

TireGround

Davide Stocco

March 2020

Generated by Doxygen 1.8.13

Contents

1	TireGround	1
2	Namespace Index	5
2.1	Namespace List	5
3	Hierarchical Index	7
3.1	Class Hierarchy	7
4	Class Index	9
4.1	Class List	9
5	Namespace Documentation	11
5.1	TireGround Namespace Reference	11
5.1.1	Detailed Description	13
5.2	TireGround::algorithms Namespace Reference	13
5.2.1	Detailed Description	13
5.2.2	Function Documentation	13
5.2.2.1	intersectPointSegment()	13
5.2.2.2	intersectRayPlane()	14
5.2.2.3	mean()	14
5.2.2.4	minmax_XY() [1/2]	14
5.2.2.5	minmax_XY() [2/2]	15
5.2.2.6	trapezoidArea()	15
5.2.2.7	weightedMean() [1/2]	15
5.2.2.8	weightedMean() [2/2]	15
5.3	TireGround::RDF Namespace Reference	16
5.3.1	Detailed Description	16
5.4	TireGround::RDF::algorithms Namespace Reference	16
5.4.1	Detailed Description	17
5.4.2	Function Documentation	17
5.4.2.1	firstToken()	17
5.4.2.2	getElement()	17
5.4.2.3	split()	17
5.4.2.4	tail()	17

6	Class Documentation	19
6.1	TireGround::RDF::BBox2D Class Reference	19
6.1.1	Detailed Description	20
6.1.2	Constructor & Destructor Documentation	20
6.1.2.1	BBox2D()	20
6.1.3	Member Function Documentation	20
6.1.3.1	print()	20
6.1.3.2	updateBBox2D()	20
6.2	TireGround::Disk Class Reference	20
6.2.1	Detailed Description	21
6.2.2	Constructor & Destructor Documentation	21
6.2.2.1	Disk()	22
6.2.3	Member Function Documentation	22
6.2.3.1	contactPlane()	22
6.2.3.2	contactTriangles()	22
6.2.3.3	getLineArea()	23
6.2.3.4	intersectPlane()	23
6.2.3.5	intersectSegment()	23
6.2.3.6	isPointInside()	24
6.2.3.7	segmentArea()	24
6.2.3.8	segmentLength()	24
6.2.3.9	set()	24
6.2.3.10	setOriginXZ()	25
6.2.3.11	y()	25
6.3	TireGround::ETRTO Class Reference	25
6.3.1	Detailed Description	26
6.3.2	Constructor & Destructor Documentation	26
6.3.2.1	ETRTO()	26
6.3.3	Member Function Documentation	26
6.3.3.1	print()	26
6.4	TireGround::MagicFormula Class Reference	26
6.4.1	Detailed Description	29
6.4.2	Constructor & Destructor Documentation	29
6.4.2.1	MagicFormula()	29
6.4.3	Member Function Documentation	30
6.4.3.1	evaluateContact()	30
6.4.3.2	fourPointsSampling()	30
6.4.3.3	getArea() [1/2]	30
6.4.3.4	getArea() [2/2]	30
6.4.3.5	getEulerAngleX()	31
6.4.3.6	getEulerAngleY()	31
6.4.3.7	getEulerAngleZ()	31

6.4.3.8	getFriction() [1/2]	31
6.4.3.9	getFriction() [2/2]	31
6.4.3.10	getMFpoint() [1/2]	31
6.4.3.11	getMFpoint() [2/2]	32
6.4.3.12	getMFpointRF() [1/2]	32
6.4.3.13	getMFpointRF() [2/2]	32
6.4.3.14	getNormal() [1/2]	32
6.4.3.15	getNormal() [2/2]	33
6.4.3.16	getRelativeCamber()	33
6.4.3.17	getRho() [1/2]	33
6.4.3.18	getRho() [2/2]	34
6.4.3.19	getVolume() [1/2]	34
6.4.3.20	getVolume() [2/2]	34
6.4.3.21	pointSampling()	34
6.4.3.22	print()	35
6.4.3.23	printETRTOGeometry()	35
6.4.3.24	setOrigin()	35
6.4.3.25	setReferenceFrame()	36
6.4.3.26	setRotationMatrix()	36
6.4.3.27	setTotalTransformationMatrix()	36
6.4.3.28	setup() [1/2]	36
6.4.3.29	setup() [2/2]	37
6.5	TireGround::RDF::MeshSurface Class Reference	37
6.5.1	Detailed Description	38
6.5.2	Constructor & Destructor Documentation	38
6.5.2.1	MeshSurface() [1/2]	38
6.5.2.2	MeshSurface() [2/2]	38
6.5.3	Member Function Documentation	38
6.5.3.1	intersectAABBtree()	38
6.5.3.2	intersectBBox()	39
6.5.3.3	LoadFile()	39
6.5.3.4	printData()	39
6.5.3.5	set()	39
6.6	TireGround::MultiDisk Class Reference	40
6.6.1	Detailed Description	43
6.6.2	Constructor & Destructor Documentation	43
6.6.2.1	MultiDisk() [1/3]	43
6.6.2.2	MultiDisk() [2/3]	43
6.6.2.3	MultiDisk() [3/3]	44
6.6.3	Member Function Documentation	44
6.6.3.1	getArea() [1/2]	44
6.6.3.2	getArea() [2/2]	45

6.6.3.3	getDiskFriction()	45
6.6.3.4	getDiskMFpoint()	45
6.6.3.5	getDiskMFpointRF()	45
6.6.3.6	getDiskNormal()	46
6.6.3.7	getDiskOriginXYZ() [1/2]	46
6.6.3.8	getDiskOriginXYZ() [2/2]	46
6.6.3.9	getDiskRho()	46
6.6.3.10	getEulerAngleX()	47
6.6.3.11	getEulerAngleY()	47
6.6.3.12	getEulerAngleZ()	47
6.6.3.13	getFriction() [1/2]	47
6.6.3.14	getFriction() [2/2]	47
6.6.3.15	getMFeffectiveR()	48
6.6.3.16	getMFeffectiveRF()	48
6.6.3.17	getMFeffectiveY()	48
6.6.3.18	getMFpoint() [1/2]	48
6.6.3.19	getMFpoint() [2/2]	49
6.6.3.20	getMFpointRF() [1/2]	49
6.6.3.21	getMFpointRF() [2/2]	49
6.6.3.22	getNormal() [1/2]	49
6.6.3.23	getNormal() [2/2]	50
6.6.3.24	getRelativeCamber()	50
6.6.3.25	getRho() [1/2]	50
6.6.3.26	getRho() [2/2]	50
6.6.3.27	getVolume() [1/2]	51
6.6.3.28	getVolume() [2/2]	51
6.6.3.29	pointSampling()	51
6.6.3.30	print()	52
6.6.3.31	printETRTGeometry()	52
6.6.3.32	setDiskOriginXZ() [1/2]	52
6.6.3.33	setDiskOriginXZ() [2/2]	52
6.6.3.34	setOrigin()	53
6.6.3.35	setReferenceFrame()	53
6.6.3.36	setRotationMatrix()	53
6.6.3.37	setTotalTransformationMatrix()	53
6.6.3.38	setup() [1/2]	54
6.6.3.39	setup() [2/2]	54
6.7	TireGround::ReferenceFrame Class Reference	54
6.7.1	Detailed Description	55
6.7.2	Constructor & Destructor Documentation	55
6.7.2.1	ReferenceFrame()	55
6.7.3	Member Function Documentation	56

6.7.3.1	getEulerAngleX()	56
6.7.3.2	getEulerAngleY()	56
6.7.3.3	getEulerAngleZ()	56
6.7.3.4	set()	56
6.7.3.5	setOrigin()	56
6.7.3.6	setRotationMatrix()	56
6.7.3.7	setTotalTransformationMatrix()	57
6.8	TireGround::SamplingGrid Class Reference	57
6.8.1	Detailed Description	57
6.8.2	Constructor & Destructor Documentation	58
6.8.2.1	SamplingGrid() [1/2]	58
6.8.2.2	SamplingGrid() [2/2]	58
6.8.3	Member Function Documentation	58
6.8.3.1	set() [1/2]	58
6.8.3.2	set() [2/2]	59
6.8.3.3	setSwitchNumber()	59
6.9	TireGround::Shadow Class Reference	59
6.9.1	Detailed Description	59
6.9.2	Constructor & Destructor Documentation	59
6.9.2.1	Shadow()	60
6.9.3	Member Function Documentation	60
6.9.3.1	update()	60
6.10	TicToc Class Reference	60
6.11	TireGround::Tire Class Reference	61
6.11.1	Detailed Description	63
6.11.2	Constructor & Destructor Documentation	63
6.11.2.1	Tire()	63
6.11.3	Member Function Documentation	63
6.11.3.1	evaluateContact()	64
6.11.3.2	getArea() [1/2]	64
6.11.3.3	getArea() [2/2]	64
6.11.3.4	getEulerAngleX()	64
6.11.3.5	getEulerAngleY()	64
6.11.3.6	getEulerAngleZ()	65
6.11.3.7	getFriction() [1/2]	65
6.11.3.8	getFriction() [2/2]	65
6.11.3.9	getMFpoint() [1/2]	65
6.11.3.10	getMFpoint() [2/2]	65
6.11.3.11	getMFpointRF() [1/2]	66
6.11.3.12	getMFpointRF() [2/2]	66
6.11.3.13	getNormal() [1/2]	66
6.11.3.14	getNormal() [2/2]	66

6.11.3.15	getRelativeCamber()	67
6.11.3.16	getRho() [1/2]	67
6.11.3.17	getRho() [2/2]	67
6.11.3.18	getVolume() [1/2]	68
6.11.3.19	getVolume() [2/2]	68
6.11.3.20	pointSampling()	68
6.11.3.21	print()	69
6.11.3.22	printETRTOGeometry()	69
6.11.3.23	setOrigin()	69
6.11.3.24	setReferenceFrame()	69
6.11.3.25	setRotationMatrix()	70
6.11.3.26	setTotalTransformationMatrix()	70
6.11.3.27	setup() [1/2]	70
6.11.3.28	setup() [2/2]	70
6.12	TireGround::RDF::Triangle3D Class Reference	71
6.12.1	Detailed Description	73
6.12.2	Constructor & Destructor Documentation	73
6.12.2.1	Triangle3D()	73
6.12.3	Member Function Documentation	73
6.12.3.1	intersectEdgePlane()	73
6.12.3.2	intersectPlane()	74
6.12.3.3	intersectRay()	74
6.12.3.4	print()	74
6.12.3.5	setVertices() [1/2]	74
6.12.3.6	setVertices() [2/2]	75
6.13	TireGround::RDF::TriangleRoad Class Reference	75
6.13.1	Detailed Description	77
6.13.2	Constructor & Destructor Documentation	77
6.13.2.1	TriangleRoad()	77
6.13.3	Member Function Documentation	77
6.13.3.1	intersectEdgePlane()	77
6.13.3.2	intersectPlane()	78
6.13.3.3	intersectRay()	78
6.13.3.4	print()	78
6.13.3.5	setFriction()	78
6.13.3.6	setVertices() [1/2]	79
6.13.3.7	setVertices() [2/2]	79
Index		81

Chapter 1

TireGround

A repository for the code developed by Davide Stocco for his thesis.

Department of Industrial Engineering
Master Degree in Mechatronics Engineering

EN: Real-Time Computation of Tire/Road Contact using Tailored Algorithms

IT: Valutazione Real-Time del Contatto Pneumatico/Strada con Algoritmi Dedicati

Academic Year 2019 · 2020

Author: **Davide Stocco**

Supervisor & Co-supervisor: **Prof. Enrico Bertolazzi & Dr.Eng. Matteo Ragni**

MagicFormula tire model usage

1. Load .rdf file.

```
TireGround::RDF::MeshSurface Road(  
    "./file.rdf" // Path to the *.rdf file  
);
```

2. Initialize the MagicFormula tire model.

```
TireGround::Tire* TireSD = new TireGround::MagicFormula(  
    SectionWidth, // [m]  
    AspectRatio,  // [%]  
    RimDiameter,  // [in]  
    SwitchNumber  // Maximum RoadTriangles in the Tire Shadow (switch to sampling)  
);
```

3. Contact evaluation.

```
bool Out = TireSD->setup( Road,    // Road mesh  
                          TransfMat // 4x4 total transformation matrix  
);
```

4. Data extraction.

```
// Variable initialization (for real numbers)  
TireGround::vec3 N;  
TireGround::vec3 P;  
TireGround::real_type Friction;  
TireGround::real_type Rho;  
TireGround::real_type RhoDot;  
TireGround::real_type RelativeCamber;  
TireGround::real_type Area;  
TireGround::real_type Volume;  
  
// Data extraction (for real numbers)  
TireSD->getNormal(N);
```

```

TireSD->getMFpoint(P);
TireSD->getFriction(Friction);
TireSD->getRho(Rho);
TireSD->getRhoDot(PreviousRho,TimeStep,RhoDot);
TireSD->getRelativeCamber(RelativeCamber);
TireSD->getArea(Area);
TireSD->getVolume(Volume);

// Extract data stucture size
TireGround::int_type size = TireSD->getDisksNumber();

// Variable initialization (for vectors)
TireGround::row_vec3 NVec(size);
TireGround::row_vec3 PVec(size);
TireGround::row_vecN FrictionVec(size);
TireGround::row_vecN RhoVec(size);
TireGround::row_vecN RhoDotVec(size);
TireGround::row_vecN RelativeCamberVec(size);
TireGround::row_vecN AreaVec(size);
TireGround::row_vecN VolumeVec(size);

// Data extraction (for vectors)
TireSD->getNormal(NVec);
TireSD->getMFpoint(PVec);
TireSD->getFriction(FrictionVec);
TireSD->getRho(RhoVec);
TireSD->getRhoDot(PreviousRho,TimeStep,RhoDotVec);
TireSD->getRelativeCamber(RelativeCamberVec);
TireSD->getArea(AreaVec);
TireSD->getVolume(VolumeVec);

```

MultiDisk tire model usage

1. Load .rdf file.

```

TireGround::RDF::MeshSurface Road(
    "./file.rdf" // Path to the *.rdf file
);

```

2. Initialize the MultiDisk tire model:

(a) MultiDisk tire without sidewall radius (uniform cylinder).

```

TireGround::Tire* TireMD = new TireGround::MultiDisk(
    SectionWidth, // [m]
    AspectRatio, // [%]
    RimDiameter, // [in]
    PointsNumber, // Sampling points for each disk
    DisksNumber, // Disks number
    SwitchNumber // Maximum RoadTriangles in the Tire Shadow (switch to sampling)
);

```

(b) MultiDisk tire with sidewall radius (uniform cylinder with filleted sidewall edge).

```

TireGround::Tire* TireMD = new TireGround::MultiDisk(
    SectionWidth, // [m]
    AspectRatio, // [%]
    RimDiameter, // [in]
    SideRadius, // Sidewall radius [m]
    PointsNumber, // Sampling points for each disk
    DisksNumber, // Disks number
    SwitchNumber // Maximum RoadTriangles in the Tire Shadow (switch to sampling)
);

```

(c) MultiDisk tire with custom disks radius.

```

TireGround::Tire* TireMD = new TireGround::MultiDisk(
    SectionWidth, // [m]
    AspectRatio, // [%]
    RimDiameter, // [in]
    RadiusVec, // Disks radius vector [m]
    PointsNumber, // Sampling points for each disk
    SwitchNumber // Maximum RoadTriangles in the Tire Shadow (switch to sampling)
);

```

3. Contact evaluation.

```

bool Out = TireMD->setup( Road,      // Road mesh
                        TransfMat // 4x4 total transformation matrix
                        );

```

4. Data extraction for contact point(s).

```

// Variable initialization (for real numbers)
TireGround::vec3 N;
TireGround::vec3 P;
TireGround::real_type Friction;
TireGround::real_type Rho;
TireGround::real_type RhoDot;
TireGround::real_type RelativeCamber;
TireGround::real_type Area;
TireGround::real_type Volume;

// Data extraction (for real numbers)
TireMD->getNormal(N);
TireMD->getMFpoint(P);
TireMD->getFriction(Friction);
TireMD->getRho(Rho);
TireMD->getRhoDot(PreviousRho,TimeStep,RhoDot);
TireMD->getRelativeCamber(RelativeCamber);
TireMD->getArea(Area);
TireMD->getVolume(Volume);

// Extract data stucture size
TireGround::int_type size = TireSD->getDisksNumber();

// Variable initialization (for vectors)
TireGround::row_vec3 NVec(size);
TireGround::row_vec3 PVec(size);
TireGround::row_vecN FrictionVec(size);
TireGround::row_vecN RhoVec(size);
TireGround::row_vecN RhoDotVec(size);
TireGround::row_vecN RelativeCamberVec(size);
TireGround::row_vecN AreaVec(size);
TireGround::row_vecN VolumeVec(size);

// Data extraction (for vectors)
TireMD->getNormal(NVec);
TireMD->getMFpoint(PVec);
TireMD->getFriction(FrictionVec);
TireMD->getRho(RhoVec);
TireMD->getRhoDot(PreviousRho,TimeStep,RhoDotVec);
TireMD->getRelativeCamber(RelativeCamberVec);
TireMD->getArea(AreaVec);
TireMD->getVolume(VolumeVec);

```


Chapter 2

Namespace Index

2.1 Namespace List

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

TireGround	
Tire computations routines	11
TireGround::algorithms	
Algorithms for tire computations routine	13
TireGround::RDF	
RDF mesh computations routines	16
TireGround::RDF::algorithms	
Algorithms for RDF mesh computations routine	16

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

TireGround::RDF::BBox2D	19
TireGround::Disk	20
TireGround::ETRTO	25
TireGround::RDF::MeshSurface	37
TireGround::ReferenceFrame	54
TireGround::SamplingGrid	57
TireGround::Shadow	59
TicToc	60
TireGround::Tire	61
TireGround::MagicFormula	26
TireGround::MultiDisk	40
TireGround::RDF::Triangle3D	71
TireGround::RDF::TriangleRoad	75

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

TireGround::RDF::BBox2D	
2D Bounding Box class	19
TireGround::Disk	
Tire disk	20
TireGround::ETRTO	
Tire ETRTO denomination	25
TireGround::MagicFormula	
Pacejka MagicFormula contact model	26
TireGround::RDF::MeshSurface	
Mesh surface	37
TireGround::MultiDisk	
Multi-disk tire contact model	40
TireGround::ReferenceFrame	
Reference frame	54
TireGround::SamplingGrid	
Patch evaluation precision	57
TireGround::Shadow	
2D shadow (2D bounding box enhancement)	59
TicToc	60
TireGround::Tire	
Base class for Tire models	61
TireGround::RDF::Triangle3D	
3D triangle (pure geometrical description)	71
TireGround::RDF::TriangleRoad	
3D triangles for road representation	75

Chapter 5

Namespace Documentation

5.1 TireGround Namespace Reference

[Tire](#) computations routines.

