SpecialHost, ein Programm zum Auswerten von special-Befehlen für den Amiga

Version 1.18

vom
14. Juli 1994

von
Georg Heßmann & Olaf Barthel

Abstract

SpecialHost
ist ein Hilfsprogramm für den Previewer ShowDVI und den Druckertreiber DVIprint. Es wertet die special-Befehle aus, die die Treiber dem Programm übergeben. Es ist in der Lage, IFF-ILBM-Bilder, jegliche über die `datatypes.library` ladbare Bilder und PostScript-Dokumente in den TEX-Text einzubinden, sowie TPIC Specials auszuwerten.

Inhaltsverzeichnis:

1. Copyright und ähnliches

Das Programm SpecialHost ist "freely distributable copyrighted software". Das heißt, jeder darf es sich kopieren und verwenden, das Copyright bleibt aber vollständig bei uns (Georg Heßmann & Olaf Barthel).

Das Programm darf auch nur unverändert weitergegeben werden.

Das Programm darf ohne unsere ausdrückliche Genehmigung nicht kommerziell verkauft werden. Als PD-Software vertrieben darf der Preis 5,- DM pro Diskette nicht überschreiten.

Es darf auch nur in vollständiger und unveränderter Form weitergegeben werden.

Diese Anleitung "`spec.tex`" ist ebenfalls Copyright Georg Heßmann & Olaf Barthel und darf nicht kommerziell vervielfältigt werden. Man darf sie sich ausdrucken und auch kopieren, man darf sie aber **nicht** für Geld weitergeben.

Für weitere Erklärungen siehe das `README`-Dokument.

Einzelne Dithering-Matrizen werden mit Erlaubnis von Wolf Faust verwendet.

2. Allgemeines über SpecialHost

Jeder, der auch nur ein wenig mit TEX gearbeitet hat, kennt dessen gravierendsten Nachteil: die fehlende Möglichkeit, Graphiken zu erstellen und in den Text einbinden zu können. Es gibt zwar Ansätze, dies TEX beizubringen (in LaTEX oder PICTEX), aber sie sind zum Scheitern verurteilt, wenn die Graphiken auch nur etwas komplizierter werden.

Nur mit einem Trick kann man diese Einschränkung umgehen: die Verwendung des \special-Befehls. Glücklicherweise wurde im DVI-Dateiformat eine Möglichkeit vorgesehen, auch nicht Standardisiertes in die DVI-Datei aufzunehmen. Dazu dient der \special-Befehl. Alles was in geschweiften Klammern diesem Befehl folgt, wird unverändert in die DVI-Datei übernommen. So können treiberabhängige Anweisungen am TEX-Programm vorbei in die DVI-Datei gelangen. Der jeweilige Treiber kann den \special-Befehl auswerten und zum Beispiel eine Graphik in die aktuelle Seite einbinden.

Der Nachteil dieser Methode soll aber nicht verschwiegen werden. Sobald man mit \special-Befehlen arbeitet, ist die DVI-Datei nicht mehr portabel. Das heißt, man kann diese Datei nicht mehr auf beliebige Rechner übertragen und erwarten, daß die Ausgabe dort genau so aussieht, wie auf dem eigenen Rechner. So gravierend ist diese Einschränkung allerdings auch nicht, da im schlimmsten Fall die \special-Befehle einfach ignoriert werden und der Text eben ohne Graphik ausgedruckt wird.

Wie arbeitet SpecialHost?
SpecialHost nutzt die Möglichkeit des Multitasking und des Message-Transfers des Amiga aus, um auf möglichst flexible Art und Weise die \special-Befehle auszuwerten. SpecialHost wird parallel zu dem jeweiligen Treiber (ShowDVI oder DVIprint) gestartet und wartet auf eine hereinkommende Meldung des Treibers.

Trifft der Treiber bei der Auswertung des DVI-Files auf einen \special-Befehl, so übergibt er diesen unverändert dem SpecialHost-Programm zusammen mit ein paar weiteren Informationen, wie z.B. der derzeitigen Auflösung des Bildes.

Die Aufgabe des SpecialHost-Programms ist es, die Message des Treibers abzuholen und den \special-Befehl auszuwerten. Wird in dem Befehl der Name eines Bildes angegeben, so lädt das Programm das angegebene Bild und paßt es der eingestellten Auflösung an (verkleinert oder vergrößert es). Danach übergibt es die so erzeugte Bitmap dem Treiber. Dieser muß sie nur noch in seine eigene Bitmap, in der er eine ganze TEX-Seite hält, einkopieren.

Warum aber ein eigenes Programm für die Auswertung des \special-Befehls?
Diese Lösung ist meiner Meinung nach erheblich flexibler, als die gesamte Auswertung des \special-Befehls in jeden Treiber einzubauen. Dadurch, daß eine fest definierte Schnittstelle zwischen den Treibern und dem SpecialHost-Programm existiert, kann man die Programme alle unabhängig voneinander weiterentwickeln, ohne auf größere Schwierigkeiten zu stoßen.

