Studienarbeit
zur
Kosinus und Fourier Transformation

Studienarbeit
von
Holger Wons
aus
Niederkrüchten - Elmpt

vorgelegt am
Lehrstuhl für Praktische Informatik IV
Prof. Dr. Effelsberg
Fakultät für Mathematik und Informatik
Universität Mannheim

Mai 2000

Betreuerin : Dipl.-Math. oec. Claudia Schremmer

Inhaltsverzeichnis:

Vorwort

Als ich mich am Anfang des Wintersemesters 1999/2000 endlich dazu durchgerungen hatte, mir eine Studienarbeit zu suchen, bestand meine eigentliche Motivation schlicht darin, schnell etwas zu programmieren und damit schnell das Thema Studienarbeit abzuhaken. Dazu klapperte ich – salopp formuliert – die einzelnen Lehrstühle der Universität Mannheim ab, in der Hoffnung ein Thema zu finden, welches mich nicht allzu sehr in Anspruch nehmen würde. Schon am zweiten Lehrstuhl meiner Suche wurde ich fündig – aber nicht aufgrund der zuvor genannten „Motivation".

Da ich mich privat selber schon mit Video- und Bildbearbeitung beschäftigt habe und z.B. vorhandene Komprimierungsalgorithmen und Bildformate immer als gegeben angesehen habe, ohne zu wissen wie so etwas eigentlich funktioniert, war das Thema dieser angebotenen Studienarbeit – ein Applet zur Visualisierung der diskreten Kosinus Transformation – genau das, was ich insgeheim zu finden gehofft hatte. Ein Thema das mich wirklich interessierte! Ich befürchtete zwar, dass mich die zugrundeliegenden mathematischen Aspekte überfordern würden, doch zerstreuten sich diese Befürchtungen nach einiger Zeit. Ein weiterer Anreiz, die angebotene Studienarbeit zu akzeptieren, welche für meinen Studiengang der Wirtschaftsinformatik immer als Implementation eines Programms zu erfolgen hat, war die Tatsache, dass jene Implementation in der Programmiersprache Java zu schreiben war und somit eine Realisierung als Applet im Vordergrund stand. Nach der Erstellung einer eigenen Homepage, hatte ich das Bedürfnis den Umgang mit der Internettechnologie auf einer professionelleren Ebene fortzuführen. Ob dies auf der Software- oder der Hardwareseite geschehen sollte, war mir nicht so wichtig.

Zu Beginn der Studienarbeit hatte ich wirklich keine Ahnung von Java oder der DCT. Aber wenn ich jetzt auf die Zeit zurückblicke, in der ich verzweifelt und den Tränen nahe vor dem Bildschirm meines PCs saß, weil sich unlösbar erscheinende Probleme aufzeigten, die sich aber mit dem Fortschreiten der Zeit und dem damit einhergehenden Lernprozess schnell auflösten, so möchte ich behaupten, mindestens eine zufriedenstellende Arbeit abgeliefert zu haben, dass ich eine Menge gelernt habe, und dass ich nicht zuletzt auch meinen Spaß bei der Arbeit hatte.

Mannheim, den 28. April 2000 Holger Wons

1. Einleitung

1.1. Motivation und Aufgabenstellung

Die dieser Studienarbeit zugrundeliegende Aufgabe, war die Erstellung eines Applets in der Programmiersprache Java. Das Programm soll der Visualisierung der Anwendung von zwei Transformationsarten dienen: der diskreten Kosinus Transformation und der diskreten Fourier Transformation. Das entwickelte Programm ist ein kleiner Baustein für das Projekt VIROR und gehört zum Teilprojekt *Inhaltserstellung Informatik – Fach Multimediatechnik*.

Das Projekt VIROR - Virtuelle Hochschule Oberrhein - hat sich den Aufbau einer semivirtuellen Universität zum Ziel gesetzt. Studierende sollen zusätzlich zum traditionellen Veranstaltungsangebot in Hörsälen, Seminaren und Übungsgruppen die Möglichkeit bekommen, mit Hilfe des Computers von zu Hause aus Lernstoff über das Internet abzurufen.

Heute gängige Verfahren zur Visualisierung von Standbildern und Videoströmen (JPEG, Motion JPEG, H.261, H.263 etc.) greifen häufig die diskrete Kosinus Transformation (Discrete Cosine Transformation = DCT) auf, welche ein typisches Beispiel für verlustbehaftete Codierung ist.
Bei Studierenden wurde festgestellt, dass diese oft große Schwierigkeiten haben, die DCT vollständig zu verstehen, wobei zum Beispiel die Zerlegung eines Eingangssignals in den Frequenzraum, das Zusammenspiel von Ortsraum und Frequenzraum und die Rücktransformation die primären Probleme darstellen. Es lag also Nahe den Studenten ein Werkzeug zur Verfügung zu stellen, welches zur Beseitigung der Verständnisprobleme beitragen soll.

1.2. Programmierrahmen

Die Implementation des Applets erfolgt in Java 1.2 unter Zuhilfenahme der Swing-Klassen zur Gestaltung der Benutzeroberfläche. Die Benutzerführung erfolgt selbstverständlich in englischer Sprache, um den Einsatzbereich der Arbeit nicht nur auf den deutschsprachigen Raum zu beschränken.

1.3. Aufgabenschritte

Im folgenden werde ich die Aufgabenschritte aufzeigen, wie sie meiner Arbeit zugrunde liegen, und kurz erläutern.

Im ersten Schritt ist natürlich die Einarbeitung in die Thematik der diskreten Kosinus und Fourier Transformation von Nöten gewesen, da ich zu diesen Themen keinerlei Vorwissen einbringen konnte. Aber obwohl der mathematische Aspekt der Studienarbeit sicherlich ein größeres Problem war, musste ich diesen erst einmal zurückstellen und mich zunächst mit dem Erlernen der Programmiersprache Java beschäftigen, weil ich ebenfalls keine Erfahrungen mit den Vorteilen, Nachteilen und Eigenheiten dieser Sprache hatte.

So darf es nicht verwundern, wenn ich die Implementation des Applets direkt mit einer der Todsünden des Programmierens begonnen habe, nämlich mit dem planlosen „drauflosprogrammieren“, was für das Verständnis einer unbekannten Programmiersprache aber unumgänglich ist („Learning by doing“).
Nach dem Studium einschlägiger Literatur zu den Themen Java, Java-Swing, Kosinus und Fourier Transformation ([1],[2],[3],[5]), konnte ich mich der Appletentwicklung mit den weiter unten folgenden Inhalten widmen.

Das Applet sollte die folgenden Funktionen und Eigenschaften besitzen :

- *Das Applet soll den allgemeinen Fall - den der DFT - und den davon abgeleiteten Spezialfall der DCT veranschaulichen.*

  Die DCT ist ein Grundstein für den Komprimierungsalgorithmus, der von der Joint Photographic Experts Group (JPEG) entwickelt und auch nach ihr benannt wurde. Wird ein Standbild mit dem JPEG-Algorithmus komprimiert, wird unter anderem das benutzte Bild in Blöcke von 8x8 Bildpunkten zerlegt. Auf die einzelnen Blöcke wird dann die DCT einzeln angewendet. Da die DCT somit auf *einzelne* Zahlen angewendet wird, spricht man von einer *diskreten* Transformation. Das Programm soll den eindimensionalen Fall veranschaulichen, was zur Folge hat, dass die Zerlegung eines Bildes in einen 1x8 Block durchgeführt wird und sich so nicht 64 zu transformierende Werte ergeben, sondern nur acht.
  Die DCT ist ein Sonderfall der diskreten Fourier Transformation (DFT). Es werden beide Fälle im Programm veranschaulicht.

- *Es soll eine Auswahlbox verschiedener diskreter eindimensionaler Eingangssignale geben. Dabei wird jede Funktion sowohl als Werte-Array angezeigt, als auch geplottet. Der zu zeigende Periodenbereich sollte dabei im Intervall [0;7] liegen, zum einen aufgrund der Anwendung in der Bildverarbeitung, zum anderen aufgrund des JPEG-Standards.*

  Mit dem Anzeigen der Funktion ist gemeint, dass nach Auswahl eines Eingangssignals, eben diese Auswahl weiterhin irgendwo sichtbar sein soll und der Benutzer nicht erst durch unnötige Aktionen wieder in Erfahrung bringen muß, was er eben eingestellt hat. Mit dem Plotten der Funktion soll dem Benutzer eine grafische Repräsentation der Signale gegeben werden, also durch das Anzeigen eines Koordinatensystems mit anschließenden Zeichnen der Funktion, die sich aus den Eingangsignalen ergibt. Das Intervall [0;7] wurde gewählt, da eine Transformation veranschaulicht werden soll, welche mit acht Werten zu vollziehen ist, wie aus dem vorherigen Punkt bekannt ist.

- *Die Amplituden bzw. Koeffizienten sollen textuell ausgegeben werden. Die Rekonstruktion des Signals soll in einer anderen Farbe über das Eingangsignal geplottet werden.*

  Dies bedeutet, dass der Benutzer auch erkennen soll, was er mit den Schiebereglern (siehe nächster Punkt) bewirkt. Wenn man beispielsweise einen Schieberegler etwas nach oben bewegt, so wirkt sich diese Aktion sofort sichtbar für den Benutzer aus. Zum einen anhand der Veränderung des Plotts der Funktion, die aus den Werten rekonstruiert wird, welche der Benutzer mit den Reglern eingestellt hat, zum anderen anhand der konkreten Zahlenwerte, die in Textfeldern angezeigt werden.

- *Jede mögliche Frequenz des Spektrums bekommt einen Schieberegler.*

  Der Koeffizient einer Frequenz wird durch einen Schieberegler eingestellt. Jede mögliche Einstellung eines Schiebereglers, die innerhalb eines minimal- und maximal einstellbaren Bereiches liegt, entspricht einem bestimmten Wert. Genauso bewegen sich die möglichen Ausprägungen eines Koeffizienten innerhalb einer minimalen Amplitude und einer maximalen Amplitude. Durch die Verwendung eines Schiebereglers, wird die Assoziation der Position des Reglers mit dem Wert des Koeffizienten beabsichtigt. Verändert der Benutzer die Position des Schiebereglers, so kann er direkt feststellen, welche Auswirkung die Veränderung des Koeffizienten der Frequenz auf den Plott der Rekonstruktion hat (siehe auch „Das **scrollbar panel**“ im Abschnitt 3.2.2.).

- *Der Plott des rekonstruierten Signals soll sich nach den Einstellungen der Schieberegler richten. Werden Schieberegler verstellt, so ändert sich der Plott der Rücktransformation.*

  Dieser Punkt wird aus den beiden vorherigen sofort klar !

- *Das Applet soll eine intuitive Oberfläche erhalten.*

  Damit ist eine Benutzeroberfläche gemeint, die anwenderfreundlich ist. Was sich hier so einfach anhört, ist leider nicht ganz so einfach zu realisieren, weshalb ich auf die Konstruktion der Benutzeroberfläche des Applets im dritten Kapitel noch etwas ausführlicher eingehen werde.

- *Bei der Kosinus Transformation wird der Quantisierungsschritt hinzugefügt.*

  Während man bei der diskreten Fourier Transformation die Koeffizienten dadurch erhält, dass das Eingangssignal einfach nur transformiert wird, wird bei der diskreten Kosinus Transformation noch ein weiteren Schritt vollführt, indem man die vorläufigen Koeffizienten durch eine Zahl dividiert und sich so die entgültigen Koeffizienten ergeben. Wird ein Schieberegler betätigt, wird der zugehörige selbst eingestellte Koeffizientenwert zunächst mit der Zahl multipliziert, mit der eben geteilt wurde. Danach erfolgt die Rücktransformation und das Plotten dieser Rekonstruktion. Jetzt bemerkt man: je höher die Zahl ist, durch die man dividiert, desto ungenauer wird die Rekonstruktion aufgrund von Rundungsfehlern. D.h. die beiden Funktionen („Original“ und „Rekonstruktion“) überlagern sich nicht mehr vollständig, obwohl man die „richtigen“ Koeffizienten eingestellt hat.

Nach dieser Einführung folgt nun im nächsten Kapitel der mathematische Hintergrund dieser Studienarbeit.

2. Mathematischer Hintergrund von DCT und DFT

Da sich der Algorithmus zur Übertragung eines Standbildes im JPEG-Bildformat unter anderem auf die diskrete Kosinus Transformation stützt, womit eine Kompression aufgrund der Quantisierung einhergeht, möchte ich in diesem Kapitel kurz erläutern, warum überhaupt die Kompression von Standbildern und Videos notwendig ist. Ebenfalls werde ich den mathematischen Hintergrund der Transformation im allgemeinen, und der Transformation in den konkreten Fällen der DFT und der DCT beleuchten.

2.1. Notwendigkeit der Kompression

Dass Kompression notwendig ist, möchte ich anhand von vier Beispielen belegen. In jedem Beispiel wird gezeigt, wie viele Daten auf einer handelsüblichen 1.44 MByte Diskette Platz finden. Auf eine Diskette passen, bei Formatierung mit dem FAT Dateisystem, 1.423 KByte. Das entspricht 1.457.152 Byte und das wiederum 11.657.216 Bit.

- **Normaler Text**:
  Eine Seite Text bestehe aus 64 Zeilen, jede Zeile bestehe aus 80 Zeichen und ein Zeichen sei 1 Byte groß. Das Datenvolumen einer Seite ist also:
  80 Byte/Zeile x 64 Zeilen/Seite x 8 = 40960 Bit/Seite. Es passen somit
  11.657.216 Bit / 40960 Bit/Seite = 284,6 Seiten Text auf eine Diskette.

- **Standbilder**:
  Ein Standbild bestehe aus 512x512 Bildpunkten, jeder Bildpunkt besitze eine Farbtiefe von 24 Bit (je acht Bit für Rot-, Gelb- und Blauanteil). Ein Bild hat folglich eine Größe von:
  512 x 512 Bildpunkte/Bild x 24 Bit/Bildpunkt = 6.291.456 Bit/Bild. Man kann 11.657.216 Bit / 6.291.456 Bit/Bild = 1,9 Bilder auf Diskette speichern. Besser gesagt: Es passt nur ein einziges Bild auf eine Diskette.

- **Audio in CD Qualität**:
  Es wird eine Sample Frequenz von 44.100 Hz bei 16 Bit/Sample und Stereoton verwendet. Also:
  44.100 Samples/Sekunde x 16 Bit/Sample x 2 = 1.411.200 Bit/Sekunde. Auf eine Diskette passen 11.657.216 Bit / 1.411.200 Bit/Sekunde = 8 Sekunden Audio.

- **Video**:
  Es sollen 30 Bilder pro Sekunde aufgenommen werden. Jedes Bild besitze eine Größe von 360x240 Bildpunkten (also 86.400 Bildpunkte) und jeder Bildpunkt habe eine Farbtiefe von 24 Bit. Das ergibt:
  86.400 Bildpunkte x 24 Bit x 30 Bilder/Sekunde = 622.080.000 Bit/Sekunde. Auf eine Diskette passen somit 11.657.216 Bit / 622.080.000 Bit/Sekunde = 0,19 Sekunden Video.

