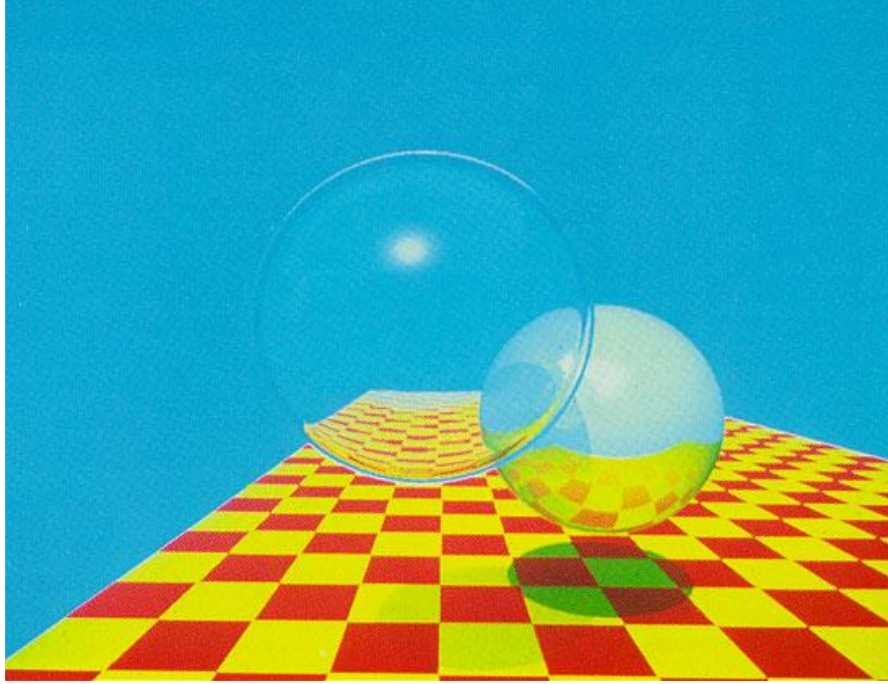
Γραφικά Υπολογιστών

Αναδρομική Παρακολούθηση Ακτίνας (Ray-tracing)

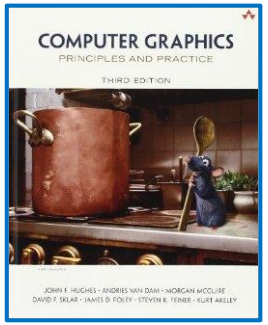
Andreas Aristidou
andarist@ucy.ac.cy
http://www.andreasaristidou.com



Images:
(Left) http://cg.alexandra.dk/?p=278,
(Upper Right) Turner Whitted,"An improved illumination model for shaded display, " Bell Laboratories 1980.
(Bottom Right), NVIDIA booth SIGGRAPH 2018 (Don Greenberg, AvD)

Αναφορά

- Περισσότερα στο Κεφάλαιο 7, 15 του βιβλίου



"Computer Graphics: Principles and Practice", J. F. Hughes, A. van Dam, M. McGuire, D. F. Sklar, J. D. Foley, S. K. Feiner, K. Akeley, Addison-Wesley, 2013

Rendered with NVIDIA I-ray, a photorealistic rendering solution which adds physically accurate global illumination on top of ray tracing using a combination of physically-based rendering techniques

iray Gallery, *Image credit*

Παραδείγματα Ray-Tracing

Images from the POVRAY Hall of Fame: hof.povray.org

Παραδείγματα Ray-Tracing

Images from the POVRAY Hall of Fame: hof.povray.org

Παραδείγματα Ray-Tracing

Images from the POVRAY Hall of Fame: hof.povray.org

Παραδείγματα Ray-Tracing

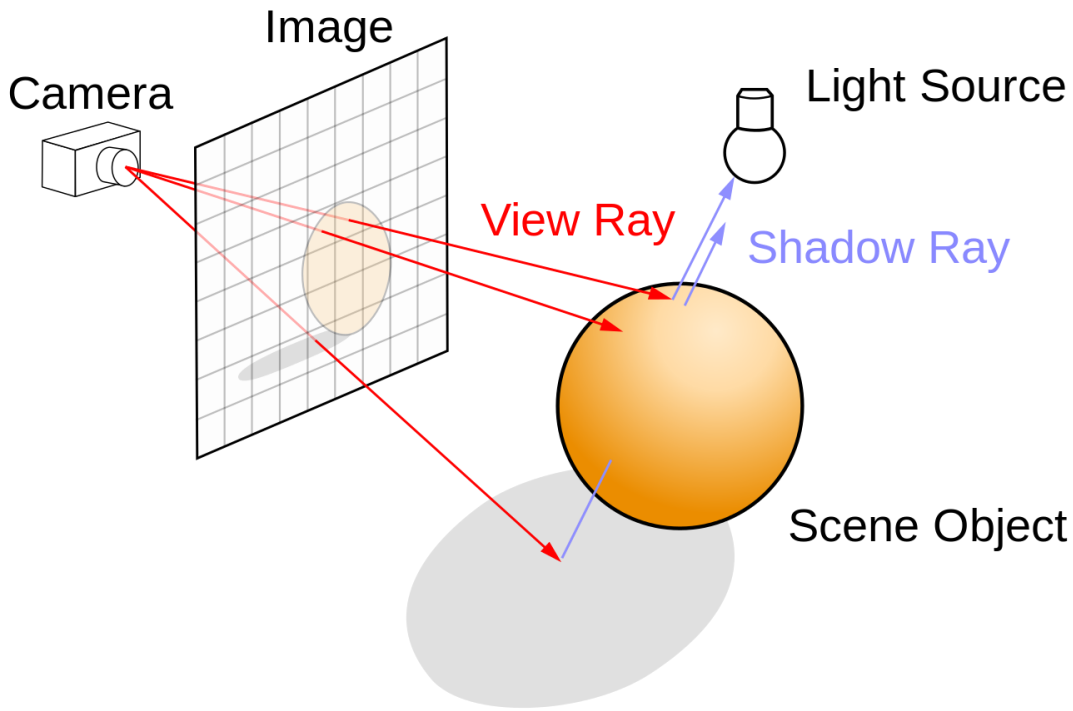
Images from the POVRAY Hall of Fame: hof.povray.org

Παραδείγματα Ray-Tracing

Images from the POVRAY Hall of Fame: hof.povray.org

Σύνοψη - Ray Shooting (βολή ακτίνας)

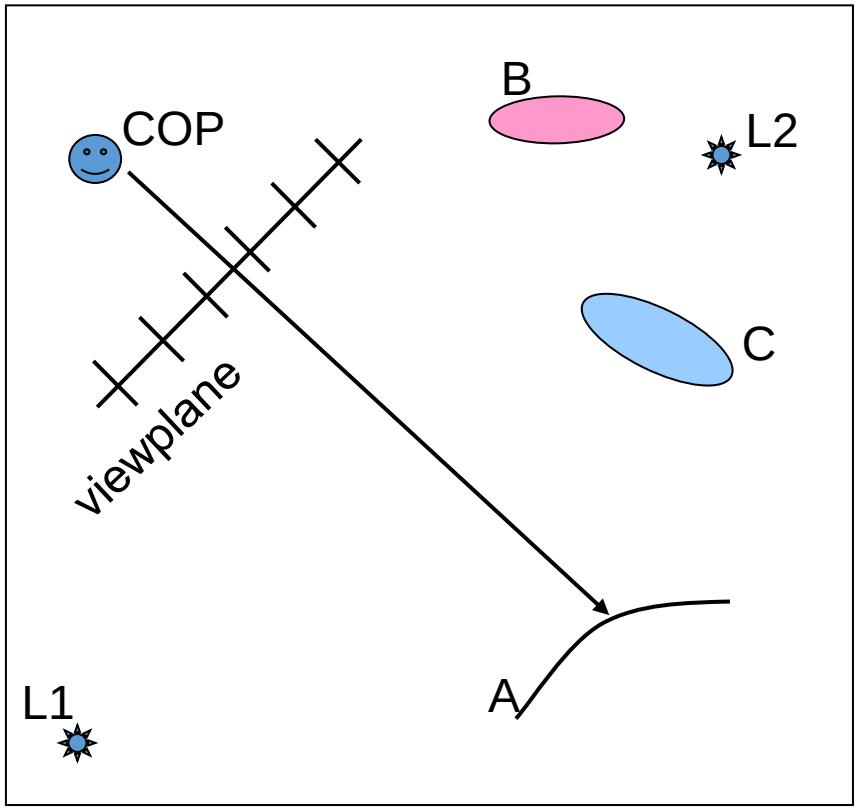
- Πως μπορούμε να δημιουργούμε εικόνες
  - Υπολογίζουμε την ακτίνα από την κάμερα και το κέντρο κάθε pixel
  - Βρίσκουμε την τομή με το πιο κοντινό αντικείμενο
  - Βρίσκουμε τον τοπικό φωτισμό στο σημείο τομής
  - Χρησιμοποιούμε αυτό το χρώμα για να ζωγραφίσουμε το pixel



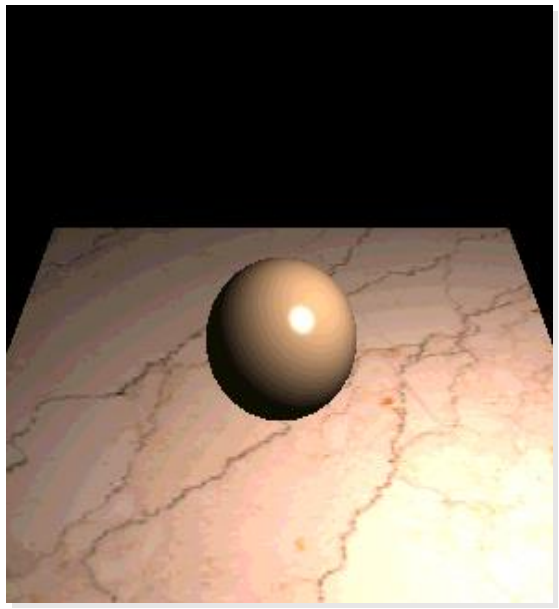
Σύνοψη: τοπικός φωτισμός

$$I_r = k_a I_a + \sum_{j=1}^{p} I_j \left(k_d (n.l_i) + k_s (h_j.n)^m \right)$$

Τοπικός Φωτισμός = Έμμεσος + Για κάθε φωτεινή πηγή (Διάχυτος + Κατευθυνόμενος Φωτισμός)

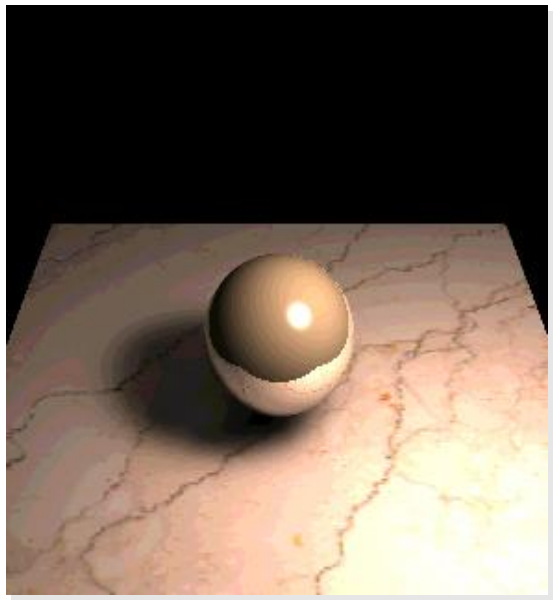


Σύνοψη: Local vs. Global Illumination



Local

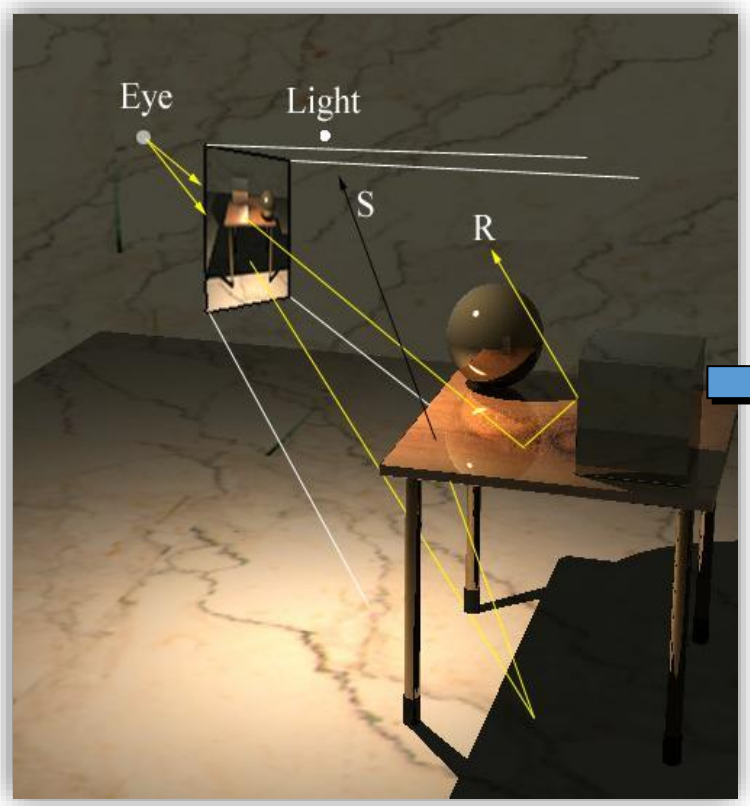
Ο τοπικός φωτισμός εξαρτάται μόνο από το αντικείμενο και τις φωτεινές πηγές



Global

Ο φωτισμός σε ένα σημείο εξαρτάται και από άλλα αντικείμενο της σκηνής

Σύνοψη: Η λύση εξαρτάται από τη οπτική γωνία (Ray-tracing)



Scene Geometry

Solution determined only for directions through pixels in the viewport

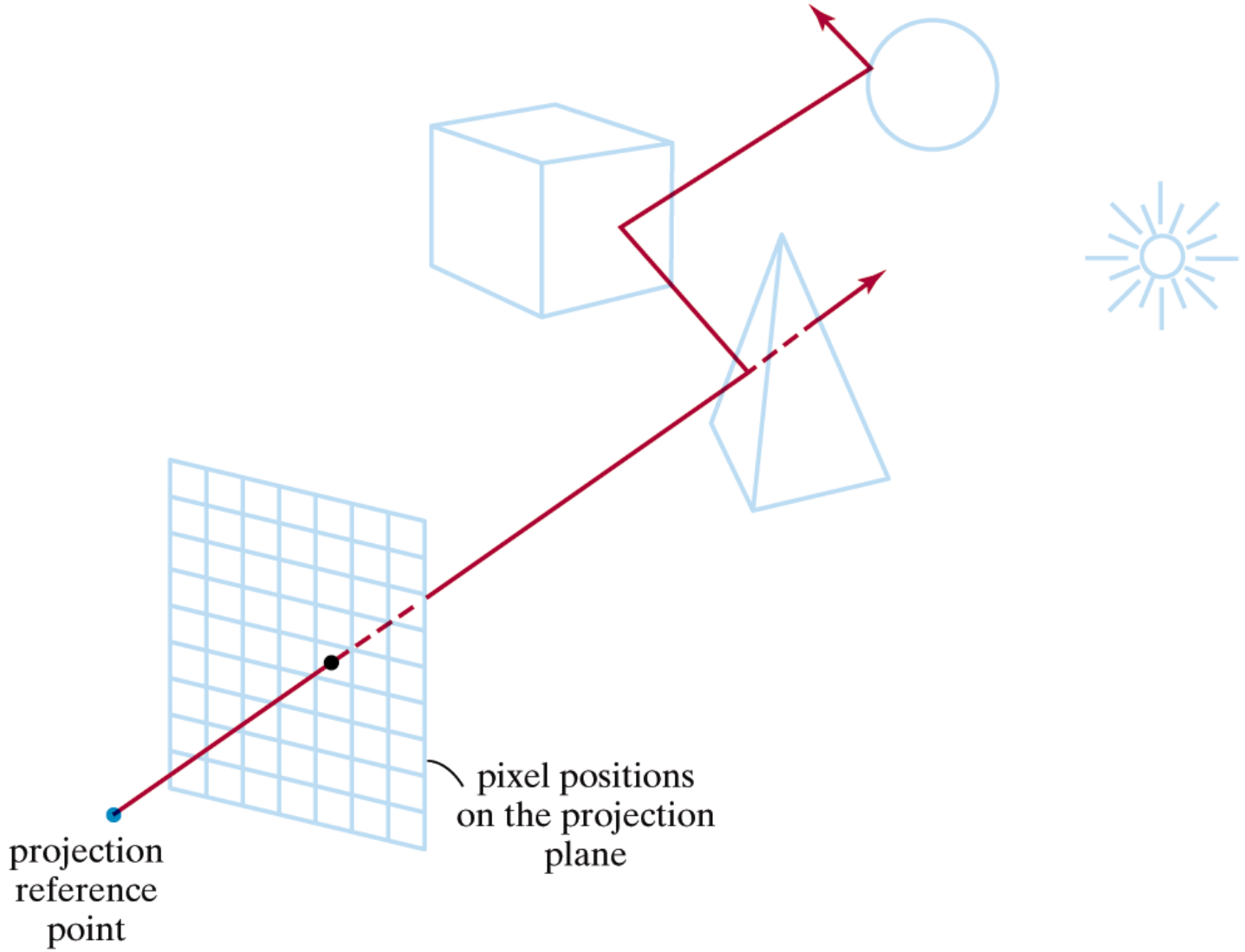
Σήμερα

- Αναδρομική Παρακολούθηση Ακτίνας
- Σκιές
- Γενικός φωτισμός
- Κανόνας του Snell’s για διάθλαση
- Πότε σταματά η αναδρομή;
- Antialiasing

Ray Tracing

- Παρακολούθηση μιας ακτίνας του φωτός δια μέσου της σκηνής
- Οι ακτίνες ρίχνονται σε κάθε pixel. Αντανακλώνται, διαθλώνται, ή απορροφώνται κάθε φορά που τέμνονται με αντικείμενα
- Ας δούμε ένα παράδειγμα

Ray-Tracing Setup



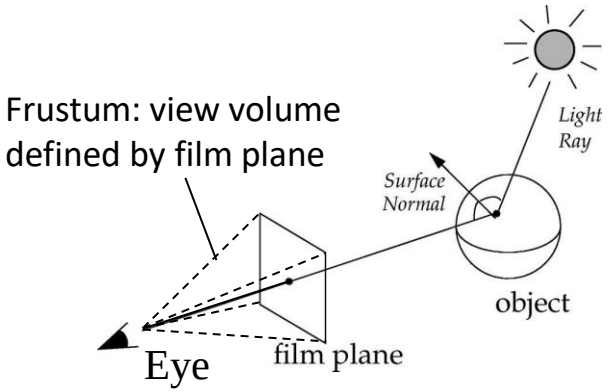
Images taken from Hearn & Baker, "Computer Graphics with OpenGL" (2004)

