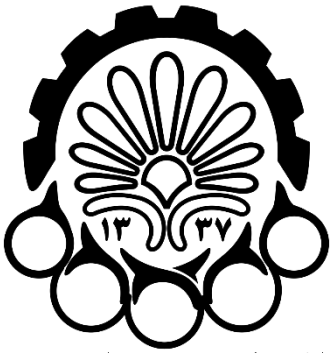
به نام خدا




دانشگاه صنعتی امیرکبیر
( پلی تکنیک تهران )

# تمرین درس پردازش تصویر–سری چهارم

فردین آیار

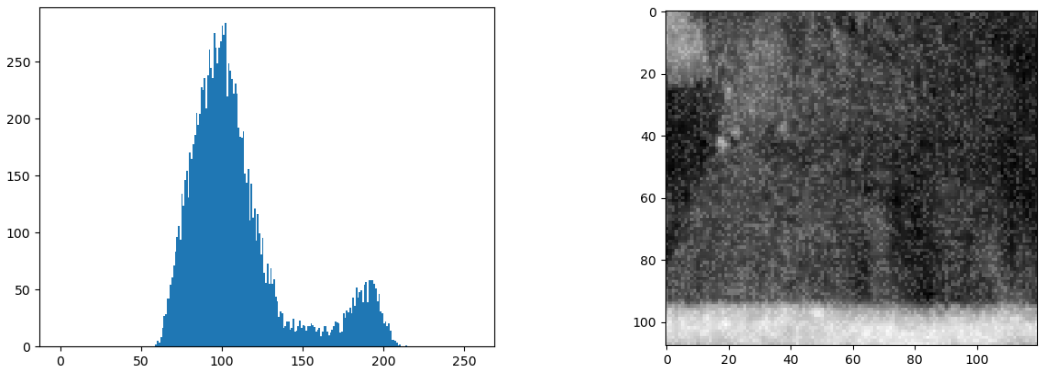
شماره دانشجویی: ۹۹۱۳۱۰۴۰

استاد: دکتر رحمتی

دانشکده کامپیوتر– زمستان ۹۹

# ۱)

a) کد مربوط به این قسمت در فایل a.py قراردارد. یک بخش نسبتا یکنواخت[۱] از تصویر و هیستوگرام آن در شکل ۱-۱ نشان داده شده‌است. با توجه به این شکل، تصویر دارای نویز گاووسی می‌باشد که برای رفع آن از فیلتر میانگین هندسی با اندازه ۳ استفاده می‌کنیم. نتیجه که نسبت به تصویر اصلی بهبود یافته در شکل ۱-۲ ارائه شده‌است.

