The Visual Computer

1/86

International Journal
of Computer Graphics

Volume 2 Number 1
January 1986

Springer
International

The Visual Computer

International Journal of Computer Graphics

Contents

Volume 2 Number 1 1986

Cover: Composition with windowpanes and cristal ball by Donald R. Leich. © Digital Effects Inc.

Springer International

The Visual Computer

International Journal of Computer Graphics

Official Journal of the Computer Graphics Society, CGS

The Visual Computer is dedicated to reporting on the state-of-the-art technology in the fields of computer vision, graphics, and imaging with specific focus on applications.

The journal's scope includes detection and communication of visual data, intermediate data structures and processing techniques for visual data and computer graphics, and graphical representations of images.

Contributions to the journal may be original papers, survey reports, or tutorial papers. A regular column carries product news from both academia and industry. A calendar of events provides timely information on upcoming meetings and symposia. Items for this column as well as for the product news section are welcome.

Specific areas of interest include:

- Architecture and Design for Vision Systems
- Networks for Communicating Visual Data
- Computer Vision and Pattern Recognition
- Graphics Workstations
- VLSI for Vision and Graphics Applications
- Languages for Vision and Graphics Applications
- Data Management and Structures for Vision Information
- Geometric Modelling
- Robotics
- Image Generation Techniques
- Numerical Analysis for Computer Graphics
- Computer Animation
- CAD/CAM, CAE, CAT
- Graphics Applications for Business, Education, Medicine and the Sciences
- Ergonomics and Human Factors

Springer International

Copyright

Submission of a manuscript implies: that the work described has not been published before (except in the form of an abstract or as part of a published lecture, review, or thesis); that it is not under consideration for publication elsewhere; that its publication has been approved by all coauthors, if any, as well as by the responsible authorities at the institute where the work has been carried out; that, if and when the manuscript is accepted for publication, the authors agree to automatic transfer of the copyright to the publisher; and that the manuscript will not be published elsewhere in any language without the consent of the copyright holders.

All articles published in this journal are protected by copyright, which covers the exclusive rights to reproduce and distribute the article (e.g., as offprints), as well as all translation rights. No material published in this journal may be reproduced photographically or stored on microfilm, in electronic data bases, video disks, etc., without first obtaining written permission from the publisher.

The use of general descriptive names, trade names, trademarks, etc., in this publication, even if not specifically identified, does not imply that these names are not protected by the relevant laws and regulations.

While the advice and information in this journal is believed to be true and accurate at the date of its going to press, neither the authors, the editors, nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Special regulations for photocopies in the USA: Photocopies may be made for personal or in-house use beyond the limitations stipulated under Section 107 or 108 of U.S. Copyright Law, provided a fee is paid. This fee is US $0.20 per page, or a minimum of US $1.00 if an article contains fewer than five pages. All fees should be paid to the Copyright Clearance Center, Inc., 21 Congress Street, Salem, MA 01970, USA, stating the ISSN 0178-2789, the volume, and the first and last page numbers of each article copied. The copyright owner's consent does not include copying for general distribution, promotion, new works, or resale. In these cases, specific written permission must first be obtained from the publisher.

Other regulations: Authors publishing in this journal can, under certain conditions, benefit from library and photocopy fees collected by VG WORT. Authors of German nationality and those resident in the Federal Republic of Germany or Berlin (West), as well as citizens of Austria, Switzerland and member countries of the European Community, may apply to Verwertungsgesellschaft WORT, Abteilung Wissenschaft, Goethestrasse 49, D-8000 München 2, for detailed information.

Subscription information

Volume 2 with 6 issues will appear in 1986.

North America. Annual subscription rate: approx. US $114.00 (single issue price: approx. US $19.00) including carriage charges. Subscriptions are entered with prepayment only. Orders should be addressed to:
Springer-Verlag New York Inc., Service Center Secaucus, 44 Hartz Way, Secaucus, NJ 07094, USA,
Tel. (201) 348-4033, Telex 023-125994

All other countries: Annual subscription rate: DM 296.00, plus carriage charges. Airmail delivery on request only. Volume price: DM 296.00, single issue price: DM 59.20, plus carriage charges. Airmail delivery on request only. Carriage charges for SAL (Surface Airmail Lifted) to Japan, India, Australia and New Zealand are available on request. Orders can either be placed with your bookdealer or sent directly to: Springer-Verlag, Heidelberger Platz 3, 1000 Berlin 33, Tel. (0)30/8207-1, Telex 1-83319

Changes of address: Allow six weeks for all changes to become effective. All communications should include both old and new addresses (with Postal Codes) and should be accompanied by a mailing label from a recent issue.

