



دانشگاه صنعتی امیر کبیر
(پلی تکنیک تهران)

درس پیتاپی کامپیوونر

استاد درس جناب آقای دکتر صفابخش

(تمرین سری پنجم)

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

نیمسال اول سال تحصیلی ۱۴۰۱-۱۴۰۲



فهرست پاسخ ها

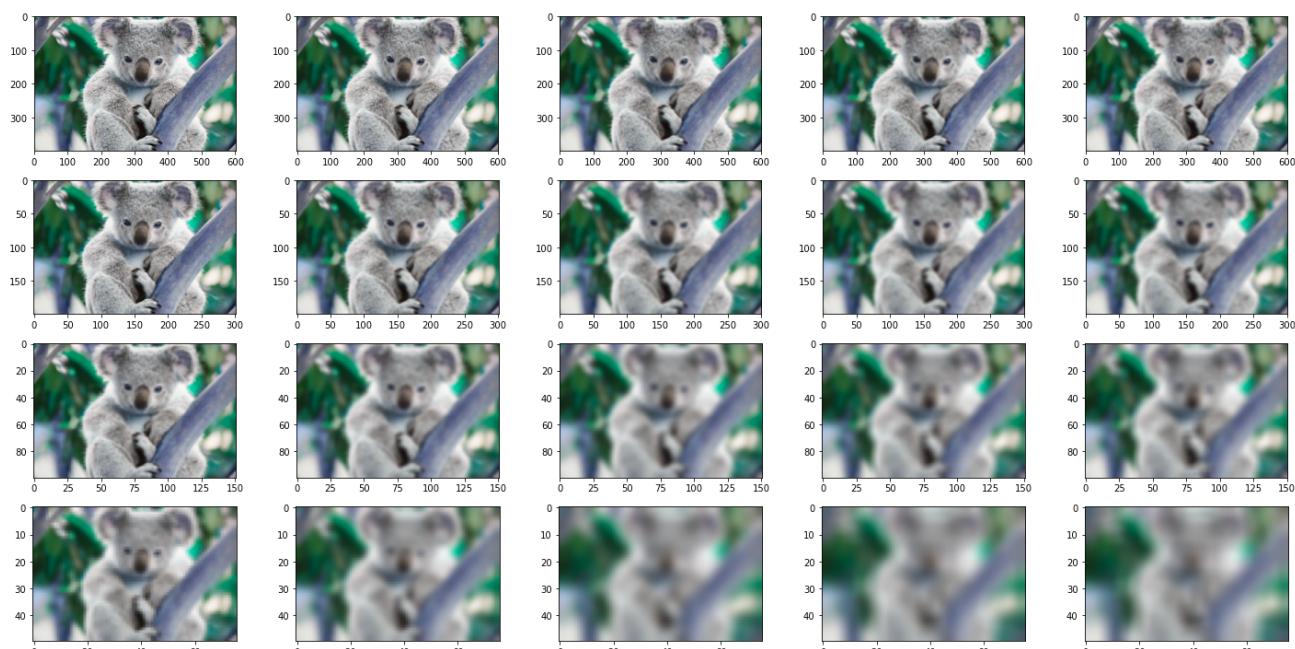
۳	مسئله اول: فرآیند استخراج ویژگی SIFT
۷	مسئله دوم: تشخیص چرخش و بازیابی جهت اولیه با SIFT
۱۰	مسئله سوم: تشخیص چرخش و بازیابی جهت اولیه با FREAK
۱۳	فعالیت بیشتر: توصیف گر ORB
۱۵	فعالیت بیشتر: مقاومت در برابر نویز

مسئله اول

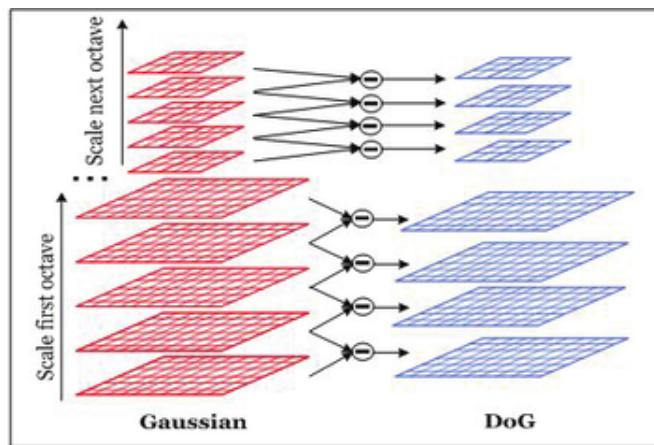
مسئله اول: فرآیند استخراج ویژگی SIFT

SIFT عبارت اختصار یافته‌ی scale-invariant feature transform می‌باشد که هدف آن تطبیق و تبدیل ویژگی محلی مستقل از مقیاس(مکانی) می‌باشد بطوریکه بتوان ویژگی هایی از تصویر و اشیای موجود استخراج کرد که در مقابل بزرگنمایی یا کوچکنمایی و همچنین چرخش(یا حتی تغییرات روشنایی) مقاوم باشد؛ سپس بتوان با تطبیق ویژگی های استخراجی بین دو عکس، شرایط یا اشیای مشابه را تشخیص داد. عملکرد SIFT را میتوان در شش مرحله توضیح داد که در ادامه شرح داده می‌شود؛ برای قسمت‌های مختلف که مثال آورده شده است همگی حاصل خروجی های دستی خود می‌باشد:

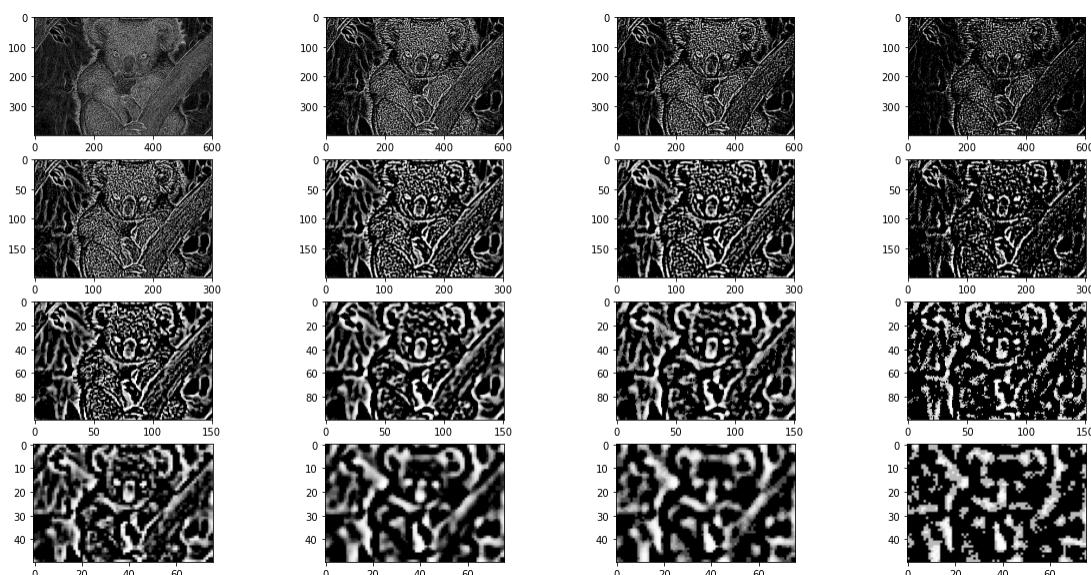
۱) ایجاد فضای مقیاس: در اولین مرحله بایستی از تصویر مدنظر ارائه های مختلفی ایجاد و به مقیاس های متفاوتی نگاشت کرد تا ویژگی های استخراجی بتوانند مستقل از مقیاس و اندازه باشند. برای این منظور، تصویر به تعداد مشخصی با درجه متفاوت تاری، تار می‌شود که مجموع آنها در ابعاد برابر یک Octave نامیده می‌شود. حال ابعاد تصویر ورودی نصف و مجدداً تعداد مشخصی نسخه تار از آن ایجاد شده و یک octave دیگر را پدید می‌آورد. تعداد نسخه های تار و تعداد octave بر حسب نوع تصویر می‌تواند متفاوت باشد. تصویر زیر(کوآلا) مثالی از ایجاد فضای مقیاس را نشان می‌دهد. هر سطر نشان دهنده یک octave می‌باشد که همه‌ی تصاویر موجود همسایز هستند و در هر یک از چهار به راست درجه تاری افزایش پیدا کرده است.



