

Erkennen eines Graphen aus einer Punktwolke

Wir sind hier:

Einleitung

Vorverarbeitung

Implementierung des Algorithmus

Der Algorithmus

Umsetzung in Java

Graphische Darstellung der Ergebnisse

Organisation

Das Problem

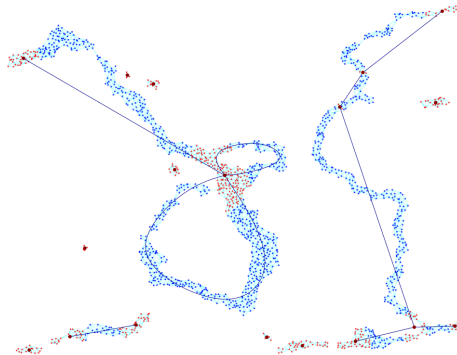
Eingabe ein metrischer Raum (X, d) , z.B.

1. GPS-Koordinaten eines Autos, das in einer Stadt herumfährt
2. Koordinaten der schwarzen Pixel in einem Schwarz-Weiß-Bild

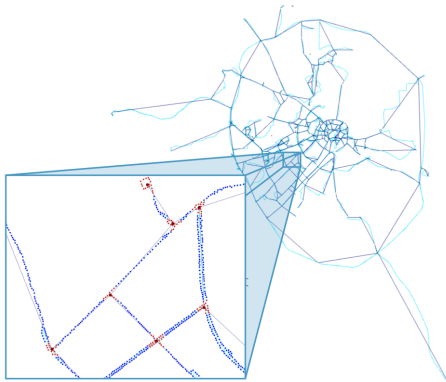
Ausgabe eine Approximation des metrischen Graphen, der dem metrischen Raum zugrunde liegt, z.B.

1. das Straßennetz der Stadt
2. der Graph, der im Bild abgebildet wird
→ Schrifterkennung

Beispiel (Erdbeben)



Beispiel (Straßennetz)



Aufgabenbereiche

1. Vorverarbeitung (Daniel, Terese)
2. Implementierung des Algorithmus (Lea, Mahmoud, Moritz)
3. graphische Darstellung der Ergebnisse (Jiang)

Wir sind hier:

Einleitung

Vorverarbeitung

Implementierung des Algorithmus

Der Algorithmus

Umsetzung in Java

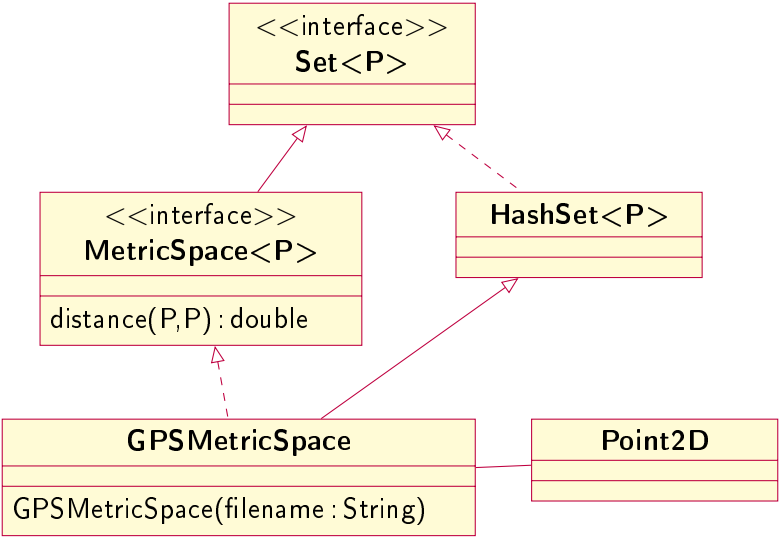
Graphische Darstellung der Ergebnisse

Organisation

Vorverarbeitung

1. Menge von Punkten erstellen
 - ▶ Koordinaten aus GPS-Spur-Dateien extrahieren
 - ▶ Koordinaten der schwarzen Pixel eines Bildes bestimmen
2. ggf. Punktmenge auf repräsentative Teilmenge reduzieren
 - ▶ metrisches ε -Netz
 - ▶ furthest point sampling
3. auf die Punktmenge basierenden α -Komplex erstellen
 - ▶ α -Komplex: Unterkomplex der Delaunay-Triangulierung
 - ▶ nur nahe Punkte werden durch eine Kante verbunden
4. Abstandsmethode implementieren
 - ▶ $d(x, y) :=$ Länge des kürzesten Pfades zwischen x und y