Microform

Microform editions are available from: University Microfilms International, 300 N. Zeeb Road, Ann Arbor, MI 48106, USA

Production
Springer-Verlag, Journal Production Department II, Postfach 105280, D-6900 Heidelberg 1, Federal Republic of Germany, Tel. (0)6221/487-439, Telex 4-61723

Responsible for advertisements
E. Lückermann, Springer-Verlag, Heidelberger Platz 3, 1000 Berlin 33, Tel. (0)30/8207-1, Telex 1-85411

Printers
Universitätsdruckerei H. Stürtz AG
D-8700 Würzburg

Publisher

Springer-Verlag GmbH & Co. KG
D-1000 Berlin 33, Printed in Germany

The Visual Computer

Dispersive refraction in ray tracing*

Spencer W. Thomas

Computer Science Department,
University of Utah, Salt Lake City,
UT 84112, USA

①

Dispersive refraction is the property that gives gemstones their fire, and that makes prisms produce a spectrum from white light. Modeling dispersion in a ray tracing environment requires solution of some new problems, but allows production of more exciting images. The mechanism of dispersive refraction is discussed, and its implementation is described. Pictures of a prism and of several diamonds are included. Images generated by this technique are realistic, but are computationally expensive.

Key words: Ray tracing – Refraction – Image synthesis

Ray tracing is a common method of accurately modelling the interaction of light with objects in a synthetic image (Whitted 1980; Hall and Greenberg 1983; Cook et al. 1984). It produces the most realistic looking images of any image synthesis technique to date. Most ray tracing implementations model reflection of light from shiny and matte surfaces, and refraction of light through transparent objects. However, they make the simplifying assumption that the index of refraction of an object is constant over the entire wavelength range of the visible spectrum. Most refractive media do not satisfy this assumption, but instead refract different colors of light through different angles. We develop a simple method of modeling such *dispersive* refraction, and outline an implementation.

Dispersive refraction

Simple modeling of refraction and reflection, as has traditionally been done in ray tracing, is insufficient to model the "fire" of a diamond or the action of a prism. This comes from *dispersive* refraction, that is, from the variation in the refractive index of the material with the wavelength of the light. In most dispersive materials, blue light rays are refracted more strongly than red rays. The difference between the index of refraction of a material for a short, or blue, wavelength and that for a long, or red, wavelength is called the *dispersion* of the material. Dispersive refraction of a light ray by a prism is illustrated in Fig. 1. The variation of the index of refraction with wavelength for two types of crown glass is shown in Fig. 2, (data from the *Handbook of Chemistry and Physics* (Weast 1971)). However, such detailed information is seldom available. For example, most gemstone references cite only the index of refraction at the sodium D line, 589 nm, and indicate the dispersion as the difference of the indices of refraction at two other wavelengths, commonly the B (687 nm) and G (431 nm) lines. From this information, simple linear interpolation can be used to find the approximate index of refraction for any given wavelength. A phenomenon familiar to most people is *total internal reflection* (Born 1975). This occurs when a light ray passing from a medium with a high index of refraction to one with a lower index of refraction would be refracted by more than 90 degrees from the normal. In this case, none of the light is transmitted and the interface acts as a perfect mirror.

---

* This work was supported in part by the National Science Foundation (DCR-8341796 and MCS-8121750), the Defense Advanced Research Projects Agency (DAAK11-84-K-0017), and the Office of Naval Research (N00014-82-K-0351). All opinions, findings, conclusions or recommendations expressed in this document are those of the author and do not necessarily reflect the views of the sponsoring agencies

