

TireGround

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	PatchTire Namespace Reference	11
5.1.1	Detailed Description	11
5.2	PatchTire::algorithms Namespace Reference	11
5.2.1	Detailed Description	12
5.2.2	Function Documentation	12
5.2.2.1	intersectPointSegment()	12
5.2.2.2	intersectRayPlane()	12
5.2.2.3	minmax_XY() [1/2]	13
5.2.2.4	minmax_XY() [2/2]	13
5.2.2.5	trapezoidArea()	13
5.2.2.6	weightedMean() [1/2]	14
5.2.2.7	weightedMean() [2/2]	14
5.3	RDF Namespace Reference	14
5.3.1	Detailed Description	15
5.4	RDF::algorithms Namespace Reference	15
5.4.1	Detailed Description	15
5.4.2	Function Documentation	15
5.4.2.1	firstToken()	15
5.4.2.2	getElement()	16
5.4.2.3	split()	16
5.4.2.4	tail()	16
5.5	TireGround Namespace Reference	16
5.5.1	Detailed Description	17

6	Class Documentation	19
6.1	RDF::BBox2D Class Reference	19
6.1.1	Detailed Description	19
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	PatchTire::Disk Class Reference	20
6.2.1	Detailed Description	21
6.2.2	Constructor & Destructor Documentation	21
6.2.2.1	Disk()	21
6.2.3	Member Function Documentation	21
6.2.3.1	contactPlane()	22
6.2.3.2	contactTriangles()	22
6.2.3.3	set()	22
6.2.3.4	setOriginXZ()	23
6.3	PatchTire::ETRTO Class Reference	23
6.3.1	Detailed Description	23
6.3.2	Constructor & Destructor Documentation	23
6.3.2.1	ETRTO()	23
6.3.3	Member Function Documentation	24
6.3.3.1	print()	24
6.4	PatchTire::MagicFormula Class Reference	24
6.4.1	Detailed Description	27
6.4.2	Constructor & Destructor Documentation	27
6.4.2.1	MagicFormula()	27
6.4.3	Member Function Documentation	27
6.4.3.1	evaluateContact()	27
6.4.3.2	fourPointsSampling()	28
6.4.3.3	getArea() [1/2]	28
6.4.3.4	getArea() [2/2]	28
6.4.3.5	getEulerAngleX()	28
6.4.3.6	getEulerAngleY()	28
6.4.3.7	getEulerAngleZ()	29
6.4.3.8	getFriction() [1/2]	29
6.4.3.9	getFriction() [2/2]	29
6.4.3.10	getMFpoint() [1/2]	29
6.4.3.11	getMFpoint() [2/2]	29
6.4.3.12	getMFpointRF() [1/2]	30
6.4.3.13	getMFpointRF() [2/2]	30
6.4.3.14	getNormal() [1/2]	30
6.4.3.15	getNormal() [2/2]	30
6.4.3.16	getRelativeCamber()	31

6.4.3.17	getRho() [1/2]	31
6.4.3.18	getRho() [2/2]	31
6.4.3.19	getRhoDot() [1/2]	31
6.4.3.20	getRhoDot() [2/2]	32
6.4.3.21	getVolume() [1/2]	32
6.4.3.22	getVolume() [2/2]	32
6.4.3.23	pointSampling()	33
6.4.3.24	print()	33
6.4.3.25	printETRTOGeometry()	33
6.4.3.26	setOrigin()	34
6.4.3.27	setReferenceFrame()	34
6.4.3.28	setRotationMatrix()	34
6.4.3.29	setTotalTransformationMatrix()	34
6.4.3.30	setup()	35
6.5	RDF::MeshSurface Class Reference	35
6.5.1	Detailed Description	36
6.5.2	Constructor & Destructor Documentation	36
6.5.2.1	MeshSurface() [1/2]	36
6.5.2.2	MeshSurface() [2/2]	36
6.5.3	Member Function Documentation	36
6.5.3.1	intersectAABBtree()	36
6.5.3.2	intersectBBox()	36
6.5.3.3	LoadFile()	37
6.5.3.4	printData()	37
6.5.3.5	set()	37
6.6	PatchTire::MultiDisk Class Reference	37
6.6.1	Detailed Description	40
6.6.2	Constructor & Destructor Documentation	40
6.6.2.1	MultiDisk() [1/3]	40
6.6.2.2	MultiDisk() [2/3]	41
6.6.2.3	MultiDisk() [3/3]	41
6.6.3	Member Function Documentation	42
6.6.3.1	getArea() [1/2]	42
6.6.3.2	getArea() [2/2]	42
6.6.3.3	getDiskFriction()	42
6.6.3.4	getDiskMFpoint()	43
6.6.3.5	getDiskMFpointRF()	43
6.6.3.6	getDiskNormal()	43
6.6.3.7	getDiskOriginXYZ() [1/2]	43
6.6.3.8	getDiskOriginXYZ() [2/2]	44
6.6.3.9	getDiskRho()	44
6.6.3.10	getDiskRhoDot()	44
6.6.3.11	getEulerAngleX()	44
6.6.3.12	getEulerAngleY()	45

6.6.3.13	getEulerAngleZ()	45
6.6.3.14	getFriction() [1/2]	45
6.6.3.15	getFriction() [2/2]	45
6.6.3.16	getMFpoint() [1/2]	45
6.6.3.17	getMFpoint() [2/2]	46
6.6.3.18	getMFpointRF() [1/2]	46
6.6.3.19	getMFpointRF() [2/2]	46
6.6.3.20	getNormal() [1/2]	46
6.6.3.21	getNormal() [2/2]	47
6.6.3.22	getRelativeCamber()	47
6.6.3.23	getRho() [1/2]	47
6.6.3.24	getRho() [2/2]	47
6.6.3.25	getRhoDot() [1/2]	48
6.6.3.26	getRhoDot() [2/2]	48
6.6.3.27	getVolume() [1/2]	48
6.6.3.28	getVolume() [2/2]	48
6.6.3.29	pointSampling()	50
6.6.3.30	print()	50
6.6.3.31	printETRTGeometry()	50
6.6.3.32	setDiskOriginXZ() [1/2]	51
6.6.3.33	setDiskOriginXZ() [2/2]	51
6.6.3.34	setOrigin()	51
6.6.3.35	setReferenceFrame()	51
6.6.3.36	setRotationMatrix()	52
6.6.3.37	setTotalTransformationMatrix()	52
6.6.3.38	setup()	52
6.7	PatchTire::ReferenceFrame Class Reference	52
6.7.1	Detailed Description	53
6.7.2	Constructor & Destructor Documentation	53
6.7.2.1	ReferenceFrame()	53
6.7.3	Member Function Documentation	54
6.7.3.1	getEulerAngleX()	54
6.7.3.2	getEulerAngleY()	54
6.7.3.3	getEulerAngleZ()	54
6.7.3.4	set()	54
6.7.3.5	setOrigin()	54
6.7.3.6	setRotationMatrix()	54
6.7.3.7	setTotalTransformationMatrix()	55
6.8	PatchTire::SamplingGrid Class Reference	55
6.8.1	Detailed Description	55
6.8.2	Constructor & Destructor Documentation	56
6.8.2.1	SamplingGrid() [1/2]	56
6.8.2.2	SamplingGrid() [2/2]	56
6.8.3	Member Function Documentation	56

