CS-570 Computer Vision

Nazar Khan

Department of Computer Science
University of the Punjab

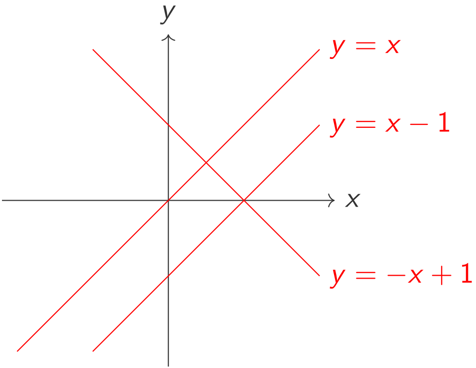
9. Hough Transform

The Hough Transform

- A powerful method for detecting curves from boundary information.
- Exploits the duality between points on a curve and parameters of the curve.
- Can detect analytic as well as non-analytic curves.

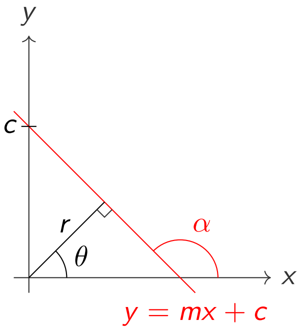
Analytic representation of a line

- In the analytic representation of a line $y = mx + c$, every choice of parameters (m, c) represents a different line.
- This is known as the *slope-intercept* parameter space.
- Weakness: vertical lines have $m = \infty$.



Polar representation of a line

- Solution: Polar representation (r, θ) where
  - r = perpendicular distance of line from origin
  - θ = angle of vector orthogonal to the line
- Every (r, θ) pair represents a 2D line.



$$
\begin{aligned}
y &= mx + c \\
m &= \tan(\alpha) = \tan(\theta + \frac{\pi}{2}) \\
&= \frac{\sin(\theta + \frac{\pi}{2})}{\cos(\theta + \frac{\pi}{2})} = \frac{\cos(\theta)}{-\sin(\theta)} \\
c &= \frac{r}{\sin(\theta)} \\
y &= -\frac{\cos(\theta)}{\sin(\theta)}x + \frac{r}{\sin(\theta)} \\
r &= x\cos(\theta) + y\sin(\theta)
\end{aligned}
$$

Hough Transform for Line Detection

- An algorithm for finding lines given some edge points.
- Given point (x, y), line passing through it with angle θ must have perpendicular $r = x\cos(\theta) + y\sin(\theta)$.
- Given any edge pixel (x, y), potentially 180 lines could pass through it assuming angular resolution of 1°.
- Looping through the angles gives (r, θ) pairs for all lines through (x, y).
- So pixel (x, y) should *vote for* all those lines.

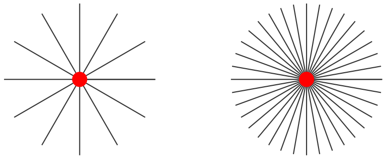


Figure: Lines passing through a point. **Left**: Angular resolution of 30°. **Right**: Angular resolution of 10°. Author: N. Khan (2021)

Hough Transform for Line Detection

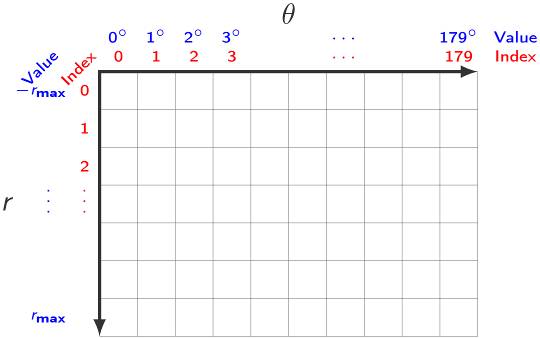


Figure: The accumulator array used to gather votes for each line. Each (r, θ) pair needs to be quantized into bin-indices before casting a vote. Author: N. Khan (2021).