The Visual Computer (1986) 2:3–8

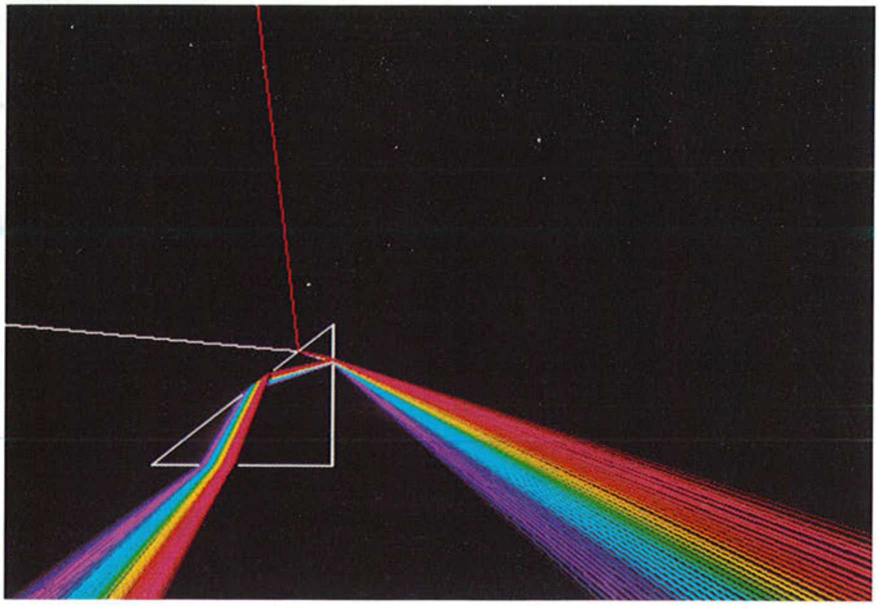

Fig. 1. Dispersive refraction by a prism

However, the transition is not an abrupt one from high transmission to no transmission; the amount of light transmitted, and hence the amount of light reflected, depends on the angle of incidence of the ray. This behavior is embodied by the *Fresnel equations*, which are formulae for the reflection and transmission coefficients at the interface between two materials. If both materials are totally transparent, the Fresnel equations (Born and Wolf 1975) can be written

$$r_{\parallel} = \frac{n_2 \cos\theta_1 - n_1 \cos\theta_2}{n_2 \cos\theta_1 + n_1 \cos\theta_2}$$

$$r_{\perp} = \frac{n_1 \cos\theta_1 - n_2 \cos\theta_2}{n_1 \cos\theta_1 + n_2 \cos\theta_2}$$

$$R = (r_{\parallel}^2 + r_{\perp}^2)/2$$

$$T = 1 - R$$

where n_1 and n_2 are the indices of refraction of the two materials, θ_1 and θ_2 are the angle of incidence and refraction, respectively, $r_{\parallel}$ is the reflectivity for the component polarized parallel to the surface, $r_{\perp}$ is the reflectivity for the component polarized perpendicular to the surface, R is the reflection coefficient, and T is the transmission coefficient. The form of the Fresnel equation given above is particularly useful in computer graphics applications, as the angle cosines are easily calculated. The equation for the reflection coefficient is also given, in a different form, in (Cook and Torrance 1982).

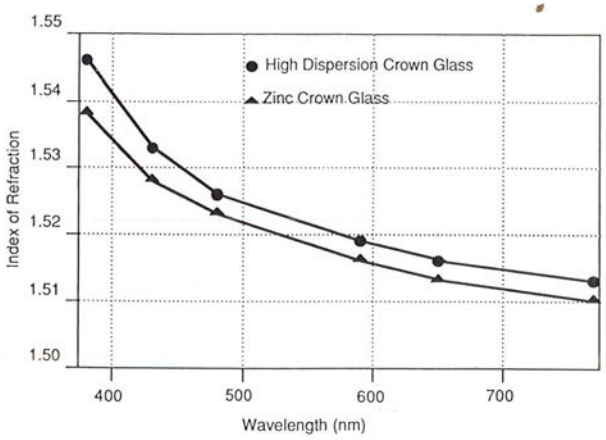

Fig. 2. Dispersion in glass

Light and color

Proper simulation of dispersion and color absorption requires that colors be represented by the intensity of the wavelengths comprising the returned light ray. It is necessary to transform this *spectral distribution* function to the red, green and blue intensities necessary for driving a color CRT display to make pictures. This can be accomplished by go-