So kann man zum Beispiel das Format des \special-Befehls umdefinieren oder neue Fähigkeiten in das SpecialHost-Programm einbauen, wie z.B. die Auswertung von Vektorgraphiken, ohne auch nur eine Winzigkeit in den Treibern ändern zu müßen. Oder falls ich etwas in den Treibern ändere und jemand sich ein eigenes SpecialHost-Programm geschrieben hat, sollten keine Probleme bei der Zusammenarbeit dieser Programme auftreten.

3. Der Aufruf von SpecialHost

SpecialHost läßt sowohl von der Shell als auch von der Workbench aus aufrufen. Beim Aufruf können dem Programm entweder über Piktogramm-Merkmale (Icon-Tooltypes) oder Befehlszeilenargumente folgende Parameter übergeben werden:

`TRANSFER` SpecialHost kann ShowDVI und DVIprint auf verschiedene Arten ein Bild übermitteln, bzw. statt eines Bildes nur einen Platzhalter übertragen. Mit dem Schlüsselwort `TRANSFER` muß einer der Parameter `FILE` (= das Bild wird in einer Datei über den Umweg über die Festplatte, Diskette, etc. übertragen), `MEMORY` (= das Bild wird im Speicher des Amiga aufgebaut und übertragen) oder `NONE` (= es wird kein Bild, sondern stattdessen ein Platzhalter in Form eines leeren Bildes übertragen). Als Beispiel: `TRANSFER=MEMORY`

`RENDER` Hiermit läßt sich bestimmen, ob das Programm Bilder unverändert an ShowDVI und DVIprint weitergeben, oder ob ein Rahmen um sie herum gezeichnet werden, bzw. eine schwarze Fläche gezeichnet werden soll. Zum Schlüsselwort `RENDER` muß einer der Parameter `FRAME` (= ein Rahmen wird um das Bild gezeichnet), `CLEAR` (= eine schwarze Fläche wird statt des Bildes gezeichnet) oder `PLAIN` (= das Bild wird nicht verändert) angegeben werden. Als Beispiel: `RENDER=PLAIN`

`BASEDPI` Sofern eine Bilddatei keine Informationen bereitstellt, in welcher Auflösung es gezeichnet wurde, wird auf den hier angegebene Parameter

zurückgegriffen. Näheres zu seiner Funktion ist unter der Beschreibung der Bedienungselemente der Benutzeroberfläche nachzulesen. Als Beispiel: `BASEDPI=100`

`INVERT` Dieser Parameter bewirkt, daß jedes Bild nach der Umrechnung invertier wird (Weiß wird Schwarz und umgekehrt).

`USESCREEN` Wird dieser Parameter angegeben, versucht SpecialHost sein Ausgabefenster auf dem Bildschirm von ShowDVI zu öffnen. Ist dies jedoch nicht zu machen, wird das Fenster auf dem Workbench-Bildschirm geöffnet.

`NOGUI` Üblicherweise verwendet SpecialHost eine ausgefeilte Benutzeroberfläche, die allerdings nur dann zur Verfügung steht, wenn die benötigten Komponenten des MUI-Pakets installiert sind. Ist dies nicht der Fall, wird auf eine vereinfachte Oberfläche zurückgeschaltet, in der das Programm lediglich seine Meldungen ins Ausgabefenster schickt und durch Drücken der Tastenkombination `Control+C` beendet werden kann.

SpecialHost liest **erst** die Konfigurationsdatei und wertet **danach** die über Kommandozeile/Tooltypes übergebenen Parameter aus.

4. Der Aufbau des Fensters von SpecialHost

Hinweis: Die folgende Beschreibung gilt nur für den Fall, daß die Benutzeroberfläche **nicht** über den Schalter `NOGUI` abgeschaltet wurde.

Das Fenster von SpecialHost ist in drei Teile unterteilt:

Der obere Teil dient der Eingabe. Dort sind alle Kontrollfelder zu finden, mit denen das Programm gesteuert wird.

Der mittlere Teil ist als Kontrollfenster für die Aktivitäten des Programms gedacht. Dort wird mitprotokolliert, was das Programm gerade macht. Unter der Liste befindet sich eine Füllanzeige, mit der das Fortschreiten bestimmter Arbeitsschritte dokumentiert wird.

Der untere Teil enthält Funktionstasten, mit denen einzelne Programmfunktionen

direkt gesteuert werden können, die nicht die Bearbeitung von \special-Befehlen betreffen.

5. Die Bedienung von SpecialHost

Das Programm wird vollständig über Gadgets gesteuert. Dies sind:

`Transfer` SpecialHost kann ShowDVI und DVIprint auf verschiedene Arten ein Bild übermitteln, bzw. statt eines Bildes nur einen Platzhalter übertragen. Der Schalter `Transfer` kann auf die folgenden Werte gestellt werden:

- `Image in memory` Das Bild wird im Speicher des Amiga aufgebaut und direkt an ShowDVI, bzw. DVIprint übertragen. Dies hat zum einen den Vorteil, daß nach dem Umrechnen des Bildes kein Zugriff auf Festplatte oder Diskette mehr stattfindet, aber auch den Nachteil, daß ein jedes Bild beim erneuten Anzeigen noch einmal neu gelesen und umgerechnet werden muß, selbst wenn sich seit dem letzten Anzeigen weder Bild noch Auflösung geändert haben.
- `Image on disk` Ein einmal umgerechnetes Bild wird in eine Datei geschrieben für den Fall, daß es noch einmal gebraucht wird. Auf diese Weise muß ein Bild nicht jedesmal neu umgerechnet werden, wenn es angezeigt werden soll, aber sich weder Bild noch Auflösung geändert haben.
- `No image` Statt eines Bildes wird ein Platzhalter übergeben, der einem leeren Bild entspricht. Der Platzhalter hat dieselbe Größe wie das Bild, das nicht gezeigt wird.

