11. März 2003
Prüfungsprotokoll
Nebenfach Mathematik
Prüfer: ***Professor Weispfenning***
Fächer: ***Roboterkinematik, Computeralgebra***

1 Roboterkinematik, Weispfenning, WS 2002/03

Prof: Nun, wir haben da ja eine ganze Reihe Dinge gemacht. Bei Industrierobotern werden ja häufig **Kugelgelenke** verwendet. Wenn Sie so eines jetzt mit Drehgelenken **simulieren** wollen, was für Möglichkeiten kennen Sie da?

Ich: [Zeichne folgende drei Möglichkeiten auf:]

1. Torsionsgelenk, Drehgelenk mit senkr. Achse, Torsionsgelenk (PUMA)
2. Drehgelenk mit senkr. Achse, Torsionsgelenk, Drehgelenk mit Achse parallel zur ersten (Stanford)
3. Drei Torsionsgelenke, mittlere Achse um 45 Grad geneigt (T3)

Prof: Wie sieht das jetzt im Gegensatz dazu mit **Kardangelenken** aus?

Ich: Da kann man nur um zwei Achsen drehen, also zwei Drehgelenke mit zueinander senkrechten Achsen senkrecht auf Ausgangsachse.

Prof: OK, jetzt dreht sich ja alles hauptsächlich um Bewegungen im $\mathbb{R}^3$. Wie stellt man jetzt solche **Bewegungen** dar?

Ich: Bewegungen sind Kompositionen aus einer Rotation und einer Translation.

Prof: Und wie schreibt man so eine Bewegung auf?

Ich: Als eine **homogene Matrix**. Die hat eine Rotationsmatrix links oben und einen Translationsvektor rechts. Die letzte Zeile ist immer (0,0,0,1).

Prof: Was ist nun eine **Rotationsmatrix**?

Ich: Das ist eine spezielle orthogonale Matrix, also eine orthogonale Matrix mit Determinante 1.

Prof: Solche Rotationen kann man nun auf verschiedene Arten darstellen. Naemlich?

Ich: Also einmal per Rotationsachse und Winkel ... Dann als **Roll-Pitch-Yaw**.

Prof: Wie sieht das aus?

Ich: Man hat ein festes Dreibein. Dann rotiert man zuerst um dessen X-Achse (roll) dann um die alte feste Y-Achse (pitch) und zuletzt um die alte Z-Achse (yaw). Im Gegensatz dazu gibt es noch die **ZYX-Euler-Winkel**. Dort rotiert man jeweils um die neuen, mitrotierten Achsen.

Prof: Gibt es da einen Zusammenhang zwischen diesen beiden Darstellungen?

Ich: Es stellt sich heraus, dass man, wenn man die jeweiligen Rotationen hintereinander multipliziert, nur die Reihenfolge umdrehen muss, um zur selben Stellung zu kommen. Dann gibt es noch **ZYZ-Euler-Winkel**, da rotiert man auch um die neuen Achsen aber die dritte Rotation geht nochmal um die selbe Achse wie die erste Rotation (allerdings ist diese jetzt bereits rotiert).

Prof: Wie sieht das jetzt bei der Kugelgelenksimulation aus? Bei Nummer 3 ist es etwas schwer zu sehen, aber bei den anderen?

Ich: Da rotiere ich - sagen wir mal - zuerst um die X-Achse, dann um die neue Y-Achse und dann wieder um die neue X-Achse. Also ist das vom Typ ZYZ-Euler-Winkel. [Ich glaube, die andere Simulation ist auch von diesem Typ, aber da bin ich mir jetzt nicht ganz sicher; in der Prüfung hatte ich mich da ein wenig verhaspelt.]

Prof: ... wenn Sie ihre Nomenklatur ändern und Z statt X nehmen, ja.
Angenommen man hat jetzt nun so einen Standard-Industrieroboter, also allgemein eine offene kinematische Kette mit mehreren Gliedern. Gibt es nun eine bestimmte Art und Weise, wie man dem Roboter Dreibeine zuweist, so dass die Matrizen besonders einfach werden?

Ich: Ja, da gibt es die sogenannte **Denavit-Hartenberg-Notation**.

Prof: Nehmen wir an, wir haetten nur Drehgelenke. Dann haben Sie also eine "frühere" und eine "spätere" Drehachse. Wie legen Sie jetzt das Dreibein?

