



دانشگاه صنعتی امیر کبیر
(پلی تکنیک تهران)

درس پیتاپی کامپیوونر

استاد درس جناب آقای دکتر صفابخش

(تمرین سری سوم)

محسن عبادپور | ۴۰۰۱۳۱۰۸۰ | m.ebadpour@aut.ac.ir

نیمسال اول سال تحصیلی ۱۴۰۱-۱۴۰۲



فهرست پاسخ ها

۳.....	مسئله اول
۵.....	مسئله دوم
۷.....	مسئله سوم
۹.....	مسئله چهارم
۱۰	استفاده از تکنیک سیگنالی برای تعیین وجود چرخش از کلیشه

مسئله اول

مسئله اول

الگوریتم OTSU بر اساس رابطه و فرآیند ذکر شده در کلاس درس پیاده سازی شده است؛ در این الگوریتم ابتدا هیستوگرام نرمال شده‌ی تصویر ورودی را بدست می‌آوریم و آن را p_i می‌نامیم که احتمال رخ داد سطح خاکستری i ام را نشان می‌دهد. حال به ازای هر سطح خاکستری ممکن برای حد آستانه شدن (k)، روابط زیر را محاسبه می‌کنیم؛ سطوحی میتوانند به عنوان حد آستانه کاندید شوند که هم کوچکتر و هم بزرگتر از آن سطوح دیگر وجود داشته باشد لذا در سطح خاکستری l بیتی که سطوح ما میتوانند بین ۰ الی ۲۵۵ تغییر کنند، سطح ممکن برای حد آستانگی برابر است با $1 \text{ الى } 254$.

$$P_1(k) = \sum_{i=0}^k p_i$$

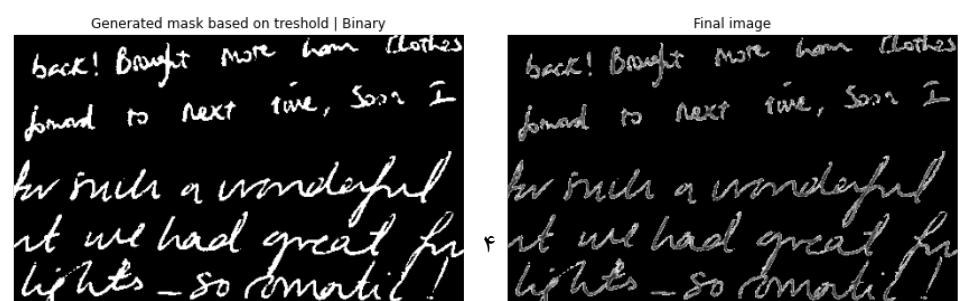
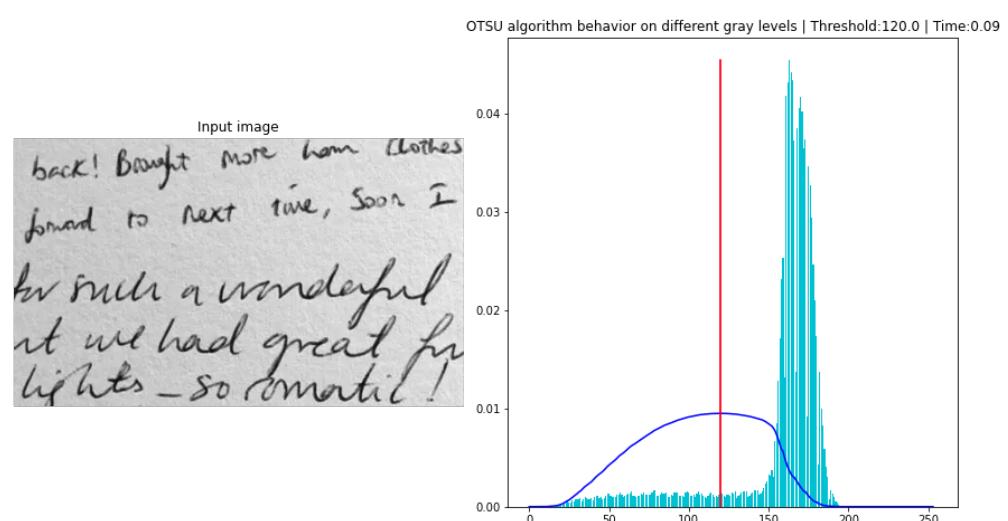
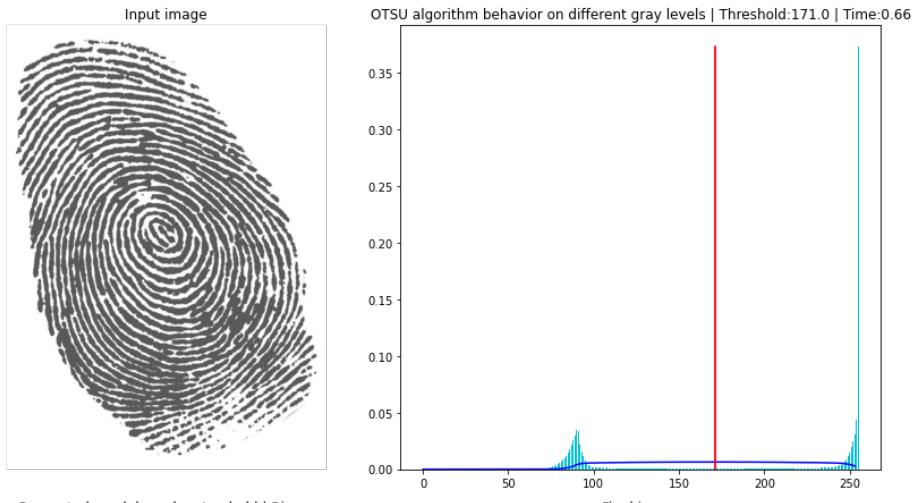
$$m_1(k) = \sum_{i=0}^k ip_i$$

