Examples

October 20, 2018

In [27]:
```
import sys
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
sys.path.append("src")
import util.data.bmp as bmp
```

In [28]:
```
img=bmp.read_image("test/images/avion.bmp")
```

In [29]:
```
def proccess_image(image, func, *args, **kwargs):
    result = dict()
    for chanel in image:
        result[chanel] = func(image[chanel],*args, **kwargs)
    return bmp._merge_rgb(result).astype(np.uint8)

def from_dict(img):
    return bmp._merge_rgb(img).astype(np.uint8)

def show_result(img, result):
    f, axarr = plt.subplots(1,2, figsize = (10, 15))
    axarr[0].imshow(from_dict(img))
    axarr[1].imshow(result)
```

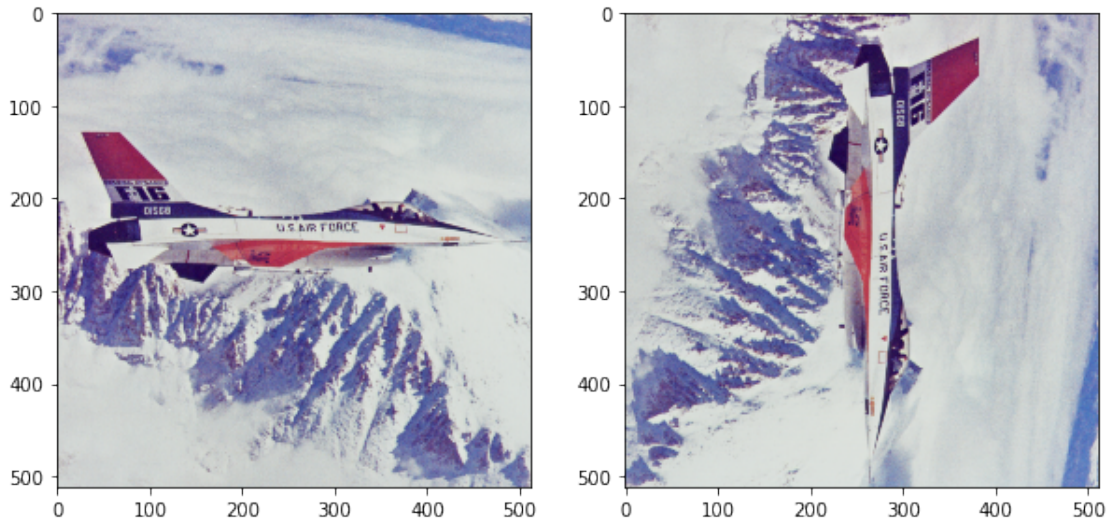
1

In [30]:
```
from transform.rotate import rot
```

In [31]:
```
%%timeit
result = proccess_image(img, rot, degree=90)
```

```
285 ms ś 36.9 ms per loop (mean ś std. dev. of 7 runs, 1 loop each)
```

In [49]:
```
show_result(img, result)
```



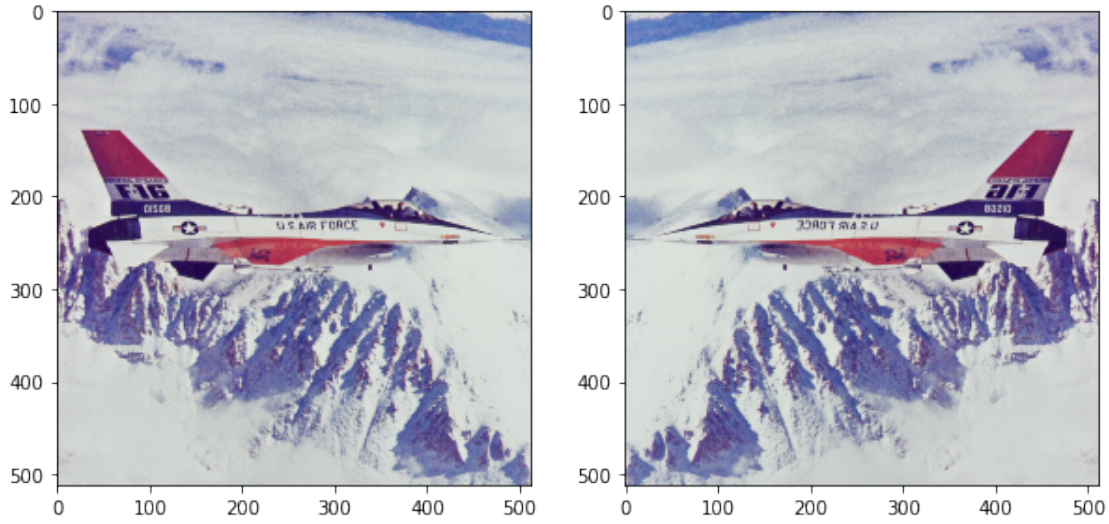
2

```
In [33]: from transform.mirror import mirror

In [34]: %%timeit
         result = proccess_image(img, mirror, 'x')
```

```
139 ms ś 5.41 ms per loop (mean ś std. dev. of 7 runs, 10 loops each)
```

```
In [51]: show_result(img, result)
```



