Prof. Dr. Günther Greiner
Lehrstuhl für Graphische Datenverarbeitung
der Universität Erlangen–Nürnberg

5. August 2011

Algorithmik kontinuierlicher Systeme — 5. August 2011

Angaben zur Person (Bitte in DRUCKSCHRIFT ausfüllen!):

Name, Vorname: .....

Geburtsdatum: .....

Matrikelnummer: .....

Studienfach: .....

Nicht von der Kandidatin bzw. vom Kandidaten auszufüllen !

Bewertung:

Aufgabe	1	2	3	4	5	6	7	8	9	10
Max. Punktzahl	4	10	7	8	10	10	12	10	9	10
Erreichte Punkte										

Gesamtpunktzahl	
Note	

Organisatorische Hinweise

Die folgenden Hinweise bitte aufmerksam lesen und die Kenntnisnahme durch Unterschrift bestätigen!

- Bitte legen Sie Ihren Studentenausweis und einen Lichtbildausweis zur Personenkontrolle bereit.
- Hilfsmittel (außer Schreibmaterial **und** Taschenrechner) sind nicht zugelassen. Andere elektronische Geräte sind auszuschalten.
- Fragen zu den Prüfungsaufgaben werden grundsätzlich nicht beantwortet.
- Die Lösung einer Aufgabe muss auf das jeweilige Aufgabenblatt geschrieben werden. Sollte der Platz nicht reichen, so verwenden Sie die Zusatz-Seiten am Ende der Klausur. Fügen Sie einen Hinweis in Ihre Lösung ein, dass die Lösung auf den Zusatz-Seiten fortgesetzt wurde und beschriften Sie diese mit Namen und Aufgabennummer.
- Es können durch die Aufsicht zusätzlich Seiten eingeheftet werden, sollte mehr Platz benötigt werden. Bitte beschriften Sie den Kopf dieser Seiten mit Ihrem Namen und der Aufgabennummer. Streichen Sie alles, was nicht bewertet werden soll doppelt aus.
- Auf Ihrem Platz befinden sich einige lose Blätter Schmierpapier. Bei Bedarf können Sie zusätzliches Schmierpapier von der Aufsicht anfordern. Das Schmierpapier muss abgegeben werden, es wird aber nicht bewertet.
- Wenn Sie die Prüfung aus gesundheitlichen Gründen abbrechen müssen, so muss Ihre Prüfungsunfähigkeit durch eine Untersuchung in der Universitätsklinik nachgewiesen werden. Melden Sie sich bei der Aufsicht und lassen Sie sich das entsprechende Formular aushändigen.
- Die Bearbeitungszeit beträgt 90 Minuten, es sind alle zehn Aufgaben mit 90 Punkten zu bearbeiten.
- Überprüfen Sie die Prüfungsaufgaben auf Vollständigkeit (30 Seiten inklusive Deckblatt) und einwandfreies Druckbild.
- Vergessen Sie nicht, auf dem Deckblatt die Angaben zur Person einzutragen und die Erklärungen auf dieser Seite zu unterschreiben.
- Viel Erfolg!

Erklärungen

Durch meine Unterschrift bestätige ich den Empfang der vollständigen Klausurunterlagen und die Kenntnisnahme der obigen Informationen.

Erlangen, 5. August 2011 .....
(Unterschrift)

Ich bin damit einverstanden, dass mein Prüfungsergebnis unter Angabe der Matrikelnummer anonymisiert veröffentlicht wird:

ja: ☐ nein: ☐

Erlangen, 5. August 2011 .....
(Unterschrift)

1 Komplexität (4 Punkte)

Bestimmen Sie die Komplexität bzw. die Fehlerordnung der folgenden Operationen. Dabei sollen Sie annehmen, dass die Länge der Spaltenvektoren n und die Größe von Matrizen $n \times n$ ist.