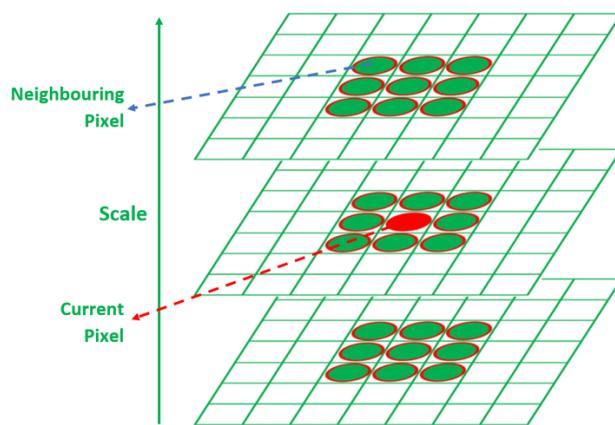
۲) تقریب عملگر LoG: خروجی فیلتر LoG (Laplacian of Gaussian) لبه ها و گوشه های تصویر و اشیای موجود در تصویر را مشخص می کند که در امر تعیین نقاط شاخص و مهم مفید هستند چرا که لبه ها و گوشه ها ساختار و ویژگی تصویر را میتوانند نشان دهند. اما، عملگر LoG پرهزینه می باشد لذا از تقریب آن استفاده می شود که حاصل تفاضل خروجی دو فیلتر گوسی با درجه تاری متفاوت در یک ابعاد می باشد؛ این عمل تفاضل گیری به ازای تصاویر متوالی هر octave انجام شده و صفحات DoG حاصل می شود؛ در تصویر زیر شماتیک این عمل نشان داده شده است:



همچنین برای مثال کوآلا DoG ها محاسبه شده و در زیر قابل مشاهده بهتر نتایج و جزئیات حاصل، هموارسازی هیستوگرام برای تصویر قبل از نمایش انجام شده است؛ همانطور که طبق شماتیک بالا قابل انتظار است تعداد DoG های حاصل از هر octave یکی کمتر حاصل می شود:



۳) یافتن نقاط شاخص: صفحات DoG مناسبی برای استخراج ویژگی بوده و برای یافتن نقاط شاخص از آن استفاده می‌شود؛ برای این منظور و در صفحات DoG، برای هر نقطه یک همسایگی 3×3 در نظر گرفته می‌شود و تناظر این همسایگی‌ها در تصویر DoG بعدی و قبلی در یک octave یکسان نیز مد نظر گرفته می‌شود؛ اگر مقدار نقطه از همه این همسایگی‌ها ۹ همسایه در DoG قبلی، ۹ همسایه در DoG بعدی و ۸ همسایه در DoG فعلی) بزرگتر یا کوچکتر باشد، آن نقطه را به عنوان نقطه‌ی شاخص در نظر می‌گیریم؛ شماتیک این فرآیند شامل انتخاب پیکسل‌های همسایه در تصویر زیر قابل مشخص است:



قابل انتظار است که در صفحات اول و آخر امکان انتخاب ۲۶ همسایه وجود نداشته و لذا در این صفحات عمل تعیین نقاط شاخص انجام نمی‌پذیرد. همچنین کاملاً ممکن است محل وقوع نقطه شاخص (کمینه یا بیشنه) دقیقاً روی پیکسل‌ها نباشد و بین پیکسل‌ها واقع شود؛ برای حل این مورد و یافتن محل دقیق نقاط شاخص می‌توان از روش‌های تخمین و درون‌یابی استفاده کرد که در روش SIFT از بسط سری تیلور D (تا درجه ۲) استفاده شده و مشتق آن را برابر با صفر قرار می‌دهیم تا محل دقیق بدست آید.

۴) حذف لبه‌ها و پاسخ‌های کنتراست پایین: بین تعداد نقاط شاخصی که در مرحله قبل استخراج می‌شود، تعدادی نقاط شاخص از اهمیت و تمایزگری کمتری برخوردار هستند و استناد به این نقاط می‌تواند چالشی و حتی گمراه کننده باشد (همچنین باعث افزایش بار محاسبات، پردازش‌ها و کاهش سرعت می‌شود) لذا بهتر است به جای کل شاخص‌های قسمت قبل، از شاخص‌های با اهمیت استفاده شود. برای این منظور می‌توان با اعمال حد آستانه یا با استفاده از لبه‌یابی و حذف نقاط با پاسخ قوی در یک امتداد به این هدف رسید و تعداد شاخص‌ها را کاهش داد.

۵) تخصیص جهت به نقاط شاخص: زمانی که نقاط شاخص نهایی شدن، حال بایستی جهتی برای آن نقاط محاسبه کنیم که جهتی معنادار بوده و دارای اطلاعات (محلی) و همسایگی باشد تا بتوان ویژگی‌های آن نقطه شاخص را بیان کند. برای این

منظور، همسایگی ای با پنجره‌ی با اندازه ۱.۵ برابر مقیاس تاری در نظر گرفته و برای هر کدام از همسایه‌های هر کدام از نقاط شاخص، جهت گرادیان و اندازه گرادیان را محاسبه می‌کنیم؛ سپس یک هیستوگرام ۳۶ بازه‌ای برای جهت‌های بدست آمده از گرادیان همسایه‌ها رسم می‌کنیم؛ در این هیستوگرام هر همسایه بر حسب دوری اش از نقطه شاخص(یه تابع گوسی) و اندازه گرادیانش، جهت گرادیان خود را وزن دار کرده و هیستوگرام مذکور تکمیل می‌گردد؛ به عبارتی دیگر وزن هر همسایه در هیستوگرام واحد نبوده و مبتنی بر فاصله از نقطه شاخص و اندازه گرادیان خود می‌باشد. جهت متناظر(بازه ۱۰ درجه‌ای) متناظر با ماکسیمم این هیستوگرام برای نقطه شاخص ثبت شده و به همراه مقیاس و محل نقطه ذخیره می‌شود. با این کار ما برای نقطه شاخص خود جهتی تعیین می‌کنیم که اطلاعات گرادیان خود و همسایگی نیز دخیل هستند. (اگر جهت دیگری در هیستوگرام باشد که بیشتر از ۸۰ درصد نقطه ماکسیمم را داشته باشد، آن نیز ذخیره می‌شود)