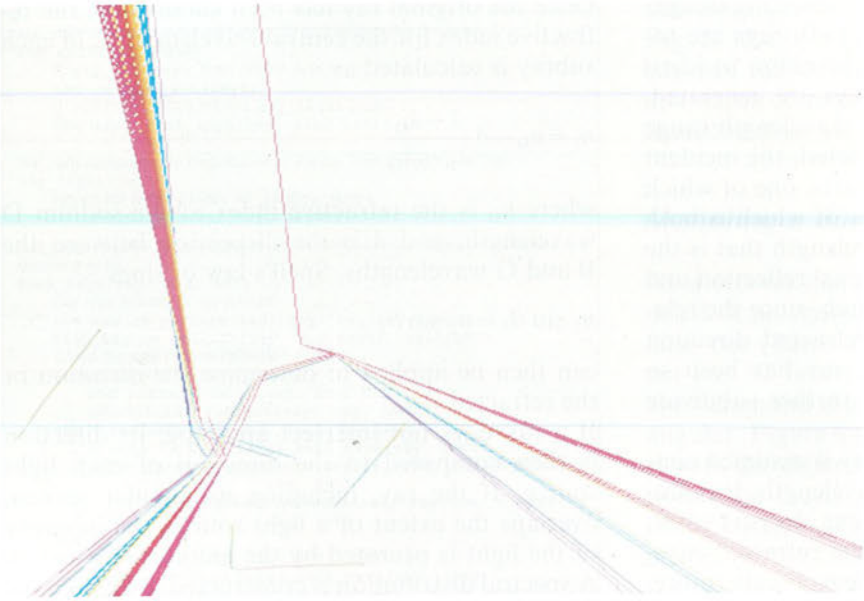

Fig. 3. Ray clumping due to incorrect dispersion calculation

ing through the CIE *trilinear* color space (Hall and Greenberg 1983). To transform a spectral distribution into trilinear coordinates, X, Y, and Z, three integrals are calculated,

$$X=\int i(\lambda)\,x(\lambda)\,d\lambda,$$
$$Y=\int i(\lambda)\,y(\lambda)\,d\lambda,$$
$$Z=\int i(\lambda)\,z(\lambda)\,d\lambda.$$

The trilinear coordinates can then be transformed to RGB numbers by referring to the coordinates of each of the phosphors of the color CRT display. If the color lies outside the *gamut* of the display, one or more of the RGB coefficients will be out of range. In this case, it is necessary to desaturate the color to display it. The intersection of a line between the given color and a white of equal brightness with the boundary of the gamut of the display gives the appropriate desaturated color for display.

Implementation

Dispersion is modeled by breaking a ray, as needed, into several subrays, each covering only a portion of the spectrum, at refractive interfaces. As might be expected, this process can generate many rays, and it is not uncommon to fire over 100 rays for a single pixel when modeling a highly dispersive medium. Each ray carries two new pieces of information: the portion of the spectrum covered by the ray, and the angular spread of the ray. The spread is necessary when splitting a ray into subrays, and also enters into the decision of whether to split a ray. In addition to the angular spread, a spread vector, with magnitude *tan (spread/2)* defines the plane of divergence of the ray.

Modeling dispersion

Each object is given an index of refraction and a dispersion value. When a ray encounters an interface between two different indices of refraction, two new rays, one for each endpoint of the wavelength range covered by the incident ray, are refracted. The angular spread between the two refracted rays is computed and compared to the angular extent of the smallest light source. If it is larger than some fraction of the light source extent, the incident ray is subdivided into several rays so that the spread of each refracted ray will be small enough. Through experimentation, it was determined that it is necessary for about six subrays to hit a light source. If fewer rays are used, various quantization effects appear in the resulting image. If both the low and high wavelength rays are totally reflected at the interface, the entire incident ray is reflected, and no other rays are generated. If the ray at one endpoint of the wavelength range is reflected and the other is refracted, the incident ray must first be split into two parts, one of which is totally reflected, and the other of which is both refracted and reflected. The wavelength that is the dividing point between total internal reflection and refraction is found by binary search, since the relation between wavelength and refracted direction is nonlinear. After the incident ray has been so divided, it may be necessary to further subdivide the refracted part.

If the direction of the incident ray is assumed constant over the entire range of wavelengths it represents, anomalous results can occur. In particular, a clumping effect is visible in the refracted rays. Figure 3 demonstrates the problem. Furthermore, in this figure, the rays coming out of the diagonal of the prism should not cross. To correct for these effects, when an incident ray is subdivided the resulting rays must have their direction of incidence adjusted according to their position in the wavelength range. Figure 4 illustrates this process. In the left part of the figure, a ray incident on a surface is shown, with the angular spread of the ray shown by the lighter rays. If it is to be split into two subrays, each subray will be given a direction that is the average of the direction of the incident ray and one of the "spread" rays. These are shown by the dashed rays. To avoid recalculating the intersection of the ray and the surface, the subrays are assumed to be incident at the same point as the original. The effective result is illustrated in the right half of the figure. The new rays have been labeled so that their identity can be established between the two pictures.