`Rendering` Hier läßt sich einstellen, ob das an ShowDVI oder DVIprint zu schickende Bild verändert werden soll:

- `No modification` Das Bild wird nicht verändert.
- `Draw a frame` Das Bild wird mit einem dünnen Rahmen umgeben.
- `Clear image` Das Bild wird durch eine schwarze Fläche ersetzt.

`Invert` Ist dieser Schalter aktiv, so wird jedes Bild vor der Übertragung an ShowDVI oder DVIprint invertiert (Schwarz wird Weiß und umgekehrt).

`Base DPI` Sofern eine Bilddatei (PostScript-Dokumente sind hiervon nicht betroffen) keine Informationen über die Auflösung des Bildes enthält, wird stattdessen der hier angegebene Wert verwendet.

`Jump to ShowDVI` Drückt man auf diesen Knopf, wird versucht, das Ausgabefenster auf den Bildschirm des ShowDVI-Programmes zu verlegen. Ist dies nicht zu machen, wird das Fenster auf dem Workbench-Bildschirm geöffnet.

`ShowDVI to front` Wird dieser Schalter betätigt, wird das ShowDVI-Programm – sofern es gerade läuft – in den Vordergrund gebracht und aktiviert.

`Clear list` Dieser Schalter löscht die darüber befindliche Liste von Nachrichten.

6. Das Menü

Bestimmte Funktionen von SpecialHost können auch über ein Menü aufgerufen werden.

`About...` Gibt ein paar Zeilen Copyright-Text aus.

`Load settings` Liest das Konfigurationsfile erneut ein. Setzt also alles wieder in den Urzustand zurück.

`Save settings` Speichert die derzeitige Konfiguration in eine Datei ab.

`Use public screen` Ist diese Funktion aktiv, versucht SpecialHost das Ausgabefenster auf dem Bildschirm von ShowDVI zu öffnen.

`Quit` Beendet das Programm.

7. Das Format des special-Befehls

Damit die TEX-Treiber ShowDVI und DVIprint das Programm SpecialHost nach einzubindenden Bildern abfragen, müssen diese wissen, nach was sie fragen sollen. Dies steht im TEX-Text, die Anweisung ist der \special-Befehl. Alles was bei

\special{beliebiger Text}

dem \special-Befehl zwischen den geschweiften Klammern steht, wird vom TEX-Programm ignoriert und unverändert zum Treiber durchgereicht. Trifft nun ShowDVI oder DVIprint auf solch einen Befehl, so wird überprüft, ob ein SpecialHost-Programm im Hintergrund läuft. Ist dies nicht der Fall, so wird der \special-Befehl einfach ignoriert. Existiert das SpecialHost-Programm, so bekommt dieses den vollständigen Befehl per Nachricht übermittelt.

Es ist also das SpecialHost-Programm, welches den Special-Befehl auswertet und die Bedeutung der einzelnen Wörter des Befehls festlegt. Da dieser Special-Befehl in keiner Weise standardisiert ist, kann sich das Format von Treiber zu Treiber ändern.

Das vorgestellte Format gilt also nur für diese Kollektion von Treibern und ist auch hier noch nicht endgültig festgelegt. In neueren Versionen werden sicherlich noch

Funktionen hinzukommen. Es wird aber auf Abwärtskompatibilität Wert gelegt, so daß man ohne Probleme auf neuere Versionen umsteigen kann.

Das derzeitige Format der \special-Befehle:

```
\special{anw1 anw2 anw3}
```

Die einzelnen Anweisungen innerhalb der Klammern werden mit ein oder mehreren Leerschritte voneinander getrennt.

Eine Anweisung wiederum besteht aus einem Schlüsselwort und einem Wert, so daß eine genauere Betrachtung des \special-Befehle folgendes Format hervorbringt:

```
\special{key1=val1 key2=val2 key3=val3}
```

Zwischen Schlüsselwort (key) und Wert (value) muß immer ein Gleichheitszeichen stehen.

