دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

مدل های گرافی احتمالاتی

گزارش کار تمرین اول

آیدین روزبه - ۹۹۳۱۳۰۳

استاد:

دکتر احمد نیک‌آبادی

دانشکده مهندسی کامپیوتر

تیر ۱۴۰۳

فهرست مطالب

لیست کد ها و کاربرد هر کدام:

ابتدا به توضیح هر کدام از فایل های استفاده در این شبیه‌سازی می‌پردازیم، برای دیباگ کردن راحت‌تر و استفاده مجدد، تلاش شده است که تا حد امکان هر فانکشن به صورت یک ماژول جدا نوشته شود. همچنین در تمام بخش های همه فایل ها کامنت گذاری مناسب انجام شده تا هیچ ابهامی باقی نماند و گزارش‌کار کوتاه تر شود. تمام کد ها در متلب نوشته شده اند:

فایل image_to_label:

با گرفتن تصویر سیاه و سفید، برچسب هر پیکسل را مشخص می‌کند.

فایل label_to_image:

با گرفتن برچسب ها، تصویر سیاه و سفید مربوطه را می‌سازد.

فایل get_accuracy:

درصد تشابه (درایه به درایه) دو ورودی را محاسبه می‌کند.

فایل sim_ann_cost:

هزینه یک state (هر state در واقع یک ماتریس از برچسب هاست) را با توجه به میزان نویز اولیه و همچنین همسایگی ها محاسبه می‌کند

فایل get_delta_energy:

از آنجایی که در الگوریتم Simulated Annealing میزان اختلاف انرژی دو state (یا حالت) مهم است، این کد صرفا این اختلاف را محاسبه می‌کند تا از محاسبات اضافه جلوگیری شود و سرعت کد بهبود یابد.

فایل get_neighbor_4 و get_neighbor_8:

به عناون یک ساب‌ماژول از آن در دو تابع قبل استفاده شده و بخش مرتبط با همسایگی از تابع هزینه (انرژی)را حساب می‌کند (یعنی فقط ترم دوم)

فایل give_points:

برابری دو همسایه را مثل تابع دلتای کرونکر بررسی می‌کند. از آن در تابع قبلی استفاده شده است

فایل new_state:

یک حالت تصادفی جدید برای استفاده در الگوریتم simulated annealing ایجاد می‌کند. (لازم به ذکر است که ماتریس برچسب ها را به طور منظم از یک گوشه بررسی می‌کند. انتخاب کاملا تصادفی برچسب ها – که البته حالت همسایه حساب نمی‌شود – به نفع ما نیست و احتمال پیدا کردن یک حالت با انرژی کمتر در هر iteration کم می‌شود). با این منطق البته، زمان اجرای الگوریتم به صورت کامل، شدیدا افزایش خواهد یافت.

فایل get_superpixelized:

تصویر ورودی را به صورت superpixel در می‌آورد. با تعداد سوپرپیکسل های مشخص. ابعداد ورودی و خروجی یکی است. الگوریتم SCIL به صورت نهفته در متلب وجود دارد و این الگوریتم توسط دانشجو پیاده سازی نشده است.

فایل part_AB:

پیاده سازی خواسته های مسئله در پارت اول بخش الف و ب

فایل part_CD:

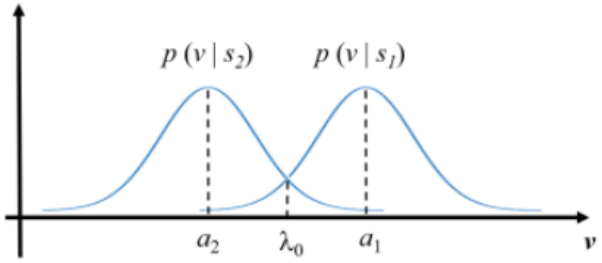
پیاده سازی خواسته های مسئله در پارت اول بخش ج و د

فایل part_AB_sp:

پیاده سازی دوباره خواسته های پارت اول بخش الف و ب ولی این بار با سوپرپیکسل

پارت اول – شبیه سازی بخش الف و ب

در اینجا یک مسئله classification ساده مشابه مباحث مخابرات دیجیتال و ماشین لرنینگ مطرح است که صرفا باید برچسبی را انتخاب کرد که بیشترین احتمال را با توجه به ورودی دارد (اصطلاحا MAP – که در اینجا چون ما توزیع اولیه برچسب های ورودی را یکنواخت فرض کرده‌ایم، این تخمین عملا تخمین ML خواهد بود). لازم به ذکر است که با توجه به تابع توزیع نویز ورودی (گاوسی با میانگین صفر) و همچنین فرض توزیع اولیه یکنواخت، مرز ناحیه تصمیم، در مراکز بین مقادیر صحیح (منظور از صحیح، valid است) که در فاصله برابر از طرفین قرار دارد. مثل شکل زیر (برای حالت باینری):



لازم به ذکر این شرایط برای مرز ناحیه تصمیم در مورد هر توزیع نویز دیگری که متقارن و با میانگین صفر باشد، برقرار است که البته در اینجا از اثبات آن صرف نظر می‌کنیم.