$$m_G = \sum_{i=0}^{L-1} ip_i$$

$$\sigma_B^2(k) = \frac{[m_G P_1(k) - m_1(k)]^2}{P_1(k)[1 - P_1(k)]}$$

وقتی به ازای تمامی سطوح خاکستری رابطه فوق را حساب کنیم، سطح خاکستری متناظری که آن را بیشینه کرده باشد به عنوان حد آستانه ما انتخاب خواهد شد. اگر چندین سطح خاکستری مختلف آن را بیشینه کرده باشد، میانگین آنها در نظر گرفته می‌شود؛ لذا ممکن است حد آستانه ما عددی اعشاری نیز حاصل شود. در زیر به ازای دو تصویر انتخابی نتیجه جدا سازی پس زمینه آورده شده است.

تصویر اول، سطح خاکستری تصویر ورودی می‌باشد که نمایش داده شده است. تصویر دوم، نمودار هیستوگرام نرمال شده را به همراه حد آستانه‌ی بدست آمده و مقادیر تابع فوق به ازای k را نشان میدهد (برای نمایش بهتر، scale شده است)؛ در قسمت عنوان مدت زمان اجرایی و حد آستانه‌ی بدست امده درج شده است. تصویر سوم، تصویر mask تولید شده به ازای حد آستانه‌ی بدست آمده را نمایش می‌دهد و تصویر چهارم نیز تصویر نهایی را نشان می‌دهد که در آن سطوح پس زمینه مشکی رنگ شده است.



مسئله دوم

مسئله دوم

الگوریتم iterative یا همان تکراری، الگوریتمی است که با یک فرض اولیه و طی تکرار های متعدد سعی می کند یک حد آستانه ای مناسب بین میانگین پیکسل های کلاس پس زمینه و شی قرار دهد.

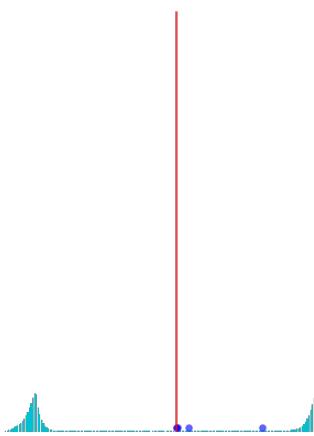
در این روش ابتدا فرض می شود که چهار پیکسل گوشی تصویر پس زمینه بوده (دسته اول) و مابقی شی (دسته دوم) مورد نظر می باشد، سپس میانگین هر دسته محاسبه می شود؛ حال حد آستانه ای جدید برابر با میانگین دو میانگین فوق تعیین می شود. حال بر اساس حد آستانه ای جدید، مجددا همه پیکسل ها در دو دسته قرار گرفته (بالاتر از حد آستانه و پایین تر از آن) و دوباره میانگین هر دسته محاسبه شده و فرآیند تکرار می شود و این روند تا زمانی ادامه پیدا می کند که حد آستانه همگرا شده باشد. از جایی که حد آستانه توسط میانگین گیری محاسبه می شود لذا قابل انتظار است که حد آستانه میتواند اعشاری حاصل شود.

