フーリエ変換概説

デルタ関数$\delta(t)$の定義

任意の関数$f(t)$について

$$\int_{-\infty}^{\infty} f(t)\,\delta(t)\,dt = f(0) \qquad (1)$$

が成立する$\delta(t)$を考え、これを$\delta(t)$の（形式的）定義とする（一種の公理）。デルタ関数については積分に意味があり、単独ではあまり意味がない。上記の性質を持つものを総称して$\delta(t)$で表す。

デルタ関数の性質

$\int_{-\infty}^{\infty} f(t)\,\delta(\tau - t)\,dt$　（$f(t)$と$\delta(t)$のコンボルーション）について、$\tau - t = t'$と置くと

$= \int_{-\infty}^{\infty} f(\tau - t')\,\delta(t')\,dt'$　（これはコンボルーションの一般的性質 $f * g = g * f$から導いてもよい）　デルタ関数の定義から

$= f(\tau - t')\Big|_{t'=0} = f(\tau)$　よって

$$\int_{-\infty}^{\infty} f(t)\,\delta(\tau - t)\,dt = f(\tau) \qquad (2)$$

フーリエ変換および逆変換

$f(t)$について

$F(\omega) = F\{f(t)\} = \int_{-\infty}^{\infty} f(t)\,e^{-j\omega t}\,dt$を形式的に定義する。また$F(\omega)$の逆フーリエ変換$f(t)$を

$F^{-1}\{F(\omega)\} = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)\,e^{j\omega t}\,d\omega = f(t)$によって形式的に定義する。

デルタ関数$\delta(t)$についてフーリエ変換は

$F\{\delta(t)\} = \int_{-\infty}^{\infty} \delta(t)\ e^{-j\omega t}dt = e^{-j\omega 0} = e^{0} = 1$　これを形式的に逆変換したものは

$$F^{-1}\{1\} = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{j\omega t}\,d\omega$$

（上記で、ここの用は済むが、ついでに：

$= \frac{1}{2\pi}\int_{-\infty}^{\infty}\left(\cos\omega t + j\sin\omega t\right)d\omega$ 後半は奇関数のため0。$\cos\omega t$は偶関数だから

$= \frac{1}{\pi}\int_{0}^{\infty}\cos\omega t\, d\omega = \frac{1}{\pi}\frac{\sin\omega t}{t}\Big|_{0}^{\infty} = \frac{1}{\pi}\lim_{\omega\to\infty}\frac{\sin\omega t}{t}$　）

一般の逆変換の性質$F^{-1}\left\{F(\omega)\right\} = f(t)$が任意の$f(t)$について妥当することを示すに先だって、この性質が$\delta(t)$については成立することをここでは公理とする（実際はRiemann-Lebesgueの補助定理から導ける）。このとき右辺は$\delta(t)$となる。よって

$\frac{1}{2\pi}\int_{-\infty}^{\infty}e^{j\omega t}d\omega = \delta(t)$　　(3)　　（標記として　$\delta(t) \leftrightarrow 1$がよく使われる。$\frac{1}{2\pi}\int_{-\infty}^{\infty}\cos\omega t\, d\omega = \delta(t)$の形も記憶しておくとよい）

フーリエ逆変換の妥当性

$F(\omega) = F\left\{f(t)\right\} = \int_{-\infty}^{\infty}f(t)\,e^{-j\omega t}dt$なるフーリエ変換に対し逆変換

$F^{-1}\left\{F(\omega)\right\} = \frac{1}{2\pi}\int_{-\infty}^{\infty}F(\omega)\,e^{j\omega t}d\omega = f(t)$が成立することを示す（すなわち、$f(t) \to F(\omega)$　に対し$f(t) \leftarrow F(\omega)$　）。

$F^{-1}\left\{F(\omega)\right\} = \frac{1}{2\pi}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}f(t')\,e^{-j\omega t'}dt'\right]e^{j\omega t}d\omega$　積分順序を変更して

$= \int_{-\infty}^{\infty}f(t')\left[\frac{1}{2\pi}\int_{-\infty}^{\infty}e^{j\omega(t-t')}d\omega\right]dt'$　　内側の積分は(3)より$\delta(t-t')$よって

$= \int_{-\infty}^{\infty}f(t')\,\delta(t-t')\,dt'$　　(2)から

$= f(t)$

参考書

久保田　一：分りやすいフーリエ解析、オーム社、2500円

A.パポリス著、大槻　喬、平岡寛二：工学のための応用フーリエ積分、オーム社、1500円　（よい本だが、残念ながら絶版）

D.H. Ballard, C.M. Brown　福村晃夫ほか訳：コンピュータビジョン、日本コンピュータ協会、8034円（フーリエ変換の使い方が簡潔にまとまっているので参照するとよい）

点広がり関数について

$f(x) = \delta(x)$のとき

$$h(y) = \int_{-\infty}^{\infty} \delta(x) g(y-x) dx = g(y)$$

すなわち、入力に$\delta(x)$を入れたときの各点の応答が$g(x)$である。

.