Ich: Zuerst lege ich die Z-Achse in Richtung der früheren Drehachse. Dann sehe ich mir die beiden Drehachsen an. Es gibt immer eine (oder mehrere, wenn die Achsen parallel sind) Gerade, die auf beiden Achsen senkrecht steht. Nun wähle ich die Richtung meiner X-Achse in Richtung dieser Senkrechten.

Prof: ... wobei ich da natürlich von wo nach wo gehe?

Ich: Also halt von der früheren Achse zur späteren?!

Prof: Gut, das erste und letzte Dreibein jetzt mal ausgenommen ...
So, nun beschreiben Sie mal bitte die **allgemeinen Problemstellungen** der direkten und der inversen Kinematik.

Ich: Direkte Kinematik bedeutet, dass man die Gelenkparamter (Winkel bzw. Auslenkung) gegeben hat und die Position des Endeffektors bestimmen möchte.
Indirekte Kinematik ist eigentlich der interessantere Fall. Da möchte man wissen, wie man die Gelenkparameter wählen muss, um den Endeffektor in eine bestimmte Position zu bringen.

Prof: Auf welche Problemstellungen laufen diese beiden denn hinaus?

Ich: Also das direkte Kinematik-Problem ist im Endeffekt nur **Matrixmultiplikation**. Das indirekte Problem läuft auf das **Lösen von Polynom-Gleichungen** hinaus.

Prof: Direkte Kinematik ist also eigentlich recht einfach. Wie komme ich nun auf Gleichungen bei indirekter Kinematik? Was für einen Typ von Gleichungen hat man dann und wieviele Gleichungen in **wievielen Variablen** (nur Drehgelenke)?

Ich: [Erstmal ein wenig hin- und her, wie ich von der Matrix-Darstellung zu Gleichungen komme. Dann:] Man hat dann $3 * 4 = 12$ nicht-triviale Gleichungen. Man hat zwar 4x4 Matrizen, aber da die letzte Zeile ja $(0, 0, 0, 1)$ ist, sind diese Gleichungen trivial. Prinzipiell hat man als Variablen $\sin(\varphi_i)$ und $\cos(\varphi_i)$. Also $2 * n$ Variablen, wobei n die Anzahl der Gelenke ist. Man hat demnach gemischt algebraisch-trigonometrische Gleichungen.

Prof: Sie wollten ja aber auf **polynomielle Gleichungen** hinaus, was nun?

Ich: Man kann jedes dieser $\sin(\varphi_i)$ durch eine neue Variable x_i und $\cos(\varphi_i)$ durch y_i ersetzen. Da aber noch $\sin^2(\varphi_i) + \cos^2(\varphi_i) = 1$ gelten muss, muss man das System noch um Gleichungen $x_i^2 + y_i^2 = 1$ erweitern.

Prof: Kennen Sie eine Möglichkeit, die **Anzahl der Variablen zu reduzieren**?

Ich: Man kann $\sin(\varphi_i)$ und $\cos(\varphi_i)$ ersetzen durch Ausdrücke mit $\tan(\varphi_i/2)$ [Gott sei Dank wollte er nicht die genaue Formel dafür haben ...]. Damit hat man die Anzahl der Variablen auf die Hälfte gedrückt.

Prof: Dann führt man also noch wie vorher für die $\tan(\varphi_i)$ Variable x_i ein. Muss man dann noch zusätzliche Bedingung zufügen?

Ich: Aehm, im Prinzip eigentlich nicht. [Ah!] Allerdings muss man beachten, dass ja der Tangens nicht überall definiert ist. Deshalb muss man diese Stellen noch einmal gesondert einzeln nachrechnen.

Prof: Gut, so. Noch letzte Fragen: Können Sie mir mal das Schema des - sagen wir mal - **PUMA-Typ** Roboters zeigen?

Ich: [Male also feste Basis, Torsionsgelenk und zwei Drehgelenke hin.] Dann ist normalerweise noch ein Kugelgelenk der Form 1 vor dem Endeffektor.

Prof: Mhm. Wissen Sie noch, wie ein **Stanford-Arm** aussieht?

Ich: [Ups, vergessen. Male Basis, Torsionsgelenk und dann?] Der hat noch ein prismatisches Gelenk ...

