

マルチメディア工学

コンピュータグラフィックス
ボリュームグラフィックス

佐藤 嘉伸

大阪大学 大学院医学系研究科
放射線統合医学講座

yoshi@image.med.osaka-u.ac.jp

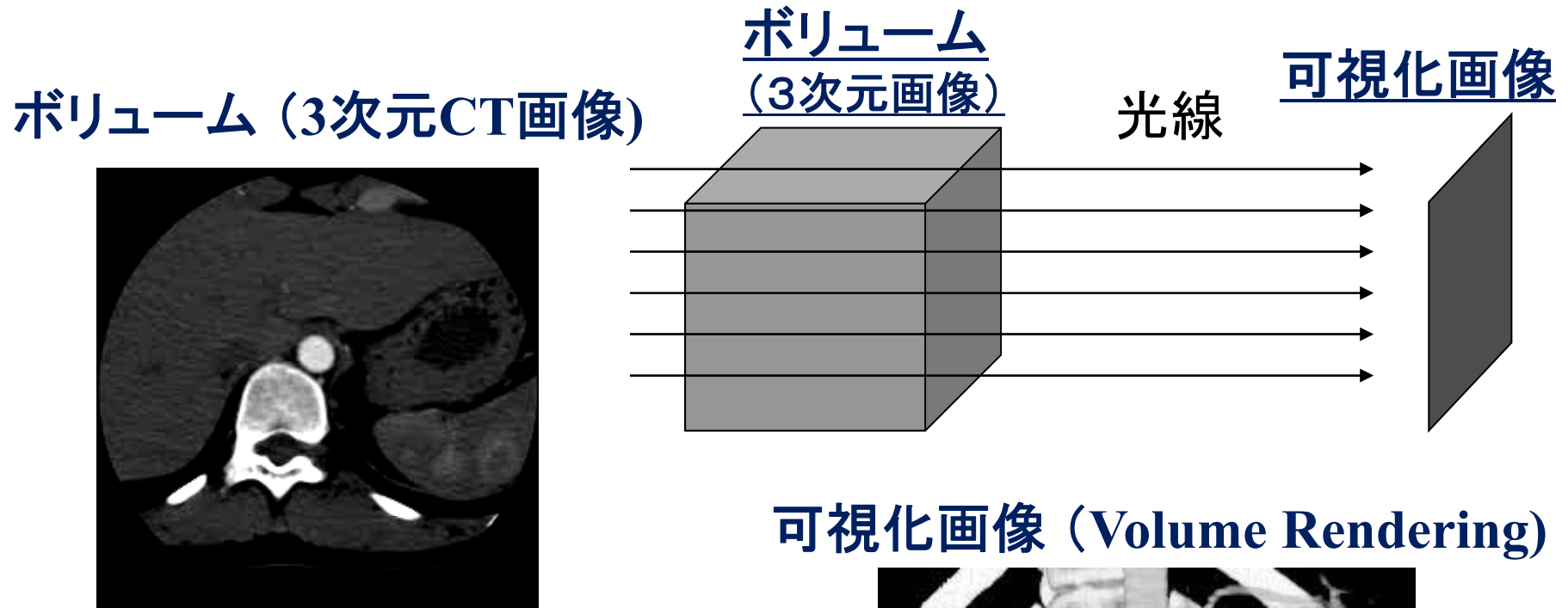
<http://www.image.med.osaka-u.ac.jp/member/yoshi/>

講義ホームページ: 日本語ページ → 授業の資料 → マルチメディア工学

CG (Computer Graphics)

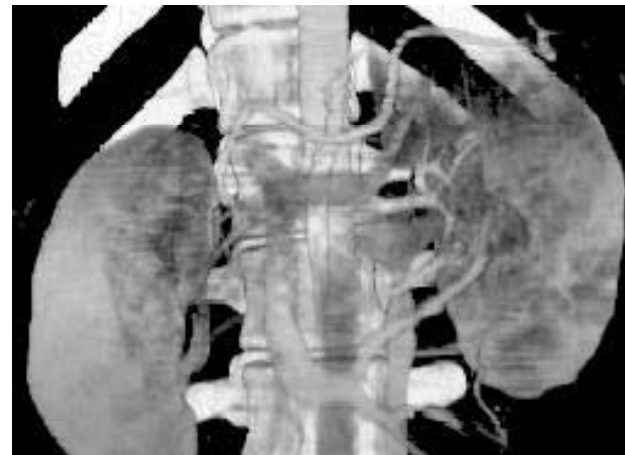
**ボリユームグラフィックス
(Volume Graphics: VG)**

ボリュームグラフィックス

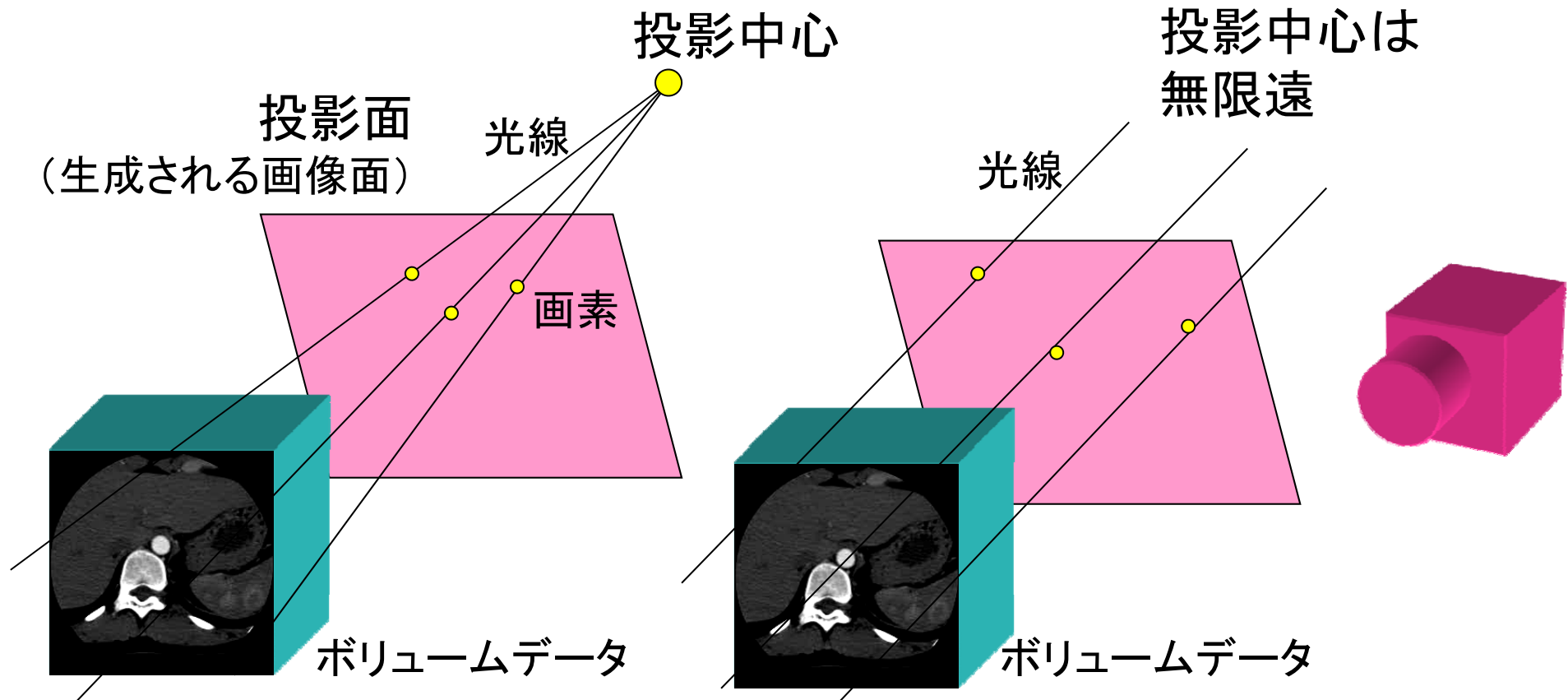


Voxel

可視化画像 (Volume Rendering)



ボリュームデータの2次元画像への投影



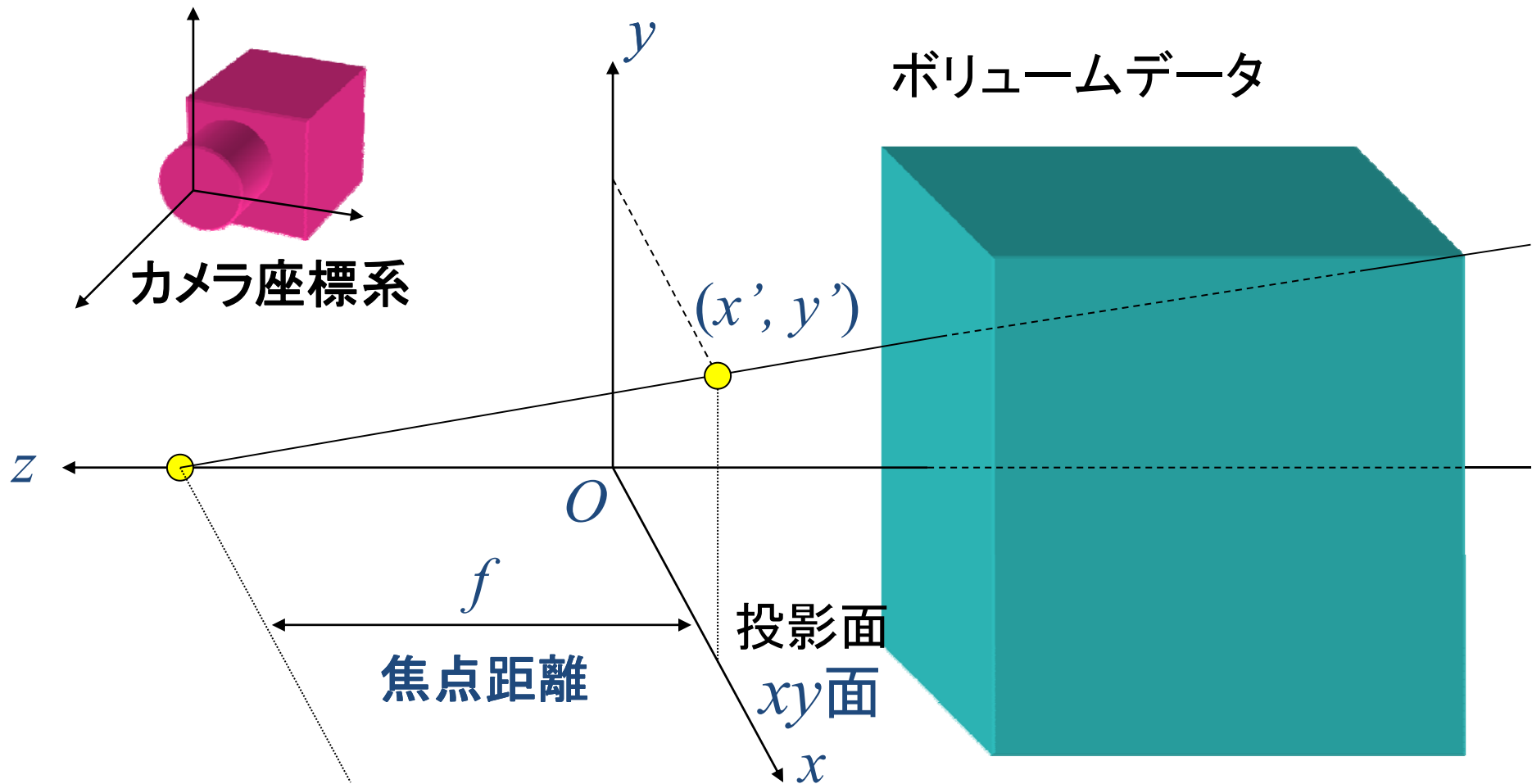
中心投影(透視投影)

平行投影(直交投影)

ボリュームグラフィックスの基本処理: 生成される画像の各画素は、3次元空間中の光線に対応する。ボリュームグラフィックスでは、光線に沿ったボリュームデータのボクセル値プロファイルに基づき、生成される画像の画素値を決定する。

ボリュームデータの2次元画像への投影

中心投影(透視投影)



ボリュームグラフィックスの基本処理: 生成される画像の各画素は、3次元空間中の光線に対応する。ボリュームグラフィックスでは、光線に沿ったボリュームデータのボクセル値プロファイルに基づき、生成される画像の画素値を決定する。

ボリュームグラフィックス法の分類

Maximum Intensity Projection (MIP)

最大値投影 (MIP)

極大値投影 (LMIP: Local MIP)

Surface Rendering (SR)

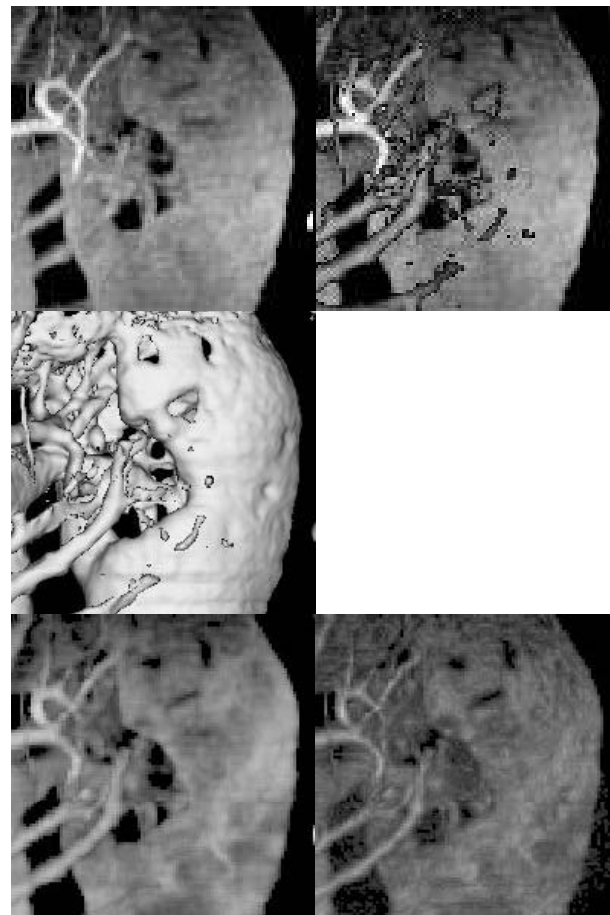
表面の陰影表示

Volume Rendering (VR)

ボリュームの透過表示

陰影なし (Unshaded)

陰影つき (Shaded)

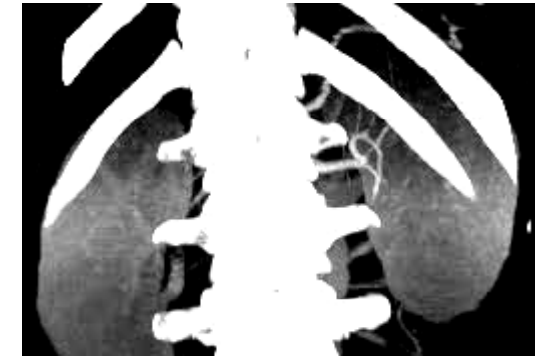


ボリュームデータ可視化法の比較

	SR	VR Unshaded	VR Shaded	MIP	LMIP
濃淡	×	○	△	◎	◎
形状	◎	○	○	△	△
空間関係	◎	○	○	×	◎
客観性	○	△	△	◎	○
多様性	△	○	◎	×	×