6.8.3.1	set() [1/2]	56
6.8.3.2	set() [2/2]	57
6.8.3.3	setSwitchNumber()	57
6.9	PatchTire::Shadow Class Reference	57
6.9.1	Detailed Description	57
6.9.2	Constructor & Destructor Documentation	57
6.9.2.1	Shadow()	58
6.9.3	Member Function Documentation	58
6.9.3.1	update()	58
6.10	TicToc Class Reference	58
6.11	PatchTire::Tire Class Reference	58
6.11.1	Detailed Description	61
6.11.2	Constructor & Destructor Documentation	61
6.11.2.1	Tire()	61
6.11.3	Member Function Documentation	61
6.11.3.1	evaluateContact()	61
6.11.3.2	getArea() [1/2]	62
6.11.3.3	getArea() [2/2]	62
6.11.3.4	getEulerAngleX()	62
6.11.3.5	getEulerAngleY()	62
6.11.3.6	getEulerAngleZ()	62
6.11.3.7	getFriction() [1/2]	62
6.11.3.8	getFriction() [2/2]	63
6.11.3.9	getMFpoint() [1/2]	63
6.11.3.10	getMFpoint() [2/2]	63
6.11.3.11	getMFpointRF() [1/2]	63
6.11.3.12	getMFpointRF() [2/2]	64
6.11.3.13	getNormal() [1/2]	64
6.11.3.14	getNormal() [2/2]	64
6.11.3.15	getRelativeCamber()	64
6.11.3.16	getRho() [1/2]	65
6.11.3.17	getRho() [2/2]	65
6.11.3.18	getRhoDot() [1/2]	65
6.11.3.19	getRhoDot() [2/2]	66
6.11.3.20	getVolume() [1/2]	66
6.11.3.21	getVolume() [2/2]	66
6.11.3.22	pointSampling()	66
6.11.3.23	print()	67
6.11.3.24	printETRTOGeometry()	67
6.11.3.25	setOrigin()	67
6.11.3.26	setReferenceFrame()	68
6.11.3.27	setRotationMatrix()	68
6.11.3.28	setTotalTransformationMatrix()	68
6.11.3.29	setup()	68

6.12	RDF::Triangle3D Class Reference	69
6.12.1	Detailed Description	70
6.12.2	Constructor & Destructor Documentation	70
6.12.2.1	Triangle3D()	70
6.12.3	Member Function Documentation	70
6.12.3.1	intersectEdgePlane()	70
6.12.3.2	intersectPlane()	71
6.12.3.3	intersectRay()	71
6.12.3.4	print()	71
6.12.3.5	setVertices() [1/2]	73
6.12.3.6	setVertices() [2/2]	73
6.13	RDF::TriangleRoad Class Reference	73
6.13.1	Detailed Description	75
6.13.2	Constructor & Destructor Documentation	75
6.13.2.1	TriangleRoad()	75
6.13.3	Member Function Documentation	75
6.13.3.1	intersectEdgePlane()	75
6.13.3.2	intersectPlane()	76
6.13.3.3	intersectRay()	76
6.13.3.4	print()	76
6.13.3.5	setFriction()	77
6.13.3.6	setVertices() [1/2]	77
6.13.3.7	setVertices() [2/2]	77
	Index	79

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.

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

2. Initialize the MagicFormula tire model.

```
PatchTire::Tire* TireSD = new PatchTire::MagicFormula(  
    SectionWidth, // [mm]  
    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)  
PatchTire::vec3 N;  
PatchTire::vec3 P;  
PatchTire::real_type Friction;  
PatchTire::real_type Rho;  
PatchTire::real_type RhoDot;  
PatchTire::real_type RelativeCamber;  
PatchTire::real_type Friction;  
PatchTire::real_type Area;  
PatchTire::real_type Volume;  
PatchTire::real_type RelativeCamber;  
  
// Data extraction (for real numbers)  
TireSD->getNormal(N);  
TireSD->getPoint(P);
```

```

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

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

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

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

```

MultiDisk tire model usage

1. Load .rdf file.

```

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

```

2. Initialize the MultiDisk tire model:

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

```

PatchTire::Tire* TireMD = new PatchTire::MultiDisk(
    SectionWidth, // [mm]
    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).

```

PatchTire::Tire* TireMD = new PatchTire::MultiDisk(
    SectionWidth, // [mm]
    AspectRatio,  // [%]
    RimDiameter,  // [in]
    SideRadius,   // Sidewall radius [mm]
    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.

```

PatchTire::Tire* TireMD = new PatchTire::MultiDisk(
    SectionWidth, // [mm]
    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)
PatchTire::vec3 N;
PatchTire::vec3 P;
PatchTire::real_type Friction;
PatchTire::real_type Rho;
PatchTire::real_type RhoDot;
PatchTire::real_type RelativeCamber;
PatchTire::real_type Friction;
PatchTire::real_type Area;
PatchTire::real_type Volume;
PatchTire::real_type RelativeCamber;

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

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

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

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

```


Chapter 2

Namespace Index

2.1 Namespace List

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

PatchTire	
Tire computations routines	11
PatchTire::algorithms	
Algorithms for tire computations routine	11
RDF	
RDF mesh computations routines	14
RDF::algorithms	
Algorithms for RDF mesh computations routine	15
TireGround	
Typedefs for tire computations routine	16

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

RDF::BBox2D	19
PatchTire::Disk	20
PatchTire::ETRTO	23
RDF::MeshSurface	35
PatchTire::ReferenceFrame	52
PatchTire::SamplingGrid	55
PatchTire::Shadow	57
TicToc	58
PatchTire::Tire	58
PatchTire::MagicFormula	24
PatchTire::MultiDisk	37
RDF::Triangle3D	69
RDF::TriangleRoad	73

Chapter 4

Class Index

4.1 Class List

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

RDF::BBox2D	
2D Bounding Box class	19
PatchTire::Disk	
Tire disk	20
PatchTire::ETRTO	
Tire ETRTO denomination	23
PatchTire::MagicFormula	
Pacejka MagicFormula contact model	24
RDF::MeshSurface	
Mesh surface	35
PatchTire::MultiDisk	
Multi-disk tire contact model	37
PatchTire::ReferenceFrame	
Reference frame	52
PatchTire::SamplingGrid	
Patch evaluation precision	55
PatchTire::Shadow	
2D shadow (2D bounding box enhancement)	57
TicToc	58
PatchTire::Tire	
Base class for Tire models	58
RDF::Triangle3D	
3D triangle (pure geometrical description)	69
RDF::TriangleRoad	
3D triangles for road representation	73

Chapter 5

Namespace Documentation

5.1 PatchTire Namespace Reference

[Tire](#) computations routines.