Prof: Genau, dann kann man sich eh schon ausrechnen, wie er aussieht, oder?

Ich: [Also noch ein Drehgelenk und das prismatische hingezeichnet]

Prof: Jawoll. Dann gehen wir doch mal zur Computeralgebra über.

2 Computeralgebra, Prof. Weispfenning, SS 2002

Prof: So, jetzt haben wir ja im Bereich der Roboterkinematik eindrucksvoll gesehen, dass wir gerne die Nullstellen einer Menge von multivariaten Polynomen bestimmen würden. Sagen wir also, wir haben so eine Menge von multivariaten Polynomen, sagen wir mal mit rationalen oder ganzzahligen Koeffizienten. Was können wir jetzt über Anzahl und Lage der gemeinsamen Nullstellen sagen?

Ich: [Hätte es eigentlich lieber gehabt, wenn er im Univariaten angefangen hätte.] Also leider ist es ja nicht so, dass ich das Problem auf den Fall eines einzigen Polynoms zurückführen könnte, wie das im Univariaten der Fall ist, da ein multivariater Polynomring kein Hauptidealbereich ist.

Prof: Ja, sehen wir uns erstmal den **univariaten Fall** an, den brauchen wir dann später vielleicht sowieso.

Ich: [Juhuu!] Die Varietät, also die Menge der Nullstellen eines Polynomsystems ist gleich der Varietät des ggT dieser Menge.

Prof: Wie würden Sie den **ggT berechnen**?

Ich: Mit dem Euklidischen Algorithmus [wollte er gar nicht näher wissen].

Prof: Gut, jetzt sind wir bei einem univariaten Polynom. Was kann man jetzt über die **Anzahl der Nullstellen** sagen?

Ich: Im Allgemeinen ist die Anzahl der Nullstellen kleiner gleich dem Grad des Polynoms. Im Falle eines algebraisch abgeschlossenen Körpers ist die Anzahl gleich dem Grad des quadratfreien Anteils des Polynoms.

Prof: Warum muss ich zum **quatratfreien Anteil** übergehen?

Ich: Weil dort alle Nullstellen einfach sind. [Sollte dann natürlich noch erwähnen, dass ein Polynom dieselben Nullstellen hat, wie sein quadratfreier Anteil].

Prof: Jetzt wissen wir, wie man den quadratfreien Anteil berechnen würde, wenn man die Nullstellen kennt. Aber wie kann man ihn ausrechnen, wenn man diese nicht kennt?

Ich: Dafür muss man die Ableitung eines Polynoms definieren; wie in der Analysis bekannt. Dann gilt $f^* = \frac{f}{ggT(f,f')}$, da man damit alle mehrfachen Nullstellen "rauskürzt".

Prof: Gut, jetzt haben wir eine Abschätzung für die Anzahl der komplexen Nullstellen. Wie sieht es jetzt mit den **reellen Nullstellen** aus?

Ich: Dafür verwendet man die **Sturm'sche Kette bzw. die Sturm-Sylvester-Kette**. Soll ich die mal definieren?

Prof: Nun, ja, machen Sie mal.

Ich: [Wollte ich gleich mit $f_0 = ...$ loslegen, aber da wollte er etwas sauberer hören:] Also ich habe zwei Polynome f und g. Dabei soll f quadratfrei sein und f und g teilerfremd, also $ggT(f, g) = 1$. Zusätzlich habe ich noch ein Intervall (a, b) gegeben. [Weispfenning erinnert mich:] ... wobei noch $f(a) * f(b) \neq 0$ gelten muss.

Prof: Was wäre, wenn zB $f(a) = 0$ wäre?

Ich: Dann würde ich das Polynom durch $(X - a)$ teilen und wüsste, dass die Division aufgeht.
Nun definiere ich eine Folge von Funktionen: $f_0 = f$, $f_1 = f' * g$ (wobei man das g weglässt für die Sturm'sche Kette) und $f_i = -REST(f_{i-2}, f_{i-1})$.
Jetzt habe ich: Anzahl der Nullstellen von f an denen g positiv ist minus Anzahl der Nullstellen von f an denen g negativ ist, ist gleich der Anzahl der Zeichenwechsel der Sturm-Sylvester-Kette von f und g ausgewertet an der Stelle a minus demselben ausgewertet an b, also:
N(f,g,1,(a,b))-N(f,g,-1,(a,b)) = ZW(SS(f,g)(a))-ZW(SS(f,g)(b))

Prof: Nun sieht es ja so aus, als hätte man zwar die Differenz aber nicht die interessierenden Größen selbst?