Lösen von $\mathbf{A}\vec{x} = \vec{b}$, für untere Dreiecksmatrizen $\mathbf{A}$	$\mathbf{O}$ ( )
Auswerten eines Polynoms vom Grad n mittels Hornerschema	$\mathbf{O}$ ( )
Bestimmung der QR-Zerlegung mittels Householder-Spiegelungen	$\mathbf{O}$ ( )
Bestimmung der Determinante von $\mathbf{A}$ bei gegebener LU-Zerlegung $\mathbf{A} = \mathbf{LU}$	$\mathbf{O}$ ( )
Bestimmung eines Interpolanten vom Grad n in der Newton-Basis	$\mathbf{O}$ ( )
Berechnung des Winkels zwischen zwei Vektoren	$\mathbf{O}$ ( )
Fehlerordnung bei Integration mittels iterierter Trapezregel (Länge der Teilintervalle ist h)	$\mathbf{O}$ ( )
Fehlerordnung bei Integration mittels iterierter Simpsonregel (Länge der Teilintervalle ist h)	$\mathbf{O}$ ( )

2 Speicherung dünn besetzter Matrizen (10 Punkte)

Gegeben ist die Matrix $\mathbf{M}$:

$$\mathbf{M} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 7 & 8 & 0 \\ 0 & 2 & 3 & 0 & 0 \\ 0 & 3 & 2 & 1 & 0 \\ 0 & -4 & 0 & -3 & 0 \end{bmatrix}.$$

a) Sie sollen $\mathbf{M}$ im `CCS`-Format abspeichern (Compressed Column Storage). Geben Sie hierzu die interne Datenstruktur an. Die Indizierung beginnt stets bei 0.

b) Die Matrix $\mathbf{M}$ soll nun transponiert werden. Wie können Sie $\mathbf{M}^T$ speichern, damit die Transformation kostengünstig durchgeführt werden kann? Inwiefern muss die interne Datenstruktur für das neue Matrixformat abgeändert werden?

c) Gegeben ist nun eine Matrix der Breite 4 im `CRS`-Format (Compressed Row Storage):

Wertearray: `4 3 2 2 1 4`
Spaltenindexarray: `1 2 3 1 2 3`
Zeilenpointerarray: `0 1 1 3 6 6`

Geben Sie die Matrix an.

3 LU-Zerlegung (7 Punkte)

Gegeben sei die Matrix $\mathbf{A}$ mit

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 4 \\ 4 & 3 & 9 \\ -4 & 1 & -3 \end{bmatrix}.$$

a) Bestimmen Sie die $\mathbf{LU}$-Zerlegung (ohne Pivotisierung) von $\mathbf{A}$.

Gegeben sei die Matrix $\mathbf{B}$ sowie deren $\mathbf{LU}$-Zerlegung und ein Vektor $\vec{b}$:

$$\mathbf{B} = \begin{bmatrix} 1 & 2 & 0 & 1 \\ -1 & -1 & 2 & -1 \\ 2 & 4 & 2 & 2 \\ 1 & 3 & 6 & -1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 2 & 0 & 1 & 0 \\ 1 & 1 & 2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 & 0 & 1 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & -2 \end{bmatrix}; \qquad \vec{b} = \begin{bmatrix} 1 \\ 2 \\ -1 \\ 0 \end{bmatrix}.$$

b) Lösen Sie $\mathbf{B}\vec{x} = \vec{b}$ mit Hilfe der $\mathbf{LU}$-Zerlegung.

4 Least-Squares Approximation (8 Punkte)

Gegeben seien die folgenden 2D-Datenpunkte $\vec{p}_i = \left[x_i, y_i\right]^T$:

$$\vec{p}_0 = \begin{bmatrix} -1 \\ 1 \end{bmatrix},\ \vec{p}_1 = \begin{bmatrix} 0 \\ -2 \end{bmatrix},\ \vec{p}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix},\ \vec{p}_3 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}.$$

a) Bestimmen Sie das lineare überbestimmte Gleichungssystem $\mathbf{A}\vec{u} = \vec{b}$, welches die Bedingungen $f(x_i) = y_i$ für eine quadratische Funktion $f(x) = ax^2 + bx + c$ repräsentiert. Geben Sie dazu $\mathbf{A}$, $\vec{u}$ und $\vec{b}$ explizit an.

b) Gegeben sei das folgende überbestimmte Gleichungssystem:

$$\begin{bmatrix} -2 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 0 & 2 \\ 3 & 2 & 0 \end{bmatrix} \begin{bmatrix} c \\ b \\ a \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 0 \\ 4 \end{bmatrix}.$$

Bestimmen Sie nun dasjenige lineare Gleichungssystem, dessen Lösung das vorgegebene System im Least-Squares Sinn löst. Geben Sie explizit die berechnete Systemmatrix, sowie die berechnete rechte Seite des Gleichungssystems an.

c) Neben der Gaußschen Normalengleichung gibt es noch andere Verfahren um überbestimmte lineare Gleichungssysteme im Least-Squares Sinn zu lösen. Nennen Sie ein weiteres Verfahren.