Man sieht also, dass die Archivierung und Übertragung von Daten ein Problem darstellt, da das Datenvolumen zu groß ist. Möchte man z.B. eine Stunde Videomaterial archivieren, so wäre ein Speichermedium nötig mit einer Kapazität von
3600 Sekunden x 622.080.000 Bit/Sekunde = $2{,}24 \times 10^{12}$ Bit.
Um das Problem zu lösen, muss man das Datenaufkommen reduzieren, indem die Daten komprimiert werden. Es gibt zwei Grundideen bei der Datenkompression: einerseits kann ein verlustfreies Kompressionsverfahren verwendet werden oder andererseits kann ein verlustbehaftetes Kompressionsverfahren benutzt werden:

Verlustfreie Kompressionsverfahren zeichnen sich dadurch aus, dass nach der Rekonstruktion zuvor komprimierter Daten (z.B. Bilder, Videos oder Audioinformationen), die Originalinformationen wieder vollständig wiederhergestellt werden können.
Die Idee der verlustbehafteten Komprimierung ist, Informationen, welche für den Menschen gar nicht oder nur teilweise wahrzunehmen sind, einfach aus den Originaldaten zu entfernen und dadurch die zu komprimierende Datenmenge zu verkleinern.
Als Bespiel für eine verlustbehaftete Komprimierung lässt sich die Umwandlung von Audiodaten in das MP3 Format aufführen. Hier wird unter anderem eine Kompression dadurch erzielt, dass für das menschliche Ohr unhörbare Frequenzen nicht abgespeichert werden. Das menschliche Ohr vermag Frequenzen aus einem Bereich von 20 Hz bis 20.000 Hz wahrzunehmen. Im Gegensatz zu Audiodaten bietet sich bei Bildern eine Möglichkeit das Datenvolumen zu verringern, durch die Wahl des Farbmodells zur Speicherung des Bildes. Benutzt man das RGB Farbmodell so werden für jeden einzelnen Bildpunkt die Werte für Rot-,Gelb- und Blauanteil gesichert (4 Pixel x 3 Farben x 8 Bit = 96 Bit/4 Pixel). Macht man sich hingegen die Schwäche des menschlichen Sehvermögens zunutze, empfindlicher auf Helligkeitsveränderungen als auf Farbveränderungen zu reagieren, kann man das YUV Farbmodell heranziehen. Dort werden zwar, wie beim RGB Modell auch, drei Komponenten gespeichert, aber nicht für jeden Bildpunkt. Bei YUV kommt nämlich noch ein Quantisierungsschritt hinzu.

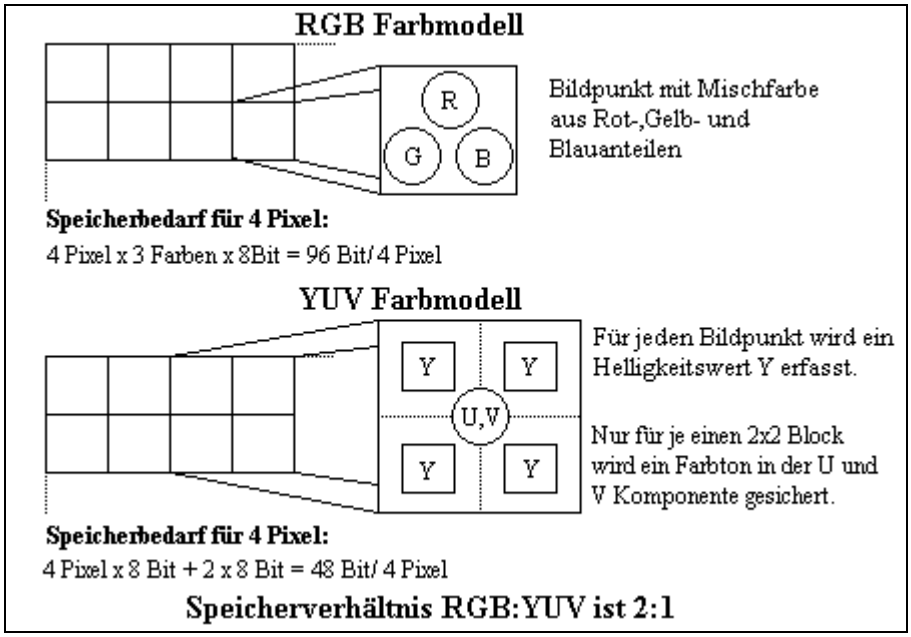

Abbildung 2-1 : Die RGB und YUV Farbmodelle.
Beim RGB Farbmodell ist jeder Bildpunkt aus der Mischfarbe von Rot-, Gelb- und Blauanteil zusammengesetzt. Das YUV Farbmodell sichert für jeden Bildpunkt den Helligkeitswert, aber nur für einen 2x2 Bildpunkte großen Block die Farbwerte U und V (Chrominanz).

In der Y-Komponente wird für jeden Bildpunkt der Helligkeitswert gesichert, aber nur für jeden 2x2 Bildpunkte großen Block wird ein Farbwert erfasst (siehe Abbildung 2-1). Betrachtet man nur die Farbkomponenten (U und V Komponente), so hat das Bild nur noch die halbe Auflösung der Helligkeitskomponente. Durch die bereits genannte Sehschwäche des menschlichen Auges erzielt man eine Reduzierung des Datenvolumens, durch Bevorzugung des YUV Bildformates gegenüber dem RGB Bildformat, auf die Hälfte des Volumens
(4 Pixel x 8 Bit + 2 x 8 Bit = 48 Bit/4 Pixel) bei einem Informationsverlust, der kaum oder sogar überhaupt nicht feststellbar ist.

2.2. Transformation im Allgemeinen

Die eigentliche Grundidee der Transformation ist relativ schnell erklärt :
Durch Transformieren versucht man die Komplexität einer gestellten Aufgabe zu vermindern. Das möchte ich anhand des nächsten Beispiels verdeutlichen.

Wurde die Aufgabe gestellt, 100 beliebig große Zahlen Y_0 bis Y_{99} durch Bildung der Quotienten X_0/Z_0 bis X_{99}/Z_{99} zu berechnen, wobei man keinen Taschenrechner zur Hand hat, so wird man wohl einige Zeit damit verbringen, die Berechnung nach Schulmethode durchzuführen. Da aber glücklicherweise eine Logarithmentafel vorhanden ist, kann man die gestellte Aufgabe schneller lösen, beispielhaft gezeigt für die erste der hundert Rechnungen, also $Y_0 = X_0/Z_0$: Anstatt X_0 durch Z_0 zu dividieren, schlägt man in der Logarithmentafel die Werte für $\log(X_0)$ und $\log(Z_0)$ nach und führt anschließend eine Subtraktion aus, also $\log(X_0) - \log(Z_0)$. Es ergibt sich $\log(Y_0)$. Nun sucht man in der Logarithmentafel den Wert für $\log(Y_0)$ heraus und erhält die Lösung für $Y_0 = X_0/Z_0$.

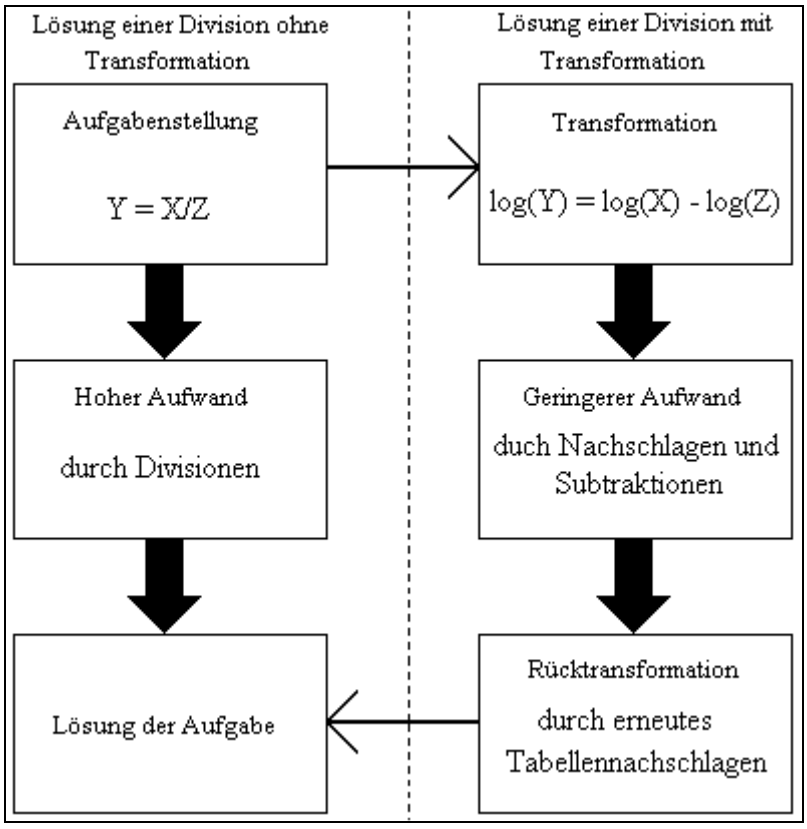

Abbildung 2-2 : Die Möglichkeit eine Aufgabe mit und ohne die Anwendung der Transformation zu lösen.

Was hat das ganze jetzt mit Transformation zu tun? Ganz einfach :

Man hat die (allgemeine) Problemstellung $Y = X/Z$ transformiert und zwar zu $\log(X) - \log(Z) = \log(Y)$. Indem man den Wert für $\log(Y)$ in der Logarithmentafel nachgeschlagen hat, hat man auch gleich schon eine Rücktransformation durchgeführt, es ist also $\log(Y)$ zurück zu Y transformiert worden. Abbildung 2-2 verdeutlicht das gesagte noch einmal.

2.3. Diskrete Kosinus und diskrete Fourier Transformation

Dieser Teil des Kapitels behandelt nun einige konkreten Fälle - die DCT und die DFT -in denen die Transformation angewendet wird. Auch wenn es sich in dieser Studienarbeit um die eindimensionalen Versionen der Kosinus und Fourier Transformation handelt, werde ich aus Verständnisgründen auch Beispiele und Erklärungen für den zweidimensionalen Fall heranziehen. Man kann sich ein eindimensionales Problem aus einem zweidimensionalen Problem heraussuchen, indem man nur eine Spalte oder eine Zeile betrachtet.

Die diskrete Kosinus Transformation ist, wie schon erwähnt wurde, einer der Grundsteine für das JPEG Bildformat. Man brachte die DCT und die digitale Bildverarbeitung zusammen, als man die Ähnlichkeit von Eigenschaften der DCT mit Eigenschaften von anderen Transformationarten bemerkte, welche unkorrelierte Koeffizienten nach Anwendung der Transformation hervorbringt. Dass Koeffizienten in keiner Wechselbeziehung untereinander stehen, ist wichtig für die Kompression, da man so jeden Koeffizienten einzeln behandeln kann.

2.3.1. Diskrete Kosinus Transformation (DCT)

Während das Verfahren der JPEG-Komprimierung eine Zerlegung eines beliebigen MxN Bildpunkte großen Bildes in einzelne 8x8 Blöcke vorsieht, auf die dann jeweils die DCT angewendet wird (was dem zweidimensionalen Fall der DCT entspricht), wird hingegen bei der eindimensionalen DCT eine Zerlegung in eine 1x8 Matrix durchgeführt.
Es werden also acht willkürlich gewählte Grauwerte betrachtet, wobei diese in einem Bereich von 0 bis 255 liegen. Man spricht hier von einer Farbtiefe von acht Bit, da so 256 verschiedene Farben bzw. Grauwerte dargestellt werden können. Bevor man aber jetzt die acht Werte transformiert, werden sie zunächst einem sogenannten *zero-shift* unterzogen, d.h. von jedem der acht Grauwerte wird 128 subtrahiert und sie liegen somit in einem Bereich von -128 bis 127. Die Verwendung des *zero-shifts* hat folgenden Hintergrund:

Da man 256 verschieden Grauwerte hat, liegt es Nahe, sie als vorzeichenlose acht Bit Zahlen darzustellen. Bei arithmetischen Operationen, wie z.B. der Subtraktion zweier Bilder, können sich negative Werte ergeben. Weil diese negativen Grauwerte nicht dargestellt werden können, treten sie aufgrund der Zweierkomplementdarstellung als hohe positive Werte auf. In der Zweierkomplementdarstellung ist das höchstwertigste Bit auf eins gesetzt. Die Zahl -3 wird mit $-3 \text{ modulo } 256 = 253$ zum Grauwert 253.

Folglich gibt es zwei unterschiedliche Arten Bilder mit Grauwerten darzustellen – eine mit vorzeichenbehafteten Zahlen und eine ohne vorzeichenbehaftete Zahlen. Algorithmen zur Bildbearbeitung müssten also in Lage sein, Operationen mit Bildern durchzuführen, deren Farbwerte alle einheitlich mit oder ohne Vorzeichen versehen sind. Ebenso muss aber auch der Fall abgedeckt sein, bei dem der gemischte Fall auftritt, also in einer Operation werden

Bilder mit und ohne vorzeichenbehafteten Grauwerten verarbeitet.

Dieser unnötige Aufwand wird dadurch umgangen, dass alle Bilder mit Vorzeichen versehen werden. Dies wird durch die Verschiebung des Wertebereichs [0,255] nach [−128,127] durch Subtraktion von 128 bei jedem einzelnen Grauwert bewerkstelligt. Abbildung 2-3 veranschaulicht diesen Sachverhalt.

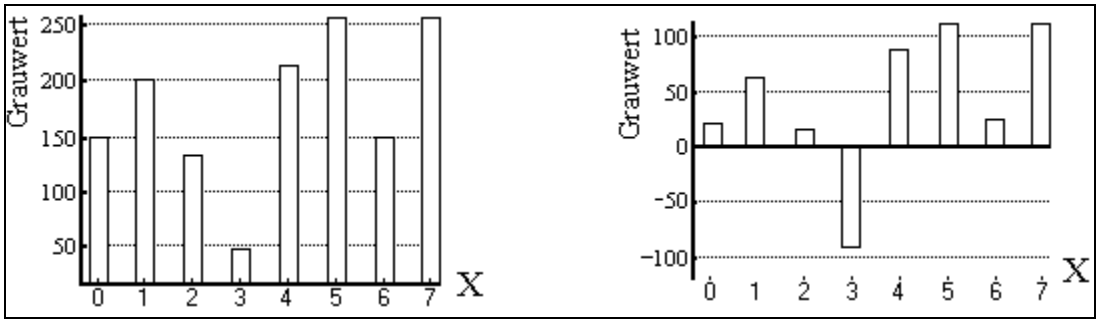

Abbildung 2-3 : *zero-shift* von acht Grauwerten durch Subtraktion 128 von jedem Wert. Das linke Bild zeigt den Originalzustand der Grauwerte, das rechte Bild das Ergebnis.

Jeder der Grauwerte wird nun vom Ortsraum in den Frequenzraum transformiert unter Verwendung dieser Formel für den eindimensionalen Fall :

$$S(u) = \frac{C(u)}{2} * \sum_{x=0}^{7} s(x) * \cos\left(\frac{(2x+1) * u * \pi}{16}\right)$$

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{für } u = 0 \\ 1 & \text{für } u > 0 \end{cases}$$

$s(x)$ 1−Dim Grauwert

$S(u)$ 1−Dim DCT Koeffizient

Formel 2-1 : Die eindimensionale diskrete Kosinus Transformation.

Eine Transformation ist mathematisch zunächst nichts anderes als ein Basiswechsel. In der Anwendung jedoch hat diese Transformation viele Vorteile, die im nächsten Abschnitt dargestellt werden.

2.3.2. Ortsraum und Frequenzraum

In der digitalen Bildbearbeitung werden Bilder als zweidimensionale Matrizen behandelt, d.h. ein Bild ist aus einzelnen Bildpunkten (*picture element*, kurz: *pixel*) zusammengesetzt. Somit ist jedes Bild eine Linearkombination von Bildern, die nur einen einzigen Bildpunkt an der betrachteten Stelle haben. Die Bilder, die linearkombiniert werden, nennt man Basisbilder.

