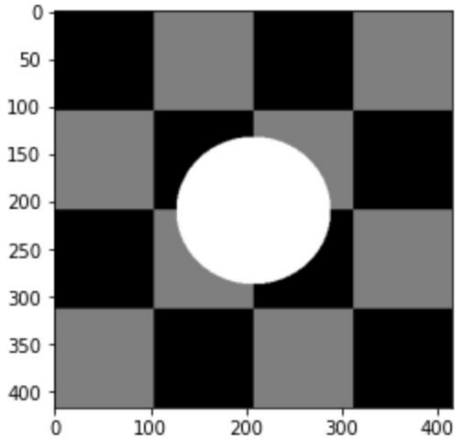
بخش ۱

هدف این بخش دینویز کردن از طریق قطعه‌بندی تصویر است. تصویر اولیه به صورت زیر است:



قسمت الف

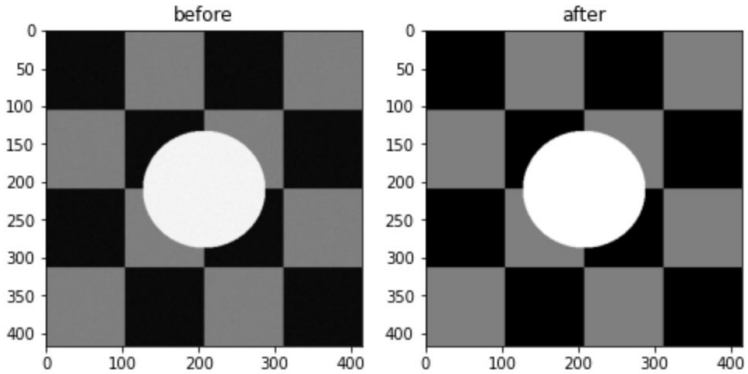
در این قسمت تابع make_noisy طراحی شد. این تابع بر اساس میانگین و واریانس و به صورت گوسین، نویز را به تصویر اولیه اضافه می‌کند.

همچنین دسته‌بند بیز ساده نیز طراحی شد تا به کمک آن تصاویری که در مرحله قبل نویزی به آن‌ها اضافه شد دینویز شود. نتایج بیز ساده در قسمت‌های بعد نمایش داده می‌شود.

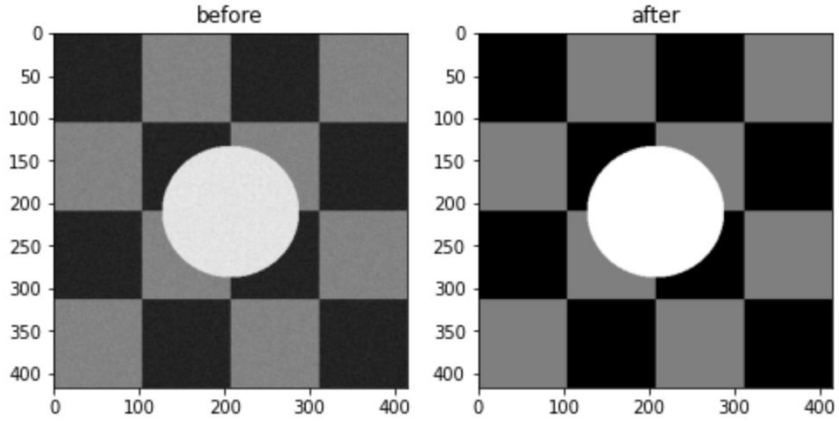
قسمت ب

نویز گوسی با میانگین صفر و با سه واریانس مختلف به تصویر اولیه اضافه شد و به کمک دسته‌بند بیز ساده قطعه‌بندی شد:

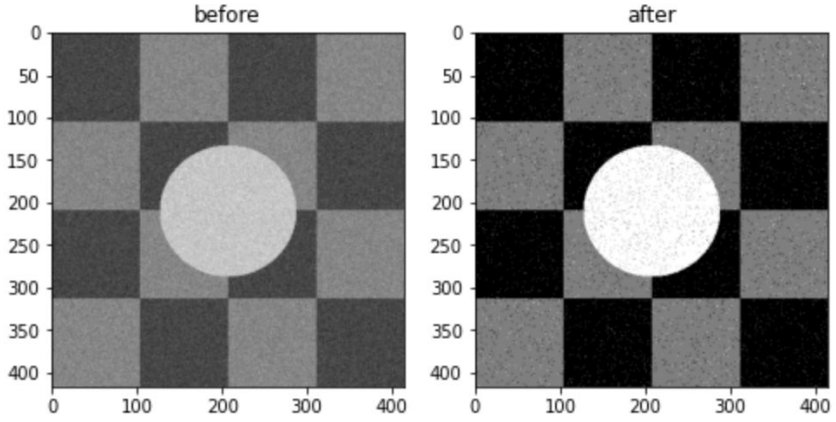
واریانس ۱۰:



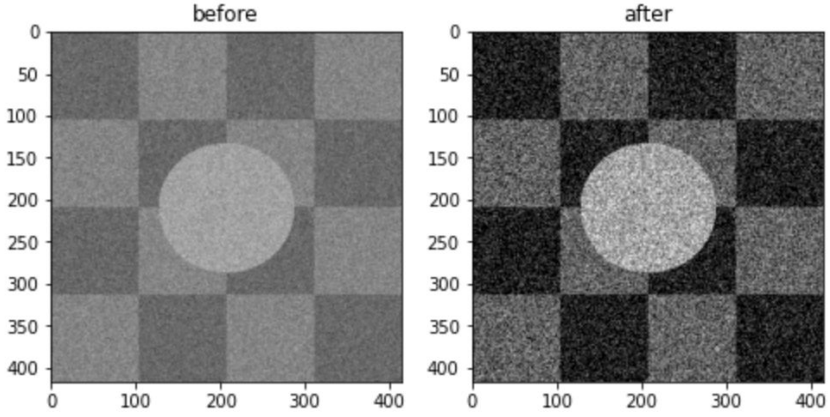
واریانس ۱۰۰:



واریانس ۱۰۰۰:



واریانس ۱۰۰۰۰:



همانطور که مشاهده می‌شود، دسته‌بند بیز ساده همواره باعث بهبود تصویر نویزی اولیه می‌شود. عملکرد این قطعه‌بند برای واریانس‌های نسبتا کوچک مانند ۱۰ و ۱۰۰ قابل توجه است و تقریبا به صورت کامل و درست قطعه‌بندی را انجام می‌دهد. ولی با افزایش واریانس به مقادیری مانند ۱۰۰۰ و ۱۰۰۰۰، نتایج به خوبی قبل نیست و نمی‌تواند تصویر را به حالت اولیه برگرداند.

قسمت ج

در این قسمت نیاز است تا از MRF برای قطعه‌بندی استفاده شود. به این صورت که همسایه‌های هر پیکسل در تعیین کلاس آن پیکسل نقش دارند. همچنین برای بهینه‌سازی آن، از روش Simulated Annealing با تابع انرژی زیر استفاده می‌کنیم:

$$U(\omega) = \sum_{s} \left(\log\left(\sqrt{2\pi\sigma_{\omega_s}}\right) + \frac{(f_s - \mu_{\omega_s})^2}{2\sigma_{\omega_s}^2} \right) + \sum_{s,r} \beta\delta(\omega_s, \omega_r)$$

