Hough Transforms

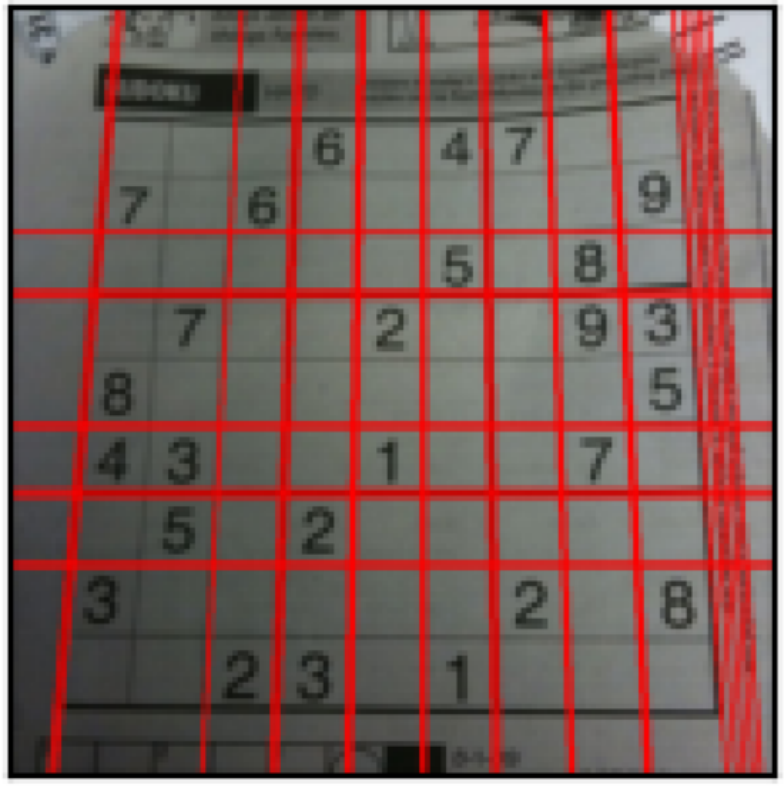
김수환

https://www.soohwan.kim

Edge Detection

https://towardsdatascience.com/canny-edge-detection-step-by-step-in-python-computer-vision-b49c3a2d8123

Line/Circle Detection?




학습목표

학습목표

1. Image Processing
    1. Image Thresholding
    2. Image Blending
    3. Image Filtering
    4. Morphological Transformations
    5. Image Gradients
    6. Hough Transforms

Hough Transforms

- Line Hough Transform
- Circle Hough Transform
- (RANSAC)

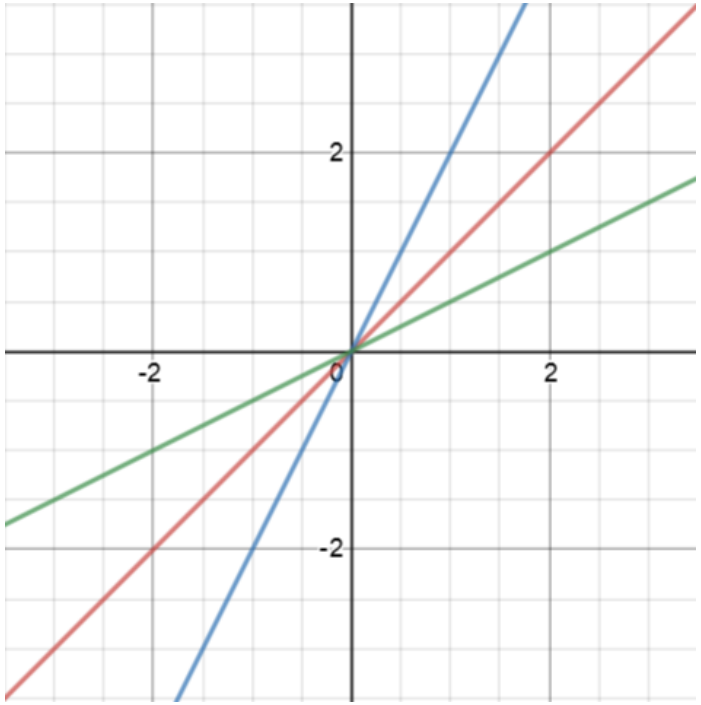
$$y = ax + b$$

$$y = ax$$

- 일차함수
- 그래프: 원점을 지나는 직선
- 기울기: a

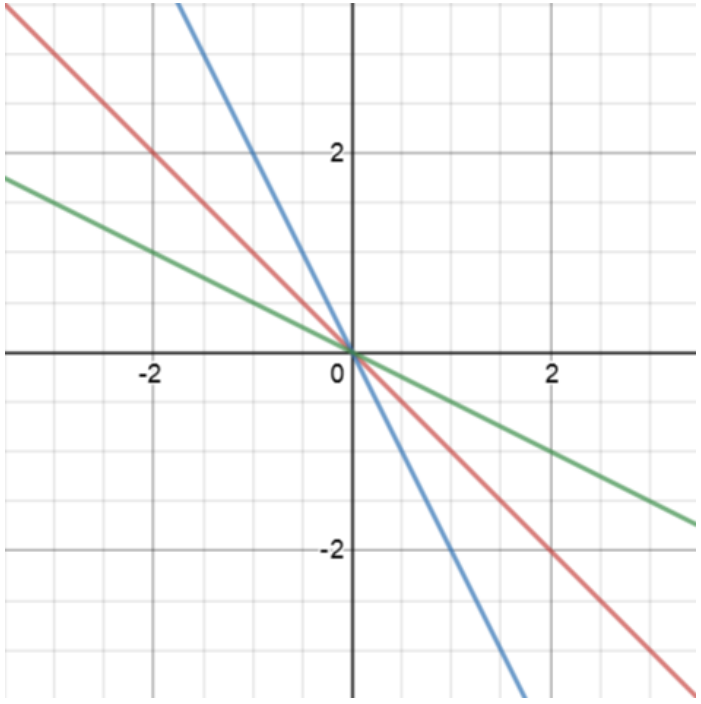
$y = ax$

$a > 0$



$y = 2x,\ \ y = x,\ \ y = \frac{1}{2}x$

$a < 0$



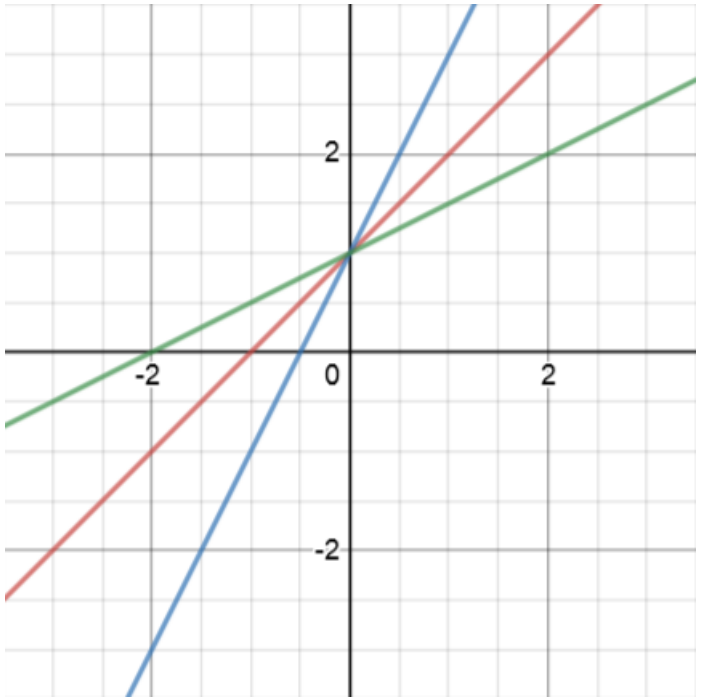
$y = -2x, y = -x, y = -\frac{1}{2}x$

$$y = ax + b$$

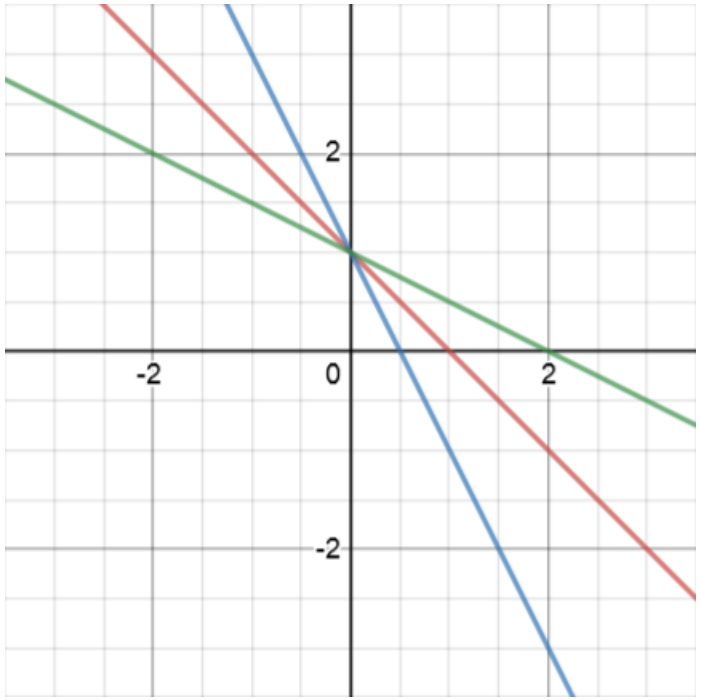
- 일차함수
- 그래프: y축 $(0, b)$를 지나는 직선
- 기울기: a
- y-절편: b
- $y = ax$의 그래프를 y축으로 b만큼 평행이동

$y = ax + b$

$a > 0$



$a < 0$



$y = 2x + 1$, $y = x + 1$, $y = \frac{1}{2}x + 1$

$y = -2x + 1$, $y = -x + 1$, $y = -\frac{1}{2}x + 1$

두 점을 지나는 직선

- 직선의 방정식

$$y = ax + b$$

- 직선 위의 두 점

$$(x_1, y_1),\ (x_2, y_2)$$

- 연립일차방정식

$$\begin{cases} y_1 = ax_1 + b \\ y_2 = ax_2 + b \end{cases}$$

- 해

$$\begin{cases} a = \frac{y_2 - y_1}{x_2 - x_1}, \quad x_1 \neq x_2 \\ b = y_1 - \frac{y_2 - y_1}{x_2 - x_1} x_1 \end{cases}$$

- 직선의 방정식

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1}(x - x_1)$$

두 점을 지나는 직선: 예

- 직선의 방정식

$$y = ax + b$$

- 직선 위의 두 점

$$(-1, 1),\ (3, 2)$$

- 연립일차방정식

$$\begin{cases} 1 = -a + b \\ 2 = 3a + b \end{cases}$$

- 해

$$\begin{cases} a = \frac{1}{4} \\ b = \frac{5}{4} \end{cases}$$

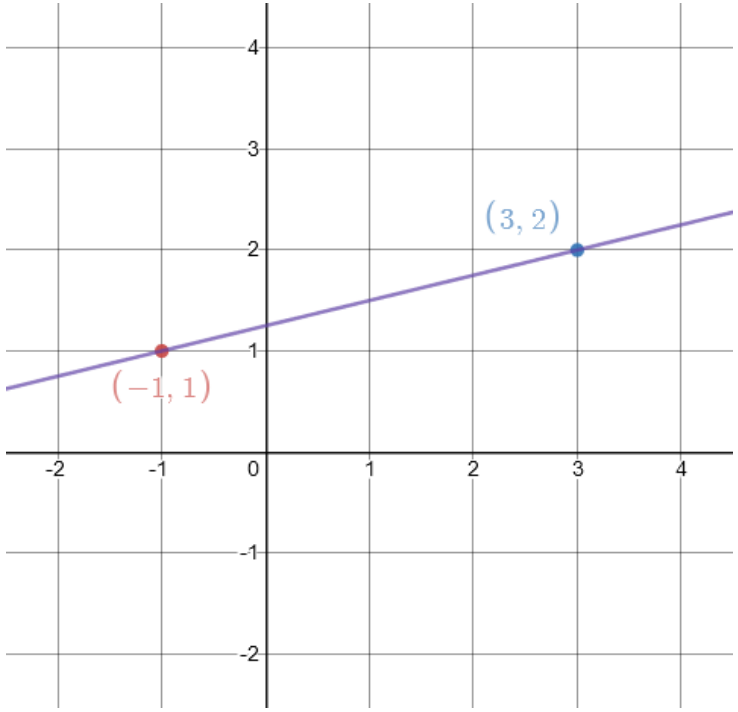
- 직선의 방정식

$$y - 1 = \frac{1}{4}(x + 1)$$

https://www.desmos.com/calculator/armsaifnvd?lang=ko

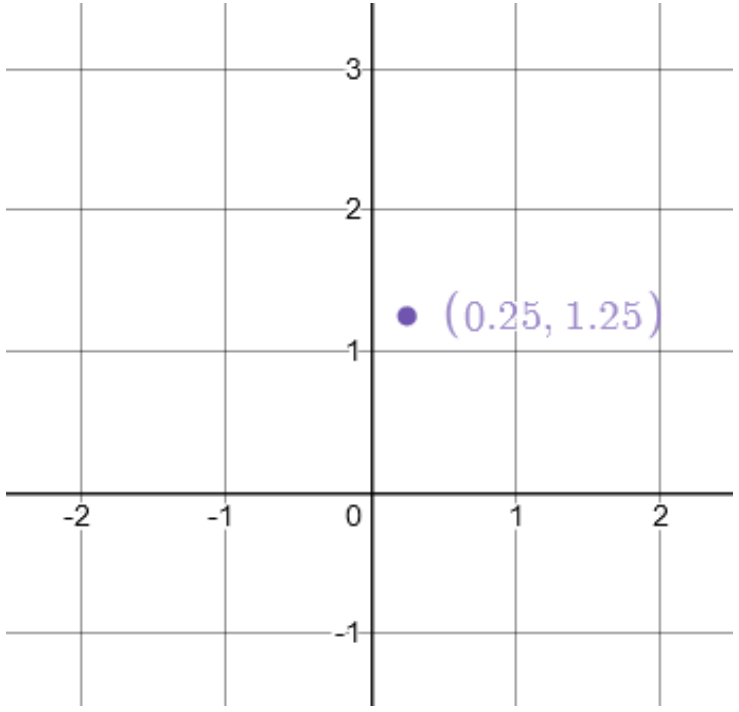
Image Space vs Hough Parameter Space

- 두 점을 지나는 직선



$$y - 1 = \frac{1}{4}(x + 1)$$

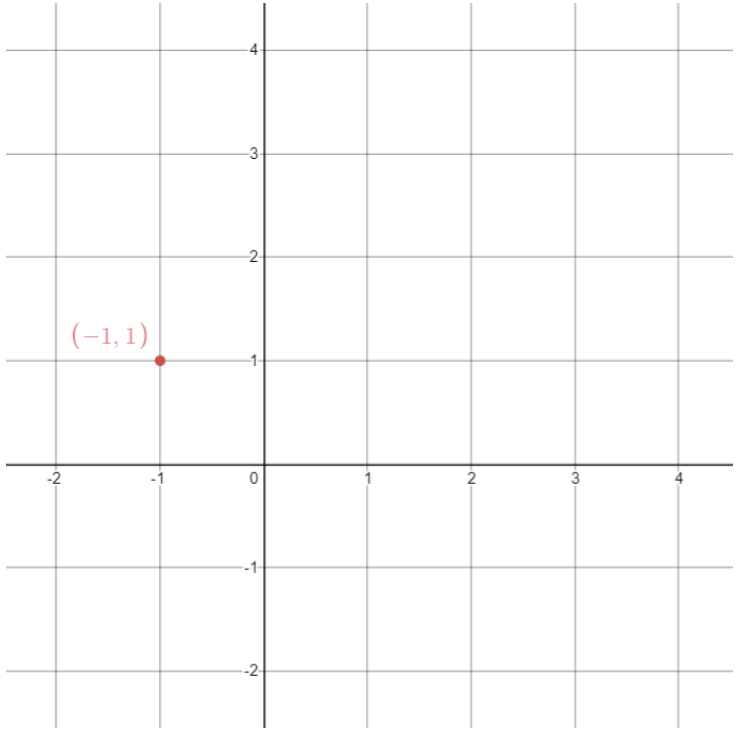
- 한 점 in Hough Parameter Space



$$a = \frac{1}{4}, b = \frac{5}{4}$$

한 점을 지나는 직선들

- 한 점



$$(x, y) = (-1, 1)$$

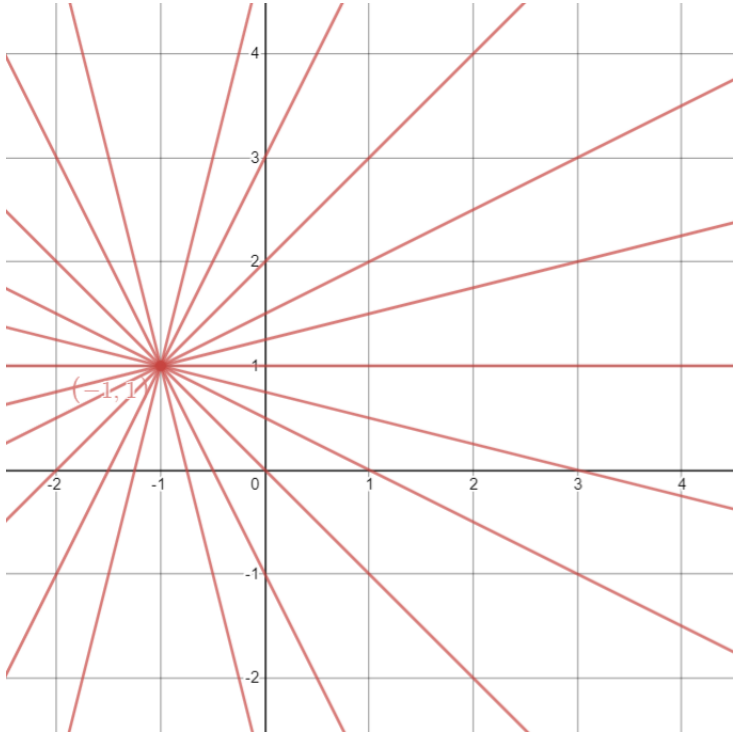
- 한 점을 지나는 직선들

$a > 0$	$a < 0$
$y = 4x + 5$	$y = -4x - 3$
$y = 2x + 3$	$y = -2x - 1$
$y = x + 2$	$y = -x$
$y = \frac{1}{2}x + \frac{3}{2}$	$y = -\frac{1}{2}x + \frac{1}{2}$
$y = \frac{1}{4}x + \frac{5}{4}$	$y = -\frac{1}{4}x + \frac{3}{4}$

$$1 = -a + b \;\Rightarrow\; b = a + 1$$

한 점을 지나는 직선들

- 한 점



$$(x, y) = (-1, 1)$$

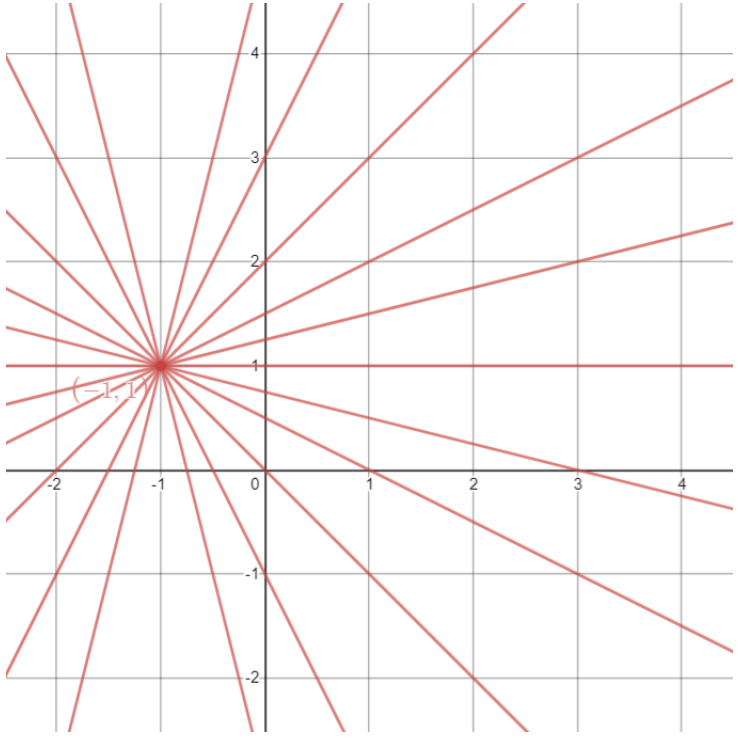
- 한 점을 지나는 직선들

$a > 0$	$a < 0$
$y = 4x + 5$	$y = -4x - 3$
$y = 2x + 3$	$y = -2x - 1$
$y = x + 2$	$y = -x$
$y = \frac{1}{2}x + \frac{3}{2}$	$y = -\frac{1}{2}x + \frac{1}{2}$
$y = \frac{1}{4}x + \frac{5}{4}$	$y = -\frac{1}{4}x + \frac{3}{4}$

