pixel model for the current frame. By using the prior frame as the model for the current frame, my implementation exhibits temporal coherence in compressing the data.

3.

Listed from best to worst performance:

PriorFrameAdaptive (mine)	670,092 bytes
Context adaptive	909,144 bytes
Adaptive	1,063,224 bytes
Static	1,064,024 bytes

My scheme outperformed all of those that were intended for text. The best performance of the schemes built for text was the context adaptive one when applied to video compression.

4.

A great deal of the pixel intensities don't change very much from frame to frame, if they change at all. Though my scheme makes use of this by choosing a model for a pixel based on the pixel in the prior frame, it doesn't capitalize upon the potential redundancy eliminating aspect that differential coding introduces. By using the differential/residual technique, I think that the performance of my scheme would improve.