

**Untersuchung von Fehlertoleranztechniken zur
realzeitorientierten Videokommunikation**

**Fehlertoleranz als Mittel zur Verbesserung der Bildqualität bei
Videokommunikation mit Echtzeitanforderungen**

Studienarbeit

Tim Suchanek

Universität Hamburg
Fachbereich Informatik
Arbeitsgruppe Telekommunikation und Rechnernetze (TKRN)

Betreuer: PD Dr. Klaus-Dieter Heidtmann

17.01.2000

Inhalt

Abbildungsverzeichnis.....	3
1 Einführung	4
2 Grundlegende Begriffe und Konzepte.....	6
2.1 funktionale Gliederung eines Videokonferenzsystems	6
2.2 Ursachen für Datenverlust.....	6
2.3 Gestaltungsaspekt Fehlertoleranz im Kommunikationsbereich.....	7
2.3.1 Fehlertoleranzstrategie	7
2.3.2 Fehlertoleranztechnik.....	8
2.3.3 Systementwurf	9
3 Anwendungsumfeld.....	11
3.1 Anwendungsszenario realzeitorientierte Videokommunikation.....	11
3.1.1 Realzeitorientierte Videokommunikation als Dienst	11
3.1.2 Dienstgüteunterstützung für Videokonferenzen	13
3.2 Kommunikationssystem	13
3.2.1 Ein Referenzmodell für ein Kommunikationssystem.....	14
3.2.2 Kommunikationssystem Internet	15
3.2.3 Weiterentwicklung des Internet.....	16
3.2.4 Fehlertoleranz im Internet	17
3.3 Medienstandards für realzeitorientierte Videokommunikation	18
3.3.1 Motion-JPEG.....	18
3.3.2 H.261	21
3.3.3 H.263+	22
3.3.4 MPEG-4	22
4 Fehlertoleranz	24
4.1 Allgemeine Fehlertoleranz.....	24
4.2 Fehlertoleranz und Videokommunikation	26
4.3 senderbasierte Fehlertoleranztechniken	28
4.3.1 Wiederholung auf Anfrage.....	28
4.3.2 Aufeinanderfolgende Einheiten in verschiedenen Paketen	29
4.3.3 Aufeinanderfolgende Pakete über verschiedene Wege senden	31
4.3.4 Skalierung	32
4.3.5 Bitorientierte Vorwärtsfehlerkorrektur.....	33
4.3.6 Inhaltsorientierte Vorwärtsfehlerkorrektur	35
4.4 Empfängerbasierte Fehlertoleranztechniken	37
4.4.1 Spleißen	37
4.4.2 Einfügung von Stille	38
4.4.3 Wiederholung voriger Daten.....	39
4.4.4 Wellenmustersuchen.....	40
4.4.5 Zeitstreckung.....	41
4.4.6 Rückgriff auf übertragene Zustandsinformationen.....	42
4.4.7 Modellbasierte Schätzung.....	42
5 Umgang mit Fehlern bei Bildtelefonie im Internet	44
5.1 Netzmanagement.....	44
5.2 Sitzungsmanagement	45
6 Zusammenfassung und Ausblick	49
Literatur.....	53
Anhang A: Ausgewählte Internetstandards (RFCs) und deren Status.....	56
Anhang B: RTP-Nutzlasten (RTP-payloads) und deren Status.....	57

Abbildungsverzeichnis

Abbildung 1: Funktionale Gliederung eines Videokonferenzsystemes	6
Abbildung 2: Schema einer verteilten Fehlertoleranztechnik.....	8
Abbildung 3: Anordnung der Dienstkategorien in der Benutzungshierarchie und Verteilung der Aufgaben [nach Fro 1997, S.13].....	11
Abbildung 4: Hybrides Referenzmodell.....	14
Abbildung 5: Sanduhrmodell des „Internetworking“	16
Abbildung 6: Schematischer Ablauf der JPEG-Kodierung	19
Abbildung 7: Begriffsgliederung zur Fehlertoleranz (aus [Gör 1989, S.26]).....	24
Abbildung 8: Fehlertoleranztechniken bei Datenübertragung.	28
Abbildung 9: Senderbasierte Fehlertoleranztechniken.	28
Abbildung 10: Streuung aufeinanderfolgender Pakete (<i>interleaving</i>)	30
Abbildung 11: Skalierung	33
Abbildung 12: Bitorientierte Vorwärtsfehlerkorrektur.....	33
Abbildung 13: Vorwärtsfehlerkorrektur als Transformator.....	35
Abbildung 14: Inhaltsorientierte Vorwärtsfehlerkontrolle.....	36
Abbildung 15: Empfängerbasierte Fehlertoleranztechniken	37
Abbildung 16: Spleißen von Einheiten (<i>splicing</i>)	38
Abbildung 17: Einfügung von Stille	38
Abbildung 18: Wiederholen voriger Daten	39
Abbildung 19: Wellenmustersuchen	40
Abbildung 20: Zeitstreckung	41
Abbildung 21: Rückgriff auf übertragene Zustandsinformationen	42
Abbildung 22: Modellbasierte Schätzung	42
Abbildung 23: schematischer Ablauf einer Sitzung beim Empfänger	47
Abbildung 24: Entscheidungsablauf vor einfacher Schätzung.....	48

1 Einführung

„For a generation that grew up watching Kirk, Picard, Sisko and Janeway hobnob with aliens over crystal-clear videoconferencing displays, the quality of today's videoconferencing technology is somewhat disappointing. Improvements in computer processing power and algorithms will help, but what's really missing is bandwidth – and not just a fat pipeline, but a fat undisturbed pipeline.“ [Bro 1998]

Videokonferenzsysteme die dem älteren Standard H.320 folgen, erreichen gute Bild- und Tonqualität bei einer fixen Datenübertragungsrate von 384Kbits⁻¹. Dies entspricht 6 ISDN-Kanälen. Werden weniger als 3 Kanäle verwendet, entsteht bei einer Bildgröße von 176x144 Bildpunkten und 10 Bildern pro Sekunde der Eindruck eines Notbetriebes. Systeme die zusätzlich den Standard H.323 unterstützen, ermöglichen Videokonferenzen auch über das lokale Netzwerk (LAN) und Internet. Auch bei guter Ausstattung des Netzwerkes (z.B. mit Fast-Ethernet und Maßnahmen zur Glättung des Datenverkehrs) treten durch Überlagerung des Datenverkehrs von verschiedenen Applikationen Übertragungsstörungen und Verzögerungen auf. Für lokale Anwendungen wie Überwachung oder campusweitem Fernunterricht bieten einige Firmen heute proprietäre Lösungen, die realzeitnah arbeiten, aber nicht über öffentliche Netze eingesetzt werden können.[Bro 1998]

Der erfolgreiche Entwurf und Einsatz von realzeitorientierten Applikationen über das Internet ist heute so schwierig weil

- im Internet unprognostizierbarer Datenverlust bei der Übertragung auftritt;
- die Realzeitanforderungen der Applikation nur wenig Zeit für eine Reaktion auf Übertragungsfehler lassen;
- die Erhöhung der Übertragungslast durch zusätzliche Informationen zur Sicherung gegen Fehler zu Überlast im Netz und damit zu weiteren Verlusten führen kann;
- viele Teilnehmer an der Sitzung teilnehmen könnten und sowohl individuelle als auch gemeinschaftliche Verluste auftreten könnten;
- neben den Daten selbst auch Synchronisationsinformationen wiederhergestellt werden müssen, damit Bild und Ton zueinander passend präsentiert werden können;
- das Verhalten des Teilnehmers und damit auch die Netzlast (z.B. in Form von Video- und Audiodaten) unprognostizierbar ist.

[WaZ 1998]

Realzeitorientierte Applikationen sollen in Zukunft durch ein Management unterstützt werden, welches die Ressourcen des Netzes verwaltet und die Netzlast steuert. Für diese Dienstgütemanagement-Komponente werden Modelle benötigt über

- die Netzinfrastruktur;
- die Anwendungen, die Ressourcen anfordern werden;
- die jeweils erzeugte Netzlast;
- Umfang und Auswirkung von Fehlern.

Diese Arbeit beschäftigt sich mit dem Fehlertoleranzaspekt einer Applikation zur IP-basierten, realzeitorientierten Bildfolgenübertragung und liefert damit einen Beitrag für Modelle über die Anwendungen und die jeweils erzeugte Netzlast.

Dabei wird zurückgegriffen werden auf

- Erfahrungen mit realzeitorientierten Applikationen wie der Funktelefonie und Audio-über-Internet [z.B. BAN 1998, PHH 1998, ASW 1998];
- Erfahrungen mit der Kompression von digitalen Mediendaten [Gea 1998, DÜM 1999];

- Erfahrungen aus dem Bereich des Entwurfs und Betriebes fehlertoleranter Systeme [Ech 1990, Gör 1989];
- Messungen von Internetverkehr [z.B. Ste 1998, WeC 1999];
- Untersuchung der Fehlertoleranz bei Kommunikationsstandards [z.B. FGV 1998, HKZ 1999, Pea 1997].

Diese Arbeit verfolgt vier Ziele. Zunächst wird herausgearbeitet, welche Aspekte realzeitorientierter Videokommunikation von Datenverlusten und Fehlertoleranz betroffen sind. Anschließend soll ein Überblick gegeben werden über verschiedene Fehlertoleranztechniken und die Verwendbarkeit für die Übertragung von Bildfolgen wird untersucht. Abschließend werden Kombinationsmöglichkeiten einzelner Techniken für Bildtelefonie diskutiert und ein Ansatz zur weiteren Untersuchung vorgeschlagen.

Zunächst wird im zweiten Kapitel auf die funktionale Gliederung eines Videokonferenzsystems eingegangen und auf die Ursachen für Datenverluste. Ferner wird der Fehlertoleranzaspekt beim Entwurf eines IT-Systems erläutert und die Begriffe Fehlertoleranzstrategie und Fehlertoleranztechnik definiert. Aus dem Einsatz einer Fehlertoleranztechnik für realzeitorientierte Videokommunikation ergeben sich Bedingungen die im dritten Kapitel erläutert werden. Im vierten Kapitel werden dann Techniken vorgestellt, die z.T. im Zusammenhang mit Funktelefonie und Audio-über-Internet untersucht wurden. Als Kernteil der Arbeit werden die Techniken auf eine Verwendbarkeit für Bildfolgen untersucht. Abschließend wird im fünften Kapitel diskutiert, wie einzelne dieser Techniken sinnvoll kombiniert werden können. Die Arbeit schließt mit einer Zusammenfassung und einem Ausblick.

2 Grundlegende Begriffe und Konzepte

2.1 funktionale Gliederung eines Videokonferenzsystems

In diesem Abschnitt wird ein IT-System zur realzeitorientierten Videokommunikation schematisch vorgestellt, Ursachen für Datenverlust erläutert und der Gestaltungsaspekt Fehlertoleranz vorgestellt. Das folgende Ablaufmodell realzeitorientierter Videokommunikation orientiert sich an der komponentenbasierten Definition eines Kommunikationssystems bei Froitzheim [Fro 1997, Kap.1] und der funktionalen Gliederung eines Videokommunikationssystems bei Wang und Zhu [WaZ 1998] (Abbildung 1):

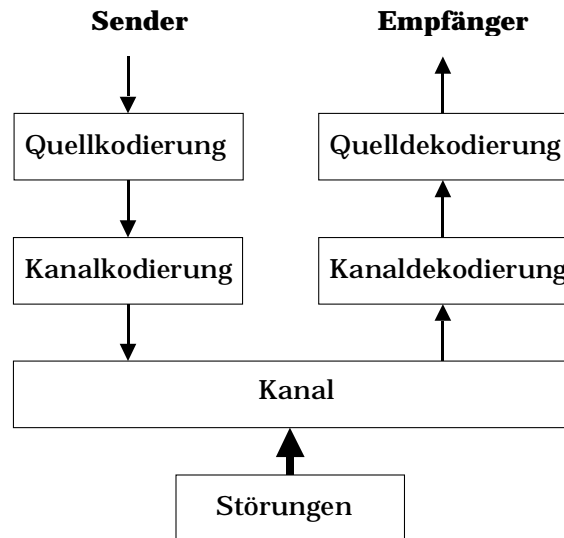


Abbildung 1: Funktionale Gliederung eines Videokonferenzsystemes

Eine Applikation zur realzeitorientierten Videokommunikation steuert beim Empfänger Aufnahmegeräte für Audio- und Videodaten und nimmt gleichzeitig eine Kodierung der Daten vor. Diese Kodierung erfolgt in zwei Schritten: In der *Quellkodierung* werden die analogen Daten in digitale umgewandelt, überflüssige Redundanz wird entfernt und Steuerdaten hinzugefügt, um die resultierenden Daten speichern, übertragen und dekodieren zu können. In der *Kanalkodierung* werden die digitalen Daten für die Übertragung über das unterliegende Kommunikationssystem aufbereitet, indem Sie auf Pakete verteilt und Header generiert werden. Alle algorithmischen Schritte, die ohne Wissen über den Inhalt der Daten durchgeführt werden, werden der Kanalkodierung zugeordnet. Über das Kommunikationssystem, das einen oder mehrere Kommunikationskanäle zur Verfügung stellt, werden die Pakete zum Empfänger transportiert. Dort findet zunächst eine Kanal-Dekodierung statt, gefolgt von einer Quell-Dekodierung und Präsentation der resultierenden Daten. Wenn der Kanal einen Vollduplexbetrieb erlaubt, d.h. eine simultane Übertragung in beide Richtungen, können alle Teilnehmer gleichzeitig als Sender und Empfänger fungieren. [Fro 1997, Kap. 1; WaZ 1998] Das Ablaufmodell vernachlässigt die Querschnittsaufgaben des Sitzungs- und Netzmanagements. Auf diese wird in Abschnitt 2.1.1 näher eingegangen.

2.2 Ursachen für Datenverlust

Es wird angenommen, daß die Anwendungen lokal auf ausreichend ausgestatteten Computern ablaufen und von deshalb keine Datenverluste in den Applikationen auftreten. Der Kanal kann eine einfache Übertragungsstrecke sein (z.B. ein Draht

oder Luft) oder ein komplexes paketvermitteltes Netz (z.B. das Internet). Diese starke Vergrößerung in bezug auf Fehler ist angemessen, weil sich auch bei größerer Detaillierung des Kanales bis heute keine Modelle (Fehlerprofile) haben finden lassen, die hinreichend genaue Vorhersagen über Auftreten und Umfang von Fehlern zulassen [z.B. Cea 1998]. Das liegt an den vielfältigen Ursachen, die Verluste haben können. Hier folgt eine Auswahl: [Tan 1997, Kap.2]

- bei Übertragung von elektromagnetischen Wellen über Luft (abhängig von der Frequenz): Störung durch Witterung, Störstrahlung durch fremde Elektroanlagen, Brechung in der Atmosphäre, Absorption durch Gegenstände;
- bei Übertragung von elektrischem Strom über Kupferkabel: Dämpfung, d.h. Energieverlust bei Übertragung und unkorrekte Verstärkung;
- bei Übertragung von Licht über Glasfaser: kaum meßbare Verluste, nur etwas Dämpfung;
- unkorrekte Algorithmen für die Bearbeitung der Pakete in Vermittlungsrechnern;
- Speicherüberlauf in Vermittlungsrechnern (*router congestion*);
- Kollisionen von Paketen in gemeinsam genutzten Medien;
- u.v.a.m..

Untersuchungen über die Auswirkung von Übertragungsfehlern auf Bilddatenströme haben gezeigt, daß bereits der Verlust von geringen Datenmengen durch Fehlerfortpflanzung, besonders eine fehlerhafte Differenzendekodierung, zu starken Einbußen in der Bildqualität führen können. [Koh 1999, SEK 1999]

2.3 Gestaltungsaspekt Fehlertoleranz im Kommunikationsbereich

Die beiden folgenden Fragestellungen müssen bei der Gestaltung einer Applikation in bezug auf Übertragungsfehler berücksichtigt werden:

- Welche Methoden zur Fehlerbehandlung sollen implementiert werden?
- Wie sollen diese Methoden zur Laufzeit gesteuert werden?

Die erste Frage betrifft den Mix aus Techniken, welcher verwendet werden soll. Die zweite fragt danach, *wie* diese Techniken eingesetzt werden sollen. Die Beantwortung dieser Fragen stellt die Festlegung einer Fehlertoleranzstrategie dar.

In dieser Arbeit bezieht sich Fehlertoleranz alleine auf den Umgang mit Übertragungsfehlern. Ein allgemeinerer Zusammenhang zwischen Fehlertoleranz und Videokommunikation ist in Kapitel 3 dargestellt.

2.3.1 Fehlertoleranzstrategie

Eine Fehlertoleranzstrategie ist die Verwirklichung eines Mixes verschiedener Techniken zur Behandlung von Übertragungsfehlern. Dazu folgendes Beispiel: Der Entwickler entscheidet, zwei Methoden zur Behandlung von Paketfehlern zu implementieren: Eine aufwendige zur exakten Rekonstruktion und eine einfache zur annähernden Rekonstruktion. Wenn zur Laufzeit ein Paketfehler entdeckt wird, wird zunächst ermittelt wieviel Zeit zur Behandlung bleibt bis die verlorenen Daten präsentiert werden müssen. Wird ein Schwellwert unterschritten, wird die einfache Methode benutzt, sonst die aufwendige.

Die Fehlertoleranzstrategie

- hat eine globale Sicht der Dinge;
- wird durch Implementation realisiert;
- ist zur Laufzeit in das Sitzungs- und/oder Netzmanagement integriert und darum durch den Benutzer steuerbar;
- löst global den Zielkonflikt zwischen bestmöglicher Präsentationsqualität bei geringstmöglichem Ressourcenverbrauch (Sicherung der Dienstgüte);

- ist ein Gestaltungsaspekt des Systemdesigns und unterliegt damit globalen Randbedingungen der Realisation.

2.3.2 Fehlertoleranztechnik

Eine Fehlertoleranztechnik ist eine Methode zur Fehlerbehandlung. Die Behandlung kann dabei nur kurzfristig im Rahmen der Reaktionsmöglichkeiten erfolgen, die die Fehlertoleranzstrategie zuläßt. Eine Fehlertoleranztechnik im Kommunikationsbereich läßt sich konzeptionell aufteilen in eine Vorverarbeitung im Sender und eine Fehlerbehandlung im Empfänger (Abbildung 2).

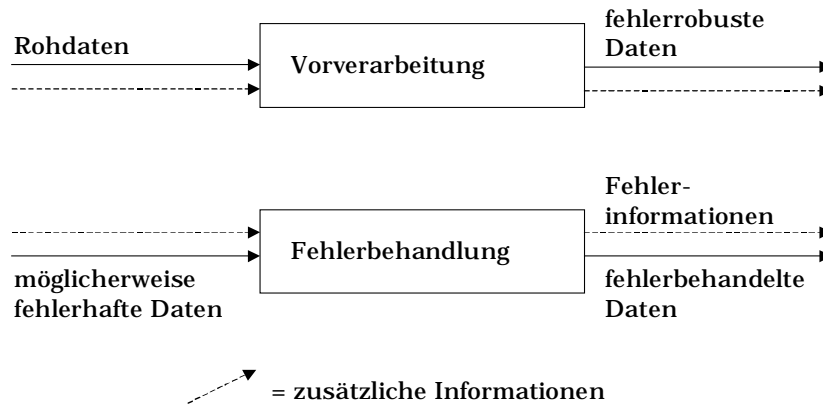


Abbildung 2: Schema einer verteilten Fehlertoleranztechnik

Zielsetzung jeder Fehlertoleranztechnik ist es, fehlerbehandelte Daten zu erhalten, die so weiterverarbeitet werden, daß das Ergebnis für den Anwender einen höheren Nutzen im Sinne der Anwendung darstellt als ohne Fehlerbehandlung. Fehlertoleranztechniken können geschachtelt angewandt werden. Techniken, bei denen Vorverarbeitung und Fehlerbehandlung zur Laufzeit kommunizieren, sind möglich.

Die Fehlerbehandlung kann folgende Schritte umfassen [Tal 1998]:

- Fehlererkennung und Lokalisierung (*error detection and localization*);
- Resynchronisation des Datenstromes (Wiederherstellung des richtigen Timings);
- exakte Wiederherstellung der Daten (*data recovery*);
- Ausfüllen verbleibender Lücken (*error concealment*) mit statistischen Schätzern.

Je nach Technik können ein oder mehrere Behandlungsschritte fehlen. Auch eine Methode, die nur Informationen über den Umfang von Datenverlusten sammelt, ist demnach eine Fehlertoleranztechnik.

Für den Bereich der Datenübertragung unterscheiden Perkins, Hodson und Hardman [PHH 1998] sender- und empfängerbasierte Techniken:

- Bei senderbasierten Techniken basiert die Fehlerbehandlung beim Empfänger auf einer Vorverarbeitung beim Sender (Beispiel: Auswertung einer Prüfsumme).
- Bei empfängerbasierten Techniken ist die Fehlerbehandlung unabhängig von einer Vorverarbeitung beim Sender. (Beispiel: Einfügen von Stille bei Verlust von Sprachdaten).

Senderbasierte Techniken benutzen eine Erwartung über das Auftreten der Verluste, das sogen. Fehlerprofil, um die Daten gegen dieses Profil exakt oder annähernd in der Vorverarbeitung abzusichern. Treten Fehler unerwartet auf, d.h. außerhalb des Fehlerprofils, kann die Behandlung wirkungslos sein oder sogar zu Fehlerfortpflanzung führen. Empfängerbasierte Techniken benutzen eine Erwartung

über den Inhalt der Daten (Wissen über das verwendete Protokoll oder über das Verhalten des Teilnehmers vor Kamera und Mikro) um aufgrund *dieses* statistischen Modelles einen Schätzer für die entstandene Lücke im Datenstrom zu erzeugen. Unerwartetes Verhalten des Teilnehmers kann diese Technik wirkungslos machen oder sogar zu Fehlerfortpflanzung führen. Empfängerbasierte Techniken arbeiten ohne Vorverarbeitung beim Sender.

Einfache Techniken gehen von strengen Annahmen aus, d.h. einem sehr eingeschränkten Fehlerprofil, und arbeiten effektiv und effizient, sind aber nicht flexibel. Komplexe Techniken berücksichtigen weichere Annahmen und verbrauchen darum üblicherweise mehr Ressourcen bei in der Regel ungenaueren Resultaten. Neben den genannten Annahmen können weitere Informationen einfließen (z.B. Zeitinformationen).

Auf welcher Ebene eine Fehlertoleranztechnik angesiedelt wird, hängt von dem Wissen ab, das die Technik zur Fehlerkorrektur benutzt. Werden z.B. gezielt Bewegungsvektoren eines H.261-kodierten Videostromes geschätzt, so ist die Technik in der Quellkodierung anzusiedeln, weil Wissen über den Inhalt der Daten verwendet wird. Werden angekommene Pakete zu einem Datenstrom zusammengesetzt und dabei bitorientiert die Prüfsummen aus den Paketköpfen ausgewertet, so ist diese Technik in der Kanalkodierung anzusiedeln. Wenn zur Übertragung das Internet benutzt wird, könnten zur Laufzeit auch dort Fehlertoleranztechniken eingesetzt werden, z.B. um Bitfehler automatisch zu korrigieren oder um wiederholte Sendung verlorener Pakete automatisch anzufordern.

Eine Fehlertoleranztechnik

- hat nur eine lokale Sicht der Dinge und arbeitet relativ isoliert im Gesamtsystem;
- hat eigene Stärken und Schwächen: Manche können bestimmte Fehlertypen besonders gut behandeln, andere behandeln alle Fehler gleich gut, aber mit unterschiedlichem Rechenaufwand, eine dritte Kategorie verwendet Ressourcen, die nicht immer verfügbar sind etc.;
- ist zur Laufzeit nur begrenzt über Parameter von außen zu steuern oder reguliert sich unabhängig selbst;
- löst lokal und weitgehend automatisch den Zielkonflikt zwischen bestmöglicher Korrektur bei geringstmöglichem Ressourcenverbrauch (Speicher, Datenrate, Prozessor);
- stellt zur Realisierung konkrete Forderungen an das Gesamtsystem.

