Institut de Recherche
en Informatique de Toulouse
CNRS - INP - UT3 - UT1 - UT2J

PostFX Shadow Layers

Projet de Pré-maturation

financée par la région Occitanie

David VANDERHAEGHE : vdh@irit.fr
François DESRICHARD
Gauthier BOUYJOU

Le projet PostFX Shadow Layers

Objectif

Implementation shadow layers dans un logiciel professionnel

- Démonstration de faisabilité pour aller vers un transfert industriel

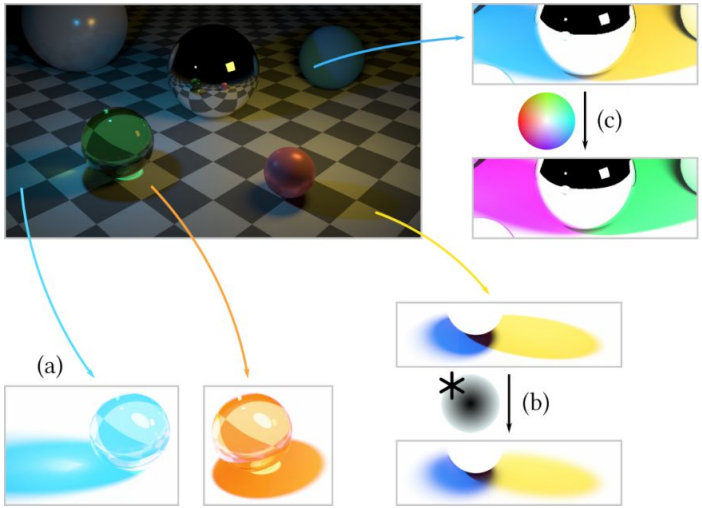
Contexte

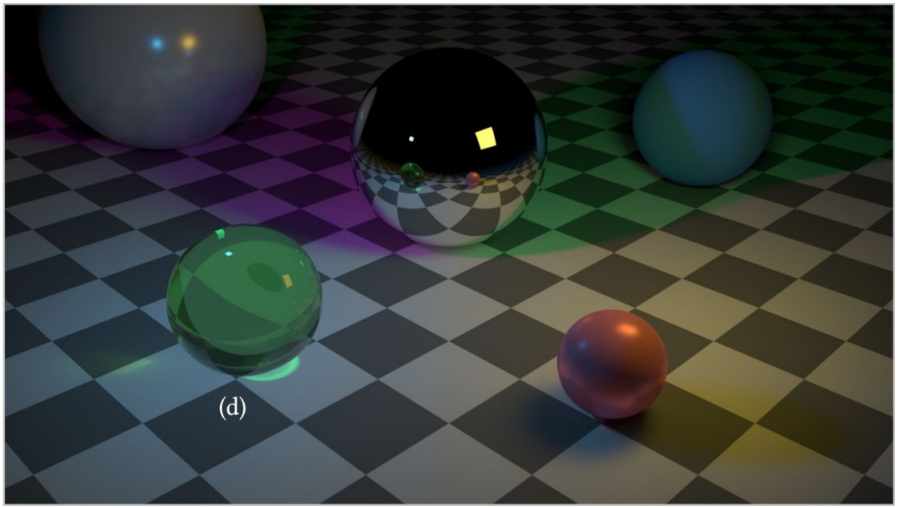
Projet de pré-maturation

Shadow Layers

Domaine d’application :

- Cinéma, VFX (Visual Effects), image de synthèse
- Édition / contrôle artistique
- Post-processing, espace image





Global Illumination Shadow Layers (GISL)
Computer Graphics Forum, Wiley, 2019, 38(4).
F. Desrichard, D. Vanderhaeghe, M. Paulin, 2019

Environnement logiciel

ARNOLD

Arnold Renderer

Moteur de rendu *path tracing* hors-ligne (même méthode de rendu que GISL)

Compatible avec de nombreux logiciels de modélisation 3D, notamment :

Maya, Cinema 4D, Houdini, Katana, 3ds Max, Softimage





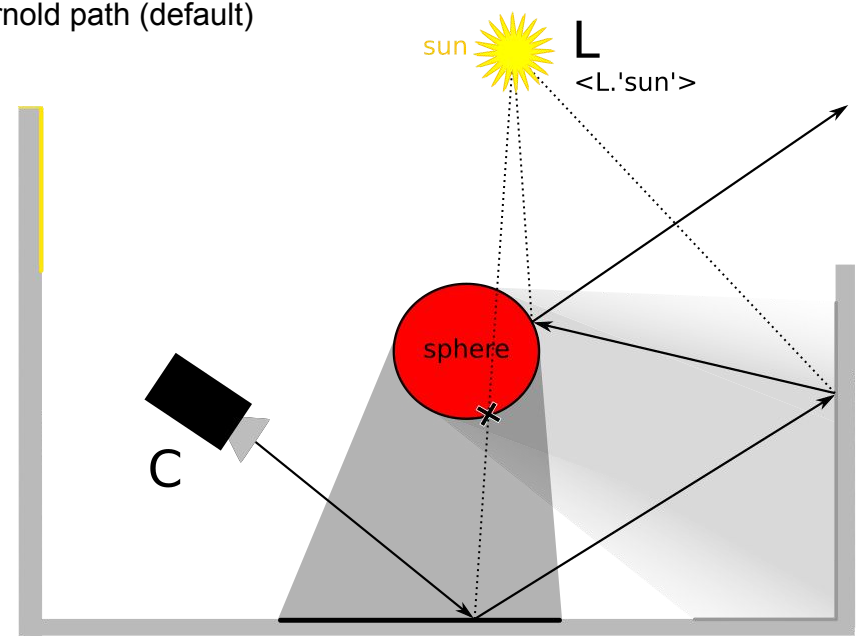

Objectifs de mission

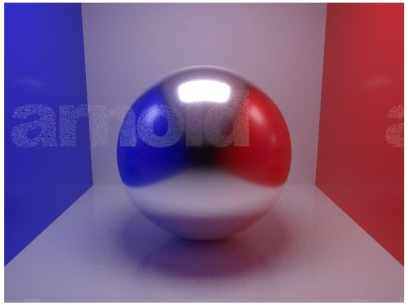
1. Adaptation de l’algorithme de rendu
   - Calcul d’un calque d’ombre pour toute la scène
   - Calcul calques séparés par objet
   - Split direct/indirect, split lights, ombre propre

2. Contrôle depuis l’interface utilisateur
   - Selection objet dans le viewport
   - Activation des différents calques

Adaptation de l’algorithme de rendu

Arnold path (default)



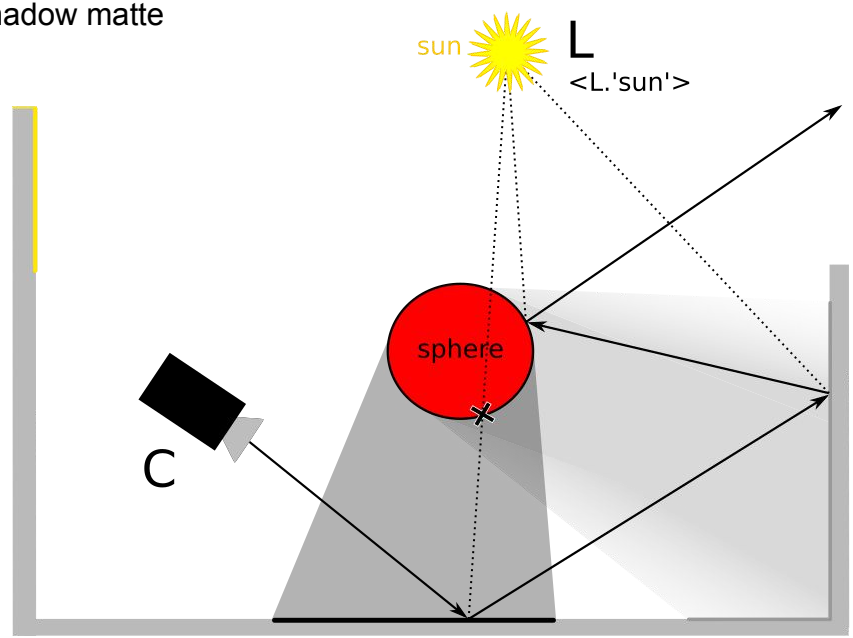


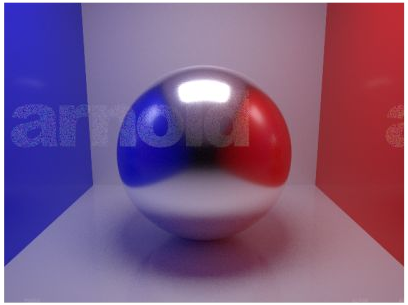
Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés

default path (Arnold)
shadow ray (Arnold)

