matfyz

Computer Graphics Charles University

Ray-Tracing in GrCis

© 2016-2020 Josef Pelikán
CGG MFF UK Praha

pepca@cgg.mff.cuni.cz
https://cgg.mff.cuni.cz/~pepca/

Ray-tracing-related projects

Best for work & demo

- **048rtmontecarlo-script**
  - switches for super-sampling, shadows, reflections, refractions, multi-threading, CS-script scene definition

Animation

- 046cameranim
  - camera animation (going round the scene)
- **062animation-script**
  - more general project, able to animate any scene part

Ray-tracing application

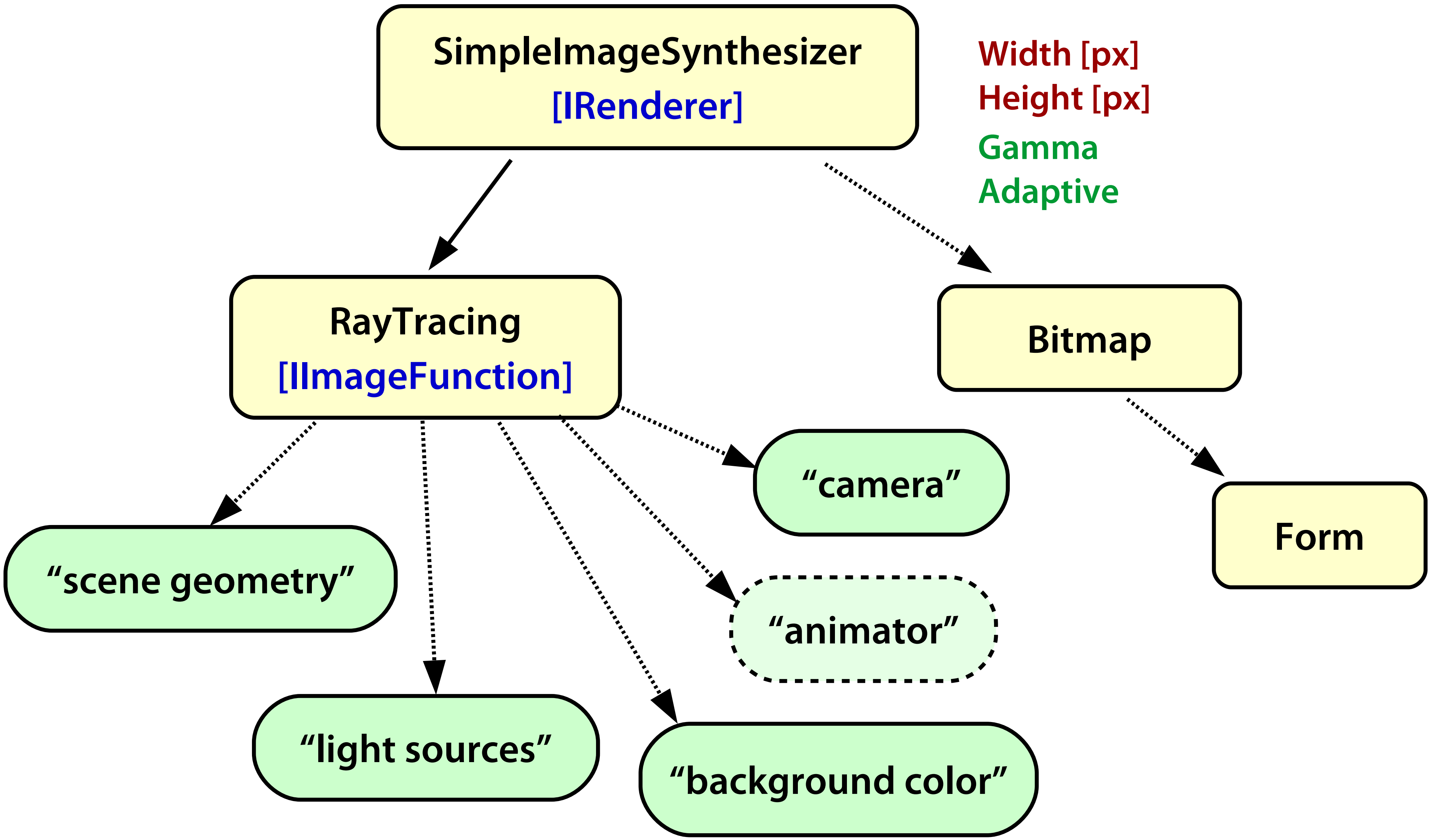
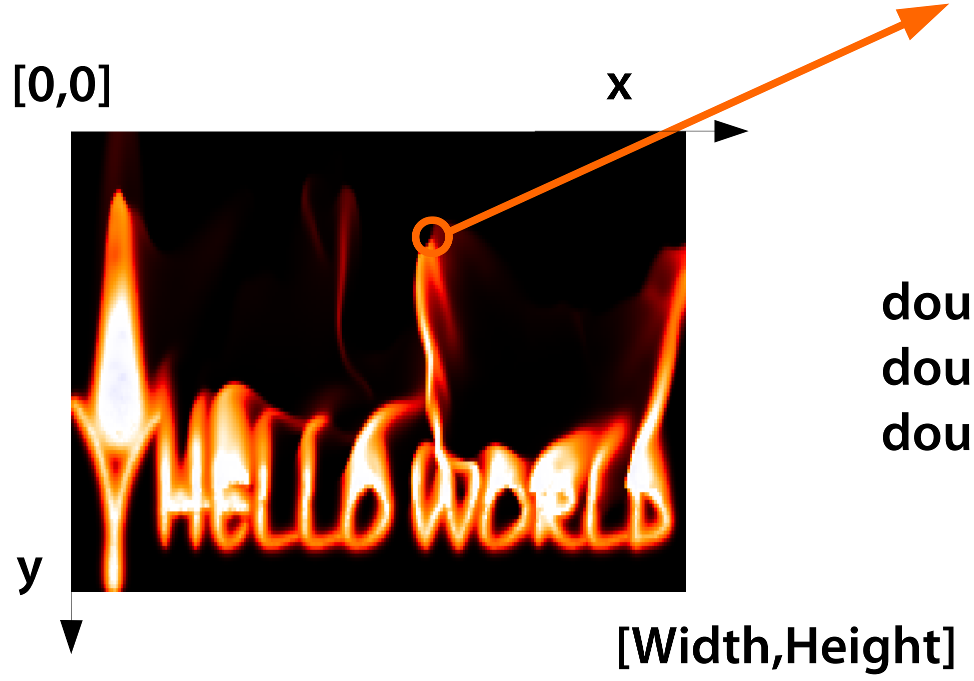


Image Function [IImageFunction]

