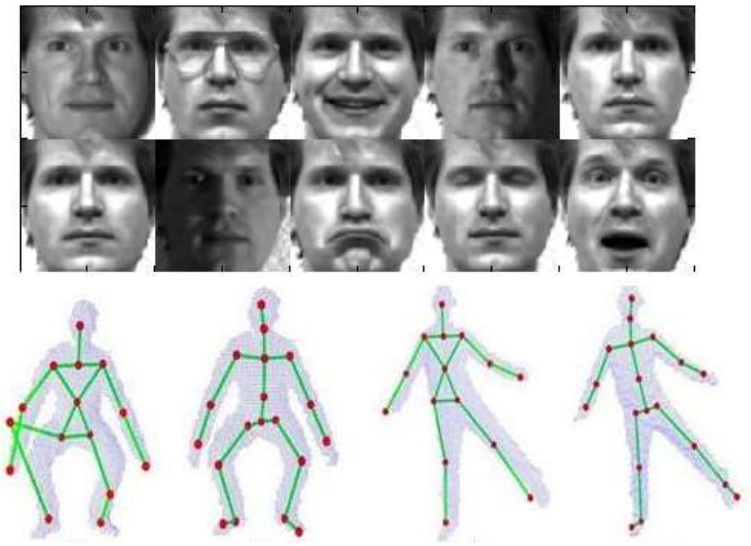
بسمه تعالی

دانشگاه صنعتی شریف

دانشکده مهندسی برق

گروه سیستم‌های دیجیتال



آزمایشگاه یادگیری و بینایی ماشین

دستور کار آزمایش پنجم: نرم‌سازی، کانولوشن، گرادیان و لبه یابی

زمان لازم برای انجام آزمایش: حداکثر یک جلسه

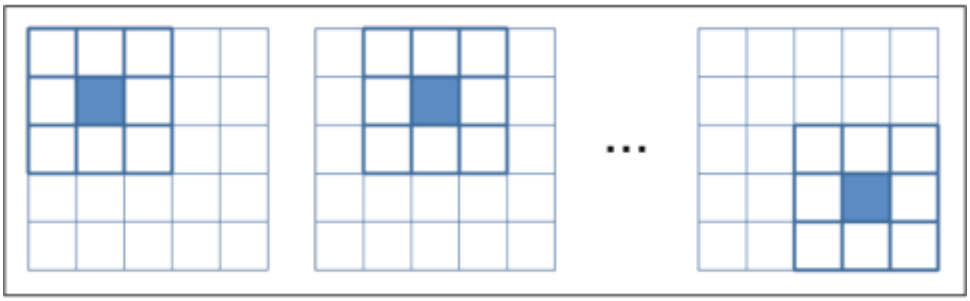
آزمایش پنجم: نرم‌سازی، کانولوشن، گرادیان و لبه یابی تصویر

از این آزمایش به بعد، با کتابخانه OpenCV[1] که یک کتابخانه قدرتمند برای پردازش تصویر و ویدئو است استفاده خواهیم کرد. این کتابخانه در زبان‌های گوناگون از جمله C و پایتون ایجاد شده است که ما از زبان پایتون استفاده می‌کنیم. در این آزمایش، ما به بررسی مفاهیم و آزمایش نرم‌سازی، کانولوشن، گرادیان و لبه‌یابی تصویر می‌پردازیم.

تصاویر سه‌کاناله (RGB) داده شده را بخوانید و یک‌کاناله (gray-scale) کرده و تصاویر اصلی و یک‌کاناله را نمایش دهید. سپس، تصاویر یک‌کاناله را ذخیره نمایید.

در پردازش تصویر، فیلتر کردن به این معنی است که کاری با تصویر کنیم تا ویژگی‌های خاصی پیدا کند یا برخی از خصوصیات خود را از دست دهد تا یک سری خصوصیات دیگر آن بیشتر جلوه کند. برای فیلتر کردن، چندین روش وجود دارد که برخی از آنها در حوزه فرکانس و برخی از آنها در حوزه مکانی (پیکسلی) انجام می‌شود. در اینجا ما با برخی فیلترهای مهم حوزه مکانی آشنا می‌شویم.