Namespaces

- [algorithms](#)
Algorithms for tire computations routine.
- [RDF](#)
RDF mesh computations routines.

Classes

- class [Disk](#)
Tire disk.
- class [ETRTO](#)
Tire ETRTO denomination.
- class [MagicFormula](#)
Pacejka [MagicFormula](#) contact model.
- class [MultiDisk](#)
Multi-disk tire contact model.
- class [ReferenceFrame](#)
Reference frame.
- class [SamplingGrid](#)
Patch evaluation precision.
- class [Shadow](#)
2D shadow (2D bounding box enhancement)
- class [Tire](#)
Base class for [Tire](#) models.

Typedefs

- typedef double [real_type](#)
Real number type.
- typedef int [int_type](#)
Integer number type.
- typedef Eigen::Vector2i [vec2_int](#)
2D vector type of real integer type
- typedef Eigen::Vector2d [vec2](#)
2D vector type of real number type
- typedef Eigen::Vector3d [vec3](#)
3D vector type of real number type
- typedef Eigen::Vector4d [vec4](#)
4D vector type of real number type
- typedef Eigen::Matrix3d [mat3](#)
3x3 matrix type of real number type
- typedef Eigen::Matrix4d [mat4](#)
4x4 matrix type of real number type
- typedef Eigen::Matrix< [real_type](#), 1, Eigen::Dynamic > [row_vecN](#)
Row vector type real number type.
- typedef Eigen::Matrix< [real_type](#), Eigen::Dynamic, 1 > [col_vecN](#)
Column vector type real number type.
- typedef Eigen::Matrix< [real_type](#), Eigen::Dynamic, Eigen::Dynamic > [matN](#)
Matrix type of real number type.
- typedef Eigen::Matrix< [vec2](#), 1, Eigen::Dynamic > [row_vec2](#)
Row vector type of 2D vector.
- typedef Eigen::Matrix< [vec2](#), Eigen::Dynamic, 1 > [col_vec2](#)
Column vector type of 2D vector.
- typedef Eigen::Matrix< [vec2](#), Eigen::Dynamic, Eigen::Dynamic > [mat_vec2](#)
Matrix type of 2D vector.
- typedef Eigen::Matrix< [vec3](#), 1, Eigen::Dynamic > [row_vec3](#)
Row vector type of 3D vector.
- typedef Eigen::Matrix< [vec3](#), Eigen::Dynamic, 1 > [col_vec3](#)
Column vector type of 3D vector.
- typedef Eigen::Matrix< [vec3](#), Eigen::Dynamic, Eigen::Dynamic > [matN_vec3](#)
Matrix type of 3D vector.
- typedef Eigen::Matrix< [mat4](#), 1, Eigen::Dynamic > [row_mat4](#)
Matrix type of 4x4 matrix.
- typedef std::basic_ostream< char > [ostream_type](#)
Output stream type.

Variables

- [real_type](#) const [epsilon](#) = std::numeric_limits<[real_type](#)>::epsilon()
Epsilon type.

5.1.1 Detailed Description

Tire computations routines.

Typedefs for tire computations routine.

file: [PatchTire.hh](#)

file: [TireGround.hh](#)

5.2 TireGround::algorithms Namespace Reference

Algorithms for tire computations routine.

Functions

- [vec3 mean](#) ([row_vec3](#) const &Values)
Calculate arithmetic weighted mean for 3D vectors.
- [real_type weightedMean](#) ([row_vecN](#) const &Values, [row_vecN](#) const &Weights)
Calculate arithmetic weighted mean for real numbers.
- [vec3 weightedMean](#) ([row_vec3](#) const &Values, [row_vecN](#) const &Weights)
Calculate arithmetic weighted mean for 3D vectors.
- [bool intersectPointSegment](#) ([vec2](#) const &Point1, [vec2](#) const &Point2, [vec2](#) const &PointQ)
- [bool intersectRayPlane](#) ([vec3](#) const &planeN, [vec3](#) const &planeP, [vec3](#) const &RayPoint, [vec3](#) const &RayDirection, [vec3](#) &IntersectionPt)
Check if a segment hits a plane and find the intersection point.
- [void minmax_XY](#) ([row_vec3](#) const &Points, [vec2](#) &XYmin, [vec2](#) &XYmax)
Calculate mininum and maximum in XY plane for 3D vectors.
- [void minmax_XY](#) ([row_vec2](#) const &Points, [vec2](#) &XYmin, [vec2](#) &XYmax)
Calculate mininum and maximum in XY plane for 2D vectors.
- [real_type trapezoidArea](#) ([real_type](#) const Base2, [real_type](#) const Base1, [real_type](#) const Height)
Calculate area of a trapeziod [m²].

5.2.1 Detailed Description

Algorithms for tire computations routine.

5.2.2 Function Documentation

5.2.2.1 intersectPointSegment()

```
bool TireGround::algorithms::intersectPointSegment (  
    vec2 const & Point1,  
    vec2 const & Point2,  
    vec2 const & PointQ )
```

Check if a point lays inside or outside a line segment

Warning: The point query point must be on the same rect of the line segment!

Parameters

<i>Point1</i>	Line segment point 1
<i>Point2</i>	Line segment point 2
<i>PointQ</i>	Query point

5.2.2.2 intersectRayPlane()

```
bool TireGround::algorithms::intersectRayPlane (
    vec3 const & planeN,
    vec3 const & planeP,
    vec3 const & RayPoint,
    vec3 const & RayDirection,
    vec3 & IntersectionPt )
```

Check if a segment hits a plane and find the intersection point.

Parameters

<i>planeN</i>	Plane normal vector
<i>planeP</i>	Plane known point
<i>RayPoint</i>	Ray point
<i>RayDirection</i>	Ray direction
<i>IntersectionPt</i>	Intersection point

5.2.2.3 mean()

```
vec3 TireGround::algorithms::mean (
    row_vec3 const & Values )
```

Calculate arithmetic weighted mean for 3D vectors.

Parameters

<i>Values</i>	Values (3D vectors)
---------------	---------------------

5.2.2.4 minmax_XY() [1/2]

```
void TireGround::algorithms::minmax_XY (
    row_vec3 const & Points,
    vec2 & XYmin,
    vec2 & XYmax )
```

Calculate minimum and maximum in XY plane for 3D vectors.

Parameters

<i>Points</i>	3D points vector
<i>XYmin</i>	Minimum (X , Y) values
<i>XYmax</i>	Maximum (X , Y) values

5.2 TireGround::algorithms Namespace Reference

5.2.2.5 minmax_XY() [2/2]

```
void TireGround::algorithms::minmax_XY (
    row_vec2 const & Points,
    vec2 & XYmin,
    vec2 & XYmax )
```

Calculate minumum and maximum in XY plane for 2D vectors.

Parameters

<i>Points</i>	2D points vector
<i>XYmin</i>	Minimum (X , Y) values
<i>XYmax</i>	Maximum (X , Y) values

5.2.2.6 trapezoidArea()

```
real_type TireGround::algorithms::trapezoidArea (
    real_type const Base2,
    real_type const Base1,
    real_type const Height ) [inline]
```

Calculate area of a trapeziod [m^2].

Parameters

<i>Base2</i>	Base 1
<i>Base1</i>	Base 2
<i>Height</i>	Heigh

5.2.2.7 weightedMean() [1/2]

```
real_type TireGround::algorithms::weightedMean (
    row_vecN const & Values,
    row_vecN const & Weights )
```

Calculate arithmetic weighted mean for real numbers.

Parameters

<i>Values</i>	Values (real numbers)
<i>Weights</i>	Weights (real numbers)

5.2.2.8 weightedMean() [2/2]

```
vec3 TireGround::algorithms::weightedMean (
    row_vec3 const & Values,
    row_vecN const & Weights )
```

Calculate arithmetic weighted mean for 3D vectors.

Parameters

<i>Values</i>	Values (3D vectors)
<i>Weights</i>	Weights (real numbers)

5.3 TireGround::RDF Namespace Reference

[RDF](#) mesh computations routines.

Namespaces

- [algorithms](#)
Algorithms for [RDF](#) mesh computations routine.

Classes

- class [BBox2D](#)
2D Bounding Box class
- class [MeshSurface](#)
Mesh surface.
- class [Triangle3D](#)
3D triangle (pure geometrical description)
- class [TriangleRoad](#)
3D triangles for road representation

Typedefs

- typedef std::shared_ptr< [TriangleRoad](#) > [TriangleRoad_ptr](#)
Shared pointer to [TriangleRoad](#) object.
- typedef std::vector< [TriangleRoad_ptr](#) > [TriangleRoad_list](#)
Vector of shared pointers to [TriangleRoad](#) objects.

5.3.1 Detailed Description

[RDF](#) mesh computations routines.

5.4 TireGround::RDF::algorithms Namespace Reference

Algorithms for [RDF](#) mesh computations routine.

Functions

- void [split](#) (std::string const &in, std::vector< std::string > &out, std::string const &token)
Split a string into a string array at a given token.
- std::string [tail](#) (std::string const &in)
Get tail of string after first token and possibly following spaces.
- std::string [firstToken](#) (std::string const &in)
Get first token of string.
- template<typename T >
T const & [getElement](#) (std::vector< T > const &elements, std::string const &index)
Get element at given index position.

5.4.1 Detailed Description

Algorithms for [RDF](#) mesh computations routine.

5.4.2 Function Documentation

5.4.2.1 firstToken()

```
std::string TireGround::RDF::algorithms::firstToken (
    std::string const & in )
```

Get first token of string.

Parameters

<i>in</i>	Input string
-----------	--------------

5.4.2.2 getElement()

```
template<typename T >
T const& TireGround::RDF::algorithms::getElement (
    std::vector< T > const & elements,
    std::string const & index )
```

Get element at given index position.

Parameters

<i>elements</i>	Elements vector
<i>index</i>	Index position

5.4.2.3 split()

```
void TireGround::RDF::algorithms::split (
    std::string const & in,
    std::vector< std::string > & out,
    std::string const & token )
```

Split a string into a string array at a given token.

Parameters

<i>in</i>	Input string
<i>out</i>	Output string vector
<i>token</i>	Token

5.4.2.4 tail()

```
std::string TireGround::RDF::algorithms::tail (
```

```
std::string const & in )
```

Get tail of string after first token and possibly following spaces.

Parameters

<i>in</i>	Input string
-----------	--------------

Chapter 6

Class Documentation

6.1 TireGround::RDF::BBox2D Class Reference

2D Bounding Box class

```
#include <RoadRDF.hh>
```

Public Member Functions

- [BBox2D](#) ()
Default constructor.
- [BBox2D](#) ([vec3](#) const Vertices[3])
Variable set constructor.
- void [setXmin](#) ([real_type](#) const _Xmin)
Set X_{min} shadow domain.
- void [setYmin](#) ([real_type](#) const _Ymin)
Set Y_{min} shadow domain.
- void [setXmax](#) ([real_type](#) const _Xmax)
Set X_{max} shadow domain.
- void [setYmax](#) ([real_type](#) const _Ymax)
Set Y_{max} shadow domain.
- [real_type](#) [getXmin](#) (void) const
Get X_{min} shadow domain.
- [real_type](#) [getYmin](#) (void) const
Get Y_{min} shadow domain.
- [real_type](#) [getXmax](#) (void) const
Get X_{max} shadow domain.
- [real_type](#) [getYmax](#) (void) const
Get Y_{max} shadow domain.
- void [clear](#) (void)
Clear the bounding box domain.
- void [print](#) ([ostream_type](#) &stream) const
Print bounding box domain.
- void [updateBBox2D](#) ([vec3](#) const Vertices[3])
Update the bounding box domain with three input vertices.

6.1.1 Detailed Description

2D Bounding Box class

6.1.2 Constructor & Destructor Documentation

6.1.2.1 BBox2D()

```
TireGround::RDF::BBox2D::BBox2D (
    vec3 const Vertices[3] ) [inline]
```

Variable set constructor.

Parameters

<i>Vertices</i>	Vertices reference vector
-----------------	---------------------------

6.1.3 Member Function Documentation

6.1.3.1 print()

```
void TireGround::RDF::BBox2D::print (
    ostream_type & stream ) const [inline]
```

Print bounding box domain.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

6.1.3.2 updateBBox2D()

```
void TireGround::RDF::BBox2D::updateBBox2D (
    vec3 const Vertices[3] )
```

Update the bounding box domain with three input vertices.

Parameters

<i>Vertices</i>	Vertices reference vector
-----------------	---------------------------

The documentation for this class was generated from the following file:

- include/RoadRDF.hh

6.2 TireGround::Disk Class Reference

[Tire](#) disk.

```
#include <PatchTire.hh>
```

Public Member Functions

- [Disk](#) ([Disk](#) &&)=default
Enable && operator.
- [Disk](#) ()
Default constructor.
- [Disk](#) ([vec2](#) const &_OriginXZ, [real_type](#) _OffsetY, [real_type](#) _Radius)
Variable set constructor.
- void [set](#) ([Disk](#) const &in)
Copy the [Disk](#) object.
- void [setOriginXZ](#) ([vec2](#) const &_OriginXZ)
Set origin on XZ plane.
- [vec2](#) const & [getOriginXZ](#) (void) const
Get origin vector XZ-axes coordinates.
- [vec3](#) [getOriginXYZ](#) (void) const
Get origin vector XYZ-axes coordinates.
- [real_type](#) [getOffsetY](#) (void) const
Get origin Y-axis coordinate.
- [real_type](#) [getRadius](#) (void) const
Get [Disk](#) radius.
- void [contactTriangles](#) ([RDF::TriangleRoad_list](#) const &TriList, [ReferenceFrame](#) const &RF, [vec3](#) &Normal, [real_type](#) &Friction, [real_type](#) &Area) const
- void [contactPlane](#) ([vec3](#) const &Normal, [vec3](#) const &Point, [ReferenceFrame](#) const &RF, [real_type](#) &Area) const
- void [pointOnDisk](#) ([vec3](#) const &Normal, [ReferenceFrame](#) const &RF, [vec3](#) &DiskPoint, [vec3](#) &NormalOnDisk) const
Get the points on [Disk](#) the circumference and on a given plane.
- [real_type](#) [segmentArea](#) ([real_type](#) const Length) const
- bool [isPointInside](#) ([vec2](#) const &Point) const
Check if a point in [Disk](#) reference frame is inside or outside the [Disk](#).
- [real_type](#) [y](#) ([real_type](#) const x) const
Evaluate Y at a query X value on the lower side [Disk](#) circumference.
- [real_type](#) [segmentLength](#) ([vec2](#) const Point1, [vec2](#) const Point2) const
Evaluate a generic segment length given 2 points on the [Disk](#) circumference.
- [int_type](#) [intersectSegment](#) ([vec2](#) const &Point1, [vec2](#) const &Point2, [vec2](#) &Intersect1, [vec2](#) &Intersect2) const
- bool [intersectPlane](#) ([vec3](#) const &Plane_Normal, [vec3](#) const &Plane_Point, [vec3](#) &Line_↵ Direction, [vec3](#) &Line_Point) const
- [real_type](#) [getLineArea](#) ([vec2](#) const &Point1_XZ, [vec2](#) const &Point2_XZ) const
Get a two points line segment area [m²] (as output) inside the [Disk](#).