Als Schlüsselwörter gibt es bisher:

`ifffile`	Definiert den Dateinamen eines zu ladenden Bildes.
`psfile`	Definiert den Dateinamen eines zu ladenden PostScript-Dokumentes.
`psinitfile`	Dient zum Ändern des Namens der Datei, die zum Initialisieren des PostScript-Interpreters verwendet wird. Standardmäßig wird `TeX:ps/init.ps` verwendet.
`psinitstring`	Ist ein bis zu 256 Zeichen langer PostScript-Befehl, der nach dem Aufruf der Initialisierungsdatei des PostScript-Interpreters ausgeführt wird.
`hoffset`	Gibt den horizontalen Versatz des Bildes an.
`voffset`	Gibt den vertikalen Versatz des Bildes an.
`hsize`	Legt die horizontale Größe des Bildes fest,
`vsize`	Legt die vertikale Größe des Bildes fest.
`scale`	Legt den Faktor fest, um den das Bild in der Größe verändert werden soll.
`hscale`	Legt den horizontalen Vergrößerungs-/Verkleinerungsfaktor fest.
`vscale`	Legt den vertikalen Vergrößerungs-/Verkleinerungsfaktor fest.
`mode`	Legt die Methode fest, mit der das Bild in Schwarz/Weiß umgerechnet werden soll.
`threshold`	Gibt den prozentualen Schwellwert für das Schattierungsverfahren “Blue noise” an. Gebräuchliche Werte bewegen sich zwischen 30% und 50%.
`red`	Legt den Rot-Anteil des Farbbildes fest. Damit kann der Rot-Anteil des Bildes verstärkt oder vermindert werden.
`green`	Legt den Grün-Anteil des Farbbildes fest.
`blue`	Legt den Blau-Anteil des Farbbildes fest.
`bright`	Definiert die Helligkeit für das Farbbild.
`contrast`	Definiert den Kontrast für das Farbbild.
`gamma`	Definiert die “Gamma-Korrektur” für das Farbbild.
`transfer`	Bestimmt, wie ein Bild übertragen werden soll. `transfer=memory` überträgt es im Speicher des Amiga, `transfer=file` überträgt

es im Speicher des Amiga, greift aber im Bedarfsfall auf ein auf Diskette/Festplatte zwischengespeichertes Bild zurück und `transfer=none` überträgt statt des Bildes einen Platzhalter. Wird der `transfer` Parameter nicht angegeben, werden die aktuellen Programmeinstellungen verwendet.

`render` Legt fest, ob das Bild so wie es ist, übertragen wird, oder ob es auf eine bestimmte Art und Weise verändert werden soll. `render=plain` verändert das Bild nicht, `render=frame` zeichnet einen dünnen Rahmen um das Bild und `render=clear` zeichnet statt des Bildes eine schwarze Fläche. Wird der `render` Parameter nicht angegeben, werden die aktuellen Programmeinstellungen verwendet.

`basedpi` Sofern eine Bilddatei keine Informationen bereitstellt, in welcher Auflösung es gezeichnet wurde, wird auf den hier angegebenen Parameter zurückgegriffen (z.B. `basedpi=100`). Fehlt der Parameter `basedpi`, werden die aktuellen Programmeinstellungen verwendet.

`invert` Mit diesem Schalter läßt sich das Bild vor der Übertragung noch invertieren. Mit `invert=on` wird die Invertierung aktiviert, mit `invert=off` wird sie abgeschaltet. Fehlt der Parameter `invert`, werden die aktuellen Programmeinstellungen verwendet.

`patchcolours` Bilder, die mit den alten IFF-Routinen (1985-1987) geschrieben wurden, wie z.B. von *DeluxePaint I-III* enthalten keine absolut korrekten Farbinformationen. So hat die Farbe Weiß statt des RGB-Tripels `FF:FF:FF` den Wert `F0:F0:F0`. Aus der weißen Farbe wird so ein blasses Hellgrau. Hierdurch können krasse Fehler beim Umrechnen der Farbwerte entstehen. Der Parameter `patchcolours` sorgt dafür, daß eine solche defekte Farbpalette im Bedarfsfall angepaßt wird.

`ditheropt` Eine Zahl, die die Größe der von den Schattierungsroutinen zu benutzende Matrix beeinflußt. Gegenwärtig kann dies entweder 4 oder 8 sein. Je größer der Wert, desto kleiner die Matrix. Dies hat seinen Sinn, wenn Matrizen, die große Strukturen im Bild erzeugen, auf kleine Bilder angewendet werden, wobei viele Bildinformationen verlorengehen können. Reduziert man die Größe der Matrix, so schränkt dies zwar die Anzahl der darstellbaren Farbschattierungen ein, aber es werden nicht ganz soviele Bilddetails verschluckt. Bisher unterstützen nur die folgenden Schattierungsverfahren die Definition der Matrizengröße: `fwdbrick`, `bckbrick`, `halftone` und `spiraldot`.

Bei `ifffile` und `psfile` ist als Wert der Name einer Datei anzugeben. Dazu ist zu beachten, daß die jeweils aktuelle Schublade jene ist, in welcher das SpecialHost-Programm gestartet wurde. Aus diesem Grund sollte jeder Dateiname immer den **kompletten** Pfadnamen enthalten.

Über die beim Programm `dvips` mitgelieferten Formatdateien `epsf.sty` und

`epsf.tex` eingebundenen EPSF-Dateien werden von SpecialHost direkt gelesen und bearbeitet.