۶) ساخت توصیف‌گر(بردار ویژگی): حال بایستی برای نقاط شاخص بردارهایی تعیین نمود که حاوی اطلاعات آن شاخص باشد و میتوانیم با استفاده از این بردارها عمل تطبیق یا شناسایی را انجام دهیم؛ برای این منظور، یک همسایگی $16*16$ از تصویر هموار شده با مقیاس نقطه در نظر می‌گیریم؛ مقدار گرادیان همسایگی را محاسبه و بر حسب یک تابع گوسی و فاصله از نقطه شاخص وزن دار می‌کنیم. برای ایجاد مقاومت در مقابل دوران و چرخش، جهت گرادیان را بر حسب جهت نقطه شاخص محاسبه می‌کنیم. در هر همسایگی $4*4$ از این ناحیه $16*16$ ، یک هیستوگرام ۸ بازه‌ای(هر 45 درجه یک ستون) از زاویه‌های گرادیان ها می‌سازیم که در آن وزن هر پیکسل برابر با ندازه گرادیان وزن دار شده است. در نهایت 16 هیستوگرام ۸ بازه‌ای خواهیم داشت که با کنار هم گذاشتن یکدیگر به یک بردار 128 خواهیم رسید که این بردار همان بردار ویژگی یا بردار توصیف برای این نقطه شاخص خواهد بود که مستقل از چرخش می‌باشد و تاثیر همسایگی‌ها نیز در آن سنجیده شده است.(برای حذف تاثیر روشنایی میتوانیم نرمال کنیم)

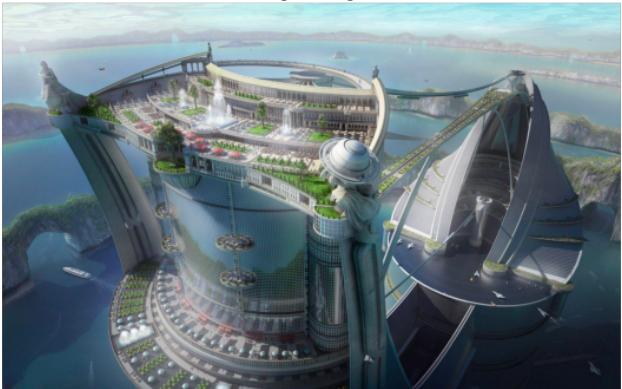
در نهایت ما به تعداد نقاط شاخص استخراجی یک بردار 128 عنصری خواهیم داشت که در کنار یکدیگر توصیف کننده‌ی تصویر و اشیای آن هستند؛ این نقاط حاصل از محاسبات در مقیاس‌های مختلف بوده و حاوی اطلاعات محلی نیز هستند که مستقل از چرخش و اندازه هستند و میتوانیم از آنان برای تطبیق یا شناسایی استفاده کنیم.

مسئله دوم: تشخیص چرخش و بازبینی جهت اولیه با SIFT

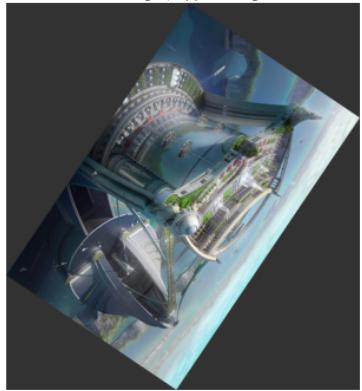
برای پاسخ گویی این بخش، چند تابع پیاده سازی شده است تا فرآیند خروجی گرفتن اصولی‌تر باشد. برای چرخش تصاویر چه در اول(زاویه تصادفی) و چه در آخر(بازبینی جهت اولیه) یک تابع با عنوان `RotateImage` پیاده سازی شده است که با استفاده از کتابخانه `scipy` انجام پذیرفته و تصویر برگشتی دارای حاشیه‌های مورد نیاز می‌باشد. تابع بعدی با هدف تولید سه زاویه تصادفی در بازه ۳۳۵-۲۵ درجه با عنوان `get_three_random_angles` پیاده سازی شده که این بازه به این جهت اعمال گردیده است که اندازه چرخش بسیار کم و شبیه به خود تصویر نباشد تا بتوان تغییرات را ملاحظه نمود. تابع بعدی `report` نام دارد که تصاویر ورودی و خروجی به همراه زمان اجرا و زوايا را در قالب یک رسم چهار تصویری خروجی می‌دهد که این تابع را می‌توان برای همه الگوریتم‌ها استفاده نمود. تابع بعدی `get_SIFT_Rotation_angle` نام دارد که دو تصویر را ورودی گرفته(تصویر اصلی و تصویر چرخش یافته با زاویه تصادفی) و سه پارامتر زاویه بدست محاسبه شده، تصویر نقاط تطبیق یافته و زمان اجرای توصیف‌گر را بر می‌گرداند. در این تابع ابتدا با استفاده از استخراج‌گر و سپس توصیف‌گر SIFT نقاط شاخص و بردار‌های متناظر برای هر کدام از تصاویر استخراج شده و توسط تطبیق نزدیک ترین همسایه(`knnMatch`) با یکدیگر تطبیق داده شده‌اند. در صورتی که فاصله‌ی بردار ویژگی یکی از نقاط با دیگری کمتر از ۰.۳ باشد، آن جفت نقطه‌ی تطبیقی به عنوان یک تطبیق خوب برای بازسازی و چرخش مجدد تصویر در نظر گرفته می‌شود. با استفاده از لیست بدست آمده از نقاط مذکور و با استفاده از تابع `findHomography` ماتریس تطبیق بدست می‌آید که با استفاده تانزانت معکوس و ماتریس فوق زاویه دوران را استخراج و محاسبه می‌کنیم. ضمناً توابع پیاده‌سازی شده با کمک از دستیار `Copilot` انجام شده است.

تصویر انتخابی برای آزمایشات یک تصویر با وضوح بالا و جزئیات ریز بوده هدف از این امر مقایسه مناسب‌تر بین توصیف‌گر‌ها و قدرت آنان می‌باشد. در زیر نتایج برای سه زاویه تصادفی آورده شده است. تصویر اول مربوط به تصویر ورودی، تصویر دوم مربوط به تصویر چرخش یافته(در عنوان زاویه درجه است)، تصویر سوم مربوط به نقاط تطبیق یافته(در قسمت عنوان زاویه‌ی محاسبه شده برای بازسازی و مدت زمان توصیف عکس‌ها درج شده است) و تصویر چهارم مربوط به چرخش مجدد تصویر برای بازسازی می‌باشد. ضمناً برای شهود بهتر تصویر پس زمینه انتخابی برای حاشیه در هر مرحله متفاوت در نظر گرفته شده است.