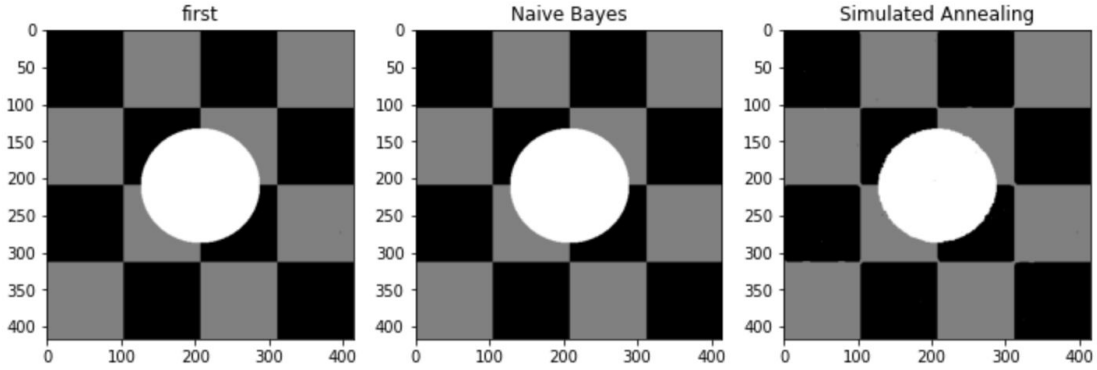
به این منظور کلاسی به اسم SimulatedAnnealing طراحی شد که پارامترهای آن را لیبل‌ها و پارامتری‌های تابع انرژی تشکیل می‌دهند.

قسمت د

حال روش SA را با روش بیز ساده مقایسه می‌کنیم. برای این کار از تصاویر نویزی با واریانس‌های مختلف استفاده می‌کنیم.
پارامترهای مورد استفاده در روش SA در این سوال به صورت betha=100, iterations=1000000, 'scheduler='exponential_schedule است.
نتایج تقریبا به صورت بصری قابل مقایسه است. برای نشان دادن نتیجه به صورت کمی، هر کدام از تصاویر تولید شده با تصویر درست اولیه مطابقت داده شد و درصد شباهت پیکسل‌ها را نیز گزارش شد.

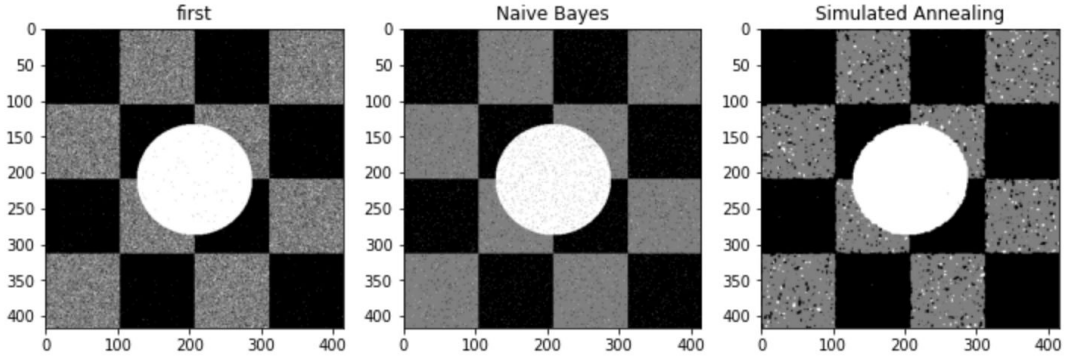
واریانس ۱۰۰:

```
Noisy Match Percent:  0.9999884429805553
Naive Bayes Match Percent:  1.0
Simulated Annealing Match Percent:  0.996290196758256
```



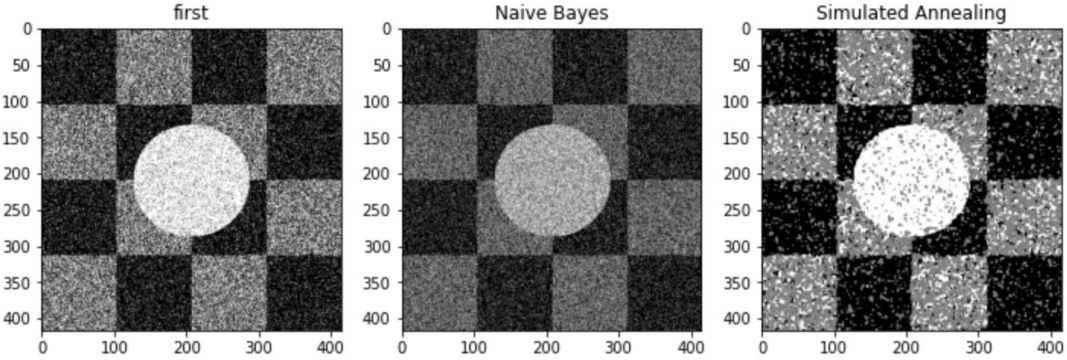
واریانس ۱۰۰۰:

```
Noisy Match Percent:  0.9177082430441189
Naive Bayes Match Percent:  0.9714079338938488
Simulated Annealing Match Percent:  0.9487561757822658
```



