یاسمین سرچشمه‌پور

تمرین دوم

۹۴۱۰۹۸۲۸

۱. همانطور که در صورت سوال ذکر شده است $interest\ point$های بدست آمده در عکس $res۰۱.png$ ضمیمه شده است که برای بدست آوردن آنها از $cv۲.orb$ استفاده شده است.

سپس با استفاده از $cv۲.sift$ نقاط متناظر را پیدا کردیم که در عکس $res۰۲.png$ ذخیره شده است.

با کمک $matplotlib$ نقاط متناظر را بهم وصل کردیم و عکس به دست آمده را در $res۰۳.png$ ذخیره کردیم.

حال ۲۰ نقطه اول متناظر را در $res۰۴.png$ نمایش دادیم.

سپس با کمک $skimage$ و $ransak$ ماتریس فاندامنتال و $inlier$ها را پیدا به دست آوردیم که نقاط $inlier$ در تصویر $res۰۵.png$ ذخیره شده‌اند.

$$N = \frac{log(۱ - p)}{log(۱ - w^s)}$$

فرض می‌کنیم $p = ۰.۹۹, s = ۸$ و می‌دانیم $w = ۰.۳$ می‌باشد. با توجه به فرمول به دست می‌آید $N = ۷۰۱۸۷.۸$ که در حین زدن $ransac$ آپدیت می‌شود.

برای بررسی نقاط $inlier$ تک تک آنها را در تصویری که در پوشه‌ی $inlier$ می‌باشد ذخیره کرده‌ایم و همانطور که مشاهده می‌شود بعضی از آن‌ها اشتباه هستند که در پوشه‌ی $inlier$ می‌توان آن‌ها را دید.

برای بررسی نقاط $outlier$ تک تک آنها را در تصویری که در پوشه‌ی $outlier$ می‌باشد ذخیره کرده‌ایم و همانطور که مشاهده می‌شود بهضی از آن‌ها درست هستند و نباید $outlier$ می‌بودند.

ماتریس فاندامنتال ماتریس زیر می‌باشد.

$$\begin{matrix} 6.37440503e-07 & -9.42484957e-07 & 1.72297021e-03 \\ 1.48420269e-06 & 1.39967505e-06 & -3.22057002e-03 \\ -4.78575160e-03 & -2.46856699e-03 & 6.27029668e+00 \end{matrix}$$

میانگین $inlier$ ارور : ۰,۷۸۸۸۳۸۰۱۲۱۱۱

ارور $inlier$کمینه:۰,۰۸۳۰۲۶۶۸۴۶۱۹۶

ارور$inlier$ بیشینه: ۳,۱۵۹۸۳۸۰۳۳۲۲

انحراف از معیار$inlier$ ارور: ۰,۷۲۲۲۱۸۲۶۳۴۴۳

میانگین $outlier$ ارور : ۱,۸۳۶۰۲۲۴۱۵۴۳

ارور $outlier$کمینه:۰,۰۰۷۳۶۴۳۸۴۳۱۷۷

ارور$outlier$ بیشینه: ۸,۸۵۷۶۱۴۳۰۳۴۵

انحراف از معیار$outlier$ ارور: ۱,۸۳۹۴۶۷۰۹۸۲۹

برای محاسبه‌ی نقاط $epipolar$ از ماتریس F و روش svd استفاده کرده ایم.

که $apipolar\ point$ عکس اول برابر است با:

(-2030.5819982032838, -1.4286479858064898)

و برای عکس دوم داریم:

(1324.5654243166346, 2655.5806548337127)

برای ران کردن کد این قسمت $python\ prob۱.py$ را ران کنید.(دقت داشته باشید که عکس ها باید موجود باشند)

۲. ابتدا ترتیب مناسب عکس‌هارا پیدا کردم، سپس ماتریس تبدیل هر عکس با عکس مجاور آن را در تابع $get\_sift\_homography$ محاسبه کردم که در آن با استفاده از $ransac$ و نقاط به دست آمده از $sift$ ماتریس $homography$ به دست می‌آید.

سپس، ماتریس تبدیل هر عکس‌را به عکس مرجع که عکس چهارم می‌باشد محاسبه می‌کنیم به این صورت که به ازای هر عکس i به طوری که $i > ۴$ باشد $H_{۴,i} = H_{۴,i-۱} \times H_{i_۱,i}$ و به ازای $i = ۴$ داریم $H_{i,۴} = I$ و به ازای هر $i < ۴$ داریم $H_{۴,i} = H_{i,۴}^{-۱}$ که در آن $H_{i,۴} = H_{i,i+۱} \times H_{i+۱,۴}$ می‌باشد. سپس در تابع $get\_stitched\_images$ عکس‌هارا در یک صفحه به صورت پاناروما نمایش می‌دهیم.