Adaptation de l’algorithme de rendu

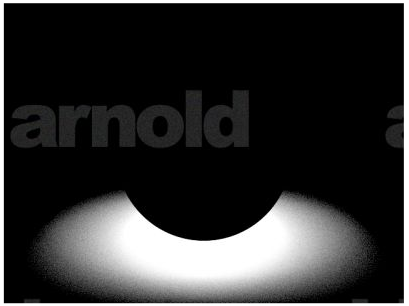
Shadow matte





shadow_matte

(Arnold integrated shader)

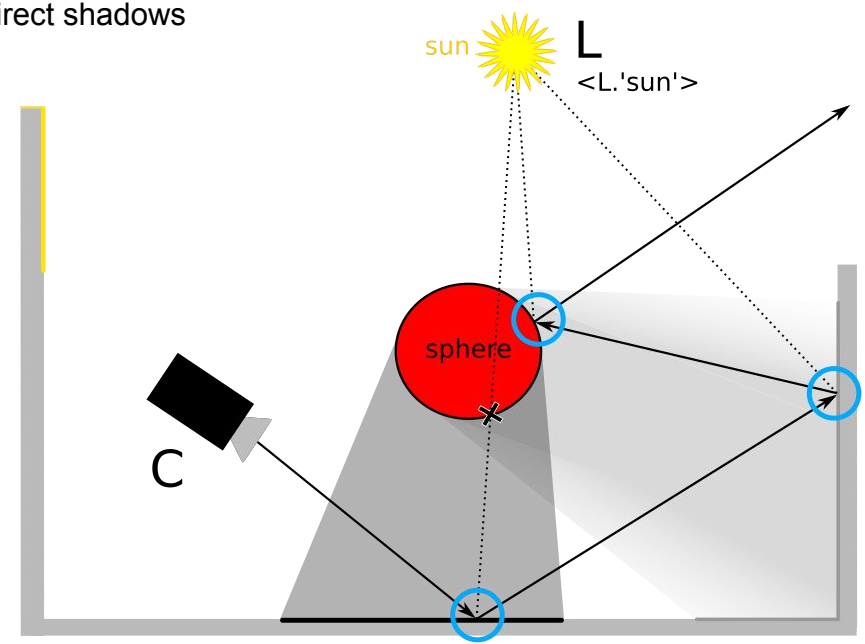


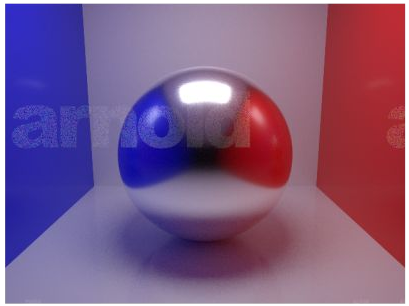
Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)
→ default path (Arnold)
⋯→ shadow ray (Arnold)

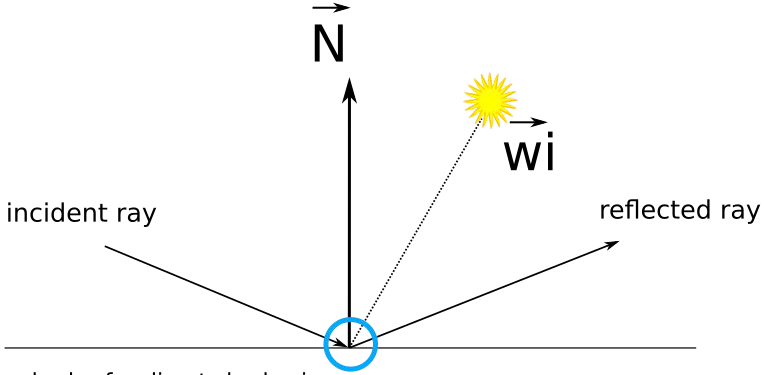
Le shader shadow matte donne qu'une information de visibilité avec les sources de lumières directes

Adaptation de l'algorithme de rendu

Direct shadows







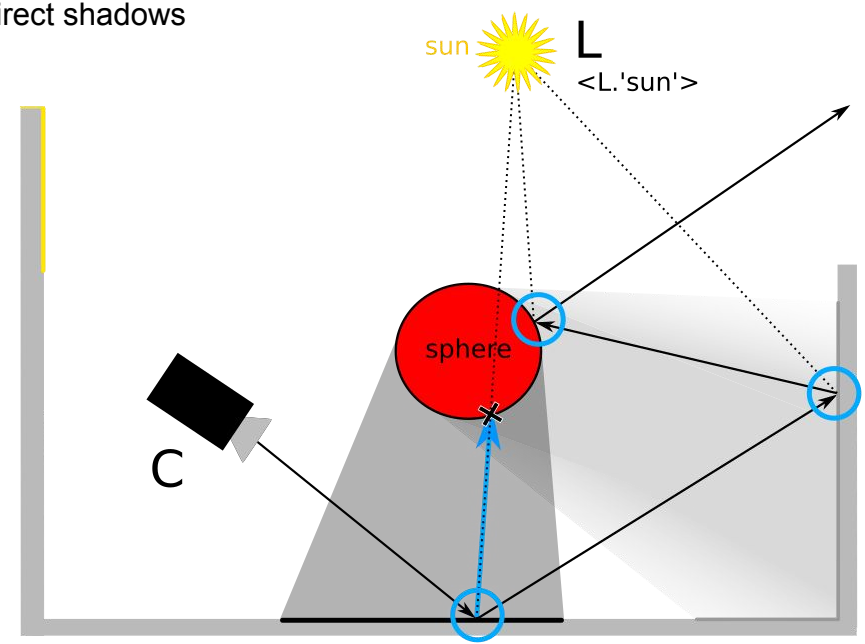
Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)
→ default path (Arnold)
····· shadow ray (Arnold)

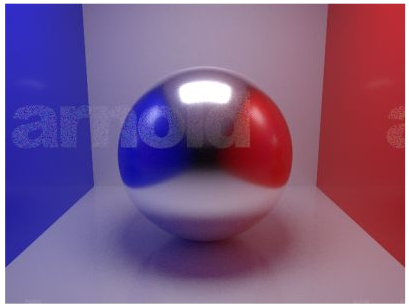
```
// our shader for direct shadowing
if catcher
   for ls in AiLightsGetSample() // light loops
      if not AiColorIsSmall(ls.Lo)
         ...
```

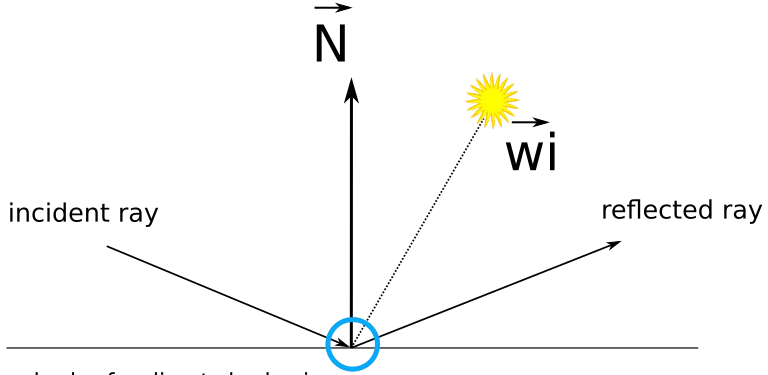
Arnold fait lui même cette intégration au niveau des lumières
Le code n'est pas accessible et modifiable en interne