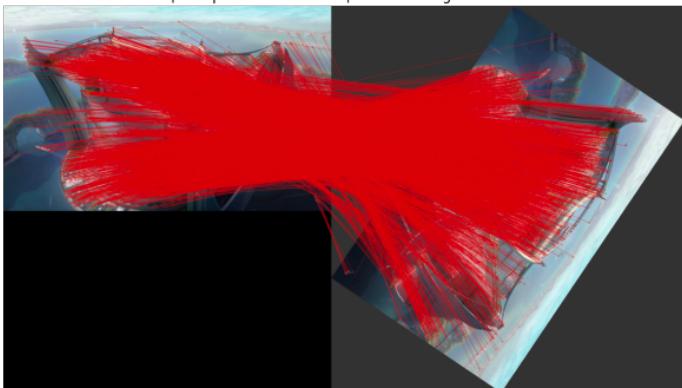
Original Image



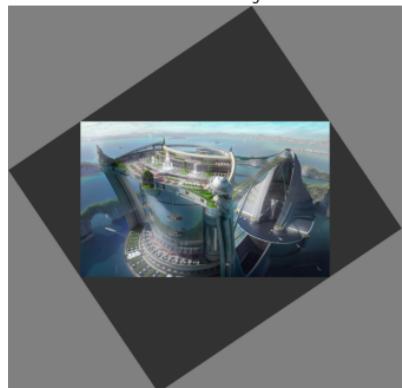
Rotated Image | Applied angle: 236



SIFT Matches | Comput Time: 1.4291s | Detected angle to restore: 123.9985



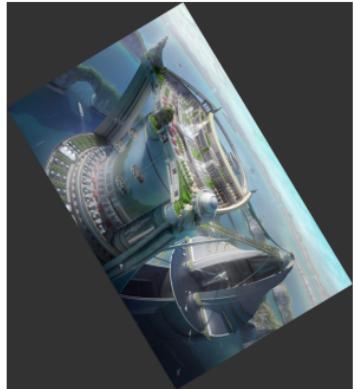
Re-Rotated Image



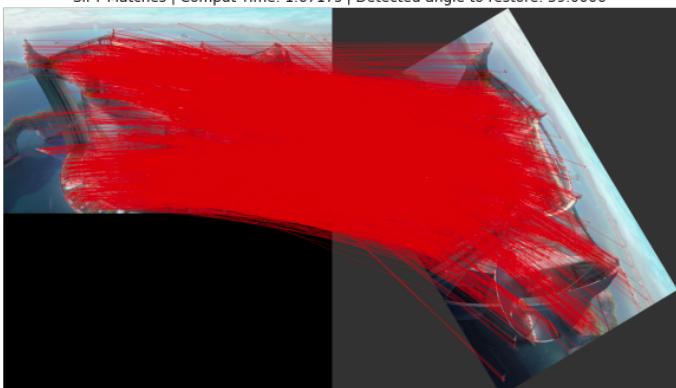
Original Image



Rotated Image | Applied angle: 301



SIFT Matches | Comput Time: 1.6717s | Detected angle to restore: 59.0006

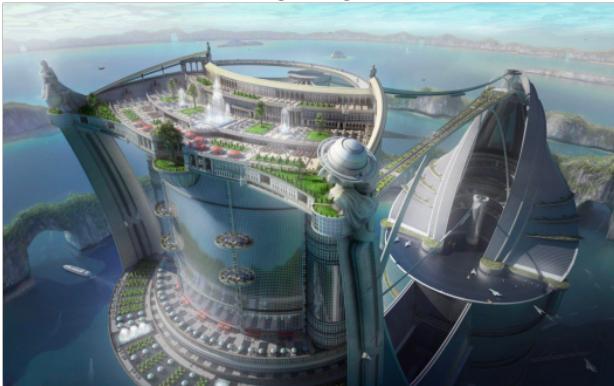


Re-Rotated Image

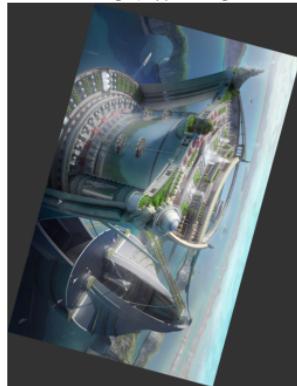


A

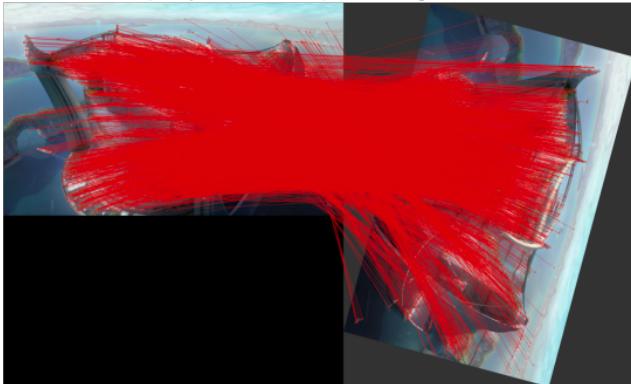
Original Image



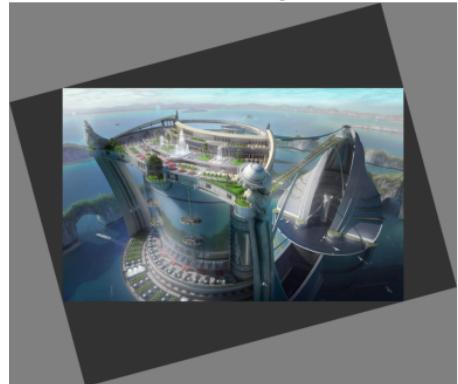
Rotated Image | Applied angle: 255



SIFT Matches | Comput Time: 1.0542s | Detected angle to restore: 104.9995



Re-Rotated Image



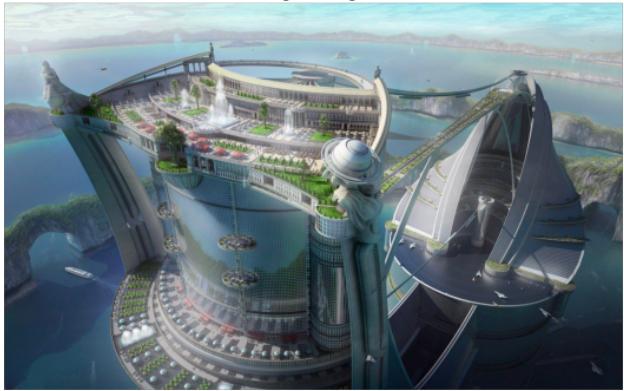
مسئله سوم: تشخیص چرخش و بازیابی جهت اولیه با FREAK

برای حل این قسمت، همانند توضیحات و توابع شرح شده در مسئله‌ی قبل اقدام شده است با این تفاوت که در قسمت توصیف‌گر از توصیف‌گر FREAK استفاده شده است. نتایج به ازای سه زاویه تصادفی اعمالی در مسئله‌ی قبل در ادامه قابل مشاهده است.(از جایی که ممکن است مدت زمان پردازش به ازای تطبیق تفاوت داشته باشد، زاویه‌های تصادفی جدیدی استفاده نشده است تا بتوانیم به جهت زمان مقایسه را انجام دهیم)

همانطور که در کلاس درس نیز مورد بررسی قرار گرفته بود، پردازش‌های متعدد در مقیاس‌های متفاوت ابعاد و تاری بروی تصویر ورودی برای توصیف عکس بسیار پر هزینه و زمان‌بر می‌باشد ولو آن که قدرت بالایی در استخراج ویژگی به نمایش می‌گذارد؛ طبق مقایسه‌ی انجامی(در عنوان زمان‌ها درج شده است) به جهت سرعت و زمان اجرا، الگوریتم FREAK با تفاوت فاحش نزدیک به ده برابر سریع‌تر از الگوریتم SIFT ظاهر شده است چرا که پردازش‌های سنگینی نظیر SIFT نداشته و صرفا با چند نقطه همسایگی میتواند تصویر را توصیف نماید.

