

## Corso di Laurea in Scienze dell'Architettura

### Corso di Fondamenti e Applicazioni di Geometria Descrittiva

Riccardo Migliari<sup>1</sup>,  
Leonardo Baglioni<sup>2</sup>, Jessica Romor<sup>3</sup>, Marta Salvatore<sup>4</sup>

1 Professore ordinario di Fondamenti e applicazioni della geometria descrittiva – titolare del corso  
2, Ricercatore, 3 e 4 Dottori di ricerca in Rilievo e rappresentazione dell'architettura e dell'ambiente

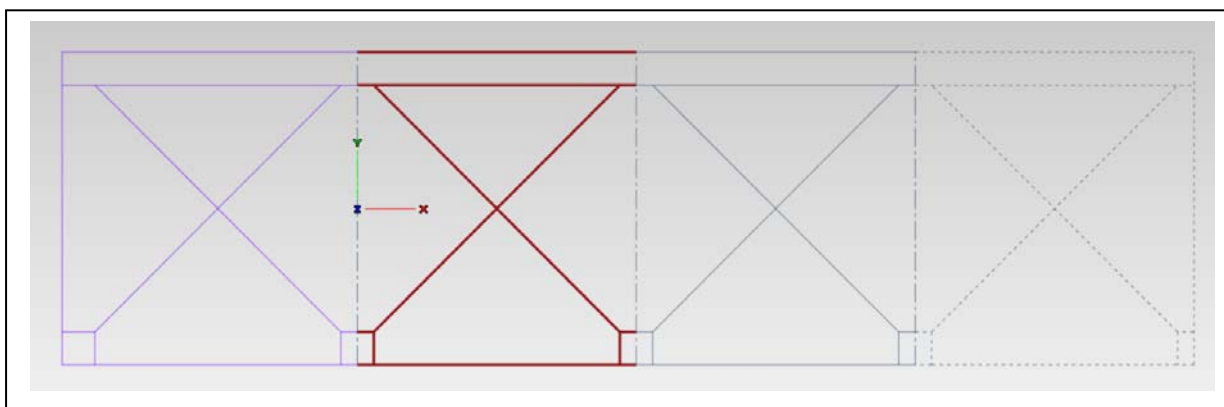
Lezione 14 – 17 Novembre 2014

#### Argomenti

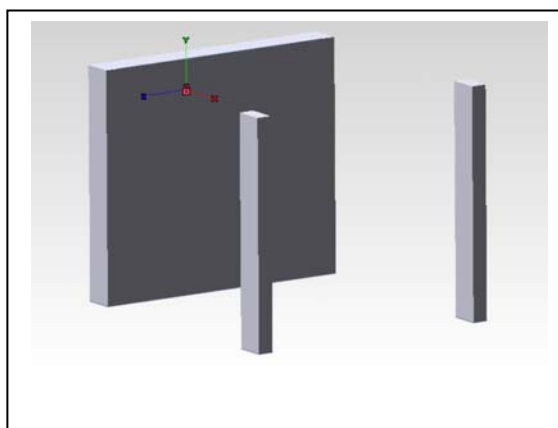
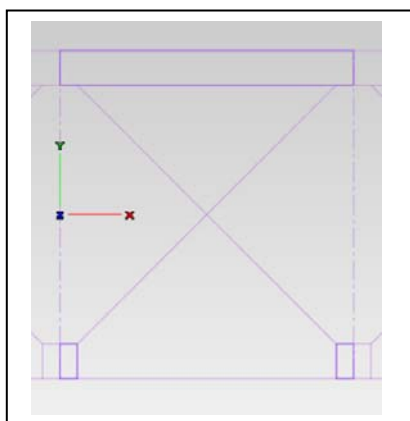
**Le volte (2).** Costruzione e modellazione solida della volta crociera.  
Controllo della tassellazione ed esportazione in ambiente di rendering. Parametri essenziali del rendering.  
Impostazione della prospettiva della volta a crociera. L'osservatore come modulo dello spazio prospettico.  
**Esercitazione in aula:** costruzione di una volta a crociera.

#### Costruzione e modellazione solida della volta crociera

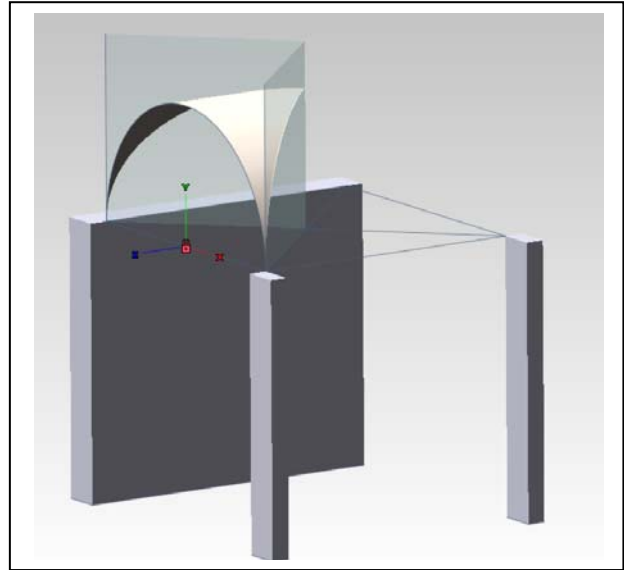
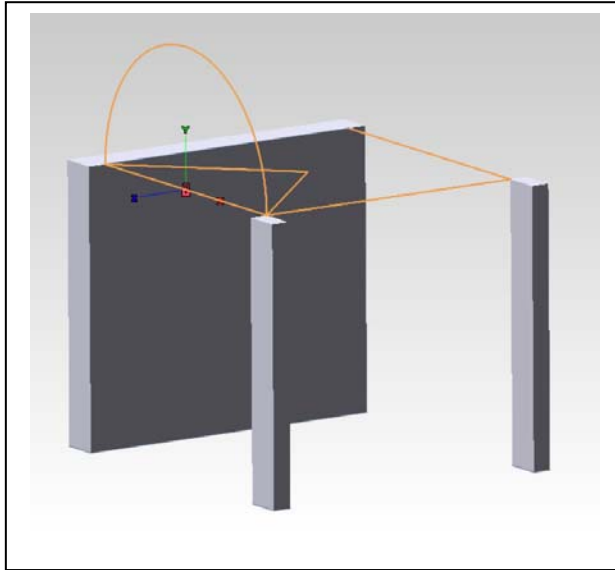
Immaginiamo un sistema di volte a crociera che coprono un portico nel quale l'elemento che compare a sinistra nella figura qui sotto è la testata, mentre l'elemento in rosso si ripete eguale  $n$  volte.  
Vogliamo costruire il modello solido di questo elemento ricorsivo per utilizzarlo nella rappresentazione prospettica e chiaroscurale del portico.



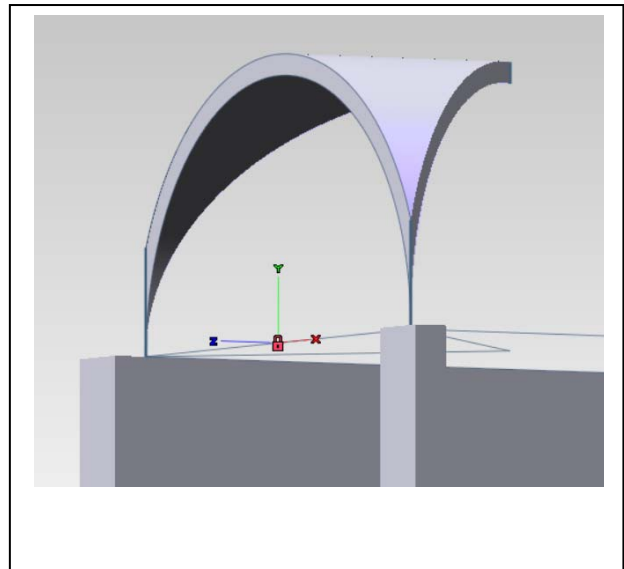
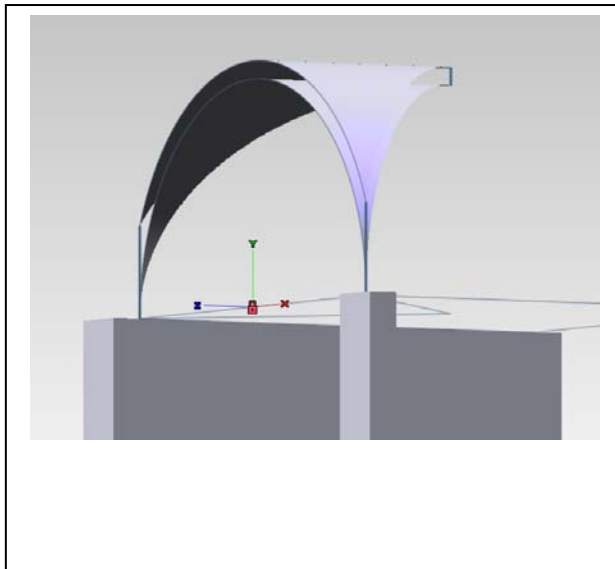
- Costruzione degli elementi portanti: i profili (a sinistra, sotto).
- Costruzione dei solidi degli elementi portanti (a destra, sotto).



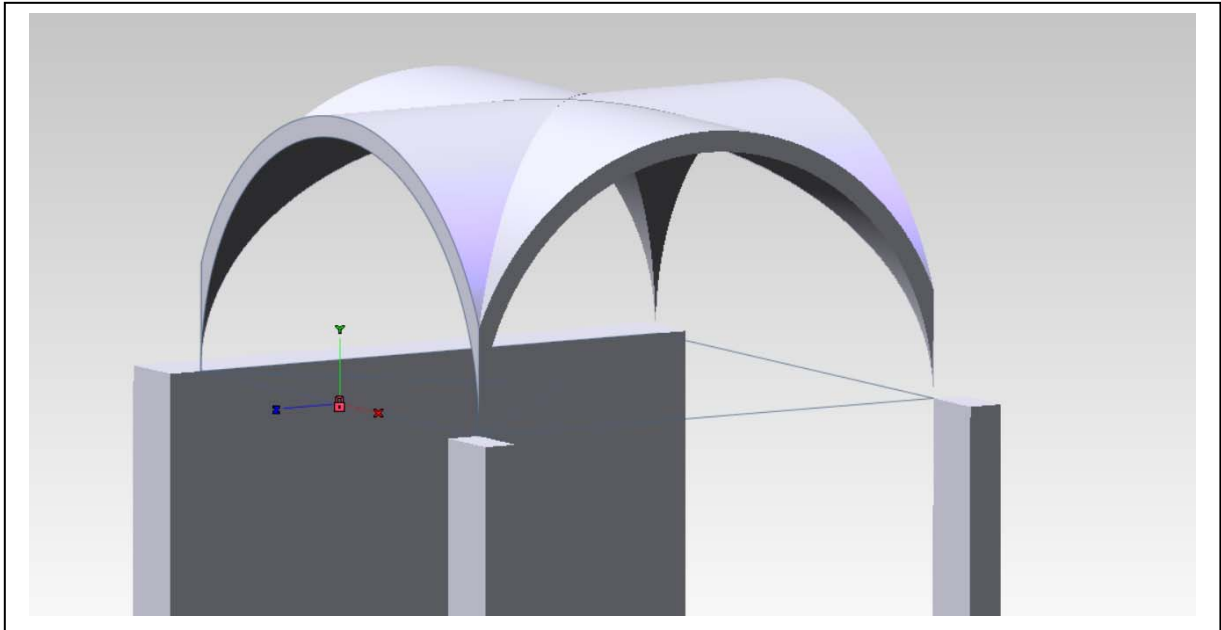
- c. Costruzione del profilo dell'arco di imposta di un'unghia e delle proiezioni dei costoloni, sul piano di imposta della volta (sotto, a sinistra).
- d. Costruzione dell'intradosso dell'unghia (sotto, a destra).



