

THUNDER

Generated by Doxygen 1.8.14

Contents

Chapter 1

File Index

1.1 File List

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

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

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 \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 \mathbf{R} .
- void [angle](#) (double &phi, double &theta, double &psi, const dvec4 &src)
Calculate ϕ , θ and ψ of the rotation represented by the quaternion \mathbf{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 \mathbf{q} for representing the rotation, given the rotation matrix \mathbf{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)

Calculate 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)

Calculate 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 \mathbf{R} which aligns a direction vector \mathbf{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 M of reflection against a certian plane, which is represented by its normal vector \mathbf{n} .

2.1.1 Detailed Description

some description about [Euler.h](#)

Details about [Euler.h](#)

2.1.2 Function Documentation

2.1.2.1 alignZ()

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

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

Parameters

out	<i>dst</i>	\mathbf{R}
in	<i>vec</i>	\mathbf{v}

2.1.2.2 `angle()` [1/3]

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

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

Parameters

out	<i>phi</i>	ϕ
out	<i>theta</i>	θ
in	<i>src</i>	\mathbf{v}

2.1.2.3 `angle()` [2/3]

```
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} .

Parameters

out	<i>phi</i>	ϕ
out	<i>theta</i>	θ
out	<i>psi</i>	ψ
in	<i>src</i>	\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 \mathbf{q} .

Parameters

out	<i>phi</i>	ϕ
out	<i>theta</i>	θ
out	<i>psi</i>	ψ
in	<i>src</i>	\mathbf{q}

2.1.2.5 direction()

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

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

Parameters

out	<i>dst</i>	\mathbf{v}
in	<i>phi</i>	ϕ
in	<i>theta</i>	θ

2.1.2.6 quaternion() [1/3]

```
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 .

Parameters

out	<i>dst</i>	\mathbf{q}
in	<i>phi</i>	ϕ
in	<i>theta</i>	θ
in	<i>psi</i>	ψ

2.1.2.7 quaternion() [2/3]

```
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 .

Parameters

out	<i>dst</i>	q
in	<i>phi</i>	ϕ
in	<i>axis</i>	r

2.1.2.8 quaternion() [3/3]

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

Calculate the quaternion **q** for representing the rotation, given the rotation matrix **R**.

Parameters

out	<i>dst</i>	q
in	<i>src</i>	R

2.1.2.9 quaternion_conj()

```
dvec4 quaternion_conj (
    const dvec4 & quat )
```

Calculate the conjugate quaternion of a quaternion.

Returns

the conjugate quaternion

Parameters

in	<i>quat</i>	a quaternion
----	-------------	--------------

2.1.2.10 quaternion_mul()

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

Calculate the product of two quaternions.

Parameters

out	<i>dst</i>	product, a quaternion
in	<i>a</i>	left multiplier, quaternion
in	<i>b</i>	right multiplier, quaternion

2.1.2.11 reflect3D()

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

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

Parameters

out	<i>dst</i>	\mathbf{M}
in	<i>plane</i>	\mathbf{n}

2.1.2.12 rotate2D() [1/2]

```
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} .

Parameters

out	<i>dst</i>	\mathbf{R}
in	<i>vec</i>	\mathbf{v}

2.1.2.13 rotate2D() [2/2]

```
void rotate2D (
    dmat22 & dst,
    const double phi )
```

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

Parameters

out	<i>dst</i>	R
in	<i>phi</i>	ϕ

2.1.2.14 rotate3D() [1/3]

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

Calculate the rotation matrix **R**, given the rotation angle ϕ , θ and ψ .

Parameters

out	<i>dst</i>	R
in	<i>phi</i>	ϕ
in	<i>theta</i>	θ
in	<i>psi</i>	ψ

2.1.2.15 rotate3D() [2/3]

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

Calculate the rotation matrix **R**, given the unit quaternion **q** which represents this rotation.

Parameters

out	<i>dst</i>	R
in	<i>src</i>	q

2.1.2.16 rotate3D() [3/3]

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

Calculate the rotation matrix **R** which represents the rotation along the axis **v** with rotation angle ϕ .

Parameters

out	<i>dst</i>	R
in	<i>phi</i>	ϕ
in	<i>axis</i>	v

2.1.2.17 rotate3DX()

```
void rotate3DX (
    dmat33 & dst,
    const double phi )
```

Calculate the rotation matrix **R** which represents the rotation along X-axis with rotation angle ϕ .

Parameters

out	<i>dst</i>	R
in	<i>phi</i>	ϕ

2.1.2.18 rotate3DY()

```
void rotate3DY (
    dmat33 & dst,
    const double phi )
```

Calculate the rotation matrix **R** which represents the rotation along Y-axis with rotation angle ϕ .

Parameters

out	<i>dst</i>	R
in	<i>phi</i>	ϕ

2.1.2.19 rotate3DZ()

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

Calculate the rotation matrix **R** which represents the rotation along Z-axis with rotation angle ϕ .

Parameters

out	<i>dst</i>	R
in	<i>phi</i>	ϕ