در زیر به ازای دو تصویر انتخابی نتیجه جدا سازی پس زمینه آورده شده است. تصویر اول نشان دهنده سطح خاکستری تصویر ورودی می باشد. تصویر دوم نمودار هیستوگرام تصویر به همراه حد آستانه ای بدست امده را نشان می دهد. نقاط توپر نیز حد آستانه های بدست آمده در هر تکرار و حرکت آن را به سوی همگرایی نشان می دهد.

در قسمت عنوان نمودار نیز تعداد تکرار انجام شده، مدت زمان اجرایی و مقدار حد آستانه ای بدست آمده درج شده است. تصویر سوم mask تولیدی بر اساس حد آستانه ای محاسبه شده را نشان می دهد و تصویر چهارم نیز تصویر نهایی را نشان می دهد که در آن سطوح پس زمینه مشکی رنگ شده است.



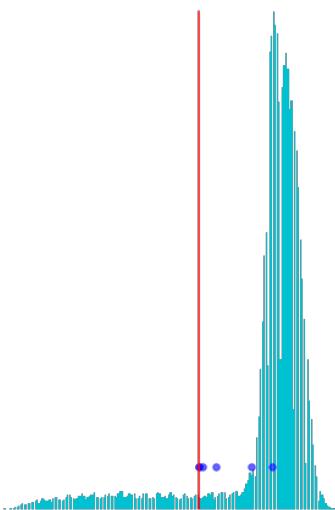
Iterative algorithm | iterate:5 | Threshold:172.39 | Time:1.59



Iterative algorithm | iterate:6 | Threshold:121.28 | Time:0.1

Input image

back! Bought more from Clothes
forward to next time, soon I
for such a wonderful
it we had great fire
lights - so romantic!



Generated mask based on threshhold | Binary

back! Bought more from Clothes
forward to next time, soon I
for such a wonderful
it we had great fire
lights - so romantic!

Final image

back! Bought more from Clothes
forward to next time, soon I
for such a wonderful
it we had great fire
lights - so romantic!