Ich: Stimmt, allerdings sind die interessierenden Größen **Lösungen folgenden Gleichungssystems**:

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & -1 \\ 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \text{N(f,g,1,(a,b))} \\ \text{N(f,g,0,(a,b))} \\ \text{N(f,g,-1,(a,b))} \end{pmatrix} = \begin{pmatrix} ZW(S(f^*)(a)) - ...(b)) \\ ZW(SS(h, g)(a)) - ...(b)) \\ ZW(SS(h, g^2)(a)) - ...(b)) \end{pmatrix}$$

[Ich glaube, Weispfenning war ganz angetan, dass ich das ein wenig erklären konnte, deshalb wiederhole ich das auch mal:]
Die erste Zeile in der rechtesten Matrix bekomme ich ja, wenn ich die Nullstellen von f nehme, egal was g ist; deshalb ist es einfach die Differenz der ZW der Sturm'schen Kette. Allerdings brauche ich für diesen Satz ein quadratfreies Poylnom, also f^*. Zweite Zeile bedeutet Nullstellen von f mit $g > 0$ minus Nullstellen von f mit $g < 0$, also genau die Formel in der Antwort davor. Allerings muss f quadratfrei und teilerfremd zu g sein. Deshalb verwendet man $h = f^*$, dann ist es quadratfrei und teilt es noch durch den ggT von f und g um Teilerfremdheit zu garantieren.

Prof: Schön. Jetzt haben wir das also für den univariaten Fall. Wie sieht das jetzt mit **multivariaten Polynomsystemen** aus?

Ich: Nun, da funktioniert das ja nicht so schön, dazu brauche ich dann Gröbnerbasen.

Prof: Sehen wir das mal als Black-Box an und gehen davon aus, wir hätten also für Ideale eine Gröbnerbasis. Wie hilft uns jetzt das weiter für die Nullstellensuche?

Ich: Es gibt den **Hilbertschen Nullstellensatz**, der besagt, dass die Varietät eines Systems von Polynomen F eine Teilmenge der Varietät eines einzelnen Polynoms g ist, genau dann wenn $g \in Rad(F)$, also eine Potenz von g im Ideal von F ist. Das bedeutet speziell, dass $V_L(F) = \emptyset$ genau dann wenn $1 \in Id(F)$. Und das testet man schnell, da dies genau dann der Fall ist, wenn die **Gröbnerbasis**, die $Id(F)$ erzeugt ein **konstantes Polynom** enthält.

Prof: Jetzt haben Sie aber eine wichtige Voraussetzung des HNS vergessen. [Ups, was denn???] Was ist das denn für ein Grundkörper L?

Ich: Ach ja natürlich. Das muss ein **algebraisch abgeschlossener Körper** sein, also zum Beispiel $\mathbb{C}$.

Prof: So, jetzt haben wir also ein Kriterium dafür, dass das System überhaupt eine Nullstelle hat. Was kann man jetzt über die Anzahl genauer sagen? Also erstmal ob es **endlich viele Nullstellen** gibt und dann vielleicht noch genauere Abschätzung? [Langsam wurde auch die Zeit knapp, deshalb alles etwas oberflächlicher.]

Ich: Also, erstmal kann sagen, dass wenn man ein **nulldimensionales Ideal** hat, dann ist die Anzahl der komplexen Nullstellen endlich. Dabei heißt nulldimensional, dass die Dimension des Restklassenrings $K[X_1, ..., X_n]/I$ für ein Ideal I endlich ist. Diese Dimension ist dann auch eine Obergrenze der Anzahl der Nullstellen.

Prof: Und wie bekomme ich jetzt diese **Dimension**?

Ich: Mittels der Menge der reduzierten Terme bezüglich der Gröbnerbasis kann man eine Basis des Restklassenrings darstellen und daher ist die Kardinalität dieser Menge gleich der Dimension.

Prof: Was ist die **Menge der reduzierten Terme**?

