Intersezione trà alberi di tipo AABB

Intersezione trà alberi di tipo AABB

Volendo intersecare due semplici AABB, quali:

$$A = [\texttt{A.minX}, \texttt{A.maxX}, \texttt{A.minY}, \texttt{A.maxY}]$$
$$B = [\texttt{B.minX}, \texttt{B.maxX}, \texttt{B.minY}, \texttt{B.maxY}]$$

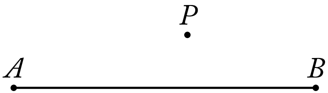
verrà usata la seguente funzione:

```
function intersect(A,B) {
    return (A.minX <= B.maxX && A.maxX >= B.minX) &&
    (A.minY <= B.maxY && A.maxY >= B.minY)
}
```

Intersezione trà entità geometriche

Intersezione segmento-punto

Dato un punto $P = (x_p, y_p)$ e un segmento definito dai punti $A = (x_A, y_B)$ e $B = (x_B, y_B)$.

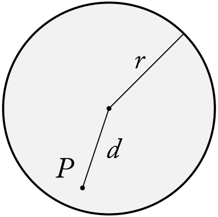


Per determinare se il punto P è interno al segmento gli *step* sono:

1. creazione di un vettore $\vec{AB}$ e di un vettore $\vec{AP}$;
2. calcolo del prodotto vettoriale $\vec{AB} \times \vec{PA}$, se il modulo del vettore risultante è nullo allora il punto P appartiene al segmento considerato;
3. calcolo del prodotto scalare tra $\vec{AB}$ e $\vec{AP}$, se è nullo allora $P \equiv A$, se è pari al modulo di $\vec{AB}$ allora il $P \equiv B$, se è compreso tra 0 il modulo di $\vec{AB}$, allora il punto P giace all'interno del segmento considerato.

Intersezione punto-cerchio

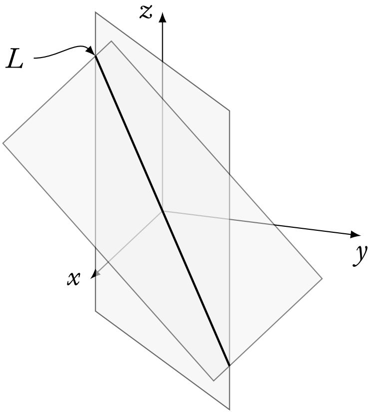
Data una circonferenza con centro $C = (x_c, y_c)$ e raggio r, il problema consiste nel trovare se un generico punto $P = (x_p, y_p)$ è all'interno, all'esterno o corrispondente alla circonferenza.



La soluzione al problema è semplice: la distanza tra il centro della circonferenza C e il punto P è data dal teorema di Pitagora, ovvero:

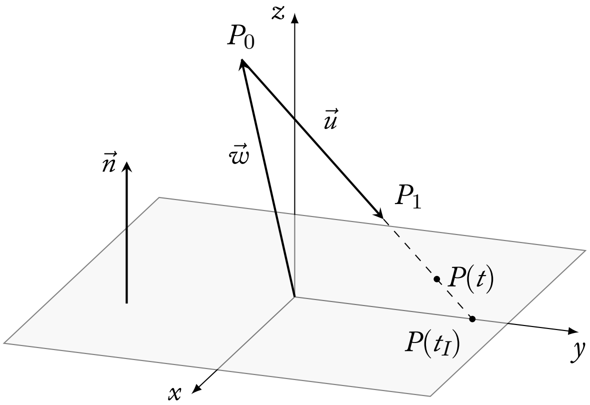
$$d = \sqrt{(x_p - x_c)^2 + (y_p - y_c)^2}$$

Intersezione piano-piano



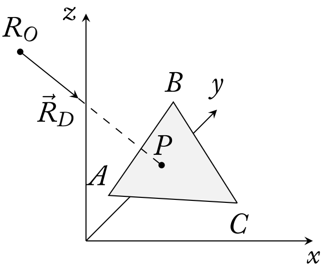
$$L(s) = \frac{(d_2\vec{n}_1 - d_1\vec{n}_2) \times \vec{u}}{|\vec{u}|^2} + s\vec{u}$$

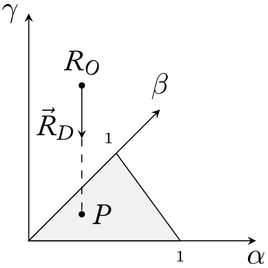
Intersezione piano-segmento e piano-raggio



$$\vec{n} \cdot (\vec{w} + t\vec{u}) = 0 \qquad t_I = -\frac{\vec{n} \cdot \vec{w}}{\vec{n} \cdot \vec{u}}$$

