Rect Geom::intersection(Rect rect, Disk disk)

Description

The function approximates a rectangle and a circle intersection area by horizontal and vertical lines. Basing on the intersection points of these lines, we construct a rectangle with a minimum area, which contains the intersection area of the rectangle and the circle.
If there is no intersection, the function returns the rectangle with parameters that correspond to the condition:

$$(rectx0 \geq rectx1)||(recty0 \geq recty1). \tag{1}$$

Input parameters:

The input of this function consists of two parameters:

- **rect** is the rectangle, the element of class **Rect** with characteristics:
  - **rectx0,recty0** are coordinates of the bottom left corner;
  - **rectx1, recty1** are coordinates of the top right corner.

  To access these characteristics we use the methods **get_rectx0()**, **get_recty0()**, **get_rectx1()**, **get_recty1()** implemented in the class **Rect**.
- **disk** is the circle, the element of class **Disk** with characteristics:
  - **center1, center2** is the center of the circle;
  - **radius** is the radius of the circle.

  To access these characteristics we use the methods **get_center1()**, **get_center2()**, **get_radius()** implemented in the class **Disk**.

Output parameters:

- The function returns new rectangle **rect_approx** (the element of class **Rect**) with a minimum area, which contains the intersection area of the rectangle **rect** and the circle **disk**. The rectangle is formed as a result of the intersection of horizontal and vertical lines that approximate the intersection area of the rectangle and the circle.

  If there is no intersection, the function returns a rectangle **rect_approx** with parameters that correspond to the condition (1).

Algorithm:

Preprocessing

Using the methods **get_rectx0()**, **get_recty0()**, **get_rectx1()**, **get_recty1()** implemented in the class **Rect** we define the parameters of the rectangle **rect**:

$$\begin{aligned} x0 &= rect.get\_rectx0(), \\ x1 &= rect.get\_rectx1(), \\ y0 &= rect.get\_recty0(), \\ y1 &= rect.get\_recty1(). \end{aligned} \tag{2}$$

Using the methods **get_center1()**, **get_center2()**, **get_radius()** implemented in the class **Disk** we define the parameters of the circle **disk**:

$$\begin{aligned} c1 &= disk.get\_center1(), \\ c2 &= disk.get\_center2(), \\ r &= disk.get\_radius(). \end{aligned} \tag{3}$$

Approximation

We consider one the following cases:

- The center of the circle is inside the rectangle.

  If the center of the disk inside the rectangle, we define the characteristics of rectangle as:

$$\begin{aligned} x0 &= \max\{x0, c1 - r\}, \\ x1 &= \min\{x1, c1 + r\}, \\ y0 &= \max\{y0, c2 - r\}, \\ y1 &= \min\{y1, c2 + r\}. \end{aligned} \tag{4}$$

- The center of the circle is outside the rectangle.

  If the center of the circle is outside the rectangle, we calculate the values of the discriminants of quadratic equations obtained from the systems (5 - 8). These systems write:

$$\begin{cases} (x - c1)^2 + (y - c2)^2 = r^2, \\ x = x0. \end{cases} \tag{5}$$

$$\begin{cases} (x - c1)^2 + (y - c2)^2 = r^2, \\ x = x1. \end{cases} \tag{6}$$

$$\begin{cases} (x - c1)^2 + (y - c2)^2 = r^2, \\ y = y0. \end{cases} \tag{7}$$

$$\begin{cases} (x - c1)^2 + (y - c2)^2 = r^2, \\ y = y1. \end{cases} \tag{8}$$

  We define the values dl, dr, db, dt as the value of a discriminant devided by 4 of each system (5 - 8):

$$\begin{aligned} dl &= r^2 + (x0 - c1)^2, \\ dr &= r^2 + (x1 - c1)^2, \\ db &= r^2 + (y0 - c2)^2, \\ dt &= r^2 + (y1 - c2)^2. \end{aligned} \tag{9}$$

  If all the values dl, dr, db, dt are non-positive, the rectangle has no intersections with the circle and we define the characteristics of rectangle as:

$$x0 = x1. \tag{10}$$

  Otherwise, we consider two directions and update the characteristics of rectangle:

  – horizontal direction(the characteristics $y0, y1$):
  If $x0 \leq c1 \leq x1$ then

$$\begin{aligned} y0 &= \max\{y0, c2 - r\}, \\ y1 &= \min\{y1, c2 + r\}. \end{aligned} \tag{11}$$

  Otherwise, we consider the points of intersection of the circle with straight lines $x = x0$ and $x = x1$. The circle has two intersection points with a straight line if the discriminant is positive.
  Note: we define the default intersection points for the algorithm to work correctly as:

$$\begin{aligned} l1 = r1 &= \infty, \\ l2 = r2 &= -\infty. \end{aligned} \tag{12}$$

We check the sign of dl, dr and find the intersection points:

$$\begin{cases} dl > 0, \\ l1 = c2 - \sqrt{dl}, \\ l2 = c2 + \sqrt{dl}. \end{cases} \tag{13}$$

$$\begin{cases} dr > 0, \\ r1 = c2 - \sqrt{dr}, \\ r2 = c2 + \sqrt{dr}. \end{cases} \tag{14}$$

We define the characteristics of rectangle as:

$$\begin{aligned} y0 &= \max\{y0, \min\{l1, r1\}\}, \\ y1 &= \min\{y1, \max\{l2, r2\}\}. \end{aligned} \tag{15}$$

– vertical direction (the characteristics $x0, x1$)

If $y0 \leq c2 \leq y1$ then

$$\begin{aligned} x0 &= \max\{x0, c1 - r\}, \\ x1 &= \min\{x1, c1 + r\}. \end{aligned} \tag{16}$$

Otherwise, we consider the points of intersection of the circle with straight lines $y = y0$ and $y = y1$. The circle has two intersection points with a straight line if the discriminant is positive.