به جهت عملکرد و دقت نیز هر دو الگوریتم FREAK و SIFT نتیجه مشابه به همی ارائه می‌کند و نمیتوان تفاوت فاحش بین این دو قائل شد؛ در حالتی که تصویر دارای چرخش خیلی زیادی نباشد، هر دو الگوریتم تا دو رقم اعشار زاویه درست و دقیقی را برای بازسازی بدست می‌آورند اما وقتی چرخش زیاد باشد حداکثر خطای SIFT برابر 0.01 درجه می‌باشد و این در حالتی است که FREAK تا خطای 0.05 درجه میتواند انحراف داشته باشد(هر چند به جهت شهودی و منطقی هر دو قابل قبول هستند)؛ در مقایسه‌ی دیگری اگر تعداد نقاط استخراجی و تطبیق یافته توسط این دو توصیف‌گر را مورد مقایسه قرار دهیم، تعداد نقاط تطبیقی SIFT بسیار بیشتر از تعداد نقاط تطبیقی FREAK می‌باشد و این باعث می‌شود که عملکرد و نتیجه‌ی SIFT دقیق‌تر و اتکاپذیرتر حاصل شود(فارغ از اینکه باعث کندی می‌شود؛ زیاد بودن تعداد نقاط میتواند نکته‌ی مثبت دیگری نیز داشته باشد و آن نیز این است که میتواند در مقابل نویز مقاومت ایجاد کرده و عملکرد بهتری را ارائه کند.طبق نتایج حاصل ملاحظه می‌کنیم که هر دو الگوریتم در مقابل چرخش مقاوم هستند و میتوانند هر درجه چرخش را بازسازی کنند و اگر خطای زاویه بازسازی را مد نظر قرار دهیم، الگوریتم SIFT در مقابل چرخش مقاوم‌تر می‌باشد.

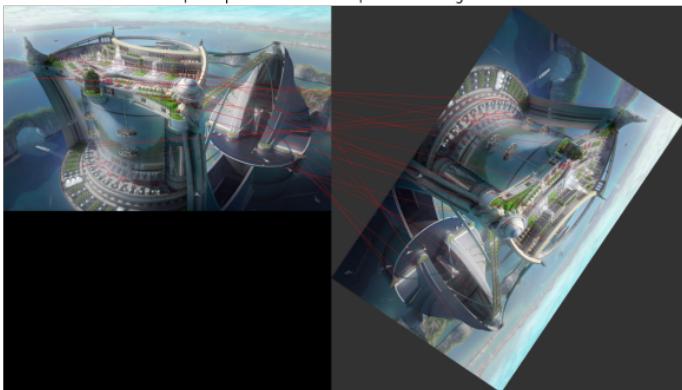
Original Image



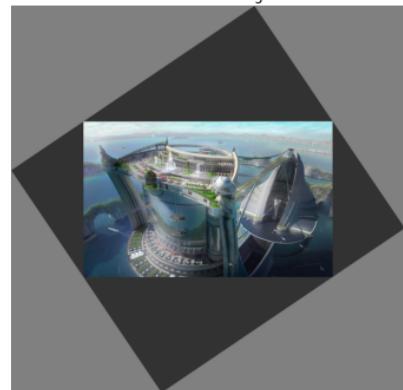
Rotated Image | Applied angle: 236



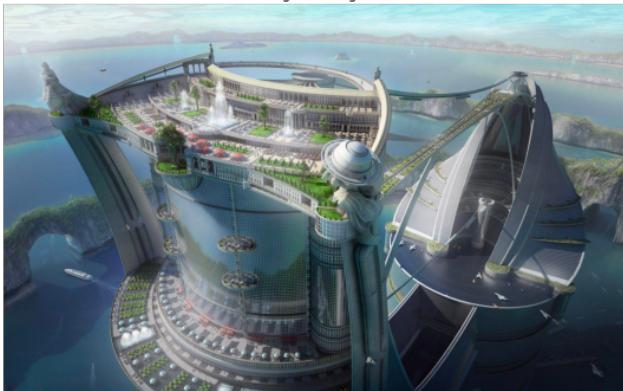
FREAK Matches | Comput Time: 0.1504s | Detected angle to restore: 124.062