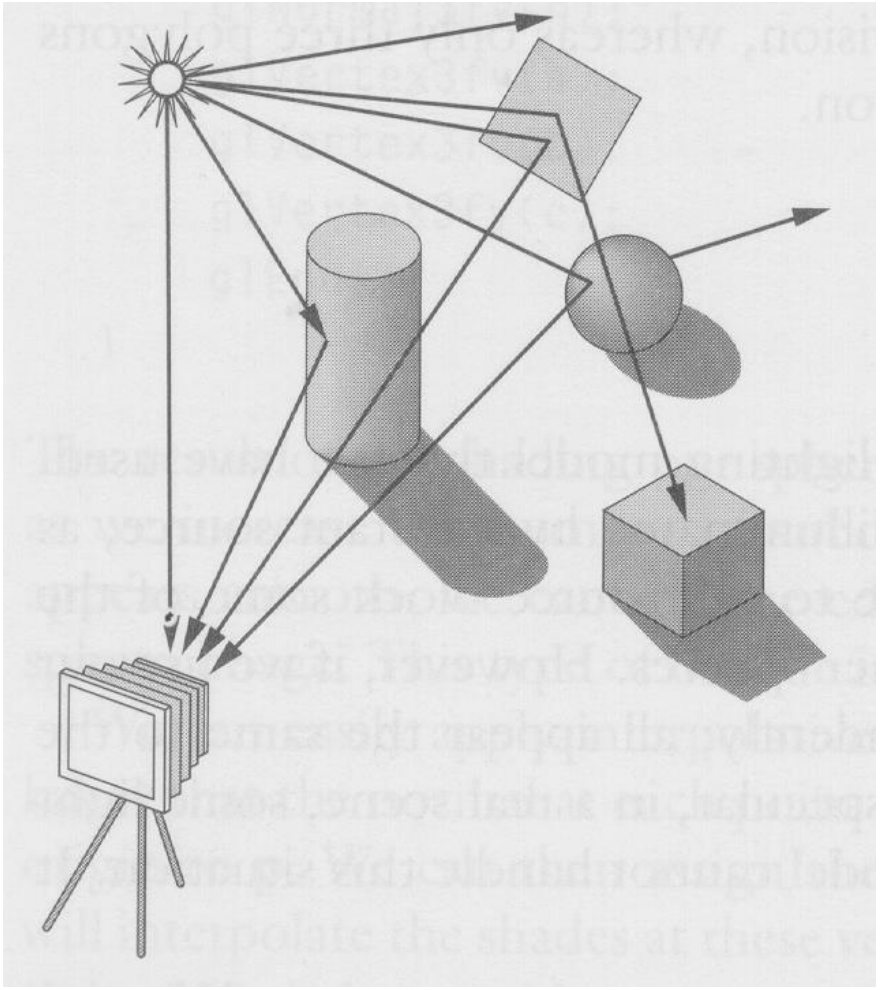
Ray Tracing

- Η παρακολούθηση ακτίνας γίνεται ως εξής:
  - Ρίχνουμε μια ακτίνα από κάθε pixel στη σκηνή κατά μήκος της διαδρομής προβολής
  - Προσδιορίστε ποιες επιφάνειες τέμνει η ακτίνα και ορίστε αυτές σύμφωνα με την απόσταση τους από το pixel
  - Η πλησιέστερη επιφάνεια στο pixel είναι η ορατή επιφάνεια για αυτό το pixel
  - Αντανακλάστε την ακτίνα από την ορατή επιφάνεια κατά μήκος της κατοπτρικής γωνίας ανάκλασης
  - Για τις διαφανείς επιφάνειες στείλετε επίσης μια ακτίνα μέσω της επιφάνειας στην κατεύθυνση διάθλασης
  - Επαναλάβετε τη διαδικασία για αυτές τις δευτερεύουσες ακτίνες

Ray Tracing Overview

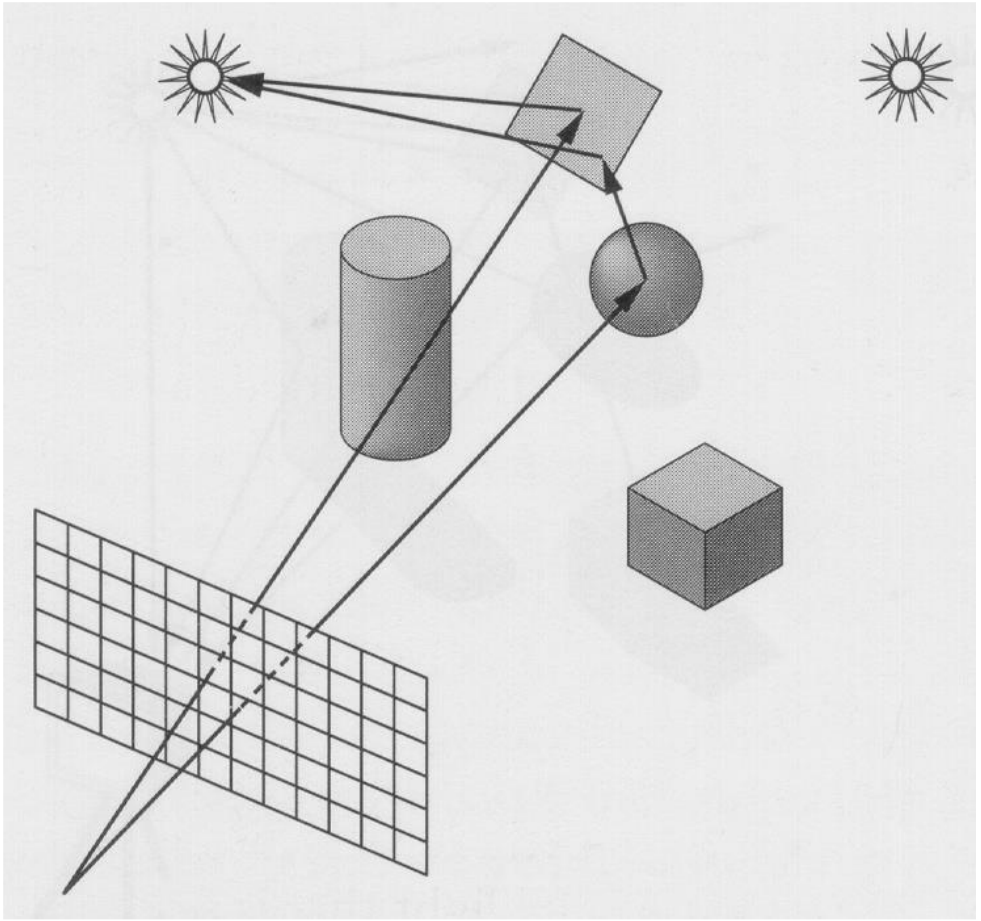
- **Generate primary ray**
  - shoot rays from eye through sample points on view (film) plane
  - sample point is typically center of a pixel, but alternatively supersample pixels
- **Ray-object intersection – 3D computational geometry**
  - find first object in scene that ray intersects with (if any)
  - solves VSD/HSR problem – use parametric line equation for ray, so smallest t value
- **Calculate lighting (i.e., color)**
  - use illumination model to determine direct contribution from light sources (light rays)
  - reflective objects recursively generate secondary rays (indirect contribution) that also contribute to color; RRT (Recursive Ray Tracing) only uses specular reflection rays because they are easy to compute and specular reflections are typically brighter than average. Should also handle refraction and shadows
  - sum of contributions determines color of sample point (underlit => ambient term)
  - no diffuse reflection rays → RRT is limited approximation to global, direct illumination
- **Finesse shading rule – evaluate full lighting eqn at each sample point, based on *n***





Forward ray tracing
(and real world)

Backward ray tracing



Ray Tracing vs. Triangle Scan Conversion

- How is ray tracing different from polygon scan conversion/rasterization? Shapes and Scene-view both use polygon scan conversion to render objects in a scene and have same pseudocode that starts in normalized world space, i.e., “object precision” a la Sutherland:

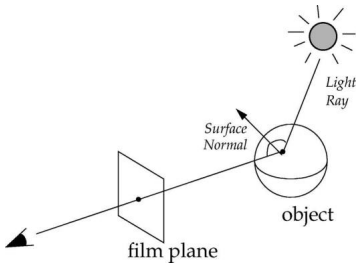
```
for each object in scene:
    for each triangle in object:
        pass vertex geometry, camera matrices, and
 lights to the shader program,
        which renders each triangle (using z-buffer)
 into framebuffer; use simple or complex
       illumination model
```

Ray Tracing vs. Triangle Scan Conversion

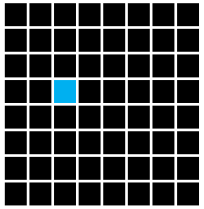
- Ray tracing uses the following pseudocode, “image precision”, a la Sutherland

```
for each sample in film plane:
    determine closest object in scene hit by
a ray going through that sample from eye point
    set color based on calculation of simple/complex
        illumination model for intersected object
```

- Note the distinction: polygonal scan conversion iterates over all **VERTICES** whereas ray tracing iterates over **2D (sub)PIXELS** and calculates intersections in a 3D scene



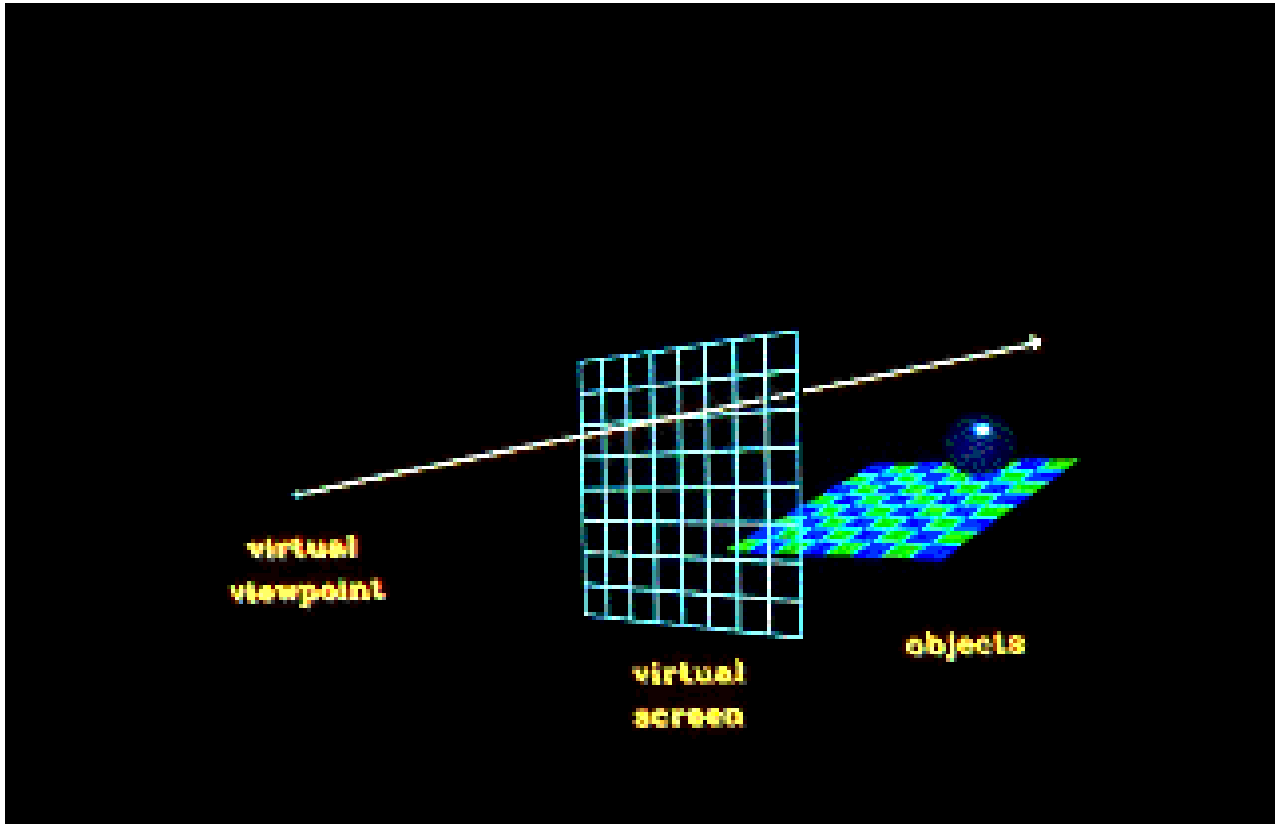
(ray intersection rendered to screen)



- For polygonal scan conversion must mesh curved objects while with ray tracing we can use an implicit equation directly (if it can be determined) – see Slide 16

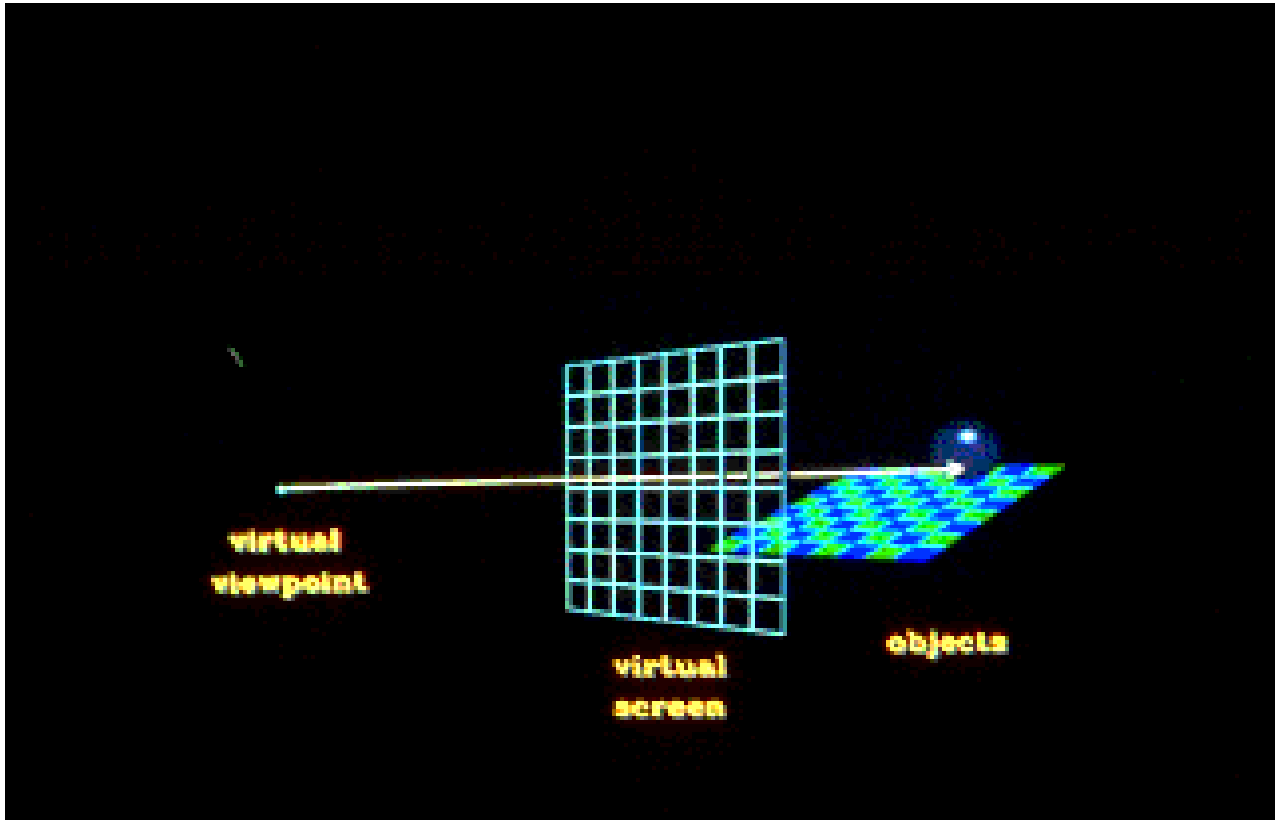
Ray Tracing

- Όταν η ακτίνα δεν προσπίπτει πάνω σε κάποιο αντικείμενο, το pixel χρωματίζεται με το χρώμα του φόντου

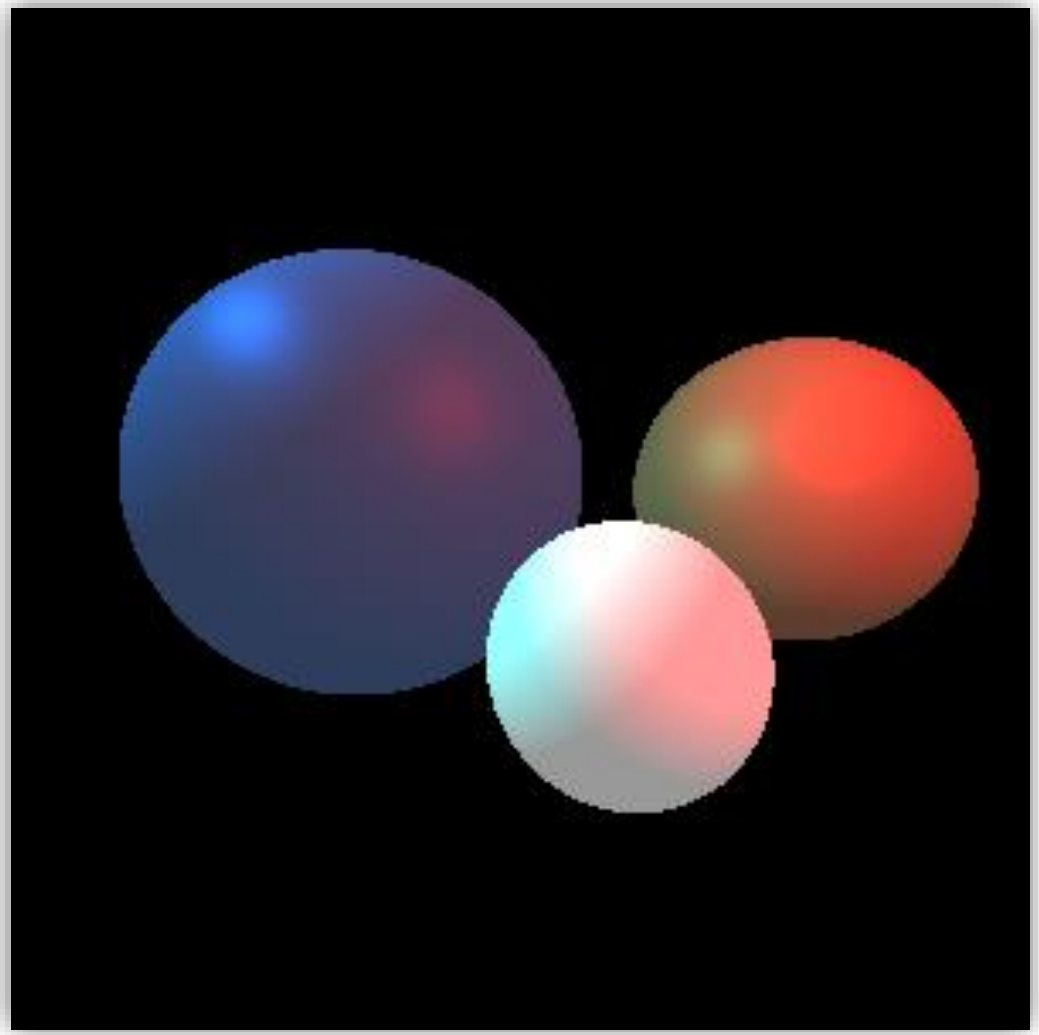


Ray Tracing

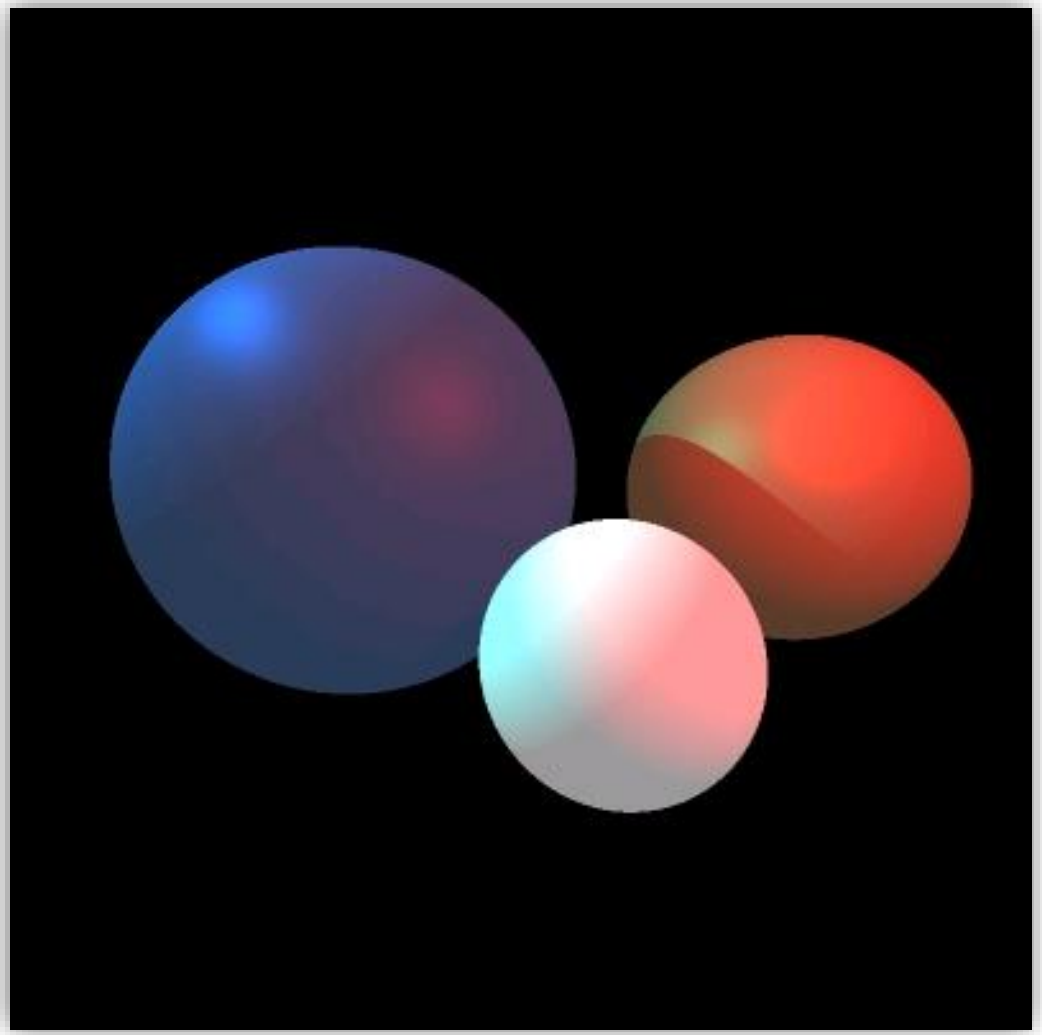
- Όταν η ακτίνα χτυπά ένα αντικείμενο: μια δευτερεύουσα ακτίνα, αποκαλούμενη «σκιά» ακτίνα, φεύγει προς τις φωτεινές πηγές