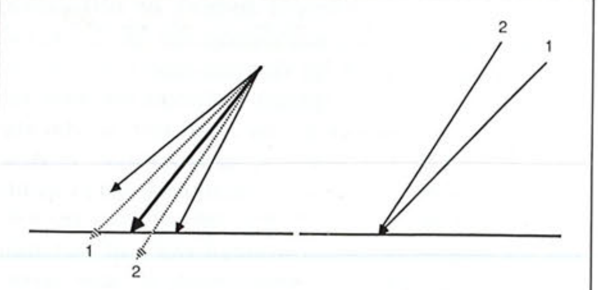

Fig. 4. Perturbing the incidentray

Once the original ray has been subdivided, the refractive index for the central wavelength, λ, of each subray is calculated as

$$n_\lambda = n_D - \delta \frac{\lambda - \lambda_D}{\lambda_B - \lambda_G}$$

where n_D is the refractive index at the sodium D wavelength, and δ is the dispersion between the B and G wavelengths. Snell's law of sines

$$n_1 \sin\theta_1 = n_2 \sin\theta_2$$

can then be applied to determine the direction of the refracted ray.

If a ray does not intersect anything, its direction is then compared to the direction of each light source. If the ray, including its angular spread, overlaps the extent of a light source, the intensity of the light is prorated by the amount of overlap. A spectral distribution is constructed, assigning the calculated intensity to the range of the spectrum covered by the ray. This is returned as the color of the ray. To calculate the color of a ray that has been split, the spectral distributions of the subrays are added together. Note that when combining a reflected ray with a refracted ray, each must be multiplied by the appropriate reflection or transmission coefficient.

Opaque, diffusely reflecting objects are treated in the same way as they are in standard ray tracing, except that their color must be modeled as a spectral distribution. The angular extents of the lights must be taken into account when computing the diffuse lighting component. The area covered by a light with an angular radius of s is $2\pi(1-\sqrt{1-s^2})$. This is multiplied by the brightness of the light to get its contribution to the total illumination.

To prevent a ray from bouncing around inside an object indefinitely, two pruning criteria are used. First, each ray has a significance factor, which is just the product of all the reflection or transmission coefficients that will apply to the result returned by the ray. When this factor becomes very small, the ray is assumed to return black, because, in any case, its total contribution to the pixel color will be insignificant. This is similar to the "adaptive free-depth control" of (Hall and Greenberg 1983). As a back up, a maximum recursion depth can be set.

The algorithms described above are given in a simplified pseudo-code form below.

```
ray_solid intersects a ray with the solid and returns the ray color.
ray_solid(ray, sig)
    if sig < thresh then return black;
    int := find_int(ray);
    if null(int) then return ray_light(ray)
    else return fork_rays(ray, sig, int);

ray_light computes the contribution to the ray from the light sources.
ray_light(ray)
    return for each light in lights sum
        overlap_amount(ray, light);

fork_rays decides what happens at a particular ray-object intersection.  It
returns a color.
fork_rays(ray, sig, int)
    Get rays at limits of ray spread
    low_ray := perturb(ray, ray.low, ray.low);
    high_ray := perturb(ray, ray.high, ray.high);
    Check for total internal reflection
    if reflectp(low_ray, int)
        and reflectp(high_ray, int) then
        return total_reflect(ray, sig, int)
    else if reflectp(low_ray, int) then begin
        r_wave := find_total_refl_wave(ray, int);
        return fork_rays(perturb(ray, r_wave, ray.high),
                                sig, int)
            + total_reflect(perturb(ray, ray.low, r_wave),
                                sig, int)
    end
    else if reflectp(high_ray, int) then begin
        r_wave := find_total_refl_wave(ray, int);
        return fork_rays(perturb(ray, ray.low, r_wave),
                                sig, int)
            + total_reflect(perturb(ray, r_wave, ray.high),
                                sig, int)
    end
    else begin
        Calculate number of new rays from spread of dispersed rays
        nrays := 1 + 6*(ray.spread +
                    disperse_spread(low_ray, high_ray)) /
                light_spread;
        d_wave := (ray.high-ray.low) / nrays;
        return for i := 1 to nrays sum
            refrect(perturb(ray, ray.low+(i-1)*d_wave,
                        ray.low+i*d_wave), sig, int)
    end

refrect generates the reflected and refracted rays and returns a color.
refrect(ray, sig, int)
    Generate refracted ray using Snell's law
    refract_ray := refract(ray, int);
    Calculate transmission coefficient
    T := fresnel(ray, refract_ray, int);
    return ray_solid(refract_ray, T*sig) * T
            + ray_solid(reflect(ray, int), (1-T)*sig)
                * (1-T);

total_reflect generates a totally reflected ray and returns its color.
total_reflect(ray, sig, int)
    return ray_solid(reflect(ray, sig, int), sig);

perturb generates a ray corresponding to the portion of the given ray between
the low and high wavelengths.
perturb(ray, low, high)
    low_dir := (2*(low-ray.low)/(ray.high-ray.low) - 1)
            * ray.spread_vec + ray.dir;
    high_dir := (2*(high-ray.low)/(ray.high-ray.low) - 1)
            * ray.spread_vec + ray.dir;
    new_ray(dir, low, high, spread, spread_vec)
    return new_ray(normalize((low_dir+high_dir)/2),
                low, high, length((high_dir-low_dir)/2),
                (high_dir-low_dir)/2);
```

