package util.opencv;

public class Line

{

private final Point \_start;

private final Point \_end;

private float \_a = Float.NaN;

private float \_b = Float.NaN;

private boolean \_vertical = false;

public Line(Point start, Point end)

{

\_start = start;

\_end = end;

if (\_end.x - \_start.x != 0)

{

\_a = ((\_end.y - \_start.y) / (\_end.x - \_start.x));

\_b = \_start.y - \_a \* \_start.x;

}

else

{

\_vertical = true;

}

}

/\*\*

\* Indicate whereas the point lays on the line.

\*

\* @param point

\* - The point to check

\* @return <code>True</code> if the point lays on the line, otherwise return <code>False</code>

\*/

public boolean isInside(Point point)

{

float maxX = \_start.x > \_end.x ? \_start.x : \_end.x;

float minX = \_start.x < \_end.x ? \_start.x : \_end.x;

float maxY = \_start.y > \_end.y ? \_start.y : \_end.y;

float minY = \_start.y < \_end.y ? \_start.y : \_end.y;

if ((point.x >= minX && point.x <= maxX) && (point.y >= minY && point.y <= maxY))

{

return true;

}

return false;

}

/\*\*

\* Indicate whereas the line is vertical. <br>

\* For example, line like x=1 is vertical, in other words parallel to axis Y. <br>

\* In this case the A is (+/-)infinite.

\*

\* @return <code>True</code> if the line is vertical, otherwise return <code>False</code>

\*/

public boolean isVertical()

{

return \_vertical;

}

/\*\*

\* y = <b>A</b>x + B

\*

\* @return The <b>A</b>

\*/

public float getA()

{

return \_a;

}

/\*\*

\* y = Ax + <b>B</b>

\*

\* @return The <b>B</b>

\*/

public float getB()

{

return \_b;

}

/\*\*

\* Get start point

\*

\* @return The start point

\*/

public Point getStart()

{

return \_start;

}

/\*\*

\* Get end point

\*

\* @return The end point

\*/

public Point getEnd()

{

return \_end;

}

@Override

public String toString()

{

return String.format("%s-%s", \_start.toString(), \_end.toString());

}

}