k-Means Clustering for Image Segmentation

I implemented k-Means Clustering. K-Means Clustering is an unsupervised learning image segmentation algorithm. The algorithm aims to partition the samples into clusters with the mean of the samples serving as a prototype for the cluster. A colour image is used as the input, with the pixel's values being represented in a 3D array with red, green, and blue intensities. The algorithm starts by assigning k centers randomly or systematically. The distances to each center are calculated. Each pixel's class is decided by which center is closest to the pixel. All the pixels are clustered using this method. Once the clusters have been created, k new centers are calculated using the mean of the pixels in each cluster. This process is repeated until we achieve convergence. Convergence is declared when there is no change in the clusters between iterations. An image is included below to illustrate the algorithm in 2 dimensions. In the end, all the pixels in each cluster are given the value of that clusters center, creating an image comprised of k colours.

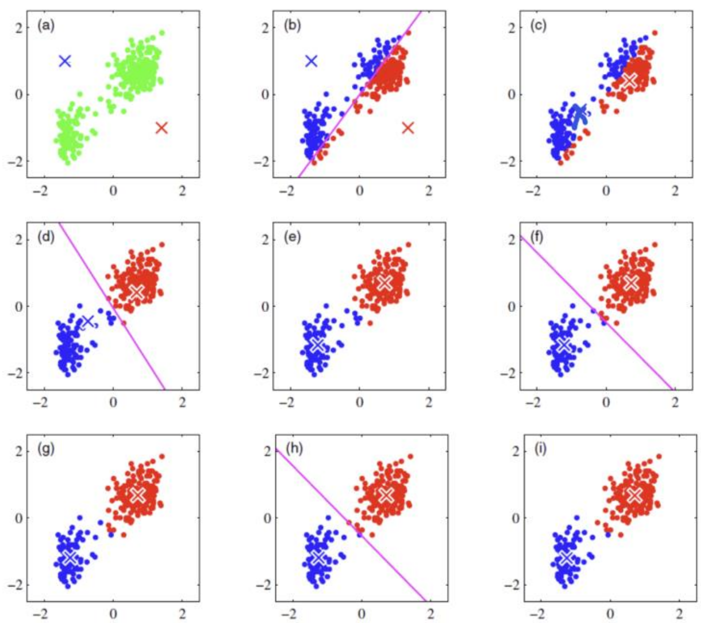


Figure 1. PRML [1], Fig. 9.1, pp. 426

The initialization strategy I used for the third round of processing was a modified version of the popular k-mean++ algorithm. The algorithm is comprised of 4 steps.

1. Choose the first center at random from the data points.
2. Calculate the distances between all points excluding the centers, to all centers.
3. Select the smallest distance (which is the closest center) of each point.
4. Each point now has one distance associated with it. Choose the point with the largest distance to be the next center. Add this center to list of centers.
5. Repeat step 2 – 4 until k centers have been chosen.

I chose to not create a probability distribution proportional to the distances as suggested in [2] for simplicity. The above algorithm creates sufficient distances between centers. Below is a plot of k=10 initial centers using the random strategy and the described algorithm.

