Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Knihovna pro výpočet průniku triangulace a voroného diagramu ve 2D a 3D

KIV/VAM

1. února 2021 Dominik Poch

Obsah

1 Zadání

Mějme danou doménu 2D prostoru (obdélník), ve které je definována triangulace T a Voroného diagram V, přičemž obě tyto struktury jsou obecné a navzájem nezávislé. Určete množinu konvexních oblastí C takových, že každá oblast je průnikem právě jednoho trojúhelníku t z T a jedné Voroného oblasti v z V tak, že C pokrývá celou vstupní doménu. Při konstrukci C využijte faktu, že hranice oblastí z množiny C vždy představují buďto přechod do jiné Voroného oblasti nebo do jiného trojúhelníku. Pokud budou vstupní struktury T a V vybaveny vhodnými datovými reprezentacemi poskytujícími informaci o sousednosti, pak je možné množinu C určit průchodem (traversováním) se složitostí odvozenou primárně od $|C|$ a nikoli od $|V| * |T|$. Řešení konstruujte tak, aby bylo možné zobecnění do 3D a otestujte ho na náhodně vygenerovaných planárních T a V.

2 Datová reprezentace

Vstupní struktury, jmenovitě Voroného diagram V a triangulace T, jsou v knihovně reprezentovány třídou `VolumeData`, respektive `VolumeData2D` a `VolumeData3D`, které obsahují seznam buněk `Cell`, z nichž každá buňka obsahuje seznam poloprostorů `Edge`, kterými je definována. Poloprostory jsou popsány obecnou nerovnicí

$$\mathbf{a} \cdot \mathbf{x} + b \leq 0$$

a obsahují odkaz na zdrojové a cílové buňky. Přičemž je dáno, že normála poloprostoru míří vždy směrem do zdrojové buňky. Odkazy na buňky budou později sloužit k traversování.

2.1 Převod triangulace

Pro převod triangulace T do datové struktury slouží `VolumeData.FromTriangulation`. Vstupem jsou vrcholy triangulace a seznam buněk (trojúhelníků či tetrahedronů) s indexy vrcholů.

Algoritmus pro převod triangulace převádí buňky na seznam poloprostorů. Pro každý trojúhelník je vykonáno několik kroků:

1. Spočítá se centroid trojúhelníka.
2. Každá hrana se převede na poloprostor.
3. Pro každou hranu se nastaví zdrojová a cílová buňka.

Složitost algoritmu je $O(n)$, kde n je počet buněk triangulace.

2.1.1 Převod hrany na poloprostor

Pro převod hrany na poloprostor je nutné vypočítat obecnou nerovnici poloprostoru.

Ve 2D se z vrcholů hrany trojúhelníka určí vektor a následně jeho normála. Skalárním součinem normály s jedním z vrcholů hrany se vypočítá poslední člen obecné rovnice.

Pořadí vrcholů není předem známé, proto se pro kontrolu orientace normály dosadí centroid trojúhelníka do obecné rovnice. Pokud normála směřuje mimo zdrojovou buňku, jsou otočena znaménka prvků obecné rovnice. Pokud je výsledek záporný, změní se znaménka členů obecné rovnice.

Ve 3D je použit podobný postup. Normála se vypočítá vektorovým součinem dvou hran stěny tetrahedronu a následně se pokračuje stejně jako ve dvou rozměrech.

2.1.2 Nalezení cílové buňky

Určení zdrojové buňky poloprostoru je přímočaré, jelikož se výpočet provádí pro každou hranu buňky. Je tudíž předem známá. Nalezení cílové hrany již tak jasné není. Proto jsou indexy vrcholů hran a nově vytvořený objekt poloprostoru ukládány do slovníku. Pokud již byla hrana jednou navštívena v jiném trojúhelníku, nalezne se ve slovníku a nastaví se cílové buňky v novém i existujícím objektu.

2.2 Převod Voroného diagramu

Převod Voroného diagramu je umožněn metodou `VolumeData.FromVoronoi`. Zadán je pouze seznam generátorů Voroného diagramu. Pro výpočet diagramu z generátorů byla využita knihovna MIConvexHull[1]. Knihovna vrací sousednost mezi jednotlivými generátory pomocí trojúhelníků ve 2D a tetrahedronů ve 3D. Vrcholy jednoho trojúhelníku či tetrahedronu jsou tedy sousední Voroného buňky. Což je pro algoritmus převodu ideální.