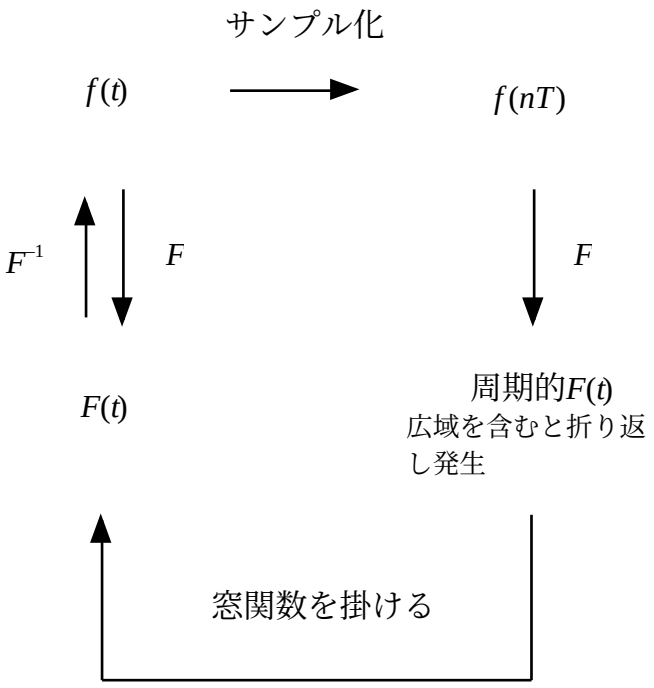
画像のフーリエ変換

Ballard参照

標本化定理

Ballard参照

サンプルからの元データの再現



サンプリング定理の図式

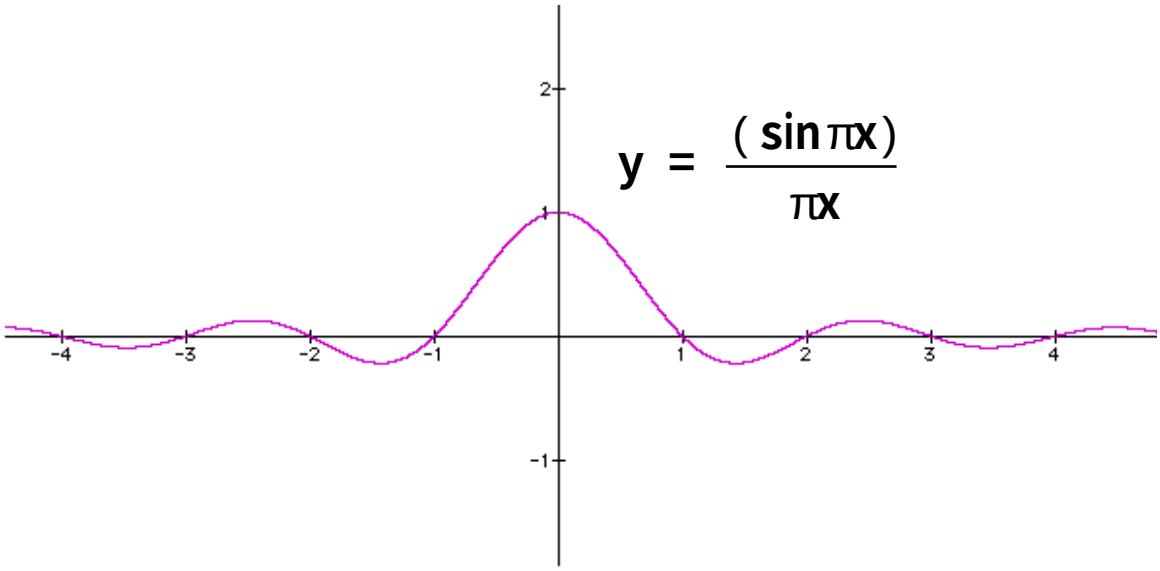
サンプル定理によると

$F\{f(nT)\}P_W = F(t)$　　両辺を逆変換して

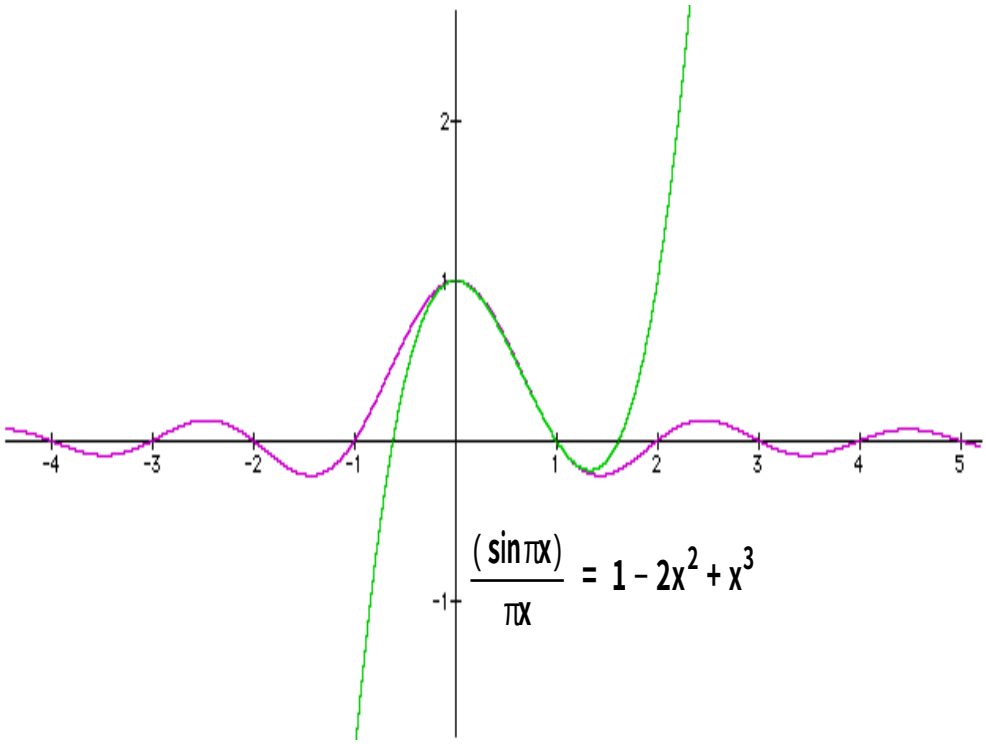
$f(t) = f(nT) * F^{-1}\{P_W\}$　　(*はコンボルーション)

$$= f(nT) \ * \frac{\sin W\pi x}{\pi x}$$

サンプルされた元の信号に、窓の逆変換をぼかし関数として施せば、元の信号を復元できる。



窓の逆関数（sinc関数）（Apple社グラフ計算機使用）



sinc関数の多項式近似

エッジデータの構造化

エッジ抽出・細線化しただけでは線分が求まったことにならない。構造化する必要がある。

線の抽出

候補点の抽出

エッジオペレータと閾値処理、細線化などでエッジの候補点を求める。

線の追跡

追跡の出発点を求める、隣接する点を最適に結合する。よく似た方向を持つエッジ要素を結合。

エッジ要素　確実なものだけを残したい←→弱い線もとらえたい　矛盾する要求

（とぎれがちとなる）　（間違った線が生じがち）

一旦確実な（と思われる）線を出し、その近くで弱い線も探す

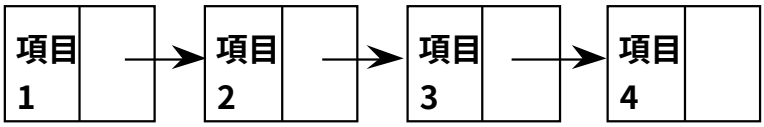
参考　ストラテジ、ヒューリスティック、セマンティック（コンテキスト）センシティブ　　単純一様処理

データ構造

一般に計算機上のデータの蓄え方をデータ構造という。高度な処理を行うためにはふさわしいデータ構造が必要である。１次元配列を線形データ構造という。画像を処理するには2次元配列を用いるのが普通である。