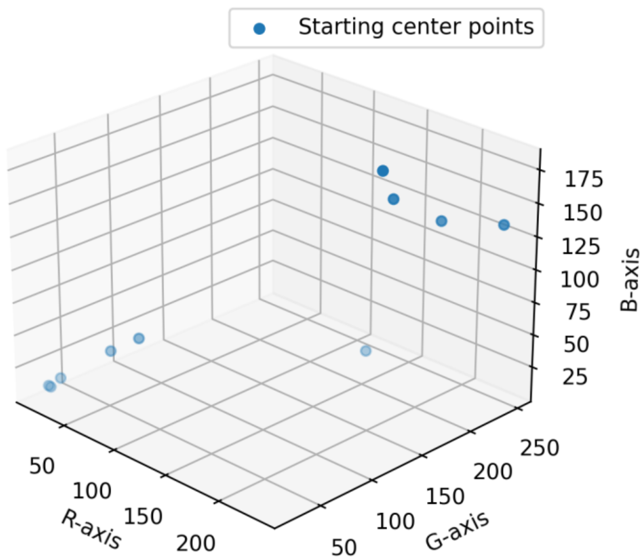


Figure 2. K=10 - Random center initialization

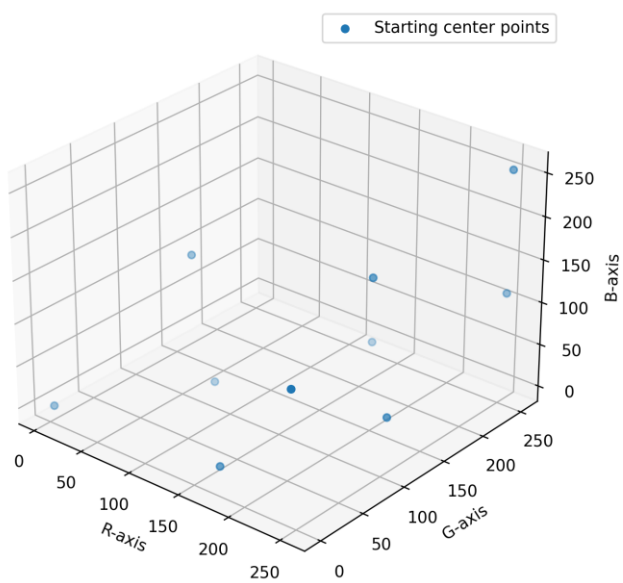
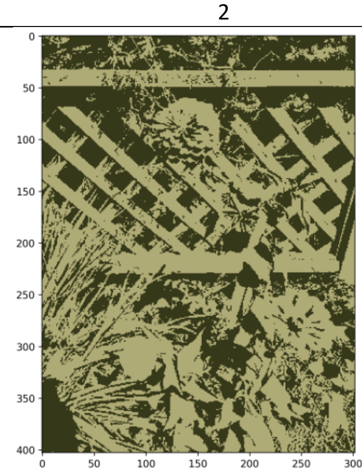


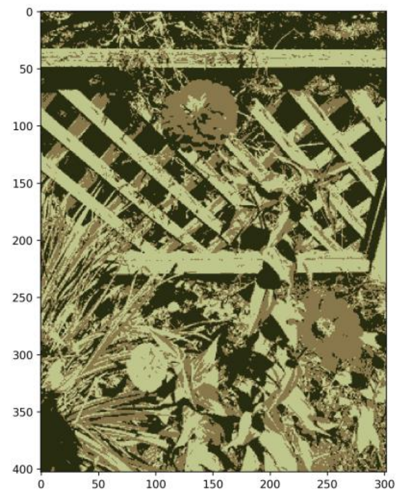
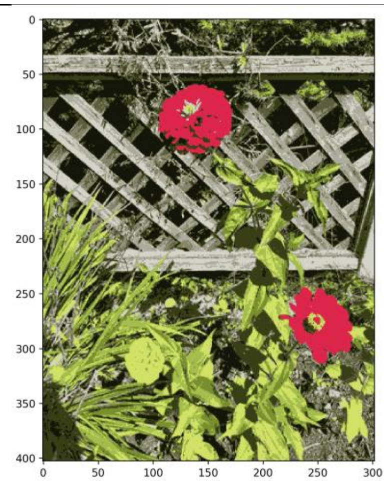
Figure 3. K=10 - K-mean++ algorithm for initialization

Image 1.

Image 2.

Table 1. Initialization set to random (1)