2.3.3 Systementwurf

Bei der Entwicklung einer Applikation ist eine Strategie festzulegen. Dabei werden Randbedingungen für die Techniken geschaffen. Andererseits stellen die einzelnen Techniken Anforderungen an das System. Randbedingungen und Forderungen müssen durch das Design effektiv und effizient zusammengebracht werden. Nicht nur über Fehlertoleranz sind strategische Entscheidungen zu treffen. Die Systemgestaltung besteht vielmehr aus der Festlegung einer Sammlung von Strategien, die alle Aspekte der Applikation umfassen. Die Strategien sind teils verwoben. Alle Strategien gemeinsam geben die Randbedingungen vor, die die Grenzen der Implementationen darstellen. Fehlertoleranz ist dann erfolgreich, wenn die Strategie erfolgreich gewichtet mit den anderen integriert wurde.

In dieser Arbeit werden Fehlertoleranztechniken für realzeitorientierte Videokommunikation über Internet untersucht. Daraus ergeben sich als Bedingungen für die Entwicklung einer Fehlertoleranzstrategie:

- Anwendungsszenario: Videokommunikation mit Echtzeitanforderungen;
- Kommunikationssystem: verlustbehaftetes Paketvermittlungsnetz (Internet);
- Standards für Medienströme: MPEG-x, H.26x.

Diese Bedingungen werden im nächsten Kapitel vorgestellt.

3 Anwendungsumfeld

3.1 Anwendungsszenario realzeitorientierte Videokommunikation

Erfolgreiche Fehlertoleranztechniken unterstützen die Dienstgüte effizient und effektiv. Durch ihren Einsatz erhöht sich der Nutzen für den Benutzer. Aber was ist die Dienstgüte? Inwieweit läßt sich dieser Begriff bei realzeitorientierter Videokommunikation konkretisieren? Dazu werden in diesem Abschnitt folgende Fragen behandelt:

- Welche realen Situationen sollen durch den Dienst unterstützt werden?
- Was ergibt sich daraus für den Gestaltungsaspekt Fehlertoleranz?

3.1.1 Realzeitorientierte Videokommunikation als Dienst

Froitzheim [Fro 1997, Kap. 5.3] ordnet die realzeitorientierte Videokommunikation als „kooperativer Telepräsenzdienst“ in die Kategorie der Anwendungsdienste ein. Anwendungsdienste dienen „dem Austausch strukturierter Information“ und werden abgegrenzt gegen Übertragungs- und Steuerungsdienste [Fro 1997, Kap. 1.2]. Abbildung 3 verdeutlicht diese Taxonomie und nennt die Hauptaufgaben der Dienstkategorien.

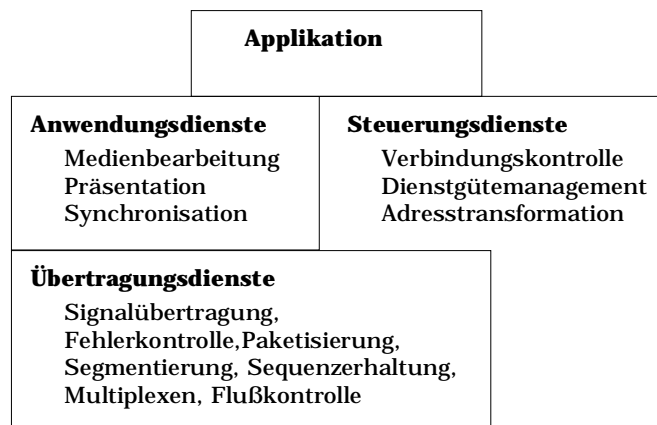


Abbildung 3: Anordnung der Dienstkategorien in der Benutzungshierarchie und Verteilung der Aufgaben [nach Fro 1997, S.13]

Menschen können die Applikation benutzen oder direkt Anwendungs- oder Steuerungsdienste in Anspruch nehmen. Die Applikation ihrerseits nutzt Anwendungs- und Steuerungsdienste. Anwendungsdienste nutzen Übertragungsdienste. Steuerungsdienste können Übertragungsdienste nutzen oder auch nicht und stellen das Management dar.

Kooperative Telepräsenzdienste sind Dienste, die Besprechungen ermöglichen sollen ohne physische Bewegung nötig zu machen. Der Dienst überträgt die Medienströme zwischen den entfernten Teilnehmern so, daß der Eindruck entsteht sie wären am selben Ort. Dabei ist das Ziel, möglichst viele organisatorische Optionen von Besprechungen technisch zu unterstützen, um den Dienst möglichst flexibel einsetzbar zu machen. Beispiele für organisatorische Optionen von Besprechungen folgen. Die Besprechung kann

- zwischen zwei oder mehr Teilnehmern stattfinden;
- nur eingeladenen Personen zugänglich sein oder öffentlich sein;

- für Zuschauer öffentlich sein, Beiträge dürfen aber nur bestimmte Personen liefern;
- von einem Moderator gelenkt werden oder unmoderiert sein;
- vorzeitig von manchen Teilnehmern verlassen werden, andere könnten später hinzukommen;
- protokolliert werden oder nicht;
- von fester Dauer sein oder zeitlich offen;
- Beiträge von mehreren Beteiligten zulassen.

Die organisatorischen Optionen werden durch die Teilnehmer oder durch Außenstehende wahrgenommen, indem diese eine bestimmte Rolle (Sender, Empfänger, Moderator, Hausmeister) übernehmen und die damit verbundene Funktion ausführen. Eine zusätzliche Dimension der Flexibilität wird erreicht, wenn nicht nur möglichst viele Rollen unterstützt werden, sondern wenn zusätzlich Wechsel zwischen allen Rollen während der Besprechung technisch möglich sind. Die Frage nach den realen Situationen, die unterstützt werden sollen, hat also auf die Frage geführt, welche Rollen den Teilnehmern technisch angeboten werden sollten und wie Rollenwechsel stattfinden.

Bei Videokommunikation sind die Teilnehmer durch einen Sprachkanal, einen Bildkanal und einen Steuerkanal verbunden. Im allgemeinsten Fall hat jeder Teilnehmer zu jedem anderen einen Vollduplexkanal technisch zur Verfügung. Klar ist, daß jeder Teilnehmer seine Aufmerksamkeit nicht auf alle anderen gleichzeitig richten kann. Trotzdem scheint es wünschenswert, wenn alle mit der Besprechung zusammenhängenden Daten an jeden Partner übertragen werden, um technisch möglichst einfach Rollenwechsel zu ermöglichen. Die wichtigsten Rollen von Teilnehmern an einer Besprechung wurden bereits genannt und werden nun kurz erläutert:

- Sender: Hat auf sich die Aufmerksamkeit gelenkt, sendet einen Beitrag.
- Empfänger: Spendet Aufmerksamkeit, nimmt den Beitrag auf, um eventuell mit eigenem Beitrag zu reagieren.
- Moderator: Entscheidet über Kommunikationslenkung.
- Zuschauer: Erweckt den Anschein, in Zukunft nicht Sender werden zu wollen.
- Hausmeister: Stellt die Infrastruktur (Raum, Medium) zur Verfügung.

Im Verlauf einer Besprechung wechseln die Teilnehmer in der Regel häufig ihre Rollen. Nach ihrem Ende (oder nachdem sie eine gewisse Zeit läuft und man eine Theorie vom Verhalten der Teilnehmer hat) kann man versuchen, die Besprechung einem Anwendungsprofil zuzuordnen. Bekannte Profile sind

- Bildtelefonie: Zwei Partner führten ein Gespräch über Bildtelefon.
- Videokonferenz: Mehr als zwei Partner kommunizierten miteinander.
- Live-Übertragung: Viele Zuschauer verfolgten die Beiträge weniger Sender. Ein Moderator lenkte die Kommunikation.

Weitere sind denkbar.

Bei realzeitorientierter Videokommunikation werden die Daten direkt nach der Entstehung beim Sender zum Empfänger übertragen und dort unverzüglich abgespielt. Die Teilnehmer haben erheblichen Einfluß auf die konkrete Ausgestaltung der Kommunikation (Ablauf und Inhalt).

Bei nicht realzeitorientierter Videokommunikation liegen die zu übertragenden Daten bereits vor Beginn der Kommunikation vor. Der Vollständigkeit halber seien als bekannte Anwendungsprofile hier erwähnt:

- Video-on-demand: Zuschauer bestellten fertige Beiträge und spielten sie bei sich (während oder nach) dem Herunterladen ab.

- Broadcast: Ein Sender sendet einen (fertigen oder während einer Sendung entstehenden) Beitrag. Die Zuschauer haben höchstens geringe und zeitverzögerte Eingriffsmöglichkeiten.
- Multicast: Wie Broadcast, die Anzahl der Zuschauer ist aber organisatorisch begrenzt.

Die Übergänge zwischen den einzelnen Anwendungsprofilen sind fließend. Zur Laufzeit (also wenn die Kommunikation tatsächlich stattfindet) ist eine Klassifikation u.U. schwierig, besonders, wenn während der Kommunikation Partner dazukommen oder ausscheiden dürfen. Realzeitnah ist eine Kommunikation dann, wenn ausreichend der Eindruck unverzögerter Datenwiedergabe erweckt wird (Die Literatur erwähnt 0,15-0,25s als Richtwert [PeH 1998]).

3.1.2 Dienstgüteunterstützung für Videokonferenzen

Die Fehlertoleranz hat die Aufgabe, jedem Teilnehmer die Dienstgüte anzubieten, die mindestens notwendig ist, um eine sinnvolle Erfüllung der Rolle zu gewährleisten, die der Teilnehmer einzunehmen gedenkt. Für Sender und Empfänger wären das Vollduplex 20 Bilder pro Sekunde beim Video und 3000Hz Bandbreite bei Audio und die Informationen über die Besprechung, die der Moderator für sie freigegeben hat. Für Zuschauer reicht ein Simplexkanal. Der Hausmeister braucht adäquate Indikatoren über die subjektive und objektive Leistung der Infrastruktur. Der Moderator stellt die höchsten Anforderungen an die Fehlertoleranz, weil ihm neben der Möglichkeit zum Senden und Empfangen zusätzlich noch Eingriffsmöglichkeiten in die Agitationsmöglichkeiten der anderen Teilnehmer geboten werden müssen.

In einem System zur realzeitorientierten Videokommunikation ist die Fehlertoleranzstrategie erfolgreich, die Wissen über die Rollen der Teilnehmer benutzt und zur Laufzeit wechselnde Rollen verarbeiten kann. Sie kann nämlich die Wichtigkeit verlorener Informationen für jede Rolle einschätzen und so die Ressourcen effizienter nutzen als starre Strategien. Ein zu einer bestehenden Besprechung hinzukommender Teilnehmer könnte seiner lokalen Applikation signalisieren, daß er nicht in den Gesprächsverlauf eingreifen will, sondern nur zum Zuschauen gekommen ist. Daraufhin könnte die Applikation die eintreffenden Daten einige Sekunden verzögert abspielen und damit schon entscheidend weniger Paketverluste erfahren, als das bei realzeitorientierter Präsentation der Fall wäre, wo Daten rechtzeitig präsentiert werden müssen.

Es ist sehr wahrscheinlich, daß ein Teilnehmer nach dem Senden eines eigenen längeren Beitrages eine zeitlang schweigt, so daß die Fehlertoleranztechnik während ein Beitrag gesendet wird, bereits eine Redundanzsendung vorbereiten könnte, die direkt nach dem Ende des Beitrages über den Sprachkanal des Teilnehmers gesendet wird. So wird die für Audio reservierte Bandbreite kurzfristig anderweitig genutzt, ohne daß die Besprechung dadurch gestört wird. Weitere Beispiele sind denkbar.

3.2 Kommunikationssystem

Im vorigen Abschnitt wurde die Vorstellung entwickelt, reale Situationen würden durch eine flexible Applikation unterstützt. Diese Applikation kann ihren Dienst aber nur unter Zuhilfenahme eines Kommunikationssystemes erbringen, das die Übertragungsdienste realisiert und die Steuerungsdienste ermöglicht. Heute gibt es viele verschiedene Kommunikationssysteme mit verschiedenen Fehlerprofilen. Dieser Abschnitt geht folgenden Fragen nach:

- Wie kann ein Kommunikationssystem modelliert werden?

- Was charakterisiert das Kommunikationssystem Internet?
- Wie wird das Internet weiterentwickelt?
- Wo können Fehlertoleranztechniken in Internetumgebungen eingeordnet werden (d.h. in welcher Schicht und in welchen Protokollen?)

3.2.1 Ein Referenzmodell für ein Kommunikationssystem

In Abbildung 4 wird das Kommunikationssystem allgemein in Anlehnung an das hybride Referenzmodell für Computernetzwerke von Tanenbaum [Tan 1997, S. 61] in Schichten unterteilt:

Verarbeitungsschicht
Transportschicht
Vermittlungsschicht
Sicherungsschicht
Bitübertragungsschicht

Abbildung 4: Hybrides Referenzmodell

Die Applikation zur realzeitorientierten Videokommunikation wird der Verarbeitungsschicht zugeordnet. Jede Schicht besteht lokal aus einem (oder mehreren) Diensten, d.h. laufenden Programmen. Pakete mit Daten können nur zwischen Diensten verschiedener Schichten über Dienstzugriffspunkte der Dienste übergeben werden. Wenn z.B. ein Anwender neben der Applikation noch eine Textverarbeitung aufgerufen hat, so ist diese als weitere Applikation in der Verarbeitungsschicht anzusehen.

Nur auf der Bitübertragungsschicht besteht tatsächliche eine physikalische Verbindung. Dienste der Sicherungsschicht kommunizieren unter Benutzung von Diensten der Bitübertragungsschicht so, als wären Sie direkt verbunden. Die Applikationen nutzen direkt Dienste der Transportschicht. Auf Dienste der Vermittlungsschicht können die Applikationen nur indirekt über entsprechende Dienste der Transportschicht zugreifen.

Das Schichtmodell strukturiert das Kommunikationssystem durch Einführung hierarchischer Sichtbarkeitsbereiche. In jeder höheren Schicht werden neue Informationen zur Bewältigung der Aufgaben des Kommunikationssystems hinzugenommen und so von unten nach oben komplexere Dienste angeboten. Die unterste Schicht zur Bitübertragung abstrahiert soweit möglich vom real benutzten Übertragungsmedium und der dazu passend eingesetzten Übertragungstechnik und stellt nach oben die Sichtweise einer „bit-transportfähigen Röhre“ zur Verfügung. Durch diese können Bits als kontinuierlicher Strom oder paketweise geschickt werden. Die Sicherungsschicht setzt auf diese Schicht auf und stellt die Benutzung dieser „Röhre“ sicher, ohne daß es zu einfach steuerbaren Fehlern kommt wie Bitstau oder Vermischung unterschiedlicher Sendungen. Die dritte Schicht, die Vermittlungsschicht, leistet einen erheblichen weiteren Abstraktionsschritt, indem

nach oben das gesamte verfügbare Netz als zugreifbarer Adreßraum dargestellt wird. Sendungen werden nicht mehr Hardware-orientiert als Rahmen zwischen einem physikalisch vorhandenem Sender und einem direkt angeschlossenen physikalisch vorhandenen Empfänger dargestellt, sondern als abstrakte Pakete zwischen Sender- und Empfängerobjekten. Das können Prozesse oder Objekte sein, die nach Bedarf entstehen und verschwinden. Die vierte Schicht ist die Transportschicht, die vom Netz abstrahiert und Sender- und Empfängerobjekte applikationsnah verwaltet. Die Dienste der Transportschicht nutzen Kenntnisse über die von der Applikation gewünschte Dienstgüte und die von der Vermittlungsschicht erbringbaren Leistungen zusammen mit den lokalen Ressourcen um die Dienstgüte applikationsnah zu verbessern.

Das Schichtmodell beschreibt den statischen Aufbau des Kommunikationssystems. Die dynamische Komponente wird in der Regel durch analytische, simulative oder hybride Ablaufmodelle modelliert, in denen die einzelnen Dienste in ihren logischen Zusammenhängen beschrieben werden und die durch die Benutzung erzeugte Last untersucht wird. Wurde ein System realisiert, ist entschieden, in welcher Schicht welche Aufgaben wie realisiert wurden. Die Modelle über den Ablauf und Erkenntnisse zur Laufzeit dienen der (mehr oder weniger automatisierten) schichtübergreifenden Steuerung des Kommunikationssystems und der Applikationen. Auf der Verarbeitungsebene an der Schnittstelle zur Transportschicht findet die Kanalkodierung statt.

3.2.2 Kommunikationssystem Internet

Wie schon erwähnt, sprengt die Behandlung der vielfältigen existierenden Übertragungsdienste in Bezug auf Fehlertoleranztechniken für realzeitorientierte Videokommunikation den Rahmen dieser Arbeit. Eine Übersicht liefert z.B. Tanenbaum [Tan 1997].

Spezifikationsvorschläge für Dienste im Internet werden durch die *Internet Engineering Task Force* (IETF, www.ietf.org) erarbeitet und als *Request for Comments* (RFCs) veröffentlicht.

Das Internet verbindet heute über 100.000 Teilnetze mit sehr vielen verschiedenen Techniken auf den unteren beiden Schichten über das Vermittlungsprotokoll „IPv4“ (Internet-Protocol Version 4, RFC 791*) und die Namensverwaltung „DNS“ (Domain Name Service, RFC 1035*) zu einem globalen Adreßraum [Tan 1997]. Es bietet nur in sehr geringem Maße vorverhandelbare Dienstgüte und stellt damit einen Transportdienst nach besten Möglichkeiten (*best-effort*) dar. Es ist dezentral verwaltet, stellt ein irregulär vermaschtes Netz mit hierarchischem Adreßraum dar und erlaubt nur eine schwache Kontrolle der Wegeermittlung. Heute wird eine große Anzahl von Protokollen auf IPv4 eingesetzt. Auf Transportebene sind das *Transmission Control Protocol* (TCP, RFC 793*) und das *User Datagram Protocol* (UDP, RFC 768*) am verbreitetsten. Im Zusammenhang mit der Dienstvielfalt auf allen Schichten außer der Vermittlungsschicht wird auch vom „Sanduhrmodell des Internetworking“ gesprochen [zitiert in Tra 1999, S.69] (Abbildung 5)

Das Internet bietet damit folgende Anreize, Applikationen zur realzeitorientierten Videokommunikation zu realisieren:

- IP abstrahiert von der zugrundeliegenden Kommunikationstechnik und macht Applikationen damit weitgehend unabhängig vom Netzbetreiber.
- DNS ermöglicht eine weltweite Kommunikationsvermittlung.

* Siehe „Auswahl einiger Internetstandards und deren Status“ im Anhang

- Eine Ausbreitung der Applikation ist Schritt-für-Schritt möglich durch die dezentrale Verwaltung des Internet und weil keine neue Technik auf niedrigen Schichten nötig ist.
- Die Knappheit der Adressen in IPv4 kann durch den Einsatz des Protokolles zur dynamischen Hostverwaltung (*dynamic host control protocol* DHCP, RFC 2131*) überwunden werden.
- Neben Unicast (Punkt-zu-Punkt-Verbindungen) gemäß IPv4 können durch Einsatz des *Multicast Backbone Network* (MBONE, RFC 1112*) auch Mehrpunktverbindungen unterstützt werden.
- Fortschritte in der Netztechnologie lassen auch für die Zukunft starke Steigerungen der Leistungsfähigkeit von paketvermittelten Netzen erwarten
- Intensiv wird zur Zeit die Standardisierung von verbesserten Protokollen zur Unterstützung von Multimediadiensten und des Netzmanagements vorangetrieben. (Siehe dazu auch den nächsten Abschnitt).
- Existierende realzeitorientierte Dienste im Internet wie die für virtuelle Unterhaltungen (Chat, Talk), vernetztes Spielen (Morgengrauen, MUD) oder Steuerung entfernter Rechner (rlogin, telnet) werden verbreitet und erfolgreich eingesetzt.

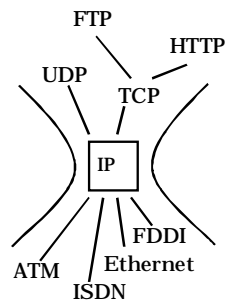


Abbildung 5: Sanduhrmodell des „Internetworking“

Der heutige Stand der Technik stellt dabei aber folgende Hürden für Applikationen zur realzeitorientierten Videokommunikation dar:

- Nur geringe Unterstützung von Querschnittsfunktionen für optimale Dienstgüte (wie Fehlertoleranzstrategien) und damit nur geringe Steuerbarkeit der Dienstgüte im Einsatz.
- Zu geringe Dienstgüteunterstützung bei Kommunikationsbeginn (nur *best-effort*) und damit unvorhersehbare (nichtdeterministische) Fehlerprofile im Einsatz.
- Durch Aufsetzen auf verschiedenen (und teils alten) Techniken auf unteren Schichten empfindliche unkorrigierbare Effizienzverluste, die zur strukturell bedingten Nichteinsetzbarkeit der Applikation führen, weil die Mindestdienstgüte nicht erreicht wird.
- In einem globalen dezentral verwalteten Netz kann es keine allgemeingültigen Modelle für die auftretende Last geben.
- Heute wird der Paketverkehr im Internet durch vom WWW-Protokoll HTTP erzeugten TCP-Verkehr dominiert [Ste 1998].

3.2.3 Weiterentwicklung des Internet

Folgende Standards sind in der Entwicklung und Erprobung um realzeitorientierte Kommunikationsdienste in Zukunft zu ermöglichen:

- IP Version 6 (IPv6, RFC 2460*) soll die Schwächen des alten IPv4 beheben, die in heutigen Implementationen durch DHCP und MBONE verdeckt werden.

* Siehe „Auswahl einiger Internetstandards und deren Status“ im Anhang

* Siehe „Auswahl einiger Internetstandards und deren Status“ im Anhang

- Zeitgleich soll das alte Internet Control Message Protocol (ICMP, RFC 792*) durch ICMP Version 6 (ICMPv6, RFC 2463*) ersetzt werden, um eine schichtübergreifende Steuerung der Dienste zur Laufzeit besser zu ermöglichen.
- Das *Resource Reservation Protocol* (RSVP, RFC 2205*) soll die Reservierung von Ressourcen beim Aufbau einer virtuellen Verbindung ermöglichen.
- Das *Session Initiation Protocol* (SIP, RFC 2543*) soll Netzmanagement durch Zuordnung einer virtuellen Verbindung zu einem bestimmten Typ ermöglichen.
- Das *Session Description Protocol* (SDP, RFC 2327*) arbeitet mit SIP zusammen. Es ermöglicht Applikationen, Sitzungstypen zu definieren.
- Das *Real-Time Transport Protocol* (RFC 1889*) soll u.a. die Kapselung eines Stromes von Mediendaten (*Application Level Framing*) normieren. Es enthält das *Real-Time Transport Control Protocol* (RTCP), das die Qualität des Transportservices zur Laufzeit beobachtet und Informationen über die Teilnehmer einer Sitzung befördert und damit ein Sitzungsmanagement ermöglichen soll. RTP-kompatible Applikationen müssen
 - einen Typ für die Art des übertragenen Datenstromes angeben, das sogenannte *RTP-payload* für RTP. Im November 1999 sind bereits 9 Nutzlasten normiert. (Siehe „RTP-Nutzlasten (RTP-payloads) im November 1999“ im Anhang.)
 - einen Typ der Sitzung für RTCP angeben, das sogenannte *RTP-profile*. Im November 1999 ist eines für Sitzungen mit minimaler Kontrolle (RFC 1890*) normiert.

RTP garantiert keine Dienstgüte, sondern unterstützt nur die paketorientierte Verwaltung einer Sitzung, die Mediendatenströme erzeugt. Die Sicherstellung von rechtzeitiger Zustellung und angemessener Dienstgüte wird unterliegenden Diensten überlassen. [Sea 1996]

3.2.4 Fehlertoleranz im Internet

Das schichtenübergreifende Management des Internet ist unterentwickelt. ICMP-Meldungen zu generieren und zu bearbeiten ist in den Spezifikationen nur optional. Darum gibt es auch keine Fehlertoleranzstrategie im Internet. Auf manchen Schichten arbeiten Fehlertoleranztechniken (z.B. bei TCP), allerdings ohne globale Steuerungsmöglichkeiten.