در فایل part_AB.m ابتدا به تصویر سه نویز با انحراف معیار های 0.01 و 0.1 و 0.3 به تصویر سیاه‌وسفید اضافه می‌شود. بهترین تخمین برای برچسب های این تصاویر به دست می‌آیند و سپس از روی این برچسب ها دوباره تصاویر بازسازی می‌شوند که در این صورت

نتیجه شبیه‌سازی در نمودار زیر ترسیم شده است. هچنین میزان دقت این تخمین برای هر سه نویز (بر حسب درصد) به شرح زیر است

```
Command Window
  Accuracy for different noise level are respectively (%):
     100

    99.062725723036024

    70.681575221750307

fx >>
```

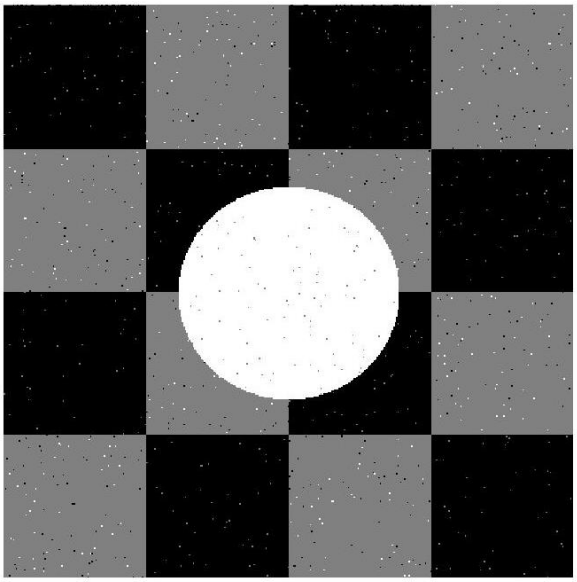
Original Image
Original Image
Original Image
Image with Noise 1
Image with Noise 2
Image with Noise 3
Reconstructed image
Reconstructed image
Reconstructed image

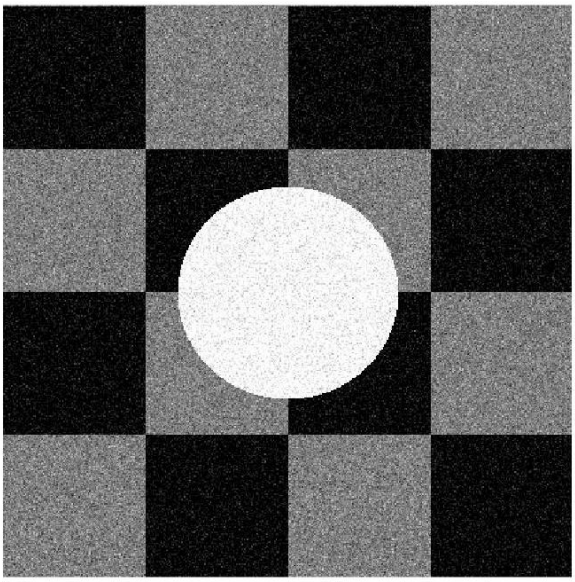
پارت اول – شبیه سازی بخش ج و د

در این بخش به منظور پیدا کردن بهترین تخمین از برچسب ها، تابع هزینه ای که روی این برچسب ها تعریف می‌شود (و مشابه تعریف دو دسته فاکتور است). در واقع اگر از بخش مرتبط با تاثیر همسایگی صرف نظر شود، در بهترین حالت مشابه همان تخمین ML در بخش قبل عمل خواهد کرد. با توجه به این که همسایگی مشابه، به معنای احتمال رخداد بالاتر و متعاقبا انرژی کمتر است، می‌توان گفت مقدار beta باید منفی باشد (تا در صورت رخ دادن مقادیر همسایگی مشابه بیشتر، مقدار تابع انرژی کاهش یابد). همچنین برای رفتن از یک حالت به حالت همسایه، فقط مقدار یکی از برچسب ها را تغییر می‌دهیم و این کار را با نظم انجام می‌دهیم تا همه برچسب ها به حالت بهینه برسند. برای کاهش میزان محاسبات به جای محاسبه انرژی در هر دو حالت فعلی و حالت جدید. مستقیما اختلاف این دو انرژی محاسبه می‌شود. نتیجه شبیه‌سازی برای این تصویر با این مشخصات به شرح زیر است:

$$Noise\ Standard\ deviation = 0.1$$
$$T = 170000$$
$$a = 0.9999$$
$$Number\ of\ neighbors\ to\ consider = 8$$
$$Beta = -20$$

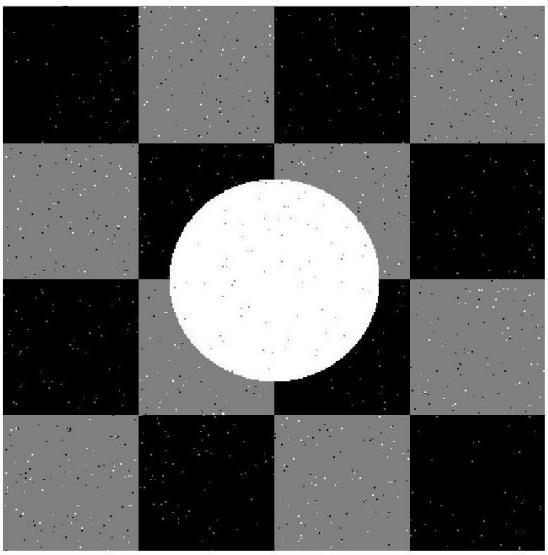
قبل و بعد از تقسیم بندی معنایی با این روش:

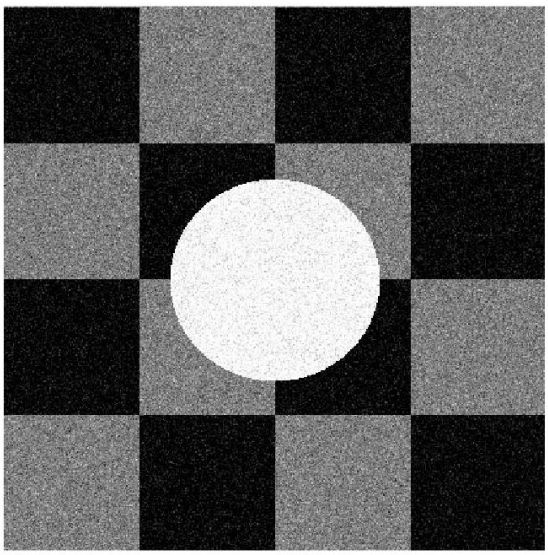




پارت اول- پاسخ بخش های ه، و، ز، ح

ه) برای سنجش تاثیر این همسایگی، شبیه سازی بخش قبل را با همان تنظیمات قبلی اجرا می‌کنیم با این فرق که تعداد همسایگی را از ۸ به ۴ کاهش می‌دهیم. نتیجه شبیه سازی به صورت زیر خواهد بود. انتظار می‌رود در SNR های بالا این تاثیر روی نتیجه کلی مثبت باشد زیرا احتمال رخ دادن خطا در پیکسل های مجاور بیشتر می‌شود، اما در SRN های پایین بهتر است مقدار همسایگی زیاد باشد، زیرا به بیانی پیکسل های خطا sparse تر هستند.





و) هر مقدار beta به طور مطلق بیشتر باشد، این الگوریتم اهمیت بیشتری به بحث همسایگی می‌دهد. با شبیه سازی برای مقادیر beta برابر با 1- و 40- می‌توان این تاثیر را مشاهده کرد:

ز) در حالت کلی در الگوریتم Simulated Annealing مقدار دهی اولیه دستی از آنجایی که حالت اولیه را به Global Minimum نزدیک‌تر می‌کند، احتمال آن که پس از انتخاب یک حالت همسایه جدید به مینیمم نزدیک‌تر بشویم نیز بیشتر است.

ح) هرچه دمای ابتدایی بیشتر باشد، همانند همان آهن گداخته، زمان بیشتری باید بگذرد تا دما کاهش یابد و در نتیجه الگوریتم با احتمال بیشتر به جواب بهینه همگرا می‌شود.

