THUNDER

Generated by Doxygen 1.8.14

Contents

Index

1	File	Index			1
	1.1	File Lis	st		1
2	File	Docume	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]	5
			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]	7
			2.1.2.8	quaternion() [3/3]	7
			2.1.2.9	quaternion_conj()	7
			2.1.2.10	quaternion_mul()	8
			2.1.2.11	randRotate2D()	8
			2.1.2.12	randRotate3D()	8
			2.1.2.13	reflect3D()	9
			2.1.2.14	rotate2D() [1/2]	9
			2.1.2.15	rotate2D() [2/2]	9
			2.1.2.16	rotate3D() [1/3]	10
			2.1.2.17	rotate3D() [2/3]	10
			2.1.2.18	rotate3D() [3/3]	10
			2.1.2.19	rotate3DX()	11
			2.1.2.20	rotate3DY()	11
			2.1.2.21	rotate3DZ()	11
			2.1.2.22	swingTwist()	12

13

Chapter 1

File Index

1.1 File List

Here is a list of all documented files with brief descriptions:

include/Geometry/Euler.h

Euler.h contains several functions, for operations of quaternions, converting between Euler angles, rotation matrices and unit quaternions and sampling rotation matrices from even distribution 3

2 File Index

Chapter 2

File Documentation

2.1 include/Geometry/Euler.h File Reference

Euler.h contains several functions, for operations of quaternions, converting between Euler angles, rotation matrices and unit quaternions and sampling rotation matrices from even distribution.

```
#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 \mathbf{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 ${\bf R}$.

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

Calculate ϕ , θ 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 unit 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 unit 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 unit 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 $\mathbf R$ which represents the rotation along X-axis with rotation angle ϕ .

void rotate3DY (dmat33 &dst, const double phi)

Calculate the rotation matrix $\mathbf R$ which represents the rotation along Y-axis with rotation angle ϕ .

• void rotate3DZ (dmat33 &dst, const double phi)

Calculate the rotation matrix \mathbf{R} which represents the rotation along Z-axis with rotation angle ϕ .

void alignZ (dmat33 &dst, const dvec3 &vec)

Calculate the rotation matrix ${\bf R}$ which aligns a direction vector ${\bf v}$ to Z-axis.

void rotate3D (dmat33 &dst, const double phi, const dvec3 &axis)

Calculate the rotation matrix \mathbf{R} which represents the rotation along the axis \mathbf{v} with rotation angle ϕ .

void reflect3D (dmat33 &dst, const dvec3 &plane)

Calculate the transformation matrix ${\bf M}$ of reflection against a certian plane, which is represented by its normal vector ${\bf n}$

void swingTwist (dvec4 &swing, dvec4 &twist, const dvec4 &src, const dvec3 &vec)

Calculate the two quaternions $\mathbf{q_s}$ and $\mathbf{q_t}$, which represent swing and twist along axis \mathbf{v} respectively, representing the rotation represented by quaternion \mathbf{q} .

void randRotate2D (dmat22 &rot)

Sample a 2D rotation matrix ${f R}$ from even distribution.

void randRotate3D (dmat33 &rot)

Sample a 3D rotation matrix ${f R}$ from even distribution.

2.1.1 Detailed Description

Euler.h contains several functions, for operations of quaternions, converting between Euler angles, rotation matrices and unit quaternions and sampling rotation matrices from even distribution.

Quaternions are a number system that extends the complex numbers. Unit quaternions provide a convenient mathematical notation for representing rotations of objects in 3D. Compared to Euler angles, they are simpler to compose and aovid the problem of glimbal lock. Compared to rotation matrices, they are more compact and more efficient. Moroever, unlike Euler angles, unit quaternions do not rely on the choosing and order of the rotation axes.

To be noticed, Euler angles in this file follow the standard of ZXZ Euler system. In other words, Euler angle set $\{\phi,\theta,\psi\}$ stands for rotating along Z axis with ϕ , followed by rotating along X axis with θ , and followed by rotating along Z axis with ψ .

2.1.2 Function Documentation

2.1.2.1 alignZ()

Calculate the rotation matrix ${\bf R}$ which aligns a direction vector ${\bf v}$ to Z-axis.

Parameters

out	dst	\mathbf{R}
in	vec	v

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

const dvec3 & src)

Parameters

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

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

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

```
2.1.2.4 angle() [3/3]
```

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

```
double & theta,
double & psi,
const dvec4 & src )
```

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

Parameters

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

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

2.1.2.7 quaternion() [2/3]

Calculate the unit 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]

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

Parameters

out	dst	\mathbf{q}
in	src	\mathbf{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.

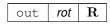
Parameters

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

2.1.2.11 randRotate2D()

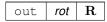
Sample a 2D rotation matrix $\boldsymbol{\mathrm{R}}$ from even distribution.

Parameters



2.1.2.12 randRotate3D()

Sample a 3D rotation matrix ${\bf R}$ from even distribution.



2.1.2.13 reflect3D()

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

Calculate the transformation matrix ${\bf M}$ of reflection against a certian plane, which is represented by its normal vector ${\bf n}$.

Parameters

out	dst	\mathbf{M}
in	plane	n

2.1.2.14 rotate2D() [1/2]

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

Parameters

out	dst	R
in	vec	v

2.1.2.15 rotate2D() [2/2]

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

out	dst	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]

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

Parameters

out	dst	R
in	src	q

2.1.2.18 rotate3D() [3/3]

Calculate the rotation matrix ${\bf R}$ which represents the rotation along the axis ${\bf v}$ with rotation angle ϕ .

out	dst	R
in	phi	ϕ
in	axis	v

2.1.2.19 rotate3DX()

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

Calculate the rotation matrix ${\bf R}$ 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 )
```

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

Parameters

out	dst	\mathbf{R}
in	phi	ϕ

2.1.2.21 rotate3DZ()

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

out	dst	R
in	phi	φ

2.1.2.22 swingTwist()

Calculate the two quaternions $\mathbf{q_s}$ and $\mathbf{q_t}$, which represent swing and twist along axis \mathbf{v} respectively, representing the rotation represented by quaternion \mathbf{q} .

out	swing	$\mathbf{q_s}$
out	twist	$\mathbf{q_t}$
in	src	\mathbf{q}
in	vec	v

Index

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

Euler.h, 9, 10