Die verbreiteten Netztechnologien betreiben automatische Bitfehlerkorrektur auf der Sicherungsschicht. Dadurch werden ohne große Verzögerung und ohne großen lokalen Ressourcenverbrauch viele Bitfehler behoben. IPv4 verwirft ohne Meldung fehlerhaft empfangene Sendungen. Außerdem werden große Pakete mit hoher Wahrscheinlichkeit beim Sender von IP in viele kleine zerteilt (*Fragmentation*), um Netze auf Schicht eins und zwei nutzen zu können, die nur kleine Pakete erlauben (z.B. Zellen zu je 53 bit bei ATM). Beim Empfänger bewirkt dann der Verlust eines einzelnen Fragmentes, daß das ganze Paket nicht defragmentiert werden kann und verworfen wird. Diese Fehlerfortpflanzung kann durch die Wahl kleiner Pakete verhindert werden. Mit kleineren Paketen wird zwar die Häufigkeit verringert, mit dem aus einem Fragmentverlust ein Paketverlust entsteht, andererseits entsteht damit eine höhere Verzögerung (*delay*) bei der Übertragung, weil das verbindungslos arbeitende IP bei der Weiterleitung (*Routing*) von jedem Paket das Ziel neu ermitteln muß durch Verarbeitung des Headers. Diese erhöhte Verarbeitungslast belastet besonders ältere Vermittlungsrechner (*router*) und könnte zu Überlast (*router congestion*) führen. UDP verwirft ohne Meldung fehlerhaft angekommene Pakete. TCP verwirft ohne Meldung fehlerhaft angekommene Pakete und fordert uhrgesteuert vom Sender eine Kopie des Paketes an, bis das Paket korrekt empfangen wurde. Im Gegensatz zu UDP paßt TCP Paketlängen und Zeiteinstellungen der beobachteten Leistungsfähigkeit des Netzes an. IPv4, TCP und UDP werfen ohne Meldung korrekt ankommende Pakete, wenn der lokale

Speicher zur Zwischenspeicherung der Pakete voll ist. (*buffer overflow, router congestion*)

3.3 Medienstandards für realzeitorientierte Videokommunikation

In diesem Kapitel wird auf Medienstandards eingegangen, die heute auf realzeitorientierte Videokommunikation über Internet anwendbar sind. Schwerpunktmäßig wird dabei auf Normen zur Bewegtbildbearbeitung eingegangen. Am Ende werden Anforderungen der Standards an Fehlertoleranztechniken definiert.

Applikationen verschiedener Hersteller arbeiten auf Basis internationaler Standards zur Datenerzeugung, -verarbeitung und -präsentation zusammen. Diesen Standards zu folgen ist besonders für Applikationen zur Kommunikationsunterstützung wichtig, damit möglichst viele Anwender miteinander kommunizieren können. Heutige Standards zur Bewegtbildbearbeitung basieren auf JPEG-Kodierung, auf die darum zuerst eingegangen werden soll.

3.3.1 Motion-JPEG

Die *Joint Photographic Experts Group* der ISO (*International Standardization Organization*) schuf 1988 einen Standard (DIN EN ISO/IEC 10918) zur Komprimierung digitaler Einzelbilder, der über Kodierungsoptionen mehrere Algorithmen vereint. JPEG arbeitet transformationsbasiert, d.h. die digitalen Bildinformationen werden zunächst in einen anderen Raum transformiert, in dem die Daten stärker korreliert sind. Anschließend wird quantisiert, d.h. die Daten werden so verkürzt, daß nur die für die Wahrnehmung wichtigsten Informationen übrigbleiben. Abschließend werden die Daten mit einem Entropiekodierer möglichst kompakt repräsentiert. JPEG arbeitet auf digitalen Abtastwerten für jeden Bildpunkt und für jede Farbdimension. Gute Kompression werden für Bilder von realen Szenen erreicht, für Bilder nicht-realer Szenen wie Zeichnungen, Text, Bildschirmfotos (*screenshots*) oder Overhead-Folien ist JPEG nicht geeignet. Genauere Einführungen in JPEG liefern Gibson [Gea 1998, Chp. 9] und Froitzheim [Fro 1997]. Abbildung 6 zeigt schematisch den Ablauf des Standard-JPEG-Codecs (*baseline-algorithm*).

Für die Kodierung von Videos werden heute JPEG-ähnliche Verfahren angewendet, deren einziger Parameter ein Wert für die Quantisierung ist. Dieser beschreibt eine Kennlinie, die benutzt wird um die ermittelten Koeffizienten so zu filtern, daß die für die menschliche Wahrnehmung (erfahrungsgemäß) schlecht wahrnehmbaren Frequenzen nicht mit kodiert werden bzw. nur durch kleine Werte repräsentiert werden. Ein höherer Quantisierungsfaktor resultiert in geringerer Qualität und höherer Kompression (d.h. einem verschwommeneren Bild und weniger resultierenden Daten), ein geringerer Quantisierungsfaktor führt entsprechend zu höherer Qualität und geringerer Kompression. Die resultierende Liste von Koeffizienten ist geordnet, d.h. je weiter links ein Wert in der resultierenden Liste steht, desto wichtiger ist er für die Qualität des dekodierten Bildes. Wegen der Unschärfe des Qualitätsbegriffes wird häufig auf eine genaue Klassifikation der Koeffizienten verzichtet und es wird nur willkürlich in wichtige hohe Koeffizienten und weniger wichtige niedrige Koeffizienten unterteilt.

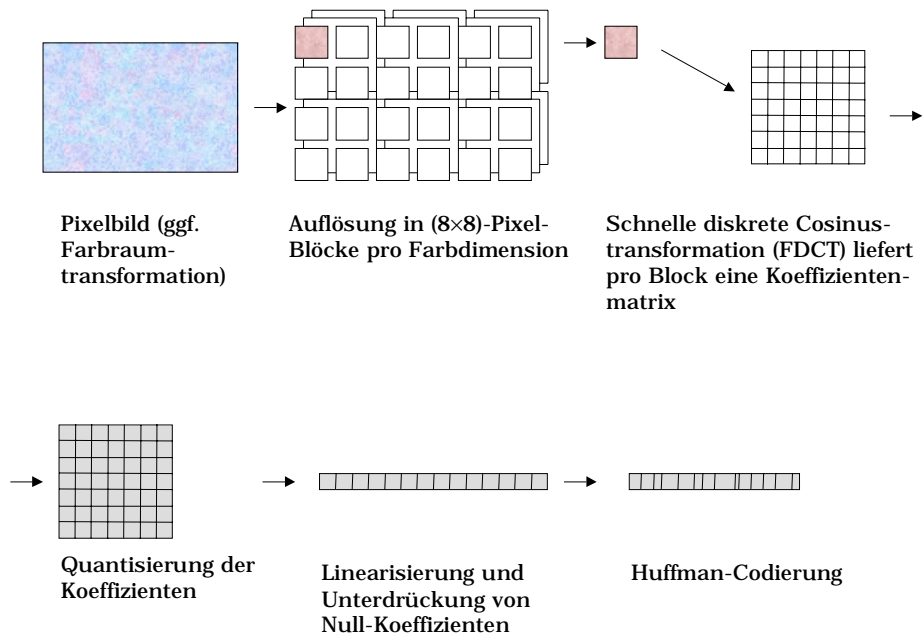


Abbildung 6: Schematischer Ablauf der JPEG-Kodierung

JPEG wurde unter der Vorgabe möglichst hoher Kompression von Bildern von realen Szenen unter möglichst geringem Qualitätsverlust entwickelt und hat sich wegen seiner Effizienz heute durchgesetzt. Kein anderes standardisiertes Verfahren erreicht heute bei Übertragung über sichere Kanäle bei so geringer Datenrate eine so gute Präsentationsqualität.

Folgendes Szenario macht eine weitere Eigenschaft der JPEG-Kodierung deutlich, die für realzeitorientierte Applikationen wertvoll ist: Angenommen, eine Menge Bilder wird nacheinander JPEG-kodiert (als *motion-jpeg*) und über einen sicheren Kanal mit träge schwankender Bandbreite zu einem Empfänger übertragen. Der Empfänger mißt die Bandbreitenschwankungen und sendet ein Modell über den zukünftig möglichen Durchsatz zum Sender zurück. Dieser variiert anhand dieser Informationen die Quantisierung dergestalt, daß das gleichmäßige Eintreffen der kodierten Bilder beim Empfänger eher gewährleistet ist als bei konstanter Quantisierung. Dieses Vorgehen stellt eine Fehlertoleranzstrategie für motion-jpeg dar: Obwohl der Kanal sicher ist, gehen für den Empfänger aufgrund der Bandbreitenschwankungen bei konstanter Quantisierung Bilder verloren, weil sie zu spät eintreffen, um noch rechtzeitig präsentiert werden zu können. Der Betrachter empfindet das Fehlen der Bilder als ruckelnde Bewegung. Werden durch variable Quantisierung Bildverluste durch verspätetes Eintreffen ausgeschaltet, so empfindet der Betrachter die Durchsatzschwankungen als Schwankung in der Bildschärfe. Diese Technik behebt keinen Fehler, sondern verarbeitet die Leistungsschwankungen des Kanals lediglich anders, indem Zucken durch Schwankungen der Bildschärfe getauscht wird. Es ist aber zu erwarten, daß Schwankungen der Bildschärfe vom Betrachter als nicht so gravierende Qualitätsstörung erfahren werden wie Zucken. Das legen Erfahrungen aus dem Bereich der Funktelefonie nahe, wo diese Technik des „sanften Qualitätsabfalles“ (*graceful degradation*) z.B. in GSM standardisiert ist. Grundsätzlich wird bessere Robustheit gegen Fehler mit geringerer Kompressionseffizienz erkaufte.

Ein zweiter wichtiger Aspekt ist der des sanften Qualitätsabfalles bei unvollständiger Dekodierung: Wenn ein Bild nur mit den wichtigen hohen Koeffizienten jedes Blockes dekodiert wird, so ist die Qualität zwar deutlich geringer, aber der Inhalt des Bildes noch erkennbar. Diese Eigenschaft wird bei neueren Ansätzen genutzt, die versuchen, JPEG-basierte Codecs fehlerrobuster zu machen. Auf diesen Aspekt wird später ausführlicher eingegangen. Das Übertragungsformat für einzelne JPEG-kodierte Bilder ist im JFIF (*JPEG File Interchange Format*) standardisiert. Den um Steuerdaten ergänzten komprimierten Daten wird ein Header vorangestellt mit Größeninformationen, Quellformat, Quantisierungs- und Huffmantabellen. Weil die komprimierten Daten von variabler Länge sind und nur in eine Richtung entschlüsselt werden können, kann ein einzelner Bitfehler dazu führen, daß der Dekoder für längere Zeit unsynchronisiert ist (bis zufällig das nächste gültige Kodewort gefunden wird), so daß ein Bitfehler zum Verlust ganzer Bilder führen kann (*error propagation*). Nachdem Resynchronisation erfolgt ist, ist mindestens ein Block verlorengegangen, so daß dem Betrachter quadratische Ausschnitte im Bild fehlen. Verlust ganzer Bitfolgen (*bursts*) hat genau dieselbe verheerende Wirkung. JPEG als Teil der Quellkodierung ist nicht fehlerrobust und JFIF als Kanalkodierung für ein ganzes Bild ebenfalls nicht. Weil bei der Standardisierung Kompression im Vordergrund stand, sind aufwendige technische Maßnahmen nötig, wenn JPEG-kodierte Bilder über unsichere Kanäle transportiert werden sollen.

Heute lassen sich dreidimensionale Signale nicht computertechnisch verarbeiten. Die Illusion von bewegten Bildern wird durch Ausnutzung der Trägheitseigenschaft des menschlichen Auges erzeugt, indem einzelne, sich sehr ähnliche Bilder in so schneller Folge dem Betrachter präsentiert werden, daß im Gehirn der Eindruck einer Bewegung entsteht. Wird jedes Bild JPEG-kodiert, so wird die räumliche Redundanz aus jedem Bild entfernt, d. h. vorher vorhandene Korrelation wird aus den Daten entfernt. Bildfolgen enthalten aber auch zeitliche Redundanz, weil sich benachbarte Bilder in der Regel sehr ähnlich sind (wenn zwischen den Bildern kein Szenenwechsel liegt). Diese Redundanz wird durch vorausschauende Kodierung (*predictive coding*) entfernt, indem geprüft wird ob eine Transformation der Differenz des aktuellen Blocks mit dem Block des vorhergehenden (oder folgenden) Bildes weniger Daten liefert als eine absolute Transformation. Dieses ist in der Regel der Fall. Absolut kodierte Blöcke werden I-Blöcke genannt, weil die Kodierung im Intramodus (als ohne Differenzbildung) stattgefunden hat. Einseitig aus Differenzen kodierte Blöcke werden P-Blöcke genannt, zweiseitig aus Differenzen kodierte Blöcke werden B-Blöcke genannt. Wenn ein Bild oder eine Bildscheibe mindestens einen differenzenkodierten Block enthält, wird es B- bzw. P-Bild genannt. I-Bilder bestehen nur aus I-Blöcken.

Wenn diese Vorhersage-Kodierung auch innerhalb eines Bildes angewandt werden soll, so wird für jeden Block geprüft, ob es günstiger ist, statt absolut zu kodieren nur die Informationsdifferenz zu einem (benachbarter) Block in demselben Bild zu komprimieren und zusätzlich einen Vektor zu speichern, der angibt, von welchem 8x8-Pixel-Bildausschnitt die Differenz gebildet wurde. Dieses Vorgehen heißt Bewegungersatz (*motion compensation*). Dieser Name ist aber irreführend, weil es in dem Verfahren NICHT darum geht, sich bewegende Objekte zu modellieren, sondern nur um eine weitere Verringerung der Datenmenge.

Bei der JPEG-ähnlichen Kompression von Bildfolgen werden aus jedem Block folgende Daten erzeugt:

- Information über den Kodiermodus (*coding mode*), d.h. ob absolut (Intramodus) oder mit Bezug kodiert wurde.

- bei Bezug eine Referenz darauf, von welchem Block die Differenz gebildet wurde. (*motion vector*)
- Liste geordneter DCT-Koeffizienten. Die vorderen werden als wichtiger angesehen als die hinteren.

Nun werden Normen für Kodierer und Dekodierer (Bildfolgencodecs) vorgestellt, die diese Techniken mit jeweiligen Eigenheiten benutzen. Diese Standards sind H.261, H.263+ und MPEG-4. Der erstere ist in die Standardfamilie H.323 eingebettet, die alle Dienststandards der ITU-T für realzeitorientierte Videokommunikation über Internet umfaßt.

3.3.2 H.261

Die *International Telecommunication Union – Telecommunication Standardization Sector (ITU-T)* ging aus der CCITT hervor und verwaltet zahlreiche Standardfamilien. Für Videokonferenzen über Internet wurde 1996 der Standard H.323 verabschiedet. Er umfaßt u.a.:

- H.255.0 als Standard für das Multiplexen der Datenströme;
- H.245 für Verbindungsauf- und abbau und Steuerung;
- G.711 verbindlich für Sprachdaten, 4 weitere optional;
- H.261 verbindlich für Bilddaten, den jüngeren H.263 optional;
- T.123 zur gemeinsamen Benutzung von Datenobjekten (*application sharing*).

Ausführlicher zu den Standards schreibt Gibson [Gea 1998, Chp.10]. Übertragungsdienste werden in H.323 nicht standardisiert, der Standard wurde aber mit Blick auf die Verwendung des Internetprotokolles RTP entworfen.

H.261 wurde 1993 von der ITU-T zur Verwendung mit S-ISDN normiert. Seit 1996 existiert eine Spezifikation zur Nutzung im Internet. Neben einer Bildebene gibt es in H.261 eine Blockgruppen-Ebene (GOB, *group of blocks*) und eine Makroblock-Ebene. Präsentationsformate sind CIF (*common intermediate format*, 352x288 Punkte) und QCIF (*quarter CIF*, 176x144 Punkte). Ein Makroblock besteht aus vier Blöcken mit Helligkeitsinformationen und zwei Blöcken mit Farbinformationen. Farbe mit der halben Auflösung wie Helligkeit kodiert. Das liegt daran daß das menschliche Auge Helligkeit besser auflösen kann als Farbinformationen. Regelmäßig wird das Übertragen von intrakodierten Stützblöcken empfohlen (vorgeschlagen wird alle 132 Bilder). Zwischenbilder können nur aus Differenzen mit dem vorhergehenden Bild gebildet werden. Am Beginn der Kodierung wird eine konstante Bitrate vorgegeben (als Vielfaches von 64 Kbit/s, daher auch der Name px64 für diesen Codec), diese Bitrate wird zur Laufzeit durch Anpassung der Quantisierung oder notfalls durch Weglassen ganzer Bilder strikt eingehalten. Globale Schutzmaßnahmen über Pakete fester Länge sind als Option standardisiert.

Die Kombination von nur einseitig entschlüsselbarer Kodierung variabler Länge und zusätzliche Differenzenkodierung verschärft die schon bei JPEG angesprochenen Anfälligkeiten gegen Übertragungsfehler. Im Standard wird die Verwendung einer blockorientierten BCH(511, 493)-Kodierung zur Vorwärtsfehlerkorrektur als Option vorgesehen, die auf 493 Bits 18 Prüfbits generiert. Diese Maßnahme zur Bitfehlerkorrektur scheint ausreichend für den Einsatz über ISDN, greift im Internet aber zu kurz. Das liegt daran, daß Daten in ISDN verbindungsorientiert und synchron als Bitstrom übertragen werden. Der Bitstrom wird durch Rahmen strukturiert. Bei der Übertragung treten nur gering Bitfehler auf. Im Internet dagegen wird verbindungslos und asynchron übertragen. Vermehrt wird der Verlust ganzer Pakete beobachtet. Die Länge des verlorenen Paketes ist unbekannt und die Struktur des Inhalts kann vom Empfänger nicht automatisch erkannt werden, sondern muß algorithmisch bekannt gemacht werden. [Fro 1997, Kap. 2.3.2, ITU 1993]

3.3.3 H.263+

H.263 stellte 1996 eine Weiterentwicklung von H.261 dar und ist zur Benutzung über analoge Telefontechnik oder Internet gedacht. Bei annähernd gleicher Bildqualität wie H.261 liefert diese Norm etwa die halbe Datenmenge. Dieses wird vor allem auf den Bewegungsausgleich mit halber Pixelgenauigkeit zurückgeführt [HKZ 1999]. Seit 1996 wurde der Standard um einige Anhänge ergänzt, die optionale Betriebs-Modi enthalten um die Fehlerrobustheit zu erhöhen. 1998 wurde „H.263 Version 2“ verabschiedet. Dieser Standard wird wegen der großen Anzahl von Anhängen (20 Stück) auch H.263+ genannt. Damit hat sich H.263 ähnlich wie JPEG zu einem „Menü von Algorithmen“ weiterentwickelt. Hier werden die Betriebsmodi vorgestellt, die in den Anhängen K, D, R und N standardisiert sind.

H.263+ unterstützt das Einfügen von Synchronisationswörtern alle n Blockgruppen innerhalb eines Bildes (Anhang K des Standards). Die Daten zwischen zwei Synchronisationswörtern werden Bildscheiben (*slices*) genannt. [Kam 1998, Chp. 4] Anhang D des Standards unterstützt die Kodierung der Bewegungsvektoren mit einem reversiblen Kode variabler Länge (*Reversible Variable Length Code - RVLC*), um bei Verlust von Synchronisation rückwärts dekodieren zu können. [Kam 1998, Chp. 4]

Anhang R beschreibt Unabhängiges Segment-Kodieren (*independent segment decoding - ISD*). Dieser Modus kann nicht gleichzeitig mit dem Bildscheiben-Modus betrieben werden. Beim unabhängigen Segment-Kodieren wird jede Blockgruppe als eigenständiges „Subvideo“ behandelt, d.h. Grenzen der Blockgruppen werden wie Bildgrenzen behandelt. Dadurch daß Bewegungsausgleich nur innerhalb der Grenzen einer Blockgruppe erfolgt, sinkt die Effizienz, Fehlerfortpflanzung kann aber nicht aus einer Blockgruppe hinaus erfolgen, wird also begrenzt. Anhang N beschreibt den Referenzbild-Auswahlmodus (*Reference Picture Selection*). Sender und Empfänger verständigen sich zur Vermeidung von Fehlerfortpflanzung über einen Rücklaufkanal auf ein Bild, was beim Empfänger fehlerfrei angekommen ist als Referenzbild für Differenzenkodierung.[FGV 1998]

3.3.4 MPEG-4

Parallel zur ITU-T standardisiert die *Moving Pictures Expert Group* (MPEG) der ISO/IEC Audio- und Videodaten. Während ITU-T-Standards die technisch unterliegende Infrastruktur bewußt mit einbeziehen und so auch Verfahren standardisieren, die im Betrieb unter Umständen nicht die minimal geforderte Dienstgüte erbringen und damit Systeme zur „Benutzung im Notfall“ realisieren, konzentrierte sich MPEG auf applikationsnahe Standards zur Lieferung höchster Qualität. Gerade im Audibereich liefern ITU-T-Standards selten mehr als Sprachqualität (sie sind wie alte Telefondienste auf eine Bandbreite von 1000-4000 Hz beschränkt) während MPEG den gesamten hörbaren Bereich von 50-10000Hz digitalisiert und auch Stereoausgabe vorsieht und so „like-to-use“-Systeme realisiert. Während die älteren Standards MPEG-1 und MPEG-2 im Hinblick auf bestimmte zukünftige Anwendungen erarbeitet wurden, soll der neuere MPEG-4-Standard eine ganze Bandbreite von Anwendungen unterstützen. Im Gegensatz zu mit den älteren Standards kodierten Datenströmen, die eine Datenrate von 1,5Mbit/s bzw. zwischen 4 und 10Mbit/s erzeugen, können MPEG-4-kodierte Datenströme eine Datenrate von unter 64Kbit/s erreichen. Damit wird MPEG-4 für realzeitorientierte Videokommunikation interessant.

Die produzierte Datenrate ist grundsätzlich variabel, allerdings wird die Kompressionsrate im Kodierer so angepaßt, daß vorgegebene Höchstwerte nicht überschritten werden. Gegenüber den älteren Standards erlaubt MPEG-4 die

Kodierung willkürlich geformter Bildobjekte als Subvideo und die Komposition dieser Objekte beim Empfänger. Im Modus „fehlerrobuster Betrieb“ sind MPEG-4-kodierte Daten fehlerrobuster als H.263+-kodierte Daten. Konsequenter als bei H.263+ werden rückwärtsdekodierbare Kodewörter variabler Länge benutzt (RVLC). Kamosa macht aber auch bei MPEG-4 noch Schwächen in Bezug auf Rückwärtsdekodierbarkeit aus und entwirft einen eigenen Vorschlag für einen zukünftigen Codec. [Kam 1998]

Talluri [Tal 1998] hebt zwei weitere Fehlertoleranzoptionen hervor. Zum einen führt er die Möglichkeit an, Wörter zur Resynchronisation (*synchronisation marker*) in regelmäßigen Abständen (alle k Bits) im kodierten Datenstrom einzufügen und so schnellere Resynchronisation zu ermöglichen als wenn die Resynchronisation an Blockgruppen gekoppelt ist und damit nur in variablen Abständen erfolgen kann. Zum anderen nennt er die Trennung von Bewegungs- und Musterdaten (*data partitioning*). Im Gegensatz zu anderen Codecs werden hier die wichtigeren Bewegungsdaten (die Vektoren zum Bewegungsausgleich) von den weniger wichtigen Musterdaten (Farbinformationen) getrennt, so daß bei einem Fehler in den Musterdaten wenigstens noch die Bewegungsdaten verwendet werden können. Damit wird dem Umstand Rechnung getragen, daß typische Bildblöcke in Videokonferenzen in hohem Maße so ähnlich aussehen wie ein benachbarter Block und nur in geringem Maße neue Informationen enthält. Dadurch kann mit Hilfe der Bewegungsdaten eine bessere Fehlerbehebung erreicht werden als mit den Farbinformationen.