客観性: しきい値などへの依存性

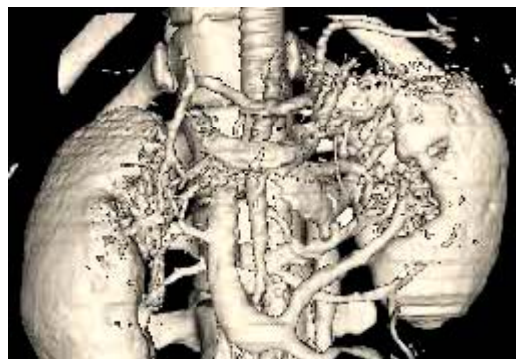
多様性: 多彩な描画表現能力



MIP



LMIP



SR

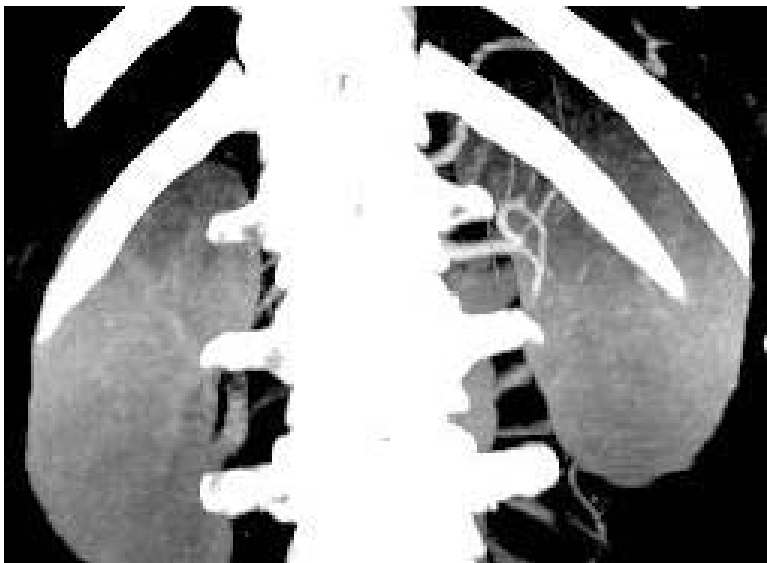


VR Unshaded



VR Shaded

Maximum Intensity Projection (MIP)



MIP



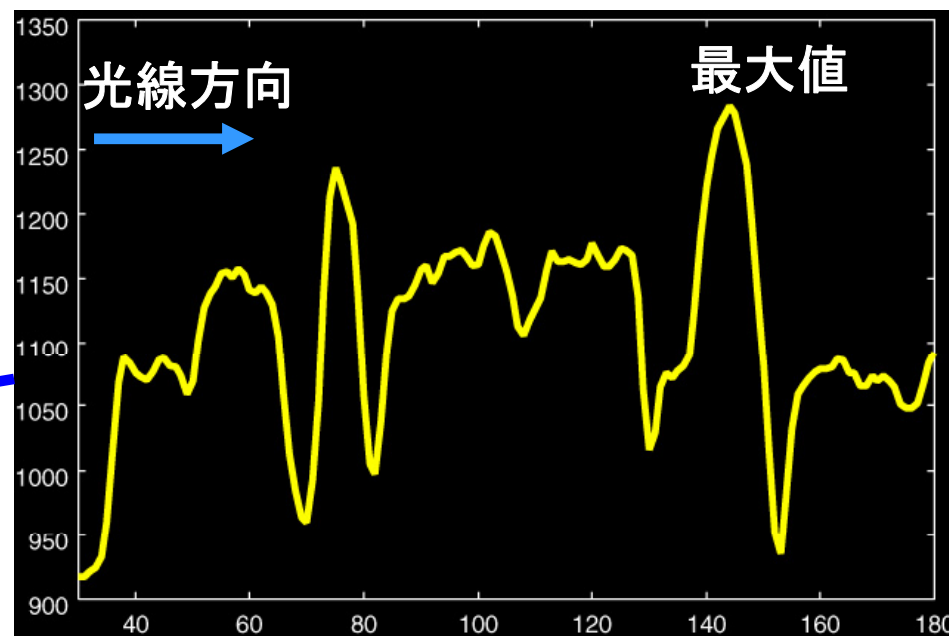
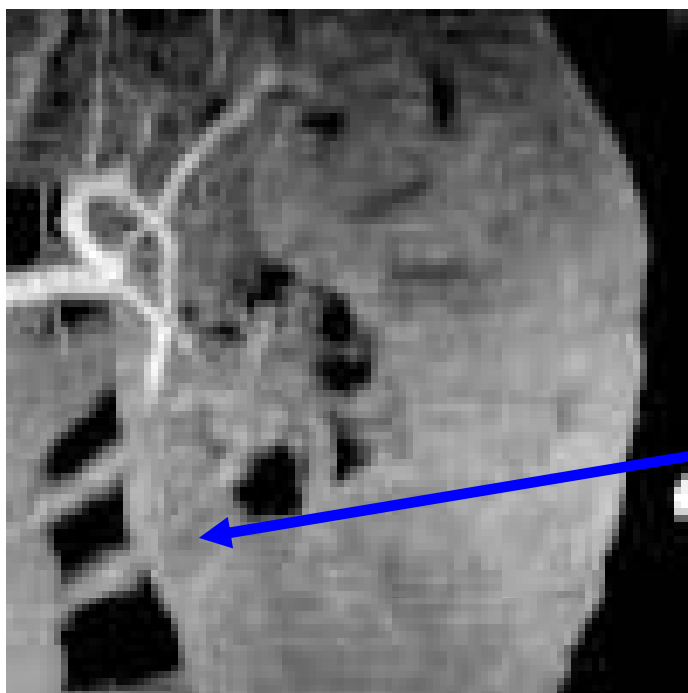
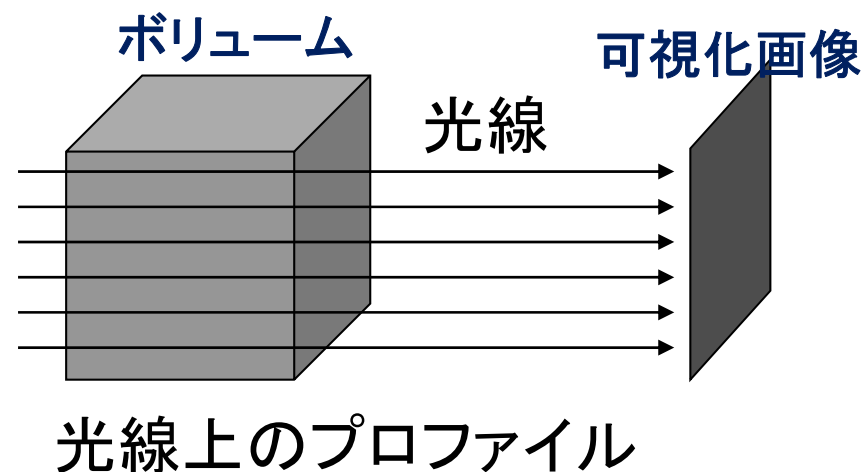
LMIP

MIP: Maximum Intensity Projection

ボリューム
(3次元画像)

↓ 光線上の最大値を選択

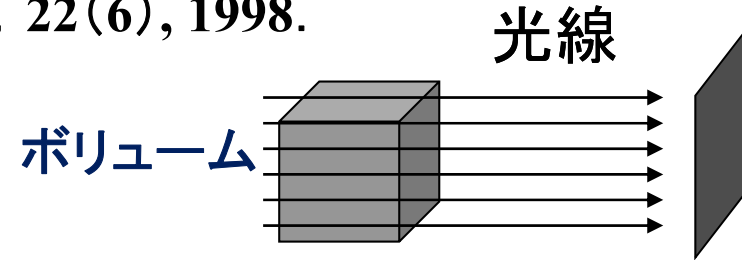
可視化画像



LMIP: Local Maximum Intensity Projection

Y. Sato, et al., *JCAT*, 22(6), 1998.

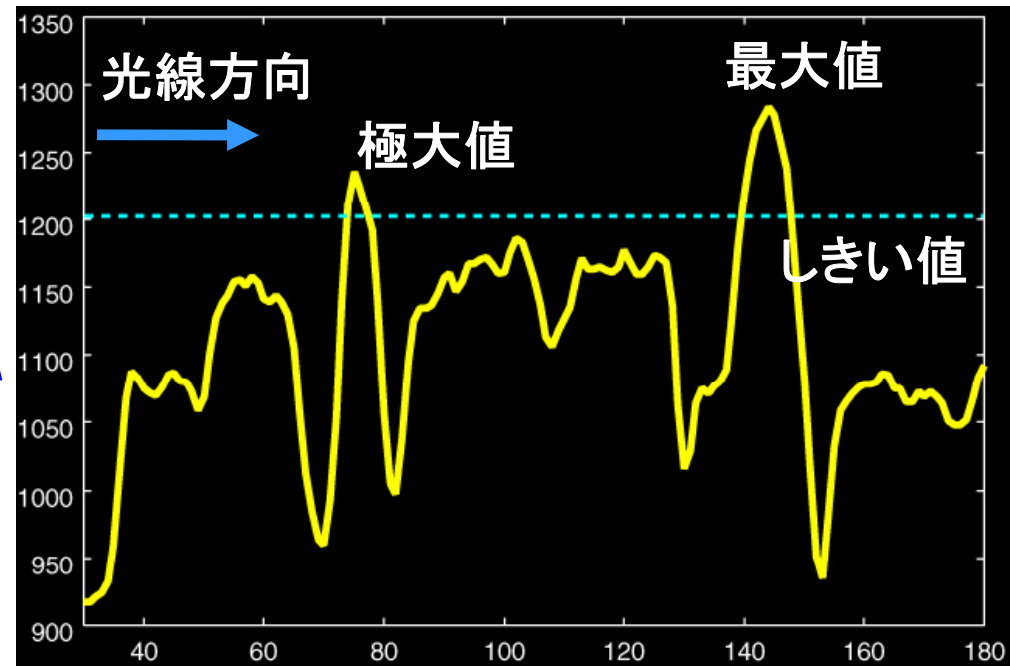
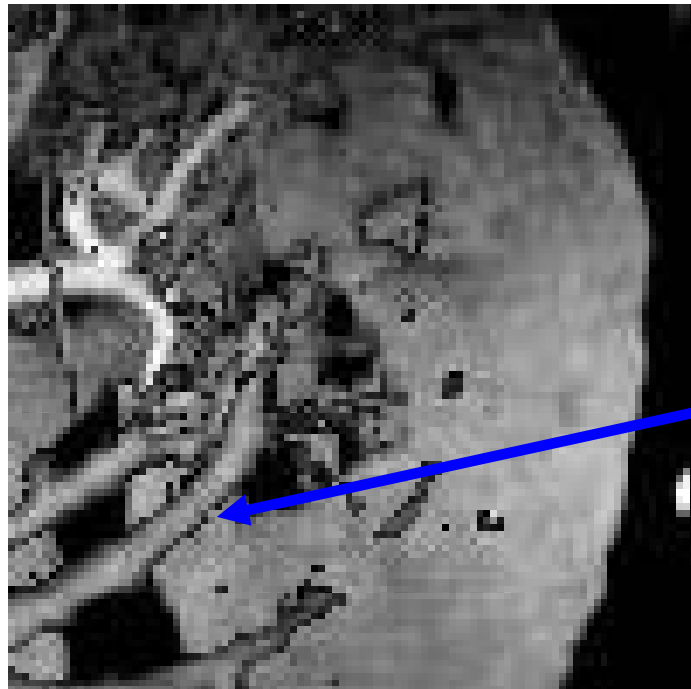
ボリューム
(3次元画像) ← しきい値



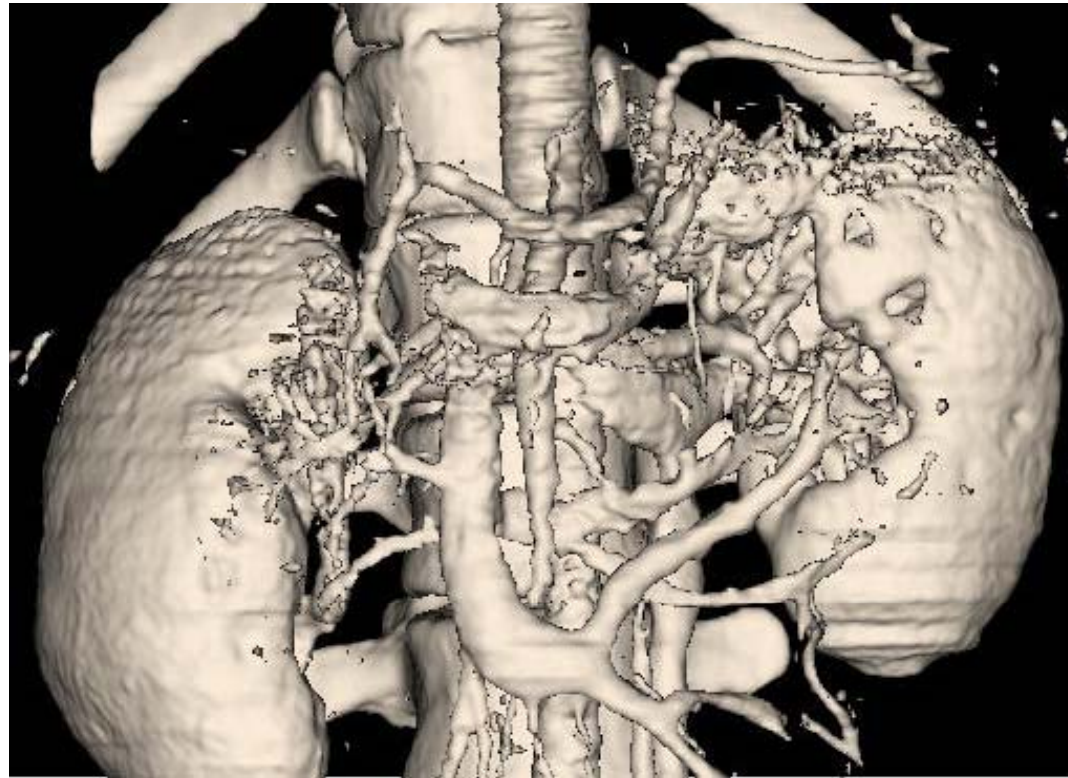
↓ 光線上の最初の極大値を選択(しきい値以上)

可視化画像

光線上のプロファイル

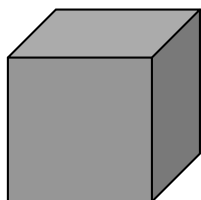


Surface Rendering



Surface Rendering

ボリューム

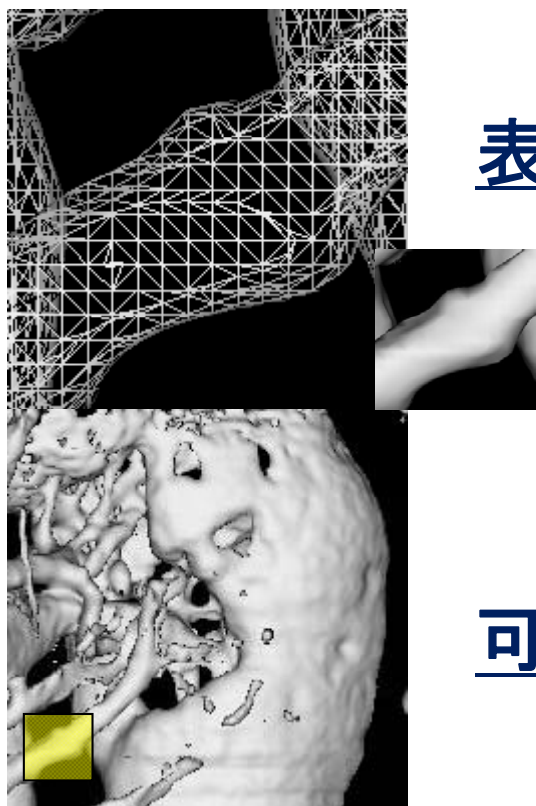


ボリューム

(3次元画像) ← しきい値



モデリング: マーチンキューブ法



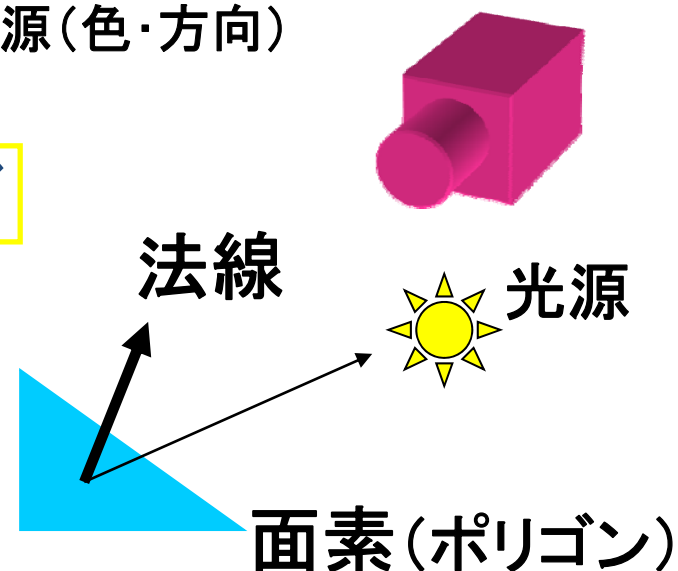
表面形状モデル

← 表面(面素)の色・材質
光源(色・方向)