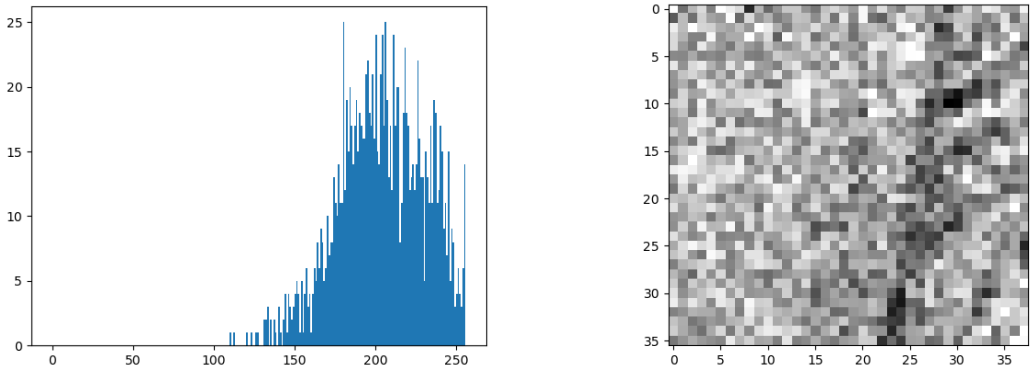



*شکل ۱-۱*

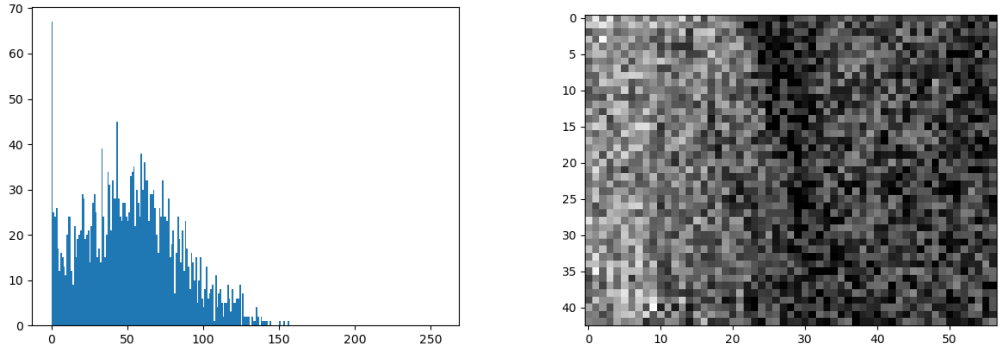

*شکل ۱-۲*

b) کد مربوط به این قسمت در شکل b.py قرار دارد. برای دو بخش یکنواخت روشن و تیره از تصویر، هیستوگرام تصویر در شکل‌های ۱-۳ و ۱-۴ نمایش داده شده‌است. با توجه به این شکل‌ها به نظر می‌رسد بخش‌های روشن دارای نویز نمک و گاووسی و بخش‌های تیره دارای نویز فلفل و گاووسی می‌باشد. برای بهبود تصویر، ابتدا با یک فیلتر میانه، نقاط بسیار روشن و یا بسیار تاریک را با مقدار میانه جایگزین می‌کنیم، سپس روی کل تصویر از فیلتر میانگین هندسی استفاده می‌کنیم. نتیجه در تصویر ۱-۵ ارائه شده‌است.

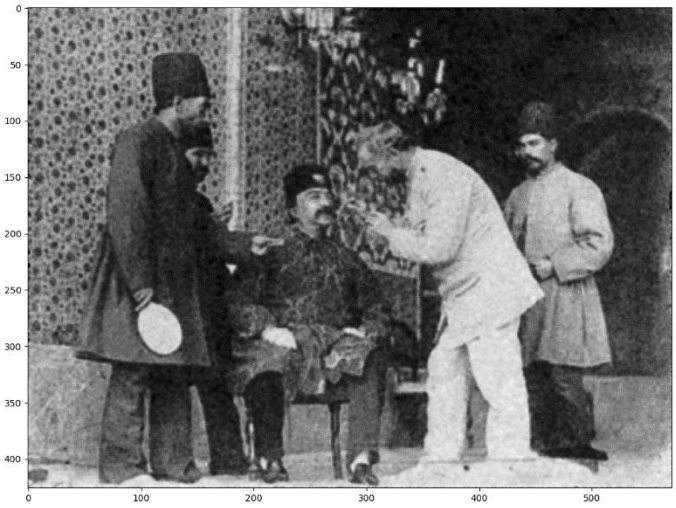
[۱] این بخش هنگام اجرای برنامه از کاربر دریافت می‌شود.