Hough Transform for Line Detection

- By repeating this process for all edge pixels, actual lines will get a high number of votes.

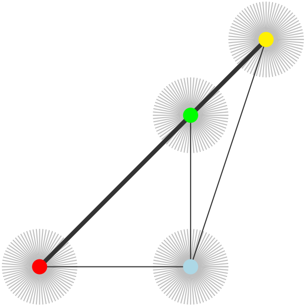


Figure: Each point votes for every line that passes through it. Genuine lines will get more votes. Author: N. Khan (2021)

Hough Transform for Line Detection
Pseudocode

initialize 2D (vote) accumulator array A to all zeros.
for every edge point (x, y)
    for $\theta = 0$ to π
        compute $r = x\cos(\theta) + y\sin(\theta)$
        compute indices $(r_{\text{ind}}, \theta_{\text{ind}})$ corresponding to (r, θ)
        increment $A(r_{\text{ind}}, \theta_{\text{ind}})$ by 1 $\longleftarrow$ vote of point (x, y) for line (r, θ)
valid lines are where $A >$ threshold

Hough Transform for Line Detection
Detailed Pseudocode

1. $\theta_{\text{range}} = 180°$
2. $\theta_{\text{binsize}} = 1°$ (for example)
3. $\theta_{\text{size}} = \left\lceil \frac{\theta_{\text{range}}}{\theta_{\text{binsize}}} \right\rceil$
4. $r_{\text{max}} =$ length of image diagonal
5. $r_{\text{range}} = 2r_{\text{max}}$
6. $r_{\text{binsize}} = 1$ pixel (for example)
7. $r_{\text{size}} = \left\lceil \frac{r_{\text{range}}}{r_{\text{binsize}}} \right\rceil$
8. initialize 2D (vote) accumulator array A of size $(r_{\text{size}}, \theta_{\text{size}})$ to all zeros.
9. for every edge point (x, y)
10. for $\theta = 0$ to θ_{range}
11. compute $r = x\cos(\theta) + y\sin(\theta)$
12. $r_{\text{ind}} = \text{round}\left(\frac{r + r_{\text{max}}}{r_{\text{binsize}}}\right)$
13. $\theta_{\text{ind}} = \text{round}\left(\frac{\theta \bmod 180}{\theta_{\text{binsize}}}\right)$
14. increment $A(r_{\text{ind}}, \theta_{\text{ind}})$ by 1 $\longleftarrow$ vote of point (x, y) for line (r, θ)

Hough Transform for Line Detection
Detailed Pseudocode

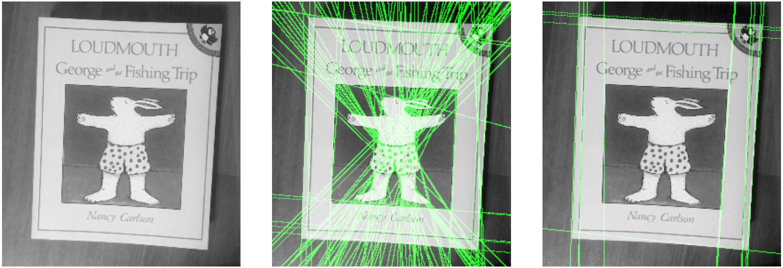
15. smooth votes via Gaussian convolution with kernel G_{σ_h} to account for uncertainties in the gradient direction
16. perform non-maxima suppression in $k \times k$ neighborhoods to remove fake lines around real ones
17. valid lines[1] are where $A > \tau$ which can be computed as a percentile

[1] Step 17 will yield indices of A. They will need to be converted back into (r, θ) values.

Improvement

- After edge detection, we already know the gradient direction at (x, y).
  - So there is no need to iterate over all possible θ.
  - Use the correct θ from the gradient direction.
  - This removes the loop at step 10.
  - Pixel (x, y) only votes for the line that was actually passing through it.
- This speeds-up the algorithm.
- This also avoids ghost lines.

