

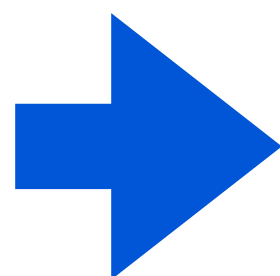
# メディアンフィルタ

(教科書p.42)

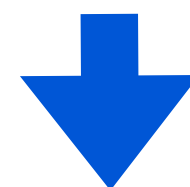
- 非線型フィルタの一つ
  - 中央値による計算（メディアン）
  - 局所的特徴を保持したままノイズを除去

8	9	1
2	3	8
7	6	1

画素値の例

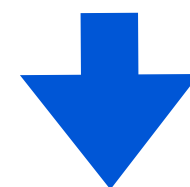


注目画素と周辺画素  
8,9,1,2,3,8,7,6,1



並べ替え

1,1,2,3,6,7,8,8,9



中央値

6

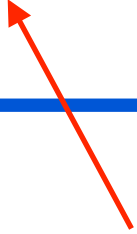
# 演習: メディアンフィルタ

- プロジェクト名: median
- OpenCVのメディアンフィルタ関数を用いて実装

```
cv::medianBlur(入力画像, 出力画像, フィルタサイズ);
```

- 入力画像をhouse.jpgに変更

※数字だけでよい  
(例えば3とか5)



関数を使わない方法は  
教科書p.45を参照

# メディアンフィルタ処理の流れ

1. 画像をグレースケールで入力
2. 出力画像の宣言
3. メディアンフィルタ処理
4. 表示

先週のgaussianプログラムを参考に  
median.cppを作成しましょう

ポイント : gaussianBlur→medianBlur

演習中 . . .



# median.cppの例

```
#include <iostream>
#include <opencv2/opencv.hpp>
#define FILE_NAME "house.jpg"

#define WINDOW_NAME_INPUT "input"    //ウィンドウ名
#define WINDOW_NAME_OUTPUT "output"
#define FILTER_SIZE (5) //フィルタサイズ (3以上の奇数)

int main(int argc, const char * argv[]) {
    //1. 画像をグレースケールで入力
    cv::Mat src_img = cv::imread(FILE_NAME, cv::IMREAD_GRAYSCALE);
    if (src_img.empty()) { //入力失敗の場合
        fprintf(stderr, "Cannot read image file: %s.\n", FILE_NAME);
        return (-1);
    }
    //2. 出力画像の宣言
    cv::Mat dst_img;
    //3. メディアンフィルタ処理
    cv::medianBlur(src_img, dst_img, FILTER_SIZE);
    //4. 表示
    cv::imshow(WINDOW_NAME_INPUT, src_img);
    cv::imshow(WINDOW_NAME_OUTPUT, dst_img);
    cv::waitKey();    //キー入力待ち

    return 0;
}
```

5×5のメディアンフィルタ



# メディアンフィルタ処理結果例



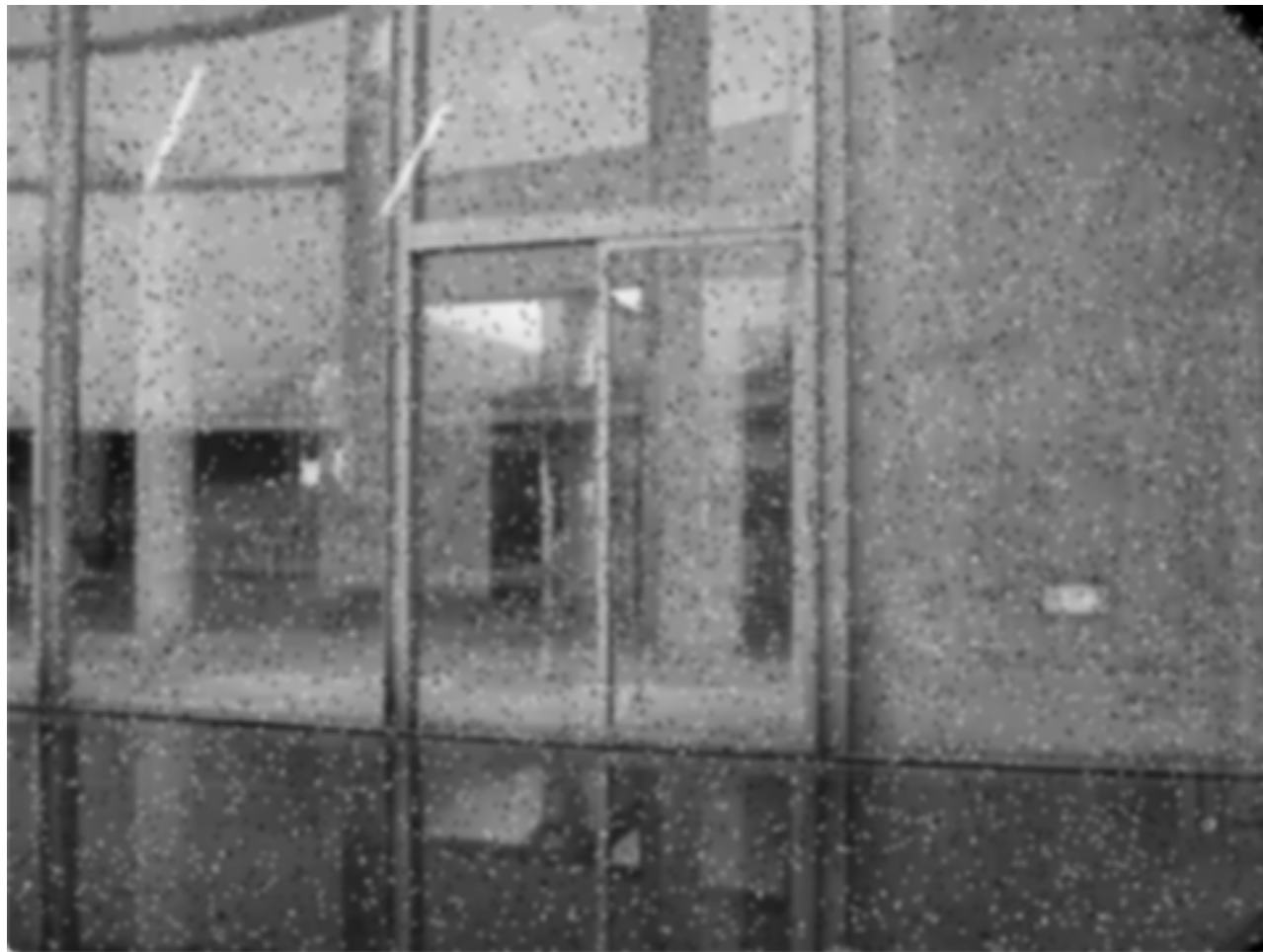
入力画像



メディアンフィルタ  
処理後の画像



# 参考：ガウシアンフィルタとの比較



ガウシアンフィルタ  
( $5 \times 5$ )  
処理後の画像



メディアンフィルタ  
処理後の画像