مسئله سوم

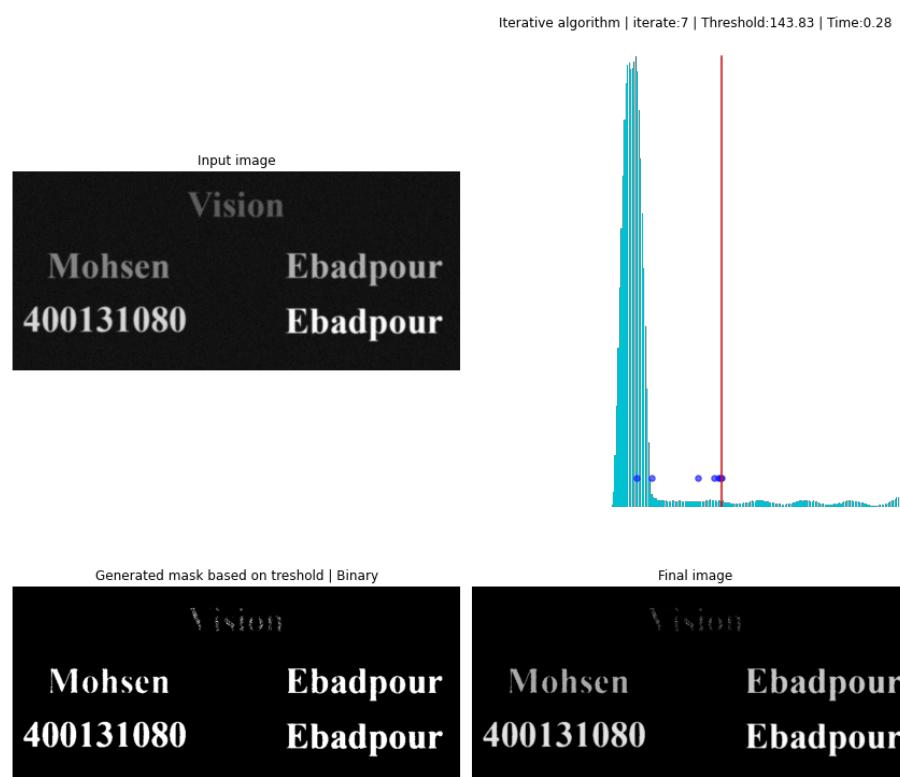
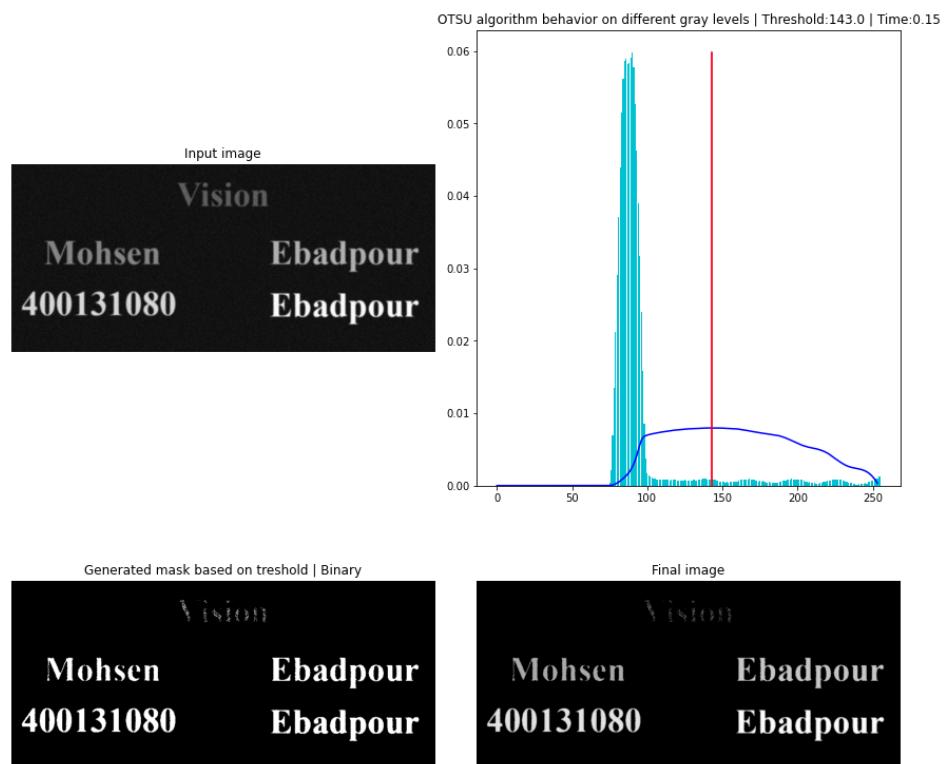
مسئله سوم

همانطور که در عنوان نمودار ها قابل مشاهده است، مدت زمان اجرایی الگوریتم تکراری (iterative) بیشتر از الگوریتم OTSU می باشد لذا در بحث سرعت، الگوریتم تکراری ضعیف تر حاصل شده است. دلیل این امر آن است که در محاسبات الگوریتم تکراری ابعاد تصویر و تعداد پیکسل های موجود در آن یک عامل تاثیرگذار است چرا که در هر تکرار همهی پیکسل های تصویر بایستی پردازش و در یک دسته قرار گیرند و میانگین انan محاسبه شود ولی این در صورتی است که در الگوریتم OTSU ابعاد تصویر و تعداد پیکسل ها موجود در محاسبات میانی هیچ تاثیری ندارد.