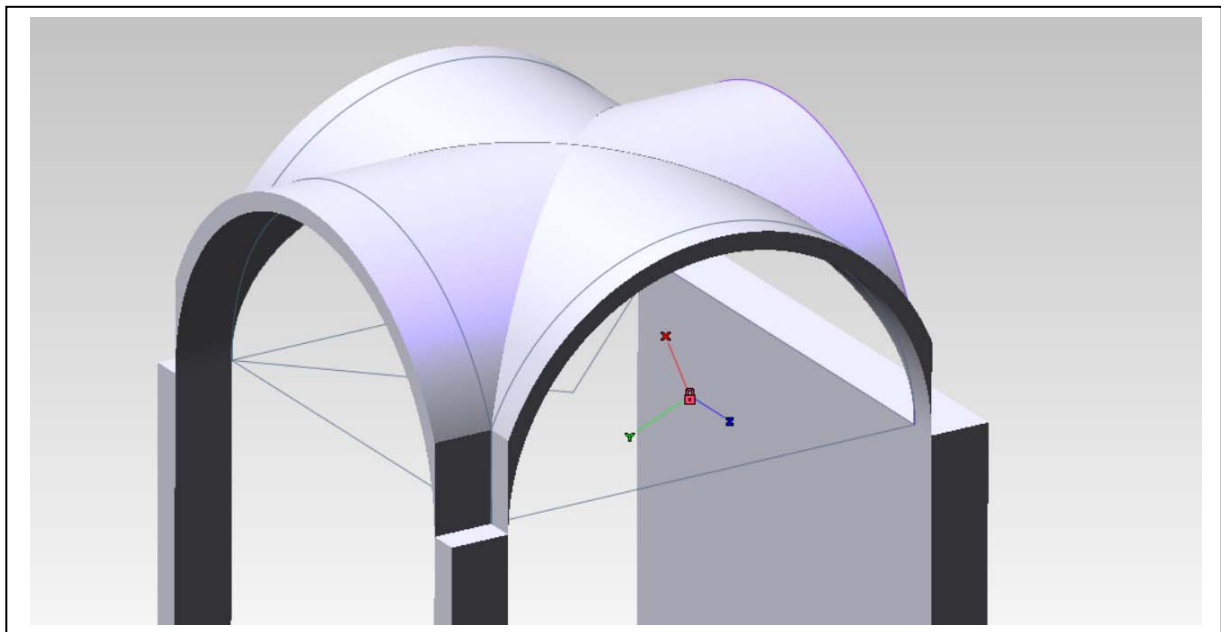
- e. Costruzione dell'estradosso dell'unghia e degli spigoli che lo connettono all'intradosso (sotto, a sinistra).
- f. Costruzione delle superfici che chiudono il solido dell'unghia (sotto, a destra).



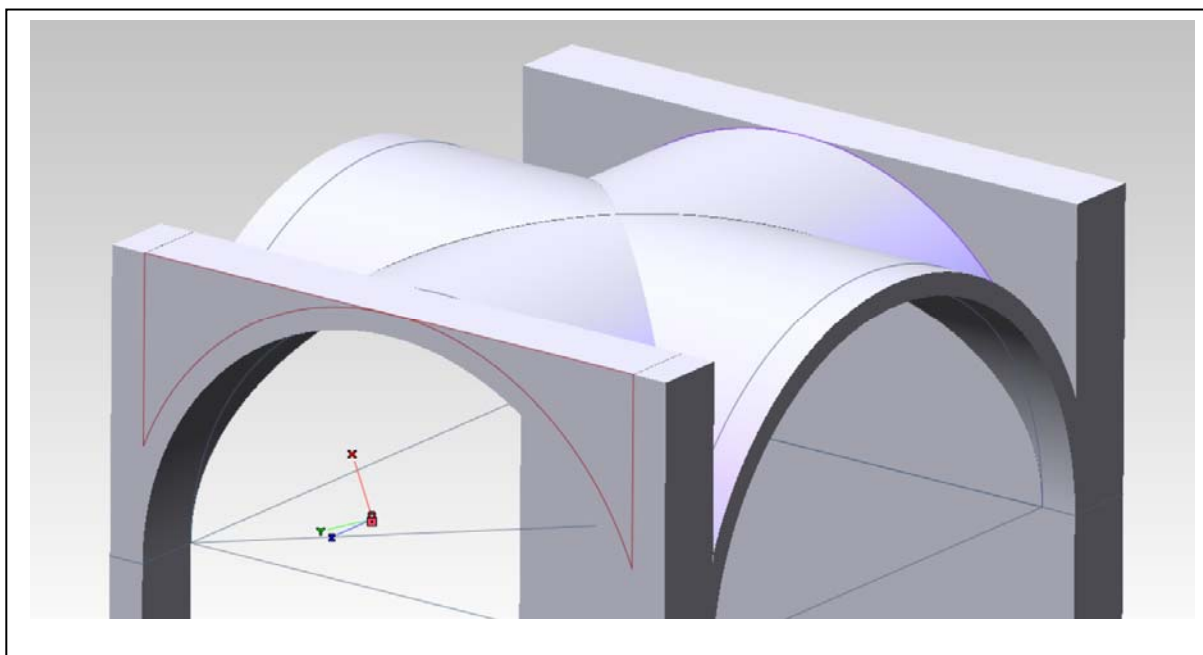
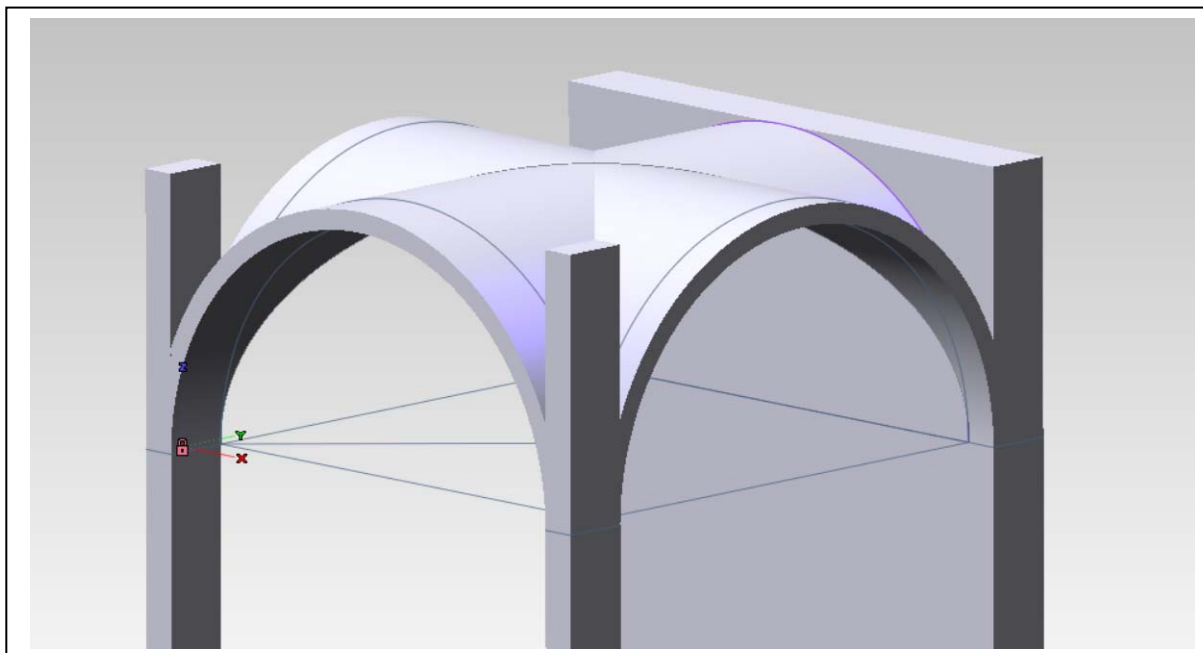
- g. L'insieme di superfici dell'unghia viene implosa in solido e il solido copiato tre volte e ruotato intorno all'asse verticale che passa per il cervello della volta, in modo da formare la crociera (sotto, alla pagina seguente).



h. Applicando tre volte la lavorazione Appendice lineare, si realizzano gli archi di imposta delle tre unghie che poggino sui pilastri (sotto).

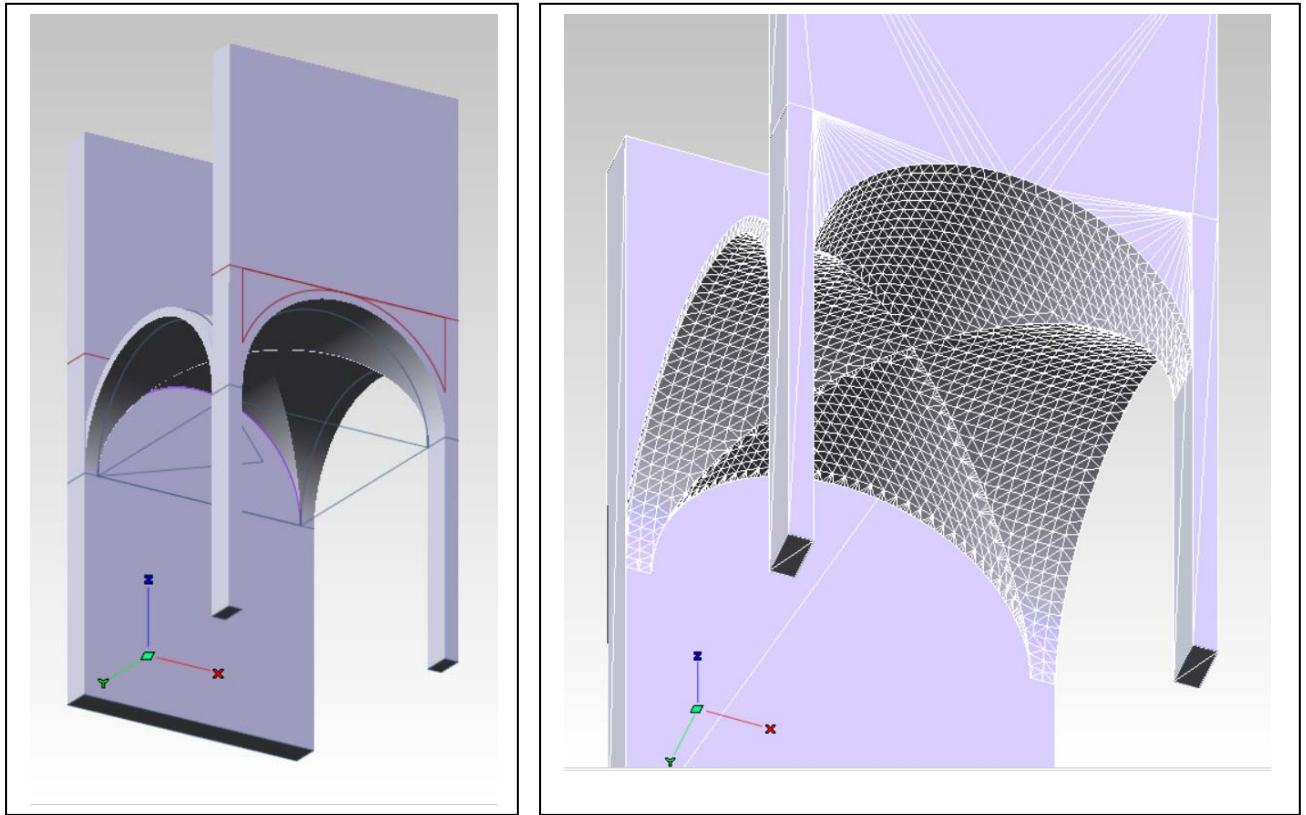


- i. Con la stessa lavorazione si alzano i pilastri e la parete cieca fino all'altezza di estradosso del cervello della volta (sotto)



- j. Per chiudere i vuoti sopra l'estradosso dell'arco sul fronte si disegna il profilo in rosso nella figura e si applica, ancora, la lavorazione Appendice lineare (sopra).

- k. Da ultimo si costruiscono le strutture del primo piano (a destra) e si cura la tassellazione prima delle operazioni di esportazione nell'ambiente di rendering. È possibile limitare la lunghezza del lato dei poligoni della tassellazione con il comando Strumenti/Opzioni/Proprietà documento/Generale/Avanzato – Tassellazione: Abilita lunghezza spigolo max.