شکل ۱–۳

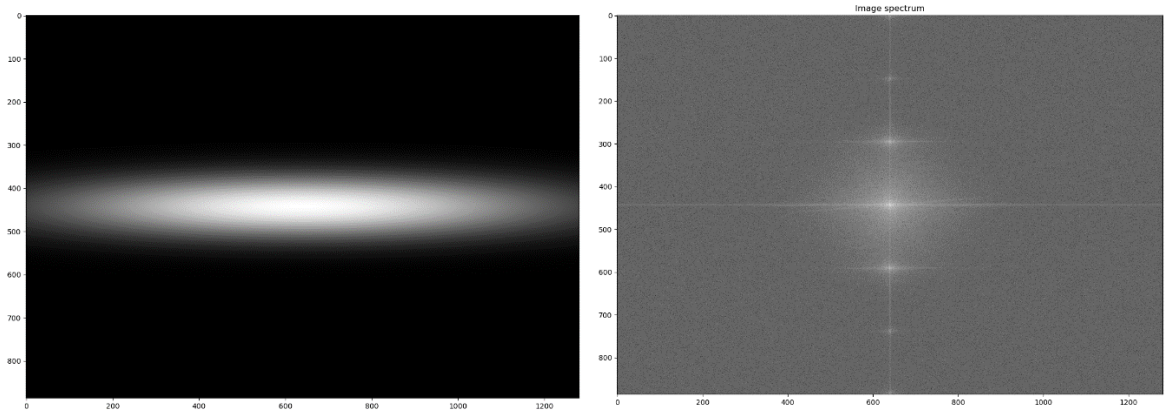


شکل ۱–۴



شکل ۱–۵

c) کد مربوط به این قسمت در فایل c.py قرار دارد. نویز این تصویر به وضوح یک نویز متناوب است بنابراین باید در دامین فرکانس برطرف شود. در شکل ۱–۶ تبدیل فوریه تصویر و فیلتر استفاده شده برای این تصویر و در شکل ۱–۷ تصویر بهبود یافته ارائه شده‌است.




*شکل ۱–۶*


*شکل ۱–۷*

d) کد مربوط به این قسمت در فایل d.py قرار دارد. به نظر می‌رسد تصویر دارای نویز نمک و فلفل است؛ بنابراین از فیلتر میانه با اندازه ۵ برای بهبود آن استفاده می‌کنیم. نتیجه در شکل ۱–۸ ارائه شده‌است. همانطور که مشاهده می‌شود، اگرچه نویز برطرف شده اما کیفیت تصویر نیز اندکی کاهش یافته است.


*شکل ۱-۸*

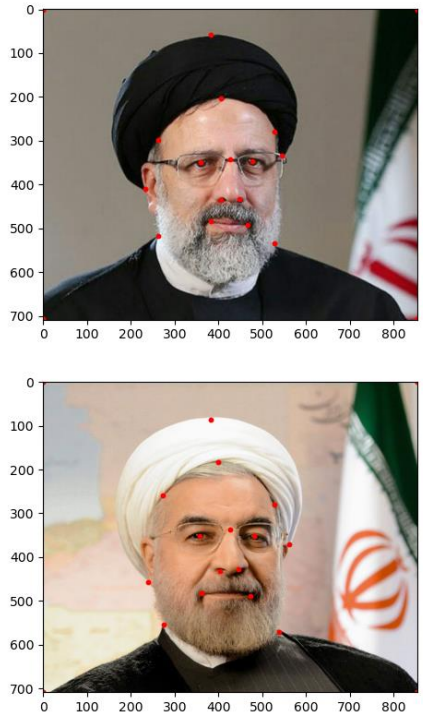
## ۲)

a) کد مربوط به این قسمت در فایل a.py قرار دارد. مقادیر آلفا با فاصله یکسان از بازه صفر تا یک انتخاب می‌شود. خروجی به ازای مقادیر مختلف در شکل ۱-۲ و خروجی به ازای آلفا برابر با ۰.۵ در شکل ۲-۲ ارائه شده‌است. با توجه به شکل ۲-۲، تصویر ترکیبی در بخش‌های مختلف دارای افکت ghosting است؛ به ویژه در بخش‌های چشم سمت راست و لباس.

