Podstawy Grafiki Komputerowej

Wykłady 9 i 10: Reprezentacje

Andrzej Łukaszewski

Pracownia Grafiki Komputerowej
(LIGHT — Laboratory of Imaging and GrapHic Techniques)

Instytut Informatyki, Uniwersytet Wrocławski

9/16 grudnia 2019

Reprezentacje: hierarchia

Co reprezentujemy:

- pojedyncze obiekty
- całe sceny
- animacje złożone z wielu klatek

Na najniższym poziomie dla obiektu pamiętamy:

- Geometrie np. współrzędne wierzchołków
- Atrybuty (czy dla wierzchołków?):
  - wektory normalne
  - własności materiału (np. współczynniki modelu Phonga)
  - współrzędne tekstury

Szczególne obiekty: Światła, Kamery

Różne wymiary reprezentacji:

- 0D chmury punktów
- 2D reprezentacje brzegowe
- 3D wolumetryczne

Reprezentacje: Scene Graph

Hierarchiczna reprezentacja sceny: SceneGraph

Przykłady standardów: Inventor, różne wersje VRML

Tiling danych: porządek scanline gorszy niż kwadraty dla szybkiego działania cache lub drzewa czwórkowe.

Reprezentacje: skąd są dane

Skąd biorą się modele 3D i jakie są oryginalne reprezentacje:

1. z sieci ?
2. skanowanie 3D
3. modelowanie: blender, wings3D, 3Dstudio, Maya,...
4. proceduralne metody generowania, wizualizacje funkcji, funkcje uwikłane,
5. dane medyczne np. tomografia
6. dane z symulacji

Reprezentacje: siatki wielokątów

Slajdy S. Marshnera do podręcznika “Fundamentals of Computer Graphics”:

http://www.cs.cornell.edu/Courses/cs4620/2008fa/lectures/11trimesh.pdf

(siatki trójkątów, topologia, Winged-Edge,. . .)

Reprezentacje: powierzchnie uwikłane

Zbiór punktów spełniających równanie:

$$F(x, y, z) = 0$$

gdzie F jest najczęściej wielomianem ustalonego stopnia n.

Dla n=1 mamy płaszczyznę, dla n=2 powierzchnie kilku sklasyfikowanych typów: ...

elipsoidy, cylindry, stożki, hiperboloidy, paraboloidy

Dla większych n brak klasyfikacji i wygląd mniej przewidywalny.

Reprezentacje: powierzchnie parametryczne

Kanciastość siatek wielokątów i czajnik z Utah:

- wersja parametryczna 32 powierzchnie po 16 punktów 3D
- wersja wielościenna 2048 czworokątów (około 8 razy więcej) dla podobnej jakości

$$S(u,v) = (X(u,v), Y(u,v), Z(u,v))$$

Wielomiany X,Y,Z stopnia n. Reprezentacja w postaci współczynników wielomianów mało intuicyjna. Dlatego używane różne wersje są oparte na punktach kontrolnych które powierzchnia:

- interpoluje
- aproksymuje

Np. pow. Beziere'a, różne pow. sklejane, pow. Hermite'a

Reprezentacje: krzywe Beziere'a

Historia: 1969 Renault Pierre Beziere: UNISURF system do modelowania karoserii samochodów

Definition

Def: Wielomiany Bernsteina stopnia n dla i=0... n

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

Definition

Def: Krzywa Beziere'a stopnia n oparta na punktach kontrolnych $P_o, P_1, \ldots P_n$ to:

$$C(t) = \sum_{i=0}^{n} B_{i,n}(t) P_i$$

Reprezentacje: własności krzywych Beziere'a

Dla przedziału parametrów krzywej $t \in [0, 1]$ mamy:

$$\sum_{i=0}^{n} B_{i,n}(t) = 1$$

$$B_{i,n}(t) \geq 0$$

$$\frac{d}{dt} B_{i,n}(t) = n(B_{i-1,n-1}(t) - B_{i,n-1}(t))$$

$\Rightarrow$ **własność otoczki wypukłej** (kombinacja wypukła...)
$\Rightarrow$ pochodna: $C'(0) = n(P_1 - P_0)$ i $C'(1) = n(P_n - P_{n-1})$
Uwagi:

1. krzywa przechodzi zazwyczaj tylko przez punkty końcowe
2. krzywa oparta na $n + 1$ punktach ma stopień n
3. zmiana 1 punktu ma globalny wpływ na kształt krzywej

Reprezentacje: własności krzywych Beziere'a

Gładkie łączenie, ciągłość geometryczna: Ciągłość wektora pochodnej C^1 i ciągłość prostej stycznej do krzywej G^1 (nie ma implikacji w żadną stronę).

Algorytm de Casteljaux: Metoda podziału i obliczania dowolnego punktu $C(\lambda)$

Zastosowania:

- rysowanie krzywej,
- metoda podziału,
- znajdowanie przecięcia z prostą (metoda podziału: test z otoczką wypukła i podział rekurencyjnie).

Reprezentacje: powierzchnie Beziere'a

Iloczyn tensorowy:

$$S(u,v) = \sum_{i=0}^{n} \sum_{j=0}^{m} B_{i,n}(u) B_{j,m}(v) P_{ij}$$

powierzchnia generowana przez krzywe z krzywych w jednym kierunku co widać z :

$$S(u,v) = \sum_{i=0}^{n} B_{i,n}(u) \sum_{j=0}^{m} B_{j,m}(v) P_{ij}$$

Algorytm de Casteljaux działa podobnie, koszt podziału: ...

Reprezentacje: trójkątne powierzchnie

Trójkątne powierzchnie Beziere'a definiujemy przy pomocy odpowiednich bazowych funkcji Bernsteina:

$$B^n_{ijk}(u,v,w) = \frac{n!}{i!j!k!}u^i v^j w^k$$

oczywiście $i + j + k = n$

Definition

Trójkątną powierzchnie/płat Beziere'a definiujemy jako

$$S(u,v,w) = \sum_{i,j,k=0;i+j+k=n}^{n} B^n_{ijk}(u,v,w)P_{ijk}$$

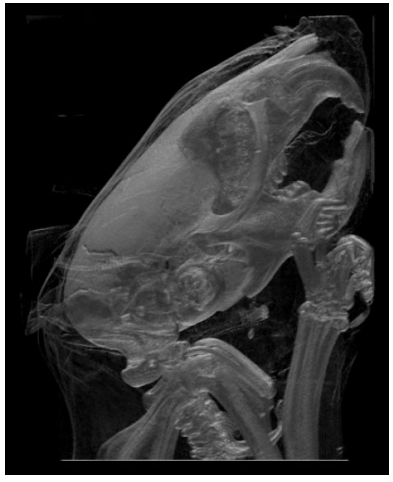
współrzędne barycentryczne punktu P względem trójkąta $\triangle ABC$:

$$P = uA + vB + wC, \text{ gdzie } u + v + w = 1$$

można policzyć ze stosunków pól np. $u = \frac{Area(PBC)}{Area(ABC)}, \ldots$

Reprezentacje: wolumetryczne

- Dane określające własności pewnej objętości: in/out lub np. gęstość: bit, byte, short, float,... na każdy **woksel/Voxel**
- Zastosowania medyczne np. różne wersje tomografii komputerowej: CT, MRT, PET, ...



Wizualizacja skanu CT czaszki myszy, Źródło: Wiki, użytkownik Lackas

- Duża ilość danych: nawet przy niedużej rozdzielczości 512x512x512 potrzebujemy ...

Reprezentacje: zbiory wokseli

Najprostsza reprezentacja: tablica wartości dla wokseli np. 1 bajt na woksel

Figure 14.21: The game Minecraft models the entire world with 1 m^3 voxels, enabling efficient, real-time illumination, simulation, and rendering for fully dynamic Earth-scale worlds.

