Original image:  
A picture containing text, outdoor, sky, apartment building

Description automatically generated

Final image:  


Code:

wiki\_img\_url = "https://upload.wikimedia.org/wikipedia/commons/thumb/e/ea/Ray\_and\_Maria\_Stata\_Center\_%28MIT%29.JPG/230px-Ray\_and\_Maria\_Stata\_Center\_%28MIT%29.JPG"  
wiki\_img = download\_img(wiki\_img\_url)  
wiki\_img\_arr = img2arr(wiki\_img)  
display(wiki\_img)

wiki\_img\_colors = wiki\_img\_arr.reshape((-1, 3))

cache = {} # for reruns

def find\_silhouette\_score(n\_clusters, data):  
 if n\_clusters not in cache.keys():  
 k\_means = KMeans(n\_clusters=n\_clusters)  
 k\_means.fit(data)  
 cache[n\_clusters] = silhouette\_score(data, k\_means.labels\_)  
 return cache[n\_clusters]  
  
x\_arr = np.arange(2, 15)  
scores = [find\_silhouette\_score(x, wiki\_img\_colors) for x in x\_arr]

## Find appropriate number of clusters  
plt.plot(x\_arr, scores)  
plt.xlabel("Clusters")  
plt.ylabel("Score")  
plt.show()

## select cluster number with highest score  
index\_max\_score = np.argmax(scores)  
n\_clusters\_max\_score = x\_arr[index\_max\_score]

kmeans = KMeans(n\_clusters=n\_clusters\_max\_score)  
kmeans.fit(wiki\_img\_colors)

## plot result  
rg\_chroma\_plot(wiki\_img\_arr, kmeans.cluster\_centers\_)  
  
## display result  
replaced = replace\_nearest\_color(wiki\_img\_arr, kmeans.cluster\_centers\_)  
display(arr2img(replaced))

I’ve used the silhouette score to determine an appropriate number of clusters. Which in my case seemed to be three clusters, after observing the plotted the scores.