به جهت دقت نیز، حد آستانه های بدست امده در تصاویر انتخابی نزدیک به هم است و تفاوت فاحشی وجود ندارد. اما اگر بخواهیم تئوری این دو الگوریتم را از جهت دقت حاصل مورد مطالعه قرار دهیم، بایستی اشاره نمود که الگوریتم تکراری بصورت حریصانه در صدد یافتن حد آستانه ای می باشد که در حد فاصل میانگین دو دستهی پس زمینه و شی قرار داشته باشد (به تعبیری میتوان گفت در حال یافتن ماکسیمم محلی می باشد) و این در صورتی است که الگوریتم OTSU با محاسبه واریانس (بین کلاسی پس زمینه و شی) به ازای تمامی حد های آستانهی ممکن سعی می کند بهترین حد آستانه را انتخاب کند (به تعبیری دیگر میتوان گفت در حال یافتن ماکسیمم سراسری). با عنایت به توضیحات و استدلال های ارائه شده میتوان انتظار داشت که الگوریتم OTSU میتواند در تصاویر چالشی با دقت تر حاضر شود چرا که همهی حالات ممکن را در نظر گرفته است (و این در حالی است که سریع تر نیز محاسبه می شود).

اشارة به این نکته ضروری می باشد که هر دو الگوریتم یاد شده برای تشخیص پس زمینه ای مناسب هستند که سطوح روشنایی شی و پس زمینه با هم فاصله قابل قبول داشته و در عین حال خود شی یا اشیا نیز دارای سطح روشنایی بسیار گسترده نباشد چرا که صرفا با یک حد آستانه نمیتوان پس زمینه را از چند شی با سطوح روشنایی مختلف شناسایی کرد. تصویر زیر در فتوشاپ تولید شده و دارای چند شی با سطوح روشنایی متعدد می باشد که همان نوشته های ما هستند. همانطور که می بینیم هر دو الگوریتم نتوانستند حد آستانه را طوری تعیین کنند که کلمه Vision نیز جدا شود و دلیل آنست که سطح روشنایی کلمه مذکور با پس زمینه اختلاف کمتری نسبت به بقیه دارد. (ضعف هر دو الگوریتم)

در این تصویر نیز ملاحظه می شود که به جهت زمانی همچنان الگوریتم OTSU بهتر عمل کرده است.