Namespaces

- [algorithms](#)

Algorithms for tire computations routine.

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.

5.1.1 Detailed Description

[Tire](#) computations routines.

file: [PatchTire.hh](#)

5.2 PatchTire::algorithms Namespace Reference

Algorithms for tire computations routine.

Functions

- `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 minimum and maximum in XY plane for 3D vectors.
- `void minmax_XY (row_vec2 const &Points, vec2 &XYmin, vec2 &XYmax)`
Calculate minimum and maximum in XY plane for 2D vectors.
- `real_type trapezoidArea (real_type const Base_A, real_type const Base_B, real_type const Height)`
Calculate area of a trapezoid [m²].

5.2.1 Detailed Description

Algorithms for tire computations routine.

5.2.2 Function Documentation

5.2.2.1 intersectPointSegment()

```
bool PatchTire::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 PatchTire::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 minmax_XY() [1/2]

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

Calculate minumum 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

5.2.2.4 minmax_XY() [2/2]

```
void PatchTire::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

5.2.2.5 trapezoidArea()

```
real_type PatchTire::algorithms::trapezoidArea (
    real_type const Base_A,
    real_type const Base_B,
    real_type const Height ) [inline]
```

Calculate area of a trapeziod [m^2].

Parameters

<i>Base_A</i>	Base 1
<i>Base_B</i>	Base 2
<i>Height</i>	Heigth

5.2.2.6 `weightedMean()` [1/2]

```
real_type PatchTire::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)

5.2.2.7 `weightedMean()` [2/2]

```
vec3 PatchTire::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 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 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 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& 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 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 RDF::algorithms::tail (
    std::string const & in )
```

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

Parameters

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

5.5 TireGround Namespace Reference

Typedefs for tire computations routine.

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.

5.5.1 Detailed Description

Typedefs for tire computations routine.

file: [TireGround.hh](#)

Chapter 6

Class Documentation

6.1 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()

```
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 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 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 PatchTire::Disk Class Reference

[Tire](#) disk.

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

Public Member Functions

- [Disk](#) ([Disk](#) &&)=default

- *Enable `EE` 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.

6.2.1 Detailed Description

`Tire` disk.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Disk()

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

Variable set constructor.

Parameters

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

6.2.3 Member Function Documentation

6.2.3.1 contactPlane()

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

6.2.3.3 set()

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

Copy the [Disk](#) object.

Parameters

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

6.2.3.4 setOriginXZ()

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

Set origin on XZ plane.

Parameters

_OriginXZ	New origin on XZ plane
---------------------------	--------------------------

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

- `include/PatchTire.hh`

6.3 PatchTire::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()

```
PatchTire::ETRTO::ETRTO (
    real\_type _SectionWidth,
    real\_type _AspectRatio,
    real\_type _RimDiameter ) [inline]
```

Variable set constructor.

Parameters

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

6.3.3 Member Function Documentation

6.3.3.1 `print()`

```
void PatchTire::ETRT0::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:

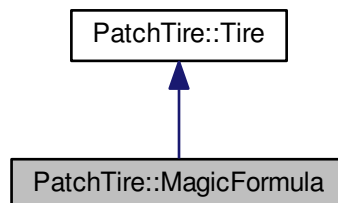
- `include/PatchTire.hh`

6.4 PatchTire::MagicFormula Class Reference

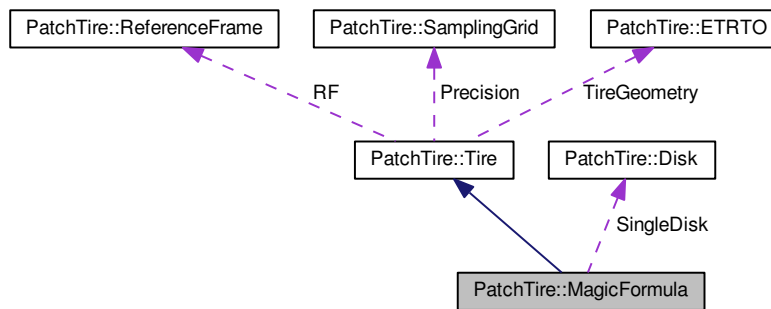
Pacejka [MagicFormula](#) contact model.

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

Inheritance diagram for PatchTire::MagicFormula:



Collaboration diagram for PatchTire::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) const override
- void [getRho](#) ([row_vecN](#) &Rho) const override
- void [getRhoDot](#) ([real_type](#) const &Rho, [real_type](#) const &Time, [real_type](#) &RhoDot) const override
Get contact depth time derivative [m/s].
- void [getRhoDot](#) ([row_vecN](#) const &Rho, [real_type](#) const &Time, [row_vecN](#) &RhoDot) const override
Get contact depth time derivative vector [m/s].
- void [getArea](#) ([real_type](#) &_Area) const override
Get approximated contact area on [Disk](#) plane [m²].
- void [getArea](#) ([row_vecN](#) &_Area) const override
Get approximated contact area 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 volume vector [m³].
- bool `setup` (`RDF::MeshSurface` &Mesh, `mat4` const &TM) override
Update current tire position and find contact parameters.
- void `print` (`ostream_type` &stream) const override
Print contact parameters.
- void `printETRTOGeometry` (`ostream_type` &stream) const
Display Tire ETRTO geometry data.
- 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 $\pm 0.1 * R$ along X and $\pm 0.3 * W$ 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 (not for Magic Formula)
- `vec3` `DiskPoint`
Contact point on undeformed Disk circumference (for Magic Formula)
- `real_type` `Friction`

- [Contact friction](#).
Contact friction.
- [real_type Area](#)
Contact area [m²].
- [SamplingGrid Precision](#)
Contact patch evaluating precision.
- [ETRTO TireGeometry](#)
Tire ETRTO denomination.
- [ReferenceFrame RF](#)
ReferenceFrame.

6.4.1 Detailed Description

Pacejka [MagicFormula](#) contact model.

6.4.2 Constructor & Destructor Documentation

6.4.2.1 MagicFormula()

```
PatchTire::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 [mm]
<i>AspectRatio</i>	Tire aspect ratio [%]
<i>RimDiameter</i>	Rim diameter [in]
<i>SwitchN</i>	Maximum RoadTriangles in the Tire Shadow (switch to sampling)

6.4.3 Member Function Documentation

6.4.3.1 evaluateContact()

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

Evaluate contact with RoadTriangles.

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.2 fourPointsSampling()

```
void PatchTire::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 PatchTire::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 [PatchTire::Tire](#).

6.4.3.4 getArea() [2/2]

```
void PatchTire::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 [PatchTire::Tire](#).