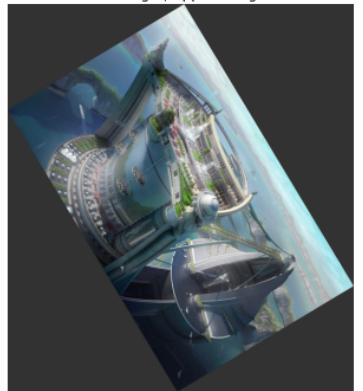
Re-Rotated Image



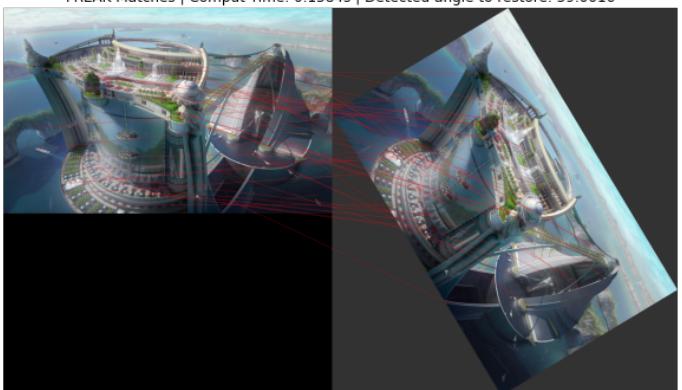
Original Image



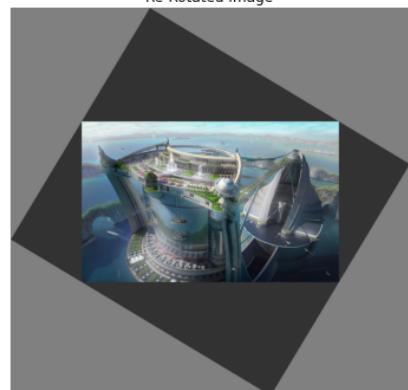
Rotated Image | Applied angle: 301



FREAK Matches | Comput Time: 0.1584s | Detected angle to restore: 59.0016



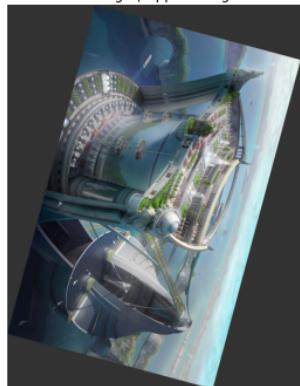
Re-Rotated Image



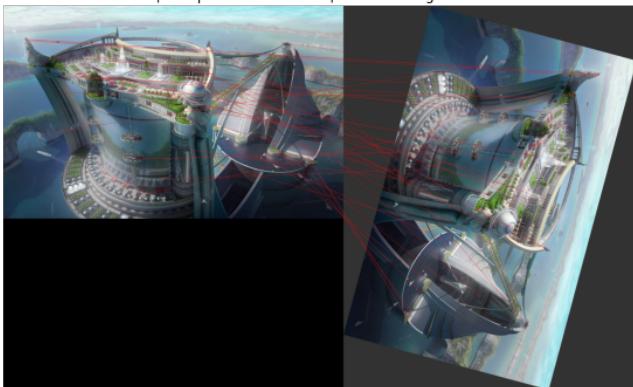
Original Image



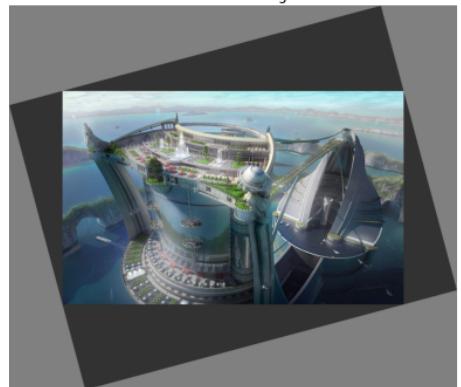
Rotated Image | Applied angle: 255



FREAK Matches | Comput Time: 0.1394s | Detected angle to restore: 104.9803



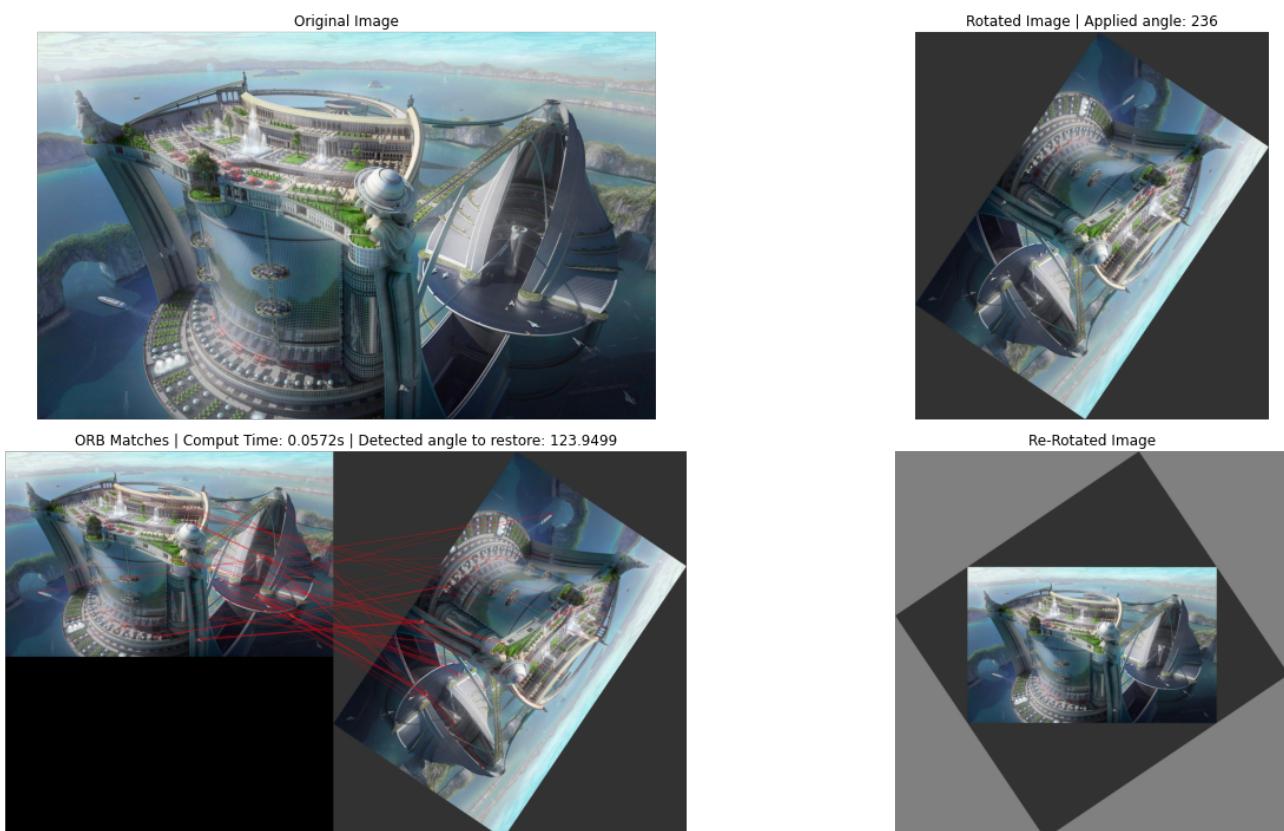
Re-Rotated Image

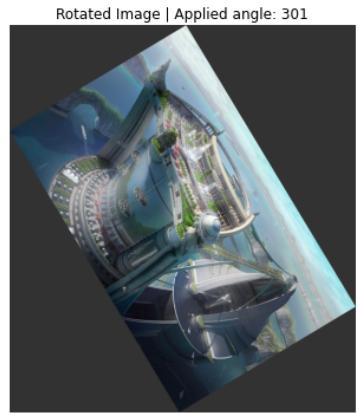


فعالیت بیشتر

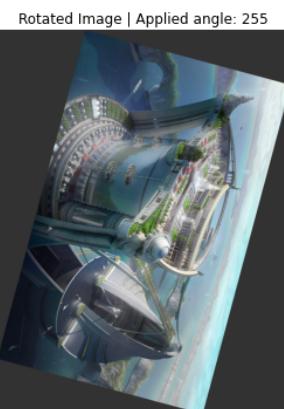
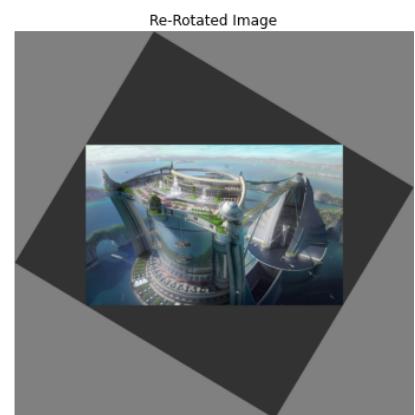
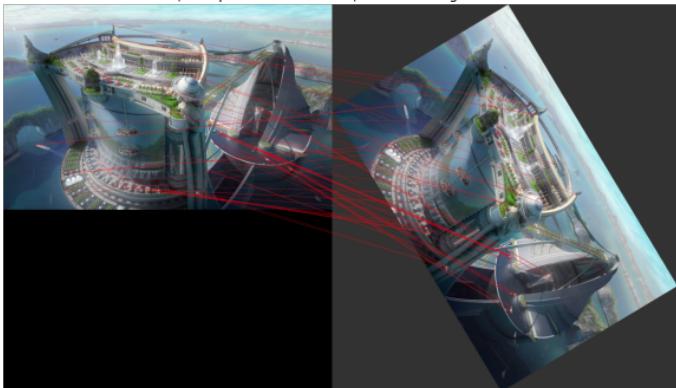
فعالیت بیشتر: توصیف‌گر ORB

در این قسمت و به عنوان فعالیت بیشتر از توصیف‌گر ORB از کتابخانه open-cv استفاده شده است تا بتوان تصویر چرخش یافته را بازسازی نمود؛ این توصیف‌گر که عبارت کامل آن Oriented FAST and Rotated BRIEF می‌باشد که مبتنی بر تشخیص نقاط شاخص با استفاده از الگوریتم FAST (که در کلاس درس نیز معرفی شده است) و توصیف‌گر BRIEF می‌باشد که در مقابل چرخش مقاوم شده اند و مستقل از دوران هستند. نتایج مربوط به این توصیف‌گر نیز در زیر آورده شده است؛ به جهت زمان و سرعت عملکرد بسیار بهتری از هر دو الگوریتم SIFT و FREAK داشته است. (در حد ۱۵ برابر سریع‌تر از SIFT و در حد ۵ برابر سریع‌تر از FREAK) به جهت دقت نیز حداکثر در حد ۰.۰۲ درصد تفاوت در تخمین زاویه صحیح خطا داشته است که قابل قبول است.