データ構造には

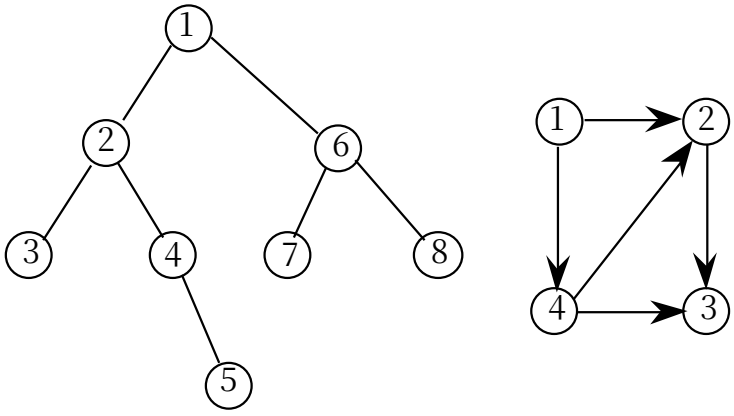
線形、リスト、グラフ、リング、木等がある。



番地	名前	次
0	-	1
1	項目1	3
2	項目4	0
3	項目2	4
4	項目3	2

リスト構造の例

リスト構造はデータの加除が容易である。



木

グラフ

参考書　A.V.エイホ、J.E.ホップクロフト、J.D.ウルマン著、野崎、野下ほか訳、アルゴリズムの設計と解析I・II、サイエンス社

チェインコード（符号化）法

3	2	1
4		0
5	6	7

エッジ点を抽出し、それを追跡した結果は順序付けられた点の列になる。これを合理的に表したい。

各点における線の向きを8方向の1つを表す数字（チェインコード）で表し、その数字のつながりで図形を表す。チェインコードで表した曲線（線図形）は、データ量の節約になり、一斉に数字を加減することで回転させることができる（例：一斉に+1すると、π / 4回転する）。

Hough変換

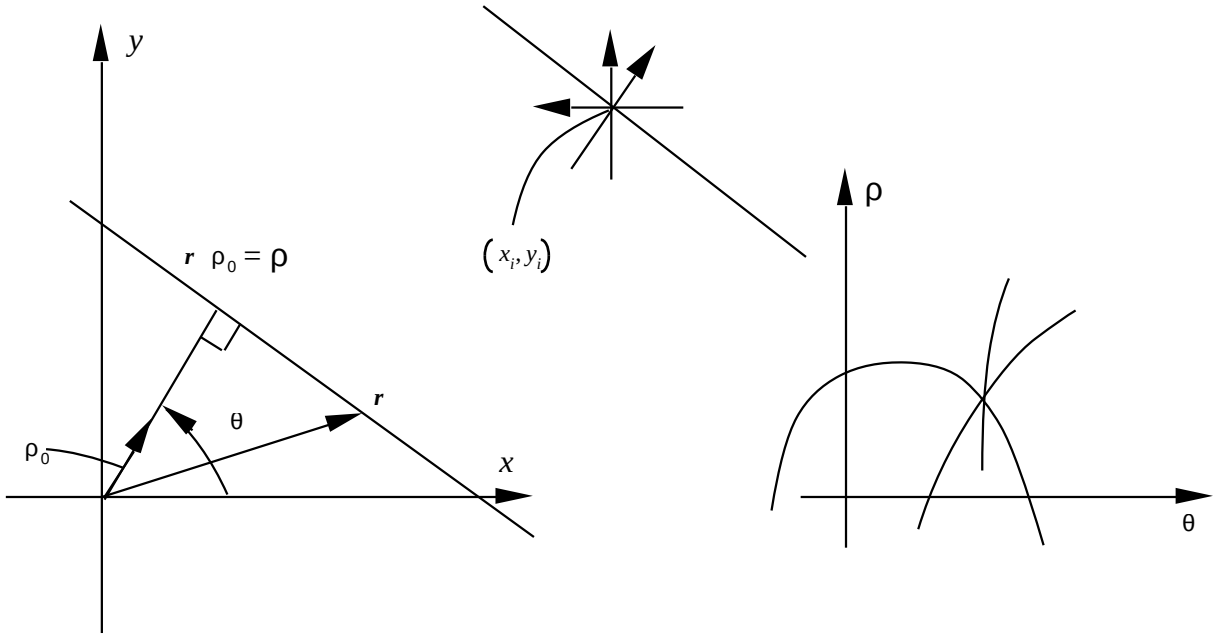
準備

直線の表し方

$y = ax + b$　　（y をxの従属関数とみなしている）

$ax + by = 1$　　（y とx対等、切片表示）

$x \cos\theta \; y \sin\theta = \rho$　　（ρ – θ形式、Hesseの標準形）



・線分を求めるのに追跡などの方法では制御が複雑なので、計算量が増えてもよいから機械的に求まる方法があるとよい。

・エッジ点は線分の１構成要素と考える。1つのエッジ点は、ある直線上の1点と考えるとその点を中心として、多数の可能性がある。その可能性を全て仮定してみる。

・1つの直線上の点からなる仮定は、個々にはばらばらでも。必ず同一点（真実）の仮定を含む。

Hough変換で直線を抽出する方法

・（エッジ点抽出）直線の候補をなす点群の入力

・ρ、θで番号づけられる２次元アキュムレータ配列を用意し、各セルを0にする。

・$x_i\cos\theta + y_i\sin\theta = \rho$　に従って、各(x_i, y_i)ごとに該当するρ、θの曲線に相当するセルに値1を加算。