Ihre Matrixdarstellung besteht aus lauter Nullen und nur an einer einzigen Stelle steht eine Eins.

Durch Multiplikation eines Basisbildes mit einer Zahl zwischen 0 und 255, kann der betrachtete Bildpunkt einen jeden der 256 verschiedenen Grauwerte annehmen.
Multipliziert man nun jedes Basisbild mit einem Koeffizienten und addiert anschließend alle erhaltenen Bilder, so lässt sich jedes Bild aus Grautönen darstellen. Abbildung 2-4 zeigt ein 2x2 Bild und wie es aus vier Basisbildern konstruiert wird.

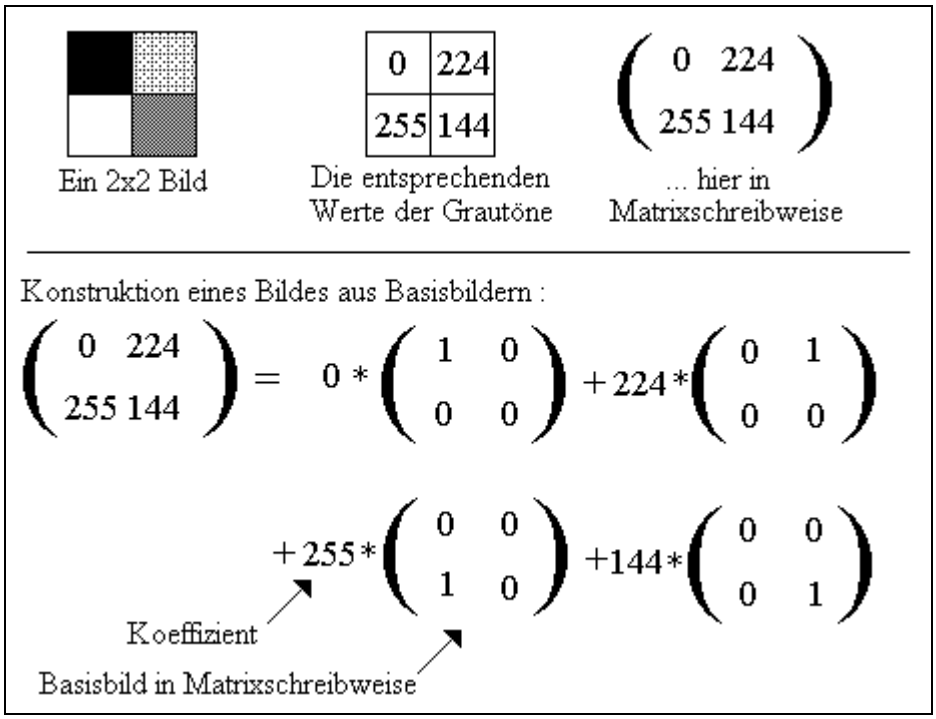

Abbildung 2-4 : Konstruktion eines MxN Bildes (M=N=2) aus seinen M*N (hier: 4) Basisbildern.

Um eine Überführung in eine andere Bilddarstellung – eine Transformation – besser zu verstehen, kann man sich ein aus MxN Bildpunkten bestehendes Bild als einen Punkt in einem MxN Vektorraum vorstellen. Hat ein Bild eine Breite von N Bildpunkten und eine Höhe von M Bildpunkten, so bilden die zugehörigen Basisbilder einen MxN Vektorraum über dem Raum der reellen Zahlen **R**. Bei Änderung des Koordinatensystems werden nur die Koordinaten verändert, das Bild an sich jedoch nicht. Man sieht also dieselbe Information nur aus einem anderen Blickwickel.

Wenden wir uns dem Frequenzraum zu.
Analog zum vorher besprochenen Ortsraum wird hier eine Funktion mit Hilfe von Basisfunktionen dargestellt. Speziell bei der diskreten Kosinus Transformation werden natürlich Kosinusfunktionen benutzt, welche an acht Stellen abgetastet werden - daher *diskrete* Kosinus Transformation. Die Anzahl der acht Abtaststellen ergibt sich aus dem JPEG Standard, da im eindimensionalen Fall der DCT, Bilder in 1x8 Matrizen zerlegt werden. Da das Ziel im eindimensionalen Fall der DCT die Transformation von acht Bildpunkten bzw. deren Grauwerten ist, sind also acht Basisfunktionen vorhanden. Wie man in Abbildung 2-5 sieht, ist die erste Basisfunktion (für Frequenz u=0) eine einfache Konstante. Die zweite Basisfunktion (u=1) ist eine halbe Kosinusschwingung. Die Basisfunktionen alternieren stärker, je größer die Frequenzen werden. Diese Funktionen spannen einen Vektorraum der

Dimension acht über **R** auf. Die Basisfunktionen haben aber auch die Orthogonalitätseigenschaft:

Nimmt man zwei verschiedene Basisfunktionen und multipliziert die korrespondierenden Abtastwerte miteinander, so erhält man, nachdem alle acht Werte aufsummiert wurden, das Ergebnis Null. Wird eine Basisfunktion mit sich selbst multipliziert, so ergibt sich an den einzelnen Abtastpunkten das Quadrat des entsprechenden Abtastwerts. Addiert man nun alle acht Werte (die Quadrate) wieder auf, erhält man eine positive Konstante.

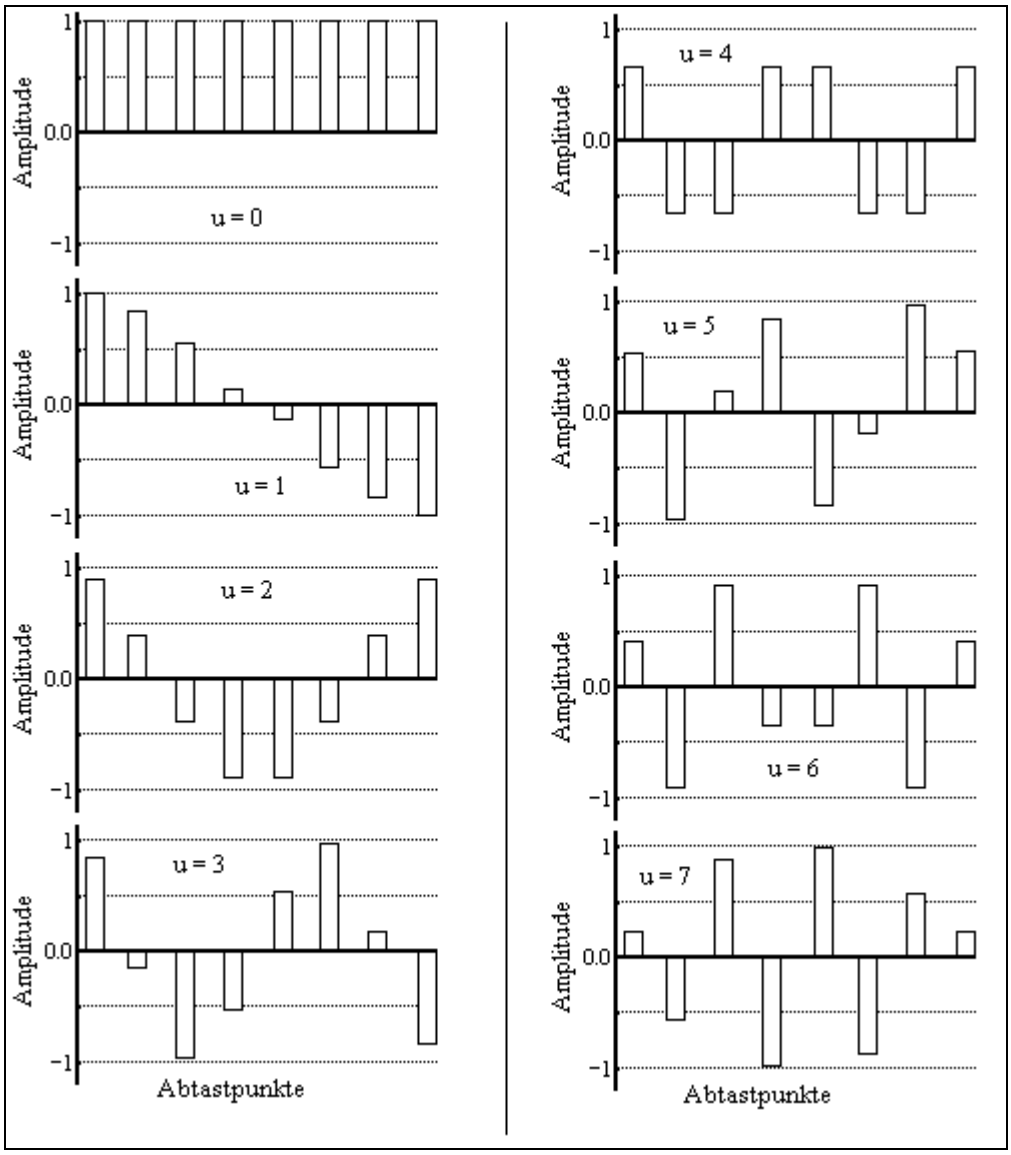

Abbildung 2-5 : Die acht Basiskosinusfunktionen abgetastet an acht Stellen. Für Frequenz u=0 ist die Funktion eine Konstante. Je größer die Frequenz wird, desto stärker alterniert die Funktion.

Da die Kosinusfunktionen eine Basis eines Vektorraums bilden, lassen sich also beliebige Funktionen darstellen, indem man die einzelnen Basisfunktionen mit einem passenden Koeffizienten multipliziert und anschließend die resultierenden Funktionen an den entsprechenden Abtastpunkten addiert.

Wenn man nun die Formel für die eindimensionale DCT auf jeden der acht Grauwerte anwendet, ergeben sich als Ergebnis genau jene acht Koeffizienten, mit den man die korrespondierenden Basisfunktionen multiplizieren muss, um die Frequenzraumrepräsentation der acht Bildpunkte mit Grauwerten zu erhalten.

2.3.3. Diskrete Fourier Transformation (DFT)

Die Fourier Transformation wird als analytisches Hilfsmittel in verschiedenen Bereichen wie Optik, Wahrscheinlichkeitstheorie, Quantenphysik, Antennen oder Signaltheorie angewendet. Zum Beispiel lassen sich diese typischen Anwendungsgebiete der Fourier Transformation nennen :

- In der Stochastik ist die charakteristische Funktion einer Zufallsvariablen die Fouriertransformierte dessen Autokorrelationsfunktion.
- Partielle Differenzialgleichungen lassen sich mit der Fourier Transformation lösen.
- Die Amplitudenverteilung des Lichts auf der vorderen und der hinteren Brennebene einer konvexen Linse folgt der Beziehung der Fourier Transformation.

Die diskrete Fourier Transformation hielt aber erst später Einzug in die genannten Anwendungsgebiete, da sie trotz hoher Rechengeschwindigkeiten von modernen Computern sehr lange Rechenzeiten benötigte. Aber mit der Entwicklung eines Algorithmus zur schnellen Berechnung der diskreten Fourier Transformation wurden diese Rechenzeiten deutlich verkürzt.

Die DCT ist ein Spezialfall der DFT. Es gibt zwei Unterschiede zur DCT. Zum einen wird eine andere Formel benutzt, nämlich :

$$S(u) = \sum_{x=0}^{N-1} s(x) * e^{-i\ 2\pi\ x\ u\ /\ N}$$

N	Anzahl der Abtastwerte (1 – dim Fall z.B. N = 8)
s(x)	1 – Dim Grauwert
S(u)	1 – Dim DFT Koeffizient

Formel 2-2 : Die eindimensionale diskrete Fourier Transformation.

Zum anderen findet bei der DFT eine Transformation von Werten aus dem reellen Zahlenraum in den komplexen Zahlenraum statt. Um nicht mit komplexen Zahlen rechnen zu müssen, kann man unter Zuhilfenahme der Eulerschen Formel

$$e^{ix} = \cos(x) + i * \sin(x)$$
$$e^{-ix} = \cos(x) - i * \sin(x)$$

Formel 2-3 : Die Eulersche Formel.

Formel 2-2 umschreiben :

$$S_{\Re}(u) = \sum_{x=0}^{N-1} s(x) * \cos\left(\frac{2*\pi*x*u}{N}\right) \qquad \text{Realteil}$$

$$S_{\Im}(u) = \sum_{x=0}^{N-1} -s(x) * \sin\left(\frac{2*\pi*x*u}{N}\right) \qquad \text{Imaginärteil}$$

$$\left(S_{\Re}(u), S_{\Im}(u)\right) \qquad 1-\text{Dim DFT Koeffizient im Komplexen}$$

Formel 2-4 : Alternative Schreibweise für Formel 2-2

So gibt es nun acht Koeffizienten für den Realteil und weitere acht Koeffizienten für den Imaginärteil, da jetzt nicht nur acht Basis*kosinus*funktionen vorhanden sind, sondern auch acht Basis*sinus*funktionen.

2.3.4. Rücktransformation

Bei der Rücktransformation werden die mit Hilfe der Transformation gewonnenen Frequenzen wieder zurück in den Ortsraum transformiert. Die eindimensionale inverse DCT erfolgt nach folgender Formel :

$$s(x) = \sum_{u=0}^{7} \frac{C(u)}{2} * S(u) * \cos\left(\frac{(2x+1)*u*\pi}{16}\right)$$

$$C(u) = \begin{cases} \frac{1}{\sqrt{2}} & \text{für } u = 0 \\ 1 & \text{für } u > 0 \end{cases}$$

$s(x)$ 1-Dim Grauwert

$S(u)$ 1-Dim DCT Koeffizient

Formel 2-5 : Die inverse diskrete Kosinus Transformation.

Die eindimensionale inverse DFT wird nach dieser Formel berechnet :

$$S(x) = \frac{1}{N} * \sum_{u=0}^{N-1} S(u) * e^{i 2 \pi x u / N}$$

N Anzahl der Abtastwerte (1-dim Fall z.B. $N = 8$)

$s(x)$ 1-Dim Grauwert

$S(u)$ 1-Dim DFT Koeffizient

Formel 2-6 : Die inverse diskrete Fourier Transformation

bzw.

$$s_{\Re}(x) = \frac{1}{N} * \sum_{u=0}^{N-1} S_{\Re}(u) * \cos\left(\frac{2*\pi*x*u}{N}\right) \qquad \text{Realteil}$$

$$s_{\Im}(x) = \frac{1}{N} * \sum_{u=0}^{N-1} - S_{\Im}(u) * \sin\left(\frac{2*\pi*x*u}{N}\right) \qquad \text{Imaginärteil}$$

Formel 2-7 : Alternative Schreibweise für die inverse DFT.

Da man bei der inversen DFT aufgrund der Eulerschen Formel (Formel 2-3) an dieser Stelle noch nicht die erwarteten Grauwerte erhalten hat, muss man im Fall für acht Abtastpunkte noch acht Differenzen bilden. Läuft x von 0 bis 7 so ergeben sich durch Bildung von

$$\begin{aligned} s_{\Re}(0) - s_{\Im}(0) &= s(0) \\ s_{\Re}(1) - s_{\Im}(1) &= s(1) \\ &\vdots \\ s_{\Re}(7) - s_{\Im}(7) &= s(7) \end{aligned}$$

Formel 2-8 : Differenzbildung führt zu den gewünschten Grauwerten.

die gesuchten Grauwerte.

2.3.5. Quantisierung

Bei der DCT hat es sich eingebürgert, als Nachbearbeitungsschritt eine Quantisierung durchzuführen. Die Quantisierung wird nach der Transformation von Werten aus dem Ortsraum in den Frequenzraum auf die erhaltenen Koeffizienten der Frequenzen angewendet. Man macht sich zunutze, dass das menschliche Auge auf niedrigere Frequenzen empfindlicher reagiert als auf hohe Frequenzen.