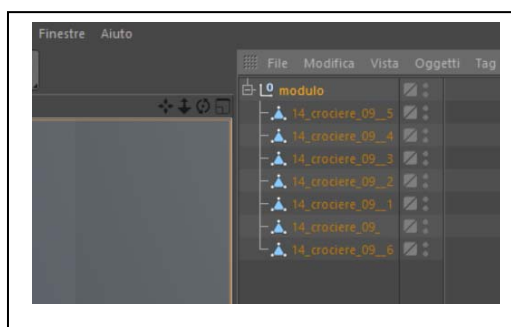
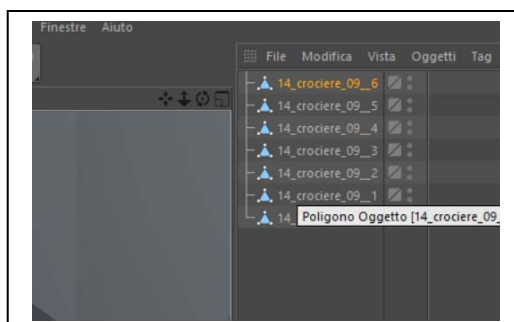
Per esportare il modello nell'ambiente di rendering, occorre salvare la tassellazione nel formato stl (esistono anche altre opzioni, come il formato obj, il wrml, il dxf).

Il comando che esegue questa funzione in thinkdesign è File/Salva come. Occorre indicare il formato stl nella finestra Salva come.

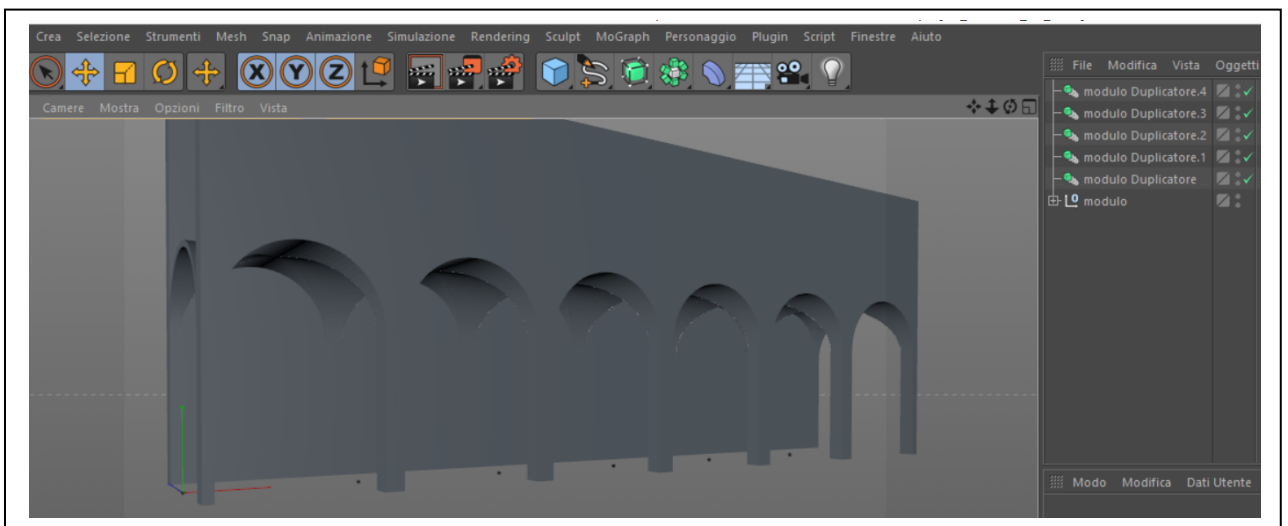
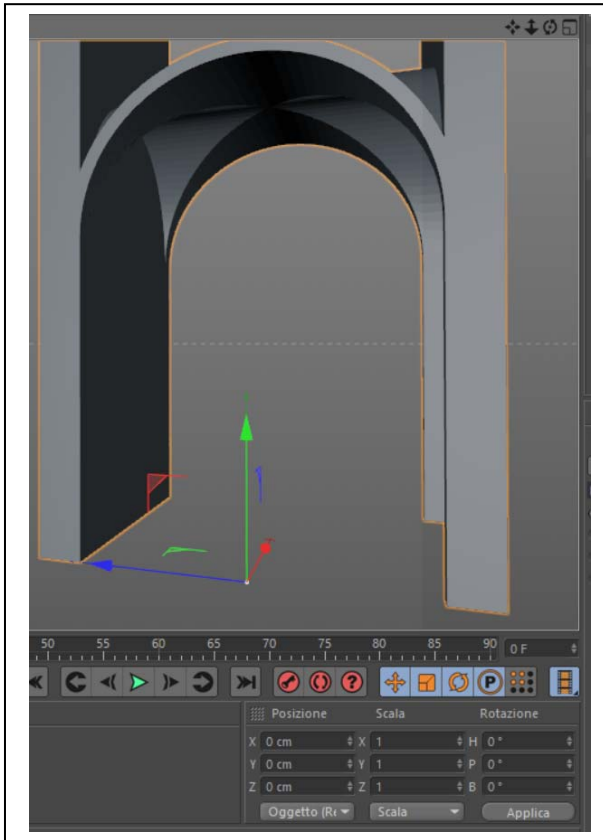
In Opzioni indicare, prima File multipli, poi Usa tassellazione corrente.

### ***Importazione del modello in ambiente di rendering (Cinema 4D)***

1. In Cinema 4D (nel seguito C4D): File/Aggiungi e ripetere l'operazione per tutti i file generati che sono uno per ognuno dei solidi creati. Se necessario, quando i solidi sono troppo numerosi, conviene unirli prima della esportazione in formato stl. Questa operazione può essere velocizzata premendo Ctrl+Shift+O (che aggiunge un file), selezionando il file e premendo due volte di seguito invio.
2. Quando tutti i file sono stati caricati, nella storia del modello bisogna selezionarli tutti (con una finestra di selezione – vedi sotto a sinistra) e raggrupparli con il comando Oggetti/Raggruppa oggetti del menu della storia (a destra). Rinominare il gruppo “Nullo”.



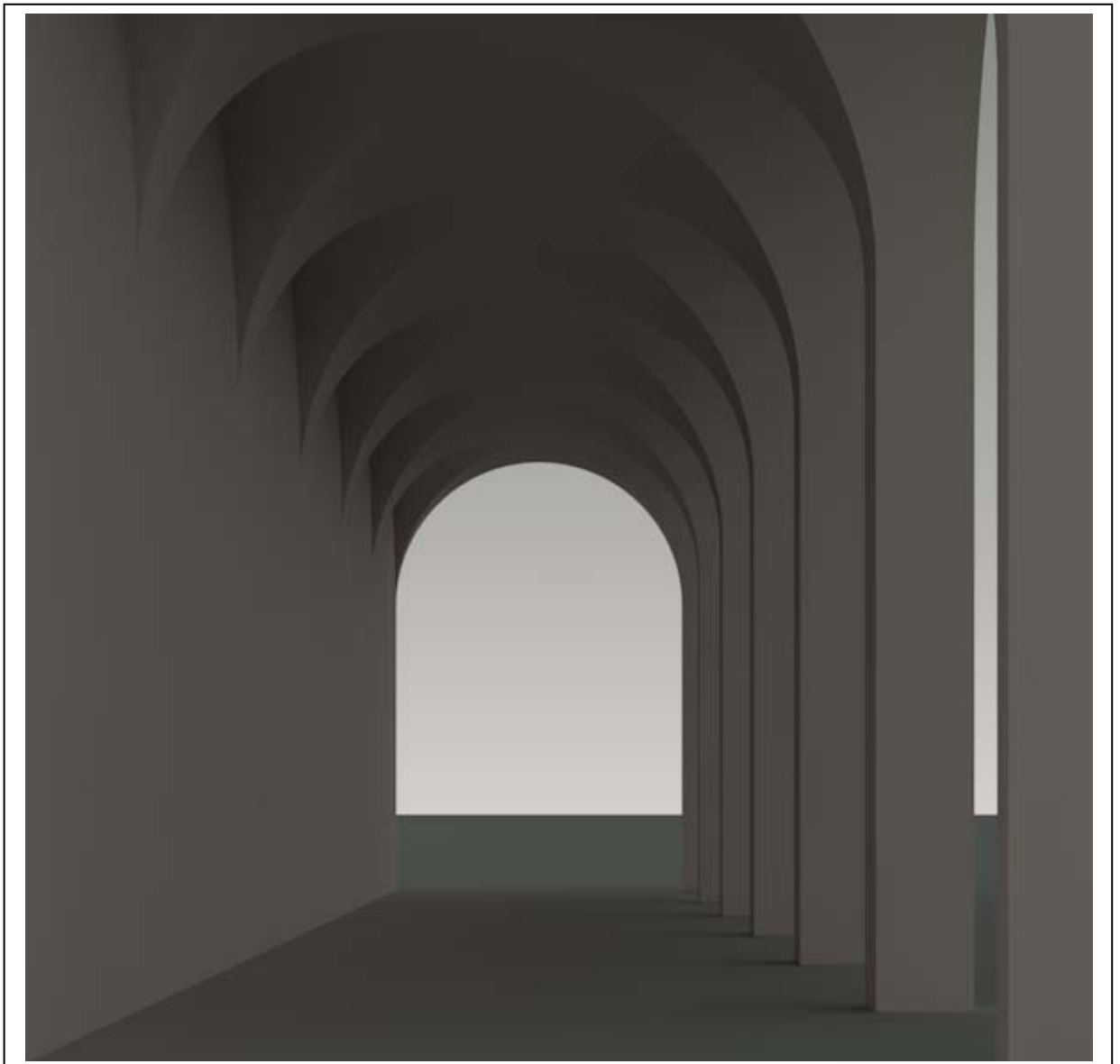
3. Conviene applicare al gruppo una variazione di scala (in genere riducendola 10 volte) per evitare che risulti troppo grande rispetto ad altri oggetti già presenti nel software come l'icona degli assi o le mappe degli shader di irregolarità delle superfici (vedi sotto a sinistra).



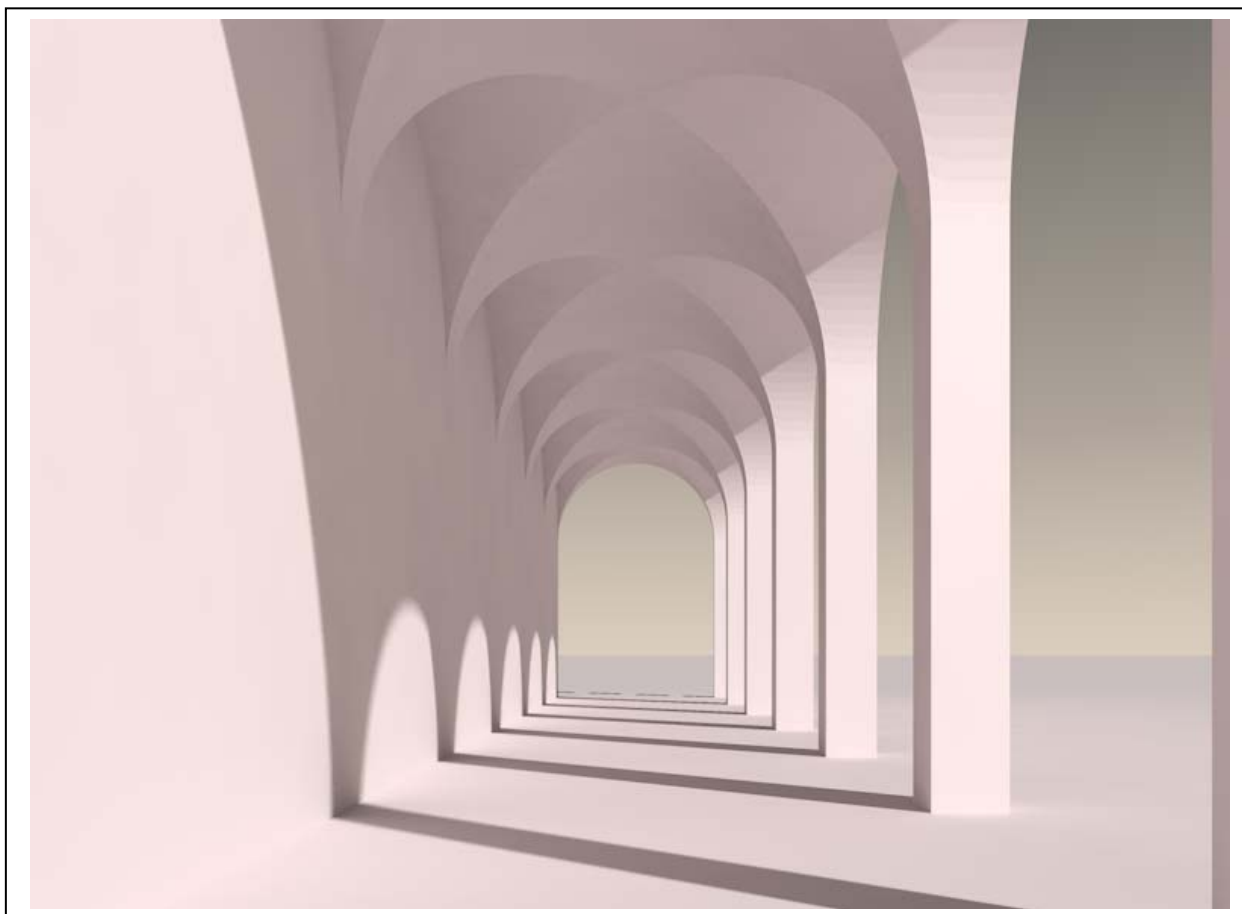
4. Per creare un sistema di volte, bisogna replicare il modulo  $n$  volte. A questo scopo si seleziona il gruppo e si dà il comando Crea/Modellazione/Duplicatore  $n$  volte. Il software creerà altrettante copie sovrapposte all'originale. Per formare il sistema, basta selezionare le copie una ad una e traslarle lungo l'asse  $x$  di una quantità pari alla larghezza del modulo (vedi sotto, a destra).
5. A questo punto ogni modifica o integrazione eseguita sul modulo verrà automaticamente ripetuta sulle copie. Per completare il modello sarà bene creare una superficie orizzontale: Crea/Scena/Pavimento.