6.4.3.5 getEulerAngleX()

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

Get contact point friction.

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.9 getFriction() [2/2]

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

Get contact point friction vector.

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.10 getMFpoint() [1/2]

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

Get Magic Formula contact point.

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.11 getMFpoint() [2/2]

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

Get Magic Formula contact point vector.

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.12 getMFpointRF() [1/2]

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

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

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.13 getMFpointRF() [2/2]

```
void PatchTire::MagicFormula::getMFpointRF (
    row\_mat4 & MFpointRF ) const [inline], [override], [virtual]
```

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

Parameters

<code>MFpointRF</code>	Magic Formula ontact point reference frames vector
------------------------	--

Implements [PatchTire::Tire](#).

6.4.3.14 getNormal() [1/2]

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

Get contact normal versor.

Parameters

<code>Normal</code>	Contact point normal versor
---------------------	-----------------------------

Implements [PatchTire::Tire](#).

6.4.3.15 getNormal() [2/2]

```
void PatchTire::MagicFormula::getNormal (
    row\_vec3 & Normal ) const [inline], [override], [virtual]
```

Get contact normal versors vector.

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.16 getRelativeCamber()

```
void PatchTire::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 PatchTire::MagicFormula::getRho (
    real_type & Rho ) const [inline], [override], [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
------------	-----------------------

Implements [PatchTire::Tire](#).

6.4.3.18 getRho() [2/2]

```
void PatchTire::MagicFormula::getRho (
    row_vecN & Rho ) const [inline], [override], [virtual]
```

Get contact depth matrix [*m*]

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

Parameters

<i>Rho</i>	Depth matrix
------------	--------------

Implements [PatchTire::Tire](#).

6.4.3.19 getRhoDot() [1/2]

```
void PatchTire::MagicFormula::getRhoDot (
    real_type const & Rho,
```

```

    real_type const & Time,
    real_type & RhoDot ) const [inline], [override], [virtual]

```

Get contact depth time derivative [m/s].

Parameters

<i>Rho</i>	Previous time step Rho [m]
<i>Time</i>	Time step [s]
<i>RhoDot</i>	Penetration derivative [m/s]

Implements [PatchTire::Tire](#).

6.4.3.20 getRhoDot() [2/2]

```

void PatchTire::MagicFormula::getRhoDot (
    row_vecN const & Rho,
    real_type const & Time,
    row_vecN & RhoDot ) const [inline], [override], [virtual]

```

Get contact depth time derivative vector [m/s].

Parameters

<i>Rho</i>	Previous time step Rho [m]
<i>Time</i>	Time step [s]
<i>RhoDot</i>	Penetration derivative [m/s]

Implements [PatchTire::Tire](#).

6.4.3.21 getVolume() [1/2]

```

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

```

Get approximated contact volume [m^3].

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.22 getVolume() [2/2]

```

void PatchTire::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 [PatchTire::Tire](#).

6.4.3.23 pointSampling()

```
bool PatchTire::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

6.4.3.24 print()

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

Print contact parameters.

Parameters

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

Implements [PatchTire::Tire](#).

6.4.3.25 printETRTOGeometry()

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

Display [Tire ETRTO](#) geometry data.

Parameters

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

6.4.3.26 setOrigin()

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

Set a new tire origin.

Parameters

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

6.4.3.27 setReferenceFrame()

```
void PatchTire::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
------------	--

6.4.3.28 setRotationMatrix()

```
void PatchTire::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
-----------------------	-----------------

6.4.3.29 setTotalTransformationMatrix()

```
void PatchTire::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
-----------	---------------------------------

6.4.3.30 setup()

```
bool PatchTire::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 [PatchTire::Tire](#).

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

- include/PatchTire.hh

6.5 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::TriangleRoad_list](#) &TrianglesList) const
Update the local intersected TriangleRoad vector list.
- bool [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.

6.5.1 Detailed Description

Mesh surface.

6.5.2 Constructor & Destructor Documentation

6.5.2.1 MeshSurface() [1/2]

```
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]

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

Update the local intersected [TriangleRoad](#) vector list.

Parameters

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

6.5.3.2 intersectBBox()

```
bool 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 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 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 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:

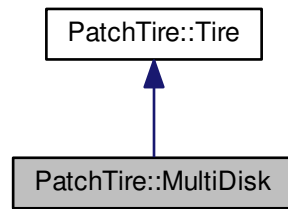
- include/RoadRDF.hh

6.6 PatchTire::MultiDisk Class Reference

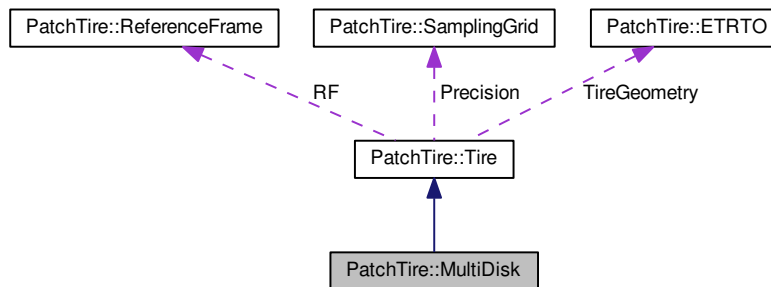
Multi-disk tire contact model.

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

Inheritance diagram for PatchTire::MultiDisk:



Collaboration diagram for PatchTire::MultiDisk:



Public Member Functions

- `~MultiDisk ()`
Default destructor.
- `MultiDisk (real_type const SectionWidth, real_type const AspectRatio, real_type const Rim↔Diameter, 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 Rim↔Diameter, 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 Rim↔Diameter, 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) const
- *Set disks origin (X, Y, Z).*
- void `setDiskOriginXZ` (int_type const i, vec2 &Origin) const
- *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 `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) const override
- void `getRho` (row_vecN &Rho) const override
- void `getDiskRho` (int_type const i, real_type &Rho) const
- void `getRhoDot` (real_type const &Rho, real_type const &Time, real_type &RhoDot) const override
- *Get contact depth time derivative [m/s].*
- void `getRhoDot` (row_vecN const &Rho, real_type const &Time, row_vecN &RhoDot) const override
- *Get contact depths derivative vector [m/s].*
- void `getDiskRhoDot` (int_type const i, real_type const &Rho, real_type const &Time, real_type &RhoDot) const
- *Get i-th Disk contact depth derivative [m/s].*
- 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].*
- bool `setup` (RDF::MeshSurface &Mesh, mat4 const &TM) override
- *Update current tire position and find contact parameters.*

- void `print` (`ostream_type` &stream) const override
Print contact parameters.
- void `printETRTOGeometry` (`ostream_type` &stream) const
Display Tire ETRTO geometry data.
- 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.

6.6.1 Detailed Description

Multi-disk tire contact model.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 MultiDisk() [1/3]

```
PatchTire::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>mm</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]

```
PatchTire::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>mm</i>]
<i>AspectRatio</i>	Tire aspect ratio [%]
<i>RimDiameter</i>	Rim diameter [<i>in</i>]
<i>SideRadius</i>	Sidewall radius [<i>mm</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]

```
PatchTire::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>mm</i>]
<i>AspectRatio</i>	Tire aspect ratio [%]
<i>RimDiameter</i>	Rim diameter [<i>in</i>]

Parameters

<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 PatchTire::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 [PatchTire::Tire](#).

