画像処理

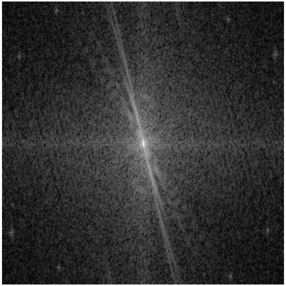
２次元フーリエ変換の利用

宮崎大学 工学部 情報システム工学科

３年後期
第 ６ 回

パワースペクトル

各周波数成分の波の強さ（振幅）を図示したもの
対数スケールで表示することが多い

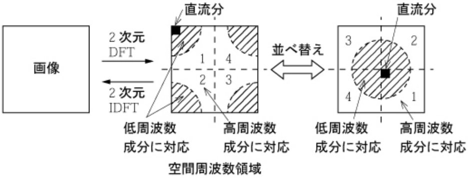


原点：直流成分

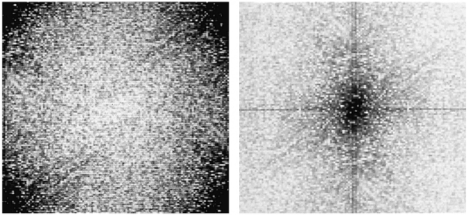
(k, l)：周波数 k/M, l/N

周波数成分の配置(p.42)

前述の定式化では，直流成分は画像の左上にくる



直流成分が中央になるように再配置を行う



(a)　2次元DFTによる　　(b)　光学的2次元DFTによる

畳み込み(p.40)，微分(p.43)

空間フィルタリングは，周波数領域での掛算と等価

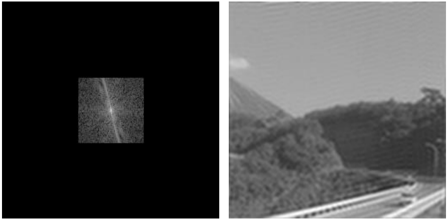
$$
\begin{aligned}
h(t) &= f(t) \otimes g(t) = \int_{-\infty}^{\infty} f(\tau) g(t-\tau) d\tau \\
H(\omega) &= F(\omega) G(\omega)
\end{aligned}
$$

空間微分も，周波数領域で演算に変換可能

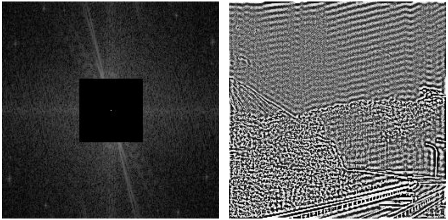
$$
\begin{aligned}
f(x, y) &\leftrightarrow F(k, l)\text{の時} \\
df(x, y)/dx &\leftrightarrow j2\pi k F(k, l) \\
df(x, y)/dy &\leftrightarrow j2\pi l F(k, l) \\
d^2 f(x, y)/dxdy &\leftrightarrow (j2\pi)^2 kl F(k, l) \\
\triangle f(x, y) &\leftrightarrow (j2\pi)^2 (k^2 + l^2) F(k, l) \\
(&= \partial^2 f/\partial x^2 + \partial^2 f/\partial y^2)
\end{aligned}
$$

周波数フィルタ (p.43)

一定の周波数成分のみを出力



ローパス



ハイパス

周波数フィルタ (図2.32)

(a)　原画像（Girl）（128×128 画素, 8 ビット）

(b)　低域通過フィルタによる処理結果

(c)　高域通過フィルタによる処理結果

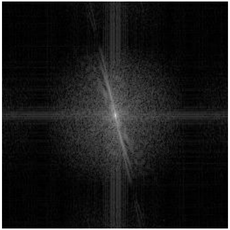
ハイパスフィルタのプログラム

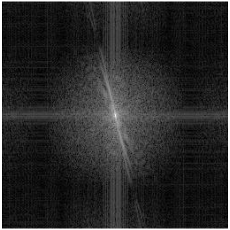
```
void hi_path(K_IMAGE *iimg, K_IMAGE *oimg, int rx, int ry
{ float ***iptr = (float ***)k_data(iimg);
  float ***optr = (float ***)k_data(oimg);
  int M = k_xsize(iimg); int N = k_ysize(iimg);
  for(int y = 0; y < N; y++) {
  for(int x = 0; x < M; x++) {
    if (sqr(x-M/2)/sqr(rx) + sqr(y-N/2)/sqr(ry)<1) {
      optr[0][y][x] = optr[1][y][x] = 0.0;
    }else{
      optr[0][y][x] = iptr[0][y][x];
      optr[1][y][x] = iptr[1][y][x];
    }
  } }
  optr[0][N/2][M/2] = iptr[0][N/2][M/2];
  optr[1][N/2][M/2] = iptr[1][N/2][M/2];
}
```

高域強調フィルタ

高周波数を強調することにより，鮮鋭化

$$G'(k,l) = G(k,l) \times (\sqrt{k^2 + l^2} + 1.0)$$



画像の復元 (P.117)

