Report TP4 Computer Vision

Created by: Phan Mạnh Tùng & Louis Choules

Created at: November 2022

Contents

List of Figure

List of Table

1. Implement K-means with **x** values being the intensity values in the image.

K-means clustering algorithm is an unsupervised algorithm and it is used in an image to segment the interest area from the background. We implement the algorithm in C with the following steps:

1. In the first step, we define k number of centroids and randomize RGB values for these centroids.
Code:

Table 1. generate_random_centroid

```
int* generate_random_centroid(int k_cluster,  int maxval){
  int i;
  int *cluster = malloc(sizeof(int) * k_cluster * pixel_val);
  for(i = 0; i < k_cluster * pixel_val; i++){
    cluster[i] = rand() % (maxval + 1);
  }

  return cluster;
}
```

2. We then create a function to calculate the Euclidean distance of each pixel in the image to the randomized centroids, given the formula: $E\_distance = \text{sqrt}((R - R_k)^2 + (G - G_k)^2 + (B - B_k)^2)$

Code:

Table 2. euclidean_distance

```
int euclidean_distance(gray *pixel, int k_cluster, int *cluster){
  double min = __INT_MAX__ * 1.0;
  int k_val = -1;
  int i, j;
  for(i = 0; i < k_cluster; i++){
    double temp = 0;
    for(j = 0; j < pixel_val; j++){
      temp += (pixel[j] - cluster[i* pixel_val + j]) * (pixel[j] -
cluster[i* pixel_val + j]);
    }
    temp = sqrt(temp);
    if(temp < min){
      min = temp;
      k_val = i;
    }
  }
  return k_val;
}
```

3. Assign each pixel to their group through the lowest Euclidean distance value to the centroid.

Code:

Table 3. map

```
int* map(int k_cluster, int *cluster){
  int i, j;
  int *temp = malloc(sizeof(int) * cols * rows);
  for(i = 0; i < cols; i++){
    for(j = 0; j < rows; j++){
      int index = i * (rows * pixel_val) + (pixel_val * j);
      temp[i * rows + j] = euclidean_distance(&image[index],
k_cluster, cluster);
    }
  }

  return temp;
}
```

4. After assigning all the pixels to their centroids. We calculate the new centroids by averaging all the pixels within one group.

Code:

Table 4. update_cluster

```
int *update_cluster(int k_cluster, int *mark){
  int *cluster = malloc(sizeof(int) * k_cluster * pixel_val);
  unsigned long long int *temp = malloc(sizeof(unsigned long long
int) * k_cluster * pixel_val);
  int *count = malloc(sizeof(int) * k_cluster);

  int i,j,k;
  for(i=0;i<k_cluster;i++)
    count[i]= 0;

  for(i=0; i<cols; i++){
    for(j=0; j<rows; j++){
      int k_val = mark[i*rows + j];
      count[k_val]++;
      for(k=0; k<pixel_val; k++){
        temp[k_val*pixel_val + k] += image[i * rows + (pixel_val * j)
+ k];
      }
    }
  }

  for(i=0; i < k_cluster; i++){
    for(j=0; j<pixel_val; j++){
      if(count[i] != 0)
        cluster[i*pixel_val + j] = (int) (temp[i * pixel_val + j] /
count[i]);
    }
```

```
    }

    return cluster;
}
```

5. Update the new centroids and re-do the process from step 3 with predefined n times

2. What is the influence of the initial values for region centers?

Different starting points for region centers will provide different results since after the first calculations, the centroid will use its own points in their cluster to decide where to move. The centroids' position will converge differently providing different starting points.