*شکل ۲-۱*

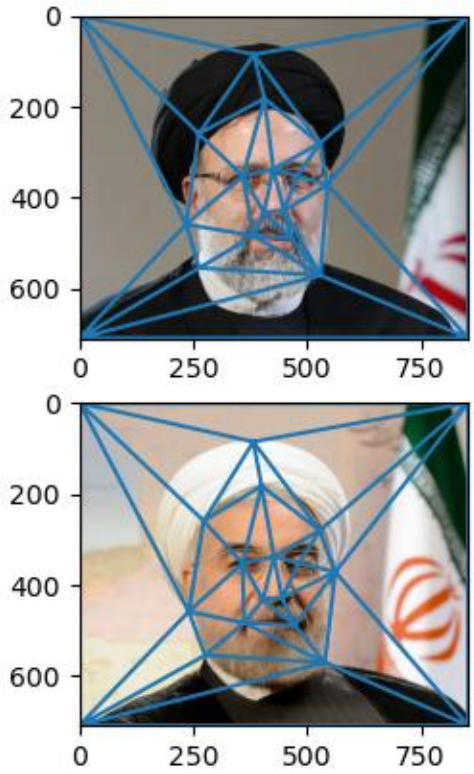
شکل ۲-۲

b) کد مربوط به این بخش در فایل b.py قرار دارد. نقاط برای هر تصویر جداگانه و باترتیب یکسان انتخاب می‌شوند. در شکل ۲-۳ نقاط انتخاب شده مشخص شده‌اند.(۱۵ نقطه برای هر تصویر)



شکل ۲-۳

c) کد مربوط به این قسمت در فایل c.py قرار دارد. خروجی در شکل ۲-۴ ارائه شده است.




شکل ۲-۴

d) کد مربوط به این قسمت در فایل d.py قرار دارد. خروجی به ازای مقادیر مختلف در شکل ۲-۵ و خروجی به ازای آلفا برابر با ۰.۵ در شکل ۲-۶ ارائه شده‌است. با توجه به شکل ۲-۶، افکت ghosting در این روش به میزان قابل توجه کاهش یافته‌است؛ اگرچه در نواحی مربوط به لباس این افکت همچنان وجود دارد که با افزایش تعداد نقاط کلیدی، می‌توان آن را برطرف کرد.

شکل ۲-۵

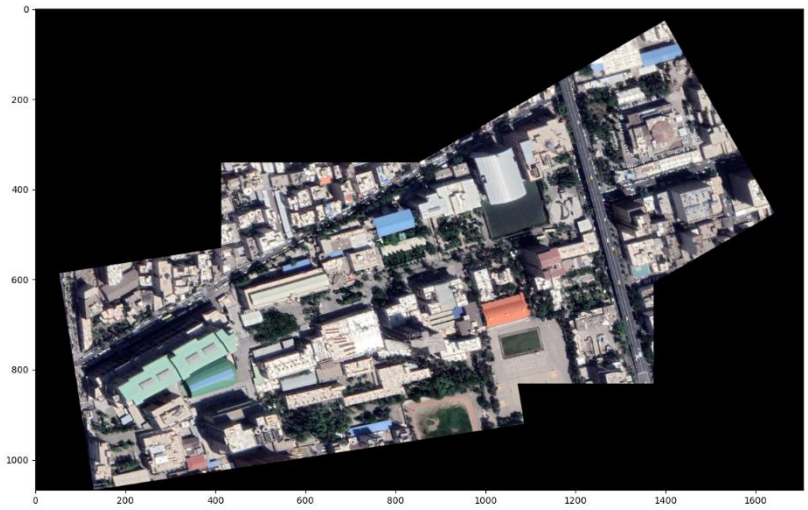

*شکل ۲-۶*

e) کد مربوط به این قسمت در فایل e.py قرار دارد. فایل ویدیویی در پوشه مربوط به این سوال با نام video.avi ذخیره شده‌است.

# ۳)

برای انجام عمل Blending، هیچگونه عمل میانگین‌گیری استفاده نشده و صرفا تصویر اول روی تصویر دوم قرار داده می‌شود. برای ترکیب تصاویر از یک روش حریصانه استفاده شده‌است. فرض کنیم همه تصاویر از ابتدا در آرایه image_list قرار داشته باشند. در هرمرحله دو تصویری که کمترین فاصله را از هم داشته باشند از image_list حذف می‌شوند و ترکیب آن‌ها به image_list اضافه می‌شود. این کار تا زمانی که فقط یک تصویر باقی مانده باشد ادامه می‌یابد.

a) کد مربوط به این قسمت در فایل a.py قرار دارد. خروجی در شکل ۳-۱ ارائه شده‌است.