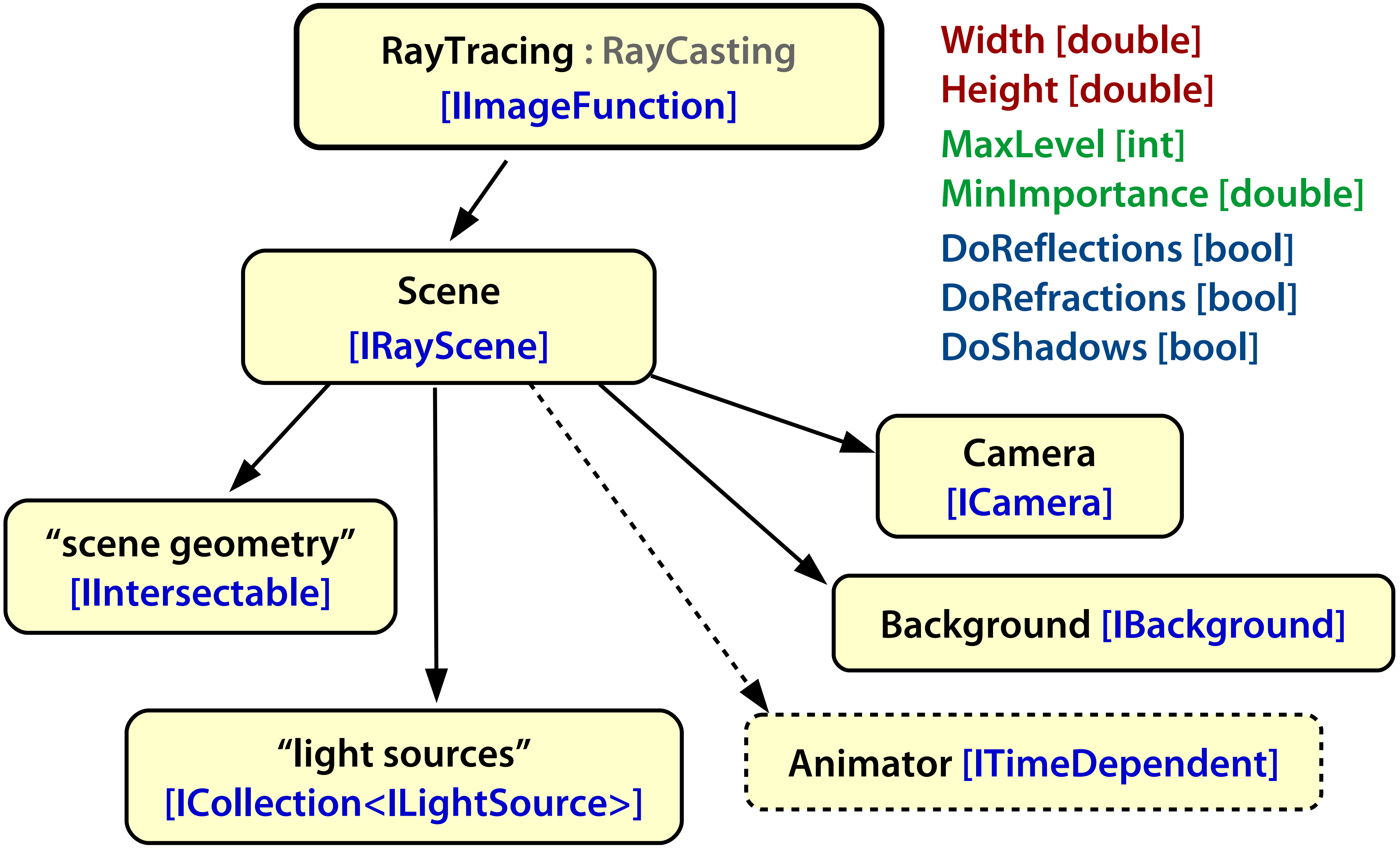
[interface IImageFunction]

double Width
double Height
long GetSample (double x, double y, double[] color)



double[] color ..
double[3] // RGB
double[len] // spectral color

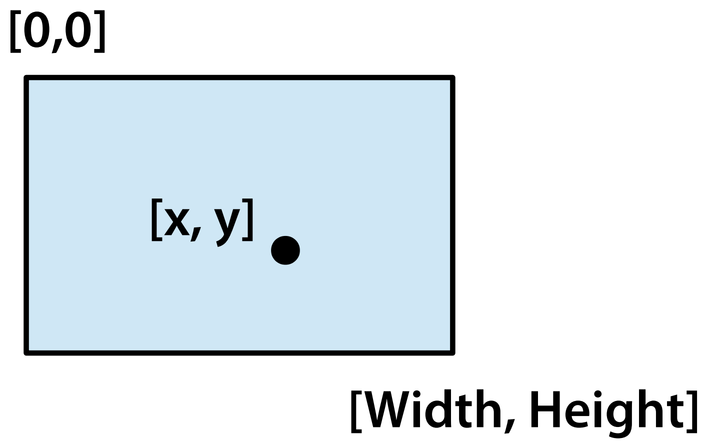
RayCasting, RayTracing



Width [double]
Height [double]
MaxLevel [int]
MinImportance [double]
DoReflections [bool]
DoRefractions [bool]
DoShadows [bool]

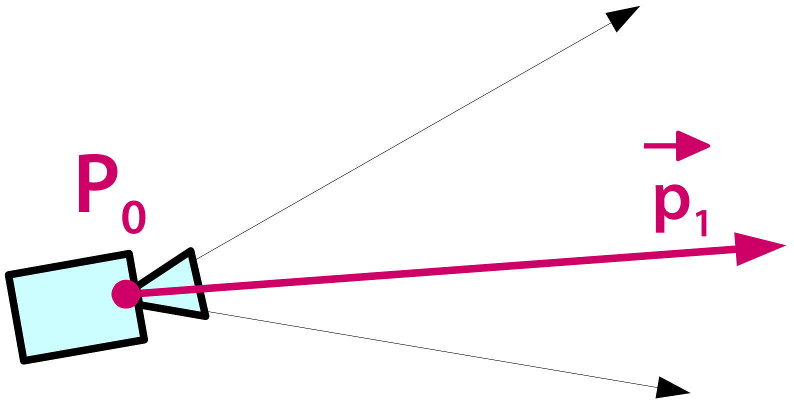
Camera [ICamera]

[interface ICamera]



double AspectRatio
double Width
double Height

```
bool GetRay ( double x, double y,
              out Vector3D p0, out Vector3D p1 )
```