K value	Image		Number of Iterations		MSE	
	1	2	1	2	1	2
2			29	11	1024.9	1602

3			16	26	639	1048
10			47	94	175.6	277.5
20			103	133	84.7	164.5

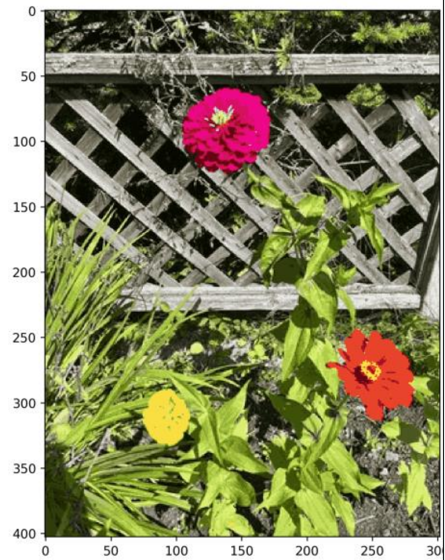
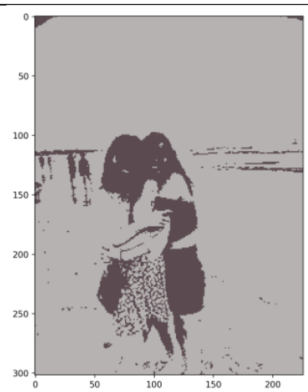
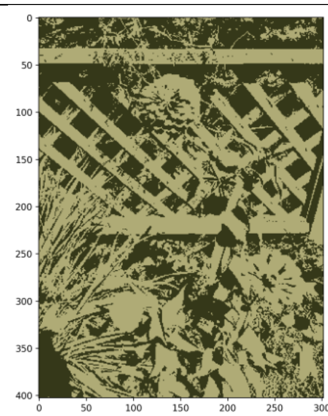
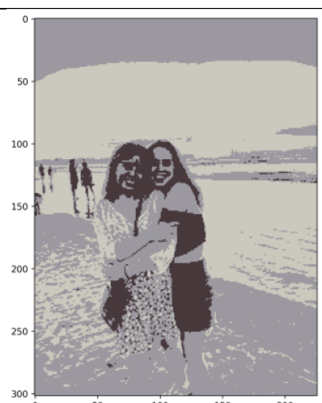
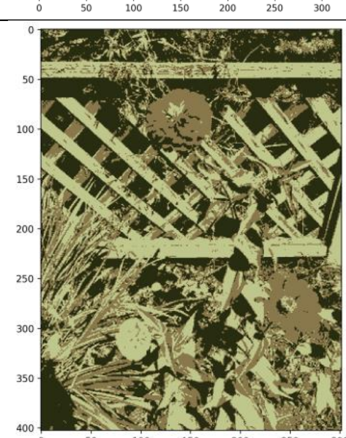
40			12 7	30 5	43.4	74.3

Table 2. Initialization set to random (2)

K value	Image		Number of Iterations		MSE	
	1	2	1	2	1	2
2			30	12	1024.9	1602
3			30	26	639	1024.3

