Navigation: Up, Table of Contents, Bibliography, Index, Title Page

Class

CGAL::Circle_2<Kernel>

Definition

An object of type *Circle_2*<*Kernel*> is a circle in the two-dimensional Euclidean plane $\mathbb{E}^2$. The circle is oriented, i.e. its boundary has clockwise or counterclockwise orientation . The boundary splits $\mathbb{E}^2$ into a positive and a negative side, where the positive side is to the left of the boundary. The boundary also splits $\mathbb{E}^2$ into a bounded and an unbounded side. Note that the circle can be degenerated, i.e. the squared radius may be zero.

Creation

Circle_2<*Kernel*> *c* (*Point_2*<*Kernel*> *center*,
Kernel::*FT* *squared_radius*,
Orientation *ori* = *COUNTERCLOCKWISE*);

introduces a variable *c* of type *Circle_2*<*Kernel*>. It is initialized to the circle with center *center*, squared radius *squared_radius* and orientation *ori*.
Precondition: ori $\neq$ *COLLINEAR*, and further, *squared_radius* $\geq$ 0.

Circle_2<*Kernel*> *c* (*Point_2*<*Kernel*> *p*,
Point_2<*Kernel*> *q*,
Point_2<*Kernel*> *r*);

introduces a variable *c* of type *Circle_2*<*Kernel*>. It is initialized to the unique circle which passes through the points *p*, *q* and *r*. The orientation of the circle is the orientation of the point triple *p*, *q*, *r*.
Precondition: p, *q*, and *r* are not collinear.

Circle_2<*Kernel*> *c* (*Point_2*<*Kernel*> *p*,
Point_2<*Kernel*> *q*,
Orientation *ori* = *COUNTERCLOCKWISE*);

introduces a variable *c* of type *Circle_2*<*Kernel*>. It is initialized to the circle with diameter *pq* and orientation *ori*.
Precondition: ori $\neq$ *COLLINEAR*.

Circle_2<*Kernel*> *c* (*Point_2*<*Kernel*> *center*,
Orientation *ori* = *COUNTERCLOCKWISE*);

introduces a variable *c* of type *Circle_2*<*Kernel*>. It is initialized to the circle with center *center*, squared radius zero and

orientation *ori*.
Precondition: ori ≠ *COLLINEAR*.

Postcondition: c.*is_degenerate()* = *true*.

Access Functions

Point_2<Kernel>	*c.center ()*	returns the center of *c*.
Kernel::FT	*c.squared_radius ()*	returns the squared radius of *c*.
Orientation	*c.orientation ()*	returns the orientation of *c*.
bool	*c.operator == (circle2)*	returns *true*, iff *c* and *circle2* are equal, i.e. if they have the same center, same squared radius and same orientation .
bool	*c.operator != (circle2)*	returns *true*, iff *c* and *circle2* are not equal.

Predicates

bool	*c.is_degenerate ()*	returns *true*, iff *c* is degenerate, i.e. if *c* has squared radius zero.
Oriented_side	*c.oriented_side (Point_2<Kernel> p)*	returns either the constant *ON_ORIENTED_BOUNDARY*, *ON_POSITIVE_SIDE*, or *ON_NEGATIVE_SIDE*, iff *p* lies on the boundary, properly on the positive side, or properly on the negative side of *c*, resp.
Bounded_side	*c.bounded_side (Point_2<Kernel> p)*	returns *ON_BOUNDED_SIDE*, *ON_BOUNDARY*, or *ON_UNBOUNDED_SIDE* iff *p* lies properly inside, on the boundary, or properly outside of *c*, resp.
bool	*c.has_on_positive_side (Point_2<Kernel> p)*	
bool	*c.has_on_negative_side (Point_2<Kernel> p)*	
bool	*c.has_on_boundary (Point_2<Kernel> p)*	
bool	*c.has_on_bounded_side (Point_2<Kernel> p)*	
bool	*c.has_on_unbounded_side (Point_2<Kernel> p)*	

Miscellaneous

Circle_2<Kernel>	*c.opposite ()*	returns the circle with the same center and squared radius as *c* but with opposite orientation .
Circle_2<Kernel>	*c.orthogonal_transform (Aff_transformation_2<Kernel> at)*	returns the circle obtained by applying *at* on *c*. *Precondition: at* is an orthogonal transformation.
Bbox_2	*c.bbox ()*	returns a bounding box containing *c*.

See Also

Kernel::Circle_2

Next: *Direction_2<Kernel>*

Navigation: Table of Contents, Bibliography, Index, Title Page

The CGAL Project . Tue, December 21, 2004 .