$$1 = -a + b \;\Rightarrow\; b = a + 1$$

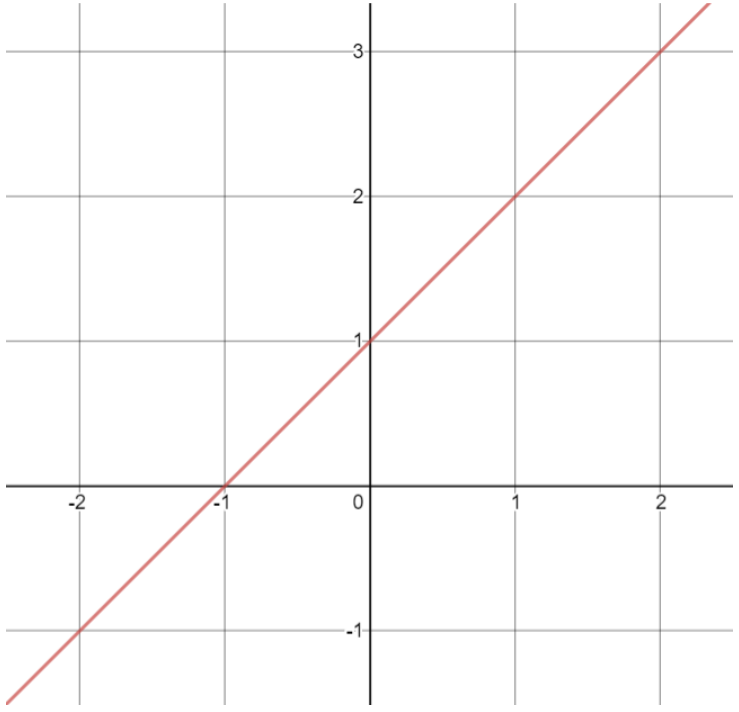
Image Space vs Hough Parameter Space

- 한 점을 지나는 직선들



$(x, y) = (-1, 1)$

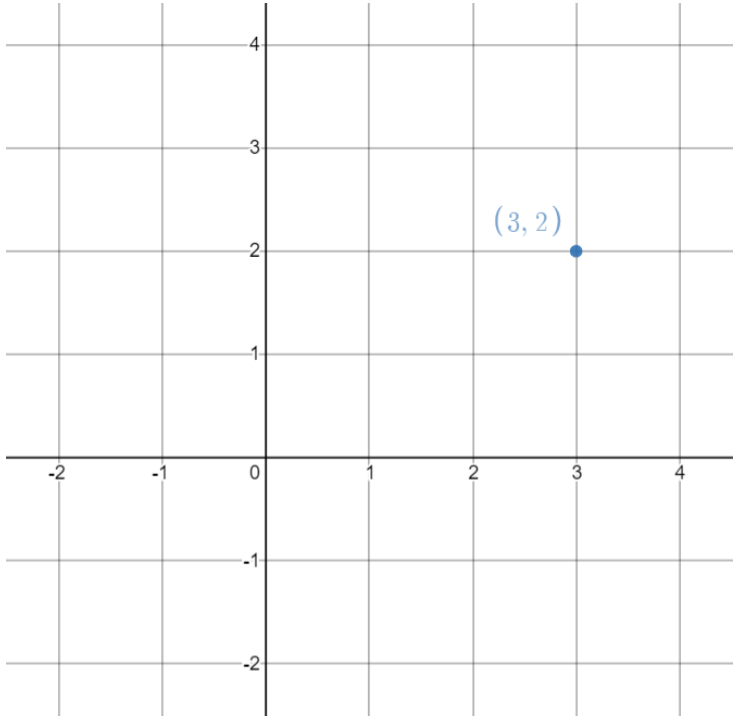
- 한 직선 in Hough Parameter Space



$b = a + 1$

또 다른 점을 지나는 직선들

- 또 다른 점



$$(x, y) = (3, 2)$$

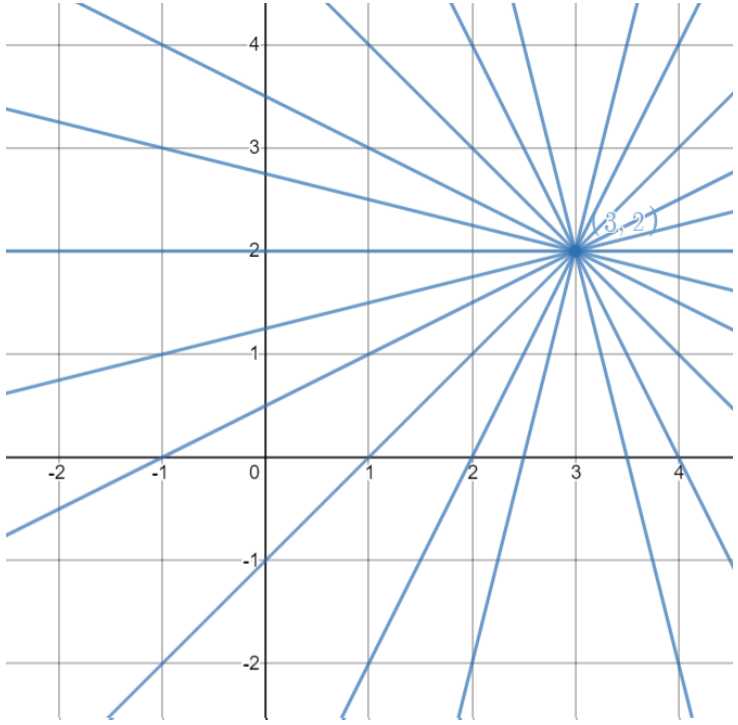
- 또 다른 점을 지나는 직선들

$a > 0$	$a < 0$
$y = 4x - 10$	$y = -4x + 14$
$y = 2x - 4$	$y = -2x + 8$
$y = x - 1$	$y = -x + 5$
$y = \frac{1}{2}x + \frac{1}{2}$	$y = -\frac{1}{2}x + \frac{7}{2}$
$y = \frac{1}{4}x + \frac{5}{4}$	$y = -\frac{1}{4}x + \frac{11}{4}$

$$2 = 3a + b \;\Rightarrow\; b = -3a + 2$$

또 다른 점을 지나는 직선들

- 또 다른 점



$(x, y) = (3, 2)$

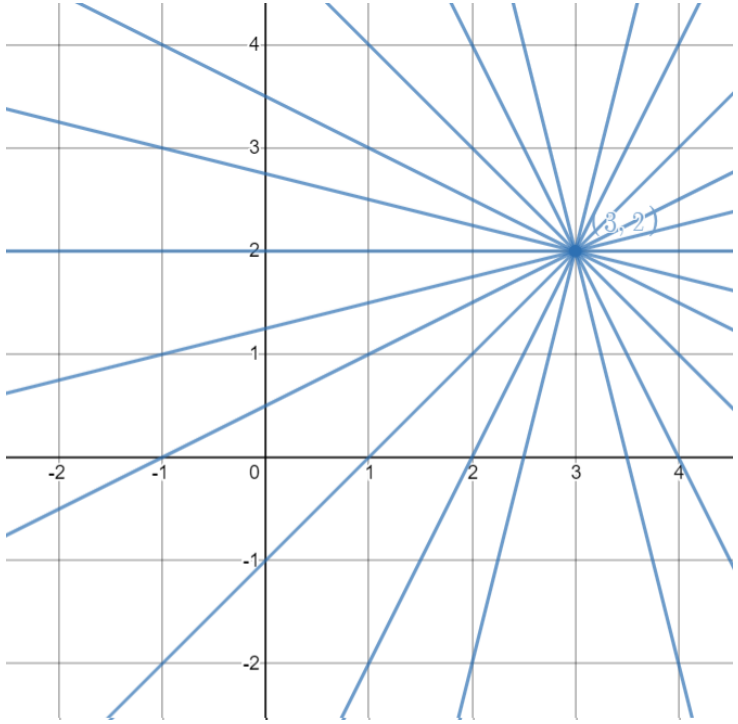
- 또 다른 점을 지나는 직선들

$a > 0$	$a < 0$
$y = 4x - 10$	$y = -4x + 14$
$y = 2x - 4$	$y = -2x + 8$
$y = x - 1$	$y = -x + 5$
$y = \frac{1}{2}x + \frac{1}{2}$	$y = -\frac{1}{2}x + \frac{7}{2}$
$y = \frac{1}{4}x + \frac{5}{4}$	$y = -\frac{1}{4}x + \frac{11}{4}$

$$2 = 3a + b \Rightarrow b = -3a + 2$$

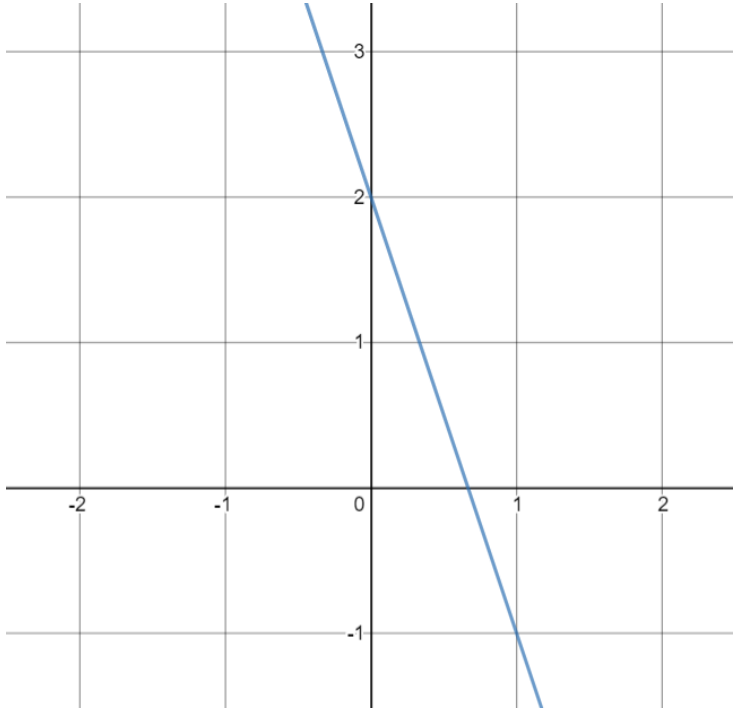
Image Space vs Hough Parameter Space

- 또 다른 점을 지나는 직선들



$(x, y) = (3, 2)$

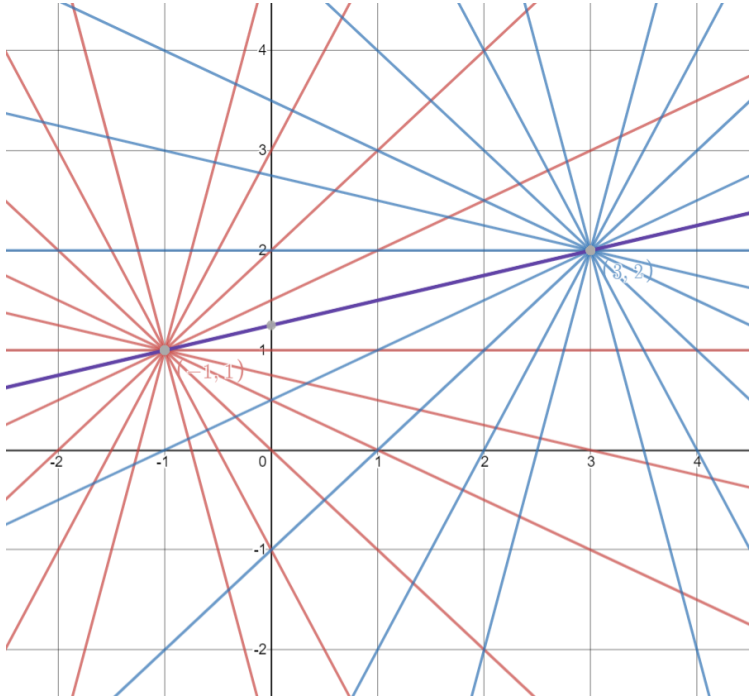
- 또 다른 직선 in Parameter Space



$b = -3a + 2$

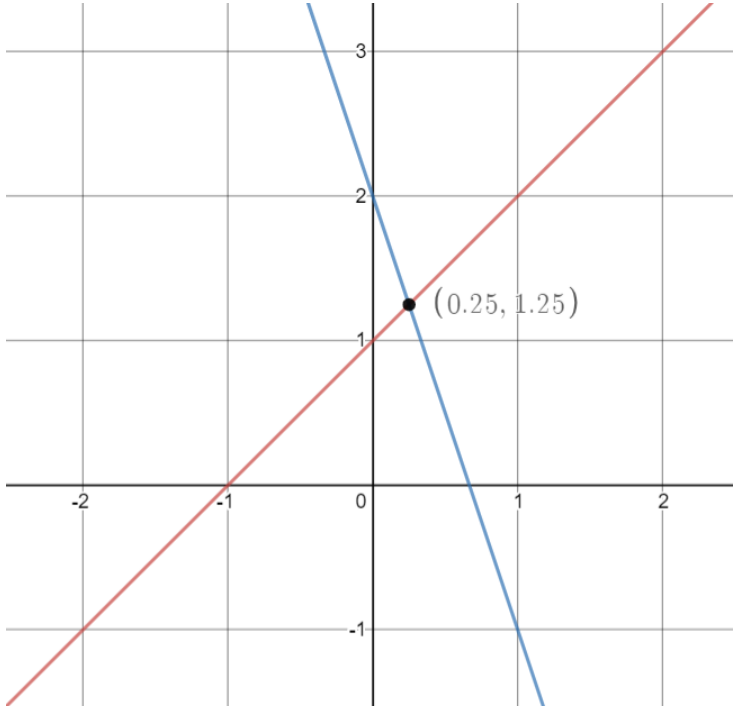
Image Space vs Hough Parameter Space

- 두 점을 각각 지나는 직선들



$(x, y) = (-1, 1),\ (3, 2)$

- 두 직선의 교점 in Parameter Space



$b = a + 1,\ \ b = -3a + 2$

두 점을 지나는 직선: Duality

- 직선의 방정식

$$y = ax + b$$

- 직선 위의 한 점

$$(-1, 1)$$

- 직선 in Hough Parameter Space

$$1 = -a + b \;\Rightarrow\; b = a + 1$$

- 직선의 방정식

$$y = ax + b$$

- 직선 위의 한 점

$$(3, 2)$$

- 직선 in Hough Parameter Space

$$2 = 3a + b \;\Rightarrow\; b = -3a + 2$$

- 두 직선의 교점 in Hough Parameter Space

$$\begin{cases} b = a + 1 \\ b = -3a + 2 \end{cases} \Rightarrow \begin{cases} a = \frac{1}{4} \\ b = \frac{5}{4} \end{cases}$$

두 점을 지나는 직선: Duality

- 직선의 방정식

$$y = ax + b$$

- 직선 위의 한 점

$$(x_1, y_1)$$

- 직선 in Hough Parameter Space

$$y_1 = ax_1 + b \;\Rightarrow\; b = -x_1 a + y_1$$

- 직선의 방정식

$$y = ax + b$$

- 직선 위의 한 점

$$(x_2, y_2)$$

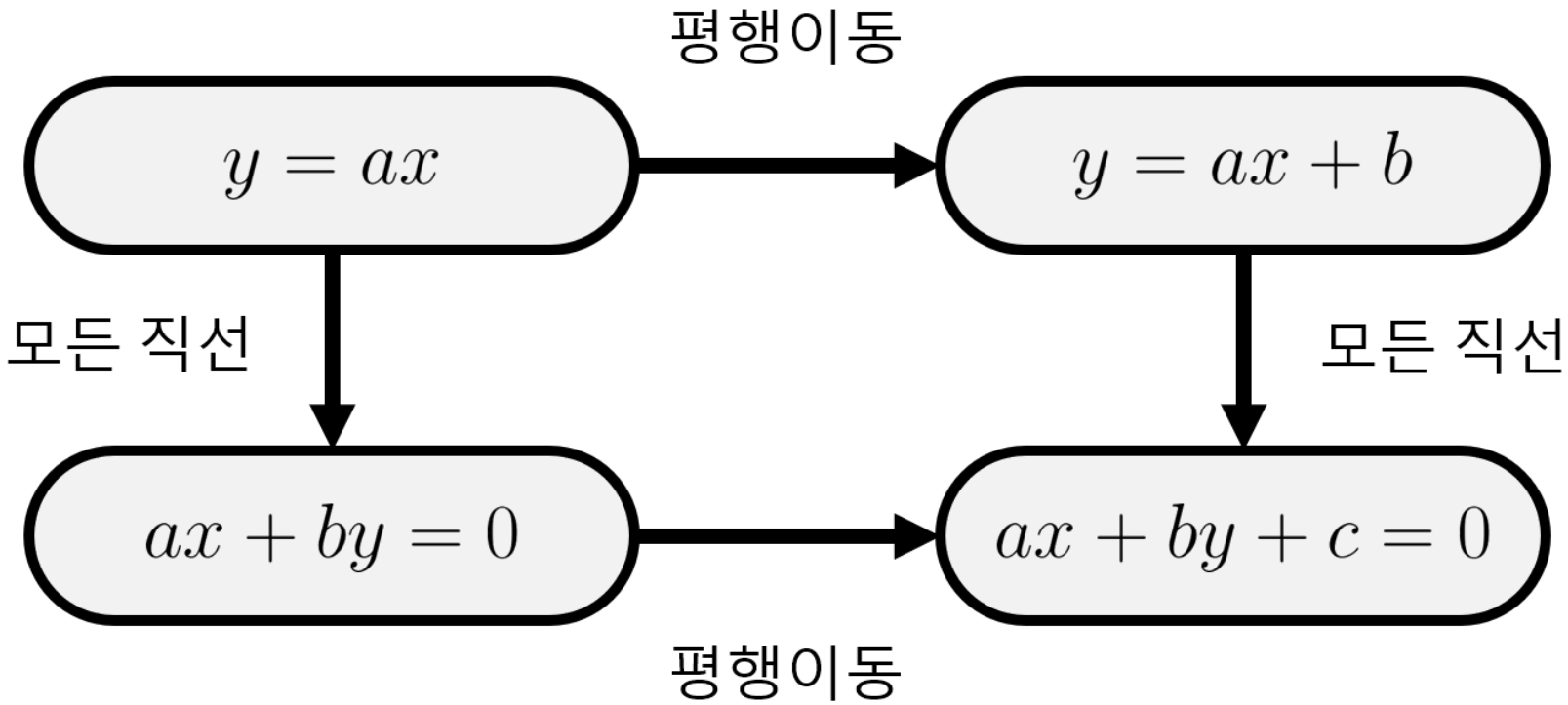
- 직선 in Hough Parameter Space

$$y_2 = ax_2 + b \;\Rightarrow\; b = -x_2 a + y_2$$

- 두 직선의 교점 in Hough Parameter Space

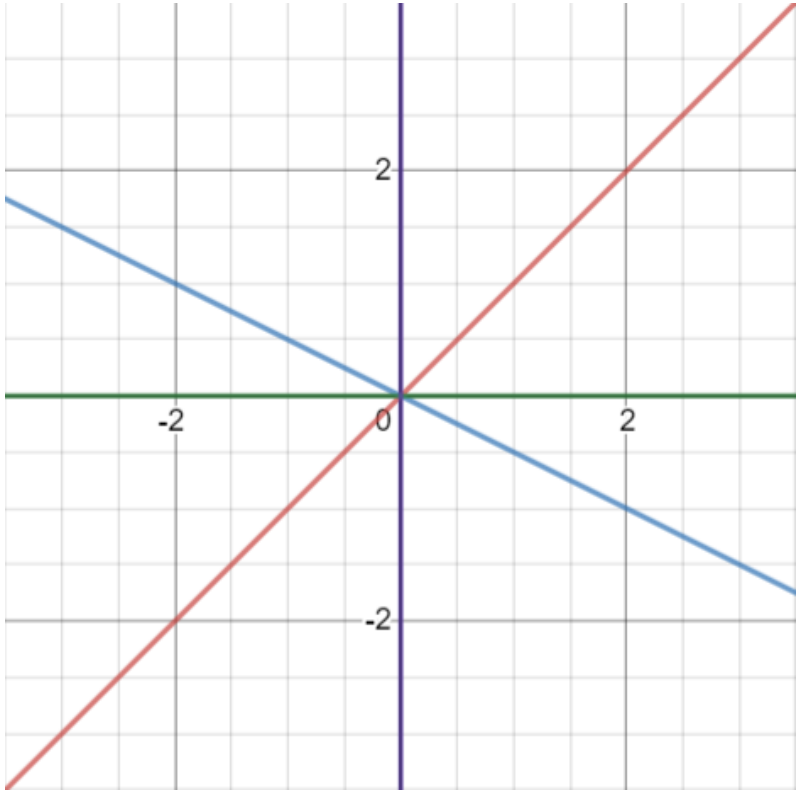
$$\begin{cases} b = -x_1 a + y_1 \\ b = -x_2 a + y_2 \end{cases} \Rightarrow \begin{cases} a = \frac{y_2 - y_1}{x_2 - x_1}, \quad x_1 \neq x_2 \\ b = y_1 - \frac{y_2 - y_1}{x_2 - x_1} x_1 \end{cases}$$