画像劣化の原因を数学的にモデル化し，
その逆操作を加えることにより
原画像を推定する処理

画像劣化のモデル

得られた画像 g(x,y) は，

原画像	f(x,y) に，
点広がり関数	h(x,y) が畳み込まれ，
ノイズ	n(x,y) が加算されたとみなす．

$$g(x,y) = \iint h(x-\alpha, y-\beta) f(\alpha,\beta) d\alpha d\beta + n(x,y)$$

点広がり関数の例（ 焦点ぼけ ）

円形のレンズ領域を通る全ての光線が加算された像

$$h(x, y) = \begin{cases} 1 & \text{for } x^2 + y^2 < \alpha \\ 0 & \text{else} \end{cases}$$

点広がり関数の例（大気擾乱）

光線周囲の明度が，光線からの距離に応じた重みで加算された像

$$h(x, y) = \exp\left(\frac{x^2 + y^2}{\sigma^2}\right)$$

点広がり関数の例（動きぶれ）

動きによる移動範囲内の光線が加算された像

$$h(x, y) = \exp\left(y^2/\sigma^2\right)$$
$$h(x, y) = \exp\left(x^2/\sigma^2\right)$$

原画像

垂直方向のぶれ

水平方向のぶれ

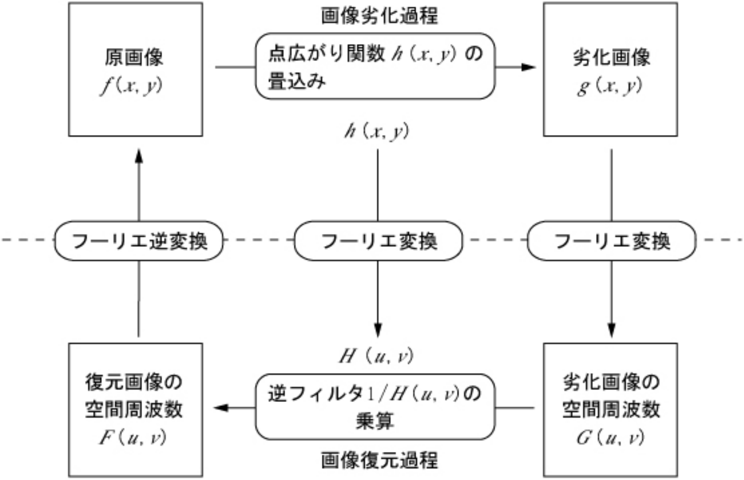
逆フィルタ

雑音を無視し，周波数領域で考えると原画像 F(u,v) は，
得られた画像 G(u,v) を点広がり関数 H(u,v) で割ったもの

$$
\begin{aligned}
g(x,y) &= \iint h(x-\alpha, y-\beta) f(\alpha,\beta) d\alpha d\beta \\
G(u,v) &= H(u,v)F(u,v) \\
F(u,v) &= \frac{G(u,v)}{H(u,v)}
\end{aligned}
$$

$\frac{1}{H(u,v)}$: 逆フィルタ

逆フィルタによる復元


画像劣化過程
原画像
f(x, y)
点広がり関数 h(x, y) の畳込み
劣化画像
g(x, y)
h(x, y)
フーリエ逆変換
フーリエ変換
フーリエ変換
H(u, v)
復元画像の空間周波数
F(u, v)
逆フィルタ 1/H(u, v)の乗算
劣化画像の空間周波数
G(u, v)
画像復元過程

逆フィルタの問題点

- 実際に観測される画像にはノイズが含まれる
  ⇒ 雑音成分も復元される
- 点広がり関数の値が非常に小さくなる場合がある
  ⇒ 数値的に不安定

特に，高周波成分でこの傾向が強まる

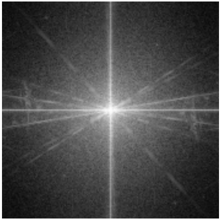
解決法

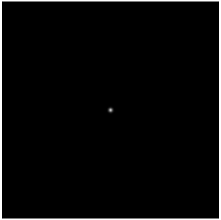
カットオフ周波数 ω_0 を定め，下記のフィルタ M(u,v) により復元する

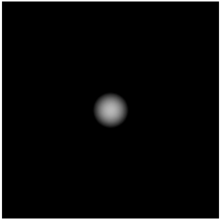
$$M(u,v) = \begin{cases} \frac{1}{H(u,v)} & \left(\sqrt{u^2+v^2} \leq \omega_0\right) \\ 1 & \left(\sqrt{u^2+v^2} > \omega_0\right) \end{cases}$$