Adaptation de l'algorithme de rendu

Direct shadows







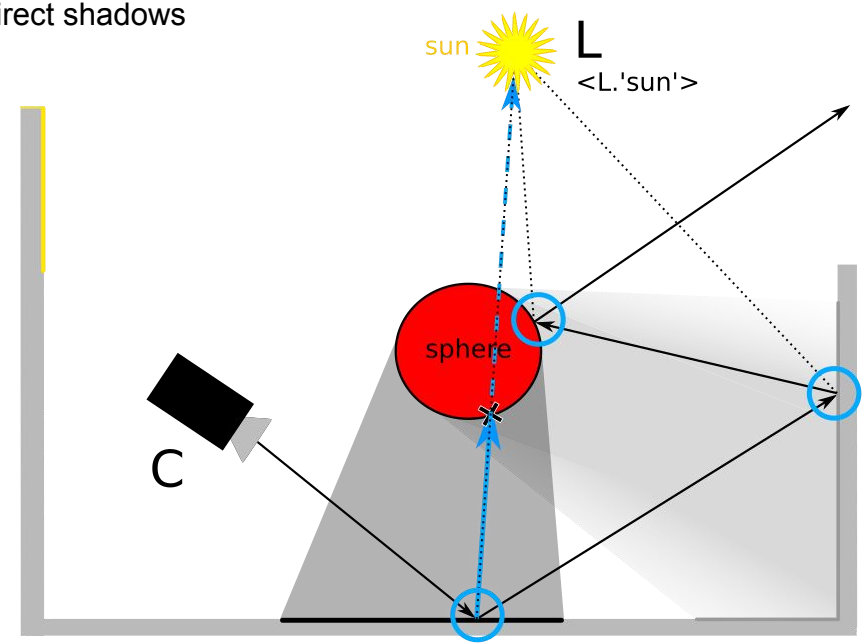
Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)
→ default path (Arnold)
shadow ray (Arnold)
→ first intersect (sl)

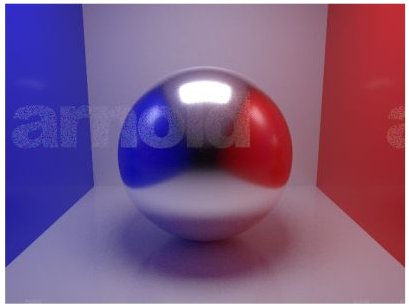
```
// our shader for direct shadowing
if catcher
   for ls in AiLightsGetSample() // light loops
      if not AiColorIsSmall(ls.Lo)
         if firstHitObject
            ...
```

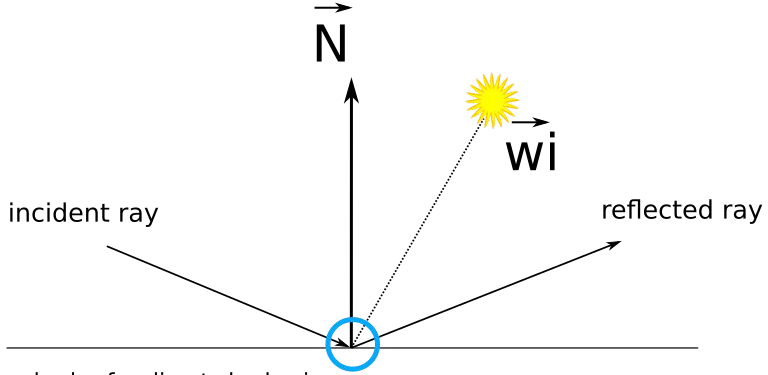
Arnold fait lui même cette intégration au niveau des lumières
Le code n'est pas accessible et modifiable en interne

Adaptation de l'algorithme de rendu

Direct shadows







Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)

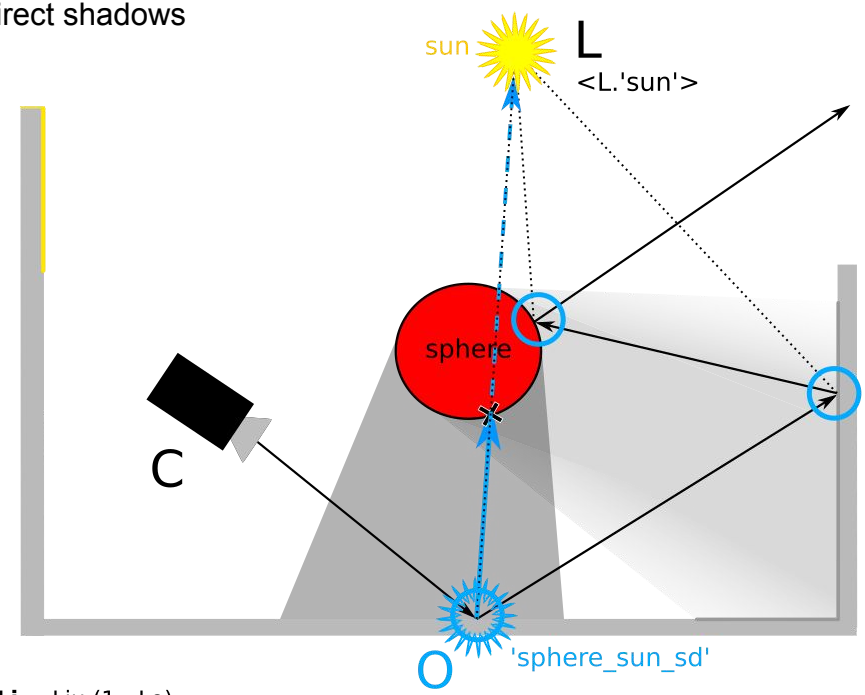
- default path (Arnold)
- shadow ray (Arnold)
- first intersect (sl)
- second intersect, skip direct (sl)

```
// our shader for direct shadowing
if catcher
   for ls in AiLightsGetSample() // light loops
      if not AiColorIsSmall(ls.Lo)
         if firstHitObject and not secondHitObject
            sl_direct = BSDF * cos(N,wi) * ls.Liu * ls.Lo / ls.pdf
```

Arnold fait lui même cette intégration au niveau des lumières
Le code n'est pas accessible et modifiable en interne

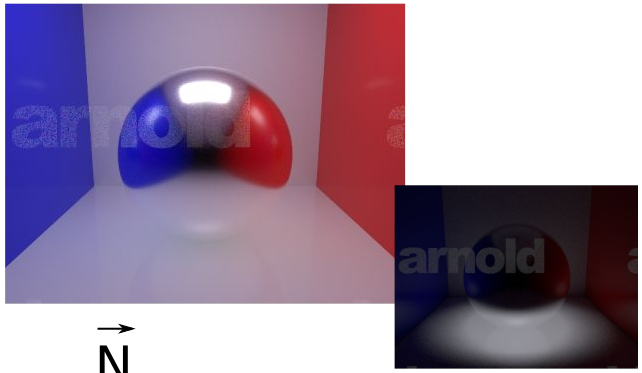
Adaptation de l'algorithme de rendu

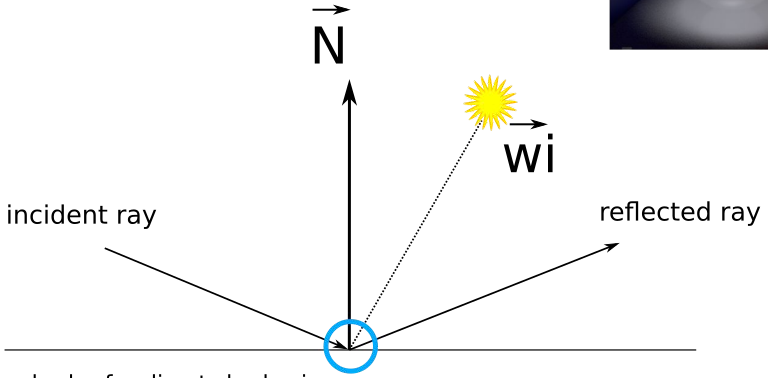
Direct shadows



Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)

- default path (Arnold)
- shadow ray (Arnold)
- first intersect (sl)
- second intersect, skip direct (sl)