There are two major influences:

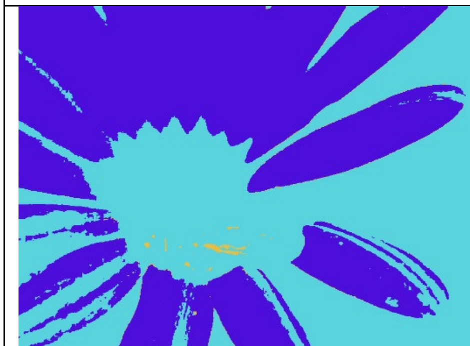
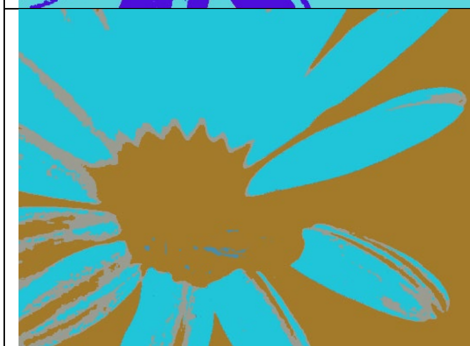
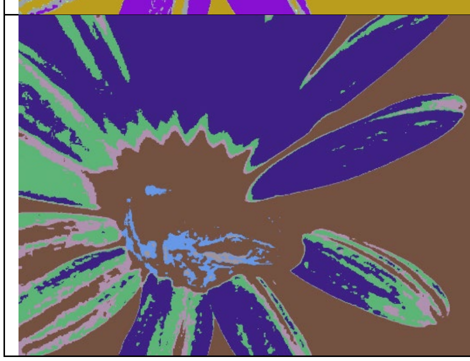
1. The boundary decision will be shifted slightly between different results since pixels at the edge are undecided of which cluster to go to
2. If we choose a starting point close to the frog color, we will identify the frog as a cluster

Image	Initial Value			Last Value			Itteration	K_Cluster
	R	G	B	R	G	B		
	176	198	4	189	89	159	1	3
	57	3	1	205	104	176		
	254	151	237	232	107	196		

3. What is the influence of the number of regions K?

Because the frog image only contains 3 major regions: the frog, the flower, and the background. Adding more centroids does not add more distinct regions but make pixels from one region with different level of intensity from their new colonies. Thus, increasing the unnecessary level of detail of the image as if we expect to identify more distinct regions/segments.

Image	Initial Value	Last Value	Itteration	K_Cluster

<table>
<tr><td></td><td>R</td><td>G</td><td>B</td><td>R</td><td>G</td><td>B</td><td></td><td></td></tr>
<tr><td rowspan="3"></td><td>232</td><td>190</td><td>4</td><td>206</td><td>101</td><td>176</td><td rowspan="3">100</td><td rowspan="3">3</td></tr>
<tr><td>13</td><td>33</td><td>141</td><td>231</td><td>107</td><td>196</td></tr>
<tr><td>207</td><td>190</td><td>231</td><td>213</td><td>101</td><td>182</td></tr>
<tr><td rowspan="4"></td><td>107</td><td>242</td><td>248</td><td>230</td><td>107</td><td>195</td><td rowspan="4">100</td><td rowspan="4">4</td></tr>
<tr><td>118</td><td>106</td><td>45</td><td>241</td><td>105</td><td>201</td></tr>
<tr><td>235</td><td>232</td><td>249</td><td>202</td><td>100</td><td>172</td></tr>
<tr><td>222</td><td>239</td><td>40</td><td>212</td><td>100</td><td>182</td></tr>
<tr><td rowspan="5"></td><td>250</td><td>201</td><td>24</td><td>186</td><td>87</td><td>156</td><td rowspan="5">100</td><td rowspan="5">5</td></tr>
<tr><td>51</td><td>66</td><td>27</td><td>215</td><td>102</td><td>184</td></tr>
<tr><td>151</td><td>66</td><td>167</td><td>230</td><td>107</td><td>196</td></tr>
<tr><td>92</td><td>34</td><td>62</td><td>240</td><td>106</td><td>201</td></tr>
<tr><td>172</td><td>114</td><td>180</td><td>204</td><td>101</td><td>174</td></tr>
<tr><td rowspan="6"></td><td>3</td><td>77</td><td>246</td><td>228</td><td>108</td><td>195</td><td rowspan="6">100</td><td rowspan="6">6</td></tr>
<tr><td>29</td><td>142</td><td>221</td><td>240</td><td>106</td><td>200</td></tr>
<tr><td>63</td><td>88</td><td>21</td><td>204</td><td>101</td><td>174</td></tr>
<tr><td>192</td><td>232</td><td>105</td><td>205</td><td>96</td><td>176</td></tr>
<tr><td>135</td><td>129</td><td>143</td><td>174</td><td>83</td><td>146</td></tr>
<tr><td>167</td><td>131</td><td>26</td><td>238</td><td>103</td><td>198</td></tr>
</table>