Ich: Das sind diejenigen Terme, die in Normalform sind, das heißt sich bezüglich der Gröbnerbasis nicht mehr weiter reduzieren lassen. Also alle Terme, für die es kein Element in der Gröbnerbasis gibt, dessen höchster Term den Term teilt.

Prof: OK, dann sind wir eh schon über der Zeit und ich gebe mal an Herrn Graf weiter. Wie hieß die Vorlesung doch gleich noch mal?

Professor Weispfenning ist ein sehr ruhiger und angenehmer Prüfer, der einem gerne Zeit zum Nachdenken gibt. Wenn er merkt, dass man mal nicht so recht weiter weiß, dann hilft er auch schnell mal weiter. Ich hatte nicht den Eindruck, dass er besonders bohren würde, wenn man wo nicht so bewandert ist. Bei mir hat er eher auf einen groben Überblick Wert gelegt als auf kleine Details und Ausnahmen.
Ich hatte mich eigentlich schon darauf vorbereitet, dass er gerne ein zwei Dinge bewiesen hätte, aber das hat er zu Gunsten von den allgemeinen Verfahren und Methoden ziemlich weggelassen. Wenn man allerdings etwas begründen kann, dann freut er sich auch, das zu hören. Mich hat er auch keine Beispiele ausrechnen lassen (eigentlich schade, da geht ja immer gleich ein ganzes Stück Zeit drauf ...).
Benotung: "Kam ja alles was ich hören wollte" ...

11. März 2003
Prüfungsprotokoll
Nebenfach Mathematik
Prüfer: ***Professor Graf***
Fach: ***Bilddatenkompression***

3 Bilddatenkompression, Prof. Graf, WS 2002/03

Prof: So, also wir haben eine Menge digitalisierter Bilder. Nun will man diese natürlich möglichst gut komprimiert speichern. Was für Methoden gibt es denn da?

Ich: [Wo soll man denn da anfangen?] Also man muss die Bilder kodieren in eine Zeichenfolge von Nullen und Einsen. Dies versucht man so zu machen, dass die Länge dieses Codes möglichst klein ist.
Es gibt prinzipiell mal verlustfreie und verlustbehaftete Methoden.

Prof: Was gibt es denn mal für **verlustfreie Methoden**?

Ich: Da ist zum Beispiel der **Huffman-Code**.

Prof: Am Besten Sie erzählen mir mal wie der so funkioniert und was für Eigenschaften der hat?!

Ich: Also man hat gegeben die Wahrscheinlichkeiten der einzelnen Grauwerte $p_1, ..., p_k$, also k verschiedene Grauwerte. Nun suche ich diejenigen beiden Grauwerte l_1 und l_2, deren Wahrscheinlichkeit am kleinsten ist. Verbinde die beiden zu einem neuen Grauwert l^* mit Wahrscheinlichkeit $p_{l^*} = p_{l_1} + p_{l_2}$ und beschrifte die Verbindungen mit 0 und 1.
Dieses Verfahren setze ich fort, bis ich nur noch einen einzigen Grauwert habe. Dann ist der Code für einen Grauwert l gegeben durch die Beschriftung auf dem Pfad zwischen Wurzel des Binärbaums und dem Blatt p_l.

Prof: Was mache ich jetzt mit diesen Codes?

Ich: Man schreibt die Codes einfach hintereinander, ohne irgendein Trennzeichen. Man kann dann die einzelnen Codes ohne weiteres wieder trennen.

Prof: Woran liegt das?

Ich: Die Definition eines Codes verlangt, dass **kein Code Anfangsstück eines anderen Codes** ist. Daher weiß ich, dass wenn ein Code passt, es der richtige ist.

Prof: Und wieso ist das jetzt bei dieser Konstruktion der Fall?

Ich: [Aeh?! Oh! Mit Graf zusammen]: Nun einfach deshalb, weil ich beim Code ablesen immer bis zu den Blättern gehe und nicht vorher abbrechen kann.

Prof: Warum verwendet man jetzt diese Methode? Man könnte sich ja sonstwelche Arten der Kodierung überlegen?

Ich: Nun, dieser Code ist in einer gewissen Weise **optimal**. Nämlich liegt die mittlere Bildcodelänge in der Nähe der ideellen Entropie.