6.6.3.2 `getArea()` [2/2]

```
void PatchTire::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 [<i>m</i> ²]
-----------------	--

Implements [PatchTire::Tire](#).

6.6.3.3 `getDiskFriction()`

```
void PatchTire::MultiDisk::getDiskFriction (
    int\_type const i,
    real\_type & _Friction ) const [inline]
```

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

Parameters

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

6.6.3.4 getDiskMFpoint()

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

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

Parameters

i	i -th Disk
$_DiskPoint$	Disk Magic Formula contact point

6.6.3.5 getDiskMFpointRF()

```
void PatchTire::MultiDisk::getDiskMFpointRF (
    int_type const i,
    mat4 & PointRF ) const [inline]
```

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

Parameters

i	i -th Disk
$PointRF$	Magic Formula contact point reference frame

6.6.3.6 getDiskNormal()

```
void PatchTire::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 PatchTire::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 PatchTire::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 PatchTire::MultiDisk::getDiskRho (
    int_type const i,
    real_type & Rho ) const [inline]
```

Get i -th [Disk](#) contact depth [m]

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

Parameters

i	i -th Disk
Rho	Disk contact depth

6.6.3.10 getDiskRhoDot()

```
void PatchTire::MultiDisk::getDiskRhoDot (
    int_type const i,
    real_type const & Rho,
    real_type const & Time,
    real_type & RhoDot ) const [inline]
```

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

Parameters

i	i -th Disk
Rho	Previous time step Rho [m]
$Time$	Time step [s]
$RhoDot$	Disk contact depth derivative [m/s]

6.6.3.11 getEulerAngleX()

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

Get current Euler angles [rad] for X -axis

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

6.6.3.12 getEulerAngleY()

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

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

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

6.6.3.13 getEulerAngleZ()

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

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

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

6.6.3.14 getFriction() [1/2]

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

Get area weighted mean contact friction.

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.15 getFriction() [2/2]

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

Get contact frictions vector.

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.16 getMFpoint() [1/2]

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

Get Magic Formula contact point.

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.17 getMFpoint() [2/2]

```
void PatchTire::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 [PatchTire::Tire](#).

6.6.3.18 getMFpointRF() [1/2]

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

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

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.19 getMFpointRF() [2/2]

```
void PatchTire::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 [PatchTire::Tire](#).

6.6.3.20 getNormal() [1/2]

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

Get contact normal mean versor.

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.21 getNormal() [2/2]

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

Get contact normal versors vector.

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.22 getRelativeCamber()

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

Get relative camber angle [*rad*].

Parameters

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

6.6.3.23 getRho() [1/2]

```
void PatchTire::MultiDisk::getRho (
    real_type & Rho ) const [inline], [override], [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
------------	-----------------------

Implements [PatchTire::Tire](#).

6.6.3.24 getRho() [2/2]

```
void PatchTire::MultiDisk::getRho (
    row_vecN & Rho ) const [inline], [override], [virtual]
```

Get contact depths vector [*m*]

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

Parameters

<i>Rho</i>	Contact depths vector
------------	-----------------------

Implements [PatchTire::Tire](#).

6.6.3.25 getRhoDot() [1/2]

```
void PatchTire::MultiDisk::getRhoDot (
    real_type const & Rho,
    real_type const & Time,
    real_type & RhoDot ) const [inline], [override], [virtual]
```

Get contact depth time derivative [m/s].

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.26 getRhoDot() [2/2]

```
void PatchTire::MultiDisk::getRhoDot (
    row_vecN const & Rho,
    real_type const & Time,
    row_vecN & RhoDot ) const [inline], [override], [virtual]
```

Get contact depths derivative vector [m/s].

Parameters

<i>Rho</i>	Previous time step Rho [m]
<i>Time</i>	Time step [s]
<i>RhoDot</i>	Contact depths derivative vector [m/s]

Implements [PatchTire::Tire](#).

6.6.3.27 getVolume() [1/2]

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

Get approximated contact volume [m^3].

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.28 getVolume() [2/2]

```
void PatchTire::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 [PatchTire::Tire](#).

6.6.3.29 pointSampling()

```
bool PatchTire::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 PatchTire::MultiDisk::print (
    ostream_type & stream ) const [override], [virtual]
```

Print contact parameters.

Parameters

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

Implements [PatchTire::Tire](#).

6.6.3.31 printETRTOGeometry()

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

Set disks origin (X, Y, Z).

Parameters

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

6.6.3.33 setDiskOriginXZ() [2/2]

```
void PatchTire::MultiDisk::setDiskOriginXZ (
    int_type const i,
    vec2 & Origin ) const [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 PatchTire::Tire::setOrigin (
    vec3 const & Origin ) [inline], [inherited]
```

Set a new tire origin.

Parameters

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

6.6.3.35 setReferenceFrame()

```
void PatchTire::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 PatchTire::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 PatchTire::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()

```
bool PatchTire::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 [PatchTire::Tire](#).

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

- `include/PatchTire.hh`

6.7 PatchTire::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()

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

Variable set constructor.

Parameters

<i>_Origin</i>	Origin position
<i>_RotationMatrix</i>	3x3 rotation matrix

6.7.3 Member Function Documentation

6.7.3.1 getEulerAngleX()

```
real_type PatchTire::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 PatchTire::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 PatchTire::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 PatchTire::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 PatchTire::ReferenceFrame::setOrigin (
    vec3 const & _Origin ) [inline]
```

Set origin position.

Parameters

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

6.7.3.6 setRotationMatrix()

```
void PatchTire::ReferenceFrame::setRotationMatrix (
```

```
mat3 const & _RotationMatrix ) [inline]
```

Set 3x3 rotation matrix.

Parameters

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

6.7.3.7 setTotalTransformationMatrix()

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

Set 4x4 total transformation matrix.

Parameters

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

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

- include/PatchTire.hh

6.8 PatchTire::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]

```
PatchTire::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]

```
PatchTire::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 PatchTire::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 PatchTire::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 PatchTire::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 PatchTire::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()

```
PatchTire::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 PatchTire::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:

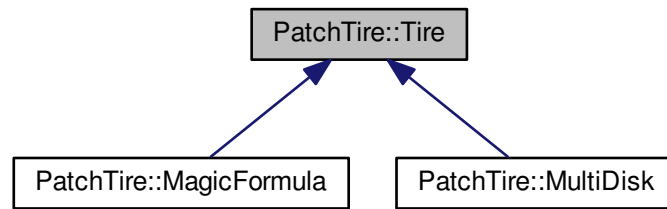
- include/TicToc.hh

6.11 PatchTire::Tire Class Reference

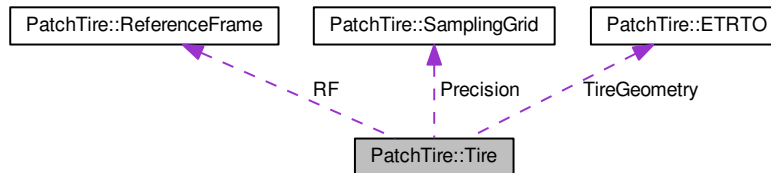
Base class for [Tire](#) models.