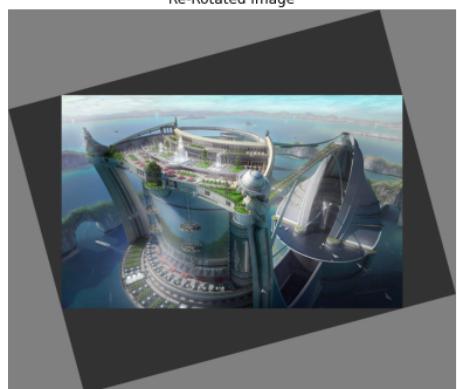
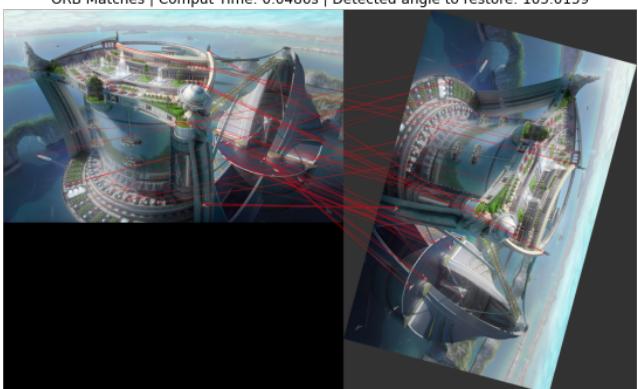




ORB Matches | Comput Time: 0.0559s | Detected angle to restore: 59.021

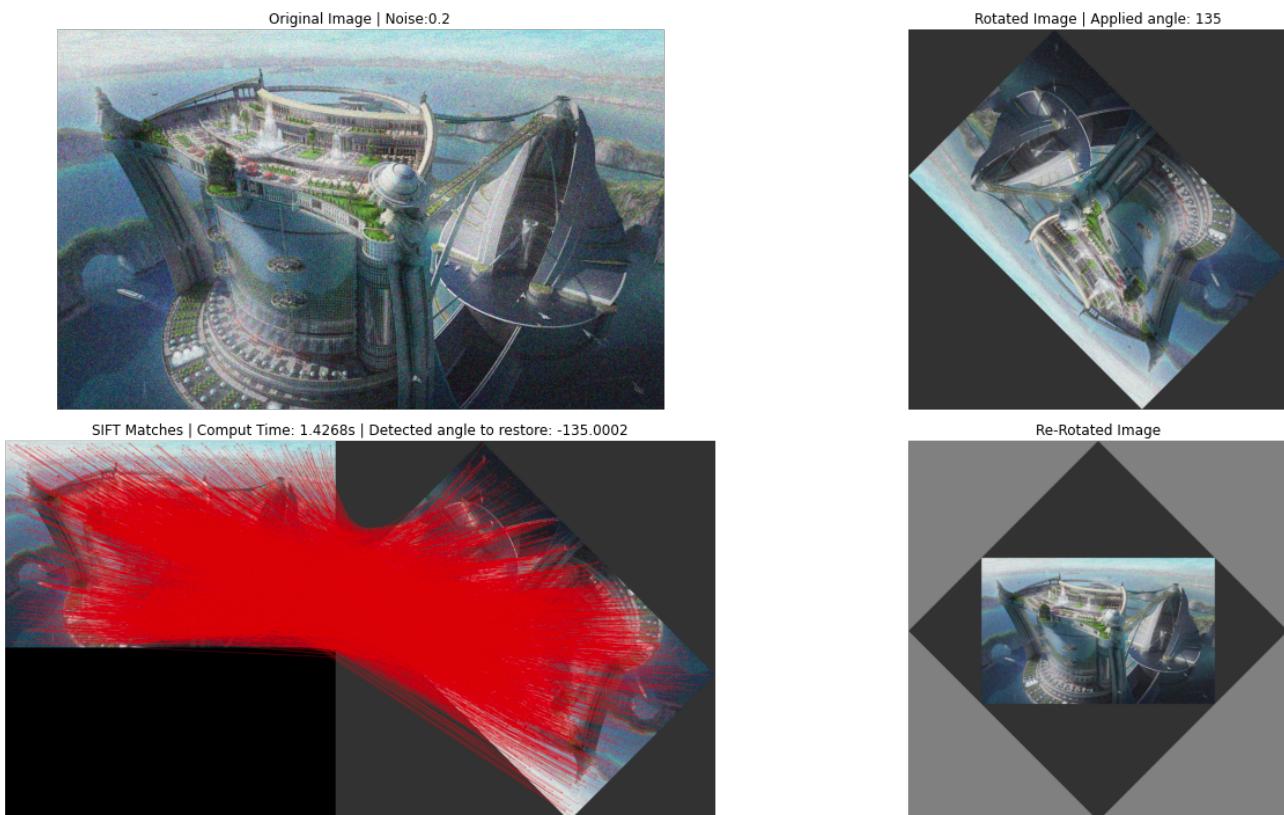


ORB Matches | Comput Time: 0.0486s | Detected angle to restore: 105.0159



فعالیت بیشتر: مقاومت در برای نویز

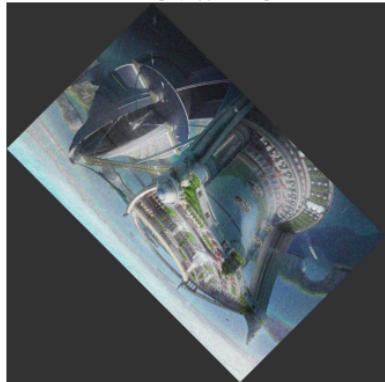
در این قسمت و به عنوان فعالیت بیشتر میخواهیم مقاومت الگوریتم های SIFT و FREAK را در مقابل نویز ارزیابی کنیم. برای این هدف، تصویر ورودی را توانان نویز اضافه کرده و چرخش می دهیم؛ نویز اعمالی یک نویز نرمال می باشد و شرایط اعمال و اجرای توصیفگرها تغییر پیدا نکرده اند. به ازای ضرایب 0.1 و 0.2 روی تصاویر نویز اعمال کرده و تطبیق نقاط شاخص و باز چرخش را انجام می دهیم؛ نتایج به صورت زیر حاصل می شود. طبق توضیحات مورد اشاره در مسئله سوم، زیاد بودن تعداد نقاط شاخص در SIFT در مقابل FREAK میتواند نکته‌ی مثبت مبنی بر مقاومت در مقابل نویز داشته و عملکرد بهتری را ارائه کند. می‌بینیم که تعداد نقاط شاخص و تطبیقی در FREAK در حضور نویز بشدت کاهش می‌یابد تا حدی که به ازای حضور نویز با ضریب 0.2 الگوریتم FREAK نتوانسته تصویر را باز چرخش کند لذا در حضور نویز، الگوریتم SIFT کاملاً بهتر ظاهر شده و موفق‌تر از FREAK می‌باشد.



