باسمه تعالی

دانشگاه صنعتی شریف

دانشکده مهندسی برق

درس سیگنال ها وسیستم ها

گزارش پروژه امتیازی

علی محرابیان

96102331

پرهام محمدی

96102342

استاد

دکتر کربلایی

در مسئله اول قصد داریم که که با در نظرگرفتن پیکسل های تصویر به عنوان گراف،بخش بندی تصویر را انجام دهیم.احتمال این که هر پیکسل جرئی از هواپیما یا آسمان باشد،بدین صورت در نظر می گیریم که پس از تبدیل تصویر به مقادیر double،پیکسل هایی که آبی رنگ و جز آسمان هستند،مقداری 100 و پیکسل های هواپیما که سفید هستند،مقادیری نزدیک 160 دارند.پس:

$$a_i = \frac{100}{260} \qquad b_i = \frac{160}{260}$$

برای به دست آوردن وزن بین رئوس،راه های مختلفی وجود دارد.اولین مورد،این است که اختلاف دو مقدار بالا را به عنوان وزن قرار دهیم.مورد دوم،که در عکس پایین شرح داده شده،استفاده از روابط عکس است.

$$A(i,j) = \begin{cases} \frac{\epsilon_1 + f(I_i, I_j)}{d(p_i, p_j)} & (I_i \neq I_j) \\ \frac{\epsilon_2}{d(p_i, p_j)} & (I_i = I_j) \end{cases} \tag{3.1}$$

in which I_i, I_j are intensities of pixels p_i and p_j. $d(p_i, p_j)$ is a distance measure of pixel p_i to p_j. A monotonically increasing function $f(I_i, I_j)$ calculates the difference between the intensity of pixel p_i and p_j. We could either connect every pixels pair or only connect all the neighboring pixels for less computational complexity. The intuition behind this graph generation approach is that human vision tends to focus on high-contrast places. Our approach emphasizes the close high-difference pixel pairs by giving the corresponding graph edges larger weights.

این مورد پیاده سازی شد ولی خروجی مطلوبی گرفته نشد.

To begin with, a pixel in the image is a vertex in the graph labeled in row-major order. These vertices are called pixel vertices. In addition, there are two extra vertices that serve as the source and the sink.

There are two types of edges in our graph. The first type is called n-links, which connects neighboring pixel vertices in a 4-neighboring system. The second type of edges is called t-links. These links connect the source or sink vertex with the pixel vertices.

The n-link edges must have weights carefully computed in order to reflect inter-pixel similarities. Concretely, we want the weight of an edge to be big with the two pixels are similar, and small when they are quite different. One idea is to let the weight be boundary penalty, a function that maps two pixel intensities to a positive integer. Let I_p be the brightness, or intensity, of the pixel vertex p. For any edge $(p,q) \in E$, the boundary penalty $B(I_p, I_q)$ is defined to be

$$B(I_p, I_q) = 100 \cdot \exp\left(\frac{-(I_p - I_q)^2}{2\sigma^2}\right)$$

با توجه به متن بالا،برای این که پیکسل های مشابه وزن بیشتر و پیکسل ها متفاوت وزن کمتری داشته باشند،تابع بالا را اعمال می کنیم.

با توجه به این که هدف ما max کردن عبارت زیر است،می توان آن را به صورت دیگری نوشت.

$$q(A,B) = \sum_{i\in A} a_i + \sum_{j\in B} b_j - \sum \quad p_{ij}$$

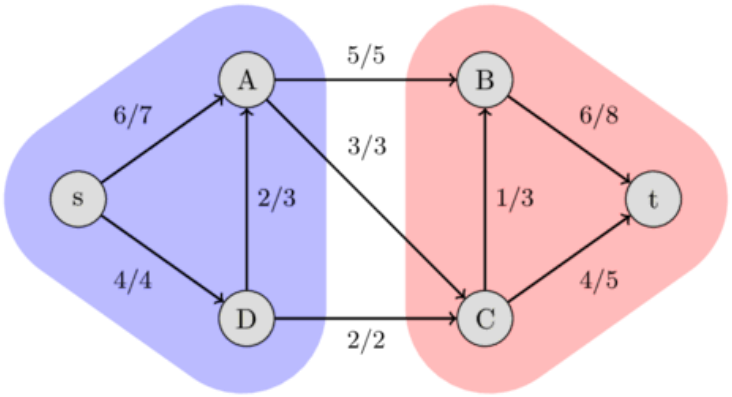
به جای دو عبارت اول،می توان عبارت زیر را جایگزین کرد.