Quantisieren bedeutet, jeden erhaltenen Frequenzwert durch eine zunächst beliebige Zahl zu teilen. Dies kann eine Konstante oder ein Eintrag aus einer Tabelle sein. Diese Tabelle wird dann Quantisierungstabelle genannt. Man kann also Frequenzen verschieden stark quantisieren.

Durch das Runden auf ganzzahlige Werte bei der Quantisierung kann es zu Rundungsfehlern kommen, wenn die Quantisierung wieder rückgängig gemacht wird. Die zuvor quantisierten Frequenzen werden wieder mit der Zahl multipliziert, mit der vorher dividiert wurde.

Aufgrund der bereits eingangs genannten Unvollkommenheit des menschlichen Sehvermögens, werden solche Verfälschungen, die mit der Quantisierung einhergehen, bis zu einer bestimmten Schwelle nicht wahrgenommen. Nimmt man stärkere Verfälschungen in Kauf, kann ein höherer Quantisierungsfaktor gewählt werden. Abbildung 2-6 demonstriert den Qualitätsverlust bei steigendem Quantisierungsfaktor. Im linken Originalbild sind 160 verschiedene Graustufen vorhanden. Mit steigendem Quantisierungsfaktor nimmt die Qualität des Bildes ab. Obwohl im zweiten Bild von links mit 80 Grauwerten nur noch die Hälfte der Abstufungen existent sind, erkennt man (fast) keinen Unterschied. Die letzten drei Bilder weisen in dieser Reihenfolge nur noch 10,5 bzw. 2 verschiedene Grautöne auf. Dadurch dass man eine schlechtere Bildqualität zulässt, erreicht man aber eine stärkere Kompression, da immer mehr Amplituden im Frequenzraum auf Null gesetzt werden, je höher der

Quantisierungsfaktor gewählt wird. Dividiert man Amplituden durch einen genügend großen Wert, so ergibt sich nach der Rundung auf die nächste Ganzzahl eine Null.

Abbildung 2-6 : Der Quantisierungsfaktor nimmt von links nach rechts zu. Im linken Originalbild sind 160 Grauabstufungen, in den folgenden Bilder sind es 80, 10, 5 bzw. 2 Grauwerte.

3. Das Applet

Dieses letzte Kapitel beschäftigt sich nun mit dem Applet selbst. Hier möchte ich die Handhabung des Applets angeben. Weiterhin gehe ich auf die Wahl der Auswahlobjekte ein, also die Klärung von Fragen der Art, warum z.B. Textfelder benutzt wurden, um dem Benutzer die Auswahl eines Eingangssignals zu zeigen. Kurz gesagt, möchte ich die Gestaltung der Benutzeroberfläche erläutern.

Bevor ich gleich zum ersten Abschnitt komme, möchte ich noch darauf hinweisen, dass es sich bei meinem in der Programmiersprache JAVA geschriebenen Programm genaugenommen nicht um ein Applet, sondern um ein Frame handelt.

Ein Frame (Rahmen) ist ein Fenster auf der obersten Ebene des Bildschirms, d.h. ein Fenster, das nicht innerhalb eines anderen Fensters enthalten ist. Ein Applet ist hingegen ist immer in das Fenster eines Internet Browsers eingebettet. Es bietet sich jedoch die Möglichkeit, aus einem Applet heraus ein Frame zu erzeugen, so wie ich es auch realisiert habe. Da man also beides – Applet und Frame – in einem Browser betrachten kann, betrachte ich sie als synonym und werde im folgenden immer das Wort Applet zur Beschreibung meines Programms benutzen.

3.1. Gestaltung der Oberfläche

Da mein Programm wie oben gesagt, in einem Internet Browser ausgeführt wird, ist die Benutzung durch viele verschiedene Benutzer gegeben. Alle diese Benutzer haben in irgendeiner Weise mit Computern zu tun, welche wiederum mit verschiedenen Betriebssystemen ausgestattet sind. Diese Betriebssysteme unterscheiden sich nicht nur in Funktionalität, sondern auch optisch.

Ein Ziel meines Applets war es nun eine *intuitive* Benutzeroberfläche zu gestalten, d.h. zum einen, dass sich der Benutzer bei der Arbeit mit meinem Programm „wie zu Hause“ fühlt, zum anderen soll die - im optimalen Fall - selbsterklärend sein.
Natürlich nutzt die beste Benutzeroberfläche nichts, wenn der Benutzer keine Ahnung vom Inhalt des Programms hat, so dass speziell bei meinem Applet ein gewisses Vorwissen in der Thematik Kosinus und Fourier Transformation notwendig ist.

Betrachten wir zunächst einmal das Applet in dem Erscheinungsbild, wie es sich nach dem Laden im Browser zeigt (Abbildung 3-1).
Auf den ersten Blick erkennt man zwei unterschiedliche Bereiche. Auf der linken Seite ist ein Gebiet mit einem Koordinatensystem und zwei Funktionen zu sehen. Dieser Bereich trägt die Bezeichnung *show panel*. Das Wort *Panel* ist ein Begriff aus der Programmiersprache JAVA und bedeutet Grundfläche, d.h. man kann ein Panel dazu benutzen, um darin zu zeichnen, z.B. Funktionen. Ein Panel fungiert aber auch als Container für andere Elemente und kann mit anderen Elementen gefüllt werden.

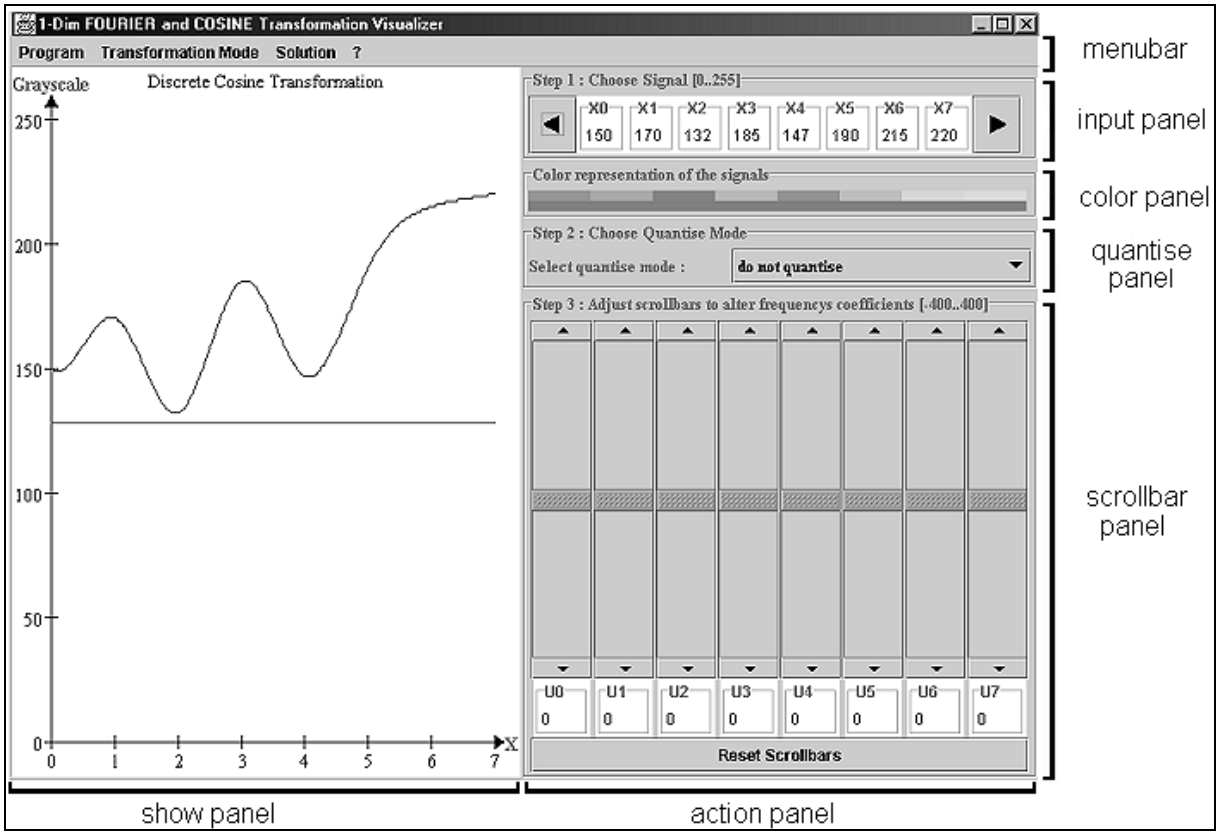

Abbildung 3-1 : Das Applet nach dem Laden im Browser. Der linke Bereich enthält das *show panel*. Das *action panel* rechts daneben besteht aus mehreren anderen Panels. Es wird die Situation der DCT angezeigt (vgl. Schriftzug im oberer Bereich des *show panels*). Im *show panel* ist die Funktion des Eingangssignals aus dem *input panel* dargestellt (die „oberer" Funktion). Die konstante Funktion ist die Repräsentation der rücktransformierten Werte aus dem *scrollbar panel*. Da alle Koeffizienten gleich Null sind, ist die Funktion konstant. Aufgrund des *zero-shifts* bei der DCT (vgl. Kapitel 3) sind die Funktionswerte nicht gleich Null, sondern gleich 128.

Auf der rechten Seite befindet sich ein Bereich mit Textfeldern, Auswahlelementen und Schiebereglern. Diesen Bereich habe ich *action panel* genannt. Die Namensgebung kann dadurch erklärt werden, dass im *action panel* Aktionen getätigt werden, welche sich optisch im *show panel* auswirken.
Das *action panel* wiederum setzt sich aus (zunächst) vier anderen Bereichen zusammen : *input panel*, *color panel*, *quantise panel* und *scrollbar panel*. Welche Funktionen sich hinter diesen einzelnen Bereichen verbergen, werde ich im nächsten Abschnitt diese Kapitels aufdecken.

Dass das Applet eigentlich aus drei anstatt zwei Hauptelementen besteht, lässt sich in Abbildung 3-2 besser erkennen. Diese schematische Darstellung hebt noch die Menuleiste hervor. Die Menüleiste stellt, wie bei anderen Anwendungen auch, einige Grundfunktionen zur Verfügung, die mit dem eigentlichen Geschehen unterhalb der Menüleiste inhaltlich nichts zu tun haben. Zum Beispiel steht die im Menüpunkt **?** (vgl. Abbildung 3-1, Menüleiste) befindliche Option **About** mit der dargestellten Situation, der Visualisierung der DCT, inhaltlich in keinerlei Zusammenhang.

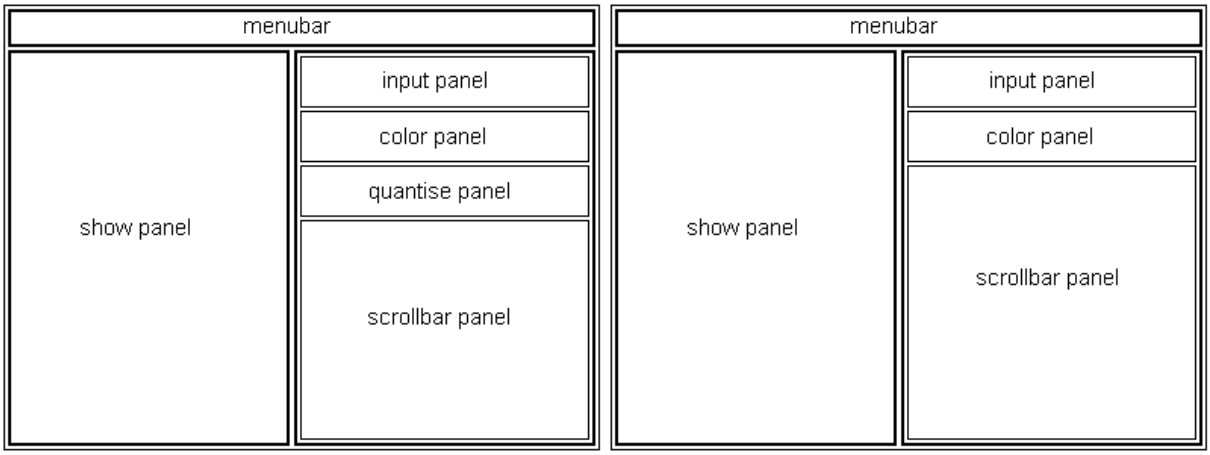

Abbildung 3-2 : Schematische Darstellung der Oberflächenaufteilung. Links die Situation bei der DCT, rechts die Situation bei der DFT.

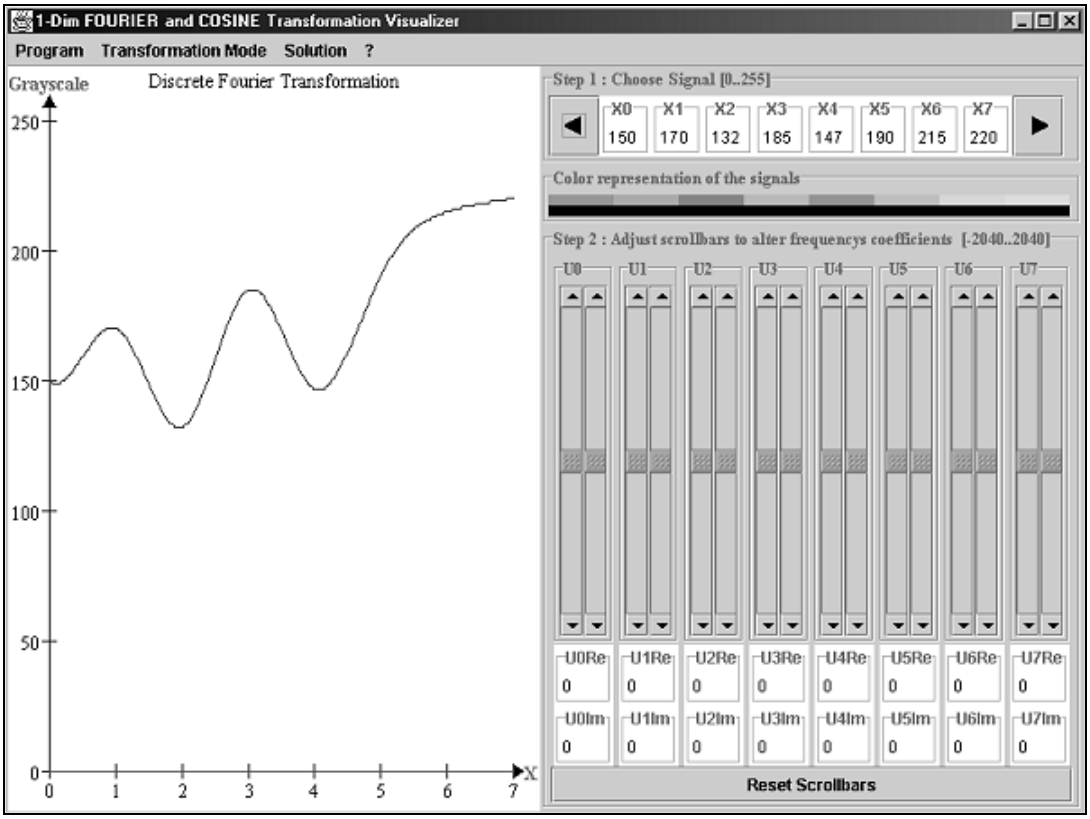

Abbildung 3-3 : Das Applet, nachdem der Transformations Modus gewechselt wurde, indem in der Menüleiste die entsprechende Option gewählt wurde. Die Oberfläche hat sich verändert : das *quantise panel* ist verschwunden und das *scrollbar panel* wurde durch einen andere Version ersetz, welche die doppelte Anzahl an Schiebereglern und Textfeldern hat, da ein Koeffizient nun aus einem Real- und einem Imaginärteil zusammengesetzt ist. Wieder ist im *show panel* die Funktion des Eingangssignals aus dem *input panel* zu sehen. Da bei der DFT kein *zero-shift* vorgenommen wird, sind die Funktionswerte der Funktion der Rücktransformierten gleich Null und ist deshalb genau über die X-Achse gezeichnet.