レンダリング

可視化画像

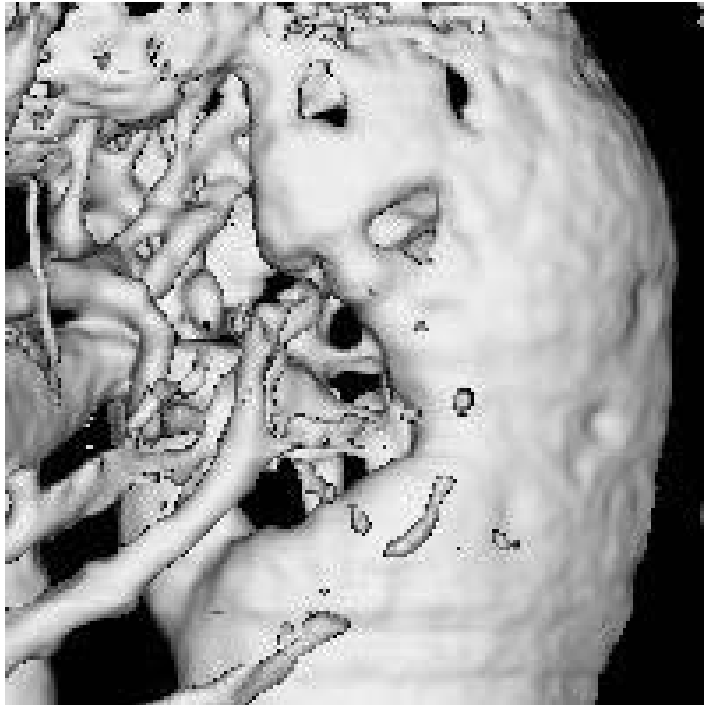


モデリングの原理：マーチンキューブ法

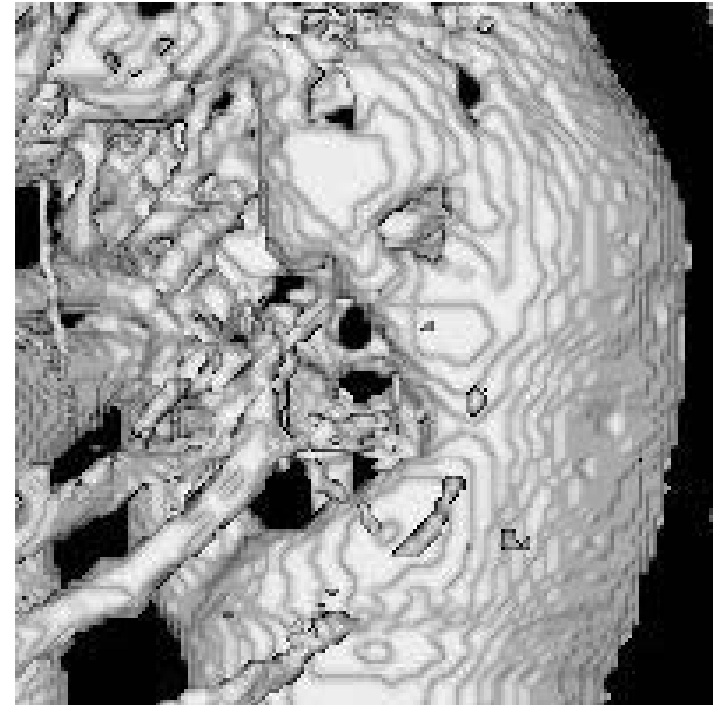
高解像度3次元形状モデリング手法

W. E. Lorensen and H. E. Cline, *Computer Graphics*, 21(3), 1987.

原画像に適用



二値画像に適用



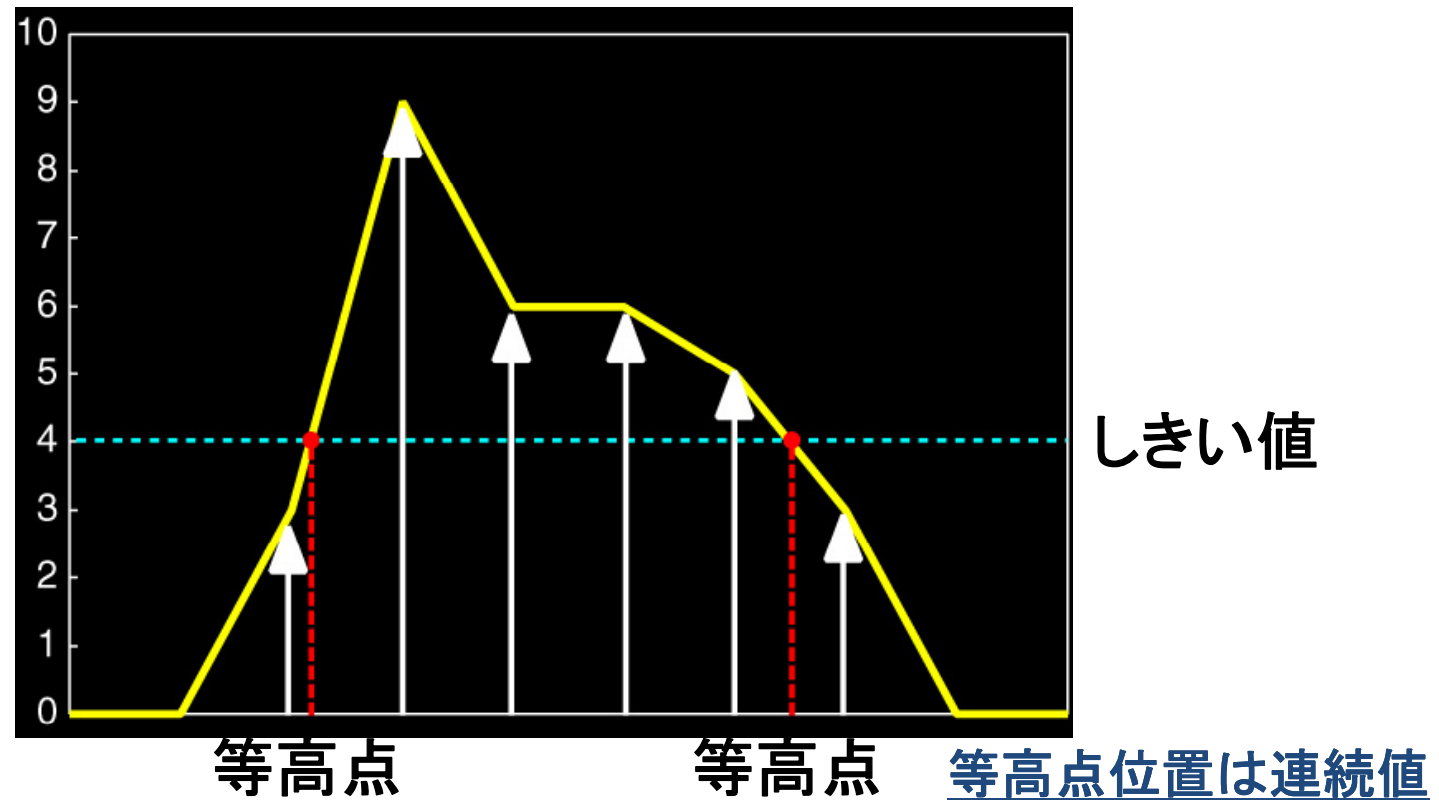
マーチンキューブ法

- 3次元画像(ボリュームデータ)の等濃淡値表面を抽出(再構成)する方法
- サブボクセル(1ボクセル以下)の解像度で再構成可能な方法
- 各ボクセル値の線形補間を行いながら、系統的にポリゴンデータ(表面形状モデル)に変換する。

モデリングの原理：マーチンキューブ法

1次元マーチンキューブ(インターバル)法

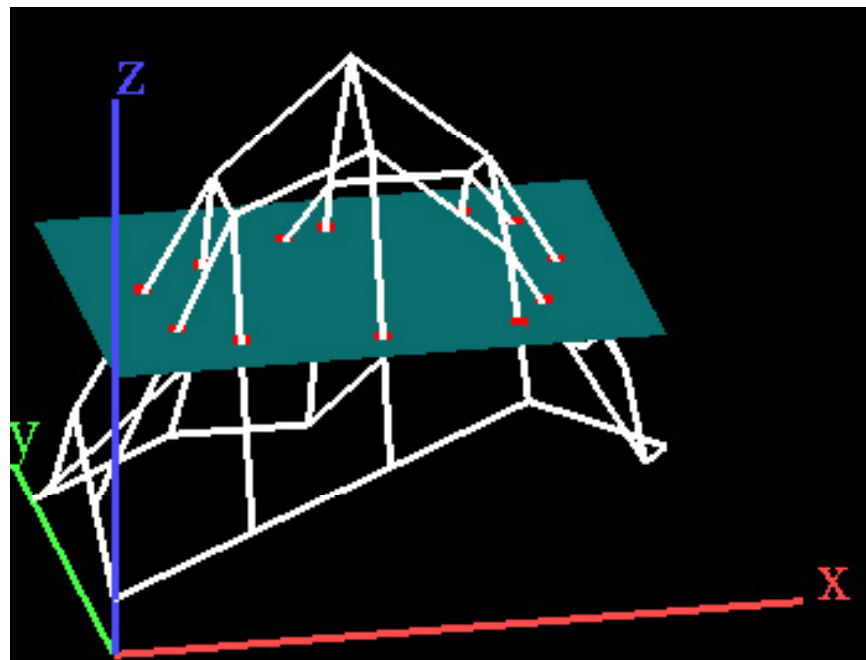
離散点での濃淡値を線形内挿して、2次元空間にプロットし、しきい値に対応する直線で切った切断点 (等高点)



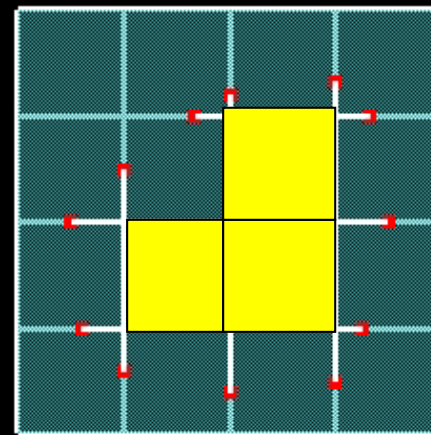
モデリングの原理：マーチンキューブ法

2次元マーチンキューブ(スクエア)法

3次元空間のしきい値平面で切った切断輪郭(等高線)

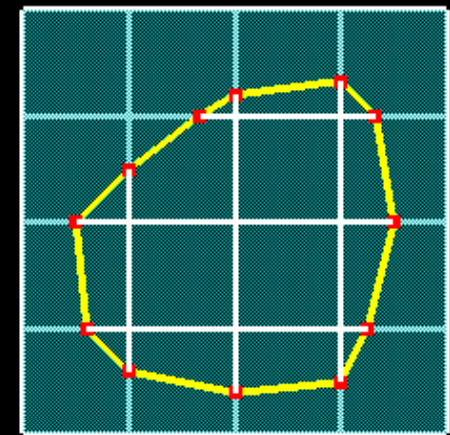


等高点



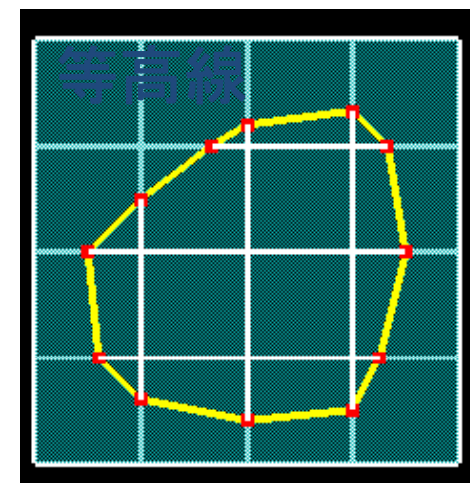
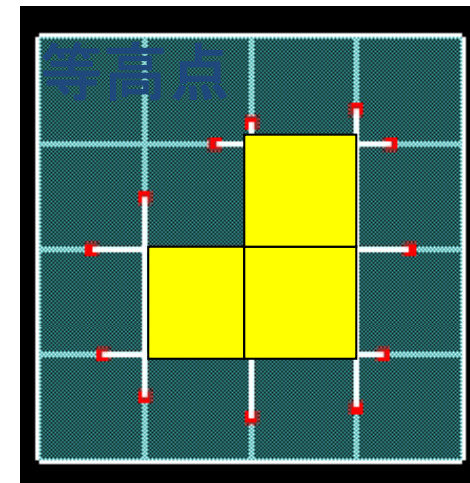
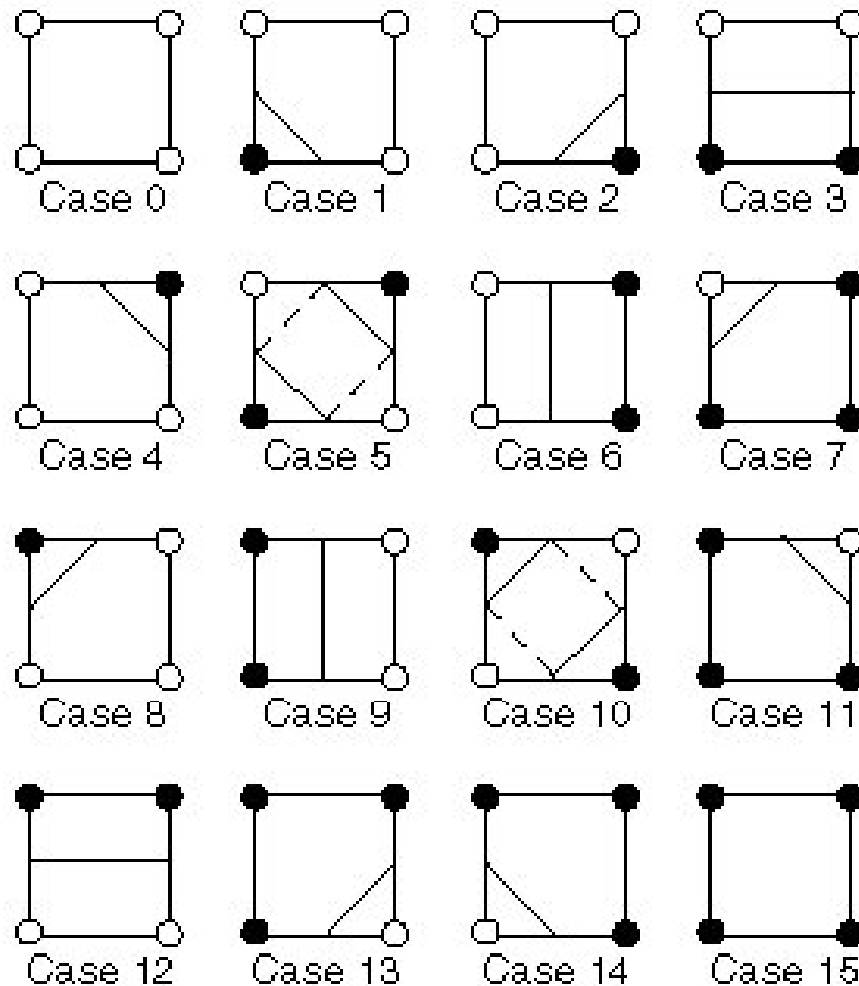
等高点位置は
格子線上