일반화



$$ax + by + c = 0$$

$$ax + by = 0$$



- 원점을 지나는 모든 직선의 방정식

$$x - y = 0$$
$$x + 2y = 0$$
$$y = 0$$
$$x = 0$$

$$ax + by + c = 0$$

- 모든 직선의 방정식

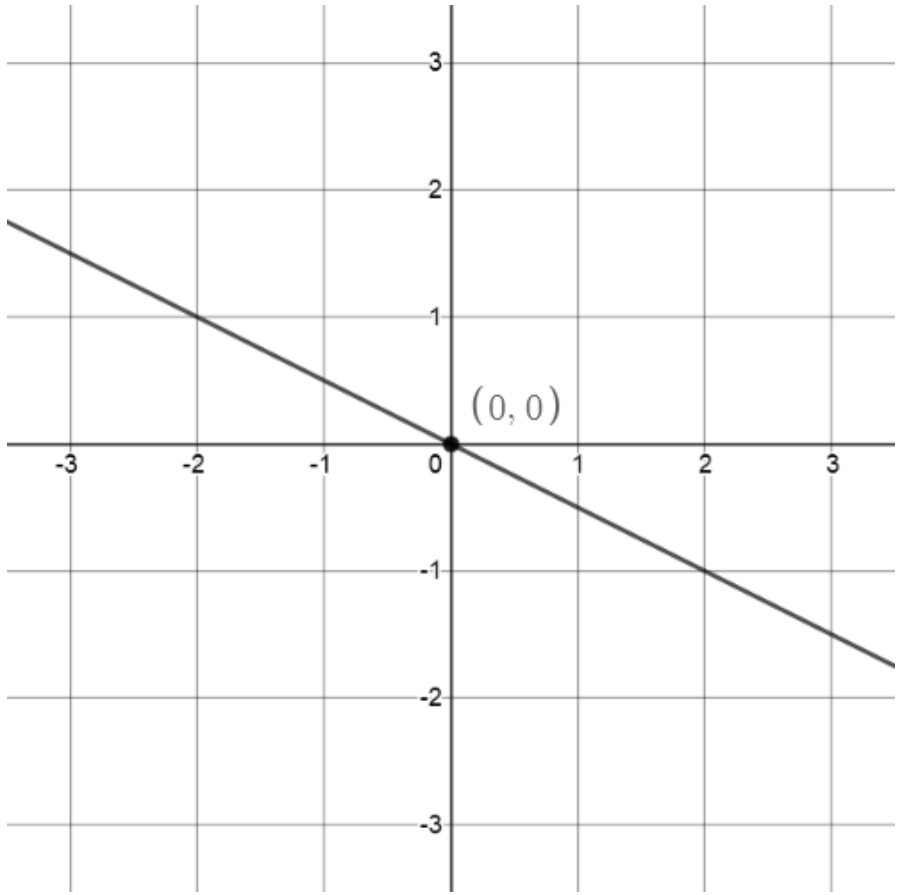
$$x - y + 2 = 0$$
$$x + y - 4 = 0$$
$$y - 1 = 0$$
$$x - 1 = 0$$

평행벡터와 법선벡터

원점을 지나는 직선의 평행벡터와 법선벡터

- 원점을 지나는 직선

$$ax + by = 0$$



$$x + 2y = 0$$

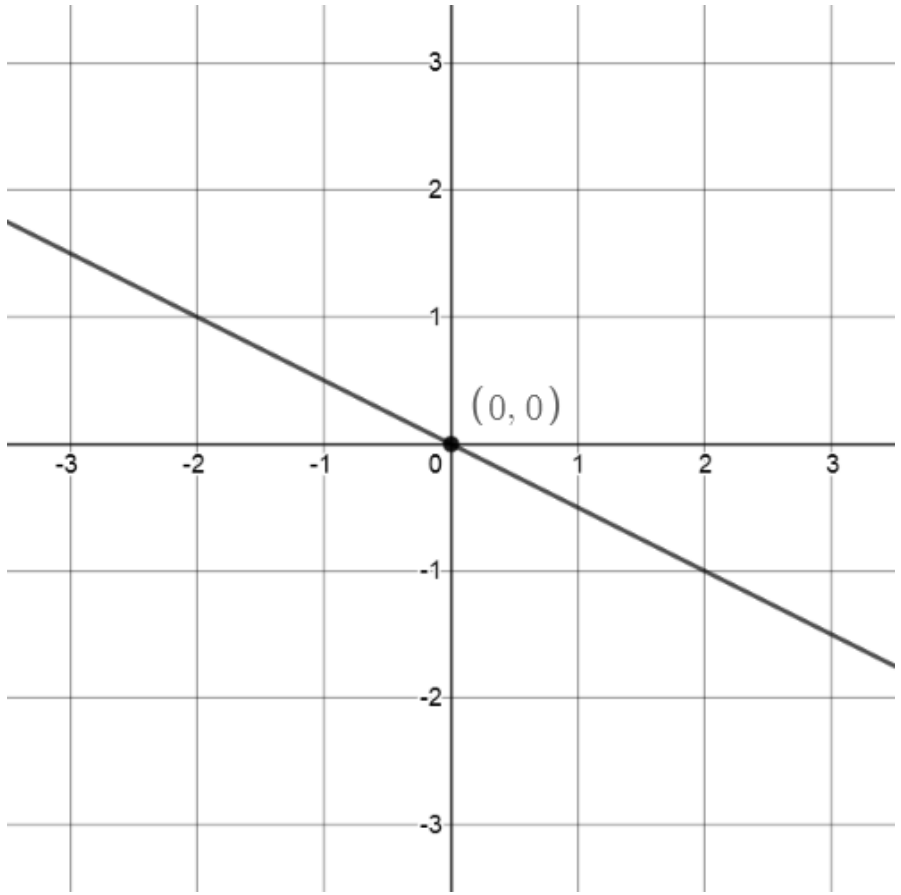
원점을 지나는 직선의 평행벡터와 법선벡터

- 원점을 지나는 직선

$$ax + by = 0$$

- 평행벡터

$$\mathbf{t} = ?$$



$$x + 2y = 0$$

원점을 지나는 직선의 평행벡터와 법선벡터

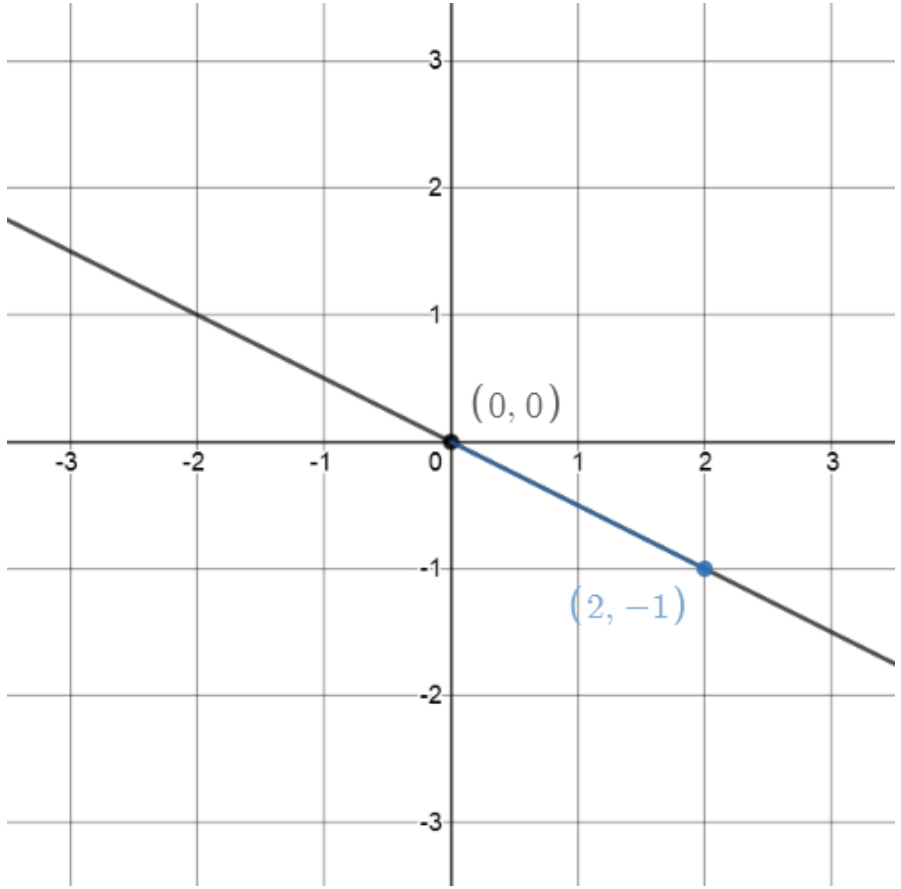
- 원점을 지나는 직선

$$ax + by = 0$$

- 대표 평행벡터

$$\mathbf{t} = (b, -a)^\top$$

$$\because a(b) + b(-a) = 0$$



$$x + 2y = 0 \;\; // \;\; (2, -1)^\top$$

원점을 지나는 직선의 평행벡터와 법선벡터

- 원점을 지나는 직선

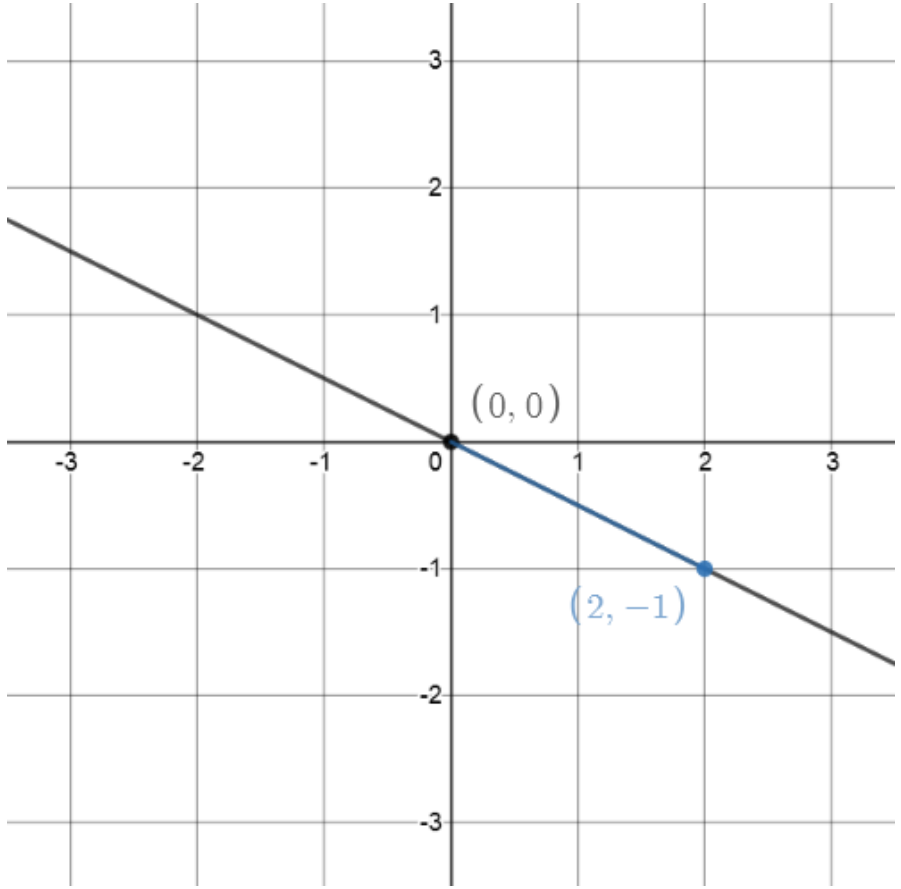
$$ax + by = 0$$

- 대표 평행벡터

$$\mathbf{t} = (b, -a)^\top$$

- 법선벡터

$$\mathbf{n} = ?$$



$$x + 2y = 0 \;\; // \;\; (2, -1)^\top$$

원점을 지나는 직선의 평행벡터와 법선벡터

- 원점을 지나는 직선

$$ax + by = 0$$

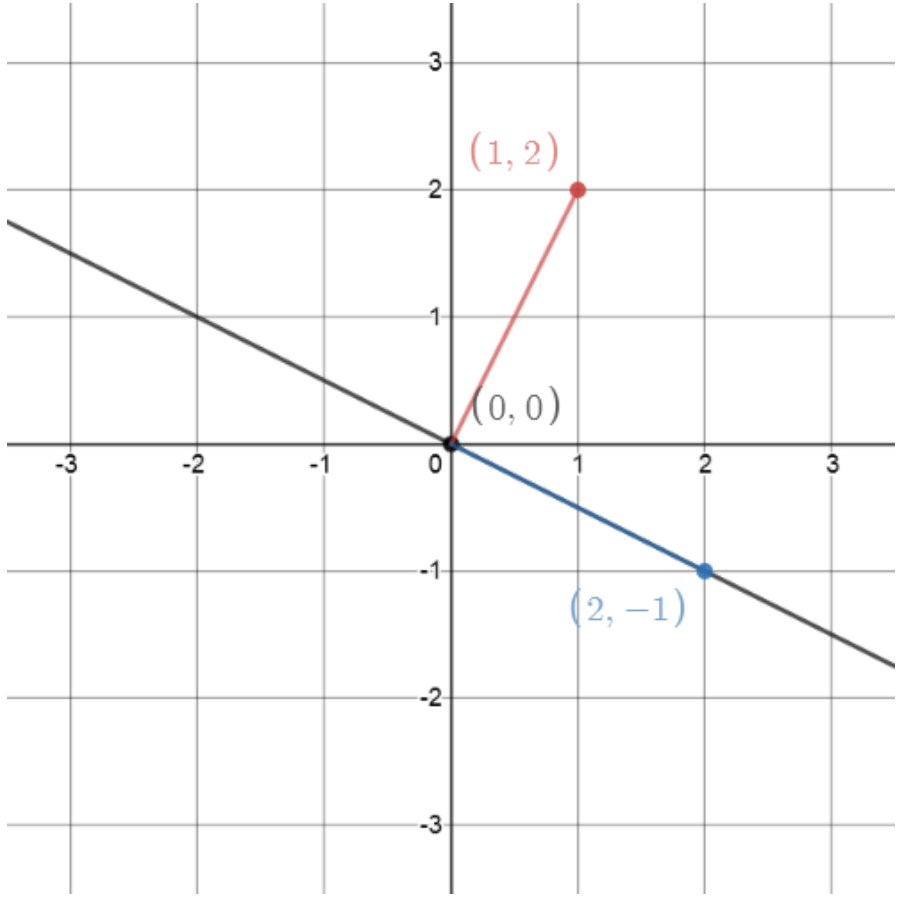
- 대표 평행벡터

$$\mathbf{t} = (b, -a)^\top$$

- 대표 법선벡터

$$\mathbf{n} = (a, b)^\top$$

$$\because \mathbf{n} \cdot \mathbf{x} = ax + by = 0$$



$$x + 2y = 0 \;\; /\!/ \;\; (2, -1)^\top \;\; \perp \;\; (1, 2)^\top$$

원점을 지나는 직선의 평행벡터와 법선벡터

- 원점을 지나는 직선

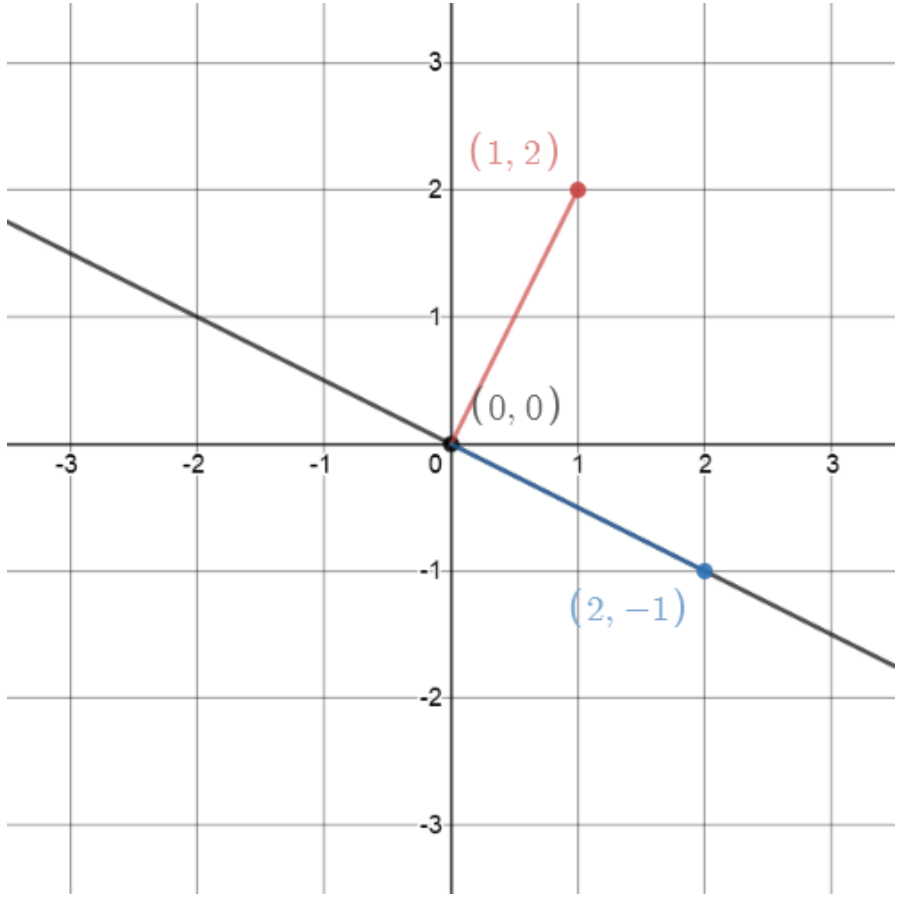
$$ax + by = 0$$

- 대표 평행벡터

$$\mathbf{t} = (b, -a)^\top$$

- 대표 법선벡터

$$\mathbf{n} = (a, b)^\top$$

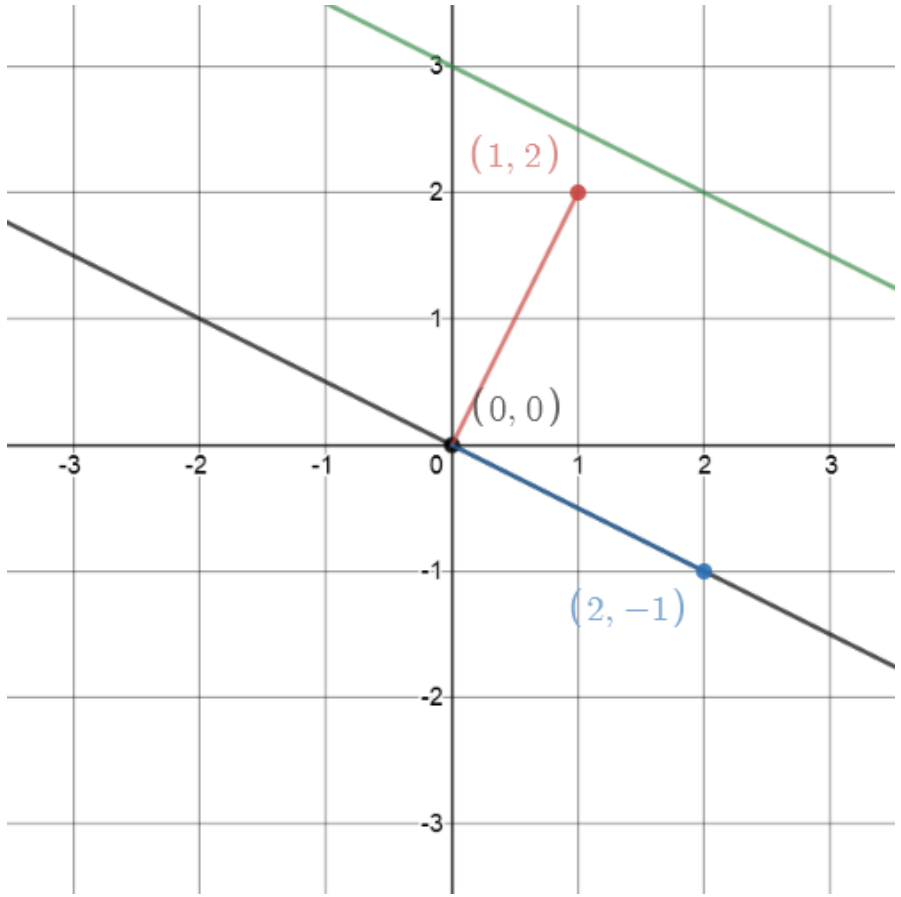


$$x + 2y = 0 \;\; /\!/ \;\; (2, -1)^\top \;\; \perp \;\; (1, 2)^\top$$

임의의 직선의 평행벡터와 법선벡터

- 임의의 직선

$$ax + by + c = 0$$



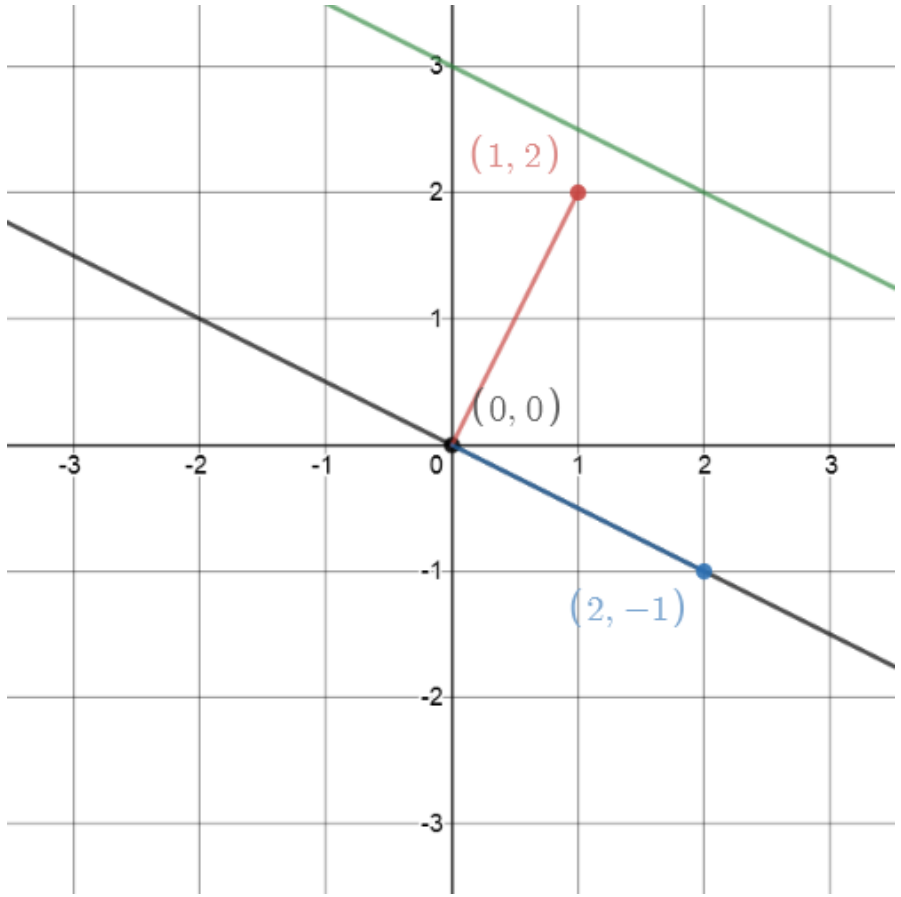
$$x + 2y = 0 \quad \Rightarrow \quad x + 2y - 6 = 0$$