Weiterhin ist die Menüleiste dazu da, zwischen verschiedenen Modi zu wechseln. So kann man im Menüpunkt **Transformation Mode** entweder die Veranschaulichung von DCT oder von DFT auswählen. Wäre eine derartige Option irgendwo im *action panel* untergebracht, würde das einer intuitiven Benutzerführung widersprechen, da der Benutzer dadurch evtl. verwirrt werden könnte.
Wechselt man den Transformation Modus durch Wahl der **Discrete Fourier Transformation** im **Transformation Mode** Menü, so ändert sich das Erscheinungsbild des *action panels* :
Das *quantise panel* verschwindet und das *scrollbar panel* wird durch eine andere Version mit der doppelten Anzahl an Schiebereglern und Textfeldern ersetzt, da ein Koeffizient nun aus einem Real- und einem Imaginärteil zusammengesetzt ist (vgl. Abbildung 3-3).

Um auf die benutzerfreundliche Gestaltung einer Oberfläche zurückzukommen, möchte ich an einem Beispiel demonstrieren, wie leicht die beabsichtigte Wirkung eines Auswahlelements (Schieberegler, Button etc.) missverstanden werden kann:

Das *input panel* im oberen Bereich des *action panels* hatte nicht immer das jetzige Aussehen, wie in Abbildung 3-4 ersichtlich ist.

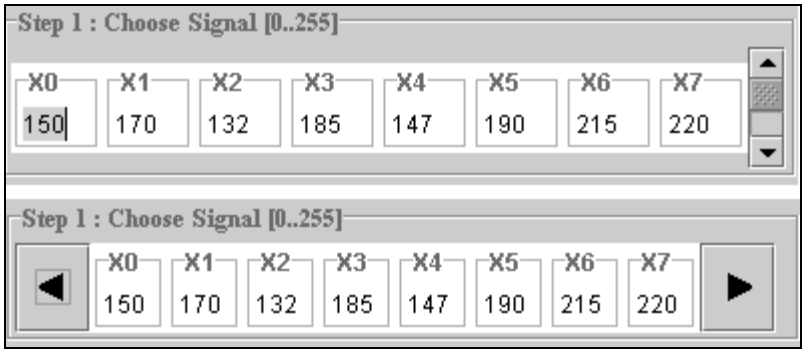

Abbildung 3-4 : Verschiedene Ausgaben des *input panels*. Die obere Version führte zu Missverständnissen bei Benutzern und wurde so durch die untere Variante ersetzt.

Jetzt besteht das *input panel* aus acht Textfeldern, die zwischen zwei Knöpfen (Buttons) angeordnet sind. Man erinnere sich, dass bei der Transformation, sei es DCT oder DFT, in dieser Studienarbeit der eindimensionale Fall betrachtet wird, also acht Grauwerte transformiert werden. In diesen mit X_0 bis X_7 beschrifteten Textfeldern sind nun diese acht Zahlenwerte der entsprechenden Grautöne numerisch dargestellt. Die acht Werte werden auch in das Koordinatensystem des *show panels* eingetragen, so dass der Eindruck einer kontinuierlichen Funktion entsteht, der eigentlich nicht gegeben ist, da es sich bei den Grautönen um acht diskrete Werte handelt. Das Verbinden der Werte soll nur der besseren Übersicht dienen. Mit den beiden Buttons (vgl. Abbildung 3-4 untere Variante) können verschiedene voreingestellte Eingangssignale, jedes bestehend aus acht Ziffern, durchgeschaltet werden (genauso als würde man an einer Fernbedienung für einen Fernseher, die Tasten betätigen, mit denen man einen Kanal vor- bzw. zurückschaltet). Also steht das beim Start des Applets angezeigte Eingangssignal, das aus den Werten (150, 170, 132, 185, 147, 190, 215, 220) besteht, nicht mit dem „darauffolgenden“ Signal (165, 185, 130, 190, 175, 196, 223, 199) in Zusammenhang. Das „darauffolgende“ Signal wird durch Betätigung des Buttons rechts der Textfelder erreicht.

Die ursprüngliche Implementation sah aber anstatt der beiden Knöpfe einen Schieberegler rechts von den Textfeldern vor, wie in Abbildung 3-4 in der oberen Ausgabe

des *input panels* zu sehen ist. Die mir offensichtlich erscheinende Intention des Schiebereglers, die gleiche Funktion zu haben wie sie jetzt von den beiden Buttons erfüllt wird, wurde durch Tests an Versuchspersonen, von diesem zunächst völlig falsch interpretiert:

Durch die Möglichkeit einen Schieberegler an dessen Haltefläche festzuhalten und herauf- und herunterbewegen zu können, meinten einige Benutzer, das sie sich durch ein nichtdiskretes Kontinuum bewegen (vgl. Abbildung 3-5), das soll heißen, dass sich diese Benutzer vorgestellt haben, dass das *show panel* ein Ausschnitt einer Funktion zeigt, die links und rechts vom aktuell dargestellten Abschnitt noch weiterführt.

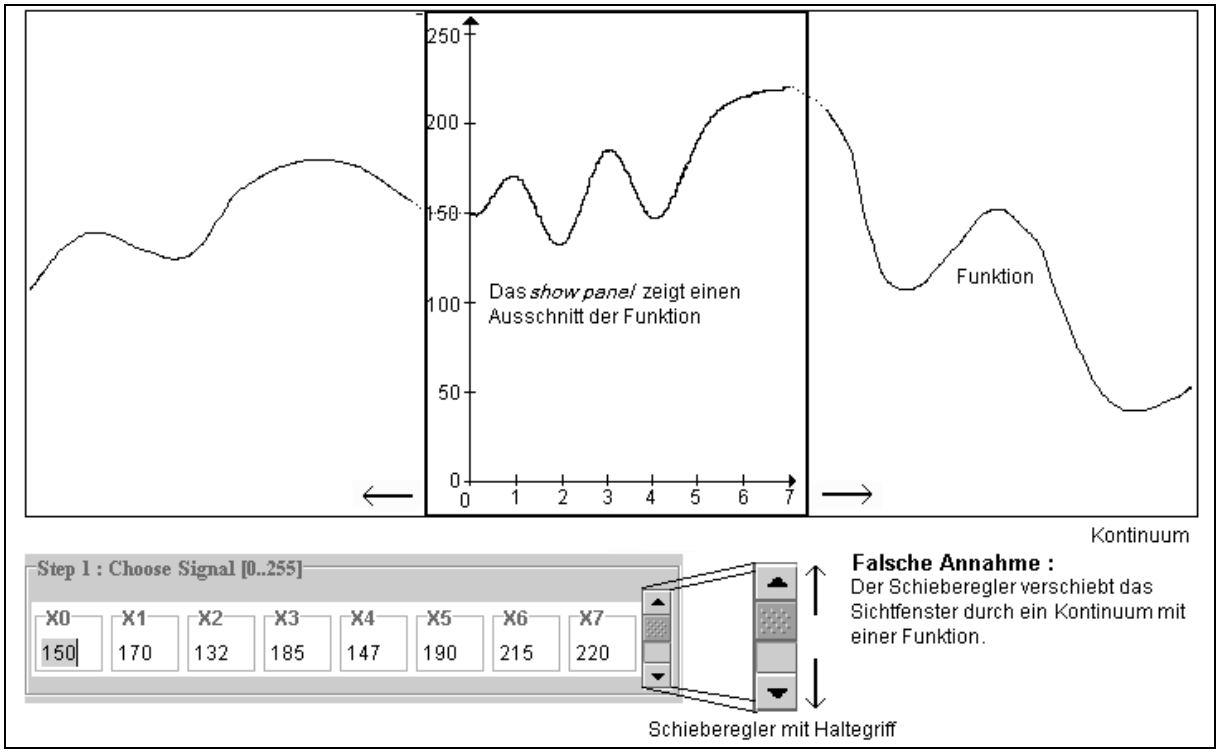

Abbildung 3-5 : Die falsche Annahme, dass der Schieberegler in der veralteten Version des *input panels* ein Sichtfenster über einer Funktion verschiebt. Der im Sichtfenster angezeigte Ausschnitt der Funktion ist dann im *show panel* zu sehen.

Dieses tiefgreifende Missverständnis wurde nur durch die falsche Wahl eines einzigen Auswahlelements heraufbeschworen. Jetzt versteht der Leser sicherlich, was es für Überlegungen bedarf, eine komplett eingerichtete Benutzeroberfläche zu gestalten.

Weiterhin lässt sich noch aufführen, dass nicht nur die Auswahl einzelner Auswahlelemente (Buttons, Schieberegler etc.) für das Verständnis der Funktion der Benutzeroberfläche ausschlaggebend ist, sondern auch das gesamte Erscheinungsbild. Wie bereits am Anfang dieses Abschnitts erwähnt wurde, arbeiten Benutzer mit den verschiedensten Betriebssystemen, die bekanntesten sind MS Windows und Unix bzw. Linux.
Dies konnte man gut beobachten, wenn Benutzer, welche das Unix System gewöhnt sind, versuchen, das Applet zu benutzen : Sie wirkten zwar nicht hilflos, aber doch sehr unsicher beim Navigieren durch die verschiedenen Menüs oder beim schlichten Betätigen einiger Auswahlelemente. Das kann man sich dadurch erklären, dass bei der Konstruktion von JAVA Programmen, der sogenannte Look-and-Feel (LaF) Manager das Aussehen der

Bedienelemente auf einen Standard – genannt Metal - setzt, welcher sich deutlich von der Darstellung einer Benutzeroberfläche von Betriebssystemen unterscheidet.
JAVA bietet jedoch die Möglichkeit, diesem LaF Manager mitzuteilen, welches Aussehen gewünscht ist. Also wurde von mir eine Option vorgesehen, zur Laufzeit des Applets die Oberflächendarstellung den Wünschen des Benutzers anzupassen. Diese Option ist in der Menüleiste unter **Program** zu finden (vgl. Abbildung 3-6). Im Anhang sind einige Darstellungsarten der Oberfläche präsentiert.

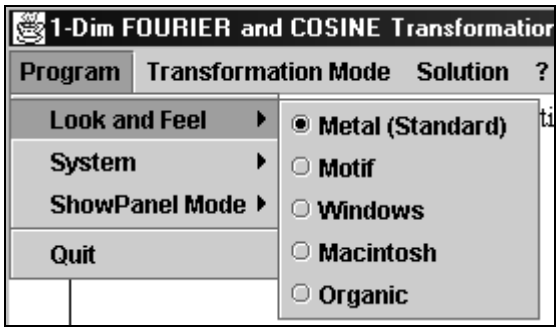

Abbildung 3-6 : Das **Program** Menü mit geöffnetem **Look and Feel** Untermenü. Es stehen mehrere verschiedene Möglichkeiten zu Verfügung, das Aussehen der Benutzeroberfläche anzupassen.

Zum Ende dieses Abschnitt möchte ich noch auf die Auswahl einiger anderer Auswahlelemente eingehen.
Im *input panel* sind acht Textfelder mit den acht Grauwerten eines Eingangsignals zu sehen. Es stellt sich natürlich die Frage, warum Textfelder zum Anzeigen der Werte der Grautöne gewählt wurden. Ich hatte mir mehrere Alternativen zu den Textfeldern überlegt.

Eine Möglichkeit bestand darin, genau wie im *quantise panel* eine ComboBox zu benutzen (vgl. Abbildung 3-7) und so dem Benutzer eine Auswahl von Eingangssignalen anzubieten.

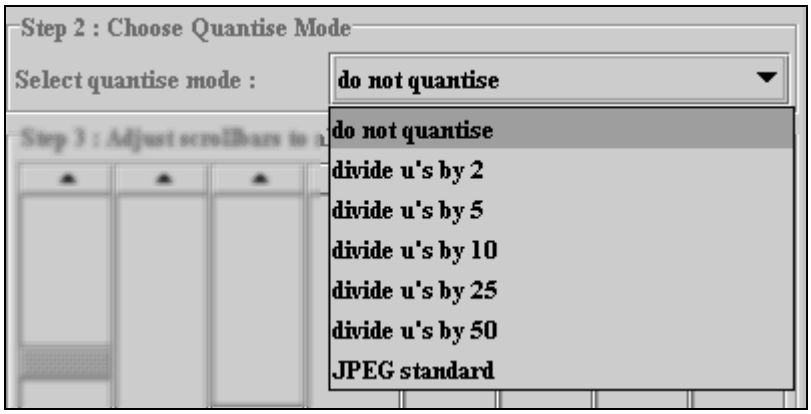

Abbildung 3-7 : Die geöffnete ComboBox aus dem *quantise panel*. Eine ComboBox hätte als alternative Darstellungsmöglichkeit für die Auswahl von Eingangssignalen im *input panel* dienen können.

Diese Alternative hat den Vorteil, sehr leicht realisiert und programmiert zu werden, ebenso sind die gerade eingestellten Werte immer gut sichtbar. Allerdings konnten die Variablenbezeichnungen der Werte der Grautöne X_0 bis X_7 nicht vernünftig (d.h. gut

erkenntlich und damit auch gut verständlich) in der ComboBox untergebracht werden , sodass eine Verwendung von Bezeichnung und Wert einer Variablen in einer ComboBox nicht sinnvoll zu realisieren war.
Eine zweite Alternative bot eine Lösung, indem man acht Labels (ein Bereich, der Text anzeigen kann) benutzt, in die man die entsprechenden Werte des Eingangssignals einträgt. Um das Label herum kann ein dekorativer Rahmen gezogen werden, in dem man auch einen Text schreiben kann, also auch die Bezeichnung der Variablen X_0 bis X_7. Das Aussehen ähnelt sehr der Darstellung des *input panels* mit den Textfeldern, z.B. wie in Abbildung 4-4.
Dass ein Label auf keine Benutzereingaben reagiert, stellte sich aber wieder als Nachteil heraus, weil ich dem Benutzer nicht nur vordefinierte Eingangssignale zur Verfügung stellen wollte, sondern dem Benutzer sollte auch die Möglichkeit zuteil werden, selbstdefinierte Signale zu erstellen bzw. voreingestellte Werte zu manipulieren. Das lässt sich meiner Meinung nach nur mit Textfeldern bewerkstelligen:
Einerseits eignen sich Textfelder zum Darstellen von Informationen und, genau wie bei den Labels, zur Bezeichnung der Informationen durch Benutzen eines Rahmens samt Textzeile, andererseits können in Textfeldern vorhandene Informationen manipuliert oder selbst erstellt werden.

Für die im *quantise panel* befindliche ComboBox (vgl. Abbildung 3-7) wollte ich zuerst auch Textfelder verwenden, um die Ziffern anzuzeigen mit denen die transformierten Eingangssignale dividiert werden. Das hieße aber, dass acht Textfelder zu sehen wären, welche achtmal dieselbe Zahl anzeigen, für den Fall, dass alle acht Werte eines Eingangssignals durch eine Zahl geteilt werden und nicht Zahlen aus einer Quantisierungtabelle benutzt werden.
Da das mehrfache Anzeigen einer Zahl unsinnig ist, habe ich eine ComboBox benutzt, in der man Einstellungen wie z.B. den Standardfall <do not quantise> oder Einstellungen wie <divide u's by 2> auswählen kann. Es wird also in Textform der Quantisierungsgrad zur Auswahl gestellt und angezeigt.