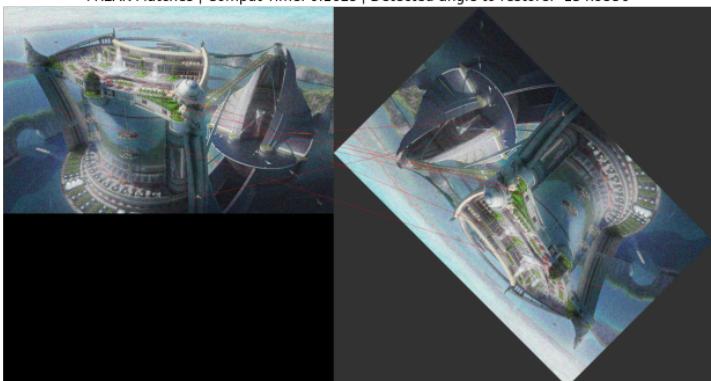
Original Image | Noise:0.2



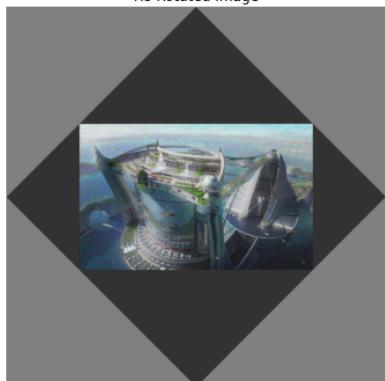
Rotated Image | Applied angle: 135



FREAK Matches | Comput Time: 0.162s | Detected angle to restore: -134.9556



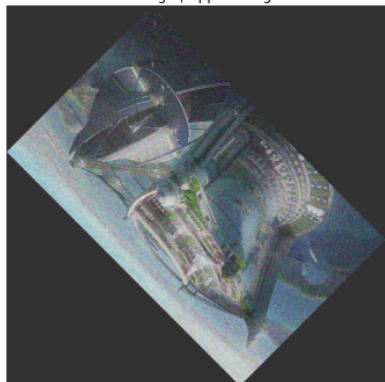
Re-Rotated Image



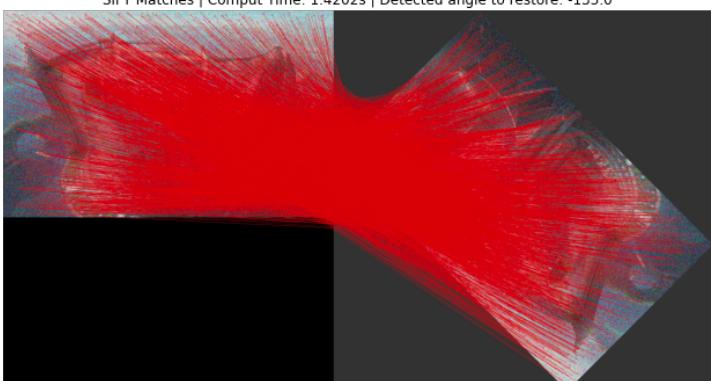
Original Image | Noise:0.4



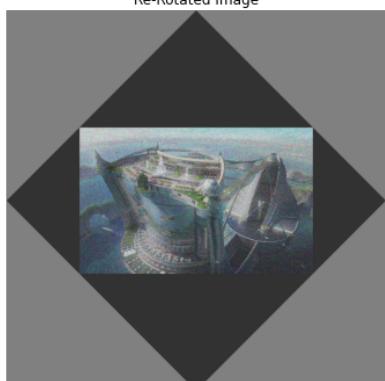
Rotated Image | Applied angle: 135



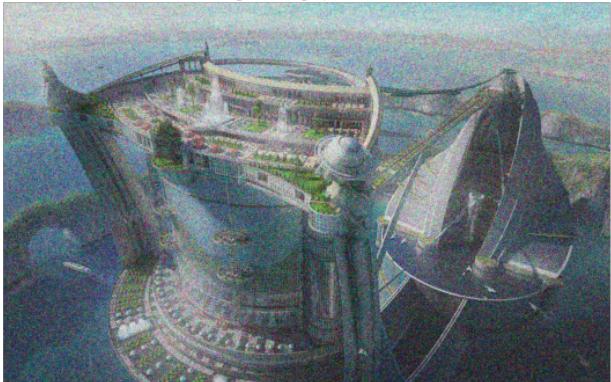
SIFT Matches | Comput Time: 1.4202s | Detected angle to restore: -135.0



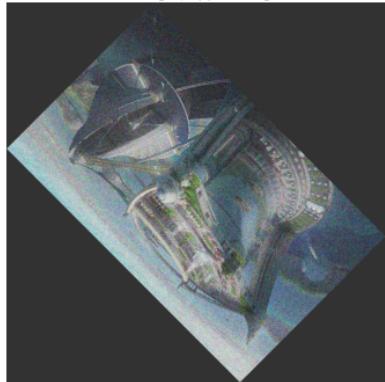
Re-Rotated Image



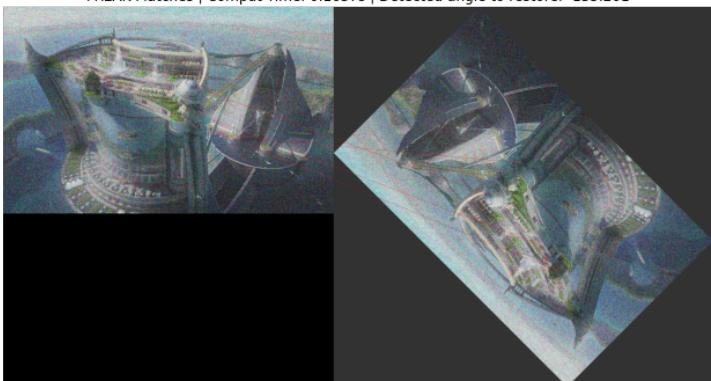
Original Image | Noise:0.4



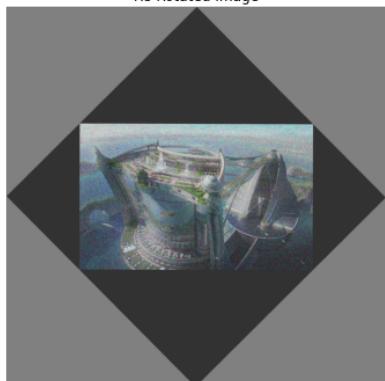
Rotated Image | Applied angle: 135



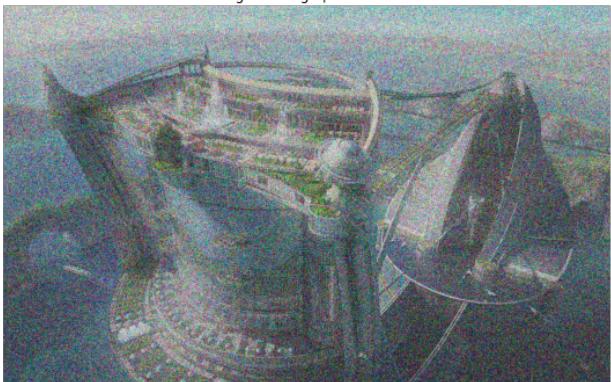
FREAK Matches | Comput Time: 0.1637s | Detected angle to restore: -135.201



Re-Rotated Image



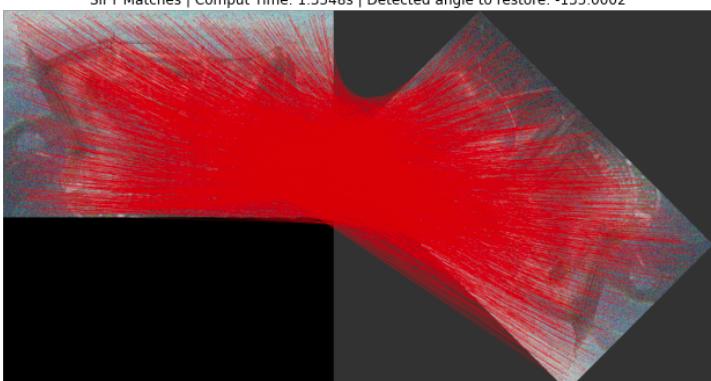
Original Image | Noise:0.6



Rotated Image | Applied angle: 135



SIFT Matches | Comput Time: 1.3548s | Detected angle to restore: -135.0002



Re-Rotated Image