임의의 직선의 평행벡터와 법선벡터

- 임의의 직선

$$ax + by + c = 0$$

$$ax + by + c = 0 \;\; // \;\; ax + by = 0$$



$$x + 2y - 6 = 0 \;\; // \;\; x + 2y = 0$$

임의의 직선의 평행벡터와 법선벡터

- 임의의 직선

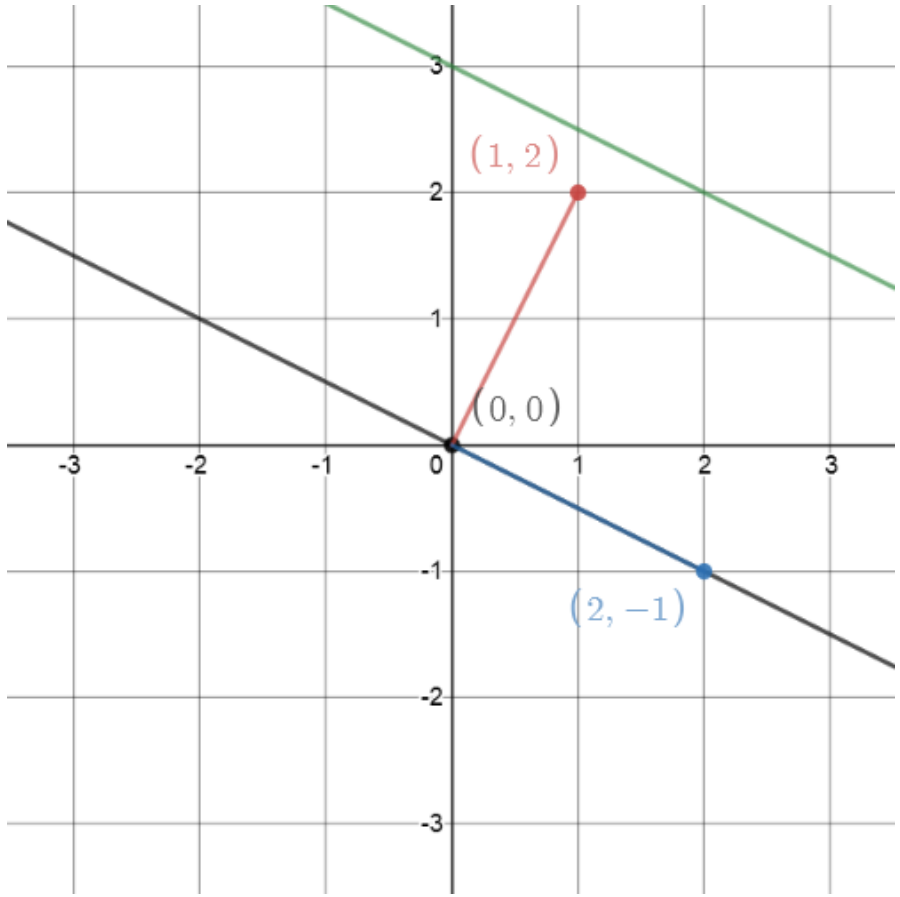
$$ax + by + c = 0$$

- 대표 평행벡터

$$\mathbf{t} = (b, -a)^\top$$

- 대표 법선벡터

$$\mathbf{n} = (a, b)^\top$$



$$x + 2y - 6 = 0 \;\; /\!/ \;\; (2, -1)^\top \;\; \perp \;\; (1, 2)^\top$$

임의의 직선의 평행벡터와 법선벡터

- 원점을 지나는 직선

$$ax + by = 0$$

- 대표 평행벡터

$$\mathbf{t} = (b, -a)^\top$$

- 대표 법선벡터

$$\mathbf{n} = (a, b)^\top$$

- 임의의 직선

$$ax + by + c = 0$$

- 대표 평행벡터

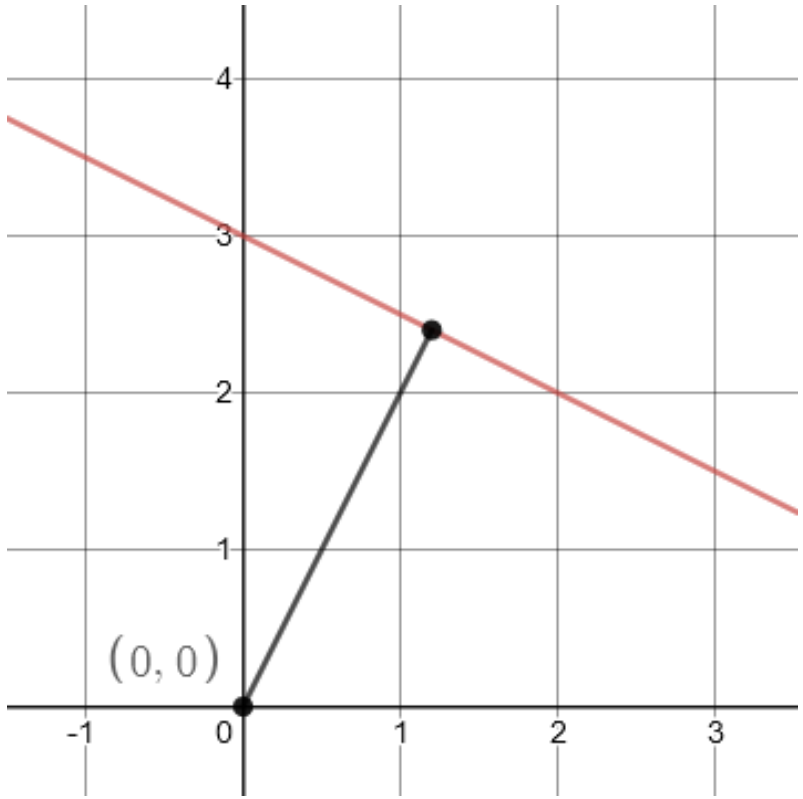
$$\mathbf{t} = (b, -a)^\top$$

- 대표 법선벡터

$$\mathbf{n} = (a, b)^\top$$

원점에서 직선까지의 거리

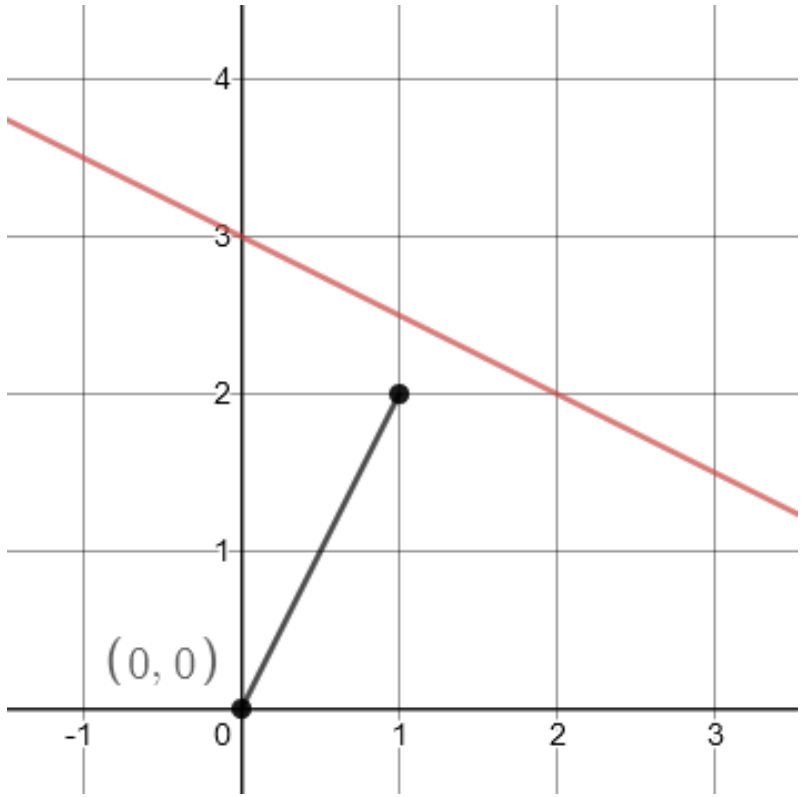
① 원점에서 직선까지의 거리



$ax + by + c = 0$

- 법선벡터?

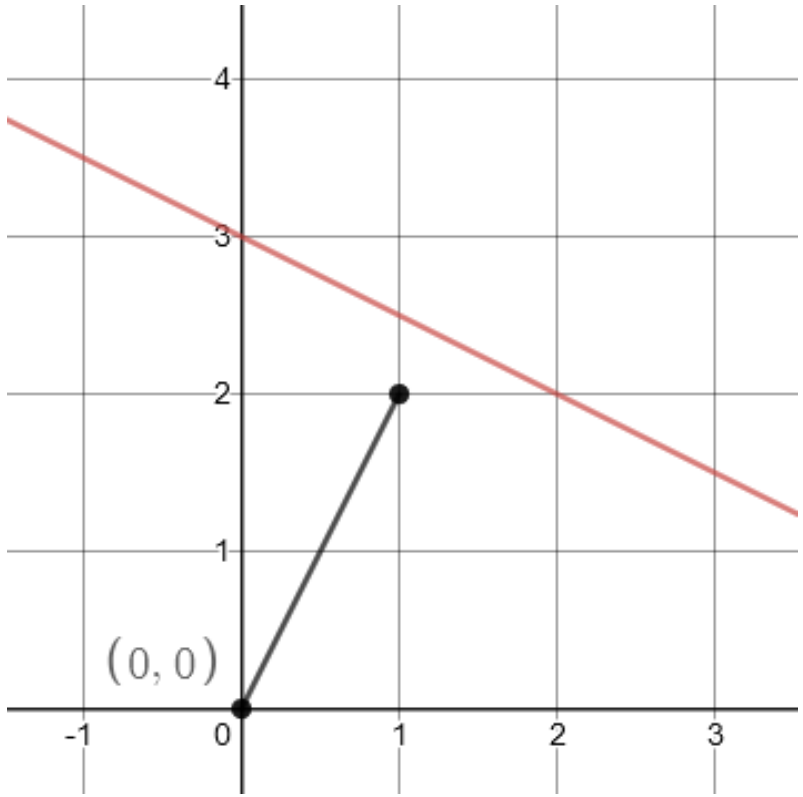
① 원점에서 직선까지의 거리



$ax + by + c = 0$

- 법선벡터: (a, b)

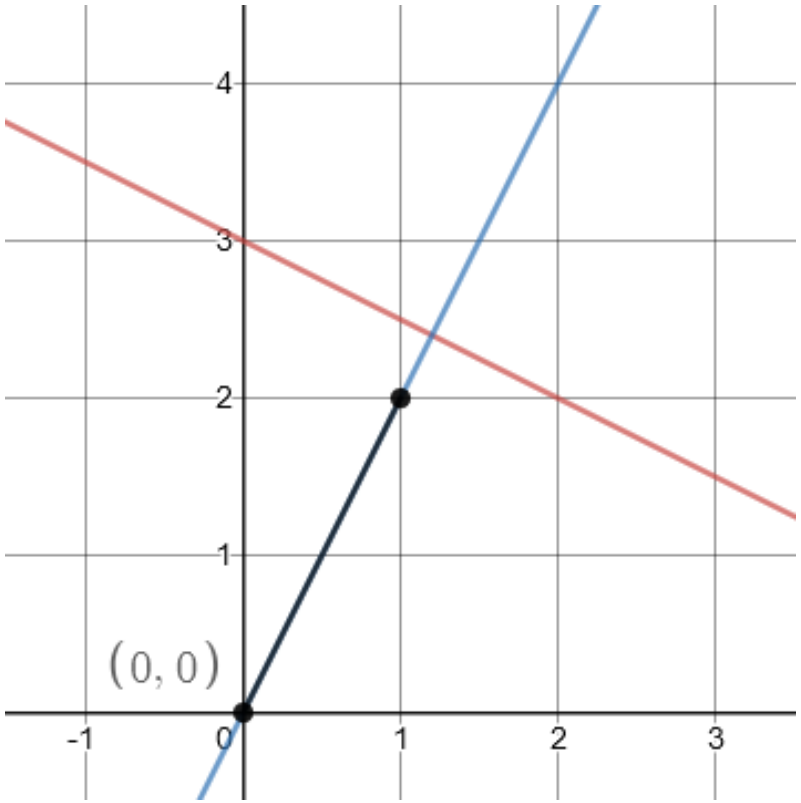
① 원점에서 직선까지의 거리



$ax + by + c = 0$

- 법선벡터: (a, b)
- 원점을 지나고 이 벡터에 평행한 직선?

① 원점에서 직선까지의 거리

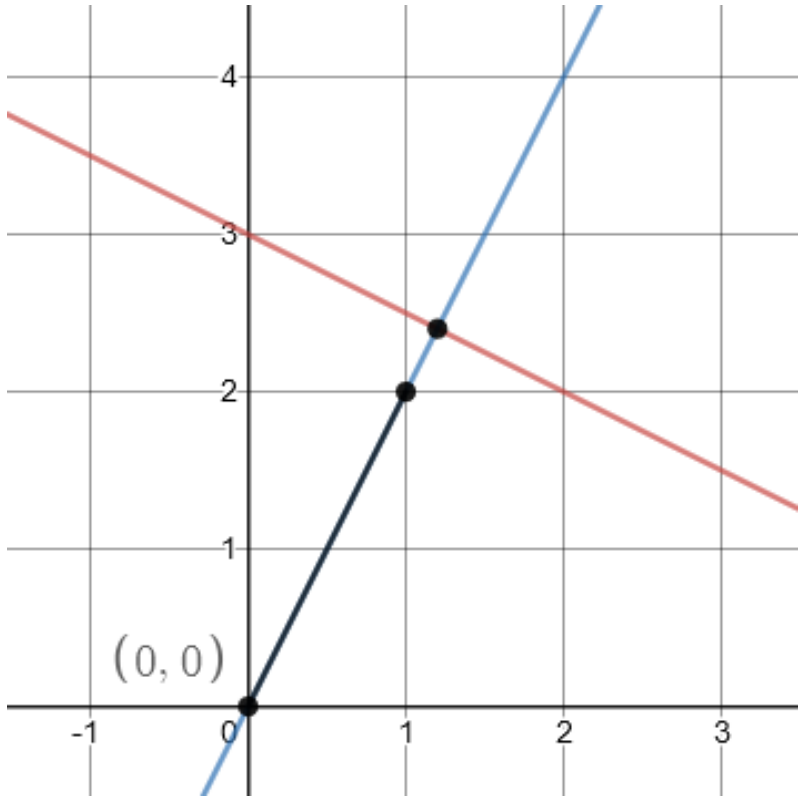


$ax + by + c = 0$

- 법선벡터: (a, b)
- 원점을 지나고 이 벡터에 평행한 직선

$$bx - ay = 0$$

① 원점에서 직선까지의 거리



$ax + by + c = 0$

- 법선벡터: (a, b)
- 원점을 지나고 이 벡터에 평행한 직선

$$bx - ay = 0$$

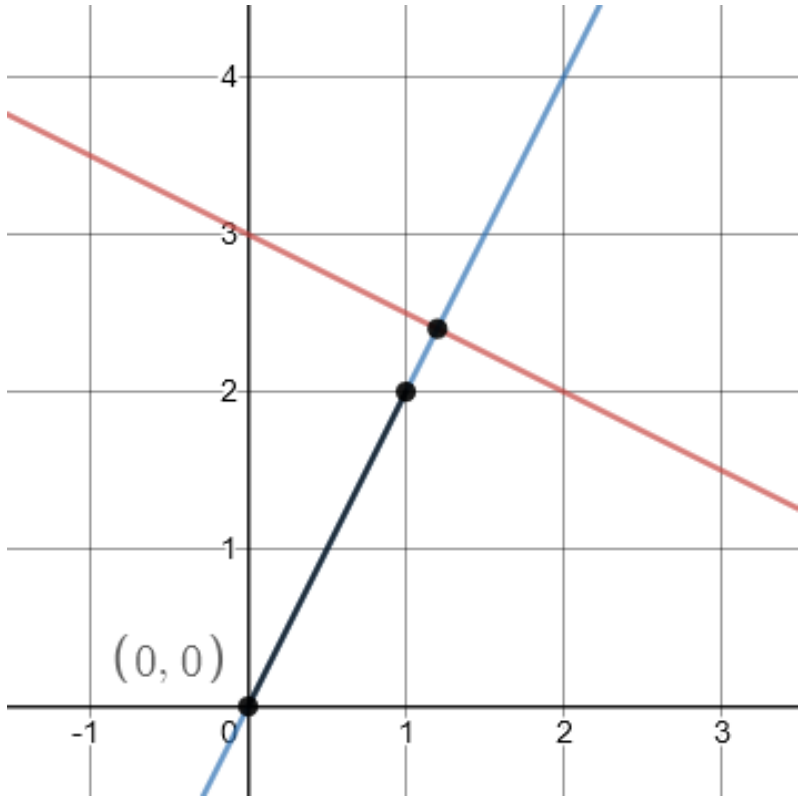
- 교점

$$\begin{cases} ax + by = -c \\ bx - ay = 0 \end{cases}$$

$$\Rightarrow \begin{cases} a^2x + aby = -ac \\ b^2x - aby = 0 \end{cases}$$

$$\Rightarrow \begin{cases} x = \frac{-ac}{a^2+b^2} \\ y = \frac{-bc}{a^2+b^2} \end{cases}$$