Zur Wahl der Schieberegler zum Ändern der Koeffizienten der Frequenzen, kann man eigentlich nicht viel sagen. Zum einen, weil die Benutzung von Schiebereglern ausdrücklich in der Aufgabendefinition (vgl. Kapitel 1) gefordert wurde, zum anderen offerieren Schieberegler gerade die Möglichkeit „herumzuspielen", indem man durch das Ziehen der Haltefläche der Regler relativ schnell einen gewünschten Wert einstellen kann. Eventuell nötige Feineinstellungen lassen sich durch die am oberen und unteren Ende angebrachten, in den Schiebregler integrierten Knöpfe vornehmen.

Im nächsten Abschnitt wird die Benutzerführung beschrieben, indem die Funktion der verschiedenen Panels (*input panel*, *color panel* etc.) untersucht wird.

3.2. Benutzerführung

Zunächst erfolgt eine grobe Darstellung der Funktionsweise des Applets. Im Anschluss werden die Panels unter die Lupe genommen.

3.2.1. Grober Überblick

Startet man das Applet, wird die Situation, die in Abbildung 3-1 zu sehen ist, dargestellt. Es sind das *show panel* und das aus mehreren anderen Panels zusammengesetzte *action panel* zu erkennen. Oben im *action panel* ist das *input panel* zu sehen. Im *input panel* stellt man die Eingangssignale ein. Die acht Zahlenwerte werden im *color panel*, in der oberen der beiden Zeilen, in entsprechenden Grautönen dargestellt. Ebenso werden die acht Werte X_0 bis X_7 in das Koordinatensystem des *show panels* eingetragen und miteinander verbunden, so dass der Eindruck einer kontinuierlichen Funktion entsteht, der nur der besseren Visualisierung dienen soll. Der Benutzer soll nun die angezeigte Funktion des Eingangssignals nachbauen, indem die Einstellung der Schieberegler im *scrollbar panel*, also im Frequenzraum verändert wird. Ob nun als Transformationsmethode die diskrete Kosinus Transformation oder die diskrete Fourier Transformation verwendet wird ist nicht so wichtig, da das Prinzip das gleiche ist.
Mit jedem Schieberegler kann ein Koeffizient einer Basisfunktionen eingestellt werden (siehe ‚Das *scrollbar panel*' weiter unten). Durch Einstellen jedes Schiebereglers kann die Funktion jedes Eingangssignal exakt durch die Funktion des Ausgangssignals im *show panel* nachgebildet werden, sofern bei der DCT keine Quantisierung angewendet wird.

3.2.2. Die Panels und ihre Funktion

Das *input panel*

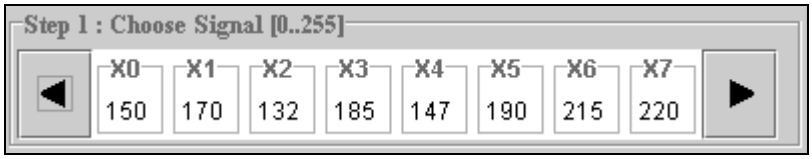

Abbildung 3-8 : Das input panel. Hier werden die Eingangssignale für die Transformation durch Drücken der Buttons ausgewählt. Ebenso können einzelne Textfelder mit einem benutzerdefiniertem Wert versehen werden.

Das *input panel* (Abbildung 3-8) besteht aus acht Textfeldern, wovon jedes mit einem dekorativen Rahmen umgeben ist, in dem eine Bezeichnung bzw. Beschreibung in Form einer Textzeile zu sehen ist. Neben der Reihe der acht Textfelder sind zwei Köpfe, je einer links und rechts der Textfelder, angebracht. Der gesamte Inhalt des Panels ist mit einem weiteren Rahmen umgeben.

Durch Betätigen der beiden Knöpfe kann aus mehreren voreingestellten Eingangssignalen ausgewählt werden. Diese Eingangssignale bestehen aus acht Zahlen, welche in einem

Bereich von 0 bis 255 liegen. Diese Zahlen X_0 bis X_7 geben wiederum die numerische Repräsentation der Grauwerte an, welche mit Hilfe von DCT oder DFT transformiert werden sollen. Es ist auch möglich, durch Benutzen der Tastatur oder Maus, die Einträge der Textfelder zu ändern, um einen eigenen Wert einzutragen. Bestätigt man die Eingabe durch Drücken der Return bzw. Enter Taste, werden eventuell auftretende falsche Eingaben durch eine Fehlerbehandlungsroutine abgefangen. Es sind keine Eingaben von Zahlen ausserhalb des Intervalls [0 , 255] zulässig. Nicht numerische Eingaben stellen ebenso eine Fehleingabe dar. Möchte man ein komplett anderes Eingangssignal eingeben, wählt man das Textfeld mit der Bezeichnung X_0 an (z.B. mit der Maus), gibt den gewünschten Wert ein und gelangt durch Betätigen der TAB Taste zum nächsten Textfeld. So kann ein eigenes Eingangssignal relativ schnell eingestellt werden.
Ändert sich das Eingangssignal, sei es durch Benutzung eines der beiden Buttons oder durch die manuelle Eingabe einer Zahl in ein Textfeld, so ändert sich sowohl der Plot der Funktion des Eingangssignals, als auch die im *color panel* angezeigten Farben.

Das *color panel*

Abbildung 3-9 : Das *color panel*. Die obere Zeile gibt die Grauwertdarstellung des im *input panels* eingestellten Eingangssignals wieder. Der linke Farbbalken ist der Grauton, der durch die Zahl im Textfeld X_0 eingestellt werden kann. Die untere Zeile ist die Grauwertrepräsentation des Ausgangssignals, das der Benutzer im *scrollbar panel* selbst einstellen kann.

Das *color panel* (Abbildung 3-9) besteht aus zwei Farbzeilen, jede bestehend aus acht Farbbalken. Das Panel ist mit einem Rahmen umgeben, der eine Bezeichnung enthält.

Die Funktion des *color panels* ist es, dem Benutzer eine Farbrepräsentation von Eingangs- als auch Ausgangssignal (siehe ‚Das *scrollbar panel*' weiter unten) zu geben. Die obere Zeile des Panels zeigt acht Farbbalken, deren Grauwerte mit den Zahlen des Eingangssignals übereinstimmen, das gerade im *input panel* eingestellt ist. Eine dunkle Farbe entspricht einer kleineren Zahl (minimal Null), also wenn z.B. im Textfeld X_0 eine Null steht, zeigt das *color panel* in der oberen Zeile ganz links einen schwarzen Balken. Der maximal einstellbare Wert ist 255, welcher der Farbe Weiß entspricht. Ist im *input panel* abwechselnd 0 und 255 in den Textfeldern eingestellt, spiegelt sich das im *color panel* wieder, indem abwechselnd schwarze und weiße Balken in der oberen Zeile zu sehen sind.
Die untere Zeile entspricht den acht Farben des Ausgangssignals, das der Benutzer mit den Schiebereglern im *scrollbar panel* selber einstellt.
Wird als Transformationsart die DCT verwendet und stehen alle Schieberegler in Mittelposition, besteht die untere Zeile aus acht Farbbalken mit demselben Grauton (Abbildung 3-9). Da die Subtraktion von 128 von jedem Wert eines Eingangssignals (zero-shift, siehe Kapitel 3) wieder rückgängig gemacht werden muss, wird zu jedem Wert des Ausgangssignals 128 hinzu addiert, sodass die untere Zeile einen einheitlichen Grauton mit Wert 128 hat. Genauer :
Nach Start des Applets sind im *scrollbar panel* alle Koeffizienten gleich Null, diese werden rücktransformiert und es ergibt sich ein Ausgangssignal bestehend aus acht Nullen, dann wird

der zero-shift rückgängig gemacht und das Ausgangssignal ist nun aus acht Zahlen gleich 128 zusammengesetzt.
Ist aber die DFT eingestellt, so entfällt der zero-shift vor und nach der Transformation und die acht Werte des Ausgangssignals, das sich nach der Rücktransformation von Koeffizienten gleich Null ergibt, bleiben Null, so dass die untere Zeile des *color panels* aus acht schwarzen Balken besteht.

Das *quantise panel*

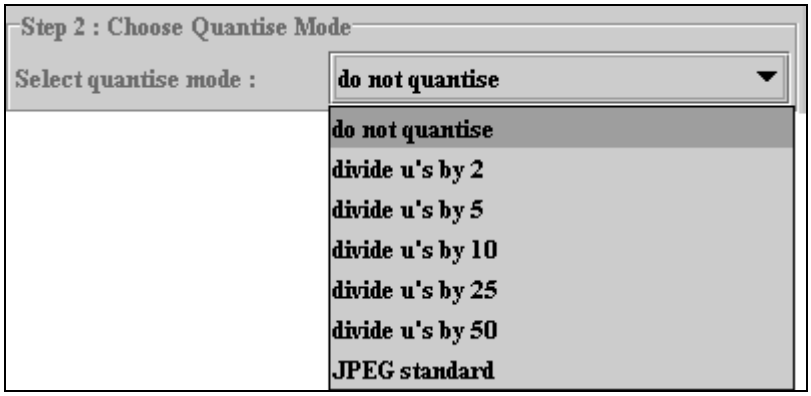

Abbildung 3-10 : Das quantise panel. Die ComboBox ist geöffnet und die Wahlmöglichkeiten werden angezeigt. Von oben nach unten nimmt der Quantisierungsfaktor zu. Bei der **JPEG standard** – Einstellung nimmt eine Sonderstellung ein : Die transformierten Werte der korrespondierenden Stellen des im input panel angezeigten Eingangssignals werden gemäß JPEG-Standard durch den Vektor (16,11,10,16,24,40,51,61) dividiert.

Das *quantise panel* (Abbildung 3-10) wird angezeigt, wenn die DCT verwendet wird. Es besteht aus nur zwei Komponenten : einem Label und einer ComboBox. Um das Panel ist wieder ein Rahmen gezogen worden.

Um den Quantisierungsfaktor bei Verwendung der diskreten Kosinus Transformation einzustellen, wird eine ComboBox verwendet, die mehrere Möglichkeiten zur Auswahl stellt. In der Standardeinstellung <do not quantise> wird nicht quantisiert. Die <divide u's by n> Optionen stehen für die möglichen Quantisierungsfaktoren, z.B. n=2 oder n=10. Bei Verwendung einer dieser Optionen werden die transformierten Werte des Eingangssignals X_0 bis X_7 durch denselben Wert geteilt. Nur die letzte Auswahlmöglichkeit der ComboBox **JPEG standard** bietet die Gelegenheit die Werte durch Einträge einer Quantisierungstabelle zu dividieren. Bei diese Einstellung werden die transformierten Werte von X_0 bis X_7 durch den entsprechenden Wert von (16,11,10,16,24,40,51,61) dividiert, d.h. der transformierte Wert von X_0 (der Koeffizient einer Frequenz!) wird durch 16 geteilt, der transformierte Wert von X_1 wird durch 11 geteilt usw. .
Bei der Rekonstruktion des Ausgangssignals aus den Werten der Einstellung der Schieberegler im *show panel* wird der Vorgang umgekehrt ausgeführt :
Ein Wert, der der Einstellung eines Schiebereglers im *show panel* entspricht (dieser Wert ist in den Textfeldern unter diesem Schieberegler angezeigt), wird erst mit dem entsprechenden Quantisierungsfaktor multipliziert und anschließend rücktransformiert. Wird dies für alle acht Werte der Schieberegler durchgeführt (bzw. 16 Schieberegler bei Verwendung der DFT), so ergibt sich das Ausgangssignal.

Wie das in der Praxis aussieht wird bei der folgenden Erklärung des *scrollbar panels* verdeutlicht.

Das *scrollbar panel*

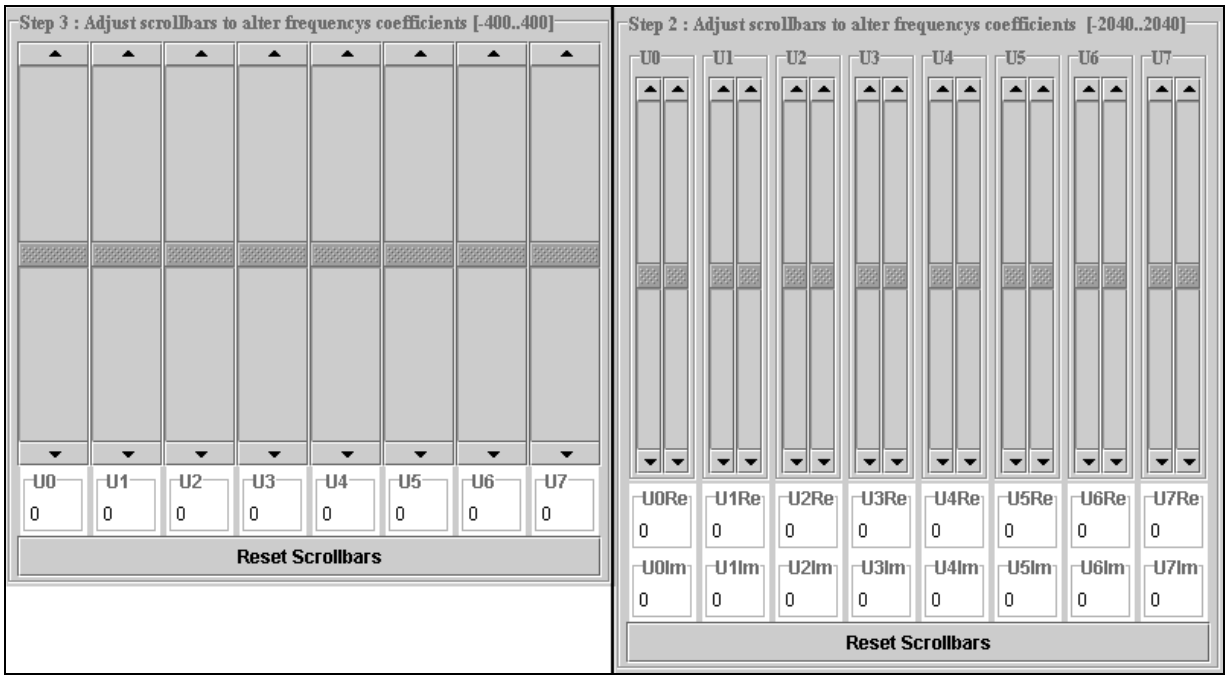

Abbildung 3-11 : Das *scrollbar panel*. Links die Darstellung bei Verwendung der DCT, rechts das Aussehen, wenn die DFT benutzt wird. Die Schieberegler, deren Wert in den Textfeldern angezeigt wird, können zur Einstellung der Koeffizienten benutzt werden. Andersherum werden die Positionen der Regler angepasst, wenn der Eintrag von Textfeldern verändert wird.