Źródło: Hughes,van Damm – “Computer Graphics – Principles and Practice”, 3rd Edition, Addison–Wesley 2014:

Reprezentacje: maszerujące sześciany

Metoda maszerujących sześcianów: szeroko używana w wizualizacji danych wolumetrycznych metoda tworzenia dwuwymiarowych izopowierzchni.

Np. dla rekonstrukcji izopowierzchni o zadanej gęstości granicznej w tomografii dla uzyskania obrazu kości.

Dla wersji 3D dużo przypadków dlatego zajmiemy się prostszym ale analogicznym algorytmem 2D (maszerujące kwadraty):

Wersja 2D: 16 przypadków wartości znaku w rogach dla wyrażenia $F(x, y) - c > 0$

Większość poza dwoma jednoznacznie wyznacza izolinie. Jak wybrać dla przypadków niejednoznacznych i jakie są skutki ?

Reprezentacje: drzewa ósemkowe i BSP

Drzewa ósemkowe (w 2D drzewa czwórkowe): dzielimy aż do dokładności jednego woksela dzięki powtarzalności oszczędzamy pamięć.

Np. dla powierzchni sfery mamy: $O(n^2)$ zamiast $O(n^3)$

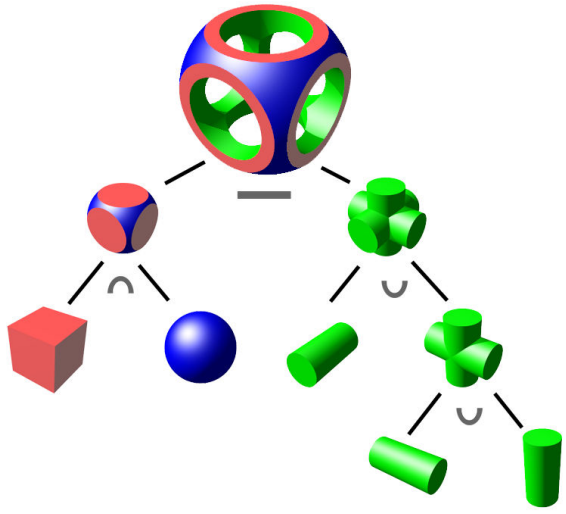
Drzewa BSP (Binary Space Partitioning): podział płaszczyzny lub przestrzeni.

- w liściach 1/0 (in/out) lub lista obiektów w danym kawałku przestrzeni.
- niejednoznaczność reprezentacji,
- dowolne podziały czy prostopadłe do osi układu współrzędnych ?
- poza reprezentacja także algorytmy oparte na BSP np. dla widoczności, czy też przyspieszanie metody śledzenie promieni.

Reprezentacje: CSG

CSG - Constructive Solid Geometry:
Pewien zbiór obiektów prymitywnych:
prostopadłościany, kule, cylindry, stożki,...
Operacje boolowskie:
przekrój, suma, dopełnienie.

$$(A \cap B) \setminus C$$



Źródło: Wiki, użytkownik Zottie

- Naturalność modelowania (wywiercenie — wycięcie).
- Drzewo lub wyrażenie algebraiczne opisują nam wyciętą część przestrzeni.
- Jaki jest Minimalny zbiór obiektów?
- Do wizualizacji najczęściej potrzebna konwersja do innego formatu.

Level of Detail: LoD

Literatura w kontekście zastosowań do renderowania terenu: P. Cozzi, K.Ring – "3D Engine for Virtual Globes", CRC Press 2011, a także na stronie: `http://vterrain.org/LOD`.

Ogólna idea: dla modelu/siatki możemy żądać jego wersji w jednej z mniejszych rozdzielczości lub tą rozdzielczość (LoD) dobierać lokalnie.