$$q(A,B) = Q - \sum_{i\in A} b_i - \sum_{j\in B} a_j - \sum p_{ij}$$

پس هدف ما،معادل به کمینه سازی عبارت زیر است.

$$q'(A,B) = \sum_{i \in A} b_i + \sum_{j \in B} a_j + \sum p_{ij}$$

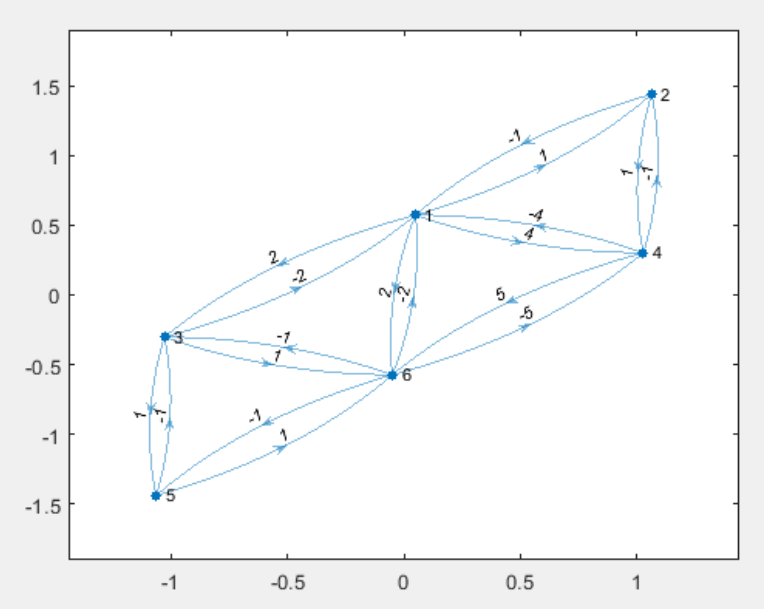
که این مورد معادل با پیدا کردن برش کمینه در گراف مد نظر است.در الگوریتم ford_fulkerson، ثابت شده است که پیدا کردن برش کمینه گرافی،معادل به پیدا max flow در گراف ما می باشد که مسئله معروفی است.



برای پیدا کردن flow max،وزن بین رئوس به صورت ظرفیت در نظر گرفته می شود و در هر مرحله مسیری از s به t درنظر می گیریم و max flow قابل عبور را یادداشت می کنیم.flow عبوری از مسیر همواره min ظرفیت مسیر کمتر است.در نهایت،با تشکیل residual graph در هر مرحله، مسئله هنگامی تمام می شود که دیگر مسیری وجود نداشته باشد.

به طور مثال،برای ماتریسی با وزن زیر،گراف نهایی به صورت زیر است.

```
p=[0 1 2 4 0 2;
   0 0 0 4 5 0;
   0 0 0 0 9 1;
   0 0 0 0 0 10;
   0 0 0 0 0 8;
   0 0 0 0 0 0];
```



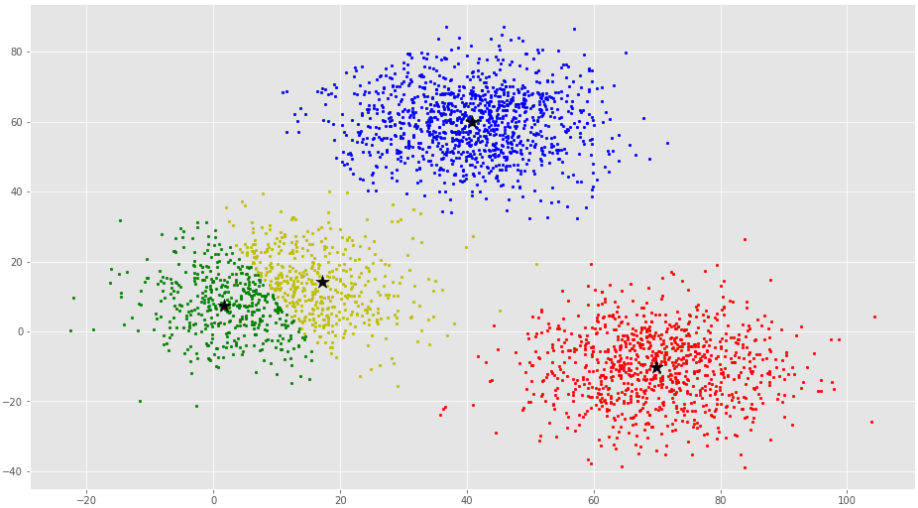
در هر مرحله،increment path پیدا شده و مقادیر جدید محاسبه می شوند.

