Aufgabe 1

Um den Median in linearer Zeit zu bestimmen, kann der Median of Medians Algorithmus verwendet werden. Im Folgenden ist der Algorithmus in Form von Pseudocode näher erläutert.

function SELECT(L, k)
  if *L enthält* 10 *oder weniger Elemente* **then**
    sort(L)
    return Element an $k$ter Stelle
  end if

  Teile L in Subsets $S[i]$ mit jeweils 5 Elementen auf ▷ Es gibt n/5 Subsets

  for $i = 1 \rightarrow n/5$ **do**
    X *add* $select(S[i], 3)$ ▷ Median von den Subsets
  end for

  $M = select(X[i], n/10)$

  Teile L auf in L1 $<$ M, L2 $=$ M, L3 $>$ M
  if $k \leq |L1|$ **then**
    return $select(L1, k)$
  else if $k > |L1| + |L2|$ **then**
    $select(L3, k - |L1| - |L2|)$
  else
    return M
  end if
end function

Aussage: Der o.g. Algorithmus liefert den Median in $O(n)$.

BEWEIS: Wir werfen entweder $L3$ (Werte größer als M) oder $L1$ (Werte kleiner als M) weg. Angenommen wir werfen $L3$ weg. Unter den $n/5$ Werten von X befinden sich $n/10$ größere Werte als M. Wenn ein Wert aus X größer als M ist, dann sind genau 2 Werte aus $S[i]$ ebenfalls größer als der Wert aus X. Daraus folgt, dass $L3$ mindestens 3 Elemente in jeder der $n/10$ Gruppen von $S[i]$ hat, woraus sich mindestens $3n/10$ Elemente ergeben. Das gleiche gilt für $L1$, wodurch sich höchstens $7n/10$ Elemente ergeben ($T(7n/10)$).

Da 20% der Elemente der Liste Mediane sind folgt:

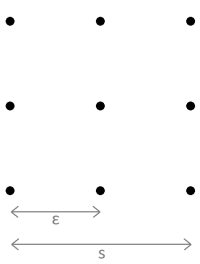
$T(n) \leq T(n/5) + T(7n/10) + O(n)$

Die Laufzeit $O(n)$ kommt durch das Partitionieren der einzelnen Listen zu Stande und die Laufzeit zur Auswahl der einzelnen Mediane aus diesen Listen beträgt wiederum $O(1)$, woraus wiederum folgt:

$T(n) \leq c * n * (1 + (9/10) + (9/10)^2 + ...) = O(n) \quad \square$

Aufgabe 2

Eine Zelle der Kantenlänge s kann bei einem Punkt-Mindestabstand ϵ maximal $n_{max} = (\frac{s}{\epsilon} + 1)^2$ Punkte enthalten, da am meisten Punkte in die Zelle passen, wenn die Punkte Gitterförmig angeordnet sind:



Ein Octree hat bei einer Tiefe t maximal $n = 8^t$ Blätter, bzw. Punkte. Nach t umgestellt ergibt sich für die maximale Tiefe bei n_{max} Punkten:

$$t_{max} = log_8(n_{max}) = 2 \cdot log_8(\tfrac{s}{\epsilon} + 1)$$

Aufgabe 3

Ein weiteres volumetrisches Primitiv wäre ein Prisma mit einem gleichseitigen Dreieck als Grundfläche (Dreiecke können in 2D auf einfache Weise selbstähnlich angeordnet werden). Eine Unterteilungsstrategie könnte das Ausgangsprimitiv folgendermaßen in acht Unterprimitive teilen:

Die Spitze des Ausgangsprismas zeige in negative Y-Richtung. Alle Unterprimitive werden zuerst (d.h. als letzte Transformationsmatrix) um 0.5 skaliert.
Folgende Translationen werden für die ersten vier Primitiven durchgeführt (h sei die Höhe des Ausgangsprimitivs):

$$\begin{pmatrix} 0 \\ -\frac{h}{4} \\ 0 \end{pmatrix}, \begin{pmatrix} -\frac{h}{4} \\ \frac{h}{4} \\ 0 \end{pmatrix}, \begin{pmatrix} \frac{h}{4} \\ \frac{h}{4} \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ \frac{h}{4} \\ 0 \end{pmatrix}$$