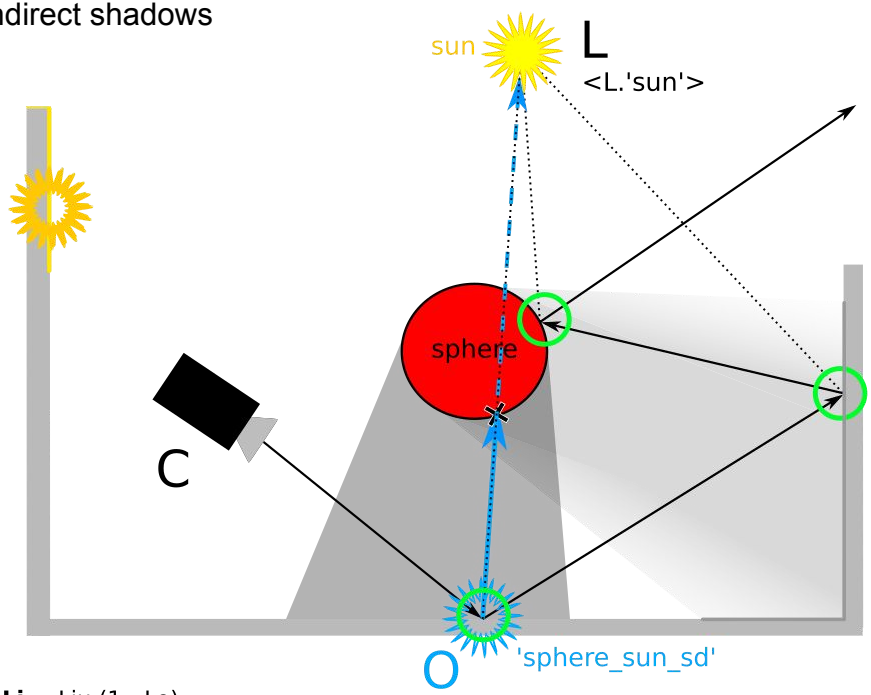


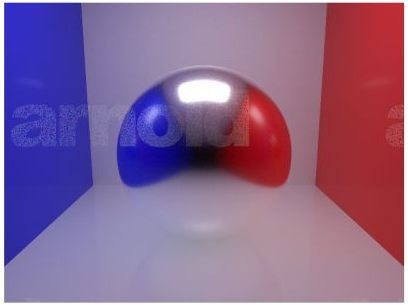

```
// our shader for direct shadowing
if catcher
   for ls in AiLightsGetSample() // light loops
      if not AiColorIsSmall(ls.Lo)
         if firstHitObject and not secondHitObject
            sl_direct = BSDF * cos(N,wi) * ls.Liu * ls.Lo / ls.pdf
            closure = AiClosureEmission(color=sl_direct)
            AiClosureSetLabel(closure, firstHitObject.name + "_" + ls.name + "_sd")
            out.CLOSURE().add(closure)
```

Arnold fait lui même cette intégration au niveau des lumières
Le code n'est pas accessible et modifiable en interne

Adaptation de l’algorithme de rendu

Indirect shadows





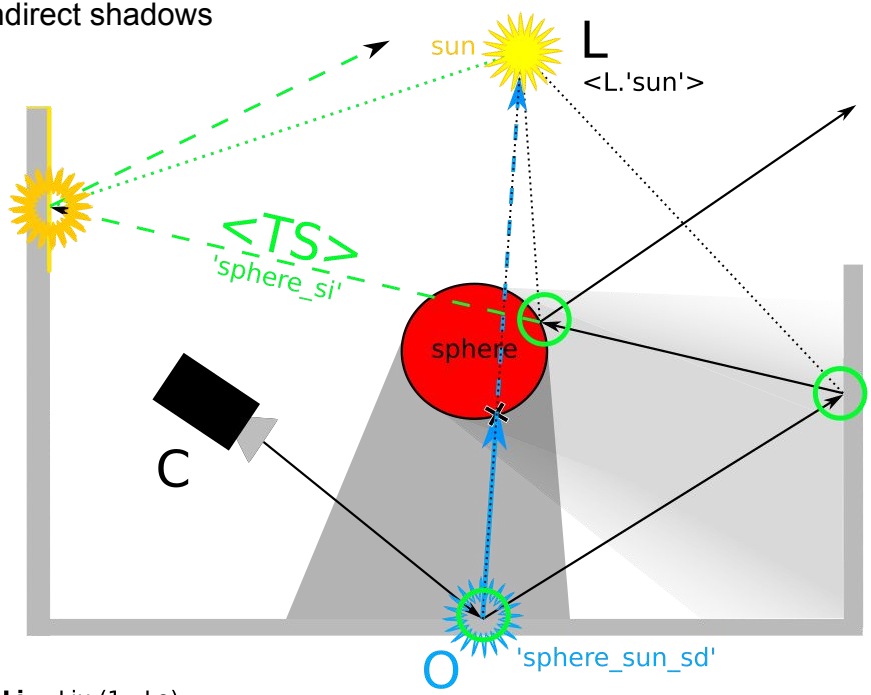
Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)

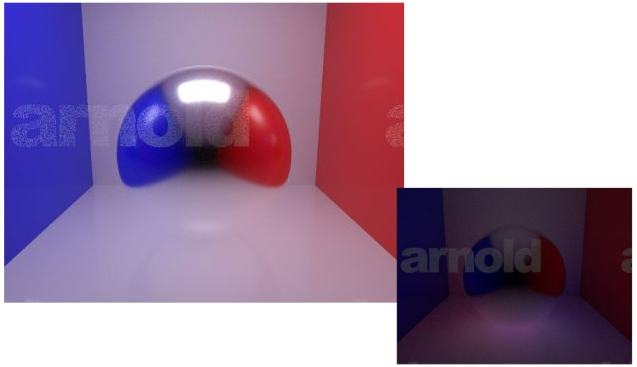
- default path (Arnold)
- shadow ray (Arnold)
- first intersect (sl)
- second intersect, skip direct (sl)
- skipped indirect path (sl)
- shadow ray for skipped path (sl)

```
// our shader for indirect shadowing
if caster
   ...
```

Adaptation de l'algorithme de rendu

Indirect shadows





Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)

- default path (Arnold)
- shadow ray (Arnold)
- first intersect (sl)
- second intersect, skip direct (sl)
- skipped indirect path (sl)
- shadow ray for skipped path (sl)

```
// our shader for indirect shadowing
if caster
   bsdf = AiMicrofacetRefractionBSDF(distribution=GGX
                                     BECKMANN, ior=1.0,
                                     label=caster.name + "_si")
   out.CLOSURE().add(bsdf)
```

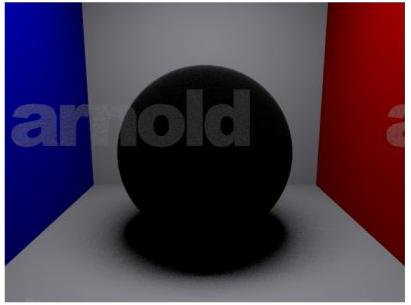
Adaptation de l’algorithme de rendu

LPE (Light Path Expression)

diffuse_direct

C<RD>L

<RD> : Diffuse reflection



diffuse_indirect

C<RD>[DSVOB].*

V : Volume scattering

B : Background emission

S : Specular reflection or refraction

D : Diffuse reflection, transmission or subsurface scattering

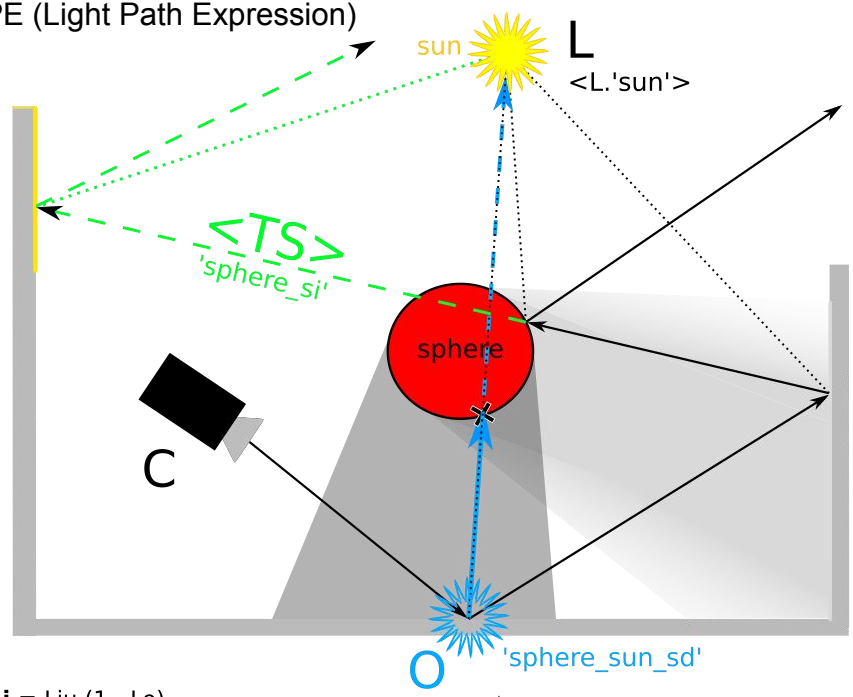
specular_direct

C<RS>L

<RS> : Specular reflection

Adaptation de l'algorithme de rendu

LPE (Light Path Expression)

