Prof. Dr. Leif Kobbelt
Dr. Jan Möbius
Patrick Schmidt, Moritz Ibing

Visual Computing Institute | RWTH AACHEN UNIVERSITY

Basic Techniques in Computer Graphics

Assignment 11

Date Published: January 14th 2019, Date Due: January 21st 2019

- All assignments (programming and text) have to be completed in teams of 3–4 students. Teams with fewer than 3 or more than 4 students will receive no points.
- Hand in **one solution per team per assignment**.
- Every team must work independently. Teams with identical solutions will receive no points.
- Solutions are due 14:30 on January 21st 2019. Late submissions will receive zero points. No exceptions!
- Instructions for **programming assignments**:
  - Download the solution template (a zip archive) through the Moodle course room.
  - Complete the solution.
  - Prepare a new zip archive containing your solution. It must contain exactly those files that you changed. **Only change those files you are explicitly asked to change in the task description.** The directory layout must be the same as in the archive you downloaded.
  - Upload your zip archive through Moodle before the deadline. Use the Moodle group submission feature. Only in the first week (when Moodle groups have not been created yet), list all members of your group in the file `assignmentXX/MEMBERS.txt`. Remember, only one submission per group.
  - Your solution must compile and run correctly **on our lab computers** using the exact same `Makefile` provided to you. Do not include additional libraries and do not change code outside of the specified sections. If it does not compile on our machines, you will receive no points.
- Instructions for **text assignments**:
  - Prepare your solution as a single pdf file per group. Submissions on paper will not be accepted.
  - If you write your solution by hand, write neatly! Anything we cannot decipher will receive zero points. No exceptions!
  - Add the names and student ID numbers of all team members to every pdf.
  - Unless explicitly asked otherwise, always justify your answer.
  - Be concise!
  - Submit your solution via Moodle, together with your coding submission.

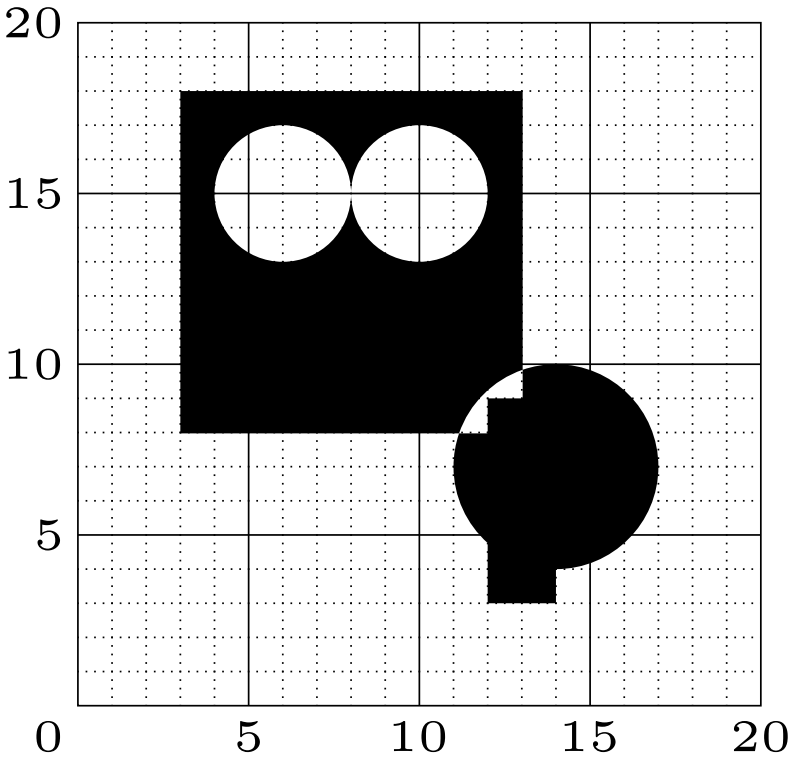
Prof. Dr. Leif Kobbelt
Dr. Jan Möbius
Patrick Schmidt, Moritz Ibing

Visual Computing Institute | RWTH AACHEN UNIVERSITY

Exercise 1 Constructive Solid Geometry [10 Points]

Using Constructive Solid Geometry (CSG) complex solids can be obtained by performing Boolean operations (union, intersection, difference) on some simpler primitives.

The black object depicted below, for instance, can be constructed out of 5 or less primitives (i.e. rectangles and circles).



(a) [3 Points]

Give an abstract specification (e.g. "a rectangle of size 4×3 with its lower left corner at $(4, 2)^T$") of the 5 or less shapes required to produce the black object depicted above.

(b) [3 Points]

For each shape described in the previous subtask, define an implicit function $p_i : \mathbb{R}^2 \to \mathbb{R}, \quad 1 \leq i \leq 5$ such that its value is negative on the inside and positive on the outside of the respective shape.