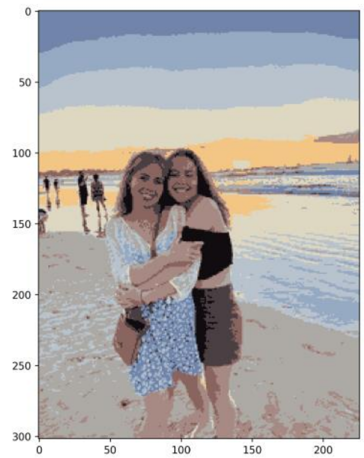
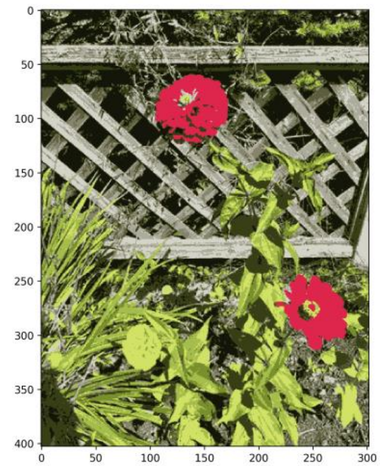
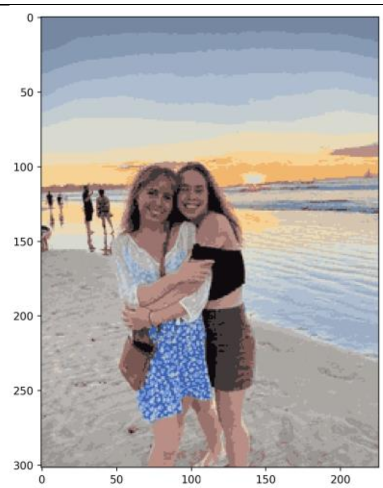
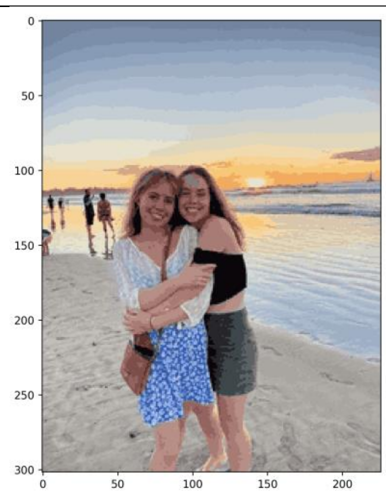
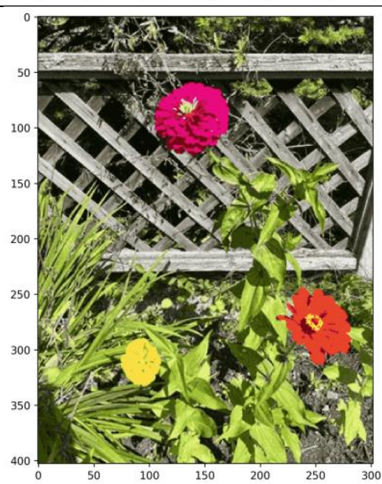
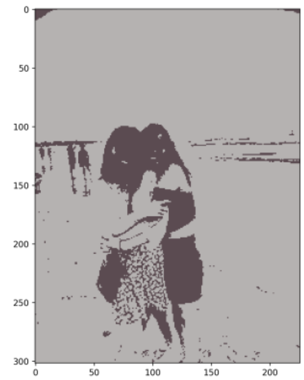
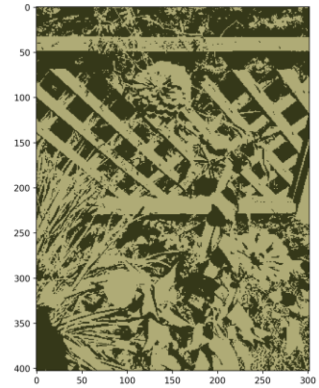
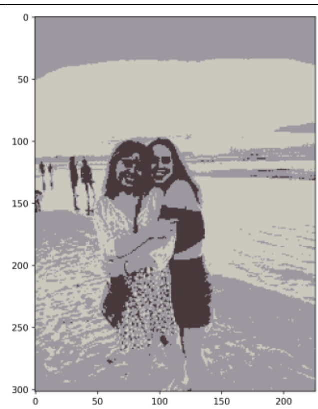
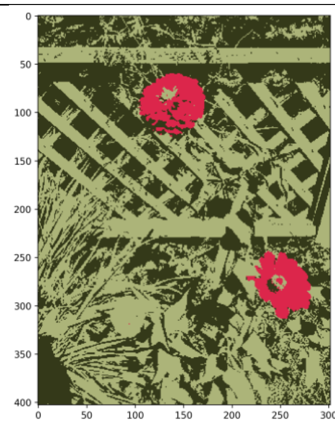
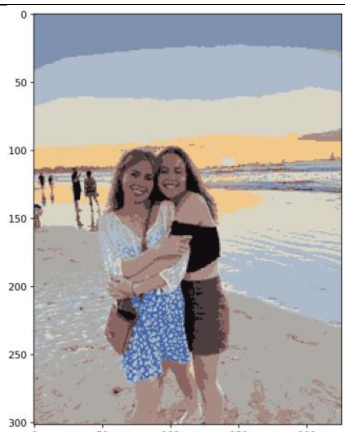
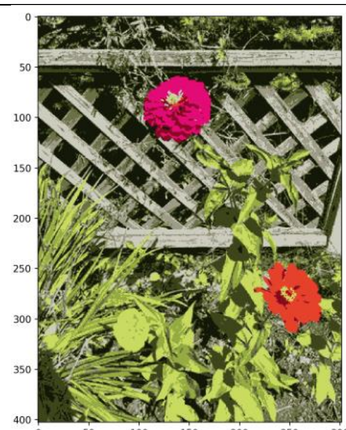
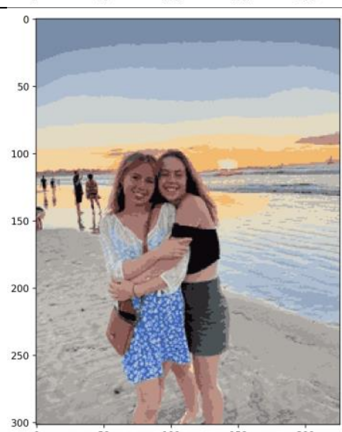
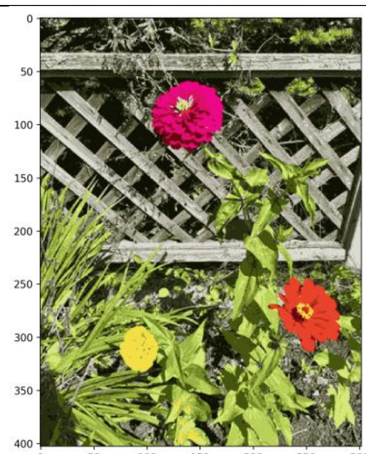
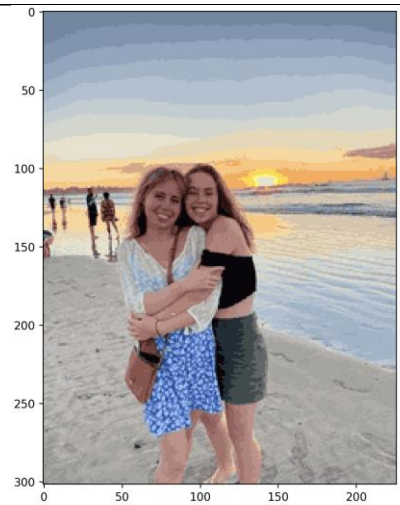
10			62	166	173	277.5
20			46	198	85.2	164.2
40			161	430	44.7	74.5

