بسمه تعالی

دانشگاه صنعتی شریف

دانشکده علوم کامپیوتر

Image processing - دکتر مصطفی کمالی

HW2 q2 report

سوال دو -template matching :

در این سوال هم مثل سوال قبل تابع normalize را داریم . در ابتدا عکس اصلی و template را ورودی گرفته و grayscale میکنیم ، سپس روی تصویر اصلی یک فیلتر گاوس با اندازه 13 در 13 و σ =3 اعمال میکنیم تا کمی blur شود ، زیرا به تجربه نتیجه بهتری میدهد. پس از مقداری بررسی کردن با استفاده از توابع آماده و روش های مختلف برای template matching در نهایت از ترکیب دو روش SSD و zero-mean cross-correlation استفاده میکنیم که در ادامه توضیح خواهیم داد .

برای اینکه اجرای الگوریتم سریع باشد ابتدا عکس را به 32 درصد مقدار ابعاد خود resize میکنیم ، این عدد به طور تجربی بدست آمده و اینکه چون توان 2 است کار با آن بهتر است . همچنین عکس template را به 16 درصد ابعاد کوچک میکنیم ، مثل این است که هیچکدام را در ابتدا کوچک نکرده باشیم و template را به نصف اندازه اولیه کوچک کنیم . این مراحل را preprocess روی عکس ها در نظر میگیریم .

در مرحله بعدی دو ماتریس حاصل از SSD و zm-cross-corr را بدون استفاده از توابع آماده و خودمان بدست می‌آوریم، به اینصورت که ابتدا دو ماتریس به اندازه تصویر ورودی با همه درایه های صفر در نظر میگیریم سپس مقدار درایه SSD[m,n] طبق فرمول برابر است با مجموع توان دوی درایه های ماتریس تفاضل template و یک پنجره از تصویر که نقطه [m,n] چوشه بالا چپ آن است و اندازه آن برابر است با template ، همچینن مقدار ZM_ccor[m,n] برابر است با مجموع درایه های ماتریس حاصل ضرب template و پنجره گفته شده که از پنجره گفته شده میانگین آن هم کم شده است . این کار را برای m از 0 تا ارتفاع عکس منهای ارتفاع template و n از 0 تا عرض عکس منهای عرض تمپلیت انجام میدهیم که در نهایت دو ماتریس SSD و zm-cross-corr بدست می آید که به صورت شکل صفحه بعد اند.

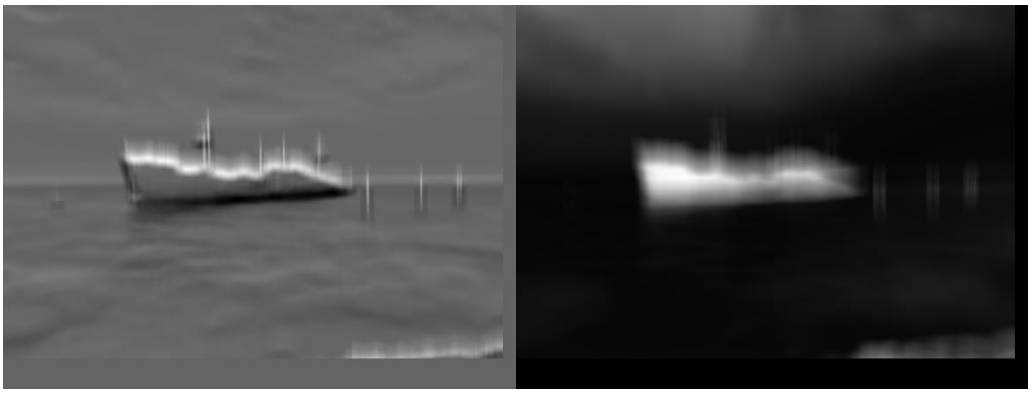
- Method 2: Zero-Mean Cross-Correlation

$$h[m,n] = \sum_{k,l} (g[k,l] - \bar{g}) f[m+k, n+l]$$

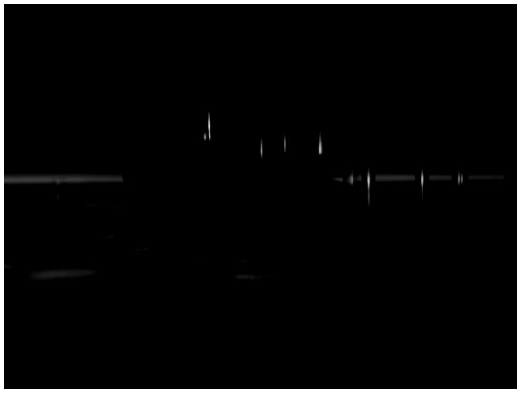
- Method 3: SSD

$$h[m,n] = \sum_{k,l} (g[k,l] - f[m+k, n+l])^2$$

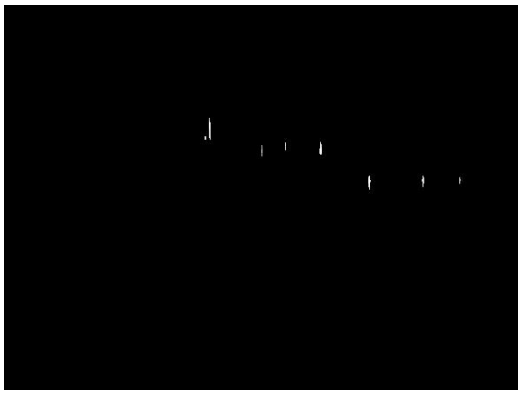
Zero mean cross correlation SSD



مرحله بعدی استفاده از این دو ماتریس برای بدست آوردن محل بهترین match هاست ، برای اینکار میدانیم بهترین match در zm_ccor برابر نقاط بیشینه و در SSD برابر نقاط کمینه است اما خیلی کاری به این مورد نداریم ، همانطور که از شکل پیداست مقادیر خطا در هر دو روش وجود دارد ، برای این کار کاملا تجربی و با توجه به این مسیله خاص ابتدا عبارت ZM_ccor-0.1(SSD) را حساب کرده و محل هایی که مقدار منفی دارند را صفر میکنیم که حاصل به صورت شکل زیر است که normalize شده:



سپس در شکل بالا یک threshold برابر با 110 اعمال کرده و مقادیر کمتر برابر با 0 و مقادیر بیشتر برابر با 255 که میشود :



سپس این ماتریس بدست آمده با scale=1/0.32 بزرگ میکنیم تا برابر با شکل اولیه شود و کارهای زیر را انجام میدهیم.

اگر مقادیر 255 در ماتریس بدست آمده را در شکل رسم کنیم مقدار زیادی مستطیل حول محل های پیدا شده بدست میآید. برای حل این مشکل در تصویر بالا در هر ستون از ماتریس فقط یک مقدار را برابر 255 کرده و بقیه را صفر میکنیم که مختصات آن مقدار برابر میانگین مختصات نقاطی در آن ستون است که مقدار 255 دارند و همه آنها را در آرایه pnt میریزیم. سپس pnt_final را به اینصورت پر میکنیم که نقاطی را از بین نقاط pnt انتخاب میکنیم که اولا فاصله دو نقطه مجاور از هم بیشتر از 30 باشد و درثانی مختصات y دو نقطه مجاور از 20 باشد ، تعداد نقاط موجود در pnt برابر با 138 ولی تعداد نقاط موجود در pnt_final برابر با 9 عدد است.

نقاط بدست آمده برابر مختصات بالا سمت چپ مستطیل هایی به اندازه template است که match بهتری دارند ، با رسم این مستطیل ها به شکل نهایی میرسیم