Das Programm lädt alle bekannten IFF-ILBM-Formate (2..256 Farben, HAM, HAM8, 8 Bit Graustufen, 12..24 Bit Echtfarben). Steht die `datatypes.library` zur Verfügung, die Bestandteil der Workbench 3.0 ist, werden Bilder, die nicht im IFF-ILBM-Format vorliegen, auf diesem Weg gelesen. Dies setzt natürlich voraus, daß ein jeweils passendes Lademodul zur Verfügung steht. Zum Zeitpunkt der Erstellung dieses Dokuments existierten frei verfügbare Lademodule für die Formate `GIF`, `JPEG`, `PCX`, `MacPaint` und `Windows Bitmap`.

Statt realer Dateien kann auch die Zwischenablage (clipboard) ausgelesen werden (Hinweis: die zu lesenden Daten müssen dabei zwingend im IFF-ILBM-Format vorliegen). Hierzu muß man als Parameter `iffile=clipboard:<Nummer>` angeben. Die Nummer gibt an, welcher der 256 verschiedenen Kanäle der Zwischenablage ausgelesen werden soll (z.B. `clipboard:1`). Die in spitzen Klammern stehende Nummer kann entfallen, es wird dann Kanal Nr. 0 ausgelesen, der standardmäßig vom Betriebssystem benutzt wird.

Damit ein über das `psfile`-Schlüsselwort angegebenes PostScript-Dokument gelesen und ausgewertet werden kann, muß die `post.library` im System installiert sein. Bevor das angegebene Dokument gelesen wird, ruft SpecialHost im Normallfall die Datei `TeX:ps/init.ps` auf, die zur Initialisierung des PostScript-Treibers gedacht ist (über den Special-Befehl kann der Name der aufzurufenden Datei geändert werden). Sie dient u.a. dazu, die Pfade zum Suchen der PostScript-Fonts einzustellen und sollte vom Anwender vor dem ersten Probelauf konfiguriert werden. Nähere Informationen hierzu sind der Dokumentation des “Post”-Paketes zu entnehmen.

Bei den Schlüsselwörtern `hoffset`, `voffset`, `hsize` und `vsize` wird als Wert eine Dimensionsangabe verlangt. Diese besteht aus einer reellen Zahl plus einer Einheitsbezeichnung. Folgende Einheiten stehen zur Verfügung:

`pt`	point (wird von TEX verwendet)
`pc`	pica $1pc = 12pt$
`in`	inch $1in = 72.27pt$
`bp`	big point $72bp = 1in$
`cm`	centimeter
`mm`	millimeter
`dd`	didôt point $1157dd = 1238pt$
`cc`	cicero $1cc = 12dd$

Die Parameter der Schlüsselwörter `vscale`, `hscale` und `scale` legen den Faktor fest, um den das Bild in der Größe verändert werden soll. Der Wert 0.5 entspricht eine Halbierung der Größe, der Wert 2.0 eine Verdoppelung der Größe.

Bei der Umrechnung eines Bildes in Schwarz/Weiß müssen die Farbwerte jedes Punktes nach einem bestimmten Verfahren in Graustufen umgerechnet werden. Dies geschieht, indem die Rot-, Grün- und Blaukomponenten eines jeden Farbwertes

mit bestimmten Faktoren multipliziert, addiert und die Summe durch 100 geteilt wird. Die Faktoren lassen sich über die Schlüsselwörter `red`, `green` und `blue` einstellen. Standardmäßig werden die Parameter `red=30 green=59 blue=11` verwendet, was die Farbkomponenten ihren Helligkeitswerten entsprechend in die Rechnung einfließen läßt. Die Summe der Faktoren darf 100 nicht überschreiten, im Bedarfsfall werden die Werte entsprechend angepaßt.

Die Schlüsselwörter `bright`, `contrast` und `gamma` bewirken eine Veränderung des Bildes. Die Parameter können sich im Bereich -100..100 bewegen, der Wert 0 läßt das Bild unverändert. Während `bright` und `contrast` Helligkeit und Kontrast des Bildes jeweils linear verändern, wirkt sich der `gamma`-Parameter nicht-linear auf beide Eigenschaften aus. Ein Gamma-Wert größer als 0 hellt das Bild auf, ein Wert kleiner als 0 dunkelt es ab.

Dem Schlüsselwort `mode` muß einer der Parameter `bw`, `fs`, `burkes`, `sierra`, `jjn`, `stevenson`, `stucki`, `bluenoise`, `ordered`, `random`, `halftone`, `bckbrick`, `fwdbrick`, `hexagon`, `spiraldot` oder `horizontal` folgen. Standardmäßig wird `mode=fs` verwendet. Der jeweilige Parameter gibt an, mit welchem Schattierungsverfahren ein Bild in Schwarz/Weiß-Darstellung umgerechnet werden soll:

`bw` Es wird ein simples Schwellwertverfahren verwendet. Liegt ein Helligkeitswert unter der Schwelle, erscheint er als schwarzer Punkt, liegt er über der Schwelle, erscheint er als weißer Punkt.
Beispiel:

`fs` Das Verfahren von R.W. Floyd und L. Steinberg (4 gestreute Fehlerwerte, serpentinenartige Verteilung) wird verwendet, das sehr feine, chaotische Muster erzeugt. Dies ist das klassische Fehlerstreuungsverfahren.
Beispiel:

