فیلتر میانگین

منا رداد

اطلاعات گزارش

تاریخ:
1400/9/1

واژگان کلیدی:
Box filter
robert
کانولوشن
فیلتر میانگین

چکیده

در این بخش از تمارین فیلتر میانگین را به کمک عملیات کانولوشن پیاده سازی می کنیم. یکی از خصوصیات فیلتر میانگین box filter می باشد که در این تمرین قصد داریم روشی را جهت افزایش سرعت این فیلتر پیاده سازی کنیم و با از ویژگی بیشتر آشنا بشویم.در قسمت دوم با آشکار سازی لبه رابرت آشنا می شویم و با اعمال آن بر تصویر mosque شدت لبه ها را تعیین می نماییم. جهت انجام این عملیات فیلتر میانگین را در اندازه های متفاوت اعمال می کنیم. در انتها دو نوع فیلتر 5*5 و 7*7 جدیدی را نیز طراحی می کنیم.

1-مقدمه

فیلتر میانگین دارای پنجره کشویی است و می توان آن را از نوع فیلتر فضایی (space) دانست که کاری که این فیلتر انجام می دهد این است که مقدار مرکزی در پنجره را با میانگین تمام پیکسل های موجود در پنجره جایگزین میکند. عموما شکل پنجره به صورت مربعی است اما این امر ثابت نیست و میتوان پنجره با اشکال دیگر نیز داشته باشیم.box filter گونه ای از فیلتر میانگین است که در آن تمام همسایه ها دارای وزن 1 می باشند. با افزایش سایز فیلتر میتوان تصویر را هموارتر کرد اما باید به این امر توجه داشت که با افزایش سایز زمان پردازش نیز افزایش می یابد. به منظور حل این مشکل به بررسی راه حل جهت افزایش سرعت می پردازیم و سپس فیلتر میانگین در سایز های مختلف را به تصویر mosque اعمال کرده و به مقایسه هریک می پردازیم.

1-2- راه حل جهت افزایش سرعت

در آغاز با استفاده از ویژگی فیلتر جعبه ای، فیلتر میانگین را پیاده سازی می کنیم برای انجام این کار اندازه های داده شده در صورت تمرین را برای فیلتر میانگین خود قرار می دهیم حال در ادامه به صورت ضرب درایه در درایه اولین پیکسل موجود در فیلتر را در پیکسل های تصویر ضرب می کنیم سپس به جمع مقادیر بدست آمده می پردازیم و آن را تحت عنوان مقدار جدید پیکسل اول در تصویر نهایی قرار می دهیم همین روند را برای پیکسل دوم و به ترتیب برای سایر پیکسل ها انجام می دهیم و به این شکل تمامی پیکسل ها مقادیر جدید خود را دریافت می کنند.حال برای افزایش سرعت ایده این است که به ازای هربار اعمال فیلتر n*n در وهله اول پیکسل ها را به صورت درایه ای در فیلتر ها ضرب می کنیم سپس n-1 ستون آخر(سمت راست) را

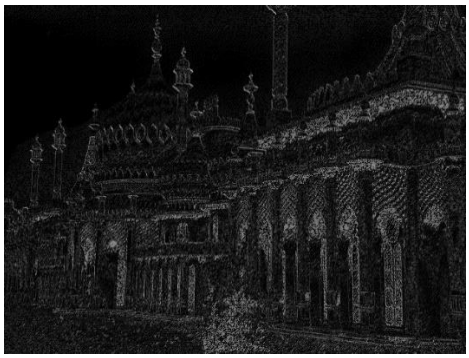
که n-1 ستون ابتدایی پنجره در مرحله بعد می باشد را در یک ماتریس کمکی save می کنیم و از این مرحله به بعد تنها ستون n ام را در ستون آخر ضرب می کنیم. که این کار باعث افزایش سرعت و کارایی الگوریتم ما میشود چرا که ضرب های انجام شده کاهش می یابد و زمان اجرای الگوریتم به نسبت الگوریتم اولیه ..... مقدار کاهش می یابد.

2-2- آشکار سازی لبه Robert

ما برای تشخیص لبه های یک شی از بین اشیا دیگر از آشکار سازی استفاده می کنیم.که تغییرات به صورت تغییر رنگ و تغییر شدت روشنایی(تغییرات فیزیکی) بر روی لبه ها صورت می گیرد.