واریانس ۱۰۰۰۰:

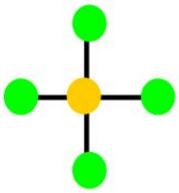
```
Noisy Match Percent:  0.5905232440553582
Naive Bayes Match Percent:  0.6699141891306232
Simulated Annealing Match Percent:  0.7849354251538528
```

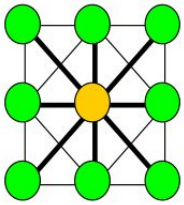


همانطور که قابل مشاهده است، در واریانس‌های پایین بیز ساده با اطمینان و درستی بیشتری قطعه‌بندی را انجام می‌دهند و SA به دلیل داشتن randomness، به خوبی بیز ساده عمل نمی‌کند. اما هرچه که واریانس افزایش می‌یابد، SA به کمک پیکسل‌های همسایه، نتایج بهتری را نسبت به بیز ساده از خود نشان می‌دهد.

قسمت ه

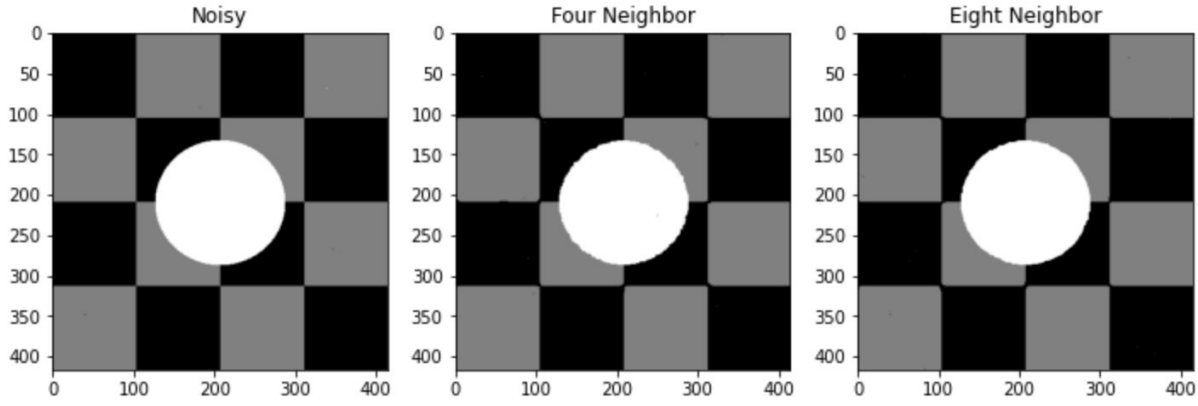
در این قسمت SA را با دو حالت چهار همسایگی و هشت همسایگی اجرا می‌کنیم. در این دو حالت همسایه‌های تاثیرگذار هر پیکسل به صورت زیر است:





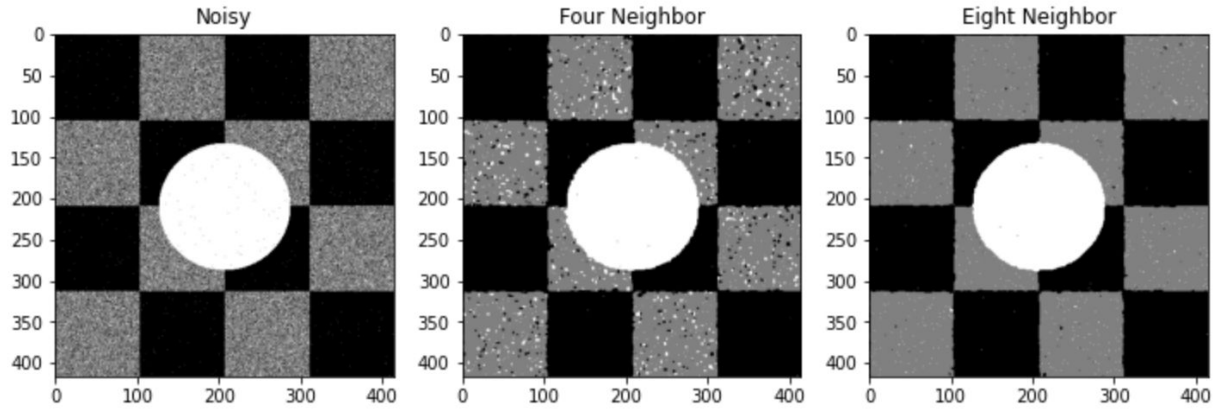
واریانس ۱۰۰:

```
Noisy Match Percent:  0.9999768859611107
Four Neighbor Match Percent:  0.9968391551818786
Eight Neighbor Match Percent:  0.99770593164023
```