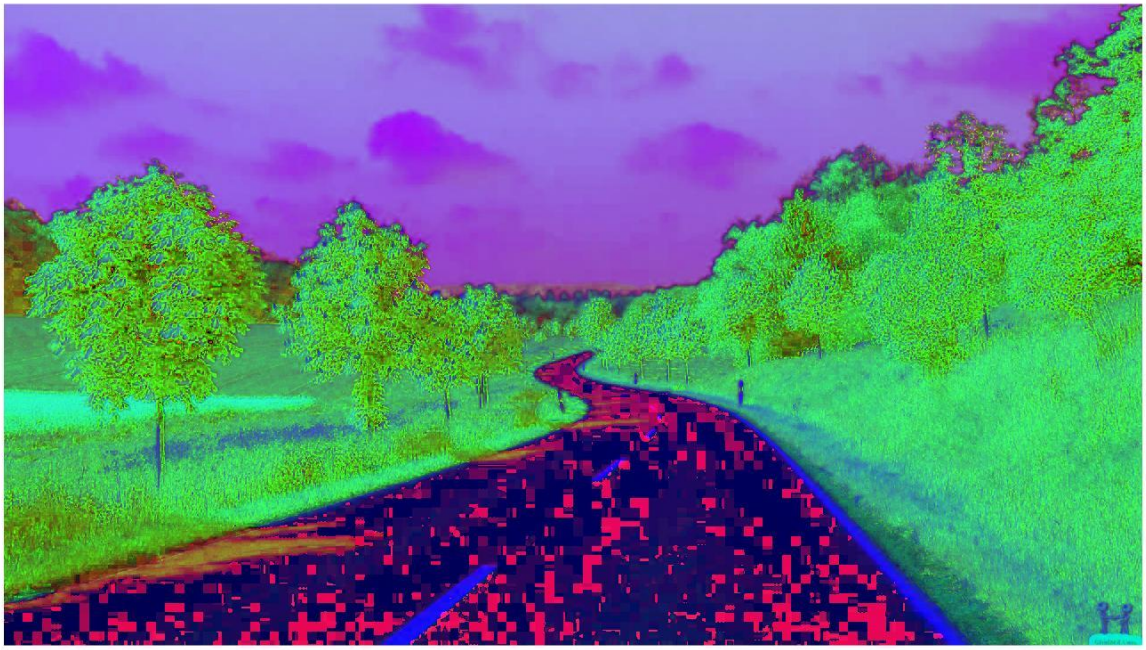
پارت دوم- شبیه سازی بخش الف

فایل part_2A تقریبا مشابه فایل part_CD بوده و صرفا ورودی ها و تنظیمات شبیه‌سازی تغییر کرده اند. انتظار می‌رود که که فضای خاکستری در این تصویر به دلیل اختلاف بسیار کم بخش های مختلف (در تصویر خاکستری) خروجی مطلوبی نداشته باشد، به همین دلیل برای کاهش محاسبات و صرفا گرفتن نتیجه‌ی مدعا، سایز تصویر را به 0.4 سایز اولیه کاهش می‌دهیم. خروجی این شبیه‌سازی با تنظیمات زیر، نشان داده شده است:

$$Noise\ Standard\ deviation = 0.1$$
$$T = 600'000$$
$$a = 0.99999$$
$$Number\ of\ neighbors\ to\ consider = 8$$
$$Beta = -20$$

قبل و بعد از اجرای الگوریتم تفاوت بسیار بسیار ناچیز است.

به همین دلیل به سراغ Hue می‌رویم:

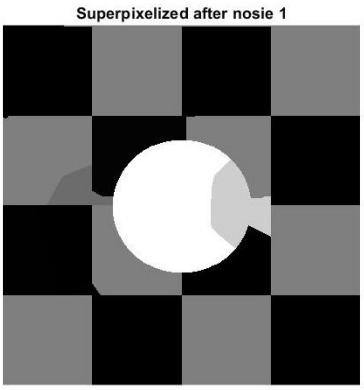


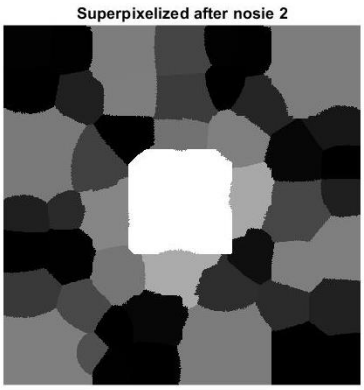
واضح است که در این فضا، تقسیم بندی بسیار بهتر صورت خواهد گرفت