Das Aussehen des *scrollbar panels* (Abbildung 3-11) ist abhängig von der verwendeten Transformationsmethode. Bei Verwendung der DCT (Abbildung 3-11 linke Version), besteht das Panel aus acht Schiebereglern mit vertikaler Ausrichtung. Je ein Textfeld mit einer Bezeichnung befindet sich unter einem der Schieberegler. Wird die DFT verwendet (Abbildung 3-11 rechte Version), so sind 16 Schieberegler vorhanden. Je zwei Regler sind von einem Rahmen mit einer Bezeichnung umgeben. Unterhalb jeder Zweiergruppierung von Schiebereglern, sind zwei Textfelder angebracht. Beide Varianten des *scrollbar panels* haben gemeinsam, dass sich unter den Textfelder ein Button befindet, der die gesamte Breite des Panels einnimmt.
Bewegt man einen Schieberegler, wird der Wert, der der Position des Reglers entspricht, im Textfeld unterhalb des Schiebereglers numerisch angezeigt. Die Position eines Schiebereglers passt sich der Eingabe eines zulässigen Werts in ein zugehöriges Textfeld an. Eine zulässige Eingabe ist eine Zahl, die innerhalb des Intervalls liegt, dass in der Textzeile oberhalb der Schieberegler angezeigt wird.
Durch Betätigung des Buttons wird in alle Textfelder der Wert Null eingetragen und die Schieberegler werden zurück in Mittelposition gebracht.

Im *scrollbar panel* lässt sich das bereits erwähnte Ausgangssignal einstellen. Das Ausgangssignal, dessen kontinuierliche Funktion im *show panel* gezeichnet wird, ergibt sich

aus der Rücktransformation der Werte, die den Einstellungen der Schieberegler entsprechen (numerisch angezeigt in den Textfeldern). Daraus erkennt man, dass man eigentlich keinen direkten Einfluss auf den Plott des Ausgangssignals hat, da mit den Schiebereglern nur Koeffizienten von Basisfunktionen (vgl. Kapitel 2) manipuliert werden können und somit bei Veränderung eines Koeffizienten der gesamte Funktionsverlauf des Ausgangssignals verändert wird.

Im Fall der Verwendung der Transformationsmethode der diskreten Kosinus Transformation, steht der äußerst linke Schiebregler für die Frequenz u=0, der Schieberegler rechts daneben steht für die Frequenz u=1 usw.. Da es sich bei der diskreten Fourier Transformation, um eine Transformation in den komplexen Zahlenraum handelt, setzt sich der zu einer Frequenz gehörende Koeffizient, aus einem Real- und einem Imaginärteil zusammen. Deshalb sind zwei Schieberegler für die Einstellung, von z.B. Frequenz u=0, zuständig: der linke Schieberegler aus der Gruppierung, die durch einen Rahmen mit der Beschriftung „U0“ angedeutet ist, wird zum Verändern des Realteils des Koeffizienten benutzt. Analog wird der rechte Schieberegler aus der Gruppierung zum Einstellen des Imaginärteils verwendet.

Was passiert, wenn man Schieberegler bewegt?
Um einen gewissen Wiedererkennungswert zu Abbildung 2-5 zu erreichen, sollte man zunächst nur einen einzigen Schieberegler benutzen:
Verschiebt man, bei Verwendung der DCT, den linken Schieberegler für „U0“ auf und ab, so bewegt sich die konstante Funktion im *show panel* ebenso auf- und ab. Verwendet man hingegen den Schieberegler „U1“ für Frequenz u=1, erkennt man, dass der Plott des Ausgangssignals einer halben Kosinusschwingung ähnelt. Ändert man also die Position von einzelnen anderen Schiebereglern, so sieht man die Ähnlichkeit des Funktionsverlaufs, der sich aus der Rücktransformation der Koeffizienten (die jeweils alle - bis auf einen - gleich Null sind) ergibt, zu den in Abbildung 2-5 angezeigten Basiskosinusfunktionen. Diese „verzerrten“ Basisfunktionen, die bei Betätigung eines Schiebereglers im *show panel* dargestellt werden, sind aber nichts anderes, als mit einem Koeffizient multiplizierte Basisfunktionen. Abbildung 3-12 zeigt den Plott des Ausgangssignals, wenn nur Schieberegler für Frequenz u=0 bzw. Frequenz u=1 bewegt wird.

Bei Veränderung der Einstellung mehrerer Schieberegler, werden die Funktionswerte der Funktionen, die sich aus der Multiplikation der Koeffizienten mit den korrespondierenden Basisfunktionen ergeben, addiert. Bewegt man zunächst Schieberegler „U1“, ergibt sich wieder der Plott des Ausgangssignals, der einer halben Kosinusschwingung ähnelt. Wird anschließend noch der Schieberegler für Frequenz u=0 bewegt, so kann man offenbar die „Höhe“ der halben Kosinusschwingung verstellen (Abbildung 3-13).

Man sieht also, dass man durch geschickte Anwendung der Schieberegler, den Plott des Ausgangssignals, durch Addition der verschiedenen Basisfunktionen, „in Form“ bringen kann. Die Stärke, in der sich eine Basisfunktion im Plott wiederspiegeln soll, lässt sich durch Vergrößern oder Verkleinern der zu einer Frequenz gehörenden Koeffizienten bestimmen.

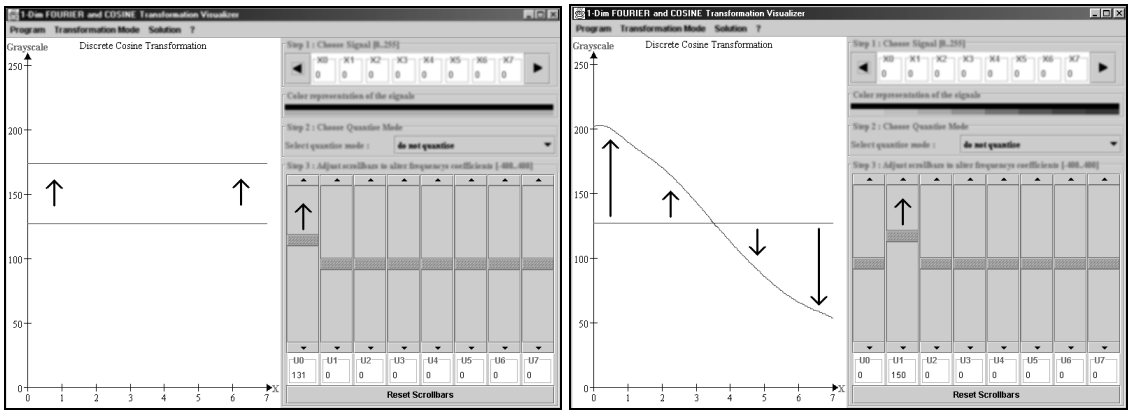

Abbildung 3-12 : Beispiele für das Bewegen eines *einzigen* Schiebereglers. Wird nur jeweils ein einziger Schieberegler bewegt, erkennt man die Basisfunktionen aus Abbildung 2-5 wieder. Diese sind nun aber mit einem Koeffizienten multipliziert, welcher sich durch das Verstellen eines Reglers bestimmen lässt.

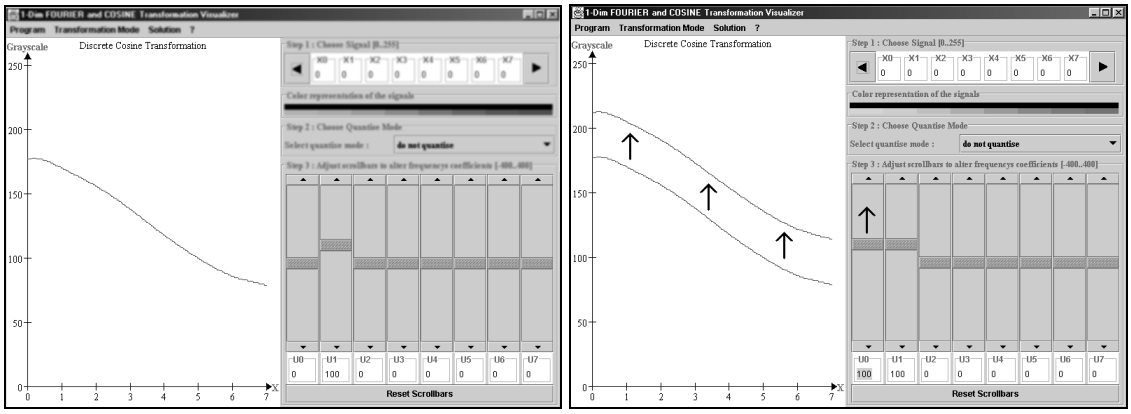

Abbildung 3-13 : Ein Beispiel für das Verschieben von zwei Schiebereglern. Das linke Bild ergibt sich durch Verschieben von Regler „U1“ auf einen Wert von 100. Anschließend wird Schieberegler „U0“ ebenfalls auf einen Wert von 100 verschoben, so dass sich das rechte Bild ergibt. Die beiden Basisfunktionen, deren Koeffizient sich mit den Schiebereglern „U0“ und „U1“ einstellen lässt, addieren sich an ihren Funktionswerten.

3.3. Nachbildung eines Eingangssignals

In diesem letzten Abschnitt des Kapitels und der vorliegenden Studienarbeit, möchte ich noch kurz zeigen, wie man ein Eingangssignal nachbaut, indem die passenden Schieberegler auf bestimmte Werte eingestellt werden. Es wird das nach Start des Programms voreingestellte Eingangssignal (150,170,132,185,147,190,215,220) und die DCT verwendet.

Wie in der Ausgangssituation im linken oberen Bild der Abbildung 3-14 ersichtlich ist, liegen alle Funktionswerte des Plotts des Eingangssignals über den Funktionswerten des Ausgangssignals. Es ist also Schieberegler für Frequenz u=0 nach oben zu verschieben, um den Koeffizienten zu vergrößern. Man verschiebt den Regler solange, bis im zugehörigen

Textfeld der Wert 136 erreicht ist (oder man gibt den Wert direkt ins Textfeld ein). Das Ergebnis wird in Abbildung 3-14 im rechten oberen Bild gezeigt.

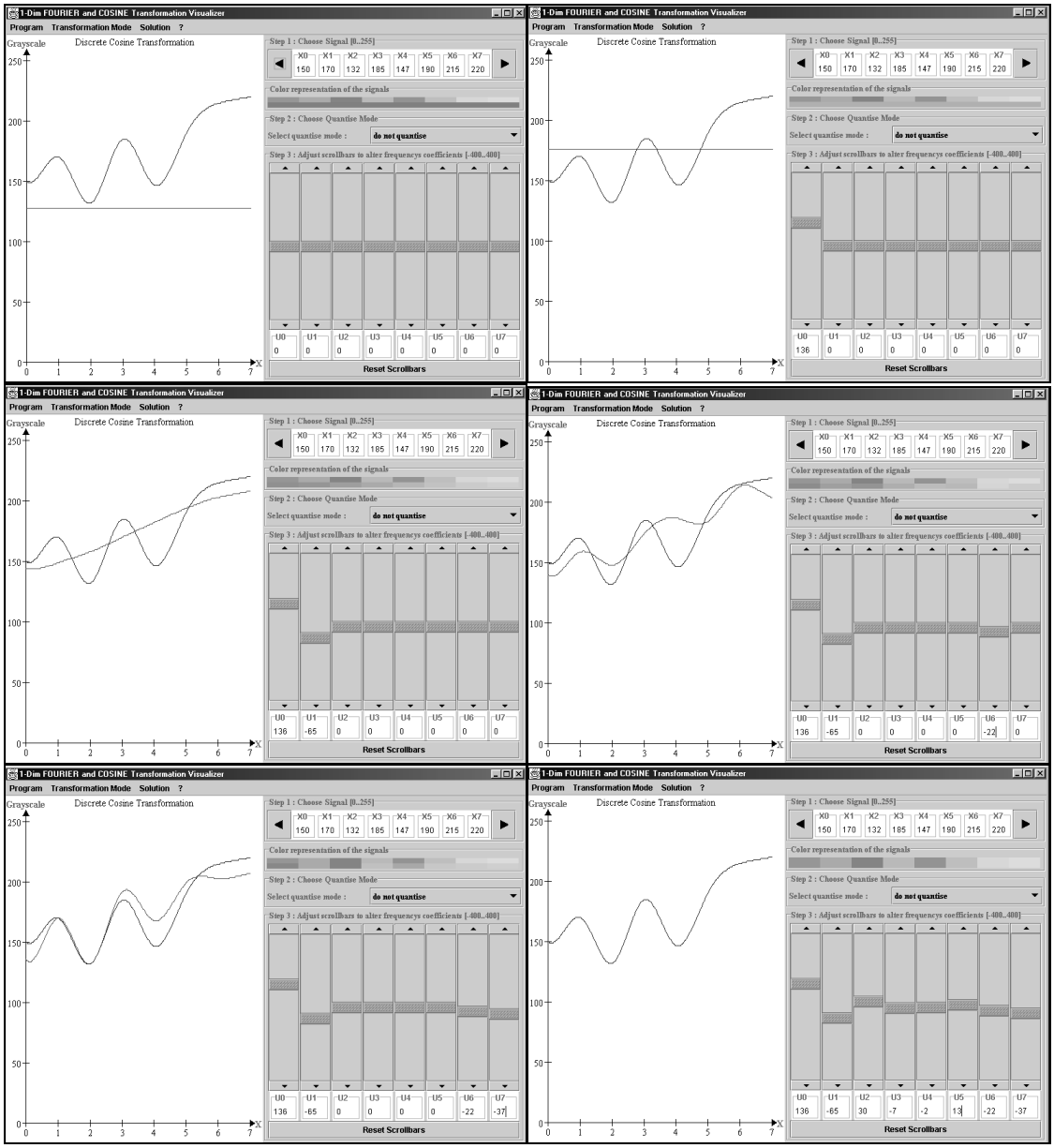

Abbildung 3-14 : Schrittweise Nachbildung des Ausgangssignals (150,170,132,185,147,190,215,220) unter Verwendung der DCT. Das Bild oben links zeigt die Ausgangssituation. Nach Eingabe des Werts 136 in das zur Frequenz u=0 gehörende Textfeld, ergibt sich die Situation im Bild oben rechts. Dann wird –65 für „U1“ eingegeben. Das Ergebnis ist im mittleren linken Bild dargestellt. Die Eingabe von –22 ins Textfeld für „U6“ und –37 ins Textfeld „U7“, ergeben die Situation, wie sie im Bild in der Mitte rechts bzw. unten links angezeigt werden. Das letzte Bild stellt die Situation dar, nachdem die Werte 30, –7, –2 und 13 für die Felder „U2“ bis „U5“ eingeben wurden, in der der Plott des Eingangssignals dem Plott des Ausgangssignals gleicht.

Um den leicht „ansteigenden Trend“ der Funktion des Eingangssignals auf die Funktion des Ausgangssignals zu übertragen, muss der Schieberegler für u=1 auf einen Wert von –65 eingestellt werden. In Abbildung 3-14 im Bild in der Mitte links sieht man das Ergebnis. Es

sind zwar noch Einstellungen für die restlichen Schieberegler nötig, um das Ausgangssignal vollständig dem Eingangssignal anzupassen, aber wenn man sich zunächst einmal darauf beschränkt, die Werte –22 und –37 in die für Frequenz u=6 bzw. Frequenz u=7 vorgesehenen Textfelder einzutragen, dann sieht man schon deutlich, dass der Plott des Ausgangssignals den Plott des Eingangssignals relativ gut approximiert. Die Auswirkung der Eingaben, lässt sich im Bild in der Mitte rechts bzw. im unteren linken Bild der Abbildung 3-14 ablesen. Gibt man noch die Werte 30, –7, –2 bzw. 13 für die Felder „U2“ bis „U5“ ein, so bemerkt man, dass die zugehörigen Frequenzen nur einen marginalen Einfluss auf den Plott des Ausgangssignals haben, der aber nun vollständig dem Plott des Eingangssignals gleicht (Abbildung 3-14 unteres rechtes Bild). Verschiebt man nun noch einmal Schieberegler „U0“, so erkennt man, dass die beiden Funktionsverläufe genau übereinander gelegen haben.