`burkes` Das Verfahren von D. Burkes (7 gestreute Fehlerwerte, serpentinenartige Verteilung) wird verwendet. Auch dies ist ein Fehlerstreuungsverfahren, das die Fehler aber im Gegensatz zum Floyd-Steinberg Verfahren auf wesentlich mehr Punkte verteilt. Das Resultat ist eine gleichmäßigere Streuung der Punkte.
Beispiel:

`sierra` Das Verfahren von F. Sierra (10 gestreute Fehlerwerte, serpentinenartige Verteilung) wird verwendet.
Beispiel:

`jjn` Das Verfahren von J.F. Jarvis, C.N. Judice und W.H. Ninke (12 gestreute Fehlerwerte, serpentinenartige Verteilung) wird verwendet. Dieses Verfahren hebt – was ein unerwünschter Nebeneffekt sein kann – die Schärfe des jeweiligen Bildes.
Beispiel:

`stucki` Das Verfahren von P. Stucki (12 gestreute Fehlerwerte, serpentinenartige Verteilung) wird verwendet. Dieses Verfahren ähnelt dem von Jarvis, Judice & Ninke, jedoch wird die Bildschärfe nicht erhöht.
Beispiel:

`bluenoise` R. Ulichney's Verfahren "Blue noise" (4 gestreute Fehlerwerte, serpentinenartige Verteilung) wird verwendet. Im Zusammenhang mit diesem Verfahren wird ein Schwellwert-Parameter verlangt (anzugeben über das Schlüsselwort `threshold`). Fehlt dieser Parameter, entspricht dieses Verfahren in seiner Wirkung dem vom Floyd und Steinberg.
Beispiel:

`stevenson` Das Verfahren von R.L. Stevenson und G.R. Arce (12 versetzte gestreute Fehlerwerte, serpentinenartige Verteilung) wird verwendet.
Beispiel:

`random` Ein Zufallsmuster wird über die Bilddaten gelegt, die dadurch sehr rauh wirken. DeluxePaint verwendet dieses Verfahren zum Mischen von Farben.
Beispiel:

`ordered` B.E. Bayer's Verfahren wird verwendet. Hierbei wird ein schachbrettartiges Muster über die Bilddaten gelegt. Das entstehende Bild erhält eine sehr regelmäßige Maserung.
Beispiel:

`halftone` Ein sehr verbreitetes Verfahren aus der Druckindustrie, das z.B. zum

Drucken von Fotos in Zeitungen oder beim Siebdruck eingesetzt wird. In niedrigen Auflösungen wirkt es sehr grob, es hat aber auch seine Vorteile: will man Fotokopien eines Ausdrucks machen, kippen Bilder, die mit den Verfahren `fs`, `random` und `ordered` erzeugt wurden, aufgrund der feinen Strukturen, die die Algorithmen erzeugen, sehr leicht in zu helle oder zu dunkle Bilder um. Diesen Effekt kann man bei `halftone` nicht beobachten. Dieses Verfahren erzeugt kreisförmige Strukturen im sich ergebenden Bild.
Beispiel:

`fwdbrick` Dieses Verfahren erzeugt diagonale Linienstrukturen im sich ergebenden Bild.
Beispiel:

`bckbrick` Dieses Verfahren erzeugt diagonale Linienstrukturen im sich ergebenden Bild (ähnlich wie `fdwbrick`, nur gespiegelt).
Beispiel:

`hexagon` Dieses Verfahren erzeugt rautenförmige Strukturen im sich ergebenden Bild.
Beispiel:

`spiraldot` Dieses Verfahren erzeugt quadratische Strukturen im sich ergebenden Bild.
Beispiel:

`horizontal` Dieses Verfahren erzeugt horizontale Linienstrukturen im sich ergebenden Bild.
Beispiel:

Die maximale Bildgröße ist auf 32.768 × 32.768 Punkte beschränkt, die Seitengröße eines PostScript-Dokuments sogar auf 30.000 × 30.000 Punkte. Überschreitet ein Bild diese Grenzen, wird gnadenlos abgeschnitten.

Hier ein paar Beispiele für einen \special-Befehl:
• `\special{ifffile=Bridge.pic hsize=8cm vsize=5cm mode=jjn}`

Hier wird die IFF-ILBM-Datei “Bridge.pic” an der aktuellen Textposition in der Größe 8 × 5 cm eingebunden, zum Umrechnen in Schwarz/Weiß wird das Verfahren von Jarvis, Judice & Ninke benutzt:

- `\special{ifffile=Bridge.pic hsize=8cm vsize=5cm hoffset=1.5cm mode=halftone}`

  Diesmal wird das Bild um 1,5 Zentimeter nach rechts verschoben und das ‘Halftone’-Schattierungsverfahren benutzt:

- `\special{ifffile=AmigaWorld.pic hsize=8cm vsize=5cm mode=bluenoise threshold=50}`