در این تابع‌های $get\_stitched\_images\_mean$ و $get\_stitched\_images\_blend$ با استفاده از ماتریس‌های تبدیل و ترتیب عکس‌ها مختصات نقاط گوشه‌ای عکس را بدست می‌آوریم و برای حذف مختصات منفی همه‌ی

عکس‌ها را به اندازه گوشه‌های منفی $shift$ می‌دهیم که این کار را با کمک ماتریس $transform\_array$ انجام می‌دهیم. سپس با کمک ماتریس تبدیل ($H_{i,۴}$) و $transformed\_array$ جای هر عکس را در صفحه‌ی پاناروميک به دست می‌آورم و با استفاده از توابع $mean\_normalize\_color$ و $blend\_normalize\_color$ اختلاف رنگ‌هارا درست می‌کنیم.

تابع $mean\_normalize\_color$: در این تابع پس از اضافه کردن هر عکس به ازای هر پیکسل در عکس جدید مقدار آن را در صورت وجود در عکس قبلی برابر با میانگین این دو نقطه می‌گذاریم.

تابع $get\_stitched\_images\_blend$ در این تابع به ازای هر پیکسل نگه می‌داریم کدام پیکسل‌های عکس‌ها روی این پیکسل قرار می‌گیرند و به هر پیکسل وزنی برابر با فاصله‌ی آن عکس با پس زمینه می‌دهمیم و میانگین وزن دار این پیکسل‌ها را محاسبه می‌کنیم. برای محاسبه‌ی نزدیک ترین نقطه‌ی پس زمینه برای هر پیکسل در عکس‌ها از داینامیک استفاده می‌کنیم.(در تابع‌های $get\_nearest\_white\_up\_left$ ، $get\_nearest\_white\_up\_right$، $get\_nearest\_white\_down\_left$، $get\_nearest\_white\_down\_right$)

برای ران کردن کد این قسمت $python\ prob۲.py$ را ران کنید.(دقت داشته باشید که عکس ها باید موجود باشند)

۳. ابتدا دو نقطه نمایانگر قد خانم که در بالا و پایین تصویر او می‌باشد و دو نقطه نمایانگر قد دیوار در تصویر ۳۱۱.JPG مشخص می‌کنیم سپس با کمک خطوط موازی در دو جهت مختلف ۲ $vanishing\ point$ به دست می‌آوریم و با استفاده از این دو نقطه $vanishing\ line$ را پیدا می‌کنیم.

می‌دانیم که همه‌ی $vanishing\ point$ها روی این خط هستند حال با توجه به مباحث گفته شده در کلاس برای بدست آوردن ارتفاع واقعی دیوار نقطه‌ی تقاطع خط واصل پایین دیوار و پایین خانم و $vanishingline$ به دست می‌آوریم و آن را v می‌نامیم. حال از نقطه‌ی v به بالای سر خانم وصل کرده و برخورد آن خط را با خط مربوط به قد دیوار را t می‌نامیم. حال اگر نقطه‌ی پایینی دیوار b و نقطه‌ی بالایی آن r باشد و ارتفاع واقعی خانم H و ارتفاع واقعی دیوار R باشد، می‌دانیم تمام $vanishing\ point$ها از $vanishing\ line$ می‌گذرند پس برای به دست آوردن v_z (نقطه‌ی افق در راستای محور z) تقاطع خط دیوار با $vanishing\ line$ را محاسبه می‌کنیم.

حال برای محاسبه‌ی R با توجه به فرمول داریم.

$$R = H \times \frac{|b-t||v_z-r|}{|b-r||v_z-t|}$$

برای پیاده‌سازی این روش ابتدا ۴ نقطه‌ی گفته شده را به دست می‌آوریم سپس با کمک $hough$ و $ransac$ دو $vanishing\ point$ در راستای مختلف را به دست می‌آوریم به این صورت که ابتدا لبه‌های عکس را به دست می‌آوریم و به صورت تصادفی دو تا از آن‌ها که موازی هستند را انتخاب می‌کنیم و نقاط تقاطع آن را به دست می‌آوریم و بررسی می‌کنیم که چند خط دیگر از این نقطه می‌گذرند حال اگر خطوط قابل ملاحظه‌ای ازین نقطه می‌گذشتند این نقطه را به عنوان $vanishing\ point$ انتخاب می‌کنیم سپس این خطوط را که خطوط $inlier$ هستند از لبه‌ها حذف می‌کنیم و سپس روی لبه‌های به دست آمده‌ی جدید همین کار را تکرار می‌کنیم تا $vanishing\ point$ دیگر به دست بیاید. حال به توجه به این دو نقطه و نقاط دیوار و خانم و v_z به دست آمده ارتفاع واقعی دیوار را همانطور که گفته شد محاسبه می‌کنیم. در ۳ راستا $vanishing\ point$ را انتخاب می‌کنیم و ارتفاع دیوار را با استفاده از $vanishing\ line$ ۲ به دوی آن‌ها محاسبه می‌کنیم و بیشترین مقدار برابر با پاسخ مسئله می‌باشد چرا که در بدترین حالت یکی از راستاها راستای z می‌باشد و عدد بدست‌امده کم می‌شود. برای روش اتوماتیک $python\ prob۳.py$ را ران کنید

و برای روش دستی $python\ prob۳\ manual$ را ران کنید

مشاهده می‌شود که عدد بدست آمده تقریبا ۶ متر می‌باشد.