6.2.1 Detailed Description

[Tire](#) disk.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Disk()

```
TireGround::Disk::Disk (
    vec2 const & _OriginXZ,
    real_type _OffsetY,
    real_type _Radius ) [inline]
```

Variable set constructor.

Parameters

<i>_OriginXZ</i>	(X_0, Z_0) origin coordinate
<i>_OffsetY</i>	Y_0 origin coordinate (offset from center)
<i>_Radius</i>	Radius

6.2.3 Member Function Documentation

6.2.3.1 contactPlane()

```
void TireGround::Disk::contactPlane (
    vec3 const & Normal,
    vec3 const & Point,
    ReferenceFrame const & RF,
    real_type & Area ) const
```

Get the contact area [m^2] inside the single [Disk](#) given a plane in absolute reference frame

Parameters

<i>Normal</i>	Plane normal in absolute reference frame
<i>Point</i>	Plane point in absolute reference frame
<i>RF</i>	Tire ReferenceFrame
<i>Area</i>	Contact area [m^2]

6.2.3.2 contactTriangles()

```
void TireGround::Disk::contactTriangles (
    RDF::TriangleRoad_list const & TriList,
    ReferenceFrame const & RF,
    vec3 & Normal,
    real_type & Friction,
    real_type & Area ) const
```

Get area weighted mean road normal versor, area weighted mean friction and contact area [m^2] inside the single [Disk](#) of segments described by the intersection of triangles on XZ -plane

Parameters

<i>TriList</i>	Shadow / MeshSurface intersected triangles
<i>RF</i>	Tire ReferenceFrame
<i>Normal</i>	Area weighted mean road normal versor

6.2 TireGround::Disk Class Reference

Parameters

<i>Friction</i>	Area weighted mean contact friction
<i>Area</i>	Contact area [m^2]

6.2.3.3 getLineArea()

```
real_type TireGround::Disk::getLineArea (
    vec2 const & Point1_XZ,
    vec2 const & Point2_XZ ) const
```

Get a two points line segment area [m^2] (as ouput) inside the [Disk](#).

Parameters

<i>Point1_XZ</i>	Point 1 in Disk reference frame
<i>Point2_XZ</i>	Point 2 in Disk reference frame

6.2.3.4 intersectPlane()

```
bool TireGround::Disk::intersectPlane (
    vec3 const & Plane_Normal,
    vec3 const & Plane_Point,
    vec3 & Line_Direction,
    vec3 & Line_Point ) const
```

Check if two plane intersects and find the intersecting rect given two points in [Disk](#) reference frame

Parameters

<i>Plane_Normal</i>	Plane normal vector in Disk reference frame
<i>Plane_Point</i>	Plane known point in Disk reference frame
<i>Line_Direction</i>	Rect direction vector in Disk reference frame
<i>Line_Point</i>	Plane known point in Disk reference frame

6.2.3.5 intersectSegment()

```
int_type TireGround::Disk::intersectSegment (
    vec2 const & Point1,
    vec2 const & Point2,
    vec2 & Intersect1,
    vec2 & Intersect2 ) const
```

Find the intersection points between the [Disk](#) and a two points line segment in [Disk](#) reference frame (output integer gives number of intersection points)

Parameters

<i>Point1</i>	Line segment point 1 in Disk reference frame
---------------	--

Parameters

<i>Point2</i>	Line segment point 2 in Disk reference frame
<i>Intersect1</i>	Intersection point 1 in Disk reference frame
<i>Intersect2</i>	Intersection point 2 in Disk reference frame

6.2.3.6 isPointInside()

```
bool TireGround::Disk::isPointInside (
    vec2 const & Point ) const
```

Check if a point in [Disk](#) reference frame is inside or outside the [Disk](#).

Parameters

<i>Point</i>	Query point in Disk reference frame
--------------	---

6.2.3.7 segmentArea()

```
real\_type TireGround::Disk::segmentArea (
    real\_type const Length ) const [inline]
```

Get the contact patch area under the intersection plane in absolute reference frame [m^2]

Parameters

<i>Length</i>	Chord length
---------------	--------------

6.2.3.8 segmentLength()

```
real\_type TireGround::Disk::segmentLength (
    vec2 const Point1,
    vec2 const Point2 ) const [inline]
```

Evaluate a generic segment length given 2 points on the [Disk](#) circumference.

Parameters

<i>Point1</i>	Point 1
<i>Point2</i>	Point 2

6.2.3.9 set()

```
void TireGround::Disk::set (
    Disk const & in ) [inline]
```

Copy the [Disk](#) object.

Parameters

<i>in</i>	Disk object to be copied
-----------	--

6.2.3.10 setOriginXZ()

```
void TireGround::Disk::setOriginXZ (
    vec2 const & _OriginXZ ) [inline]
```

Set origin on XZ plane.

Parameters

<i>_OriginXZ</i>	New origin on XZ plane
------------------	--------------------------

6.2.3.11 y()

```
real\_type TireGround::Disk::y (
    real\_type const x ) const [inline]
```

Evaluate Y at a query X value on the lower side [Disk](#) circumference.

Parameters

<i>x</i>	Query X value
----------	-----------------

The documentation for this class was generated from the following file:

- include/PatchTire.hh

6.3 TireGround::ETRTO Class Reference

[Tire ETRTO](#) denomination.

```
#include <PatchTire.hh>
```

Public Member Functions

- [ETRTO](#) ()
Default constructor.
- [ETRTO](#) ([real_type](#) _SectionWidth, [real_type](#) _AspectRatio, [real_type](#) _RimDiameter)
Variable set constructor.
- [real_type](#) getSidewallHeight (void) const
Get sidewall height [m].
- [real_type](#) getTireDiameter (void) const
Get external tire diameter [m].
- [real_type](#) getTireRadius (void) const
Get external tire radius [m].
- [real_type](#) getSectionWidth (void) const

Get section width [m].

- void `print (ostream_type &stream) const`

Display tire data.

6.3.1 Detailed Description

`Tire ETRTO` denomination.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 ETRTO()

```
TireGround::ETRTO::ETRTO (
    real_type _SectionWidth,
    real_type _AspectRatio,
    real_type _RimDiameter ) [inline]
```

Variable set constructor.

Parameters

<code>_SectionWidth</code>	<code>Tire</code> section width [<i>m</i>]
<code>_AspectRatio</code>	<code>Tire</code> aspect ratio [%]
<code>_RimDiameter</code>	Rim diameter [<i>in</i>]

6.3.3 Member Function Documentation

6.3.3.1 print()

```
void TireGround::ETRTO::print (
    ostream_type & stream ) const [inline]
```

Display tire data.

Parameters

<code>stream</code>	Output stream type
---------------------	--------------------

The documentation for this class was generated from the following file:

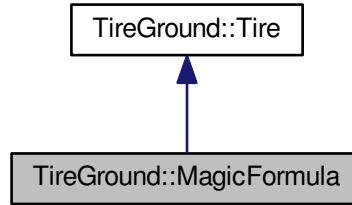
- `include/PatchTire.hh`

6.4 TireGround::MagicFormula Class Reference

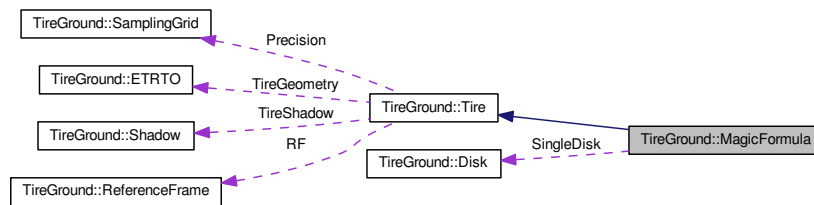
Pacejka `MagicFormula` contact model.

```
#include <PatchTire.hh>
```

Inheritance diagram for TireGround::MagicFormula:



Collaboration diagram for TireGround::MagicFormula:



Public Member Functions

- `~MagicFormula ()`
Default destructor.
- `MagicFormula (real_type const SectionWidth, real_type const AspectRatio, real_type const RimDiameter, int_type const SwitchN)`
Variable set constructor.
- `void getNormal (vec3 &_Normal) const override`
Get contact normal versor.
- `void getNormal (row_vec3 &_Normal) const override`
Get contact normal versors vector.
- `void getMFpoint (vec3 &_DiskPoint) const override`
Get Magic Formula contact point.
- `void getMFpoint (row_vec3 &_DiskPoint) const override`
Get Magic Formula contact point vector.
- `void getFriction (real_type &_Friction) const override`
Get contact point friction.
- `void getFriction (row_vecN &_Friction) const override`
Get contact point friction vector.
- `void getMFpointRF (mat4 &PointRF) const override`
Get Magic Formula contact point reference frame with 4x4 transformation matrix.

- void `getMFpointRF` (`row_mat4` & `_MFpointRF`) const override
Get Magic Formula contact point reference frame vector with 4x4 transformation matrix.
- void `getRho` (`real_type` & `Rho`, `real_type` & `RhoDot`, `real_type` const `RhoOld`, `real_type` const `Time`) const override
- void `getRho` (`row_vecN` & `Rho`, `row_vecN` & `RhoDot`, `row_vecN` const `RhoOld`, `real_type` const `Time`) const override
- void `getArea` (`real_type` & `_Area`) const override
Get approximated contact area on `Disk` plane [m^2].
- void `getArea` (`row_vecN` & `_Area`) const override
Get approximated contact area vector on `Disk` plane [m^2].
- void `getVolume` (`real_type` & `_Volume`) const override
Get approximated contact volume [m^3].
- void `getVolume` (`row_vecN` & `Volume`) const override
Get approximated contact volume vector [m^3].
- bool `setup` (`RDF::MeshSurface` & `Mesh`, `mat4` const & `TM`) override
Update current tire position and find contact parameters.
- void `setup` (`vec3` const & `Plane_Normal`, `vec3` const & `Plane_Point`, `real_type` const `Plane_↔Friction`, `mat4` const & `TM`) override
- void `print` (`ostream_type` & `stream`) const override
Print contact parameters.
- void `printETRTOGeometry` (`ostream_type` & `stream`) const
Display `Tire ETRTO` geometry data.
- `G2lib::AABBtree::PtrAABB` const `getAABBtree` (void) const
Get total `Tire Shadow` `G2Lib::AABBtree` (3D projection on ground)
- `G2lib::AABBtree::PtrAABB` const `getUpperSideAABBtree` (void) const
Get upper side `Tire Shadow` `G2Lib::AABBtree` (3D projection on ground)
- `G2lib::AABBtree::PtrAABB` const `getLowerSideAABBtree` (void) const
Get lower side `Tire Shadow` `G2Lib::AABBtree` (3D projection on ground)
- void `setReferenceFrame` (`ReferenceFrame` const & `_RF`)
- `ReferenceFrame` const & `getReferenceFrame` (void) const
Get tire `ReferenceFrame` object.
- void `setOrigin` (`vec3` const & `Origin`)
Set a new tire origin.
- void `setRotationMatrix` (`mat3` const & `RotationMatrix`)
- void `setTotalTransformationMatrix` (`mat4` const & `TM`)
- `real_type` `getEulerAngleX` (void) const
- `real_type` `getEulerAngleY` (void) const
- `real_type` `getEulerAngleZ` (void) const
- void `getRelativeCamber` (`real_type` & `RelativeCamber`) const
Get relative camber angle [rad].
- `int_type` `getDisksNumber` (void) const
Dimension of the contact points data structure (disks number)

Protected Member Functions

- `MagicFormula` (`MagicFormula` const &)=delete
Deleted copy constructor.
- `MagicFormula` const & `operator=` (`MagicFormula` const &)=delete
Deleted copy operator.
- void `evaluateContact` (`RDF::TriangleRoad_list` const & `TriList`) override

Evaluate contact with RoadTriangles.

- void [fourPointsSampling](#) ([RDF::TriangleRoad_list](#) const &TriList, [vec3](#) &P_star)
Perform triangles sampling on 4 points at $\pm 0.1 \cdot R$ along X and $\pm 0.3 \cdot W$ along Y .
- bool [pointSampling](#) ([RDF::TriangleRoad_list](#) const &TriList, [vec3](#) const &RayOrigin, [vec3](#) const &RayDirection, [vec3](#) &SampledPt, [real_type](#) &TriFriction=quietNaN, [vec3](#) &TriNormal=[vec3_NaN](#)) const
Perform one point sampling (ray-triangle intersection)

Protected Attributes

- [Disk](#) [SingleDisk](#)
Single Disk.
- [vec3](#) [Normal](#)
Contact normal versor.
- [vec3](#) [MeshPoint](#)
Contact point on Mesh (for Magic Formula)
- [vec3](#) [DiskPoint](#)
Contact point on undeformed [Disk](#) circumference (not for Magic Formula)
- [real_type](#) [Friction](#)
Contact friction.
- [real_type](#) [Area](#)
Contact area [m^2].
- [SamplingGrid](#) [Precision](#)
Contact patch evaluating precision.
- [ETRTO](#) [TireGeometry](#)
Tire ETRTO denomination.
- [ReferenceFrame](#) [RF](#)
ReferenceFrame.
- [Shadow](#) [TireShadow](#)
Tire shadow.

6.4.1 Detailed Description

Pacejka [MagicFormula](#) contact model.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 MagicFormula()

```
TireGround::MagicFormula::MagicFormula (
    real\_type const SectionWidth,
    real\_type const AspectRatio,
    real\_type const RimDiameter,
    int\_type const SwitchN ) [inline]
```

Variable set constructor.

Parameters

<i>SectionWidth</i>	Tire section width [<i>m</i>]
<i>AspectRatio</i>	Tire aspect ratio [%]
<i>RimDiameter</i>	Rim diameter [<i>in</i>]
<i>SwitchN</i>	Maximum RoadTriangles in the Tire Shadow (switch to sampling)

6.4.3 Member Function Documentation

6.4.3.1 evaluateContact()

```
void TireGround::MagicFormula::evaluateContact (
    RDF::TriangleRoad_list const & TriList ) [override], [protected], [virtual]
```

Evaluate contact with RoadTriangles.

Parameters

<i>TriList</i>	Shadow/MeshSurface intersected triangles
----------------	--

Implements [TireGround::Tire](#).

6.4.3.2 fourPointsSampling()

```
void TireGround::MagicFormula::fourPointsSampling (
    RDF::TriangleRoad_list const & TriList,
    vec3 & P_star ) [protected]
```

Perform triangles sampling on 4 points at $\pm 0.1 \cdot R$ along X and $\pm 0.3 \cdot W$ along Y .

Parameters

<i>TriList</i>	Shadow/MeshSurface intersected triangles
----------------	--

6.4.3.3 getArea() [1/2]

```
void TireGround::MagicFormula::getArea (
    real_type & _Area ) const [inline], [override], [virtual]
```

Get approximated contact area on [Disk](#) plane [m^2].

Parameters

<i>_Area</i>	Contact area [m^2]
--------------	------------------------

Implements [TireGround::Tire](#).

6.4.3.4 getArea() [2/2]

```
void TireGround::MagicFormula::getArea (
    row_vecN & _Area ) const [inline], [override], [virtual]
```

Get approximated contact area vector on [Disk](#) plane [m^2].

Parameters

<i>_Area</i>	Contact area vector [m^2]
--------------	-------------------------------

Implements [TireGround::Tire](#).

6.4.3.5 getEulerAngleX()

```
real_type TireGround::Tire::getEulerAngleX (
    void ) const [inline], [inherited]
```

Get current Euler angles [*rad*] for *X*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.4.3.6 getEulerAngleY()

```
real_type TireGround::Tire::getEulerAngleY (
    void ) const [inline], [inherited]
```

Get current Euler angles [*rad*] for *Y*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.4.3.7 getEulerAngleZ()

```
real_type TireGround::Tire::getEulerAngleZ (
    void ) const [inline], [inherited]
```

Get current Euler angles [*rad*] for *Z*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.4.3.8 getFriction() [1/2]

```
void TireGround::MagicFormula::getFriction (
    real_type & _Friction ) const [inline], [override], [virtual]
```

Get contact point friction.

Parameters

<i>_Friction</i>	Contact point friction
------------------	------------------------

Implements [TireGround::Tire](#).

6.4.3.9 getFriction() [2/2]

```
void TireGround::MagicFormula::getFriction (
    row_vecN & _Friction ) const [inline], [override], [virtual]
```

Get contact point friction vector.

Parameters

<i>_Friction</i>	Contact point friction vector
------------------	-------------------------------

Implements [TireGround::Tire](#).

6.4.3.10 getMFpoint() [1/2]

```
void TireGround::MagicFormula::getMFpoint (
    vec3 & _DiskPoint ) const [inline], [override], [virtual]
```

Get Magic Formula contact point.