  Hier wird die IFF-ILBM-Datei “AmigaWorld.pic” an der aktuellen Textposition in der Größe 8 × 5 cm eingebunden. Zum Umrechnen in Schwarz/Weiß wird das Verfahren “Blue noise” mit einem Schwellwert von 50% benutzt:

- `\special{psfile=golfer.eps hsize=2.5in vsize=4.0in scale=0.33 render=frame}`

  Das PostScript-Dokument "golfer.eps" wird auf 1/3 verkleinert in einem 2,5 × 4,0 Zoll großen Bereich angezeigt, um das Bild wird ein Rahmen gezeichnet:

Bei der Verwendung der Schlüsselwörter `hsize` und `vsize` muß übrigens nicht mehr auf die richtige Einstellung der "Base DPI" geachtet werden, da dann das Bild auf jeden Fall auf die geforderte Größe gebracht wird. Die Orginalgröße des Bildes wird dann ignoriert.

Mit Ausnahme der TPIC Specials, die im Anschluß erläutert werden, muß derzeit immer ein Dateiname angegeben werden.

8. TPIC Unterstützung

SpecialHost versteht auch die Befehle der TPIC Sprache. In dieser Sprache gibt es Kommandos zum Zeichnen von Linien, Kreisen, Splines usw. Die Sprache TPIC ist genormt und wird auch von einigen anderen Programmen unterstützt. Zum Beipiel von dem Makropaket EEPIC, das Bilder im TPIC Format erzeugt.

Die TPIC Specials werden etwas anders verarbeitet, als die bisher besprochenen Special-Befehle. Die TPIC Kommandos werden von den Treibern vorbereitet und SpecialHost bekommt nur noch komplette TPIC Kommandos übergeben. Dadurch wird etwas Kommunikation zwischen den Programmen eingespart.

9. FIG-Bilder in TEX-Texte

Das "FIG" Programm ist ein vektororientiertes Graphicprogramm, mit dem man Zeichnungen erstellen kann und im "fig" Format abspeichern kann. Diese Zeichnungen kann man nun auch auf sehr einfache Art und Weise in einen TEX-Text einbinden. Dazu muß man lediglich ein paar Schritte ausführen.

Als erstes wird das Bild mit dem `FIG` Programm zeichnen und abspeichern. Z.B. `pict.fig`. Nun muß man das Bild in ein anderes Format konvertieren. Dazu wird das Programm `Fig2Dev` verwendet. Der Aufruf:

```
Fig2Dev >pict.eepic -L eepic pict.fig
```

Dies konvertiert das Programm in das "EEPIC" Format. EEPIC ist ein Makropacket das zusammen mit LATEX verwendet werden kann. Nun schreibt man ein kleines TEX-Dokument (z.B. `pict.tex`):

```
\documentstyle[11pt,epic,eepic]{article}
\begin{document}
\input pict.eepic
\end{document}
```

Dieses Dokument muß nun mit LATEX übersetzt werden. Also z.B. mit:

```
virtex &lplain pict
```

Die entstehende DVI-Datei kann man wie üblich mit ShowDVI ansehen und mit DVIprint ausdrucken. Allerdings muß dabei natürlich SpecialHost im Hintergrund laufen, da das EEPIC Format TPIC Specials verwendet.

10. Das optimale Schattierungsverfahren

Wenn ein Programm schon eine derartig große Menge von Schattierungsverfahren zur Verfügung stellt, liegt die Frage nahe, welches Verfahren mit welchen

Parametern zu einem optimalen Ergebnis führt. Hierzu läßt sich keine allgemein gültige Antwort finden, es gibt allerdings eine Reihe von Regeln, an denen man sich orientieren kann.

Zunächst sollte man diese Dokumentation einmal auf dem Drucker ausgeben lassen, den man später verwenden möchte und sich die ausgedruckten Bilder anschauen. Die Schwächen und Vorteile der einzelnen Verfahren zeigen sich meist schon beim ersten Ausdruck. Im Gegensatz zur Darstellung am Computerbildschirm überlappen sich die einzelnen Punkte auf dem bedruckten Papier. Insbesondere Verfahren wie `fs`, `jjn` und `ordered` versagen leicht bei diesen Bedingungen, sie führen zumeist nur dann zu guten Ergebnissen, wenn die einzelnen Bildpunkte diskret angeordnet sind. Ist dies nicht der Fall, sollten beim Ausdruck eher Verfahren wie `halftone` oder `hexagon` verwendet werden. Sollen später Fotokopien der Ausdrucke gemacht werden, ist wiederum in der Mehrzahl der Fälle ein Verfahren wie `halftone` oder `hexagon` im Vorteil, da der Kopiervorgang üblicherweise diskret angeordnete Bildpunkte zusammenfaßt und verwischt.

11. Die Software-Schnittstelle zu den Treibern

Hier ein kurzer Einblick in die Arbeitsweise des SpecialHost Programms.

11.1. Die Kommunikation über Ports

Die Kommunikation der Treiber mit dem SpecialHost-Programm läuft mittels eines Message-Ports ab. Sobald SpecialHost gestartet wird, öffnet es einen Port mit dem Namen "`special_dvi`". An diesem Port wartet das Programm auf Meldungen der Treiber.