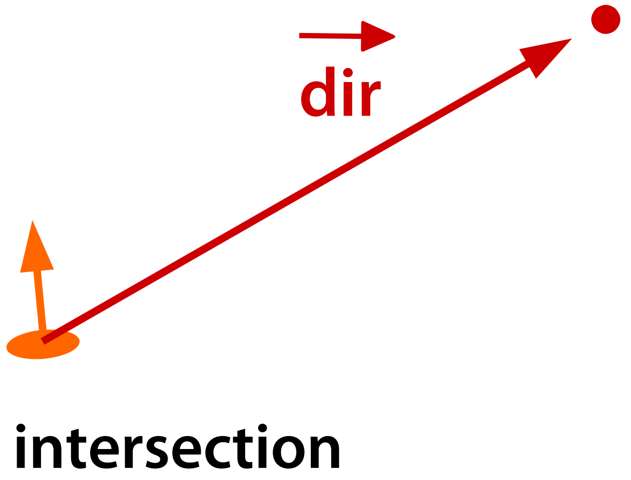
Ray: $P_0 + t \cdot \vec{p}_1$

$0 \le t$

Light Source [ILightSource]

[interface ILightSource]

```
double[] GetIntensity ( Intersection intersection,
                        out Vector3D dir )
```

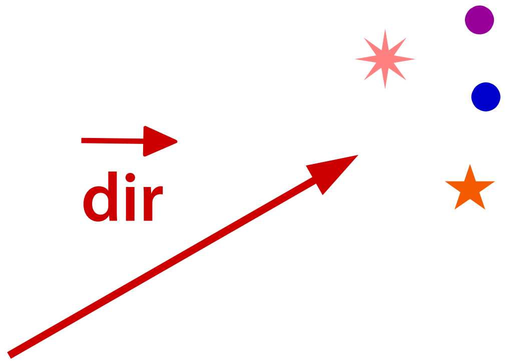


return:	color (intensity)
dir:	direction toward the light, zero for omnidirectional

Background [IBackground]

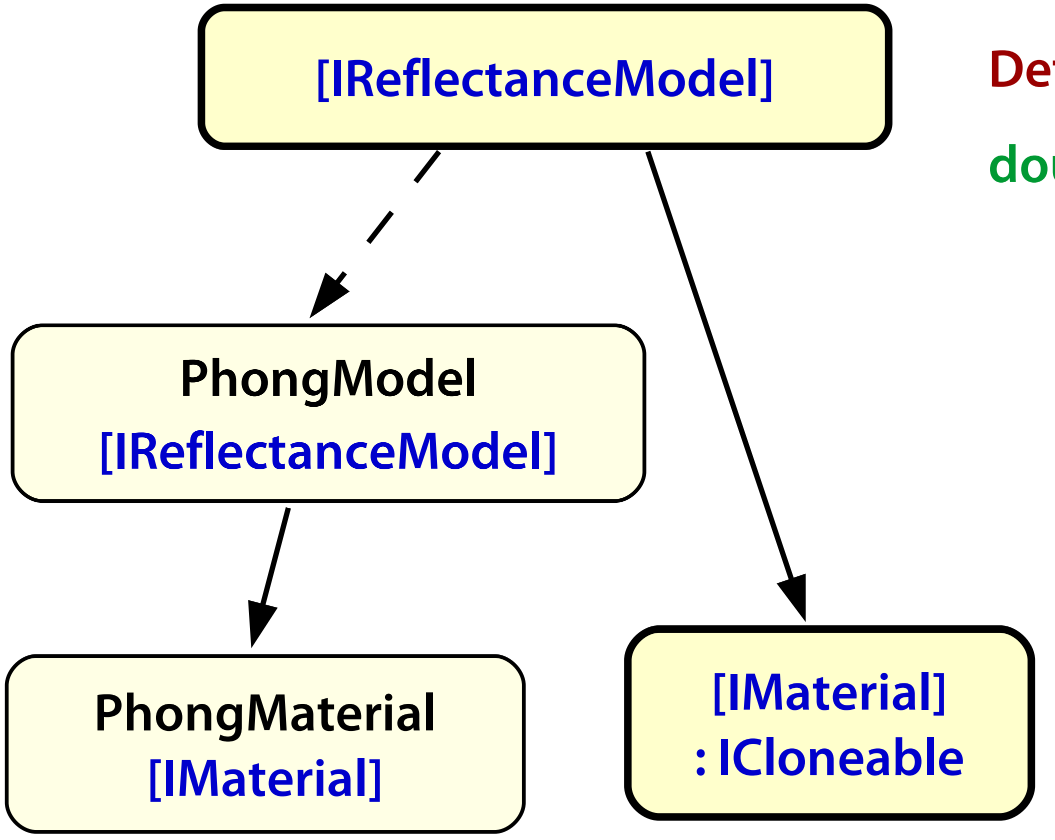
[interface IBackground]

long GetColor (Vector3d dir, double[] color)