فیلترهای مکانی دارای یک پنجره معمولا متقارن (مثلا 9×9) هستند که هر یک از خانه‌های آن یک ضریب (وزن) است. به این پنجره، کرنل فیلتر گفته می‌شود. این پنجره روی تصویر می‌لغزد و هر یک از خانه‌هایش که روی یک پیکسل قرار می‌گیرد، در مقدار آن پیکسل ضرب می‌شود. سپس این حاصلضرب‌ها (مثلا ۹ حاصلضرب) با هم جمع می‌شوند و به جای مقدار یکی از این ۹ پیکسل در تصویر فیلتر شده قرار می‌گیرند. این پیکسل که مقدار می‌گیرد، پیکسل anchor نامیده می‌شود و معمولا پیکسل وسط قرار داده می‌شود. لغزش کرنل بر تصویر را در شکل زیر مشاهده می‌کنید:



[1] https://en.wikipedia.org/wiki/OpenCV

این لغزش کرنل بر تصویر و عملیات ضرب و جمع، همان مفهوم کانولوشن کرنل در تصویر است. اگر خروجی فیلتر را O، کرنل را K و تصویر را I فرض کنیم و اندازه کرنل (2r+1)×(2r+1) باشد و پیکسل anchor پیکسل وسط باشد، داریم:

$$O(i,j) = \sum_{m=-r}^{+r} \sum_{n=-r}^{+r} I(i+m, j+n) \cdot K(m,n)$$

اکنون، به فیلتر نرم‌ساز[۲] می‌پردازیم. کرنل این فیلتر به روش های مختلف مانند محاسبه ی میانگین، محاسبه ی میانه و ... عمل می کند و لذا لبه‌های تصویر را که دارای فرکانس‌های بالا هستند، حذف یا کمرنگ می‌کند. کرنل فیلتر می تواند به صورت میانگین، میانه و ... در نظر گرفته شود. یک نوع پرکاربرد از فیلتر نرم ساز، فیلتر نرم ساز گوسی نام دارد. این فیلتر دارای توزیع گوسی دوبعدی است و به عنوان مثال اگر 3×3 باشد، کرنلی به صورت زیر خواهد داشت:

$$\begin{bmatrix} 0.36 & 0.6 & 0.36 \\ 0.6 & 1 & 0.6 \\ 0.36 & 0.6 & 0.36 \end{bmatrix} \times \frac{1}{0.65}$$

روی تصاویری که ذخیره کرده اید، فیلترهای نرم ساز مختلف مانند میانگین‌گیر، میانه‌گیر و فیلتر گاوسی را با اندازه های مختلف ۳، ۵ و ۷ اعمال کنید و نتایج فیلتر را با هم مقایسه کنید. چه نتیجه‌ای می‌گیرید؟ آن را تحلیل کنید.

نوع دیگری از فیلترها که به آنها می پردازیم، فیلترهای لبه یاب هستند. گرادیان یک تصویر همان مشتق جهتی تصویر در راستای x و y است که به صورت زیر تعریف می شود:

$$\nabla f(i,j) = \begin{bmatrix} f_x \\ f_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f(i,j)}{\partial x} \\ \\ \frac{\partial f(i,j)}{\partial y} \end{bmatrix}$$

[2] https://docs.opencv.org/3.0.0/d4/d13/tutorial_py_filtering.html

که در آن، f به معنی روشنایی پیکسل است. گرادیان به این معنی است که هر پیکسل با پیکسل بعدی‌اش (در هر یک از راستاهای x و y) چقدر تفاوت روشنایی دارد. از این گرادیان، برای لبه‌یابی استفاده می‌شود زیرا رفتار فیلتر بالاگذر دارد. روش‌های گوناگونی از جمله Sobel، Canny و ... برای یافتن گرادیان وجود دارد.