### ***Parametri principali di rendering (Cinema 4D)***

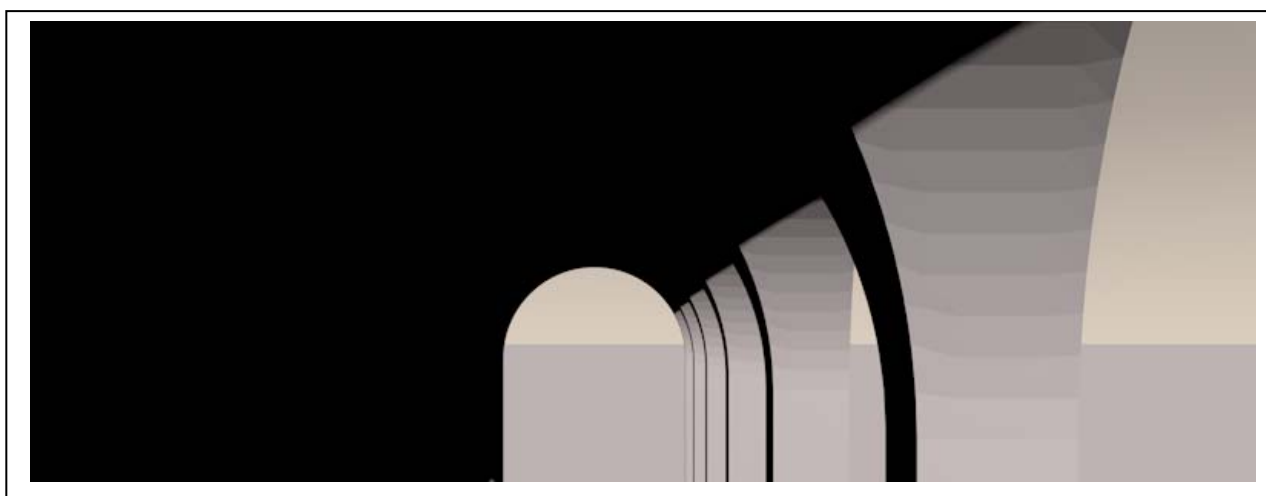
1. **illuminazione.** La luce più importante non è quella diretta, come si potrebbe pensare, ma è la luce diffusa dal cielo. Crea/Scena/Cielo. Per rendere il cielo occorre attribuirgli un 'materiale'. Perciò, nella scheda dei materiali in basso a sinistra File/Nuovo materiale. Rinominatelo. Apritelo e assegnate un gradiente al canale colore. **ATTENZIONE** a non usare colori saturi: un cielo colorato produce luce colorata, che colora anche tutti gli oggetti che illumina. Attribuite anche un colore grigio chiaro al modulo, nel suo insieme e al pavimento. Associate un materiale chiaro anche al modulo volta e al pavimento.
2. **Primo test dell'illuminazione diffusa.** Nel Menu principale: Rendering/Modifica i settaggi di rendering. In Effetto, scegliere Illuminazione globale. In Metodo secondario: Irradiance cache. I due Metodi (Primario e Secondario) riguardano il primo l'illuminazione diretta, il secondo i riflessi, ovvero la luce indiretta, che proviene dalle superfici illuminate. I vari 'Metodi' sono algoritmi che si differenziano per la qualità della simulazione degli effetti reali della luce sui corpi. Possono richiedere da pochi secondi a ore di calcolo. A riguardo consultate la guida. Per studiare il risultato: nel Menu principale: Rendering/Renderizza in visualizzatore immagini.
3. **Per impostare una prima prospettiva.** Crea/Camera/Camera e, nel menu della vista, Camere/Camera di scena/ Camera. Posizionare la camera ad altezza d'uomo. Impostare una lunghezza focale lunga (90 mm) e un Offset y negativo del 25%. Il risultato dovrebbe essere simile a questo:



4. **Per impostare la luce del Sole.** A questo punto è possibile illuminare la scena con la luce solare e generare le ombre. Crea/Luec/Infinita inserisce una sorgente di luce a raggi paralleli. Con questo oggetto selezionato, nella scheda del proprietà, in basso a destra, impostare Ombra: Area. Nella scheda delle coordinate dell'oggetto luce infinita, impostare la rotazione H=-45; P=-35; B=0. Ripetere il test di rendering. Il risultato dovrebbe essere simile a questo:

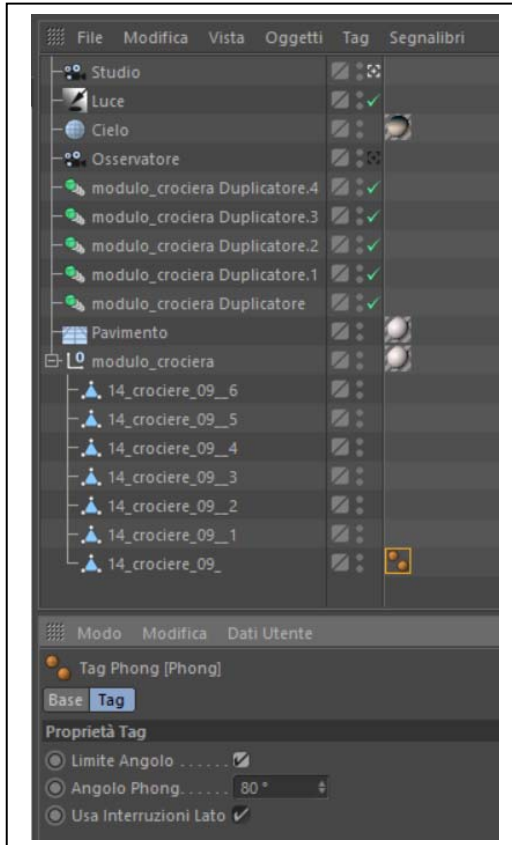


5. **L'algoritmo di Phong.** Nella parte in cui le volte si appoggiano sul pilastro, si nota un passaggio brusco dei colori, l'assenza della sfumatura che sarebbe lecito aspettarsi, dato che la superficie è curva. Ciò avviene perché la superficie, in realtà, non è curva, ma è un poliedro. Il fenomeno è ancor meglio visibile se si disabilita l'Illuminazione globale, osservandolo da vicino (vedi sotto).





6. **Come attivare l'algoritmo di Phong.** L'algoritmo di Phong calcola il grado di illuminazione delle superfici applicando la legge di Lambert (già studiata a proposito del chiaroscuro e della scala delle tinte) ma, in più, applica se richiesto una interpolazione che rende graduale sfumato il passaggio tra i vari toni di grigio associati alle facce del poliedro. Per applicarlo agli oggetti, si seleziona l'oggetto nella Storia del modello e si dà, nel Menu relativo, il comando Tag/Cinema 4D tag/Phong. Accanto all'oggetto, nella Storia, comparirà un'icona formata da due piccole sfere.



Nelle proprietà del Tag Phong (sotto) bisogna attivare l'algoritmo mettendo un segno di spunta accanto a Limite angolo. Il valore dell'angolo è un parametro che serve per indicare all'algoritmo quei valori dell'angolo solido formato da due superfici al di sopra del quale NON deve essere applicata la transizione graduale; altrimenti anche gli angoli retti apparirebbero stonati.

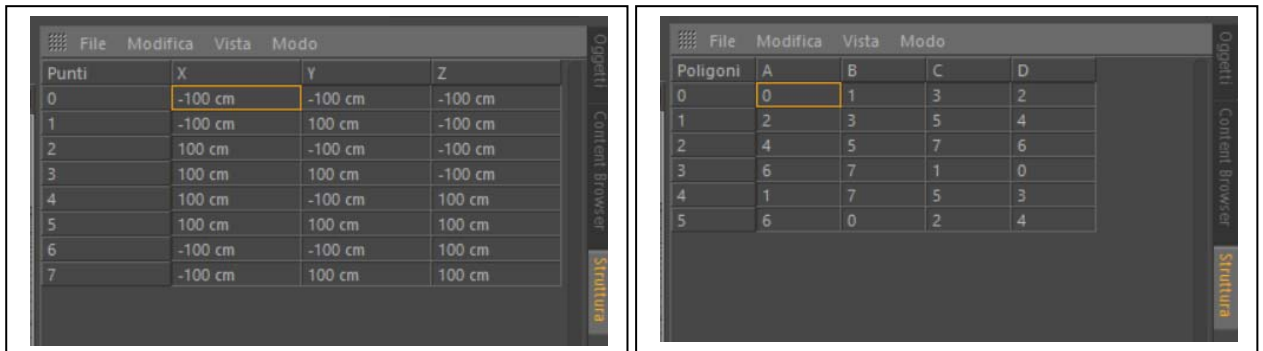
L'operazione va ripetuta per tutti i solidi e, se occorre, l'angolo deve essere aggiustato di volta in volta.

Il risultato di questa operazione è mostrato in basso.



### Concetti essenziali.

1. Per portare un modello dall'ambiente della rappresentazione matematica (fatta di equazioni) all'ambiente della rappresentazione numerica o poligonale (fatta di liste di coordinate e simboli) bisogna esportare il modello in uno dei formati che ne descrivono la tassellazione (stl – stereo litografico adatto anche alle stampanti 3D; wrl o wrml – adatto ad applicazioni di realtà virtuale; obj).
2. In C4D la struttura del modello importato può essere visualizzata aprendo la scheda 'Struttura' che si trova nella Storia del modello sul bordo destro (ultime versioni del software). La struttura è composta da: una lista di coordinate di vertici del poliedro dove ogni vertice è contraddistinto da un numero; una lista di collegamenti tra i vertici, dove ogni faccia è contraddistinta da un numero (vedi sotto). Si passa da una lista all'altra attraverso la funzione 'Modo'.



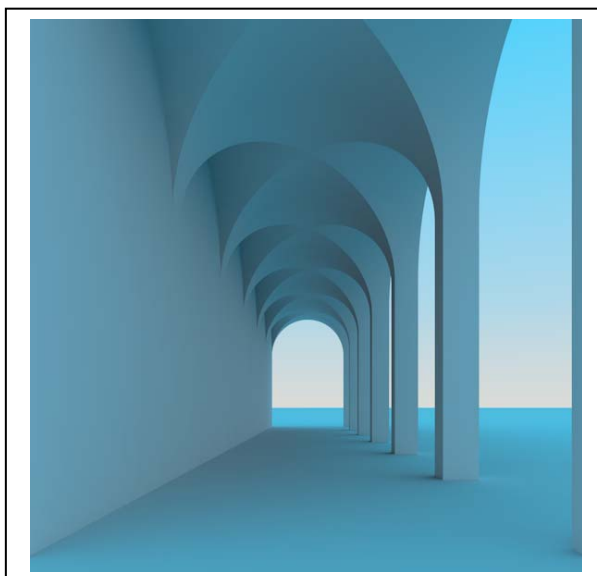
Punti	X	Y	Z
0	-100 cm	-100 cm	-100 cm
1	-100 cm	100 cm	-100 cm
2	100 cm	-100 cm	-100 cm
3	100 cm	100 cm	-100 cm
4	100 cm	-100 cm	100 cm
5	100 cm	100 cm	100 cm
6	-100 cm	-100 cm	100 cm
7	-100 cm	100 cm	100 cm

Poligoni	A	B	C	D
0	0	1	3	2
1	2	3	5	4
2	4	5	7	6
3	6	7	1	0
4	1	7	5	3
5	6	0	2	4

Nella figura di sinistra si vede la lista delle coordinate dei vertici di un cubo; nella figura a destra la lista degli spigoli che formano le facce (poligoni) del medesimo cubo.