Table 1. Initialization set to large distances between centers

K value	Image		Number of Iterations		MSE	
	1	2	1	2	1	2

2			32	11	1024.9	1602
3			27	13	639	1302.6
10			40	50	174.6	302.7
20			84	291	84.5	153

40			70	201	42.6	86.4

The mean squared error of the reconstructed images decreases as the number of clusters increases. This is expected as the image should render closer and closer to the original as the number of colours allotted increases. The number of iterations increase on average as the number of clusters increase.

Below I will summarize my findings for the best initialization strategy:

- Image 1
  - K = 2
    - MSE → same
    - Visual → same
  - K = 3
    - MSE → same
    - Visual → same
  - K = 10
    - MSE → random (2)
    - Visual → same
  - K = 20
    - MSE → large distances
    - Visual → same
  - K = 40
    - MSE → large distances
    - Visual → large distances (the sun is more vibrant)
- Image 2
  - K = 2
    - MSE → same
    - Visual → large distances (the dark colour is a bit deeper giving it more depth)
  - K = 3
    - MSE → random (2)
    - Visual → large distances (two flowers are coloured pink/red compared to no distinct colour for the random initialization)

- K = 10
    - MSE → random (1) / random (2)
    - Visual → large distances (more distinct colours in the flowers)
- K = 20
    - MSE → large distances
    - Visual → large distances (the orange flowers is properly displayed as orange)
- K = 40
    - MSE → random (1)
    - Visual → large distances (closest in similarity to the original image – can see this in the petal details)

The differences in initialization methods are more apparent in the second image of the flowers. I chose this image in hopes to highlight some differences after not seeing many in image 1. The clear advantage over initializing the centers sufficiently distant from each other is seen in the flower colours. The differences between the pink and orange flower are very apparent to the viewers eye, however the difference is less stark to the program. By starting the centers far away from each other, the orange-adjacent and pink-adjacent pixels have a better chance at converging to two different centers.

The MSE for the random strategy was lower for K = 3, 10, and 40 for image 2 however this does not mean it produces a more visually appealing reconstruction. I conclude that large distances initialization produces images of same or better quality than random initialization.

References

[1] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006 (ISBN 9780387848570), available for free download

[2] "K-means++," Wikipedia, https://en.wikipedia.org/wiki/K-means%2B%2B (accessed Nov. 26, 2023).