① 원점에서 직선까지의 거리



$ax + by + c = 0$

- 법선벡터: (a, b)
- 원점을 지나고 이 벡터에 평행한 직선

$$bx - ay = 0$$

- 교점

$$\begin{cases} ax + by = -c \\ bx - ay = 0 \end{cases}$$

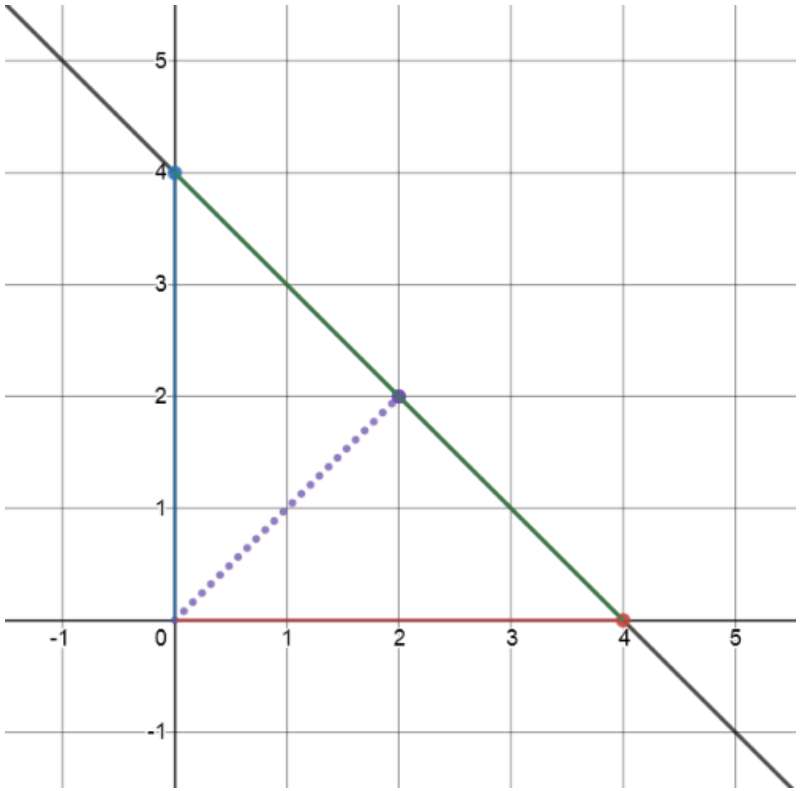
$$\Rightarrow \begin{cases} a^2x + aby = -ac \\ b^2x - aby = 0 \end{cases}$$

$$\Rightarrow \begin{cases} x = \frac{-ac}{a^2+b^2} \\ y = \frac{-bc}{a^2+b^2} \end{cases}$$

- 거리

$$d = \sqrt{\left(\frac{-ac}{a^2+b^2}\right)^2 + \left(\frac{-bc}{a^2+b^2}\right)^2} = \frac{|c|}{\sqrt{a^2+b^2}}$$

② 원점에서 직선까지의 거리



$ax + by + c = 0$

- x축 절편: $-\frac{c}{a} \Rightarrow$ 삼각형의 밑변: $w = \frac{|c|}{|a|}$
- y축 절편: $-\frac{c}{b} \Rightarrow$ 삼각형의 높이: $h = \frac{|c|}{|b|}$
- 삼각형의 대각선

$$l = \sqrt{\frac{c^2}{a^2} + \frac{c^2}{b^2}} = \sqrt{\frac{c^2(a^2+b^2)}{a^2b^2}} = \frac{|c|\sqrt{a^2+b^2}}{|a||b|}$$

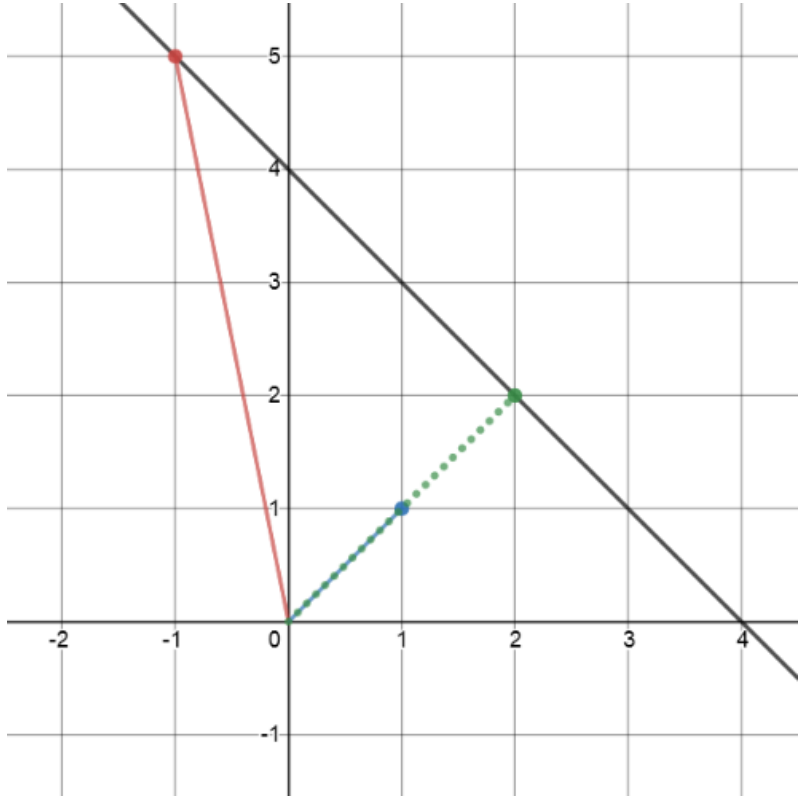
- 삼각형의 넓이

$$s = \frac{1}{2}wh = \frac{1}{2}dl$$

$$\Rightarrow \frac{|c|}{|a|}\frac{|c|}{|b|} = d\frac{|c|\sqrt{a^2+b^2}}{|a||b|}$$

$$\Rightarrow d = \frac{|c|}{\sqrt{a^2+b^2}}$$

③ 원점에서 직선까지의 거리



$ax + by + c = 0, \quad \text{or} \quad \mathbf{n} \cdot \mathbf{x} = -c$

- Projection Vector의 크기

$$\|\mathbf{x}_{\|}\| = \mathbf{x} \cdot \hat{\mathbf{n}} = \frac{\mathbf{x} \cdot \mathbf{n}}{\|\mathbf{n}\|}$$

- 원점에서 직선까지의 거리 ($d > 0$)

$$\Rightarrow \; d = \frac{|\mathbf{x} \cdot \mathbf{n}|}{\|\mathbf{n}\|} = \frac{|c|}{\sqrt{a^2 + b^2}}$$

Hough Transforms

직선의 방정식

- 일차함수

$$y = ax + b$$

- 단점: 기울기가 무한대인 직선을 표현할 수 없다.

$$x = c$$

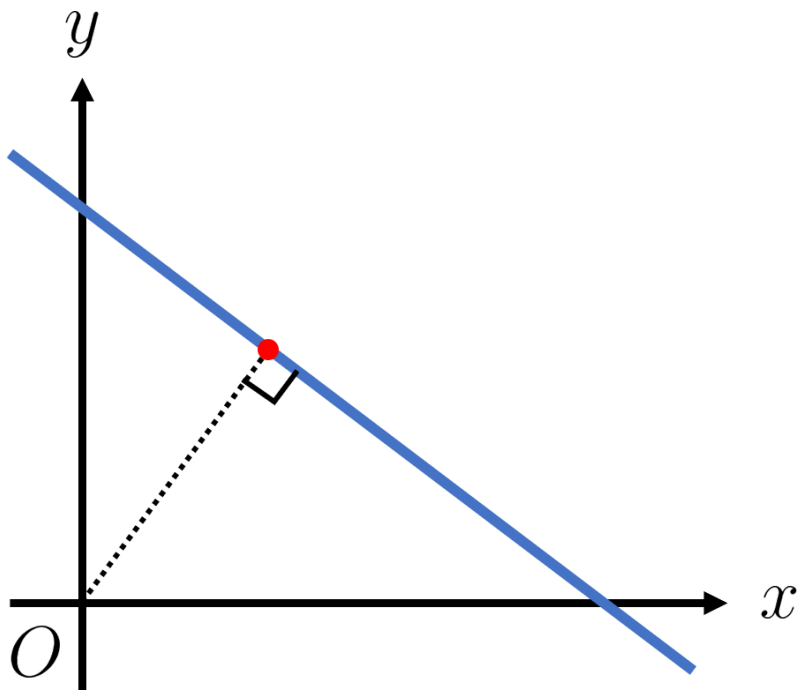
- 대안: 일차방정식

$$ax + by + c = 0$$

- 단점: Parameter의 개수가 3개다.

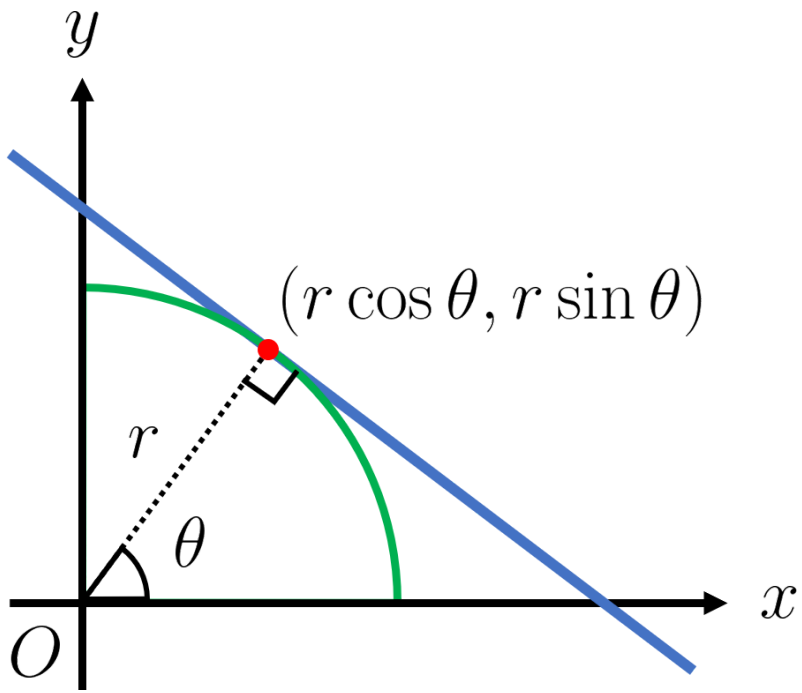
극좌표계를 이용한 직선의 방정식

- 한 점을 지나고
  원 점에서의 수선의 발이 그 점인
  직선의 방정식



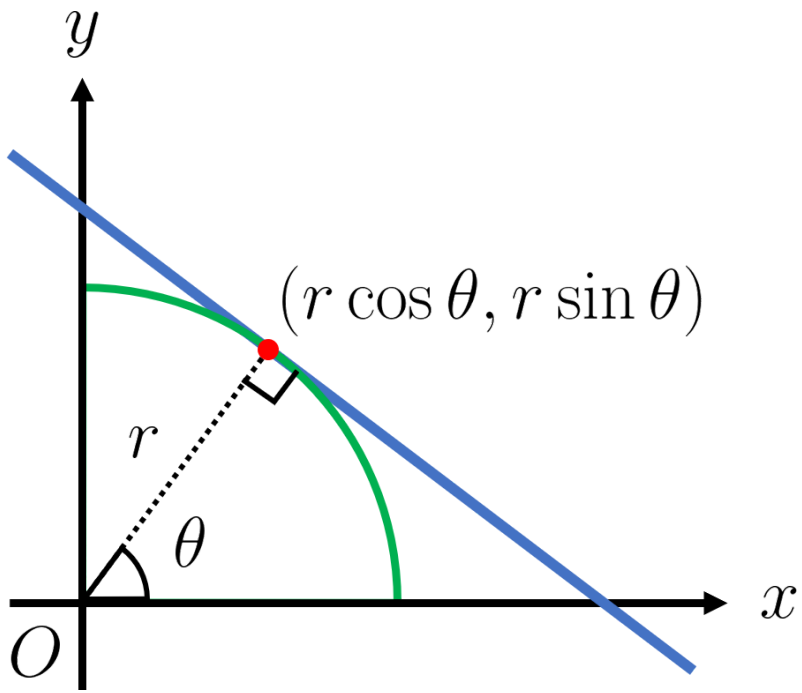
극좌표계를 이용한 직선의 방정식

- 지나는 점: $(r\cos\theta, r\sin\theta)$



극좌표계를 이용한 직선의 방정식

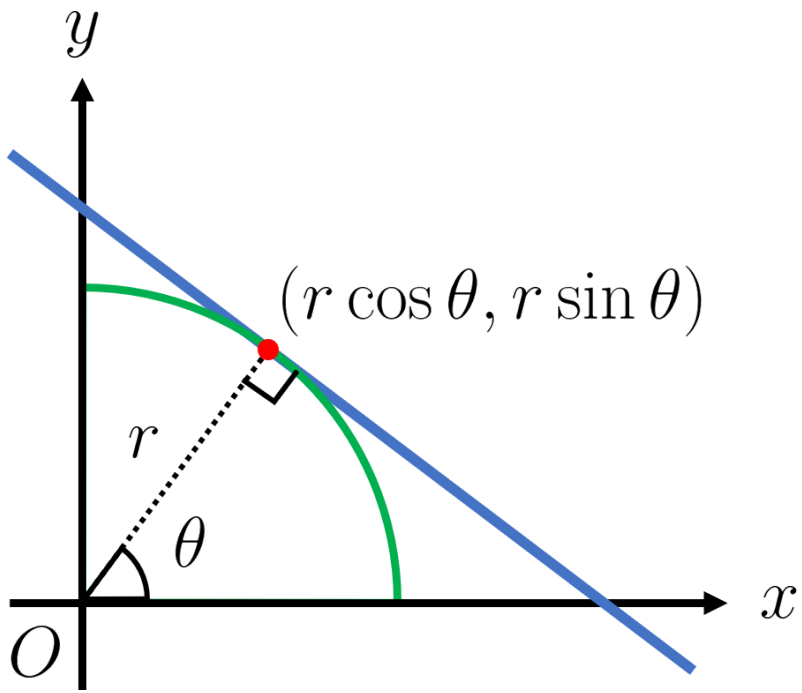
- 지나는 점: $(r\cos\theta, r\sin\theta)$
- 법선벡터: $(\cos\theta, \sin\theta)$



극좌표계를 이용한 직선의 방정식

- 지나는 점: $(r\cos\theta, r\sin\theta)$
- 법선벡터: $(\cos\theta, \sin\theta)$
- 직선의 방정식

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

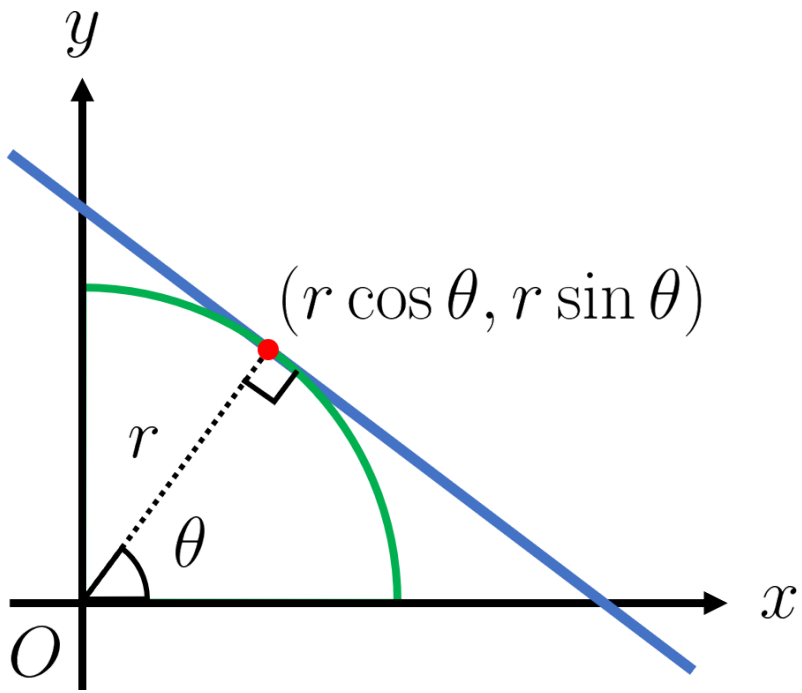


극좌표계를 이용한 직선의 방정식

- 지나는 점: $(r\cos\theta, r\sin\theta)$
- 법선벡터: $(\cos\theta, \sin\theta)$
- 직선의 방정식

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

- 원 점에서의 거리: r

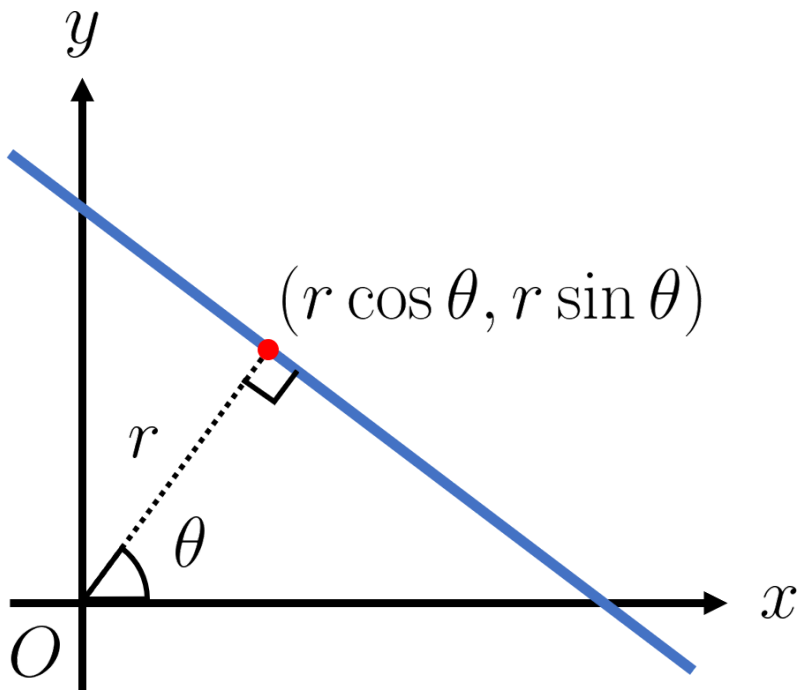


극좌표계를 이용한 직선의 방정식

- 직선의 방정식

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

- 장점: Parameter의 개수: 2
  - r: 원점에서 직선까지의 거리
  - θ: x축과 수선의 발 사이의 각도
- 장점: Parameter가 유한하다.
  - $r \in [0, R]$: 이미지의 대각선 길이
  - $\theta \in [0, 2\pi]$: 회전각

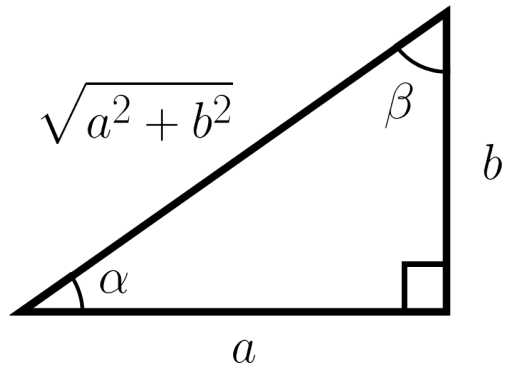


sin과 cos의 합

- 삼각함수의 합차공식

$$\sin(\alpha \pm \beta) = \sin\alpha\cos\beta \pm \cos\alpha\sin\beta$$

$$\cos(\alpha \pm \beta) = \cos\alpha\cos\beta \mp \sin\alpha\sin\beta$$



- sin과 cos의 합

$$\begin{aligned}
&a\cos\theta + b\sin\theta \\
&= \sqrt{a^2+b^2}\left(\frac{a}{\sqrt{a^2+b^2}}\cos\theta + \frac{b}{\sqrt{a^2+b^2}}\sin\theta\right) \\
&= \sqrt{a^2+b^2}(\cos\theta\cos\alpha + \sin\theta\sin\alpha) \\
&= \sqrt{a^2+b^2}\cos(\theta-\alpha)
\end{aligned}$$