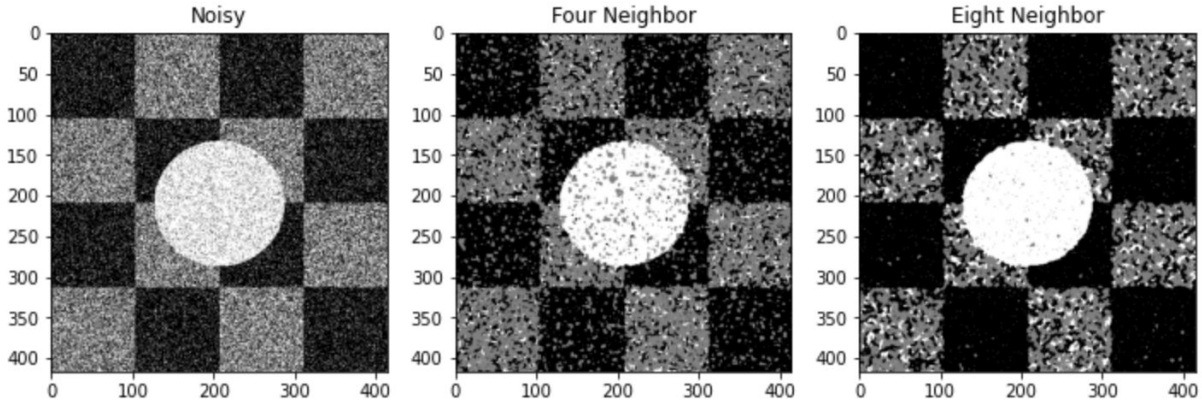
واریانس ۱۰۰۰:

```
Noisy Match Percent:  0.9192684406691515
Four Neighbor Match Percent:  0.9524833145531767
Eight Neighbor Match Percent:  0.9811909508537748
```



واریانس ۱۰۰۰۰:

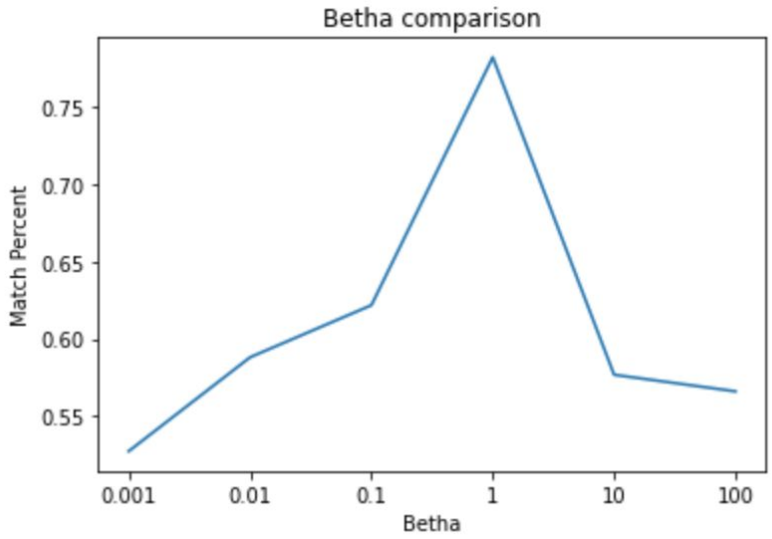
```
Noisy Match Percent:  0.5925399439484557
Four Neighbor Match Percent:  0.7845598220219006
Eight Neighbor Match Percent:  0.793152465979024
```



همانطور که مشاهده می‌شود، استفاده از هشت همسایه نتایج بهتری را نسبت به چهار همسایه می‌دهد.

