

**بسمه تعالی**

**دانشگاه صنعتی شریف**

**دانشکده علوم کامپیوتر**

**Image processing - دکتر مصطفی کمالی**

**HW4 q4 report**

**سوال 4 - segmentation:**

این سوال را میخواهیم با ترکیب دو روش حل کنیم ، ترکیب دوروش oversegmentation با روش felzenszwalb و سپس استفاده از بانک فیلتر ، علت اینکه از felzenszwalb استفاده میکنیم اینست که اندازه superpixel ها غیر یکنواخت میباشد و لذا بخش بزرگی از زیمنه یک بخش میشود ولی برای مثال روش slic به ما تعداد زیادی بخش میداد که یکنواخت بودند و در این مسیله خوب نبود . برای اینکه کد سریع شود اول عکس را به 25% اندازه اصلی کوچک میکنیم سپس با استفاده از تابع آماده felzenszwalb آنرا قطعه بندی میکنیم .

حال میخواهیم پاسخ قسمت های عکس را به فیلترهای بانک فیلتر بدست آوریم ، برای اینکار از فیلترهای RFS استفاده میکنیم ولی فقط scale اول آنرا در نظر میگیریم که بعدا میخواهیم بین rotation های مختلف مقدار ماکزیمم را بگیریم که میشود همان فیلترهای MR4 ، اندازه همه فیلتر ها هم 11 در 11 انتخاب شده ؛ کلیه اعداد با تست های متعدد و آزمون و خطا بدست آمده اند. در روش MR4 دو فیلتر گوسی و لاپلاس گوسی داریم که برای بدست آوردن response عکس به این دو فیلتر با استفاده از توابع آماده ابتدا با σ =10 و تابع cv2. GaussianBlur پاسخ عکس به فیلتر گاوسی را بدست میاوریم و سپس با استفاده از تابع cv2. Laplacian پاسخ به فیلتر لاپلاس بدست می‌آید.

حال میخواهیم برای 7 زاویه از 0 تا 180 درجه و با اخلاف 30 درجه ، فیلترهای مشتق اول و دوم گاوسی را داشته باشیم که σx=1 , σy=3 انتخاب شده اند . میدانیم که این فیلترها جدایی پذیرند لذا ابتدا دو بردار یک بعدی با تابع کرنل ساز گوسی یعنی cv2.getGaussianKernel میسازیم که یکی برای جهت x با زیگما x و دیگری برای y است سپس از تیوری میدانیم که مشتق اول گاوسی هر کدام به صورت زیر است که G(x) همین کرنل گاوسی بود ( یعنی همان گاوس عادی بدون مشتق ) : لذا مشتق مرتبه اول به صورت زیر است :

برای جهت y هم همین است ، حال برای دو جهت را با استفاده از تابع np.outer در هم ضرب کرده و یک فیلتر مشتق اول گاوس با اندازه 11×11 بدست میاید .

برای مشتق دوم گاوس هم مثل همین روش برای هر کدام از جهت های x,y بدست میاید : (بعدا هم نرمال میکنیم)

که G”x \* transpose(G”y) فیلتری که ما میخواهیم یعنی مشتق دوم گاوس را میدهد.

حال که مشتق اول و دوم گاوس بدست آمد ، برای زوایای مختلف با استفاده از تابع cv2.getRotationMatrix2D یک ماتریس دوران میسازیم و با cv2.warpAffine بر این دو فیلتر اعمال میکنیم تا مشتق اول و دوم گاوس برای دو جهت بدست آید . سپس مقدار مشتق دوم گاوس را تقسیم بر مجموع درایه هایش هم میکنیم که جمع درایه ها یک شود.

حال یک دیکشنری به اسم RFS داریم که این 14 فیلتر در آن قرار دارند ( البته روش ما کاملا MR4 نیست ولی منطق همان است ) . این دیکشنری کلید هایی به صورت (id1,id2) دارد که id1 دو مقدار 0 و 1 را میگیرد که نشان میدهد مشتق گاوس اول یا دوم ایت و id2 مقدار 0 تا 6 است که مشخص میکند در کدام زاویه است .

حال فیلترهای گاوس مشتق اول در زوایای مختلف را بر عکس اعمال کرده و هرکدام پاسخ بیشینه داشت انتخاب شده و در درایه اول آرایه response قرار میگیرد ؛ همینکار برای مشتق دوم گاوس انجام شده و در درایه دوم response میشیند ، در درایه سوم و چهارم response هم پاسخ به فیلتر گاوسی و لاپلاس قرار میگیرد .

حال در ابتدا خودمان به صورت دستی یکی دو پرنده را جدا کرده و پاسخ فیلتر ها را برای آنها در نظر گرفیتم ، در نهایت هم برای هر چهار فیلتر یک بازه مجاز درمیاید که با تست کردن مقدار آن جوری تنظیم شده که نتیجه بهتری بدهد و با عوض کردن هر کدام از اعدادی که در کد قرار داده شده نتیجه به شدت عوض میشود ( مثل عوض کردن scale شدن اولیه عکس یا اندازه فیلتر ها یا سیگماها یا پارامترهای felzenszwalb)

سپس برای هر کدام از segment هایی که felzenszwalb بدست آورده ( که حدودا 200 ناحیه میشود با توجه به پارامترهایی که تنظیم کردیم ) بررسی میشود که میانگین پاسخ آن ناحیه به هر کدام از فیلتر ها به چه صورت است ، اگر این میانگین برای هر چهار فیلتر در بازه های مجاز مربوطه قرار داشت ، این ناحیه را به عنوان پرنده انتخاب میکنیم.برای همه ناحیه ها این کار را میکنیم و label ناحیه های انتخاب شده در آرایه birds قرار دارد.

سپس یک ماتریس خالی با درایه های صفر به اندازه عکس میسازیم به نام Result و سپس در این ماتریس بخش هایی که در birds انتخاب شدند را از عکس اصلی قرار میدهیم سپس از آن لاپلاسین میگیریم که مرز های پرنده ها بدست آید سپس با استفاده از توابع مورفولوژی برای نمایش بهتر و اینکه مرزها واضح تر شود ابتدا روش closing و سپس deliate را انجام میدهیم و این مرزها را در تصویر اولیه میکشیم و نتیجه را ذخیره میکنیم .