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

Inheritance diagram for PatchTire::Tire:



Collaboration diagram for PatchTire::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.
- `void setReferenceFrame (ReferenceFrame const &_RF)`
- `ReferenceFrame const &getReferenceFrame () 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 () const`
- `real_type getEulerAngleY () const`
- `real_type getEulerAngleZ () const`
- `void getRelativeCamber (real_type &RelativeCamber) const`
Get relative camber angle [rad].
- `int_type getDisksNumber () const`
Dimension of the contact points data structure (disks number)
- `virtual void getRho (real_type &Rho) const =0`

- virtual void `getRho` (`row_vecN` &Rho) const =0
- virtual void `getRhoDot` (`real_type` const &Rho, `real_type` const &Time, `real_type` &RhoDot) const =0
Get contact depth time derivative [m/s].
- virtual void `getRhoDot` (`row_vecN` const &Rho, `real_type` const &Time, `row_vecN` &RhoDot) const =0
Get contact depth time derivative vector [m/s].
- 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 [m²].*
- virtual void `getArea` (`row_vecN` &Area) const =0
*Get approximated contact areas vector on *Disk* plane [m²].*
- virtual void `getVolume` (`real_type` &Volume) const =0
Get approximated contact volume [m³].
- virtual void `getVolume` (`row_vecN` &_Volume) const =0
Get approximated contact volume [m³].
- 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 `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.

6.11.1 Detailed Description

Base class for [Tire](#) models.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 Tire()

```
PatchTire::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>mm</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 PatchTire::Tire::evaluateContact (
    RDF::TriangleRoad_list const & TriList ) [pure virtual]
```

Evaluate contact with RoadTriangles.

Parameters

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

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

6.11.3.2 getArea() [1/2]

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

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

Parameters

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

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

6.11.3.3 getArea() [2/2]

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

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

Parameters

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

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

6.11.3.4 getEulerAngleX()

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

Get contact point friction.

Parameters

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

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

6.11.3.8 getFriction() [2/2]

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

Get contact frictions vector.

Parameters

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

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

6.11.3.9 getMFpoint() [1/2]

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

Get Magic Formula contact point.

Parameters

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

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

6.11.3.10 getMFpoint() [2/2]

```
virtual void PatchTire::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 [PatchTire::MultiDisk](#), and [PatchTire::MagicFormula](#).

6.11.3.11 getMFpointRF() [1/2]

```
virtual void PatchTire::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 [PatchTire::MultiDisk](#), and [PatchTire::MagicFormula](#).

6.11.3.12 getMFpointRF() [2/2]

```
virtual void PatchTire::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 [PatchTire::MultiDisk](#), and [PatchTire::MagicFormula](#).

6.11.3.13 getNormal() [1/2]

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

Get contact normal versor.

Parameters

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

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

6.11.3.14 getNormal() [2/2]

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

Get contact normal versors vector.

Parameters

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

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

6.11.3.15 getRelativeCamber()

```
void PatchTire::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 PatchTire::Tire::getRho (
    real_type & Rho ) 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
------------	-----------------------

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

6.11.3.17 getRho() [2/2]

```
virtual void PatchTire::Tire::getRho (
    row_vecN & Rho ) const [pure virtual]
```

Get contact depth vector [*m*]

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

Parameters

<i>Rho</i>	Depth vector [<i>m</i>]
------------	---------------------------

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

6.11.3.18 getRhoDot() [1/2]

```
virtual void PatchTire::Tire::getRhoDot (
    real_type const & Rho,
    real_type const & Time,
    real_type & RhoDot ) const [pure virtual]
```

Get contact depth time derivative [*m/s*].

Parameters

<i>Rho</i>	Previous time step Rho [<i>m</i>]
<i>Time</i>	Time step [<i>s</i>]
<i>RhoDot</i>	Penetration derivative [<i>m/s</i>]

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

6.11.3.19 getRhoDot() [2/2]

```
virtual void PatchTire::Tire::getRhoDot (
    row_vecN const & Rho,
    real_type const & Time,
    row_vecN & RhoDot ) const [pure virtual]
```

Get contact depth time derivative vector [m/s].

Parameters

<i>Rho</i>	Previous time step Rho [m]
<i>Time</i>	Time step [s]
<i>RhoDot</i>	Penetration derivative [m/s]

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

6.11.3.20 getVolume() [1/2]

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

Get approximated contact volume [m^3].

Parameters

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

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

6.11.3.21 getVolume() [2/2]

```
virtual void PatchTire::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 [PatchTire::MultiDisk](#), and [PatchTire::MagicFormula](#).

6.11.3.22 pointSampling()

```
bool PatchTire::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
<i>SampledPt</i>	Intersection point
<i>TriFriction</i>	Intersected triangle friction
<i>TriNormal</i>	Intersected triangle normal

6.11.3.23 print()

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

Print contact parameters.

Parameters

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

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

6.11.3.24 printETRTOGeometry()

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

Display [Tire ETRTO](#) geometry data.

Parameters

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

6.11.3.25 setOrigin()

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

Set a new tire origin.

