**Problem Definition**

**Input:** Grayscale photo and color photo of similar content  
**Output:** Colorized version of the grayscale image

**Challenges**

• Want fully automated colorization  
• Output has higher dimension than input  
• Colorization solution is not uniquely defined

**Training**

1. Input a color image
   
2. Separate the luminance from the color (αβ) channels
   
3. Discretize the color space using K-means (32 colors)
   
4. Compute luminance SURF texture features and FFT for a square neighborhood around every pixel (556 features)
   
5. Perform PCA to reduce dimensionality (30 features)
   
6. Train a collection of SVMs, one per discrete color

**Testing**

1. Input a grayscale image
   
2. Compute texture features using the same steps as in training (SURF + FFT)
   
3. Project the data into the subspace chosen by PCA in the training step
   
4. Compute the margin each SVM assigns to each test pixel
   
5. Detect edges in the grayscale image using the Sobel filter
   
6. Solve an energy minimization problem with the above objective function (includes label cost and spatial coherency terms). Use Graph Cut algorithm to solve for optimal color labels

The output when trained and tested on the same landscape image. Almost perfect colorization is achieved.

Another colorization using images from the Pasadena Houses dataset. Challenging due to the presence of areas with little texture.

Conclusions / Future Work

• Has minimum input from user  
• Incorporates spatial coherency into predictions  
• Future Work: Improving weights to “snap” to edges

**References**