Zuletzt möchte ich noch einmal genauer auf die **MAGIC** Option im Menüpunkt **Solution**, der sich in der Menüleiste befindet, zu sprechen kommen. Da es recht mühselig ist, jedes Eingangssignal nachzubauen, nur um herauszufinden, welche Koeffizienten einzustellen sind, kann man die **MAGIC** Option benutzen. Diese Option hat die Funktion, genau die gesuchten Koeffizienten in die Textfelder der korrespondieren Frequenzen einzutragen.

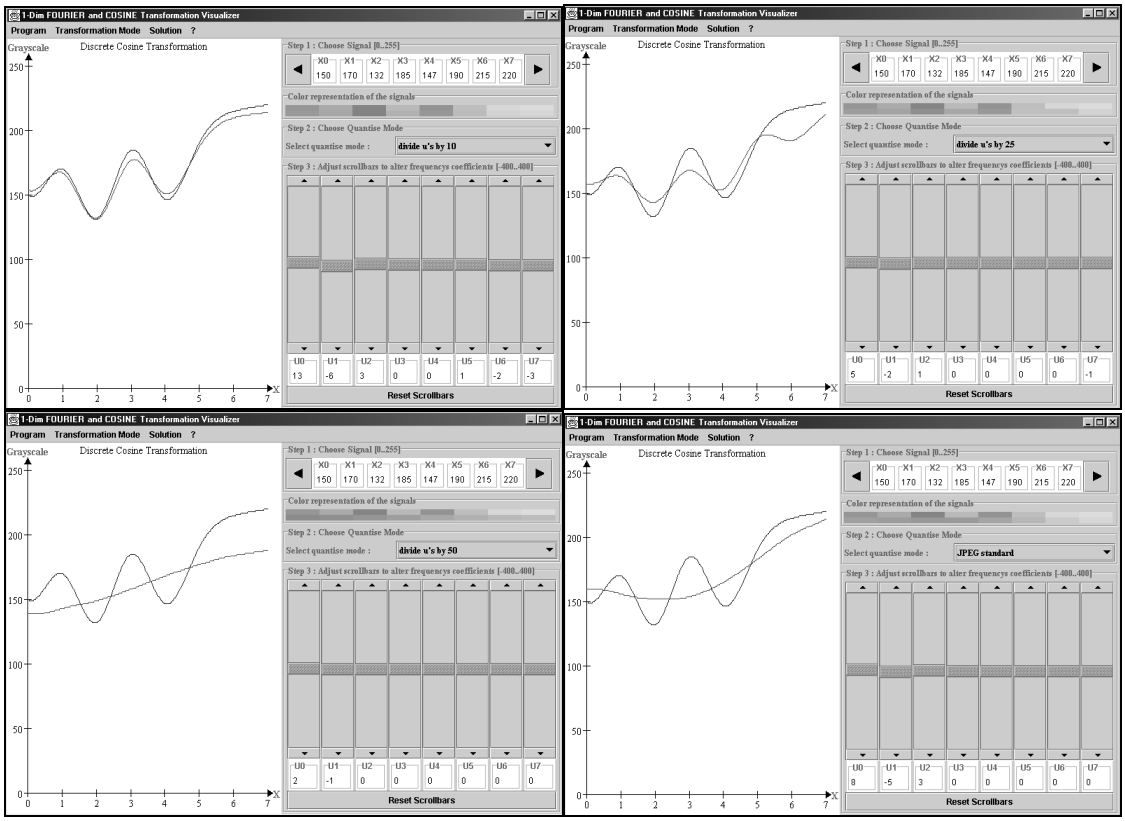

Abbildung 3-15 : Der Plott des Eingangssignals (150,170,132,185,147,190,215,220) wird unter Benutzung der **MAGIC** Option, die sich im **Solution** Menüpunkt in der Menüleiste befindet, nachgebildet. Oben links ist ein Quantisierungsfaktor von 10 benutzt worden, im rechten oberen Bild ein Qantisierungsfaktor von 25, im linken unteren Bild ein Quantisierungsfaktor von 50 und schließlich im unteren rechten Bild wird der JPEG Standard benutzt. D.h. die Koeffizienten werden durch die entsprechenden Werte des Vektors (16,11,10,16,24,40,51,61) dividiert. Die beiden oberen Bilder und das Bild unten rechts demonstrieren nachhaltig, dass bei steigendem Quantisierungsfaktor, die Approximation von Ausgangs- an Eingangssignalsignal schlechter wird.

Notiert man sich, nach Anwendung der **MAGIC** Option, die in den Textfeldern „U0“ bis „U7“ stehenden Zahlen und setzt anschließend Textfelder und Schieberegler zurück in Ausgangsposition (mit dem **Reset Scrollbars** Button unter den Textfeldern im *scrollbar panel*), so kann man sich davon überzeugen, dass die „manuelle“ Eingabe, der acht Zahlen in die entsprechenden Textfelder, zu demselben Ergebnis führt. Auf diese Weise kann man sich noch einmal das Quantisieren verständlich machen, indem man im *quantise panel* einen Quantisierungsfaktor einstellt und anschließend die **MAGIC** Option benutzt. Je größer der Quantisierungsfaktor wird, desto schlechter ist die Annäherung des Plott des Ausgangssignals an den Plott des Eingangssignals. Stellt man zum Beispiel einen Quantisierungsfaktor von 50 ein und lässt sich die Koeffizienten berechnen, so werden viele Koeffizienten der höheren Frequenzen auf Null gesetzt (durch die Rundung auf den nächsten ganzzahligen Wert). Abbildung 3-15 zeigt einige Beispiele. Dort werden verschiedene Quantisierungsfaktoren benutzt, bei dem Versuch den Plott des Ausgangssignals dem Plott des Eingangssignals anzugleichen, indem die **MAGIC** Option verwendet wird.

4. Literaturverzeichnis

[1] Martin Schader / Lars Schmidt-Thieme. *Java – Eine Einführung, 2. Auflage*, Springer-Verlag 1999.

[2] Cay S. Horstmann / Gary Cornell. *Core JAVA 2 Band 1 – Grundlagen*, Prentice Hall 1999.

[3] David M. Geary Graphic. *JAVA 2 – Mastering the JFC Volume 2, 3rd Edition*, Prentice Hall 1999.

[4] Josef Stoer. *Einführung in die Numerische Mathematik 1*, Springer Verlag 1972.

[5] Elbert Oran Brigham. *FFT: Schnelle Fourier-Transformation*, R.Oldenbourg Verlag 1995.

Anhang Darstellung des Applets mit anderen Look-and-Feel Managern

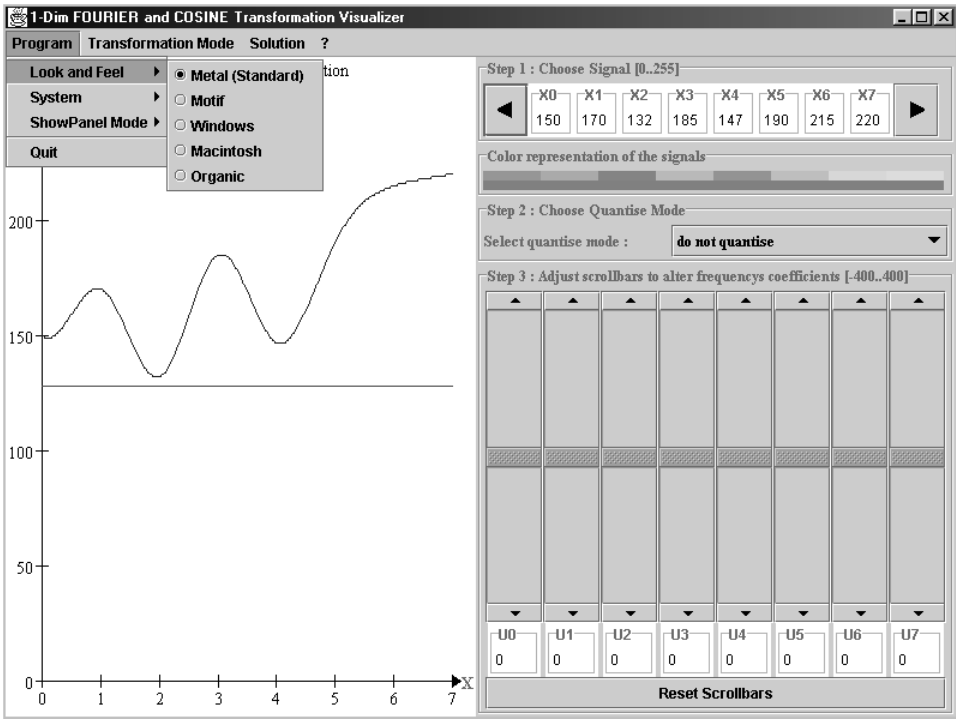

Abbildung A-1 : Der Standard Look-and-Feel (LaF) Manager „Metal“

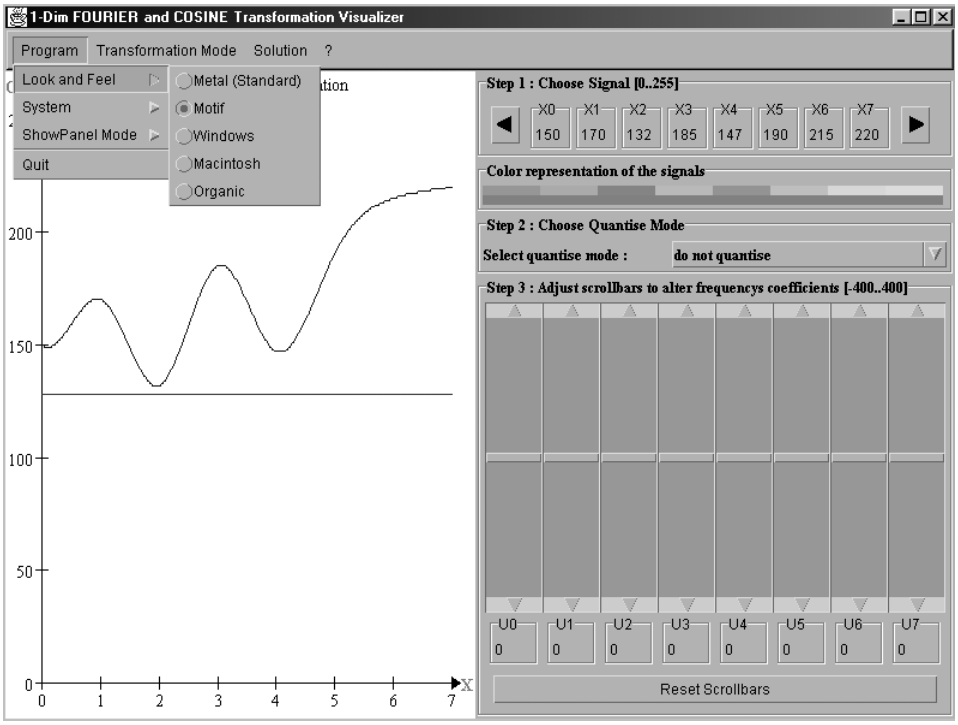

Abbildung A-2 : Der LaF Manager „Motif“ (Unix)

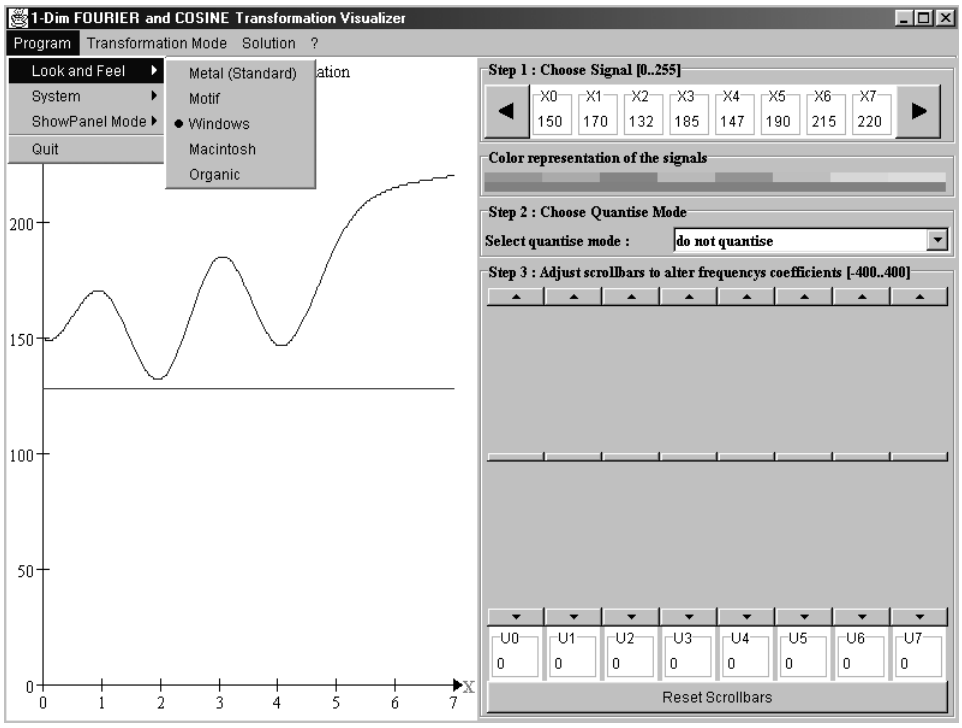

Abbildung A-3 : Der LaF Manager „Windows“.

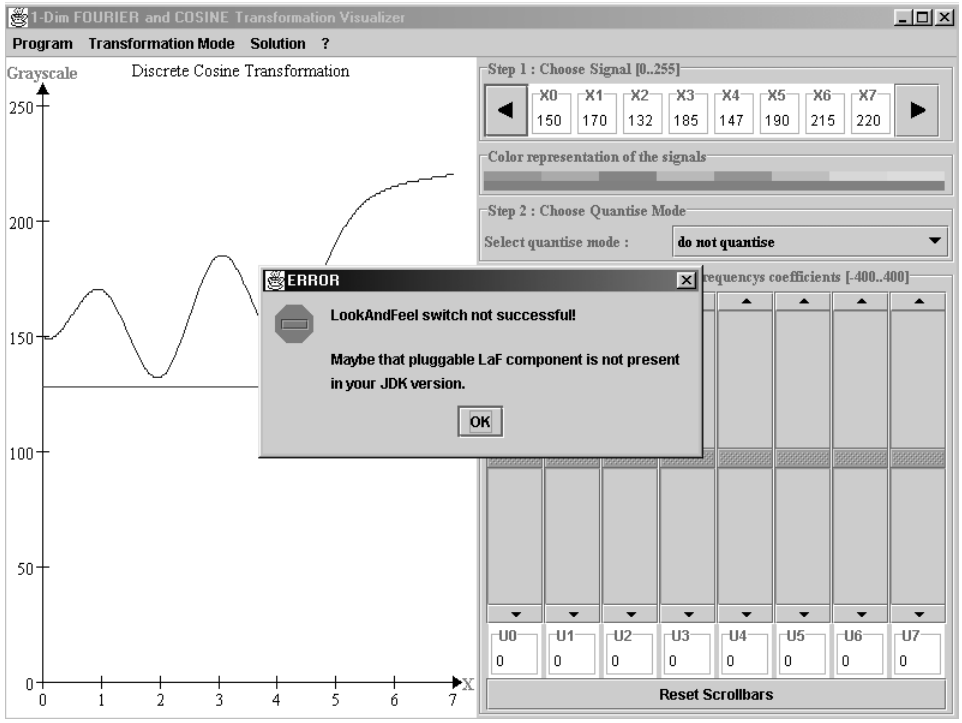

Abbildung A-4 : Der LaF Manager wurde nicht erfolgreich gewechselt.