Übungen - Bildgenierung

Übung 03.

Jose Jimenez

Angewandte Informatik
Bergische Universität Wuppertal

November 9, 2022

Table of Contents

Aufgabe 07 Gefüllte Polygone

Gefüllte Polygone

Ihr habt schon den Algorithmus 3.22 mit Dr. Galgon entwikelt.

Sortierung in horizontaler Anordnung; es gibt keine sich überschneidenden Kanten, deshalb liegt eine Kante "links neben" einer weiteren Kante, wenn:

Aufgabe 07 Gefüllte Polygone

Rahmen Program

Sortierung in horizontaler Anordnung; es gibt keine sich überschneidenden Kanten, deshalb liegt eine Kante "links neben" einer weiteren Kante, wenn:

- sie am (zur Zeit) untersten Punkt links neben der zweiten Kante liegt, oder
- beide Kanten zwar an der selben x-Koordinate beginnen, die Erste aber eine steilere Steigung hat.

Aufgabe 07 Gefüllte Polygone

Rahmen Program

Sortierung in horizontaler Anordnung; es gibt keine sich überschneidenden Kanten, deshalb liegt eine Kante "links neben" einer weiteren Kante, wenn:

- sie am (zur Zeit) untersten Punkt links neben der zweiten Kante liegt, oder
- beide Kanten zwar an der selben x-Koordinate beginnen, die Erste aber eine steilere Steigung hat.

Die oberen Punkte spielen dabei keine Rolle (es gibt in dieser Variante keine Liste für Kanten, die an einer bestimmten y-Koordinate inaktiv werden).

Aufgabe 07 Gefüllte Polygone

Rahmen Program

Sortierung in horizontaler Anordnung; es gibt keine sich überschneidenden Kanten, deshalb liegt eine Kante "links neben" einer weiteren Kante, wenn:

- sie am (zur Zeit) untersten Punkt links neben der zweiten Kante liegt, oder
- beide Kanten zwar an der selben x-Koordinate beginnen, die Erste aber eine steilere Steigung hat.

Die oberen Punkte spielen dabei keine Rolle (es gibt in dieser Variante keine Liste für Kanten, die an einer bestimmten y-Koordinate inaktiv werden).
Die zweite Bedingung ist nötig, um neu hinzukommende Kanten einfach einsortieren zu können, ohne die alten Kanten neu sortieren zu müssen.

Aufgabe 07 Gefüllte Polygone

Rahmen Program

Sortierung in horizontaler Anordnung; eine Kante liegt "links neben" einer weiteren Kante, wenn:

- beide Kanten zwar an der selben x-Koordinate beginnen, die Erste aber eine steilere Steigung hat.

Die Bedingung ist nötig, um neu hinzukommende Kanten einfach einsortieren zu können, ohne die alten Kanten neu sortieren zu müssen.

```
struct Kante{
    ...
    friend bool operator<(const Kante& k1, const Kante& k2)
     {
     if (k1.x != k2.x)
        return k1.x < k2.x;
      else
       return k1.einsdurchm < k2.einsdurchm;
} // für die sortierung, mann kann einfach e.g. kanten_list.sort();
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: Kante struct

```
struct Kante{
    int ymax; // maximale y-Koordinate
    double x; // AKTUELLE x-Koordinate
    double einsdurchm; // Da die Kanten in y-Richtung ausgewertet werden,
                       // wird die inverse Steigung benötigt.
    friend bool operator<(const Kante& k1, const Kante& k2){
        if (k1.x != k2.x)
            return k1.x < k2.x;
        else
            return k1.einsdurchm < k2.einsdurchm;
    }
};

typedef list<Kante> KantenTabelle;
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
void drawPatternPolygon( Drawing& pic, const vector<IPoint2D>& ecken,
                         const QImage& muster ){
  // HIER ERGÄNZEN
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
void drawPatternPolygon( Drawing& pic, const vector<IPoint2D>& ecken,
                         const QImage& muster ){
  // HIER ERGÄNZEN
}
```

Ok, wir arbeiten mit...

- Drawing pic
- Vector von IPoint2D
- int: colour
- KantenTabelle

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
void drawPatternPolygon( Drawing& pic, const vector<IPoint2D>& ecken,
                         const QImage& muster ){
  // HIER ERGÄNZEN
}
```

Ok, wir arbeiten mit...

