دانشگاه صنتعی شریف

دانشکده مهندسی برق

پروژه درس سیگنال ها و سیستم

دکتر بابک حسین خلج

فاز اول ددلاین :1400/01/20

به نکات زیر توجه کنید:

- کلاس های توجیهی پروژه و دیگر مطالب آموزشی با توجه به کتابخانه های پایتون انجام می پذیرد. اما اجباری در انتخاب محیط برنامه نویسی برای شما برای انجام پروژه نمی باشد.یعنی می‌توانید از متلب هم برای انجام پروژه استفاده کنید.
- فایل تحویلی شما باید شامل سه بخش باشد :
  1. فایل کد ها
  2. نتایج گفته شده در هر قسمت به همان اسمی که گفته شده
  3. گزارش (نکته: در صورت استفاده از Notebook Jupyter یا LiveScript متلب نیازی به فایل جداگانه برای توضیحات نیست و کافیست در همان جا توضیحات مربوط به کدهایتان را بنویسید)
- گزارش باید شامل توضیح کد و مقایسه نتایج(در صورت وجود) باشد .
- در ابتدای هر فاز پروژه یک کلاس توجیهی برگزار می شود. علاوه بر آن ابهامات خودتان را هم می توانید از در گاهی که در CW درس ایجاد می شود مطرح کنید و هم با تی ای های پروژه درس بصورت ایمیل در میان بگذارید:
  1. امیررضا حاتمی پور arhp78@gmail.com
  2. سروین موتمن s.motamen@gmail.com
  3. کیمیا محسنیان mohseniankimia@gmail.com

1-1 Basics of image: (10 نمره)

هدف از این قسمت آشنایی اولیه با مفاهیم پردازش تصویر در پایتون می باشد.

یک تصویر به دلخواه خود انتخاب کنید . برای load و save تصاویر در پایتون می توانید از این لینک استفاده کنید.

```
#read image
target=cv2.imread("2.target.jpg")

#save image
dst=dst.astype("uint8")
cv2.imwrite("res2.jpg", dst)
```

- بعد از خواندن تصویر ، فیلتر های Mean و Gaussian را بطور جداگانه و مستقل از هم روی تصویر اعمال کنید و نتیجه را به ترتیب با نام های res01.jpg و res02.jpg ذخیره نمایید.(فایل کد را با نام q1_1.py ذخیره کنید.)

- تصویر noise.jpg وnoise1.jpg را بخوانید و با اعمال اندازه های مختلف از فیلتر Median آنرا denoising کرده و بهترین نتیجه را با نام res03.jpg ذخیره کنید.می توانید نتایج حاصل از فیلتر های مختلف را در گزارش خود بیاورید.(فایل کد را با نام q1_2.py ذخیره کنید.)

-

1-2 Template Matching: (20 نمره)

در این بخش هدف تشخیص یک شی در یک تصویر می باشد.می توانید برای این کار از توابع آماده opencv استفاده کنید.

ابتدا توصیه می شود که این لینک را مطالعه کنید.

- کدی بنویسید که تصاویر چرخ را با استفاده از توابع آماده در تصویر car.jpg پیدا کند و محل آنها را نمایش دهد.نتیجه را با نام Template1.jpg ذخیره کنید (فایل car.jpg و wheel.jpg در پوشه پروژه موجود می باشد)

- **همین کار را برای یک تصویر و یک patch به دلخواه خودتان انجام دهید و نتیجه را با نام Template2.jpg ذخیره کنید.**(فایل کد را با نام q2.py ذخیره کنید.)

1-3 : Line Detection (20 نمره)

در این قسمت باید با استفاده از توابع آماده پایتون خط های شکل را تشخیص دهید.

ابتدا توصیه می شود که این لینک را مطالعه کنید.

- یک تصویر به دلخواه خود انتخاب کنید . با استفاده از توابع آماده پایتون خط های این تصویر را شناسایی و مشخص کنید و نتیجه را با نام Line1.jpg ذخیره کنید

- **همان بخش قبل را برای تصویر Line.jpg تکرار کنید با این تفاوت که این بار مقادیر را به ازای minlength=10 , minlength=100, minlength=250 انجام دهید و نتایج حاصل از هر قسمت را به ترتیب با نام های res3_10.jpg , res3_100.jpg , res3_250.jpg ذخیره کنید.** (فایل کد را با نام q3.py ذخیره کنید.)

1-4:Hybrid Images (50 نمره)

هنگامی که از نزدیک به یک تصویر نگاه می کنیم جزئیات تصویر را می بینیم و هنگامی که از دور به آن نگاه می کنیم کلیات آن را مشاهده می کنیم. جزئیات یک تصویر معادل فرکانس بالای تصویر می باشد و کلیات آن معادل فرکانس های پایین آن .