4. Consider now for **x** values both intensity and location in the image:

a. How does it change the results?

To take the location into account. We change the Euclidean distance formula as provided below:

$$\text{E\_distance} = \text{sqrt}((R - R_k)^2 + (G - G_k)^2 + (B - B_k)^2 + (x - x_k)^2 + (y - y_k)^2)$$

The new formula is arbitrary since x, y coordinates and RGB values are not normalized in the same value range. Therefore, the results obtained are not good.

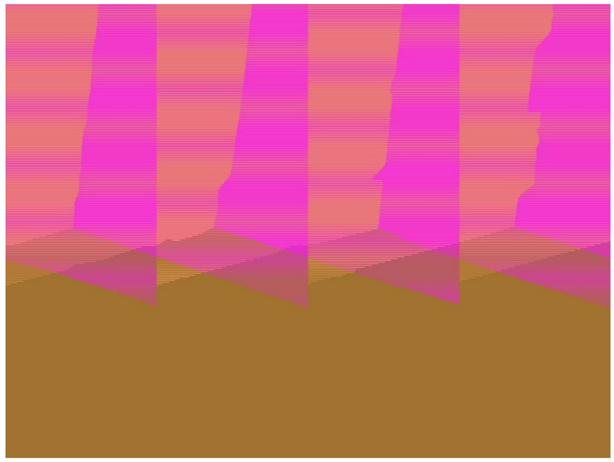


Figure 1. Cluster Image with color and coordinate

Table 5. Value for Figure above

Initial Cluster					Last Cluster					Itteration	K_cluster
X	Y	Red	Green	Blue	X	Y	Red	Green	Blue		
454	696	61	199	196	381	799	221	98	183	10	3
320	499	59	249	134	582	293	210	105	181		
109	420	94	216	52	181	289	227	111	196		

b. How can we balance the influence of colors and locations in the image?

To change the results, we need to balance out the weights colors, and locations in the formula. In other words, decide which factors are more important for our objectives and add more weight to them...

Cluster	X	Y	Red	Green	Blue
1	50	75	215	15	153
2	15	20	123	186	66

					With Calculating the the position too			We not counting the position		
X	Y	Red	Green	Blue	Cluster1	Cluster2	Clustering	Cluster1	Cluster2	Clustering
9	33	50	42	98	185.537	165.209	Cluster2	176.009	164.587	Cluster2
27	82	206	109	141	98.1784	149.77	Cluster1	95.1893	135.805	Cluster1
23	70	91	240	124	259.992	99.3378	Cluster2	258.538	85.4634	Cluster2
25	32	68	37	83	171.66	160.496	Cluster2	164.295	159.734	Cluster2
63	83	189	48	85	81.3757	173.303	Cluster1	79.9312	154.146	Cluster1
32	60	86	28	192	137.405	209.995	Cluster1	135.392	205.448	Cluster1
25	0	142	85	135	129.626	125.79	Cluster2	102.728	123.786	Cluster1
78	78	17	12	203	206.17	260.027	Cluster1	204.238	245.522	Cluster1
74	17	9	6	146	215.652	235.13	Cluster1	206.315	227.587	Cluster1
58	54	169	144	63	165.415	83.0301	Cluster2	163.881	62.3618	Cluster2
0	51	16	204	208	285.347	181.997	Cluster2	279.905	178.709	Cluster2

Figure 2. Example calculation clustering with position and without position