Βολή ακτίνας – χωρίς σκιές (μοντέλο Phong)



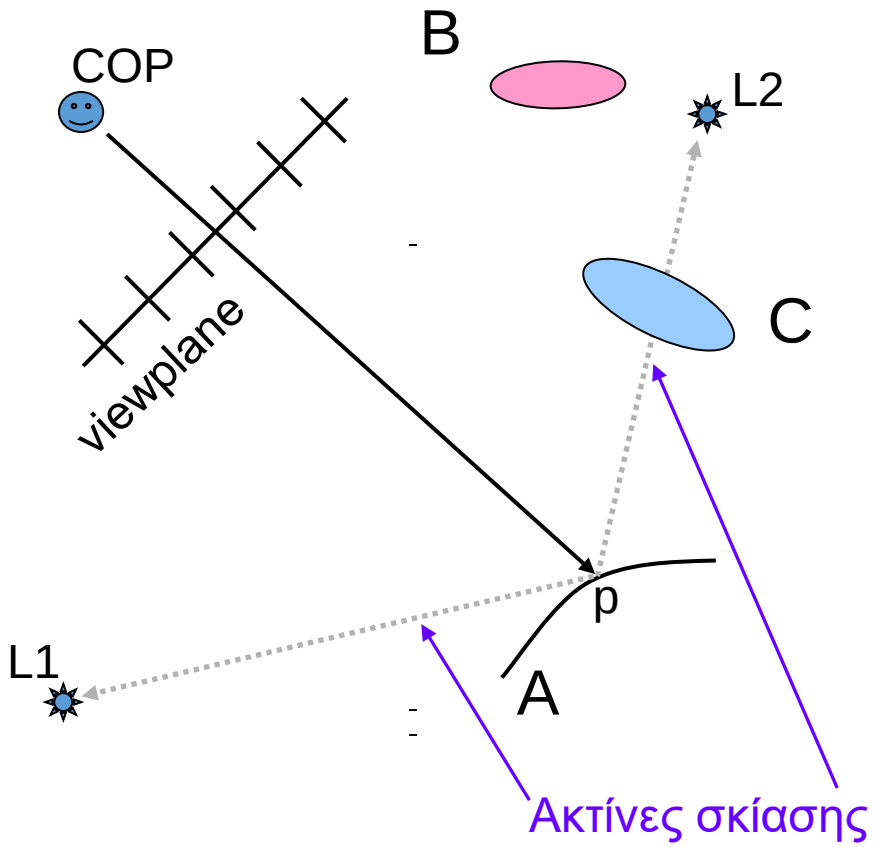
Βολή ακτίνας – με σκιές



Σκιές: *ακτίνες σκίασης (shadow feeler rays)*

- Η διαδρομή από τη διασταύρωση προς την πηγή φωτός είναι γνωστή ως **ακτίνες σκίασης** (shadow ray)
- Εάν αυτή η ακτίνα σκίασης χτυπήσει ένα άλλο αντικείμενο πριν να χτυπήσει μια φωτεινή πηγή, τότε το πρώτο σημείο τομής βρίσκεται στη σκιά του δεύτερου αντικειμένου.
- Εφαρμόζουμε μόνο τον έμμεσο φωτισμό για την εν λόγω πηγή φωτός.
- Εάν η ακτίνα σκίασης φθάσει στην πηγή φωτός χωρίς να χτυπήσει οποιοδήποτε αντικείμενο, υπολογίζουμε τον τοπικό φωτισμό (φωτισμού Phong) και καθορίζουμε το χρώμα του pixel εκεί.
- Κατά συνέπεια, οι σκιές παράγονται φυσικά κατά τη διάρκεια της διαδικασίας απόδοσης.

Σκιές: *ακτίνες σκίασης (shadow feeler rays)*



Διόρθωση για κρυμμένες φωτεινές πηγές

- Πως πρέπει να αλλάξουμε την πιο πάνω εξίσωση;

$$I_r = k_a I_a + \sum_{j=1}^{N} I_j \left(k_d \left(n \cdot l_j \right) + k_s \left(h_j \cdot n \right)^m \right)$$

Διόρθωση για κρυμμένες φωτεινές πηγές

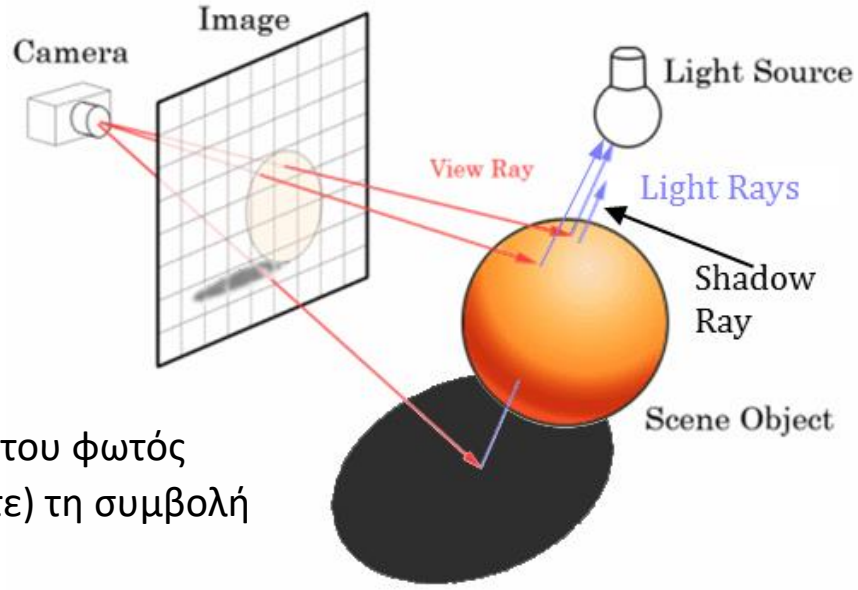
$$I_r = k_a I_a + \sum_{j=1}^{N} S_j I_j \left(k_d \left(n \cdot l_j \right) + k_s \left(h_j \cdot n \right)^m \right)$$

- Όπου S_j είναι το αποτέλεσμα της τομής της ακτίνας σκίασης με τα αντικείμενα (0 ή 1)
- Σημειακές φωτεινές πηγές:
  - $0 < t < 1$
- Κατευθυνόμενες φωτεινές πηγές:
  - $t > 0$
- Προβλήματα ακρίβειας (Precision errors)
  - Βεβαιωθείτε ότι ο δέκτης δεν λαμβάνεται και σαν «occluder»

Σκιές

- Κάθε φως στη σκηνή συμβάλλει στο χρώμα και την ένταση ενός στοιχείου σε μια επιφάνεια…

$$objectIntensity_\lambda = ambient + \sum_{light=1}^{numLights} attenuation \cdot lightIntensity_\lambda \cdot [diffuse + specular]$$



- **Εάν και μόνο αν η φωτεινή πηγή φτάσει στο αντικείμενο**!
  - θα μπορούσε να εμποδίζεται από άλλα αντικείμενα στη σκηνή
  - ή από το ίδιο το αντικείμενο
- Κατασκευάστε μια ακτίνα από την τομή επιφάνειας σε κάθε φως
- Ελέγξτε αν το φως τέμνει κάποιο αντικειμένο
  - αν το πρώτο αντικείμενο που τέμνει είναι το φως, μετρήστε την πλήρη συμβολή του φωτός
  - εάν το πρώτο αντικείμενο που τέμνει δεν είναι το φως, μην υπολογίζετε (αγνοείτε) τη συμβολή του φωτός
  - δημιουργεί σκληρές σκιές. Οι μαλακές σκιές είναι πιο δύσκολο να υπολογιστούν
- Τι γίνεται με διαφανή ή κατοπτρικά (ανακλαστικά) αντικείμενα; Τέτοιες συμβολές φωτισμού είναι η αρχή του παγκόσμιου φωτισμού ⟹ χρειάζονται αναδρομική ανίχνευση ακτίνων

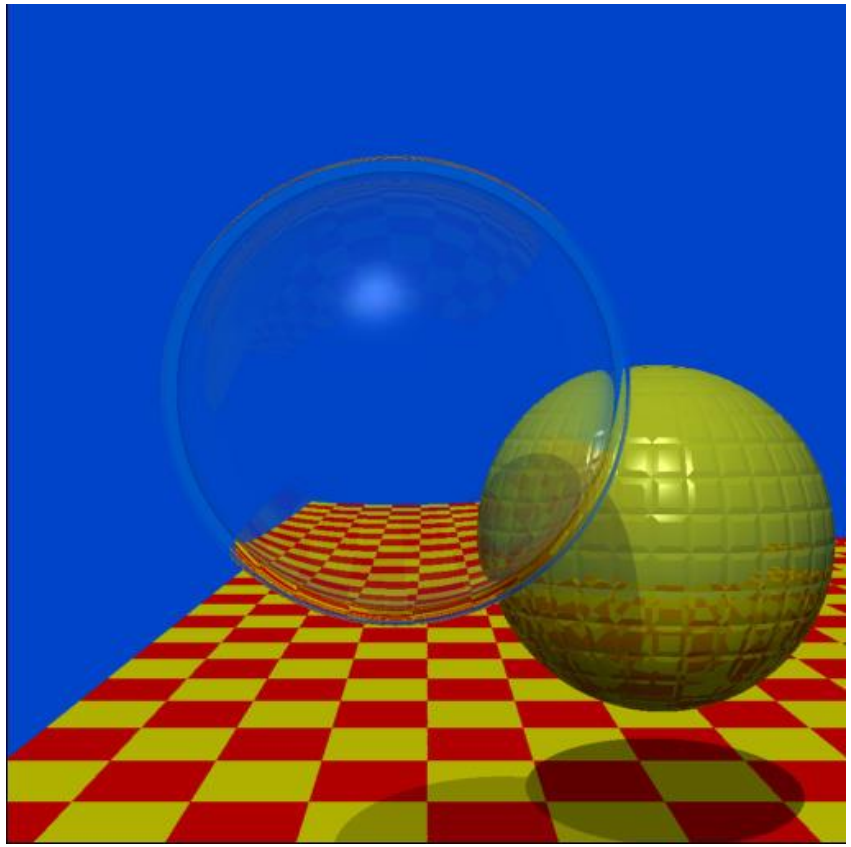
Επαναληπτική παρακολούθηση ακτίνας

- Η νέα εξίσωση φωτισμού είναι (Phong lighting + specular reflection + transmission):
- $$I_\lambda = \underbrace{L_{a\lambda} k_a O_{a\lambda}}_{ambient} + \sum_{lights} f_{att} L_{p\lambda} [\underbrace{k_d O_{d\lambda} (\vec{n} \bullet \vec{l})}_{diffuse} + \underbrace{k_s O_{s\lambda} (\vec{r} \bullet \vec{v})^n}_{specular}] + \underbrace{\underbrace{k_s O_{s\lambda} I_{r\lambda}}_{reflected} + \underbrace{k_t O_{t\lambda} I_{t\lambda}}_{transmitted}}_{recursive\ rays}$$
  - *I* is the total color at a given point (lighting + specular reflection + transmission, λ subscript for each r,g,b)
    - Its presence in the transmitted and reflected terms implies recursion
  - *L* is the light intensity; L_P is the intensity of a point light source
  - *k* is the attenuation coefficient for the object material (ambient, diffuse, specular, etc.)
  - *O* is the object color
  - f_{att} is the attenuation function for distance
  - ***n*** is the normal vector at the object surface
  - ***l*** is the vector to the light
  - ***r*** is the reflected light vector
  - ***v*** is the vector from the eye point (view vector)
  - *n* is the specular exponent
  - note: intensity from recursive rays calculated with the same lighting equation at the intersection point
  - light sources contribute specular and diffuse lighting
- Note: single rays of light do not attenuate with distance considering just angles; purpose of f_{att} is to simulate diminishing intensity per unit area as function of distance for point lights (typically an inverse quadratic polynomial)

Αναδρομική παρακολούθηση ακτίνας (Recursive Ray-tracing)

- Θεωρούμε ότι **όλες** οι επιφάνειες ανακλούν σαν να είναι **ιδανικές ανακλαστικές** (ideal specular).
- Μπορούμε να μιμηθούμε την μετάδοση specular-specular κομψά στέλνοντας δευτερεύουσες ακτίνες από τα σημεία τομής.
- Πρέπει προφανώς να επιλέξουμε ένα βάθος λήξης για να αντιμετωπίσουμε τις πολλαπλάσιες αντανακλάσεις.
- Το ray-tracing πρωτοπαρουσιάστηκε στα Γραφικά Υπ. το 1980 από τον ***Turner Whitted****

Demo: https://youtu.be/0KrCh5qD9Ho

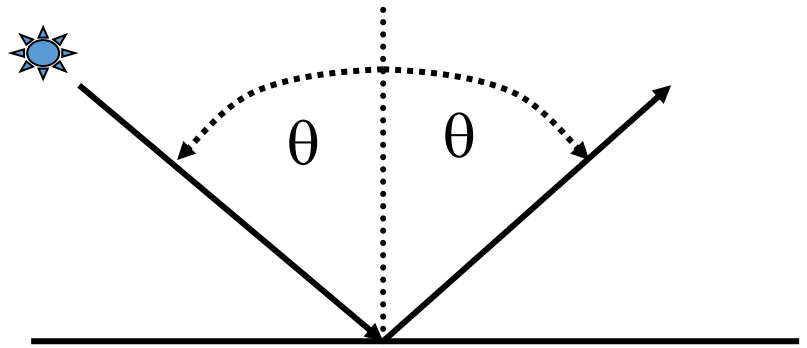


* Whitted, Turner. "An improved illumination model for shaded display." In ACM Siggraph 2005 Courses, p. 4. ACM, 2005.

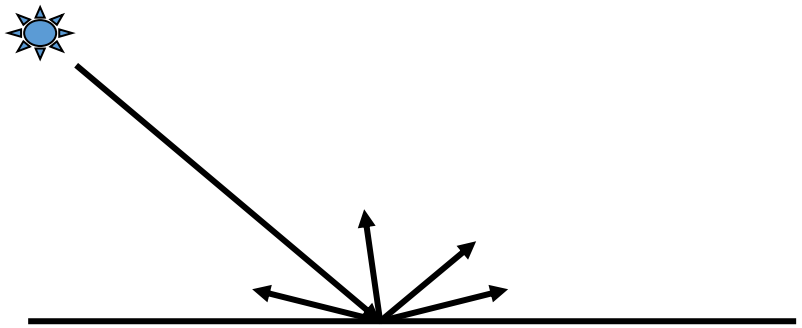
Currently at Nvidia Research: https://blogs.nvidia.com/blog/2018/08/01/ray-tracing-global-illumination-turner-whitted/

Είδη ανακλάσεων

- Κατευθυνόμενη ανάκλαση (Specular reflection)



- Διάχυτη ανάκλαση (Diffuse reflection)



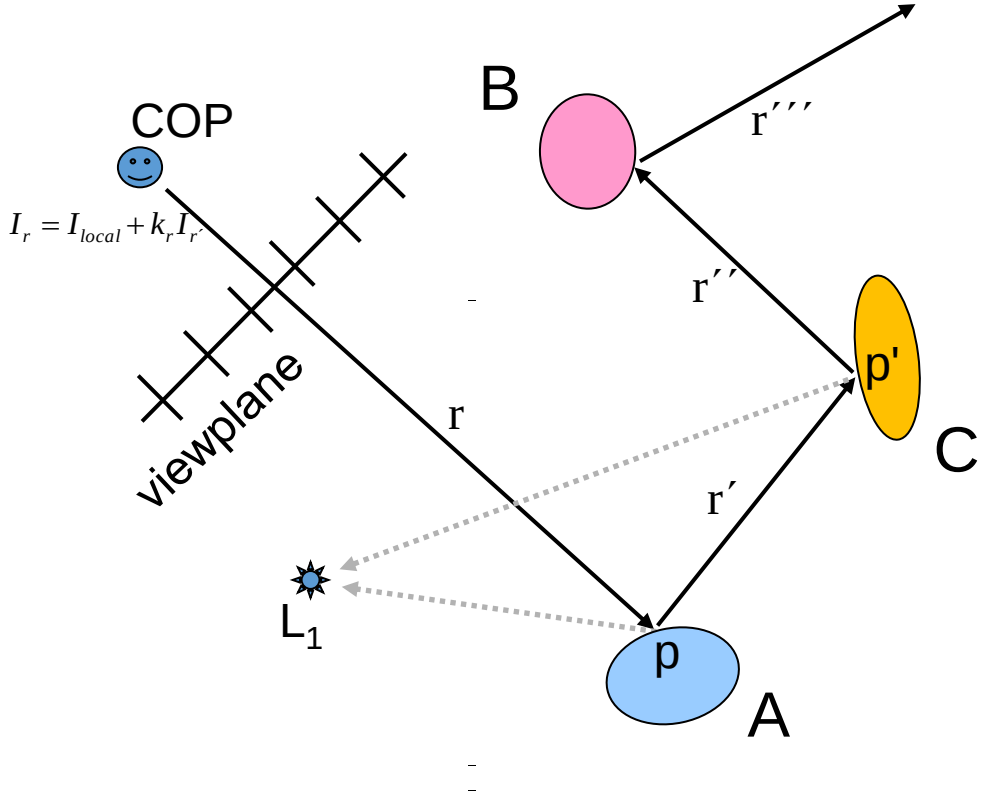
Αναδρομική παρακολούθηση ακτίνας (Recursive Ray-tracing)

- Εάν το αντικείμενο είναι ένα κατοπτρικό αντικείμενο, πρέπει να είμαστε προσεκτικοί για να φτιάξουμε μια ρεαλιστική εικόνα. Αυτό γίνεται με τη λήψη μιας άλλης ακτίνας που ονομάζεται ακτίνα αντανάκλασης προς την κατεύθυνση κατοπτρικής ανάκλασης.
- Όταν η ακτίνα αντανάκλασης χτυπήσει ένα άλλο αντικείμενο, κάνουμε τον ίδιο υπολογισμό όπως αναφέραμε παραπάνω. Ρίχνουμε δηλαδή μια ακτίνα, ελέγχουμε αν βρίσκεται στην περιοχή σκίασης ή όχι. Εάν είναι, χρωματίζουμε με το χρώμα του περιβάλλοντος και, αν όχι, με τον τοπικό φωτισμό. Στη συνέχεια, το χρώμα αυτό επιστρέφει στο προηγούμενο pixel όπου συνέβη η κατοπτρική ανάκλαση, και προστίθεται ως το κατοπτρικό χρώμα.