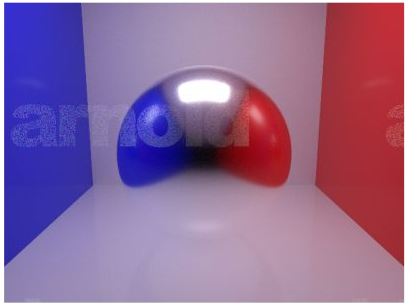


Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)

- default path (Arnold)
- shadow ray (Arnold)
- first intersect (sl)
- second intersect, skip direct (sl)
- skipped indirect path (sl)
- shadow ray for skipped path (sl)

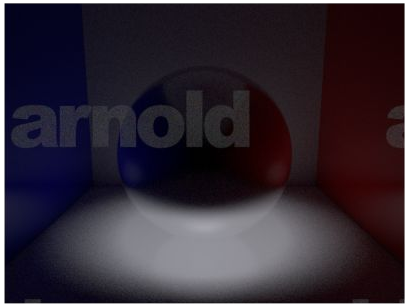
LPE syntax events :
C : Camera
. : Any event
A* : 0 or more A events
L : Emission from all lights
'myevent' : Custom event
O : Surface/volume emission
<TS> : Specular refraction
[ABC] : A, B or C

C.*



sl_direct

.*'sphere_sun_sd'



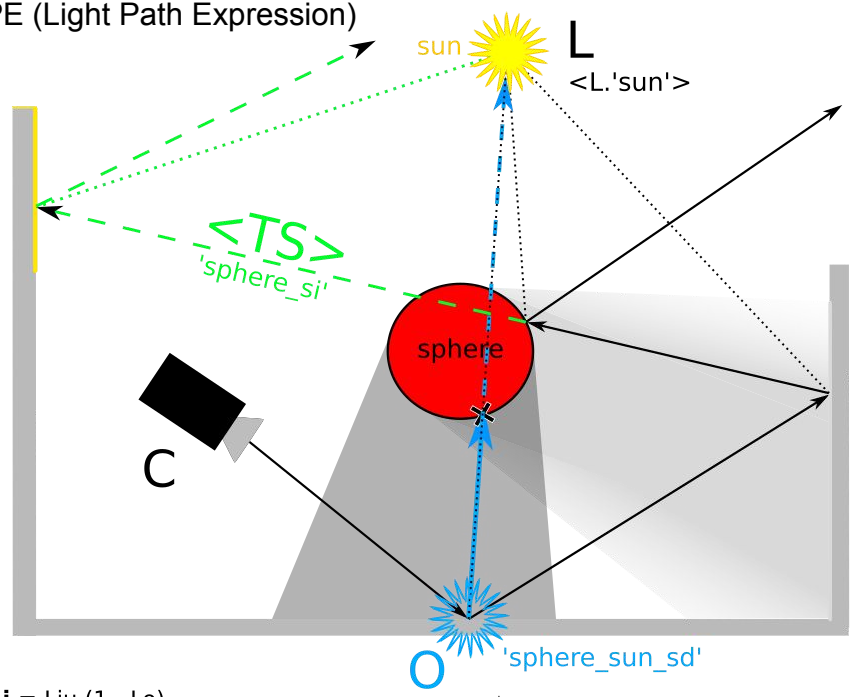
sl_indirect

.*'sphere_si'.*

.*'sphere_si'.*<L.'sun'>

Adaptation de l'algorithme de rendu

LPE (Light Path Expression)

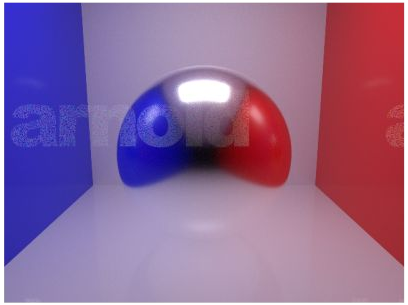


Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)

- default path (Arnold)
- shadow ray (Arnold)
- first intersect (sl)
- second intersect, skip direct (sl)
- skipped indirect path (sl)
- shadow ray for skipped path (sl)

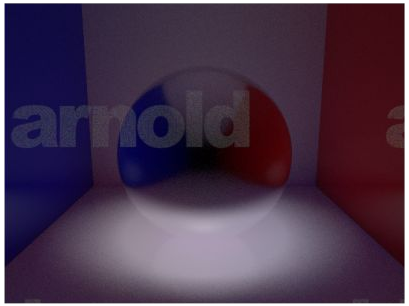
LPE syntax events :
C : Camera
. : Any event
A* : 0 or more A events
L : Emission from all lights
'myevent' : Custom event
O : Surface/volume emission
<TS> : Specular refraction
[ABC] : A, B or C
[^A] : Any event except A

C.*



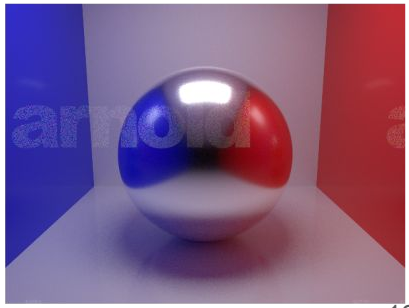
sl

.*['sphere_sun_sd''sphere_si'].*



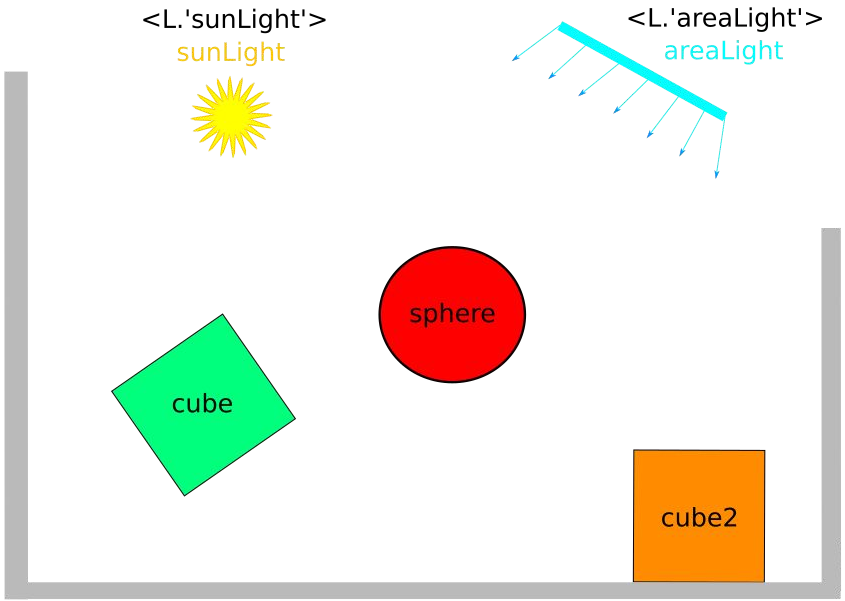
sl_beauty

C[^'sphere_sun_sd''sphere_si']*



Activation de calques (AOVs : Arbitrary Output Variables)

Arnold scene description file (.ass)



Activation de calques (AOVs : Arbitrary Output Variables)