4 Fehlertoleranz

Zunächst wird Fehlertoleranz allgemein definiert, dann wird der Begriff auf die Handhabung von Übertragungsfehlern in Kommunikationsnetzen eingeschränkt.

4.1 Allgemeine Fehlertoleranz

Nach Görke ist Fehlertoleranz „die Fähigkeit eines Systems, auch mit einer begrenzten Zahl fehlerhafter Subsysteme seine spezifizierte Funktion erfüllen zu können.“ [Gör 1989, S.11] Unter einem System wird dabei ein Rechensystem verstanden. Subsysteme könnten z.B. eine Anwendungsfunktion, die Prozeßverwaltung oder das Kommunikationsmedium sein. Fehlertoleranz ist ein Mittel zur Unterstützung der Zuverlässigkeit des Systems. (Abbildung 7) Sie kann „nur durch Anwendung nützlicher Redundanz erzielt werden. Nichtredundante Systeme erreichen ihre Zuverlässigkeit nur durch Vermeidung von Fehlern.“ [Gör 1989, S.22]

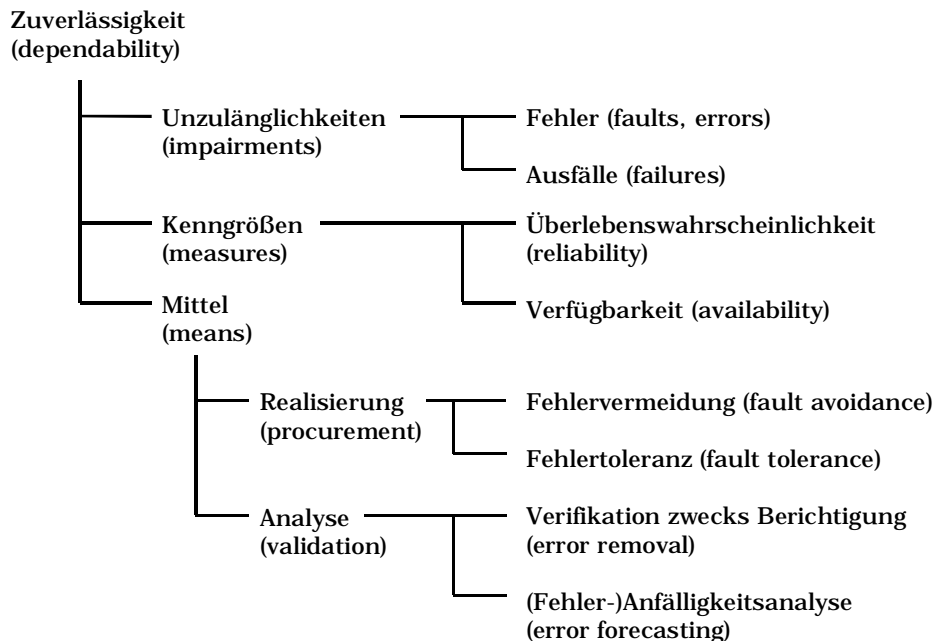


Abbildung 7: Begriffsgliederung zur Fehlertoleranz (aus [Gör 1989, S.26]).

Fehler können Entwurfsfehler, Herstellungsfehler und Betriebsfehler sein. [Gör 1989, S.35] Zu den Anwendungsbereichen fehlertoleranter Rechensysteme stellt Görke fest „Im Normalfall sind die Anforderungen an Rechensysteme von den Anwendungsbereichen abhängig, in denen das fehlertolerante Rechensystem eingesetzt werden soll. Diese Anwendungsbereiche lassen sich auf ganz unterschiedliche Weise einteilen.“ [Gör 1989, S.16] Neben einer Unterteilung nach den Zuverlässigkeits- und Wartbarkeitskriterien stellt er eine Einteilung nach „gegenwärtigen Einsatzbereichen der Systeme“ vor, der hier gefolgt werden soll. Es werden unterschieden:

- Realzeitsysteme (zur Steuerung und Überwachung technischer Systeme);
- Vermittlungssysteme;
- Transaktionssysteme (Steuerung von Zugriffen auf Datenbanken);
- Dienstleistungsrechner in lokalen Netzen.

Echtle bemerkt: „Neu hinzukommende Anwendungsbereiche und Rechensystemumgebungen verlangen angepaßte Fehlertoleranz-Verfahren. Diese weichen von den bisherigen nicht grundsätzlich ab, beziehen sich aber auf neuartige Komponenten-

mengen, Fehler-, Lokalisierungs- und Behandlungsbereiche.“ [Ech 1990, S.302] Darüber hinaus stellt Görke fest, daß Redundanz, Fehlerdiagnose und Fehlerbehebung erforderlich sind, um Fehlertoleranz zu erreichen. [Gör 1989, S.27]

Der Begriff der Fehlertoleranz wird im Zusammenhang mit Kommunikation in einem „Schichtenmodell für Mehrrechnersysteme“ erläutert, indem Subsysteme zur Funktionserbringung in sechs Schichten angeordnet werden. Als Redundanzarten, die bei Realisierung von Fehlertoleranz miteinander verzahnt sind unterscheidet Görke:

- strukturelle Redundanz (Erweiterung des Systems um Reserve-Objekte);
- funktionelle Redundanz (Erweiterung des Systems um zusätzliche Funktionen);
- Informationsredundanz (redundante Kodierung der Daten);
- Zeitredundanz (Vorsehen von mehr Rechenzeit zur Funktionserbringung).

Fehlertoleranz kann also durch Einbettung von Redundanz in das System erreicht werden. In welchem Umfang Redundanz vorgesehen wird hängt vom Fehlermodell ab, welches für das System festgelegt wird. „Das Fehlermodell macht eine Annahme über Art, Anzahl und Möglichkeiten auftretender Fehler“. Es sollte „realistisch im Hinblick auf die Anwendungsumgebung des Rechnersystems sein und dazu dienen, zu jedem möglichen Zeitpunkt die Unzulänglichkeiten eines Rechnersystems zu beschreiben.“ [Gör 1989, S.34] Er betont, daß Fehlertoleranzmaßnahmen grundsätzlich nicht alle Fehlermöglichkeiten gleichzeitig tolerieren können. Vielmehr seien folgende Fragen zu berücksichtigen [Gör 1989, S. 37]:

- Gegen welche Fehler lassen sich Fehlertoleranzmaßnahmen einrichten?
- Welche Fehler lassen sich bei gegebenem Aufwand optimal tolerieren?
- Wie lassen sich aufgetretene Fehler feststellen und identifizieren?

Zum Entwurf von Fehlertoleranz-Verfahren merkt Echte an: „Übersicht über die Realisierungsmöglichkeiten kommt deshalb eine besondere Bedeutung zu, weil die Hauptprobleme bei der Schaffung ‚guter‘ Fehlertoleranz-Verfahren weniger im Finden völlig neuer Ansätze als vielmehr in einer günstigen Auswahl und in der Verfeinerung bekannter Vorgehensweisen liegen, um die üblicherweise vielfältig vorliegenden Anforderungen nach Effizienz, Rechnersystem- und Anwendungs-Verträglichkeit zu erfüllen. Aus Gründen der Effizienz und/oder besonderer Fehlervorgaben bietet sich häufig die Kombination unterschiedlicher Fehlertoleranz-Verfahren in verschiedenen Subsystemen an.“ [Ech 1990, S.271]

Die Fehlerbehebung kann nach Echte auf drei Weisen erfolgen:

- Rückwärtsbehebung, bei der das System in einen Zustand gebracht wird, den es in der Vergangenheit angenommen hat oder hätte einnehmen können;
- Vorwärtsbehebung, die zu irgendeinem zulässigen Zustand führt, indem Wissen über die Anwendung benutzt wird, um einen Schätzer für verlorene Werte aufgrund von aktuellen Werten zu finden;
- Fehlerkompensierung, bei der das System im fehlerhaften Zustand belassen wird, zukünftige Ergebnisse aber auf fehlerfreie Ergebnisse abgebildet werden, indem Wissen über die Art des Fehlers benutzt wird. (Beispiel: 2-aus-3-Systeme, deren Ergebnisse als korrekt angenommen werden, wenn 2 der Systeme dasselbe Ergebnis liefern).

Echte postuliert: „Fehlertoleranz sollte eine rein rechnerspezifische und keine anwendungsspezifische Eigenschaft sein, so daß sich der Benutzer nicht weiter mit ihr befassen muß.“ [Ech 1990, S.297] Diese Forderung wird eine Seite weiter als „Unabhängigkeit gegenüber der Anwendung“ bezeichnet. Etwas weiter wird darauf hingewiesen, daß diese Forderung bei Vorwärtsbehebung nicht erfüllbar sei und bei anderen Verfahren z.T. zu Lasten der Effizienz ginge.

Echtle kümmert sich auch um den Begriff der Rekonfiguration. Rekonfiguration bedeutet die Ausgrenzung fehlerhafter Subsysteme. Ausgrenzung bedeutet, daß ihre Funktionen in Zukunft nicht mehr beansprucht werden. Sie dient der Behandlung von Funktionsausfällen, nicht der Behebung von Fehlern.[Ech 1990, Kap. 5]

4.2 Fehlertoleranz und Videokommunikation

Der Anwendungsbereich für realzeitorientierte Videokommunikation wurde bereits im vorigen Kapitel umrissen.

Systeme, die Anwendungen zur realzeitorientierten Videokommunikation unterstützen, werden hier den Realzeitsystemen zugeordnet, weil Rechtzeitigkeit deutlich wichtiger ist für die Anwendung als die Vermittlungsfunktion, die für den Start der Kommunikation ebenfalls zuverlässig erbracht werden muß.

Ein Rechensystem im oben angegebenen Sinn ist die Gesamtheit aus den Applikationen der Teilnehmer, den Endsystemen, auf denen die Applikationen laufen sowie das Kommunikationssystem. Im Kontext der Kommunikation sind als Subsysteme die Dienste zu identifizieren, die „Erfüllung der Funktionen“ ist dann die Erbringung des Dienstes in einer für die Anwendung nützlichen Form.

Häufig wird im Zusammenhang von Dienstleistung der Begriff der Dienstgüte eingeführt und die „Erbringung des Dienstes in einer für die Anwendung nützlichen Form“ als Erbringung des Dienstes mit mindestens der Mindestdienstgüte beschrieben. Dienstgüte wird dabei definiert als „Summe aller quantifizierbaren Eigenschaften eines Dienstes“ die so konfiguriert werden, daß eine optimale Qualität des Dienstes (QoS, quality of service) erreicht wird. Mindestdienstgüte ist dabei die Konfiguration, die die vom Benutzer gerade noch akzeptierte Dienstqualität liefert. Dürst und Mühlhäuser ordnen Zuverlässigkeit als eine Eigenschaft eines Dienstes und damit als Unterbegriff für Dienstgüte ein [DüM 1999].

In dieser Arbeit wird Fehlertoleranz im Kontext von Übertragungsfehlern untersucht, d.h. der Fall daß falsche Werten in den transportierten Daten (Bitfehlern) auftreten oder Pakete ganz fehlen (Paketverluste). Natürlich entstehen solche Fehler teilweise durch physikalische Einflüsse beim Betrieb, sind also Betriebsfehler; ein nicht unerheblicher Teil entsteht aber auch durch Entwurfsfehler. Ein Beispiel dafür ist z.B. wenn ein Dienst so spezifiziert ist, daß er einerseits dem Benutzer eine Mindestdienstgüte garantiert und dabei andererseits auf untergeordneten Diensten basiert, die ihren Dienst bestmöglich (best-effort) erbringen, d.h. keine Garantien geben. Fehlertoleranz im Kontext von Kommunikation führt zu der bereits im ersten Kapitel beschriebenen Aufteilung in Fehlertoleranzmaßnahmen im Sender und im Empfänger.

Fehlertoleranz kann erreicht werden durch Redundanz, Fehlerdiagnose und Fehlerbehebung [Gör 1989]. Redundanz kann im Zusammenhang mit Übertragungsfehlern verwendet werden in Form von

- Informationsredundanz, d.h. redundanten Datenobjekten, die zusätzlich zu den ursprünglichen Datenobjekten übertragen werden (z.B. Vorwärtsfehlerkorrektur);
- funktionelle Redundanz, d.h. redundante Algorithmen zur zusätzlichen Überwachung (Monitoring) der Übertragung (z.B. Wiederholung auf Anfrage).

Zeitredundanz und strukturelle Redundanz werden in dieser Arbeit nicht betrachtet. Fehlerdiagnose kann

- kurzfristig automatisch erfolgen für gut bekannte Fehler ;
- mittel- und langfristig erfolgen durch Erkennen des Benutzers.

Fehlerbehebung kann

- kurzfristig verlustfrei und automatisch erfolgen, wenn gut bekannte Fehler behandelt werden und eine andere Art der Behandlung die Rechtzeitigkeit nicht gewährleistet;
- kurzfristig verlustbehaftet automatisch erfolgen (Fehlerverschleierung), wenn eine verlustfreie Behandlung die Rechtzeitigkeit nicht gewährleistet und die Verschleierung in Zukunft keine Qualitätseinbuße der Dienstleistung erwarten läßt;
- mittel- und langfristig durch Rekonfiguration des Systems durch Erkennen des Benutzers erfolgen.

Ein Fehlermodell für realzeitorientierte Videokommunikation über Paketvermittlungsnetze ist in der Vergangenheit nicht gelungen wegen des starken Wachstums der Applikationen und der Verbreitung von IT-Systemen, die für die Benutzung für realzeitorientierte Videokommunikation infrage kommen. In dieser Arbeit wird darum der Weg gegangen, für jede einzelne Fehlertoleranztechniken Fehlermodelle zu nennen, die sich eignen bzw. nicht eignen.

Mittel- und langfristige Fehlertoleranz durch den Benutzer steht nicht im Mittelpunkt dieser Arbeit. Diese Art von Fehlertoleranz wird interessant werden, wenn es genügend Erfahrungen mit dem Betrieb von IT-Systemen gibt, die realzeitorientierte Videokommunikation unterstützen (Stichwort Management). Beim heutigen Stand von Technik und Wissenschaft sind automatische Algorithmen zur kurzfristigen Fehlerbehandlung interessant, um realzeitorientierte Applikationen robuster zu machen gegen das unberechenbare Verhalten heutiger best-effort-Kommunikationsnetze und so eine Dienstgüte nicht aufgrund von Garantien oder Verträgen zu erreichen, sondern aufgrund von intelligenter Nutzung zusätzlicher Ressourcen (wie mehr Prozessorzeit oder mehr Übertragungskapazität). Diese Anstrengungen stehen im Zusammenhang mit der Entwicklung (Softwaretechnik). Aspekte des Managements werden in dieser Arbeit nur insoweit behandelt, wie bei der Entwicklung von Applikationen Optionen zum Management zur Laufzeit berücksichtigt werden müssen.

Das Transparenz-Postulat von Echte ist für realzeitorientierte Videokommunikation nicht aufrecht zu erhalten. Zum einen hat es sich gezeigt, daß die von Echte eingeräumten Effizienzverluste beträchtlich sind. Besonders viel Zeit wird verloren, wenn Anwendungswissen nicht genutzt wird. Zum anderen hat die von Echte von diesem Postulat ausgenommene Vorwärtsbehebung im Zusammenhang mit Multimedia erheblich an Bedeutung gewonnen, weil die menschliche Wahrnehmung kleine Fehler gut toleriert und Fehlertoleranz sich heute nicht nur um die exakte Wiederherstellung von Steuerdaten kümmert, sondern auch um wahrnehmbare Daten. Mit abnehmender Transparenz von Fehlertoleranz wächst die Einbindung in einen Anwendungskontext und verallgemeinerte Aussagen lassen sich immer schwerer treffen.

Im Rest dieses Kapitels werden einzelne Techniken vorgestellt, die Bausteine einer Fehlertoleranzstrategie sein können. Im darauffolgenden Kapitel werden Kombinationsmöglichkeiten untersucht. Fehlertoleranztechniken für realzeitorientierte Videokommunikation wird unterteilt in sender- und empfängerbasierte Techniken. Fehlertoleranztechniken werden senderseitig genannt, wenn Redundanz verwendet wird, um die Daten fehlerrobust zu machen. Wird Redundanz aufgrund einer Rückmeldung vom Empfänger gesendet, spricht man von einer aktiven Technik, sonst von einer passiven. Empfängerseitig werden Fehlertoleranztechniken

genannt, die Fehlerverschleierung vornehmen. Verschleierung ist die verlustbehaftete Wiederherstellung verlorener Daten (das Gewinnen eines Schätzers), für deren exakte Wiederherstellung nicht genügend Informationen vorhanden sind oder deren exakte Wiederherstellung vermutlich nicht rechtzeitig möglich ist. Je nach dem algorithmischen Aufwand, der zum Finden des Schätzers betrieben wird, werden Techniken zum einfachen Einfügen, zum geglätteten Einfügen und zur Rekonstruktion unterschieden. Abbildung 8 gibt einen Überblick über die Unterteilung. Die Nummerierung bezieht sich auf die entsprechenden Abschnitte dieser Arbeit.

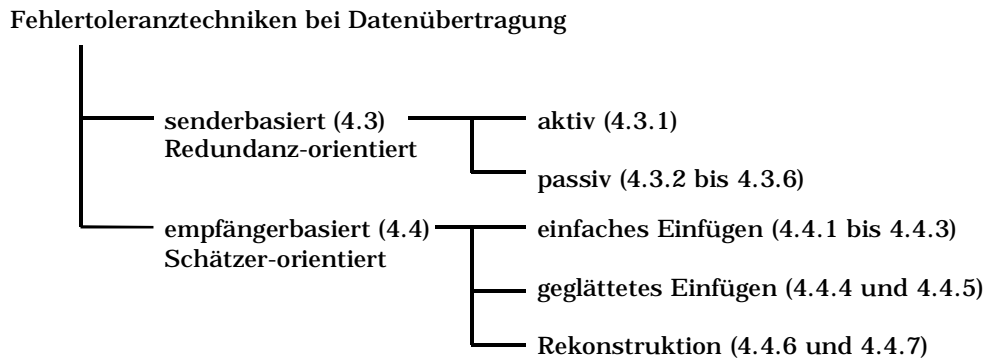


Abbildung 8: Fehlertoleranztechniken bei Datenübertragung.

4.3 senderbasierte Fehlertoleranztechniken

Abbildung 9 gibt eine Übersicht über Fehlertoleranztechniken beim Sender.

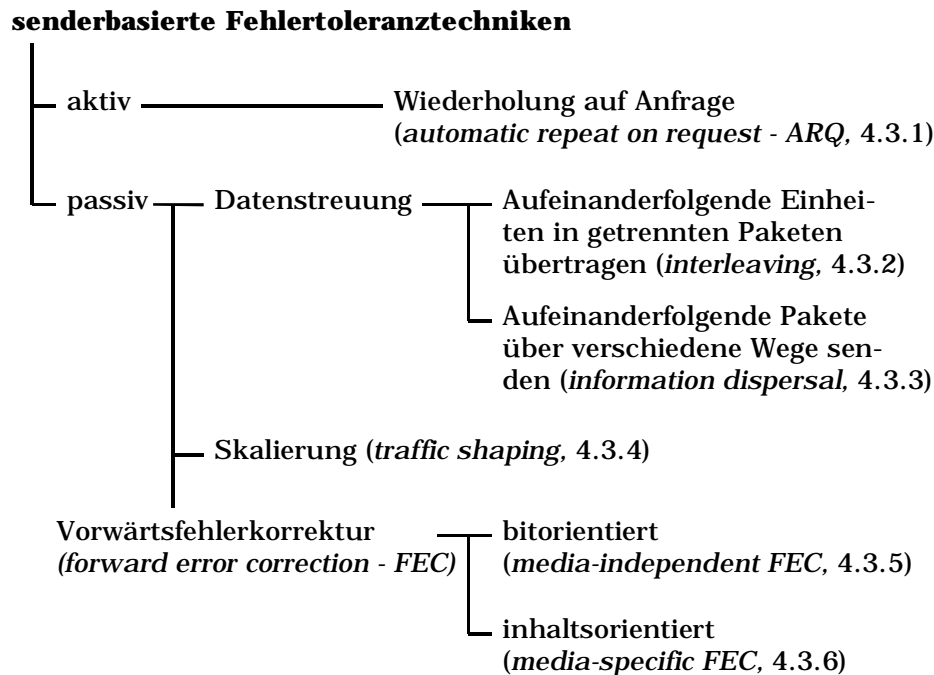


Abbildung 9: Senderbasierte Fehlertoleranztechniken.

4.3.1 Wiederholung auf Anfrage

Dieser Ansatz beruht auf der Idee, dem Empfänger durch eine Nachricht an den Sender die Möglichkeit zu geben, eine Wiederholung einer verlorenen Sendung zu erreichen. Man spricht auch von *aktiver* Fehlertoleranz, weil der Sender erst nach aktiver Anfrage des Empfängers die Wiederholung beginnt.

Mit dieser Technik können nicht nur zuverlässige Verbindungen über unzuverlässige Kanäle realisiert werden (wie bei TCP im Internet der Fall), sondern auch eine Verbesserung der Dienstgüte erreicht werden bei realzeitorientierten Verbindungen über unzuverlässige Kanäle. Dafür sind allerdings Implementationen nötig, die eine Anfrage nur schicken, wenn bestimmte Bedingungen erfüllt sind, die in der Anwendung durch Indikatoren zur Laufzeit gemessen werden müssen. Solche Indikatoren können sein:

- Messungen am Netz, die Informationen darüber liefern, ob es realistisch erscheint, daß die Antwort innerhalb der Realzeitanforderungen eintrifft (Schätzung der Antwortzeit (*round-trip-time*)).
- Vorabschätzung der empfängerseitigen Fehlerbehebung, ob mit einer Wiederholung Informationen eintreffen werden, die bessere Darstellungsergebnisse erwarten lassen, als das aufgrund bereits vorhandener Informationen ohne Wiederholung alleine mit empfängerseitiger Fehlerbehebung möglich wäre.

Daraus wird deutlich, daß diese Technik alleine keine zufriedenstellende Fehlerrobustheit in realzeitorientierten Anwendungen erreichen kann. Eine Implementation braucht zusätzlich eine Technik, die den Verlust behandelt, wenn der Darstellungszeitpunkt für die verlorene Einheit kommt und die Wiederholung nicht eingetroffen ist. Im zweiten Indikator ist bereits angedeutet, daß in realzeitorientierten Anwendungen häufig auf eine Wiederholung einer exakten Kopie des Verlustes aus verschiedenen Gründen verzichtet wird und stattdessen gezielte redundante Informationen gesendet werden, um die Schätzung empfängerseitig zu verbessern. Gründe sind:

- Die zusätzliche Last im Kommunikationssystem soll so gering wie möglich gehalten werden. Ein anderer Ansatz schlägt vor, auf mehrere Wiederholungsanfragen mit einer Sendung zu antworten, wenn die Wiederholung technisch bedingt alle Empfänger erreicht.
- Der Sender muß nur wenig Informationen für eventuelle Anfragen zwischenspeichern.
- Der Sender soll mit dem Abarbeiten der Anfragen nicht so stark belastet werden, daß die Kodierung aktueller Medienströme darunter leidet.

Die Technik erlaubt grundsätzlich auch, mit einer Anfrage ganz spezielle Informationen vom Sender anzufordern („dritter Koeffizient des vierten Blockes“) um eventuell nur teilweise empfangene Pakete soweit gewünscht zu vervollständigen und die Wiederholung von Redundanz zu vermeiden. Wenn auf eine Wiederholungsanfrage mit dem Senden von bereits früher kodierten Daten reagiert wird, ist diese Technik alleine auf Transportebene (im Kanalkodierer) angesiedelt. Wenn die Daten für die Wiederholungssendung erst durch den Quellkodierer erzeugt werden müssen, ist diese Technik schichtenübergreifend realisiert.