Parameters

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

6.11.3.26 setReferenceFrame()

```
void PatchTire::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
------------	--

6.11.3.27 setRotationMatrix()

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

Set a new 3x3 rotation matrix

Warning: Rotation matrix must be orthonormal!

Parameters

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

6.11.3.28 setTotalTransformationMatrix()

```
void PatchTire::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
-----------	---------------------------------

6.11.3.29 setup()

```
virtual bool PatchTire::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 [PatchTire::MultiDisk](#), and [PatchTire::MagicFormula](#).

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

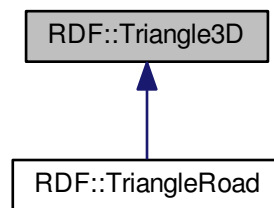
- `include/PatchTire.hh`

6.12 RDF::Triangle3D Class Reference

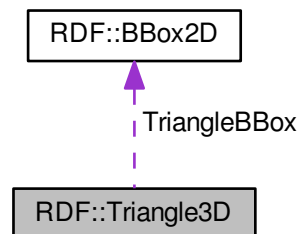
3D triangle (pure geometrical description)

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

Inheritance diagram for RDF::Triangle3D:



Collaboration diagram for 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` bounding 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()`

```
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 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

<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

6.12.3.2 intersectPlane()

```

bool 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 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 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 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 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:

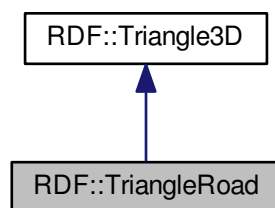
- include/RoadRDF.hh

6.13 RDF::TriangleRoad Class Reference

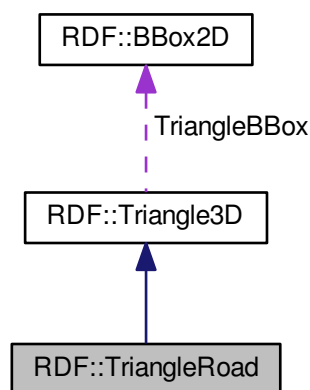
3D triangles for road representation

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

Inheritance diagram for `RDF::TriangleRoad`:



Collaboration diagram for `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()

```
RDF::TriangleRoad::TriangleRoad (
    vec3 const _Vertices[3],
    real_type _Friction ) [inline]
```

Variable set constructor.

Parameters

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

6.13.3 Member Function Documentation

6.13.3.1 intersectEdgePlane()

```
int_type 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 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 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 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 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 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 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
 - RDF::BBox2D, [20](#)
- contactPlane
 - PatchTire::Disk, [21](#)
- contactTriangles
 - PatchTire::Disk, [22](#)
- Disk
 - PatchTire::Disk, [21](#)
- ETRTO
 - PatchTire::ETRTO, [23](#)
- evaluateContact
 - PatchTire::MagicFormula, [27](#)
 - PatchTire::Tire, [61](#)
- firstToken
 - RDF::algorithms, [15](#)
- fourPointsSampling
 - PatchTire::MagicFormula, [27](#)
- getArea
 - PatchTire::MagicFormula, [28](#)
 - PatchTire::MultiDisk, [42](#)
 - PatchTire::Tire, [61](#), [62](#)
- getDiskFriction
 - PatchTire::MultiDisk, [42](#)
- getDiskMFpoint
 - PatchTire::MultiDisk, [42](#)
- getDiskMFpointRF
 - PatchTire::MultiDisk, [43](#)
- getDiskNormal
 - PatchTire::MultiDisk, [43](#)
- getDiskOriginXYZ
 - PatchTire::MultiDisk, [43](#)
- getDiskRho
 - PatchTire::MultiDisk, [44](#)
- getDiskRhoDot
 - PatchTire::MultiDisk, [44](#)
- getElement
 - RDF::algorithms, [16](#)
- getEulerAngleX
 - PatchTire::MagicFormula, [28](#)
 - PatchTire::MultiDisk, [44](#)
 - PatchTire::ReferenceFrame, [54](#)
 - PatchTire::Tire, [62](#)
- getEulerAngleY
 - PatchTire::MagicFormula, [28](#)
 - PatchTire::MultiDisk, [44](#)
 - PatchTire::ReferenceFrame, [54](#)
 - PatchTire::Tire, [62](#)
- getEulerAngleZ
 - PatchTire::MagicFormula, [28](#)
 - PatchTire::MultiDisk, [45](#)
 - PatchTire::ReferenceFrame, [54](#)
 - PatchTire::Tire, [62](#)
- getFriction
 - PatchTire::MagicFormula, [29](#)
 - PatchTire::MultiDisk, [45](#)
 - PatchTire::Tire, [62](#), [63](#)
- getMFpoint
 - PatchTire::MagicFormula, [29](#)
 - PatchTire::MultiDisk, [45](#)
 - PatchTire::Tire, [63](#)
- getMFpointRF
 - PatchTire::MagicFormula, [30](#)
 - PatchTire::MultiDisk, [46](#)
 - PatchTire::Tire, [63](#), [64](#)
- getNormal
 - PatchTire::MagicFormula, [30](#)
 - PatchTire::MultiDisk, [46](#)
 - PatchTire::Tire, [64](#)
- getRelativeCamber
 - PatchTire::MagicFormula, [31](#)
 - PatchTire::MultiDisk, [47](#)
 - PatchTire::Tire, [64](#)
- getRho
 - PatchTire::MagicFormula, [31](#)
 - PatchTire::MultiDisk, [47](#)
 - PatchTire::Tire, [65](#)
- getRhoDot
 - PatchTire::MagicFormula, [31](#), [32](#)
 - PatchTire::MultiDisk, [47](#), [48](#)
 - PatchTire::Tire, [65](#)
- getVolume
 - PatchTire::MagicFormula, [32](#)
 - PatchTire::MultiDisk, [48](#)
 - PatchTire::Tire, [66](#)
- intersectAABBtree
 - RDF::MeshSurface, [36](#)
- intersectBBox
 - RDF::MeshSurface, [36](#)
- intersectEdgePlane
 - RDF::Triangle3D, [70](#)
 - RDF::TriangleRoad, [75](#)
- intersectPlane
 - RDF::Triangle3D, [71](#)
 - RDF::TriangleRoad, [76](#)