فیلتر Sobel[3] به صورت زیر است (مولفه‌های گرادیان):

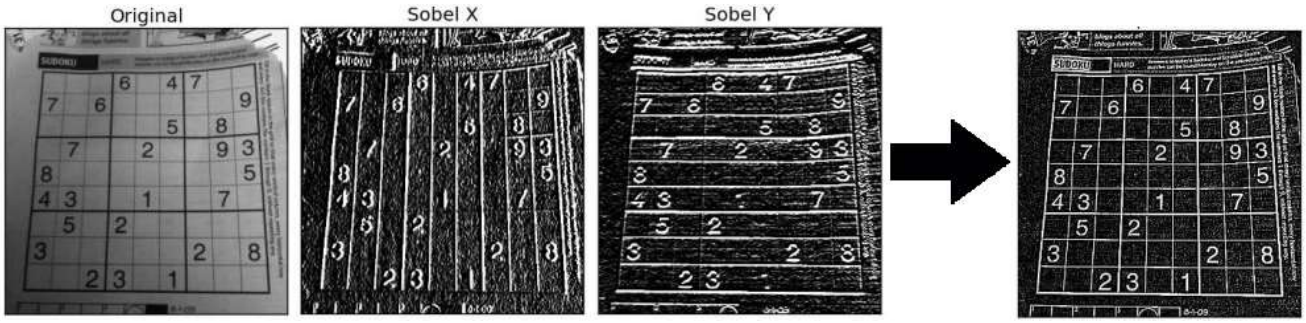
$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I = f_x * I$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I = f_y * I$$

و اندازه خروجی فیلتر Sobel به صورت زیر محاسبه می‌شود:

$$G = \sqrt{G_x^2 + G_y^2}$$

دو فیلتر fx و fy به ترتیب بیانگر یافتن لبه‌های افقی و عمودی تصویر است. نمونه‌ای از این عملکرد را در زیر مشاهده می‌کنید.



نوع دیگری از فیلتر لبه‌یاب، فیلتر قدرتمند Canny[4] است. این فیلتر در ابتدا روی تصویر برای حذف نویز فیلتر گوسی اعمال می‌کند. سپس، از تصویر گرادیان می‌گیرد. در این روش هم به دامنه و هم به زاویه گرادیان دقت می‌شود:

[3] https://docs.opencv.org/3.0.0/d5/d0f/tutorial_py_gradients.html
[4] https://docs.opencv.org/3.4.1/da/d22/tutorial_py_canny.html

$$\nabla f = \begin{bmatrix} f_x \\ f_y \end{bmatrix}$$

$$|\nabla f| = \sqrt{f_x^2 + f_y^2}$$

$$< \nabla F = \tan^{-1} \frac{f_y}{f_x}$$

اگر مقدار دامنه گرادیان در هر پیکسل، از دامنه گرادیان در دو پیکسل اطرافش در راستای زاویه گرادیان، بیشتر باشد، نامزد لبه بودن می‌شود وگرنه رد می‌شود. این کار برای این است که لبه کلفت نداشته باشیم و فقط لبه‌های تیز یافت شود.

سپس، در این روش، دو مقدار آستانه بالا و پایین تعریف می‌شود. نامزدهای لبه‌ای که مقدار دامنه‌ی گرادیان آنها بالاتر از آستانه بالا و یا پایین‌تر از آستانه پایین است، به ترتیب به عنوان لبه قبول و رد می‌شوند. لبه‌های نامزدی که در بین این دو آستانه هستند، فقط به شرطی به عنوان لبه انتخاب می‌شوند که یا در مجاورت حداقل یک پیکسل لبه باشند یا در مجاورت حداقل یک پیکسل نامزد دیگر باشند. این مرحله باعث می‌شود که لبه‌های تکی و کوچک (نقطه‌ای) که مثل نویز هستند، حذف شوند.