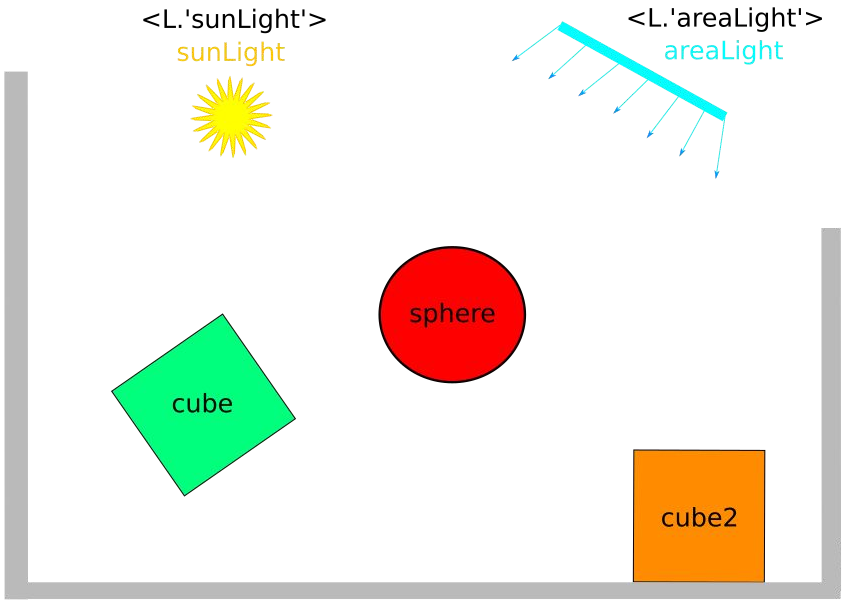
Arnold scene description file (.ass)

```
options
{
 ...
 light_path_expressions 22 1 STRING
  "sl_beauty C[^'sphere_sunLight_sd''sphere_areaLight_sd''cube_sunLight_sd''cube_areaLight_sd''cube2_sunLight_sd''cube2_areaLight_sd''sphere_si''cube_si''cube2_si']*"
  "sl .*['sphere_sunLight_sd''sphere_areaLight_sd''cube_sunLight_sd''cube_areaLight_sd''cube2_sunLight_sd''cube2_areaLight_sd''sphere_si''cube_si''cube2_si'].*"
  "sl_direct .*['sphere_sunLight_sd''sphere_areaLight_sd''cube_sunLight_sd''cube_areaLight_sd''cube2_sunLight_sd''cube2_areaLight_sd']"
  "sl_indirect .*['sphere_si''cube_si''cube2_si'].*"
  "sl_sphere_sunLight .*('sphere_sunLight_sd'|('sphere_si'.*<L.'sunLight'>))"
  "sl_sphere_sunLight_d .*'sphere_sunLight_sd'"
  "sl_sphere_sunLight_i .*('sphere_si'.*<L.'sunLight'>)"
  "sl_sphere_areaLight .*('sphere_areaLight_sd'|('sphere_si'.*<L.'areaLight'>))"
  "sl_sphere_areaLight_d .*'sphere_areaLight_sd'"
  "sl_sphere_areaLight_i .*('sphere_si'.*<L.'areaLight'>)"
  "sl_cube_sunLight .*('cube_sunLight_sd'|('cube_si'.*<L.'sunLight'>))"
  "sl_cube_sunLight_d .*'cube_sunLight_sd'"
  "sl_cube_sunLight_i .*('cube_si'.*<L.'sunLight'>)"
  "sl_cube_areaLight .*('cube_areaLight_sd'|('cube_si'.*<L.'areaLight'>))"
  "sl_cube_areaLight_d .*'cube_areaLight_sd'"
  "sl_cube_areaLight_i .*('cube_si'.*<L.'areaLight'>)"
  "sl_cube2_sunLight .*('cube2_sunLight_sd'|('cube2_si'.*<L.'sunLight'>))"
  "sl_cube2_sunLight_d .*'cube2_sunLight_sd'"
  "sl_cube2_sunLight_i .*('cube2_si'.*<L.'sunLight'>)"
  "sl_cube2_areaLight .*('cube2_areaLight_sd'|('cube2_si'.*<L.'areaLight'>))"
  "sl_cube2_areaLight_d .*'cube2_areaLight_sd'"
  "sl_cube2_areaLight_i .*('cube2_si'.*<L.'areaLight'>)"

 // AOVs
 outputs 5 1 STRING
   "sl_beauty RGBA filter driver"
   "sl RGBA filter driver2"
   "sl_direct RGBA filter driver3"
   "sl_indirect RGBA filter driver4"
   "sl_cube2_areaLight_i RGBA filter driver5"
}
```

```
gaussian_filter {
  name filter
  width 3
}

driver_png {
  name driver
  filename "beauty.png"
  color_space "sRGB"
}
```



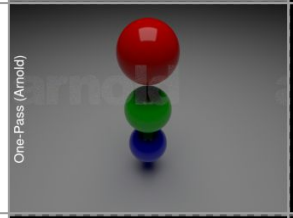
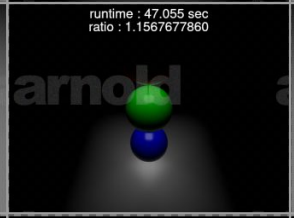
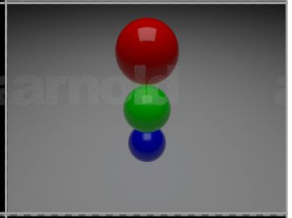
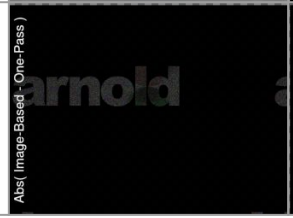
Activation de calques (AOVs : Arbitrary Output Variables)

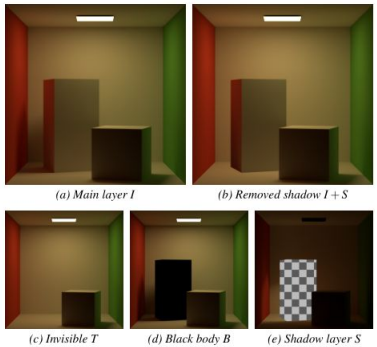
Maya

Voir vidéo de présentation

Test de validation

Test par différences d'images (espace image)

	Beauty	sI	Beauty + sI
Référence			
Implémentation Arnold			
Ref - Impl			

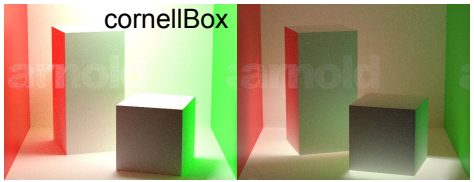


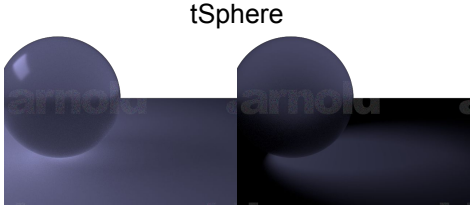
© Global Illumination Shadow Layers (2019)

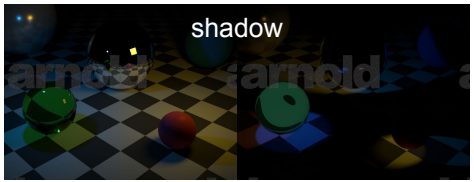
= 0

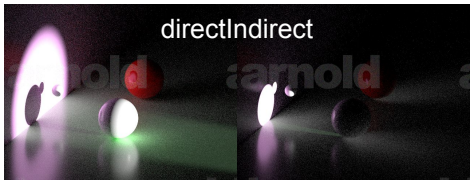
Mesures performances

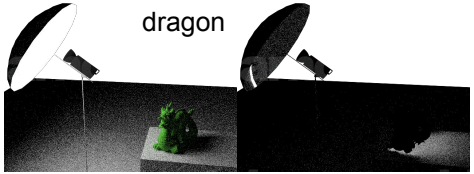
- GISL pbrt : 30%
- Arnold : ?