- Drawing pic
- Vector von IPoint2D
- int: colour
- KantenTabelle

Wir laufen von y_{min} bis y_{max}.
Und.. Wie viele ecken haben wir? (C++)

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Ok, wir arbeiten mit...

- Drawing pic
- Vector von IPoint2D
- int: colour
- KantenTabelle

```
void drawPatternPolygon( Drawing& pic, const vector<IPoint2D>& ecken,
                         const QImage& muster ){
  int ymin = pic.getHeight() + 1;
  int ymax = -1;
  unsigned int n = ecken.size();
}
```

OK, dann... Wir haben ein Cpp-vector von Ecken (IPoint2D).
Wir mussen die kleinste und größte Bildzeile bestimmen. Wie?

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

OK, dann... Wir haben ein Cpp-vector von Ecken (IPoint2D).
Wir mussen die kleinste und größte Bildzeile bestimmen. Wie?

```
void drawPatternPolygon( Drawing& pic, const vector<IPoint2D>& ecken,
                         const QImage& muster ){
  ...
  for (unsigned int i = 0; i < n; i++)     {
       if (ecken[i].y < ymin)
         ymin = ecken[i].y;
       if (ecken[i].y > ymax)
         ymax = ecken[i].y;
     }

  ...
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Active edge table (AET): Tabelle der momentan aktiven Kanten. Enthält alle Kanten, zwischen denen in der aktuellen Zeile spans gezeichnet werden müssen. Die Kanten sind horizontal sortiert (struct Kante).
Wie intialisiert man eine leere Liste?

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Active edge table (AET): Tabelle der momentan aktiven Kanten. Enthält alle Kanten, zwischen denen in der aktuellen Zeile spans gezeichnet werden müssen. Die Kanten sind horizontal sortiert (struct Kante).
Wie intialisiert man eine leere Liste?

```
    void drawPatternPolygon( Drawing& pic, const vector<IPoint2D>& ecken,
                             const QImage& muster ){
  ...

    KantenTabelle aet(0);


  ...
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Wir haben:

- kleinste und größte Bildzeile (y_{min} und y_{max}).
- Punkteliste (Ecken[i]). $i = 0, ...,$ecken.size() .
- Active Edge Table (AET)

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Wir haben:

- kleinste und größte Bildzeile (y_{min} und y_{max}).
- Punkteliste (Ecken[i]). $i = 0, ...,$ecken.size() .
- Active Edge Table (AET)

Für jede Pixelzeile des Polygons, verwalte eine Liste an Kanten, die bei dieser y-Koordinate beginnen. So können diese Kanten leicht int die AET eingefügt werden.