Υπολογισμός ανάκλασης

$$I_r = I_{local} + k_r I_{r'}$$

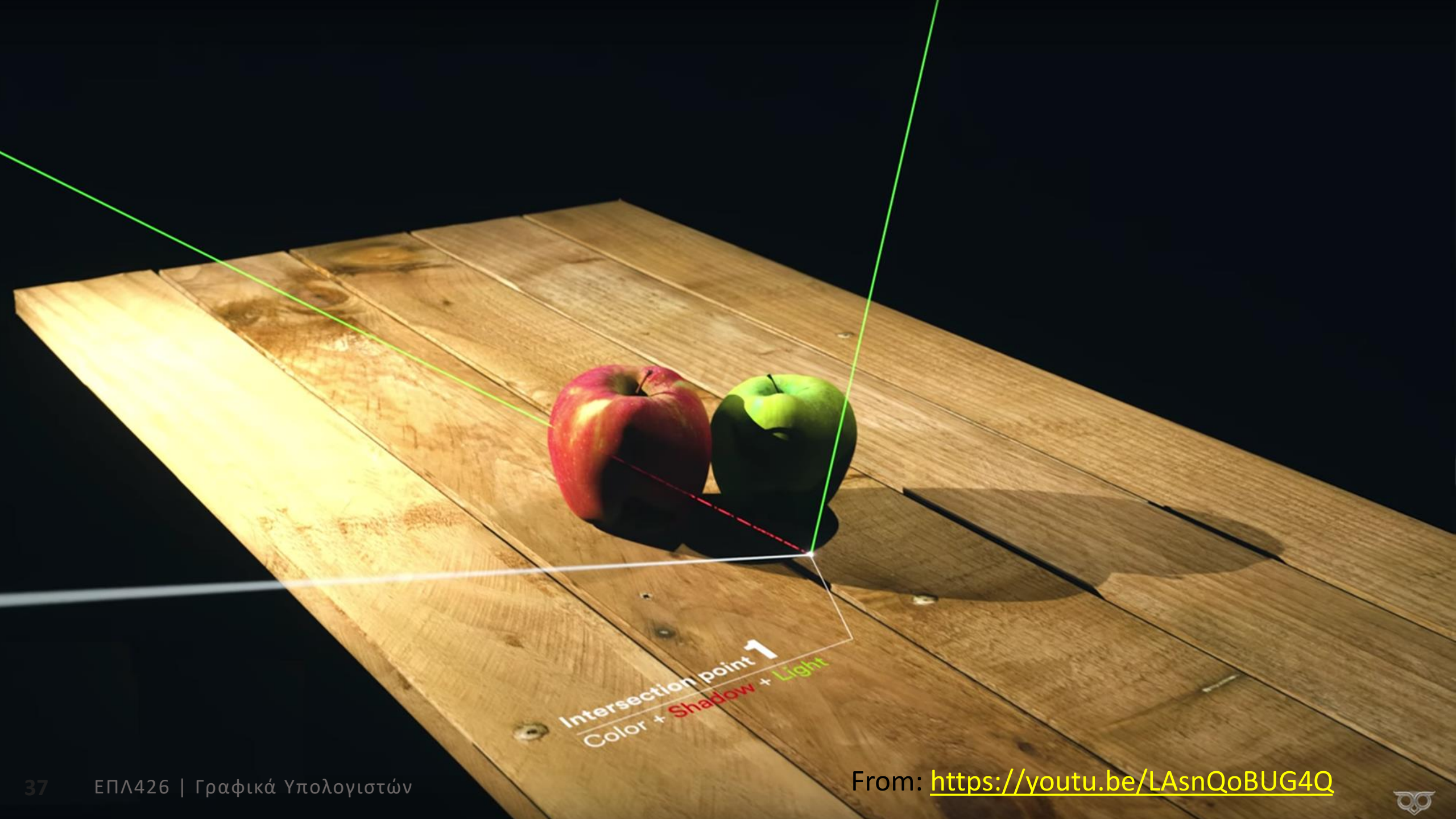
- Όπου το I_{local} υπολογίζεται όπως προηγουμένως
- Η ακτίνα r' is ορίζεται από το σημείο τομής και την ανακλώμενη (R), και προχωρά όπως η πρωτεύουσα ακτίνα
- k_r είναι ο συντελεστής ανάκλασης του p



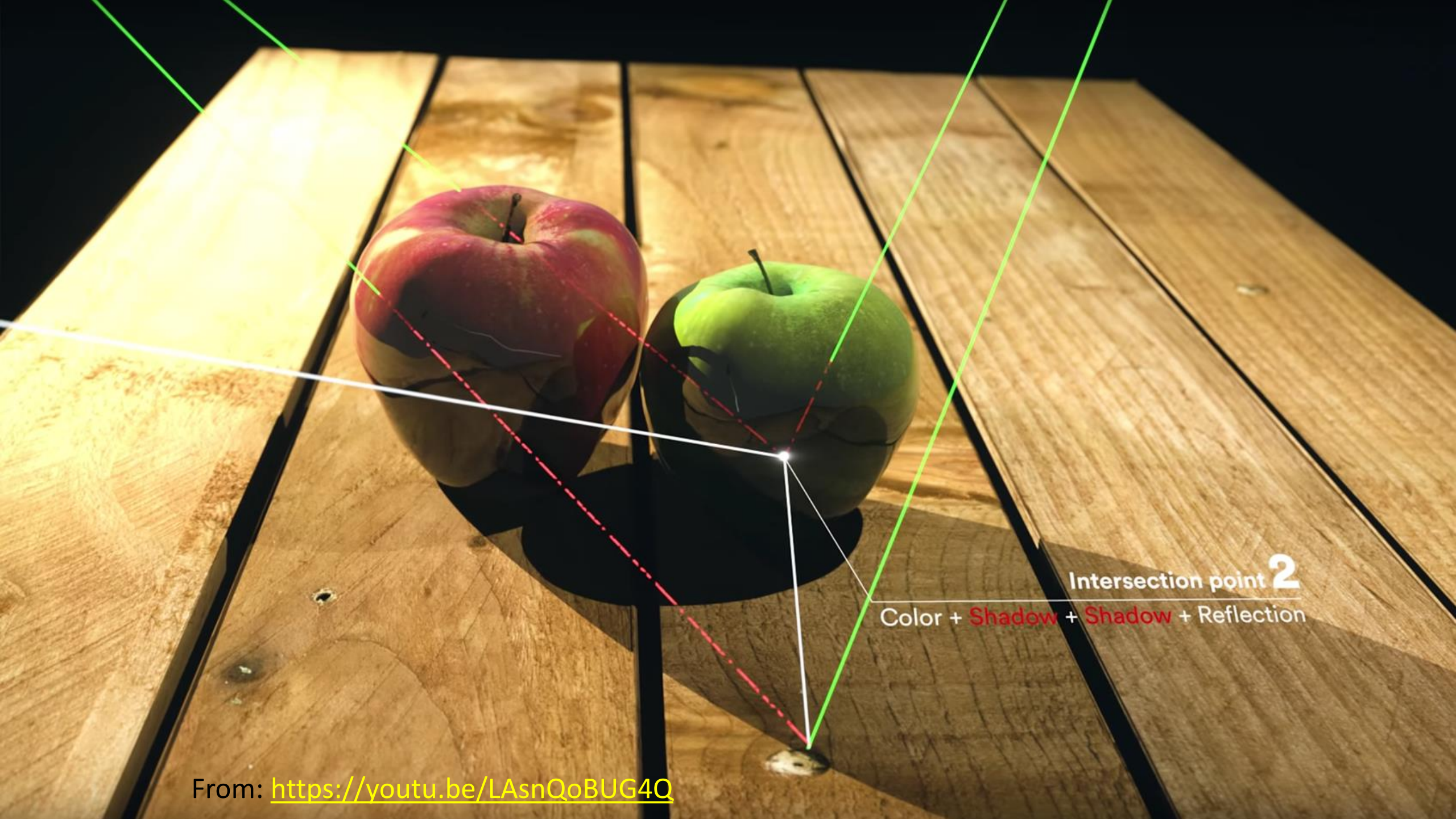
Ψευδοκώδικας

```
/**
 * Recursively trace a ray in the scene.
 * @param p The ray start position.
 * @param D The ray direction.
 * @param depth Current recursion depth.
 * @return Color at the intersection point.Black if we reach
 *         the maximum intersection depth.
 */
Color RayTrace(Point p, Vector D, int depth) {
   Point pd;    /* σημείο τομής */
   Boolean intersection;

   if (depth > MAX) return Color.Black;
   Intersect(p, D, &pd, &intersection);
   if (!intersection) return Background;
   I_local = k_a I_a + Σ_i s (I_i.(k_d(n.l) + k_s.(h.n)^m));

   R = ComputeSecondary(p, D, pd);
   return I_local + k_r * RayTrace(pd, R, depth+1);
}
```

Normally $k_r = k_s$



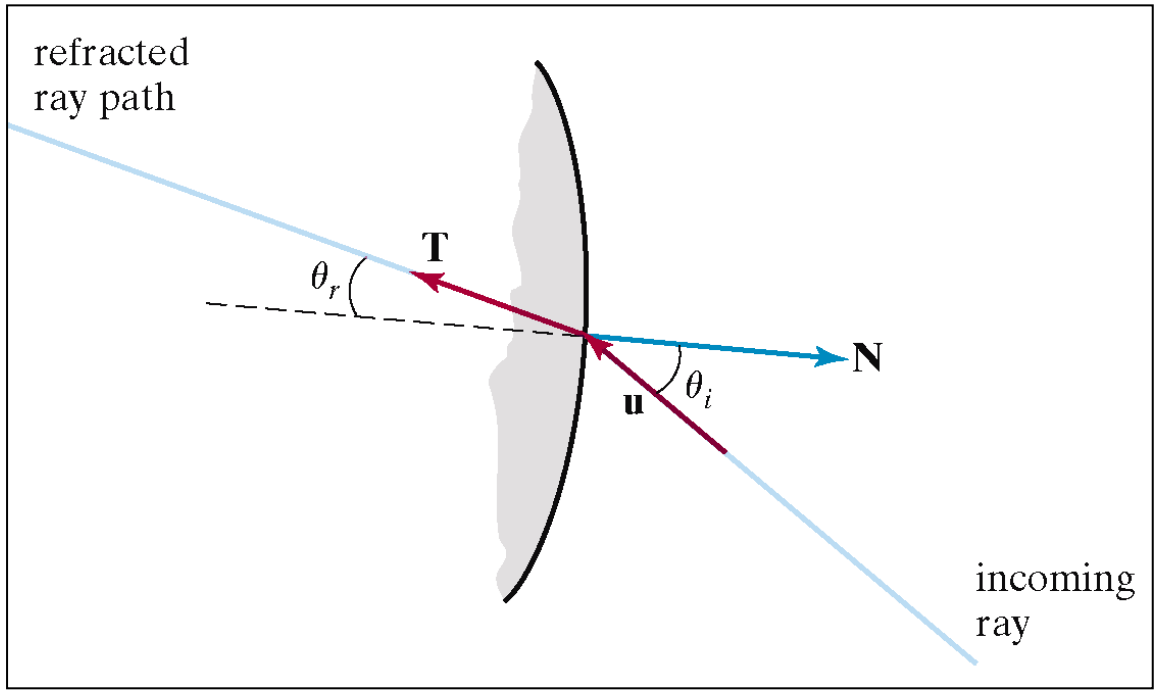
From: https://youtu.be/LAsnQoBUG4Q



From: https://youtu.be/LAsnQoBUG4Q

Ray-Tracing & Διαφανείς επιφάνειες

- Για τις διαφανείς επιφάνειες πρέπει να υπολογίσουμε μια ακτίνα για να αντιπροσωπεύσουμε το φως που διαπερνά από το υλικό
- Η κατεύθυνση της ακτίνας διάθλασης προσδιορίζεται από το δείκτη διάθλασης του υλικού

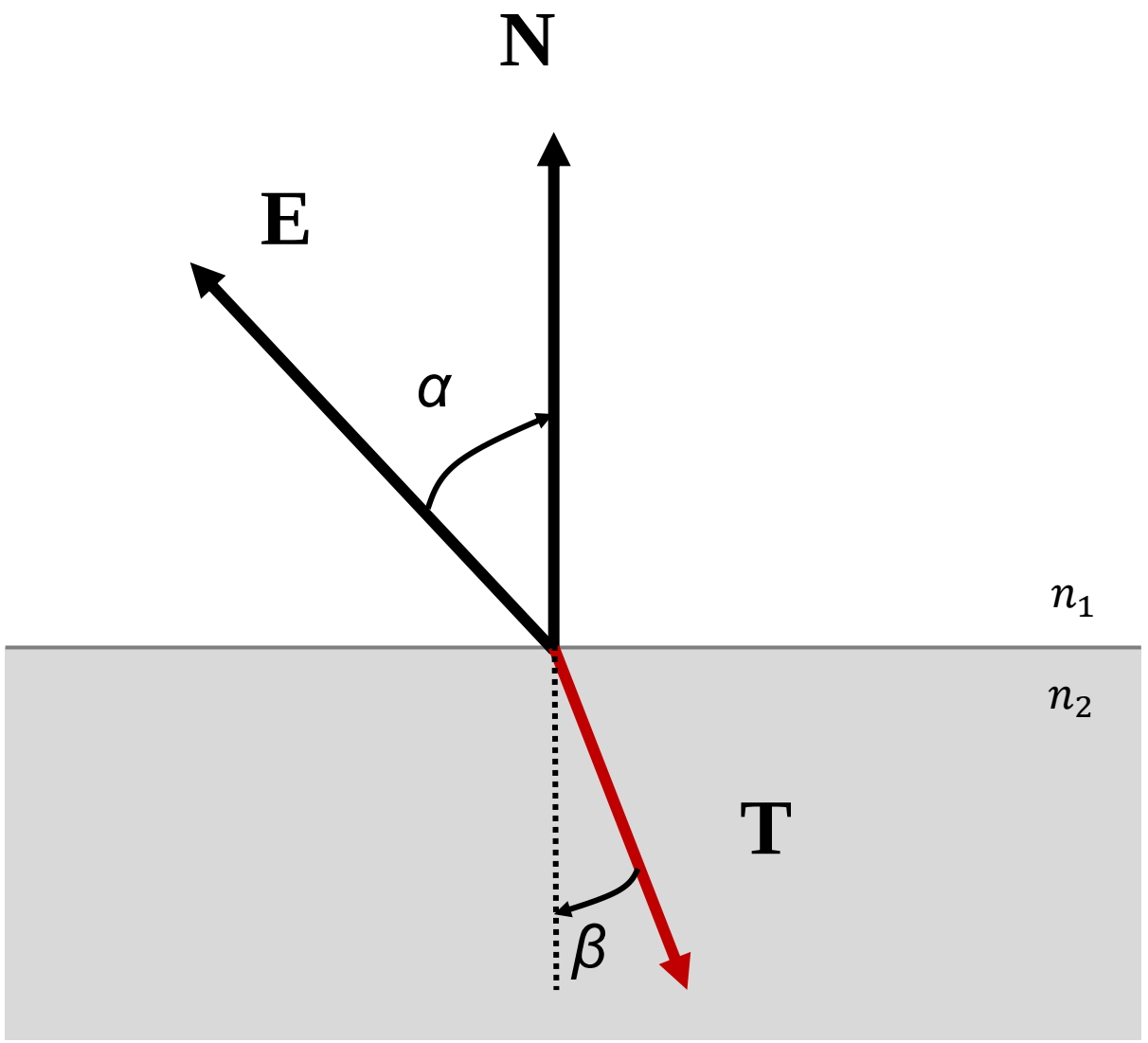


Images taken from Hearn & Baker, "Computer Graphics with OpenGL" (2004)

Refraction Ray

- Αν το αντικείμενο είναι διαφανές, ρίχνουμε μια ακτίνα που ονομάζεται Refraction Ray.
- Αυτή η ακτίνα θα ακολουθήσει τους νόμους του Snell's.
- Όπως με την ανακλώμενη ακτίνα, αν η ακτίνα διάθλασης χτυπά ένα αντικείμενο τότε εφαρμόζεται ένα τοπικό μοντέλο φωτισμού στο σημείο τομής και το αποτέλεσμα μεταφέρεται στο πρώτο σημείο διασταύρωσης.

Διαφάνεια (Perfect Specular Transmission)



Ο νόμος του Snell

$$\frac{\sin\alpha}{\sin\beta} = \frac{\eta_2}{\eta_1}$$

n = δείκτης διάθλασης υλικού

Χρήση του νόμου του Snell

$$\frac{\sin\alpha}{\sin\beta} = \frac{\eta_2}{\eta_1} = \eta_{21}$$

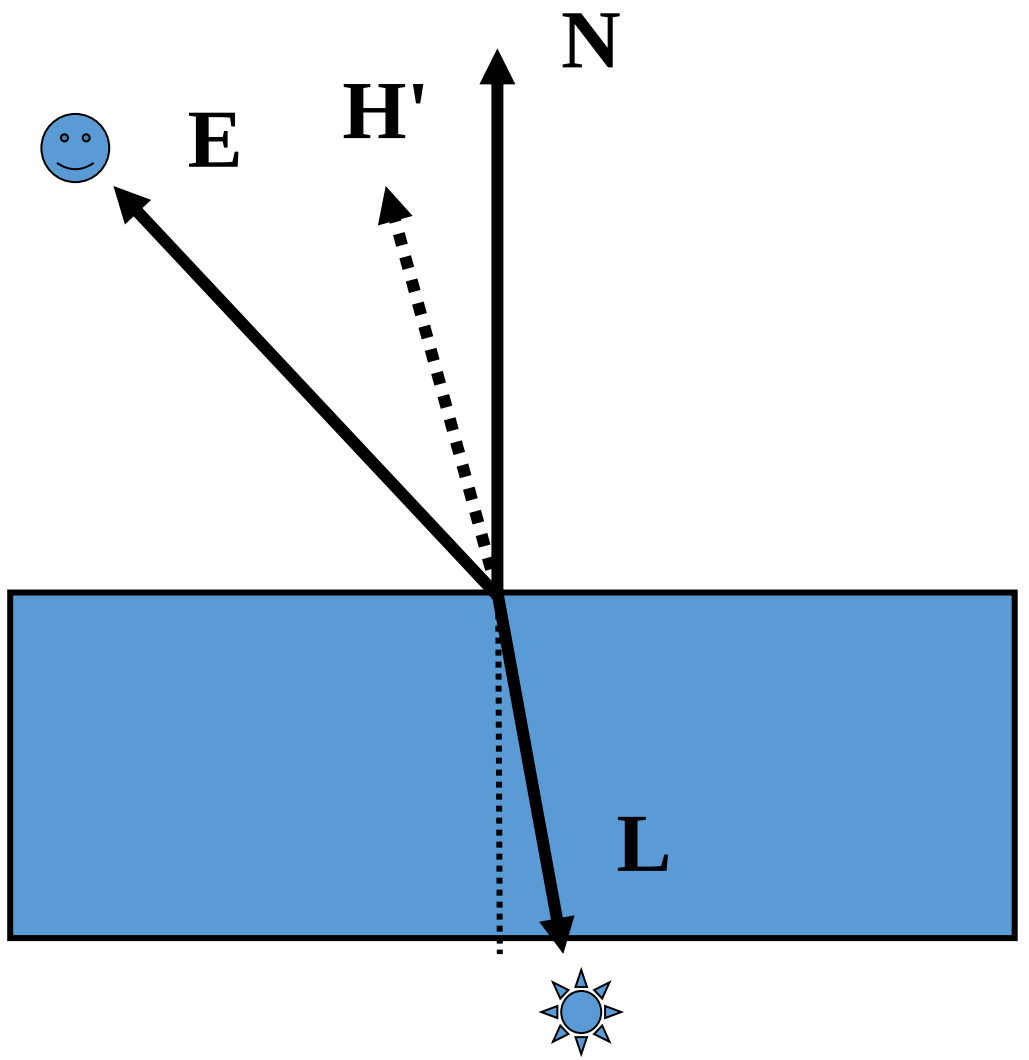
- Βάση αυτού του νόμου μπορούμε να δείξουμε:

$$T = -\eta_{12}E + N\left(\eta_{12}\cdot\cos\alpha - \sqrt{1+\eta_{12}^2\cdot\left(\cos^2\alpha - 1\right)}\right)$$