3

```
In [36]: from filter.sobel import sobel
         def add_128_sobel(*argc, **kwargs):
             res = sobel(*argc, **kwargs)
             for i in range(len(res)):
                 for j in range(len(res[0])):
                     res[i][j] += 128
             return res
```

```
In [37]: %%timeit
         result = proccess_image(img, add_128_sobel, direction = 'x', mode = 'odd')
```

```
13.8 s ś 614 ms per loop (mean ś std. dev. of 7 runs, 1 loop each)
```

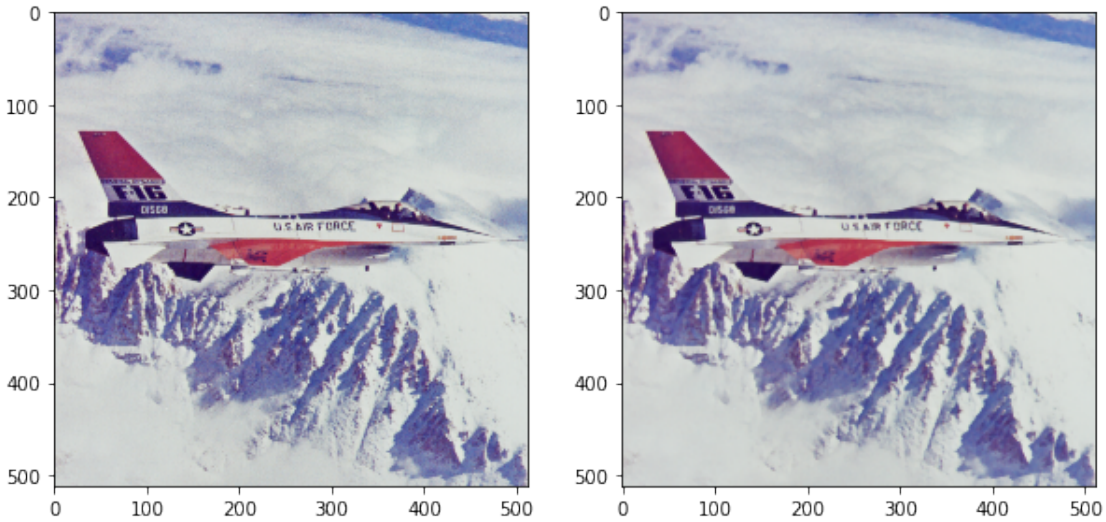
4

```
In [39]: from filter.median import median
```

```
In [40]: %%timeit
         result = proccess_image(img, median, rad=1)
```

```
50.6 s ś 1.1 s per loop (mean ś std. dev. of 7 runs, 1 loop each)
```

```
In [55]: show_result(img, result)
```



