THUNDER

Generated by Doxygen 1.8.14

Contents

Index

1	File	Index			1
	1.1	File Lis	st		1
2	File	Docum	entation		3
	2.1	include	e/Geometry	y/Euler.h File Reference	3
		2.1.1	Detailed	Description	4
		2.1.2	Function	Documentation	4
			2.1.2.1	alignZ()	4
			2.1.2.2	angle() [1/3]	4
			2.1.2.3	angle() [2/3]	5
			2.1.2.4	angle() [3/3]	5
			2.1.2.5	direction()	6
			2.1.2.6	quaternion() [1/3]	6
			2.1.2.7	quaternion() [2/3]	6
			2.1.2.8	quaternion() [3/3]	7
			2.1.2.9	quaternion_conj()	7
			2.1.2.10	quaternion_mul()	7
			2.1.2.11	randRotate2D()	8
			2.1.2.12	randRotate3D()	8
			2.1.2.13	reflect3D()	8
			2.1.2.14	rotate2D() [1/2]	8
			2.1.2.15	rotate2D() [2/2]	9
			2.1.2.16	rotate3D() [1/3]	9
			2.1.2.17	rotate3D() [2/3]	9
			2.1.2.18	rotate3D() [3/3]	10
			2.1.2.19	rotate3DX()	10
			2.1.2.20	rotate3DY()	10
			2.1.2.21	rotate3DZ()	11

13

Chapter 1

File Index

4	4	 	i		- 4
٦.	п.	-11	Δ	ΙI	CT.

Here is a list of all documented files with brief description

include/Geometry/Euler.h												
Some description about Euler.h	 	 										3

2 File Index

Chapter 2

File Documentation

2.1 include/Geometry/Euler.h File Reference

some description about Euler.h

```
#include <cmath>
#include <gsl/gsl_math.h>
#include "Macro.h"
#include "Typedef.h"
#include "Precision.h"
#include "Random.h"
#include "Functions.h"
```

Functions

void quaternion_mul (dvec4 &dst, const dvec4 &a, const dvec4 &b)

Calculate the product of two quaternions.

dvec4 quaternion_conj (const dvec4 &quat)

Calculate the conjugate quaternion of a quaternion.

• void angle (double &phi, double &theta, const dvec3 &src)

Calculate ϕ and θ given a certain direction ${\bf v}$.

• void angle (double &phi, double &theta, double &psi, const dmat33 &src)

Calculate ϕ , θ and ψ of the rotation represented by the rotation matrix \mathbf{R} .

• void angle (double &phi, double &theta, double &psi, const dvec4 &src)

Calculate $\phi,\,\theta$ and ψ of the rotation represented by the quaternion ${\bf q}.$

• void quaternion (dvec4 &dst, const double phi, const double theta, const double psi)

Calculate the quaternion \mathbf{q} for representing the rotation, given 3 Euler angles ϕ , θ and ψ .

• void quaternion (dvec4 &dst, const double phi, const dvec3 &axis)

Calculate the quaternion \mathbf{q} for representing the rotation, given the rotation axis \mathbf{r} and the rotation angle around this axis ϕ .

void quaternion (dvec4 &dst, const dmat33 &src)

Calculate the quaternion ${\bf q}$ for representing the rotation, given the rotation matrix ${\bf R}$.

void rotate2D (dmat22 &dst, const dvec2 &vec)

Calculate the rotation matrix (2D) \mathbf{R} , which rotates the unit vector $\mathbf{v_0} = \{1, 0\}$ to the given unit vector \mathbf{v} .

void rotate2D (dmat22 &dst, const double phi)

Calculate the rotation matrix (2D) \mathbf{R} , given the rotation angle ϕ .

· void direction (dvec3 &dst, const double phi, const double theta)

Caclulate the unit direction vector \mathbf{v} , given the rotation angle ϕ and θ .

• void rotate3D (dmat33 &dst, const double phi, const double theta, const double psi)

Caclulate the rotation matrix \mathbf{R} , given the rotation angle ϕ , θ and ψ .

void rotate3D (dmat33 &dst, const dvec4 &src)

Calculate the rotation matrix \mathbf{R} , given the unit quaternion \mathbf{q} which represents this rotation.

void rotate3DX (dmat33 &dst, const double phi)

Calculate the rotation matrix ${f R}$ which represents the rotation along X-axis with rotation angle ϕ .

- void rotate3DY (dmat33 &dst, const double phi)
- void rotate3DZ (dmat33 &dst, const double phi)
- void alignZ (dmat33 &dst, const dvec3 &vec)
- void rotate3D (dmat33 &dst, const double phi, const dvec3 &axis)
- void reflect3D (dmat33 &dst, const dvec3 &plane)
- void swingTwist (dvec4 &swing, dvec4 &twist, const dvec4 &src, const dvec3 &vec)
- void randDirection (dvec2 &dir)
- void randRotate2D (dmat22 &rot)
- void randQuaternion (dvec4 &quat)
- void randRotate3D (dmat33 &rot)

2.1.1 Detailed Description

some description about Euler.h

Details about Euler.h

2.1.2 Function Documentation

2.1.2.1 alignZ()

This function calculates the rotation matrix for aligning a direction vector to Z-axis.

dst	the rotation matrix
vec	the direction vector

```
2.1.2.2 angle() [1/3]
```

```
void angle (
double & phi,
```

```
double & theta,
const dvec3 & src )
```

Calculate ϕ and θ given a certain direction ${\bf v}.$

Parameters

out	phi	ϕ
out	theta	θ
in	src	\mathbf{v}

2.1.2.3 angle() [2/3]

Calculate ϕ , θ and ψ of the rotation represented by the rotation matrix ${\bf R}.$

Parameters

out	phi	ϕ
out	theta	θ
out	psi	ψ
in	src	\mathbf{R}

2.1.2.4 angle() [3/3]

Calculate ϕ , θ and ψ of the rotation represented by the quaternion ${\bf q}$.

out	phi	ϕ
out	theta	θ
out	psi	ψ
in	src	q

2.1.2.5 direction()

Caclulate the unit direction vector \mathbf{v} , given the rotation angle ϕ and θ .