Wenn ein Treiber in der DVI-Datei auf einen \special-Befehl stößt, testet er zuerst, ob der Port des SpecialHost-Programms existiert. Wenn nicht, ignoriert er den Befehl. Falls doch, so erzeugt der Treiber einen Port mit dem Namen "`special_reply`" und richtet ihn als Reply-Port ein.

Über den Port "`special_dvi`" schickt er dem SpecialHost-Programm eine Meldung, in der die derzeitige Auflösung des Druckers (oder Bildschirms) sowie der \special-Befehl steht.

SpecialHost wertet diese Message aus und fügt die `special_map` Struktur hinzu. Diese enthält die Größe des Bildes, den Offset und auch den Namen des Bitmap-Files oder einen Pointer auf die Bitmap. Diese Meldung schickt es per "ReplyMsg" an den Treiber zurück.

Sobald der Treiber alle Informationen der "special_map" ausgewertet hat, schickt er dem SpecialHost-Programm noch eine Meldung, damit dieses weiß, daß der Speicherplatz der "special_map" wieder freigegeben werden kann. Diese Meldung wird von SpecialHost sofort bestätigt.

Danach löscht der Treiber seinen Reply-Port.

Während diese Kommunikation abläuft – zwischen dem Versenden der ersten Meldung durch dem Treiber und dem Reply der zweiten Message durch das

SpecialHost Programm – dürfen die beiden Partner der Unterhaltung auf keinen Fall unterbrochen werden. Sobald einer der beiden seine Kommunikation einstellt, entsteht ein Deadlock. Das andere Programm wartet bis zum nächsten Reset. Da kein Timeout implementiert ist, hilft in dem Fall nur ein Reboot weiter. Wenn man das SpecialHost Programm ändert, dann muß man darauf achten, daß während der Kommunikation das Programm nie abgebrochen werden kann. Auch bei einem Fehlerfall muß die Kommunikation immer zu Ende geführt werden.

11.2. Das Bitmap-Dateiformat

Um nicht immer wieder das zeitaufwendige Vergrößern oder Verkleinern der Bilder durchführen zu müssen, gibt es die Möglichkeit, die schon auf die richtige Größe gebrachten Bilder abzuspeichern.

Damit man die Bilder immer in den verschiedenen Auflösungen parat hat, werden die Dateien beim Abspeichern in einer Schublade untergebracht, deren Name von der der Auflösung des Druckers, bzw. des Bildschirms abhängt. Speichert man beim Previewer mit einer Auflösung von 91dpi das Bild `testbild` ab, so wird die Datei unter `SpecialHost_91x91/testbild` erzeugt. Existiert noch keine Schublade unter dem Namen `SpecialHost_91x91`, so wird sie angelegt.

Diese Datei hat ein ganz besonderes Format.
Die ersten vier Bytes des Files sind das 'Magic Word' "SPEC". An diesem erkennt man, daß es eine Bitmap-Datei des SpecialHost Programms ist.

Die nächsten vier Bytes ergeben sich, wenn man über die einzelnen Zeichen des Special-Befehls, der zu diesem Bild gehört, eine CRC-32 Prüfsumme zieht. Dies hat den Sinn, daß man daran erkennt, ob sich der Special-Befehl geändert hat, seitdem man die Datei erzeugt hat. Ist dies der Fall, so wird das Bild neu berechnet und abgespeichert.

Nun folgen zwei Langwörter, die die Breite und Höhe des Bildes in Pixel angeben.

Zum Schluß kommt die Bitmap, Zeile für Zeile. Die Länge einer Zeile wird bis auf 16 Bit aufgerundet. Für eine Zeile mit 100 Bit benötigt man also sieben 16-Bit-Wörter.

Noch ein weiteres Datum wird indirekt mit der Datei gespeichert. Das Erstellungsdatum. Es wird dazu verwendet zu überprüfen, ob das Ursprungsbild jüngeren Datums ist als die Bitmap-Datei. Ist dies der Fall, so muß natürlich die Bitmap neu berechnet werden, da sich das Bild allem Anschein nach geändert hat.

Durch diesen Datumsvergleich und durch das Überprüfen des \special-Befehls wird sichergestellt, daß keine veraltete Bitmap-Datei verwendet wird. Man kann also ohne weiteres den Zeichenmodus `Transfer picture in a file` als Standard verwenden. So muß ein Bild für eine Auflösung nur einmal berechnet werden, und steht dann immer als Datei zur Verfügung. Das einzige, daß dabei zum Problem werden kann, ist der Speicherplatz, da die Bitmap-Dateien ungepackt abgespeichert werden.

12. Benötigte Soft- und Hardware

Zur Hardware ist nur zu sagen: ein Amiga, und zwar nach Möglichkeit einer mit etwas mehr Speicher. Sonst kann es bei großen Bildern und hoher Auflösung zu Speichermangel kommen.

Als Software werden die TEX-Treiber ShowDVI und DVIprint benötigt. Um PostScript-Dokumente einbinden zu können, wird die `post.library` (mindestens Version 1.5) benötigt.

13. Bekannte Fehler

Bisher sind (leider?) keine Programmfehler bekannt.