Feature detection, Matching

Shape Features → extract Shapes → St, Circle, Arbitrary
Architectural model, Pose estimation, Analysis of docs

St. eq = $y = mx + c$ m → slope c → intersection of y Co-ordinate

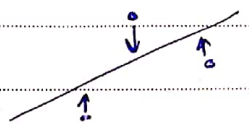
how to fit line?

① Successive approximation → Curve → picewise linear / Polylinear / BSpline
② least Square fit (over Constraint) → More than 2 points required
③ RANSAC (Constrain) → only 2 points is required
④ hough transform (under Constrain) → one point is required

① Successive approximation (Curve) → simpler representation
⇒ line simplification Method → picewise liner / Polyliner / BSpline
⇒ oldest, Smallest method
⇒ recursive subdevide the Curve at the point furthest away from line joining 2 end points

② least Square fit (over Constraint) More than 2 points
⇒ Standard linear Solution to estimate unknowns
if we know which Point belong to which line
(or) if there is only one line

$y = mx + c = f(x, m, c)$
slope → intersect with y



Minimum

$$E = \sum [y_i - f(x, m, c)]^2$$

take derivative of m, c → 0

/ /

Line Fitting

→ $B = AD$

→ $D = (A^T A)^{-1} A^T B$

---

③ RANSAC (Random Sampling and Consensus)

(Constraint) only 2 points are needed

① Randomly select 2 points to fit line

② find error between the estimated sol. and all other points

if the error is less than tolerance → quit

else go to step 1

Advantage → No accumulator array, space efficient, No voting

Disadvantage → Many hypothesses may need to be generated

فرضيات

---

④ Line hough transform (under Constraint → only one point)

→ having edges (vote) for plausible line locations

each edge point votes for all possible line pass through it and lines corresponding to high accumulator or bin values are examined for potential line fit

↘

not (x, y) Co-ordinate → (Polar) (r, θ) Co-ordinate

$r_i(\theta) = x_i \cos\theta + y_i \sin\theta$

oriented hough transform

An edge is parameterized in Polar (r, θ)

→ $n_1 = (\cos\theta, \sin\theta)$ $r_i = n_i \cdot x_i$

$+ \quad y$

(r,θ) accumulator array showing Notes for edges

r = قيمة سالبة → $-r_{max}$, r_{max} ✓

θ = قيمة موجبة → $0°$ → $360°$ ✓

hough transform Algo based on oriented edge Segments

hough (x, y, θ) → $\hat{n}$ / line

① clear accumulator arry

② for each detected edge at location (x,y) and orientation $\theta = \tan^{-1}\frac{n_y}{n_x}$

Compute $d = x n_x + y n_y$ ←

and increment Corresponding (θ, d) accumulator

③ find Peaks in the accumulator Corresponding to lines

④ optionally refit lines to the Constituent goals

Equation of line in (C,m) space

Line fit

① line equation = $y = mx + c$ m → slope

② Re-write equation ⇒ $c = (-x)m + y$ c → intesect with y

③ for Particular edge ⇒ $c = (-x_i)m + y_i$

④ equation of line in (c,m) space

hough Algorith for fitting st line

① Quantize the Parameter space

$P[c_{min} \dots c_{max}, m_{min} \dots m_{max}]$

② For each point (x,y)

$m = min$, $m \le m_{max}$, $m{+}{+}$

$c = (-x)m + y$

$P(c,m) = P(c,m) + 1$

③ find local maxima in parameter space

$C_i = (-x)M + y$

Problems ① Slope → ∞
② Vertical lines m, C grow to ∞

⇒ Polar Co-ordinate

$\rho = x\cos\theta + y\sin\theta$

θ from gradient

Image gradient

(S_x / S_y) gradient vector

magnitude $= \sqrt{S_x^2 + S_y^2}$, direction $= \theta = \tan^{-1}\frac{S_y}{S_x}$

Line fitting examples → ideal, noisy, very noisy

noise factor → number of votes that the real line of 20 points gets with increasing noise

noise level ↑ ⇒ number of votes ↓

as noise increases in picture without a line, the number of points in the max cell goes up

Difficulties

what is the increment for θ, ρ?

too large → we can not distinguish between different lines

too small → noise cause line to be missed

Application Dependent

no clear consensus

depend on problem ← successive Hough, RANSAC

Circle fitting → 3 unknowns $(x-x_0)^2 + (y-y_0)^2 = r^2$

Construct 3D accumulator array (A).

x_0, y_0, r

- Fix one of parameter and loop for the others
- increment Corresponding entry (A)
- find local maxima in (A)
- use the tangent direction (θ) at the edge Point
- Compute x_0, y_0, r

$$x_0 = x - r\cos\theta$$
$$y_0 = y - r\sin\theta$$

Generalized hough transform x_c, y_c

used for shapes with no analytical expression

Require trainings
- object of known shape
- Generate model (R-table)

Similar approach to line, Circle fitting during detection

① Generating R-table
- Compute Centroid
- For each edge, Compute its distance to Centroid
- find edge orientation (gradient angle)
- Construct table of angles, r values

known
- ✓ edge point (x,y)
- ✓ Gradient angle with (θ)
- ✓ R-table of shape needs

For each edge point find (θ) Store it in Corresponding row R-table

→ Construct accumulator of 2d-Array

Hough transform

Rotation, Scale invariente

Rotation around Z axis

$\Rightarrow x' = x\cos\alpha - y\sin\alpha$

$y' = x\sin\alpha + y\cos\alpha$

Scaling

$\Rightarrow x' = sx$

$\Rightarrow y' = sy$

Rotation + Scaling

$x' = s(x\cos\alpha - y\sin\alpha)$

$y' = s(x\sin\alpha + y\cos\alpha)$

Vanishing point

Structurally input lines have the same vanishing point because they parallel in 3D

1. Horizontal, Vertical building edges
2. Zebra Crossing
3. Railway tracks
4. edges in tables, dresses
5. ubiquitous Calibration Pattern

It Can help → relative position of lines in images

→ intrinsic, extrinsic oreintation of Camera