تصاویر هیبریدی تصاویری می باشن که کلیات (فرکانس های پایین) آن از یک تصویر آمده و جزئیات(فرکانس های بالا) آن از تصویر دیگری آمده است.بطوری که اگر از دور به آن نگاه کنید یک تصویر را خواهید دید و اگر از نزدیک نگاه کنید تصویر دیگری را مشاهده می کنید.

بطور مثال:

یک جفت تصویر به دلخواه خود انتخاب کنید. تصاویر موجود در سایت ها استفاده نکنید. برای به دست آوردن تصویر هیبریدی مناسب، دو تصویر انتخاب شده باید از یک نوع باشند، برای مثال هر دو چهره انسان باشند، یکی دوچرخه و یکی موتور باشد، هر دو تصویر خودرو باشند، و یا هر دو تصویری که اجزای مشابه داشته باشند. تصاویر شما باید رنگی باشند. تصویر هیبریدی حاصل هم باید رنگی باشد.

تصویری که می خواهید از نزدیک دیده شود را near01.jpg و تصویری که می خواهید از دور دیده شود را far02.jpgبنامید

برای به دست آوردن نتیجه بهتر، باید دو تصویر را هم اندازه کرده و با هم منطبق نمایید، یعنی قسمت های مشابه یا معادل را روی هم قرار دهید. برای مثال اگر از دو تصویر صورت انسان استفاده می کنید این کار را می توانید با تطابق چشم های دو شخص روی هم انجام دهید.(از ابتدا سعی کنید که دو تصویری که انتخاب می کنید نزدیک هم باشن که نیاز چندانی به تطابق نداشته باشن)

در این قسمت می توانید از اطلاعاتی که در مورد تصاویر دارید، برای مثال اینکه تصاویر چه چیزهایی هستند، استفاده کنید. تصاویر منطبق شده را با نام های near03.jpg و far04.jpg به ترتیب برای تصویر نزدیک و تصویر دور ذخیره نمایید .

در تصویری که می خواهید از نزدیک دیده شود باید فرکانس های پایین را حذف کرده و فرکانس های بالا را نگه دارید ودر تصویری که می خواهید از دور دیده شود فرکانس های بالا را حذف نموده و فرکانس های پایین را نگه دارید.

نحوه گرفتن تبدیل فوریه از تصاویر

برای تر کیب دو تصویر، ابتدا هر دو تصویر را به دامنه فرکانس ببرید. سپس، در تصویر اول (تصویری که می خواهید از نزدیک دیده شود) فرکانس های پایین را حذف نموده و فرکانس های بالا را نگه دارید.

در تصویر دوم (تصویری که می خواهید از دور دیده شود) فرکانس های بالا را حذف نموده و فرکانس های پایین را نگه دارید.

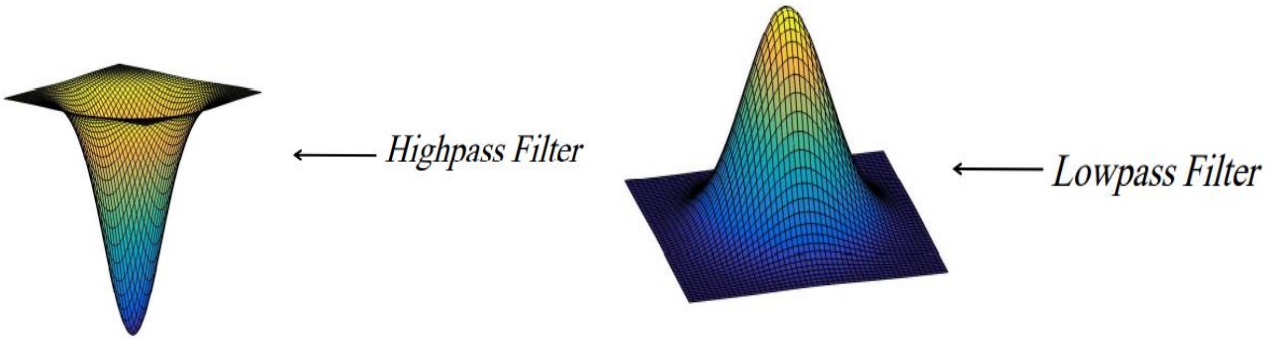
حال می توانید این دو تصویر در دامنه فرکانس را با هم ترکیب کنید، برای مثال با هم جمع کنید یا میانگین بگیرید. در نهایت، تصویر هیبریدی حاصل را به حوزه مکان برگردانید.

برای این منظور مراحل زیر را انجام دهید:

هر دو تصویر را به دامنه فرکانس ببرید. بزرگی ضرایب تبدیل فوریه را محاسبه کرده و لگاریتم آن ها را نمایش دهید .تصاویر حاصل را به ترتیب با نام های near_dft_05.jpg و far_dft_06.jpg ذخیره نمایید.

در تصویری که می خواهید از نزدیک دیده شود، فرکانس های بالا را حفظ نموده و فرکانس های پایین ر ا حذف کنید (فیلترینگ بالاگذر).