Cele: efektywność i poprawność

1. Generowanie
2. Wybór LoD (kryteria: ilość trójkątów, FPS, dokładność)
3. Przełączanie

Level of Detail: klasyfikacja metod

Dyskretne LoD: mamy ustalony zbiór przeliczonych siatek dla różnych poziomów szczegółowości od najdokładniejszej $N = 0, 1, \ldots, N$

Ciągłe LoD: tworzymy uproszczone siatki w ciągły sposób przez lokalne operacje: kolapsowanie krawędzi, rozdzielanie wierzchołka
CLOD (Lindstrom et al., 1996), ROAM (Duchaineau et al., 1997), Progressive meshes (Hoppe, 1998)

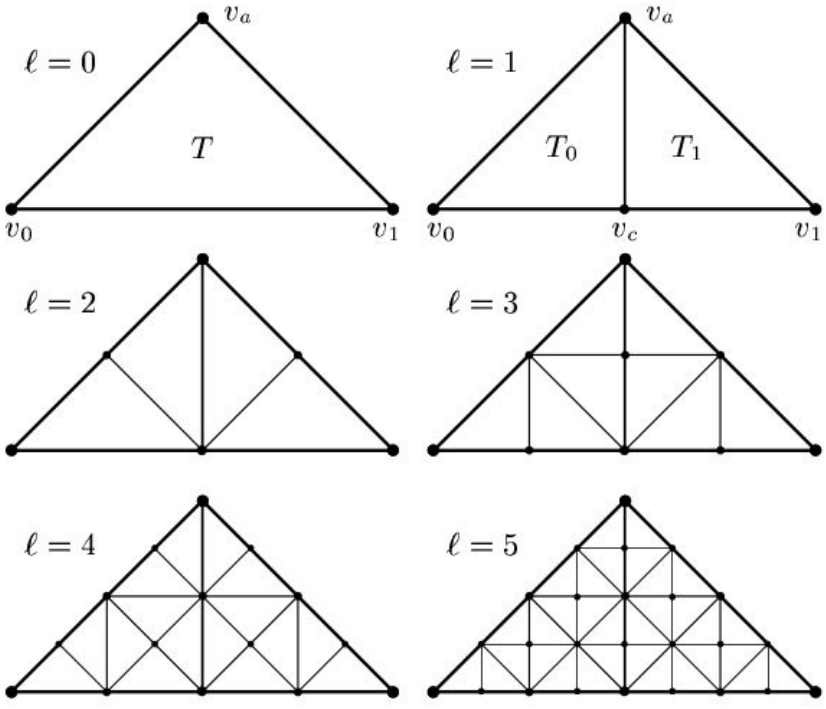
Hierarchiczny LoD: Cel zredukowanie transferu CPU–GPU, dane dzielone na porcje/kafelki często pamiętane w drzewie czwórkowym
Clipmaps (Tanner et al., 1998; Asirvatham, Hoppe 2005), HLOD (Erikson et al., 2001)