در residual graph،تمام رئوسی که با مسیر مستقیم از راس اول قابل دسترسی هستند،دسته اول و بقیه رئوس دسته دوم را تشکیل می دهند.

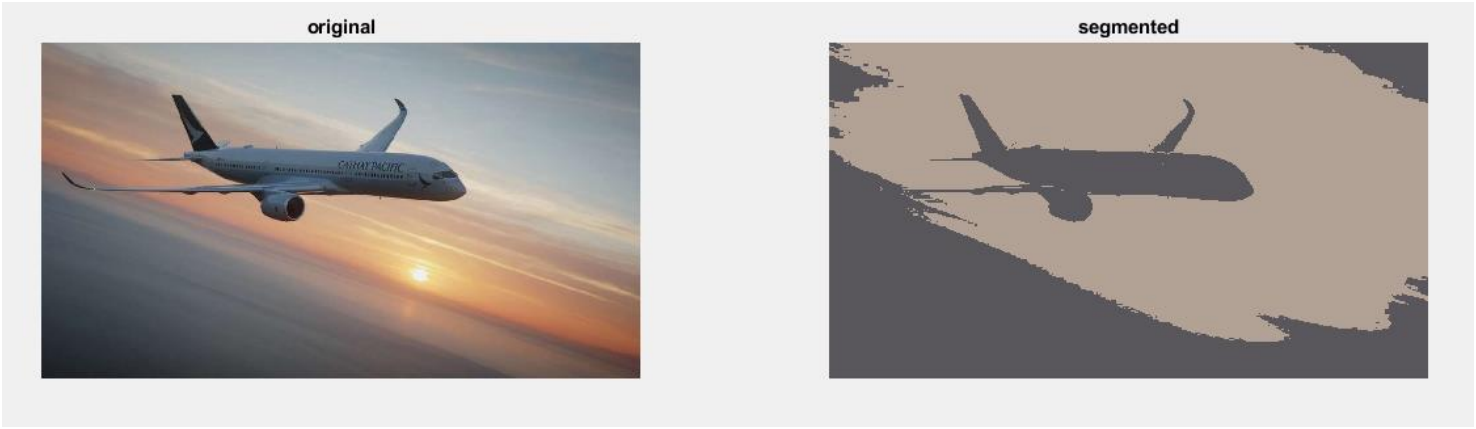
برای یافتن مسیر درهر مرحله،الگوریتم BFS را پیاده سازی کردیم.

K MEANS

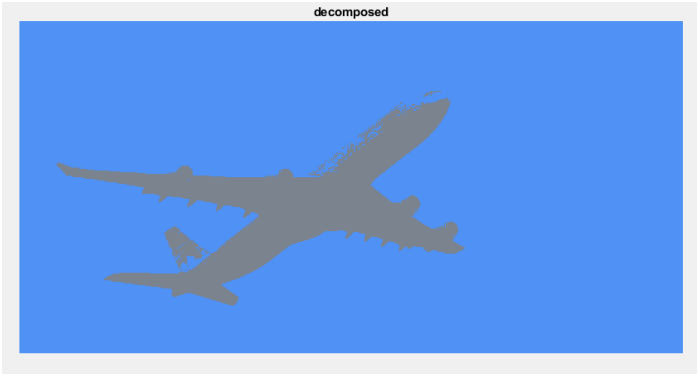
در توضیح این روش می توان گفت که برای خوشه بندی یک سری داده در فضای nبعدی،تعدادی نقطه تصادفی به اندازی تعداد خوشه های مد نظر در فضا در نظر می گیریم.در هر مرحله،فاصله نقاط تصادفی را از همه نقاط فضا پیدا می کنیم.با توجه به این که این نقاط به کدام نقاط فضا نزدیک اند، Min فاصله ها را پیدا کرده و نقطه موردنظر را به میانگین این نقاط انتقال می دهیم.عمل مورد نظر را چندین بار انجام داده تا دیگر تغییری در نقاط تصادفی ایجاد نشود.