Die Umsetzung dieser Technik erfordert ein Sitzungs- und Netzmanagement, das die Anfragenexplosion im Fehlerfall vermeidet. Das ist gerade bei Sitzungen mit vielen Teilnehmern notwendig. Der Vorteil dieser Technik ist für den Empfänger, daß er im Bedarfsfall vertrauenswürdige Daten vom Sender zur Fehlerkorrektur erhalten kann und sich nicht auf eigene Schätzungen zur Korrektur verlassen muß, also die Authentizität der Korrektur erhöhen kann. Der Vorteil für den Sender liegt darin, daß die gesendete Redundanz nur im Bedarfsfall erhöht wird, sonst aber Bandbreite gespart wird.

4.3.2 Aufeinanderfolgende Einheiten in verschiedenen Paketen

Häufig entscheidet man sich dafür, mehrere Dateneinheiten des Quellkodierers (z.B. Blöcke, Bildscheiben oder die Informationen über ein ganzes Bild) im Kanalkodierer zu einem Paket zusammenzufassen. Werden aufeinanderfolgende

Einheiten in einem Paket zusammengefaßt und geht dieses Paket verloren, so entsteht bei der Präsentation eine große Lücke. Empfängerbasierte Fehlerkorrekturtechniken (auf die später eingegangen werden wird) arbeiten aber effektiver auf kleinen und weitverstreuten Lücken als auf größeren und dichter zusammenliegenden [PHH 1998]. Darum wird bei dieser Technik durch Umsortieren der Einheiten im Kanalkodierer sichergestellt, daß aufeinanderfolgende Einheiten nicht in demselben Paket transportiert werden. Abbildung 10 zeigt ein Beispiel mit vier Einheiten pro Paket.

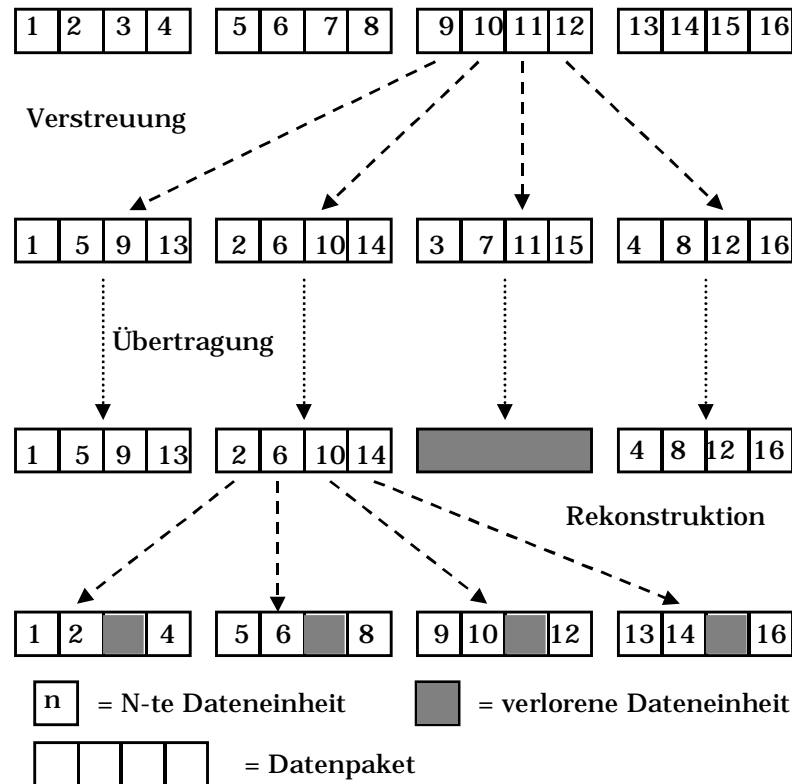


Abbildung 10: Streuung aufeinanderfolgender Pakete (*interleaving*)

Der Vorteil dieser Technik liegt darin, daß Verluste äquidistant gestreut werden. Sind die Einheiten vom Nutzer alleine kaum wahrnehmbar (kurze Audioteile, kleine Bildausschnitte) so ist ein Verlust ohne weitere Maßnahmen des Empfängers unauffällig. Empfängerbasierte Fehlerkorrekturmaßnahmen werden erleichtert. Diese Technik erzeugt keine weitere Last für das Kommunikationssystem, d.h. es werden nur die Dateneinheiten versendet. Nachteil dieser Technik ist die zusätzliche Verzögerung, die durch das Umsortieren der Einheiten beim Sender und Empfänger eingeführt wird. Die Anzahl der Einheiten pro Paket kann vom Benutzer eingestellt werden oder kann aufgrund von Messungen in Sender und Empfänger festgelegt werden.

Diese Technik ist für realzeitorientierte Videokommunikation über ATM angewandt worden [WaZ 1998] um Zellverluste zu verschleiern und als Kanalkodierung bei Audioanwendungen über das Internet. [PHH 1998]. In der Bildübertragung kann diese Technik im Kanalkodierer angewendet werden, um

- den Ausfall aufeinanderfolgender Bilder zu verhindern;
- den Ausfall aufeinanderfolgender Bildscheiben zu verhindern;
- den Ausfall aufeinanderfolgender Blöcke zu verhindern;
- den Ausfall aufeinanderfolgender Koeffizienten zu verhindern.

Auf welcher dieser Ebenen die Technik angesiedelt wird, hängt stark mit der Fähigkeit des Kodierungsschemas zur Synchronisation zusammen, denn diese Technik setzt voraus, daß der rekonstruierte Strom vom Quellkodierer auch korrekt dargestellt werden kann. Dieses ist nicht der Fall, wenn die Dateneinheiten von variabler Länge sind und Kodewörter zur Synchronisation fehlen oder sich die Inhalte der Einheiten aufeinander beziehen.

Daraus folgt, daß diese Technik im Kanalkodierer auf schwächer komprimierte Datenströme besser anwendbar ist als auf stark komprimierte Einheiten, denn der Kanalkodierer kennt typischerweise die inhaltlichen Abhängigkeiten der Dateneinheiten nicht. Diese Probleme lassen sich umgehen, wenn man die Technik im Quellkodierer ansiedelt, der ja das volle Wissen über die Inhalte der Daten hat. Eingebettet in die Quellkodierung könnte die Verzögerung möglicherweise verringert werden und die Dateneinheiten klug gemischt dem Kanalkodierer übergeben werden. Der Nachteil bei Einbettung der Technik in die Quellkodierung ist, daß diese Technik dann an alle Empfänger propagiert werden müßte, um eine korrekte Darstellung zu ermöglichen. Gegen eine Ansiedlung der Technik in der Kanalkodierung bei realzeitorientierten Anwendungen spricht auch, daß im Internet Anwendungen mit mehreren Benutzern unterstützt werden sollen und die lokale Anwendung dieser Technik im Kanalkodierer (zur Überbrückung EINER gestörten Verbindung) wegen der damit auch für nachfolgende Transportverbindungen eingeführten Verzögerung generell inakzeptabel ist.

4.3.3 Aufeinanderfolgende Pakete über verschiedene Wege senden

Bei dieser Technik wird der Paketstrom über möglichst disjunkte Pfade zum Empfänger gelenkt, indem gleichzeitig mehrere verschiedene Einwahlknoten bei Übertragung über ein Weitverkehrsnetz gewählt werden (*information dispersal*). So soll die Wahrscheinlichkeit, mehrere Pakete in Folge zu verlieren (z.B. durch Speicherüberlauf eines Vermittlungsrechners) gesenkt werden und die Last soll gleichmässiger eingespeist werden [ASW 1998]. Ein Vorteil dieser Technik ist, daß sie keine zusätzliche Last erzeugt, die übertragen werden muß. Applikationen, die diese Technik unterstützen, könnten zur Laufzeit den Provider des Kommunikationssystems (oder sogar das Kommunikationssystem selbst) wechseln, wenn Überlast beobachtet wird.

Nachteilig wirkt sich aus, daß der Sender üblicherweise nur Einfluß auf die Einwahlknoten (d.h. den Vermittlungsrechner) hat und sich im weiteren Verlauf der Sendung sehr wohl zufällig ein ähnlicher Pfad für die Ströme ergeben könnte. Ein weiterer Nachteil besteht darin, daß verschiedene Pfade eventuell auch zu verschiedenen Verzögerungen führen und dadurch viele Paketverluste entstehen können, daß die eine Verbindung wesentlich schneller ist als die andere. Beim Telefonnetz muß „ein anderer Einwahlknoten“ nicht unbedingt ein anderer erster Vermittlungsrechner sein, könnte auch eine nur kleine Abweichung mittendrin bedeuten.

Diese Technik ist im Kanalkodierer angesiedelt und stellt ein „*least-cost-routing*“ im Sinne von Verlustwahrscheinlichkeiten dar. Eine ähnliche Technik wird von Shin und Han [ShH 1998] vorgestellt: Jederzeit wird ein Kanal zur Übertragung benutzt und ein zweiter für den Fall eines Staus im Netz als Sicherheitskanal bereitgehalten. Eine Anwendung dieser Technik bietet sich besonders für längere Kommunikationen an, weil das Netzverhalten sich in kurzen Zeiträumen nur wenig ändert. Heute scheinen Techniken vielversprechender, die willkürlich den Netzzugang wählen und dann eine Verbindung „so gut es geht“ etablieren. Dauert die Kommunikation dann länger, könnte das Sitzungsmanagement sich nach Wegen umsehen, die Qualität durch Wechsel der Transporttechnik zu verbessern.

4.3.4 Skalierung

Diese Technik beruht auf der Idee, zur Laufzeit die Kodierung anzupassen, wenn Indikatoren das notwendig erscheinen lassen. Dabei können Parameter der Quellkodierung angepasst werden oder auf eine ganz andere Quellkodierung umgestellt werden. Käppner [Käp 1997] schlägt darüber hinaus Skalierung auf der Vermittlungs- und Transportschicht vor.

Blockbasierte Bildkodierung kennt drei Parameter zur Kontrolle der resultierenden Datenrate. Über den Quantisierungsfaktor wird bestimmt, wie stark der Verlust bei der Digitalisierung des Analogsignals sein soll (je geringer der Faktor, desto besser die Qualität) und über den Komprimierungsmodus wird bestimmt, welche benachbarten Bilder zusätzlich zur Komprimierung herangezogen werden sollen. Über Bewegungsausgleich (*motion compensation*) wird zusätzlich versucht, mithilfe räumlich oder zeitlich benachbarter Blöcke eine gute Schätzung des zu komprimierenden Blockes zu erhalten.

Diese Parameter erlauben die Fehlerrobustheit des resultierenden Datenstromes der Quellkodierung durch das Sitzungs- und Netzmanagement zu steuern. Wenn Indikatoren anzeigen, daß die Netzlast gesenkt werden sollte, so kann die Quantisierung erhöht werden, was für die Teilnehmer zu verschwommeneren Bildern führt. Werden vermehrt Verluste beobachtet, so könnten mehr Blöcke ohne Abhängigkeiten zu zeitlich benachbarten Blöcken kodiert werden, um Fehlerfortpflanzung (*error propagation*) zu vermeiden. Allerdings wird dadurch die Netzlast erhöht, weil weniger stark komprimierte Bilder übertragen werden. Außerdem könnte die Fehlerrobustheit erhöht werden durch Einschränkung des Bereiches, indem Bewegungsausgleich erfolgt, um weniger Abhängigkeiten zwischen den Blöcken zu haben. Die Wahrnehmung dieser Alternativen stellen auch eine Fehlertoleranztechnik dar und werden als Alternativkodierung verstanden, weil man ihre Auswahl (bei einem Verzicht auf ein Sitzungsmanagement) auch dem Teilnehmer überlassen kann.

Neuere Standards unterstützen diese Technik deutlich besser als alte z.B. durch Steuerungsmöglichkeiten der Differenzkodierung. Eigenschaften der neueren Standards sind stichpunktartig im vorigen Kapitel erläutert. Die angesprochenen Kodierungsarten sind nicht disjunkt-alternativ zu verstehen, sondern können kombiniert werden, um das für eine Sitzung optimale Verhältnis aus Komprimierung und Fehlerrobustheit durch Redundanz in der Quellkodierung zu erreichen. Abbildung 11 zeigt, wie die Anpassung der Quellkodierung aussehen könnte, wenn die Fehlertoleranzstrategie vorsieht, daß die Redundanz genau dieselbe Bandbreite umfaßt wie intrakodierte Blöcke.

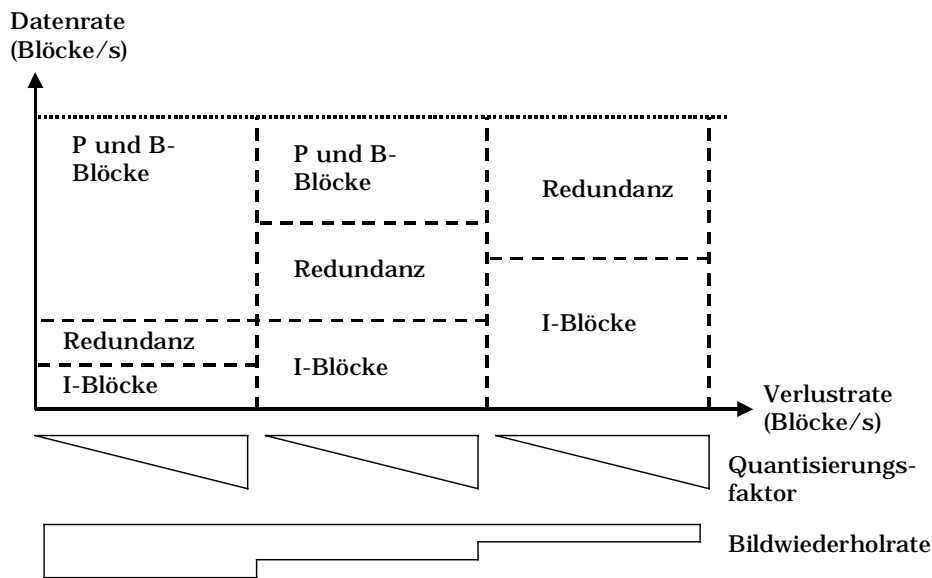


Abbildung 11: Skalierung

4.3.5 Bitorientierte Vorwärtsfehlerkorrektur

Ursprünglich zur Erkennung und/oder Korrektur von Bitfehlern in einem binären Kodewort entwickelt (fehlerkorrigierende Codes), lässt sich dieses Verfahren auch auf ganze Blöcke von Rohdaten anwenden, um Blockverlust erkennen und bestenfalls exakt aus den mitübertragenen Redundanzdaten rekonstruieren zu können. Dabei wird das i -te Bit aus jedem Paket benutzt, um das i -te Bit des Redundanzpaketes zu erzeugen. (Darum bitorientierte Vorwärtsfehlerkontrolle). Abbildung 12 zeigt die Verwendung eines Paketes mit redundanten Daten für drei Dateneinheiten.

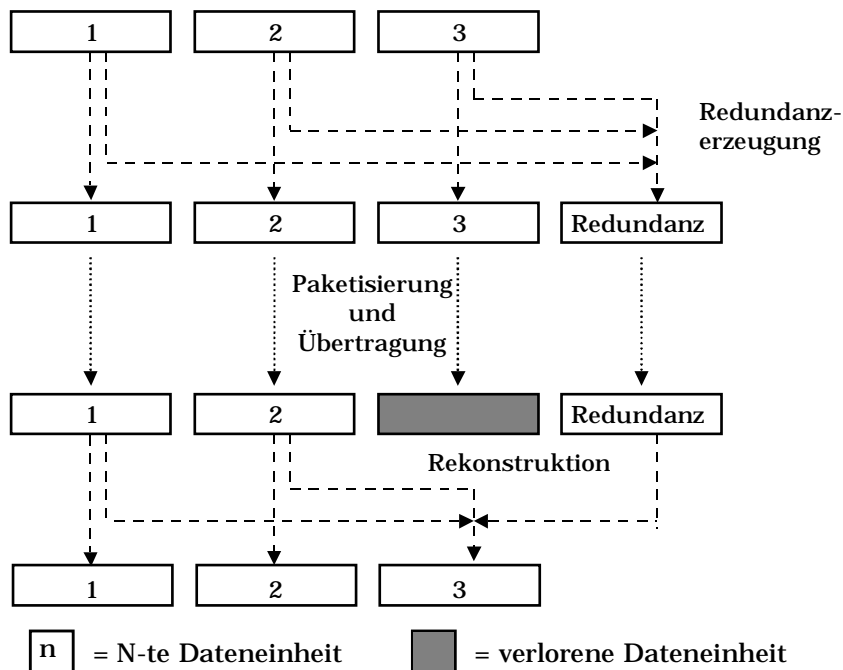


Abbildung 12: Bitorientierte Vorwärtsfehlerkorrektur

Es gibt eine große Anzahl von Techniken zur blockorientierten Vorwärtsfehlerkorrektur. Zur Verwendung als Nutzlast für RTP werden zur Zeit von der *Audio/Video Transport Working Group* der *Internet Engineering Task Force IETF* zwei

zur Standardisierung vorbereitet: Vorwärtsfehlerkorrektur, die auf Paritätskodieren basiert, und Reed-Solomon-Kodieren.

Beim Paritätskodieren wird die logische XOR-Operation über Gruppen von $n-k$ Paketen angewandt, um k korrespondierende Paritätspakete zu erzeugen. Eine große Bandbreite von Kombinationen ist erprobt und keine spezielle wird von der IETF gefordert. Rosenberg und Schulzrinne [RoS 1999] nennen als Beispielkombination, die bis drei Paketverluste in Folge repariert und dafür eine Verzögerung von vier Paketen benutzt:

$a,b,c,d,e,f \rightarrow f(a), f(b), f(a,b,c), f(c), f(a,c,d), f(a,b,d), f(d), \dots$

Dabei sind a bis f Pakete aus dem Datenstrom und f ist die XOR-Funktion.

Reed-Solomon-Kodes sind bekannt für ihre ausgezeichneten Fehlerkorrektureigenschaften, besonders wegen ihrer Widerstandsfähigkeit gegen Bündelfehler, die auf algebraischen Eigenschaften von Polynomen beruhen. Diese Kodierung benutzt zur Redundanzzeugung neben den Datenpaketen ein Polynom und eine zugehörige Nummernbasis. Die Mathematik hinter diesem Verfahren ist sehr komplex und bei weitem zu umfangreich für diese Arbeit. Wenn keine Verluste auftreten, dauert Kodieren und Dekodieren etwa gleichlange, treten Verluste auf dauert das Dekodieren deutlich länger [PHH 1998].

Vorteile bitorientierter Vorwärtsfehlerkorrektur sind:

- Unabhängigkeit vom Inhalt der Pakete,
- exakte Wiederherstellung der Daten bei erkanntem Verlust und
- relativ einfach zu implementierende Erzeugung der Redundanzpakete.

Nachteile dieser Technik sind:

- zusätzliche Verzögerung (unter Umständen asymmetrisch, d.h. Dekodierung dauert länger als Kodierung),
- erhöhte Netzlast durch zusätzliche Pakete,
- kein garantiertes Entdecken aller Fehler (weil sich zwei Fehler in der Wirkung aufheben könnten),
- eventuell keine Korrektur möglich, falls mehr Fehler auftreten, als durch die Technik wiederhergestellt werden können.

Es gibt keinen sanften Qualitätsabfall (*graceful degradation*), sondern im Ergebnis nur eine exakte rechtzeitige Wiederherstellung oder einen Totalverlust.

Diese Technik ist im Kanalkodierer angesiedelt, verwendet also kein Wissen über den Inhalt der Pakete und kann eventuell lokal angewandt werden, um fehlerhafte Transportverbindungen zuverlässiger zu machen. Für realzeitorientierte Anwendungen liegt die Implementierung einer dynamischen Steuerung aufgrund von Indikatoren durch das Sitzungs- oder Netzmanagement nahe, um wirklich nur benötigte Redundanz zu erzeugen. Wenn der Kanalkodierer nur gleichartige Schnittstellen (Kanäle) zum Kommunikationssystem hat, der Quellkodierer aber eine geschichtete Kodierung leistet (d.h. der Quellkodierer Daten verschiedener Priorität für die Darstellung liefert), dann können mit dieser Technik auf Kosten der Bandbreite zuverlässigere Kanäle für wichtigere Daten bereitgestellt werden (Kanaltransformation). Abbildung 13 zeigt bitorientierte Vorwärtsfehlerkontrolle (13a) im Gegensatz zu inhaltsorientierter Vorwärtsfehlerkontrolle (13b), die Teil der Quellkodierung ist und eine Quellentransformation ermöglicht. (siehe nächster Abschnitt)

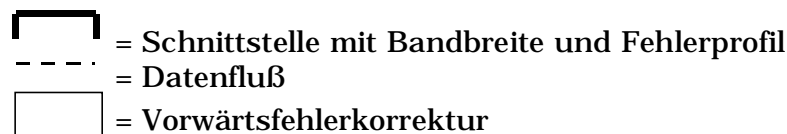
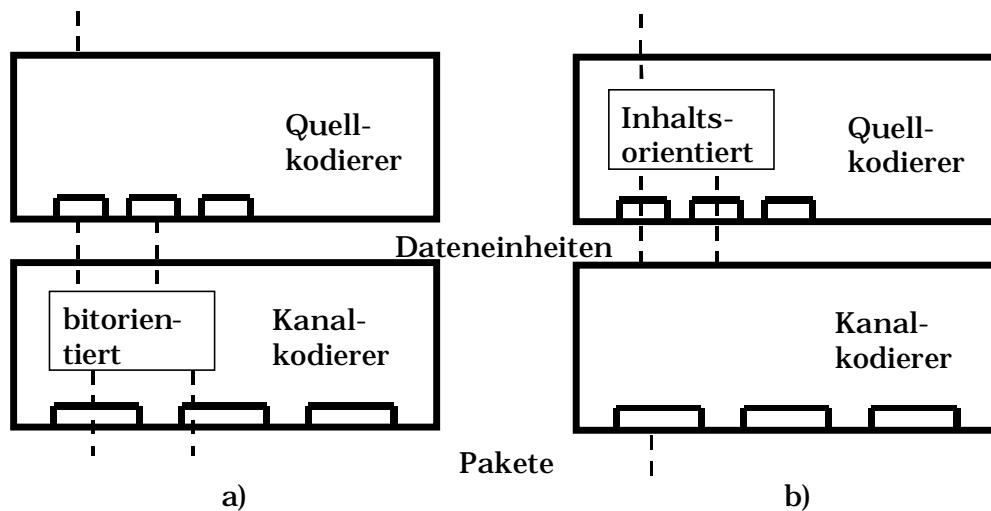


Abbildung 13: Vorwärtsfehlerkorrektur als Transformator

4.3.6 Inhaltsorientierte Vorwärtsfehlerkorrektur

Inhaltsorientierte Vorwärtsfehlerkorrektur nutzt Wissen über die Art der zu übertragenden Rohdaten aus, um die zu übertragende Redundanz zu erzeugen. Diese Technik ist in der Quellkodierung angesiedelt. Zuerst im Zusammenhang mit Funktelefonie untersucht, ließen sich bald hohe Performanzgewinne nachweisen, wenn statt klassischer algebraischer Verfahren medienabhängige Verfahren verwendet wurden [PHH 1998].

Im Zusammenhang mit inhaltsorientierter Vorwärtsfehlerkorrektur werden die Begriffe der *Primärkodierung* und der *Sekundärkodierung* eingeführt: Einige Implementationen verwenden eine qualitativ gute Quellkodierung zur Erzeugung der Rohdaten, die unbehandelt verschickt werden. Gleichzeitig werden durch eine stärker komprimierende Quellkodierung (und damit einhergehend schlechterer Präsentationsqualität) aus demselben Inhalt sekundärkodierte Rohdaten erzeugt, die als Redundanz mitgesendet werden. Beim Empfänger werden die primärkodierte Einheiten vorrangig zur Präsentation verwendet; bei deren Verlust wird auf die redundant mitgesendeten sekundärkodierte Einheiten zur Fehlerverschleierung zurückgegriffen (siehe Abbildung 14).