等高線



等高線位置は
サブ画素精度

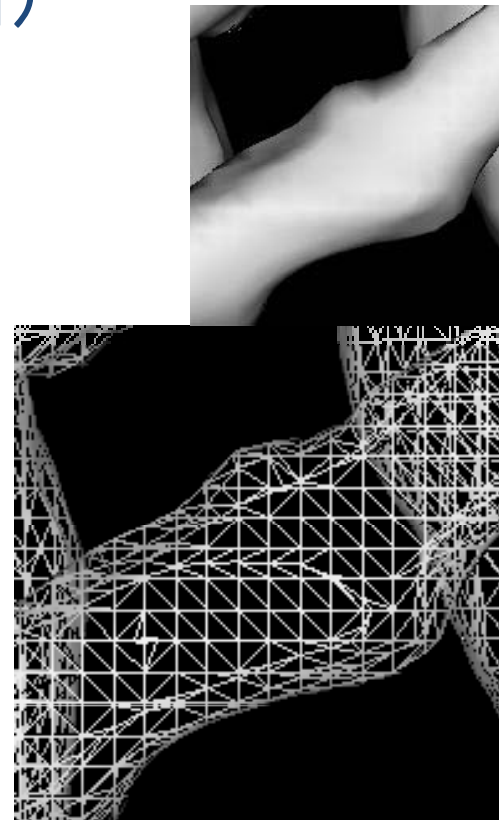
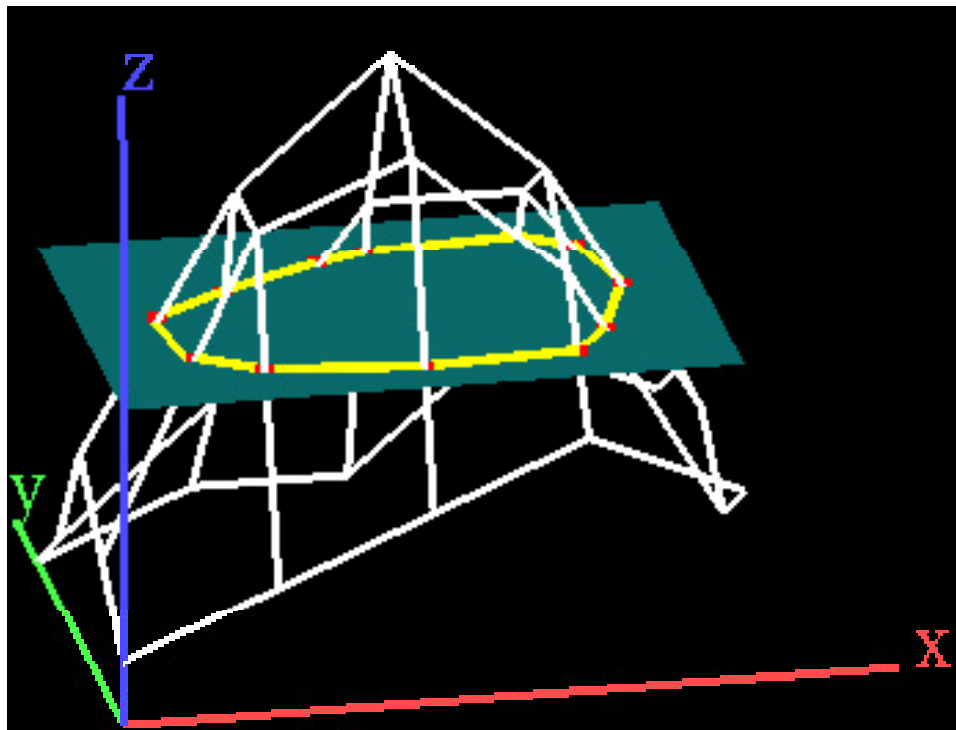
2次元マーチンキューブ(スクエア)法 等高点から等高線へ



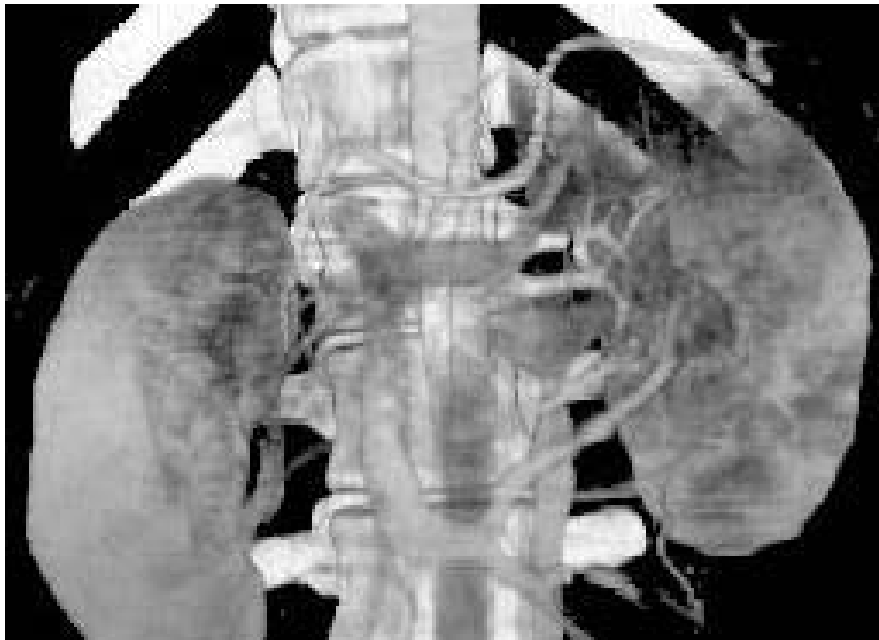
モデリングの原理：マーチンキューブ法

3次元マーチンキューブ法

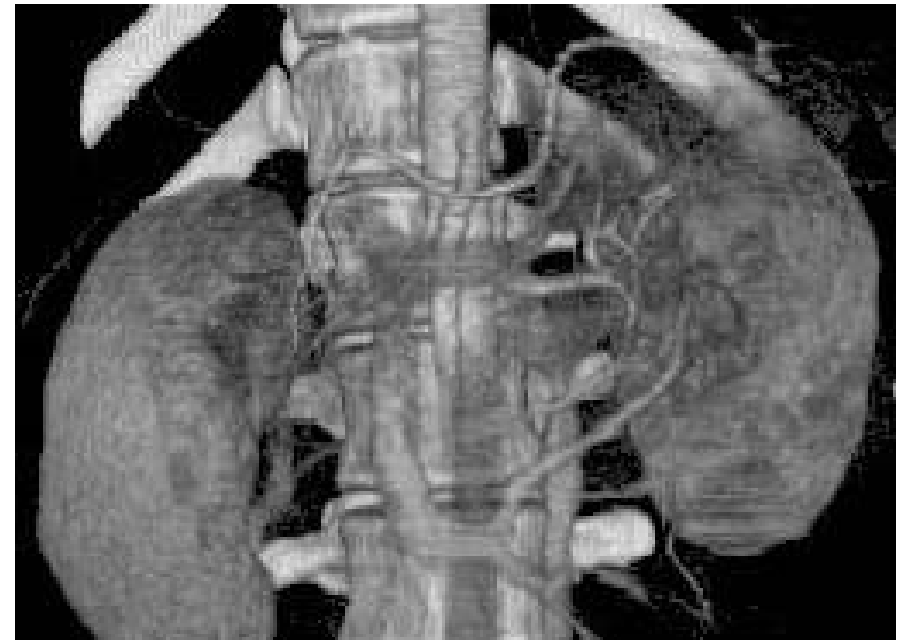
4次元空間のしきい値超平面で切った切断表面
(サブボクセル精度の等高面)



Volume Rendering

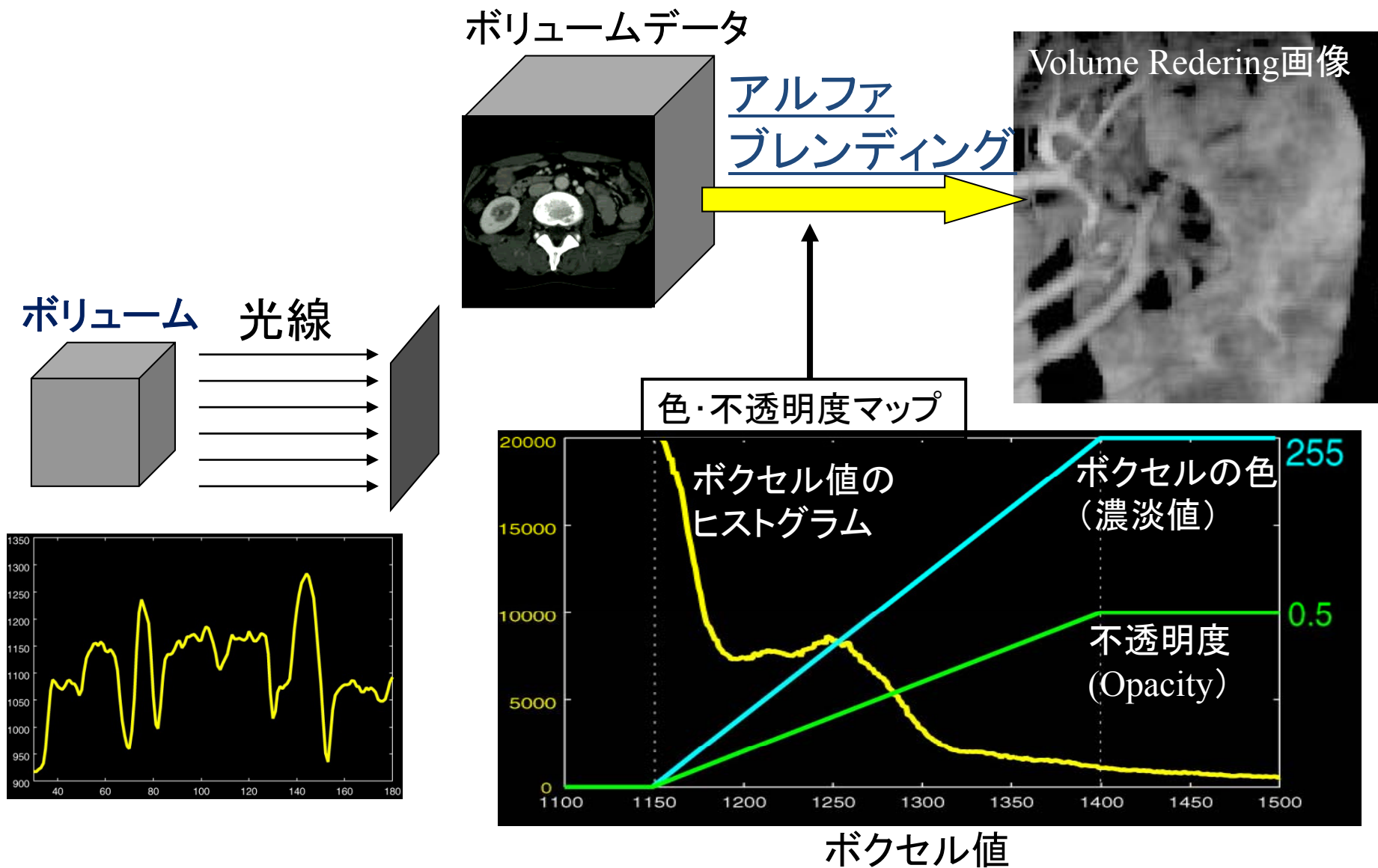


VR Unshaded



VR Shaded

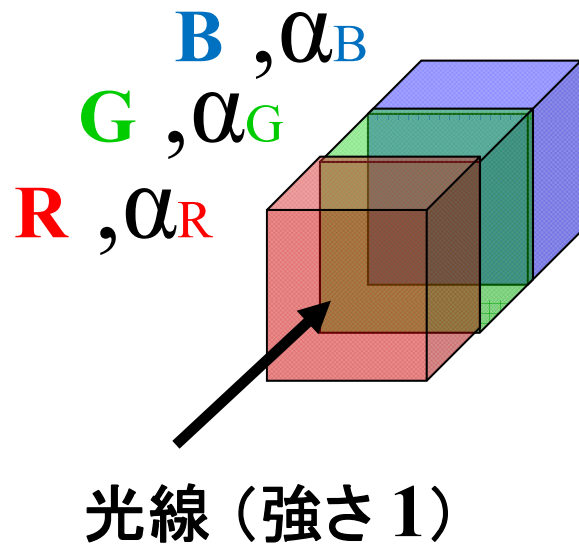
Volume Rendering: Unshaded (陰影なし)



不透明度合成の原理: アルファ・ブレンディング

ボクセル = 半透明の色つきゼリー

色, 不透明度 (0~1.0) 各ボクセルの画素値への寄与



透過した光線の強さ × 不透明度 × 色
 $(1 - \alpha_R) (1 - \alpha_G) \times \alpha_B \times B$

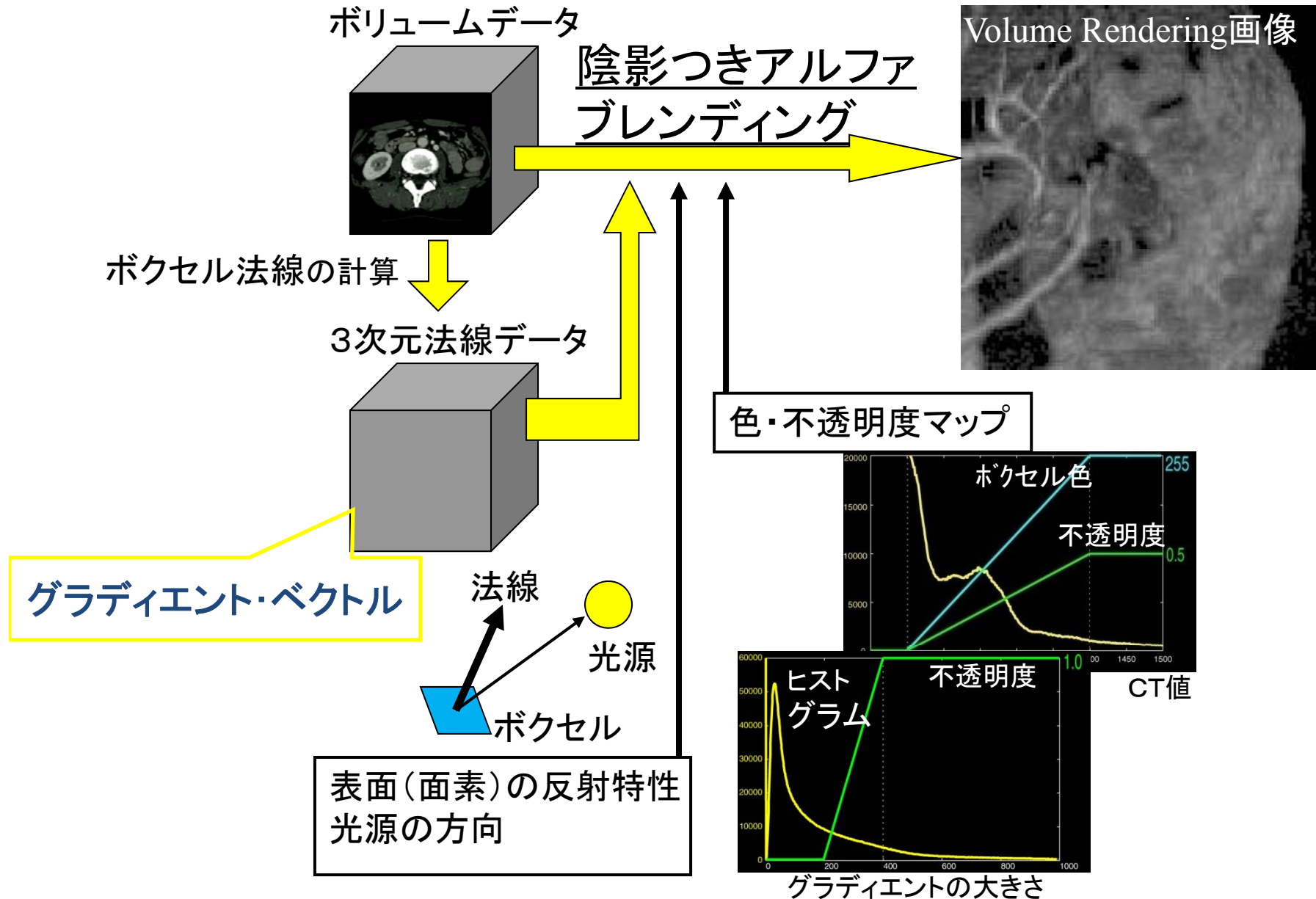
+
 $(1 - \alpha_R) \times \alpha_G \times G$