Parameters

<i>_DiskPoint</i>	Magic Formula contact point
-------------------	-----------------------------

Implements [TireGround::Tire](#).

6.4.3.11 getMFpoint() [2/2]

```
void TireGround::MagicFormula::getMFpoint (
    row_vec3 & _DiskPoint ) const [inline], [override], [virtual]
```

Get Magic Formula contact point vector.

Parameters

<i>_DiskPoint</i>	Contact point vector on Disk
-------------------	--

Implements [TireGround::Tire](#).

6.4.3.12 getMFpointRF() [1/2]

```
void TireGround::MagicFormula::getMFpointRF (
    mat4 & PointRF ) const [override], [virtual]
```

Get Magic Formula contact point reference frame with 4x4 transformation matrix.

Parameters

<i>PointRF</i>	Magic Formula contact point reference frame
----------------	---

Implements [TireGround::Tire](#).

6.4.3.13 getMFpointRF() [2/2]

```
void TireGround::MagicFormula::getMFpointRF (
    row_mat4 & _MFpointRF ) const [inline], [override], [virtual]
```

Get Magic Formula contact point reference frame vector with 4x4 transformation matrix.

Parameters

<i>_MFpointRF</i>	Magic Formula ontact point reference frames vector
-------------------	--

Implements [TireGround::Tire](#).

6.4.3.14 getNormal() [1/2]

```
void TireGround::MagicFormula::getNormal (
    vec3 & _Normal ) const [inline], [override], [virtual]
```


Get contact normal versor.

Parameters

<i>_Normal</i>	Contact point normal versor
----------------	-----------------------------

Implements [TireGround::Tire](#).

6.4.3.15 getNormal() [2/2]

```
void TireGround::MagicFormula::getNormal (
    row_vec3 & _Normal ) const [inline], [override], [virtual]
```

Get contact normal versors vector.

Parameters

<i>_Normal</i>	Contact point normal direction vector
----------------	---------------------------------------

Implements [TireGround::Tire](#).

6.4.3.16 getRelativeCamber()

```
void TireGround::Tire::getRelativeCamber (
    real_type & RelativeCamber ) const [inherited]
```

Get relative camber angle [*rad*].

Parameters

<i>RelativeCamber</i>	Relative camber angle
-----------------------	-----------------------

6.4.3.17 getRho() [1/2]

```
void TireGround::MagicFormula::getRho (
    real_type & Rho,
    real_type & RhoDot,
    real_type const RhoOld,
    real_type const Time ) const [override], [virtual]
```

Get contact depth at center point [*m*] and it time derivative [*m/s*]
Warning: (if negative the tire does not touch the ground)!

Parameters

<i>Rho</i>	Depth at center point [<i>m/s</i>]
<i>RhoDot</i>	Contact depth derivative [<i>m/s</i>]
<i>RhoOld</i>	Previous time step Rho [<i>m</i>]
<i>Time</i>	Time step [<i>s</i>]

Implements [TireGround::Tire](#).

6.4.3.18 getRho() [2/2]

```
void TireGround::MagicFormula::getRho (
    row_vecN & Rho,
    row_vecN & RhoDot,
    row_vecN const RhoOld,
    real_type const Time ) const [inline], [override], [virtual]
```

Get contact depth matrix [m] and it time derivatives [m/s]

Warning: (if negative the tire does not touch the ground)!

Parameters

<i>Rho</i>	Depth matrix [m/s]
<i>RhoDot</i>	Contact depth derivative matrix [m/s]
<i>RhoOld</i>	Previous time step Rho matrix [m]
<i>Time</i>	Time step [s]

Implements [TireGround::Tire](#).

6.4.3.19 getVolume() [1/2]

```
void TireGround::MagicFormula::getVolume (
    real_type & _Volume ) const [inline], [override], [virtual]
```

Get approximated contact volume [m^3].

Parameters

<i>_Volume</i>	Contact volume [m^3]
----------------	--------------------------

Implements [TireGround::Tire](#).

6.4.3.20 getVolume() [2/2]

```
void TireGround::MagicFormula::getVolume (
    row_vecN & Volume ) const [inline], [override], [virtual]
```

Get approximated contact volume vector [m^3].

Parameters

<i>Volume</i>	Contact volume vector [m^3]
---------------	---------------------------------

Implements [TireGround::Tire](#).

6.4.3.21 pointSampling()

```
bool TireGround::Tire::pointSampling (
```

6.4 TireGround::MagicFormula Class Reference

```
RDF::TriangleRoad_list const & TriList,  
vec3 const & RayOrigin,  
vec3 const & RayDirection,  
vec3 & SampledPt,  
real_type & TriFriction = quietNaN,  
vec3 & TriNormal = vec3_NaN ) const [protected], [inherited]
```

Perform one point sampling (ray-triangle intersection)

Parameters

<i>TriList</i>	Shadow/MeshSurface intersected triangles
<i>RayOrigin</i>	Ray origin
<i>RayDirection</i>	Ray direction
<i>SampledPt</i>	Intersection point
<i>TriFriction</i>	Intersected triangle friction
<i>TriNormal</i>	Intersected triangle normal

6.4.3.22 print()

```
void TireGround::MagicFormula::print (  
    ostream_type & stream ) const [override], [virtual]
```

Print contact parameters.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

Implements [TireGround::Tire](#).

6.4.3.23 printETRTOGeometry()

```
void TireGround::Tire::printETRTOGeometry (  
    ostream_type & stream ) const [inline], [inherited]
```

Display [Tire ETRTO](#) geometry data.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

6.4.3.24 setOrigin()

```
void TireGround::Tire::setOrigin (  
    vec3 const & Origin ) [inline], [inherited]
```

Set a new tire origin.

Parameters

<i>Origin</i>	Tire origin
---------------	-----------------------------

6.4.3.25 setReferenceFrame()

```
void TireGround::Tire::setReferenceFrame (
    ReferenceFrame const & _RF ) [inline], [inherited]
```

Copy the tire [ReferenceFrame](#) object

Warning: Rotation matrix must be orthonormal!

Parameters

<i>_RF</i>	ReferenceFrame object to be copied
------------	--

6.4.3.26 setRotationMatrix()

```
void TireGround::Tire::setRotationMatrix (
    mat3 const & RotationMatrix ) [inline], [inherited]
```

Set a new 3x3 rotation matrix

Warning: Rotation matrix must be orthonormal!

Parameters

<i>RotationMatrix</i>	Rotation matrix
-----------------------	-----------------

6.4.3.27 setTotalTransformationMatrix()

```
void TireGround::Tire::setTotalTransformationMatrix (
    mat4 const & TM ) [inline], [inherited]
```

Set 4x4 total transformation matrix

Warning: Rotation matrix must be orthonormal!

Parameters

<i>TM</i>	4x4 total transformation matrix
-----------	---------------------------------

6.4.3.28 setup() [1/2]

```
bool TireGround::MagicFormula::setup (
    RDF::MeshSurface & Mesh,
    mat4 const & TM ) [override], [virtual]
```

Update current tire position and find contact parameters.

Parameters

<i>Mesh</i>	MeshSurface object (road)
<i>TM</i>	4x4 total transformation matrix

Implements [TireGround::Tire](#).

6.4.3.29 setup() [2/2]

```
void TireGround::MagicFormula::setup (  
    vec3 const & Plane_Normal,  
    vec3 const & Plane_Point,  
    real_type const Plane_Friction,  
    mat4 const & TM ) [override], [virtual]
```

Update current tire position and find contact parameters with external plane

Parameters

<i>Plane_Normal</i>	Plane normal vector
<i>Plane_Point</i>	Plane known point
<i>Plane_Friction</i>	Friction on plane
<i>TM</i>	4x4 total transformation matrix

Implements [TireGround::Tire](#).

The documentation for this class was generated from the following file:

- include/PatchTire.hh

6.5 TireGround::RDF::MeshSurface Class Reference

Mesh surface.

```
#include <RoadRDF.hh>
```

Public Member Functions

- [MeshSurface](#) ()
Default set constructor.
- [MeshSurface](#) ([TriangleRoad_list](#) const &_PtrTriangleVec)
Variable set constructor.
- [MeshSurface](#) (std::string const &Path)
Variable set constructor.
- [TriangleRoad_list](#) const & [getTrianglesList](#) (void) const
Get all triangles inside the mesh as a vector.
- [TriangleRoad_ptr](#) const [getTriangle](#) (unsigned i) const
Get i-th TriangleRoad.
- G2lib::AABBtree::PtrAABB const [getAABBPtr](#) (void) const
Get AABBtree object.
- void [printData](#) (std::string const &FileName) const

Print data in file.

- `std::vector< G2lib::BBox::PtrBBox > const & getPtrBBoxList () const`

Get the mesh G2lib bounding boxes pointers vector.

- `void set (MeshSurface const &in)`

Copy the MeshSurface object.

- `bool LoadFile (std::string const &Path)`

Load the RDF model and print information on a file.

- `bool intersectAABBtree (G2lib::AABBtree::PtrAABB const &AABBTreePtr, RDF::Triangle↵
Road_list &TrianglesList) const`

Intersect the mesh AABB tree with an external AABB tree.

- `bool intersectBBox (std::vector< G2lib::BBox::PtrBBox > const &BBoxPtr, RDF::Triangle↵
Road_list &TrianglesList) const`

Update the mesh AABBtree with an external G2lib::BBox object pointer vector.

6.5.1 Detailed Description

Mesh surface.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 MeshSurface() [1/2]

```
TireGround::RDF::MeshSurface::MeshSurface (
    TriangleRoad_list const & _PtrTriangleVec ) [inline]
```

Variable set constructor.

Parameters

<i>_PtrTriangleVec</i>	Road triangles pointer vector list
------------------------	------------------------------------

6.5.2.2 MeshSurface() [2/2]

```
TireGround::RDF::MeshSurface::MeshSurface (
    std::string const & Path ) [inline]
```

Variable set constructor.

Parameters

<i>Path</i>	Path to the RDF file
-------------	----------------------

6.5.3 Member Function Documentation

6.5.3.1 intersectAABBtree()

```
bool TireGround::RDF::MeshSurface::intersectAABBtree (
    G2lib::AABBtree::PtrAABB const & AABBTreePtr,
    RDF::TriangleRoad_list & TrianglesList ) const
```

Intersect the mesh AABB tree with an external AABB tree.

Parameters

<i>AABBTreePtr</i>	External AABBtree object pointer
<i>TrianglesList</i>	Intersected TriangleRoad vector list

6.5.3.2 intersectBBox()

```
bool TireGround::RDF::MeshSurface::intersectBBox (
    std::vector< G2lib::BBox::PtrBBox > const & BBoxPtr,
    RDF::TriangleRoad_list & TrianglesList ) const
```

Update the mesh AABBtree with an external G2lib::BBox object pointer vector.

Parameters

<i>BBoxPtr</i>	External G2lib::BBox object pointer vector
<i>TrianglesList</i>	Intersected TriangleRoad vector list

6.5.3.3 LoadFile()

```
bool TireGround::RDF::MeshSurface::LoadFile (
    std::string const & Path )
```

Load the [RDF](#) model and print information on a file.

Parameters

<i>Path</i>	Path to the RDF file
-------------	--------------------------------------

6.5.3.4 printData()

```
void TireGround::RDF::MeshSurface::printData (
    std::string const & FileName ) const
```

Print data in file.

Parameters

<i>FileName</i>	File name in which print data
-----------------	-------------------------------

6.5.3.5 set()

```
void TireGround::RDF::MeshSurface::set (
    MeshSurface const & in ) [inline]
```

Copy the [MeshSurface](#) object.

Parameters

<i>in</i>	MeshSurface object to be copied
-----------	---

The documentation for this class was generated from the following file:

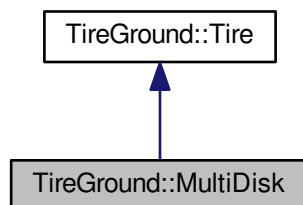
- `include/RoadRDF.hh`

6.6 TireGround::MultiDisk Class Reference

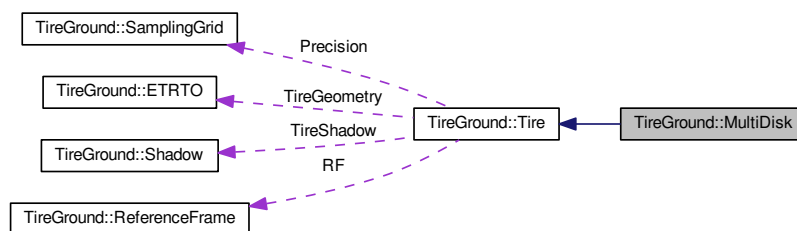
Multi-disk tire contact model.

```
#include <PatchTire.hh>
```

Inheritance diagram for TireGround::MultiDisk:



Collaboration diagram for TireGround::MultiDisk:



Public Member Functions

- [~MultiDisk](#) ()
Default destructor.
- [MultiDisk](#) ([real_type](#) const SectionWidth, [real_type](#) const AspectRatio, [real_type](#) const RimDiameter, [int_type](#) const PointsN, [int_type](#) const DisksN, [int_type](#) const SwitchN)
Variable set constructor.

- **MultiDisk** (**real_type** const SectionWidth, **real_type** const AspectRatio, **real_type** const RimDiameter, **real_type** const SideRadius, **int_type** const PointsN, **int_type** const DisksN, **int_type** const SwitchN)
Variable set constructor.
- **MultiDisk** (**real_type** const SectionWidth, **real_type** const AspectRatio, **real_type** const RimDiameter, **row_vecN** const DisksRadius, **int_type** const PointsN, **int_type** const SwitchN)
Variable set constructor.
- **real_type** **getPointstep** (void) const
Get grid step on X-axis between sampling points [m].
- **real_type** **getDiskStep** (void) const
Get step on Y-axis between disks [m].
- void **getNormal** (**vec3** &_Normal) const override
Get contact normal mean versor.
- void **getDiskOriginXYZ** (**row_vec3** &Origin) const
Get disks origin (X, Y, Z).
- void **getDiskOriginXYZ** (**int_type** const i, **vec3** &Origin) const
Get i-th Disk origin (X, Y, Z).
- void **setDiskOriginXZ** (**row_vec2** &Origin)
Set disks origin (X, Y, Z).
- void **setDiskOriginXZ** (**int_type** const i, **vec2** &Origin)
Set i-th Disk origin (X, Y, Z).
- void **getNormal** (**row_vec3** &_NormalVec) const override
Get contact normal versors vector.
- void **getDiskNormal** (**int_type** const i, **vec3** &_Normal) const
Get i-th Disk contact normal versor.
- void **getMFpoint** (**vec3** &_DiskPoint) const override
Get Magic Formula contact point.
- void **getMFpoint** (**row_vec3** &_DiskPointVec) const override
Get Magic Formula contact points vector.
- void **getDiskMFpoint** (**int_type** const i, **vec3** &_DiskPoint) const
Get i-th Disk Magic Formula contact point.
- void **getFriction** (**real_type** &_Friction) const override
Get area weighted mean contact friction.
- void **getFriction** (**row_vecN** &_Friction) const override
Get contact frictions vector.
- void **getDiskFriction** (**int_type** const i, **real_type** &_Friction) const
Get i-th Disk contact friction.
- void **getMFeffectiveRF** (**mat4** &PointRF) const
Get effective contact point reference frame with 4x4 transformation matrix.
- void **getMFpointRF** (**mat4** &PointRF) const override
Get Magic Formula contact point reference frame with 4x4 transformation matrix.
- void **getMFpointRF** (**row_mat4** &PointRF) const override
Get Magic Formula contact point reference frames vector with 4x4 transformation matrix.
- void **getDiskMFpointRF** (**int_type** const i, **mat4** &PointRF) const
Get Disk Magic Formula contact point reference frame with 4x4 transformation matrix.
- void **getRho** (**real_type** &Rho, **real_type** &RhoDot, **real_type** const RhoOld, **real_type** const Time) const override
- void **getRho** (**row_vecN** &Rho, **row_vecN** &RhoDot, **row_vecN** const RhoOld, **real_type** const Time) const override

- void `getDiskRho` (int_type const i, real_type &Rho, real_type &RhoDot, real_type const RhoOld, real_type const Time) const
- void `getArea` (real_type &_Area) const override
Get approximated mean contact area on [Disk](#) plane [m²].
- void `getArea` (row_vecN &_AreaVec) const override
Get approximated contact areas vector on [Disk](#) plane [m²].
- void `getVolume` (real_type &Volume) const override
Get approximated contact volume [m³].
- void `getVolume` (row_vecN &Volume) const override
Get approximated contact volumes vector [m³].
- void `getMFeffectiveY` (real_type &effectiveY) const
Get effective Y-axis coordinate of contact point [m].
- void `getMFeffectiveR` (real_type &Radius) const
Get effective radius of contact point [m].
- bool `setup` (RDF::MeshSurface &Mesh, mat4 const &TM) override
Update current tire position and find contact parameters.
- void `setup` (vec3 const &Plane_Normal, vec3 const &Plane_Point, real_type const Plane_Friction, mat4 const &TM) override
- void `print` (ostream_type &stream) const override
Print contact parameters.
- void `printETRTOGeometry` (ostream_type &stream) const
Display [Tire ETRTO](#) geometry data.
- G2lib::AABBtree::PtrAABB const `getAABBtree` (void) const
Get total [Tire Shadow](#) G2Lib::AABBtree (3D projection on ground)
- G2lib::AABBtree::PtrAABB const `getUpperSideAABBtree` (void) const
Get upper side [Tire Shadow](#) G2Lib::AABBtree (3D projection on ground)
- G2lib::AABBtree::PtrAABB const `getLowerSideAABBtree` (void) const
Get lower side [Tire Shadow](#) G2Lib::AABBtree (3D projection on ground)
- void `setReferenceFrame` (ReferenceFrame const &_RF)
- ReferenceFrame const & `getReferenceFrame` (void) const
Get tire [ReferenceFrame](#) object.
- void `setOrigin` (vec3 const &Origin)
Set a new tire origin.
- void `setRotationMatrix` (mat3 const &RotationMatrix)
- void `setTotalTransformationMatrix` (mat4 const &TM)
- real_type `getEulerAngleX` (void) const
- real_type `getEulerAngleY` (void) const
- real_type `getEulerAngleZ` (void) const
- void `getRelativeCamber` (real_type &RelativeCamber) const
Get relative camber angle [rad].
- int_type `getDisksNumber` (void) const
Dimension of the contact points data structure (disks number)

Protected Member Functions

- bool `pointSampling` (RDF::TriangleRoad_list const &TriList, vec3 const &RayOrigin, vec3 const &RayDirection, vec3 &SampledPt, real_type &TriFriction=quietNaN, vec3 &TriNormal=vec3_NaN) const
Perform one point sampling (ray-triangle intersection)

Protected Attributes

- [SamplingGrid Precision](#)
Contact patch evaluating precision.
- [ETRTO TireGeometry](#)
Tire ETRTO denomination.
- [ReferenceFrame RF](#)
ReferenceFrame.
- [Shadow TireShadow](#)
Tire shadow.