dir:	direction to the infinity
color:	observed (background) color

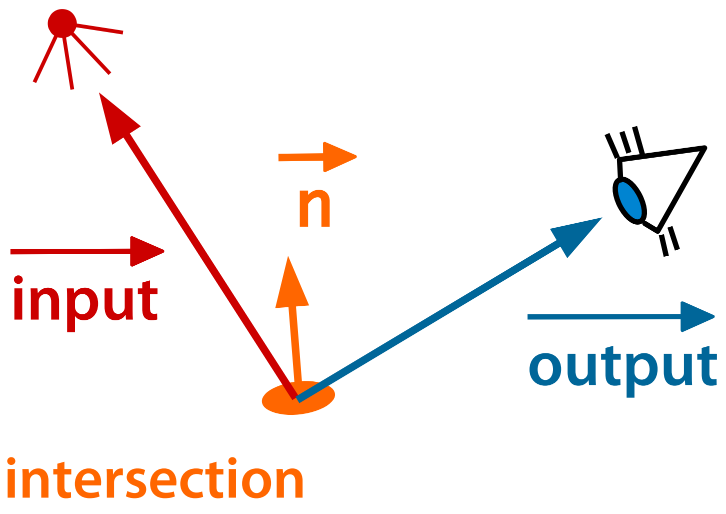
IReflectanceModel, IMaterial



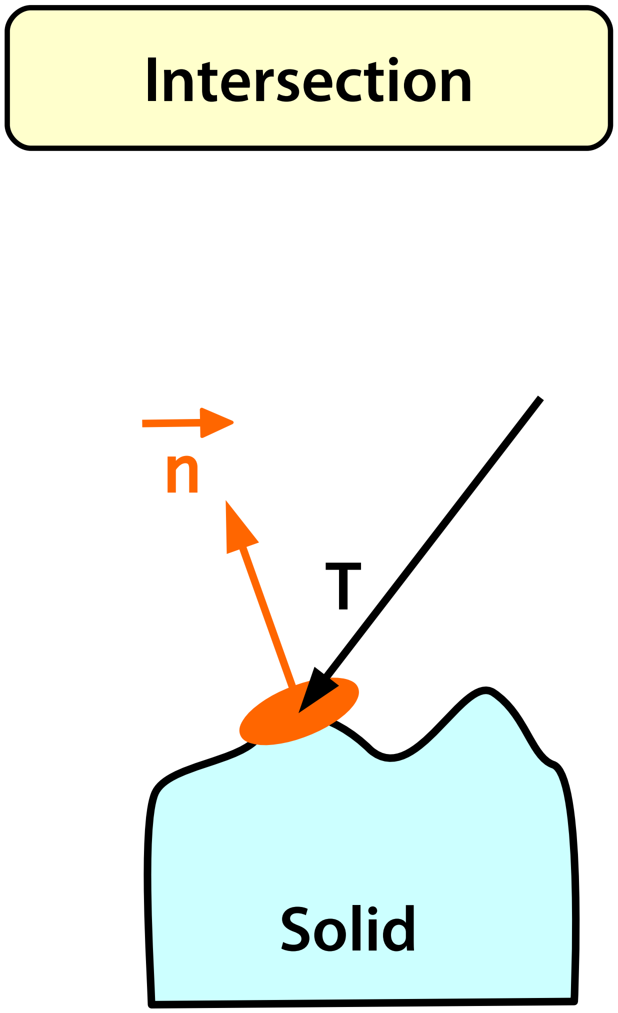
double[] Color
double Kt
double n

DefaultMaterial [IMaterial]

```
double [] ColorReflection (
    Intersection intersection,
    Vector3d input,
    Vector3d output,
    ReflectionComponent comp )
```


$\vec{n}$
$\overrightarrow{\text{input}}$
$\overrightarrow{\text{output}}$
intersection

Intersection



Enter [bool]
Front [bool]
T [double]
Solid [ISolid]
SolidData [object]

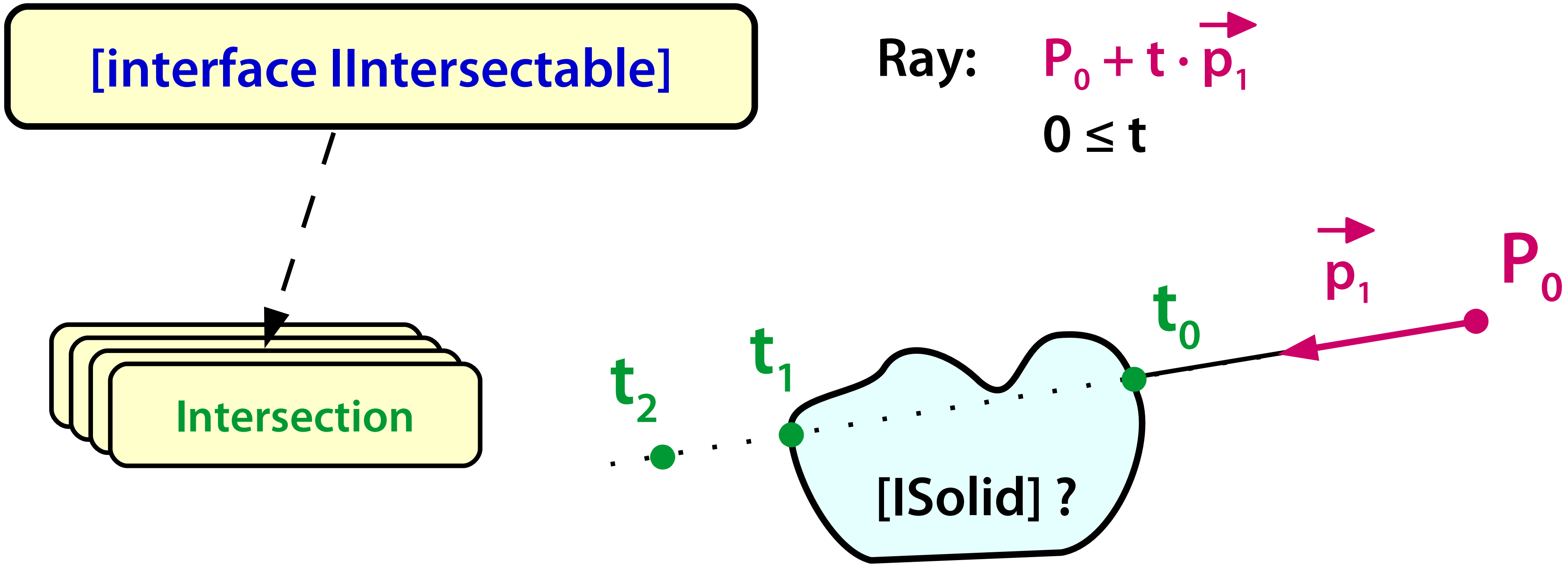
… mandatory

Normal [Vector3d]
CoordWorld [Vector3d]
CoordObject [Vector3d]
CoordLocal [Vector3d]
TextureCoord [Vector2d]
LocalToWorld [Matrix4d]
WorldToLocal [Matrix4d]
LocalToObject [Matrix4d]
SurfaceColor [double[]]
ReflectanceModel [IReflectanceModel]
Material [IMaterial]
Textures [List<ITexture>]

NormalLocal [Vector3d]
TangentU [Vector3d]
TangentV [Vector3d]

Complete();