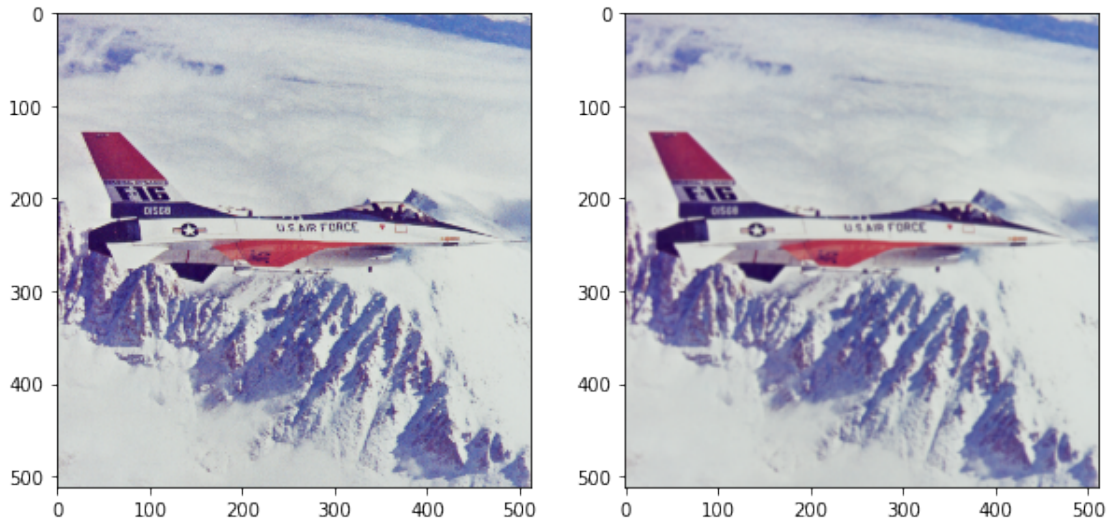
5

```
In [42]: from filter.gauss import gauss

In [43]: %%timeit
         result = proccess_image(img, gauss, mode = 'rep', sigma = 0.5)

27.6 s ś 741 ms per loop (mean ś std. dev. of 7 runs, 1 loop each)


In [57]: show_result(img, result)
```



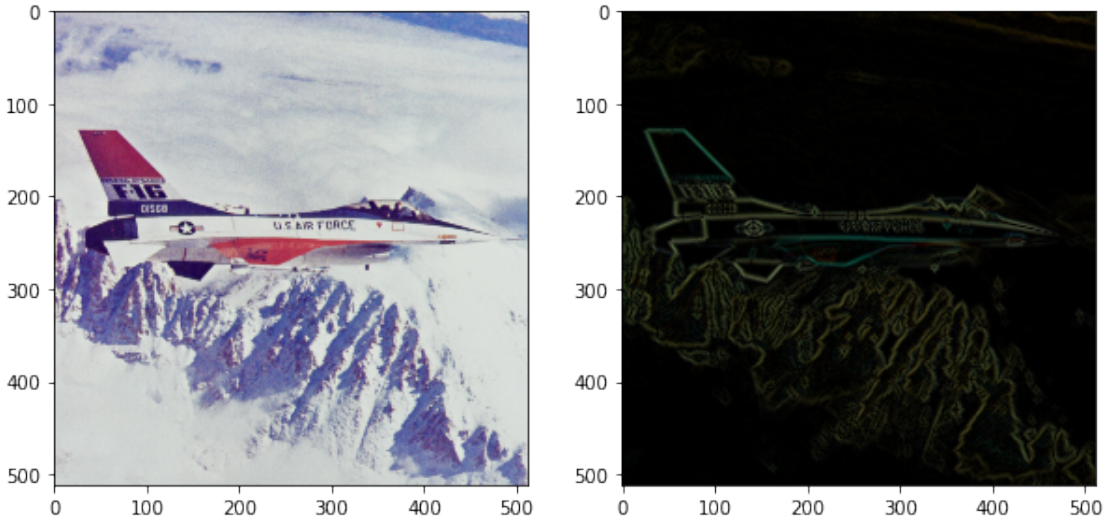
6

```
In [45]: from filter.gradient import gradient
         def normalized_gradient(*argc, **kwargs):
             res = gradient(*argc, **kwargs)
             for i in range(len(res)):
                 for j in range(len(res[0])):
                     res[i][j] = min(max(res[i][j], 0), 255)
             return res

In [46]: %%timeit
         result = proccess_image(img, normalized_gradient, 'rep', 1)

48.9 s ś 1.08 s per loop (mean ś std. dev. of 7 runs, 1 loop each)


In [47]: show_result(img, result)
```


0
100
200
300
400
500
0
100
200
300
400
500
U.S. AIR FORCE
0
100
200
300
400
500
0
100
200
300
400
500