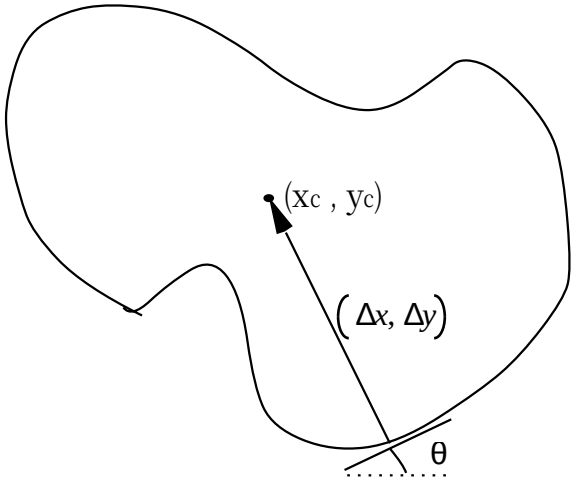
・全(x_i, y_i)について処理終了後、度数の高いセルのみを抽出する。それらに該当するρ、θの直線が存在するとみなす。

Hough変換は多数決（投票、voting）を基本としている。多くの発展形がある。上記で投票に１票づつのかわりに何らかの信頼の尺度に比例させた票を入れてもよい。たとえばエッジの強度など。ρ、θを未知としたが、θをエッジ抽出の際に得られるエッジの向きの情報を用いてもよい。

n個のパラメータで規定される対象を見つけたいとき、n次元のパラメータ空間を想定し、それに対応するアキュムレータ配列を用意して観測点１つごとに投票することによって対象を探すことができる。ただし、あまり次元が高いとアキュムレータ配列の大きさが大きくなりすぎるために現実に使えるのは次元が低いものに限られる。

Hough変換は手法として有用だが、実際の問題に適用するには計算量が多く、必ずし

もベストの方法とは限らない。



一般化Hough変換（Ballard & Brownによる）

あらかじめ形状の分っているものを画像中から探すものとする。

・元の図形から、各点ごとにその点からみて図形の重心がどの相対位置にあるか計算する。同時にその点に図形上で引いた接線の角度も求めておく。接線角度と相対位置の対応関係の表を作っておく。

・２次元の画像と同じサイズのアキュムレータ配列（物体の存在する場所を示す）を用意し0に初期化する

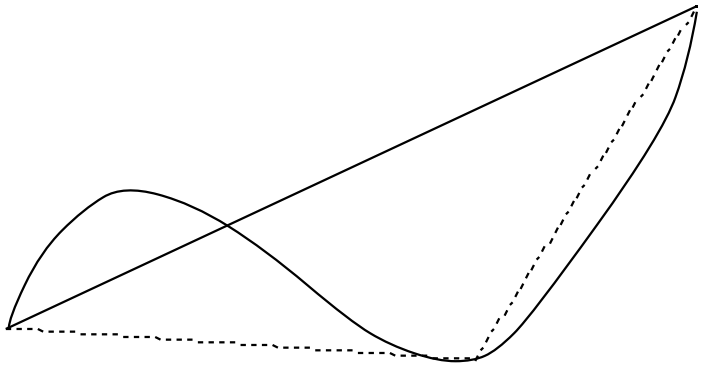
・観測した画像からエッジ点を抽出し、各点のエッジ方向を求める

・各点ごとに接線角度と相対位置の対応関係の表から、推測される重心位置を求め、その位置のアキュムレータ配列のセルに+1する

・全点について処理終了後、度数の高いセルのみを抽出する。それらに該当する重心点の位置に図形が存在するとみなす。

θ	Δ*x*	Δ*y*

Strip Tree（Ballard & Brownによる）



出発点と終点を結ぶ点列が定っているとき、区分化する。まず出発点と終点を結び、その線からのずれが最大となる点を求め、ずれが大きいときはその点で全体を二分する（ずれが小さいときはそのまま）。以後、各部分について出発点・終点が定っている点列とし、同様に繰返す。全体の処理が終ると折れ線近似された結果が得られる。