5 Hauptkomponentenanalyse (10 Punkte)

Gegeben seien die folgenden 2D-Datenpunkte $\vec{p}_i = \left[x_i, y_i\right]^T$:

$$\vec{p}_0 = \begin{bmatrix}-2\\-1\end{bmatrix},\ \vec{p}_1 = \begin{bmatrix}-1\\-2\end{bmatrix},\ \vec{p}_2 = \begin{bmatrix}0\\-1\end{bmatrix},\ \vec{p}_3 = \begin{bmatrix}2\\1\end{bmatrix},\ \vec{p}_4 = \begin{bmatrix}3\\0\end{bmatrix},\ \vec{p}_5 = \begin{bmatrix}4\\3\end{bmatrix}.$$

a) Bestimmen Sie die Covarianzmatrix $\mathbf{C}$ die zu den $\vec{p}_i$ gehört.

b) Gegeben sei die folgende Covarianzmatrix:

$$\mathbf{B} = \begin{bmatrix} 4 & -3 \\ -3 & 4 \end{bmatrix}.$$

Bestimmen Sie nun die zu **B** gehörenden Hauptachsen.

c) Gegeben sei die Eigenwertzerlgung $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ der $n \times n$ Matrix $\mathbf{A}$. Dabei sind die Spaltenvektoren von $\mathbf{V}$ die Eigenvektoren von $\mathbf{A}$ und $\mathbf{\Lambda}$ ist die Diagonalmatrix der Eigenwerte λ_i von $\mathbf{A}$ ($\mathbf{\Lambda} = diag(\lambda_1, ..., \lambda_n)$). Zusätzlich sei ein Startwert $x_0 = \sum_{i=1}^{n} \alpha_i \vec{v}_i$ durch seine Koeffizienten α_i in der Eigenbasis (aufgespannt durch die Eigenvektoren $\vec{v}_i$) gegeben. Zeigen Sie, dass $\mathbf{A}^k x_0 = \sum_{i=1}^{n} \alpha_i \lambda_i^k \vec{v}_i$.

6 Programmierung: Polynominterpolation (10 Punkte)

Die Klassen `MonomialCurve` und `NewtonCurve` sind Spezialisierungen von `PolyCurve`. Dabei repräsentieren sie Polynominterpolanten bezüglich Monom- bzw. Newtonbasis. Sie sollen dabei die entsprechenden Methoden implementieren (verwenden Sie `C++` Syntax). Entgegen der Übungen ist **keine** Fehlerbehandlung erforderlich.

```
class Point2D {
public:
    float x, y;    //! Komponenten des Punktes
};

class PolyCurve {
public:
    //! Konstruktor: Legt die Kurve an und setzt die Kontrollpunkte
    PolyCurve(std::vector<Point2D> controlPoints);

    //! Destruktor
    virtual ~PolyCurve();

    //! Gibt die Kontrollpunkte zurueck
    std::vector<Point2D>& getControlPoints();

    //! Gibt die Anzahl der Kontrollpunkte zurueck
    unsigned int getNumControlPoints() const;
    ...
protected:
    ...
};

class Matrix {
public:
    //! Konstruktor: Baut eine unitialisierte Matrix
    Matrix(unsigned int height, unsigned int width);

    //! Destruktor
    ~Matrix();

    void setZero();                                          //! Setzt alle Matrixwerte auf 0
    float& operator()(unsigned int m, unsigned int n);       //! Mutator (m=Zeile, n=Spalte)
    float  operator()(unsigned int m, unsigned int n) const; //! Akzessor (m=Zeile, n=Spalte)
    ...
private:
    ...
};
```

Bitte wenden!