Pro každé dvojice sousedů jsou vytvořeny dvě Voroného buňky. Aby se buňky neduplikovaly, je jejich existence kontrolována podle indexu příslušného generátoru. Pokud buňka pro generátor již existuje, je pouze vrácena ze slovníku, do kterého se buňky ukládají. Když buňka dosud neobsahovala poloprostor s odkazem na druhou z dvojice buněk, proběhne výpočet poloprostoru. Poloprostor představující hranu Voroného buňky se získá výpočtem vektoru mezi sousedními vrcholy (normála hrany Voroného buňky) a skalárním součinem tohoto vektoru s bodem v polovině hrany trojúhelníku.

3 Výpočet průniků

Výpočet průniků je implementován ve třídě `VolumeIntersection`, `VolumeIntersection2D` a `VolumeIntersection3D`. Třída obsahuje metodu `VolumeIntersection.Intersect`, která počítá průniky mezi triangulací a Voroného diagramem. Vstupem metody jsou vrcholy a buňky triangulace a seznam generátorů Voroného diagramu. Z nich jsou vytvořeny datové struktury `VolumeData`, které se následně procházejí.

3.1 Průchod strukturami

Průchod začíná na první buňce triangulace. Počáteční buňka Voroného diagramu je zvolena tak, aby obsahovala centroid první buňky triangulace. V průběhu průchodu jsou ukládány dvojice již navštívených buněk (jedna z triangulace a druhá z Voroného diagramu) do množiny, která kontroluje, zda byly dvojice již navštíveny. Dosud nezpracované dvojice se v průběhu procházení strukturou ukládají do fronty, kterou algoritmus postupně zpracovává, dokud není prázdná. Dvojice buněk jsou z fronty postupně vybírány a je vypočítán jejich průnik. Výpočet průniku vrací minimální počet poloprostorů, které průnik definují a jejich průsečíky. Následně je pro každý vrácený poloprostor vytvořen nový pár. Jak bylo zmíněno v kapitole , každý poloprostor obsahuje odkaz na svojí zdrojovou a cílovou buňku. Zdrojová buňka poloprostoru se porovná s buňkami v páru a v případě, že se jedna z nich rovná je vytvořen nový pár s cílovou buňkou a druhou buňkou

[1] https://github.com/DesignEngrLab/MIConvexHull

z páru. U triangulace může nastat situace, že hrana žádnou cílovou buňku nemá, jelikož byla na hranici triangulace. V takovém případě se nový pár nevytváří. Pokud ještě nebyl nový pár navštívený, přidá se do fronty.

3.2 Průnik dvou buněk

Průnik dvou buněk složených z poloprostorů je realizován brute-force algoritmem se složitostí $O(n^3)$ pro dvourozměrná a $O(n^4)$ pro třírozměrná data, kde n je počet poloprostorů tvořících jednu buňku.

Ve 2D je vypočítán průsečík pro každé dvojice poloprostorů, zatímco ve 3D je vypočítán průsečík všech trojic. Následně se zkontroluje, zda průsečík leží uvnitř obou buněk. Pokud ano, tak jsou poloprostory přidány do výsledku. Tento způsob výpočtu však přináší několik úskalí, které bylo potřeba vyřešit.

V prvé řadě se jedná o to, že se průsečíky mohou opakovat. Tomu bylo zabráněno ukládáním průsečíků do množiny. Avšak porovnání souřadnic bodů je problematické a bylo potřeba využít Eps test. Pokud je absolutní hodnota rozdílu souřadnice menší než Eps, pak se body rovnají.

Mohou se také objevit poloprostory, jejichž průsečík se nachází v obou buňkách, ale netvoří minimální průnik. Oba problémy jsou dobře vidět na . Zde průsečík dvou poloprostorů leží zároveň i na jiné hraně. Z obrázku je patrné, že je tato hrana nadbytečná a průsečík bude nalezen vícekrát. K odstranění tohoto problému byl pro uložení výsledných poloprostorů použit slovník, který pro každý poloprostor ukládá unikátní průsečíky. Který poloprostor je pro průnik nezbytný, lze odvodit z počtu průsečíků. Ve 2D musí mít každý poloprostor právě dva průsečíky (jedná se o úsečky). Ve třech rozměrech je nejmenší možnou stěnou trojúhelník. Poloprostor, tak musí generovat alespoň tři průsečíky. Všechny poloprostory, které těmto kritériím nevyhovují, jsou nakonec ze slovníku odstraněny.