6.6.1 Detailed Description

Multi-disk tire contact model.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 MultiDisk() [1/3]

```
TireGround::MultiDisk::MultiDisk (  
    real_type const SectionWidth,  
    real_type const AspectRatio,  
    real_type const RimDiameter,  
    int_type const PointsN,  
    int_type const DisksN,  
    int_type const SwitchN ) [inline]
```

Variable set constructor.

Parameters

<i>SectionWidth</i>	Tire section width [<i>m</i>]
<i>AspectRatio</i>	Tire aspect ratio [%]
<i>RimDiameter</i>	Rim diameter [<i>in</i>]
<i>PointsN</i>	Sampling points for each Disk (divisions on <i>X</i> -axis)
<i>DisksN</i>	Number of Disks (divisions on <i>Y</i> -axis –1)
<i>SwitchN</i>	Maximum RoadTriangles in the Tire Shadow (switch to sampling)

6.6.2.2 MultiDisk() [2/3]

```
TireGround::MultiDisk::MultiDisk (  
    real_type const SectionWidth,  
    real_type const AspectRatio,  
    real_type const RimDiameter,  
    real_type const SideRadius,  
    int_type const PointsN,  
    int_type const DisksN,  
    int_type const SwitchN ) [inline]
```

Variable set constructor.

Parameters

<i>SectionWidth</i>	Tire section width [<i>m</i>]
<i>AspectRatio</i>	Tire aspect ratio [%]
<i>RimDiameter</i>	Rim diameter [<i>in</i>]
<i>SideRadius</i>	Sidewall radius [<i>m</i>]
<i>PointsN</i>	Sampling points for each Disk (divisions on <i>X</i> -axis)
<i>DisksN</i>	Number of Disks (divisions on <i>Y</i> -axis –1)
<i>SwitchN</i>	Maximum RoadTriangles in the Tire Shadow (switch to sampling)

6.6.2.3 MultiDisk() [3/3]

```
TireGround::MultiDisk::MultiDisk (
    real_type const SectionWidth,
    real_type const AspectRatio,
    real_type const RimDiameter,
    row_vecN const DisksRadius,
    int_type const PointsN,
    int_type const SwitchN ) [inline]
```

Variable set constructor.

Parameters

<i>SectionWidth</i>	Tire section width [<i>m</i>]
<i>AspectRatio</i>	Tire aspect ratio [%]
<i>RimDiameter</i>	Rim diameter [<i>in</i>]
<i>DisksRadius</i>	Disks radius vector [<i>m</i>]
<i>PointsN</i>	Sampling points for each Disk (divisions on <i>X</i> -axis)
<i>SwitchN</i>	Maximum RoadTriangles in the Tire Shadow (switch to sampling)

6.6.3 Member Function Documentation

6.6.3.1 getArea() [1/2]

```
void TireGround::MultiDisk::getArea (
    real_type & _Area ) const [inline], [override], [virtual]
```

Get approximated mean contact area on [Disk](#) plane [*m*²].

Parameters

<i>_Area</i>	Contact area [<i>m</i> ²]
--------------	--

Implements [TireGround::Tire](#).

6.6.3.2 getArea() [2/2]

```
void TireGround::MultiDisk::getArea (
    row_vecN & _AreaVec ) const [inline], [override], [virtual]
```

Get approximated contact areas vector on [Disk](#) plane [m^2].

Parameters

<i>_AreaVec</i>	Contact areas vector [m^2]
-----------------	--------------------------------

Implements [TireGround::Tire](#).

6.6.3.3 getDiskFriction()

```
void TireGround::MultiDisk::getDiskFriction (
    int_type const i,
    real_type & _Friction ) const [inline]
```

Get i -th [Disk](#) contact friction.

Parameters

i	i -th Disk
<i>_Friction</i>	Disk contact friction

6.6.3.4 getDiskMFpoint()

```
void TireGround::MultiDisk::getDiskMFpoint (
    int_type const i,
    vec3 & _DiskPoint ) const [inline]
```

Get i -th [Disk](#) Magic Formula contact point.

Parameters

i	i -th Disk
<i>_DiskPoint</i>	Disk Magic Formula contact point

6.6.3.5 getDiskMFpointRF()

```
void TireGround::MultiDisk::getDiskMFpointRF (
    int_type const i,
    mat4 & PointRF ) const
```

Get [Disk](#) Magic Formula contact point reference frame with 4x4 transformation matrix.

Parameters

i	i -th Disk
<i>PointRF</i>	Magic Formula contact point reference frame

6.6.3.6 `getDiskNormal()`

```
void TireGround::MultiDisk::getDiskNormal (
    int_type const i,
    vec3 & _Normal ) const [inline]
```

Get i -th [Disk](#) contact normal versor.

Parameters

i	i -th Disk
$_Normal$	Contact normal versor

6.6.3.7 `getDiskOriginXYZ()` [1/2]

```
void TireGround::MultiDisk::getDiskOriginXYZ (
    row_vec3 & Origin ) const [inline]
```

Get disks origin (X, Y, Z).

Parameters

$Origin$	Disks origin
----------	--------------

6.6.3.8 `getDiskOriginXYZ()` [2/2]

```
void TireGround::MultiDisk::getDiskOriginXYZ (
    int_type const i,
    vec3 & Origin ) const [inline]
```

Get i -th [Disk](#) origin (X, Y, Z).

Parameters

i	i -th Disk
$Origin$	Disks origin

6.6.3.9 `getDiskRho()`

```
void TireGround::MultiDisk::getDiskRho (
    int_type const i,
    real_type & Rho,
    real_type & RhoDot,
    real_type const RhoOld,
    real_type const Time ) const
```

Get i -th [Disk](#) contact depth [m] and it time derivative [m/s]

Warning: (if negative the tire does not touch the ground)!

Parameters

<i>i</i>	<i>i</i> -th Disk
<i>Rho</i>	Disk contact depth
<i>RhoDot</i>	Contact depth derivative [<i>m/s</i>]
<i>RhoOld</i>	Previous time step Rho [<i>m</i>]
<i>Time</i>	Time step [<i>s</i>]

6.6.3.10 getEulerAngleX()

```
real_type TireGround::Tire::getEulerAngleX (
    void ) const [inline], [inherited]
```

Get current Euler angles [*rad*] for *X*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.6.3.11 getEulerAngleY()

```
real_type TireGround::Tire::getEulerAngleY (
    void ) const [inline], [inherited]
```

Get current Euler angles [*rad*] for *Y*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.6.3.12 getEulerAngleZ()

```
real_type TireGround::Tire::getEulerAngleZ (
    void ) const [inline], [inherited]
```

Get current Euler angles [*rad*] for *Z*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.6.3.13 getFriction() [1/2]

```
void TireGround::MultiDisk::getFriction (
    real_type & _Friction ) const [override], [virtual]
```

Get area weighted mean contact friction.

Parameters

<i>_Friction</i>	Area weighted mean contact friction
------------------	-------------------------------------

Implements [TireGround::Tire](#).

6.6.3.14 getFriction() [2/2]

```
void TireGround::MultiDisk::getFriction (
    row_vecN & _Friction ) const [inline], [override], [virtual]
```

Get contact frictions vector.

Parameters

<i>_Friction</i>	Contact frictions vector
------------------	--------------------------

Implements [TireGround::Tire](#).

6.6.3.15 getMFeffectiveR()

```
void TireGround::MultiDisk::getMFeffectiveR (
    real_type & Radius ) const
```

Get effective radius of contact point [*m*].

Parameters

<i>Radius</i>	Effective radius of contact point [<i>m</i>]
---------------	--

6.6.3.16 getMFeffectiveRF()

```
void TireGround::MultiDisk::getMFeffectiveRF (
    mat4 & PointRF ) const
```

Get effective contact point reference frame with 4x4 transformation matrix.

Parameters

<i>PointRF</i>	Magic Formula contact point reference frame
----------------	---

6.6.3.17 getMFeffectiveY()

```
void TireGround::MultiDisk::getMFeffectiveY (
    real_type & effectiveY ) const
```

Get effective *Y*-axis coordinate of contact point [*m*].

Parameters

<i>effectiveY</i>	Effective <i>Y</i> -axis coordinate of contact point [<i>m</i>]
-------------------	---

6.6.3.18 getMFpoint() [1/2]

```
void TireGround::MultiDisk::getMFpoint (
    vec3 & _DiskPoint ) const [inline], [override], [virtual]
```

Get Magic Formula contact point.

Parameters

<i>_DiskPoint</i>	Magic Formula contact point
-------------------	-----------------------------

Implements [TireGround::Tire](#).

6.6.3.19 getMFpoint() [2/2]

```
void TireGround::MultiDisk::getMFpoint (
    row_vec3 & _DiskPointVec ) const [inline], [override], [virtual]
```

Get Magic Formula contact points vector.

Parameters

<i>_DiskPointVec</i>	Magic Formula contact points vector
----------------------	-------------------------------------

Implements [TireGround::Tire](#).

6.6.3.20 getMFpointRF() [1/2]

```
void TireGround::MultiDisk::getMFpointRF (
    mat4 & PointRF ) const [override], [virtual]
```

Get Magic Formula contact point reference frame with 4x4 transformation matrix.

Parameters

<i>PointRF</i>	Magic Formula contact point reference frame
----------------	---

Implements [TireGround::Tire](#).

6.6.3.21 getMFpointRF() [2/2]

```
void TireGround::MultiDisk::getMFpointRF (
    row_mat4 & PointRF ) const [inline], [override], [virtual]
```

Get Magic Formula contact point reference frames vector with 4x4 transformation matrix.

Parameters

<i>PointRF</i>	Magic Formula contact point reference frames vector
----------------	---

Implements [TireGround::Tire](#).

6.6.3.22 getNormal() [1/2]

```
void TireGround::MultiDisk::getNormal (
    vec3 & _Normal ) const [inline], [override], [virtual]
```

Get contact normal mean versor.

Parameters

<i>_Normal</i>	Contact normal mean versor
----------------	----------------------------

Implements [TireGround::Tire](#).

6.6.3.23 `getNormal()` [2/2]

```
void TireGround::MultiDisk::getNormal (
    row_vec3 & _NormalVec ) const [inline], [override], [virtual]
```

Get contact normal versors vector.

Parameters

<i>_NormalVec</i>	Contact normal versors vector
-------------------	-------------------------------

Implements [TireGround::Tire](#).

6.6.3.24 `getRelativeCamber()`

```
void TireGround::Tire::getRelativeCamber (
    real_type & RelativeCamber ) const [inherited]
```

Get relative camber angle [*rad*].

Parameters

<i>RelativeCamber</i>	Relative camber angle
-----------------------	-----------------------

6.6.3.25 `getRho()` [1/2]

```
void TireGround::MultiDisk::getRho (
    real_type & Rho,
    real_type & RhoDot,
    real_type const RhoOld,
    real_type const Time ) const [override], [virtual]
```

Get contact depth at center point [*m*] and it time derivative [*m/s*]
Warning: (if negative the tire does not touch the ground)!

Parameters

<i>Rho</i>	Depth at center point [<i>m/s</i>]
<i>RhoDot</i>	Contact depth derivative [<i>m/s</i>]
<i>RhoOld</i>	Previous time step Rho [<i>m</i>]
<i>Time</i>	Time step [<i>s</i>]

Implements [TireGround::Tire](#).

6.6.3.26 `getRho()` [2/2]

```
void TireGround::MultiDisk::getRho (
    row_vecN & Rho,
    row_vecN & RhoDot,
    row_vecN const RhoOld,
    real_type const Time ) const [override], [virtual]
```

Get contact depths vector [m] and it time derivatives [m/s]
Warning: (if negative the tire does not touch the ground)!

Parameters

<i>Rho</i>	Depth matrix [m/s]
<i>RhoDot</i>	Contact depth derivative matrix [m/s]
<i>RhoOld</i>	Previous time step Rho matrix [m]
<i>Time</i>	Time step [s]

Implements [TireGround::Tire](#).

6.6.3.27 getVolume() [1/2]

```
void TireGround::MultiDisk::getVolume (
    real_type & Volume ) const [inline], [override], [virtual]
```

Get approximated contact volume [m^3].

Parameters

<i>Volume</i>	Contact volume [m^3]
---------------	--------------------------

Implements [TireGround::Tire](#).

6.6.3.28 getVolume() [2/2]

```
void TireGround::MultiDisk::getVolume (
    row_vecN & Volume ) const [inline], [override], [virtual]
```

Get approximated contact volumes vector [m^3].

Parameters

<i>Volume</i>	Contact volumes vector [m^3]
---------------	----------------------------------

Implements [TireGround::Tire](#).

6.6.3.29 pointSampling()

```
bool TireGround::Tire::pointSampling (
    RDF::TriangleRoad_list const & TriList,
    vec3 const & RayOrigin,
    vec3 const & RayDirection,
    vec3 & SampledPt,
    real_type & TriFriction = quietNaN,
    vec3 & TriNormal = vec3_NaN ) const [protected], [inherited]
```

Perform one point sampling (ray-triangle intersection)

Parameters

<i>TriList</i>	Shadow/MeshSurface intersected triangles
<i>RayOrigin</i>	Ray origin
<i>RayDirection</i>	Ray direction
<i>SampledPt</i>	Intersection point
<i>TriFriction</i>	Intersected triangle friction
<i>TriNormal</i>	Intersected triangle normal

6.6.3.30 print()

```
void TireGround::MultiDisk::print (
    ostream_type & stream ) const [override], [virtual]
```

Print contact parameters.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

Implements [TireGround::Tire](#).

6.6.3.31 printETRTOGeometry()

```
void TireGround::Tire::printETRTOGeometry (
    ostream_type & stream ) const [inline], [inherited]
```

Display [Tire ETRTO](#) geometry data.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

6.6.3.32 setDiskOriginXZ() [1/2]

```
void TireGround::MultiDisk::setDiskOriginXZ (
    row_vec2 & Origin ) [inline]
```

Set disks origin (X, Y, Z).

Parameters

<i>Origin</i>	New Disks origin vector
---------------	-------------------------

6.6.3.33 setDiskOriginXZ() [2/2]

```
void TireGround::MultiDisk::setDiskOriginXZ (
```

```
int_type const i,
vec2 & Origin ) [inline]
```

Set i -th [Disk](#) origin (X, Y, Z).

Parameters

i	i -th Disk
<i>Origin</i>	New Disks origin vector

6.6.3.34 setOrigin()

```
void TireGround::Tire::setOrigin (
    vec3 const & Origin ) [inline], [inherited]
```

Set a new tire origin.

Parameters

<i>Origin</i>	Tire origin
---------------	-----------------------------

6.6.3.35 setReferenceFrame()

```
void TireGround::Tire::setReferenceFrame (
    ReferenceFrame const & _RF ) [inline], [inherited]
```

Copy the tire [ReferenceFrame](#) object

Warning: Rotation matrix must be orthonormal!