Prof: Was ist denn die mittlere **Bildcodelänge**? Können Sie die mal definieren?

Ich: Also, M_K ist ...

Prof: K steht dann wohl für den Code.

Ich: Genau, ich summiere über alle Bilder und multipliziere die Wahrscheinlichkeit des Auftretens dieses Bildes mit der Länge des Codes für dieses Bild.

Prof: Und was ist diese Länge des Codes für das Bild?

Ich: Die Summe der Länge aller Codes für die Grauwerte aller Pixel, also insgesamt:

$$M_K = \sum_{f \in M_{m,n}(G)} P(f) \cdot \sum_{(i,j) \in I} |K(f_{ij})|$$

Dabei steht $M_{m,n}(G)$ für die Menge der Bilder über eine Grauwertmenge G und I für die Menge der Pixel.

Prof: Was gibt es für einen **Zusammenhang zu den** p_is von vorher?

Ich: Natürlich kann ich M_K auch über diese ausrechnen. Das ist dann die Summe über alle Grauwerte von p_l mal die Länge des Codes für dieses l, also:

$$M_K = mn \cdot \sum_{l \in G} p_l \cdot |K(l)|$$

Da muss ich dann noch mal die Anzahl der Pixel nehmen ...

Prof: Jetzt haben Sie etwas über Optimalität gesagt. Was muss da gelten?

Ich: Also die mittlere **Bildcodelänge eines optimalen Codes** ist gleich dem Minimum aller M_K für K beliebiger Code.

Prof: Gibt es denn immer so ein Minimum?

Ich: [Hoffentlich will er da nicht was bewiesen haben!!] Ja, man kann zeigen, dass ein Minimum existiert.

Prof: Können Sie mir so ein Minimum angeben?

Ich: [Voll auf dem Schlauch steh!] Nun, das ist schwierig ...

Prof: Sind Sie sicher?

Ich: Ach so der Huffman Code ist ja so ein Beispiel.

Prof: Genau, wie ist das jetzt mit der **Entropie**?

Ich: Also die ideelle Entropie ist definiert als $\sum_{l\in G} -p_l \log_2 p_l$. Die mittlere Bildcodelänge pro Pixel des Huffman Codes, also $\frac{1}{mn} M_K$ liegt nun zwischen der Entropie und der Entropie +1. Dies ist bei optimalen Codes der Fall.

Prof: Nun gibt es neben dem Huffman-Code auch noch eine andere Art der Kodierung die oft angewandt wird, die [Spannung ...] **arithmetische Kodierung**. Können Sie mir erklären, wie diese funktioniert?

Ich: [Der nächste Teil war mit kleinen Einwürfen/Fragen von Prof. Graf gespickt:] Also im Gegensatz zum Huffman Code werden bei der arithmetischen Kodierung nicht Codes für jeden einzelnen Grauwert erzeugt, sondern für das gesamte Bild. Man betrachtet nun die Grauwerte des Bildes als Sequenz $i_1, \ldots, i_k$, mit Anzahl der Pixel $= k$. Die Wahrscheinlichkeiten der Grauwerte $p_1, \ldots, p_q$ addieren sich ja zu 1, d.h. sie unterteilen das Einheitsintervall. Definiere nun die Startpunkte dieser Intervalle mittels $t_i = \sum_{j=i}^{i-1}$. Dann eine Abbildung vom Einheitsintervall in sich per $w_i(s) = t_i + p_i \cdot s$. Diese verwende ich jetzt zur Definition der Abbildung, die mir für ein Bild eine einzige Zahl zurückliefert: Definiere $s_{i_1,\ldots,i_k}(1) = w_{i_1} \circ \ldots \circ w_{i_k}(1)$.
Nun sehe ich mir die Binärdarstellung dieser Zahl an

$$s_{i_1,\ldots,i_k}(1) = \sum_{i=1}^{\infty} \sigma_i \cdot 2^{-i}$$

wobei nicht schließlich alle $\sigma_i = 0$. Dann nehme ich die ersten

$$r = \lceil -\log_2(p_{i_1} \cdot, \ldots, \cdot p_{i_k}) \rceil$$

Sigmas als Kodierung für das Bild.

Prof: Nun reden wir noch ein wenig über **verlustbehaftete Kompression**. Was können Sie mir über die **Karhunen-Loève-Transformation** sagen? Definition, Berechnung und vor allem Eigenschaften?