Parameters

out	dst	\mathbf{v}
in	phi	ϕ
in	theta	θ

2.1.2.6 quaternion() [1/3]

Calculate the quaternion ${\bf q}$ for representing the rotation, given 3 Euler angles ϕ , θ and ψ .

Parameters

out	dst	\mathbf{q}
in	phi	ϕ
in	theta	θ
in	psi	ψ

2.1.2.7 quaternion() [2/3]

Calculate the quaternion $\bf q$ for representing the rotation, given the rotation axis $\bf r$ and the rotation angle around this axis ϕ .

Parameters

out	dst	q
in	phi	ϕ
in	axis	r

2.1.2.8 quaternion() [3/3]

```
void quaternion ( \label{eq:dvec4 & dst,}  const dmat33 & sc )
```

Calculate the quaternion ${\bf q}$ for representing the rotation, given the rotation matrix ${\bf R}$.

Parameters

out	dst	q
in	src	R

2.1.2.9 quaternion_conj()

Calculate the conjugate quaternion of a quaternion.

Returns

the conjugate quaternion

Parameters

```
in quat a quaternion
```

2.1.2.10 quaternion_mul()

Calculate the product of two quaternions.

Parameters

out	dst	product, a quaternion
in	а	left multiplier, quaternion
in	b	right multiplier, quaternion

2.1.2.11 randRotate2D()

This function generates a random unit quaternion.

2.1.2.12 randRotate3D()

This function generates a random 3D rotation matrix.

2.1.2.13 reflect3D()

```
void reflect3D ( \mbox{dmat33 \& } dst, \mbox{const dvec3 \& } plane \mbox{)}
```

This function calculates the transformation matrix of reflection against a certain plane given by its normal vector.

Parameters

dst	the rotation matrix
plane	the normal vector the reflection plane

2.1.2.14 rotate2D() [1/2]

```
void rotate2D (

dmat22 & dst,

const dvec2 & vec)
```

Calculate the rotation matrix (2D) ${f R}$, which rotates the unit vector ${f v_0}=\{1,0\}$ to the given unit vector ${f v}$.

dst	the rotation matrix
vec	the unit vector

Parameters

out	dst	\mathbf{R}
in	vec	v

2.1.2.15 rotate2D() [2/2]

Calculate the rotation matrix (2D) ${\bf R},$ given the rotation angle $\phi.$

Parameters

out	dst	\mathbf{R}
in	phi	ϕ

2.1.2.16 rotate3D() [1/3]

Caclulate the rotation matrix \mathbf{R} , given the rotation angle ϕ , θ and ψ .

Parameters

out	dst	\mathbf{R}
in	phi	ϕ
in	theta	θ
in	psi	ψ

2.1.2.17 rotate3D() [2/3]

```
void rotate3D (

dmat33 & dst,

const dvec4 & src)
```

Calculate the rotation matrix $\boldsymbol{R},$ given the unit quaternion \boldsymbol{q} which represents this rotation.

Parameters

out	dst	R
in	src	q

```
2.1.2.18 rotate3D() [3/3]
```

This function calculates the rotation matrix of rotation along a certain axis given by a direction vector of phi.

Parameters

dst	the rotation matrix
phi	phi
axis	the direction vector indicating the axis

2.1.2.19 rotate3DX()

```
void rotate3DX ( \label{eq:dmat33 \& dst,}  const double phi )
```

Calculate the rotation matrix ${f R}$ which represents the rotation along X-axis with rotation angle ϕ .

Parameters

out	dst	\mathbf{R}
in	phi	ϕ

2.1.2.20 rotate3DY()

```
void rotate3DY ( \label{eq:dmat33 \& dst,}  const double phi )
```

This function calculates the rotation matrix of rotation along Y-axis of phi.

Parameters

dst	the rotation matrix
phi	phi

2.1.2.21 rotate3DZ()

```
void rotate3DZ ( \label{eq:dmat33 \& dst,}  const double phi )
```

This function calculates the rotation matrix of rotation along Z-axis of phi.

dst	the rotation matrix
phi	phi

Index

```
alignZ
     Euler.h, 4
angle
     Euler.h, 4, 5
direction
     Euler.h, 6
Euler.h
     alignZ, 4
     angle, 4, 5
     direction, 6
     quaternion, 6, 7
     quaternion_conj, 7
     quaternion_mul, 7
     randRotate2D, 8
     randRotate3D, 8
     reflect3D, 8
     rotate2D, 8, 9
     rotate3DX, 10
     rotate3DY, 10
     rotate3DZ, 11
     rotate3D, 9, 10
include/Geometry/Euler.h, 3
quaternion
     Euler.h, 6, 7
quaternion_conj
     Euler.h, 7
quaternion_mul
     Euler.h, 7
randRotate2D
     Euler.h, 8
randRotate3D
     Euler.h, 8
reflect3D
     Euler.h, 8
rotate2D
     Euler.h, 8, 9
rotate3DX
     Euler.h, 10
rotate3DY
     Euler.h, 10
rotate3DZ
     Euler.h, 11
rotate3D
```

Euler.h, 9, 10