Parameters

<i>_RF</i>	ReferenceFrame object to be copied
------------	--

6.6.3.36 setRotationMatrix()

```
void TireGround::Tire::setRotationMatrix (
    mat3 const & RotationMatrix ) [inline], [inherited]
```

Set a new 3x3 rotation matrix

Warning: Rotation matrix must be orthonormal!

Parameters

<i>RotationMatrix</i>	Rotation matrix
-----------------------	-----------------

6.6.3.37 setTotalTransformationMatrix()

```
void TireGround::Tire::setTotalTransformationMatrix (
    mat4 const & TM ) [inline], [inherited]
```

Set 4x4 total transformation matrix

Warning: Rotation matrix must be orthonormal!

Parameters

<i>TM</i>	4x4 total transformation matrix
-----------	---------------------------------

6.6.3.38 `setup()` [1/2]

```
bool TireGround::MultiDisk::setup (
    RDF::MeshSurface & Mesh,
    mat4 const & TM ) [override], [virtual]
```

Update current tire position and find contact parameters.

Parameters

<i>Mesh</i>	MeshSurface object (road)
<i>TM</i>	4x4 total transformation matrix

Implements [TireGround::Tire](#).

6.6.3.39 `setup()` [2/2]

```
void TireGround::MultiDisk::setup (
    vec3 const & Plane_Normal,
    vec3 const & Plane_Point,
    real_type const Plane_Friction,
    mat4 const & TM ) [override], [virtual]
```

Update current tire position and find contact parameters with external plane

Parameters

<i>Plane_Normal</i>	Plane normal vector
<i>Plane_Point</i>	Plane known point
<i>Plane_Friction</i>	Friction on plane
<i>TM</i>	4x4 total transformation matrix

Implements [TireGround::Tire](#).

The documentation for this class was generated from the following file:

- `include/PatchTire.hh`

6.7 TireGround::ReferenceFrame Class Reference

Reference frame.

```
#include <PatchTire.hh>
```

Public Member Functions

- [ReferenceFrame](#) ()
Default constructor.
- [ReferenceFrame](#) ([vec3](#) const &_Origin, [mat3](#) const &_RotationMatrix)
Variable set constructor.
- [bool isEmpty](#) (void)
Check if [ReferenceFrame](#) object is empty.
- [mat3](#) const & [getRotationMatrix](#) (void) const
Get current 3x3 rotation matrix.
- [mat3](#) [getRotationMatrixInverse](#) (void) const
Get current 3x3 rotation matrix inverse.
- [vec3](#) [getX](#) (void) const
Get current X-axis versor.
- [vec3](#) [getY](#) (void) const
Get current Y-axis versor.
- [vec3](#) [getZ](#) (void) const
Get current Z-axis versor.
- [vec3](#) const & [getOrigin](#) (void) const
Get origin position.
- void [setOrigin](#) ([vec3](#) const &_Origin)
Set origin position.
- void [setRotationMatrix](#) ([mat3](#) const &_RotationMatrix)
Set 3x3 rotation matrix.
- void [setTotalTransformationMatrix](#) ([mat4](#) const &TM)
Set 4x4 total transformation matrix.
- [mat4](#) [getTotalTransformationMatrix](#) (void)
Get 4x4 total transformation matrix.
- void [set](#) ([ReferenceFrame](#) const &in)
- [real_type](#) [getEulerAngleX](#) (void) const
- [real_type](#) [getEulerAngleY](#) (void) const
- [real_type](#) [getEulerAngleZ](#) (void) const

6.7.1 Detailed Description

Reference frame.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 ReferenceFrame()

```
TireGround::ReferenceFrame::ReferenceFrame (  
    vec3 const & _Origin,  
    mat3 const & _RotationMatrix ) [inline]
```

Variable set constructor.

Parameters

_Origin	Origin position
_RotationMatrix	3x3 rotation matrix

6.7.3 Member Function Documentation

6.7.3.1 getEulerAngleX()

```
real_type TireGround::ReferenceFrame::getEulerAngleX (
    void ) const
```

Get current Euler angles [*rad*] for *X*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.7.3.2 getEulerAngleY()

```
real_type TireGround::ReferenceFrame::getEulerAngleY (
    void ) const
```

Get current Euler angles [*rad*] for *Y*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.7.3.3 getEulerAngleZ()

```
real_type TireGround::ReferenceFrame::getEulerAngleZ (
    void ) const
```

Get current Euler angles [*rad*] for *Z*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.7.3.4 set()

```
void TireGround::ReferenceFrame::set (
    ReferenceFrame const & in ) [inline]
```

Copy the tire [ReferenceFrame](#) object

Warning: Rotation matrix must be orthonormal!

Parameters

<i>in</i>	ReferenceFrame object to be copied
-----------	--

6.7.3.5 setOrigin()

```
void TireGround::ReferenceFrame::setOrigin (
    vec3 const & _Origin ) [inline]
```

Set origin position.

Parameters

<i>_Origin</i>	Origin position
----------------	-----------------

6.7.3.6 setRotationMatrix()

```
void TireGround::ReferenceFrame::setRotationMatrix (
    mat3 const & _RotationMatrix ) [inline]
```

Set 3x3 rotation matrix.

Parameters

<code>_RotationMatrix</code>	3x3 rotation matrix
------------------------------	---------------------

6.7.3.7 setTotalTransformationMatrix()

```
void TireGround::ReferenceFrame::setTotalTransformationMatrix (
    mat4 const & TM ) [inline]
```

Set 4x4 total transformation matrix.

Parameters

<code>TM</code>	4x4 total transformation matrix
-----------------	---------------------------------

The documentation for this class was generated from the following file:

- include/PatchTire.hh

6.8 TireGround::SamplingGrid Class Reference

Patch evaluation precision.

```
#include <PatchTire.hh>
```

Public Member Functions

- [SamplingGrid \(\)](#)
Default constructor.
- [SamplingGrid \(int_type _PointsN, int_type _DisksN\)](#)
Variable set constructor.
- [SamplingGrid \(int_type _PointsN, int_type _DisksN, int_type _Switch\)](#)
Variable set constructor.
- [int_type getPointsNumber \(void\) const](#)
Get number of sampling points for each [Disk](#) (divisions on X-axis)
- [int_type getDisksNumber \(void\) const](#)
Get number of Disks (divisions on Y-axis – 1)
- [unsigned getSwitchNumber \(void\) const](#)
Get number of maximum RoadTriangles in the [Tire Shadow](#) (switch to sampling)
- [void setSwitchNumber \(int_type const _Switch\)](#)
Set number of maximum RoadTriangles in the [Tire Shadow](#) (switch to sampling)
- [void set \(int_type _PointsN, int_type _DisksN, int_type _Switch\)](#)
Set number of divisions.
- [void set \(SamplingGrid const &in\)](#)
Copy the [SamplingGrid](#) object.

6.8.1 Detailed Description

Patch evaluation precision.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 SamplingGrid() [1/2]

```
TireGround::SamplingGrid::SamplingGrid (
    int_type _PointsN,
    int_type _DisksN ) [inline]
```

Variable set constructor.

Parameters

<code>_PointsN</code>	Sampling points for each Disk (divisions on <i>X</i> -axis)
<code>_DisksN</code>	Number of Disks (divisions on <i>Y</i> -axis – 1)

6.8.2.2 SamplingGrid() [2/2]

```
TireGround::SamplingGrid::SamplingGrid (
    int_type _PointsN,
    int_type _DisksN,
    int_type _Switch ) [inline]
```

Variable set constructor.

Parameters

<code>_PointsN</code>	Sampling points for each Disk (divisions on <i>X</i> -axis)
<code>_DisksN</code>	Number of Disks (divisions on <i>Y</i> -axis – 1)
<code>_Switch</code>	Maximum RoadTriangles in the Tire Shadow (switch to sampling)

6.8.3 Member Function Documentation

6.8.3.1 set() [1/2]

```
void TireGround::SamplingGrid::set (
    int_type _PointsN,
    int_type _DisksN,
    int_type _Switch ) [inline]
```

Set number of divisions.

Parameters

<code>_PointsN</code>	Sampling points for each Disk (divisions on <i>X</i> -axis)
<code>_DisksN</code>	Number of Disks (divisions on <i>Y</i> -axis – 1)
<code>_Switch</code>	Maximum RoadTriangles in the Tire Shadow (switch to sampling)

6.8.3.2 set() [2/2]

```
void TireGround::SamplingGrid::set (
    SamplingGrid const & in ) [inline]
```

Copy the [SamplingGrid](#) object.

Parameters

<i>in</i>	SamplingGrid object to be copied
-----------	--

6.8.3.3 setSwitchNumber()

```
void TireGround::SamplingGrid::setSwitchNumber (
    int_type const _Switch ) [inline]
```

Set number of maximum RoadTriangles in the [Tire Shadow](#) (switch to sampling)

Parameters

<i>_Switch</i>	New switch number
----------------	-------------------

The documentation for this class was generated from the following file:

- include/PatchTire.hh

6.9 TireGround::Shadow Class Reference

2D shadow (2D bounding box enhancement)

```
#include <PatchTire.hh>
```

Public Member Functions

- [Shadow](#) ()
Default constructor.
- [Shadow](#) ([ETRTO](#) const &TireGeometry, [ReferenceFrame](#) const &RF)
- void [update](#) ([ETRTO](#) const &TireGeometry, [ReferenceFrame](#) const &RF)
- G2lib::AABBtree::PtrAABB const [getAABBtree](#) (void) const
Get total [Tire](#) G2Lib::AABBtree (3D projection on ground)
- G2lib::AABBtree::PtrAABB const [getUpperSideAABBtree](#) (void) const
Get upper side [Tire](#) G2Lib::AABBtree (3D projection on ground)
- G2lib::AABBtree::PtrAABB const [getLowerSideAABBtree](#) (void) const
Get lower side [Tire](#) G2Lib::AABBtree (3D projection on ground)

6.9.1 Detailed Description

2D shadow (2D bounding box enhancement)

6.9.2 Constructor & Destructor Documentation

6.9.2.1 Shadow()

```
TireGround::Shadow::Shadow (
    ETRTO const & TireGeometry,
    ReferenceFrame const & RF ) [inline]
```

Variable set constructor

Warning: Rotation matrix must be orthonormal!

Parameters

<i>TireGeometry</i>	Tire ETRTO denomination
<i>RF</i>	Tire ReferenceFrame

6.9.3 Member Function Documentation

6.9.3.1 update()

```
void TireGround::Shadow::update (
    ETRTO const & TireGeometry,
    ReferenceFrame const & RF )
```

Update the 2D tire shadow domain

Warning: Rotation matrix must be orthonormal!

Parameters

<i>TireGeometry</i>	Tire ETRTO denomination
<i>RF</i>	Tire ReferenceFrame

The documentation for this class was generated from the following file:

- include/PatchTire.hh

6.10 TicToc Class Reference

Public Member Functions

- void **tic** ()
- void **toc** ()
- real_type **elapsed_s** () const
- real_type **elapsed_ms** () const

The documentation for this class was generated from the following file:

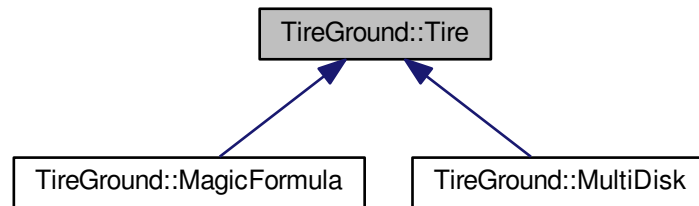
- include/TicToc.hh

6.11 TireGround::Tire Class Reference

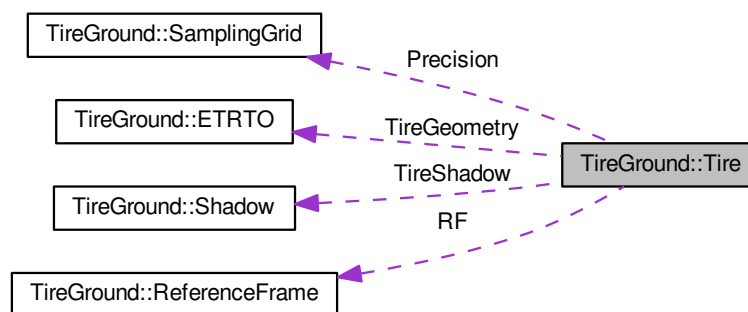
Base class for [Tire](#) models.

```
#include <PatchTire.hh>
```

Inheritance diagram for TireGround::Tire:



Collaboration diagram for TireGround::Tire:



Public Member Functions

- `~Tire()`
Default destructor.
- `Tire(real_type const SectionWidth, real_type const AspectRatio, real_type const RimDiameter, int_type const PointsN, int_type const DisksN)`
Variable set constructor.
- `void printETRTOGeometry(ostream_type &stream) const`
Display Tire ETRTO geometry data.
- `G2lib::AABBtree::PtrAABB const getAABBtree(void) const`
Get total Tire Shadow G2Lib::AABBtree (3D projection on ground)
- `G2lib::AABBtree::PtrAABB const getUpperSideAABBtree(void) const`
Get upper side Tire Shadow G2Lib::AABBtree (3D projection on ground)

- G2lib::AABBtree::PtrAABB const [getLowerSideAABBtree](#) (void) const
Get lower side [Tire Shadow](#) G2Lib:AABBtree (3D projection on ground)
- void [setReferenceFrame](#) ([ReferenceFrame](#) const &_RF)
- [ReferenceFrame](#) const & [getReferenceFrame](#) (void) const
Get tire [ReferenceFrame](#) object.
- void [setOrigin](#) ([vec3](#) const &Origin)
Set a new tire origin.
- void [setRotationMatrix](#) ([mat3](#) const &RotationMatrix)
- void [setTotalTransformationMatrix](#) ([mat4](#) const &TM)
- [real_type](#) [getEulerAngleX](#) (void) const
- [real_type](#) [getEulerAngleY](#) (void) const
- [real_type](#) [getEulerAngleZ](#) (void) const
- void [getRelativeCamber](#) ([real_type](#) &RelativeCamber) const
Get relative camber angle [rad].
- [int_type](#) [getDisksNumber](#) (void) const
Dimension of the contact points data structure (disks number)
- virtual void [getRho](#) ([real_type](#) &Rho, [real_type](#) &RhoDot, [real_type](#) const RhoOld, [real_type](#) const Time) const =0
- virtual void [getRho](#) ([row_vecN](#) &Rho, [row_vecN](#) &RhoDot, [row_vecN](#) const RhoOld, [real_type](#) const Time) const =0
- virtual void [getNormal](#) ([vec3](#) &Normal) const =0
Get contact normal versor.
- virtual void [getNormal](#) ([row_vec3](#) &Normal) const =0
Get contact normal versors vector.
- virtual void [getMFpoint](#) ([vec3](#) &Point) const =0
Get Magic Formula contact point.
- virtual void [getMFpoint](#) ([row_vec3](#) &Point) const =0
Get Magic Formula contact point vector.
- virtual void [getFriction](#) ([real_type](#) &Friction) const =0
Get contact point friction.
- virtual void [getFriction](#) ([row_vecN](#) &Friction) const =0
Get contact frictions vector.
- virtual void [getMFpointRF](#) ([mat4](#) &PointRF) const =0
Get Magic Formula contact point reference frame with 4x4 transformation matrix.
- virtual void [getMFpointRF](#) ([row_mat4](#) &PointRF) const =0
Get Magic Formula contact point reference frame vector with 4x4 transformation matrix.
- virtual void [getArea](#) ([real_type](#) &_Area) const =0
Get approximated contact area on [Disk](#) plane [m^2].
- virtual void [getArea](#) ([row_vecN](#) &Area) const =0
Get approximated contact areas vector on [Disk](#) plane [m^2].
- virtual void [getVolume](#) ([real_type](#) &Volume) const =0
Get approximated contact volume [m^3].
- virtual void [getVolume](#) ([row_vecN](#) &_Volume) const =0
Get approximated contact volume [m^3].
- virtual void [evaluateContact](#) ([RDF::TriangleRoad_list](#) const &TriList)=0
Evaluate contact with RoadTriangles.
- virtual bool [setup](#) ([RDF::MeshSurface](#) &Mesh, [mat4](#) const &TM)=0
Update current tire position and find contact parameters.
- virtual void [setup](#) ([vec3](#) const &Plane_Normal, [vec3](#) const &Plane_Point, [real_type](#) const Plane_Friction, [mat4](#) const &TM)=0
- virtual void [print](#) ([ostream_type](#) &stream) const =0
Print contact parameters.