Intersezione raggio-triangolo



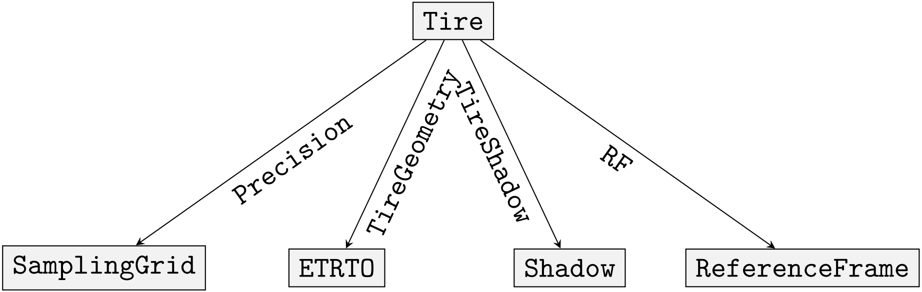


$$R_O + t\vec{R}_D = A + u(B - A) + v(C - A)$$

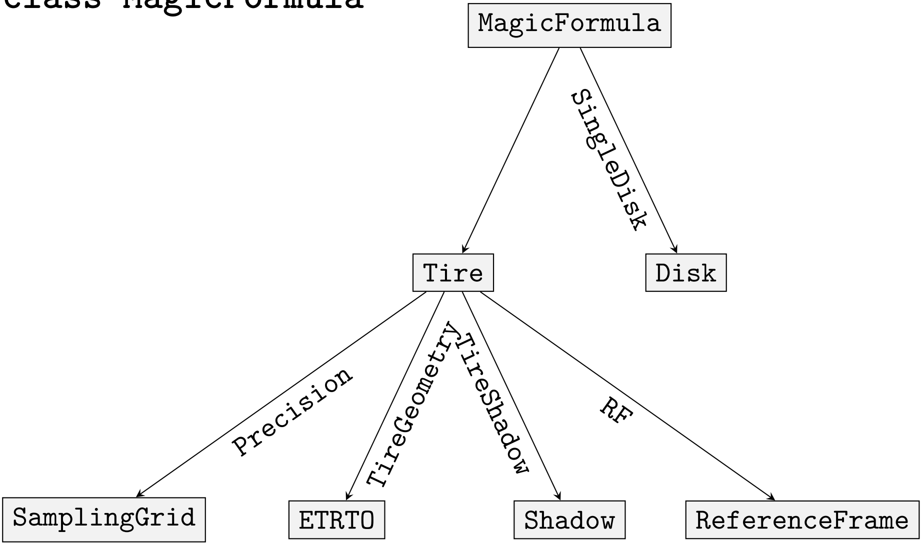
$$\begin{bmatrix} t \\ u \\ v \end{bmatrix} = \frac{1}{(D \times E_2) \cdot E_1} \begin{bmatrix} (T \times E_1) \cdot E_2 \\ (D \times E_2) \cdot T \\ (T \times E_1) \cdot D \end{bmatrix} = \frac{1}{P \cdot E_1} \begin{bmatrix} Q \cdot E_2 \\ P \cdot T \\ Q \cdot D \end{bmatrix}$$

La libreria `TireGround`

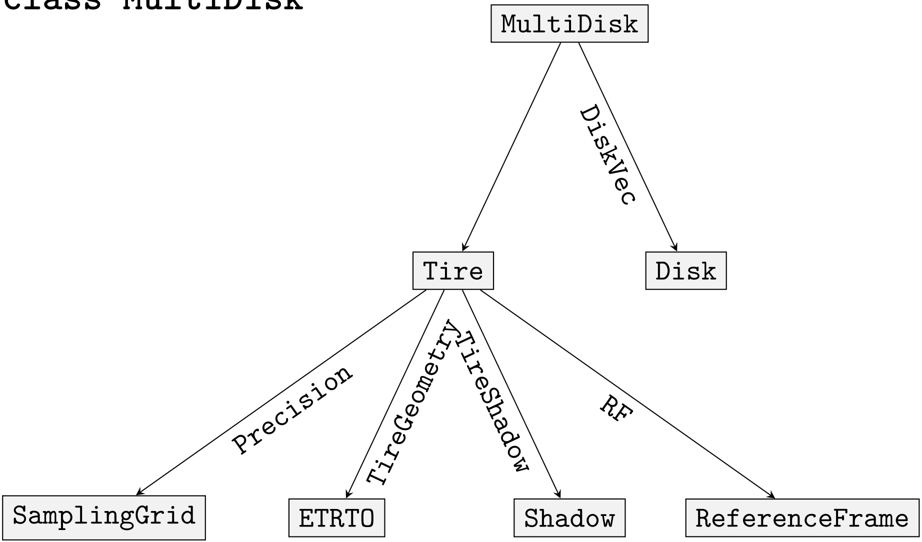
class Tire


Tire
Precision
TireGeometry
TireShadow
RF
SamplingGrid
ETRTO
Shadow
ReferenceFrame

class MagicFormula



class MultiDisk



Istanziamento della *mesh*

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

Istanziamento dello pneumatico

```
TireGround::Tire* TireSD = new TireGround::MagicFormula(
    SectionWidth, // [m]
    AspectRatio,  // [%]
    RimDiameter,  // [in]
    SwitchNumber  // Max triangles in the shadow
);
```

```
TireGround::Tire* TireMD = new TireGround::MultiDisk(
    SectionWidth, // [m]
    AspectRatio,  // [%]
    RimDiameter,  // [in]
    RadiusVec,    // Disks radius vector [m]
    PointsNumber, // Sampling points for each disk
    SwitchNumber  // Max triangles in the shadow
);
```

Posizionamento dello pneumatico nella libreria

```
bool Out = SampleTire->setup(
    Road, // Superficie stradale
    TM    // Matrice di trasformazione 4x4
);
```

```
bool Out = SampleTire->setup(
    Normal,   // Vettore normale al piano
    Point,    // Punto appartenente al piano
    Friction, // Coefficiente di attrito nel piano
    TM        // Matrice di trasformazione 4x4
);
```

Estrazione dei risultati

```
TireGround::vec3 N;
TireGround::vec3 P;
TireGround::real_type Friction;
TireGround::real_type Rho;
TireGround::real_type RhoDot;
TireGround::real_type RelativeCamber;
TireGround::real_type Area;
TireGround::real_type Volume;

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