در ابتدا تمام فریم های ویدیو جدا شده و در پوشه ی frames ذخیره شده است.

چند پوشه خالی در فایل ها جومود است که با ران کردن کد ها پر میشوند و از محتویاتشان استفاده میشود. در هر بخش از نتایج و داده های بخش قبلش استفاده هایی شده پس ترتیب اجرای کد ها برای صحت اجرا شدندشان اهمیت دارد

بخش 1:

در این سوال یک تابع پیاده سازی شده به نام warp که دو تصویر را میگیرد و هوموگرافی مناسب برای نگاشت تصویر اول به تصویر دوم را بعنوان خروجی بازگشت میدهد. در این تابع از توابع آماده sift برای نقاط متناظر و تناظر ها و از ransac برای بدست آوردن هوموگرافی استفاده شده. در بخش های بعدی این سوال نیز از این تابع warp استفاده شده است که از توضیح مجدد آن خودداری شده است. ابتدا چهار نقطه ی دلخواه که تشکیل یک مربع میدهند را از تصویر دوم یعنی فریم 450 انتخاب میکنیم و روی آن رسم میکنیم، سپس هوموگرافی مورد نیاز از فریم 270 به 450 را با تابع warp که بالاتر توضیح داده شد، میابیم و این چهار نقطه را تحت این هوموگرافی نگاشت میدهیم و با نقاط حاصل یک چند ضلعی در تصویر اول یعنی فریم 270 رسم میکنیم. سپس یک pad به ضخامت 2000 پیکسل به چپ و راست و 500 به بالا و پایین هر فریم اضافه میکنیم. سپس بار دیگر هوموگرافی را برای تصاویر padding شده پیدا کرده و روی فریم 270 پَد شده اثر میدهیم. سپس برای آنکه این دو تصویر را کنار هم قرار دهیم از تابع put که خودمان پیاده کرده ایم استفاده میکنیم. این تابع دو تصویر میگیرد که تصویر دوم را بعنوان background در نظر گرفته و پیکسل های غیر سیاه (مخالف [0,0,0]) تصویر اول را روی تصویر بک گراند قرار میدهد. پس با این کار میتوانیم ابتدا یک صفحه کاملا سیاه بسازیم، سپس فریم 270 پد دار وارپ شده را با تابع put روی صفحه سیاه قرار دهیم. سپس فریم 450 پد دار را با تابع put روی آن قرار میدهیم و تصویر حاصل خروجی مطلوب سوال است.

بخش 2:

در این سوال فریم های 90، 270، 450، 630، 810 را تصاویر کلیدی مینامیم و به ترتیب با متغیر I1، I2، I3، I4، I5 مینامیم. به اطراف هر یک pad به ضخامت افقی 2000 و ضخامت عمودی 1000 اضافه میکنیم. سپس هوموگرافی تصویر اول به دوم را محاسبه کرده و H12 مینامیم و سپس هوموگرافی تصویر دوم به سوم را محاسبه کرده و H23 مینامیم.

پس هوموگرافی تصویر اول به سوم برابر میشود با حاصل ضرب H12 در H23 که آن را H13 مینامیم. به همین روش (غیر مستقیم) هوموگرافی تصویر پنجم به سوم را میابیم. ولی هوموگرافی تصویر دوم و چهارم به تصویر سوم، به صورت مستقیم قابل محاسبه است. اکنون برای blend کردن تصاویر از هرم لاپلاسین استفاده شده که نحوه پیاده سازی آن در گزارش تمرین سری آخر پردازش تصویر ترم گذشته آورده شده. عملکرد کلی آن به این صورت است که دو تصویر را بهمراه یک ماسک گرفته و تحت این ماسک دو تصویر را ادغام میکند. مسئله مهم در اینجا یافتن ماسک مناسب برای ادغام تصاویر است. برای این کار به طور شهودی مسک به شکل زیر ساخته شده. یعنی دو نقطه ی تلاقی بین دو صفحه تصویر (نقاط قرمز)

یافت شده، و خط گذرنده از آن دو نقطه را بدست آورده

و یک سوی این خط را سفید و یک سوی دیگر را مشکی کرده

که تصویر حاصل ماسک ما است. برای بدست آوردن دو نقطه تلاقی، ابتدا برای هر یک از دو فریم پد دار، یک ماسک بدست میاوریم (ناحیه های آبی در شکل بالا) و با تابع canny اطراف این ماسک ها را بدست میاوریم (مرز سبز رنگ) سپس مقادیر این دو مرز را باهم جمع میکنیم و آن نقاطی که عدد بزرگی شده اند نشان دهنده این اند که مرز هر دو تصویر از آن دو نقطه عبور کرده اند که آن را نقطه تلاقی مینامیم. البته گاهی چند نقطه بسیار نزدیک به هم بعنوان نقطه تلاقی تشخیص داده میشدند که با تابع cleanList که خودمان پیاده سازی کردیم، از بین نقاط نزدیک بهم یکی را نگه داشته و باقی را حذف میکنیم. حال دو نقطه تلاقی داریم که خط گذرنده از آن دو را ساخته و یک سمت آن را سفید و یک سمت دیگر را سیاه کرده و از آن برای ادغام با هرم لاپلاسین استفاده میکنیم. این کار ها را تکرار میکنیم تا هر 5 تصویر ادغام شوند و نتیجه همان خروجی مطلوب است.

