توضیحات سوال 2 texture synthesis:

توضیحات کلی:

در ادرس دادن آدرس نسبی استفاده شده است و فرض شده تمامی فایل های تکسچر در همان فولدر بدون فولدربندی قبلی هستند.و خروجی ها هم در همان جا سیو میشوند. فایل های نتیجه علاوه بر نمایش در یک قاب با تکسچر به صورت جدا نیز آمده اند. جواب های res11 و res12 از پوشه گفته شده امده است.(بافت انار و بافتی که شبیه توپ گلف است texture03 و دوبافت بعدی از اسلایدها انتخاب شده اند. بافت نان و بافت طناب مانند.)

توضیح الگوریتم:

نکته1: سمپلی که از texture برمیداریم، یک مربع به اندازه sample_len است که معمولا یک دوم مینیمم طول و عرض texture است. و مقدار overlap_len هم یک مربع به اندازه یک سوم سمپل است.

ما اول یک سمپل به صورت رندم در بالا سمت چپ می گذاریم. (از تابع random_initiall_patch استفاده می کنیم و عملکردش واضح است دو عدد تصادفی در بازه مربوطه میگیرد و patch مربوط را خروجی می دهد.) وسپس برای پچ بعدی در راستای افقی که با پچ قبلی یک مستطیل عمودی اشتراک دارد. طول مستطیل که باید بیابید i*(sample_len – overlap_len) است و برای عرض هم مشابه همین است.ما حالا از این پچ یک پچ از شکل اصلی پیدا میکنیم که شباهت داشته باشد. برای این کار template matching انجام میدهیم. (از تابع similar_patch استفاده میکنیم که در ابتدا از تابع matchtemplate که مدل ان با ماسک است که در لایبرری opencv پیاده شده است. سپس نرمالیزیشن به روش minmax انجام میدهیم و از بین 10 تا مقدار بیشترین در ارایه خروجی مچ تمپلیت به تصادف یک مقدار را پیدا کرده و اندیس ان را خروجی میدهیم.) و ماسک مربوط به تمپلیت مچینگ باتوجه به اینکه قسمت اشتراک عموی یا افقی یا جفت این حالات باشد ماسک مربوط ساخته شده وبه تابع داده میشود. این کار ها در تابع random patch انجام میشود.

حال برای اینکه بدانیم از هر دو پچ داشته شده در قسمت اشتراک چجوری برداریم، از الگوریتم مطرح شده در کلاس به روش برنامه ریزی پویا (Minimize Cut Problem) استفاده میکنیم اول فاصله اقلیدسی بین این دوتا پچ را در فضای RGB بدست اورده یعنی این دو پچ که رنگی هم هستند و عمق 3 دارند را از هم کم کرده و در ارایه جدید هر نقطه iوj نرم بردار می آید .(برای محاسبه این آرایه از تابع np.linalg.norm(x,axis=2) استفاده کرده ام.)و حالا مسیر طبق الگوریتم مطرح شده در کلاس، مسیر از بالا به پایین حساب می کنیم.(در تابع vertical_minimize_cut انجام شده است). یک ارایه ماسک میسازیم و به ازای هر سطر تمام پیکسل های سمت چپ پیکسل مشخص شده در سطر را یک میگذاریم و این ماسک را خروجی میدهیم. (vertical_minimize_mask) سپس در تابع suitable_cut این ماسک را با یک فیلتر گاووسی ترکیب میکنیم فیلتر ما 9 در9 است. و وقتی در کلاس با استاد مطرح کردم کار درستی است و به آن feathering میگویند . (همچنین اگر اشتراکمان افقی باشد بازم مانند قبل از تابع های قبلی استفاده شده فقط ترانهاده ارایه فاصله اقلیدسی را در نظر گرفته و ماسک ایجاد شده را نیز ترانهاده میگیریم. در ماسک انتهایی که ترکیب عمودی و افقی است را در هر درایه که حداقل یکی از این دو ماسک 1 باشد ، 1 و درغیر این صورت 0 می گذاریم. و سپس فیلتر گاووسی را اعمال میکنیم.) ماسک بدست آمده در این تابع را در پچ قدیمی و (1-mask) را در پچ جدید ضرب کرده و این، ترکیب این دو پچ را میسازند. در واقع اینکار به معنای این است که درایه های 1 در ماسک را از درایه متناظر در پچ قدیمی و درایه های 0 را متناظرا از پچ جدید برداریم. به همین شکل جلو میرویم و سمپل های ساخته شده را در عکس خالی اولیه قرار میدهیم. (این کار در تابع synthesis انجام شده است.) سپس ان را سیو میکنیم. در این سیو کردن هم توسط matplotlib در یک قاب و هم جداگانه در یک فایل نگه میداریم که تصاویر مشخص تر باشند.و در کنار خروجی ها آمده اند.