در تصویر دیگر فرکانس های پایین را حفظ نموده و فرکانس های بالا را حذف نمایید (فیلترینگ پایین گذر). برای فیلتر کردن میتوانید از فیلتر های مختلفی استفاده کنید. اما بهتر است از فیلتر گوس برای این منظور استفاده کنید. برای فیلتر پایین گذر می توانید از یک فیلتر گاوس دو بعدی هم اندازه تصویر با انحراف معیار s استفاده کنید. برای فیلتر بالاگذر هم میتوانید یک فیتلر دوبعدی پایین گذر گاوس با انحراف معیارr استفاده کرده و در نهایت آن را از فیلتر با مقدار ثابت یک کم کرده تا فیاتر بالاگذر بدست آید.(1-G)(مقادیر r و s مناسب برای هر فیلتر را باید خودتان با امتحان مقادیر مختلف بدست آورید)



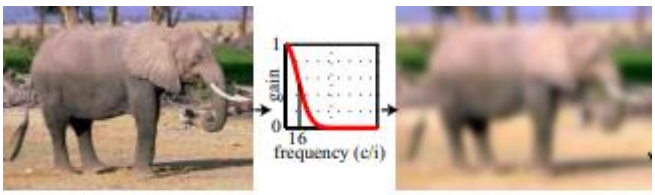
این دو فیلتر را نمایش داده و با نام های highpass07.jpg و lowpass08.jpg ذخیره نمایید .مقادیر r و s را در در گزارش خود ذکر کنید.

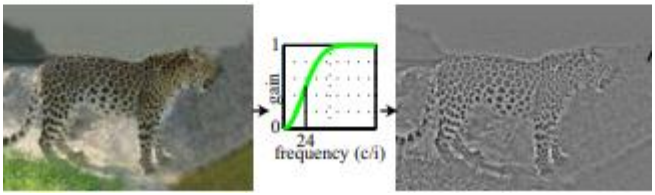
حال که دو فیلتر را بدست آوردین باید در دامنه فرکانس باید یک مقدار cutoff برای آن در نظر بگیرید.

در فیلتر پایین گذر باید ضرایبی که فاصله آنها تا مبدا کمتر از این مقدار cutoff می‌باشد را حفظ کنید و بقیه ضرایب را صفر کنید. در فیلتر بالاگذر باید عکس این کار را تکرار کنید .یعنی ضرایبی که تا مبدا بزرگتر از مقدار cutoff می باشد را حفظ کرده و بقیه ضرایب را صفر کنید.

می توانید دو مقدار cutoff را یکسان در نظر بگیرید .اما برای کسب نتایج بهتر ، بهتر است این دو مقدار را متفاوت در نظر بگیرید بطوری که در دامنه فرکانس این دو تصویر مقداری overlap داشته باشن.

بطوری که فرکانس cutoff در فیلتر بالاگذر کوچک تر از این مقدار در فیلتر پایین گذر باشد.





در اینصورت هنگام ترکیب دو تصویر در ناحیه مشترک باید از میانگین گیری وزن دار استفاده کنید.

مقدار cutoff ها را در فیلترهای متناظرشان اعمال کنید. فیلترهای به دست آمده در مرحله قبل را در تصاویر متناظرشان اعمال نموده و نتایج را با نام هاhighpassed09.jpg و lowpassed10.jpg ذخیره نمایید.

این دو تصویر را با میانگین گیری وزن دار ترکیب نموده و نتیجه حاصل را با نام frequency_hybrid11.jpgذخیره نمایید.

تصویر حاصل را به حوزه مکان برده و با نام final_near.jpg ذخیره نمایید.

یک نسخه کوچکتر این تصویر را نیز ذخیره نمایید به طوری که در آن تصویری که باید از دور دیده شود را بتوان دید. این تصویر را با نام final_far.jpg ذخیره نمایید.

- در صورت علاقه و گرفتن نتایج بهتر میتوانید این مقاله را مطالعه کنید.
- فایل کد را با نام q4.py ذخیره کنید.
- از تابعی مانند زیر می توانید برای درست کردن فیلتر گوسی متناسب با سایز تصویرتان استفاده کنید:

```
def GaussianFilter(nRows, nCols, sigma, highPass=True):
    if(nRows % 2 == 0):
       centerI =int(nRows/2)
    else:
       centerI =int(nRows/2)+1
    if(nCols % 2 == 0):
       centerJ =int(nCols/2)
    else:
       centerJ =int(nCols/2)+1
    filter_gauss=np.zeros((nRows,nCols))
    for j in range(nCols) :
     for i in range(nRows):
       g = math.exp(-1.0 * ((i - centerI)**2 + (j - centerJ)**2) / (2 * sigma**2))
       if(highPass):
           filter_gauss[i,j]=1-g
       else:
           filter_gauss[i,j]=g

    return filter_gauss
```