Klassendiagramm



Wir sind hier:

Einleitung

Vorverarbeitung

Implementierung des Algorithmus

Der Algorithmus

Umsetzung in Java

Graphische Darstellung der Ergebnisse

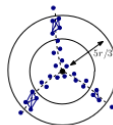
Organisation

Drei wesentliche Schritte

- 1 Bestimmen, ob Punkte zu Knoten oder Kante im späteren Graphen gehören → Punkte mit entsprechendem Label versehen
- 2 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammensetzen → Rekonstruktion des Graphen
- 3 Kanten mit Abständen versehen → Metrik wiederherstellen

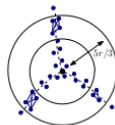
Zu Schritt 1

- 1 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammensetzen →
Rekonstruktion des Graphen
 - Betrachten einen kreisförmigen Ausschnitt um jeden Punkt herum und bestimmen Anzahl der Zusammenhangskomponenten darin



Zu Schritt 1

- 1 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammensetzen → Rekonstruktion des Graphen
 - ▶ Betrachten einen kreisförmigen Ausschnitt um jeden Punkt herum und bestimmen Anzahl der Zusammenhangskomponenten darin

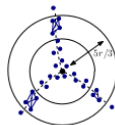


- ▶ Grad des Punkts = Anzahl der Zusammenhangskomponenten im kreisförmigen Ausschnitt

Zu Schritt 1

- 1 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammensetzen → Rekonstruktion des Graphen

- ▶ Betrachten einen kreisförmigen Ausschnitt um jeden Punkt herum und bestimmen Anzahl der Zusammenhangskomponenten darin



- ▶ Grad des Punkts = Anzahl der Zusammenhangskomponenten im kreisförmigen Ausschnitt
- ▶ Grad == 2 → edge point (wird später zu einer Kante gehören)
- ▶ Grad != 2 → preliminary branch point (wird später zu einem Knoten gehören)
- ▶ Alle Punkte, die weniger als x von einem primären branch point entfernt sind, werden ebenfalls als branch point eingeordnet

Zu Schritt 2

- 2 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammensetzen →
Rekonstruktion des Graphen

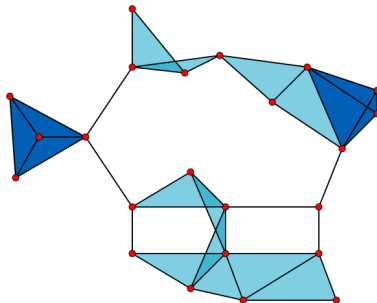
Zu Schritt 2

- 2 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammenschließen →
Rekonstruktion des Graphen
- ▶ Edge und branch points werden in zwei Mengen aufgeteilt
 - ▶ um die Anzahl der späteren Kanten und Knoten zu bestimmen, wird Rips-Vietoris-Graph erstellt

Zu Schritt 2

2 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammensetzen →
Rekonstruktion des Graphen

- ▶ Edge und branch points werden in zwei Mengen aufgeteilt
- ▶ um die Anzahl der späteren Kanten und Knoten zu bestimmen, wird Rips-Vietoris-Graph erstellt
 - ▶ Im Rips-Vietoris-Graphen werden alle Punkte, die innerhalb eines gewissen Durchmessers liegen, zu einer Zusammenhangskomponente gefasst



Zu Schritt 2

- 2 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammensetzen → Rekonstruktion des Graphen
 - ▶ Edge und branch points werden in zwei Mengen aufgeteilt
 - ▶ um die Anzahl der späteren Kanten und Knoten zu bestimmen, wird Rips-Vietoris-Graph erstellt

Zu Schritt 2

- 2 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammenschließen → Rekonstruktion des Graphen
- ▶ Edge und branch points werden in zwei Mengen aufgeteilt
 - ▶ um die Anzahl der späteren Kanten und Knoten zu bestimmen, wird Rips-Vietoris-Graph erstellt
 - ▶ Rips-Vietoris-Graphen für Menge der edge points und Menge der branch points erstellen