بخش 3:

در این بخش از pad افقی با ضخامت 2500 و pad عمودی 1000 پیکسل استفاده شده است. ابتدا ابعاد تصاویر را تقسیم بر k میکنیم و هوموگرافی را برای عکس های 1/k ام شده حساب کرده و H مینامیم. حال میدانیم هوموگرافی برای سایز اصلی از رابطه ی SHS^{-1} بدست میاید که در آن S برابر است با:

$$S = \begin{bmatrix} k & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

حال هوموگرافی هر 5 فریم کلیدی را حساب کرده و فریم های 0 تا 180 را با کمک فریم 90 و 180 تا 630 را با کمک فریم 270 و... نگاشت میدهیم و فریم های حاصل را در پوشه warped frames قرار داده و با استفاده از آنها ویدیو میسازیم. همچنین مقادیر درایه های هوموگرافی ها را در فایل تکست ذخیره میکنیم.

بخش 4:

در اینجا ابتدا فریم های وارپ شده بدست آمده در بخش قبل را به نوار های افقی به ضخامت 10 پیکسل برش میدهیم. سپس در هر گام نوار i ام تمام فریم ها را روی هم قرار داده و تابع np.median را با apply_along_axis در راستای محور 0 (که محور شماره فریم هاست) اعمال میکنیم. البته این اعمال median بر روی داده های ناصفر اثر میکند تا در جاهایی که فریم ها مقداری ندارند (به دلیل وجود pad) این مقادیر سیاه pad محاسبه نشوند. این گونه با فرض اینکه بیش از 50 درصد اوقات ما شاهد رنگ بک گراند بودیم، رنگ بک گراند تخمین زده میشود.

بخش 5:

در بخش ابتدا وارون ماتریس های هوموگرافی بدست آمده در بخش 3 را حساب کرده و سپس هر بار پانوراما‌ی بدست آمده در بخش قبل را وارپ کرده و نتیجه را قدری برش میدهیم که تصویر حاصل pad نداشته و هم سایز با فریم های ویدیوی اصلی شوند.

بخش 6:

در این بخش هر فریم از ویدیوی اصلی را منهای فریم متناظرش در ویدیوی بک گراند کرده و مجذور آن را حساب میکنیم و مقادیر بالای threshold=30000 را foreground نامیدیم. البته در این بخش چند ایده برای کاهش نویز پیاده سازی شد که فرق چندانی بین آنها نبود و هر یک معایب و مزایایی داشت که در نهایت پیاده سازی یکی از آنها در میان فایل ها قرار گرفت. میدانیم نویز ها بدلیل لرزش دوربین و خطای بخش های قبل (میانه گیری یا محاسبه هوموگرافی و...) ناشی شده اند.

ایده اول این بود که بر فریم های اصلی و فریم های بک گراند، فیلتر گوس اعمال کنیم تا به لبه های تصاویر حساس نباشند. برای مثال لبه های ساختمان در صورت عدم انطباق نباید نویز تشخیص داده شوند زیرا کلیت ساختمان حفظ شده. ولی ماشین ها به دلیل بزرگ بودنشان بعد اعمال فیلتر گوس همچنان با پس زمینه انطباق نخواهند داشت و فورگراند محسوب میشوند.

ایده دوم این بود که لبه ی های تصویر بک گراند را با استفاده از لاپلاسین حساب کنیم و اجازه ندهیم این بخش ها و اطرافشان در صورت عدم انطباق جز فورگراند تشخیص داده شوند. که این موضوع باعث شد ماشین ها نیز به خوبی تشخیص داده نشوند.

ایده سوم که در فایل کد از این ایده استفاده شده، وارپ کردن تصویر اصلی به تصویر بک گراند بود که این کار باعث میشدخطای ناشی از هوموگرافی ها که ناشی از بخش 3 بود کاهش یابد که این کمک به کمتر شدن نویز ها کرد.

بخش 7:

این بخش همانند بخش 5 است با این تفاوت که پهنای پانوراما‌ی وارپ شده را بزرگ تر در نظر میگیریم تا wide تر بنظر آید و همچنین برای آنکه سیاهی اطراف در فریم های اخر به چشم نخورد، این شرط را میگذاریم که اگه اخرین ستون تصویر فریمی کاملا سیاه بود آن فریم را به ویدیو اضافه نکن.

بخش 8:

در این بخش تمام هوموگرافی های بدست آمده در بخش 3 را روی هم قرار میدهیم (در یک آرایه به صورت np.zeros((900,9)) نگه میداریم سپس با cv2.blur با ksize=(1,41) در واقع نوعی میانگین گیری بین درایه های ماتریس هوموگرافی انجام میدهیم تا لرزش ناشی از چند فریم، با توسط فریم های اطرافش خنثی شود.