3. La luce naturale del giorno, essenziale per illuminare un modello di architettura sia nelle viste dell'esterno che nelle viste degli interni, è formata da due componenti principali: la luce diffusa del cielo e la luce solare a raggi paralleli. Per simulare la prima, in C4D si usa l'oggetto Cielo, che è una sfera di raggio indefinito collocata intorno all'oggetto: il materiale applicato a questa sfera diffonderà una luce del suo colore e perciò influenzerà il colore degli oggetti.

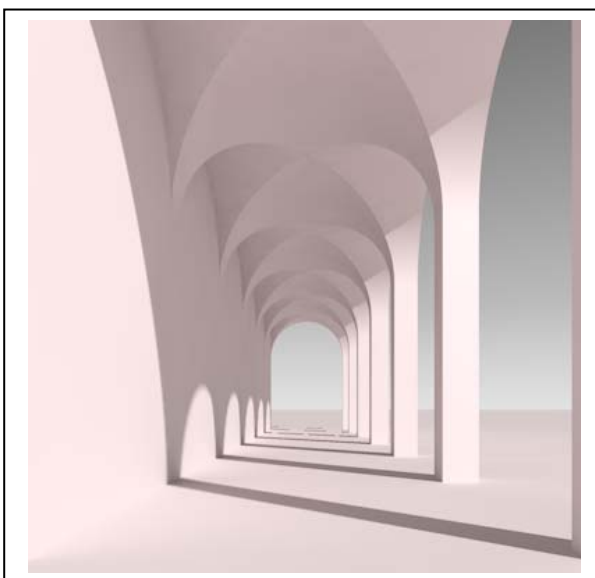


Nella figura a sinistra la scena è illuminata unicamente dalla luce diffusa dal cielo, al quale è stato applicato un gradiente di colore azzurro intenso. Si noti come anche l'architettura è azzurra, nonostante abbia, di per sé un colore grigio caldo. A destra la medesima scena è illuminata da un cielo da un bianco al quale è applicato un gradiente che va dal bianco al grigio freddo.

4. La luce del Sole può essere simulata con un oggetto che emette luce a raggi paralleli. Ai fini della resa della luce naturale in architettura sono essenziali i riflessi, vale a dire la luce emessa dai corpi illuminati. Per rendere i riflessi bisogna servirsi di algoritmi che vengono oggi raccolti nella classe GI (Global Illumination). Nelle ultime versioni C4D si distinguono due ‘Metodi’: primario e secondario. L’algoritmo applicato al primo metodo si occupa della luce diretta (sia essa del sole come diffusa dal cielo) mentre l’algoritmo applicato al secondo metodo si occupa della luce riflessa. Il parametro ‘profondità diffusione’ regola il numero di rimbalzi della luce dei quali si chiede al calcolo di tenere conto.



Nella figura a sinistra la scena è illuminata senza GI e perciò i riflessi non sono stati calcolati. Nella figura a destra la scena è illuminata con la GI, ma escludendo il Metodo secondario. Nelle figure qui sotto è stato applicato il metodo secondario con Profondità diffusione 2 e 8. Si noti come, all’aumentare del numero delle riflessioni, non solo aumenta la luminosità dell’intradosso ma anche la componente cromatica che la luce riflessa porta con sé: il pavimento è di un colore caldo, colore che finisce col diventare sempre più intenso mentre si aggiunge alle superfici dell’intradosso.



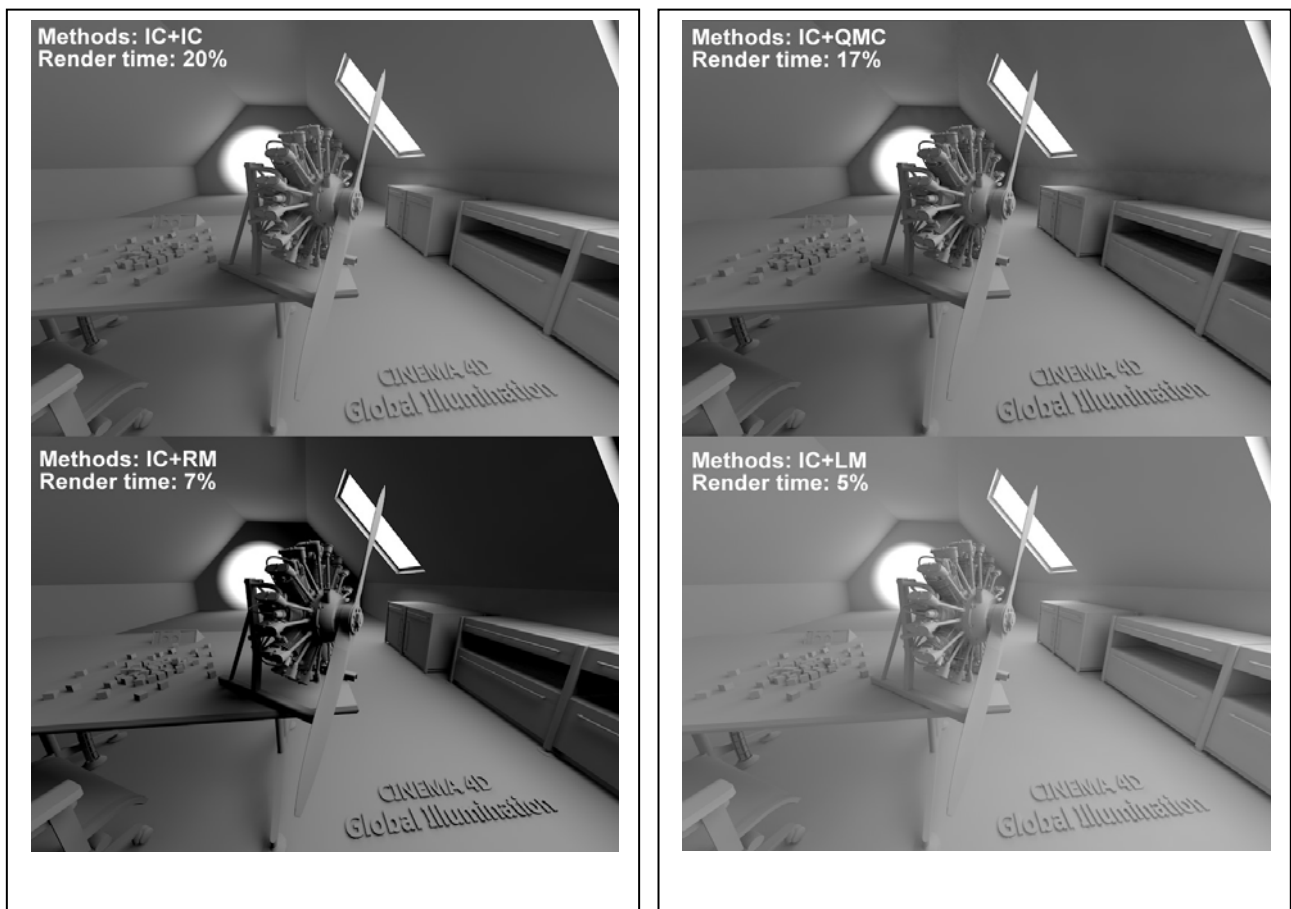
Per quanto riguarda gli Algoritmi di calcolo (Irradiance Cache, Quasi-Monte-Carlo, Radiosity Maps, Light Mapping) conviene consultare la pagina della guida che riporto qui sotto e scegliere anche tenendo conto dei tempi di calcolo.

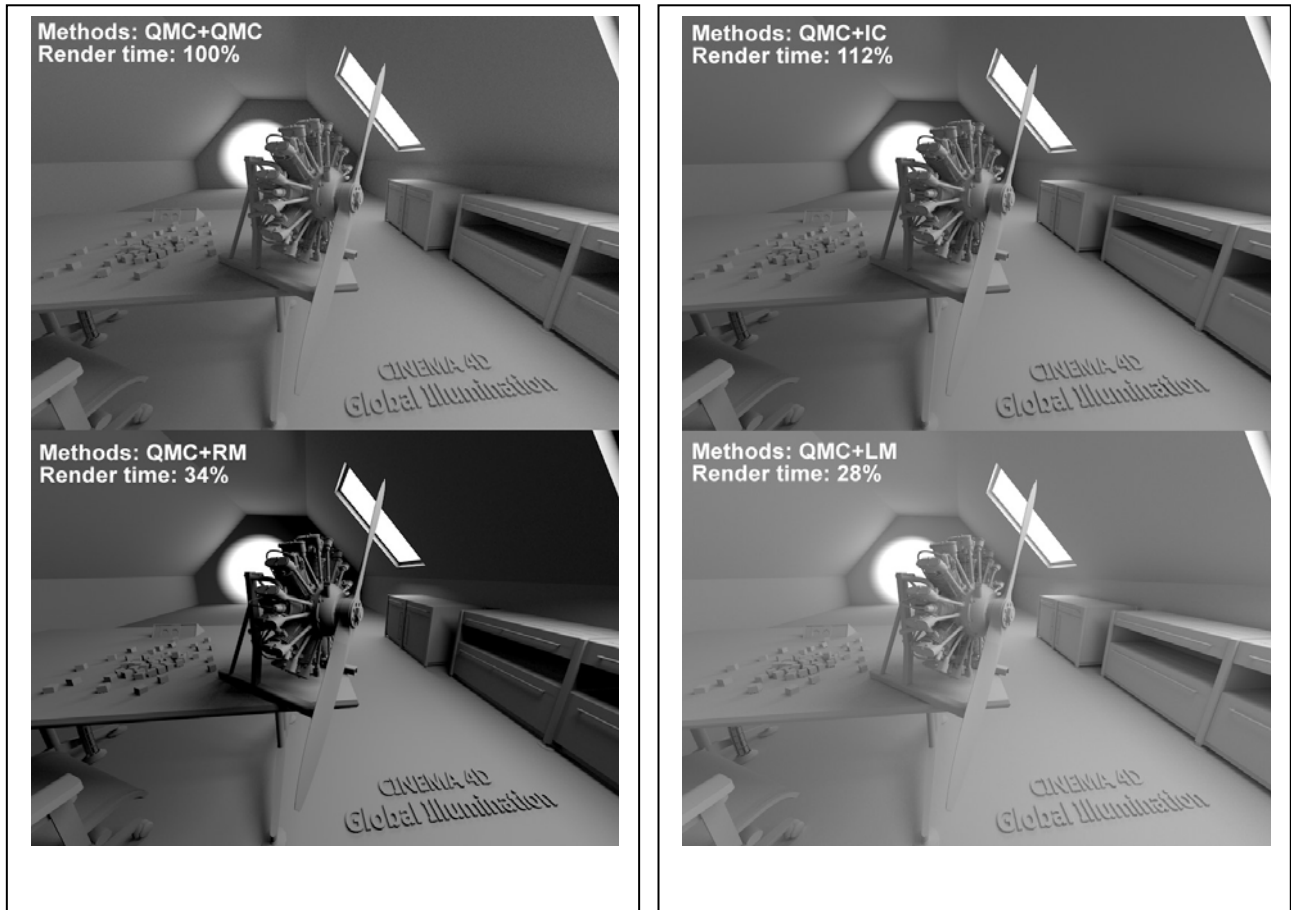
### Sample Renderings

Here you will find several sample renderings with various combinations of GI methods.

The following project is illuminated by a sky through two windows set up as GI portals and a CINEMA 4D Spot light on the back wall. The only variances are the GI methods, whereby the remaining settings reflect median quality settings. Some renderings are spotty, which is something that can be improved with higher sampling.

Note that the render times are only indicators and can vary for other projects (the render time will also be reduced dramatically for IC+LM and QMC+LM if Build Radiosity Maps is enabled). 100% render time is the most precise method for QMC+QMC.





