Der implementierte Algorithmus:

- berechnen des Gradienten(calcEuklidGradient()) und hinzufügen in priority queue (buildQueue())
- Inhalte werden von dieser automatisch sortiert
- schreiben der Elemente in eindimensionalen vector (buildQueue())
- gehe mit forschleife alle elemente in vector durch (checkPixel())
  - prüfe der reihe nach alle 4 Nachbarn
    - prüfe ob pixel kleiner als threshold oder außerhalb des Bildes → Abbruch
    - prüfe ob bereits in Superpixel: (checkInRegion())
      true → erweitern des Superpixels
      false → erstellen eines neuen Superpixels
    - füge alle Nachbarn die nicht in einem Superpixel liegen neuem Superpixel hinzu
  - gehe erstellte Superpixelliste durch und berechne die Durchschnittsfarbe (calcColor()) und färbe den Superpixel entsprechend ein (buildImage())
- säubere Speicher für späteren Durchlauf wenn Slider verstellt (cleanAll())

Bemerkungen:

- Sollten 2 benachbarte Pixel beide unterschiedlichen Regionen angehören, aber der Farbabstand beider unter dem Schwellwert liegen, so werden diese Regionen nicht verbunden, da entsprechendes nicht in der Aufgabenstellung erwähnt wurde.

Datenstrukturen:

- struct Gradient – hält Pixelkoordinaten und Gradienten
- struct Pixel - hält Pixelkoordinaten
- std::priority_queue<Gradient, vector<Gradient>, Order > myQueue
- struct Order – ändert Sortierreihenfolge der priority queue
- std::vector<std::vector<Pixel>*> regionsList – enthält Liste aller Superpixel mit deren Pixel
- cv::Mat inputImg, workImage, outputImage
- vector<Pixel>*** regionsUsed – enthält für Pixel in Superpixel Pointer zu Superpixel, sonst NULL
- vector<Gradient> sortedPixels – enthält Liste der sortierten Liste aus myQueue
- vector<Pixel> lonelyPixels – lokal, enthält alle Pixel die in keinem Superpixel sind
- vector<vector<Pixel>*> loclRegionList – lokal, enthält alle Pointer zu Superpixel der benachbarten Pixel

Das Verfahren:

Pro

- Bilder noch erkenntlich
- Farbreduzierung

contra

- Farbreduzierung nur begrenzt
- sehr viele Checks für jedes Pixel nötig ob es in einer Liste oder Array drin ist → Aufwand
- nicht ohne weiteres Informationen für ein Pixel sofort neuer Farbwert zuweisbar
  → kann damit nicht als Shader implementiert werden, da abhängig von sortierter Folge
  → Optimierung auf Grafikkarte nicht möglich
- sehr viele Superpixelobjekte am Anfang