+
 $\alpha_R \times R$

↓
光線に対応する画素値

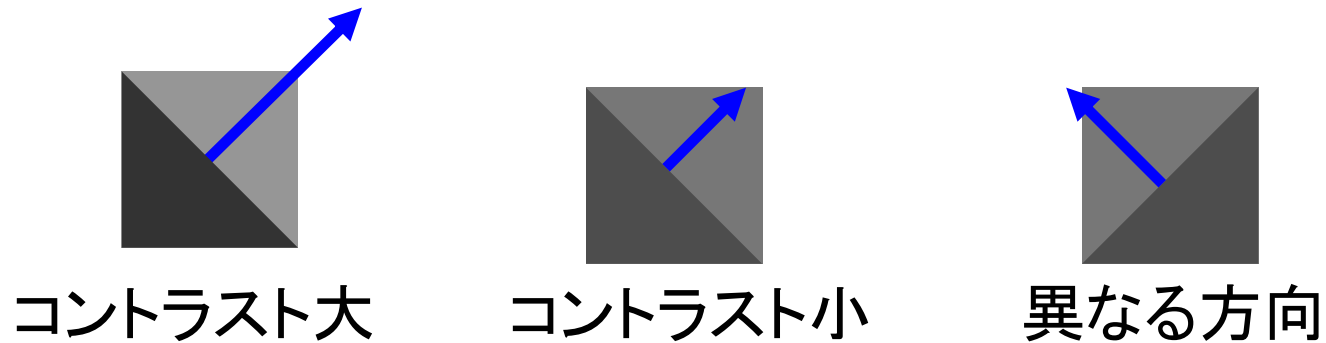
演習問題4-A: 光線に沿ったアルファブレンディングの一般式をかけ. 光線に沿ってN個のボクセルがあるとし, i ($1 \sim N$)で参照する. $i = 1$ が最も視点に近いとする. また, 不透明度を α_i , 色を C_i とする.

Volume Rendering: Shaded (陰影つき)



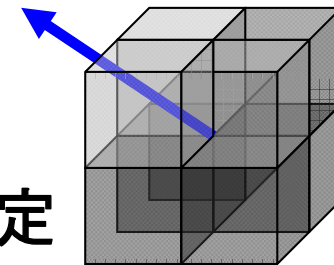
ボクセル法線計算の原理： グラディエント(勾配)ベクトル

グラディエント・ベクトルの方向：濃淡変化が最大となる方向
グラディエント・ベクトルの大きさ：濃淡変化の大きさ

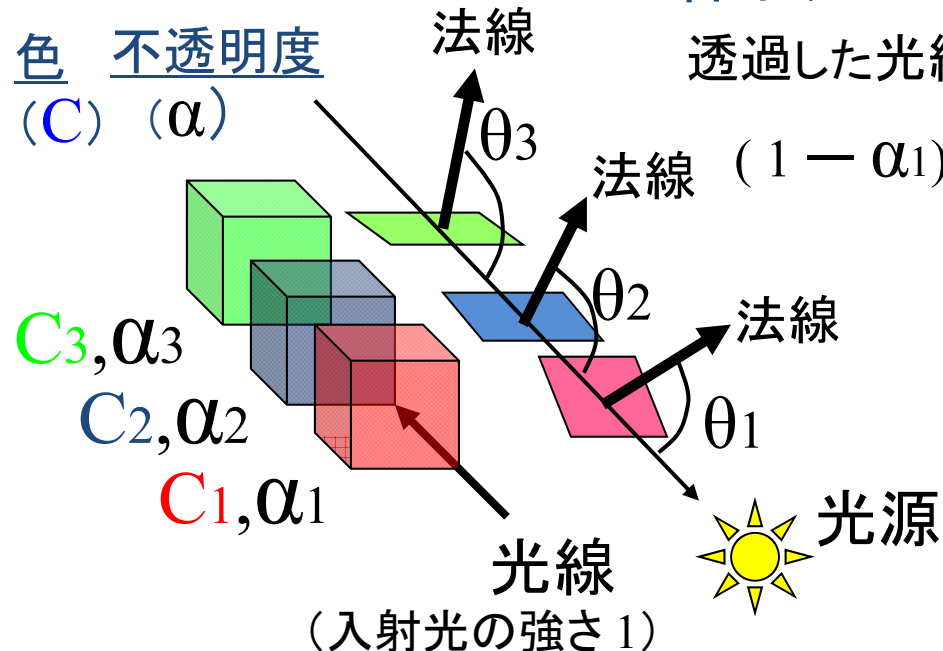


法線 = グラディエントベクトルの方向

近傍ボクセルからの推定



陰影つきアルファブレンディング



各ボクセルの画素値への寄与 =

透過した光線の強さ × 不透明度 × 陰影係数 × 色

$$\begin{aligned}
 & (1 - \alpha_1) (1 - \alpha_2) \times \alpha_3 \times \cos\theta_3 \times C_3 \\
 & + \\
 & (1 - \alpha_1) \times \alpha_2 \times \cos\theta_2 \times C_2 \\
 & + \\
 & 1 \times \alpha_1 \times \cos\theta_1 \times C_1
 \end{aligned}$$

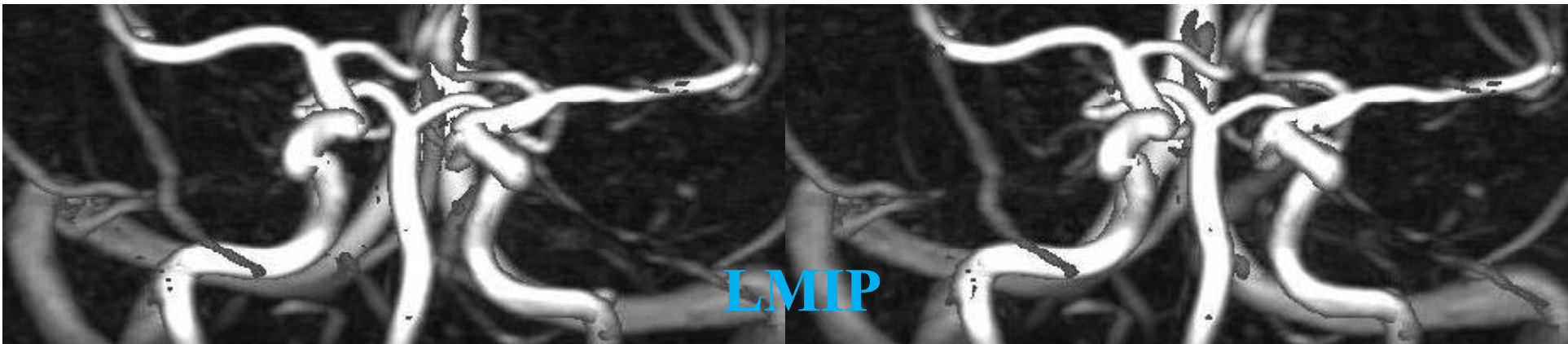
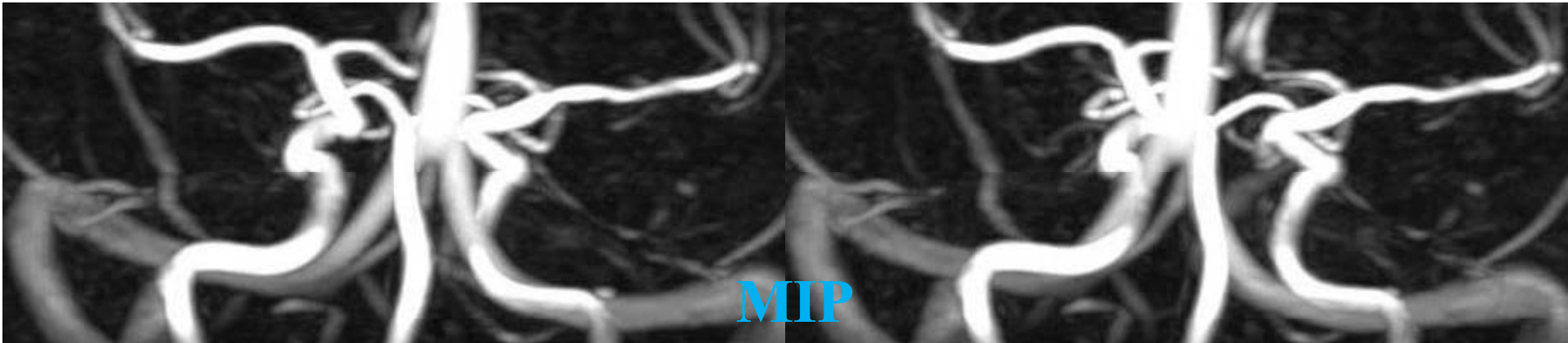
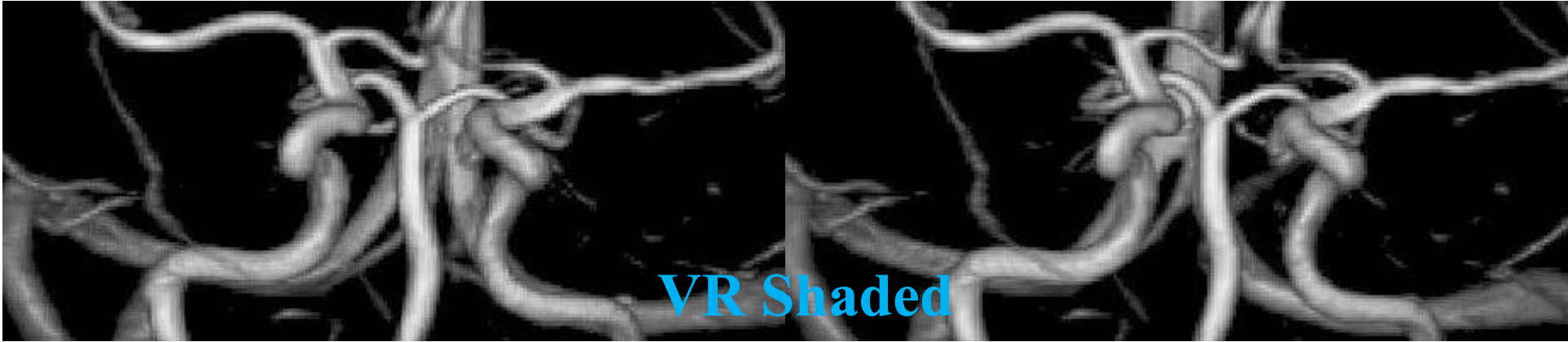
↓
光線に対応する画素値

面法線方向 \mathbf{n} (単位ベクトル) 光源方向 \mathbf{S} (単位ベクトル)

演習問題4-B: 光線に沿った陰影つきアルファブレンディングの一般式をかけ. 光線に沿ってN個のボクセルがあるとし, k ($1 \sim N$)で参照する. $k = 1$ が最も視点に近いとする. また, 不透明度を α_k , 色を C_k とする.

ボリュームグラフィックス法の比較 脳血管3次元像の例





ボリュームグラフィックス フリーソフト

Visualization Toolkit (VTK) 開発元: GE (General Electric)

動作環境: Unix, Windows <http://www.kitware.com>

機能: SR, VR (Shaded, Unshaded), 種々の3次元画像解析ツール

特徴: C++クラスライブラリ。Tcl/Tk スクリプト言語をサポート。

Volpack 開発元: スタンフォード大学

動作環境: Unix <http://graphics.stanford.edu/software/volpack/>

機能: VR (Shaded)

特徴: C関数ライブラリ。

不透明度関数設定におけるグラディエントの利用可能。

VG (Volume Graphics) プログラム演習

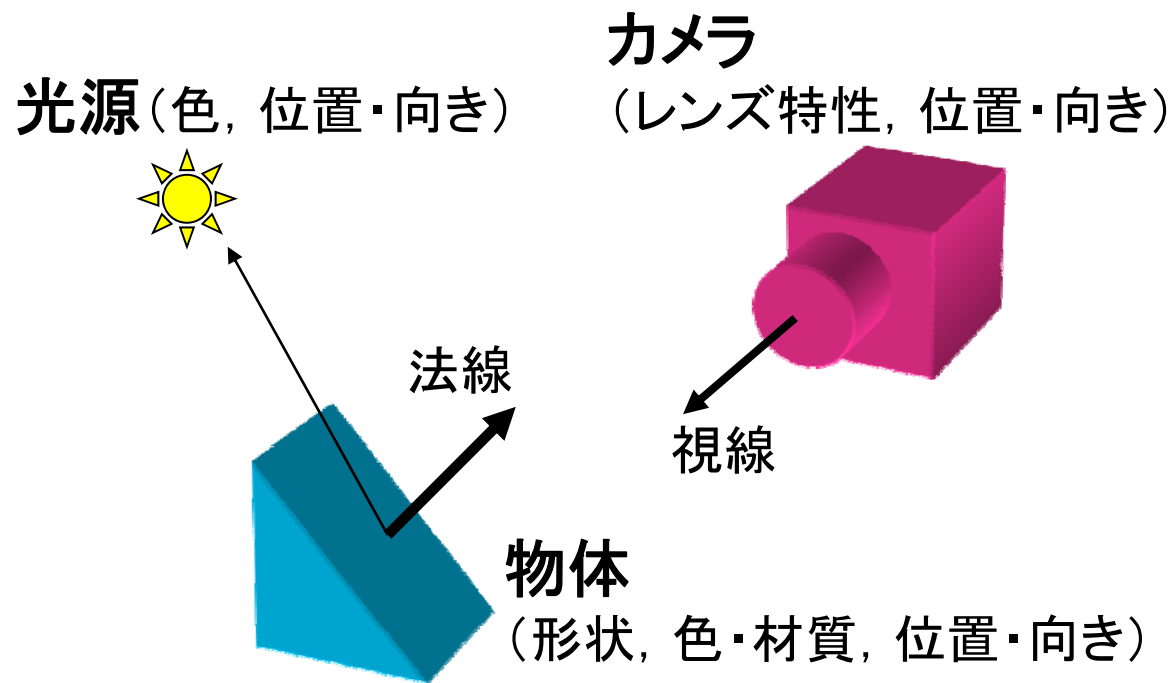
平行投影, 超楕円体, 回転,
アルファブレンディング,
グラディエントベクトル, 拡散反射

CG (Computer Graphics)

**物体形状表現
幾何学 Geometry**

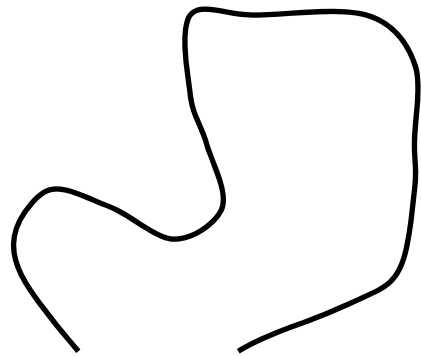
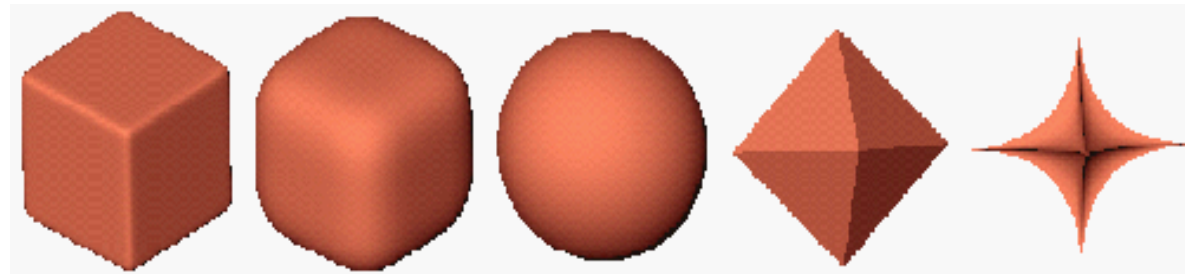
CG: 3次元世界表現と表示