Diese Technik bietet zwei Vorteile. Zum einen bietet sie gute Reparaturmöglichkeiten durch hohe Redundanz, zum anderen ist sie echtzeitgeeignet, weil verzichtet wird auf Umsortieren von Paketen und auf Algorithmen, die mehrere Pakete gleichzeitig absichern. Diese Technik verwirklicht eine Strategie des „permanenten Sendens auf zwei Qualitätskanälen“. Außerdem wird durch diese Technik im Vergleich zu bitorientierter Vorwärtsfehlerkorrektur zusätzliche Performanz gewonnen, d.h. geringerer Codec-Aufwand geht einher mit für die Präsentationsqualität besser nutzbarer Redundanz. Diese Redundanz ist im allgemeinen größer als bei bitorientierter Vorwärtsfehlerkorrektur. Diesem kann entgegengewirkt werden, indem diese Technik nur bei als wichtig eingestuft Paketen angewendet wird, beispielsweise nicht während Sprechpausen in

Sprachkommunikation. Der Preis dieser Variante ist bei Sprachkommunikation ein nicht mehr so glattes Lastprofil. Weil bei dieser Technik die verlorengegangenen Daten nur approximativ wiederhergestellt werden können, eignet sich diese Technik für die Präsentationslast einer Applikation, aber nicht für die Steuerdaten der Kodierer.

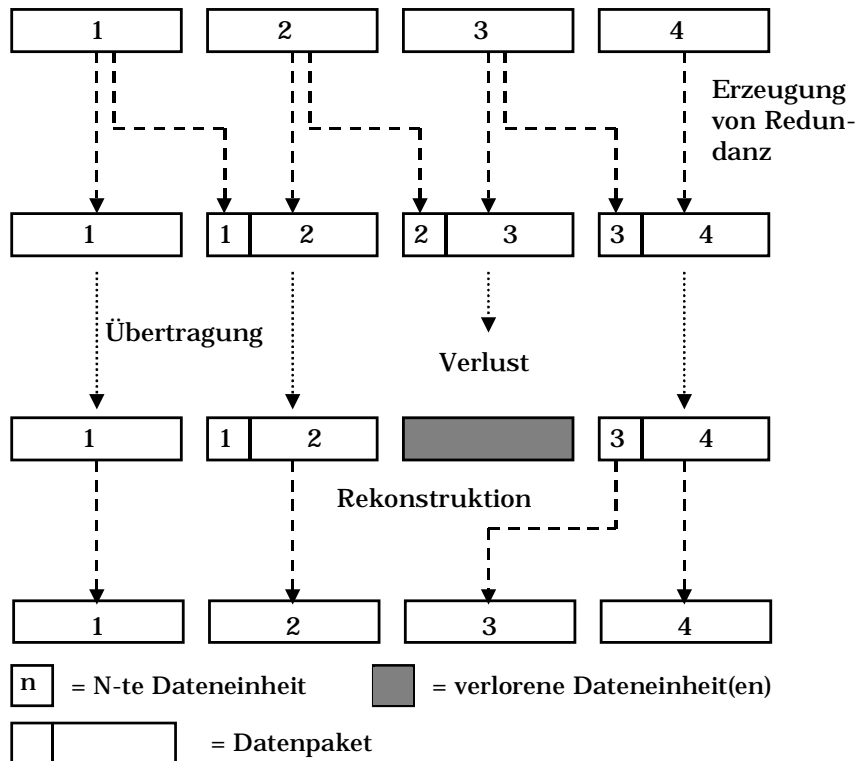


Abbildung 14: Inhaltsorientierte Vorwärtsfehlerkontrolle

Für Bewegtbildübertragung kann man sich z.B. folgende Varianten für die Sekundärkodierung vorstellen:

- gröber quantisierte Blöcke,
- die niedrigen Koeffizienten des vorherigen Blocks,
- Koeffizienten, die sich auf einen anderen Referenzblock beziehen als der Primärkodierte;
- niedrige Koeffizienten des Referenzblockes mitschicken, auf den sich der vorher geschickte primärkodierte Referenzblock bezieht.

Weitere Möglichkeiten sind denkbar. Fällt aus algorithmischen Gründen bei Erstellung der Primärkodierung die Sekundärkodierung eher an (z.B. als Vorstufe) so ist es natürlich denkbar, die Sekundärkodierung eher als die Primärkodierung zu schicken. Ist eine größere Verzögerung tolerabel, kann auch eine Tertiärkodierung eingeführt werden. Diese Technik ist sehr flexibel, ihre Parameter müssen aber durch das Sitzungsmanagement an alle Teilnehmer der Sitzung propagiert werden. Primär- wie Sekundärlast müssen Synchronisationspunkte enthalten, um im Dekodierer alleinstehend verwendet werden zu können.

Das Hinzufügen von Redundanz im selben Kontext wie die Komprimierung stellt eine „Dekomprimierung“ dar und verrät im Grunde nur, daß die Quellkodierung zu stark auf Komprimierung ausgerichtet ist und zuwenig Fehlertoleranztechniken unterstützt. Wenn man bei der eingeführten Schnittstellenbetrachtung bleibt, so stellt inhaltsorientierte Vorwärtsfehlerkorrektur eine Änderung des Datenformates des Quellkodierers dar und somit ist mit diesem Schritt die Eignung des Quellkodierers an sich infrage gestellt (Abbildung 13). Wang und Zhu [WaZ 1998]

diskutieren Quellkodierer, die nicht standardkonform sind, aber inhaltsorientierte Vorwärtsfehlerkorrektur durch ein fehlertolerantes Ausgabeformat überflüssig machen.

4.4 Empfängerbasierte Fehlertoleranztechniken

Abbildung 15 gibt eine Übersicht über Fehlertoleranztechniken beim (Sprach-) Empfänger und ist an Perkins, Hodson und Hardman [PHH 1998] angelehnt:

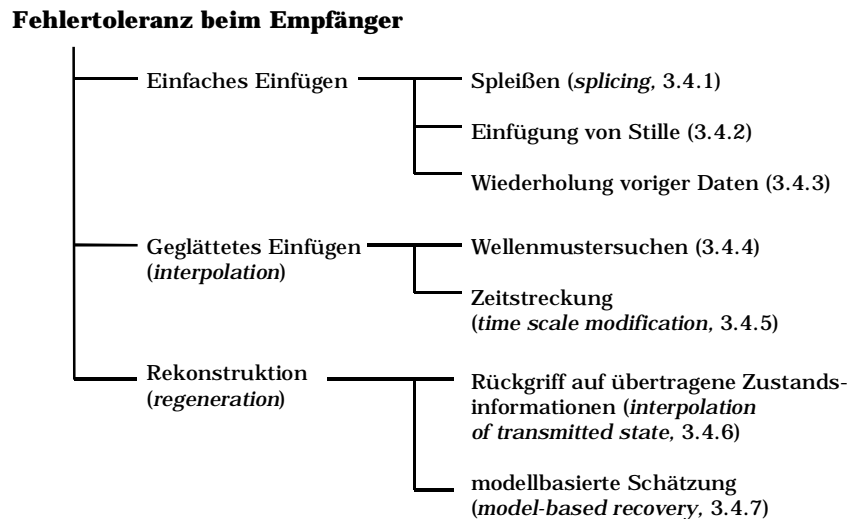


Abbildung 15: Empfängerbasierte Fehlertoleranztechniken

Nachdem die beim Empfänger entdeckten Paketverluste soweit möglich mit vom Sender angebotenen Fehlertoleranztechniken behoben wurden, verbleiben eventuell einige wenige und kleine Lücken im Datenstrom. Die folgenden Techniken bieten Lösungen, die Verluste auszugleichen, ohne dafür mit dem Sender kommunizieren zu müssen. Diese Techniken wurden im Zusammenhang mit Übertragung von Audio über das Internet untersucht und werden zunächst auch in diesem Umfeld vorgestellt. Die Anwendbarkeit auf realzeitorientierte Videokommunikation wird für jede Technik gesondert untersucht.

Weil keine Kontrollkommunikation zwischen Sender und Empfänger stattfindet, sind alle diese Techniken als passiv einzustufen. Jede dieser Techniken bedient sich einer Pseudo-Dekodierung, mit der die Qualität der Präsentation durch mehr oder weniger authentisch approximierten Lückenfüller verbessert werden soll. Diese approximierten Lückenfüller stellen mathematische Schätzer dar. Der statistische Aufwand zur Ermittlung dieser Schätzer steigt von den erstgenannten zu den letztgenannten an.

4.4.1 Spleißen

Wie bei allen Techniken des einfachen Einfügens wird die Fülleinheit generiert, ohne auf die Charakteristik des Datenstromes zurückzugreifen. Darum können diese Techniken auch als bitorientiert angesehen werden. Beim Spleißen wird der Verlust von Dateneinheiten durch Einfügen einer Fülleinheit der Länge Null behandelt (Abbildung 16).

Diese Technik ist sehr einfach zu implementieren, verändert aber das zeitliche Verhalten des Stromes. Bei Sprachübertragung arbeitet dieses Verfahren nur akzeptabel bei kurzen Dateneinheiten (4-16ms Sprache pro Dateneinheit) und geringen Verlusten (unter 3%)[PHH 1998]. Eine so geringe Verlustrate wird heute

im Internet selten erreicht, so daß die Anwendung dieser Technik für Audio nur eine Notlösung darstellt.

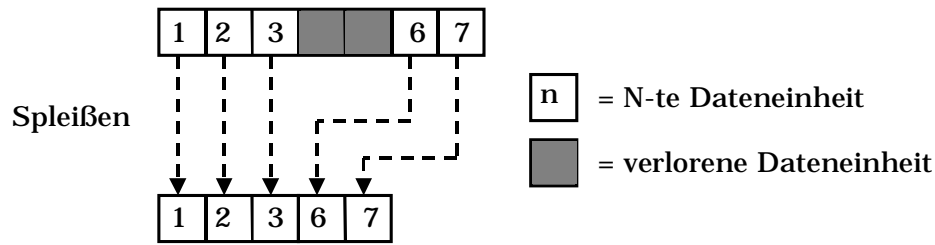


Abbildung 16: Spleißen von Einheiten (*splicing*)

Wenn eine Dateneinheit ein ganzes Bild eines Videos umfaßt, so führt diese Technik nur dann nicht zu einer niedrigeren Präsentationsqualität, wenn Spleißen die Bildrate nicht unter 16 Bilder pro Sekunde senkt. Umfaßt eine Datenrate einen Teil eines Bildes (eine Blockgruppe oder einen Makroblock), so führt Spleißen zunächst zu einem veränderten räumlichen Verhalten des Stromes (nachfolgende Bildteile sind verschoben, das Bild nicht zusammenhängend erkennbar) und anschließend zu einem veränderten zeitlichen Verhalten, wenn Bildteile eines nachfolgenden Bildes in das aktuelle Bild „nachrücken“. Spleißen ist nicht sinnvoll, wenn eine Dateneinheit nur einen Teil eines Makroblocks (z.B. einen Block, einige Koeffizienten oder Bewegungsvektoren) umfaßt, weil in diesem Falle das Weglassen einer verlorenen Dateneinheit zu Syntaxfehlern im Datenstrom und damit unkalkulierbarem Verhalten des Dekodierers führt. Nur syntaktisch zusammenhängende Daten (mindestens ein Makroblock) können gespleißt werden.

4.4.2 Einfügung von Stille

Das Einfügen von Stille ist die einfachste Technik die das zeitliche Verhalten eines Datenstromes erhält. Das zeitliche Verhalten nicht zu ändern ist besonders wichtig für Applikationen, in denen verschiedene Ströme synchronisiert werden müssen (z.B. eine Video- mit einer Audiosequenz). Beim Einfügen von Stille werden verlorene Dateneinheiten durch Füllereinheiten ersetzt. Diese Füllereinheiten werden ohne Wissen über den Inhalt der Dateneinheiten erzeugt. Diese Technik arbeitet also bitorientiert. Für Sprache ist das Einfügen von Stille oder Rauschen erprobt. Abbildung 17 zeigt schematisch, wie diese Technik abläuft.

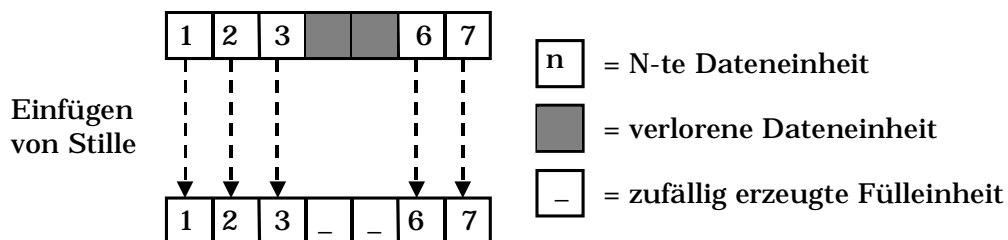


Abbildung 17: Einfügung von Stille

Die Einfügung von Stille erzielt nur für kurze Einheiten (kleiner 4ms) und geringe Verlustraten (kleiner 2%) eine akzeptable Präsentationsqualität. Obwohl heutige Funknetze diese Grenzen deutlich überschreiten, wird diese Technik häufig für Audio verwendet, weil sie einfach zu implementieren ist. Eine nachweislich bessere subjektive Qualität wird durch das Einfügen von Hintergrundrauschen erzielt. [PHH 1998]

Die Anwendung dieser Technik bei Video bedeutet, ein einfach erzeugtes Bild oder einen einfach erzeugten Bildteil zu verwenden, wenn der entsprechende Bereich verloren wurde. Die Füllereinheit könnte einfarbig sein oder ein als subjektiv unauffällig empfundenen Muster tragen. Sind Teile eines Makroblocks verloren worden (Koeffizienten, Bewegungsvektoren, ein Block) könnten die fehlenden Komponenten durch „default“-Werte ersetzt werden, um die korrekt übertragenen Daten überhaupt nutzen zu können. Wenn von einem intrakodierten Makroblock zum Beispiel nur die Daten des Chrominanz-Blockes korrekt übertragen wurden, so könnten die fehlenden Koeffizienten der Luminanzblöcke zu Null gesetzt werden, um einen syntaktisch korrekten Makroblock zu erhalten. Der behandelte Makroblock kann vom Dekodierer dargestellt werden, der unbehandelte aufgrund der defekten Syntax nicht. Durch die Behandlung werden also die vorher nicht darstellbaren Daten nutzbar gemacht. Es ist zu vermuten, daß diese Technik auch bei Video nur für kleine (zeitliche oder räumliche) Verluste gut arbeitet (nämlich solange, wie die Füllereinheiten nicht bewußt wahrnehmbar sind). Für internet-basierte Applikationen werden vermutlich weitere Techniken eingesetzt werden müssen, um die bewußt wahrnehmbaren Verluste auszugleichen.

4.4.3 Wiederholung voriger Daten

Die Selbstähnlichkeit menschlicher Sprache in kurzen Zeitintervallen führte zu dem Ansatz, entstehende Lücken durch Wiederholung der direkt vor der Lücke empfangenen Daten zu schließen. (Abbildung 18)

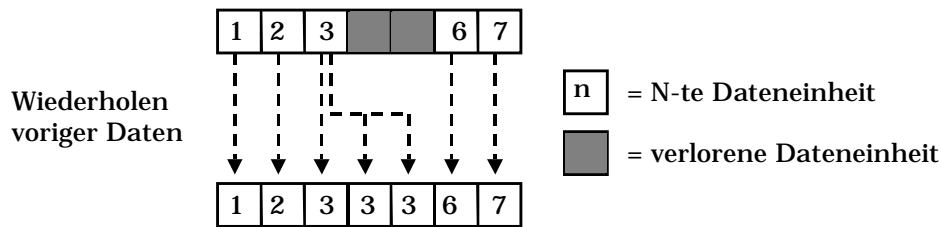


Abbildung 18: Wiederholen voriger Daten

Diese Technik führt zu besserer Präsentationsqualität als Einfügung von Rauschen, ohne viel größeren Implementations- oder Rechenaufwand zu erfordern. Aufgrund menschlicher Hörwahrnehmung arbeitet diese Technik am besten, wenn längere Lücken nicht mit einer langen Wiederholung gefüllt werden, sondern mit wiederholter Wiederholung der direkt vor der Lücke liegenden (typischerweise 20ms langen) Einheit gearbeitet wird und wenn diese bei jeder Wiederholung abgeschwächt (d.h. leiser) wird. Diese spezielle Art der Wiederholung mit Abschwächung wird im GSM-System für Funktelefonie vorgeschlagen, weil der Rechenaufwand einen guten Kompromiß darstellt zwischen den anderen Einfügungstechniken (die eine schlechte Wahrnehmungsqualität erzielen) und den folgenden Verschleierungstechniken, deren Rechenaufwand deutlich größer ist. [PHH 1998]

Diese Technik bei Bewegtbildübertragung anzuwenden, bedeutet im Falle eines Bildverlustes über die Dauer der Lücke das zuletzt empfangene Bild wiederholt darzustellen. Für den Betrachter ist das Bild dann „eingefroren“ und nach der Lücke scheint für den Betrachter ein Schnitt zu erfolgen, wenn wieder korrekt empfangene Bilder dargestellt werden. Wird diese Technik für Bildteile (Scheiben, Makroblöcke) angewendet, so scheinen Teile des Bildes eingefroren zu sein (im Vordergrund zu schweben) während die eigentliche Sequenz im Hintergrund weiterzulaufen scheint. Für Bildsequenzen, die als „gering bewegt“ eingestuft werden können, wird vermutet, daß diese Technik über kurze Lücken gute Resultate liefern kann, weil der Darstellungsfehler, der durch diese Technik

entsteht, so gering ist, daß er vermutlich nicht wahrnehmbar ist. Sitzt der Sender in einem Bildtelefonat zum Beispiel vor einem Baum, dessen Blätter sich leicht im Wind bewegen, und werden beim Empfänger verlorene Blöcke mit dieser Technik behandelt, so entsteht im Hintergrund eine kurze Diskontinuität in der Bewegung der Blätter. Diese dürfte der Empfänger auch bei längerer Dauer der Verluste kaum bemerken, weil er sich vermutlich auf das Gesicht des Senders konzentriert, statt auf den unscharfen Hintergrund. Es liegt auf der Hand, daß diese Technik für den Bereich der Szene, auf den die Kamera fokussiert ist (der also „scharf“ erscheint) vermutlich nicht so gut arbeitet.

Sind Teile eines Makroblocks verloren, so könnten diese statt durch „default“-Werte durch die Werte eines angrenzenden (benachbarten) Makroblocks ersetzt werden. Dahinter steht die Überlegung, daß dessen Werte den verlorenen Werten vermutlich ähnlicher sind als „default“-Werte. Wang und Zhu [WaZ 1998] bemerken, daß diese Technik besonders dann effektiv ist, wenn bei interkodierten Makroblöcken der Bewegungsvektor korrekt übertragen wurde, die Koeffizienten aber fehlen. Das liegt daran, daß der Bewegungsvektor bereits den Block angibt, dessen Werte in der Umgebung die größte Ähnlichkeit mit den Werten des aktuellen Blockes haben. In diesem Fall sind nur die Differenzen zwischen den Referenzwerten und den tatsächlichen Werten verloren worden. Die Werte des durch den Bewegungsvektor referenzierten Blockes sind den verlorenen ähnlicher als die Werte anderer benachbarter Blöcke.

4.4.4 Wellenmustersuchen

Anders als beim Wiederholen entspricht eine Fülleinheit, die durch Wellenmustersuchen erzeugt wurde, nicht automatisch einer anderen Dateneinheit, sondern sie wird aufgrund von Merkmalen gebildet. Diese Merkmale werden durch Analyse der Daten auf beiden Seiten der Lücke gewonnen. Diese Merkmale werden so gesucht, daß die Fülleinheit den Eindruck der Kontinuität beim Benutzer am besten unterstützt. Abbildung 19 zeigt ein Beispiel. Nach der statistischen Untersuchung der Dateneinheiten, die die Lücke umgeben, wurden die Dateneinheiten eins und sechs zum Füllen ausgewählt.

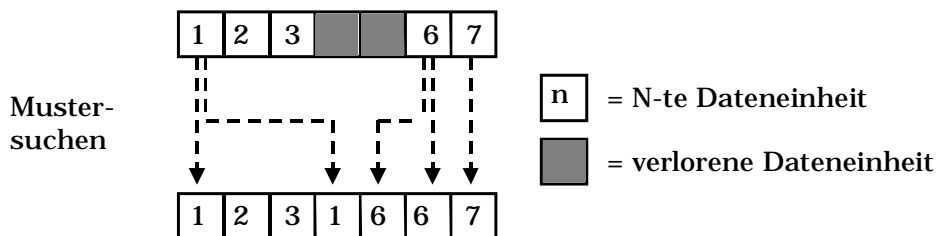


Abbildung 19: Wellenmustersuchen

Bei Übertragung von Sprache über Internet existieren Erfahrungen mit *schabloniertem* Wellenmustersuchen: Ein- oder beidseitig der Lücke werden Daten benutzt, um mittels einer Schablone ein in die Lücke passendes Wellenmuster zu finden. Beim einseitigen Verfahren wird das gefundene Muster die Lücke hindurch mehrfach wiederholt, bei beidseitigen Verfahren werden die auf jeder Seite gefundenen Muster interpoliert, um das entgültige Muster zu erhalten. Wie erwartet arbeitet das zweiseitige Schema besser als das einseitige und beide arbeiten besser als Einfügung von Stille und Wiederholung voriger Daten.

Eine Verfeinerung dieser Technik stellt das *algorithmische* Wellenmustersuchen dar. Bei dieser Technik wird zum Suchen nach geeigneten Mustern in der Sprache statt einer festen Schablone ein Algorithmus verwendet. Dadurch wird die

Schablone flexibilisiert. Die Verschleierung arbeitet wie beim schablonierten Wellenmuster-suchen, allerdings wird bei Verlusten während Sprechpausen die weniger berechnungsaufwendige Wiederholung voriger Daten angewendet. Dieses Verfahren arbeitet marginal besser als schabloniertes Wellenmustersuchen [PHH 1998].

Sind intrakodierte Bildteile verloren gegangen, würde eine Anwendung dieses Verfahrens bedeuten, daß Füllbereiche erzeugt würden, deren Werte z.B. Mittelwerte der angrenzenden, korrekt übertragenen Bereiche sind. Damit würde der Eindruck des „eingefrorenen Bildes mit anschließendem Schnitt“ verhindert, der beim Wiederholen entsteht. Vielmehr würde vermutlich eine kurzzeitige Unschärfe über den entsprechenden Bereich wahrgenommen werden. Statt der Bildung von Mittelwerten wird auch ein mehrfach gestuftes Glättungsverfahren von den Rändern des verlorenen Bereiches bis zur Mitte vorgeschlagen, um das Entstehen von „Pseudokanten“ zu verhindern. [WaZ 1998]

Sind interkodierte Bildteile verloren gegangen, so entspricht die Anwendung dieser Technik zwei Rekonstruktionsschritten für jeden Block:

- der Erzeugung eines Bewegungsvektors und
- der Erzeugung von dazugehörigen Koeffizienten.

Wang und Zhu [WaZ 1998] schlagen vor, den Bewegungsvektor eines verlorenen Blocks aus den Bewegungsvektoren umliegender Blöcke zu bilden und als Koeffizienten solche Werte zu wählen, die das Entstehen von „Pseudokanten“ an den Rändern des Blockes verhindern. Alternativ können die Koeffizienten aus den entsprechenden Werten der benachbarten Blöcke gebildet werden.

Eine weitere Verfeinerung des Verfahrens liegt darin, zusätzlich die Umgebung eines verlorenen Bereiches nach Kanten zu durchsuchen und diese in den verlorenen Bereich zu verlängern. Dieses stellt eine teilweise Abkehr von blockbasierter Bildkodierung hin zu objektbasierter Bildkodierung dar. Der algorithmische Aufwand ist so hoch, daß diese Verfeinerung für realzeitorientierte Videokommunikation heute noch nicht anwendbar erscheint.

4.4.5 Zeitstreckung

Wie der Name andeutet, arbeitet dieses Verfahren mit Modifikation der Zeitskala. Die Dateneinheiten auf jeder Seite des Verlustes werden so über die Lücke gestreckt, daß möglichst geringer Qualitätsverlust auftritt und die Lücke vollständig überdeckt wird. Abbildung 20 zeigt als Beispiel, wie umliegende Einheiten über einen Verlust gestreckt werden.