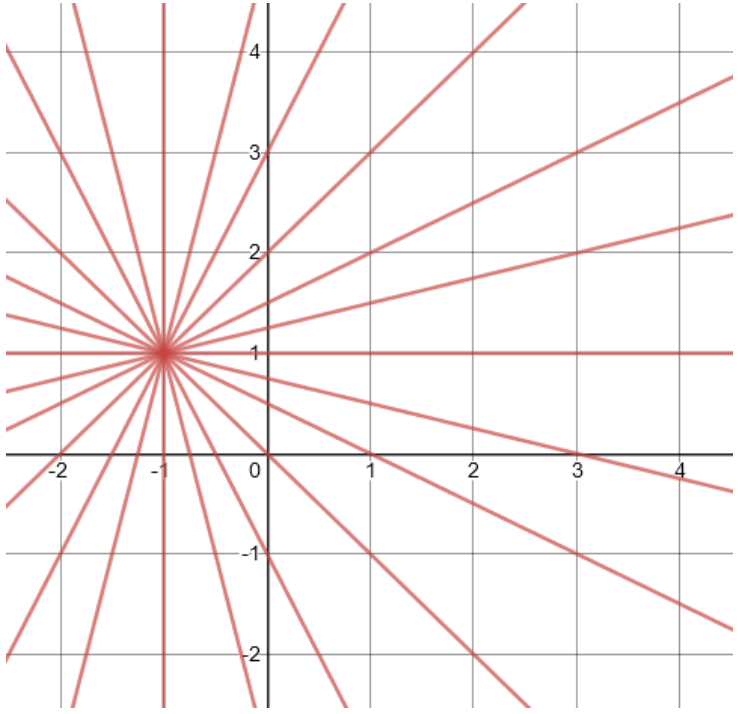
- sin과 cos의 합

$$\begin{aligned}
&a\cos\theta + b\sin\theta \\
&= \sqrt{a^2+b^2}\left(\frac{a}{\sqrt{a^2+b^2}}\cos\theta + \frac{b}{\sqrt{a^2+b^2}}\sin\theta\right) \\
&= \sqrt{a^2+b^2}(\cos\theta\sin\beta + \sin\theta\cos\beta) \\
&= \sqrt{a^2+b^2}\sin(\theta+\beta)
\end{aligned}$$

https://namu.wiki/w/삼각함수의 합차공식

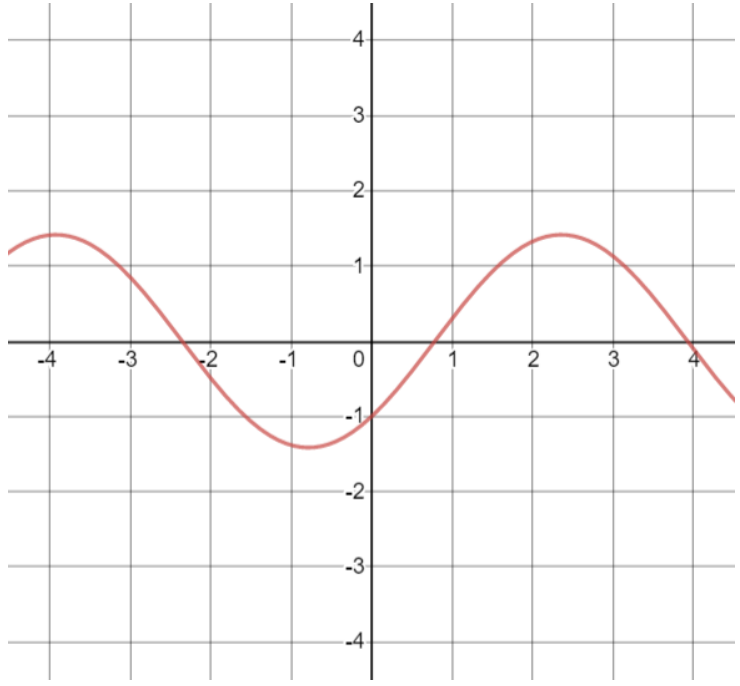
Image Space vs Hough Parameter Space

- 한 점을 지나는 직선들



$(x, y) = (-1, 1)$

- 삼각함수 in Parameter Space

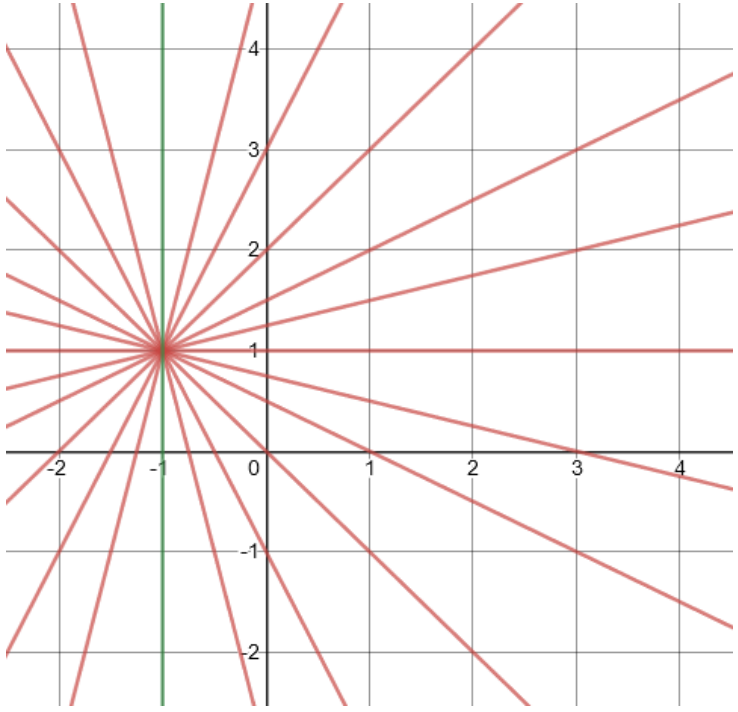


$$r = -\cos\theta + \sin\theta$$

$$= \sqrt{2}\cos(\theta - \alpha),\ \alpha = \tan^{-1}\left(\frac{1}{-1}\right) = \frac{3\pi}{4}$$

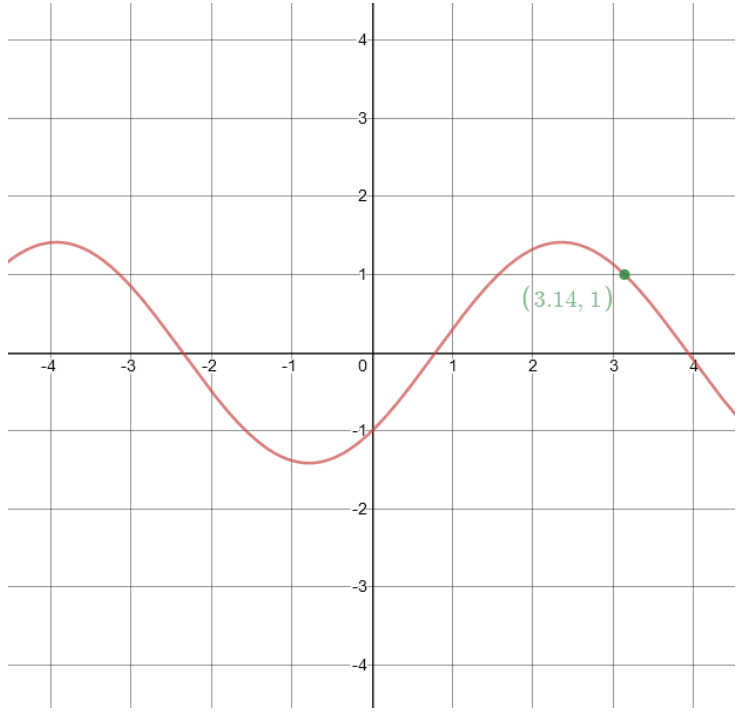
Image Space vs Hough Parameter Space

- 한 점을 지나는 직선들



$(x, y) = (-1, 1)$

- 삼각함수 in Parameter Space

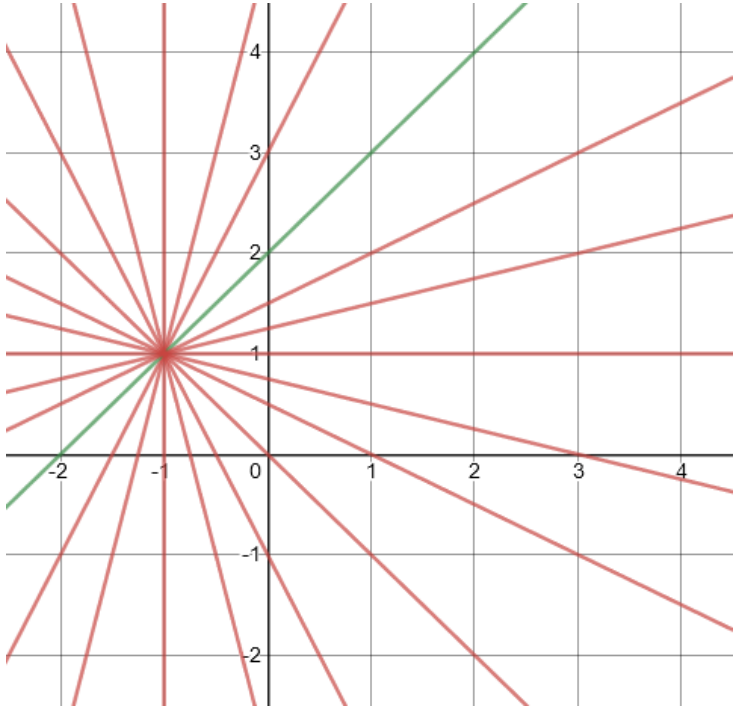


$$r = -\cos\theta + \sin\theta$$

$$= \sqrt{2}\cos(\theta - \alpha),\ \alpha = \tan^{-1}\left(\frac{1}{-1}\right) = \frac{3\pi}{4}$$

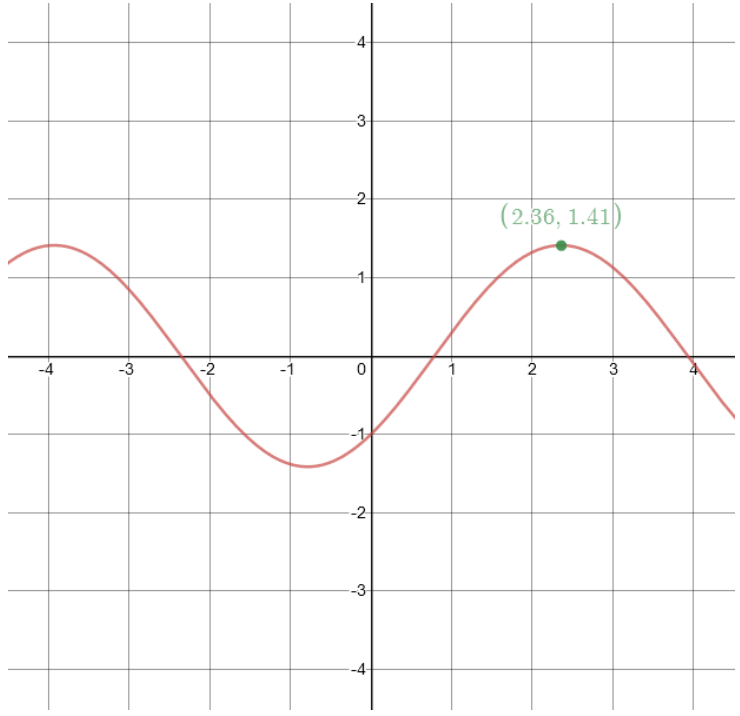
Image Space vs Hough Parameter Space

- 한 점을 지나는 직선들



$(x, y) = (-1, 1)$

- 삼각함수 in Parameter Space

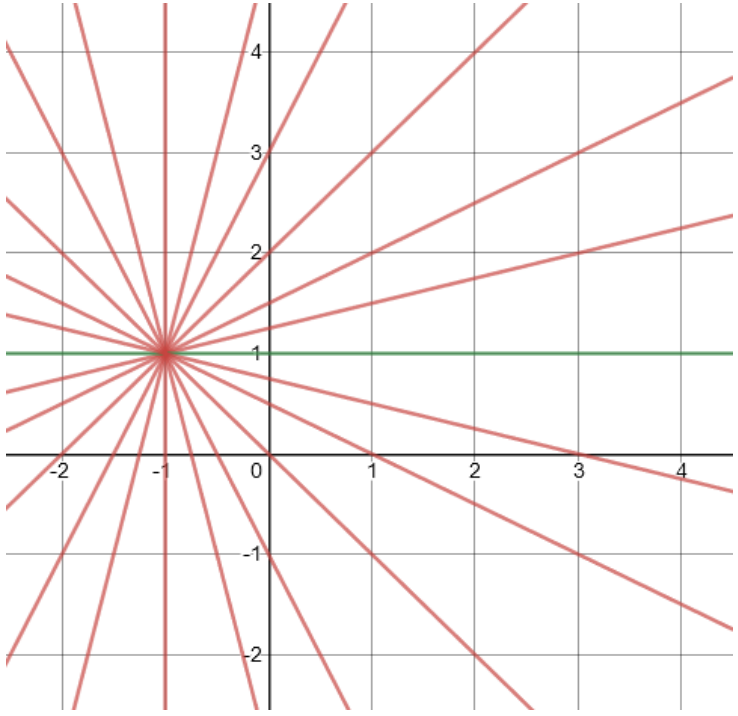


$$r = -\cos\theta + \sin\theta$$

$$= \sqrt{2}\cos(\theta - \alpha),\ \alpha = \tan^{-1}\left(\frac{1}{-1}\right) = \frac{3\pi}{4}$$

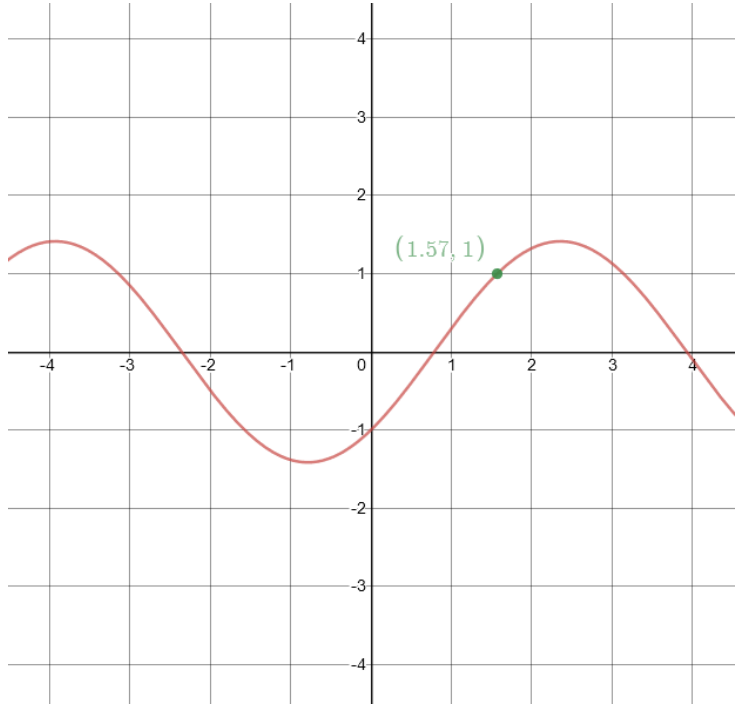
Image Space vs Hough Parameter Space

- 한 점을 지나는 직선들



$$(x, y) = (-1, 1)$$

- 삼각함수 in Parameter Space

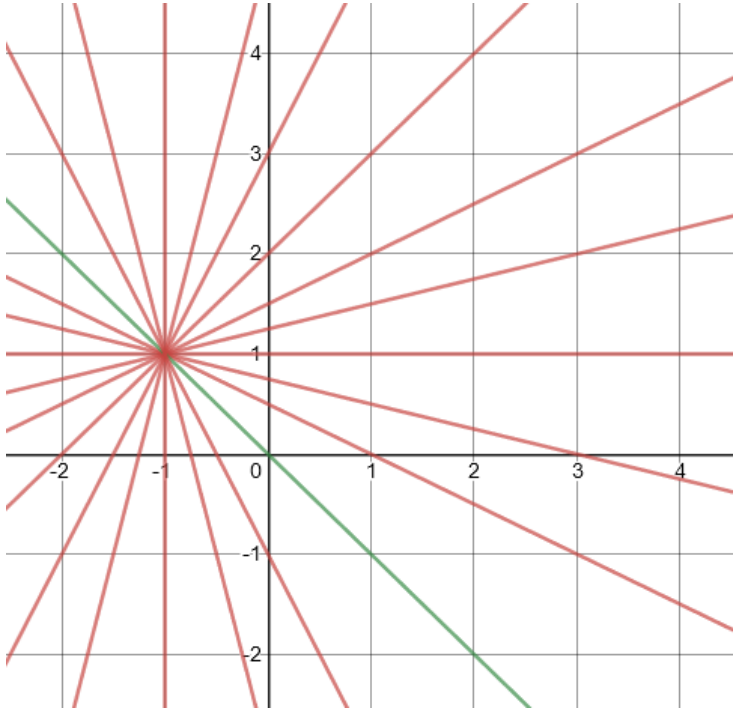


$$r = -\cos\theta + \sin\theta$$

$$= \sqrt{2}\cos(\theta - \alpha),\ \alpha = \tan^{-1}\left(\frac{1}{-1}\right) = \frac{3\pi}{4}$$

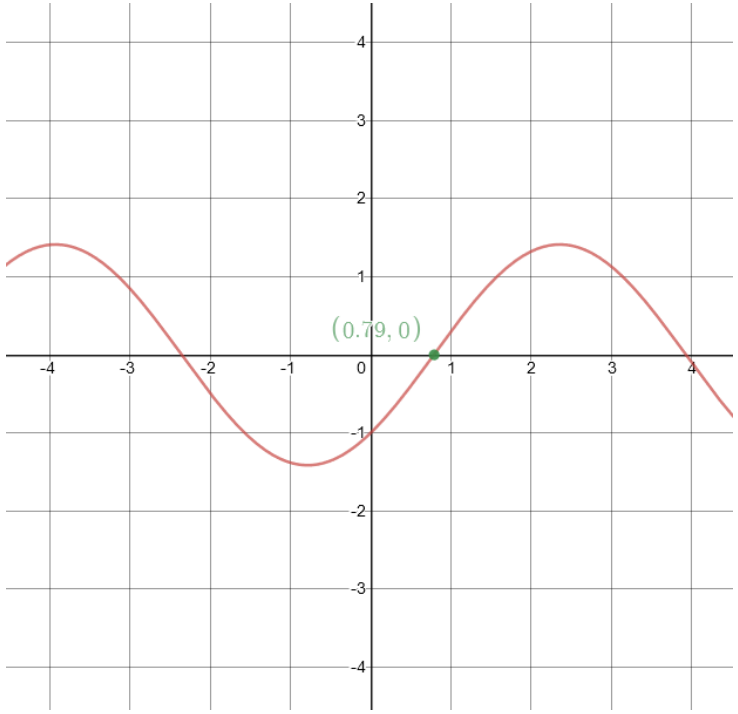
Image Space vs Hough Parameter Space

- 한 점을 지나는 직선들



$$(x, y) = (-1, 1)$$

- 삼각함수 in Parameter Space

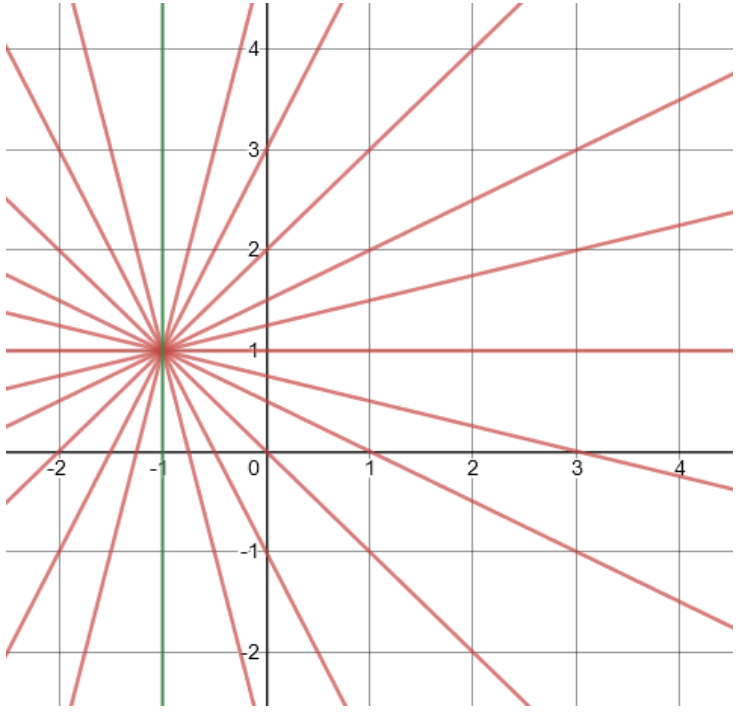


$$r = -\cos\theta + \sin\theta$$

$$= \sqrt{2}\cos(\theta - \alpha),\ \alpha = \tan^{-1}\left(\frac{1}{-1}\right) = \frac{3\pi}{4}$$

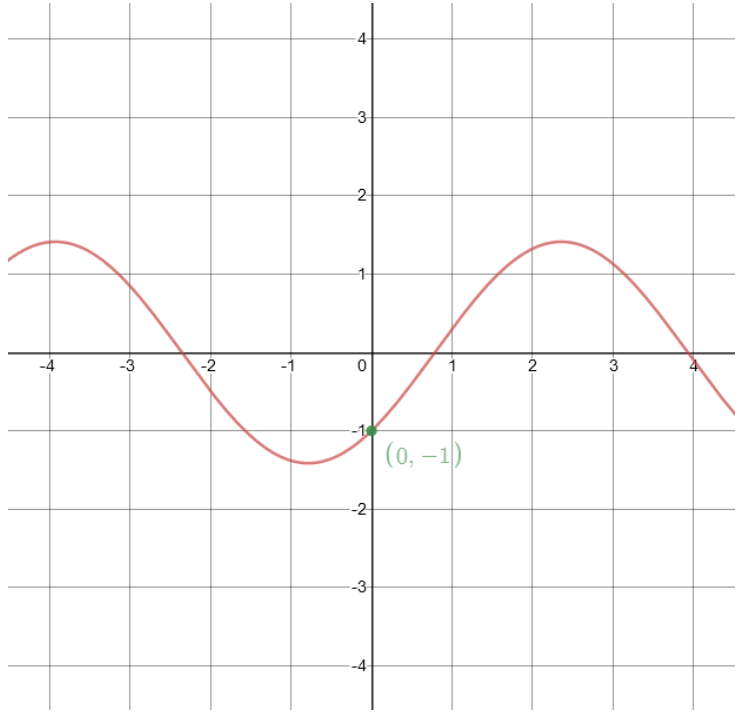
Image Space vs Hough Parameter Space

- 한 점을 지나는 직선들



$$(x, y) = (-1, 1)$$

- 삼각함수 in Parameter Space

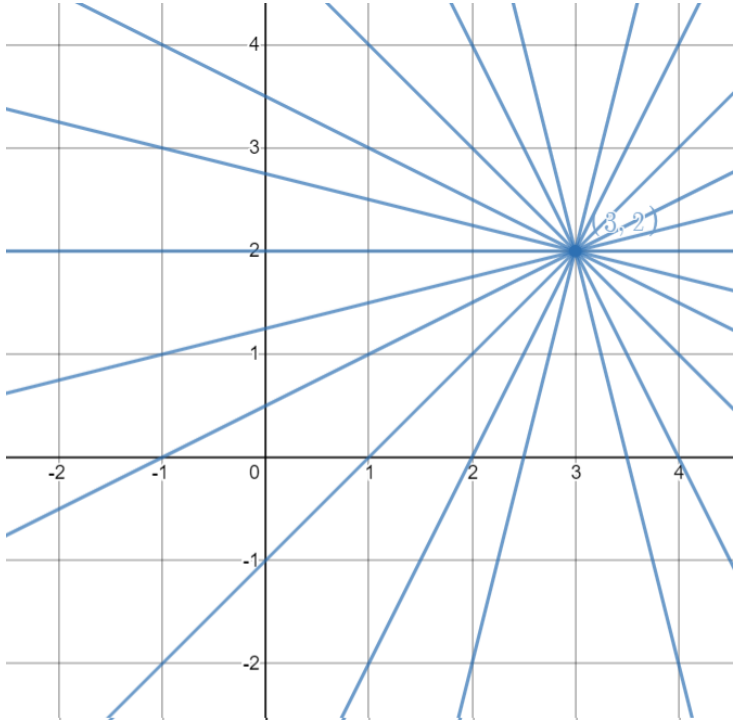


$$r = -\cos\theta + \sin\theta$$

$$= \sqrt{2}\cos(\theta - \alpha),\ \alpha = \tan^{-1}\left(\frac{1}{-1}\right) = \frac{3\pi}{4}$$