a) Implementieren Sie die Methode `Matrix MonomialCurve::getCoefficientMatrix()`. Dazu muss die Vandermonde-Matrix gebaut und zurückgegeben werden.

```
Matrix MonomialCurve::getCoefficientMatrix() const {




}
```

b) Werten Sie nun das Polynom in der Monombasis an der Stelle `x` mittels Hornerschema aus. Die Matrix `coeff` ($\in \mathbb{R}^{n\times 1}$) beschreibt die Koeffizienten des Polynoms.

```
float MonomialCurve::evaluateAt(const Matrix& coeff, const float x) const {




}
```

c) Implementieren Sie die Methode `Matrix NewtonCurve::getCoefficientMatrix()`. Dazu muss die Systemmatrix, welche die Interpolationsbedingungen bezüglich der Newtonbasis beschreibt, zurückgegeben werden.

```
Matrix NewtonCurve::getCoefficientMatrix() const {









}
```

d) Werten Sie nun das Polynom in der Newtonbasis an der Stelle `x` mittels erweitertem Hornerschema aus. Die Matrix `coeff` ($\in \mathbb{R}^{n\times 1}$) beschreibt die Koeffizienten des Polynoms.

```
float NewtonCurve::evaluateAt(const Matrix& coeff, const float x) const {









}
```

7 Programmierung: Bildbearbeitung (12 Punkte)

In dieser Aufgabe sollen verschiedene Operationen auf Graustufenbildern, repräsentiert durch die Klasse `Image`, implementiert werden (verwenden Sie `C++` Syntax). Die Bilder sind immer quadratisch und die zugehörigen Grauwerte sind als `float` gespeichert. Entgegen der Übungen ist **keine** Fehlerbehandlung erforderlich.

```
class Image {
public:
    //! Konstruktor: Baut ein uninitialisiertes, QUADRATISCHES Bild der Groesse dim x dim
    Image(unsigned int dim);

    //! Copy Konstruktor
    Image(const Image& other);

    //! Destruktor
    ~Image();

    //! Gibt die Anzahl der Zeilen/Spalten zurueck (das Bild ist quadratisch)
    unsigned int getDimension();

    float& operator()(unsigned int m, unsigned int n);       //! Mutator (m=Zeile, n=Spalte)
    float  operator()(unsigned int m, unsigned int n) const; //! Akzessor (m=Zeile, n=Spalte)
    Image& operator=(const Image& other);                    //! Zuweisungsoperator
    ...
private:
    ...
};
```

Bitte wenden!

a) Implementieren Sie folgende Funktion, welche für das übergebene Bild `const Image& image` die diskreten Ableitungen in x-Richtung (Breite) berechnet. Verwenden Sie dazu wenn möglich zentrale Differenzen und an Randpixeln entsprechend Vorwärts bzw. Rückwärtsdifferenzen. Sie können davon ausgehen, dass das Bild `const Image& result` schon die richtige Größe hat.

```
void computeDerivativeX(const Image& image, Image &result) {




}
```

b) Implementieren Sie folgende Funktion, welche für das übergebene Bild `const Image& image` den diskreten Laplace-Operator berechnet. Sie können davon ausgehen, dass das Bild `const Image& result` schon die richtige Größe hat. Verwenden Sie dazu die obige Funktion aus Teilaufgabe a), sowie die bereits gegebene Funktion `void computeDerivativeY(const Image& image, Image& result)`.

```
void computeLaplacian(const Image& image, Image& result) {




}
```

c) Implementieren Sie folgende Funktion, welche die partielle Differentialgleichung $\Delta I = 0$ auf dem Gebiet Ω mit Hilfe des Gauß-Seidel Verfahrens (**ohne** `SOR`) löst. Dabei ist I das übergebene Bild `Image& image`. Das Gebiet Ω ist durch die Maske `const Image& mask` gegeben. Diese hat die selbe Größe wie das übergebene Bild. Dabei sind Werte der Maske im Inneren von Ω gleich $1.0f$, und außerhalb $0.0f$. Sowohl die Randbedingungen (außerhalb von Ω) als auch die Startwerte (innerhalb von Ω) sind bereits in I entsprechend gesetzt. Führen Sie `unsigned int iterations` Gauß-Seidel Schritte durch. Das Ergebnis soll am Ende in `Image& image` stehen.

```
void solveLaplacianEquation(Image& image, const Image& mask, unsigned int iterations) {




}
```

8 Nichtlineare Optimierung (10 Punkte)

a) Betrachten Sie die Matrix

$$\mathbf{A} = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}$$

Bestimmen Sie zwei, zueinander **A**-orthogonale Richtungen der Länge 1.

b) Führen Sie für das quadratische Funktional

$$Q(\vec{x}) = \vec{x}^T \mathbf{A} \vec{x} + 2\vec{b}^T \vec{x} + \gamma \qquad (\mathbf{A} \text{ wie in a}),\ \vec{b} = \begin{pmatrix} 1 \\ -1 \end{pmatrix},\ \gamma = 2$$

einen Schritt des Gradientenverfahrens durch. Beginnen Sie im Punkt $\vec{x}_0 = \binom{1}{1}$ und verwenden Sie dabei die optimale Schrittweite:

$t_i := -\frac{(\mathbf{A}\vec{x}_i + \vec{b})^T \vec{s}_i}{\vec{s}_i{}^T \mathbf{A} \vec{s}_i}$ ($\vec{s}_i$ die Suchrichtung)

c) Betrachten Sie die nichtlineare Zielfunktion

$$F(x, y) = \frac{1}{4}\, x^4 + 3\, x^2 y^2 + xy - 3\, x + 2$$

Führen Sie einen Schritt das NEWTON-Verfahrens zum Auffinden des Minimums durch. Beginnen Sie im Punkt $\vec{x}_0 = \binom{0}{0}$.
Hinweis: Die Inverse einer 2×2*-Matrix:*

$$\mathbf{A}^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{\det(\mathbf{A})} \cdot \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

d) Bekanntlich liefert das NEWTON-Verfahren in der Regel eine quadratisch konvergente Folge. Erklären Sie den Begriff *die Folge* (x_i) *konvergiert quadratisch gegen* x^*

9 Interpolation (9 Punkte)

a) Die folgenden Daten sollen interpoliert werden:

i	0	1	2	3
x_i	-2	0	2	4
y_i	0	2.5	4	4.5

Bestimmen Sie den (stückweise) linearen Interpolanten $L : [-2, 4] \to \mathbb{R}$

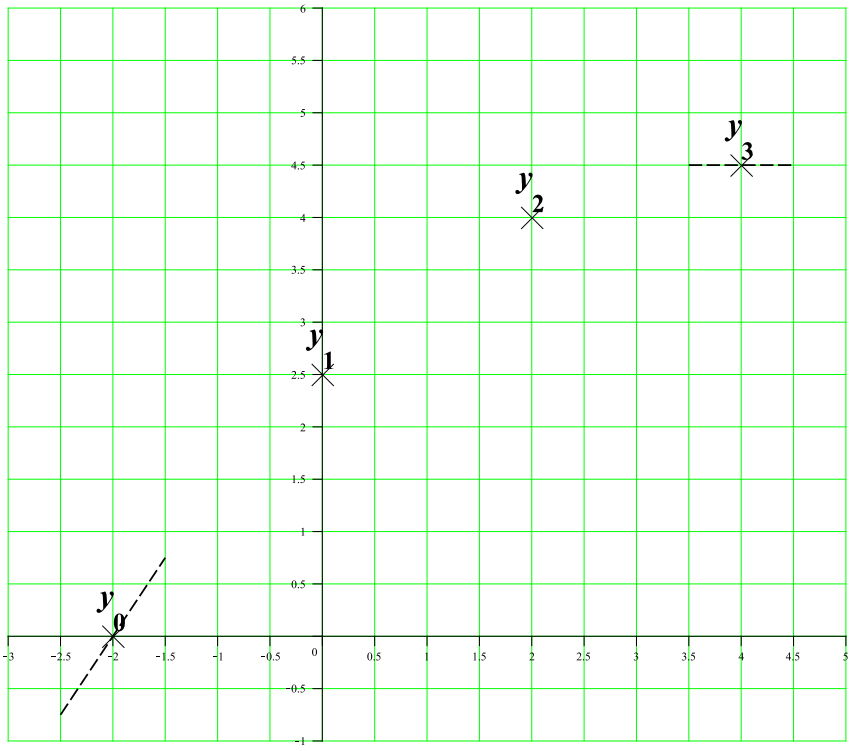
$$L(x) = \begin{cases} \\ \\ \\ \\ \\ \\ \\ \end{cases}$$

b) Skizzieren Sie den (Graphen des) CATMULL-ROM-Interpolanten $R : [-2, 4] \to \mathbb{R}$ zu diesen Daten, indem Sie zunächst die Steigungen (Ableitungen) y'_1 und y'_2 an den Stellen x_1 und x_2 mittels zentraler Differenzen ermitteln. Die Steigungen (Ableitungen) in den Randpunkten sind vorgegeben: $y'_0 = 1.5$, $y'_3 = 0$.

Die zu ermittelnden Ableitungen y'_1 und y'_2 müssen rechnerisch bestimmt werden!