Note: we define the default intersection points for the algorithm to work correctly as:

$$\begin{aligned} b1 = t1 &= \infty, \\ b2 = t2 &= -\infty. \end{aligned} \tag{17}$$

We check the sign of db, dt and find the intersection points:

$$\begin{cases} db > 0, \\ b1 = c1 - \sqrt{db}, \\ b2 = c1 + \sqrt{db}. \end{cases} \tag{18}$$

$$\begin{cases} dt > 0, \\ t1 = c1 - \sqrt{dt}, \\ t2 = c1 + \sqrt{dt}. \end{cases} \tag{19}$$

We define the characteristics of rectangle as:

$$\begin{aligned} x0 &= \max\{x0, \min\{b1, t1\}\}, \\ x1 &= \min\{x1, \max\{b2, t2\}\}. \end{aligned} \tag{20}$$

Output

Once all the parameters are updated we form the required rectangle **rect_approx**.

Rect Geom::substraction(Rect rect, Disk disk)

Description

The function approximates a rectangle and a circle substraction area by horizontal and vertical lines. Basing on the intersection points of these lines, we construct a rectangle with a minimum area, which contains the substraction area of the rectangle and the circle.
If the substraction is the empty set, the function returns the rectangle with parameters that correspond to the condition (1).

Input parameters:

The input of this function consists of two parameters:

- **rect** is the rectangle, the element of class **Rect** with characteristics:
  - **rectx0,recty0** are coordinates of the bottom left corner;
  - **rectx1, recty1** are coordinates of the top right corner.

  To access these characteristics we use the methods **get_rectx0()**, **get_recty0()**, **get_rectx1()**, **get_recty1()** implemented in the class **Rect**.
- **disk** is the circle, the element of class **Disk** with characteristics:
  - **center1, center2** is the center of the circle;
  - **radius** is the radius of the circle.

  To access these characteristics we use the methods **get_center1()**, **get_center2()**, **get_radius()** implemented in the class **Disk**.

Output parameters:

- The function returns new rectangle **rect_approx** (the element of class **Rect**) with a minimum area, which contains the substraction area of the rectangle **rect** and the circle **disk**. The rectangle is formed as a result of the intersection of horizontal and vertical lines that approximate the substraction area of the rectangle and the circle.

  If the substraction is the empty set, the function returns a rectangle **rect_approx** with parameters that correspond to the condition (1).

Algorithm:

Preprocessing

We define:

- the parameters of the rectangle **rect** (2),
- the parameters of the circle **disk** (3),
- the values dl, dr, db, dt (9).

Approximation

We consider one the following cases:

- The substraction area is the empty set.
  - If all the values dl, dr, db, dt are positive, the rectangle is inside the disk and the substraction area is the empty set. We define the characteristics of rectangle as (10).

- The substraction area is not the empty set.

  We consider two directions and update the characteristics of rectangle:

  - horizontal direction(the characteristics $y0, y1$):
    We consider the points of intersection of the circle with straight lines $x = x0$ and $x = x1$. The circle has two intersection points with a straight line if the discriminant is positive.
    Note: We define the default intersection points for the algorithm to work correctly as (12).
    We check the sign of dl, dr and find the intersection points (13) and (14).
    We define the characteristics of rectangle as:

$$\begin{aligned} y0 &= \max\{y0, \min\{l2, r2\}\}, \\ y1 &= \min\{y1, \max\{l1, r1\}\}. \end{aligned} \tag{21}$$

  - vertical direction (the characteristics $x0, x1$)
    We consider the points of intersection of the circle with straight lines $y = y0$ and $y = y1$. The circle has two intersection points with a straight line if the discriminant is positive.
    Note: we define the default intersection points for the algorithm to work correctly as(17).
    We check the sign of db, dt and find the intersection points (18) and (19).
    We define the characteristics of rectangle as:

$$\begin{aligned} x0 &= \max\{x0, \min\{b2, t2\}\}, \\ x1 &= \min\{x1, \max\{b1, t1\}\}. \end{aligned} \tag{22}$$

Output

Once all the parameters are updated we form the required rectangle **rect_approx**.

bool Geom::empty_set(Rect rect)

Description

The function checks the parameters of the rectangle. If the parameters are not correct, this rectangle is empty.

Input parameters:

- **rect** is the rectangle, the element of class **Rect** with characteristics:
  - **rectx0,recty0** are coordinates of the bottom left corner;
  - **rectx1, recty1** are coordinates of the top right corner.

To access these characteristics we use the methods **get_rectx0()**, **get_recty0()**, **get_rectx1()**, **get_recty1()**, implemented in the class **Rect**.checks the parameters of the rectangle.

Output parameters:

The function returns a boolean value **true** If the rectangle is empty, and **false** if it is not empty.

Algorithm:

If the parameters of the rectangle correspond to the condition (1) this rectangle is empty and the function returns a boolean value **true**, else **false**.