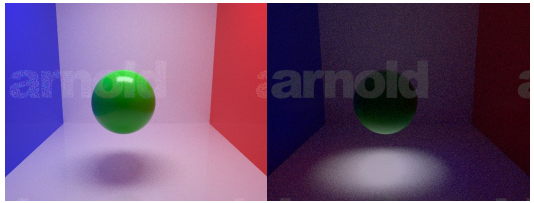


Scene	Beauty	Beauty + sl	Surcoût sl
tSphere	44s	61s	+39%
cornell	82s	118s	+44%
shadow	30s	40s	+33%
3alignSpheres	41s	47s	+16%
directIndirect	20s	26s	+29%
spotFog	12s	15s	+25%
dragon	7s	9s	+30%
cornellBox	123s	239s	+94%

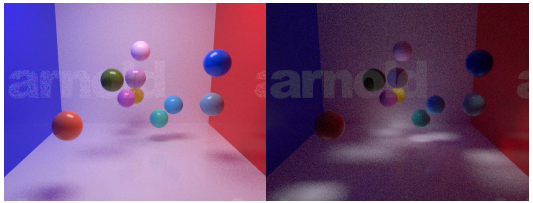
Surcoût sl de 16% à 94%
Testé sur 8 scènes

Mesures performances

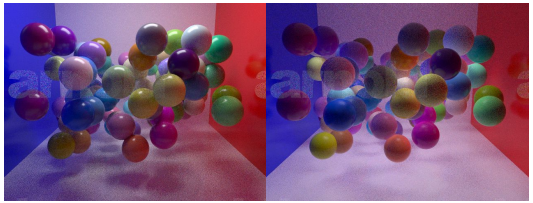
Par rapport aux nombres d’objets



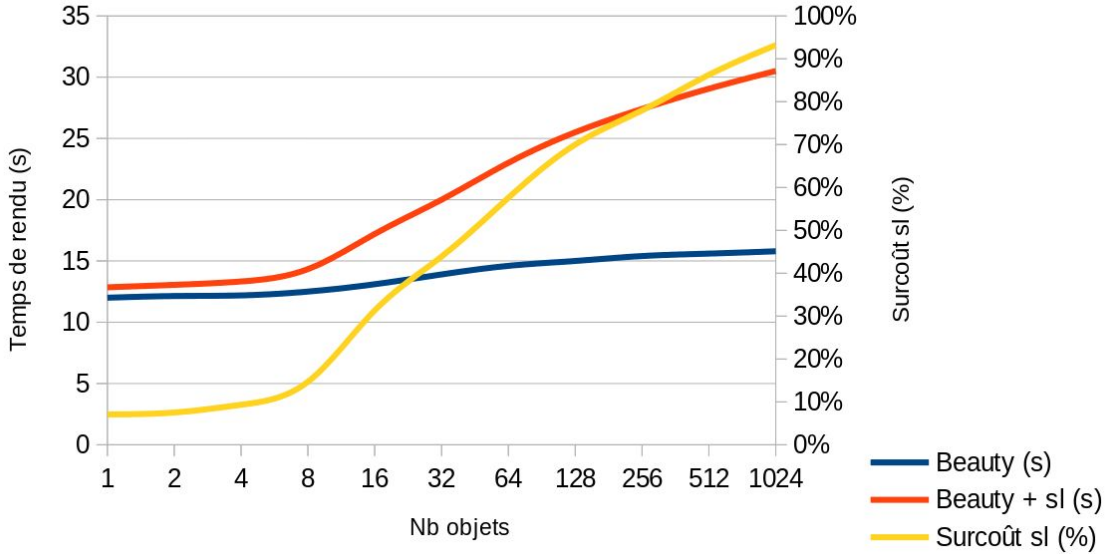
Nb objets = 1



Nb objets = 10



Nb objets = 100

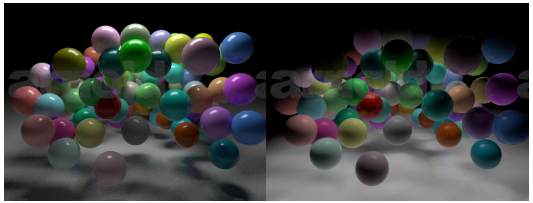


Mesures performances

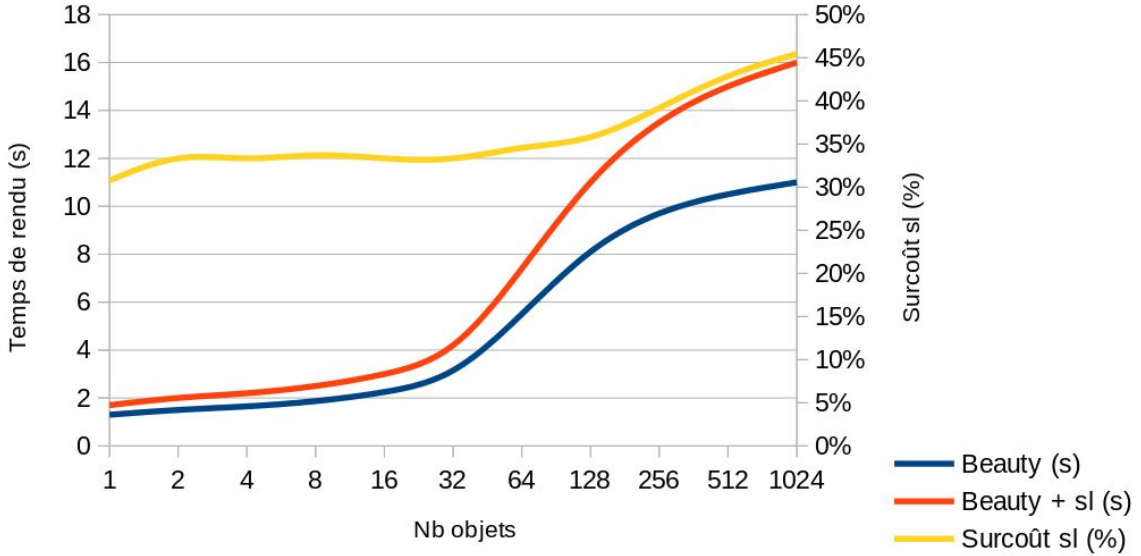
Par rapport aux nombres d'objets

Nb objets = 1

Nb objets = 10

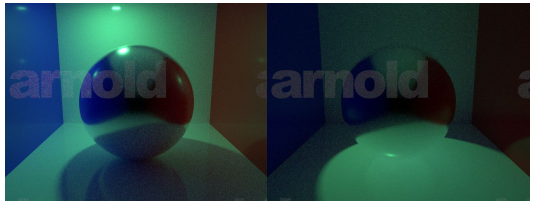


Nb objets = 100

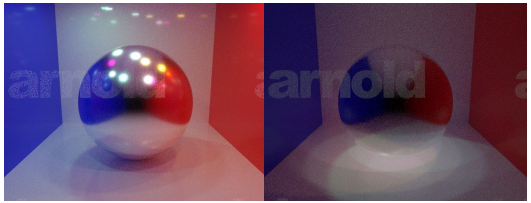


Mesures performances

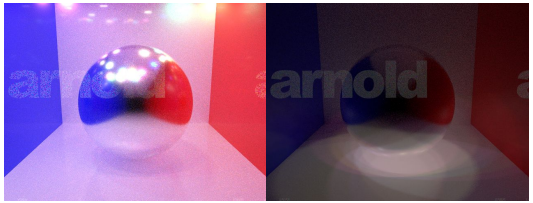
Par rapport aux nombres de lumières



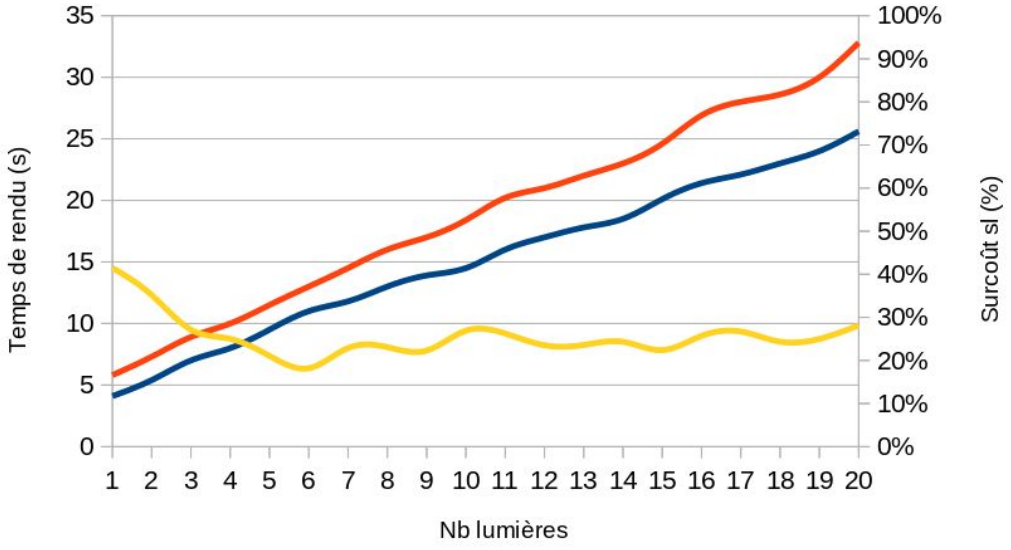
Nb lumières = 1



Nb lumières = 10

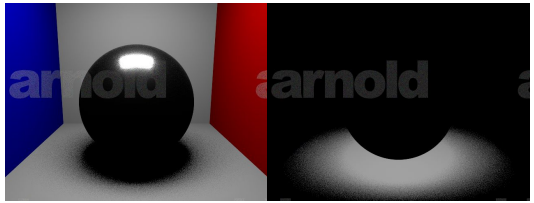


Nb lumières = 20

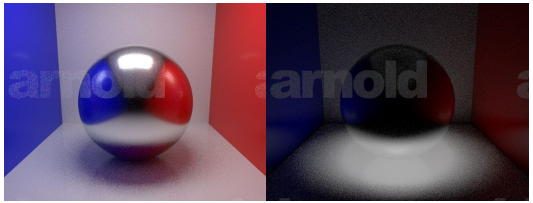


Mesures performances

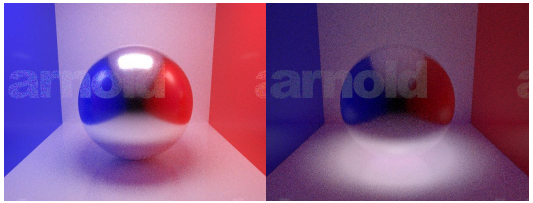
Par rapport à la profondeur maximale des chemins