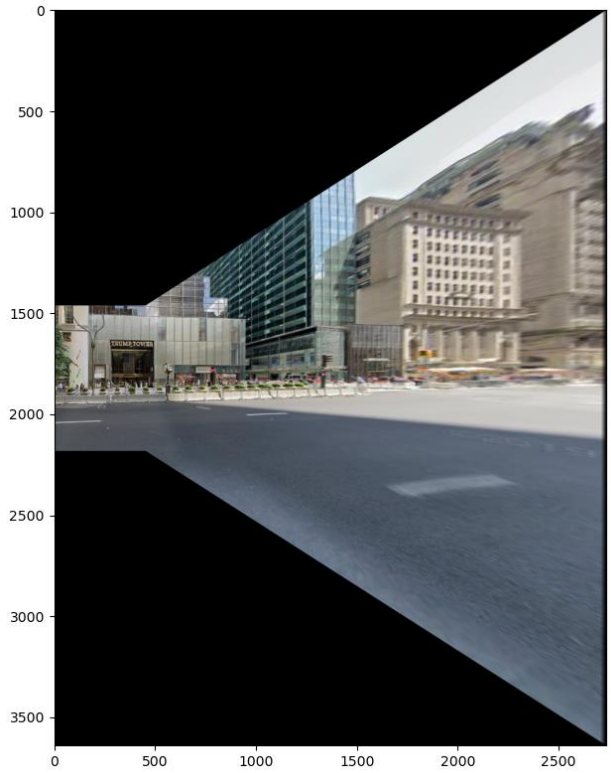


*شکل ۳-۱*

b) کد مربوط به این قسمت در فایل b.py قرار دارد. بهترین خروجی با انتخاب پارامترهای مختلف در شکل ۳-۲ ارائه شده‌است. همانطور که مشاهده می‌شود بعضی از تصاویر به درستی به هم متصل نشده‌اند.

*شکل ۳-۲*

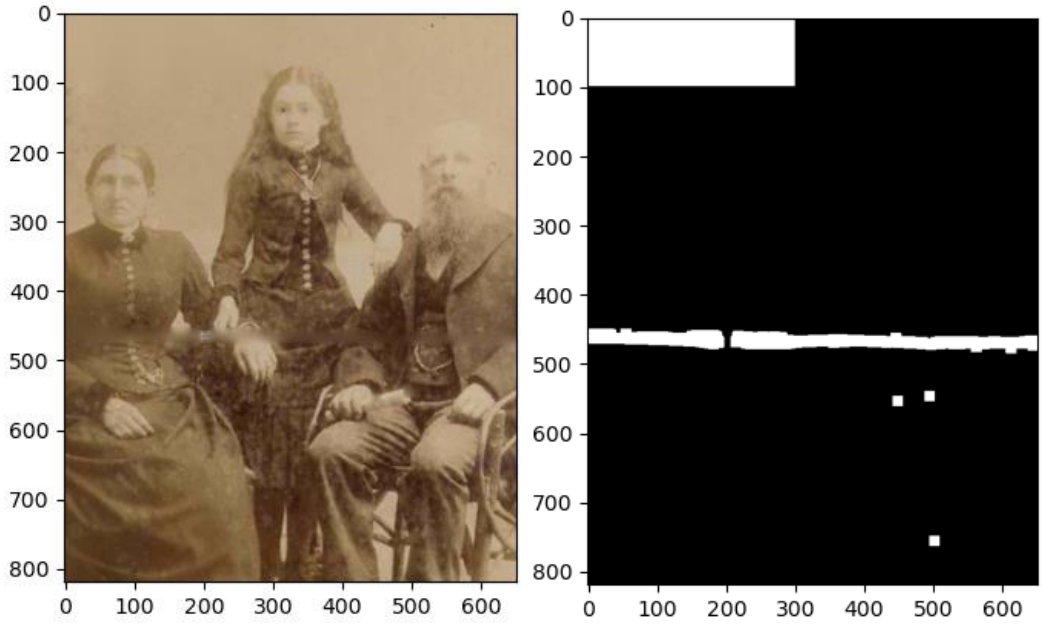
c) کد مربوط به این بخش در فایل c.py قرار دارد. اگر از همه عکس‌های داده شده برای این قسمت استفاده کنیم، حجم تصاویر به سرعت افزایش می‌یابد و قبل از تولید خروجی نهایی، نرم‌افزار با خطای حافظه مواجه می‌شود؛ بنابراین تنها از ۶ تصویر اول استفاده می‌کنیم. خروجی در شکل ۳-۳ ارائه شده‌است. همانطورکه مشاهده می‌شود خروجی نهایی از همه ۶ تصویر ایجاد نشده و این نشان دهنده نامناسب بودن برنامه برای این بخش می‌باشد. به نظر می‌رسد دلیل این مشکل، که افزایش حجم تصاویر نیز به همین علت است، این است که برای ساخت تصاویر پاناروما، باید از تبدیل استوانه ای استفاده کرد؛ حال آن‌که برنامه نوشته شده از تبدیل Homography استفاده می‌کند.