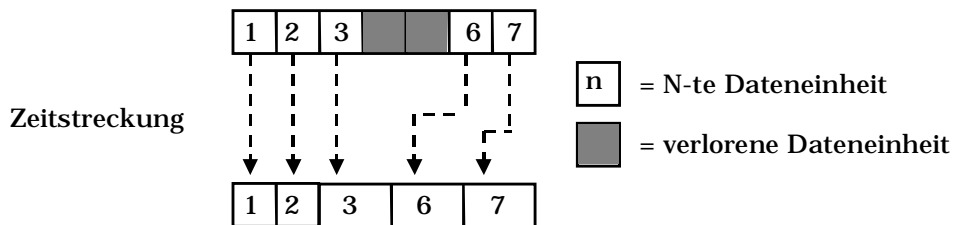


Abbildung 20: Zeitstreckung

Diese Technik ist bei Audio berechnungstechnisch aufwendig, weil verhindert werden muß, daß durch die Streckung der Eindruck von Kontinuität verloren geht. Diese Technik zeigt für Audio bessere Resultate als die zuvor genannte [PHH 1998].

Bei Videokommunikation scheint diese Technik erfolgversprechend zur Behandlung des Verlustes ganzer Bilder. Im Gegensatz zu Audiodaten, die kontinuierlich

präsentiert werden müssen, wird bei Bewegtbildern ein Bild nach dem Aufbau eine bestimmte Zeit (0,05 s bei 20 Bildern pro Sekunde) unverändert präsentiert. Durch Änderung dieser Zeit könnten einzelne Bildverluste ohne großen Berechnungsaufwand ausgeglichen werden. Diese Technik ist nicht anwendbar für Bilder, die zu einem festen Zeitpunkt präsentiert werden müssen, um die Synchronisation mit anderen Datenströmen (Audio oder gemeinsam benutzten Applikationen) sicherzustellen. Für verlorene Bildteile und für verlorene Teile eines Makroblockes ist diese Technik nicht anwendbar, weil ein Bild das zeitliche Verhalten aller untergeordneten Ebenen vorgibt.

4.4.6 Rückgriff auf übertragene Zustandsinformationen

Im Gegensatz zu den vorher genannten Techniken benutzt diese Technik Wissen über den Codec, um verlorene Dateneinheiten zu synthetisieren. Sie ist also nur codecabhängig einsetzbar. Für Audiocodex, die auf Transformation oder linearer Vorhersage beruhen, ist es für den Dekodierer möglich, zwischen Zuständen zu interpolieren. (Wie z.B. der ITU-G.723 Sprachkodierer) In Abbildung 21 scheinen die Füllereinheiten darum „aus dem Nichts“ zu entstehen.

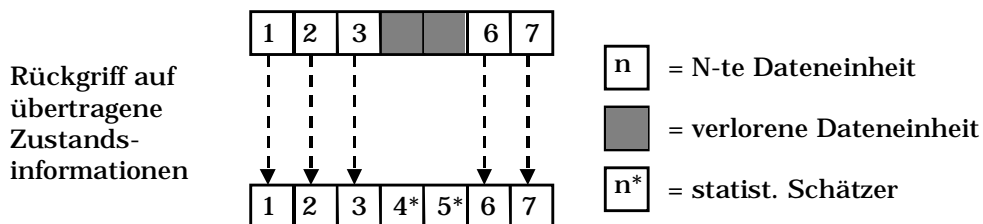


Abbildung 21: Rückgriff auf übertragene Zustandsinformationen

Andere Kodierer müssen auf beiden Seiten des Verlustes separat rekodieren und dadurch entsteht ein Grenzeffekt (Knacken). Für Audio stellen diese Codex mit Interpolation typischerweise sehr hohe Anforderungen an die Prozessoren [PHH 1998].

Die Codex zur Bewegtbildkompression, die gängigen Standards zugrunde liegen, können nicht über Lücken interpolieren, sondern erzeugen „Schlieren“, wenn ein Verlust nicht erkannt wurde. Die Entwicklung interpolationsfähiger Bildcodex bleibt eine Aufgabe für die Zukunft.

4.4.7 Modellbasierte Schätzung

Bei modellbasierter Schätzung wird die Sprache auf einer (besser beiden) Seiten des Verlustes an ein statistisches Modell (typischerweise ein autoregressives) angepaßt, welches benutzt wird, um synthetische Sprache über die Verlustperiode zu erzeugen (Abbildung 22).

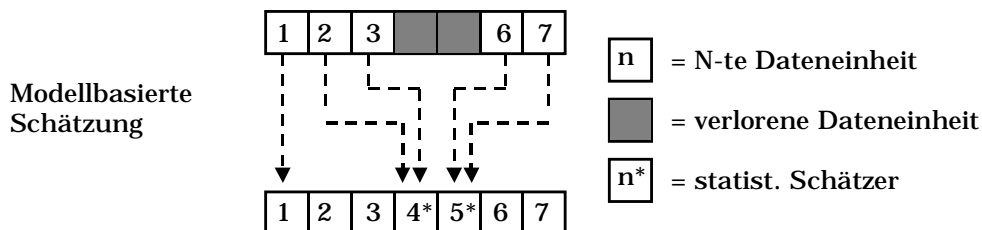


Abbildung 22: Modellbasierte Schätzung

Diese Technik erzeugt gezielt Parameter für den verwendeten Dekodierer, arbeitet als codecabhängig. Implementationen arbeiteten bei Investition entsprechender Rechenlast sehr gut.

Die Übertragung dieser Technik auf die Bewegtbildübertragung stellt eine echte Herausforderung dar: Wegen der zusätzlichen Dimensionen wären statistische Modelle wesentlich komplexer und damit rechenintensiver, außerdem müßten Schnitte durch das Sitzungsmanagement verhindert werden, um die statistische Stationarität der Daten hinreichend plausibel annehmen zu können. Durch die willkürliche Unterteilung der Bilder in Scheiben und Blöcke kommen außerdem sinnfremde Grenzen in das Modell hinein, die Prognosen aber entscheidend beeinflussen. Für realzeitorientierte Videokommunikation stellt diese Technik heute wegen ihrer algorithmischen Komplexität keine Option dar.

5 Umgang mit Fehlern bei Bildtelefonie im Internet

Fehlertoleranz ist ein zentraler Aspekt bei der Gestaltung von Systemen. Sie wird umgesetzt durch eine Fehlertoleranzstrategie, in der verschiedene Fehlertoleranztechniken kombiniert werden. Die Fehlertoleranzstrategie ist Teil des Managements des Systems. Sie hat eine globale Sicht über alle Schichten des Kommunikationssystems und über alle Bildebenen der Applikation hinweg. Fehlertoleranztechniken dagegen werden auf einer Schicht oder auf einer Bildebene realisiert und falls möglich im Rahmen des Netz- oder Sitzungsmanagements gesteuert.

5.1 Netzmanagement

Fehlertoleranz im Netzmanagement soll den rechtzeitigen und möglichst verlustfreien Transport der Dateneinheiten ermöglichen. Besonders die Ausweitung von bei der Übertragung entstandenen Bitfehlern zu Paketfehlern ist ohne große Zeitverzögerung zu vermeiden. Dieses sind Forderungen, die nicht im direkten Zusammenhang mit realzeitorientierter Videokommunikation stehen, sondern generell an Kommunikationsnetze gestellt werden und auf die neuere Protokolle besser eingehen werden als heutige.

Im Management können grundsätzlich zwei verschiedene Wege verfolgt werden:

- „like-to-use“-Management, das die notwendigen Ressourcen entweder reserviert und beim Zustandekommen der Kommunikation stets gute Qualität bietet (MPEG-Philosophie) oder den Dienst komplett abbricht;
- „better-then-nothing“-Management, das die Kommunikation in jedem Falle zustandekommen läßt, aber keine Gütegarantien gibt. Dann könnte im Laufe der Kommunikation die Qualität auch unter die Mindestdienstgüte sinken, ein Abbruch während einer Kommunikation ist aber viel unwahrscheinlicher (ITU-Philosophie).

„like-to-use“-Management wird erst in zukünftigen Netzen möglich sein, die neuere Protokolle unterstützen (Setzt man das „don't-fragment“-bit in IP oder benutzt RSVP so scheitert die Initialisierung der Kommunikation an unzureichender Leistungsfähigkeit des Netzes) Auf heutiger best-effort-Technologie basierende Techniken können nur der „better-then-nothing“-Strategie folgen. Dabei wird die Leistung des Netzes durch Konfigurationsoptionen gesteuert. Durch einen Verzicht auf Zugriff auf die unteren beiden Schichten geht viel Effizienz verloren, weil insbesondere die Fragmentierung nicht gesteuert werden kann. Einziger Ausweg ist die Wahl kleiner Pakete (unter 1500 Byte insgesamt bei Übergabe an IP) um Fragmentierung zu minimieren. Werden die Pakete noch kleiner gewählt, so wird die verfügbare Bandbreite durch steigenden Anteil der Header (die eine fixe Länge haben) an jedem übertragenen Paket unnötig eingeschränkt.

Verluste der Nutzdaten lassen sich am effizientesten auf der Verarbeitungsschicht beheben, weil dort das Wissen über den gesamten Inhalt des Paketes vorliegt (die gesamte Struktur bekannt ist). Auf tieferen Ebenen gibt es nur Kenntnis über die Struktur des schichtspezifischen Headers. Dieser ist darum auf jeder Schicht bitgenau zu übertragen (FEC). Wird die Nutzlast (*payload*) nicht unkommentiert nach oben weitergereicht, sondern Bitfehler korrigiert, so stellt dieses einen Effizienzverlust dar. Eventuell enthielt die *payload* nämlich Daten, deren rechtzeitige Übergabe wichtiger ist als die fehlerfreie. Darum ist auf bitorientierte Techniken generell zu verzichten. Jede Schicht soll möglichst wenig Paketverluste erzeugen. Dazu sind Schutzmaßnahmen auf den Headern der Pakete nötig. Über die Nutzlast der höheren Schicht sind auf der niedrigeren Schicht keine

Informationen verfügbar. Enthaltene Fehler sind also eventuell tolerabel und Fehlertoleranztechniken auf der Nutzlast sind zu unterbleiben. Eine Ausnahme bilden Fehlertoleranztechniken, die erforderlich sind, um den durch die Schicht nach oben angebotenen Dienst definitionsgemäß erbringen zu können. Beispiel: TCP bietet nach oben eine zuverlässige Verbindung an, benutzt nach unten das verbindungslose IP. Der Dienst wird realisiert, indem verlorene Pakete vom Sender automatisch angefordert werden, bis die Übertragung erfolgreich war.

Die rechtzeitige Zustellung eines Paketes kann das Netz nur gewährleisten unter Einsatz von RTP, das direkt auf IP aufsetzt. UDP dazwischenzulegen bringt keinen Nutzen, weil die bitorientierte Fehlerbehandlung von UDP zwar den Header schützt, aber auch überflüssigerweise auf den Nutzdaten arbeitet, und das auch noch als Software, so daß viel Ressourcen verbraucht werden.

Für realzeitorientierte Videokommunikation sind mehrere Kanäle mit verschiedener Dienstgüte nötig. Diese kann das Internet zum jetzigen Zeitpunkt noch nicht bieten. Eine Mischung aus „Wiederholung auf Anfrage“ und inhaltsorientierter Vorwärtsfehlerkorrektur kann diese Kanäle verschiedener Dienstgüten applikationsseitig simulieren. Kosten solcher Verfahren ist größerer Verbrauch an lokalen Ressourcen und erhöhter Bandbreitenbedarf. Erfolgreich arbeiten diese Techniken, wenn sie zur Laufzeit auf Indikatoren über die subjektiv erreichte Qualität und aufgetretene Fehler reagieren können. Maßnahmen zum dynamischen Wechsel des Kommunikationssystems oder zur Rekonfiguration der Wegewahl (*rerouting*) werden in heutiger Umgebung als organisatorisch und technisch zu aufwendig angesehen. Bei so einem Schritt würden die aufgebauten Indikatoren wertlos sein und das bis dahin nur ungenaue Wissen über das Fehlerprofil wäre nutzlos. Funkstrecken sind intern im Kommunikationssystem gegen starke Fehler zu sichern, z.B. durch aktive Netzwerkkarten.

5.2 Sitzungsmanagement

Das Sitzungsmanagement hat die Aufgabe, beim Auftreten von Fehlern für sanften Qualitätsabfall (*graceful degradation*) zu sorgen. Hier wird zwischen geringen und größeren Verlusten unterschieden. Geringe Verluste sind Verluste, die geringe Datenmengen und/oder einen geringen Teil der Bildfläche betreffen und von einzelnen Teilen eines Blockes bis zu einer Menge von Blöcken reichen. Größere Verluste reichen vom Verlust einzelner Bildteile bis zum Verlust von einem oder mehreren Bildern. Die gewählte Paketisierung der Bildobjekte hat maßgeblichen Einfluß darauf, ob häufiger geringe oder größere Verluste auftreten.

Wenn geringe Verluste verstreut auftreten, d. h. räumlich oder zeitlich benachbarte Daten des gleichen Types intakt sind, so verspricht die Wiederholung voriger Daten oder modellbasierte Schätzung mit einfachen statistischen Methoden beim Empfänger eine gute Fehlerkorrektur, weil die Verfahren schnell eine Fülleinheit für das verlorene Bildobjekt erzeugen und diese Fülleinheit dem verlorenen Objekt so hinreichend ähnlich ist, daß die Präsentationsqualität vermutlich nicht beeinträchtigt wird. Mustersuchen, also die Suche eines benachbarten Bildobjektes, daß die erwarteten Eigenschaften möglichst gut erfüllt, könnte schnell genug arbeiten, wenn der Dekoder, der verwendet wird, eine effiziente Implementierung der Differenzenkodierung hat. Das wird vermutet, weil das Mustersuchen ähnlich arbeitet, wie die Differenzenkodierung.

Empfängerbasierte Techniken arbeiten auf kleineren Bildobjekten (wie z.B. einzelnen Koeffizienten) besser als auf ganzen Blöcken. Darum wird vorgeschlagen, die oben angegebenen empfängerbasierten Verfahren in Kombination mit Verstreuung (*interleaving*) oder Stromaufteilung (*information dispersal*)

anzuwenden, weil diese Techniken sicherstellen, daß geringe, verstreute Verluste häufiger auftreten als zusammenhängendere (und damit größere) Verluste. Inhaltsorientierte Vorwärtsfehlerkorrektur kann beim Sender eingesetzt werden, um sicherzustellen, daß weniger wichtige Daten häufiger von Verlusten betroffen sind als wichtige. Es wird aber vermutet, daß diese Technik nicht schnell genug arbeitet.

Für größere Verluste werden empfängerseitige Fehlertoleranztechniken alleine vermutlich versagen, weil nicht genügend Ressourcen (insbesondere Zeit und intakte benachbarte Bildobjekte) zur Verfügung stehen, um eine Fülleinheit zu erzeugen, die den verlorenen Bildobjekten so ähnlich ist, daß die Präsentationsqualität erhalten bleibt. Die oben empfohlenen empfängerbasierten Techniken könnten weiterhin eingesetzt werden, um schnell eine Fülleinheit für verlorene Objekte zu erzeugen, neben diesen kurzfristigen Korrekturmaßnahmen sind empfängerseitig aber zusätzliche Maßnahmen einzusetzen, um mittel- bzw. langfristig die Präsentationsqualität wieder zu erhöhen.

Dafür wird eine Rückmeldung an den Sender über einen sicheren Feedback-Kanal (z.B. TCP-basiert) als Fehlertoleranztechnik vorgeschlagen. Diese Rückmeldung kann entweder Informationen über beobachtete Fehler enthalten (wie von RTP vorgeschlagen) oder konkrete Anweisungen an den Sender enthalten. Ziel ist dabei die Rekonfiguration von senderbasierten Fehlertoleranztechniken, die dort in der Kodierung und Paketisierung eingesetzt werden. Wird die Kodierung geändert, so wird Skalierung zur Fehlerkorrektur eingesetzt. Durch Skalierung können Auflösung, Bildrate und Bildgröße sowie Quantisierung angepaßt werden. Wenn kurze Übertragungszeiten beobachtet wurden, kann Skalierung auch im kurzfristigen Bereich angewendet werden, z.B. ein Wechsel des Referenzbildes bei Interkodierung nach H.263+. Empfängerseitig kann auch Zeitstreckung angewendet werden, wenn ganze Bilder verloren sind, um den Eindruck der Kontinuität zu erhalten.

Die Paketisierung beim Sender hat maßgeblichen Einfluß darauf, welche Fehler beim Empfänger beobachtet werden können. Wird beim Sender ein oder mehrere Bilder oder ein oder mehrere Bildteile in einem Paket verschickt, so kann der Empfänger die im vorigen Abschnitt diskutierten geringen Verluste nie beobachten, weil der Verlust eines Paketes den kleinstmöglichen Verlust auf der Verarbeitungsebene darstellt. In diesem Fall kann die Fehlertoleranzstrategie nur wie beim Auftreten von größeren Verlusten beschrieben reagieren. Darum wird erwartet daß Fehlerkorrektur bessere Ergebnisse zeigt, wenn ein oder mehrere Blöcke oder ein oder mehrere Blockteile zusammen paketisiert werden. Dann treten geringe Verluste vermutlich sehr viel häufiger auf als größere; empfängerbasierte Fehlertoleranztechniken können die meisten Verluste ohne auffällig geringere Präsentationsqualität ausgleichen, und nur im Einzelfall werden Netz und Sender durch die Reaktion auf größere Verluste zusätzlich belastet. Obwohl also kleine Pakete aus Sicht der Fehlerkorrektur eindeutig beim Systementwurf zu bevorzugen sind, so wird man sich in der Praxis vermutlich für „mittelgroße“ Pakete entscheiden, um die insgesamt lokal vorhandenen Ressourcen (Bandbreite, Prozessorkapazität, Arbeitsspeicher) nicht zu stark durch Erzeugung und Verarbeitung einer Vielzahl von Paketköpfen zu belasten.

Neben der sequentiellen Paketisierung, bei der die Bildobjekte in der Reihenfolge paketisiert werden, in der der Kodierer sie erzeugt, ist auch eine verstreute Paketisierung (senderbasierte Fehlertoleranztechnik *interleaving*) denkbar. Dabei kann nach zwei verschiedenen Gesichtspunkte gestreut werden: Entweder um zu verhindern, daß bei einem Paketverlust räumlich oder zeitlich benachbarte Bildobjekte verloren gehen, oder um als wichtiger eingestufte Daten (z.B. der

Kodiermodus und hohe Koeffizienten) von weniger wichtigen getrennt zu Paketisieren (geschichtete Kodierung, *priority encoded transmission -PET*). Der letzte Ansatz macht im Internet nur Sinn, wenn anschließend inhaltorientierte Vorwärtsfehlerkorrektur auf die Pakete angewendet wird. Eine Sitzung könnte so ablaufen, daß ein Teilnehmer zunächst einstellt, ob der Bild- oder der Tonstrom höher priorisiert übertragen werden soll und wie auf Bandbreitenschwankungen reagiert werden soll (sollen zuerst die Farbdaten nicht mehr übertragen werden oder soll die Quantisierung geändert werden etc.). Dann beginnt die Sitzung mit minimaler Dienstqualität bis lokal Indikatoren über das Fehlerprofil statistisch stabil vorliegen. Dieses wird dann an den Sender signalisiert, der dann nach und nach unsichere Kanäle mit zusätzlichen Informationen belegt. Werden die Indikatoren statistisch instabil, werden zuerst die Zusatzströme abgeschaltet. Das Ganze ist ein systemtheoretischer Regelkreis mit dem Problem, daß durch äußere Einflüsse jederzeit instabile Zustände auftreten können. Der Vorteil ist, daß sich nach einer gewissen Dauer der Sitzung eine „optimale“ Dienstqualität einstellen wird. Für diesen Ansatz wird vermutet, daß er zu viele Ressourcen belegt, um realzeitorientiert zu guten Ergebnisse zu führen.

Wenn keine Möglichkeit zur geschichteten Kodierung besteht, könnte eine Sitzung so ablaufen, daß der Teilnehmer bei Sitzungsbeginn zwar Prioritäten für bestimmte Dienstmerkmale festlegt, diesen im Verlauf der Sitzung aber nur bedingt gefolgt wird. Zunächst würde der Sender beginnen, mit minimaler Dienstgüte zu senden. Eventuell würde bereits einfaches Verstreuen angewendet werden, um empfängerseitige Fehlerkorrektur zu erleichtern. Der Empfänger wendet von Anfang an Wiederholung voriger Daten und eine einfache modellbasierte Schätzung zum Generieren von Fülleinheiten an. Verluste und eine „angenommene Güte der Schätzung“ würden für jedes Bildobjekt des vorigen Bildes beim Empfänger gespeichert. Wenn eine zeitlang nur geringe Verluste beobachtet wurden, würde der Empfänger eine Rückmeldung an den Sender geben, mit dem Ziel, daß dieser mehr Daten durch veränderte Skalierung sendet. Dieses würde solange geschehen, bis die Verluste gerade noch durch empfängerbasierte Techniken behandelt werden können, und auf diese Weise eine „optimale“ Dienstqualität erreicht ist. Eine schematische Übersicht über den Ablauf einer Sitzung beim Empfänger mit einer Fehlertoleranzstrategie für Videokommunikation zeigt Abbildung 23.

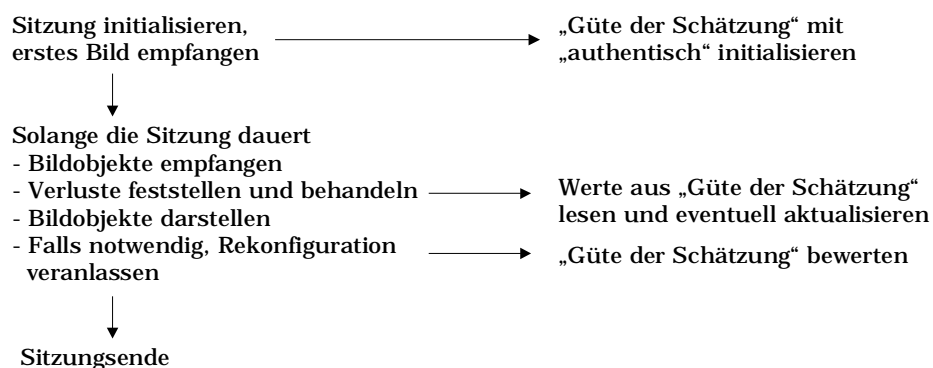


Abbildung 23: schematischer Ablauf einer Sitzung beim Empfänger

Wie ein Entscheidungsablauf vor einer einfachen modellbasierten Schätzung auf Makroblockebene ablaufen könnte, zeigt Abbildung 24. Eine „Schätzung auf Basis eines Blockes“ könnte so aussehen, daß mit Hilfe der Werte der räumlich umgebenden Blöcke eine Korrektur für die Werte des Basisblockes errechnet würde. Die entstandene Schätzung kann als „gut“ gespeichert werden, wenn sie auf „genügend“ authentischen Daten fußt. Im Gegensatz dazu errechnet eine

„Schätzung ohne Basis“ einen Füllblock ohne Daten aus dem zeitlich vorigen Bild zu verwenden. Sie fußt alleine auf den Werten der räumlich umgebenden Blöcke. Darum wird vermutet, daß die Qualität dieser Schätzung stets schlecht ist, und darum wird die resultierende Schätzung als „schlecht“ gespeichert. Die zusätzliche Speicherung dieser „Güte der Schätzung“ soll vermeiden helfen, daß aufwendige

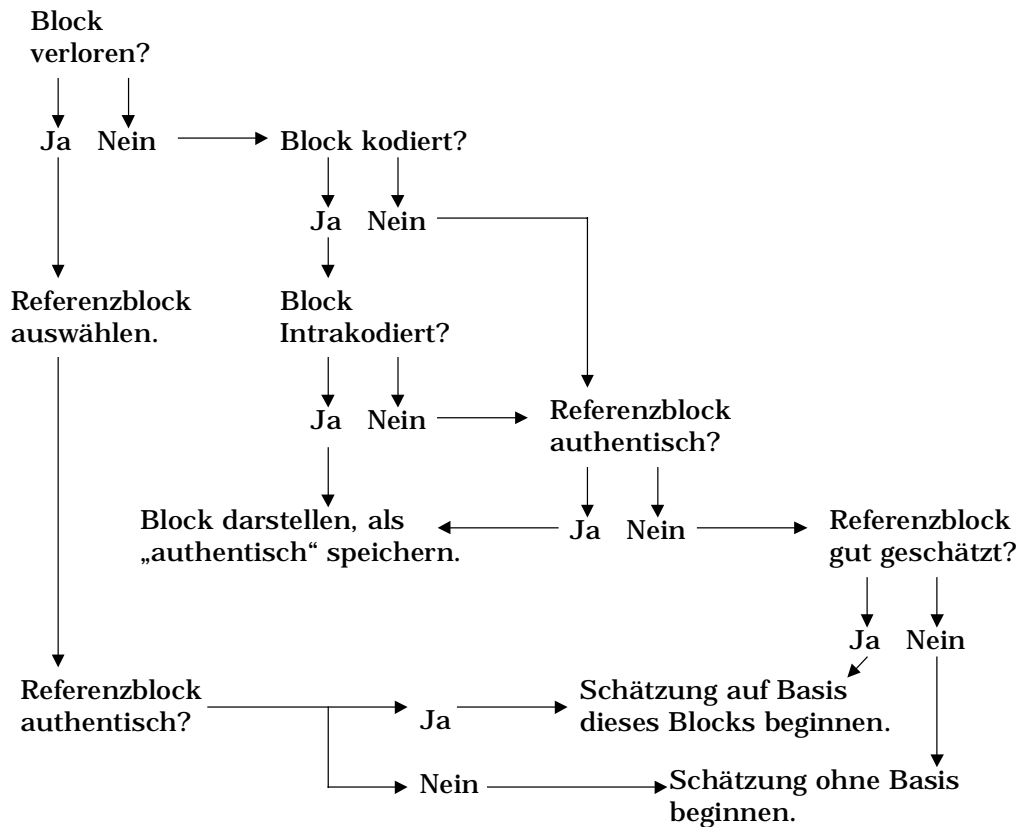


Abbildung 24: Entscheidungsablauf vor einfacher Schätzung

Berechnungen auf wenig authentischen Daten ausgeführt werden. Damit soll Fehlerausbreitung in den Schätzungen vermieden werden. Diese „Güte der Schätzung“ liefert für jedes Bild eine Aussage über die Authentizität jedes Makroblockes und kann eine Grundlage sein für die Entscheidung, wann eine Rückmeldung an den Sender angestoßen werden soll, um die Kodierung zu rekonfigurieren.