Compare the methods and note the following:

- How much faster the simplified Irradiance Cache is compared to QMC (as the primary method) for similar results.
- Interior spaces are rendered comparatively bright due to numerous light reflections when using light maps whereas ...
- ... the remaining secondary methods are darker even though they allow a maximum of 8 reflections and render longer.
- The precise shadows with QMC as primary method.
- How IC+QMC is prone to flickering.

5. **L'algoritmo di Phong.** I metodi suddetti consentono oggi un grande realismo nella simulazione della luce e dei suoi effetti sui corpi, vale a dire del chiaroscuro. Tuttavia assai poco potrebbero se, nel Giugno del 1975, un ricercatore americano, Bui Tuong Phong, non avesse pubblicato la memoria che riporto qui in allegato, nella quale propone di interpolare le normali alle superfici dei poliedri illuminati e di distribuire, di conseguenza, il grado di intensità luminosa frutto del calcolo tra le superfici limitrofe. In tal modo, ad esempio, una sfera apparirà rotonda e liscia anche se, in realtà, è formata da un numero finito di facce.

---

# Illumination for Computer Generated Pictures

Bui Tuong Phong  
University of Utah

---

**The quality of computer generated images of three-dimensional scenes depends on the shading technique used to paint the objects on the cathode-ray tube screen. The shading algorithm itself depends in part on the method for modeling the object, which also determines the hidden surface algorithm. The various methods of object modeling, shading, and hidden surface removal are thus strongly interconnected. Several shading techniques corresponding to different methods of object modeling and the related hidden surface algorithms are presented here. Human visual perception and the fundamental laws of optics are considered in the development of a shading rule that provides better quality and increased realism in generated images.**

**Key Words and Phrases:** computer graphics, graphic display, shading, hidden surface removal.

**CR Categories:** 3.26, 3.41, 8.2

## Introduction

This paper describes several approaches to the production of shaded pictures of solid objects. In the past decade, we have witnessed the development of a number of systems for the rendering of solid objects by computer. The two principal problems encountered in the design of these systems are the elimination of the hidden

---

Copyright © 1975, Association for Computing Machinery, Inc. General permission to republish, but not for profit, all or part of this material is granted provided that ACM's copyright notice is given and that reference is made to the publication, to its date of issue, and to the fact that reprinting privileges were granted by permission of the Association for Computing Machinery.

This research was supported in part by the University of Utah Computer Science Division and the Advanced Research Projects Agency of the U.S. Department of Defense, monitored by the Rome Air Development Center, Griffiss Air Force Base, NY 13440, under Contract F30602-70-C-0300. Author's address: Digital Systems Laboratory, Stanford University, Stanford, CA 94305.

parts and the shading of the objects. Until now, most effort has been spent in the search for fast hidden surface removal algorithms. With the development of these algorithms, the programs that produce pictures are becoming remarkably fast, and we may now turn to the search for algorithms to enhance the quality of these pictures.

In trying to improve the quality of the synthetic images, we do not expect to be able to display the object exactly as it would appear in reality, with texture, over-cast shadows, etc. We hope only to display an image that approximates the real object closely enough to provide a certain degree of realism. This involves some understanding of the fundamental properties of the human visual system. Unlike a photograph of a real world scene, a computer generated shaded picture is made from a numerical model, which is stored in the computer as an objective description. When an image is then generated from this model, the human visual system makes the final subjective analysis. Obtaining a close image correspondence to the eye's subjective interpretation of the real object is then the goal. The computer system can be compared to an artist who paints an object from its description and not from direct observation of the object. But unlike the artist, who can correct the painting if it does not look right to him, the computer that generates the picture does not receive feedback about the quality of the synthetic images, because the human visual system is the final receptor.

This is a subjective domain. We must at the outset define the degree of realism we wish to attain, and fix certain goals to be accomplished. Among these goals are:

1. "Real time" display of dynamic color pictures of three-dimensional objects. A real time display system is one capable of generating pictures at the rate of at least 30 frames a second.
2. Representation of objects made of smooth curved surfaces.
3. Elimination or attenuation of the effects of digital sampling techniques.

The most important consideration in trying to attain these goals is the object modeling technique.

## Existing Shading Techniques

### Methods of Object Modeling

Image quality depends directly on the effectiveness of the shading algorithm, which in turn depends on the method of modeling the object. Two principal methods of object description are commonly used:

1. Surface definition using mathematical equations.
2. Surface approximation by planar polygonal mosaic.

Several systems have been implemented to remove hidden parts for mathematically defined curved surfaces [1, 2, 3, 4, 5]. With these systems, exact information at each point of the surface can be obtained, and the result-



ing computer generated pictures are most realistic. The class of possible surfaces is restricted, however, and the computation time needed to remove the hidden parts and to perform shading is very large. Up to the present time, these systems have usually considered the class of surfaces represented by quadric patches. Although higher degree surfaces are desirable and are sometimes necessary to model an object, they have not been taken into consideration due to an increase in computation time to remove hidden surfaces and to perform shading computations. Even when only quadric surfaces are considered, the implementation of a real time display system using this type of model is too expensive and complex.

A simple method of representing curved surfaces and objects of arbitrary shape is to approximate the surfaces with small planar polygons; for example, a cone might be represented as shown in Figure 1. This type of representation has the advantage that it avoids the problem, posed by mathematically curved surface approaches, of solving higher order equations.

Planar approximation also offers the only means of reducing hidden surface computation to within reasonable bounds, without restricting the class of surfaces that can be represented. For this reason, all recent attempts to devise fast hidden surface algorithms have been based on the use of this approximation for curved surfaces; these algorithms have been summarized and classified by Sutherland et al. [6]. The next section discusses their influence on the way shading is computed.

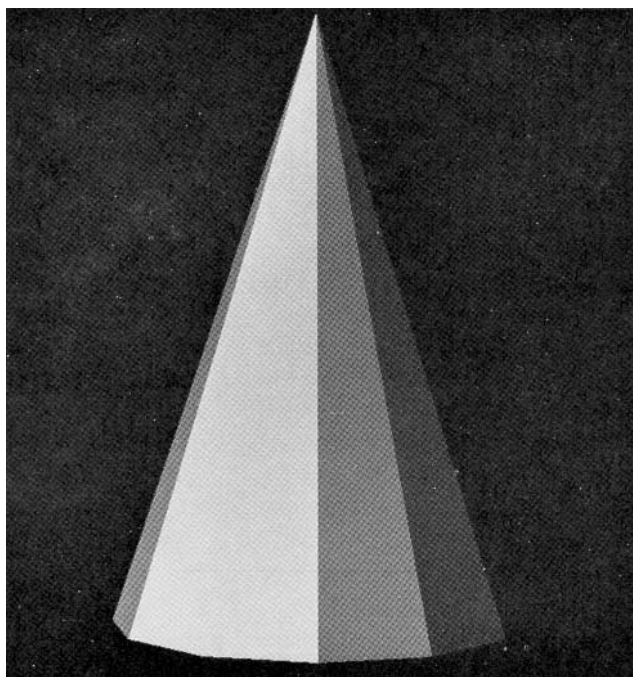
While planar approximation greatly simplifies hidden surface removal, it introduces several major problems in the generation of a realistic displayed image. One of these is the *contour edge* problem: the outline or silhouette of a polygonally approximated object is itself a polygon, not a smooth curve. The other problem is that of shading the polygons in a realistic manner. This paper is concerned with the shading problem; the contour edge problem is discussed by the author and F.C. Crow in [7].

### Influence of Hidden Surface Algorithms

The order in which a hidden surface algorithm computes visible information has a decided influence on the way shading is performed. For example Warnock, who developed one of the first such algorithms [8], computed display data by a binary subdivision process: this meant that the order of generating display data was largely independent both of the order of scanning the display and of the order of the polygons in memory. This made it difficult to perform effective shading on curved objects.

The two major advances in the development of fast hidden surface algorithms have been made by Watkins [9] and by Newell, Newell, and Sancha [10]. Watkins generates the displayed picture scan line by scan line. On each scan line he computes which polygons intersect the scan line, and then computes the visible *segment* of each polygon, where this segment is the visible strip of

Fig. 1. A cone represented by means of planar approximation.



the polygon, one screen resolution unit in height, that lies on the scan line.

Newell, Newell, and Sancha adopt a different approach, using a *frame buffer* into which the object is painted, face by face. The hidden surface problem is solved by painting the farthest face first, and the nearest last. Each face is painted scan line by scan line, starting at the top of the face.

From the shading aspect, the important attribute of these algorithms is that they both generate information scan line by scan line in order to display the faces of an object. This information is in the form of segments, one screen resolution unit high, on which the shading computation may then be performed. The main differences between the algorithms, from the point of view of shading, are (a) the order in which the segments are generated, and (b) the fact that Watkins generates each screen dot only once, whereas the Newell-Sancha algorithm may overwrite the same dot several times.

### Shading with the Polyhedral Model

When planar polygons are used to model an object, it is customary to shade the object by using the *normal vectors* to the polygons. The shading of each point on a polygon is then the product of a shading coefficient for the polygon and the cosine of the angle between the polygon normal and the direction of incident light. This cosine relationship is known in optics as the "cosine law," and allows us to compute the shading  $S_p$  for a polygon  $p$  as

$$S_p = C_p \cos(i), \quad (1)$$

where  $C_p$  is the reflection coefficient of the material of  $p$  relative to the incident wavelength, and  $i$  is the incident angle.

Fig. 2. An example of the use of Newell, Newell, and Sancha's shading technique, showing transparency and highlight effects.

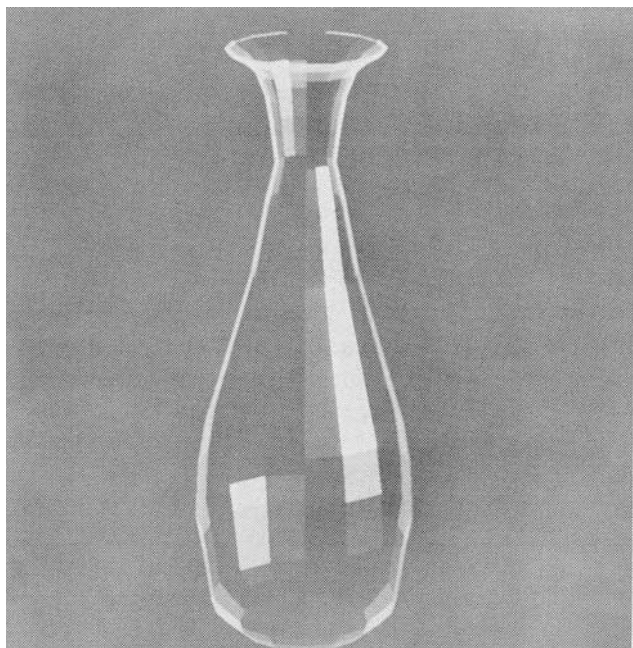


Fig. 3. Computation of the shading at point R using the Gouraud method. There are two successive linear interpolations: (1) across polygon edges, i.e. P between A and B, Q between A and D; and (2) along the scan line, i.e. R between P and Q.

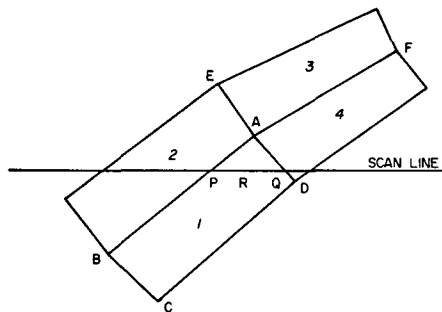
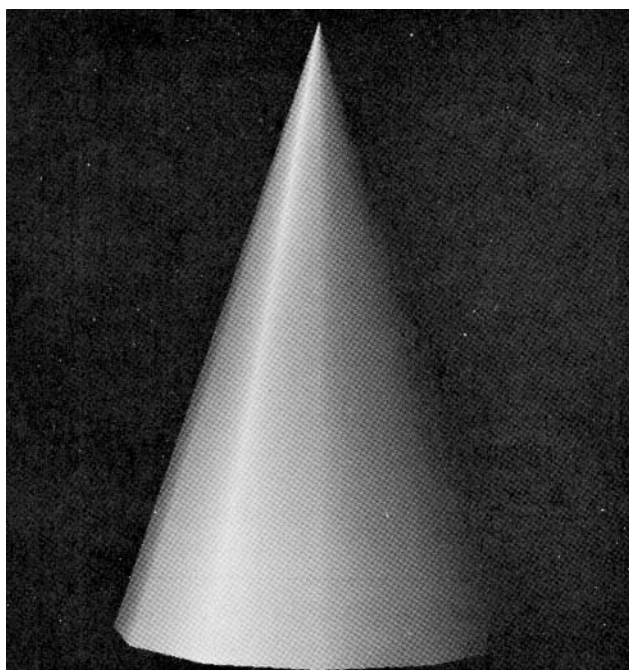


Fig. 4. Gouraud shading, applied to approximated cone of Fig. 1.