*شکل ۴-۴*

**۴)**

a) کد مربوط به این قسمت در فایل a.py قرار دارد. محدوده نوشته‌های روی عکس به صورت دستی و چین خوردگی میان تصویر با ترشهودینگ انتخاب و یک ماسک از روی آن ساخته می‌شود. با استفاده از دستور cv2.inpaint تا حدی آن‌ها را برطرف می‌کنیم. ماسک مربوطه و خروجی در شکل ۴-۱ ارائه شده‌است.

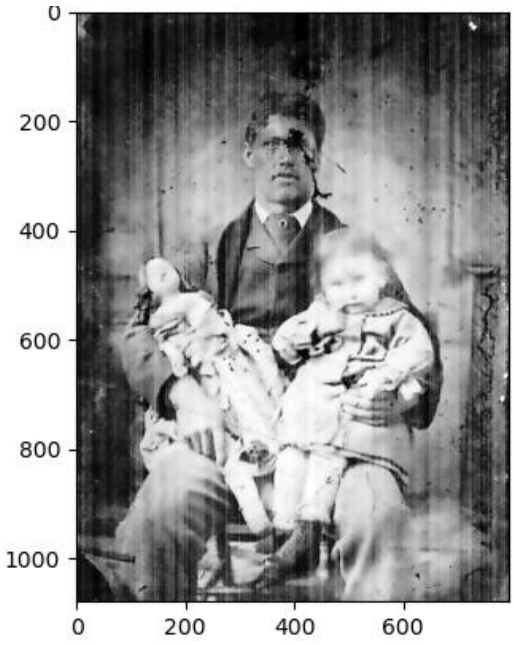



شکل ۵-۱

در ادامه با استفاده از از روش Weighted guided image filtering سعی می‌کنیم تا حدی لکه‌های موجود در تصویر را از بین ببریم. خروجی نهایی در شکل ۴-۲ ارائه شده‌است.