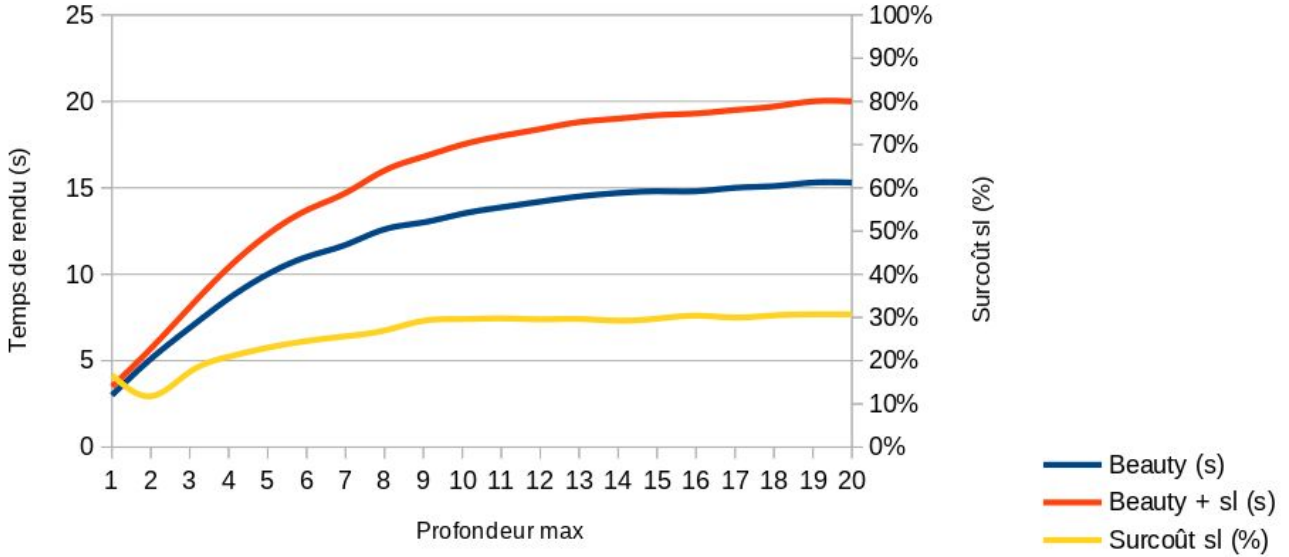
Profondeur max = 1



Profondeur max = 2



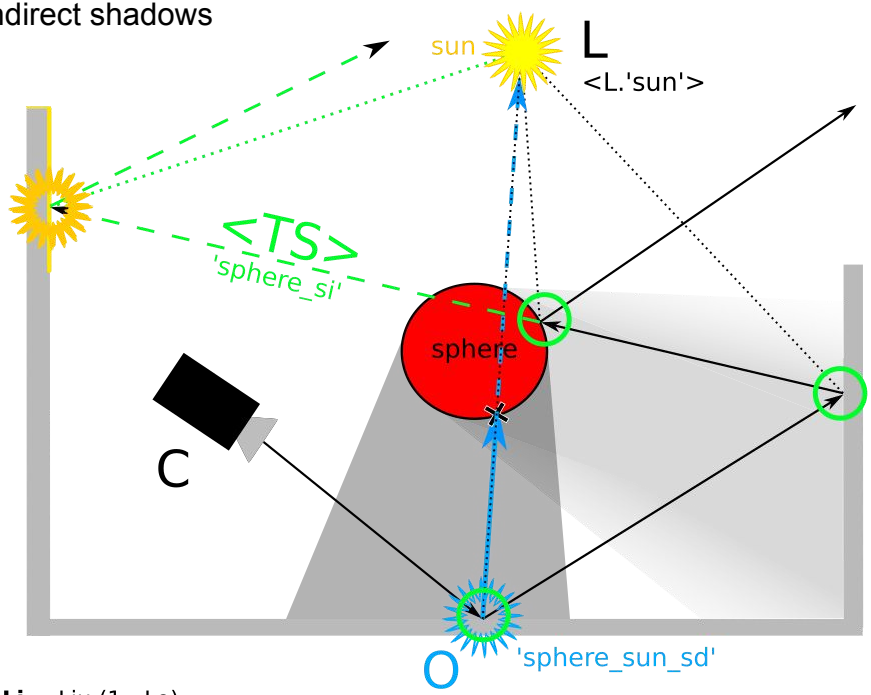
Profondeur max = 20



Merci de votre attention

Adaptation de l'algorithme de rendu

Indirect shadows

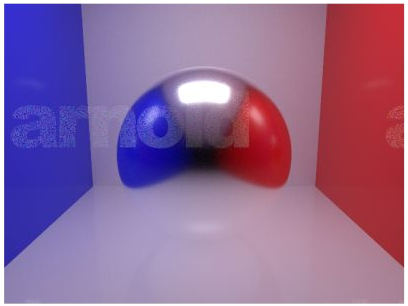


Li = Liu (1 - Lo)
Lo : facteur d'occlusion
Liu : rayonnements incidents non occultés
Energie perdue = Liu * Lo (shadow matte)

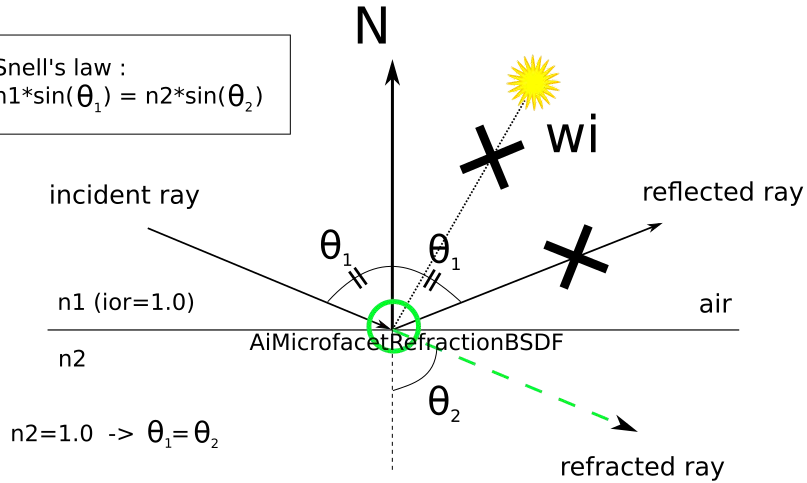
- default path (Arnold)
- shadow ray (Arnold)
- first intersect (sl)
- second intersect, skip direct (sl)
- skipped indirect path (sl)
- shadow ray for skipped path (sl)

LPE syntax events :
C : Camera
. : Any event
A* : 0 or more A events
L : Emission from all lights
'myevent' : Custom event
O : Surface/volume emission
<TS> : Specular refraction

Beauty
C.*



Snell's law :
$n1*\sin(\theta_1) = n2*\sin(\theta_2)$




```
// our shader for indirect shadowing
if caster
   bsdf = AiMicrofacetRefractionBSDF(distribution=GGX|BECKMANN, ior=1.0,
                                     label=caster.name + "_si")
   out.CLOSURE().add(bsdf)
```