حال اگر مقادیر موجود در محیط به صورت پیوسته باشد تغییرات ناگهانی و شدت آن را میتوان به کمک مشتق مشخص نماییم. اگر مقادیر محیط به صورت گسسته باشد به کمک تقریبی از مشتق میتوان تغییرات نسبت به پیکسل های مجاور را محاسبه کرد.برای تشخیص لبه های تصویر میتوان از فیلتر های خطی کمک گرفت. به منظور مشاهده عینی و درک درست تر این مفاهیم ما در آغاز فیلتر لبه یاب Robert را به تصویر mosque اعمال می کنیم و پس از آن شاهد مشاهده بهتر لبه ها و جزئیات می شویم.که نتایج حاصل را در تصاویر زیر میتوان مشاهده نمود:

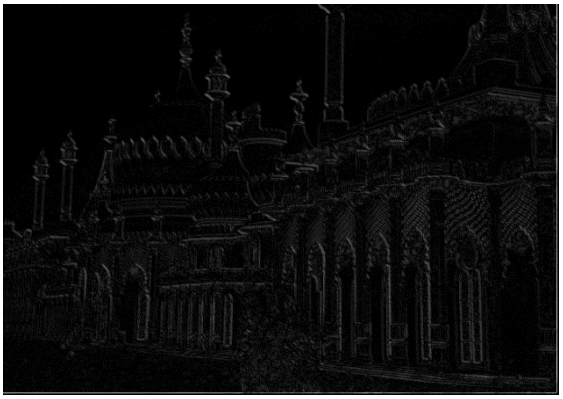
تصویر اصلی



اعمال فیلتر لبه یاب روبرت

در ادامه به اعمال فیلتر میانگین 3*3 به تصویر اصلی می پردازیم که نتایج قابل مشاهده است.

پس از آن آشکار سازی لبه Robert را به تصویری که فیلتر میانگین به آن زده شد(تصویر مرحله قبل) اعمال می کنیم که نتیجه زیر را برای ما به ارمغان می آورد:



همانطور که از تصاویر پیدا است با اعمال فیلتر میانگین بر تصویر اصلی تصویر به نسبت تصویر اولیه بلورتر و هموارتر میشود و جزئیات دیگر قابل مشاهده نمی باشد سپس هنگامی که ما فیلتر لبه یاب Robert را اعمال می کنیم دیگر قادر به مشاهده لبه های جزئی نیستیم.

همین مراحل را به کمک فیلتر میانگین 5*5 بر روی تصویر اولیه طی میکنیم و نتایج زیر را بدست می آوریم:

بار دیگر این مراحل را با فیلتر میانگین 7*7 نیز انجام می دهیم:

هنگامی که به مقایسه تصاویر با یکدیگر می پردازیم متوجه میشویم که با افزایش سایز فیلتر میانگین تصویر به مراتب بلورتر و هموارتر می شود و وقتی فیلتر لبه یاب Robert را بر روی آن ها اعمال می کنیم لبه ها به صورت کلی تر نمایان میشود به طور مثال با اعمال فیلتر میانگین 7*7 به نسبت فیلتر 5*5 میتوان میزان جزئیات کمتری را مشاهده کرد.

3-2-طراحی فیلتر 5*5 جدید

همانطور که در تمرین خواسته شده است ابتدا فیلتر میانگین 3*3 را دوبار بر روی تصویر اعمال می کنیم که نتیجه عکس زیر می شود :

حال اگر فیلتر 5*5 را به تصویر اصلی اعمال کنیم نتیجه زیر به دست می آید:

پر واضح است که این دو تصویر بدست آمده دارای نتایج مشابه نمی باشند پس باید فیلتر 5*5 ای را طراحی کنیم که نتیجه ای مشابه دوبار اعمال فیلتر 3*3 به ما بدهد . برای این کار به صورت آزمون و خطا جلو می رویم و متوجه این امر میشویم که بهتر است از یک فیلتر میانگین وزن دار 5*5 استفاده کنیم که به مرکز فیلتر بیشترین وزن را اختصاص می دهیم و هر چه از مرکز دور میشویم به همان میزان وزن های داده شده را کاهش می دهیم. حال اگر

نتایج این فیلتر طراحی شده جدید را با نتیجه دوبار اعمال فیلتر میانگین 3*3 مقایسه کنیم متوجه میشویم که نتایج مشابه یکدیگر هستند:

4-2- طراحی فیلتر میانگین 7*7

طبق آنچه در تمرین از ما خواسته شده است بر روی تصویر mosque اصلی سه بار فیلتر میانگین 3*3 را اعمال می کنیم که نتیجه زیر را به ما می دهد :