6 Zusammenfassung und Ausblick

Die erfolgreiche Entwicklung einer Applikation zur realzeitorientierten Videokommunikation hängt von der Festlegung und Einbettung einer Fehlertoleranzstrategie ab. Eine Fehlertoleranzstrategie bestimmt, welche Fehlertoleranztechniken verwendet werden und wie sie im Fehlerfall angewendet werden. Eine Fehlertoleranztechnik ist eine Methode zur Verbesserung der Dienstgüte. Senderbasierte Techniken verwenden ein Modell über das Auftreten von Fehlern, um Fehler zu behandeln. Empfängerbasierte Techniken verwenden ein Modell über das Verhalten der Teilnehmer. Sender- und empfängerbasierte Methoden können in einer Strategie vereint werden. Anwenderorientierte Dienstgüte bedeutet in Videokonferenzen die Unterstützung der Rolle, die ein Teilnehmer gerade einnimmt. Informationen über die Rolle sind wertvoll für die Fehlertoleranzstrategie.

In dieser Arbeit wurde zunächst ein System zur Unterstützung von realzeitorientierter Videokommunikation schematisch vorgestellt und herausgearbeitet, welche Aspekte von Fehlertoleranz betroffen sind. Dann wurde ein Überblick gegeben über verschiedene Fehlertoleranztechniken und die Verwendbarkeit dieser Techniken für die Übertragung von Bildfolgen untersucht. Die Kombinationsmöglichkeiten einzelner Techniken wurden untersucht und ein Ansatz zur prototypischen Umsetzung skizziert. Die Ergebnisse sind in Tabelle 1 zusammengefasst. Für jede Technik ist dabei eingetragen, wie sie sich zum Einsatz bei bestimmten Verlusten eignet. Die Bewertung stützt sich auf die subjektive Einschätzung des Gewinnes an Darstellungsqualität im Vergleich zum Ressourcenverbrauch.

Eignung einzelner Techniken bei beobachteten Verlusten		ein oder mehrere Bilder	Bildteile	Blöcke	Blockteile
Rückmeldung an den Sender	4.3.1	sehr	etwas		
Interleaving	4.3.2		etwas	etwas	sehr
Dispersal	4.3.3		etwas	etwas	sehr
Skalierung	4.3.4	sehr	sehr		
bitorientierte FEC	4.3.5				
inhaltsorientierte FEC	4.3.6			etwas	sehr
Spleißen	4.4.1				
Einfügen von Stille	4.4.2				
Wiederholung voriger Daten	4.4.3	etwas	etwas	sehr	sehr
Mustersuchen	4.4.4		etwas	etwas	
Zeitstreckung	4.4.5	etwas			
Rückgriff auf Zustände	4.4.6				
modellbasierte Schätzung	4.4.7	etwas	etwas	sehr	sehr

Tabelle 1: Eignung von Fehlertoleranztechniken bei Bildfehlern

Die Verwendung kleinerer Pakete ermöglicht eine bessere Beobachtung tatsächlich aufgetretener Übertragungsfehler, beansprucht bei gleichen Nutzdaten aber mehr Bandbreite, weil mehr Header übertragen werden müssen. Welcher Kompromiss aus Bandbreite und Informationen über Fehler geschlossen wird, hängt von der verwendeten Netztechnologie ab. In Tabelle 2 sind verschiedene Fehlertoleranzstrategien in Abhängigkeit von der gewählten Art der Paketisierung eingetragen.

Generell scheint eine Kombination von nicht zu aufwendigem Interleaving und Skalierung heute die beste Möglichkeit zum senderbasierten Schutz der Daten zu sein. Empfängerseitig hat sich Wiederholung voriger Daten und das Bilden von

Schätzern aufgrund von einfachen statistischen Modellen bewährt. Es wird empfohlen, auf beobachtete Verluste durch ein adaptives Management innerhalb der Applikation zu reagieren. Dazu sollten Rückmeldungen des Empfängers an den Sender möglich sein. Der Teilnehmer sollte an der Festlegung der Fehlertoleranzstrategie über Einstellmöglichkeiten beteiligt werden.

ein Paket enthält	Strategie bei großen Verlusten (Bildteile, Bilder)	Strategie bei kleinen Verlusten (Blöcke, Blockteile)
ein oder mehrere Bilder	Rückmeldung an den Sender, Skalierung, eventuell Wiederholung voriger Daten oder einfache modellbasierte Schätzung	-
ein oder mehrere Bildteile	Rückmeldung an den Sender, Skalierung, einfache modellbasierte Schätzung oder Wiederholung voriger Daten, eventuell Interleaving	-
ein oder mehrere Blöcke	Rückmeldung an den Sender, Skalierung und/oder Interleaving	Wiederholung voriger Daten und/oder modellbasierte Schätzung, eventuell Mustersuchen
ein oder mehrere Blockteile	Rückmeldung an den Sender, Skalierung, Interleaving, eventuell inhaltsorientierte FEC	modellbasierte Schätzung, Wiederholung voriger Daten, Mustersuchen

Tabelle 2: Fehlertoleranzstrategien in Abhängigkeit von der Paketisierung

Heute gängige Protokolle und Netztechnologien sind noch nicht auf Applikationen mit umfangreichen, strömenden Daten wie realzeitorientierte Videokommunikation ausgelegt. Durch Design- und Konfigurationsschwächen weiten sich kleine Verluste bei der Übertragung zu Bildverlusten im Dekoder aus. Schlecht ausgeprägte Optionen zum Management machen zudem nur sehr begrenzt Reaktionen auf die Fehler zur Laufzeit möglich. Fehlertoleranztechniken können darum nur sehr begrenzt eingesetzt werden und erzielen kaum befriedigende Resultate, weil eine übergreifende Fehlertoleranzstrategie durch die heutige Technologie behindert wird.

Die Bildverarbeitung ist heute aber schon so weit, daß Bildtelefonie über best-effort-Netze in greifbare Nähe gerückt ist, wie neuere Standards zeigen. Wenn verbesserte Managementmöglichkeiten durch neue Protokolle (namentlich IPv6) einhergehen mit verbesserter Übertragungstechnik, wird es sicherlich bald Bildtelefonie über Internet geben. In realzeitorientierten Anwendungen wird es immer Restfehler geben, die nicht behandelt werden können. Diese sollen den Teilnehmer möglichst wenig stören. Neuere Standards wie H.263+ und MPEG-4 stellen nicht wie ältere Kompression über Fehlerrobustheit, sondern betonen mehr Designaspekte, die Fehlerausbreitung durch Restfehler verringern. Mit neuen Protokolle wie z.B. RTP und IP-Multicast ist das Internet weiterentwickelt worden in Richtung besserer Unterstützung von Videokonferenzen. Die Umsetzung einer Fehlertoleranzstrategie wird von diesen Technologien besser ermöglicht. Ein idealer Kodierer produziert drei Ströme von Blöcken (YUV) sowie Steuerinformationen und hat einen Feedback-Eingang um auf Fehlermeldungen zu reagieren. Außerhalb der Codecs steuern Fehlertoleranzstrategien die zukünftige Kodierung und stellen verlorene Blöcke wieder her. Innerhalb des Codecs sind verschiedene Verschleierungsoptionen vorzusehen, die der Benutzer über Parameter steuert.

Für die Zukunft bleibt es, existierende Codecs darauf zu prüfen, ob sie wirklich Ströme produzieren, oder ob durch syntaktische Abhängigkeiten (relative

Bildadressierung etc.) nicht vielmehr komplexe Objekte entstehen, bei denen einzelne Bitfehler zu starker Fehlerfortpflanzung im Dekodierer führen, so daß jede Fehlertoleranztechnik nutzlos ist. Eine weitere Aufgabe ist es, ein applikationsübergreifendes adaptives Netz- und Sitzungsmanagement auf best-effort-Basis (ITU-Philosophie) aufzubauen. Das Management der Rate kann durch Regelkreise mit Sperren erfolgen, um instabile Zustände zu verhindern. Die Sendungen sind auch bei RTP an den im Netz herrschenden Verkehrszustand anzupassen (wie bei TCP). Außerdem ist die Wirkung verschiedener Skalierungsmaßnahmen (wie Verlust von Farbe oder Verlust von jedem zweiten Bild) auf die Bitraten und die Bildqualität zu untersuchen. Neuere Standards wie IPv6 und MPEG-4 sind zu untersuchen in bezug auf ihre Möglichkeiten, realzeitorientierte Videokommunikation zu unterstützen.

Eine weiterführende Arbeit könnte sich mit dem prototypischen Aufbau einer Fehlertoleranzstrategie für Bildtelefonie über Internet befassen, indem der Empfänger beim Sender durch Rückmeldung Parameter zur Paketisierung und Kodierung manipulieren kann, um auffällige Verluste zu vermeiden und um kleine Verluste empfängerseitig gut verschleiern zu können. Es werden vier Maßnahmen vorgeschlagen: Beim Sender sollte verstreut paketisiert werden, so daß bei einem Paketverlust nicht direkt benachbarte Makroblöcke verloren gehen. Beim Empfänger sollten verlorene Blöcke zunächst als Verlust markiert werden und gleichzeitig die vorigen Blöcke dieser Stelle als erste Schätzung in den Abspielpuffer eingestellt werden. Anschließend wird mehrschrittig eine „als besser angenommene“ Schätzung des verlorenen Blockes berechnet durch modellbasierte Schätzung, besonders Interpolation umgebener, authentischer Daten, bis eine als „hinreichend gut“ eingestufte Schätzung existiert oder bis der Präsentationszeitpunkt des Blockes kurz bevorsteht. Wenn die Berechnung des Schätzers „zu häufig“ abgebrochen werden mußte, weil der Präsentationszeitpunkt gekommen war, ist eine Rekonfiguration der Kodierung durch eine Rückmeldung an den Sender anzustoßen. Möglichkeiten zur Ausweitung dieser Strategie auf Videokonferenzen können dabei diskutiert werden. Die dafür notwendigen Annahmen über Anwendungsverhalten und Netzverhalten könnten anschließend stichprobenartig durch Einsatz dieses fehlertoleranten Codec geprüft werden.

Literatur

- [ASW 1998] A. Albanese, S. Siemsglueß, B. Wolfinger: Information Dispersal to improve Quality-of-Service on the Internet. Proc. SPIE Intern. Symp. on Voice, Video and Data Communications, Boston/Mass, Vol. 3529 (1998)
- [BAN 1999] D. Bull, N. Canagarajah, A. Nix (Eds.): Insights into Mobile Multimedia Communications. Academic Press (1999) 682p.
- [BCG 1995] J. Bolot, H. Crépin, A. Vega-Garcia: Analysis of Audio Packet Loss in the Internet; Proc. 5th. Intl. Workshop on Network and Operating System Support for Digital Audio and Video, Durham, New Hampshire (1995) pp.163-174
- [Boc 1996] P. Bocker: ISDN. Springer-Verlag (1996) 4. Aufl., 332 S.
- [Bra 1995] R. Brabender: Multimediale Kommunikation im Schmal- und Breitband-ISDN. GMD-Studien Nr. 273 (1995) 125 S.
- [Bro 1998] E. Brown: Guide to Videoconferencing Systems. Newmedia-Magazin, Dec. 1998 (1998), www.newmedia.com
- [CaB 1997] G. Carle, E. W. Biersack: Survey of Error Recovery Techniques for IP-Based Audio-Visual Multicast Applications. IEEE Network, Vol. 11, No. 6 (1997) pp.24-36
- [Cea 1998] G. Côté et al.: H.263+: Video Coding at Low Bit Rates. IEEE Transactions on Circuits and Systems for Video Technology, Vol. 8 No. 7 (1998) pp849-866
- [DüM 1999] M. Dürst, M. Mühlhäuser: Datenkompression. In: [ReP 1999], S.239-254
- [Ech 1990] K. Echtle: Fehlertoleranzverfahren. Springer Verlag Berlin (1990) 322S.
- [FGV 1998] N. Färber, B. Girod, J. Villasenor: Extensions of ITU-T Recommendation H.324 for Error-Resilient Transmission. IEEE Communications, Vol. 36, No. 5 (1998), pp. 120-128
- [Fro 1997] K. Froitzheim: Multimedia Kommunikation. dpunkt-Verlag (1997) 363 S.
- [Gea 1998] J.D. Gibson et al.: Digital Compression for Multimedia. Morgan Kaufmann Publishers Inc. (1998) 476p.
- [GFS 1999] B. Girod, N. Färber, E. Steinbach: Error Resilient Coding for H.263. In: [BAN 1999], Chap. 28, (1999), pp. 445-459
- [Gör 1989] W. Görke: Fehlertolerante Rechensysteme. Oldenbourg Verlag München (1989) 207S.
- [Hea 1998] J.-F. Huard et al.: Realizing the MPEG-4 Multimedia Delivery Framework. IEEE Network, Vol. 12, No. 6 (1998) pp.35-45

- [Hei 1999] K. Heidtmann: Leistungs- und Zuverlässigkeitsaspekte von Videokommunikation mit Echtzeitanforderungen. 2. WAKI-IIA-Symposium über verteilte multimedia Anwendungen und diensteintegrierte Kommunikationsnetze. Flensburg, Sep. 1999 (1999), S. 111-125
- [HKZ 1999] K. Heidtmann, C. Kohlhaas, M. Zaddach: Messung der Netzlast und Bewertung der Bildqualität bei Videokommunikation über Paketvermittlungsnetze. ARCS 99: 15. GI/ITG-Fachtagung über Architektur von Rechensystemen. Jena, Okt. 1999 (1999), S. 237-248
- [Ise 1996] M. Isenburg: Transmission of multimedia data over lossy networks. Report, ICSI Univ. Calif./Berk., (1996), 107p.
- [ITU 1993] ITU-T Telecommunication Standardization Sector of ITU: Video Codec for Audiovisual Services at p×64 kbits. ITU-T Rec. H.261 (1993) 25p.
- [ITU 1997] ITU-T: Telecommunication Standardization Sector of ITU: Video Coding for low bitrate Communication. ITU-T Rec. H.263 Version 2 (H.263+) draft-version (1995)
- [Kam 1998] G. Kamosa: Video Coding and Robust Transmission in Error-Prone Environments. Masters-Thesis, Univ. of Calif. (1998), 69p.
- [Käp 1997] T. Käppner: Entwicklung verteilter Multimedia-Applikationen. Vieweg-Verlag (1997) 282p.
- [Koh 1999] C. Kohlhaas: Untersuchung der von Videokodierern erzeugten Verkehrslasten und der durch das Verhalten von Kommunikationssystemen beeinflussten Bildqualität. Universität Hamburg. Diplomarbeit. (1999)
- [LAP 1999] X. Li, M. H. Ammar, S. Paul: Video Multicast over the Internet. IEEE Network, Vol. 13, No. 1 (1999) pp.46-60
- [Pax 1999] V. Paxson: End-to-End Internet Packet Dynamics. IEEE/ACM Transactions on Networking, Vol. 7 No. 3 (1999) pp277-292
- [Pea 1997] C. Perkins et al.: RTP Payload for Redundant Audio Data; IETF-NWG RFC 2198 (proposed Std., 1997) 11p.
- [PeH 1998] C. Perkins, O. Hodson: Options for Repair of Streaming Media; IETF-NWG RFC 2354 (Informational, 1998) 12p.
- [PHH 1998] C. Perkins, O. Hodson, V. Hardman: A Survey of Packet Loss Recovery Techniques for Streaming Audio. IEEE Network, Vol. 12, No. 5 (1998), pp.40-48
- [Pro 1998] W. E. Proebster: Rechnernetze. Oldenbourg-Verlag (1998) 432 S.
- [ReP 1997] P. Rechenberg, G. Pomberger (Hrsg.): Informatik-Handbuch. Hanser-Verlag (1997) 961 S.

- [ReP 1999] P. Rechenberg, G. Pomberger (Hrsg.): Informatik-Handbuch. 2. Aufl. Hanser-Verlag (1999) 1166 S.
- [RoS 1999] J. Rosenberg, H. Schulzrinne: An RTP Payload Format for Generic Forward Error Correction; IETF-AVT draft.ietf-avt-fec-07 (1999) 22p.
- [RRO 1998] D. Reininger, D. Raychaudhuri, M. Ott: A Dynamic Quality of Service Framework for Video in Broadband Networks. IEEE Network, Vol. 12, No. 6 (1998) pp. 22-34
- [Sch 1996] H. Schulzrinne: RTP Profile for Audio and Video Conference with Minimal Control; IETF-AVT RFC 1890 (proposed Std., 1996) 9p.
- [Sea 1996] H. Schulzrinne et al.: RTP: A Transport Protocol for Real-Time Applications; IETF-AVT RFC 1889 (proposed Std., 1996) 38p.
- [SEK 1999] A.H. Sadka, F. Eryurtlu, A.M. Kondoz: Aspects of Error Resilience for Blockbased Video Coders in Multimedia Communications. In: [BAN 1999], Chap. 27, (1999), pp. 431-443
- [ShH 1998] K. G. Shin, S. Han: Fast Low-Cost Failure Recovery for Reliable Real-Time Multimedia Communication; IEEE Network Vol. 12 No. 6 (1998), pp.56-63
- [Ste 1998] W. R. Stevens: TCP/IP Illustrated, Vol. 1: The Protocols. Addison Wesley Inc., 12th Printing (1998) 576p.
- [Tal 1998] R. Talluri: Error-Resilient Video Coding in the ISO MPEG-4 Standard. IEEE Communications, Vol. 36, No. 5 (1998), pp. 112-119
- [Tal 1999] R. Talluri: Algorithms for Low Bit Rate Video Coding. In: [BAN 1999], Chap. 19, (1999), pp. 301-318
- [Tan 1997] A.S. Tanenbaum: Computernetzwerke. Prentice Hall Verlag, 3. Aufl. (1997) 847S.
- [Tra 1999] B. Traoré: Zur Transformation multimedialer Verkehrslasten in diensteintegrierten Kommunikationssystemen. Universität Hamburg. Diplomarbeit (1999)
- [Ver 1998] O. Verscheure et. al.: User-Oriented QoS in Packet Video Delivery. IEEE Network, Vol. 12, No. 6 (1998) pp.12-21
- [WaZ 1998] Y. Wang, Q. Zhu: Error Control and Concealment for Video Communication: A Review; Proc. of the IEEE, Vol. 86, No. 5 (1998), pp.974-997
- [WeC 1999] S. Wenger, G. Coté: Using RFC2429 and H.263+ at low medium bit-rates for low-latency applications. In: Proceedings of the Packet Video Workshop, NYC, USA (1999)
- [WSC 1996] M. H. Willebeek-LeMair, Z.-Y. Shae, Y.-C. Chang: Robust H.263 Video Coding for Transmission over the Internet. Report EECS Dept. Univ. Calif./Berk. (1996) 25p.

Anhang A

Auswahl einiger Internetstandards (RFCs) und deren Status

RFC 0768	UDP	J. Postel: User Datagram Protocol (1980)	Standard
RFC 0791	IP	J. Postel: Internet Protocol (1981)	Standard
RFC 0792	ICMP	J. Postel: Internet Control Message Protocol. (1981)	Standard
RFC 0793	TCP	J. Postel: Transmission Control Protocol (1981)	Standard
RFC 1035	DNS	P.V. Mockapetris: Domain Names – Implementation and Specification (1987)	Standard
RFC 1112	MBONE	S. E. Deering: Host extensions for IP-multicasting (1989)	Standard
RFC 1889	RTP	H. Schulzrinne et al: RTP: A Transport Protocol for Real-Time Applications. (1996)	proposed Standard
RFC 1890	Miniconf	H. Schulzrinne: RTP Profile for Audio and Video Conferences with Minimal Control. (1996)	proposed Standard
RFC 2131	DHCP	R. Droms: Dynamic Host Configuration Protocol (1997)	draft Standard
RFC 2205	RSVP	R. Braden et al: Resource ReSerVation Protocol (RSVP). (1997)	proposed Standard
RFC 2327	SDP	M. Handley, V. Jacobson: SDP: Session Description Protocol. (1998)	proposed Standard
RFC 2354	Repair	C. Perkins, O. Hodson: Options for Repair of Streaming Media. (1998)	informational
RFC 2460	IPv6	S. Deering, R. Hinden: Internet Protocol, Version 6 (IPv6) Specification (1998)	draft Standard
RFC 2463	ICMPv6	A. Conta, S. Deering: Internet Control Message Protocol (ICMPv6) for IPv6. (1998)	draft Standard
RFC 2543	SIP	M. Handley et al: SIP: Session Initiation Protocol (1999)	proposed Standard

Anhang B

RTP-Nutzlasten (RTP-payload) und deren Status im Nov. 1999

RFC 2029	Sun's CellB Video	M. Speer: RTP Payload Format of Sun's CellB Video Encoding. (1996)	proposed Standard
RFC 2032	H.261	T. Turetti, C. Huitema: RTP Payload Format for H.261 Video Streams. (1996)	proposed Standard
RFC 2190	H.263	C. Zhu: RTP Payload Format for H.263 Video Streams	proposed Standard
RFC 2198	Redundant Audio	C. Perkins et al: RTP Payload for Redundant Audio Data. (1997)	proposed Standard
RFC 2250	MPEG1/ MPEG2	D. Hoffmann et al: RTP Payload Format for MPEG1/MPEG2 Video. (1998)	proposed Standard
RFC 2429	H.263+	C. Bormann et al: RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+). (1998)	proposed Standard
RFC 2431	BT.656	D. Tynan: RTP Payload Format for B.656 Video Encoding. (1998)	proposed Standard
RFC 2435	JPEG-Video	L. Berc et al: RTP Payload Format for JPEG- compressed Video. (1998)	proposed Standard
RFC 2658	PureVoice(tm)	K. McKay: RTP Payload Format for PureVoice(tm) Audio. (1999)	proposed Standard