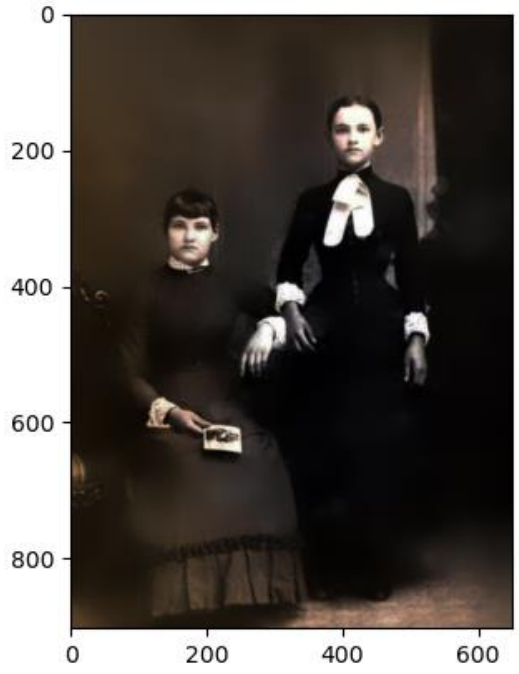

شکل ۴-۲

b) کد مربوط به این قسمت در فایل b.py قرار دارد. در نگاه اول به نظر می‌رسد این تصویر دارای نویز متناوب است؛ اما با استفاده از فیلترینگ در دامین فرکانس، تصویر بهبود چندانی نمی‌یابد پس از این عمل صرفنظر می‌کنیم. ابتدا از یکنواخت سازی هیستوگرام و سپس فیلتر wgif(مشابه قسمت a) استفاده می‌کنیم. خروجی در تصویر ۴-۳ ارائه شده‌است.



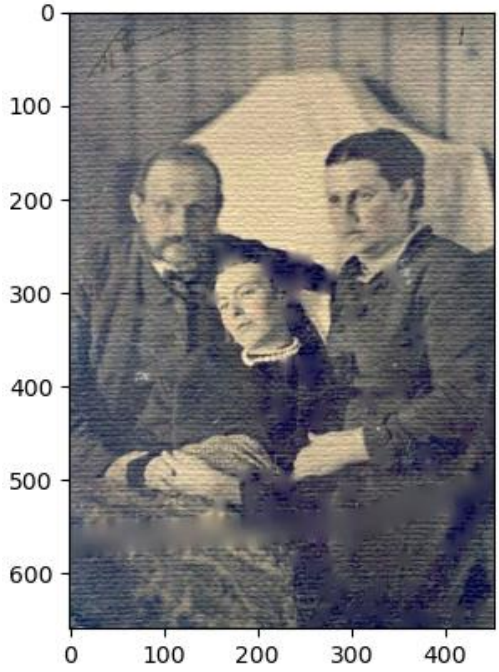

شکل ۴-۳

c) کد مربوط به این قسمت در فایل c.py قرار دارد. ابتدا ناحیه سفید تصویر را با تابع cv2.inpaint برطرف می‌کنیم و سپس از فیلتر wgif برای محو کردن لکه‌ها استفاده می‌کنیم. خروجی در شکل ۴-۴ ارائه شده‌است.