همان تصاویر ذخیره شده را با فیلترهای Sobel و Prewitt فیلتر کنید و نتایج را بررسی کنید. همچنین تصاویر ذخیره شده را با کمک فیلتر Canny فیلتر کنید و لبه‌های آن را نمایش دهید. پارامترهای فیلتر Canny را تغییر دهید، نتیجه را بررسی و با نتایج فیلترهای Sobel و Prewitt مقایسه کنید.

نوع دیگری از فیلترها که زیاد مورد استفاده قرار میگیرند تبدیل های مورفولوژی[5] هستند. این فیلترها روی تصاویر باینری اعمال می شوند و می توانند برای پرکردن حفره ها و حذف نقاط اضافی در تصویر بکار روند.

مهم ترین تبدیل های مورفولوژی عبارتند از:

[5] https://docs.opencv.org/3.0.0/d9/d61/tutorial_py_morphological_ops.html

- Erosion: این عمل مانند ساییدن یک تصویر باینری سفید با پس زمینه سیاه است. در این حالت بعد از اعمال فیلتر (کرنل) بر روی هر ناحیه، پیکسل مربوط به آن، زمانی ۱ می شود که تمام پیکسل های درون فیلتر ۱ باشد.
- Dilation: در این حالت پیکسل مربوطه زمانی ۱ می شود که حداقل یک ۱ درون فیلتر وجود داشته باشد.
- Opening: عمل erosion و سپس dilation . با این کار نویزهای سفید از بین می روند.
- Closing: عمل dilation سپس erosion. با این کار حفره های سیاه از بین می روند.

نوع دیگری از فیلترها که زیاد مورد استفاده قرار میگیرند تبدیل های مورفولوژی[6] هستند. این فیلترها روی تصاویر باینری اعمال می شوند و می توانند برای پرکردن حفره ها و حذف نقاط اضافی در تصویر بکار روند.

از راست به چپ: تصویر ورودی، خروجی dilation، خروجی erosion

سمت راست: opening، سمت چپ: closing

[6] https://docs.opencv.org/3.0.0/d9/d61/tutorial_py_morphological_ops.html

یک کاربرد فیلترهای مورفولوژی در حذف پس زمینه[7] از ویدئو می باشد. برای بررسی این مورد، دوربین وبکم را در محلی ثابت قرار دهید و یک ویدئوی چند ثانیه ای از خودتان تهیه نمایید، به نحوی که در طول این مدت کمی جابجا شوید. همچنین بدون تغییر محل دوربین و اشیا پس زمینه، تصویری بدون حضور خودتان از پس زمینه بگیرید. ویدئو را فریم به فریم بخوانید[8] و در هر فریم مراحلی که در ادامه می آید را انجام دهید. ابتدا تصویر پس زمینه را از تصویر فریم کم نموده و به Grayscale تبدیل کنید. سپس تصویر حاصل را با Threshold گذاری روی مقادیر باینری کنید[9]. در ادامه تصویر باینری حاصل را با تبدیل های مورفولوژی erode و dilate اصلاح نمایید تا یک ماسک باینری از محل بدن خودتان در فریم مربوطه ساخته شود. نحوه ی مناسب بکارگیری این تبدیل ها و تنظیم پارامترهای آنها را با سعی و خطا بیابید. در انتها، تصویر فریم را با ماسک حاصل AND کنید تا تنها تصویر خودتان باقی بماند. از کنار هم قرار دادن این فریم ها ویدئویی بدست می آید که پس زمینه ی آن حذف شده است.

تذکر: در صورتی که کار با ویدئو برای شما دشوار هست، می توانید ابتدا پروسه ی حذف پس زمینه را بر روی یک تصویر ساده انجام دهید.

[7] https://docs.opencv.org/3.2.0/d1/dc5/tutorial_background_subtraction.html
[8] https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_gui/py_video_display/py_video_display.html
[9] https://docs.opencv.org/3.0.0/d7/d4d/tutorial_py_thresholding.html