- 座標系, 座標変換, 回転の表現, 投影(カメラ)モデル
- 物体形状表現 (幾何学 geometry) (スプライン関数, 超2次関数)
- 物体陰影表現 (photometry) (分光反射特性: 拡散反射, 鏡面反射)



CG: 物体形状表現

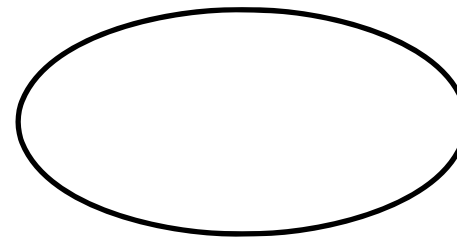
超2次関数 (Superquadrics): 超楕円体



自由形状

形状表現の多様さ

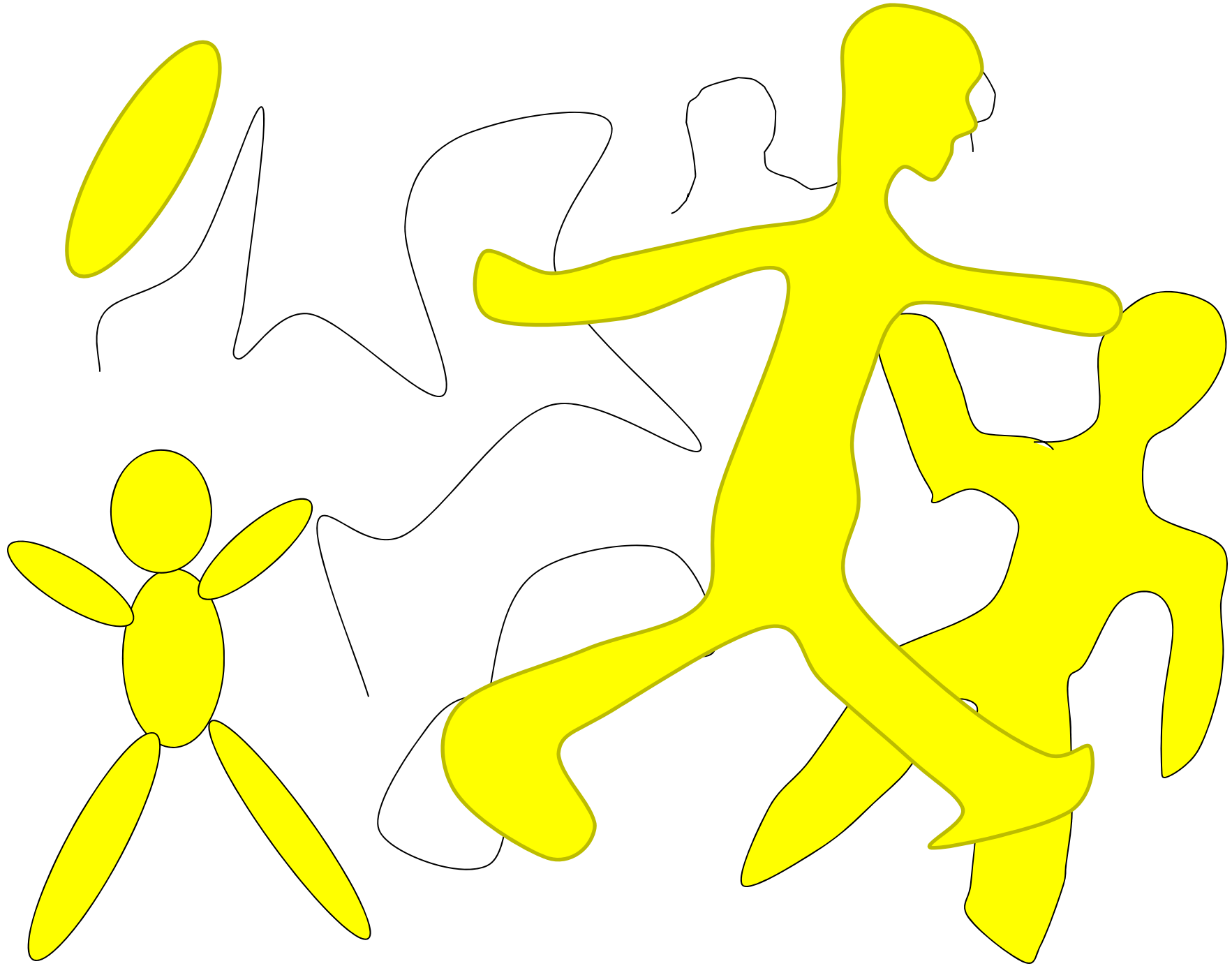
vs



解析関数

形状制御の容易さ

楕円 → 超楕円

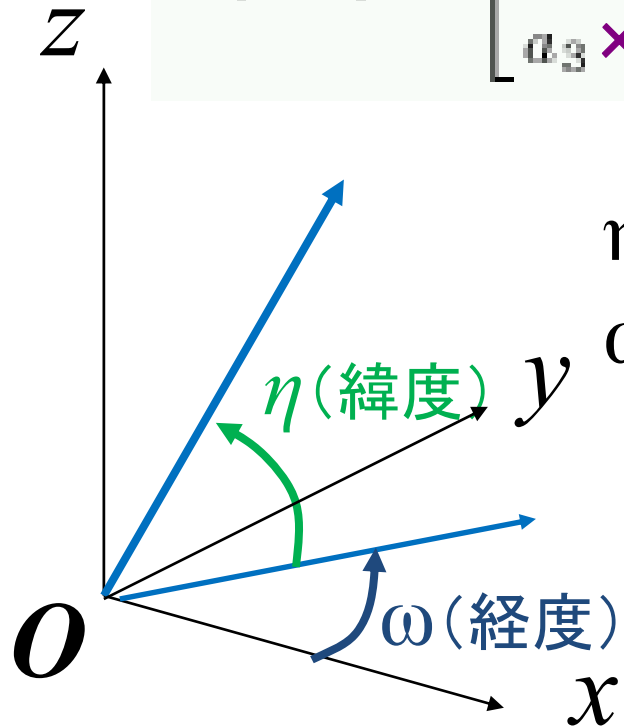


CG: 物体形状表現

2次関数 (Quadratics): 楕円体

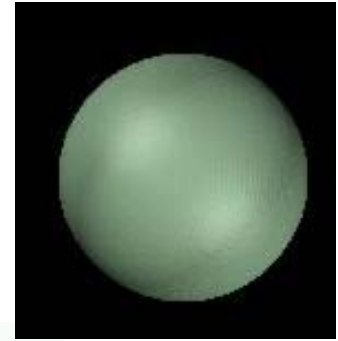
$$\left(\frac{x_1}{a_1}\right)^2 + \left(\frac{x_2}{a_2}\right)^2 + \left(\frac{x_3}{a_3}\right)^2 = 1$$

$$\underline{x}(\eta, \omega) = \begin{bmatrix} a_1 \times \cos \eta \times \cos \omega \\ a_2 \times \cos \eta \times \sin \omega \\ a_3 \times \sin \eta \end{bmatrix}, \quad \begin{array}{l} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega \leq \pi \end{array}$$



CG: 物体形状表現

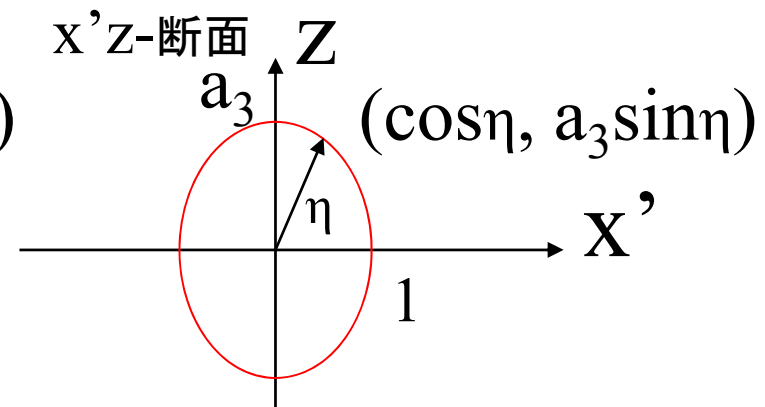
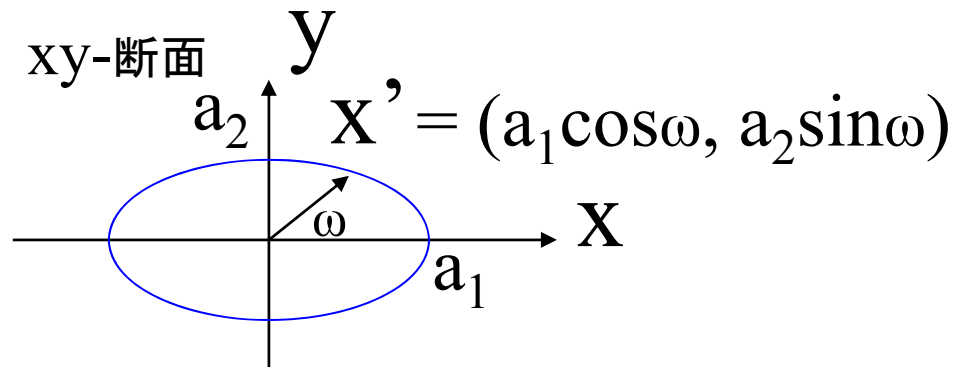
Spherical Products



$$\text{円} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} r \cos\theta \\ r \sin\theta \end{pmatrix} \quad \text{楕円} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a \cos\theta \\ b \sin\theta \end{pmatrix}$$

$$\left(\frac{x_1}{a_1}\right)^2 + \left(\frac{x_2}{a_2}\right)^2 + \left(\frac{x_3}{a_3}\right)^2 = 1$$

$$\underline{x}(\eta, \omega) = \begin{bmatrix} a_1 \cos\eta \cos\omega \\ a_2 \cos\eta \sin\omega \\ a_3 \sin\eta \end{bmatrix}, \quad \begin{matrix} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega < \pi \end{matrix}$$

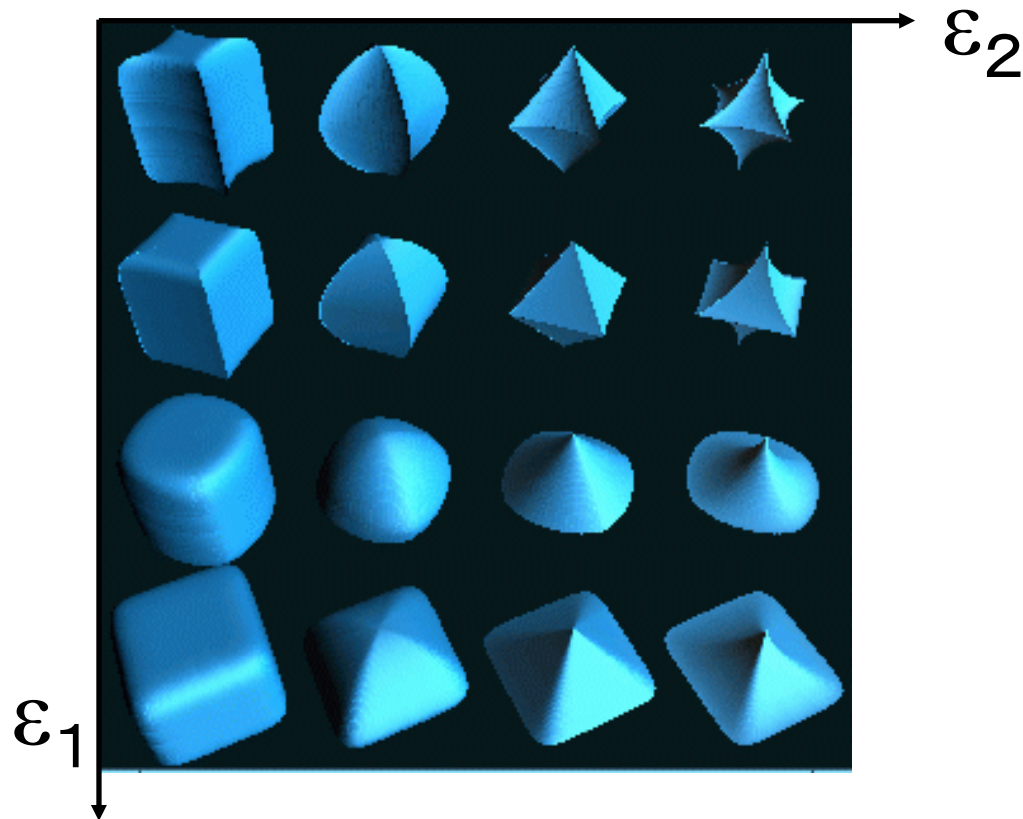


$$\cos\eta (a_1 \cos\omega, a_2 \sin\omega, 0) + a_3 \sin\eta (0, 0, 1)$$

CG: 物体形状表現

超2次関数 (Superquadrics): 超橢円体

$$\underline{x}(\eta, \omega) = \begin{bmatrix} a_1 \times \cos^{\epsilon_1} \eta \times \cos^{\epsilon_2} \omega \\ a_2 \times \cos^{\epsilon_1} \eta \times \sin^{\epsilon_2} \omega \\ a_3 \times \sin^{\epsilon_1} \eta \end{bmatrix}, \quad \begin{array}{l} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega \leq \pi \end{array}$$



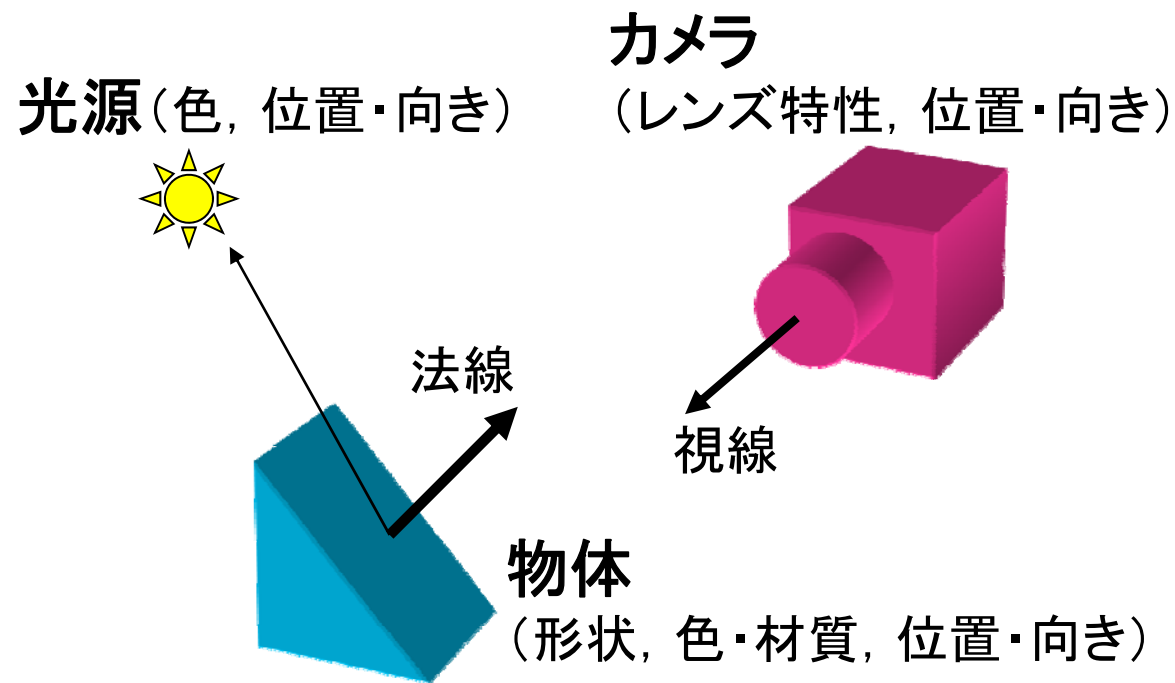
CG (Computer Graphics)