برای اتمام شرط این روش،دو دیدگاه را می توان در نظر گرفت.اولی این که یک عکس با تعداد پیکسل های زیاد را امتحان می کنیم و تعداد مراحل را عوض کرده تا به خروجی مطلوب برسیم.دیدگاه دوم این است که یک treshhold در نظر گرفته و اگر فاصله به این مقدار رسید،الگوریتم را متوقف می کنیم.ما از اولی استفاده کردیم.



در این مورد چون رنگ ها به نزدیک است،خوشه بندی درست تشخیص داده نمی شود.



در این مورد خروجی به درستی خوشه بندی می شود.

مورد اول،هزینه ای است که به دلیل کاهش محاسبات و درنظر گرفتن تنها ویژگی های رنگ رخ می دهد.

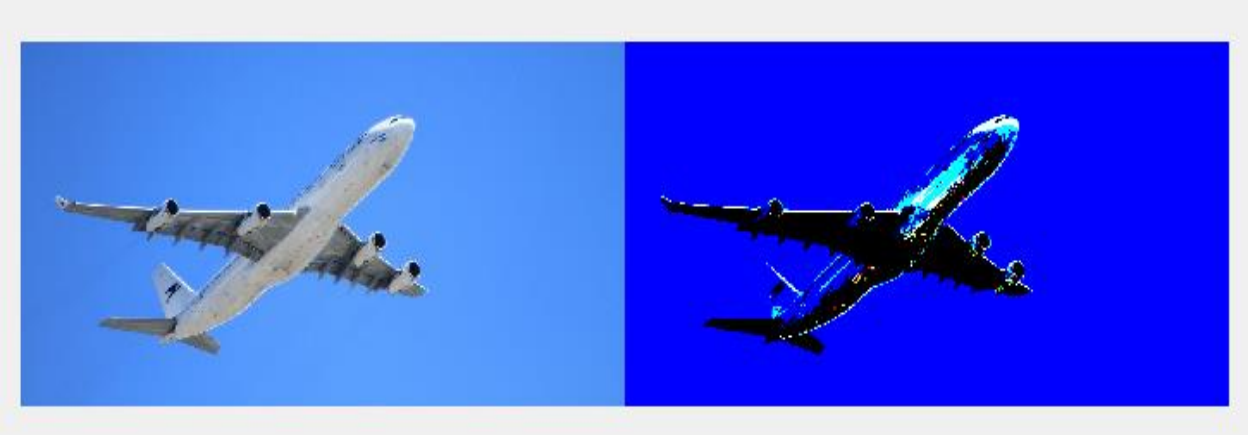
OTSU

الگوریتم بدین صورت است که در حالت ساده،به دنبال پیکسل های مشابه می گردیم و عکس را به دو کلاس تقسیم می کنیم.سپس تعداد پیکسل هارا بر کل آنها تقسیم کرده و احتمال بودن در هرکلاس را حساب می کنیم.بدیهی است که در این مورد باید treshhold قائل شویم.حال به دنبال treshhold هستیم که واریانس زیر را مینیمم کند.

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t)$$

W ها برابر با احتمال ها در treshhold درنظر گرفته شده وσ ها نیز واریانس هر کلاس هستند.

به کمک متلب خروجی هارا به دست می آوریم.



در روش اول دو مشکل داریم.اول این که برای یک تصویر با اندازه n*n،ماتریس وزنی به اندازه n^4 خانه می خواهیم که برای عکس های عادی هم چنین حافظه ای غیر ممکن است.در نتیجه باید عکس را بسیار کوچک کنیم تا بتوانیم به خروجی برسیم.

```
Elapsed time is 73.067446 seconds.
```

روش دوم،روش بهتری است،به دلیل این که پیاده سازی ساده تری دارد.ولی این مشکل که چه زمانی الگوریتم را متوقف کنیم وجود دارد.

```
Elapsed time is 68.636978 seconds.
```

در روش آخر نیز اگر تصویر foreground نسبت به background کوچک باشد،هیستوگرام به دست آمده دارای یک ماکزیمم خواهد بود و تشخیص دو جز از یکدیگر سخت می شود.