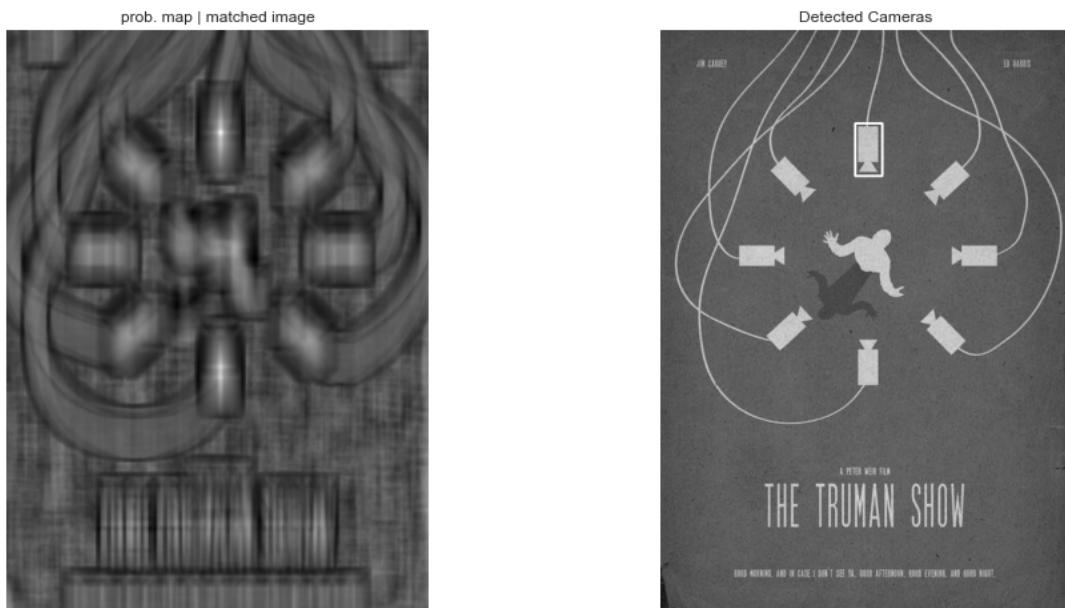


Λ

مسئله چهارم

مسئله چهارم

با استفاده از تابع ذکر شده از کتابخانه open-cv عملیات تطبیق کلیشه انجام شده و خروجی بصورت زیر حاصل شده است. تصویر سمت چپ نقشه احتمالاتی تطبیق کلیشه با تصویر ورودی می‌باشد که روشنایی هر پیکسل احتمال تطبیق کلیشه در آن محل از تصویر را نشان میدهد. همانطور که قابل ملاحظه است ما در دو ناحیه نقاط بسیار روشن داریم و حاکی از احتمال بالای تطبیق کلیشه در آن محل ها می‌باشد. احتمال بالای تطبیق کلیشه در قسمت بالایی تصویر بدیهی است اما سوال این است چرا تطبیق کلیشه در قسمت پایینی نیز احتمال بالایی به خود گرفته است؟ دلیل آن است که بخش قابل توجهی از کلیشه یک مستطیل رنگی توپر است و این مستطیل در دوربین های بالایی و پایین کاملا مشترک است لذا احتمال تطبیق کلیشه در قسمت پایینی نیز ظاهر شده است. تصویر سمت راست نیز تصویر شناسایی شده کلیشه در تصویر می‌باشد که دور آن خط کشیده شده است.



از همین نقشه احتمالاتی تطبیق کلیشه با تصویر ورودی میتوان به صراحة گفت که این الگوریتم و تابع اشاره شده در متن سوال نسبت به چرخش مقاوم نمی‌باشد چرا که در آن نواحی هیچ احتمال بالایی رخ نداده است. الگوریتم پیاده سازی شده در تابع مذکور بدین صورت عمل می‌کند که کلیشه را در تمامی محل ها ممکن در تصویر قرار داده و احتمال وجود کلیشه در آن را محاسبه میکند لذا هیچ عملیات و پردازشی برای مقاوم بودن در قبال چرخش مطرح نیست. برای حل این مشکل میتوان خود کلیشه را در درجه های مختلف چرخاند و سپس نسخه های چرخانده شده کلیشه را با تصویر ورودی تطبیق داد؛ سپس تعیین نمود به ازای

هر درجه ای از چرخش کلیشه آیا در تصویر تطبیقی از آن یافت می شود یا خبر که در ادامه جزئیات پیاده سازی و نتیجه هی آن شرح داده می شود.

استفاده از تکنیک سیگنالی برای تعیین وجود چرخش از کلیشه

یک مسئله ای که بصورت کلی در بحث جست وجوی نسخه های چرخش یافته ای کلیشه در تصاویر وجود دارد آن است که چگونه میتوان تعیین نمود به ازای چه درجه چرخش هایی از کلیشه در تصویر نمونه وجود دارد؟ طبق تحقیق و جست وجوی انجام شده، یک راهکار از مباحث سیگنالی برای این امر وجود دارد بدین صورت که به ازای $360 - 0$ درجه (یک تناوب کامل) کلیشه چرخانده می شود و فرآیند تطبیق با تصویر ورودی انجام می شود و سپس در هر چرخش بزرگترین احتمالی که محاسبه شده است، به عنوان خروجی آن چرخش انتخاب می شود. این خروجی ها بصورت ترتیبی قرار گرفته و یک سیگنال متناوب را پدید می اورد که تفاسیر مختلفی را میتوان به آن نسبت داد. ساده ترین تفسیر در تصاویر و چرخش های ساده، آن است که نقاط بیشینه محلی نقاط و چرخش هایی هستند که چرخش کلیشه در آن رویت شده است. بنده این راهکار را پیاده سازی کرده ام. برای این منظور، از تابع rotate_image برای تولید تصویر چرخش یافته ای کلیشه به ازای یک تناوب کامل استفاده کرده و سپس با تابع matchTemplate نقشه احتمالی تولید و تطبیق صورت میگیرد و سپس بیشینه ای آن ذخیره می شود تا سیگنال مد نظر تولید شود. پس از اتمام این مرحله، از تابع argrelextrema استفاده کرده و نقاط بیشینه محلی را استخراج می کنیم و سپس به اندازه چرخش متناظر، محل کلیشه ای آن را یافته و دور آن خط می کشیم؛ در تصویر زیر نتایج آورده شده است. تصویر سمت چپ سیگنال تولیدی به همراه نقاط بیشینه محلی و درجه متناظر قابل مشاهده بوده و تصویر سمت راست نیز خروجی نهایی را نشان میدهد.