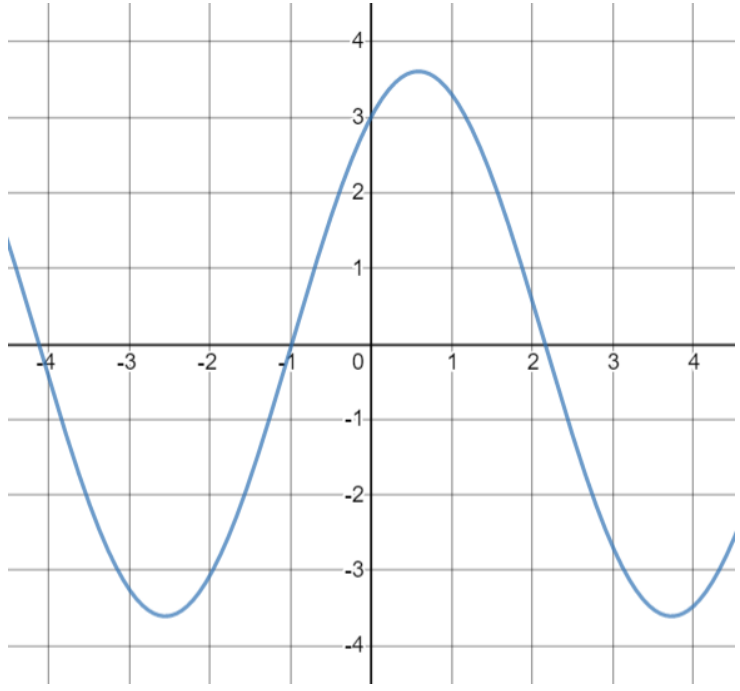
Image Space vs Hough Parameter Space

- 또 다른 점을 지나는 직선들



$$(x, y) = (3, 2)$$

- Sine/Cosine in Hough Space

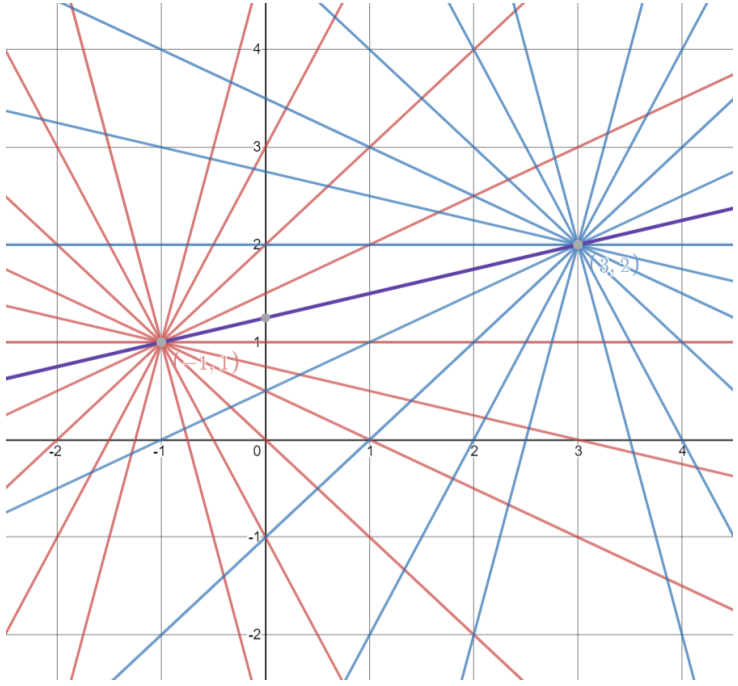


$$r = 3\cos\theta + 2\sin\theta$$

$$= \sqrt{13}\cos(\theta - \alpha),\ \alpha = \tan^{-1}\left(\frac{2}{3}\right) \approx 0.59$$

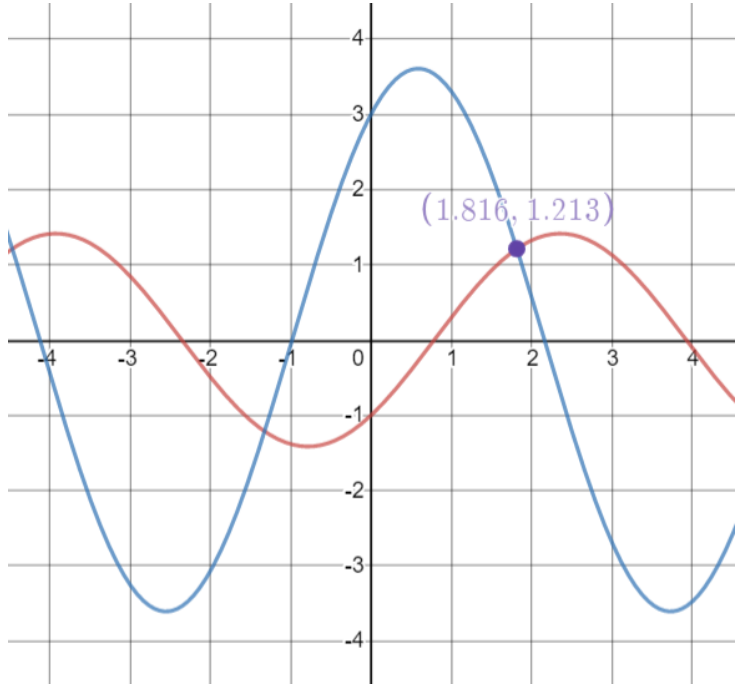
Image Space vs Hough Parameter Space

- 두 점을 각각 지나는 직선들



$(-1, 1),\ (3, 2)$

- 두 직선의 교점 in Parameter Space

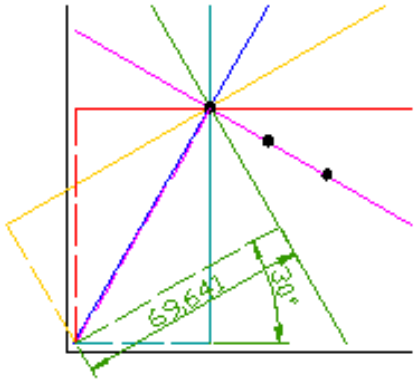


$r = -\cos\theta + \sin\theta,\quad r = 3\cos\theta + 2\sin\theta$

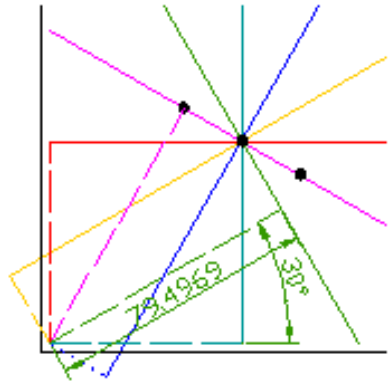
Hough Line Transform

1. Parameter Space의 Range를 정한다.
   - $r \in [0, R], \theta \in [0, 2\pi]$
   - $r \in [-R, R], \theta \in [0, \pi]$
2. Parameter Space를 Discretization한다.
   - 2D Array
3. Image 안에서 모든 Edge point를 찾는다.
4. 각 Pixel에 대해서 만일 그것이 Edge pixel이면 Parameter Space에서 Voting을 한다.
   - 각각의 θ 값에 대해서 r값을 찾아서 Accumulator를 증가시킨다.
5. Parameer Space에서 임계값을 넘는 Cell의 Parameter를 찾는다.

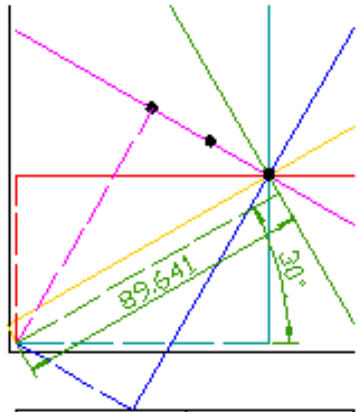
Hough Line Transform



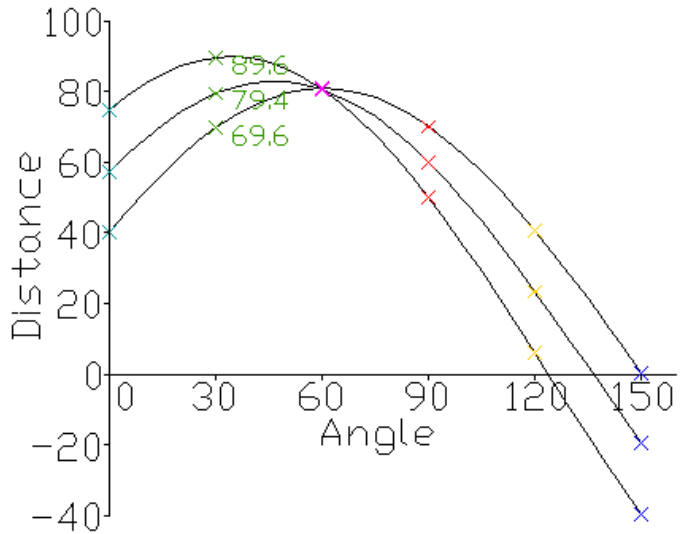
Angle	Dist.
0	40
30	69.6
60	81.2
90	70
120	40.6
150	0.4



Angle	Dist.
0	57.1
30	79.5
60	80.5
90	60
120	23.4
150	-19.5

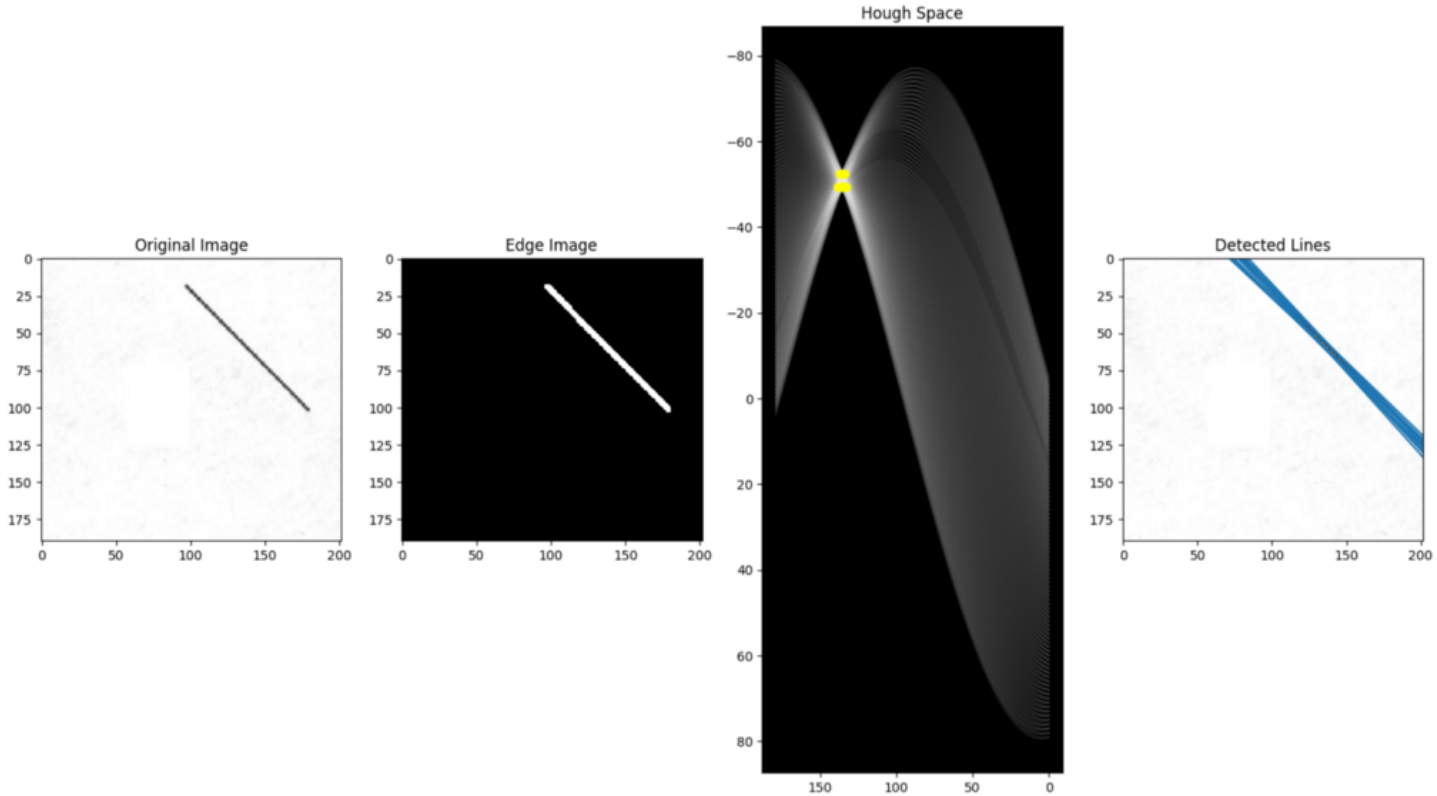


Angle	Dist.
0	74.6
30	89.6
60	80.6
90	50
120	6.0
150	-39.6



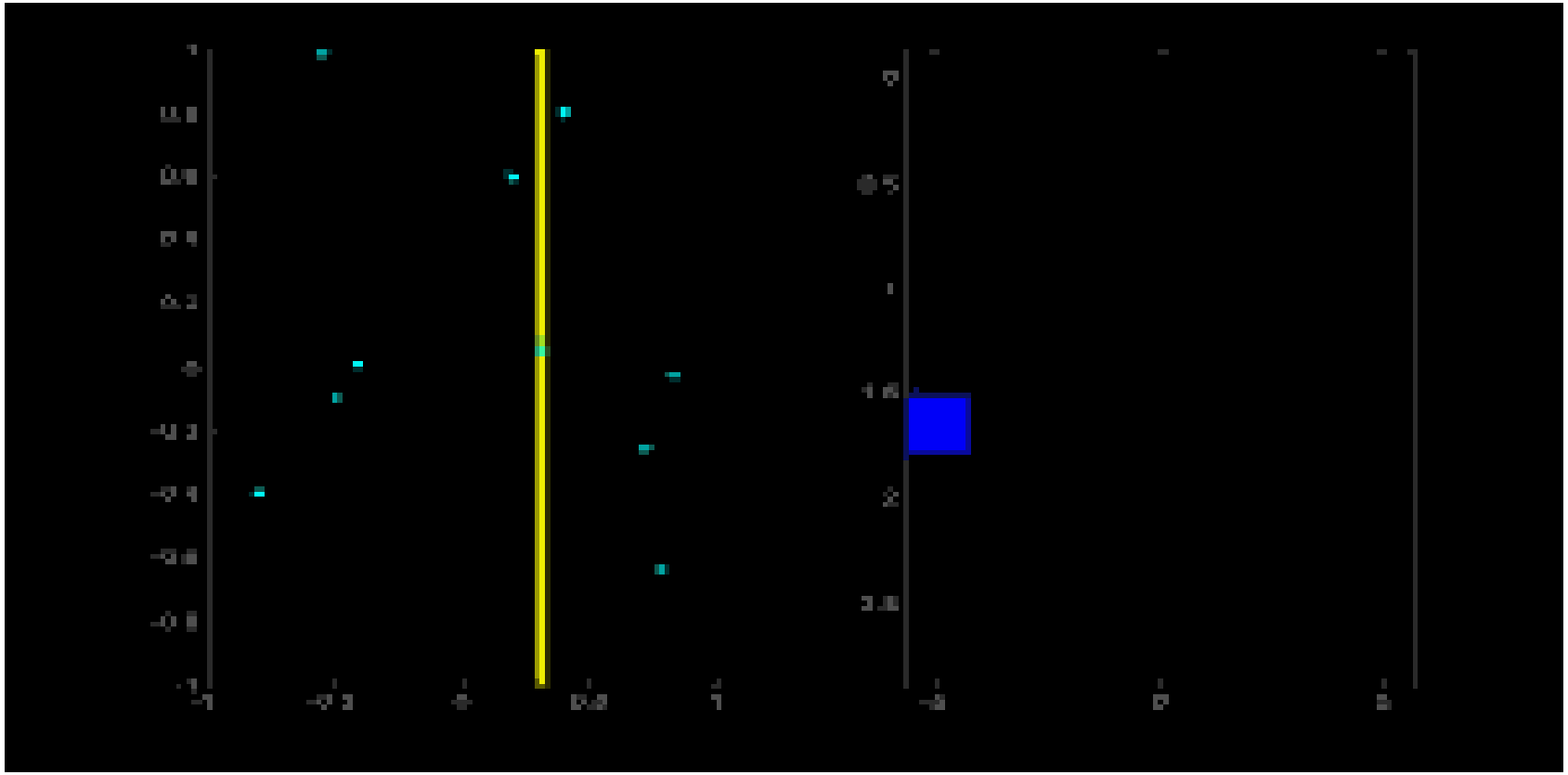
https://opencv-python.readthedocs.io/en/latest/doc/25.imageHoughLineTransform/imageHoughLineTransform.html

Hough Line Transform



https://towardsdatascience.com/lines-detection-with-hough-transform-84020b3b1549

Hough Line Transform



http://homepages.inf.ed.ac.uk/amos/hough.html

`cv.HoughLines()`[1]

```
lines = cv.HoughLines(image, rho, theta, threshold[, lines[, srn[, stn[, min_theta[, max_theta]]]]])
```

- Hough transform을 이용하여 line을 찾는다.
  - `image`: 8-bit, single-channel binary source image. The image may be modified by the function.
  - `lines`: Output vector of lines. Each line is represented by a 2 or 3 element vector (ρ,θ) or (ρ,θ,votes) . ρ is the distance from the coordinate origin (0,0) (top-left corner of the image). θ is the line rotation angle in radians (0∼vertical line,π/2∼horizontal line). votes is the value of accumulator.
  - `rho`: Distance resolution of the accumulator in pixels.
  - `theta`: Angle resolution of the accumulator in radians.
  - `threshold`: Accumulator threshold parameter. Only those lines are returned that get enough votes (> `threshold`).
  - `stn`: For the multi-scale Hough transform, it is a divisor for the distance resolution theta.
  - `min_theta`: For standard and multi-scale Hough transform, minimum angle to check for lines. Must fall between 0 and `max_theta`.
  - `max_theta`: For standard and multi-scale Hough transform, maximum angle to check for lines. Must fall between `min_theta` and `CV_PI`.

1. https://docs.opencv.org/4.5.0/dd/d1a/group__imgproc__feature.html#ga46b4e588934f6c8dfd509cc6e0e4545a

`cv.HoughLinesP()`[1]

```
lines = cv.HoughLinesP(image, rho, theta, threshold[, lines[, minLineLength[, maxLineGap]]])
```

- Probabilistic Hough transform을 이용하여 line을 찾는다.
  - `image`: 8-bit, single-channel binary source image. The image may be modified by the function.
  - `lines`: Output vector of lines. Each line is represented by a 4-element vector (x1,y1,x2,y2) , where (x1,y1) and (x2,y2) are the ending points of each detected line segment.
  - `rho`: Distance resolution of the accumulator in pixels.
  - `theta`: Angle resolution of the accumulator in radians.
  - `threshold`: Accumulator threshold parameter. Only those lines are returned that get enough votes (> `threshold`).
  - `minLineLength`: Minimum line length. Line segments shorter than that are rejected.
  - `maxLineGap`: Maximum allowed gap between points on the same line to link them.