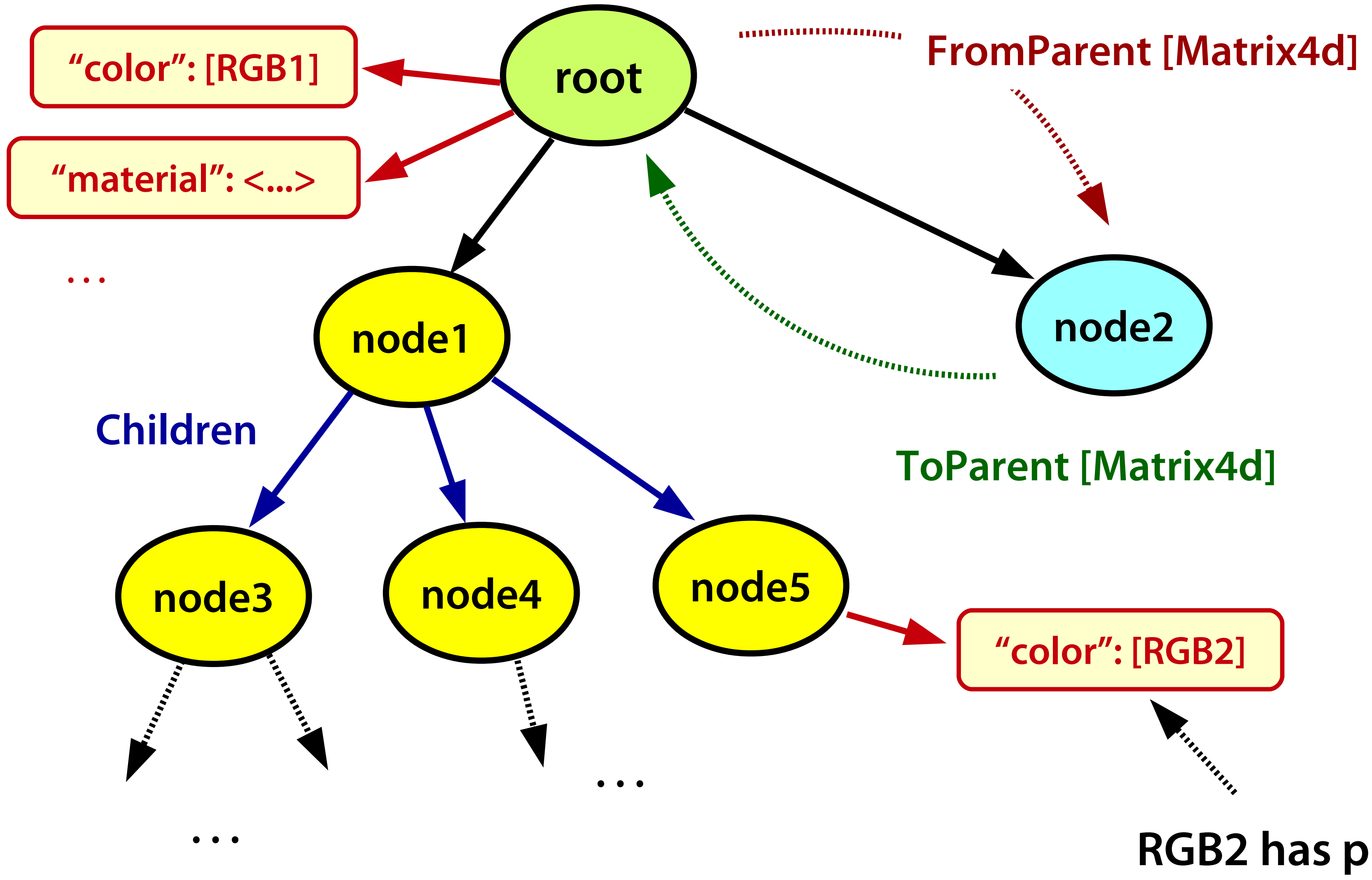
Intersectable Object [IIntersectable]



LinkedList<Intersection> Intersect (Vector3d p0, Vector3d p1)

void CompleteIntersection (Intersection inter)

Scene Hierarchy



Scene Node [ISceneNode]

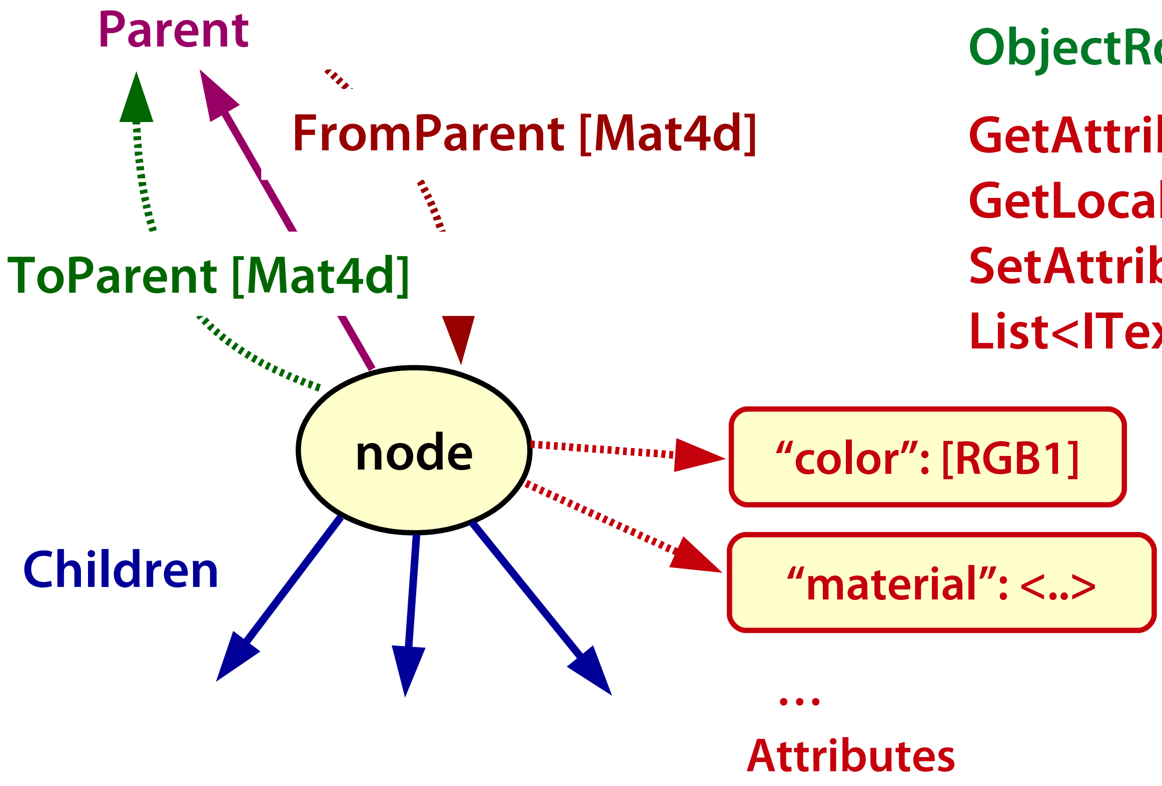
[interface ISceneNode]
: IIntersectable

Parent [ISceneNode]
Children [IsceneNodes[]]

ToParent [Matrix4d]
FromParent [Matrix3d]
ToWorld, ToObject [Matrix4d]

ObjectRoot [bool]

GetAttribute (name)
GetLocalAttribute (name)
SetAttribute (name, value)
List<ITexture> GetTextures ()



Solid [ISolid]