Results

Two images are included to demonstrate the capabilities of the algorithm. The first is a simple image of a prism with a piece of paper behind it with the word *prism* written on it. This is shown in Fig. 5. Note particularly the color fringing at the edges of the letters. The curved edges on the prism and sheet of paper are barrel distortion resulting from the close viewpoint and wide angle of view.

Modeling a diamond

As a second demonstration, a diamond was modeled. A polygonal model of a diamond in the *brilliant* cut (Bank 1973) was created. The brilliant cut has 57 facets, of which one is octagonal, 16 are "diamond shaped", and the remainder are triangular. Figure 6 shows a line drawing of the model with the different facets labeled. The angles between the facets are critical to proper brilliancy of the cut gem. They are measured from the plane of the "girdle", the broadest part of the diamond, and are also shown in the figure. The brilliant cut is designed so that almost all light entering the front of the diamond (i.e., through the table, star, kite, or upper girdle facets) will be returned through those facets, and that no light will be lost through the back of the gem. It is the standard cut for diamonds and many other gems.

Fig. 5. A prism refracting a word

The ray traced image shown in Fig. 7 includes a "setting" of seven diamonds, with six smaller diamonds arranged around a large central diamond. The group floats above a blue and white checkered cloth. The scene is lit by nine spot lights and one hemispherical light simulating ambient room light. This scene demonstrates how much of our perception of a diamond owes to being able to move

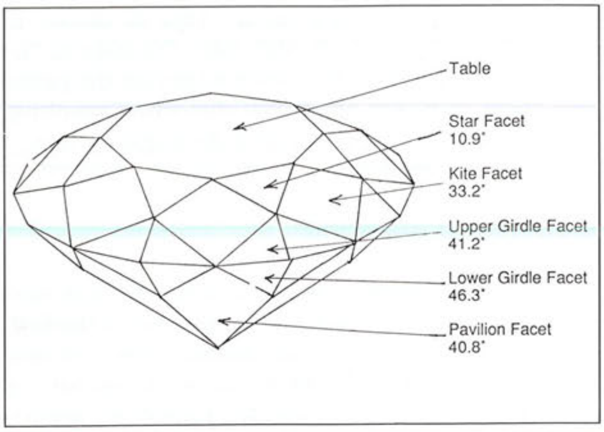

Fig. 6. A brilliant cut diamond

Fig. 7. A group of diamonds

it or ourselves while viewing it. In any fixed orientation, there are few "sparkles", and most of the diamond reflects little light back to the observer.

Conclusion

A simple model of refractive dispersion leads to realistic looking pictures of gemstones and other dispersive media. A key feature of the algorithm is the the adaptive subdivision of the spectral distribution of rays at dispersive interfaces. Like any ray tracing implementation, it is computationally expensive. It tends to be more expensive than standard ray tracing because any ray-surface intersection can generate many rays, instead of the two or three generated by a normal ray tracing algorithm.

References

Bank H (1973) From the World of Gemstones. Pinguin, Innsbruck, Austria

Born M, Wolf E (1975) Principles of Optics. Pergamon Press, Oxford

Cook RL, Torrance KE (1982) A Reflectance Model for Computer Graphics. ACM Transactions on Graphics 1:7–24

Cook RL, Porter, T, Carpenter L (1984) Distributed Ray Tracing. Comput Graphics (Proc. SIGGRAPH '84) 18:137–145

Hall RA, Greenberg DP (1983) A Testbed for Realistic Image Synthesis. IEEE Computer Graphics and Applications 3:10–20

Weast RC (ed) (1971) Handbook of Chemistry and Physics. Chemical Rubber Company, Cleveland, Ohio

Whitted T (1980) An Improved Illumination Model for Shaded Display. Comm ACM 23:343–349