Protected Member Functions

- [Tire](#) ([Tire](#) const &)=delete
Deleted copy constructor.
- [Tire](#) const & [operator=](#) ([Tire](#) const &)=delete
Deleted copy operator.
- bool [pointSampling](#) ([RDF::TriangleRoad_list](#) const &TriList, [vec3](#) const &RayOrigin, [vec3](#) const &RayDirection, [vec3](#) &SampledPt, [real_type](#) &TriFriction=quietNaN, [vec3](#) &TriNormal=[vec3_NaN](#)) const
Perform one point sampling (ray-triangle intersection)

Protected Attributes

- [SamplingGrid Precision](#)
Contact patch evaluating precision.
- [ETRTO TireGeometry](#)
Tire ETRTO denomination.
- [ReferenceFrame RF](#)
ReferenceFrame.
- [Shadow TireShadow](#)
Tire shadow.

6.11.1 Detailed Description

Base class for [Tire](#) models.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 Tire()

```
TireGround::Tire::Tire (  
    real\_type const SectionWidth,  
    real\_type const AspectRatio,  
    real\_type const RimDiameter,  
    int\_type const PointsN,  
    int\_type const DisksN ) [inline]
```

Variable set constructor.

Parameters

<i>SectionWidth</i>	Tire section width [<i>m</i>]
<i>AspectRatio</i>	Tire aspect ratio [%]
<i>RimDiameter</i>	Rim diameter [<i>in</i>]
<i>PointsN</i>	Sampling points for each Disk (divisions on <i>X</i> -axis)
<i>DisksN</i>	Number of Disks (divisions on <i>Y</i> -axis −1)

6.11.3 Member Function Documentation

6.11.3.1 evaluateContact()

```
virtual void TireGround::Tire::evaluateContact (
    RDF::TriangleRoad_list const & TriList ) [pure virtual]
```

Evaluate contact with RoadTriangles.

Parameters

<i>TriList</i>	Shadow/MeshSurface intersected triangles
----------------	--

Implemented in [TireGround::MagicFormula](#).

6.11.3.2 getArea() [1/2]

```
virtual void TireGround::Tire::getArea (
    real_type & _Area ) const [pure virtual]
```

Get approximated contact area on [Disk](#) plane [m^2].

Parameters

<i>_Area</i>	Contact area [m^2]
--------------	------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.3 getArea() [2/2]

```
virtual void TireGround::Tire::getArea (
    row_vecN & Area ) const [pure virtual]
```

Get approximated contact areas vector on [Disk](#) plane [m^2].

Parameters

<i>Area</i>	Contact areas vector [m^2]
-------------	--------------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.4 getEulerAngleX()

```
real_type TireGround::Tire::getEulerAngleX (
    void ) const [inline]
```

Get current Euler angles [*rad*] for *X*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.11.3.5 getEulerAngleY()

```
real_type TireGround::Tire::getEulerAngleY (
    void ) const [inline]
```

Get current Euler angles [*rad*] for *Y*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.11.3.6 getEulerAngleZ()

```
real_type TireGround::Tire::getEulerAngleZ (
    void ) const [inline]
```

Get current Euler angles [*rad*] for *Z*-axis

Warning: Factor as $[R_z][R_x][R_y]!$

6.11.3.7 getFriction() [1/2]

```
virtual void TireGround::Tire::getFriction (
    real_type & Friction ) const [pure virtual]
```

Get contact point friction.

Parameters

<i>Friction</i>	Contact point friction
-----------------	------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.8 getFriction() [2/2]

```
virtual void TireGround::Tire::getFriction (
    row_vecN & Friction ) const [pure virtual]
```

Get contact frictions vector.

Parameters

<i>Friction</i>	Contact frictions vector
-----------------	--------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.9 getMFpoint() [1/2]

```
virtual void TireGround::Tire::getMFpoint (
    vec3 & Point ) const [pure virtual]
```

Get Magic Formula contact point.

Parameters

<i>Point</i>	Magic Formula contact point
--------------	-----------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.10 getMFpoint() [2/2]

```
virtual void TireGround::Tire::getMFpoint (
    row_vec3 & Point ) const [pure virtual]
```

Get Magic Formula contact point vector.

Parameters

<i>Point</i>	Magic Formula Contact point vector
--------------	------------------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.11 getMFpointRF() [1/2]

```
virtual void TireGround::Tire::getMFpointRF (
    mat4 & PointRF ) const [pure virtual]
```

Get Magic Formula contact point reference frame with 4x4 transformation matrix.

Parameters

<i>PointRF</i>	Magic Formula contact point reference frame
----------------	---

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.12 getMFpointRF() [2/2]

```
virtual void TireGround::Tire::getMFpointRF (
    row_mat4 & PointRF ) const [pure virtual]
```

Get Magic Formula contact point reference frame vector with 4x4 transformation matrix.

Parameters

<i>PointRF</i>	Magic Formula ontact point reference frames vector
----------------	--

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.13 getNormal() [1/2]

```
virtual void TireGround::Tire::getNormal (
    vec3 & Normal ) const [pure virtual]
```

Get contact normal versor.

Parameters

<i>Normal</i>	Contact point normal direction
---------------	--------------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.14 getNormal() [2/2]

```
virtual void TireGround::Tire::getNormal (
    row_vec3 & Normal ) const [pure virtual]
```

Get contact normal versors vector.

Parameters

<i>Normal</i>	Contact point normal direction vector
---------------	---------------------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.15 getRelativeCamber()

```
void TireGround::Tire::getRelativeCamber (
    real_type & RelativeCamber ) const
```

Get relative camber angle [*rad*].

Parameters

<i>RelativeCamber</i>	Relative camber angle
-----------------------	-----------------------

6.11.3.16 getRho() [1/2]

```
virtual void TireGround::Tire::getRho (
    real_type & Rho,
    real_type & RhoDot,
    real_type const RhoOld,
    real_type const Time ) const [pure virtual]
```

Get contact depth at center point [*m*]

Warning: (if negative the tire does not touch the ground)!

Parameters

<i>Rho</i>	Depth at center point [<i>m/s</i>]
<i>RhoDot</i>	Contact depth derivative [<i>m/s</i>]
<i>RhoOld</i>	Previous time step Rho [<i>m</i>]
<i>Time</i>	Time step [<i>s</i>]

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.17 getRho() [2/2]

```
virtual void TireGround::Tire::getRho (
    row_vecN & Rho,
    row_vecN & RhoDot,
    row_vecN const RhoOld,
    real_type const Time ) const [pure virtual]
```

Get contact depth vector [*m*] and it time derivatives [*m/s*]

Warning: (if negative the tire does not touch the ground)!

Parameters

<i>Rho</i>	Depth matrix [<i>m/s</i>]
------------	-----------------------------

Parameters

<i>RhoDot</i>	Contact depth derivative matrix [m/s]
<i>RhoOld</i>	Previous time step Rho matrix [m]
<i>Time</i>	Time step [s]

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.18 `getVolume()` [1/2]

```
virtual void TireGround::Tire::getVolume (
    real_type & Volume ) const [pure virtual]
```

Get approximated contact volume [m^3].

Parameters

<i>Volume</i>	Contact volume [m^3]
---------------	--------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.19 `getVolume()` [2/2]

```
virtual void TireGround::Tire::getVolume (
    row_vecN & _Volume ) const [pure virtual]
```

Get approximated contact volume [m^3].

Parameters

<i>_Volume</i>	Contact volume vector [m^3]
----------------	---------------------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.20 `pointSampling()`

```
bool TireGround::Tire::pointSampling (
    RDF::TriangleRoad_list const & TriList,
    vec3 const & RayOrigin,
    vec3 const & RayDirection,
    vec3 & SampledPt,
    real_type & TriFriction = quietNaN,
    vec3 & TriNormal = vec3_NaN ) const [protected]
```

Perform one point sampling (ray-triangle intersection)

Parameters

<i>TriList</i>	Shadow/MeshSurface intersected triangles
<i>RayOrigin</i>	Ray origin
<i>RayDirection</i>	Ray direction

Parameters

<i>SampledPt</i>	Intersection point
<i>TriFriction</i>	Intersected triangle friction
<i>TriNormal</i>	Intersected triangle normal

6.11.3.21 print()

```
virtual void TireGround::Tire::print (
    ostream_type & stream ) const [pure virtual]
```

Print contact parameters.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.22 printETRTOGeometry()

```
void TireGround::Tire::printETRTOGeometry (
    ostream_type & stream ) const [inline]
```

Display [Tire ETRTO](#) geometry data.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

6.11.3.23 setOrigin()

```
void TireGround::Tire::setOrigin (
    vec3 const & Origin ) [inline]
```

Set a new tire origin.

Parameters

<i>Origin</i>	Tire origin
---------------	-----------------------------

6.11.3.24 setReferenceFrame()

```
void TireGround::Tire::setReferenceFrame (
    ReferenceFrame const & _RF ) [inline]
```

Copy the tire [ReferenceFrame](#) object

Warning: Rotation matrix must be orthonormal!

Parameters

<i>_RF</i>	ReferenceFrame object to be copied
------------	--

6.11.3.25 `setRotationMatrix()`

```
void TireGround::Tire::setRotationMatrix (
    mat3 const & RotationMatrix ) [inline]
```

Set a new 3x3 rotation matrix

Warning: Rotation matrix must be orthonormal!

Parameters

<i>RotationMatrix</i>	Rotation matrix
-----------------------	-----------------

6.11.3.26 `setTotalTransformationMatrix()`

```
void TireGround::Tire::setTotalTransformationMatrix (
    mat4 const & TM ) [inline]
```

Set 4x4 total transformation matrix

Warning: Rotation matrix must be orthonormal!

Parameters

<i>TM</i>	4x4 total transformation matrix
-----------	---------------------------------

6.11.3.27 `setup()` [1/2]

```
virtual bool TireGround::Tire::setup (
    RDF::MeshSurface & Mesh,
    mat4 const & TM ) [pure virtual]
```

Update current tire position and find contact parameters.

Parameters

<i>Mesh</i>	MeshSurface object (road)
<i>TM</i>	4x4 total transformation matrix

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

6.11.3.28 `setup()` [2/2]

```
virtual void TireGround::Tire::setup (
    vec3 const & Plane_Normal,
    vec3 const & Plane_Point,
```

```
real_type const Plane_Friction,  
mat4 const & TM ) [pure virtual]
```

Update current tire position and find contact parameters with external plane

Parameters

<i>Plane_Normal</i>	Plane normal vector
<i>Plane_Point</i>	Plane known point
<i>Plane_Friction</i>	Friction on plane
<i>TM</i>	4x4 total transformation matrix

Implemented in [TireGround::MultiDisk](#), and [TireGround::MagicFormula](#).

The documentation for this class was generated from the following file:

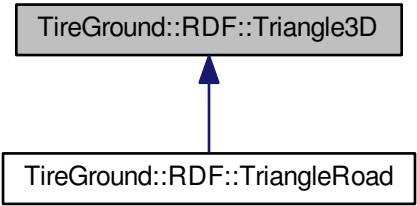
- include/PatchTire.hh

6.12 TireGround::RDF::Triangle3D Class Reference

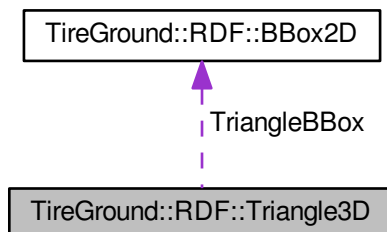
3D triangle (pure geometrical description)

```
#include <RoadRDF.hh>
```

Inheritance diagram for TireGround::RDF::Triangle3D:



Collaboration diagram for TireGround::RDF::Triangle3D:



Public Member Functions

- [Triangle3D](#) ()
Variable set constructor.
- [Triangle3D](#) ([vec3](#) const _Vertices[3])
Variable set constructor.
- void [setVertices](#) ([vec3](#) const _Vertices[3])
Set new vertices and update bounding box domain.
- void [setVertices](#) ([vec3](#) const &Vertex0, [vec3](#) const &Vertex1, [vec3](#) const &Vertex2)
Set new vertices then update bounding box domain and normal versor.
- [vec3](#) const &[getNormal](#) (void) const
Get normal versor.
- [vec3](#) const &[getVertex](#) (unsigned i) const
Get i-th vertex.
- [BBox2D](#) const &[getBBox](#) (void) const
Get [Triangle3D](#) bonding box [BBox2D](#).
- void [print](#) ([ostream_type](#) &stream) const
Print vertices data.
- bool [intersectRay](#) ([vec3](#) const &RayOrigin, [vec3](#) const &RayDirection, [vec3](#) &IntPt) const
- [int_type](#) [intersectEdgePlane](#) ([vec3](#) const &PlaneN, [vec3](#) const &PlaneP, [int_type](#) const Edge, [vec3](#) &IntPt1, [vec3](#) &IntPt2) const
- bool [intersectPlane](#) ([vec3](#) const &PlaneN, [vec3](#) const &PlaneP, std::vector< [vec3](#) > &IntPts) const

Protected Member Functions

- [Triangle3D](#) ([Triangle3D](#) const &)=delete
Deleted copy constructor.
- [Triangle3D](#) & [operator=](#) ([Triangle3D](#) const &)=delete
Deleted copy operator.

Protected Attributes

- [vec3 Vertices](#) [3]
Vertices reference vector.
- [vec3 Normal](#)
Triangle normal versor.
- [BBox2D TriangleBBox](#)
Triangle 2D bounding box (XY plane)

6.12.1 Detailed Description

3D triangle (pure geometrical description)

6.12.2 Constructor & Destructor Documentation

6.12.2.1 Triangle3D()

```
TireGround::RDF::Triangle3D::Triangle3D (  
    vec3 const &_Vertices[3] ) [inline]
```

Variable set constructor.

Parameters

<code>_Vertices</code>	Vertices reference vector
------------------------	---------------------------

6.12.3 Member Function Documentation

6.12.3.1 intersectEdgePlane()

```
int\_type TireGround::RDF::Triangle3D::intersectEdgePlane (  
    vec3 const & PlaneN,  
    vec3 const & PlaneP,  
    int\_type const Edge,  
    vec3 & IntPt1,  
    vec3 & IntPt2 ) const
```

Check if an edge of the [Triangle3D](#) object hits a and find the intersection point

Parameters

<code>PlaneN</code>	Plane normal vector
<code>PlaneP</code>	Plane known point
<code>Edge</code>	Triangle edge number (0:2)
<code>IntPt1</code>	Intersection point 1
<code>IntPt2</code>	Intersection point 2

6.12.3.2 intersectPlane()

```
bool TireGround::RDF::Triangle3D::intersectPlane (
    vec3 const & PlaneN,
    vec3 const & PlaneP,
    std::vector< vec3 > & IntPts ) const
```

Check if a plane intersects a [Triangle3D](#) object and find the intersection points

Parameters

<i>PlaneN</i>	Plane normal vector
<i>PlaneP</i>	Plane known point
<i>IntPts</i>	Intersection points

6.12.3.3 intersectRay()

```
bool TireGround::RDF::Triangle3D::intersectRay (
    vec3 const & RayOrigin,
    vec3 const & RayDirection,
    vec3 & IntPt ) const
```

Check if a ray hits a [Triangle3D](#) object through Möller-Trumbore intersection algorithm

Parameters

<i>RayOrigin</i>	Ray origin position
<i>RayDirection</i>	Ray direction vector
<i>IntPt</i>	Intersection point

6.12.3.4 print()

```
void TireGround::RDF::Triangle3D::print (
    ostream_type & stream ) const [inline]
```

Print vertices data.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

6.12.3.5 setVertices() [1/2]

```
void TireGround::RDF::Triangle3D::setVertices (
    vec3 const _Vertices[3] ) [inline]
```

Set new vertices and update bounding box domain.

Parameters

<i>_Vertices</i>	Vertices reference vector
------------------	---------------------------

6.12.3.6 setVertices() [2/2]

```
void TireGround::RDF::Triangle3D::setVertices (
    vec3 const & Vertex0,
    vec3 const & Vertex1,
    vec3 const & Vertex2 ) [inline]
```

Set new vertices then update bounding box domain and normal versor.