حال نیاز است که فیلتر 7*7 جدیدی را طراحی کنیم که نتیجه ای مشابه تصویر قبل (سه بار اعمال فیلتر 3*3) به ما بدهد بار دیگر روش پیشنهادی استفاده از فیلتر میانگین وزن دار 7*7 است که بعد از چندین آزمون و خطا به این روش می رسیم و به مرکز بیشترین وزن را اختصاص می دهیم و به پیکسل هایی که از مرکز دور میشند به میزان دور شدنشان از مرکز وزن اختصاص داده شده به آن ها را نیز کاهش می دهیم و نتیجه زیر را مشاهده می کنیم :

که همانطور که قابل حدس و پیش بینی نتیجه بدست آمده با فیلتر میانگین 7*7 جدید کاملا مشابه اعمال سه بار فیلتر میانگین 3*3 به تصویر mosque اصلی می باشد.

5-2-مراجع

[1] https://www.geeksforgeeks.org/box-blur-algorithm-with-python-implementation/

[2]https://stackoverflow.com/questions/53098631/average-filter-without-built-in-function

[3] https://gisman.ir/image-processing-filter-2/

کد های ضمیمه شده

```
[ ]  # 2
     mosque= cv2.imread("mosque.bmp")
     mosque = cv2.cvtColor(mosque,cv2.COLOR_BGR2GRAY)

     def robert_filter(image):

       robert_Xfilter = np.array([[1,0],[0,-1]])
       robert_Xfilter = np.array([[0,1],[-1,0]])

       Gx = np.zeros((image.shape[0],image.shape[1]))
       Gy = np.zeros((image.shape[0],image.shape[1]))

       for i in range(image.shape[0]):
         for j in range(image.shape[1]):
           Gx[i][j] = (robert_Xfilter * image[i:i+2,j:j+2]).sum()
           Gy[i][j] = (robert_Xfilter * image[i:i+2,j:j+2]).sum()

       G = np.sqrt(np.power(Gx,2)+np.power(Gy,2))

       return G


     def mean_filter(image,n):

       mean_filter = np.ones((n,n))/(n*n)
       G = np.zeros((image.shape[0],image.shape[1]))

       for i in range(image.shape[0]):
         if image.shape[0] - i < n:
           break
         for j in range(image.shape[1]):
           if image.shape[1] - j < n:
             break
           G[i][j] = (mean_filter * image[i:i+n,j:j+n]).sum()
```

```
  return G

mosque_robert = robert_filter(mosque)
cv2_imshow(mosque_robert)

mosque_mean_3 = mean_filter(mosque,3)
mosque_mean_3_robert = robert_filter(mosque_mean_3)
cv2_imshow(mosque_mean_3_robert)

mosque_mean_5 = mean_filter(mosque,5)
mosque_mean_5_robert = robert_filter(mosque_mean_5)
cv2_imshow(mosque_mean_5_robert)

mosque_mean_7 = mean_filter(mosque,7)
cv2_imshow(mosque_mean_7)
mosque_mean_7_robert = robert_filter(mosque_mean_7)
cv2_imshow(mosque_mean_7_robert)


def psedu_mean_nx_filter(image,n):

  if n == 5:
    mean_filter =  (np.array([[1,2,3,2,1],
                              [2,4,6,4,2],
                              [3,6,9,6,3],
                              [2,4,6,4,2],
                              [1,2,3,2,1]]))/81
  elif n == 7:
        mean_filter =  (np.array([[1,3,6,7,6,3,1],
                              [3,9,18,21,18,9,3],
                              [6,18,36,42,36,18,6],
                              [7,21,42,49,42,21,7],
                              [6,18,36,42,36,18,6],
                              [3,9,18,21,18,9,3],
                              [1,3,6,7,6,3,1]])/729)
```

```
  G = np.zeros((488,695))

  for i in range(image.shape[0]):
    if image.shape[0] - i < n:
      break
    for j in range(image.shape[1]):
      if image.shape[1] - j < n:
        break
      G[i][j] = (mean_filter * image[i:i+n,j:j+n]).sum()

  return G




mosque_mean_3 = mean_filter(mosque,3)
mosque_mean_3_2x = mean_filter(mosque_mean_3,3)
cv2_imshow(mosque_mean_3_2x)

mosque_mean_3 = psedu_mean_nx_filter(mosque,5)
cv2_imshow(mosque_mean_3)

mosque_mean_3 = mean_filter(mosque,3)
mosque_mean_3_2x = mean_filter(mosque_mean_3,3)
mosque_mean_3_3x = mean_filter(mosque_mean_3_2x,3)
cv2_imshow(mosque_mean_3_3x)

mosque_mean_7 = psedu_mean_nx_filter(mosque,7)
cv2_imshow(mosque_mean_7)
```