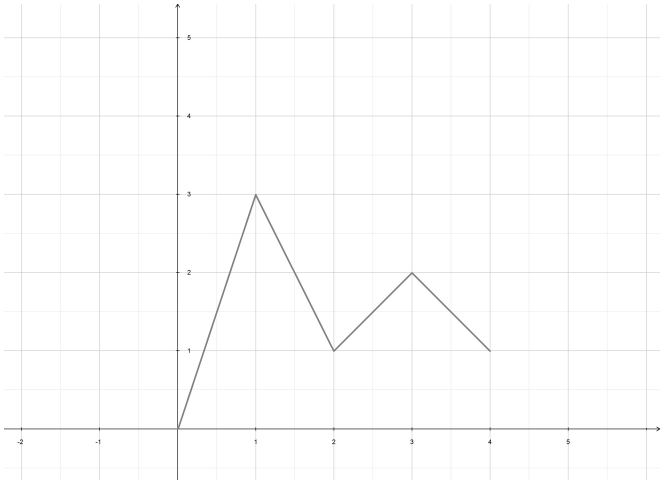
Das Prisma mit der letztgenannten Translation wird vorher um 180° um die Y-Achse gedreht.

Folgende Translationen werden für die letzten vier Primitiven durchgeführt (h sei die Höhe des Ausgangsprimitivs, l die Länge in Z-Richtung):

$$\begin{pmatrix} 0 \\ -\frac{h}{4} \\ \frac{l}{2} \end{pmatrix}, \begin{pmatrix} -\frac{h}{4} \\ \frac{h}{4} \\ \frac{l}{2} \end{pmatrix}, \begin{pmatrix} \frac{h}{4} \\ \frac{h}{4} \\ \frac{l}{2} \end{pmatrix}, \begin{pmatrix} 0 \\ \frac{h}{4} \\ \frac{l}{2} \end{pmatrix}$$

Auch hier wird das Prisma mit der letztgenannten Translation vorher um 180° um die Y-Achse gedreht.

Aufgabe 4



Die Kontrollpunkte werden linear interpoliert.
Ein B-Spline ist nicht stetig, wenn mehr als n Knots aufeinander liegen.

Aufgabe 5

Seien:
n die Anzahl der Kontrollpunkte
p_i mit $i \in \{0, 1, 2...n\}$ die Kontrollpunkte

Eine affine Transformation lässt sich in einen multiplikativen Teil (Matrix M) und einen additiven Teil (Vektor $\vec{a}$) aufteilen:

$$T(\vec{v}) = M \cdot \vec{v} + \vec{a}$$

Affine Invarianz: Es ist egal, ob zuerst T auf die Kontrollpunkte angewandt wird mit anschließender Interpolation oder umgekehrt:

$$T(\sum_{i=0}^{n} p_i \cdot L_i^n(u)) = \sum_{i=0}^{n} (T(p_i \cdot L_i^n(u)))$$

Setze gleich:

$$\begin{aligned}
T(\sum_{i=0}^{n} p_i \cdot L_i^n(u)) &= \sum_{i=0}^{n} (T(p_i \cdot L_i^n(u))) \\
M(\sum_{i=0}^{n} p_i \cdot L_i^n(u)) + \vec{a} &= \sum_{i=0}^{n} (M(p_i \cdot L_i^n(u)) + \vec{a}) \\
\sum_{i=0}^{n} M(p_i \cdot L_i^n(u)) + \vec{a} &= \sum_{i=0}^{n} M(p_i \cdot L_i^n(u)) + \sum_{i=0}^{n} \vec{a} \cdot L_i^n(u) \\
\vec{a} &= \vec{a} \cdot \sum_{i=0}^{n} L_i^n(u) \qquad \Leftrightarrow \sum_{i=0}^{n} L_i^n(u) = 1 \quad \square
\end{aligned}$$

Demnach sind beide Seiten gleich, genau dann, wenn für die Basisfunktion gilt:

$$L_i^n(u) = 1 \ \forall u \ //\text{Partition der Eins}$$