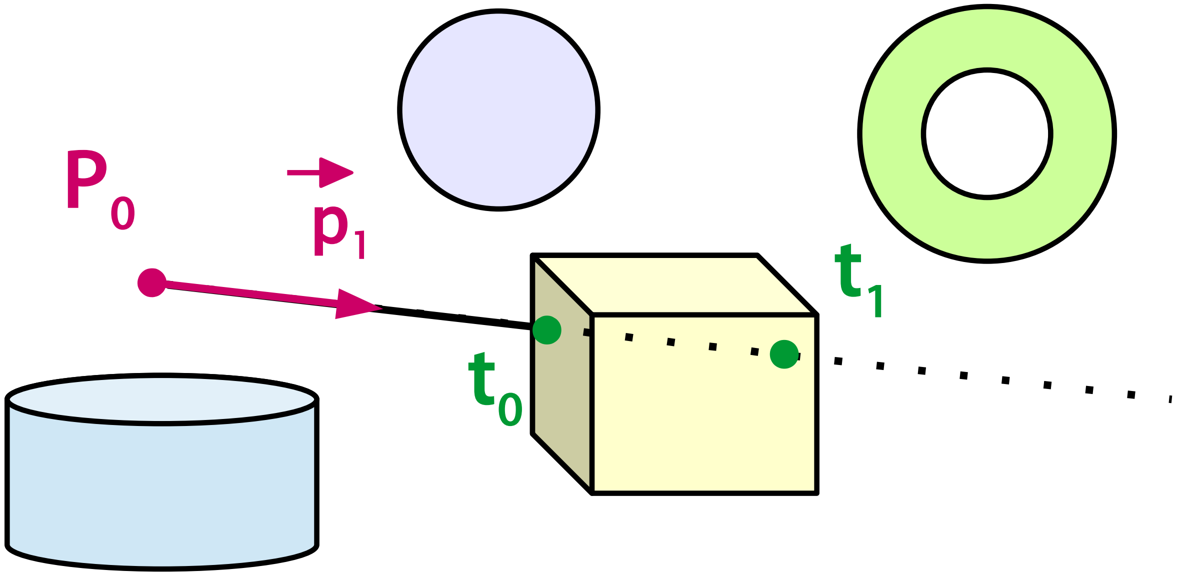
[interface ISolid]
: ISceneNode

Ray: $P_0 + t \cdot \vec{p}_1$
$0 \leq t$

LinkedList<Intersection> Intersect (Vector3d p0, Vector3d p1)

void CompleteIntersection (Intersection inter)

- Sphere
- Cube
- Plane
- Cylinder
- Torus…



Texture [ITexture]

[interface ITexture]

long Apply (Intersection inter)

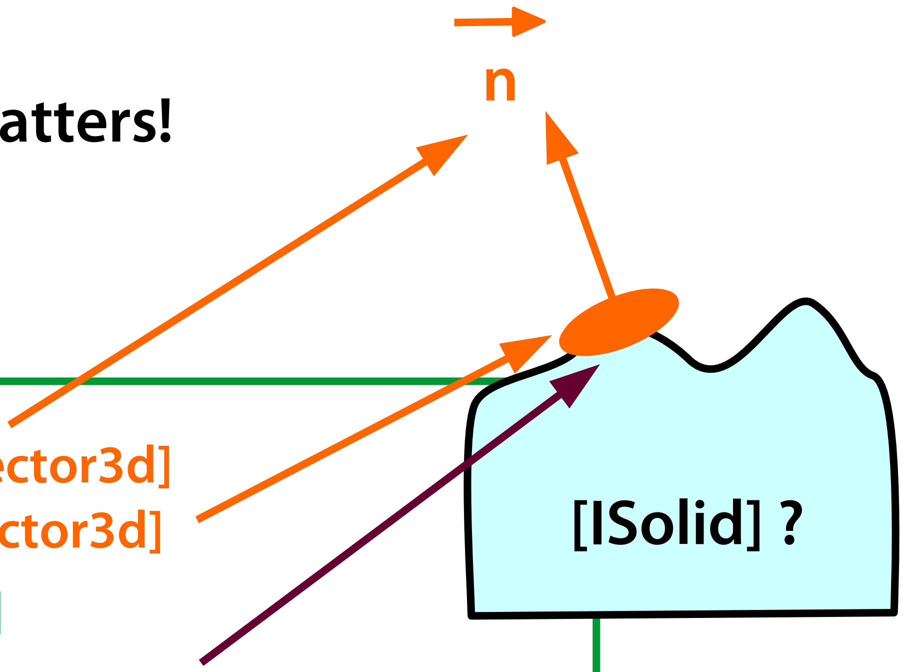
Texture order matters!

Intersection

Normal, NormalLocal [Vector3d]
TangentU, TangentV [Vector3d]

TextureCoord [Vector2d]

SurfaceColor [double[]]
ReflectanceModel [IReflectanceModel]
Material [IMaterial]



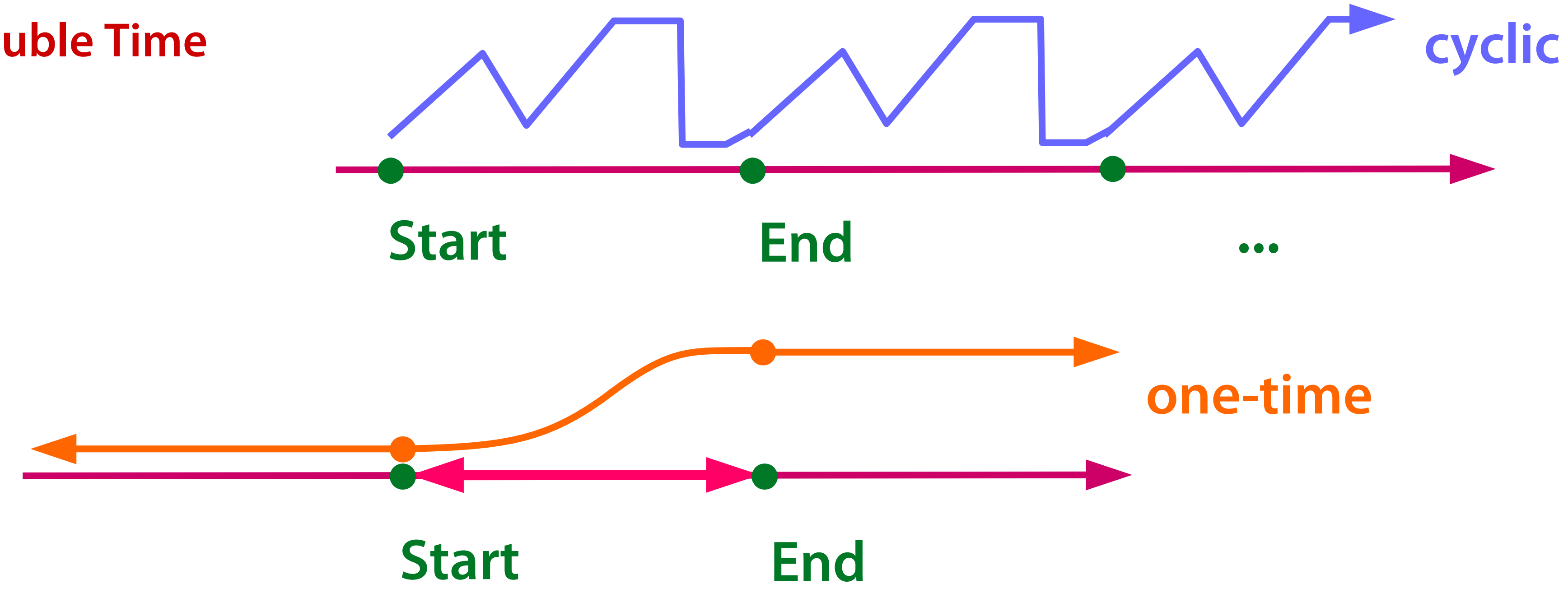
Animation [ITimeDependent]