- Προσέξτε ότι όταν η ρίζα είναι αρνητική τότε έχουμε πλήρη εσωτερική ανάκλαση και χρησιμοποιούμε απλά την ανακλώμενη ακτίνα (δεν έχουμε διάθλαση)

Κατευθυνόμενη μετάδοση (direct specular transmission)

- Μια διαφανής επιφάνεια μπορεί να φωτίζεται και από πίσω, αν υπάρχει κάποια φωτεινή πηγή.
- Αυτό λαμβάνεται υπόψιν στο $I_{local'}$



Υπολογισμός του H'

- Γίνεται χρήση του H‘ αντί του H στο specular μέρος

$$H' = \frac{E - \frac{\eta_2}{\eta_1} L}{\frac{\eta_2}{\eta_1} - 1}$$

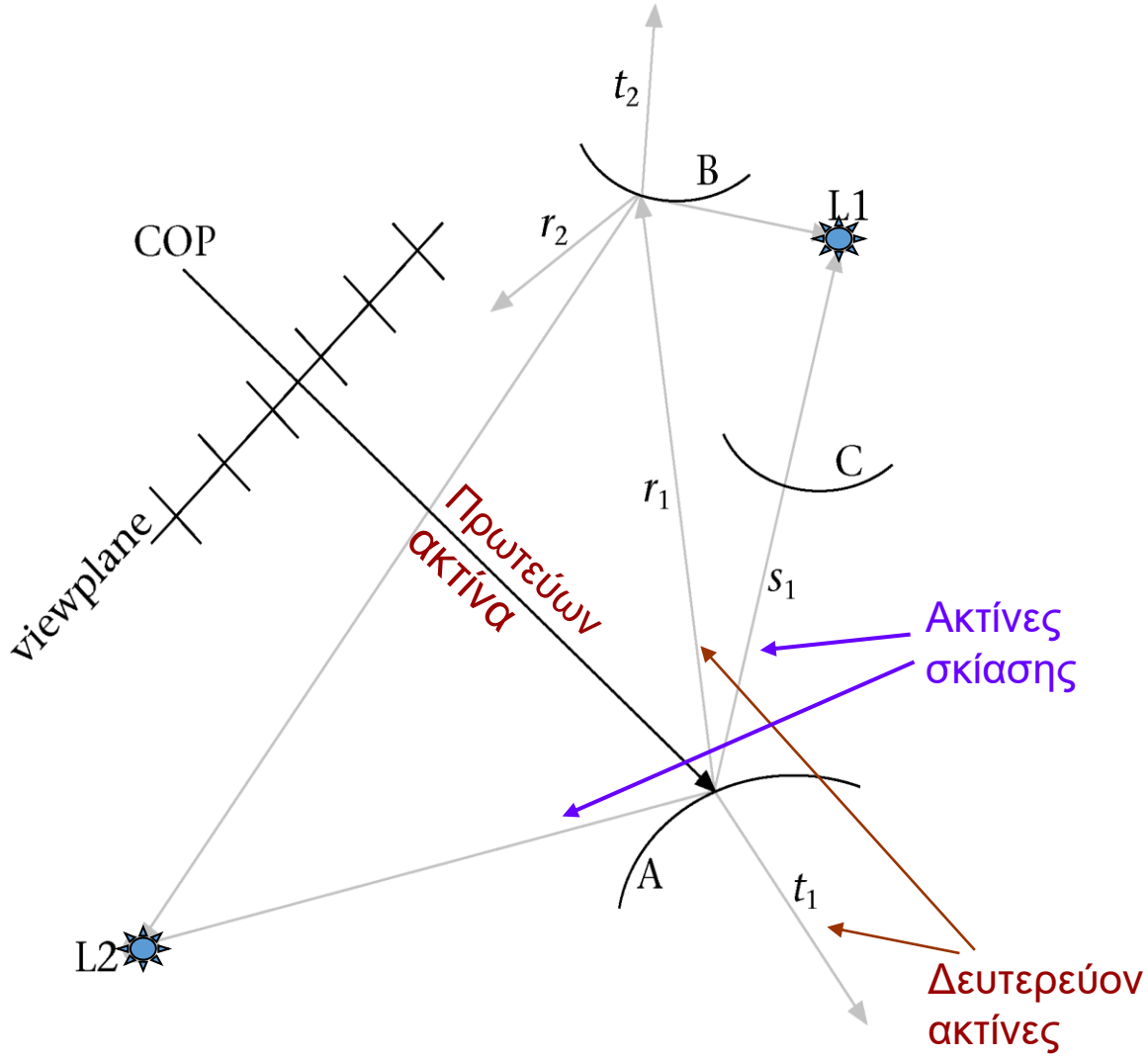
$$I_r = k_a I_a + \sum_{j=1}^{N} I_j \left(k_d \left(n \cdot l_j \right) + k_s \left(h_j \cdot n \right)^m \right)$$

Rays to be considered

- Shadow ray
- Reflection ray
- Refraction ray

Sum the color based on the three rays

$$I_r = I_{local} + k_r I_{r'} + k_t I_{t'}$$



Νέος Ψευδοκώδικας

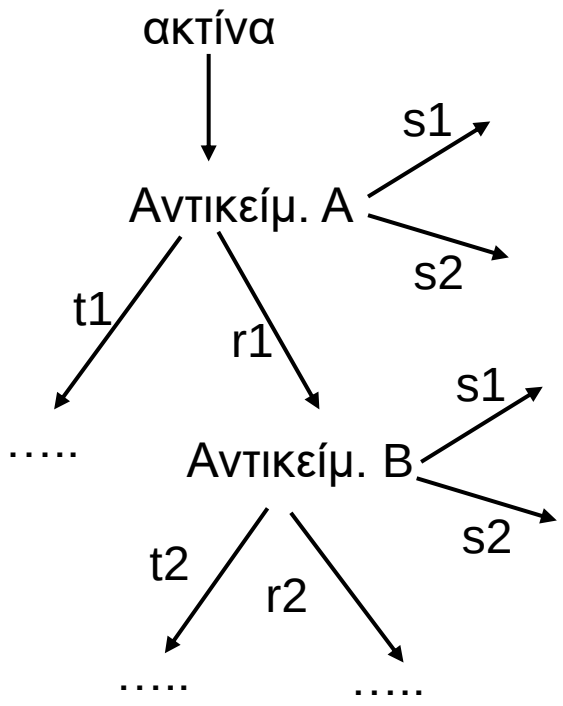
```
Color RayTrace(Point p, Vector D, int depth) {
   Point pd;    /* Intersection point */
   Boolean intersection;

   if (depth > MAX) return Black;
   Intersect(p, direction, &pd, &intersection);
   if (!intersection) return Background;
   I_local = k_a I_a + Σ_i (I_i.s.(k_d(n.l) + k_s.(h.n)^m));

   (R, T) = ComputeSecondary(p, D, pd);
   return I_local + k_r*RayTrace(pd, R, depth+1) + k_t*RayTrace(pd, T, depth+1);
}
```

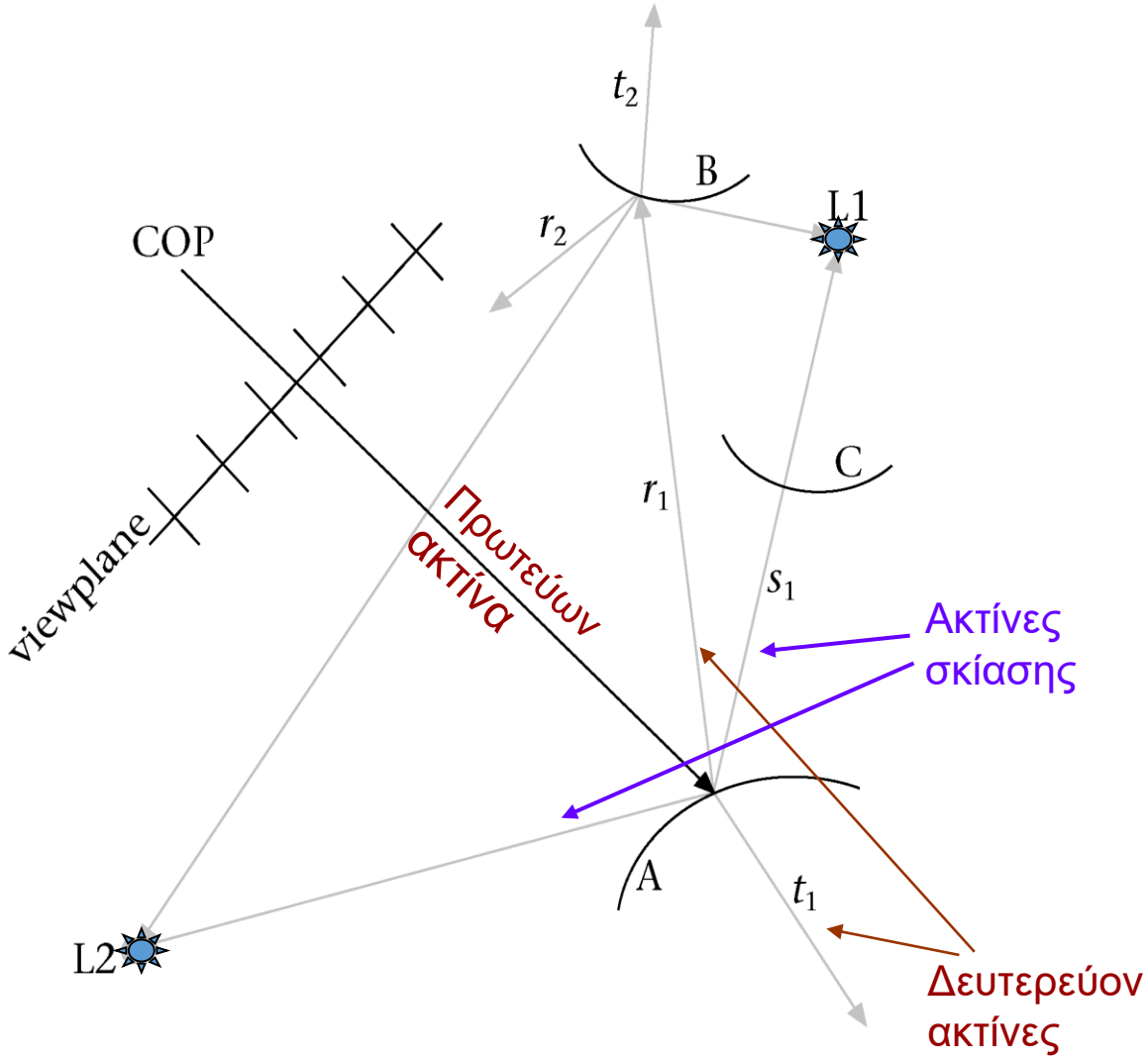
Recursive ray-tracing

- Για την αντανάκλαση και τη διάθλαση, εκτελούμε αναδρομικά τον αντίστοιχο υπολογισμό με τον ίδιο τρόπο. Οι ακτίνες ελέγχονται εάν τέμνονται με τα άλλα αντικείμενα στο περιβάλλον, ρίχνουν τις ακτίνες σκιάσεις, την αντανάκλαση και τις διαθλαστικές ακτίνες και στη συνέχεια υπολογίζουν το χρώμα.
- Ο αριθμός των ακτίνων πρόκειται να αυξηθεί εκθετικά εάν υπάρχουν τόσο αντανακλόμενα και διαθλόμενα αντικείμενα στη σκηνή. Ως εκ τούτου, μπορούμε να ονομάσουμε αυτό το δέντρο ακτίνων ως **ray tree**.

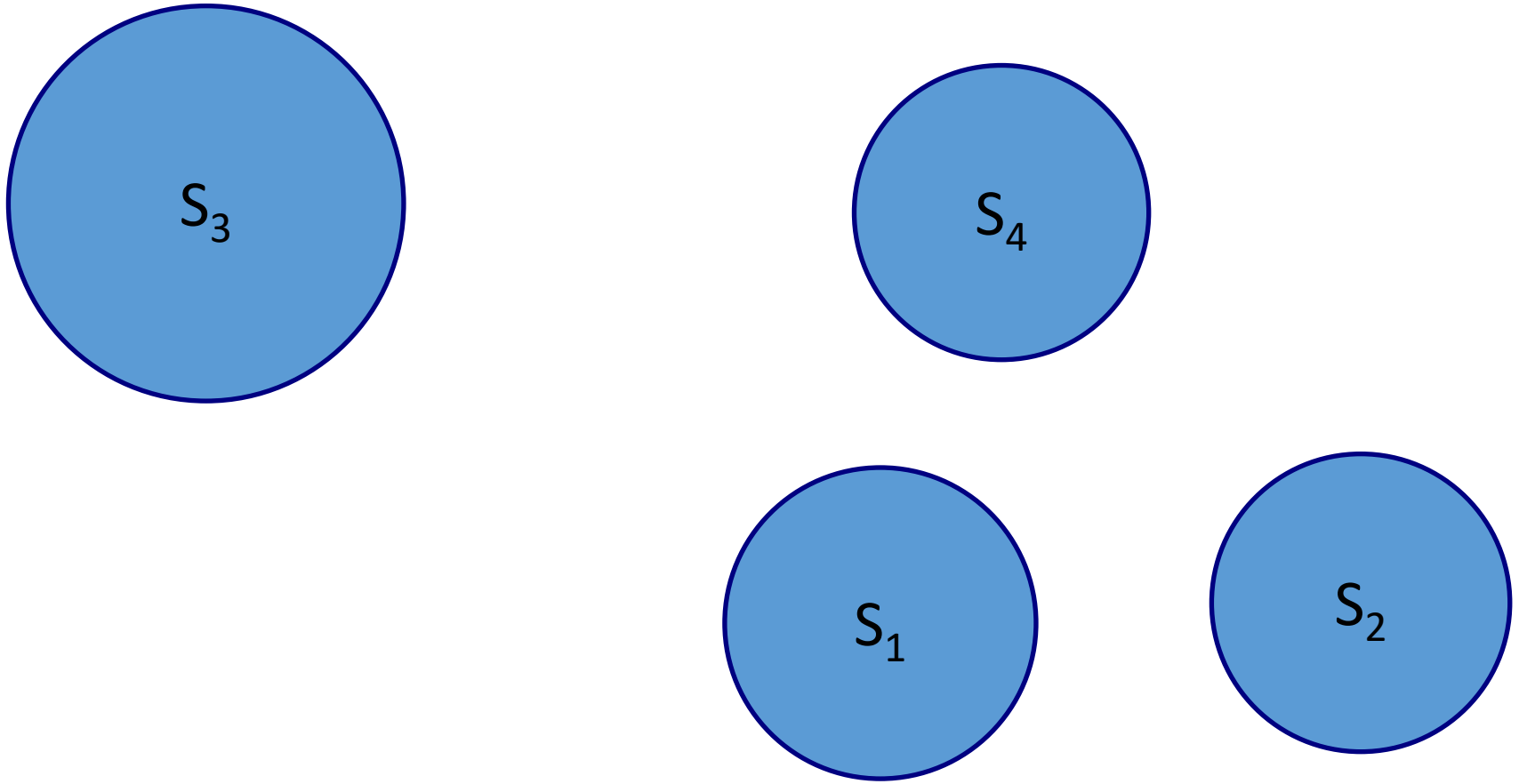
Ray-Tracing Tree

Δυαδικό δέντρο ακτίνας

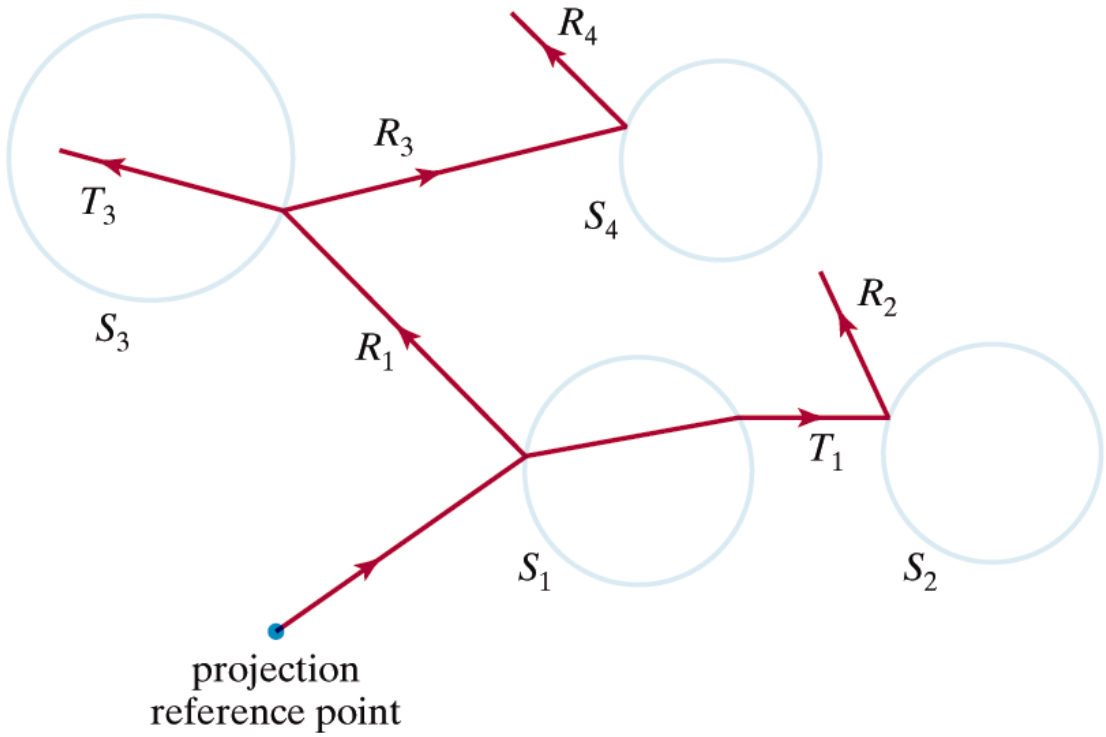


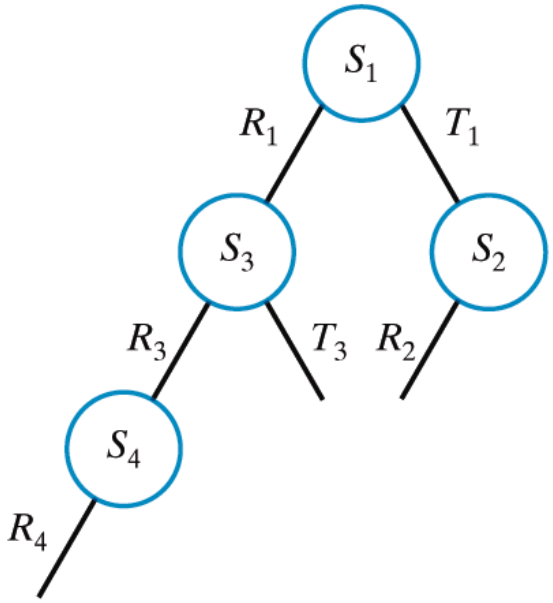


Ray-Tracing Tree Example



Ray-Tracing Tree Example





Images taken from Hearn & Baker, "Computer Graphics with OpenGL" (2004)