This shading offers only a very rough approximation of the true physical effect. It does not allow for any of the *specular* properties of the material, i.e. the ability of the material to generate highlights by reflection from its outer surface, and the position of the observer, which is ignored. A more serious drawback to this method, however, is the poor effect when using it to display smooth curved surfaces. The cosine law rule is appropriate for objects that are properly modeled with planar surfaces, such as boxes, buildings, etc., but it is inappropriate for smoothly curved surfaces such as automobile bodies. This does not mean, however, that we should abandon the use of such a polygon-oriented shading rule and search for a different rule for curved surfaces. Recent research in shading techniques demonstrates that significant results can be achieved by using the basic shading rule of eq. (1) and modifying the results to reduce the discontinuities in shading between adjacent polygons.

**1. Warnock's shading.** As three-dimensional objects are projected onto the cathode-ray tube screen, the depth sensation is lost, and the images of those objects appear flat. In order to restore the depth sensation, two effects were simulated by Warnock:

1. Decreasing intensity of the reflected light from the object with the distance between the light source and the object.

2. Highlights created by specular reflection.

Warnock placed the light source and the eye at the same position, so that the shading function was the sum of two terms, one for the normal "cosine" law, and the other term for the specularly reflected light. The resulting pictures have several desirable attributes; for example, identical parallel faces, located differently in space, will be shaded at different intensities, and facets which face directly toward the light source are brighter than adjacent facets facing slightly away from the incident light. However, the polygonal model gives a discontinuity in shading between faces of an approximated curved surface. When a curved surface is displayed, the smoothness of the curved surface is destroyed by this discontinuity. This is clearly visible in Figure 1.

**2. Newell, Newell, and Sancha's shading.** Newell, Newell, and Sancha presented some ideas on creating transparency and highlights. From observations in the real world, they found that highlights are created not only by the incident light source but also by the reflection of light from other objects in the scene; this is especially true in the case of objects made of highly reflective or transparent materials. In the Newell-Sancha model, curved surfaces are approximated with planar polygons. Unfortunately, the ability to generate highlights is severely limited due to the inability to vary light intensity over the surface of any single polygon. This problem is apparent in Figure 2.

**3. Gouraud's shading.** While working on a technique to represent curved objects made of "Coons surfaces"



or "Bezier patches," Gouraud [11] developed an algorithm to shade curved surfaces. With his algorithm, a surface represented by a patch is approximated by polygonal planar facets. Gouraud computes information about the curvature of the surface at each vertex of each of these facets. From the curvature, a shade intensity is computed and retained. For example, the shade intensity may be computed for each vertex using eq. (1), with  $i$  as the angle between the incident light and the normal to the surface at this vertex. When the surface is displayed, this shade intensity is linearly interpolated along the edge between adjacent pairs of vertices of the object. The shade at a point on the surface is also a linear interpolation of the shade along a scan line between intersections of the edges with a plane passing through the scan line (Figure 3). This very simple method gives a continuous gradation of shade over the entire surface, which in most cases restores the smooth appearance. An example of Gouraud's shading is shown in Figure 4.

With the introduction of the Gouraud smooth shading technique, the quality of computer-generated images improved sufficiently to allow representation of a large variety of objects with great realism. Problems still exist, however, one of which is the apparent discontinuity across polygon edges. On surfaces with a high component of specular reflection, highlights are often inappropriately shaped, since they depend upon the disposition and shape of the polygons used to approximate a curved surface and not upon the curvature of the object surface itself. The shading of a surface in motion (in a computer generated film) has annoying frame to frame discontinuities due to the changing orientation of the polygons describing the surface. Also the shading algorithms are not invariant under rotation.

Frame-to-frame discontinuities of shade in a computer generated film are illustrated in the following situation. A curved surface is approximated with planar facets. When this surface is in motion, all the facets which are perpendicular to the direction of the light take on a uniform shade. In the next frame the motion of the object brings these facets into a different orientation toward the light, and the intensity of the shade across their surfaces varies continuously from one end to the other. Thus the surface appears to change from one with highlights to one of uniform shade. Moreover, the position of these highlights is not steady from frame to frame as the object rotates.

**Mach Band Effect**

Many of the shading problems associated with planar approximation of curved surfaces are the result of the discontinuities at polygon boundaries. One might expect that these problems could be avoided by reducing the size of the polygons. This would be undesirable, of course, since it would increase the number of polygons and hence would increase both the memory requirements for storing the model and the time for hidden surface removal.

Fig. 5. Normal at a point along an edge.

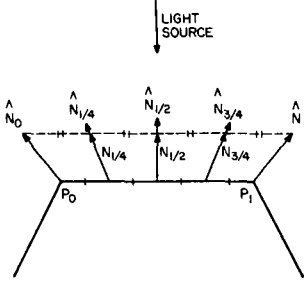
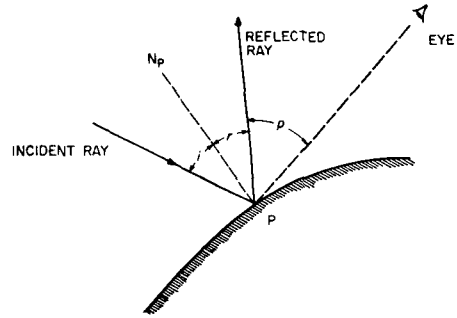


Fig. 6. Shading at a point.



Unfortunately, because of visual perception effects, the reduction of polygon size is not as beneficial as might be expected. The particular effect responsible is the *Mach Band* effect. Mach established the following principle:

Wherever the light-intensity curve of an illuminated surface (the light intensity of which varies in only one direction) has a concave or convex flexion with respect to the axis of the abscissa, that particular place appears brighter or darker, respectively, than its surroundings [E. Mach, 1865].

Whenever the slope of the light intensity curve changes, this effect appears. The extent to which it is noticeable depends upon the magnitude of the curvature change, but the effect itself is always present.

Without the Mach Band effect, one might hope to achieve accurate shading by reducing the size of polygons. Unfortunately the eye enhances the discontinuities over polygon edges, creating undesired areas of apparent brightness along the edges. Therefore unless the size of the displayed facets is shrunk to a resolution point, increasing the number of facets does not solve the problem. Using the Gouraud method to interpolate the shade linearly between vertices, the discontinuities of the shading function disappear, but the Mach Band effect is visible where the slope of the shading function changes. This can be seen in Figure 4. The subjective discontinuity of shade at the edges due to the Mach Band effect then destroys the smooth appearance of the curved surface.

A better shading rule is therefore proposed for displaying curved surfaces described by planar polygons. This new technique requires the computation of the normal to the displayed surface at each point. It is therefore more expensive in computation than Gouraud's technique; but the quality of the resulting picture, and the accuracy of the displayed highlights, is much improved.

**Using a Physical Model**

**Specular Reflection**

If the goal in shading a computer-synthesized image is to simulate a real physical object, then the shading model should in some way imitate real physical shading situations. Clearly the model of eq. (1) does not accomplish this. As mentioned before, it completely

Fig. 7(a). Determination of the reflected light.

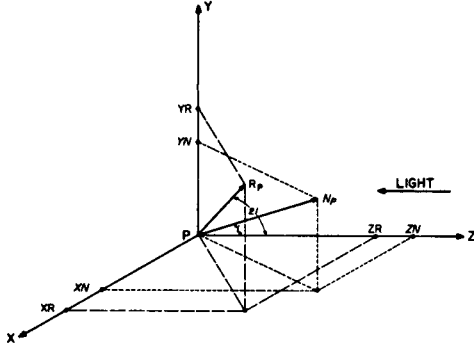
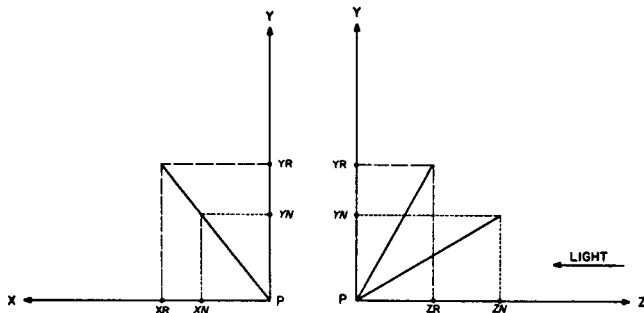


Fig. 7(b). Projections of the reflected light.



ignores both the position of the observer and the specular properties of the object. Even with the improvements introduced by Gouraud, which provide remarkably better shading, these properties are still ignored.

The first step in accounting for the specular properties of objects and the position of the observer is to determine the normal to the surface at each point to be shaded, i.e. at each point where a picture element of the raster display projects onto the surface. It is only with this knowledge that information about the direction of reflected rays can be acquired, and only with this information can we model the specular properties of objects. It is evident from the preceding discussion, however, that our polyhedral model provides information about normals only at the vertices of polygons. Thus the first step in improving our shading model is to devise a way to obtain the normal to the surface for each raster unit.

### Computation of the Normal at a Point on the Surface

The normal at each vertex can be approximated by either one of the methods described by Gouraud [10]. It is now necessary to define the normal to the surface along the edges and at a point on the surface of a polygon.

The normal to the surface at a point along the edge of a polygonal model is the result of a linear interpolation to the normals at the two vertices of that edge. An example is given in Figure 5: the normal  $N_t$  to the surface at a point between the two vertices  $P_0$  and  $P_1$  is computed as follows:

$$N_t = tN_1 + (1-t)N_0, \quad (2)$$

where  $t = 0$  at  $N_0$  and  $t = 1$  at  $N_1$ .

The determination of the normal at a point on the

surface of a polygon is achieved in the same way as the computation of the shading at that point with the Gouraud technique. The normal to the visible surface at a point located between two edges is the linear interpolation of the normals at the intersections of these two edges with a scan plane passing through the point under consideration. Note that the general surface normal is quadratically related to the vertex normal.

From the approximated normal at a point, a shading function determines the shading value at that point.

### The Shading Function Model

In computer graphics, a shading function is defined as a function which yields the intensity value of each point on the body of an object from the characteristics of the light source, the object, and the position of the observer.

Taking into consideration that the light received by the eye is provided one part by the diffuse reflection and one part by the specular reflection of the incident light, the shading at point  $P$  (Figure 6) on an object can be computed as:

$$S_p = C_p[\cos(i)(1-d)+d] + W(i)[\cos(s)]^n, \quad (3)$$

where:

$C_p$  is the reflection coefficient of the object at point  $P$  for a certain wavelength.

$i$  is the incident angle.

$d$  is the environmental diffuse reflection coefficient.

$W(i)$  is a function which gives the ratio of the specular reflected light and the incident light as a function of the incident angle  $i$ .

$s$  is the angle between the direction of the reflected light and the line of sight.

$n$  is a power which models the specular reflected light for each material.

The function  $W(i)$  and the power  $n$  express the specular reflection characteristics of a material. For a highly reflective material, the values of both  $W(i)$  and  $n$  are large. The range of  $W(i)$  is between 10 and 80 percent, and  $n$  varies from 1 to 10. These numbers are empirically adjusted for the picture, and no physical justifications are made. In order to simplify the model, and thereby the computation of the terms  $\cos(i)$  and  $\cos(s)$  of formula (3), it is assumed that:

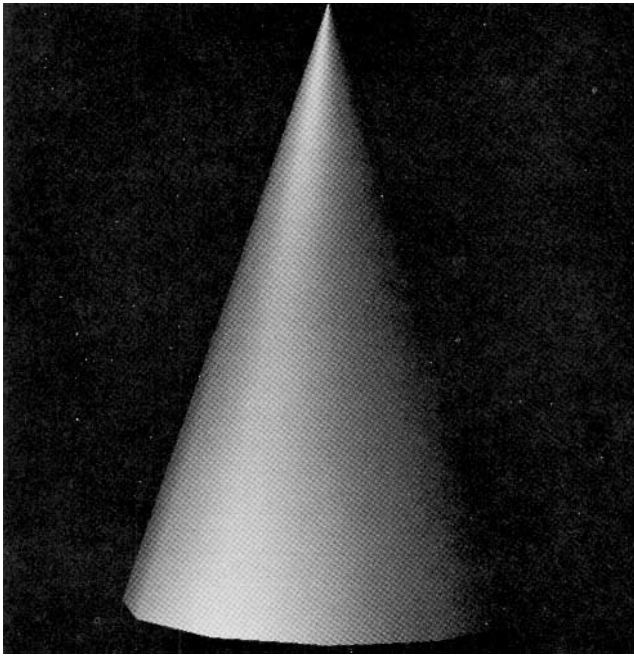
1. The light source is located at infinity; that is, the light rays are parallel.