- intersectPointSegment
 - PatchTire::algorithms, 12
- intersectRay
 - RDF::Triangle3D, 71
 - RDF::TriangleRoad, 76
- intersectRayPlane
 - PatchTire::algorithms, 12
- LoadFile
 - RDF::MeshSurface, 37
- MagicFormula
 - PatchTire::MagicFormula, 27
- MeshSurface
 - RDF::MeshSurface, 36
- minmax_XY
 - PatchTire::algorithms, 13
- MultiDisk
 - PatchTire::MultiDisk, 40, 41
- PatchTire, 11
- PatchTire::Disk, 20
 - contactPlane, 21
 - contactTriangles, 22
 - Disk, 21
 - set, 22
 - setOriginXZ, 22
- PatchTire::ETRTO, 23
 - ETRTO, 23
 - print, 24
- PatchTire::MagicFormula, 24
 - evaluateContact, 27
 - fourPointsSampling, 27
 - getArea, 28
 - getEulerAngleX, 28
 - getEulerAngleY, 28
 - getEulerAngleZ, 28
 - getFriction, 29
 - getMFpoint, 29
 - getMFpointRF, 30
 - getNormal, 30
 - getRelativeCamber, 31
 - getRho, 31
 - getRhoDot, 31, 32
 - getVolume, 32
 - MagicFormula, 27
 - pointSampling, 33
 - print, 33
 - printETRTOGeometry, 33
 - setOrigin, 34
 - setReferenceFrame, 34
 - setRotationMatrix, 34
 - setTotalTransformationMatrix, 34
 - setup, 34
- PatchTire::MultiDisk, 37
 - getArea, 42
 - getDiskFriction, 42
 - getDiskMFpoint, 42
 - getDiskMFpointRF, 43
 - getDiskNormal, 43
 - getDiskOriginXYZ, 43
 - getDiskRho, 44
 - getDiskRhoDot, 44
 - getEulerAngleX, 44
 - getEulerAngleY, 44
 - getEulerAngleZ, 45
 - getFriction, 45
 - getMFpoint, 45
 - getMFpointRF, 46
 - getNormal, 46
 - getRelativeCamber, 47
 - getRho, 47
 - getRhoDot, 47, 48
 - getVolume, 48
 - MultiDisk, 40, 41
 - pointSampling, 50
 - print, 50
 - printETRTOGeometry, 50
 - setDiskOriginXZ, 51
 - setOrigin, 51
 - setReferenceFrame, 51
 - setRotationMatrix, 51
 - setTotalTransformationMatrix, 52
 - setup, 52
- PatchTire::ReferenceFrame, 52
 - getEulerAngleX, 54
 - getEulerAngleY, 54
 - getEulerAngleZ, 54
 - ReferenceFrame, 53
 - set, 54
 - setOrigin, 54
 - setRotationMatrix, 54
 - setTotalTransformationMatrix, 55
- PatchTire::SamplingGrid, 55
 - SamplingGrid, 56
 - set, 56
 - setSwitchNumber, 57
- PatchTire::Shadow, 57
 - Shadow, 57
 - update, 58
- PatchTire::Tire, 58
 - evaluateContact, 61
 - getArea, 61, 62
 - getEulerAngleX, 62
 - getEulerAngleY, 62
 - getEulerAngleZ, 62
 - getFriction, 62, 63
 - getMFpoint, 63
 - getMFpointRF, 63, 64
 - getNormal, 64
 - getRelativeCamber, 64
 - getRho, 65
 - getRhoDot, 65
 - getVolume, 66
 - pointSampling, 66
 - print, 67
 - printETRTOGeometry, 67

- setOrigin, [67](#)
 - setReferenceFrame, [67](#)
 - setRotationMatrix, [68](#)
 - setTotalTransformationMatrix, [68](#)
 - setup, [68](#)
 - Tire, [61](#)
- PatchTire::algorithms, [11](#)
 - intersectPointSegment, [12](#)
 - intersectRayPlane, [12](#)
 - minmax_XY, [13](#)
 - trapezoidArea, [13](#)
 - weightedMean, [14](#)
- pointSampling
 - PatchTire::MagicFormula, [33](#)
 - PatchTire::MultiDisk, [50](#)
 - PatchTire::Tire, [66](#)
- print
 - PatchTire::ETRTO, [24](#)
 - PatchTire::MagicFormula, [33](#)
 - PatchTire::MultiDisk, [50](#)
 - PatchTire::Tire, [67](#)
 - RDF::BBox2D, [20](#)
 - RDF::Triangle3D, [71](#)
 - RDF::TriangleRoad, [76](#)
- printData
 - RDF::MeshSurface, [37](#)
- printETRTOGeometry
 - PatchTire::MagicFormula, [33](#)
 - PatchTire::MultiDisk, [50](#)
 - PatchTire::Tire, [67](#)
- RDF::BBox2D, [19](#)
 - BBox2D, [20](#)
 - print, [20](#)
 - updateBBox2D, [20](#)
- RDF::MeshSurface, [35](#)
 - intersectAABBtree, [36](#)
 - intersectBBox, [36](#)
 - LoadFile, [37](#)
 - MeshSurface, [36](#)
 - printData, [37](#)
 - set, [37](#)
- RDF::Triangle3D, [69](#)
 - intersectEdgePlane, [70](#)
 - intersectPlane, [71](#)
 - intersectRay, [71](#)
 - print, [71](#)
 - setVertices, [73](#)
 - Triangle3D, [70](#)
- RDF::TriangleRoad, [73](#)
 - intersectEdgePlane, [75](#)
 - intersectPlane, [76](#)
 - intersectRay, [76](#)
 - print, [76](#)
 - setFriction, [77](#)
 - setVertices, [77](#)
 - TriangleRoad, [75](#)
- RDF::algorithms, [15](#)
 - firstToken, [15](#)
 - getElement, [16](#)
 - split, [16](#)
 - tail, [16](#)
- RDF, [14](#)
- ReferenceFrame
 - PatchTire::ReferenceFrame, [53](#)
- SamplingGrid
 - PatchTire::SamplingGrid, [56](#)
- set
 - PatchTire::Disk, [22](#)
 - PatchTire::ReferenceFrame, [54](#)
 - PatchTire::SamplingGrid, [56](#)
 - RDF::MeshSurface, [37](#)
- setDiskOriginXZ
 - PatchTire::MultiDisk, [51](#)
- setFriction
 - RDF::TriangleRoad, [77](#)
- setOrigin
 - PatchTire::MagicFormula, [34](#)
 - PatchTire::MultiDisk, [51](#)
 - PatchTire::ReferenceFrame, [54](#)
 - PatchTire::Tire, [67](#)
- setOriginXZ
 - PatchTire::Disk, [22](#)
- setReferenceFrame
 - PatchTire::MagicFormula, [34](#)
 - PatchTire::MultiDisk, [51](#)
 - PatchTire::Tire, [67](#)
- setRotationMatrix
 - PatchTire::MagicFormula, [34](#)
 - PatchTire::MultiDisk, [51](#)
 - PatchTire::ReferenceFrame, [54](#)
 - PatchTire::Tire, [68](#)
- setSwitchNumber
 - PatchTire::SamplingGrid, [57](#)
- setTotalTransformationMatrix
 - PatchTire::MagicFormula, [34](#)
 - PatchTire::MultiDisk, [52](#)
 - PatchTire::ReferenceFrame, [55](#)
 - PatchTire::Tire, [68](#)
- setVertices
 - RDF::Triangle3D, [73](#)
 - RDF::TriangleRoad, [77](#)
- setup
 - PatchTire::MagicFormula, [34](#)
 - PatchTire::MultiDisk, [52](#)
 - PatchTire::Tire, [68](#)
- Shadow
 - PatchTire::Shadow, [57](#)
- split
 - RDF::algorithms, [16](#)
- tail
 - RDF::algorithms, [16](#)
- TicToc, [58](#)
- Tire
 - PatchTire::Tire, [61](#)
- TireGround, [16](#)

trapezoidArea
 PatchTire::algorithms, [13](#)
Triangle3D
 RDF::Triangle3D, [70](#)
TriangleRoad
 RDF::TriangleRoad, [75](#)

update
 PatchTire::Shadow, [58](#)
updateBBox2D
 RDF::BBox2D, [20](#)

weightedMean
 PatchTire::algorithms, [14](#)