Zda průsečík leží uvnitř buněk je zjištěno dosazením jeho souřadnic do obecné rovnice každého poloprostoru, který definuje buňku. Výsledek je porovnán Eps testem a pokud je výsledná hodnota menší než -Eps, tak bod leží mimo buňku.

Eps je nastaveno ve třídě `MathUtils`. V průběhu testování byla hodnota Eps ustálena na $1E-10$.

3.3 Výpočet průsečíku

Výpočet průsečíků vyžívá principu duality mezi bodem a přímkou v projektivním prostoru P^2 a mezi bodem a rovinou v P^3. Z dvou přímek se vypočítá jejich průsečík pomocí determinantu

$$\begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{vmatrix}$$

kde $\mathbf{i} = [1,0,0]$, $\mathbf{j} = [0,1,0]$ a $\mathbf{k} = [0,0,1]$, a, b a c jsou členy obecné rovnice přímky. Výsledkem determinantu je bod $[x, y : w]$, který je průsečíkem v projektivním prostoru. Souřadnice bodů jsou následně vyděleny homogenní složkou w a vrchol je převeden do E^2. Pro průnik rovin je nutné vypočítat determinant 4x4.

(a) Triangulace (b) Voronoi diagram (c) Průniky

Obrázek 1: Ukázka malých dat

4 Vizualizace průniků

Pro kontrolu správnosti výpočtů byla implementována vizualizace 2D a 3D dat. V obou případech jsou data vykreslena do bitmapy. Každý pixel bitmapy je přemapován na bod uvnitř bounding boxu poskytnutých dat. Pro každý bod je vyhledána buňka, která bod obsahuje a pixelu je přiřazena náhodná barva vygenerovaná pro jednotlivé buňky. V případě 3D dat je vizualizován pouze slice volumetrických dat na zadané hodnotě z souřadnice. Ukázku malých dat s 10 vrcholy a 15 tetrahedrony lze vidět na Obrázku 1.

5 Výpočet centroidů

Mimo rozsah původního zadání byl do práce přidán výpočet centroidů nalezených průniků. Průniky mohou být konvexní polygony či polyhedrony. Jejich centroid není možné spočítat přímo. K výpočtu se využívá vážená suma centroidů m trojúhelníků, z nichž se polygon skládá.

Váha w_i jednotlivých trojúhelníků se vypočítá pomocí vzorce pro výpočet plochy trojúhelníka

$$w_i = \frac{1}{2}|x_1 y_2 \cdot x_2 y_1|$$

kde $AB = (x_1, y_1)$ a $AC = (x_2, y_2)$. Pro tetrahedron je vzorec

$$w_i = \frac{1}{6}|\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c})|$$

kde **a**, **b** a **c** jsou vektory hran tetrahedronu. Centroid trojúhelníka a tetrahedronu je počítán jako

$$C_i = \frac{1}{n}\sum_{j=0}^{n} X_j$$

kde X_j jsou vrcholy trojúhelníka (tetrahedronu) a n je jejich počet. Pokud definuji sumu vah přes všechny trojúhelníky

$$W = \sum_{i=0}^{m} w_i$$

pak centroid polygonu nebo polyhedronu je

$$C = \frac{1}{W} \sum_{i=0}^{m} w_i C_i$$

Triangulace polygonu a polyhedronu je počítána knihovnou MIConvexHull, která kromě Voroného diagramu poskytuje také výpočet Delaunay triangulace.