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Für jede Pixelzeile des Polygons, verwalte eine Liste an Kanten, die bei dieser y-Koordinate beginnen. So können diese Kanten leicht int die AET eingefügt werden.
Dazu sind die einzelnen Listen in AE horizontal sortiert

```
    void drawPatternPolygon( Drawing& pic, const vector<IPoint2D>& ecken,
                             const QImage& muster ){
  ...
    KantenTabelle aet(0);

    vector<KantenTabelle> et(ymax + 1); // et(0, ..., ymin) bleiben leer

    KantenTabelle::iterator kitaet;  // Kanteniterator für AET
    KantenTabelle::iterator kitety;  // Kanteniterator für ET, Zeile y
    Kante k;
    int kymin;
    double xanf, xend;
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Wir haben:

- kleinste und größte Bildzeile (y_{min} und y_{max}).
- Punkteliste (Ecken[i]). $i = 0, ...,$ecken.size() .
- Active Edge Table (AET).
- Edge Table (ET).
- iterators für AET und ET.

AET und ET Sind noch lehr!!!

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Erzeuge die Kanten aus der Punkteliste. Sortiere die beiden Punkte einer Kante dabei nach ihrer y-Koordinate.

Die aktuelle x-Koordinate ist zu Beginn dann die x-Koordinate **des untere Punktes.**

Füge die Kante in die zu ihrer y-Koordinate gehörenden Liste **ein**.

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
for (unsigned int i = 0; i < n - 1; ++i){
  if (ecken[i].y != ecken[i + 1].y){
      if (ecken[i].y < ecken[i + 1].y){
          k.ymax = ecken[i + 1].y;
          k.x = ecken[i].x;
          kymin = ecken[i].y;
        }
      else{
          k.ymax = ecken[i].y;
          k.x = ecken[i + 1].x;
          kymin = ecken[i + 1].y;
        }
      k.einsdurchm = static_cast<double>(ecken[i + 1].x - ecken[i].x)
        (ecken[i + 1].y - ecken[i].y);
      et[kymin].push_back(k);
    }
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Das Polygon wird geschlossen, indem der letzte und der erste Punkt verbunden werden. **wie?**

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Das Polygon wird geschlossen, indem der letzte und der erste Punkt verbunden werden. **wie?**
Verwenden:
$ecken[n-1].y$ und $ecken[0].y$... C++?

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Das Polygon wird geschlossen, indem der letzte und der erste Punkt verbunden werden. **wie?**

```
if (ecken[n - 1].y != ecken[0].y){
   if (ecken[n - 1].y < ecken[0].y){
       k.ymax = ecken[0].y;
       k.x = ecken[n - 1].x;
       kymin = ecken[n - 1].y;
     }
   else{
       k.ymax = ecken[n - 1].y;
       k.x = ecken[0].x;
       kymin = ecken[0].y;
     }
   k.einsdurchm = static_cast<double>(ecken[0].x - ecken[n - 1].x) /
     (ecken[0].y - ecken[n - 1].y);
   et[kymin].push_back(k);
 }
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Wir haben alle Kanten Erzeug, aber die sind nicht sortiert.

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Sortiere die Kanten in et[y] horizontal.
So lassen sich die Kanten einfach sortiert in die AET einfügen.
Wie sortiert man eine Liste von Kanten?

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Sortiere die Kanten in et[y] horizontal.
So lassen sich die Kanten einfach sortiert in die AET einfügen.

```
for (int y = ymin; y <= ymax; ++y)
     et[y].sort();
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Wir haben:

- kleinste und größte Bildzeile (y_{min} und y_{max}).
- Punkteliste (vector<IPoint2D> ecken)
- Active Edge Table (AET). (KantenTabelle aet(0);) (list<Kante>)
- **Sortierte** Edge Table (ET). (vector<KantenTabelle> et(ymax + 1))
- iterators für AET und ET.

AET ist noch lehr!!!

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Vorbereitung ist Fertig.
Wir laufen über die y-Zeilen des Polygons.

```
for (int y = ymin; y <= ymax; ++y){
//etwas
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Vorbereitung ist Fertig.
Wir laufen über die y-Zeilen des Polygons.

```
for (int y = ymin; y <= ymax; ++y){
/* 1. Füge alle in Zeile y
      startenden Kanten sortiert ein*/

//etwas
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Vorbereitung ist Fertig.
Wir laufen über die y-Zeilen des Polygons.