Parameters

<i>Vertex0</i>	Vertex 1
<i>Vertex1</i>	Vertex 2
<i>Vertex2</i>	Vertex 3

The documentation for this class was generated from the following file:

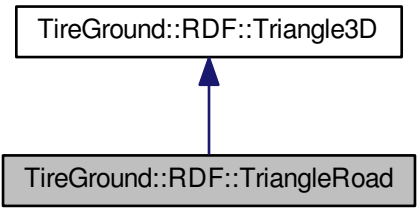
- include/RoadRDF.hh

6.13 TireGround::RDF::TriangleRoad Class Reference

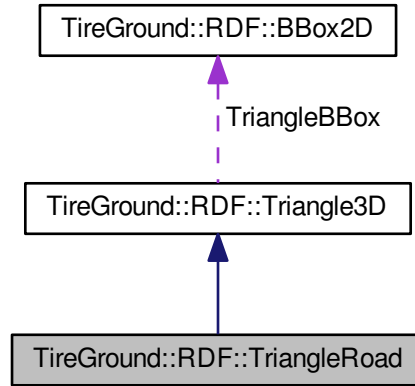
3D triangles for road representation

```
#include <RoadRDF.hh>
```

Inheritance diagram for TireGround::RDF::TriangleRoad:



Collaboration diagram for TireGround::RDF::TriangleRoad:



Public Member Functions

- [TriangleRoad](#) ()
Default set constructor.
- [TriangleRoad](#) ([vec3](#) const _Vertices[3], [real_type](#) _Friction)
Variable set constructor.
- void [setFriction](#) ([real_type](#) _Friction)
Set friction coefficient.
- [real_type](#) [getFriction](#) (void) const
Get friction coefficient on the face.
- void [setVertices](#) ([vec3](#) const _Vertices[3])
Set new vertices and update bounding box domain.
- void [setVertices](#) ([vec3](#) const &Vertex0, [vec3](#) const &Vertex1, [vec3](#) const &Vertex2)
Set new vertices then update bounding box domain and normal versor.
- [vec3](#) const & [getNormal](#) (void) const
Get normal versor.
- [vec3](#) const & [getVertex](#) (unsigned i) const
Get i-th vertex.
- [BBox2D](#) const & [getBBox](#) (void) const
Get [Triangle3D](#) bonding box [BBox2D](#).
- void [print](#) ([ostream_type](#) &stream) const
Print vertices data.
- bool [intersectRay](#) ([vec3](#) const &RayOrigin, [vec3](#) const &RayDirection, [vec3](#) &IntPt) const
- [int_type](#) [intersectEdgePlane](#) ([vec3](#) const &PlaneN, [vec3](#) const &PlaneP, [int_type](#) const Edge, [vec3](#) &IntPt1, [vec3](#) &IntPt2) const
- bool [intersectPlane](#) ([vec3](#) const &PlaneN, [vec3](#) const &PlaneP, std::vector< [vec3](#) > &IntPts) const

Protected Attributes

- [vec3 Vertices](#) [3]
Vertices reference vector.
- [vec3 Normal](#)
Triangle normal versor.
- [BBox2D TriangleBBox](#)
Triangle 2D bounding box (XY plane)

6.13.1 Detailed Description

3D triangles for road representation

6.13.2 Constructor & Destructor Documentation

6.13.2.1 TriangleRoad()

```
TireGround::RDF::TriangleRoad::TriangleRoad (
    vec3 const &_Vertices[3],
    real\_type _Friction ) [inline]
```

Variable set constructor.

Parameters

<i>_Vertices</i>	Vertices reference vector
<i>_Friction</i>	Friction coefficient

6.13.3 Member Function Documentation

6.13.3.1 intersectEdgePlane()

```
int\_type TireGround::RDF::Triangle3D::intersectEdgePlane (
    vec3 const & PlaneN,
    vec3 const & PlaneP,
    int\_type const Edge,
    vec3 & IntPt1,
    vec3 & IntPt2 ) const [inherited]
```

Check if an edge of the [Triangle3D](#) object hits a and find the intersection point

Parameters

<i>PlaneN</i>	Plane normal vector
<i>PlaneP</i>	Plane known point
<i>Edge</i>	Triangle edge number (0:2)
<i>IntPt1</i>	Intersection point 1
<i>IntPt2</i>	Intersection point 2

6.13.3.2 intersectPlane()

```
bool TireGround::RDF::Triangle3D::intersectPlane (
    vec3 const & PlaneN,
    vec3 const & PlaneP,
    std::vector< vec3 > & IntPts ) const [inherited]
```

Check if a plane intersects a [Triangle3D](#) object and find the intersection points

Parameters

<i>PlaneN</i>	Plane normal vector
<i>PlaneP</i>	Plane known point
<i>IntPts</i>	Intersection points

6.13.3.3 intersectRay()

```
bool TireGround::RDF::Triangle3D::intersectRay (
    vec3 const & RayOrigin,
    vec3 const & RayDirection,
    vec3 & IntPt ) const [inherited]
```

Check if a ray hits a [Triangle3D](#) object through Möller-Trumbore intersection algorithm

Parameters

<i>RayOrigin</i>	Ray origin position
<i>RayDirection</i>	Ray direction vector
<i>IntPt</i>	Intersection point

6.13.3.4 print()

```
void TireGround::RDF::Triangle3D::print (
    ostream_type & stream ) const [inline], [inherited]
```

Print vertices data.

Parameters

<i>stream</i>	Output stream type
---------------	--------------------

6.13.3.5 setFriction()

```
void TireGround::RDF::TriangleRoad::setFriction (
    real_type _Friction ) [inline]
```

Set friction coefficient.

Parameters

<i>_Friction</i>	New friction coefficient
------------------	--------------------------

6.13.3.6 setVertices() [1/2]

```
void TireGround::RDF::Triangle3D::setVertices (
    vec3 const _Vertices[3] ) [inline], [inherited]
```

Set new vertices and update bounding box domain.

Parameters

<i>_Vertices</i>	Vertices reference vector
------------------	---------------------------

6.13.3.7 setVertices() [2/2]

```
void TireGround::RDF::Triangle3D::setVertices (
    vec3 const & Vertex0,
    vec3 const & Vertex1,
    vec3 const & Vertex2 ) [inline], [inherited]
```

Set new vertices then update bounding box domain and normal versor.

Parameters

<i>Vertex0</i>	Vertex 1
<i>Vertex1</i>	Vertex 2
<i>Vertex2</i>	Vertex 3

The documentation for this class was generated from the following file:

- include/RoadRDF.hh

Index

- BBox2D
 - TireGround::RDF::BBox2D, 20
- contactPlane
 - TireGround::Disk, 22
- contactTriangles
 - TireGround::Disk, 22
- Disk
 - TireGround::Disk, 21
- ETRTO
 - TireGround::ETRTO, 26
- evaluateContact
 - TireGround::MagicFormula, 30
 - TireGround::Tire, 63
- firstToken
 - TireGround::RDF::algorithms, 17
- fourPointsSampling
 - TireGround::MagicFormula, 30
- getArea
 - TireGround::MagicFormula, 30
 - TireGround::MultiDisk, 44
 - TireGround::Tire, 64
- getDiskFriction
 - TireGround::MultiDisk, 45
- getDiskMFpoint
 - TireGround::MultiDisk, 45
- getDiskMFpointRF
 - TireGround::MultiDisk, 45
- getDiskNormal
 - TireGround::MultiDisk, 46
- getDiskOriginXYZ
 - TireGround::MultiDisk, 46
- getDiskRho
 - TireGround::MultiDisk, 46
- getElement
 - TireGround::RDF::algorithms, 17
- getEulerAngleX
 - TireGround::MagicFormula, 31
 - TireGround::MultiDisk, 47
 - TireGround::ReferenceFrame, 56
 - TireGround::Tire, 64
- getEulerAngleY
 - TireGround::MagicFormula, 31
 - TireGround::MultiDisk, 47
- getEulerAngleZ
 - TireGround::MagicFormula, 31
 - TireGround::MultiDisk, 47
 - TireGround::ReferenceFrame, 56
 - TireGround::Tire, 64
- getFriction
 - TireGround::MagicFormula, 31
 - TireGround::MultiDisk, 47
 - TireGround::Tire, 65
- getLineArea
 - TireGround::Disk, 23
- getMFeffectiveRF
 - TireGround::MultiDisk, 48
- getMFeffectiveR
 - TireGround::MultiDisk, 48
- getMFeffectiveY
 - TireGround::MultiDisk, 48
- getMFpoint
 - TireGround::MagicFormula, 31, 32
 - TireGround::MultiDisk, 48
 - TireGround::Tire, 65
- getMFpointRF
 - TireGround::MagicFormula, 32
 - TireGround::MultiDisk, 49
 - TireGround::Tire, 66
- getNormal
 - TireGround::MagicFormula, 32, 33
 - TireGround::MultiDisk, 49
 - TireGround::Tire, 66
- getRelativeCamber
 - TireGround::MagicFormula, 33
 - TireGround::MultiDisk, 50
 - TireGround::Tire, 67
- getRho
 - TireGround::MagicFormula, 33, 34
 - TireGround::MultiDisk, 50
 - TireGround::Tire, 67
- getVolume
 - TireGround::MagicFormula, 34
 - TireGround::MultiDisk, 51
 - TireGround::Tire, 68
- intersectAABBtree
 - TireGround::RDF::MeshSurface, 38
- intersectBBox

- TireGround::RDF::MeshSurface, 39
- intersectEdgePlane
 - TireGround::RDF::Triangle3D, 73
 - TireGround::RDF::TriangleRoad, 77
- intersectPlane
 - TireGround::Disk, 23
 - TireGround::RDF::Triangle3D, 73
 - TireGround::RDF::TriangleRoad, 77
- intersectPointSegment
 - TireGround::algorithms, 13
- intersectRay
 - TireGround::RDF::Triangle3D, 74
 - TireGround::RDF::TriangleRoad, 78
- intersectRayPlane
 - TireGround::algorithms, 14
- intersectSegment
 - TireGround::Disk, 23
- isPointInside
 - TireGround::Disk, 24
- LoadFile
 - TireGround::RDF::MeshSurface, 39
- MagicFormula
 - TireGround::MagicFormula, 29
- mean
 - TireGround::algorithms, 14
- MeshSurface
 - TireGround::RDF::MeshSurface, 38
- minmax_XY
 - TireGround::algorithms, 14
- MultiDisk
 - TireGround::MultiDisk, 43, 44
- pointSampling
 - TireGround::MagicFormula, 34
 - TireGround::MultiDisk, 51
 - TireGround::Tire, 68
- print
 - TireGround::ETRTO, 26
 - TireGround::MagicFormula, 35
 - TireGround::MultiDisk, 52
 - TireGround::RDF::BBox2D, 20
 - TireGround::RDF::Triangle3D, 74
 - TireGround::RDF::TriangleRoad, 78
 - TireGround::Tire, 69
- printData
 - TireGround::RDF::MeshSurface, 39
- printETRTOGeometry
 - TireGround::MagicFormula, 35
 - TireGround::MultiDisk, 52
 - TireGround::Tire, 69
- ReferenceFrame
 - TireGround::ReferenceFrame, 55
- SamplingGrid
 - TireGround::SamplingGrid, 58
- segmentArea
 - TireGround::Disk, 24
- segmentLength
 - TireGround::Disk, 24
- set
 - TireGround::Disk, 24
 - TireGround::RDF::MeshSurface, 39
 - TireGround::ReferenceFrame, 56
 - TireGround::SamplingGrid, 58
- setDiskOriginXZ
 - TireGround::MultiDisk, 52
- setFriction
 - TireGround::RDF::TriangleRoad, 78
- setOrigin
 - TireGround::MagicFormula, 35
 - TireGround::MultiDisk, 53
 - TireGround::ReferenceFrame, 56
 - TireGround::Tire, 69
- setOriginXZ
 - TireGround::Disk, 25
- setReferenceFrame
 - TireGround::MagicFormula, 36
 - TireGround::MultiDisk, 53
 - TireGround::Tire, 69
- setRotationMatrix
 - TireGround::MagicFormula, 36
 - TireGround::MultiDisk, 53
 - TireGround::ReferenceFrame, 56
 - TireGround::Tire, 70
- setSwitchNumber
 - TireGround::SamplingGrid, 59
- setTotalTransformationMatrix
 - TireGround::MagicFormula, 36
 - TireGround::MultiDisk, 53
 - TireGround::ReferenceFrame, 57
 - TireGround::Tire, 70
- setVertices
 - TireGround::RDF::Triangle3D, 74, 75
 - TireGround::RDF::TriangleRoad, 79
- setup
 - TireGround::MagicFormula, 36, 37
 - TireGround::MultiDisk, 54
 - TireGround::Tire, 70
- Shadow
 - TireGround::Shadow, 59
- split
 - TireGround::RDF::algorithms, 17
- tail
 - TireGround::RDF::algorithms, 17
- TicToc, 60
- Tire
 - TireGround::Tire, 63
- TireGround, 11
- TireGround::Disk, 20

- contactPlane, 22
- contactTriangles, 22
- Disk, 21
- getLineArea, 23
- intersectPlane, 23
- intersectSegment, 23
- isPointInside, 24
- segmentArea, 24
- segmentLength, 24
- set, 24
- setOriginXZ, 25
- y, 25
- TireGround::ETRTO, 25
 - ETRTO, 26
 - print, 26
- TireGround::MagicFormula, 26
 - evaluateContact, 30
 - fourPointsSampling, 30
 - getArea, 30
 - getEulerAngleX, 31
 - getEulerAngleY, 31
 - getEulerAngleZ, 31
 - getFriction, 31
 - getMFpoint, 31, 32
 - getMFpointRF, 32
 - getNormal, 32, 33
 - getRelativeCamber, 33
 - getRho, 33, 34
 - getVolume, 34
 - MagicFormula, 29
 - pointSampling, 34
 - print, 35
 - printETRTOGeometry, 35
 - setOrigin, 35
 - setReferenceFrame, 36
 - setRotationMatrix, 36
 - setTotalTransformationMatrix, 36
 - setup, 36, 37
- TireGround::MultiDisk, 40
 - getArea, 44
 - getDiskFriction, 45
 - getDiskMFpoint, 45
 - getDiskMFpointRF, 45
 - getDiskNormal, 46
 - getDiskOriginXYZ, 46
 - getDiskRho, 46
 - getEulerAngleX, 47
 - getEulerAngleY, 47
 - getEulerAngleZ, 47
 - getFriction, 47
 - getMFeffectiveRF, 48
 - getMFeffectiveR, 48
 - getMFeffectiveY, 48
 - getMFpoint, 48
 - getMFpointRF, 49
 - getNormal, 49
 - getRelativeCamber, 50
 - getRho, 50
 - getVolume, 51
 - MultiDisk, 43, 44
 - pointSampling, 51
 - print, 52
 - printETRTOGeometry, 52
 - setDiskOriginXZ, 52
 - setOrigin, 53
 - setReferenceFrame, 53
 - setRotationMatrix, 53
 - setTotalTransformationMatrix, 53
 - setup, 54
- TireGround::RDF::BBox2D, 19
 - BBox2D, 20
 - print, 20
 - updateBBox2D, 20
- TireGround::RDF::MeshSurface, 37
 - intersectAABBtree, 38
 - intersectBBox, 39
 - LoadFile, 39
 - MeshSurface, 38
 - printData, 39
 - set, 39
- TireGround::RDF::Triangle3D, 71
 - intersectEdgePlane, 73
 - intersectPlane, 73
 - intersectRay, 74
 - print, 74
 - setVertices, 74, 75
 - Triangle3D, 73
- TireGround::RDF::TriangleRoad, 75
 - intersectEdgePlane, 77
 - intersectPlane, 77
 - intersectRay, 78
 - print, 78
 - setFriction, 78
 - setVertices, 79
 - TriangleRoad, 77
- TireGround::RDF::algorithms, 16
 - firstToken, 17
 - getElement, 17
 - split, 17
 - tail, 17
- TireGround::RDF, 16
- TireGround::ReferenceFrame, 54
 - getEulerAngleX, 56
 - getEulerAngleY, 56
 - getEulerAngleZ, 56
 - ReferenceFrame, 55
 - set, 56
 - setOrigin, 56
 - setRotationMatrix, 56
 - setTotalTransformationMatrix, 57
- TireGround::SamplingGrid, 57
 - SamplingGrid, 58

- set, [58](#)
 - setSwitchNumber, [59](#)
- TireGround::Shadow, [59](#)
 - Shadow, [59](#)
 - update, [60](#)
- TireGround::Tire, [61](#)
 - evaluateContact, [63](#)
 - getArea, [64](#)
 - getEulerAngleX, [64](#)
 - getEulerAngleY, [64](#)
 - getEulerAngleZ, [64](#)
 - getFriction, [65](#)
 - getMFpoint, [65](#)
 - getMFpointRF, [66](#)
 - getNormal, [66](#)
 - getRelativeCamber, [67](#)
 - getRho, [67](#)
 - getVolume, [68](#)
 - pointSampling, [68](#)
 - print, [69](#)
 - printETRTOGeometry, [69](#)
 - setOrigin, [69](#)
 - setReferenceFrame, [69](#)
 - setRotationMatrix, [70](#)
 - setTotalTransformationMatrix, [70](#)
 - setup, [70](#)
 - Tire, [63](#)
- TireGround::algorithms, [13](#)
 - intersectPointSegment, [13](#)
 - intersectRayPlane, [14](#)
 - mean, [14](#)
 - minmax_XY, [14](#)
 - trapezoidArea, [15](#)
 - weightedMean, [15](#)
- trapezoidArea
 - TireGround::algorithms, [15](#)
- Triangle3D
 - TireGround::RDF::Triangle3D, [73](#)
- TriangleRoad
 - TireGround::RDF::TriangleRoad, [77](#)
- update
 - TireGround::Shadow, [60](#)
- updateBBox2D
 - TireGround::RDF::BBox2D, [20](#)
- weightedMean
 - TireGround::algorithms, [15](#)
- y
 - TireGround::Disk, [25](#)