1. https://docs.opencv.org/4.5.0/dd/d1a/group__imgproc__feature.html#ga8618180a5948286384e3b7ca02f6feeb

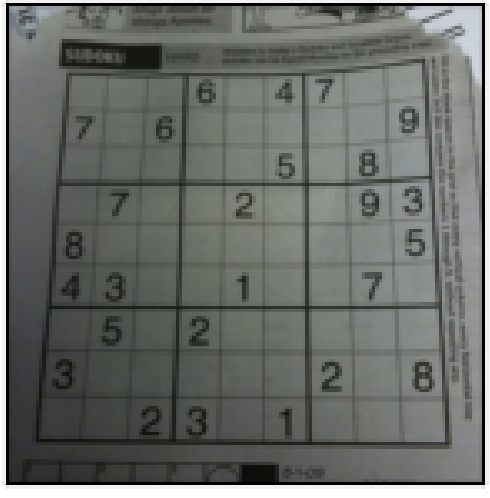
Hough Line Transform: Code 1

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

# Load a color image
img_color = cv2.imread('sudoku.png')

# Convert to a gray
img_gray = cv2.cvtColor(img_color, cv2.COLOR_BGR2GRAY)

# Find Canny edges
img_edge = cv2.Canny(img_gray, 50, 150, apertureSize = 3)

# Find Hough lines
lines = cv2.HoughLines(img_edge, 1, np.pi/180, 200)

# Draw lines
img_lines = img_color.copy()
for i in range(len(lines)):
    for rho, theta in lines[i]:
        a = np.cos(theta)
```

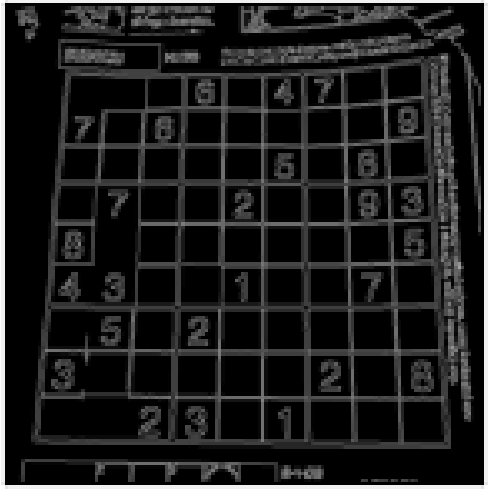
https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_gradients/py_gradients.html

Hough Line Transform: Result 1

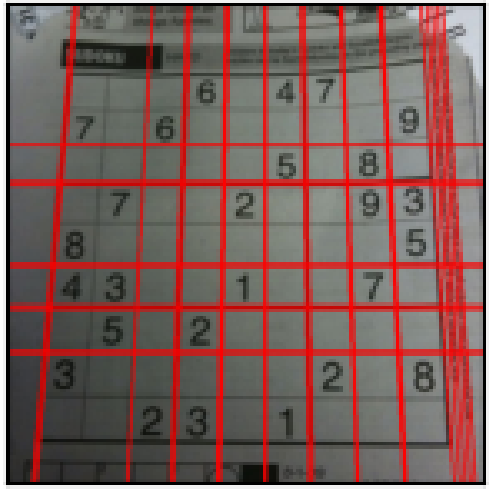
Original



Canny Edges



Hough Lines



Hough Line Transform: Code 2

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

# Load a color image
img_color = cv2.imread('building.jpg')

# Convert to a gray
img_gray = cv2.cvtColor(img_color, cv2.COLOR_BGR2GRAY)

# Find Canny edges
img_edge = cv2.Canny(img_gray, 50, 200, apertureSize = 3)

# Find Hough lines
lines = cv2.HoughLinesP(img_edge, 1, np.pi/180, 80, minLineLength=30, maxLineGap =10)

# Draw lines
img_lines = cv2.cvtColor(img_edge, cv2.COLOR_GRAY2BGR)
for i in range(len(lines)):
    for x1,y1,x2,y2 in lines[i]:
        cv2.line(img_lines, (x1,y1), (x2,y2), (0,0,255), 2)
```

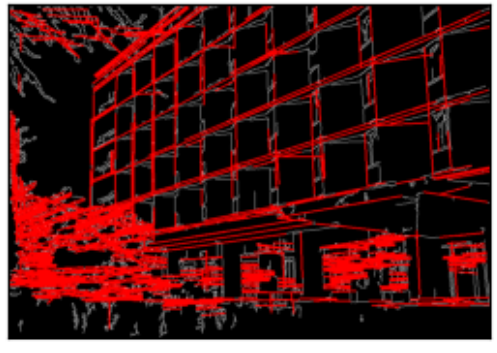
https://docs.opencv.org/4.5.0/dd/d1a/group__imgproc__feature.html#ga8618180a5948286384e3b7ca02f6feeb

Hough Line Transform: Result 2

Original

Canny Edges

Hough Lines



Circle Hough Transform

Circle Hough Transform

- 원의 방정식: 직각좌표계

$$(x - x_0)^2 + (y - y_0)^2 = r^2$$

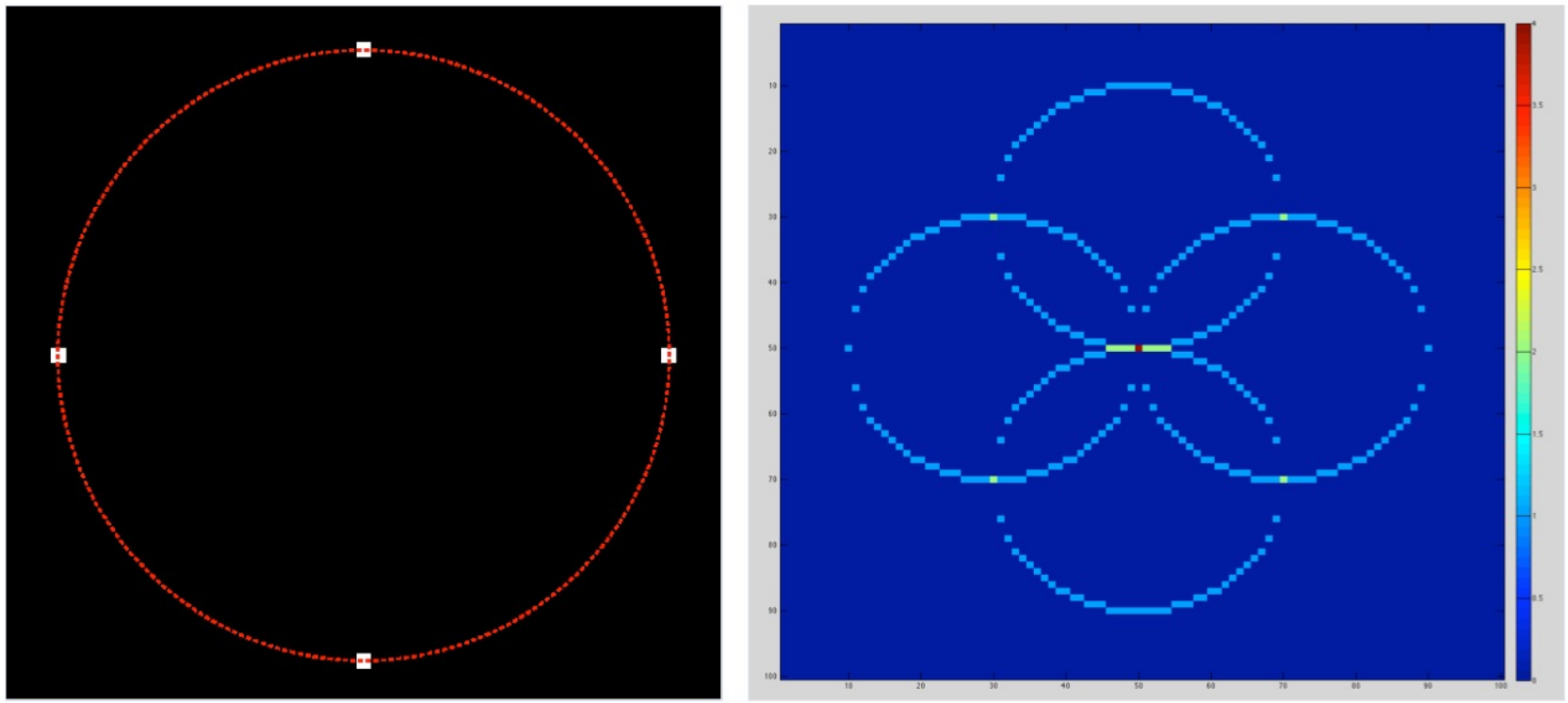
- 원의 방정식: 극좌표계

$$\begin{cases} x = x_0 + r\cos\theta \\ y = y_0 + r\sin\theta \end{cases}$$

- 3차원 공간: r, x_0, y_0
  - x_1, y_1: $(x_0 - x_1)^2 + (y_0 - y_1)^2 = r^2$ (다시 원)
  - r: 여러 값을 고려함
  - Hough Gradient Method는 edge의 gradient 값을 이용함

Circle Hough Transform

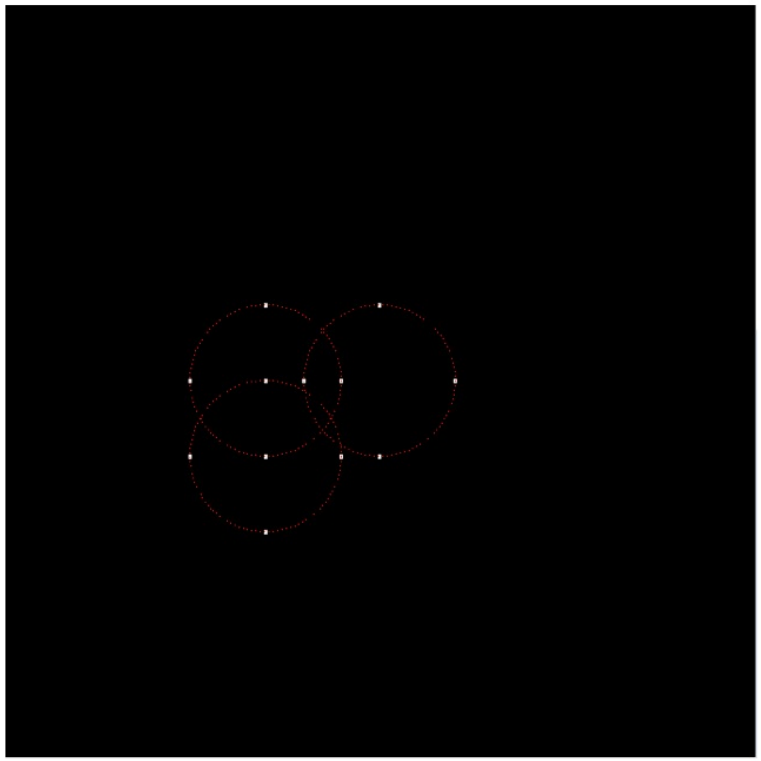
- Find parameters with known radius R

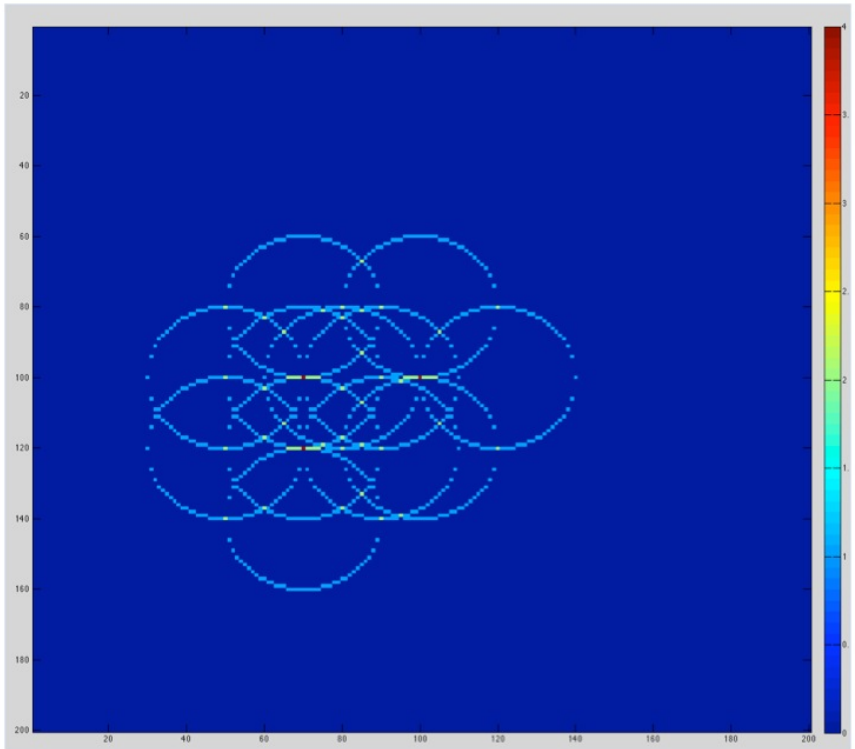


https://en.wikipedia.org/wiki/Circle_Hough_Transform

Circle Hough Transform

- Multiple circles with known radius R





https://en.wikipedia.org/wiki/Circle_Hough_Transform

`cv.HoughCircles()`[1]

circles = cv.HoughCircles(image, method, dp, minDist[, circles[, param1[, param2[, minRadius[, maxRadius]]]]])
`

- Hough transform을 이용하여 circle을 찾는다.
  - `image`: 8-bit, single-channel, grayscale input image.
  - `circles`: Output vector of found circles. Each vector is encoded as 3 or 4 element floating-point vector (x,y,radius) or (x,y,radius,votes).
  - `method`: Detection method.The available methods are `HOUGH_GRADIENT` and `HOUGH_GRADIENT_ALT`.
  - `dp`: Inverse ratio of the accumulator resolution to the image resolution. For example, if dp=1 , the accumulator has the same resolution as the input image. If dp=2 , the accumulator has half as big width and height. For `HOUGH_GRADIENT_ALT` the recommended value is dp=1.5, unless some small very circles need to be detected.
  - `minDist`: Minimum distance between the centers of the detected circles. If the parameter is too small, multiple neighbor circles may be falsely detected in addition to a true one. If it is too large, some circles may be missed.
  - `param1`: First method-specific parameter.
  - `param2`: Second method-specific parameter.
  - `minRadius`: Minimum circle radius.
  - `maxRadius`: Maximum circle radius. If <= 0, uses the maximum image dimension.

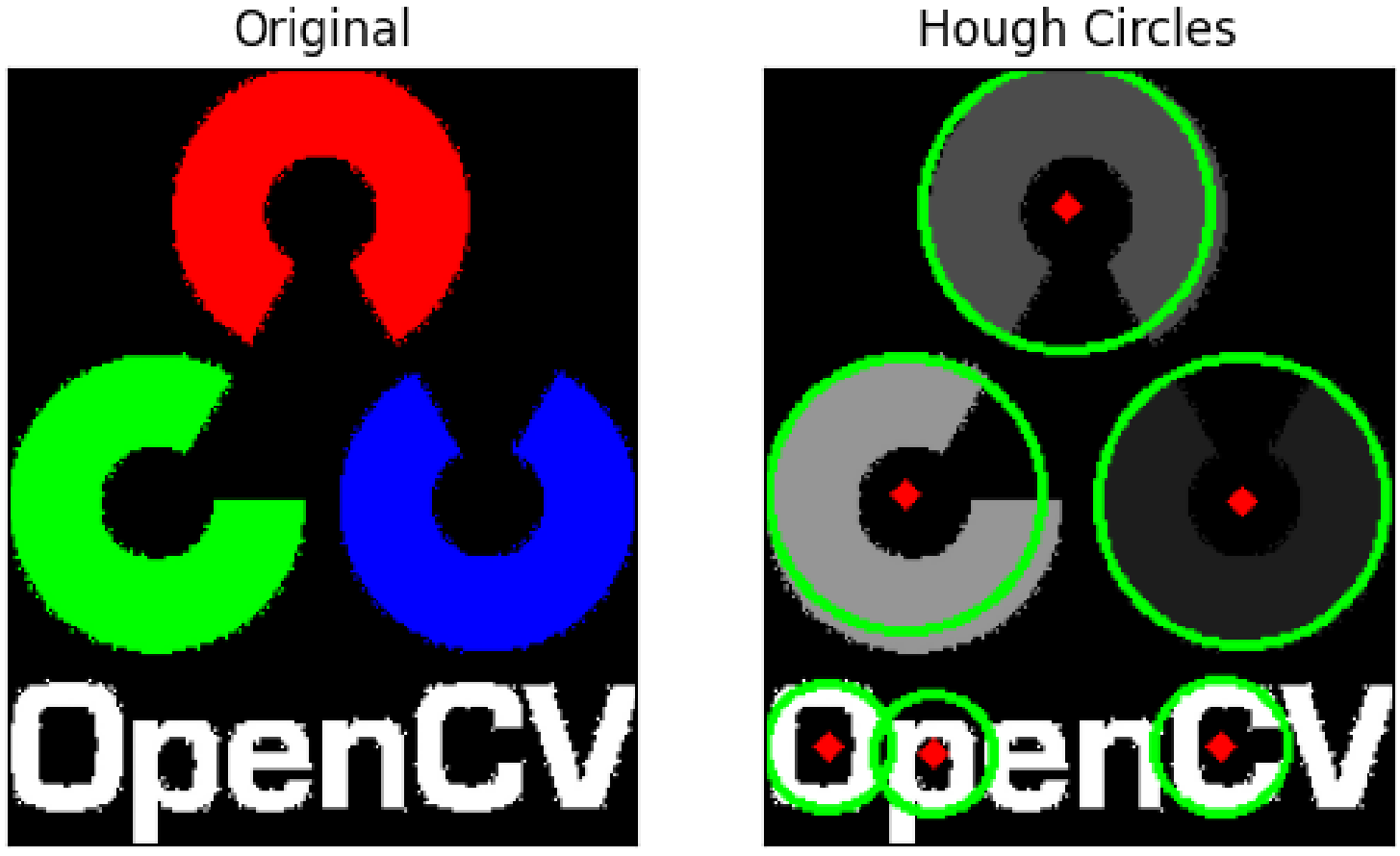
1. https://docs.opencv.org/4.5.0/dd/d1a/group__imgproc__feature.html#ga47849c3be0d0406ad3ca45db65a25d2d

Circle Hough Transform: Code

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

# Load a color image
img_color = cv2.imread('opencv-logo-white.png')

# Convert to a gray
img_gray = cv2.cvtColor(img_color, cv2.COLOR_BGR2GRAY)

# Blur the image
img_blur = cv2.medianBlur(img_gray, 5)

# Find Hough circles
circles = cv2.HoughCircles(img_blur, cv2.HOUGH_GRADIENT, 1, 20, param1=50, param2=25,
minRadius=0, maxRadius=0)
circles = np.uint16(np.around(circles))

# Draw circles
img_circles = cv2.cvtColor(img_gray, cv2.COLOR_GRAY2BGR)
for i in circles[0,:]:
```

https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_houghcircles/py_houghcircles.html#hough-circles

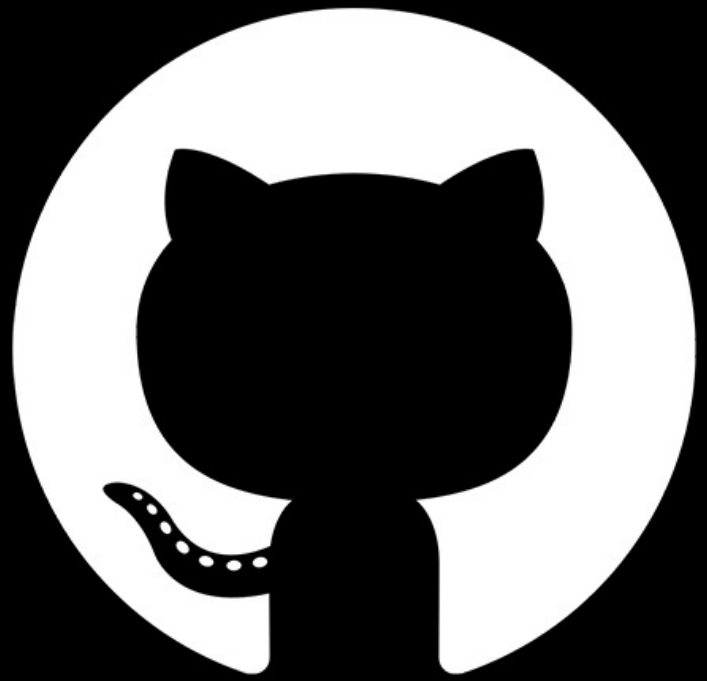
Circle Hough Transform: Result



Summary

- Hough Transforms
  - Lines, Circles
  - Image Space vs. Hough Parameter Space

Push Code to GitHub



References

References

- OpenCV Python Tutorials
  - Core Operations
    - Basic Operations on Images
    - Arithmetic Operations on Images
  - Image Processing
    - Image Thresholding
    - Smoothing Images
    - Morphological Transformations
    - Image Gradients
    - Hough Line Transform
    - Hough Circle Transform