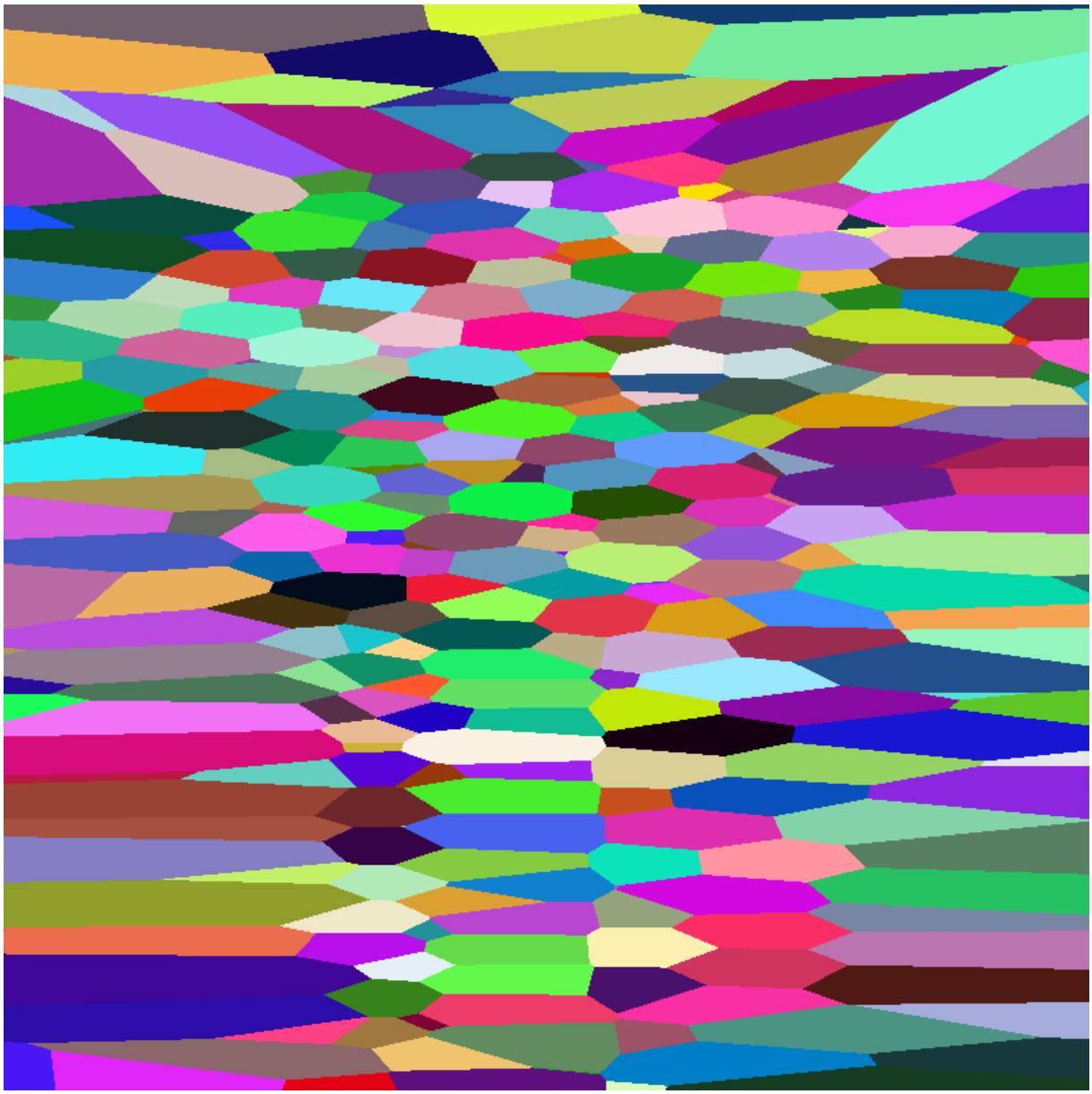
6 Závěr

Algoritmus pro výpočet průniků funguje spolehlivě pro dodaná praktická data. Poskytnutá tetrahedronizace obsahovala 9065 vrcholů a 35132 tetrahedronů. Generátorů Voroného diagramu bylo 1000. Vizualizace triangulace, Voroného diagramu a výsledku s výpočtem průniků lze vidět na Obrázcích 2, 3 a 4.

Problémy se však mohou objevit v případě, kdy dojde k selhání výpočtu způsobené nedostatečnou přesností reprezentace čísel v počítači. Důvodem by mohla být příliš hustá triangulace s velmi blízkými vrcholy a existence velmi malých průniků. Z důvodu výpočtu centroidů pro průniky může dojít i ke kompletnímu odstranění průniku z datové množiny. A to v případě, že vrcholy průniku leží na přímce a není možné je triangulovat. O takové situaci je však uživatel informován pomocí chybového hlášení. Napříč algoritmy se navíc vyskytuje několik Eps testů, které také mohou zavést nepřesnosti do výpočtu.

Práci by bylo možné dále rozšířit implementací efektivnějšího algoritmu pro výpočet průniku z poloprostorů, který však nebyl vyžadován. Pro výše uvedená testovací data trval výpočet průniků bez vizualizace 35 sekund na počítači s procesorem Intel(R) Core(TM) i5-3570 3.40GHz a 8 GB RAM.

Obrázek 2: Triangulace



Obrázek 3: Voronoi diagram

Obrázek 4: Průniky