Results



Original

Using edge pixels only

τ = 95-th percentile

Using edge pixels and

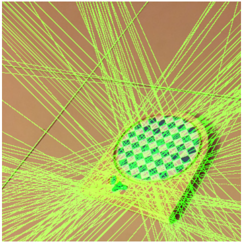
gradient orientations

τ = 70-th percentile

Figure: Line detection via Hough transform. Canny parameters: $\sigma_e = 1$, $t_h = 80$-th percentile, $t_l = 40$-th percentile. Hough parameters: $\sigma_h = \frac{\sigma_e}{5}$, $k = 3$. Author: N. Khan (2021)

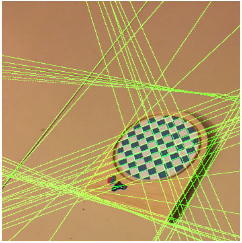
Results

Original



Using edge pixels only

τ = 95-th percentile



Using edge pixels and

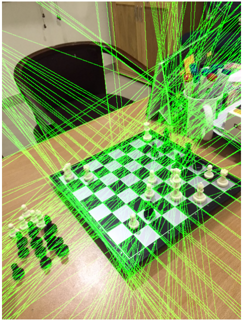
gradient orientations

τ = 70-th percentile

Figure: Line detection via Hough transform. Canny parameters: $\sigma_e = 1$, $t_h = 80$-th percentile, $t_l = 40$-th percentile. Hough parameters: $\sigma_h = \frac{\sigma_e}{5}$, $k = 3$. Author: N. Khan (2021)

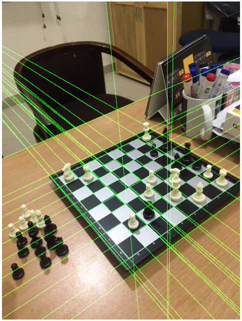
Results

Original



Using edge pixels only

$\tau = 95$-th percentile



Using edge pixels and

gradient orientations

$\tau = 90$-th percentile

Figure: Line detection via Hough transform. Canny parameters: $\sigma_e = 1$, $t_h = 80$-th percentile, $t_l = 40$-th percentile. Hough parameters: $\sigma_h = \frac{\sigma_e}{5}$, $k = 3$. Author: N. Khan (2021)

Hough Transform for Circle Detection

- Analytic representation of circle of radius r centered at (a, b) is $(x-a)^2 + (y-b)^2 - r^2 = 0$.
- Hough space has 3 parameters (a, b, r).

Pseudocode

For every boundary point (x, y)

    For every (a, b) in image plane

        Compute $r(a, b) = \sqrt{(x-a)^2 + (y-b)^2}$

        Compute $a_{\text{ind}}, b_{\text{ind}}$ and r_{ind}

        Increment $A(a_{\text{ind}}, b_{\text{ind}}, r_{\text{ind}})$ by 1

NMS$(A * G_{\sigma_h}) > \tau$ represents valid circles.

Hough Transform for Circle Detection

- If we know the gradient vector $\nabla I(x, y)$ at point (x, y), then we also know that the center (a, b) can only lie along this line.
- Hough space still has 3 parameters (a, b, r) but we search for r over a 1D space instead of a 2D plane.

Pseudocode

For every boundary point (x, y)

  For every (a, b) along gradient vector $\nabla I(x, y)$

    Compute $r(a, b) = \sqrt{(x-a)^2 + (y-b)^2}$

    Compute $a_{\text{ind}}, b_{\text{ind}}$ and r_{ind}

    Increment $A(a_{\text{ind}}, b_{\text{ind}}, r_{\text{ind}})$ by 1

NMS$(A * G_{\sigma_h}) > \tau$ represents valid circles.

Concluding Points

- Hough space becomes very large ($\text{param}_1 \times \text{param}_2 \times \cdots \times \text{param}_N$) when number of parameters N is increased.
- Using orientation information $\nabla I(x, y)$ in addition to positional information (x, y) leads to a smaller search space.
  - Speed-up
  - Fewer mistakes