逆フィルタの適用例

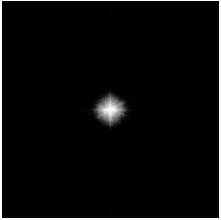

原画像

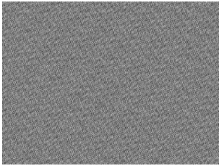

パワースペクトル


点広がり関数


パワースペクトル


ぼけ画像


パワースペクトル


逆フィルタの出力


低周波成分のみ

逆フィルタプログラム

```
#define sqr(x)  ((x)*(x))
void inv_filter( K_IMAGE *fimg,  // 入力複素画像
                 K_IMAGE *gimg,  // 点広がり関数複素画像
                 K_IMAGE *oimg,  // 出力複素画像
                 int rx, int ry) // カットオフ周波数
{ float ***fptr = (float ***)k_data(fimg);
  float ***gptr = (float ***)k_data(gimg);
  float ***optr = (float ***)k_data(oimg);
  int M = k_xsize(iimg); int N = k_ysize(iimg);

  for(int y = 0; y < N; y++){
  for(int x = 0; x < M; x++){
    if (sqr(x-M/2)/sqr(rx)+sqr(y-N/2)/sqr(ry)<1) { //カットオフの検査
      float d = sqr(gptr[0][y][x])+sqr(gptr[1][y][x]); // 複素数の割算
      optr[0][y][x] =
       (fptr[0][y][x]*gptr[0][y][x]+fptr[1][y][x]*gptr[1][y][x])/d;
      optr[1][y][x] =
       (fptr[1][y][x]*gptr[0][y][x]-fptr[0][y][x]*gptr[1][y][x])/d;
    }else{
      optr[0][y][x] = fptr[0][y][x]; // そのまま出力
      optr[1][y][x] = fptr[1][y][x];
    }
  } }
}
```

ウィナーフィルタ

原画像 $f(x, y)$ と復元画像 $f_0(x, y)$ との平均二乗誤差を最小化

$$\begin{aligned}
g(x,y) &= \iint h(x-\alpha, y-\beta) f(\alpha,\beta) d\alpha d\beta + n(x,y) \\
G(u,v) &= H(u,v)F(u,v) + N(u,v) \\
f_0(x,y) &= \iint m(x-\alpha, y-\beta) g(\alpha,\beta) d\alpha d\beta \\
F_0(u,v) &= M(u,v)G(u,v) \\
E[\| f - f_0 \|^2] &\rightarrow \min \\
E[\| F - F_0 \|^2] &\rightarrow \min \\
\ldots & \\
M(u,v) &= \frac{\overline{H(u,v)}}{| H(u,v) |^2 + E[|N(u,v)|^2]/E[|F(u,v)|^2]}
\end{aligned}$$

ウィナーフィルタの特性

ノイズの性質 N(u,v)，原画像の性質 F(u,v) は一般に未知
⇒ N(u,v)/F(u,v) を定数 Γ で置き換える

$$M(u,v) = \frac{\overline{H(u,v)}}{|H(u,v)|^2 + \Gamma}$$

Γ=0： 逆フィルタと等価
Γ 小： 雑音低減能力は低いが，復元能力は高い
Γ 大： 雑音低減能力は高いが，復元能力は低い

ウィナーフィルタの適用例

(a)　原画像

(b)　ぼけと雑音で劣化した画像

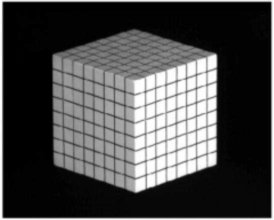
(c)　低周波成分のみの
逆フィルタによる復元

(d)　ウィーナ・フィルタによる
復元（Γ=0.1）

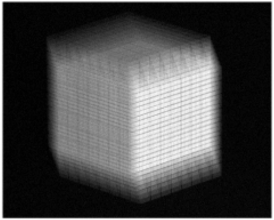
(e)　ウィーナ・フィルタによる
復元（Γ=0.01）

(f)　ウィーナ・フィルタによる
復元（Γ=0.001）

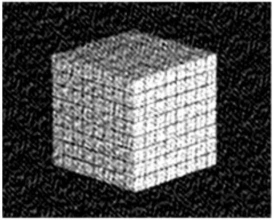
ウィナーフィルタの適用例



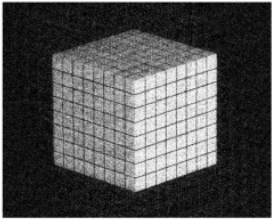
(a)　原画像



(b)　ぼけと雑音で劣化した画像



(c)　低周波成分のみの逆フィルタによる復元



(d)　ウィーナ・フィルタによる復元（$\Gamma=0.001$）

まとめ

フーリエ変換

- 周波数領域処理
  パワースペクトル／畳み込み演算
  ハイパス／ローパスフィルタ
  高域強調フィルタ

フーリエ変換を利用した画像復元

- 画像劣化のモデル
- 逆フィルタ／ウィナーフィルタ

宿題06

beamer>
ウィナーフィルタにより画像復元を行う関数
`wiener_filter()` を完成させよ

```
void wiener_filter(K_IMAGE *fimg,//入力複素画像
           K_IMAGE *gimg,        //点広がり関数複素画像
           K_IMAGE *oimg,        //出力複素画像
           float gamma)          //定数Γ
```

- fimg は入力画像のフーリエ変換結果が入った複素画像
- gimg は点拡がり関数のフーリエ変換結果が入った複素画像
- oimg はウィナーフィルタを適用した複素画像
- gamma はウィナーフィルタの定数

※ 周波数領域での処理のみ記述すればよい

<beamer>