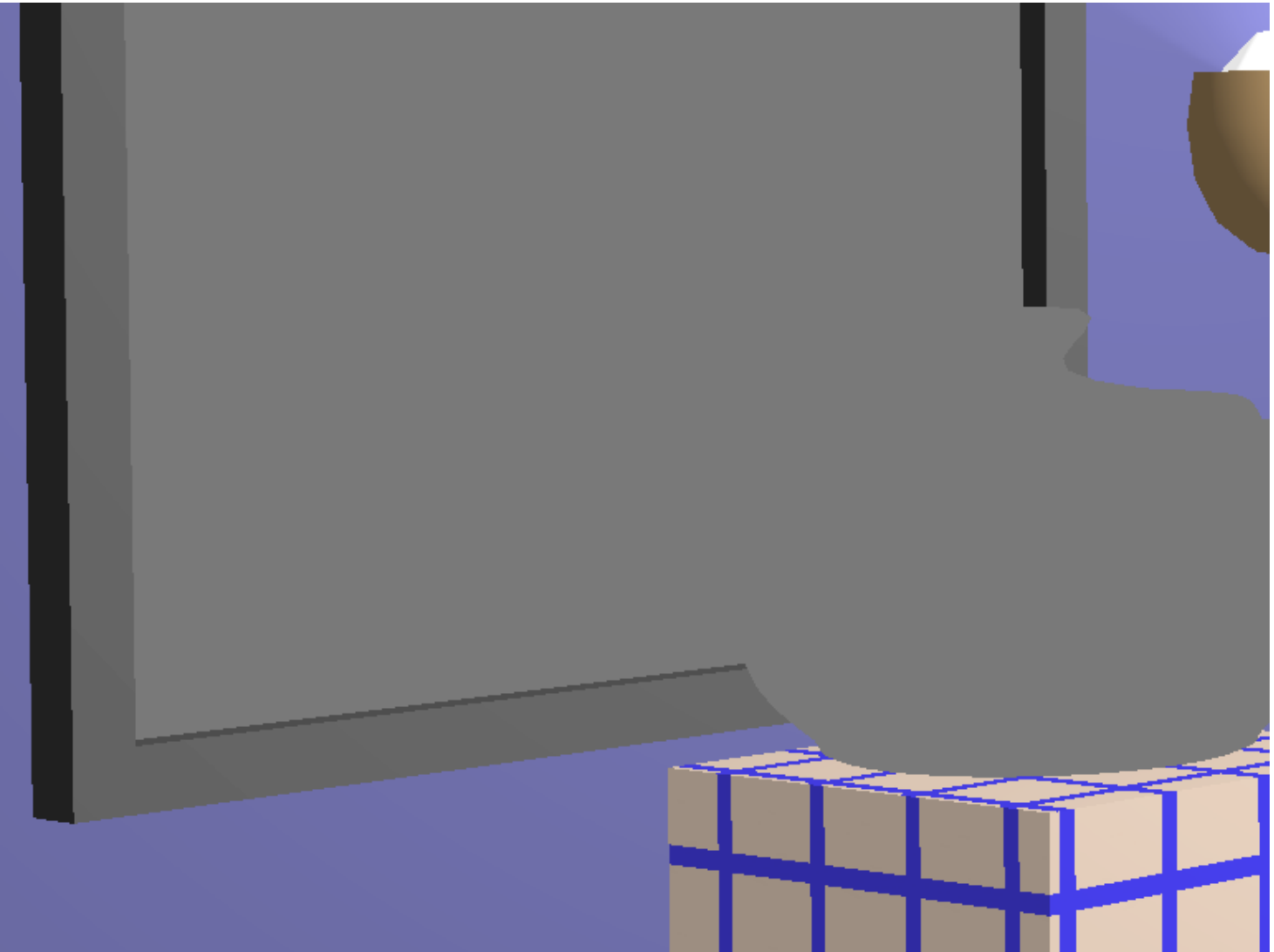
Ray-Tracing Tree

- Μετά την ολοκλήρωση του δέντρου εντοπισμού ακτίνων για ένα pixel, μαζεύουμε τις συνεισφορές κάθε έντασης
- Αρχίζουμε με τους τερματικούς κόμβους (κάτω) του δέντρου
- Η ένταση της επιφάνειας σε κάθε κόμβο εξασθενεί από την απόσταση από την γονική επιφάνεια και προστίθεται στην ένταση της γονικής επιφάνειας
- Το άθροισμα των εξασθενημένων εντάσεων στον κόμβο ρίζας αντιστοιχεί στο pixel

Test scene – ray traced

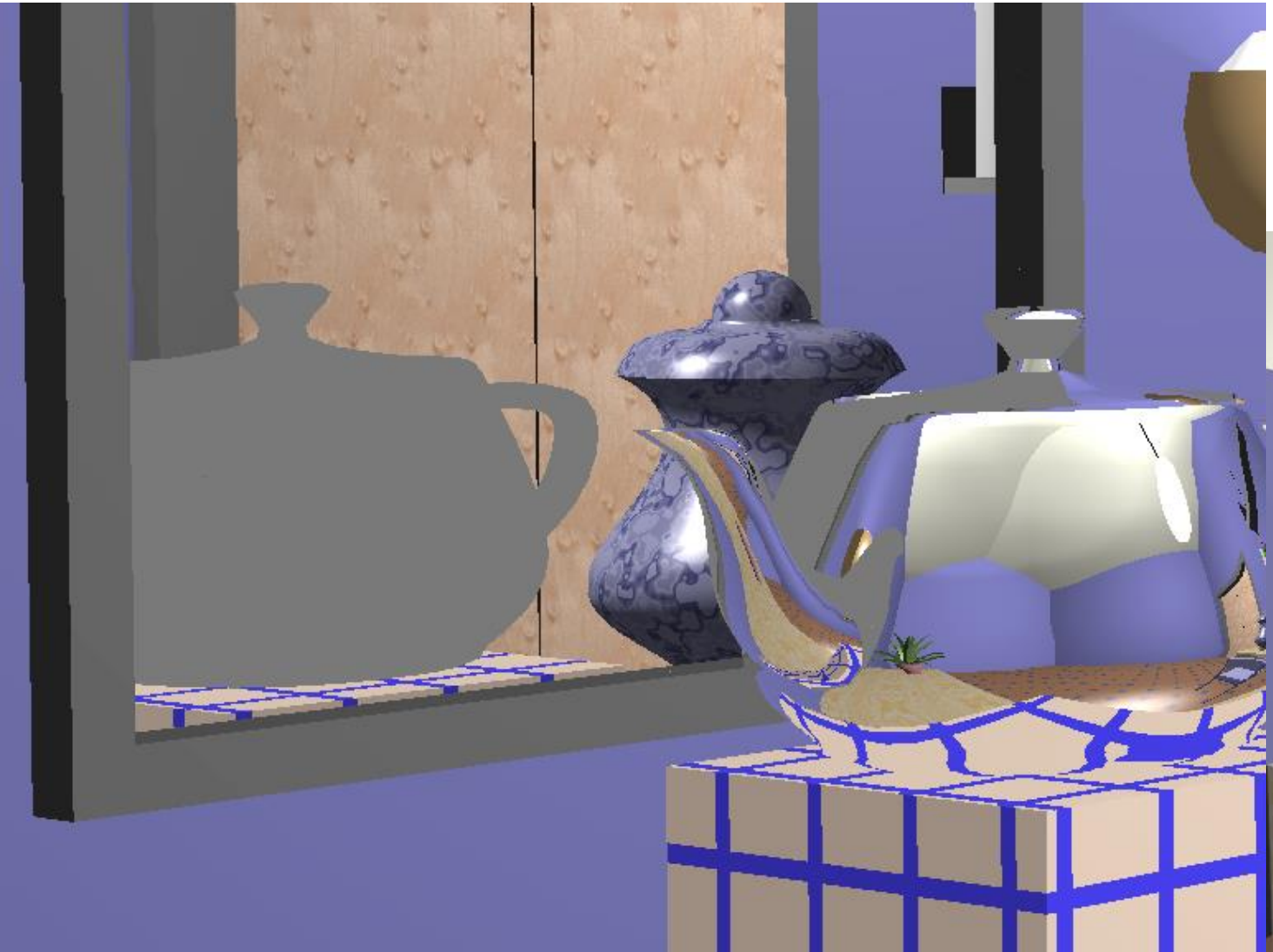
Επιλογή βάθους - βάθος 1

$$I_r = I_{local} + k_r I_{r'} \qquad I_{local} = k_a I_a + \sum_{j=1}^{N} S_j I_j \left(k_d \left(n \cdot l_j \right) + k_s \left(h_j \cdot n \right)^m \right)$$



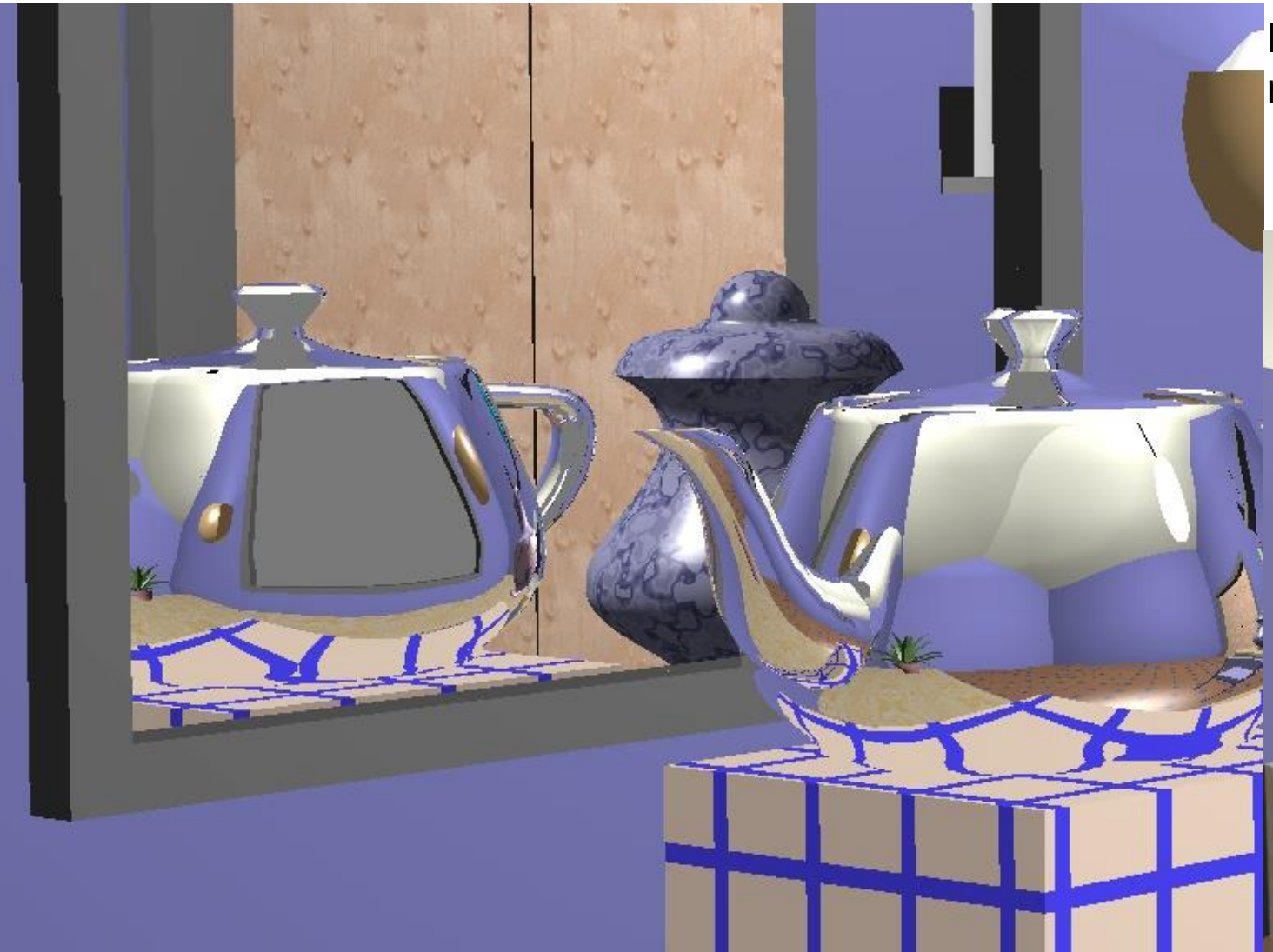
Note only ambient shade on mirror and teapot

Επιλογή βάθους - βάθος 2



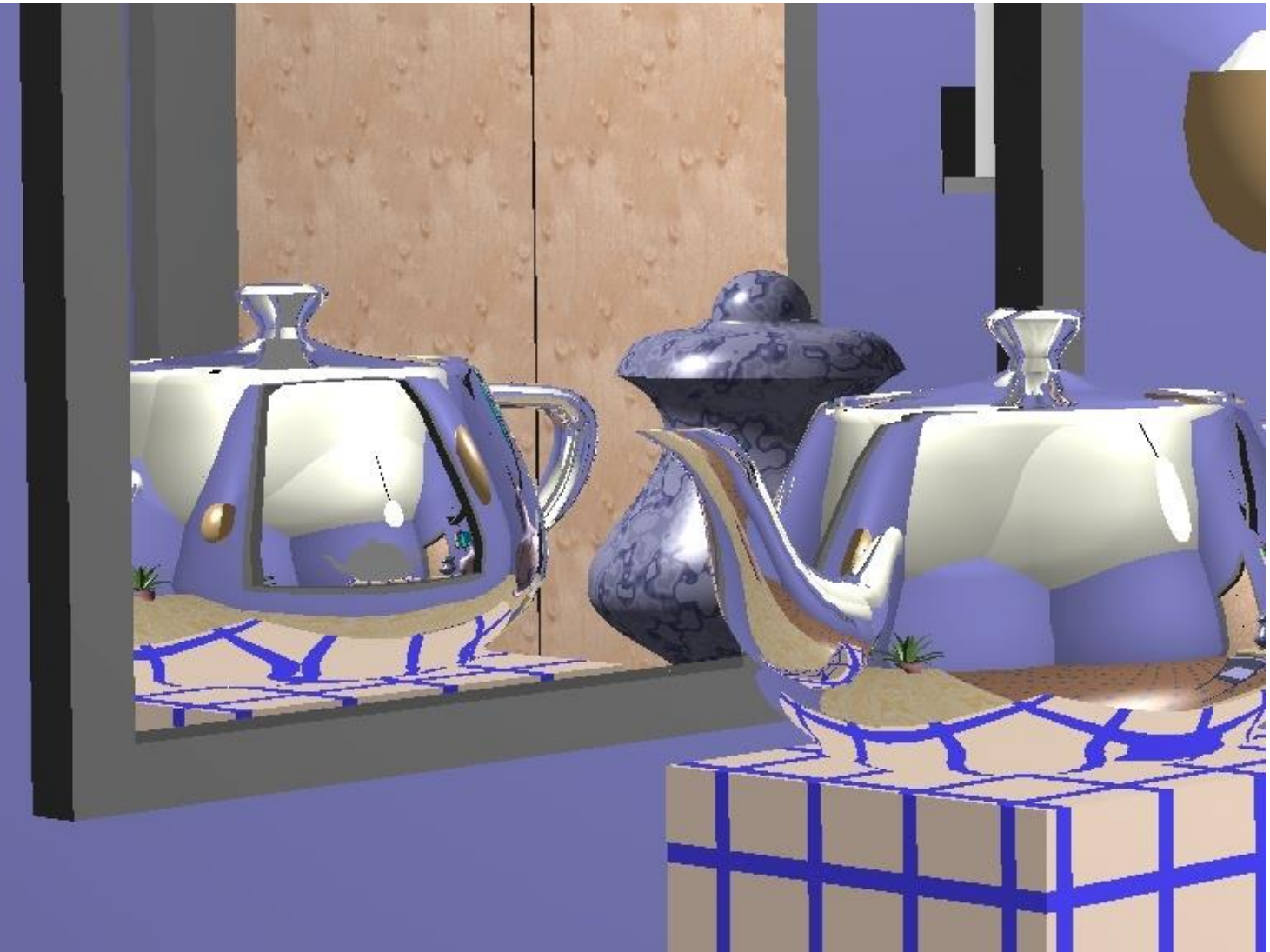
Note only ambient shade on reflection of mirror and teapot.

Επιλογή βάθους - βάθος 3



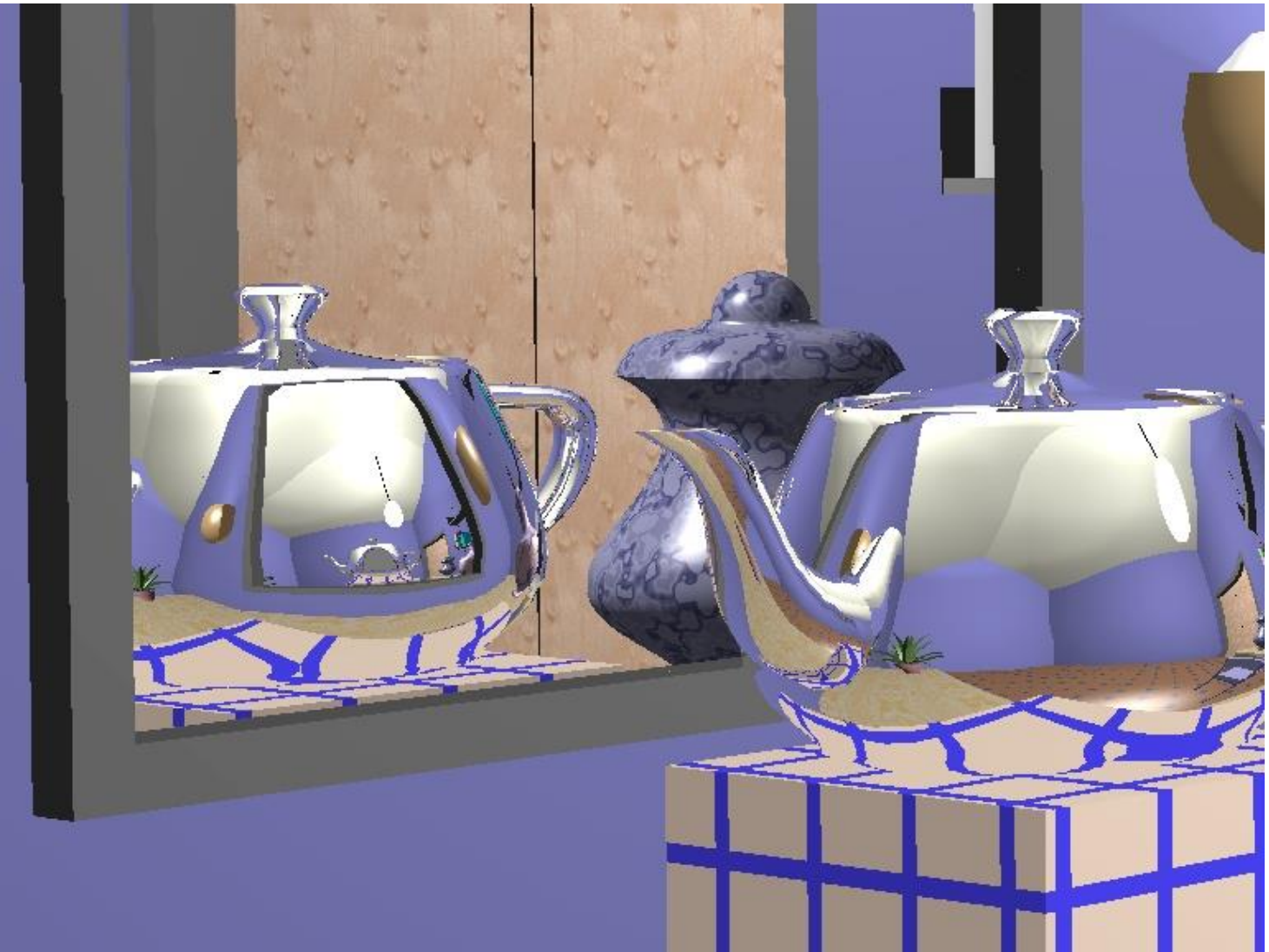
Note only ambient shade on reflection of mirror in teapot.