$y'_1 =$

$y'_2 =$



c) Betrachten Sie das Dreieck $\Delta(RST)$ mit den Eckpunkten:

$$R = \begin{pmatrix} 0 \\ 2 \end{pmatrix} \ , \quad S = \begin{pmatrix} 3 \\ 0 \end{pmatrix} \ , \quad T = \begin{pmatrix} 3 \\ 2 \end{pmatrix} \ .$$

Bestimmen Sie die baryzentrischen Koordinaten bezüglich dieses Dreiecks (jeweils ρ, σ, τ) für die folgenden Punkte:

$$P_1 := \begin{pmatrix} 1 \\ 2 \end{pmatrix} :$$

$$P_2 := \begin{pmatrix} 3 \\ 2 \end{pmatrix} :$$

$$P_3 := \begin{pmatrix} 2 \\ 1 \end{pmatrix} :$$

d) Betrachten Sie das Prisma mit dreieckigen Deckflächen $\Delta(R_0\,S_0\,T_0)$ und $\Delta(R_1\,S_1\,T_1)$, wobei

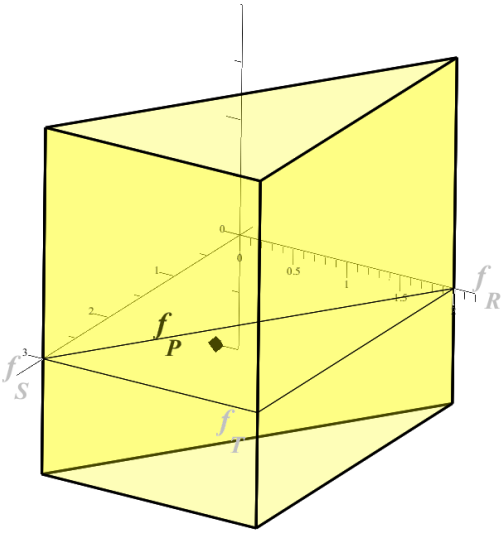
$$R_0 = \begin{pmatrix} 0 \\ 2 \\ -1 \end{pmatrix} , \quad S_0 = \begin{pmatrix} 3 \\ 0 \\ -1 \end{pmatrix} , \quad T_0 = \begin{pmatrix} 3 \\ 2 \\ -1 \end{pmatrix} , \quad R_1 = \begin{pmatrix} 0 \\ 2 \\ 2 \end{pmatrix} , \quad S_1 = \begin{pmatrix} 3 \\ 0 \\ 2 \end{pmatrix} , \quad T_1 = \begin{pmatrix} 3 \\ 2 \\ 2 \end{pmatrix} .$$

In den Eckpunkten sind Temperaturwerte bekannt:

$$f_{R_0} = 12 , \quad f_{S_0} = 48 , \quad f_{T_0} = 36 , \quad f_{R_1} = 12 , \quad f_{S_1} = 72 , \quad f_{T_1} = 72$$

Diese sollen ins Innere interpoliert werden.

Ermitteln Sie den interpolierten Wert im Punkt $P = (2, 1, 0)^T$ indem Sie zunächst entlang der senkrechten Kanten $\overline{R_0\,R_1}$, $\overline{S_0\,S_1}$ und $\overline{T_0\,T_1}$ linear interpolieren und dann die erhaltenen Werte f_R, f_S, f_T mit Hilfe baryzentrischer Koordinaten linear interpolieren.