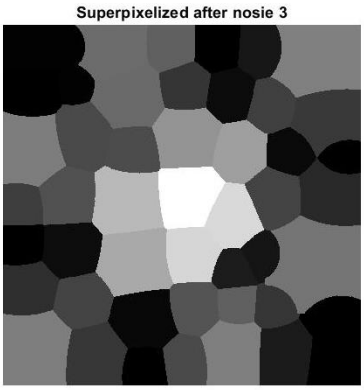
با استخراج لایه دوم از سه لایه این تصویر، به شکل زیر می‌رسیم که مشخص است از آن می‌توان برای تقسیم بندی معنایی استفاده کرد:

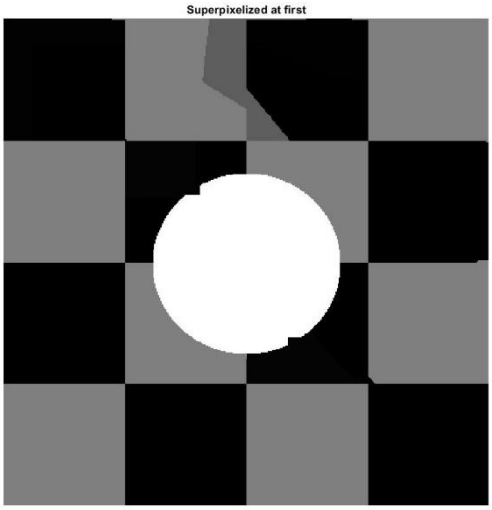
پارت اختیاری – superpixel در پارت اول بخش الف الی د

در فایل part_AB_sp برای برچسب زدن، از سوپرپیکسل به دو شکل استفاده شده است. در حالت اول ابتدا نویز اضافه شده و سپس عملیات تبدیل پیکسل به سوپرپیکسل انجام شده، و در حالت دوم، برعکس. نتیجه شبیه سازی برای هر دو حالت به صورت زیر است. لازم به ذکر است که استفاده از superpixel بار محاسباتی را به شدت کاهش می‌دهد.

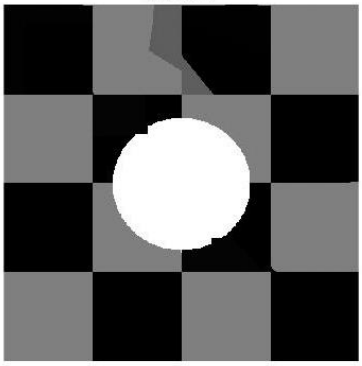

Superpixelized after nosie 1


Superpixelized after nosie 2

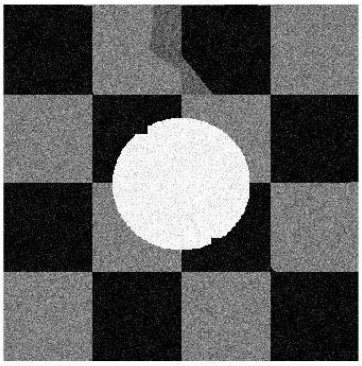

Superpixelized after nosie 3


Superpixelized at first

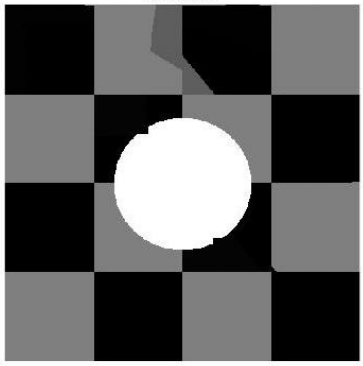
Noise 1



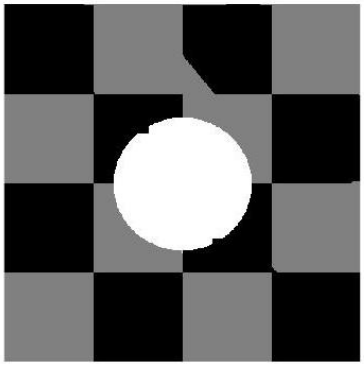
Noise 2



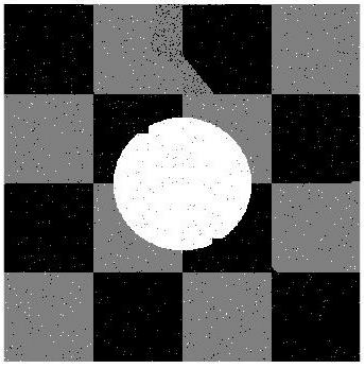
Noise 3



Labeled 1



Labeled 2



Labeled 3