Επιλογή βάθους - βάθος 4

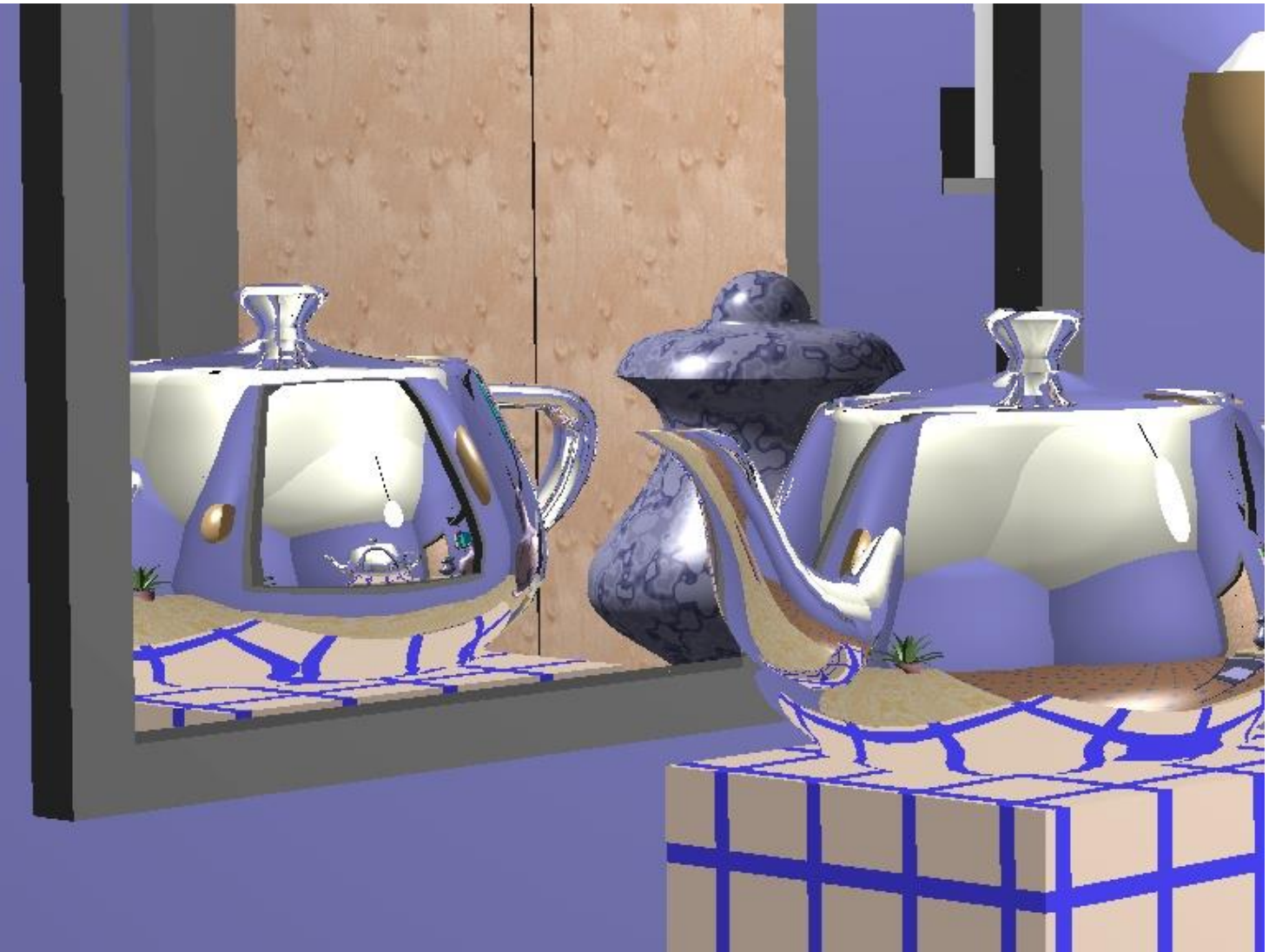


Note ambient shade on reflection of teapot in reflection of mirror in teapot.

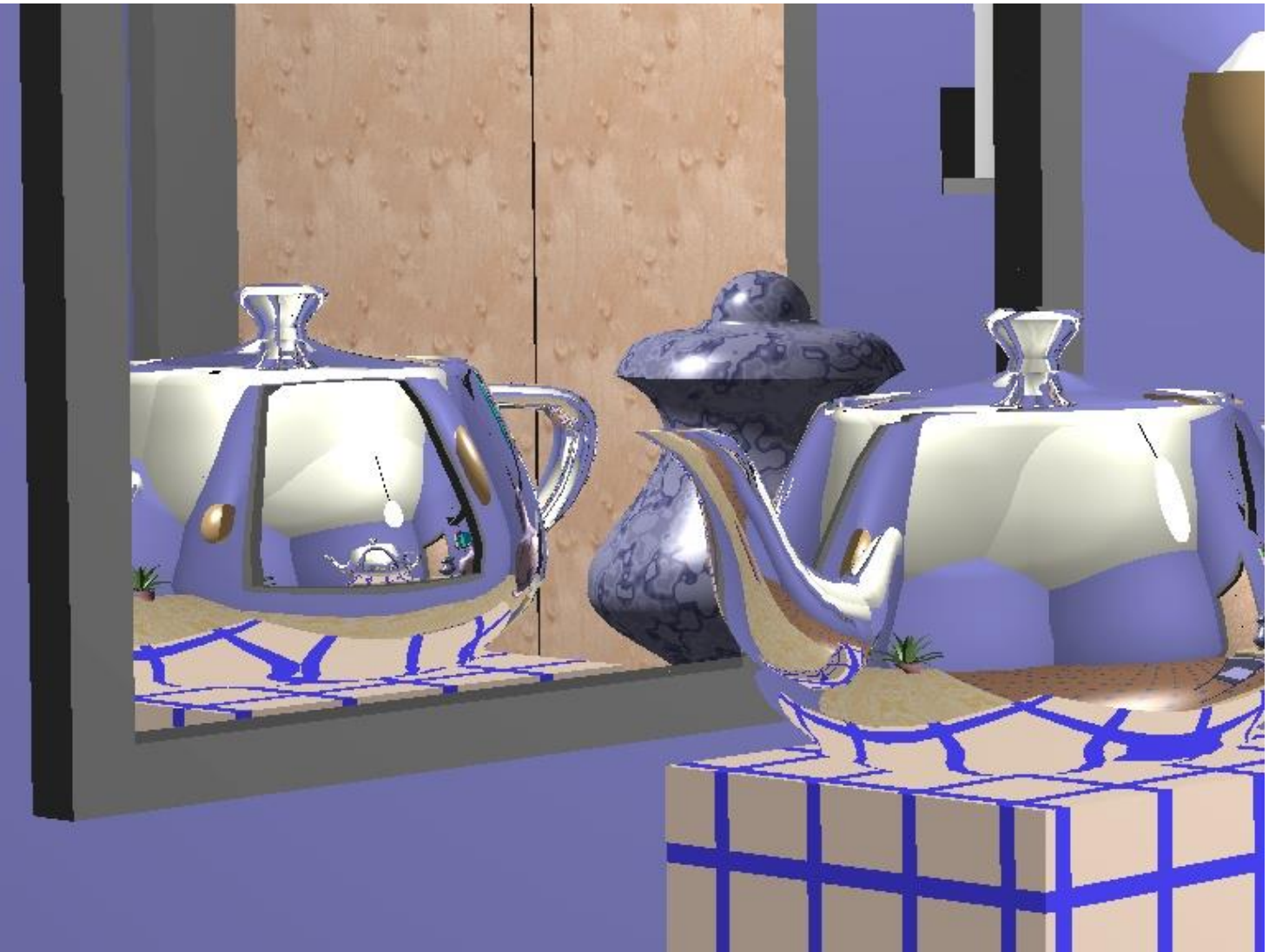
Επιλογή βάθους - βάθος 5



Επιλογή βάθους - βάθος 6



Επιλογή βάθους - βάθος 7



Color of Specular Surfaces

- Το χρώμα υπολογίζεται ανά ακτίνα

$$I = I_{local} + K_r R + K_t T$$
$$R = I'_{local} + K'_r R' + K'_t T'$$
$$R' = I''_{local} + K''_r R'' + K''_t T''$$
$$\vdots$$

- Αντικαθιστώντας τα όλα σε μια εξίσωση, παίρνουμε

$$I = I_{local} + K_r (I'_{local} + K'_r (I''_{local} + K''_r (I'''_{local} + K'''_r (\ldots))))$$

Τερματισμός αναδρομής

- Όταν η ακτίνα βγει εκτός της σκηνής
  - παίρνει το χρώμα του background
- Όταν φτάσουμε στο προκαθορισμένο βάθος αναδρομής (αναπήδησε D_{max} φορές)
- Όταν κτυπήσει σε επιφάνεια διάχυτης ανάκλασης
- Όταν η προσφορά της ακτίνας γίνει αμελητέα (Adaptive depth control)
  - Όταν το γινόμενο των k_rs που συναντά στο μονοπάτι της η ακτίνα πέσει κάτω από μια προκαθορισμένη τιμή

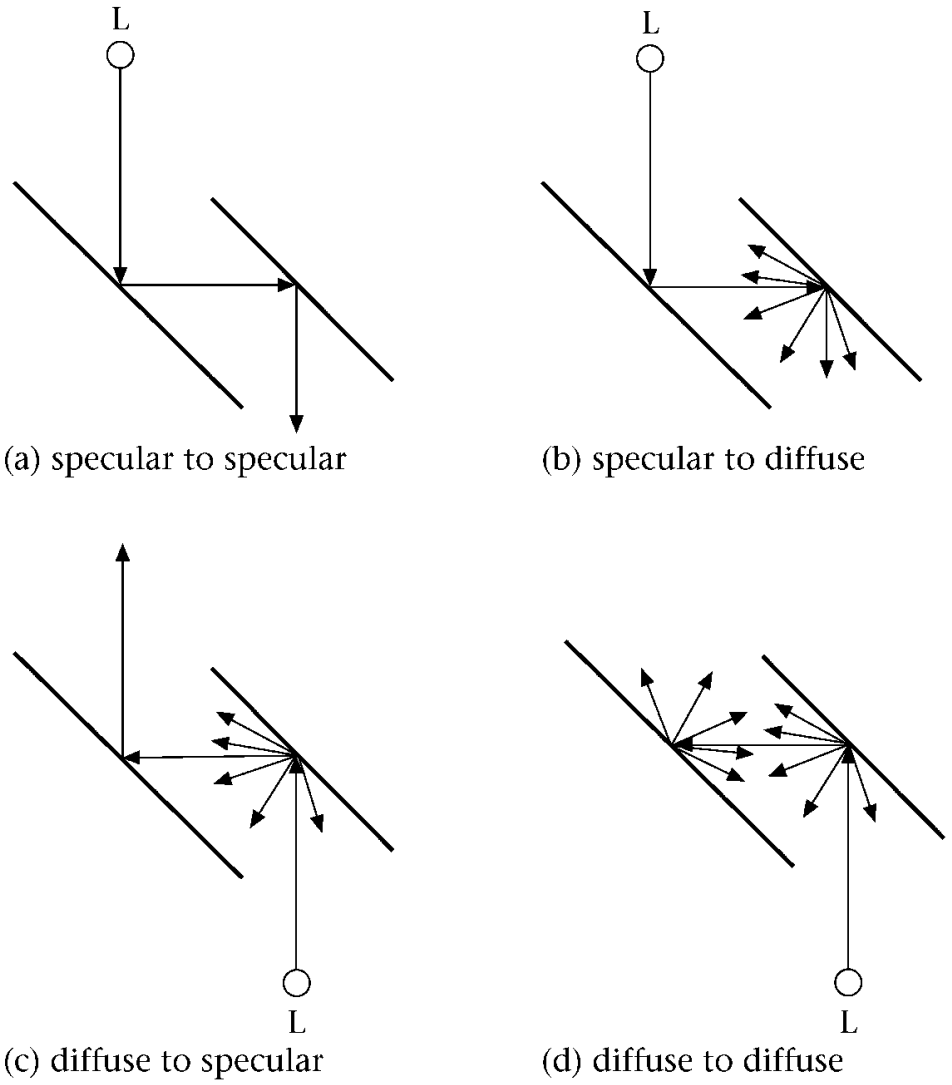
$$I = I_{local} + K_r(I'_{local} + K'_r(I''_{local} + K''_r(I'''_{local} + K'''_r(\ldots))))$$

$$K_r K'_r K''_r K'''_r \ldots < threshold$$

Παραδείγματα

Παραδείγματα

Τι δεν μπορούμε να προσομοιώσουμε;



(a) specular to specular

(b) specular to diffuse

(c) diffuse to specular

(d) diffuse to diffuse

Τομή ακτίνας-αντικειμένου

Έμμεσα αντικείμενα

- Εάν ένα αντικείμενο ορίζεται από μια συνάρτηση *f* , όπου *f(Q)* = 0 όταν το *Q* είναι ένα σημείο στην επιφάνεια του αντικειμένου, τότε η τομή ακτίνας αντικειμένου είναι σχετικά εύκολο να υπολογιστεί
  - πολλά αντικείμενα μπορούν να οριστούν σαν συνάρτηση
- Για παράδειγμα, ένας κύκλος με ακτίνα *R* είναι ένα έμμεσο αντικείμενο σε ένα επίπεδο, με εξίσωση:

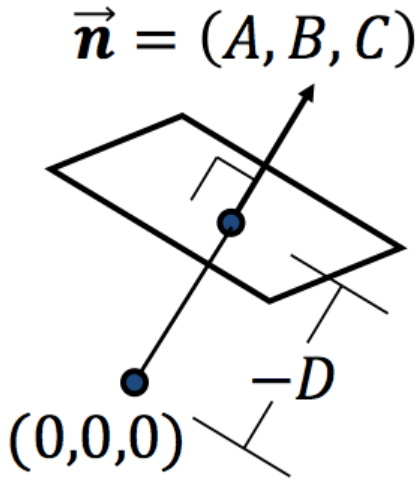
$$f(x,y) = x^2 + y^2 - R^2$$

  - *το σημείο (x,y) είναι πάνω στον κύκλο όταν f(x,y) = 0*
- Ένα άπειρο επίπεδο ορίζεται από τη συνάρτηση:

$$f(x,y,z) = Ax + By + Cz + D$$

- Μια σφαίρα με ακτίνα *R* στις 3Δ:

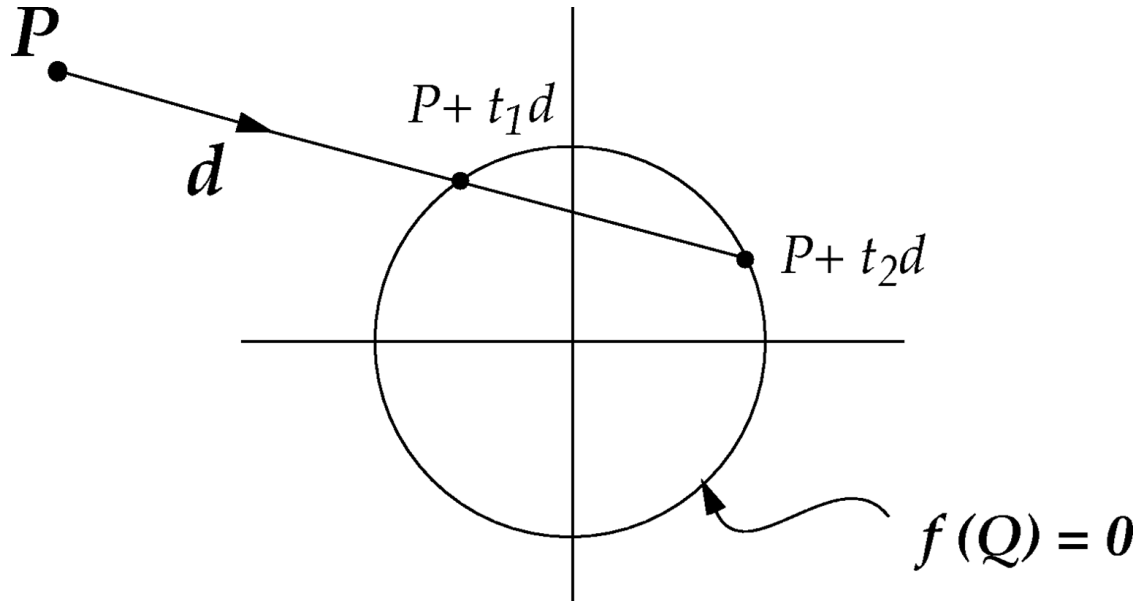
$$f(x,y,z) = x^2 + y^2 + z^2 - R^2$$



Τομή ακτίνας-αντικειμένου

Έμμεσα αντικείμενα

- Σε ποια σημεία (εάν υπάρχουν) η ακτίνα τέμνει ένα αντικείμενο;
- Τα σημεία σε μια ακτίνα έχουν τη μορφή $\boldsymbol{P} + t\boldsymbol{d}$ όπου t είναι οποιοσδήποτε μη αρνητικός πραγματικός αριθμός
- Ένα σημείο επιφάνειας Q που βρίσκεται σε ένα αντικείμενο ικανοποιεί την ιδιότητα $f(Q) = 0$
- Συνδυάζοντας, θέλουμε να γνωρίζουμε «Για ποιες τιμές του t είναι το $f(\boldsymbol{P} + t\boldsymbol{d}) = 0$;»



- Οπότε λύνουμε ένα σύστημα εξισώσεων σε x, y (in 2D) or x, y, z (in 3D)

Τομή ακτίνας-αντικειμένου

Παράδειγμα 2Δ τομής ακτίνας-κύκλου

Έστω το σημείο $\boldsymbol{P}$ = (-3, 1), το διάνυσμα κατεύθυνσης $\boldsymbol{d}$ = (.8, -.6) και ο μοναδιαίος κύκλος : $\boldsymbol{f(x,y) = x^2 + y^2 - R^2}$

- Ένα σημείο πάνω στην ακτίνα είναι:
  $Q = \boldsymbol{P} + t\boldsymbol{d} = (-3,1) + t(.8,-.6) = (-3 + .8t, 1 - .6t)$
- Συνδέοντας αυτό στην εξίσωση του κύκλου:
  $f(Q) = f(-3 + .8t, 1 - .6t) = (-3 + .8t)^2 + (1 - .6t)^2 - 1$
- Οπότε έχουμε: $9 - 4.8t + .64t^2 + 1 - 1.2t + .36t^2 - 1$
- Θέτοντας αυτό ίσο με το μηδέν, παίρνουμε: $t^2 - 6t + 9 = 0$

Τομή ακτίνας-αντικειμένου

Παράδειγμα 2Δ τομής ακτίνας-κύκλου

- Χρησιμοποιώντας την φόρμουλα: $$roots = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$
- Παίρνουμε: $$t = \frac{6 \pm \sqrt{36 - 36}}{2}, \quad t = 3, 3$$
- Επειδή έχουμε μια ρίζα πολλαπλότητας 2, η ακτίνα τέμνει τον κύκλο σε ένα μόνο σημείο (δηλαδή, είναι εφαπτόμενη στον κύκλο)
- Η διακρίνουσα D = $b^2 - 4ac$,μπορεί να προσδιορίσει γρήγορα αν υπάρχει τομή:
  - εάν $D < 0$, φανταστικές ρίζες → Όχι διασταύρωση
  - εάν $D = 0$, διπλή ρίζα → Η ακτίνα εφάπτεται του κύκλου
  - εάν $D > 0$, δύο πραγματικές ρίζες→ Η ακτίνα τέμνει το κύκλο σε 2 σημεία
- Μικρότερος μη αρνητικός πραγματικός t αντιπροσωπεύει την πλησιέστερη τομή

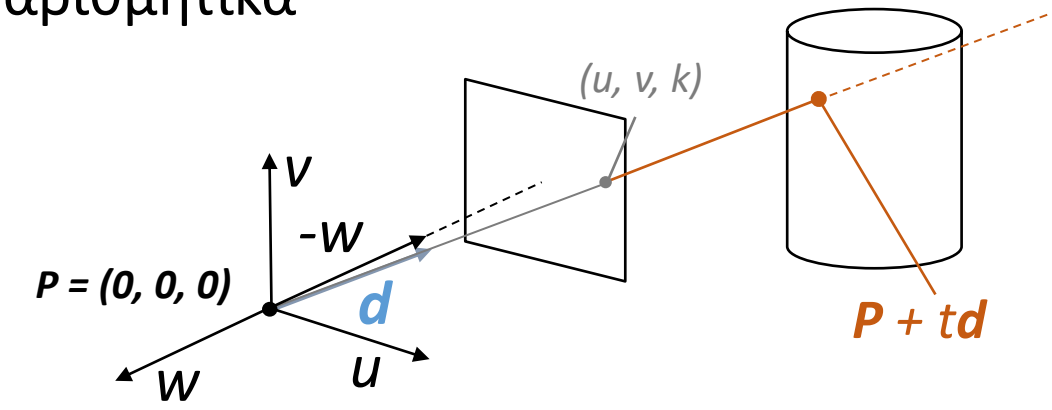
Τομή ακτίνας-αντικειμένου

Παράδειγμα 2Δ τομής ακτίνας-κύκλου

- Γενίκευση:
  - μπορούμε να εκφράσουμε μια επιφάνεια με την εξίσωση $f(Q) = 0$, μια ακτίνα με $\boldsymbol{P} + t\boldsymbol{d}$, και να τα εκφράσουμε ως μια συνάρτηση:

$$f(\boldsymbol{P} + t\boldsymbol{d}) = 0$$

- Το αποτέλεσμα, μετά από κάποια άλγεβρα, είναι μια εξίσωση με το t σαν άγνωστο
- Στη συνέχεια, λύνουμε για t, αναλυτικά ή αριθμητικά



Τομή ακτίνας-αντικειμένου

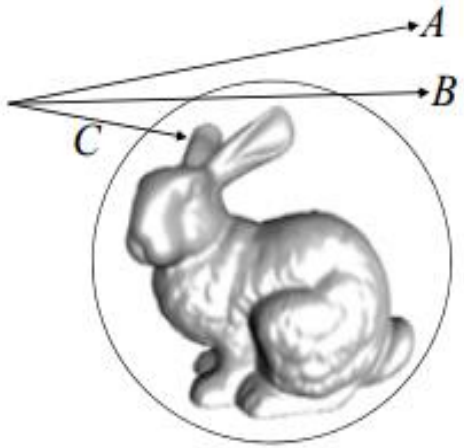
Παράδειγμα 2Δ τομής ακτίνας-κύκλου

- Αντικατέστησε την εξίσωση της ακτίνας (***P*** + *t****d***) στην εξίσωση του αντικειμένου και λύσε ως προς *t*
  - η μικρότερη μη αρνητική τιμή t είναι από η πλησιέστερη στην επιφάνεια
- Για τα περίπλοκα αντικείμενα (που δεν ορίζονται από μια ενιαία εξίσωση), γράψτε ένα σύνολο από ισότητες και ανισότητες και έπειτα τον λύστε ως προς τις μεμονωμένες επιφάνειες
- Η λύση μπορεί να γενικευτεί ώστε να χειριστεί όλα τα είδη αντικειμένων από σύνθετους συνδυασμούς
  - Constructive Solid Geometry (CSG), όπου τα αντικείμενα αποθηκεύονται ως μια ιεραρχία από πρωτόγονα αντικείμενα και 3-Δ λειτουργίες (Ένωση, τομή, διαφορά)
  - “blobby objects”, οι οποίες είναι επιφάνειες που ορίζονται από ένα σύνολο εξισώσεων (F(x,y,z)=0)
  - Mandelbulbs: https://www.shadertoy.com/view/XsXXWS

Cool Blob!