(c) [4 Point]

Construct an implicit function $p : \mathbb{R}^2 \to \mathbb{R}$ that combines the functions p_i above to describe the black shape depicted above. Again, this function should be negative for points inside the shape and positive for points outside the shape.

Note for all three subtasks: You may only use the operators $+$, $-$, $\cdot$, $/$, max(,), min(,), $|\ |$, $\sqrt{\ }$ in the implicit functions you define.

Prof. Dr. Leif Kobbelt
Dr. Jan Möbius
Patrick Schmidt, Moritz Ibing

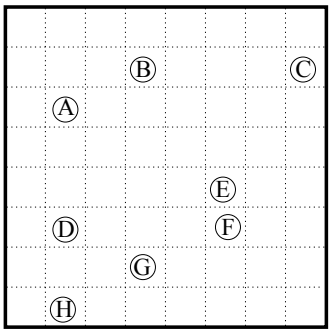
Visual Computing Institute | RWTH AACHEN UNIVERSITY

Exercise 2 Spatial Data Structures [14 Points]

To speed up many algorithms the scene is often partitioned using spatial data structures. In this task we will regard the ones most commonly used.

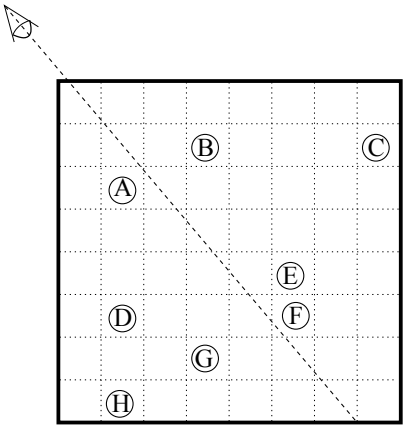
(a) Quadtree [4 Point]

Construct a Quadtree for the shown scene. Stop subdivision as soon as every node contains at most 1 object. Draw the splitting lines into the scene and the resulting tree structure next to it.



(b) kD-tree [4 Point]

Construct a kD-tree for the shown scene. Stop subdivision as soon as every node contains at most 1 object. Start with a split parallel to the x-axis. Draw the splitting lines into the scene and the resulting tree structure next to it.



(c) Painter's Algorithm [3 Point]

Use the kD-tree constructed in the last exercise to determine the order in which the objects are drawn by the Painter's Algorithm, given the drawn camera.

(d) BSP-tree [3 Point]

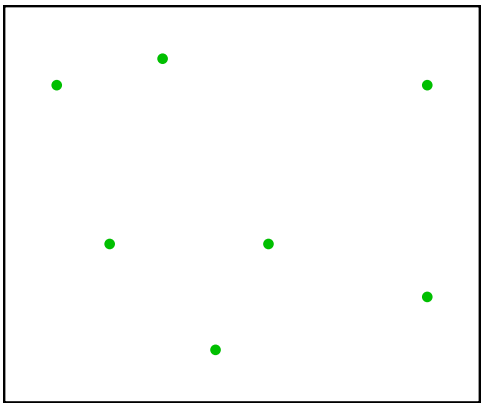
What differences are there between BSP-trees and kD-trees?
What benefits and disadvantages do BSP-trees have compared to kD-trees?

Prof. Dr. Leif Kobbelt
Dr. Jan Möbius
Patrick Schmidt, Moritz Ibing

Visual Computing Institute | RWTH AACHEN UNIVERSITY

Exercise 3 Voronoi Diagram and Delaunay Triangulation [8 Points]

Consider the following set of 2D points:



(a) [4 Point]

Draw the Voronoi Diagram of the given point set!

(b) [4 Point]

Draw the Delaunay Triangulation of the given point set!
(You can draw your solution for tasks (a) and (b) in the same sketch, if you want.)

Exercise 4 Dual of Voronoi Diagrams [8 Points]

In most cases, dualizing the Voronoi Diagram of a set of 2D points immediately yields a Delaunay Triangulation of the given points. However, there are exceptional cases where the dual is not a triangle mesh: For example, four points in a square configuration have a Voronoi Diagram that dualizes to a quadrilateral instead of two triangles.

(a) [4 Point]

Similarly, it is possible to construct a point set such that the dual of its Voronoi Diagram is an n-sided polygon (for any $n \geq 3$).
Give instructions how to construct such a point set!
For $n = 5$, draw an example sketch of the input point set, its Voronoi Diagram and the resulting dual mesh.

(b) [4 Point]

You are given a point set constructed accordingly to task (a) and the resulting n-sided polygon (with $n > 3$). The resulting polygon is now triangulated by arbitrarily picking one vertex v and connecting it to all other vertices that aren't connected to v yet.
In general, is the resulting triangle mesh a Delaunay Triangulation? Explain why or why not!