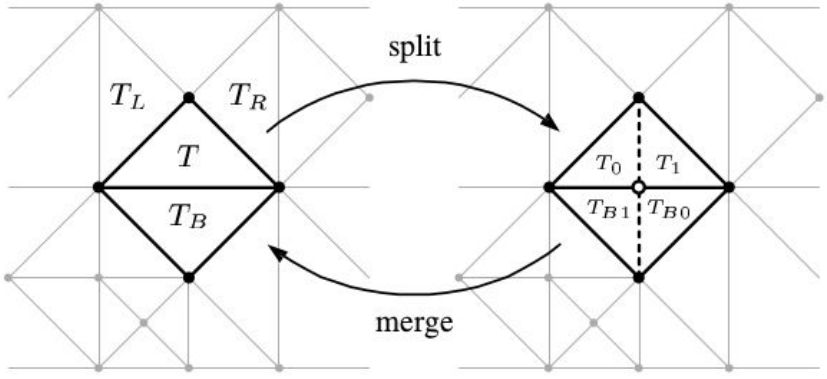
Level of Detail: artefakty

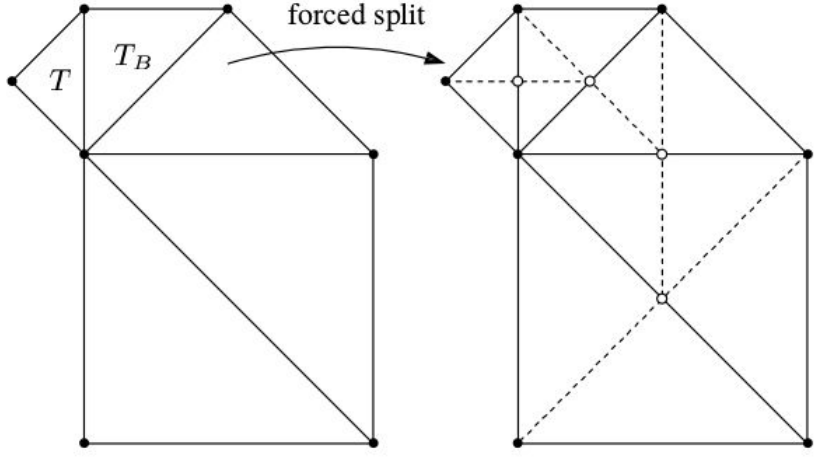
1. Cracking
2. T–juncions
3. Popping

Level of Detail: ROAM

M.Duchaineau et al., – "ROAMing Terrain: Real- Time Optimally Adapting Meshes.", Proceedings of the 8th Conference on Visualization '97.







Level of Detail: progressive meshes

H.Hoppe – "Smooth View-Dependent Level-of-Detail Control and Its Application to Terrain Rendering.", Proceedings of the Conference on Visualization '98.

także: Hughes, van Damm... strony 649–651.

Ciąg siatek powstałych przez kolapsowanie po 1 krawędzi (tworzymy kolejke priorytetową krawędzi do kolapsowania).

Koszt kolapsowania to różnica energii po i przed:

$$\Delta M = E(M') - E(M)$$

Energia:

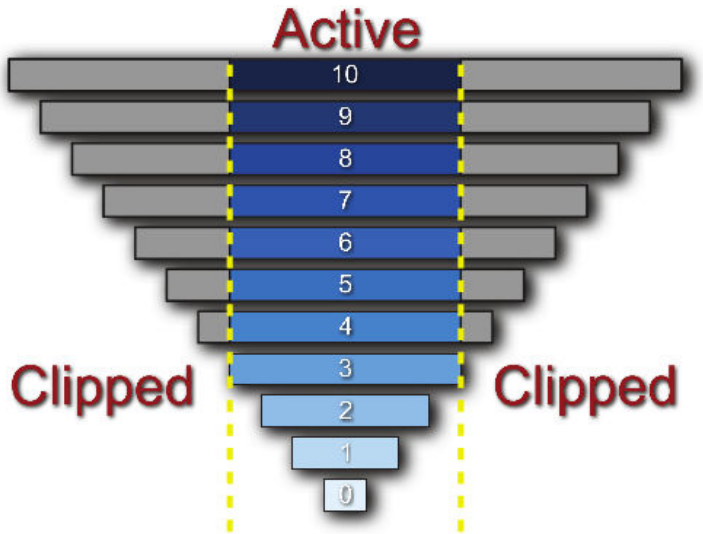
$$E(M) = E_{dist}(M) + E_{spring}(M) + E_{other}(M)$$

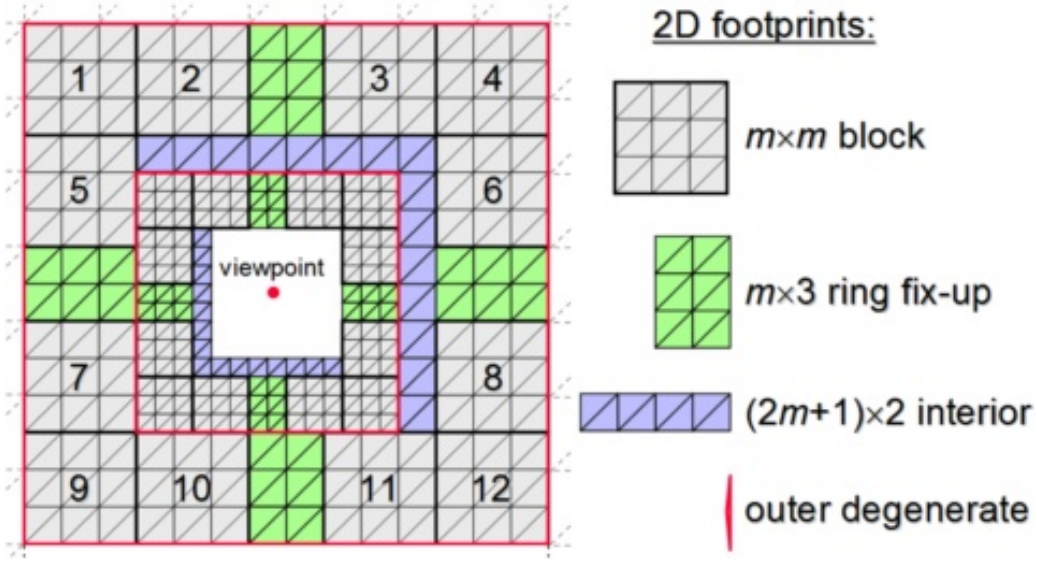
E_{dist} — opisuje sumę kwadratów różnic odległości wierzchołków, zaś:

$$E_{spring}(M) = \sum_{Edge(v_i, v_j) \in M} \kappa ||v_i - v_j||$$

Level of Detail: Clipmaps

A.Asirvatham, H.Hoppe – "Real-time terrain rendering with all data processing on the GPU", GPU Gems 2, 2005.





Źródła: P. Cozzi, K.Ring – "3D Engine for Virtual Globes", CRC Press 2011 oraz strona oryginalnej pracy H.Hoppe

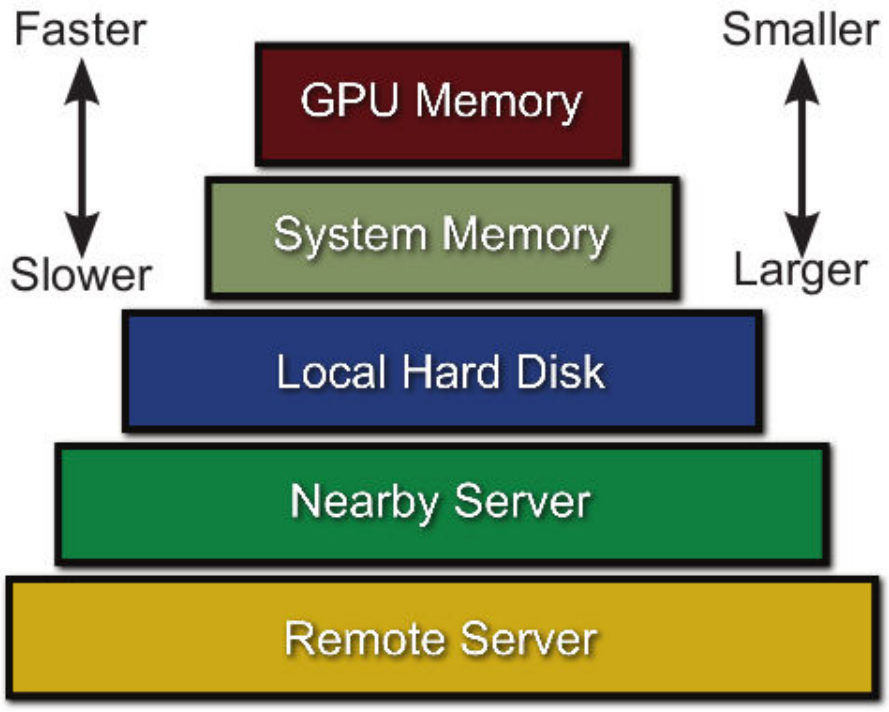
Piramida kolejnych poziomów MipMaps, obciętych do ustalonego rozmiaru dla otoczenia położenia obserwatora.

Kafelki na GPU mogą być ładowane w razie potrzeby.

Level of Detail: duże dane i cache'owanie

OOC — Out–of–core rendering

Hierarchia cache'y:



Źródło: P. Cozzi, K.Ring – "3D Engine for Virtual Globes", CRC Press 2011, str.382