物体(表面)陰影表現

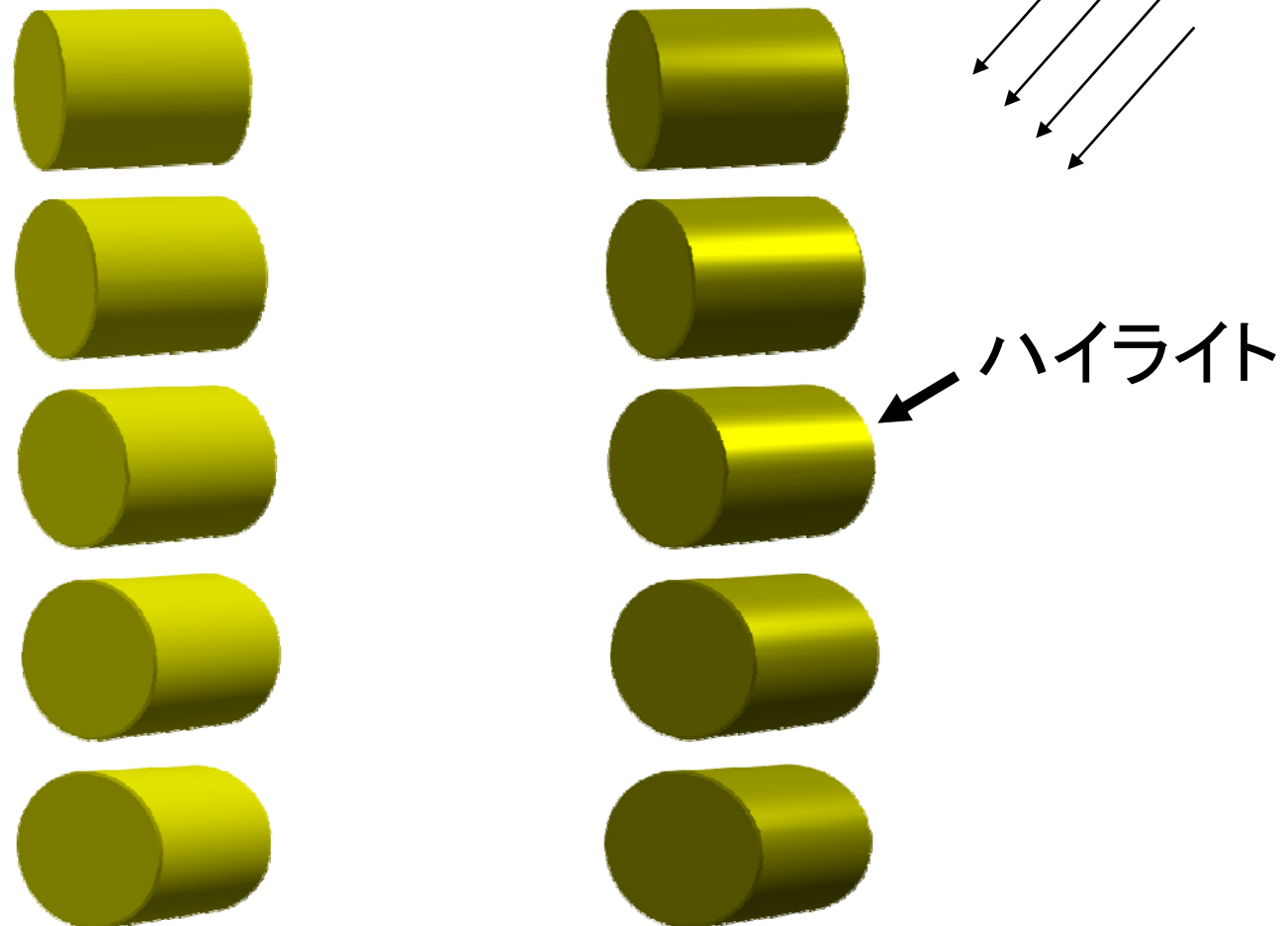
測光学 Photometry

CG: 3次元世界表現と表示

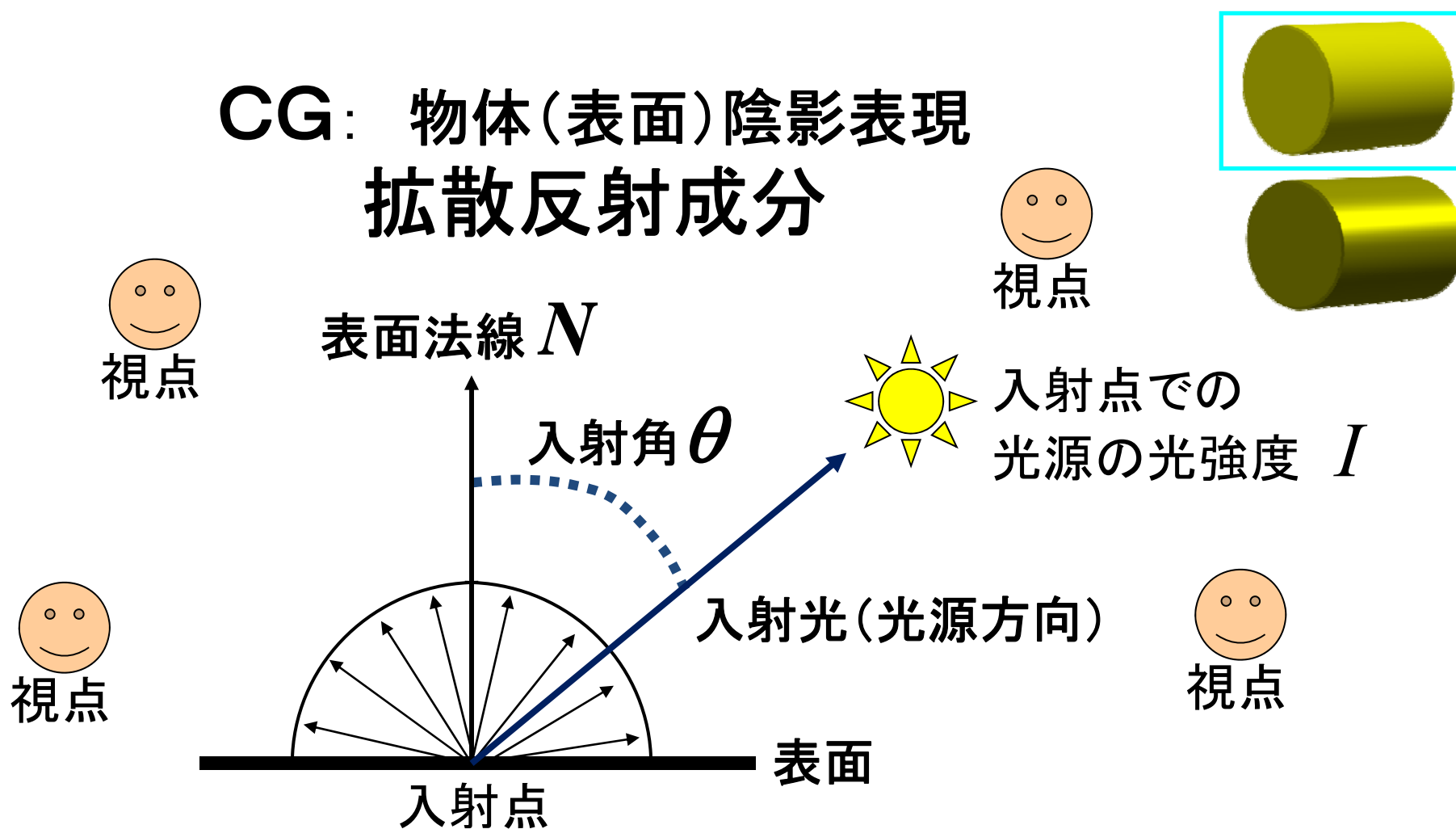
- 座標系, 座標変換, 回転の表現, 投影(カメラ)モデル
- 物体形状表現 (geometry) (スプライン関数, 超2次関数)
- 物体(表面)陰影表現 (測光学 photometry)
(分光反射特性: 拡散反射, 鏡面反射)



CG: 物体(表面)陰影表現 拡散反射 と 鏡面反射



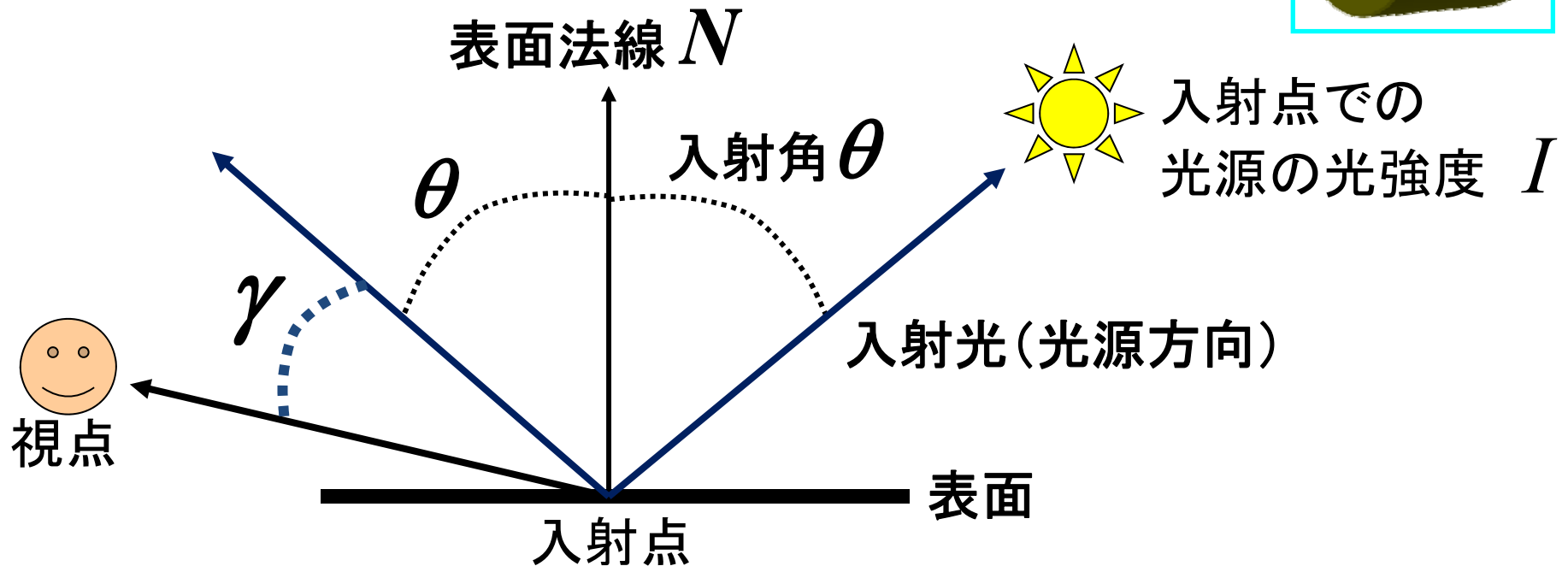
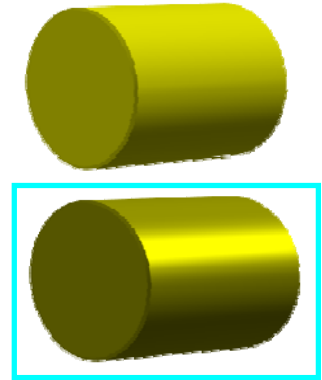
CG: 物体(表面)陰影表現 拡散反射成分



- 視点に依存しない(どこからみても同じ明るさに見える)
- みかけの明るさ I_d は、入射角のコサインに比例する。

$$I_d = k_d I \cos \theta \quad (k_d \text{ は、拡散反射係数})$$

CG: 物体(表面)陰影表現 鏡面反射成分



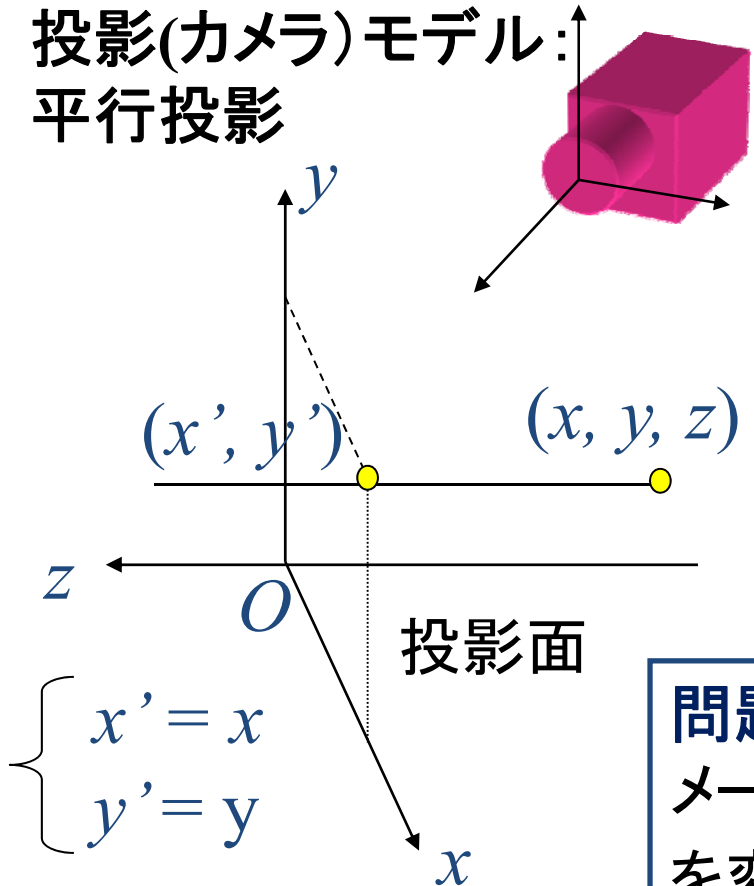
- 視点に依存する(みる方向によってみかけの明るさ I_s が変わる)
- ハイライトを生成し、その程度はハイライト指数 n に依存。

$$I_s = k_s I \cos^n \gamma \quad (k_s \text{ は、鏡面反射係数})$$

VGの原理：プログラム演習 (VG)

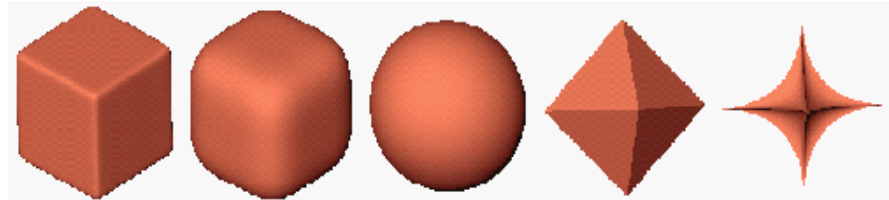
平行投影, 超楕円体, 拡散反射

投影(カメラ)モデル:
平行投影



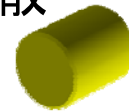
カメラ座標系を世界座標系と考える。

物体形状表現：超楕円体

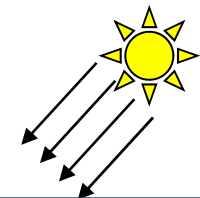


$$\underline{x}(\eta, \omega) = \begin{bmatrix} a_1 \times \cos^{\epsilon_1} \eta \times \cos^{\epsilon_2} \omega \\ a_2 \times \cos^{\epsilon_1} \eta \times \sin^{\epsilon_2} \omega \\ a_3 \times \sin^{\epsilon_1} \eta \end{bmatrix}, \quad \begin{matrix} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega < \pi \end{matrix}$$

物体陰影表現：拡散
反射成分のみ



光源：
平行光線



問題：以上の条件で，超楕円体の5つのパラメータ，回転パラメータ，および光源の方向を変化させて**陰影つきボリュームレンダリング**画像を生成できるプログラムを作り，方法・結果をレポートにまとめよ。

超楕円体ボリュームの生成: 1

楕円体

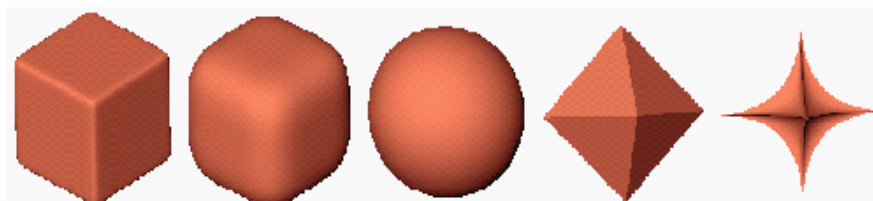
陰関数表現

$$\left(\frac{x_1}{a_1}\right)^2 + \left(\frac{x_2}{a_2}\right)^2 + \left(\frac{x_3}{a_3}\right)^2 = 1$$

パラメトリック表現

$$\underline{x}(\eta, \omega) = \begin{bmatrix} a_1 \times \cos \eta \times \cos \omega \\ a_2 \times \cos \eta \times \sin \omega \\ a_3 \times \sin \eta \end{bmatrix}, \quad \begin{matrix} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega < \pi \end{matrix}$$

超楕円体



パラメトリック表現

$$\underline{x}(\eta, \omega) = \begin{bmatrix} a_1 \times \cos^{\epsilon_1} \eta \times \cos^{\epsilon_2} \omega \\ a_2 \times \cos^{\epsilon_1} \eta \times \sin^{\epsilon_2} \omega \\ a_3 \times \sin^{\epsilon_1} \eta \end{bmatrix}, \quad \begin{matrix} -\pi/2 \leq \eta \leq \pi/2 \\ -\pi \leq \omega < \pi \end{matrix}$$

問題P2-1 超楕円体の陰関数表現を導け.

