به نام خداوند جان و خرد

احسان قاسمی - 98102108

تمرین دوم درس پردازش تصاویر دیجیتال

استاد درس: دکتر داوود پوره

دوشنبه 1402/4/5

این فایل برای توضیح کدهای اجرایی و تمامی مواردی است که در طول تمرین به کار برده شده است. خروجی ها با نام های گفته شده ذخیره شده اند و نیز فایل .ipynb نیز که حاوی تمامی خروجی ها و نیز کدهای زده شده است در پیوست ارسال شده است.

1. Hough Transform

روش الگوریتم من در این سوال به این صورت بود که ابتدا تصاویر را در ورودی دریافت میکردیم. سپس بر روی تصویر اصلی یک تابع لبه یابی پیاده سازی کرده ام. سپس روی تمامی نقاط مشاهده شده در تصویر دارای لبه پیمایش کرده ام. به ازای هر نقطه که در روی این تصویر مشاهده میکردم تمامی خطوطی که امکان گذر از این نقطه را دارند را در ماتریس تعریف شده یک مقدار افزایش داده ام. سپس یک حد معقول برای پیدا کردن خطوط در نظر گرفته ام. در صورتی که مقدار تعیین شده برای هر خط از حد مشخص شده افزایش پیدا میکرد آن را به عنوان یک خط قوی شناسایی کرده و در تصویر خروجی آن خط را نمایش میدادم. در نهایت تمامی خطوط نمایش داده شده اند. برای پیدا کردن خطوطی که فقط مربوط به صفحه شطرنج باشند آمده ام و زاویه خطوط و نیز شیب های شان را مقایسه کرده ام. در صورتی که رنج قابل قبولی از خطوط که خط های قوی باشند دارای شیب یکسانی باشند آن خطوط را به عنوان خط های مربوط به صفحه شطرنج لحاظ میکنم. سپس فاصله خطوط را از یکدیگر اندازه میگیرم و خطوطی که فاصله آنها از یکدیگر دارای آن فاصله باشند نگه داشته و سایر خطوط را پاک کردم. با این حرکت توانستم تمامی خطوطی که صرفا مربوط به صفحه شطرنج باشند را شناسایی کنم. در مرحله بعد مشکل دیگری که آشکار میشود این است که خط های زیادی که مربوط به یک خط هستند در صفحه پیدا میشوند. در گام بعدی من آمده ام و خط هایی که ثبت شده اند را در یک آرایه ذخیره کرده و خطوطی که از آن به بعد به خط های ثبت شده نزدیک باشند را ایگنور میکنم. در قسمت بعدی باید نقاط تقاطع را ثبت کنیم. برای این کار آمده ام و لیست تمامی خطوطی که ثبت کرده ام را به صورت زاویه و نیز فاصله در نظر گرفته ام. طبق فرمولی که از ریاضی دبیرستان برای تقاطع خطوط میدانیم نقاط تقاطعی را در صفحه نمایش داده ام.

2. Template matching

الگوریتم پیاده سازی شده برای این بخش به این صورت است که به صورت اسلاید وار در کل تصویر اصلی پیمایش کرده ام و اسلایدهایی که مشابه به تمپلیت بوده اند را ذخیره کرده ام. سپس یک آستانه در نظر گرفته ام که در صورتی که اختلاف تصاویر از آن آستانه کمتر بود آن قسمت را ذخیره کند. میتوانید مقدار آستانه را تغییر داده و مشاهده کنید که تطابق های بیشتر یا کمتری را ذخیره میکند. من در این سوال یک ایده زدم. شما اگر به صورت معمولی بیایید و این الگوریتم را اجرا کنید زمان خیلی زیادی را صرف اجرای آن خواهید کرد چرا که پیشروی کردن در طول کل تصویر برای پیدا کردن تمپلیت خواسته شده بسیار زمان گیر است. من از الگوریتم randomized استفاده کرده ام. به این صورت تصویر را تقسیم بندی کرده ام که با تفاوت های یکسانی پیکسل ها را به صورت تصادفی انتخاب کرده و در صورتی که تطابق از حدی کمتر شد همسایه

های آن را بررسی میکند تا به بهترین تطابق برسد. ایده دوم من هم این بود که به جای این که از خود تصویر اصلی برای پیدا کردن تمپلیت استفاده کنیم میتوانیم از روش های لبه یابی استفاده کرده و لبه ها را پیدا کنیم سپس تصویر ورودی را به جای ۲۵۵ پیکسل به تصویر باینری تبدیل کنیم و نهایتا بر روی آن الگوریتم را پیاده سازی کنیم. برای همین الگوریتم پیاده سازی شده توسط این ایده بسیار سریع تر از حالت معمولی است و میتوانید آن را اجرا کرده و سرعت بالای آن را مشاهده کنید. و در آخر هم یک کار دیگر کردم و آن هم این است که مشاهده میکنید که در این الگوریتم همسایه های تمپلیت اصلی را هم به عنوان تمپلیت معرفی میکند. آخرین گام الگوریتم من این بود که باکس هایی که مربوط به یک شی بودند را با نزدیکی مختصات آنها شناسایی کرده و حذف کردم.

3. Image Completion

این الگوریتم من بسیار مشابه به مورد قبلی بود. به این صورت عمل کردم که به صورت تصادفی نقاطی را بررسی میکردم در صورتی که تطابق نزدیکی برای جایگذاری برای اسلاید معرفی شده بودند آن را انتخاب کرده و همسایه های آن را بررسی میکردم تا بهترین تطابق را به دست آورم. من hole filling را هم به صورت تصادفی انجام دادم که مشاهده میکنید سنخیت نسبتا کمی برای پر کردن آن حفره دارد. و نیز از روش Cross Correlation برای بررسی تطابق همسایه های یک اسلاید با جایگاه آن استفاده کرده ام و بهترین تطابق را انتخاب کردم. همان طور که در تصاویر ارسالی مشاهده میکنید با این روش تطابق خیلی زیادی در پر کرده حفره مشاهده میشود.