Zu Schritt 2

- 2 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammenschließen → Rekonstruktion des Graphen
 - ▶ Edge und branch points werden in zwei Mengen aufgeteilt
 - ▶ um die Anzahl der späteren Kanten und Knoten zu bestimmen, wird Rips-Vietoris-Graph erstellt
 - ▶ Rips-Vietoris-Graphen für Menge der edge points und Menge der branch points erstellen
 - ▶ Jede Zusammenhangskomponente wird zu einem Knoten, bzw. einer Kante vereint

Zu Schritt 2

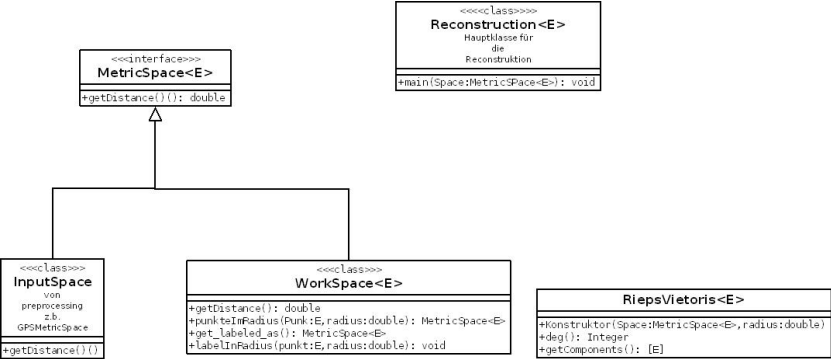
- 2 Anhand der Labels bestimmen, welche Punkte sich zu einem Knoten und welche sich zu einer Kante zusammenschließen → Rekonstruktion des Graphen
- ▶ Edge und branch points werden in zwei Mengen aufgeteilt
 - ▶ um die Anzahl der späteren Kanten und Knoten zu bestimmen, wird Rips-Vietoris-Graph erstellt
 - ▶ Rips-Vietoris-Graphen für Menge der edge points und Menge der branch points erstellen
 - ▶ Jede Zusammenhangskomponente wird zu einem Knoten, bzw. einer Kante vereint
 - ▶ Zwei Punkte werden durch eine Kante verbunden, wenn sie Punkte in ihrer Zusammenhangskomponente haben, die nur einen gewissen Abstand zu Punkten in der selben Zusammenhangskomponente einer Kante haben

Zu Schritt 3

3 Kanten mit Abständen versehen \rightarrow Metrik wiederherstellen

- ▶ Jeder Kante wird als Länge der Durchmesser ihrer Zusammenhangskomponente zugewiesen

Klassendiagramm



Verbesserung der Laufzeit

- ▶ Zunächst naive Implementierung, bis der Algorithmus läuft
- ▶ Dann eventuell Verbesserungen
 - ▶ Gitter

Wir sind hier:

Einleitung

Vorverarbeitung

Implementierung des Algorithmus

Der Algorithmus

Umsetzung in Java

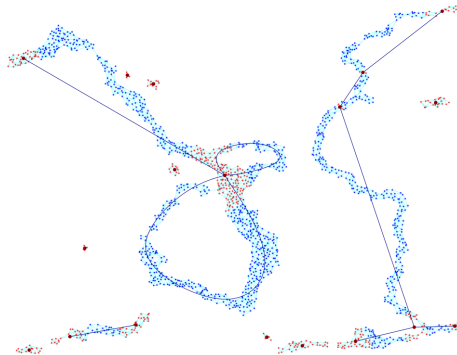
Graphische Darstellung der Ergebnisse

Organisation

Graphische Oberfläche

- ▶ Datei auswählen
 - ▶ GPS-Spur (GPS Exchange Format [.gpx])
 - ▶ Schwarz-Weiß-Bild (beliebiges Bildformat)
- ▶ Parameter des Algorithmus setzen (?)
- ▶ ursprüngliche Punktmenge wird zusammen mit dem ausgegebenen metrischen Graphen angezeigt

Beispiel (Erdbeben)



Wir sind hier:

Einleitung

Vorverarbeitung

Implementierung des Algorithmus

Der Algorithmus

Umsetzung in Java

Graphische Darstellung der Ergebnisse

Organisation

Organisation

- ▶ Entwicklungsumgebung: Eclipse
- ▶ Versionsverwaltung: Git (in Eclipse integriert)
- ▶ Aufgabenverwaltung: Trello
 - ▶ zu jeder Aufgabe wird eine Trello-Karte erstellt
- ▶ Dokumentation: Javadoc (und Git Wiki?)

Trello