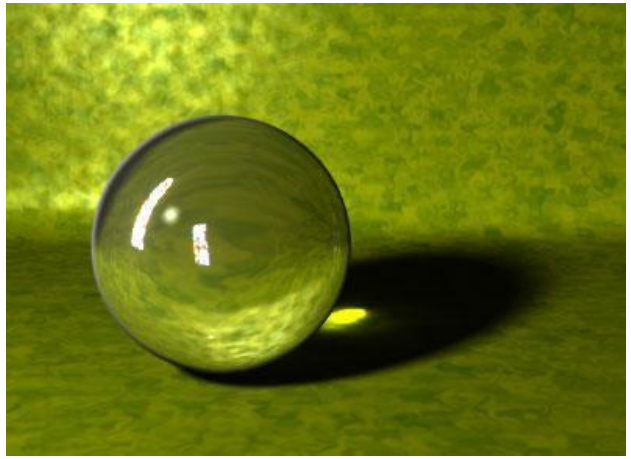
Bounding Volume: Reducing Ray-Object Intersections

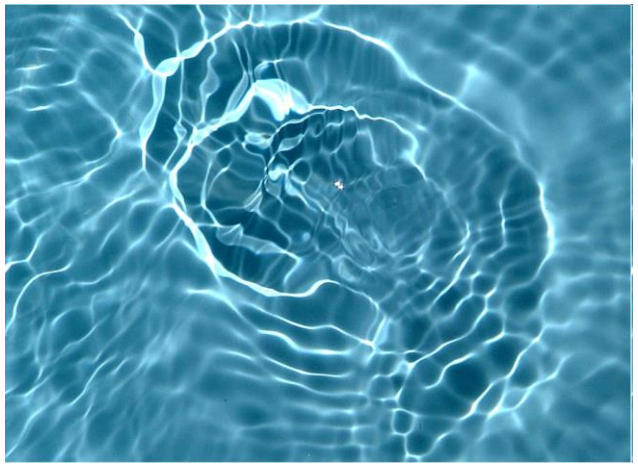
- Bounding Volumes:
- Πρώτη δοκιμή για διασταύρωση με το bounding volume
  - Μόνο εάν υπάρχει διασταύρωση, θα δοκιμάσουμε τα αντικείμενα που περικλείονται από τον όγκο.
  - Μπορεί να εξοικονομήσει πολλούς υπολογισμούς όταν η ακτίνα δεν χτυπά το λαγουδάκι
  - Μπορεί να χρησιμοποιεί κουτιά, σφαίρες κτλ



Επεκτάσεις

- Monte-Carlo Ray Tracing and Photon Mapping





Παρακολούθηση ακτίνας σε πραγματικό χρόνο

- Παραδοσιακά, η παρακολούθηση ακτίνας ήταν υπολογιστικά αδύνατο να γίνει σε πραγματικό χρόνο
  - παραλληλισμός λόγω της ανεξαρτησίας κάθε ακτίνας, έτσι ένας CPU ή πυρήνας για κάθε εικονοκύτταρο (pixel)
  - δύσκολο να πετύχουμε βελτιστοποίηση στο hardware :
    - large amount of floating point calculations
    - complex control flow structure
    - complex memory access for scene data

- Μια λύση: software-based, highly optimized raytracer using cluster with multiple CPUs
  - Prior to ubiquitous GPU-based methods, ray tracing was done on CPU clusters to take advantage of parallelism
  - Hard to have widespread adoption because of size and cost
  - Can speed up with more cores per CPU
- Large CPU render farms still dominant for non-real time CGI for movies and animations; GPU farms and path tracing are taking over
  - Weta Digital (Planet of the Apes), ILM (Jurassic World), Pixar (Monsters University)

OpenRT rendering of five maple trees and 28,000 sunflowers (35,000 triangles) on 48 CPUs. OpenRT is no longer maintained.

Παρακολούθηση ακτίνας σε πραγματικό χρόνο

- Modern solution: Use GPUs to speed up ray tracing
  - NVIDIA RTX - combo hardware/software made for real time ray tracing
    - NVIDIA RTX demo 2018: https://www.youtube.com/watch?v=tjf-1BxpR9c
    - Project Sol Part 2 demo of real time ray tracing, CES 2019 : https://www.youtube.com/watch?v=pNmhJx8yPLk&t=1s
    - Tech Focus: Ray Tracing - The Future of Gaming Graphics? https://www.youtube.com/watch?v=moKV5_BpxjM

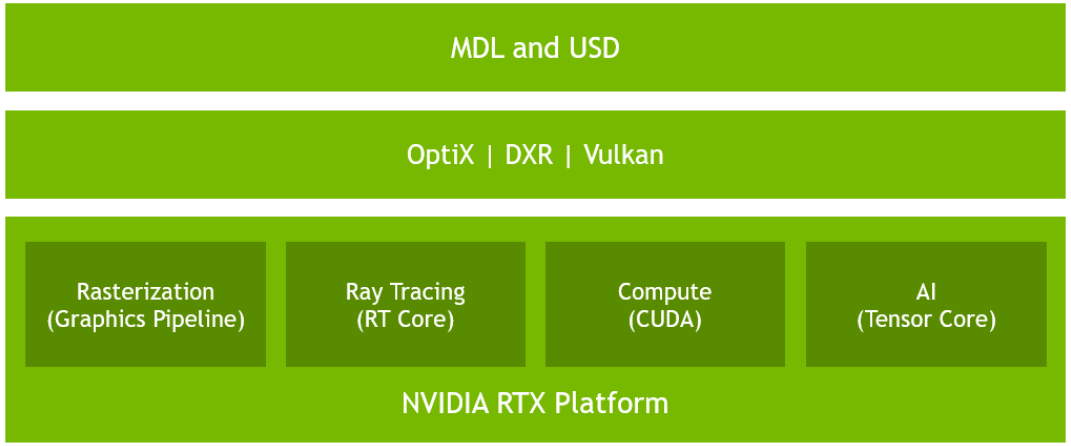
Tech Focus: Ray Tracing – The Future of Gaming Graphics?

Project Sol

NVIDIA RTX Demo

Παρακολούθηση ακτίνας σε πραγματικό χρόνο

- NVIDIA RTX GPUs:
  - Turing architecture shading processor (14 TFLOPS + 14 TIPS)
  - RT core with built-in Ray-Triangle intersection and BVH Traversal (10 Giga Rays/s!)
  - Tensor core (110 TFLOPS FP16) for AI (denoising and more)
- Learn More
  - http://developer.nvidia.com/optix
  - http://developer.nvidia.com/rtx



POV-Ray: Pretty Pictures

Free Advanced Raytracer

- Full-featured raytracer available online: povray.org
- Obligatory pretty pictures (see hof.povray.org):

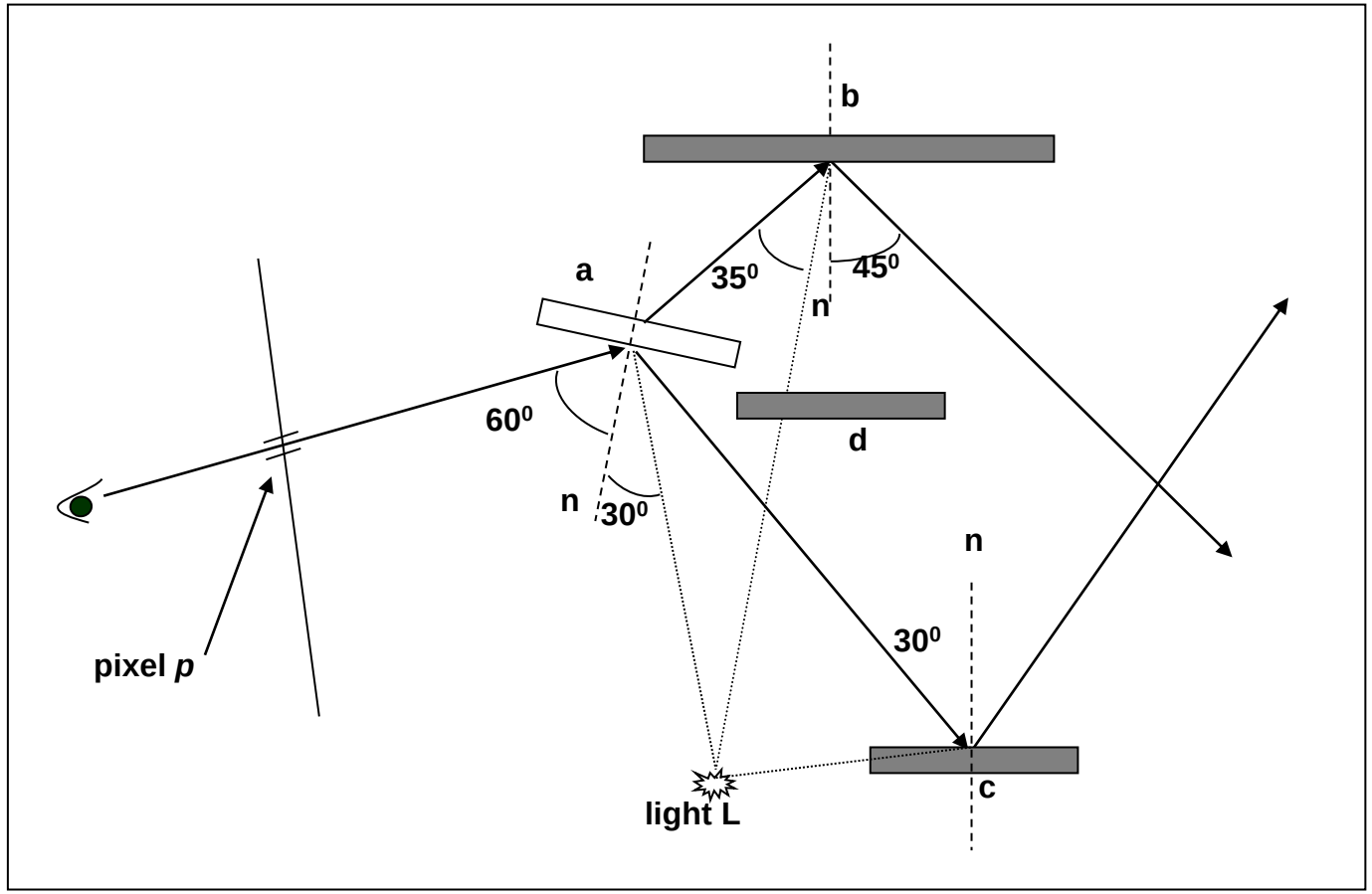
Image credits can be found on hof.povray.org

Άσκηση

- Αν υποθέσουμε μονοχρωματικό φως, υπολογίστε την ένταση (I) στο πίξελ p στην πιο κάτω εικόνα. Οι διακεκομμένες γραμμές (n) είναι οι κάθετοι των επιφανειών.
- Η ένταση της πηγής L = 1.0, ένταση του έμμεσου φωτισμού (ambient) = 0.3 και του φόντου (background) = 0.4

- Οι συντελεστές της επιφάνειας a:
  - $k_a = 0.2$
  - $k_d = 0.2$
  - $k_r = k_s = 0.3$
  - $k_t = 0.3$
  - m (shininess) = 2

- Οι συντελεστές των επιφανειών b, c και d:
  - $k_a = 0.2$
  - $k_d = 0.3$
  - $k_r = k_s = 0.5$
  - $k_t = 0$ (οι επιφάνειες είναι αδιαφανείς)



Άσκηση

- Ένταση της πηγής L = 1.0, ambient = 0.3, background = 0.4
- Επιφάνειας a: $k_a = 0.2$, $k_d = 0.2$, $k_r = ks = 0.3$, $k_t = 0.3$, m (shininess) = 2
- Επιφάνειες b, c και d: $k_a = 0.2$, $k_d = 0.3$, $k_r = k_s = 0.5$, $k_t = 0$

$$I^p = I^a$$

$$I^a = I^a_{local} + K^a_r I^a_r + K^a_t I^a_t$$

$$I^a_{local} = K^a_a I^a_a + I_L\left(K^a_d (n.l) + K^a_s (h.n)^m\right)$$
$$= 0.2x0.3 + 1\left(0.2\cos(30) + 0.3\cos(15)^2\right)$$
$$= 0.51$$

$$I^a_r = I^c = I^c_{local} + K^c_r I^c_r + K^c_t I^c_t$$

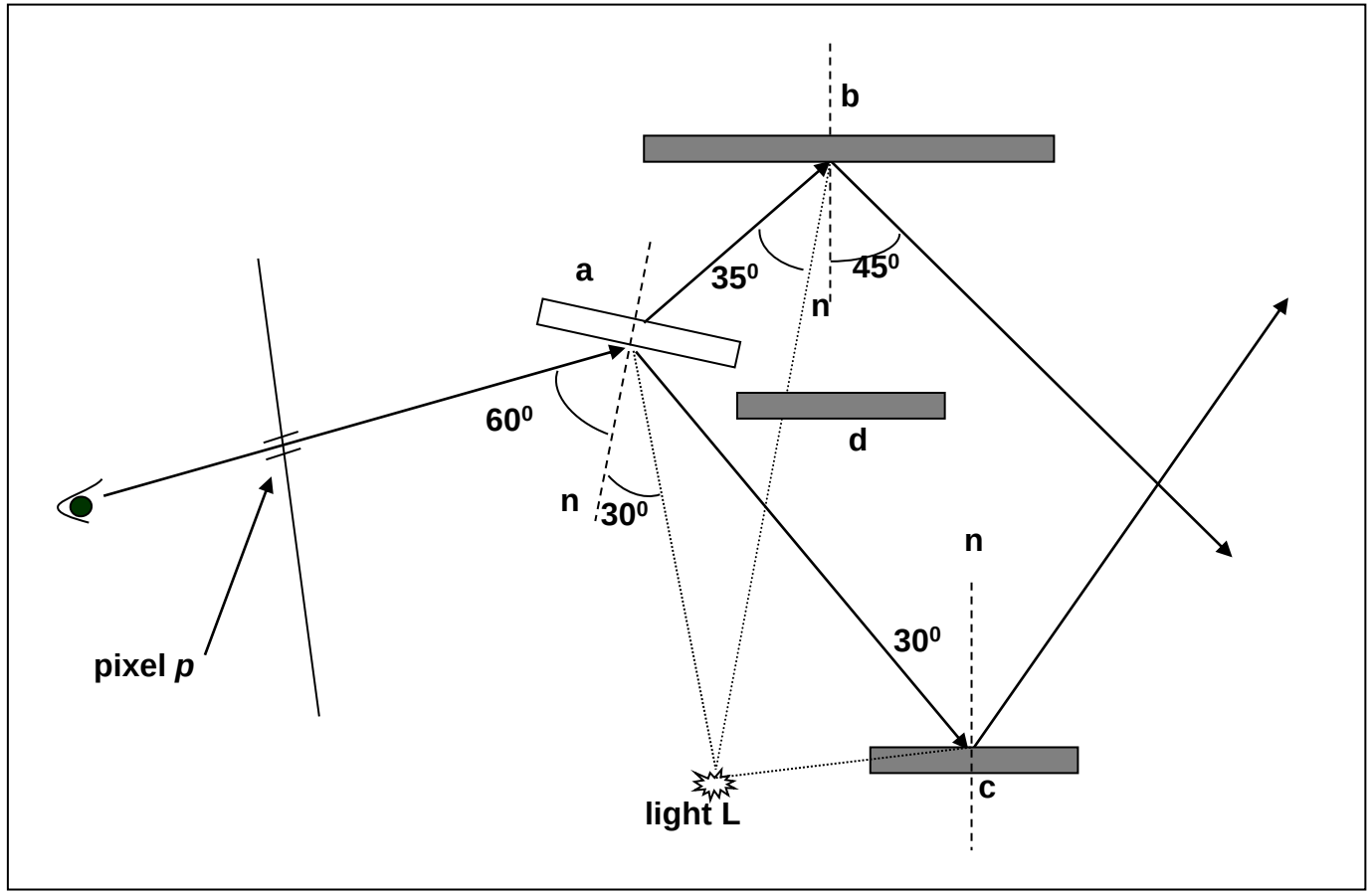
$$= I^c_{local} + K^c_r I^c_r \quad \text{αδιαφανη}$$

$$I^c_{local} = K^c_a I^c_a + I_L\left(K^c_d (n.l) + K^c_s (h.n)^m\right)$$
$$I^c_{local} = K^c_a I^c_a \quad \text{light not visible}$$
$$I^c_{local} = 0.2x0.3 = 0.06$$

$$K^c_r I^c_r = 0.5x0.4 = 0.2$$

$$I^a_r = 0.06 + 0.2 = 0.26$$

$$K^a_r I^a_r = 0.3x0.26 = 0.078$$



Περίληψη

- Η αναδρομική παρακολούθηση ακτίνας είναι μια καλή προσομοίωση των specular αντανακλάσεων
- Έχουμε δει πώς η παρακολούθηση ακτίνας μπορεί να επεκταθεί για να περιλάβει σκιές, αντανακλάσεις και διαφανείς επιφάνειες
- Εντούτοις είναι μια πολύ αργή διαδικασία και χάνουμε ακόμα μερικούς τύπους επιδράσεων!
- Το υψηλό κόστος προέρχεται από τις πολλές τομές ακτίνας-τριγώνου
- Το κόστος μπορεί να μειωθεί με τη χρήση του bounding volume hierarchy