[interface ITimeDependent]
: ICloneable

"Clone-on-write"

- for multi-threaded rendering
- cloning a copy for each thread

double Start
double End
double Time



Independent Stratified Sampling

Multi-dimensional open sampling: **[0,1]**^{D}

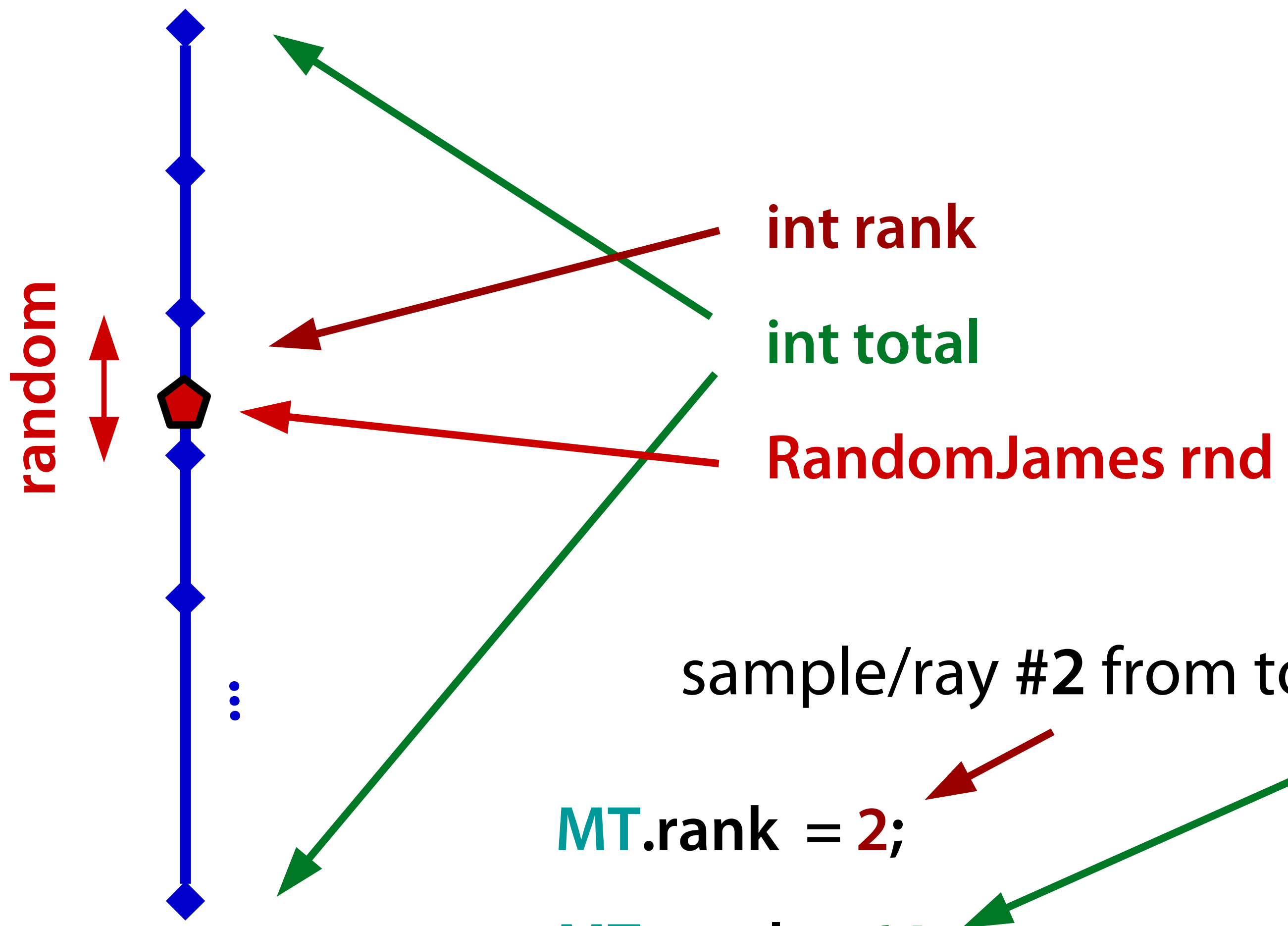
- D is not known in advance
- any internal component of a ray-tracer might be sampled (integral averaging)

Hidden sampling mechanism

- any component can use additional global values stored in the static **class MT**:

[ThreadStatic]	… TLS (automatic data instance for each thread)
int rank	… order of the current sample (in the current pixel)
int total	… total number of samples in the current pixel
RandomJames rnd	… random number generator