10 Bézier-Kurven (10 Punkte)

a) Betrachten Sie die kubische BÉZIER-Kurve $C(t)$ mit den Kontrollpunkten

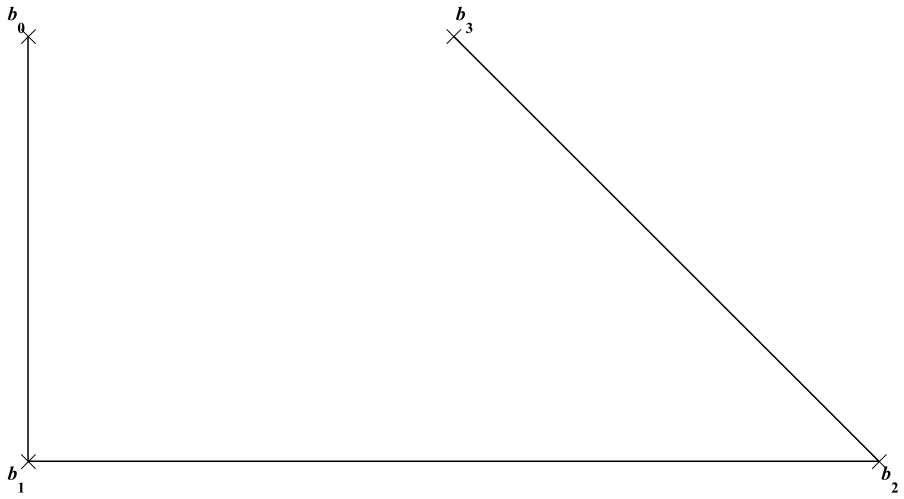
$$\vec{b}_0 = \begin{bmatrix} 0 \\ 32 \end{bmatrix}, \quad \vec{b}_1 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \quad \vec{b}_2 = \begin{bmatrix} 64 \\ 0 \end{bmatrix}, \quad \vec{b}_3 = \begin{bmatrix} 32 \\ 32 \end{bmatrix}.$$

Werten Sie diese Kurve mit Hilfe des DECASTELJAU-Algorithmus an der Stelle $t = \frac{1}{4}$ aus.

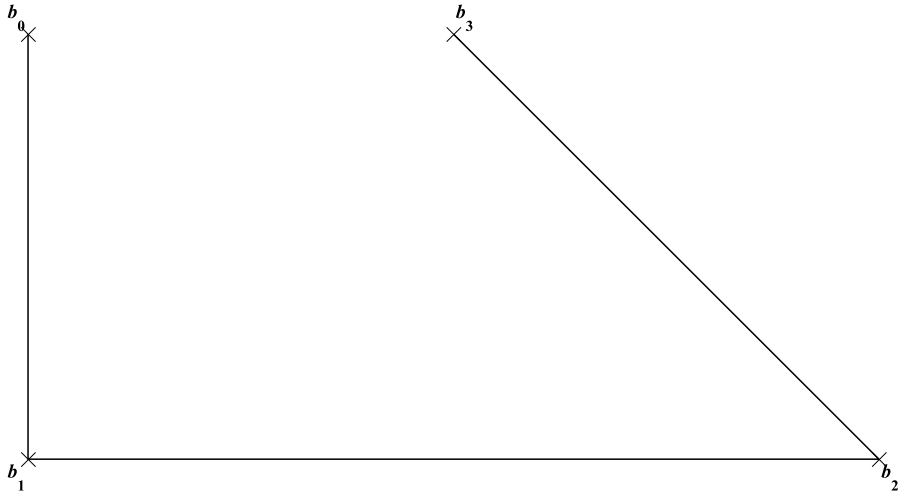
b) Führen Sie für die Kurve von Teilaufgabe a) einen *midpoint subdivision* Schritt durch und bestimmen Sie die Kontollpunkte $\vec{c}_0$, $\vec{c}_1$, $\vec{c}_2$ und $\vec{c}_3$ der linken Teilkurve sowie die Kontrollpunkte $\vec{d}_0$, $\vec{d}_1$, $\vec{d}_2$ und $\vec{d}_3$ der rechten Teilkurve.

c) Lösen Sie die Teilaufgaben a) und b) geometrisch. Ergänzen Sie dazu die nachfolgenden Abbildung und benennen Sie die Punkte $C(\frac{1}{4})$, $\vec{c}_0$, ... $\vec{c}_3$, $\vec{d}_0$, ..., $\vec{d}_3$.

DeCasteljau:



Midpoint subdivision:



d) Die BÉZIER-Kurve von Teilaufgabe a) soll zunächst um +4 Einheiten entlang der x-Achse verschoben werden und danach um 90 Grad im Uhrzeigersinn gedreht werden. Bestimmen Sie die Kontrollpunkte der resultierenden Kurve.
Hinweis: Die Drehung um 90 Grad wird durch die Abbildung $\binom{x}{y} \mapsto \binom{y}{-x}$ *beschrieben.*

e) Welche Eigenschaften der BERNSTEIN-Polynome $B_i^n(t)$ implizieren die Endpunktinterpolation von BÉZIER-Kurven $C(t) = \sum_{i=0}^{n} \vec{b}_i B_i^n(t)$
Hinweis: $B_i^n(t) = \binom{n}{i} (1-t)^{n-i} t^i$.