2. The eye is also removed to infinity.

With these two considerations, the values of  $\cos(i)$  and  $\cos(s)$  of the shading function in (3) can be rewritten as:  $\cos(i) = kN_p / |N_p|$  and  $\cos(s) = uR_p / |R_p|$  where  $k$  and  $u$  are respectively the unit vectors in the direction of the light and the line of sight,  $N_p$  is the normal vector at  $P$ , and  $R_p$  is the reflected light vector at  $P$ .

The quantity  $kN_p / |N_p|$  can be referred to as the projection of a normalized vector  $N_p$  on an axis parallel to the direction of the light. If  $|N_p|$  is unity, the previous

Fig. 8. Improved shading, applied to approximated cone of Fig. 1.



quantity is one component of the vector  $N_p$  in a coordinate system where the direction of light is parallel to one axis. In this case, the quantity  $uR_p / |R_p|$  can be obtained directly from the vector  $N_p$  in the following way.

Let us consider a Cartesian coordinate system having the origin located at point  $P$  and having the  $z$  axis parallel to the light but opposite in direction (Fig. 7(a)).

We have the following assumptions about the model:

1. The normalized vector  $N_p$  makes an angle  $i$  with the  $z$  axis, and the reflected light vector  $R_p$  makes an angle  $2i$  with the same axis.
2. Only incident angles less than or equal to 90 degrees are considered in the shading computation. For a greater angle, this means that the light source is behind the front surface. In the case where a view of the back surface is desired when it is visible, it can be assumed that the normal will always point toward the light source.
3. If  $k$  is the unit vector along the  $PZ$  axis, then by simple geometry, it may be shown that the three vectors  $k$ ,  $N_p$ , and  $R_p$  are coplanar.
4. The two vectors  $N_p$  and  $R_p$  are of unit length.

From assumption (3), the projections of the vectors  $N_p$  and  $R_p$  onto the plane defined by  $(PX, PY)$  are merged into a line segment (Figure 7(b)). Therefore,

$$X_r/Y_r = X_n/Y_n, \quad (4)$$

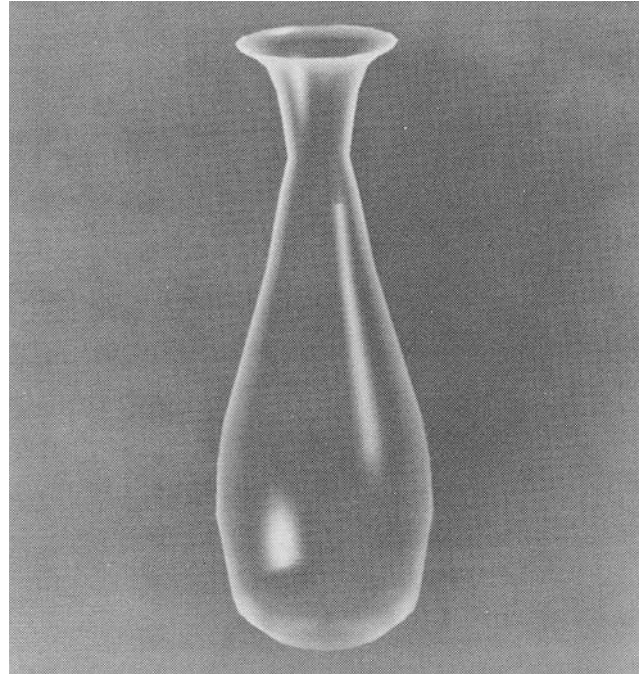
where  $X_r$ ,  $X_n$ ,  $Y_r$ , and  $Y_n$  are respectively the components of  $R_p$  and  $N_p$  in the  $x$  and  $y$  directions.

From assumptions (1) and (2), the component  $Z_n$  of  $N_p$  is:

$$Z_n = \cos(i), \quad (5)$$

where  $0 \leq i \leq 90$  degrees.

Fig. 9. Improved shading, applied to the example of Figure 2.



By simple trigonometry, we obtain the following expressions:

$$Z_r = \cos(2i) = 2[\cos(i)]^2 - 1 = 2Z_n^2 - 1, \quad (6)$$

$$X_r^2 + Y_r^2 = [\sin(2i)]^2 = 1 - [\cos(2i)]^2. \quad (7)$$

From (4) and (7), we obtain:

$$X_r = 2Z_n X_n, \quad Y_r = 2Z_n Y_n, \quad 0 \leq Z_n \leq 1.$$

The three components of  $R_p$  are then known in the light source coordinate system. The projection of the vector  $R_p$  onto the  $z$ -axis of the eye coordinate system may be found by a simple dot product of the reflected vector with this  $z$ -axis. The component of  $R_p$  on an axis parallel to the line of sight is the value of the cosine of the angle between the reflected light and the line of sight. The value of this cosine will be used in the simulation of the specular reflection.

This method of calculating the direction of the reflected light for each point from the orientation of the normal is preferred over the computation of the reflected light vector at vertices and the subsequent interpolation of them in the same way as the normal. It is faster and it requires less storage space than the interpolation scheme.

With the described method, the shading of a point is computed from the orientation of the approximated normal; it is not a linear interpolation of the shading values at the vertices. Therefore, a better approximation of the curvature of the surface is obtained, and highlights due to the simulation of specular reflection are properly rendered. Examples of application of the shading technique are shown in Figures 8 and 9. Figure 10 compares a display generated by this technique with a photograph of a real object.

Fig. 10(a). A sphere displayed with the improved shading.

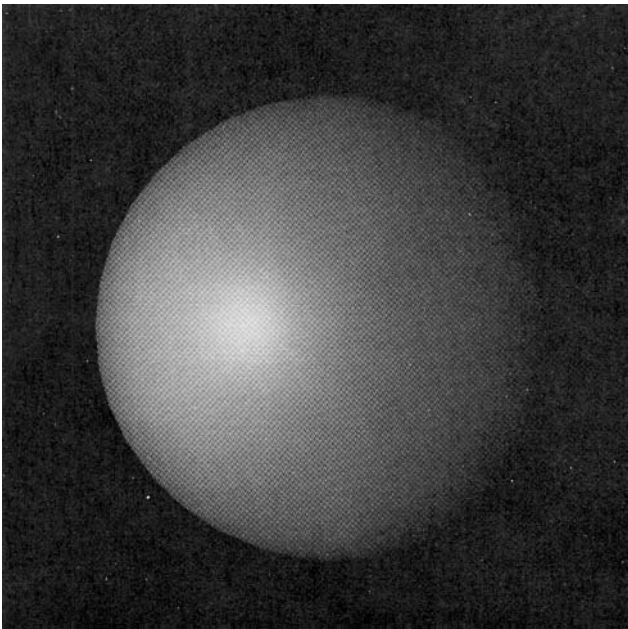
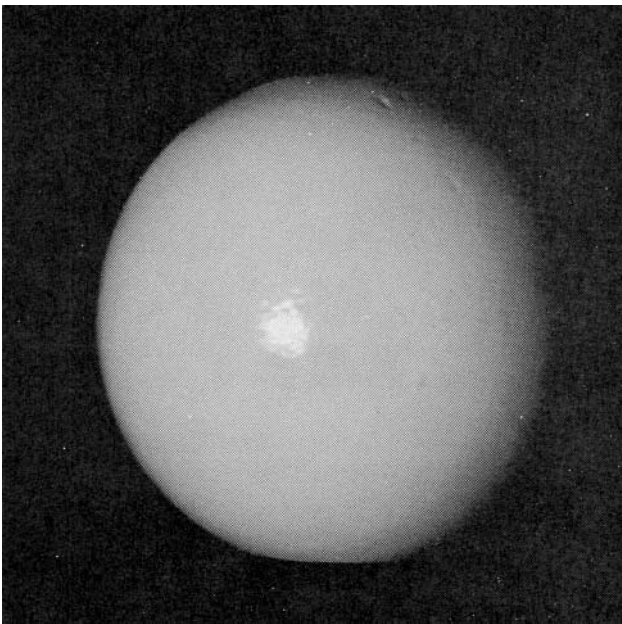


Fig. 10(b). A photograph of a real sphere.



## Conclusion

The linear interpolation scheme used here to approximate the orientation of the normal does not guarantee a continuous first derivative of the shading function across an edge of a polygonal model. In extreme cases where there is an abrupt change in the orientation of two adjacent polygons along a common edge, the subjective brightness due to the Mach Band effect will be visible along this edge. However, this effect is much less visible in the described model than in the Gouraud smooth shading model. Also, an interesting fact discussed previously on Mach Band effect shows

that this effect is visible whenever there is a great change in the slope of the intensity distribution curve, even if the curve has a continuous first derivative. When a higher degree interpolation curve is used, it will make the presence of the edges unnoticeable, although it will still give some Mach Band effect.

When a comparison was made of pictures of the same object generated with different shading techniques, it was found that little difference existed between pictures generated with the new shading and the ones created with a cubic interpolant curve for the shading computation. Furthermore, as time is the critical factor in a real time dynamic picture display system, the use of a high degree interpolation curve does not seem to be possible at the moment with the current techniques to compute the coefficients of such a function.

A hardware implementation of this shading model would of course require more hardware than the simpler Gouraud method. The Gouraud model needs one interpolator for the shading function. It must compute a new shading value for each raster unit, and hence must be very high speed to drive a real time display. The model proposed here requires three of these interpolators operating in parallel. In addition, since the results of the interpolation do not yield a unit vector, and since eqs. (6), (7), and (8) require a unit normal vector, some extra hardware is necessary to "normalize" the outputs of the interpolators. This requires a very fast mechanism for obtaining square roots. None of these problems is too difficult to solve; and judging from the improvements in image quality obtained using the new model, it may well be worth the extra expense to provide such hardware in applications for which real time display is important.

Received November 1975; revised March 1975

## References

1. MAGI, Mathematical Applications Group Inc. 3-D simulated graphics. *Datamation* 14 (Feb. 1968), 69.
2. Comba, P.G. A procedure of detecting intersections of three-dimensional objects. Rep. 39,020, IBM New York Scientific Center, Jan. 1967.
3. Weiss, R.A. BE VISION, a package of IBM 7090 FORTRAN programs to draw orthographic views of combinations of plane and quadric surfaces. *J. ACM* 13, 2 (Apr. 1966), 194-204.
4. Mahl, R. Visible surface algorithm for quadric patches. *IEEE Trans. C-21*, (Jan. 1972), 1-4.
5. Catmull, E.E. A subdivision algorithm for computer display of curved surfaces. Ph.D th., Dep. of Comput. Sci., U. of Utah.
6. Sutherland, I.E., Sproull, R.F., and Schumacker, R.A. A characterization of ten-hidden surface algorithms. *Computing Surveys* 6 (Mar. 1974), 1-56.
7. Bui Tuong Phong and Crow, F.C. Improved rendition of polygonal models of curved surfaces. To be presented at the joint USA-Japan Computer Conference.
8. Warnock, J.E. A hidden-line algorithm for halftone picture representation. Dep. of Comput. Sci., U. of Utah, TR 4-15, 1969.
9. Watkins, G.S. A real-time visible surface algorithm. Dep. of Comput. Sci., U. of Utah, UTEC-CSc-70-101, June 1970.
10. Newell, M.E., Newell, R.G., and Sancha, T.L. A new approach to the shaded picture problem. Proc. ACM 1973 Nat. Conf.
11. Gouraud, H. Computer display of curved surfaces. Dep. of Comput. Sci., U. of Utah, UTEC-CSc-71-113, June 1971. Also in *IEEE Trans. C-20* (June 1971), 623-629.