قسمت و

هدف این قسمت، آزمایش مقادیر مختلف betha و پیدا کردن بهترین مقدار آن است. بتاهای مختلف امتحان شدند و خروجی آن‌ها با هم مقایسه شد:

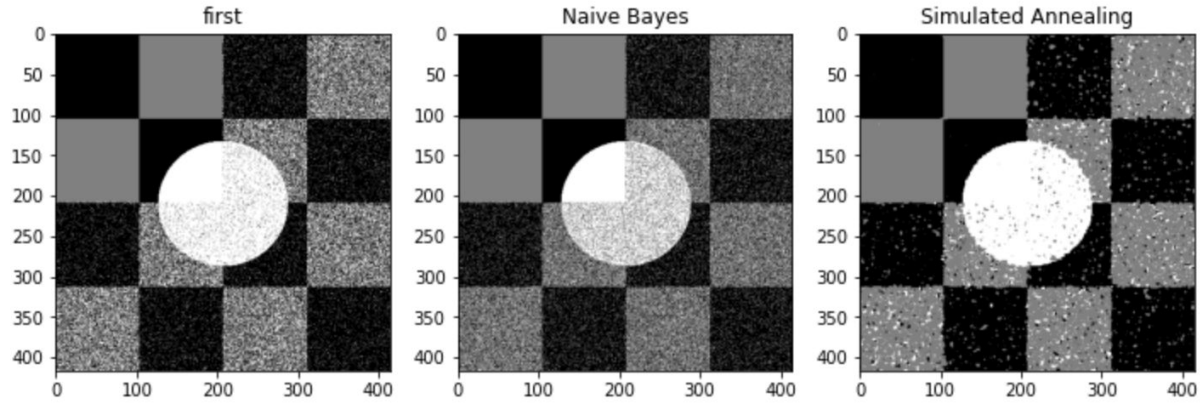


همانطور که مشاهده می‌شود، در بین بتاهای انتخاب شده، betha=1 بهترین نتیجه را داده است.

قسمت ز

در این قسمت، یک چهارم تصویر نویزدار، با استفاده از تصویر اولیه، به صورت درست مقداردهی می‌شود. مطابق انتظار، آن قسمت که به درستی مقداردهی شده است در هر دو حالت دسته‌بند بیز ساده و SA، نتایج خوبی را نشان می‌دهد.

```
Noisy Match Percent:  0.7746438993383606
Naive Bayes Match Percent:  0.822212591372685
Simulated Annealing Match Percent:  0.932096732252752
```



هرچند همانطور که مشاهده می‌شود، SA بر روی لبه‌های قسمت مربعی پیکسل‌های داده شده از بیز ساده بدتر عمل کرده است ولی در کل عملکرد بهتری داشته است.

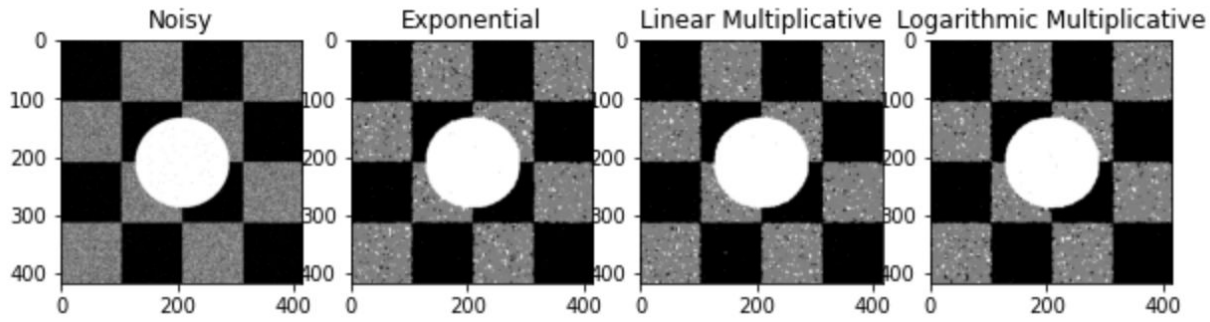
البته موردی که در نگاه اول واضح نیست، بهبود عملکرد هر دوی این قطعه‌بندها با استفاده از پیکسل‌های داده شده است. چون پارامترهایی مثل واریانس و میانگین و احتمال کلاس‌ها به مقادیر درست نزدیک‌تر می‌شوند و این مورد باعث بهبود عملکرد بیز ساده و SA می‌شود.