```
for (int y = ymin; y <= ymax; ++y){
/* 1. Füge alle in Zeile y
      startenden Kanten sortiert ein*/

     kitety = et[y].begin();
     kitaet = aet.begin();
     while (kitety != et[y].end() && kitaet != aet.end()){
         if (*kitaet < *kitety)
             ++kitaet;
         else{
             aet.insert(kitaet, *kitety);
             ++kitety;
         }
     }
//etwas
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

Am Ende sind u.U. noch Kanten in et[y] übrig. Aufgrund der Sortierung können sie hier einfach angehängt werden. (e.g. y_{min})

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
for (int y = ymin; y <= ymax; ++y){
/* 1. Füge alle in Zeile y
      startenden Kanten sortiert ein*/

     kitety = et[y].begin();
     kitaet = aet.begin();
     while (kitety != et[y].end() && kitaet != aet.end()){
         if (*kitaet < *kitety)
             ++kitaet;
         else{
             aet.insert(kitaet, *kitety);
             ++kitety;
         }
     }

     while (kitety != et[y].end())
         aet.push_back(*kitety++);
//etwas
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
for (int y = ymin; y <= ymax; ++y){
      /* 1. Füge alle in Zeile y
       startenden Kanten sortiert ein*/ // Done!

      /* 2. Entferne alle bei y endenden Kanten aus der AET.
      Das Entfernen ist notwendig, um die Spans zwischen Paaren
      von Kanten der AET malen zu können.*/

//etwas
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
/* 2. Entferne alle bei y endenden Kanten aus der AET.
Das Entfernen ist notwendig, um die Spans zwischen Paaren
von Kanten der AET malen zu können.*/
```

zum Bespiel?

```
/* 2. Entferne alle bei y endenden Kanten aus der AET.
Das Entfernen ist notwendig, um die Spans zwischen Paaren
von Kanten der AET malen zu können.*/
```

zum Bespiel? $y + 2$.
Dieser Fall tritt immer dann ein, wenn am Ende einer Kante beim aktuellen y eine neue Kante beginnt.

C++: Wir laufen noch mal mit kitaet und vergleichen y und y_{max}.

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
for (int y = ymin; y <= ymax; ++y){
    /* 1. Füge alle in Zeile y
     startenden Kanten sortiert ein*/ // Done!

    /* 2. Entferne alle bei y endenden Kanten aus der AET. */

    for (kitaet = aet.begin(); kitaet != aet.end(); )
        if (kitaet->ymax == y)
            kitaet = aet.erase(kitaet);
        else
            ++kitaet;
    //etwas. Was?
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
for (int y = ymin; y <= ymax; ++y){
    /* 1. Füge alle in Zeile y
     startenden Kanten sortiert ein*/ // Done!

    /* 2. Entferne alle bei y endenden Kanten aus der AET. */ // Done!


    /* 3. Endlich Mahlen!*/
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
for (int y = ymin; y <= ymax; ++y){
    /* 1. Füge alle in Zeile y
     startenden Kanten sortiert ein*/ // Done!

    /* 2. Entferne alle bei y endenden Kanten aus der AET. */ // Done!

    /* 3. Endlich Mahlen!*/
    for (kitaet = aet.begin(); kitaet != aet.end(); ){
        xanf = (kitaet++)->x;
        xend = (kitaet++)->x;
        for (int x = static_cast<int>(round(xanf));
            x <=  static_cast<int>(round(xend)); ++x)
            pic.drawPoint(x, y, colour, false);
    }
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
for (int y = ymin; y <= ymax; ++y){
    /* 1. Füge alle in Zeile y
     startenden Kanten sortiert ein*/ // Done!

    /* 2. Entferne alle bei y endenden Kanten aus der AET. */ // Done!

    /* 3. Endlich Mahlen!*/ // Done!

   // noch etwas?
}
```

Aufgabe 07 Gefüllte Polygone

Rahmen Program: drawFilledPolygon

```
for (int y = ymin; y <= ymax; ++y){
    /* 1. Füge alle in Zeile y
     startenden Kanten sortiert ein*/ // Done!

    /* 2. Entferne alle bei y endenden Kanten aus der AET. */ // Done!

    /* 3. Endlich Mahlen!*/ // Done!

    /* 4. Aktualisiere die x-Werte aller Kanten in AET. */
    for (kitaet = aet.begin(); kitaet != aet.end(); ++kitaet)
        kitaet->x += kitaet->einsdurchm;  //x = x + Δx = x + 1/m
}
```

Run!