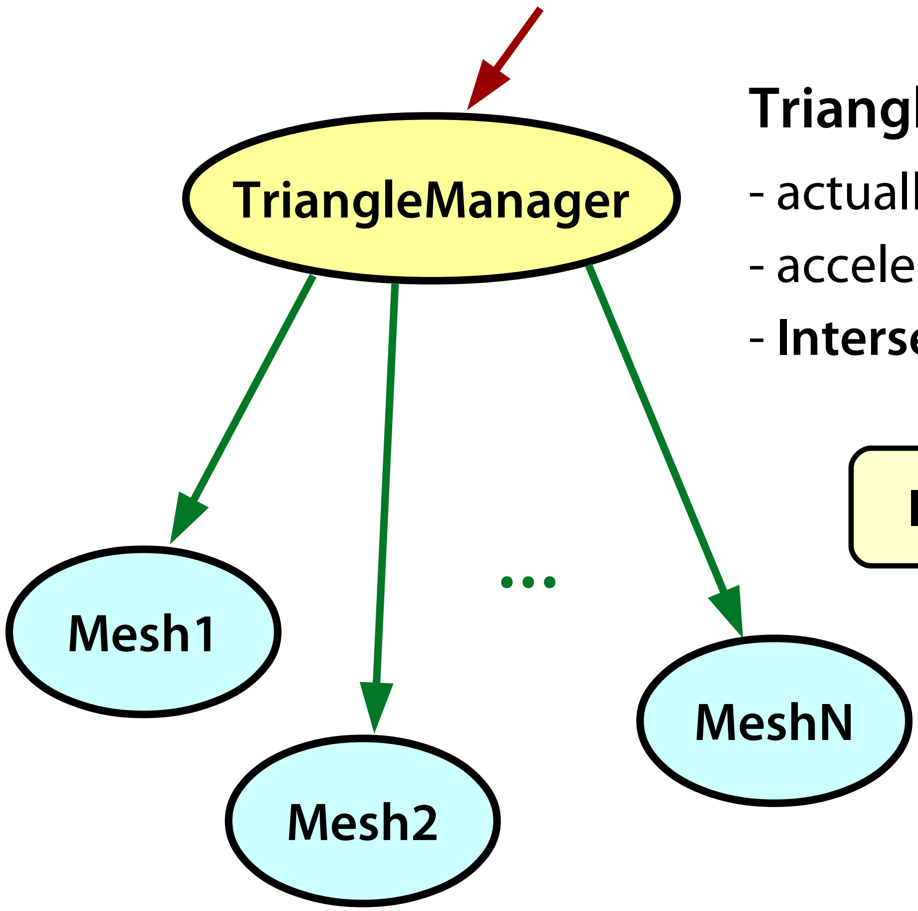
Independent Stratified Sampling



sample/ray **#2** from total **64** samples

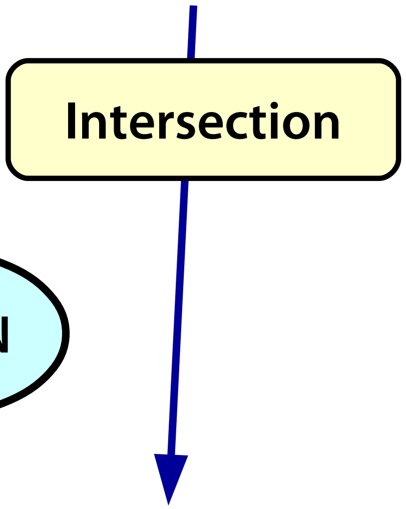
```
MT.rank  = 2;
MT.total  = 64;
```

Accelerating ray – triangle-mesh intersection



TriangleManager : AnimatedCSGInnerNode : ISolid

- actually **computing intersections**
- accelerated data structure (e.g. **BVH**)
- **Intersect()** call is **not propagated** to its Children



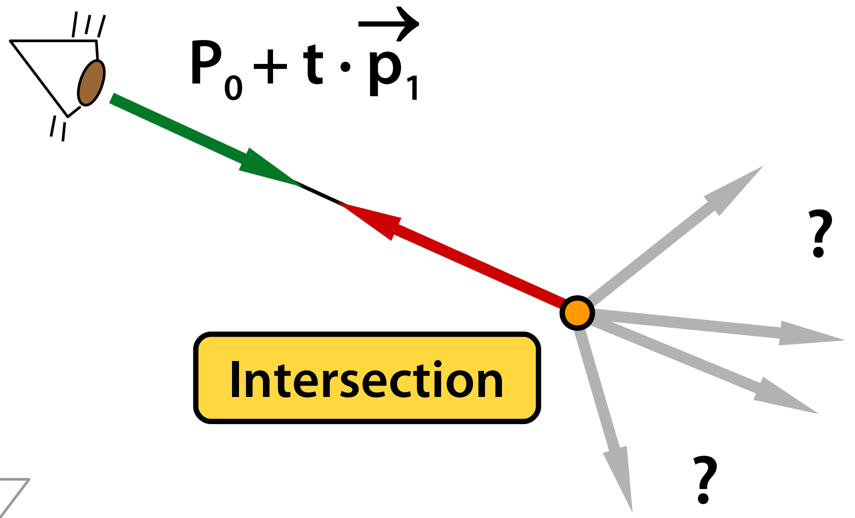
```
Solid [ISolid]
CoordLocal [Vector3d]
SolidData =
 {
  int TriangleId;
  Vector2d Barycentric;
 }
```

TriangleMesh : AnimatedCSGInnerNode : ISolid

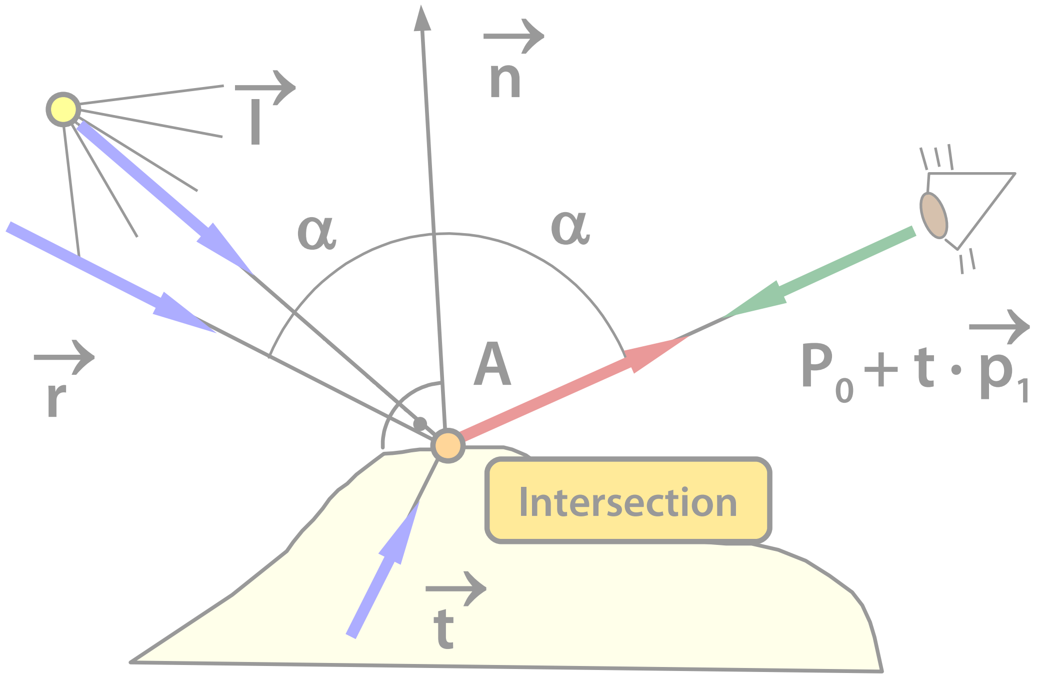
- only **completing intersections** found by Manager
- the same coordinate space as its Manager

Light Composition Tweak

Tweak – **RecursionFunction()** callback



Classical Ray-tracing (Whitted)



DirectContribution (additive color)

+

RayContribution-s

Vector3d direction
double[] coefficient
double importance

References

GitHub repository
https://github.com/pepcape/grcis

Subversion repository
svn://cgg.mff.cuni.cz/grcis/trunk

Ray-tracing in GrCis
https://cgg.mff.cuni.cz/~pepca/grcis/rt.php

GrCis library
https://cgg.mff.cuni.cz/~pepca/grcis/

Image gallery
https://cgg.mff.cuni.cz/~pepca/gr/grcis/