قسمت ح

هدف این قسمت استفاده از توابع دمای مختلف است. برای مقایسه، سه تابع انتخاب شد. (منبع توابع)

Exponential multiplicative	$T_k = T_0 \cdot \alpha^k \qquad (0.8 \le \alpha \le 0.9)$
Logarithmical multiplicative	$T_k = \frac{T_0}{1+\alpha \mathrm{Log}(1+k)}$
Linear multiplicative	$T_k = \frac{T_0}{1+\alpha k}$

```
Noisy Match Percent:  0.9175580017913381
Exponential Match Percent:  0.9521019329115021
Linear Multiplicative Match Percent:  0.9510444656323134
Logarithmic Multiplicative Match Percent:  0.9508422177920315
```



همانطور که مشاهده شد، بهبود خاصی با تغییر این توابع حاصل نشد.

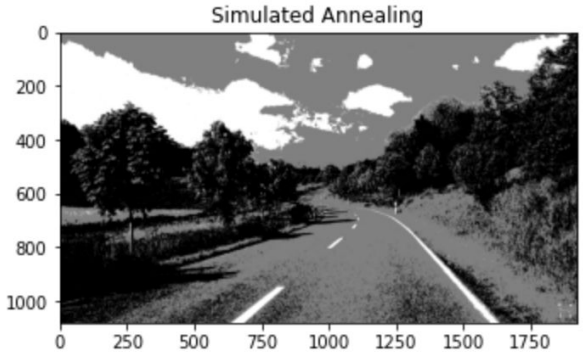
(البته با ثابت‌های متفاوت اجرا شد ولی تغییر خاصی مشاهده نکردم و فقط یک حالت از آن‌ها را گزارش کردم)

بخش ۲

هدف از این بخش پروژه، قطعه‌بندی با استفاده از روش‌های قبلی و البته گسترش‌هایی بر روی آن است.

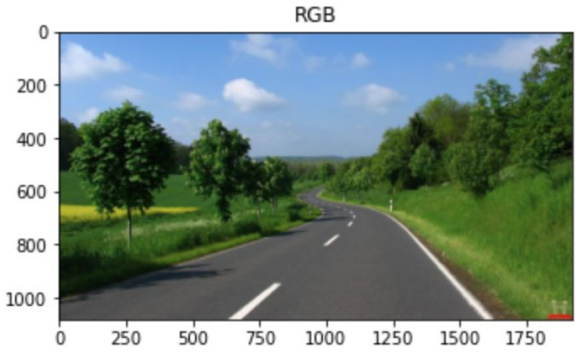
قسمت الف

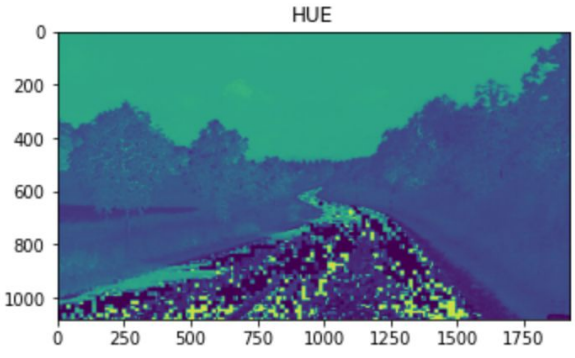
ابتدا تصویر رنگی به خاکستری تبدیل شده و با استفاده از MRF بخش قبلی به سه کلاس قطعه‌بندی می‌شود. این سه کلاس باید جاده، فضای سبز و آسمان باشد. برای این کار از همان تابع بخش ۱ استفاده شد:



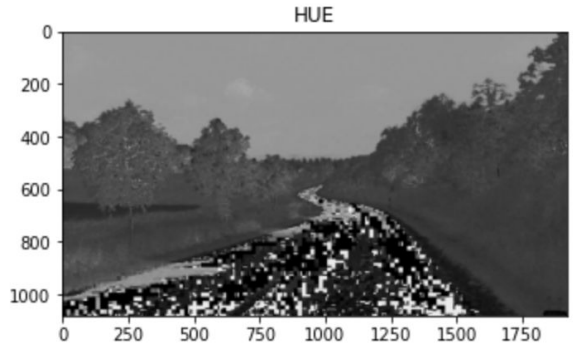
همانطور که مشاهده می‌شود، قطعه‌بندی اصلا به خوبی انجام نشده و تبدیل تصویر به سه بخش جاده و فضای سبز و آسمان موفقیت‌آمیز نبوده است.

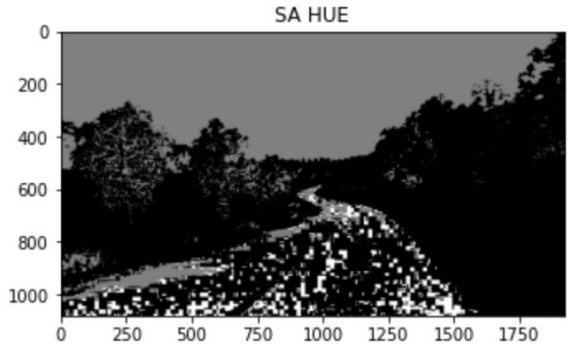
حال تصویر RGB به HSV تبدیل شده و ویژگی Hue را از آن استخراج می‌کنیم:





سپس دسته‌بندی را با استفاده از ویژگی HUE انجام می‌دهیم:

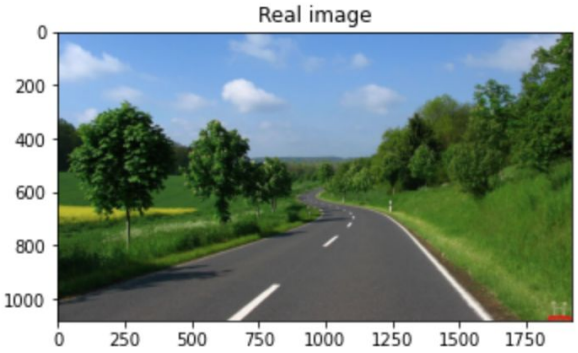


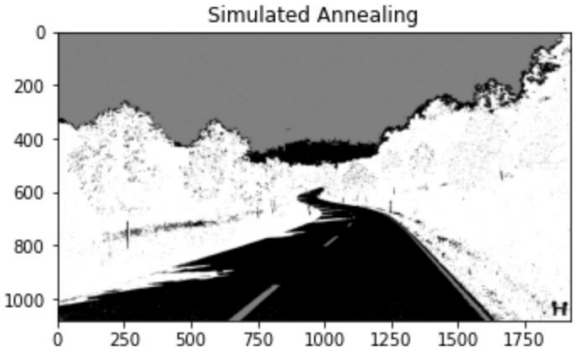


همانطور که مشاهده می‌شود، آسمان و درختان به خوبی جدا شده است ولی جاده اصلا به درستی قطعه‌بندی نشده است.

قسمت ب

حال برای قطعه‌بندی درست تر، از ویژگی رنگ استفاده می‌کنیم. در حالت‌های قبلی فقط از سطح خاکستری برای MRF و در حالت دوم از Hue استفاده می‌شد. ولی در این قسمت با استفاده از شدت سه رنگ RGB قطعه‌بندی انجام می‌شود.
با تغییر توابع اولیه، قطعه‌بندی با RGB به نتیجه زیر رسید:





همانطور که مشخص است، جاده و فضای سبز و آسمان به خوبی از هم جدا شده‌اند و به نتیجه مطلوبی رسیدیم.