Ich: [Darauf hätte ich nun ganz gerne verzichtet. Ich stand auch etwas auf dem Schlauch und Prof. Graf hakte immer wieder etwas nach / wies die Richtung. Es hat mich etwas erstaunt, dass er sich da auch mit nicht so ganz einwandfreien und seinem Skript entnommenen Formulierungen zufrieden gab. Ich geb hier auch nicht das genau Hin- und Her des Gesprächs wieder:]
Die Karhunen-Loève Transformation ist in vielerlei Hinsicht optimal. Man geht wie folgt vor: Man hat die Zufallsvariable, die die Bilder beschreibt und unterwirft dieser einer Transformation. Dann projeziert man das Ergebnis auf eine Menge M, wirft also alle Informationen über Pixel, die nicht in dieser Menge sind weg. Zur Rekonstruktion wendet man dann darauf T^{-1} an. Den Fehler, den man dabei macht versucht man nun zu minimieren.

Prof: Was genau möchte man minimieren?

Ich: $E(\|X - T^{-1} \circ \Pi_M \circ T \circ X\|^2)$ [im Skript ist eigentlich die Zufallsvariable noch explizit zentriert, also $X - E(X)$ statt X angegeben ...]
Nun ist die KL-Transformation diejenige, die für alle $N \in \mathbb{N}$, wobei N die Kardinalität der Menge M ist und alle bijektiven Transformationen diesen Erwartungswert minimiert.

Prof: Was brauche ich jetzt um so eine KL-Transformation zu bekommen?

Ich: Man braucht zuerst einmal die **Kovarianz-Transformation** T_K.

Prof: Schreiben Sie da mal die Definition hin.

Ich: Das ist die Transformation, die die **Autokovarianzfunktion** als Kern hat. Sie ist definiert als

$$K(i,j;k,l) = E((X_{ij} - E(X_{ij})) \cdot (X_{kl} - E(X_{kl})))$$

Nun benötige ich eine Orthonormalbasis aus Eigenvektoren von T_K.

Prof: Kann ich denn so eine immer finden? Und wieviele Elemente hat sie?

Ich: So eine Basis existiert immer, da die Kovarianz-Transformation positiv-semidefinit ist und damit insbesondere **selbstadjungiert** ist. Sie enthält Anzahl der Pixel viele Vektoren.

Prof: Und wie bekomme ich jetzt eine KL-Transformation? **Definition**?

Ich: Dazu nehme ich das Skalarprodukt von dem zu transformierenden Bild mit den entsprechenden Basisvektoren. Die Transformation an Pixel (i,j) ist also:

$$(L_\varphi f)_{ij} = (f | z^{\varphi^{-1}(i,j)})$$

Wobei φ^{-1} eine Nummerierung der Pixel ist, also eine Abbildung $\varphi^{-1} : I \to \{1, \dots, mn\}$ mit $mn =$ Anzahl der Pixel.

Prof: So, dann denke ich, ist die Zeit gut vorbei, wenn Sie einen Moment draußen warten würden ...

Professor Graf hat sich für mich überraschend wenig auf Formalitäten und Beweise gestürzt. Natürlich müssen die zentralen Definitionen halbwegs sitzen (mittlere Bildcodelänge, Entropie, Transformationen, ...). Aber im Großen und Ganzen legte er Wert darauf, dass man wusste, wie denn die Techniken funktionieren. Ob man das dann genauso formuliert wie im Skript oder nicht, war nicht so wichtig.
Allerdings fragt er gerne bei Kleinigkeiten nach und verbreitet für meine Begriffe eher ein wenig Hektik. Unterliegendes, grundlegendes Wissen hat er nicht geprüft. Er ergänzt immer ganz gerne was man sagt.
Mich hat etwas irritiert, dass er öfters etwas, das ich gerade gesagt hatte noch einmal etwas anders formuliert selber wiederholt hat. Ich wusste nicht immer genau, ob ich etwas Falsches gesagt hatte. Aber offensichtlich moechte er da nur sicherstellen, dass er den Prüfling auch richtig verstanden hat.
Obwohl ich nicht den Eindruck hatte, besonders geglänzt zu haben, gab es an der Note nichts zu meckern.