超楕円体ボリュームの生成: 2

超楕円体の陰関数表現を以下とする

$$F(x, y, z) = 1$$

128 × 128 × 128 (128³) の3次元配列を用意する。メモリに余裕があれば、256³でもよいし、メモリが足りなければ64³でもよい。

3次元配列の各要素に、 $F(x, y, z) > 1$ (Outside) のときは、0, それ以外 (Inside) は100を入れる。

回転させて超楕円体のボリューム生成を行うためには、回転行列をかけてから上の処理を行う。

ボリュームの平滑化

入力ボリューム (配列): $I(i, j, k)$

平滑化ボリューム (配列): $J(i, j, k)$

$$J(i, j, k) = (1/27) \sum_{p=-1}^1 \sum_{q=-1}^1 \sum_{r=-1}^1 I(i+p, j+q, k+r)$$

(一様平均による平滑化)

一様平均による平滑化を何回か繰り返す

$$J_0(i, j, k) = I(i, j, k)$$

$$J_n(i, j, k) = (1/27) \sum_{p=-1}^1 \sum_{q=-1}^1 \sum_{r=-1}^1 J_{n-1}(i+p, j+q, k+r)$$

問題P2-2: 一様平均による平滑化を n 回繰り返すと、どんな平滑化になっているか? (n が大きくなるとある有名な関数の重み平均に近づく.)

グラディエント(勾配)ベクトルの計算

平滑化ボリューム(配列): $J(i, j, k)$

グラディエントボリューム(配列)

$$x\text{成分: } G_x(i, j, k) = J(i+1, j, k) - J(i-1, j, k)$$

$$y\text{成分: } G_y(i, j, k) = J(i, j+1, k) - J(i, j-1, k)$$

$$z\text{成分: } G_z(i, j, k) = J(i, j, k+1) - J(i, j, k-1)$$

$$\text{法線方向: } \mathbf{n} = (G_x, G_y, G_z) / \text{Sqrt}(G_x^2 + G_y^2 + G_z^2)$$

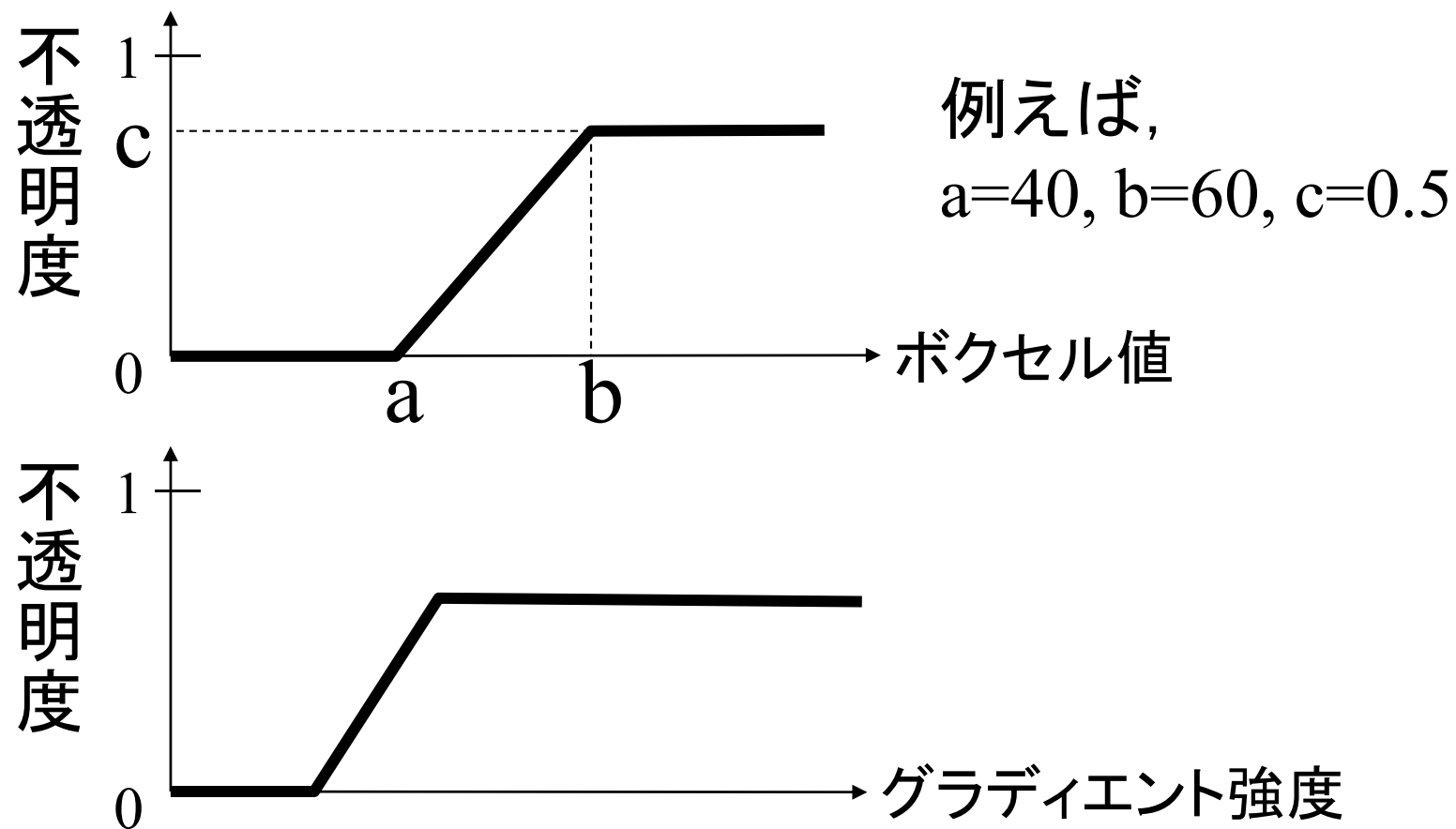
$$\text{グラディエント強度: } \text{Sqrt}(G_x^2 + G_y^2 + G_z^2)$$

(i, j, k) は, 連続変数 (x, y, z) の離散変数に対応.

不透明度関数の設定

色は, 例えば「白」に固定する.

不透明度は, 例えば以下のようにする.



実験項目および考察事項

- 超楕円体のパラメータ, 回転パラメータを変化させてボリュームレンダリングせよ.
- 光源の方向を変化させよ.
- 一様平均による平滑化の回数を変化させよ.
- 不透明度関数を変化させよ.
- その他, 各自, 創意工夫を加えて実験を行うこと.
- 以上の実験結果に考察を加えよ.
- プログラムが得意な学生は、使いやすいGUIも作成せよ。

発展課題：運動生成

- 同じ形状パラメータで、異なる2つの回転(姿勢)をもつ超楕円体の画像をそれぞれ生成せよ。
- 一方の姿勢から他方の姿勢に、超楕円体の姿勢がなめらかに変化するよう、アニメーション画像系列(2つの姿勢の間、中間姿勢を持つ複数毎の画像)を生成せよ。
 - 単位四元数により回転を表現し、四元数の補間式を用いて、2つの回転の間を、回転角度差5度程度になるよう等角度間隔サンプリングを行う。角度差は、単位四元数の内積のArcCosにより求められる。
 - 必要なら、単位四元数表現をマトリックス表現に変換して、画像生成を行う。
- 四元数に基づく補間とオイラー角に基づく補間の両方の画像系列を生成し、どのような違いがあるか調べよ。

質問1

マルチメディア工学のプログラム演習について、質問です。

内容は、超楕円体のパラメータ表現を授業の配布資料どおりにサーフェスレンダリングのプログラムで実装すると、座標の値が複素数になってしまう点があることについてです。

例えば、 $-\pi/2 \leq \eta \leq \pi/2$ なので、 η が $-\pi/2$ 、 ε_1 が0.5の場合に z 座標の値が

$$z = a^3 \times (\sin\eta)^{\varepsilon_1} = a^3 \times (\sin(-\pi/2))^{0.5} = a^3 \times (-1)^{0.5}$$

となってしまいます。この場合の対処について、授業の資料では触れられていないのでよく分かりません。

現在は $\sin\eta$ や $\cos\omega$ の符号が、超楕円体上の点が、原点から見てどの方向にあるのか、 $\sin\eta$ や $\cos\omega$ の大きさが原点からの距離にかかわる値で、 $\varepsilon_1, \varepsilon_2$ はその距離を補正する要素だと考え、以下のようなプログラムを作成しています。

$$z \text{ 座標の値} = a^3 \times |\sin\eta|^{\varepsilon_1} \times (\sin\eta/|\sin\eta|)$$

以上の方法で問題ないのか、また問題がある場合はどのように対処すべきか、が質問内容です。よろしくお願いします。

回答2

- z 座標の値 = $a^3 \times |\sin\eta|^{\epsilon_1} \times (\sin\eta/|\sin\eta|)$
- この方法で問題がないか

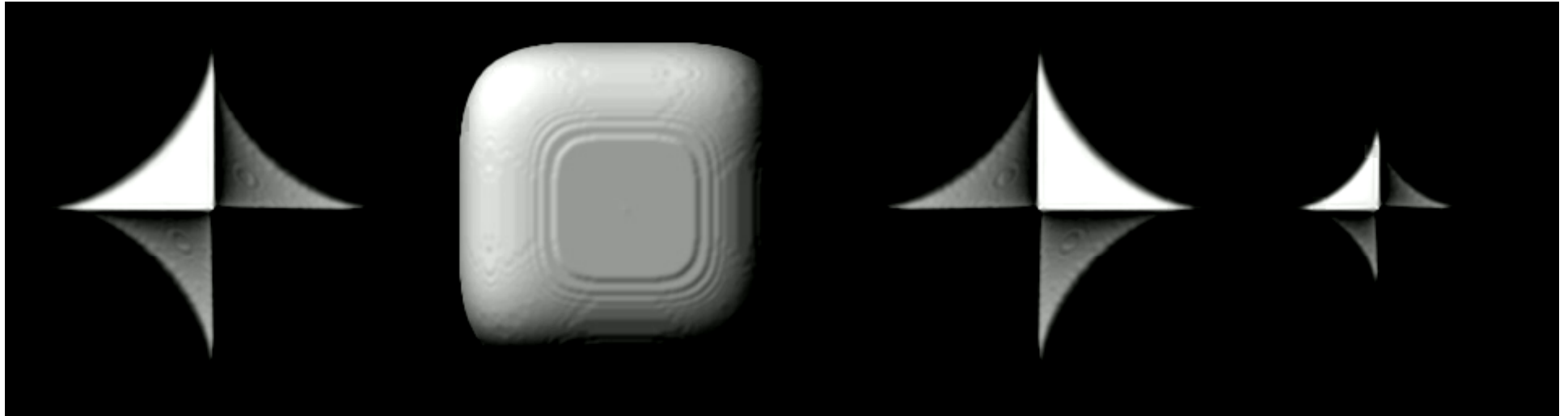
考え方は合っています。ほとんど正解ですが、このままでは $\sin\eta=0$ のときにゼロ除算になってしまいます。実際には計算精度の関係でぴったりゼロになることはないので問題は起きないかもしれませんが....

- どのように対処すべきか

($\sin\eta/|\sin\eta|$)の部分を、 $\sin\eta=0$ の時は0になるように修正すればいいです。そのための関数がサンプルコードには用意してあります (SurfaceRendering.cにあるsign()関数)ので、これを使ってもらってもOKです

.

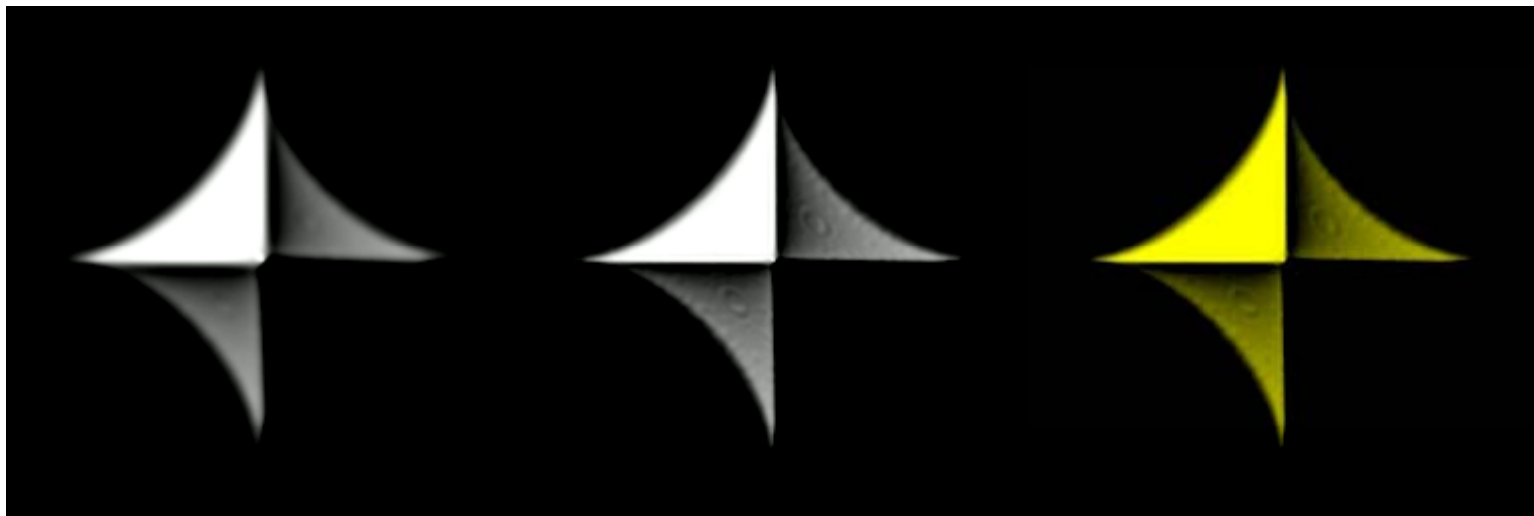
Volume Rendering 出力例



形状パラメータ

光源方向

形状サイズ



平滑化

回転

色