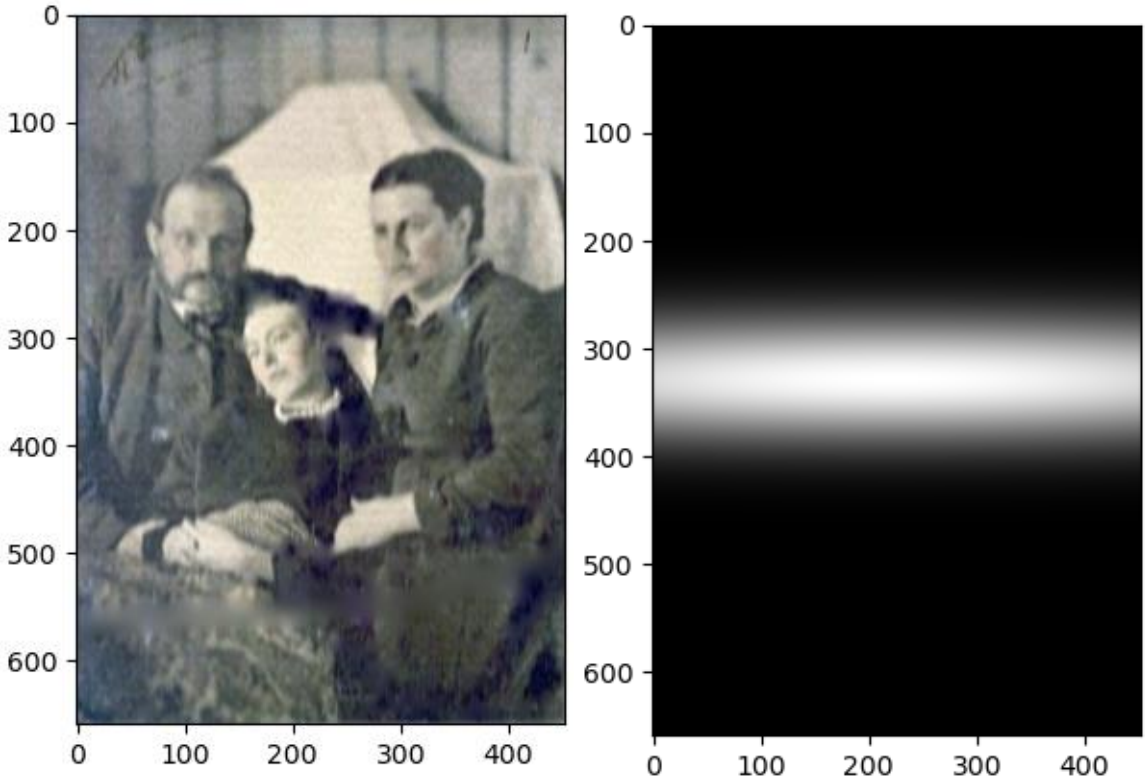



شکل ۴-۴

d) کد مربوط به این قسمت در فایل d.py قرار دارد. ابتدا ناحیه سفید تصویر را با تابع cv2.inpaint برطرف می‌کنیم(شکل ۴-۵). سپس برای حذف خطوط افقی موجود در تصویر، از فیلترینگ در دامین فرکانس استفاده می‌کنیم.(شکل ۴-۶)


0
100
200
300
400
500
600
0
100
200
300
400

شکل ۴-۵


0
100
200
300
400
500
600
0
100
200
300
400
0
100
200
300
400
500
600
0
100
200
300
400

شکل ۴-۶

## ۵)

a) نویزهای جمعی[۲] را می‌توان به صورت جمع تصویر اصلی و نویز مدل کرد؛ بنابراین مقدار این نویزها مستقل از سطح روشنایی پیکسل‌ها است. در طرف مقابل نویزهای ضربی[۳] به صورت ضرب نویز در تصویر اصلی مدل می‌شود؛ بنابراین در این گونه نویزها مقدار نویز در هر پیکسل به سطح روشنایی آن پیکسل وابسته است. با توجه به این تعریف، نویز نمک–فلفل یک نویز جمعی است، زیرا مقدار آن مستقل از سطح روشنایی تصویر است.

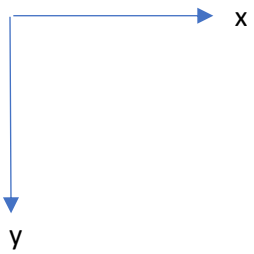
b) بدین منظور می‌توان از فیلتر گاووسی استفاده کرد. زیرا تبدیل فوریه آن نیز یک تابع گاووسی می‌باشد و بنابراین مشکل ringing به وجود نمی‌آید. هرچه فرکانس نویز در تصویر کوچکتر باشد، باید از فیلتر گاووسی با سایز بزرگتری استفاده کرد.

c) با توجه به توضیحات بخش c، می‌توان از فیلتر گاووسی در دامین فرکانس بدین منظور استفاده کرد. هرچه واریانس نویز بزرگتر باشد، واریانس فیلتر گاووسی در دامنه فرکانس باید کوچکتر باشد.

d) از آنجا که بعد از انتقال تصویر به فضای جدید، برای تعیین مقادیر پیکسل‌ها از عمل درونیابی استفاده می‌شود، حتی در صورت استفاده از توابع یک به یک این عمل معکوس پذیر نیست.

e) تغییرات مورد نظر از ضرب سه ماتریس انتقال زیر بدست می‌آید.

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \left( \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(-45) & -\sin(-45) & 0 \\ \sin(-45) & \cos(-45) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 30 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$




---

[۲] additive

[۳] multiplicative